



CS 210 C++ and Python Visual Studio Guide

Overview

This guide will walk you through the process of combining two different programming languages, in this case C++ and Python, for a single project in the Visual Studio integrated development environment (IDE). Before beginning to move through the following steps, you should already have an understanding of how to access Visual Studio and create a new Visual Studio project where you can write C++ code.

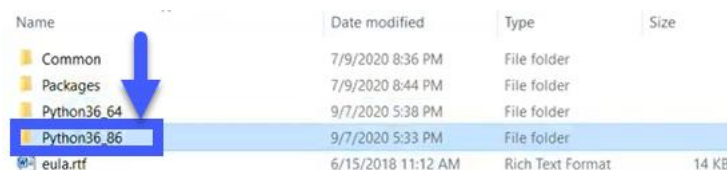
Note: If you are working in the Virtual Lab and do not have Visual Studio downloaded on your own machine, skip the Downloading Python section and instead begin at Connecting Python Libraries in Visual Studio. This is because the Virtual Lab will already have all the Python components you need to successfully complete this process downloaded for your use.

Downloading Python

1. Navigate to the [Download Python](#) page and click on the **Download Python** button to download the latest version. Then, follow the on-screen prompts to complete the installation process. Pay attention to the folder where Python's files are downloaded to, as you will need to be able to access them.



2. Locate the downloaded Python files that are now on your computer. Here you can observe that Python offers you a choice between a 64-bit and a 32-bit version. We are going to be using the 32-bit version so it will run more easily on many different types of machines. The automatic folder name for this is Python36_86. Double-click on the Python36_86 folder to review its contents.



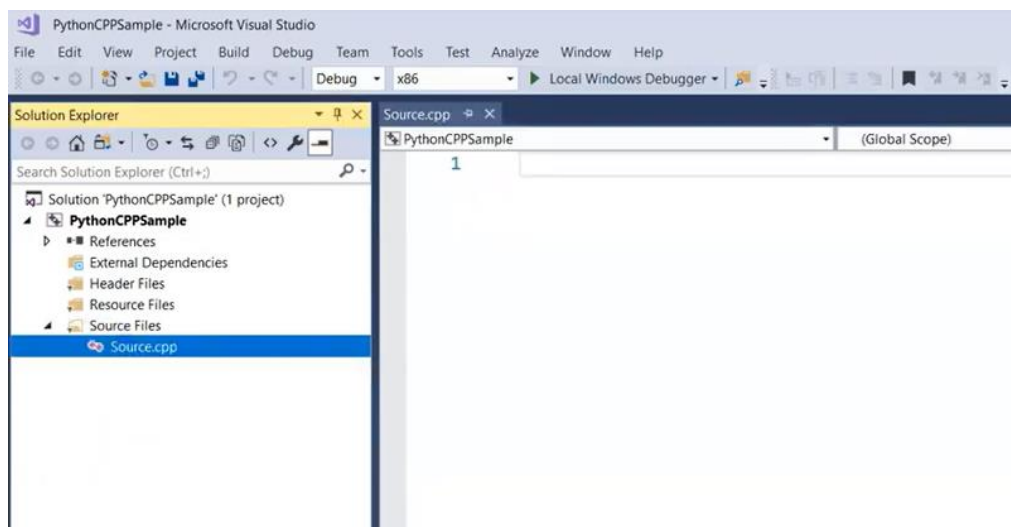
Name	Date modified	Type	Size
Common	7/9/2020 8:36 PM	File folder	
Packages	7/9/2020 8:44 PM	File folder	
Python36_64	9/7/2020 5:38 PM	File folder	
Python36_86	9/7/2020 5:33 PM	File folder	
eula.rtf	6/15/2018 11:12 AM	Rich Text Format	14 KB

3. Inside the Python36_86 folder, there is both an "include" folder and a "libs" folder. You will be using these in a moment for your Visual Studio work, so it is important to take note of them now before moving on.

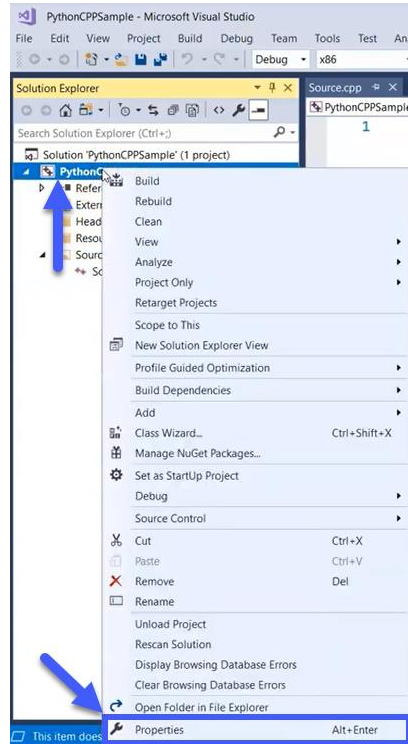
Name	Date modified	Type	Size
DLLs	9/7/2020 5:33 PM	File folder	
Doc	9/7/2020 5:33 PM	File folder	
include	9/7/2020 5:33 PM	File folder	
Lib	9/7/2020 5:33 PM	File folder	
libs	9/7/2020 5:33 PM	File folder	
Scripts	9/7/2020 5:33 PM	File folder	
tcl	9/7/2020 5:33 PM	File folder	
Tools	9/7/2020 5:33 PM	File folder	
LICENSE.txt	6/27/2018 2:54 AM	Text Document	30 KB
NEWS.txt	6/27/2018 2:54 AM	Text Document	395 KB
python.exe	6/27/2018 2:50 AM	Application	96 KB
python.pdb	6/27/2018 2:50 AM	Program Debug D...	412 KB
python3.dll	6/27/2018 2:47 AM	Application extens...	58 KB
python36.dll	6/27/2018 2:47 AM	Application extens...	3,225 KB
python36.pdb	6/27/2018 2:47 AM	Program Debug D...	9,380 KB
pythonw.exe	6/27/2018 2:50 AM	Application	95 KB
pythonw.pdb	6/27/2018 2:50 AM	Program Debug D...	420 KB
vcruntime140.dll	6/9/2016 10:46 PM	Application extens...	82 KB

Connecting Python Libraries in Visual Studio

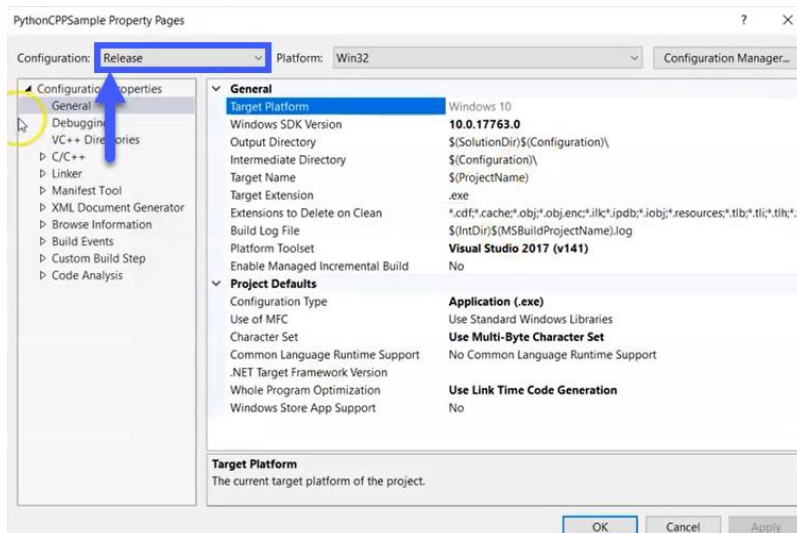
1. Now it is time to connect the necessary Python files to your Visual Studio project. Begin by opening Visual Studio and creating a new empty C++ project in Visual Studio, which you learned how to do in the previous module. Remember this includes adding a new C++ file to the source files of the project so you can begin writing code. Once you complete this, you should have an empty C++ Visual Studio project file that is ready to have Python added to it. Your Solution Explorer window should also be open from the setup work you completed thus far.



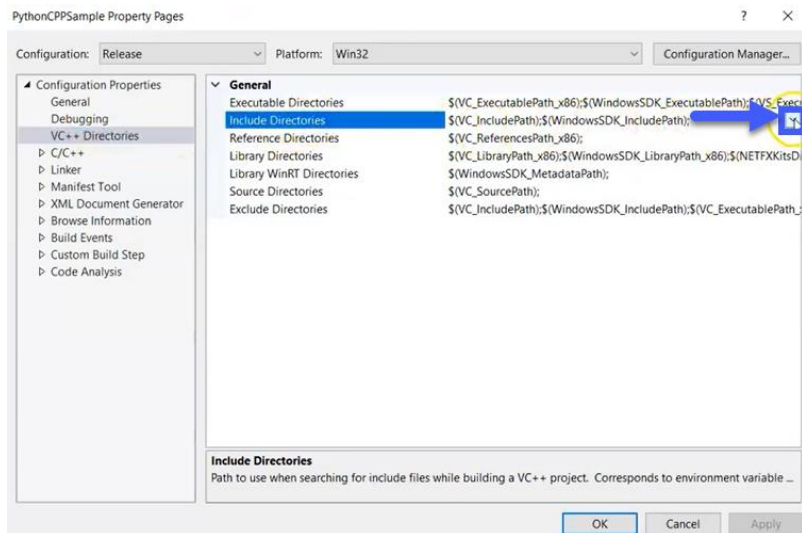
2. From the Solution Explorer window, right-click on the project's title. Then select **Properties** from the menu that pops up.



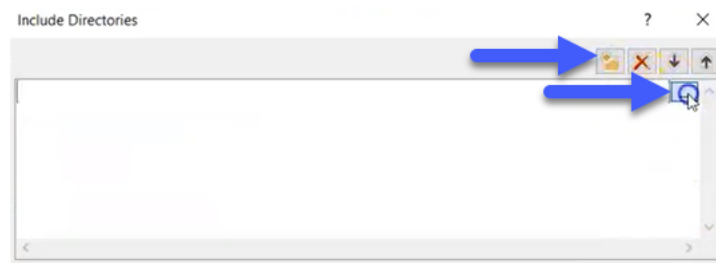
- This will open the Property Pages window. Before you start adjusting your settings, make sure you set your project to Release mode by selecting **Release** from the Configuration drop-down menu. Any Visual Studio project you create that will be working with both C++ and Python must be completed in Release mode, while any Visual Studio work that is done only in C++ should remain in Debug mode instead.



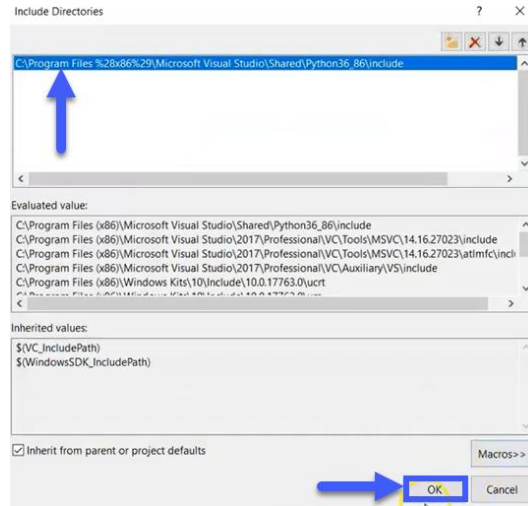
- From the menu on the left side of the screen, under Configuration Properties, select **VC++ Directories**. From the new options that open on the right, locate "Include Directories." Click the drop-down arrow next to Include Directories, and then click **Edit**.



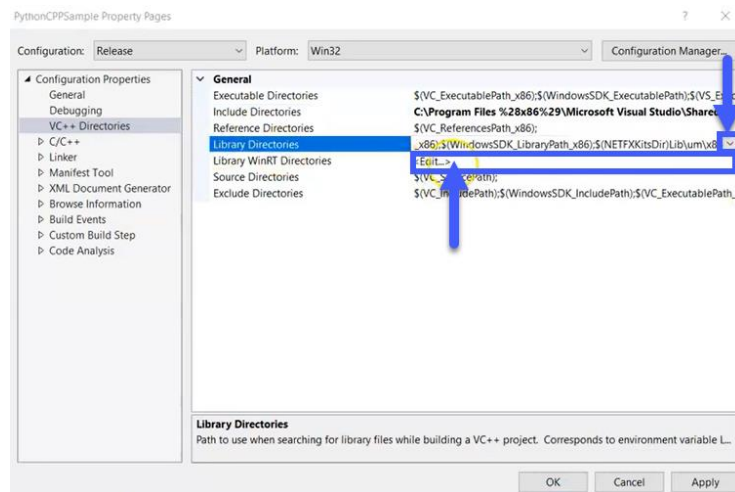
5. From the Include Directories window that opens, click on the folder icon. Next, click on the "... " icon that appears in the currently empty box from the following image.



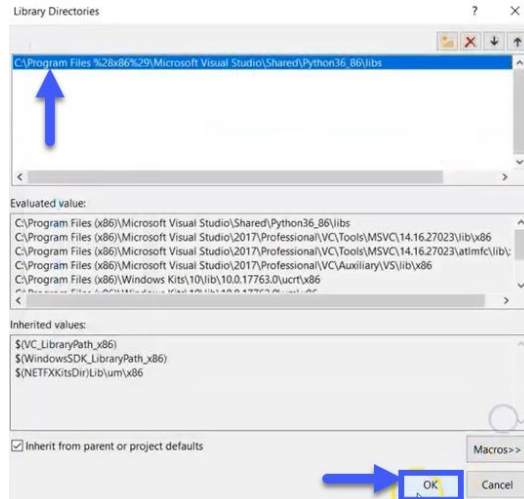
6. This will open a Documents window. Navigate to the Python folder you downloaded, and then open the Python36_86 folder. From here, select the "include" folder and click **OK** to add it. The path to that folder will now be displayed in the Include Directories window. Note that this is the only folder you need to add here. Once you are done, click **OK** in the Include Directories window.



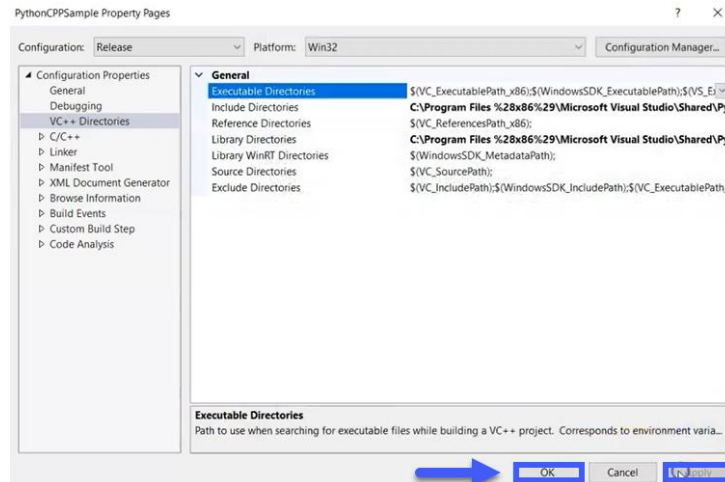
7. From the Property Pages window, under VC++ Directories like before, locate "Library Directories." Click the drop-down arrow next to Library Directories, and then select **Edit**.



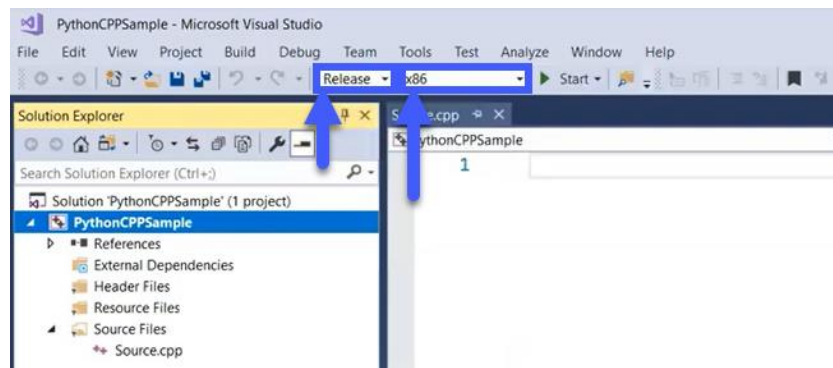
8. This will open a Documents window. Navigate to the Python folder you downloaded, and then open the Python36_86 folder. From here, select the "libs" folder, and click **OK** to add it. The path to that folder will now be displayed in the Library Directories window. Note that this is the only folder you need to add here. Once you are done, click **OK** in the Library Directories window.



9. This will return you once again to the Property Pages window. Click **Apply**, and then click **OK**.



10. Now you have a blank space for code that is set up with the Python libraries. In the top menu bar, make sure you change from "Debug" mode to "Release" mode here as well. Note that your settings should also be for "x86" if they are not automatically, since we are working with 32-bit content.



Organizing Python Files in Visual Studio

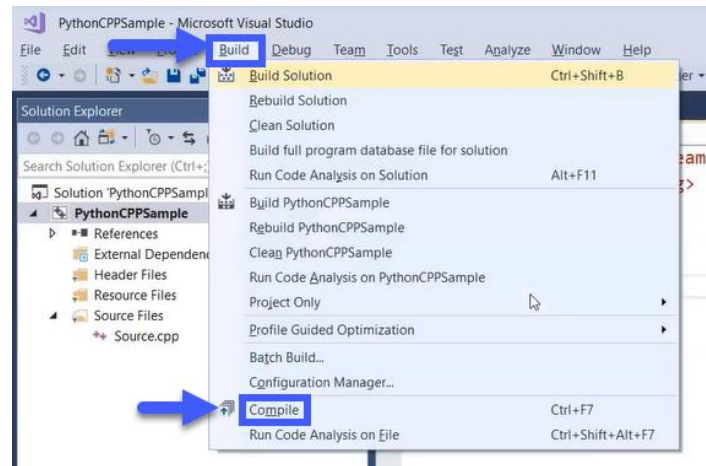
11. It is time to write a short bit of code. It does not need to do anything, but we just need to start writing it. The act of beginning to code will automatically create some project folders that you will need to manipulate in a moment. In the code box, write the following:

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

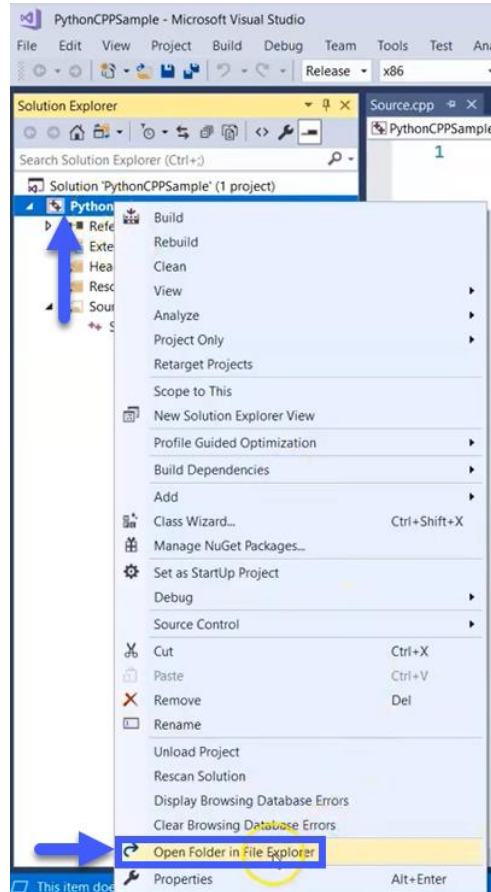
void main()
{

}
```

12. Build and compile what you have written from the Build menu at the top of the screen.



13. Once that is complete, right-click on the project's title from the Solution Explorer menu. Then select **Open Folder in File Explorer**.



14. In the Documents folder that opens, you will notice that a Release folder is included. Double-click on the **Release** folder to open it.

Name	Date modified	Type	Size
.vs	10/2/2020 6:23 AM	File folder	
Release	10/2/2020 6:26 AM	File folder	
PythonCPPSample.sln	10/2/2020 6:23 AM	Visual Studio Solut...	2 KB
PythonCPPSample.vcxproj	10/2/2020 6:26 AM	VC++ Project	7 KB
PythonCPPSample.vcxproj.filters	10/2/2020 6:26 AM	VC++ Project Filte...	1 KB
PythonCPPSample.vcxproj.user	10/2/2020 6:23 AM	Per-User Project O...	1 KB
Source.cpp	10/2/2020 6:26 AM	CPP File	1 KB

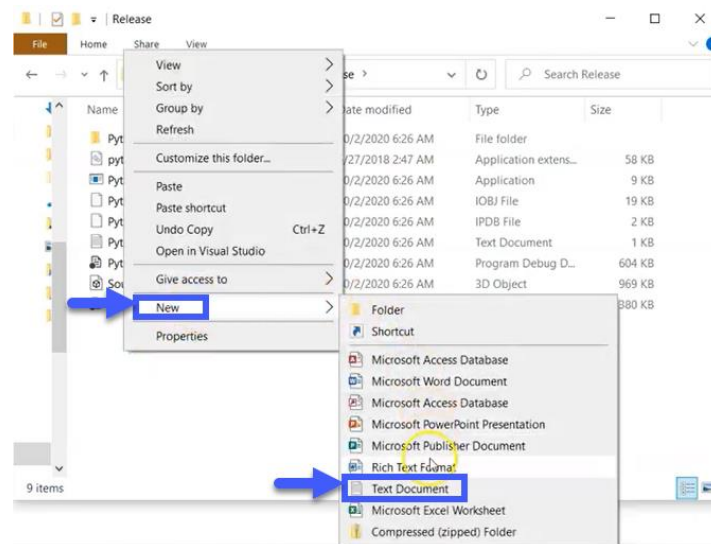
15. Before proceeding, take a moment to observe what is included in this folder. Note that it contains the project's executable (.exe) file. Now, we are going to need to place the Python DLL file within this folder; otherwise it will not be able to link to our Python libraries.

Name	Date modified	Type	Size
PythonCPPSample.tlog	10/2/2020 6:26 AM	File folder	
PythonCPPSample.exe	10/2/2020 6:26 AM	Application	9 KB
PythonCPPSample.iobj	10/2/2020 6:26 AM	IOBJ File	19 KB
PythonCPPSample.ipdb	10/2/2020 6:26 AM	IPDB File	2 KB
PythonCPPSample.log	10/2/2020 6:26 AM	Text Document	1 KB
PythonCPPSample.pdb	10/2/2020 6:26 AM	Program Debug D...	604 KB
Source.obj	10/2/2020 6:26 AM	3D Object	969 KB
vc141.pdb	10/2/2020 6:26 AM	Program Debug D...	380 KB

16. The Python DLL you will require is located in the Python folder you downloaded. Remember we are using the Python36_86 subfolder. From the Python36_86 folder, copy both "python3.dll" and "python36.dll" into the Release folder.

Name	Date modified	Type	Size
DLLs	9/7/2020 5:33 PM	File folder	
Doc	9/7/2020 5:33 PM	File folder	
include	9/7/2020 5:33 PM	File folder	
Lib	9/7/2020 5:33 PM	File folder	
libs	9/7/2020 5:33 PM	File folder	
Scripts	9/7/2020 5:33 PM	File folder	
tcl	9/7/2020 5:33 PM	File folder	
Tools	9/7/2020 5:33 PM	File folder	
LICENSE.txt	6/27/2018 2:54 AM	Text Document	30 KB
NEWS.txt	6/27/2018 2:54 AM	Text Document	395 KB
python.exe	6/27/2018 2:50 AM	Application	96 KB
python.pdb	6/27/2018 2:50 AM	Program Debug D...	412 KB
python3.dll	6/27/2018 2:47 AM	Application extens...	58 KB
python36.dll	6/27/2018 2:47 AM	Application extens...	3,225 KB
python36.pdb	6/27/2018 2:47 AM	Program Debug D...	9,390 KB
pythonw.exe	6/27/2018 2:50 AM	Application	95 KB
pythonw.pdb	6/27/2018 2:50 AM	Program Debug D...	420 KB
vcruntime140.dll	6/9/2016 10:46 PM	Application extens...	82 KB

17. Inside the Release folder, you also need to create a Python file. Right-click anywhere in the folder, then hover over **New** and select **Text Document** from the sub-menu of options that opens.

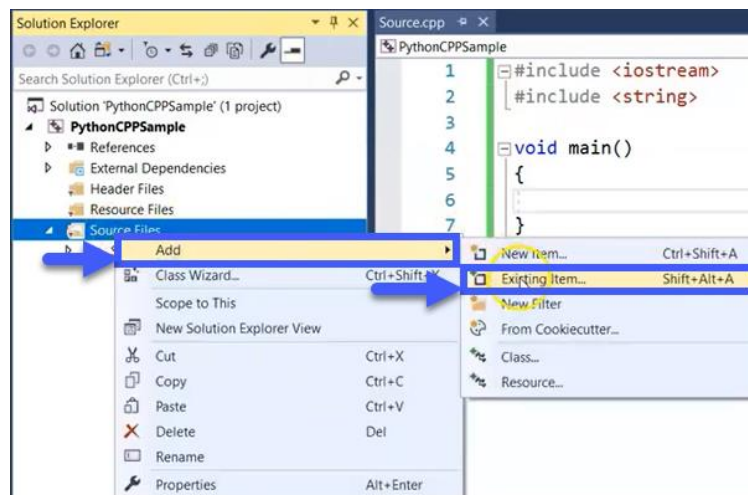


18. Rename the TXT file that is created, giving it a title you will remember and ending the file name with the ".py" tag instead of the automatic ".txt" tag. In the following example, the file is named myfirstprogram.py.

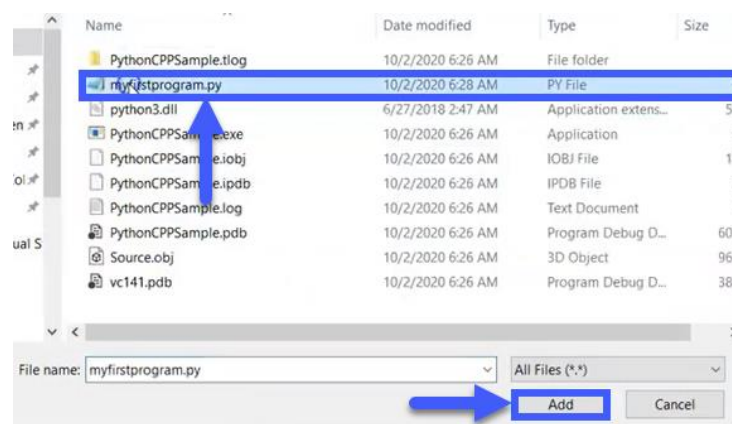
Name	Date modified	Type	Size
PythonCPPSample.tlog	10/2/2020 6:26 AM	File folder	
myfirstprogram.py	10/2/2020 6:28 AM	PY File	0 KB
python3.dll	6/27/2018 2:47 AM	Application extens...	58 KB
PythonCPPSample.exe	10/2/2020 6:26 AM	Application	9 KB
PythonCPPSample.iobj	10/2/2020 6:26 AM	IOBJ File	19 KB
PythonCPPSample.ipdb	10/2/2020 6:26 AM	IPDB File	2 KB
PythonCPPSample.log	10/2/2020 6:26 AM	Text Document	1 KB
PythonCPPSample.pdb	10/2/2020 6:26 AM	Program Debug D...	604 KB
Source.obj	10/2/2020 6:26 AM	3D Object	969 KB
vc141.pdb	10/2/2020 6:26 AM	Program Debug D...	380 KB

Adding a Python Source File in Visual Studio

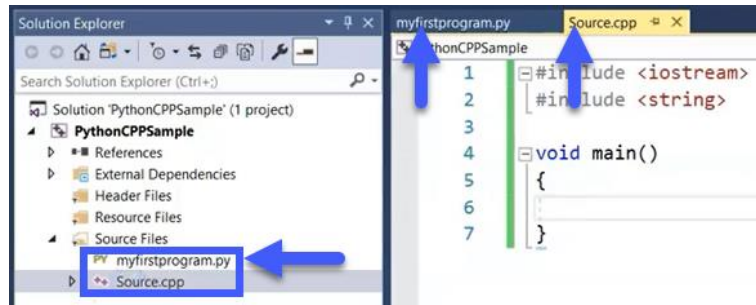
19. Now return to your Visual Studio project. Right-click on the **Source Files** from the Solution Explorer. Select **Add**, and then click **Existing Item**.



20. Navigate to the Release folder, where you just created the PY file, then select it. Click **Add** when you are ready.



21. Now in the Solution Explorer menu, you will notice both the CPP file you created previously and the new PY file you just added. At the top of the screen, there will be two tabs representing each, above where you can write your code.



22. Now it is time to link the C++ and Python files so information can be passed back and forth between the two. Paste the following C++ and Python starter code into each respective tab. Note that in the C++ code you need to call out the name of the Python file you are referring to. This code references the "myfirstprogram" example, but this should be updated to whatever name you chose. Also, make sure you only write the file name and do not include the ".py" tag.

C++ Code

```
#include <Python.h>
#include <iostream>
#include <string>

using namespace std;

void main()
{
    cout << "Start 1 \n";
    Py_Initialize();
    cout << "2\n";
    PyObject* my_module = PyImport_ImportModule("myfirstprogram");
    cerr << my_module << "\n";
    PyErr_Print();
    cout << "3\n";
    PyObject* my_function = PyObject_GetAttrString(my_module,
        "printsomething");
    cout << "4\n";
    PyObject* my_result = PyObject_CallObject(my_function, NULL);
    Py_Finalize();
}
```

Python Code

```
import re
import string
```



```
def printsomething():  
    print("Hello from Python!")
```

23. Now run the code to check if your setup was successful. You should receive the following results, though note that your third line will look different because it will display a number that corresponds to the location of your Python file on your own machine.

```
Start 1  
2  
01592AE0  
3  
4  
Hello from Python!
```

24. Now you are ready to start manipulating your own code!