Ludwig-Maximilians-Universität München
LFE Medieninformatik
Prof. Dr. Heinrich Hußmann
Henri Palleis

Übungsblatt 2: Einstieg in JOGL

Abgabe:

Dieses Übungsblatt ist einzeln zu lösen. Die Lösung ist bis **Montag, den 7. Mai 2012, 12:00 Uhr s.t.** über UniWorx (https://uniworx.ifi.lmu.de/) abzugeben.
Es werden nur die Formate PDF und Plain-Text (UTF-8) akzeptiert. Benennen Sie die Dateien nach dem Schema <Übungsblatt>-<Aufgabe>.<extension>, d.h. die Lösung der ersten Aufgabe geben Sie in einer Datei 1-1.txt oder 1-1.pdf ab. Packen Sie alle Dateien in eine ZIP-Datei und laden Sie diese bei UniWorx hoch. Wenn Sie Formatierungsvorgaben nicht einhalten, können ihre Abgaben nicht korrigiert werden.

Inhalt:

Ziel dieses Übungsblattes ist, ein erstes einfaches Programm mit JOGL zu erstellen und dabei falls notwendig die eigenen Java-Kenntnisse aufzufrischen.

Aufgabe 1: Die Window-to-Viewport-Transformation

Führen Sie sich für diese Aufgabe zunächst den Code aus Anhang 1 vor Augen.

i. Sie haben in der Vorlesung verschiedene Koordinatensysteme kennengelernt. Welche Koordinaten werden im Code-Beispiel verwendet, um die Linien zu definieren, die das Quadrat bilden?
ii. Im Code-Beispiel werden zwei zweidimensionale rechteckige Bereiche definiert. Mit welcher Code-Zeile und unter Verwendung welcher Koordinaten wird ein sogenanntes *Window* erstellt, das einen rechteckigen Ausschnitt definiert, durch den wir auf die virtuelle Welt blicken? Mit welcher Code-Zeile und unter Verwendung welcher Koordinaten wird im Code-Beispiel der sogenannte *Viewport* erstellt, der diesen Ausschnitt auf dem Bildschirm anzeigt?
iii. Beschreiben sie basierend auf den geometrischen Transformationen, die wir in Übungsblatt 1 kennengelernt haben, die Schritte, die notwendig sind, um das *Window* auf den *Viewport* abzubilden. Geben sie den Rechenweg an.
iv. Was passiert, wenn *Window* und *Viewport* nicht die gleichen Seitenverhältnisse haben?

Aufgabe 2: Einstieg in JOGL

Laden Sie sich zunächst eine aktuelle Version von JOGL für ihr Betriebssystem herunter. Detaillierte Installationshinweise finden Sie unter:
http://jogamp.org/wiki/index.php/Downloading_and_installing_JOGL

Eine Anleitung zum Einbinden der von JOGL benötigten Bibliotheken in ihr Java-Projekt (in IDEs wie Eclipse oder NetBeans, aber auch für die Arbeit mit einfachen Texteditoren) finden Sie

unter:
http://jogamp.org/wiki/index.php/Setting_up_a_JogAmp_project_in_your_favorite_IDE

Anforderungen an die Anwendung:

- Es soll ein Fenster in der Größe 500px * 500px mit Swing erstellt werden
- Das Schließen des Fensters beendet die Anwendung
- Das Fenster enthält einen GLCanvas
- Die Hintergrundfarbe des GLCanvas ist Grau (50%)
- Die positiven Achsenabschnitte eines rechthändigen, dreidimensionalen Koordinatensystems werden in den Farben Rot (X-Achse), Gelb (Y-Achse) und Grün (Z-Achse) gezeichnet (jeweils von 0 bis 1)
- Die Seitenverhältnisse bleiben auch bei Veränderungen der Fenstergröße erhalten

Aufgabe 3: Projektion und Kamera

Aufbauend auf der Anwendung aus Aufgabe 1 kommen nun folgende Anforderungen hinzu:

- Die XZ-Ebene enthält ein graues (30%) quadratisches Gitter mit der Seitenlänge 10, dessen Mittelpunkt mit dem Ursprung zusammenfällt. Das Gitter enthält genau 100 gleichgroße und quadratische Zellen.
- Die Szene enthält ein durch Linien angedeutetes Haus
- Es wird folgende orthografische Projektion verwendet:
  - Vertikale Ausschnittsflächen links bei -10 und rechts bei 10
  - Horizontale Ausschnittsflächen unten bei -10 und oben bei 10
  - Ausschnittsflächen der Tiefe bei 0 (nah) und 100 (entfernt)
- Die Szene enthält eine virtuelle Kamera, die sich anfänglich an der Position (10|10|10) befindet und auf den Ursprung gerichtet ist
- Durch Drücken der Tasten 1-4 kann zwischen verschiedenen Ansichten gewechselt werden (siehe Abbildung 1):
  - 1: Vorderansicht (Projektionsfläche parallel zur XY-Ebene)
  - 2: Seitenansicht(Projektionsfläche parallel zur YZ- Ebene)
  - 3: Ansicht von Oben (Projektionsfläche parallel zur XZ- Ebene)
  - 4: Anfängliche Ansicht
- Mit den Pfeiltasten nach oben und unten kann die Position der Kamera verändert werden. Auf welche Achse sich die Steuerung auswirkt, soll durch Drücken der Tasten x,y und z ausgewählt werden können.

Hinweise: Die Methode repaint() der GLCanvas-Klasse kann verwendet werden, um die display(GLAutoDrawable drawable)-Methode erneut aufzurufen. Zudem kann einem GLCanvas-

Objekt ein KeyListener hinzugefügt werden, um die Tastatureingaben abzufangen.

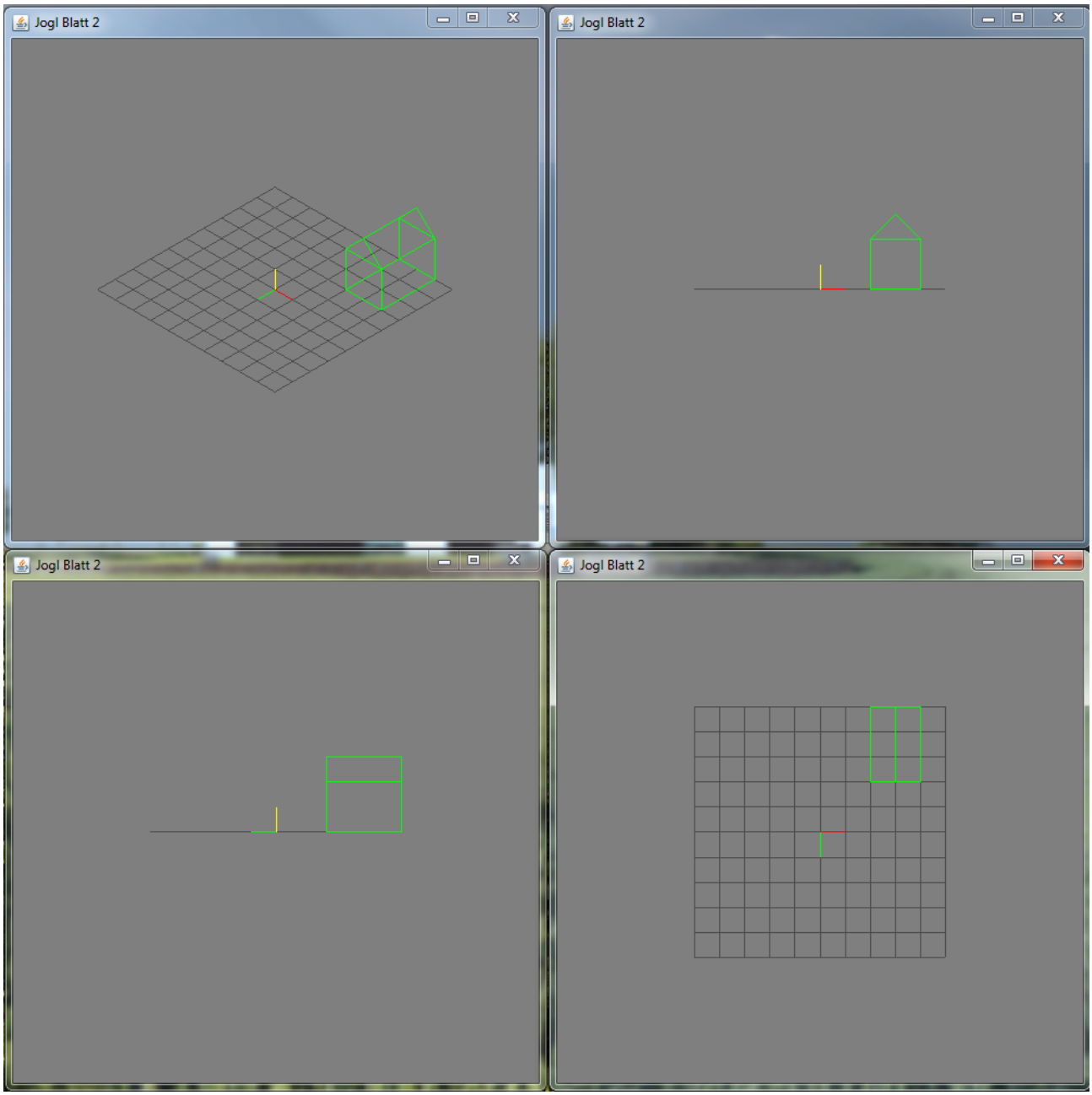

Abbildung 1 Vier geöffnete Instanzen des Programms aus Aufgabe 3, die jeweils eine der Ansichten zeigen.

Anhang 1:

```
import javax.swing.*;
import javax.media.opengl.*;
import javax.media.opengl.awt.GLCanvas;

public class JOGLExample1 extends JFrame {
	GLCanvas canvas;

	public JOGLExample1() {
		GLProfile glp = GLProfile.getDefault();
		GLCapabilities caps = new GLCapabilities(glp);
		canvas = new GLCanvas(caps);

		canvas.addGLEventListener(new SceneView());

		add(canvas);

		setTitle("Jogl Example");
		setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		setSize(300,300);
		setVisible(true);
	}

	class SceneView implements GLEventListener {

		public void init(GLAutoDrawable drawable) {
			GL2 gl = drawable.getGL().getGL2();
			gl.glClearColor(0, 0, 0, 0); // black background

			gl.glMatrixMode(GL2.GL_PROJECTION);
			gl.glLoadIdentity();
			gl.glOrthof(-3, 3, -3, 3, 0, 100);
		}

		public void display(GLAutoDrawable drawable) {
			GL2 gl = drawable.getGL().getGL2();
			gl.glClear(GL2.GL_COLOR_BUFFER_BIT); // clear background
			gl.glColor3d(1, 0, 0); //draw in red

			gl.glBegin(GL2.GL_LINE_LOOP); //draw a square
			gl.glVertex3i(-1, -1, -1);
			gl.glVertex3i(1, -1, -1);
			gl.glVertex3i(1, 1, -1);
			gl.glVertex3i(-1, 1, -1);
			gl.glEnd();

		}
```

```
            public void reshape(GLAutoDrawable drawable, int x, int y, int w, int h) {
            }

            public void dispose(GLAutoDrawable drawable) {
            }

        }

        public static void main(String args[]) {
            new JOGLExample1();
        }
}
```

Viel Erfolg.