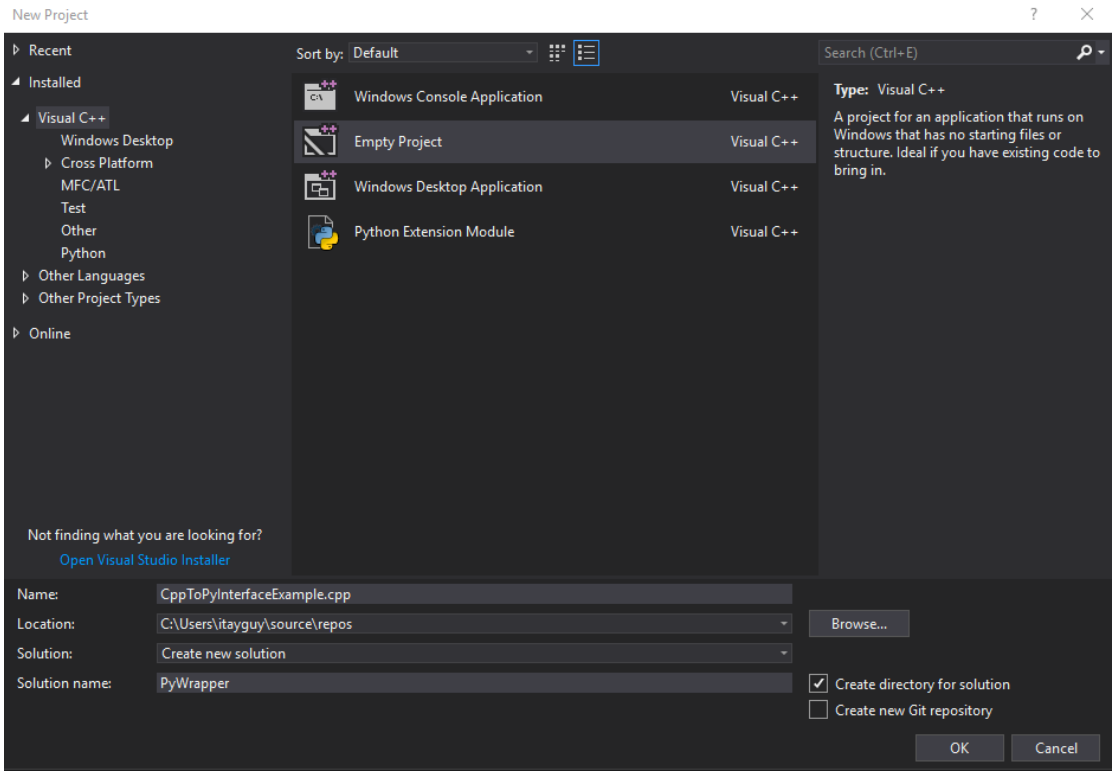
CPP Core Using Python Interface:

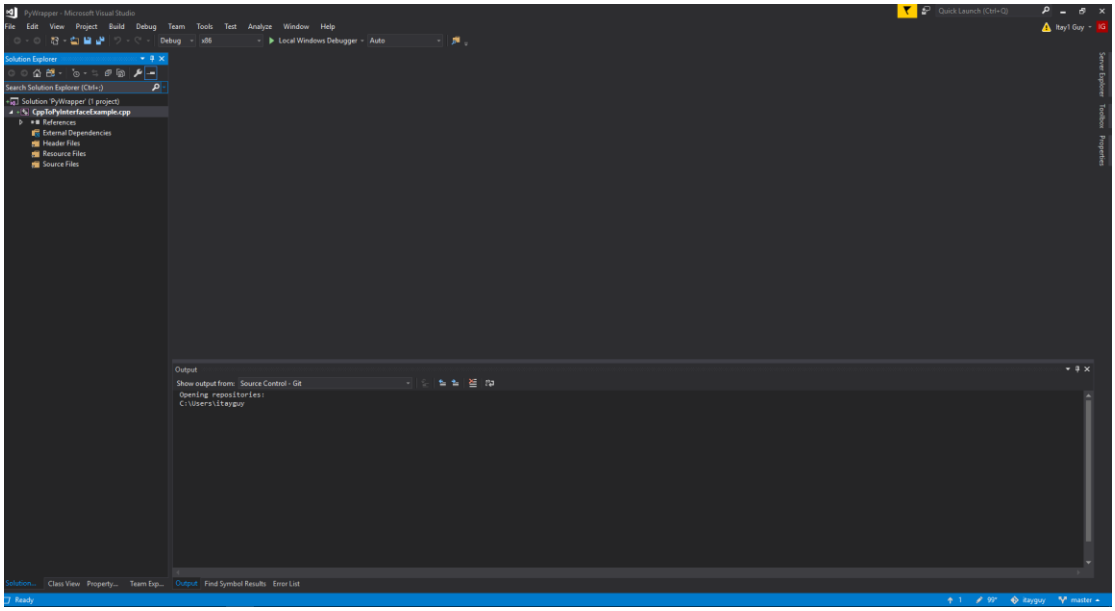
Preliminaries:

- You can generalize the next Example

- ***Installations:***
  - *Python Workspace:*
    - Python interface version (3.7.4)
      - *Installation location:*
        - *C:\Users\itayguy\AppData\Local\Programs\Python\Python37*
  - *CPP Workspace:* VS2017 Professional 15.9.4
  - *Packages:*
    - Desktop development with C++
    - Python development
      - Python native development tools
- Set Environment Variable:
  - *SET PYTHONHOME= C:\Users\itayguy\AppData\Local\Programs\Python\Python37*
- Open VS
- *File → New → Project…*
- *Select Visual C++ → Empty Project*
- *At the bottom:*
  - *Set Name:*
    - *CppToPyInterfaceExample.cpp*
  - *Set Solution name:*
    - *PyWrapper*
- *Select Ok*



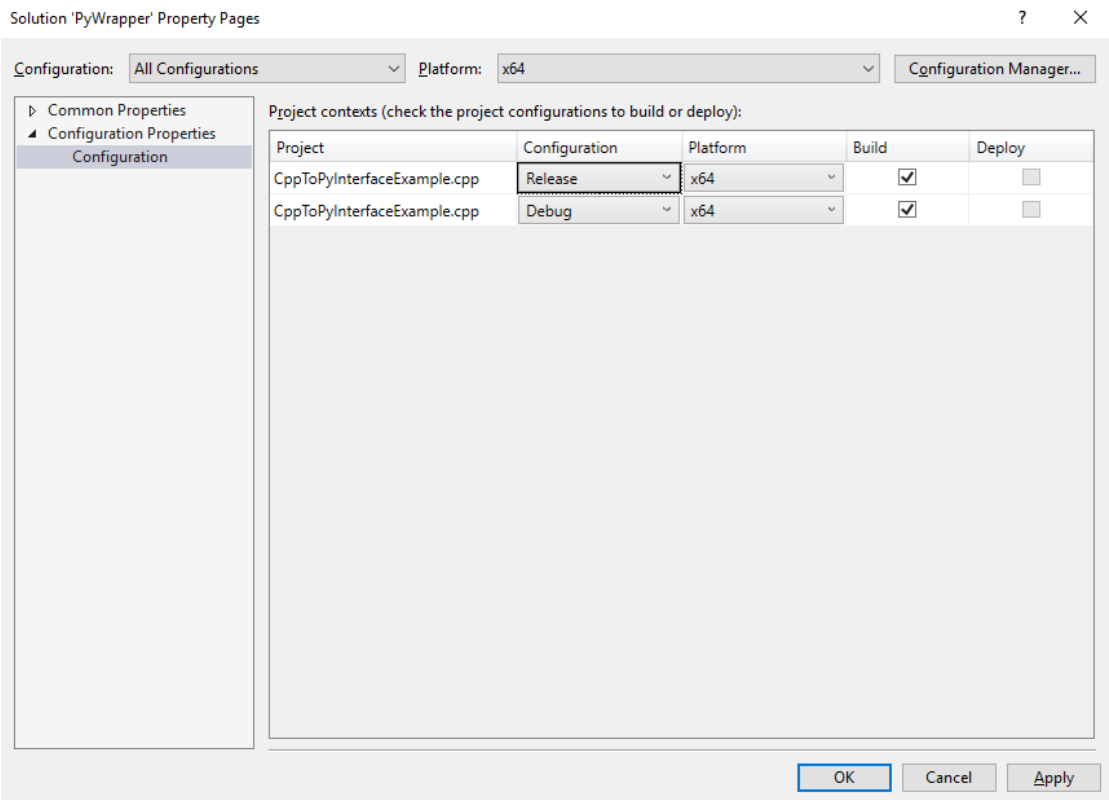
- *Waiting while the project is loading…*
- *You can see the next front:*



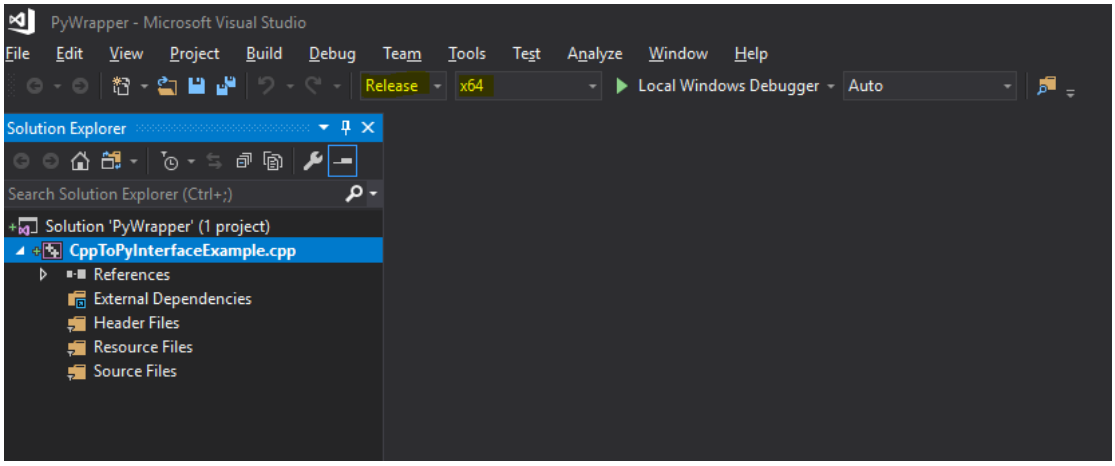
VS Environment Configurations:

- *Left-Click on the Solution → Properties*
- *Select Configuration Properties*
- *Change to Platform $x64$:*

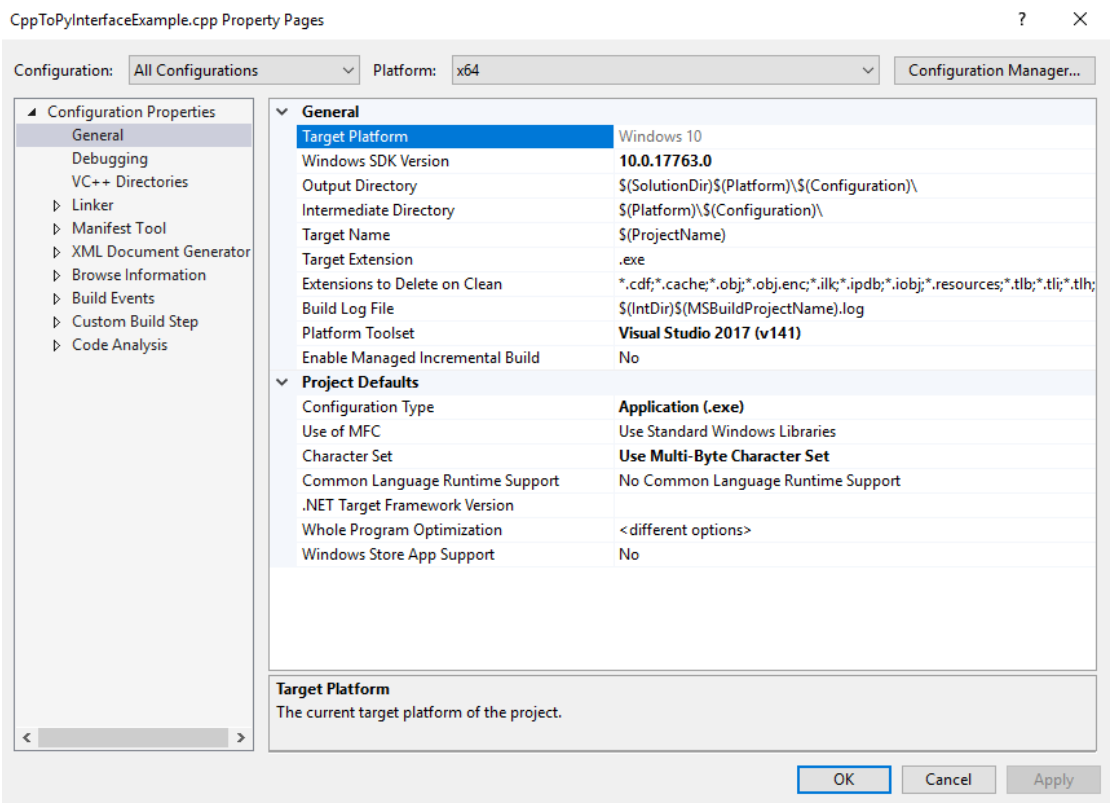
- *In project*
- *In configuration manager*
- *Be aware there is Release & Debug modes*



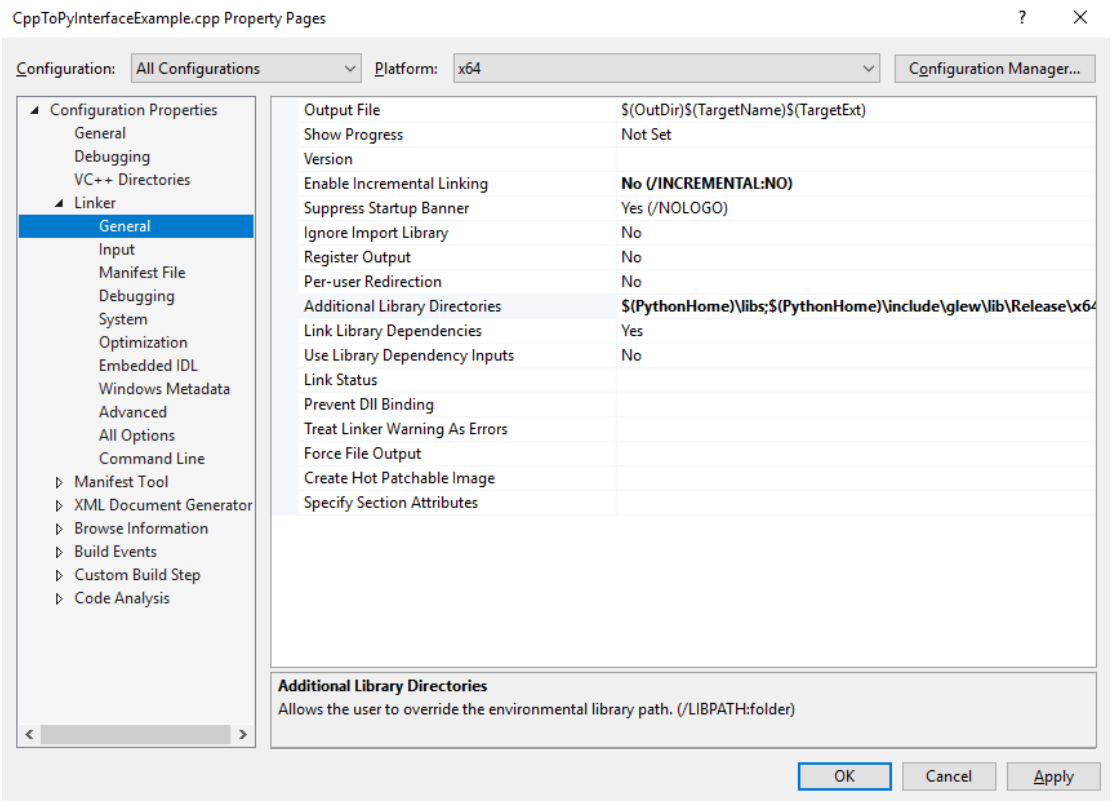
- *Select Apply → Ok*
- *Change the compile to work in Release & $x64$*



- *Left-Click on CppToPyInterfaceExample.cpp → Properties*



- *Configuration Properties*
  - *General*
    - *Target Extension → .pyd*
    - *Configuration Type → Dynamic Library (.dll)*
    - *Character Set → Not Set*
    - *Select Apply*
  - *Linker*
    - *General → Enable Incremental Linking → No*
    - *General → Additional Library Directories:*
      - *$(PythonHome)\libs*
      - *$(PythonHome)\include\glew\lib\Release \x64*
      - *$(PythonHome)\include\glfw\lib-vc2017*
    - *Select Ok*



    - *Input → Additional Dependencies:*
        - *glew32.lib*
        - *glu32.lib*
        - *glfw3.lib*
        - *glfw3dll.lib*
        - *opengl32.lib*
- *Add new Item to Source Files: CppToPy.cpp*
- *Go back to the project properties page*
- *C/C++*
    - *General → Additional Include Directories:*
        - *$(PythonHome)\include*
        - *$(PythonHome)\include\data*
        - *$(PythonHome)\include\libigl\include*
        - *$(PythonHome)\include\external\glad\include*
        - *$(PythonHome)\include\external\eigen*
        - *$(PythonHome)\include\glew\include\GL*
        - *$(PythonHome)\include\glfw\include*

        - *$(PythonHome)\include\external\pybind11\include*
    - *Select Ok → Apply*
    - *Preprocessor*
        - *Preprocessor Definitions*
            - *Add: WIN32*
    - *Select Ok → Apply → Ok*
- Download the external libreries:
    - *Libigl*
    - *Glew*
    - *Glfw*
    - *external (eigen, …)*
        - *you can reach from libigl build*
- *Locate the outer libraries at this order into:*
    - *PYTHONHOME\\include*
- *Code the next CPP code:*

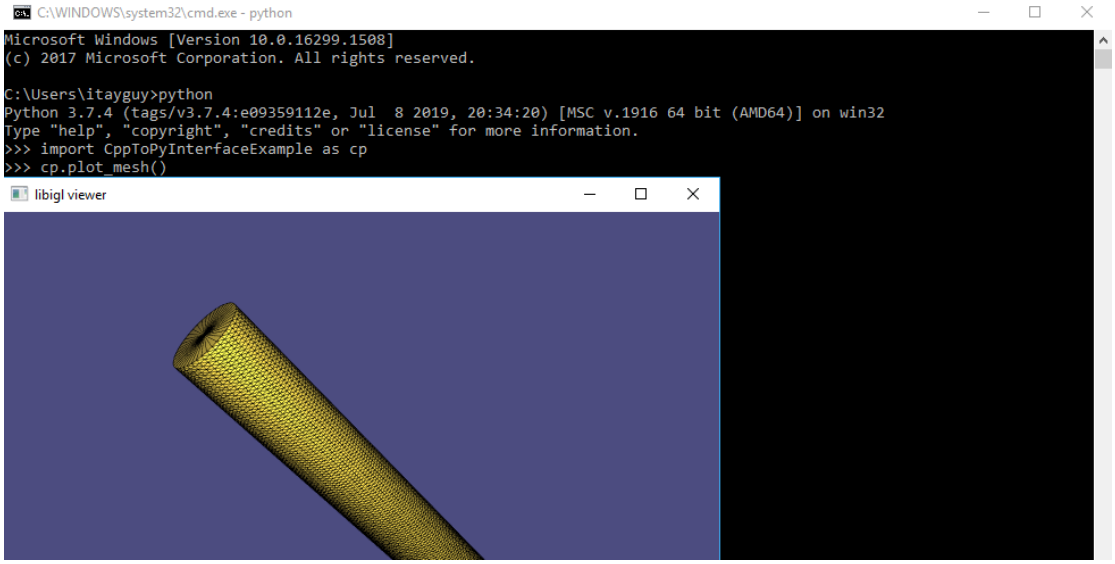
CppToPy.cpp

CppToPyInterfaceExample.cpp

```cpp
#include <pybind11/pybind11.h>

/*
.   Names:      Dr. Roi Porrane & Mr. Itay Guy
.   Purpose:    All examples i propose here have taken from libigl.
.   Date:       30/11/2019
*/

// External includes
#include <igl/readOFF.h>
#include <igl/opengl/glfw/Viewer.h>
#include <external/glad/src/glad.c>

// Internal includes
#include <iostream>
#include <cstdlib>

// Constants
#define PYENV "PYTHONHOME"
#define MODEL "cylinder.off"

// Macros
#define GET_PYENV std::getenv(std::string(PYENV).c_str())\

// Global variables
Eigen::MatrixXd V;
Eigen::MatrixXi F;

void plot_mesh() {

    std::string env_p(GET_PYENV);
    if (env_p.empty()) {
        std::cout << "Environment Variable %" << PYENV << "%: not exist." << std::endl;
        return;
    }

    // Load a mesh in OFF format
    igl::readOFF(env_p + "\\include\\data\\" + MODEL, V, F);

    // Plot the mesh
    igl::opengl::glfw::Viewer viewer;
    viewer.data().set_mesh(V, F);
    viewer.launch();
}

namespace py = pybind11;

PYBIND11_MODULE(CppToPyInterfaceExample, m) {
    m.def("plot_mesh", &plot_mesh, R"pbdoc(
        Plotting a mesh.
    )pbdoc");

#ifdef VERSION_INFO
    m.attr("__version__") = VERSION_INFO;
#else
    m.attr("__version__") = "dev";
#endif
}
```

- *Press → Build → Build Solution*
- *Copy the file at:*
    - *C:\Users\itayguy\source\repos\PyWrapper\x64\Release\ CppToPyInterfaceExample.pyd to any Python 3.7.4 interpreter DLLs and start using the dynamic library*

- *Example:*


C:\WINDOWS\system32\cmd.exe - python
Microsoft Windows [Version 10.0.16299.1508]
(c) 2017 Microsoft Corporation. All rights reserved.
C:\Users\itayguy>python
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019, 20:34:20) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import CppToPyInterfaceExample as cp
>>> cp.plot_mesh()
libigl viewer