COMP5514
Computer Image Generation in C/C++

Lab 02

Prof. George Baciu
csgeorge@comp.polyu.edu.hk
www.comp.polyu.edu.hk/~csgeorge/comp5514/lab/

Department of Computing, --- The
Hong Kong Polytechnic University

Contents: OpenGL API

PART A: OPENGL API

- OpenGL Environment
- OpenGL Architecture
- OpenGL Documentation

PART B: LIBRARIES

- GL Library
- GLU Library
- GLUT Library

PART C: EVENT HANDLING

- Basic Event-Handling
- Call-back functions
- Graphics Interactions
- Basic OpenGL program

OpenGL API

- Computing Environment

Paths and Files

- **OpenGL and GLUT libraries:**
  - see the link on the web page

- **Example Code:**
  - see the link on the web page

- **Map Network Drive: J:\cyghome**

Paths and Files

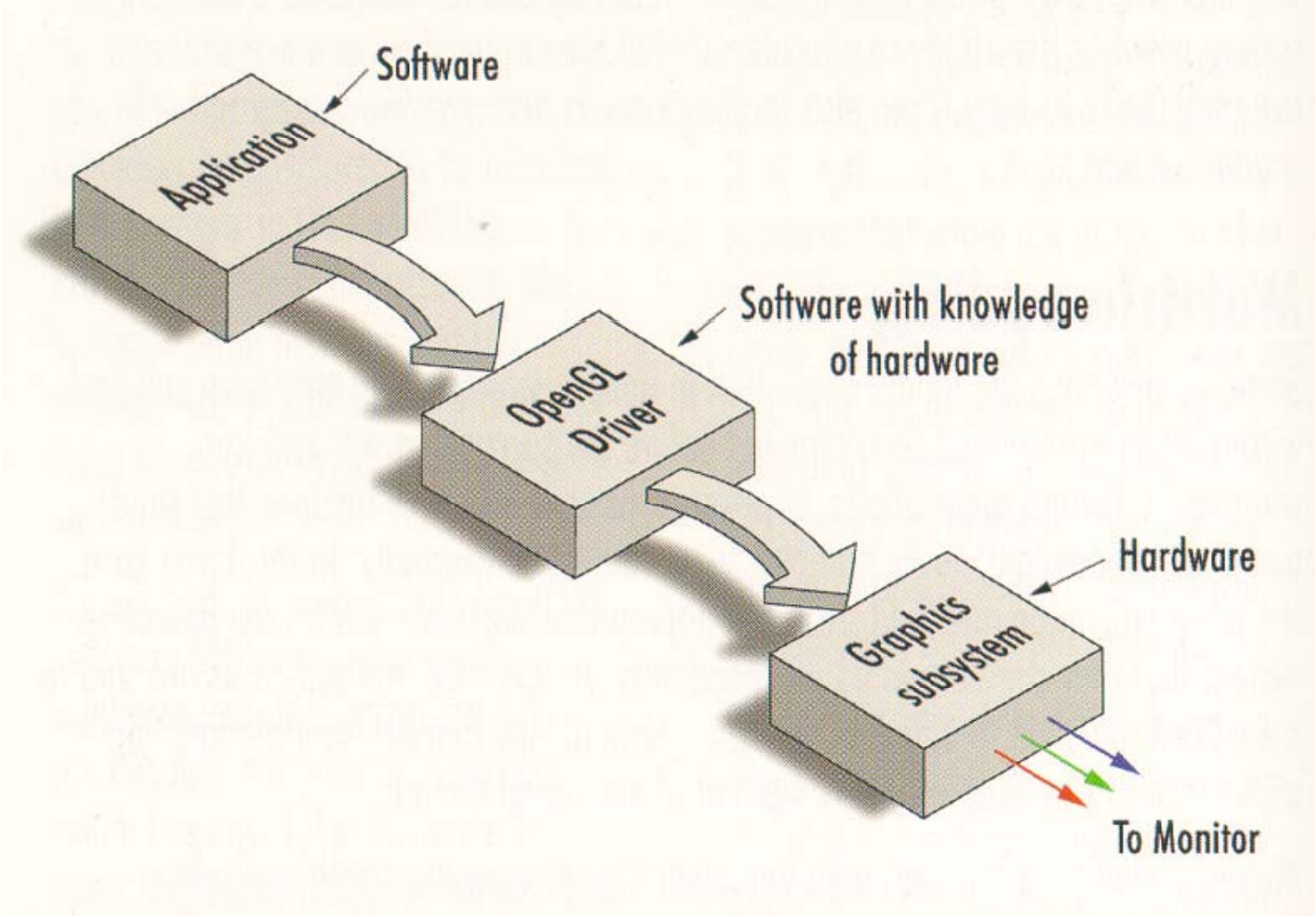
- **Cygwin Environment**
- **Gnu C/C++:**
  - **J:\cyghome**

- **Use MAKE:**
  - **see the example:** lab02.zip

OpenGL API

- OpenGL Architecture
- OpenGL Data Flow

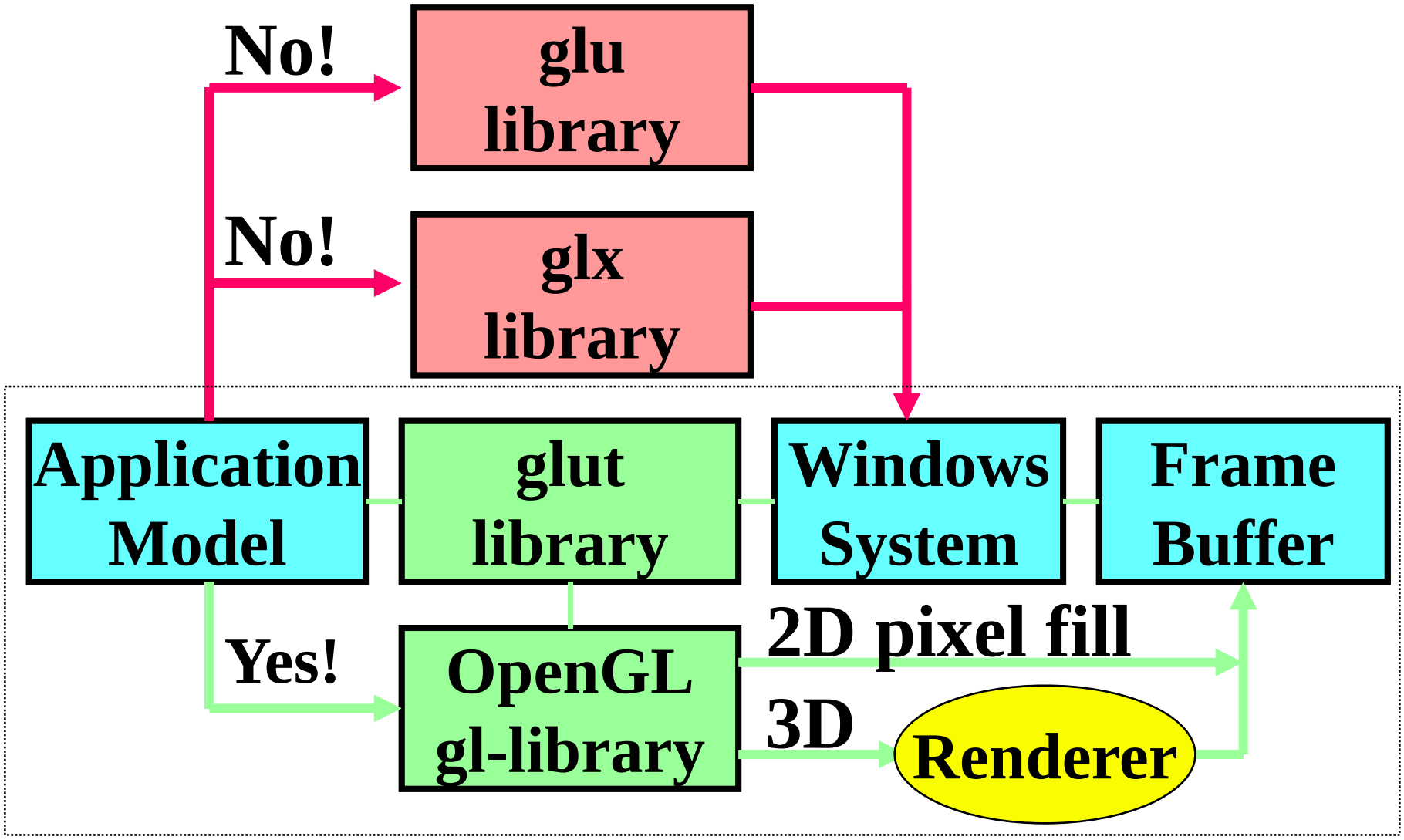
The Graphics API



OpenGL API

- OpenGL Libraries
- OpenGL Loop Feedback

OpenGL API


No!
glu library
No!
glx library
Application Model
glut library
Windows System
Frame Buffer
Yes!
OpenGL gl-library
2D pixel fill
3D
Renderer

OpenGL API

Library	*Prefix*	*Example*
OpenGL	**gl**	**glColor**
GLUT	**glut**	**glutInit**
GL Utility	**glu**	**gluSphere**
Auxiliary	**aux**	**auxInitWindow**
GL (sgi)	**None**	**winopen**

OpenGL API

- OpenGL Event Handling
- OpenGL Interactions

Interactions

- **Interactions via an *event-handler***
- **Input devices generate events**
- **Applications must handle them**
- **A response is generated:**
  - a visual change
  - another action

Event Handling

1. Polling: synchronous

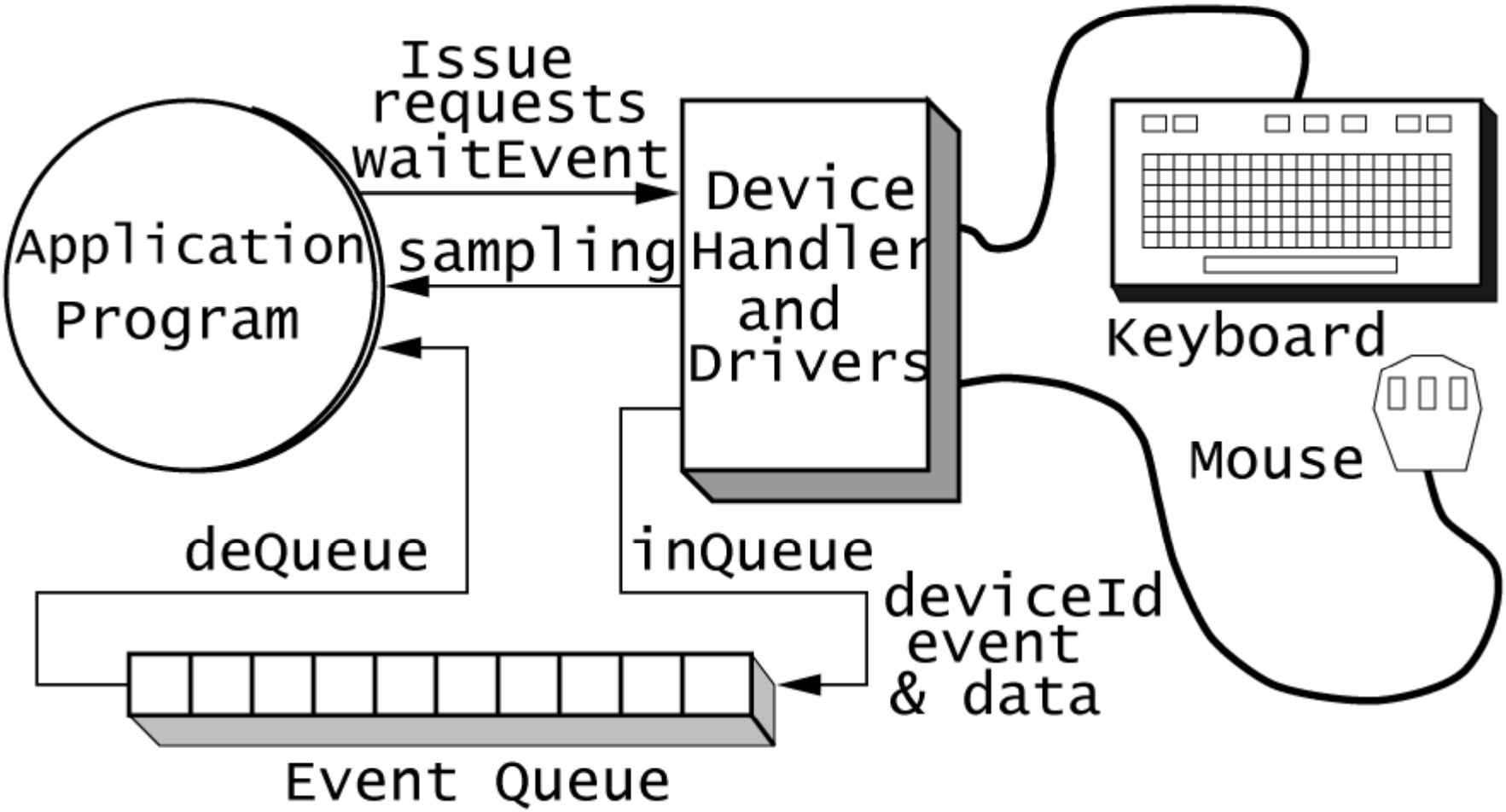
   the application is busy waiting until it receives an event.

2. Event-Driven: asynchronous

   the events generate interrupts

   events are stored in a queue

Event Handler



OpenGL Example

- OpenGL Simple C/C++
- OpenGL Programming
- Download lab02.zip

OpenGL Call-Back

```
int  main(int argc,  char** argv)
{       glutInit(&argc, argv);
        glutInitWindowPosition(0,0);
        glutInitWindowSize(400,400);
        glutCreateWindow("myWindow");
        glutReshapeFunc(myReshape);
        glutMouseFunc(myMouseCoord);
        glutDisplayFunc(myDisplay);
        glutMainLoop();
}
```

OpenGL Call-Back

```
void myDisplay (void)
{       glClearColor (0.0, 0.0, 0.0, 0.0);
        glClear (GL_COLOR_BUFFER_BIT);
        glutSwapBuffers ();
}
```

Include Files

For the Microsoft Windows environment:

```
#include <stdio.h>
#include <stdlib.h>

#include <windows.h>
#include <gl/gl.h>
#include <gl/glut.h>
```

Equivalent Loop

```
notQuit = TRUE;
while (notQuit)
{       processTheQueue();
        myDisplay();
        processTheQueue();
}
```

Queue Processing:

```
void  processTheQueue(void)
{       while ( Qtest() )
        {  device = Qread( &deviceVal );
           switch (device)
           {      case GLUT_LEFTBUTTON:
                        if (deviceValue==GLUT_MOUSEDOWN)
                              myMouseCoord(eventDataPtr);
                  break;
                  case GLUT_ESCAPE:
                        notQuit = FALSE;
                  break;
           }
        }
}
```

Try things out. . .

Download lab02.zip from the lab page!

The End