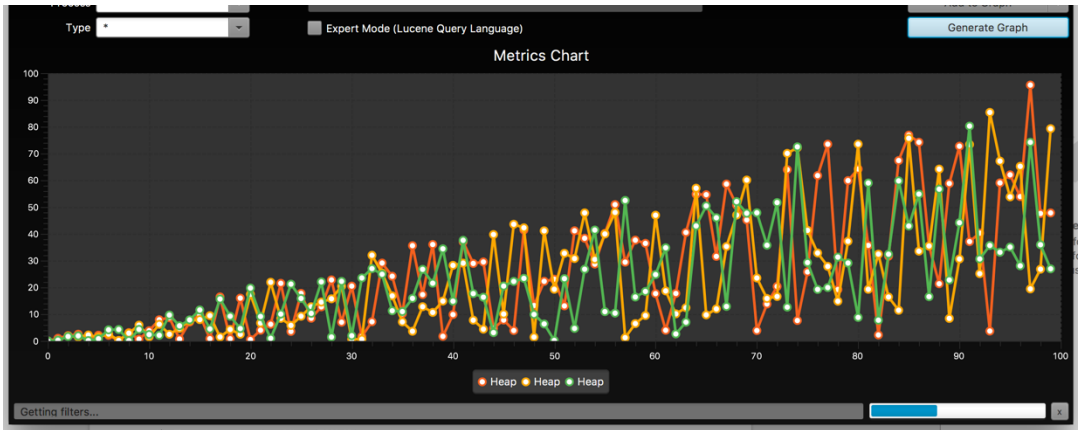
GUI Vorlesung 2019

Übung 7 - Parallelität bei grafischen Oberflächen



Beschreibung

Ziel der Übung ist es, eine (simuliert) langlaufende Aktion außerhalb des UI-Threads auszuführen. Dem Benutzer soll dabei ein Fortschritt visualisiert werden.

Aufgabe 1

Erweitern Sie den *ChartController* um eine langsame Methode „*simulateLongRunningTask*“. Benutzen Sie dazu einfach *Thread.sleep(…)*.
Simulieren Sie eine langlaufende Aktion, indem Sie die neue Methode vor der Generierung der Diagrammdaten (*chartModel.generateData()*) aufrufen.

Wie verhält sich die Oberfläche im manuellen Test?

Aufgabe 2

Verlagern Sie die nun langsame Aktion und die anschließende Generierung der Daten in einen neuen Thread.
Achten Sie dabei darauf wann Änderungen an den GUI-Daten und der Oberfläche erfolgen und in welchem Thread dies passieren muss. Benutzen Sie ggf. den Rückkanal aus der Vorlesung.

Aufgabe 3

Erweitern Sie die Chart-Ansicht der letzten Übung um eine Fortschrittsanzeige, bestehend aus einer Meldungszeile (Text) und einem Fortschrittsbalken. Implementieren Sie einen JavaFX-Service und binden Sie die Meldungszeile und den Fortschrittsbalken an.

Gehen Sie dabei wie folgt vor:

1.) Implementieren sie einen JavaFX-Task der nach einer Wartezeit als Ergebnis die zufälligen Diagrammdaten liefert. Verschieben Sie dazu die Methode *generateData* vom *ChartModel* in den *Task* und passen Sie sie an (siehe Listing 1).
2.) Implementieren Sie einen passenden JavaFX-Service, der den Task aus Schritt 1 verwendet.
3.) Verwenden Sie den Service im Controller. Beim Klick auf „Generate Graph“ soll der Service aufgerufen werden (*restart*).
4.) Verwenden sie *<service>.setOnSucceeded(…)* im Controller um das Ergebnis zu übernehmen. Setzen Sie das Ergebnis entweder direkt in das Diagramm (UI-Element) oder in das *ChartModel*.
5.) Passen Sie die Implementierung im Task an und ersetzen Sie das lange Warten durch mehrere kleine Abschnitte. Aktualisieren Sie die Meldung (*updateMessage*) und den Fortschritt (*updateProgress*) nach jedem Abschnitt.
6.) Erweitern sie die Oberfläche um die UI-Elemente für die Fortschrittsanzeige.
7.) Binden Sie die Meldungszeile und den Fortschrittsbalken an die Eigenschaften des Service.

Listing 1:

```
public class GenerateDataTask extends
Task<ObservableList<XYChart.Series<Number, Number>>> {

…

private ObservableList<XYChart.Series<Number, Number>> generateData() {
        ObservableList<XYChart.Series<Number, Number>> seriesList =
FXCollections.observableArrayList();

        final XYChart.Series<Number, Number> series = new XYChart.Series<>();
        series.setName(seriesName);
        for (int i = 0; i < 100; i++) {
            series.getData().add(new XYChart.Data<>(i, Math.random() * i));
        }
        seriesList.addAll(series);
        return seriesList;
    }

…
}
```

Aufgabe 4 (optional)

Implementieren Sie das Abbrechen der Aktion.