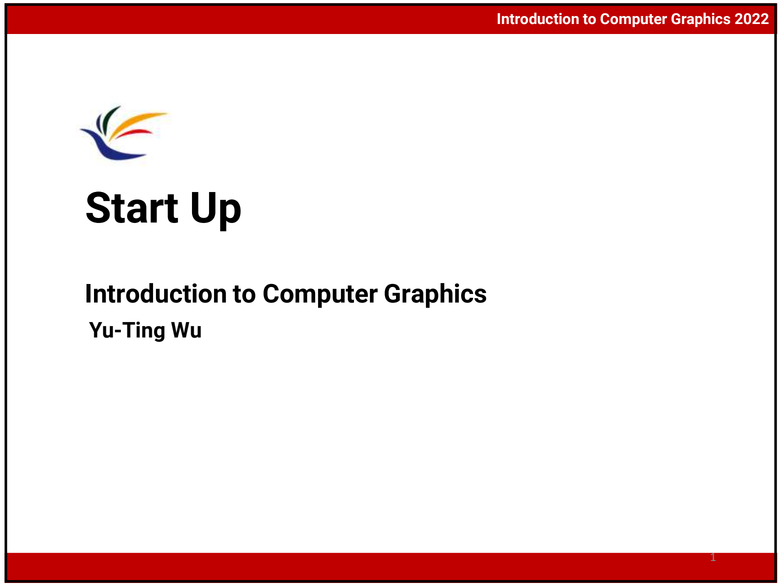
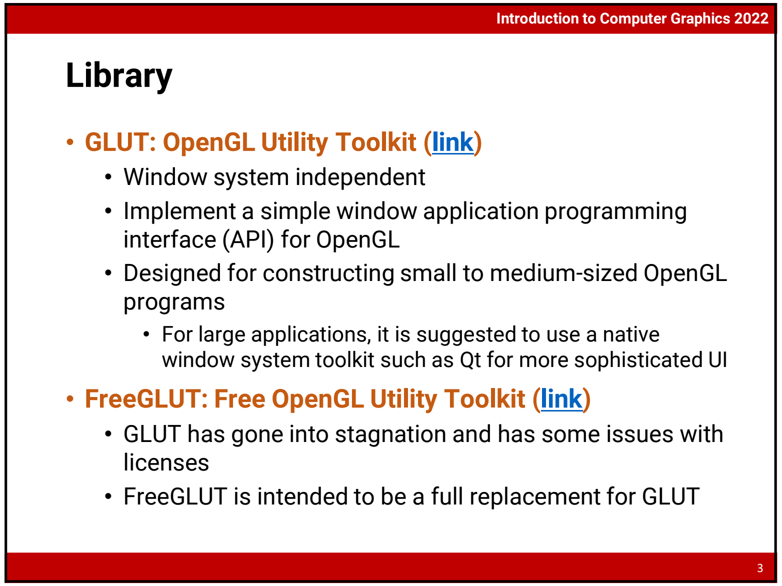


Start Up

Introduction to Computer Graphics

Yu-Ting Wu

Library



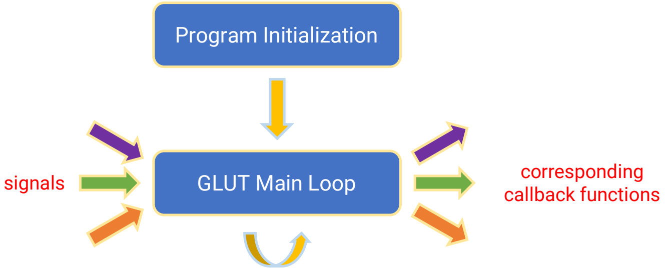
Library

- **GLUT: OpenGL Utility Toolkit (link)**
  - Window system independent
  - Implement a simple window application programming interface (API) for OpenGL
  - Designed for constructing small to medium-sized OpenGL programs
    - For large applications, it is suggested to use a native window system toolkit such as Qt for more sophisticated UI
- **FreeGLUT: Free OpenGL Utility Toolkit (link)**
  - GLUT has gone into stagnation and has some issues with licenses
  - FreeGLUT is intended to be a full replacement for GLUT

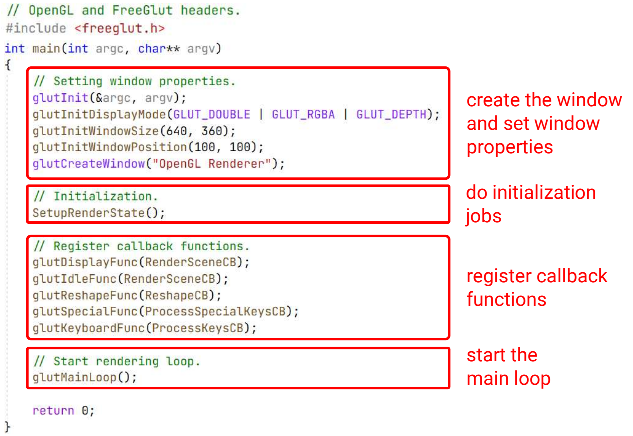
Program

Program Structure Overview

- OpenGL programs are event-driven



The First Program



```
// OpenGL and FreeGlut headers.
#include <freeglut.h>
int main(int argc, char** argv)
{
    // Setting window properties.
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGBA | GLUT_DEPTH);
    glutInitWindowSize(640, 360);
    glutInitWindowPosition(100, 100);
    glutCreateWindow("OpenGL Renderer");

    // Initialization.
    SetupRenderState();

    // Register callback functions.
    glutDisplayFunc(RenderSceneCB);
    glutIdleFunc(RenderSceneCB);
    glutReshapeFunc(ReshapeCB);
    glutSpecialFunc(ProcessSpecialKeysCB);
    glutKeyboardFunc(ProcessKeysCB);

    // Start rendering loop.
    glutMainLoop();

    return 0;
}
```

create the window and set window properties

do initialization jobs

register callback functions

start the main loop

Create a OpenGL (GLUT) Window

- *void **glutInit**(int *argc, char **argv);*
  - Initialize the GLUT library

```
glutInit(&argc, argv);
```

- *int **glutCreateWindow**(char *name);*
  - Create a top-level window

```
glutCreateWindow("OpenGL Renderer");
```

Setting Window Properties

- *void **glutInitWindowSize**(int width, int height);*
  - Set the initial window size
- *void **glutInitWindowPosition**(int x, int y);*
  - Set the initial window position

```
glutInitWindowSize(640, 360);
glutInitWindowPosition(100, 100);
```

- *void **glutInitDisplayMode**(unsigned int mode);*
  - Set the initial display mode
  - https://www.opengl.org/resources/libraries/glut/spec3/node12.html

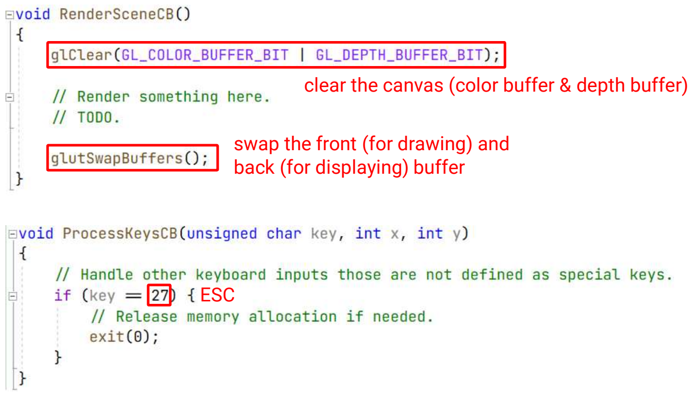
```
glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGBA | GLUT_DEPTH);
```

Setting Callback Functions

- Register the callback functions when receiving events
- Commonly used
  - *glutDisplayFunc*
  - *glutIdleFunc*
  - *glutReshapeFunc*
  - *glutKeyboardFunc / glutSpecialFunc*
  - *glutMouseFunc*
  - *glutMenuStatusFunc*
- Each callback function has its own input format
- Please refer to the following page for all possible callback functions
  - https://www.opengl.org/resources/libraries/glut/spec3/node45.html

9

Setting Callback Functions (cont.)



```
void RenderSceneCB()
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

    // Render something here.
    // TODO.

    glutSwapBuffers();
}
```

clear the canvas (color buffer & depth buffer)

swap the front (for drawing) and back (for displaying) buffer

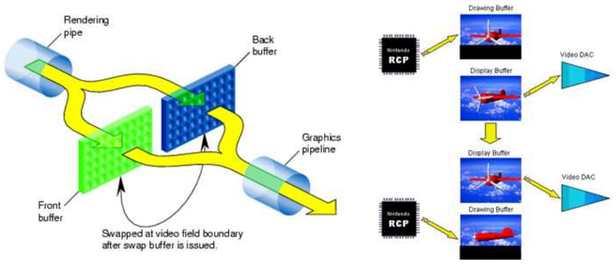
```
void ProcessKeysCB(unsigned char key, int x, int y)
{
    // Handle other keyboard inputs those are not defined as special keys.
    if (key == 27) {
        // Release memory allocation if needed.
        exit(0);
    }
}
```

ESC

10

Double Buffers

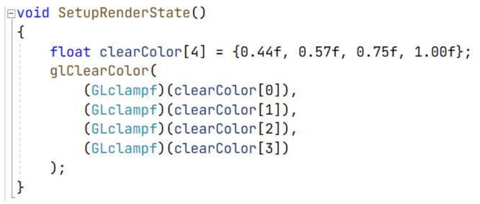
- Prevent artifacts due to potentially seeing parts of an incomplete frame (that is currently drawn)
  - Set the display mode to ***GLUT_DOUBLE*** in the ***glutInitDisplayMode*** function
  - Call ***glutSwapBuffers*** after rendering finished



11

Initialization

- *void **glClearColor**(GLfloat red, GLfloat green, GLfloat blue, GLfloat alpha);*
  - Set the color to clear the color buffer

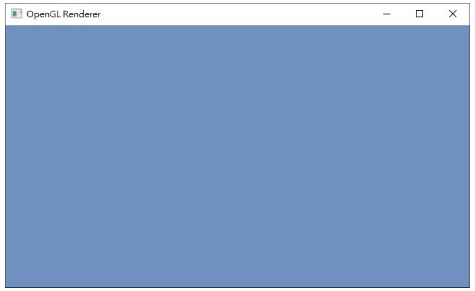


```
void SetupRenderState()
{
    float clearColor[4] = {0.44f, 0.57f, 0.75f, 1.00f};
    glClearColor(
        (GLclampf)(clearColor[0]),
        (GLclampf)(clearColor[1]),
        (GLclampf)(clearColor[2]),
        (GLclampf)(clearColor[3])
    );
}
```

12

Start the Main Rendering Loop

- *void **glutMainLoop**(void);*
  - Enter the GLUT event processing loop
  - OpenGL programs are event-driven



Any Questions?