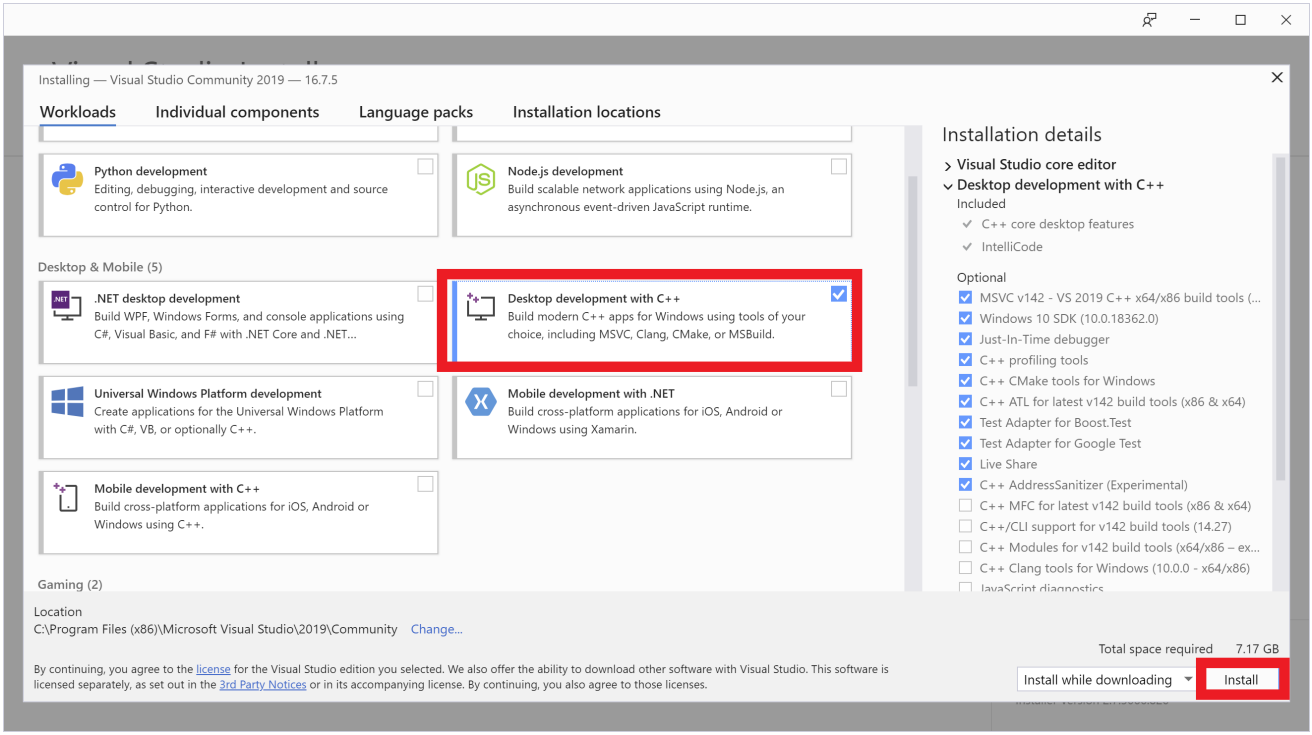
Windows setup for *Visual Studio*

This guide is for students who want the full C++ development experience on the MS Windows platform.

Prerequisites:

- Windows 10
- Visual Studio Community 2019 or newer - Download
- CMake 3.12 or newer - Download
- Git - Download

Follow the usual setup wizards for each. **For Visual Studio, make sure you tick "Desktop Development with C++" in the workloads page, leave everything else default.** You MUST reboot your machine after installation.



Once all the required software are installed. Proceed to setup vcpkg, the C++ package manager.

vcpkg setup

vcpkg is a package manager/repository similar to Maven in the Java world. You may want to read official the documentation for more in-depth explanation. Essentially, CMake provides the cross-platform build definition and vcpkg provides pre-built libraries for CMake to consume.

Make sure you run the following commands from a native *Windows* environment, for example: `cmd` or PowerShell (i.e. NOT WSL/MSYS2/Cygwin). Install vcpkg:

```
# in a suitable directory
git clone https://github.com/microsoft/vcpkg
```

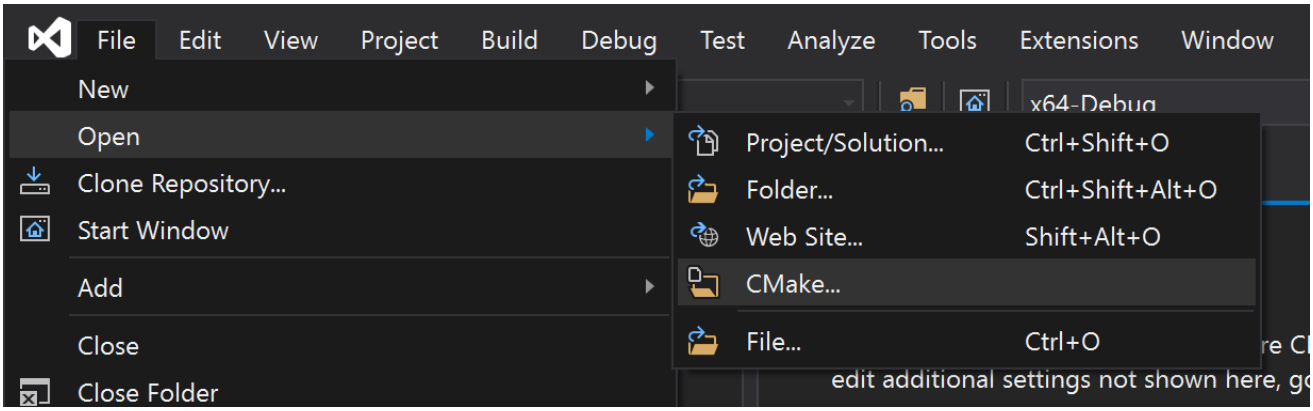
```
.\vcpkg\bootstrap-vcpkg.bat
.\vcpkg\vcpkg integrate install
```

Then, install SDL2:

```
.\vcpkg\vcpkg install sdl2:x64-windows
```

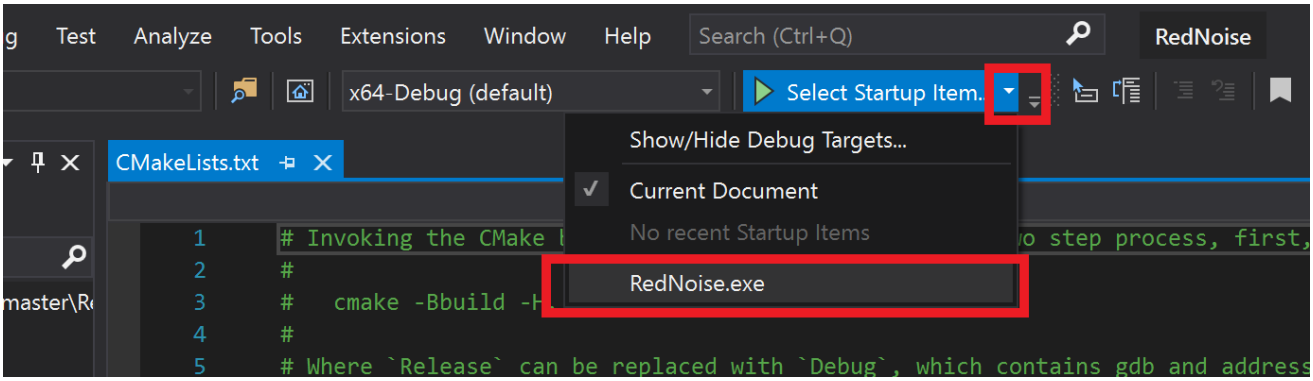
Once this is done, open Visual Studio. If you are presented with a welcome dialog, click "Continue without code →".

In the main editor window, select `File | Open | CMake...`, choose the `CMakeList.txt` file from the template project.



Visual Studio should now configure the project automatically, this may take a few minutes.

Once the project is configured, on the top main toolbar, open the "Select Startup Item" drop-down and choose the executable. You can now press the play button to compile and debug/run the program.



For development, simply edit code as usual and press play to see results.

You can search and install other packages using the vcpkg install command mentioned above. Note that vcpkg defaults to the x86 architecture, make sure you append the `:x64-windows` triplet to your package name if you are on x64.