CuteFTP Professional 6
Transfer Engine
User's Guide

by GlobalSCAPE

Table Of Contents

Transfer Engine (SDK)

Transfer Engine Overview

About the Transfer Engine

What is the Transfer Engine?

Built on a modular design platform, CuteFTP 6 Professional's FTP **Transfer Engine (TE)** is completely independent of the main application's interface. You can control the TE through an industry standard COM (Component Object Model) interface using your favorite programming or scripting language, such as Visual Basic, Perl, ASP or JavaScript.

Unique to CuteFTP's TE are powerful encryption properties that dictate how the control and data channels communicate with the FTP server. From SSL to S/key or SSH2, the TE covers all your secure file transfer needs.

Benefits

The Transfer Engine:

- Is COM enabled
- Takes up minimal resources
- Handles background transfers
- Handles transfers outside the main interface
- Handles continuous folder synchronization events
- Can be used with a custom application or script to perform FTP related tasks.

Capabilities

The TE exposes most of the popular commands previously accessibly only through the application's main interface. Some of the actions you can accomplish using the TE:

- Login to an FTP server over a defined port
- Login using an encrypted authentication mechanism, such as OTP
- Login and transfer files using industry standard SSL/TLS (explicit & implicit modes supported)
- Login and transfer files using SSH2
- Login to sites through Firewalls, Socks or Proxy servers
- Transfer files one at a time or simultaneously
- Transfer files using multi-part transfers to increase throughput
- Transfer files from one remote site to another
- Transfer files using a wildcard mask
- Transfer files obtained from server-side links
- Perform folder synchronization tasks
- Check on an item's transfer status
- Much more

Transfer Engine (TE) initial setup

As a typical COM Component, the TE Object must be registered as such on the target system. The TE will automatically perform COM Registration if it has been manually executed (run once) prior to a script or application instantiating the TE Object.

To register the TE COM component

1. Launch CuteFTP Professional
2. Close CuteFTP Professional. It should now be registered.

or

1. Locate the CuteFTP 6 Professional program folder.
2. Double-click on the file named "ftpte.exe".
3. Exit the TE by right-clicking on the TE icon in your Systray.

Note

- In order to run script files, you must have the Windows Scripting Host (WSH). WSH relies on the Visual Basic Script and JavaScript engines provided with Internet Explorer 3.0 or later. WSH is also installed as a part of Windows 98, Windows 2000, and Internet Information Server 4.0.

Running scripts

You can execute a script you have created directly in Windows, from the command line, from within CuteFTP's interface, or even as a scheduled task to be run with no user logged in.

To execute a script directly in windows

1. Launch CuteFTP Professional
2. From the menu bar, click **Tools > Macros & Scripting > Run**
3. **Browse** to the script file you created (example, sample.vbs)
4. Click **Open** to launch the script

Because you are in the CuteFTP interface, you will see the log and queue populate with session and transfer information.

To execute a script from the command line

1. Click **Start > Run** in windows
2. Type **CMD** and hit **Enter**
3. **Navigate** to the script file
4. Type in the name of the file and hit Enter

The TE launch (an icon will display in the systray) and execute your script.

To execute a script directly in Windows

1. **Navigate** to the script file in Windows Explorer
2. Double click the script file to launch it

The TE launch (an icon will display in the systray) and execute your script.

Creating scripts

You can interact with the TE directly from your own custom applications using common programming languages such as Visual Basic (VB) or in a scripting language supported by the Windows Scripting Host (WSH).

You can create a script from the development IDE of your choice, or you can create scripts from within CuteFTP 6 Professional by selecting **Tools > Macros & Scripting > New Script**. CuteFTP will open a template script file in a document window.

To create a new script file, you need to have some familiarity with programming concepts and, ideally, some experience with VB or Java. For those of you who have neither, or just want a template to work from, load one of the predefined scripts included with CuteFTP or the TE and edit it to suit your needs. A few sample scripts are included in this document.

Tip

If you don't have any programming experience, you can still create scripts using the session Record & Playback functionality in CuteFTP Professional.

Note

- Due to the wide range of scripts that CuteFTP is able to accommodate we are unable to offer technical support on individual scripts other than what is available in the help files and online Knowledge Base.
- If you are having trouble with your script, try to perform the desired action manually, using the CuteFTP GUI. If you cannot, then troubleshoot that problem first and then re-try your script.
- If you are able to perform the desired actions, and in the desired sequence when using the GUI, then the problem is not with CuteFTP or the FTP Server. The next thing to do is to troubleshoot your script line by line.
- Once the TE COM component is registered, you can create script files that will interface to it.
- Most of this document, including the various samples and glossary, are geared toward the novice programmer.

Reading the script examples

You can understand the examples, knowing what different symbols and font colors signify.

- Read maroon text as the syntax for a method or property
- Read **"Bold font in double quotes"** as parameters for a method or property
- Read red text as the method or property within an example script
- Read **'** any words after an apostrophe or single quote mark as comment that will not affect the script
- Read [Square brackets] as a place for optional items. Type the option, but not the brackets. Here's an example:
    - The syntax says;

      String Object.Option("[option name]") = true | false
    - In a program type it like this:

      MySite.Option("FilterDirs")=False
    - Don't type it like this

      MySite.Option("[FilterDirs]")=False

Methods, properties and wildcards overview

Use methods and properties to operate the Transfer Engine. Methods and properties are defined below.

Methods

A method can be described as a command or function that may accept arguments (parameters) and may return a certain type of value.

Example
Boolean Object.LocalExists(BSTR bstrName);
For this method, the type of the return value is Boolean. The command accepts an argument as a string value, here shown as BSTR (the type) and bstrName (a place holder for the argument).

In a program, you can execute a method and assign the return value to a variable all in one command.

Example
Exists = MySite.LocalExists "c:\temp\file.txt"
The argument "c:\temp\file.txt exists" is passed to the method LocalExists as a string (reason for the quotes). The variable Exists is then populated with a 1 or a 0 which is, in essence, true or false. You can then perform actions in your script based on those results.

Properties

A property is simply an attribute of a function (another word for method) or object internal to the TE framework. All properties have default values. Methods that rely on these properties will use default values unless you specify otherwise.

Most of the time, you can assign a value to a property or retrieve its value into a variable. You can set a property much like a local variable to your script.

Example
String Object.Protocol
The above property can accept a predefined set of string values or can be assigned to a variable to retrieve the currently selected value. The default in this case is "FTP".
You can assign a value to a property as follows:

Example
MySite.Protocol = "FTPS"
Upon subsequent connections, the TE will attempt to login using FTP over SSL, rather then via FTP, the default attribute for this property.

Wildcard masks

Wildcard masks are patterns of special characters used to filter file names. When a wildcard mask is matched against a file name, the two patterns are compared, letter-by-letter, from left to right until a mismatch occurs. If all the characters in both patterns compare positively, the file name matches the Wildcard Mask.

Using a script to connect to a remote site

This script will connect the TE to the public GlobalSCAPE FTP server and display the words **Connected OK** in a message box when the connection is successful.

```
Set MySite= CreateObject("CuteFTPPro.TEConnection")
MySite.Option ("ThrowError") = False
MySite.Protocol = "FTP"
MySite.Host = "ftp.cuteftp.com"
if cbool(MySite.Connect) then
```

```
MsgBox "Connected OK"
else
MsgBox MySite.ErrorDescription
end if
MySite.Disconnect
MySite.Close
```

Note

Save this script using notepad or other text editor with a .vbs extension. Then simply double-click on the file to execute the script.

Using a script to download

This script will connect to the GlobalSCAPE FTP site and download a file called 'index.txt' to two local folders; c:/temp1 and c:/temp2.

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Host = "ftp.globalscape.com"
MySite.Connect 'Recommended: call connect first
'next line changes to a predetermined folder so I can use a relative
path in the download method
MySite.RemoteFolder = "/pub/cuteftp"
MySite.LocalFolder = "c:\temp1"
MsgBox (MySite.RemoteFolder) 'display current remote folder
MySite.Download "index.txt"
MySite.Download "index.txt", "c:\temp2\index.txt"
'now verify that it downloaded okay
If MySite.LocalExists ("c:\temp1\index.txt") Then
MsgBox "File1 downloaded OK."
End If
If MySite.LocalExists ("c:\temp2\index.txt") Then
MsgBox "File2 downloaded OK."
End If
MsgBox "Done"
MySite.Disconnect
MySite.Close
```

Running scripts from the Windows Scheduled Tasks folder

The Windows Task Scheduler is the recommended tool for setting your TE scripts to run at specific times, but you can also use the NT 4 AT Scheduler.

To schedule a task

1. In Windows, click **Start**.
2. Choose **Programs > Accessories > System Tools > Scheduled Tasks**.
3. Double-click **Add Scheduled Task**. The **Scheduled Task Wizard** appears.
4. Click **Next**.
5. Click **Browse**. The **Select Program to Schedule** window appears.

6. Select your script and click **Open**. You are returned to the **Scheduled Task Wizard**.
7. Choose how often to run the script and click **Next**.
8. Select a starting time, recurrence, and starting date and click **Next**.
9. Enter a user name and password for the account that will run the script, and click **Next**.
10. Click **Finish**.

Notes

- In Windows 2000 or Windows XP you can set up the Transfer Engine so that it can run when no one is logged in.
- In Windows NT 4 you must have Internet Explorer 4 or higher and the Offline Browsing Pack installed to see the **Scheduled Tasks** folder.
- If you do not have the **Scheduled Tasks** folder, you can use the NT 4 AT scheduler.

Running scripts with the Windows NT 4 AT scheduler

The Windows Task Scheduler is the recommended tool for running TE scripts at specific times, but you can also use the NT 4 AT scheduler.

If you want to use the Windows NT 4 AT Scheduler to run TE scripts at specific times, follow these guidelines:

- A user must be logged in at the time the task is scheduled to run, though the computer can be locked.
- Include the **/interactive** switch in each task.
- Include the full path to the script with the file name and extension.

The following example runs a script at 2:12 pm, every day of the week. The script is on the computer's **C** drive, in the **example** folder, and is named **te-test.vbs**.

Example

```
at 14:12 /interactive /every:M,T,W,Th,F,S,Su c:\example\te-test.vbs
```

Notes

- For more details and switches for the AT Scheduler type **AT/?** at the command prompt.
- You can install a GUI task scheduler as part of Internet Explorer if you have version 4 or higher, see http://support.microsoft.com/default.aspx?scid=kb;EN-US;171229. The task scheduler is part of the **Offline Browsing Pack**.

Distributing the Transfer Engine

Distributing the TE

The Transfer Engine (TE) is a sub-component of CuteFTP Professional. It is COM enabled and provides an interface from which developers can access most of CuteFTP's file transfer related methods.

If you have created a script or application that calls the TE, you may wish to **distribute** your script or application to a group of end-users. In order for the TE to work properly on the end users' machines, you must **license**, **install** and **register** the TE on each machine.

Subsequently you must configure the TE so that it can run properly without the CuteFTP GUI (interface) installed, especially if you plan to run automated or scheduled tasks while not logged in, or if you plan to connect to SSL enabled FTP servers.

Licensing the TE (for distribution)

Licensing - Steps to License the TE

1. Determine how many end-user seats you will need.
2. Go to http://www.globalscape.com/cuteftppro.
3. Click on the Purchase link located on the page.
4. Purchase one license for each seat you need. If you need to purchase a license to cover a large amount of seats, please call us at 1-800-290-5054 or 1-210-308-8267.
5. Once registered, check your e-mail for the Serial Number

Licensing Considerations

- The TE inherits its registration process from CuteFTP. When running on a system that has a registered copy of CuteFTP installed, then you won't need to register the TE again. If CuteFTP is not installed (or registered), it will take registration matters into its own hands.
- If the TE finds itself alone, it goes through a 30-day trial just like the main CuteFTP application. Therefore, the TE may be distributed royalty free with unlimited runtimes for 30-days from the time it is installed on an end-user's computer.
- After residing 30 days on the target computer, it will disable itself if not properly registered. Therefore you must register the TE either prior to first launch, during the trial, or after the trial has expired.
- When you purchase a license for CuteFTP you may NOT distribute CuteFTP or any of its sub-components (such as the Transfer Engine) in an unlimited royalty free fashion.
- Each copy of the distributed Transfer Engine must be licensed for each computer where it is installed. This means that you must purchase a full license of CuteFTP for each distributed copy of the TE.
- You can purchase the necessary amount of licenses up front or after the TE has expired on the end-user machine. The benefit of licensing up front is that you can distribute the TE and register it on the end-user's machine so they Don't have to see any prompts or registration related dialogs.
- Once you have licensed the TE, you will need to install it (if not already installed) and then register it on each target machine.
- You are not permitted to distribute the TE as a component of an FTP client or other product that competes with CuteFTP or GlobalSCAPE's Secure FTP Server.

Installing the TE (for distribution)

Installation - Steps to Install the TE

1. Locate the TE component (ftpte.exe), the End User License Agreement (license.txt), and the TE's subordinate dynamic link libraries (sftp21.dll, ssl.dll, etc.) and the resource file (default.lng) on your developer (source) machine.
2. Package the TE along with the other items mentioned in Step 1, along with your installation program, script, or executable.

Note

You do NOT need to install the full copy of CuteFTP on the end-user's machine unless you require interaction from CuteFTP' GUI. If you were distributing a custom made application, this would not be the case.

Note

For setting specific options only available through the GUI (i.e. no property or method available), you can copy specific registry settings from the source computer to the destination. Most all settings are stored in the registry under the following hive:

HKEY_CURRENT_USER\Software\GlobalSCAPE\CuteFTP 6 Professional

Registering the TE (for distribution)

Registration - Automated Registration

The first method of registering the TE is the most efficient when dealing with a large amount of target systems, or when you won't have direct access to the target system. Your application installer or script can perform the steps below to transparently register the TE.

1. Purchase a license for the amount of TE seats you want to distribute.
2. Install a copy of the Transfer Engine and related components onto the target machine.
3. Create a new registry key on the target machine at the following location:
   HKEY_USERS\.DEFAULT\Software\GlobalSCAPE\CuteFTP Professional\Index
4. Create a new string value in the key above and insert the serial number as the data value.
   String Value = "1" Data = "[Your Serial Number]"
5. When the transfer engine is called for the first time, it will retrieve this value (the serial number) and use it to complete the registration with GlobalSCAPE.

Note

The target machine must have access to the Internet. If no Internet access is available, the registration process may fail and end up disabling the TE.

Note

The TE will automatically be registered if it is installed onto a computer with a registered copy of CuteFTP Professional. Keep in mind that the TE must be the same version as CuteFTP, i.e. the TE from CuteFTP Professional version 6.0 will NOT be automatically registered if it is installed on a computer with a registered copy of CuteFTP version 3.0.

Registration - Manual Registration

If you have physical access to the target machine, or you wish to register the copy installed on your developer machine, do the following:

1. Right-click on the **TE icon** in the systray to display the context menu (while running, the TE is represented by a yellow folder icon with up and down arrows)
2. Select **Enter Serial Numbe**r.
3. Paste or type the Serial Number.
4. Select **Register**.

A message should appear declaring that the registration was successful. The TE can now be used on that computer without interruptions.

If you receive an error message, verify that the serial number was typed correctly and that the number of licensed machines didn't exceed the licenses purchased. If all this is correct and you are still unable to register, contact our support department and provide them with the exact details of the error received.

COM registration of the TE (for distribution)

As a typical COM Component, the TE must be registered as such on the target system. The TE will automatically perform COM Registration if it has been manually executed prior to a script or application instantiating the TE Object.

The preferred method (for distributed versions of the TE on end-user's systems) is to set the appropriate registry entries before trying to call the TE.

COM Registration via the Registry

1. From your installation program, script, or custom application, write the registry entries below to the target system. The entries are shown in the standard .reg file notation.

Note

Include the correct the path to the Transfer Engine (/[installpath]/ftpte.exe) where **%MODULE%** is shown below.

REGEDIT4

[HKEY_CURRENT_USER\Software\Classes\CuteFTPPro.TEConnection.6]

@="TEConnection Class"

[HKEY_CURRENT_USER\Software\Classes\CuteFTPPro.TEConnection.6\CLSID]

@="{02172B7A-11D6-42b6-9550-41B281804714}"

[HKEY_CURRENT_USER\Software\Classes\CuteFTPPro.TEConnection]

@="TEConnection Class"

[HKEY_CURRENT_USER\Software\Classes\CuteFTPPro.TEConnection\CLSID]

@="{02172B7A-11D6-42b6-9550-41B281804714}"

[HKEY_CURRENT_USER\Software\Classes\CuteFTPPro.TEConnection\CurVer]

@="CuteFTPPro.TEConnection.6"

[HKEY_CURRENT_USER\Software\Classes\CLSID\{02172B7A-11D6-42b6-9550-41B281804714}]

@="TEConnection Class"

"AppID"="{310D78A7-4474-4c17-937A-7FF9D5A1B56C}"

[HKEY_CURRENT_USER\Software\Classes\CLSID\{02172B7A-11D6-42b6-9550-41B281804714}\LocalServer32]

@="%MODULE%" **<---PATH TO TE!!!**

[HKEY_CURRENT_USER\Software\Classes\CLSID\{02172B7A-11D6-42b6-9550-41B281804714}\ProgID]

@="CuteFTPPro.TEConnection.6"

[HKEY_CURRENT_USER\Software\Classes\CLSID\{02172B7A-11D6-42b6-9550-41B281804714}\Programmable]

[HKEY_CURRENT_USER\Software\Classes\CLSID\{02172B7A-11D6-42b6-9550-41B281804714}\TypeLib]

@="{1B04F22B-5012-432d-8EA0-B57DD75EBF9D}"

[HKEY_CURRENT_USER\Software\Classes\CLSID\{02172B7A-11D6-42b6-9550-41B281804714}\VersionIndependentProgID]

@="CuteFTPPro.TEConnection"

[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\AppID\ftpte.exe]

"AppID"="{310D78A7-4474-4c17-937A-7FF9D5A1B56C}"

[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\AppID\{310D78A7-4474-4c17-937A-7FF9D5A1B56C}]

@="TEConnection Class"

[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\{02172B7A-11D6-42b6-9550-41B281804714}]

@="TEConnection Class"

"AppID"="{310D78A7-4474-4c17-937A-7FF9D5A1B56C}"

[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\{02172B7A-11D6-42b6-9550-41B281804714}\LocalServer32]

@="%MODULE%" **<---PATH TO TE!!!**

[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\{02172B7A-11D6-42b6-9550-41B281804714}\ProgID]

@="CuteFTPPro.TEConnection.6"

[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\{02172B7A-11D6-42b6-9550-41B281804714}\Programmable]

[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\{02172B7A-11D6-42b6-9550-41B281804714}\TypeLib]

@="{1B04F22B-5012-432d-8EA0-B57DD75EBF9D}"

[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\{02172B7A-11D6-42b6-9550-41B281804714}\VersionIndependentProgID]

@="CuteFTPPro.TEConnection"

[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CuteFTPPro.TEConnection]

@="TEConnection Class"

[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CuteFTPPro.TEConnection\CLSID]

@="{02172B7A-11D6-42b6-9550-41B281804714}"

[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CuteFTPPro.TEConnection\CurVer]

@="CuteFTPPro.TEConnection.6"

[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CuteFTPPro.TEConnection.6]

@="TEConnection Class"

Finalizing TE distribution

Once the TE has been licensed, installed, registered, and COM registered on the target system, your script or custom application should be able to instantiate the TE object, and invoke any one of the supported methods or properties.

For a complete description of the TE, including its supported methods and properties, sample scripts and extended trouble shooting guide, please refer to the related topics in this user guide and our online knowledge base.

Troubleshooting TE distribution

Problems with running scripts *while not logged in* (suppressing message prompts or dialogs)

Your script should not contain Message Box functions or any other function that requires user input or shows a window. These prompts will not display while the system is not logged in.

You can physically suppress prompts from ever appearing. Use the –noprompt flag and execute the TE from the shell on the target machine before calling it from the script. E.g. run c:\program files\mycustomapp\ftpte.exe – noprompts.

Warning: Suppressing prompts may cause lockups if no default action is available for the event in question, or if the prompt was produced by an error.

The best way to avoid prompts is to properly configure the TE ahead of time to cover all possible prompt scenarios, such as overwrite conditions, or SSL server certificate acceptance dialogs. For example, set your overwrite actions under CuteFTP's global settings to Always Overwrite to avoid any overwrite prompts, or pre-define various rules to cover various overwrite scenarios. You can then export those rules as registry entries and import them onto the target machine.

You can also tell the TE to auto-accept SSL or SSH certificates using the AutoCloseMethod parameter of the Option method. Also refer to the following trouble shooting topic regarding running the TE while not logged in.

Locked TE

If your script or application runs the TE and doesn't return control to the script, it is possible that the TE has locked. Before trying to troubleshoot the problem, make sure you exit the locked TE process using the **Windows Task Manager** (kill the ftpte.exe process).

If your script is connecting to an SSL enabled server, double check the steps listed under Handling SSL Certificates. The primary cause for a locked TE during an SSL connection is the fact that the TE could not locate the client or server certificate, or the client certificate doesn't reside in the server's trusted list.

An incorrect FTP host address or other common connection problem (host not available, connection refused, etc.) may possibly lockup the TE. Verify that the host, proxy, socks, user and password property value are correct.

Another reason for a locked TE could be if the TE wasn't properly registered on the system and when launched, it tries to display the registration reminder (Web nag dialog) and can't.

Lastly, not properly configuring DCOM configuration for the TE to access the target user's settings may cause the TE to lock up.

Important Note

If the system is currently logged in when the TE is run, it will run as the user that you specified under DCOMCNFG. You will not see the TE icon in the system tray when running CuteFTP's main interface. Also, the hard coded user's settings are employed when the TE is run, not those of the currently logged in user. This won't be an issue if you are running the TE while the system is not logged in.

Handling SSL certificates (When running a script while not logged in)

Setting up the TE to connect to an FTP server over SSL (via scripting) involves several steps. Follow the instructions below carefully.

You must know whether the TE will require the use of server certificates only, or both server and client certificates.

Server Certificates

Most FTP SSL servers will provide a certificate to the client during authentication. The certificate proves the server's legitimacy to the client. The server's certificate must be added the client's Trusted Certificate store prior to any connection, otherwise the TE will lock up when trying to authenticate.

1. Export or obtain a copy of the server's certificate from the FTP server administrator. Otherwise connect to the server with the CuteFTP GUI (interface) from the developer machine and manually accept the server's certificate. You can then export it from the Trusted List under **Global Options > Security > SSL Security** to a staging directory of your choice.

2. Place/Install the certificate into the target machine's Default User directory. For example:

C:|Documents and Settings|Default User|Application Data|GlobalSCAPE|CuteFTP Pro|6.0|Security

3. If you will also be running the TE while logged in, place a copy of the server certificate under the USER folder on the target machine, as follows:

C:|Documents and Settings|[USERNAME]|Application Data|GlobalSCAPE|CuteFTP Pro|6.0|Security
- where [USERNAME] is the login name for the user.

4. You can also tell the TE to auto-accept SSL or SSH certificates using the AutoCloseMethod parameter of the Option method. Also refer to the following trouble shooting topic regarding running the TE while not logged in.

Now when you connect, the TE will see the server's certificate and proceed with the connection.

Client Certificates

Some FTP SSL servers require that the client provide a certificate. The certificate proves the veracity of the client and is an important factor in weeding out spurious FTP clients masquerading as legitimate ones.

To setup the target system to use client certificates

1. On the developer machine, create the client certificate and private key pair using the CuteFTP GUI (interface). The certificate creation utility is located under the **Tools > Global Settings > Security > SSL** dialog.

2. Add the client certificate to the server's trusted list. Do this by connecting to the server with the CuteFTP GUI (interface) from the developer machine. On the server, move the certificate from the Pending to the Trusted list. If your server only has a Trusted list, manually import the client certificate into the Trusted list.

3. Place the client certificate and private key into the target machine's USER folder:
C:|Documents and Settings|[USERNAME]|Application Data|GlobalSCAPE|CuteFTP Pro|6.0|Security
- where [USERNAME] is the login name for the user.

4. If you created the certificate set on the developer machine, export the following registry key:
HKEY_CURRENT_USER|Software|GlobalSCAPE|CuteFTP 6 Professional|Settings|SecuritySSL

5. Import the registry key from step 4 onto the target machine, or manually create the necessary entries. The entries are shown in standard .reg file notation.

```
REGEDIT4
[HKEY_CURRENT_USER\Software\GlobalSCAPE\CuteFTP Pro 3.0\Settings\SecuritySSL]
```

"SSLCertificate"="C:\\Documents and Settings\\[USERNAME]\\Application Data\\GlobalSCAPE\\CuteFTP Pro\\6.0\\Security\\client.crt"

"SSLPrivateKey"="C:\\Documents and Settings\\[USERNAME]\\Application Data\\GlobalSCAPE\\CuteFTP Pro\\6.0\\Security\\client.key"

"UseSSLCertificate"=dword:00000001

"UseSSLCertPassphrase"=dword:00000001

"ReuseSSLData"=dword:00000000

"WarnWhenToNonSecure"=dword:00000001

"DataTransportMethod"=dword:00000001

"SSLCertPassphrase"="[PASSWORD]"

- Where [USERNAME] is the login name for the target system and where [PASSWORD] is the encrypted password exported from the source machine. *Don't modify the password!*

Now when you connect, the TE will find the client certificate and use it to authenticate with the server.

Here is a sample script that connects to a fictitious site running SSL Implicit over port 990. It also writes to the event log for debugging purposes. if you use this script to test your setup, make sure you modify the MySite.Host line to include your actual FTP SSL server host and login information.

```
#file test.vbs
Const EVENT_SUCCESS = 0
Set objShell = Wscript.CreateObject("Wscript.Shell")
objShell.LogEvent EVENT_SUCCESS,"AT Loaded me"
Set MySite= CreateObject("CuteFTPPro.TEConnection")
MySite.Option ("ThrowError") = True
MySite.Host = "ftp://myuser:mypass@ftp.mysslserver.com:990
objShell.LogEvent EVENT_SUCCESS,"vars set"
MySite.Connect
objShell.LogEvent EVENT_SUCCESS, "connected"
MySite.Disconnect
MySite.Close
```

Methods

Finding a method alphabetically

DownloadAsync
GetList
LocalExists
LocalRemove
LocalRename
TransferURL
TransferURLAsync
Upload
UploadAsync
Wait
WriteToLOG

Finding a method by category

Here you can link to every method listed in the help file.

Connection
Connect
Disconnect
Close

Transfer
Upload
Download
S2Sxfer
Synchronize
TransferURL

Asynchronous Transfer
UploadAsync
DownloadAsync
TransferURLAsync

Files and Folders
CreateLocalFolder
LocalExists
LocalRename
LocalRemove
CreateRemoteFolder
RemoteExists
RemoteRename
RemoteRemove

Commands
AbortAsync
RemoteCommand
TECommand
Wait
WriteToLOG

Basic

Connecting to a remote server
Description
Use the **Connect** method to log in to the remote server. Before calling it, you set the protocol, host address, user name, and password to establish a successful connection.

Syntax
```
Object.Connect
```

Example
```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Protocol = "FTP"
MySite.Host = "ftp.cuteftp.net"
```

```
MySite.Login = "username"
MySite.Password = "password"
MySite.Connect
```

Notes

Any time you call a Transfer function, the connect function is called indirectly. It is recommended you explicitly invoke the **Connect** function though it is not necessary. You can always use the IsConnected property to determine whether you are connected at any given time.

Downloading files

Description

Use the **Download** method to transfer a file or folder from a remote location to your local hard drive.

Syntax

```
Object.Download (BSTR strRemoteName ,BSTR strLocalName , long
nMultiPartNumber)
```

Parameters

strLocalName	This is optional, use it only if you want to change the destination name or path for the downloaded files or folder. You can use absolute or relative paths with or without wildcards.
strRemoteName	This is the path to the remote item you are downloading. You can use absolute or relative paths with or without wildcards.
nMultiPartNumber	Use this to split the download into multiple parts. The default value = 1. The value specifies the number of parts used for the download.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Specify user, pass, host, and connect as normal...
MySite.Connect 'Recommended: call connect first
'next line changes to a predetermined folder so you can use a relative
path in the download method
MySite.RemoteFolder = "/c:/Inetpub/ftproot/Temp/Temp/"
MsgBox (MySite.RemoteFolder) 'display current remote folder
MySite.Download "agent.ini", "c:\temp\agent1.ini"
'now verify downloaded ok
If CBool(MySite.LocalExists ("c:\temp\agent1.ini")) Then
MsgBox "File downloaded OK."
End If
```

Notes

- Setting the Multi-part download attribute can greatly increase the transfer speed for larger files under certain conditions. For example, the site must support multiple concurrent connections from the same user and you must have significant bandwidth.
- The Download method is a syncrhonous command, meaning it must finish executing before subsequent commands in your script can be called. Use the method DownloadAsync (which also

supports Multi-part transfers) to asyncrhonously download files, which allows you to execute the rest of the script while the download(s) take place.

Uploading files

Description

Use the **Upload** method to transfer a file or folder from a local hard drive to a remote server.

Syntax

```
Object.Upload(BSTR strLocalName ,BSTR strRemoteName , long nMultiPartNumber)
```

Parameters

strRemoteName	This is optional, use it only if you want to change the destination name or path for the uploaded files or folders. You can use absolute or relative paths with or without wildcards.
strLocalName	This is the path to the local item you are uploading. You can use absolute or relative paths with or without wildcards.
nMultiPartNumber	Use this to split the upload into multiple parts. The default value = 1. The value specifies the number of parts used for the download.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Specify user, pass, host, and connect as normal...
MySite.Connect 'Recommended: call connect first
MySite.RemoteFolder = "Temp"
MySite.LocalFolder = "C:\123"
'using relative path, all files in folder 123 are uploaded to the folder Temp
off the current folder on the server.
MySite.Upload "*.*"
```

Notes

- You can only use multi-part uploads with servers that support the COMB command. Currently, only GlobalSCAPE Secure FTP server supports the COMB command.
- Setting the Multi-part upload attribute can greatly increase the transfer speed for larger files under certain conditions. For example, the site must support multiple concurrent connections from the same user and you must have significant bandwidth.
- The Upload method is a syncrhonous command, meaning it must finish executing before subsequent commands in your script can be called. Use the method UploadAsync (which also supports Multi-part transfers) to asyncrhonously upload files, which allows you to execute the rest of the script while the upload(s) take place.

Transferring from a URL

Description

Use the **TransferURL** method to download files directly from a Web address.

Syntax

```
Object.TransferURL(BSTR bstrRemoteName, long nMultipartNumber)
```

Parameters

bstrRemoteName	This is a string value for the URL for the file transfer such as; (ftp://ftp.globalscape.com/pub/cuteftp/cuteftp.exe).
nMultipartNumber	This is optional and will split a file into parts for transfer. The default = 1.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.TransferURL "ftp://ftp.globalscape.com/pub/cuteftp/cuteftp.exe"
```

The TE recognizes these URL formats:

```
ftp://user:pass@ ftp.sitename.com:port
ftp://user:pass@ ftp.sitename.com
ftp://user@ ftp.sitename.com
ftp:// ftp.sitename.com:port
ftp:// ftp.sitename.com
ftp://ftp.sitename.com/pub l: user p: pass ß l: user is a lowercase
"L", not "one"
ftp://ftp.sitename.com/pub:44 l: user p: pass
ftp://ftp.sitename.com/pub port:44 l: user p: pass
ftp://ftp.sitename.com/pub l/p: user/pass
ftp://ftp.sitename.com/pub:44 l/p: user/pass
ftp://ftp.sitename.com/pub p:44 l/p: user/pass
ftp://ftp.sitename.com/pub port:44 l/p: user/pass
ftp://ftp.sitename.com/pub l: user p: pass
ftp://ftp.sitename.com/pub:44 l: user p: pass
ftp://ftp.sitename.com/pub p:44 l: user p: pass
ftp://ftp.sitename.com/pub port:44 l: user p: pass
```

Creating a local folder

Description

Use the **CreateLocalFolder** method to create a new folder (directory) on your local hard drive.

Syntax

```
Object.CreateLocalFolder(BSTR strName)
```

Parameters

BstrName	This contains the folder's relative or absolute path.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.CreateLocalFolder "c:\temp\New Folder"
'now check to see if it was created
MySite.LocalFolder = "c:\temp\New Folder"
currentdir = MySite.LocalFolder
MsgBox currentdir
```

Creating a remote folder

Description

Use the **CreateRemoteFolder** method to create a new folder (directory) on a remote server.

Syntax

```
Object.CreateRemoteFolder(BSTR strName)
```

Parameters

BstrName	This contains the folder's relative or absolute path.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user,
password, etc.
MySite.Connect
MySite.CreateRemoteFolder("/dir") 'creates /dir (because absolute path
was used)
MySite.RemoteFolder = "/dir" 'now change to the newly create 'dir'
folder
MySite.CreateRemoteFolder("dir2") 'creates /dir/dir2 (because relative path
was used)
```

Renaming a remote file or folder

Description

Use the **RemoteRename** method to rename a file or folder on the remote server.

Syntax

```
Object.RemoteRename(BSTR bstrFrom, BSTR bstrTo)
```

Parameters

BstrFrom	This contains the folder's old name in a relative or absolute path. The path must be the same in both parameters.
BstrTo	This contains the folder's new name in a relative or absolute path. The path must be the same in both parameters.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user, password,
etc.
MySite.RemoteRename "/pub/user1/file.txt", "/pub/user1/file3.txt"
```

Renaming a local file or folder

Description

Use the **LocalRename** method to rename a file or folder on your local hard disk.

Syntax

```
Object.LocalRename(BSTR bstrFrom, BSTR bstrTo)
```

Parameters

BstrFrom	This contains the folder's old name in its full absolute path.
BstrTo	This contains the folder's new name in its full absolute path.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user, password, etc.
MySite.LocalRename "c:\file.txt", "e:\temp\users\file.exe"
```

Note

Make sure you specify the full source and destination path. If you had written the command as MySite.LocalRename "c:\file.txt", "file.exe", thinking it would rename it using relative paths, you might be surprised to find that your file has been moved to your profiles folder (system dependent). In essence, the LocalRename is similar to a MOVE command issued through drag and drop with the added name change sequence.

Deleting a remote file or folder

Description

Use the **RemoteRemove** method to delete a remote file or folder. You can specify the file or folder with absolute or relative paths. If the command fails, make sure you have specified the correct path and that you have the appropriate permissions required to delete the item.

Syntax

```
Object.RemoteRemove(BSTR bstrName)
```

Parameters

BstrName	This contains the folder's relative or absolute path.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user, password, etc.
MySite.RemoteRemove "/pub/user1/file.txt"
MySite.RemoteRemove("file.ext") 'removes if exact match
MySite.RemoteRemove("*.obj") 'wild card match with a specific extension
MySite.RemoteRemove("*.aaa" & Chr(10) & "*.bbb" & Chr(10) & "t*")
'various wildcard filters
```

Deleting a local file or folder

Description

Use the **LocalRemove** method to delete a local file or folder. Use the absolute path name for the item you want to delete.

Syntax

```
Object.LocalRemove(BSTR bstrName)
```

Parameters

BstrName	This contains the items full absolute path.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user,
password, etc.
MySite.LocalRemove "c:\temp\file.txt"
```

Notes

You can also you wild cards to replace the string value of bstrName. These wild card masks include "*", "?", and '\n" (new line symbol, also known as Chr(10) in VB)

Example 2

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user,
password, etc.
MySite.LocalFolder = "c:\temp"
MySite.LocalRemove("file.ext") 'removes if exact match
MySite.LocalRemove("*.obj") 'wild card match with a specific extension
MySite.LocalRemove("*.aaa" & Chr(10) & "*.bbb" & Chr(10) & "t*")
'various wildcard filters
```

Checking for a remote file or folder

Description

Use the RemoteExists method to verify that a remote file or folder exists. BstrName should be the full path.

Syntax

```
Boolean Object.RemoteExists(BSTR bstrName)
```

Parameters

Return values

true	File or folder does exist
false	File or folder does not exist

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user, password, etc.
R = MySite.RemoteExists( "/pub/user/file.txt")
If (R) Then
MsgBox "File exists on remote side"
Else
MsgBox "File doesn't exist on remote side"
End if
```

Note

If using an "if not" conditional, use CBool instead of boolean as the return type.

Correct:

```
if not CBool(MySite.RemoteExists(strRemote)) then
MsgBox "Error! " & MySite.ErrorDescription
```

Incorrect:

```
if not MySite.RemoteExists(strRemote) then
MsgBox "Error! " & MySiteErrorDescription
```

Checking for a local file or folder

Description

Use the **LocalExists** method to verify that a local file or folder exists. BstrName should be the full path.

Syntax

```
Boolean Object.LocalExists(BSTR bstrName);
```

Parameters

Return values

True	File or folder does exist
False	File or folder does not exist

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user, password, etc.
L = MySite.LocalExists("c:\temp\file.txt")
If (L) Then
MsgBox "File exists on local side"
Else
MsgBox "File doesn't exist on local side"
End if
```

Advanced

Sending commands to the Transfer Engine

Description

Use the **TECommand** method to pass various commands to the Transfer Engine component. If you leave the Transfer Engine running for a long time, it is recommended you include the **DeleteFinished** or **DeleteAll** in your scripts to occasionally empty the queue. The Transfer Engine does not empty the queue automatically.

Syntax

```
Object.TECommand(BSTR bstrParameter)
```

Parameters

"CLOSE" or **"EXIT"**	Closes the TE (all tasks will be stopped)

"EXITNOPENDING"	Closes the TE if no pending tasks are available
"UPDATESETTINGS"	Reloads settings from the registry (Note you can modify the registry from VBS)
"DELETEFINISHED"	Removes finished items from queue
"DELETEALL"	Removes all items from queue

Note

You can use any number of spaces or underscores inside these parameters. For example, "update setting" and "_UPDATE_SETTINGS_" are the same.

If the Transfer Engine has several tasks, or must run for a long time, you should include the **DeleteFinished** command in your scripts to occasionally clear items from the queue.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.TECommand("delete finished")
```

Sending commands to a server

Description

Use the **RemoteCommand** method to send the server any supported command. This function acts like the raw input command found in CuteFTP.

Syntax

```
Object.RemoteCommand(BSTR bstrCmd)
```

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user, password, etc.
MySite.RemoteCommand "PWD" 'sends a print working directory command
```

Note

If you include the word **"LIST"**, or **"RETR"**, or **"STOR"** in bstrCmd then the Transfer Engine will open the data connection, perform the operation (to the buffer) and then discard it. You should use the **GetList**, **Download** or **Upload** methods to accomplish these tasks.

You can concatenate strings to perform custom commands. For example, if you wanted to do use CHMOD on a file, you could do as shown in the example below:

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user, password, etc.
strMODE = "777" 'full permissions
strFile = InputBox("Type in the file name below ", "CHMOD Dialog",
"cftppro10.log") 'an input dialog with some default content
strvar = "SITE CHMOD " & strMODE & " " & strFile 'concatenate the values
MySite.RemoteCommand strvar 'send the command
```

Synchronizing folders with the TE

Description

Use the **Synchronize** method to perform one or two way mirrors of a remote and local folder's contents.

Note: The synchronize method contains as many as 9 parameters. Be sure to use absolute path names for both local and remote folder paths.

Syntax

```
Object.Synchronize(BSTR bstrRemoteName, BSTR bstrLocalName, long
nDirection, long nAction, long nCasehandling, BOOL bRecursive, BOOL
bIgnoreLinks, BOOL bDelDestination, BOOL bPromptDel);
```

Parameters

#	Name	Value
1	BstrRemoteName	String value that specifies the absolute path name of the remote folder
2	BstrLocalName	String value that specifies the absolute path name of the local folder
3	nDirection	0 = Mirror Local (make the remote look just like the local) 1 = Mirror Remote (make the local look just like the remote) 2 = Mirror Both
4	nAction	When nDirection = 2 (Mirror Both) 0 = Mirror the more recent file 1 = Mirror the larger file 2 = Prompt for matching file names 3 = Skip mirroring files with the same names --- When nDirection = 0 or 1 (Mirror Local or Remote) 0 = Use Global Overwrite settings in the CuteFTP shell for matching filenames 1 = Always overwrite the file with a matching name 2 = Numerate the file (filename[1]) 3 = Skip
5	nCaseHandling	0 = Transfer first and skip the rest (default) 1 = Show rename prompt 2 = Numerate Note: This action applies when matching filenames are found and the only difference is the filename case.
6	bRecursive	0 = Don't sync subfolders 1 = Apply sync to subfolders (default)
7	bIgnoreLinks	0 = Don't ignore symbolic links 1 = Ignore symbolic links (default)
8	bDelDestination	0 = Don't remove destination 1 = Remove destination if source does not exist (default) Note: This action only applies to one-way mirroring. If a file exists in the destination that isn't in the source being mirrored, then delete the destination file.
9	bPromptDel	0 = Don't prompt before removing destination

		1 = Prompt before removing destination (default) Note: Only applies to one-way mirroring when DelDestination is True.

Examples

```
'Simple synchronize using minimal parameters
Set MySite = CreateObject("CuteFTPPro.TEConnection")
‘Don’t forget to initialize all necessary fields for MySite : host name, user, password, etc.

MySite.Connect
MySite.Synchronize "/pub/myfolder", "C:\mysitesfiles", 0, 1
'This will perform a local mirror, overwriting any matching filename.
```

```
'Simple synchronize using minimal parameters
Set MySite = CreateObject("CuteFTPPro.TEConnection")
‘Don’t forget to initialize all necessary fields for MySite : host name, user, password, etc.

MySite.Connect
MySite.Synchronize "/pub/myfolder", "C:\mysitesfiles", 2, 0
'This will perform full mirror (both), overwriting older files when a matching filename is found.
```

```
'Slightly more complex synchronize routine used to synchronize bookmarks. Uses variables for the path names
strRemotePath = "\Favorites"
strLocalPath = "C:\Documents and Settings\username\Favorites"
‘Don’t forget to initialize all necessary fields for MySite : host name, user, password, etc.
MySite.Connect
        If (Not (MySite.IsConnected)) Then
        MsgBox "Unable to connect to server:" + MySite.Host
        End if
MySite.Synchronize strRemotePath, strLocalPath, 2, 3, 0, 1, 1, 0, 1
'Performs a full mirror, skips matching filenames, transfers only the first file if multiple files are found with the same name but different case, applies to subfolders, ignores symbolic links, does not remove destination files if the source doesn't exist (N/A when dealing with dual mirror), and prompt prior to deleting anything (N/A when dealing with dual mirror).
MsgBox "DONE!" ‘Alert me to the completed task
MySite.Disconnect ‘Disconnects from the site when done
MySite.Close ‘Close the Transfer Engine process
```

```
‘A full synchronizaiton VB subroutine:
Sub Sync()
```

```
Dim MySite
Set MySite = CreateObject("CuteFTPPro.TEConnection")

strHost = "ftp.yourhost.com"
strPath = "/pub"
strLocalPath = "c:\temp\sync_test"

strHost = InputBox("Enter host", "CuteFTP Pro", strHost)
strPath = InputBox("Enter remote path", "CuteFTP Pro", strPath)
strLocalPath = InputBox("Enter local path", "CuteFTP Pro", strLocalPath)

MySite.Host = strHost

MySite.CaseHandling = 1
MySite.Recursive = False
MySite.IgnoreLinks = True
MySite.DeleteDestination = False
MySite.PromptDelete = True

 nUserChoise = MsgBox ("Mirror remote: " & strHost & strPath & " to local " & strLocalPath & " ?",
 vbYesNoCancel)
If nUserChoise = vbYes Then
        MySite.Synchronize strPath, strLocalPath, 1, 0
else
         nUserChoise = MsgBox ("Mirror local: " & strHost & strLocalPath & " to remote " & strPath
         & " ?", vbYesNoCancel)
        If nUserChoise = vbYes Then
                MySite.Synchronize strPath, strLocalPath, 0, 0
        else
                  nUserChoise = MsgBox ("Mirror both: " & strHost & strPath & " <- > " &
                  strLocalPath & " ?", vbYesNoCancel)
                If nUserChoise = vbYes Then
                        MySite.Synchronize strPath, strLocalPath, 2, 1
                else
                End if
        End if
End if

End Sub
```

Transferring from site to site (FXP)

Description

Use the **S2Sxfer** method to transfer a file from one remote site to another. You must use absolute path names for the source and target folders.

Syntax

```
Boolean Object.S2Sxfer(BSTR bstrSourceName, BSTR bstrDestName, BSTR
bstrPeerHost, BSTR bstrPeerLogin, BSTR bstrPeerPassword, long Port, BSTR
bstrPeerProtocol);
```

Parameters

bstrSourceName	source file and folder name
bstrDestName	target file and folder name
bstrPeerHost	target host name
bstrPeerLogin	target login
bstrPeerPassword	target password
Port	target port
BstrPeerProtocol	target protocol (FXP can be applied for hosts with different protocols FTP, FTP-S)

Return values

true	Transfer was successful
false	Transfer failed

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user, password, etc.
MySite.Connect
MySite.S2Sxfer "/cftppro14.log", "/home/myfolder/cftppro14.log",
"ftp.destinationhost.com", "username", "passwrd" 'everything else left as
default
```

Writing messages in a log

Description

Use the **WriteToLOG** method to write a message directly to the connection log saved in the path set in CuteFTP Professional's global options. It is useful for documenting events to aid in script debugging.

Syntax

```
Object.WriteToLOG(BSTR bstr , BSTR bstrType)
```

Parameters

bstr	The log message.
bstrType	A log message, type: "STATUS", "ERROR", "NOTE", "COMMAND", "RAW". The default is "STATUS".

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user, password, etc.
MySite.TransferURLAsync "ftp://ftp.cuteftp.com/pub/cuteftp"
strResult = MySite.Wait
If (strResult = "FINISHED") then
MySite.WriteToLOG "Transfer successful!!"
MySite.WriteToLOG "Transfer successful!!", "error"
MySite.WriteToLOG "Transfer successful!!", "note"
End if
As result, the log will contain the following strings:
STATUS: > Transfer successful!!
ERROR: > Transfer successful!!
Note > Transfer successful!!
```

Resuming a transfer

Though there is no resume method you can use the **RemoteCommand** method to send the APPE (Append) command to a server. APPE is the FTP command to resume a transfer.

Example

```
MySite.RemoteCommand "APPE html/test.txt"
```

Note

The **RemoteCommand** method exists to allow you to manually pass any command to the server, even if that command is not natively supported through the TE API.

Waiting for a task to complete

Description

Use the **Wait** method to tell the Transfer Engine to hold all other tasks until a specific asynchronous task is completed. Then continue with the rest of the script.

Syntax

```
String Object.Wait (long taskIndex, long timeout)
```

Parameters

taskIndex	This is the task index in the asynchronous tasks array. The default = -1 (which is current task). It can range from 0 to the total number of tasks minus one.
timeout	Determines how long (in milliseconds) to wait for a finished, cancelled, or errored status before continuing with the script.

Return value

"CANCELLED"	Transfer was stopped by the user
"FINISHED"	Transfer was successfully finished
"ERROR"	There were errors during the transfer
"SKIPPED"	The transfer was skipped (file overwrite rules)

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user, password, etc.
MySite.TransferURLAsync "ftp://ftp.cuteftp.com/pub/cuteftp"
strResult = MySite.Wait
If (strResult = "ERROR") then
MsgBox "warning! An error has occurred!"
End if
```

Notes

- The default timeout value for the **Wait** method is 21,805,184 milliseconds, or approximately 6 hours.
- The maximum possible value for timeout is 2,147,483,647 milliseconds, or just under 25 days.
- You can set the **Wait** method to never timeout by using a timeout value of 0 (zero).
- For hints on **Wait** method timeouts, see Timeout strategies.

Stopping transfers and other events

Description

Use the **Disconnect** method to end an event in progress. It is normally used to stop a file transfer.

Syntax

```
Object.Disconnect()
```

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.TransferURLAsync
MySite.Disconnect
```

Notes

- Disconnect is not the same as the standard FTP command QUIT. Instead it is simply a command to stop the current transfer task. You can issue a QUIT command prior to invoking Disconnect by using the RemoteCommand function.
- A socket disconnect (more brutal than QUIT) to an FTP server occurs automatically after all transfers and other commands have finished. This is similar to the Connect command, which is implicitly called when a transfer method is invoked.
- The example above uses the TransferURLAsync command to pass the TE an FTP file location to download. Since the TransferURLAsync command is asynchronous, you could called TransferURLAsync multiple times consecutively.
- What happens if you issue a Disconnect command after issuing multiple TransferURLAsync commands? The last called TransferURLAsync request is the one that is cancelled.

Closing the Transfer Engine

Description

Use the **Close** function to exit the Transfer Engine. You can include parameters to only exit on certain conditions.

Syntax

```
Object.Close (BSTR bstrParameter)
```

Parameters

""(default empty), "CLOSE", "EXIT"	Closes TE (all tasks will be stopped)
"EXITNOPENDING"	Closes TE if no pending tasks available

Note

You can use any number of spaces or underscores inside these parameters. For example, "exit no pending" and "_EXIT_NO_PENDING_" are the same.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.LocalFolder = "c:\temp"
MySite.TransferURLAsync "ftp://ftp.cuteftp.com/pub/cuteftp"
MySite.Close "EXITNOPENDING"
```

Uploading items asynchronously

Description

Use the **UploadAsync** method to upload a file or folder to a remote server asynchronously. An asynchronous upload starts and then returns control to the script before the transfer finishes. This allows you to perform many simultaneous transfers because the method does not wait for the upload to end.

Immediately after you call this method subsequent methods in your script will be called, so be careful when timing certain events.

If **UploadAsync** encounters a problem when trying to complete its task, it will not throw a COM, ATL, or VB error. **UploadAsync** will also adhere to your max global and per site settings.

Syntax

```
Object.UploadAsync(BSTR strLocalName [,BSTR strRemoteName [, long
nMultiPartNumber]])
```

Parameters

strRemoteName	This is optional, use it only if you want to change the destination name or path for the uploaded files or folders. You can use absolute or relative paths with or without wildcards.
strLocalName	This is the path to the local item you are uploading. You can use absolute or relative paths with or without wildcards.
nMultiPartNumber	Use this to split the upload into multiple parts. The default value = 1. The value specifies the number of parts used for the download.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user, password, etc.
MySite.Connect
MySite.UploadAsync "c:\temp\*.vob", "/DVDFiles" 'uploads as many .vob files as
concurrent connection settings allow
```

Note

The Multi-part parameter for the **UploadAysnc** is currently limited to servers that support this operation, due to the need to recombine the files after the upload of each part has completed. As of September 2002, GlobalSCAPE's Secure FTP server was the only server to support this function.

Downloading items asynchronously

Description

Use the **DownloadAsync** method to download a file or folder to the local hard drive asynchronously. An asynchronous download starts and then returns control to the script before the transfer finishes. This allows you to perform many simultaneous transfers because the method does not wait for the download to end. Immediately after you call this method subsequent methods in your script will be called, so be careful when timing certain events.

If **DownloadAsync** encounters a problem when trying to complete its task, it will not throw a COM, ATL, or VB error. **DownloadAsync** will also adhere to your max global and per site settings.

Syntax

```
Object.DownloadAsync(BSTR strRemoteName ,BSTR strLocalName , long
nMultiPartNumber)
```

Parameters

strLocalName	This is optional, use it only if you want to change the destination name or path for the downloaded files or folder. You can use absolute or relative paths with or without wildcards.
strRemoteName	This is the path to the remote item you are downloading. You can use absolute or relative paths with or without wildcards.
nMultiPartNumber	Use this to split the download into multiple parts. The default value = 1. The value specifies the number of parts used for the download.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Host = "ftp.cuteftp.com"
MySite.Connect
MySite.DownloadAsync "/pub/cuteftp/english/*", "c:\temp" 'downloads all files
in the pub/cuteftp folder
```

Notes

You can call the **DownloadAsync** method many times sequentially in a script. Each call, in turn, opens a new data connection to the specified server, enabling you to transfer multiple files simultaneously. This, combined with the ability to transfer the file in multiple parts, greatly increases the overall transfer speed and execution of your task.

Transferring from URLs asynchronously

Description

- Use the **TransferURLAsync** method to start a download from a web address, and return control to the script immediately. This method does not wait for the transfer to end. You can use it to perform many simultaneous transfers.
- The difference between this method and DownloadAsync is simply the ability to specify all of the necessary parameters in one command, rather than having to set the user name, port, host, etc. and then transfer a file.
- Immediately after you call this method subsequent methods in your script will be called, so be careful when timing certain events.
- If **TransferURLAsync** encounters a problem when trying to complete its task, it will not throw a COM, ATL, or VB error. **TransferURLAsync** will also adhere to your max global and per site settings. You can't use wildcards in this method.

Syntax

```
Object.TransferURLAsync(BSTR bstrURL [, long nMultipartNumber])
```

Parameters

bstrURL	A string value for the URL for the file transfer (ftp://ftp.cuteftp.com/pub/cuteftp)
nMultipartNumber	An optional multipart parameter with default = 1

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.TransferURLAsync "ftp://ftp.cuteftp.com/pub/cuteftp/english"
'this will transfer the entire English CuteFTP directory from the CuteFTP.com
ftp site.
```

Normal Formats Supported

```
ftp://user:pass@ ftp.sitename.com:port
ftp://user:pass@ ftp.sitename.com
ftp://user@ ftp.sitename.com
ftp:// ftp.sitename.com:port
ftp:// ftp.sitename.com
ftp://ftp.sitename.com/pub l: user p: pass ß l: user is a lowercase
"L", not "one"
ftp://ftp.sitename.com/pub:44 l: user p: pass
ftp://ftp.sitename.com/pub port:44 l: user p: pass
ftp://ftp.sitename.com/pub l/p: user/pass
ftp://ftp.sitename.com/pub:44 l/p: user/pass
ftp://ftp.sitename.com/pub p:44 l/p: user/pass
ftp://ftp.sitename.com/pub port:44 l/p: user/pass
ftp://ftp.sitename.com/pub l: user p: pass
ftp://ftp.sitename.com/pub:44 l: user p: pass
ftp://ftp.sitename.com/pub p:44 l: user p: pass
ftp://ftp.sitename.com/pub port:44 l: user p: pass
```

Extended URL formats

Aside from the standard URL formats shown above, additional specifiers may be used to denote the direction of transfer and download path. Use the extended format to perform site to site transfers, uploads, targeted downloads, and more.

The standard url ftp://user:pass@ftp.host.com will be used in the following examples:

Format: Normal URL transfer

Example: `MySite.TransferURLAsync "ftp://user:pass@ftp.host.com"`
`'download site to the default download folder`

Format: URL [space] "-->" [space] LPATH

Example: `MySite.TransferURLAsync "ftp://user:pass@ftp.host.com --> c:\temp"`

```
'download site to the c:\temp folder
```

Format: URL [space] "<--" [space] LPATH
Example: `MySite.TransferURLAsync "ftp://user:pass@ftp.host.com <-- c:\web"`
`'upload files from c:\web to the site`

Format: URL1 [space] "<->" [space] URL2
Example: `MySite.TransferURLAsync "ftp://user:pass@ftp.host.com <->`
`ftp://user2:pass2@ftp.host2.com" 'perform a site to site transfer`

Format: URL [space] "<==" [space] LPATH
Example: `MySite.TransferURLAsync "ftp://user:pass@ftp.host.com <== c:\web"`
`'synchronize (mirror local) the c:\web folder to the site`

Format: URL [space] "==>" [space] LPATH
Example: `MySite.TransferURLAsync "ftp://user:pass@ftp.host.com ==> c:\web"`
`'synchronize (mirror remote) the site to c:\web`

Format: URL [space] "<=>" [space] LPATH
Example: `MySite.TransferURLAsync "ftp://user:pass@ftp.host.com <=> c:\web"`
`'synchronize both local and remote (mirror both)`

Stopping asynchronous transfers

Description

Use the **AbortAsync** function to stop an asynchronous task created previously by UploadAsync, DownloadAsync or TransferURLAsync. Refer to those methods for more information.

Syntax

```
Object.AbortAsync(long taskIdx)
```

Parameters

TaskIdx	This is a task index in the array of tasks created by the various asynchronous methods. The default value of taskIdx is - 1 which specifies all the asynchronous tasks in array. TaskIdx should be between 0 and AsyncTaskNumber minus one.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Option("CleanupAsync") = False
'Initialize all necessary fields for MySite : host name, user, password, etc.
MySite.Connect
MySite.DownloadAsync "*.*", "c:\temp"
MsgBox "Number of tasks created: " & MySite.AsyncTaskNumber
If MySite.AsyncTaskNumber > 10 then
MsgBox "Aborting 11th task"
MySite.AbortAsync ( 10 ) 'abort task # 11 (one less then total, since
starts from 0)
End if
```

Notes

You must set `MySite.Option("CleanupAsync") = False` so that all asynchronous tasks are counted. Otherwise, only the last asynchronous task launched will be referenced when calling the AbortAsync method.

Automatically encrypting and compressing transfers

To automatically encrypt and compress files before transfer, you will need both CuteZIP and CuteFTP Professional. You can also compress and encrypt files after transferring them by using GlobalSCAPE Secure FTP Server's Custom Site Commands. The benefits of encrypting files prior or post transfer depends on the circumstances and level of trust for the particular host.

In the example script below, an entire folder (including sub-folders) is compressed, encrypted (using Twofish 128 bit encryption) and then transferred via regular FTP to an FTP server. Since the archived file is encrypted, there is no need to connect using SSL, OTP, or SSH2 unless you wished to also protect the login process.

Example

```
Dim WshShell, MySite, Return
Set WshShell = CreateObject("WScript.Shell") 'Window's Scripting Host shell
object
'next line calls the run method of the WSH shell object. It returns true once
CuteZIP does its thing.
'The complete command line instructions for CuteZIP are located here.
If Return = WshShell.run ("c:\progra~1\global~1\CuteZIP\cutezip.exe -c -p12345
c:\archive c:\temp", 0, true) Then
  Set MySite = CreateObject("CuteFTPPro.TEConnection")
  MySite.Option ("ThrowError") = True
  MySite.Host = "ftp://user:pass@myftpsite.com" 'one of the ways to connect
using the TE
  MySite.Connect
  MySite.Upload "c:\archive.zip" 'upload the new archive, then check to see if
it made it up to the server.
  if not CBool(MySite.RemoteExists("\archive.zip")) then
     MsgBox "Failed to Upload, Exiting!"
  Else
     MsgBox "Success!"
  End If
  MySite.Disconnect
  MySite.Close
Else
MsgBox "Compression and Encryption Failed, Exiting!"
End If
```

Note

You can optionally protect the FTP login by connecting with SSL, SSH2 or OTP using CuteFTP 6 Professional's Transfer Engine (GlobalSCAPE's Secure FTP Server 2 supports SSL, OTP, and SSH2 logins). Use the Protocol property to set the connection type prior to calling MySite.Host and MySite.Connect.

Retrieving a folder listing

Description

Use the **GetList** method to download folder listings.

Syntax

```
Object.GetList(BSTR bstrPath, BSTR bstrLocalFile, BSTR bstrFormat, BOOL *pVal)
```

Parameters

bstrPath	The remote path to be listed. Leave it empty if its the current path.
bstrLocalFile	Specifies a local file name where the listing can be saved.
bstrFormat	Can be used to format the listing. If left empty, a raw listing will be returned. You can specify %NAME, %DATE, and %SIZE as return values in a string.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Host = "ftp.cuteftp.com/pub"
MySite.Connect
MySite.GetList "", "c:\temp_list.txt" 'saves a raw listing for the default
path to the file temp_list.txt
MsgBox MySite.GetResult 'retrieves and displays the listing
```

Example 2

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Host = "ftp.cuteftp.com"
MySite.Connect
MySite.GetList "/pub", "", "FILE NAME:   %NAME" 'goes to pub folder, doesn't
save the listing to file, and formats it as shown
MsgBox MySite.GetResult 'retrieves and displays the listing
```

Example 3

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Host = "ftp.cuteftp.com"
MySite.Connect
MySite.GetList "/pub", "", "NAME= %NAME SIZE= %SIZE DATE= %DATE" 'goes to pub
folder, doesn't save the listing to file, and formats it as shown
MsgBox MySite.GetResult 'retrieves and displays the listing
```

Note

GetResult is an optional method used only with GetList. It simply retrieves the data. Without GetResult the data is written to the buffer, or if defined in the parameters, the data is written to the log.

Properties

Finding a property alphabetically

Here you can link to any property listed in the Help file.

ClearDataConnection
CombSupport
DataChannel
Delay
ErrorDescription
FileSize
FileTimeModified
FileType
HomeDir
Host
IgnoreLinks
IsConnected
LocalFilterInclude
LocalFolder
Log
Login
MaxConnections
Option
Password
Port
Protocol
ProxyInfo
Recursive
RemoteFilterExclude
RemoteFilterInclude
RemoteFolder
SocksInfo
Speed
Status
TimeElapsed
TimeLeft
TotalSize
TransferredSize
TransferType
UseProxy

Finding a property by category

You can find properties here listed by:

- Connection properties
- Read-only connection properties
- Transfer properties
- Read-only transfer properties
- Filter properties

Connection properties

Protocol
Host
Login
Password
Port
UseProxy
ProxyInfo
SocksInfo
DataChannel
ClearDataConnection
ClearCommandChannel
Retries
Delay
Links
LocalFolder
RemoteFolder
MaxConnections
Option

Read-only connection properties

IsConnected
HomeDir
ErrorDescription
Log

Transfer properties

TransferType
Recursive
LocalFolder
IgnoreLinks
RemoteFolder
Option
AutoRename

Read-only transfer properties

AsyncTaskNumber
Speed
Combsupport
TimeLeft
IsPending
TimeElapsed
Status
FileType
RestSupport
FileSize
TotalSize
FileTimeModified
TransferredSize

Filter properties

LocalFilterInclude
RemoteFilterInclude
LocalFilterExclude
RemoteFilterExclude
RemoteSiteFilter

Connection Properties

Setting protocols
Description
Use the **Protocol** property to set or retrieve the value for the protocol type.

Syntax

```
String Object.Protocol
```

Parameters

"FTP"	File Transfer Protocol
"FTPS"	FTP using SSL in explicit mode (standard port 21)
"FTPS_IMPLICIT"	Extension of FTP using SSL -- FTP using SSL in implicit mode

	(specific port)
"SFTP"	Secure protocol based on SSH2
"FTP_SKEY_MD4"	Secure one time password login using MD4
"FTP_SKEY_MD5"	Secure one time password login using MD5
"HTTP"	Hypertext Transfer Protocol
"HTTPS"	HTTP with SSL

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Protocol = "FTPS_IMPLICIT "
MySite.Host = "ftp.ftp.net"
MySite.Port = 990
MySite.Login = "username"
MySite.Password = "password"
MySite.Connect
```

Note

You can find more options for using SSL in ClearDataChannel.

Setting a host address for a connection

Description

Use the **Host** property to set or retrieve the value for the host name of a site when you are connecting.

Syntax

```
int Object.Host
```

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Protocol = "FTP"
MySite.Host = "ftp.ftp.net"
MySite.Port = 21
MySite.Login = "username"
MySite.Password = "password"
```

Setting your user name

Description

Use the **Login** property to set or retrieve the value for the user name (login) you use to connect.

Syntax

```
String Object.Login
```

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Protocol = "FTP"
MySite.Host = "ftp.cuteftp.com"
MySite.Login = "username"
```

```
MySite.Password = "password"
MySite.Connect
```

Setting the password for a connection

Description

Use the **Password** property to set or retrieve the value for the password you use to connect.

Syntax

```
String Object.Password
```

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Protocol = "FTP"
MySite.Host = "ftp.ftp.net"
Object.Login = "username"
MySite.Password = "password"
MySite.Connect
```

Setting the port for a connection

Description

Use the **Port** property to set or retrieve the value for the Port on the server when you connect.

Syntax

```
int Object.Port
```

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Protocol = "FTP"
MySite.Host = "ftp.cuteftp.com"
MySite.Port = 21
MySite.Login = "username"
MySite.Password = "password"
MySite.Connect
```

Transferring files in an unencrypted data channel

Description

Use the **ClearDataConnection** property to specify whether the data channel should be encrypted or not when using FTP over SSL.

Syntax

```
long Object.ClearDataConnection
```

Parameters

True	Your login is encrypted but data transfers are not. This is the default.
False	Your login and data transfers are encrypted.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Protocol = "FTPS"
MySite.ClearDataConnection = false ‘encrypt FTP data channel
```

Notes

- The default is true (encrypted), so only set this property if you wish to log in securely but transfer data in the clear.
- This property will only work if the remote server supports SSL and clear data channel connections (PROT C command according to RFC 2228) and you have specified FTPS as the protocol type.

Sending unencrypted commands over an SSL connection

Description

Use the **ClearCommandChannel** property to specify that the command channel should not be encrypted over SSL.

Syntax

```
long Object.ClearCommandChannel
```

Parameters

True	Your login is encrypted but as data transfers start, commands are not encrypted.
False	Your login and subsequent commands are encrypted. This is the default

Example

```
MySite.Host = "host"
MySite.Protocol = "FTPS"
MySite.ClearCommandChannel = true '<-- CCC will be sent prior to the first
data connection operation.
MySite.Connect
```

Notes

- The default is false (encrypted), so only set this property if you wish to log in securely but send subsequent commands in the clear.
- This property will only work if the remote server supports SSL and clear command channel connections (according to RFC 2228) and you have specified FTPS as the protocol type.

Connecting through a proxy or SOCKS server

Description

Use the **UseProxy** property to retrieve or set the value for the type of SOCKS or proxy server that is being (or should be) used. Since OFF is the default, you don't need UseProxy for regular connections which do not pass through proxy or SOCKS servers.

Syntax

```
String Object.UseProxy
```

Parameters

"OFF"	Direct connection without any socks and proxy
"SOCKS"	SOCKS server only - the user must specify the SOCKS parameters by setting the SocksInfo property

"PROXY"	Proxy server only - the user must specify the proxy server parameters by setting the ProxyInfo property
"BOTH"	Use both SOCKS and proxy - the user should specify the appropriate information for both the proxy and the socks server with ProxyInfo and SocksInfo.

Example

```
Object.UseProxy = "SOCKS" 'specify that socks will be used, then configure
SocksInfo
Object.SocksInfo = "socks5://globalscape.com:1080"
'Rest of connection code follows...
```

Setting and retrieving proxy server configurations

Description

Use the **ProxyInfo** property to set or retrieve FTP and HTTP proxy server configurations. Do not use this function if you do not connect through a proxy server.

Authentication Parameters

"ftp://proxyusername:proxypassword@proxyhostname:proxyport"	For FTP proxies that require authentication.
"http://proxyusername:proxypassword@proxyhostname:proxyport"	For HTTP proxies that require authentication.
"ftp:// proxyhostname:proxyport"	For FTP proxies that don't require Authentication.
"http:// proxyhostname:proxyport"	For HTTP proxies that don't require Authentication.

Additional parameters

"proxyusername"	The user name for login to the proxy server
"proxypassword"	The password for login to the proxy server
"proxyhostname"	The proxy server address
"proxyport"	The proxy server connection port

Syntax

```
String Object.ProxyInfo
```

Example 1

```
MySite.ProxyInfo = http://globalscape.com:8000 'use http proxy without
authorization
```

Example 2

```
MySite.ProxyInfo = ftp://joeuser:maypass@globalscape.com:21 'use ftp proxy
with authorization for user "joesuser" & password "mypass"
```

Example 3

```
str = MySite.SocksInfo 'retrieve the current value of SocksInfo (empty if
none)
MsgBox str 'now display it
```

Setting or retrieving values for SOCKS servers

Description

Use the **SocksInfo** property to set or retrieve values for SOCKS4 or SOCKS5 servers. (SOCKS is a protocol for a TCP proxy across firewalls.)

Syntax

```
String Object.SocksInfo
```

Authentication Parameters

"socks4://socksusername:sockspassword@sockshostname:socksport"	For SOCKS4 servers that require authentication.
"socks5://socksusername:sockspassword@sockshostname:socksport"	For SOCKS5 servers that require authentication.
"socks4:// sockshostname:socksport"	For SOCKS4 servers that do not require authentication.
"socks5:// sockshostname:socksport"	For SOCKS5 servers that do not require authentication.

Additional parameters

"socksusername"	The user name or log in name to the SOCKS server.
"sockspassword"	The password for the SOCKS server.
"sockshostname"	The address and port for the SOCKS server.

Example 1

```
MySite.SocksInfo = "socks4//globalscape.com:1080" 'use socks4 without
authorization
```

Example 2

```
MySite.SocksInfo = "socks5://joeuser:mypass@globalscape.com:1080" 'use socks5
with authorization for user "joeuser" with password "mypass"
```

Example 3

```
str = MySite.SocksInfo 'retrieve the current value of SocksInfo (empty if
none)
MsgBox str 'now display it
```

Choosing ASCII, binary, or auto transfer types

Description

Use the **TransferType** property to retrieve or set the value for the way the Transfer Engine should transfer files (ASCII, binary or auto).

Syntax

```
String Object.TransferType
```

Parameters

"ASCII"	All files should be transferred in ASCII mode
"BINARY"	All files should be transferred in BINARY mode
"AUTO" (default)	The TE will reference an internal list editable from CuteFTP's **Global Options** to determine the proper transfer type for that particular file. For example, if the ASCII list contains a filter mask of "txt" then all files with extension ".txt" will be transferred in ASCII. If a file doesn't correspond to any mask in this list then it will be transferred in binary mode.

Example

```
Object.TransferType = "AUTO" 'the Transfer Engine will use the shell's default settings to establish whether the transfer should occur in binary or ASCII. 'write file transfer code next...
```

Choosing a PASV or PORT connection

Description

Use the **DataChannel** property to set or retrieve values for the method in which the data port is established for the data channel (PASV or PORT). The default is PORT.

Syntax

```
String Object.DataChannel
```

Parameters

"PORT"	The client specifies what port to use for the data connection
"PASV"	Lets the server specify the port used for the data connection port
"DEFAULT"	Uses the method defined in **Global Options**

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Host = "ftp://ftp.cuteftp.com"
MySite.useProxy = "off"
MySite.DataChannel = "PASV"
MySite.RemoteCommand("LIST")
MySite.DataChannel = "PORT"
MySite.RemoteCommand("LIST")
MsgBox MySite.Log
```

Setting the number of retry attempts

Description

Use the **Retries** property to retrieve or set the value for the maximum number of attempts the Transfer Engine should make to connect to a remote host.

Syntax

```
int Object.Retries
```

Example

```
MySite.Retries = 10
```

Note

Some older versions of CuteFTP Pro help files state that you can use the **Retries** property with transfers. That is incorrect, **Retries** only works with connections.

Setting the delay between retries

Description

Use the **Delay** property to set the delay between connection retry attempts (in seconds).

Syntax

```
int Object.Delay
```

Example

```
MySite.Delay = 90
```

Note

Some older versions of the CuteFTP Pro help file state that you can use the **Delay** property with transfers. That is incorrect, you can only use **Delay** with connections.

Resolving or retrieving links

Description

Use the **Links** property to both set and retrieve values for working with links on a remote server.

Syntax

```
String Object.Links
```

Parameters

"**Resolve**"	The TE will attempt to resolve the link prior to transfer.
"**GetAsFile**"	The TE will assume it's a file and transfer as is (This is used to avoid endless loops in a large multi-directory transfer).

Example

```
MySite.Links = "Resolve"
```

Ignoring links

Description

Use the **IgnoreLinks** property when working with Unix servers. Use **IgnoreLinks** to set or retrieve the value that determines whether or not you want to bypass links.

Syntax

```
Bool Object.IgnoreLinks
```

Parameters

"True"	Skip symlinks during synchronization
"False"	Handle symlinks during synchronization as files or folders

Example

```
Mysite.IgnoreLinks = "False"
```

Related topics

Other properties

Choosing a local folder

Description

Use the **LocalFolder** property to retrieve or set the current folder on your local hard disk. You can use relative or absolute paths.

Syntax

```
String Object.LocalFolder
```

Example

```
Object.LocalFolder = "c:\temp"
```

Choosing a remote folder

Description

Use the **RemoteFolder** property to retrieve or set the current remote folder. You can use absolute or relative paths.

Syntax

```
String Object.RemoteFolder
```

Example

```
'Connect to site following previous samples
MySite.RemoteFolder = "/pub/cuteftp" 'Changes the remote folder to
/pub/cuteftp (absolute paths used)
'subsequently
MySite.RemoteFolder = "test" 'Changes to the folder called "test" located in
"/pub/cuteftp", using relative paths.
```

Using the auto-renaming feature in transfers

Description

Use the **AutoRename** property to set or retrieve the renaming values prior to the transfer of a file or folder. To configure the auto-rename rules open the CuteFTP Professional interface and go to **Tools > Global Options > Transfer > Rename Rules**.

Syntax

```
String MySite.AutoRename
```

Parameters

"ON"	AutoRename feature turned on
"OFF"	AutoRename feature turned off

Example

```
MySite.AutoRename = "ON"
```

Setting the maximum number of connections

Description

Use the **MaxConnections** property to set or retrieve the most connections the script is allowed to open.

Syntax

```
long Object.MaxConnections
```

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.MaxConnections = 1 'restrict connections to 1
```

Choosing to include or exclude subfolders in a task

Description

Use the **Recursive** property to set or retrieve the value that determines whether or not subfolders will be included in a transfer task. This property is used often in synchronization scripts.

Syntax

```
Bool Object.Recursive
```

Parameters

"True"	Include subfolders
"False"	Do not include subfolders

Example

```
Mysite.Recursive = "True"
```

Setting or retrieving other settings

Description

Use the **Option** property to set or retrieve various settings, such as Auto-rename, include folder names when filtering, cache invalidation, error handling, cleaning up of asynchronous tasks, and to auto-close prompts.

Syntax

```
String Object.Option("[option name]") = true | false
```

Parameters

"ThrowError"	(defaults to True) - TE COM will call AtlReportError (showing a VB runtime error message box) on connection/transfer/IO/other error which will terminate script execution. If set to False, then if some transfer method fails

	the script will continue to the next command.
"AutoRename"	(defaults to False) - The same as the AutoRename property.
"CleanupAsync"	(defaults to True) - All task IDs accumulated by previous Async methods will be lost. If set to False, all task IDs will be added to the ones created by previous Async methods.
"InvalidateCache"	(defaults to True) - Remove cached file (containing listings) before uploading, downloading, renaming or deleting files. If false, then don't remove cached listing. You can obtain the specific file information when needed by using one of the file property methods. InvalidateCache optimizes LIST traffic.
"FilterDirs"	(defaults to True) - Apply filters to folder names. If set to False, then don't apply.
"AutoCloseMethod"	(default to 0) 1 - auto accept, 2 - auto reject, 0 – Don't accept (default). This handles hidden prompts (such as SSL Accept Cert Prompt when running a script while not logged in).
"AutoCloseDelay"	(default value is 60 seconds). Time delay before CuteFTP should perform the action specified by the AutoCloseMethod option.

Example 1

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Host = "ftp://ftp.cuteftp.com"
MySite.Option("FilterDirs")=False          'don't filter folder names
MySite.LocalFilterExclude= "TDImon; *.txt; *.vbs"
                                           'now set the item names to exclude
MySite.Upload "c:\test"
                                           'uploads all of test including sub
dirs.
                                           'Even sub dirs that match the
filter, such as TDIMon.
                                           'If I had left FilterDirs=True,
then the folter
                                           'TDImond would not have been
uploaded.
```

Example 2

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Host = "ftp://ftp.somestrangedomain.com"
MySite.Option("ThrowError") = false
if not CBool(MySite.Connect) then
  MsgBox "Error : " & MySite.ErrorDescription
end if
```

Example 3

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Host = "ftps://ftp.asecuredomain.com"
MySite.Option("AutoCloseMethod") = 1  'will auto accept cert and continue
MySite.Option("AutoCloseDelay") = 5   'wait 5 seconds before accepting
MySite.Connect                        'now connect to the secure site. The TE
will accept the server's SSL cert after 5 seconds and continue executing the
rest of the script.
```

Read-only Properties

Checking for a connection

Description

Use the **IsConnected** property to indicate whether or not you are presently connected to the remote site.

Syntax

```
Boolean Object.IsConnected
```

Parameters

"True"	The Transfer Engine is currently connected to the server.
"False"	The Transfer Engine is not currently connected to the server.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
If Cbool(MySite.IsConnected) Then
MsgBox "Connected to server:" + MySite.Host
End if
```

Note

You can set how long the Transfer Engine will leave the data connection open after a completed transfer. In the CuteFTP GUI, go to **Tools > Global Options > Transfer Settings** and set the time in **Close the file transfer *n* seconds after the transfer is completed**.

Retrieving the number of Asynchronous tasks

Description

Use the **AsyncTaskNumber** property to return the number of tasks created by the UploadAsync, DownloadAsync and TransferURLAsync methods.

Syntax

```
Long Object.AsyncTaskNumber
```

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user, password,
etc.
MySite.Option("CleanupAsync") = False
MySite.DownloadAsync "/path/*.*"
MsgBox "task number created" & MySite.AsyncTaskNumber
```

Notes

You must set MySite.Option("CleanupAsync") = False so that all asynchronous tasks are counted. Otherwise, only the last asynchronous task launched will be referenced when calling the AsyncTaskNumber property.

Checking if a server supports multi-part uploads

Description

Use the **CombSupport** property to check if the server supports the COMB (multi-part upload) command.

Syntax

```
Boolean Object.CombSupport
```

Return values

-1	Server supports COMB
0	Server does not support COMB

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user, password, etc.
If (not MySite.CombSupport) Then
MsgBox "This server doesn't support the COMB command!"
End if
```

Retrieving error descriptions

Description

Use the **ErrorDescription** property to get the string describing the last error condition. It may consist of some messages taken from the transfer log.

Syntax

```
String Object.ErrorDescription (long taskIdx)
```

Parameter

TaskIdx	This is a task index in the array of tasks created by the various asynchronous methods. The default value of taskIdx is - 1 which specifies all the asynchronous tasks in array. TaskIdx should be between 0 and AsyncTaskNumber minus one.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Option("ThrowError") = false 'disable ATL exceptions
if cbool(MySite.Connect) then
MsgBox "Connected OK"
else
MsgBox "Error! " & MySite.ErrorDescription
```

Retrieving the server's home directory

Description

Use the **HomeDir** property to return a string value containing the name of the server's home directory.

Syntax

```
String Object.HomeDir
```

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user, password, etc.
MsgBox "Home dir for server " + MySite.Host + " is: " + MySite.HomeDir
```

Checking if a transfer is active

Description

Use the **IsPending** property to determine whether a transfer is active or if it is already finished with success or error. This can be useful in combination with async commands or during an interactive script. The **IsPending** property will return a value of either true or false.

Syntax

```
Boolean Object.IsPending(long taskIdx)
```

Parameter

TaskIdx	This is a task index in the array of tasks created by the various asynchronous methods. [0.. AsyncTaskNumber minus one, or - 1 (last asynchronous task started)] This has a default value (if nothing is specified) of ALL tasks. Therefore, IsPending will return true if any task is still pending. It will return false if none are pending.

Example 1

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Regular connection code here:
MySite.Download inbound/*.*", "c:\temp"
If CBool(MySite.IsPending) Then
MsgBox "task is in working state" + MySite.Host
End if
```

Example 2

Here is another example that checks an asynchronous transfer and will return various transfer progress properties of each transfer while IsPending is true.

Warning: If you copy and paste this code, be aware that line breaks may be inserted into the code.

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Protocol = "FTP"
MySite.Port = "21"
MySite.Host = "ftp.url"
MySite.login = "loginname"
MySite.Password = "your pass"
MySite.Connect
MySite.Option("CleanupAsync") = False
' line break
strNL = (Chr(13) & Chr(10))
MySite.MaxConnections = 3
MySite.Option("ThrowError") = false
MySite.DownloadAsync "inbound/*.*", "c:\temp"
bContinue = true
while CBool(MySite.IsPending) and bContinue
str = "LOOP, Total: " & MySite.AsyncTaskNumber & strNL
```

```
for i = 0 to MySite.AsyncTaskNumber - 1 step 1
str = str & i & ": size: " & MySite.TransferredSize(i) & ", speed: " &
MySite.Speed(i) & ", time left: " & MySite.TimeLeft(i) & ", status: " &
MySite.Status(i) & strNL
next
str = str & "YES - continue loop, NO - stop tasks, CANCEL - exit loop"
nUserChoise = MsgBox(str, vbYesNocancel) 'press YES many time to see transfer
progresses
if nUserChoise = vbCancel then
bContinue = false
elseif nUserChoise = vbNO then
MySite.AbortAsync 'abort all tasks
bContinue = false
end if
wend
str = "DONE, Total: " & MySite.AsyncTaskNumber & strNL
for i = 0 to MySite.AsyncTaskNumber - 1 step 1
str = str & i & ": size: " & MySite.TransferredSize(i) & ", speed: " &
MySite.Speed(i) & ", time left: " & MySite.TimeLeft(i) & ", status: " &
MySite.Status(i) & strNL
next
MsgBox str
```

Retrieving the status of a transfer

Description

Use the **Status** property to determine whether a transfer is active or it is already finished with success or error. This can be useful in combination with asynchronous commands or during an interactive script. The **Status** property will return a string.

Syntax

String Object.Status (long taskIdx)

Parameters

TaskIdx	This is the task index in the array of tasks created by asynchronous methods [0.. AsyncTaskNumber or - 1 (last started)]

Return Values

"WAIT"	Transfer action invoked but not initiated yet (followed by connecting)
"CANCELED"	Active transfer canceled by user
"FINISHED"	Transfer completed
"ERROR"	Error in transfer (any possible client or server error)
"SUSPENDED"	Added to queue but no transfer initiated yet
"SCHEDULED"	Item is scheduled for future transfer
"BLOCKED"	An internal status used for navigation and does not pertain to active or pending transfers. You can cancel any transfer requests in BLOCKED status

	without any ill effects.
"CHILDWAIT"	Condition when transfer item is waiting for a dependant item to finish transferring (*)
"SKIPPED"	Transfer skipped by user or automatically per overwrite rules
"CONNECTING"	Connecting to server (status right after WAIT)
"CANCELLING"	Cancel initiated but not completely stopped yet
"WORKING"	After connecting but before transferring. Could be opening data connection, or setting REST params, etc.
"TRANSFERRING"	File transfer in progress
"UNKNOWN"	Another string was returned other than one of the above. The string was unrecognized.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
If not CBool(MySite.IsPending) Then
MsgBox "Task done, final status is " + MySite.Status
End if
```

Checking if a server can resume downloads

Description

Use the **RestSupport** property to check if the server supports the REST (resume download) command.

Syntax

```
Boolean Object.RestSupport
```

Return values

-1	Server supports REST.
0	Server does not support REST.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
'Initialize all necessary fields for MySite : host name, user, password, etc.
If (not cbool(MySite.RestSupport)) Then
MsgBox "This server doesn't support the REST command"
End if
```

Retrieving a log as a string

Description

Use the **Log** property to return the entire log as a string which can be handled separately in your VB application or VB script.

Syntax

```
String Object.Log (long taskIdx)
```

Parameters

TaskIdx	This is the task index in the array of tasks created by asynchronous methods [0.. AsyncTaskNumber minus one, or - 1 (last asynchronous task started)] It has a default value of -1 = current task.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Host = "ftp://ftp.cuteftp.com"
MySite.Connect
MsgBox MySite.Log
```

Checking transfer progress

Description

Use the **TotalSize, TransferredSize, Speed, TimeLeft,** and **TimeElapsed** properties to determine a transfer's progress and various other aspects of the transfer.

Syntax

```
long Object.TotalSize (long taskIdx) = size in bytes
long Object.TransferredSize (long taskIdx) = size in bytes
long Object.Speed (long taskIdx) = bytes/second
long Object.TimeLeft (long taskIdx) = seconds
long Object.TimeElapsed (long taskIdx) =seconds
```

Parameters

TaskIdx	This is the task index in the array of tasks created by asynchronous methods [0.. AsyncTaskNumber minus one, or - 1 (last asynchronous task started)] It has a default value of - 1 = current task.

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
MySite.Host = host 'set all relavant host, login, pass, etc. properties
MySite.Download "aaa", "c:\temp"
MsgBox "Size: " & MySite.TotalSize & ", speed: " & MySite.Speed & ", time: " &
MySite.TimeElapsed
```

Retrieving file information

Description

Use the **FileType, FileSize,** and **FileTimeModified** properties to retrieve information about a particular item.

Syntax

```
long Object.FileType
long Object.FileSize
string Object.FileTimeModified
```

Return values

For FileType:

1	Directory
2	File
3	Link

For FileSize:
the size in bytes of the file

For FileTimeModfied:
mm/dd/yyyy hh:mm:ss

Example

```
Set fserv = CreateObject("CuteFTPPro.TEConnection")
strRemote = "somefilename.ext"
if cbool(fserv.RemoteExists(strRemote)) then
MsgBox "Remote file/folder info: Type=" & fserv.FileType & ", Size=" &
fserv.fileSize & ", Date=" & fserv.FileTimeModified & strNL & "(type:1-dir,2-
file,3-link)"
else
MsgBox "Error! " & fserv.ErrorDescription
End If
```

Filter Properties

Including local files in lists and transfers

Description

Use the **LocalFilterInclude** property to set or retrieve the values for including local files or folders in transfers and listings. The string values may contain wildcards and you can add multiple filters by separating them with a semicolon "**;**".

Syntax

```
String Object.LocalFilterInclude
```

Example

```
MySite.LocalFilterInclude = "*.jpg;*.gif"
```

Note

If filter causes no file to be transferred then no folders will be created.

Excluding local files from lists and transfers

Description

Use the **LocalFilterExclude** property to set and retrieve the values for excluding local files or folders from transfers and listings. The string values may contain wildcards and you can add multiple filters by separating them with a semicolon "**;**".

Syntax

```
String Object.LocalFilterExclude
```

Example

```
MySite.LocalFilterExclude = "*.mp3;*.jpg"
```

Note

If filter causes no file to be transferred then no folders will be created.

Excluding remote files from lists and transfers

Description

Use the **RemoteFilterExclude** property to set and retrieve the values for excluding remote files or folders from transfers and listings. The string values may contain wildcards and you can add multiple filters by separating them with a semicolon "**;**".

Syntax

```
String Object.RemoteFilterExclude
```

Example

```
MySite.RemoteFilterExclude = "*.txt;*.swp"
MySite.Download "*.*" ' will download all files except those with extensions
of *.txt and *.swp
```

Note

If filter causes no file to be transferred then no folders will be created.

Including remote files in lists and transfers

Description

Use the **RemoteFilterInclude** property to set or retrieve the values used to include remote files or folders in transfers and listings. The string values may contain wildcards and you can add multiple filters by separating them with a semicolon ";".

Syntax

```
String Object.RemoteFilterInclude
```

Example

```
MySite.RemoteFilterInlcude = "*.jpg;*.gif"
```

Note

If filter causes no file to be transferred then no folders will be created.

Adding filters to the LIST command

Description

Use the **RemoteSiteFilter** property to specify a string which will be used as a filter parameter by the LIST command.

Syntax

```
string Object. RemoteSiteFilter
```

Example

```
Set MySite = CreateObject("CuteFTPPro.TEConnection")
```

```
MySite. RemoteSiteFilter = "-l"
'This will send a "List -l" command to the server.
```

Note

These optional parameters to the list command are basically parameters passed to the UNIX ls program running on the server (if applicable):

-l	long listing
-a	show hidden files
-t	sort by timestamp
-R	recursive

There are various additional parameters. Refer to a UNIX manual or the Web for more details. You may want to check http://www.bsdi.com/bsdi-man.

Troubleshooting

Disabling prompts

To run the Transfer Engine with prompts disabled

Launch the TE manually and supply the –noprompts parameter.

Example

1. From the **Start** menu choose **Run**.
2. Type: **"C:\Program Files\Globalscape\CuteFTP Professional\TE\ftpte.exe" –noprompts**.
3. Click **OK**.

Warning: Suppressing prompts may cause lockups if no default action is available for the event in question, or if the prompt was produced by an error. When running the TE in this mode using Windows scheduler from a different user account, the TE will not be accessible from its COM or GUI interface and can only be unloaded via the Task Manager.

The best way to avoid prompts is to properly configure the TE ahead of time to cover all possible prompt scenarios, such as overwrite conditions, or SSL server certificate acceptance dialogs. For example, set your overwrite actions under CuteFTP's global settings to Always Overwrite to avoid any overwrite prompts, or pre-define various rules to cover various overwrite scenarios. You can then export those rules as registry entries and import them onto the target machine.

You can also tell the TE to auto-accept SSL or SSH certificates using the AutoCloseMethod parameter of the Option method. Also refer to the following trouble shooting topic regarding running the TE while not logged in.

High memory usage

If you leave the Transfer Engine running for extended lengths of time, it may begin to consume large amounts of memory. Set CuteFTP to remove successful items from the queue to avoid this problem.

When you run many tasks, or schedule many recurring tasks, the successful tasks build up in the queue, unless you set CuteFTP to automatically remove them. Each finished queue item takes about 500 bytes of memory until the Transfer Engine is closed.

To remove successful items from the queue

1. Open the CuteFTP 6 Professional user interface.
2. On the menu bar click **Tools > Global Options**.
3. Click **Navigation Settings**.
4. Select the **Remove successfully finished items from the queue automatically** check box.
5. Click **OK**. Successful items will be removed from the queue, even if you use the Transfer Engine without the user interface.

To remove successful items from the queue in a script

- Use the TECommand method with the DELETEFINISHED or DELETEALL property

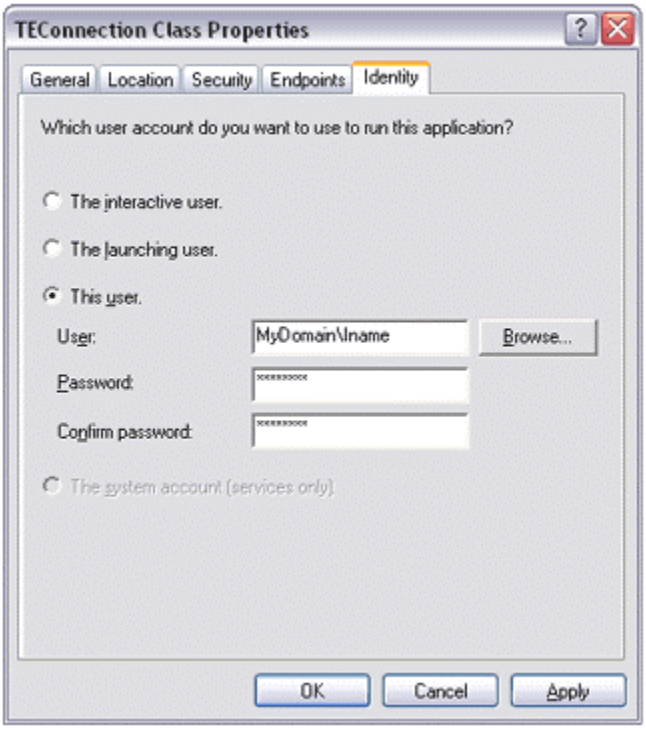
Setting the TE to run without a user present

The Transfer Engine is a process, not a service. However, you can execute scripts while the system is not logged in if you properly configure Windows DCOM configuration for the Transfer Engine. Carefully read and follow the steps below. A sample script is provided.

To configure the TE under the DCOM configuration applet

1. Click **Start**.
2. Click **Run**.
3. Type **DCOMCNFG** and click **OK**.
4. Select **TE** from the list.
5. Click **Properties**.
6. Click the **Identity** tab.
7. You must select **This user:** and provide a valid login name and password. *It must be the same user as established under your task scheduler's* **run as** property (if applicable), so that the scheduler service can call the COM object as a user. This will enable the Transfer Engine to run with the user settings, and if necessary, access pre-established SSL certificates (for secure connections).

Screenshot of Identity tab



Note

If the system is currently logged in when the TE is run, it will run as the user which you specified under DCOMCNFG. You will not see the TE icon in the system tray when running CuteFTP's main interface. Also, the hard coded user's settings are employed when the TE is run, not those of the currently logged in user. *This won't be an issue if you are running the TE while the system is not logged in.*

You should test your script from the command line while logged in. You can also write to the event viewer or a local text file to debug a script when run on system startup if necessary. Here is a sample script which includes event viewer logging of transactions.

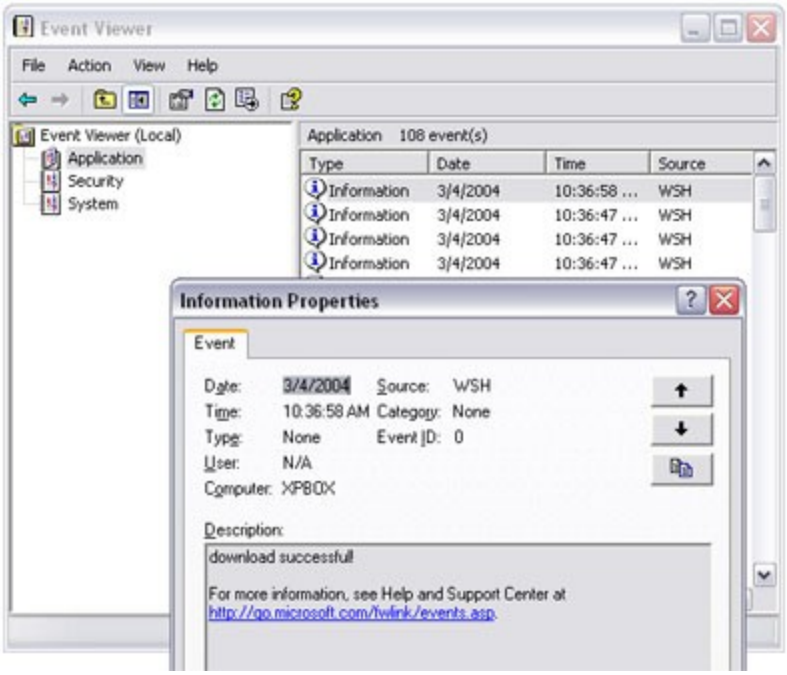
IMPORTANT

Your script should not contain Message Box functions or any other function that requires user input or shows a window. Windows cannot display these prompts while the system is not logged in. Set your overwrite actions under CuteFTP's global options to "Always Overwrite" to avoid any overwrite prompts, or pre-define various rules to cover various overwrite scenarios.

```
Const EVENT_SUCCESS = 0
Set objShell = Wscript.CreateObject("Wscript.Shell")
objShell.LogEvent EVENT_SUCCESS,"AT Loaded me"
Set MySite= CreateObject("CuteFTPPro.TEConnection")
MySite.Option ("ThrowError") = False
MySite.Host = "ftp://anonymous:user@user.com@ftp.globalscape.com/pub/cuteftp"
objShell.LogEvent EVENT_SUCCESS,"vars set"
MySite.Connect
objShell.LogEvent EVENT_SUCCESS, "connected"
MySite.Download "index.txt", "c:\temp"
objShell.LogEvent EVENT_SUCCESS, "downloaded"
```

```
MySite.Disconnect
MySite.Close
```

Screenshot of Event Viewer Application Log showing successful download



Automatically encrypting and compressing transfers

To automatically encrypt and compress files before transfer, you will need both CuteZIP and CuteFTP Professional. You can also compress and encrypt files after transferring them by using GlobalSCAPE Secure FTP Server's Custom Site Commands. The benefits of encrypting files prior or post transfer depends on the circumstances and level of trust for the particular host.

In the example script below, an entire folder (including sub-folders) is compressed, encrypted (using Twofish 128 bit encryption) and then transferred via regular FTP to an FTP server. Since the archived file is encrypted, there is no need to connect using SSL, OTP, or SSH2 unless you wished to also protect the login process.

Example

```
Dim WshShell, MySite, Return
Set WshShell = CreateObject("WScript.Shell") 'Window's Scripting Host shell
object
'next line calls the run method of the WSH shell object. It returns true once
CuteZIP does its thing.
'The complete command line instructions for CuteZIP are located here.
If Return = WshShell.run ("c:\progra~1\global~1\CuteZIP\cutezip.exe -c -p12345
c:\archive c:\temp", 0, true) Then
  Set MySite = CreateObject("CuteFTPPro.TEConnection")
  MySite.Option ("ThrowError") = True
  MySite.Host = "ftp://user:pass@myftpsite.com" 'one of the ways to connect
using the TE
```

```
  MySite.Connect
  MySite.Upload "c:\archive.zip" 'upload the new archive, then check to see if
it made it up to the server.
  if not CBool(MySite.RemoteExists("\archive.zip")) then
     MsgBox "Failed to Upload, Exiting!"
  Else
     MsgBox "Success!"
  End If
  MySite.Disconnect
  MySite.Close
Else
MsgBox "Compression and Encryption Failed, Exiting!"
End If
```

Note

You can optionally protect the FTP login by connecting with SSL, SSH2 or OTP using CuteFTP 6 Professional's Transfer Engine (GlobalSCAPE's Secure FTP Server 2 supports SSL, OTP, and SSH2 logins). Use the Protocol property to set the connection type prior to calling MySite.Host and MySite.Connect.

Running the Transfer Engine from an SQL job

You can use SGL function sp_OACreate to create a TEConnection object. SQL runs ftpte and hides all windows. SQL uses a special instance of ftpte exe, which cannot prompt for user name and password.

Specify a local download path or filename. While optional, it's a good idea to make sure that it is going where you intend. Also, make sure that the local path has the appropriate NTFS permissions to allow the TE to have full control.

For scripted or scheduled tasks use an otherwise unused account for scheduled TE tasks. Once the CuteFTP TE (transfer engine) is started, another instance cannot be created. Windows treats the logged-in user differently than the logged-out-user and DCOM errors may occur. For instance, if a recurring background process has already run before the user logs in, once the user logs in and the background process starts again, a DCOM error will occur.

Example transfer script

```
DECLARE @property varchar(255)
DECLARE @object int
DECLARE @hr int
DECLARE @src varchar(255), @desc varchar(255)
PRINT '---start'
EXEC @hr = sp_OACreate 'CuteFTPPro.TEConnection', @object OUT
EXEC @hr = sp_OASetProperty @object, 'Host', 'ftp.mysite.com'
EXEC @hr = sp_OAGetProperty @object, 'Host', @property OUT
PRINT @property
EXEC @hr = sp_OAGetProperty @object, 'Login', @property OUT
PRINT @property
EXEC @hr = sp_OAGetProperty @object, 'Protocol', @property OUT
PRINT @property
EXEC @hr = sp_OAMethod @object, 'Connect'
EXEC @hr = sp_OAMethod @object, 'download', NULL, '/pub', "e:/eee"
```

```
EXEC @hr = sp_OAGetProperty @object, 'Log', @property OUT
PRINT @property
EXEC @hr = sp_OADestroy @object
PRINT '---finish'
```

Notes

- SQL cannot use an already running instance of ftpte exe. The CuteFTP GUI cannot communicate with an ftpte instance created by SQL.
- When a scheduled task runs as user X while user X is logged in, the profile path will be set to that user's directory (C:\Documents and Settings\X\). However, when a scheduled task runs as user X while user X is *not* logged in, the profile path will be set to the default user directory (C:\Documents and Settings\Default User\). Therefore, the desired SSL certificates from C:\Documents and Settings\X\Application Data\GlobalSCAPE\CuteFTP Professional\6.0\certs.crt should be copied to C:\Documents and Settings\Default User\Application Data\GlobalSCAPE\CuteFTP Professional\6.0\certs.crt
- If you use **UseProxy**, be sure to specify any proxy information. Please see Connecting through a SOCKS or Proxy Server for more information.

Troubleshooting tips

- Add a couple of log lines so that it can write to the log whenever something happens successfully. Add it after the connect line, for instance, so that you can see how far it gets.
- To make sure you are connecting long enough to get data, use the **GetList** method to write a list to a local file.

No timeout when connecting to an unavailable host

If the Transfer Engine's Connection method will not timeout when connecting to a non-existing or temporarily unavailable host try setting the throw error to true. The Connection method does not have a built-in timeout value. It will keep trying to connect indefinitely. Because the event is synchronous, subsequent lines in the script, including conditional statements for determining the connection status of the TE, will never get called.

Here are a few possible workarounds, and help for accepting and rejecting certificates.

Examples:

MySite.Option("AutoCloseMethod") = 2 '1 - auto accept, 2 - auto reject, 0 - default no auto

MySite.Option("AutoCloseDelay") = 12 ' default value is 60 sec

Set Option("AutoCloseMethod") property to 1 or 2 in order for script can continue its processing:

MySite.Option("AutoCloseMethod") = 1 will auto accept cert and continue

MySite.Option("AutoCloseMethod") = 2 will auto reject cert and finish with error

MySite.Option("AutoCloseMethod") = 0 default: will not close prompt on timeout

Timeout strategies for the Wait method

The default timeout value for the **Wait** method is 21,805,184 milliseconds, which is approximately 6 hours. The timeout value is a SIGNED LONG data type, meaning its maximum possible value is 2,147,483,647 milliseconds, which is roughly 596.5 hours or just under 25 days. This is probably enough time for even the slowest transfer.

The **Wait** method supports a "0" timeout value which means "keep waiting forever or until the transfer reaches a state of CANCELED, FINISHED, ERROR, SUSPENDED, SKIPPED, or BLOCKED."

You can also write scripts so that they check the condition of a transfer and if it is still in the "TRANSFERRING" state, to wait on it again.

Three timeout strategies for long transfer tasks

1. Specify a large timeout value.
2. Upon timeout, check the transfer's status.
3. Wait forever.

1) Specify a large timeout value in the script call.

Because the first parameter to the **Wait** method is a task index, this example uses a "-1" which means "current task." For this example, the timeout is set for 10 hours or, 10 * 60 * 60 * 1000 = 36000000 milliseconds.

Example

```
strResult = strataFTP.Wait( -1, 36000000 )
```

2) After a Wait() function has timed out, check the STATUS of the transfer.

In this scenario, use the program (or script) logic to keep trying after a **Wait** times out, if the transfer is still in the TRANSFERRING state.

In other words, your polling for the termination status has timed out, but not necessarily the transfer itself, so you keep going.

In this example, you will wait up to 10 hours for the transfer, and if that times out, you will check the status of the transfer; if it is still TRANSFERRING, you will do it again (please note the last two conditional statements):

Example

```
Do
   strResult = strataFTP.Wait( -1, 36000000 )
Loop While ( strResult <> "CANCELED") and ( strResult <> "FINISHED" ) and
     ( strResult <> "ERROR" ) and ( strResult <> "SKIPPED" ) and
     ( strResult <> "SUSPENDED" ) and ( strResult <> "BLOCKED" )
```

Alternatively, you can take the more positive outlook that we keep going while the transfer task is either WORKING, CONNECTING, or TRANSFERRING:

Example

```
Do
   strResult = strataFTP.Wait( -1, 36000000 )
Loop While ( strResult = "TRANSFERRING") or ( strResult = "WORKING" ) or
     ( strResult = "CONNECTING" )
```

3) Wait forever, or until the transfer reaches some termination point.

Most transfers eventually either FINISH or receive an error from the server but there is a minor chance that the transfer in the queue is perpetually stuck in a "TRANSFERRING" state, so this strategy might be considered a little riskier than the first two.

Example

```
strResult = strataFTP.Wait( -1, 0 )
```

My scheduled scripts no longer run while not logged in

If you have installed the Security Update for Microsoft Data Access Components (MDAC) Security Patch MS03-033 you may have problems using scripts to connect to secure sites.

You can get your scripts to work again by moving your certificate file. The CuteFTP certificate files generally reside in a specific user's folder, for example; **C:\Documents and Settings\jsmith\Application Data\GlobalSCAPE\CuteFTP Professional\6.0\Security\certs.crt**.

Simply move the certs.crt file to the Default User Folder, for example; **C:\Documents and Settings\Default User\Application Data\GlobalSCAPE\CuteFTP Professional\6.0\Security\certs.crt**, and your scripts should now work.

Note
The file name for Security Patch MS03-033 is Q823718_MDAC_SecurityPatch.exe.

Scripting technical support

- Due to the wide range of scripts that CuteFTP Professional is able to accommodate we are unable to offer technical support on individual scripts, other than what is available in the help files and online Knowledge Base.
- If you are having trouble with your script, try to perform the desired action manually, using the CuteFTP GUI. If you cannot, then troubleshoot that problem first and then re-try your script.
- If you are able to perform the desired actions, and in the desired sequence when using the GUI, then the problem is not with CuteFTP or the FTP Server. The next thing to do is to troubleshoot your script line by line.

Handling SSL certificates (When running a script while not logged in)

Setting up the TE to connect to an FTP server over SSL (via scripting) involves several steps. Follow the instructions below carefully.

You must know whether the TE will require the use of server certificates only, or both server and client certificates.

Server Certificates

Most FTP SSL servers will provide a certificate to the client during authentication. The certificate proves the server's legitimacy to the client. The server's certificate must be added the client's Trusted Certificate store prior to any connection, otherwise the TE will lock up when trying to authenticate.

1. Export or obtain a copy of the server's certificate from the FTP server administrator. Otherwise connect to the server with the CuteFTP GUI (interface) from the developer machine and manually accept the server's certificate. You can then export it from the Trusted List under **Global Options > Security > SSL Security** to a staging directory of your choice.

2. Place/Install the certificate into the target machine's Default User directory. For example:

C:|Documents and Settings|Default User|Application Data|GlobalSCAPE|CuteFTP Pro|6.0|Security

3. If you will also be running the TE while logged in, place a copy of the server certificate under the USER folder on the target machine, as follows:

C:\Documents and Settings\[USERNAME]\Application Data\GlobalSCAPE\CuteFTP Pro\6.0\Security
- where [USERNAME] is the login name for the user.

4. You can also tell the TE to auto-accept SSL or SSH certificates using the AutoCloseMethod parameter of the Option method. Also refer to the following trouble shooting topic regarding running the TE while not logged in.

Now when you connect, the TE will see the server's certificate and proceed with the connection.

Client Certificates

Some FTP SSL servers require that the client provide a certificate. The certificate proves the veracity of the client and is an important factor in weeding out spurious FTP clients masquerading as legitimate ones.

To setup the target system to use client certificates

1. On the developer machine, create the client certificate and private key pair using the CuteFTP GUI (interface). The certificate creation utility is located under the **Tools > Global Settings > Security > SSL** dialog.

2. Add the client certificate to the server's trusted list. Do this by connecting to the server with the CuteFTP GUI (interface) from the developer machine. On the server, move the certificate from the Pending to the Trusted list. If your server only has a Trusted list, manually import the client certificate into the Trusted list.

3. Place the client certificate and private key into the target machine's USER folder:
C:\Documents and Settings\[USERNAME]\Application Data\GlobalSCAPE\CuteFTP Pro\6.0\Security
- where [USERNAME] is the login name for the user.

4. If you created the certificate set on the developer machine, export the following registry key:
HKEY_CURRENT_USER\Software\GlobalSCAPE\CuteFTP 6 Professional\Settings\SecuritySSL

5. Import the registry key from step 4 onto the target machine, or manually create the necessary entries. The entries are shown in standard .reg file notation.

```
REGEDIT4
[HKEY_CURRENT_USER\Software\GlobalSCAPE\CuteFTP Pro 3.0\Settings\SecuritySSL]
"SSLCertificate"="C:\\Documents and Settings\\[USERNAME]\\Application
Data\\GlobalSCAPE\\CuteFTP Pro\\6.0\\Security\\client.crt"
"SSLPrivateKey"="C:\\Documents and Settings\\[USERNAME]\\Application
Data\\GlobalSCAPE\\CuteFTP Pro\\6.0\\Security\\client.key"
"UseSSLCertificate"=dword:00000001
"UseSSLCertPassphrase"=dword:00000001
"ReuseSSLData"=dword:00000000
"WarnWhenToNonSecure"=dword:00000001
"DataTransportMethod"=dword:00000001
"SSLCertPassphrase"="[PASSWORD]"
```

- Where [USERNAME] is the login name for the target system and where [PASSWORD] is the encrypted password exported from the source machine. *Don't modify the password!*

Now when you connect, the TE will find the client certificate and use it to authenticate with the server.

Here is a sample script that connects to a fictitious site running SSL Implicit over port 990. It also writes to the event log for debugging purposes. if you use this script to test your setup, make sure you modify the MySite.Host line to include your actual FTP SSL server host and login information.

```
#file test.vbs
Const EVENT_SUCCESS = 0
Set objShell = Wscript.CreateObject("Wscript.Shell")
objShell.LogEvent EVENT_SUCCESS,"AT Loaded me"
Set MySite= CreateObject("CuteFTPPro.TEConnection")
MySite.Option ("ThrowError") = True
MySite.Host = "ftp://myuser:mypass@ftp.mysslserver.com:990
objShell.LogEvent EVENT_SUCCESS,"vars set"
MySite.Connect
objShell.LogEvent EVENT_SUCCESS, "connected"
MySite.Disconnect
MySite.Close
```

License Agreement

GlobalSCAPE Texas, LP CuteFTP® Version 6 Professional License

THIS SOFTWARE IS LICENSED, NOT SOLD. YOU MAY USE THIS SOFTWARE ONLY AS DESCRIBED IN THIS AGREEMENT.

IF YOU DO NOT AGREE TO THE TERMS OF THIS AGREEMENT, DO NOT INSTALL THE SOFTWARE OR USE ANY REGISTRATION NUMBER THAT WAS PROVIDED WITH THE SOFTWARE. YOU MAY CONTACT THE WEB SITE OR STORE WHERE YOU PURCHASED THE LICENSE FOR A REFUND IF YOU HAVE NOT USED THE REGISTRATION NUMBER.

1. SOFTWARE. The capitalized term "Software" refers to the object code for the computer program known as CuteFTP 6 Professional, any updates, supplemental code or programs provided to you by GlobalSCAPE with or in connection with CuteFTP 6 Professional, the user's manual and Help file, any components, any related media and printed materials, and any related "online" or electronic documentation.

2. GRANT OF LICENSE. Evaluation License. If you acquired the license for the Software on an evaluation or trial basis, you may use the Software without charge for the evaluation period. Your evaluation period begins on the day you install the Software. You must pay the license fee and register your copy to continue to use the Software after the evaluation period. To pay the license fee and register your copy, you should visit www.globalscape.com or an authorized sales agent. For so long as the Software is the most current version of CuteFTP 6 Professional distributed by GlobalSCAPE, you may give exact copies of the evaluation Software to anyone. You may not charge any fee for the copy or use of the evaluation Software itself, but you may charge a distribution fee that is reasonably related to any cost you incur distributing the evaluation Software (e.g. packaging). You must not represent in any way that you are selling the Software itself. Your distribution of the evaluation Software will not entitle you to any compensation from GlobalSCAPE. You must distribute a copy of this license with any copy of the Software and anyone to whom you distribute the Software is subject to this license. You may not remove any copyright, trademark or reservation of rights language displayed on, in or with the Software.

Registered License. When you purchase a license, you will be provided with a registration number. You must enable the registered license for the Software by entering the registration number as prompted by the Software. The term of the license is perpetual unless you purchased a limited term license. You may use the registered Software on that number of computers for which you have purchased a separate license as indicated on the invoice or sales receipt. If the Software is installed on a network server or other storage device, you must purchase a license for each separate computer on which the Software is used. A license for the Software may not be shared by alternating use of the Software between different computers. If you acquired the Software for a reduced price as an upgrade from a previous version of the Software, you may no longer use the previous version. The primary user of a computer for which a license has been purchased may make and use one copy of the Software on his or her portable computer. You may also make one copy of the Software for back-up or archival purposes. Otherwise, you may not copy the Software in whole or in part. You may permanently transfer all of your rights under this license if the recipient agrees to the terms of this license, you destroy any copy of the Software not transferred to the recipient, and, if the Software was licensed to you at a reduced price as an upgrade from a previous version, you destroy any copy of any previous version that is not transferred to the recipient. If you purchased the Software on a multi-user basis, you may permanently transfer your rights to one person only, but that person may use the Software on that number of computers for which you purchased a license.

3. RESTRICTIONS. You may not reduce the Software to human readable form, reverse engineer, de-compile, disassemble, merge, adapt, or modify the Software, except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation. You may not rent, lease, or lend the Software. You may not use the Software to perform any unauthorized transfer of information, such as copying or transferring a file in violation of a copyright, or for any illegal purpose.

4. SUPPORT SERVICES. GlobalSCAPE may provide you with support services related to the Software. Use of support services is governed by the user's manual, online documentation, and other GlobalSCAPE materials, as they may be modified from time to time. GlobalSCAPE may use any information you provide as part of obtaining support services for its business purposes, including product support and development.

5. INFORMATION COLLECTION. The Software includes a feature that assigns a unique number to your computer based on system information. The Software reports this number to us either when you install the Software or enter your registration number, or both. The Software may also identify and report to us your Windows language identifier setting, IP address, and the date and time of installation and/or registration. We use this information to count the number of installations, detect piracy of the Software, and develop rough statistical data regarding the geographic location of our users. If you register our Software, we tie this information to the personally identifiable information that we have about you.

6. TERMINATION. This license terminates if you fail to comply with its terms and conditions. If your license terminates, you must destroy all copies of the Software. The termination of this license does not limit GlobalSCAPE's other rights it may have by law.

7. INTELLECTUAL PROPERTY. You acknowledge that you have only the limited, non-exclusive right to use and copy the Software as expressly stated in this license and that GlobalSCAPE retains title to the Software and all other rights not expressly granted. You agree not to remove or modify any copyright, trademark, patent or other proprietary notices that appear, on, in or with the Software. The Software is protected by United States copyright, patent and trademark laws and international treaty provisions.

8. EXPORT RESTRICTIONS. You may not export or re-export the Software in violation of the export laws of the United States, or the applicable laws of any other jurisdiction. Among other things, U.S. laws provide that the Software may not be exported or re-exported to certain countries that are embargoed or restricted or to certain restricted persons. Embargoed and restricted countries currently include Cuba, Iran, Iraq, Libya and Sudan. THE SOFTWARE CONTAINS ENCRYPTION TECHNOLOGY THAT IS CONTROLLED FOR EXPORT BY THE U.S. GOVERNMENT UNDER THE EXPORT ADMINISTRATION ACT. IN ADDITION TO OTHER RESTRICTIONS DESCRIBED IN THIS SECTION, YOU MAY NOT USE THE SOFTWARE, OR EXPORT THE SOFTWARE TO ANY DESTINATION WHERE YOU KNOW OR HAVE REASON TO KNOW THAT THE SOFTWARE MAY BE USED, IN CONNECTION WITH THE PROLIFERATION OF NUCLEAR, CHEMICAL OR BIOLOGICAL WEAPONS OR MISSILES.

9. NO WARRANTIES: **TO THE EXTENT PERMITTED BY APPLICABLE LAW, THE SOFTWARE AND ANY SUPPORT SERVICES ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IF APPLICABLE LAW REQUIRES A WARRANTY, THE REQUIRED WARRANTY IS LIMITED TO NINETY (90) DAYS FROM YOUR RECEIPT OF A COPY OF THE SOFTWARE. COMPUTER PROGRAMS ARE INHERENTLY COMPLEX, AND THE SOFTWARE MAY NOT BE FREE OF ERRORS. THE SOFTWARE IS PROVIDED WITH ALL FAULTS AND THE ENTIRE RISK AS TO SATISFACTORY QUALITY, PERFORMANCE, ACCURACY AND EFFORT IS WITH YOU.**

10. **LIMITATION OF LIABILITY. GLOBALSCAPE IS NOT LIABLE TO YOU FOR ANY CONSEQUENTIAL, SPECIAL, INCIDENTAL, OR INDIRECT DAMAGES OF ANY KIND ARISING OUT OF THE DELIVERY, PERFORMANCE, OR USE OF THE SOFTWARE, EVEN IF GLOBALSCAPE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. UNLESS APPLICABLE LAW PROVIDES OTHERWISE, GLOBALSCAPE'S LIABILITY FOR ANY CLAIM RELATED TO YOUR PURCHASE OF A LICENSE FOR OR USE OF THE SOFTWARE, WHETHER IN CONTRACT, TORT, OR ANY OTHER THEORY OF LIABILITY WILL NOT EXCEED THE GREATER OF U.S. $5.00 OR THE LICENSE FEE PAID BY YOU.**

11. U.S. GOVERNMENT. The Software is commercial computer software developed solely at private expense. The rights of civilian and non-civilian agencies of the U.S. Government to use, disclose and

reproduce the Software are governed by the terms of this License. Publisher is GlobalSCAPE Texas, LP, 6000 Northwest Parkway, Suite 100, San Antonio, Texas, 78249, USA.

12. SECURITY. You agree that GlobalSCAPE is not liable to you for security breaches resulting from your use of the Software. The security of the Software depends primarily on you selecting a secure password and keeping it confidential. You should not select a password that may be easily discovered by others. For example, you should not use a word or sequential series of numbers. We recommend a random choice of at least 6 mixed alpha and numeric characters, with variations between upper and lower case.

13. AUDIT. You agree that on GlobalSCAPE's request you will certify in writing your compliance with the terms of this license, including your use of the Software only on the number of computers/servers licensed.

14. MISCELLANEOUS. This license is governed by the laws of the State of Texas, U.S.A. This license is not governed by the United Nations Convention of Contracts for the International Sale of Goods. You agree to submit to the jurisdiction of courts sitting in the State of Texas for all purposes. Sole and exclusive venue for any dispute arising under or relating to this agreement shall be in a court sitting in Bexar County, San Antonio, Texas. This license constitutes the complete and exclusive agreement between us, notwithstanding any provision in any purchase order or other written document, except for our except for the definition of any evaluation period or limited term license appearing on the Web or other documentation accompanying the license and the statement of the number of separate user license fees for which you have paid as described in Section 2, above. This license may only be modified by a written document signed by GlobalSCAPE. No GlobalSCAPE dealer or distributor is authorized to change the terms of this license. If any provision of this license is held to be unenforceable, the remainder of the license shall not be affected, and the unenforceable provision shall be reformed to the extent necessary to make the provision enforceable. If you are located outside the United States, then the following provision applies: Les parties aux présentés conferment leur volonté que cette convention de même que tous les documents y compris tout avis qui SHY rattaché, soient redigés en langue anglaise. (Translation: "The parties confirm that this Agreement and all related documentation is and will be in the English language.")

Should you have any questions concerning this license, or if you desire to contact GlobalSCAPE for any reason, please contact GlobalSCAPE by mail at: 6000 Northwest Parkway, Suite 100, San Antonio, Texas, 78249, USA, by telephone at: (210) 308-8267, or by electronic mail from: http://www.globalscape.com/support/mail6.asp.

Distribution of the Transfer Engine

If you have created a script or application that calls the TE, you may wish to distribute your script or application to a group of end-users.

CuteFTP's Transfer Engine (TE) is subject to End User License Agreement and can only be distributed in its current form (evaluation software). However you can purchase a license for and register the trial version of the TE on each machine you install it on.

Subsequently you must configure the TE so that it can run properly without the CuteFTP GUI (interface) installed, especially if you plan to run automated or scheduled tasks while not logged in, or if you plan to connect to SSL enabled FTP servers.

Refer to the Licensing and Distribution topic for more details.

Note

You may wish to contact GlobalSCAPE and request special licensing arrangements when distributing to large amounts of users. Refer to the Contacting GlobalSCAPE section for details on how to contact us.

Registration and Trademarks

This program includes software developed by Info-Zip, zlib, David Wincelberg, and the OpenSSL Projects. See the Info-Zip License Agreement, OpenSSL License Agreement, and zlib license agreement. A portion of this program is derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm.

OpenSSL License Agreement

This program includes software developed by the OpenSSL Project which was used by GlobalSCAPE pursuant to the following license.

Copyright (c) 1998-2003 The OpenSSL Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (http://www.openssl.org/)"

4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org.

5. Products derived from this software may not be called "OpenSSL" nor may "OpenSSL" appear in their names without prior written permission of the OpenSSL Project.

6. Redistributions of any form whatsoever must retain the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/)"

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

Info-Zip License Agreement

This program includes Info-Zip Software which was used by GlobalSCAPE pursuant to the following license.

This is version 2000-Apr-09 of the Info-ZIP copyright and license. The definitive version of this document should be available at ftp://ftp.info-zip.org/pub/infozip/license.html indefinitely.

For the purposes of this copyright and license, "Info-ZIP" is defined as the following set of individuals:

Mark Adler, John Bush, Karl Davis, Harald Denker, Jean-Michel Dubois, Jean-loup Gailly, Hunter Goatley, Ian Gorman, Chris Herborth, Dirk Haase,Greg Hartwig, Robert Heath, Jonathan Hudson, Paul Kienitz, David Kirschbaum,Johnny Lee, Onno van der Linden, Igor Mandrichenko, Steve P. Miller,Sergio Monesi, Keith Owens, George Petrov, Greg Roelofs, Kai Uwe Rommel,Steve Salisbury, Dave Smith, Christian Spieler, Antoine Verheijen,Paul von Behren, Rich Wales, Mike White

This software is provided "as is," without warranty of any kind, express or implied. In no event shall Info-ZIP or its contributors be held liable for any direct, indirect, incidental, special or consequential damages arising out of the use of or inability to use this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. Redistributions of source code must retain the above copyright notice, definition, disclaimer, and this list of conditions.
2. Redistributions in binary form must reproduce the above copyright notice, definition, disclaimer, and this list of conditions in documentation and/or other materials provided with the distribution.
3. Altered versions--including, but not limited to, ports to new operating systems, existing ports with new graphical interfaces, and dynamic, shared, or static library versions--must be plainly marked as such and must not be misrepresented as being the original source. Such altered versions also must not be misrepresented as being Info-ZIP releases--including, but not limited to, labeling of the alteredversions with the names "Info-ZIP" (or any variation thereof, including, but not limited to, different capitalizations), "Pocket UnZip," "WiZ" or "MacZip" without the explicit permission of Info-ZIP. Such altered versions are further prohibited from misrepresentative use of the Zip-Bugs or Info-ZIP e-mail addresses or of the Info-ZIP URL(s).
4. Info-ZIP retains the right to use the names "Info-ZIP," "Zip," "UnZip," "WiZ," "Pocket UnZip," "Pocket Zip," and "MacZip" for its own source and binary releases.

zlib License Agreeement

This program includes Info-Zip Software which was used by GlobalSCAPE pursuant to the following license.

zlib.h -- interface of the 'zlib' general purpose compression library version 1.2.1, November 17th, 2003

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software. Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

3. This notice may not be removed or altered from any source distribution.

Jean-loup Gailly jloup@gzip.org

Mark Adler madler@alumni.caltech.edu

Index