#include <GL/glut.h>

#include <cmath>

#include <iostream>

void display();

void reshape(int width, int height);

void keyboard(unsigned char key, int x, int y);

void createPopupMenus();

void processMenuOptions(int option);

void drawCircleExtrusion();

void drawSquareExtrusion();

void draw2DMesh();

bool drawCircle = false;

bool drawSquare = false;

bool drawMesh = false;

float shapeColor[3] = { 1.0f, 1.0f, 1.0f };

void display() {

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glColor3fv(shapeColor);

if (drawCircle) {

drawCircleExtrusion();

}

else if (drawSquare) {

drawSquareExtrusion();

}

else if (drawMesh) {

draw2DMesh();

}

glutSwapBuffers();

}

void reshape(int width, int height) {

glViewport(0, 0, width, height);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glOrtho(-400, 400, -300, 300, -1, 1);

glMatrixMode(GL\_MODELVIEW);

}

void keyboard(unsigned char key, int x, int y) {

switch (key) {

case 'r':

shapeColor[0] = 1.0f;

shapeColor[1] = 0.0f;

shapeColor[2] = 0.0f;

break;

case 'g':

shapeColor[0] = 0.0f;

shapeColor[1] = 1.0f;

shapeColor[2] = 0.0f;

break;

case 'b':

shapeColor[0] = 0.0f;

shapeColor[1] = 0.0f;

shapeColor[2] = 1.0f;

break;

case 'w':

shapeColor[0] = 1.0f;

shapeColor[1] = 1.0f;

shapeColor[2] = 1.0f;

break;

case 27:

exit(0);

break;

}

glutPostRedisplay();

}

void createPopupMenus() {

int menu = glutCreateMenu(processMenuOptions);

glutAddMenuEntry("Parallelepiped from Circle", 1);

glutAddMenuEntry("Parallelepiped from Square", 2);

glutAddMenuEntry("2D Mesh", 3);

glutAttachMenu(GLUT\_RIGHT\_BUTTON);

}

void processMenuOptions(int option) {

switch (option) {

case 1:

drawCircle = true;

drawSquare = false;

drawMesh = false;

shapeColor[0] = 1.0f;

shapeColor[1] = 0.0f;

shapeColor[2] = 0.0f;

glutPostRedisplay();

break;

case 2:

drawCircle = false;

drawSquare = true;

drawMesh = false;

shapeColor[0] = 0.0f;

shapeColor[1] = 1.0f;

shapeColor[2] = 0.0f;

glutPostRedisplay();

break;

case 3:

drawCircle = false;

drawSquare = false;

drawMesh = true;

shapeColor[0] = 0.0f;

shapeColor[1] = 0.0f;

shapeColor[2] = 1.0f;

glutPostRedisplay();

break;

default:

break;

}

}

void drawCircleExtrusion() {

glClearColor(1, 1, 1, 1);

glClear(GL\_COLOR\_BUFFER\_BIT);

glColor3fv(shapeColor);

glPointSize(5);

for (int i = 0; i < 50; i++) {

glBegin(GL\_LINE\_LOOP);

for (int j = 0; j < 360; j += 10) {

float angle = j \* 3.14159 / 180;

glVertex2f(100 + 50 \* cos(angle) + i, 100 + 50 \* sin(angle) + i);

}

glEnd();

}

glFlush();

}

void drawSquareExtrusion() {

glClearColor(1, 1, 1, 1);

glClear(GL\_COLOR\_BUFFER\_BIT);

glColor3fv(shapeColor);

glPointSize(5);

for (int i = 0; i < 50; i++) {

glBegin(GL\_LINE\_LOOP);

glVertex2i(50 + i, 50 + i);

glVertex2i(150 + i, 50 + i);

glVertex2i(150 + i, 150 + i);

glVertex2i(50 + i, 150 + i);

glEnd();

}

glFlush();

}

void draw2DMesh() {

int rows = 10;

int cols = 10;

float x\_min = -100;

float x\_max = 100;

float y\_min = -100;

float y\_max = 100;

float dx = (x\_max - x\_min) / cols;

float dy = (y\_max - y\_min) / rows;

glColor3fv(shapeColor);

glBegin(GL\_LINES);

for (int i = 0; i <= rows; i++) {

glVertex2f(x\_min, y\_min + i \* dy);

glVertex2f(x\_max, y\_min + i \* dy);

}

for (int j = 0; j <= cols; j++) {

glVertex2f(x\_min + j \* dx, y\_min);

glVertex2f(x\_min + j \* dx, y\_max);

}

glEnd();

glFlush();

}

int main(int argc, char\*\* argv) {

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB | GLUT\_DEPTH);

glutInitWindowSize(800, 600);

glutCreateWindow("OpenGL Interactive Drawing Application");

glutDisplayFunc(display);

glutReshapeFunc(reshape);

glutKeyboardFunc(keyboard);

createPopupMenus();

glClearColor(1.0f, 1.0f, 1.0f, 1.0f);

glEnable(GL\_DEPTH\_TEST);

glutMainLoop();

return 0;

}