How to upload a file using FTP

Introduction

The tsNet external allows you to transfer data over more than just the HTTP and HTTPS protocols. It also supports transferring files via the FTP, FTPS and SFTP protocols.

This lesson shows how you can upload a file via the FTP protocol using a publicly available FTP server. Note that the server used in this code automatically removes uploaded files from the server as soon as the transfer has completed for security reasons.

Lay out the Stack

Create a new stack then drag a button and two fields onto it.

Set the name of the fields to "Output" and "Server Response".

Save this stack.
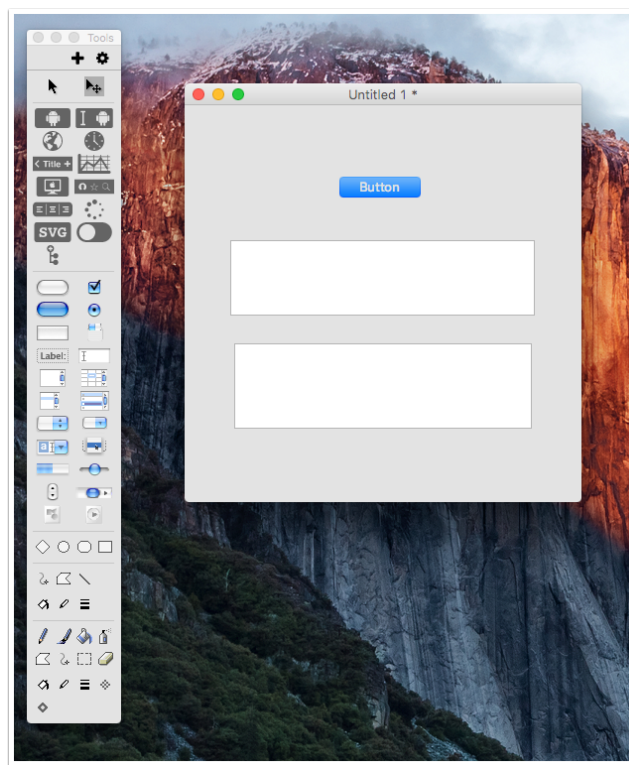
Create the Script

Edit the script of the button that you placed on the stack by right clicking on the button and selecting "edit script".

Add the following code.

```
on mouseUp pMouseButton
    -- Store the FTP server details in some variables
    put "ftp://speedtest.tele2.net" into tFtpServer
    put "/upload" into tFtpDirectory

    -- The tSettings array can be used to provide additional configuration details, in this case
    -- the username and password to use when transferring the files via FTP
    put "anonymous" into tSettings["username"]
    put "test" into tSettings["password"]

    -- Prompt the user for a file to upload and only continue if they did not cancel
    answer file "Select a file to upload"
    if the result is not "Cancel" then
        -- Store the path to the file the user chose
        put it into tFileToUpload

        -- Get the file name of the file they selected (everything past the last slash)
        set the itemDel to slash
        put the last item of tFileToUpload into tFileName

        -- Since we are using tsNetUploadSync, we have to retrieve the contents of the file they chose
        -- store it in a variable for uploading
        put URL("file:" & tFileToUpload) into tFileContents

        -- Let the user know that the transfer is starting
        put "Uploading file" && tFileName && "to" && tFtpServer & tFtpDirectory & cr into field "Output"

        -- Perform the upload
        put tsNetUploadSync(tFtpServer & tFtpDirectory & slash & tFileName, tHeaders, tFileContents, tRe

        -- Output the response to the user
        put "Transfer complete with server response code" && tResult after field "Output"
        put tRecvHeaders into field "Server Response"
    end if
end mouseUp
```
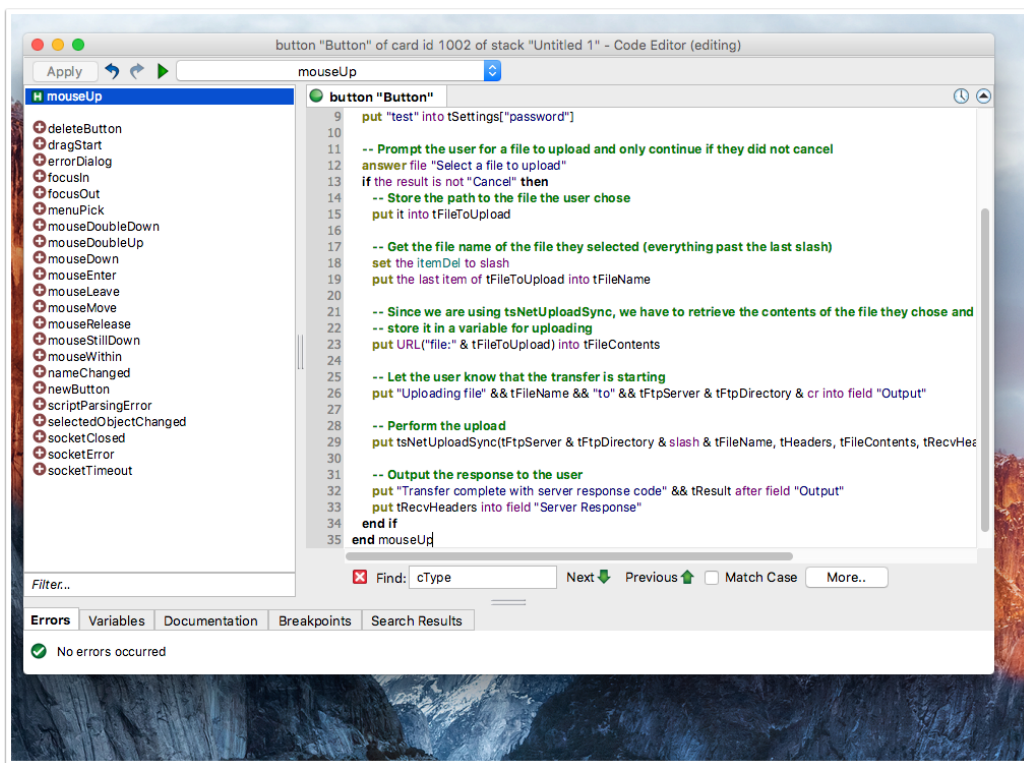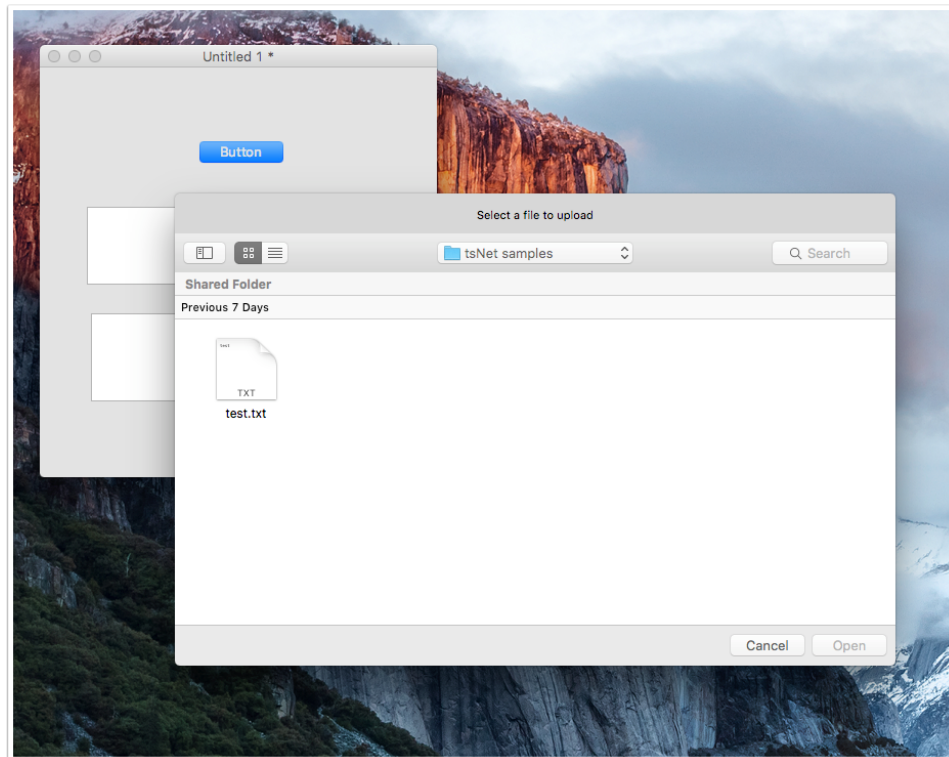
(851756/show_image?image_id=1348772)

Test

Switch to Run mode and click the button. The application will prompt the user for a file to upload.

(851756/show_image?image_id=1348766)

After selecting a file, the application will perform the transfer.

More information

As can be seen in the example above, the **tsNetUploadSync** command returns various pieces of information that relate to the transfer.

In particular, the **tResult** variable used in the above example will contain the status code of the last FTP command that took place as part of the transfer. Unlike transfers over the HTTP protocol, FTP transfers often involve several commands being issued by tsNet. These commands happen transparently and perform tasks such as changing the current directory and authenticating to the FTP server.

By default, tsNet does not automatically close the FTP connection after transferring any data. In these cases, the last command to be issued during a transfer is the upload/download request itself. A successful server response code to this command is usually a 226 code (meaning "Transfer Complete") as seen above.

If tsNet has been set to close the connection (see the "no_reuse" option that can be passed to the **tSettings** array in the LiveCode dictionary for more information) then the last command to be issued will be the disconnect request. This usually results in a server response code of 221 (meaning "Goodbye").

Note however that if the transfer is unsuccessful, the **tResult** variable will contain an error message that begins with "tsneterr:".

Another piece of information returned by the function is stored in the **tRecvHeaders** variable and can be useful in diagnosing the FTP server responses. It provides a full log of all the server responses during the course of a transfer.

---

2 Comments

Carel Tuesday Nov 27 2018 at 07:15 AM

I've tried the lesson but just can't get it to work.
My server only allows TLS/SSL Explicit encryption so I've tried the code with adding:

put true into tSettings["use_ssl"]

I get the following error:

Transfer complete with server response code tsneterr: (35) schannel:
SNI or certificate check failed: SEC_E_WRONG_PRINCIPAL (0x80090322) -
The target principal name is incorrect.

Any idea what I'm doing wrong?

Charles Warwick Thursday Nov 29 2018 at 11:38 PM

Hi Carel,

This error usually means that the server name you are connecting to does not match the server name that the SSL certificate on the server is issued to. For example, if you are connecting to a URL of ftp://ftp.example.com/path/to/file.txt (ftp://ftp.example.com/path/to/file.txt) then the server name is ftp.example.com

In many cases, servers will have multiple server names that can be used when connecting to them. However, when you connect via SSL, you must connect using the server name that is specified on the SSL certificate.

If this is just for testing purposes and security is not really an issue, you can also disable the SSL verification inside tsNet by issuing the following command:

tsNetVerifySSLPeer false

Regards,

Charles.