Øving 1G LAG-grafikkdel

1 Innstaller JOGL på egen PC. Se Installasjonsveiledning

2 På forelesningen ble det utdelt utskrift av kode som skal skrives inn i f.eks. TextPad og fylles ut til å bli kjørbare programmer. Utskriften kan hentes på mitt kontor, eller dere kan bruke den etterfølgende koden. Dere skal tegne ut en trekant, en firkant og en sirkel i et vindu på skjermen.

Mvh
Jan

```
/*
 * GrafikkEksempelOv1JOGL.java  JHN 2011-01-18
 *
 * Filen inneholder to klasser:
 * Vindu: Et vindu med en tegning
 *
 * GrafikkEksempelOv1JOGL: Inneholder main()-metode som viser fram vinduet med tegningen
 *
 * Tegningen er gitt som en egen klasse TegningOv1_1JOGL som ligger på
 * en egen fil kalt TegningOv1_1JOGL.java. Det som tegnes ut er to trekanter,
 * en firkant og en sirkel. JOGL er javabindingen som benyttes mot OPENGL.
 */

import java.awt.*;  // klassene Color og Graphics
import javax.swing.*; // klassene JFrame og JPanel
import java.util.*;
import javax.media.opengl.*;  //JOGL klasser
import javax.media.opengl.glu.*;  glu klasser


class Vindu extends JFrame {
  public Vindu(String tittel) {
    setTitle(tittel);
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    TegningOv1_1JOGL tegningen = new TegningOv1_1JOGL(400, 400);
    add(tegningen);
    pack();
  }
}

/*Klassen som inneholder main*/

class GrafikkEksOv1JOGL {
  public static void main(String[] args) {
    Vindu etVindu = new Vindu("V2005 Øving 1: Enkel grafikk");
    etVindu.setVisible(true);
  }
}
```

```
/*

TegningOv1_1JOGL.java   JHN 2011-01-18

Draws one quad, two triangles and one circle.  JOGL binding towards OpenGL

*/

// Java  classes

   import java.awt..*;
   import javax.swing.*; // klassene JFrame og JPanel
   import java.util.*;
//JOGL klasser
   import javax.media.opengl.*;  //JOGL klasser
   import javax.media.opengl.glu.*;

public class TegningOv1_1JOGL extends JPanel implements GLEventListener{

          /* interfacet GLEventListener innholder følgende 4 metoder som må implemeenteres:
                - display(GLDrawable drawable)
                   Called by the drawable to initiate OpenGL rendering by the client
                - displayChanged(GLDrawable drawable, boolean modeChanged, boolean deviceChanged)
                   Called by the drawable when the display mode or the display device associated with the GLDrawable has
                   changed.
                - init(GLDrawable drawable)
                   Called by the drawable immediately after the OpenGL context is initialized.
                - reshape(GLDrawable drawable, int x, int y, int width, int height)
                   Called by the drawable during the first repaint after the component has been resized.
     */

    private GLCanvas canvas;
    private float angle;
   private  GLU glu = new GLU();

      public TegningOv1_1JOGL(int width, int hight)  {
        super();

           GLCapabilities capabilities = new GLCapabilities();
          capabilities.setHardwareAccelerated(true);      //We want hardware acceleration
          capabilities.setDoubleBuffered(true);          //And double buffering
          canvas = new GLCanvas(capabilities);
          canvas.addGLEventListener(this);
          this.add(canvas);
          this.setSize(width,hight);
          canvas.setSize(width,hight); //We want the JPanel and the GLCanvas to have the same size
          canvas.setVisible(true);             //This is somehow necessary
     }


    public void init(GLAutoDrawable glDrawable) {
        GL gl = glDrawable.getGL();                  //Get the GL object from glDrawable
        gl.glClearColor(1.0f, 1.0f, 1.0f, 1.0f); // Sets the background color to white

        gl.glMatrixMode(GL.GL_PROJECTION);       // Select The Projection Matrix
```

```
        gl.glLoadIdentity();                                // Reset the view matrix to the identity matrix
        glu.gluPerspective(45.0,1.25,2.0,9.0); // Spesifize the projection matrix (fov, w/h, near plane, far plane)

        gl.glMatrixMode(GL.GL_MODELVIEW);
    }


    public void reshape(GLAutoDrawable glDrawable, int i, int i1, int i2, int i3) {
            // Has to be implementet due to the GLEventListener interface
     }


/* Draw two triangles and one quad  */

       public void drawGLScene(GLAutoDrawable glDrawable)  {
          GL gl = glDrawable.getGL();
          gl.glClear(GL.GL_COLOR_BUFFER_BIT | GL.GL_DEPTH_BUFFER_BIT);  //Clear The Screen
             //        and The Depth Buffer
          gl.glLoadIdentity();                          // Reset The View matrix
          gl.glTranslatef(-1.5f,0.0f,-8.0f);            // Move Left 1.5 Units and into The Screen 8 units
          gl.glColor3f(?.??,?.??,?.??);                 // Set the Color to Red
                                                        // Start drawing a triangle
                                                         // Top
                                                        // Bottom Left
                                                        // Bottom Right
                                                        // Finished Drawing The Triangle

                                                        // Move Right 3 Units
                                                        // Set the Color to Green
                                                        // Start drawing a Quad
                                                        // Top Left
                                                        // Top Right
                                                        // Bottom Right
                                                        // Bottom Left
                                                        // Done drawing the Quad


    }// drawGLScene()
```

```
/* Tegner en  sirkel */

 public void drawGLScene2(GLAutoDrawable glDrawable)  {
          GL gl = glDrawable.getGL();
          gl.glLoadIdentity();                    // Reset The View matrix
          gl.glTranslatef(-0.1f,-1.0f,-7.0f);       // Move Left 0.1, down 1.0 units and into the screen 7 units
           final double PI = 3.1415926535898;      // Initiate constant PI
          int circle_points = 100;              // Initiate circle_points ( number of points to construct the circle)
          gl.glColor3f(?.??,?.??,?.??);                 // Set Color to Blue
          gl.glBegin(GL.????????);          // Draw a lines between circle  points using
           double angle = 0.0;                          // Initiate angle

           for(int i = 0; i < circle_points; i++){                // for loop
              angle = 2 * PI * i/circle_points;                   // calculate new angle
             gl.glVertex2f((float)Math.cos(angle), (float)Math.sin(angle));   // calculate vertex points on the circle
           }

          gl.glEnd();        // Done drawing the circle
     }

    /** void display() Draw to the canvas. */
   /* Purely a Java thing. Simple calls drawGLScene once GL is initialized */

     public void display(GLAutoDrawable glDrawable) {
           GL gl = glDrawable.getGL();
          ??????????(glDrawable);                    // Calls drawGLScene
          ??????????(glDrawable);                    // Calls drawGLScene2
          glDrawable.swapBuffers();//ligger ikke i glut slik som i c, her er det viktig for ikke å få flimmer!!

          gl.glFlush();      // Tvinger tidligere buffrede OpenGL kommandoer til å utføres med en gang.

      }

     public void displayChanged(GLAutoDrawable glDrawable, boolean b, boolean b1) {
          //  Must be present due to the GLEventListener interface
     }
 } // TegningOv1_1JOGL
```