Subject: Re: FTP class and reference example for U++
Posted by Oblivion on Wed, 12 Apr 2017 20:34:01 GMT
View Forum Message <> Reply to Message

Hello guys,

Today marks a new milestone for FTP package. The new version (1.1) of FTP package is finally here.
After several iterations, I believe the package has become stable.
This version brings multithreaded file transfers to the ftp class, via two convenience functions: FtpAsyncGet() and FtpAsyncPut():

int FtpAsyncGet(const String& remote_file, const String& local_file, const String& host, int port = 21, const String& user = Null,
	const String& pass = Null, Event<int, int, String, int64, int64> progress = CNULL, bool active = false, bool ssl = false,
	bool ascii = false);

int FtpAsyncPut(const String& local_file, const String& remote_file, const String& host, int port = 21, const String& user = Null,
	const String& pass = Null, Event<int, int, String, int64, int64> progress = CNULL, bool active = false, bool ssl = false,
	bool ascii = false);

As you can see the syntax is very similar to the single-threaded FtpGet() and FtpPut() functions. Except these async functions do concurrent upload/download while retaining the flexibility of the Ftp class.
Moreover, these functions are implemented in a general way that, I believe, they can serve also as a reference implementation. While the Ftp class itself is still single threaded.

Also, in this version, the FEAT command is implemented. It is now possible to query the capabilities of ftp servers easily.

One more addition to the package is a multithreaded simple FTP browser example.
Example browser can use secure connection (FTPS), download, upload, manipulate files, and manipulate directories, show file information etc.

First on my todo list is to implement REST command to allow restart/resume of interrupted transfers.

And Mirek, I know that you are a very busy person, but you'd asked me about the advantages of my ftp implementation over the current one in Core/Ftp.

I now believe FTP package is eligible to Bazaar, considering that it has IPV6, FTPS, multithreading support, better integration with U++ core and UI classes, a clean code-base, active development, is well documented, and easily extendible (using SendCommand() method), and has an example browser built around it. It makes a good alternative. What do you think?

As usual, you can find the development log and updated package in the first post of this topic.

I tested FTP package with several public FTP servers, and private servers I set up.
I can recommend,

ftp.uni-erlangen.de, User: Anonymous, Password: Anonymous.
test.rebex.net, User: demo, Password: password. Allows FTPS
demo.wftpserver.com, User: demo-user, Password: demo-user. Allows FTPS and temporary uploads.

Cheers!

Oblivion
