
Subject: Re: SFTP or full SSH2 support for U++?
Posted by [Oblivion](#) on Sun, 06 Mar 2016 22:39:46 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hello guys,

A major update:

SSH package is re-written using the new version of AsyncQueue class which can utilize ordinary callbacks and lambdas, instead of the cumbersome VarArgs class. SSH package relies heavily on lambda functions, therefore it will require C++11. I believe it is much more elegant now, since it drastically simplified the task and reduced the source code significantly. It is almost complete, and first release is at hand. (SFtp component, at least).

Below code snippet is taken from the new version of a simple download test app that concurrently downloads a number of jpg files from a well-known public Sftp test server. As you can see, unnecessary state tracking is removed:

```
struct Job {
    SFtp  sftp;
    FileOut file;
    int   index;
};

Array<Job> jobs;

//.....

const char *sftp_server  = "demo.wftpserver.com";
const char *sftp_user    = "demo-user";
const char *sftp_password = "demo-user";
const char *remote_path  = "/download";

bool SFtpDownloader::ReadDir(Ssh::DirList& ls)
{
    // We'll get the directory listing, using a synchronous (blocking) call.

    SFtp browser;
    browser.WhenDo << [&] {ProcessEvents(); }; // GUI should be
responsive...
    if(browser.Connect(sftp_server, 2222, sftp_user, sftp_password)) {

        // Let us use a convenience method which works on paths, not file descriptors.
```

```

        if(browser.ListDir(remote_path, ls))
            return true;
    }
    Exclamation(browser.GetErrorDesc());
    return false;
}

void SFtpDownloader::Download()
{
    // Wait for the queue to be processed.

    if(!jobs.IsEmpty())
        return;

    Ssh::DirList ls;
    if(!ReadDir(ls))
        return;
    for(int i = 0, j = 0; i < ls.GetCount(); i++) {

        // We can work on directory entries easily.

        Ssh::DirEntry& e      = ls[i];
        const Ssh::Attrs& attrs = e.GetAttrs();

        // Let us simply try to download all files with *.jpg extension to the current directory,
        asynchronously.
        // Here, thanks to AsyncQueue, asynchronous calls work in a fire-and-forget fashion. Similar
        to "batch processing".
        // E.g., any number of Sftp commands/jobs can be queued to be processed without
        intervention (till they're finished, or failed, of course).

        if(attrs.IsFile() && GetFileExt(e.GetName()).IsEqual(".jpg")) {
            Job& job  = jobs.Add();
            job.index = j;
            job.file.Open(Format("%s/SFtpExample-%d-%s", GetHomeDirectory(), job.index,
e.GetName()));
            job.sftp.StartConnect(sftp_server, 2222, sftp_user, sftp_password);
            job.sftp.StartGet(job.file, Format("%s/%s", remote_path, e.GetName()), Ssh::READ, 0755,
THISBACK2(DownloadProgress, job.index, e.GetName()));
            j++;
        }
    }

    // Below loop does asynchronous job processing.

    while(!jobs.IsEmpty()) {
        int i = 0;
        SocketWaitEvent we;

```

```

we.Add(jobs[i].sftp.GetSocket());
we.Wait(10);
while(i < jobs.GetCount()) {
    SFtp& sftp = jobs[i].sftp;
    sftp.Do();
    if(!sftp.InProgress()) {
        // For the sake of simplicity, we do not differentiate between success and failure.
        jobs.Remove(i);
        break;
    }
    ProcessEvents();
    i++;
}
}
}

//.....

```

Before I release the first public [beta] version, I'll have to write its docs. It'll probably happen this week.

Regards,

Oblivion.

File Attachments

1) [Sftp - New version.jpeg](#), downloaded 3161 times

main.cpp
SftpBrowser.h
Help Topics
SSH.cpp
SSH.h
SFTP.cpp
SftpBrowser.log

Symbol/lineno (Ctrl+G)

All
Upp (SSH/SFTP.cpp)
SSH/SFTP.cpp
Upp::Sftp

Upp (SSH/SFTP.cpp)
Upp : namespace
SSH/SFTP.cpp
LLOG(x)
Upp::Sftp
StartFStat(Ssh::Attrs& attrs, bool set) : v
StartConnect(const String& host, int po
StartDisconnect() : void
StartOpen(const String& path, unsigne
StartClose() : void
StartRemove(const String& path) : void
StartRename(const String& oldpath, co
StartSync() : void
StartOpenDir(const String& path) : void
StartListDir(Ssh::DirList& list, Gate1<Ssh
StartListDir(const String& path, Ssh::Dir
StartMakeDir(const String& path, long
StartRemoveDir(const String& path) : v
StartPut(const Stream& in, Gate2<int64,
StartPut(const Stream& in, const String&
StartGet(Stream& out, Gate2<int64, int6
StartGet(Stream& out, const String& pa
Seek(uint64 position) : bool
Tell() : uint64
Sftp()

SSH/SSH.h (183)

202
203 void Sftp::StartMakeDir(const String& path, long mo
204 {
205 AddJob() << [&, path, mode] {
206 int rc = libssh2_sftp_mkdir(sftp_session,
207 if(rc == 0) {
208 LLOG("++ SSH2::SFTP: Directory succesf
209 ProcessQueue();
210 }
211 else
212 if(rc != LIBSSH2_ERROR_EAGAIN)
213 SetError(LIBSSH2);
214 };
215 }
216
217 void Sftp::StartRemoveDir(const String& path)
218 {
219 AddJob() << [&, path] {
220 int rc = libssh2_sftp_rmdir(sftp_session, p
221 if(rc == 0) {
222 LLOG("++ SSH2::SFTP: Directory succesf
223 ProcessQueue();
224 }
225 else
226 if(rc != LIBSSH2_ERROR_EAGAIN)
227 SetError(LIBSSH2);
228 };
229 }
230
231 void Sftp::StartPut(const Stream& in, Gate2<int64,
232 {
233 packet_length = 0;
234 const Stream *io_stream = ∈
235 AddJob() << [&, io_stream, progress] {
236 int64 remaining = io_stream->GetSize() - p
237 int rc = libssh2_sftp_write(sftp_handle, co
238 if(rc < 0) {
239 if(rc != LIBSSH2_ERROR_EAGAIN)
240 SetError(LIBSSH2);
241 return;
242 }
243 packet_length += rc;
244 if(progress(io_stream->GetSize(), packet_le
245 SetError(INTERNAL, t_("File upload abor
246 return;