Subject: Re: SSH package for U++
Posted by Oblivion on Tue, 07 Aug 2018 21:32:41 GMT
View Forum Message <> Reply to Message

Hello Mirek,

I'll start committing the changes in a couple of days. But before I start I'd like to clear some points and ask for your opinion on some points.

Quote:
SFtpHandle* -> SFtpHandle - if something is "HANDLE", it should not be a pointer to handle.

The thing is, libbsh2 itself returns a pointer to a handle (SFtpHandle is actually an alias for LIBSSH2_SFTP_HANDLE)
I'm reluctant to change that pointer into a real handle as it'll make the things slightly more complex.
However, if we must change the naming I would suggest we use "SFtpObject" instead, since we are dealing with remote file system objects here.

Quote:
Interestingly, it looks like the most important Gets / Puts are missing there

```
int Get(SFtpHandle h, void *ptr, int size);
bool Put(SFtpHandle h, const void *ptr, int size);
```

Easy to implement. I'll propose:

```
int      Get(SFtpObject* obj, void* buffer, int size);
String     Get(const String& path, int size, int64 offset = 0);
String     GetAll(const String& path);

int      Put(SFtpObject* obj, const void* buffer, int size);
int      Put(const String& path, const String& data, int size, dword flags, long mode);
int      Put(const String& path, const String& data, int size, int64 offset = 0);
bool       PutAll(const String& path, const String& data);
```

As for the stream versions of Gets and Puts:
Stream versions come very handy when transferring files > 2GB, and processing the I/O immediately (e.g when filtering).

I've thought about adding SFtpStream variants(eg. SFtpFileIn, SFtpFileOut, etc.) to the package, as you suggessted.
But in the end I concluded that they'll only add another layer of abstraction which can be confusing (as you know, SSH package already has a plenty of classes). Thus I propose to simply rename

them and change their parameter order.

E.g.

```
bool GetStream(const String& path, Stream& out);
bool PutStream(const String& path, Stream& in);
// And other variants...
```

About the reference examples:

My idea is to create a reference -console- example for each class (SFtp, Scp, SshExec, SshShell, SshTunnel), and have them each contain separate and basic blocking, non-blocking, and multitihreaded code as functions that can be selected/compiled using preprocessor directives.(defs,e.g. BLOCKING, NONBLOCKING, MULTITHREADED). And use these examples as tutorial in Topic++.

What do you think?

Best regards,
Oblivion