

bumpy road to code

personal path to master code

 Menu

How to install and use fltk-1.3.4 in Visual Studio 2017 [complete guide]

Blog, Principles and Practice Using C++ August 5, 2017December 16, 2017 8 Minutes

Last successfully tested: 30.11.2017

Chapter 12 to 16 of Bjarne Stroustrup's – ["Programming Principles and Practice Using C++"](#) (<http://amzn.to/2y3yx9f>) uses a GUI library called FLTK (Fast Light Tool Kit, "full tick"). This guide demonstrates how to download, compile from source, build and use the fltk version 1.3.4 with Visual Studio 2017 Community.

As it was quiet frustrating getting everything to run, I thought I would take the extra time to make it extra detailed, I hope it is not confusing or over complicating things for who ever is struggling with this.

(NOTE: Before you start, even though the following path will get you to compile and run fltk projects, and it is the Bjarne Stroustrup's Programming Principles and Practice way, the fltk.org team recommended to do it like this [guide](#) (<http://wp.me/p8xwHM-jw>) shows. It is to prevent cross contamination and keep your Visual Studio 2017 environment clean and tidy. The original discussion can be found [here](#) (<https://groups.google.com/d/msg/fltkgeneral/eLXGNQ3mndo/c0hBqxUhCQAJ>). I just left this guide on my page in case people persist on doing it the Bjarne Stroustrup way.)

Step 1: Downloading fltk-1.3.4.

Go to: <http://www.fltk.org/index.php> (<http://www.fltk.org/index.php>)

Go to the download tab and click on **fltk-1.3.4-1-source.tar.gz**

The screenshot shows the FLTK Download page with the 'Source Code' section selected. A table lists public releases for various versions of FLTK, including source tarballs and documentation PDFs. The table includes columns for Version, Filename, Size, and MD5 Sum.

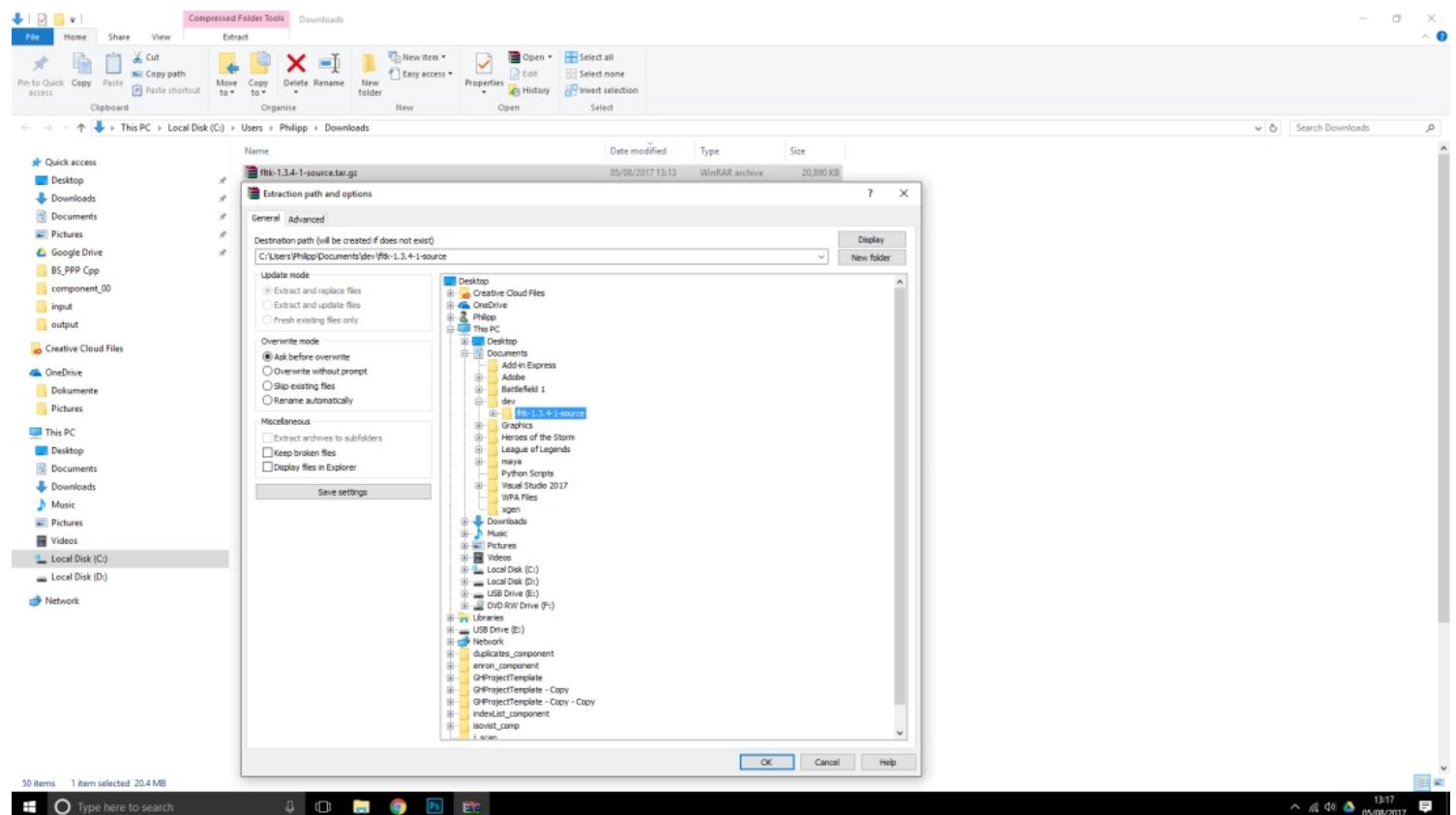
Version	Filename	Size	MD5 Sum
1.3.4	fltk-1.3.4-1-source.tar.gz	5163k	d7fc27a0928648e1a1366dd2e273970
	fltk-1.3.4-docs-pdf.tar.gz	4798k	837f21e38efdb4b76a58b2229ad87cf3
	fltk-1.3.4-docs-html.tar.gz	7329k	a494b7880767f7766a5e5d037a30bce4
1.3.3	fltk-1.3.3-source.tar.gz	4870k	9ccdb0d19dc104b87179bd9fd10822e3
	fltk-1.3.3-docs-html.tar.gz	6919k	290cb2b41b603cfbfa2d4de66910e8
	fltk-1.3.3-docs-pdf.tar.gz	4356k	f111b98ab4a4a86139a0225095a3771762
1.3.2	fltk-1.3.2-source.tar.gz	4162k	9f7e707d4fb7a5a76f0f9b73ff70623d
	fltk-1.3.2-docs-html.tar.gz	3592k	3a989e3c48590decf52f603cd200053a
	fltk-1.3.2-docs-pdf.tar.gz	4311k	dc8ff9a56f1b430cd7b3cce23189ef8a1
1.3.1	fltk-1.3.1-source.tar.gz	4158k	b5b84ed63751ab3f76aa8d36ccc59e7
	fltk-1.3.1-docs-html.tar.gz	3554k	3d98e587321c12fe02b864cc6d91b6ac
	fltk-1.3.1-docs-pdf.tar.gz	3512k	3bb7cd19c17d4166e0ede328111003
1.3.0	fltk-1.3.0-source.tar.gz	4015k	44d5d7ba06afdd36ea17da6b4b703ca3
	fltk-1.3.0-docs-html.tar.gz	3482k	ee7915ccfc211e1d70a3add3f170ef
	fltk-1.3.0-docs-pdf.tar.gz	3373k	8c8aa85f4230b701bed4616f9f2420f
1.1.10	fltk-1.1.10-source.tar.gz	2606k	e6378a76calef073bc0b92df1ef3ba55
	fltk-1.1.10-source.tar.bz2	2163k	a176594abc427ff892c360809fe1fa672
	fltk-1.1.10-source.zip	3483k	d66da1bf9ee70cc0007de44591f251f7
1.0.11	fltk-1.0.11-source.tar.gz	1125k	4c4b71defc32734966f1fe049c3796
	fltk-1.0.11-source.tar.bz2	976k	e917767d121bc08953a5d811e68134ff
	fltk-1.0.11-source.zip	1479k	8e9bc36eb25c5ff01eb69477f60d8437

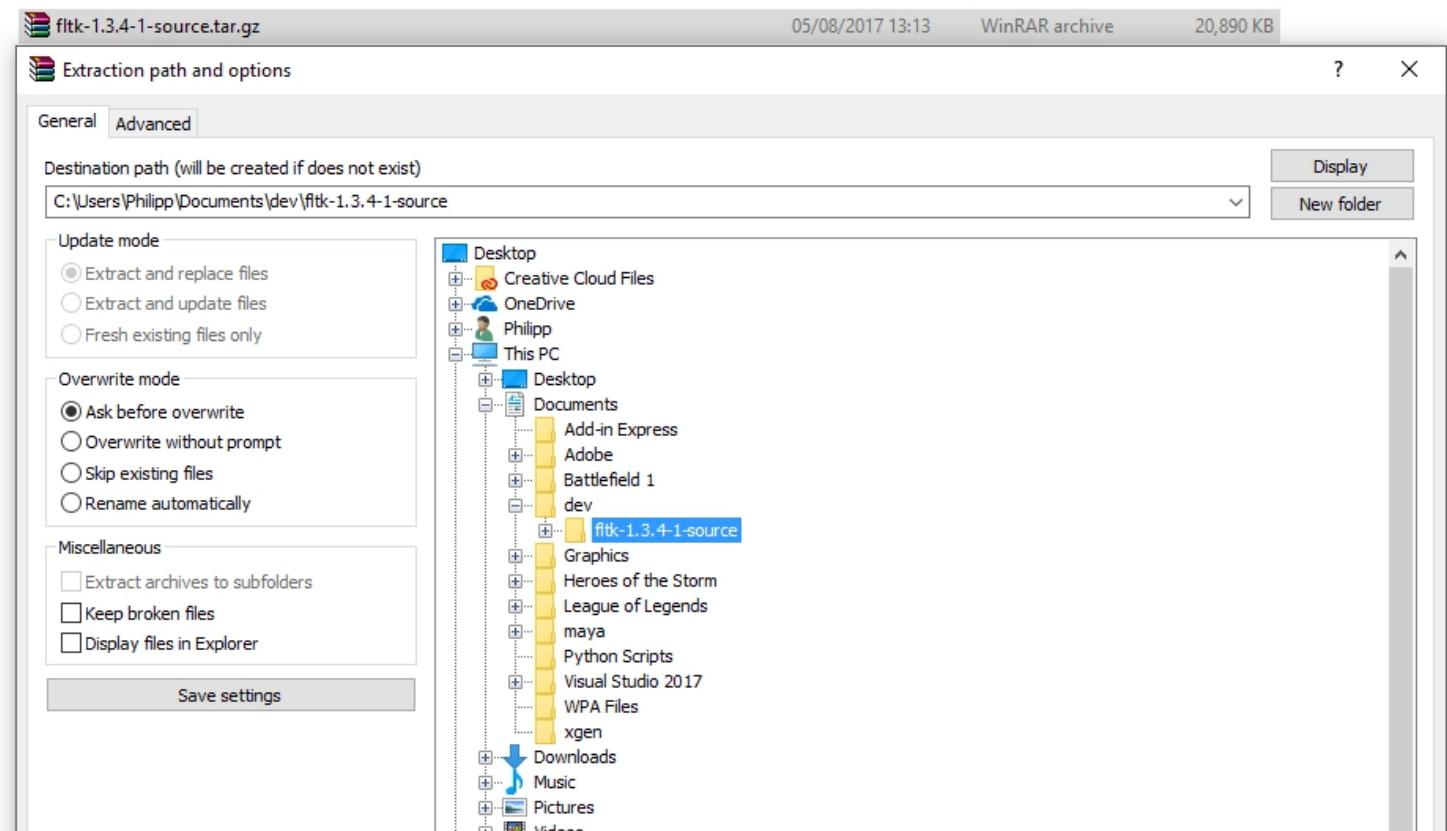
The **fltk-1.3.4-1-source.tar.gz** should be automatically downloaded.

The screenshot shows the FLTK Download page with a message indicating the file is being downloaded. The status bar at the bottom shows the download progress: "fltk-1.3.4-1-sour...tar.gz 100%".

The screenshot shows the FLTK Download page. At the top, there's a browser header with the URL www.fltk.org/software.php?VERSION=1.3.4&FILE=fltk/1.3.4/fltk-1.3.4-1-source.tar.gz. Below the header, the FLTK logo is on the left, and the word "Download" is prominently displayed. A navigation bar at the top right includes links for "FLTK Apps", "FLTK Library", "Forums", "Links", "Log", "[Home | Art]", and "[v1.3.4]". A message below the navigation bar says, "Your download should begin shortly. If not, please [click here](#) to download the file from the current location." In the center, it says "You are downloading: **fltk-1.3.4-1-source.tar.gz**". At the bottom, a note states, "Comments are owned by the poster. All other content is copyright 1998-2015 by Bill Spitzak and others. This project is released under the MIT license." The entire page has a light blue background.

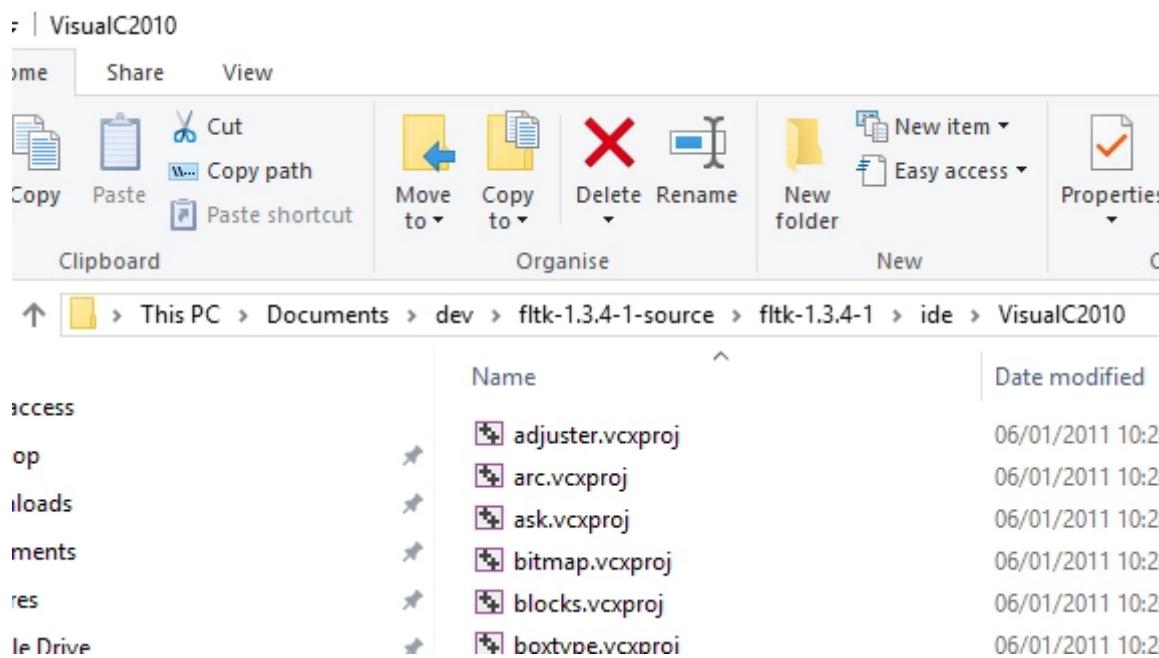
I used [winRar](http://www.win-rar.com/start.html?&L=0) (<http://www.win-rar.com/start.html?&L=0>) to extract **fltk-1.3.4-1-source.tar.gz** to my documents folder.

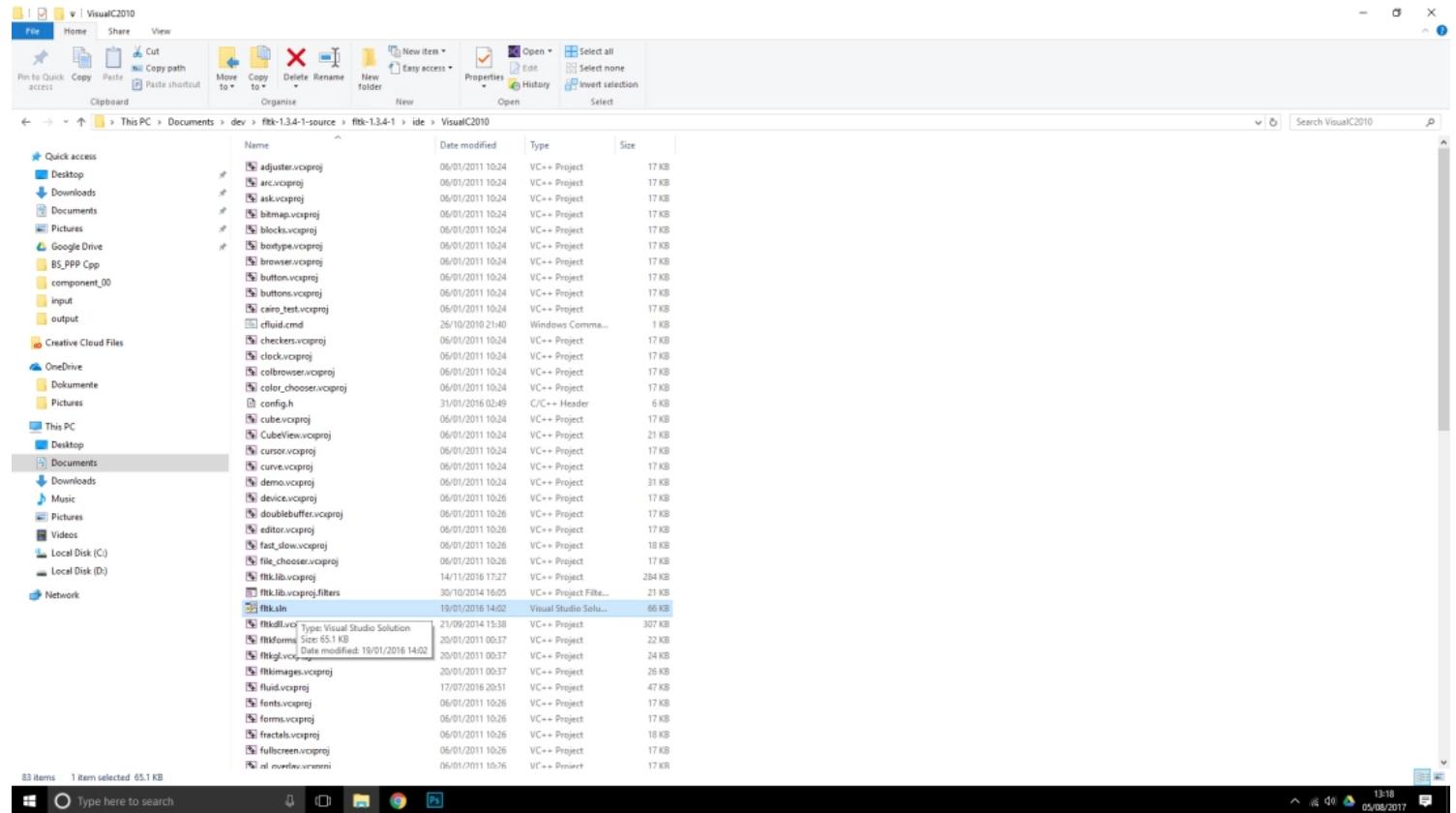




Step 2: Compiling and building fltk-1.3.4 from source.

Go to your extracted **fltk-1.3.4 folder**. In my case `C:\Users\Philipp\Documents\dev\fltk-1.3.4-1-source\fltk-1.3.4-1\ide\VisualC2010`

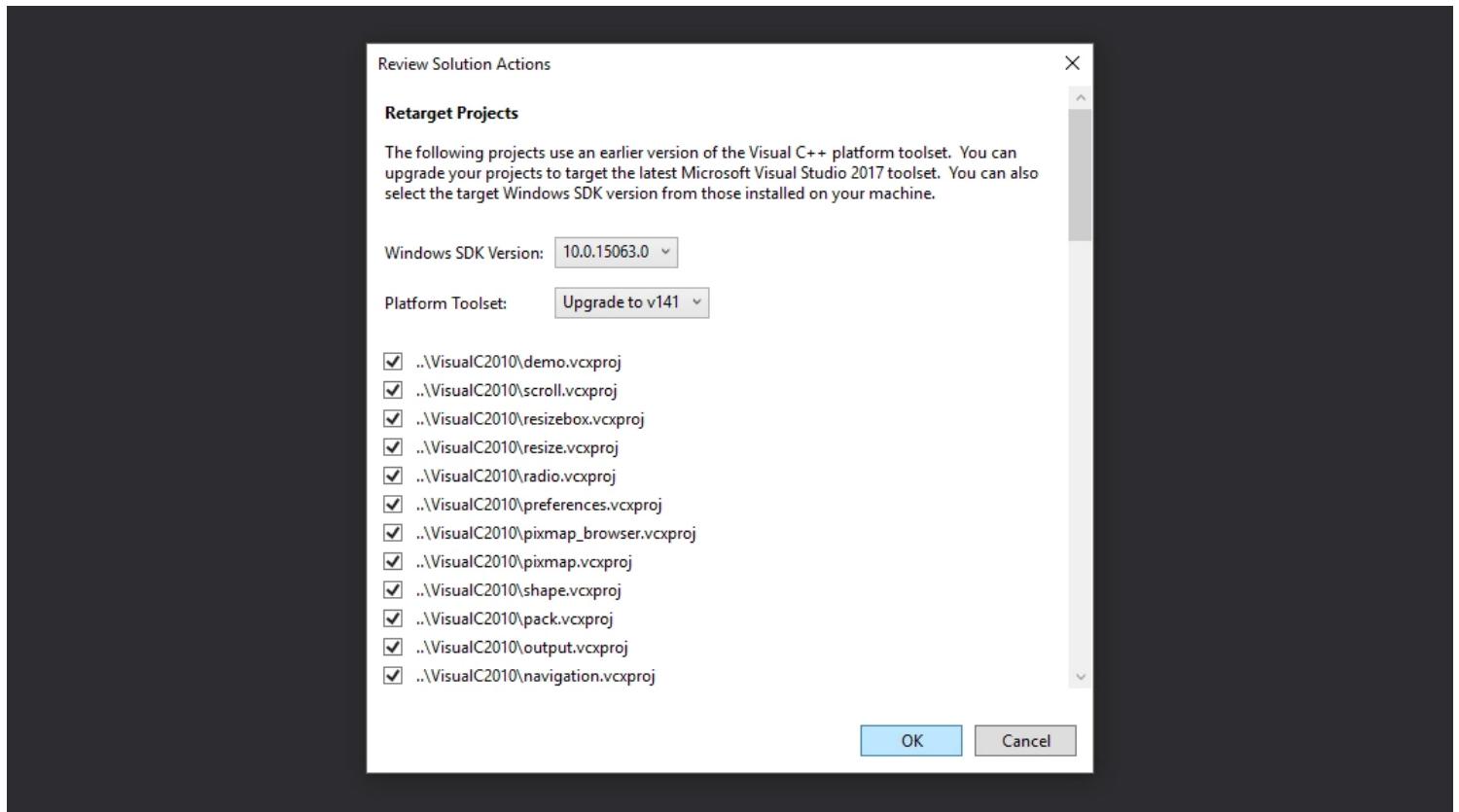
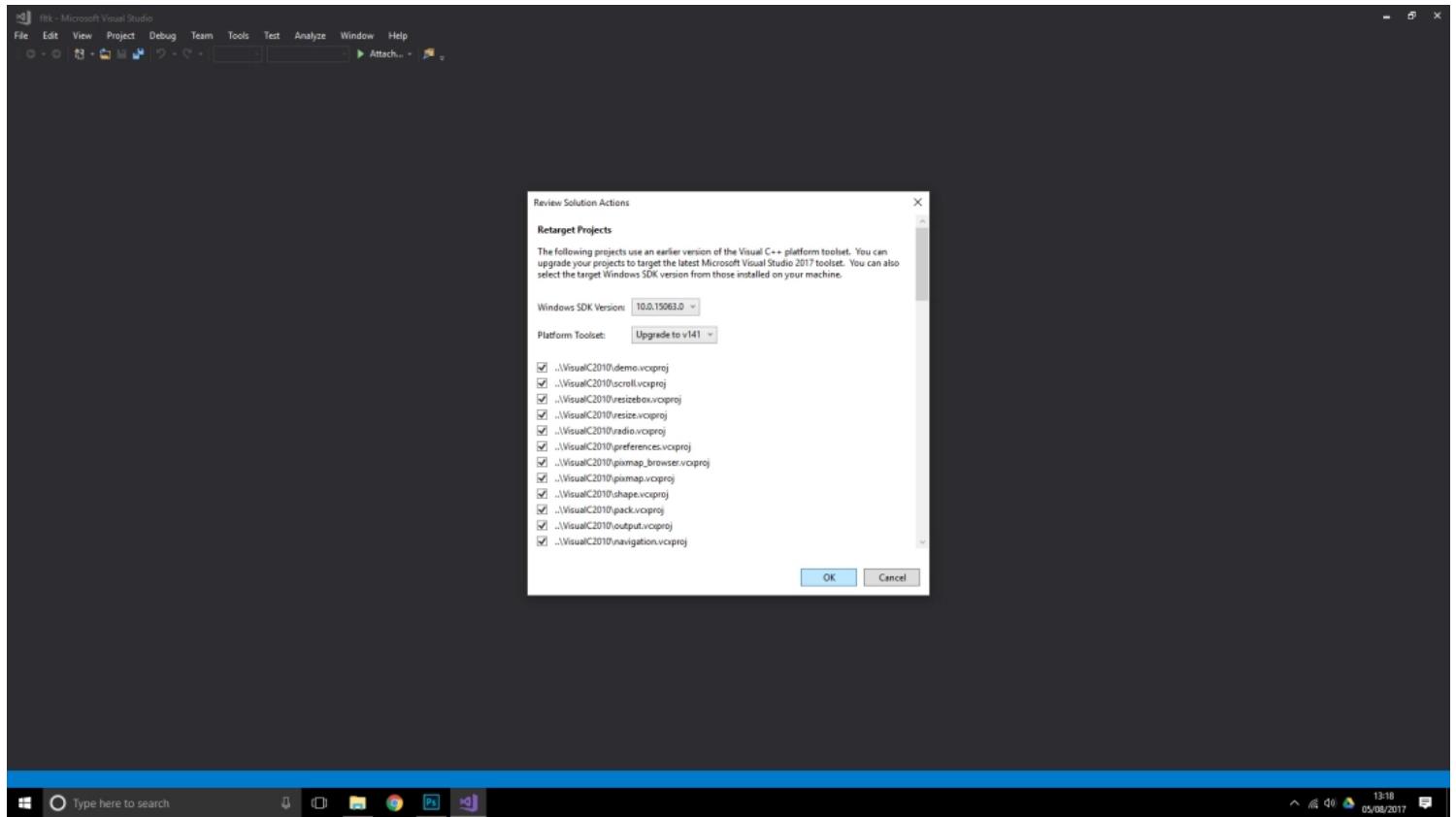




Open (double-click) fltk.sln.

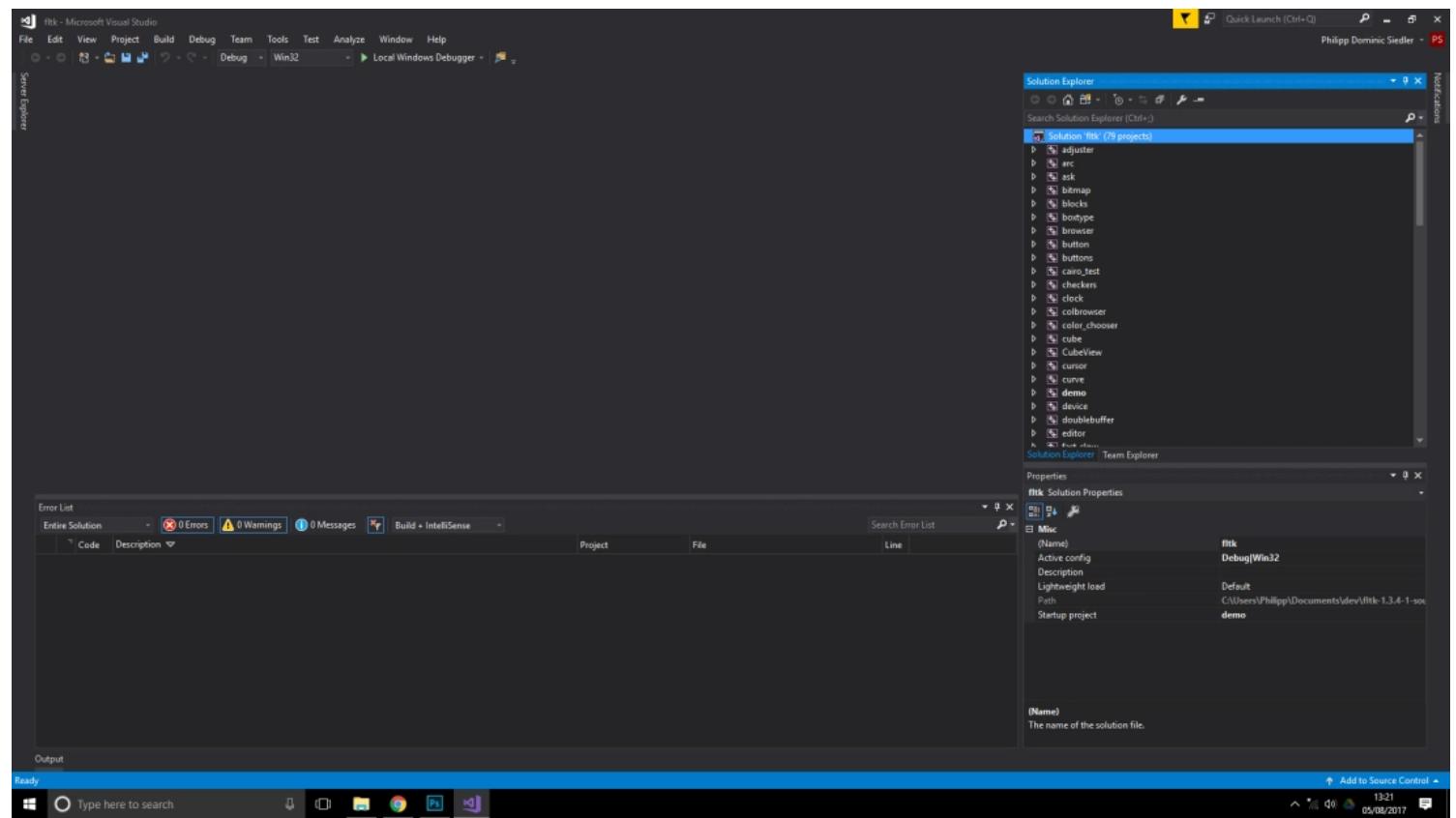
demo.vcxproj	06/01/2011 10:24	VC++ Project	31 KB
device.vcxproj	06/01/2011 10:26	VC++ Project	17 KB
doublebuffer.vcxproj	06/01/2011 10:26	VC++ Project	17 KB
editor.vcxproj	06/01/2011 10:26	VC++ Project	17 KB
fast_slow.vcxproj	06/01/2011 10:26	VC++ Project	18 KB
file_chooser.vcxproj	06/01/2011 10:26	VC++ Project	17 KB
fltk.lib.vcxproj	14/11/2016 17:27	VC++ Project	284 KB
fltk.lib.vcxproj.filters	30/10/2014 16:05	VC++ Project File...	21 KB
fltk.sln	19/01/2016 14:02	Visual Studio Solu...	66 KB
fltkd.dll.vcxproj	21/09/2014 15:38	VC++ Project	307 KB
fltkforms.vcxproj	20/01/2011 00:37	VC++ Project	22 KB
fltkgl.vcxproj	20/01/2011 00:37	VC++ Project	24 KB
fltkimages.vcxproj	20/01/2011 00:37	VC++ Project	26 KB
fluid.vcxproj	17/07/2016 20:51	VC++ Project	47 KB
fonts.vcxproj	06/01/2011 10:26	VC++ Project	17 KB
forms.vcxproj	06/01/2011 10:26	VC++ Project	17 KB
fractals.vcxproj	06/01/2011 10:26	VC++ Project	18 KB
fullscreen.vcxproj	06/01/2011 10:26	VC++ Project	17 KB
niflue.vcxproj	06/01/2011 10:26	VC++ Project	17 KB

Visual Studio 2017 Community will open, and ask you if you want to update the files.

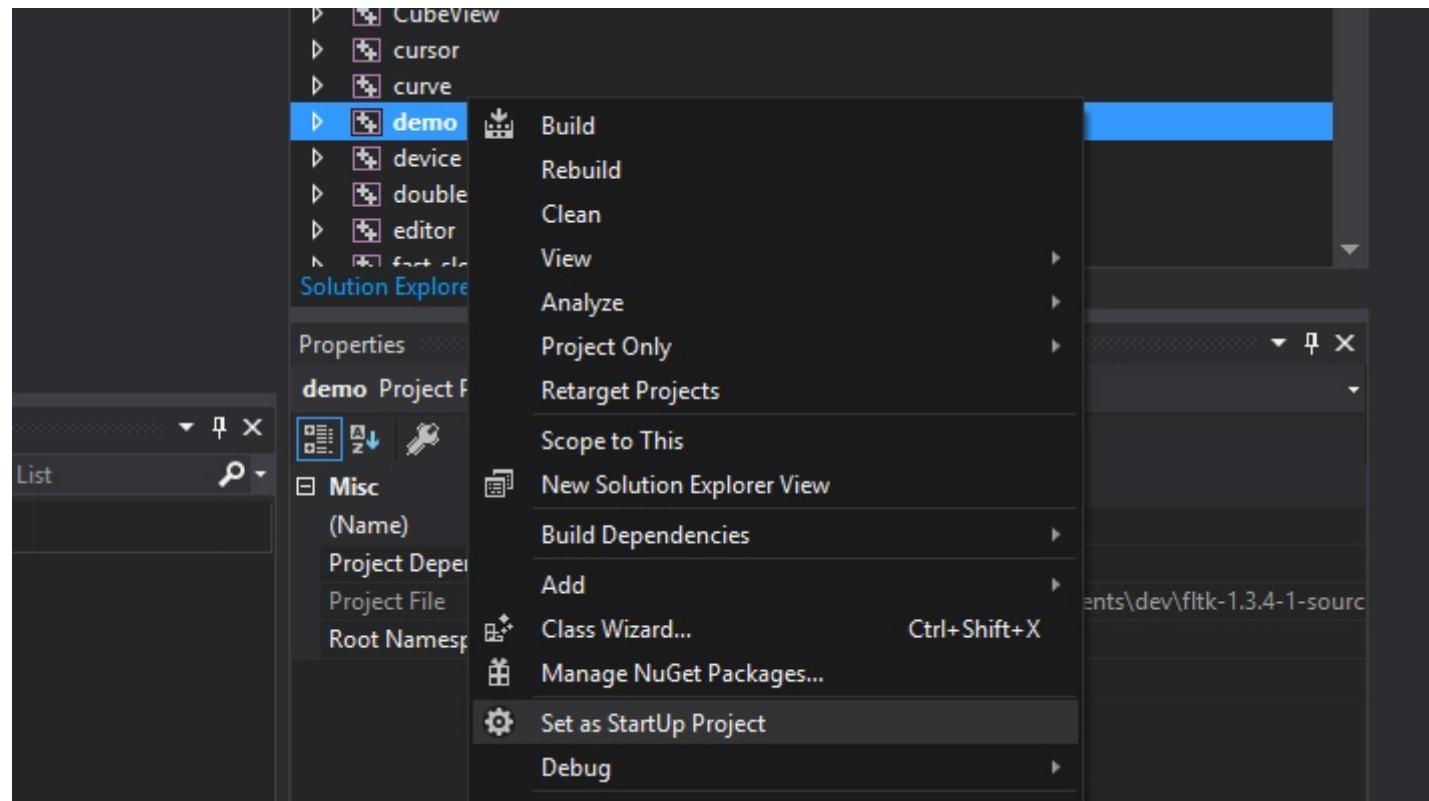


Click **OK** and wait for a few moments.

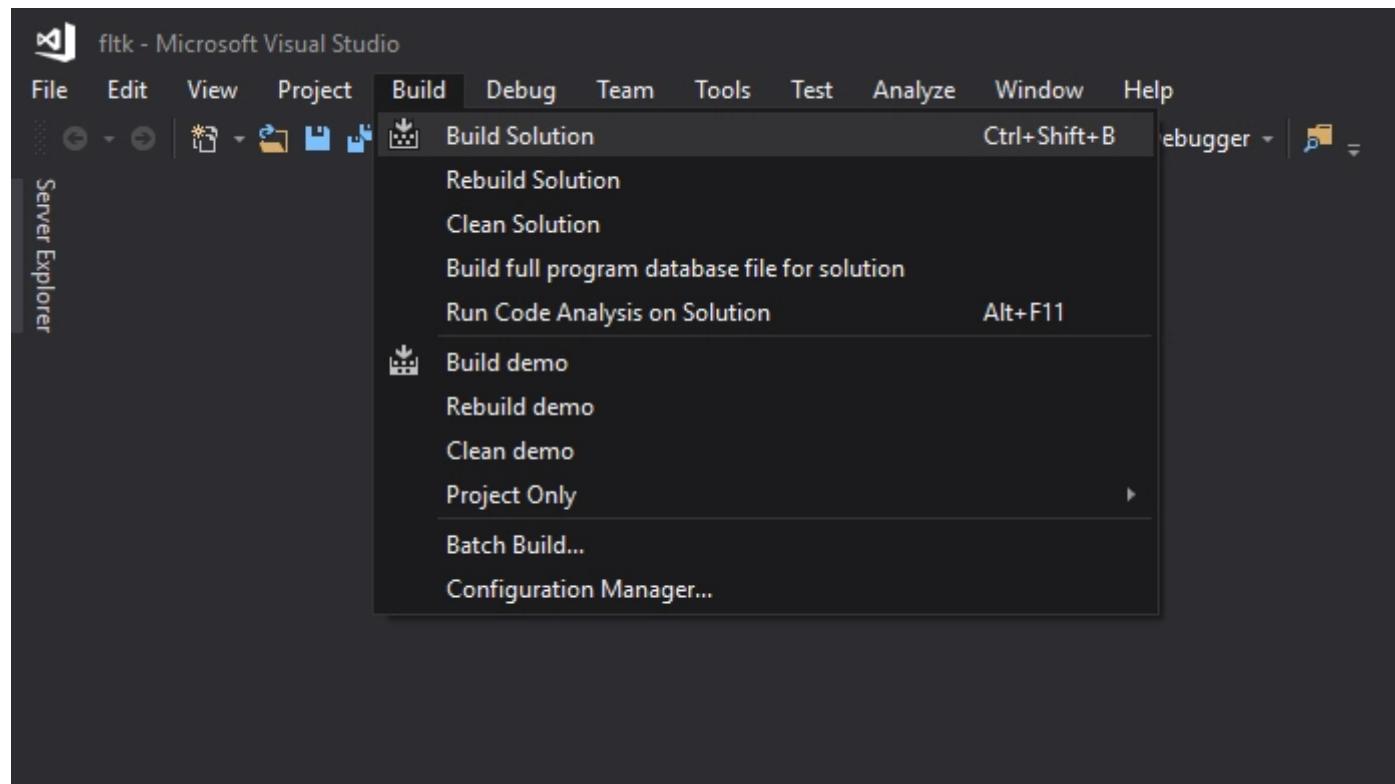
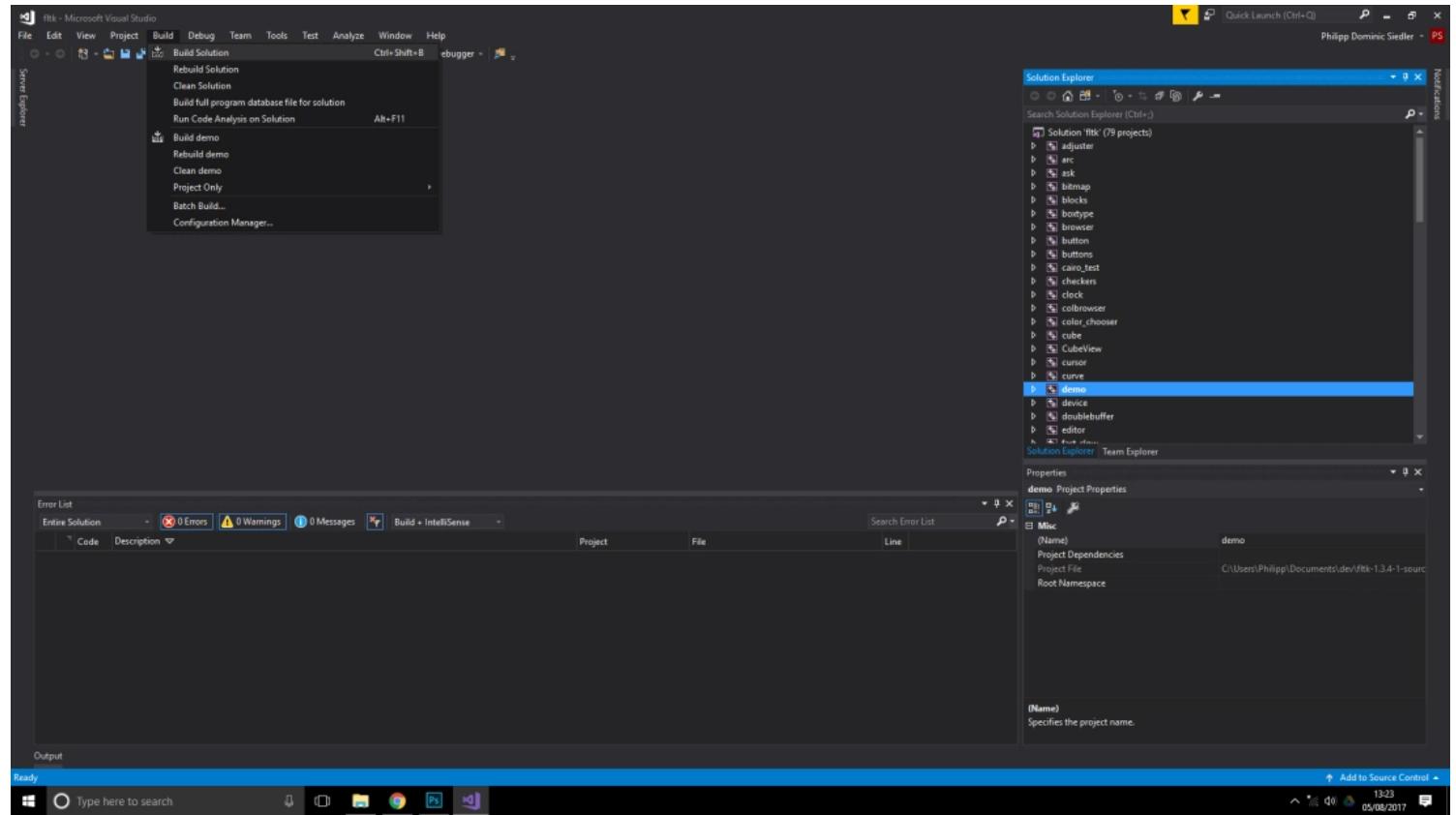
Your window will look like this:



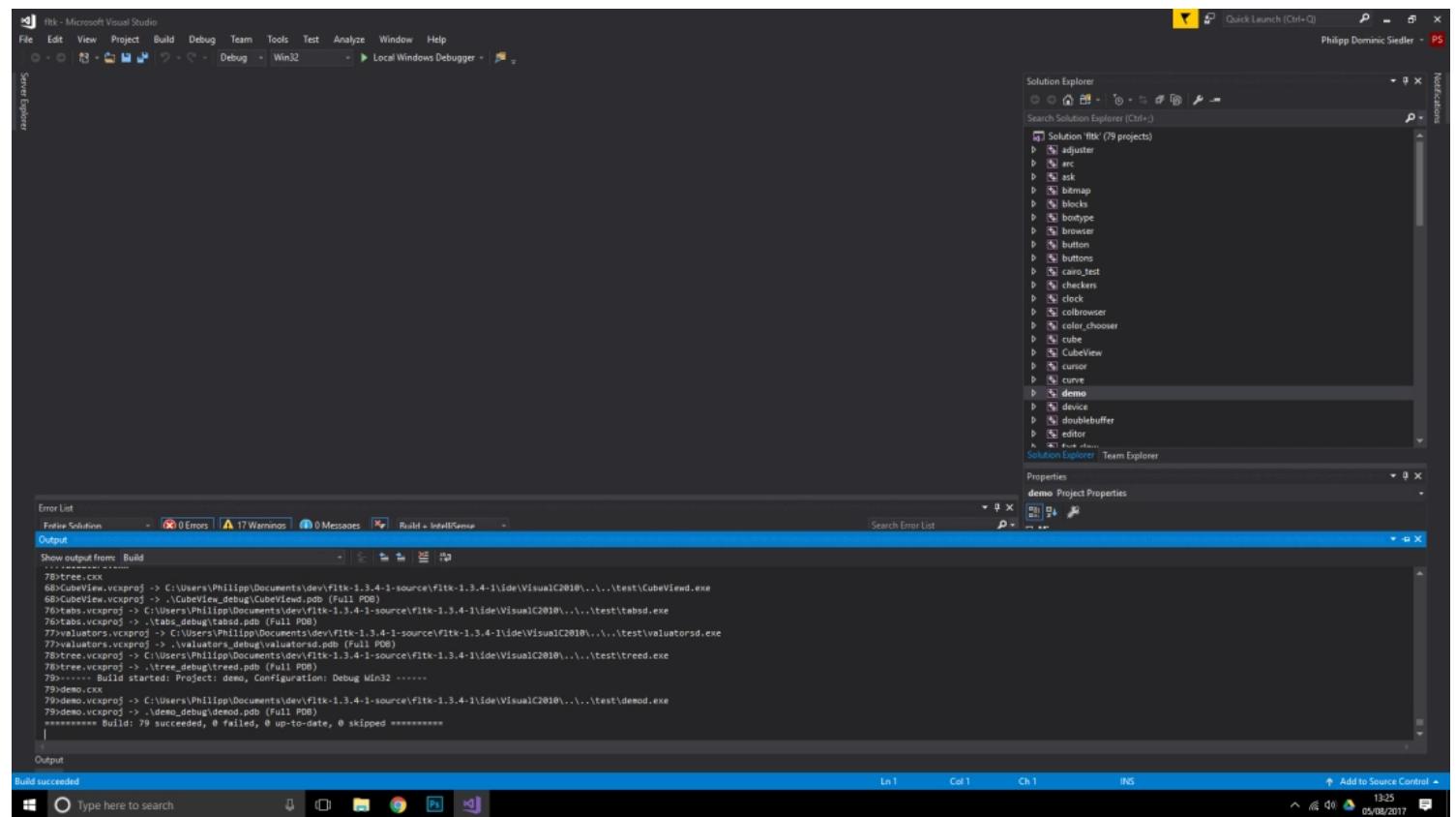
Right-click the **demo** solution file.
Click on **Set as StartUp Project**.



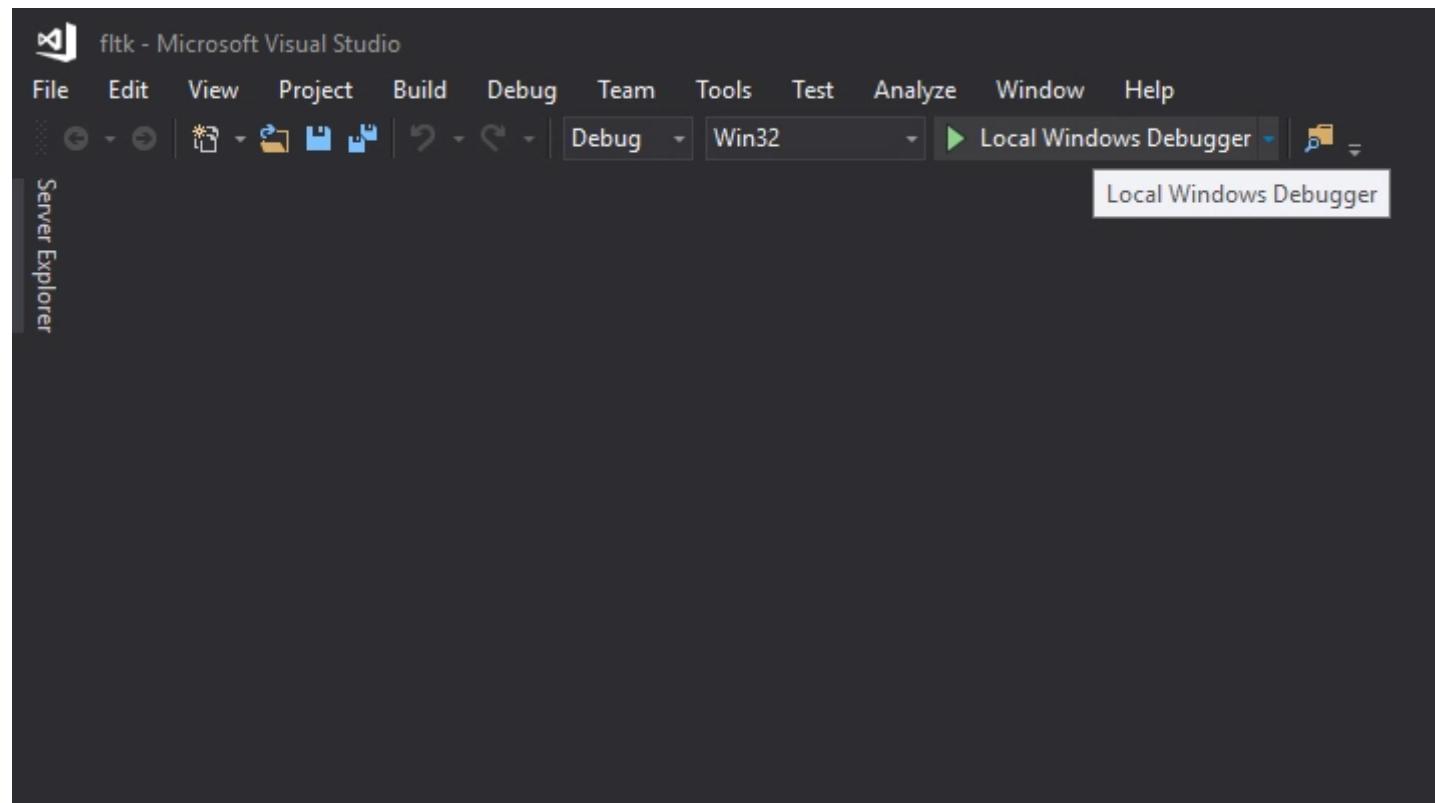
Now click on **Build** and **Build Solution**.



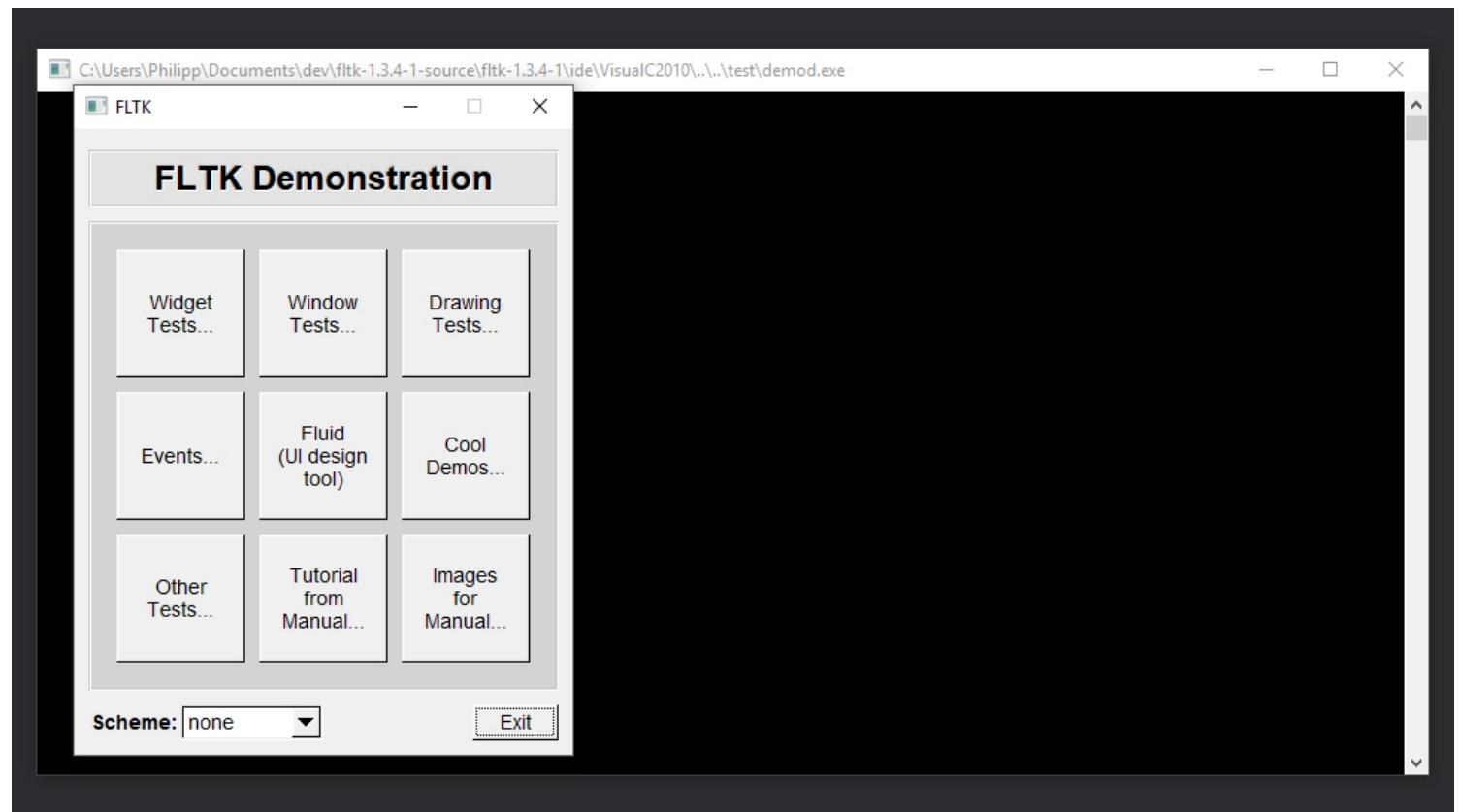
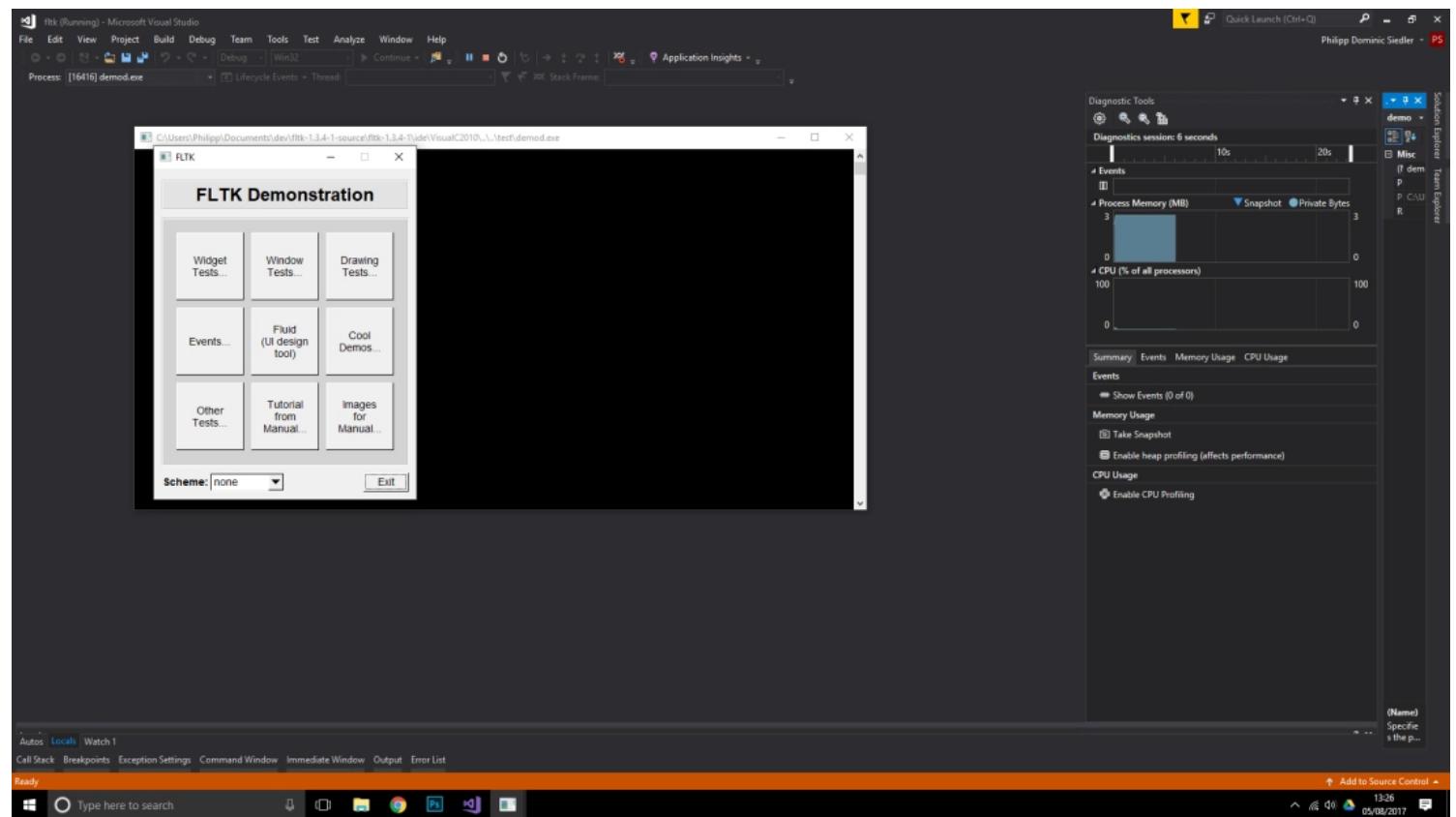
After a couple of moments your **Output** window shows **Build: 79 succeeded...**



Make sure to set **Debug** mode.

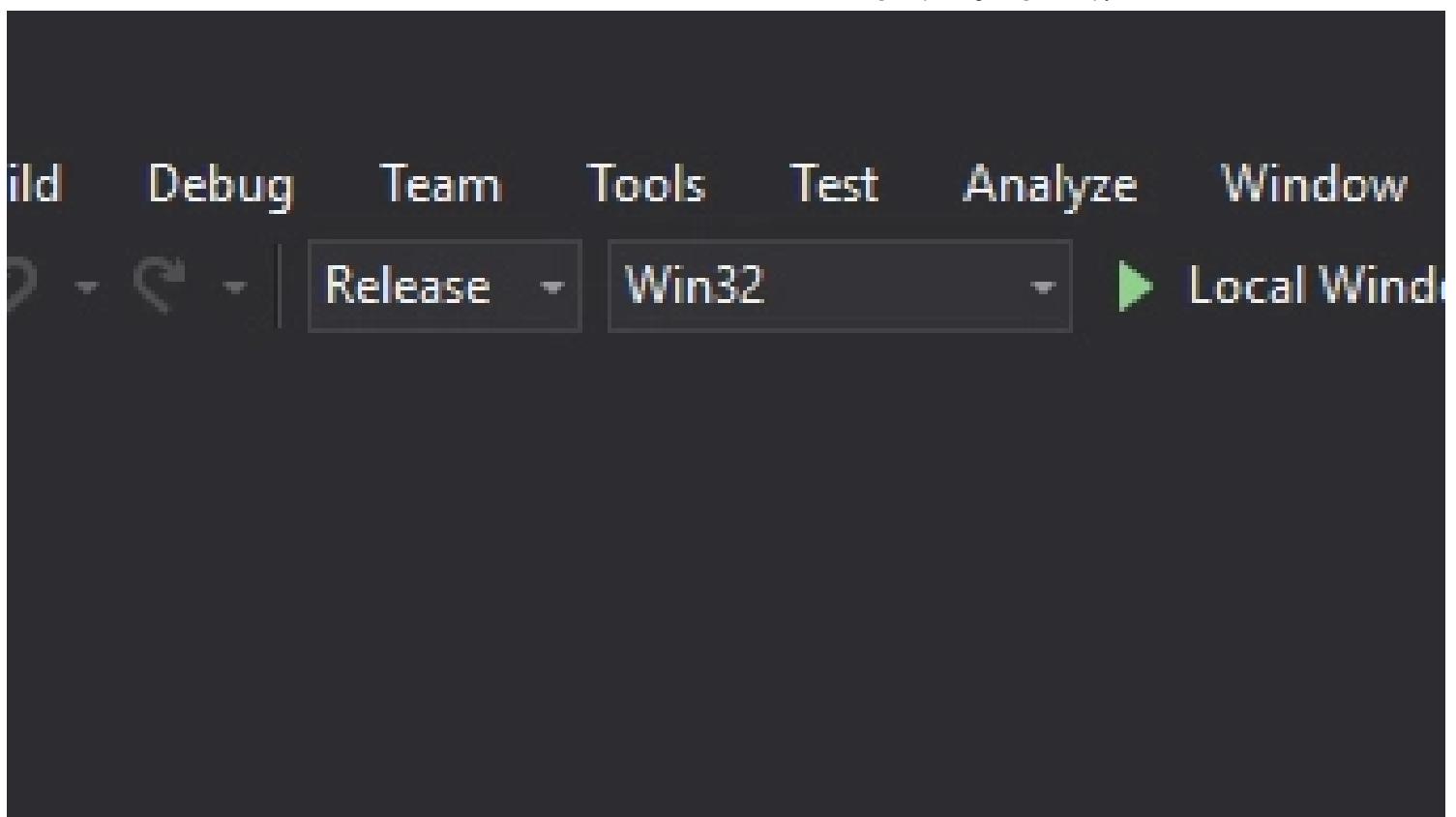


Press **F5** on your keyboard. Or the playbutton on top of your canvas **Local Windows Debugger**.



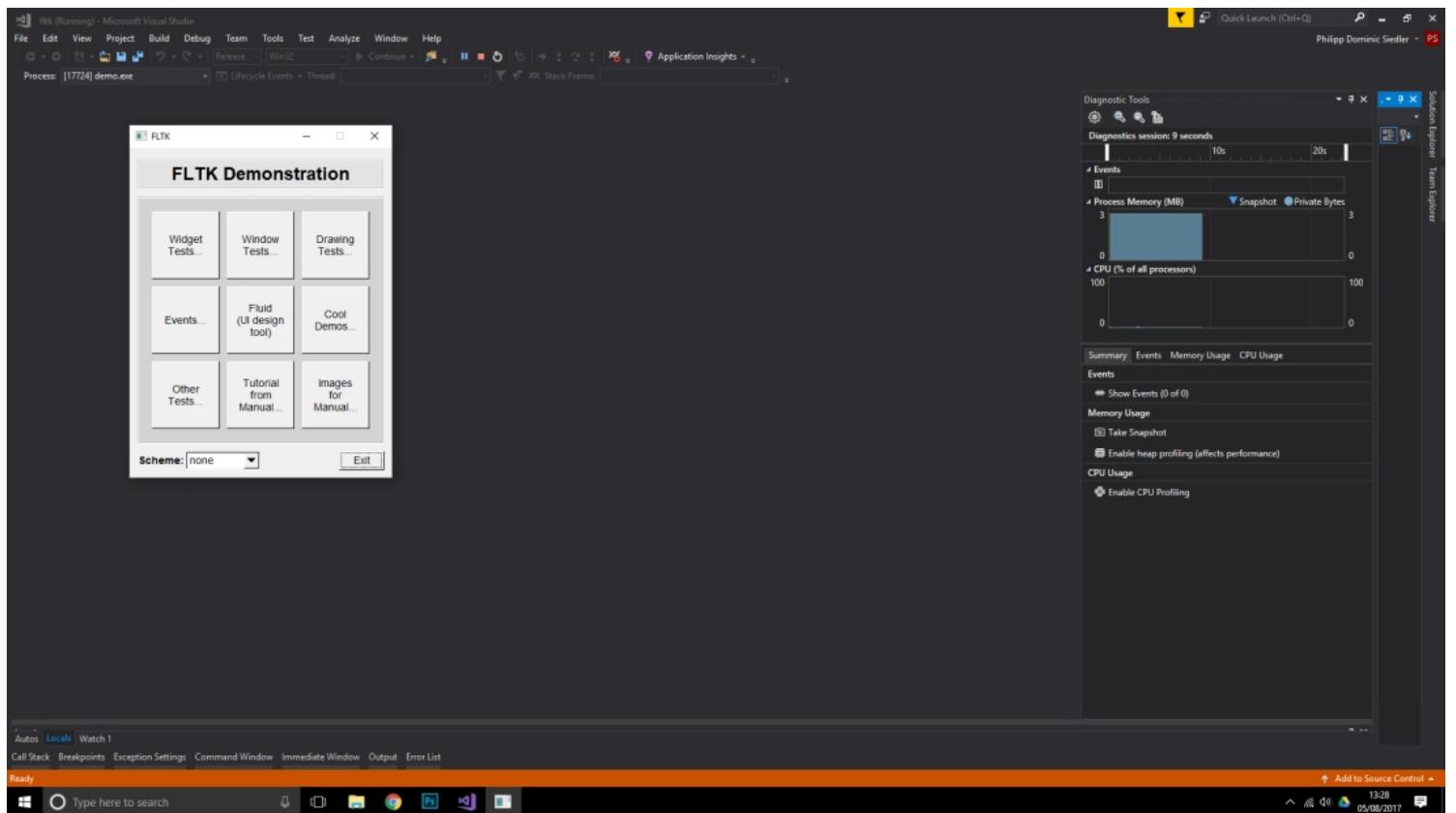
So far so good. You should get two windows, one **console window** and another one showing the **fltk-test layout**.

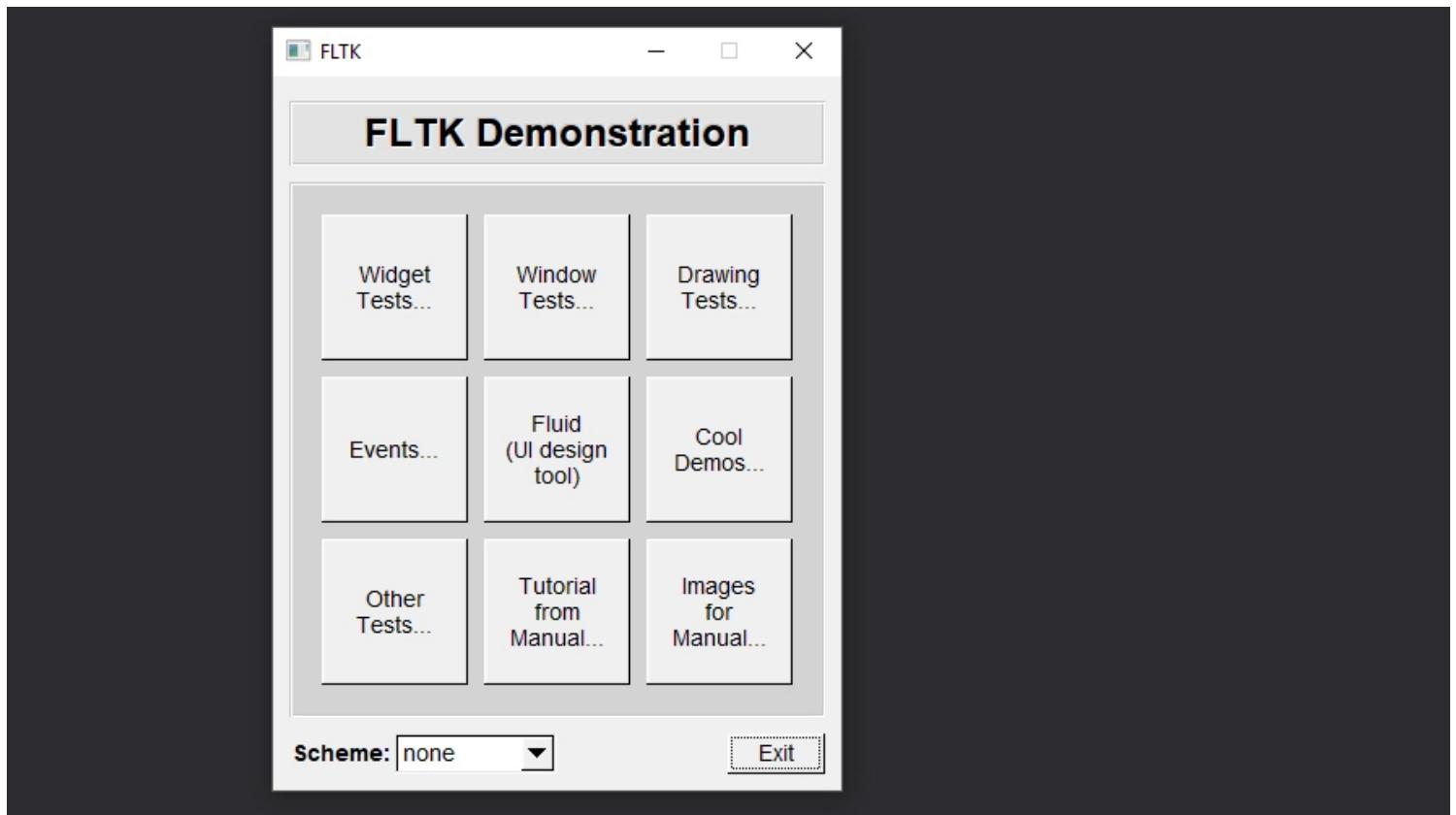
Now change the **Debug** mode to **Release** mode on the top left corner of your Visual Studio window.



Again: Press **F5** on your keyboard. Or the playbutton on top of your canvas **Local Windows Debugger** to run your project.

You should now see the **fltk-test** layout and no **console window**.



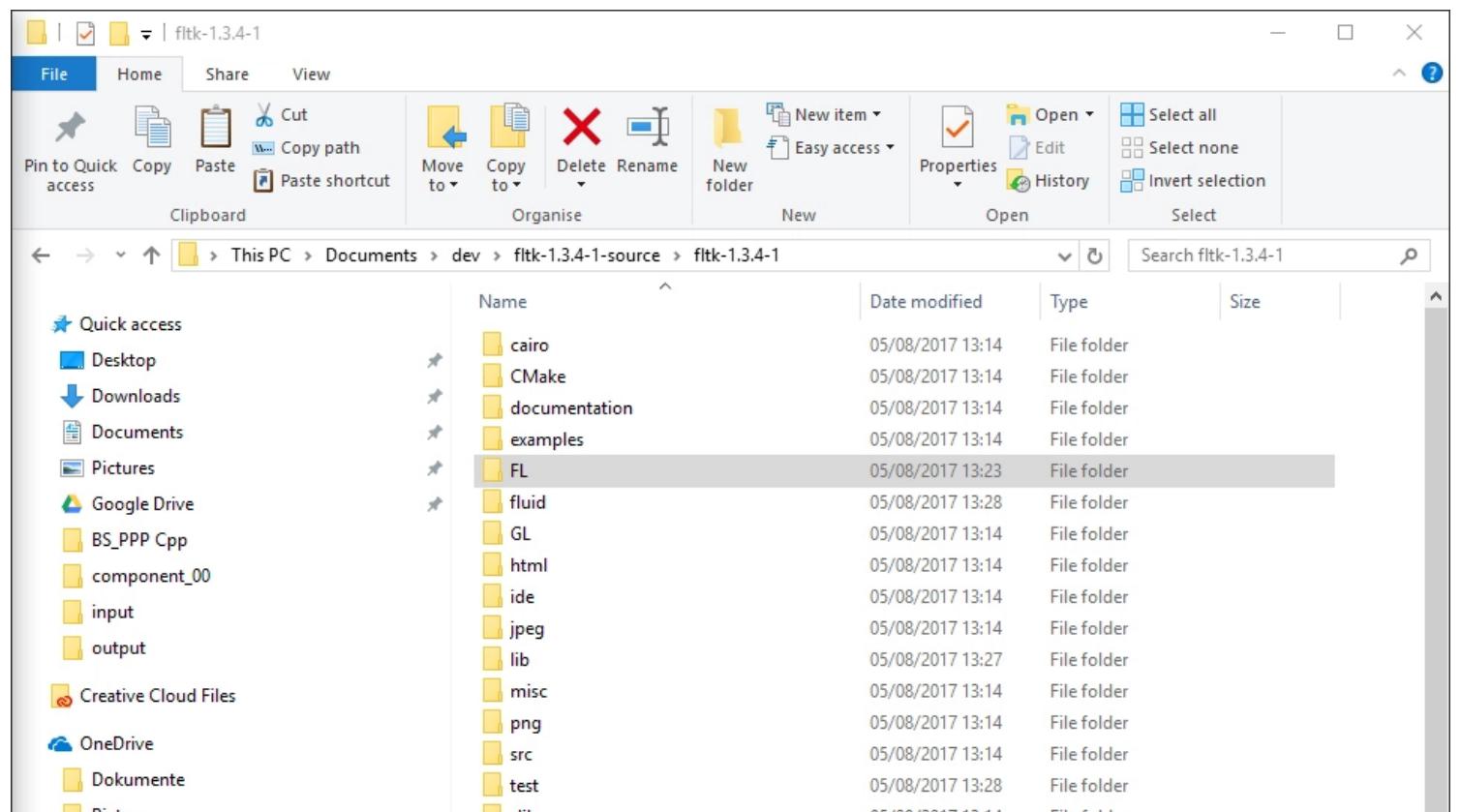
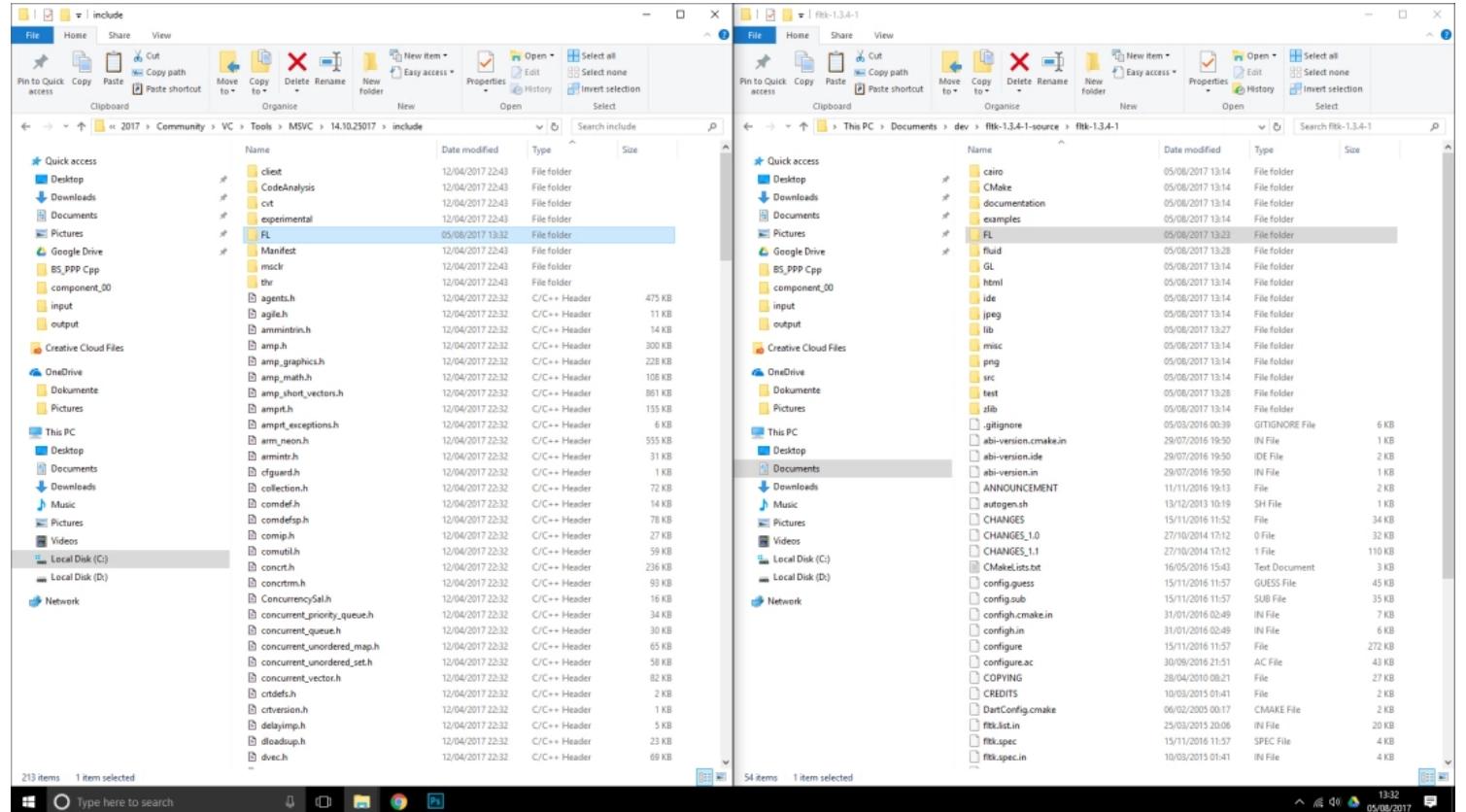


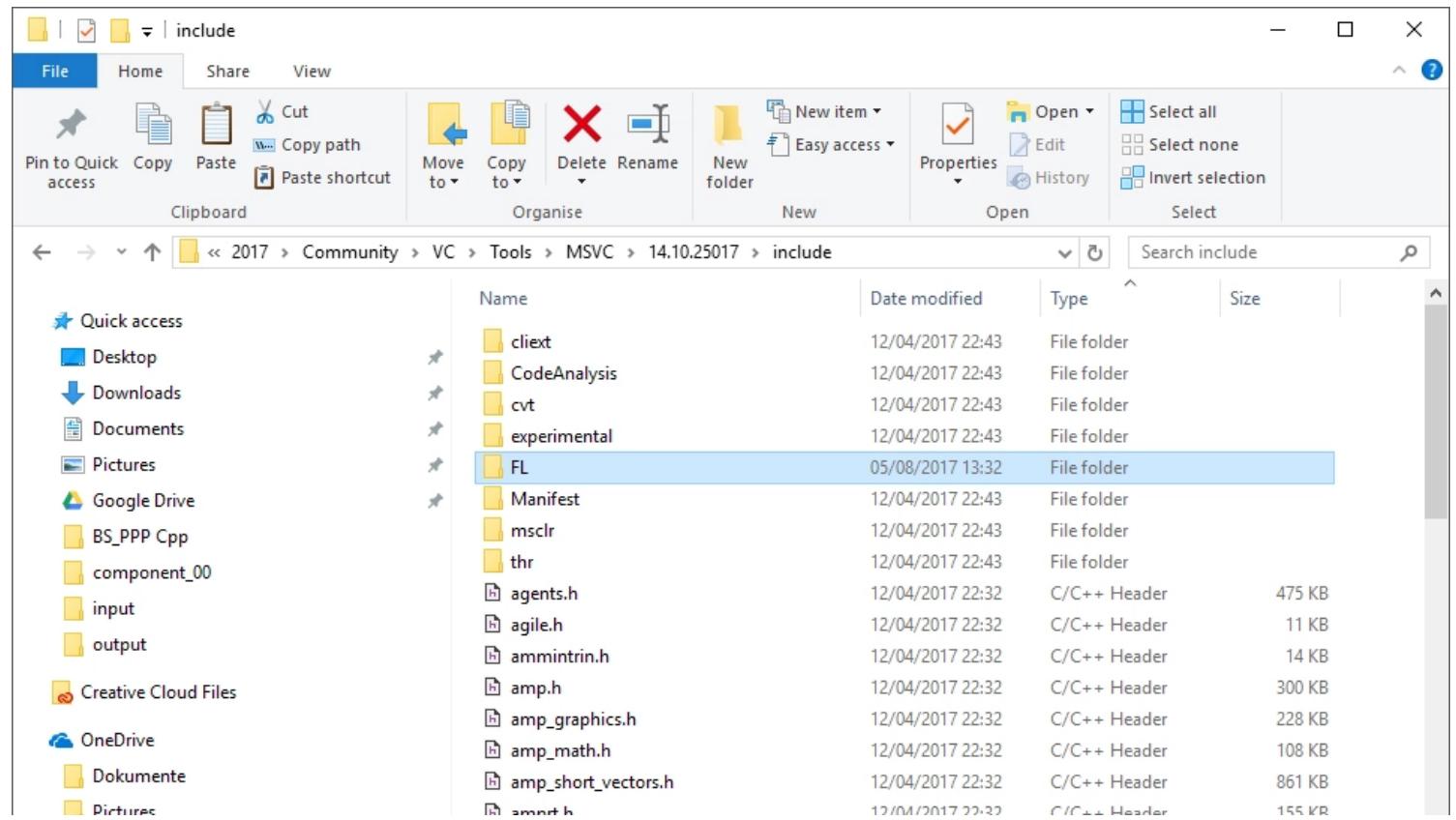
Good job. Done compiling and test-running.

Step 3: Integrating fltk-1.3.4 files into your Visual Studio 2017 Community folder structure.

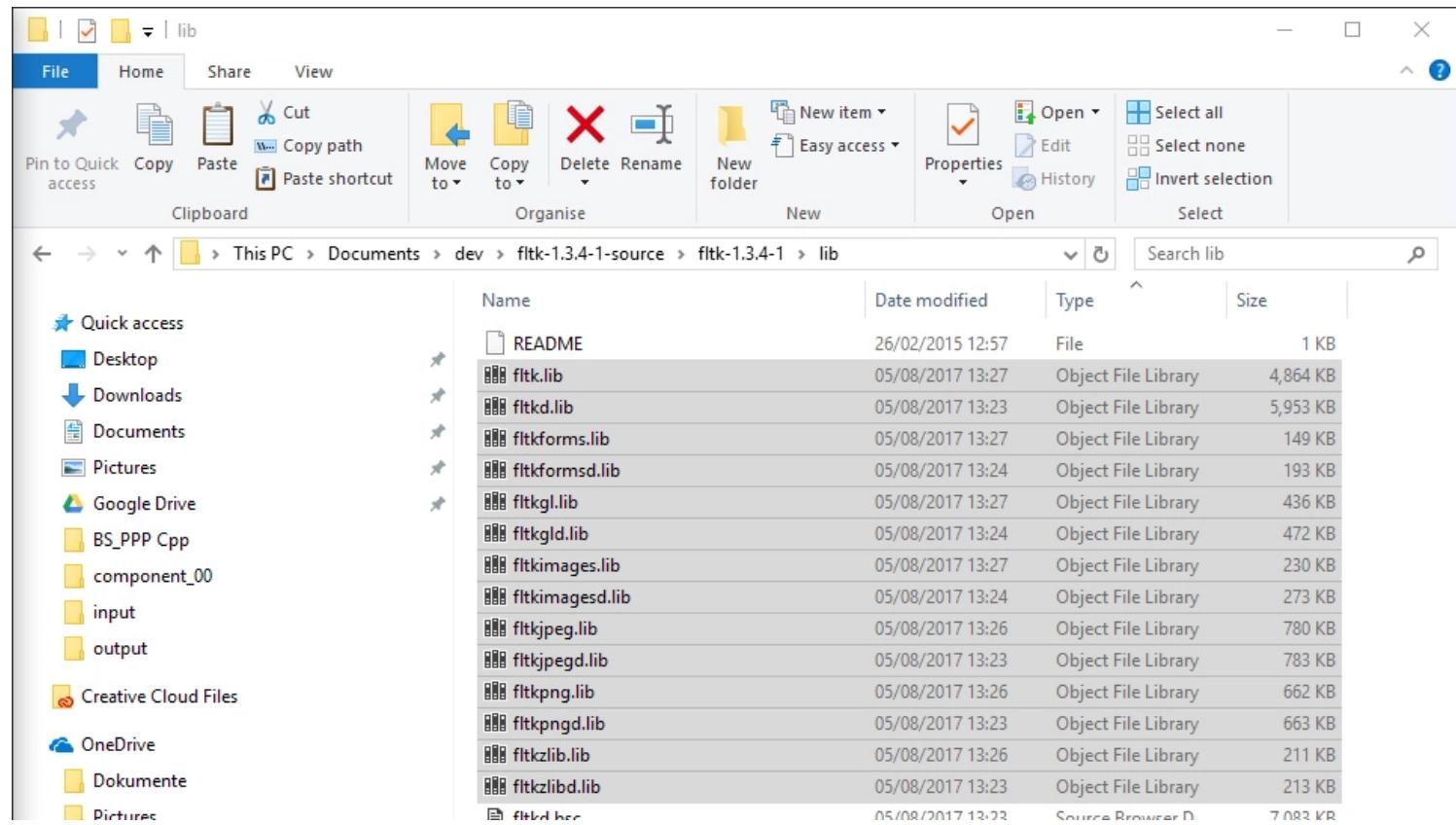
Now you need to copy those freshly compiled files (explained in three steps in the next paragraph) into your Visual Studio 2017 Community folder structure so it knows where to find them:

1. Go to: your **fltk-1.3.4-1** folder (in my case: `C:\Users\Philipp\Documents\dev\fltk-1.3.4-1-source\fltk-1.3.4-1`) and copy the `Fl` folder into your **include** folder (in my case): `C:\Program Files (x86)\Microsoft Visual Studio\2017\Community\VC\Tools\MSVC\14.10.25017\include`

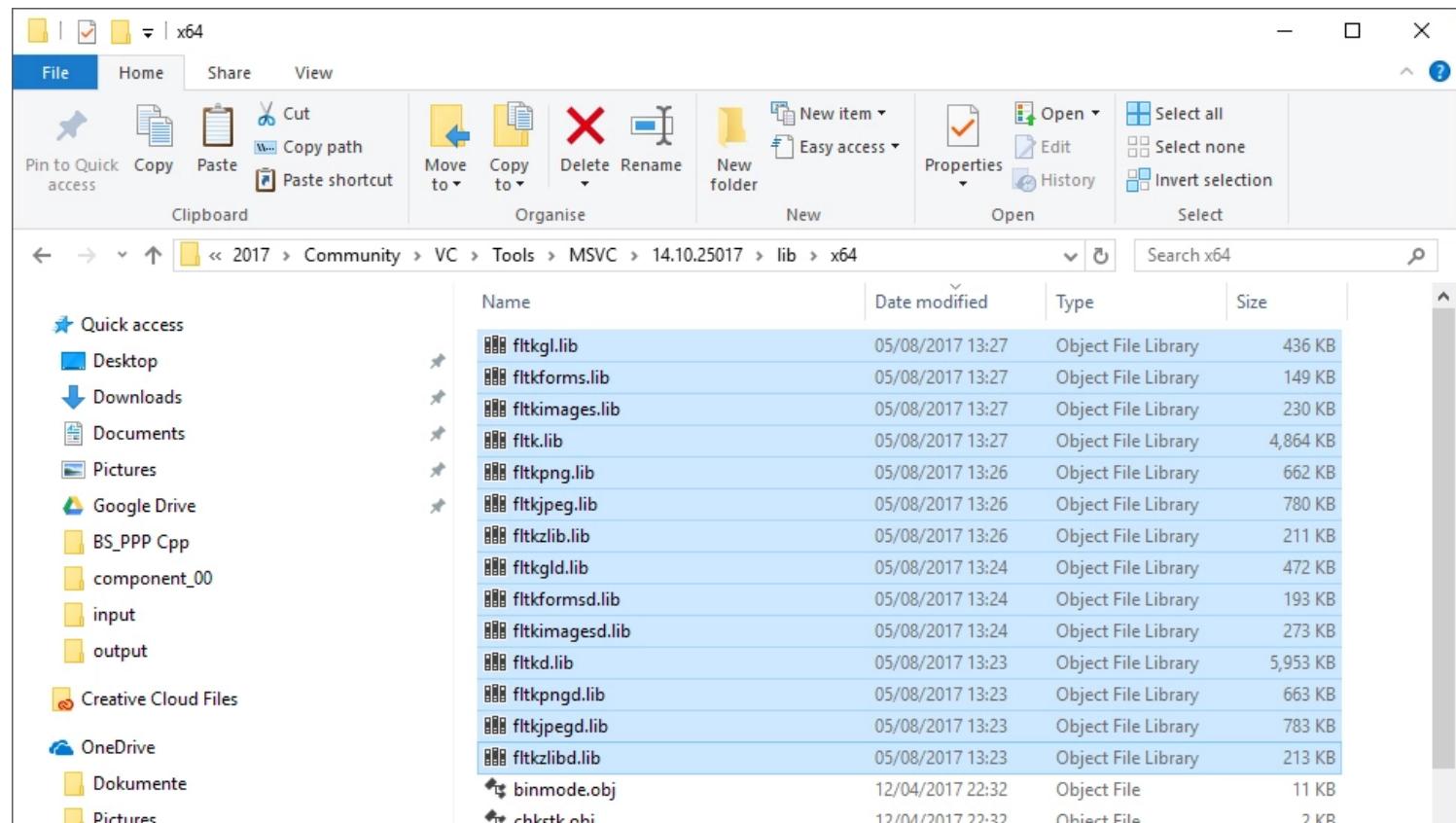




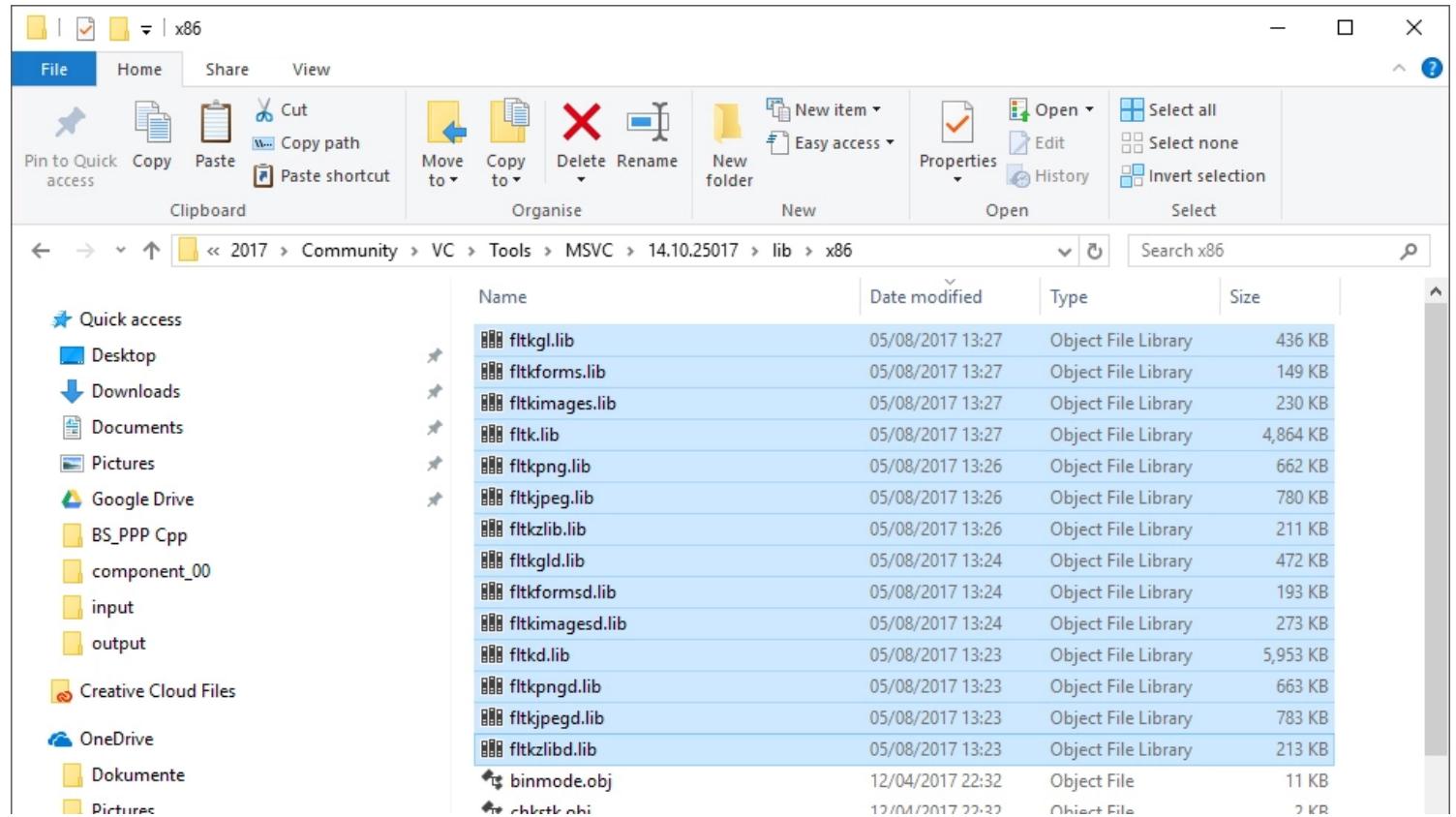
2. Go to: the **lib** folder in your fltk source folder (in my case **C:\Users\Philipp\Documents\dev\fltk-1.3.4-1-source\fltk-1.3.4-1\lib**) and copy all 14 .lib files (Note that there are pairs of 2, one of them named with a "d" for debug in the end). And paste them into your Visual Studio 2017 lib folder (in my case: **C:\Program Files (x86)\Microsoft Visual Studio\2017\Community\VC\Tools\MSVC\14.10.25017\lib\x64**). Of course you noticed that in the lib folder there are two more folders: **x64** and **x86**. I just pasted the 14.lib files into both of them (In case there is someone out there who can tell me if that is wrong or right, please contact me).



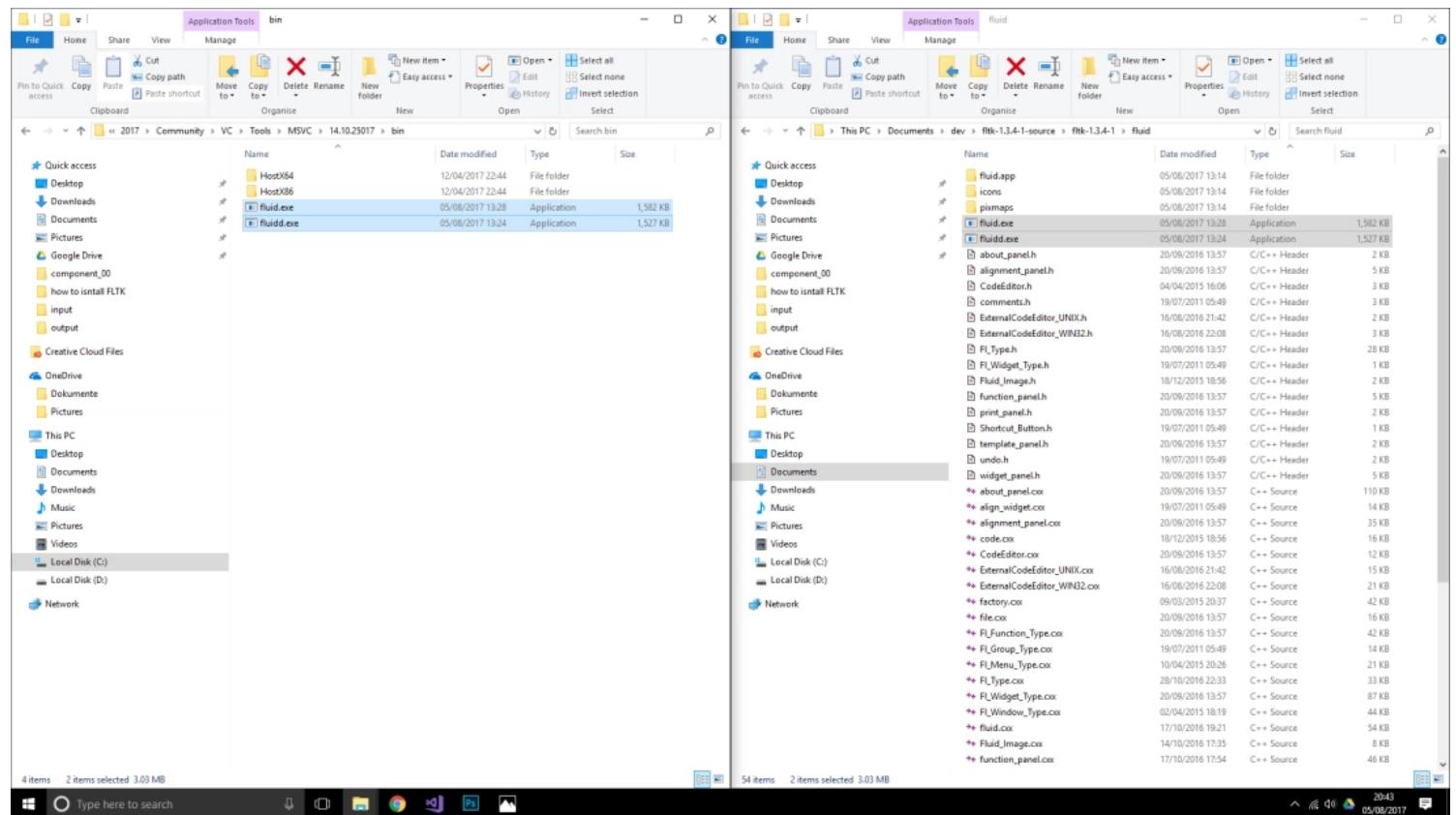
fltk-1.3.4-1\lib to VC\Tools\MSVC\14.10.25017\lib\x64



fltk-1.3.4-1\lib to VC\Tools\MSVC\14.10.25017\lib\x86



3. Go to: the fluid folder in your fltk source folder (in my case: C:\Users\Philipp\Documents\dev\fltk-1.3.4-1-source\fltk-1.3.4-1\fluid) and copy fluid.exe and fluidd.exe into your Visual Studio 2017 bin folder (in my case: C:\Program Files (x86)\Microsoft Visual Studio\2017\Community\VC\Tools\MSVC\14.10.25017\bin).

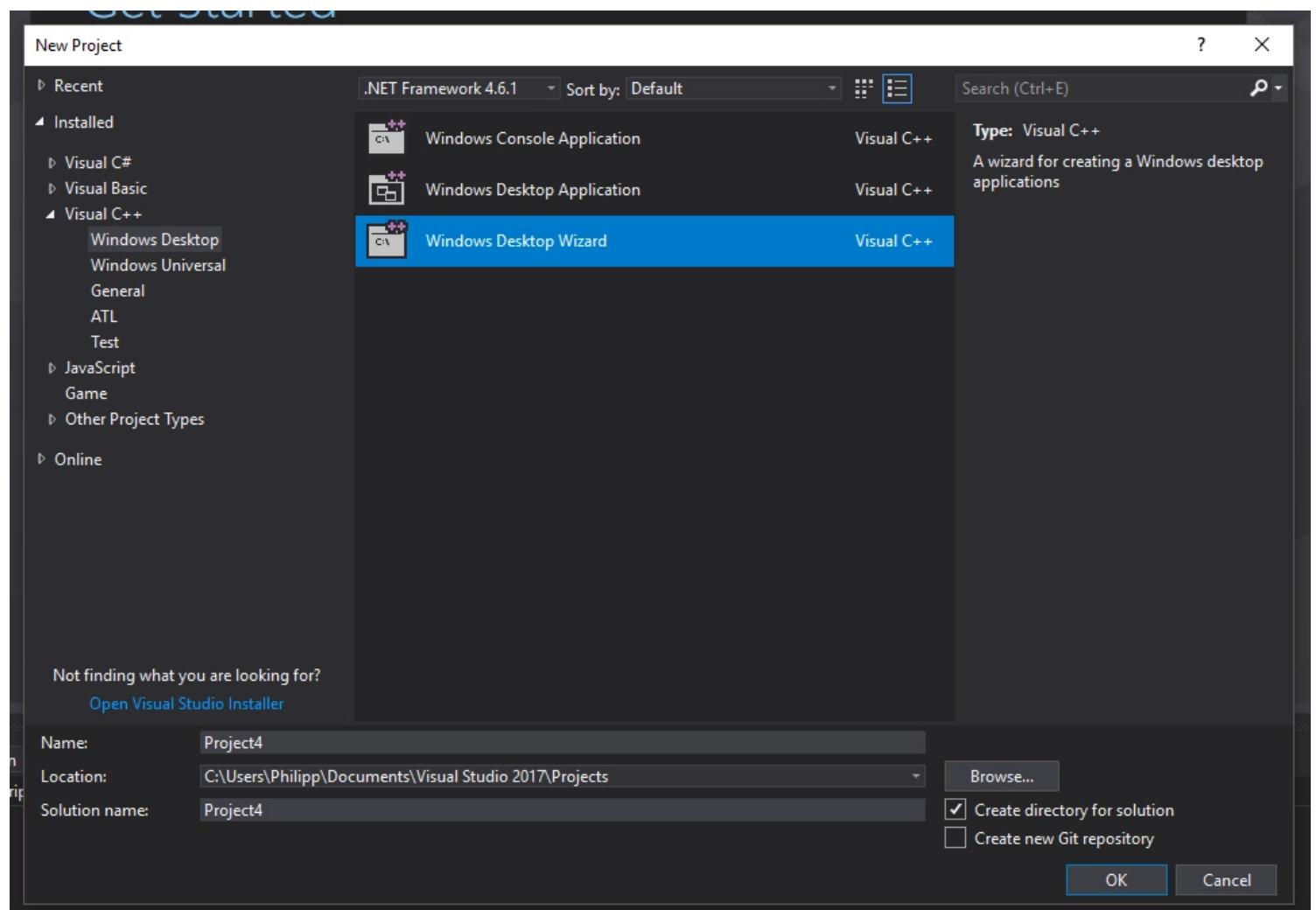
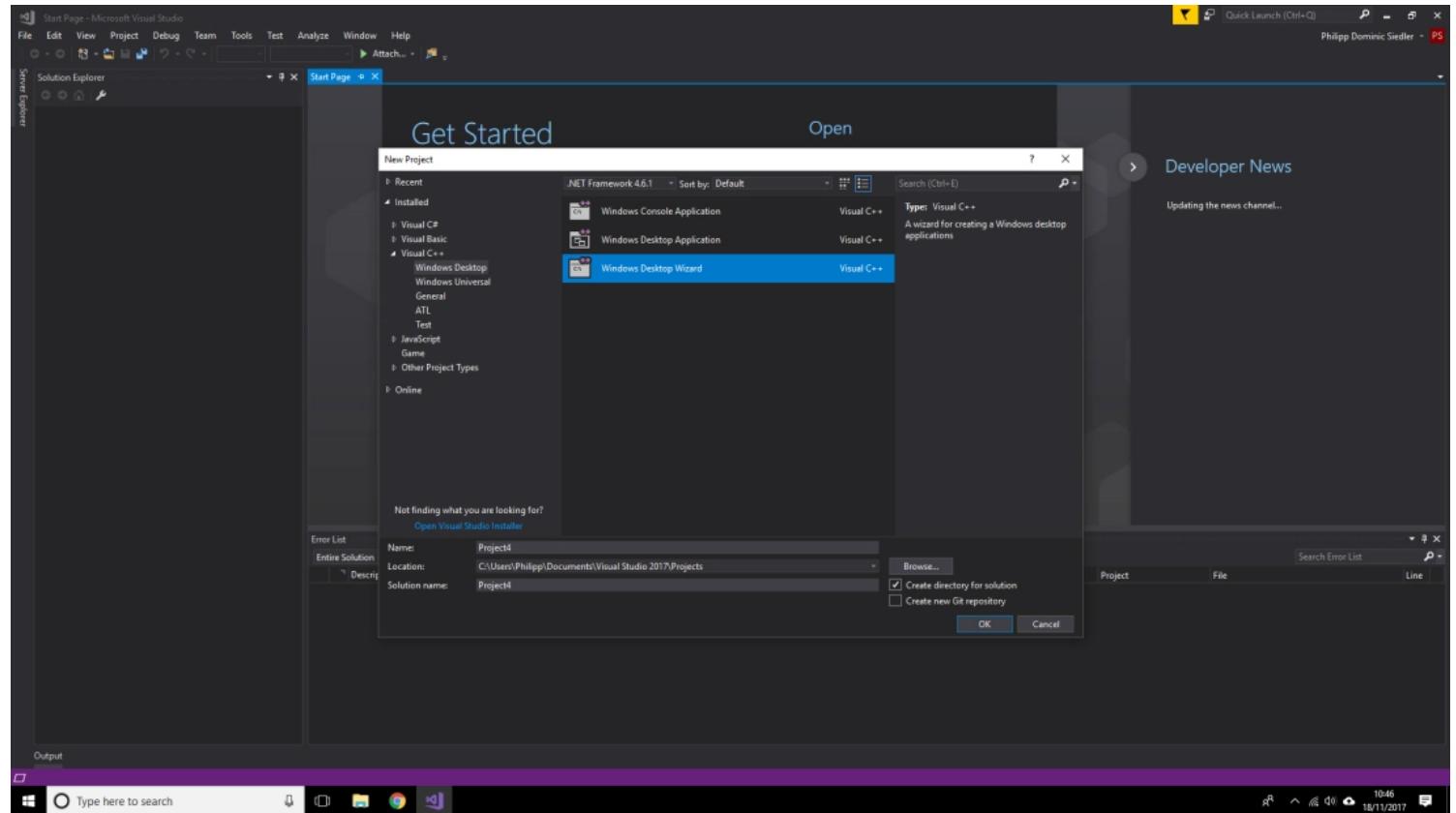


Name	Date modified	Type	Size
fluid.app	05/08/2017 13:14	File folder	
icons	05/08/2017 13:14	File folder	
pixmaps	05/08/2017 13:14	File folder	
fluid.exe	05/08/2017 13:28	Application	1,582 KB
fluidd.exe	05/08/2017 13:24	Application	1,527 KB
about_panel.h	20/09/2016 13:57	C/C++ Header	2 KB
alignment_panel.h	20/09/2016 13:57	C/C++ Header	5 KB
CodeEditor.h	04/04/2015 16:06	C/C++ Header	3 KB
comments.h	19/07/2011 05:49	C/C++ Header	3 KB
ExternalCodeEditor_UNIX.h	16/08/2016 21:42	C/C++ Header	2 KB
ExternalCodeEditor_WIN32.h	16/08/2016 22:08	C/C++ Header	3 KB
Fl_Type.h	20/09/2016 13:57	C/C++ Header	28 KB
Fl_Widget_Type.h	19/07/2011 05:49	C/C++ Header	1 KB
Fluid_Image.h	18/12/2015 18:56	C/C++ Header	2 KB
function_panel.h	20/09/2016 13:57	C/C++ Header	5 KB
print_panel.h	20/09/2016 13:57	C/C++ Header	2 KB

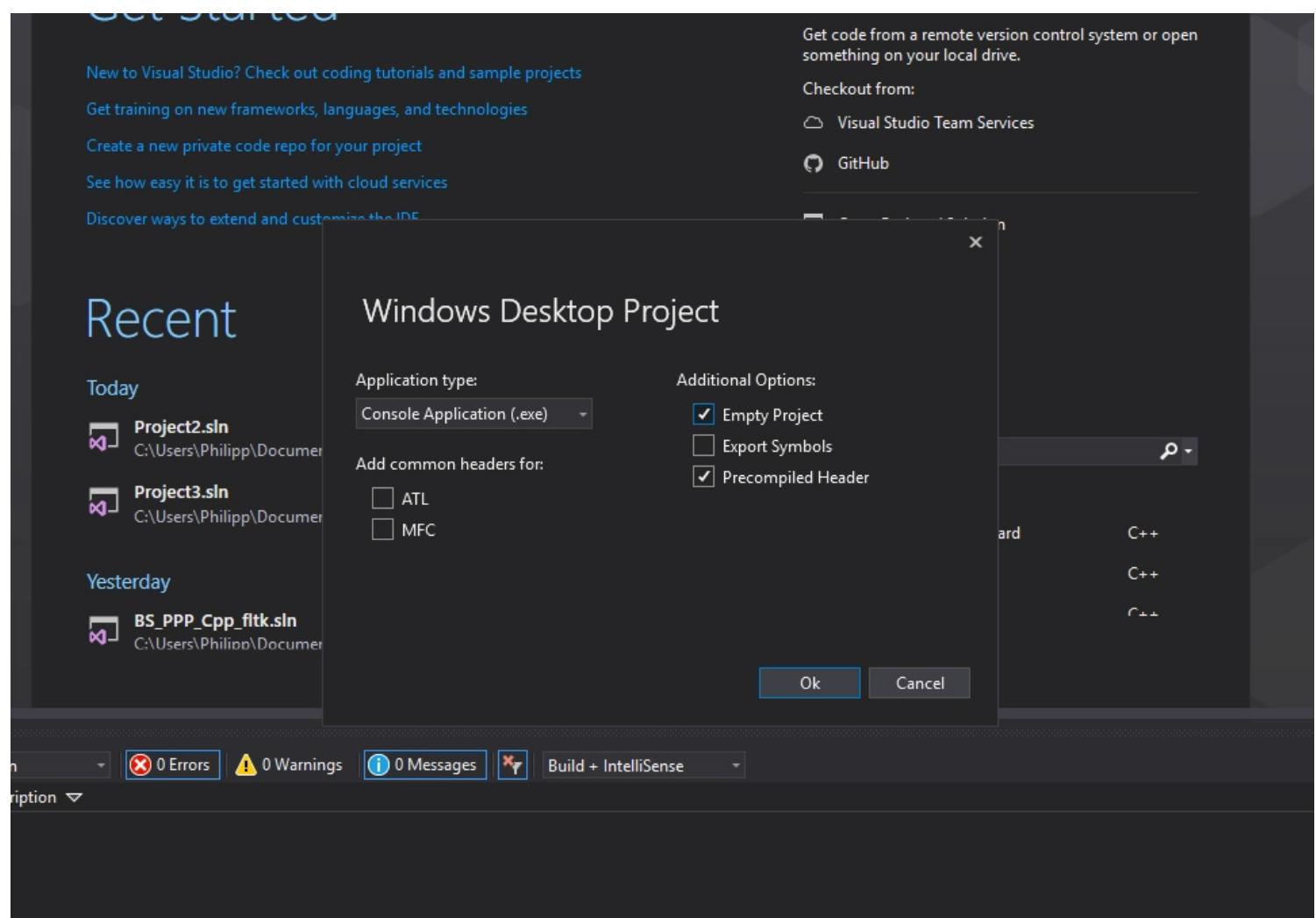
Name	Date modified	Type	Size
HostX64	12/04/2017 22:44	File folder	
HostX86	12/04/2017 22:44	File folder	
fluid.exe	05/08/2017 13:28	Application	1,582 KB
fluidd.exe	05/08/2017 13:24	Application	1,527 KB

Excellent, that's all done. Now lets setup a new project and see if Visual Studio 2017 Community is able to locate and use the files we just copied over.

Let's open Visual Studio 2017 Community (or any other of course) and under the **Visual C++ – Windows Desktop** tab create a new **Windows Desktop Wizard** project.



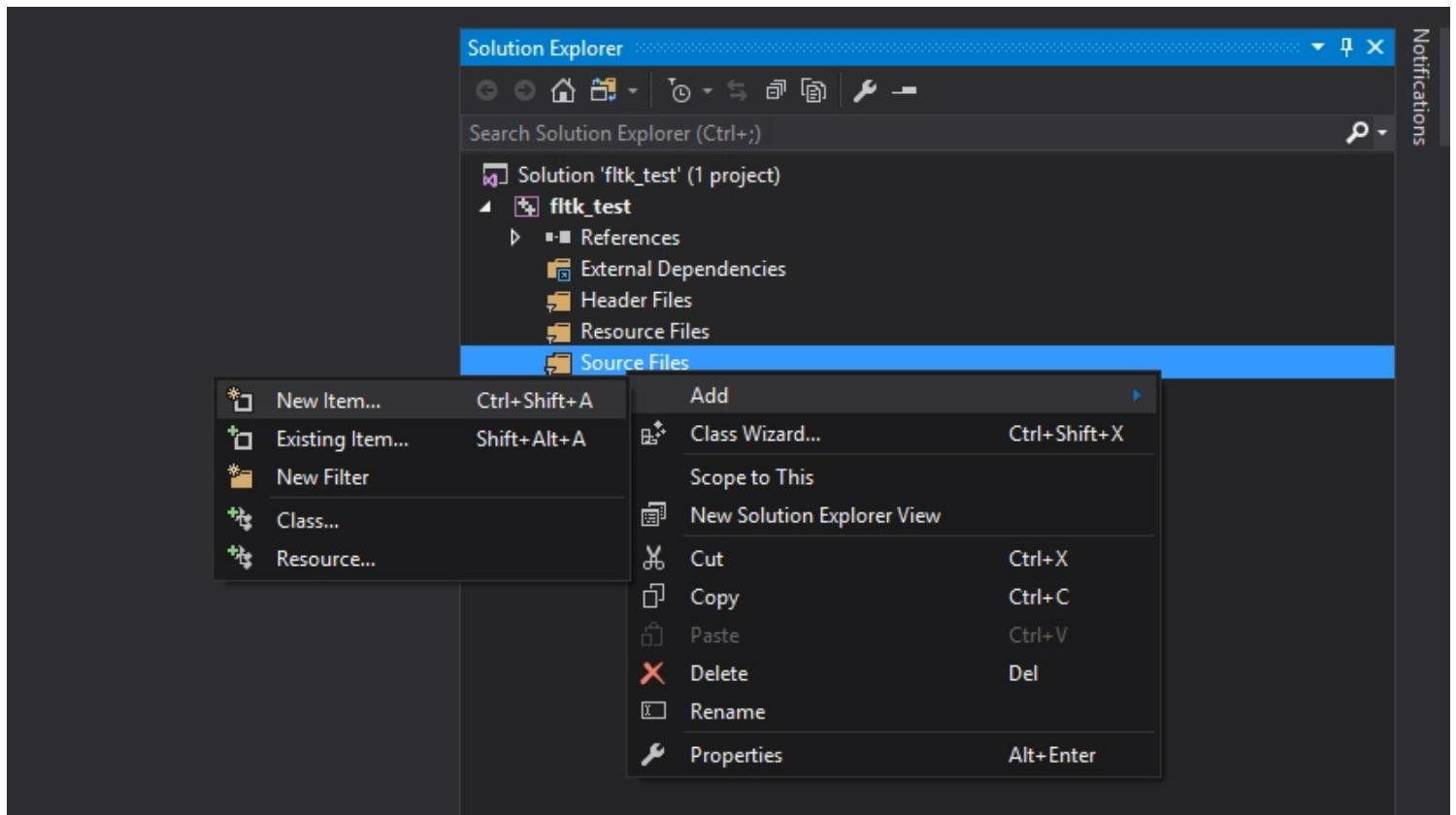
Tick the box **Empty project** (very important).



I called it `fltk_test`.

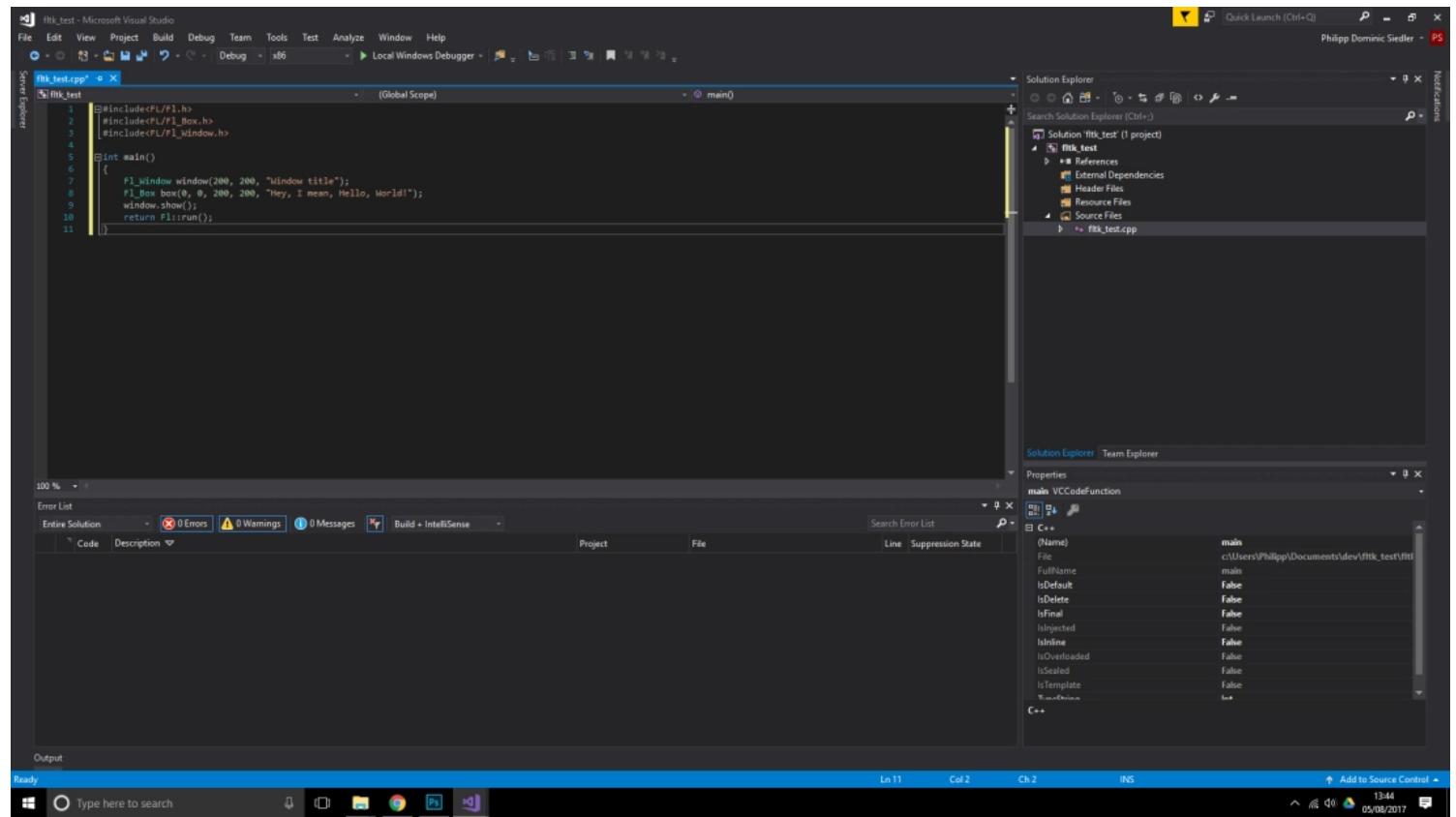
In your newly created **Empty Project** right click the **Source Files** folder and **Add a New Item** which is going to be a `.cpp` file.

I called it `fltk_test.cpp`.



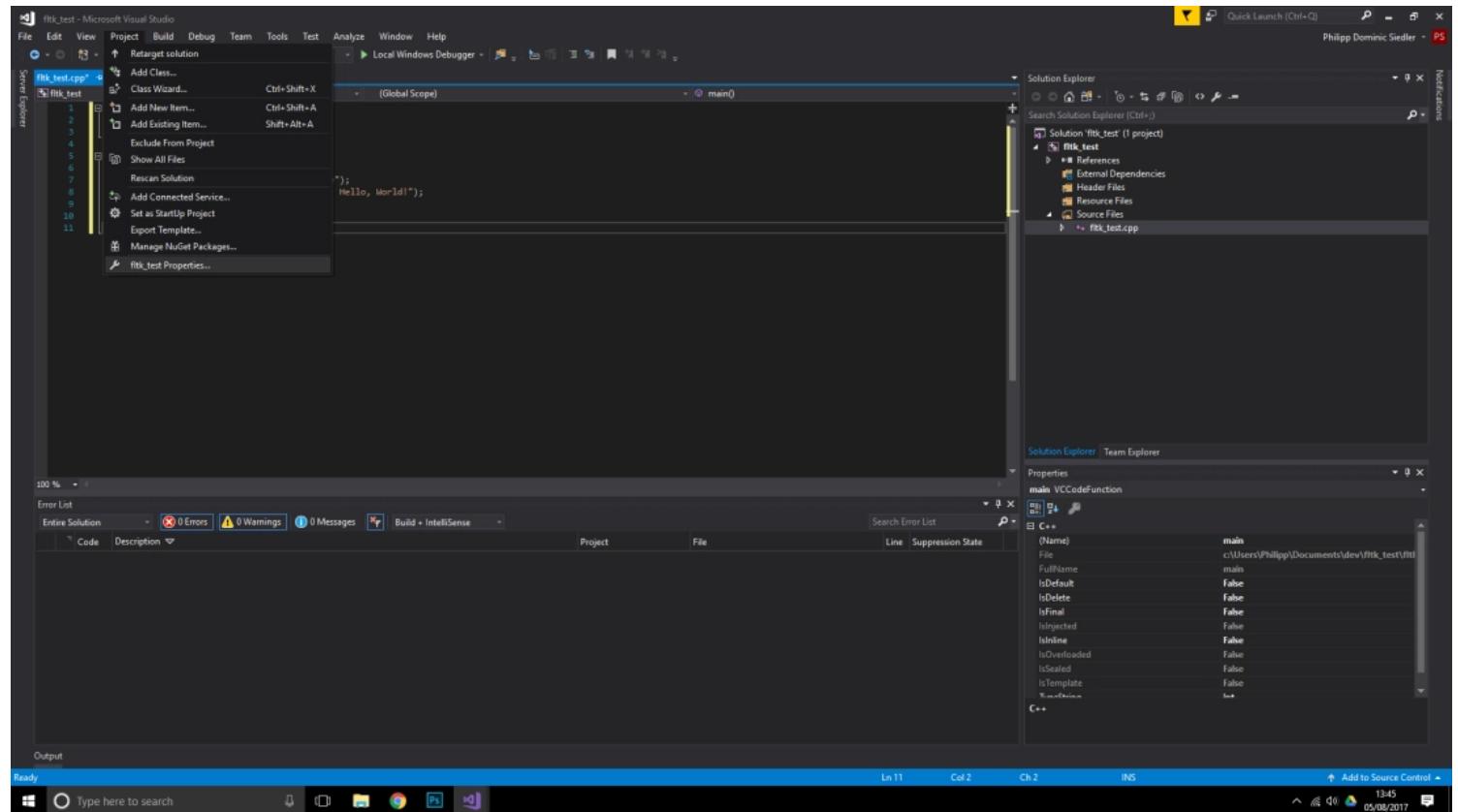
Copy the following code into your .cpp file:

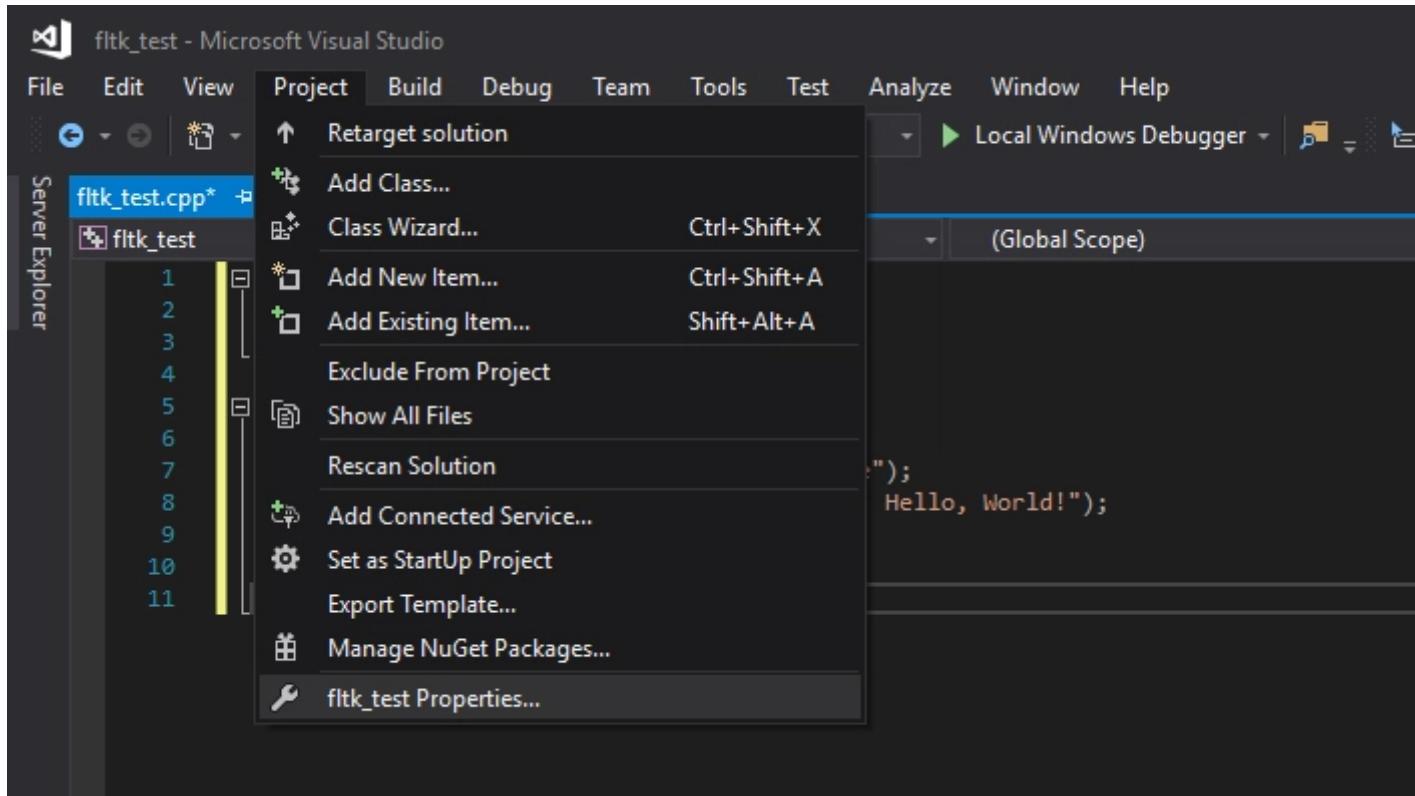
```
1 #include<FL/Fl.h>
2 #include<FL/Fl_Box.h>
3 #include<FL/Fl_Window.h>
4
5 int main()
6 {
7     Fl_Window window(200, 200, "Window title");
8     Fl_Box box(0, 0, 200, 200, "Hey, I mean, Hello, World!");
9     window.show();
10    return Fl::run();
11 }
```



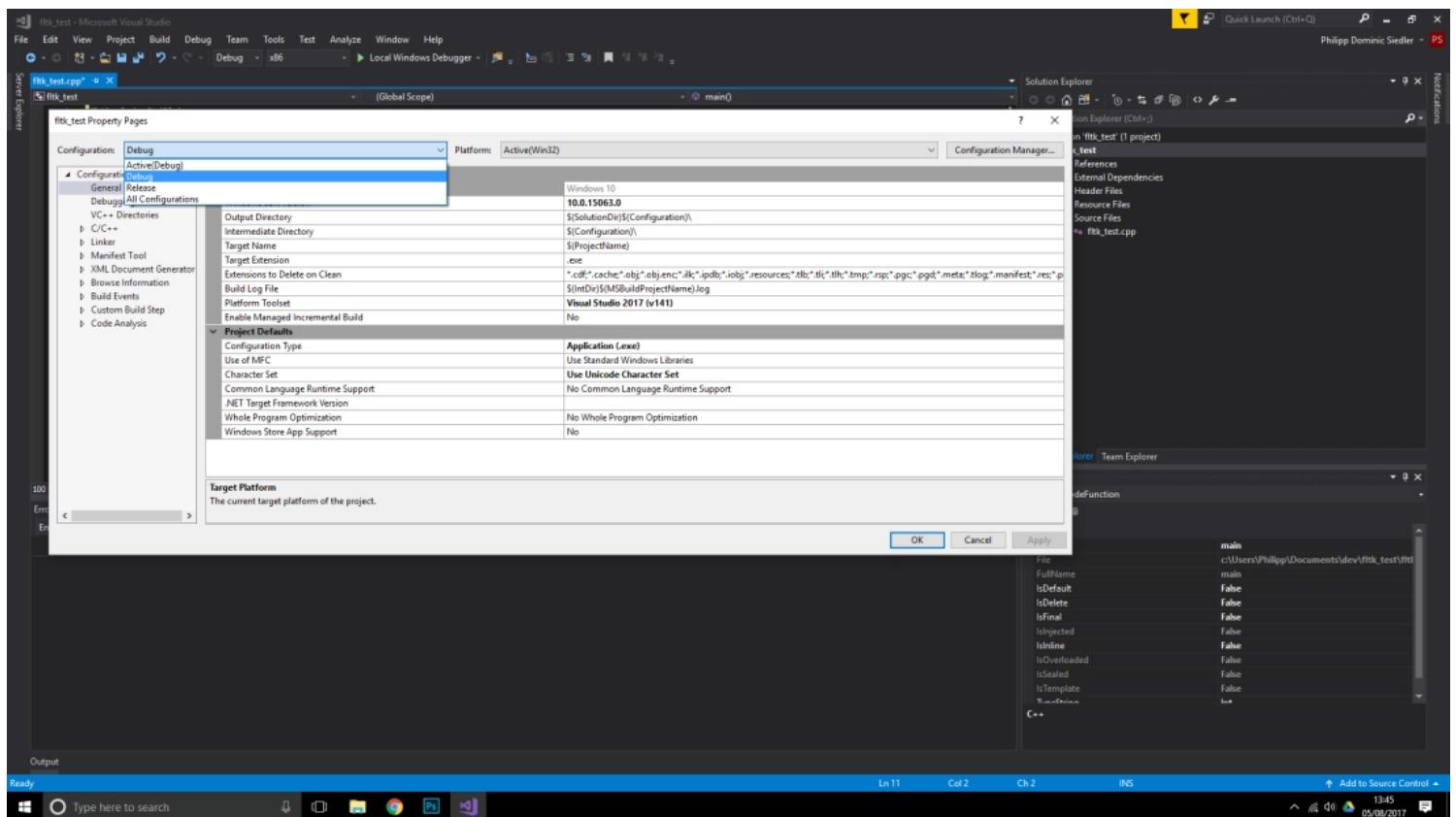
(NOTE: The following Properties setup needs to be done for every new fltk project you are creating! I will tell you later to come back to this point.)

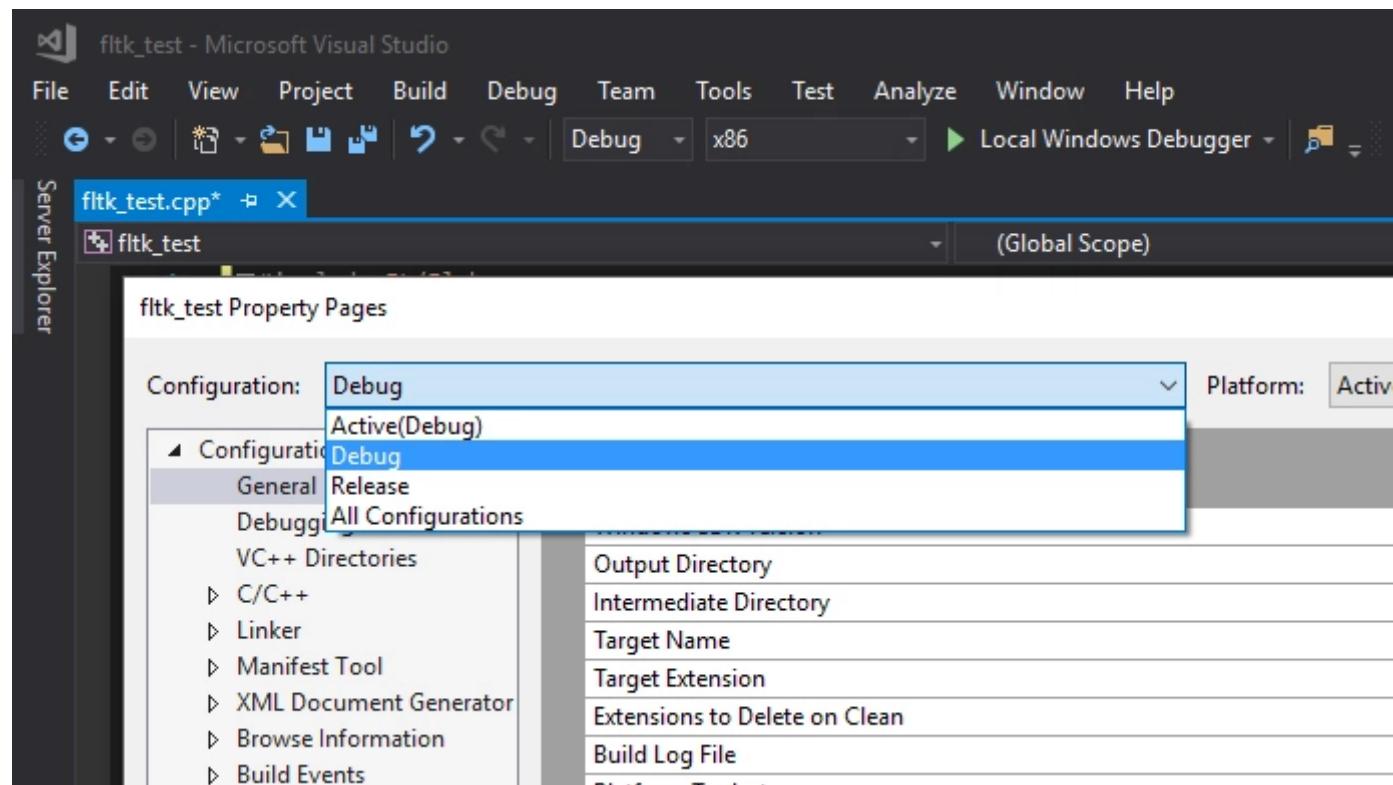
Then go to your top navigation bar and click Project and in the drop-down menu Properties.



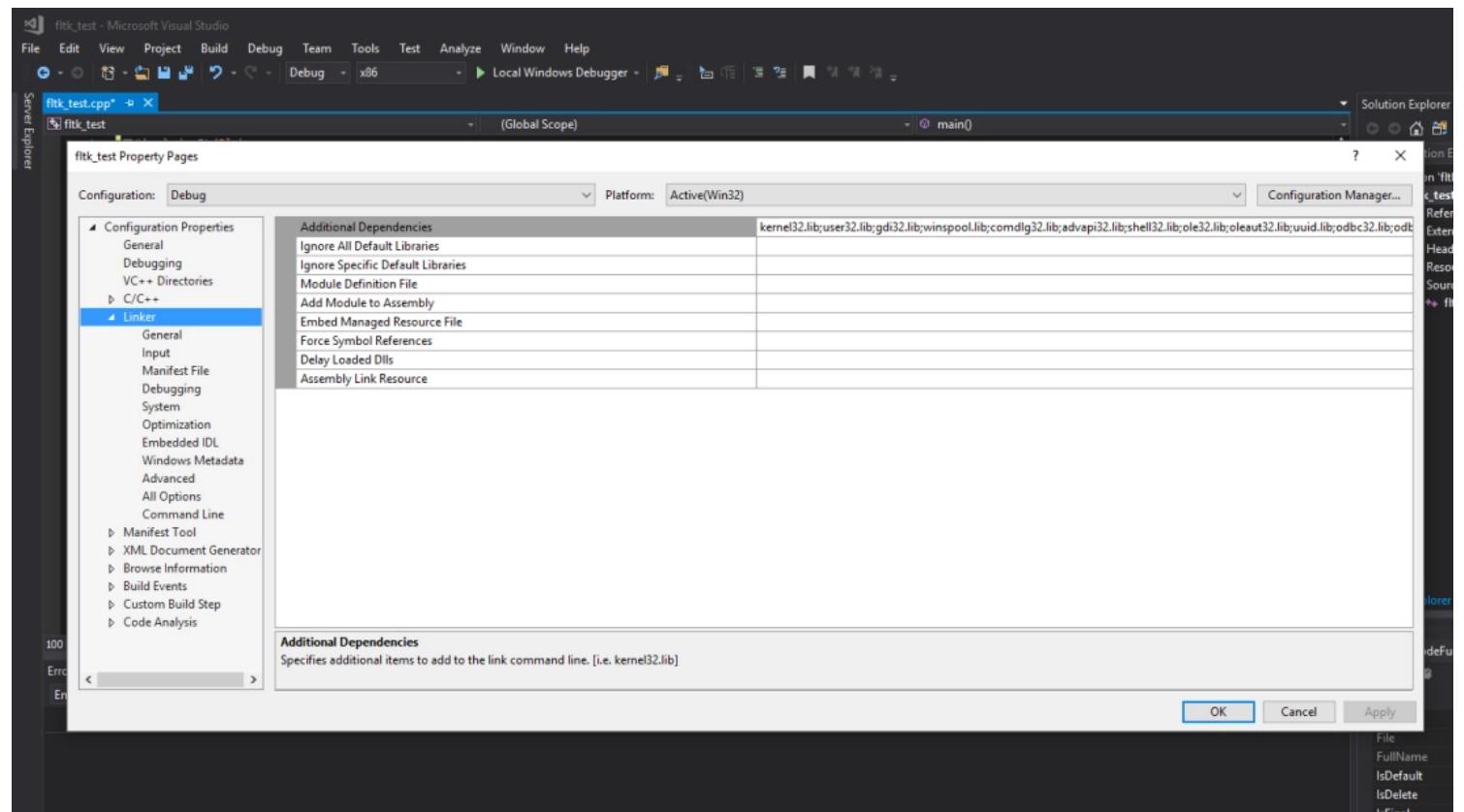


Make sure to set the Configuration to **Debug**.

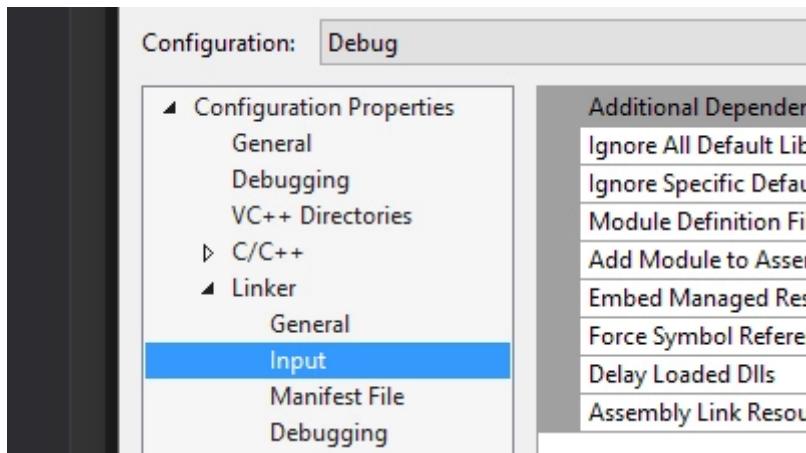




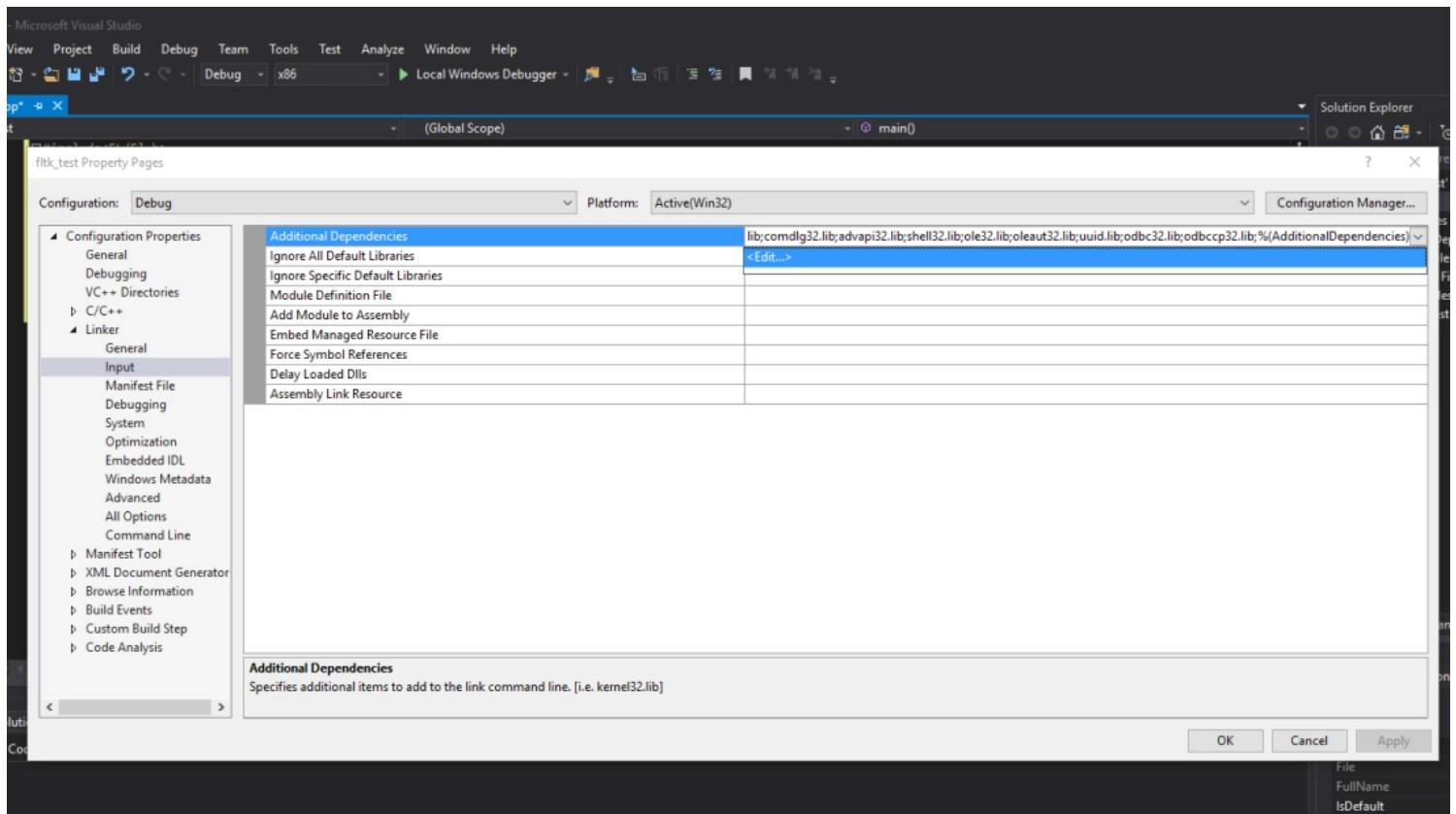
On your left hand side go to **Linker**.



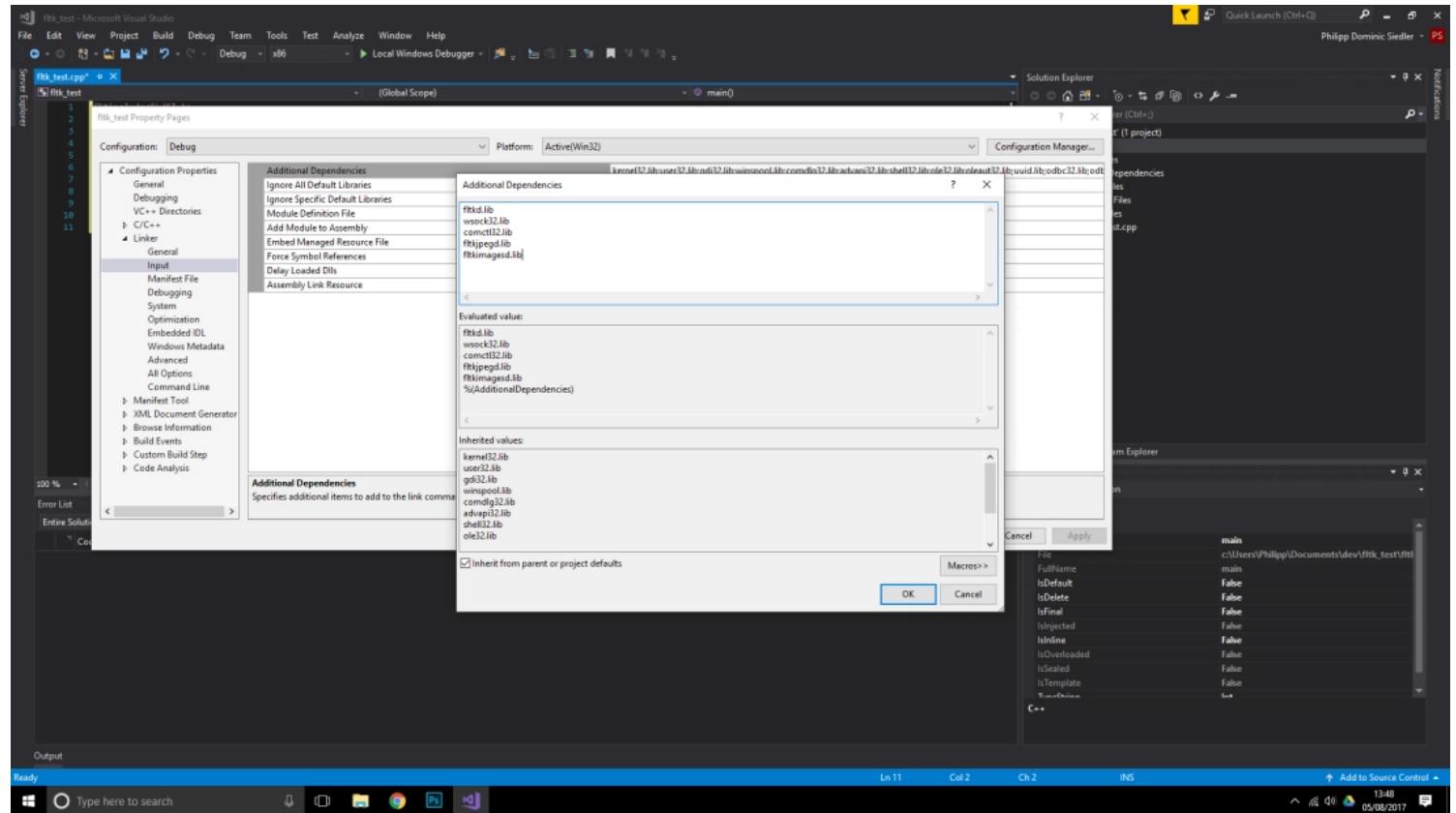
And **Input**.



Click on the drop-down arrow in the **Additional Dependencies** tab.



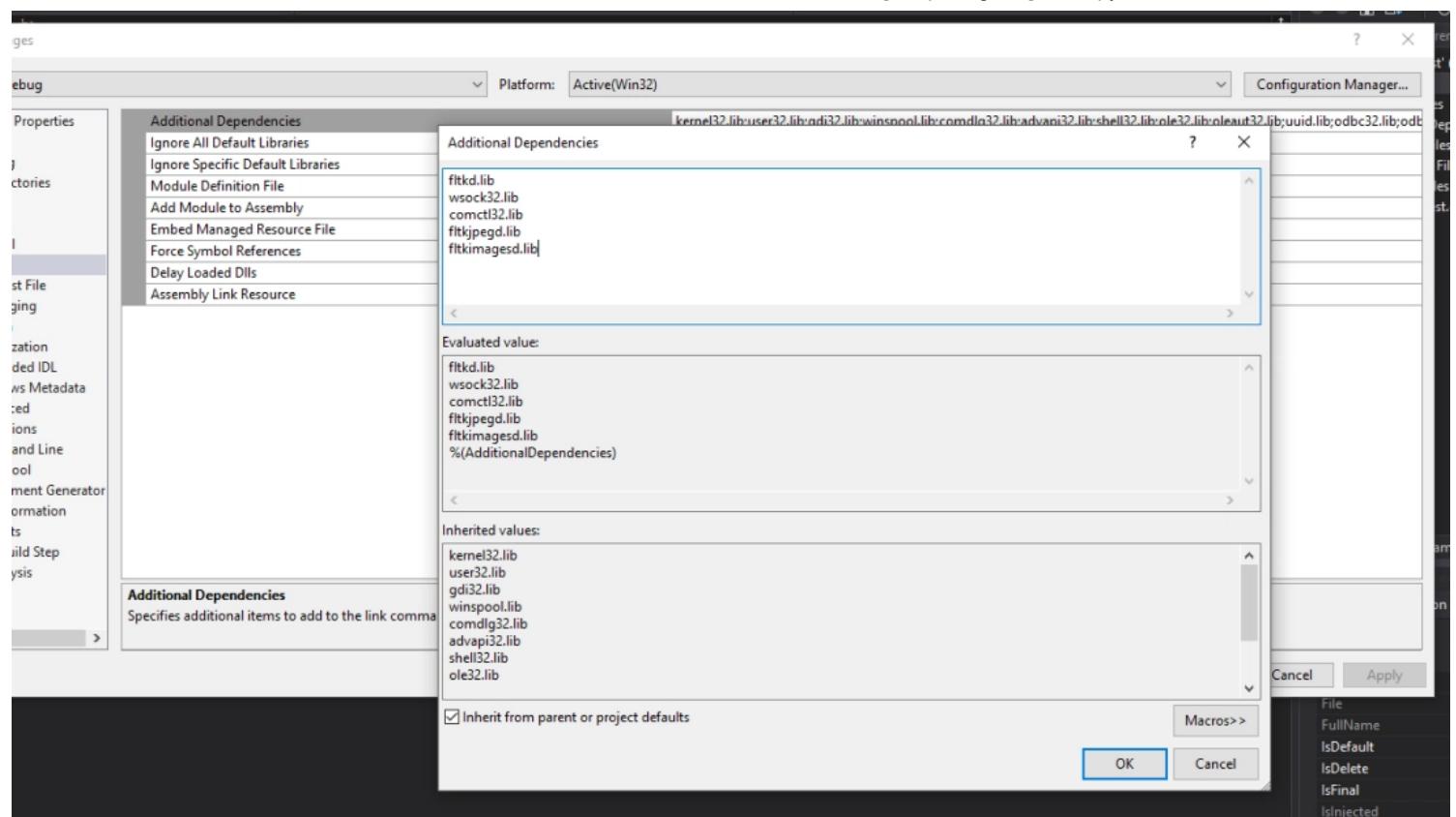
And click on Edit.



A new window will pop up. In the screenshot shown I filled in already the following entries:

fltkd.lib
wsock32.lib
comctl32.lib
fltkjpegd.lib
fltkimagesd.lib

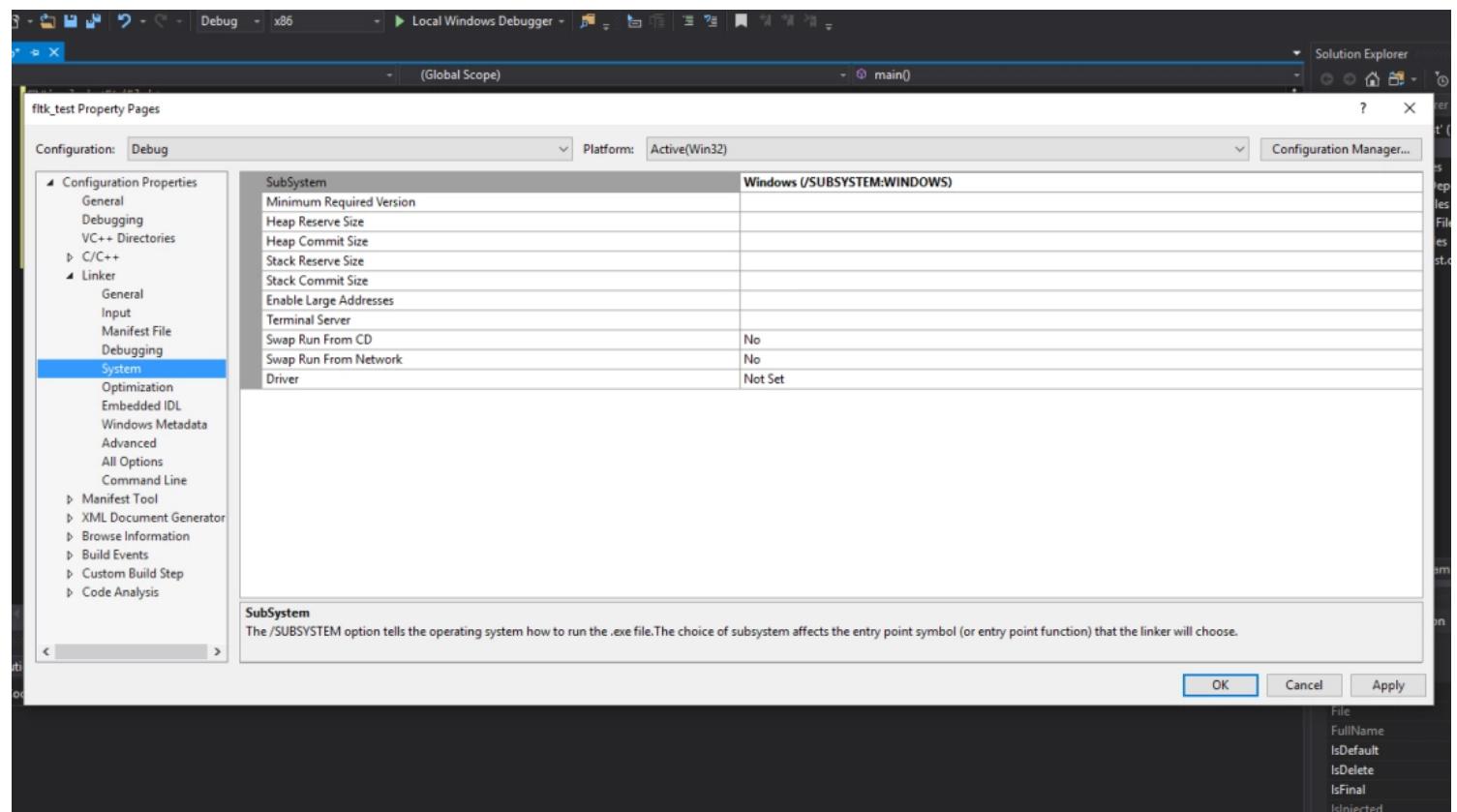
Please do the same.

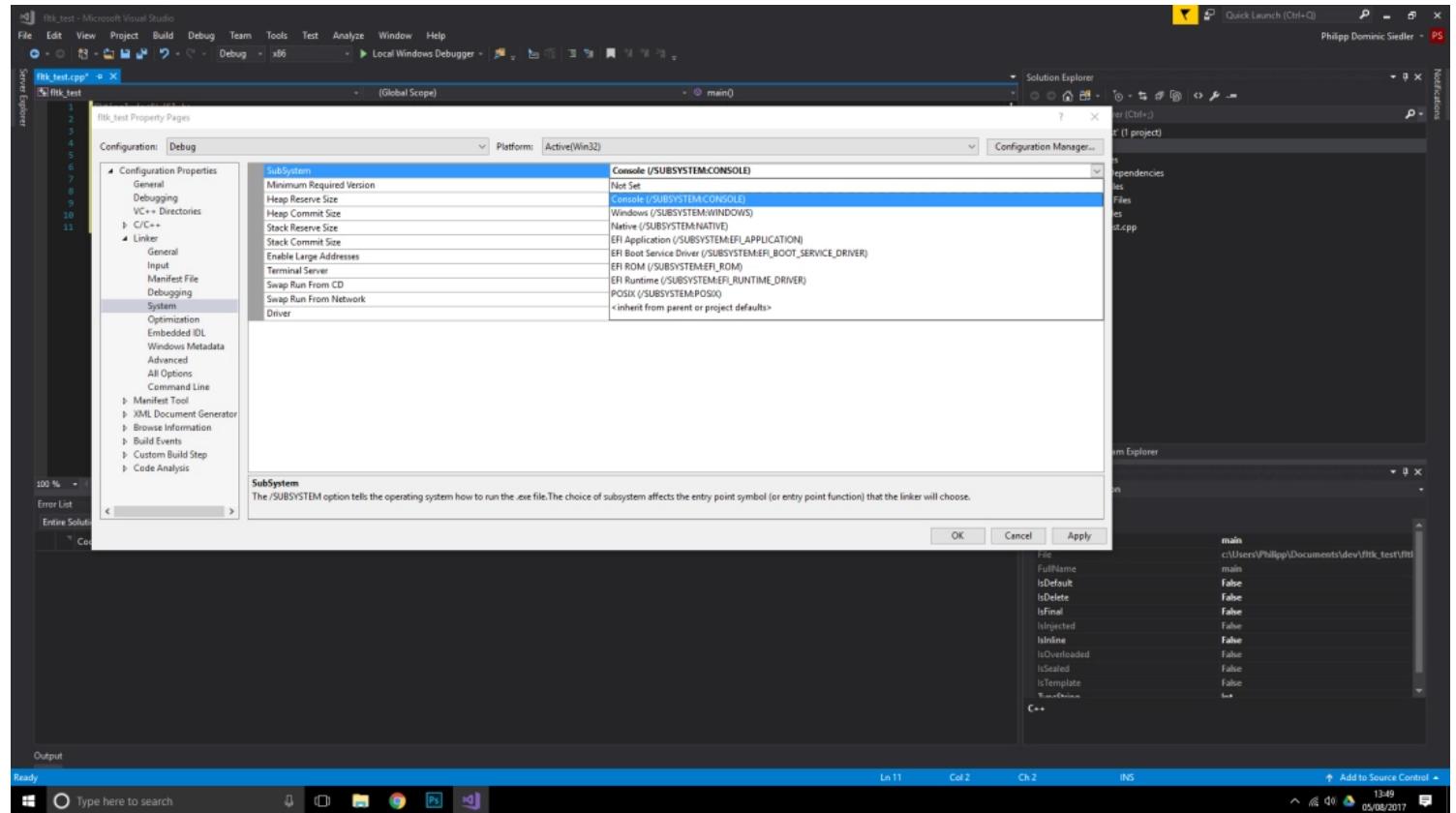


Click **OK** to confirm your entry.

Now go to **System**.

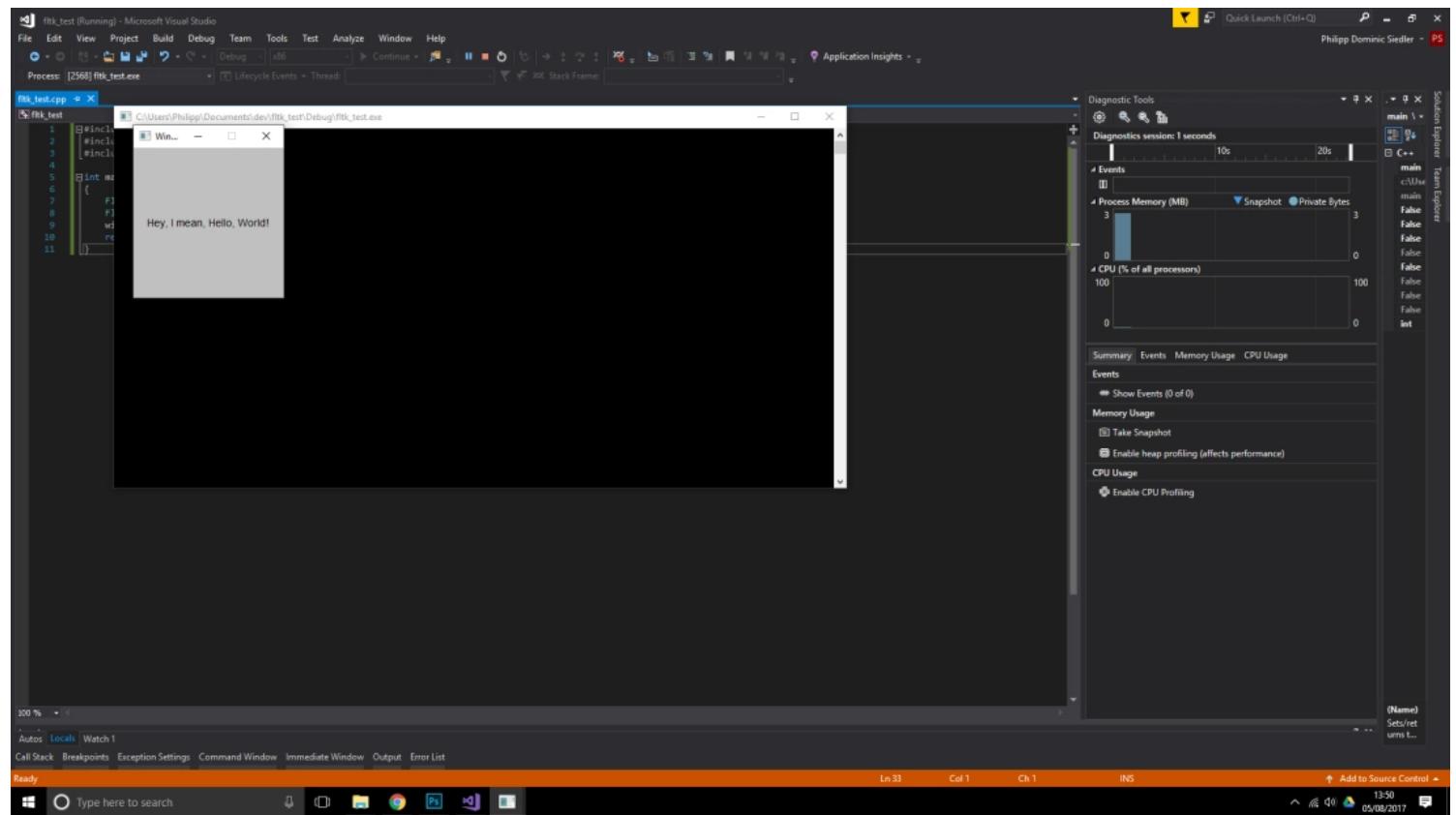
And set the **SubSystem** to **Console (/SUBSYSTEM:CONSOLE)** with the help of the drop-down menu.



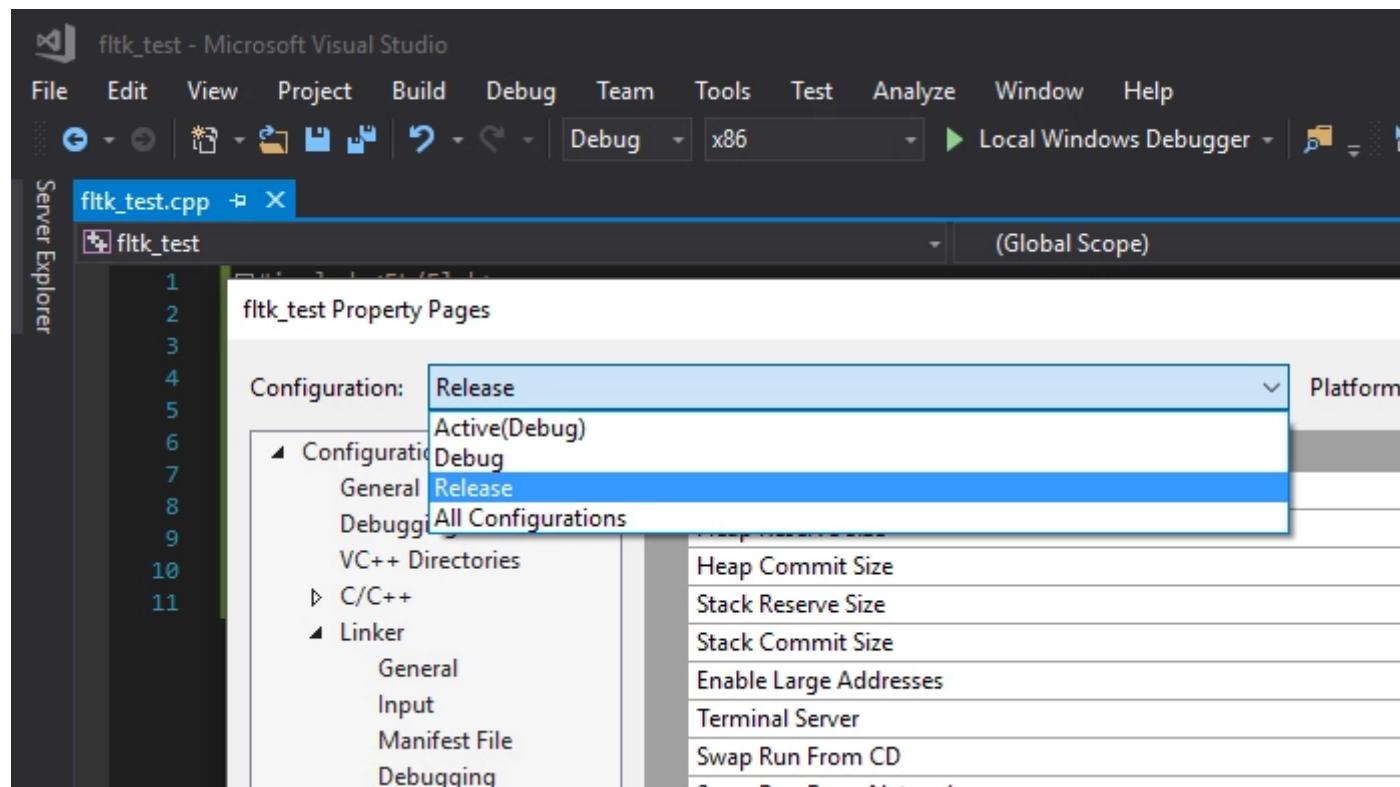


Click **Apply** and hit **F5** to test-run the project.

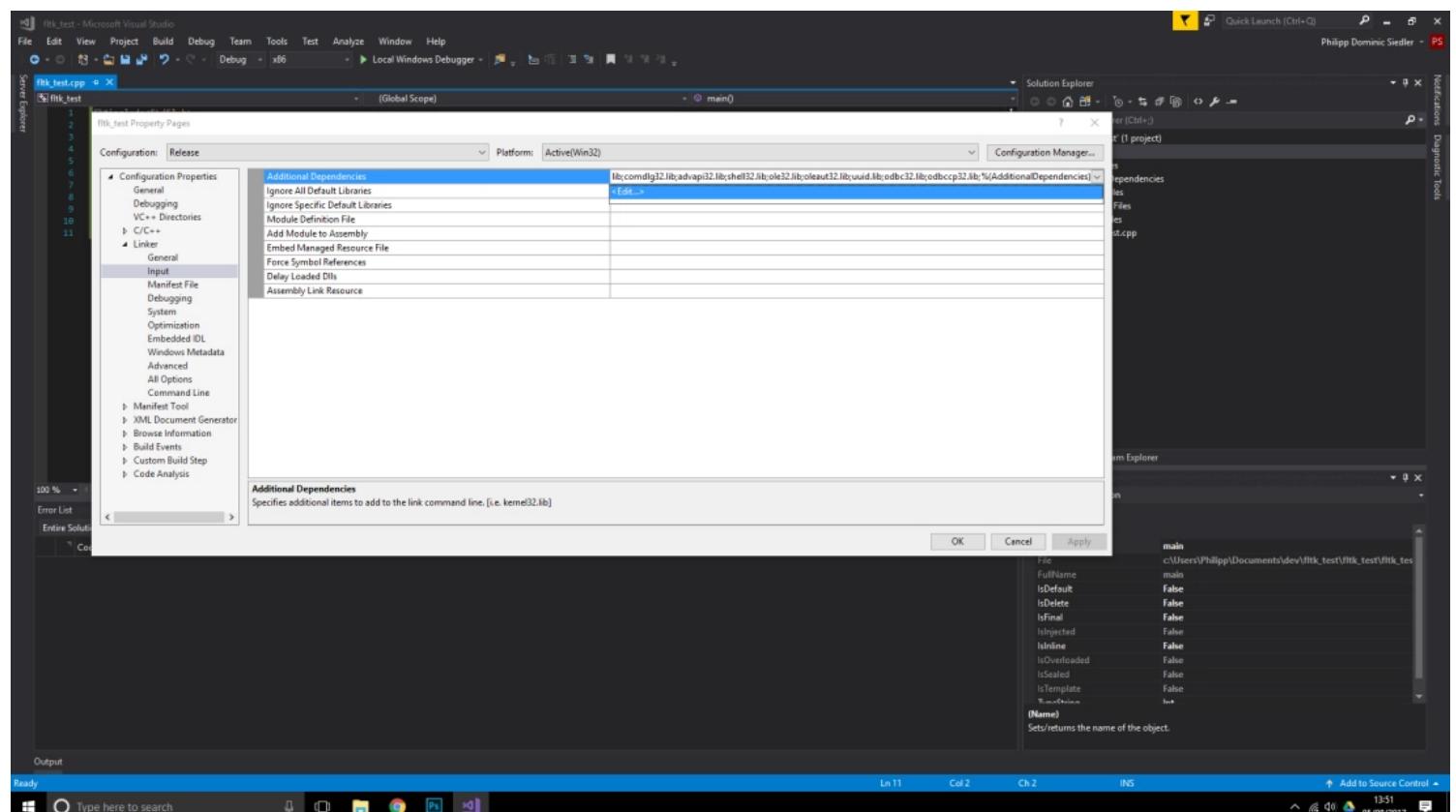
If you have done everything I told you so far your project should compile and run and show something like this:



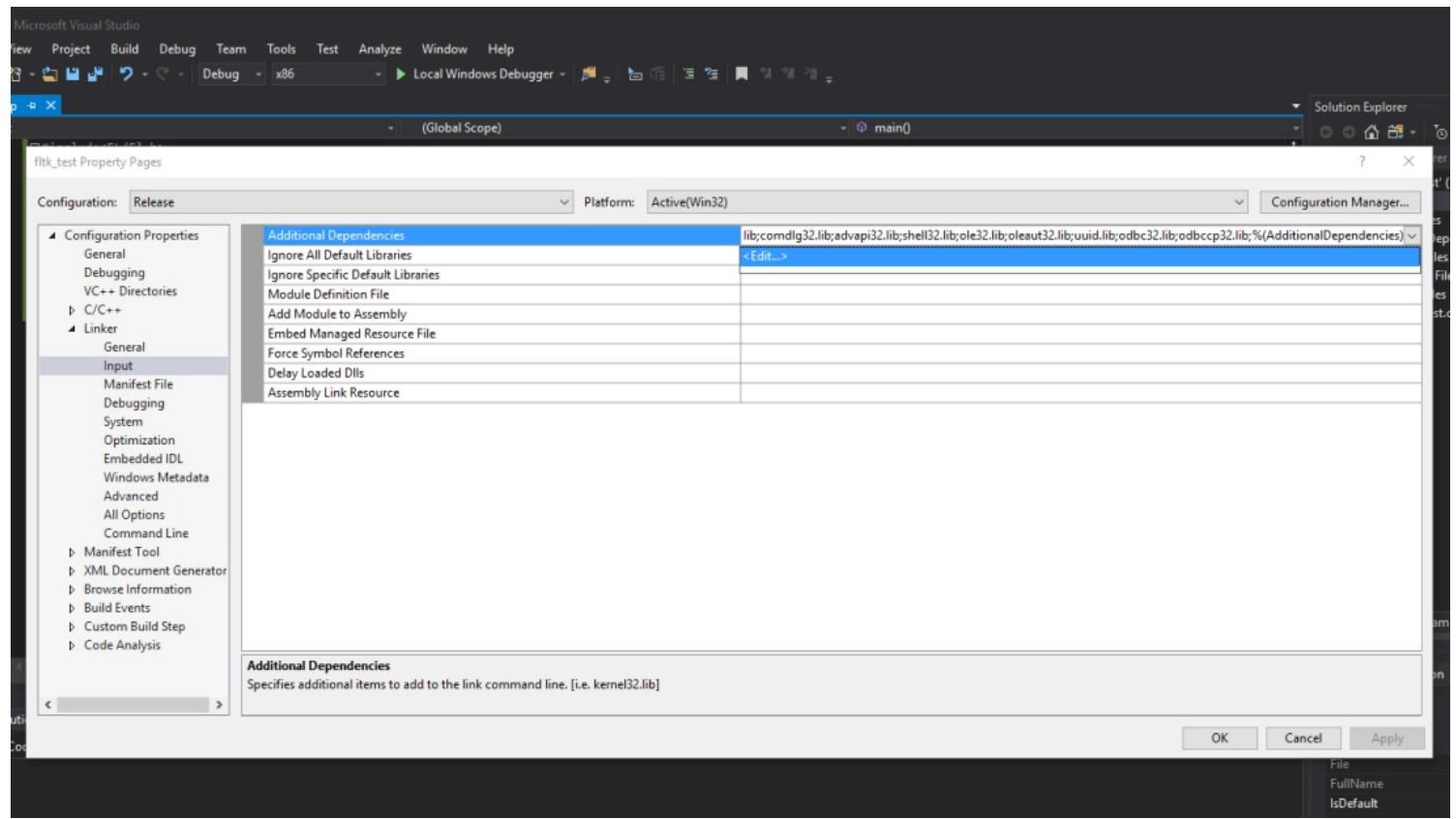
Great, now go back to your **Properties – Project** and **Properties**.
And chose from the Configuration **Release**.



On the left hand side to **Linker** and then to **Input**, just like you did before.

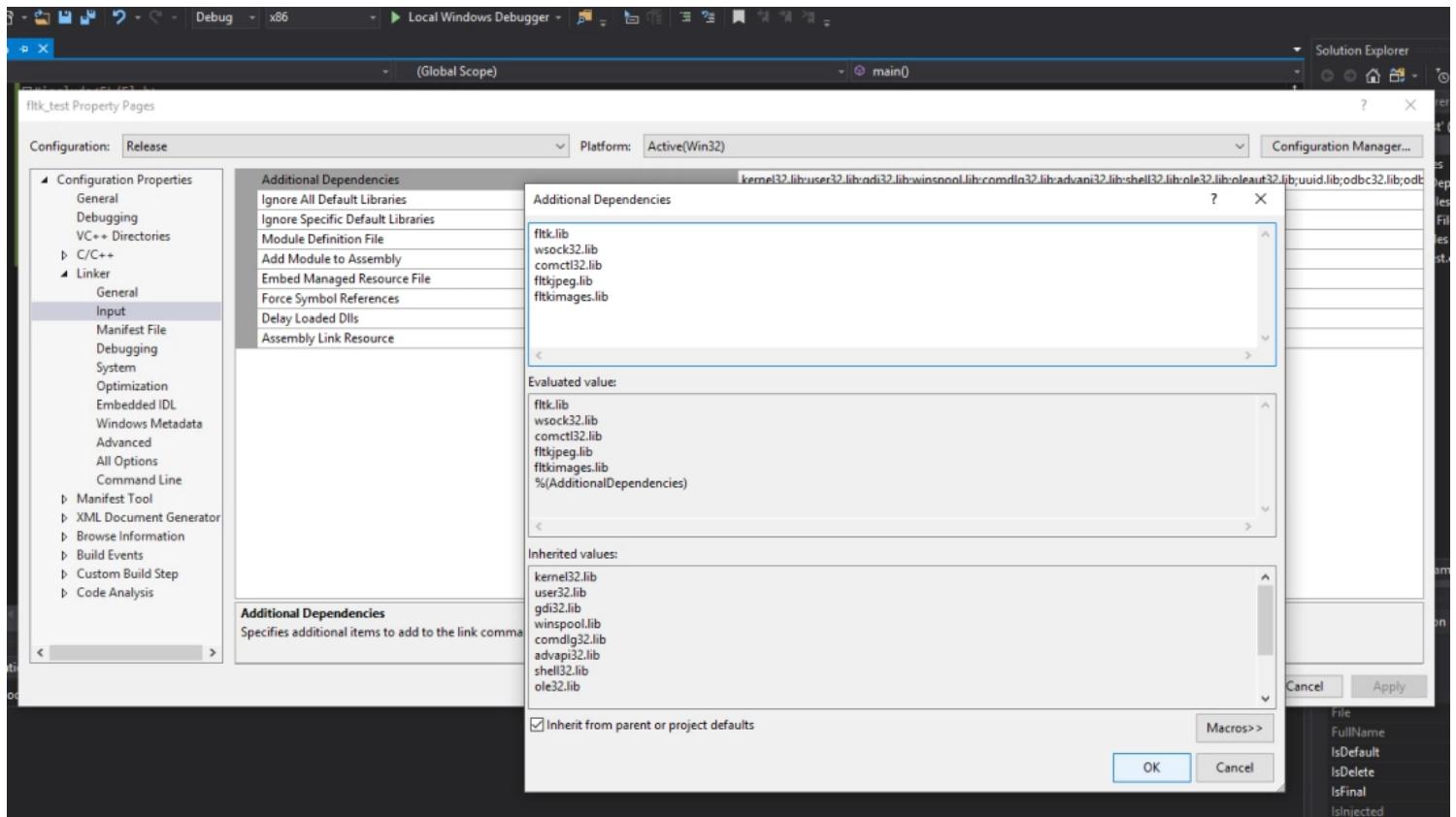


Click on the drop-down arrow to show the **Additional Dependencies** menu and select **Edit**.

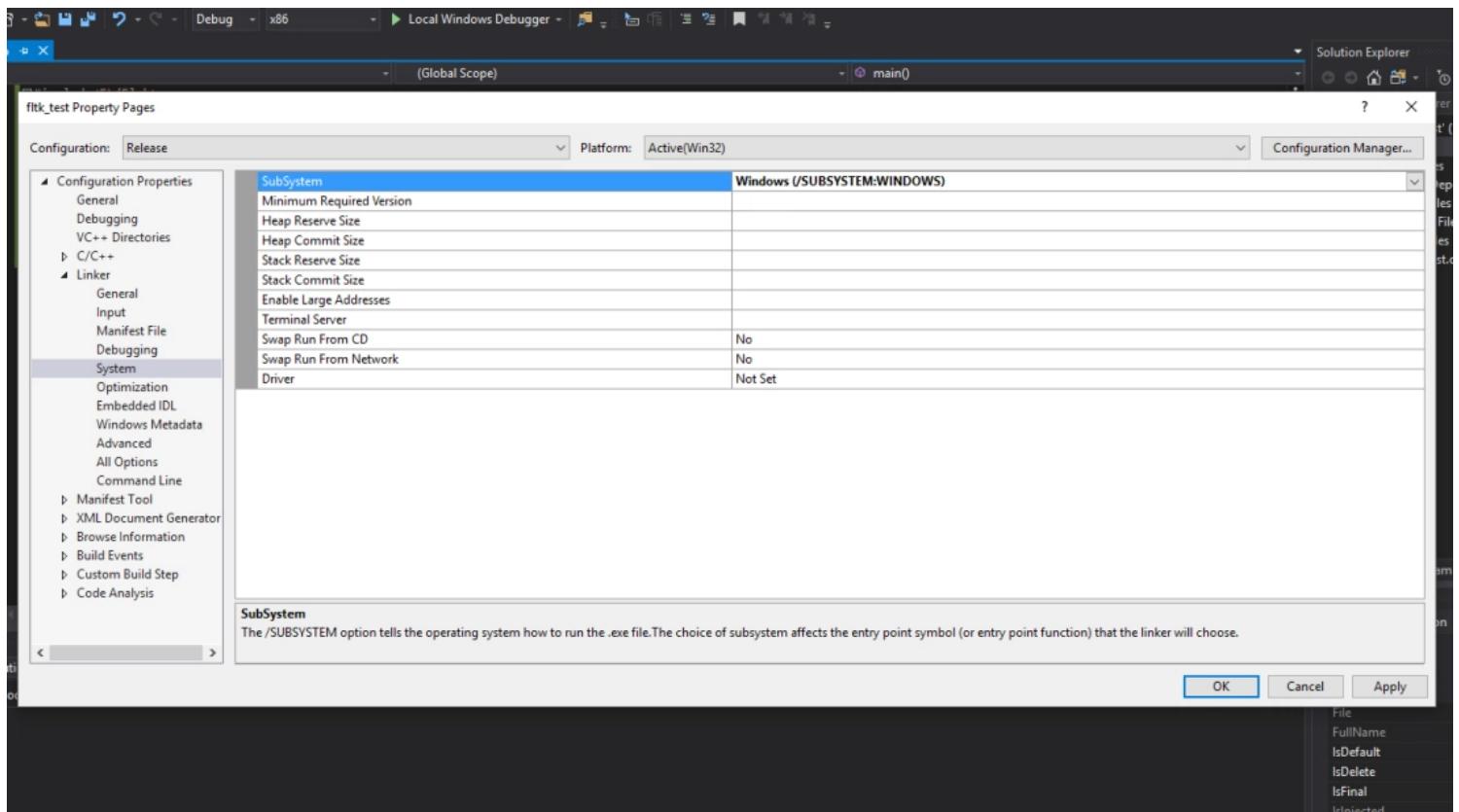


Copy paste the following entries into the empty field (I filled them out already, Note: Since this is the Release Category there is no d for debug in the .lib names):

fltk.lib
wsock32.lib
comctl32.lib
fltkjpeg.lib
fltkimages.lib



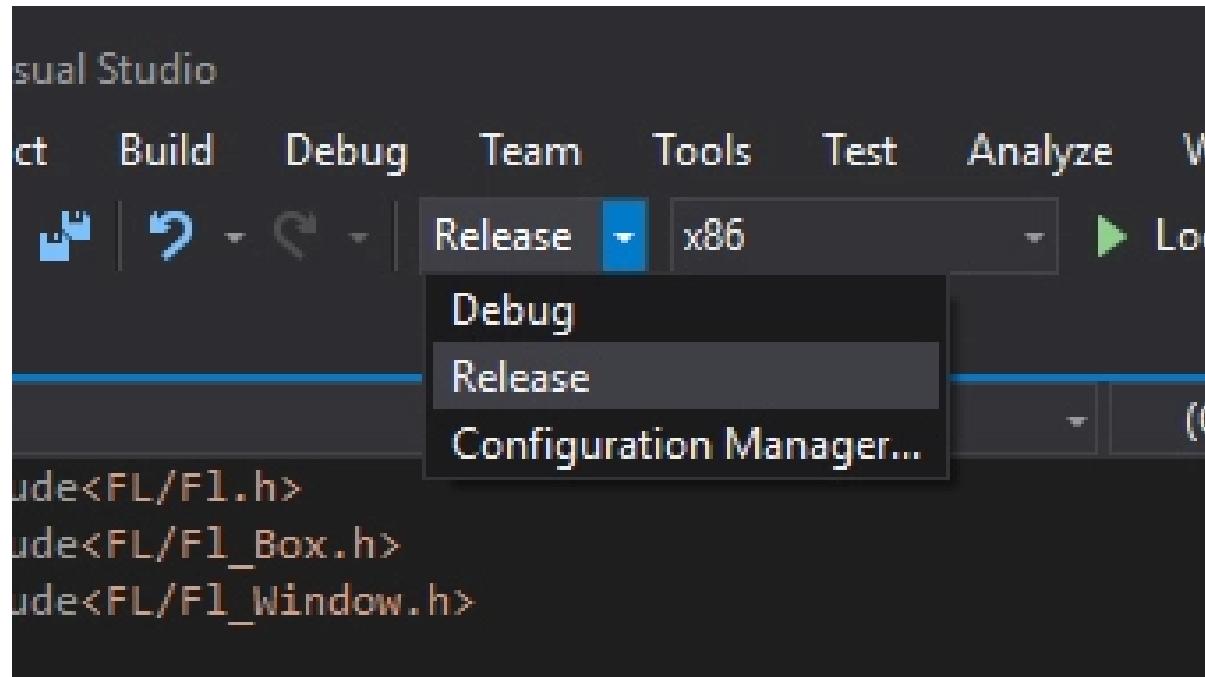
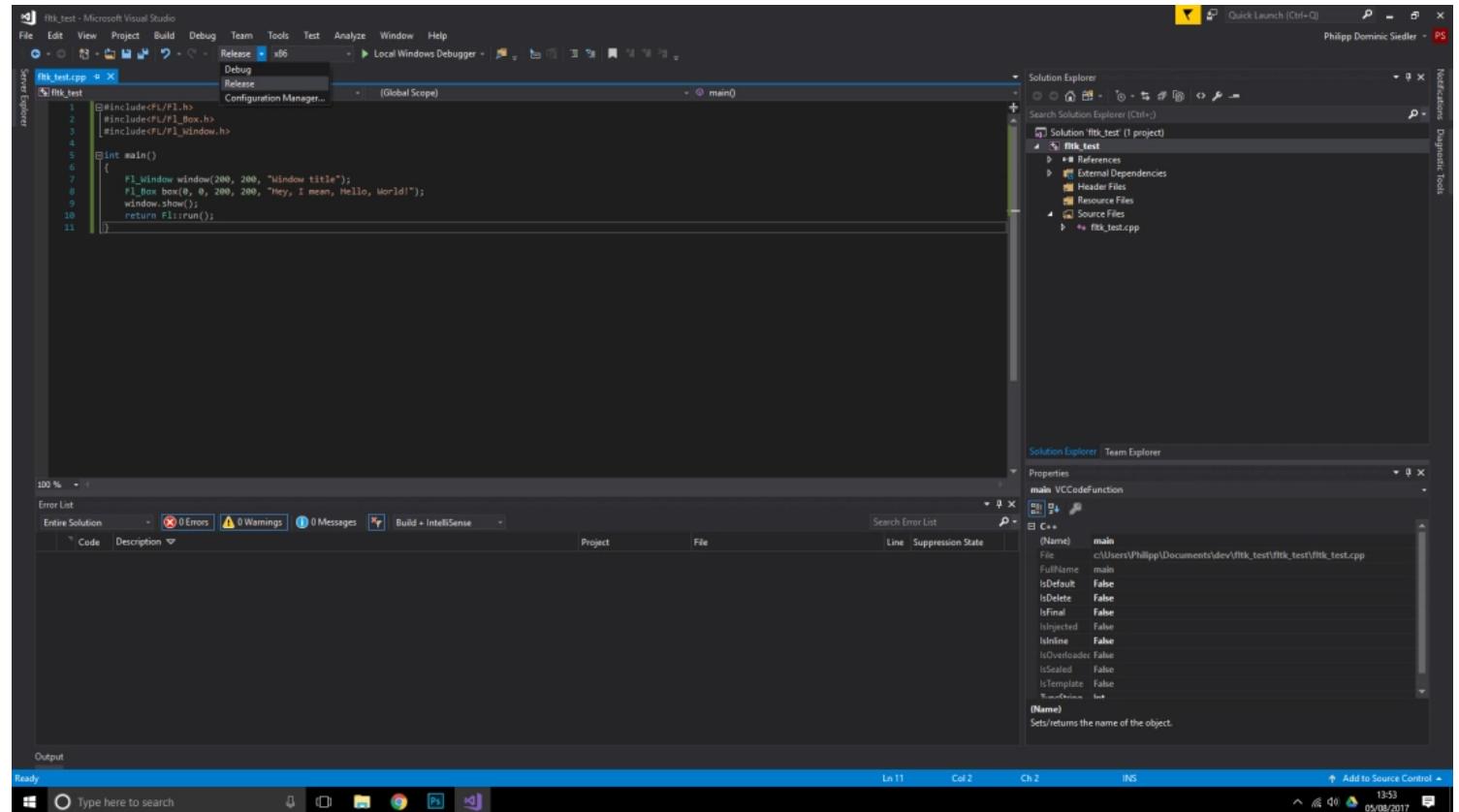
Click OK and go to System.



Set the **SubSystem** to **Windows (/SUBSYSTEM:WINDOWS)**, which can be found in the drop-down menu and click OK.

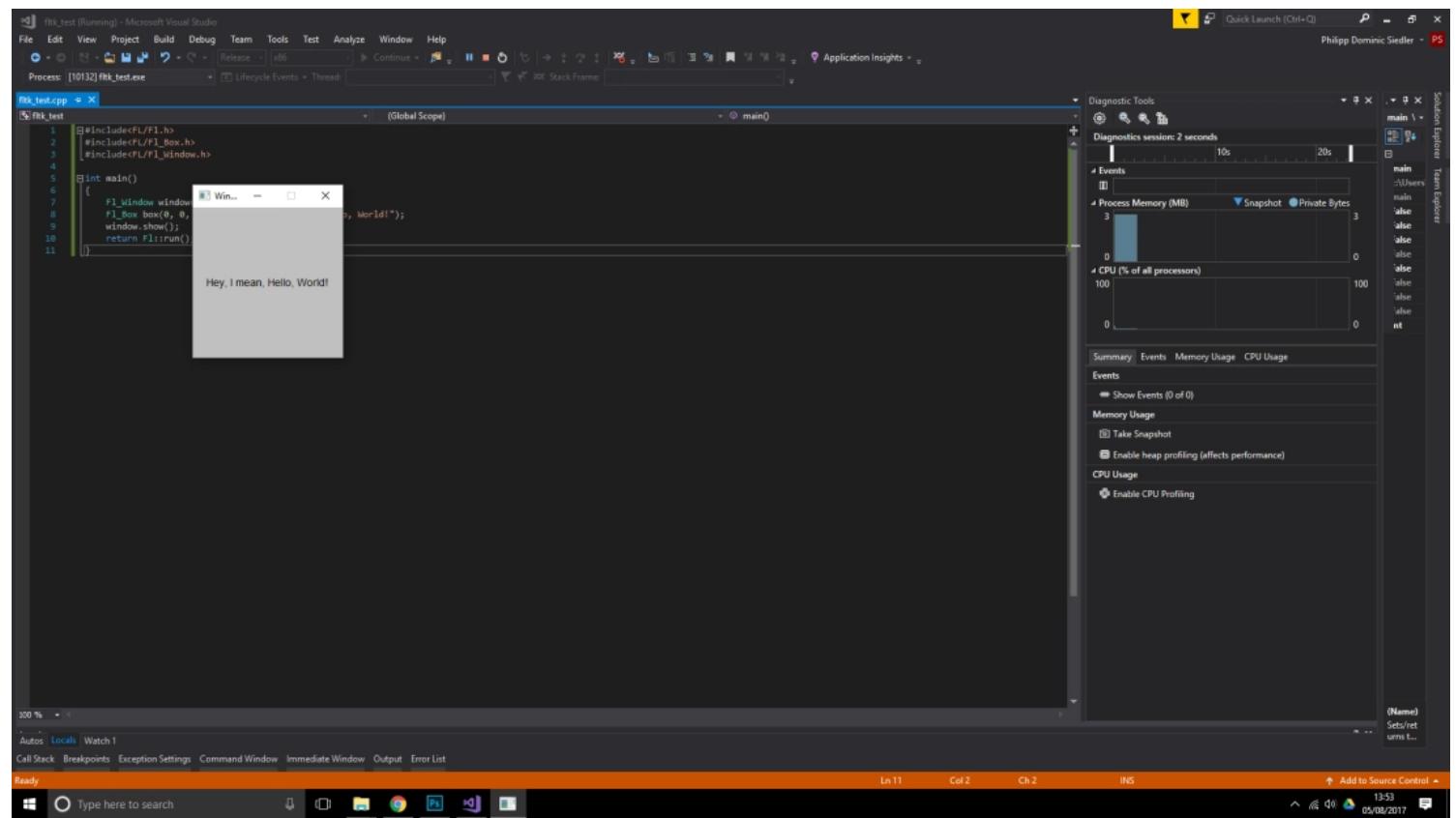
Apply your changes and exit your **Properties** window with **OK**.

Now change the **Solution Configurations** to **Release**.



And run your project by pressing **F5** on your keyboard, or the playbutton on the top of your canvas **Local Windows Debugger**.

If everything went well you should be able to look at something like this (application window with the fltk-interface and no console window, if this didn't go as planned please go to the next step):



At this point maybe people experience an error message saying “... cannot open include file ‘x11/xlib.h’ no such file or directory ...”. Here is how to fix this:
 (If everything works for you just fine, please skip ahead!)

Double click the error message in your errors tab, the x.H file will open in **Visual Studio 2017 Community** as a new tab, if not go to your **External Dependencies** folder in your **Solution Explorer** within Visual Studio and look for the file manually and open it. As you may notice already, the errors will be underlined in curly red.

```

x.H*  Source.cpp
Project3  (Global Scope)

1 // $Id: x.H 10932 2015-11-26 16:34:58Z manolo $
2 // "X11 header file for the Fast Light Tool Kit (FLTK)."
3 //
4 // Copyright 1998-2012 by Bill Spitzak and others.
5 //
6 // This library is free software. Distribution and use rights are outlined in
7 // the file "COPYING" which should have been included with this file. If this
8 // file is missing or damaged, see the license at:
9 //
10 //     http://www.fltk.org/COPYING.php
11 //
12 // Please report all bugs and problems on the following page:
13 //
14 //     http://www.fltk.org/str.php
15 //
16 //
17 //
18 // These are internal fltk symbols that are necessary or useful for
19 // calling Xlib. You should include this file if (and ONLY if) you
20 // need to call Xlib directly. These symbols may not exist on non-X
21 // systems.
22 //
23
24 #if !defined(FL_X_H) && !defined(FL_DDOXYGEN)
25 # define FL_X_H
26
27 # include "Enumerations.H"
28
29 # ifdef WIN32
30 #   include "win32.H"
31 # elif defined(__APPLE__)
32 #   include "mac.H"
33 # else
34 #   if defined(_ABIN32) || defined(_ABI64) // fix for broken SGI Irix X.h files
35 #     pragma set woff 3322
36 #   endif
37 #   include <X11/Xlib.h>
38 #   include <X11/Xutil.h>
39 #   if defined(_ABIN32) || defined(_ABI64)
40 #     pragma reset woff 3322
41 #   endif
42 #   include <X11/Xatom.h>
43 #   include "Fl_Window.H"

```

100 %

Now go to line 28, between `# include "Enumerations.H"` on line 27 and `# ifdef WIN32` and include `# define WIN32` like this:

...

```

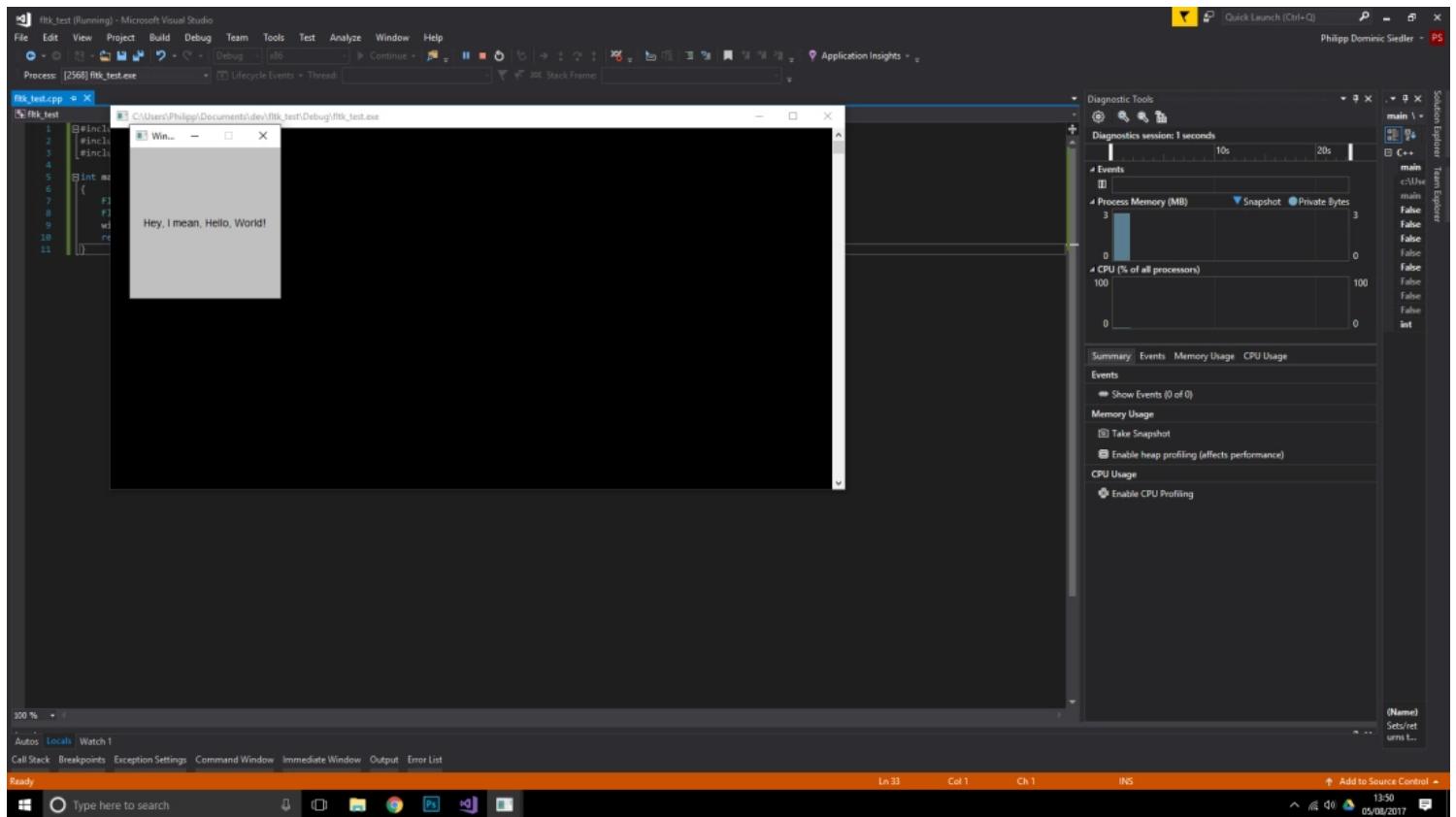
# include "Enumerations.H"
# define WIN32
# ifdef WIN32
...
```

```

21 // need to call Xlib directly. These symbols m
22 // systems.
23
24 #if !defined(FL_X_H) && !defined(FL_DDOXYGEN)
25 # define FL_X_H
26
27 # include "Enumerations.H"
28 # define WIN32
29 # ifdef WIN32
30 #   include "win32.H"
31 # elif defined(__APPLE__)
32 #   include "mac.H"
33 # else
34 #   if defined(_ABIN32) || defined(_ABI64) //
35 #     pragma set woff 3322

```

Safe the file and you should be good to go, now: Hit F5 to test-run the project and it should look like this:

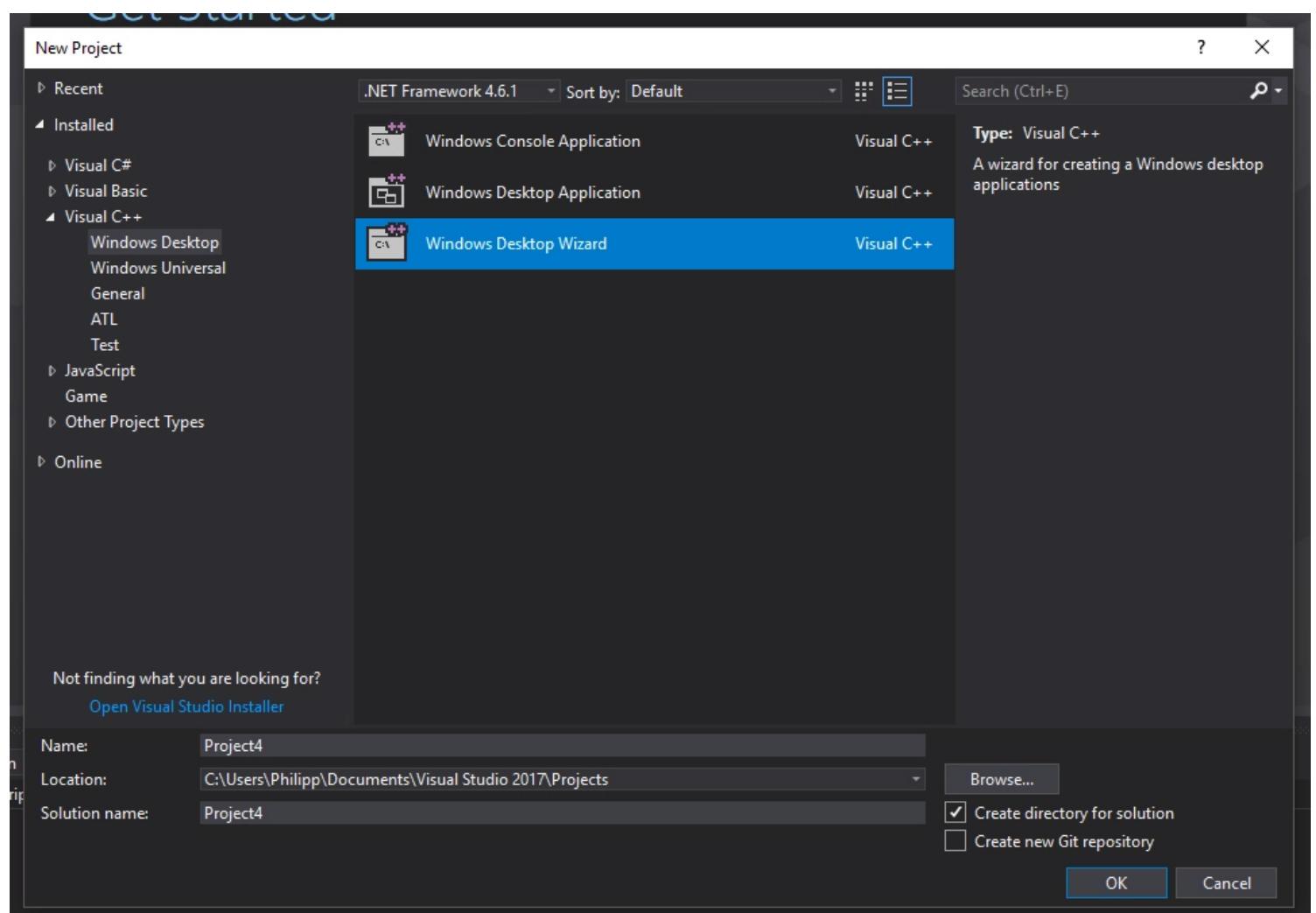
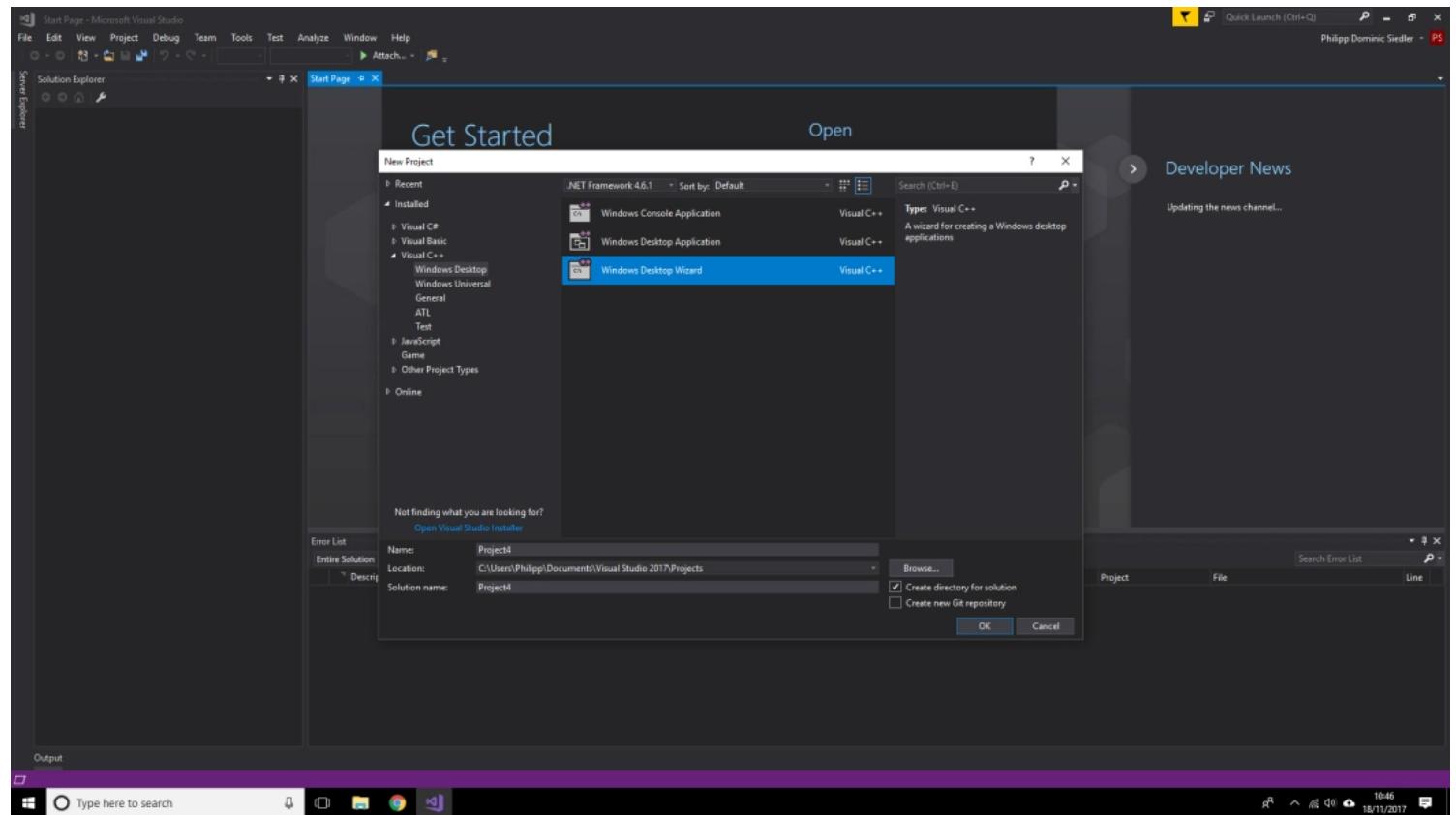


If you got to this point: **Congratulations, you made it!**

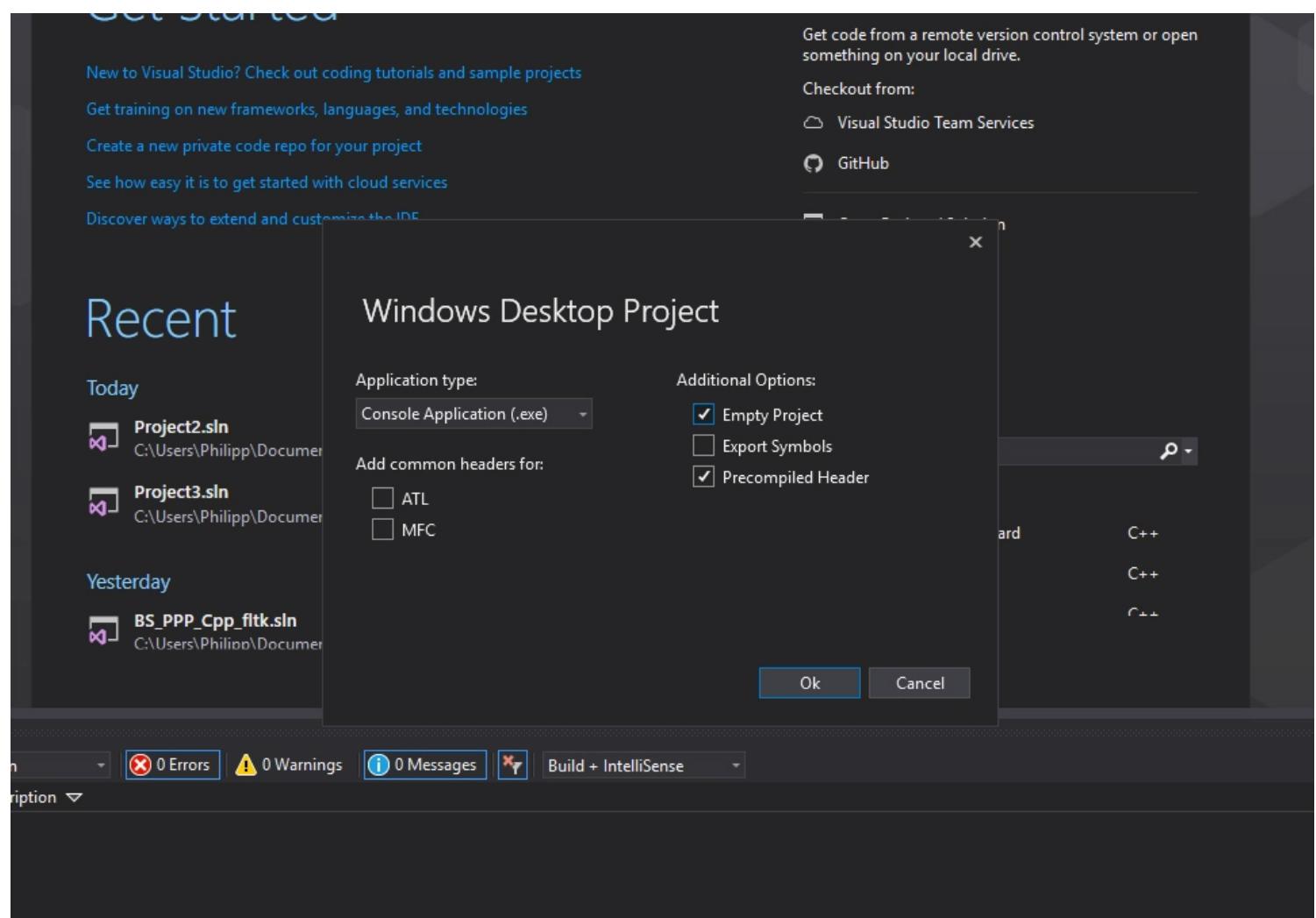
For who ever was interested in just getting the **fltk-1.3.4** GUI library running and applying it on a test project, this is the end of your journey and this guide. If you are working through Bjarne Stroustrup's Programming Principles and Practice Using C++ and you are in Chapter 12 starting to use fltk keep on reading for instructions on how to use it within the "Stroustrup"-environment:

Step 4: Using fltk-1.3.4 with Bjarne Stroustrup's header and .cpp files.

Safe your fltk test-project and create a new project. Under the **Visual C++ – Windows Desktop** tab create a new **Windows Desktop Wizard** project, just like you did before (I will go through this a bit faster as you should be familiar by now).

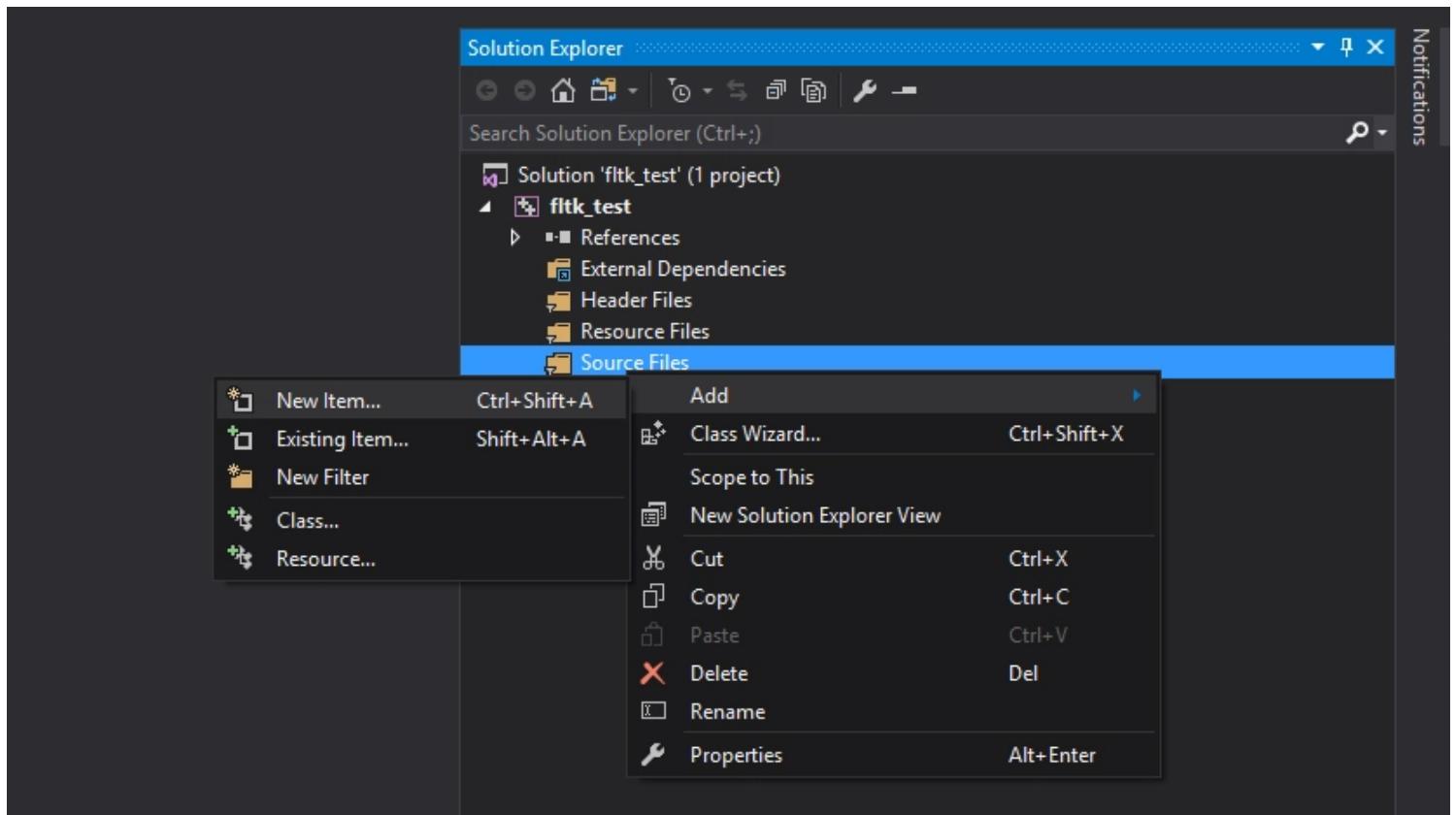


Tick the box **Empty project** (very important).



I called mine BS_PPP_Cpp_fltk.

Right-click the **Source Files** folder in your **Solution Explorer** and **Add a New Item**. Create a new .cpp file, I called mine p437_d_1_.cpp.



Copy and paste the following code from Bjarne Stroustrup's Programming Principles and Practice Using C++, Drill 1 page 437:

```

1 #include "Graph.h"
2 #include "Simple_window.h"
3
4 int main()
5 {
6     using namespace Graph_lib;
7
8     Point tl(150, 150);
9     Simple_window win(tl, 600, 400, "My window");
10    win.wait_for_button();
11 }
```

Go to **Project** and **Properties** and setup your project in the exact same way we did with our fltk test-project above.

Now this time, we have to add a couple of header files from Bjarne Stroustrup, and also some additional .cpp files.

Let's begin with the **header files**:

Most of the people reading, working through Programming Principles and Practice Using C++ included the **std_lib_facilities.h** already many times so for who ever does not know where to get it from, this is the [link \(<http://www.stroustrup.com/Programming/PPP2code/>\)](http://www.stroustrup.com/Programming/PPP2code/). Download it and copy it in your project folder (in my case: C:\Users\Philipp\Documents\Visual Studio 2017\Projects\BS_PPP_Cpp_fltk\BS_PPP_Cpp_fltk).

As well as the following header files:

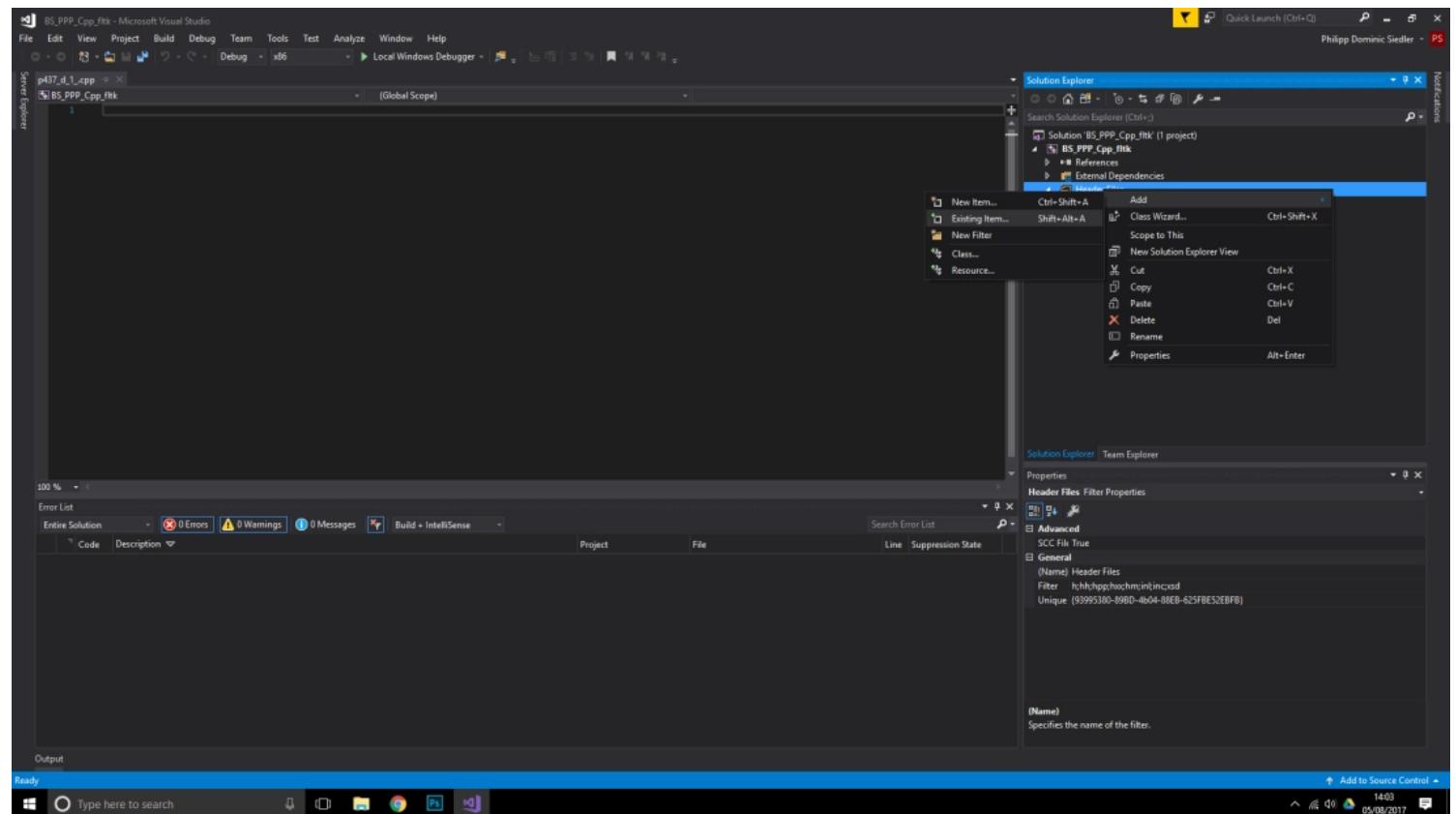
std_lib_facilities.h

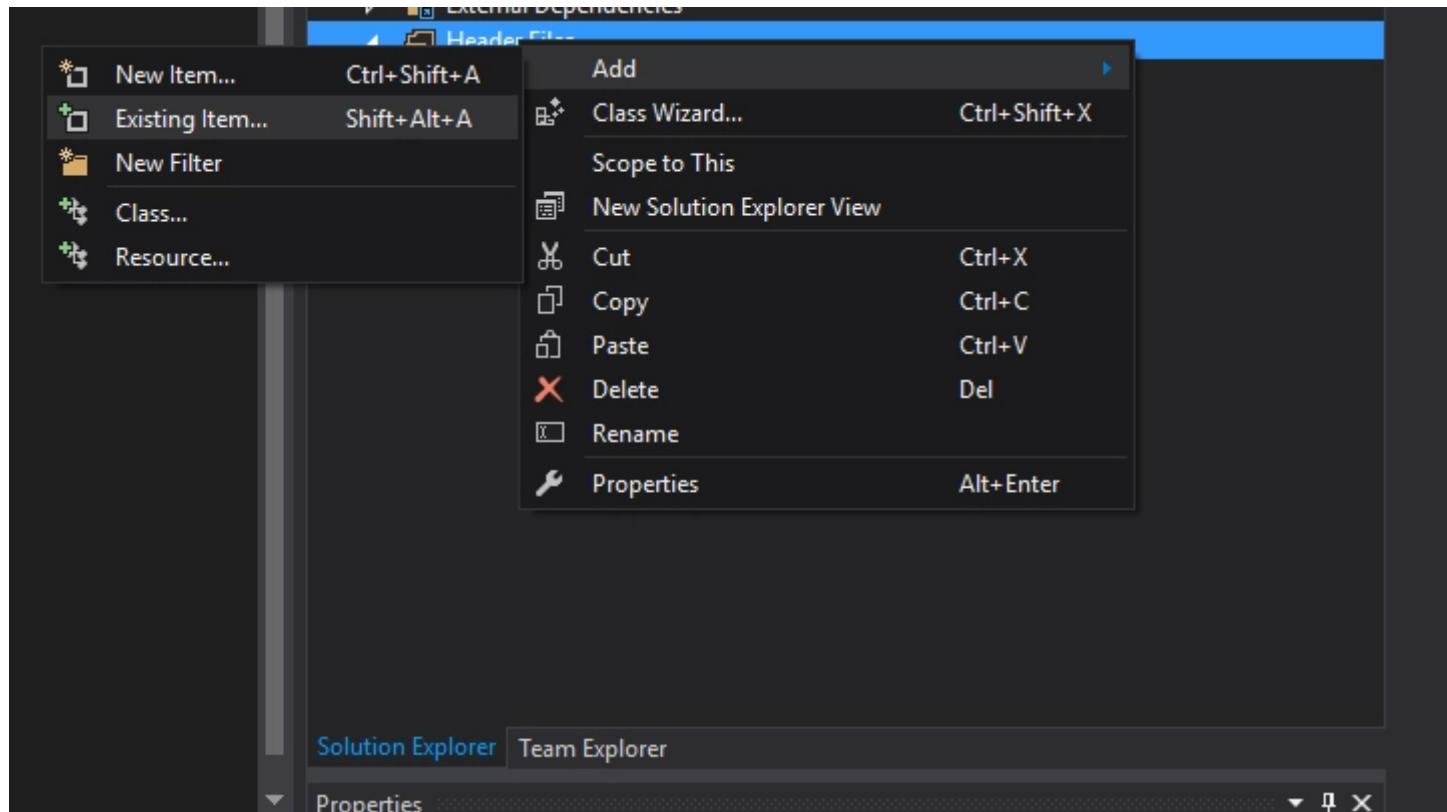
fltk.h**Graph.h****GUI.h****Point.h****Simple_window.h****Window.h**

and while you're on it also download and copy the following .cpp files into your project directory (as mentioned above):

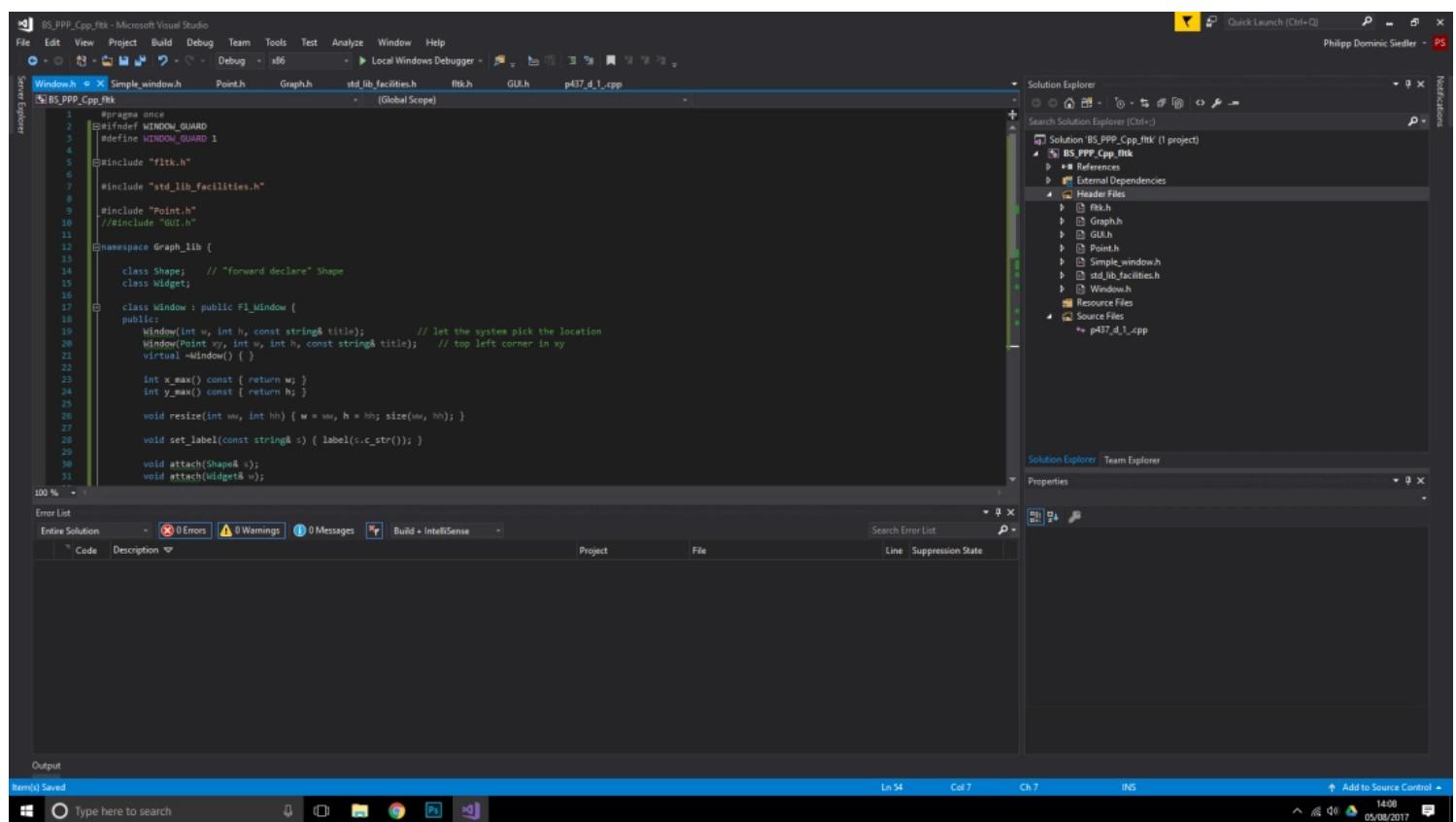
Graph.cpp**GUI.cpp****Window.cpp**

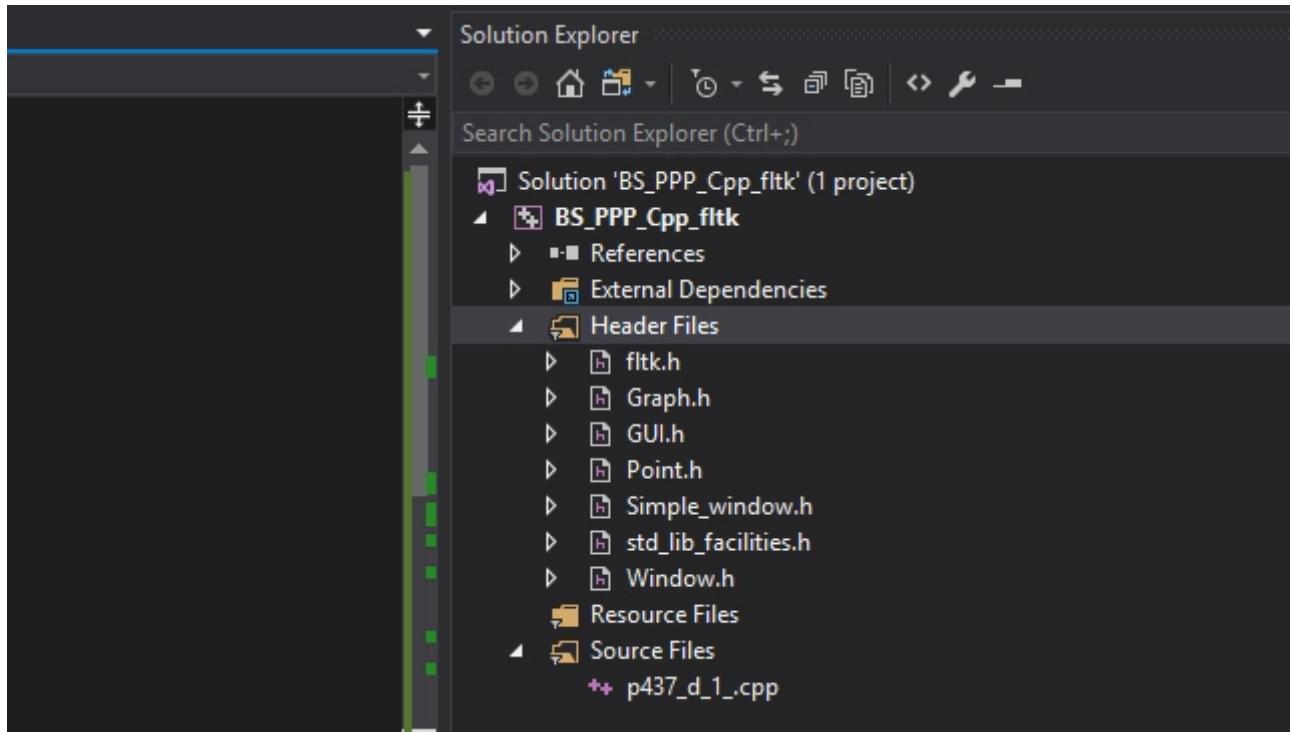
Right-click the **Header Files** folder in your project's solution explorer. **Add an Existing Item**.



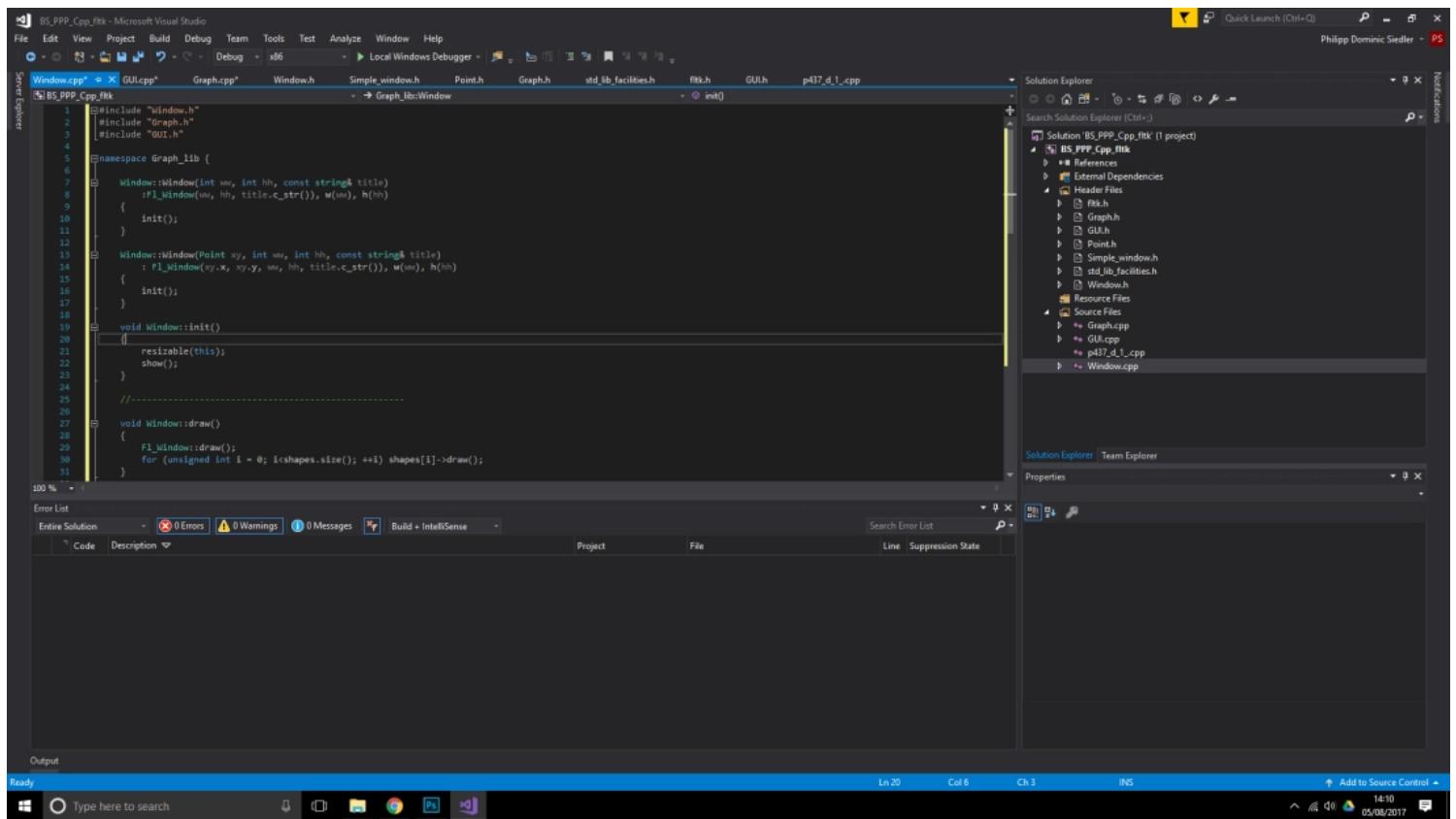


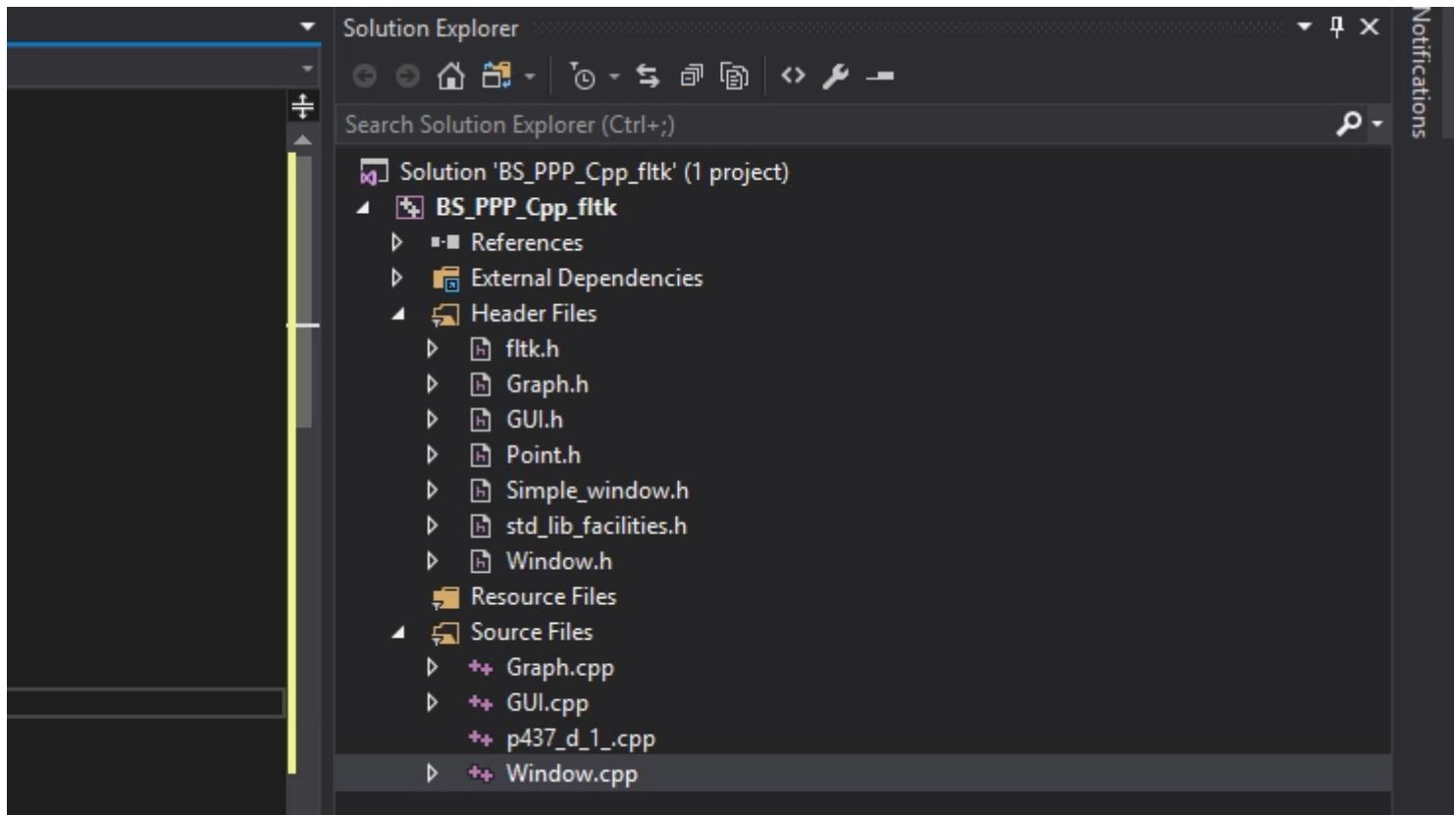
Once you added all your header files the **Header Files** folder should look like this:



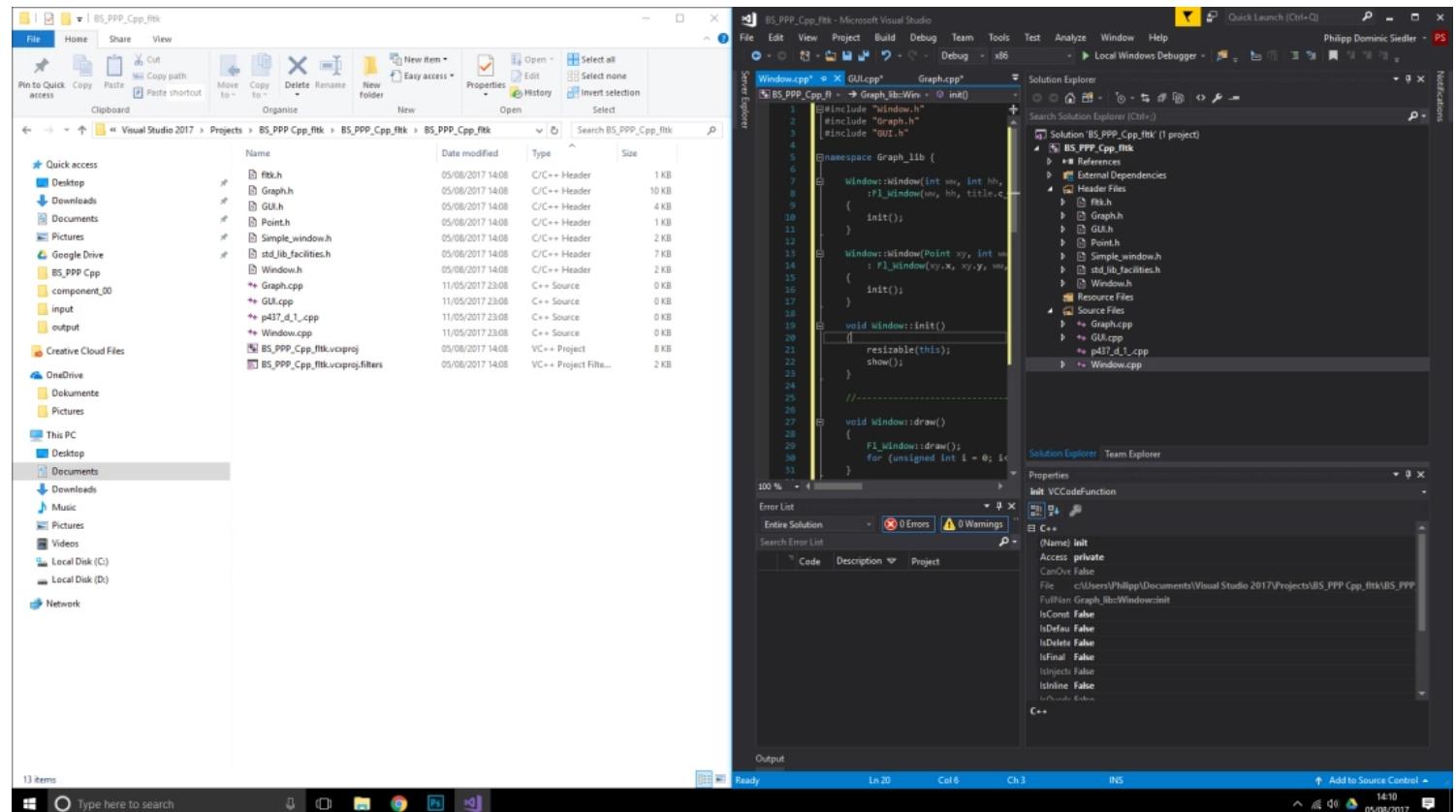


Now also add the .cpp files: **Graph.cpp**, **GUI.cpp** and **Window.cpp** to your **Source Files** folder. Your Source Files folder should look like this:



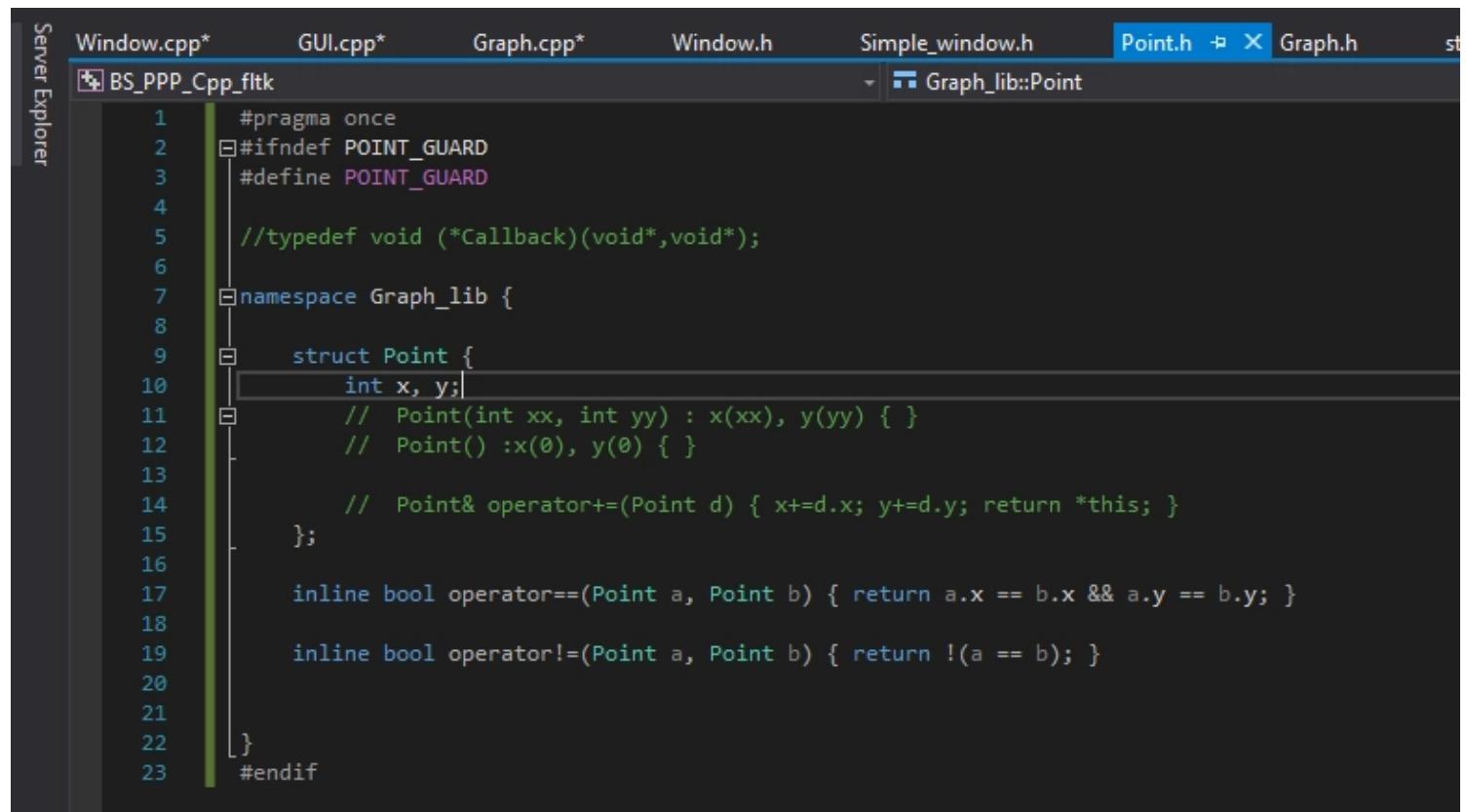
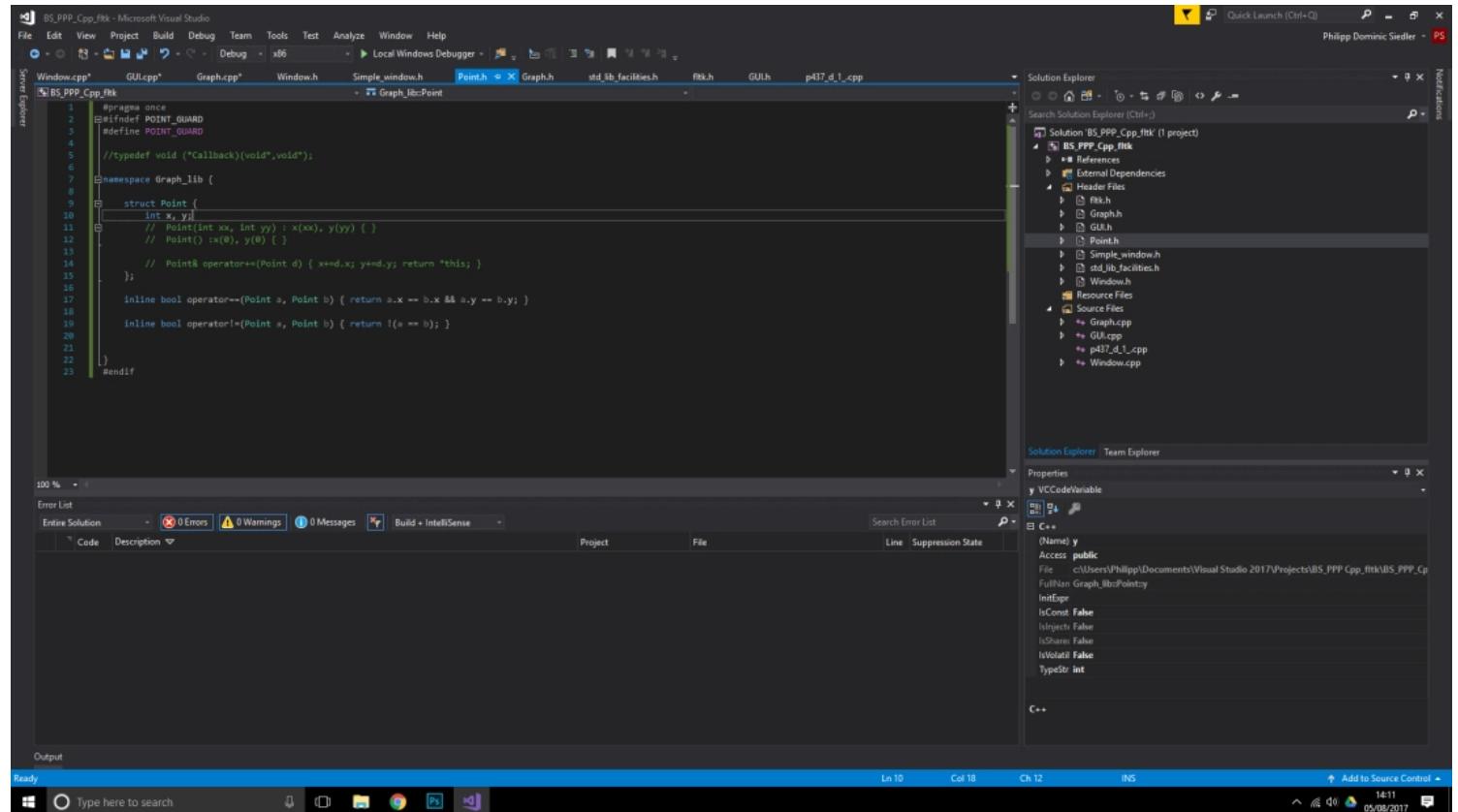


Your project directory folder and your **Solution Explorer** in Visual Studio 2017 should look like this:



Now you are almost there. We just have to make a couple of changes to those Added files.

Open the **Point.h** header file by double clicking it:



Remove all the notation double forward-slashes:

```

1 #pragma once
2 #ifndef POINT_GUARD
3 #define POINT_GUARD
4
5 typedef void (*Callback)(void*,void*);
6
7 namespace Graph_lib {
8
9     struct Point {
10         int x, y;
11         Point(int xx, int yy) : x(xx), y(yy) { }
12         Point() : x(0), y(0) { }
13
14         Point& operator+=(Point d) { x+=d.x; y+=d.y; return *this; }
15     };
16
17     inline bool operator==(Point a, Point b) { return a.x == b.x && a.y == b.y; }
18
19     inline bool operator!=(Point a, Point b) { return !(a == b); }
20
21 }
22
23 #endif

```

Open the **GUI.cpp** file by double clicking it:

```

1 #include "BS_PPP.h"
2 #include "std_lib_facilities.h"
3 #include <iostream>
4
5 using namespace Graph_lib;
6
7 void Button::attach(Window& win)
8 {
9     pw = new Fl_Button(loc.x, loc.y, width, height, label.c_str());
10    pw->callback(reinterpret_cast<Fl_Callback*>(do_it), &win); // pass the window
11    own = &win;
12 }
13
14 int In_box::get_int()
15 {
16     Fl_Input& pi = reference_to<Fl_Input>(pw);
17     // return atoi(pi.value());
18     const char* p = pi.value();
19     if (!isdigit(p[0])) return -99999;
20     return atoi(p);
21 }
22
23 string In_box::get_string()
24 {
25     Fl_Input& pi = reference_to<Fl_Input>(pw);
26     return string(pi.value());
27 }
28
29 void In_box::attach(Window& win)
30 {
31     pw = new Fl_Input(loc.x, loc.y, width, height, label.c_str());
32     own = &win;
33 }
34
35 void Out_box::put(int i)
36 {
37     Fl_Output& po = reference_to<Fl_Output>(pw);

```

Add **Graph_lib::** in front of **Window&** of **Button::attach** (line 8)

The screenshot shows the Visual Studio IDE with the 'GUI.cpp' file open. The code snippet is as follows:

```

1  #include "GUI.h"
2  #include "std_lib_facilities.h"
3  #include <sstream>
4
5  using namespace Graph_lib;
6
7
8  void Button::attach(Window& win)
9  {
10    pw = new Fl_Button(loc.x, loc.y, width, height);
11    pw->callback(reinterpret_cast<Fl_Callback*>(button_callback));
12    own = &win;
13 }

```

In_box::attach (line 30) and **Out_box::attach** (line 49).

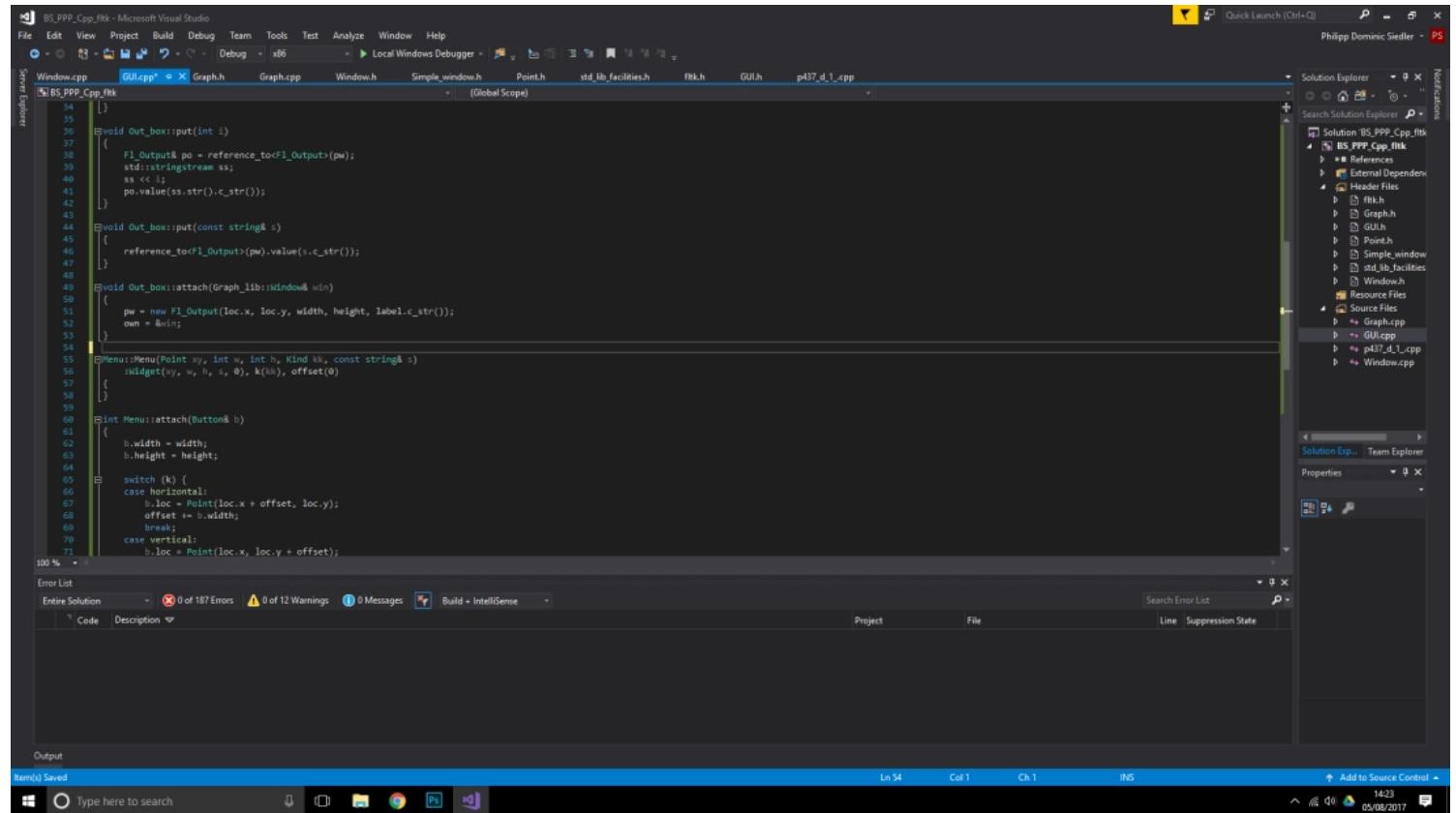
The screenshot shows the Visual Studio IDE with the 'GUI.cpp' file open. The code snippet is as follows:

```

29
30  void In_box::attach(Graph_lib::Window& win)
31  {
32    pw = new Fl_Input(loc.x, loc.y, width, height, label.c_str());
33    own = &win;
34  }
35
36  void Out_box::put(int i)
37  {
38    Fl_Output& po = reference_to<Fl_Output>(pw);
39    std::stringstream ss;
40    ss << i;
41    po.value(ss.str().c_str());
42  }
43
44  void Out_box::put(const string& s)
45  {
46    reference_to<Fl_Output>(pw).value(s.c_str());
47  }
48
49  void Out_box::attach(Graph_lib::Window& win)
50  {

```

and remove or note the constructor **Menu::Menu** as it is already defined in **GUI.h**.



```

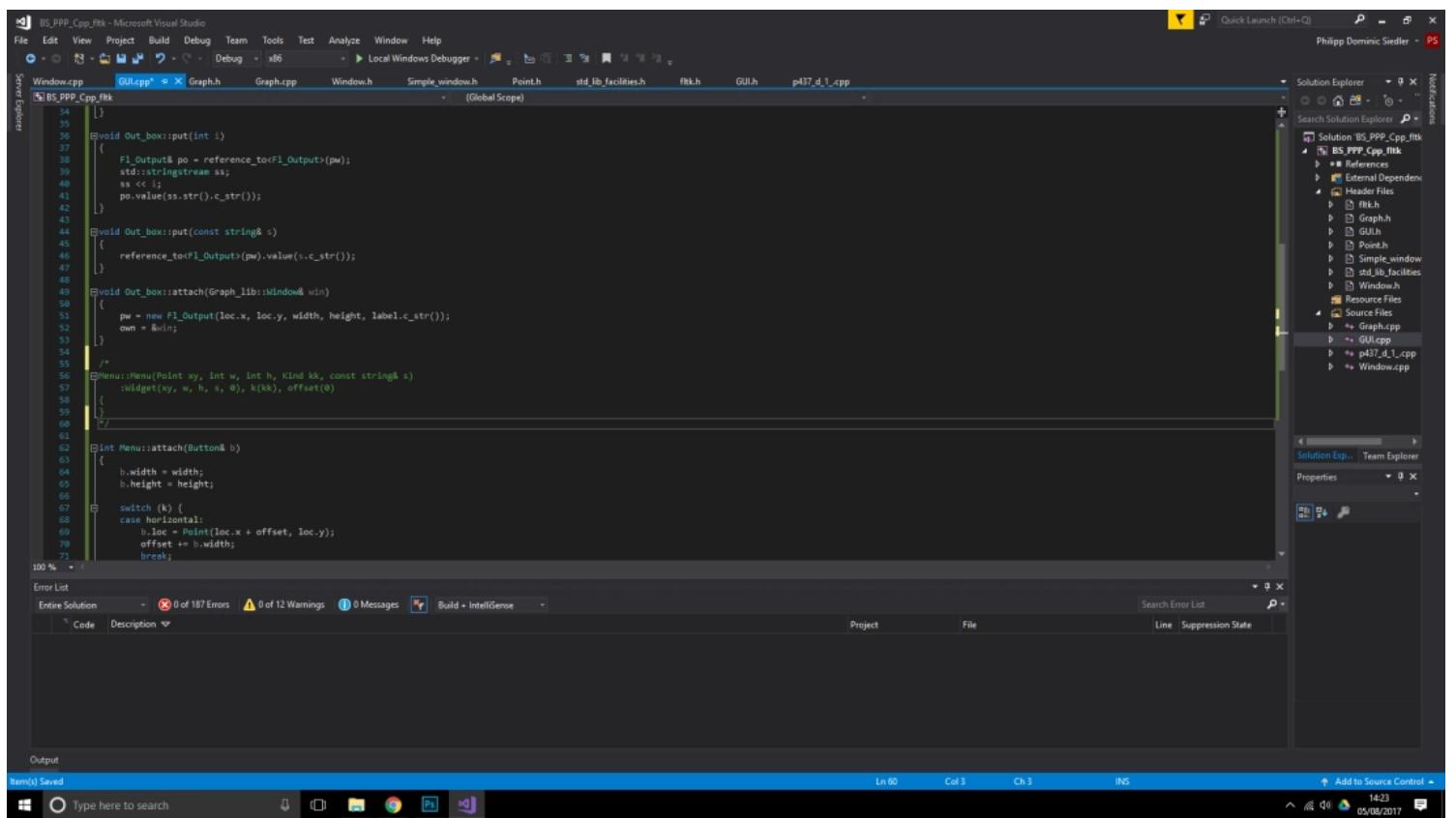
45     {
46         reference_to<Fl_Output>(pw).value(s.c_str());
47     }
48
49     void Out_box::attach(Graph_lib::Window& win)
50     {
51         pw = new Fl_Output(loc.x, loc.y, width, height, label.c_str())
52         own = &win;
53     }
54
55     Menu::Menu(Point xy, int w, int h, Kind kk, const string& s)
56         :Widget(xy, w, h, s, 0), k(kk), offset(0)
57     {
58     }
59
60     int Menu::attach(Button& b)
61     {
62         b.width = width;
63         b.height = height;
64         switch (k) {
65             case horizontal:
66                 b.loc = Point(loc.x + offset, loc.y);
67                 offset += b.width;
68                 break;
69             case vertical:
70                 b.loc = Point(loc.x, loc.y + offset);
71         }
72     }

```

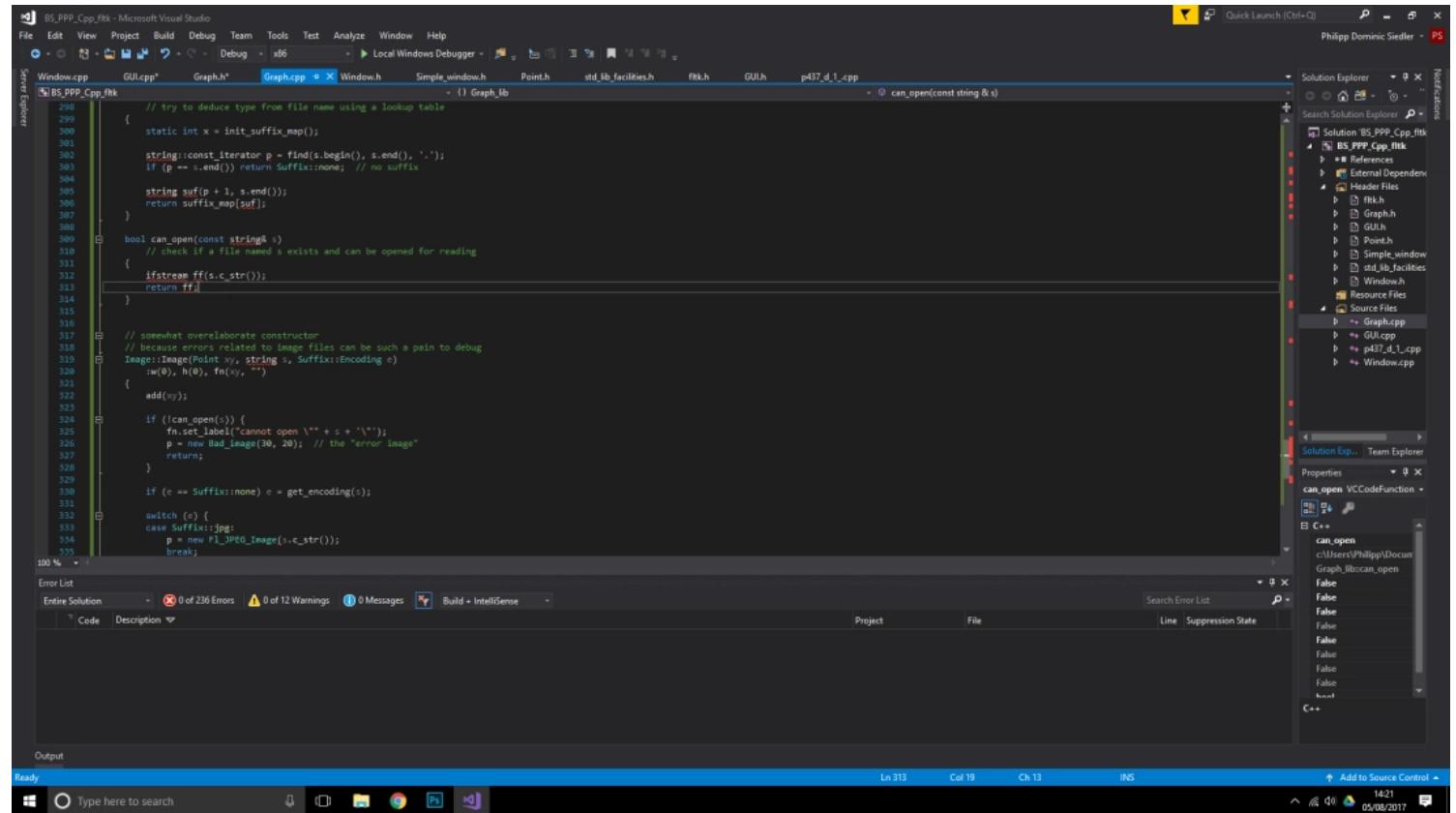
```

47  []
48
49  void Out_box::attach(Graph_lib::Window& win)
50  {
51      pw = new Fl_Output(loc.x, loc.y, width, height, label.c_str());
52      own = &win;
53  }
54
55  /*
56  Menu::Menu(Point xy, int w, int h, Kind kk, const string& s)
57      :Widget(xy, w, h, s, 0), k(kk), offset(0)
58  {
59  }
60 */
61
62  int Menu::attach(Button& b)
63  {
64      b.width = width;
65      b.height = height;
66

```



Open the **Graph.cpp** file:



And go to line 313. Change **return ff;** to **return bool(ff);**

```

307     }
308
309     bool can_open(const string& s)
310         // check if a file named s exists
311     {
312         ifstream ff(s.c_str());
313         return ff;
314     }
315
316
317     // somewhat overelaborate constructor
318     // because errors related to image files can be such a pain to debug
319     Image::Image(Point xy, string s, Suffix::Encoding e)
320         :w(0), h(0), fn(xy, "")
```

```

307     }
308
309     bool can_open(const string& s)
310         // check if a file named s exists
311     {
312         ifstream ff(s.c_str());
313         //return ff;
314         return bool(ff);
315     }
316
317
318     // somewhat overelaborate constructor
319     // because errors related to image files can be such a pain to debug
320     Image::Image(Point xy, string s, Suffix::Encoding e)
321         :w(0), h(0), fn(xy, "")
```

The last change you have to make is in `Graph.h`, open and `#include "std_lib_facilities.h"`:

The screenshot shows the Microsoft Visual Studio interface for a C++ project named "BS_PPP_Cpp_fltk".

Code Editor: The main window displays the file "gui.cpp" with the following content:

```
1 // sprague once
2 #ifndef GRAPH_GUARD
3 #define GRAPH_GUARD
4
5 #include "Point.h"
6 #include <vector>
7 #include <std::facilities.h>
8
9 // #include <string>
10 // #include <cmath>
11 // #include <fltk.h>
12 // #include <std::lib_facilities.h>
13
14 namespace GraphLib {
15     // defense against ill-behaved Linux macros
16     #undef major
17     #undef minor
18
19     struct Color {
20         enum Color_type {
21             red = FL_RED, blue = FL_BLUE, green = FL_GREEN,
22             yellow = FL_YELLOW, white = FL_WHITE, black = FL_BLACK,
23             magenta = FL_MAGENTA, cyan = FL_CYAN, dark_red = FL_DARK_RED,
24             dark_green = FL_DARK_GREEN, dark_yellow = FL_DARK_YELLOW, dark_blue = FL_DARK_BLUE,
25             dark_magenta = FL_DARK_MAGENTA, dark_cyan = FL_DARK_CYAN
26         };
27         enum Transparency { invisible = 0, visible = 255 };
28
29         Color(Color_type cc) : cf(FL_Color(cc)), v(visible) { }
30         Color(Color_type cc, Transparency vv) : cf(FL_Color(cc)), v(vv) { }
31         Color(int cc) : cf(FL_Color(cc)), v(visible) { }
32         Color(Transparency vv) : cf(FL_Color()), v(vv) { }
33
34         int as_int() const { return cc; }
35         char visibility() const { return vv; }
36         void set_visibility(Transparency vv) { v = vv; }
37     private:
38         unsigned char v; // 0 or 1 for now
39     };
40
41     Color(cc) : cf(FL_Color(cc)), v(visible) { }
42     Color(cc, vv) : cf(FL_Color(cc)), v(vv) { }
43     Color(cc) : cf(FL_Color(cc)), v(visible) { }
44     Color(vv) : cf(FL_Color()), v(vv) { }
45
46     int as_int() const { return cc; }
47     char visibility() const { return vv; }
48     void set_visibility(Transparency vv) { v = vv; }
49
50     private:
51         unsigned char v; // 0 or 1 for now
52     };
53 }
```

Solution Explorer: Shows the project structure:

- BS_PPP_Cpp_fltk
- References
- External Dependencies
- Header Files
- fltk
- Graph
- GUI
- Point
- Simple_window
- std.lib_facilities
- Window
- Source Files
 - Graph.cpp
 - GUI.cpp
 - p437_d_1.cpp
 - Window.cpp

Error List: One error is listed:

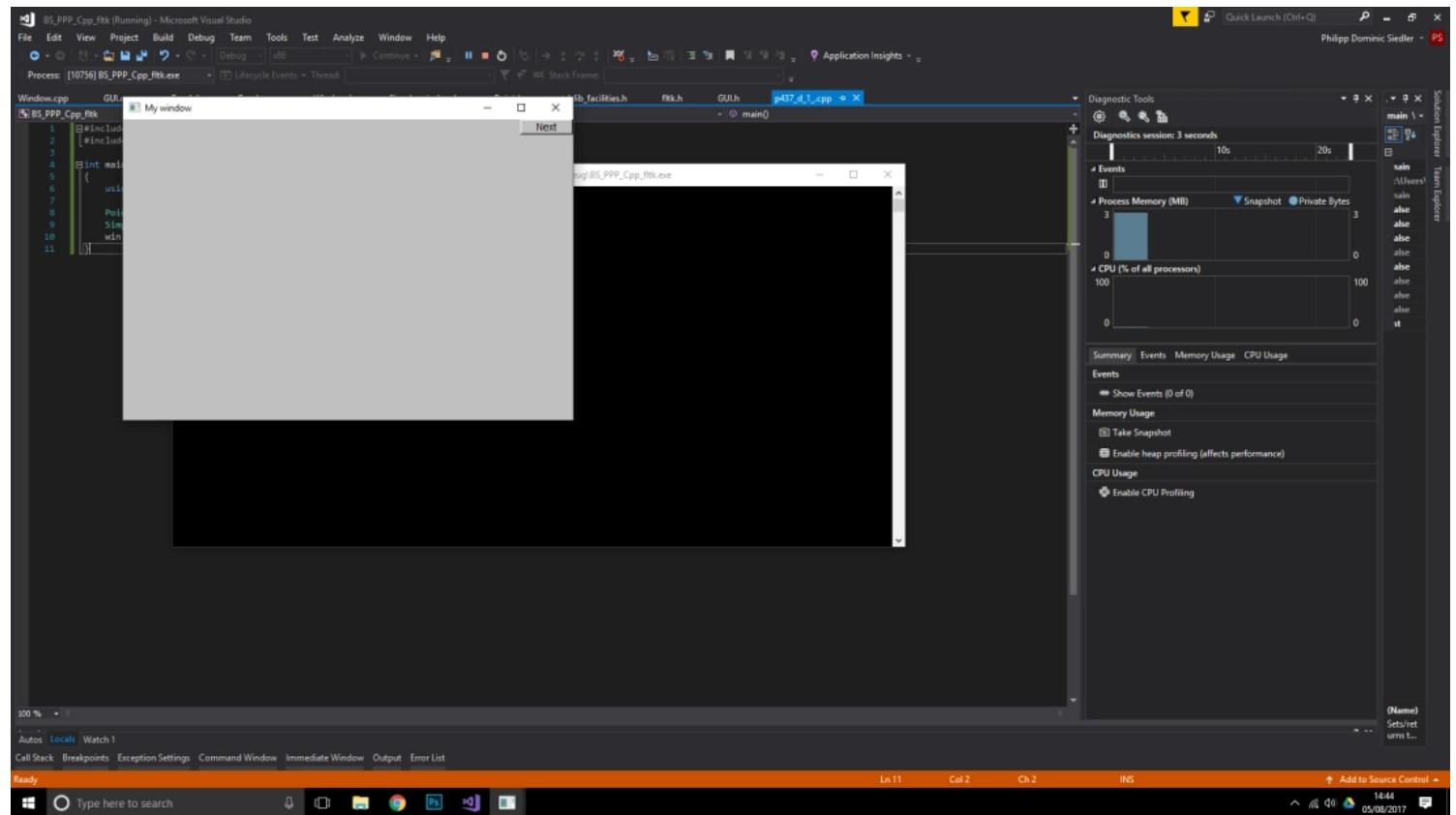
Code	Description	Project	File	Line	Suppression State
C2044	function 'GraphLib::Menu::Menu(GraphLib::Point,int,GraphLib::Menus::Kind,const std::string&)' already has a body	BS_PPP_Cpp_fltk	gui.cpp	56	

Output: Ready

```
1 #pragma once
2 #ifndef GRAPH_GUARD
3 #define GRAPH_GUARD 1
4
5 #include "Point.h"
6 #include <vector>
7 #include "std_lib_facilities.h"
8
9 // #include <string>
10 // #include <cmath>
11 #include "fltk.h"
12 // #include "std_lib_facilities.h"
13
```

This is it. Now save all your files and run your project. I like to just hit F5.

If everything goes right and you were able to follow my instructions you should be able to see the following:



CONGRATULATION, ALL DONE!

If you made it so far and everything worked out I am very happy I could help you. At this point I should mention Benjamin Wuethrich. I followed his tutorial on installing and using fltk on Visual Studio 2015 and with the help of a bunch of other peoples forum posts and other informations I was able to transfer the knowledge to make it happen on Visual Studio 2017 Community. So thanks to Benjamin Wuethrich. His tutorial for Visual Studio 2015 can be found [here](https://bewuethr.github.io/installing-fltk-133-under-visual-studio/) (<https://bewuethr.github.io/installing-fltk-133-under-visual-studio/>).

Thank you for going through this with me, if you have any questions I am very happy to help. Also if you have any suggestions for improvement, please let me know.

Tagged:

Bjarne Stroustrup,
C++,
FLTK,
Principles and Practice Using C++

Published by bumpyroadtocode



[View all posts by bumpyroadtocode](#)