1. Installer CMake https://cmake.org/download/

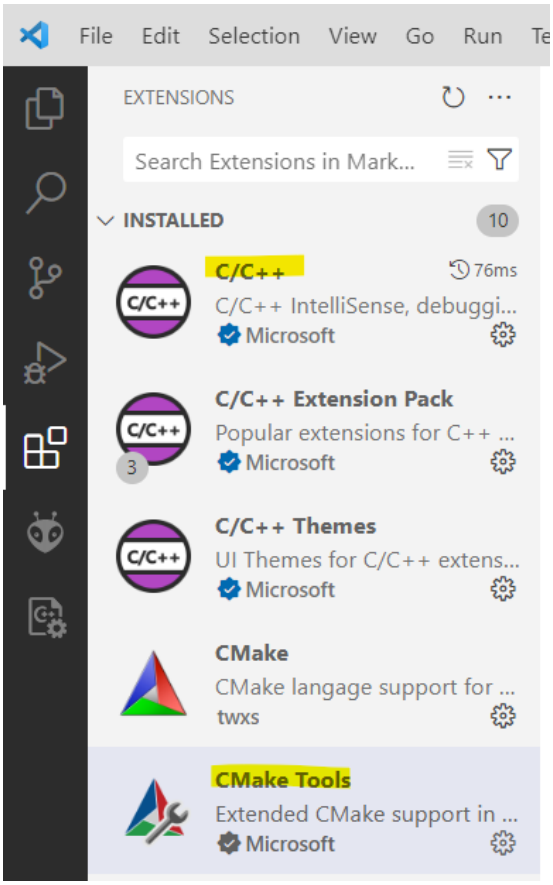
Source distributions:

Platform	Files
Unix/Linux Source (has \n line feeds)	cmake-3.29.0-rc1.tar.gz
Windows Source (has \r\n line feeds)	cmake-3.29.0-rc1.zip

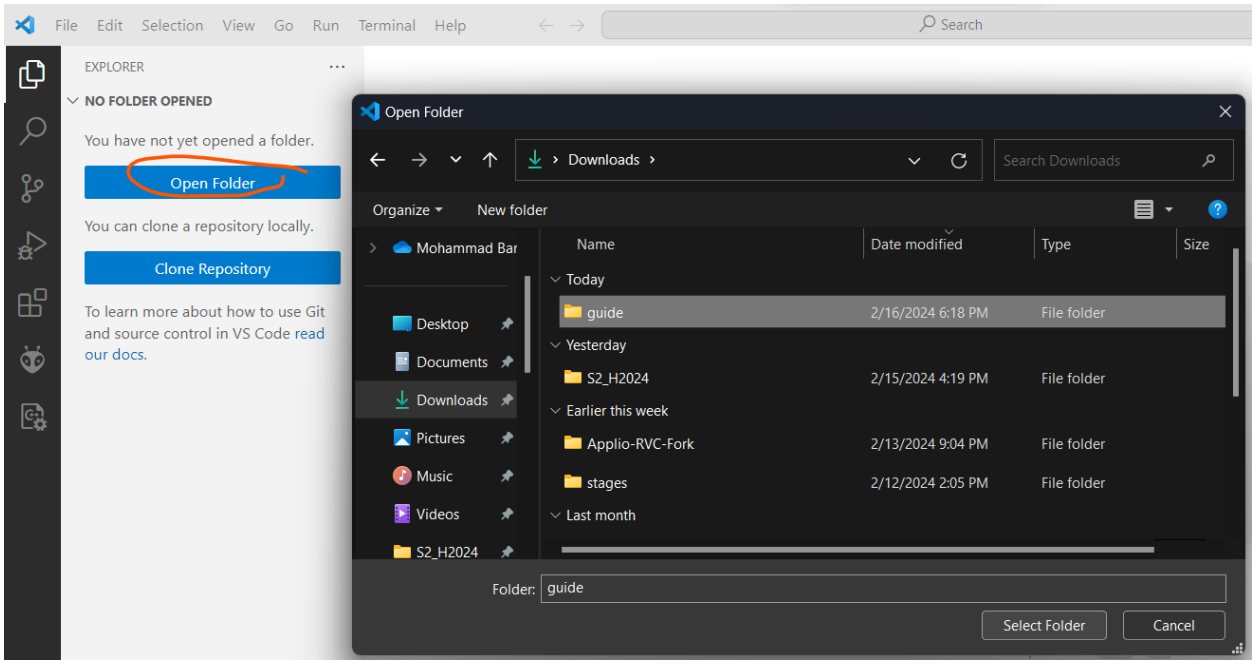
Binary distributions:

Platform	Files
Windows x64 Installer:	cmake-3.29.0-rc1-windows-x86_64.msi

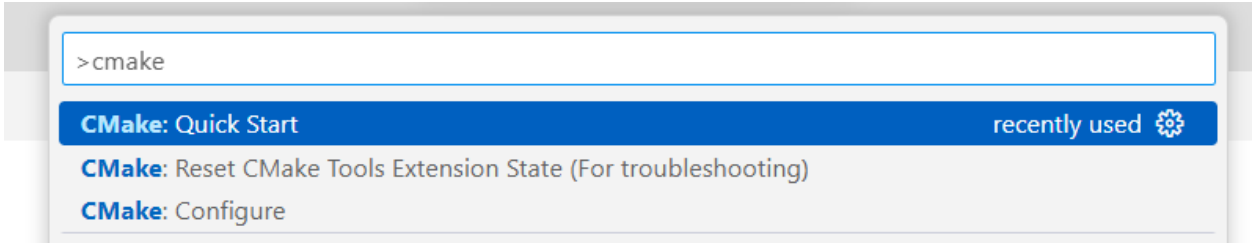
2. Installer les extensions C/C++ et CMake Tools dans VSCode.



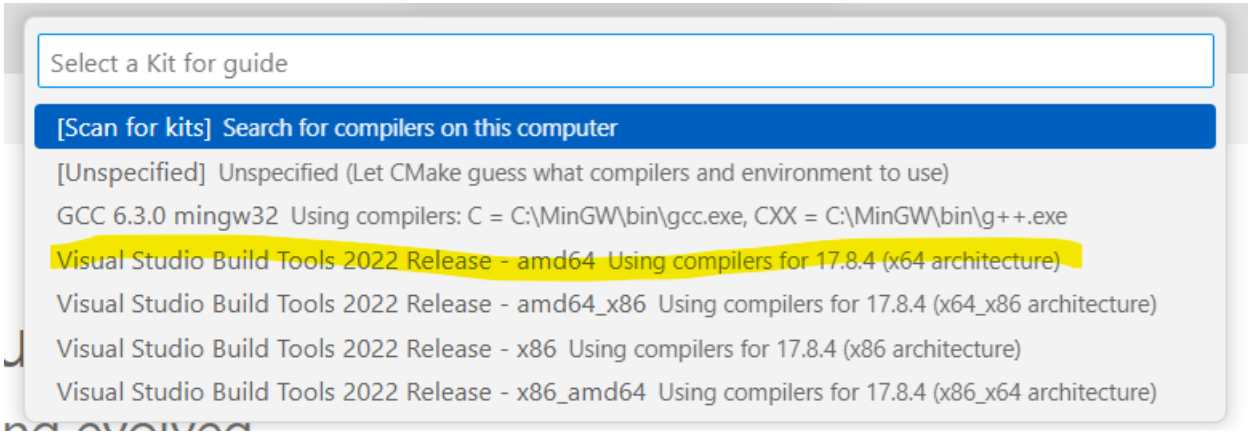
3. Créer un dossier et l'ouvrir dans VSCode.



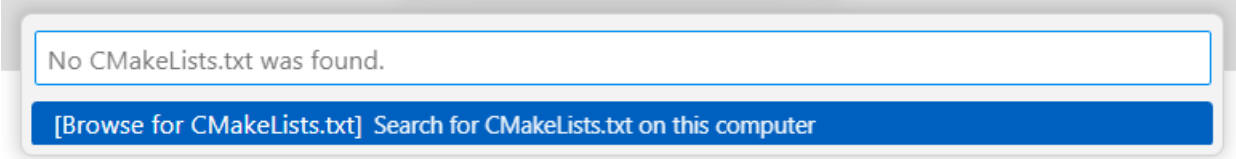
4. Dans la barre de recherche de VSCode (en haut), taper >cmake et choisir CMake: Quick Start



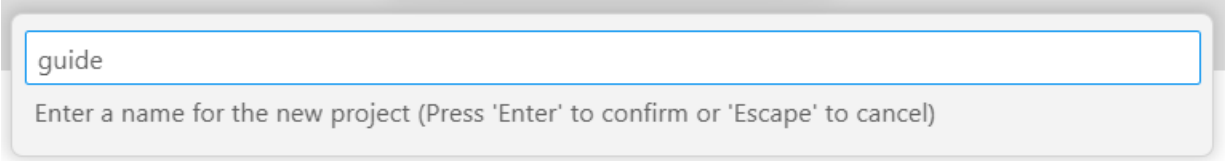
5. Choisir le compilateur (par exemple, celui pour l'architecture x64).



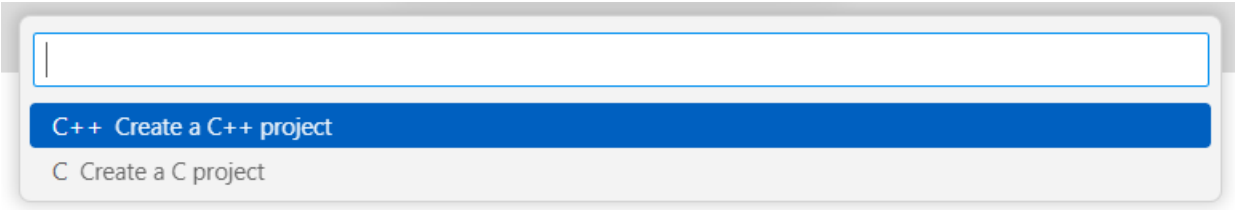
6. Le fichier CMakeLists.txt n'existe pas. Cliquer ailleurs et ça devrait passer à l'étape suivante.



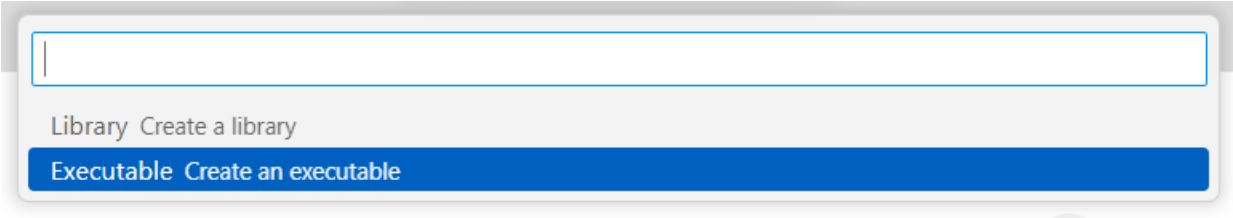
7. Choisir un nom pour le projet.



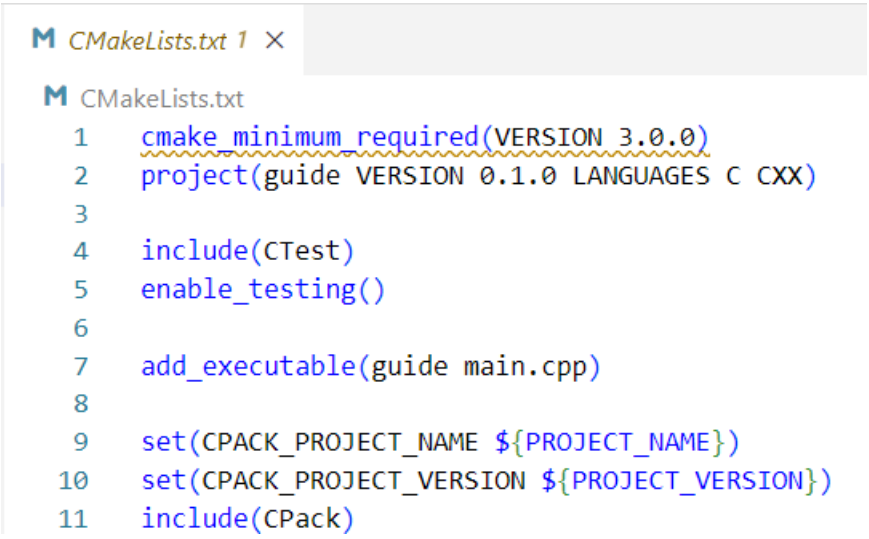
8. Choisir le langage.



9. Choisir Executable.



10. Le fichier CMakeLists.txt a été créé.

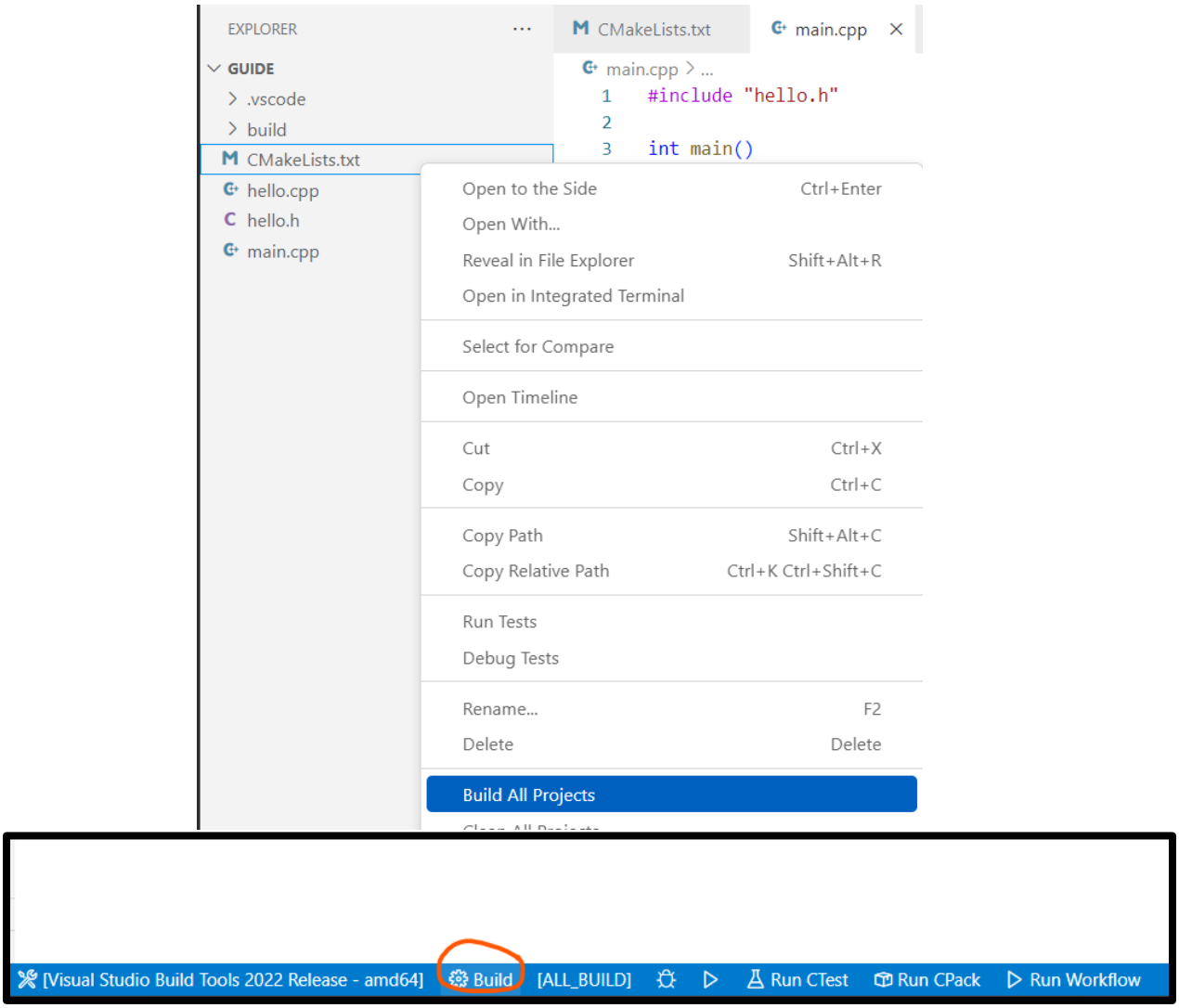


11. Vous pouvez alléger le contenu du fichier ainsi (avec vos noms de fichiers source, pas les .h).

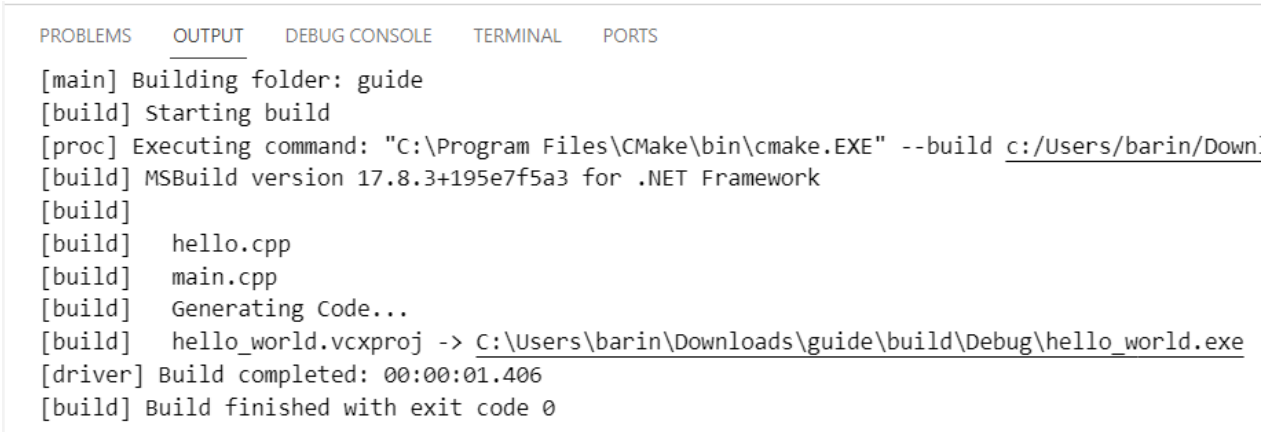
```
cmake_minimum_required(VERSION 3.24)
project(hello_world)

# Our Project
set(SOURCE_FILES
    main.cpp
    hello.cpp)

add_executable(${PROJECT_NAME} ${SOURCE_FILES})
```

12. Dans mon cas, j'ai un fichier main.cpp, un fichier hello.cpp et un fichier hello.h. J'appelle une fonction de hello.cpp dans le main.cpp.

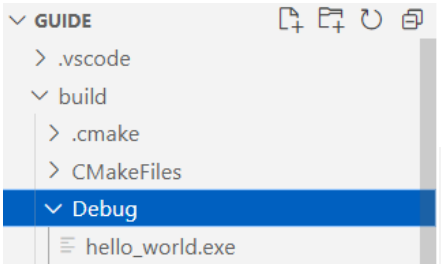
13. Pour compiler, vous pouvez soit faire un clic droit sur le fichier CMakeLists.txt et choisir Build All Projects ou cliquer sur Build en bas dans VSCode.



14. Si tout se passe bien, vous aurez un message de succès.



15. Pour exécuter le code, il suffit de trouver le fichier exécutable. Dans mon cas, il se trouvait dans le dossier Debug du dossier build.




```
PS C:\Users\barin\Downloads\guide> .\build\Debug\hello_world.exe
Hello, from main.cpp!
Hello, from hello.cpp!
PS C:\Users\barin\Downloads\guide>
```