Zürcher Hochschule
für Angewandte Wissenschaften

School of Engineering

InIT Institut für Angewandte Informations-technologie

Wissenssicherung

V2 – GUI-Architekturen

Lernziel

Sie sind in der Lage, eine einfache GUI-Architektur nach dem MVVM Pattern zu entwerfen und umzusetzen.

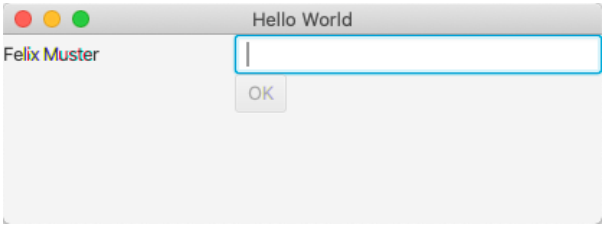
Einleitung

Es soll mit Hilfe des MVVM-Patterns eine JavaFX Anwendung erstellt werden. Die Lernaufgabe V2.2 (JavaFX-Anwendung nach MVP) soll als Vorlage dienen.

Aufgabe V2.3: Erstellen Sie eine JavaFX-Anwendung nach dem MVVM Pattern.

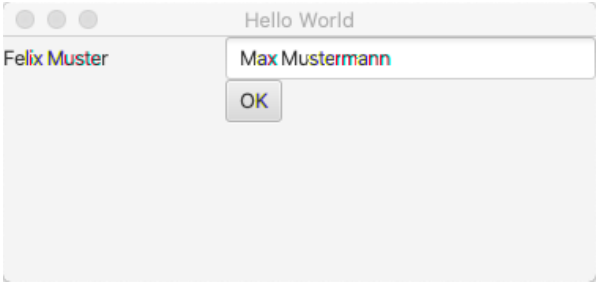
Die Anforderungen sind identisch mit der Lernaufgabe V2.2:

1. Beim Starten der Anwendung wird links in einem Label der Initialwert aus dem Domain Model angezeigt.



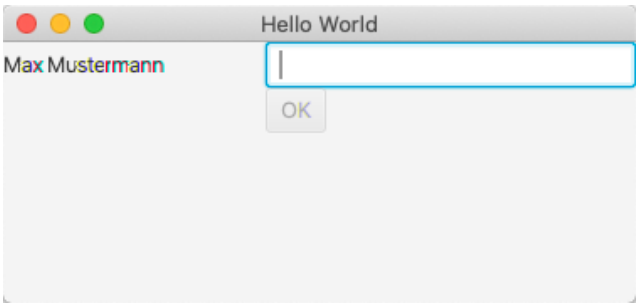
   Sofern kein Eingabewert im Textfeld steht ist der Button OK nicht aktiv.

2. Der Benutzer kann einen neuen Namen eingeben.

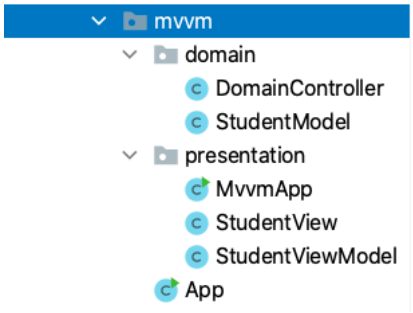


3. Die Eingabe wird mittels [Enter] oder mittels Klick auf [OK] über den View und das ViewModel im Domain Model gespeichert.

4. Das Domain Model informiert über einen `PropertyChangeEvent` das ViewModel über die Änderung. Das ViewModel aktualisiert seine eigenen Properties und damit über die Data Bindings die View. Das Textfeld wird geleert.



Empfohlene Struktur:



Vorgehen

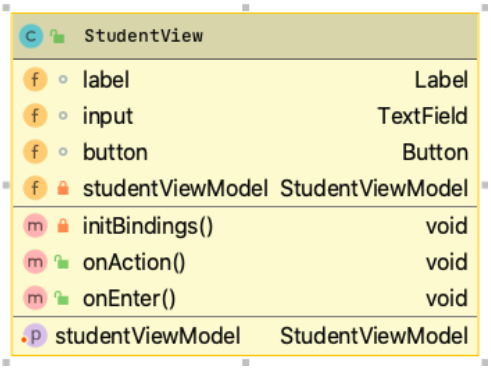
Kopieren Sie die beiden Klassen Domain Controller und Student Model aus der Aufgabe V2.1. Kopieren Sie die Datei `StudentView.fxml` im Resource-Folder in einen Ordner, welcher dem Package von StudentView.java entspricht.

1. View

Die View mit Name `StudentView.fxml` ist bereits vorgeben. Sie ist mit der Klasse StudentView über das FXML Binding verbunden. Insgesamt gibt es drei Steuerelemente: label, input und button. Über die Methoden `onAction` und `onEnter` informiert die fxml-Datei die StudentView.

2. StudentView

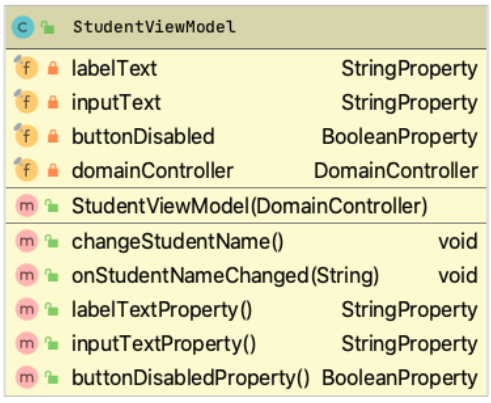
Die StudentView Klasse soll eine Methode beinhalten, um den StudentViewModel zu initialisieren. Die Initialisierung soll das Property studentViewModel initialisieren und initBindings() aufrufen. Die Methoden `onEnter` und `onAction` leiten den Aufruf an das ViewModel weiter.




```
private void initBindings() {
    label.textProperty().bind(studentViewModel.labelTextProperty());
    input.textProperty().bindBidirectional(studentViewModel.inputTextProperty());
    button.disableProperty().bind(studentViewModel.buttonDisabledProperty());
}
```

3. StudentViewModel

Die Klasse StudentViewModel soll einen Konstruktur beinhalten, welcher den DomainController initialisiert, das Binding von inputText.isEmpty nach buttonDisabled durchführt und den Initialwert aus dem DomainModel liest.

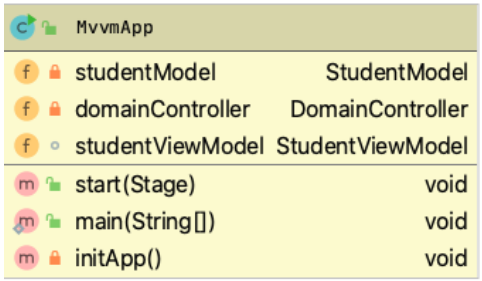



```
public StudentViewModel(DomainController domainController) {
    this.domainController = domainController;
    buttonDisabled.bind(inputText.isEmpty());
    labelText.set(domainController.getStudentName());
}
```

Die Methode changeStudentName() ruft die Systemoperation im DomainController auf. Die Methode onStudentNameChanged wird via PropertyChangeEvent aufgerufen. Die PropertyFields werden über die Methoden labelTextProperty, inputTextProperty und buttonDisabledProperty der View für das Binding zur Verfügung gestellt.

4. MvvmApp

Die Klasse MvvmApp kann aus der Klasse MvpApp der Lernaufgabe V2.2 übernommen werden. Zusätzlich muss das StudentViewModel noch instanziiert werden.



Ergebnis

Ein funktionierendes JavaFX-Programm nach dem MVVM Pattern implementiert.

Zeit: 60’