Thema, Ziele: Weitere GUI-Programme mit Qt.

Aufgabe 1: Qt Counter: Layout Manager

Auf dem Skriptserver, im Unterverzeichnis "QtExamples\demo31_QtCounter1" finden Sie ein Qt-Programm mit dem ein Zähler realisiert wurde.

Probieren Sie dieses Programm als Erstes aus.

Ändern Sie das Programm nun so, dass das Label und die zwei Buttons *nebeneinander* zu stehen kommen: links der *Decrement*-Button, in der Mitte das Label mit dem Zählerwert und rechts der *Increment*-Button.

Die beiden Buttons sollen nun so definiert werden, dass sie auch mit den Tasten <*Alt*>+*I* und <*Alt*>+*D* (sogenannte "Shortcut's") bedient werden können. – Anleitung: Schreiben Sie ein &-Zeichen beim Buttontext vor den Shortcut-Buchstaben. Also "`&Increment`" und "`&Decrement`". Probieren Sie andere Buchstaben.

Aufgabe 2: Qt Counter: Reset-Button

Verwenden Sie als Basis erneut das Programm von "QtExamples\demo31_QtCounter1".

Erweitern Sie dieses Programm so, dass mit Hilfe eines zusätzlichen Reset-Buttons der Zählerstand auf 0 zurückgesetzt werden kann.

Frage: Ist das möglich, ohne die Klasse "Counter" zu ändern?

Aufgabe 3: Qt Counter: zweiter Zähler

Verwenden Sie als Basis erneut das Programm von "QtExamples\demo31_QtCounter1".

Erweitern Sie dieses Programm so, dass zusätzlich noch ein zweiter Zählerstand angezeigt wird und verändert werden kann. Dazu soll unterhalb der bestehenden GUI-Elemente vom ersten Zählerstand, noch ein zweiter Zähler (ein Label und zwei PushButtons) hinzugefügt werden. (Die beiden Zähler sollen unabhängig voneinander funktionieren.)

Aufgabe 4: Qt Counter: zwei Zähler, GridLayout

Die Darstellung des Programms von Aufgabe 3 soll dahingehend geändert werden, dass die GUI-Elemente eines Zählers nebeneinander zu liegen kommen.

Spalte 1 enthält also das Label und die zwei Buttons für Zählerwert 1, Spalte 2 diejenigen für Zählerwert 2.

Verwenden Sie fürs Layout die Klasse "QGridLayout". Studieren Sie dazu diese Klasse.

Aufgabe 5: Qt Counter: Eigene Widget-Klasse

Verwenden Sie erneut als Basis das Programm von "QtExamples\demo31_QtCounter1"

Unschön bei dieser Lösung ist, dass der Aufbau des GUI's im Hauptprogramm 'main()' erfolgt.

Eine bessere Lösung besteht darin, statt wie jetzt (im "main()") ein QDialog-Objekt zu instanziieren und danach diesem Objekt seine Child-Widgets (ein Label und zwei PushButtons) hinzuzufügen, eine eigene Klasse (z.B. "MyDialog") zu erstellen, die von QDialog abgeleitet ist.

Das 'main()' wird dadurch erheblich vereinfacht und sieht dann etwa so aus:

```
int main(int argc, char *argv[])
{
    QApplication app(argc, argv);

    MyDialog windowTop(0);

    windowTop.show();
    return app.exec();
}
```

Die Initialisierung erfolgt nun im Konstruktor der Klasse "MyDialog".

Ändern Sie das gegebene Programm ("QtExamples\demo31_QtCounter1") entsprechend.