CMCS427 Microlab: Installing JOGL and OpenGL Try by Oct. 3rd

Before we start programming with OpenGL you need to check which version of OpenGL you have, and get a basic OpenGL program running.

In this class we'll officially support OpenGL through Eclipse and the Java-based JOGL implementation. We'll give instructions for this implementation. If you wish to use a different IDE, or can manage OpenGL in C++, you're free to do so as long as you finish required labs and projects.

Step 1: Testing your OpenGL installation

There's an app for it. The company realTech VR maintains an app that will query your OpenGL installation, both software and hardware, and run demos to test framerates.

http://realtech-vr.com/admin/glview

OpenGL gone through releases 1, 2, 3 and 4, with versions up to 4.6. Some computers may only run 4.3, or 4.1, or even 3.3. Macs in particular are behind, and some older Macs only support up to version 3. In coding OpenGL you may need to tweak code for backward compatibility, so it's good to know your version.

Mac OS:
1) Under Apple Menu, About This Mac, you can get a report on your system that includes the "Graphics/Display". This will give basic hardware information.

2) Get the free **OpenGL Extensions Viewer** app from the App Store, and run it to find the OpenGL software and hardware characteristics of your system. Note in particular the highest version of OpenGL.

Windows:
1) Under the Windows, you can query the graphics card functionality with the DirectX utility **dxdiag**. Either search for it under the start menu, or get a command line up and run it. But, DirectX is Microsoft's graphics library, not OpenGL, so you get primarily hardware info.

2) You can also get the **OpenGL Extensions Viewer** app. See the link above for details.

Linux:
For Linux the mesa-utils package includes a command called **glxinfo**, and if run will output several things about the OpenGL implementation on the computer, including its version. To focus filter its output you can run it as "glxinfo | grep OpenGL". But, first you have to install Mesa, which is an open source implementation of OpenGL.

Step 2: Getting JOGL

OpenGL is a not a language, it is a set of low level library calls (or API) that access functions on your graphics card (or, without a card, a software implementation of the functions.)

OpenGL must be bound to a language for use, which can be C++, Java, Python, or others. JOGL is a binding of OpenGL to Java, and we'll use it with the Java widget set Swing.

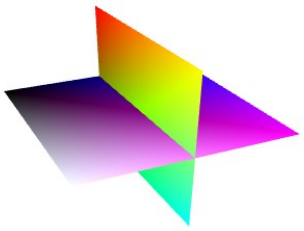
Before getting JOGL, here's a way to use advanced features of OpenGL and JOGL in Processing. If this program runs you know JOGL runs on your system. Processing includes the JOGL jar files.

Step 2a: Advanced OpenGL in Processing

Download the file **OpenGLAdvanced.zip** from the course web site. When decompressed you should see a folder with three files:

OpenGLAdvanced.pde
vert.glsl
frag.glsl

The latter two are shader programs, which you don't need to understand now. The program displays two interlaced rectangles rotating in space.



The advantage of approaching OpenGL this way is that you can take advantage of many parts of Processing – the camera method, the event handlers, easy access to the mouse. But, we want to move to a full implementation of OpenGL in Java using Eclipse.

Step 2b: OpenGL in Java Swing and Eclipse

For this step we'll depend on instructions in external links.

First, get JOGL. Use the first link to get the stable version of JOGL jogamp-all-platforms:

https://jogamp.org/wiki/index.php/Downloading_and_installing_JOGL

This link will do this directly: jogamp-all-platforms.7z

Now uncompress the file (compress with 7zip and Mac users may need to get a utility for this.) You'll have a JOGL folder.

Second, create a project named JOGL in Eclipse to house JOGL. You'll use this to link to the JOGL files. Follow the instructions for Eclipse given here.

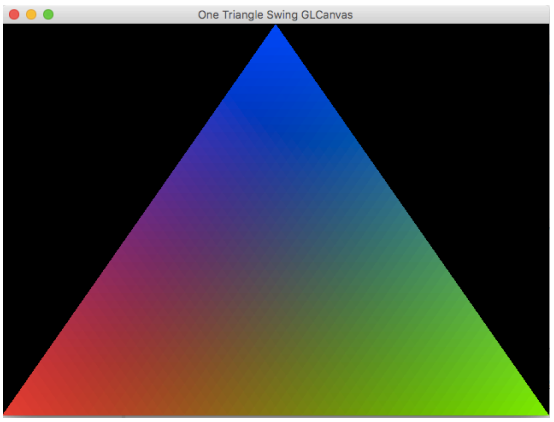
https://jogamp.org/wiki/index.php/Setting_up_a_JogAmp_project_in_your_favorite_IDE#Eclipse_IDE_project

Third, create a project to use JOGL. I've used the source code from this link:

https://jogamp.org/wiki/index.php/Using_JOGL_in_AWT_SWT_and_Swing

Create a Java project in Eclipse and add two source files, OneTriangle.java and OneTriangleSwingGLCanvas.java. These files are on that link. Ignore the AWT and SWT versions. Using the instructions in the link in the Second step above, link this project to your JOGL project. We'll never run the JOGL project, only projects that link to it.

If all works you should get this picture.



At this point you should have a running OpenGL program in Processing, and in Java.