IN4089 - Data Visualization - Project 2: Volume Visualization

Install Guide Linux (Visual Studio Code)

Note: This install guide has been tested with Ubuntu 20.04 and Visual Studio Code. Variations might work, but you will be responsible for adjusting the setup yourself.

⚠️ **Please use pathes/folders for the framework and vcpkg without spaces or special characters.** ⚠️

Third party software

- Download Microsoft Visual Studio Code from the Ubuntu Software store.

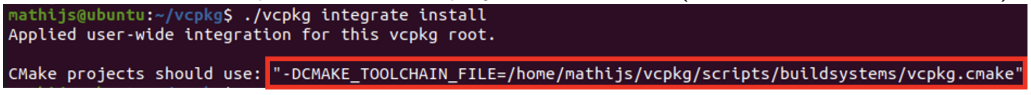
We will need some additional software to setup. The text in `monospaced font` correspond to the terminal commands for the respective tools, assuming an apt-based system such as Ubuntu.

- Install the essential build tools and a modern C++ compiler (g++9 or newer)
  `sudo apt install build-essential g++-9`
- Install CMake 3.14.0 or newer
  `sudo apt install cmake`
- Install git, curl, unzip and tar (required to build vcpkg)
  `sudo apt install git curl unzip tar`
- Install the OpenGL development libraries
  `sudo apt install libxinerama-dev libxcursor-dev xorg-dev libglu1-mesa-dev`

vcpkg

Next, we will install vcpkg, a package manager that provides 3rd party libraries needed for the provided framework. Open a terminal window. Navigate to the folder where you want to store vcpkg (avoid spaces and special characters in the path to this folder) Note: this path has no relation to the path where you put the volume visualization framework. Then run the following commands:

- Clone vcpkg from github
  `git clone https://github.com/microsoft/vcpkg.git`
- Change into the checked out directory
  `cd vcpkg`
- Inside the directory, compile vcpkg
  `./bootstrap-vcpkg.sh -disableMetrics`
- Now we enable vcpkg integration in Visual Studio Code. Run
  `./vcpkg integrate install`
  and note down the output after CMake projects should use: (similar to the red box below)



Building/Running the Framework

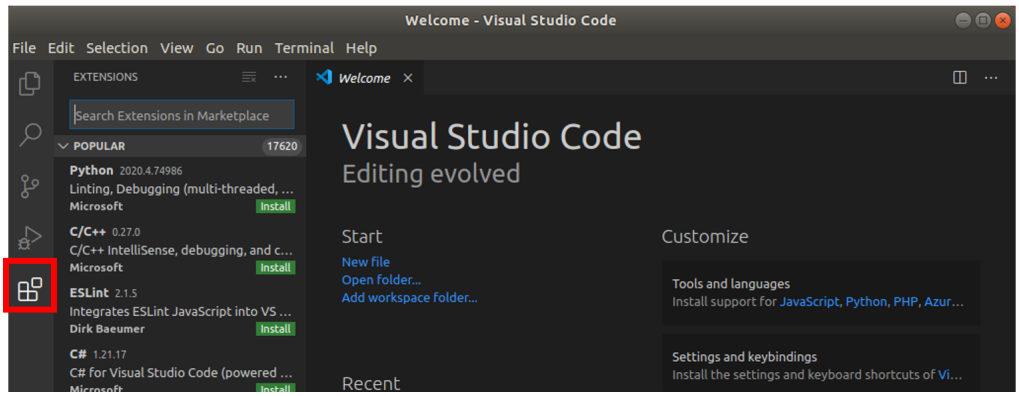


Figure 1: Open Visual Studio Code and click the framed icon to install the "C/C++" and "CMake Tools" extensions by Microsoft

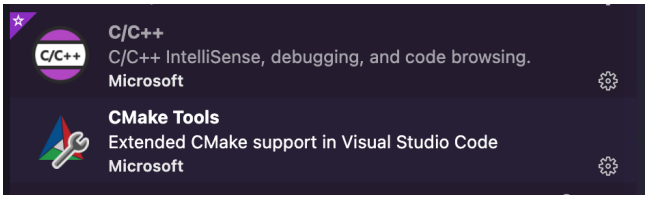


Figure 2: The "C/C++" and "CMake Tools" extensions by Microsoft

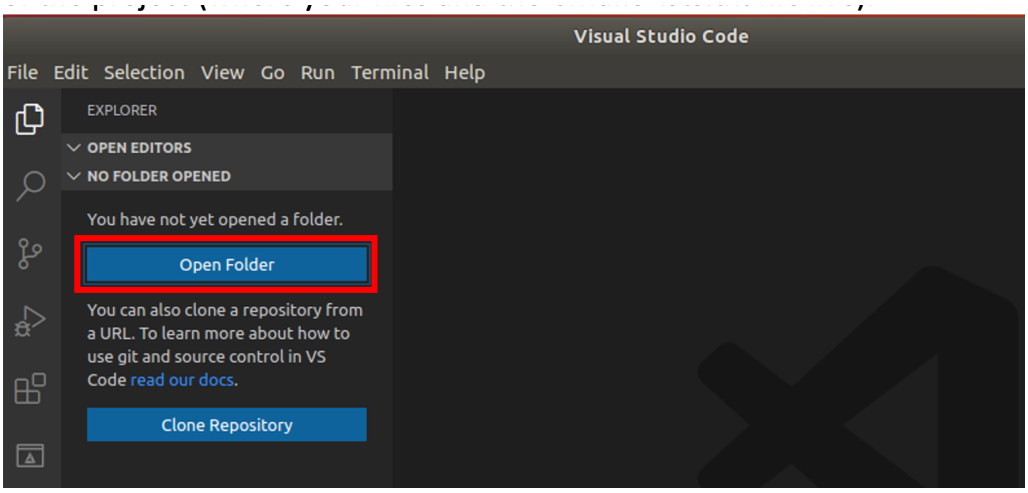


Figure 3: In the main view click on *Open Folder* (or *File* ⇒ *Open Folder*) and select the root of the project framework (where your files and the *CMakeLists.txt* file live). (See below for what the content of the folder should look like.)

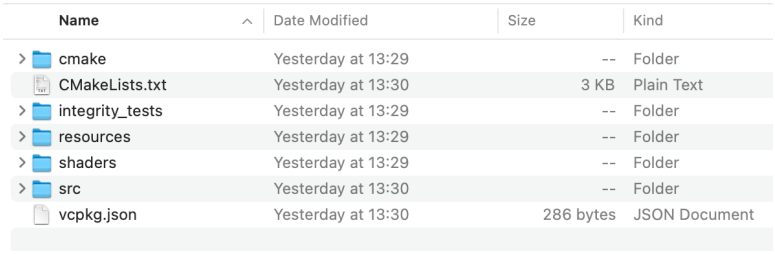


Figure 4: The content of the folder you should open in Visual Studio Code.

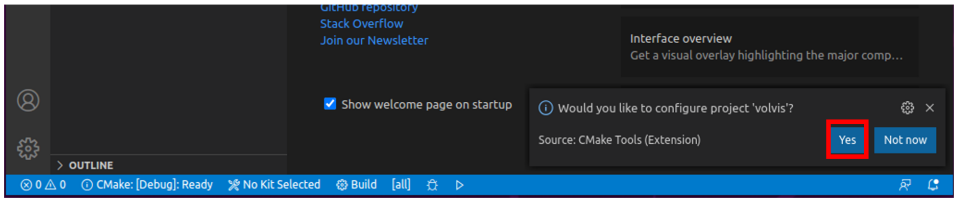


Figure 5: After opening the folder, a popup should show up asking to configure the project. Click *Yes*.

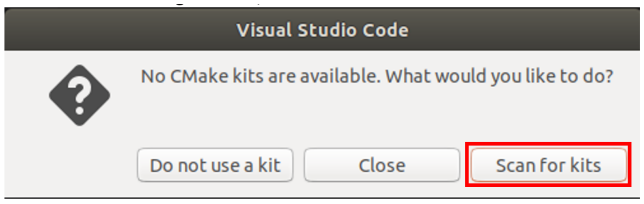


Figure 6: The first-time running VS Code, it will ask about CMake kits. Click *Scan for kits*.

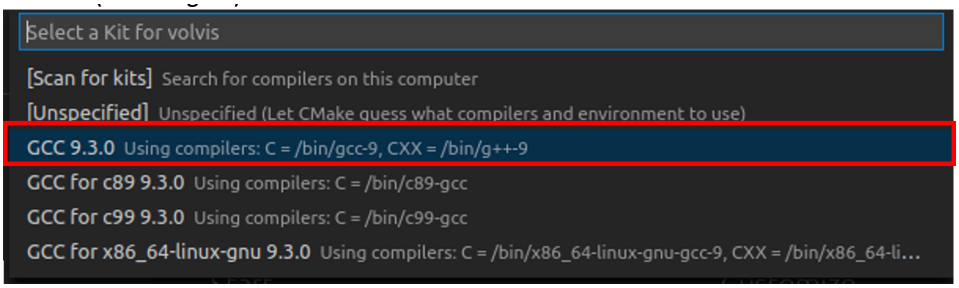


Figure 7: After scanning you will be asked to select a kit. Select the latest version of GCC that is installed (9.0 or higher).

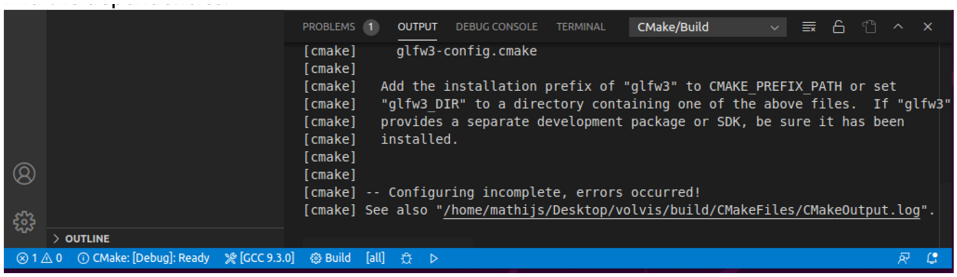


Figure 8: CMake will now start configuring your project. Initially, this will fail because it cannot find the dependencies.

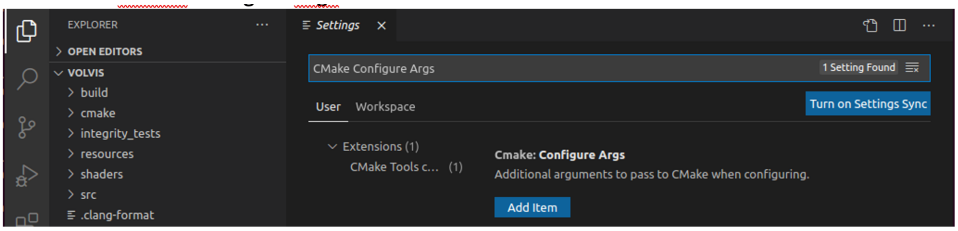


Figure 9: Open the settings menu by going to Code ⇒ Preferences ⇒ Settings, and search for *CMake Configure Args*.

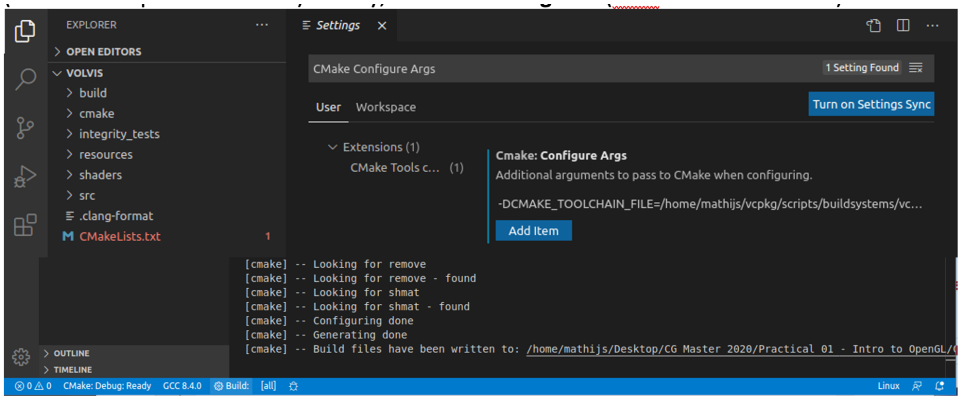


Figure 10: Click the *Add Item* and paste the text (without the quotation marks) that you after running `vcpkg integrate install` during the installation of vcpkg. **Then save the settings file** (File ⇒ Safe).

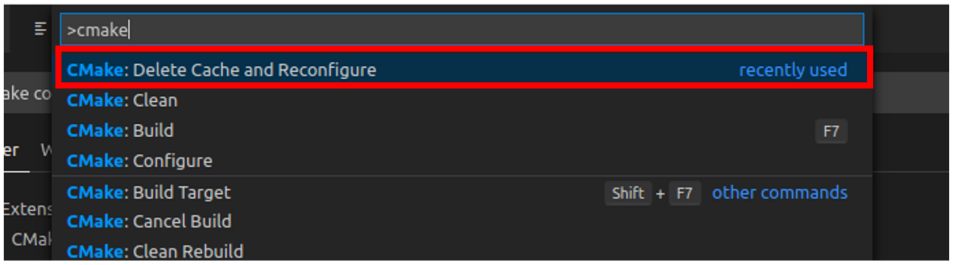


Figure 11: Now we will reconfigure the project with the changes we just made. Open the Command Palette (`Ctrl + shift + p`) and search for *CMake: Delete Cache and Reconfigure* (and press enter).

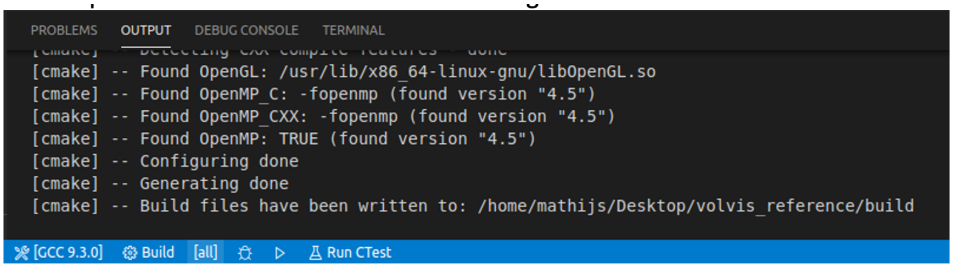


Figure 12: The output window should now look something like this.

Figure 13: The project has two targets: *IntegrityTests* and *Viewer* (the application). To build and run Viewer, we first need to select it. In the blue bottom bar, click on *[all]* and then select *Viewer* in the menu that pops up at the top.

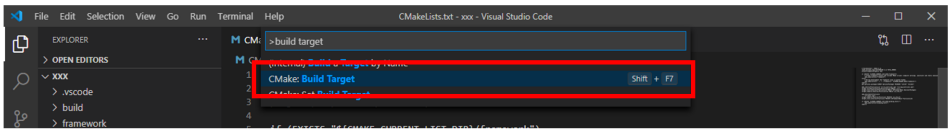


Figure 14: To compile the project, open the Command Palette (`Ctrl + shift + p`) and search for *CMake: Build Target* or use the shortcut `shift + F7`.