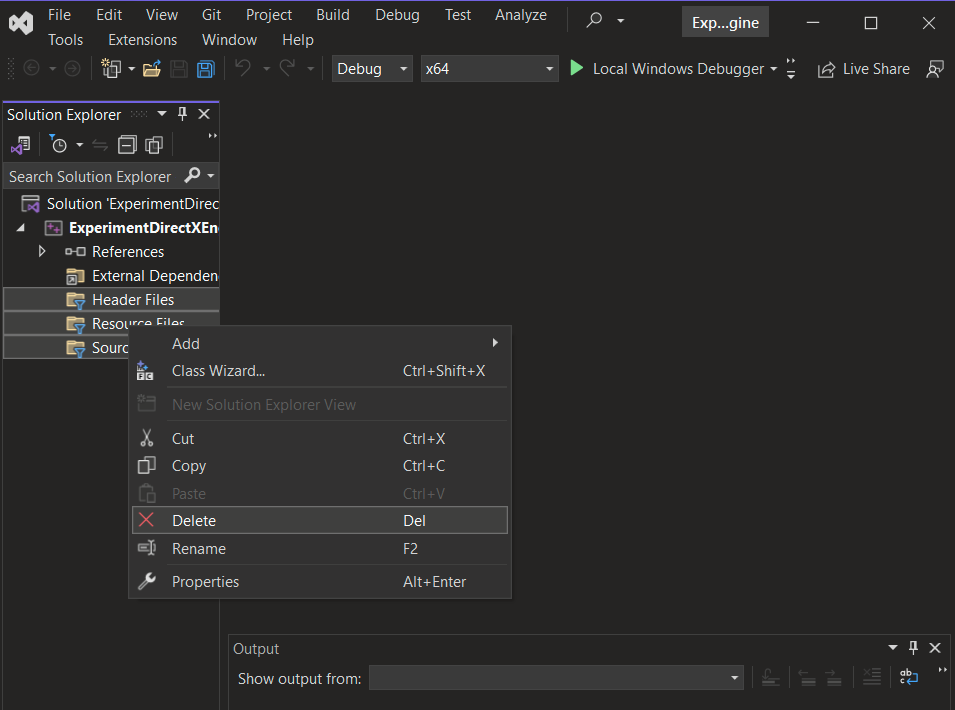
Readme: This project is licensed under the terms of the MIT license.

Video: [C++ 3D Game Tutorial 1: Creating a Window with Win32 API](https://www.youtube.com/watch?v=ssGka-bSTvQ&list=PLv8DnRaQOs5-ST_VDqgbbMRtzMtpK36Hy&index=1)

Author: Parcode

**1.Creating a Window**

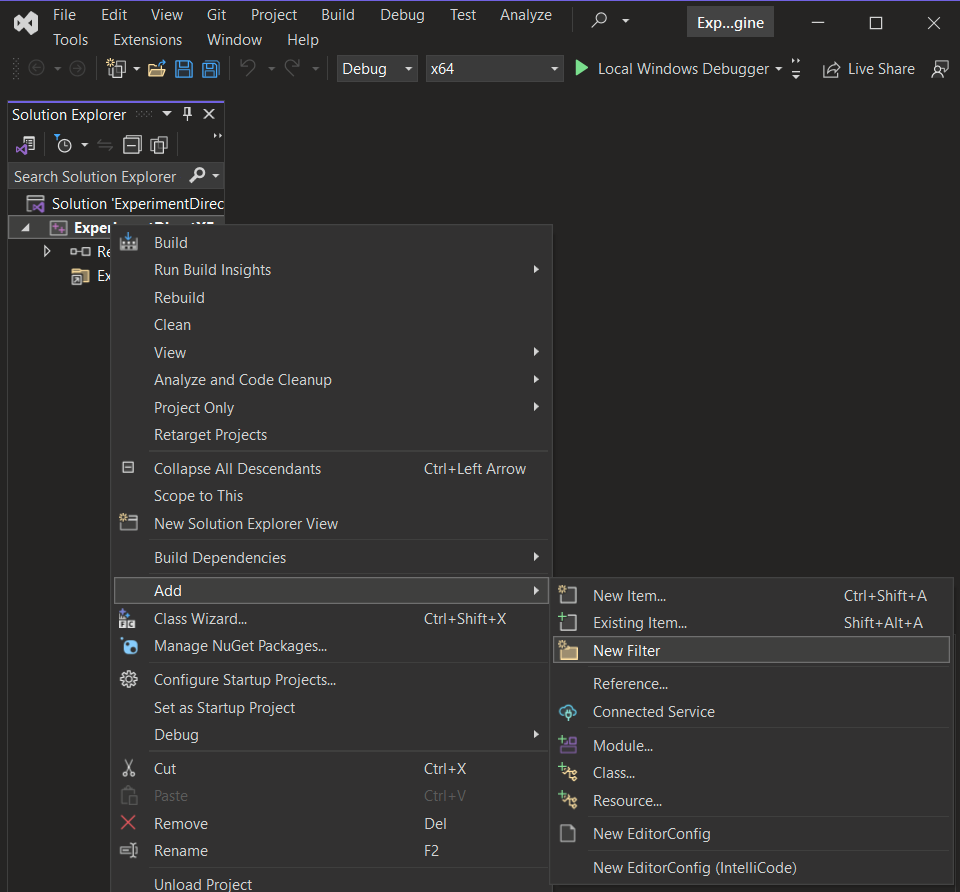
1)



Delete the folder’s named:

1. Header Files
2. Resource Files
3. Source Files

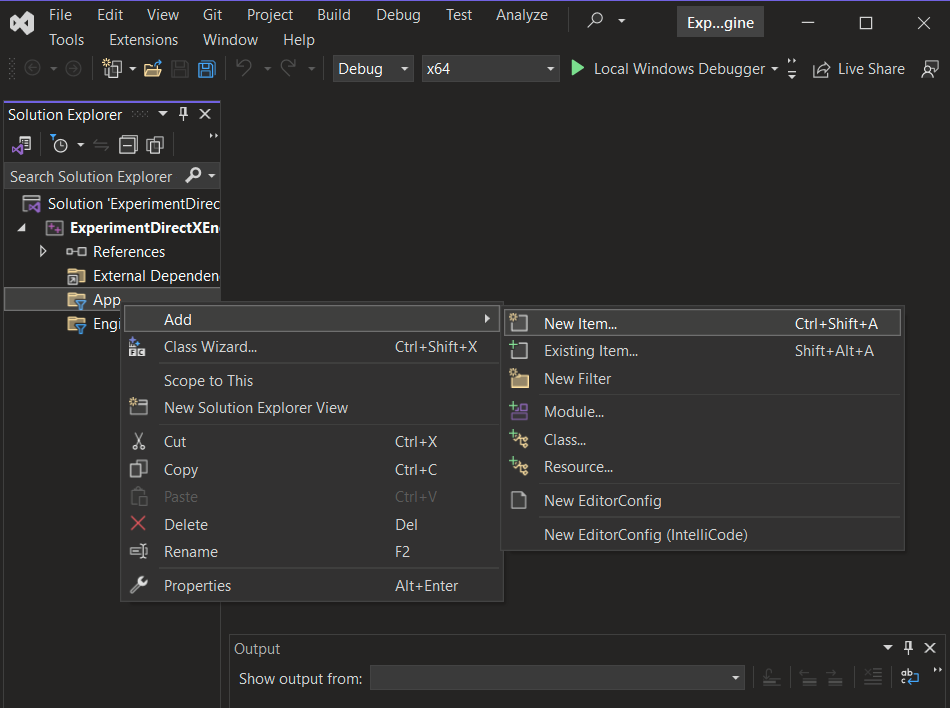
2)



Add folder’s named:

1. App
2. Engine

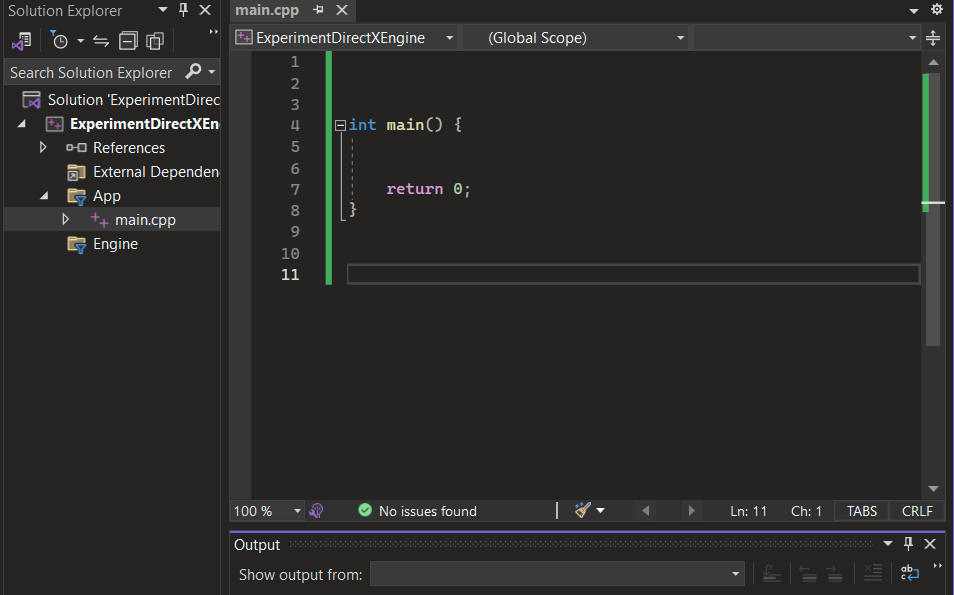
3)



Add item to App folder:

1. main.cpp

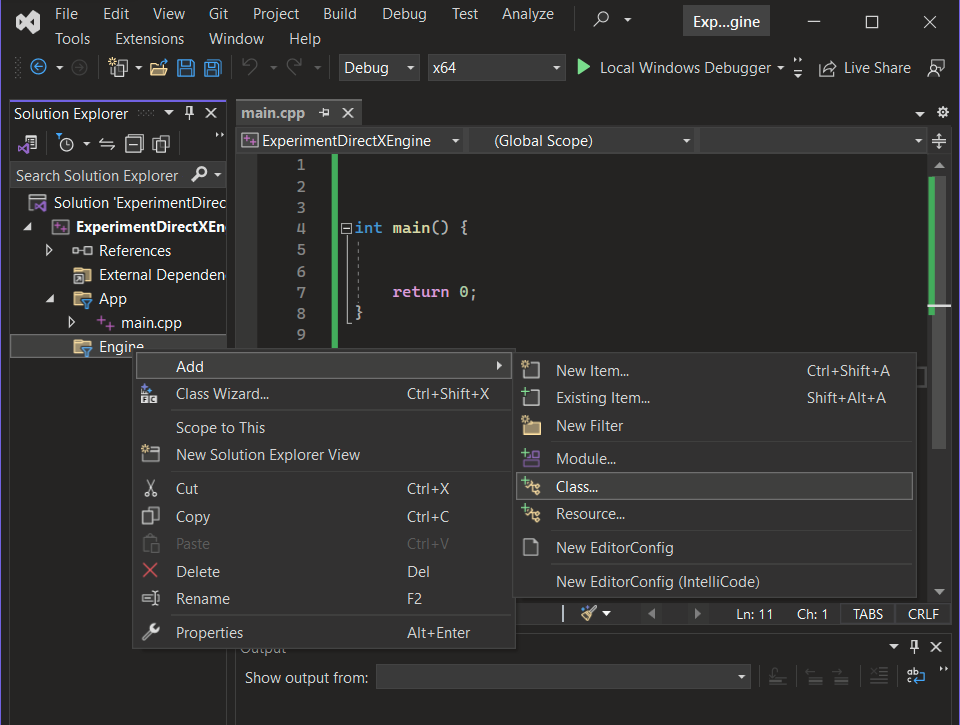
4)

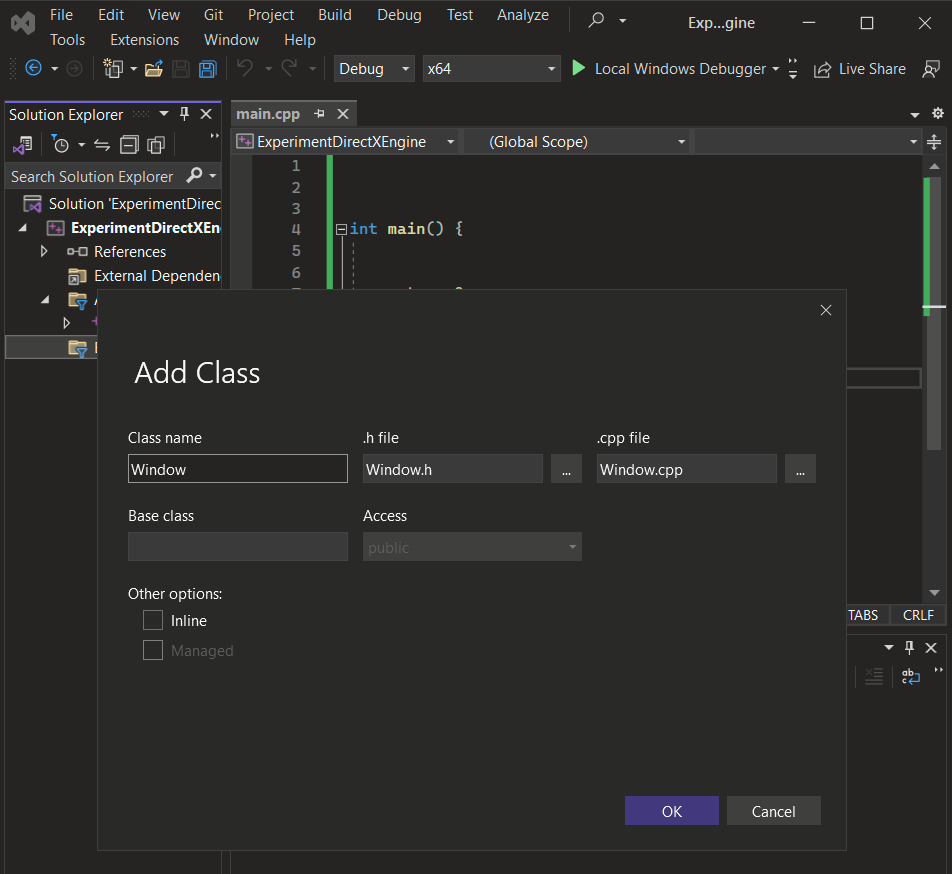


Add the code inside the main.cpp file:

1. int main()

5)

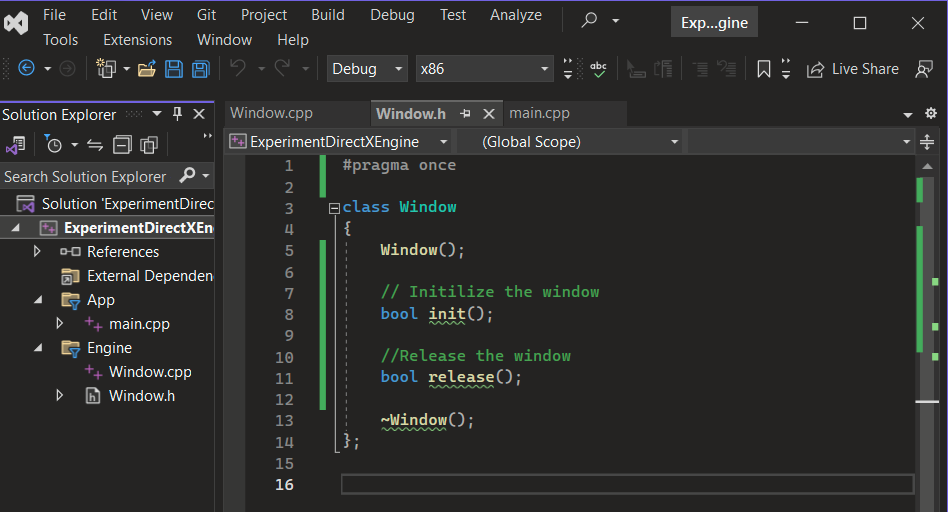




Move the created files into the Engine Folder:

1. Window.cpp
2. Window.h

6)



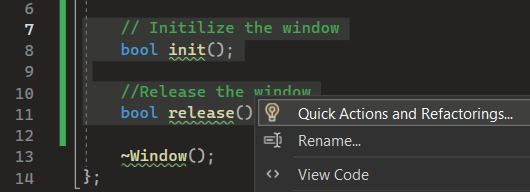
Inside the Window Class add:

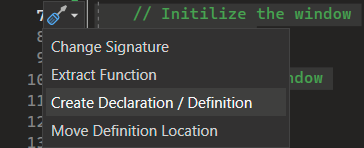
1. Window();
2. bool init();
3. bool release();
4. ~Window();

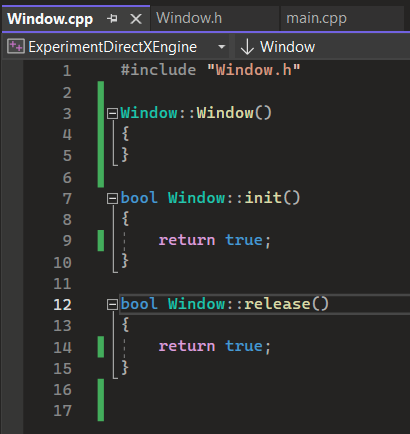
The ~ symbol is the destructor and it gets called to free up memory.

<https://stackoverflow.com/questions/5343437/meaning-of-tilde-symbol-in-c>

7)





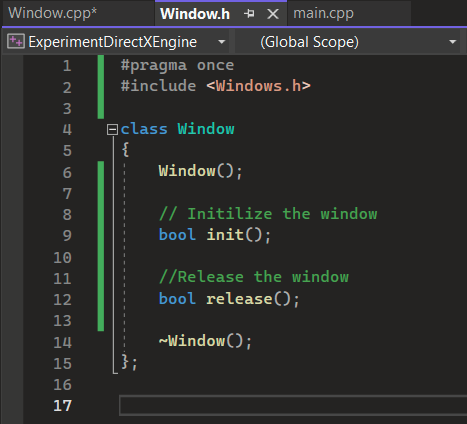


By using quick actions the functions **init()** and **release()** get created automatically.

Change the **false** statements to **true** in:

1. init()
2. release()

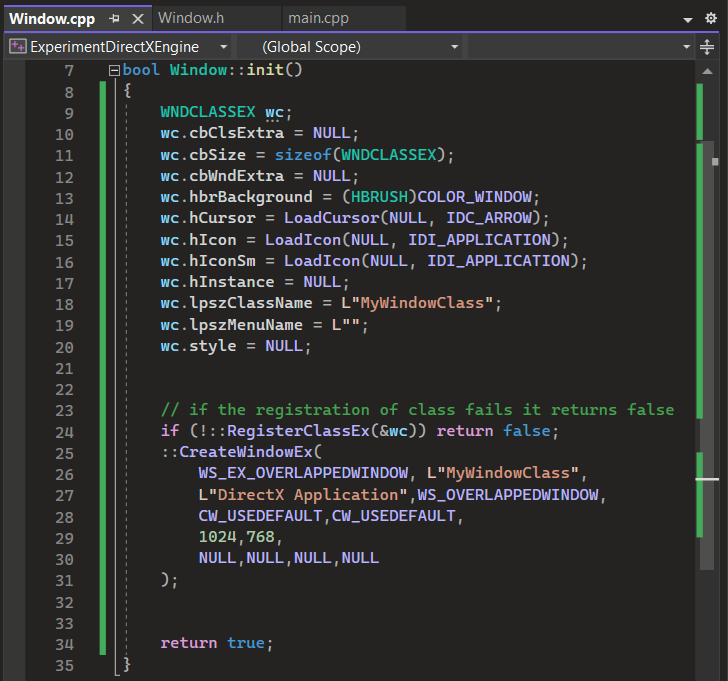
8)



Add the reference to the **Window.h** file:

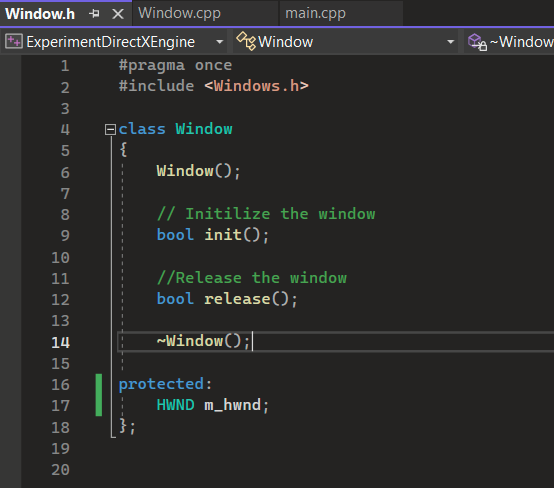
1. #include <Windows.h>

9)



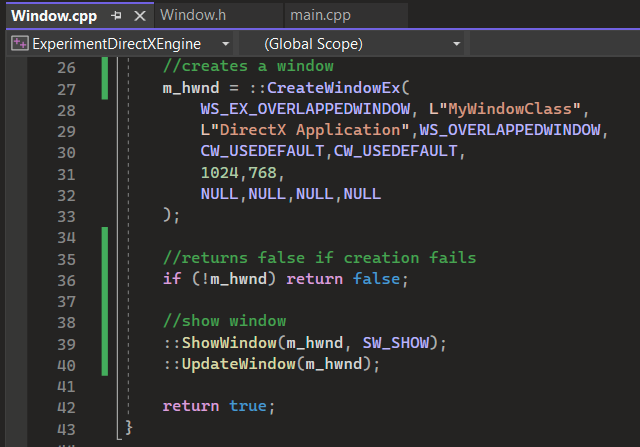
1. **WNDCLASSEX** structures: [MicrosoftWindowsLink](https://learn.microsoft.com/en-us/windows/win32/api/winuser/ns-winuser-wndclassexa)
   1. **cbCLSExtra** - allocates extra memory for the class structure itself meaning all windows get affected
   2. **cbSize** - setting the size of data for corruption check [StackOverflowLink](https://stackoverflow.com/questions/189622/what-is-the-purpose-of-the-cbsize-member-in-win32api-structs)
   3. **cbWndExtra** - allocates extra memory for each instance of a window meaning individual windows are allocated
   4. **hbrBackground** - defines the background color or pattern for the window
      1. **(HBRUSH)COLOR\_WINDOW** - sets the background of the window to the default system window color
   5. **hCursor** - determines the mouse pointer's appearance
      1. **LoadCursor(NULL, IDC\_ARROW)** - loads the standard arrow cursor
   6. **hIcon** - small image at the top left side of the title bar
   7. **hIconSm** - appearance when pressed Alt-Tab
   8. **hInstance** - differentiates itself from other applications
   9. **lpszClassName** - identifies and creates windows based on this class
      1. **L** - creates a wide-character string (Unicode string)
   10. l**pszMenuName** - tells which menu to use for that class
   11. **style** - define the initial style of windows created from the window class
   12. [RegisterClassEx](https://msdn.microsoft.com/en-us/library/windows/desktop/ms633587.aspx) - registers a  [*window class*](https://msdn.microsoft.com/en-us/library/windows/desktop/ms632596.aspx): [StackOverflowLink](https://stackoverflow.com/questions/31930556/what-is-the-difference-between-wndclassex-and-createwindow)
       1. & - passes a pointer by taking the memory address of **wc**
   13. **CreateWindowEx** - creates a new window by using:
       1. **dwExStyle** - extended window style
       2. **lpClassName** - window class name
       3. **lpWindowName** - window title
       4. **dwStyle** - window style
       5. **x, y** - position
       6. **nWidth**, **nHeight** - size
       7. **hWndParent**- parent window
       8. **hMenu** - menu handle
       9. **hInstance** - application instance
       10. **lpParam** - user-defined data
   14. **WS\_EX\_OVERLAPPEDWINDOW** - includes features such as a title bar, a system menu, minimize and maximize buttons, and a close button

10)



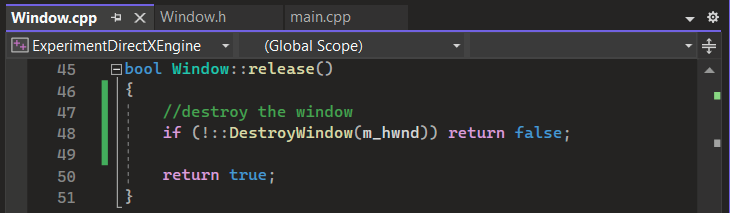
1. **Protected** - members are accessible only within the class and its subclasses
2. **HWND** - used for manipulating windows and GUI elements

11)



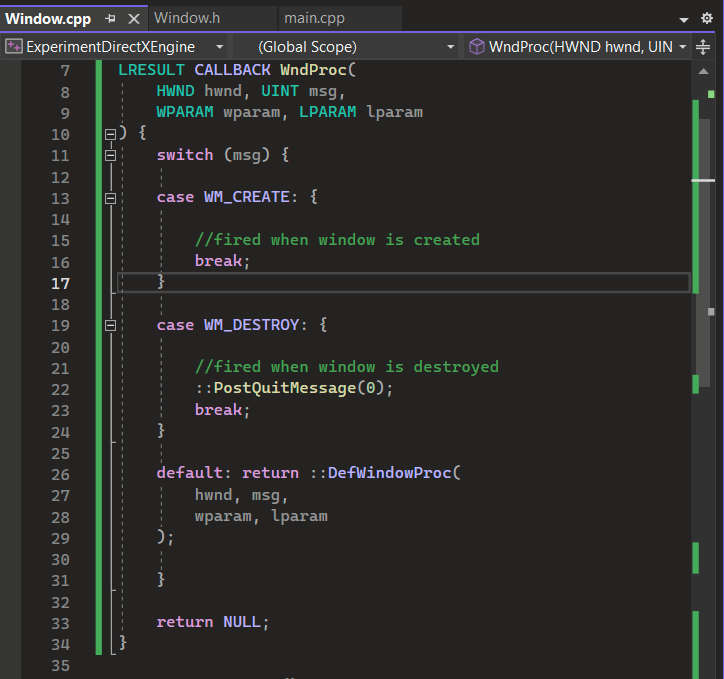
1. add **m\_hwnd** to **CreateWindowEX()**
2. **ShowWindow** - shows the window
   1. **SW\_SHOW** - shows the window in its normal state
3. **UpdateWindow** - forces window to repaint itself immediately

11)



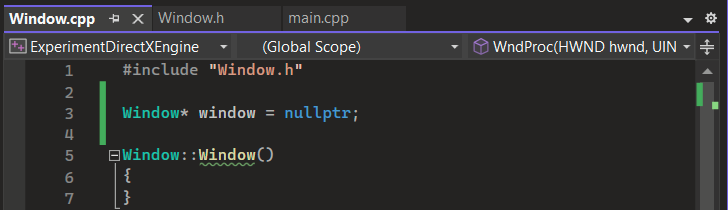
1. **DestroyWindow** - closes the window

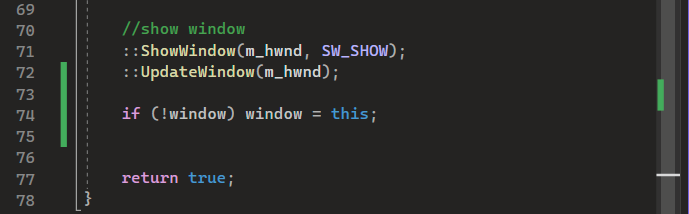
12)



1. **LRESULT** - returns various types of information such as messages of errors
2. **CALLBACK** - specifies how the function is called and how its parameters are passed
3. **lpfnWndProc** - handles messages sent to a window of this class
   1. **UINT** - unsigned integer meant to store positive values only
   2. **WPARAM** - word parameter meant for passing messages
   3. **LPARAM** - long parameter meant for passing
4. **switch** - control structure used to make decisions
5. **WM\_CREATE** - message is sent by the system to notify the window about its creation
6. **break** - exits out of the current control structure
7. **WM\_DESTROY** - sends message when the window is about to be closed
8. **PostQuitMessage** - posts a message that signals the application to exit
   1. **0** - indicates a normal application termination
9. **default** - used when no other option is selected
10. **DefWindowProc** - default behavior for window messages

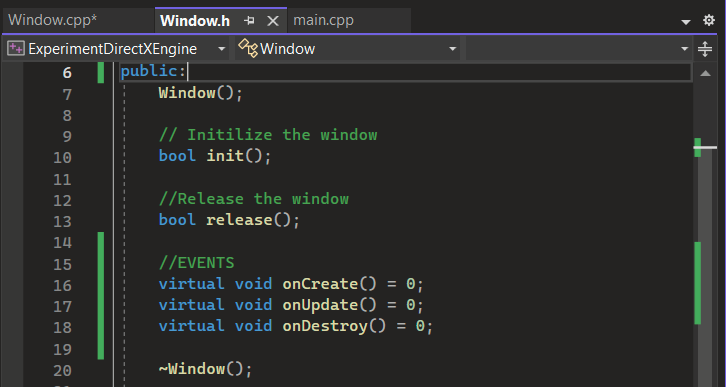
13)





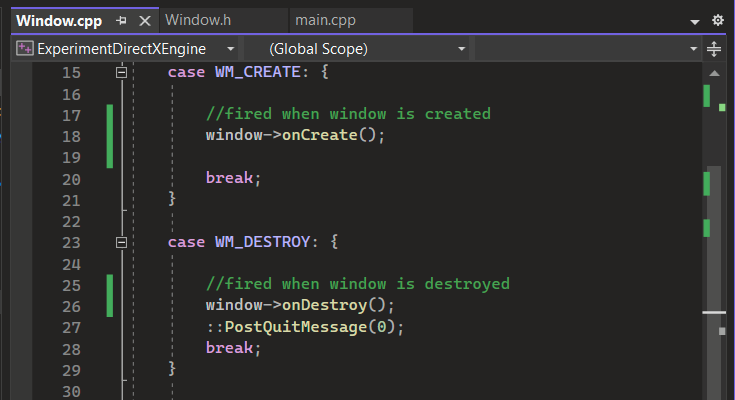
1. **Window** - refers to a rectangular on-screen area
   1. **\*** - used to declare pointers
2. **nullptr** - represents a null or empty pointer
3. **this** - refer to the current object or instance within a class

14)



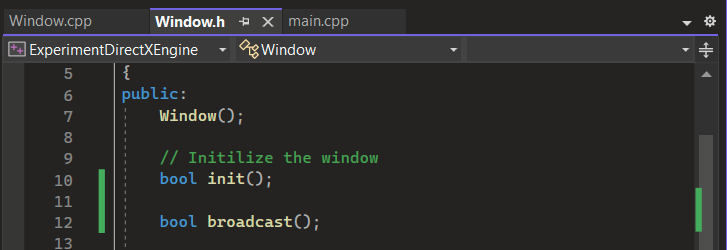
1. **public** - accessible from any part of the program
2. **virtual** - base class which lets objects manipulate themselves by using that base class
3. **void** - code that doesn’t have a type

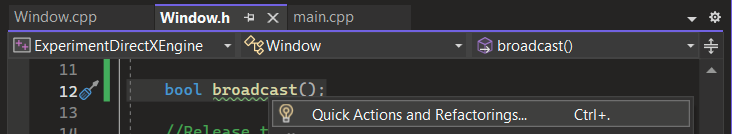
15)

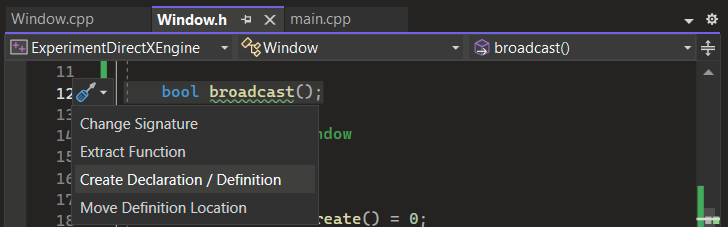


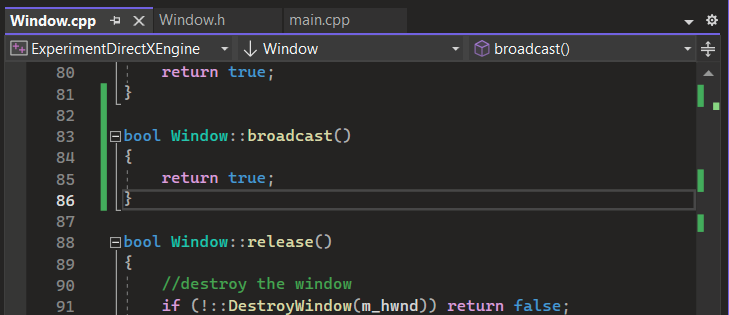
1. **->** - used to access members of an object or a structure through a pointer

16)

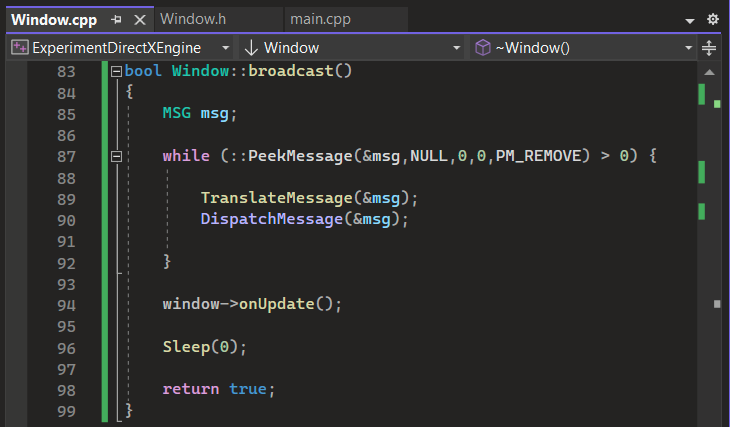






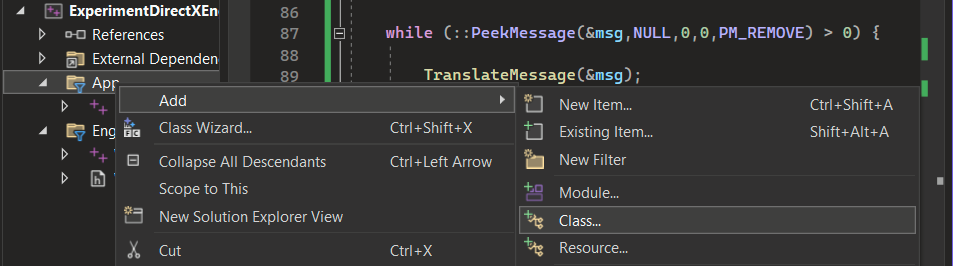


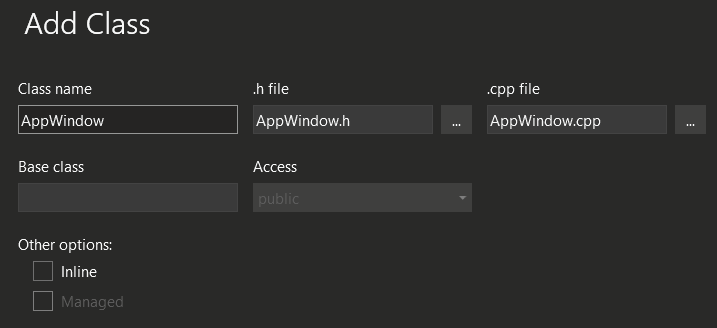
17)



1. **while** - continuously runs a code as long as the condition is true
2. **PeekMessage** - checks messages without removing them
   1. **PM\_REMOVE** - message should be removed
3. **TranslateMessage** - processes keyboard or mouse input from the user
4. **DispatchMessage** - sends the processed inputs to the window
5. **Sleep** - how long the program should pause
   1. **0** - milliseconds

18)

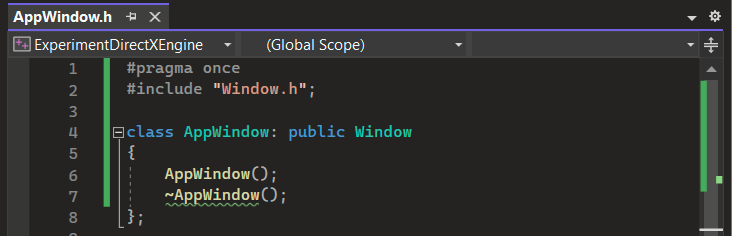


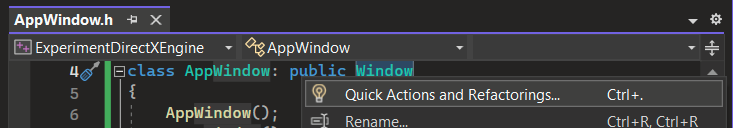


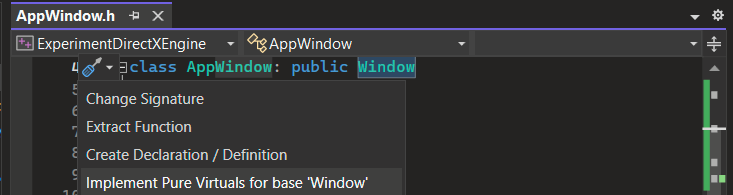
Move the created files to the **App** folder:

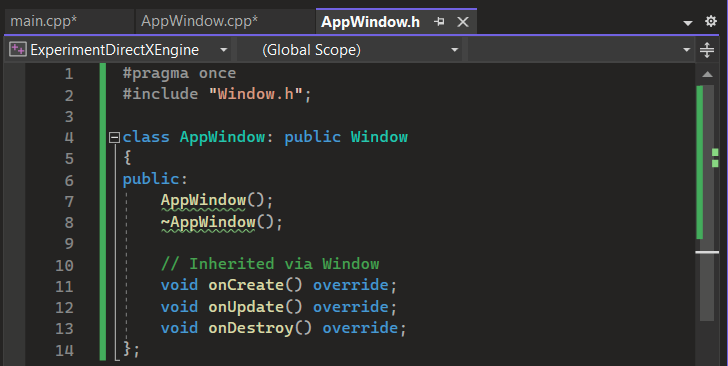
1. AppWindow.cpp
2. AppWindow.h

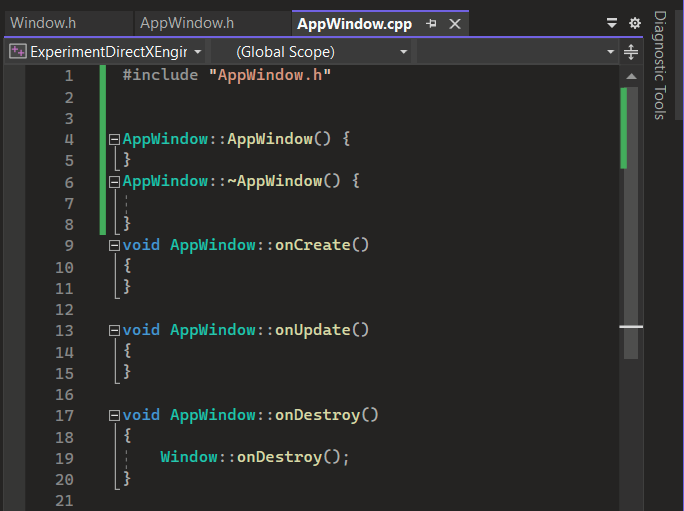
19)



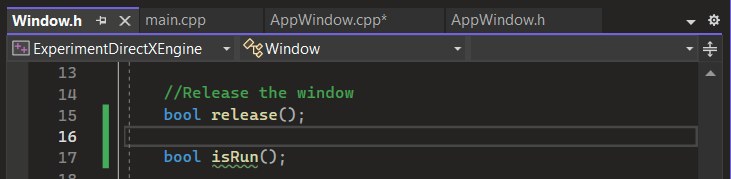


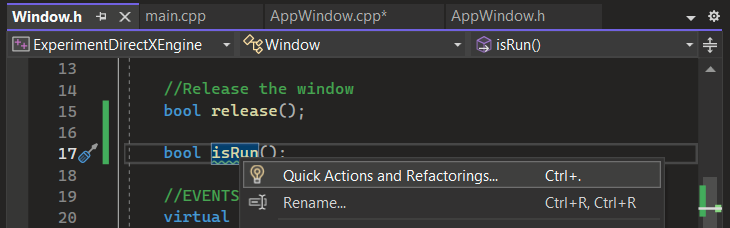


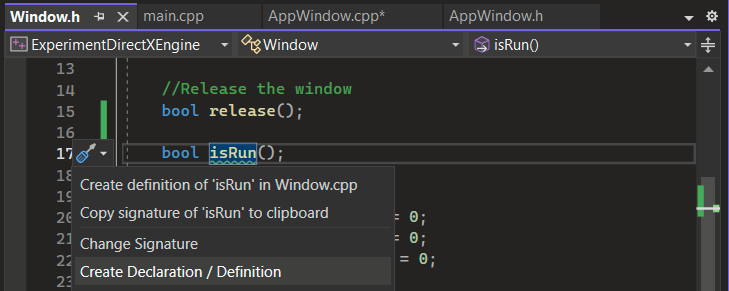


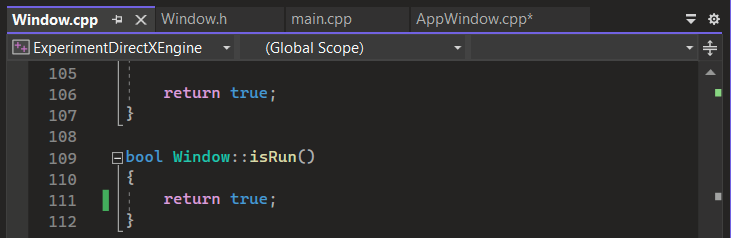


20)

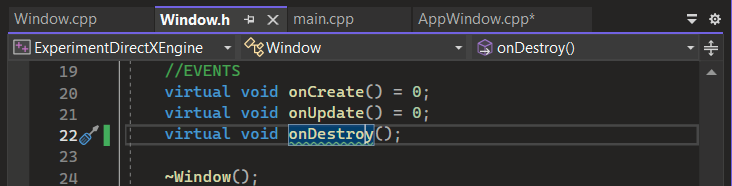


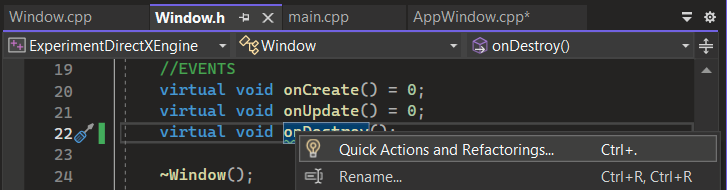


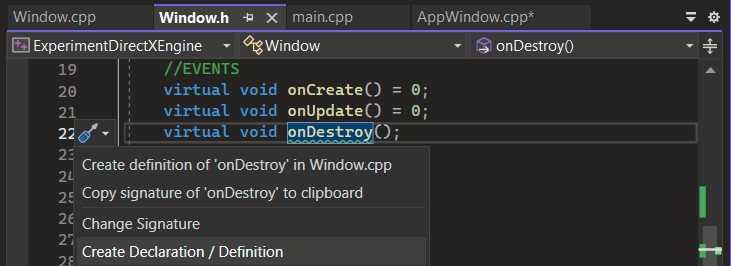


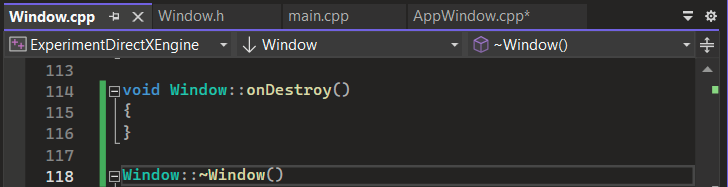


21)

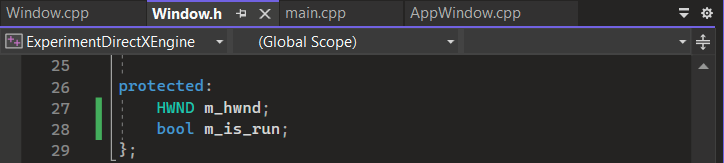


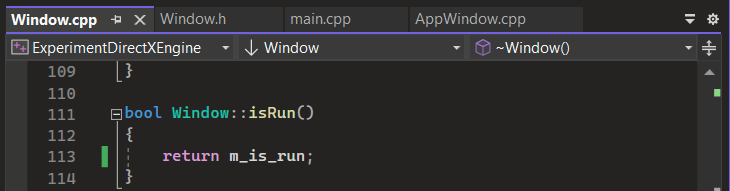


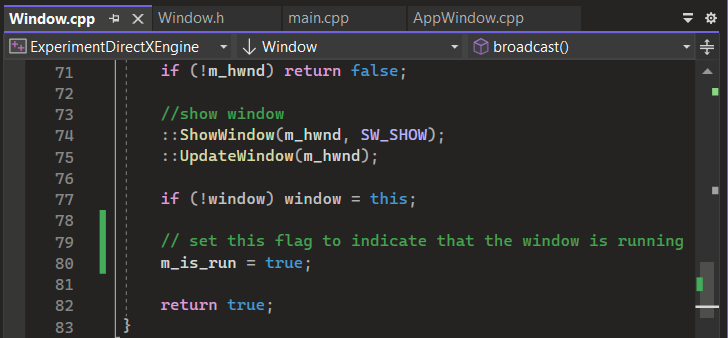


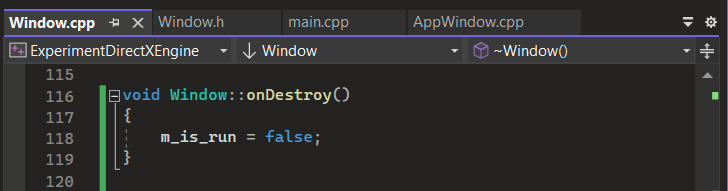


22)

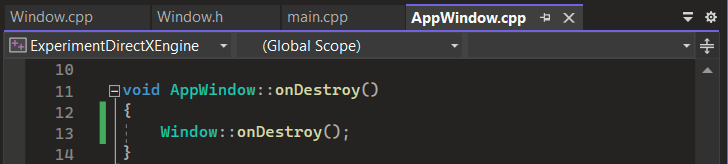


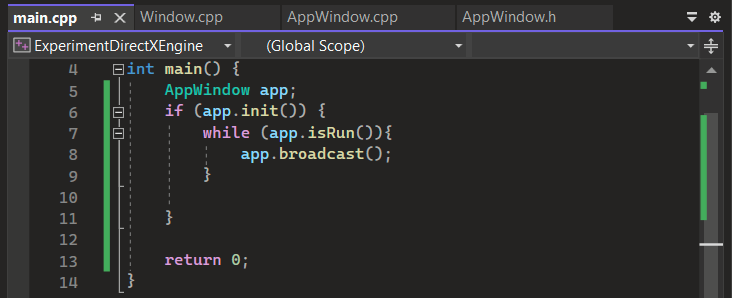


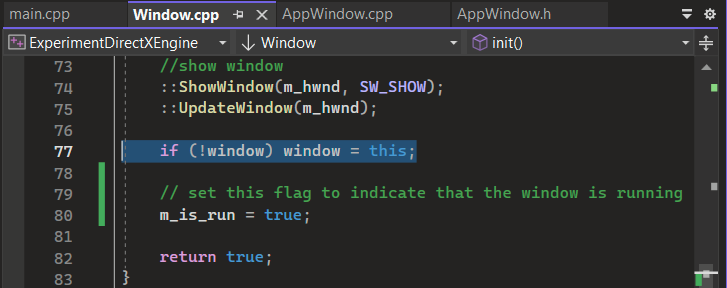


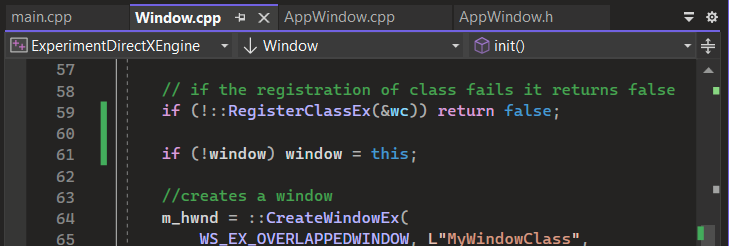


23)

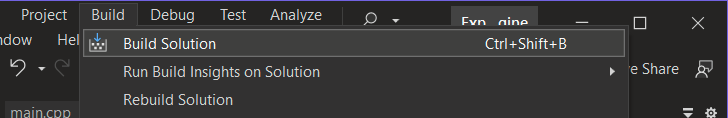








24)



25)

