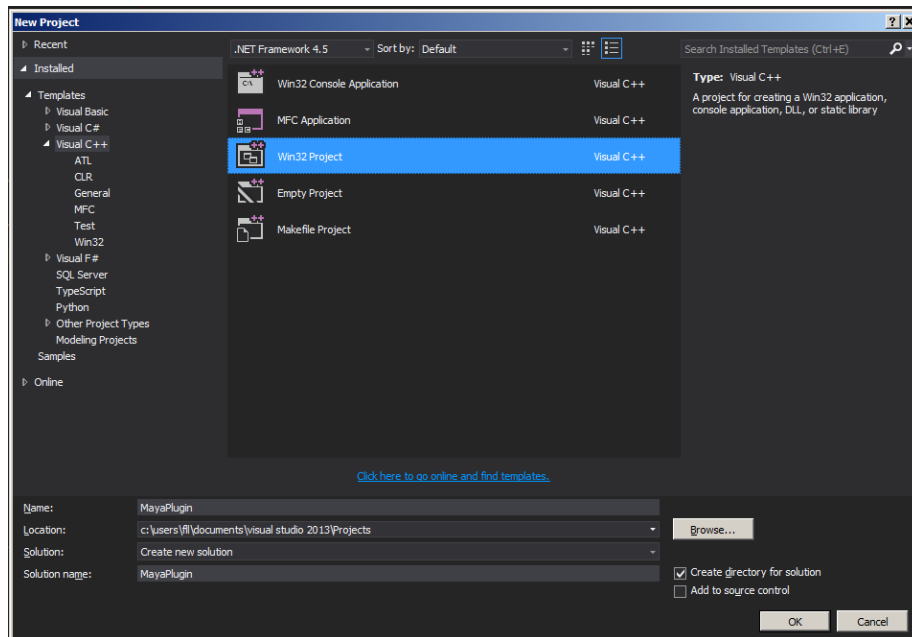
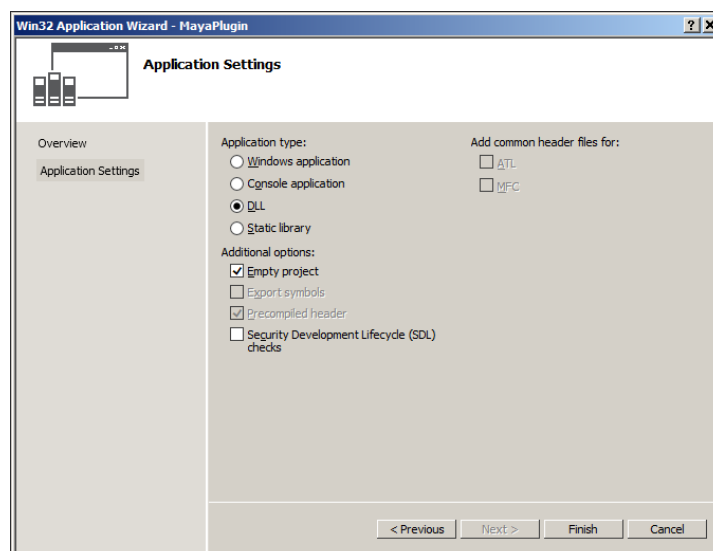


Project setup Visual Studio + Maya Plugin

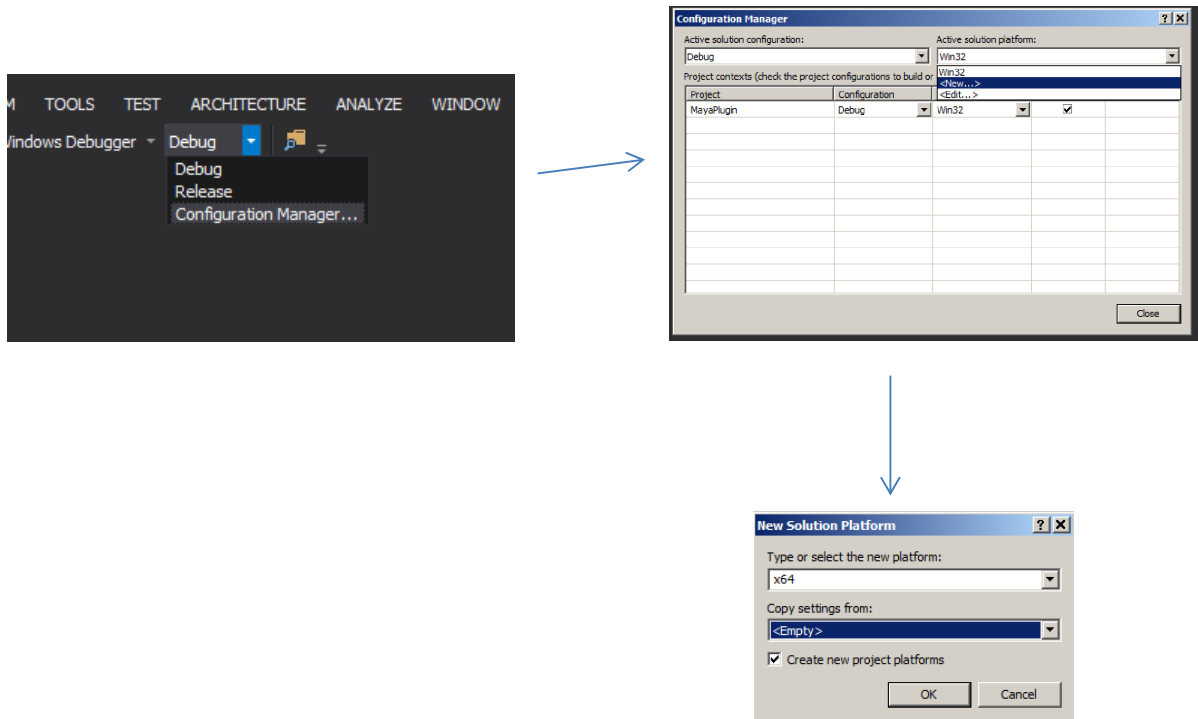
Create a new Win32 Project in Visual Studio, give it any name you want and click OK.



For the project settings, select DLL and Empty Project options, as shown in the image. Then click on Finish.

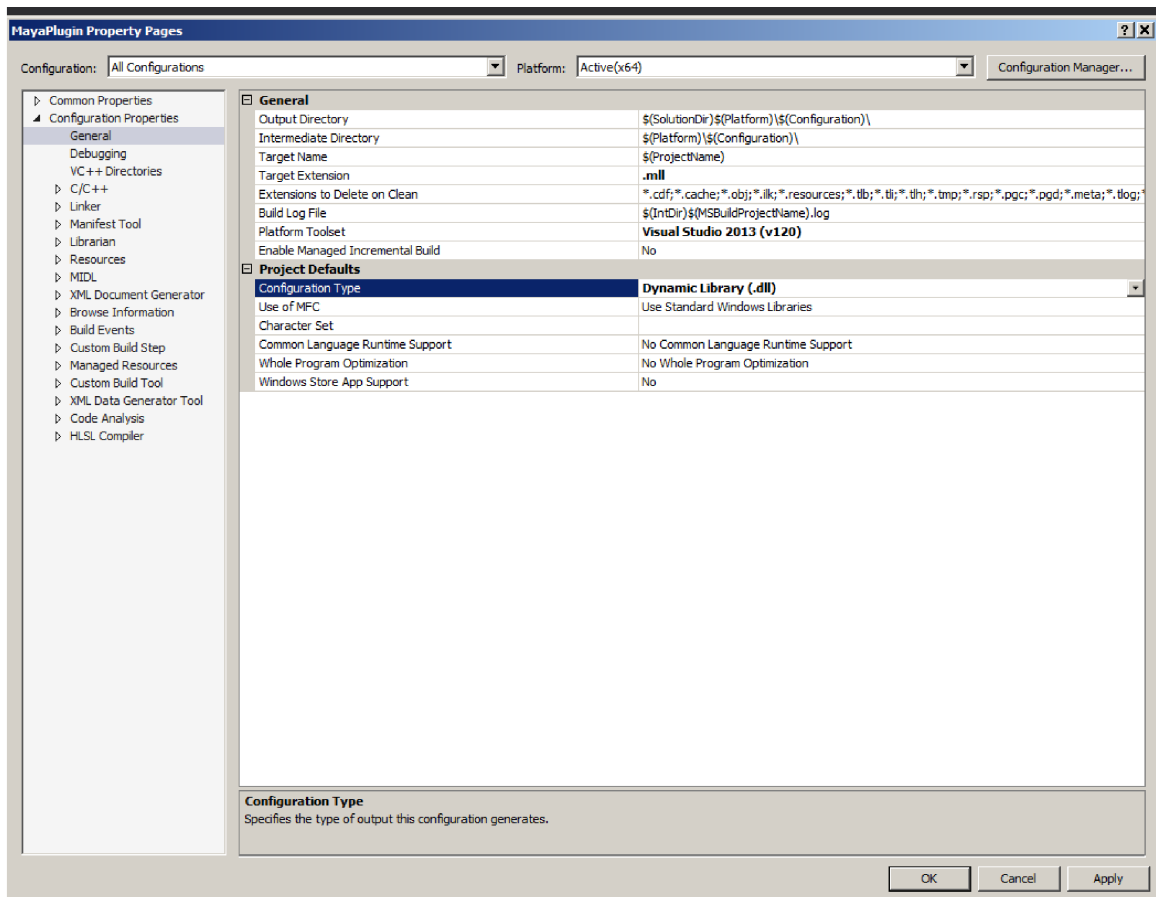


Once the project has been created, click on the Configuration Manager option, and create a new project platform using x64 (64 bits architecture)

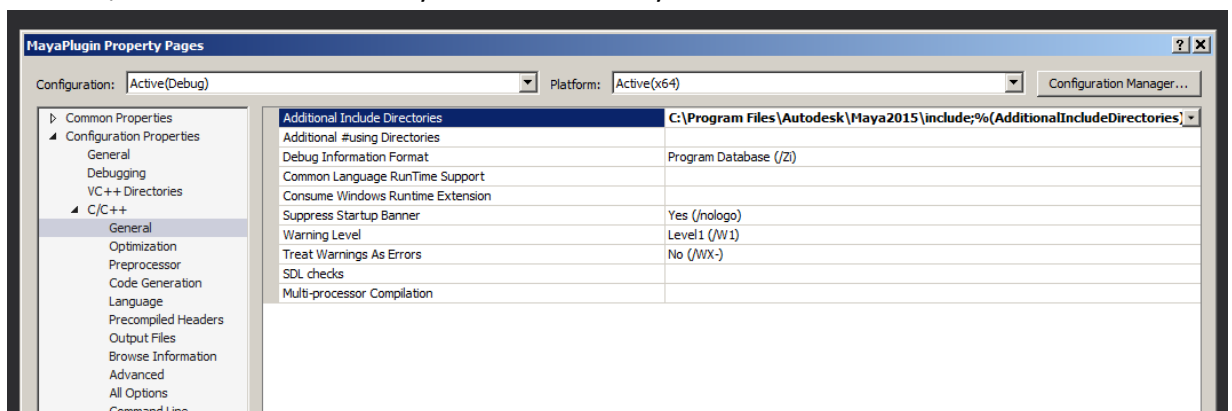


Next, go to properties of the Project, to the General configuration sheet, and change:

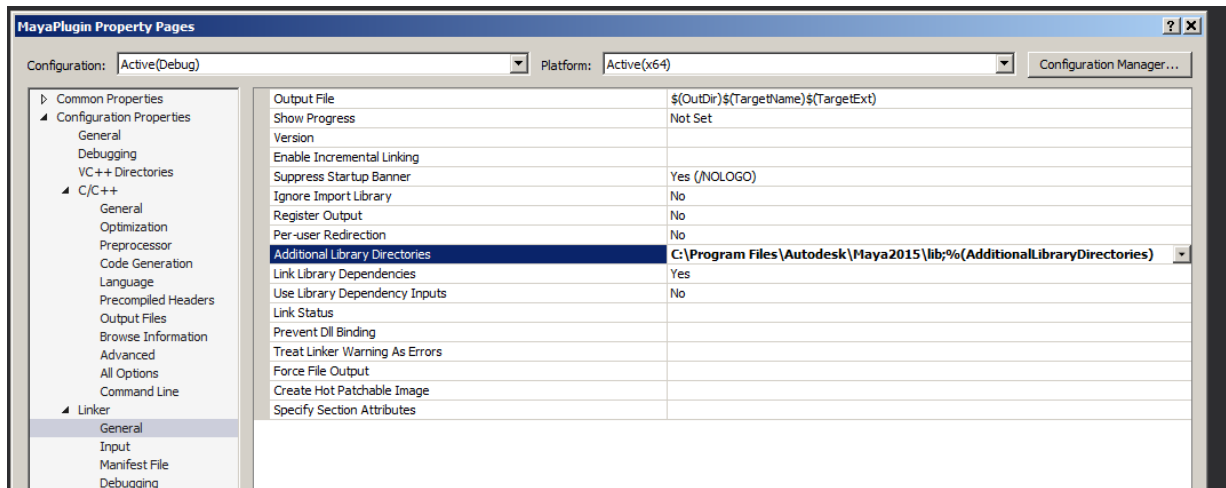
- Target Extension to “ml” (it’s just an extension for our library)
- Configuration Type to “dll” to make this a Dynamic Library.



Under C/C++ → General add Maya include directory to have access to the header files.

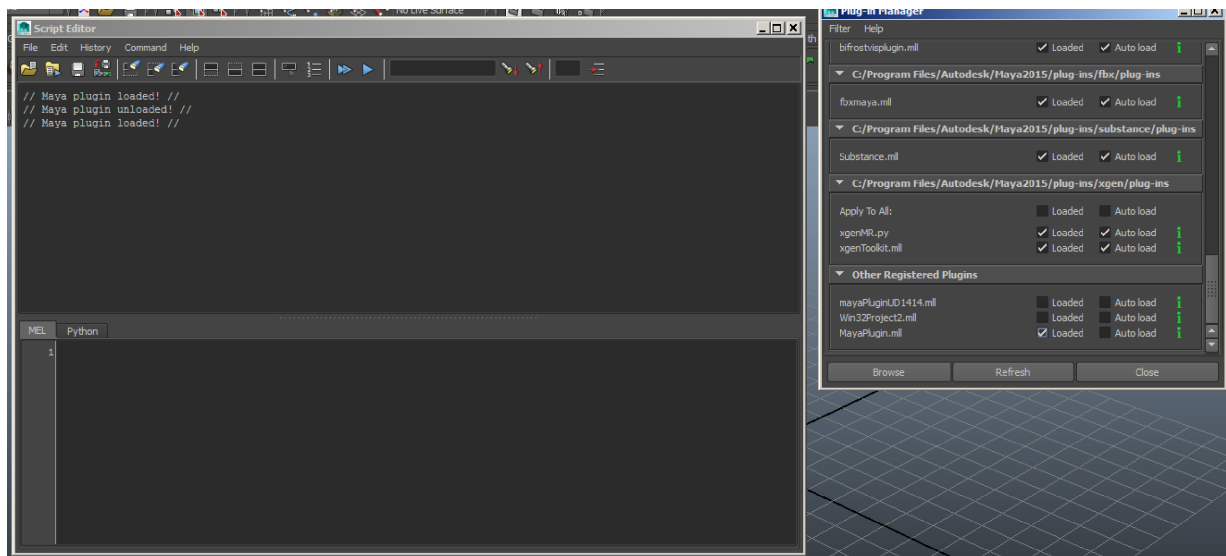


In the Linker configuration, under General, add Maya's lib directory to be able to link against Maya's Libraries.

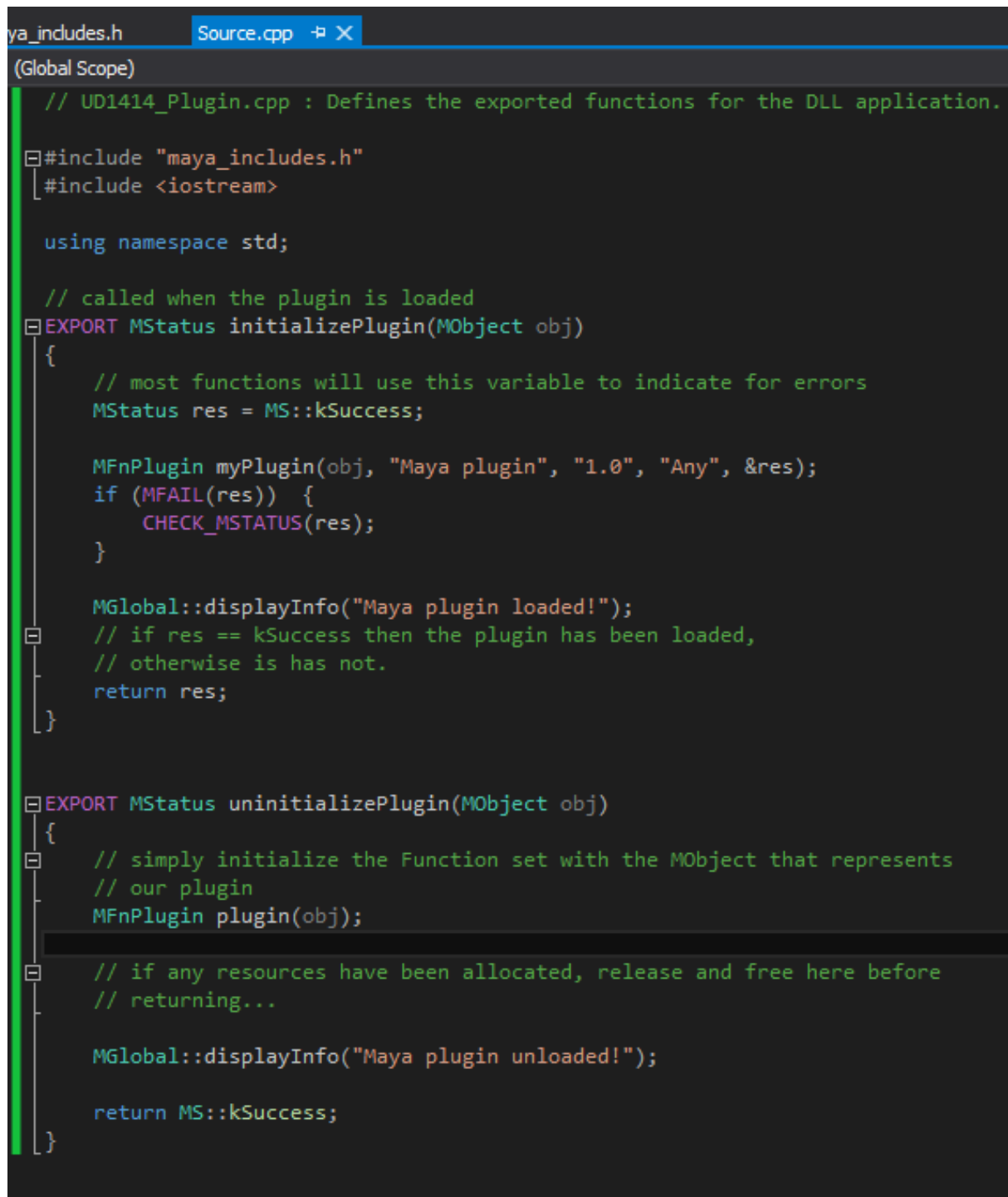


Next, add a blank .cpp file to your project to write the plugin code, and .h file with some handy Maya includes.

After compiling, you should obtain a file with .ml extension which can be loaded by Maya's plugin manager as shown in the following image.



Minimum content for the CPP file, type in the code in Visual Studio



The image shows a Visual Studio code editor window with two tabs: 'maya_includes.h' and 'Source.cpp'. The 'Source.cpp' tab is active, showing the following C++ code:

```
(Global Scope)

// UD1414_Plugin.cpp : Defines the exported functions for the DLL application.

#include "maya_includes.h"
#include <iostream>

using namespace std;

// called when the plugin is loaded
EXPORT MStatus initializePlugin(MObject obj)
{
    // most functions will use this variable to indicate for errors
    MStatus res = MS::kSuccess;

    MFnPlugin myPlugin(obj, "Maya plugin", "1.0", "Any", &res);
    if (MFAIL(res)) {
        CHECK_MSTATUS(res);
    }

    MGlobal::displayInfo("Maya plugin loaded!");
    // if res == kSuccess then the plugin has been loaded,
    // otherwise it has not.
    return res;
}

EXPORT MStatus uninitializePlugin(MObject obj)
{
    // simply initialize the Function set with the MObject that represents
    // our plugin
    MFnPlugin plugin(obj);

    // if any resources have been allocated, release and free here before
    // returning...

    MGlobal::displayInfo("Maya plugin unloaded!");

    return MS::kSuccess;
}
```

Check for maya_includes.h here: <http://tinyurl.com/qee8q3v>