10/1(금) 실습내용(1) : Keyboard Callback 응용I(1)

- [실습과제3-1] 다음과 같은 기능이 수행되도록 코드5-3 수정
  - a 또는 A : 4개의 vertex 좌표가 동일하게 왼쪽으로 0.1씩 이동
  - f 또는 F : 4개의 vertex 좌표가 동일하게 오른쪽으로 0.1씩 이동
  - r 또는 R : 4개의 vertex 좌표가 동일하게 아래쪽으로 0.1씩 이동하고, 빨간색으로 Polygon 칠함
  - v 또는 V : 4개의 vertex 좌표가 동일하게 위쪽으로 0.1씩 이동
  - b 또는 B : 파란색으로 Polygon 칠함

- [조건]
  - 4개의 vertex 좌표가 어디 있든지 상관없이 구동

- [Due Date]
  - 10/07(목) 23:59

♣ 과제의 문제 중 이미 정의된 것 이외에 구현된 내용은 반드시 주석처리를 해서 본인이 어떠한 프로그램을 구현한 것인지 명확히 알려야 합니다.

10/1(금) 실습내용(1) : Keyboard Callback 응용I(2)

코드 5-3

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

void MyInit() {
    glClearColor (1.0, 1.0, 1.0, 1.0);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(-1.0, 1.0, -1.0, 1.0, -1.0, 1.0);
}

void MyDisplay( ) {
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(0.5, 0.5, 0.5);
    glBegin(GL_POLYGON);
        glVertex3f(-0.5, -0.5, 0.0);
        glVertex3f(0.5, -0.5, 0.0);
        glVertex3f(0.5, 0.5, 0.0);
        glVertex3f(-0.5, 0.5, 0.0);
    glEnd( );
    glFlush( );
}

void MyKeyboard(unsigned char key, int x, int y) {
    switch (key) {
        case 'Q': exit(0); break;
        case 'q': exit(0); break;
        case 27: exit(0); break;
     }
     glutPostRedisplay();
}
```

```
void MySpecial(int key, int x, int y) {
     switch (key) {
          case GLUT_KEY_F1 :
               break;
          case GLUT_KEY_LEFT :
               break;
          default :
               break;
     }
}


int main(int argc, char** argv) {
    glutInit(&argc,argv);
    glutInitDisplayMode(GLUT_RGB);
    glutInitWindowSize(300, 300);
    glutInitWindowPosition(0, 0);
    glutCreateWindow("Keyboard Callback");
    MyInit();
    glutDisplayFunc(MyDisplay);
    glutKeyboardFunc(MyKeyboard);
    glutSpecialFunc(MySpecial);

    glutMainLoop();
    return 0;
}
```

10/1(금) 실습내용(2) : Keyboard Callback 응용II

- [실습과제3-2] 다음과 같은 기능이 수행되도록 코드5-3 수정
  - ↑ : 4개의 vertex 좌표가 동일하게 위쪽으로 0.1씩 이동
  - ↓ : 4개의 vertex 좌표가 동일하게 아래쪽으로 0.1씩 이동
  - ← : 4개의 vertex 좌표가 동일하게 왼쪽으로 0.1씩 이동
  - → : 4개의 vertex 좌표가 동일하게 오른쪽으로 0.1씩 이동
  - PageUp : 4개의 vertex 좌표로 이루어진 Polygon이 각 방향으로 0.1씩 연속적으로 확대(Zoom In)
  - PageDown : 4개의 vertex 좌표로 이루어진 Polygon이 각 방향으로 0.1씩 연속적으로 축소(Zoom Out)

- [조건]
  - Polygon의 크기가 가시적으로 최대/최소일 때 더 이상 확대/축소가 되지 않도록 고정
  - 4개의 vertex 좌표가 어디 있든지 상관없이 구동
  - 모든 경우에서 동일하게 4개의 vertex 좌표가 윈도우 영역을 벗어나지 않도록 함

- [Due Date]
  - 10/07(목) 23:59

10/1(금) 실습내용(3) : Mouse Callback 응용

- [실습과제3-3] 다음과 같은 기능이 수행되도록 코드5-3 수정
  - 마우스 왼쪽 버튼을 click
    - 4개의 vertex 좌표가 동일하게 오른쪽으로 0.1씩 연속적으로 이동
  - 마우스 오른쪽 버튼을 click
    - 이동하고 있는 Polygon이 멈춤

- [조건]
  - 4개의 vertex 좌표가 어디 있든지 상관없이 구동
  - 모든 경우에서 동일하게 4개의 vertex 좌표가 윈도우 영역을 벗어나지 말 것

- [Due Date]
  - 10/07(목) 23:59

10/1(금) 실습내용(4) : Menu Callback 응용(1)

- [실습과제3-4] 다음과 같은 기능이 수행되도록 코드5-6 수정
  - 주메뉴 : Draw Sphere, Draw Torus, Draw Teapot, Change Color, Exit
  - Draw Sphere 하부 메뉴 : Small Sphere, Large Sphere
  - Draw Torus 하부 메뉴 : Small Torus , Large Torus
  - Draw Teapot 하부 메뉴 : Small Teapot , Large Teapot
  - Change Color 하부 메뉴 : Red, Green, Blue

- [조건]
  - 각 메뉴를 클릭하였을 때 선택 항목에 맞는 도형 및 색이 그려지도록 함.
  - 각 도형의 Small과 Large는 각각의 크기를 구분할 수 있도록 적절히 설정.

- [Due Date]
  - 10/07(목) 23:59

10/1(금) 실습내용(4) : Menu Callback 응용(2)

코드 5-6

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

GLboolean IsSphere = true;    GLboolean IsSmall = true;

void MyDisplay() {
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(0.0, 0.5, 0.5);

    if ((IsSphere) && (IsSmall))
            glutWireSphere(0.5, 30, 30);
    else if ((IsSphere) && (!IsSmall))
            glutWireSphere(0.7, 30, 30);
    else if ((!IsSphere) && (IsSmall))
            glutWireTorus(0.1, 0.3, 40, 20);
    else   glutWireTorus(0.2, 0.5, 40, 20);
    glFlush( );
}

void MyMainMenu(int entryID) {
    if(entryID == 1)            IsSphere = true;
    else if (entryID == 2)      IsSphere = false;
    else if(entryID == 3)       exit(0);
    glutPostRedisplay();
}

void MySizeMenu(int entryID) {
    if(entryID == 1)            IsSmall = true;
    else if (entryID == 2)      IsSmall = false;
    glutPostRedisplay();
}
```

```
void MyInit() {
    glClearColor (1.0, 1.0, 1.0, 1.0);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity( );

    glOrtho(-1.0, 1.0, -1.0, 1.0, -1.0, 1.0);

    GLint MySizeID = glutCreateMenu(MySizeMenu);
    glutAddMenuEntry("Small", 1);
    glutAddMenuEntry("Large", 2);

    GLint MyMainMenuID = glutCreateMenu(MyMainMenu);
    glutAddMenuEntry("Draw Sphere", 1);
    glutAddMenuEntry("Draw Torus", 2);
    glutAddSubMenu("Size", MySizeID);
    glutAddMenuEntry("Exit", 3);
    glutAttachMenu(GLUT_RIGHT_BUTTON);
}

int main(int argc, char** argv) {
    glutInit(&argc,argv);       glutInitDisplayMode(GLUT_RGB);
    glutInitWindowSize(500, 500);     glutInitWindowPosition(0, 0);
    glutCreateWindow("Menu Callback");
    MyInit();
    glutDisplayFunc(MyDisplay);

    glutMainLoop();
    return 0;
}
```

10/1(금) 실습내용(5) : Idle Callback 응용(1)

- [실습과제3-5] 다음과 같이 수행되도록 코드5-7 수정
  - ↑ : 4개의 vertex 좌표가 동일하게 위쪽으로 0.1씩 이동
  - ↓ : 4개의 vertex 좌표가 동일하게 아래쪽으로 0.1씩 이동
  - ← : 4개의 vertex 좌표가 동일하게 왼쪽으로 0.1씩 이동
  - → : 4개의 vertex 좌표가 동일하게 오른쪽으로 0.1씩 이동

- [조건]
  - 모든 경우에서 동일하게 4개의 vertex 좌표가 윈도우 영역을 벗어나지 말 것

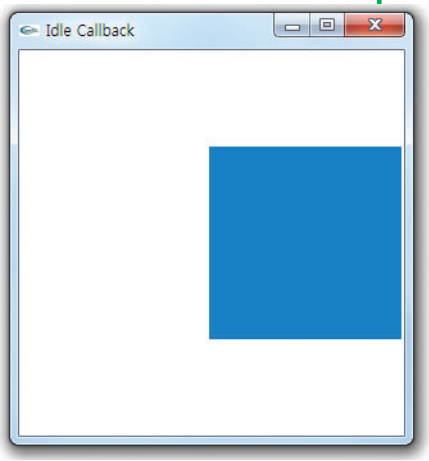
- [Due Date]
  - 10/07(목) 23:59

10/1(금) 실습내용(5) : Idle Callback 응용(2)

코드 5-7

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

GLfloat Delta = 0.0;

void MyDisplay( ) {
    glClear(GL_COLOR_BUFFER_BIT);
    glBegin(GL_POLYGON);
        glColor3f(0.0, 0.5, 0.8);
        glVertex3f(-1.0 + Delta, -0.5, 0.0);
        glVertex3f(0.0 + Delta, -0.5, 0.0);
        glVertex3f(0.0 + Delta, 0.5, 0.0);
        glVertex3f(-1.0 + Delta, 0.5, 0.0);
    glEnd( );
    glutSwapBuffers( );
}

void MyIdle( ) {
    Delta = Delta + 0.001;

    glutPostRedisplay( );
}
```

```
void MyInit() {
    glClearColor (1.0, 1.0, 1.0, 1.0);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();

    glOrtho(-1.0, 1.0, -1.0, 1.0, 1.0, -1.0);
}

int main(int argc, char** argv) {
    glutInit(&argc,argv);
    glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE);
    glutInitWindowSize(300, 300);
    glutInitWindowPosition(0, 0);
    glutCreateWindow("Idle Callback");
    MyInit();
    glutDisplayFunc(MyDisplay);
    glutIdleFunc(MyIdle);

    glutMainLoop( );
    return 0;
}
```



10/1(금) 실습내용(6) : Timer Callback 응용(1)

- [실습과제3-6] 다음과 같이 수행되도록 코드5-8 수정
  - 프로그램이 실행되면 코드5-8에 있는 도형이 오른쪽으로 0.1씩 연속적으로 움직이게 하고, 오른쪽 경계에 부딪히면 도형의 색깔이 변하게 한 후, 다시 도형이 왼쪽으로 0.1씩 연속적으로 움직이는 과정을 반복함.
  - 마우스 왼쪽 키를 누르면 움직이는 도형이 멈춤.

- [조건]
  - 모든 경우에서 동일하게 4개의 vertex 좌표가 윈도우 영역을 벗어나지 말 것
  - 도형의 색깔은 지정하지 않으니 자유롭게 설정함.

- [Due Date]
  - 10/07(목) 23:59

10/1(금) 실습내용(6) : Timer Callback 응용(2)

코드 5-8

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

GLfloat Delta = 0.0;

void MyDisplay( ) {
    glClear(GL_COLOR_BUFFER_BIT);
    glBegin(GL_POLYGON);
        glColor3f(0.0, 0.5, 0.8);
        glVertex3f(-1.0 + Delta, -0.5, 0.0);
        glVertex3f(0.0 + Delta, -0.5, 0.0);
        glVertex3f(0.0 + Delta, 0.5, 0.0);
        glVertex3f(-1.0 + Delta, 0.5, 0.0);
    glEnd( );
    glutSwapBuffers( );
}
void MyTimer(int value) {
    Delta = Delta + 0.001;

    glutPostRedisplay( );
    glutTimerFunc(40, MyTimer, 1);
      // 40msec 후에 호출

}
```

```
void MyInit() {
    glClearColor (1.0, 1.0, 1.0, 1.0);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();

    glOrtho(-1.0, 1.0, -1.0, 1.0, 1.0, -1.0);
}

int main(int argc, char** argv) {
    glutInit(&argc,argv);
    glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE);
    glutInitWindowSize(300, 300);
    glutInitWindowPosition(0, 0);
    glutCreateWindow("Timer Callback");
    MyInit();
    glutDisplayFunc(MyDisplay);
    glutTimerFunc(40, MyTimer, 1);

    glutMainLoop( );
    return 0;
}
```