

CS770/870 Fall 2008

Introduction: Computer Graphics with OpenGL

10:43

CS770/870 Fall 2008 Bergeron/Plumlee

1

Preview

- Problem of drawing to a screen
- Setting up your first window with GLUT
- Drawing your first OpenGL primitives
- Changing sizes, colors, etc. (OpenGL state)
- An object-oriented approach

10:43

CS770/870 Fall 2008 Bergeron/Plumlee

2

The Problem

- How do you control the contents of a computer screen from a program?
 - What accelerator hardware is present?
 - What format do you send a frame (picture) in?
 - How do you synchronize with display hardware to show a frame exactly when you want?
 - How do you get keyboard and mouse input from the user in an interactive application?

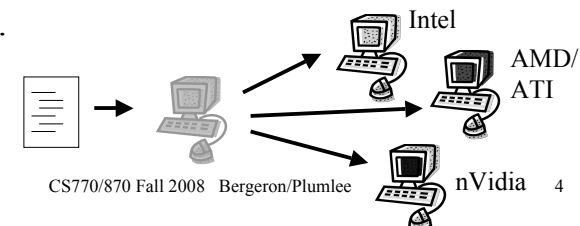
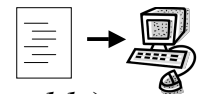
10:43

CS770/870 Fall 2008 Bergeron/Plumlee

3

Some Solutions

- Write to specific hardware (*not portable*)
- Write to a model of the hardware, create drivers to map the model → hardware
 - DirectX (*Windows-centric*)
 - Java (*cross-platform*)
 - OpenGL (*cross-platform, supports 3D well*)
 - Others...



10:43

CS770/870 Fall 2008 Bergeron/Plumlee

4

OpenGL & GLUT

- OpenGL talks to the graphics hardware
 - Frame buffer to construct image in right format
 - Synchronization of frames for display
 - Provides programming model that allows underlying drivers to exploit hardware acceleration
- GLUT does same for talking to OS
 - Creating/resizing windows
 - Accepting user input (mouse, keyboard)

10:43

CS770/870 Fall 2008 Bergeron/Plumlee

5

A GLUT/OpenGL Program

```
int main( int argc, char **argv )
{
    glutInit( &argc, argv ); // initialize toolkit
    // set the display mode & size
    glutInitDisplayMode( GLUT_SINGLE | GLUT_RGB );
    glutInitWindowSize( windowWidth, windowHeight );
    // set window position on screen
    glutInitWindowPosition( 100, 150 );
    // open the screen window
    glutCreateWindow( "My New Program!" );
    // register callback functions
    glutDisplayFunc( redraw );
    glutReshapeFunc( reshape );
    glutKeyboardFunc( doKey );
    glClearColor( 1.0, 1.0, 1.0, 1.0 ); // white
    glutMainLoop(); // go into a perpetual loop
}
```

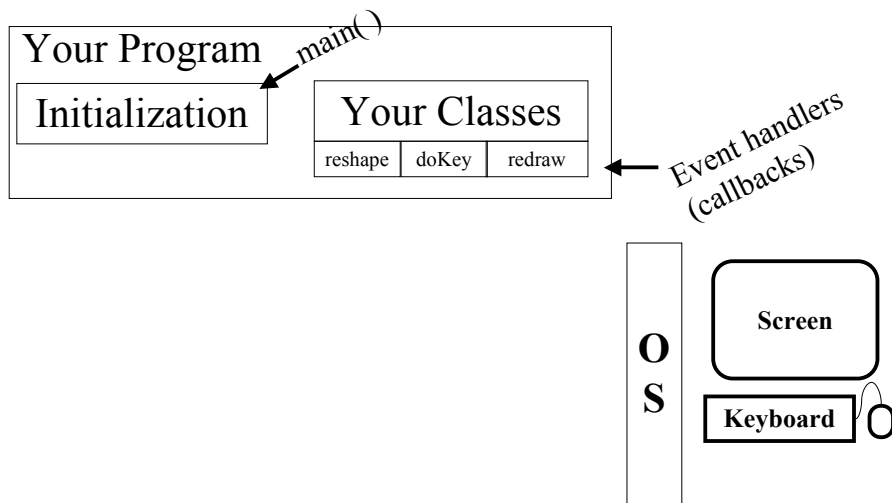


10:43

CS770/870 Fall 2008 Bergeron/Plumlee

6

A GLUT/OpenGL Program

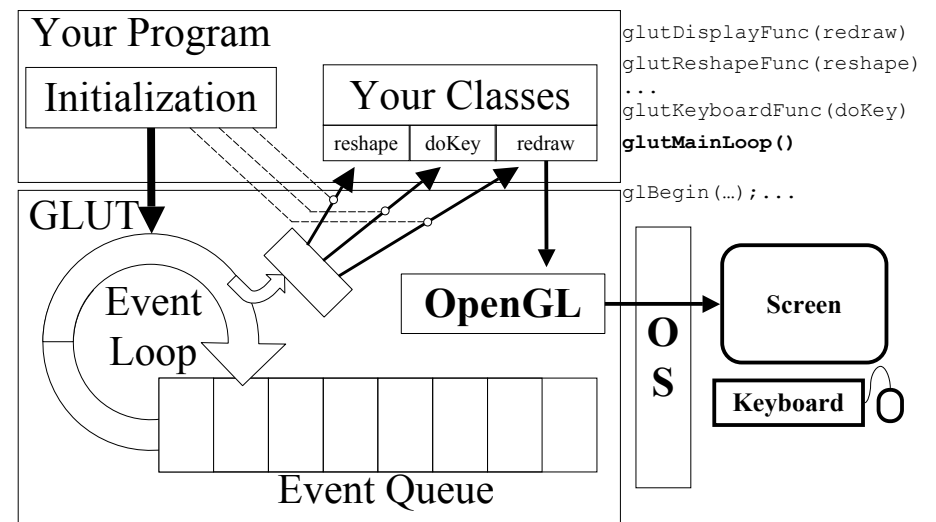


10:43

CS770/870 Fall 2008 Bergeron/Plumlee

7

A GLUT/OpenGL Program



10:43

CS770/870 Fall 2008 Bergeron/Plumlee

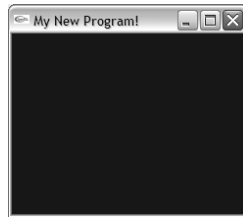
8

A GLUT/OpenGL Program

```
void redraw( void ) {
    glClear( GL_COLOR_BUFFER_BIT ); // clear screen
    glFlush(); // send all output to display
}

void doKey( unsigned char key, int x, int y ) {
    // set the bg color to a vibrant red
    glClearColor( 1.0, 0.0, 0.0, 1.0 );
    redraw();
}

void reshape ( int x, int y ) {
    redraw();
}
```

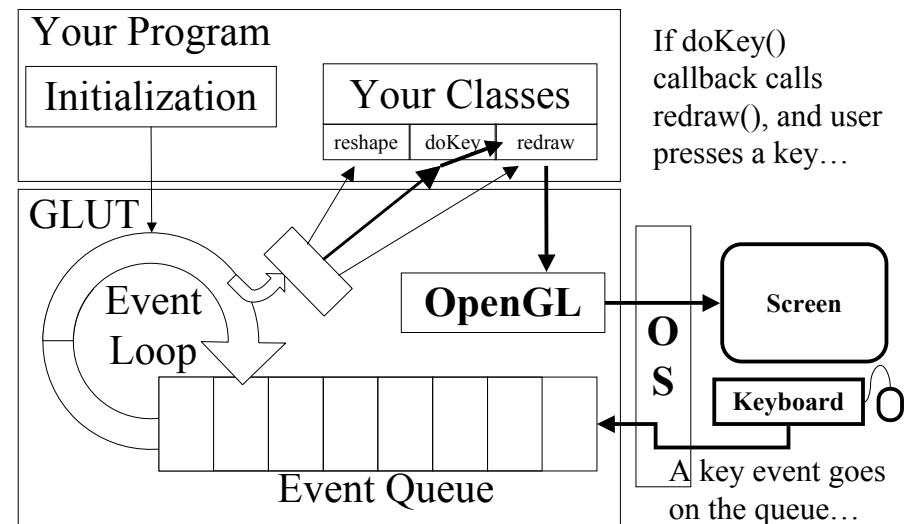


10:43

CS770/870 Fall 2008 Bergeron/Plumlee

9

A GLUT/OpenGL Program



10:43

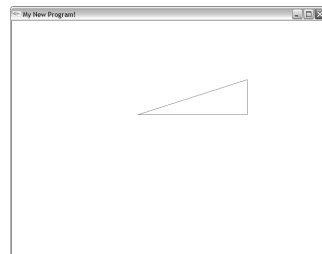
CS770/870 Fall 2008 Bergeron/Plumlee

10

Drawing with OpenGL

- In redraw function after `glClear(...)` call

```
glColor3f(0.3f, 0.3f, 1.0);
glBegin(GL_LINE_STRIP);
    glVertex2f( -0.2, 0.2 );
    glVertex2f( 0.5, 0.2 );
    glVertex2f( 0.5, 0.5 );
    glVertex2f( -0.2, 0.2 );
glEnd();
```



10:43

CS770/870 Fall 2008 Bergeron/Plumlee

11

Drawing with OpenGL

- Four functions already...
 - Changing color


```
glColor3f( 0.3f, 0.3f, 1.0 );
```
 - Begin a primitive drawing mode


```
glBegin( GL_LINE_STRIP );
```
 - Specify 2D vertices


```
glVertex2f( -0.2, 0.2 );
```
 - End drawing block for last primitive


```
glEnd();
```

10:43

CS770/870 Fall 2008 Bergeron/Plumlee

12

OpenGL State

- Most functions alter OpenGL state
- In some ways, OpenGL is *like* a singleton class, functions are *like* object methods
 - Only one instance
 - Maintains state between calls
- A few state quantities (of dozens):
 - Line-width, point-size
 - Material color, clear color (background)
 - Drawing mode, matrix mode

`glPointSize`
`glColor3f`
`glBegin`

10:43

CS770/870 Fall 2008 Bergeron/Plumlee

13

OpenGL Coordinate System

- Default:
 - Center at (0.0, 0.0)
 - Lower-left: (-1.0, -1.0); upper-right: (1.0, 1.0)
- Some initialization magic (*explained later...*)

```
glMatrixMode( GL_PROJECTION );  
glLoadIdentity();  
gluOrtho2D( 0.0, windowWidth, 0.0, windowHeight );
```

- Lower-left at (0, 0)
- Upper-right at (windowWidth, windowHeight)

`glVertex2f`

10:43

CS770/870 Fall 2008 Bergeron/Plumlee

14

Learning OpenGL and GLUT

- Great resources: online manuals
 - OpenGL
<http://www.opengl.org/sdk/docs/man/>
 - GLUT
<http://www.opengl.org/documentation/specs/glut/spec3/spec3.html>
- What they give you:
 - More argument constants, options, explanations
 - Additional commands
 - Ideas for that last 10%?

10:43

CS770/870 Fall 2008 Bergeron/Plumlee

15

An Object-Oriented Approach

- Want to be able to have objects draw themselves: houses, teapots, alien spacecraft
- OpenGL only provides primitive operations
 - Draw point
 - Draw line
 - Draw polygon
 - Others we'll see later
- What's a good design?

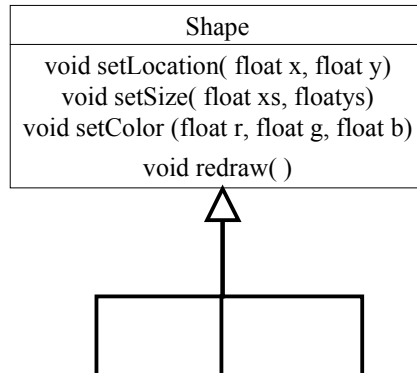
10:43

CS770/870 Fall 2008 Bergeron/Plumlee

16

What's a Good O-O Design?

- A class of objects that
 - Keeps own state: position, size, color, ...
 - Can draw itself
 - and more ...
- A mechanism for drawing all these shapes from the redraw callback in main()



10:43

CS770/870 Fall 2008 Bergeron/Plumlee

17

What's it Look Like in C++?

- Base class declaration (in header file)

```
class Shape
{
public:
    Shape();
    virtual ~Shape();
    void setLocation( float x, float y );
    void setSize( float xs, float ys );
    void setColor( float r, float g, float b );
    virtual void redraw() = 0;
protected:
    float xLoc, yLoc; // location of the object
    float xSize, ySize; // size of the object
    float red, green, blue; // color
};
```

See Demo1 Source for full implementation

10:43

CS770/870 Fall 2008 Bergeron/Plumlee

18

What's it Look Like in C++?

- One way to create a list of Shapes

```
#include <vector>
...
std::vector<Shape> shapes;
...
std::vector<Shape*>::iterator it;
for (it = shapes.begin(); it < shapes.end(); it++)
    (*it).redraw();
```

See Demo1 Source for full implementation

10:43

CS770/870 Fall 2008 Bergeron/Plumlee

19

Review

- OpenGL one solution to drawing problem
- Created first simple OpenGL/GLUT app
- OpenGL state
- OpenGL coordinates
- An object-oriented approach

10:43

CS770/870 Fall 2008 Bergeron/Plumlee

20

Next

- Simple Interactivity
 - Mouse input & rubberbanding
 - Keyboard input
 - GLUT Menus
- GLUI—A basic user interface API