

Fortgeschrittene Programmiertechniken

Prof. Dr. J. Pauli, Dipl.-Inf. Fabian Bürger, B. Sc. N. Gur, P. Zdankin

Übungsblatt 1

Seite 1 von 3

In der Vorlesung haben Sie das Architekturmuster MVC kennengelernt. Dieses eignet sich gut für die Implementierung von interaktiven Softwaresystemen mit Benutzeroberflächen. Dabei wird das System in die drei Einheiten (Model, View und Controller) unterteilt. Diese Aufteilung bringt unter anderem den Vorteil der Wiederverwendbarkeit von Komponenten.

Während der Bearbeitung der Übungen soll ein PC-HardwareShop erstellt werden. Jedes Übungsblatt beschäftigt sich mit einem Teil des Shops. Achten Sie also darauf, dass Sie wiederverwendbare Komponenten schreiben.

Aufgabe 1

Model, View und Controller

Zunächst soll das Grundgerüst des Programms mithilfe des MVC-Architekturmusters implementiert werden. Verwenden Sie dazu JavaFX. Durch das darin enthaltene Observer-Pattern und durch die lambda Ausdrücke, die seit Java8 möglich sind, ist das MVC Muster in nur wenigen Zeilen Code realisierbar.

In der Übung werden wir auf alle nötigen Komponenten wie `StringProperty`, `ListView`, `TableView`, `TableColumn`, `HBox`, `BorderPane`, `ToolBar`, `Button`, `TextField` und `Label` eingehen.

- a) Erzeugen Sie die Klassen *Product*, *ProductList* und *Order*, die die gleichnamigen Interfaces aus dem Paket *fpt.com* implementieren.
Die Klasse *Product* soll als Repräsentation eines Produkts dienen und beinhaltet den Namen, die ID, den Preis und die vorhandene Stückzahl eines Produkts.
Die Klasse *Order* enthält zu den Waren auch die bestellte Anzahl und deren Preis.
Bei den Klassen *Order*, *ProductList* kann es sich um eine Ableitung der Klasse *java.util.ArrayList* handeln, die ausschließlich Waren enthält.
Achten Sie darauf bereits hier Property-Objekte zu erstellen, da Sie diese für die GUI benötigen und sich dadurch viel Arbeit sparen (z.B. *SimpleStringProperty*).
- b) Schreiben Sie die Klasse *ModelShop*, welche die Daten von der GUI hält.
Diese Klasse soll von *ModifiableObservableListBase* ableiten und durch Nutzung eines Delegate (*ProductList*) die nötigen Funktionen des Interfaces *ProductList* realisieren.
- c) Die Klasse *ViewShop* dient als Benutzerschnittstelle für den Shop.
Es sollen die eingetragenen Produkte angezeigt werden.
- d) Schreiben Sie die Klasse *ControllerShop*, welche für die Verarbeitung der Benutzereingaben des Händlers zuständig sein soll. Diese Klasse bindet auch die nötigen Objekte aus dem *ModelShop* an die Elemente der View.
- e) Es soll möglich sein neue Produkte einzufügen und bestehende Produkte zu löschen. Erweitern Sie ihre Klasse *ViewShop* um die dafür nötigen GUI-Elemente.
Implementieren Sie außerdem die nötige Funktionalität.
Prüfen Sie die zu machenden Eingaben so früh wie möglich, um Fehler zur Laufzeit zu vermeiden.
Defekte Programme bzw. Fehler bei der Abnahme führen zu Punktabzug.

Fortgeschrittene Programmiertechniken

Prof. Dr. J. Pauli, Dipl.-Inf. Fabian Bürger, B. Sc. N. Gur, P. Zdankin

Übungsblatt 1

Seite 2 von 3

Aufgabe 2

Bestellformular ViewCustomer

Schreiben Sie hier nur die grafische Oberfläche ohne Funktionalität.

(Die Implementierung der Klasse *ControllerCustomer* wird in späteren Aufgaben folgen.)

- Die Klasse *ViewCustomer* soll alle Produkte so anzeigen, dass der Name, der Preis und dessen Anzahl sichtbar sind. Außerdem soll die Möglichkeit bestehen, eine Stückzahl für die Bestellung zu wählen.
- Die Anzeige soll zudem eine Historie der getätigten Bestellungen anzeigen.

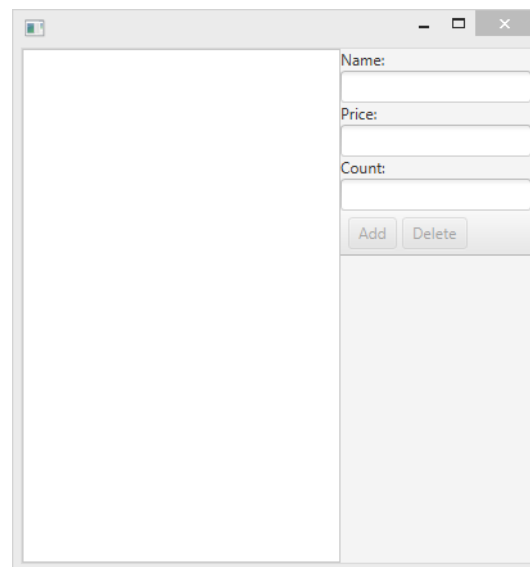


Abbildung 1: Beispiel Produkt einfügen

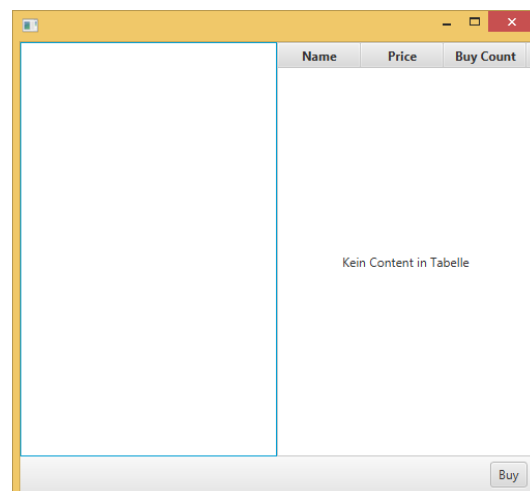


Abbildung 2: Bestellanzeige

Fortgeschrittene Programmiertechniken

Prof. Dr. J. Pauli, Dipl.-Inf. Fabian Bürger, B. Sc. N. Gur, P. Zdankin

Übungsblatt 1

Seite 3 von 3

HINWEIS: Ihr Programm muss nur die beschriebene Funktionalität erfüllen. Die grafischen Nutzeroberflächen in den Abbildungen dienen nur als Beispiele. Sie müssen nicht im Detail nachgebaut werden und können von der Abbildung abweichen.