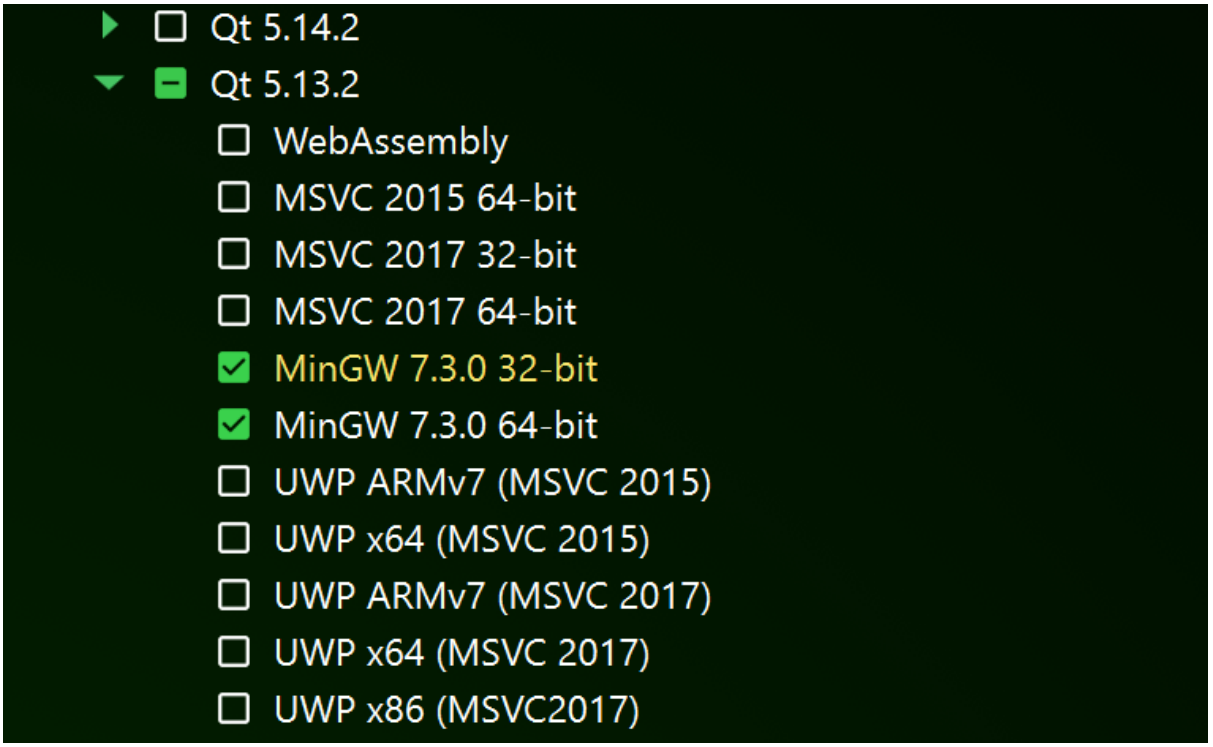
NxNandManager : Set up and build project with Qt (GUI)

Intel 64 Processor - Windows - MinGW64
2022, January

Pre-requisites :

Download Qt (open source) : https://www.qt.io/download
During the installation, make sure to install at least QT 5.13+ for MinGW



Set up & build project :

My installation folder is S:/dev but you should choose your own.

First of all, clone NxNandManager's git repo inside installation folder :

```
elibo@DESKTOP-AM14A6I MINGW64 /s/dev
$ git clone https://github.com/eliboa/NxNandManager
Cloning into 'NxNandManager'...
remote: Enumerating objects: 3089, done.
remote: Counting objects: 100% (894/894), done.
remote: Compressing objects: 100% (637/637), done.
remote: Total 3089 (delta 611), reused 517 (delta 253), pack-reused 2195
Receiving objects: 100% (3089/3089), 33.47 MiB | 19.52 MiB/s, done.
Resolving deltas: 100% (2081/2081), done.
```

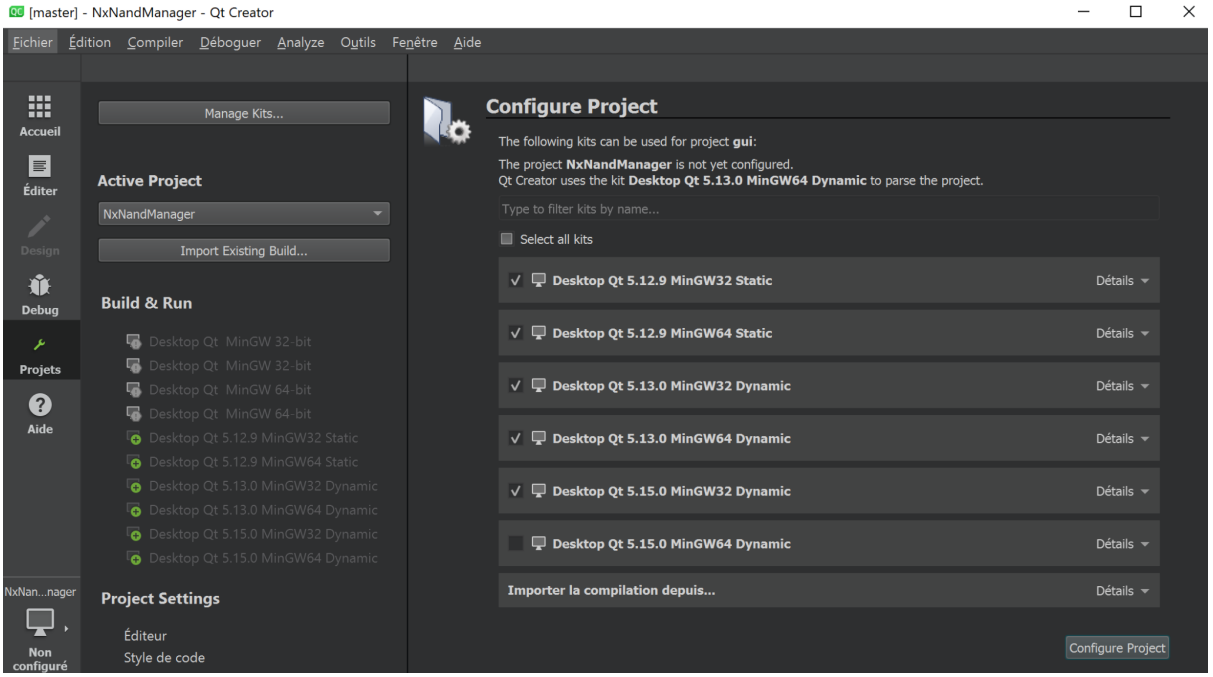
Now, download OpenSSL's pre-compiled binaries for MinGW :
https://www.eliboa.com/OpenSSL_mingw_build.rar

Extract the content of the archive in the installation folder.
This should be the content of your installation folder :

```
elibo@DESKTOP-AM14A6I MINGW64 /s/dev
$ ll
total 36636
drwxr-xr-x 1 elibo 197611        0 janv. 25 10:47 NxNandManager/
-rw-r--r-- 1 elibo 197611 37509969 janv. 25 11:23 OpenSSL_mingw_build.rar
drwxr-xr-x 1 elibo 197611        0 août  10  2019 OpenSSL_mingw32/
drwxr-xr-x 1 elibo 197611        0 août  10  2019 OpenSSL_mingw64/
```

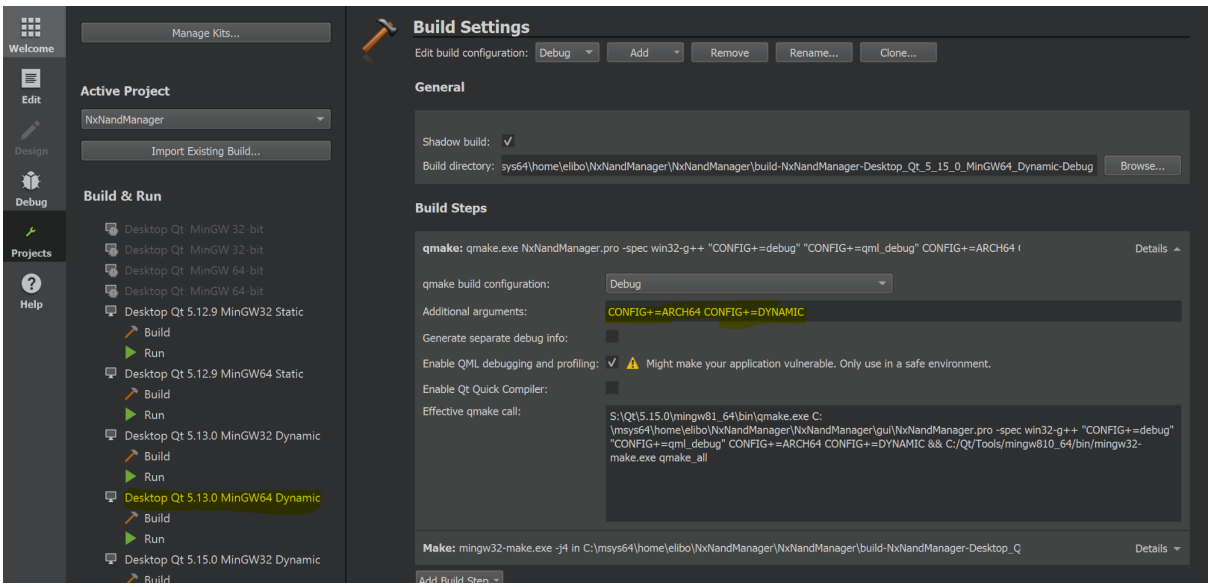
Open “QT Creator” as an administrator (important), then open project’s file : S:\dev\NxNandManager\NxNandManager\gui\NxNandManager.pro

The first time the project is opened, you’ll have some config to do. You should select every Qt kit you want to build the project with. For the purpose of this tutorial, you should at least select MinGW64 5.13+

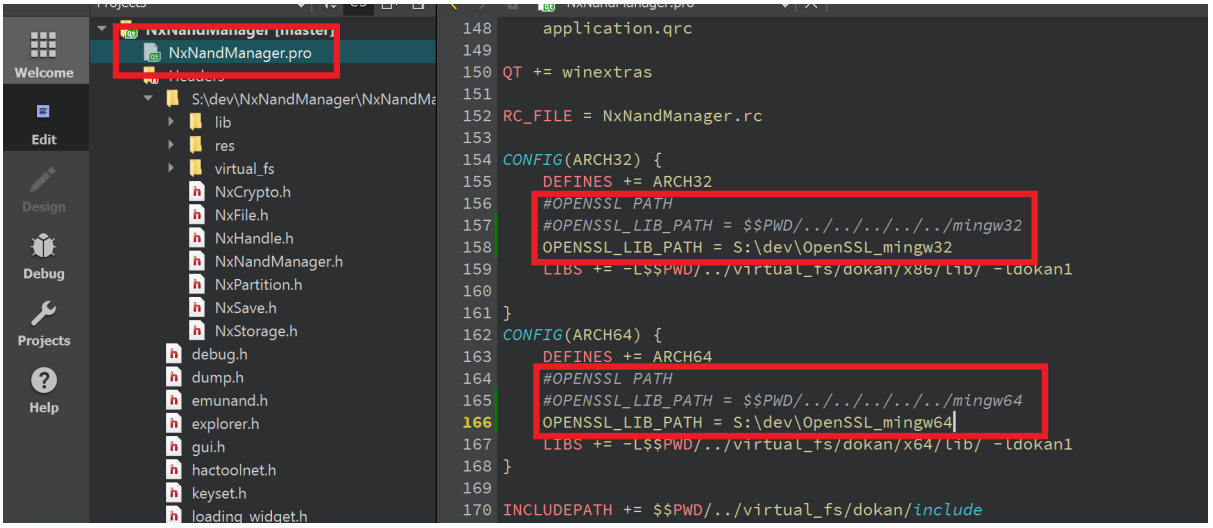


NB : I have custom kits to build statically but in your case you shouldn’t see any mention of “Static” or “Dynamic” in your kit list.

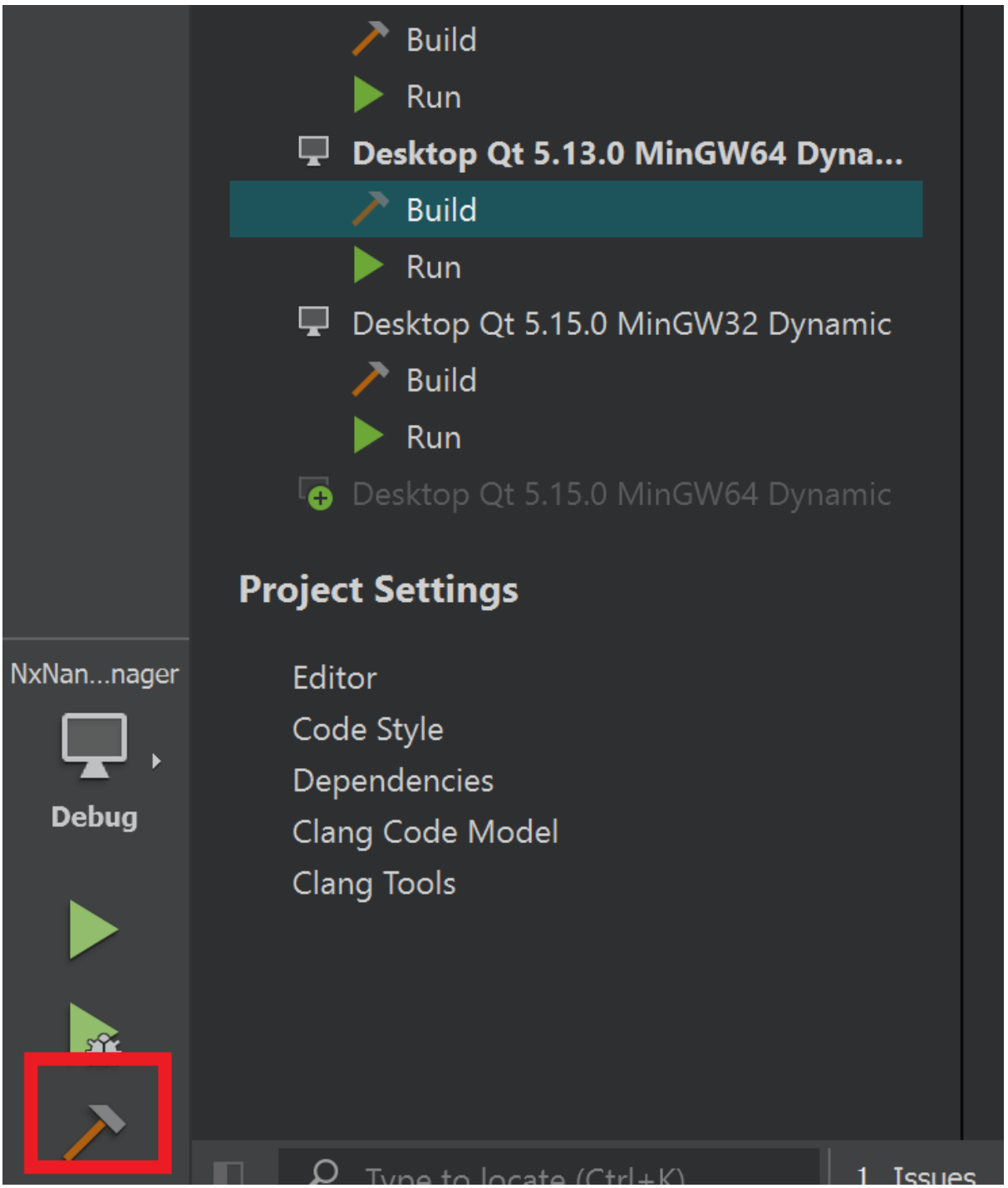
Add extra arguments in Project Settings (ex: CONFIG+=ARCH64 CONFIG+=DYNAMIC)



Set the path to OpenSSL library (OPENSSL_LIB_PATH, NxNandManager.pro file)



Now build the project :



Run :

To run the program, move a copy dokan1.dll inside the build folder.
In my case :

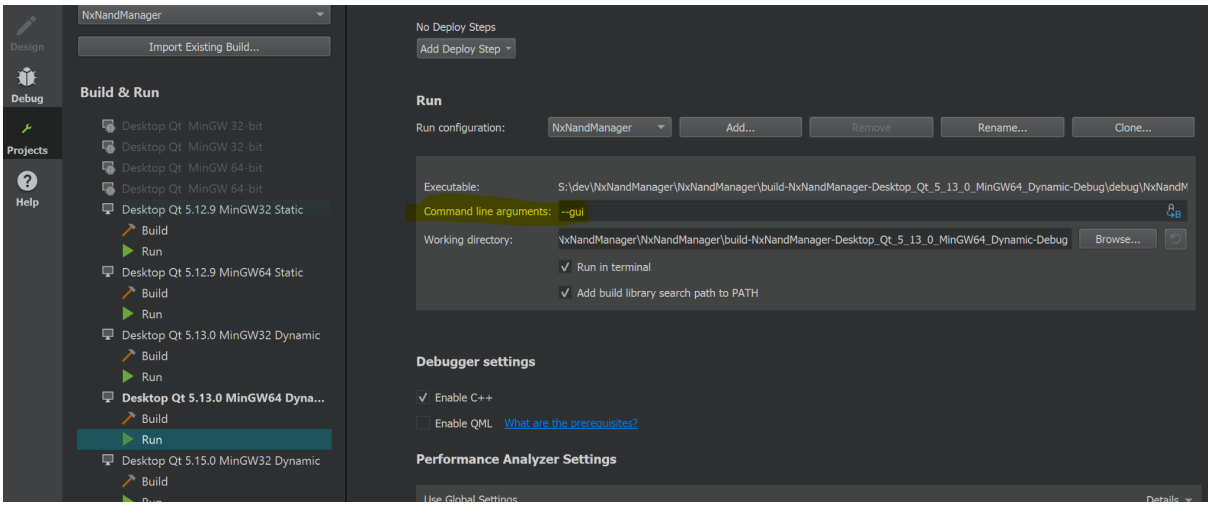
```
elibo@DESKTOP-AM14A6I MINGW64 /s/dev
$ cp NxNandManager/NxNandManager/virtual_fs/dokan/x64/dokan1.dll NxNandManager/NxNandMan
ager/build-NxNandManager-Desktop_Qt_5_13_0_MinGW64_Dynamic-Debug/M64_D
```

er > NxNandManager > build-NxNandManager-Desktop_Qt_5_13_0_MinGW64_Dynamic-Debug

Nom	Modifié le	Type	Taille
debug	25/01/2022 11:31	Dossier de fichiers	
release	25/01/2022 11:15	Dossier de fichiers	
res	25/01/2022 11:48	Dossier de fichiers	
.qmake.stash	25/01/2022 11:15	Fichier STASH	1 Ko
dokan1.dll	25/01/2022 11:52	Extension de l'applic...	513 Ko
Makefile	25/01/2022 11:27	Fichier	32 Ko
Makefile.Debug	25/01/2022 11:27	Fichier DEBUG	1 141 Ko

NB : You'll have to copy the DLL inside"build-NxNand….-Release" if you are building the "release" version (obviously).

Go to "Projects" then select MinGW64 Kit "Run" and add –gui argument :



You can now run the program :)

Deploy :

Default Qt framework is build dynamically so you'll need to deploy the project if you want to run the program outside Qt's environment (Qt Creator), a.k.a put all the required DLL's in the same folder as your executable :

- Either use Qt "Windows Deployment Tool" : https://doc.qt.io/qt-5/windows-deployment.html
- Or grab all the *.dll files inside this previous build of NxNandManager (and put them in the same folder as NxNandManager.exe. Don't forget dokan1.dll (not in the archive, you can grab a copy here)