CSE 167 HW 0 - Due Thur. Jan 18th at 11:59 p.m.

This homework will help you set up OpenGL on your computer.

First, let's make sure we have all of the starter code downloaded.

https://github.com/HT413/CSE167StarterCode

MAC (Go to the second part of the tutorial if you are using windows)

Once you have unzipped the project, open XCode and create a new project.

1. Select Command Line Tool as the template for the new project
2. Name your project any name of your choosing

3. Upon completion, you should be presented with something like this:

4. Let's open the directory enclosing our newly created XCode project alongside the unzipped starter code directory. Drag and drop the files as shown below

5. If presented with the option, replace the already existing files (main.cpp in this case)

   All is good for now. We will come back to this later.

Next, make sure you have a recent version of CMake. You can download CMake from here: https://cmake.org/

You may not already have the /Developer folder. If so, you'll have to make the directory. To do this, open a terminal and type:
ls /Developer

- If you get an error message that /Developer does not exist we have to create the developer folder.

sudo mkdir /Developer

- You may be prompted to enter your password.

From now on, when you're running commands inside /Developer, you'll need to prepend them with sudo.

Install GLFW


1. Unzip the downloaded folder
2. Drag and drop the glfw folder in its entirety into your /Developer folder. Authenticate accordingly.
3. Open a terminal. Replace 3.1.2 with the version you downloaded.

- cd /Developer/glfw-3.1.2
- sudo cmake -DBUILD_SHARED_LIBS=ON .  
- sudo make

Install GLM
Begin by downloading GLM: http://glm.g-truc.net

1. Unzip the downloaded folder
2. Drag and drop the glm folder in its entirety into your /Developer folder. Authenticate accordingly.
3. Open a terminal

- cd /Developer/glm
- sudo cmake . #(Don't miss the dot!)
- sudo make

XCode Setup
The last step is to set up XCode to find where we put all of our code, libraries and headers

1. Open up the XCode project
2. In the far left panel of the Xcode window, you will see a folder matching the project name.
   - Right click this folder and click "Add files to <project name>".
   - Choose the files as shown below and press “Add"

3. Above this folder, you will see the top level which shares the same name as your project.
- Click this and choose the "Build Settings" option at the top.
- Scroll down to "Search Paths".
- Add the following paths as shown below so XCode knows where to find all of the glfw and glm headers in our code. Replace 3.1.2 with the glfw version you downloaded. Select "non-recursive" from the dropdown after adding the path.

  /Developer/glfw-3.1.2/include
4. Do the same for the library paths, adding the following path. Again, replace 3.1.2 with the version you downloaded.
  - /Developer/glfw-3.1.2/src

5. At the top of this window, click "Build Phases"
  - Open up the "Link Binary with Libraries" pull down
  - Press the little "+" sign and type "OpenGL"
  - Choose "OpenGL.framework" and press "Add"
  - Press the "+" sign again and choose "Add Other..."
- Navigate to /Developer/glfw-3.1.2/src (replace 3.1.2 with the version you downloaded), choose "libglfw.dylib" and press "Open"
6. Optional: We like to see a good frame rate, so we prefer you run in Release (at least during the grading demo).

- Open Product > Scheme > Edit Scheme, and choose “Release”
7. Run that baby (press the play button at the top left or press Command + r).

PLEASE CHANGE THE COLOR OF THE CUBE (ANY COLOR OTHER THAN THE DEFAULT BLUE.)

8. Optional: The compiler might give you a lot warnings regarding documentation. You can turn them off in Build Settings.
Windows

The very first thing you'll be doing is downloading and installing the Microsoft Visual Studio 2015/2017 Community Edition for free.

Once that is done and you have unzipped your starter code, we're ready to begin configuring our Microsoft Visual Studio solution to work with OpenGL.

1. Open the solution GLFWStarterProject.sln with Microsoft Visual Studio.

2. Open the **NuGet Package Manager** by selecting Tools->NuGet Package Manager->Manage NuGet Packages for Solution...

3. Make sure you're in the Browse tab. Search for nupengl and install the **nupengl.core** package (NOT nupengl.core.redist). Alternatively, if you feel that this is too overkill (since nupengl contains 5 or 6 different commonly used libraries in OpenGL applications), download the glew and glfw packages (NOT the .redist packages).
   - If the install button is greyed out, make sure the GLFWStarterProject checkbox is on.

4. Similarly, install the **glm** package.
5. Now that we have all our packages, we just need to Link against the OpenGL libraries.

6. We can do that by going to our project's (note: NOT solution) Properties. Right-click on GLFWStarterProject project in the Solution Explorer.

7. Once in project properties, go to Configuration dropdown to change it to All Configurations. This will ensure we edit both the debug/release configuration at the same time, and don’t have to repeat the step.
8. Navigate to Linker->Input to prepend `opengl32.lib;glu32.lib` to Additional Dependencies.

9. Change the active configuration to **release** for faster execution! If you want to do breakpoint debugging, remember that you'll have to change this back to **debug** to make sure you can see the variables.

10. Once this is done, build and run the program(CTRL+F5) to see a spinning cube!

    **PLEASE CHANGE THE COLOR OF THE CUBE (ANY COLOR OTHER THAN THE DEFAULT BLUE.)**
SUBMISSION

Turn in this PDF (not the code) on GradeScope by Jan 18th 11:59 p.m.

Come to the grading session and do the demo (rotating cube) on Friday Jan 19th.