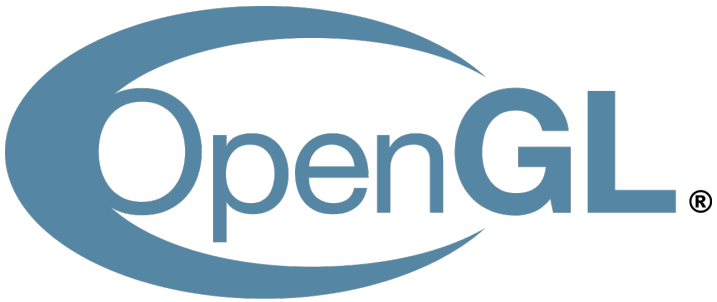
Intro to Opengl

Dane Christensen, Brigham H. Keys, Esq.

BYU-Idaho

key13005@byui.edu

August 8, 2015



What is OpenGL?

OpenGL is the industry's foundation for high performance graphics, from games to virtual reality, mobile phones, and supercomputers. It is a cross-language, cross-platform application programming interface for rendering 2D and 3D vector graphics. The API is typically used to interact with a graphics processing unit (GPU), to achieve hardware-accelerated rendering. It does not handle windows, keyboard, mouse or any other kind of input from the user. However there are other APIs that work next to OpenGL to accomplish these tasks.

History of OpenGl

OpenGL has been out there for a while, with over 200 function and many legacy and deprecated functions. At the time of this writing the most up to date version of OpenGL is 4.5.

- 1980's – Lacking an open standard, developers had to often write custom interfaces for each piece of hardware, this resulted in the duplication of effort and was largely inefficient.
- Early 1990's – Silicon Graphics was the leader of the 3D graphics for workstations. Their API called IRIS GL was the de facto industry standard. It's main competitor was the open standard named PHIGS. However IRIS GL was more popular because it was easier to use and it supported intermediate mode. However many companies such as IBM and Sun Microsystems were also pushing the PHIGS standard, offering extensions and weaking Silicon Graphics's market share. To combat this Silicon Graphics turned IRIS GL into an open standard called OpenGL.
- 1995 – Microsoft released Direct 3D, which to this day is the main competitor to OpenGL.
- 2006 – Control of the OpenGL API was transfered to the non-profit organization, the Khronos Group, who to this day control the API and many other open standards, some being WebGL and OpenGL ES.
- 2015 – The Kronos Group announces the Vulkan API, a legacy free graphics API for next generation graphics.

Libraries used in this course

OpenGL

Chances are you already have the OpenGL installed as it is often bundled with your graphics card driver. However you likely are going to need to install the development files.

The OpenGL Utility Library (GLU)

Is a collection of functions that do drawing on a higher level, and has functions that generally make our lives easier. It is bundled with OpenGL and uses OpenGL directly.

FreeGlut

Another collection of functions that have high level drawing. But it's main purpose for us is to handle the keyboard, mouse and window handling. This will likely need to be installed.

Setting up the compiler

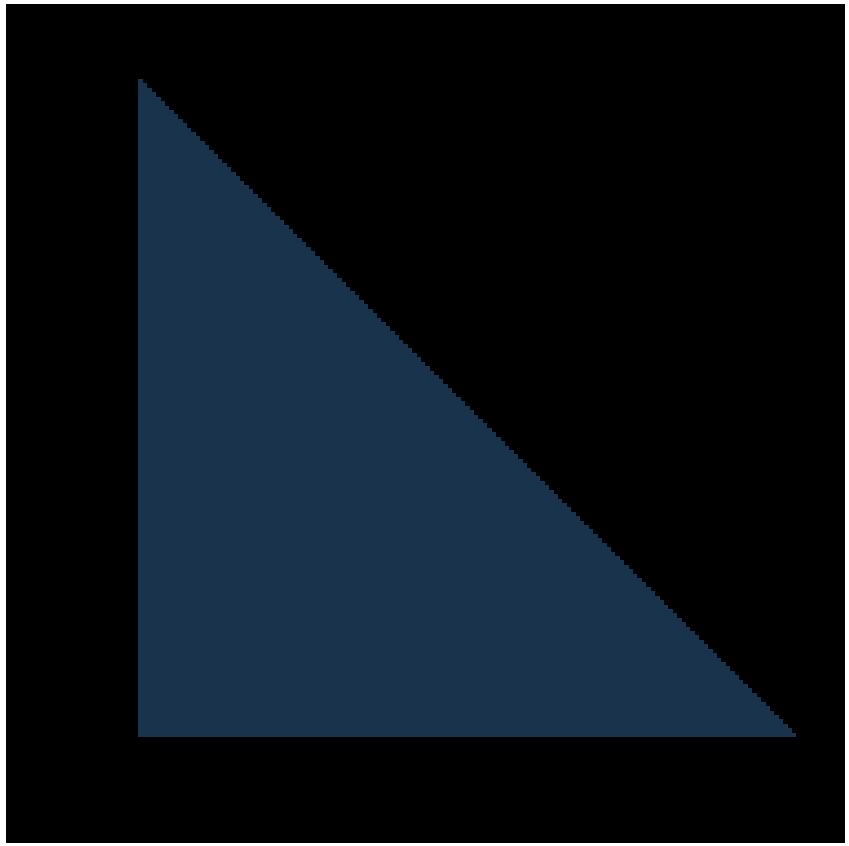
To compile code containing OpenGL functions, we need to let the compiler know that we are calling them in the first place by linking the libraries to our binary. Below are the individual flags we need to give to the compiler to use these utilities. Here is the syntax to compile the code from the command line:

Compiler	**File**	**OpenGL**	**GLU**	**FreeGlut**
g++	main.cpp	-lGL	-lGLU	-lglut

Hello Triangle code

This code creates a triangle that looks like this:

```
#ifdef __MINGW32__
#include <windows.h>
#endif

#ifdef WIN32
#include <windows.h>
#endif

#include <GL/freeglut.h>

/*F**********************************
 * main(int argc, char **argv)
 *
 * PURPOSE : entry into the program
 *
 * RETURN :  int
 *
 * NOTES :   Best viewed with emacs
 *F*/
int main(int argc, char **argv) {

  /*---- initialization -------------*/
  glutInit(&argc, argv);
  glutCreateWindow("Welcome to OpenGL");
  glutInitWindowSize(600, 300);
  glutReshapeWindow(600, 300);
  glutInitWindowPosition(50, 50);
  glutInitDisplayMode(GLUT_DOUBLE);
  /*---------------------------------*/

  /*---- draw the triangle ----------*/
  glBegin(GL_TRIANGLES);
  glColor3f(0.1, 0.2, 0.3);
  glVertex2f(0, 0);
  glVertex2f(1, 0);
  glVertex2f(0, 1);
  glEnd();
  glFlush();
  glFinish();
  glutMainLoop();
  /*---------------------------------*/

  return EXIT_SUCCESS;
}
```



Shapes we can draw

Triangles are one of the many things we can draw with the glBegin and glEnd functions, here is a list of a few things we can pass into glBegin and draw within minutes:

- GL_POINTS – A point is drawn for each call to glVertex.
- GL_LINES – Treats each pair of calls to glVertex as a line segment. At least 2 glVertex calls needed.
- GL_TRIANGLES – Treats each triplets of calls to glVertex as a triangle. At least 3 glVertex calls needed.
- GL_QUADS – Treats each quartet of calls to glVertex as a quad. At least 4 glVertex calls needed.
- GL_POLYGON – Draws a single polygon with an arbitrary amount of sides. Unlike the other shapes every polygon requires it's own glBegin and glEnd, whereas you can draw multiple triangles with only one glBegin and glEnd.

References

 https://en.wikipedia.org/wiki/OpenGL (2015)

 https://www.opengl.org/ (2015)