```java
// Solução possível para o efólio-A

import java.util.Random;

import javax.media.opengl.*;
import javax.media.opengl.glu.GLU;
import com.sun.opengl.util.GLUT;

@SuppressWarnings("serial")
//adicionada a classe MouseMotionListener como classe base de J1_0_Point.java
public class Cadeira extends J1_4_Line {
  GLU glu = new GLU(); // interface para a biblioteca GLU

  private float [] cor; // para conter o rgb da cor da cadeira
  Random gerador; // geração de valores aleatórios
  int contador;
  float angulo;
  float eixoX, eixoY, eixoZ;
  float moveX, moveY;

  public Cadeira() {
 // utiliza o construtor da classe mãe e ativa duplo buffering
    capabilities.setDoubleBuffered(true);
    cor = new float [] {1.0f, 0.2f, 0.0f}; // define cor inicial
    gerador = new Random(); // inicializa gerador aleatório
    contador = 0; // Inicializa demais atributos
    angulo = 0;
    eixoX = 0.0f;
    eixoY = 1.0f;
    eixoZ = 0.0f;
    moveX = 0.0f;
    moveY = 0.0f;
    gl.glColor3f(cor[0], cor[1], cor[2]); // define a cor inicial
    gl.glEnable (GL.GL_DEPTH_TEST);
  }

// Invocado sempre para refrescar a cena
  public void display(GLAutoDrawable drawable) {
    gl.glClearColor (1.0f, 1.0f, 1.0f, 0.1f); // Define o branco como cor de fundo

    viewPort1(); // Visualiza cena em quatro viewports diferentes
    viewPort2();
    viewPort3();
    viewPort4();

    // torna refrescamento de desenho mais lento
    try {
        Thread.sleep(20);
    } catch (Exception ignore) {
        }
   }

  // Escreve as palavras no centro da cena
  void escrevePalavras() {
        //gl.glColor3f(cor[0], cor[2], 0.0f);
        gl.glRasterPos3f(WIDTH/2, HEIGHT/2, 0); // posição final
        glut.glutBitmapCharacter(GLUT.BITMAP_HELVETICA_18, 'B');
```

```
        glut.glutBitmapString(GLUT.BITMAP_HELVETICA_18, "itmap");

        gl.glPushMatrix();
            gl.glTranslatef(WIDTH/2-20, HEIGHT/2-30, 0); // posição final
            gl.glScalef(0.2f, 0.2f, 0.2f);
            glut.glutStrokeCharacter(GLUT.STROKE_ROMAN, 'S');
            glut.glutStrokeString(GLUT.STROKE_ROMAN, "troke");
        gl.glPopMatrix();
  }


  // 1º Viewport - canto superior esquerdo
  void viewPort1() {
    int w = WIDTH, h = HEIGHT;

// executa limpeza de buffer da cor, é feito apenas no 1º viewport!
// Se não for feito, borra o fundo...
    gl.glClear (GL.GL_COLOR_BUFFER_BIT|GL.GL_DEPTH_BUFFER_BIT);
    gl.glViewport(0, h/2, w/2, h/2);
    gl.glMatrixMode(GL.GL_PROJECTION);
    gl.glLoadIdentity();

    gl.glOrtho(300, WIDTH-300, 300, HEIGHT-300, -200.0f, 200.0f);
    desenhaCena1();
 }

 // 2º Viewport - canto superior direito
  void viewPort2() {
     int w = WIDTH, h = HEIGHT;
     gl.glViewport(w/2, h/2, w/2, h/2);
     gl.glMatrixMode(GL.GL_PROJECTION);
     gl.glLoadIdentity();
     gl.glOrtho(0, WIDTH, 0, HEIGHT, -200.0f, 200.0f);
     desenhaCena1();

// É necessário mandar limpar o depth buffer, senão, nada aparece no viewport!
     gl.glClear (GL.GL_DEPTH_BUFFER_BIT);
 }

  // 3º Viewport - canto inferior direito
 void viewPort3() {
     int w = WIDTH, h = HEIGHT;
     gl.glViewport(w/2, 0, w/2, h/2);
     gl.glMatrixMode(GL.GL_PROJECTION);
     gl.glLoadIdentity();
     gl.glOrtho(200, WIDTH, 200, HEIGHT, -500.0f, 500.0f);
     desenhaCena2();

// É necessário mandar limpar o depth buffer, senão, nada aparece no viewport!
     gl.glClear (GL.GL_DEPTH_BUFFER_BIT);
 }

//4º Viewport - canto inferior esquerdo
void viewPort4() {
     int w = WIDTH, h = HEIGHT;
     gl.glViewport(0, 0, w/2, h/2);
     gl.glMatrixMode(GL.GL_PROJECTION);
```

```
     gl.glLoadIdentity();
     gl.glOrtho(200, WIDTH, 200, HEIGHT, -500.0f, 500.0f);
     desenhaCena3();

//É necessário mandar limpar o depth buffer, senão, nada aparece no viewport!
     gl.glClear (GL.GL_DEPTH_BUFFER_BIT);
}

//Método que compõem a cena
  void desenhaCena1() {
        escrevePalavras(); // Escreve texto no centro

// A rotação na cadeira é incrementada de forma gradual
        if (angulo > 360) angulo = 0;
        else ++angulo;

        if (contador >= 200) { // muda a orientação da rotação de vez em quando
            eixoX = 0.1f * gerador.nextInt(10); // eixos arbitrários
            eixoY = 0.1f * gerador.nextInt(10);
            eixoZ = 0.1f * gerador.nextInt(10);

//A cada execução atualiza a cor de forma aleatória
            cor[0] = gerador.nextInt(255)/255.0f;
            cor[1] = gerador.nextInt(255)/255.0f;
            cor[2] = gerador.nextInt(255)/255.0f;

            gl.glColor3f(cor[0], cor[1], cor[2]); // bem como a cor
            contador = 0;
        }

        ++ contador; // Incrementa o contador

        gl.glTranslatef(x, y, 0.0f); // move para a posição do rato
        gl.glPushMatrix();
          gl.glRotatef(angulo, eixoX, eixoY, eixoZ);
          desenhaCadeira();
        gl.glPopMatrix();
     }

  void desenhaCena2() { // Cadeira desloca-se da esquerda para direita
// O zooming na cadeira é incrementada de forma gradual
      if (moveX > WIDTH) moveX = 0;

      ++moveX;

      gl.glPushMatrix();
          gl.glTranslatef(moveX, HEIGHT/2, 0.0f);
          gl.glRotatef(180.0f, 1.0f, 1.0f, 1.0f);
          desenhaCadeira();
      gl.glPopMatrix();
  }

  void desenhaCena3() {// Cadeira desloca-se de baixo para cima
    // O zooming na cadeira é incrementada de forma gradual
          if (moveY > HEIGHT) moveY = 0;

          ++moveY;
```

```
            gl.glPushMatrix();
                gl.glTranslatef(WIDTH/2, moveY, 0.0f);
                gl.glRotatef(180.0f, 1.0f, 1.0f, 1.0f);
                desenhaCadeira();
            gl.glPopMatrix();
    }
// Monta a cadeira
    void desenhaCadeira() {
            gl.glPushMatrix();
               gl.glRotatef(90.0f, 1.0f, 0.0f, 0.0f);
     // Desenha assento
               desenhaCaixa(60, 60, 10);
     // Desenha o encosto para as costas
               gl.glPushMatrix();
                   gl.glRotatef(90.0f, 1.0f, 0.0f, 0.0f);
                   desenhaCaixa(60, 100, 10);
               gl.glPopMatrix();
     // Desenha as 4 pernas
               gl.glPushMatrix();
                   gl.glTranslatef(-5.0f, -5.0f, -100.0f);
                      desenhaCaixa(10, 10, 100);
                   gl.glPopMatrix();
                   gl.glPushMatrix();
                      gl.glTranslatef(50.0f, -5.0f, -100.f);
                      desenhaCaixa(10, 10, 100);
                   gl.glPopMatrix();
                   gl.glPushMatrix();
                      gl.glTranslatef(5.0f, 50.0f, -100.f);
                      desenhaCaixa(10, 10, 100);
                   gl.glPopMatrix();
                   gl.glPushMatrix();
                      gl.glTranslatef(50.0f, 50.0f, -100.0f);
                      desenhaCaixa(10, 10, 100);
                   gl.glPopMatrix();
          // Desenha um bule em arame perto do tampo da cadeira
                   gl.glPushMatrix();
                      gl.glRotatef(90.0f, 1.0f, 0.0f, 0.0f);
                      gl.glTranslatef(20.0f, 10.0f, -50.0f);
                      gl.glColor3f(1.0f, 0.0f, 0.0f);
                      glut.glutWireTeapot(15.0f);
                      gl.glColor3f(cor[0], cor[1], cor[2]);
                   gl.glPopMatrix();
              gl.glPopMatrix();

          gl.glFlush();
    }

// Desenha uma caixa que ajuda a compor a cadeira (feita de várias caixas)
     @SuppressWarnings("static-access")
     void desenhaCaixa(float w, float h, float d) {
          gl.glBegin(gl.GL_POLYGON);
              gl.glVertex3f(0, h, 0);
              gl.glVertex3f(w, h, 0);
              gl.glVertex3f(w, 0, 0);
              gl.glVertex3f(0, 0, 0);
          gl.glEnd();
```

```
        gl.glBegin(gl.GL_POLYGON);
            gl.glVertex3f(0, h, d);
            gl.glVertex3f(w, h, d);
            gl.glVertex3f(w, h, 0);
            gl.glVertex3f(0, h, 0);
        gl.glEnd();
        gl.glBegin(gl.GL_POLYGON);
            gl.glVertex3f(0, 0, 0);
            gl.glVertex3f(w, 0, 0);
            gl.glVertex3f(w, 0, d);
            gl.glVertex3f(0, 0, d);
        gl.glEnd();
        gl.glBegin(gl.GL_POLYGON);
            gl.glVertex3f(0, h, 0);
            gl.glVertex3f(0, 0, 0);
            gl.glVertex3f(0, 0, d);
            gl.glVertex3f(0, h, d);
        gl.glEnd();
        gl.glBegin(gl.GL_POLYGON);
            gl.glVertex3f(w, 0, 0);
            gl.glVertex3f(w, h, 0);
            gl.glVertex3f(w, h, d);
            gl.glVertex3f(w, 0, d);
        gl.glEnd();
        gl.glBegin(gl.GL_POLYGON);
            gl.glVertex3f(0, h, d);
            gl.glVertex3f(w, h, d);
            gl.glVertex3f(w, 0, d);
            gl.glVertex3f(0, 0, d);
        gl.glEnd();
    }

    public static void main(String[] args) {
        Cadeira efolio = new Cadeira();

        efolio.setTitle("Efólio-A");
        efolio.setSize(WIDTH, HEIGHT);
        efolio.setVisible(true);
    }
}
```