What is OpenGL?

• Low level 2D and 3D Graphics Library
• Competitor to Direct3D (the rendering part of DirectX)
• Used in: CAD, virtual reality, scientific visualization, information visualization, flight simulation, **video games**
• An abstraction layer to the graphics hardware - hides complexity of talking to different systems with different hardware behind a single clean API
OpenGL Pipeline

- Vertex data
- Display lists
- Rasterization
- Per-vertex operations
- Per-fragment operations
- Per-pixel operations
- Texture assembly
- Framebuffer
OpenGL Code

```c
void display() {
    glClear(GL_COLOR_BUFFER_BIT);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glFrustum( -1, 1, -1, 1, 1, 1000 );
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    glBegin(GL_POLYGON);
    glColor3f(1,0,0);
    glVertex3f( -1, -1, 0 );
    glVertex3f( -1, 1, 0 );
    glVertex3f( 1, 1, 0 );
    glVertex3f( 1, -1, 0 );
    glEnd();
}
```

- write background color to every pixel in working buffer ("active command")
- select which matrix to modify ("state change")
- multiply on a projection transformation
- overwrite a matrix with the identity matrix
- multiply on a projection transformation
- set some per-vertex attribute data
- submit some vertices (in "modeling coordinates")
- each vertex will be red because state has not changed
- describe the shape to be rendered
GLUT

• OpenGL Utility Toolkit
• Performs system-level I/O with the operating system
  – Simple window manager that handles graphics contexts, input via callbacks and timing
• Easily draws primitives (cubes, spheres, teapots, etc.)
• Cross-platform (GLX for X Window System, GLEW for Microsoft Windows)
• Designed for small to medium sized applications
Setup a Window with GLUT

```c
#include <GL/glut.h>

int main(int argc, char** argv)
{
    glutInit(&argc, argv); // init GLUT and let it grab any arguments it wants (must be called before any other GLUT function)
    glutInitWindowSize(width, height); // set the window size
    glutInitDisplayMode(GLUT_RGBA); // lots of options here for different modes
    glutCreateWindow(argv[0]); // create the window

    // register callbacks (next slide)

    glutMainLoop(); // hand over control to GLUT
    // code written past here will never run!
    return 0;
}
```
GLUT Callbacks

// called when the screen should be redrawn
void cb_display();

// called on mouse state change
void cb_mouse(int button, int state, int x, int y);

// called on mouse moves
void cb_motion(int x, int y);

// called on keyboard press
void cb_keyboard(unsigned char key, int x, int y);

// called when window is resized
void cb_reshape(int w, int h);

// called when GLUT gets bored
void cb_idle();

// register the callbacks so GLUT knows which functions to call
glutDisplayFunc(cb_display);
glutMouseFunc(cb_mouse);
glutMotionFunc(cb_motion);
glutMotionFunc(cb_motion);
glutKeyboardFunc(cb_keyboard);
glutReshapeFunc(cb_reshape);
glutIdleFunc(cb_idle);
Resources

- **OpenGL Programming Guide** (The Red Book)
  http://www.glprogramming.com/red/

- **Drawing Geometric Primitives**
  http://www.glprogramming.com/red/chapter02.html#name2

- **PyOpenGL Man Pages** (Great OpenGL API doc)
  http://pyopengl.sourceforge.net/documentation/manual/index.xml

- **NeHe OpenGL Tutorials**
  http://nehe.gamedev.net/