3.– 6. September 2012
in Nürnberg

Herbstcampus

Wissenstransfer
par excellence

Leitern mit Stil

Eine Einführung in ScalaFX

Philipp Dörfler
imbus AG

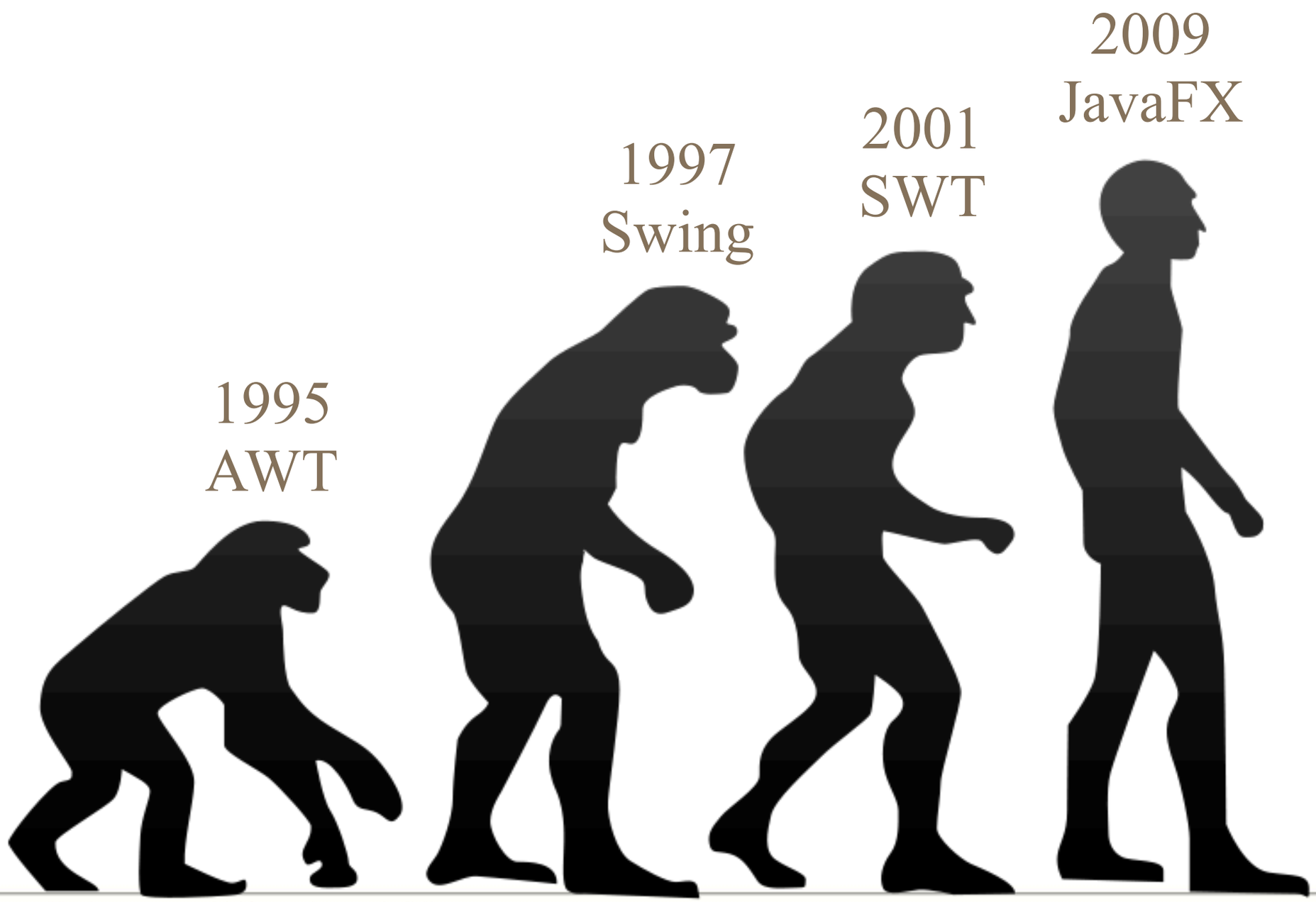
Agenda

- Vorstellung
- JavaFX Status Quo
  - Überblick & Ursprung
  - Tool-Landschaft
  - Polyglot
- ScalaFX
  - Konzept
  - Umsetzung
    - Binding
    - Animation

Vorstellung

- Philipp Dörfler
- 2. Semester Medieninformatik an der „Georg-Simon-Ohm-Hochschule für angewandte Wissenschaften - Fachhochschule Nürnberg“
- Twitter: @phdoerfler
- Bitbucket: phdoerfler
- GitHub: phdoerfler
- Google: phdoerfler

JavaFX oder „Nun wird‘s aber mal Zeit“



http://en.wikipedia.org/wiki/File:Human_evolution.svg

Beispiele

JavaFX - Aufbau

- Scenegraph
  - Einheitliche API
  - Hoher Abstraktionsgrad
  - Simpel, gut optimierbar
- Properties
- Binding
- Animationen einfach möglich
- Styling mit CSS3
- Accessibility: W3C ARIA

JavaFX - Glass Windowing Toolkit

- JavaFX Application Thread
  - Hauptthread
  - Für „live“ Objekte
  - Wichtig: **Nicht** der EDT
- Prism Render Thread
  - N Rendering, N+1 Processing
  - Evtl. Mehrere Rasterisierungs-Threads
- Media Thread

Zielgruppe

- Primär: Enterprise
- Gut mit Swing und SWT vereinbar
  - JavaFX in Swing
  - Swing in JavaFX
- Windows, Linux, Mac OS X, (ARM)
- Ab Java 7u6 dabei
- Corporate Design einfach umsetzbar
- Fallback auf Java2D

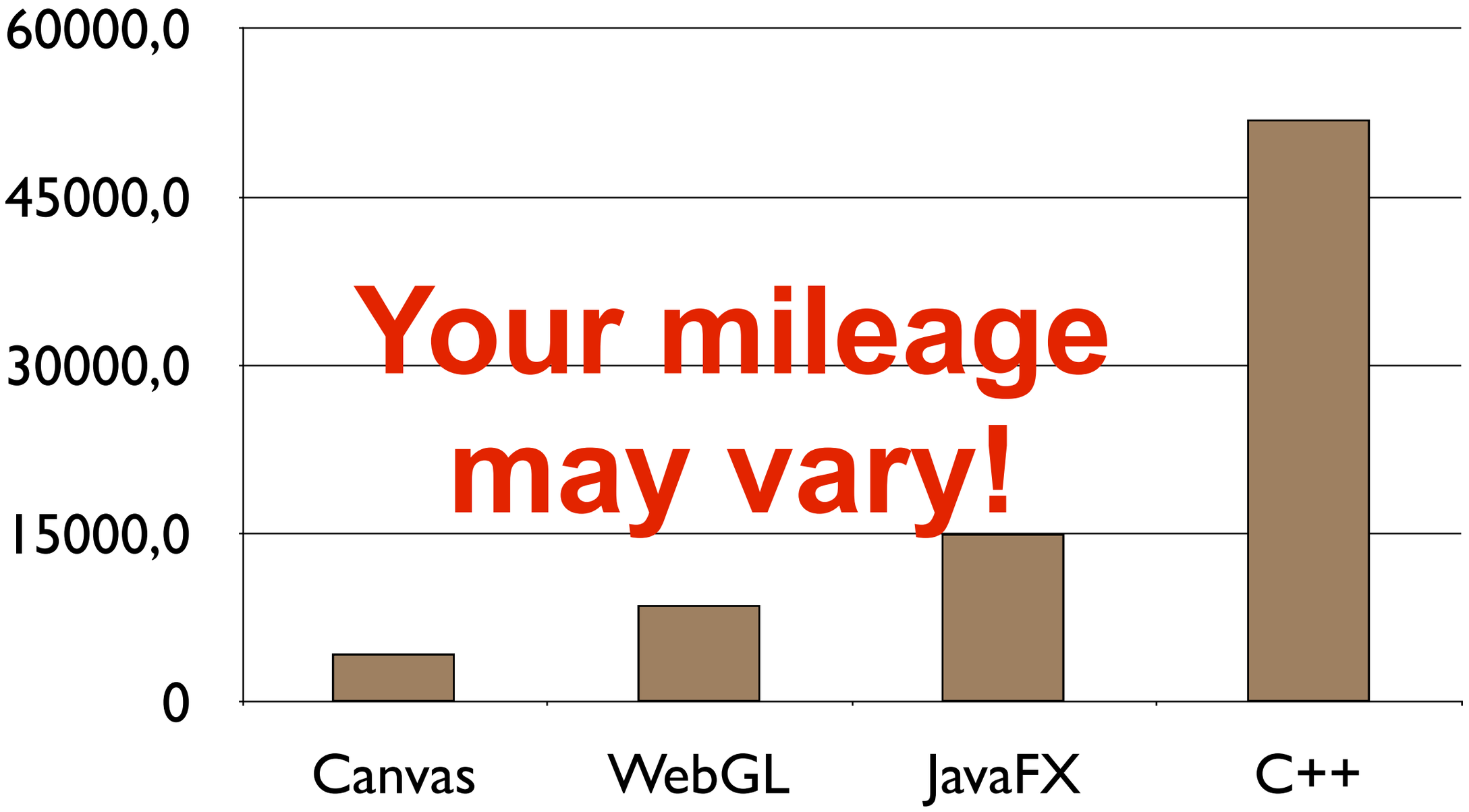
Alternativen?

- Swing
  - Offiziell durch JavaFX abgelöst
  - Animationen schwierig
- Flash
  - End Of Life
- SWT
  - Geringerer Abstraktionsgrad...
  - ...dafür Native Widgets
- HTML5 + JS + CSS
  - Keine JVM
  - Performance schlechter

Performance

- Prism: OpenGL / DirectX 9 / DirectX 11 / Java2D
- Bündelt Events
- Zeichnet nur geänderte Bereiche neu
  - Scissor-Test in OpenGL
- Performance
  - Schneller als HTML 5 Canvas inkl. HW-Accel.
  - Schneller als WebGL
  - ~ 3x langsamer als C++ mit Direct 3D
- Disclaimer: Werte vom 25.11.2011 **ohne** JFX Canvas
  - http://weblogs.java.net/blog/opinali/archive/2011/11/25/javafx-balls-30
- Ausrichtung: „statische“ GUIs und wenig CPU-Zeit

Benchmark, alt (!)



Tool-Landschaft

- Scene Builder
- e(fx)clipse
- svg2fxml
- JavaFX Ensemble
- Scenic View
- JFX Extras + Ensemble
- Netbeans + ANT-Task
- FX Experience Tools
  - Caspian Styler
  - Animation Spline Editor
  - Derived Color

Polyglot

- JavaFX Imperative
- JavaFX Builder
- FXML
- FX.js
- Visage
- GroovyFX
- ScalaFX
- JythonFX
- ~~SmileyFX~~

ScalaFX

```
object HelloFX extends JFXApp {
  stage = new Stage {
    scene = new Scene {
      content = new Label {
        text = "meep!"
      }
    }
  }
}
```

ScalaFX

- Wo: https://code.google.com/p/scalafx/
- Wer: Stephen Chin, Peter Pilgrim
- Status
  - Aktiv
  - Wichtigste Klassen vorhanden
- JavaFX Version: 2.0
  - Baut mit JavaFX 2.2
- Priorität auf Typsicherheit
- Interop vollständig: JFX <=> SFX

ScalaFX - Architektur

- Inspiriert von JavaFX Script
- Implicits für JavaFX <=> ScalaFX
- Delegates (vgl. Scala Swing)
- Weniger Boilerplate für
  - Properties
  - Animationen
  - Binding
- Hierarchie entspricht der von JavaFX
- Optimierung durch @specialize

ScalaFX - Installation

- hg clone ...
- export JAVAFX_HOME=...
- build.sbt anpassen
  - `unmanagedJars in Compile += Attributed.blank(file(System.getenv("JAVAFX_HOME") + "/jre/lib/jfxrt.jar"))`
- Mnemonic.scala: `delegate.setNode(n)`
- sbt publish-local
- `libraryDependencies += "org.scalafx" % "scalafx" % "1.0-SNAPSHOT"`

Properties

- get

```
jfx.getWidth
jfx.widthProperty.get
```

```
sfx.width()
sfx.width.get
```

- set

```
jfx.setWidth(42)
jfx.widthProperty.set(42)
jfx.widthProperty.setValue(42)
```

```
sfx.width = 42
sfx.width() = 42
```

- Property

```
jfx.widthProperty
```

```
sfx.width
```

Properties

- ChangeListener

```
jfx.widthProperty.addListener(new ChangeListener[Number] {
  override def changed(value: ObservableValue[_ <: Number],
      oldValue: Number, newValue: Number): Unit = {
    println("meep")
  }
})
```

```
sfx.width.onChange(println("meep"))
```

Binding

- *Das* Feature von JavaFX Script
- Verknüpft zwei Properties miteinander
- Dort: kompiliert, dadurch sehr schnell
- Hier: Lazy Evaluation, ~ 15% langsamer
- Low-Level für Performance
- High-Level für Abstraktion

```
jfx.widthProperty.bindBidirectional(foo.widthProperty)
```

```
sfx.width <==> foo.width
```

Binding

- when

```
jfx.widthProperty.bind(when(greaterThanOrEqual(
    foo.heightProperty, 200)).then(300).otherwise(400))
```

```
sfx.width <== when (foo.height >= 200) then 300 otherwise 400
```

Binding

static BooleanBinding	greaterThanOrEqual(ObservableNumberValue op1, long op2) Creates a new BooleanBinding that holds true if the value of a ObservableNumberValue is greater than or equal to a constant value.
static BooleanBinding	greaterThanOrEqual(ObservableNumberValue op1, ObservableNumberValue op2) Creates a new BooleanBinding that holds true if the value of the first ObservableNumberValue is greater than or equal to the value of the second.
static BooleanBinding	greaterThanOrEqual(ObservableStringValue op1, ObservableStringValue op2) Creates a new BooleanBinding that holds true if the value of the first ObservableStringValue is greater than or equal to the value of the second.
static BooleanBinding	greaterThanOrEqual(ObservableStringValue op1, java.lang.String op2) Creates a new BooleanBinding that holds true if the value of a ObservableStringValue is greater than or equal to a constant value.
static BooleanBinding	greaterThanOrEqual(java.lang.String op1, ObservableStringValue op2) Creates a new BooleanBinding that holds true if a constant value is greater than or equal to the value of a ObservableStringValue.
static IntegerBinding	integerValueAt(ObservableList<? extends java.lang.Number> op, int index) Creates a new IntegerBinding that contains the element of an ObservableList at the specified position.
static IntegerBinding	integerValueAt(ObservableList<? extends java.lang.Number> op, ObservableIntegerValue index) Creates a new IntegerBinding that contains the element of an ObservableList at the specified position.
static <K> IntegerBinding	integerValueAt(ObservableMap<K,? extends java.lang.Number> op, K key) Creates a new IntegerBinding that contains the mapping of a specific key in an ObservableMap.
static <K> IntegerBinding	integerValueAt(ObservableMap<K,? extends java.lang.Number> op, ObservableValue<? extends K> key) Creates a new IntegerBinding that contains the mapping of a specific key in an ObservableMap.
static <E> BooleanBinding	isEmpty(ObservableList<E> op) Creates a new BooleanBinding that holds true if a given ObservableList is empty.
static <K,V> BooleanBinding	isEmpty(ObservableMap<K,V> op) Creates a new BooleanBinding that holds true if a given ObservableMap is empty.
static <E> BooleanBinding	isEmpty(ObservableSet<E> op) Creates a new BooleanBinding that holds true if a given ObservableSet is empty.
static BooleanBinding	isNotNull(ObservableObjectValue<?> op) Creates a new BooleanBinding that holds true if the value of an ObservableObjectValue is not null.
static BooleanBinding	isNull(ObservableObjectValue<?> op) Creates a new BooleanBinding that holds true if the value of an ObservableObjectValue is null.
static BooleanBinding	lessThan(double op1, ObservableNumberValue op2) Creates a new BooleanBinding that holds true if a constant value is less than the value of a ObservableNumberValue.
static BooleanBinding	lessThan(float op1, ObservableNumberValue op2) Creates a new BooleanBinding that holds true if a constant value is less than the value of a ObservableNumberValue.

Binding

- `sfx.width <== foo.width + 42`
- `sfx.visible <== foo.visible and foo.width < 20`
- `sfx.width <== max(foo.height, foo.width)`
- `sfx.fill <== when(foo.height === 42) then Color.AZURE otherwise Color.CHOCOLATE`
- Tückisch: `sfx.disable <== stillLoading && foo.disable`
- Nicht: `sfx.disable <== foo.disable && stillLoading`
- `sfx.visible <== foo.height === 43+-3`

Animationen

- Keyframes
- Operator-Overloading
- Interpolation ähnlich wie JavaFX Script

```
new Timeline(60) {
  keyFrames = Seq(
    at(0 s) { sfx.width -> 200 },
    at(10 s) { sfx.width -> 400 tween DISCRETE }
  )
}.play


sfx.onMouseClicked = {
  Timeline(at(3 s) { sfx.height -> 42 }).play
}
```

Zum Nachschlagen

- Pro JavaFX 2, Apress
- JavaFX 2.0 - Introduction by Example, Apress
- https://code.google.com/p/scalafx/
- http://steveonjava.com
- http://www.javafx.com
- http://fxexperience.com
- http://efxclipse.org
- http://jfxtras.org

Vielen Dank!

Philipp Dörfler
imbus AG