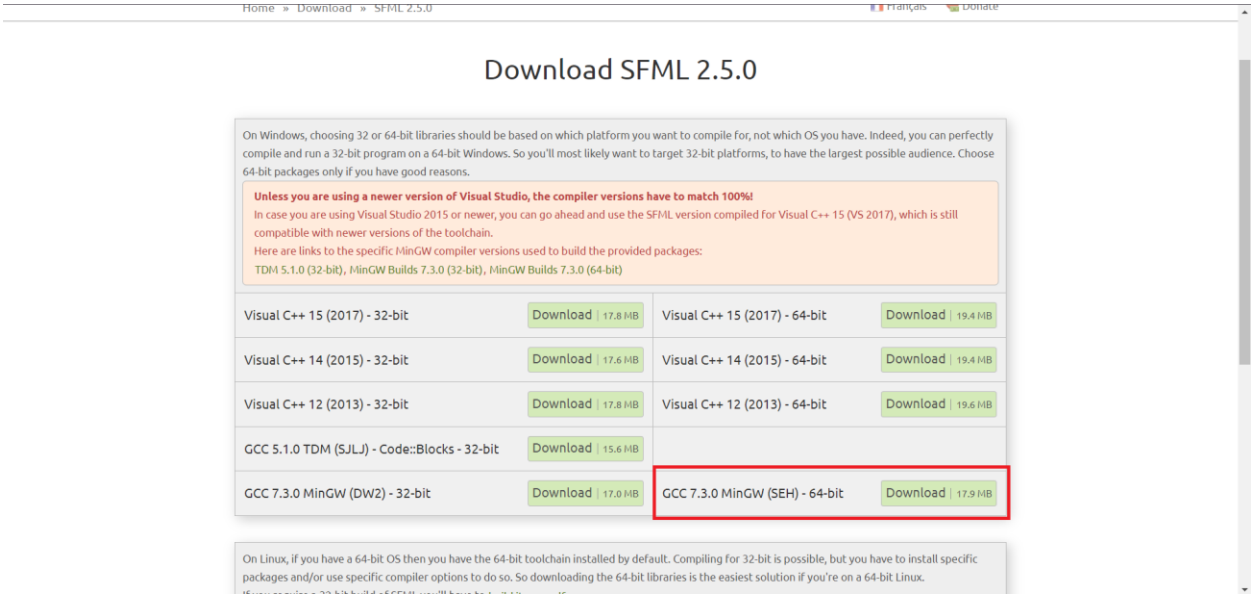
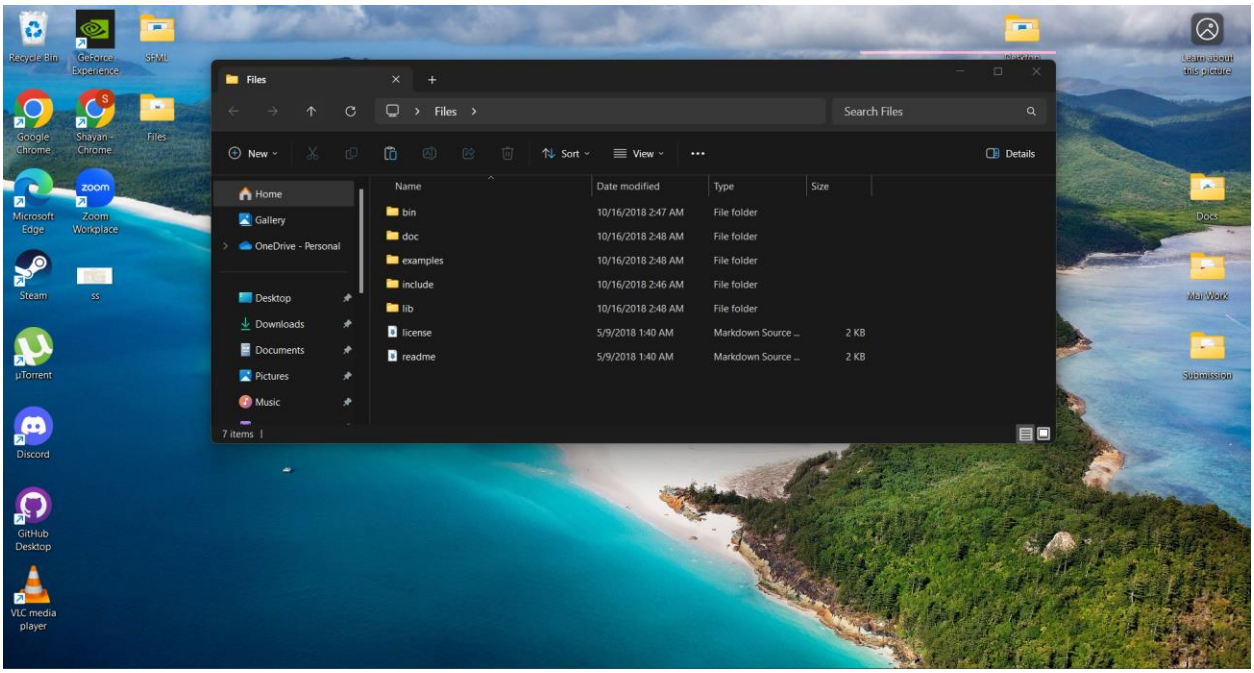
SFML Installation Guide

Step 1: Downloading SFML

Let's begin the installation by heading over to the SFML website and downloading the correct version of SFML. This link will open up the download page for SFML 2.5.0. The reason we are not installing the latest version is that the version of MinGW that is installed on your PCs is old, and so the latest version of SFML will not be compatible with your version. Now click on the button to download the SFML library:

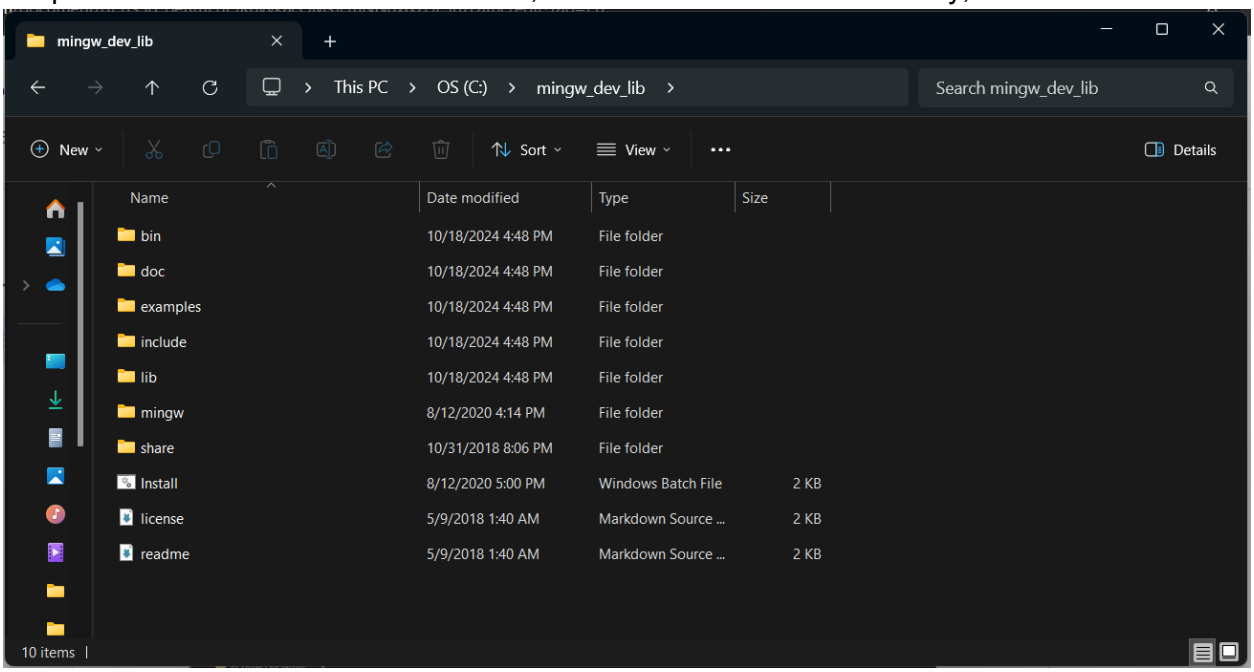


Once your SFML library has been downloaded extract it to a folder. Here, for example, I have extracted the library inside a **Files** folder on the Desktop:



Step 2: Installing SFML

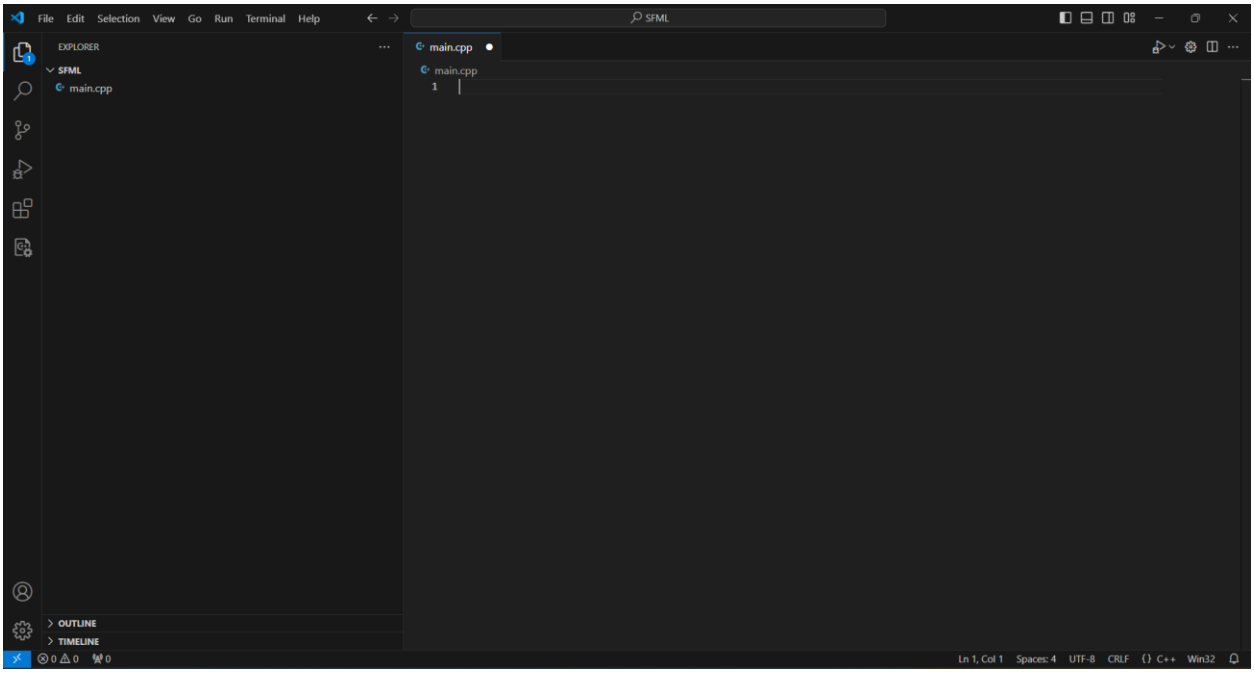
Now copy the files in the directory, and head over to your MinGW installation folder. Here you will paste the contents of the **Files** folder, which contains the SFML library, and :



Once, you have copied the contents, SFML will be ready to use in your projects.

Step3: Testing our Installation

Now that SFML is completely installed, we will open up a sample project folder and create a simple C++ file. Here I've created a folder titled **SFML** on the Desktop:

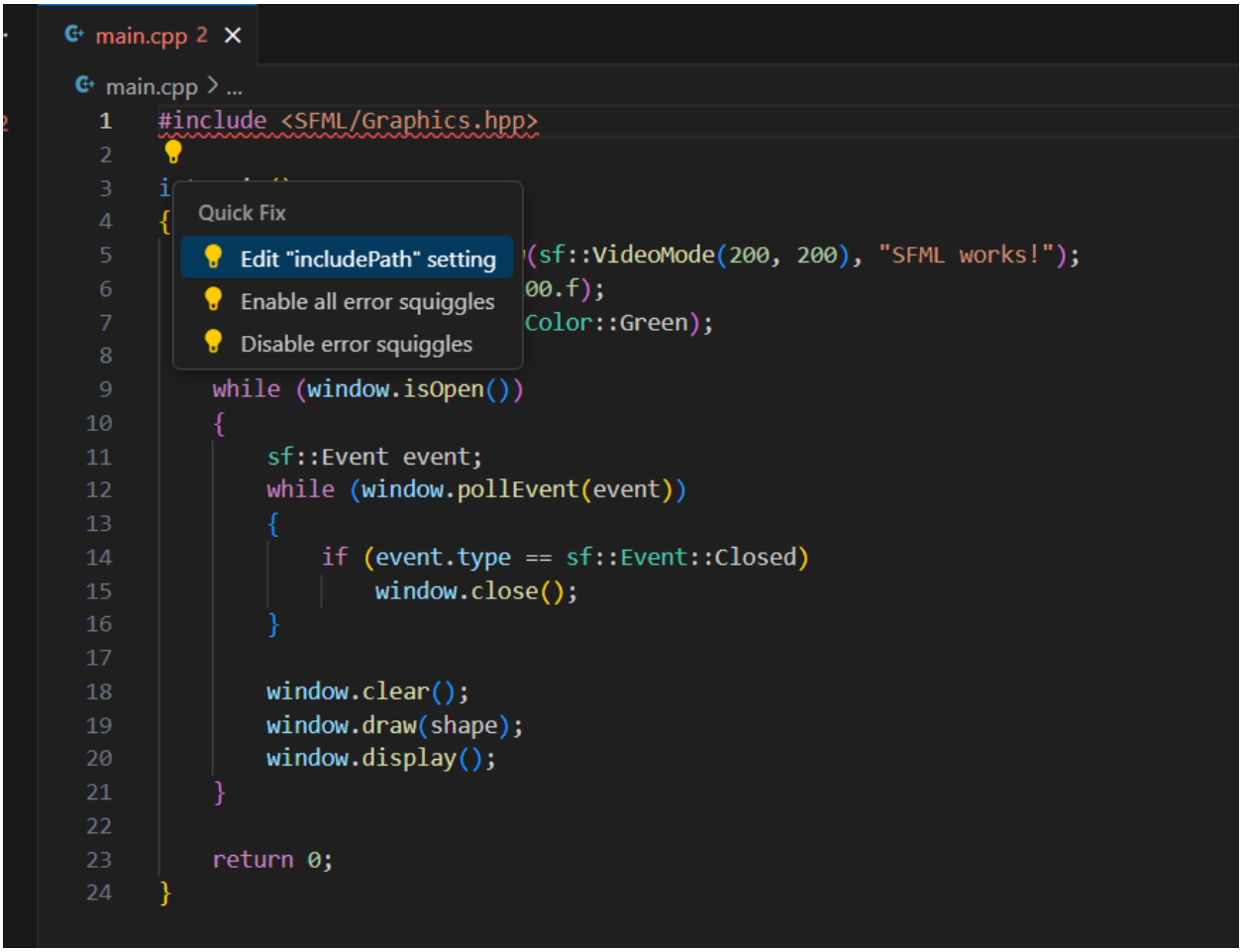


Next, copy and paste the following code inside the **main.cpp** file:

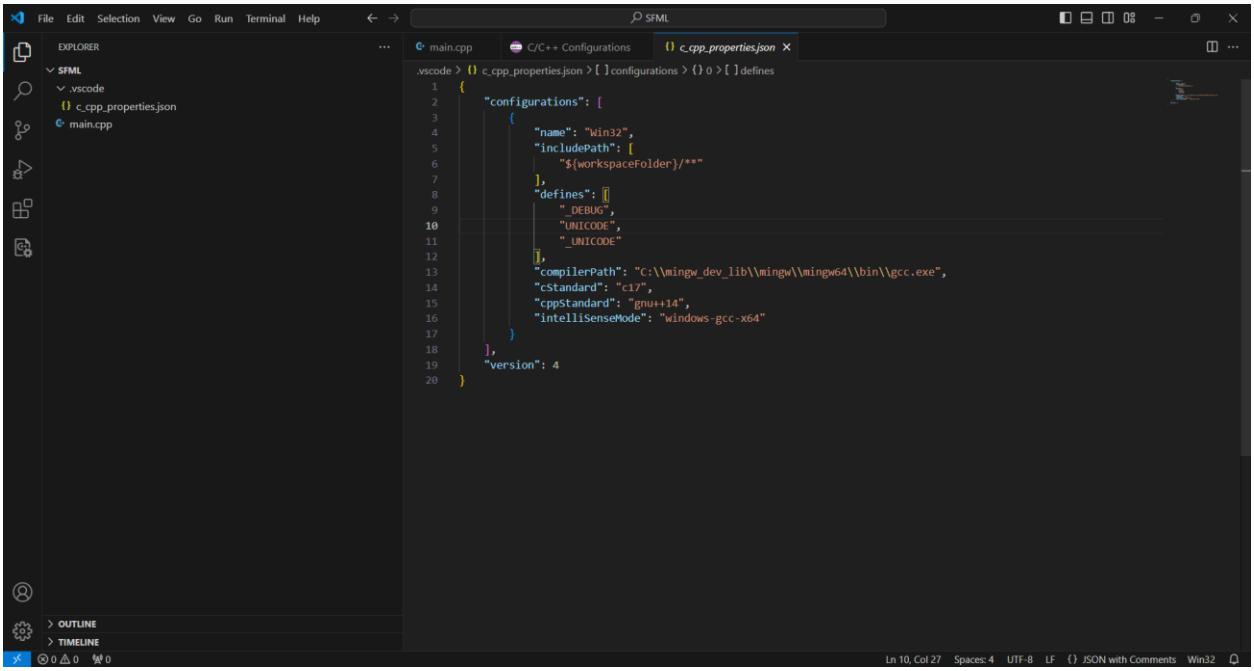
```cpp
#include <SFML/Graphics.hpp>

int main()
{
    sf::RenderWindow window(sf::VideoMode(200, 200), "SFML works!");
    sf::CircleShape shape(100.f);
    shape.setFillColor(sf::Color::Green);

    while (window.isOpen())
    {
        sf::Event event;
        while (window.pollEvent(event))
        {
            if (event.type == sf::Event::Closed)
                window.close();
        }

        window.clear();
        window.draw(shape);
        window.display();
```

```
    }

    return 0;
}
```

Ss you can see Visual Studio will give you an include error that the SFML graphics library is not detected. Though this error is irrelevant to the installation, you can fix this by simply clicking on the **light bulb icon** to open the **Quick Fix** bar. Here, select the **Edit "includePath" setting** button:



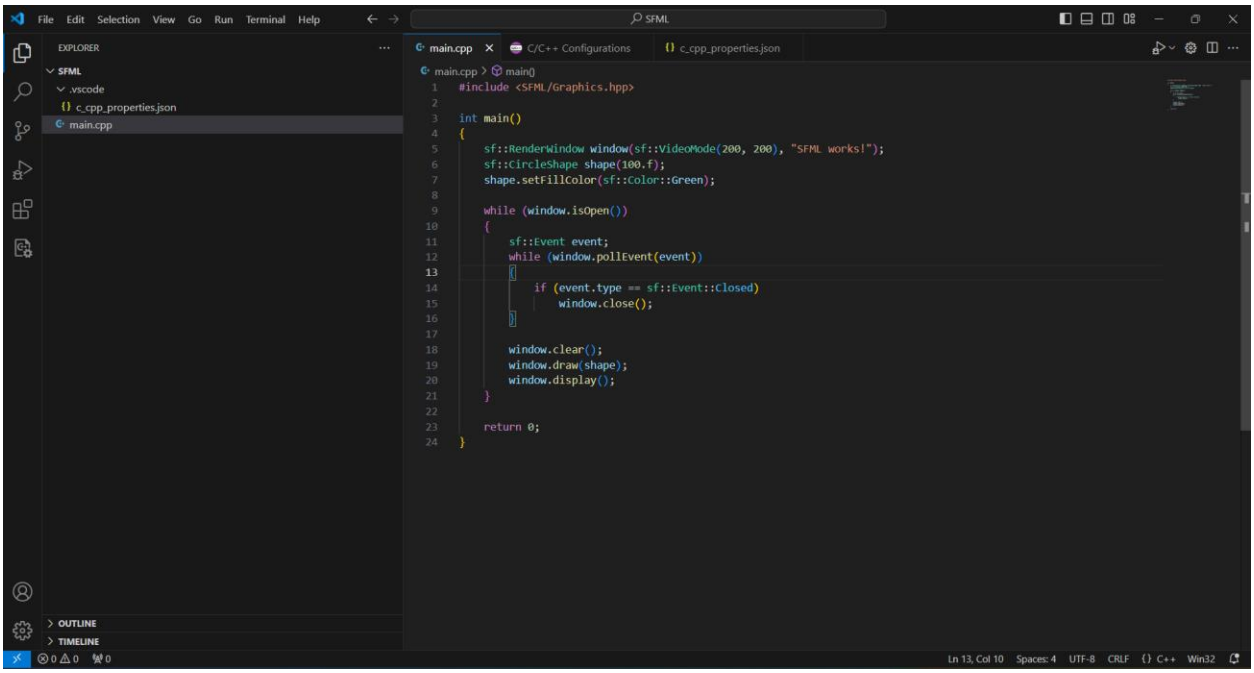
This will create a `.vscode` folder in your project directory containing a `c_cpp_properties.json` file. Next, you will open up the JSON file:



Here inside the `includePath` list, you will add a path to your MinGW **include** folder as follows:

```
.vscode > {} c_cpp_properties.json > [ ] configurations > {} 0 > [ ] includePath > ...
{
    "configurations": [
        {
            "name": "Win32",
            "includePath": [
                "${workspaceFolder}/**",
                "C:\\mingw_dev_lib\\include"
            ],
            "defines": [
                "_DEBUG",
                "UNICODE",
                "_UNICODE"
            ],
            "compilerPath": "C:\\mingw_dev_lib\\mingw\\mingw64\\bin\\gcc.exe",
            "cStandard": "c17",
            "cppStandard": "gnu++14",
            "intelliSenseMode": "windows-gcc-x64"
        }
    ],
    "version": 4
}
```

This should eliminate the **include** error:



Now all we need to do is to compile our `main.cpp` file. To do this, we will open up the terminal and run the following command:

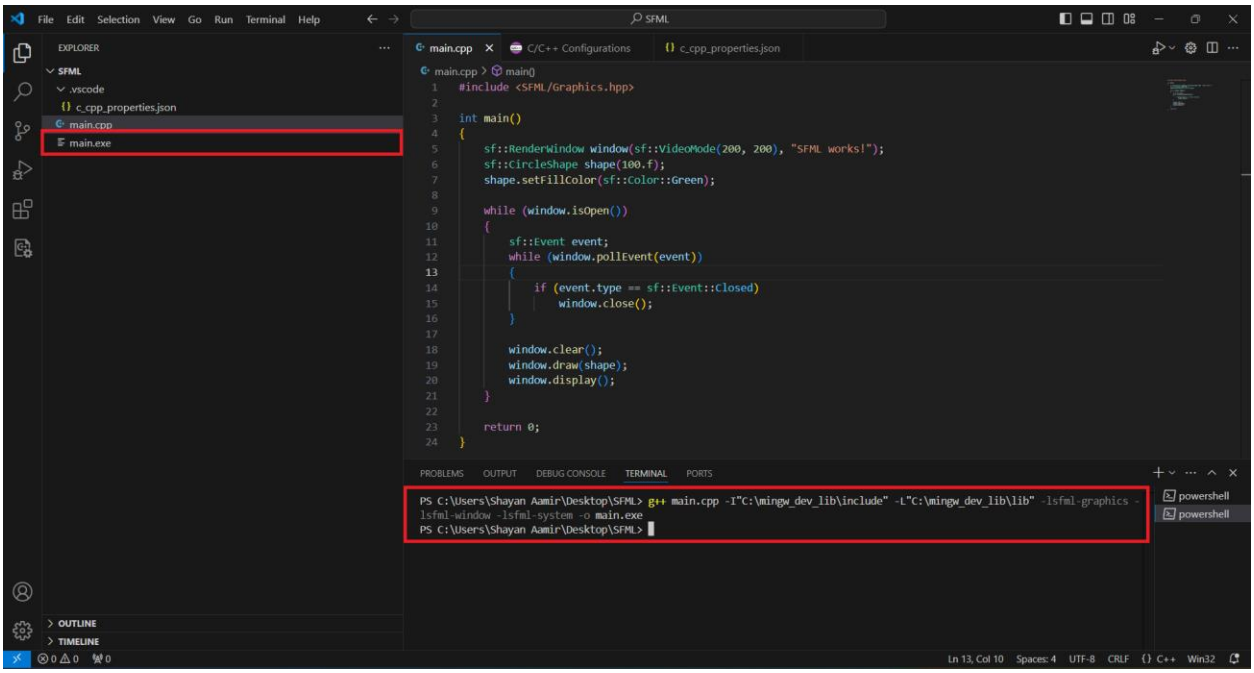
```
g++ main.cpp -I"<Path-To-Your-MinGW-Include-Folder>" -L"<Path-To-Your-MinGW-Lib-Folder>" -lsfml-graphics -lsfml-window -lsfml-system -o main.exe
```

Make sure to replace the tags with your actual path. For example, replacing the tags for my laptops will result in the following command:

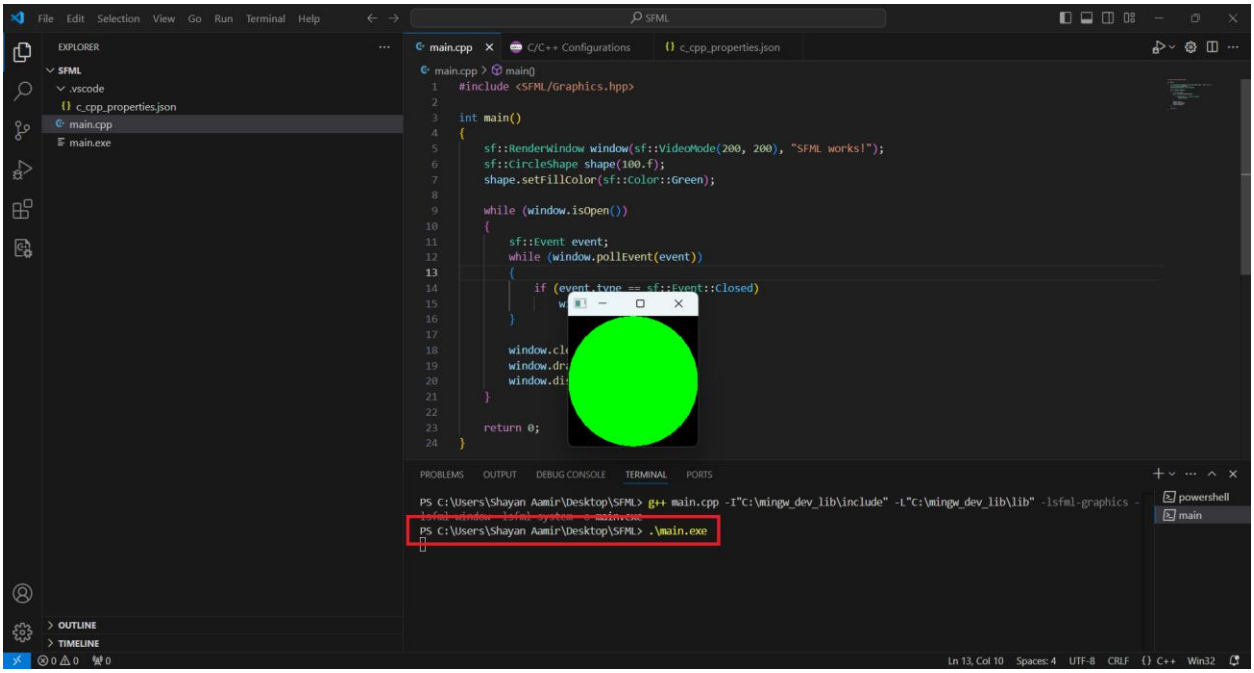
```
g++ main.cpp -I"C:\mingw_dev_lib\include" -L"C:\mingw_dev_lib\lib" -lsfml-graphics -lsfml-window -lsfml-system -o main.exe
```

Optionally, you can modify the `-o` tag to change the name of your output file.

Now open up the terminal, and run the command. This will create a `main.exe` file in your project directory as shown below:



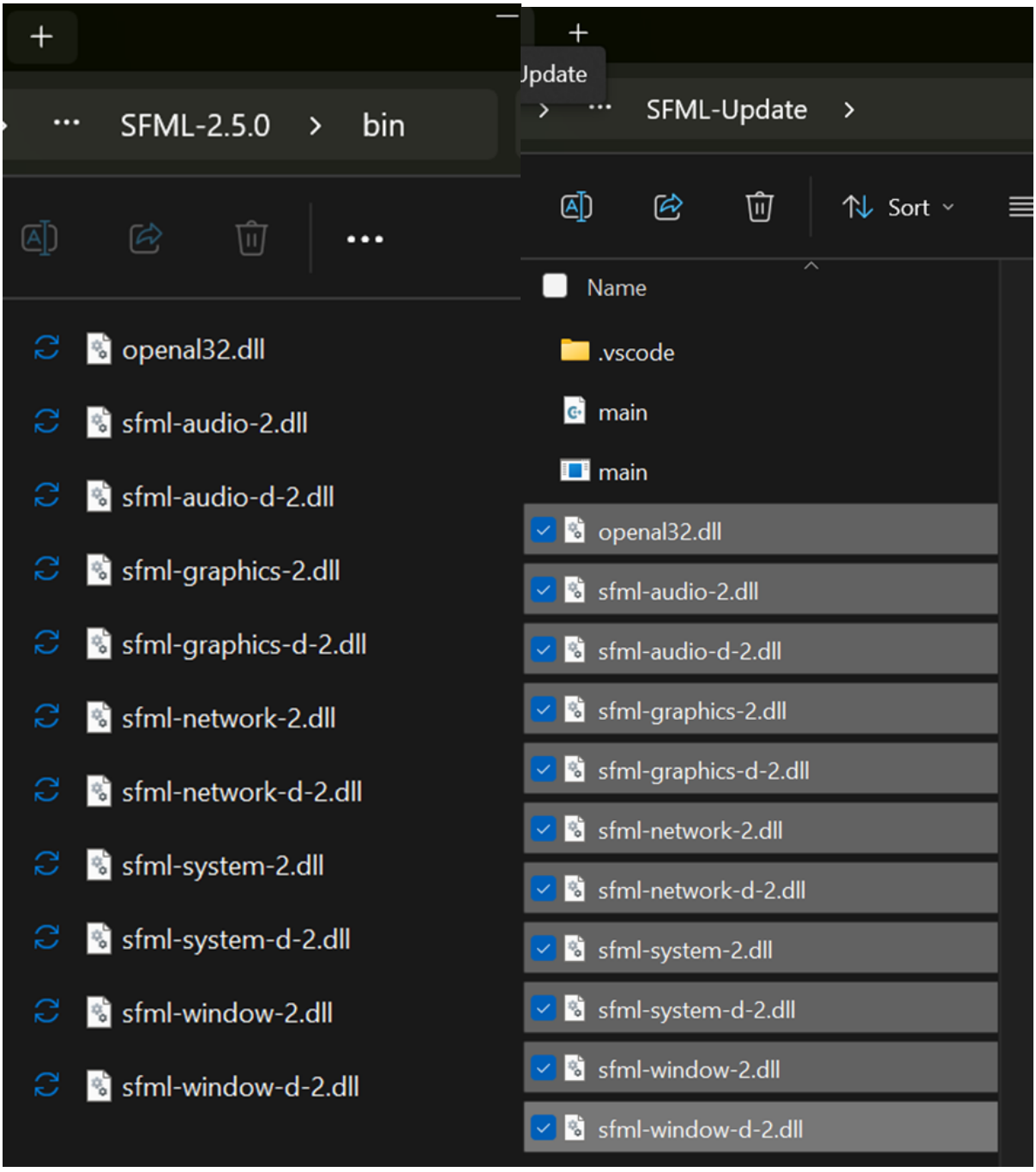
Now we can run the `main.exe` file and here is what the output looks like:



Alternative Method:

In case your program does not run, (or if you are using Lab PC's) go to the **Files** folder and copy all of the files in the SMFL **bin** folder to the folder where your **main.cpp** is located (And other lab files will reside) like as follows:

So copy everything from here -> to here



Then, all we need to do is to compile our `main.cpp` file. To do this, we will open up the terminal

and run the following command:

```
g++ main.cpp -I"<Path-To-Your-SMFL-Include-Folder>" -L"<Path-To-Your-SMFL-Lib-Folder>" -lsfml-graphics -lsfml-window -lsfml-system -o main.exe
```

Note that this SMFL Include Folder and SMFL Lib Folder is the .zip that you downloaded at the start of the lab!