



Inhaltsverzeichnis

Dietmar Ratz, Jens Scheffler, Detlef Seese, Jan Wiesenberger

Grundkurs Programmieren in Java

ISBN (Buch): 978-3-446-44073-9

ISBN (E-Book): 978-3-446-44110-1

Weitere Informationen oder Bestellungen unter

<http://www.hanser-fachbuch.de/978-3-446-44073-9>

sowie im Buchhandel.

Inhaltsverzeichnis

Vorwort	17
1 Einleitung	19
1.1 Java – mehr als nur kalter Kaffee?	19
1.2 Java für Anfänger – das Konzept dieses Buches	20
1.3 Zusatzmaterial und Kontakt zu den Autoren	22
1.4 Verwendete Schreibweisen	22
2 Einige Grundbegriffe aus der Welt des Programmierens	23
2.1 Computer, Software, Informatik und das Internet	23
2.2 Was heißt Programmieren?	26
I Einstieg in das Programmieren in Java	29
3 Aller Anfang ist schwer	31
3.1 Mein erstes Programm	31
3.2 Formeln, Ausdrücke und Anweisungen	32
3.3 Zahlenbeispiele	33
3.4 Verwendung von Variablen	34
3.5 „Auf den Schirm!“	34
3.6 Das Programmgerüst	35
3.7 Eingeben, übersetzen und ausführen	37
3.8 Übungsaufgaben	38
4 Grundlagen der Programmierung in Java	39
4.1 Grundelemente eines Java-Programms	39
4.1.1 Kommentare	41
4.1.2 Bezeichner und Namen	43
4.1.3 Literale	44
4.1.4 Reservierte Wörter, Schlüsselwörter	44

4.1.5	Trennzeichen	45
4.1.6	Interpunktionszeichen	46
4.1.7	Operatorsymbole	46
4.1.8	import -Anweisungen	47
4.1.9	Zusammenfassung	48
4.1.10	Übungsaufgaben	48
4.2	Erste Schritte in Java	49
4.2.1	Grundstruktur eines Java-Programms	50
4.2.2	Ausgaben auf der Konsole	51
4.2.3	Eingaben von der Konsole	52
4.2.4	Schöner programmieren in Java	53
4.2.5	Zusammenfassung	54
4.2.6	Übungsaufgaben	54
4.3	Einfache Datentypen	55
4.3.1	Ganzzahlige Datentypen	55
4.3.2	Gleitkommatypen	57
4.3.3	Der Datentyp char für Zeichen	59
4.3.4	Zeichenketten	60
4.3.5	Der Datentyp boolean für Wahrheitswerte	60
4.3.6	Implizite und explizite Typumwandlungen	60
4.3.7	Zusammenfassung	62
4.3.8	Übungsaufgaben	62
4.4	Der Umgang mit einfachen Datentypen	63
4.4.1	Variablen	63
4.4.2	Operatoren und Ausdrücke	67
4.4.2.1	Arithmetische Operatoren	68
4.4.2.2	Bitoperatoren	70
4.4.2.3	Zuweisungsoperator	72
4.4.2.4	Vergleichsoperatoren und logische Operatoren	73
4.4.2.5	Inkrement- und Dekrementoperatoren	75
4.4.2.6	Priorität und Auswertungsreihenfolge der Operatoren	76
4.4.3	Allgemeine Ausdrücke	77
4.4.4	Ein- und Ausgabe	78
4.4.4.1	Statischer Import der IOTools-Methoden	79
4.4.5	Zusammenfassung	81
4.4.6	Übungsaufgaben	81
4.5	Anweisungen und Ablaufsteuerung	84
4.5.1	Anweisungen	85
4.5.2	Blöcke und ihre Struktur	85
4.5.3	Entscheidungsanweisung	86
4.5.3.1	Die if -Anweisung	86

4.5.3.2	Die switch -Anweisung	87
4.5.4	Wiederholungsanweisungen, Schleifen	89
4.5.4.1	Die for -Anweisung	89
4.5.4.2	Vereinfachte for -Schleifen-Notation	90
4.5.4.3	Die while -Anweisung	91
4.5.4.4	Die do -Anweisung	91
4.5.4.5	Endlosschleifen	92
4.5.5	Sprungbefehle und markierte Anweisungen	93
4.5.6	Zusammenfassung	95
4.5.7	Übungsaufgaben	95
5	Referenzdatentypen	105
5.1	Felder	107
5.1.1	Was sind Felder?	109
5.1.2	Deklaration, Erzeugung und Initialisierung von Feldern	110
5.1.3	Felder unbekannter Länge	113
5.1.4	Referenzen	115
5.1.5	Ein besserer Terminkalender	119
5.1.6	Mehrdimensionale Felder	121
5.1.7	Mehrdimensionale Felder unterschiedlicher Länge	124
5.1.8	Vorsicht, Falle: Kopieren von mehrdimensionalen Feldern	126
5.1.9	Vereinfachte for -Schleifen-Notation	127
5.1.10	Zusammenfassung	129
5.1.11	Übungsaufgaben	129
5.2	Klassen	132
5.2.1	Was sind Klassen?	133
5.2.2	Deklaration und Instantiierung von Klassen	134
5.2.3	Komponentenzugriff bei Objekten	135
5.2.4	Ein erstes Adressbuch	136
5.2.5	Klassen als Referenzdatentyp	138
5.2.6	Felder von Klassen	141
5.2.7	Vorsicht, Falle: Kopieren von geschachtelten Referenzdatentypen	144
5.2.8	Auslagern von Klassen	145
5.2.9	Zusammenfassung	147
5.2.10	Übungsaufgaben	147
6	Methoden, Unterprogramme	149
6.1	Methoden	150
6.1.1	Was sind Methoden?	150
6.1.2	Deklaration von Methoden	151
6.1.3	Parameterübergabe und Ergebnisrückgabe	152

6.1.4	Aufruf von Methoden	154
6.1.5	Überladen von Methoden	155
6.1.6	Variable Argument-Anzahl bei Methoden	157
6.1.7	Vorsicht, Falle: Referenzen als Parameter	158
6.1.8	Sichtbarkeit und Verdecken von Variablen	160
6.1.9	Zusammenfassung	162
6.1.10	Übungsaufgaben	162
6.2	Rekursiv definierte Methoden	163
6.2.1	Motivation	163
6.2.2	Gute und schlechte Beispiele für rekursive Methoden	165
6.2.3	Zusammenfassung	168
6.3	Die Methode <code>main</code>	168
6.3.1	Kommandozeilenparameter	169
6.3.2	Anwendung der vereinfachten <code>for</code> -Schleifen-Notation	170
6.3.3	Zusammenfassung	171
6.3.4	Übungsaufgaben	171
6.4	Methoden aus anderen Klassen aufrufen	173
6.4.1	Klassenmethoden	173
6.4.2	Die Methoden der Klasse <code>java.lang.Math</code>	174
6.4.3	Statischer Import	175
6.5	Methoden von Objekten aufrufen	176
6.5.1	Instanzmethoden	176
6.5.2	Die Methoden der Klasse <code>java.lang.String</code>	177
6.6	Übungsaufgaben	180
 II Objektorientiertes Programmieren in Java		185
7	Die objektorientierte Philosophie	187
7.1	Die Welt, in der wir leben	187
7.2	Programmierparadigmen – Objektorientierung im Vergleich	188
7.3	Die vier Grundpfeiler objektorientierter Programmierung	190
7.3.1	Generalisierung	190
7.3.2	Vererbung	192
7.3.3	Kapselung	195
7.3.4	Polymorphismus	196
7.3.5	Weitere wichtige Grundbegriffe	197
7.4	Modellbildung – von der realen Welt in den Computer	198
7.4.1	Grafisches Modellieren mit UML	198
7.4.2	Entwurfsmuster	199
7.5	Zusammenfassung	200
7.6	Übungsaufgaben	201

8	Der grundlegende Umgang mit Klassen	203
8.1	Vom Referenzdatentyp zur Objektorientierung	203
8.2	Instanzmethoden	205
8.2.1	Zugriffsrechte	205
8.2.2	Was sind Instanzmethoden?	206
8.2.3	Instanzmethoden zur Validierung von Eingaben	209
8.2.4	Instanzmethoden als erweiterte Funktionalität	210
8.3	Statische Komponenten einer Klasse	211
8.3.1	Klassenvariablen und -methoden	212
8.3.2	Klassenkonstanten	214
8.4	Instantiierung und Initialisierung	217
8.4.1	Konstruktoren	217
8.4.2	Überladen von Konstruktoren	219
8.4.3	Der statische Initialisierer	221
8.4.4	Der Mechanismus der Objekterzeugung	224
8.5	Zusammenfassung	229
8.6	Übungsaufgaben	229
9	Vererbung und Polymorphismus	249
9.1	Wozu braucht man Vererbung?	249
9.1.1	Aufgabenstellung	249
9.1.2	Analyse des Problems	250
9.1.3	Ein erster Ansatz	250
9.1.4	Eine Klasse für sich	251
9.1.5	Stärken der Vererbung	252
9.1.6	Vererbung verhindern durch final	255
9.1.7	Übungsaufgaben	256
9.2	Die super -Referenz	257
9.3	Überschreiben von Methoden und Variablen	259
9.3.1	Dynamisches Binden	259
9.3.2	Überschreiben von Methoden verhindern durch final	261
9.4	Die Klasse <code>java.lang.Object</code>	262
9.5	Übungsaufgaben	265
9.6	Abstrakte Klassen und Interfaces	266
9.7	Übungsaufgaben	269
9.8	Weiteres zum Thema Objektorientierung	274
9.8.1	Erstellen von Paketen	274
9.8.2	Zugriffsrechte	276
9.8.3	Innere Klassen	277
9.8.4	Anonyme Klassen	282
9.9	Zusammenfassung	284
9.10	Übungsaufgaben	284

10 Exceptions und Errors	295
10.1 Eine Einführung in Exceptions	296
10.1.1 Was ist eine Exception?	296
10.1.2 Übungsaufgaben	298
10.1.3 Abfangen von Exceptions	298
10.1.4 Ein Anwendungsbeispiel	299
10.1.5 Die <code>RuntimeException</code>	302
10.1.6 Übungsaufgaben	303
10.2 Exceptions für Fortgeschrittene	305
10.2.1 Definieren eigener Exceptions	305
10.2.2 Übungsaufgaben	307
10.2.3 Vererbung und Exceptions	307
10.2.4 Vorsicht, Falle!	311
10.2.5 Der finally -Block	313
10.2.6 Die Klassen <code>Throwable</code> und <code>Error</code>	317
10.2.7 Zusammenfassung	319
10.2.8 Übungsaufgaben	319
10.3 Assertions	320
10.3.1 Zusicherungen im Programmcode	320
10.3.2 Compilieren des Programmcodes	321
10.3.3 Ausführen des Programmcodes	322
10.3.4 Zusammenfassung	322
11 Fortgeschrittene objektorientierte Programmierung	323
11.1 Aufzählungstypen	324
11.1.1 Deklaration eines Aufzählungstyps	324
11.1.2 Instanzmethoden der enum -Objekte	325
11.1.3 Selbstdefinierte Instanzmethoden für enum -Objekte	325
11.1.4 Übungsaufgaben	327
11.2 Generische Datentypen	329
11.2.1 Generizität in alten Java-Versionen	329
11.2.2 Generizität ab Java 5.0	332
11.2.3 Einschränkungen der Typ-Parameter	334
11.2.4 Wildcards	336
11.2.5 Bounded Wildcards	337
11.2.6 Generische Methoden	339
11.2.7 Ausblick	341
11.2.8 Übungsaufgaben	341
11.3 Sortieren von Feldern und das Interface <code>Comparable</code>	346
12 Einige wichtige Hilfsklassen	349
12.1 Die Klasse <code>StringBuffer</code>	349

12.1.1	Arbeiten mit <code>String</code> -Objekten	349
12.1.2	Arbeiten mit <code>StringBuffer</code> -Objekten	352
12.1.3	Übungsaufgaben	354
12.2	Die Wrapper-Klassen (Hüll-Klassen)	355
12.2.1	Arbeiten mit „eingepackten“ Daten	355
12.2.2	Aufbau der Wrapper-Klassen	356
12.2.3	Ein Anwendungsbeispiel	359
12.2.4	Automatische Typwandlung für die Wrapper-Klassen	360
12.2.5	Übungsaufgaben	362
12.3	Die Klassen <code>BigInteger</code> und <code>BigDecimal</code>	363
12.3.1	Arbeiten mit langen Ganzzahlen	363
12.3.2	Aufbau der Klasse <code>BigInteger</code>	365
12.3.3	Übungsaufgaben	367
12.3.4	Arbeiten mit langen Gleitkommazahlen	367
12.3.5	Aufbau der Klasse <code>BigDecimal</code>	370
12.3.6	Viele Stellen von Nullstellen gefällig?	373
12.3.7	Übungsaufgaben	374
12.4	Die Klasse <code>DecimalFormat</code>	375
12.4.1	Standard-Ausgaben in Java	375
12.4.2	Arbeiten mit <code>Format</code> -Objekten	376
12.4.3	Vereinfachte formatierte Ausgabe	378
12.4.4	Übungsaufgaben	379
12.5	Die Klassen <code>Date</code> und <code>Calendar</code>	379
12.5.1	Arbeiten mit „Zeitpunkten“	380
12.5.2	Auf die Plätze, fertig, los!	381
12.5.3	Spezielle <code>Calendar</code> -Klassen	382
12.5.4	Noch einmal: Zeitmessung	384
12.5.5	Übungsaufgaben	386
12.6	Die Klassen <code>SimpleDateFormat</code> und <code>DateFormat</code>	386
12.6.1	Arbeiten mit <code>Format</code> -Objekten für Datum/Zeit-Angaben	386
12.6.2	Übungsaufgaben	391
12.7	Die <code>Collection</code> -Klassen	391
12.7.1	„Sammlungen“ von Objekten – der Aufbau des Interface <code>Collection</code>	391
12.7.2	„Sammlungen“ durchgehen – der Aufbau des Interface <code>Iterator</code>	394
12.7.3	Mengen	395
12.7.3.1	Das Interface <code>Set</code>	395
12.7.3.2	Die Klasse <code>HashSet</code>	395
12.7.3.3	Das Interface <code>SortedSet</code>	397
12.7.3.4	Die Klasse <code>TreeSet</code>	398
12.7.4	Listen	399

12.7.4.1	Das Interface List	400
12.7.4.2	Die Klassen ArrayList und LinkedList	400
12.7.4.3	Suchen und Sortieren – die Klassen Collections und Arrays	402
12.7.5	Übungsaufgaben	405
12.8	Die Klasse StringTokenizer	406
12.8.1	Übungsaufgaben	408
 III Grafische Oberflächen in Java		409
13	Aufbau grafischer Oberflächen in Frames – von AWT nach Swing	411
13.1	Grundsätzliches zum Aufbau grafischer Oberflächen	411
13.2	Ein einfaches Beispiel mit dem AWT	413
13.3	Let's swing now!	415
13.4	Etwas „Fill-in“ gefällig?	417
13.5	Die AWT- und Swing-Klassenbibliothek im Überblick	419
13.6	Übungsaufgaben	421
14	Swing-Komponenten	423
14.1	Die abstrakte Klasse Component	423
14.2	Die Klasse Container	424
14.3	Die abstrakte Klasse JComponent	425
14.4	Layout-Manager, Farben und Schriften	426
14.4.1	Die Klasse Color	427
14.4.2	Die Klasse Font	429
14.4.3	Layout-Manager	430
14.4.3.1	Die Klasse FlowLayout	431
14.4.3.2	Die Klasse BorderLayout	433
14.4.3.3	Die Klasse GridLayout	434
14.5	Einige Grundkomponenten	436
14.5.1	Die Klasse JLabel	438
14.5.2	Die abstrakte Klasse AbstractButton	438
14.5.3	Die Klasse JButton	440
14.5.4	Die Klasse JToggleButton	441
14.5.5	Die Klasse JCheckBox	442
14.5.6	Die Klassen JRadioButton und ButtonGroup	443
14.5.7	Die Klasse JComboBox	445
14.5.8	Die Klasse JList	448
14.5.9	Die abstrakte Klasse JTextComponent	451
14.5.10	Die Klassen JTextField und JPasswordField	452
14.5.11	Die Klasse JTextArea	454
14.5.12	Die Klasse JScrollPane	456

14.5.13 Die Klasse <code>JPanel</code>	458
14.6 Spezielle Container, Menüs und Toolbars	460
14.6.1 Die Klasse <code>JFrame</code>	460
14.6.2 Die Klasse <code>JWindow</code>	461
14.6.3 Die Klasse <code>JDialog</code>	461
14.6.4 Die Klasse <code>JMenuBar</code>	465
14.6.5 Die Klasse <code>JToolBar</code>	467
14.7 Übungsaufgaben	470
15 Ereignisverarbeitung	473
15.1 Zwei einfache Beispiele	474
15.1.1 Zufällige Grautöne als Hintergrund	474
15.1.2 Ein interaktiver Bilderrahmen	477
15.2 Programmiervarianten für die Ereignisverarbeitung	481
15.2.1 Innere Klasse als Listener-Klasse	481
15.2.2 Anonyme Klasse als Listener-Klasse	481
15.2.3 Container-Klasse als Listener-Klasse	482
15.2.4 Separate Klasse als Listener-Klasse	483
15.3 Event-Klassen und -Quellen	485
15.4 Listener-Interfaces und Adapter-Klassen	489
15.5 Listener-Registrierung bei den Event-Quellen	494
15.6 Auf die Plätze, fertig, los!	498
15.7 Übungsaufgaben	502
16 Einige Ergänzungen zu Swing-Komponenten	507
16.1 Zeichnen in Swing-Komponenten	507
16.1.1 Grafische Darstellung von Komponenten	507
16.1.2 Das Grafik-Koordinatensystem	508
16.1.3 Die abstrakte Klasse <code>Graphics</code>	509
16.1.4 Ein einfaches Zeichenprogramm	512
16.1.5 Layoutveränderungen und der Einsatz von <code>revalidate</code>	514
16.2 Noch mehr Swing gefällig?	517
16.3 Übungsaufgaben	518
17 Applets	521
17.1 Erstellen und Ausführen von Applets	521
17.1.1 Vom Frame zum Applet am Beispiel	521
17.1.2 Applet in HTML-Datei einbetten	523
17.1.3 Applet über HTML-Datei ausführen	525
17.2 Die Methoden der Klasse <code>JApplet</code>	526
17.3 Zwei Beispiele	528
17.3.1 Auf die Plätze, fertig, los!	529

17.3.2	Punkte verbinden im Applet	532
17.4	Details zur HTML-Einbettung	533
17.4.1	Der Applet-Tag	533
17.4.2	Die Methode <code>showDocument</code>	536
17.5	Sicherheitseinschränkungen bei Applets	538
17.6	Übungsaufgaben	542
IV	Threads, Datenströme und Netzwerk-Anwendungen	545
18	Parallele Programmierung mit Threads	547
18.1	Ein einfaches Beispiel	547
18.2	Threads in Java	549
18.2.1	Die Klasse <code>Thread</code>	550
18.2.2	Das Interface <code>Runnable</code>	554
18.2.3	Threads vorzeitig beenden	556
18.3	Wissenswertes über Threads	558
18.3.1	Lebenszyklus eines Threads	558
18.3.2	Thread-Scheduling	560
18.3.3	Dämon-Threads und Thread-Gruppen	560
18.4	Thread-Synchronisation und -Kommunikation	561
18.4.1	Das Leser/Schreiber-Problem	562
18.4.2	Das Erzeuger/Verbraucher-Problem	566
18.5	Threads in Frames und Applets	573
18.5.1	Auf die Plätze, fertig, los!	573
18.5.2	Spielereien	577
18.5.3	Swing-Komponenten sind nicht Thread-sicher	579
18.6	Übungsaufgaben	580
19	Ein- und Ausgabe über I/O-Streams	583
19.1	Grundsätzliches zu I/O-Streams in Java	584
19.2	Dateien und Verzeichnisse – Die Klasse <code>File</code>	584
19.3	Ein- und Ausgabe über Character-Streams	587
19.3.1	Einfache <code>Reader</code> - und <code>Writer</code> -Klassen	588
19.3.2	Gepufferte <code>Reader</code> - und <code>Writer</code> -Klassen	591
19.3.3	Die Klasse <code>StreamTokenizer</code>	593
19.3.4	Die Klasse <code>PrintWriter</code>	594
19.3.5	Die Klassen <code>IOTools</code> und <code>Scanner</code>	596
19.3.5.1	Was machen eigentlich die <code>IOTools</code> ?	596
19.3.5.2	Konsoleneingabe über ein <code>Scanner</code> -Objekt	597
19.4	Ein- und Ausgabe über Byte-Streams	598
19.4.1	Einige <code>InputStream</code> - und <code>OutputStream</code> -Klassen	599
19.4.2	Die Serialisierung und Deserialisierung von Objekten	601

19.4.3	Die Klasse <code>PrintStream</code>	603
19.5	Einige abschließende Bemerkungen	603
19.6	Übungsaufgaben	604
20	Client/Server-Programmierung in Netzwerken	607
20.1	Wissenswertes über Netzwerk-Kommunikation	608
20.1.1	Protokolle	608
20.1.2	IP-Adressen	610
20.1.3	Ports und Sockets	611
20.2	Client/Server-Programmierung	612
20.2.1	Die Klassen <code>ServerSocket</code> und <code>Socket</code>	613
20.2.2	Ein einfacher Server	615
20.2.3	Ein einfacher Client	618
20.2.4	Ein Server für mehrere Clients	619
20.2.5	Ein Mehrzweck-Client	622
20.3	Wissenswertes über URLs	625
20.3.1	Client/Server-Kommunikation über URLs	625
20.3.2	Netzwerkverbindungen in Applets	626
20.4	Übungsaufgaben	627
V	Aktuelles, Ausblick und Anhang	631
21	Neuerungen in Java 7	633
21.1	Spracherweiterungen	633
21.1.1	Elementare Datentypen und Anweisungen	633
21.1.1.1	Binäre ganzzahlige Literalkonstanten	633
21.1.1.2	Unterstrich als Trennzeichen in Literalkonstanten	634
21.1.1.3	Strings in der switch -Anweisung	635
21.1.2	Verkürzte Notation bei generischen Datentypen	638
21.1.3	Ausnahmebehandlung	642
21.1.3.1	Mehrere Ausnahme-Typen in einem catch -Block	642
21.1.3.2	try -Block mit Ressourcen	645
21.2	Erweiterungen der Klassenbibliothek	648
21.2.1	Dateien und Verzeichnisse	648
21.2.1.1	Das Interface <code>Path</code> und die Klasse <code>Paths</code>	648
21.2.1.2	Die Klasse <code>Files</code>	649
21.2.2	Grafische Oberflächen	652
22	Neuerungen in Java 8	655
22.1	Lambda-Ausdrücke	655
22.1.1	Lambda-Ausdrücke in Aktion – zwei Beispiele	656
22.1.2	Lambda-Ausdrücke im Detail	659

22.1.3	Lambda-Ausdrücke und funktionale Interfaces	661
22.1.4	Vordefinierte funktionale Interfaces und Anwendungen auf Datenstrukturen	663
22.1.5	Methoden-Referenzen als Lambda-Ausdrücke	668
22.1.6	Zugriff auf Variablen aus der Umgebung innerhalb eines Lambda-Ausdrucks	670
22.2	Interfaces mit Default-Methoden und statischen Methoden	672
22.2.1	Deklaration von Default-Methoden	672
22.2.2	Deklaration von statischen Methoden	673
22.2.3	Auflösung von Namensgleichheiten bei Default-Methoden	674
22.2.4	Interfaces und abstrakte Klassen in Java 8	676
22.3	Streams und Pipeline-Operationen	676
22.3.1	Streams in Aktion	677
22.3.2	Streams und Pipelines im Detail	679
22.3.3	Erzeugen von endlichen und unendlichen Streams	680
22.3.4	Die Stream-API	682
23	Blick über den Tellerrand	687
23.1	Der Vorhang fällt	687
23.2	A fool with a tool	688
23.3	Alles umsonst?	689
23.4	Und fachlich?	690
23.5	Zu guter Letzt	692
A	Der Weg zum guten Programmierer	693
A.1	Die goldenen Regeln der Code-Formatierung	694
A.2	Die goldenen Regeln der Namensgebung	697
A.3	Zusammenfassung	699
B	Die Klasse <code>IOTools</code> – Tastatureingaben in Java	701
B.1	Kurzbeschreibung	701
B.2	Anwendung der <code>IOTools</code> -Methoden	702
C	Der Umgang mit der API-Spezifikation	705
C.1	Der Aufbau der API-Spezifikation	705
C.2	Der praktische Einsatz der API-Spezifikation	706
D	Glossar	711
	Literaturverzeichnis	725
	Stichwortverzeichnis	729