mitp
Elisabeth Jung
Java 9
Das Übungsbuch
Über 200 Aufgaben mit vollständigen Lösungen

Inhaltsverzeichnis

	Einleitung	25
1	**Klassendefinition und Objektinstanziierung**	33
1.1	Klassen und Objekte	33
	☆ Aufgabe 1.1: Definition einer Klasse	36
	☆ Aufgabe 1.2: Objekt (Instanz) einer Klasse erzeugen	37
1.2	Die Member einer Klasse: Felder und Methoden	37
	☆ Aufgabe 1.3: Zugriff auf Felder	38
	☆☆ Aufgabe 1.4: Aufruf von Methoden	38
1.3	Das Überladen von Methoden	39
	☆ Aufgabe 1.5: Eine Methode überladen	39
1.4	Die Datenkapselung, ein Prinzip der objektorientierten Programmierung	40
	☆ Aufgabe 1.6: Zugriffsmethoden	40
1.5	Das »aktuelle Objekt« und die »this-Referenz«	40
	☆☆ Aufgabe 1.7: Konstruktordefinitionen	41
1.6	Die Wert- und Referenzübergabe in Methodenaufrufen	41
	☆☆☆ Aufgabe 1.8: Wertübergabe in Methoden (»call by value«)	42
1.7	Globale und lokale Referenzen	42
	☆☆ Aufgabe 1.9: Der Umgang mit Referenzen	42
	☆☆☆ Aufgabe 1.10: Wiederholungsaufgabe	43
1.8	Selbstreferenzierende Klassen und Felder (»self-referential classes and fields«)	44
	☆☆☆ Aufgabe 1.11: Der Einsatz von selbstreferenzierenden Feldern	44
1.9	Java-Pakete	45
	☆ Aufgabe 1.12: Die package-Anweisung	46
	☆ Aufgabe 1.13: Die import-Anweisung	47
1.10	Die Modifikatoren für Felder und Methoden in Zusammenhang mit der Definition von Paketen	47
	☆☆ Aufgabe 1.14: Pakete und die Sichtbarkeit von Membern einer Klasse	47

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

1.11 Standard-Klassen von Java 48
☆☆☆ Aufgabe 1.15: Aufruf von Methoden der Klasse Math . . . 50
☆☆ Aufgabe 1.16: Wiederholungsaufgabe 51
1.12 Die Wrapper-Klassen von Java und das Auto(un)boxing 52
☆☆ Aufgabe 1.17: Die Felder und Methoden von Wrapper-Klassen 53
☆ Aufgabe 1.18: Das Auto(un)boxing 53
1.13 Arrays (Reihungen) und die Klassen Array und Arrays 54
☆☆ Aufgabe 1.19: Der Umgang mit Array-Objekten 55
1.14 Zeichenketten und die Klasse String 56
☆ Aufgabe 1.20: Der Umgang mit String-Objekten. 56
1.15 Mit der Version 5.0 eingeführte Spracherneuerungen für Arrays und Methoden 56
☆ Aufgabe 1.21: Einfache und erweiterte for-Schleifen . . . 57
☆ Aufgabe 1.22: Methoden mit variablen Argumentenlisten 58
1.16 Das Initialisieren von Klassen- und Instanzfeldern 58
☆ Aufgabe 1.23: Das Initialisieren von Instanzfeldern . . . 59
☆ Aufgabe 1.24: Das Initialisieren von Klassenfeldern . . . 59
1.17 Private Konstruktoren 59
☆ Aufgabe 1.25: Ein Objekt mit Hilfe eines privaten Konstruktoren erzeugen 60
☆☆ Aufgabe 1.26: Mehrere konstante Werte (Objekte) mit Hilfe eines privaten Konstruktoren erzeugen 60
1.18 Lösungen 61
Lösung 1.1 61
Lösung 1.2 61
Lösung 1.3 61
Lösung 1.4 63
Lösung 1.5 65
Lösung 1.6 66
Lösung 1.7 67
Lösung 1.8 69
Lösung 1.9 71
Lösung 1.10 73
Lösung 1.11. 78
Lösung 1.12 80
Lösung 1.13. 80
Lösung 1.14 81

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

Lösung 1.15 82
Lösung 1.16 84
Lösung 1.17 87
Lösung 1.18 89
Lösung 1.19 92
Lösung 1.20 95
Lösung 1.21 97
Lösung 1.22 98
Lösung 1.23 100
Lösung 1.24 102
Lösung 1.25 103
Lösung 1.26 104

2 Abgeleitete Klassen und Vererbung 107
2.1 Abgeleitete Klassen 107
2.2 Die Konstruktoren von abgeleiteten Klassen 107
2.3 Abgeleitete Klassen und die Sichtbarkeit von Feldern und Methoden 107
☆☆ Aufgabe 2.1: Test von Sichtbarkeitsebenen 108
2.4 Das Verdecken von Klassenmethoden und das statische Binden von Methoden 109
☆☆ Aufgabe 2.2: Der Aufruf von verdeckten Klassenmethoden 110
2.5 Das Überschreiben von Instanzmethoden und das dynamische Binden von Methoden 110
☆ Aufgabe 2.3: Das dynamische Binden von Methoden . . . 111
2.6 Vererbung und Komposition 111
☆ Aufgabe 2.4: Die Komposition 112
☆ Aufgabe 2.5: Die Vererbung 112
2.7 Kovariante Rückgabetypen in Methoden 113
☆☆ Aufgabe 2.6: Die Benutzung von kovarianten Rückgabetypen 113
2.8 Verdeckte Felder 114
☆☆☆ Aufgabe 2.7: Wiederholungsaufgabe 114
2.9 Vergrößernde und verkleinernde Konvertierung (»up- und down-casting«) 116
☆☆☆ Aufgabe 2.8: Up- und Down-Casts 116
☆☆ Aufgabe 2.9: Der Unterschied zwischen »ist-ein-« und »hat-ein- Beziehungen« 117

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

2.10 Der Polymorphismus, ein Prinzip der objektorientierten Programmierung ... 118
☆☆☆ Aufgabe 2.10: Der »Subtyp-Polymorphismus« im Kontext einer Klassenhierarchie ... 119
2.11 Lösungen ... 120
Lösung 2.1 ... 120
Lösung 2.2 ... 123
Lösung 2.3 ... 125
Lösung 2.4 ... 127
Lösung 2.5 ... 129
Lösung 2.6 ... 130
Lösung 2.7 ... 133
Lösung 2.8 ... 137
Lösung 2.9 ... 141
Lösung 2.10 ... 144

3 Abstrakte Klassen und Interfaces ... 147
3.1 Abstrakte Klassen ... 147
3.2 Abstrakte Java-Standard-Klassen und eigene Definitionen von abstrakten Klassen ... 147
☆ Aufgabe 3.1: Die abstrakte Klasse Number und ihre Unterklassen ... 147
☆ Aufgabe 3.2: Definition einer eigenen abstrakten Klasse ... 148
3.3 Die Referenzen vom Typ einer abstrakten Klassen ... 149
☆ Aufgabe 3.3: Der Subtyp-Polymorphismus für Methoden im Kontext einer Klassenhierarchie mit abstrakten Klassendefinitionen ... 149
☆ Aufgabe 3.4: Der Subtyp-Polymorphismus für Felder im Kontext einer Klassenhierarchie mit abstrakten Klassendefinitionen ... 149
3.4 Interfaces (Schnittstellen) ... 150
☆☆ Aufgabe 3.5: Die Definition eines Interface ... 150
3.5 Die Entscheidung zwischen abstrakten Klassen und Interfaces ... 151
☆☆☆ Aufgabe 3.6: Paralleler Einsatz von Interfaces und abstrakten Klassen ... 152
3.6 Oberinterfaces ... 153
☆☆ Aufgabe 3.7: Das Ableiten von Interfaces ... 153
3.7 Implementieren von mehreren Interfaces für eine Klasse ... 154
☆☆ Aufgabe 3.8: Wiederholungsaufgabe ... 154

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

3.8 Die Vererbung an Beispielen von Java-Standard-Klassen und Standard-Interfaces ... 155
3.9 Das Klonen von Objekten ... 156
☆ Aufgabe 3.9: Das Klonen von Instanzen der eigenen Klasse ... 156
☆ Aufgabe 3.10: Das Klonen von Instanzen anderer Klassen ... 157
☆☆ Aufgabe 3.11: Das Klonen von Arrays ... 157
☆ Aufgabe 3.12: Das Überschreiben der clone()-Methode in Java 5.0 ... 157
3.10 Die Gleichheit von Objekten ... 158
☆ Aufgabe 3.13: Die Gleichheit von geklonten Objekten ... 158
3.11 Das oberflächliche und das tiefe Klonen (»shallow und deep cloning«) ... 159
☆ Aufgabe 3.14: Das Klonen und der Copy-Konstruktor ... 159
☆☆☆ Aufagbe 3.15: Tiefes Klonen am Beispiel von Array-Objekten ... 160
☆☆☆ Aufgabe 3.16: Oberflächliches und tiefes Klonen für Referenztypen ... 160
3.12 Der Garbage Collector und das Beseitigen von Objekten ... 161
☆ Aufgabe 3.17: Das Zerstören von Instanzen ... 161
3.13 Lösungen ... 162
Lösung 3.1 ... 162
Lösung 3.2 ... 163
Lösung 3.3 ... 164
Lösung 3.4 ... 165
Lösung 3.5 ... 167
Lösung 3.6 ... 169
Lösung 3.7 ... 172
Lösung 3.8 ... 173
Lösung 3.9 ... 177
Lösung 3.10 ... 178
Lösung 3.11 ... 178
Lösung 3.12 ... 179
Lösung 3.13 ... 180
Lösung 3.14 ... 181
Lösung 3.15 ... 183
Lösung 3.16 ... 185
Lösung 3.17 ... 187

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

4 Einführung in die graphische Programmierung ... 189
4.1 Das AWT (Abstract Windowing Toolkit) und Swing ... 189
4.2 Fenster unter graphischen Oberflächen ... 193
4.3 Die Klassen Graphics und Graphics2D ... 193
4.4 Methoden zum Zeichnen ... 194
☆ Aufgabe 4.1: Eine einfache AWT-Komponente vom Typ Frame ... 197
☆ Aufgabe 4.2: Eine Einfache Swing-Komponente vom Typ JFrame ... 197
☆ Aufgabe 4.3: Ein JWindow-Fenster ... 198
☆ Aufgabe 4.4: Ein JDialog-Fenster ... 198
☆☆ Aufgabe 4.5: Das Neuzeichnen einer Swing-Komponente ohne Benutzung des Clip-Rectangles ... 198
☆☆ Aufgabe 4.6: Das Neuzeichnen einer Swing-Komponente mit Benutzung des Clip-Rectangles ... 199
☆☆ Aufgabe 4.7: Das Parametrisieren der paint()-Methode ... 199
☆ Aufgabe 4.8: Die Instanzen der Klasse Color ... 200
4.5 Die Transparenz-Eigenschaft und der Hintergrund von Komponenten ... 200
4.6 Layout-Manager ... 201
☆ Aufgabe 4.9: Die vordefinierten Layout-Manager für Standard-Klassen ... 203
☆☆☆ Aufgabe 4.10: Die Layout-Manager von AWT-Komponenten ... 203
☆☆☆ Aufgabe 4.11: Die Layout-Manager von Swing-Komponenten ... 204
☆☆ Aufgabe 4.12: Das BoxLayout ... 204
☆☆☆ Aufgabe 4.13: Das GridBagLayout ... 205
☆☆ Aufgabe 4.14: Das null-Layout ... 206
4.7 Das Überlappen von Komponenten ... 206
☆ Aufgabe 4.15: Das Verhalten von AWT-LW- und -HW-Komponenten ... 207
☆☆ Aufgabe 4.16: AWT-Container mit LW- und HW-Kindkomponenten ... 207
☆☆☆ Aufgabe 4.17: Z-Order-Index für AWT-LW-Komponenten ... 208
☆ Aufgabe 4.18: Das Verhalten von Swing-Komponenten ... 208
☆☆☆ Aufgabe 4.19: Swing-Container und der Z-Order-Index für Swing-Komponenten ... 209

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

4.8 Das System als Auslöser für Zeichenoperationen 209
☆☆ Aufgabe 4.20: Resizing von AWT-Komponenten 210
☆☆ Aufgabe 4.21: Resizing von Swing-Komponenten 210
4.9 Eventbehandlung 211
4.10 Events auf niedriger Ebene 212
☆ Aufgabe 4.22: Das Interface WindowListener und die Klasse WindowEvent 212
☆ Aufgabe 4.23: Die Klasse WindowAdapter 212
4.11 Events auf höherer Ebene 213
☆ Aufgabe 4.24: Das Interface ActionListener und die Klasse ActionEvent 213
☆ Aufgabe 4.25: Das Interface KeyListener und die Klasse KeyEvent 213
4.12 Das Delegationsmodell in der Eventbehandlung 214
☆☆ Aufgabe 4.26: Die Ereignisbehandlung in einer separaten Klasse definieren 214
4.13 Lösungen 215
Lösung 4.1 215
Lösung 4.2 216
Lösung 4.3 217
Lösung 4.4 218
Lösung 4.5 219
Lösung 4.6 220
Lösung 4.7 222
Lösung 4.8 223
Lösung 4.9 225
Lösung 4.10 226
Lösung 4.11 229
Lösung 4.12 232
Lösung 4.13 233
Lösung 4.14 235
Lösung 4.15 236
Lösung 4.16 237
Lösung 4.17 239
Lösung 4.18 240
Lösung 4.19 242
Lösung 4.20 245
Lösung 4.21 247
Lösung 4.22 249

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

Lösung 4.23 250
Lösung 4.24 251
Lösung 4.25 252
Lösung 4.26 253

5 Erweiterte graphische Programmierung 255
5.1 Der RootPane-Container 255
☆☆ Aufgabe 5.1: Die LayeredPane einer Fensterkomponente 257
☆☆ Aufgabe 5.2: Eine beliebige Instanz der Klasse JLayeredPane 258
☆☆☆ Aufgabe 5.3: Das Positionieren von Komponenten in und innerhalb von Ebenen 259
☆☆☆ Aufgabe 5.4: Eine GlassPane, die Ereignisse empfängt 259
☆☆☆ Aufgabe 5.5: Eine benutzerdefinierte GlassPane-Komponente 260
5.2 Interne Fenster 260
☆ Aufgabe 5.6: Die Instanzen der Klasse JInternalFrame 261
☆☆ Aufgabe 5.7: Die Verschachtelung von internen Fenstern 261
5.3 Die Applikation als Auslöser für Zeichenoperationen 261
☆☆ Aufgabe 5.8: Aufruf der repaint()-Methode für AWT-Komponenten 262
☆☆ Aufgabe 5.9: Aufruf der repaint()-Methode für Swing-Komponenten 262
☆☆ Aufgabe 5.10: Die getGraphics()-Methode 263
☆☆ Aufgabe 5.11: Die Methoden paint(), getGraphics() und repaint() gleichzeitig nutzen 263
☆☆☆ Aufgabe 5.12: Weiterführendes Zeichnen (»incremental painting«) 263
5.4 Das Interface Shape 265
☆ Aufgabe 5.13: Die Methode draw() der Klasse Graphics2D 265
☆ Aufgabe 5.14: Wiederholungsaufgabe 266
☆ Aufgabe 5.15: Wiederholungsaufgabe 266
5.5 Praxisnahe Zeichenvorgänge und Eventbehandlungen 266
☆☆☆ Aufgabe 5.16: Wiederholungsaufgabe 266

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

5.6 Benutzerdefinierte Event-Objekte und Event-Listener 267
☆☆☆ Aufgabe 5.17: Das Eweitern der Klasse EventObject und des Interface EventListener 268
5.7 Lösungen 269
Lösung 5.1 269
Lösung 5.2 270
Lösung 5.3 272
Lösung 5.4 274
Lösung 5.5 276
Lösung 5.6 278
Lösung 5.7 279
Lösung 5.8 281
Lösung 5.9 282
Lösung 5.10 283
Lösung 5.11 284
Lösung 5.12 286
Lösung 5.13 288
Lösung 5.14 290
Lösung 5.15 291
Lösung 5.16 293
Lösung 5.17 297

6 Das Erscheinungsbild einer Anwendung mit graphischer Oberfläche 303
6.1 Die Architektur Model View Controller (MVC) von Swing-Komponenten. 303
6.2 Benutzerdefinierte Modelle, die Standard-Model-Interfaces implementieren 303
☆ Aufgabe 6.1: Die AWT-Klasse List und ein DefaultComboBox-Modell 304
☆☆☆ Aufgabe 6.2: Benutzerdefiniertes ComboBox-Modell ohne Eventbehandlung 305
☆☆☆ Aufgabe 6.3: Benutzerdefiniertes ComboBox-Modell mit Eventbehandlung 307
☆☆☆ Aufgabe 6.4: Benutzung des ComboBox-Modells für eine Viewer-Komponente vom Typ JComboBox 307
☆☆☆ Aufgabe 6.5: Die Vorlage für ein benutzerdefiniertes Modell als Erweiterung der Klasse PlainDocument definieren 308
6.3 Standard-Modelle am Beipiel der Klasse JTree 309

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

☆ Aufgabe 6.6: Die Klassen JTree und DefaultMutableTreeNode 309
☆☆☆ Aufgabe 6.7: Die Klassen DefaultTreeModel und DefaultTreeSelectionModel 309
☆☆☆ Aufgabe 6.8: Benutzerdefiniertes Tree-Modell mit Eventbehandlung 310
6.4 Die UI-Delegationsklassen und ihre Instanzen, der UI-Delegate 311
6.5 Java-Standard-User-Interface-Delegationsklassen. 312
☆ Aufgabe 6.9: Der Standard-UI-Delegate 312
☆ Aufgabe 6.10: Das Setzen der bevorzugten Größe für Komponenten 312
☆☆ Aufgabe 6.11: Das Setzen einer minimalen Größe für Komponenten in Abhängigkeit von der Größe eines zu zeichnenden Bildes 313
☆☆☆ Aufgabe 6.12: Das Setzen einer minimalen Größe für Komponenten in Abhängigkeit von der Größe der beim Zeichnen verwendeten Schrift. 313
6.6 Benutzerdefinierte User-Interface-Delegaten 314
☆☆☆ Aufgabe 6.13: Ein benutzerdefinierter UI-Delegate und das Setzen der bevorzugten Größe von Komponenten 314
☆☆☆ Aufgabe 6.14: Ein benutzerdefinierter UI-Delegate für eine benutzerdefinierte Komponente 315
☆☆☆ Aufgabe 6.15: Wiederholungsaufgabe 316
6.7 Die Klassen LookAndFeel, UIDefaults, UIManager und das Interface UIResource 317
☆☆ Aufgabe 6.16: Lesen von Werten der UIDefaults-Tabelle 318
☆ Aufgabe 6.17: Setzen von Werten der UIDefaults-Tabelle 319
☆☆☆ Aufgabe 6.18: Setzen des UI-Delegationsobjektes mit der Methode put() der UIManager-Klasse 319
6.8 Standard-LookAndFeel-Komponenten. 320
☆☆☆ Aufgabe 6.19: LookAndFeel-spezifische Wiedergabe von Komponenten 321
☆☆☆ Aufgabe 6.20: Wechseln zwischen allen installierten LookandFeel-Komponenten 322
6.9 Das Erweitern der LookandFeel-Klasse 322
☆☆☆ Aufgabe 6.21: Benutzerdefinierte LookAndFeel-Komponenten 322

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

☆☆☆ Aufgabe 6.22: Das Zusammenspiel zwischen den drei MVC-Komponenten Model, Viewer und Controller ... 323
6.10 Lösungen ... 324
Lösung 6.1 ... 324
Lösung 6.2 ... 326
Lösung 6.3 ... 328
Lösung 6.4 ... 331
Lösung 6.5 ... 333
Lösung 6.6 ... 336
Lösung 6.7 ... 337
Lösung 6.8 ... 340
Lösung 6.9 ... 345
Lösung 6.10 ... 346
Lösung 6.11 ... 347
Lösung 6.12 ... 349
Lösung 6.13 ... 351
Lösung 6.14 ... 354
Lösung 6.15 ... 357
Lösung 6.16 ... 360
Lösung 6.17 ... 362
Lösung 6.18 ... 363
Lösung 6.19 ... 364
Lösung 6.20 ... 369
Lösung 6.21 ... 373
Lösung 6.22 ... 377

7 **Innere Klassen** ... 383
7.1 Die Definition von inneren Klassen und deren Instanzen ... 383
☆ Aufgabe 7.1: Instanziieren von Member-Klassen innerhalb der umgebenden Klasse ... 384
☆☆ Aufgabe 7.2: Instanziieren von Member-Klassen außerhalb der umgebenden Klasse ... 385
☆ Aufgabe 7.3: Instanziieren von Static-Member-Klassen innerhalb der umgebenden Klasse ... 385
☆☆ Aufgabe 7.4: Instanziieren von Static-Member-Klassen außerhalb der umgebenden Klasse ... 386
☆ Aufgabe 7.5: Lokale Klassen ... 386
☆ Aufgabe 7.6: Anonyme Klassen ... 387

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

☆ Aufgabe 7.7: Member-Interfaces 387
☆☆ Aufgabe 7.8: Member-Interface mittels einer anonymen Klasse implementieren 388
☆☆ Aufgabe 7.9: Member-Interface von einer anderen Klasse implementieren 388
☆☆ Aufgabe 7.10: Member-Interfaces und Member-Klassen 389
7.2 Innere Klassen am Beispiel von Event-Listener und Event-Adapter ... 389
☆ Aufgabe 7.11: Den WindowAdapter als Member-Klasse definieren 389
☆ Aufgabe 7.12: Den WindowAdapter mittels einer anonymen Klasse implementieren 390
☆ Aufgabe 7.13: Den ActionListener mittels einer anonymen Klasse implementieren 390
☆☆☆ Aufgabe 7.14: Benutzdefinierte Event-Objekte und Event-Listener für JButton- und JTextField-Komponenten 390
☆☆☆ Aufgabe 7.15: Benutzdefinierte Event-Objekte und Event-Listener für JRadioButton-Komponenten 391
☆☆☆ Aufgabe 7.16: Benutzderfinierte Event-Objekte und Event-Listener für JLabel-Komponenten 391
7.3 Weitere Beispiele mit inneren Klassendefinitionen 392
☆☆ Aufgabe 7.17: Die Methoden der Klasse JOptionPane aus einer anonymen Klasse aufrufen 393
☆☆☆ Aufgabe 7.18: Einen Farbauswahldialog und die main()-Methode innerhalb von inneren Klassen definieren 393
☆☆☆ Aufgabe 7.19: Wiederholungsaufgabe 394
7.4 Lösungen: 395
Lösung 7.1 395
Lösung 7.2 397
Lösung 7.3 400
Lösung 7.4 401
Lösung 7.5 404
Lösung 7.6 406
Lösung 7.7 407
Lösung 7.8 408
Lösung 7.9 409
Lösung 7.10 411

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

Lösung 7.11 413
Lösung 7.12 414
Lösung 7.13 414
Lösung 7.14 415
Lösung 7.15 420
Lösung 7.16 423
Lösung 7.17 427
Lösung 7.18 429
Lösung 7.19 431

8 Generics 435
8.1 Die Generizität 435
8.2 Generische Klassen und Interfaces 436
☆ Aufgabe 8.1: Generischer Datentyp als Behälter für die Instanzen vom Typ des Klassenparameters 437
☆ Aufgabe 8.2: Generischer Datentyp als »Über-Typ« für die Instanzen vom Typ des Klassenparameters 437
☆☆☆ Aufgabe 8.3: Generischer Stack 438
8.3 Wildcardtypen 438
☆☆ Aufgabe 8.4: Ungebundene Wildcardtypen 439
☆☆ Aufgabe 8.5: Obere Schranke (»upper bound wildcard«) für Wildcardtypen 439
☆☆ Aufgabe 8.6: Untere Schranke (»lower bound wildcard«) für Wildcardtypen 440
8.4 Legacy Code, Erasure und Raw-Typen 441
☆☆ Aufgabe 8.7: Raw-Typen am Beispiel einer generischen Klasse mit zwei Typparametern 442
☆ Aufgabe 8.8: Generische Interfaces 443
☆☆ Aufgabe 8.9: Brückenmethoden (»bridge methods«) . . . 443
8.5 Generische Arrays 444
☆☆ Aufgabe 8.10: Erzeugen von generischen Arrays 444
8.6 Generische Methoden 445
☆☆☆ Aufgabe 8.11: Generische Methodendefinitionen 445
8.7 for-each-Schleifen für Collectionen 446
☆☆☆ Aufgabe 8.12: Generische Arrays in generischen Methodendefinitionen 446
8.8 Generische Standard-Klassen und -Interfaces 447
☆ Aufgabe 8.13: Die Klasse ArrayList<E> und die Schnittstelle List<E> 448

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

☆☆ Aufgabe 8.14: Die Klasse Vector<E> und die Schnittstelle Collection<E> ... 448
☆☆ Aufgabe 8.15: Die Klasse TreeMap<K,V> ... 449
☆☆☆ Aufgabe 8.16: Wiederholungsaufgabe ... 450
8.9 Enumerationen und die generische Klasse Enum<E extends Enum<E>> ... 450
☆☆ Aufgabe 8.17: Die Definition von Enumerationen ... 451
☆☆☆ Aufgabe 8.18: Konstruktoren und Methoden von enum-Klassen ... 451
8.10 Die Interfaces Enumeration<E>, Iterable<T> und Iterator<E> sowie Map<K,V> und Set<E> ... 452
☆☆ Aufgabe 8.19: Weitere generische Schnittstellen ... 452
8.11 Die Einträge der UIDefaults-Tabelle als Instanz der Klasse Hashtable<Object, Object> ... 453
☆☆ Aufgabe 8.20: Das Ändern der font-Eigenschaft von Swing-Komponenten ... 453
8.12 Die generischen Klassen Class<T> und Constructor<T> und das »dynamische« Erzeugen von Objekten ... 454
8.13 Das Reflection-API ... 454
☆☆ Aufgabe 8.21: Die Klasse Class<T> ... 458
☆☆☆ Aufgabe 8.22: Die Klasse Constructor<T> ... 459
☆☆ Aufgabe 8.23: Erzeugen von generischen Arrays mit Hilfe eines Class-Objekts ... 460
☆☆ Aufgabe 8.24: Mit Reflection Informationen zu Klassen, Oberklassen und Interfaces holen ... 461
☆☆☆ Aufgabe 8.25: Das Interface GenericDeclaration und die Unterinterfaces von Type ... 462
8.14 Lösungen ... 464
Lösung 8.1 ... 464
Lösung 8.2 ... 465
Lösung 8.3 ... 466
Lösung 8.4 ... 468
Lösung 8.5 ... 469
Lösung 8.6 ... 471
Lösung 8.7 ... 472
Lösung 8.8 ... 476
Lösung 8.9 ... 477
Lösung 8.10 ... 478

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

Lösung 8.11 479
Lösung 8.12 481
Lösung 8.13 482
Lösung 8.14 483
Lösung 8.15 485
Lösung 8.16 486
Lösung 8.17 488
Lösung 8.18 489
Lösung 8.19 492
Lösung 8.20 495
Lösung 8.21 496
Lösung 8.22 498
Lösung 8.23 501
Lösung 8.24 502
Lösung 8.25 507

9 Exceptions und Errors 515
9.1 Ausnahmen auslösen 515
9.2 Ausnahmen abfangen oder weitergeben 516
☆ Aufgabe 9.1: Unbehandelte RuntimeExceptions 516
☆ Aufgabe 9. 2: Behandelte RuntimeExceptions 517
☆☆ Aufgabe 9.3: Die Weitergabe von Ausnahmen 517
9.3 Das Verwenden von finally in der Ausnahmebehandlung 518
☆ Aufgabe 9.4: Der finally-Block 518
☆☆☆ Aufgabe 9.5: Geschachtelte try/catch-Blöcke 519
9.4 Ausnahmen manuell auslösen 520
☆☆ Aufgabe 9.6: Standard-Ausnahmen manuell auslösen 520
9.5 Exception-Unterklassen erzeugen 521
☆ Aufgabe 9.7: Benutzerdefinierte Ausnahmen manuell auslösen 521
☆☆☆ Aufgabe 9.8: Wiederholungsaufgabe 521
9.6 Ketten von Ausnahmen 523
☆☆☆ Aufgabe 9.9: Exception-Ketten 523
9.7 Die Ausnahmen bei einem Wechsel von LookAndFeel-Komponenten 524
☆☆☆ Aufgabe 9.10: Die LookAndFeel-spezifischen Einträge der UIDefaults-Tabelle 524

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

9.8 Lösungen ... 525
Lösung 9.1 ... 525
Lösung 9.2 ... 526
Lösung 9.3 ... 527
Lösung 9.4 ... 529
Lösung 9.5 ... 530
Lösung 9.6 ... 534
Lösung 9.7 ... 536
Lösung 9.8 ... 538
Lösung 9.9 ... 542
Lösung 9.10 ... 545

10 Java-Typen ... 553
10.1 Typen in Java ... 553
10.2 Typprüfung und Typsicherheit mittels Generics ... 556
10.3 Subtyping für parametrisierte Typen ... 562
10.4 Die extends-Klausel ... 563
10.5 Typinferenz für Methoden ... 564
10.6 Typinferenz beim Erzeugen von Instanzen eines generischen Typs ... 565
10.7 Heap Pollution ... 567
10.8 Wildcard-Capture ... 569
☆ Aufgabe 10.1: Typinferenz beim Instanziieren von generischen Klassen ... 570
☆ Aufgabe 10.2: Der Diamond-Operator ... 571
☆☆ Aufgabe 10.3: Schranken für Typvariablen und Typinferenz für Methoden ... 572
☆☆☆ Aufgabe 10.4: Parametrisierte Typen und Wildcardtypen ... 575
☆☆ Aufgabe 10.5: Generische Arraytypen ... 579
☆☆ Aufgabe 10.6: Subtyping von Referenztypen ... 583
10.9 Multi-catch-Klausel und verbesserte Typprüfung beim Rethrowing von Exceptions ... 584
☆ Aufgabe 10.7: Disjunction-Typ für Exceptions ... 585
☆☆ Aufgabe 10.8: Typprüfung beim Rethrowing von Exceptions ... 587
10.10 Lösungen ... 589
Lösung 10.1 ... 589
Lösung 10.2 ... 593

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

Lösung 10.3 598
Lösung 10.4 604
Lösung 10.5 621
Lösung 10.6 627
Lösung 10.7 634
Lösung 10.8 636

11 Java 8 Lambdas und Streams 641
11.1 Mittels anonymer Klassen Code an Methoden übergeben. 641
11.2 Funktionale Interfaces 643
11.3 Syntax und Deklaration von Lambda-Ausdrücken 643
☆ Aufgabe 11.1: Lambda-Ausdruck ohne Parameter versus anonymer Klasse 648
☆ Aufgabe 11.2: Lambda-Ausdruck mit Parameter versus anonymer Klasse 651
☆ Aufgabe 11.3: Weitere Beispiele mit anonymen Klassen und Lambda-Ausdrücken. 651
11.4 Scoping und Variable Capture 652
☆☆ Aufgabe 11.4: Die Umgebung von Lambda-Ausdrücken 653
☆☆ Aufgabe 11.5: Die neuen funktionalen Interfaces Consumer<T> und Predicate<T> und die Übergabe von Lambda-Ausdrücken in Methoden. 654
☆☆ Aufgabe 11.6: Wiederholungsaufgabe 656
11.5 Methoden- und Konstruktor-Referenzen. 659
☆ Aufgabe 11.7: Methoden-Referenzen in Zuweisungen 661
☆☆ Aufgabe 11.8: Methoden-Referenzen als Argumente in Methodenaufrufen übergeben 662
☆☆ Aufgabe 11.9: Konstruktor-Referenzen und die neuen funktionalen Interfaces Supplier<T> und Function<T,R> 664
11.6 Default-Methoden und statische Methoden in Interfaces 666
11.7 Das neue Interface Stream 668
11.8 Die forEach-Methoden von Iterator, Iterable und Stream 672
11.9 Die Default-Methoden des Map-Interface 674
☆ Aufgabe 11.10: Die forEach()-Methode von Iterable 676
☆ Aufgabe 11.11: Die forEach()-Methode des Iterator-Interface. 677

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

☆ Aufgabe 11.12: Die funktionalen Interfaces BiConsumer<T,U>, BiPredicate<T,U> und BiFunction<T,U,R> ... 678
☆☆ Aufgabe 11.13: Die Methoden des Interface Stream und die Behandlung von Exceptions in Lambda-Ausdrücken ... 680
☆ Aufgabe 11.14: Die forEach- und replace-Methoden des Map-Interface ... 683
11.10 Das Interface Collector und die Klasse Collectors. Reduktion mittels Methoden von Streams und Kollektoren ... 684
☆☆ Aufgabe 11.15: Weitere Methoden des Interface Stream: limit(), count(), max(), min(), skip(), reduce() und collect() ... 689
☆☆☆ Aufgabe 11.16: Das Interface Collector und die Klasse Collectors ... 696
11.11 Parallele Streams ... 698
11.12 Die Bulk-Operationen der Klasse ConcurrentHashMap ... 702
☆☆☆ Aufgabe 11.17: Parallele Streams ... 705
☆☆☆ Aufgabe 11.18 : Die neuen Methoden von ConcurrentHashMap ... 707
11.13 Lösungen ... 709
Lösung 11.1 ... 709
Lösung 11.2 ... 716
Lösung 11.3 ... 718
Lösung 11.4 ... 720
Lösung 11.5 ... 725
Lösung 11.6 ... 729
Lösung 11.7 ... 738
Lösung 11.8 ... 739
Lösung 11.9 ... 749
Lösung 11.10 ... 754
Lösung 11.11 ... 755
Lösung 11.12 ... 756
Lösung 11.13 ... 760
Lösung 11.14 ... 770
Lösung 11.15 ... 774
Lösung 11.16 ... 795
Lösung 11.17 ... 803
Lösung 11.18 ... 818

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

12 Die neuen Features von Java 9 ... 827
12.1 Einführung ... 827
12.2 Factory-Methoden in Collections ... 827
☆ Aufgabe 12.1: Factory-Methoden für List, Set und Map ... 828
12.3 Stream-Erweiterungen ... 829
☆ Aufgabe 12.2: Die neuen Stream-Methoden ... 830
12.4 Die map() und flatMap()-Methoden von Stream und Optional ... 831
☆☆ Aufgabe 12.3 : map() versus flatMap() ... 834
12.5 Optional-Erweiterungen ... 837
☆☆ Aufgabe 12.4: Die neuen Methoden der Optional-Klasse ... 837
12.6 Kollektoren in Java 9 ... 841
☆☆ Aufgabe 12.5: Die filtering() und mapping()-Kollektoren ... 841
12.7 Process-API-Erweiterungen ... 843
☆☆ Aufgabe 12.6: Die neuen Interfaces ProcessHandler und ProcessHandler.Info ... 844
12.8 Reactive Programming ... 847
☆☆ Aufgabe 12.7: Die Interfaces der Flow-API implementieren ... 849
☆☆☆ Aufgabe 12.8: Ein Publisher, der zwei Subscriber beliefert ... 851
12.9 Das Java-Modulsystem ... 855
☆ Aufgabe 12.9: Eine einfache Modul-Definition ... 863
☆☆ Aufgabe 12.10: Eine Applikation mit mehreren Modulen ... 865
☆☆ Aufgabe 12.11: Implizites Lesen von Modulen ... 867
☆☆ Aufgabe 12.12: Eine modulbasierte Service-Implementierung ... 868
12.10 Platform-Logging ... 869
☆☆ Aufgabe 12.13: Eine Default-Implementation von System.LoggerFinder ... 871
☆☆ Aufgabe 12.14: Ein benutzerdefinierter System.Logger ... 874
12.11 Multi-Resolution Images ... 876
☆☆ Aufgabe 12.15: Das Interface MultiResolutionImage und die Klasse BaseMultiResolutionImage ... 878
12.12 Die Java-Shell (JShell) ... 881
☆ Aufgabe 12.16: Die JShell-API ... 886

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

12.13 Andere Änderungen aus dem JDK 9 887
☆ Aufgabe 12.17: Der Diamond-Operator in Java 9 888
☆ Aufgabe 12.18: Private Interface-Methoden 890
☆ Aufgabe 12.19: Die try-with-resources-Anweisung..... 894
12.14 Java 9 Internals 895
12.15 Lösungen 896
Lösung 12.1 896
Lösung 12.2 898
Lösung 12.3 902
Lösung 12.4 910
Lösung 12.5 917
Lösung 12.6 923
Lösung 12.7 930
Lösung 12.8 935
Lösung 12.9 943
Lösung 12.10 944
Lösung 12.11..... 950
Lösung 12.12 956
Lösung 12.13 960
Lösung 12.14 964
Lösung 12.15 971
Lösung 12.16 980
Lösung 12.17 982
Lösung 12.18 985
Lösung 12.19 988

Stichwortverzeichnis 991

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

Einleitung

Das Java 9 Übungsbuch ist aus der Erkenntnis entstanden, dass zu umfangreiche Beispiele mit komplizierten Algorithmen beim Lernen von Java am Anfang keine echte Hilfe bieten. Darum liegt der Schwerpunkt des Buches nicht auf der Umsetzung von komplizierten Vorgängen, sondern auf der Art und Weise, wie man die nicht immer für den Leser verständlich formulierten Erläuterungen zu Java-Klassen und -Interfaces aus der Dokumentation mit einfachen Beispielen erklären und gleichzeitig die zugrunde liegenden Konzepte damit erörtern kann.

Das Java 9 Übungsbuch wendet sich in erster Linie an Lehrer, Schüler und Studenten als Begleitliteratur zum Lernen der Programmiersprache Java, ist aber auch zum Selbststudium für alle Interessenten an dem Erlernen der Programmiersprache geeignet.

Durch die Einfachheit und Vollständigkeit der Aufgabenlösungen sowie die unterschiedlichen Lösungsmöglichkeiten erhält der Leser ein fundiertes Verständnis für die Aufgabenstellungen und deren Lösungen.

Durch das Lösen von Aufgaben soll der in Referenz- und Lehrbüchern von Java angebotene Stoff vertieft werden, und die dabei erzielten Ergebnisse können anhand der Lösungsvorschläge überprüft werden. Die Beispiele im Buch sind eher selten von zu komplexer Natur, sodass der eigentliche Zweck nicht in den Hintergrund tritt, und alle beschriebenen Themen können tiefgehend und präzise damit eingeübt werden.

Vorkenntnisse

Es ist Voraussetzung, dass der Leser zusätzlich mit einem Lehrbuch zu Java arbeitet bzw. bereits damit gearbeitet hat. Die grundlegenden Erläuterungen zu Java in diesem Buch können lediglich als Wiederholung des bereits vorhandenen Wissens dienen, reichen aber nicht aus, um die Sprache Java erst neu zu lernen.

Als weitere Voraussetzung gelten Grundlagen im Bereich der Programmierung und im Umgang mit dem Betriebssystem. Ein paralleler Zugriff auf die Java-Online-Dokumentation kann Hilfe zu den Java-Standard-Klassen bieten.

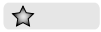
© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

Aufbau des Buches

Jedes Kapitel beginnt mit einer kurzen und knappen Wiederholung des Stoffes, der in den Übungsaufgaben dieses Kapitels verwendet wird. Danach folgen alle Aufgabenstellungen der Übungen. Am Ende des Kapitels folgen gesammelt alle Lösungen der Übungsaufgaben mit Kommentaren, Erläuterungen und Hinweisen.

Alle Kapitel enthalten Wiederholungsaufgaben, in denen versucht wird, die bis zur angelangten Stelle definierten Begriffe zusammenzutragen.

Die Aufgaben haben unterschiedliche Schwierigkeitsgrade. Dieser wird im Aufgabenkopf durch 1 bis 3 Sternchen gekennzeichnet:



1 Sternchen für besonders einfache Aufgaben, die auch von Anfängern leicht bewältigt werden können;

2 Sternchen für etwas kompliziertere Aufgaben, die einen durchschnittlichen Aufwand benötigen;

3 Sternchen für Aufgaben, die sich an geübte Programmierer richten und einen wesentlich höheren Aufwand oder die Kenntnis von speziellen Details erfordern.

Die Programme aus früheren Übungen werden teilweise in späteren Übungen gebraucht und es wird auch immer wieder auf theoretische Zusammenhänge zurückgekommen oder hingewiesen.

Die Lösungsvorschläge verfügen über umfangreiche Kommentare, sodass ein Verständnis für die durchgeführte Aufgabe auch daraus abgeleitet werden kann und dies jede einzelne Aufgabe im Gesamtkontext unabhängig erscheinen lässt.

In den Kapiteln 1, 2 und 3 liegt das Hauptmerkmal auf den Eigenheiten der objektorientierten Programmierung mit Java. Durch eine Vielzahl von Beispielen wird gezeigt, was die Java-Standard-Klassen an Funktionalitäten bieten und wie diese sinnvoll in die Definition von eigenen Klassen eingebettet werden können.

Die Kapitel 4 bis 6 erproben umfangreiche Kenntnisse über die Java-GUI-Programmierung mit AWT und Swing.

Die darauf folgenden Spezialkapitel über Innere Klassen, Generics und Exceptions bringen parallel zu den Aufgaben, die einer umfangreichen Darstellung von neu

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

eingeführten Begriffen und Definitionen dienen, auch weiterführende Aufgabenbeispiele zu den bisherigen Kapiteln.

Weil der Schwerpunkt des Buches nicht auf der Umsetzung von aufwendigen Algorithmen liegen soll, werden einfache Beispiele mit Zahlen, Buchstaben, Wörtern, Büchern, Wochentagen, geometrischen Figuren etc. und teilweise auch mit ganz abstrakten Klassennamen wie Klasse1, Klasse2, KlasseA, KlasseB etc. verwendet. In den Beispielen zur graphischen Programmierung werden nicht alle Komponenten behandelt – dies machen ja vorrangig Referenzbücher – sondern die Beispiele dienen dazu, die Basiskonzepte von Swing-Architekturen in sich ergänzend zu beschreiben und können ganz einfach auch für andere Swing-Komponenten umgesetzt werden.

Das Kapitel 10 wurde um Erweiterungen zu den Themen Java-Typen, Generizität und Reflection ergänzt, um eine bessere Abgrenzung von Begriffen und Interpretationen, die sich im Laufe der Jahre etabliert haben, zu erreichen. Neue Aufgaben, die der Erklärung von Typinferenz, Subtyping von parametrisierten und Wildcardparametrisierten Typen, Heap-Pollution und Wildcard-Capture dienen sollen, sind hinzugekommen. Andere wiederum demonstrieren die Benutzung der Interfaces des Reflection-API in der Abfrage von generischen und nichtgenerischen Informationen zu Klassenstrukturen.

Mit der im März 2014 freigegebenen Java-Version 8.0 eröffnen sich ganz neue Betrachtungsweisen und Programmiertechniken in der Entwicklung von Applikationen mit Java, die in Kapitel 11 präsentiert werden.

Eine der wichtigsten Neuerungen in Java 8 sind neue Sprachmittel, die sogenannten Lambda-Ausdrücke, eine Art anonyme Methoden, die auf funktionalen Interfaces basieren. Diese besitzen jedoch eine viel kompaktere Syntax als Methoden. Das resultiert daraus, dass in ihrer Benutzung auf Namen, Modifikatoren, Rückgabetyp, `throws`-Klausel und in vielen Fällen auch auf Parameter verzichtet werden kann. Mit ihnen kann Funktionalität ausgeführt, gespeichert und übergeben werden, wie dies bisher nur von Instanzen in Java bekannt war.

Damit verbundene Themen wie die Gegenüberstellung zu anonymen Klassen, Syntax und Semantik, Behandlung von Exceptions, Scoping und Variable Capture, Methoden- und Konstruktor-Referenzen werden in den ersten Unterkapiteln des 11. Kapitels dieses Buches beschrieben und an Hand von vielen Beispielen erläutert.

Des Weiteren finden Sie hier die Beschreibung aller neuen funktionalen Interfaces und deren Methoden. Mit Java 8 wurden sogenannte Default-Methoden eingeführt. Diese werden in der Literatur auch als »virtual extension«- bzw. »defender«-Methoden bezeichnet und Schnittstellen, die über derartige Methoden verfügen, als erweiterte Schnittstellen. Damit können Interfaces zusätzlich zu abstrakten Methoden konkrete Methoden in Form von Standard-Implementierungen definieren und in Java wird die Mehrfachvererbung von Funktionalität ermöglicht. Neben Default-Methoden können Interfaces in Java 8 auch statische Methoden enthalten. Anders

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

als die statischen Methoden von Klassen werden diese jedoch nicht von abgeleiteten Typen geerbt.

Die nachfolgenden Unterkapitel beschäftigen sich im Detail mit der Definition und Nutzung von Streams. Ein Stream besteht aus einer Folge von Werten (in der Literatur wird auch von Sequenzen von Elementen gesprochen), die nur teilweise von mehreren in einer Pipeline dazwischenliegenden Operationen ausgewertet und durch eine abschließende Operation bereitgestellt werden. Diese Operationen werden in Java als Methodenaufrufe formuliert, die Funktionalität in Form von Lambdas und Methoden-Referenzen entgegennehmen können und diese auf alle Elemente der Folge anwenden.

Aufbauend auf Lambda-Ausdrücken ermöglichen diese eine ganz neue Arbeitsweise mit Arrays, Collections und Maps. Sie bringen vielversprechende Optimierungen mit sich, wie z.B. die Memoisation, eine Technik, um Programme zu beschleunigen, indem Rückgabewerte von Funktionen zwischengespeichert statt neu berechnet werden. Dazu gehören auch neue Parallelisierungstechniken, die auf den mit Java 7 eingeführten Fork/Join-Mechanismen basieren, interne Iterationen anstelle von den bisher bekannten externen benutzen und in Bibliotheken direkt integriert wurden. Als Beispiel im Umgang mit parallelen Streams wird im letzten Unterkapitel auf die Neuerungen aus der Klasse ConcurrentHashMap eingegangen.

Anstatt zu spezifizieren, »wie ein Ergebnis zu berechnen ist«, indem eine Datenstruktur von Anfang bis Ende durchlaufen wird und für jedes Element, das eventuell einer bestimmten Bedingung genügt, ein Wert berechnet und z.B. einer Summe hinzugefügt wird, braucht man mit Java 8 nur zu formulieren, »was zu erledigen ist«, wie z.B. das Berechnen der Summe oder eines maximalen/minimalen Werts von Elementen, die die erwünschte Bedingung erfüllen. Damit wird die Logik des Durchlaufens einer Datenstruktur von der Funktionalität, die auf jedem ihrer -Elemente durchgeführt wird, getrennt.

Mit einer Vielzahl von Aufgaben basierend auf Lambdas, Streams und Kollektoren (in denen Stream-Elemente angesammelt und reduziert werden können) werden die neuen Techniken aus Java 8 angewandt und alle neuen Begriffe erklärt.

In Kapitel 12 werden die Neuerungen zu Java 9 präsentiert. Aus der großen Anzahl von JEPs (JDK Enhancements Proposals), die in die neue Version eingeflossen sind, werden einige wichtige Bereiche, die die Themen aus diesem Buch ergänzen, und erste Schritte in der Modularisierung näher erläutert und wie immer anhand von Aufgaben eingeübt.

Die ersten Unterkapitel beschäftigen sich mit den neuen Factory-Methoden aus Collections, den Erweiterungen des Stream-Interface und der Optional-Klasse sowie mit den neuen Kollektoren von Java 9.

Java 9 bringt mehrere Updates in der Process-API mit sich, die dazu dienen sollen, auf einfachere Art und Weise als bisher Informationen über laufende Prozesse zu ermitteln, vor allem für den JVM-Prozess und selbst gestartete Prozesse. Auf diese

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

wird im Unterkapitel 12.7 Bezug genommen.

Das Unterkapitel 12.8 widmet sich der Unterstützung von Reactive Streams. Im Reactive-Programming-Model existieren ein Publisher und ein Subscriber. Der Publisher veröffentlicht einen Stream von Daten, an den der Subscriber asynchron angeschlossen ist. Die Verbindung, die zwischen einem Publisher und einem Subscriber aufgebaut wird, wird mit einer Subscription beschrieben. Wie die Beispiele aus diesem Unterkapitel zeigen, kann über den dazwischen geschalteten Vermittler gesteuert werden, dass die Subscriber Daten anfordern und die Publisher möglichst so viele Daten senden, wie von den Subscribern nicht-blockierend verarbeitet werden können.

Die mit Sicherheit wichtigste und umfangreichste Änderung, die Java 9 mit sich bringt, ist das neue Java-Modulsystem. Mit dem neuen Modulsystem wurde Java selbst modular gemacht und es können eigene Applikationen und Bibliotheken modularisiert werden.

Im Unterkapitel 12.9 werden im Detail beschrieben:

- Das Erzeugen von Modulen und deren Abhängigkeiten
- Das Packaging von Modul-Code
- Die Implementierung von Services
- Die Modularisierung der Java-Plattform

Mit einer Vielzahl von Aufgaben können die neuen Strukturen eingeübt werden.

In Ergänzung dazu werden in den Unterkapiteln 12.10 und 12.11 das Platform-Logging und Multi-Resolution-Images mittels modulbasierter Applikationen präsentiert.

Im JDK 9 ist mit dieser Version auch ein neuer Kommandointerpreter, die Java-Shell (JShell) zu finden. Sie ist ein einfaches Tool, mit dem kleine Java-Programme und einzelne Anweisungen, sogenannte Snippets, interaktiv getestet werden können.

Die JShell kann im Gegensatz zu JavaScript nicht als Skript-Sprache in Java integriert werden. Stattdessen verfügt diese über eine eigene API. Ein Beispiel dazu finden Sie ebenfalls in diesem Unterkapitel.

Weitere Änderungen aus dem JDK 9 wie:

- der Diamond-Operator für innere anonyme Klassen,
- private Methoden in Interfaces,
- Ergänzungen für `try-with-resources`-Blocks

sind Thema des Unterkapitels 12.13 und im letzten Unterkapitel 12.14 werden Java-Internals wie Compact-Strings und die neue Versionierung von Java angesprochen.

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

Alle in diesem Buch enthaltenen Beispiele von Java-Applikationen sind und bleiben auch ohne Modularisierung unter den nachfolgenden Java-Versionen ablaufbar und dienen in erster Linie dazu, die Java-Programmiersprache kennenzulernen. Sollen diese modularisiert werden, kann man sich nach den in der Datei Java9Migration.pdf präsentierten Empfehlungen richten, die zusätzlich zu dem gesamten Source-Code aus diesem Buch auf der Webseite des Verlags zum Download angeboten wird.

An dieser Stelle möchte ich auf das dem Buch zugrunde liegende Konzept hinweisen, dass parallel zu einfachen Aufgaben, die zu allen eingeführten Definitionen und Begriffen gebracht werden, auch Aufgaben von einem höheren Schwierigkeitsgrad präsentiert werden. Dabei werden anhand von inhaltlichen Zusammenhängen zwischen den Beispielen viele Basiskonzepte von Java erläutert.

Das Buch soll möglichst parallel zu einer Vielzahl von Java-Lehrbüchern eingesetzt werden können und einen Beitrag dazu leisten, die große Fülle von Informationen, die auf uns über die API-Dokumentation zukommt, besser einzuordnen und korrekt anwenden zu können.

Im Inhaltsverzeichnis werden alle Aufgaben mit einer Aufgabennummer und einer Kurzbeschreibung aufgelistet. Das Stichwortverzeichnis enthält alle wichtigen Begriffe sowie unter dem Oberbegriff »Klassen-« bzw. »Schnittstellendefinitionen« die Namen aller Beispiel-Klassen und -Interfaces.

Benötigte Software

Für die Durchführung der Übungen in den Kapiteln 1 bis 9 können Sie das Java Development Kit (JDK) der Java Standard Edition (SE) Version 5.x oder 6.x verwenden. Für die Übungen von Kapiteln 10 brauchen Sie die Version 7.0 oder höher, für die von Kapitel 11 die Version 8.0 oder höher und für die von Kapitel 12 die neue Version 9.

Das aktuelle Java Development Kit der Java Standard Edition können Sie sich kostenlos von der Java-Homepage von Oracle `http://www.oracle.com/technetwork/` herunterladen. Das JDK umfasst sowohl die Software zur Programmerstellung als auch das JRE (Java Runtime Environment) für die Programmausführung.

Graphische Entwicklungsoberflächen (wie z.B. Eclipse) sind keine Voraussetzung und auch nicht Bestandteil dieses Buches. Die Programme lassen sich grundsätzlich mit einem Texteditor wie z.B. Notepad++ oder auch Wordpad eingeben und über die Kommandozeile durch den Aufruf der Programme `javac`, `jar` und `java` übersetzen, paketieren und starten. Die vollständigen Programmaufrufe sind bei jeder Aufgabe angegeben.

Sollte Ihnen beim Erlernen der Programmiersprache selbst eine Entwicklungsumgebung nicht zu aufwändig oder unübersichtlich erscheinen, steht Ihnen nichts im Wege, die zu den Aufgaben zugehörigen Programm- und Klassendateien zum Testen oder auch Ergänzen in eine solche einzubetten.

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

Website

Die Website zum Buch mit der Adresse `http://www.mitp.de/647` beinhaltet den plattformunabhängigen Quellcode der Lösungsvorschläge und die für Windows 7 kompilierte ausführbare Version als Download-Archiv. Diese Archivdatei enthält alle Java-Quellcodes, übersetzten Klassen und Bilddateien in einer Verzeichnisstruktur, die mit der im Buch beschriebenen übereinstimmt.

Ich wünsche Ihnen viel Erfolg beim Programmieren mit Java.

Elisabeth Jung

Autorin

Frau Elisabeth Jung ist freie Autorin und wohnhaft in Frankfurt am Main.

Nach dem Studium der Mathematik an der Universität Temeschburg in Rumänien hat Elisabeth Jung Grundlagen der Informatik und Fortran unterrichtet. Im Jahr 1982 hat sie bereits eine Aufgabensammlung für Fortran an der gleichen Universität veröffentlicht.

Zwischen den Jahren 1984 und 2001 hat Elisabeth Jung bei der Firma Siemens in Frankfurt in einer Vielzahl von Projektarbeiten umfangreiche Erfahrungen gesammelt in den Bereichen Programmiersprachen (Assembler, Fortran, Pascal, C, C++, Java), Datenbanken (Text-Retrieval, relationale Systeme, Client-Server-Architekturen) und der Entwicklung und dem Test von Hardwarenaher Systemsoftware.

Seit 2001 beschäftigt sie sich mit ihren bevorzugten Themen, der Mathematik und den objektorientierten Programmiersprachen, insbesondere Java. Ihr Buch ist aus der Erfahrung im Unterricht und dem Erlernen von Programmiersprachen entstanden, bei dem insbesondere das eigene Programmieren und die praktische Anwendung des Gelernten eine große Rolle spielt.

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

Kapitel 1

Klassendefinition und Objektinstanziierung

1.1 Klassen und Objekte

Klassen und Objekte bilden die Basis in der objektorientierten Programmierung. Eine Klasse ist eine Ansammlung von Attributen, die Eigenschaften definieren und Felder genannt werden, sowie von Funktionen, die deren Zustände und Verhaltensweisen festlegen und als Methoden bezeichnet werden. Felder und Methoden werden auch als Member einer Klasse bezeichnet.

Klassen werden mit dem Schlüsselwort `class` eingeleitet und definieren eine logische Abstraktion, die eine Basisstruktur für Objekte vorgibt. Sie sind als eine Erweiterung der primitiven Datentypen zu sehen. Während Klassen Modelle definieren, sind Objekte konkrete Exemplare, auch Instanzen der Klasse genannt.

Eine Schnittstelle (Interface) ist eine reine Spezifikation, die definiert, wie eine Klasse sich zu verhalten hat. Sie wird mit dem Schlüsselwort `interface` eingeleitet und kann keine Implementationen von Feldern und Methoden enthalten, mit Ausnahme von Konstantendefinitionen.

Ein Objekt von einer Klasse wird in Java mit dem `new`-Operator und einem Konstruktor erzeugt. Damit werden auch seine Felder initialisiert und der erforderliche Speicherplatz für das Objekt reserviert. Ein Objekt wird über eine Referenz angesprochen. Eine Referenz entspricht in Java in etwa einem Zeiger in anderen Programmiersprachen und ist einem Verweis auf das Objekt gleichzustellen, womit dieses identifiziert werden kann.

Mit Referenztypen werden Datentypen bezeichnet, welche im Gegensatz zu den primitiven Datentypen vom Entwickler selbst definiert werden. Diese können vom Typ einer Klasse, einer Schnittstelle oder eines Arrays sein. Ein Arraytyp identifiziert ein Objekt, das mehrere Werte von ein und demselben Typ speichern kann.

Die mit dem Modifikator `static` deklarierten Felder und Methoden in einer Klasse werden als Klassenfelder bzw. Klassenmethoden bezeichnet. Alle anderen Felder und Methoden einer Klasse werden auch Instanzfelder bzw. Instanzmethoden genannt.

Die Klassen bilden in Java eine Klassenhierarchie. Jede Klasse hat eine Oberklasse, deren Felder und Methoden sie erbt. Die Oberklasse aller Klassen in Java ist die Klasse `java.lang.Object` (s. Kapitel 2, Abgeleitete Klassen und Vererbung und Kapitel 3, Abstrakte Klassen und Interfaces).

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

Beim Definieren von Klassen ist zu beachten, dass eine Klasse eine in sich funktionierende Einheit darstellt, die alle benötigten Felder und Methoden definiert.

Konstruktoren

Für die Initialisierungen eines Objektes der Klasse werden sogenannte Konstruktoren genutzt. Diese sind eine spezielle Art von Methoden. Sie haben den gleichen Namen wie die Klasse, zu der sie gehören, und verfügen über keinen Rückgabewert. Weil der `new`-Operator schon eine Referenz auf das erzeugte Objekt zurückgibt, ist eine zusätzliche Rückgabe von Werten in Konstruktoren nicht mehr erforderlich. Jede Klasse besitzt einen impliziten Konstruktor, der auch als Standard-Konstruktor bezeichnet wird. Dieser hat eine leere Parameterliste und übernimmt das Initialisieren der Instanzfelder mit den Defaultwerten der jeweiligen Datentypen. Eine Klasse kann mehrere explizite Konstruktoren definieren, die sich durch ihre Parameterlisten unterscheiden. Der parameterlose Konstruktor wird nur dann vom Compiler generiert, wenn die Klasse keinen expliziten Konstruktor definiert. Ist dies jedoch der Fall und die Klasse möchte auch den parameterlosen Konstruktor benutzen, so muss dieser ebenfalls explizit definiert werden.

Klassenfelder und Klassenmethoden

Klassenfelder gehören nicht zu einzelnen Objekten, sondern zu der Klasse, in der sie definiert wurden. Alle durchgeführten Änderungen ihrer Werte werden von der Klasse und allen ihren Objekten gesehen. Jedes Klassenfeld ist nur einmal vorhanden. Darum sollten Klassenfelder benutzt werden, um Informationen, die von allen Objekten der Klasse benötigt werden, zu speichern. Diese Felder können direkt über den Klassennamen angesprochen werden und stehen zur Verfügung, bevor irgendein Objekt der Klasse erzeugt wurde.

Klassenmethoden werden ebenfalls über den Klassennamen angesprochen.

Innerhalb der eigenen Klasse können alle Klassenfelder und Klassenmethoden auch ohne Klassennamen angesprochen werden, sollten aber, um den Richtlinien der objektorientierten Programmierung zu genügen, immer mit diesem verwendet werden.

Instanzfelder und Instanzmethoden

Instanzfelder sind mehrfach vorhanden, da für jedes Objekt eine Kopie von allen Instanzfeldern einer Klasse erstellt wird. Die Instanzen einer Klasse unterscheiden sich voneinander durch die Werte ihrer Instanzfelder. Innerhalb einer Klasse kann der Zugriff darauf direkt über ihren Namen erfolgen oder in der Form `this.name` bzw. `objekt.name` (wobei `objekt` eine Referenz auf ein Objekt der Klasse ist).

Das Schlüsselwort `this` bezeichnet die Referenz auf das Objekt, an dem die Methode bzw. der Konstruktor aufgerufen wird, auch das »aktuelle Objekt« bzw. das »aufrufende Objekt« genannt. Konstruktoren können nur mit dem Schlüssel-

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

wort `new` aufgerufen werden, sowie innerhalb eines anderen Konstruktors über das Schlüsselwort `this`, gefolgt von der Parameterliste.

Java-Programme

Die Java-Programmtechnologie basiert auf der Zusammenarbeit von einem Compiler und einem Interpreter. Die Programme werden zuerst kompiliert, was einer syntaktischen Prüfung und der Erstellung von Bytecode entspricht. Es entsteht dadurch noch kein ausführbares Programm, sondern ein plattformunabhängiger Code, welcher an einen Interpreter, die virtuelle Java-Maschine (JVM), übergeben wird. Die JVM ist ein plattformspezifisches Programm, welches Bytecode lesen, interpretieren und ausführen kann.

Ein Java-Programm manipuliert Werte, die durch Typ und Name gekennzeichnet werden. Über die Festlegung von Name und Typ wird eine Variable definiert. Man spricht von Variablen in Zusammenhang mit einem Programm und von Feldern in Zusammenhang mit einer Klassendefinition. Variablen können auf mehrere Arten und Weisen klassifiziert werden. Eine Aufteilung wäre in Variablen von einem primitiven Datentyp bzw. einem Referenztyp und eine zweite in Variablen, die einen Wert oder mehrere Werte (Arrayvariablen) aufnehmen können. Außerdem spricht man von Klassenvariablen, wenn die Felder einer Klasse gemeint sind, Instanzvariablen, wenn die Felder einer Instanz gemeint sind, und Arrayelementen, wenn die einzelnen Elemente eines Array gemeint sind. Von Parametervariablen wird in Methodendeklarationen gesprochen und von lokalen Variablen, wenn diese innerhalb von einem Methodenrumpf oder Anweisungsblock definiert wurden.

Eine Variable ist im Grunde genommen ein symbolischer Name für eine Speicheradresse. Während für primitive Variablen der Typ des Wertes, der an dieser Adresse gespeichert wird, gleich dem Typ des Namens der Variable ist, wird im Falle einer Referenzvariablen nicht der Wert von einem bestimmten Objekt an dieser Adresse gespeichert, sondern die Angabe, wo das Programm den Wert (das Objekt) von diesem Typ finden kann.

Im Gegensatz zu lokalen Variablen, die keinen Standardwert haben und deswegen nicht verwendet werden können, bevor sie nicht explizit initialisiert wurden, werden alle Felder in einer Klassendefinition automatisch mit Defaultwerten initialisiert (mit `0`, `0.0`, `false`, primitive Typen und mit `null`, Referenztypen).

Die Definition einer Referenzvariablen besteht aus dem Namen der Klasse bzw. eines Interfaces gefolgt vom Namen der Variablen. Eine so definierte Referenzvariable kann eine Referenz auf ein beliebiges Objekt der Klasse oder einer Unterklasse oder den Defaultwert `null` aufnehmen. Weil Arrays als Objekte implementiert werden, müssen Arrays mit einem Array-Initialisierer oder mit dem `new`-Operator erzeugt werden.

Bevor ein Programm Objekte von einer Klasse bilden kann, wird diese mit dem Java-Klassenlader (Klasse `java.lang.ClassLoader`) geladen und mit dem Java Bytecode-Verifier geprüft.

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

Nach der Art der Ausführung existieren mehrere Arten von Java-Programmen:

- Ein Java-Applet ist ein Java-Programm, das im Kontext eines Webbrowsers mit bestimmten Sicherheitseinschränkungen abläuft. Es wird mittels einer HTML-Seite gestartet und kann im Browser oder mit Hilfe des Appletviewer ausgeführt werden.
- Ein Servlet ist ein Java-Programm, das im Kontext eines Webservers abläuft.
- Eine Java-Applikation ist ein eigenständiges Programm, das direkt von der JVM gestartet wird.

Jede Java-Applikation benötigt eine `main()`-Methode, welche einen Eingangspunkt für die Ausführung des Programms durch die JVM definiert. Diese Methode muss für alle Klassen der JVM zugänglich sein und deshalb mit dem Modifikator `public` definiert werden. Sie muss auch mit dem Modifikator `static` als Klassenmethode deklariert werden, da ein Aufruf dieser Methode möglich sein muss, ohne dass eine Instanz der Klasse erzeugt wurde. Von hier aus werden alle anderen Programmabläufe gesteuert.

Auf die Definition der Parameterliste der `main()`-Methode wird in nachfolgenden Programmbeispielen hingewiesen.

Gleich in den ersten Beispielen werden für Bildschirmausgaben die Methoden `System.out.print(…)` und `System.out.println(…)` der Klasse `java.io.PrintStream` verwendet. Mit der Methode `System.out.print(…)` wird in der gleichen Bildschirmzeile weiter geschrieben, in welcher eine vorangehende Ausgabe erfolgte. Die Methode `System.out.println()` ohne Parameter schließt eine vorher ausgegebene Bildschirmzeile ab und bewirkt, dass danach in eine neue Zeile geschrieben wird. Ein Aufruf der Methode `System.out.println(…)` mit Parameter ist äquivalent mit dem Aufruf von `System.out.print(…)` gefolgt von einem Aufruf von `System.out.println()` ohne Parameter, d.h. dieser Aufruf führt immer zu einem Zeilenende. Auch auf die Definition von diesen Methoden kommen wir in der Beschreibung der Java-Standard-Klasse `java.lang.System` noch mal zurück.

Ein Programm wird als Quelltext in einer oder mehreren `.java-Dateien` und als übersetztes Programm in einer oder mehreren `.class`-Dateien abgelegt.

Aufgabe 1.1 ☆

Definition einer Klasse

Definieren Sie eine Klasse `KlassenDefinition`, welche die `main()`-Methode als einzige Klassenmethode implementiert. Aus dieser soll die Bildschirmanzeige »Dies ist eine einfache Klassendefinition« erfolgen.

Hinweise für die Programmierung:

Ein Erzeugen von Instanzen der Klasse ist nicht erforderlich.

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

Achten Sie auf den richtigen Abschluss der Ausgabezeile.

Java-Dateien: `KlassenDefinition.java`
Programmaufruf: `java KlassenDefinition`

Aufgabe 1.2 ☆

Objekt (Instanz) einer Klasse erzeugen

Definieren Sie eine Klasse `ObjektInstanziierung`, die in einem parameterlosen Konstruktor die Bildschirmanzeige »Instanz einer Java-Klasse« vornimmt und in ihrer `main()`-Methode eine Instanz der Klasse erzeugt.

Java-Dateien: `ObjektInstanziierung.java`
Programmaufruf: `java ObjektInstanziierung`

1.2 Die Member einer Klasse: Felder und Methoden

Aufruf von Instanz- und Klassenmethoden

Aufruf von Instanzmethoden:

- aus der gleichen Klasse
  - aus einer Instanzmethode – direkt über den Methodennamen und die Parameterliste.
  - aus einer Klassenmethode – es muss ein Objekt der Klasse instanziiert werden und die Instanzmethode wird an diesem Objekt aufgerufen.
- aus einer anderen Klasse
  - es muss ein Objekt der Klasse instanziiert werden und die Instanzmethode wird an diesem Objekt aufgerufen.

Aufruf von Klassenmethoden:

- aus der gleichen Klasse
  - direkt über den Methodennamen und die Parameterliste.
- aus einer anderen Klasse
  - über den Klassennamen, gefolgt von einem Punkt und dem Methodennamen und der Parameterliste.

Zugriff auf Instanz- und Klassenfelder

Zugriff auf Instanzfelder:

- aus der gleichen Klasse
  - aus einer Instanzmethode – direkt über den Feldnamen.

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

- aus einer Klassenmethode – es muss ein Objekt der Klasse instanziiert werden und auf das Feld wird über die Objektreferenz gefolgt von einem Punkt und dem Feldnamen zugegriffen.

- aus einer anderen Klasse
  - es muss ein Objekt der Klasse instanziiert werden und auf das Feld wird über die Objektreferenz zugegriffen.

Zugriff auf Klassenfelder:

- aus der gleichen Klasse
  - direkt über den Feldnamen.

- aus einer anderen Klasse
  - über den Klassennamen, gefolgt von einem Punkt und dem Feldnamen.

Aufgabe 1.3 ☆

Zugriff auf Felder

Eine Klasse `FeldZugriffe` soll den Unterschied zwischen Klassen- und Instanzfeldern demonstrieren.

Sie definiert ein Klassenfeld `zaehlerAlsKlassenfeld` und ein Instanzfeld `zaehlerAlsInstanzfeld`. Die Werte von beiden Feldern werden im parameterlosen Konstruktor der Klasse inkrementiert (um eins erhöht). Definieren Sie eine Klassenmethode `anzeigeKlsMeth()` und eine Instanzmethode, `anzeigeInstMeth()`, welche jeweils beide Feldwerte am Bildschirm ausgeben.

Zum Testen dieser Klassendefinition wird die Klasse `FeldZugriffeTest` erstellt, die in ihrer `main()`-Methode eine Instanz der Klasse `FeldZugriffe` erzeugt, an welcher die Instanzmethode der Klasse aufgerufen wird. Ihre Klassenmethode wird über den Klassennamen aufgerufen.

Java-Dateien: `FeldZugriffe.java`, `FeldZugriffeTest.java`
Programmaufruf: `java FeldZugriffeTest`

Aufgabe 1.4 ☆☆

Aufruf von Methoden

Definieren Sie eine Klasse `MethodenAufrufe1`, welche die Instanzmethoden `instMethode1()` und `instMethode2()` und die Klassenmethoden `klsMethode1()` und `klsMethode2()` implementiert und eine zweite Klasse `MethodenAufrufe2` mit der Instanzmethode `instMethode()` und der Klassenmethode `klsMethode()`. Die Methode `instMethode1()` gibt am Bildschirm die Zeichenkette: "1. Instanzmethode der Klasse MethodenAufrufe1" aus und ruft die Metho-

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

den `instMethode2()` und `klsMethode1()` auf. Die Methode `instMethode2()` gibt die Zeichenkette: "2. Instanzmethode der Klasse MethodenAufrufe1« aus und die Methode `klsMethode1()` die Zeichenkette: "1. Klassenmethode der Klasse MethodenAufrufe1«. Die Methode `klsMethode2()`, gibt am Bildschirm die Zeichenkette: "2. Klassenmethode der Klasse MethodenAufrufe1" aus und ruft die Methoden `instMethode2()` und `klsMethode1()` auf.

Die Methoden der Klasse `MethodenAufrufe2` rufen beide die Methoden `instMethode1()` und `klsMethode2()` der Klasse `MethodenAufrufe1` auf.

Zum Testen des Aufrufes von Instanz- und Klassenmethoden aus Methoden der gleichen Klasse oder einer anderen Klasse wird die Klasse `MethodenAufrufeTest` erstellt, die in ihrer `main()`-`Methode` eine Instanz der Klasse `MethodenAufrufe2` erzeugt und deren Methoden aufruft.

Java-Dateien: `MethodenAufrufe1.java`, `MethodenAufrufe2.java`, `MethodenAufrufeTest.java`
Programmaufruf: `java MethodenAufrufeTest`

1.3 Das Überladen von Methoden

Eine Klasse kann mehrere Methoden mit gleichem Namen besitzen, wenn diese eine verschiedene Anzahl von Parametern bzw. Parameter von unterschiedlichen Typen im Methodenkopf definieren. Dabei ist ohne Bedeutung, ob es sich um Klassen- oder Instanzmethoden handelt.

Parallel zur Parameterliste unterscheidet sich auch die Aufrufsyntax der Methode. Diese Vorgehensweise ist unter dem Namen »Überladen von Methoden« bekannt.

Aufgabe 1.5 ☆

Eine Methode überladen

Definieren Sie eine Klasse `QuadratDefinition`, die ein Instanzfeld `a` vom Typ `int` besitzt, welches die Seitenlänge eines Quadrates angibt. Im Konstruktor der Klasse wird ein `int`-Wert zum Initialisieren des Instanzfeldes übergeben.

Implementieren Sie zwei Methoden für die Berechnung des Flächeninhaltes eines Quadrates mit der Formel f = a*a. Definieren Sie eine parameterlose Instanzmethode `flaeche()` und eine Klassenmethode, die die Instanzmethode überlädt und eine Referenz vom Typ der eigenen Klasse übergeben bekommt.

Die Klasse `QuadratDefinitionTest` erzeugt eine Instanz der Klasse `QuadratDefinition`, berechnet auf zwei Arten deren Flächeninhalt über den Aufruf der Methoden der Klasse und zeigt die errechneten Ergebnisse am Bildschirm an.

Java-Dateien: `QuadratDefinition.java`, `QuadratDefinitionTest.java`
Programmaufruf: `java QuadratDefinitionTest`

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

1.18 Lösungen

Lösung 1.1

Die Klasse KlassenDefinition

```
class KlassenDefinition {
   public static void main(String args[]) {
// Ausgabe einer Zeichenkette
     System.out.println("Definition einer einfachen "+
       "Java-Klasse");
   }
}
```

Programmausgaben

```
Definition einer einfachen Java-Klasse
```

Lösung 1.2

Die Klasse ObjektInstanziierung

```
class ObjektInstanziierung {
// Konstruktordefinition
   public ObjektInstanziierung() {
     System.out.println("Instanz einer Java-Klasse erzeugen");
   }
   public static void main(String args[]) {
     ObjektInstanziierung instanz = new ObjektInstanziierung();
   }
}
```

Programmausgaben

```
Instanz einer Java-Klasse erzeugen
```

Lösung 1.3

Die Klasse FeldZugriffe

```
class FeldZugriffe {
// Instanzfeld
   int zaehlerAlsInstanzfeld;
// Klassenfeld
   static int zaehlerAlsKlassenfeld;
// Konstruktordefinition
     FeldZugriffe() {
       zaehlerAlsKlassenfeld++;
       zaehlerAlsInstanzfeld++;
   }
```


© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

```
// Instanzmethoden, haben direkten Zugriff auf Instanz- und
// Klassenfelder
   public void anzeigeInstMeth() {
      System.out.print("Zaehler als Instanzfeld: "
        + zaehlerAlsInstanzfeld);
// Zeilenumbruch ausgeben
      System.out.println();
      System.out.print("Zaehler als Klassenfeld: "
        + FeldZugriffe.zaehlerAlsKlassenfeld);
// Innerhalb der Klasse kann auf den Klassennamen verzichtet
// werden
      System.out.print("*"+zaehlerAlsKlassenfeld);
      System.out.println();
   }
// Klassenmethoden haben direkten Zugriff auf Klassenfelder,
// darin können nur die Instanzfelder eines erzeugten Objektes
// der Klasse angesprochen werden
   public static void anzeigeKlsMeth() {
      System.out.print("Zaehler als Klassenfeld: "
       + zaehlerAlsKlassenfeld);
      System.out.print("*"+FeldZugriffe.zaehlerAlsKlassenfeld);
// Zeilenumbruch
      System.out.println();
    // System.out.println(zaehlerAlsInstanzfeld); // Fehler
// Instanz vom Typ der Klasse erzeugen
      FeldZugriffe t = new FeldZugriffe();
      System.out.print("Zaehler als Instanzfeld: "
        + t.zaehlerAlsInstanzfeld);
      System.out.print("*"+t.zaehlerAlsKlassenfeld);
   }
}
Die Klasse FeldZugriffeTest
class FeldZugriffeTest {
   public static void main(String args[]) {
      FeldZugriffe t = new FeldZugriffe();
      t.anzeigeInstMeth();
      FeldZugriffe.anzeigeKlsMeth();
   }
}
```

Programmausgaben

```
Zaehler als Instanzfeld: 1
Zaehler als Klassenfeld: 1*1
Zaehler als Klassenfeld: 1*1
Zaehler als Instanzfeld: 1*2
```


© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

Hinweise zu den Programmausgaben

Es ist zu erkennen, dass das Instanzfeld für jedes erzeugte Objekt neu initialisiert wird, das Klassenfeld jedoch für alle erzeugten Instanzen der Klasse zählt.

Lösung 1.4

Die Klasse MethodenAufrufe1

```
class MethodenAufrufe1 {
// Instanzmethoden
   public void instMethode1() {
      System.out.println("1. Instanzmethode der Klasse "
        + "MethodenAufrufe1");
// Aufruf einer Instanzmethode aus einer Instanzmethode der
// gleichen Klasse
      instMethode2();
// Aufruf einer Klassenmethode aus einer Instanzmethode der
// gleichen Klasse
      MethodenAufrufe1.klsMethode1();
   }
   public void instMethode2() {
      System.out.println("2. Instanzmethode der Klasse "
       + "MethodenAufrufe1");
   }
// Klassenmethoden
   public static void klsMethode1() {
      System.out.println("1. Klassenmethode der Klasse "
        + "MethodenAufrufe1");
   }
   public static void klsMethode2() {
      System.out.println("2. Klassenmethode der Klasse "
        + "MethodenAufrufe1");
// Aufruf einer Instanzmethode aus einer Klassenmethode der
// gleichen Klasse
      MethodenAufrufe1 instanz = new MethodenAufrufe1();
      instanz.instMethode2();
// Aufruf einer Klassenmethode aus einer Klassenmethode der
// gleichen Klasse
      MethodenAufrufe1.klsMethode1();
   }
}
```

Die Klasse MethodenAufrufe2

```
class MethodenAufrufe2 {
 // Instanzmethode
   public void instMethode() {
      System.out.println("Instanzmethode der Klasse "
```


© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

```
      + "MethodenAufrufe2");
// Aufruf einer Instanzmethode aus einer Instanzmethode einer
// anderen Klasse
     MethodenAufrufe1 instanz1 = new MethodenAufrufe1();
     instanz1.instMethode1();
// Aufruf einer Klassenmethode aus einer Instanzmethode einer
// anderen Klasse
     MethodenAufrufe1.klsMethode2();
   }
// Klassenmethode
   public static void klsMethode() {
     System.out.println("Klassenmethode der Klasse "
       + "MethodenAufrufe2");
// Aufruf einer Instanzmethode aus einer Klassenmethode einer
// anderen Klasse
     MethodenAufrufe1 instanz2 = new MethodenAufrufe1();
     instanz2.instMethode1();
// Aufruf einer Klassenmethode aus einer Klassenmethode einer
// anderen Klasse
     MethodenAufrufe1.klsMethode2();
   }
}
```

Die Klasse MethodenAufrufeTest

```
class MethodenAufrufeTest {
   public static void main(String args[]) {
     MethodenAufrufe2.klsMethode();
     MethodenAufrufe2 instanz = new MethodenAufrufe2();
     instanz.instMethode();
   }
}
```

Programmausgaben

```
Klassenmethode der Klasse MethodenAufrufe2
1. Instanzmethode der Klasse MethodenAufrufe1
2. Instanzmethode der Klasse MethodenAufrufe1
1. Klassenmethode der Klasse MethodenAufrufe1
2. Klassenmethode der Klasse MethodenAufrufe1
2. Instanzmethode der Klasse MethodenAufrufe1
1. Klassenmethode der Klasse MethodenAufrufe1
Instanzmethode der Klasse MethodenAufrufe2
1. Instanzmethode der Klasse MethodenAufrufe1
2. Instanzmethode der Klasse MethodenAufrufe1
1. Klassenmethode der Klasse MethodenAufrufe1
2. Klassenmethode der Klasse MethodenAufrufe1
```


© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

```
2. Instanzmethode der Klasse MethodenAufrufe1
1. Klassenmethode der Klasse MethodenAufrufe1
```

Lösung 1.5

Die Klasse QuadratDefinition

```
class QuadratDefinition  {
   int a;
// Konstruktordefinition
   QuadratDefinition(int x) {
      a = x;
   }
// Instanzmethode zum Berechnen des Flächeninhaltes
   public int flaeche() {
      int f = a*a;
      return f;
   }
// Gleichnamige Klassenmethode zum Berechnen des Flächeninhaltes
   public static int flaeche(QuadratDefinition q) {
      int f = q.a*q.a;
      return f;
   }
}
```

Die Klasse QuadratDefinitionTest

```
class QuadratDefinitionTest {
   public static void main(String args[]) {
      System.out.println("Instanz der Klasse erzeugen");
      QuadratDefinition quadrat = new QuadratDefinition(4);
      System.out.println("Aufruf der Instanzmethode");
      int finst = quadrat.flaeche();
      System.out.println("Flaeche: "+finst);
      System.out.println("Aufruf der Klassenmethode");
      int fkls2 = QuadratDefinition.flaeche(quadrat);
      System.out.println("Flaeche: "+fkls2);
   }
}
```

Programmausgaben

```
Instanz der Klasse erzeugen
Aufruf der Instanzmethode
Flaeche: 16
Aufruf der Klassenmethode
Flaeche: 16
```


© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

Stichwortverzeichnis

Symbole

...–Operator 58
<>-Operator *siehe* Diamond-Operator
==-Operator 53, 156, 158

A

Abbildungen *siehe* Map
Abklopfung 832
AbstractAction 393, 394
AbstractBorder 315, 319
AbstractButton 304, 312, 313, 315, 321, 525
AbstractMap 447, 519, 703
AbstractMap.SimpleEntry 703
AbstractMap.SimpleImmutableEntry 703
AbstractMultiResolutionImage 877
accept() 654, 678, 680
AccessibleObject 455
Accumulator 685, 695, 698
Action 393, 394
ActionEvent 211, 213, 214, 268, 391
ActionListener 213, 259, 266, 268, 390, 393, 395, 522, 642, 651
actionPerformed() 213, 259, 267, 393
add() 257, 266, 305, 309, 393, 568, 571, 642, 680, 695
addActionListener() 213
addAll() 695
addWindowListener() 212, 250
Adressraum 49
allProcesses() 844
and() 657
andThen() 679, 833
Ansicht *siehe* Viewer
Appletviewer 36
apply() 660, 675, 678
Arc2D.Float 321
Argumentenliste 58
Array 54, 55, 444, 445, 461
ArrayIndexOutOfBoundsException 585
Array-Initialisierer 35, 54
ArrayList 448, 694, 696, 705, 707
Array-Objekt 54, 55, 110, 160, 200, 444, 449
Arrays 54, 55, 580, 659, 699
ArrayStoreException 580
Arraytyp 33, 556, 579, 583
Arrayvariable 35
asList() 677, 690, 696
AtomicBoolean 852
AtomicInteger 147, 701
AtomicLong 147
Aufzählungen *siehe* Enumerationen
Aufzählungskonstanten 451
Ausnahme *siehe* Exception
Auto(un)Boxing 53, 148
Autoboxing 53
AutoCloseable 849, 892
average() 690
AWT
    Abstract Windowing Toolkit 189
AWT-Container 189, 207
AWTEvent 211
AWT-Komponente 189, 191, 203, 210, 262, 305

B

BaseMultiResolutionImage 877
BaseStream 672, 699
BasicArrowButton 440
BasicButtonUI 314
BasicLookAndFeel 323
BasicTextAreaUI 316
BasicTextFieldUI 324
Behälter 436, 437
Benutzer-Defaults 317
Beschriftung 278, 312, 314, 321, 394
Betriebssystem 49, 189, 190
BiConsumer 674, 678, 708
BiFunction 675, 678, 704, 709
BigDecimal 147, 148
BigInteger 147
Bild *siehe* Image
Bilddatei *siehe* Image
Bildschirmausgaben 36
BinaryOperator 679, 685, 688, 708
BiPredicate 678
Boolean 52
Border 315
BorderFactory 202
BorderLayout 201, 204, 213, 246, 249, 320
Box 191, 205
BoxLayout 202, 204

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

bridge methodes *siehe* Brückenmethoden
Brückenmethoden 113, 442, 443
BufferedReader 892
builder() 684, 691
Bulk-Operationen 669, 672, 698, 700, 702, 703, 709
Button 213, 214
ButtonModel 304
ButtonUI 317, 323
Byte 52, 147
Bytecode 35, 161, 650
byteValue() 53, 572

C

call by reference
    Referenzübergabe 42
call by value 41
    Wertübergabe 42
Callable 642
CallSite 650
cancel() 848, 853
Canvas 210, 246, 263, 264, 266, 393
catch-Block 49, 519, 585
ChangeEvent 211, 391
Character 52
characteristics() 698
Class 48, 155, 454, 458, 460, 517, 585
ClassCastException 555, 585
Classifier 686, 697
Class-Literal 560
ClassNotFoundException 524, 560, 584
Classpath 858
Class-Token *siehe* Class-Literal
Clipping Region 199
Clip-Rectangle 198, 199
clip-rectangle-Parameter 194
clone() 155, 156, 159, 160
Cloneable 155, 157, 160
CloneNotSupportedException 156
Code as data 644
collect() 686, 689, 695, 697, 698, 705, 878
Collection 268, 447, 448, 519, 657, 670, 673, 675, 683, 703
Collections 577, 642, 663
Collector 684, 686, 696, 698
Collectors 684, 686, 687, 696, 701, 705, 708
Color 196, 200, 203, 204, 267, 319, 385, 394
color-Parameter 194
ColorUIResource 318, 524
Combiner 685, 695, 698
ComboBoxModel 305
command() 844
Comparable 155, 158, 573, 577, 662
Comparator 577, 642, 659, 662
compare() 659
compareTo() 156, 159, 662
compareToIgnoreCase() 662
Component 189, 193, 196, 207, 210, 226, 246, 255
ComponentEvent 211
ComponentUI 311, 317
compute() 676, 708
computeIfAbsent() 676
concat() 56, 516, 693
ConcurrentHashMap 702, 703, 707
ConcurrentHashMap.KeySetView 703
ConcurrentMap 702, 705, 708
Constraints
    Einschränkungen 202
Constructor 454, 459, 517, 586, 587
Consumer 654, 655, 671, 677, 680, 704, 707, 708
Container 190, 198, 203, 207, 210, 246, 255, 440
contains() 321
ContentPane 255, 256, 259, 260, 320, 384, 642
Controller 191, 311, 323
Copy-Konstruktor 41, 159, 160
count() 685, 689
counting() 701
create() 886
createUI() 311

D

Datenkapselung 40
Datentyp 33, 149, 152, 154
deep cloning
    tiefes Klonen 159
DefaultButtonModel 304, 308
DefaultCaret 316
DefaultComboBoxModel 304
Default-Methode 666, 667, 673, 674, 684
DefaultMutableTreeNode 309, 310
DefaultTreeModel 309, 310
DefaultTreeSelectionModel 309
destroy() 843
destroyForcibly() 843
Destruktor 161
Dialog 190, 199, 392
Diamond-Operator 566, 570, 571, 667, 887
Dictionary 702
digit() 53
Dimension 197, 394
distinct() 699
Document 303
Double 52, 147, 439, 662
DoubleBinaryOperator 704
DoubleStream 672
doubleValue() 148
down-casting
    verkleinernde Konvertierung 116, 119, 141
draw() 265, 266, 388
drawImage() 260
drawString() 260
dropWhile() 830, 831

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

E

effectively final 641
Eigenschaften
  -background 201
  -bounds 207
  -clip-rectangle 195
  -contentAreaFilled 308, 313
  -focus 319
  -font 194, 453
  -font-ascent 196
  -font-descent 196
  -foreground 194
  -opaque 258, 273, 308
  -optimizedDrawingEnabled 195
elements() 452
Ellipse2D.Float 388
empty() 831
EmptyStackException 522
entrySet() 675, 697
Enum 450
enum *siehe* Enumerationen
Enumeration 452
Enumerationen 447, 450, 451, 519
  -Tag 451, 489, 490, 534
  -WTag 451, 488, 490, 534
EnumMap 447, 529, 702
equals() 155, 157, 158
Erasure 441
Ereignisbehandlung *siehe* Eventbehandlung
Ereignisse *siehe* Events
Error 49, 515
Erscheinungsbild 303, 317
Erweitern *siehe* Klasse -abgeleitete
eval() 886
Event-Adapter 211
Eventbehandlung 211, 214, 305, 310, 389
EventListener 211, 267, 268, 390, 392
EventListenerList 305, 306
EventObject 211, 268, 390, 392
Events auf höherer Ebene *siehe* Komponenten-Events
Events auf niedriger Ebene *siehe* Fester-Events
Exception 49, 515, 521, 681
exit() 49
exports 857
extends-Klausel 107, 563

F

Farbauswahldialog 393
Farbe *siehe* Color
Farbverlauf *siehe* GradientPaint
Felddeklarationen 58
Felder 33
  -geerbte 107, 109, 152
  -selbstreferenzierende 44, 438
  -verdeckte 114
Fenster 193
Fenster-Events
  Low-Level-Events 211
Field 459, 463
File 49, 520, 523
Files.lines() 834
fill() 266
fillOval() 197
filter() 671, 832
filtering() 841, 842
final 641, 652
finalize() 155, 161
finally-Block 515, 518, 519
findAny() 671
findFirst() 671
fireTreeNodesInserted() 310
flatMap() 832, 833, 835
flatMapping() 841, 842
Float 52, 147, 439
floatValue() 52, 148
Flow-API 847
FlowLayout 201, 204, 246, 249, 320
FocusEvent 211
Focus-Rectangle 308, 312, 313, 315
Font 196, 350, 385
FontMetrics 198
font-Parameter 194
FontUIResource 318, 524
forEach() 671, 673, 674, 680, 683, 694, 700, 704, 708
forEachKey() 708
forEachOrdered() 707
forEachRemaining() 673, 677, 700
for-each-Schleife 57, 446, 450
Fork/Join 669
forName() 455, 585
for-Schleife 56
Fragezeichen-Operator 247
Frame 190, 197, 203
Function 660, 664, 680, 687, 704, 709
Function descriptor 648, 679
Funktion 643

G

Garbage Collector 155, 161
generate() 684, 692, 705
GenericArrayType 458, 464
GenericDeclaration 458, 462
Generics 435
Generische Arrays 444, 446, 460
Generizität *siehe* Generics
get() 317, 450, 459, 460, 519, 660, 671, 684, 703, 832
getActualTypeArguments() 462, 464
getAndDecrement() 701
getAndIncrement() 701
getAscent() 198
getAsDouble() 690

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

getBounds() 190, 265, 463
getClass() 155, 445, 456, 573
getClipBounds() 199
getColor() 318, 319, 321
getComparator() 700, 706
getComponent() 316
getComponentZOrder() 256, 257
getConstructor() 460, 588
getCrossPlatformLookAndFeelClassName() 320
getDeclaredField() 459, 460, 461
getDeclaredFields() 463
getDefaults() 318, 319, 453
getDefaultToolkit() 196, 197
getDescent() 198
getDocument() 304
getenv() 49
getException() 523
getField() 460
getFocus() 313
getFontMetrics() 196
getGenericDeclaration() 463
getGenericInterfaces() 461
getGenericSuperclass() 461
getGenericType() 463
getGraphics() 194, 263, 267, 392
getImage() 198
getInsets() 315
getInstalledLookAndFeels() 320
getInt() 55
getInterfaces() 461
getItems() 306
getJLayeredPane() 257
getKeyChar() 214
getKeyCode() 214
getKeyModifiersText() 214
getKeyText() 214
getListeners() 306
getLogger() 870
getLowerBounds() 464
getMaximumSize() 192, 311
getMessage() 515, 524
getMinimumSize() 192, 311, 313
getModel() 304, 308
getModifiers() 214
getName() 461, 463
getNumericValue() 53
getOwnerType() 463
getPackage() 461
getParameterTypes() 457
getPreferredSize() 192, 199, 204, 210, 311, 313, 314, 315
getRawType() 462, 463
getResolutionVariant() 877
getResolutionVariants() 877
getReturnType() 457
getRGBColorComponents() 319
getRootPane() 255
getRuntime() 49
getScreenSize() 196, 197
getSuperclass() 461
getSystemLookAndFeelClassName() 320
getText() 321
getTypeParameters() 457, 458, 463
getUI() 312
getUpperBounds() 464
getValue() 440
getX() 190
getY() 190
GlassPane 255, 256, 259, 260
GradientPaint 315
Graphics 193, 196, 199, 260, 264, 265
Graphics2D 193, 265, 266, 388
Graphikkontext 194, 263, 267, 392
GregorianCalendar 60
GridBagConstraints 202, 205
GridBagLayout 205
GridLayout 201, 204, 214, 440
groupingBy() 688, 697, 701, 705, 708, 841
groupingByConcurrent() 705
GUI
    Graphical-User-Interface 189

H

happens-before 703, 704
hasFocus() 525
HashMap 447, 702, 703
hashMoreElements() 452
hashNext() 452
Hashtable 317, 447, 702, 703
hasNext() 672, 675, 683
hasNextLine() 677
hat ein-Beziehung 112, 117
headMap() 519
Heap 161
Heap-Pollution 567
Hintergrund 200
Hüllenklassen *siehe* Wrapper-Klassen
HW-Komponente 190, 192, 204, 206, 207

I

Icon 322
identity() 679, 687
if-Anweisung 53
ifPresent() 672, 832, 835
ifPresentOrElse() 837, 839
IllegalAccessException 517, 524, 584
IllegalArgumentException 517, 520
IllegalStateException 693, 707
Image 198, 348
ImageIcon 200, 259, 322
implements-Klausel 388, 564
import-Anweisung 45, 46, 47, 48, 109
incremental painting
    weiterführendes Zeichnen 194, 263

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

IndexOutOfBoundException 523
initCause() 523
initClassDefaults() 318, 323
initComponentDefaults() 318, 323
InputStream 48
insert() 310
Insets 190, 319
installDefaults() 316
installUI() 311, 321
instanceof-Operator 158, 458, 463, 554, 560
InstantiationException 517, 584
Instanz *siehe* Objekt
Instanz-Block-Initialisierer 58, 59
Instanzfelder 33, 34, 37
    -selbreferenzierende 59
Instanz-Feld-Initialisierer 58, 59
Instanzmethoden 33, 34, 37, 44, 50, 111, 152
    -dynamisches Binden 110, 111, 118
    -überschreiben 110, 114
Instanzvariable 35
IntBinaryOperator 704
Integer 52, 147, 438, 439, 581, 663
Interface 150, 151
    -ableiten 153
    -funktionales 643, 648
    -generisches 443, 452
    -Member 387, 388
    -Static-Member 384, 438
Interne Fenster *siehe* JInternalFrame
IntStream 672
intValue() 52, 53, 148, 572
InvocationTargetException 517, 585
invoke() 681
invokedynamic 644
invokeinterface 650
invokespecial 650
invokestatic 650
invokevirtual 650
isArmed() 308
isDirectory() 520
isFile() 520
isLoggable() 871
isOptimizedDrawingEnabled() 192
isPresent() 671
isPressed() 525
ist ein-Beziehung 111, 117
ItemEvent 211, 268, 391
ItemListener 268
Iterable 446, 447, 452, 657, 673, 676
iterate() 684, 692, 830, 831
Iterationen
    -externe 672, 673, 677
    -interne 672, 673, 677, 698
Iterationsvariablen 57
Iterator 452, 672, 677, 683
iterator() 452, 673, 675

J

Java
    -2D-API 193, 265
    -Anwendung 49
    -API 52
    -Applet 36
    -Applikation 36, 320
    -Event-Listener-API 212
    -Programme 35
    -Programmiersprache 52
    -Servlet 36
Java 5 49, 52, 53, 55, 56, 57, 113, 118, 147, 155, 156, 157, 190, 197, 208, 257, 435, 446, 450, 452, 453, 454
Java 7 556, 565, 566, 582, 584, 650
Java 8 641, 650, 699
Java Collection Framework 447
java.awt 192
java.awt.geom 265
java.awt.List 304, 305, 306
java.awt.peer 189
java.io 48
java.lang 48
java.lang.reflect 55, 454, 460, 586
java.math 148
java.util 48, 55, 447
java.util.function 655
java.util.stream 669
javax.swing 192
javax.swing.tree 309
JButton 257, 259, 266, 304, 323, 390, 440
JColorChooser 392, 393
JComboBox 304, 305, 307, 323, 438
JComponent 226, 249, 311, 321, 642
JDesktopPane 191, 192, 261
JDialog 191, 198, 202, 255, 392
JEditorPane 303, 308
JFileChooser 392
JFrame 191, 196, 197, 202, 255
JInternalFrame 191, 255, 257, 260, 261
JLabel 204, 206, 213, 214, 249, 259, 266, 391, 394, 522
JLayeredPane 191, 192, 255, 257, 258, 260, 261, 272, 394
JList 304
JMenu 257, 395
JMenuBar 255, 257, 260, 395
JMenuItem 304, 395
joining() 697
JOptionPane 392, 393
JPanel 191, 195, 201, 204, 210, 312, 323, 390, 392, 394
JRadioButton 268, 323, 391
JRootPane 191, 255
JScrollPane 191, 309, 316
JShell 881, 886
JShell-API 885

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

JSlider 323
JSplitPane 191
JTabbedPane 191
JTextArea 303, 308, 316
JTextComponent 303, 304, 312
JTextField 213, 268, 303, 308, 323, 390
JToggleButton 304
JToolBar 266, 393
JTree 304, 309, 310, 525
JVC
    Java Foundation Classes 189
JViewport 191, 192, 316
JVM
    Virtuelle Java Maschine 35, 36, 161, 436
JWindow 191, 198, 255

K

KeyEvent 211, 213, 214
KeyListener 213, 214
keyPressed() 214
keyReleased() 214
keys() 452, 453
keySet() 450
keyTyped() 214
Klasse 33
    -abgeleitete 107, 111, 112, 114
    -abstrakte 147, 151
    -anonyme 383, 387, 388, 390, 395, 641
    -generische 118, 437
    -innere 212, 383, 389
    -lokale 383, 386, 641
    -Member 383, 384, 385, 438
    -selbstreferenzierende 44
    -Static-Member 383, 385, 386
    -umgebende 384, 385, 386
Klassen-Block-Initialisierer 58, 59, 319
Klassendatei 45, 455
Klassendefinitionen
    -AbstractMapObjekteundExceptions 519, 530
    -AndersAnordnenTest 118, 143
    -AnordnenTest 117, 140
    -AnzeigevonLayoutManager 203, 225
    -Apfel 575, 604
    -ArraysKlonen 157, 178
    -ArrayTest1 55, 92
    -ArrayTest2 57, 97
    -AWTContainer 207, 237
    -AWTLabel 203, 228
    -AWTPaintComponent 210, 246
    -AWTPaintKreis 207, 236
    -BenutzerBorder 319, 363
    -BenutzerButtonUI 320, 363
    -BiConsumerBiPredicateundBiFunction 678, 756
    -Birne 575, 605
    -Bleistift 680, 760
    -Blumen 119, 144
    -BlumenTest 119, 145
    -BrueckenMethoden 443, 477
    -Buch 44, 78, 656, 729, 838, 841, 867, 951
    -BuchalsLayeredPane 394, 431
    -BuchBewertung 842, 867, 917, 955
    -BuchBewertungApp 867, 951
    -BuchComparator 662, 739
    -BuchKatalog 656, 732
    -Buchstabe 157, 178
    -BuchTest 45, 80
    -BuchUtils 662, 741
    -BuchVerkauf 654, 726
    -CanvasmitMouseEvent 267, 293
    -CollectorsMethoden 696, 795
    -ColorIcon 322, 369
    -ComponentmitPaint 264, 286
    -ComponentmitUpdate 264, 287
    -ConcurrentHashMapmitJava8 707, 818
    -ContainerfuerComponent 210, 245
    -CustomArea 316, 357
    -CustomAreaUI 316, 357
    -CustomBasicButtonUI 524, 546
    -CustomBorder 315, 354
    -CustomButton 524, 545
    -CustomButton1 524, 545
    -CustomButtonUI 316, 354
    -CustomCaret 316, 358
    -CustomField 323, 377
    -CustomFieldLookAndFeel 323, 379
    -CustomFieldUI 323, 378
    -CustomLookAndFeel 524, 547
    -Datum 865, 946
    -DatumUhrzeit 866, 946
    -DatumundUhrzeit 865, 945
    -DatumundUhrzeitApp 865, 944
    -DefaultLoggerFinder 872, 960
    -DefaultSystemLoggerJava9 872, 960
    -DefinitionvonFarben 200, 223
    -Destruktor 161, 187
    -DestruktorTest 161, 187
    -DialogmitFrameOwner 199, 222
    -Dialogmitrepaint 200, 222
    -DiamondOperator 571, 593
    -DiamondOperatorJava9 888, 982
    -DiamondOperatorJava9Test 888, 983
    -Dodekaeder 154, 176
    -Drachen 114, 135
    -DreieckArten 152, 169
    -DreieckArtenTest 152, 171
    -Enumerationen 451, 488
    -EnumerationenmitMethoden 451, 489
    -EnumerationIteratorIterable 452, 492
    -ExceptionKetten 523, 542
    -ExceptionsundFinally 518, 529
    -FarbenAuswahl 153, 173
    -FarbenTest 154, 173
    -FeldZugriffe 38, 61

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

-FeldZugriffeTest 38, 62
-FontIcon 322, 369
-FrameMaske 203, 227
-FramemitActionListener 213, 251
-FramemitContainerundPaintKreis 207, 238
-FramemitListenerKlasse 214, 253
-Framemitpaint 197, 215
-FramemitPaintKreis 207, 236
-Framemitrepaint 262, 281
-FramemitUpdate 264, 287
-FramemitWindowAdapter 213, 250
-FramemitWindowAdapterundAnonyme-Klasse 390, 414
-FramemitWindowAdapterundMemberKlasse 389, 413
-FramemitWindowListener 212, 249
-FramemitZOrder 208, 239
-FullStackException 522, 540
-GanzeZahlen 43, 73
-GenericArray 444, 478
-GenericArrayTypen 579, 621
-GenericArrayundGenericMethoden 446, 481
-GenericClass 458, 496
-GenericClassmitGenericArray 460, 501
-GenericClassmitGenericConstructor 459, 498
-GenericFields 462, 507
-GenericMapObjekte 449, 485, 683, 770
-GenericMapObjektemitForEach 450, 486
-GenericMethoden 445, 479
-GenericPunkt 442, 473
-GenericPunktmitBounds 572, 598
-GenericStack 438, 466
-GenericStackfuerInversWort 521, 538
-GenericStdKlsundInt 448, 483
-GenericSubTyp 443, 477
-GenericTyp 443, 477
-GenericTyp1 437, 464
-GenericTyp2 437, 465
-GenericTyp3 570, 589
-GenericTypmitExtend 439, 469
-GenericTypmitSuper 440, 471
-GenericTypmitWildcard 439, 468
-GeometrischeKoerper 148, 163
-GeometrischeKoerpermitGenericStdKls 448, 482
-GeometrischeKoerperTest 148, 164
-GeomFigMemberInterface 388, 409
-GescannteTastaturEingaben 755
-GleichschenkligesDreieck 152, 170
-GleichseitigesDreieck 152, 170
-GlobaleReferenzen 42, 71
-GrossBuchstaben 308, 333
-GrundMaskefuerAWT 203, 227
-GrundMaskefuerSwing 204, 229
-HalloJavamitAnonymerKlasse1 648, 709
-HalloJavamitAnonymerKlasse2 649, 710
-HalloJavamitForEach 676, 754
-HalloJavamitLambda 649, 710
-HalloJavamitMethodenReferenz 661, 738
-HalloJavaUtils 661, 739
-IconButton 308, 333
-Ikosaeder 154, 175
-InitialisierungInstanzfelder 59, 100
-InitialisierungKlassenfelder 59, 102
-InnereKlassemitmainMethode 393, 429
-InstanzZaehlerTest 110, 124
-InterfaceMethodenTest 891, 987
-ItemSubscriber 849, 930
-JavaTestKlasse 863, 943
-JButtonmitCustomButtonUI 315, 355
-JButtonmitIcon 313, 347
-JButtonmitUeberschrift 313, 349
-JButtonmitZeichnung 312, 346
-JComponentmitpaintundclipBounds 199, 220
-JComponentmitpaintundohneclipBounds 198, 219
-JDialogmitpaint 198, 218
-JFramefuerGeomFigMemberInterface 388, 410
-JFramefuerKomponentenmitZahlen 391, 418
-JFramefuerMemberKlassen 385, 398
-JFramefuerMemberKlassenStatic 386, 403
-JFramefuerRadioButtonEventundListener 391, 422
-JFramefuerUserEventundListener 268, 300
-JFramefuerUserKomponente 392, 425
-JFrameMaske 204, 229
-JFramemitActionListenerundAnonymeKlasse 390, 414
-JFramemitActionListenerundLambda 651, 718
-JFramemitBenutzerButtonUI 320, 364
-JFramemitBoxLayout 204, 232
-JFramemitContainerundPaintKreis 209, 243
-JFramemitDefaultComboBoxModel 304, 324
-JFramemitDefaultTreeModel 309, 337
-JFramemitDialogFenster 393, 427
-JFramemitGenericStackfuerInversWort 522, 540
-JFramemitgetGraphics 263, 283
-JFramemitgetLayeredPane 257, 269
-JFramemitGridBagLayout 205, 233
-JFramemitInternenFenster 261, 279
-JFramemitJInternalFrame 261, 278
-JFramemitJTree 309, 336
-JFramemitKeyListener 213, 252
-JFramemitKreiseMalenAnonymeKlasse 387, 406
-JFramemitKreiseMalenLokaleKlasse 386, 404

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

-JFramemitKreiseMalenLokaleKlasseundFinal 387
-JFramemitKreiseMalenMemberInterface 387, 407
-JFramemitKreiseMalenMemberInterfaceundMemberKlasse 389, 411
-JFramemitKreiseMalenMemberKlasse 384, 395
-JFramemitKreiseMalenStaticMemberKlasse 385, 400
-JFramemitLayeredPane 259, 272
-JFramemitLayeredPaneundGlassPane 259, 274
-JFramemitLookAndFeelAuswahl 322, 370
-JFramemitLookAndFeelundJTree 525, 549
-JFramemitLookAndFeelundStdButtonUI 323
-JFramemitLookAndFeelundTextFieldUI 324, 380
-JFramemitMatrix2D 266, 290
-JFramemitMatrix2DAnders 266, 291
-JFramemitMyButtonUI 321, 367
-JFramemitnewLayeredPane 258, 270
-JFramemitNullLayout 206, 235
-JFramemitpaint 197, 216
-JFramemitPaintKreis 208, 241
-JFramemitpaintundgetGraphics 263, 285
-JFramemitRectangle2DMemberInterface 388, 408
-JFramemitRectangle2DMemberInterfaceundLambda 651, 719
-JFramemitrepaint 262, 282
-JFramemitUIDefaultsHashTable 453, 495
-JFramemitUIDefaultsLesen 318, 360
-JFramemitUIDefaultsSetzen 319, 362
-JFramemitUserButtonUI 315, 353
-JFramemitUserComboBoxModel 307, 331
-JFramemitUserDefaultTreeModel 310, 341
-JFramemitUserModell 308, 334
-JFramemitVektor2D 265, 288, 680, 765
-JPanelmitLayeredPaneundGlassPane 260, 276
-JPanelmitpaint 312, 345
-JShellApiTestKlasse 886, 980
-JWindowmitpaint 198, 217
-Kegel 148, 164
-KindleEdition 654, 725
-Klasse 47, 80
-Klasse1 47, 81, 108, 120
-Klasse2 47, 81, 109, 120
-Klasse3 109, 121
-Klasse4 109, 122
-KlasseA 110, 123
-KlasseB 110, 123
-KlasseC 111, 125
-KlasseD 111, 125
-KlassenDefinition 36, 61
-KlassenInformationen 461, 502
-KlassenTest 47, 81
-KlonenTest 157, 178
-KollektorenJava9 842, 919
-Kombinationen 116, 138
-KombinationenausVariationen 117, 139
-KombinationenhabenPermutationen 117, 142
-KomponentenEvent 268, 297
-KomponentenmitZahlen 390, 415
-KomponentenUeberlappen 316, 359
-KongruenteDreiecke 50, 82
-KongruenteDreieckeTest 51, 84
-KonstruktorReferenz 664, 749
-KopievonReferenzTypen 160, 186
-KovarianteRueckgabeTypen 113, 131
-Kreis 51, 85
-KreisComponent 210, 245
-KreisContainer 209, 242
-KreishatPunkt 112, 127
-KreishatPunktTest 112, 113, 128
-KreisistPunkt 113, 129
-KreisistPunktTest 113, 130
-Kugel 148, 163
-LabelmitMouseEvent 267, 294
-Lehrer 149, 165
-LeseProdukt 459, 497
-Leser 838, 910
-ListenerKlasse 215, 253
-ListSetMapFactoryMethoden 828, 896
-Logger1 871, 872, 960, 964
-Logger2 871, 872, 960, 965
-MalenmitderMaus 266, 295
-MapvsFlatMap 835, 907
-MemberKlassen 385, 397
-MemberKlassenStatic 386, 401
-Menge 151, 167
-MengenDuplizieren 160, 183
-MengenDuplizierenTest 160, 185
-MengeTest 151, 168
-MethodenAufrufe1 38, 63
-MethodenAufrufe2 39, 63
-MethodenAufrufeTest 39, 64
-MethodenmitVararg 58, 98
-MethodenParameter 42, 69
-MethodenReferenzenfuerInstanzMethoden 665, 752
-MouseAdapterfuergetGraphics 263, 284
-MultipleExceptionTypen 585, 634
-MultiResolutionImagesJava9 880, 973
-MultiResolutionImagesJava9ohnePlatformLogging 878, 971
-MutableBuchstabe 653, 720
-MyButton 321, 364
-MyButtonUI 321, 365
-MyException 521, 536
-MyException1 523, 542
-MyLookAndFeel 323, 373
-MyResources 873, 963

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

-MyStdButtonUI 322, 373
-Name 459, 500
-NashiBirne 577, 605
-Nelke 119, 145
-NewInstanceException 517, 527
-Objekt 158, 180
-Objekt1 156, 177
-Objekt2 157, 179
-Objekte 113, 130
-ObjektInstanziierung 37, 61
-Obst 575, 604
-ObstUtils 742
-Oktaeder 154, 175
-OptionalFlatMapMethode 834, 905
-OptionalMethodenJava9 837, 838, 912
-PackageTest 46, 80
-PackageTest1 47, 82
-PaintKreis 208, 240
-ParalleleStreams 705, 803
-Parallelogramm 114, 134
-Permutationen 116, 137
-Person 149, 164, 834, 904
-PersonTest 149, 165
-PersonTyp 149, 165
-PersonTypTest 150, 166
-PlatformLoggerJava9 874, 967
-PointEqual 443, 476
-PrivaterKonstruktor 60
-PrivaterKonstruktorTest 60
-ProcessApiJava9 844, 923
-Produkt 461, 502
-ProgrammArgumentTest 520, 534
-ProgrammArgumentTestmitMyException 521, 536
-Punkt 40, 66, 112, 127
-PunktNeu 112, 129
-PunktStreckeKreisTest 52, 86
-PunktTest 40, 66
-Quadrat 114, 133
-QuadratDefinition 39, 65
-QuadratDefinitionTest 39, 65
-RadioButtonEventundListener 391, 420
-RationaleZahlen 43, 75
-RawTyp 442, 472
-ReactiveProgrammingJava9 849, 932
-RechenOperationenmitAnonymerKlasse 651, 716
-RechenOperationenmitLambda 651, 717
-Rechteck 114, 133
-ReflectionmitTypeInterface 463, 509
-RegelmaessigeGeometrischeKoerper 154, 174
-RegelmaessigeGeometrischeKoerperTest 154, 176
-RethrowingExceptionsmitJava7 587, 636
-Rose 119, 144
-SchreibProdukt 459, 497
-Schueler 149, 165
-Scopingofthis 653, 721
-ScopingundVariableCapture 653, 723
-Strasse 459, 500
-StreamFlatMapMethode 834, 902
-StreamMethoden 689, 774
-StreamMethodenJava9 830, 898
-Strecke 51, 84
-StringTest 56, 95
-SubTypingfuerReferenzTypen 583, 627
-SwingComponent 210, 247
-SwingContainer 210, 248
-SwingLabel 204, 230
-SwingPaintComponent 210, 248
-Tage 451, 489, 490, 534
-Tetraeder 154, 174
-TextAnalyzer1 869, 958
-TextAnalyzer2 869, 958
-TextAnalyzer3 869, 959
-TextAnalyzerApp 869, 956
-Tier 834, 904
-TierArzt 834, 904
-TransformProcessor 849, 931
-TryWithResourcesJava9 894, 988
-Tulpe 119, 144
-TypLehrer 150, 166
-TypSchueler 150, 166
-TypUmwandlungen 147, 162
-TypvariablenundParametrisierteTypen 573, 600
-Uhrzeit 865, 947
-UngepruefteExceptionsmitBehandlung 517, 526
-UngepruefteExceptionsohneBehandlung 516, 525
-UnterklassevonWindowAdapter 213, 251
-UserButton 268, 299
-UserButtonUI 314, 351
-UserButtonUI1 314, 351
-UserButtonUI2 314, 352
-UserComboBoxModel 307, 328
-UserComboBoxModelohneEventBehandlung 305, 326
-UserComboBoxModelohneEventBehandlungTest 306, 327
-UserComboBoxModelTest 307, 330
-UserDefaultTreeModel 311, 340
-UserEventundListener 392, 424
-UserGlassPane 260, 276
-UserKomponente 391, 423
-UserLoggerFinder 874, 966
-UserRadioButton 268, 299
-UserTextField 268, 298
-Variationen 116, 138
-VariationenhabenPermutationen 117, 141
-Vektor 41, 67
-VektorKopieren 159, 181
-VektorKopierenTest 159, 182
-VektorTest 41, 68

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

-ViereckeTest 115, 136
-WildcardTypen 575, 606, 663, 742
-WochenTage 60, 104
-WochenTageTest 60, 105
-Wohnort 459, 500
-Wort 160, 185
-Worte 113, 131
-WortPublisher 851, 935
-WortPublisherundSubscriber 851, 940
-WortSubscriber 851, 939
-WortSubscription 851, 936
-WrapperKlassen 53, 87
-WrapperKlassenmitAutoBoxing 53, 89
-WrapperKlassenmitAutoBoxingTest 54, 91
-WrapperKlassenTest 53, 88
-Wuerfel 154, 174
-Zahl 160, 185
-Zahlen 113, 131
-ZahlenTest 43, 76
Klassenfelder 33, 34, 38
-selbstreferenzierende 59
Klassen-Feld-Initialisierer 58, 59
Klassenhierarchie 33, 149
Klassenlader 35, 45
Klassenmethoden 33, 34, 37, 44, 51, 111, 152
-statisches Binden 109, 110
-verdecken 109
Klassenobjekt 455, 461, 463
Klassenparameter 437
Klassenvariablen 35
Klassifikationsfunktion *siehe* Classifier
Kollektoren 684, 686
Komponente 189
-bevorzugte Größe 191
-Eigenschaften 191
-Kind 189
-leichtgewichtige 190
-maximale Größe 191
-minimale Größe 191
-schwergewichtige 190
-Standard 193
-Vater 189
Komponenten-Events 211, 213
Komposition, Anreihung 112
Konsole 48
Konstantendefinition 33, 43, 152
Konstruktor 33, 34, 53, 451
-privater 59, 60
Konstruktordefinitionen 41
Konstruktor-Referenzen 659, 664, 687
Konvertierung *siehe* Typumwandlung
Kovariante Rückgabetypen 113, 442, 443

L

Label 203, 207
Lambda-Ausdruck 643
Lambda-Expression *siehe* Lambda-Ausdruck
LayeredPane 255, 256, 257, 259, 395
Layout-Manager 189, 191
LayoutManager 201, 203
LayoutManager2 201
Laziness 673
Legacy code 441
Level 870
limit() 684, 689, 692, 699
List 306, 448, 577, 657, 680, 827
ListDataEvent 307
ListDataListener 305, 307
ListModel 305
Locale 870
log() 872, 879
Logger 869
Logging-API 855
Long 52, 147
LongStream 672
look and feel 303
-spezifisch 195, 316, 317, 321, 524
LookAndFeel 317, 318, 322, 323, 524
LookAndFeel-Defaults 317
lower bound wildcard
untere Schranke 438, 440, 561
LW-Komponente 190, 192, 200, 206, 207

M

main() 36, 393, 395
Map 447, 452, 519, 674, 683, 684, 702, 827
Map.Entry 683
map() 671, 684, 696, 832, 835
MapEntry 828
Marker-Interface 155, 211, 318
Maskenprototyp 203
Math 48, 50, 51, 259
max() 689
Member 455
MenuBar 255
MetalButtonUI 314, 320, 322
MetalLookAndFeel 368
MetalScrollButton 440
MethodeHandle 650
Methoden 33, 52, 451
-erben 107, 129, 152, 313, 316
-generische 445, 446, 559, 573, 575
-statische 666
-überladen 39, 58
-überschreiben 149, 157
Methoden-Referenzen 659, 661, 662, 687, 709
min() 689
Model 191, 323
Modelle 304
-ComboBox 305, 307
-DefaultComboBox 304
-Tree 310

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

Modelle *siehe* Model
Modellklassen 304
Modifikator 33, 40, 46, 47, 107, 112
Modularisierung 855
Moduldeskriptordatei 856
Modul-Path 858
Modul-Source-Path 858
MotifLookAndFeel 368
MouseAdapter 263, 391
mouseClicked() 316
mouseDragged() 393
MouseEvent 211
MouseListener 259, 262, 263, 267, 391
MouseMotionListener 267, 393, 394
mousePressed() 262, 267
moveToBack() 395
moveToFront() 394
Multi-catch-Klausel 584
MultiResolutionImage 877
MVC-Architektur
    Model-View-Controller-Architektur 191, 303

N

negate() 657
newInstance() 55, 444, 445, 460, 517, 566, 585, 587, 588
new-Operator 33, 34, 35, 55, 59, 152, 161, 201, 384, 385, 435, 444, 458, 565, 641
next() 452, 672, 675, 683
nextElement() 452, 453
nextLine() 677
Noninterference 701
NoSuchFieldException 585
NoSuchMethodException 585
null-Layout 206
Number 147, 148, 439, 572
NumberFormatException 520

O

Oberinterfaces 153
Oberklasse, Superklasse 107, 110, 111, 116, 119, 147, 312
Oberklassenreferenz 118
Object 48, 111, 112, 151, 155, 156, 436
Objekt 33
    -aktuelle (aufrufende) 34, 40, 159
    -beseitigen 161
    -Gleichheit 158
    -klonen 156
Objektzustände 43, 200, 336
of() 671, 684, 691, 827, 829, 831, 838
ofNullable() 685, 829, 830, 831
onComplete() 847, 853, 854
onError() 847, 854
onExit() 843
onNext() 847, 853
onSubscribe() 854
onSubscription() 847
opens 857
Operation
    -Concurrent-Reduction 698, 705
    -group by 688
    -Multi-Level-Reduction 688
    -Mutable reduction 686
    -Reduction 698, 705
    -stateful 699
    -stateless 699
    -terminale 669
Operatoren 52
    -arithmetische 52
    -logische 52
Optional 671, 684, 690, 831, 835, 837
or() 657, 837
ordinal() 450
orElse() 671
orElseGet() 835
orElseThrow() 835
Out-of-order execution 673
override equivalent 648

P

pack() 196, 197, 199, 209
package-Anweisung 45, 46, 109
Packaging 860
paint() 194, 196, 197, 198, 199, 200, 207, 210, 262, 263, 264, 283, 311, 315, 316, 321, 388, 525
paintBorder() 195, 308, 313, 315, 319, 324
paintButtonPressed() 314
paintChildren() 195, 206, 210, 316
paintComponent() 195, 196, 199, 201, 260, 267, 308, 312, 313, 324, 384
Pakete 45, 47, 109, 317
Panel 190
parallel() 672, 698
parallelPrefix() 699, 706
parallelSetAll() 699, 706
parallelSort() 664, 699, 706
parallelStream() 698
ParameterizedType 458, 461, 463
Parametervariable 35
Parametrisierung 438, 556
parseDouble() 56
parseInt() 56, 516, 518
parseShort() 56
partitioningBy() 689, 841
peek() 671, 700
Pipeline 669
    Stufen 673
PlainDocument 303, 308
Platform-APIs 862
Point 257, 264, 394, 443
Polygon 388
Polymorphismus 118, 148

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

-implizite 118, 203, 435
-parametrisierte 118, 435
-Subtyp 119, 149
-überladene 118
Predicate 654, 655, 686
primitive Datentypen 33, 148, 158
printStackTrace() 515
PrintStream 36, 48, 677
Process 48
Process-API 843
ProcessBuilder 48, 49, 844
ProcessBuilder.start() 49
ProcessHandle 843
ProcessHandle.Info 843
Processor 847, 849
Programmdateien 47
Programmunterbrechung 520
provides 857
Prozess 49
Publisher 847, 849, 851
Punkt-Operator 44
put() 317, 319, 320, 323, 450, 519, 684
putAll() 519
putDefaults() 323
putIfAbsent() 684, 702

R

Rand *siehe* Border
random() 259
Raw-Typ 441, 442, 554, 560
Reactive Programming 847, 855
read() 878
Rectangle 388
Rectangle2D 265
Rectangle2D.Float 265, 266, 388
RectangleShape 265
reduce() 684, 685, 687, 689, 694, 697, 698, 704, 705
reduceEntries() 708
Reducer 704
reducing() 687, 688, 697, 705
Reduktion 684, 688
Referenzen 33
-globale 42
-lokale 42
Referenztyp 33, 118, 148, 150, 157, 160
Referenzvariable 35, 149, 155, 439
Reflection 454, 462, 585, 681
ReflectiveOperationException 584
remove() 310, 702
repaint() 194, 200, 262, 264
replace() 675, 683, 702
replaceAll() 675, 683
request() 848, 853
requires 857
Resizing 210
ResourceBundle 870
retainAll() 519
Rethrowing 584, 587
RGB-Modell 196, 319
RootLayout 255
RootPane 255, 395
RootPaneContainer 255
Runnable 642
Runtime 48, 49
Runtime.exec() 49
RuntimeException 516, 520, 521, 682

S

Scanner 677
Schnittstelle *siehe* Interface
Schnittstellendefinitionen
-Buchliste 657, 731, 838, 918
-Buchliste2 838, 911
-Buchliste3 838, 911
-Buchliste4 838, 912
-Equality 443, 476
-Farben 153, 172
-FarbenDunkel 153, 172
-FarbenHell 153, 172
-FarbeundSchrift 267, 293
-FlaechenInhalt 152, 169
-InKugelRadius 154, 173
-InterfaceDefinition 891, 986
-KomponentenListener 268, 298
-LabelFarben 205, 226
-PotporivonInterfaceMethoden 890, 985
-Services 150, 167
-StackInterface 521, 539
-TextAnalyzerService 868, 957
-UmKugelRadius 154, 173
Schrift *siehe* Font
Scoping 652
ScrollPane 190
search() 704, 708
searchValues() 708
sequential() 672
Serializable 581
Service 861
Service Consumer 861
Service Provider 861
Service-Interface 861, 868
ServiceLoader 861
ServiceLoader-API 870
Set 447, 452, 577, 675, 683, 827
setBackground() 197, 204, 394
setBounds() 189, 190, 198, 202, 206, 256, 258, 274, 394
setColor() 197
setComponentZOrder() 190, 208, 256
setDefaultCloseOperation() 193, 197
setDocument() 304
setInt() 55
setJMenuBar() 260

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

setLayer() 257
setLayeredPane() 258
setLayout() 201, 206
setLocation() 190, 202
setLookAndFeel() 318, 320, 321, 524, 525
setMaximumSize() 191
setMenuBar() 257, 395
setMinimumSize() 191
setModel() 304, 307, 310
setOpaque() 192, 200, 204, 208
setPreferredSize() 191, 197, 313
setSize() 190, 202
setUI() 315, 318, 324, 524
setVisible() 196, 216
shallow cloning
    oberflächliches Klonen 159
Shape 265, 266, 321, 388
Short 52, 147
showDialog() 393
Sichtbarkeitsebenen 40, 47, 108
Sichtbarkeitsmodifikatoren 857
skip() 689, 699
Snippet 882, 886
SnippetEvent 886
sort() 577, 659, 662, 664
sorted() 699
SortedMap 447, 519
SourceCodeAnalysis 886
split() 834
Spliterator 700, 707
Spliterator.OfDouble 707
Spliterator.OfInt 707
Spliterator.OfLong 707
Spliterators 707
Standardausgabe 48
Standardeingabe 48
Standard-Konstruktor 34
Standard-Layout-Manager 202, 203
startPipeline() 846
Steuerung *siehe* Controller
Stream 669, 672, 680, 684, 696, 704, 829, 835
stream() 670, 672, 689, 706, 837, 838, 840
Streams 684
    -parallele 672, 698, 701, 705
    -sequenzielle 672, 698, 701
StreamSupport 706
String 52, 56, 203, 204, 267, 394, 438, 581, 662
subMap() 519
SubmissionPublisher 849
subscribe() 847, 851, 854
Subscriber 847, 849, 851
Subscription 847
substring() 56, 516, 518
Subtyp 113, 118, 157, 207, 436, 554
Subtyping 562, 583
sum() 685
summingDouble 697
super.paint() 217, 219, 283
super.paintChildren() 206, 211
super.paintComponent() 195, 201, 206, 208, 211, 308
super() 107, 521
Supertyp 436, 554
Supplier 660, 664, 687, 695, 873
Swing 189
Swing-Container 189
Swing-Komponente 189, 191, 203, 204, 210, 255, 262, 305, 311
switch-Anweisung 43
System 36, 48, 515, 870
System.exit() 49
System.gc() 161
System.Logger 870
System.Logger.Level 871
System.LoggerFinder 870, 874
System.out.print() 36
System.out.println() 36
System-Defaults 317
Systemeigenschaften 49
SystemLoader 870

T

tailMap() 519
takeWhile() 830, 831
Target-Typ 645, 679, 687
Tastatureingaben 211
test() 654, 678
TextListener 268
this-Referenz 40, 652
Thread 643
Thread-Interferenz 701
Throwable 515, 523
throw-Anweisung 49, 585
throws 50, 516, 517, 585
TitledBorder 258
toArray() 449, 671, 695, 878
toBinaryString() 52
toConcurrentMap() 706, 707, 708
ToDoubleBiFunction 704
toHexString() 52
toList() 696
ToLongFunction 704
toMap() 693, 707
toOctalString() 53
Toolkit 196
toSet() 708
toString() 55, 112, 151, 155, 306, 451, 518
totalCpuDuration() 844
toUpperCase() 308
transitive 868
translate() 265
translation-Parameter 194
Transparenz-Eigenschaft *siehe* opaque-Eigenschaft
TreeMap 447, 449, 683, 702
TreeModel 309

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by
mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647

TreeModelListener 310
TreeNode 310
TreePath 310
TreeSelectionListener 310
TreeSelectionModel 309
tryAdvance() 700
try-Block 50, 515, 519, 585
trySplit() 700
try-with-resources 891
Typ
  -Disjunction 584, 585
  -dynamischer 555
  -generischer 436, 437, 443, 557
  -inferred 646
  -Intersection 557, 584
  -parametrisierter 435, 556, 558, 575, 583
  -reifizierbarer 441, 557
  -statischer 463, 553, 555, 568
  -synthetischer 644
  -Wildcard-parametrisierter 440, 562, 584
Typargument 435, 554, 575
Type 457, 461, 462, 463
TypeVariable 458, 463
Typinferenz 556, 564, 565, 571, 644
Typinvarianz 561, 575
Typisierung 556
Typkompatibilität 441
Typlöschung *siehe* Erasure
Typparameter 156, 435, 437, 554, 557
Typprüfung 436, 555, 584
Typsicherheit 53, 435, 441, 460, 556, 580
Typsystem 553
Typumwandlung 148, 436
Typvariable 435, 438, 556, 557

U

Überlappen von Komponenten 192, 206
Über-Typ 436, 437
UIDefaults 317, 323, 453
UIDefaults-Tabelle 318, 319, 320, 321, 322, 323, 453, 524
UI-Delegationsklasse 191, 195, 201, 303, 308, 311, 314, 316, 317, 321, 322, 323, 524
UI-Delegationsobjekt 191, 192, 201, 319
UIManager 317, 319, 320, 321
UIResource 317, 318, 323
Umgebungsvariable 45
UnaryOperator 679
uninstallUI() 311
unordered() 699, 702
UnsupportedLookAndFeelException 524
Unterklasse, Subklasse 107, 111, 117, 119, 147
up-casting
  vergrößernde Konvertierung 116, 119, 141, 149
update() 194, 195, 201, 262, 264, 267, 311
upper bound wildcard
  obere Schranke 438, 439, 561
user() 844
uses 857

V

value compatible 645
valueOf() 518
values() 450, 519, 703
varargs-Methoden 57, 553
Variable
  -lokale 35, 42, 58, 386, 393
Variable Capture 652
Vector 448
Vererbung 107, 111, 155
Verflachung 832
Viewer 191, 304, 307, 311, 323
void compatible 645
Vordergrund 318

W

Web Browser 36, 190
while-Schleife 45
Wildcard-Ausdruck *siehe* Wildcardtypen
Wildcard-Capture 569, 576, 578
WildcardType 458
Wildcardtypen 438, 445, 556, 561, 575
Window 190, 212
WindowAdapter 212, 389, 390
WindowEvent 211, 212, 214
WindowListener 212, 264
Wrapper-Klassen 52, 53

Z

Zeichenoberfläche 193, 210, 263, 267, 384, 388, 393
Zeichenoperationen 209, 261, 316
Zieltyp *siehe* Target-Typ
Z-Order 190
  -Index 190, 208, 209, 240, 244, 256, 257, 259, 270, 272, 273
Zugriffsmethoden 40, 44, 112, 116, 440

© des Titels »Java 9 – Das Übungsbuch« (ISBN 9783958456471) 2017 by mitp Verlags GmbH & Co. KG, Frechen. Nähere Informationen unter: http://www.mitp.de/647