Inhaltsverzeichnis

Vorwort 9

1 Warum Scala? 11

1.1 Was ist Scala? 11
1.2 Warum Scala statt Java? 12
1.3 Warum Scala statt Groovy, Clojure und Co? 16

2 Entwicklungsumgebung 17

2.1 Kommandozeilenwerkzeuge 17
2.1.1 scalac 18
2.1.2 scala 19
2.1.3 scaladoc 20
2.2 sbt 21
2.3 IDE 24

3 Das Fallbeispiel ScalaTrain 27

4 Erste Gehversuche in der REPL 29

4.1 Variablen 29
4.1.1 Unveränderliche Variablen 29
4.1.2 Veränderliche Variablen 31
4.2 Methoden 31
4.2.1 Alles hat ein Ergebnis 32
4.2.2 Unit-Methoden 33
4.3 Funktionen 34

5 Grundlagen der Objektorientierung 37

5.1 Vorbereitung: Projekt initialisieren 37
5.2 Klassen 39
5.2.1 Klassenparameter und Konstruktoren 39
5.2.2 Felder 41
5.2.3 Methoden 44
5.2.4 Benannte Argumente und Standardwerte 48

Bibliografische Informationen
http://d-nb.info/1076877605

digitalisiert durch DEUTSCHE NATIONAL BIBLIOTHEK

5.3 Pakete und Sichtbarkeit 49
5.3.1 Verschachtelte Pakete 49
5.3.2 Imports 51
5.3.3 Sichtbarkeit 53
5.4 Singleton Objects 53
5.4.1 Companion Objects 54
5.4.2 Predef 54
5.5 Case Classes 55
5.6 Projektcode: aktueller Stand 58

6 Testen von Scala-Programmen 59

6.1 Testabdeckung mit scoverage 59
6.2 Unit Tests mit ScalaTest 60
6.2.1 WordSpec 61
6.2.2 Matcher 62
6.3 Testdaten mit ScalaCheck 64
6.4 Projektcode: aktueller Stand 67

7 Erste Schritte mit der funktionalen Programmierung 69

7.1 Scala Collections 69
7.1.1 Klassenhierarchie 70
7.1.2 Collection-Instanzen erzeugen 71
7.1.3 Typparameter 72
7.1.4 Tupel 73
7.1.5 Unveränderliche und veränderliche Collections 74
7.1.6 Collections in ScalaTrain 76
7.2 Funktionale Collections 77
7.2.1 Funktionsliterale 77
7.2.2 Funktionstypen 79
7.2.3 Funktionale Collections in ScalaTrain 81
7.2.4 „map“, „flatMap“ und „filter“ im Detail 84
7.3 For-Ausdrücke und For-Schleifen 87
7.3.1 For-Ausdrücke 89
7.3.2 For-Schleifen und foreach 92
7.4 Projektcode: aktueller Stand 93

8 Vererbung und Traits **95**

8.1 Vererbung 95
8.1.1 Unterklassen mit „extends" definieren 95
8.1.2 Felder und Methoden überschreiben 97
8.1.3 Abstrakte Klassen 99
8.1.4 Scala-Typhierarchie 103
8.2 Traits 105
8.2.1 Traits hineinmixen 106
8.2.2 Linearisierung 108
8.2.3 Beispiel: „Ordered" implementieren 110
8.2.4 Einschub: By-Name Parameters 112
8.3 Projektcode: aktueller Stand 116

9 Pattern Matching **119**

9.1 Match-Ausdrücke 119
9.2 Welche Patterns gibt es? 120
9.2.1 Wildcard Pattern 120
9.2.2 Constant Pattern 120
9.2.3 Variable Pattern und Typed Pattern 121
9.2.4 Tuple Pattern 121
9.2.5 Constructor Pattern 121
9.2.6 Sequence Pattern 122
9.3 Pattern Guards und Variable Binding 123
9.4 Pattern Matching außerhalb von Match-Ausdrücken 124
9.5 Projektcode: aktueller Stand 126

10 Implicits **129**

10.1 Implicit Conversions 129
10.1.1 Implicit Conversions zum Expected Type 130
10.1.2 Implicit Conversions des Receivers 134
10.2 Implicit Classes 134
10.3 Implicit Parameters 136
10.4 Type Classes 139
10.5 Projektcode: aktueller Stand 142

11 Zielbahnhof – Fortgeschrittene Konzepte **145**

11.1 Rekursion 145
11.2 Upper Bounds und Context Bounds 148
11.2.1 Einschub: Package Objects 148
11.2.2 Einschub: Varianz 149
11.2.3 Upper Bounds 150
11.2.4 Context Bounds 151
11.3 Vertiefung objektfunktionale Programmierung 153
11.3.1 Problemstellung 153
11.3.2 Lösungsansatz 154
11.3.3 Etappen und Teilstrecken 156
11.3.4 Verbindungen ermitteln 158
11.4 Projektcode: aktueller Stand 160

12 Scala-Bibliotheken **163**

12.1 Validieren mit Scalactic 163
12.2 Akka HTTP 168
12.3 Projektcode: finaler Stand 174

Stichwortverzeichnis **179**