Ludwig-Maximilians-Universität München
LFE Medieninformatik
Prof. Dr. Andreas Butz
Hendrik Richter, Raphael Wimmer

Übungsblatt 3: QT, Debugging

Abgabe:
Dieses Übungsblatt ist einzeln zu lösen. Die Lösung ist bis **Dienstag,** den **18. Mai 2010, 12:00 Uhr s.t.** über UniWorx (http://www.pst.ifi.lmu.de/uniworx) abzugeben.
Es werden nur die Formate PDF und Plain-Text (UTF-8) akzeptiert. Erstellen Sie für jede Aufgabe ein Unterverzeichnis nach dem Schema <Übungsblatt>-<Aufgabe>, d.h. die Lösung der ersten Aufgabe kommt in ein Verzeichnis *3-1/*. Packen Sie alle Dateien in eine ZIP-Datei und laden Sie diese bei UniWorx hoch. Wenn Sie Formatierungsvorgaben nicht einhalten, werden bis zu zwei Punkte abgezogen. Lösungen müssen zumindest im CIP-Pool fehlerfrei kompilieren und laufen. Bitte geben Sie nur Quellcode ab, keine kompilierten Dateien.

Inhalt:
Ziel dieses Übungsblattes ist, erste Erfahrungen mit QT, Slots und Signals, Widgets und Layouts zu sammeln. Desweiteren sollen Sie ein einfaches OpenGL-Widget einbinden. Mit diesem werden wir in den nächsten Übungsblättern arbeiten. Und zur Wiederholung des Vorlesungsstoffes gibt es auch noch eine Übungsaufgabe zu Projektion und Rastern. Es können maximal 20 Punkte erreicht werden.

Vorbereitungen:
Laden Sie den QT Creator von http://qt.nokia.com/downloads herunter und installieren Sie ihn. Der QT Creator ist eine IDE für C++ mit guter QT-Unterstützung. Wir werden ihn im weiteren Übungsbetrieb verwenden. Stellen Sie sicher, dass ein C++-Compiler installiert ist (unter Windows wird die mingw-Toolchain mitinstalliert). Machen Sie sich mit der IDE vertraut. **Lesen Sie einführende Tutorials zu QT und machen Sie sich mit der API-Dokumentation vertraut.**

Aufgabe 1: Einfache Verflachung (6 Punkte)

Gegeben sei ein Dreieck p mit

$$p_1 = \begin{pmatrix} -3 \\ -2 \\ -4 \end{pmatrix}, p_2 = \begin{pmatrix} 6 \\ -4 \\ -8 \end{pmatrix}, p_3 = \begin{pmatrix} -0{,}5 \\ 1 \\ -2 \end{pmatrix}$$

Dieses soll durch eine perspektivische Projektion auf eine Bildebene mit z = -1 abgebildet werden.

- Geben Sie die Eckpunkte des Dreiecks in Bildkoordinaten an.
- Rastern Sie das Dreieck in ein Bitmap der Größe 9x9 Pixel unter Verwendung des Bresenham-Algorithmus.
- Geben Sie eine PDF-Datei 3-1.pdf mit den Eckpunkten und dem gerasterten Bild ab.

Tipp: (-1,1) liegt auf Bildpixel (0,0) und (1,-1) liegt auf Bildpixel (8,8)

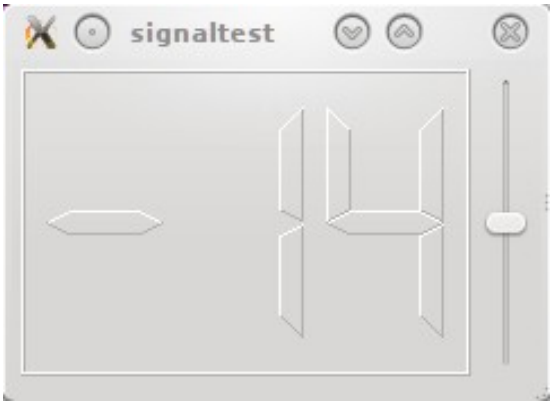
Ludwig-Maximilians-Universität München
LFE Medieninformatik
Prof. Dr. Andreas Butz
Hendrik Richter, Raphael Wimmer

<u>Aufgabe 2:</u> Völker, hört die QT-Signale (8 Punkte)

Implementieren Sie ein QT-Programm *signaltest* mit folgenden Eigenschaften:

- Alle UI-Komponenten sind von Hand geschrieben. Verwenden Sie nicht den QT Designer.
- Das Hauptfenster enthält auf der rechten Seite ein vertikales Slider-Widget. Der Slider ist in einer mittleren Position.
- Auf der linken Seite befindet sich ein QLCDNumber-Widget, das mit “0” initialisiert wird.
- Wenn man den Slider verschiebt, wird die Zahl im QLCDNumber-Widget erhöht oder erniedrigt. Sobald man den Slider loslässt, springt dieser wieder auf die Mittelstellung zurück. Die angezeigte Zahl bleibt stehen und kann erneut erhöht oder erniedrigt werden.
- Wenn die angezeigte Zahl nach dem Loslassen des Sliders eine zweistellige Schnapszahl ist, d.h. nur aus gleichen Ziffern besteht ('22', '44'), dann beendet sich das Programm.

Implementieren Sie die Kommunikaton zwischen Slider und Ziffernanzeige mit Signals und Slots. Erzeugen Sie dafür eine Klasse *Model*, die Slider-Änderungen empfängt, die Ziffernanzeige updatet. und ggf. das Programm beendet.
Implementieren Sie eine Klasse ResetSlider, die von QSlider erbt und beim Loslassen des Sliders diesen wieder in die Mittelstellung bewegt.
Geben Sie Source- und Headerdateien sowie die Projektdatei *signaltest.pro* in einem Unterverzeichnis 3-2/ ab.



Ludwig-Maximilians-Universität München
LFE Medieninformatik
Prof. Dr. Andreas Butz
Hendrik Richter, Raphael Wimmer

Aufgabe 3: 0D-OpenGL (6 Punkte)

In dieser Aufgabe haben Sie zum ersten Mal mit OpenGL zu tun – wenn auch noch nicht mit 3D-Grafik.
Implementieren Sie ein QT-Programm *gltest* mit folgenden Eigenschaften:

- Alle UI-Komponenten sind von Hand geschrieben. Verwenden Sie nicht den QT Designer.
- Das Hauptfenster enthält auf der rechten Seite drei vertikale Slider-Widgets mit einem Wertebereich von je 0-255. Jeder Slider steht für eine der drei Grundfarben.
- Auf der linken Seite befindet sich ein MyGLWidget, das von QGLWidget erbt.
- Durch Verschieben der Slider lässt sich die Hintergrundfarbe des MyGLWidgets anpassen.

Überschreiben Sie dazu die Methoden initializeGL(), resizeGL(int width, int height) und paintGL() von QGLWidget und fügen Sie jeweils

```
glClearColor(red, green, blue, 1.0); // red, green, blue are of type GLclampf
glClear(GL_COLOR_BUFFER_BIT);
```

ein. Diese Zeilen setzen die Hintergrundfarbe der OpenGL-Anzeige. Verwenden Sie Signals und Slots zum Setzen der Hintergrundfarbe.

Tipp: Um OpenGL-Unterstützung einzubinden, muss in der Projektdatei "`QT += opengl`" stehen.

Geben Sie Source- und Headerdateien sowie die Projektdatei *gltest.pro* in einem Unterverzeichnis 3-3/ ab.

Viel Erfolg.