

Inhaltsverzeichnis

Vorwort	17
Einleitung	19
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
Über die Autoren	22
 I Einstieg in das Programmieren in Java	 23
1 Einige Grundbegriffe aus der Welt des Programmierens	25
1.1 Computer, Software, Informatik und das Internet	25
1.2 Was heißt Programmieren?	27
1.3 Welche Werkzeuge brauchen wir?	30
 2 Aller Anfang ist schwer	 33
2.1 Installation der Entwicklungswerkzeuge	33
2.1.1 Rundum sorglos mit Eclipse	33
2.1.2 Traditionelle JDK-Installation	36
2.2 Mein erstes Programm	37
2.2.1 Quellcode eingeben, übersetzen und ausführen	37
2.2.2 Das Programmgerüst	40
2.2.3 Verwendung von Variablen	41
2.2.4 Formeln, Ausdrücke, Zuweisungen	41
2.2.5 „Auf den Schirm!“	42
2.2.6 Die Kurzversion zum Vergleich	43
2.3 Übungsaufgaben	44
 3 Grundlagen der Programmierung in Java	 45
3.1 Grundelemente eines Java-Programms	45
3.1.1 Kommentare	47

3.1.2	Bezeichner und Namen	49
3.1.3	Literale	50
3.1.4	Reservierte Wörter, Schlüsselwörter	50
3.1.5	Trennzeichen	51
3.1.6	Interpunktionszeichen	52
3.1.7	Operatorsymbole	53
3.1.8	import -Anweisungen	53
3.1.9	Zusammenfassung	54
3.1.10	Übungsaufgaben	54
3.2	Erste Schritte in Java	55
3.2.1	Grundstruktur eines Java-Programms	56
3.2.2	Ausgaben auf der Konsole	57
3.2.3	Eingaben von der Konsole	58
3.2.4	Schöner programmieren in Java	58
3.2.5	Zusammenfassung	59
3.2.6	Übungsaufgaben	60
3.3	Einfache Datentypen	60
3.3.1	Ganzzahlige Datentypen	61
3.3.1.1	Literalkonstanten in anderen Zahlensystemen	62
3.3.1.2	Unterstrich als Trennzeichen in Literalkonstanten	63
3.3.2	Gleitkommatypen	64
3.3.3	Der Datentyp char für Zeichen	66
3.3.4	Der Datentyp String für Zeichenketten	67
3.3.5	Der Datentyp boolean für Wahrheitswerte	67
3.3.6	Implizite und explizite Typumwandlungen	68
3.3.7	Zusammenfassung	69
3.3.8	Übungsaufgaben	69
3.4	Der Umgang mit einfachen Datentypen	70
3.4.1	Variablen	70
3.4.2	Operatoren und Ausdrücke	74
3.4.2.1	Arithmetische Operatoren	76
3.4.2.2	Bitoperatoren	78
3.4.2.3	Zuweisungsoperator	80
3.4.2.4	Vergleichsoperatoren und logische Operatoren	81
3.4.2.5	Inkrement- und Dekrementoperatoren	83
3.4.2.6	Priorität und Auswertungsreihenfolge der Operatoren	84
3.4.3	Allgemeine Ausdrücke	85
3.4.3.1	Reihenfolge der Operationen in Ausdrücken	85
3.4.3.2	Potenzielle Probleme bei der Auswertung von Ausdrücken	86
3.4.4	Ein- und Ausgabe	87
3.4.4.1	Statischer Import der IOTools-Methoden	89
3.4.5	Zusammenfassung	90

3.4.6	Übungsaufgaben	90
3.5	Anweisungen und Ablaufsteuerung	94
3.5.1	Anweisungen	94
3.5.2	Blöcke und ihre Struktur	94
3.5.3	Entscheidungsanweisungen	95
3.5.3.1	Die if -Anweisung	95
3.5.3.2	Die switch -Anweisung	98
3.5.3.3	Die vereinfachte switch -Anweisung	102
3.5.3.4	Switch-Ausdrücke	104
3.5.4	Wiederholungsanweisungen, Schleifen	105
3.5.4.1	Die for -Anweisung	106
3.5.4.2	Vereinfachte for -Schleifen-Notation	107
3.5.4.3	Die while -Anweisung	107
3.5.4.4	Die do -Anweisung	108
3.5.4.5	Endlosschleifen	109
3.5.5	Sprungbefehle und markierte Anweisungen	110
3.5.6	Zusammenfassung	112
3.5.7	Übungsaufgaben	112
4	Referenzdatentypen	121
4.1	Felder (Arrays)	123
4.1.1	Was sind Felder?	125
4.1.2	Deklaration, Erzeugung und Initialisierung von Feldern	127
4.1.3	Felder unbekannter Länge	129
4.1.4	Referenzen	131
4.1.5	Ein besserer Terminkalender	136
4.1.6	Mehrdimensionale Felder	138
4.1.7	Mehrdimensionale Felder unterschiedlicher Länge	142
4.1.8	Vorsicht, Falle: Kopieren von mehrdimensionalen Feldern	144
4.1.9	Vereinfachte for -Schleifen-Notation	145
4.1.10	Zusammenfassung	146
4.1.11	Übungsaufgaben	147
4.2	Klassen	150
4.2.1	Willkommen in der ersten Klasse!	151
4.2.2	Komponentenzugriff bei Objekten	155
4.2.3	Ein erstes Adressbuch	155
4.2.4	Klassen als Referenzdatentyp	157
4.2.5	Felder von Klassen	160
4.2.6	Vorsicht, Falle: Kopieren von geschachtelten Referenzdaten- typen	163
4.2.7	Zusammenfassung	164
4.2.8	Übungsaufgaben	164
5	Methoden, Unterprogramme	167
5.1	Methoden	168

5.1.1	Was sind Methoden?	168
5.1.2	Deklaration von Methoden	169
5.1.3	Parameterübergabe und Ergebnisrückgabe	170
5.1.4	Aufruf von Methoden	172
5.1.5	Überladen von Methoden	174
5.1.6	Variable Argumentanzahl bei Methoden	175
5.1.7	Vorsicht, Falle: Referenzen als Parameter	176
5.1.8	Sichtbarkeit und Verdecken von Variablen	178
5.1.9	Zusammenfassung	180
5.1.10	Übungsaufgaben	180
5.2	Rekursiv definierte Methoden	181
5.2.1	Motivation	181
5.2.2	Gute und schlechte Beispiele für rekursive Methoden	183
5.2.3	Zusammenfassung	186
5.3	Die Methode <code>main</code>	186
5.3.1	Kommandozeilenparameter	187
5.3.2	Anwendung der vereinfachten for -Schleifen-Notation	188
5.3.3	Zusammenfassung	189
5.3.4	Übungsaufgaben	189
5.4	Methoden aus anderen Klassen aufrufen	191
5.4.1	Klassenmethoden	191
5.4.2	Die Methoden der Klasse <code>Math</code>	192
5.4.3	Statischer Import	194
5.5	Methoden von Objekten aufrufen	195
5.5.1	Instanzmethoden	195
5.5.2	Die Methoden der Klasse <code>String</code>	196
5.6	Übungsaufgaben	198

II Objektorientiertes Programmieren in Java 203

6	Die objektorientierte Philosophie	205
6.1	Die Welt, in der wir leben	205
6.2	Programmierparadigmen – Objektorientierung im Vergleich	206
6.3	Die vier Grundpfeiler objektorientierter Programmierung	208
6.3.1	Generalisierung	208
6.3.2	Vererbung	210
6.3.3	Kapselung	213
6.3.4	Polymorphie	214
6.3.5	Weitere wichtige Grundbegriffe	215
6.4	Modellbildung – von der realen Welt in den Computer	216
6.4.1	Grafisches Modellieren mit UML	216
6.4.2	Entwurfsmuster	217
6.5	Zusammenfassung	218
6.6	Übungsaufgaben	219

7	Der grundlegende Umgang mit Klassen	221
7.1	Vom Referenzdatentyp zur Objektorientierung	221
7.2	Instanzmethoden	223
7.2.1	Zugriffsrechte	223
7.2.2	Was sind Instanzmethoden?	224
7.2.3	Instanzmethoden zur Validierung von Eingaben	227
7.2.4	Instanzmethoden als erweiterte Funktionalität	228
7.3	Statische Komponenten einer Klasse	229
7.3.1	Klassenvariablen und -methoden	230
7.3.2	Klassenkonstanten	232
7.4	Instanziierung und Initialisierung	235
7.4.1	Konstruktoren	235
7.4.2	Überladen von Konstruktoren	237
7.4.3	Der statische Initialisierer	239
7.4.4	Der Mechanismus der Objekterzeugung	241
7.5	Zusammenfassung	246
7.6	Übungsaufgaben	247
8	Vererbung und Polymorphie	267
8.1	Wozu braucht man Vererbung?	267
8.1.1	Aufgabenstellung	267
8.1.2	Analyse des Problems	268
8.1.3	Ein erster Ansatz	268
8.1.4	Eine Klasse für sich	269
8.1.5	Stärken der Vererbung	270
8.1.6	Vererbung verhindern durch final	273
8.1.7	Übungsaufgaben	274
8.2	Die super -Referenz	275
8.3	Überschreiben von Methoden und Variablen	277
8.3.1	Dynamisches Binden	277
8.3.2	Überschreiben von Methoden verhindern durch final	279
8.4	Die Klasse Object und der Umgang mit instanceof und @Override	279
8.4.1	Methoden der Klasse Object sinnvoll überschreiben	280
8.4.2	Hilfe beim Überschreiben: Die Annotation @Override	283
8.4.3	Der Operator instanceof und das Pattern-Matching	286
8.4.4	Das Pattern-Matching für switch	288
8.5	Übungsaufgaben	290
8.6	Abstrakte Klassen und Interfaces	291
8.6.1	Einsatzszenarien am Beispiel	291
8.6.2	Abstrakte Klassen im Detail	294
8.6.3	Interfaces im Detail	297
8.7	Interfaces mit Default-Methoden und statischen Methoden	301
8.7.1	Deklaration von Default-Methoden	301

8.7.2	Deklaration von statischen Methoden	302
8.7.3	Auflösung von Namensgleichheiten bei Default-Methoden .	302
8.7.4	Interfaces und abstrakte Klassen im Vergleich	304
8.8	Weiteres zum Thema Objektorientierung	305
8.8.1	Erstellen von Paketen	305
8.8.2	Zugriffsrechte	308
8.8.3	Innere Klassen	309
8.8.4	Anonyme Klassen	315
8.9	Zusammenfassung	318
8.10	Übungsaufgaben	318
9	Exceptions und Errors	331
9.1	Eine Einführung in Exceptions	332
9.1.1	Was ist eine Exception?	332
9.1.2	Übungsaufgaben	334
9.1.3	Abfangen von Exceptions	334
9.1.4	Ein Anwendungsbeispiel	335
9.1.5	Die RuntimeException	338
9.1.6	Übungsaufgaben	339
9.2	Exceptions für Fortgeschrittene	341
9.2.1	Definieren eigener Exceptions	341
9.2.2	Übungsaufgaben	343
9.2.3	Vererbung und Exceptions	343
9.2.4	Vorsicht, Falle!	347
9.2.5	Der finally -Block	349
9.2.6	Die Klassen Throwable und Error	353
9.2.7	Zusammenfassung	355
9.2.8	Übungsaufgaben	355
9.3	Assertions	356
9.3.1	Zusicherungen im Programmcode	356
9.3.2	Ausführen des Programmcodes	357
9.3.3	Zusammenfassung	358
9.4	Mehrere Ausnahmetypen in einem catch -Block	358
9.5	Ausblick: try -Block mit Ressourcen	360
10	Fortgeschrittene Themen der objektorientierten Programmierung . . .	361
10.1	Aufzählungstypen	361
10.1.1	Deklaration eines Aufzählungstyps	362
10.1.2	Instanzmethoden der enum -Objekte	362
10.1.3	Selbstdefinierte Instanzmethoden für enum -Objekte	363
10.1.4	Übungsaufgaben	364
10.2	Generische Datentypen	367
10.2.1	Herkömmliche Generizität	367
10.2.2	Generizität durch Typ-Parameter	369
10.2.3	Einschränkungen der Typ-Parameter	371

10.2.4	Wildcards	373
10.2.5	Bounded Wildcards	375
10.2.6	Generische Methoden	377
10.2.7	Verkürzte Notation bei generischen Datentypen	379
10.2.8	Ausblick	382
10.2.9	Übungsaufgaben	382
10.3	Sortieren von Feldern und das Interface Comparable	387
10.3.1	Einsatz der Klasse Arrays	387
10.3.2	Implementierung des Interface Comparable	388
10.3.3	Übungsaufgaben	390
10.4	Versiegelte Klassen und Interfaces	390
10.4.1	Der Mechanismus der Versiegelung	391
10.4.2	Versiegelung am Beispiel von Klassen	392
10.4.3	Versiegelung am Beispiel mit einem Interface	395
10.4.4	Hilfreiche Konsequenzen für das Pattern-Matching	398
10.4.5	Übungsaufgaben	399
10.5	Records	400
10.5.1	Motivation für die Nutzung von Records	400
10.5.2	Records im Detail	401
10.5.2.1	Regeln zur Vererbung für Records	403
10.5.2.2	Mögliche Ergänzungen von Records	403
10.5.3	Records und das Pattern-Matching	405
10.5.4	Übungsaufgaben	406
11	Einige wichtige Hilfsklassen	407
11.1	Die Klasse StringBuffer	407
11.1.1	Arbeiten mit String-Objekten	407
11.1.2	Arbeiten mit StringBuffer-Objekten	410
11.1.3	Übungsaufgaben	412
11.2	Die Wrapper-Klassen (Hüll-Klassen)	413
11.2.1	Arbeiten mit „eingepackten“ Daten	413
11.2.2	Aufbau der Wrapper-Klassen	415
11.2.3	Ein Anwendungsbeispiel	417
11.2.4	Automatische Typwandlung für die Wrapper-Klassen	418
11.2.5	Übungsaufgaben	420
11.3	Die Klassen BigInteger und BigDecimal	420
11.3.1	Arbeiten mit langen Ganzzahlen	421
11.3.2	Aufbau der Klasse BigInteger	422
11.3.3	Übungsaufgaben	424
11.3.4	Arbeiten mit langen Gleitkommazahlen	425
11.3.5	Aufbau der Klasse BigDecimal	428
11.3.6	Viele Stellen von Nullstellen gefälltig?	431
11.3.7	Übungsaufgaben	432
11.4	Die Klasse DecimalFormat	433

11.4.1	Standardausgaben in Java	433
11.4.2	Arbeiten mit Format-Objekten	434
11.4.3	Vereinfachte formatierte Ausgabe	436
11.4.4	Übungsaufgaben	437
11.5	Die Klassen Date und Calendar	437
11.5.1	Arbeiten mit „Zeitpunkten“	438
11.5.2	Auf die Plätze, fertig, los!	439
11.5.3	Spezielle Calendar-Klassen	440
11.5.4	Noch einmal: Zeitmessung	442
11.5.5	Übungsaufgaben	444
11.6	Die Klassen SimpleDateFormat und DateFormat	444
11.6.1	Arbeiten mit Format-Objekten für Datum/Zeit-Angaben . .	444
11.6.2	Übungsaufgaben	449
11.7	Die Collection-Klassen	449
11.7.1	„Sammlungen“ von Objekten – der Aufbau des Interface Collection	449
11.7.2	„Sammlungen“ durchgehen – der Aufbau des Interface Iterator	452
11.7.3	Mengen	453
11.7.3.1	Das Interface Set	453
11.7.3.2	Die Klasse HashSet	453
11.7.3.3	Das Interface SortedSet	455
11.7.3.4	Die Klasse TreeSet	456
11.7.4	Listen	457
11.7.4.1	Das Interface List	458
11.7.4.2	Die Klassen ArrayList und LinkedList	458
11.7.4.3	Suchen und Sortieren – die Klassen Collections und Arrays	460
11.7.5	Verkürzte Notation bei Collection-Datentypen	463
11.7.6	Übungsaufgaben	464
11.7.7	Assoziative Sammlungen mit Maps	465
11.7.7.1	Das Interface Map	465
11.7.7.2	Die Klassen HashMap und TreeMap	466
11.7.8	Übungsaufgaben	469
11.8	Die Klasse StringTokenizer	469
11.8.1	Übungsaufgaben	473

III Grafische Oberflächen in Java 475

12	Aufbau grafischer Oberflächen in Frames – von AWT nach Swing . . .	477
12.1	Grundsätzliches zum Aufbau grafischer Oberflächen	477
12.2	Ein einfaches Beispiel mit dem AWT	478
12.3	Let's swing now!	481
12.4	Etwas „Fill-in“ gefällig?	483

12.5 Die AWT- und Swing-Klassenbibliothek im Überblick	485
12.6 Übungsaufgaben	487
13 Swing-Komponenten	489
13.1 Die abstrakte Klasse Component	489
13.2 Die Klasse Container	490
13.3 Die abstrakte Klasse JComponent	491
13.4 Layout-Manager, Farben und Schriften	492
13.4.1 Die Klasse Color	493
13.4.2 Die Klasse Font	495
13.4.3 Layout-Manager	496
13.4.3.1 Die Klasse FlowLayout	497
13.4.3.2 Die Klasse BorderLayout	499
13.4.3.3 Die Klasse GridLayout	500
13.5 Einige Grundkomponenten	502
13.5.1 Die Klasse JLabel	503
13.5.2 Die abstrakte Klasse AbstractButton	505
13.5.3 Die Klasse JButton	505
13.5.4 Die Klasse JToggleButton	507
13.5.5 Die Klasse JCheckBox	508
13.5.6 Die Klassen JRadioButton und ButtonGroup	509
13.5.7 Die Klasse JComboBox	511
13.5.8 Die Klasse JList	514
13.5.9 Die abstrakte Klasse JTextComponent	517
13.5.10 Die Klassen JTextField und JPasswordField	517
13.5.11 Die Klasse JTextArea	520
13.5.12 Die Klasse JScrollPane	522
13.5.13 Die Klasse JPanel	524
13.6 Spezielle Container, Menüs und Toolbars	525
13.6.1 Die Klasse JFrame	526
13.6.2 Die Klasse JWindow	527
13.6.3 Die Klasse JDialog	527
13.6.4 Die Klasse JMenuBar	530
13.6.5 Die Klasse JToolBar	533
13.7 Übungsaufgaben	536
14 Ereignisverarbeitung	539
14.1 Zwei einfache Beispiele	540
14.1.1 Zufällige Grautöne als Hintergrund	540
14.1.2 Ein interaktiver Bilderrahmen	543
14.2 Programmiervarianten für die Ereignisverarbeitung	547
14.2.1 Innere Klasse als Listener-Klasse	547
14.2.2 Anonyme Klasse als Listener-Klasse	547
14.2.3 Container-Klasse als Listener-Klasse	548
14.2.4 Separate Klasse als Listener-Klasse	549

14.3	Event-Klassen und -Quellen	551
14.4	Listener-Interfaces und Adapter-Klassen	554
14.5	Listener-Registrierung bei den Event-Quellen	560
14.6	Auf die Plätze, fertig, los!	563
14.7	Übungsaufgaben	568
15	Einige Ergänzungen zu Swing-Komponenten	573
15.1	Zeichnen in Swing-Komponenten	573
15.1.1	Grafische Darstellung von Komponenten	573
15.1.2	Das Grafikkoordinatensystem	574
15.1.3	Die abstrakte Klasse Graphics	575
15.1.4	Ein einfaches Zeichenprogramm	578
15.1.5	Layoutveränderungen und der Einsatz von revalidate	580
15.2	Noch mehr Swing gefällig?	582
15.3	Übungsaufgaben	583
IV	Nebenläufige und verteilte Anwendungen	587
16	Parallele Programmierung mit Threads	589
16.1	Ein einfaches Beispiel	589
16.2	Threads in Java	591
16.2.1	Die Klasse Thread	592
16.2.2	Das Interface Runnable	596
16.2.3	Threads vorzeitig beenden	598
16.3	Wissenswertes über Threads	600
16.3.1	Lebenszyklus eines Threads	600
16.3.2	Thread-Scheduling	602
16.3.3	Dämon-Threads und Thread-Gruppen	602
16.4	Thread-Synchronisation und -Kommunikation	603
16.4.1	Das Leser/Schreiber-Problem	604
16.4.2	Das Erzeuger/Verbraucher-Problem	607
16.5	Threads in Swing-Anwendungen	615
16.5.1	Auf die Plätze, fertig, los!	615
16.5.2	Spielereien	618
16.5.3	Swing-Komponenten sind nicht Thread-sicher	621
16.6	Übungsaufgaben	622
17	Ein- und Ausgabe über I/O-Streams	625
17.1	Grundsätzliches zu I/O-Streams in Java	626
17.2	Dateien und Verzeichnisse – die Klasse File	626
17.3	Ein- und Ausgabe über Character-Streams	629
17.3.1	Einfache Reader- und Writer-Klassen	630
17.3.2	Gepufferte Reader- und Writer-Klassen	633
17.3.3	Die Klasse StreamTokenizer	635

17.3.4	Die Klasse <code>PrintWriter</code>	636
17.3.5	Die Klassen <code>IOTools</code> und <code>Scanner</code>	638
17.3.5.1	Was machen eigentlich die <code>IOTools</code> ?	638
17.3.5.2	Konsoleneingabe über ein <code>Scanner</code> -Objekt	639
17.4	Ein- und Ausgabe über Byte-Streams	640
17.4.1	Einige <code>InputStream</code> - und <code>OutputStream</code> -Klassen	641
17.4.2	Die Serialisierung und Deserialisierung von Objekten	643
17.4.3	Die Klasse <code>PrintStream</code>	647
17.5	Streams im <code>try</code> -Block mit Ressourcen	648
17.6	Einige abschließende Bemerkungen	650
17.6.1	Das Paket <code>java.nio</code>	651
17.6.2	Das Paket <code>java.nio.file</code>	651
17.6.2.1	Das Interface <code>Path</code> und die Klasse <code>Paths</code>	652
17.6.2.2	Die Klasse <code>Files</code>	653
17.7	Übungsaufgaben	656
18	Client/Server-Programmierung in Netzwerken	659
18.1	Wissenswertes über Netzwerkkommunikation	660
18.1.1	Protokolle	660
18.1.2	IP-Adressen	662
18.1.3	Ports und Sockets	663
18.2	Client/Server-Programmierung	664
18.2.1	Die Klassen <code>ServerSocket</code> und <code>Socket</code>	665
18.2.2	Ein einfacher Server	667
18.2.3	Ein einfacher Client	670
18.2.4	Ein Server für mehrere Clients	671
18.2.5	Ein Mehrzweck-Client	674
18.3	Übungsaufgaben	677
V	Funktionale Programmierung	681
19	Lambda-Ausdrücke, Streams und Pipeline-Operationen	683
19.1	Lambda-Ausdrücke	683
19.1.1	Lambda-Ausdrücke in Aktion – zwei Beispiele	684
19.1.2	Lambda-Ausdrücke im Detail	687
19.1.3	Lambda-Ausdrücke und funktionale Interfaces	689
19.1.4	Vordefinierte funktionale Interfaces	691
19.1.5	Anwendungen auf Datenstrukturen	693
19.1.6	Methodenreferenzen als Lambda-Ausdrücke	695
19.1.7	Zugriff auf Variablen aus der Umgebung innerhalb eines Lambda-Ausdrucks	698
19.1.8	Übungsaufgaben	699
19.2	Streams und Pipeline-Operationen	700
19.2.1	Streams in Aktion	701

19.2.2 Streams und Pipelines im Detail	703
19.2.3 Erzeugen von endlichen und unendlichen Streams	704
19.2.4 Die Stream-API	706
19.2.5 Übungsaufgaben	709
VI Abschluss, Ausblick und Anhang	711
20 Blick über den Tellerrand	713
20.1 JShell für kleine Skripte	714
20.2 Das Java-Modulsystem	718
20.3 Bühne frei für JavaFX	725
20.4 Beginn einer neuen Zeitrechnung	734
20.5 Webprogrammierung und verteilte Systeme	737
20.6 Zu guter Letzt	739
A Der Weg zum guten Programmierer	741
A.1 Die goldenen Regeln der Code-Formatierung	742
A.2 Die goldenen Regeln der Namensgebung	745
A.3 Zusammenfassung	747
B Ohne Werkzeug geht es nicht	749
B.1 Die API-Dokumentation zum Nachschlagen	750
B.2 Die IDE, dein Freund und Helfer	752
B.3 Alle Versionen stets im Griff	754
B.4 Testen bitte nicht vergessen	756
B.5 Der Automat kann es besser	758
C Die Klasse <code>IOTools</code> – Tastatureingaben in Java	761
C.1 Kurzbeschreibung	761
C.2 Anwendung der <code>IOTools</code> -Methoden	762
Glossar	765
Literaturverzeichnis	779
Stichwortverzeichnis	783