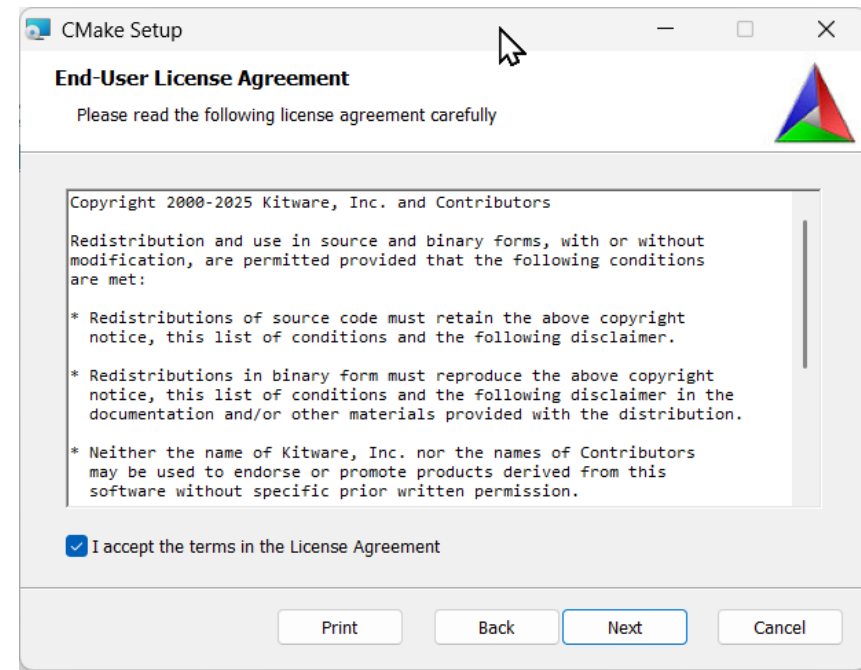
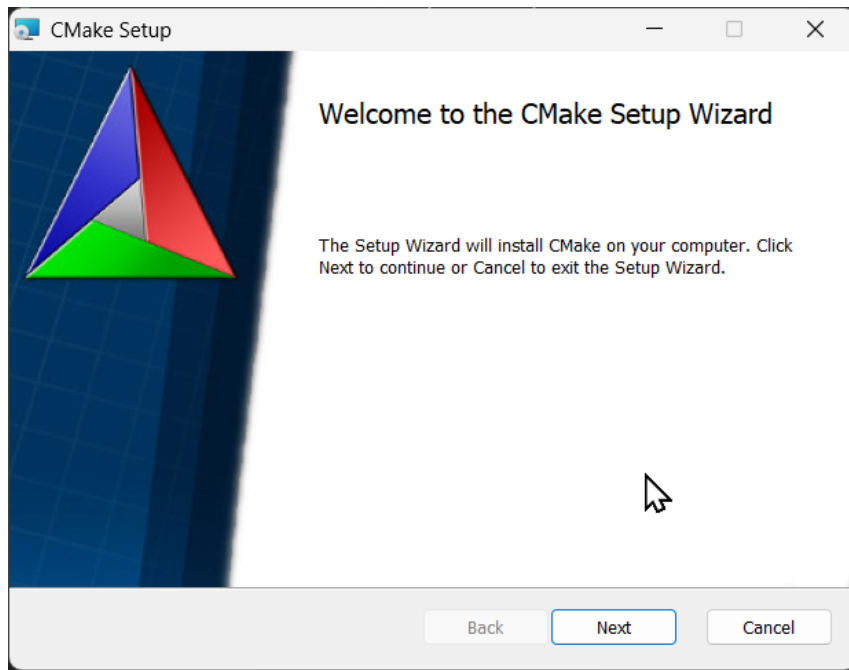
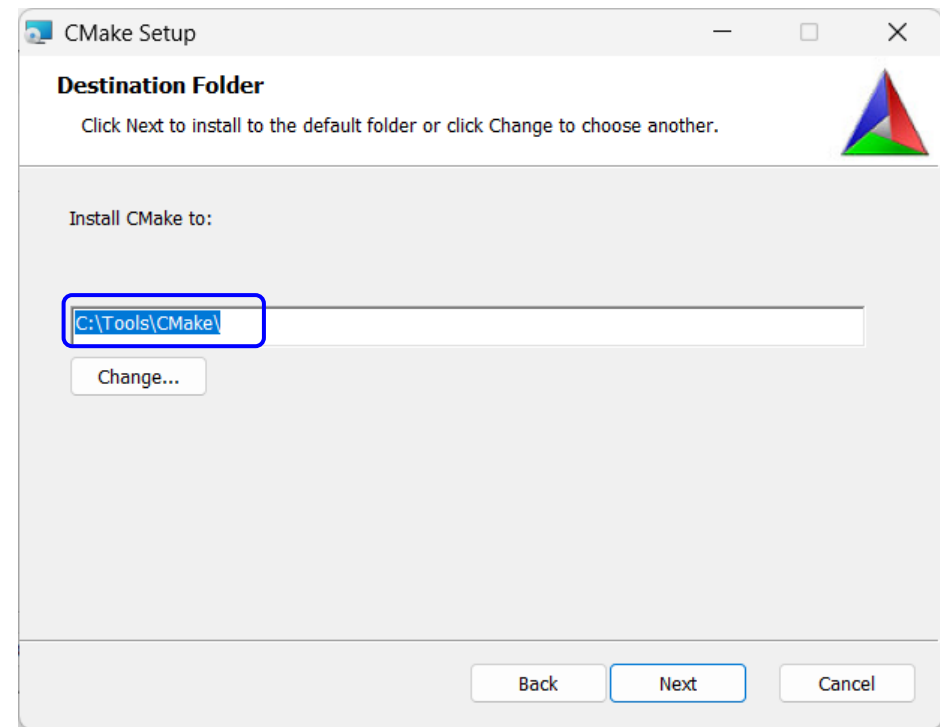
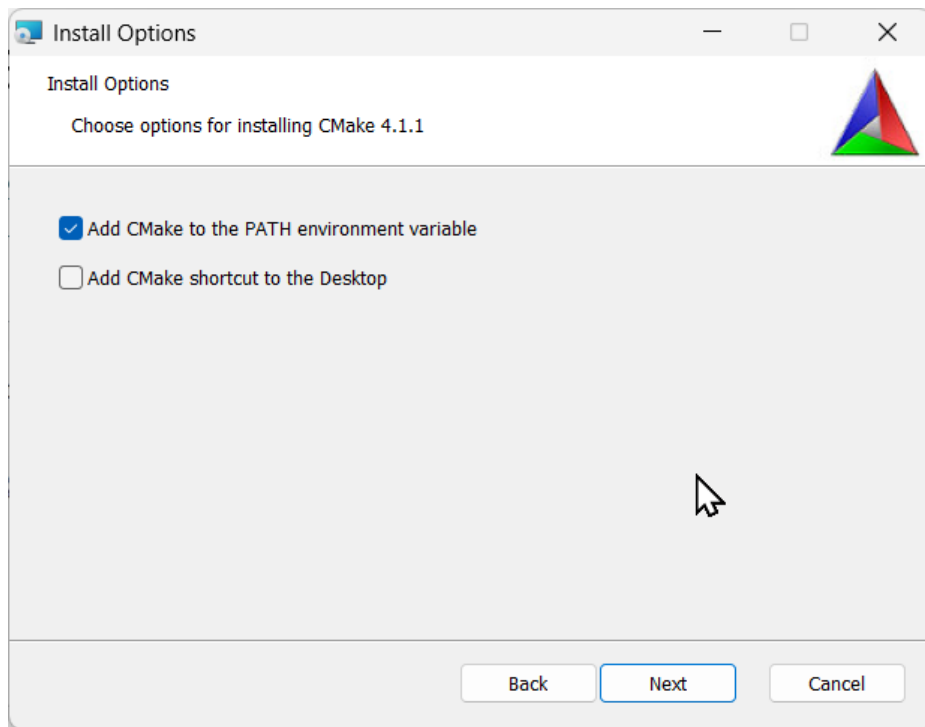


# Step1: CMake Installation (Windows)

[https://github.com/Kitware/CMake/releases/download/v4.1.2/cmake-4.1.2-windows-x86\\_64.msi](https://github.com/Kitware/CMake/releases/download/v4.1.2/cmake-4.1.2-windows-x86_64.msi)





コマンドラインで使うソフトウェアは、スペースを含まない英語のディレクトリにインストールするのがベストです。不必要なトラブルを避けられます。

```
C:\Users\sxwin>cmake --version
cmake version 4.1.2

CMake suite maintained and supported by Kitware (kitware.com/cmake).
```

「cmake --version」をコマンドラインで実行してインストールの成功を確認する。

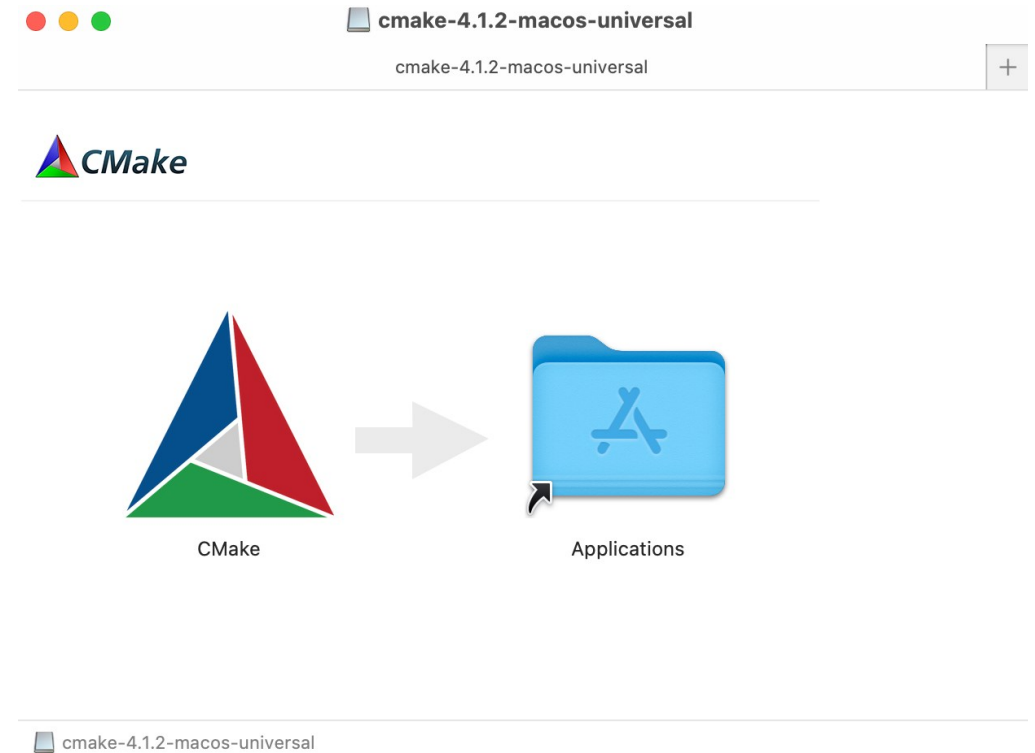
# Step1: CMake Installation (Mac)

macOS 10.13 or later

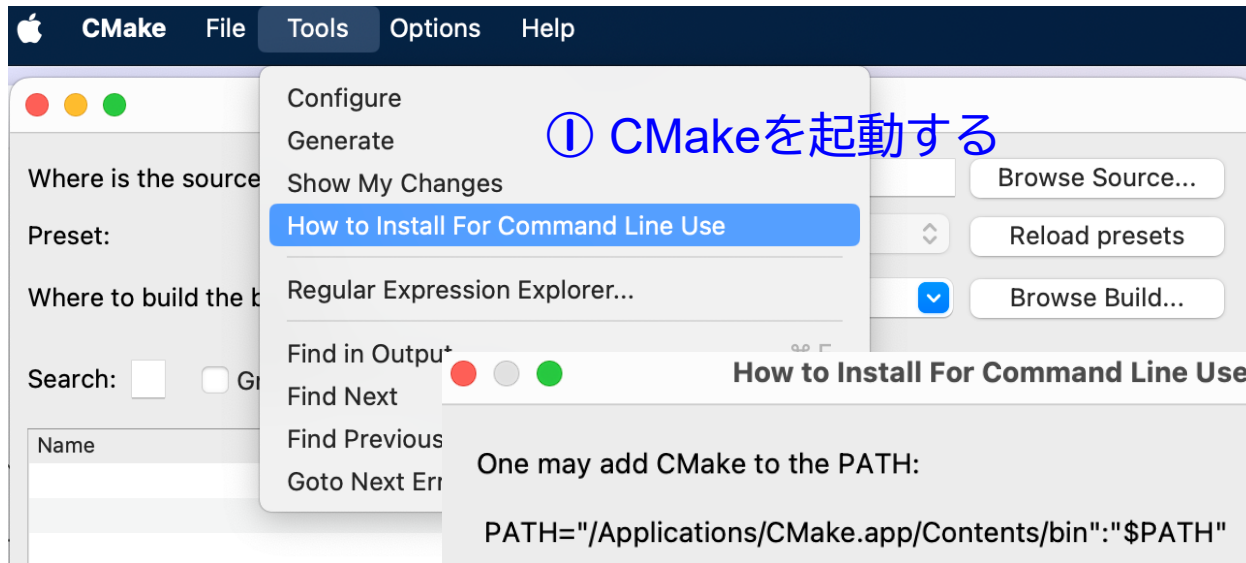
<https://github.com/Kitware/CMake/releases/download/v4.1.2/cmake-4.1.2-macos-universal.dmg>

macOS 10.10 or later

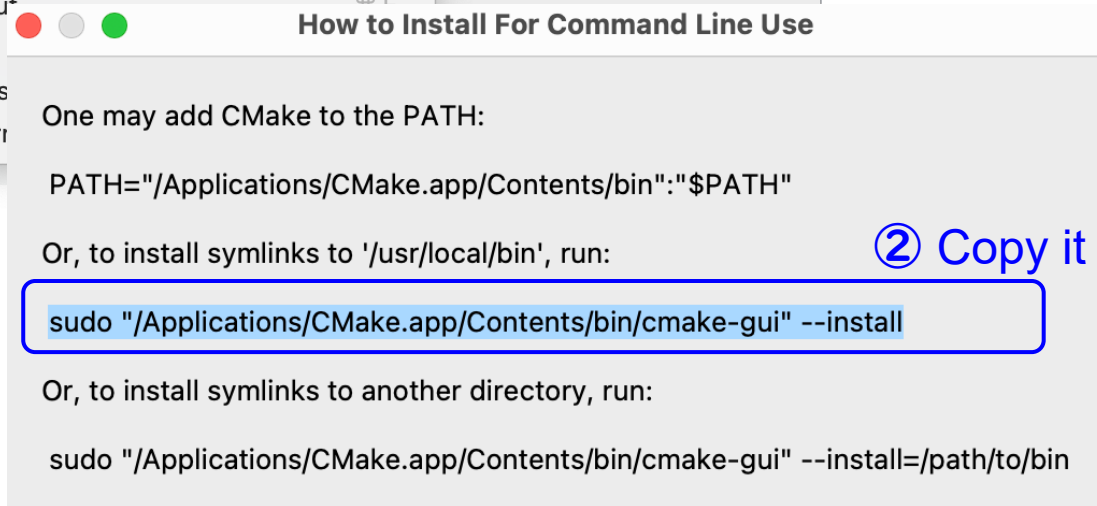
<https://github.com/Kitware/CMake/releases/download/v4.1.2/cmake-4.1.2-macos10.10-universal.dmg>



**Download & install**



① CMakeを起動する



② Copy it

③ ターミナルでこのコマンド  
を実行する

```
~ % sudo "/Applications/CMake.app/Contents/bin/cmake-gui" --install
Password:
Linked: '/usr/local/bin/cmake' -> '/Applications/CMake.app/Contents/bin/cmake'
Linked: '/usr/local/bin/ctest' -> '/Applications/CMake.app/Contents/bin/ctest'
Linked: '/usr/local/bin/cpack' -> '/Applications/CMake.app/Contents/bin/cpack'
Linked: '/usr/local/bin/cmake-gui' -> '/Applications/CMake.app/Contents/bin/cmak
```

```
~ % cmake --version
cmake version 4.1.2
```

④ 確認

「cmake --version」

CMake suite maintained and supported by Kitware (kitware.com/cmake).

# Step2: Setup CMake in VSCode

The screenshot shows the Visual Studio Code interface with the Extensions Marketplace open. The search bar at the top of the marketplace contains the text "cmake". The left sidebar shows the "Extensions" view with a blue badge indicating 3 extensions. The main panel displays the details for the "CMake Tools" extension by Microsoft. The extension is highlighted with a green box, and its "Install" button is also highlighted with a green box. The extension details show it has 52.7M downloads and a 4.5-star rating. The "Install" button is a blue button with a dropdown arrow. Below the "Install" button, there is a checkbox for "Auto Update" which is checked. A note states: "This extension is recommended based on the files you recently opened." The bottom right corner shows the "Marketplace" section with the identifier "ms-vscode.cmake-tools".

EXTENSIONS: MARKETPLACE

cmake

**CMake** 45.4M ★ 3  
CMake language support for Visual S...  
twxs [Install](#)

**CMake Tools** 52.7M ★ 4.5  
Extended CMake support in Visual S...  
Microsoft [Install](#) [▼](#)

**CMake Language Sup...** 1.6M ★ 4  
CMake language support for VS Co...  
Jose Torres [Install](#)

**cmake-format** 471K ★ 4  
Format listfiles so they don't look li...  
cheshirekow [Install](#)

**CMake Tools**  
Microsoft [microsoft.com](#) 52,7  
Extended CMake support in Visual St...  
[Install](#) [▼](#) ☒ Auto Update [⚙️](#)  
★ This extension is recommended based on the files you recently opened.

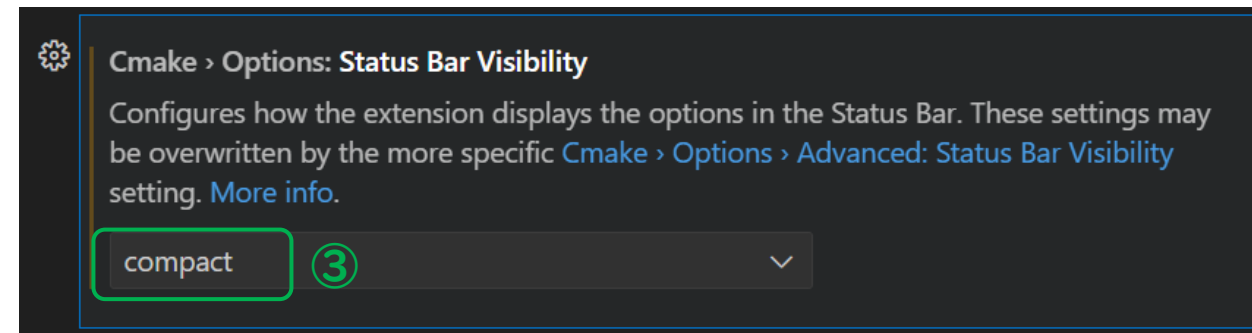
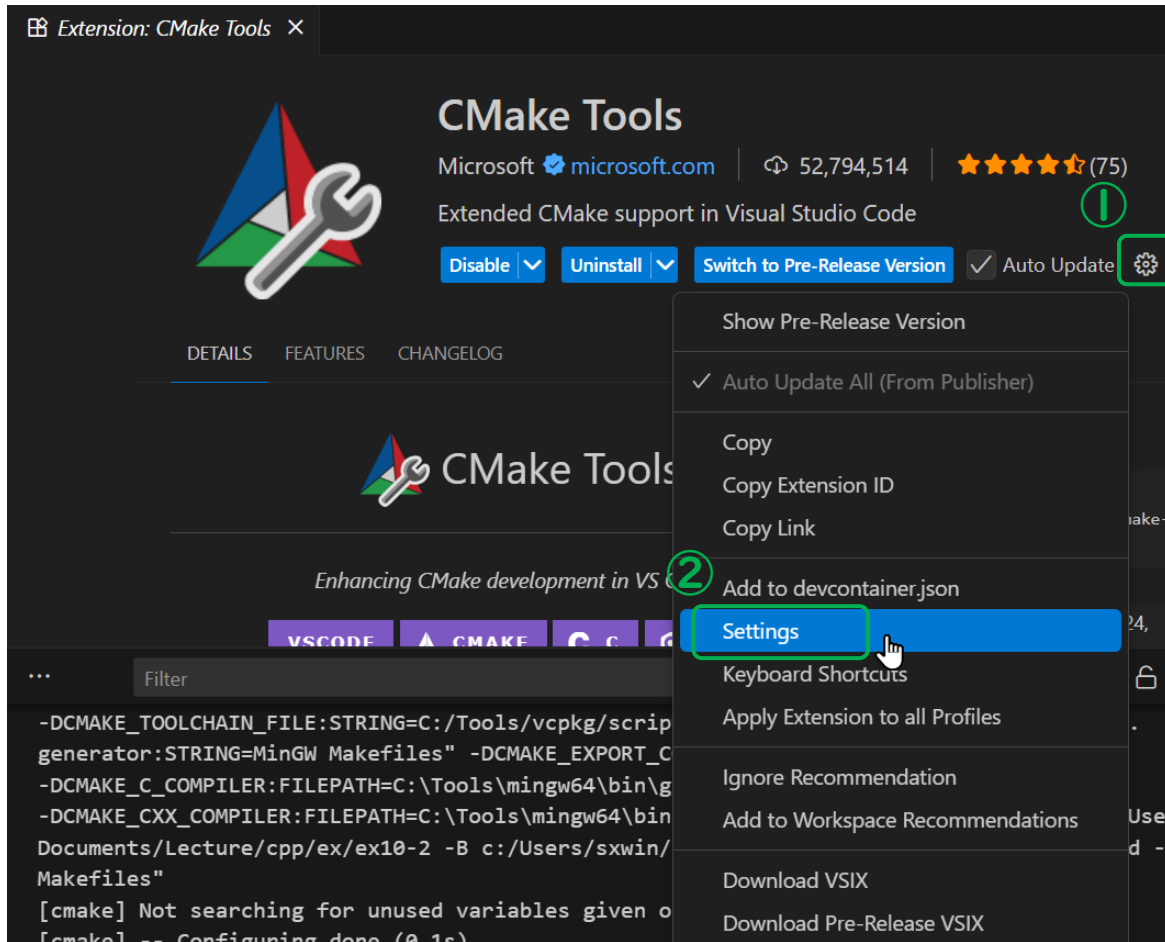
DETAILS FEATURES CHANGELOG

**CMake Tools**

Marketplace

Identifier ms-vscode.cmake-tools

# Step2: Setup CMake in VSCode



# Step3: Setup Project

1. 作業用のフォルダを新しく作成する。
2. リンク先にある4つのファイルを、そのフォルダの中にコピーする。

<https://drive.google.com/drive/folders/1G3ieFaqYydFYpbQd3qYYZNWqwU5rntYw?usp=sharing>

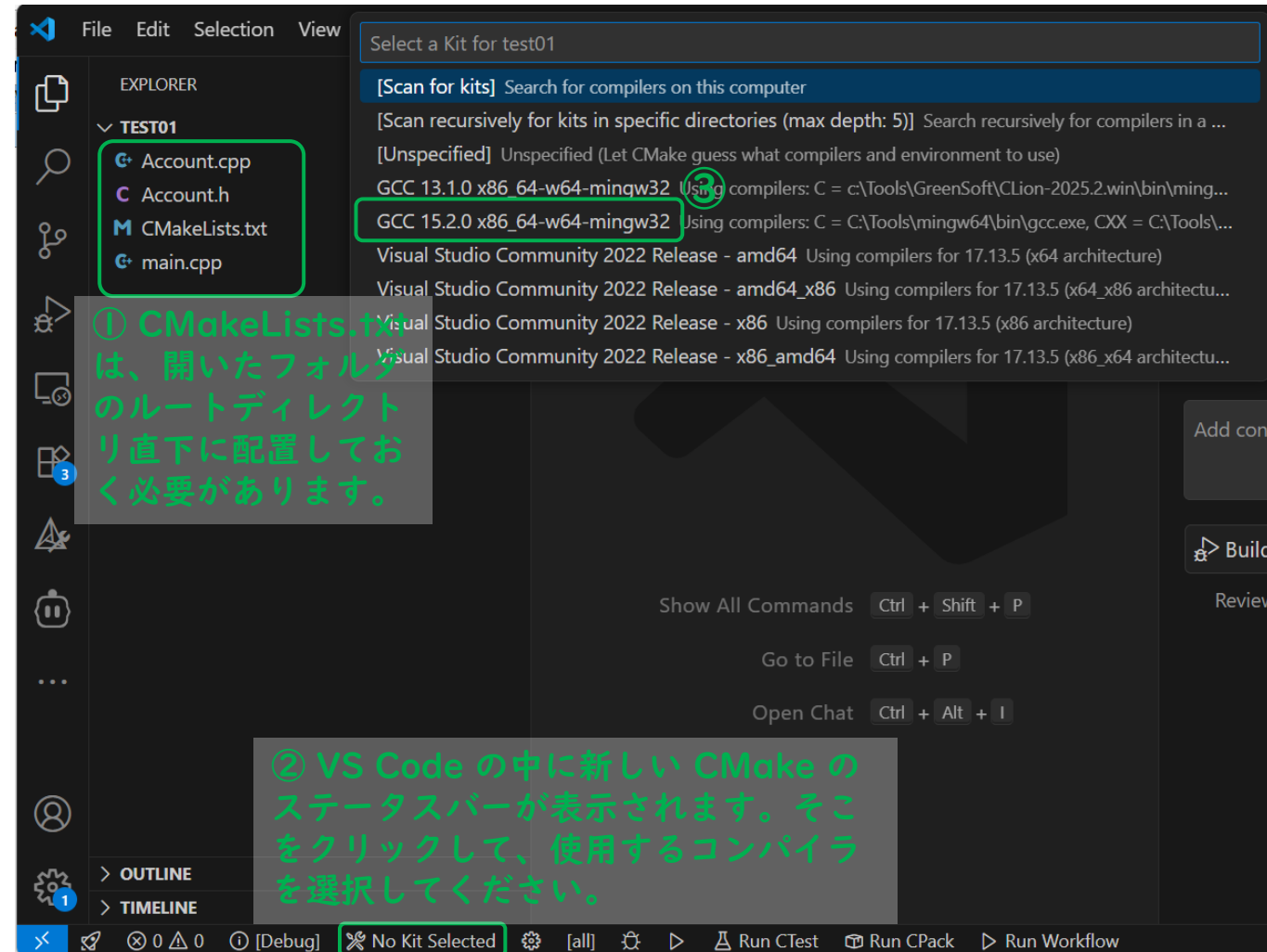
Account.cpp

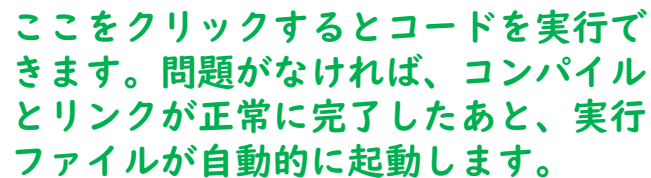
Account.h

CMakeLists.txt

main.cpp

3. VS Code で「File ⇒ Open Folder」を選び、作成したフォルダを開く。







# CMakeLists.txt とは？

- **CMake** が利用する設定用スクリプトファイルで、ファイル名は必ず **CMakeLists.txt** にします。
- このファイルの中で、
  - プロジェクト名は何か
  - どの言語を使うか (**C++** など)
  - どのソースファイルをビルド対象にするか
  - 実行ファイルやライブラリなど、どんな成果物を作るかといった情報を指定します。
- **VS Code** や **CLion**、あるいはコマンドラインから **cmake** コマンドを実行したときは、この **CMakeLists.txt** を読み取って、実際にコンパイル・リンクを行うためのビルド用ファイルを自動生成しています。
- 実際の大規模**C++**プロジェクトでは、**CMakeLists.txt** には外部ライブラリやヘッダのパス、プリプロセッサマクロ、コンパイル／最適化オプションなど、多くのビルド設定が追加されます。こうした設定によって、**CMake** はプラットフォームごとに適切なコンパイルコマンドを自動生成し、複雑な依存関係を一元的に管理できるようになります。

# CMakeLists.txt の基本

- シンプルな C++ プロジェクトであれば、右側の CMakeLists.txt テンプレートをそのまま利用できます。
  - (1) の欄にプロジェクト名（英数字のみ・スペース不可）を入力
  - (2) の欄に生成する実行ファイル名を入力
  - (3) の欄にコンパイルしてリンクしたい複数のソースファイル名を入力
- CMake のより複雑な使い方については、こちらを参照してください。
  - [C++初心者のためのCMake基本構文と使い方を徹底解説](#)
  - [【超入門】1日で理解するCmake](#)
- もちろん、AIに自分の要件をしっかりと伝えてしまえば、CMakeLists.txtを直接生成してもらうのが、実は効率的な方法です。

```
# 1. このプロジェクトに必要な最低限の CMake
バージョンを指定します
cmake_minimum_required(VERSION 3.10)

# 2. プロジェクト名を定義します。
# 'CXX'はC++言語を使用することを示します。
project(myapp CXX)

# 3. C++ 標準バージョンを指定（例: C++17）
set(CMAKE_CXX_STANDARD 17)
set(CMAKE_CXX_STANDARD_REQUIRED True)

# 4. 実行可能ターゲットの定義
# add_executable(目標名 ソースファイルのリスト)
# CMake は依存関係とリンクを自動的に処理します
add_executable(
    main
    main.cpp
    Account.cpp
)
```



```
g++ -std=c++17 main.cpp Account.cpp -o main
```