Introduction

The µTasker supports an optional single user FTP. This operates always in active FTP mode and optionally in passive FTP mode. The basic idea of using FTP is not as a data server where a multitude of users can store or collect large amounts of data, but rather as a practical and comfortable method for developers who would like to do such things as modifying HTML files for use by the web server or collect data files with recordings registered by the application. Of course this can also be useful for general users who may want to also perform their own web site modification (for example to translate text to another language), without needing to be able to recompile these changes in the main project code.

Depending on exact requirements, the FTP server can be set up to allow uploads, downloads and deleting files from the µFileSystem, which is usually in internal FLASH but can also be in an external device connected via SPI.

The FTP server either accepts anonymous login or else can check user name and password credentials.

This document is valid for the µTasker project from V1.3 with SP with passive support.

How does FTP work?

Before looking at how to set up the FTP server it may be worth while reviewing how FTP operates [*FTP is specified in RFC 959*]. The first thing to know is that it uses TCP connections on the standard FTP command port 21, and the server listens for a connection from an FTP client, such as Internet Explorer or Firefox (FireFTP), or other favourite ones, or even the FTP DOS command line client.

The client establishes a TCP connection with the server and the server responds with a message with the code `220`. This is basically signalling that it is available and will accept a login, together with a human readably welcome message – for example "***Welcome M5223X FTP***" when operating on the M5223X Coldfire.

The client will now send its user name, which is either a real user name or the anonymous user name "ANONYMOUS". The FTP server responds with the code `331`, signalling that a password should now be entered. If the µTasker FTP server is configured to accept anonymous login (not require authentication) it will always accept the anonymous user with any password.

If configured to not accept anonymous login (for example to increase security against other people making changes in the file system) a pop-up will normally appear to prompt the user to enter his/her user name and password. Most Browser based FTP clients will automatically attempt anonymous login and so immediately provoke the pop-up, although some will also offer session management where the user name/password pair are automatically entered.

Note that it is also possible to automatically enter the address with a user name and password by using the following syntax in the FTP client browser address line:

    ftp://ADMIN:uTasker@192.168.0.3

where the user name and password are ADMIN and uTasker respectively, which are the default values in the µTasker demo project. *The user name and password disappear once the connection attempt has been stared.*

A successful login with be acknowledged by the code `230`.

Once the FTP command connection has been established and the user has successfully logged in, the FTP process continues with some negotiation of options. Exactly what takes place depends on the FTP client in use but typically the system type will be requested and the data mode will be set to Image type (binary). The FTP client may try to move to a certain directory in the file system and activate Passive mode of operation – but we don't want to get stuck in the details here. As mentioned in the introduction, the idea of the µTasker FTP implementation is not to enable a full-blown FTP server to operate but instead to use the useful properties of FTP for our work. For this reason most of these command and requests are simply acknowledged in a way that enables the start up sequence to get to a workable state. If there are unknown commands received, these are simply acknowledged with the unknown command (syntax error) code `500`, which allows the command to be ignored.

Typically the FTP client will start by send a directory listing request (LIST) so that the present contents of the directory can be viewed. This listing illustrates the use of the FTP data connection, where the exact details of operation depend on the FTP Active/Passive mode setting. The operation of both of these modes is now shown:

Active FTP mode

The active FTP mode is the default mode of operation for FTP servers. If the client does not switch the mode to the passive mode, this is the mode which will be used.

When data - rather than commands - is to be exchanged, a data connection is required. This is a second TCP connection on the well known port 20. This second connection is established for the period of data transfer and so is used in the directory listing to send the actual listing itself.

The active mode means that the FTP server should actively open the data port. In order to be able to do this, it must know the IP address of the client and also the port number that the client will accept the data at. Don't forget that the server has a FTP data socket which will use the well known port address and needs a unique data port as destination for the connection. The FTP client also has to set a temporary listening socket to this port number and informs the server of the details be using the `PORT REQUEST` command over the FTP command connection.

Here is an example of the `PORT REQUEST` for clarity:

"PORT 192,168,254,21,21,42\r\n"

This string request contains the IP address 192.168.254.21 followed by 21,42. This is interpreted as the port number of the temporary listening socket (21 * 256) + 42 = 5418, so it is telling the server to use the address `192.168.254.21:5418` for the FTP data connection when sending the directory listing.

The server will now establish a TCP connection to this port, send the data (a directory listing in this case) and then close the data connection. The FTP data connection thus exists only during data transfers, whereas the FTP command connection is a permanent connection (up to program termination or any time out durations).

Passive FTP mode

When the client specifies the Passive FTP mode of operation, the server switches to this mode and data connections operate a little differently.

In the passive mode, the FTP server doesn't actively establish FTP connections but leaves this to the client. The Firefox browser for example always uses Passive mode of operation (I haven't been able to find a method of configuring it to use active FTP) whereas the Internet Explorer can be set up to operate with either of the two modes. *Note however that the Firefox browser will only allow the files at the FTP server to be viewed and downloaded, for more useful operation it requires the FireFTP add-on. Browers types (FTP clients) will be considered in more detail later in this document.*

The FTP client requests the passive mode by sending the `PASV` command over the FTP command connection. The server responds with the `227` response which includes details of the port for use during the data transfer.

Eg: "227 PASV Mode (192,168,0,3,117,74).\r\n"

The server indicates its IP address and the port which it is listening on – in this case `192.168.0.3:30026` – so that the client can 'actively' establish the connection. In this case the client and server use random port numbers and the well known FTP data port is not actually used(!).

Once the data connection has been established, the server transfers the data - in this case the directory listing - and closes the data connection again.

It is thus seen that the passive and active modes refer to the activity of the *server* when establishing the FTP data connection. The passive side listens on a temporary FTP data socket, the details of which were transmitted either in the `PORT` request or in the answer to the `PASV` request. Once data transfer has completed (the transfer operation itself is not dependant on the mode) the data connection is closed again.

Also most FTP servers have an inactivity timeout period after which the FTP command connection is terminated.

Other activities, such as uploading and downloading data or deleting data, involve a sequence of data connections. Typically there is an initial LIST request followed by a data transfer or file deletion and afterwards another LIST request to ensure that the FTP browser knows the present directory contents.

Now that the basic operation has been discussed, the use of the FTP service in the µTasker project can begin…

Configuring the FTP server support

In order to use FTP in a project it must first be included by setting the define `USE_FTP` in `config.h`. This will activate demo code to configure and start the FTP server as well as activate the contents of `ftp.c`.

The following basic FTP defines exist, with which code can be saved if certain features are not required:

```
#define FTP_USER_LOGIN         // enable user/password authentication support
```

When this define is not set, the FTP server will not require authentication and thus will accept anonymous login. When set, authentication is supported – however whether authentication is actually used can be set as a configuration option when starting the server (this allows user configuration changes).

```
#define FILE_NAMES_PER_FTP_FRAME    6   // limit size of buffers required to
            display files to this many names (remove to use maximum possible)
```

When performing a directory listing, the directory content is sent in a number of TCP frames (the exact number depends on the number of files to display). This value can be used to control the number of file descriptions packed into each TCP buffer and thus control the buffer length used. In systems with relaxed memory restrictions simply remove this and full length TCP frames will always be used.

```
#define FTP_SOCKETS 2          // reserve 2 TCP sockets for command and data
```

This allocates 2 TCP sockets to the TCP socket pool for use by the single user FTP server. This must therefore always be defined.

```
#define FTP_SUPPORTS_NAME_DISPLAY   // show details of files
```

Directory file contents will be displayed as normally viewed in a directory listing. By removing this define, file display will be removed and the available files simply counted. This option is generally necessary for normal operation.

```
#define FTP_SUPPORTS_DELETE    // enable delete of files via FTP
```

If files do not need to be deleted via FTP this option can be removed.

```
#define FTP_SUPPORTS_DOWNLOAD  // support read of files via FTP
```

If files are not to be transferred from the embedded system to the FTP client this option can be removed. It will still be possible to upload files to the FTP server but not to copy them back.

```
#define FTP_VERIFY_DATA_PORT   // check that the data port is valid when in
                                  active mode
```

It has been found that the DOS FTP client sometimes send the `PORT` command with an IP address of 0.0.0.0 and data port 0. To avoid this resulting in the connection stalling while waiting for this invalid data connection to be established, the option checks for this case and informs the FTP client that the data port can not be connected.

```
#define FTP_PASV_SUPPORT       // enable passive mode support
```

This option enables the FTP server to also support the passive transfer operation. If not set, the FTP server will reject the passive command and the FTP client will often simply continue in active mode - However there may be some FTP client policies which will fail if the passive mode is rejected!

Starting the FTP server

To start the FTP server use:

```
   extern void fnStartFtp(unsigned short usFTPTimeout,
                          unsigned char ucFTP_operating_mode);
```

`usFTPTimeout` is the connection timeout in seconds. A value of 0xffff will be interpreted as infinite.

`ucFTP_operating_mode` can be either 0 or `FTP_AUTHENTICATE`, to demand user authentication rather than accept anonymous login. The demo project uses the user name and password in the parameter block (`cUserName[]` and `cUserPass[]`) to authenticate with. The default setting is ***ADMIN*** and ***uTasker***.

To stop the FTP server use:

```
   extern void fnStopFtp(void);
```

It is possible to start and stop the server as desired, each time changes to its mode of operation can be performed. This can be useful for absolute file security. If the FTP server is not started, or is stopped, it is not possible to access it and so it becomes totally safe. FTP authentication is not very secure since it uses plain text name and password which can be quite simply eavesdropped.

That is all there is to it. FTP is otherwise fully autonomous. It has its jobs to do, which are basically to allow files to be transferred between the FTP client and the file system and doesn't need any further application interaction to get these jobs done.

Using FTP clients

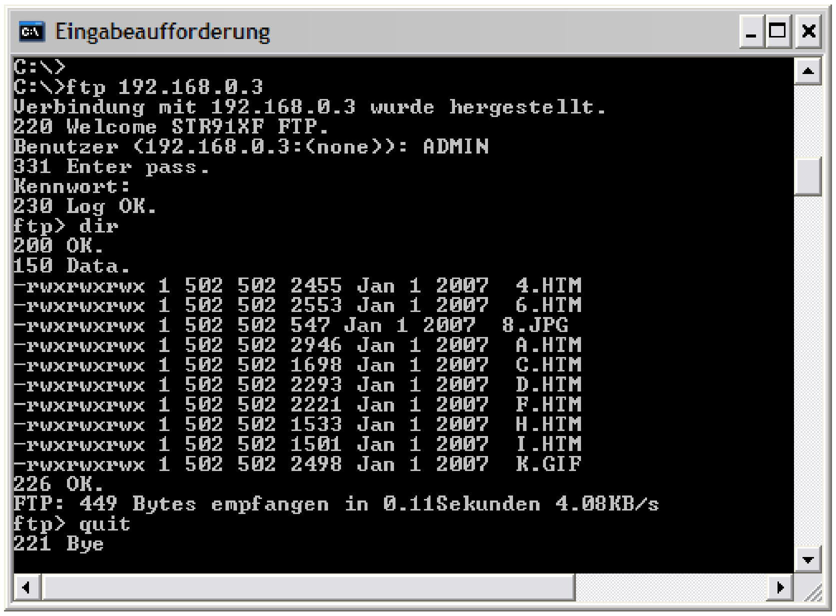
Now this is the trickier bit!

FTP clients are not all the same and this can cause some difficulties. Therefore the µTasker FTP server has been developed with a few goals and a few FTP clients in mind. Although it works well with most FTP clients, not all have been fully tested and there may be possible issues which are not know in these cases. If your favourite FTP client doesn't work, or not as well as you would like, please make an Ethereal recording of what happens and send this to one of the µTasker support addresses. There is often an explanation or a fix can be made to sort it out.

Not all FTP clients are perfect either (surprise, surprise!!) so there are also a few workarounds to ensure that know problems with these are also avoided. In the following section three well know FTP clients are detailed and tips and tricks given to ensure that your project works well with their or µTasker implementation quirks…

DOS FTP client

When all else fails, this is not a bad little tool to do simple tests and even get real work done. It is included in DOS and can be started from a DOS command shell. It can also be called from a script in order to automate file loading or performing tests.



```
C:\>
C:\>ftp 192.168.0.3
Verbindung mit 192.168.0.3 wurde hergestellt.
220 Welcome STR91XF FTP.
Benutzer (192.168.0.3:(none)): ADMIN
331 Enter pass.
Kennwort:
230 Log OK.
ftp> dir
200 OK.
150 Data.
-rwxrwxrwx 1 502 502 2455 Jan 1 2007  4.HTM
-rwxrwxrwx 1 502 502 2553 Jan 1 2007  6.HTM
-rwxrwxrwx 1 502 502 547 Jan 1 2007  8.JPG
-rwxrwxrwx 1 502 502 2946 Jan 1 2007  A.HTM
-rwxrwxrwx 1 502 502 1698 Jan 1 2007  C.HTM
-rwxrwxrwx 1 502 502 2293 Jan 1 2007  D.HTM
-rwxrwxrwx 1 502 502 2221 Jan 1 2007  F.HTM
-rwxrwxrwx 1 502 502 1533 Jan 1 2007  H.HTM
-rwxrwxrwx 1 502 502 1501 Jan 1 2007  I.HTM
-rwxrwxrwx 1 502 502 2498 Jan 1 2007  K.GIF
226 OK.
FTP: 449 Bytes empfangen in 0.11Sekunden 4.08KB/s
ftp> quit
221 Bye
```

This screen shot shows a login (note that the password [Kennwort on the German DOS version] is not displayed) after entering the ftp command to the IP address 192.168.0.3. A simple 'dir' request causes a directory listing to be returned. The 'quit' terminates the session.

Using 'help', the FTP commands are displayed and files can be transferred using `PUT` or `GET`, depending on the direction, or deleted using `DEL`.

Notice that the response codes and human readable comments are visible. This includes the `220` welcome message. This can be customised by setting the string define `FTP_WELCOME_RESPONSE` in your `app_hw_xxxx.h` header.

Here is an example of a batch file* content which performs repeated tests of a command file which transfers various files.

```
ftp -s:"ftp_cmds.txt" -A -w:256 192.168.0.3
```

The file '`ftp_cmds.txt`' contains commands such as

```
binary
dir
put "0Main.html"
put "3Stats.html"
put "6_I_O.html"
dir
quit
```

which does some listings and file transfers before terminating the FTP connection.

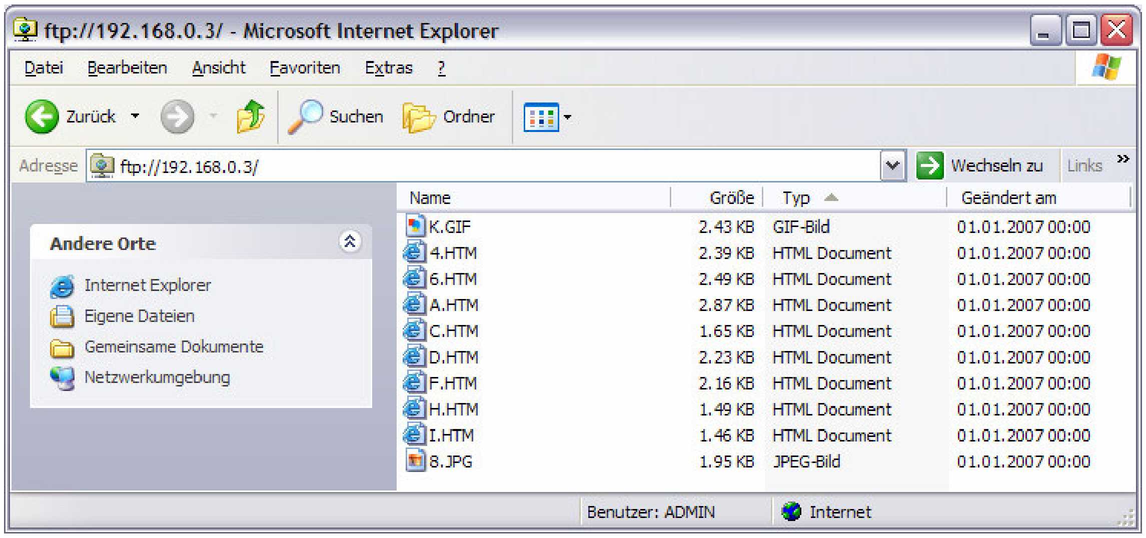
**Thanks to S.T. of CA USA for the input to this one!*

Internet Explorer

The Internet Explorer 6 (IE6) includes a full featured FTP client. To connect to the FTP server simply enter an FTP address rather than an HTTP address and the FTP connection takes place automatically. The connection is attempted using anonymous login and if this fails a pop-up appears allowing a user name and a password to be entered.

The Passive mode can be configured under extended internet options. If the passive option is disabled in the µTasker project configuration IE will automatically fall back to the active mode even if passive support is defined.

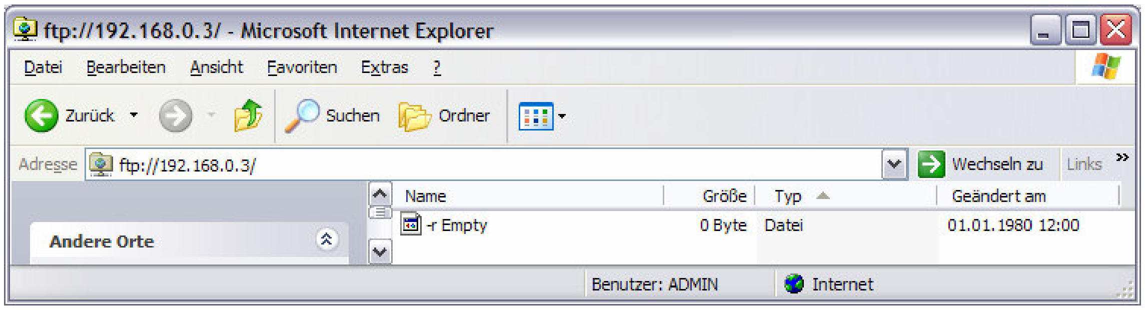
To connect, simple enter the FTP address – eg. ftp://192.168.0.3 as shown in the address field below.



ftp://192.168.0.3/

Name	Größe	Typ	Geändert am
K.GIF	2.43 KB	GIF-Bild	01.01.2007 00:00
4.HTM	2.39 KB	HTML Document	01.01.2007 00:00
6.HTM	2.49 KB	HTML Document	01.01.2007 00:00
A.HTM	2.87 KB	HTML Document	01.01.2007 00:00
C.HTM	1.65 KB	HTML Document	01.01.2007 00:00
D.HTM	2.23 KB	HTML Document	01.01.2007 00:00
F.HTM	2.16 KB	HTML Document	01.01.2007 00:00
H.HTM	1.49 KB	HTML Document	01.01.2007 00:00
I.HTM	1.46 KB	HTML Document	01.01.2007 00:00
8.JPG	1.95 KB	JPEG-Bild	01.01.2007 00:00

In IE, an empty µFileSystem will be displayed by a dummy file called –r Empty. This is used by µTasker to show clearly that no files have been loaded as well as to avoid a small difficulty in the FireFTP program (see later).


ftp://192.168.0.3/ - Microsoft Internet Explorer
Datei Bearbeiten Ansicht Favoriten Extras ?
Zurück
Suchen
Ordner
Adresse
Wechseln zu
Links
Andere Orte
Benutzer: ADMIN
Internet

ftp://192.168.0.3/

Name	Größe	Typ	Geändert am
-r Empty	0 Byte	Datei	01.01.1980 12:00

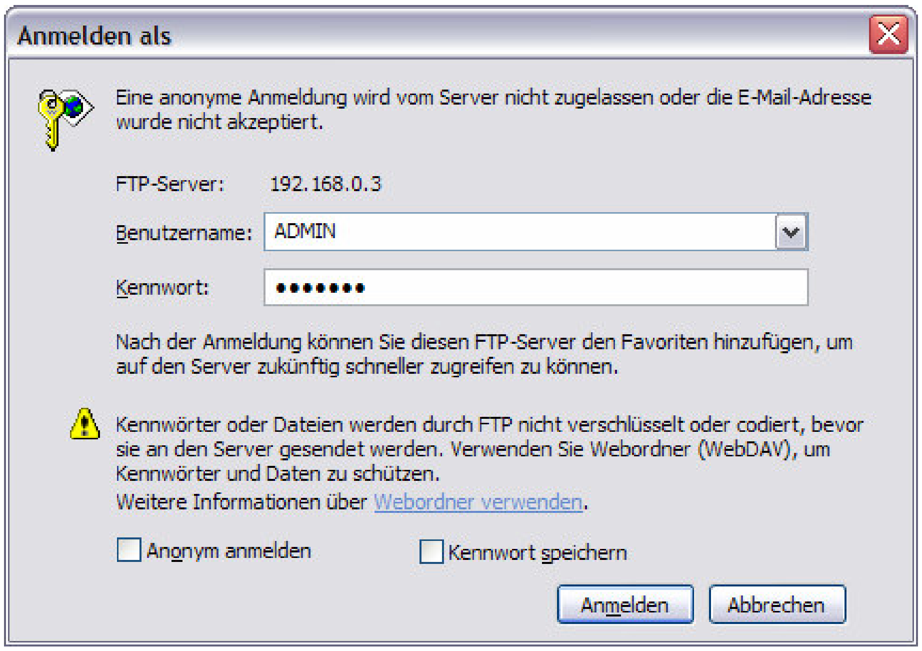
The nice thing about IE 6 is that it works well with the µTasker FTP server in anonymous mode (i.e. without authentication) and allows simple file transfers using ‘drag-and-drop’. This is generally very practical for development word.

The problems with IE6 are these:

1. If you try to connect to a none-existent FTP server address the thing will more or less hang while trying to connect. The same is true when a connection is lost (for example when a break point is set at the target and so the TCP connection times out). One trick which often works is to click on the back button so that it returns to the previous explorer page rather than seemingly no longer reacting.

2. Authenticated login only works well when the user name and address are entered together with the FTP address.

This screen shot shows the pop up for login when anonymous login is not allowed:


Anmelden als

Eine anonyme Anmeldung wird vom Server nicht zugelassen oder die E-Mail-Adresse wurde nicht akzeptiert.

FTP-Server: 192.168.0.3

Benutzername: ADMIN

Kennwort: •••••••

Nach der Anmeldung können Sie diesen FTP-Server den Favoriten hinzufügen, um auf den Server zukünftig schneller zugreifen zu können.

Kennwörter oder Dateien werden durch FTP nicht verschlüsselt oder codiert, bevor sie an den Server gesendet werden. Verwenden Sie Webordner (WebDAV), um Kennwörter und Daten zu schützen.
Weitere Informationen über Webordner verwenden.

The text is German due to the operating system used but you should get the idea. The login works fine but for some reason the sequence uses two FTP connections (not supported by the µTasker FTP server). The first connection performs the log in and then a second connection logs in parallel with the existing connection using the, now known to be good, credentials. The first connection then remains open but unused. The µTasker FTP server will reject the second connection and so, although logged in, the connection is useless.

If a short FTP connection timeout is set (remember the define when starting the server) the first connection will timeout and the connection can then be re-established after a short wait, where it will immediately be successful using the known credentials. In fact once the IE 6 Browser instance knows the credentials, all further work is fine until the Browser is terminated.

To get around the problem described above when authentication is desired, simply connect using the address **ftp://ADMIN:uTasker@192.168.0.3** . In this case the credentials are immediately made known to the Browser and the initial login uses them and so avoids the 'double- connection' issue.

Internet Explorer 7

Newer versions of Windows are ‘forced’ to use Internet Explorer 7 and it is also not possible to have another Internet Explorer version (like IE 6) installed at the same time.

IE 7 is rather different to IE 6. It logs on nicely to the FTP server and displays the files in the file system but doesn’t allow FTP work as known from IE 6 without taking further steps. It is necessary to first enable the option “*Activate FTP Folder View (outside of Internet Explorer)*” after which it is possible to click on “*Open FTP-Site in Windows-Explorer*”. This in turn opens another FTP window which is more or less the same as the IE 6 FTP window and allows ‘drag-and’drop’ work as with IE 6.

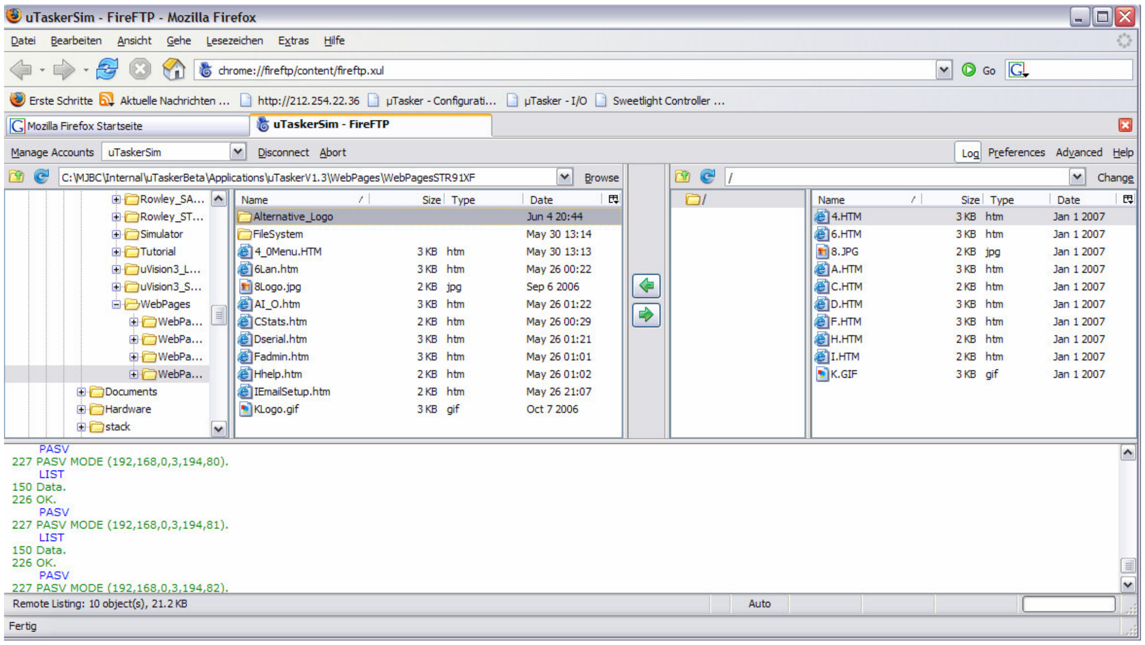
There is however a catch. The second window requires a second FTP connection (the first window is still connected) and so will be rejected by the µTasker FTP server (it is single user don’t forget). The way to get around this difficulty is to configure a short FTP connection time out of a few seconds so that the first connection effectively times out by the time the second windows gets opened and then all works fine. Using IE with quite short timeouts (say 5s) is in fact no problem since the browser simple re-establishes a connection when it is needed, which is invisible to the (normal) user.

FireFox and FireFTP

Firefox is a great browser but its standard FTP client is not very useful for transferring files. It also only works in Passive mode (it will also not fall back to active mode if the passive connection is rejected) so it is advisable to activate passive support in the µTasker FTP server if the FireFox FTP (file viewer) is to be used. It is not possible to upload or delete files from the FTP (file viewer) but there is an add-on called FireFTP which can be installed from the Mozilla Web site (try the following direct connection: https://addons.mozilla.org/de/firefox/addon/684). And it is not bad at all!! It transforms FireFox into a good FTP management tool.

An account can be managed including setting passive / active support and user name/password combination which will automatically be used.

Here is a screen shot of it being used:


uTaskerSim - FireFTP - Mozilla Firefox
Datei Bearbeiten Ansicht Gehe Lesezeichen Extras Hilfe
chrome://fireftp/content/fireftp.xul
Go
Erste Schritte Aktuelle Nachrichten ... http://212.254.22.36 µTasker - Configurati... µTasker - I/O Sweetlight Controller ...
Mozilla Firefox Startseite
uTaskerSim - FireFTP
Manage Accounts
uTaskerSim
Disconnect Abort
Log Preferences Advanced Help
Browse
Change
Rowley_SA...
Rowley_ST...
Simulator
Tutorial
uVision3_L...
uVision3_S...
WebPages
WebPa...
WebPa...
WebPa...
WebPa...
Documents
Hardware
stack
/
Auto

C:\MJBC\Internal\uTaskerBeta\Applications\uTaskerV1.3\WebPages\WebPagesSTR91XF

Name	Size	Type	Date
Alternative_Logo			Jun 4 20:44
FileSystem			May 30 13:14
4_0Menu.HTM	3 KB	htm	May 30 13:13
6Lan.htm	3 KB	htm	May 26 00:22
8Logo.jpg	2 KB	jpg	Sep 6 2006
AI_O.htm	3 KB	htm	May 26 01:22
CStats.htm	2 KB	htm	May 26 00:29
Dserial.htm	3 KB	htm	May 26 01:21
Fadmin.htm	3 KB	htm	May 26 01:01
Hhelp.htm	2 KB	htm	May 26 01:02
IEmailSetup.htm	2 KB	htm	May 26 21:07
KLogo.gif	3 KB	gif	Oct 7 2006

Name	Size	Type	Date
4.HTM	3 KB	htm	Jan 1 2007
6.HTM	3 KB	htm	Jan 1 2007
8.JPG	2 KB	jpg	Jan 1 2007
A.HTM	3 KB	htm	Jan 1 2007
C.HTM	2 KB	htm	Jan 1 2007
D.HTM	3 KB	htm	Jan 1 2007
F.HTM	3 KB	htm	Jan 1 2007
H.HTM	2 KB	htm	Jan 1 2007
I.HTM	2 KB	htm	Jan 1 2007
K.GIF	3 KB	gif	Jan 1 2007

```
    PASV
227 PASV MODE (192,168,0,3,194,80).
    LIST
150 Data.
226 OK.
    PASV
227 PASV MODE (192,168,0,3,194,81).
    LIST
150 Data.
226 OK.
    PASV
227 PASV MODE (192,168,0,3,194,82).
```

Remote Listing: 10 object(s), 21.2 KB

Fertig

As can be seen, the local directories are at the left of the screen and the FTP server directories are at the right. An account called *uTaskerSim* has been created in passive mode, including the user credentials which are sent on link establishment (where ‘Disconnect’ can be seen, there was ‘Connect’ to establish the link).

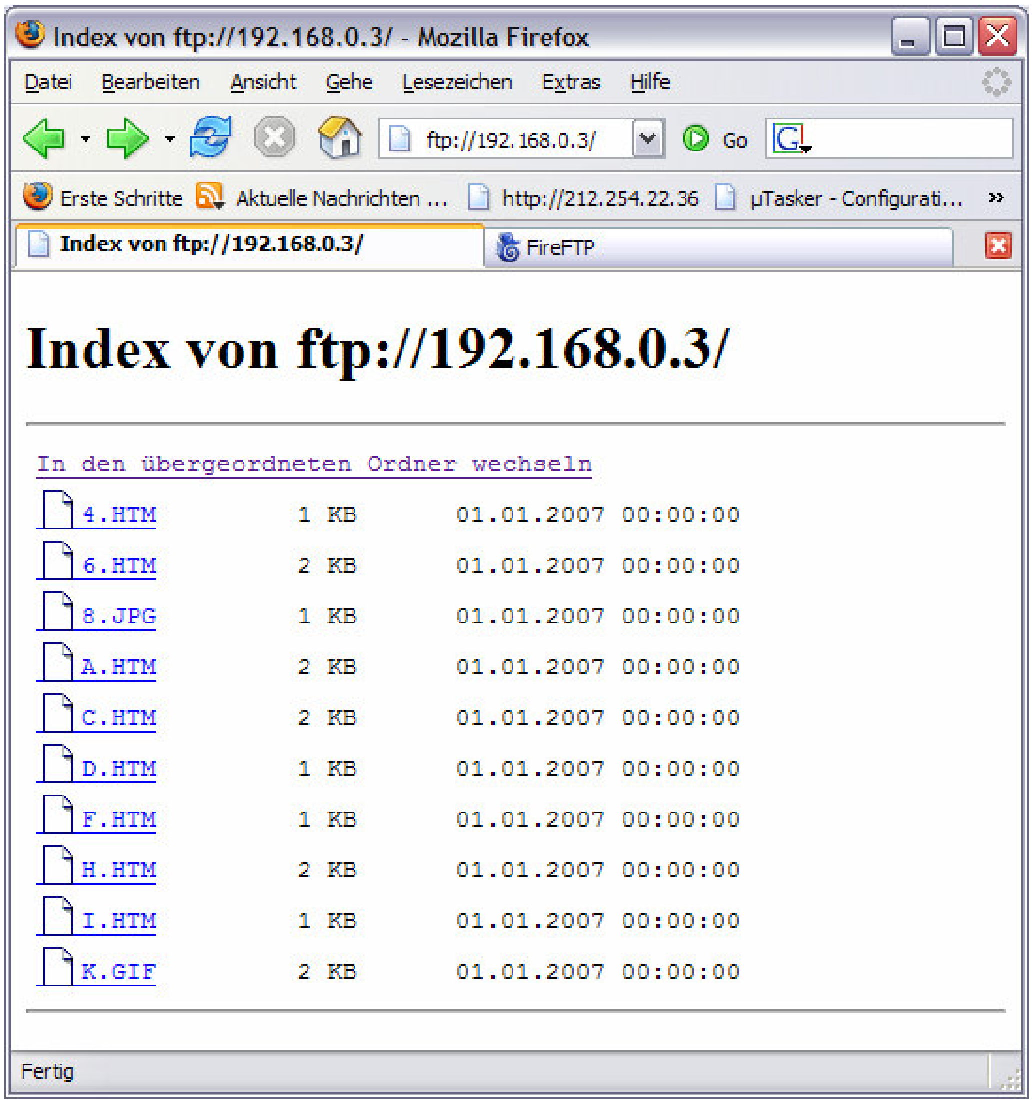
Files can be deleted and dragged from the left to right sides, and vice versa, or transferred by selecting and clicking on the direction arrows. The commands and responses can be seen in the log window (if not disabled) and there are lots of potentially useful configurations in the account manager. All in all very useful…

There are however a couple of points to note:

1. The FTP server timeout should not be set very short (as is useful for IE 7 for example) since FireFTP keeps re-establishing the connection after timeouts even if there is no work to be performed. It requires an open connection to work and doesn’t automatically establish one when needed (like IE does). Maybe there is a setting to allow this but I haven’t been able to find it (yet).

2. FireFTP doesn't like the empty directory method used by the µTasker FTP server. It actually fails to operate correctly if the dummy file 'Empty' (as used in previous µTasker versions) is displayed but does work fine when there are 'real' files to be displayed. It was found that it is also necessary to have some file attributes associated with the 'dummy' file to avoid the problem. For this reason the empty dummy file has been re-defined to "–r Empty". FireFTP interprets –r as a part of a file attribute and no longer fails. It doesn't actually display the file since its attribute list is not complete but the empty directory that it displays adequately shows that the µFileSystem is empty.

3. When re-connecting to an FTP server which was previously being displayed, it is useful to refresh the display by clicking on the blue circled arrow near to the FTP server contents display. The reason is that sometimes there is no listing performed when a new connection is established and the display may not necessarily be up to date.

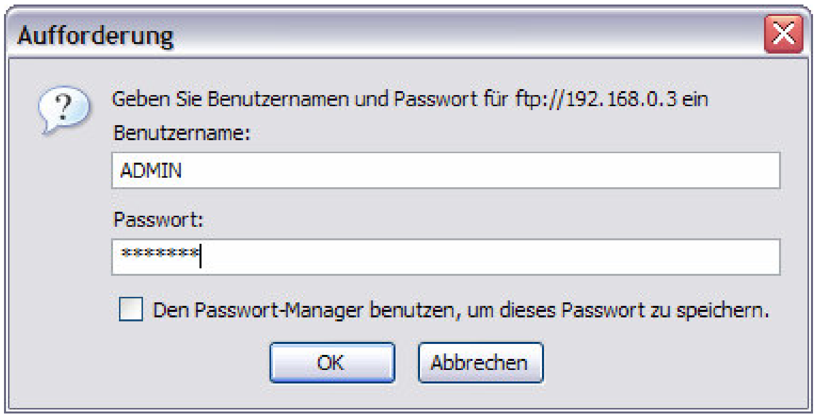


Index von ftp://192.168.0.3/

In den übergeordneten Ordner wechseln

```
4.HTM        1 KB     01.01.2007 00:00:00
6.HTM        2 KB     01.01.2007 00:00:00
8.JPG        1 KB     01.01.2007 00:00:00
A.HTM        2 KB     01.01.2007 00:00:00
C.HTM        2 KB     01.01.2007 00:00:00
D.HTM        1 KB     01.01.2007 00:00:00
F.HTM        1 KB     01.01.2007 00:00:00
H.HTM        2 KB     01.01.2007 00:00:00
I.HTM        1 KB     01.01.2007 00:00:00
K.GIF        2 KB     01.01.2007 00:00:00
```

Typical FTP view in FireFox (not FireFTP)


Aufforderung

Geben Sie Benutzernamen und Passwort für ftp://192.168.0.3 ein

Benutzername: ADMIN

Passwort: *******

Den Passwort-Manager benutzen, um dieses Passwort zu speichern.

Password entry pop-up when using FireFox (not FireFTP)

Note that the FireFox FTP viewer doesn't close down the FTP connection until it either times out or FireFox is terminated. Therefore it can 'hog' the µTasker FTP server…
FireFox alone is not recommended for general use due to its limited capabilities, together with the FireFTP add on it is however highly recommended!

Conclusion

The µTasker FTP server implementation has been described along with some background of FTP operation. The configuration of user authentication and passive/active FTP options have been shown to be important also to the operation of typical FTP clients.

The use of DOS FTP, Internet Explorer and FireFox/FireFTP were then introduced including tips for best operation with the µTasker FTP server as well as strengths and weaknesses of the various solutions.

The µTasker FTP implementation is of small footprint size to operate together with the µFileSystem in an autonomous manor [*eg. approx. 2,5k program size on an ARM processor in Thumb mode with all FTP options activated*]. Projects can benefit from the comfort of transferring files for use by the web server or various other storage requirements of an application using user friendly FTP clients, either in a local area network or via the Internet. This capability avoids for example the necessity to compile pages to be served by the web server into a project, which allows simple modification to be made by developers or even end users.

Document state:

- 7.7.2007 Initial version for the V1.3 project
- 8.7.2007 Typos clean-up