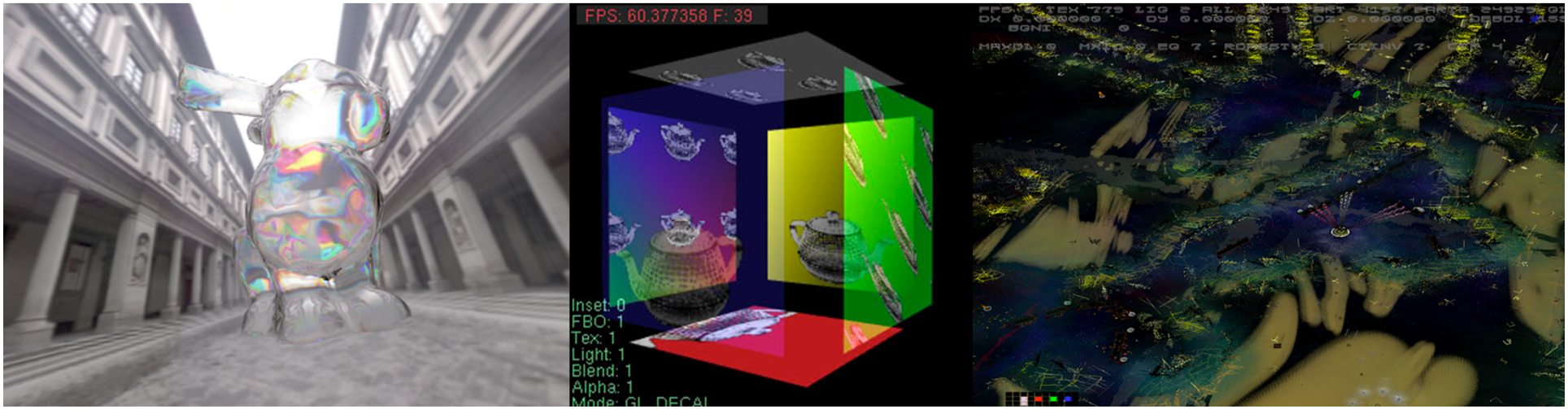
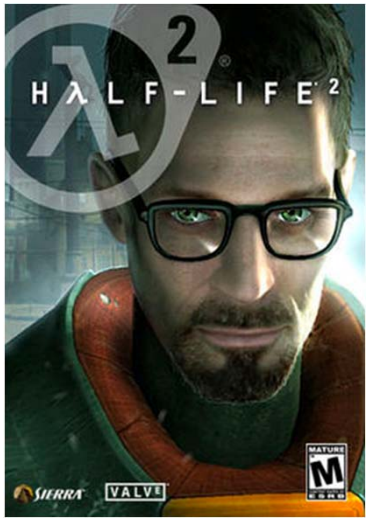
Introduction to OpenGL and GLUT

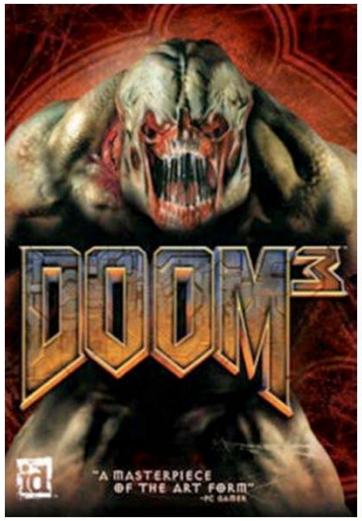
What's OpenGL?

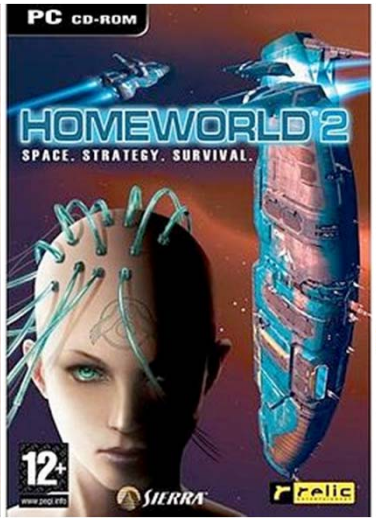
- An Application Programming Interface (API)
- A low-level graphics programming API
  - Contains over 250 functions
  - Developed by Silicon Graphics Inc. (SGI) in 1992
  - Most recent version: 4.2, released on 08/08/2011

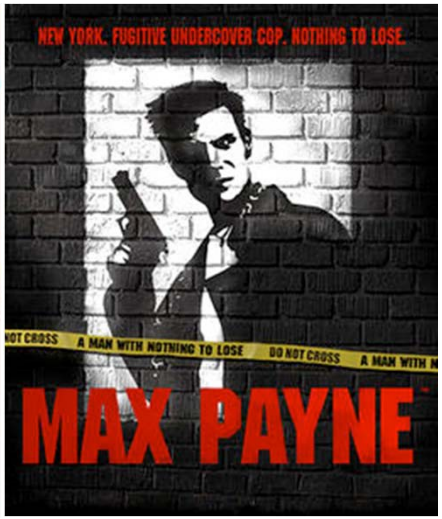


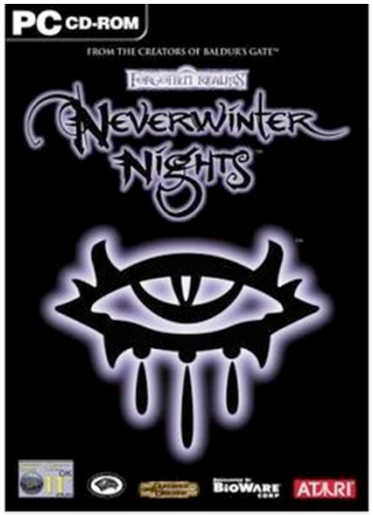
OpenGL Applications



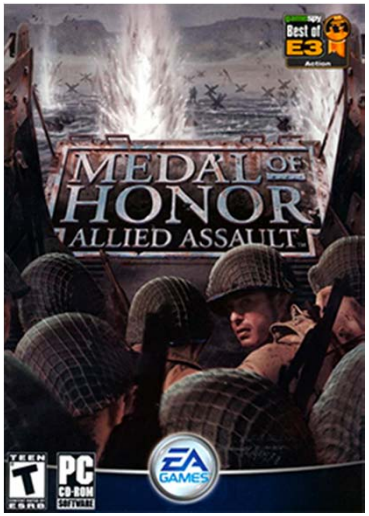


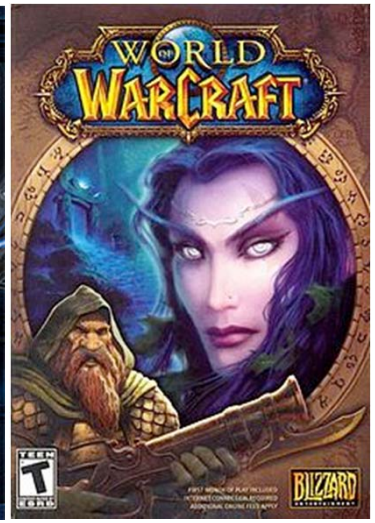






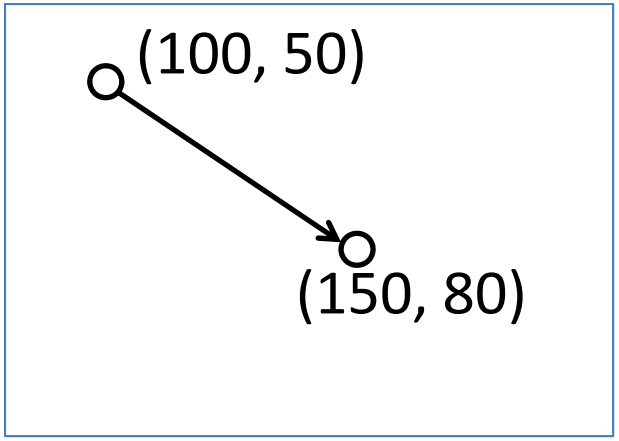
And more…





Portability

- Independent of display devices
- Independent of window systems
- Independent of operating systems



- Device 1:
  Line(100, 50, 150, 80)
- Device 2:
  MoveTo(100, 50)
  LineTo(150, 80)

OpenGL Basics

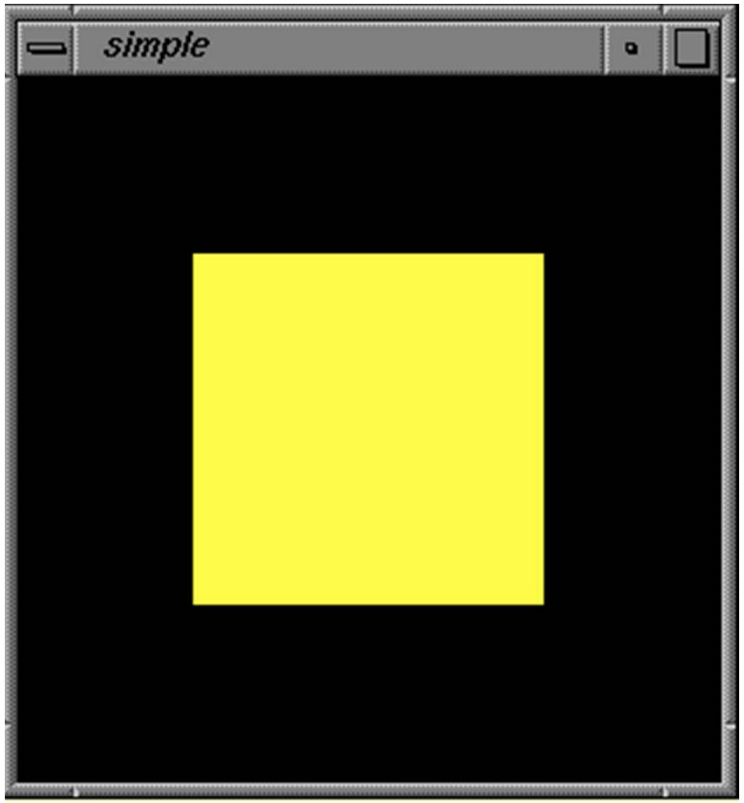
- The main use: Rendering.

> **Rendering** is the process of generating an image from a model (or models in what collectively could be called a *scene* file), by means of computer programs.
>
> -----Wikipedia

- OpenGL can render:
  - Geometric primitives
  - Bitmaps and images

An Example

```
void Display()
{
    glClear(GL_COLOR_BUFFER_BITS);
    glColor4f(1, 1, 0, 1);
    glBegin(GL_POLYGON);
    glVertex2f(-0.5, -0.5);
    glVertex2f(-0.5,  0.5);
    glVertex2f( 0.5,  0.5);
    glVertex2f( 0.5, -0.5);
    glEnd();
    glutSwapBuffers();
}
```



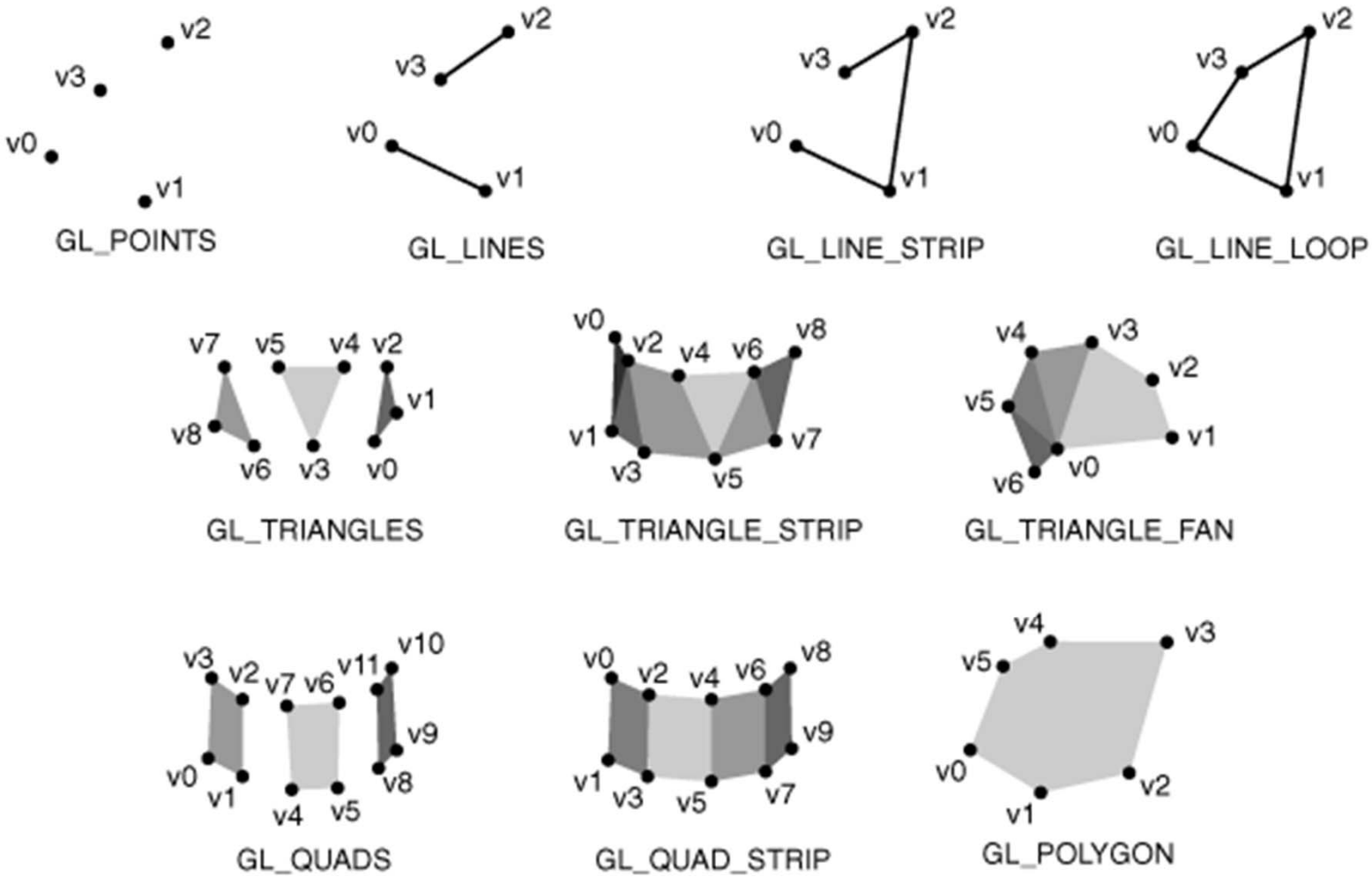
Primitives

- Primitives are specified as:

```
glBegin(primType);
// your primitive vertices here.
// ...
// ...
glEnd();
```

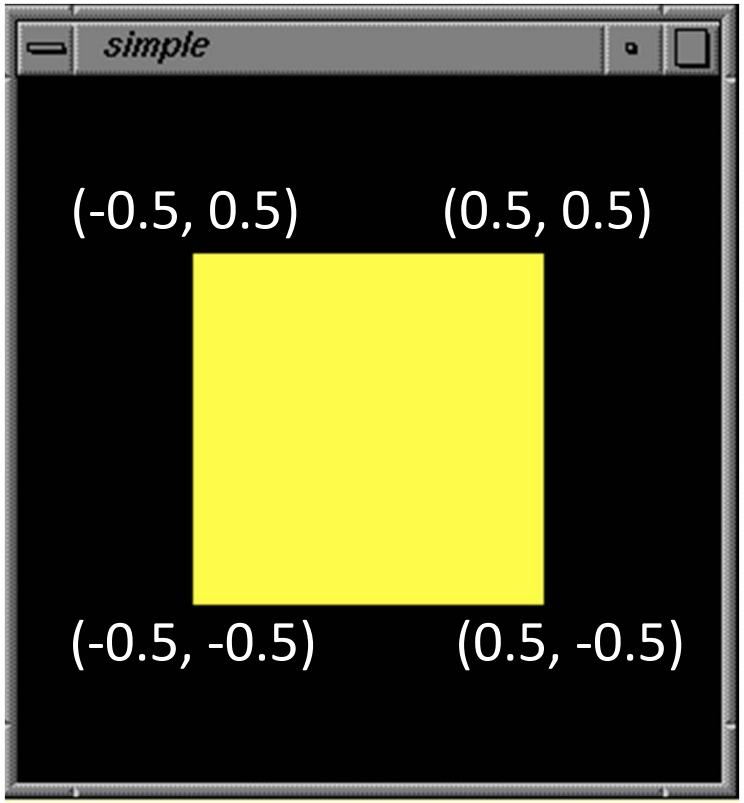
- PrimType:
  - GL_POINTS, GL_LINES, GL_TRIANGLES, …

Primitive Types



An Example

```
void Display()
{
    glClear(GL_COLOR_BUFFER_BITS);
    glColor4f(1, 1, 0, 1);
    glBegin(GL_POLYGON);
    glVertex2f(-0.5, -0.5);
    glVertex2f(-0.5,  0.5);
    glVertex2f( 0.5,  0.5);
    glVertex2f( 0.5, -0.5);
    glEnd();
    glutSwapBuffers();
}
```



Vertices

v stands for vectors

`glVertex2fv(x, y)`

`glVertex2f(x, y)`

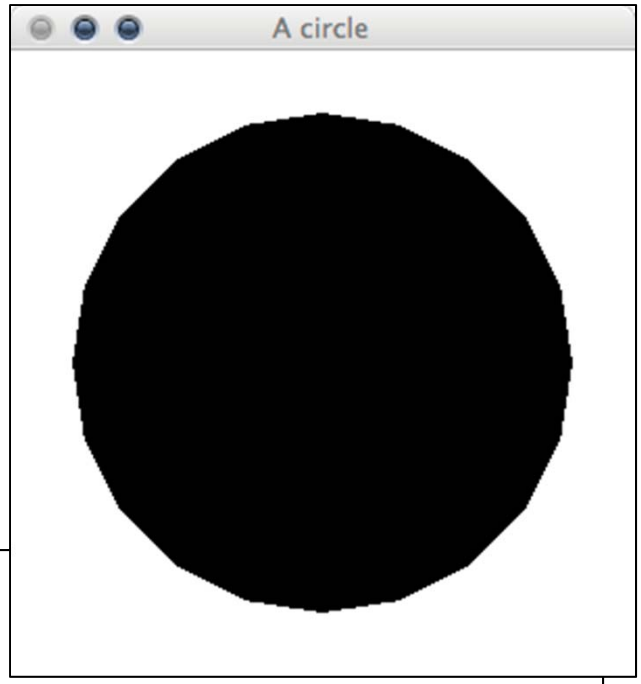
Number of dimensions/components

- 2: (x, y)
- 3: (x, y, z) or (r, g, b)
- 4: (x, y, z, w) or (r, g, b, a)

Format

- b: byte
- ub: unsigned byte
- i: int
- ui: unsigned int
- f: float
- d: double

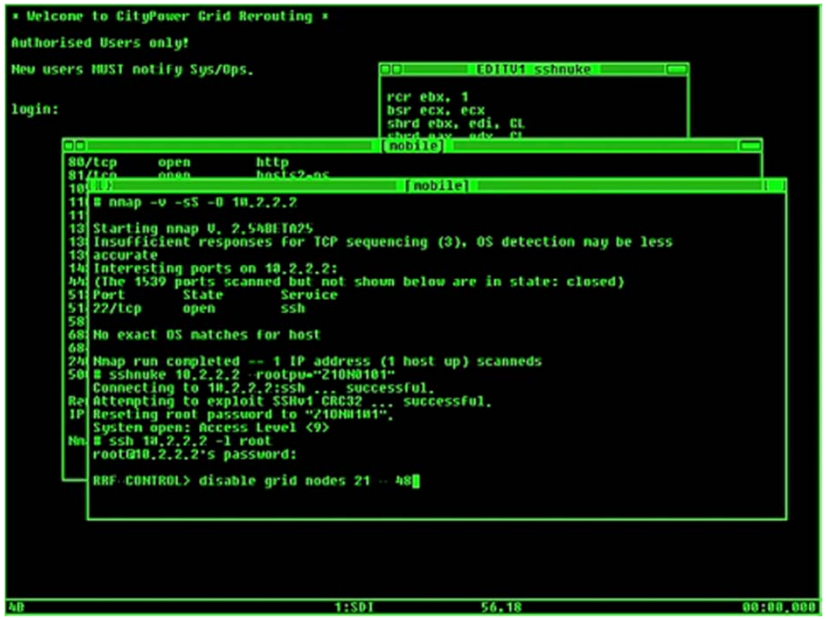
An Example




```
void Draw_Circle()
{
    glColor4f(0, 0, 0, 1);
    int   number = 20;                //number of vertices
    float radius = 0.8f;              //radius
    float  twoPi = 2.0f * 3.14159f;
    glBegin(GL_POLGYON);
    for(int i = 0; i <number; i++)
        glVertex2f(radius*cosf(i*twoPi/number),
                   radius*sinf(i*twoPi/number));
    glEnd();
}
```

Window-Based Programming

- Most modern operating systems are windows-based.



Not window-based

Window-based

Window-Based Programming

- OpenGL is independent of window systems.
  - No functions for window operations, such as creating, resizing, event handling, …
  - This is to ensure portability across different platforms
  - Now we have a problem: OpenGL doesn’t work by itself.

Graphics User Interfaces

- A Graphics User Interface (GUI) provides functions to:
  - Create windows, sliders, menus, buttons…
  - Resize windows
  - Handle mouse click, keyboard….
  - …

GLX for X window system
AGL for Mac
WGL for Microsoft Windows

No portability…

GLUT (OpenGL Utility Toolkit)

- GLUT provides basic functions to operate windows.
- GLUT is cross-platform: X window, MS window, and Mac OS.
- No sliders, no buttons, no menus, …

GLUT Basics

- Program structure:
  1. Configure and open a window (GLUT)
  2. Initialize OpenGL. (Optional)
  3. Register callback functions. (GLUT)
     a. Display (OpenGL)
     b. Resize (OpenGL)
     c. Input/output
  4. Enter an event processing loop. (GLUT)

An Example

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

void main(int argc, char **argv)
{
    int mode=GLUT_RGB|GLUT_DOUBLE;
    glutInit(&argc, argv);
    glutInitDisplayMode(mode);
    glutInitWindowSize(800, 800);
    glutCreateWindow("Hello Graphics World");
    init();
    glutDisplayFunc(display);
    glutKeyboardFunc(key);
    glutMainLoop();
}
```

An Example

Display mode: RGB or color index?
Single frame buffer or double?

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

void main(int argc, char **argv)
{
    int mode=GLUT_RGB|GLUT_DOUBLE;
    glutInit(&argc, argv);
    glutInitDisplayMode(mode);
    glutInitWindowSize(800, 800);
    glutCreateWindow("Hello Graphics World");
    init();
    glutDisplayFunc(display);
    glutKeyboardFunc(key);
    glutMainLoop();
}
```

An Example

A window with resolution 800*800

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

void main(int argc, char **argv)
{
    int mode=GLUT_RGB|GLUT_DOUBLE;
    glutInit(&argc, argv);
    glutInitDisplayMode(mode);
    glutInitWindowSize(800, 800);
    glutCreateWindow("Hello Graphics World");
    init();
    glutDisplayFunc(display);
    glutKeyboardFunc(key);
    glutMainLoop();
}
```

An Example

Create the window with a title:
"hello graphics world"

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

void main(int argc, char **argv)
{
    int mode=GLUT_RGB|GLUT_DOUBLE;
    glutInit(&argc, argv);
    glutInitDisplayMode(mode);
    glutInitWindowSize(800, 800);
    glutCreateWindow("Hello Graphics World");
    init();
    glutDisplayFunc(display);
    glutKeyboardFunc(key);
    glutMainLoop();
}
```

An Example

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

void main(int argc, char **argv)
{
    int mode=GLUT_RGB|GLUT_DOUBLE;
    glutInit(&argc, argv);
    glutInitDisplayMode(mode);
    glutInitWindowSize(800, 800);
    glutCreateWindow("Hello Graphics World");
    init();
    glutDisplayFunc(display);
    glutKeyboardFunc(key);
    glutMainLoop();
}
```

An optional init function

An Example

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

void main(int argc, char **argv)
{
    int mode=GLUT_RGB|GLUT_DOUBLE;
    glutInit(&argc, argv);
    glutInitDisplayMode(mode);
    glutInitWindowSize(800, 800);
    glutCreateWindow("Hello Graphics World");
    init();
    glutDisplayFunc(display);
    glutKeyboardFunc(key);
    glutMainLoop();
}
```

Let GLUT know which function to call, when an event happens

An Example

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

void main(int argc, char **argv)
{
    int mode=GLUT_RGB|GLUT_DOUBLE;
    glutInit(&argc, argv);
    glutInitDisplayMode(mode);
    glutInitWindowSize(800, 800);
    glutCreateWindow("Hello Graphics World");
    init();
    glutDisplayFunc(display);
    glutKeyboardFunc(key);
    glutMainLoop();
}
```

A loop that waits for events

Event-Driven Callbacks

- Most window-based programs are event-driven:
  - Do nothing until an event happens,
  - Then call the corresponding callback function

- Events
  - Key press, mouse button click/release, resize, ….

GLUT Display

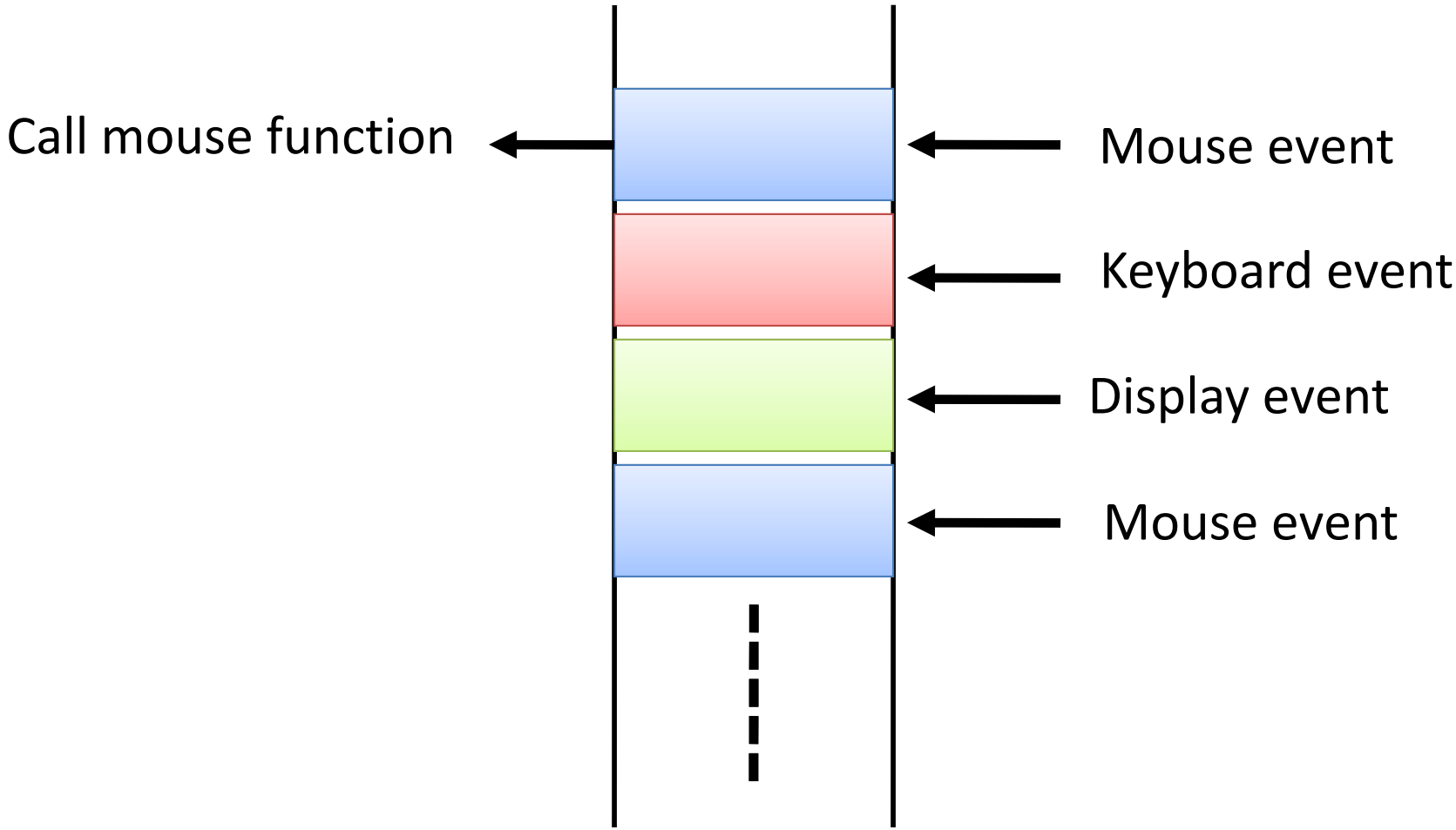
```
glutDisplayFunc(mydisplay);
```

- *`mydisplay`* is the name of a function that you will provide.
- You can use OpenGL to draw something.
- This function is called when the window needs to be refreshed.
- You can also signal a display event by youself:

```
glutPostRedisplay();
```

Event Queue

```
glutMainLoop();
```



Some Callback Functions

- `glutKeyboardFunc`
- `glutMouseFunc`
- `glutMotionFunc`
- `glutSpecialFunc`
- `glutIdleFunc`