



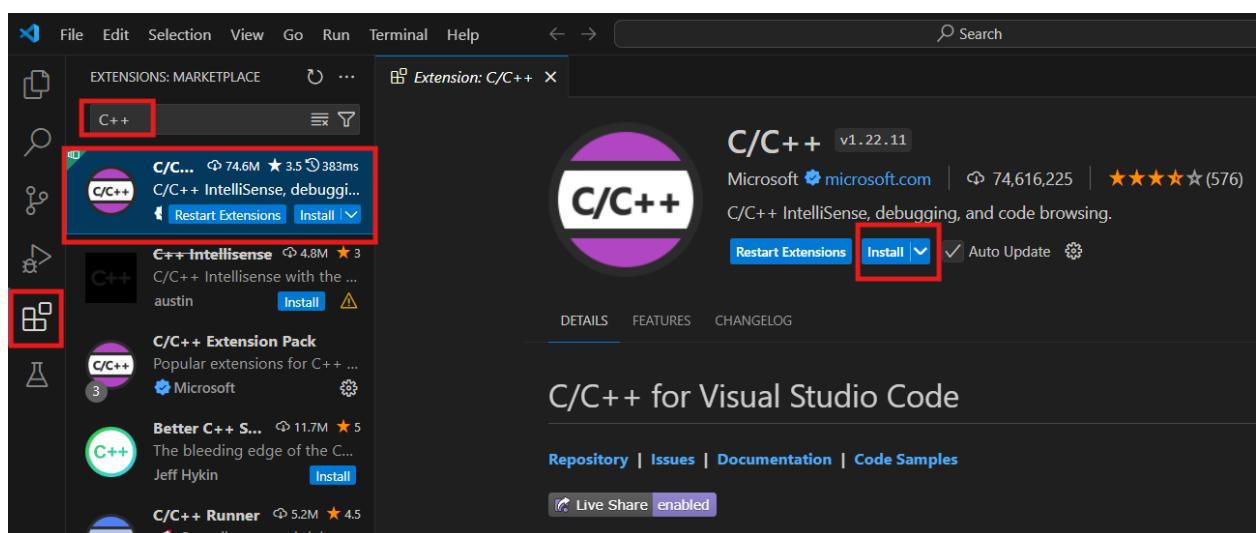
C++ Installation Guide for VS Code on Windows PCs.

There are many ways to configure VS Code to run C++. This tutorial guides you to the installation method that uses “Visual Studio Build Tools”. Comparison between other installation methods, is available in the bottom of the document under [“Appendix 01”](#).

This document contains a short version of installation guide. The detailed official Microsoft tutorial can be found here. <https://code.visualstudio.com/docs/cpp/config-msvc>. If any errors come up while installing or running, please refer to the original documentation in the above link.

Installation Guide:

1. Install VS Code: <https://code.visualstudio.com/download>
2. Install VS C++ extension[s].



3. Download and install build tools for C++
<https://visualstudio.microsoft.com/downloads/#build-tools-for-visual-studio-2022>

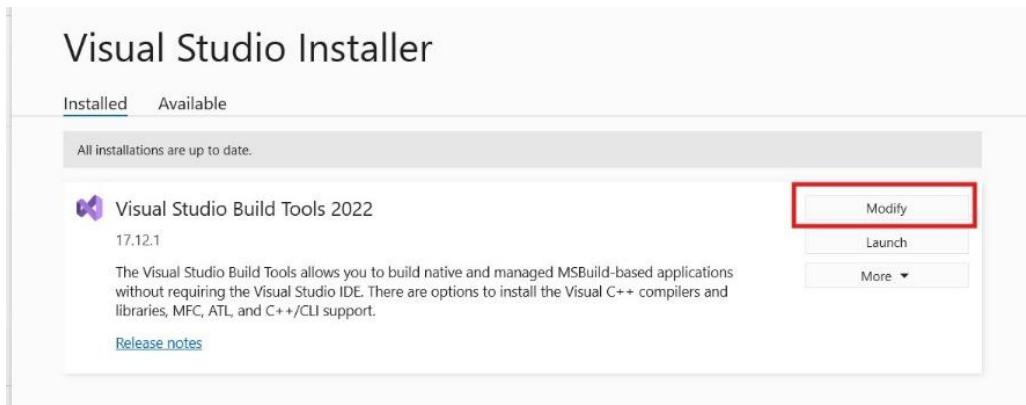
Build Tools for Visual
Studio 2022

These Build Tools allow you to build Visual Studio projects from a command-line interface. Supported projects include: ASP.NET, Azure, C++ desktop, ClickOnce, containers, .NET Core, .NET Desktop, Node.js, Office and SharePoint, Python, TypeScript, Unit Tests, UWP, WCF, and Xamarin. Use of this tool requires a valid Visual Studio license, unless you are building open-source dependencies for your project. See the [Build Tools license](#) for more details.

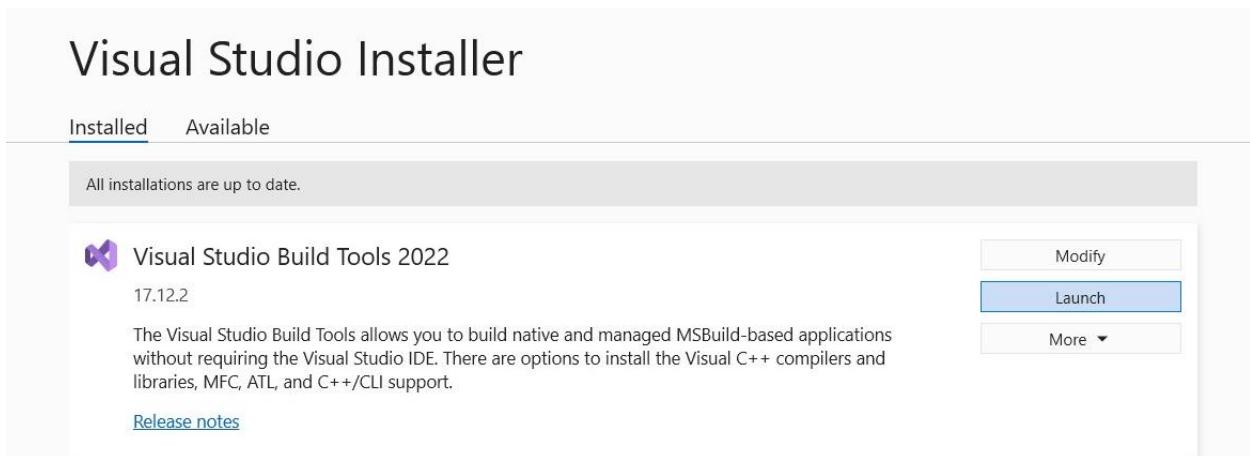
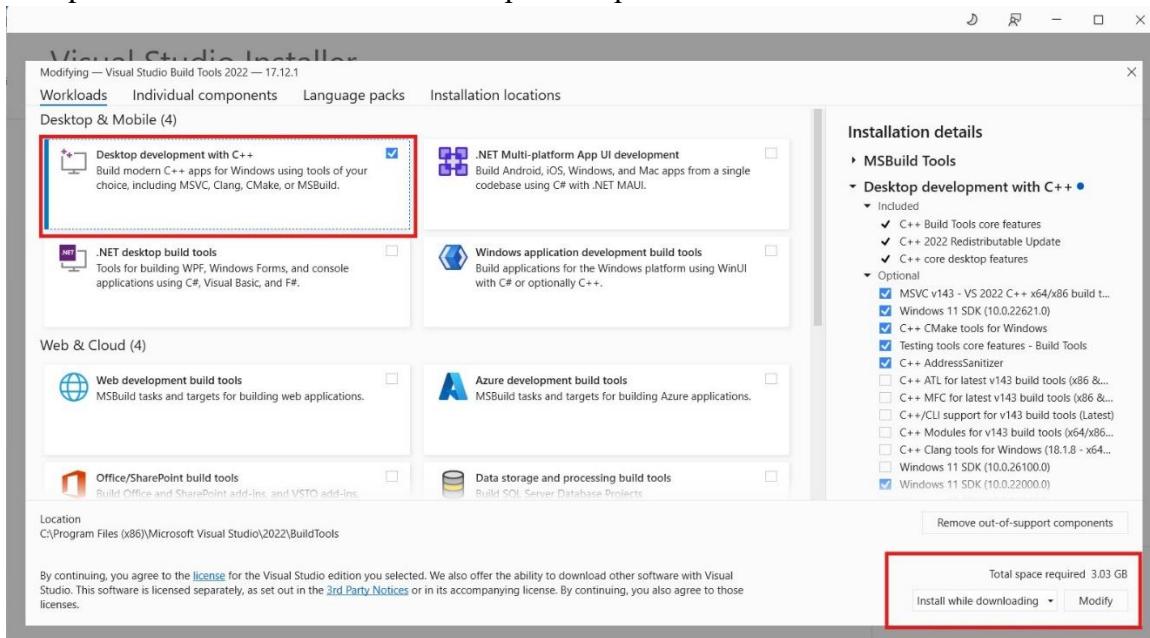
[Download](#)

Are you looking for one of the Visual Studio 2022 long term servicing baselines (LTSCs)? You can find them [here](#).

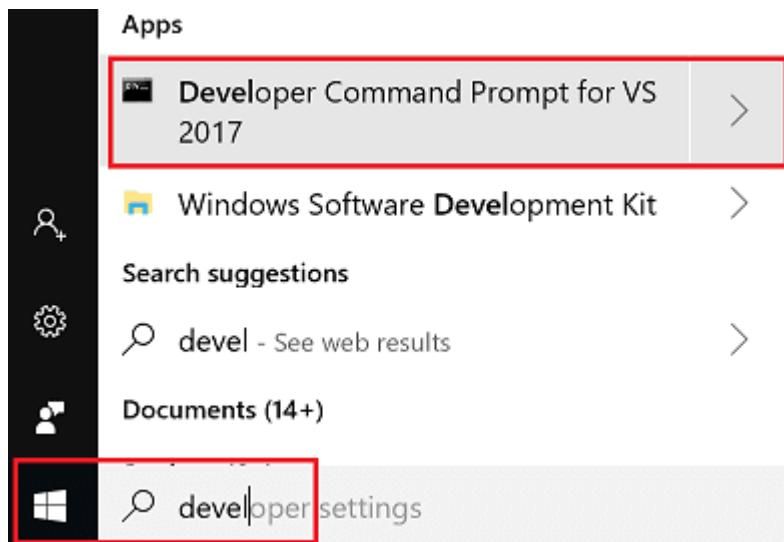
4. Run the installer file. Click on “Modify”



5. Complete the installation with “Desktop development with C++” ticked.



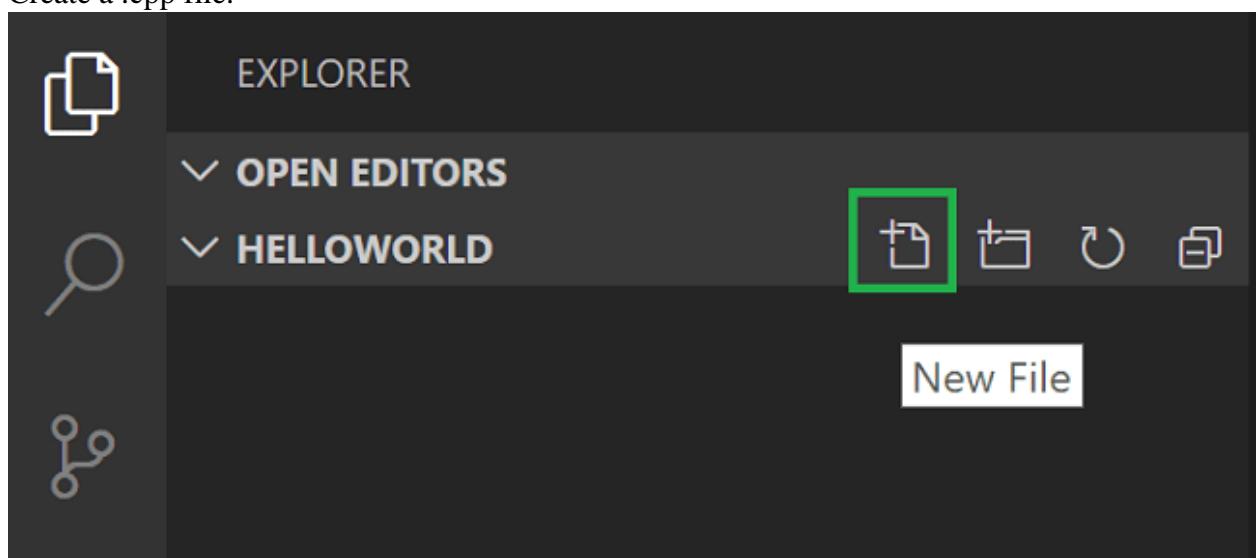
6. Run “Developer Command Prompt for VS [year]” from start menu.



7. Create a directory for your helloworld program in a location at your preference and navigate to that location using command line interface. (commands: mkdir, cd, cd ..)

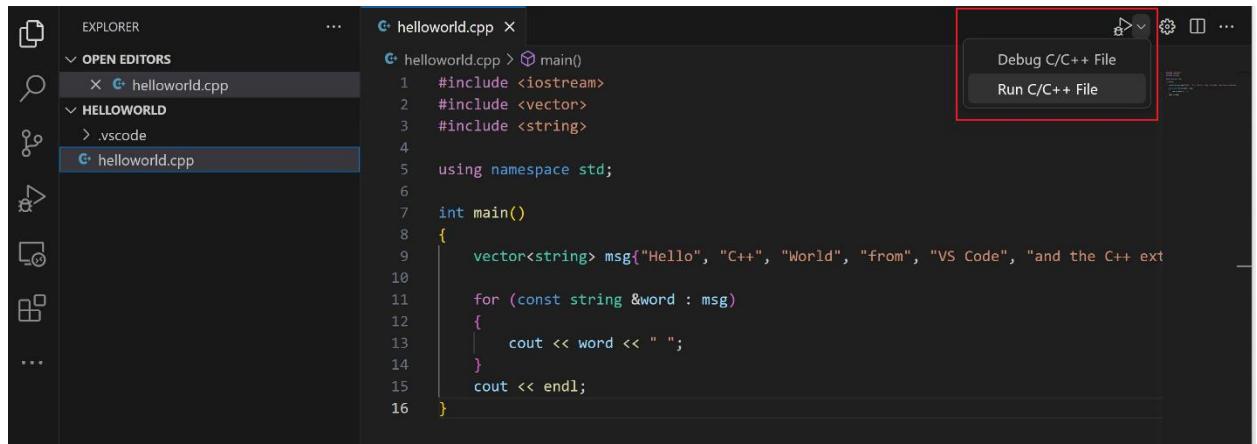
```
mkdir projects
cd projects
mkdir helloworld
cd helloworld
code .
```

8. Open the folder in VS Code using command line interface as above mentioned. (**code .**)
(Opening from CLI is a must)
9. Create a .cpp file.



10. Write a sample helloworld program.

11. Run the program using the below icons and selections.

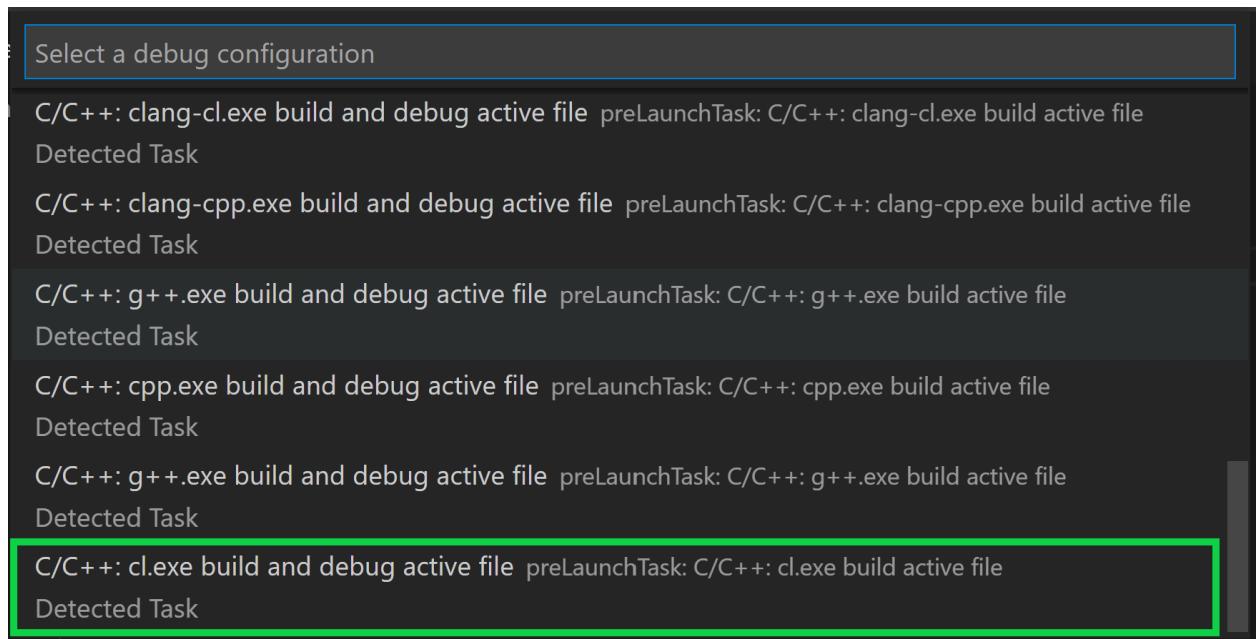


```
#include <iostream>
#include <vector>
#include <string>

using namespace std;

int main()
{
    vector<string> msg("Hello", "C++", "World", "from", "VS Code", "and the C++ ext");
    for (const string &word : msg)
    {
        cout << word << " ";
    }
    cout << endl;
}
```

The screenshot shows the VS Code interface with the 'helloworld.cpp' file open in the editor. The code prints the string "Hello C++ World". In the top right corner of the editor, there is a context menu with two options: 'Debug C/C++ File' and 'Run C/C++ File'. The 'Run C/C++ File' option is highlighted with a red box.



Now your helloworld output should be visible in the terminal.

Appendix 01

(Below part is an additional info. Not necessary to follow)

Comparison between each C++ installation methods on VS Code with pros and cons.

Each method mentioned has its own merits and trade-offs, and the choice depends on factors like ease of use, download size, and the student's level of expertise. Here's a breakdown to decide:

1. Using Visual Studio Build Tools (via winget):

- **Advantages:**
 - Minimal Visual Studio installation (no need to install the full IDE).
 - Provides robust, official Microsoft tools for C++ development, including the MSVC compiler.
 - Automatically sets up environment variables for use with VS Code.
- **Drawbacks:**
 - Larger download compared to MinGW or MSYS2.
 - May feel more complex due to command-line installation.

Recommendation: Ideal for students on Windows who need a reliable, modern C++ compiler without the full Visual Studio IDE.

2. Using MinGW:

- **Advantages:**
 - Lightweight installation compared to Microsoft tools.
 - Works seamlessly with VS Code for beginners.
 - Straightforward setup process.
- **Drawbacks:**
 - MinGW's tools are not as up-to-date or performant as Microsoft's MSVC.
 - Limited debugging capabilities compared to MSVC.

Recommendation: A good choice for students who want a quick and simple way to start with C++.

3. Using MSYS2:

- **Advantages:**
 - Offers up-to-date GCC tools and a Unix-like environment on Windows.
 - Suitable for students familiar with Linux/Unix systems.
- **Drawbacks:**
 - Slightly more complex installation process (requires multiple package installations).
 - Heavier download size than MinGW.

Recommendation: Best for students who are comfortable with Unix-like tools or need GCC features.

4. Using Full Visual Studio:

- **Advantages:**
 - Comprehensive solution with built-in tools, debugging, and GUI.
 - Easy to switch between VS Code and Visual Studio for advanced projects.
- **Drawbacks:**
 - Significantly larger download size.
 - Overkill for basic C++ projects in VS Code.

Recommendation: Use if students are expected to use Visual Studio IDE later or require full functionality.
