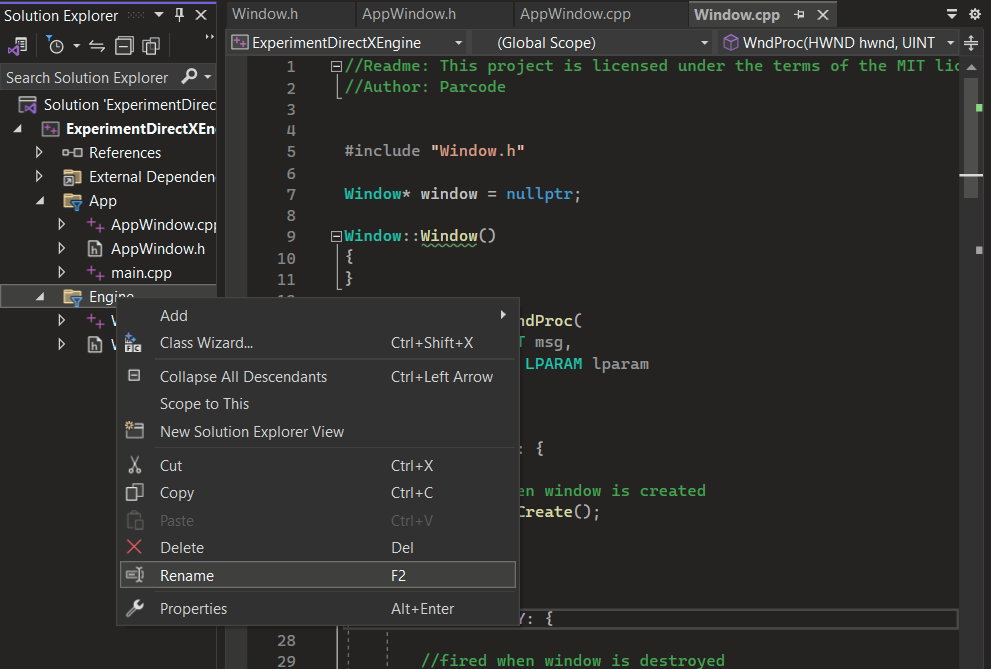
Readme: This project is licensed under the terms of the MIT license.

Video: [C++ 3D Game Tutorial 2: Creating 3D Graphics Engine - Initialization](https://www.youtube.com/watch?v=j7JP0aiRQyY&list=PLv8DnRaQOs5-ST_VDqgbbMRtzMtpK36Hy&index=2)

Author: Parcode

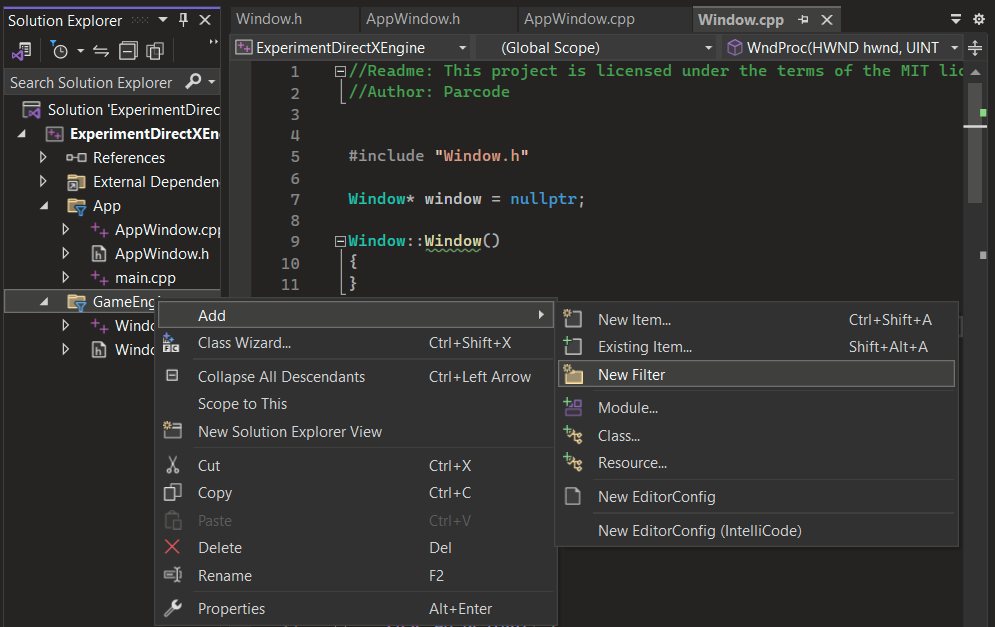
**2.Creating the GraphicsEngine**

1)



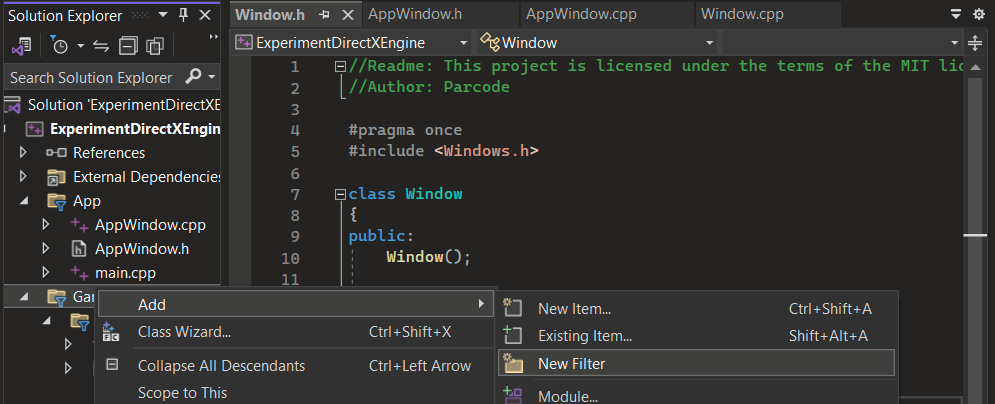
1. Rename the file **Engine** to **GameEngine**.

2)

****

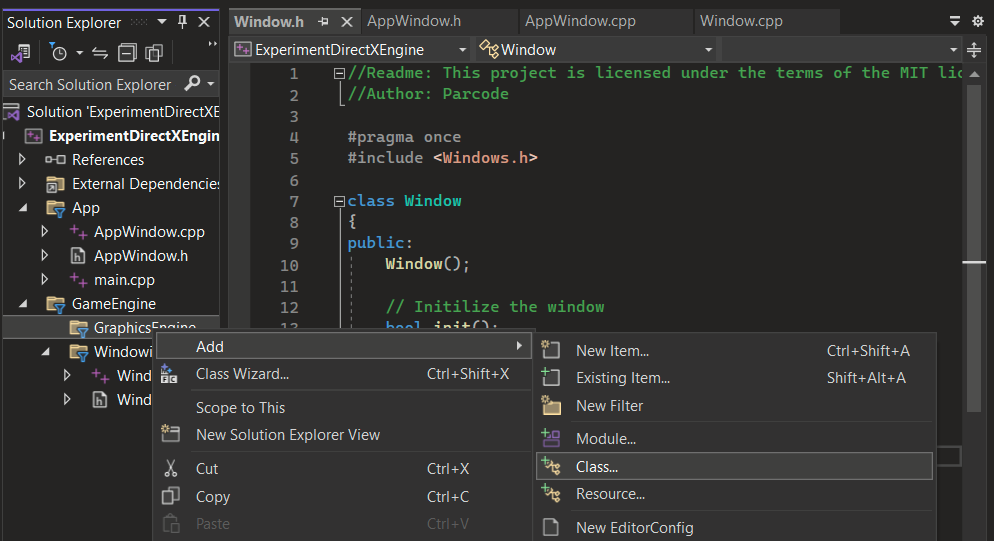
1. Add folder to **GameEngine**
2. Name it **WindowingSystem**
3. Move Window.h and Window.cpp to **WindowingSystem**

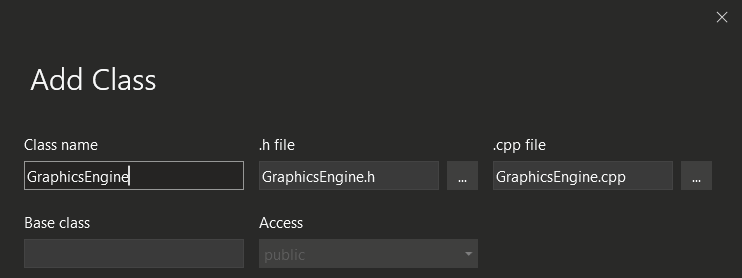
3)



1. Add folder **GrapicsEngine** to **GameEngine** folder

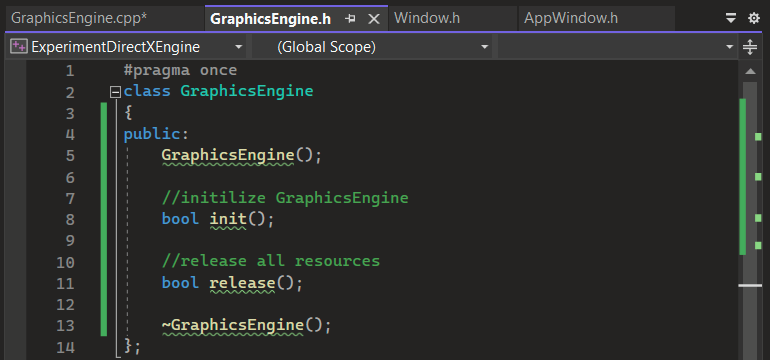
4)

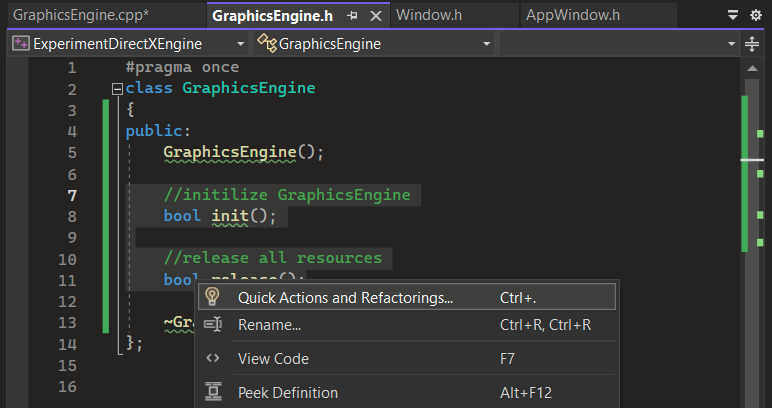


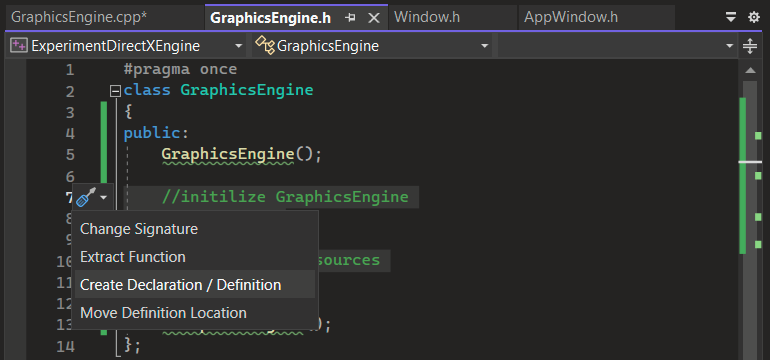


1. Move created files inside the **GraphicsEngine** folder.

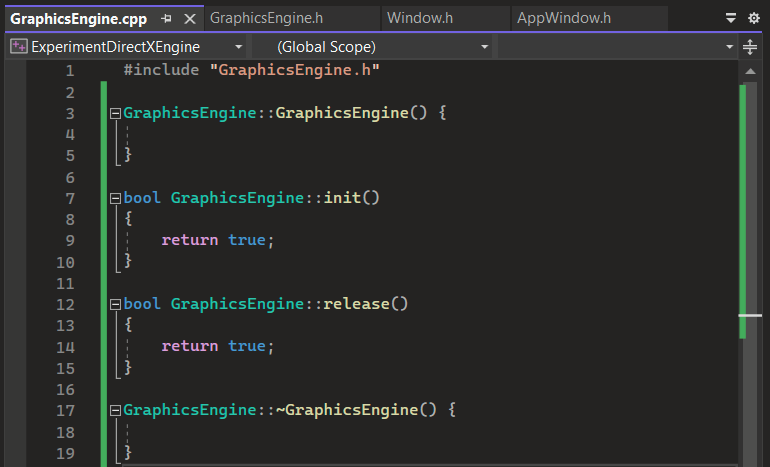
5)



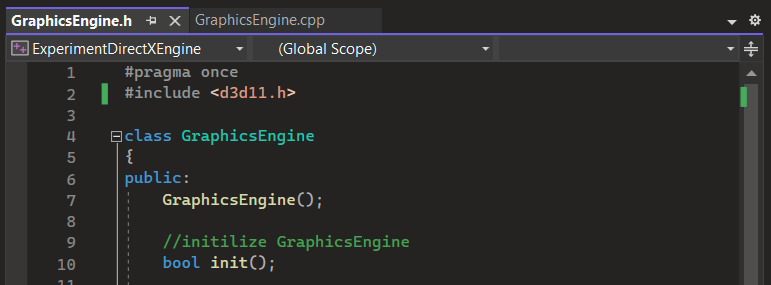




6)



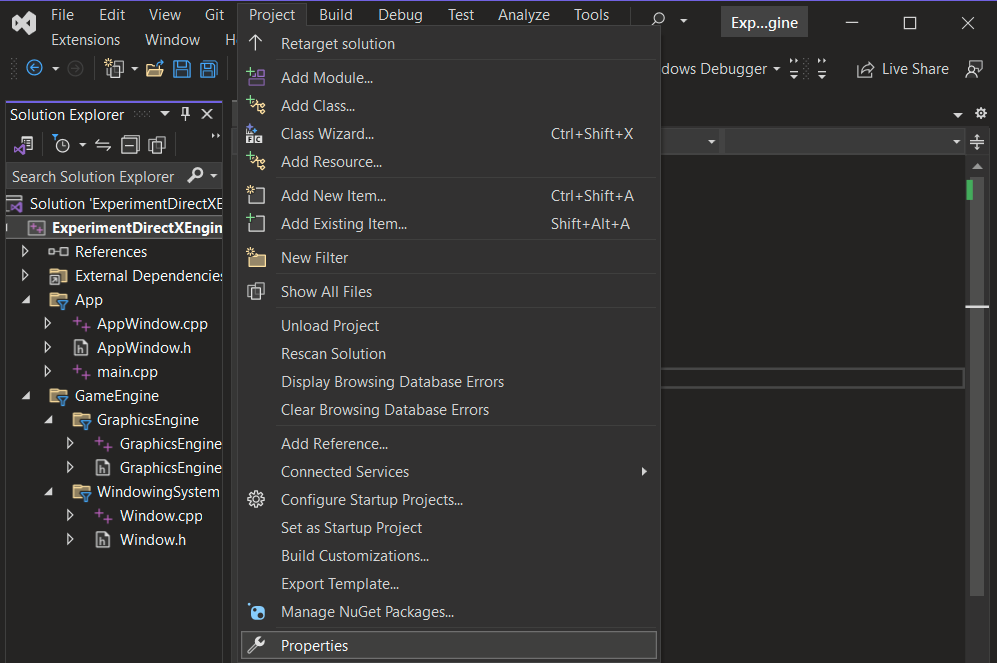
7)

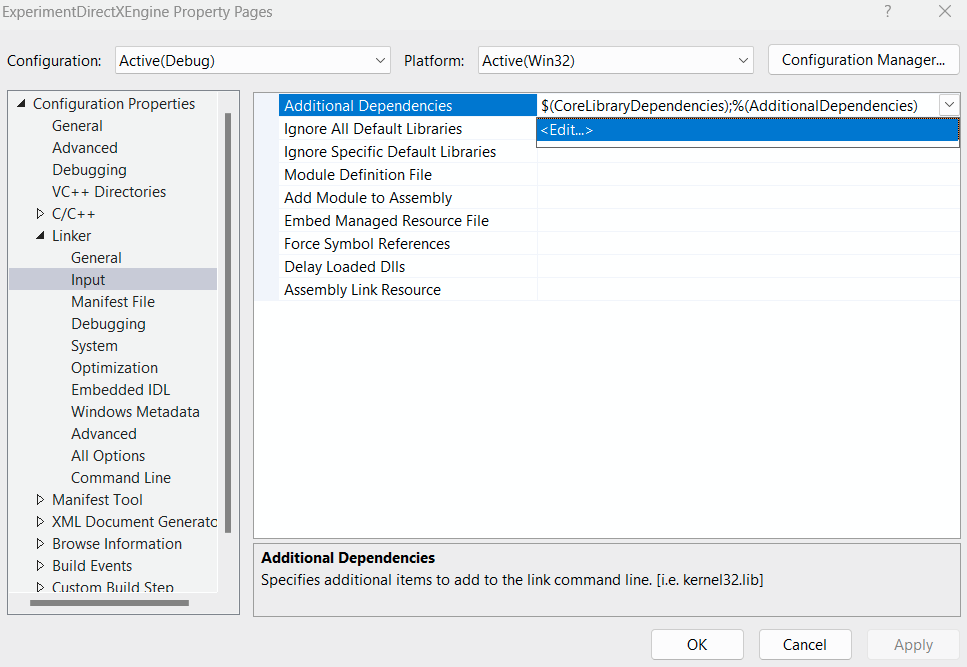


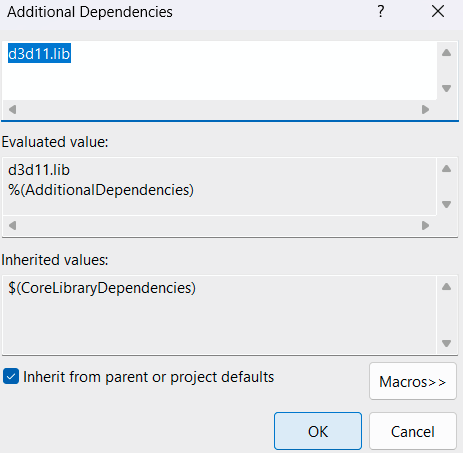
Add the references:

1. #include <d3d11.h>

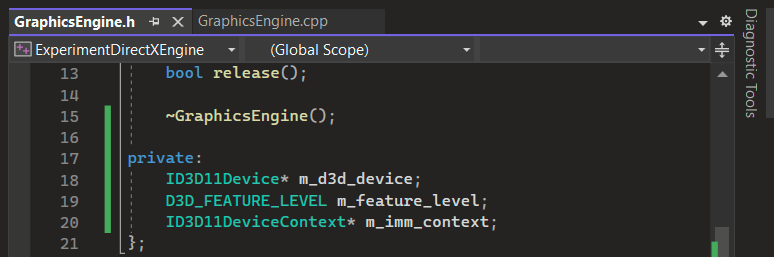
8)





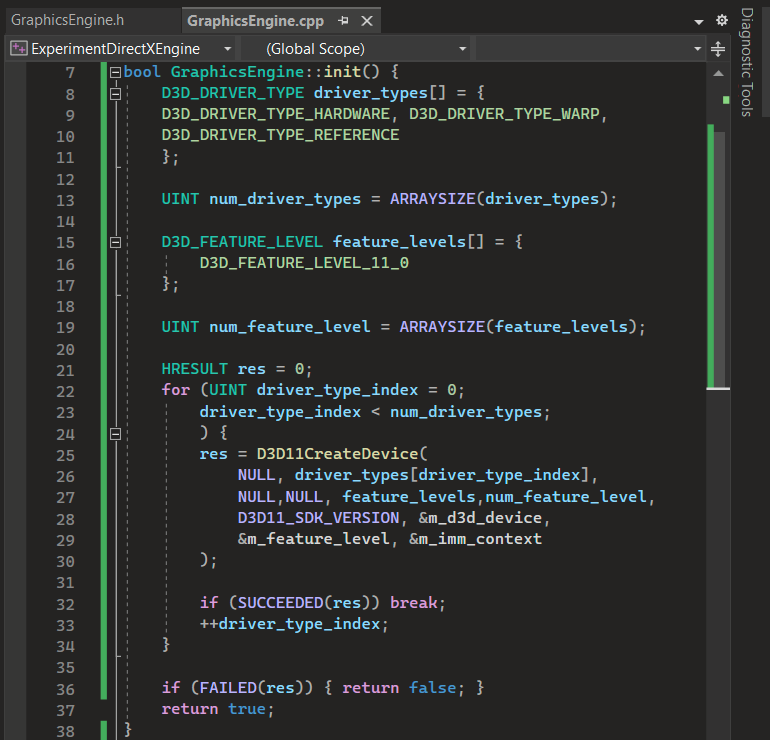


9)



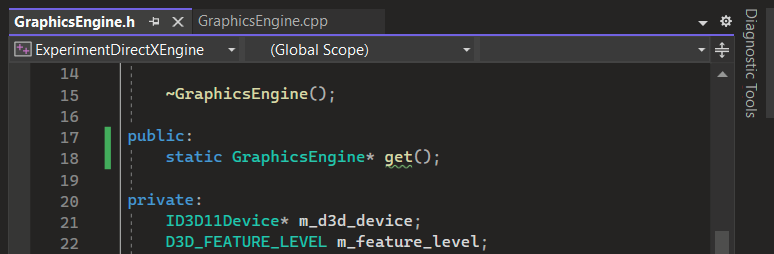
1. **ID3D11Device** - creates and manages graphics resources related to 3D rendering
2. **D3D\_FEATURE\_LEVEL** - determine the available capabilities of the GPU
3. **ID3D11DeviceContext** - specifies how 3D objects are drawn to screen

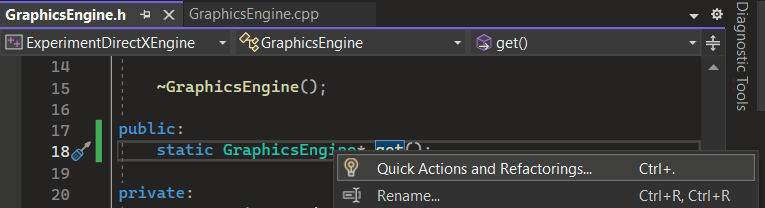
10)

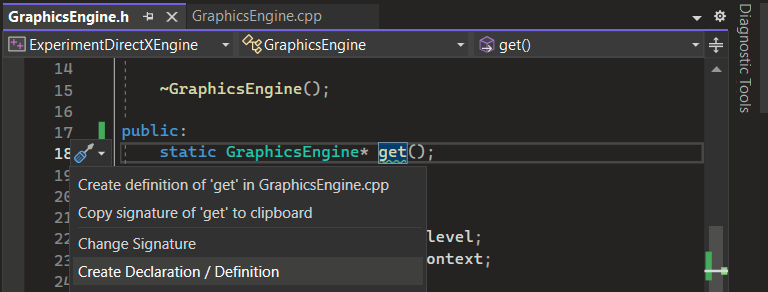


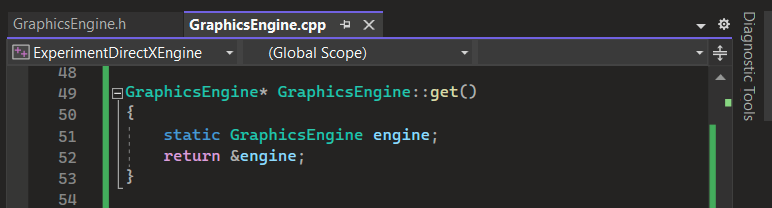
1. **D3D\_DRIVER\_TYPE** - specifies the type of graphics driver to be used
2. **D3D\_DRIVER\_TYPE\_HARDWARE** - uses the hardware-accelerated graphics driver
3. **D3D\_DRIVER\_TYPE\_WARP** - renders on CPU when there isn't a dedicated GPU
4. **D3D\_DRIVER\_TYPE\_REFERENCE** - tests compatibility with other hardware
5. **ARRAYSIZE** - calculates the number of elements in an array
6. **HRESULT** - checks whether the function succeeded or failed
7. **D3D11CreateDevice** - creates a Direct3D device to render 3D graphics
8. **SUCCEEDED** - checks if function succeeded
9. **FAILED** - checks if function failed

11)

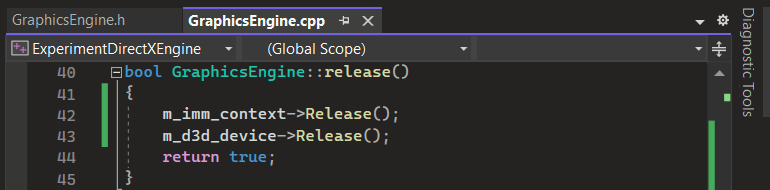




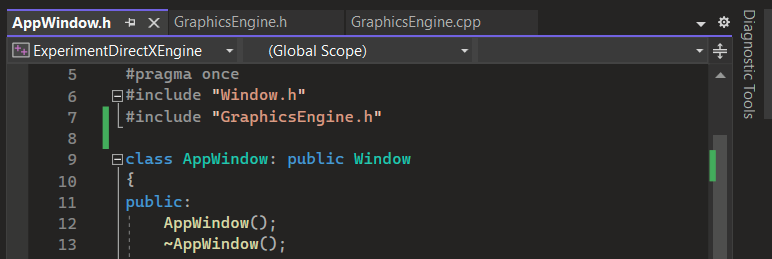


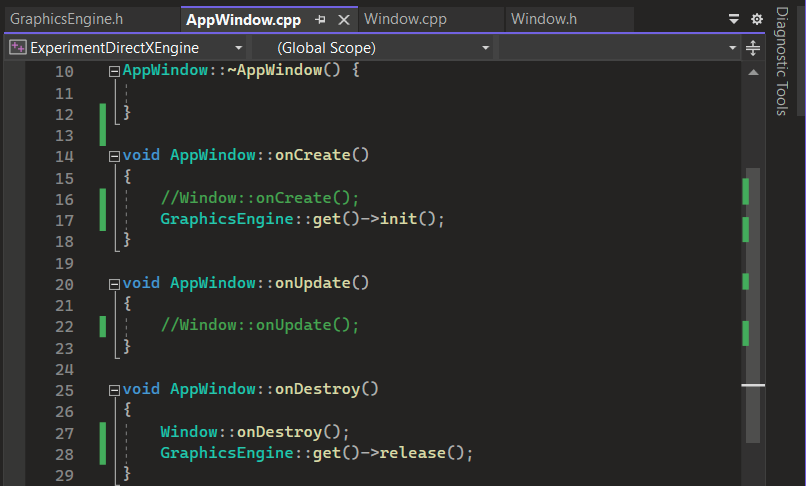


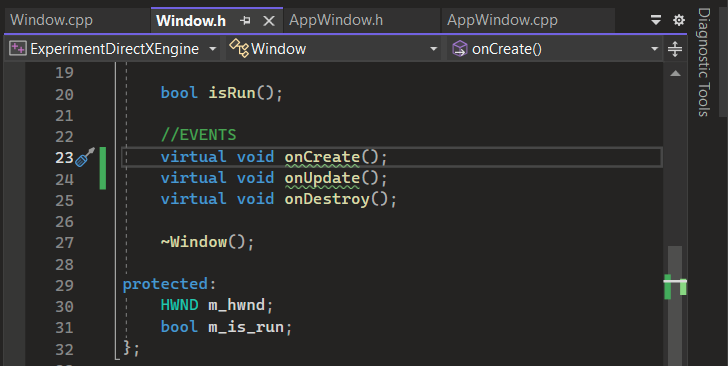
12)



13)







14)

