

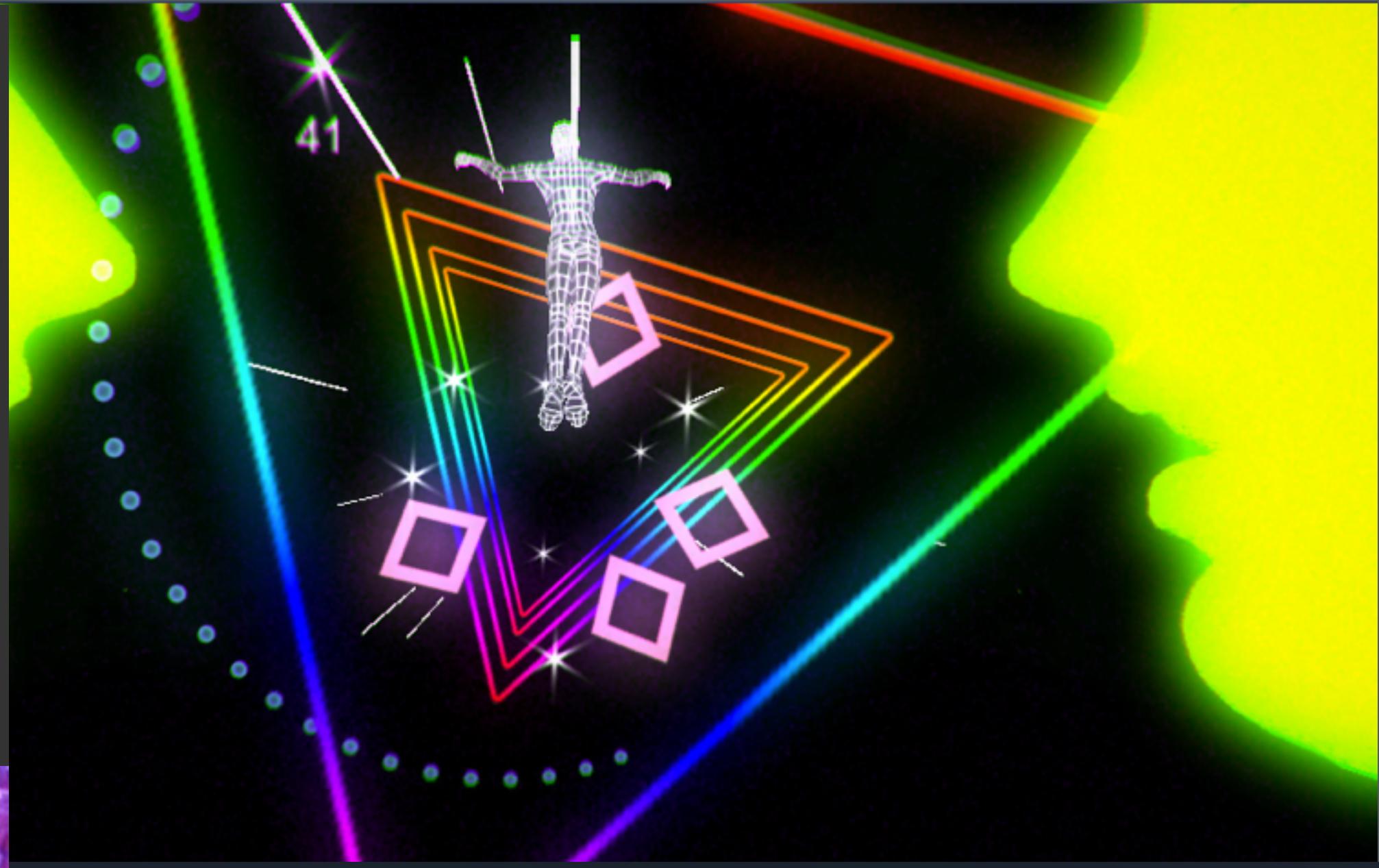
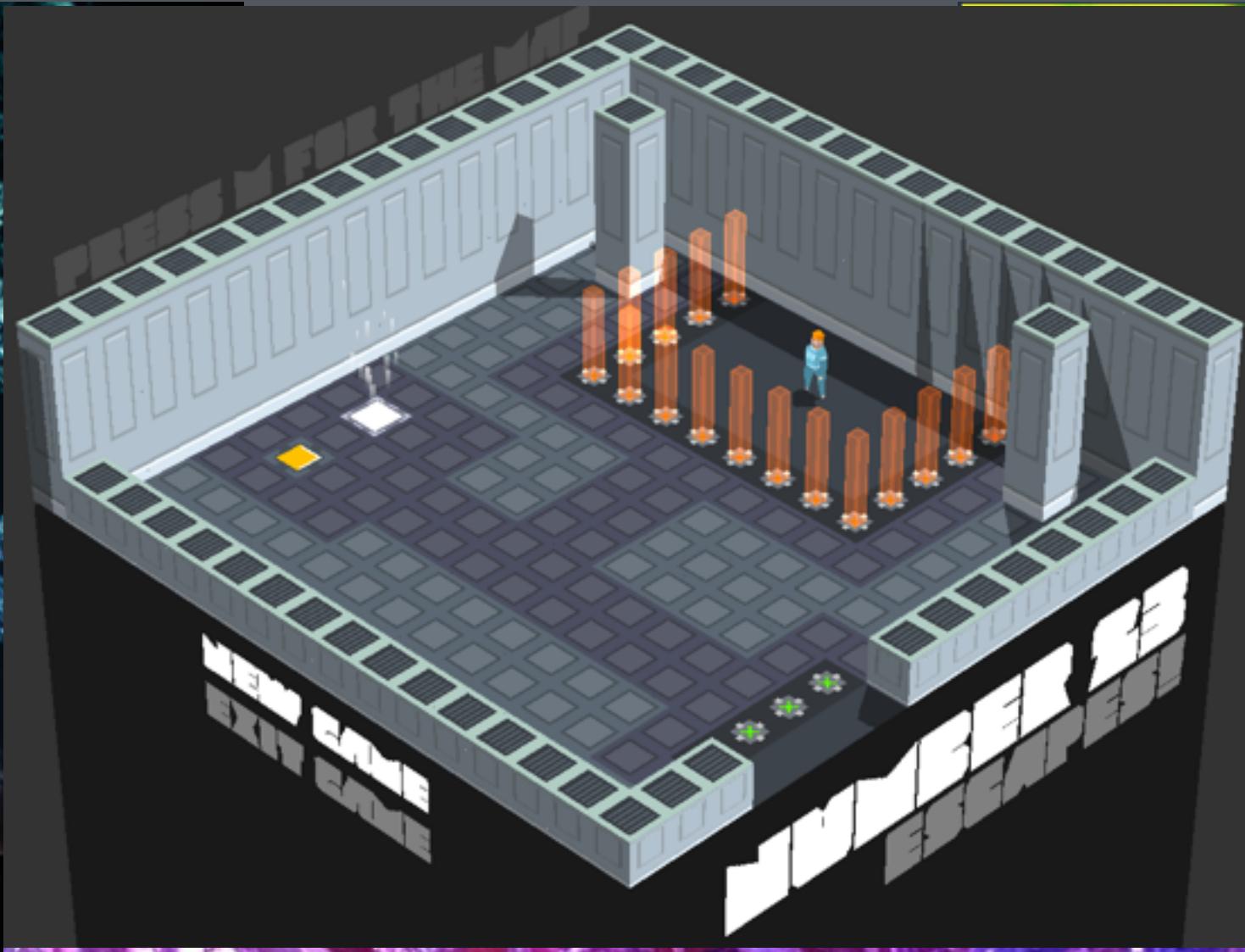
CS-3113

Introduction to game programming

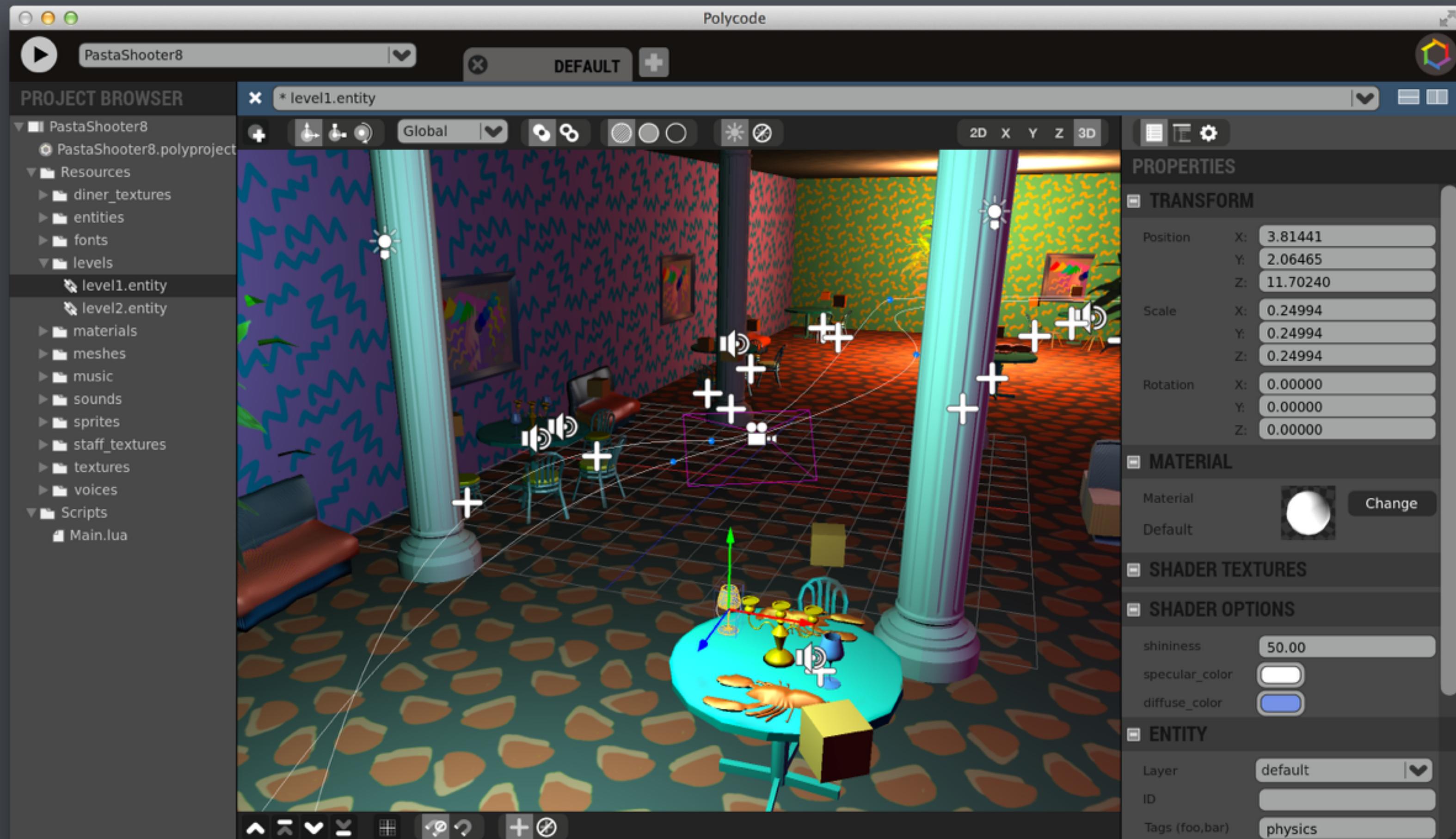


Hello!

Prof. Ivan Safrin



Polycode



polycode.org

hush

heyhush.com



The class...

Intro to Game Programming • CS-3113 • Spring 2016

Tuesdays and Thursdays, 4:30 p.m. – 5:50 p.m. • MetroTech 2, 8th floor, Room 812

Professor Ivan Safrin

Email: is1296@nyu.edu

Office hours: Tuesdays and Thursdays 6 p.m. - 7 p.m.

Prerequisites

CS-2134 Data Structures and Algorithms.

Course Description

“Intro to Game Programming” is a comprehensive course designed to provide an overview of modern game programming practices and their hands-on implementations. We will study the basic building blocks of interactive computer games and learn how to implement them as cross-platform C++ applications using the SDL and OpenGL libraries. Throughout the course, we will be building simple game prototypes around the topics learned in class, building up to a final game project to be completed by the end of the semester.

Course Topics

- Cross-platform game programming using C++ and SDL.
- Rendering 2D and 3D graphics using OpenGL ES2.
- Game physics and collision detection.
- Matrix algebra, linear transformations.
- Animation and special effects.

Course Overview

1. Introductions, overview of tools, setting up a development environment.
2. Basic graphics using OpenGL .
3. Input, time-based movement, basic collision detection.
4. Sprites, text and sprite animation.
5. Game physics, fixed time step.
6. Level editing and procedural generation.
7. Playing sounds.
8. Matrix transformations, advanced collision detection.
9. Particle systems, complex animation.
10. AI programming.
11. Introduction to 3D graphics.
12. Advanced 3D graphics.
13. Shaders, lighting and post-processing.
14. Using an external physics engine.
15. Cross-platform and mobile development.
16. Final project demos.

Reading materials

None required. All needed information will be provided in class and online, however the books below are recommended supplementary reading.

Class slides and other helpful materials are available on Github at
<https://github.com/ivansafrin/CS3113>

Very comprehensive book covering in great depth many of the concepts of graphics programming that we will be studying:

Foundations of 3D Computer Graphics (Steven J. Gortler)
<http://www.amazon.com/Foundations-Computer-Graphics-Steven-Gortler/dp/0262017350/>

And if you need to brush up on your C++:

Programming: Principles and Practice Using C++ (2nd Edition) (Bjarne Stroustrup)
<http://www.amazon.com/dp/0321992784/>

Grading / Assignments

Each week, you will have an assignment to complete by the beginning of next week's first class. Most of these assignments will involve implementing the concepts we learned that week as a simple game prototype. As part of the class, you will be expected to create a Github account if you don't already have one and submit your assignments via a git repository (we will go over how to do this during the first class). **Assignments must be submitted on time (before next week's first class, unless otherwise specified) or your assignment grade will be docked half a grade for each day it is late.**

You will also be expected to create a larger game by the end of the semester, which will serve as your final project. You may team up with another person for this project.

Your final course grade will be based on completion of weekly assignments (40%), tests (20%) and your final game project (40%).

Attendance Policy

Attendance will be taken every class. **If you miss more than three classes, your final grade will be affected.** Please keep in mind that we have a lot of material to cover in a fairly short time and missing even a single class will likely set you back!

Use of external code

For all of the assignments, including the final project, you will be expected to write all of the code yourself. Some example and helper code will be provided for you in class and online. If you wish to use other code, such as an open-source library or snippets of open-source code to implement features **NOT** covered in the class, you may do so, **but you must check in with me beforehand** and all open-source code used in your projects must be clearly marked and properly attributed.

<https://github.com/ivansafrin/CS3113>

A word of warning!

Prerequisites

CS2134 Data Structures and Algorithms.

Surprise C++ quiz!

What are games?

What are video games?

What is game programming?

What are the common
game components

?

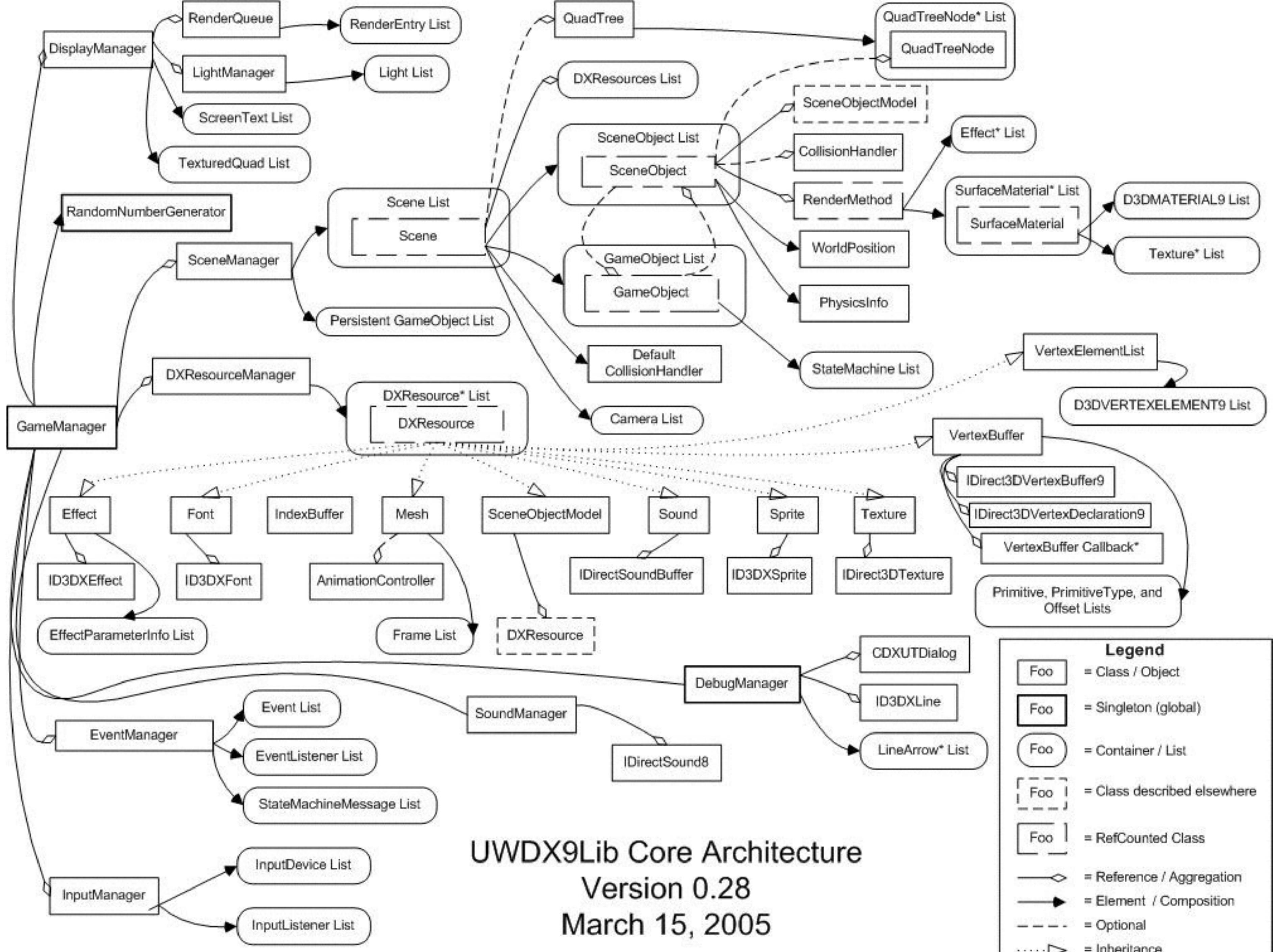
What are the common game components

?

- ▶ Graphics
- ▶ Input
- ▶ Sound
- ▶ Game Logic (Physics, AI)

Typical game structure.

```
Setup();  
while(gameIsRunning) {  
    ProcessInput();  
    UpdateGameWorld();  
    Render();  
}
```



UWDX9Lib Core Architecture
Version 0.28
March 15, 2005

Our tools

C++

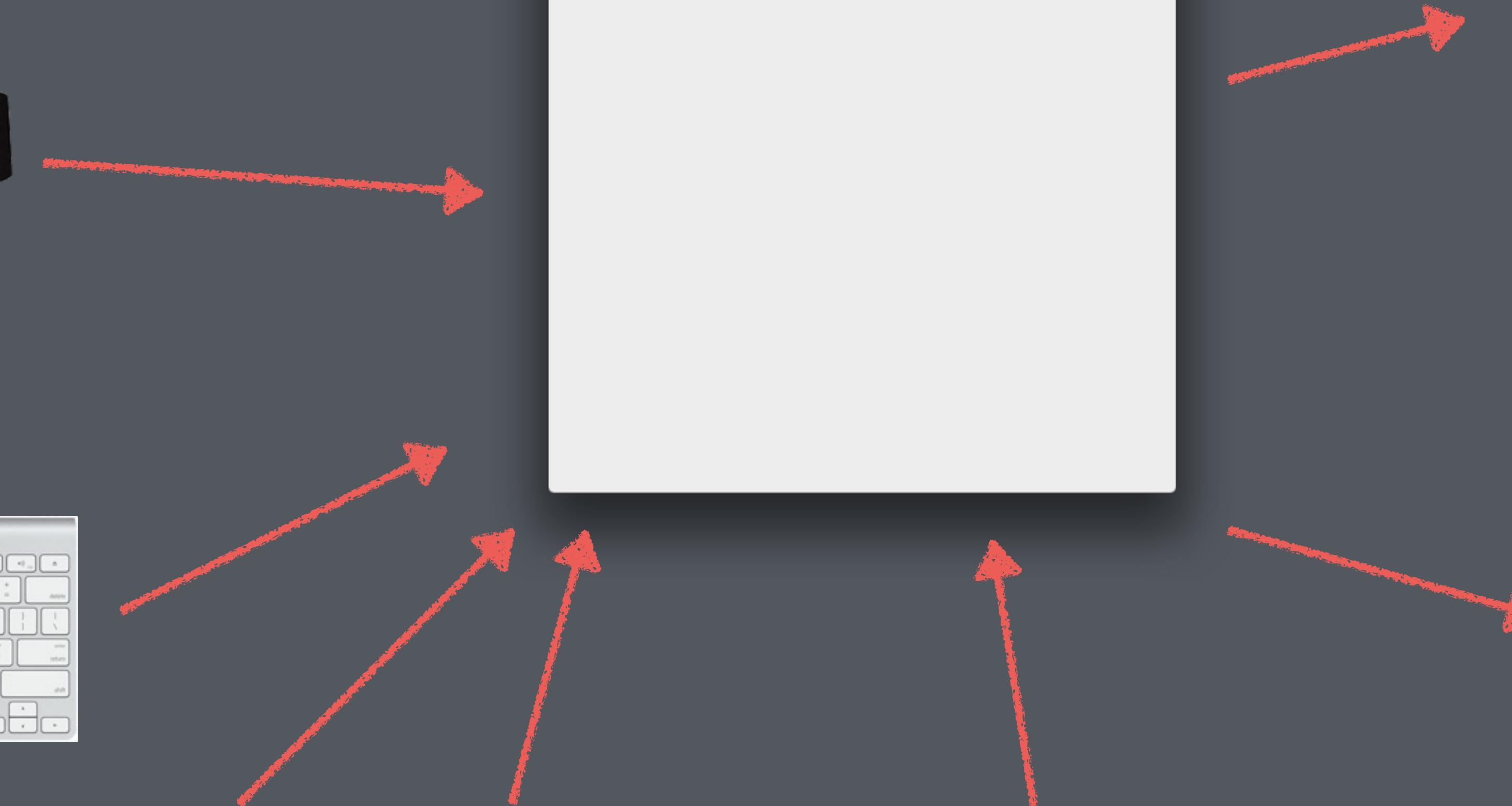
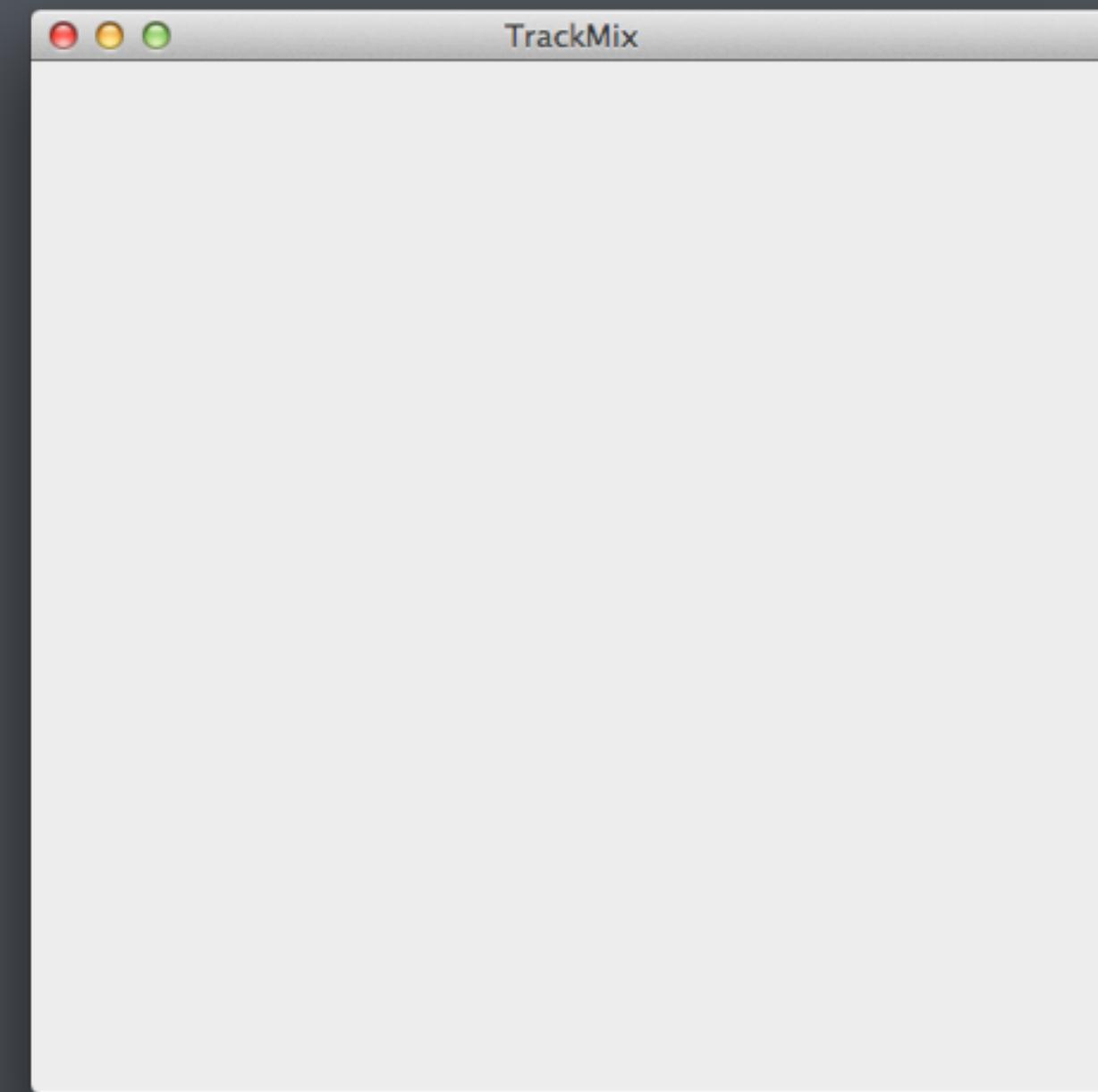
SDL (Input, Windowing, Sound)

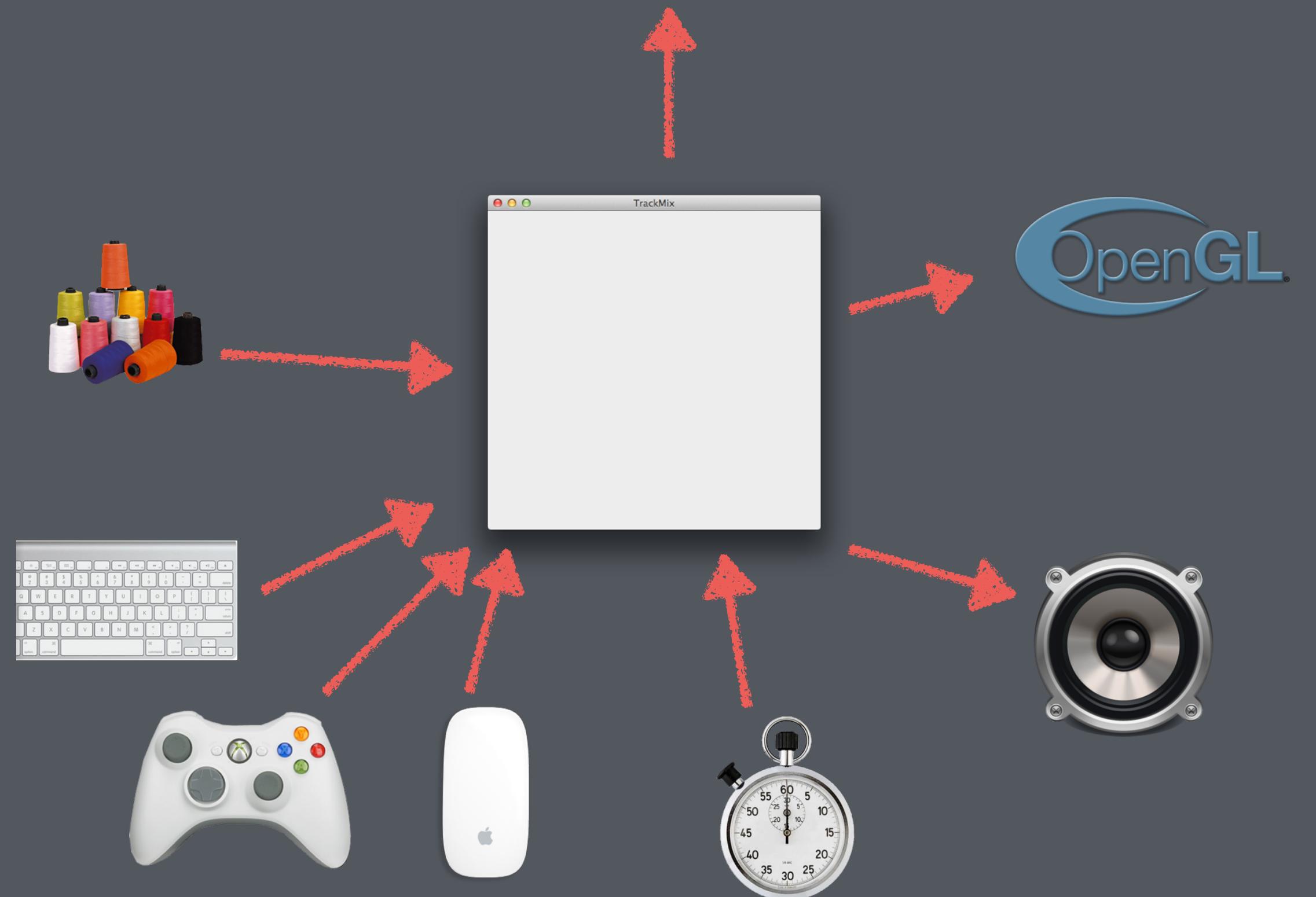
OpenGL (Graphics)

Why C++, SDL and OpenGL?

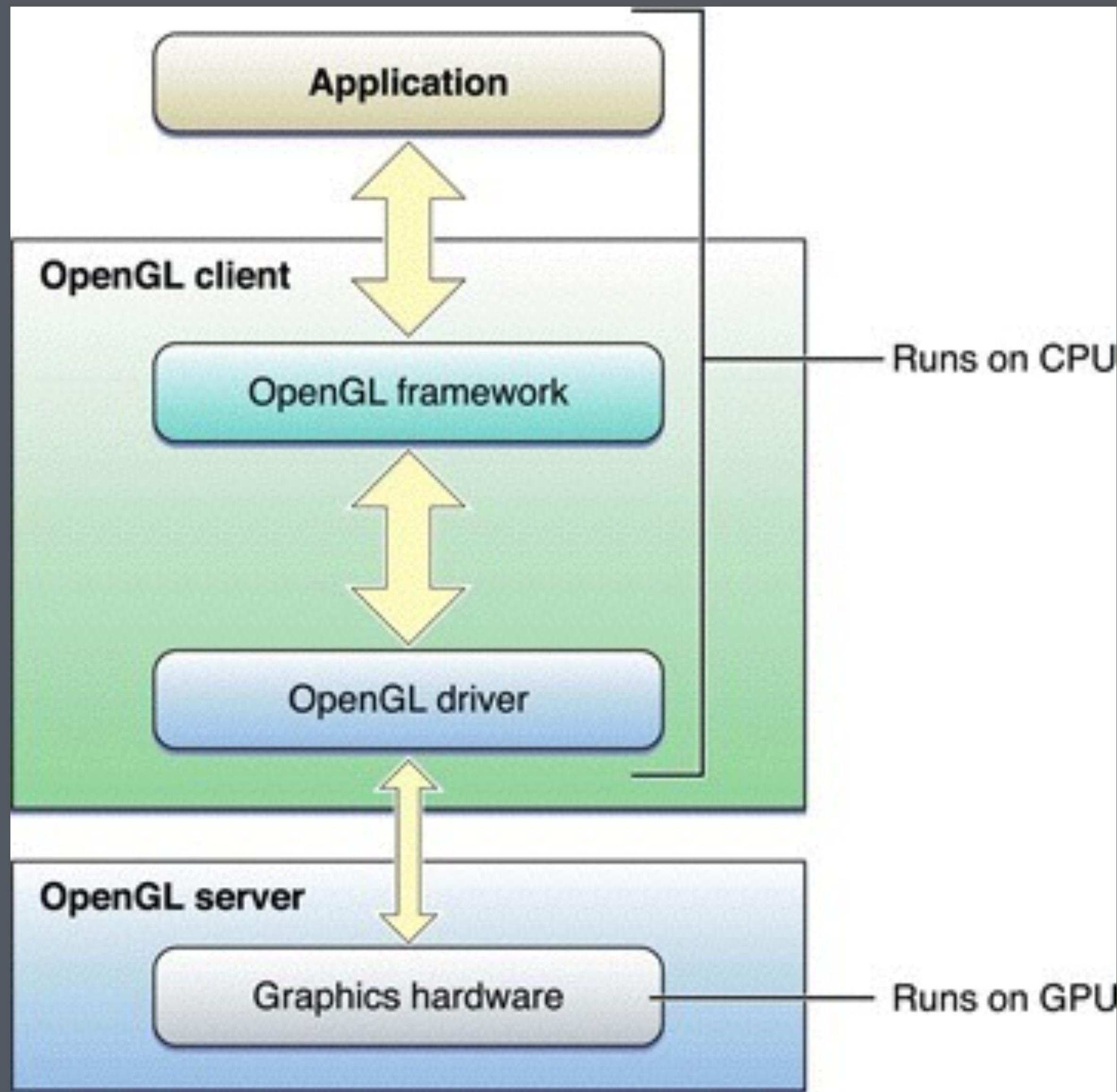
- Widely used for game development
- Fast and free
- Allows us to focus on general programming concepts
- Builds a low level understanding

What is SDL?





What is OpenGL?





Setup your development environment.



Setup a Github account.

github.com

Install git if you don't already have it



git-scm.com

If you don't know how to use git,
follow this tutorial.

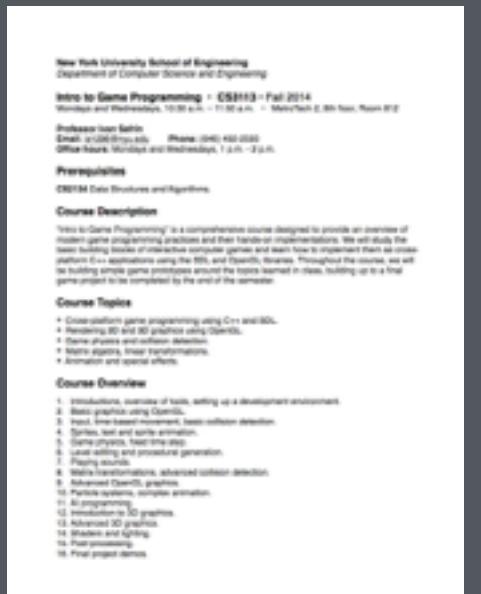
try.github.io

Clone the class repository at
github.com/ivansafrin/CS3113

```
git clone https://github.com/ivansafrin/CS3113.git
```

What's in the repo

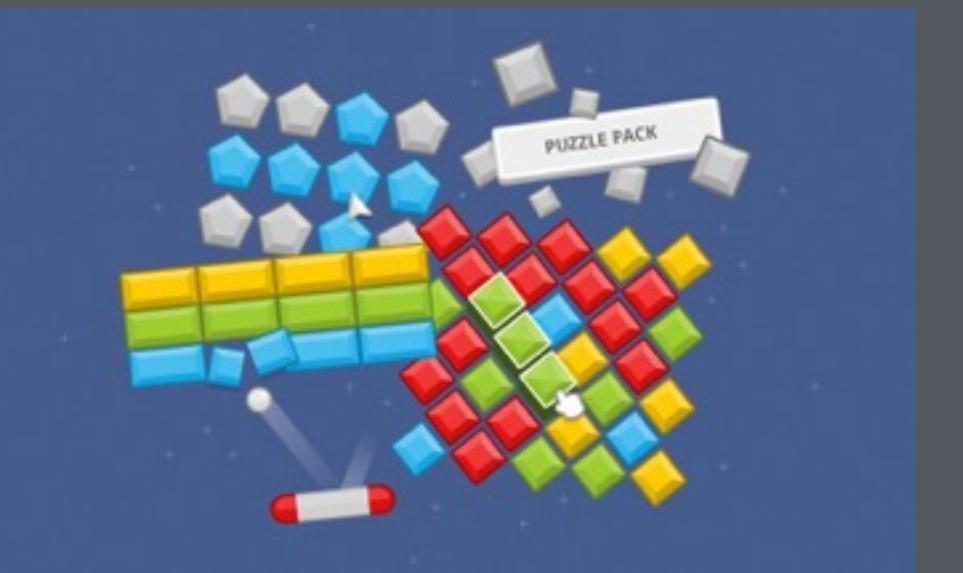
?



Syllabus



Project templates
and helper code



CC0 Game Assets



Class slides

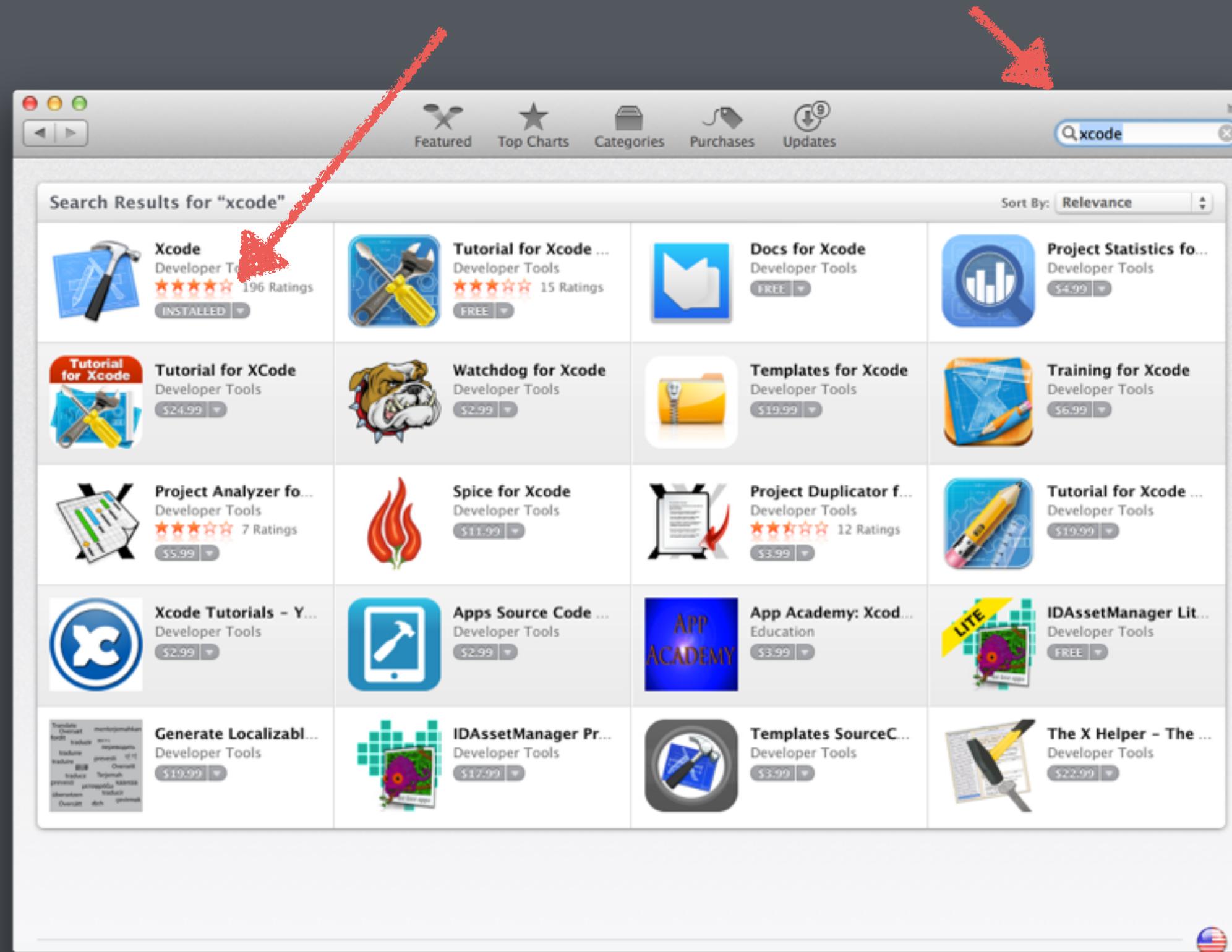
Create a Github repository for this class

<https://help.github.com/articles/create-a-repo>

Install **Xcode** if you are on
Mac or **Visual Studio** if you
are on **Windows**.

To install Xcode

Search for **Xcode** in the AppStore



Click Install

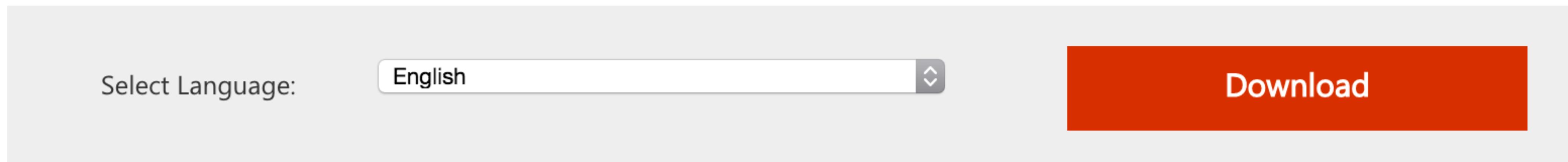


To install Visual Studio Express

Go to

<http://www.microsoft.com/en-us/download/details.aspx?id=48131>

Microsoft Visual Studio Express 2013 for Windows Desktop with Update 5



Do not install VS 2015!



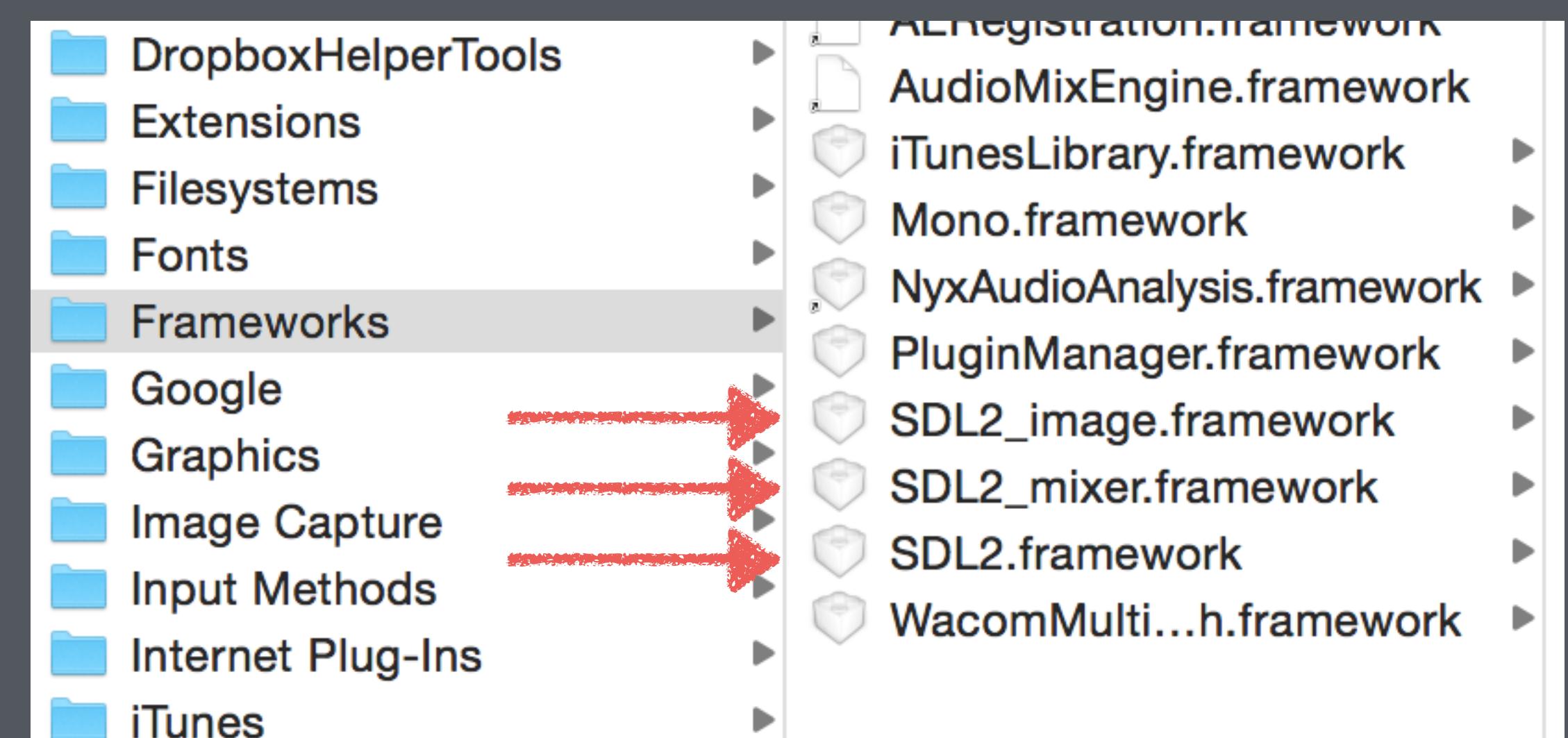
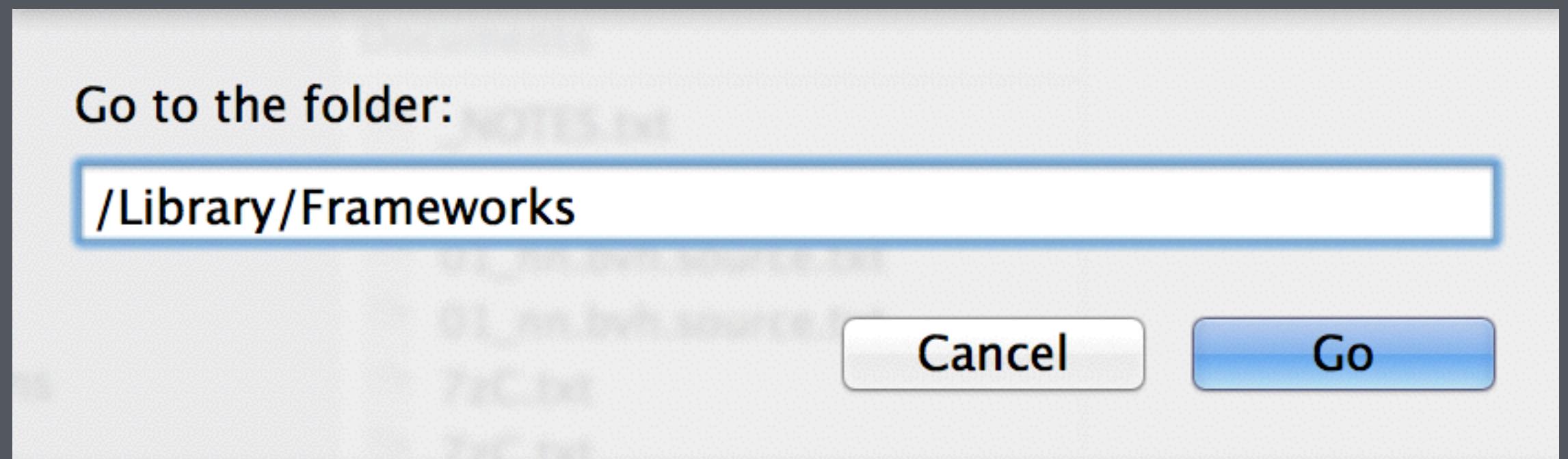
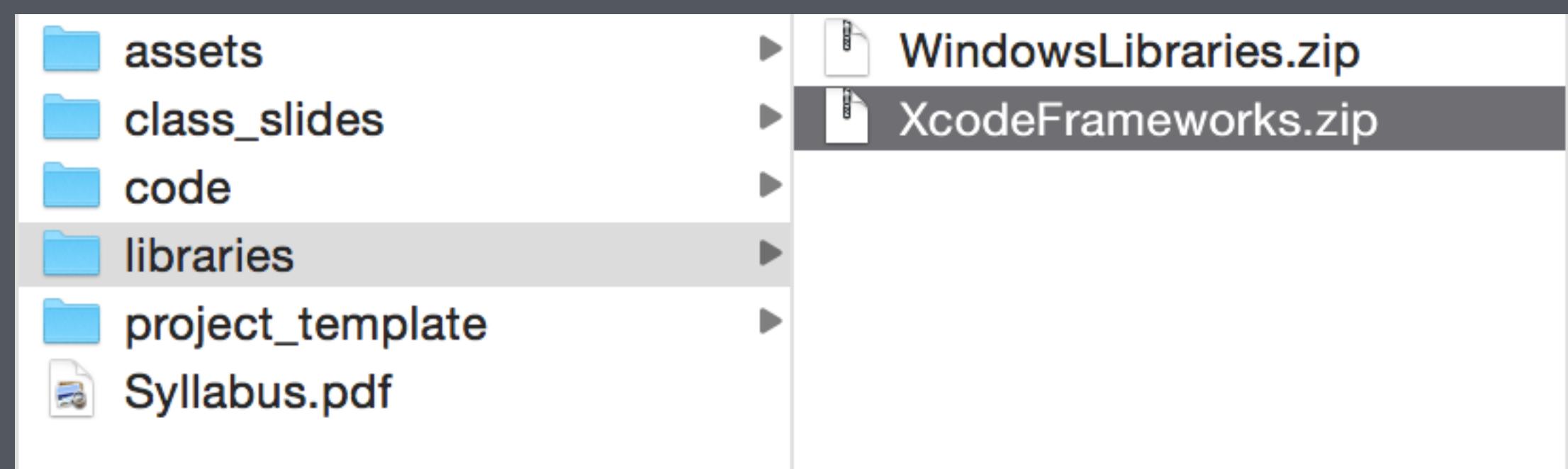
~~Visual Studio 2015~~

~~Tools for every developer and
every app.~~

Install **SDL** libraries

On Mac

Unzip XcodeFrameworks.zip under libraries in the class repository to /Library/Frameworks

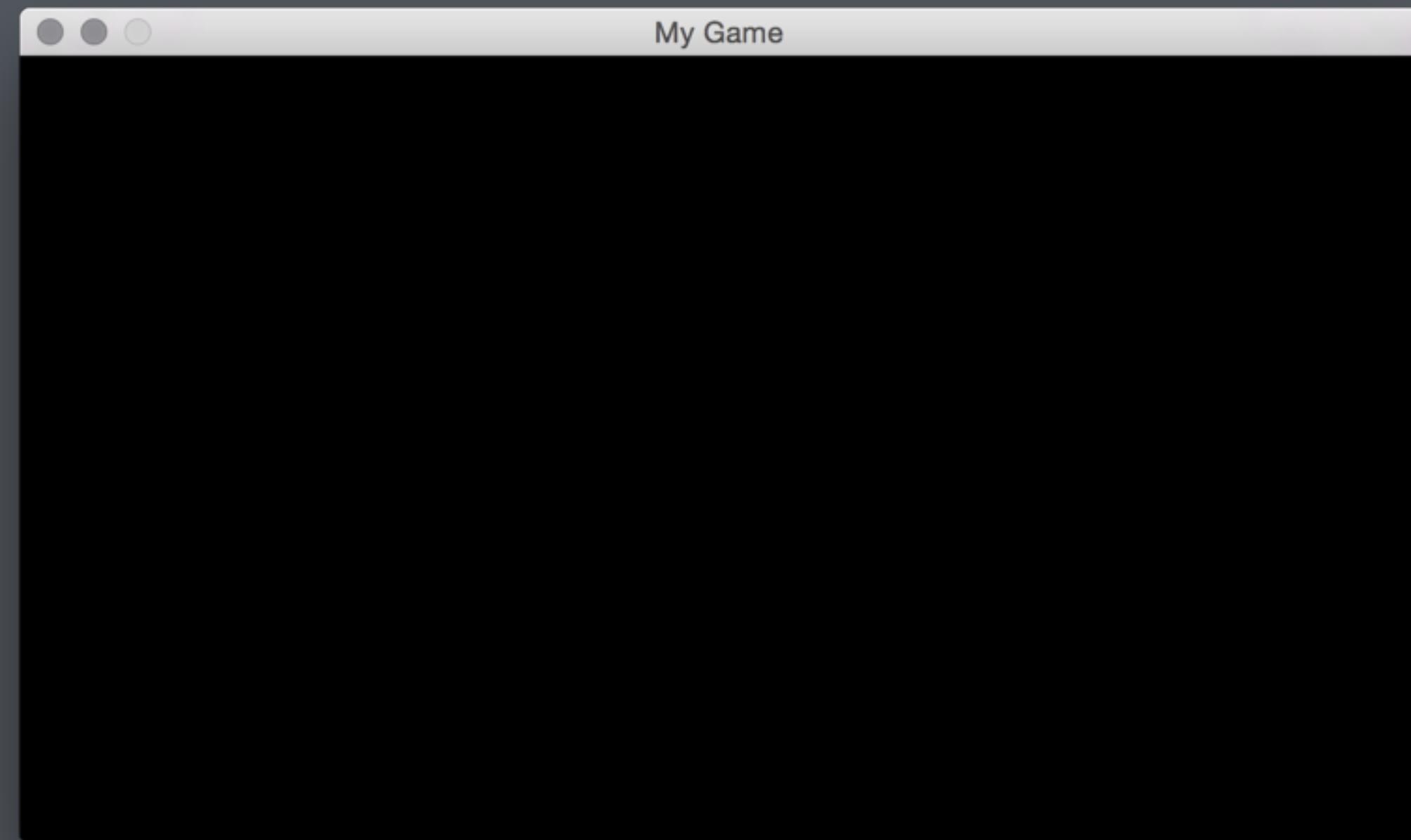


On Windows

Unzip **WindowsLibraries.zip** under
libraries in the **class repository**
to **C:**



Build and run the template project
from the class git repository.



SDL boilerplate code.

```
#ifdef _WINDOWS
    #include <GL/glew.h>
#endif
#include <SDL.h>
#include <SDL_opengl.h>
#include <SDL_image.h>

#ifdef _WINDOWS
    #define RESOURCE_FOLDER ""
#else
    #define RESOURCE_FOLDER "NYUCodebase.app/Contents/Resources/"
#endif

SDL_Window* displayWindow;

int main(int argc, char *argv[])
{
    SDL_Init(SDL_INIT_VIDEO);
    displayWindow = SDL_CreateWindow("My Game", SDL_WINDOWPOS_CENTERED, SDL_WINDOWPOS_CENTERED, 640, 360, SDL_WINDOW_OPENGL);
    SDL_GLContext context = SDL_GL_CreateContext(displayWindow);
    SDL_GL_MakeCurrent(displayWindow, context);
#ifdef _WINDOWS
    glewInit();
#endif

    SDL_Event event;
    bool done = false;
    while (!done) {
        while (SDL_PollEvent(&event)) {
            if (event.type == SDL_QUIT || event.type == SDL_WINDOWEVENT_CLOSE) {
                done = true;
            }
        }
        glClear(GL_COLOR_BUFFER_BIT);
        SDL_GL_SwapWindow(displayWindow);
    }

    SDL_Quit();
    return 0;
}
```

Typical game structure.

```
Setup();  
while(gameIsRunning) {  
    ProcessInput();  
    UpdateGameWorld();  
    Render();  
}
```

Home assignments

- Set up a **github account** if you don't have one.
- Install **git** if not installed and **learn how to use it**.
- Clone the **class repository**.
- Setup **your own repository** for this class.
- Setup **Xcode** or **Visual Studio** and install **SDL**.
- Make sure the **template project** in the class repository **builds and runs!**

Please email me if you run into any issues:
is1296@nyu.edu