
Inhaltsverzeichnis

Teil I Objektorientiertes Programmieren

1	Objekte und Klassen	3
1.1	Objekte	3
1.2	Beschreibung von Objekten: Klassen	6
1.3	Klassen und Konstruktormethoden	8
1.3.1	Beispiel: Punkte im \mathbb{R}^2	8
1.3.2	Klassen in JAVA	9
1.3.3	Konstruktor-Methoden	10
1.4	Objekte als Attribute von Objekten	14
1.4.1	Beispiel: Linien im \mathbb{R}^2	14
1.4.2	Anonyme Objekte	16
1.5	Objekte in Reih und Glied: Arrays	16
1.5.1	Beispiel: Polygone im \mathbb{R}^2	17
1.5.2	Arrays: Eine erste Einführung	18
1.6	Zusammenfassung: Objekte und Klassen	21
2	Typen, Werte und Variablen	23
2.1	Die elementaren Datentypen von JAVA	24
2.1.1	Die Wahrheitswerte	25
2.1.2	Die ganzen Zahlen \mathbb{Z}	26
2.1.3	Die Gleitpunktzahlen \mathbb{R}	27
2.1.4	Ascii und Unicode	29
2.1.5	Strings	31
2.2	Typen und Klassen, Werte und Objekte	33
2.3	Die Benennung von Werten: Variablen	34
2.4	Konstanten: Das hohe Gut der Beständigkeit	36
2.5	Metamorphosen	37
2.5.1	Casting	37
2.5.2	Von Typen zu Klassen (und zurück)	39
2.6	Zusammenfassung	40

3	Methoden	41
3.1	Methoden sind Prozeduren oder Funktionen	41
3.1.1	Funktionen	41
3.1.2	Prozeduren	43
3.1.3	Methoden und Klassen	44
3.1.4	Overloading (Überlagerung)	46
3.2	Lokale Variablen und Konstanten	47
3.2.1	Lokale Variablen	47
3.2.2	Lokale Konstanten	48
3.2.3	Parameter als verkappte lokale Variablen*	49
3.3	Beispiele: Punkte und Linien	50
3.3.1	Die Klasse <code>Point</code>	50
3.3.2	Die Klasse <code>Line</code>	54
3.3.3	Private Hilfsmethoden	55
3.3.4	Methoden mit variabler Parameterzahl*	56
3.3.5	Fazit: Methoden sind Funktionen oder Prozeduren	56
4	Programmieren in Java – Eine erste Einführung	59
4.1	Programme schreiben und ausführen	59
4.1.1	Der Programmierprozess	60
4.1.2	Die Hauptklasse und die Methode <code>main</code>	62
4.2	Ein Beispiel mit Physik	63
4.3	Bibliotheken (Packages)	66
4.3.1	Packages: Eine erste Einführung	67
4.3.2	Öffentlich, halböffentlich und privat	67
4.3.3	Standardpackages von JAVA	68
4.3.4	Die Java-Klasse <code>Math</code>	68
4.3.5	Die Java-Klasse <code>System</code>	70
4.3.6	Die Klassen <code>Terminal</code> und <code>Console</code> : Einfache Ein-/Ausgabe	71
4.3.7	Kleine Beispiele mit Grafik	73
4.3.8	Zeichnen in JAVA: Elementare Grundbegriffe	75

Teil II Ablaufkontrolle

5	Kontrollstrukturen	81
5.1	Ausdrücke	81
5.2	Elementare Anweisungen und Blöcke	82
5.3	Man muss sich auch entscheiden können	83
5.3.1	Die <code>if</code> -Anweisung	84
5.3.2	Die <code>switch</code> -Anweisung	85
5.4	Immer und immer wieder: Iteration	87
5.4.1	Die <code>while</code> -Schleife	87
5.4.2	Die <code>for</code> -Schleife	90

5.4.3	Die <code>break</code> - und <code>continue</code> -Anweisung (und <code>return</code>)	91
5.5	Beispiele: Schleifen und Arrays	93
5.6	Die <code>assert</code> -Anweisung	98
5.7	Zusammenfassung: Kontrollstrukturen	99
6	Rekursion	101
6.1	Rekursive Methoden	102
6.2	Funktioniert das wirklich?	104

Teil III Eine Sammlung von Algorithmen

7	Aspekte der Programmiermethodik	109
7.1	Man muss sein Tun auch erläutern: Dokumentation	109
7.1.1	Kommentare im Programm	110
7.1.2	Allgemeine Dokumentation	111
7.2	Zusicherungen (Assertions)	112
7.2.1	Die wichtigsten Regeln des Hoare-Kalküls	115
7.2.2	Terminierung	117
7.3	Aufwand	120
7.4	Testen	124
7.5	Beispiel: Mittelwert und Standardabweichung	126
7.6	Beispiel: Fläche eines Polygons	127
7.7	Beispiel: Sieb des Eratosthenes	130
7.8	Beispiel Primzahltest: Zeuge der Verteidigung	132
7.8.1	Zur Motivation: Kryptographie mittels Primzahlen	132
7.8.2	Ein probabilistischer Primzahltest	134
7.9	Beispiel: Zinsrechnung	138
8	Suchen und Sortieren	143
8.1	Ordnung ist die halbe Suche	143
8.2	Wer sucht, der findet (oder auch nicht)	144
8.2.1	Lineares Suchen: Die <i>British-Museum Method</i>	144
8.2.2	Suchen mit Bisektion	146
8.3	Wer sortiert, findet schneller	148
8.3.1	<i>Selectionsort</i>	151
8.3.2	<i>Insertionsort</i>	152
8.3.3	<i>Quicksort</i>	154
8.3.4	<i>Mergesort</i>	158
8.3.5	<i>Heapsort</i>	160
8.3.6	Mit Mogeln gehts schneller: <i>Bucketsort</i>	166
8.3.7	<i>Verwandte Probleme</i>	166

9	Numerische Algorithmen	169
9.1	Vektoren und Matrizen	169
9.2	Gleichungssysteme: Gauß-Elimination	172
9.2.1	Lösung von Dreieckssystemen	175
9.2.2	<i>LU</i> -Zerlegung	176
9.2.3	Pivot-Elemente	178
9.2.4	Nachiteration	179
9.2.5	Testen mit Probe	180
9.3	Wurzelberechnung und Nullstellen von Funktionen	180
9.4	Differenzieren	183
9.5	Integrieren	186
9.6	Polynom-Interpolation	189
9.6.1	Für Geizhalse: Speicherplatz sparen	194
9.6.2	Extrapolation	195
9.7	Spline-Interpolation	198
9.8	Interpolation für Grafik (Spline, B-Spline, Bezier)	205
9.8.1	Parametrische Splines	205
9.8.2	Bezier-Splines	206
9.8.3	B-Splines	207
9.9	Lösung einfacher Differenzialgleichungen	213
9.9.1	Einfache Einschrittverfahren	215
9.9.2	Runge-Kutta-Verfahren	216
9.9.3	Mehrschrittverfahren	217
9.9.4	Extrapolation	218
9.9.5	Schrittweitensteuerung	218

Teil IV Weitere Konzepte objektorientierter Programmierung		
---	--	--

10	Vererbung	223
10.1	Vererbung = Subtyp?	223
10.2	Sub- und Superklassen in JAVA	226
10.2.1	„Mutierte“ Vererbung und dynamische Bindung	227
10.2.2	Was bist du?	229
10.2.3	Ende der Vererbung: <code>Object</code> und <code>final</code>	230
10.2.4	Mit <code>super</code> zur Superklasse	232
10.2.5	Casting: Zurück zur Sub- oder Superklasse	233
10.3	Abstrakte Klassen	234
11	Interfaces	237
11.1	Mehrfachvererbung und Interfaces	237
11.2	Anwendung: Suchen und Sortieren richtig gelöst	241
11.2.1	Das Interface <code>Sortable</code>	242
11.2.2	Die JAVA-Interfaces <code>Comparable</code> und <code>Comparator</code>	244
11.3	Anwendung: Methoden höherer Ordnung	244

11.3.1	Fun als Interface	245
11.3.2	Interpolation als Implementierung von Fun	246
11.3.3	Ein bisschen Eleganz: Methoden als Resultate	247
12	Generizität (Polymorphie)	251
12.1	Des einen Vergangenheit ist des anderen Zukunft	251
12.2	Die Idee der Polymorphie (Generizität)	252
12.3	Generische Klassen und Interfaces in JAVA	252
12.3.1	Beschränkte Typparameter	255
12.3.2	Vererbung und Generizität	256
12.4	Generische Methoden in JAVA	259
13	Und dann war da noch	261
13.1	Einer für alle: <code>static</code>	261
13.1.1	Statischer Import	264
13.1.2	Initialisierung	264
13.2	Innere und lokale Klassen	265
13.3	Anonyme Klassen	267
13.4	Enumerationstypen in JAVA 5	268
14	Namen, Scopes und Packages	269
14.1	Das Prinzip der (Un-)Sichtbarkeit	269
14.2	Gültigkeitsbereich (<i>Scope</i>)	270
14.2.1	Klassen als Gültigkeitsbereich	271
14.2.2	Methoden als Gültigkeitsbereich	272
14.2.3	Blöcke als Gültigkeitsbereich	272
14.2.4	Verschattung (<i>holes in the scope</i>)	273
14.2.5	Überlagerung	274
14.3	Packages: Scopes „im Großen“	274
14.3.1	Volle Klassennamen	276
14.3.2	Import	276
14.4	Geheimniskrämerei	277
14.4.1	Geschlossene Gesellschaft: <code>Package</code>	277
14.4.2	Herstellen von Öffentlichkeit: <code>public</code>	277
14.4.3	Maximale Verschlossenheit: <code>private</code>	278
14.4.4	Vertrauen zu Subklassen: <code>protected</code>	279
14.4.5	Zusammenfassung	279

Teil V Datenstrukturen

15	Hashtabellen	283
15.1	Von Arrays zu Hashtabellen	283

15.2	Zum Design von Hashfunktionen	285
15.2.1	Hashfunktionen	286
15.2.2	Kollisionsauflösung	287
15.2.3	Aufwand	289
15.3	Hashing in JAVA	291
15.3.1	Die Klassen <code>Hashtable</code> , <code>HashMap</code> und <code>HashSet</code>	291
15.3.2	Die Methode <code>hashCode</code> der Klasse <code>Object</code>	292
15.4	Weitere Anwendungen von Hashverfahren	293
16	Referenzen	295
16.1	Nichts währt ewig: Lebensdauern	295
16.2	Referenzen: „Ich weiß, wo mans findet“	297
16.3	Referenzen in JAVA	298
16.3.1	Zur Funktionsweise von Referenzen	298
16.3.2	Referenzen und Methodenaufrufe	301
16.3.3	Wer bin ich?: <code>this</code>	303
16.4	Gleichheit und Kopien	303
16.5	Die Wahrheit über Arrays	305
16.6	Abfallbeseitigung (<i>Garbage collection</i>)	306
17	Listen	309
17.1	Listen als verkettete Objekte	309
17.1.1	Listenzellen	310
17.1.2	Elementares Arbeiten mit Listen	312
17.1.3	Traversieren von Listen	313
17.1.4	Generische Listen	315
17.1.5	Zirkuläre Listen	316
17.1.6	Doppelt verkettete Listen	317
17.1.7	Eine methodische Schwäche und ihre Gefahren	318
17.2	Listen als Abstrakter Datentyp (<code>LinkedList</code>)	319
17.3	Listenartige Strukturen in JAVA	322
17.3.1	<code>Collection</code>	324
17.3.2	<code>List</code>	325
17.3.3	<code>Set</code>	326
17.3.4	<code>LinkedList</code> , <code>ArrayList</code> und <code>Vector</code>	326
17.3.5	<code>Stack</code>	327
17.3.6	<code>Queue</code> („Warteschlange“)	328
17.3.7	<code>Priority Queues</code> : Vordrängeln ist erlaubt	329
17.4	Einer nach dem andern: Iteratoren	329
17.4.1	Implementierung	331
17.4.2	Neue <code>for</code> -Schleife in JAVA 5 und das Interface <code>Iterable</code>	332

18	Bäume	333
18.1	Bäume: Grundbegriffe	333
18.2	Implementierung durch Verkettung	335
18.2.1	Binärbäume	335
18.2.2	Allgemeine Bäume	337
18.2.3	Binärbäume als Abstrakter Datentyp	338
18.3	Traversieren von Bäumen: Baum-Iteratoren	340
18.4	Suchbäume (geordnete Bäume)	344
18.4.1	Suchbäume als Abstrakter Datentyp: SearchTree	346
18.4.2	Implementierung von Suchbäumen	348
18.5	Balancierte Suchbäume	353
18.5.1	2-3-Bäume und 2-3-4-Bäume	354
18.5.2	Rot-Schwarz-Bäume	356
18.6	Baumdarstellung von Sprachen (Syntaxbäume)	363
19	Graphen	369
19.1	Beispiele für Graphen	369
19.2	Grundbegriffe	371
19.3	Eine abstrakte Sicht auf Graphen	372
19.4	Adjazenzlisten und Adjazenzmatrizen	376
19.5	Traversierung von Graphen	379
19.5.1	Entwurfsmuster für Graphtraversierung	379
19.5.2	Tiefen- und Breitensuche	382
19.5.3	Eine genuin objektorientierte Sicht von Graphtraversierung	383
19.6	Anwendungen der Graphtraversierung	385
19.6.1	Erreichbarkeit (von einem Knoten aus)	385
19.6.2	Variation: Aufspannender Baum	386
19.6.3	Variation: Kürzeste Wege (von einem Knoten aus)	387
19.6.4	Transitive Hülle	392
19.7	Relationen-Probleme als Graphprobleme	395
19.7.1	Topologisches Sortieren	395
19.7.2	Strenge Zusammenhangskomponenten	398
19.8	Weitere Graphalgorithmen	402

Teil VI Programmierung von Software-Systemen

20	Keine Regel ohne Ausnahmen: Exceptions	407
20.1	Manchmal gehts eben schief	407
20.2	Exceptions	409
20.3	Man versucht halt mal: try und catch	411
20.4	Exceptions verkünden: throw	413
20.5	Methoden mit Exceptions: throws	414

21 Ein- und Ausgabe	417
21.1 Ohne Verwaltung geht gar nichts	418
21.1.1 Pfade und Dateinamen in Windows und Unix	419
21.1.2 <code>File</code> : Die Klasse zur Dateiverwaltung	420
21.1.3 Programmieren der Dateiverwaltung	422
21.2 Was man Lesen und Schreiben kann	423
21.3 Dateien mit Direktzugriff („Externe Arrays“)	426
21.4 Sequenzielle Dateien („Externe Listen“, Ströme)	428
21.4.1 Die abstrakte Superklasse <code>InputStream</code>	429
21.4.2 Die konkreten Klassen für Eingabeströme	429
21.4.3 Ausgabeströme	431
21.4.4 Das Ganze nochmals mit Unicode: Reader und Writer	432
21.5 Programmieren mit Dateien und Strömen	433
21.6 Terminal-Ein-/Ausgabe	435
21.7 ... und noch ganz viel Spezielles	438
21.7.1 Serialisierung	438
21.7.2 Interne Kommunikation über Pipes	439
21.7.3 Konkatenation von Strömen: <code>SequenceInputStream</code>	440
21.7.4 Simulierte Ein-/Ausgabe	440
22 Konkurrenz belebt das Geschäft: Threads	441
22.1 Threads: Leichtgewichtige Prozesse	441
22.2 Die Klasse <code>Thread</code>	445
22.2.1 Entstehen – Arbeiten – Sterben	446
22.2.2 Schlafe nur ein Weilchen ... (<code>sleep</code>)	447
22.2.3 Jetzt ist mal ein anderer dran ... (<code>yield</code>)	448
22.2.4 Ich warte auf dein Ende ... (<code>join</code>)	448
22.2.5 Unterbrich mich nicht! (<code>interrupt</code>)	450
22.2.6 Ich bin wichtiger als du! (Prioritäten)	451
22.3 Synchronisation und Kommunikation	452
22.3.1 Vorsicht, es klemmt!	454
22.3.2 Warten Sie, bis Sie aufgerufen werden! (<code>wait</code> , <code>notify</code>)	455
22.4 Das Interface <code>Runnable</code>	458
22.5 Ist das genug?	459
22.5.1 Gemeinsam sind wir stark (Thread-Gruppen)	459
22.5.2 Dämonen sterben heimlich	460
22.5.3 Zu langsam für die reale Zeit?	460
22.5.4 Vorsicht, veraltet!	461
22.5.5 Neues in Java 5	461
23 Das ist alles so schön bunt hier: Grafik in JAVA	463
23.1 Historische Vorbemerkung	463
23.1.1 Awt und Swing (und andere)	464
23.1.2 Entwicklungsumgebungen	465
23.2 Grundlegende Konzepte von GUIs	466

24 GUI: Layout	469
24.1 Die Superklassen: Component und JComponent	471
24.2 Elementare GUI-Elemente	472
24.2.1 Beschriftungen: Label / JLabel	473
24.2.2 Zum Anklicken: Button / JButton	473
24.2.3 Editierbarer Text: TextField / JTextField	475
24.3 Behälter: Container	478
24.3.1 Das Hauptfenster: Frame / JFrame	479
24.3.2 Lokale Container: Panel / JPanel	483
24.3.3 GUIs entwerfen	483
24.3.4 Layout-Manager	484
24.3.5 Verwendung des statischen Imports von Java 5	490
24.3.6 Mehr über Farben: Color	491
24.3.7 Fenster-Geometrie: Point und Dimension	492
24.3.8 Größenbestimmung von Fenstern	493
24.4 Selbst Zeichnen	495
24.4.1 Die Methode paint	497
24.4.2 Die Methode paintComponent	498
24.4.3 Wenn man nur zeichnen will	498
24.4.4 Zeichnen mit Graphics und Graphics2D	499
25 Hallo Programm! – Hallo GUI!	501
25.1 Auf GUIs ein- und ausgeben	501
25.2 Von Ereignissen getrieben	502
25.3 Immerzu lauschen	504
25.3.1 Beispiel: Eingabe im Displayfeld	504
25.3.2 Arbeiten mit Buttons	506
25.3.3 Listener-Arten	508
26 Beispiel: Taschenrechner	511
26.1 Taschenrechner: Die globale Struktur	512
26.2 Taschenrechner: <i>Model</i>	513
26.3 Taschenrechner: <i>View</i>	516
26.4 Taschenrechner: <i>Control</i>	523
26.5 Fazit	526

Teil VII Ausblick

27 Es gäbe noch viel zu tun	529
27.1 Java und Netzwerke: Von Sockets bis Jini	529
27.1.1 Die OSI-Hierarchie	530
27.1.2 Sockets	533
27.1.3 Wenn die Methoden weit weg sind: RMI	533
27.1.4 Wie komme ich ins Netz? (Jini)	535

27.2	Java und das Web	535
27.2.1	Applets	535
27.2.2	Servlets (<i>Server Applets</i>)	539
27.2.3	JSP: <i>JavaServer Pages</i>	540
27.2.4	Java und XML	540
27.2.5	Java und Email	541
27.3	Sicher ist sicher: Java-Security	541
27.3.1	Sandbox und Security Manager	542
27.3.2	Verschlüsselung und Signaturen	543
27.4	Reflection und Introspection	543
27.5	Java-Komponenten-Technologie: Beans	544
27.6	Java und Datenbanken: JDBC	547
27.7	Direktzugang zum Rechner: Von JNI bis Realzeit	547
27.7.1	Die Java Virtual Machine (JVM)	547
27.7.2	Das Java Native Interface (JNI)	548
27.7.3	Externe Prozesse starten	549
27.7.4	Java und Realzeit	549
A	Anhang: Praktische Hinweise	551
A.1	Java beschaffen	551
A.2	Java installieren	552
A.3	Java-Programme übersetzen (<i>javac</i>)	553
A.3.1	Verwendung von zusätzlichen <i>Directorys</i>	554
A.3.2	Verwendung des <i>Classpath</i>	555
A.3.3	Konflikte zwischen Java 1.4 und Java 5/Java 6	556
A.4	Java-Programme ausführen (<i>java</i> und <i>javaw</i>)	556
A.5	<i>Directorys</i> , <i>Classpath</i> und <i>Packages</i>	558
A.6	Java-Archive verwenden (<i>jar</i>)	559
A.7	Dokumentation generieren mit <i>javadoc</i>	561
A.8	Weitere Werkzeuge	563
A.9	Die Klassen <i>Terminal</i> und <i>Pad</i> dieses Buches	563
A.10	Materialien zu diesem Buch	564
	Literaturverzeichnis	565
	Sachverzeichnis	569

Hinweis: Eine Errata-Liste und weitere Hinweise zu diesem Buch sind über die Web-Adresse <http://www.uebb.cs.tu-berlin.de/books/java> zu erreichen. Näheres findet sich im Anhang.