Scala (Programmiersprache)

Ein Teaser und allgemeinere Gedanken

Sebastian Eidecker

16. März 2016

sieht in jedem Problem einen Nagel.

Paul Watzlawick

Wer als Werkzeug nur einen Hammer hat,

Worüber reden wir? IT im Wandel

Worüber reden wir? IT im Wandel

Herausforderungen

Manifeste

IT im Wandel

Herausforderungen

IT im Wandel Herausforderungen

Manifeste
Scala

IT im Wandel

Herausforderungen

Manifeste

Scala

Management Summary

IT im Wandel

Herausforderungen

Manifeste

Scala

Management Summary

Ein wenig Code

IT im Wandel

Herausforderungen

Manifeste

Scala

Management Summary

Ein wenig Code

Spannendes

IT im Wandel

Herausforderungen

Software Engineering

Software Engineering

· Stabilität und Resilienz





- · Stabilität und Resilienz
- Wertbeitrag

- · Stabilität und Resilienz
- Wertbeitrag
- Businesstreiber

- · Stabilität und Resilienz
- Wertbeitrag
- Businesstreiber

Matthias Magnor – CEO Surface und Contract Logistics

IT im Wandel

Manifeste

 Antwortbereit, Widerstandsfähig, Elastisch, Nachrichtenorientiert (2013)

• Gut gefertigt, Stets Mehrwert, Gemeinschaft aus Experten, Produktive Partnerschaften (2009)

 Individuen und Interaktionen, Funktionierende Software, Zusammenarbeit mit dem Kunden, Reagieren auf Veränderung (2001)

- Antwortbereit, Widerstandsfähig, Elastisch, Nachrichtenorientiert (2013)
- Gut gefertigt, Stets Mehrwert, Gemeinschaft aus Experten, Produktive Partnerschaften (2009)
- Individuen und Interaktionen, Funktionierende Software, Zusammenarbeit mit dem Kunden, Reagieren auf Veränderung (2001)

Wo stehen wir?

Wo stehen wir im Wettbewerb?

Scala

Management Summary

Scalable Language



Objektorientiert

- Objektorientiert
- Funktional

- Objektorientiert
- Funktional
- Statisch typisiert

- Objektorientiert
- Funktional
- Statisch typisiert
- Gewohnte Syntax ("Java ohne Semikolon")

- Objektorientiert
- Funktional
- Statisch typisiert
- Gewohnte Syntax ("Java ohne Semikolon")
- Ausdrucksstark (APIs/DSLs schreiben)

- Objektorientiert
- Funktional
- Statisch typisiert
- Gewohnte Syntax ("Java ohne Semikolon")
- Ausdrucksstark (APIs/DSLs schreiben)
- Immutable Objects als Konzept

- Objektorientiert
- Funktional
- Statisch typisiert
- Gewohnte Syntax ("Java ohne Semikolon")
- Ausdrucksstark (APIs/DSLs schreiben)
- Immutable Objects als Konzept
- Jung (2004, Hype gegen 2011)



Versprechen

Produktivitätssteierung

- Produktivitätssteierung
- · Höhere Codequalität

- Produktivitätssteierung
- · Höhere Codequalität
- durch
- Weniger Code

- Produktivitätssteierung
- · Höhere Codequalität
- durch
- · Weniger Code
- · Höheres Abstraktionsniveau

- Produktivitätssteierung
- · Höhere Codequalität
- durch
- Weniger Code
- · Höheres Abstraktionsniveau
- Mehr Spaß

• Java-Bytecode, läuft auf JVM

Scala und die Java-Plattform

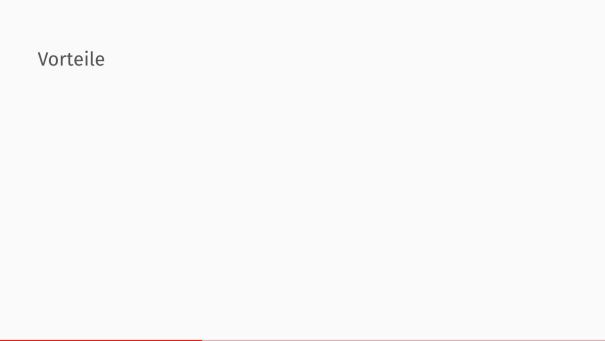
· Java-Bytecode, läuft auf JVM

· Java-Bibliotheken nutzbar

- · Java-Bytecode, läuft auf JVM
- · Java-Bibliotheken nutzbar

Bekannte IDFs

- · Java-Bytecode, läuft auf JVM
- · Iava-Bibliotheken nutzbar
- Bekannte IDFs
- · Ähnlicher Paketierungs- und Buildprozess (sbt)



Für moderne
 Architekturen geeignet

- Für moderne
 Architekturen geeignet
- · Verständlich funktional

- Für moderne
 Architekturen geeignet
- Verständlich funktional
- · Java-Ökosystem

- Für moderne
 Architekturen geeignet
- Verständlich funktional
- Java-Ökosystem
- · Macht Spaß

- Für moderne
 Architekturen geeignet
- Verständlich funktional
- Java-Ökosystem
- Macht Spaß
- Statisch typisiert

- Für moderne Architekturen geeignet
- · Verständlich funktional
- Java-Ökosystem
- Macht Spaß
- Statisch typisiert

- Für moderne Architekturen geeignet
- · Verständlich funktional
- Java-Ökosystem
- Macht Spaß
- Statisch typisiert

Nachteile

Komplex

- Für moderne Architekturen geeignet
- Verständlich funktional
- Java-Ökosystem
- Macht Spaß
- Statisch typisiert

- Komplex
- · Zukunftssicher?

- Für moderne Architekturen geeignet
- Verständlich funktional
- Java-Ökosystem
- Macht Spaß
- Statisch typisiert

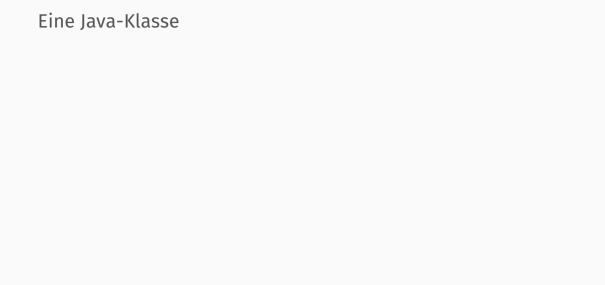
- Komplex
- · Zukunftssicher?
- Zahl an Entwicklungssklaven

- Für moderne Architekturen geeignet
- Verständlich funktional
- Java-Ökosystem
- Macht Spaß
- Statisch typisiert

- Komplex
- · Zukunftssicher?
- Zahl an Entwicklungssklaven
- Binärkompatibilität nicht in alle Ewigkeit

Scala

Ein wenig Code



Eine Java-Klasse

5

7

8

10

11

12 13

```
public class Person {
  private final String firstName;
  private final String lastName:
  public Person(String firstName, String lastName) {
      this.firstName = firstName:
      this.lastName = lastName:
  public String getFirstName() {
      return firstName:
  public String getLastName() {
      return lastName:
```

Eine Java-Klasse

5

8

9

10

11

12 13

```
aOverride
public boolean equals(Object o) {
    if (this == 0) return true:
    if (o == null || getClass() != o.getClass()) return false;
    Person person = (Person) o:
    if (firstName != null ?
        !firstName.equals(person.firstName) :
        person.firstName != null) return false:
    if (lastName != null ?
        !lastName.equals(person.lastName) :
        person.lastName != null) return false:
    return true:
```

Eine Java-Klasse

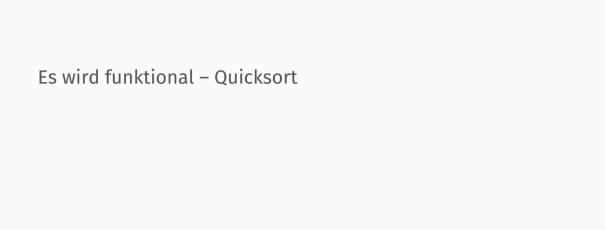
```
@Override
public int hashCode() {
    int result = firstName != null ? firstName.hashCode() : 0;
    result =
        31 * result + (lastName != null ? lastName.hashCode() :
            0);
    return result;
}
```

Businesslogik?



Dasselbe in Scala

case class Person(firstName:String, lastName:String)



Es wird funktional – Quicksort

Schnelldurchlauf Scala – Endlich!

Klassen und Objekte

Klassen und Objekte

class

val und var – Immutables

val und var – Immutables

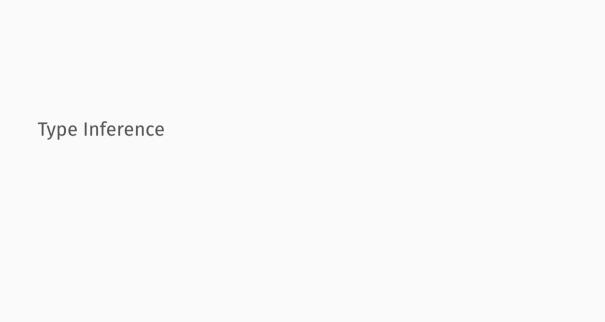
class



Listen

Pattern Matching

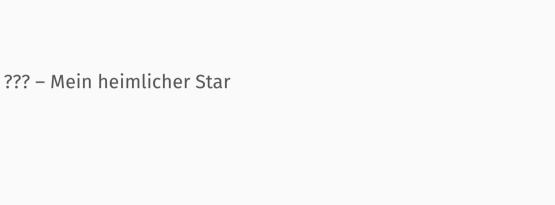
Pattern Matching



Type Inference

Alles hat einen Typ – aber kein return

Alles hat einen Typ – aber kein return



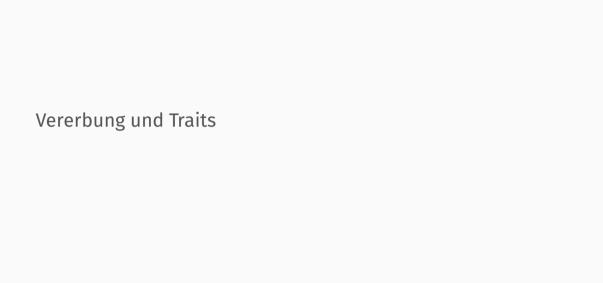
??? – Mein heimlicher Star

def quickSort[A <% Ordered[A]](xs: List[A]): List[A] = ???</pre>

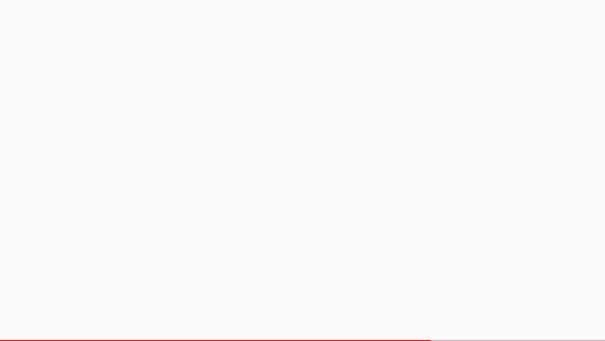
??? – Mein heimlicher Star

Kompilierbar, aber nicht gefährlich.

def quickSort[A <% Ordered[A]](xs: List[A]): List[A] = ???</pre>



Vererbung und Traits



Funktionen – Benannte Parameter

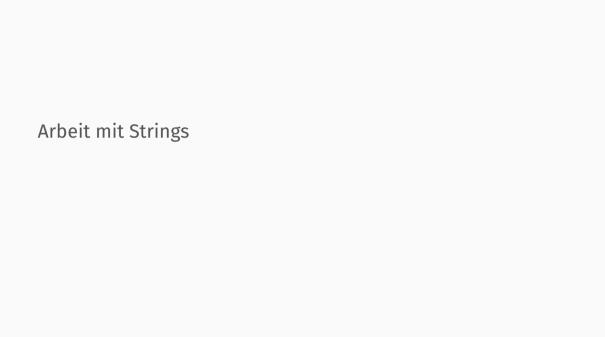
Funktionen – Benannte Parameter

Funktionen funktional – Lambdas schön

Flatmap that shit!

class

Flatmap that shit!



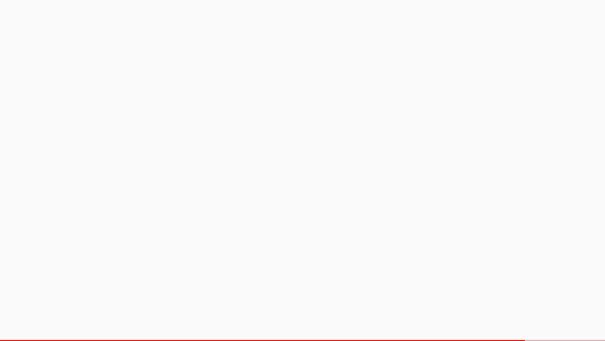
Arbeit mit Strings



Tupel class



Implicits class



Scala

Spannendes



Akka

Scalable real-time transaction processing

Akka

- Scalable real-time transaction processing
- · Will die aktuellen Probleme lösen

Akka

- Scalable real-time transaction processing
- · Will die aktuellen Probleme lösen





ScalaTest . .

ScalaTest			

Scala

Mehr für Nerds



Mehr für Nerds

Sprecht mich an









Mehr für Nerds

- Sprecht mich an

- · Hands on-Termin bei Interesse

Mehr für Nerds

- Sprecht mich an
- · Hands on-Termin bei Interesse
- Heiko Seeberger: "Durchstarten mit Scala. Tutorial für Einsteiger (2. Aufl.)"

