CSC 307 1.0
Graphics Programming

Budditha Hettige
Department of Statistics and Computer Science

04

Graphics Programming
GLUT

Events in OpenGL

Event	Example	OpenGL Callback Function
Keypress	KeyDown KeyUp	`glutKeyboardFunc`
Mouse	leftButtonDown leftButtonUp	`glutMouseFunc`
Motion	With mouse press Without	`glutMotionFunc` `glutPassiveMotionFunc`
Window	Moving Resizing	`glutReshapeFunc`
System	Idle Timer	`glutIdleFunc` `glutTimerFunc`
Software	What to draw	`glutDisplayFunc`

GLUT Callback functions

- **Event-driven:** Programs that use windows
  - Input/Output
  - Wait until an event happens and then execute some pre-defined functions according to the user's input
- **Events** – key press, mouse button press and release, window resize, etc.
- *Your OpenGL program will be in infinite loop*

GLUT Callback Functions

- **Callback function** : Routine to call when an event happens
  - Window resize or redraw
  - User input (mouse, keyboard)
  - Animation (render many frames)
- "Register" callbacks with GLUT
  - glutDisplayFunc(my_display_func);
  - glutIdleFunc(my_idle_func);
  - glutKeyboardFunc(my_key_events_func);
  - glutMouseFunc (my_mouse_events_func);

Rendering Callback

- Callback function where all our drawing is done
- Every GLUT program must have a display callback
- glutDisplayFunc(***my_display_func***);
  /* this part is in main.c */

```
void my_display_func (void )
{
  glClear( GL_COLOR_BUFFER_BIT );
  glBegin( GL_TRIANGLE );
    glVertex3fv( v[0] );
    glVertex3fv( v[1] );
    glVertex3fv( v[2] );
  glEnd();
  glFlush();
}
```


Idle Callback Function

- Use for animation and continuous update
  - Can use *glutTimerFunc* or *timed callbacks* for animations
- glutIdleFunc(***idle***);
  - **glutIdleFunc(void (*func)(void)).**
  - glutIdleFunc(**MyidleFun**);

Function Name

```
void MyidleFun( void )
{
  /* change something */
  t += dt;
  glutPostRedisplay();
}
```


User Input Callbacks

- Process user input (Keyboard/ Mouse)
- glutKeyboardFunc(my_key_events); // for keyboard
- glutMouseFunc(my_mouse); // for mouse

glutKeyboardFunc(void (*func)(unsigned char key, int x, int y))
glutKeyboardFunc sets the keyboard callback for the *current window*

```
void my_key_events (char key, int x, int y )
{
   switch ( key )
   {
     case 'q' :     case 'Q' :
                    exit ( EXIT_SUCCESS);
                    break;
     case 'r' :     case 'R' :
                    rotate = GL_TRUE;
                    break;
     ...
   }
}
```

glutSpecialFunc

- The new special callback function
- sets the special keyboard callback for the *current window*
  - GLUT_KEY_F1 F1 function key.
  - GLUT_KEY_F2 F2 function key.
  - GLUT_KEY_F3 F3 function key.
  - GLUT_KEY_F4 F4 function key.
  - GLUT_KEY_F5 F5 function key.
  - GLUT_KEY_LEFT Left directional key.
  - GLUT_KEY_UP Up directional key.
  - GLUT_KEY_RIGHT Right directional key.

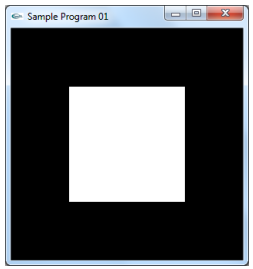
glutSpecialFunc Example

```
void SpecialKeys(int key, int x, int y)
{
    if(key == GLUT_KEY_UP)
        // Some functions
    if(key == GLUT_KEY_DOWN)
        // Some functions
    if(key == GLUT_KEY_LEFT)
    if(key == GLUT_KEY_RIGHT)
        // Some functions
    glutPostRedisplay();
}
```

```
int main(int argc, char *argv[])
{
glutInit(&argc, argv);
glutInitDisplayMode(GLUT_DOUBLE |
GLUT_RGB);
glutInitWindowSize(800, 600);
glutSpecialFunc(SpecialKeys);
glutDisplayFunc(RenderScene);
glutMainLoop();
return 0;
}
```


Example

- Insert a keyboard function to display the simple polygon with following optons
  - Left Increase the width
  - Right Descries the width
  - Up Increase the height
  - Down Descries the height
  - F1 Change the color as red
  - F2 Change the color as blue
  - F3 Change the color white
  - F5 Set the default values



Mouse Callback

- Captures mouse press and release events
- glutMouseFunc(my_mouse);

```
void myMouse(int button, int state, int x, int y)
{
    if (button == GLUT_LEFT_BUTTON)
    {
    …
    }
}
```

Button GLUT LEFT BUTTON, GLUT MIDDLE BUTTON, or GLUT RIGHT BUTTON.

State GLUT_DOWN GLUT_UP

Example

```
void mouse(int button, int state, int x, int y)
{
switch (button)
{
case GLUT_LEFT_BUTTON:
    if (state == GLUT_DOWN)
        glutIdleFunc(spinDisplay);
    break;
case GLUT_MIDDLE_BUTTON:
    if (state == GLUT_DOWN)
        glutIdleFunc(NULL);
    break;
default:    break;
}
}
```

```
int main(int argc, char** argv)
{
glutInit(&argc, argv);
glutInitDisplayMode(GLUT_DOUBLE |
GLUT_RGB);
glutInitWindowSize(250, 250);
glutInitWindowPosition(100, 100);
glutCreateWindow(argv[0]);
init();
glutDisplayFunc(display);
glutReshapeFunc(reshape);
glutMouseFunc(mouse);
glutMainLoop();
return 0;
}
```


Reshape

- **glutReshapeFunc(void (**func)(int w, int h))**
  indicates what action should be taken when the window is resized

- **Example**
  glutReshapeFunc(*resize*);

```
void resize(GLsizei w, GLsizei h)
  {
    if (h==0)
     h=1;          // avoid div by 0
    glViewport(0, 0, w, h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    if (w<=h)
      glOrtho(0, 250, 0, 250*h/w, 1.0, -1.0);
    else
      glOrtho(0, 250*w/h, 0, 250, 1.0, -1.0);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
   }
```


Timer Function

- glutTimerFunc registers a timer callback to be triggered in a specified number of milliseconds
- Wait ms, value pass to call back function
- void glutTimerFunc(unsigned int *msecs, void (*func)(int value), int value);*

```
void TimerFunction(int value)
{

}
```

Main

```
glutTimerFunc(33, TimerFunction, 1);
```


OpenGL Menu

- GLUT supports simple cascading pop-up menus
- designed to let a user select various modes within a program
- pop-up menu facility with an attempt to create a full-featured user interface
- It is illegal to create, destroy, change, add, or remove menu items while a menu are in use

Menu Functions

- **glutCreateMenu**
  - glutCreateMenu creates a new pop-up menu.
  - int glutCreateMenu(void (*func)(int value));
- **glutAddSubMenu**
  - glutAddSubMenu adds a sub-menu trigger to the bottom of the *current menu.*
  - void glutAddSubMenu(char *name, int menu);
- **glutAddMenuEntry**
  - glutAddMenuEntry adds a menu entry to the bottom of the *current menu*
  - void glutAddMenuEntry(char *name, int value);

Menu Functions

- **glutRemoveMenuItem**
  - glutRemoveMenuItem remove the specified menu item.
  - void glutRemoveMenuItem(int entry);
- **glutAttachMenu**
  - attaches a mouse button for the *current window to the identifier of the current menu;*
  - void glutAttachMenu(int button);
- **glutDetachMenu**
  - detaches an attached mouse button from the *current window.*
  - void glutDetachMenu(int button);

Menu Example

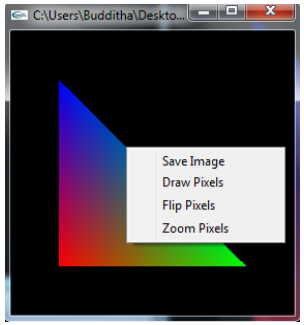
```
glutCreateMenu(ProcessMenu);

  glutAddMenuEntry("Save Image",0);
  glutAddMenuEntry("Draw Pixels",1);
  glutAddMenuEntry("Flip Pixels",2);
  glutAddMenuEntry("Zoom Pixels",3);

glutAttachMenu(GLUT_RIGHT_BUTTON);


void ProcessMenu(int value)
{
  if(value == 0)
  {
     // TASK FOR 0 VALUE
  }

glutPostRedisplay();
}
```



Window Management

- GLUT supports two types of windows: top-level windows and subwindows
- glutCreateWindow
  - glutCreateWindow creates a top-level window.
  - int glutCreateWindow(char *name);
- glutCreateSubWindow
  - glutCreateSubWindow creates a subwindow.
  - int glutCreateSubWindow(int win, int x, int y, int width, int height);
- glutSetWindow, glutGetWindow
  - glutSetWindow sets the current window;
  - glutGetWindow returns the identifier of the current window.
  - void glutSetWindow(int win);
  - int glutGetWindow(void);

Window Management

- glutDestroyWindow
  - glutDestroyWindow destroys the specified window.
  - void glutDestroyWindow(int win);
- glutPostRedisplay
  - glutPostRedisplay marks the current window as needing to be redisplayed.
  - void glutPostRedisplay(void);
- glutSwapBuffers
  - glutSwapBuffers swaps the buffers of the current window if double buffered.
  - void glutSwapBuffers(void);

Window Management

- glutPositionWindow
  - glutPositionWindow requests a change to the position of the current window.
  - void glutPositionWindow(int x, int y);
- glutReshapeWindow
  - glutReshapeWindow requests a change to the size of the current window.
  - void glutReshapeWindow(int width, int height);
- glutFullScreen
  - glutFullScreen requests that the *current window be made full screen.*
  - void glutFullScreen(void);

Window Management

- glutPopWindow, glutPushWindow
  - glutPopWindow and glutPushWindow change the stacking order of the current window relative to its siblings.
  - void glutPopWindow(void);
  - void glutPushWindow(void);
- glutSetWindowTitle, glutSetIconTitle
  - glutSetWindowTitle and glutSetIconTitlechange thewindow or icon title respectively of the current top-level window.
  - void glutSetWindowTitle(char *name);
  - void glutSetIconTitle(char *name);
- glutShowWindow, glutHideWindow, glutIconifyWindow
  - glutShowWindow, glutHideWindow, and glutIconifyWindow change the display status of the current window.
  - void glutShowWindow(void);
  - void glutHideWindow(void);
  - void glutIconifyWindow(void);

Window Management

- **glutSetCursor**
  - glutSetCursor changes the cursor image of the *current window.*
  - void glutSetCursor(int cursor);
  - glutSetCursor(GLUT_CURSOR_NONE);
    - GLUT_CURSOR_RIGHT_ARROW
    - GLUT_CURSOR_LEFT
    - GLUT_CURSOR_INFO
    - GLUT_CURSOR_DESTROY.
    - GLUT_CURSOR_HELP
    - GLUT_CURSOR_CYCLE
    - GLUT_CURSOR_SPRAY
    - GLUT_CURSOR_WAIT
    - GLUT_CURSOR_NONE
    - GLUT_CURSOR_INHERIT