

---

Subject: Re: SFTP or full SSH2 support for U++?  
Posted by [Oblivion](#) on Tue, 26 Jan 2016 20:02:12 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Hello,

Just to give you all an update:

U++ wrapper for libssh2 is in good shape.

SSHSession class: implemened. I borrowed the U++ memory allocation code (re/alloc, free) from Core/SSL package and after a slight modification glued it to the SSH package. (It is working!)

Sftp class: Implemented all the necessary commands (seek/seek64/statvfs not yet implemented).

I don't have access to Windows right now, so the code is currently tested only on an up-to-date arch linux installation (with a KDE/Plasma5 desktop).

Below is a screenshot of a very simple (and easy to write) sftp downloader example with 10 concurrent downloads, demonstrating non-blocking/async operation capabilities.

Below is the actual code responsible for 10 concurrent downloads, demonstrating sftp basic async api (Some parameters are hard coded. I was being lazy.)

```
// Async jobs.
```

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

```
enum Command { OPEN, READ, CLOSE, FINISH };
```

```
const char *sftp_server = "demo.wftpserver.com";  
const char *sftp_user  = "demo-user";  
const char *sftp_pass  = "demo-user";  
const char *remote_file = "/download/F11_wallpaper_06_1600x1200.jpg";
```

```
void SFtpExample::Download()  
{
```

```
// Initialize and fill an array of Job(s)
```

```
for(int i = 0; i < 10; i++) {  
    Job& job = jobs.Add();  
    job.sftp.User(sftp_user, sftp_pass);  
    job.sftp.StartConnect(sftp_server, 2222);  
    job.sftp.WhenDo = THISBACK(UpdateGui);  
    job.file.Open(Format("/home/testuser/picture-%d.jpg", i));  
    job.cmd = OPEN;  
    job.index = i;  
}
```

```
// Actual loop: connects to the server and downloads a file in a concurrent way.
```

```
while(jobs.GetCount()) {  
    for(int i = 0; i < jobs.GetCount(); i++) {  
        Job& job = jobs[i];  
        SocketWaitEvent e;  
        e.Add(job.sftp.GetSocket());  
        e.Wait(10);  
        job.sftp.Do();  
        if(!job.sftp.InProgress()) {  
            if(job.sftp.IsSuccess()) {  
                switch(job.cmd) {  
                    case OPEN:  
                        job.sftp.StartOpen(remote_file, SSH::READ);  
                        job.path = remote_file;  
                        job.cmd = READ;  
                        continue;  
                    case READ:  
                        job.sftp.StartGet(job.file, THISBACK2(DownloadProgress, job.index, job.path));  
                        job.cmd = CLOSE;  
                        continue;  
                    case CLOSE:  
                        job.sftp.StartClose();  
                        job.cmd = FINISH;  
                        continue;  
                    case FINISH:  
                        break;  
                }  
            }  
            else  
                if(job.sftp.IsFailure())  
                    list.Set(job.index, 1, DeQtF(job.sftp.GetErrorDesc()));  
            jobs.Remove(i);  
            list.Remove(i);  
            for(int n = 0; n < jobs.GetCount(); n++)  
                jobs[n].index = n;  
        }  
    }  
}
```

```
}  
}  
}
```

P.s. I delayed the upload of the package, since it still has some rough edges to iron-out.

Regards,  
Oblivion

#### File Attachments

---

1) [SFTP Concurrent Downloads.jpeg](#), downloaded 1368 times

---