

N'GINE - CODE::BLOCKS Configuration

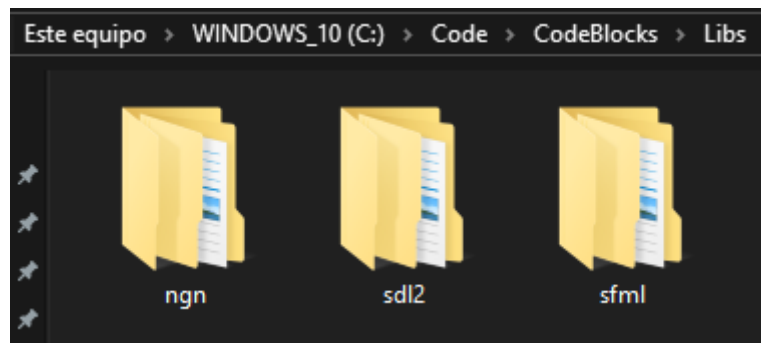
Installation of Libraries - Windows

We will begin by downloading the additional libraries from their official websites:

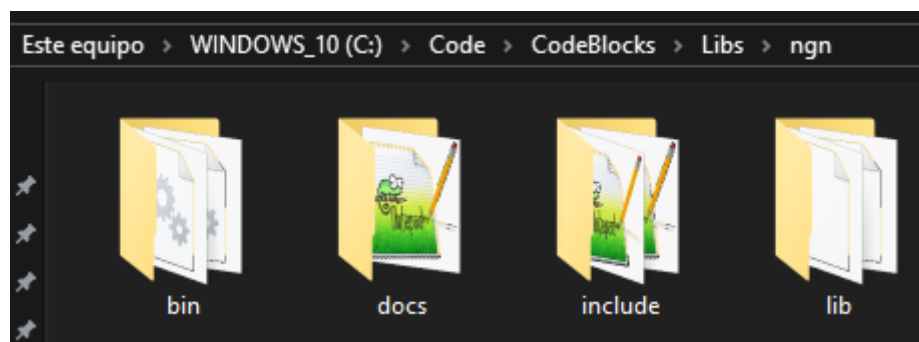
SDL2: <https://www.libsdl.org/download-2.0.php>

SFML: <https://www.sfml-dev.org/>

Next, we will create a folder named "Libs" within the directory where CODE::BLOCKS is installed. Inside this folder, we will create the "ngn," "sdl2," and "sfml" folders.

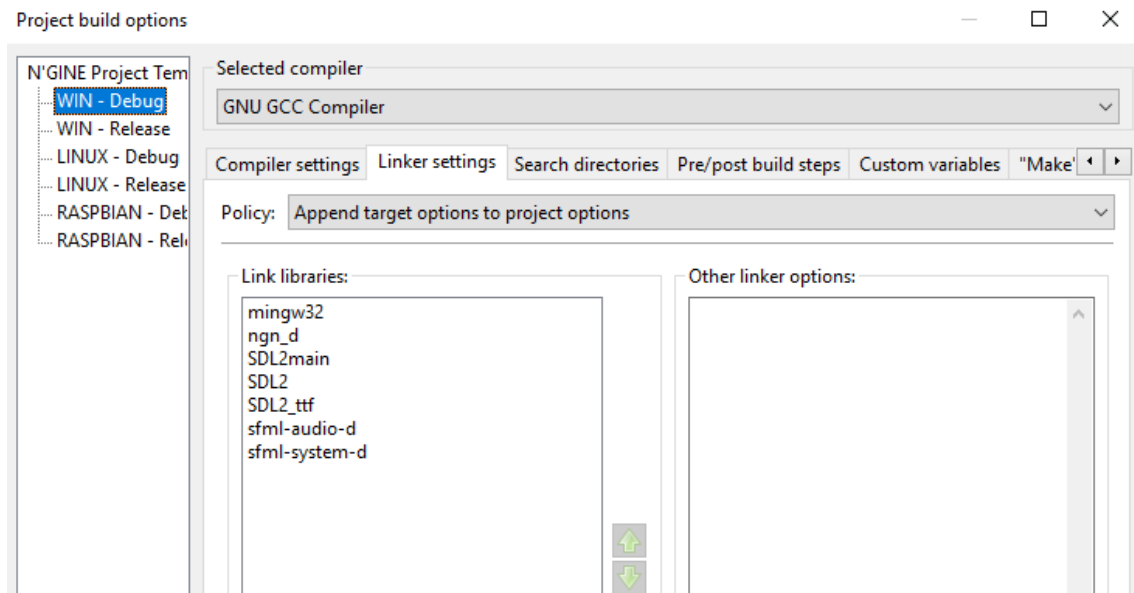


We will copy the "include", "lib", and optionally "bin" and "docs" directories corresponding to each library into each folder.

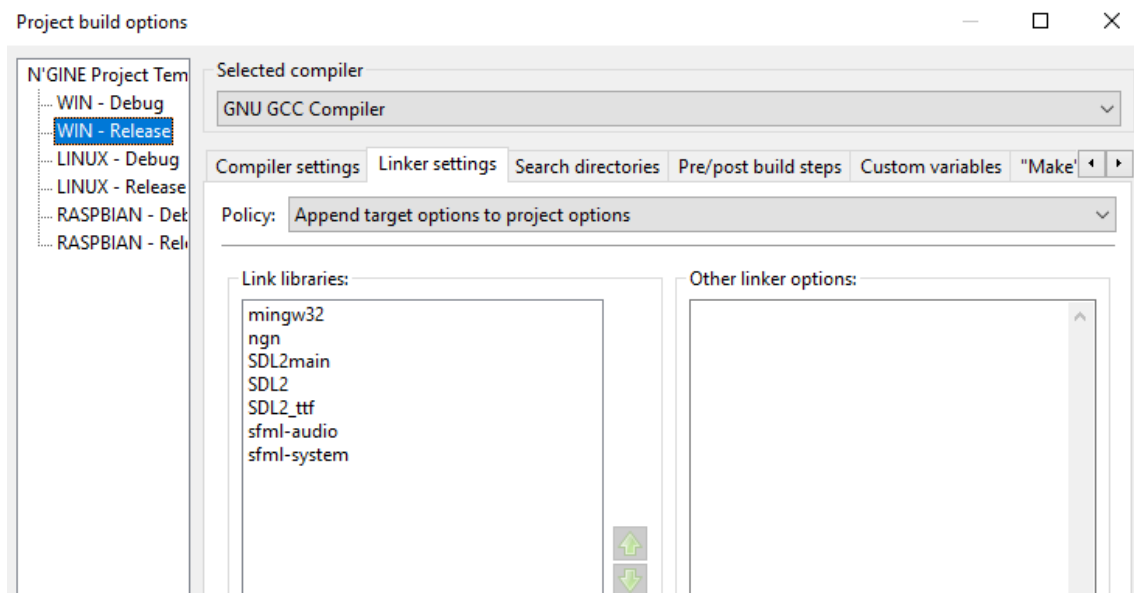


Project Configuration - CODE::BLOCKS

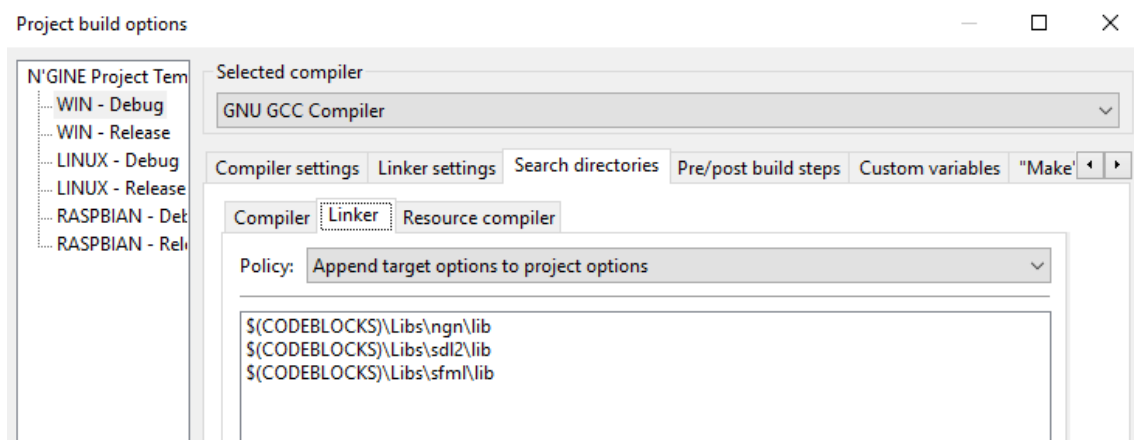
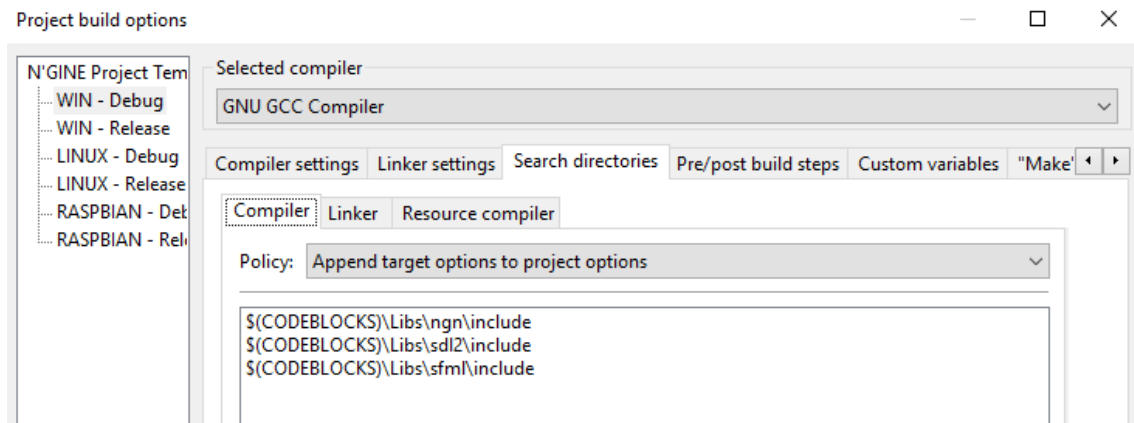
In “Project build options”, in the “WIN-Debug” section, we will configure the following parameters:”



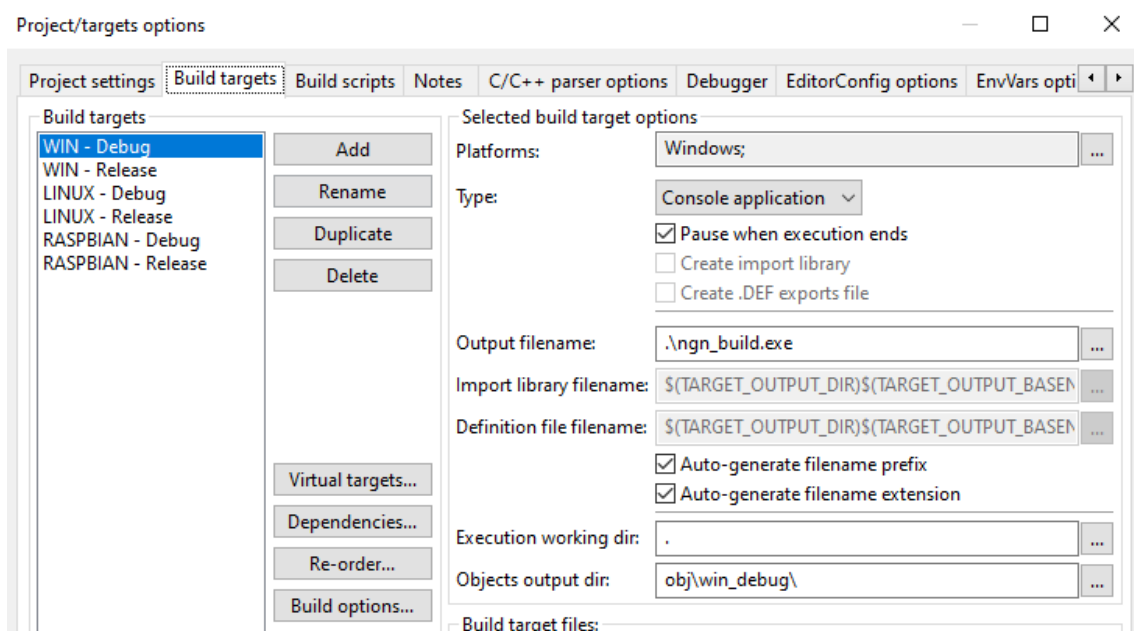
In “Project build options”, in the “WIN-Release” section, we will configure the following parameters:

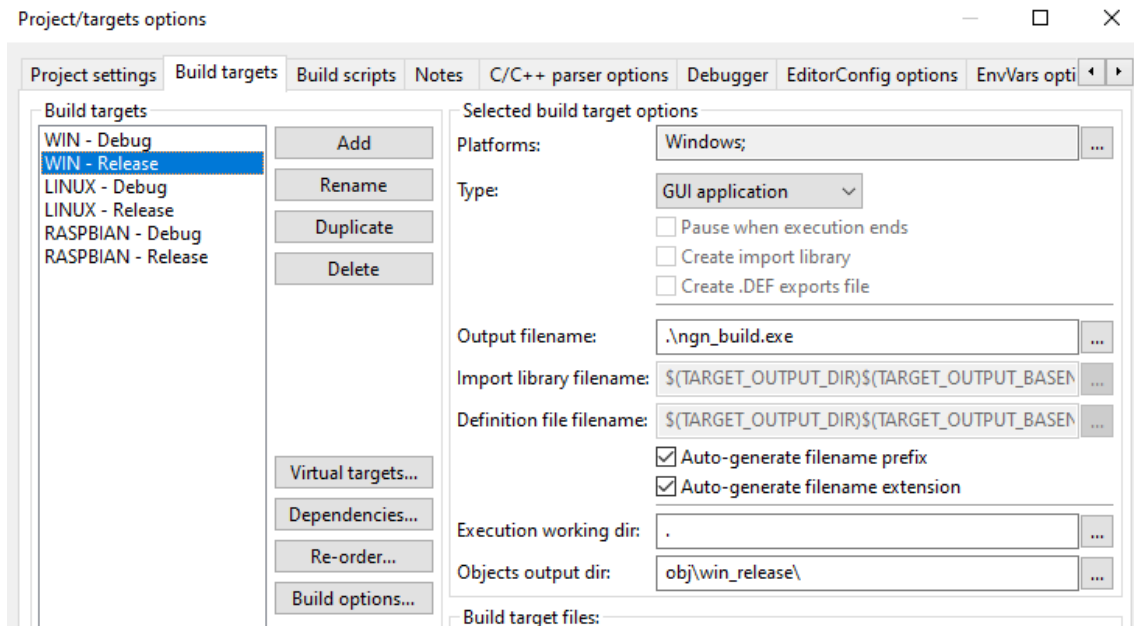


In “Project build options”, we will configure the following parameters in both the “WIN-DEBUG” and “WIN-RELEASE” sections:



In “Project/target options”, we will configure the following parameters:





MinGW Compiler Update on CODE::BLOCKS 20 – (Windows)

The version 20 of CODE::BLOCKS does not include the latest version of the MinGW compiler (MinGW-W64 project, version 8.1.0, 32/64 bit, SEH). If you have installed CODE::BLOCKS that includes this compiler, or if it is not installed (a clean installation of CODE::BLOCKS), you will need to install or change the compiler version. To perform this installation or update, follow these steps:

1 – Download the MSYS2 utility from its official website:

<https://www.msys2.org/#installation>

The website itself contains a tutorial on how to perform the installation of the utility.

2 – Once installed, proceed to install the latest version of the MinGW compiler. From the console that opens with the “MSYS2 MINGW64” icon, execute the following commands:

```
pacman -Syu
```

Follow the on-screen instructions to update the package repository.

Once the installation is complete, you should see something similar to this:

```
msys2-launcher-1.5-3-x86_64      96.4 KiB   227 KiB/s  00:00 [#####] 100%
libxcrypt-4.4.37-1-x86_64      76.9 KiB   184 KiB/s  00:00 [#####] 100%
Total (13/13)                   2.9 MiB   1845 KiB/s  00:02 [#####] 100%
(13/13) checking keys in keyring [#####] 100%
(13/13) checking package integrity [#####] 100%
(13/13) loading package files [#####] 100%
(13/13) checking for file conflicts [#####] 100%
(13/13) checking available disk space [#####] 100%
:: Processing package changes...
( 1/13) upgrading bash-completion [#####] 100%
( 2/13) upgrading libiconv [#####] 100%
( 3/13) upgrading libxcrypt [#####] 100%
( 4/13) upgrading libssh2 [#####] 100%
( 5/13) upgrading libedit [#####] 100%
( 6/13) upgrading libcurl [#####] 100%
( 7/13) upgrading curl [#####] 100%
( 8/13) upgrading libargp [#####] 100%
( 9/13) upgrading getent [#####] 100%
(10/13) upgrading libhogweed [#####] 100%
(11/13) upgrading libnettle [#####] 100%
(12/13) upgrading msys2-launcher [#####] 100%
(13/13) upgrading nettle [#####] 100%
:: Running post-transaction hooks...
(1/1) Updating the info directory file...

NightFox@FOXCAVE MINGW64 ~
$
```

3 - 3 - Now, install the latest version of the MinGW compiler by typing the following command in the console:

```
pacman -S mingw-w64-x86_64-toolchain
```

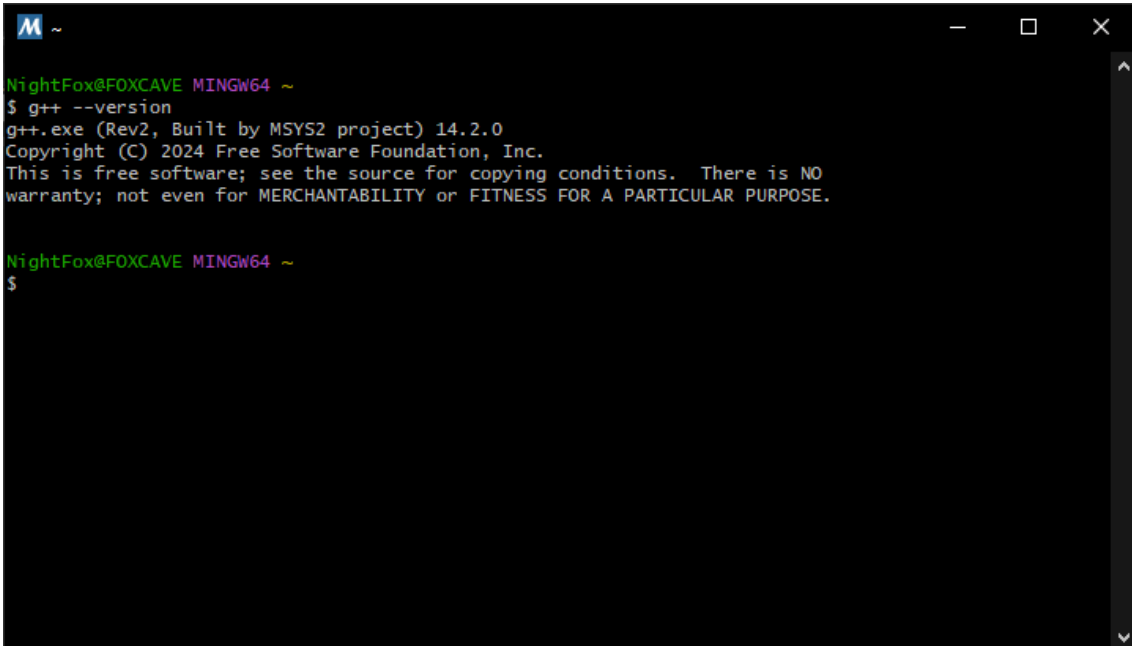
After finishing, you should see something like this:

```
Optional dependencies for mingw-w64-x86_64-openssl
mingw-w64-x86_64-ca-certificates
(25/40) installing mingw-w64-x86_64-termcap [#####] 100%
(26/40) installing mingw-w64-x86_64-readline [#####] 100%
(27/40) installing mingw-w64-x86_64-sqlite3 [#####] 100%
Optional dependencies for mingw-w64-x86_64-sqlite3
mingw-w64-x86_64-tcl: for sqlite3_analyzer [pending]
(28/40) installing mingw-w64-x86_64-tcl [#####] 100%
(29/40) installing mingw-w64-x86_64-tk [#####] 100%
(30/40) installing mingw-w64-x86_64-xz [#####] 100%
(31/40) installing mingw-w64-x86_64-tzdata [#####] 100%
(32/40) installing mingw-w64-x86_64-python [#####] 100%
(33/40) installing mingw-w64-x86_64-xxhash [#####] 100%
(34/40) installing mingw-w64-x86_64-gdb [#####] 100%
Optional dependencies for mingw-w64-x86_64-gdb
mingw-w64-x86_64-python-pygments: for syntax highlighting
(35/40) installing mingw-w64-x86_64-gdb-multiarch [#####] 100%
Optional dependencies for mingw-w64-x86_64-gdb-multiarch
mingw-w64-x86_64-python-pygments: for syntax highlighting
(36/40) installing mingw-w64-x86_64-libmangle-git [#####] 100%
(37/40) installing mingw-w64-x86_64-make [#####] 100%
(38/40) installing mingw-w64-x86_64-pkgconf [#####] 100%
(39/40) installing mingw-w64-x86_64-tools-git [#####] 100%
(40/40) installing mingw-w64-x86_64-winstorescompat-git [#####] 100%

NightFox@FOXCAVE MINGW64 ~
$
```

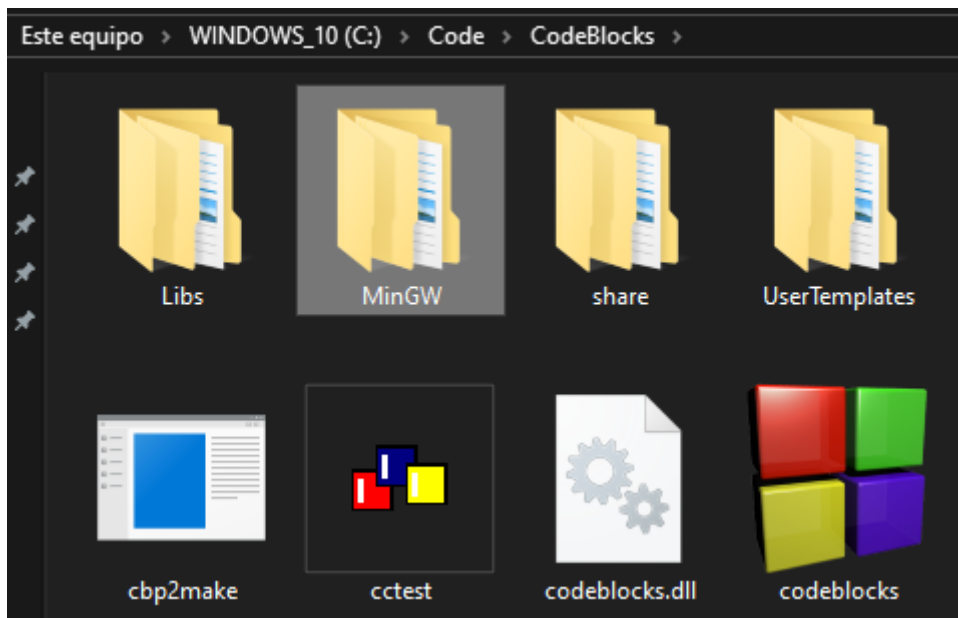
4 - Verify the installed compiler version using the command:

```
g++ --version
```

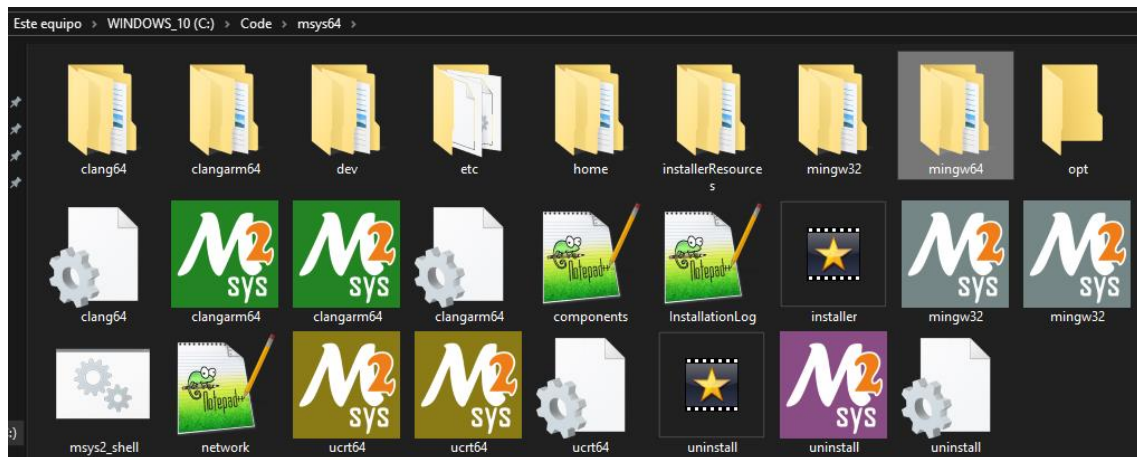
A terminal window with a black background and white text. The window title bar shows a blue 'M' icon and a tilde '~'. The prompt is 'NightFox@FOXCAVE MINGW64 ~'. The command '\$ g++ --version' has been entered, and the output is displayed: 'g++.exe (Rev2, Built by MSYS2 project) 14.2.0', 'Copyright (C) 2024 Free Software Foundation, Inc.', and a disclaimer: 'This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.' The prompt '\$' is shown again at the bottom.

```
NightFox@FOXCAVE MINGW64 ~  
$ g++ --version  
g++.exe (Rev2, Built by MSYS2 project) 14.2.0  
Copyright (C) 2024 Free Software Foundation, Inc.  
This is free software; see the source for copying conditions. There is NO  
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  
  
NightFox@FOXCAVE MINGW64 ~  
$
```

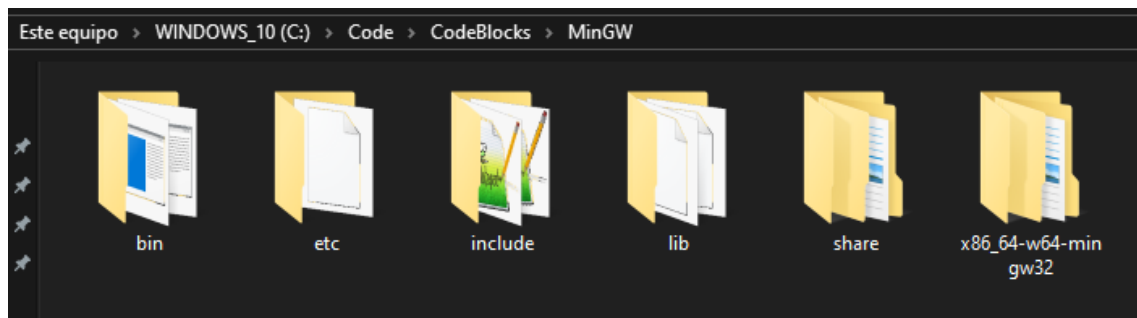
5 - In the folder where CODE::BLOCKS is installed, create a folder named MinGW. If it already exists, delete all its contents.



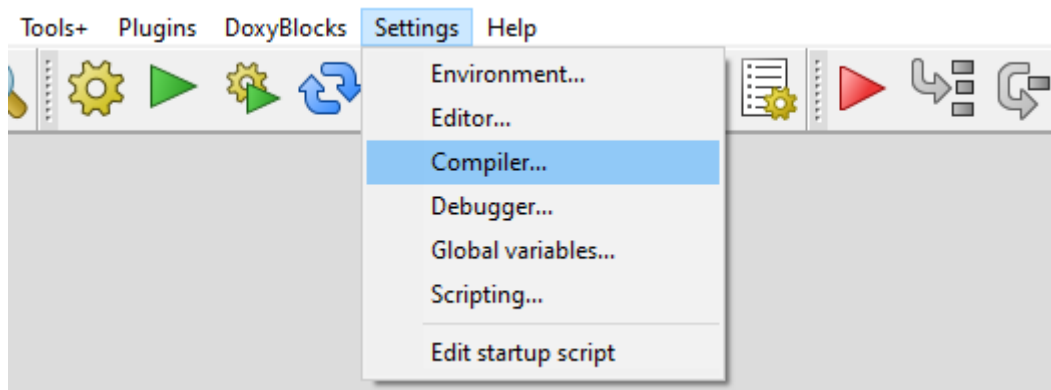
6 - Locate the MSYS2 utility installation folder and search for the “mingw64” folder.



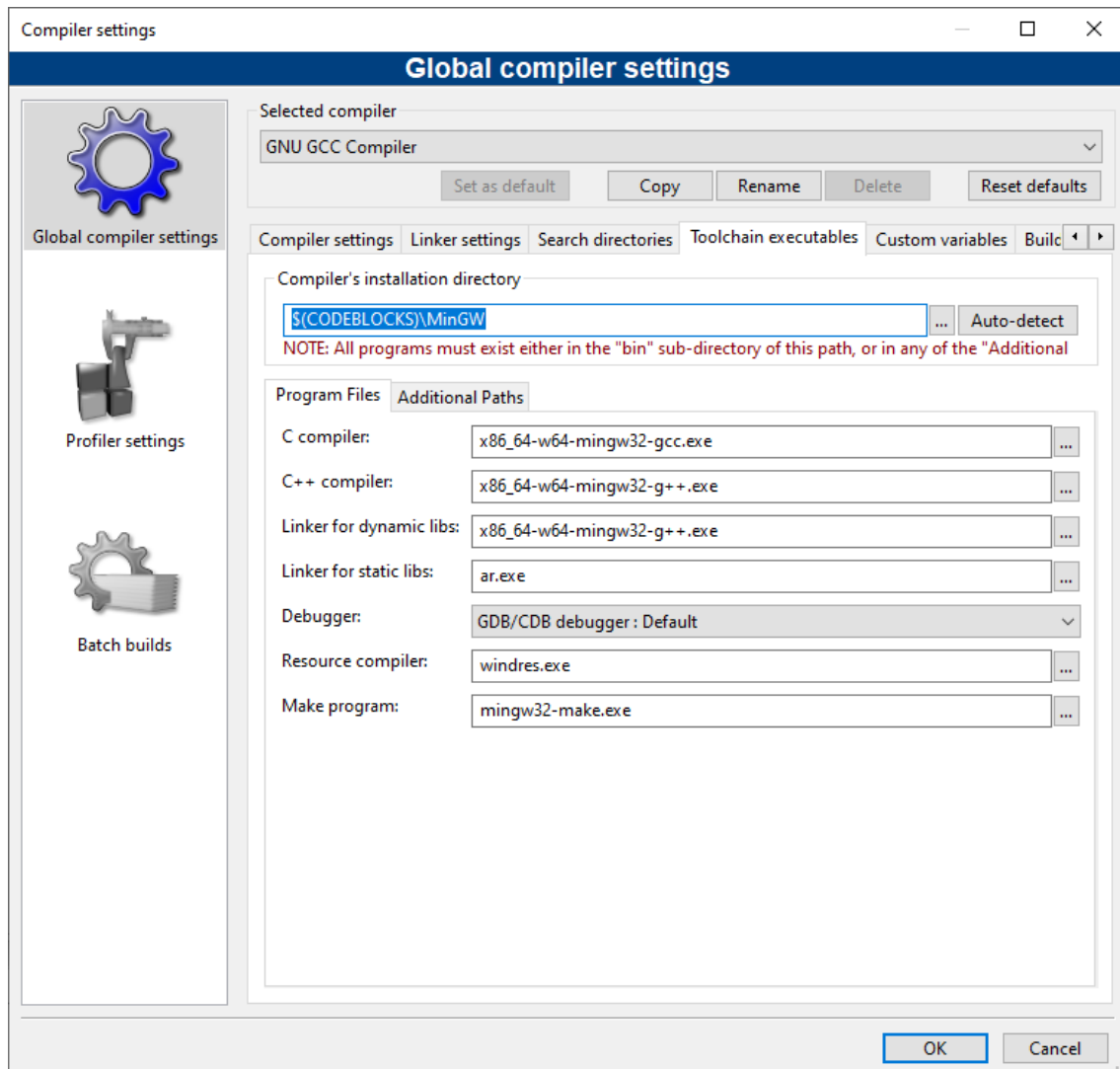
7 - Copy all the contents of this folder and paste it inside the MinGW folder that you previously created in the CODE::BLOCKS installation directory.



8 - Run CODE::BLOCKS so it detects the compiler you just installed. If it does not detect it automatically, go to SETTINGS, COMPILER:



Go to the TOOLCHAIN EXECUTABLES tab and enter the path where the compiler files are located. If you want it to be portable, you can use the system variable \$(CODEBLOCKS) as the path. The AUTO-DETECT button should detect the necessary executables.



Once done, CODE::BLOCKS should compile without issues.

Installation of Libraries - Linux and Raspberry PI OS

From the terminal, we will search for the SDL2 libraries with the following command:

```
sudo apt-cache search libsdl2
```

Next, we will install the libraries with the following command:

```
sudo apt update  
sudo apt install libsdl2-dev
```

We will search for the SDL2-TTF libraries with the following command:

```
sudo apt-cache search libsdl2-ttf
```

Then, we will install the libraries with the following command:

```
sudo apt update  
sudo apt install libsdl2-ttf-dev
```

We will search for the SFML libraries with the following command:

```
sudo apt-cache search sfml
```

Next, we will install the libraries with the following command:

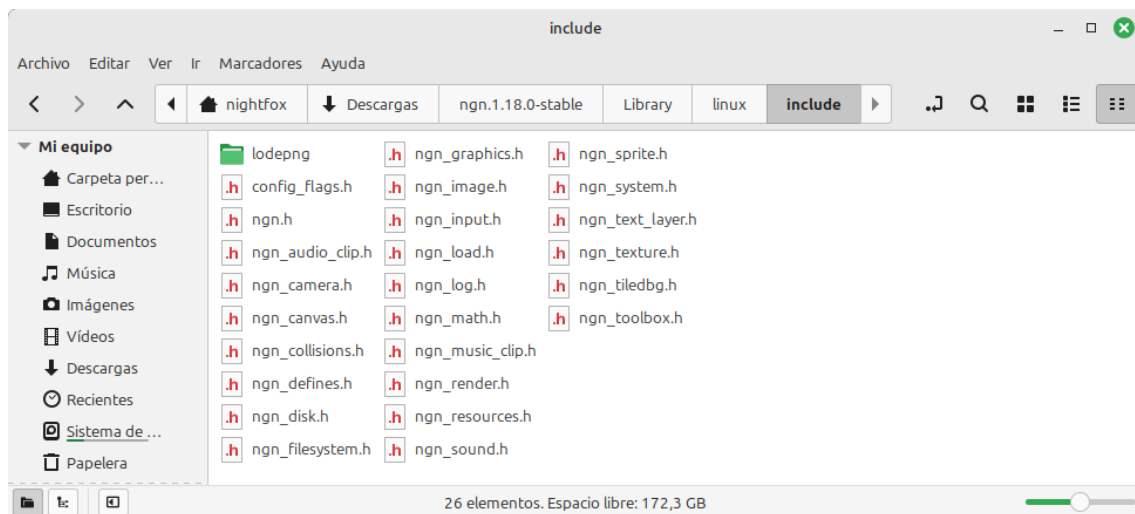
```
sudo apt update  
sudo apt install libsFML-dev
```

It's possible that some C++ compiler dependencies are missing. In that case, we will install them with the following commands:

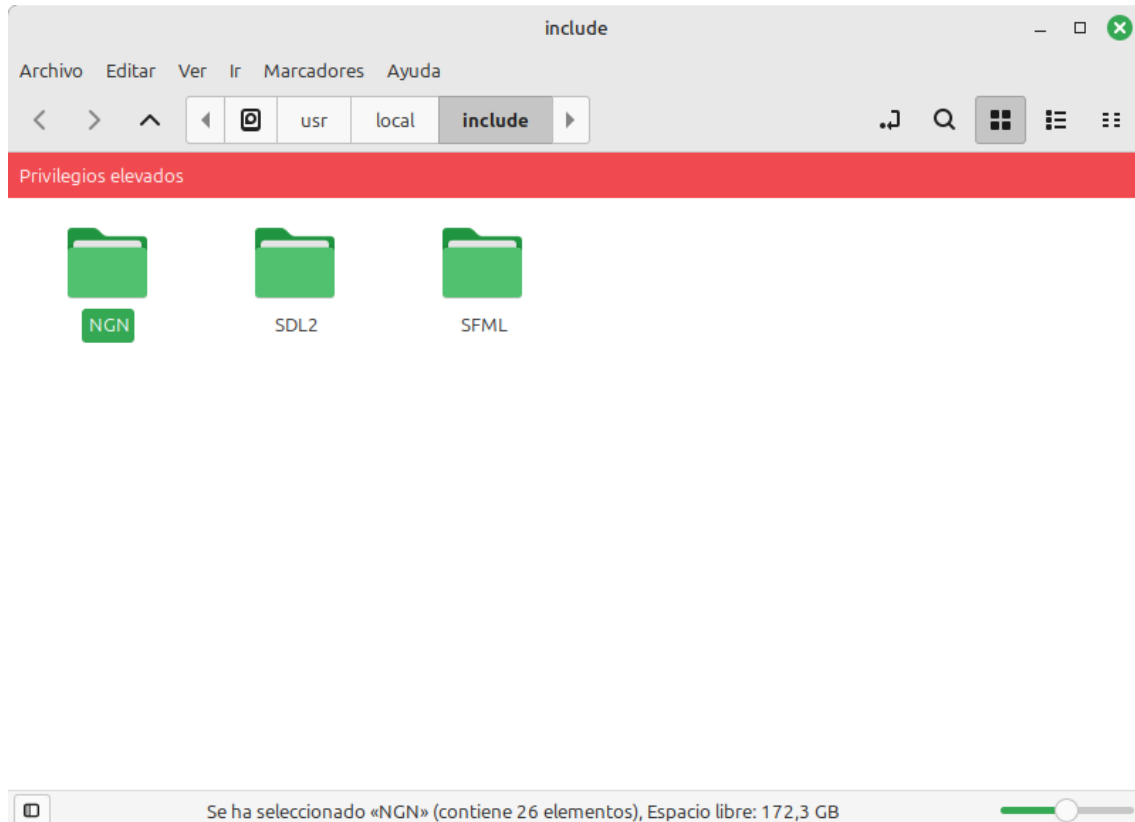
```
sudo apt update  
sudo apt install build-essential
```

For the installation of the N'gine library, we will follow these steps:

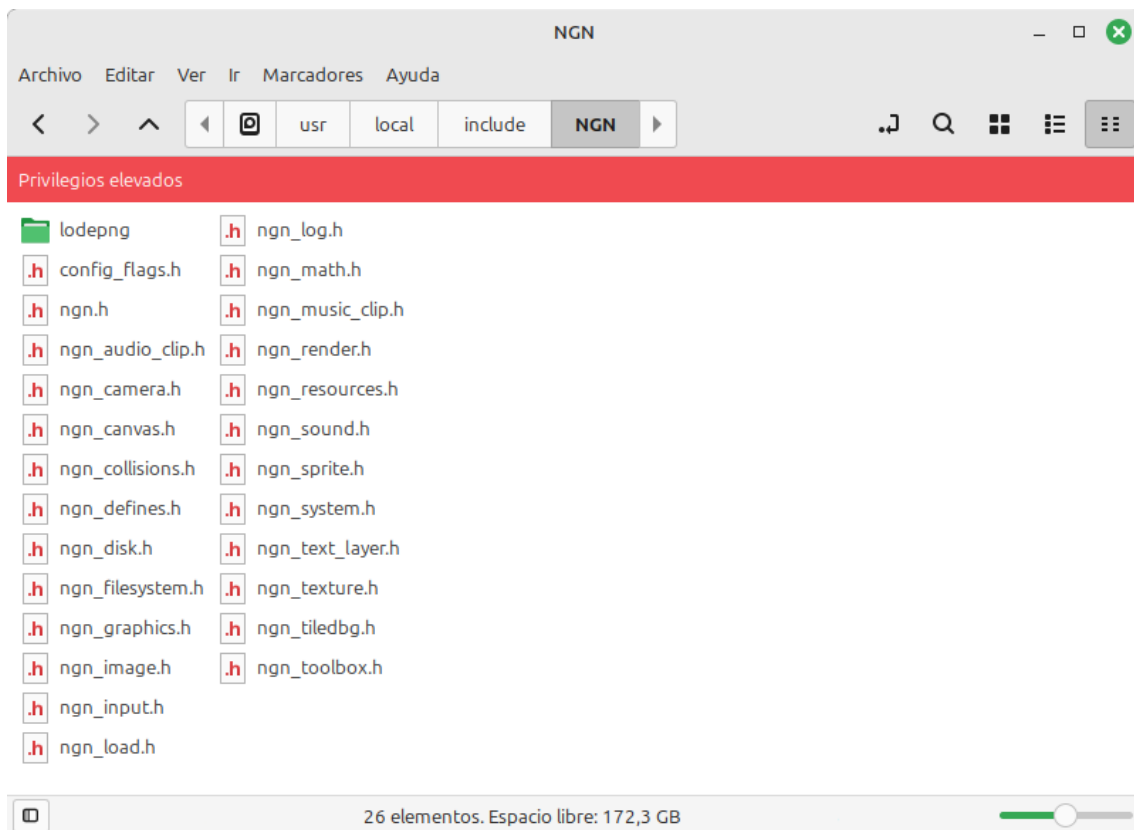
Copy the INCLUDE (.h) files contained in the library:



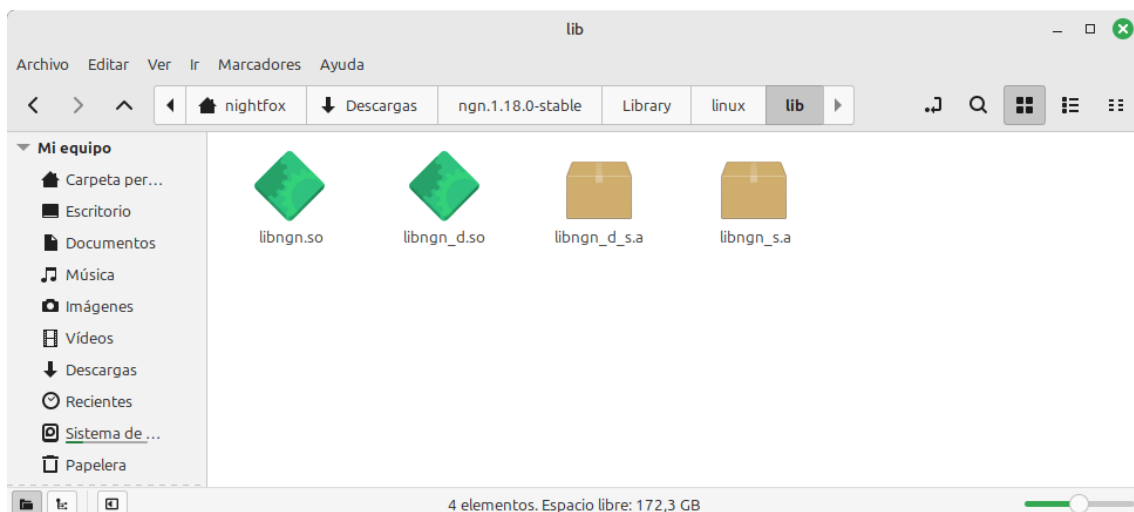
We will navigate to the system folder “/usr/local/include” with elevated privileges and create a folder with the name 'NGN' (in uppercase):



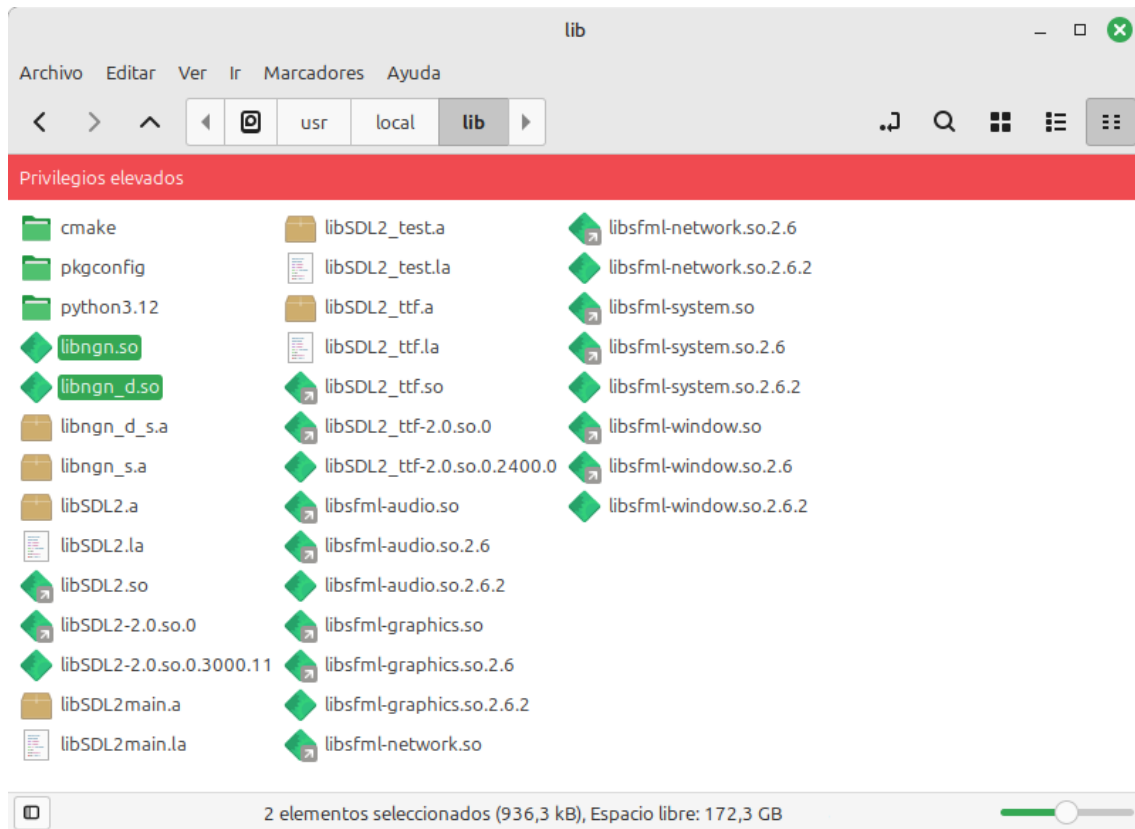
Next, we will paste the .h files inside it:



Now we will copy the binary files of the library (.a and .so) from the lib folder:



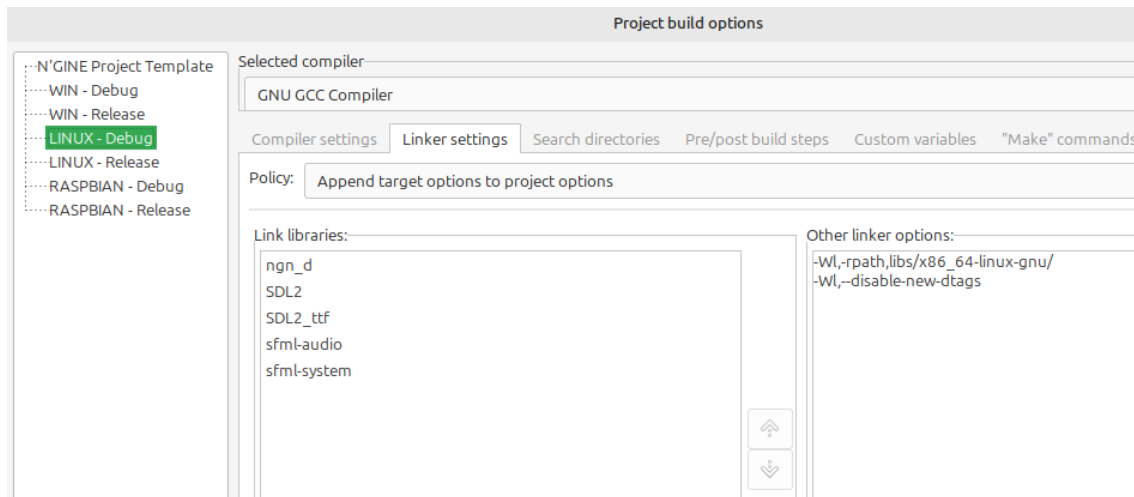
Now we will navigate to the system folder “/usr/local/lib” with elevated privileges and we will paste the .a and .so files inside this folder:



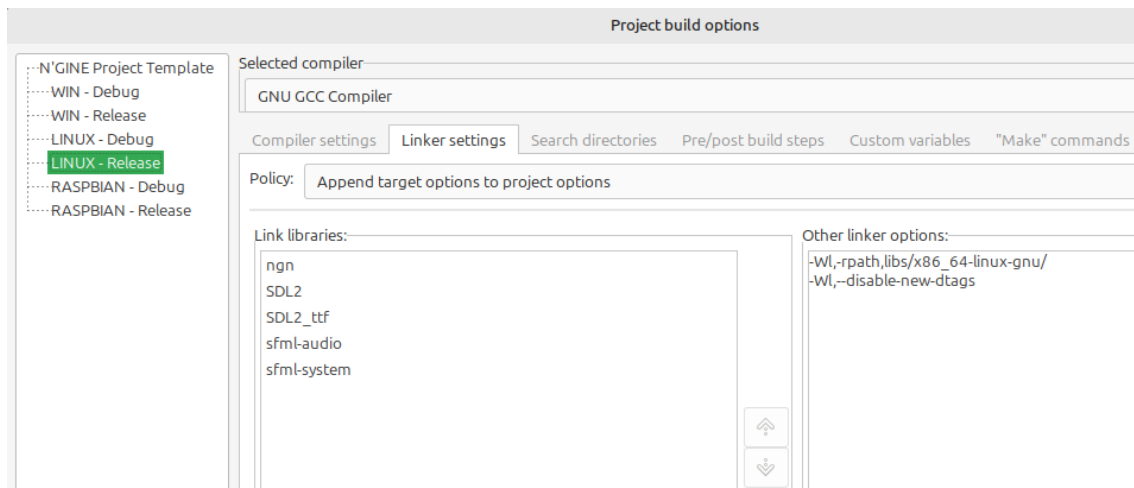
Project Configuration - CODE::BLOCKS

In this case, we will also assume that you have manually compiled the SDL2, SDL2_ttf, and SFML Libraries to keep them up to date with the latest versions, so they will be located in the /usr/local directory.

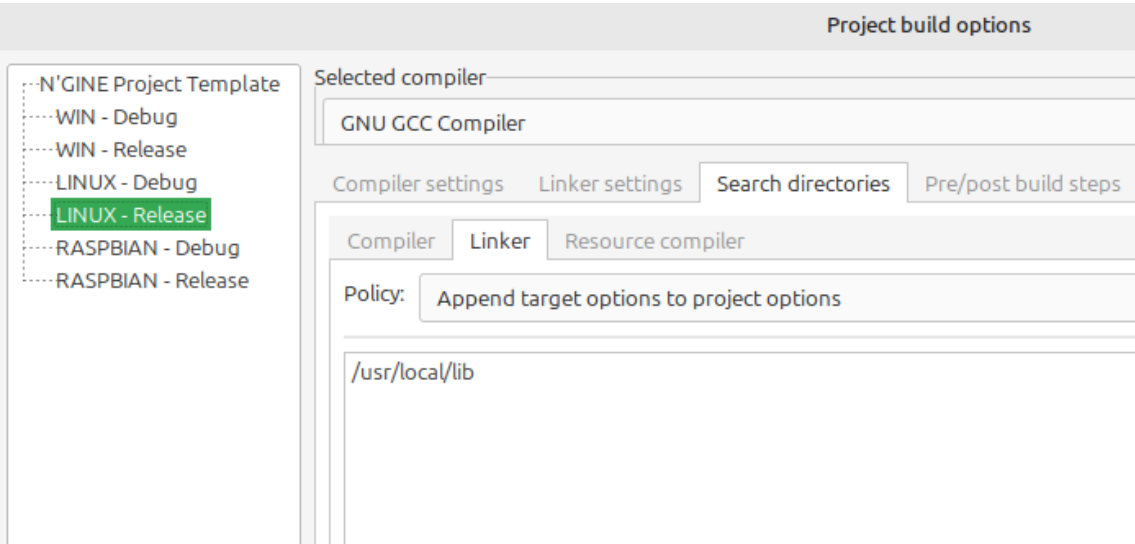
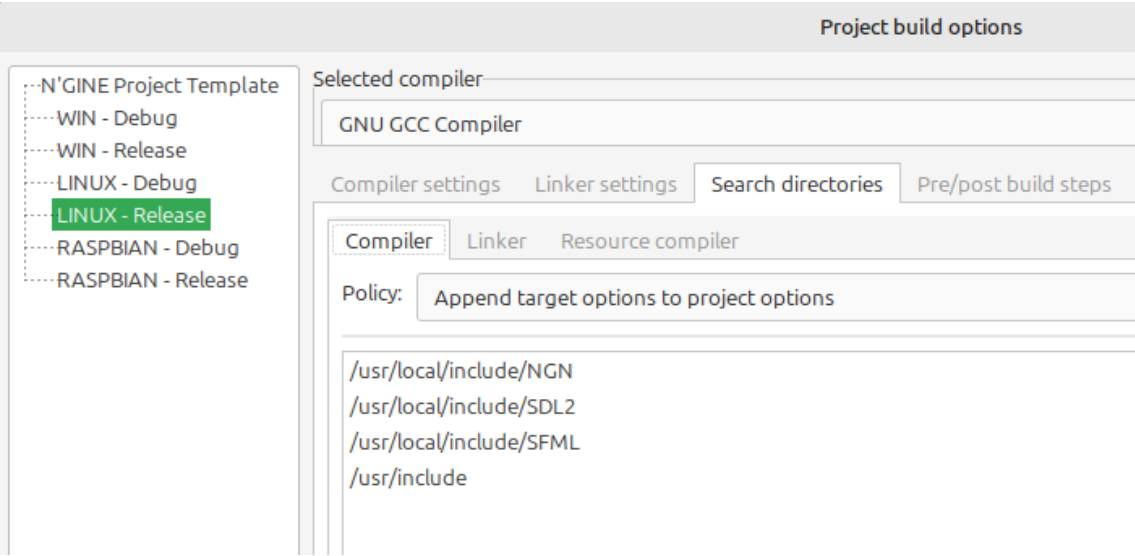
In “Project build options”, in the “LINUX-Debug” section, we will configure the following parameters:



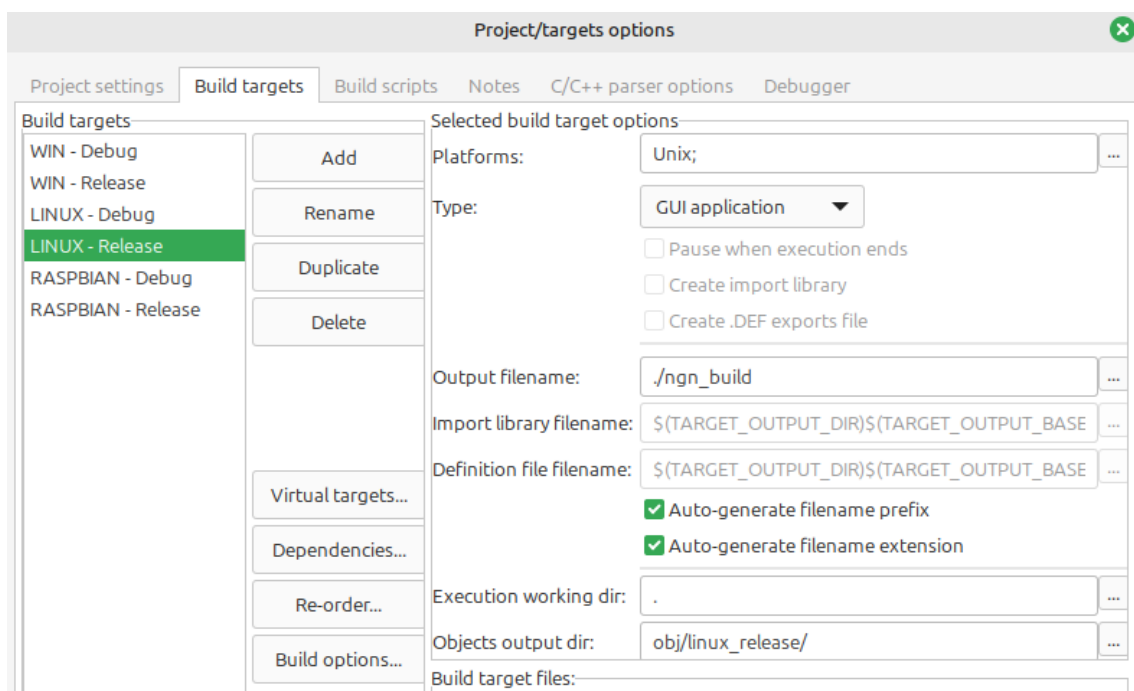
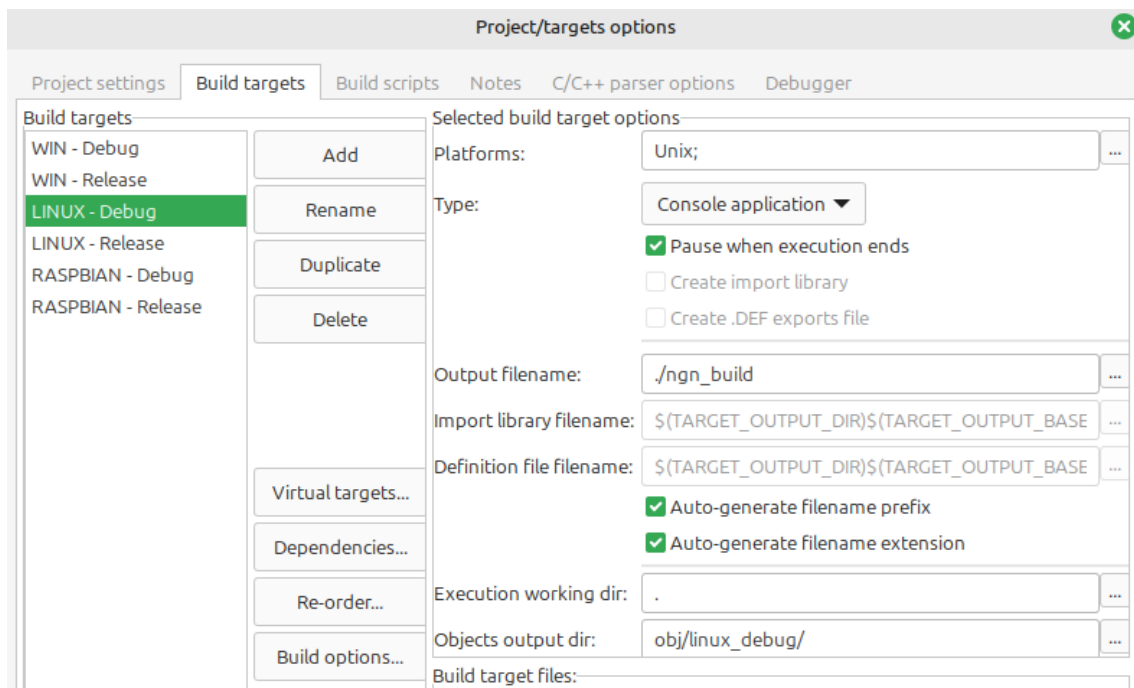
In “Project build options”, in the “LINUX-Release” section, we will configure the following parameters:



In “Project build options”, we will configure the following parameters in both the “LINUX-Debug” and “LINUX-Release” sections:



In 'Project/target options,' we will configure the following parameters:



Optionally, the library includes various templates with all these options pre-configured, both for Windows and Linux/Raspberry PI OS. If you use these templates when creating projects, you will only need to perform the library installation step.

For Linux Mint and Raspberry PI OS environments, there are also automated installation scripts with the latest versions of the N'gine library, as well as the used versions of SDL2, SDL2_ttf, and SFML2. These versions will be installed in the corresponding “/usr/local” folders and may replace existing versions of these libraries.

- Revision date: Monday, January 13, 2023.