Task 12 - Lab Summary Report

Spike: Task_12
Title: SDL2 Concepts
Author: Thomas Horsley, 103071494

Goals & Deliverables

Aim: Develop a foundational knowledge surrounding the implementation of and functionality behind the SDL2 Library.

Deliverables:

- Functional code
- Lab Summary Report

Technology, Tools and Resources

Tech and Tools

The project was scripted in C++ 17 using Visual Studio Community 2022.

UML's and charts are made with *www.Lucidchart.com*

Source control is handled using Git.

Resources

- SDL2 Keyboard and Mouse Input https://www.youtube.com/watch?v=Gjhvz4banWA
- How to link SDL 2 with Visual Studio on Windows https://www.youtube.com/watch?v=tmGBhM8AEj8
- SDL - WikiBooks https://shorturl.at/rDOQ1
- SDL API Search https://wiki.libsdl.org/SDL2/APIByCategory

Task 12 - Lab Summary Report

Tasks Undertaken

Q&A

SDL (Simple DirectMedia Layer): https://www.libsdl.org/
License: zlib license

API Search: https://wiki.libsdl.org/SDL2/APIByCategory

SDL2 is written in C. Officially supported platforms include:

- Linux
- Windows / WCE
- BeOS
- MacOS/X
- FreeBSD
- NetBSD

The SDL code contains unofficial support for:

- AmigaOS
- Dreamcast
- Atari
- AIX
- OSF/Tru64
- RISC OS
- Symbian OS
- OS/2

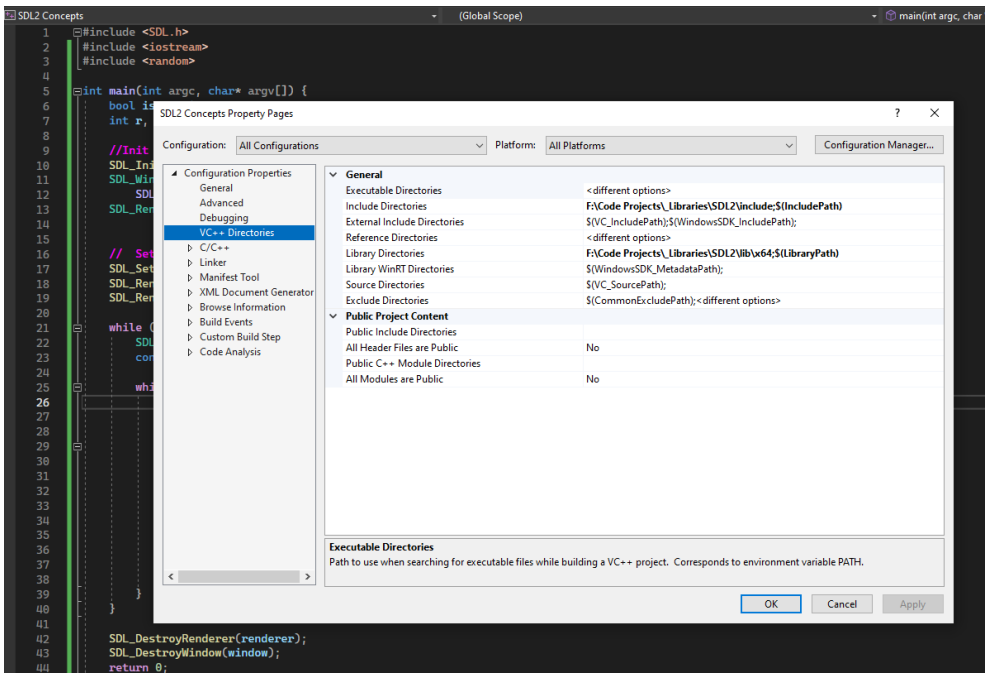
1. SDL can be downloaded as “SourceCode”, “Runtime Binaries” or “Development Libraries”. What is the difference?
    a. SourceCode refers to uncompiled code files which the programmer must compile before use. Whilst often difficult for large codebases it’s possible to extend on the base functionality of a Library, API or Framework.
    b. Binary Files are precompiled programs and therefore can only be used and not extended upon.
    c. The Development Libraries contain modularized code functionality which can be called upon by other code files in a project. These have to be linked with the project and will most often hold a .dll extension.

2. For the different download options, which one do you personally want to use with your IDE setup? (If you use a different way to setup SDL, such as a package manager, state what you have used or will use.)
    a. For my purposes, I will download SDL2 Development Libraries and configure them to work with Visual Studio using dynamic linking.

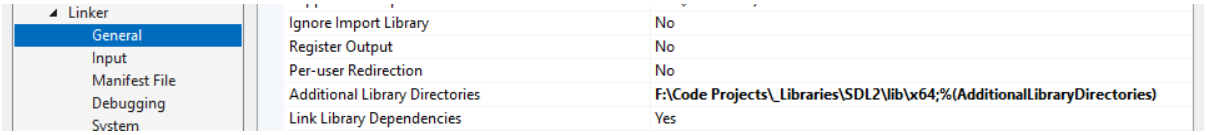
3. In simple terms, what are some of the differences between a multimedia library like SDL and a “game framework”? Think about what SDL is trying to provide compared to what a game framework tries to provide.
    a. By their name, a games Framework provides a developer the tools to implement auxiliary functionality such as audio and input handling without specifying how the functionality is implemented. SDL on the other-hand, uses a collection of functions and API calls to handle implementation and take some of the workload from the developer.

4. For your particular IDE and setup, what settings are needed to make your project work with SDL2?
    a. This part was very enjoyable…

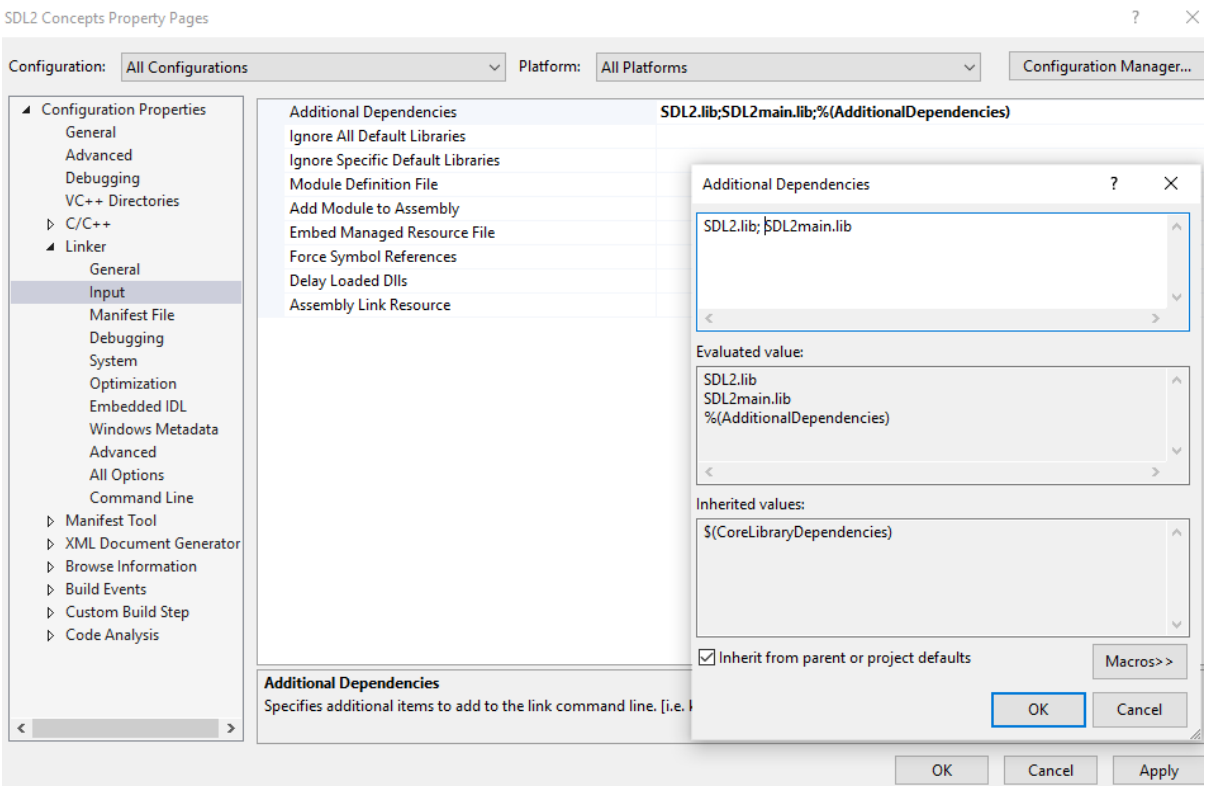
        I setup this project in Visual Studio 2022, luckily there’s plenty of resources available for VS in regards to SDL setup and implementation. After downloading the SDL2 dev library, and storing it in a separate directory (for reuse) it was only a matter of setting up references to the library and linking the necessary files.



Include and library paths for Visual C++ Dirs



I didn't set *Link Library Dependencies* to true. This caused a bit of headache.



Linker input changes.

5. What tutorial(s)/internet resource(s) did you find most useful when creating your demo program?
    a. Setting up input with keyboard and mouse: https://www.youtube.com/watch?v=Gjhvz4banWA.
    This video worked as an easy to follow launch point for learning about events and polling in SDL2
    b. Linking SDL2 with VS Community 2021: https://www.youtube.com/watch?v=tmGBhM8AEj8
    The only man wise enough to prompt dummies like me about the *Link Library Dependencies* button…
    c. SDL2 documentation is quite concise and a good reference when I needed it (E.g. referencing CreateWindow flags)

Task 12 - Lab Summary Report

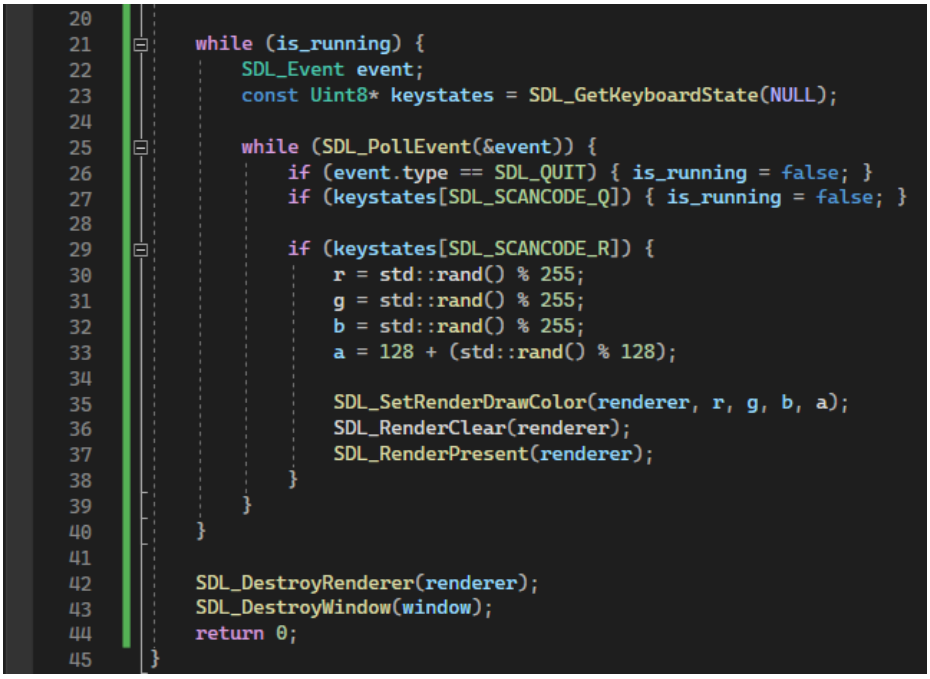
Implementation

Git Commit History

Code

```
5   int main(int argc, char* argv[]) {
6       bool is_running = true;
7       int r, g, b, a;
8
9       //Init SDL subsystems, window and renderer.
10      SDL_Init(SDL_INIT_EVERYTHING);
11      SDL_Window* window = SDL_CreateWindow("Task 12 - SDL2 Concepts, Thomas Horsley 103071494",
12          SDL_WINDOWPOS_CENTERED, SDL_WINDOWPOS_CENTERED, 600, 800, SDL_WINDOW_SHOWN);
13      SDL_Renderer* renderer = SDL_CreateRenderer(window, -1, 0);
14
```

Subsystem, window and renderer initializations



```
20
21      while (is_running) {
22          SDL_Event event;
23          const Uint8* keystates = SDL_GetKeyboardState(NULL);
24
25          while (SDL_PollEvent(&event)) {
26              if (event.type == SDL_QUIT) { is_running = false; }
27              if (keystates[SDL_SCANCODE_Q]) { is_running = false; }
28
29              if (keystates[SDL_SCANCODE_R]) {
30                  r = std::rand() % 255;
31                  g = std::rand() % 255;
32                  b = std::rand() % 255;
33                  a = 128 + (std::rand() % 128);
34
35                  SDL_SetRenderDrawColor(renderer, r, g, b, a);
36                  SDL_RenderClear(renderer);
37                  SDL_RenderPresent(renderer);
38              }
39          }
40      }
41
42      SDL_DestroyRenderer(renderer);
43      SDL_DestroyWindow(window);
44      return 0;
45  }
```

Changing colors on input and destroying the window and renderer when the loop ends

Note that the alpha channel for the color selection is bound to the range 0.5 - 1.

What was Learned?

After this lab I can source information about the SDL2 developer library quickly and efficiently as well as hold a baseline understanding of window and render instantiation / destruction and SDL2 events.
