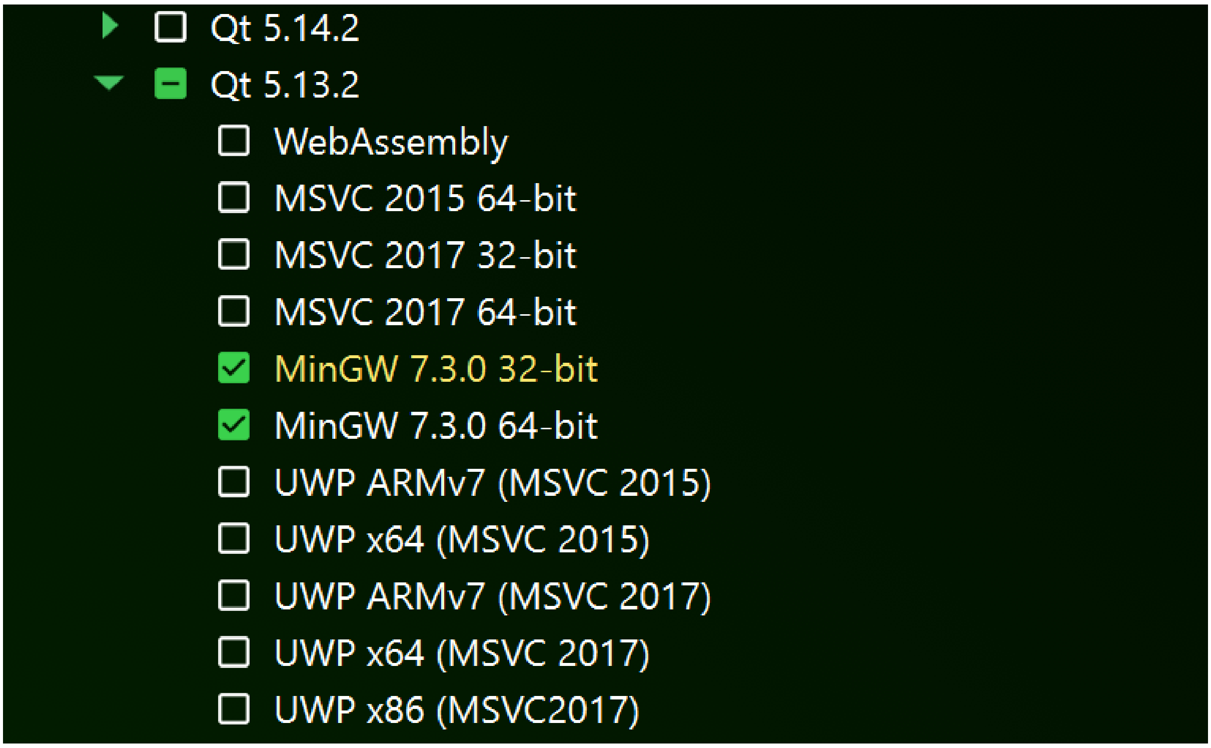
NxNandManager : Set up and build project with Qt (GUI)

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

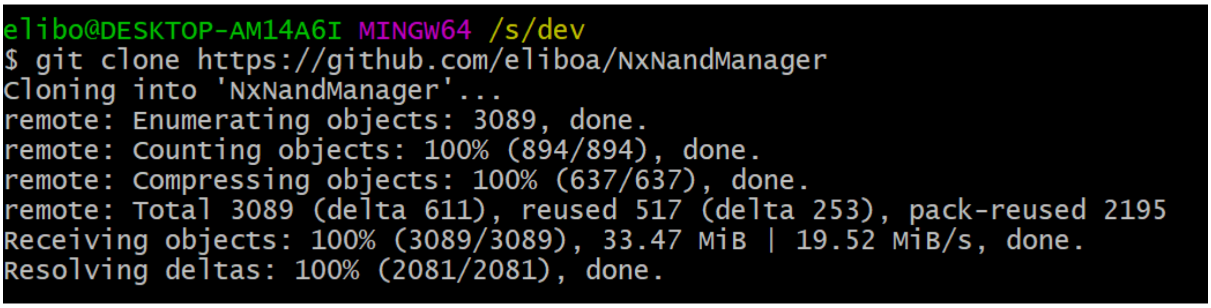


- ▶ □ Qt 5.14.2
- ▼ ▣ Qt 5.13.2
  - □ WebAssembly
  - □ MSVC 2015 64-bit
  - □ MSVC 2017 32-bit
  - □ MSVC 2017 64-bit
  - ☑ MinGW 7.3.0 32-bit
  - ☑ MinGW 7.3.0 64-bit
  - □ UWP ARMv7 (MSVC 2015)
  - □ UWP x64 (MSVC 2015)
  - □ UWP ARMv7 (MSVC 2017)
  - □ UWP x64 (MSVC 2017)
  - □ UWP x86 (MSVC2017)

Set up & build project :

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

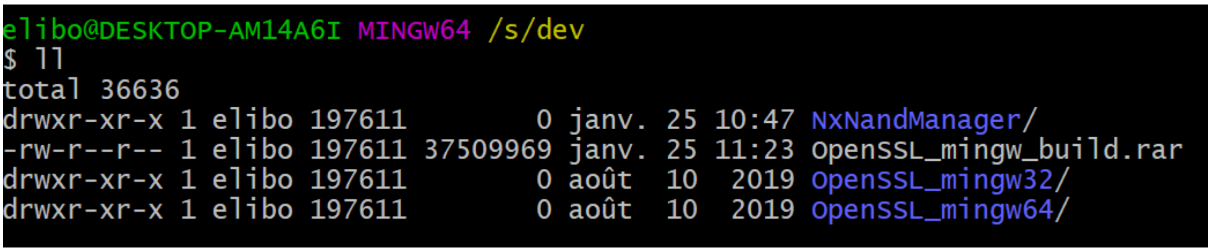
First aff all, clone NxNandManager's git repo inside installation folder :



```
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

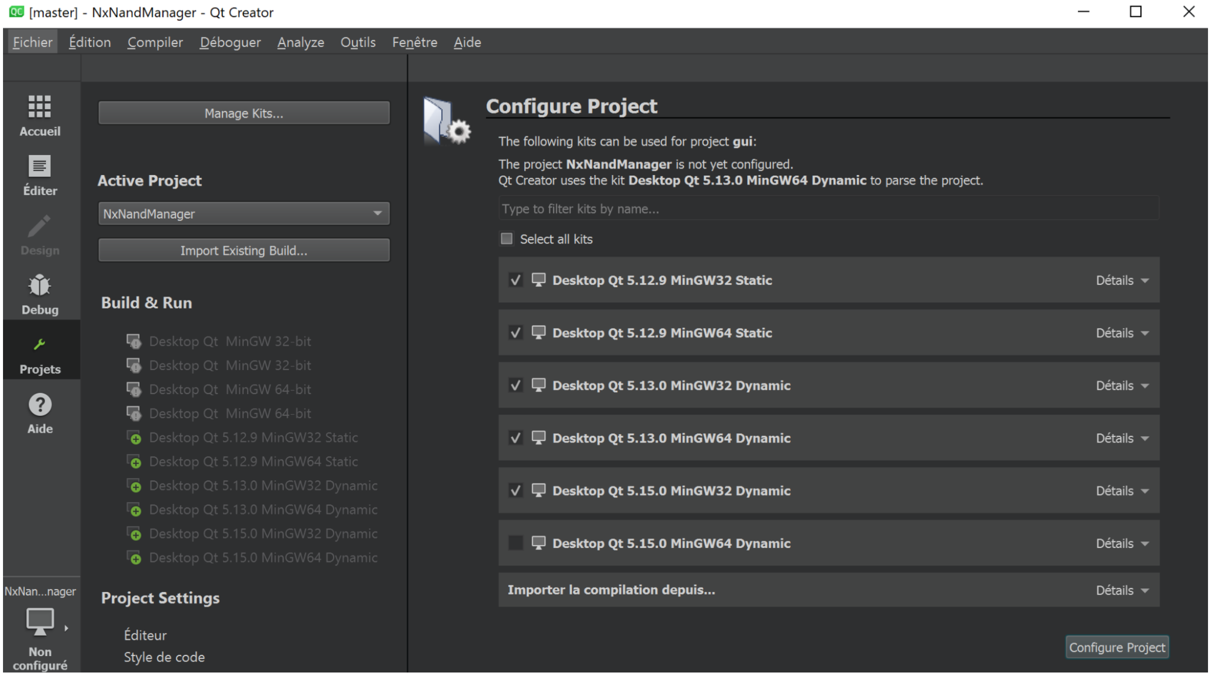
Extract the content of the archive in the installation folder.
This should be the content of your installation folder :



```
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+



Configure Project

The following kits can be used for project **gui**:

The project **NxNandManager** is not yet configured.
Qt Creator uses the kit **Desktop Qt 5.13.0 MinGW64 Dynamic** to parse the project.

Type to filter kits by name...

□ Select all kits

☑ **Desktop Qt 5.12.9 MinGW32 Static** Détails

☑ **Desktop Qt 5.12.9 MinGW64 Static** Détails

☑ **Desktop Qt 5.13.0 MinGW32 Dynamic** Détails

☑ **Desktop Qt 5.13.0 MinGW64 Dynamic** Détails

☑ **Desktop Qt 5.15.0 MinGW32 Dynamic** Détails

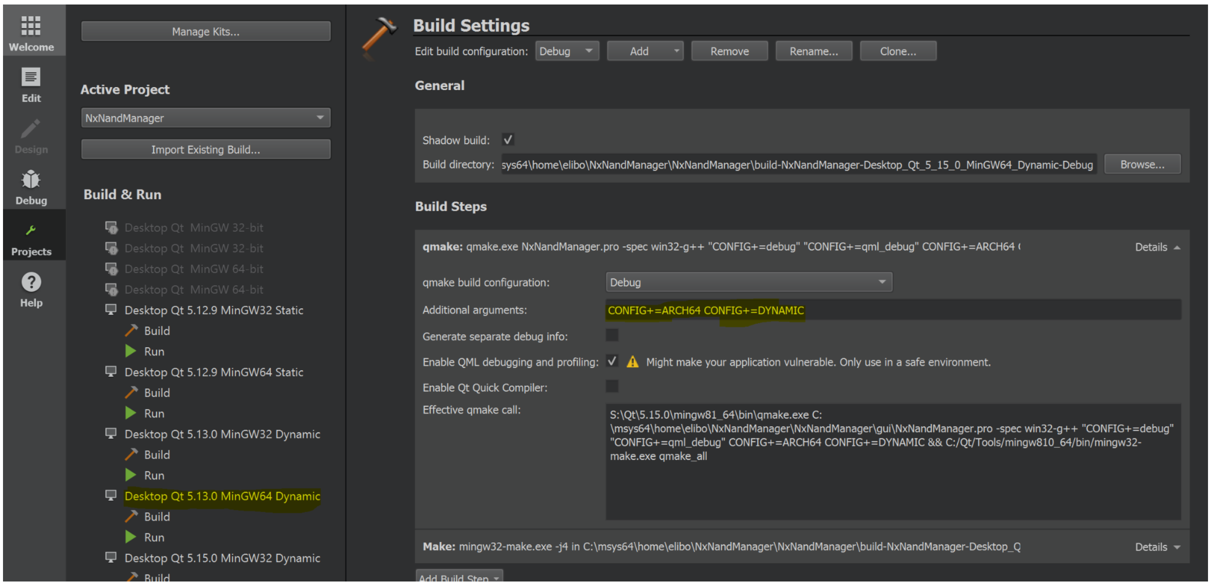
□ **Desktop Qt 5.15.0 MinGW64 Dynamic** Détails

Importer la compilation depuis... Détails

Configure Project

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)



Active Project: NxNandManager

Build & Run
- Desktop Qt MinGW 32-bit
- Desktop Qt MinGW 32-bit
- Desktop Qt MinGW 64-bit
- Desktop Qt MinGW 64-bit
- Desktop Qt 5.12.9 MinGW32 Static
  - Build
  - Run
- Desktop Qt 5.12.9 MinGW64 Static
  - Build
  - Run
- Desktop Qt 5.13.0 MinGW32 Dynamic
  - Build
  - Run
- Desktop Qt 5.13.0 MinGW64 Dynamic
  - Build
  - Run
- Desktop Qt 5.15.0 MinGW32 Dynamic
  - Build

Build Settings

Edit build configuration: Debug

General

Shadow build: ☑

Build directory: sys64\home\elibo\NxNandManager\NxNandManager\build-NxNandManager-Desktop_Qt_5_15_0_MinGW64_Dynamic-Debug

Build Steps

qmake: qmake.exe NxNandManager.pro -spec win32-g++ "CONFIG+=debug" "CONFIG+=qml_debug" CONFIG+=ARCH64 (

qmake build configuration: Debug

Additional arguments: CONFIG+=ARCH64 CONFIG+=DYNAMIC

Generate separate debug info: □

Enable QML debugging and profiling: ☑ ⚠ Might make your application vulnerable. Only use in a safe environment.

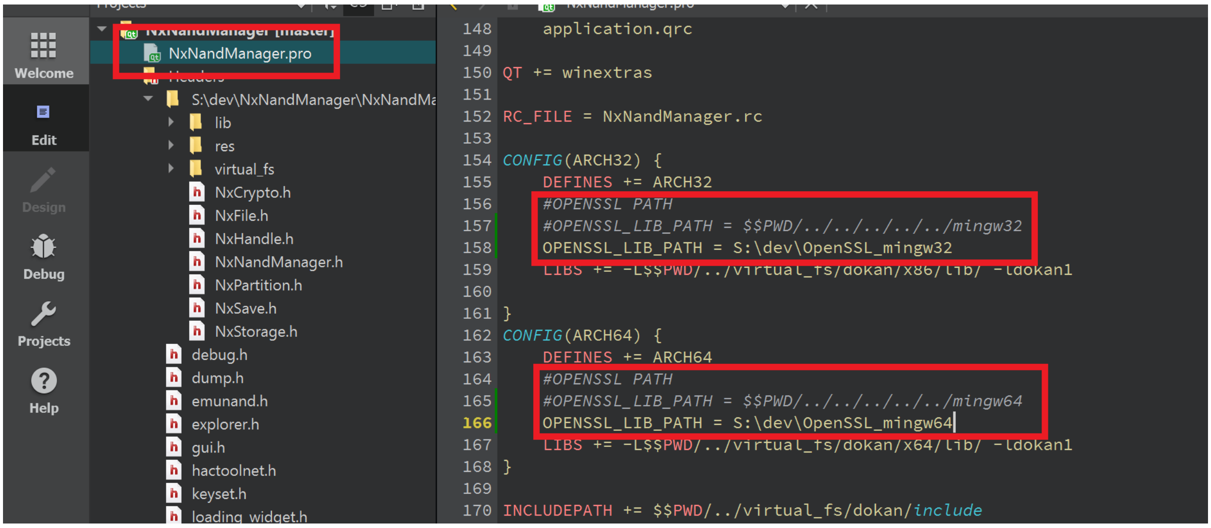
Enable Qt Quick Compiler: □

Effective qmake call:

```
S:\Qt\5.15.0\mingw81_64\bin\qmake.exe C:
\msys64\home\elibo\NxNandManager\NxNandManager\gui\NxNandManager.pro -spec win32-g++ "CONFIG+=debug"
"CONFIG+=qml_debug" CONFIG+=ARCH64 CONFIG+=DYNAMIC && C:/Qt/Tools/mingw810_64/bin/mingw32-
make.exe qmake_all
```

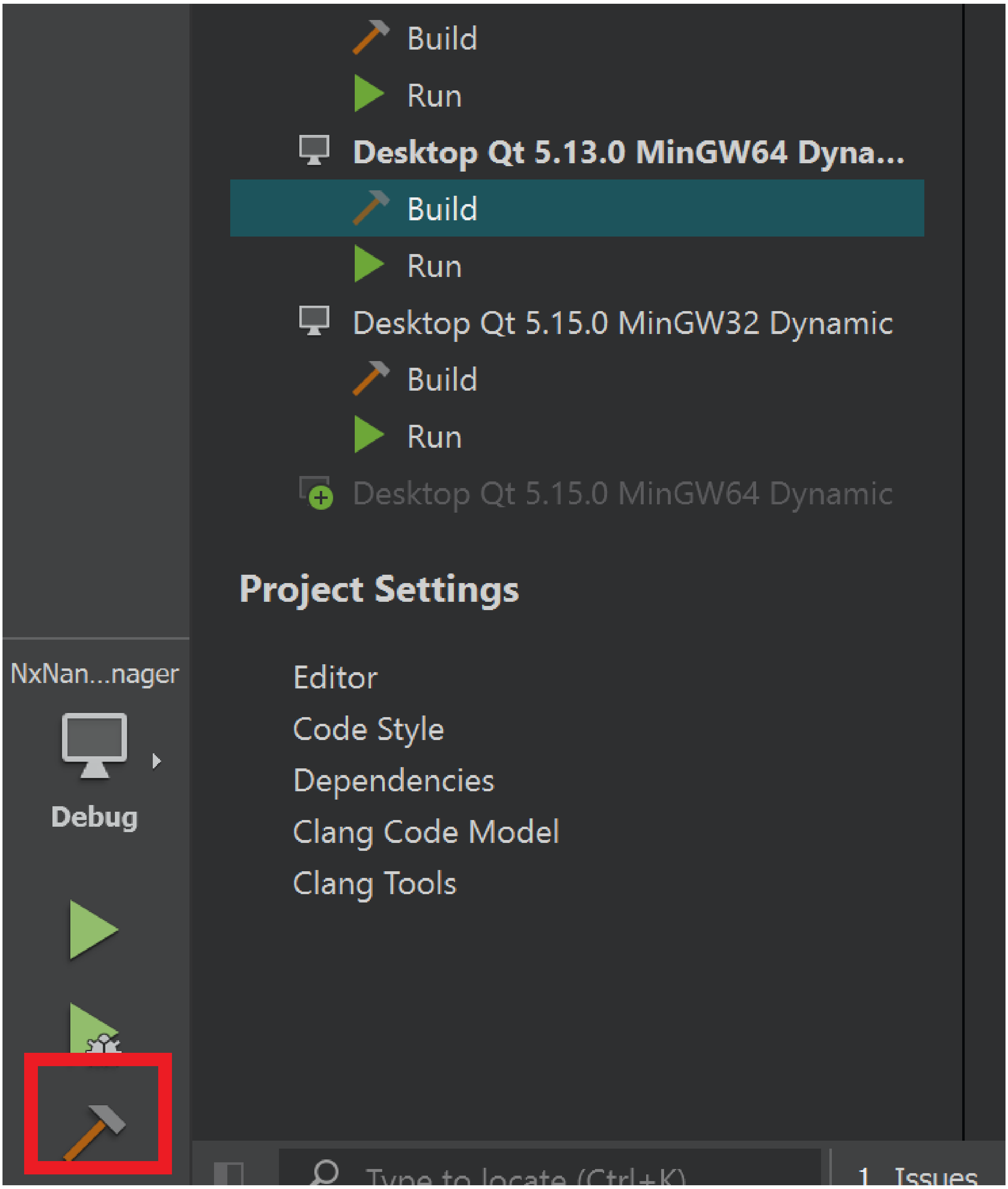
Make: mingw32-make.exe -j4 in C:\msys64\home\elibo\NxNandManager\NxNandManager\build-NxNandManager-Desktop_Q

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



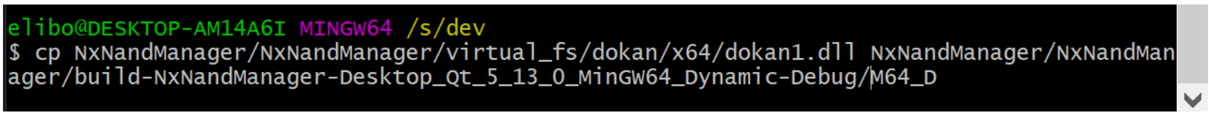
```
    application.qrc

QT += winextras

RC_FILE = NxNandManager.rc

CONFIG(ARCH32) {
    DEFINES += ARCH32
    #OPENSSL PATH
    #OPENSSL_LIB_PATH = $$PWD/../../../../../mingw32
    OPENSSL_LIB_PATH = S:\dev\OpenSSL_mingw32
    LIBS += -L$$PWD/../virtual_fs/dokan/x86/lib/ -ldokan1

}
CONFIG(ARCH64) {
    DEFINES += ARCH64
    #OPENSSL PATH
    #OPENSSL_LIB_PATH = $$PWD/../../../../../mingw64
    OPENSSL_LIB_PATH = S:\dev\OpenSSL_mingw64
    LIBS += -L$$PWD/../virtual_fs/dokan/x64/lib/ -ldokan1
}

INCLUDEPATH += $$PWD/../virtual_fs/dokan/include
```

Now build the project :

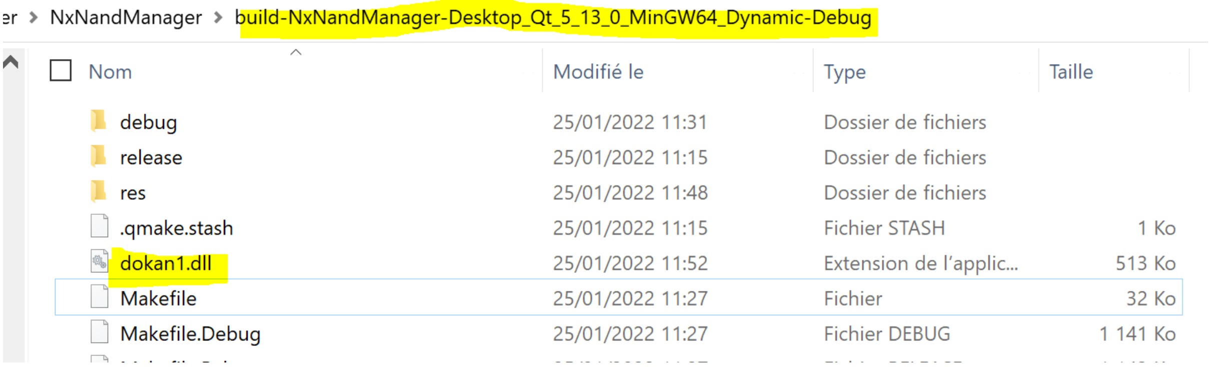


Run :

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



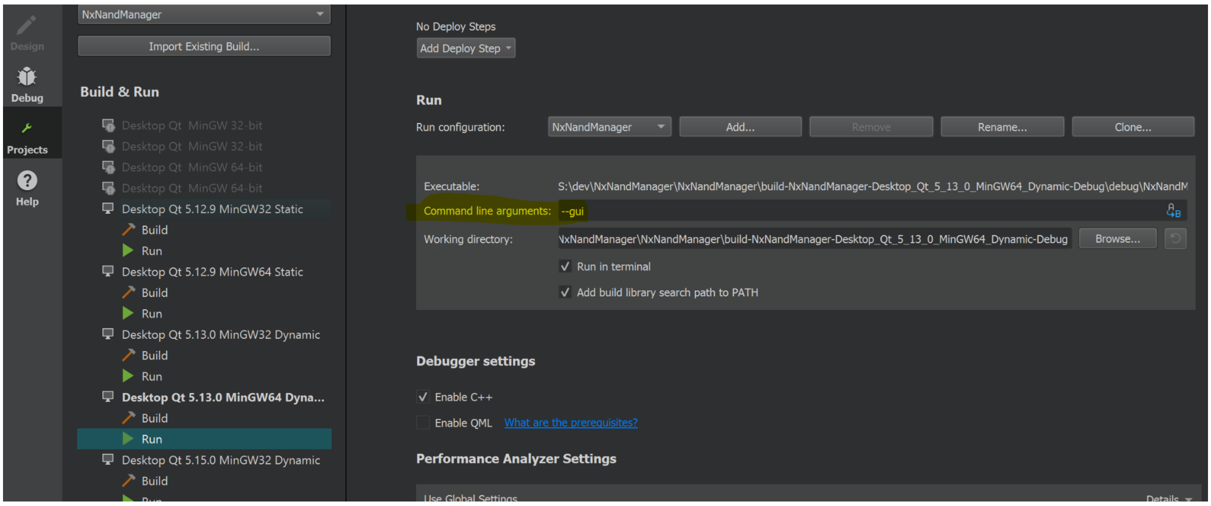
```
elibo@DESKTOP-AM14A6I MINGW64 /s/dev
$ cp NxNandManager/NxNandManager/virtual_fs/dokan/x64/dokan1.dll NxNandManager/NxNandMan
ager/build-NxNandManager-Desktop_Qt_5_13_0_MinGW64_Dynamic-Debug/M64_D
```



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

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



Run

Run configuration: NxNandManager

Executable: S:\dev\NxNandManager\NxNandManager\build-NxNandManager-Desktop_Qt_5_13_0_MinGW64_Dynamic-Debug\debug\NxNandM

Command line arguments: --gui

Working directory: \xNandManager\NxNandManager\build-NxNandManager-Desktop_Qt_5_13_0_MinGW64_Dynamic-Debug

☑ Run in terminal

☑ Add build library search path to PATH

Debugger settings

☑ Enable C++

□ Enable QML

Performance Analyzer Settings

You can now run the program :)