

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	Beispiel: Elementare Datentypen von JAVA	24
2.2	Typen und Klassen, Werte und Objekte	27
2.3	Die Benennung von Werten: Variablen	27
2.4	Konstanten: Das hohe Gut der Beständigkeit	29
2.5	Metamorphosen	30
2.5.1	Casting	30
2.5.2	Von Typen zu Klassen (und zurück)	32
2.6	Zusammenfassung	33
3	Methoden	35
3.1	Methoden sind Prozeduren oder Funktionen	35
3.1.1	Funktionen	35
3.1.2	Prozeduren	37

3.1.3	Methoden und Klassen	38
3.1.4	Overloading (Überlagerung)	40
3.2	Lokale Variablen und Konstanten	40
3.2.1	Lokale Variablen	40
3.2.2	Lokale Konstanten	42
3.2.3	Parameter als verkapselte lokale Variablen*	42
3.3	Beispiele: Punkte und Linien	44
3.3.1	Die Klasse Point	44
3.3.2	Die Klasse Line	47
3.3.3	Private Hilfsmethoden	47
3.3.4	Fazit: Methoden sind Funktionen oder Prozeduren	48
4	Programmieren in Java – Eine erste Einführung	51
4.1	Programme schreiben und ausführen	51
4.1.1	Der Programmierprozess	52
4.1.2	Die Hauptklasse und die Methode main	54
4.2	Ein einfaches Beispiel (mit ein bisschen Physik)	55
4.3	Bibliotheken (Packages)	58
4.3.1	Packages: Eine erste Einführung	59
4.3.2	Öffentlich, halböffentlich und privat	59
4.3.3	Standardpackages von JAVA	59
4.3.4	Die Java-Klasse Math	60
4.3.5	Die Klasse Terminal: Einfache Ein-/Ausgabe	62
4.3.6	Kleine Beispiele mit Grafik	63
4.3.7	Zeichnen in JAVA: Elementare Grundbegriffe	66

Teil II Ablaufkontrolle

5	Kontrollstrukturen	71
5.1	Ausdrücke	71
5.2	Elementare Anweisungen und Blöcke	73
5.3	Man muss sich auch entscheiden können	74
5.3.1	Die if-Anweisung	74
5.3.2	Die switch-Anweisung	76
5.4	Immer und immer wieder: Iteration	78
5.4.1	Die while-Schleife	78
5.4.2	Die for-Schleife	80
5.4.3	Die break- und continue-Anweisung	81
5.5	Beispiele: Schleifen und Arrays	83
6	Rekursion	89
6.1	Rekursive Methoden	90
6.2	Funktioniert das wirklich?	92

7	Aspekte der Programmiermethodik	97
7.1	Man muss sein Tun auch erläutern: Dokumentation	97
7.1.1	Kommentare	98
7.2	Zusicherungen (Assertions)	99
7.2.1	Allgemeine Dokumentation	101
7.3	Aufwand	102
7.4	Beispiel: Mittelwert und Standardabweichung	106
7.5	Beispiel: Fläche eines Polygons	107
7.6	Beispiel: Sieb des Eratosthenes	110
7.7	Beispiel: Zinsrechnung	112
8	Suchen und Sortieren	117
8.1	Ordnung ist die halbe Suche	117
8.2	Wer sucht, der findet (oder auch nicht)	118
8.2.1	Lineares Suchen: Die <i>British-Museum Method</i>	118
8.2.2	Suchen mit Bisektion	120
8.3	Wer sortiert, findet schneller	122
8.3.1	<i>Selection sort</i>	125
8.3.2	<i>Insertion sort</i>	126
8.3.3	<i>Quicksort</i>	128
8.3.4	<i>Mergesort</i>	132
8.3.5	<i>Heapsort</i>	134
8.3.6	Mit Mogeln gehts schneller: <i>Bucket sort</i>	140
8.3.7	<i>Verwandte Probleme</i>	140
9	Numerische Algorithmen	141
9.1	Vektoren und Matrizen	141
9.2	Gleichungssysteme: Gauß-Elimination	144
9.2.1	Lösung von Dreieckssystemen	147
9.2.2	<i>LU</i> -Zerlegung	148
9.2.3	Pivot-Elemente	150
9.2.4	Nachiteration	151
9.3	Wurzelberechnung und Nullstellen von Funktionen	152
9.4	Differenzieren	155
9.5	Integrieren	157
9.6	Interpolation	160
9.6.1	Für Geizhalse: Speicherplatz sparen	165
9.6.2	Extrapolation	166
9.7	Lösung einfacher Differenzialgleichungen	169
9.7.1	Einfache Einschrittverfahren	170
9.7.2	Mehrschrittverfahren	171
9.7.3	Extrapolation	172

Teil IV Weitere Konzepte objektorientierter Programmierung

10 Vererbung	177
10.1 Vererbung = Subtyp?	177
10.2 Sub- und Superklassen in JAVA	180
10.2.1 „Mutierte“ Vererbung und dynamische Bindung	181
10.2.2 Was bist du?	183
10.2.3 Ende der Vererbung: <code>Object</code> und <code>final</code>	184
10.2.4 Mit <code>super</code> zur Superklasse	186
10.2.5 Casting: Zurück zur Sub- oder Superklasse	187
10.3 Abstrakte Klassen	188
11 Interfaces	191
11.1 Mehrfachvererbung und Interfaces	191
11.2 Anwendung: Suchen und Sortieren richtig gelöst	195
11.2.1 Das Interface <code>Sortable</code>	196
12 Generizität (Polymorphie)	199
12.1 Des einen Vergangenheit ist des anderen Zukunft	199
12.2 Die Idee der Polymorphie (Generizität)	200
12.3 Generizität in JAVA 1.5	201
13 Und dann war da noch	203
13.1 Einer für alle: <code>static</code>	203
13.2 Initialisierung	206
13.3 Innere und lokale Klassen	206
13.4 Anonyme Klassen	207
13.5 Enumerationstypen in Java 1.5	209
13.6 Anwendung: Methoden höherer Ordnung	209
13.6.1 <code>Fun</code> als Interface	210
13.6.2 Verwendung anonymer Klassen	211
13.6.3 Interpolation als Implementierung von <code>Fun</code>	212
13.7 Ein bisschen Eleganz: Methoden als Resultate	212
14 Namen, Scopes und Packages	215
14.1 Das Prinzip der (Un-)Sichtbarkeit	215
14.2 Gültigkeitsbereich (<i>Scope</i>)	216
14.2.1 Klassen als Gültigkeitsbereich	217
14.2.2 Methoden als Gültigkeitsbereich	218
14.2.3 Blöcke als Gültigkeitsbereich	218
14.2.4 Verschattung (<i>holes in the scope</i>)	219
14.2.5 Überlagerung	220

14.3 Packages: Scopes „im Großen“	220
14.3.1 Volle Klassennamen	222
14.3.2 Import	222
14.4 Geheimniskrämerei	223
14.4.1 Geschlossene Gesellschaft: Package	223
14.4.2 Herstellen von Öffentlichkeit: <code>public</code>	223
14.4.3 Maximale Verschllossenheit: <code>private</code>	224
14.4.4 Vertrauen zu Subklassen: <code>protected</code>	224
14.4.5 Zusammenfassung	225

Teil V Datenstrukturen

15 Referenzen	229
15.1 Nichts währt ewig: Lebensdauern	229
15.2 Referenzen: „Ich weiß, wo mans findet“	231
15.3 Referenzen in JAVA	232
15.3.1 Zur Funktionsweise von Referenzen	232
15.3.2 Referenzen und Methodenaufrufe	235
15.3.3 Wer bin ich?: <code>this</code>	237
15.4 Gleichheit und Kopien	237
15.5 Die Wahrheit über Arrays	239
15.6 Abfallbeseitigung (<i>Garbage collection</i>)	240
16 Listen	243
16.1 Listen als verkettete Objekte	243
16.1.1 Listenzellen	244
16.1.2 Elementares Arbeiten mit Listen	246
16.1.3 Traversieren von Listen	247
16.1.4 Generische Listen	249
16.1.5 Zirkuläre Listen	250
16.1.6 Doppelt verkettete Listen	251
16.1.7 Eine methodische Schwäche und ihre Gefahren	252
16.2 Listen als Abstrakter Datentyp (<code>LinkedList</code>)	253
16.3 Listenartige Strukturen in JAVA	256
16.3.1 <code>Collection</code>	258
16.3.2 <code>List</code>	259
16.3.3 <code>Set</code>	259
16.3.4 <code>LinkedList</code> , <code>ArrayList</code> und <code>Vector</code>	259
16.3.5 <code>Stack</code>	260
16.3.6 <code>Queue</code> („Warteschlange“)	261
16.3.7 <code>Priority Queues</code> : Vordrängeln ist erlaubt	262
16.4 Einer nach dem andern: Iteratoren	263
16.5 Neue <code>for</code> -Schleife in JAVA 1.5	264

17	Bäume	267
17.1	Bäume: Grundbegriffe	267
17.2	Implementierung durch Verkettung	268
17.2.1	Binärbäume	269
17.2.2	Allgemeine Bäume	271
17.2.3	Binärbäume als Abstrakter Datentyp	272
17.3	Traversieren von Bäumen: Baum-Iteratoren	273
17.4	Suchbäume (geordnete Bäume)	276
17.4.1	Suchbäume als Abstrakter Datentyp: <code>SearchTree</code>	278
17.4.2	Implementierung von Suchbäumen	279
17.5	Balancierte Suchbäume	284
17.5.1	2-3-Bäume und 2-3-4-Bäume	286
17.5.2	Rot-Schwarz-Bäume	288
17.6	Baumdarstellung von Sprachen (Syntaxbäume)	293
18	Graphen	299
18.1	Beispiele für Graphen	299
18.2	Grundbegriffe	301
18.3	Adjazenzlisten und Adjazenzmatrizen	302
18.4	Erreichbarkeit und verwandte Aufgaben	304
18.4.1	Konzeptueller Entwurf	305
18.4.2	Klassische Programmierung in Java	306
18.4.3	Eine genuin objektorientierte Sicht von Graphalgorithmen	308
18.4.4	Tiefen- und Breitensuche	309
18.5	Kürzeste Wege (von einem Knoten aus)	311
18.6	Aufspannende Bäume	312
18.7	Transitive Hülle	313
18.8	Weitere Graphalgorithmen	316

Teil VI Programmierung von Software-Systemen

19	Keine Regel ohne Ausnahmen: Exceptions	321
19.1	Manchmal gehts eben schief	321
19.2	Exceptions	323
19.3	Man versucht halt mal: <code>try</code> und <code>catch</code>	325
19.4	Exceptions verkünden: <code>throw</code>	327
19.5	Methoden mit Exceptions: <code>throws</code>	328
20	Ein- und Ausgabe	331
20.1	Ohne Verwaltung geht gar nichts	332
20.1.1	Pfade und Dateinamen in Windows und Unix	333
20.1.2	<code>File</code> : Die Klasse zur Dateiverwaltung	334
20.1.3	Programmieren der Dateiverwaltung	336

20.2	Was man Lesen und Schreiben kann	337
20.3	Dateien mit Direktzugriff („Externe Arrays“)	339
20.4	Sequenzielle Dateien („Externe Listen“, Ströme)	340
20.4.1	Die abstrakte Superklasse <code>InputStream</code>	342
20.4.2	Die konkreten Klassen für Eingabeströme	342
20.4.3	Ausgabeströme	344
20.4.4	Das Ganze nochmals mit Unicode: <code>Reader</code> und <code>Writer</code> ..	345
20.5	Programmieren mit Dateien und Strömen	346
20.6	Terminal-Ein-/Ausgabe	347
20.7	... und noch ganz viel Spezielles	351
20.7.1	Serialisierung	351
20.7.2	Interne Kommunikation über Pipes	352
20.7.3	Konkatenation von Strömen: <code>SequenceInputStream</code> ..	352
20.7.4	Simulierte Ein-/Ausgabe	353
21	Konkurrenz belebt das Geschäft: Threads	355
21.1	Threads: Leichtgewichtige Prozesse	355
21.2	Die Klasse <code>Thread</code>	359
21.2.1	Entstehen – Arbeiten – Sterben	360
21.2.2	Schlafe nur ein Weilchen ... (<code>sleep</code>)	361
21.2.3	Jetzt ist mal ein anderer dran ... (<code>yield</code>)	362
21.2.4	Ich warte auf dein Ende ... (<code>join</code>)	362
21.2.5	Unterbrich mich nicht! (<code>interrupt</code>)	364
21.2.6	Ich bin wichtiger als du! (Prioritäten)	365
21.3	Synchronisation und Kommunikation	366
21.3.1	Vorsicht, es klemmt!	368
21.3.2	Warten Sie, bis Sie aufgerufen werden! (<code>wait</code> , <code>notify</code>)	369
21.4	Das Interface <code>Runnable</code>	372
21.5	Ist das genug?	373
21.5.1	Gemeinsam sind wir stark (Thread-Gruppen)	373
21.5.2	Dämonen sterben heimlich	374
21.5.3	Zu langsam für die reale Zeit?	374
21.5.4	Vorsicht, veraltet!	375
21.5.5	Neues in Java 1.5	375
22	Das ist alles so schön bunt hier: Grafik in JAVA	377
22.1	Historische Vorbemerkung	377
22.1.1	Awt und Swing	378
22.1.2	Entwicklungsumgebungen	379
22.2	Grundlegende Konzepte von GUIs	380

23	GUI: Layout	383
23.1	Die Superklassen: <code>Component</code> und <code>JComponent</code>	385
23.2	Elementare GUI-Elemente	386
23.2.1	Beschriftungen: <code>Label</code> / <code>JLabel</code>	386
23.2.2	Zum Anklicken: <code>Button</code> / <code>JButton</code>	387
23.2.3	Editierbarer Text: <code>TextField</code> / <code>JTextField</code>	389
23.3	Behälter: <code>Container</code>	392
23.3.1	Das Hauptfenster: <code>Frame</code> / <code>JFrame</code>	392
23.3.2	Lokale Container: <code>Panel</code> / <code>JPanel</code>	396
23.3.3	Layout-Manager	397
23.3.4	Statischer Import in Java 1.5	399
23.3.5	Mehr über Farben: <code>Color</code>	400
23.3.6	Fenster-Geometrie: <code>Point</code> und <code>Dimension</code>	402
23.3.7	Größenbestimmung von Fenstern	402
23.4	Selbst Zeichnen	405
23.4.1	Die Methode <code>paint</code>	406
23.4.2	Die Methode <code>paintComponent</code>	407
23.4.3	Wenn man nur zeichnen will	408
23.4.4	Zeichnen mit <code>Graphics</code> und <code>Graphics2D</code>	409
24	Hallo Programm! – Hallo GUI!	411
24.1	Auf GUIs ein- und ausgeben	411
24.2	Von Ereignissen getrieben	412
24.3	Immerzu lauschen	414
24.3.1	Beispiel: Eingabe im Displayfeld	414
24.3.2	Arbeiten mit Buttons	416
24.3.3	Listener-Arten	418
25	Beispiel: Taschenrechner	421
25.1	Taschenrechner: Die globale Struktur	422
25.2	Taschenrechner: <i>Model</i>	423
25.3	Taschenrechner: <i>View</i>	426
25.4	Taschenrechner: <i>Control</i>	433
25.5	Fazit	436

Teil VII Ausblick

26	Es gäbe noch viel zu tun	439
26.1	Java und Netzwerke: Von Sockets bis Jini	439
26.1.1	Die OSI-Hierarchie	440
26.1.2	Sockets	443
26.1.3	Wenn die Methoden weit weg sind: RMI	443
26.1.4	Wie komme ich ins Netz? (Jini)	445
26.2	Java und das Web	445

26.2.1	Applets	445
26.2.2	Servlets (<i>Server Applets</i>)	449
26.2.3	JSP: <i>JavaServer Pages</i>	450
26.2.4	Java und XML	450
26.2.5	Java und Email	451
26.3	Sicher ist sicher: Java-Security	451
26.3.1	Sandbox und Security Manager	452
26.3.2	Verschlüsselung und Signaturen	453
26.4	Reflection und Introspection	453
26.5	Java-Komponenten-Technologie: Beans	454
26.6	Java und Datenbanken: JDBC	457
26.7	Direktzugang zum Rechner: Von JNI bis Realzeit	457
26.7.1	Die Java Virtual Machine (JVM)	457
26.7.2	Das Java Native Interface (JNI)	458
26.7.3	Externe Prozesse starten	459
26.7.4	Java und Realzeit	459
A	Anhang: Praktische Hinweise	461
A.1	Java beschaffen	461
A.2	Java installieren	462
A.3	Java-Programme übersetzen (javac)	463
A.3.1	Verwendung von zusätzlichen Directories	464
A.3.2	Verwendung des Classpath	465
A.3.3	Konflikte zwischen Java 1.4 und Java 1.5	466
A.4	Java-Programme ausführen (java und javaw)	466
A.5	Directories, Classpath und Packages	468
A.6	Java-Archive verwenden (jar)	469
A.7	Dokumentation generieren mit javadoc	471
A.8	Weitere Werkzeuge	473
A.9	Die Klassen Terminal und Pad dieses Buches	473
A.10	Materialien zu diesem Buch	474
	Literaturverzeichnis	475
	Sachverzeichnis	477

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.