SDL-Setup-Tutorial for Mac OS with Xcode

Based on the previous SDL-Setup-tutorial by Andreas Kramer, I managed to build the project files on the latest Mac OSX (Version 10.11.1) with Xcode 7.0.1. I would like to share my experience, update the changes and explain some confusing settings in details for the other Mac users in DH2323.

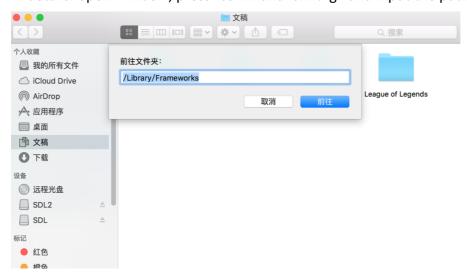
Step1: Download SDL 1.2 (http://www.libsdl.org/download-1.2.php).

I choose this version just because its installation package has the "SDLMain.m" file inside of the "devel-lite" folder while SDL 2.0 have no such one.

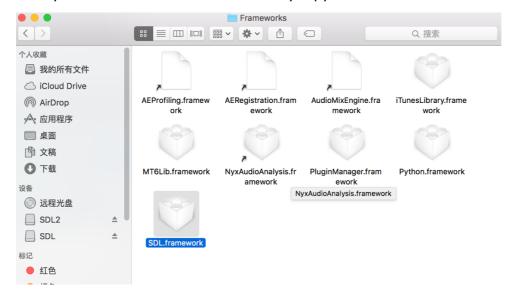
Step2: Install "SDL.framework"

Copy it to the path "/Library/Frameworks".

In details: open "Finder", press "command+shift+g" and input the path.



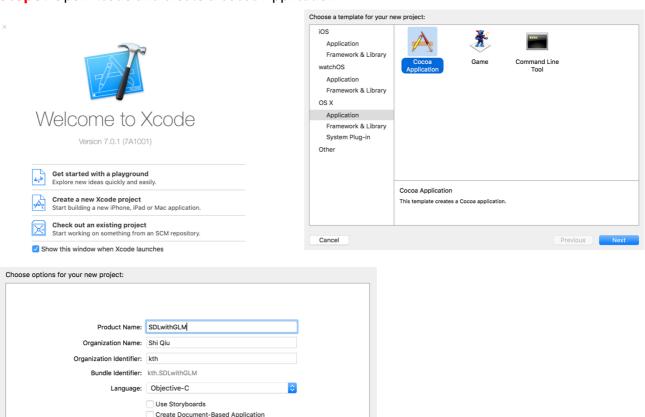
Then you come to the folder and can easily copy the "SDL.framework" to it.



Step3: Open Xcode and create a Cocoa Application.

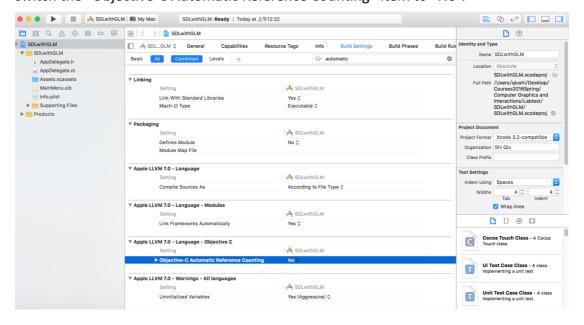
Document Extension: myde

Use Core Data
Include Unit Tests
Include UI Tests

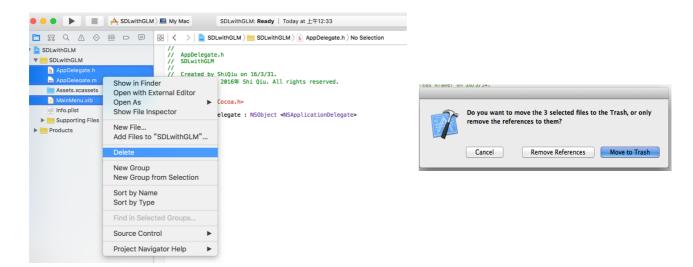


Step4: Select the blue icon of your project and go to the "Building Settings". Then switch the "Objective-C Automatic Reference Counting" item to "No".

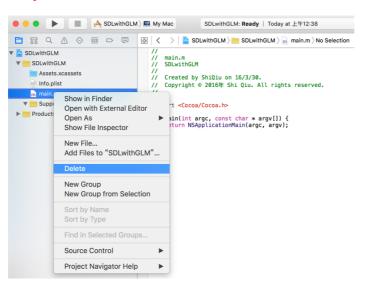
Previous Next



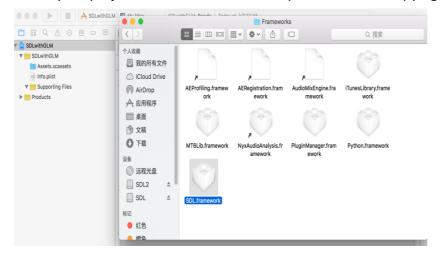
Step5: Delete AppDelegate and MainMenu, and move them to trash.



Step6: Delete "main.m", and move it to trash.

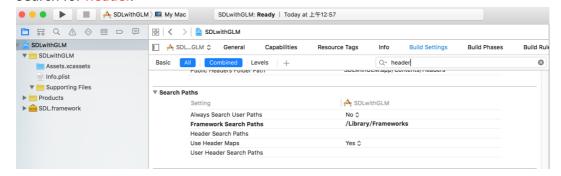


Step7: Open the "/Library/Frameworks" folder and drag the file "SDL.framework" under your project "SDLwithGLM". And import WITHOUT copying and add to target.

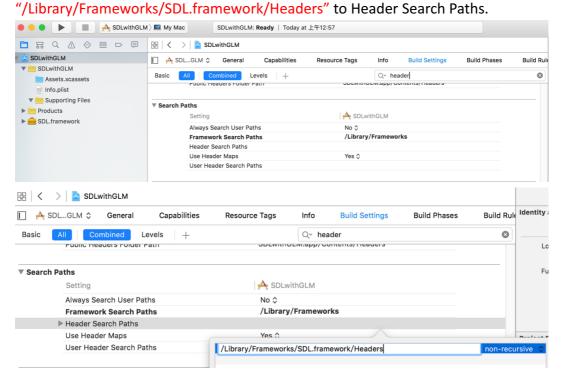




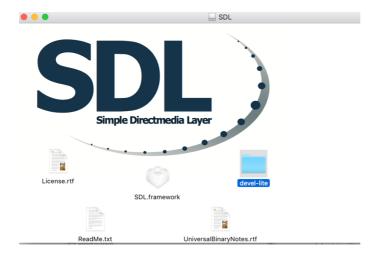
Step8: Select the blue icon of your project and go to the "Building Settings". And Search for header.



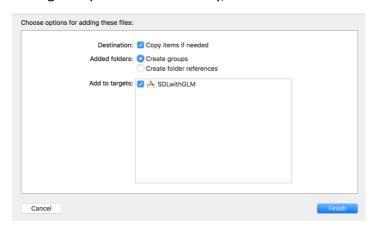
Step9: Add "/Library/Frameworks" to Framework Search Paths and Add

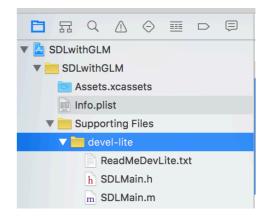


Step10: Find the "devel-lite" folder in the installation package of SDL 1.2

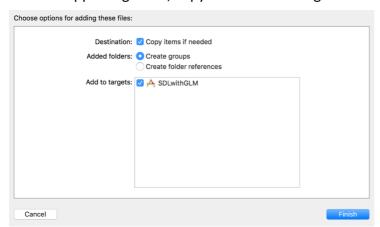


Step11: Drag it to your project under the folder "Supporting Files", copy it and add to target. If you done it correctly, the icon of it should be yellow.



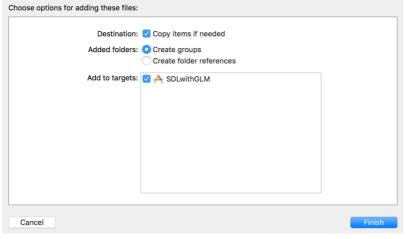


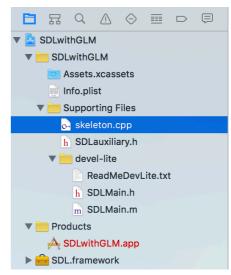
Step12: Find "SDLauxiliary.h" in "CgLab1", drag it to your project under the folder "Supporting Files", copy it and add to target.



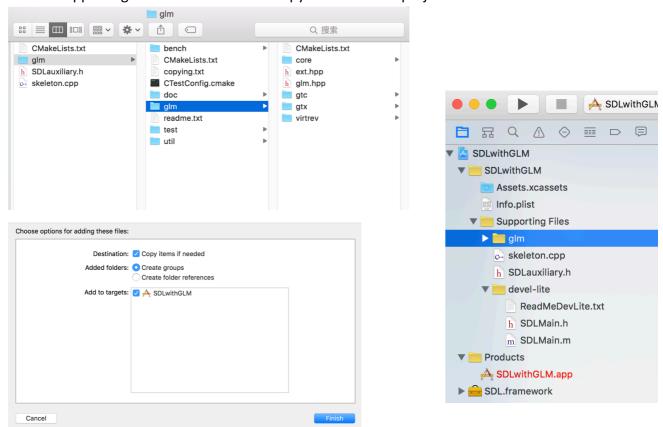


Step13: Similarly find "skeleton.cpp" in "CgLab1", drag it to your project under the folder "Supporting Files", copy it and add to target.

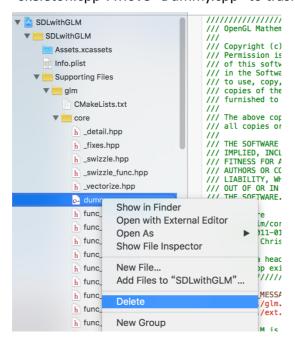




Step 14: Time to import GLM - you want the folder within the GLM folder with the same name. When you found that folder - drag it into Xcode under the folder "Supporting Files". Make Xcode to copy it into its own project folder.



Step 15: Delete the "Dummy.cpp" file within the "core" folder inside GLM - this file contains a main function which will make Xcode execute that file instead of "skeleton.cpp". Move "Dummy.cpp" to trash.



Step 16: As for "skeleton.cpp", after importing and setting up build settings - Xcode should recognize GLM and suggest to include it with quotes instead of angelbrackets.



Step 17: Do the same procedures to "SDLauxiliary.h":

```
#include 'glm/glm.hpp' file not found with angled> include; use 'quotes' instead led. After calling this function do the actual drawing.

#include 'splm/glm.hpp' with "glm/glm.hpp" do the actual drawing.

#include 'glm/glm.hpp'

// Initializes SDL (video and timer). SDL creates a window where you can draw.

// A pointer to this SDL_Surface is returned. After calling this function

// you can use the function PutPixelSDL to do the actual drawing.

SDL_Surface* InitializeSDL( int width, int height, bool fullscreen = false );
```

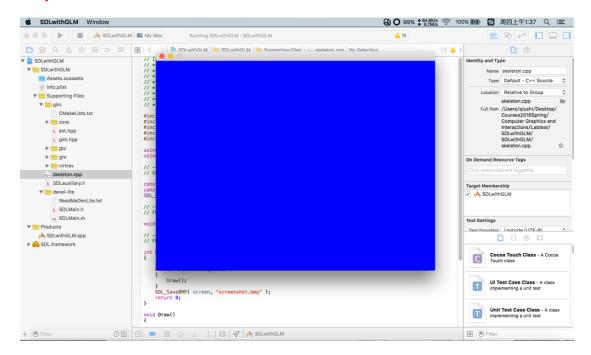
Step18: Build and run.

```
● ● ● ■ | → SDLwithGLM | ■ My Mac | SDLwithGLM: Ready | Today at 上午1:36
                                                                                                                                                                                      <u>^</u> 5

    SDLwithGLM > SDLwithGLM > SUpporting Files > Skeleton.cpp > No Selection

                                                                                                                                                                                                            < 🛕 >
                                                          // Introduction lab that covers:
 SDLwithGLM
                                                         // Introduction case that covers
// * C0+
// * SDL
// * 2D graphics
// * Plotting pixels
// * Video memory
// * Color representation
// * Linear interpolation
// * glm::vec3 and std::vector
 ▼ <u>SDLwithGLM</u>
       Assets.xcassets
        Info.plist
    ▼ Supporting Files
       ▼ iii glm
                                                         #include "SDL.h"
#include <iostream>
#include 'glm/glm.hpp"
#include <vector>
#include "SDLauxiliary.h"
              CMakeLists.txt
          ▶ iii core
             h glm.hpp
          ▶ 📒 gtc
                                                          using namespace std;
using glm::vec3;
          ▶ 📒 gtx
           ▶ iii virtre
                                                          // GLOBAL VARIABLES
                                                          const int SCREEN_WIDTH = 640;
const int SCREEN_HEIGHT = 480;
SDL_Surface* screen;
          h SDLauxiliarv.h
        ▼ | devel-lite
              ReadMeDevLite.txt
             h SDLMain.h
                                                          // -----
// FUNCTION DECLARATIONS
                                                          void Draw();
      A SDLwithGLM.app
```

Step 19: Success.



Happy coding:)

Original: Andreas Kramer

Updated: Shi Qiu (shiq@kth.se)