

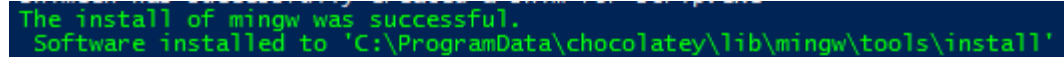
VS code setup steps

# Installing package manager

- <https://chocolatey.org/>
- Recommend to install from PowerShell.exe
  - Check command **Get-ExecutionPolicy**
  - **Set-ExecutionPolicy AllSigned**
  - Run following command:  
**Set-ExecutionPolicy Bypass -Scope Process -Force; iex ((New-Object System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))**

# Preparations

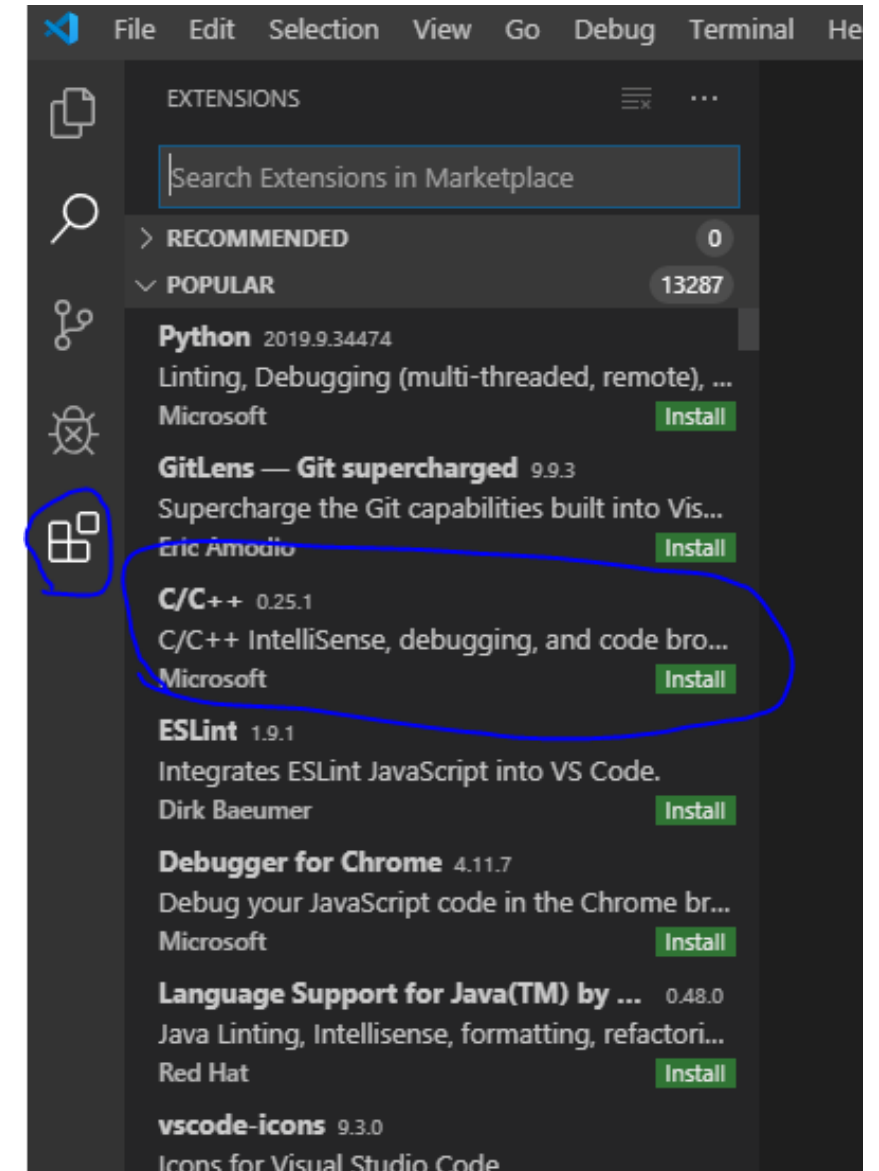
- Install compiler
  - **choco install mingw -y**
- Install VisualStudio Code
  - **choco install vscode -y**

A screenshot of a terminal window with a dark blue background and green text. The text indicates a successful installation of mingw and provides the installation path.

```
The install of mingw was successful.  
Software installed to 'C:\ProgramData\chocolatey\lib\mingw\tools\install'
```

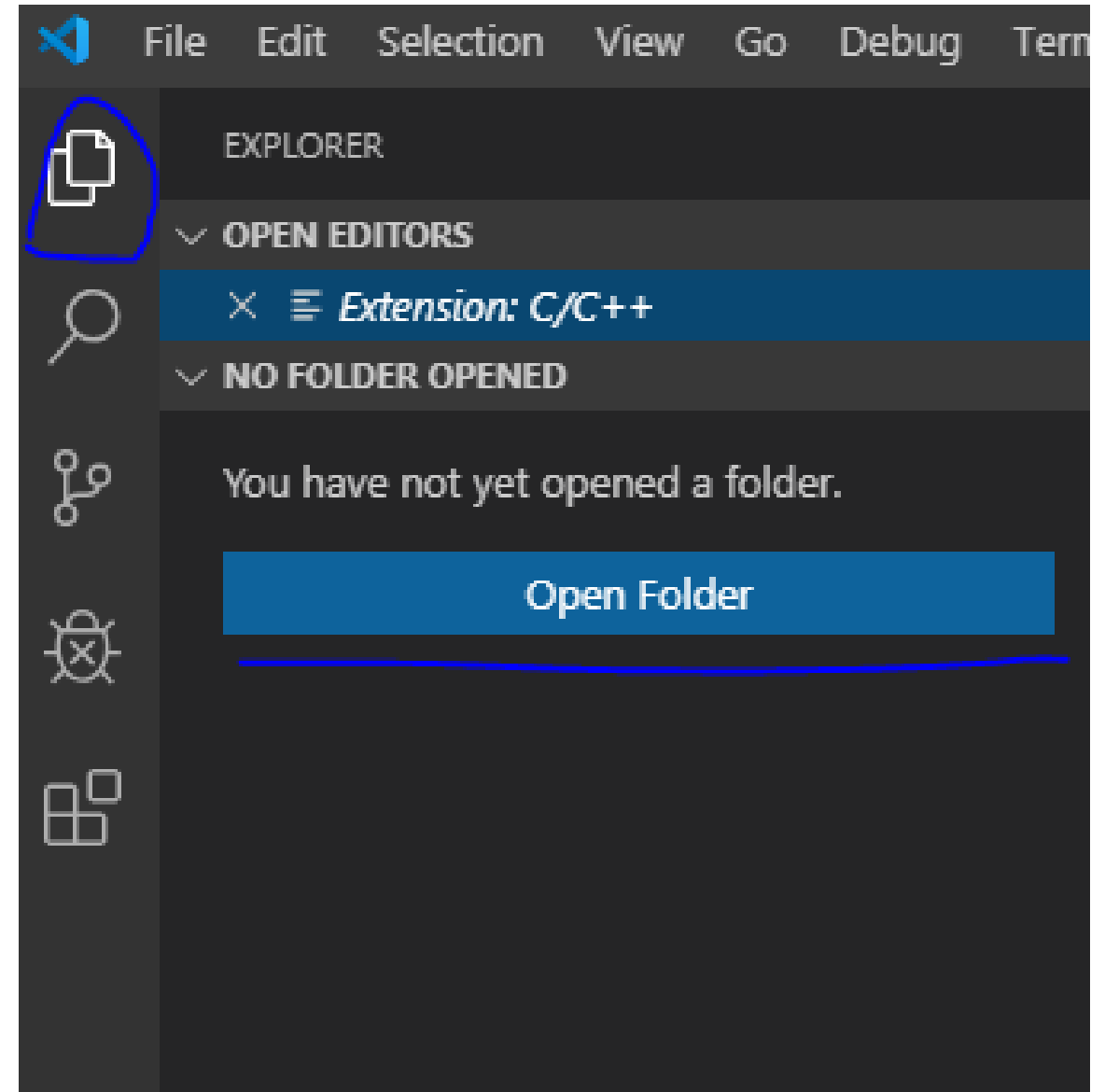
# Install VS Code extension

- We are interesting for C/C++, but you are can install also Python/C#/etc.



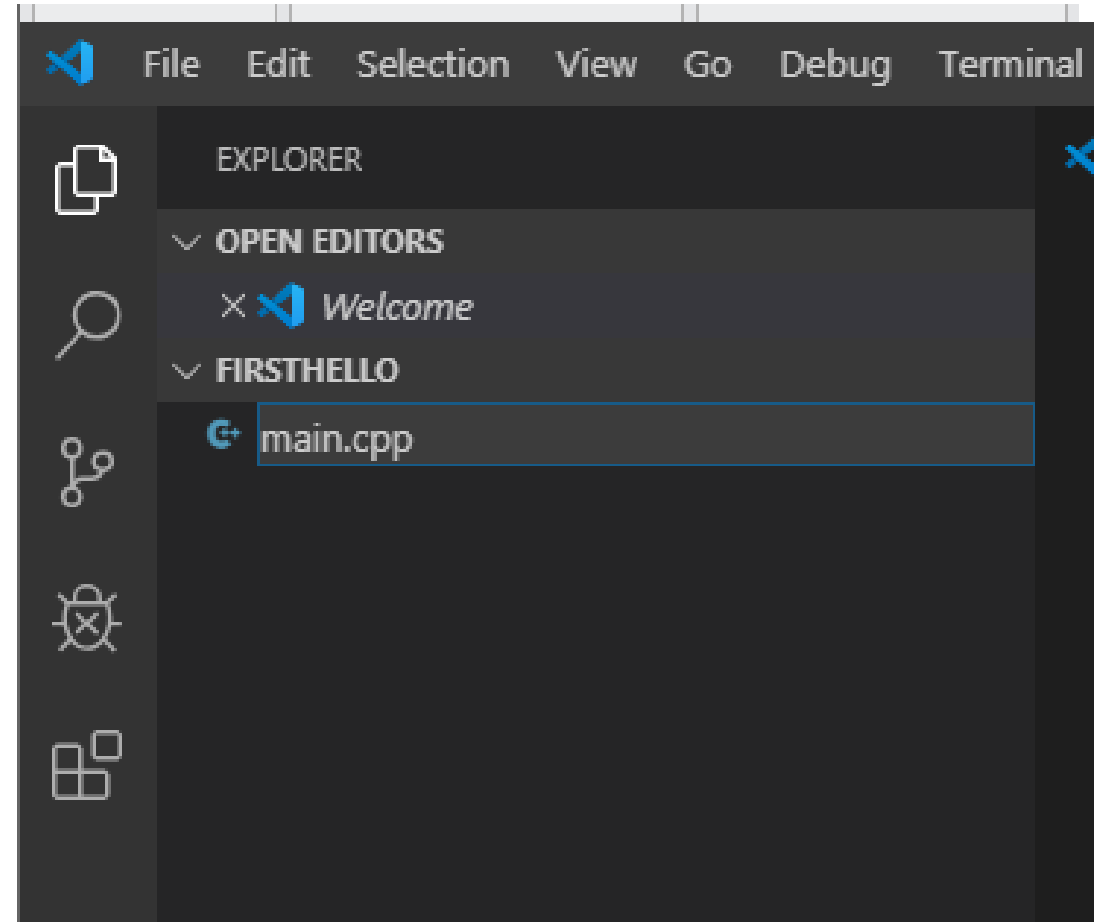
# Create first env

- Click “Explorer” and chose folder



# Create first env

- Click “Explorer” and chose folder
- Click on add file and type “main.cpp”

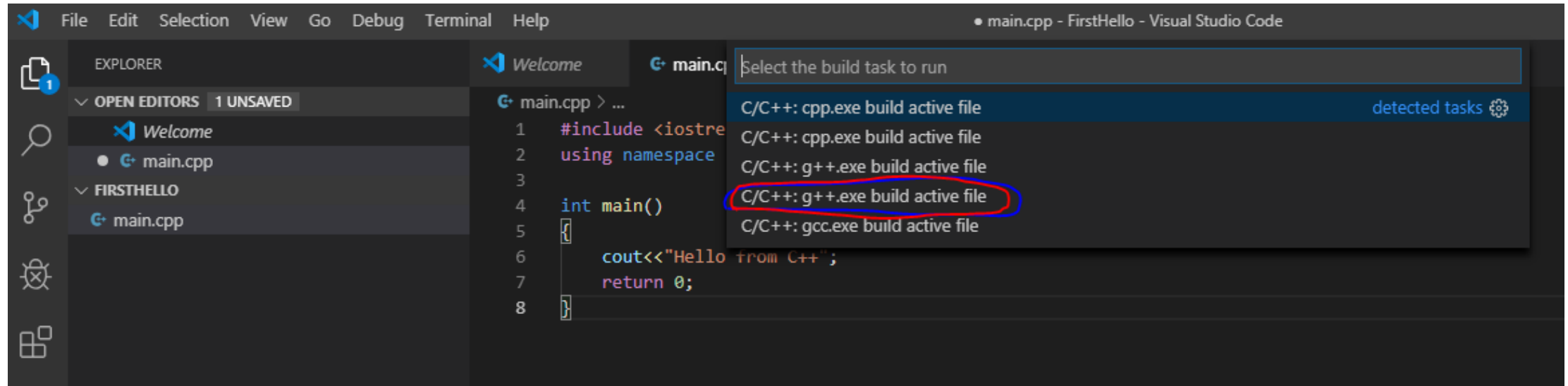


# Write “dummy” code in the editor

```
#include <iostream>  
using namespace std;
```

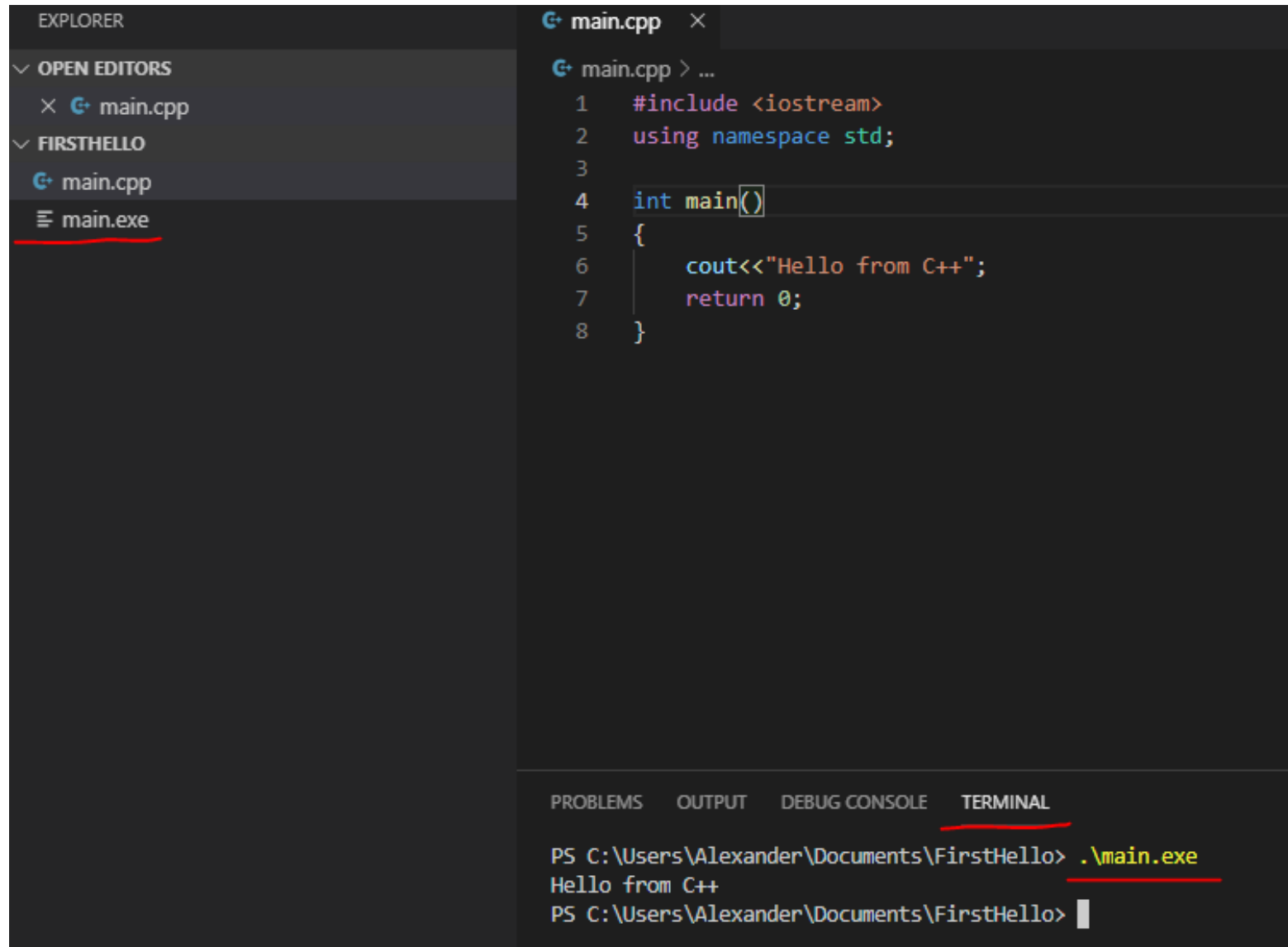
```
int main()  
{  
    cout<<"Hello from C++";  
    return 0;  
}
```

# Click Ctrl-Shift-B





# Running dummy executable



The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar shows a project named 'FIRSTHELLO' with two files: 'main.cpp' and 'main.exe'. 'main.exe' is highlighted with a red underline. The main editor area shows the code for 'main.cpp', which includes the `<iostream>` header, uses the `std` namespace, and contains a `main` function that prints 'Hello from C++' and returns 0. At the bottom, the TERMINAL panel shows the command `.\main.exe` being executed, resulting in the output 'Hello from C++'.

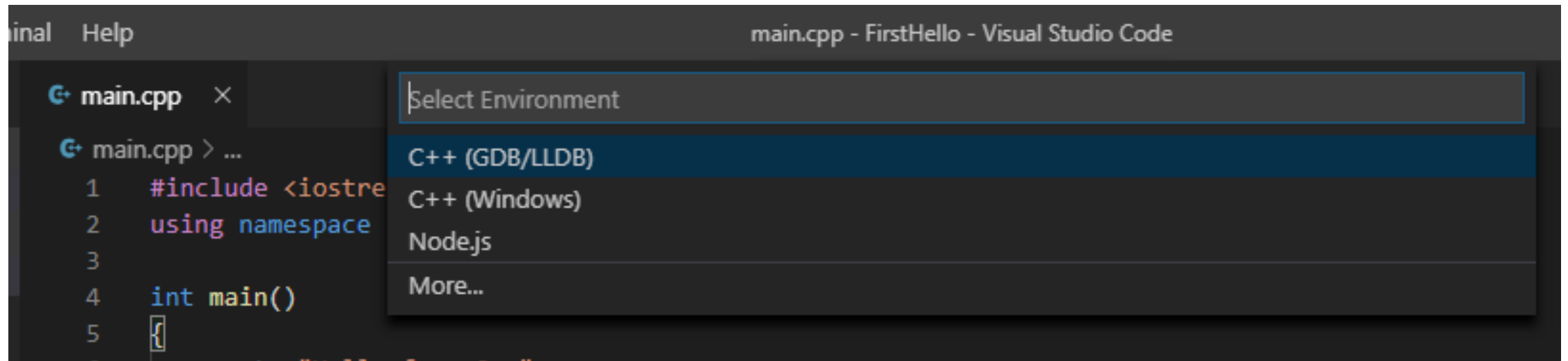
```
EXPLORER
└─ OPEN EDITORS
    × main.cpp
└─ FIRSTHELLO
    main.cpp
    main.exe

main.cpp
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      cout<<"Hello from C++";
7      return 0;
8  }
```

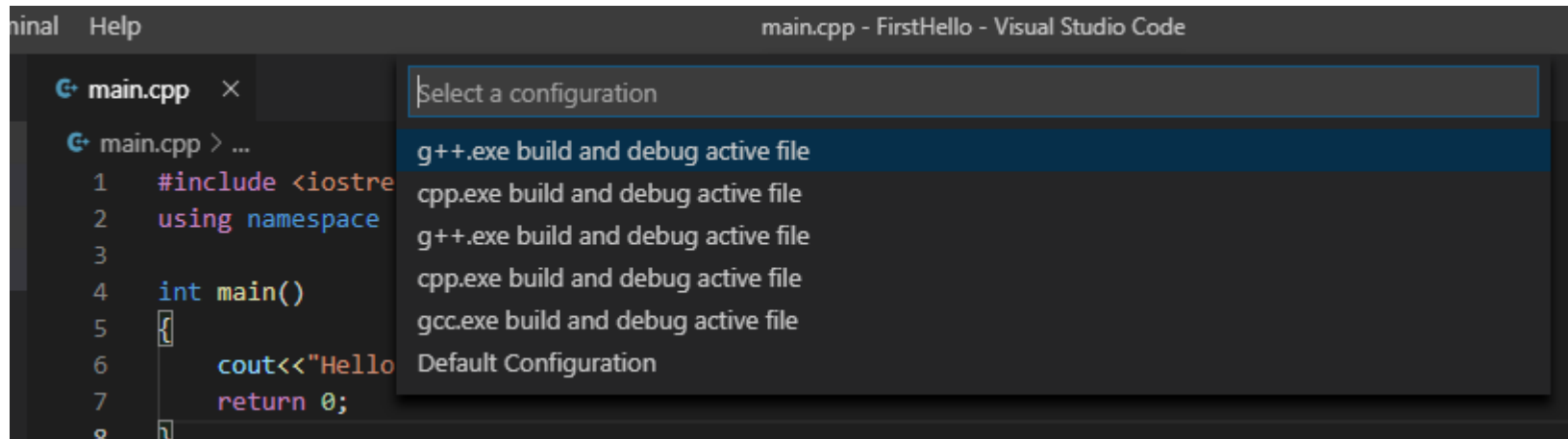
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS C:\Users\Alexander\Documents\FirstHello> .\main.exe
Hello from C++
PS C:\Users\Alexander\Documents\FirstHello>
```

# Debugging (press F5)

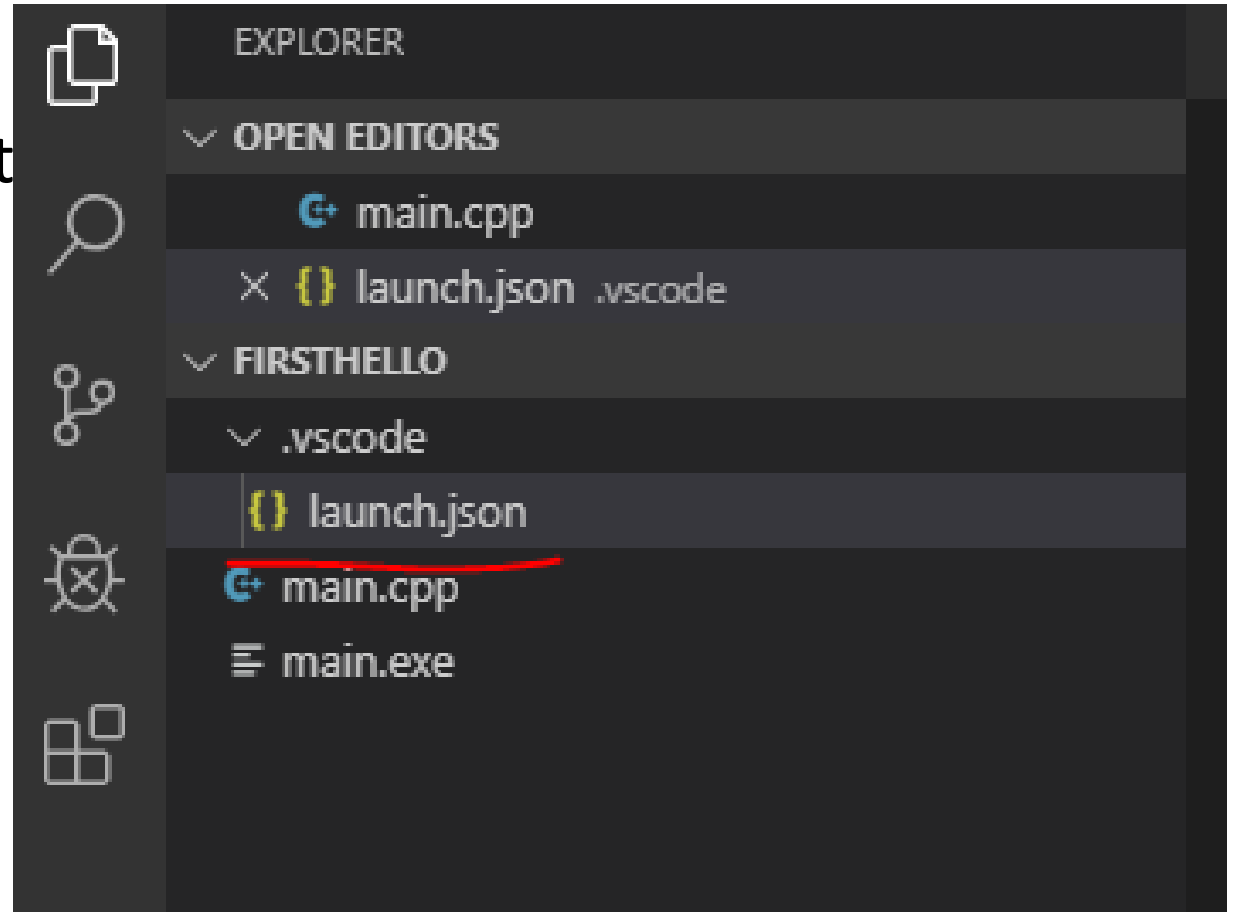


# Debugging



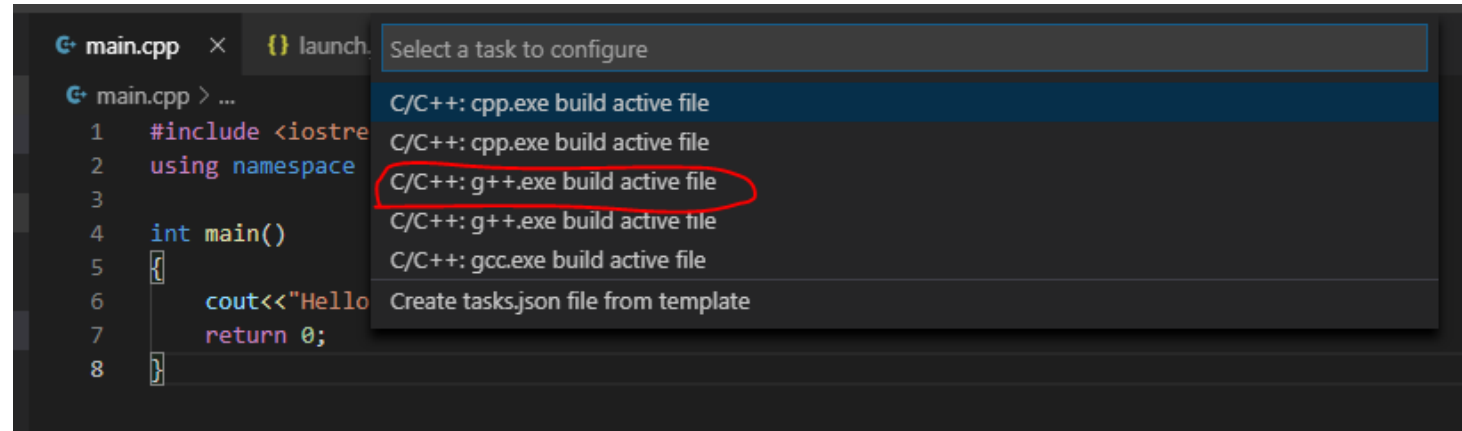
# Debugging

- launch.json will be created
- Ensure that **miDebuggerPath** set properly
- Press F5 again

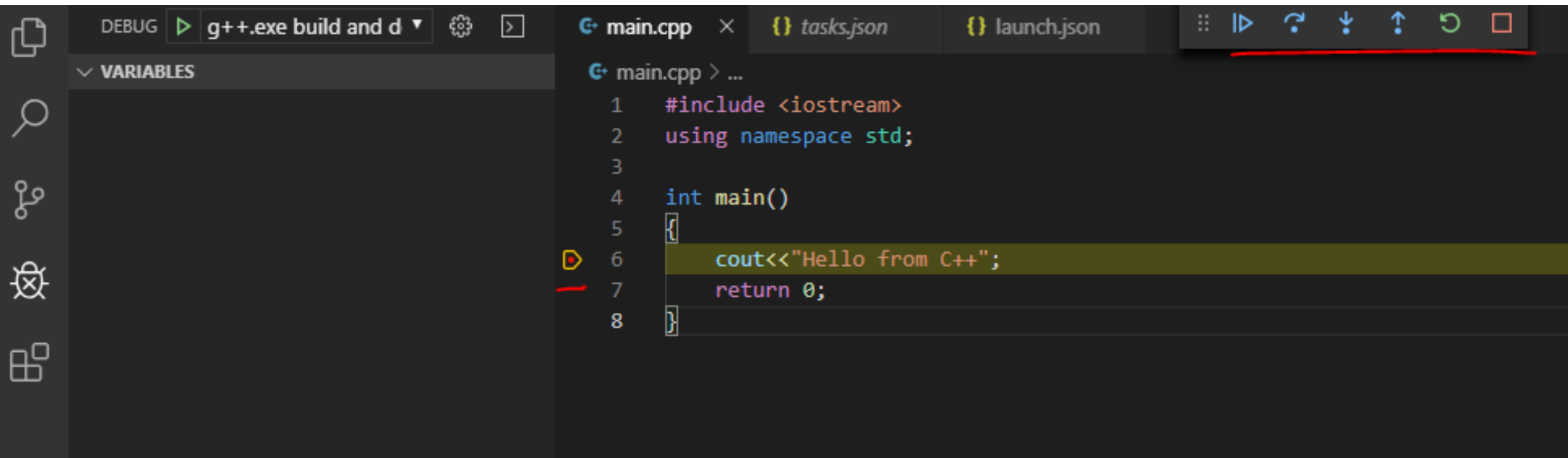


# Debugging

- task.json will be created



# Debugging



# Sample tasks.json

```
{  
  "version": "2.0.0",  
  "tasks": [  
    {  
      "type": "shell",  
      "label": "g++.exe build active file",  
      "command": "C:\\ProgramData\\chocolatey\\bin\\g++.exe",  
      "args": [  
        "-g",  
        "${file}",  
        "-o",  
        "${fileDirname}\\${fileBasenameNoExtension}.exe"  
      ],  
      "options": {  
        "cwd": "C:\\ProgramData\\chocolatey\\bin"  
      },  
      "problemMatcher": [  
        "$gcc"  
      ]  
    }  
  ]  
}
```

# Sample launch.json

```
{
  "version": "0.2.0",
  "configurations": [
    {
      "name": "g++.exe build and debug active file",
      "type": "cppdbg",
      "request": "launch",
      "program": "${fileDirname}\\${fileBasenameNoExtension}.exe",
      "args": [],
      "stopAtEntry": false,
      "cwd": "${workspaceFolder}",
      "environment": [],
      "externalConsole": false,
      "MIMode": "gdb",
      "miDebuggerPath": "C:\\ProgramData\\chocolatey\\bin\\gdb.exe",
      "setupCommands": [
        {
          "description": "Enable pretty-printing for gdb",
          "text": "-enable-pretty-printing",
          "ignoreFailures": true
        }
      ],
      "preLaunchTask": "g++.exe build active file"
    }
  ]
}
```



# Alternatives

- Use Visual Studio Community
- xCode
- etc.