wxWidgets: A New Way To See C++

By: Chase Lichtman

C++ is a very expansive programming language that lets the user do many different things. However, the standard library does have its own limitations and is not able to allow the user t sometimes to expand their horizons on their programs. That is where libraries come in, to support C++ and make the users experience more diverse. One of the most expansive libraries that C++ has to offer is wxWidgets. wxWidgets allows a whole new world of C++! Instead of having outputs only to the terminal, we are now able to make screens and graphics and are now able to display high levels of code! Downloading wxWidgets was a bit of a hassle. I downloaded it from the website, https://wxwidgets.org/downloads/ and unzipped it. With that on my computer.

To get it working and compiled I followed my professor's guide to how to set it up on my IDE CLion while also using a CMAKE file.

The functions in the wxWidgets library are very straightforward and are easy to incorporate into your code. I will show a few functions and their uses later in this post.

We need to first create our window that we want to show. I have created a frame class and added an Initialize function to it. In it, we have called the function Create. This allows us to create a window with our specific parameters. We can call the window whatever we want, and determine how big the window will end up being.

Frame.h Frame.cpp

```
 */
#include "pch.h"
#include "Frame.h"
/**
 * Initialize the Frame window.
 */

void Frame::Initialize()
{
    Create( parent: nullptr,  id: wxID_ANY,  title: L"Chase Andrew Lichtman",
            pos: wxDefaultPosition,  size: wxSize(  xx: 1000, yy: 800 ));


}

```

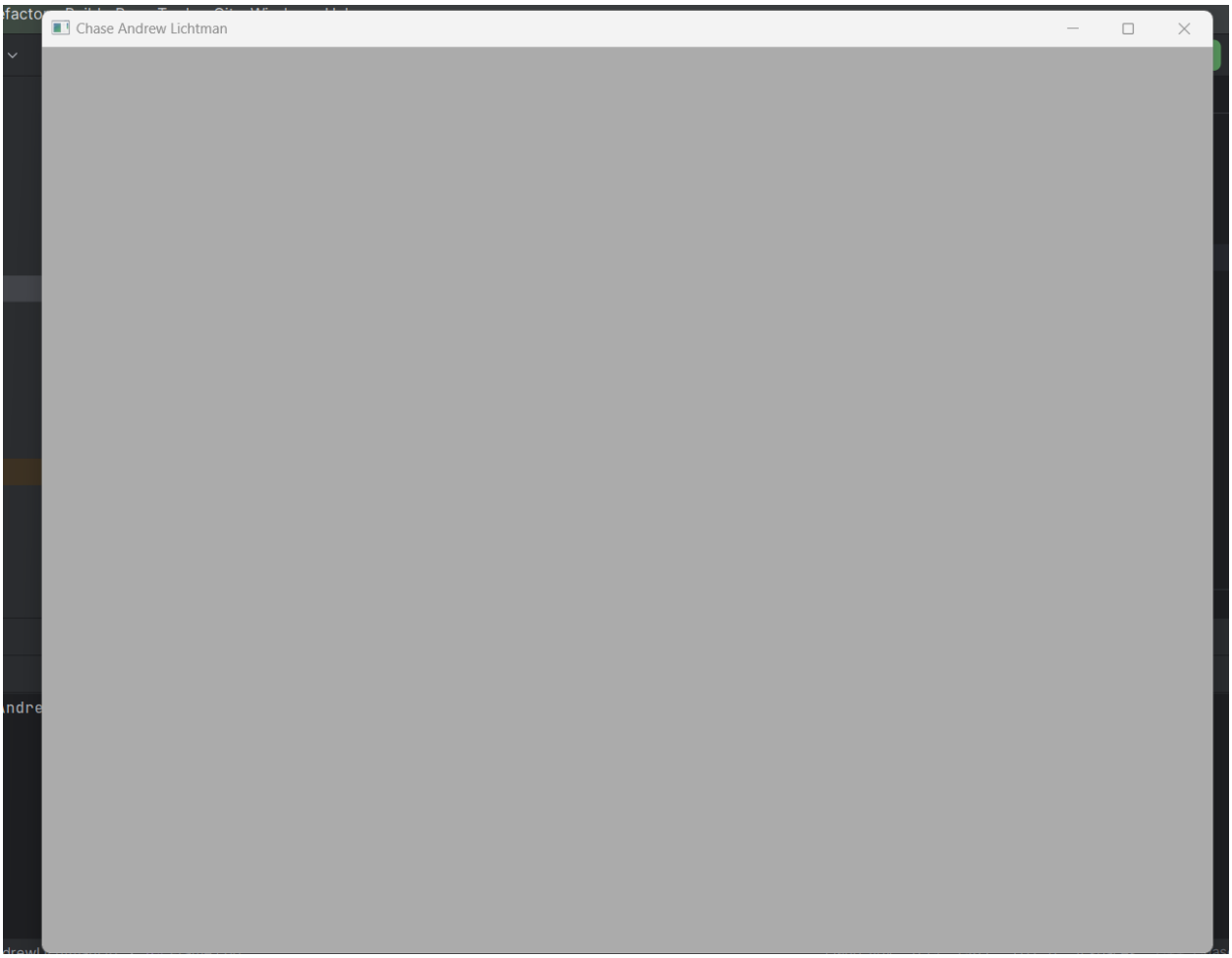
With a Frame class and an Initialize function made, we can now start making a new frame to try and make a popup window. The show function is apart of the wxWindows class from the library that lets us show the window.

```
#include "ChaseClass.h"
#include <Frame.h>
/**
 * Initializes the new frame
 * @return
 */
bool ChaseClass::OnInit()
{
    if (!wxApp::OnInit())
        return false;

    auto frame = new Frame();
    frame->Initialize();
    frame->Show( show: true);

    return true;
}
```

With that being set, we can now run our program. wxWidgets makes it very easy to run without needing a main function. In our main.cpp, we just need to call wxIMPLEMENT_APP with our class name and that will run the program.

```
wxIMPLEMENT_APP(ChaseClass);
```

With that running we will get a popup window! We have officially ditched the terminal and have a whole new world to display our code and make new and exciting programs.



We are now able to use our imaginations and create whatever seen we want. wxWidgets has plenty of classes allowing us to showcase and display our creative minds. There are pens to draw out lines and functions like wxColour that can change the color of those lines. We are also able to change the background with the function SetBackground(*wx…) with whatever color we want replacing the ellipses all in caps (EX. wxYELLOW).

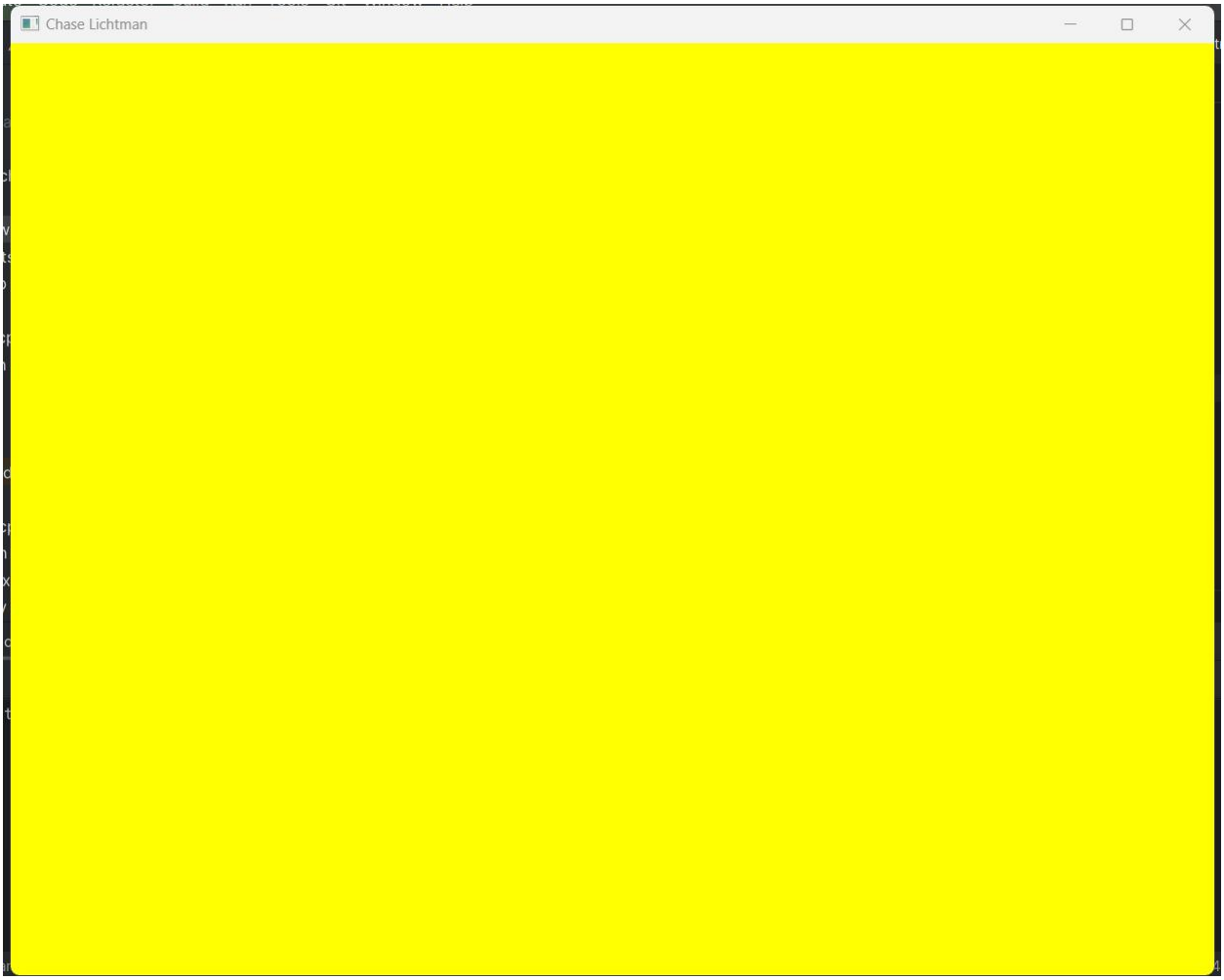
In this example I have made a new view class. Here we can initialize what we want to see on the screen and be able to paint and draw pictures. The bind function will then call our paint function. The wxAutoBufferedPaintDC is a huge object. This will act as our canvas and actually make it so we can add things to the screen. We then ask it to set the background to the color yellow.

```
15  void View::InitializeT(wxFrame* parent)
16  {
17      Create(parent, id: wxID_ANY);
18      SetBackgroundStyle( style: wxBG_STYLE_PAINT);
19      Bind( eventType: wxEVT_PAINT, method: &View::OnPaint, handler: this);
20
21  }
22  /**
23   * Paint event, draws the window.
24   * @param event Paint event object
25   */
26  void View::OnPaint(wxPaintEvent& event)
27  {
28      wxAutoBufferedPaintDC dc( win: this);
29
30      dc.SetBackground( brush: *wxYELLOW);
31      dc.Clear();
```

After declaring these functions, lets go back to our frame class and initialize our new view. After creating the frame lets make a new view object. We will then pass the frame that we just created into it and initialize the view.

```
void Frame::Initialize()
{
    Create( parent: nullptr, id: wxID_ANY, title: L"Chase Lichtman",
            pos: wxDefaultPosition, size: wxSize( xx: 1000, yy: 800 ));

    auto view = new View();
    view->InitializeT(this);

}
```

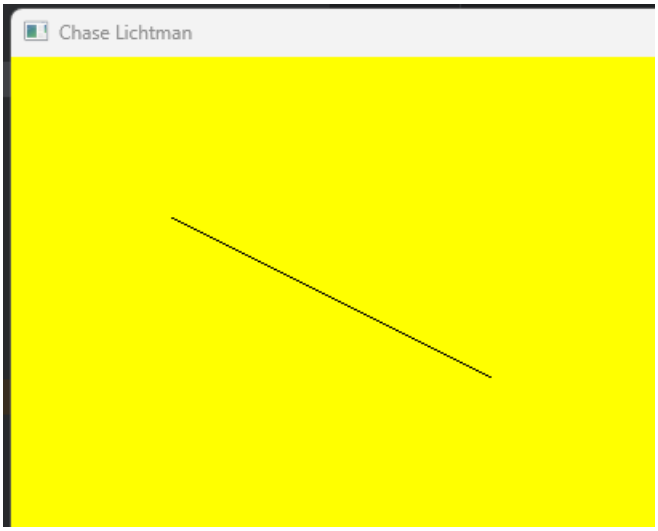
Once this is all done and we run again, we can now see that the entire background is yellow!



The last thing I want to show you is to draw something onto our background. With wxWidgets this is fairly simple. We just need to make a pen and set that pen to a color. Then we just need to call the Drawline function with the parameters of how long to make the line and where to place the line

```
wxPen pen( col: wxColour(*wxBLACK));
dc.SetPen(pen);
dc.DrawLine( x1: 100,  y1: 100,  x2: 300,  y2: 200);
```

Here is the results.



That is just the surface of what this library can accomplish. We can make full pictures with different shapes and all different colors plus text.

All in all, this library is pretty simple to set up and can get something popping up on our systems in a matter of minutes. The functions are straight forward to use an add to your code and their reference page is filled with detail and description of each function, plus thousands of other uses of this library that I did not even get to. I believe the hardest thing with this library is getting it set up and configured properly on your system (but that goes for all libraries that are not apart of the standard library). This library is a must go to for someone wanting to start getting into graphics and wants to elevate their C++ programming to a whole new level!