Selbststudiums-Aufgabe 11

Layers, MVC

1. Selbststudium zu Layers

Lesen Sie in den Unterrichts-Unterlagen das Kapitel 9.3 über *Schichtenarchitekturen*. Bei Fragen dazu kontaktieren Sie Ihren Dozenten.

a) Überlegen Sie sich verschiedene Vor- und Nachteile von *strikten* und *nicht-strikten Schichtenarchitekturen* (Aufgabe Seite 2 oben). Denken Sie zur Inspiration an verschiedene Anwendungen von Schichtenarchitekturen, wie z.B.
   - klassische Informationsverarbeitungssysteme (S. 2 unten),
   - eine Datei-System-Schicht über einer Schicht, die einzelne Diskblöcke lesen und schreiben kann,
   - Spiel-Programme mit intensiver Graphik-Nutzung, die schnell und plattformunabhängig sein soll,
   - Überlegen Sie auch, was es braucht, um (Unit-)Tests für einzelne Schichten machen zu können.

b) Unten auf Seite 2 finden Sie drei Beispiele klassischer Schichtenarchitekturen, wie sie bei Informationssystemen häufig sind. Zeichnen Sie jeweils für zwei nebeneinander dargestellte Beispiele ein, welche Schichten des einen Beispiels welchen Schichten (oder welcher Schicht) des jeweils anderen entsprechen. Betrachten Sie dazu die den Schichten zugewiesene Verantwortung.

2. Micro MVC

Als ein Beispiel für die Benutzung von *MVC im Kleinen* finden Sie im Projekt *ch.fhnw.swa.mvc.adder* die einfache Applikation, die in den Unterrichtsunterlagen im Kapitel 9.4 gezeigt wird. Lösen Sie damit die *Lernaufgabe JavaFX-Properties* auf Seite 5 der Unterrichtsunterlagen.

3. Makro MVC

Der *Adder* aus Aufgabe 2 ist in einer einzigen Klasse programmiert. Dazu kommt ein *.fxml*-File, in welchem das Layout der Controls deklarativ programmiert ist.

Teilen Sie den Code aus der einen Klasse auf insgesamt vier Klassen auf: *Model*, *View*, *Controller* – jeweils mit der Verantwortung gemäss dem MVC-Muster. In der vierten Klasse (*Main*) soll nur der Code verbleiben, den es zum Starten des Systems und zum Zusammensetzen von Objekten der anderen drei Klassen braucht.

Das *.fxml*-File wurde mit einem Hilfsprogramm (JavaFX Scene Builder) erzeugt. Sie können es aber auch einfach als *.xml*-Datei lesen und bearbeiten. Im Tag *AnchorPane* finden Sie u.a. das Attribut *fx:controller*. Hier steht der Name derjenigen Klasse, die Variablen zu den Controls (mit *@fxml*-Annotation) deklariert. Den Wert dieses Attributs müssen Sie anpassen, wenn diese Variablen in einer anderen als der Klasse *Main* stehen. Welche ist das bei Ihnen?

4. BidirectionalBindings mit JavaFX – Extra-Aufgabe für Interessierte

Bindings mit *JavaFX*-Properties sehen nach einem einfachen und schlanken Mechanismus aus, um Verbindungen zwischen *Controls* und *Werten* herzustellen. Dank des einheitlich überall eingebauten – und zum Teil gut versteckten – Observer Patterns fällt es leicht, auch komplexe Beziehungen durch verbundener Properties darzustellen. Der im Unterricht betrachtete *Adder* ist hierfür ein einfaches Beispiel.

Um zu verstehen, welche Datenstruktur dabei im Hintergrund aufgebaut wird (logische Sicht), analysieren Sie den Code, der durch folgende Anweisung ausgeführt wird:

```
@FXML
private TextField op1;
private IntegerProperty value1 = new SimpleIntegerProperty();
...
Bindings.bindBidirectional(op1.textProperty(), value1, new NumberStringConverter());
```

Zeichnen Sie ein oder mehrere Diagramme, so dass aus Ihrer Dokumentation ersichtlich wird, welche Objekte wie verbunden werden und welche Aufrufe es zwischen diesen Objekten gibt, wenn

```
value1.set(3);
```

ausgeführt wird.

Abgabe bis 3. Juni 2019