SPRAWOZDANIE

Zajęcia: Grafika komputerowa

Prowadzący: prof. dr hab. Vasyl Martsenyuk

Laboratorium: 3

Data: 09.03.2022

Temat: "Modelowanie hierarciczne w grafice 2D";

Przemysław Holisz

Informatyka I stopień, stacjonarne,

4 semestr, Gr.4/2b

1. Polecenie:

Opracowa´c scene¸ hierarchiczna¸ zgodnie z obrazem u¨zywaja¸c zamiast k´o l wieloka¸ty obracaja¸ce sie¸ (animacja!) wed lug wariantu. Opracowanie powinno by´c w jed nym z je¸zyk´ow: Java lub JavaScript, na dwa sposoby: (a) u¨zywaja¸c hierarchije¸ funkcje (spos´ob subroutinowy) (b) tworza¸c graf sceny (spos´ob obiektowy).

2. Wprowadzane dane:

a) SubroutineHierarhy

```
rotatingPolygon(g2, 100, -1.02, -0.05);
rotatingPolygon(g2, 100, 1.04, -0.98);
rotatingPolygon(g2, 80, -1.379, 1.40);
rotatingPolygon(g2, 80, -3.13, 2.23);
rotatingPolygon(g2, 60, 0.9, 2.05);
rotatingPolygon(g2, 60, 2.12, 1.45);

S_Line(g2,1,1.05, 0, -0.5, Color.RED);
S_Line(g2,0.85,0.95, -2.65, 1.90,Color.RED);
S_Line(g2,0.6,0.70, 2.5, 2.5,Color.RED);

Triangle(g2 ,0.5,0.5,0,-2,Color.BLUE);
Triangle(g2 ,0.35,0.35,-2.25,0.75,new Color(199, 21, 133));
Triangle(g2 ,0.25,0.25, 1.5,1, Color.GREEN);
```

b) SceneGraph

```
private TransformedObject Triangle_1;
private TransformedObject Triangle_2;
private TransformedObject Triangle_3;
private TransformedObject Line_1;
private TransformedObject Line_2;
private TransformedObject Line_3;
private TransformedObject Polygon_1;
private TransformedObject Polygon_2;
private TransformedObject Polygon_3;
private TransformedObject Polygon_4;
private TransformedObject Polygon_5;
private TransformedObject Polygon_6;
```

Wykorzystane komendy:

a) SubroutineHierarchy

```
 private void updateFrame() {
            frameNumber++;
            // TODO: If other updates are needed for the next frame, do them
here.
      }
            // TODO: Define methods for drawing objects in the scene.
      private void S_Line (Graphics2D g2,double skala_x , double skala_y, double
translate_x ,double translate_y, Color color  )
      {
            AffineTransform saveTransform = g2.getTransform();
            g2.scale(skala_x,skala_y);
            Line(g2, translate_x, translate_y);
            g2.setTransform(saveTransform);
      }
      private void Line (Graphics2D g2, double translate_x ,double translate_y  )
      {
            g2.setColor(Color.RED);
            g2.translate(translate_x,translate_y);
      g2.rotate(-Math.PI/8);
      g2.scale(2.3, 0.15);
            filledRect(g2);
      }
    private void Triangle (Graphics2D g2,double skala_x , double skala_y, double
translate_x ,double translate_y ,Color color )
    {
      AffineTransform saveTransform = g2.getTransform();
      g2.setColor(color);
      g2.translate(translate_x,translate_y);
      g2.scale(skala_x,skala_y);
      g2.fillPolygon(new int[] {0,1,-1}, new int[] {3,0,0},3 );
      g2.setTransform(saveTransform);
    }

      private void rotatingPolygon(Graphics2D g2,double r, double translate_x
,double translate_y) // polygon
      {

               AffineTransform saveTransform = g2.getTransform();
            Color saveColor = g2.getColor();
            g2.setTransform(saveTransform);
            g2.setStroke(new BasicStroke(2));

            int n=13;
            double t=0,
            k=(Math.PI*2)/n;

            int[] x1 = new int[n];
            int[] y1 = new int[n];

            for (int i=0;i<n;i++)
            {
                x1[i]= (int) (r*Math.sin(t));
                y1[i]= (int) (r*Math.cos(t));
```

```
            t+=k;
        }

        Polygon polygon = new Polygon(x1,y1,n);
        g2.translate(translate_x,translate_y);
        g2.setColor( Color.black );
        g2.rotate( Math.toRadians( frameNumber*0.75 ));
        g2.scale( 0.005, 0.005 );

        for(int i=0;i<n;i++)
        {
              g2.drawLine( x1[i],y1[i],0,0 );
        }
        g2.draw(polygon);
        g2.setColor(saveColor);
        g2.setTransform(saveTransform);
}
```

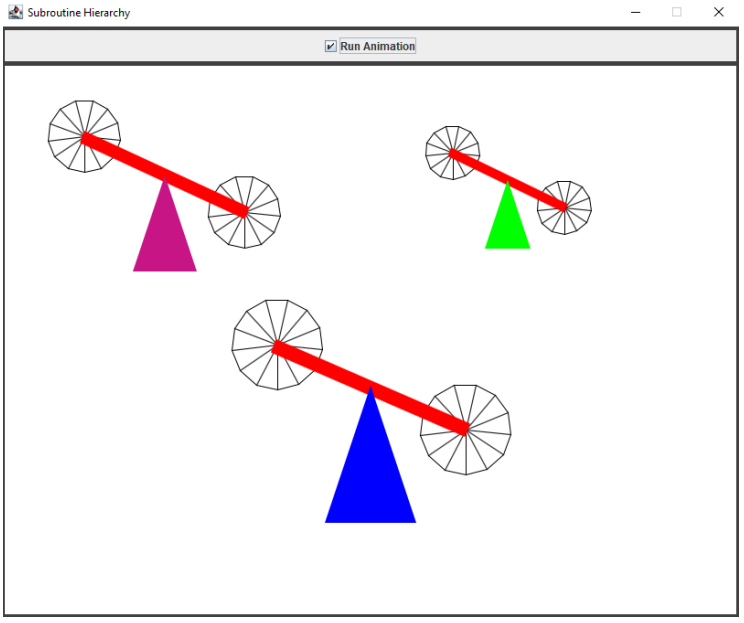
b) SceneGraph

```
private void createWorld() {

        world = new CompoundObject();  // Root node for the scene graph.

        // TODO: Create objects and add them to the scene graph.
        // Root node for the scene graph.
        Triangle_1 = new TransformedObject(filledTriangle);
        Triangle_2 = new TransformedObject(filledTriangle);
        Triangle_3 = new TransformedObject(filledTriangle);
        Line_1          = new TransformedObject(filledRect);
        Line_2          = new TransformedObject(filledRect);
        Line_3          = new TransformedObject(filledRect);
        Polygon_1  = new TransformedObject(F_Polygon);
        Polygon_2  = new TransformedObject(F_Polygon);
        Polygon_3  = new TransformedObject(F_Polygon);
        Polygon_4  = new TransformedObject(F_Polygon);
        Polygon_5  = new TransformedObject(F_Polygon);
        Polygon_6  = new TransformedObject(F_Polygon);

        Triangle_1.setScale(0.5, 1.2).setTranslation(0, -
2).setColor(Color.BLUE);
        Triangle_2.setScale(0.5, 1).setTranslation(-2.25, 0.5).setColor(new
Color(199, 21, 133));
        Triangle_3.setScale(0.5, 0.8).setTranslation(1.5,
1).setColor(Color.GREEN);
        Line_1.setRotation(-22.5).setScale(2, 0.1).setTranslation(0, -
0.8).setColor(Color.RED);
        Line_2.setRotation(-22.5).setScale(1.8, 0.1).setTranslation(-2.2,
1.50).setColor(Color.RED);
        Line_3.setRotation(-22.5).setScale(1.5, 0.08).setTranslation(1.5,
1.8).setColor(Color.RED);
        Polygon_1.setScale(0.3, 0.3).setTranslation(-0.889, -0.42);
        Polygon_2.setScale(0.3, 0.3).setTranslation(0.899, -1.189);
        Polygon_3.setScale(0.25, 0.25).setTranslation(-3, 1.825);
        Polygon_4.setScale(0.25, 0.25).setTranslation(-1.4, 1.18);
        Polygon_5.setScale(0.2, 0.2).setTranslation(0.83, 2.07);
        Polygon_6.setScale(0.2, 0.2).setTranslation(2.16, 1.52);

        world.add(Polygon_1);
        world.add(Polygon_2);
        world.add(Polygon_3);
        world.add(Polygon_4);
        world.add(Polygon_5);
        world.add(Polygon_6);
        world.add(Line_1);
        world.add(Line_2);
        world.add(Line_3);
        world.add(Triangle_1);
        world.add(Triangle_2);
        world.add(Triangle_3);

} // end createWorld()


/**
```

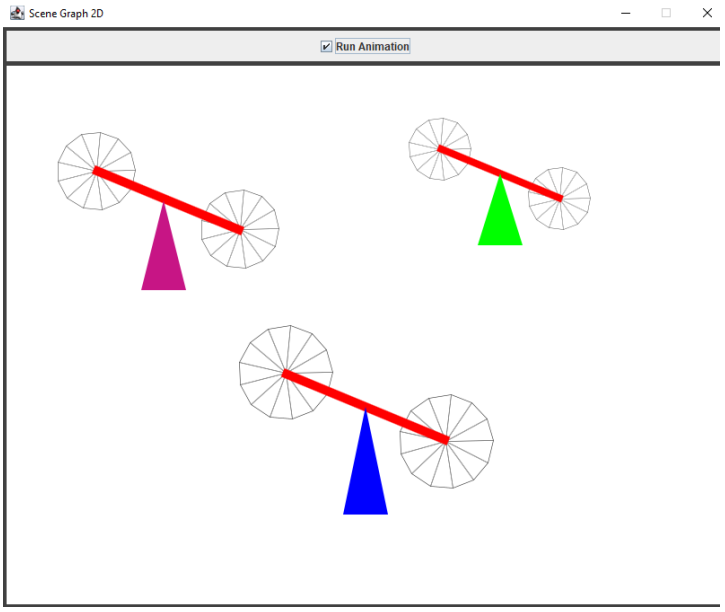
```
        * This method is called just before each frame is drawn.  It updates the
modeling
        * transformations of the objects in the scene that are animated.
        */
       public void updateFrame() {
              frameNumber++;

              // TODO: Update state in preparation for drawing the next frame.

              Polygon_1.setRotation(frameNumber*0.75);
              Polygon_2.setRotation(frameNumber*0.75);
              Polygon_3.setRotation(frameNumber*0.75);
              Polygon_4.setRotation(frameNumber*0.75);
              Polygon_5.setRotation(frameNumber*0.75);
              Polygon_6.setRotation(frameNumber*0.75);

       }
```

3. Wynik działania:

a) Hieracrhy

b) 

c) Graph



4. Wnioski:

Na podstawie otrzymanego wyniku można stwierdzić, że w języku Java możemy stworzyć animację dostępnych kształtów, obrazków w dowolny sposób.