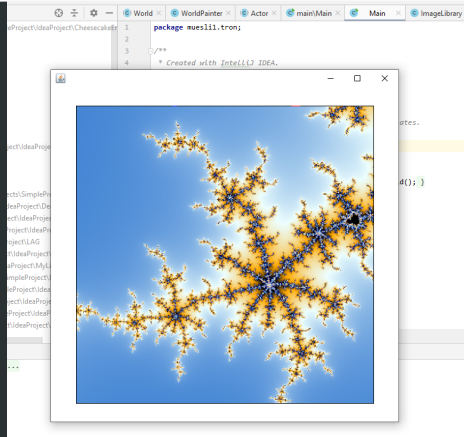


FOP Recap #13

GUIs



Das steht heute auf dem Plan



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Git-Days!

Optional

GUIs

JavaFX-Klassen

Bindings



Git

Git ist ein dezentralisiertes Versionsverwaltungssystem, mit dem man sehr gut und leicht Quellcode verwalten, Änderungen tracken und mit anderen Menschen oder zwischen mehreren eigenen Geräten synchronisieren kann.

Git lohnt sich auch, wenn man nur alleine arbeitet, da es nie schlecht ist, ein Changelog zu haben.

Bei den Git-Days könnt ihr als Studierende jedes Semesters und Studiengangs kostenlos eine betreute Einführung in Git erhalten, nutzt dieses Angebot auf jeden Fall, wenn ihr Zeit habt!

Git-Days!

Infoseite der Fachschaft



TECHNISCHE
UNIVERSITÄT
DARMSTADT

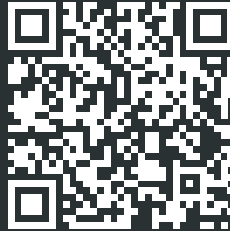


Abbildung: Infoseite für die Git-Days

<https://www.d120.de/de/studierende/git-days/>

Das steht heute auf dem Plan



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Git-Days!

Optional

Grundlagen

Beispiele

GUIs

JavaFX-Klassen

Bindings



- Wrapper-Datentyp, der die An-/Abwesenheit eines Wertes modelliert
- Eingeführt in Java 8
- Vor allem als Rückgabebetyp zu verwenden: Methode gibt Wert zurück oder eben nicht
 - -> Aufrufende Methode muss Behandlung übernehmen
- Sehr gutes Konzept, das auch in vielen anderen Programmiersprachen vorhanden ist:
 - Haskell: `Maybe a`
 - Rust: `Option<T>`
 - C++: `std::optional<T>`
 - ...
 - Implementation natürlich von Sprache abhängig
- Generisch mit einem Typparameter `T`
- Wichtige Methoden:
 - Erstellen von Objekten: `ofNullable`, `empty`, ...
 - Überprüfen, ob das `Optional` leer ist: `isEmpty`, `isPresent`, ...
 - Erhalten des Wertes oder Behandlung der Abwesenheit: `orElseThrow`, `orElse`, ...
 - Bearbeiten des eventuell vorhandenen Wertes: `map`, `filter`, `stream`, ...

Optional

Beispiele



TECHNISCHE
UNIVERSITÄT
DARMSTADT

```
1 <T> Optional<T> findFirst(List<? extends T> list,  
2                           Predicate<? super T> predicate) {  
3     Optional<T> option =  
4     ↪ list.stream().<T>map(Function.identity()).findFirst();  
5     return option.filter(pred);  
6 }
```

```
1 <T> T findFirst(List<? extends T> list,  
2               Predicate<? super T> predicate) {  
3     Optional<T> option =  
4     ↪ list.stream().<T>map(Function.identity()).findFirst();  
5     return option.filter(pred).orElseThrow(() ->  
6         new RuntimeException("Unable to get first element!"));  
7 }
```

Das steht heute auf dem Plan



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Git-Days!

Optional

GUIs

Was ist ein GUI?

Was ist JavaFX?

JavaFX-Grundlagen

JavaFX-Klassen

Bindings

GUIs

Was ist ein GUI?



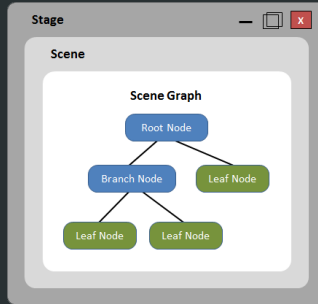
TECHNISCHE
UNIVERSITÄT
DARMSTADT

- Graphical User Interface
- Knöpfe, Bilder, Textboxen, Slider, ...
- Es gibt mehrere Java-Bibliotheken, die das Erstellen von GUIs ermöglichen



- Cross-Plattform Java-Bibliothek zum Erstellen von GUIs, funktioniert u.a. auf:
 - ▣ Linux-Distros
 - ▣ Windows
 - ▣ Apple-Geräte
 - ▣ Android
- Nicht Teil der Standardbibliothek!
- In der FOP nur sehr oberflächlich behandelt, weil sehr komplex

- Hauptklasse: erbt von `Application`
- Stage: „Bühne“ für alles
- Scene: Enthält Graphen an Kindern, der Graoh besteht aus Knoten
- Node: Alle Knoten im Szenengraphen, fast alles ist ein Knoten



GUIs

JavaFX-Grundlagen — Mein erstes JavaFX-Programm



TECHNISCHE
UNIVERSITÄT
DARMSTADT

```
1  public class MainApp extends Application {
2      public static void main(String[] args) {
3          launch(args);
4      }
5      @Override
6      public void start(Stage primaryStage) throws Exception {
7          // set title
8          primaryStage.setTitle("First JavaFX Application");
9          // create layout manager
10         BorderPane root = new BorderPane();
11         // create button
12         Button button = new Button("Hello World!");
13         button.setOnAction(x -> Platform.exit());
14         // add button to layout manager
15         root.setCenter(button);
16         // show frame
17         primaryStage.setScene(new Scene(root));
18         primaryStage.show();
19     }
20 }
```

Das steht heute auf dem Plan



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Git-Days!

Optional

GUIs

JavaFX-Klassen

LayoutManager

JavaFX - Wichtige Elemente des Scene Graphs

Bindings



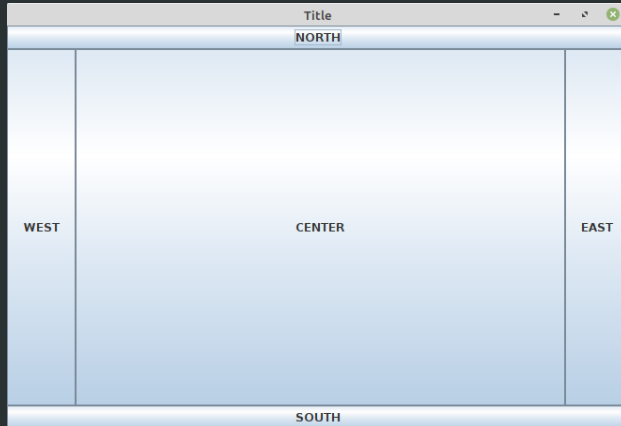
- Legen Position und Größe der verschiedenen Komponenten fest
- Manche erfordern Extra-Parameter beim Verwenden von add
- Häufig genutzte Klassen sind hierbei:
 - ▣ BorderLayout
 - ▣ GridLayout
 - ▣ ...



```
1  BorderPane root = new BorderPane();  
2  
3  root.setCenter(new Button("CENTER"));  
4  root.setTop(new Button("NORTH"));  
5  root.setBottom(new Button("SOUTH"));  
6  root.setRight(new Button("EAST"));  
7  root.setLeft(new Button("WEST"));
```

JavaFX-Klassen

LayoutManager — BorderPane

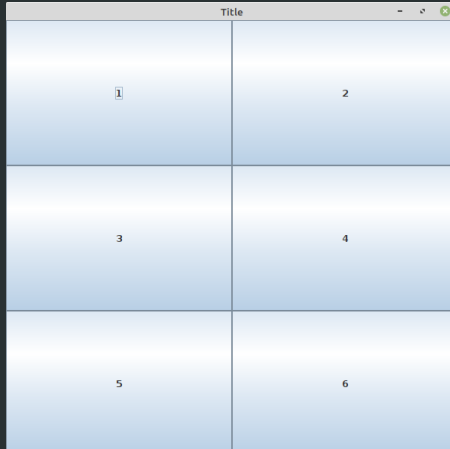




```
1 GridPane root = new GridPane();
2
3 root.add(new Button("1"), 0, 0);
4 root.add(new Button("2"), 1, 0); // column = 1, row = 0
5 root.add(new Button("3"), 2, 0);
6 root.add(new Button("4"), 0, 1);
7 root.add(new Button("5"), 1, 1);
8 root.add(new Button("6"), 2, 1);
```

JavaFX-Klassen

LayoutManager — GridPane





Button beschreibt einen Knopf

- Können Aktion bei Klick ausführen (.setOnAction)
- Können Text oder Bild anzeigen



`Label` beschreibt ein Text-Anzeige-Element

- Kann Text anzeigen
- Kann Textformatierung (z.B. fett) haben
- Kann Textfarbe haben



`TextField` beschreibt ein Text-Eingabe-Element

- Kann Text anzeigen
- Kann Textformatierung (z.B. fett) haben
- Kann Textfarbe haben
- Kann Text ändern
- Text kann über `getText()` ausgelesen werden



Slider beschreibt ein Schieberegler-Element

- Kann Wert anzeigen
- Kann Wert ändern
- Wertebereich kann festgelegt werden
- Kann Wert über `getValue()` ausgelesen werden



CheckBox beschreibt ein Checkbox-Element

- Kann für Ja/Nein-Fragen verwendet werden
- Visualisiert Zustand mit Häkchen
- Zustand kann über `isSelected()` ausgelesen werden



VBox und HBox beschreiben Horizontale und Vertikale Container oder Gruppen

- Elemente per `getChildren().add()` hinzufügen
- Zentrieren per `setAlignment()`

Das steht heute auf dem Plan



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Git-Days!

Optional

GUIs

JavaFX-Klassen

Bindings

Grundlagen

Beispiel



- Bindings sind ein Feature von JavaFX und im Package `javafx.beans` zu finden
- Idee:
 - ▣ Berechne Wert, der von einer oder mehreren Quellen/Abhängigkeiten abhängig ist
 - ▣ Wert wird automatisch geändert, wenn sich Quellwerte ändern (nicht ganz richtig, für unsere Zwecke ausreichend)
 - ▣ Quellen sind in der Regel beobachtbar, umgesetzt durch Interfaces
 - ▣ -> Verknüpfung von Zustand zweier Objekte
- Bindings können unidirektional oder bidirektional sein
- Bindings lassen sich auch verknüpfen und haben dafür sehr viele Methoden
- Hilfsklasse Bindings stellt nochmal sehr viele Hilfsmethoden bereit
-> Vorlesungsfolien/Oracle Docs
- Viele Klassen und Interfaces, zum Beispiel:
 - ▣ Property: Wrapper, der Binding-Funktionalität für sämtlichen Datentypen bereitstellt

Bindings

Beispiel



TECHNISCHE
UNIVERSITÄT
DARMSTADT

```
1  @Override
2  public void start(Stage primaryStage) throws Exception {
3      primaryStage.setTitle("Nutzungsbedingungen und Datenverkauf zustimmen");
4      Button button = new Button("Fortfahren");
5      CheckBox checkBox = new CheckBox("Klick mich oder du darfst nicht fortfahren haha :P");
6      // create binding
7      button.disableProperty().bind(checkBox.selectedProperty().not());
8      // create box for components
9      VBox vBox = new VBox(10, checkBox, button);
10     // set box alignment
11     vBox.alignmentProperty().setValue(Pos.CENTER);
12     // show frame
13     primaryStage.setScene(new Scene(root));
14     primaryStage.show();
15 }
```

Bindings

Beispiel



TECHNISCHE
UNIVERSITÄT
DARMSTADT

☐

Klick mich oder du darfst nicht weitermachen haha :P

Fortfahren



Klick mich oder du darfst nicht weitermachen haha :P

Fortfahren

Live-Coding!