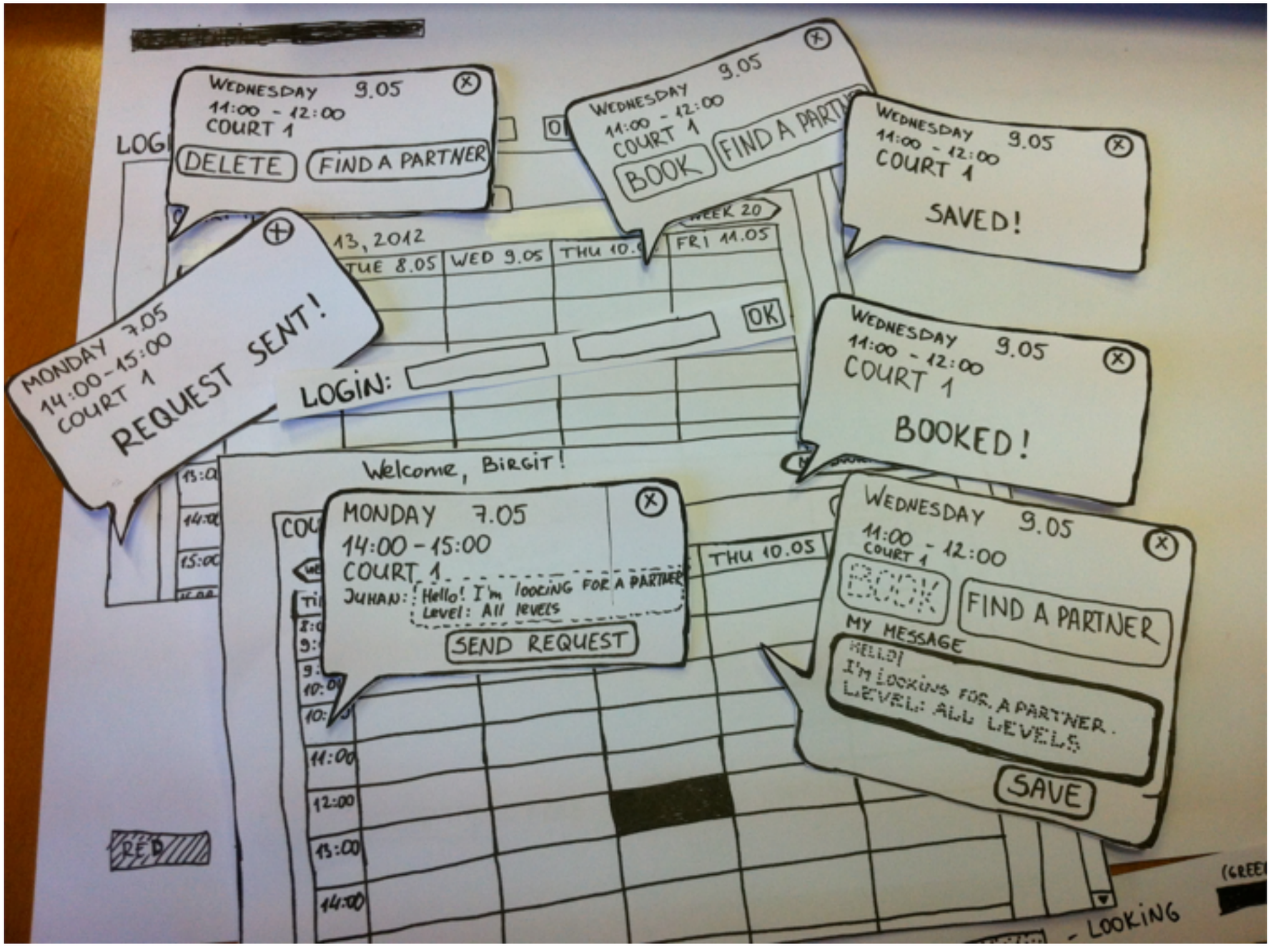
GUI Entwicklung

Softwarepraktikum 2015

Matthias Höschele

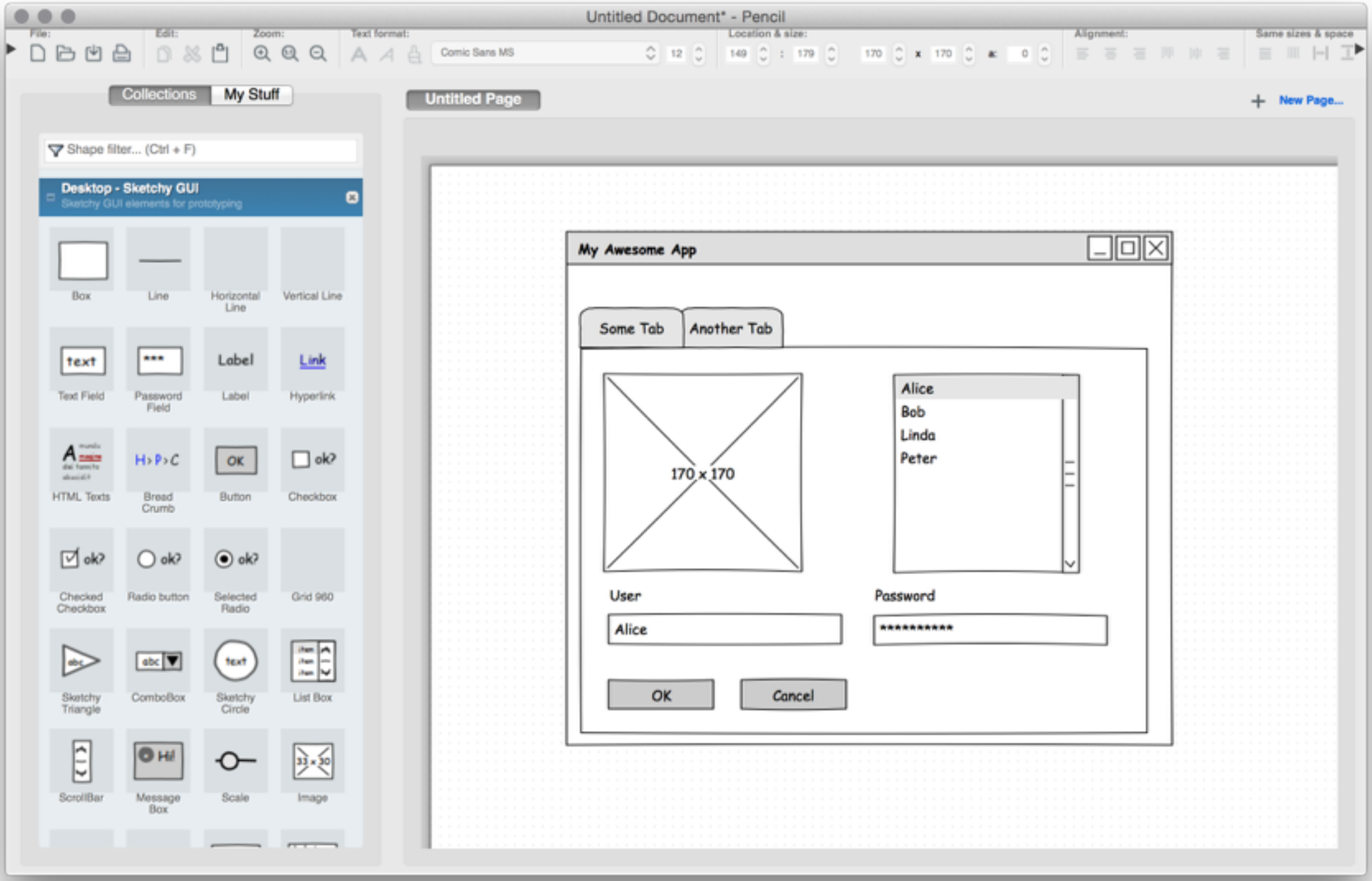
Paper Prototyping



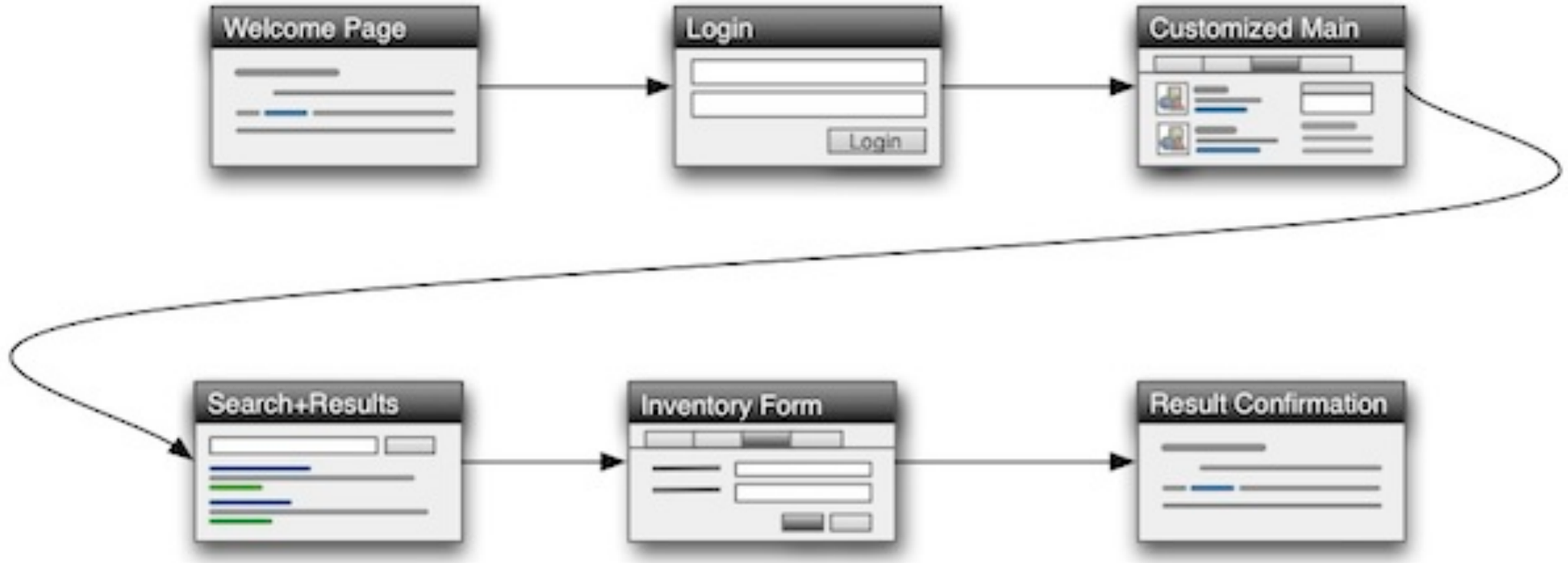
Vorteile

- Schnelles Erstellen eines Mockups ohne dazu programmieren zu müssen.
- Entdeckt frühzeitig Probleme im Design
- Ermöglicht das Einholen von User-Feedback vor dem Implementieren.
- Ermöglicht Zusammenarbeit von Teams aus mehreren Disziplinen.

Wireframing



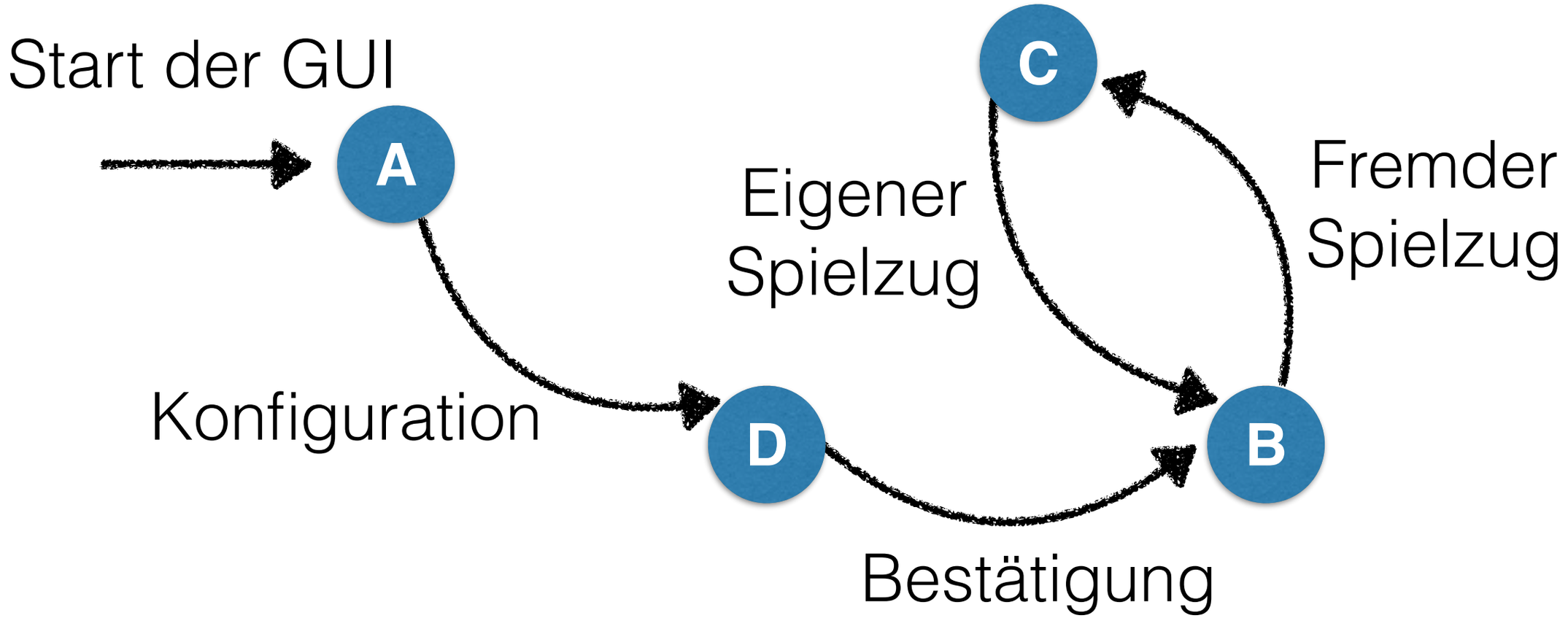
Storyboards



GUI Abnahme

- Paper Prototype/Wireframe (Papier oder Digital) mit klar erkennbaren Elementen.
- Status-Diagramme
- Status-Beschreibungen
- Status-Übergangsbeschreibung
- Interaktionsbeschreibung: GUI zu Datenquelle

Status Diagramme



Status Beschreibungen

A. **Uninitialisiert:** Fenster: *Main*. Panels: *Main:Control*. Alle Buttons gesperrt, *Menu>File>(Connect, Quit)* verfügbar.

B. **Warten auf Mitspieler:** Command Buttons nicht nutzbar, Spielfeld präsent in Panel *Main:Map*. Spielstände in Panel *Main:Info* angezeigt.

C. **Eigener Spielzug:** …

D. **Warten auf Konfiguration:** …

Status Übergänge

Start der GUI: Erstellung des Hauptfensters (*Main*) mit Panel *Main:Control*.

Konfiguration: *Menu>File>Connect* öffnet Dialog: *Config*

Bestätigung: Dialog: *Config:Connect-Button* + gültige Einstellungen schließt Dialog und erstellen Panel *Main:Main* und *Main:Info*.

...

Interaktionen

GUI zu Backend

Main:Control:Connect#click	Erstelle neue Verbindung zum Server
Main:Main:Build#click	Sende einen Build Command zum Server

Interaktionen

Backend zu GUI

Empfange Build Event: Aktualisiere die Anzeige und assoziiere die Strecke mit dem Mitspieler in der GUI.

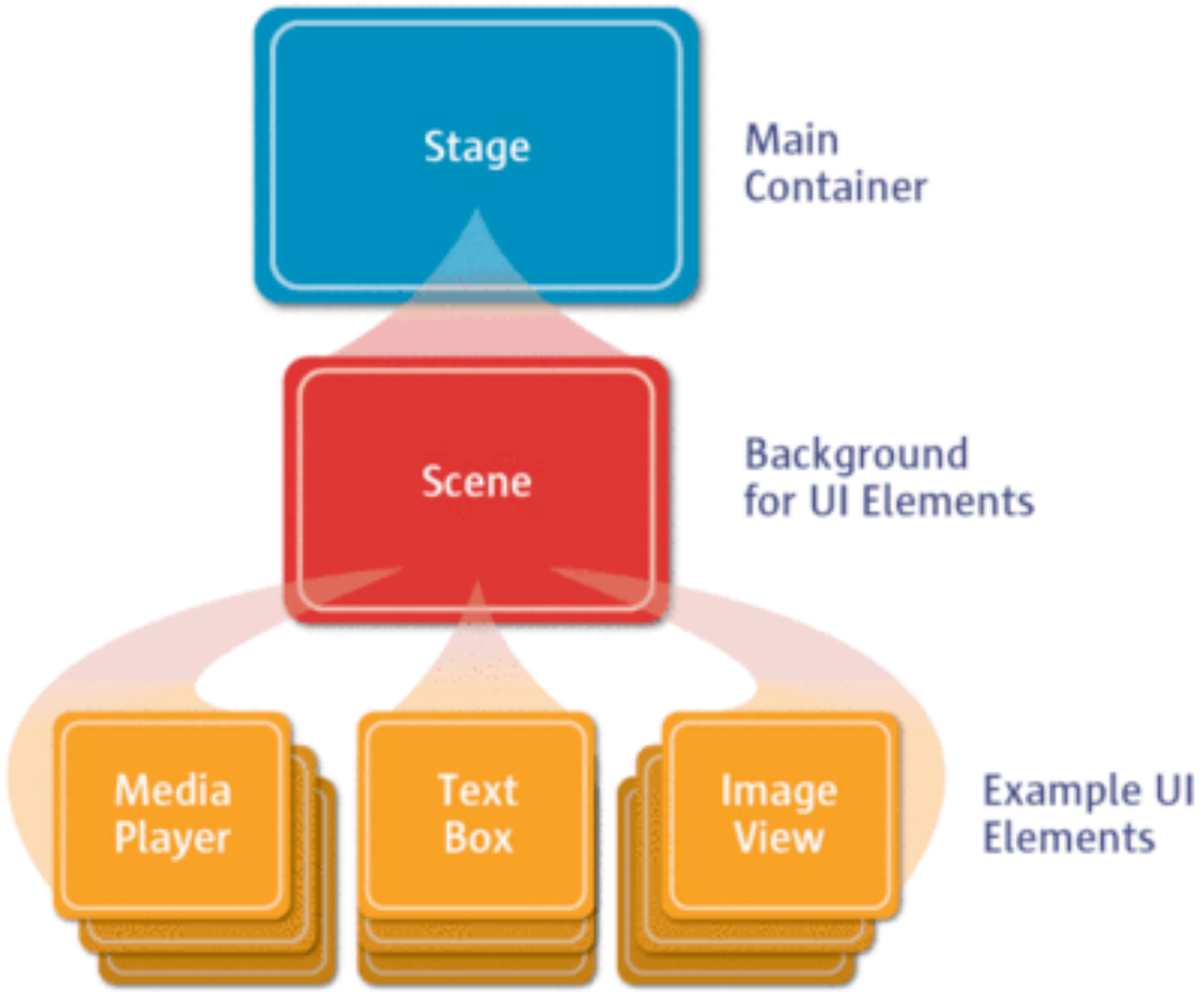
Empfange finale Events: Setzt GUI in End-Status und veranlasst Anzeigen des Punktestands.

JavaFX

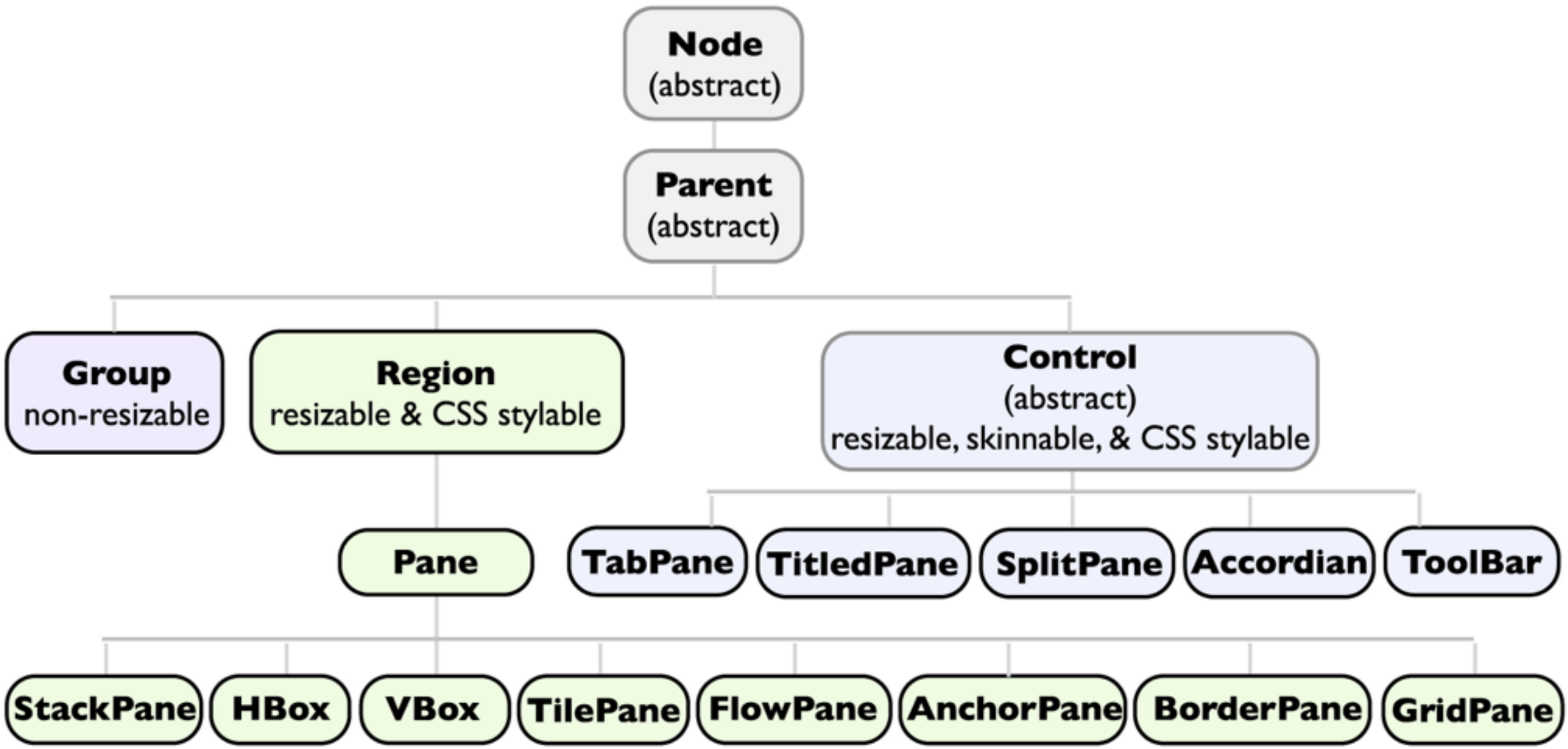
- Deklaratives GUI Framework

- Trennung von Code und Design (FXML)
- Nur enthalten im Oracle JDK!
- Eclipse Integration: e(fx)clipse

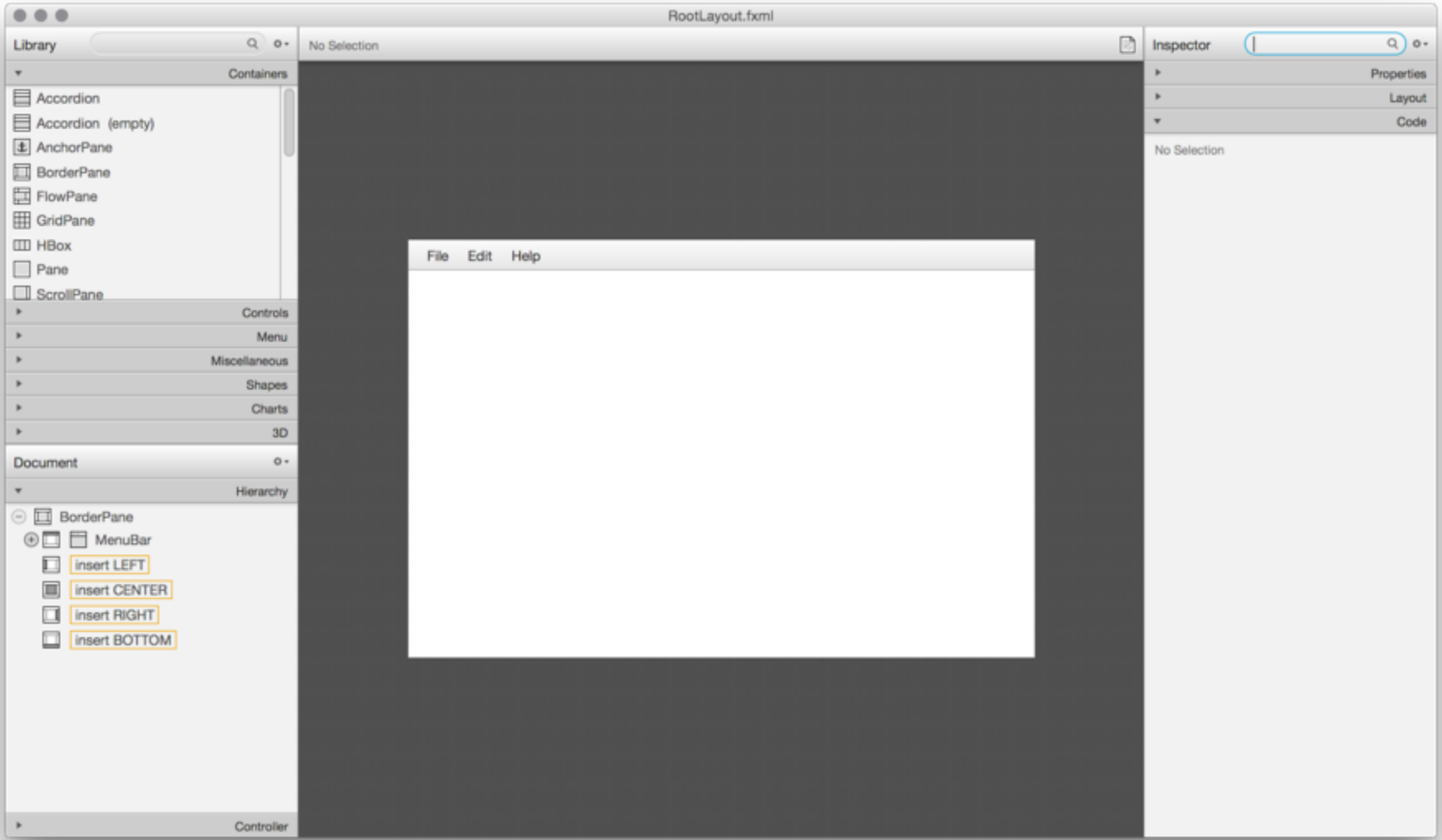
JavaFX UI Elemente



JavaFX UI Elemente



Demo: Scene Builder



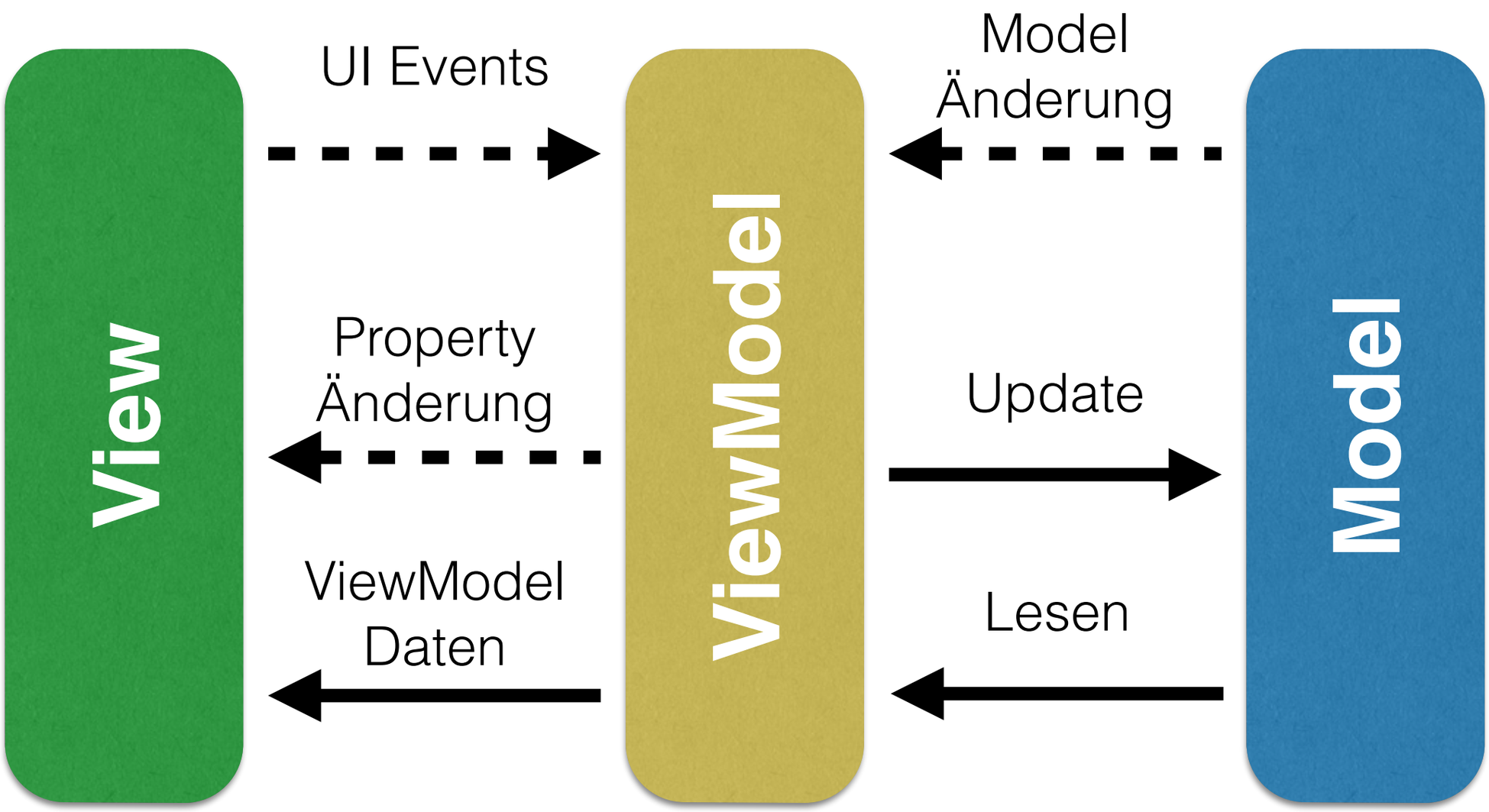
Demo: FXML Verknüpfen

- `@FXML` Annotation für Felder und Methoden
- Automatische Initialisierung
- Manuelle Initialisierung mit `initialize()`

Demo: Data Bindings

- `Property` kapselt `Observable` Werte
- Uni- & Bidirektionale Bindung von Werten
- Vereinfachung von GUI Updates

Model View ViewModel



Nebenläufigkeit

- JavaFX ist **nicht Thread-Safe!**
- Lange Berechnungen und I/O können UI blockieren
- Lösung: Benutzen von `Task` und `Service` Klassen
- UI Änderungen aus anderen Threads mit `Plattform.runLater()`

Demo: Selektoren

Finden von GUI Kompenenten:

- “`.button`”: Komponenten vom Typ `Button`
- “`.myClass`”: Komponenten mit Klasse “class”
- “`#myId`”: Komponenten mit ID “myId”

Demo: CSS Styling

- Ändern der Darstellung von Nodes
- Auswählen von Nodes mit Selektoren
- Verfügbare Eigenschaften: https://docs.oracle.com/javase/8/javafx/api/javafx/scene/doc-files/cssref.html
- **Nützlich:** Scene Builder - CSS Analyzer

Demo: Images und Shapes

- Geometrische Formen in der API
- Stylebar mit CSS
- Verknüpfung mit Events
- Bilder in der API

Nützliche Links

- http://code.makery.ch/library/javafx-8-tutorial/

- http://docs.oracle.com/javase/8/javafx/get-started-tutorial/get_start_apps.htm