

Inhalt

| | | |
|------------|---|-----------|
| 1 | Einführung | 19 |
| 1.1 | Was ist Java? | 20 |
| 1.1.1 | Java – die Sprache | 20 |
| 1.1.2 | Java – die Laufzeitumgebung | 21 |
| 1.1.3 | Java – die Standardbibliothek | 22 |
| 1.1.4 | Java – die Community | 23 |
| 1.1.5 | Die Geschichte von Java | 24 |
| 1.2 | Die Arbeitsumgebung installieren | 26 |
| 1.3 | Erste Schritte in Netbeans | 28 |
| 1.4 | Das erste Programm | 30 |
| 1.4.1 | Packages und Imports | 31 |
| 1.4.2 | Klassendefinition | 32 |
| 1.4.3 | Instanzvariablen | 33 |
| 1.4.4 | Der Konstruktor | 34 |
| 1.4.5 | Die Methode »count« | 35 |
| 1.4.6 | Die Methode »main« | 36 |
| 1.4.7 | Ausführen von der Kommandozeile | 38 |
| 1.5 | In Algorithmen denken, in Java schreiben | 40 |
| 1.5.1 | Beispiel 1: Fibonacci-Zahlen | 40 |
| 1.5.2 | Beispiel 2: Eine Zeichenkette umkehren | 43 |
| 1.5.3 | Algorithmisches Denken und Java | 45 |
| 1.6 | Die Java-Klassenbibliothek | 45 |
| 1.7 | Dokumentieren als Gewohnheit – Javadoc | 48 |
| 1.7.1 | Den eigenen Code dokumentieren | 49 |
| 1.7.2 | Package-Dokumentation | 52 |
| 1.7.3 | HTML-Dokumentation erzeugen | 53 |
| 1.7.4 | Was sollte dokumentiert sein? | 54 |
| 1.8 | JARs erstellen und ausführen | 54 |
| 1.8.1 | Die Datei »MANIFEST.MF« | 55 |
| 1.8.2 | JARs ausführen | 56 |
| 1.8.3 | JARs erzeugen | 57 |
| 1.8.4 | JARs einsehen und entpacken | 58 |

| | | |
|-------------|--|----|
| 1.9 | Mit dem Debugger arbeiten | 59 |
| 1.9.1 | Ein Programm im Debug-Modus starten | 59 |
| 1.9.2 | Breakpoints und schrittweise Ausführung | 60 |
| 1.9.3 | Variablenwerte und Callstack inspizieren | 61 |
| 1.9.4 | Übung: Der Debugger | 62 |
| 1.10 | Das erste eigene Projekt | 64 |
| 1.11 | Zusammenfassung | 65 |

2 Variablen und Datentypen 67

| | | |
|------------|---|----|
| 2.1 | Variablen | 67 |
| 2.1.1 | Der Zuweisungsoperator | 69 |
| 2.1.2 | Scopes | 69 |
| 2.1.3 | Primitive und Objekte | 70 |
| 2.2 | Primitivtypen | 70 |
| 2.2.1 | Zahlentypen | 71 |
| 2.2.2 | Rechenoperationen | 75 |
| 2.2.3 | Bit-Operatoren | 79 |
| 2.2.4 | Übung: Ausdrücke und Datentypen | 80 |
| 2.2.5 | Character-Variablen | 81 |
| 2.2.6 | Boolesche Variablen | 82 |
| 2.2.7 | Vergleichsoperatoren | 83 |
| 2.3 | Objekttypen | 84 |
| 2.3.1 | Werte und Referenzen | 85 |
| 2.3.2 | Der Wert »null« | 85 |
| 2.3.3 | Vergleichsoperatoren | 86 |
| 2.3.4 | Allgemeine und spezielle Typen | 87 |
| 2.3.5 | Strings – primitive Objekte | 88 |
| 2.4 | Objekt-Wrapper zu Primitiven | 89 |
| 2.4.1 | Warum? | 89 |
| 2.4.2 | Explizite Konvertierung | 90 |
| 2.4.3 | Implizite Konvertierung | 90 |
| 2.5 | Array-Typen | 92 |
| 2.5.1 | Deklaration eines Arrays | 92 |
| 2.5.2 | Zugriff auf ein Array | 93 |
| 2.6 | Zusammenfassung | 94 |

| | |
|---|-----|
| 3.1 Entweder-oder-Entscheidungen | 95 |
| 3.1.1 Übung: Star Trek – sehen oder nicht? | 96 |
| 3.1.2 Mehrfache Verzweigungen | 99 |
| 3.1.3 Übung: Body-Mass-Index | 100 |
| 3.1.4 Der ternäre Operator | 100 |
| 3.2 Logische Verknüpfungen | 101 |
| 3.2.1 Boolesche Operatoren | 102 |
| 3.2.2 Verknüpfungen mit und ohne Kurzschluss | 103 |
| 3.2.3 Übung: Boolesche Operatoren | 104 |
| 3.2.4 Übung: Solitaire | 106 |
| 3.3 Mehrfach verzweigen mit »switch« | 108 |
| 3.3.1 »switch« mit Strings, Zeichen und Zahlen | 109 |
| 3.3.2 Übung: »Rock im ROM« | 110 |
| 3.3.3 Enumerierte Datentypen und »switch« | 111 |
| 3.3.4 Durchfallendes »switch« | 112 |
| 3.3.5 Übung: »Rock im ROM« bis zum Ende | 112 |
| 3.3.6 Übung: »Rock im ROM« solange ich will | 113 |
| 3.3.7 Der Unterschied zwischen »switch« und »if... else if ...« | 113 |
| 3.4 Zusammenfassung | 113 |

4 Wiederholungen

| | |
|---|-----|
| 4.1 Bedingte Wiederholungen mit »while« | 115 |
| 4.1.1 Kopfgesteuerte »while«-Schleife | 116 |
| 4.1.2 Übung: Das kleinste gemeinsame Vielfache | 117 |
| 4.1.3 Fußgesteuerte »while«-Schleifen | 117 |
| 4.1.4 Übung: Zahlen raten | 118 |
| 4.2 Abgezählte Wiederholungen – die »for«-Schleife | 119 |
| 4.2.1 Übung: Zahlen validieren | 120 |
| 4.3 Abbrechen und überspringen | 121 |
| 4.3.1 »break« und »continue« mit Labels | 122 |
| 4.4 Zusammenfassung | 124 |

| | | |
|-------------|--|-----|
| 5.1 | Klassen und Objekte | 126 |
| 5.1.1 | Klassen anlegen | 126 |
| 5.1.2 | Objekte erzeugen | 127 |
| 5.2 | Access Modifier | 128 |
| 5.3 | Felder | 130 |
| 5.3.1 | Felder deklarieren | 130 |
| 5.3.2 | Zugriff auf Felder | 130 |
| 5.4 | Methoden | 131 |
| 5.4.1 | Übung: Eine erste Methode | 133 |
| 5.4.2 | Rückgabewerte | 133 |
| 5.4.3 | Übung: Jetzt mit Rückgabewerten | 135 |
| 5.4.4 | Parameter | 135 |
| 5.4.5 | Zugriffsmethoden | 137 |
| 5.4.6 | Übung: Zugriffsmethoden | 139 |
| 5.5 | Warum Objektorientierung? | 140 |
| 5.6 | Konstruktoren | 142 |
| 5.6.1 | Konstruktoren deklarieren und aufrufen | 142 |
| 5.6.2 | Übung: Konstruktoren | 145 |
| 5.7 | Statische Felder und Methoden | 146 |
| 5.7.1 | Übung: Statische Felder und Methoden | 147 |
| 5.7.2 | Die »main«-Methode | 148 |
| 5.7.3 | Statische Importe | 148 |
| 5.8 | Unveränderliche Werte | 149 |
| 5.8.1 | Unveränderliche Felder | 150 |
| 5.8.2 | Konstanten | 150 |
| 5.9 | Spezielle Objektmethoden | 152 |
| 5.10 | Zusammenfassung | 154 |

6 Objektorientierung

| | | |
|------------|------------------------------------|-----|
| 6.1 | Vererbung | 156 |
| 6.1.1 | Vererbung implementieren | 157 |
| 6.1.2 | Übung: Tierische Erbschaften | 159 |

| | | |
|------------|--|------------|
| 6.1.3 | Erben und überschreiben von Members | 159 |
| 6.1.4 | Vererbung und Konstruktoren | 164 |
| 6.1.5 | Übung: Konstruktoren und Vererbung | 165 |
| 6.1.6 | Vererbung verhindern | 165 |
| 6.1.7 | Welchen Typ hat das Objekt? | 167 |
| 6.2 | Interfaces und abstrakte Datentypen | 169 |
| 6.2.1 | Abstrakte Klassen | 170 |
| 6.2.2 | Interfaces | 171 |
| 6.2.3 | Default-Implementierungen | 174 |
| 6.3 | Übung: Objektorientierte Modellierung | 176 |
| 6.4 | Innere Klassen | 177 |
| 6.4.1 | Statische innere Klassen | 178 |
| 6.4.2 | Nichtstatische innere Klassen | 180 |
| 6.4.3 | Anonyme Klassen | 183 |
| 6.5 | Enumerationen | 185 |
| 6.6 | Zusammenfassung | 187 |
| | | |
| 7 | Unit Testing | 189 |
| <hr/> | | |
| 7.1 | Das JUnit-Framework | 191 |
| 7.1.1 | Der erste Test | 192 |
| 7.1.2 | Die Methoden von »Assert« | 194 |
| 7.1.3 | Testfälle ausführen in NetBeans | 194 |
| 7.1.4 | Übung: Den GGT-Algorithmus ändern | 196 |
| 7.1.5 | Übung: Tests schreiben für das KGV | 197 |
| 7.2 | Fortgeschrittene Unit Tests | 197 |
| 7.2.1 | Testen von Fehlern | 198 |
| 7.2.2 | Vor- und Nachbereitung von Tests | 199 |
| 7.2.3 | Mocking | 200 |
| 7.3 | Besseres Design durch Testfälle | 203 |
| 7.3.1 | Übung: Testfälle für den BMI-Rechner | 205 |
| 7.4 | Zusammenfassung | 206 |

| | |
|---|-----|
| 8.1 Zahlen | 207 |
| 8.1.1 »Number« und die Zahlentypen | 207 |
| 8.1.2 Mathematisches aus »java.lang.Math« | 208 |
| 8.1.3 Übung: Satz des Pythagoras | 211 |
| 8.1.4 »BigInteger« und »BigDecimal« | 211 |
| 8.1.5 Übung: Fakultäten | 212 |
| 8.2 Strings | 213 |
| 8.2.1 Unicode | 213 |
| 8.2.2 »String«-Methoden | 214 |
| 8.2.3 Übung: Namen zerlegen | 218 |
| 8.2.4 Übung: Römische Zahlen I | 218 |
| 8.2.5 »StringBuilder« | 219 |
| 8.2.6 Übung: Römische Zahlen II | 221 |
| 8.2.7 »StringTokenizer« | 221 |
| 8.3 Reguläre Ausdrücke | 222 |
| 8.3.1 Einführung in reguläre Ausdrücke | 222 |
| 8.3.2 String-Methoden mit regulären Ausdrücken | 224 |
| 8.3.3 Reguläre Ausdrücke als Objekte | 226 |
| 8.3.4 Übung: Flugnummern finden | 228 |
| 8.4 Zeit und Datum | 229 |
| 8.4.1 Zeiten im Computer und »java.util.Date« | 229 |
| 8.4.2 Neue Zeiten – das Package »java.time« | 229 |
| 8.4.3 Übung: Der Fernsehkalender | 233 |
| 8.5 Internationalisierung und Lokalisierung | 234 |
| 8.5.1 Internationale Nachrichten mit »java.util.ResourceBundle« | 235 |
| 8.5.2 Nachrichten formatieren mit »java.util.MessageFormat« | 236 |
| 8.5.3 Zeiten und Daten lesen | 239 |
| 8.5.4 Zahlen lesen | 241 |
| 8.6 Zusammenfassung | 242 |

| | |
|--|-----|
| 9.1 Exceptions werfen und behandeln | 243 |
| 9.1.1 try-catch | 245 |
| 9.1.2 Übung: Fangen und noch mal versuchen | 247 |
| 9.1.3 try-catch-finally | 247 |

| | | |
|------------|--|------------|
| 9.1.4 | try-with-resources | 249 |
| 9.1.5 | Fehler mit Ursachen | 250 |
| 9.2 | Verschiedene Arten von Exceptions | 250 |
| 9.2.1 | Unchecked Exceptions | 251 |
| 9.2.2 | Checked Exceptions | 252 |
| 9.2.3 | Errors | 255 |
| 9.3 | Invarianten, Vor- und Nachbedingungen | 255 |
| 9.4 | Zusammenfassung | 257 |

10 Arrays und Collections 259

| | | |
|-------------|---|------------|
| 10.1 | Arrays | 259 |
| 10.1.1 | Grundlagen von Arrays | 260 |
| 10.1.2 | Übung: Primzahlen | 262 |
| 10.1.3 | Mehrdimensionale Arrays | 263 |
| 10.1.4 | Übung: Das pascalsche Dreieck | 264 |
| 10.1.5 | Utility-Methoden in »java.util.Arrays« | 264 |
| 10.1.6 | Übung: Sequenziell und parallel sortieren | 268 |
| 10.2 | Die for-each-Schleife | 268 |
| 10.3 | Variable Parameterlisten | 269 |
| 10.4 | Collections | 270 |
| 10.4.1 | Listen und Sets | 272 |
| 10.4.2 | Iteratoren | 275 |
| 10.4.3 | Übung: Musiksammlung und Playlist | 276 |
| 10.5 | Typisierte Collections – Generics | 276 |
| 10.5.1 | Generics außerhalb von Collections | 278 |
| 10.5.2 | Eigenen Code generifizieren | 280 |
| 10.5.3 | Übung: Generisches Filtern | 286 |
| 10.6 | Maps | 286 |
| 10.6.1 | Übung: Lieblingslieder | 288 |
| 10.7 | Zusammenfassung | 288 |

11 Lambda-Ausdrücke 289

| | | |
|-------------|---|------------|
| 11.1 | Was sind Lambda-Ausdrücke? | 290 |
|-------------|---|------------|

| | | |
|-------------|---|------------|
| 11.1.1 | Die Lambda-Syntax | 291 |
| 11.1.2 | Wie funktioniert das? | 294 |
| 11.1.3 | Übung: Zahlen selektieren | 297 |
| 11.1.4 | Funktionale Interfaces nur für Lambda-Ausdrücke | 297 |
| 11.1.5 | Übung: Funktionen | 302 |
| 11.2 | Die Stream-API | 302 |
| 11.2.1 | Intermediäre und terminale Methoden | 304 |
| 11.2.2 | Übung: Temperaturdaten auswerten | 314 |
| 11.2.3 | Endlose Streams | 315 |
| 11.2.4 | Übung: Endlose Fibonacci-Zahlen | 315 |
| 11.2.5 | Daten aus einem Stream sammeln – »Stream.collect« | 316 |
| 11.2.6 | Übung: Wetterstatistik für Fortgeschrittene | 319 |
| 11.3 | Un-Werte als Objekte – »Optional« | 319 |
| 11.3.1 | Die wahre Bedeutung von »Optional« | 321 |
| 11.4 | Eine Warnung zum Schluss | 322 |
| 11.5 | Zusammenfassung | 323 |

12 Dateien, Streams und Reader 325

| | | |
|-------------|---|------------|
| 12.1 | Dateien und Verzeichnisse | 326 |
| 12.1.1 | Dateien und Pfade | 326 |
| 12.1.2 | Dateioperationen aus »Files« | 329 |
| 12.1.3 | Übung: Dateien kopieren | 329 |
| 12.1.4 | Verzeichnisse | 330 |
| 12.1.5 | Übung: Musik finden | 331 |
| 12.2 | Reader, Writer und die »anderen« Streams | 332 |
| 12.2.1 | Lesen und Schreiben von Textdaten | 333 |
| 12.2.2 | Übung: Playlists – jetzt richtig | 339 |
| 12.2.3 | »InputStream« und »OutputStream« – Binärdaten | 340 |
| 12.2.4 | Übung: ID3-Tags | 342 |
| 12.3 | Objekte lesen und schreiben | 344 |
| 12.3.1 | Serialisierung | 344 |
| 12.4 | Netzwerkkommunikation | 347 |
| 12.4.1 | Übung: Dateitransfer | 349 |
| 12.5 | Zusammenfassung | 350 |

| | | |
|-------------|---|-----|
| 13.1 | Threads und Runnables | 352 |
| 13.1.1 | Threads starten und Verhalten übergeben | 352 |
| 13.1.2 | Übung: Multithreaded Server | 356 |
| 13.1.3 | Geteilte Ressourcen | 356 |
| 13.2 | Atomare Datentypen | 359 |
| 13.3 | Synchronisation | 360 |
| 13.3.1 | »synchronized« als Modifikator für Methoden | 362 |
| 13.3.2 | Das synchronized-Statement | 362 |
| 13.3.3 | Deadlocks | 365 |
| 13.3.4 | Übung: Zufallsverteilung | 367 |
| 13.4 | Fortgeschrittene Koordination zwischen Threads | 367 |
| 13.4.1 | Signalisierung auf dem Monitor-Objekt | 368 |
| 13.4.2 | Daten produzieren, kommunizieren und konsumieren | 371 |
| 13.4.3 | Threads wiederverwenden | 373 |
| 13.5 | Die Zukunft – wortwörtlich | 374 |
| 13.5.1 | Lambdas und die Zukunft – »CompletableFuture« | 376 |
| 13.6 | Das Speichermodell von Threads | 378 |
| 13.7 | Zusammenfassung | 380 |

14 Servlets – Java im Web

| | | |
|-------------|--|-----|
| 14.1 | Einen Servlet-Container installieren | 382 |
| 14.1.1 | Installation des Tomcat-Servers | 382 |
| 14.1.2 | Den Tomcat-Server in Netbeans einrichten | 386 |
| 14.2 | Die erste Servlet-Anwendung | 388 |
| 14.2.1 | Die Anwendung starten | 390 |
| 14.2.2 | Was passiert, wenn Sie die Anwendung aufrufen? | 393 |
| 14.3 | Servlets programmieren | 399 |
| 14.3.1 | Servlets konfigurieren | 400 |
| 14.3.2 | Mit dem Benutzer interagieren | 401 |
| 14.3.3 | Übung: Das Rechen-Servlet implementieren | 404 |
| 14.4 | Java Server Pages | 406 |
| 14.4.1 | Übung: Playlisten anzeigen | 411 |
| 14.4.2 | Übung: Musik abspielen | 411 |

| | | |
|-------------|---|------------|
| 14.5 | Langlebige Daten im Servlet – Ablage in Session und Application | 412 |
| 14.5.1 | Die »HTTPSession« | 413 |
| 14.5.2 | Übung: Daten in der Session speichern | 414 |
| 14.5.3 | Der Application Context | 414 |
| 14.6 | Fortgeschrittene Servlet-Konzepte – Listener und Initialisierung | 414 |
| 14.6.1 | Listener | 415 |
| 14.6.2 | Übung: Die Playliste nur einmal laden | 416 |
| 14.6.3 | Initialisierungsparameter | 416 |
| 14.7 | Zusammenfassung | 418 |

15 Datenbanken und Entitäten 419

| | | |
|-------------|---|------------|
| 15.1 | Was ist eine Datenbank? | 420 |
| 15.1.1 | Relationale Datenbanken | 420 |
| 15.1.2 | JDBC | 424 |
| 15.1.3 | JPA | 425 |
| 15.2 | Mit einer Datenbank verbinden über die JPA | 427 |
| 15.2.1 | Datenbank in Netbeans anlegen | 427 |
| 15.2.2 | Das Projekt anlegen | 428 |
| 15.2.3 | Eine Persistence Unit erzeugen | 429 |
| 15.2.4 | Die »EntityManagerFactory« erzeugen | 430 |
| 15.3 | Anwendung und Entitäten | 431 |
| 15.3.1 | Die erste Entität anlegen | 432 |
| 15.3.2 | Übung: Personen speichern | 434 |
| 15.4 | Entitäten laden | 435 |
| 15.4.1 | Abfragen mit JPQL | 435 |
| 15.4.2 | Übung: Personen auflisten | 437 |
| 15.4.3 | Entitäten laden mit ID | 437 |
| 15.4.4 | Übung: Personen bearbeiten | 438 |
| 15.4.5 | Benannte Queries | 439 |
| 15.5 | Entitäten löschen | 440 |
| 15.6 | Beziehungen zu anderen Entitäten | 441 |
| 15.6.1 | Eins-zu-eins-Beziehungen | 441 |
| 15.6.2 | Übung: Kontakte mit Adressen | 444 |
| 15.6.3 | Eins-zu-vielen-Beziehungen | 444 |

| | | |
|-------------|---|------------|
| 15.6.4 | Viele-zu-eins-Beziehungen | 445 |
| 15.6.5 | Beziehungen in JPQL | 447 |
| 15.7 | Zusammenfassung | 448 |
| | | |
| 16 | GUIs mit JavaFX (Gastbeitrag von Philip Ackermann) | 449 |
| <hr/> | | |
| 16.1 | Einführung | 449 |
| 16.2 | Installation | 450 |
| 16.3 | Architektur von JavaFX | 450 |
| 16.3.1 | Application | 451 |
| 16.3.2 | Scenes | 452 |
| 16.3.3 | Scene Graph | 452 |
| 16.3.4 | Typen von Nodes | 453 |
| 16.4 | GUI-Komponenten | 453 |
| 16.4.1 | Beschriftungen | 454 |
| 16.4.2 | Schaltflächen | 454 |
| 16.4.3 | Checkboxen und Choiceboxen | 456 |
| 16.4.4 | Eingabefelder | 458 |
| 16.4.5 | Menüs | 458 |
| 16.4.6 | Sonstige Standardkomponenten | 460 |
| 16.4.7 | Geometrische Komponenten | 463 |
| 16.4.8 | Diagramme | 463 |
| 16.5 | Layouts | 464 |
| 16.5.1 | BorderPane | 464 |
| 16.5.2 | HBox | 466 |
| 16.5.3 | VBox | 466 |
| 16.5.4 | StackPane | 467 |
| 16.5.5 | GridPane | 468 |
| 16.5.6 | FlowPane | 469 |
| 16.5.7 | TilePane | 470 |
| 16.5.8 | AnchorPane | 471 |
| 16.5.9 | Fazit | 473 |
| 16.6 | GUI mit Java-API – Urlaubsverwaltung | 474 |
| 16.6.1 | Initialisierung des Menüs | 475 |
| 16.6.2 | Initialisierung der Tabs | 475 |
| 16.6.3 | Initialisierung des Inhalts von Tab 1 | 476 |
| 16.6.4 | Initialisierung des Inhalts von Tab 2 | 477 |

| | | |
|--------------|---|-----|
| 16.7 | Event-Handling | 479 |
| 16.7.1 | Events und Event-Handler | 479 |
| 16.7.2 | Typen von Events | 481 |
| 16.7.3 | Alternative Methoden für das Registrieren von Event-Handle | 484 |
| 16.8 | JavaFX-Properties und Binding | 485 |
| 16.8.1 | JavaFX-Properties | 485 |
| 16.8.2 | JavaFX-Properties und Listener | 487 |
| 16.8.3 | JavaFX-Properties im GUI | 487 |
| 16.8.4 | JavaFX-Properties von GUI-Komponenten | 489 |
| 16.8.5 | Binding | 489 |
| 16.9 | Deklarative GUIs mit FXML | 491 |
| 16.9.1 | Vorteile gegenüber programmatisch erstellten GUIs | 491 |
| 16.9.2 | Einführung | 493 |
| 16.9.3 | Aufruf eines FXML-basierten GUI | 494 |
| 16.9.4 | Event-Handling in FXML | 495 |
| 16.10 | Layout mit CSS | 497 |
| 16.10.1 | Einführung in CSS | 497 |
| 16.10.2 | JavaFX-CSS | 497 |
| 16.10.3 | JavaFX-Anwendung mit CSS | 498 |
| 16.10.4 | Urlaubsverwaltung mit JavaFX-CSS | 498 |
| 16.11 | Transformationen, Animationen und Effekte | 501 |
| 16.11.1 | Transformationen | 501 |
| 16.11.2 | Animationen | 504 |
| 16.12 | Zusammenfassung | 508 |
| | | |
| 17 | Hinter den Kulissen | 509 |
| <hr/> | | |
| 17.1 | Klassenpfade und Class Loading | 509 |
| 17.1.1 | Klassen laden in der Standardumgebung | 510 |
| 17.1.2 | Ein komplexeres Szenario – Klassen laden im Servlet-Container | 511 |
| 17.1.3 | ClassLoader und Klassengleichheit | 513 |
| 17.1.4 | ClassLoader als Objekte | 514 |
| 17.2 | Garbage Collection | 515 |
| 17.2.1 | Speicherlecks in Java | 518 |
| 17.2.2 | Weiche und schwache Referenzen | 519 |

| | | |
|-------------------------|---|-----|
| 17.3 | Flexibel codieren mit der Reflection-API | 521 |
| 17.3.1 | Übung: Templating | 526 |
| 17.4 | Zusammenfassung | 527 |
| 18 Und dann? | | 529 |
| <hr/> | | |
| 18.1 | Java Enterprise Edition | 530 |
| 18.1.1 | Servlet | 530 |
| 18.1.2 | JPA | 532 |
| 18.1.3 | Enterprise Java Beans | 532 |
| 18.1.4 | Java Messaging Service | 533 |
| 18.1.5 | Java Bean Validation | 534 |
| 18.2 | Open-Source-Software | 534 |
| 18.3 | Android | 535 |
| 18.4 | Ergänzende Technologien | 536 |
| 18.4.1 | SQL und DDL | 536 |
| 18.4.2 | HTML, CSS und JavaScript | 537 |
| 18.5 | Andere Sprachen | 539 |
| 18.5.1 | Scala | 540 |
| 18.5.2 | Clojure | 540 |
| 18.5.3 | JavaScript | 541 |
| 18.6 | Programmieren Sie! | 541 |
| Anhang | | 543 |
| <hr/> | | |
| A | Java-Bibliotheken | 545 |
| B | Lösungen zu den Übungsaufgaben | 553 |
| C | Glossar | 653 |
| D | Kommandozeilenparameter | 667 |
| Index | | 673 |