

# Inhaltsverzeichnis

<b>Vorwort</b> . . . . .	<b>17</b>
<b>Einleitung</b> . . . . .	<b>19</b>
Java – mehr als nur kalter Kaffee? . . . . .	19
Java für Anfänger – das Konzept dieses Buches . . . . .	20
Zusatzmaterial und Kontakt zu den Autoren . . . . .	21
Verwendete Schreibweisen . . . . .	22
<b>I Einstieg in das Programmieren in Java</b> . . . . .	<b>23</b>
<b>1 Einige Grundbegriffe aus der Welt des Programmierens</b> . . . . .	<b>25</b>
1.1 Computer, Software, Informatik und das Internet . . . . .	25
1.2 Was heißt Programmieren? . . . . .	27
1.3 Welche Werkzeuge brauchen wir? . . . . .	30
<b>2 Aller Anfang ist schwer</b> . . . . .	<b>33</b>
2.1 Mein erstes Programm . . . . .	33
2.2 Formeln, Ausdrücke und Anweisungen . . . . .	34
2.3 Zahlenbeispiele . . . . .	35
2.4 Verwendung von Variablen . . . . .	36
2.5 „Auf den Schirm!“ . . . . .	36
2.6 Das Programmgerüst . . . . .	37
2.7 Eingeben, übersetzen und ausführen . . . . .	39
2.8 Übungsaufgaben . . . . .	40
<b>3 Grundlagen der Programmierung in Java</b> . . . . .	<b>41</b>
3.1 Grundelemente eines Java-Programms . . . . .	41
3.1.1 Kommentare . . . . .	43
3.1.2 Bezeichner und Namen . . . . .	45
3.1.3 Literale . . . . .	46
3.1.4 Reservierte Wörter, Schlüsselwörter . . . . .	47
3.1.5 Trennzeichen . . . . .	47

3.1.6	Interpunktionszeichen . . . . .	48
3.1.7	Operatorsymbole . . . . .	49
3.1.8	<b>import</b> -Anweisungen . . . . .	49
3.1.9	Zusammenfassung . . . . .	50
3.1.10	Übungsaufgaben . . . . .	51
3.2	Erste Schritte in Java . . . . .	52
3.2.1	Grundstruktur eines Java-Programms . . . . .	52
3.2.2	Ausgaben auf der Konsole . . . . .	53
3.2.3	Eingaben von der Konsole . . . . .	55
3.2.4	Schöner programmieren in Java . . . . .	55
3.2.5	Zusammenfassung . . . . .	56
3.2.6	Übungsaufgaben . . . . .	57
3.3	Einfache Datentypen . . . . .	57
3.3.1	Ganzzahlige Datentypen . . . . .	57
3.3.1.1	Literalkonstanten in anderen Zahlensystemen . . . . .	59
3.3.1.2	Unterstrich als Trennzeichen in Literalkonstanten . . . . .	60
3.3.2	Gleitkommatypen . . . . .	61
3.3.3	Der Datentyp <b>char</b> für Zeichen . . . . .	63
3.3.4	Zeichenketten . . . . .	63
3.3.5	Der Datentyp <b>boolean</b> für Wahrheitswerte . . . . .	64
3.3.6	Implizite und explizite Typumwandlungen . . . . .	64
3.3.7	Zusammenfassung . . . . .	65
3.3.8	Übungsaufgaben . . . . .	66
3.4	Der Umgang mit einfachen Datentypen . . . . .	66
3.4.1	Variablen . . . . .	67
3.4.2	Operatoren und Ausdrücke . . . . .	70
3.4.2.1	Arithmetische Operatoren . . . . .	72
3.4.2.2	Bitoperatoren . . . . .	74
3.4.2.3	Zuweisungsoperator . . . . .	76
3.4.2.4	Vergleichsoperatoren und logische Operatoren . . . . .	76
3.4.2.5	Inkrement- und Dekrementoperatoren . . . . .	79
3.4.2.6	Priorität und Auswertungsreihenfolge der Operatoren . . . . .	79
3.4.3	Allgemeine Ausdrücke . . . . .	81
3.4.4	Ein- und Ausgabe . . . . .	81
3.4.4.1	Statischer Import der IOTools-Methoden . . . . .	83
3.4.5	Zusammenfassung . . . . .	84
3.4.6	Übungsaufgaben . . . . .	84
3.5	Anweisungen und Ablaufsteuerung . . . . .	88
3.5.1	Anweisungen . . . . .	88
3.5.2	Blöcke und ihre Struktur . . . . .	89
3.5.3	Entscheidungsanweisung . . . . .	89
3.5.3.1	Die <b>if</b> -Anweisung . . . . .	89
3.5.3.2	Die <b>switch</b> -Anweisung . . . . .	91

3.5.4	Wiederholungsanweisungen, Schleifen . . . . .	95
3.5.4.1	Die <b>for</b> -Anweisung . . . . .	95
3.5.4.2	Vereinfachte <b>for</b> -Schleifen-Notation . . . . .	96
3.5.4.3	Die <b>while</b> -Anweisung . . . . .	97
3.5.4.4	Die <b>do</b> -Anweisung . . . . .	97
3.5.4.5	Endlosschleifen . . . . .	98
3.5.5	Sprungbefehle und markierte Anweisungen . . . . .	99
3.5.6	Zusammenfassung . . . . .	101
3.5.7	Übungsaufgaben . . . . .	101
<b>4</b>	<b>Referenzdatentypen . . . . .</b>	<b>111</b>
4.1	Felder . . . . .	113
4.1.1	Was sind Felder? . . . . .	115
4.1.2	Deklaration, Erzeugung und Initialisierung von Feldern . . . . .	117
4.1.3	Felder unbekannter Länge . . . . .	119
4.1.4	Referenzen . . . . .	121
4.1.5	Ein besserer Terminkalender . . . . .	126
4.1.6	Mehrdimensionale Felder . . . . .	128
4.1.7	Mehrdimensionale Felder unterschiedlicher Länge . . . . .	132
4.1.8	Vorsicht, Falle: Kopieren von mehrdimensionalen Feldern . . . . .	134
4.1.9	Vereinfachte <b>for</b> -Schleifen-Notation . . . . .	135
4.1.10	Zusammenfassung . . . . .	136
4.1.11	Übungsaufgaben . . . . .	137
4.2	Klassen . . . . .	140
4.2.1	Willkommen in der ersten Klasse! . . . . .	141
4.2.2	Komponentenzugriff bei Objekten . . . . .	145
4.2.3	Ein erstes Adressbuch . . . . .	145
4.2.4	Klassen als Referenzdatentyp . . . . .	147
4.2.5	Felder von Klassen . . . . .	150
4.2.6	Vorsicht, Falle: Kopieren von geschachtelten Referenzdatentypen . . . . .	153
4.2.7	Zusammenfassung . . . . .	154
4.2.8	Übungsaufgaben . . . . .	154
<b>5</b>	<b>Methoden, Unterprogramme . . . . .</b>	<b>157</b>
5.1	Methoden . . . . .	158
5.1.1	Was sind Methoden? . . . . .	158
5.1.2	Deklaration von Methoden . . . . .	159
5.1.3	Parameterübergabe und Ergebnissrückgabe . . . . .	160
5.1.4	Aufruf von Methoden . . . . .	161
5.1.5	Überladen von Methoden . . . . .	163
5.1.6	Variable Argumentanzahl bei Methoden . . . . .	164
5.1.7	Vorsicht, Falle: Referenzen als Parameter . . . . .	165
5.1.8	Sichtbarkeit und Verdecken von Variablen . . . . .	168
5.1.9	Zusammenfassung . . . . .	169

5.1.10	Übungsaufgaben . . . . .	170
5.2	Rekursiv definierte Methoden . . . . .	171
5.2.1	Motivation . . . . .	171
5.2.2	Gute und schlechte Beispiele für rekursive Methoden . . . .	173
5.2.3	Zusammenfassung . . . . .	176
5.3	Die Methode <code>main</code> . . . . .	176
5.3.1	Kommandozeilenparameter . . . . .	176
5.3.2	Anwendung der vereinfachten <code>for</code> -Schleifen-Notation . . .	178
5.3.3	Zusammenfassung . . . . .	178
5.3.4	Übungsaufgaben . . . . .	179
5.4	Methoden aus anderen Klassen aufrufen . . . . .	180
5.4.1	Klassenmethoden . . . . .	181
5.4.2	Die Methoden der Klasse <code>java.lang.Math</code> . . . . .	182
5.4.3	Statischer Import . . . . .	183
5.5	Methoden von Objekten aufrufen . . . . .	184
5.5.1	Instanzmethoden . . . . .	184
5.5.2	Die Methoden der Klasse <code>java.lang.String</code> . . . . .	185
5.6	Übungsaufgaben . . . . .	187
 <b>II Objektorientiertes Programmieren in Java . . . . .</b>		<b>193</b>
<b>6</b>	<b>Die objektorientierte Philosophie . . . . .</b>	<b>195</b>
6.1	Die Welt, in der wir leben . . . . .	195
6.2	Programmierparadigmen – Objektorientierung im Vergleich . . . .	196
6.3	Die vier Grundpfeiler objektorientierter Programmierung . . . . .	198
6.3.1	Generalisierung . . . . .	198
6.3.2	Vererbung . . . . .	200
6.3.3	Kapselung . . . . .	203
6.3.4	Polymorphie . . . . .	204
6.3.5	Weitere wichtige Grundbegriffe . . . . .	205
6.4	Modellbildung – von der realen Welt in den Computer . . . . .	206
6.4.1	Grafisches Modellieren mit UML . . . . .	206
6.4.2	Entwurfsmuster . . . . .	207
6.5	Zusammenfassung . . . . .	208
6.6	Übungsaufgaben . . . . .	209
<b>7</b>	<b>Der grundlegende Umgang mit Klassen . . . . .</b>	<b>211</b>
7.1	Vom Referenzdatentyp zur Objektorientierung . . . . .	211
7.2	Instanzmethoden . . . . .	213
7.2.1	Zugriffsrechte . . . . .	213
7.2.2	Was sind Instanzmethoden? . . . . .	214
7.2.3	Instanzmethoden zur Validierung von Eingaben . . . . .	217
7.2.4	Instanzmethoden als erweiterte Funktionalität . . . . .	218
7.3	Statische Komponenten einer Klasse . . . . .	219

---

7.3.1	Klassenvariablen und -methoden . . . . .	220
7.3.2	Klassenkonstanten . . . . .	222
7.4	Instanziierung und Initialisierung . . . . .	225
7.4.1	Konstruktoren . . . . .	225
7.4.2	Überladen von Konstruktoren . . . . .	227
7.4.3	Der statische Initialisierer . . . . .	229
7.4.4	Der Mechanismus der Objekterzeugung . . . . .	231
7.5	Zusammenfassung . . . . .	236
7.6	Übungsaufgaben . . . . .	237
<b>8</b>	<b>Vererbung und Polymorphie . . . . .</b>	<b>257</b>
8.1	Wozu braucht man Vererbung? . . . . .	257
8.1.1	Aufgabenstellung . . . . .	257
8.1.2	Analyse des Problems . . . . .	258
8.1.3	Ein erster Ansatz . . . . .	258
8.1.4	Eine Klasse für sich . . . . .	259
8.1.5	Stärken der Vererbung . . . . .	260
8.1.6	Vererbung verhindern durch <b>final</b> . . . . .	263
8.1.7	Übungsaufgaben . . . . .	264
8.2	Die <b>super</b> -Referenz . . . . .	265
8.3	Überschreiben von Methoden und Variablen . . . . .	267
8.3.1	Dynamisches Binden . . . . .	267
8.3.2	Überschreiben von Methoden verhindern durch <b>final</b> . . . . .	269
8.4	Die Klasse <code>java.lang.Object</code> . . . . .	269
8.5	Übungsaufgaben . . . . .	273
8.6	Abstrakte Klassen und Interfaces . . . . .	273
8.6.1	Einsatzszenarien am Beispiel . . . . .	273
8.6.2	Abstrakte Klassen im Detail . . . . .	277
8.6.3	Interfaces im Detail . . . . .	280
8.7	Interfaces mit Default-Methoden und statischen Methoden . . . . .	283
8.7.1	Deklaration von Default-Methoden . . . . .	283
8.7.2	Deklaration von statischen Methoden . . . . .	284
8.7.3	Auflösung von Namensgleichheiten bei Default-Methoden . . . . .	285
8.7.4	Interfaces und abstrakte Klassen im Vergleich . . . . .	287
8.8	Weiteres zum Thema Objektorientierung . . . . .	287
8.8.1	Erstellen von Paketen . . . . .	287
8.8.2	Zugriffsrechte . . . . .	289
8.8.3	Innere Klassen . . . . .	290
8.8.4	Anonyme Klassen . . . . .	296
8.9	Zusammenfassung . . . . .	299
8.10	Übungsaufgaben . . . . .	299
<b>9</b>	<b>Exceptions und Errors . . . . .</b>	<b>311</b>
9.1	Eine Einführung in Exceptions . . . . .	312
9.1.1	Was ist eine Exception? . . . . .	312

9.1.2	Übungsaufgaben	314
9.1.3	Abfangen von Exceptions	314
9.1.4	Ein Anwendungsbeispiel	315
9.1.5	Die <code>RuntimeException</code>	318
9.1.6	Übungsaufgaben	319
9.2	Exceptions für Fortgeschrittene	321
9.2.1	Definieren eigener Exceptions	321
9.2.2	Übungsaufgaben	323
9.2.3	Vererbung und Exceptions	323
9.2.4	Vorsicht, Falle!	327
9.2.5	Der <b>finally</b> -Block	329
9.2.6	Die Klassen <code>Throwable</code> und <code>Error</code>	333
9.2.7	Zusammenfassung	335
9.2.8	Übungsaufgaben	335
9.3	Assertions	336
9.3.1	Zusicherungen im Programmcode	336
9.3.2	Ausführen des Programmcodes	337
9.3.3	Zusammenfassung	338
9.4	Mehrere Ausnahmetypen in einem <b>catch</b> -Block	338
9.5	Ausblick: <b>try</b> -Block mit Ressourcen	340
<b>10</b>	<b>Fortgeschrittene objektorientierte Programmierung</b>	<b>341</b>
10.1	Aufzählungstypen	341
10.1.1	Deklaration eines Aufzählungstyps	342
10.1.2	Instanzmethoden der <b>enum</b> -Objekte	342
10.1.3	Selbstdefinierte Instanzmethoden für <b>enum</b> -Objekte	343
10.1.4	Übungsaufgaben	344
10.2	Generische Datentypen	346
10.2.1	Herkömmliche Generizität	347
10.2.2	Generizität durch Typ-Parameter	349
10.2.3	Einschränkungen der Typ-Parameter	351
10.2.4	Wildcards	353
10.2.5	Bounded Wildcards	354
10.2.6	Generische Methoden	356
10.2.7	Verkürzte Notation bei generischen Datentypen	358
10.2.8	Ausblick	361
10.2.9	Übungsaufgaben	361
10.3	Sortieren von Feldern und das Interface <code>Comparable</code>	366
<b>11</b>	<b>Einige wichtige Hilfsklassen</b>	<b>369</b>
11.1	Die Klasse <code>StringBuffer</code>	369
11.1.1	Arbeiten mit <code>String</code> -Objekten	369
11.1.2	Arbeiten mit <code>StringBuffer</code> -Objekten	372
11.1.3	Übungsaufgaben	374
11.2	Die Wrapper-Klassen (Hüll-Klassen)	375

11.2.1	Arbeiten mit „eingepackten“ Daten	375
11.2.2	Aufbau der Wrapper-Klassen	377
11.2.3	Ein Anwendungsbeispiel	379
11.2.4	Automatische Typwandlung für die Wrapper-Klassen	380
11.2.5	Übungsaufgaben	382
11.3	Die Klassen <code>BigInteger</code> und <code>BigDecimal</code>	382
11.3.1	Arbeiten mit langen Ganzzahlen	383
11.3.2	Aufbau der Klasse <code>BigInteger</code>	384
11.3.3	Übungsaufgaben	386
11.3.4	Arbeiten mit langen Gleitkommazahlen	387
11.3.5	Aufbau der Klasse <code>BigDecimal</code>	390
11.3.6	Viele Stellen von Nullstellen gefällig?	392
11.3.7	Übungsaufgaben	394
11.4	Die Klasse <code>DecimalFormat</code>	394
11.4.1	Standardausgaben in Java	394
11.4.2	Arbeiten mit <code>Format</code> -Objekten	395
11.4.3	Vereinfachte formatierte Ausgabe	398
11.4.4	Übungsaufgaben	398
11.5	Die Klassen <code>Date</code> und <code>Calendar</code>	399
11.5.1	Arbeiten mit „Zeitpunkten“	399
11.5.2	Auf die Plätze, fertig, los!	400
11.5.3	Spezielle <code>Calendar</code> -Klassen	401
11.5.4	Noch einmal: Zeitmessung	404
11.5.5	Übungsaufgaben	405
11.6	Die Klassen <code>SimpleDateFormat</code> und <code>DateFormat</code>	405
11.6.1	Arbeiten mit <code>Format</code> -Objekten für Datum-/Zeit-Angaben	405
11.6.2	Übungsaufgaben	410
11.7	Die <code>Collection</code> -Klassen	410
11.7.1	„Sammlungen“ von Objekten – der Aufbau des Interface <code>Collection</code>	411
11.7.2	„Sammlungen“ durchgehen – der Aufbau des Interface <code>Iterator</code>	413
11.7.3	Mengen	414
11.7.3.1	Das Interface <code>Set</code>	414
11.7.3.2	Die Klasse <code>HashSet</code>	415
11.7.3.3	Das Interface <code>SortedSet</code>	416
11.7.3.4	Die Klasse <code>TreeSet</code>	417
11.7.4	Listen	419
11.7.4.1	Das Interface <code>List</code>	419
11.7.4.2	Die Klassen <code>ArrayList</code> und <code>LinkedList</code>	420
11.7.4.3	Suchen und Sortieren – die Klassen <code>Collections</code> und <code>Arrays</code>	422
11.7.5	Verkürzte Notation bei <code>Collection</code> -Datentypen	424
11.7.6	Übungsaufgaben	426

11.8	Die Klasse <code>StringTokenizer</code> . . . . .	426
11.8.1	Übungsaufgaben . . . . .	429
<b>III</b>	<b>Grafische Oberflächen in Java . . . . .</b>	<b>431</b>
<b>12</b>	<b>Aufbau grafischer Oberflächen in Frames – von AWT nach Swing . . .</b>	<b>433</b>
12.1	Grundsätzliches zum Aufbau grafischer Oberflächen . . . . .	433
12.2	Ein einfaches Beispiel mit dem AWT . . . . .	434
12.3	Let's swing now! . . . . .	437
12.4	Etwas „Fill-in“ gefällig? . . . . .	439
12.5	Die AWT- und Swing-Klassenbibliothek im Überblick . . . . .	441
12.6	Übungsaufgaben . . . . .	443
<b>13</b>	<b>Swing-Komponenten . . . . .</b>	<b>445</b>
13.1	Die abstrakte Klasse <code>Component</code> . . . . .	445
13.2	Die Klasse <code>Container</code> . . . . .	446
13.3	Die abstrakte Klasse <code>JComponent</code> . . . . .	447
13.4	Layout-Manager, Farben und Schriften . . . . .	448
13.4.1	Die Klasse <code>Color</code> . . . . .	449
13.4.2	Die Klasse <code>Font</code> . . . . .	451
13.4.3	Layout-Manager . . . . .	452
13.4.3.1	Die Klasse <code>FlowLayout</code> . . . . .	453
13.4.3.2	Die Klasse <code>BorderLayout</code> . . . . .	455
13.4.3.3	Die Klasse <code>GridLayout</code> . . . . .	456
13.5	Einige Grundkomponenten . . . . .	458
13.5.1	Die Klasse <code>JLabel</code> . . . . .	459
13.5.2	Die abstrakte Klasse <code>AbstractButton</code> . . . . .	461
13.5.3	Die Klasse <code>JButton</code> . . . . .	461
13.5.4	Die Klasse <code>JToggleButton</code> . . . . .	463
13.5.5	Die Klasse <code>JCheckBox</code> . . . . .	464
13.5.6	Die Klassen <code>JRadioButton</code> und <code>ButtonGroup</code> . . . . .	465
13.5.7	Die Klasse <code>JComboBox</code> . . . . .	467
13.5.8	Die Klasse <code>JList</code> . . . . .	470
13.5.9	Die abstrakte Klasse <code>JTextComponent</code> . . . . .	473
13.5.10	Die Klassen <code>JTextField</code> und <code>JPasswordField</code> . . . . .	473
13.5.11	Die Klasse <code>JTextArea</code> . . . . .	476
13.5.12	Die Klasse <code>JScrollPane</code> . . . . .	478
13.5.13	Die Klasse <code>JPanel</code> . . . . .	480
13.6	Spezielle Container, Menüs und Toolbars . . . . .	481
13.6.1	Die Klasse <code>JFrame</code> . . . . .	482
13.6.2	Die Klasse <code>JWindow</code> . . . . .	483
13.6.3	Die Klasse <code>JDialog</code> . . . . .	483
13.6.4	Die Klasse <code>JMenuBar</code> . . . . .	486
13.6.5	Die Klasse <code>JToolBar</code> . . . . .	489



13.7	Übungsaufgaben	492
<b>14</b>	<b>Ereignisverarbeitung</b>	<b>495</b>
14.1	Zwei einfache Beispiele	496
14.1.1	Zufällige Grautöne als Hintergrund	496
14.1.2	Ein interaktiver Bilderrahmen	499
14.2	Programmiervarianten für die Ereignisverarbeitung	503
14.2.1	Innere Klasse als Listener-Klasse	503
14.2.2	Anonyme Klasse als Listener-Klasse	503
14.2.3	Container-Klasse als Listener-Klasse	504
14.2.4	Separate Klasse als Listener-Klasse	505
14.3	Event-Klassen und -Quellen	507
14.4	Listener-Interfaces und Adapter-Klassen	510
14.5	Listener-Registrierung bei den Event-Quellen	516
14.6	Auf die Plätze, fertig, los!	519
14.7	Übungsaufgaben	524
<b>15</b>	<b>Einige Ergänzungen zu Swing-Komponenten</b>	<b>529</b>
15.1	Zeichnen in Swing-Komponenten	529
15.1.1	Grafische Darstellung von Komponenten	529
15.1.2	Das Grafikkoordinatensystem	530
15.1.3	Die abstrakte Klasse Graphics	531
15.1.4	Ein einfaches Zeichenprogramm	534
15.1.5	Layoutveränderungen und der Einsatz von revalidate	536
15.2	Noch mehr Swing gefällig?	538
15.3	Übungsaufgaben	540
<b>IV</b>	<b>Threads, Datenströme und Netzwerkanwendungen</b>	<b>543</b>
<b>16</b>	<b>Parallele Programmierung mit Threads</b>	<b>545</b>
16.1	Ein einfaches Beispiel	545
16.2	Threads in Java	547
16.2.1	Die Klasse Thread	548
16.2.2	Das Interface Runnable	552
16.2.3	Threads vorzeitig beenden	554
16.3	Wissenswertes über Threads	556
16.3.1	Lebenszyklus eines Threads	556
16.3.2	Thread-Scheduling	558
16.3.3	Dämon-Threads und Thread-Gruppen	558
16.4	Thread-Synchronisation und -Kommunikation	559
16.4.1	Das Leser/Schreiber-Problem	560
16.4.2	Das Erzeuger/Verbraucher-Problem	563
16.5	Threads in Swing-Anwendungen	571
16.5.1	Auf die Plätze, fertig, los!	571

16.5.2	Spielereien . . . . .	574
16.5.3	Swing-Komponenten sind nicht Thread-sicher . . . . .	577
16.6	Übungsaufgaben . . . . .	578
<b>17</b>	<b>Ein- und Ausgabe über I/O-Streams . . . . .</b>	<b>581</b>
17.1	Grundsätzliches zu I/O-Streams in Java . . . . .	582
17.2	Dateien und Verzeichnisse – die Klasse File . . . . .	582
17.3	Ein- und Ausgabe über Character-Streams . . . . .	585
17.3.1	Einfache Reader- und Writer-Klassen . . . . .	586
17.3.2	Gepufferte Reader- und Writer-Klassen . . . . .	589
17.3.3	Die Klasse StreamTokenizer . . . . .	591
17.3.4	Die Klasse PrintWriter . . . . .	592
17.3.5	Die Klassen IOTools und Scanner . . . . .	594
17.3.5.1	Was machen eigentlich die IOTools? . . . . .	594
17.3.5.2	Konsoleneingabe über ein Scanner-Objekt . . . . .	595
17.4	Ein- und Ausgabe über Byte-Streams . . . . .	596
17.4.1	Einige InputStream- und OutputStream-Klassen . . . . .	597
17.4.2	Die Serialisierung und Deserialisierung von Objekten . . . . .	599
17.4.3	Die Klasse PrintStream . . . . .	601
17.5	Streams im <code>try</code> -Block mit Ressourcen . . . . .	602
17.6	Einige abschließende Bemerkungen . . . . .	604
17.6.1	Das Paket <code>java.nio</code> . . . . .	605
17.6.2	Das Paket <code>java.nio.file</code> . . . . .	606
17.6.2.1	Das Interface <code>Path</code> und die Klasse <code>Paths</code> . . . . .	606
17.6.2.2	Die Klasse <code>Files</code> . . . . .	607
17.7	Übungsaufgaben . . . . .	610
<b>18</b>	<b>Client/Server-Programmierung in Netzwerken . . . . .</b>	<b>613</b>
18.1	Wissenswertes über Netzwerkkommunikation . . . . .	614
18.1.1	Protokolle . . . . .	614
18.1.2	IP-Adressen . . . . .	616
18.1.3	Ports und Sockets . . . . .	617
18.2	Client/Server-Programmierung . . . . .	618
18.2.1	Die Klassen <code>ServerSocket</code> und <code>Socket</code> . . . . .	619
18.2.2	Ein einfacher Server . . . . .	621
18.2.3	Ein einfacher Client . . . . .	624
18.2.4	Ein Server für mehrere Clients . . . . .	625
18.2.5	Ein Mehrzweck-Client . . . . .	628
18.2.6	Client/Server-Kommunikation über URLs . . . . .	631
18.3	Übungsaufgaben . . . . .	632
<b>19</b>	<b>Lambda-Ausdrücke, Streams und Pipeline-Operationen . . . . .</b>	<b>637</b>
19.1	Lambda-Ausdrücke . . . . .	637
19.1.1	Lambda-Ausdrücke in Aktion – zwei Beispiele . . . . .	638
19.1.2	Lambda-Ausdrücke im Detail . . . . .	641

19.1.3	Lambda-Ausdrücke und funktionale Interfaces . . . . .	643
19.1.4	Vordefinierte funktionale Interfaces und Anwendungen auf Datenstrukturen . . . . .	645
19.1.5	Methodenreferenzen als Lambda-Ausdrücke . . . . .	649
19.1.6	Zugriff auf Variablen aus der Umgebung innerhalb eines Lambda-Ausdrucks . . . . .	652
19.1.7	Übungsaufgaben . . . . .	653
19.2	Streams und Pipeline-Operationen . . . . .	654
19.2.1	Streams in Aktion . . . . .	655
19.2.2	Streams und Pipelines im Detail . . . . .	657
19.2.3	Erzeugen von endlichen und unendlichen Streams . . . . .	658
19.2.4	Die Stream-API . . . . .	661
19.2.5	Übungsaufgaben . . . . .	664
<b>V</b>	<b>Abschluss, Ausblick und Anhang . . . . .</b>	<b>667</b>
<b>20</b>	<b>Blick über den Tellerrand . . . . .</b>	<b>669</b>
20.1	JShell für kleine Skripte . . . . .	670
20.2	Das Java-Modulsystem . . . . .	674
20.3	Bühne frei für JavaFX . . . . .	681
20.4	Beginn einer neuen Zeitrechnung . . . . .	690
20.5	Webprogrammierung und verteilte Systeme . . . . .	692
20.6	Zu guter Letzt . . . . .	695
<b>A</b>	<b>Der Weg zum guten Programmierer ... . . . .</b>	<b>697</b>
A.1	Die goldenen Regeln der Code-Formatierung . . . . .	698
A.2	Die goldenen Regeln der Namensgebung . . . . .	701
A.3	Zusammenfassung . . . . .	703
<b>B</b>	<b>Ohne Werkzeug geht es nicht . . . . .</b>	<b>705</b>
B.1	Die API-Dokumentation zum Nachschlagen . . . . .	706
B.2	Die IDE, dein Freund und Helfer . . . . .	708
B.3	Alle Versionen stets im Griff . . . . .	710
B.4	Testen bitte nicht vergessen . . . . .	712
B.5	Der Automat kann es besser . . . . .	714
<b>C</b>	<b>Die Klasse <code>IOTools</code> – Tastatureingaben in Java . . . . .</b>	<b>717</b>
C.1	Kurzbeschreibung . . . . .	717
C.2	Anwendung der <code>IOTools</code> -Methoden . . . . .	718
	<b>Glossar . . . . .</b>	<b>721</b>
	<b>Stichwortverzeichnis . . . . .</b>	<b>741</b>