MVVM mit JavaFX

Max Wielsch & *Manuel Mauky*
@simawiel @manuel_mauky

02.11.2016

JUG Görlitz

Saxonia Systems
So geht Software.

Manuel Mauky
Software Architekt

manuel.mauky@saxsys.de
http://lestard.eu
@manuel_mauky

Max Wielsch
Software Engineer

max.wielsch@saxsys.de
http://max-wielsch.blogspot.com
@simawiel

SPEAKER

JUG Görlitz

Saxonia Systems

So geht Software.

Individual-Softwareentwicklung

Dresden, München, Berlin, Hamburg, Leipzig, **Görlitz**

225 Mitarbeiter

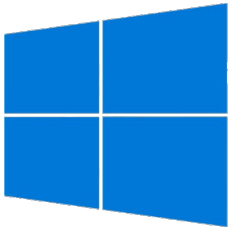
3 Scrum-Teams in JavaFX-Projekten

± 7 Consultants, die JavaFX-Projekte bei Kunden unterstützen

JavaFX

JavaFX

- Toolkit für Java UIs
- Teil der JRE
- Offizieller Ersatz für Swing



Windows

Mac OS

diverse
Linux-Distros

JavaFX Features

Media Engine

Hardware Acceleration

Charts

Timelines

Effects & Animations

Properties

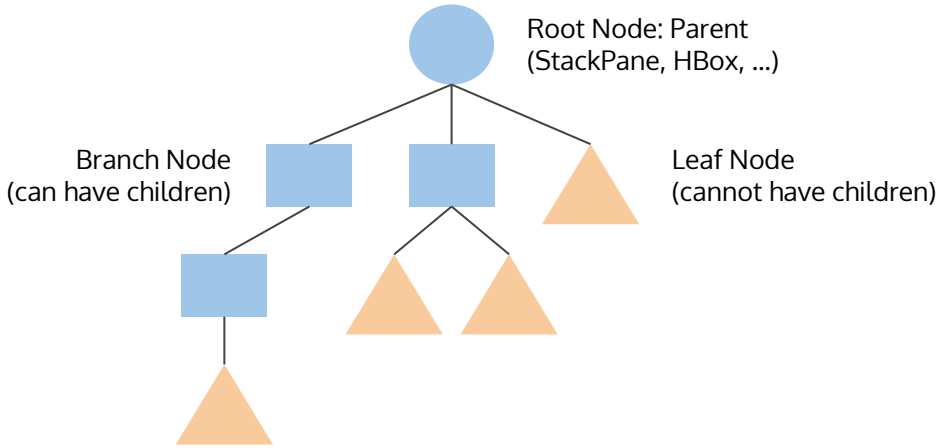
Web View

FXML + CSS

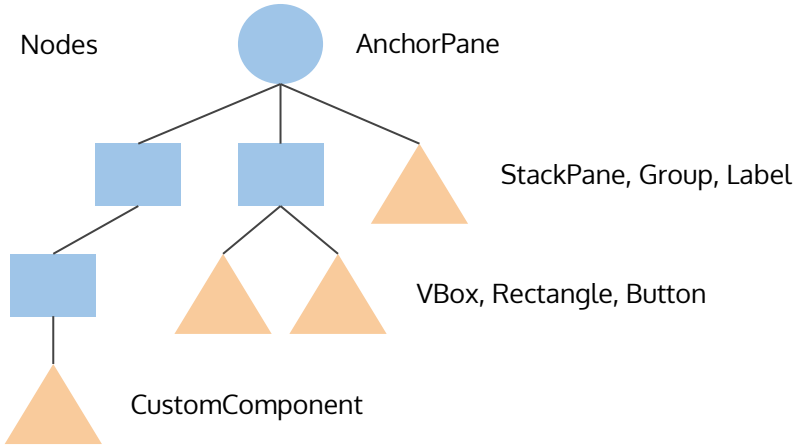
Multi Touch

Packaging

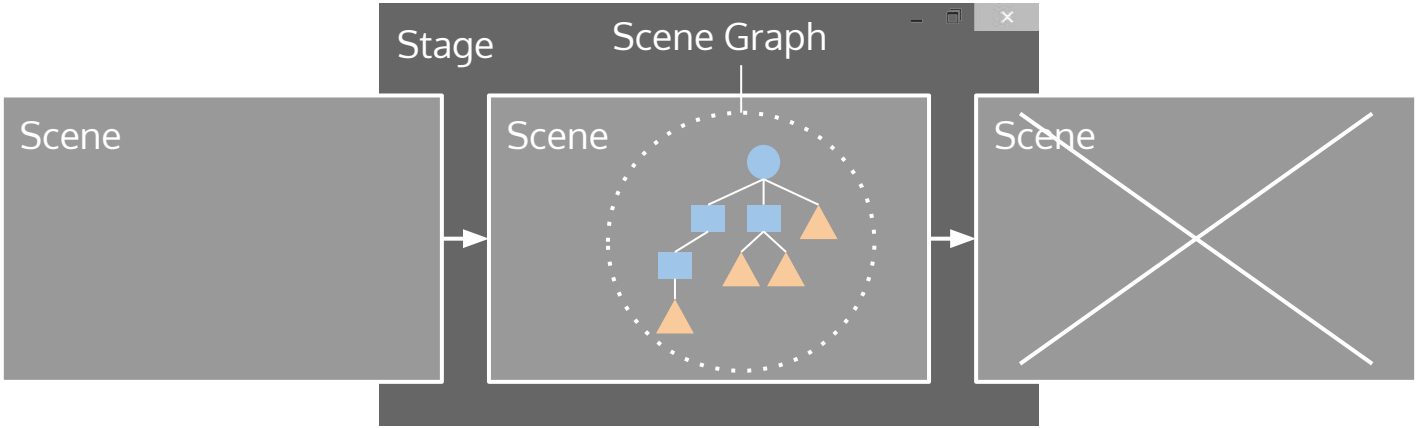
Scene Graph


Root Node: Parent
(StackPane, HBox, ...)
Branch Node
(can have children)
Leaf Node
(cannot have children)

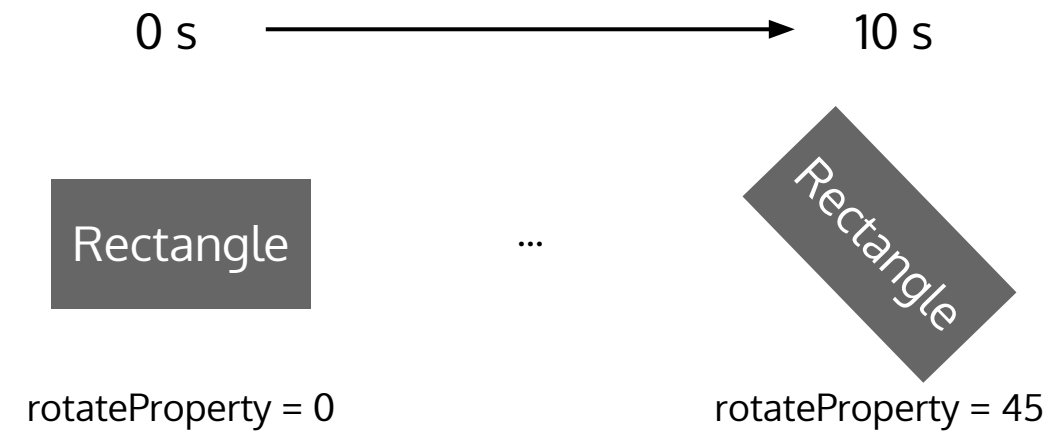
Scene Graph


Nodes
AnchorPane
StackPane, Group, Label
VBox, Rectangle, Button
CustomComponent

Scene Graph



Bsp.: Transitions




```
RotateTransition rotateTransition = new RotateTransition(
                                                Duration.seconds(10),
                                                reactangle);

rotateTransition.setByAngle(45);
rotateTransition.play();
```

Trennung von Darstellung und Verhalten

```
<BorderPane id=“border“>
       <top>
          <Label text=“Page Title“/>
       </top>
       <center>
          <Label text = “Some Data Here“/>
       </center>
</BorderPane>
```

Test.fxml

Structure

```
@FXML
private final BorderPane border;

public MainView() {
  border.setDisabled(true);
}
```

Test.java

Verhalten

```
#border {
    -fx-background-color:white;
}

.Label {
    -fx-text-fill: black;
}
```

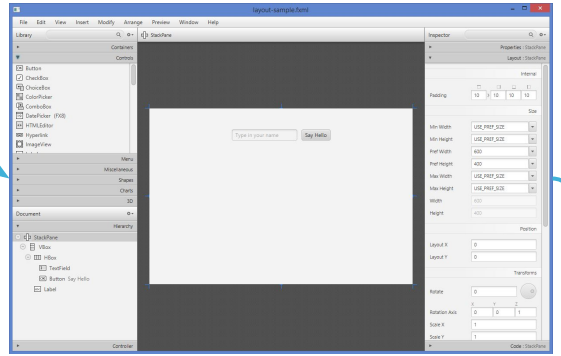
Test.css

Darstellung

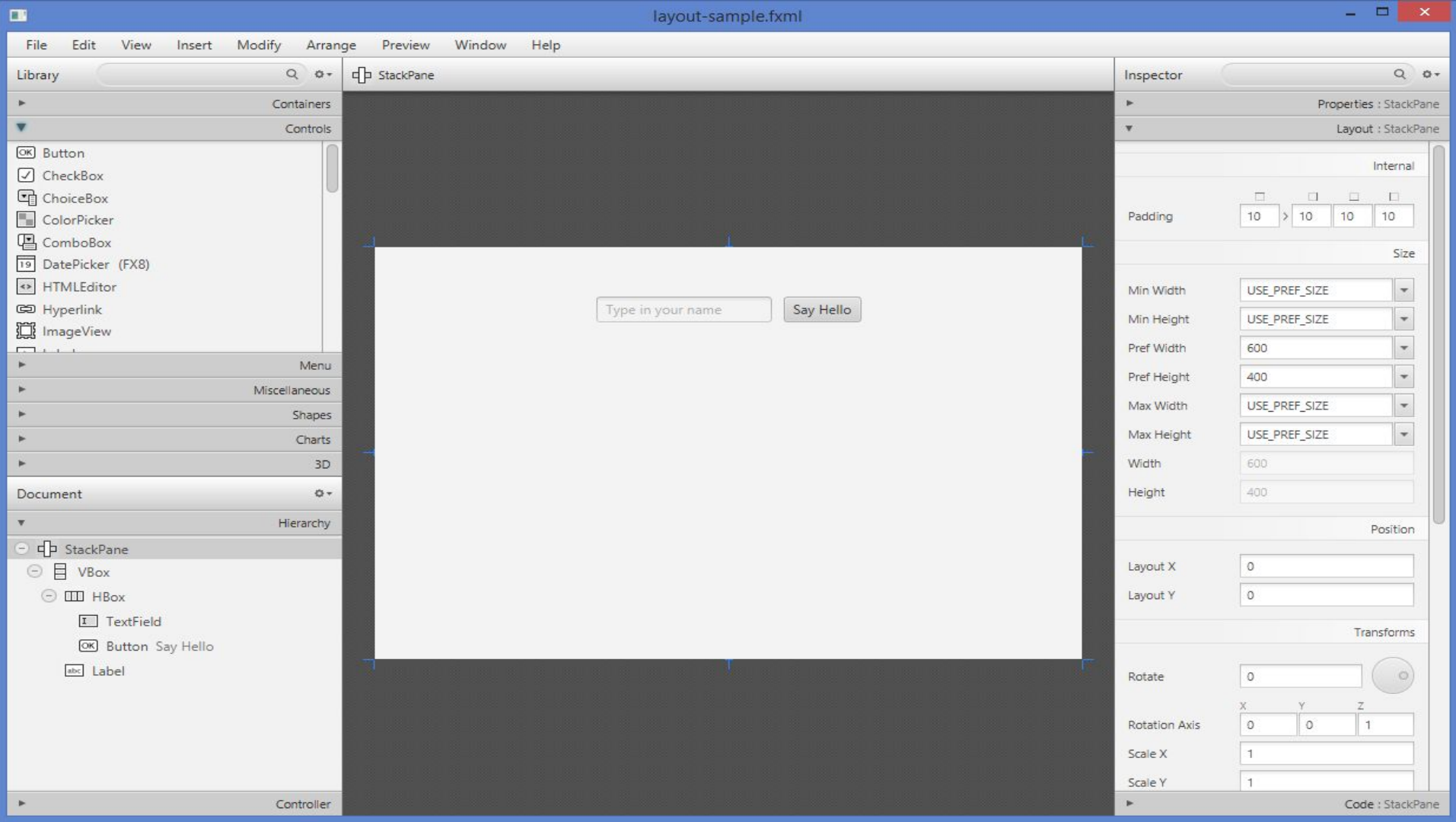
Saubere Trennung dieser Aspekte

Gluon Scene Builder

```
<BorderPane id="border">
        <top>
           <Label text="Page Title"/>
        </top>
        <center>
           <Label text = "Some Data Here"/>
        </center>
</BorderPane>
```



```
<BorderPane id="border">
        <top>
           <Label text="Page Title"/>
        </top>
        <center>
           <Label text = "Some Data Here"/>
        </center>
</BorderPane>
```

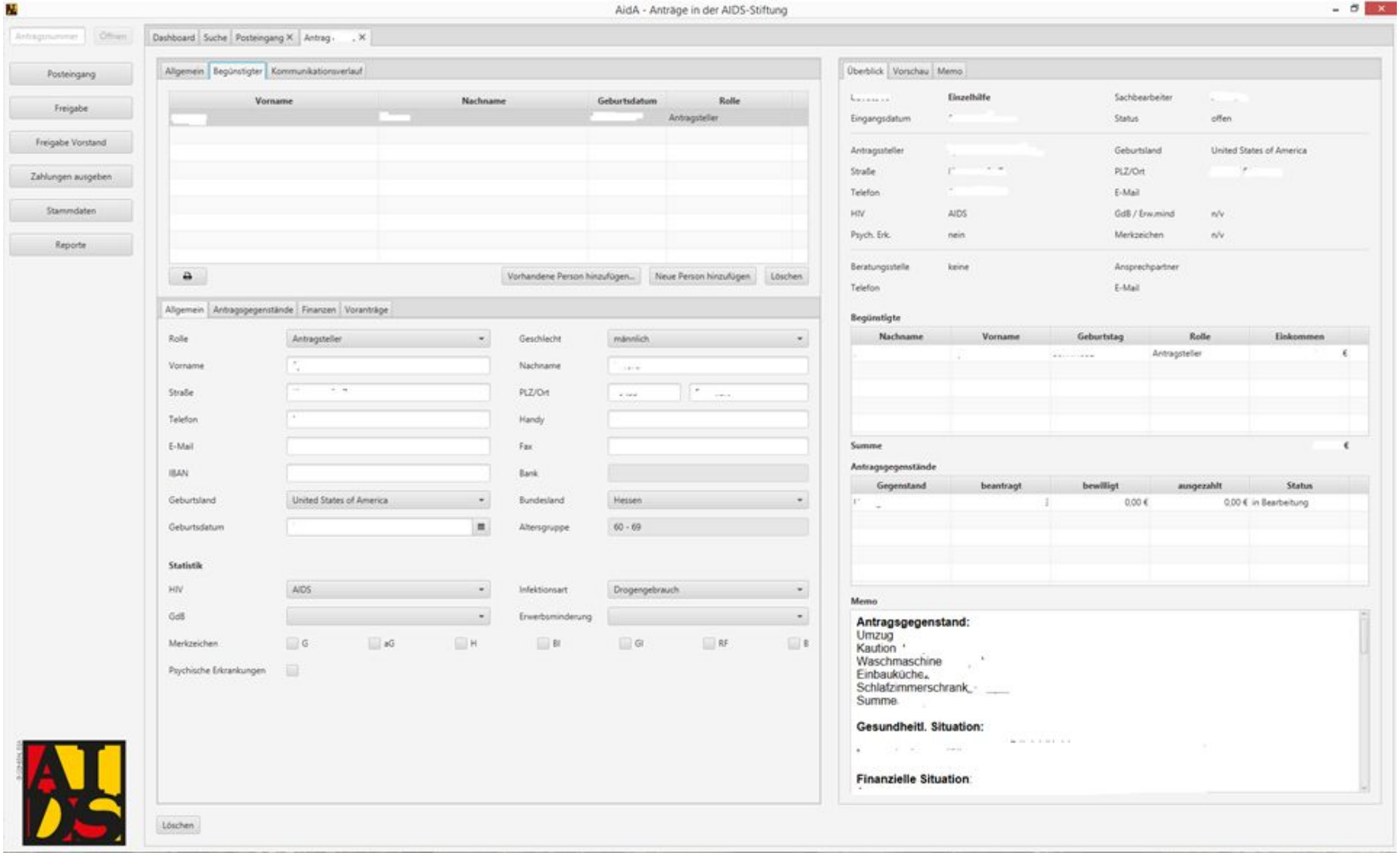

layout-sample.fxml
File
Edit
View
Insert
Modify
Arrange
Preview
Window
Help
Library
Containers
Controls
Button
CheckBox
ChoiceBox
ColorPicker
ComboBox
DatePicker (FX8)
HTMLEditor
Hyperlink
ImageView
Menu
Miscellaneous
Shapes
Charts
3D
Document
Hierarchy
StackPane
VBox
HBox
TextField
Button Say Hello
Label
Controller
StackPane
Type in your name
Say Hello
Inspector
Properties : StackPane
Layout : StackPane
Internal
Padding
10
10
10
10
Size
Min Width
USE_PREF_SIZE
Min Height
USE_PREF_SIZE
Pref Width
600
Pref Height
400
Max Width
USE_PREF_SIZE
Max Height
USE_PREF_SIZE
Width
600
Height
400
Position
Layout X
0
Layout Y
0
Transforms
Rotate
0
X
Y
Z
Rotation Axis
0
0
1
Scale X
1
Scale Y
1
Code : StackPane

JavaFX @ SaxSys


ETEO


e·t·e·o Board
ETEO


1. Sprint - Spielwiese
Time
Effort
ETEO Open
ETEO In Progress
ETEO In Review
ETEOB-1
ETEOB-2
ETEOB-3
ETEOB-4
ETEOB-5
Site
Call for bids
Excavation work
Lawn
Jungle gym
P3 ETEOB-207
P3 ETEOB-209
P3 ETEOB-37
Seesaw
Carousel
Benches
Slide
eteoBoard Premium -
Software inclusive Hardware InFocus Mondopad


AidA - Anträge in der AIDS-Stiftung
Antragsnummer
Öffnen
Dashboard
Suche
Posteingang ×
Antrag ×
Posteingang
Freigabe
Freigabe Vorstand
Zahlungen ausgeben
Stammdaten
Reporte
Allgemein
Begünstigter
Kommunikationsverlauf
Vorname
Nachname
Geburtsdatum
Rolle
Antragsteller
Vorhandene Person hinzufügen...
Neue Person hinzufügen
Löschen
Allgemein
Antragsgegenstände
Finanzen
Voranträge
Rolle
Antragsteller
Geschlecht
männlich
Vorname
Nachname
Straße
PLZ/Ort
Telefon
Handy
E-Mail
Fax
IBAN
Bank
Geburtsland
United States of America
Bundesland
Hessen
Geburtsdatum
Altersgruppe
60 - 69
Statistik
HIV
AIDS
Infektionsart
Drogengebrauch
GdB
Erwerbsminderung
Merkzeichen
G
aG
H
Bl
Gl
RF
B
Psychische Erkrankungen
Löschen
Überblick
Vorschau
Memo
Einzelhilfe
Sachbearbeiter
Eingangsdatum
Status
offen
Antragsteller
Geburtsland
United States of America
Straße
PLZ/Ort
Telefon
E-Mail
HIV
AIDS
GdB / Erw.mind
n/v
Psych. Erk.
nein
Merkzeichen
n/v
Beratungsstelle
keine
Ansprechpartner
Telefon
E-Mail
Begünstigte
Nachname
Vorname
Geburtstag
Rolle
Einkommen
Antragsteller
Summe
Antragsgegenstände
Gegenstand
beantragt
bewilligt
ausgezahlt
Status
0,00 €
0,00 €
in Bearbeitung
Memo
Antragsgegenstand:
Umzug
Kaution
Waschmaschine
Einbauküche
Schlafzimmerschrank
Summe
Gesundheitl. Situation:
Finanzielle Situation:

APPLIKATIONS STRUKTUR

Applikations Modell

Aussehen

Verhalten

MODEL VIEW VIEWMODEL

Model View ViewModel

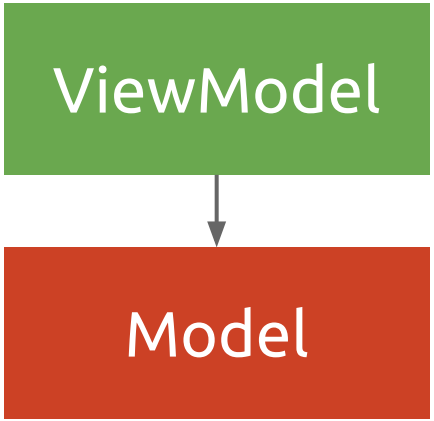
- Basiert auf Presentation Model Pattern
- 2005 von Microsoft publiziert
- WPF (.net), JavaScript (knockout.js), ...

Model

- Applikationsmodell
- Unabhängig von UI
- Backend-System usw.

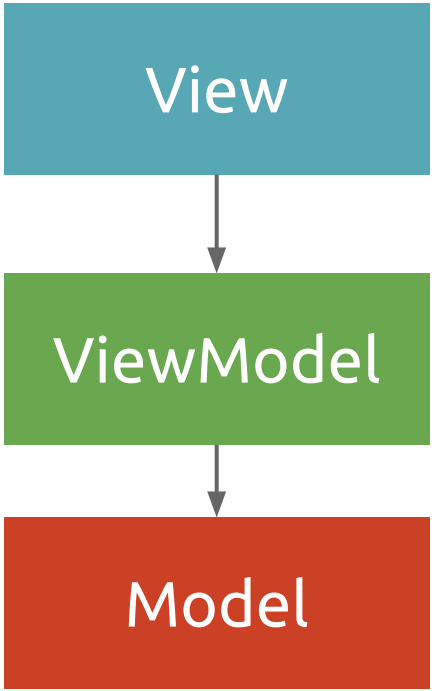
ViewModel

- UI-Zustand
- Präsentationslogik
- Kommunikation mit Backend
- Aufbereitung von Modell-Daten

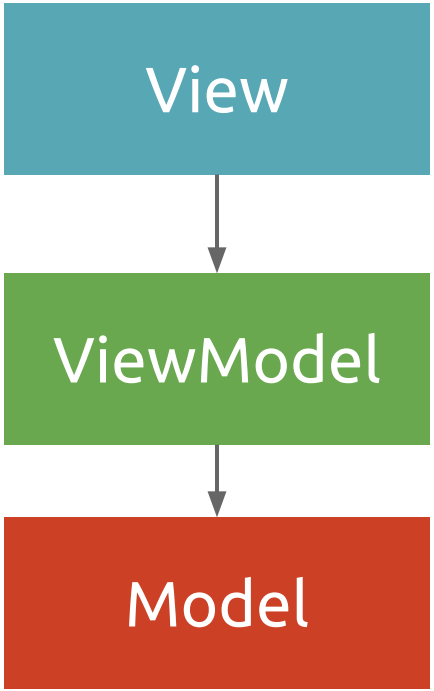


View

- Daten aus ViewModel anzeigen
- Nutzereingaben an ViewModel weitergeben
- UI-Zustand im ViewModel aktualisieren

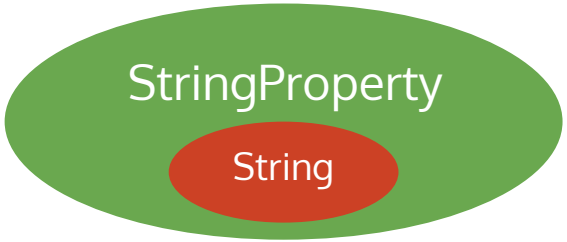


VIEW UND VIEWMODEL SYNCHRON HALTEN?


View
ViewModel
Model

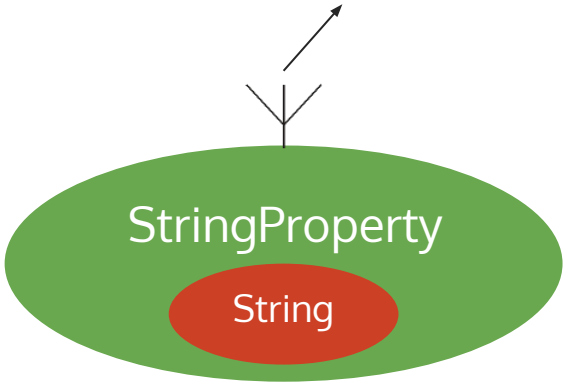
Data Binding und Properties

Data Binding und Properties

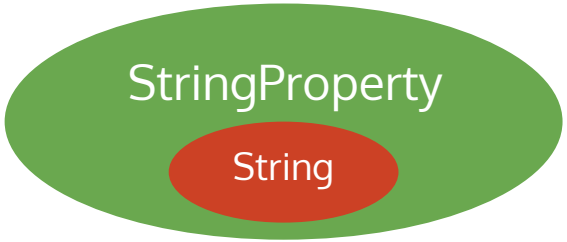


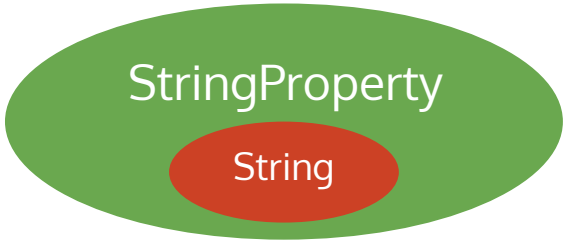
Data Binding und Properties

notifications about changes
(events)

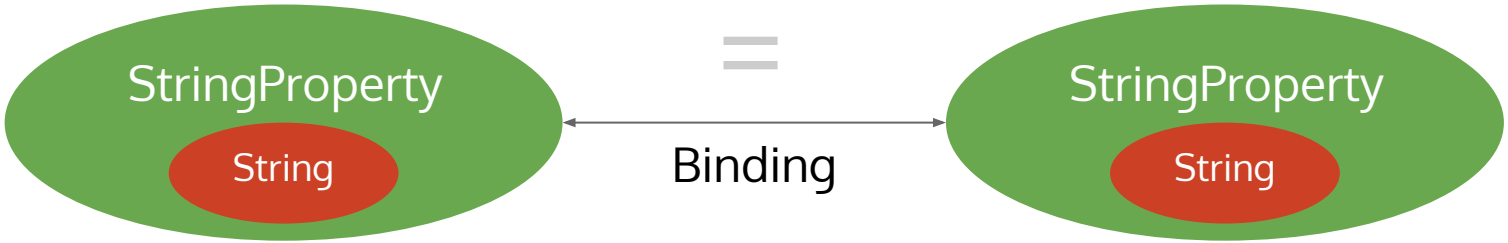


Data Binding und Properties





Data Binding und Properties



```
StringProperty a = new SimpleStringProperty();
StringProperty b = new SimpleStringProperty();

a.bindBidirectional(b);

a.setValue(“Hallo”);
System.out.println(b.getValue()); // “Hallo”
b.setValue(“World”);
System.out.println(a.getValue()); // “World”
```

```
IntegerProperty
DoubleProperty
StringProperty
BooleanProperty
ListProperty<T>
MapProperty<K,V>
ObjectProperty<T>
```

Data Binding in MVVM

Model

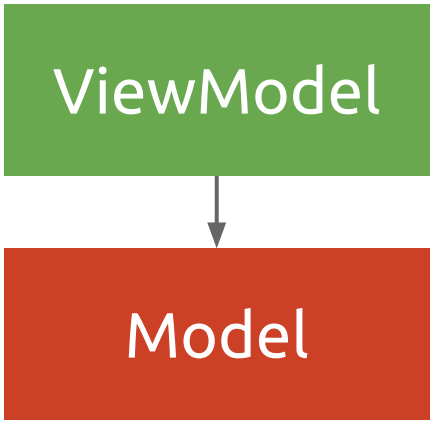
`firstName`

"Max"

`lastName`

"Wielsch"

Data Binding in MVVM



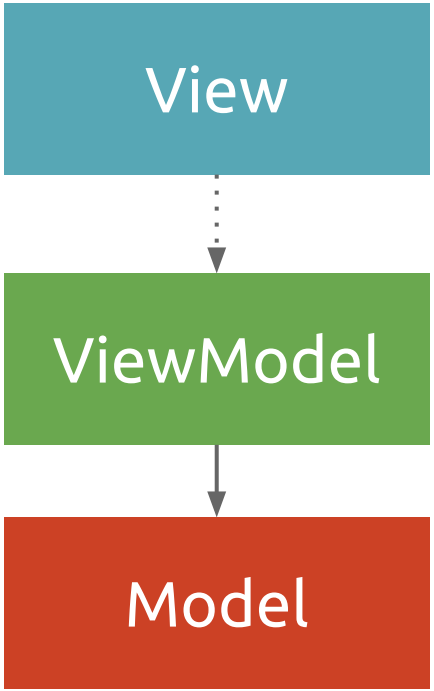

```
welcomeMessageProperty.set(
     “Hallo “
     + user.getFirstName() + “ ”
     + user.getLastName());
```

“Hallo Max Wielsch”

`firstName` → *“Max”*

`lastName` → *“Wielsch”*

Data Binding in MVVM




```
welcomeLabel.textProperty().bind(
    viewModel.welcomeMessageProperty());

welcomeMessageProperty.set(
      “Hallo “
      + user.getFirstName() + “ ”
      + user.getLastName());
```

“Hallo Max Wielsch”

`firstName` → *“Max”*

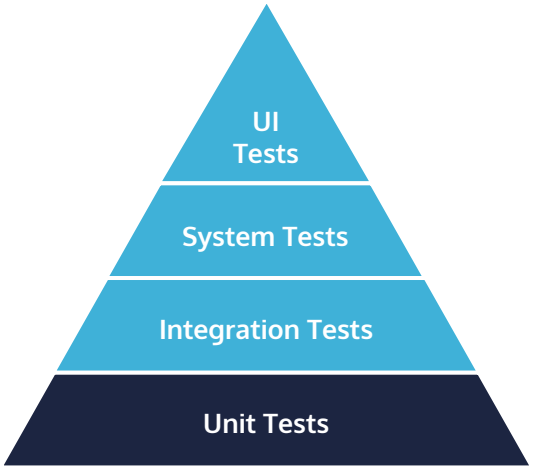
`lastName` → *“Wielsch”*

Nutzen

- gesamte Präsentationslogik ist nur im ViewModel
- ViewModel ist mit Unit-Tests testbar

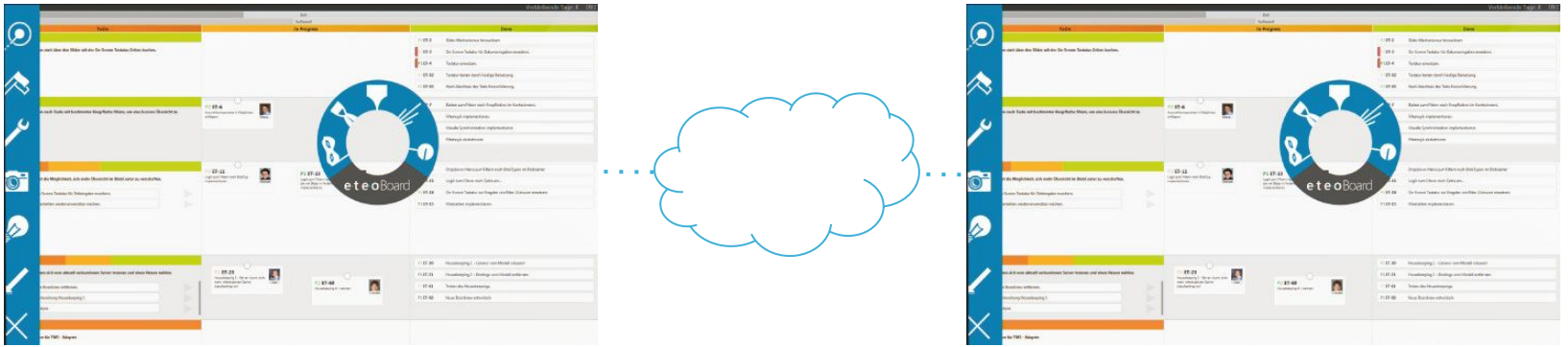
→ Hohe Testbarkeit von Frontend-Code

→ Test-Driven-Development der UI



Nutzen

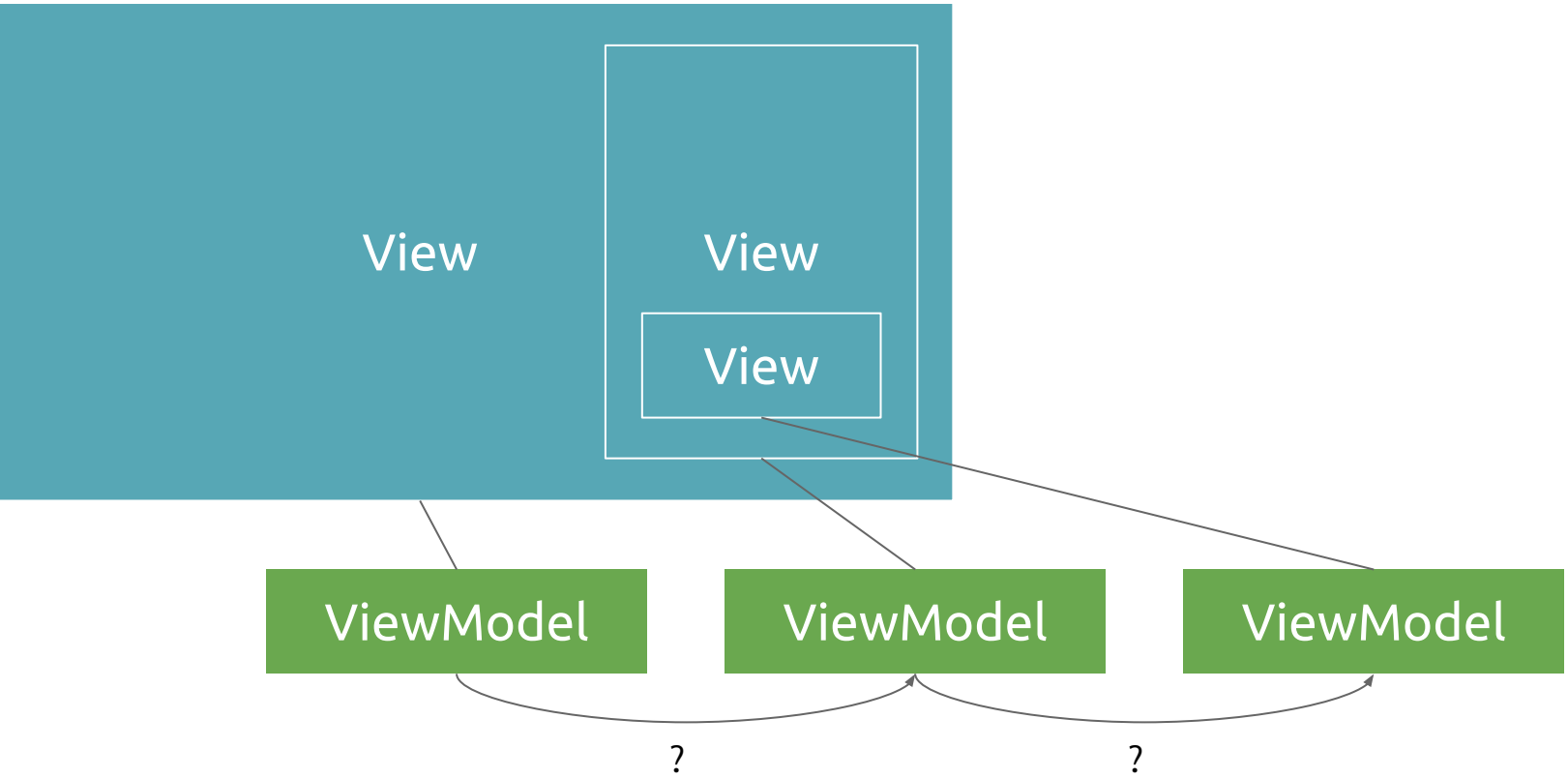
- UI-Zustand ist serialisierbar
  - Persistieren und Wiederherstellen von Nutzer-Sessions
  - Synchronisierung über Netzwerk



LIVE CODING TDD

Example: Todo List

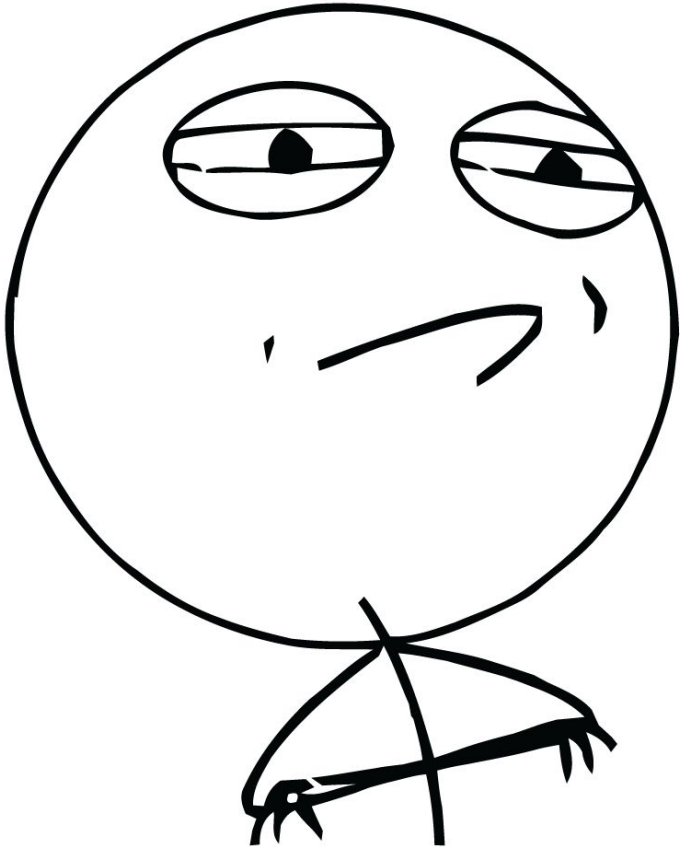
View-Hierarchien



Herausvorderungen mit MVVM

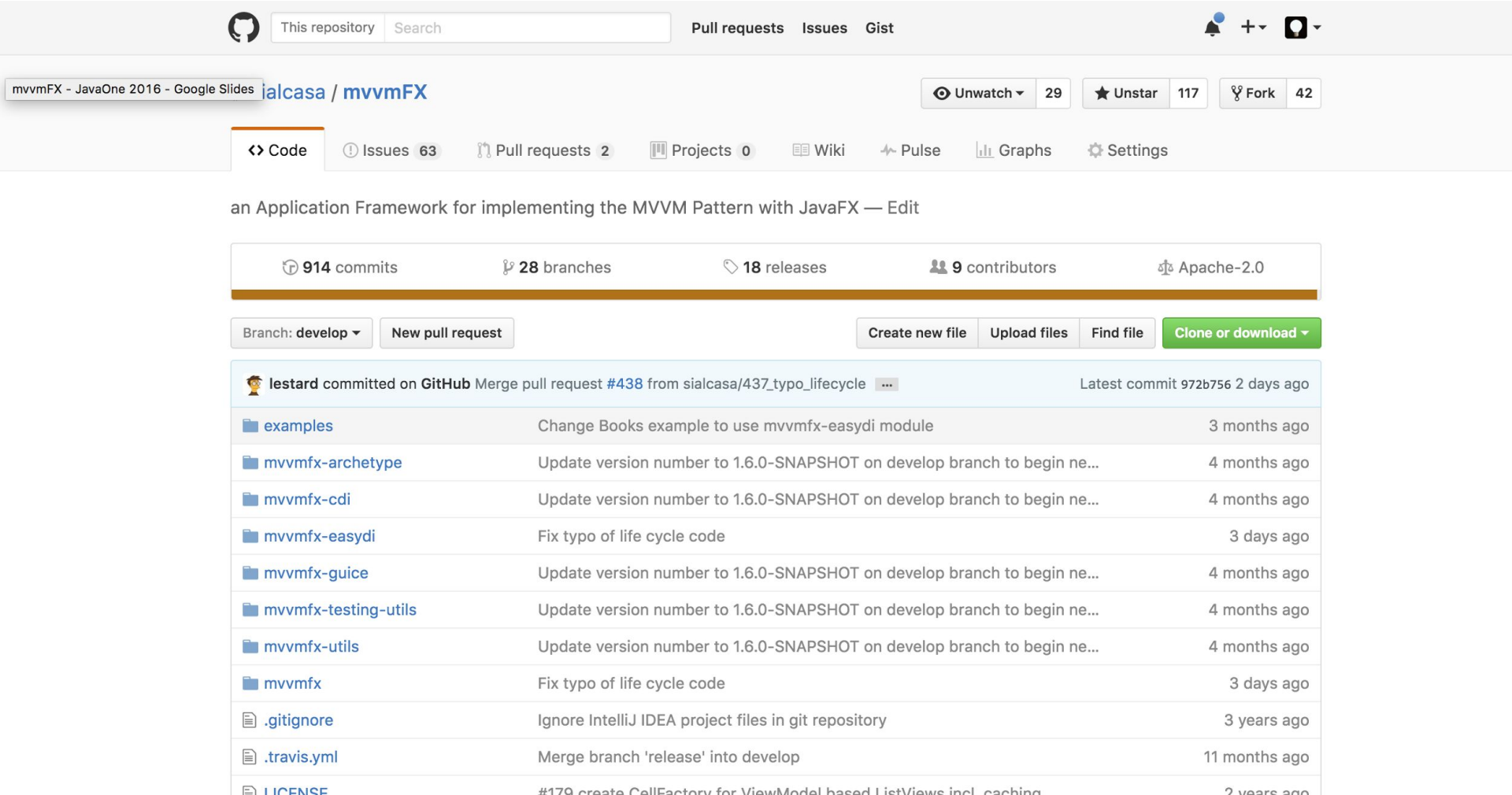
- Kommunikation zwischen VMs in komplexeren Anwendungen
- Erfordert Disziplin, Sichtbarkeiten nicht zu verletzen
- Mehr Code für Indirektionsschicht nötig

CHALLENGE ACCEPTED



... ist ein Application-Framework, das benötigte Komponenten für die Verwendung des Model-View-ViewModel-Patterns mit JavaFX zur Verfügung stellt.

https://github.com/sialcasa/mvvmFX


This repository
Search
Pull requests
Issues
Gist
mvvmFX - JavaOne 2016 - Google Slides
ialcasa / mvvmFX
Unwatch 29
Unstar 117
Fork 42
Code
Issues 63
Pull requests 2
Projects 0
Wiki
Pulse
Graphs
Settings
an Application Framework for implementing the MVVM Pattern with JavaFX — Edit
914 commits
28 branches
18 releases
9 contributors
Apache-2.0
Branch: develop
New pull request
Create new file
Upload files
Find file
Clone or download
lestard committed on GitHub Merge pull request #438 from sialcasa/437_typo_lifecycle
Latest commit 972b756 2 days ago
examples Change Books example to use mvvmfx-easydi module 3 months ago
mvvmfx-archetype Update version number to 1.6.0-SNAPSHOT on develop branch to begin ne... 4 months ago
mvvmfx-cdi Update version number to 1.6.0-SNAPSHOT on develop branch to begin ne... 4 months ago
mvvmfx-easydi Fix typo of life cycle code 3 days ago
mvvmfx-guice Update version number to 1.6.0-SNAPSHOT on develop branch to begin ne... 4 months ago
mvvmfx-testing-utils Update version number to 1.6.0-SNAPSHOT on develop branch to begin ne... 4 months ago
mvvmfx-utils Update version number to 1.6.0-SNAPSHOT on develop branch to begin ne... 4 months ago
mvvmfx Fix typo of life cycle code 3 days ago
.gitignore Ignore IntelliJ IDEA project files in git repository 3 years ago
.travis.yml Merge branch 'release' into develop 11 months ago

Basic Classes for MVVM

Extended FXML Loader

Dependency Injection Support

ResourceBundles

ModelWrapper

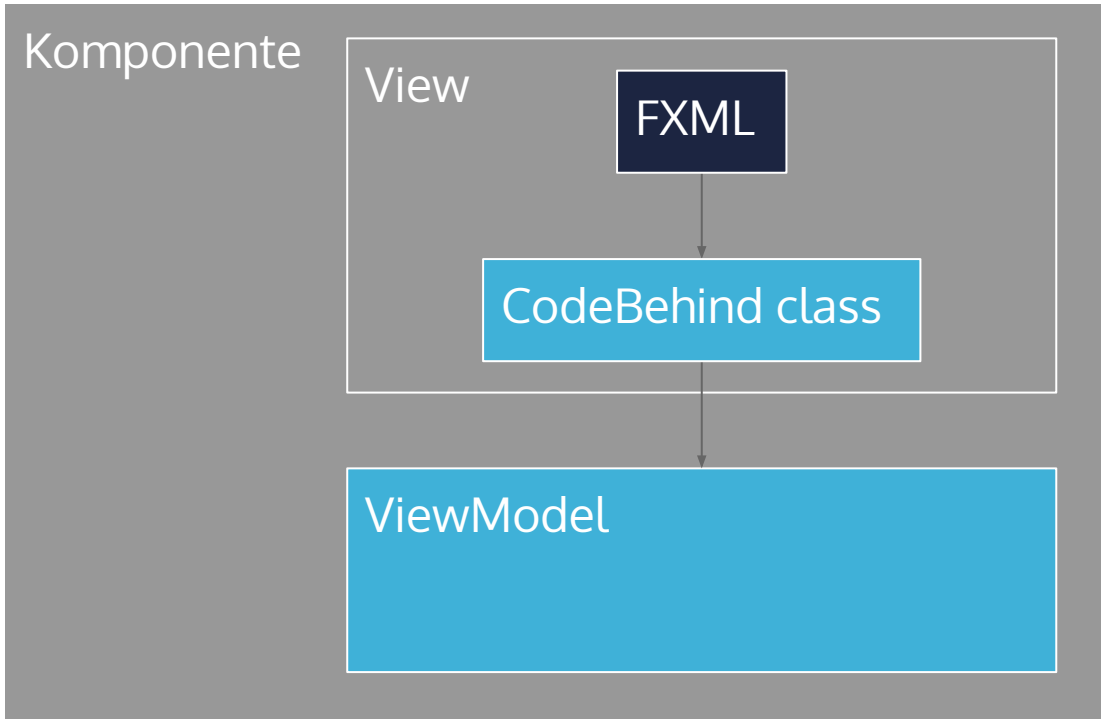
Notifications

Commands

Validation

Scopes

Wie erstellt man eine MVVM-Komponente?



Konzept

XML-Code

Java-Code

Base Classes / Interfaces

```
class TodolistViewModel implements ViewModel {
    …
}

class TodolistView implements FxmlView<TodolistViewModel> {

    @InjectViewModel
    private TodolistViewModel viewModel;

}
```

TodolistView.fxml:

```
<?xml version="1.0" encoding="UTF-8"?>
…
<VBox xmlns:fx="http://javafx.com/fxml"
    fx:controller="de.saxsys.todo.TodolistView">
  <children>
     ...
  </children>
</VBox>
```

Wie lädt man eine MVVM Komponente?

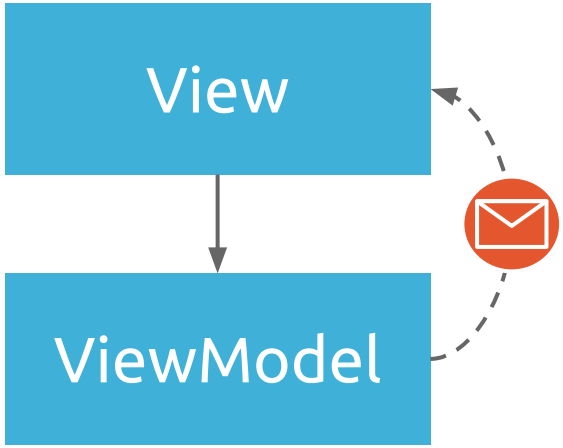
```
URL url = getClass().getResource(“/de/saxsys/MyView.fxml”);
FXMLLoader loader = new FXMLLoader(url);
loader.load();
loader.getRoot(); // loaded node
loader.getController(); // controller class
```

Wie lädt man eine MVVM Komponente?

```
URL url = getClass().getResource(“/de/saxsys/MyView.fxml”);
FXMLLoader loader = new FXMLLoader(url);
loader.load();
loader.getRoot(); // loaded node
loader.getController(); // controller class
```

```
ViewTuple tuple = FluentViewLoader.fxmlView(MyView.class).load();
tuple.getView(); // loaded node (View)
tuple.getCodeBehind(); // controller class (View)
tuple.getViewModel(); // ViewModel
```

Wie löst man ein Event in der View aus?



Notifications

```
public class MyView implements FxmlView<MyViewModel> {
    @InjectViewModel
    private MyViewModel viewModel;
    …
    viewModel.subscribe(“messageKey”, (k,v) ->  doSomething());
    …
}
public class MyViewModel implements ViewModel {
    …
    publish(“messageKey”);
    …
}
```

Wie löst man Abhängigkeiten auf?

```
class MyViewModel implements ViewModel {

    private Service service = new ServiceImpl(); // ?




}
```

Wie löst man Abhängigkeiten auf?

```
class MyViewModel implements ViewModel {

    private Service service = new ServiceImpl(); // ?


    @Inject
    private Service service;

}
```

Dependency Injection

- Inversion of Control
- Keine statische Abhängigkeit zu einer spezifischen Implementierung, nur zu Interfaces
- Mock-Implementierungen für Unit-Tests nutzen
- Lebenszyklus von Instanzen konfigurieren

Dependency Injection Support

```
<dependency>
   <groupId>de.saxsys</groupId>
   <artifactId>mvvmfx-cdi</artifactId>
</dependency>
```

```
<dependency>
   <groupId>de.saxsys</groupId>
   <artifactId>mvvmfx-guice</artifactId>
</dependency>
```

```
<dependency>
   <groupId>de.saxsys</groupId>
   <artifactId>mvvmfx-easydi</artifactId>
</dependency>
```

or

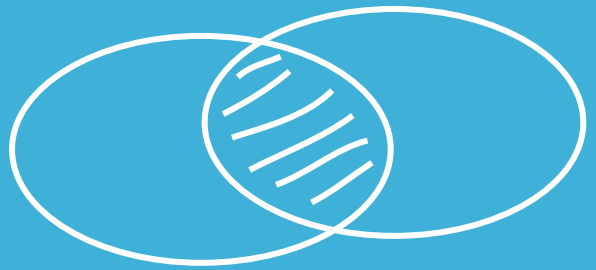
```
MvvmFX.setCustomDependencyInjector(...);
```

Dependency Injection Support

```
public class MyFxApp extends Application {…}
```

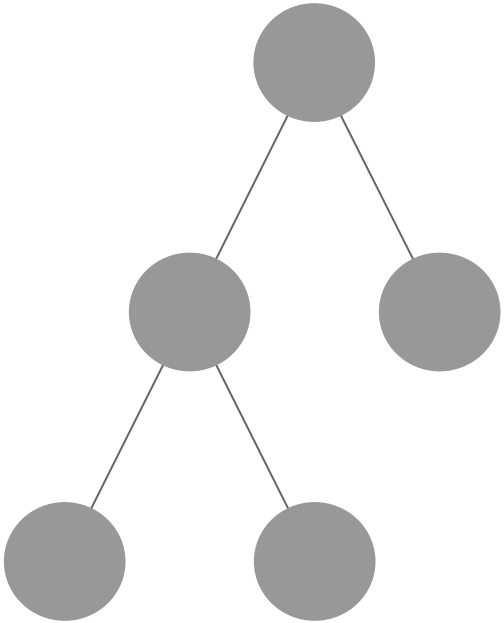


```
public class MyFxApp extends MvvmfxCdiApplication {…}

public class MyFxApp extends MvvmfxGuiceApplication {…}

public class MyFxApp extends MvvmfxEasyDIApplication {…}
```

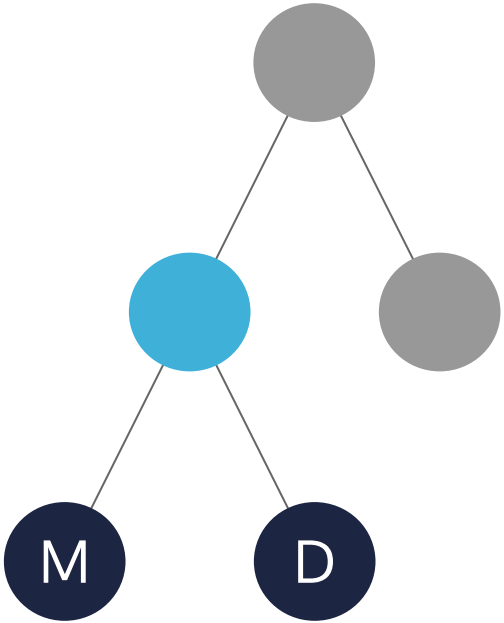


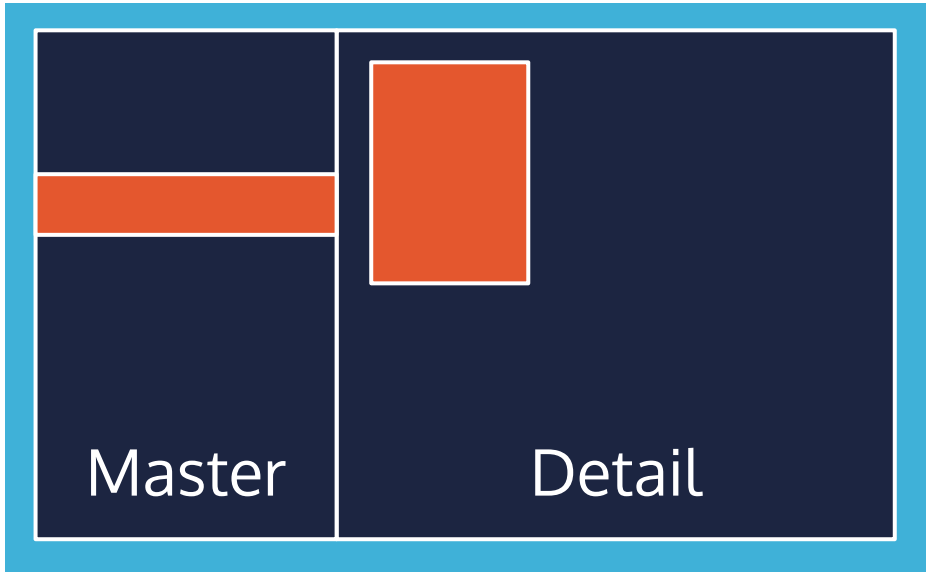
SCOPES

MVVM-Komponenten sind hierarchisch

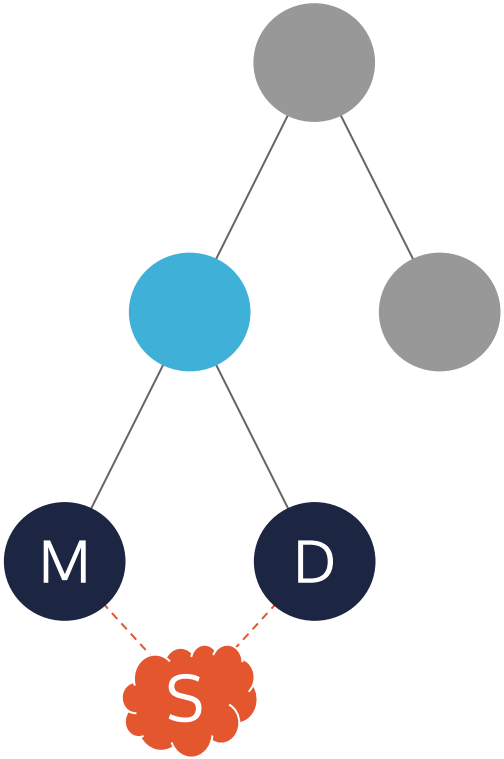


MVVM-Komponenten haben gemeinsamen Zustand





Scopes: Informationen teilen

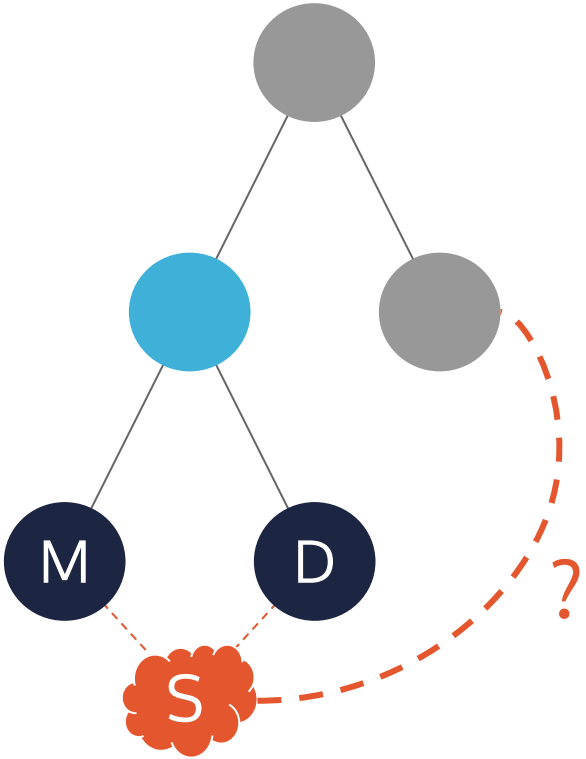


Inspiration: Angular

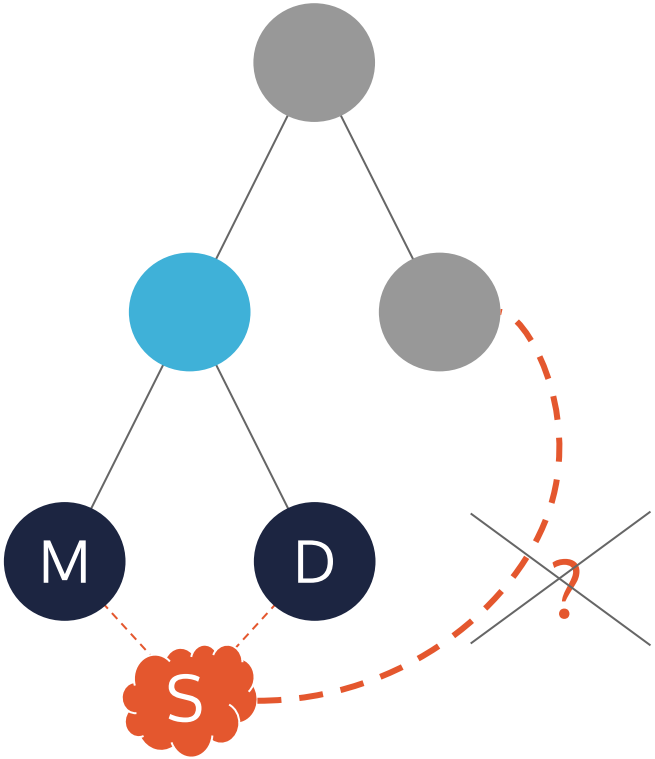
Scope definieren

```
public class PersonScope implements Scope {

    private ObjectProperty<Person> selectedPerson
                                        = new SimpleObjectProperty();

    // Getters & Setters

}
```

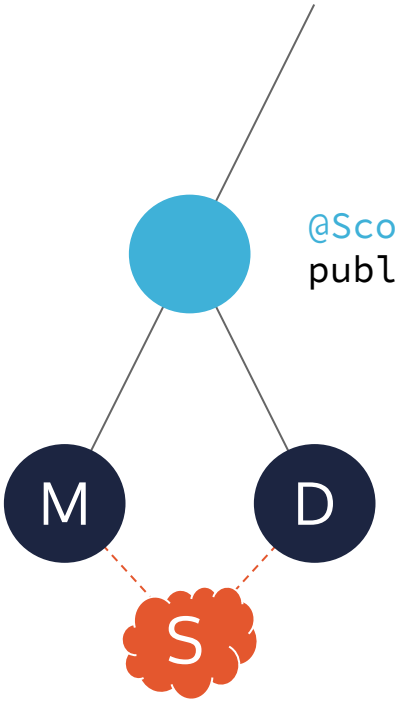
Wie werden Scopes abgegrenzt?



Wie werden Scopes abgegrenzt?

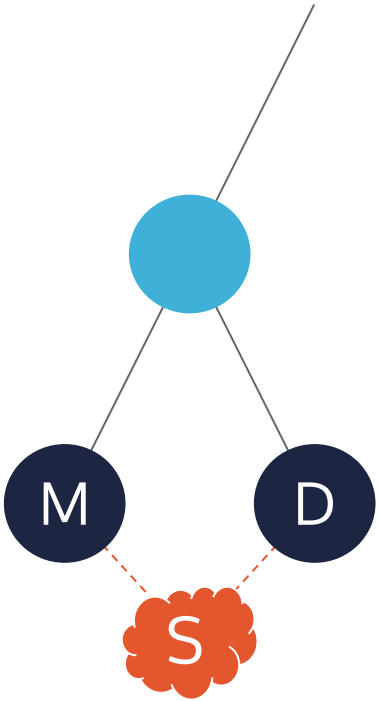


Eine Komponente in der Hierarchie definiert einen Scope




```
@ScopeProvider(scopes={PersonScope.class})
public class PersonViewModel implements ViewModel { }
```

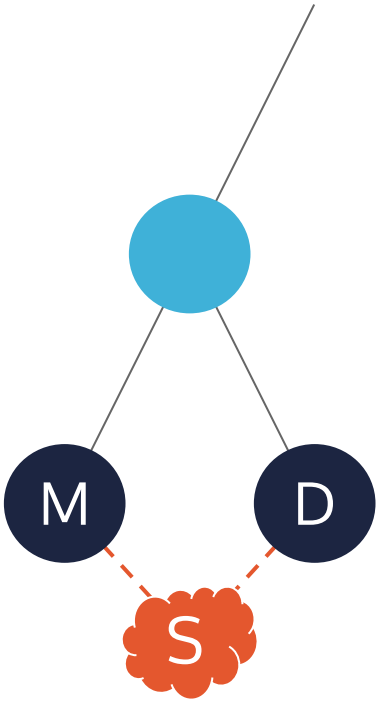
*Unterkomponenten können **dieselbe Scope Instanz** injekten*




```
class PersonsOverviewViewModel implements ViewModel {
    @InjectScope
    private PersonScope scope;
}
```

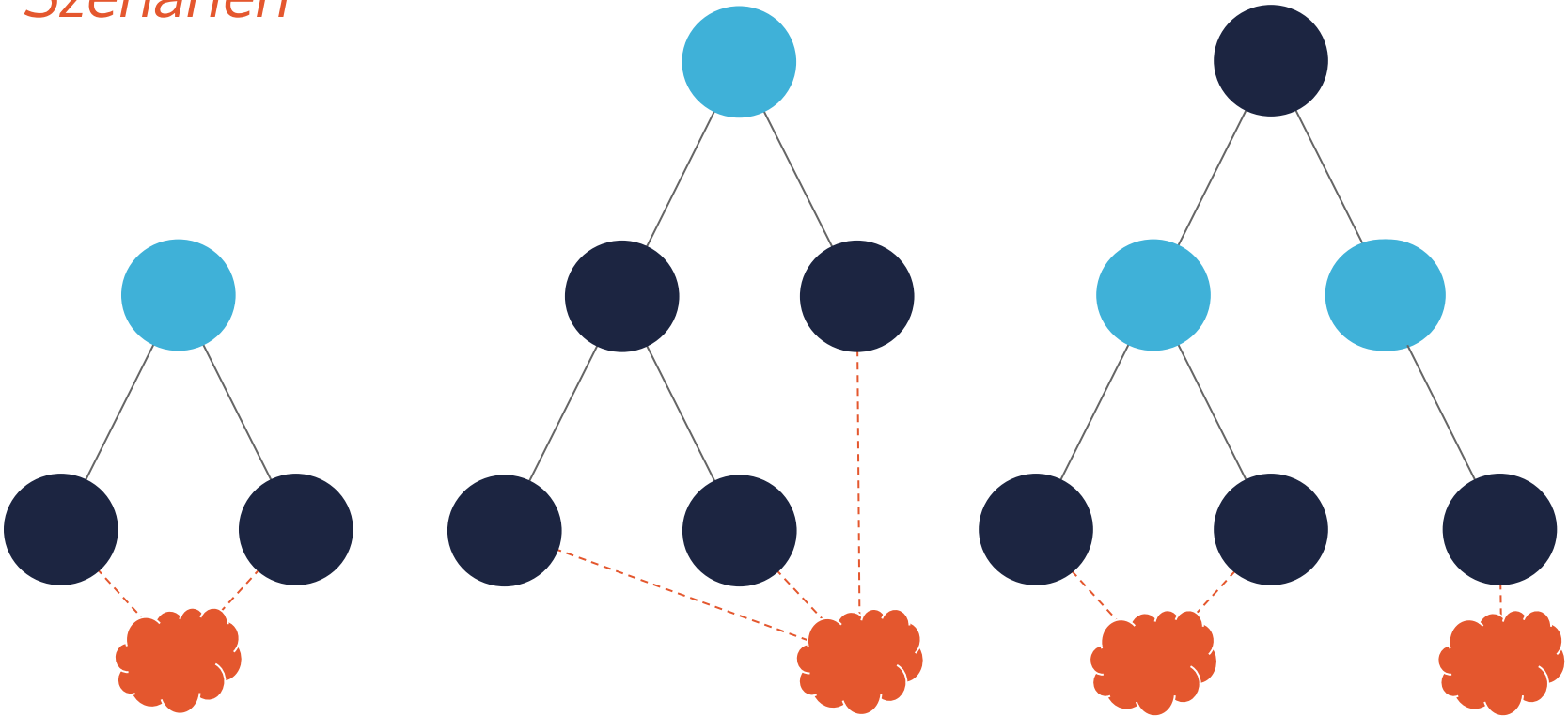
```
class PersonDetailView implements ViewModel {
    @InjectScope
    private PersonScope scope;
}
```

Kommunikation entkoppeln



Scopes können zur Entkopplung der Kommunikation zwischen Komponenten mittels hierarchischer Dependency Injection genutzt werden.

Szenarien



Scope Provider Komponente Scope

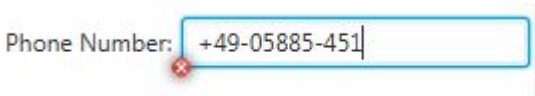
VALIDATION

Validation

Validation logic

```
boolean isPhoneNumberValid(String input) {
    return Pattern.compile("\\+?[0-9\\s]{3,20}")
            .matcher(input).matches();
}
```

Visualization



Validation

ControlsFX ValidationSupport

```
TextField textField = ...;

ValidationSupport validationSupport = new ValidationSupport();

validationSupport.registerValidator(textField,
      Validator.createRegexValidator("Wrong Number", "\\+?[0-9\\s]{3,20}", Severity.ERROR));
```

Validation

```
// ViewModel

private final Validator phoneValidator = ...

public ValidationStatus phoneValidation() {
   return phoneValidator.getValidationStatus();
}

// View

ValidationVisualizer validVisualizer= new ControlsFxVisualizer();
validVisualizer.initVisualization(viewModel.phoneValidation(), textField);
```

Validation

```
StringProperty phoneNumber = new SimpleStringProperty();

Predicate<String> predicate = input -> Pattern.compile("\\+?[0-9\\s]{3,20}")
                                                    .matcher(input).matches();

Validator phoneValidator = new FunctionBasedValidator<>(
     phoneNumber, predicate, ValidationMessage.error("Not a valid phone number");
```

Validation

```
Validator phoneValidator = new FunctionBasedValidator(...);
Validator emailValidator = new ObservableRuleBasedValidator(...);

Validator formValidator = new CompositeValidator();
formValidator.addValidators(phoneValidator, emailValidator);
```

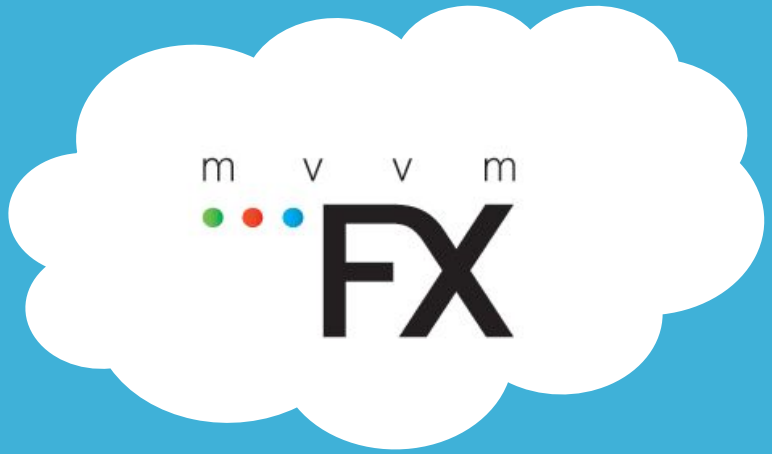
LIFECYCLE

Lifecycle

- Reagieren, wenn Komponenten zur Szene hinzugefügt oder entfernt werden
- Nutzen: Listener hinzufügen / entfernen
- Beispiel: Dialog wird geschlossen

Lifecycle

```
class DialogViewModel implements ViewModel, SceneLifecycle {
     private NotificationObserver observer = (k,v) -> { … };

     @Override
     public void onViewAdded() {
          notificationCenter.subscribe("something", observer);
     }
     @Override
     public void onViewRemoved() {
          notificationCenter.unsubscribe(observer);
     }
}
```



www.mvvmfx.de

Source Code, Wiki, Tutorials: **https://github.com/sialcasa/mvvmFX**

Feedback, Bug Reports, Feature Requests erwünscht :-)

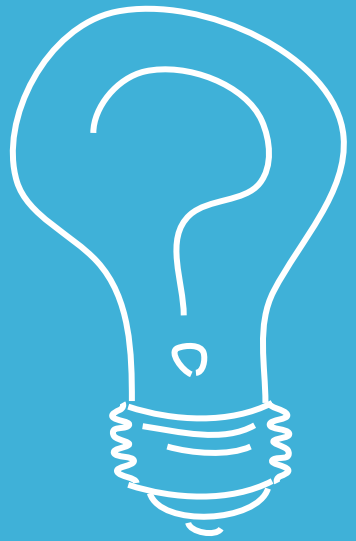
Q & A

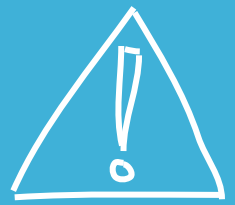
Manuel Mauky

manuel.mauky@saxsys.de
http://lestard.eu
@manuel_mauky

Max Wielsch

max.wielsch@saxsys.de
http://max-wielsch.blogspot.com
@simawiel





PATTERN VERSTÖßE FESTSTELLEN

Wie stelle ich sicher, dass das Pattern eingehalten wird?

- Vorteile von MVVM wirken nur, wenn das Pattern auch eingehalten wird
- Wie findet man Fehler und falsche Benutzung der API?

Wie stelle ich sicher, dass das Pattern eingehalten wird?

- Vorteile von MVVM wirken nur, wenn das Pattern auch eingehalten wird
- Wie findet man Fehler und falsche Benutzung der API?
- AspectJ compile time checking (beta)

```
class MyViewModel implements ViewModel {

    public void initValidation(Label usernameLabel) {
        validationSupport.registerValidator(label,
            Validator
                .createRegexValidator("Error", "...",
            Severity.ERROR);
    }
}
```

```
class MyViewModel implements ViewModel {

    public void initValidation(Label usernameLabel) {
        validationSupport.registerValidator(label,
            Validator
                .createRegexValidator("Error", "...",
            Severity.ERROR);
    }
}
```

javafx.controls.Label used in ViewModel

```
> mvn aspectj:compile
```

```
> mvn aspectj:compile
…
[WARNING] Methods taking UI elements as arguments is invoked within
the ViewModel layer
/.../PersonsViewModel.java:34
validationSupport.registerValidator(label,
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```