How to Download VTK 7.1.1

Shubham Gupta

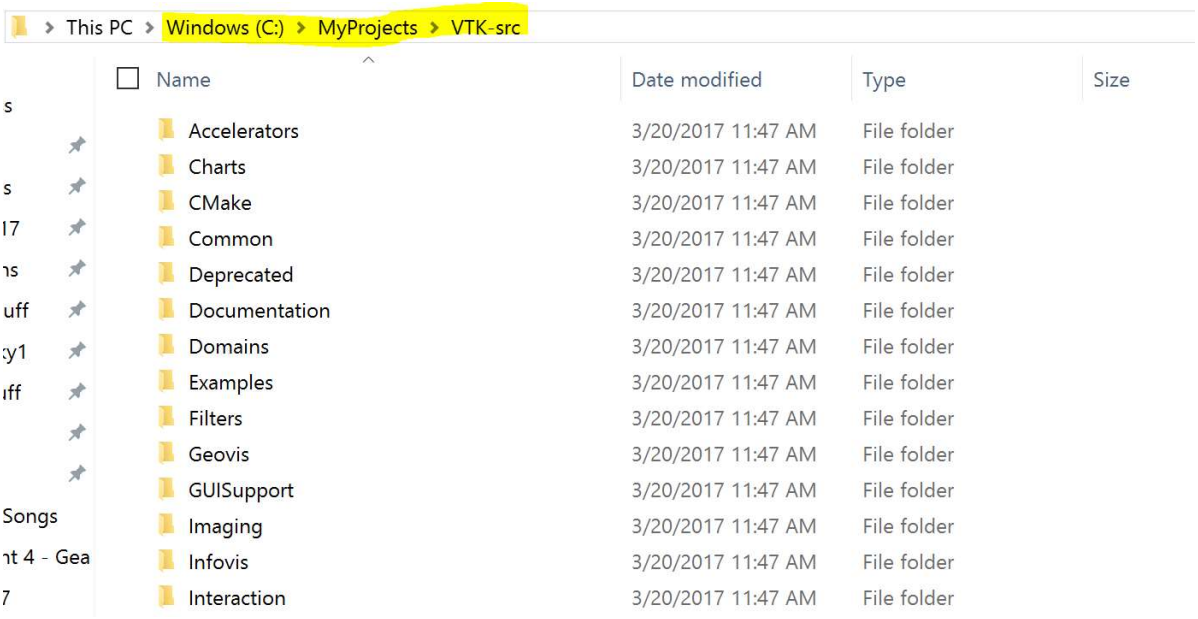
Step 1 – Download VTK

Go to http://www.vtk.org/download/ and scroll down to download the source file (.zip or .tar.gz)

Latest Release (7.1.1)

Platform	Files
Source	VTK-7.1.1.zip
	VTK-7.1.1.tar.gz
Standalone Python Interface (Installer)	vtkpython-7.1.1-Windows-64bit.exe
	vtkpython-7.1.1-Darwin-64bit.dmg
	vtkpython-7.1.1-Linux-64bit.tar.gz
Data	VTKData-7.1.1.zip
	VTKData-7.1.1.tar.gz
	VTKLargeData-7.1.1.zip
	VTKLargeData-7.1.1.tar.gz
Documentation	vtkDocHtml-7.1.1.tar.gz

Extract the compressed folder (using 7-Zip, WinZip, etc.) into a new folder called 'VTK-src'. (Ex. C:/MyProjects/VTK-src). Try not the make the file path too long. **NOTE:** If you extract to a different chosen path, you will need to point to that when asked to point to the VTK source folder.



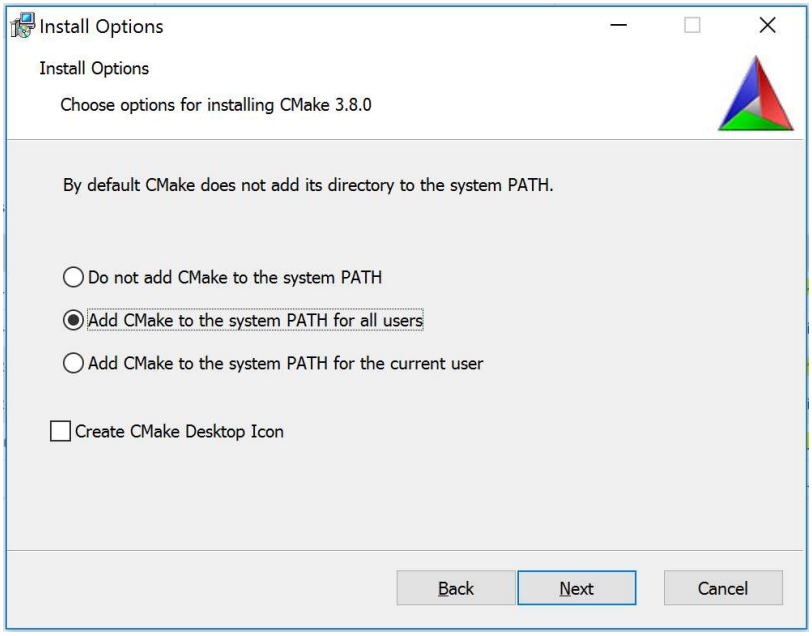
Step 2 – Download CMake 3.8

Go to https://cmake.org/download/ and download/run the installer (recommended unless you would prefer to install it manually). You may have to uninstall previous versions unless you want to use that instead. **NOTE:** Newer versions may have been released after the creation of this manual. Use the "Latest Release" version rather than the "Release Candidate".

Binary distributions:

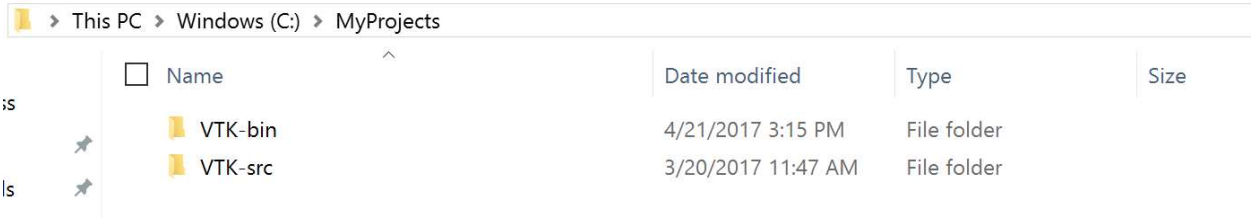
Platform	Files
Windows win64-x64 Installer: **Installer tool has changed. Uninstall CMake 3.4 or lower first!**	cmake-3.8.0-win64-x64.msi
Windows win64-x64 ZIP	cmake-3.8.0-win64-x64.zip
Windows win32-x86 Installer: **Installer tool has changed. Uninstall CMake 3.4 or lower first!**	cmake-3.8.0-win32-x86.msi
Windows win32-x86 ZIP	cmake-3.8.0-win32-x86.zip
Mac OSX 10.6 or later	cmake-3.8.0-Darwin-x86_64.dmg
	cmake-3.8.0-Darwin-x86_64.tar.gz

Make sure to choose 'Add CMake to the system PATH for (all/current) user



Go through the installer by clicking 'Next'. You may change the install directory if you would like.

Create a build folder called 'VTK-bin' where you installed the VTK source.



Step 3 – Download Visual Studio 2017 (if not done already)

Go to https://www.visualstudio.com/downloads/ and download the Visual Studio Community executable file. Run the file.

Select the 'Universal Windows Platform Development' and the 'Desktop development with C++' workloads. Click 'Individual components' and ensure these are downloaded (you can leave components checked if they are not in the list unless you have space issues):

.NET

- .NET Framework 4.5 targeting pack
- .NET Framework 4.6.1 SDK
- .NET Framework 4.6.1 targeting pack
- .NET Native
- .NET Portable Library targeting pack

Cloud, database, and server

- CLR data types for SQL Server
- Data sources and service references

Code tools

- ClickOnce Publishing
- Developer Analytics tools
- NuGet package manager
- Static analysis tools
- Text Template Transformation

Compilers, build tools, and runtimes

- C# and Visual Basic Roslyn compilers
- C++/CLI support
- MSBuild
- VC++ 2017 v141 toolset (x86,x64)
- Visual C++ tools for CMake

Debugging and testing

- C++ profiling tools
- JavaScript diagnostics
- Just-In-Time debugger
- Profiling tools
- WebSocket4Net

Development activities

- C# and Visual Basic
- JavaScript and Type Script language support
- Visual Studio C++ core features

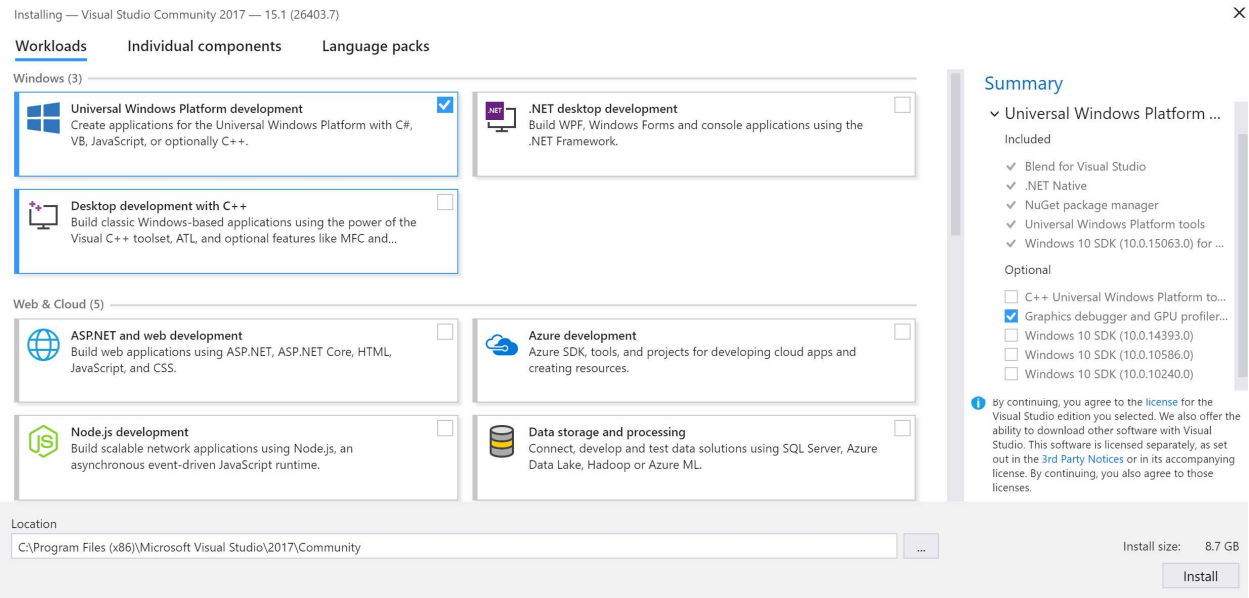
Games and Graphics

- Graphics debugger and GPU profiler for DirectX
- Image and 3D model editors

SDKs, libraries, and frameworks

- Graphics Tools Windows 8.1 SDK
- TypeScript 2.2 SDK
- Visual C++ ATL support
- Windows 10 SDK (10.0.15063.0) for Desktop C++ x86 and x64
- Windows 10 SDK (10.0.15063.0) for UWP: C#, VB, JS
- Windows 10 SDK (10.0.15063.0) for UWP: C++
- Windows 8.1 SDK

Press ‘Install’ in the bottom right.

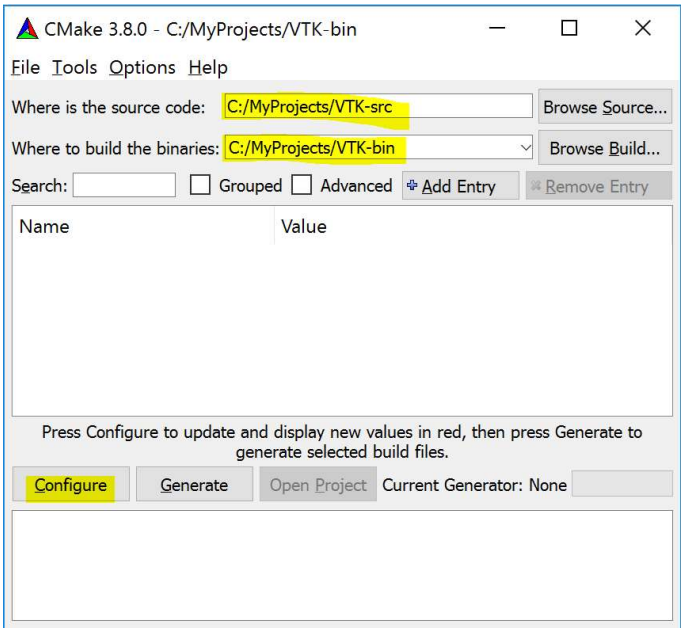


Restart the computer as requested.

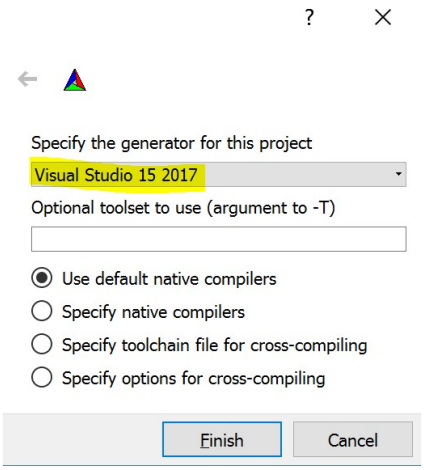
Step 4 – Run CMake

Run the CMake GUI in …/CMake/bin.

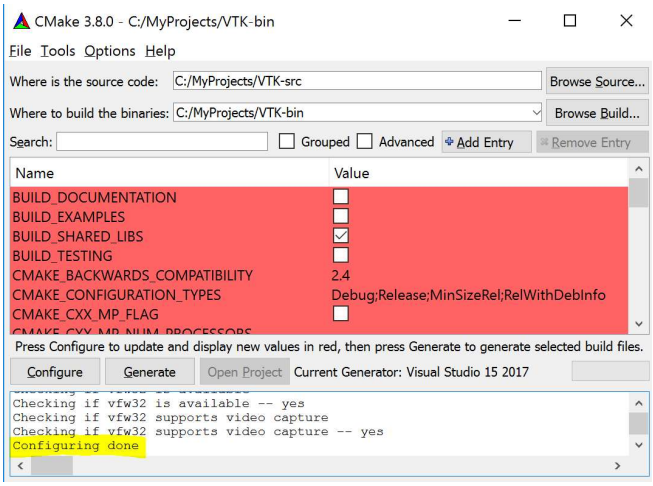
Click ‘Browse Source’ and select the VTK-src directory. Click ‘Browse Build’ and select the VTK-bin directory. Click ‘Configure’.



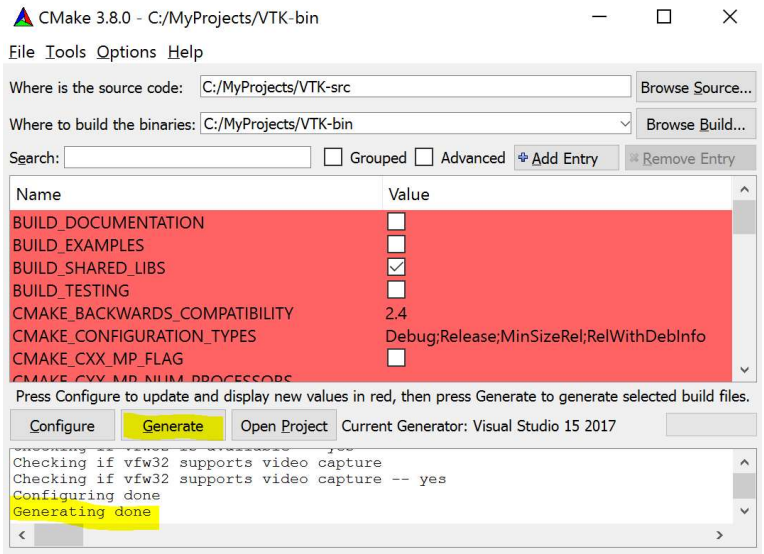
After clicking ‘Configure’, it may prompt you to ‘Specify the generator for this project’. In that case, select your version of Visual Studio (in my case it’s 2017).



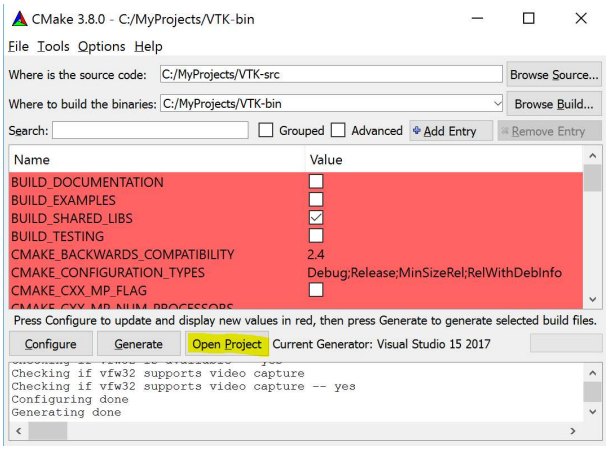
Press ‘Finish’ and press ‘Configure’ if it hasn’t started configuring already. You should see ‘Configuring done’ in the message box when it has finished (takes a while).



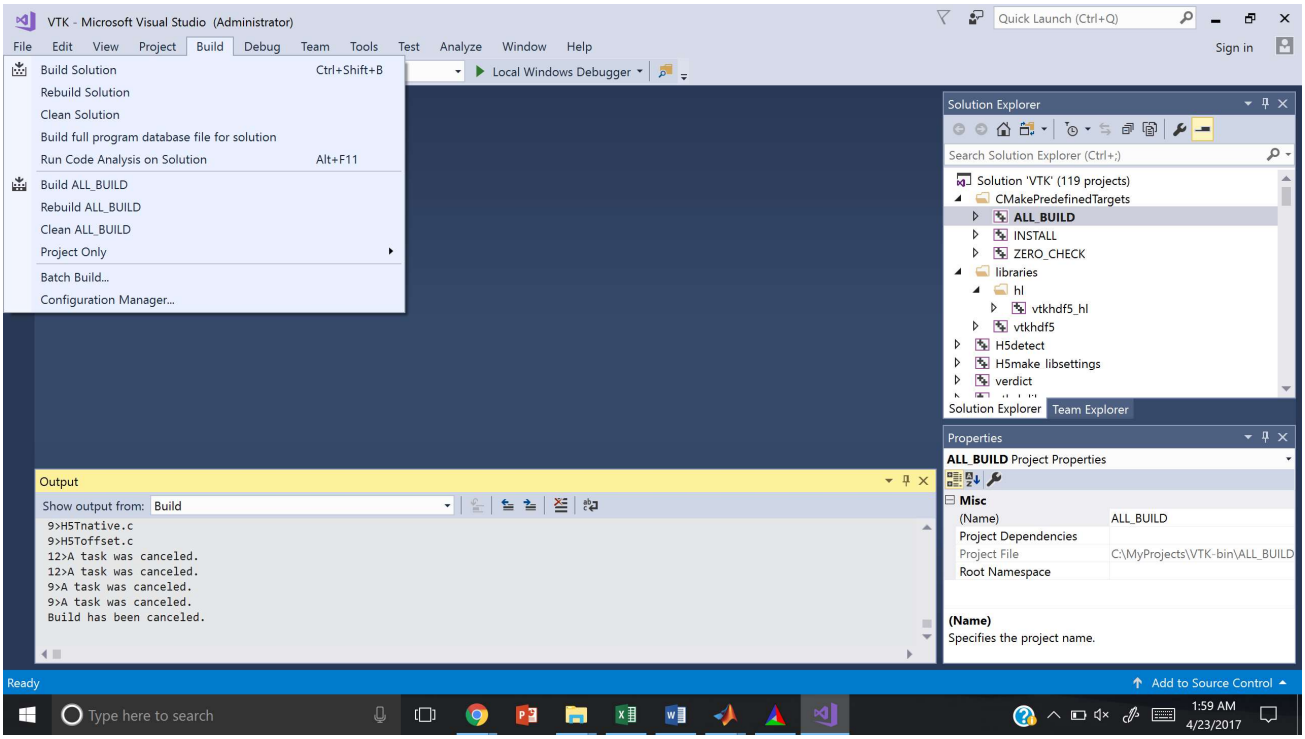
Click ‘Generate’. You should see ‘Generating done’ in the message box.



Click ‘Open Project’ and open with Visual Studio.



Wait for the program to say ‘Ready’ in the bottom left corner. Press ‘Build -> Build ALL_BUILD’ or ‘Build -> Build -> ALL_BUILD’ to build VTK. Wait for this process to finish entirely before moving to the next step.



Edit the system variable Path to include the directory to your VTK build (‘VTK-bin’) and to ‘VTK-bin/bin/Debug’.

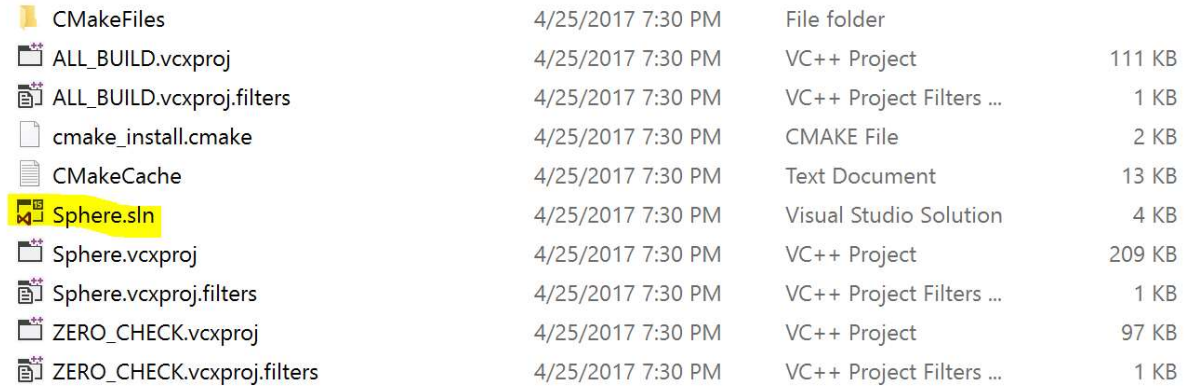
Step 4 – Test VTK

Go to http://www.vtk.org/Wiki/VTK/Examples/Cxx/GeometricObjects/Sphere. Download Sphere.tar and extract it.

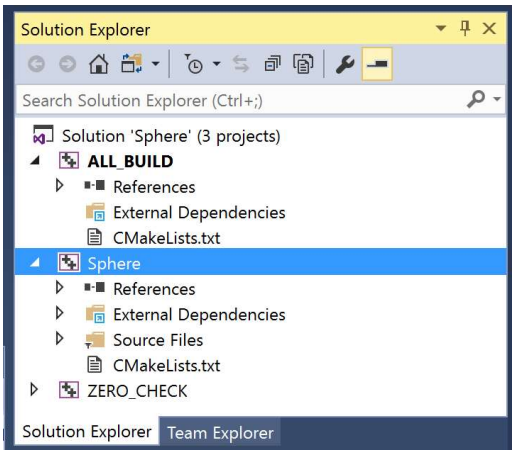
Open up Command Prompt and move to the build directory ('Sphere/build'). Type 'cmake ..' or 'cmake -DVTK_DIR:PATH=C:/MyProjects/VTK-bin ..', and wait for the build to complete. **NOTE:** The path is the path to your VTK build.

```
C:\VTK\Sphere\build>cmake -DVTK_DIR:PATH=C:/VTK/VTK-Build ..
-- Building for: Visual Studio 15 2017
-- The C compiler identification is MSVC 19.10.25019.0
-- The CXX compiler identification is MSVC 19.10.25019.0
-- Check for working C compiler: C:/Program Files (x86)/Microsoft Visual Studio/2017/Community/VC/Tools/MSVC/14.10.25017
/bin/HostX86/x86/cl.exe
-- Check for working C compiler: C:/Program Files (x86)/Microsoft Visual Studio/2017/Community/VC/Tools/MSVC/14.10.25017
/bin/HostX86/x86/cl.exe -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working CXX compiler: C:/Program Files (x86)/Microsoft Visual Studio/2017/Community/VC/Tools/MSVC/14.10.250
17/bin/HostX86/x86/cl.exe
-- Check for working CXX compiler: C:/Program Files (x86)/Microsoft Visual Studio/2017/Community/VC/Tools/MSVC/14.10.250
17/bin/HostX86/x86/cl.exe -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: C:/VTK/Sphere/build
```

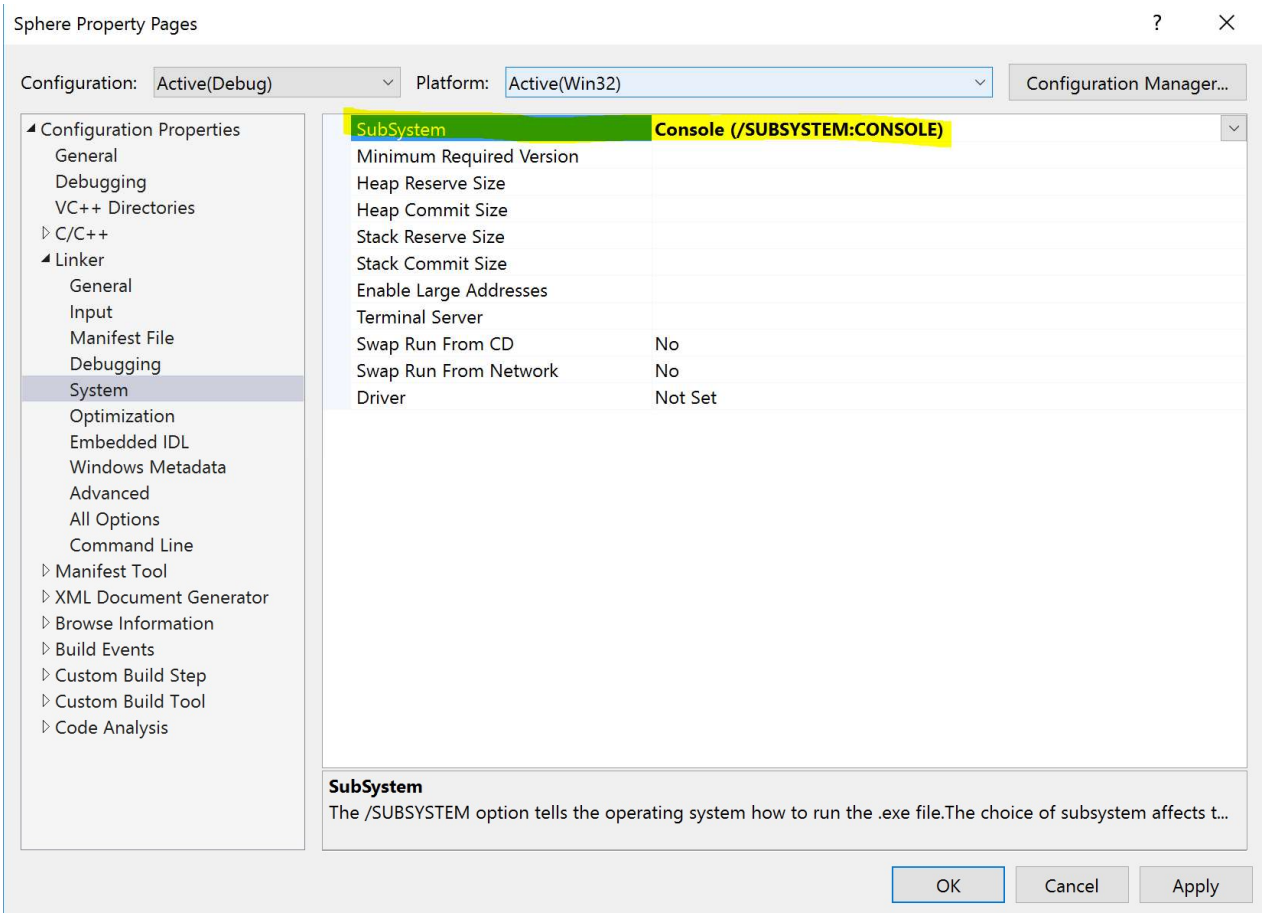
After this, there should be a 'Sphere.sln' file in the build directory. Click on it to have Visual Studio open up the project.



In the 'Solution Explorer', make sure 'Sphere' is highlighted. If not, just click on it.

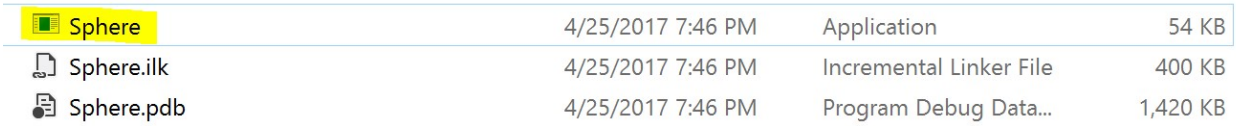


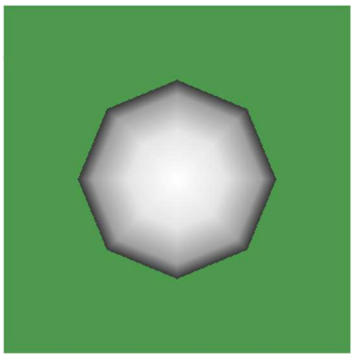
Go to ‘Project -> Properties’ to open up the Sphere Property Pages. Under ‘Linker -> System’, there is a field called SubSystem. Make sure the value says ‘Console (/SUBSYSTEM:CONSOLE)’. If not, change it.



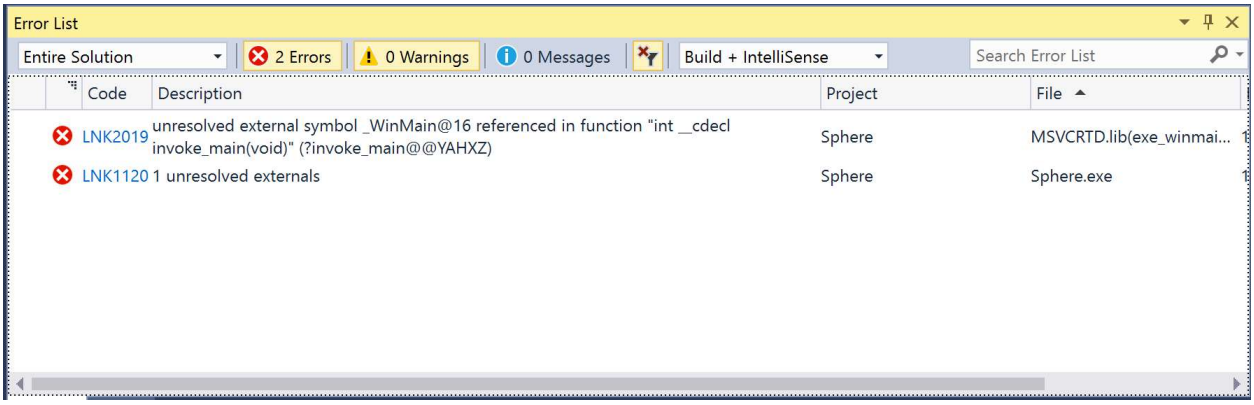
Press ‘OK’.

Ensuring that ‘Sphere’ is highlighted, click ‘Build -> Build Sphere’. This should build the solution, creating an executable (.exe) file in ‘Sphere/build/Debug’. Run the .exe and enjoy!





Possible Errors



➔ Make sure the SubSystem is Console.

".dll files do not exist on your computer" after running Sphere.exe

➔ Make sure the system Path includes the directory 'VTK-bin/bin/Debug'.

If you have any questions or concerns, feel free to email me at sgupta2020@gwu.edu.