

Inhalt

Vorwort	39
1 Java ist auch eine Sprache	55
1.1 Der erste Kontakt	55
1.2 Historischer Hintergrund	55
1.3 Eigenschaften von Java	57
1.3.1 Bytecode und die virtuelle Maschine	57
1.3.2 Objektorientierung in Java	58
1.3.3 Java-Security-Modell	59
1.3.4 Zeiger und Referenzen	59
1.3.5 Bring den Müll raus, Garbage-Collector!	60
1.3.6 Ausnahmebehandlung	60
1.3.7 Kein Präprozessor für Textersetzungen	61
1.3.8 Keine benutzerdefinierten überladenen Operatoren	62
1.3.9 Java als Sprache, Laufzeitumgebung und Bibliothek	62
1.3.10 Wofür sich Java weniger eignet	63
1.3.11 Java im Vergleich zu anderen Sprachen	64
1.3.12 Java ist Open Source	65
1.4 Die Rolle von Java im Web	66
1.4.1 Vollwertige Applikationen statt Applets	66
1.5 Java-Plattformen: Java SE, Java EE und Java ME	66
1.5.1 Java SE-Plattform	67
1.5.2 Java für die Kleinen	68
1.5.3 Java für die Großen	68
1.6 Installation von Suns Java Platform Standard Edition (Java SE)	68
1.6.1 Das Java SE von Sun	69
1.6.2 Download vom JDK	69
1.6.3 Java SE unter Windows installieren	69
1.7 Das erste Programm compilieren und testen	71
1.7.1 Ein Quadratzahlen-Programm	71
1.7.2 Der Compilerlauf	72
1.7.3 Die Laufzeitumgebung	74
1.7.4 Häufige Compiler- und Interpreterprobleme	74
1.8 Entwicklungsumgebungen im Allgemeinen	75
1.8.1 Die Entwicklungsumgebung Eclipse	75
1.8.2 NetBeans von Sun	75
1.8.3 Ein Wort zu Microsoft, Java und zu J++	76
1.9 Eclipse im Speziellen	77
1.9.1 Eclipse starten	77

1.9.2	Das erste Projekt anlegen	78
1.9.3	Eine Klasse hinzufügen	80
1.9.4	Übersetzen und Ausführen	81
1.9.5	JDK statt JRE	82
1.9.6	Start eines Programms ohne Speicheraufforderung	82
1.9.7	Projekt einfügen, Workspace für die Programme wechseln	83
1.9.8	Plugins für Eclipse	84
1.10	Zum Weiterlesen	84

2 Sprachbeschreibung 87

2.1	Elemente der Programmiersprache Java	87
2.1.1	Token	87
2.1.2	Textkodierung durch Unicode-Zeichen	88
2.1.3	Literale	90
2.1.4	Bezeichner	90
2.1.5	Reservierte Schlüsselwörter	92
2.1.6	Kommentare	93
2.1.7	Die API-Dokumentation	94
2.2	Anweisungen formen Programme	95
2.2.1	Anweisungen	95
2.2.2	Eine Klasse bildet den Rahmen	96
2.2.3	Die Reise beginnt am main()	97
2.2.4	Funktionsaufrufe als Anweisungen	98
2.2.5	print(), println() und printf() für Bildschirmausgaben	98
2.2.6	Ausdrucksanweisung	100
2.2.7	Erste Idee der Objektorientierung	100
2.2.8	Modifizierer	101
2.2.9	Anweisungen und Blöcke	102
2.3	Datentypen, Typisierung, Variablen und Zuweisungen	102
2.3.1	Primitive Datentypen im Überblick	103
2.3.2	Variablendeklarationen	105
2.3.3	Zuweisungsoperator	106
2.3.4	Variablendeklaration mit Wertinitialisierung	106
2.3.5	Wahrheitswerte	108
2.3.6	Ganzzahlige Datentypen	108
2.3.7	Die Fließkommazahlen float und double	110
2.3.8	Alphanumerische Zeichen	111
2.3.9	Gute Namen, schlechte Namen	112
2.4	Blöcke, Initialisierung und Sichtbarkeit	113
2.4.1	Blöcke und Anweisungen	113
2.4.2	Initialisierung von lokalen Variablen	113
2.4.3	Sichtbarkeit und Gültigkeitsbereich	114

2.5	Ausdrücke, Operanden und Operatoren	115
2.5.1	Arithmetische Operatoren	116
2.5.2	Unäres Minus und Plus	119
2.5.3	Zuweisung mit Operation	120
2.5.4	Präfix- oder Postfix-Inkrement und -Dekrement	120
2.5.5	Die relationalen Operatoren und die Gleichheitsoperatoren	122
2.5.6	Logische Operatoren Und, Oder, Xor, Nicht	123
2.5.7	Rang der Operatoren in der Auswertungsreihenfolge	124
2.5.8	Die Typanpassung (das Casting)	126
2.5.9	Überladenes Plus für Strings	132
2.6	Bedingte Anweisungen oder Fallunterscheidungen	133
2.6.1	Die if-Anweisung	133
2.6.2	Die Alternative mit einer if/else-Anweisung wählen	135
2.6.3	Die switch-Anweisung bietet die Alternative	137
2.7	Schleifen	140
2.7.1	Die while-Schleife	141
2.7.2	Die do-while-Schleife	142
2.7.3	Die for-Schleife	143
2.7.4	Schleifenbedingungen und Vergleiche mit ==	146
2.7.5	Ausbruch planen mit break und Wiedereinstieg mit continue	148
2.7.6	break und continue mit Sprungmarken	151
2.8	Methoden einer Klasse	152
2.8.1	Bestandteil einer Funktion	153
2.8.2	Signatur-Beschreibung in der Java-API	153
2.8.3	Aufruf einer Methode	155
2.8.4	Methoden ohne Parameter deklarieren	155
2.8.5	Statische Methoden (Klassenmethoden)	156
2.8.6	Parameter, Argument und Wertübergabe	157
2.8.7	Methoden vorzeitig mit return beenden	159
2.8.8	Nicht erreichbarer Quellcode bei Funktionen	159
2.8.9	Rückgabewerte	160
2.8.10	Methoden überladen	163
2.8.11	Vorgegebener Wert für nicht aufgeführte Argumente	165
2.8.12	Finale lokale Variablen	165
2.8.13	Rekursive Methoden	167
2.9	Weitere Operatoren	169
2.9.1	Bits und Bytes	169
2.9.2	Operationen auf Bit-Ebene	170
2.9.3	Die Verschiebeoperatoren	172
2.9.4	Ein Bit setzen, löschen, umdrehen und testen	174
2.9.5	Bit-Funktionen der Integer- und Long-Klasse	175
2.9.6	Der Bedingungsoperator	176

2.9.7	Operator vermisst	177
2.10	Einfache Benutzereingaben	178
2.11	Zum Weiterlesen	179

3 Klassen und Objekte 181

3.1	Objektorientierte Programmierung	181
3.1.1	Warum überhaupt OOP?	181
3.1.2	Denk ich an Java, denk ich an Wiederverwendbarkeit	182
3.2	Eigenschaften einer Klasse	183
3.2.1	Die Klasse Point	183
3.3	Die UML (Unified Modeling Language)	184
3.3.1	Hintergrund und Geschichte zur UML	184
3.3.2	Wichtige Diagrammtypen der UML	185
3.3.3	UML-Werkzeuge	186
3.4	Neue Objekte erzeugen	187
3.4.1	Anlegen eines Exemplars einer Klasse mit dem new-Operator	187
3.4.2	Deklariieren von Referenzvariablen	189
3.4.3	Zugriff auf Variablen und Methoden mit dem ».«	189
3.4.4	Konstruktoren nutzen	192
3.5	Pakete und import-Deklarationen nutzen	193
3.5.1	Volle Qualifizierung und import-Deklaration	193
3.5.2	import *	194
3.6	Mit Referenzen arbeiten	195
3.6.1	Zuweisungen bei Referenzen	195
3.6.2	Methoden mit nicht-primitiven Parametern	196
3.7	Identität und Gleichheit	197
3.7.1	Identität von Objekten	197
3.7.2	Gleichheit und die Methode equals()	198
3.7.3	Die null-Referenz	200
3.8	Wrapper-Klassen und Autoboxing	201
3.8.1	Erzeugen von Wrapper-Objekten	202
3.8.2	Konvertierungen in eine String-Repräsentation	203
3.8.3	Die Klasse Integer	204
3.8.4	Die Klassen Double und Float für Fließkommazahlen	205
3.8.5	Die Basisklasse Number für numerische Wrapper-Objekte	206
3.8.6	Die Boolean-Klasse	207
3.8.7	Autoboxing: Boxing und Unboxing	208
3.9	Compilationseinheiten und eigene Pakete schnüren	211
3.9.1	Die package-Anweisung	211
3.9.2	Importieren von Klassen mit import	211
3.9.3	Hierarchische Strukturen und das Default-Package	212
3.9.4	Paketnamen	213

3.9.5	Klassen mit gleichen Namen in unterschiedlichen Paketen	213
3.9.6	Compilationseinheit (Compilation Unit)	213
3.9.7	Statischer Import	213
3.9.8	Eine Verzeichnisstruktur für eigene Projekte	214
3.10	Arrays	215
3.10.1	Deklaration von Arrays	215
3.10.2	Arrays mit Inhalt	216
3.10.3	Die Länge eines Arrays über das Attribut length	216
3.10.4	Zugriff auf die Elemente über den Index	216
3.10.5	Array-Objekte mit new erzeugen	217
3.10.6	Fehler bei Arrays	218
3.10.7	Die erweiterte for-Schleife	219
3.10.8	Arrays mit nicht-primitiven Elementen	221
3.10.9	Mehrdimensionale Arrays	222
3.10.10	Vorinitialisierte Arrays	224
3.10.11	Mehrere Rückgabewerte	226
3.10.12	Methode mit variabler Argumentanzahl (Vararg)	226
3.10.13	Klonen kann sich lohnen – Arrays vermehren	228
3.10.14	Feldinhalte kopieren	228
3.10.15	Die Klasse Arrays zum Vergleichen, Füllen und Suchen	229
3.11	Der Einstiegspunkt für das Laufzeitsystem: main()	235
3.11.1	Kommandozeilen-Argumente verarbeiten	235
3.11.2	Der Rückgabewert von main() und System.exit()	236
3.12	Zum Weiterlesen	236

4 Der Umgang mit Zeichenketten 237

4.1	Einzelne Zeichen mit der Character-Klasse behandeln	237
4.2	Strings und deren Anwendung	238
4.2.1	String-Literale als String-Objekte für konstante Zeichenketten ...	242
4.2.2	String-Länge und Test auf Leerstring	242
4.2.3	Nach enthaltenen Zeichen und Zeichenfolgen suchen	243
4.2.4	Gut, dass wir verglichen haben	245
4.2.5	String-Teile extrahieren	247
4.2.6	Strings anhängen, Groß-/Kleinschreibung und Leerraum	249
4.2.7	Suchen und ersetzen	251
4.2.8	String-Objekte mit Konstruktoren neu anlegen	253
4.3	Konvertieren zwischen Primitiven und Strings	256
4.3.1	Unterschiedliche Typen in String-Repräsentationen konvertieren	256
4.3.2	String in primitives Element konvertieren	257
4.4	Veränderbare Zeichenketten mit StringBuilder/StringBuffer	259
4.4.1	Anlegen von StringBuilder-/StringBuffer-Objekten	259
4.4.2	Die Länge eines StringBuilder-/StringBuffer-Objekts	260

4.4.3	Daten anhängen	261
4.4.4	Zeichen(folgen) setzen, erfragen, löschen und umdrehen	262
4.4.5	Vergleichen von String/StringBuilder/StringBuffer	263
4.4.6	hashCode() bei StringBuffer/StringBuilder	264
4.5	Sprachabhängiges Vergleichen und Normalisierung	264
4.5.1	Die Klasse Collator	264
4.5.2	Effiziente interne Speicherung für die Sortierung	267
4.5.3	Normalisierung	269
4.6	Reguläre Ausdrücke	269
4.6.1	Arbeiten mit der Fassade: String.matches()	269
4.6.2	Die Klassen Pattern und Matcher	271
4.6.3	Quantifizierer und Wiederholungen	273
4.6.4	Finden und nicht matchen	273
4.6.5	Gierige und nicht gierige Operatoren	274
4.6.6	Mit MatchResult alle Ergebnisse einsammeln	275
4.7	Zerlegen von Zeichenketten	276
4.7.1	Splitten von Zeichenketten mit split()	277
4.7.2	Die Klasse Scanner	278
4.7.3	StringTokenizer	283
4.7.4	BreakIterator als Zeichen-, Wort-, Zeilen- und Satztrenner	285
4.8	Zeichenkodierungen und Base64	288
4.8.1	Über die Klasse String Kodierungen vornehmen	288
4.8.2	Konvertieren mit OutputStreamWriter-Klassen	289
4.8.3	Das Paket java.nio.charset	289
4.8.4	Base64-Kodierung	290
4.9	Formatieren von Ausgaben	291
4.9.1	Formatieren mit format() aus String	291
4.9.2	Die Format-Klassen im Überblick	294
4.9.3	Zahlen, Prozente und Währungen mit NumberFormat und DecimalFormat formatieren	295
4.10	Zum Weiterlesen	297

5 Mathematisches **299**

5.1	Repräsentation ganzer Zahlen – das Zweierkomplement	299
5.2	Fließkommaarithmetik in Java	300
5.2.1	Mantisse und Exponent	300
5.2.2	Spezialwerte Unendlich, Null, NaN	301
5.3	Wertebereich eines Typs und Überlaufkontrolle	303
5.3.1	Behandlung des Überlaufs	304
5.4	Die Eigenschaften der Klasse Math	305
5.4.1	Attribute	307
5.4.2	Absolutwerte und Maximum/Minimum	307

5.4.3	Winkelfunktionen	308
5.4.4	Runden von Werten	309
5.4.5	Wurzel und Exponentialfunktionen	311
5.4.6	Der Logarithmus	311
5.4.7	Rest der ganzzahligen Division	312
5.4.8	Zufallszahlen	313
5.5	Mathe bitte strikt	313
5.5.1	Strikt Fließkomma mit strictfp	314
5.5.2	Die Klassen Math und StrictMath	314
5.6	Die Random-Klasse	314
5.6.1	Objekte aufbauen und der Seed	315
5.6.2	Zufallszahlen erzeugen	315
5.6.3	Pseudo-Zufallszahlen in der Normalverteilung	316
5.7	Große Zahlen	316
5.7.1	Die Klasse BigInteger	316
5.7.2	Methoden von BigInteger	319
5.7.3	Ganz lange Fakultäten	321
5.7.4	Große Fließkommazahlen mit BigDecimal	322
5.7.5	Mit MathContext komfortabel die Rechengenauigkeit setzen	324
5.8	Zum Weiterlesen	325

6 Eigene Klassen schreiben 327

6.1	Eigene Klassen mit Eigenschaften deklarieren	327
6.1.1	Attribute deklarieren	327
6.1.2	Methoden deklarieren	328
6.1.3	Die this-Referenz	332
6.2	Privatsphäre und Sichtbarkeit	335
6.2.1	Für die Öffentlichkeit: public	335
6.2.2	Kein Public Viewing – Passwörter sind privat	336
6.2.3	Wieso nicht freie Methoden und Variablen für alle?	337
6.2.4	Privat ist nicht ganz privat: Es kommt darauf an, wer's sieht	337
6.2.5	Zugriffsmethoden für Attribute deklarieren	338
6.2.6	Setter und Getter nach der JavaBeans-Spezifikation	339
6.2.7	Paketsichtbar	341
6.2.8	Zusammenfassung zur Sichtbarkeit	342
6.3	Statische Methoden und statische Attribute	344
6.3.1	Warum statische Eigenschaften sinnvoll sind	344
6.3.2	Statische Eigenschaften mit static	345
6.3.3	Statische Eigenschaften über Referenzen nutzen?	346
6.3.4	Warum die Groß- und Kleinschreibung wichtig ist	346
6.3.5	Statische Variablen zum Datenaustausch	347
6.3.6	Statische Eigenschaften und Objekteigenschaften	348

6.4	Konstanten und Aufzählungen	349
6.4.1	Konstanten über öffentliche statische finale Variablen	349
6.4.2	Eincompilierte Belegungen der Klassenvariablen	350
6.4.3	Typ(un)sichere Aufzählungen	351
6.4.4	Aufzählungen mit enum	352
6.5	Objekte anlegen und zerstören	355
6.5.1	Konstruktoren schreiben	355
6.5.2	Der Default-Konstruktor	356
6.5.3	Parametrisierte und überladene Konstruktoren	357
6.5.4	Copy-Konstruktor	359
6.5.5	Einen anderen Konstruktor der gleichen Klasse aufrufen	360
6.5.6	Ihr fehlt uns nicht – der Garbage-Collector	362
6.5.7	Private Konstruktoren, Utility-Klassen, Singleton, Fabriken	364
6.6	Klassen- und Objektinitialisierung	366
6.6.1	Initialisierung von Objektvariablen	366
6.6.2	Statische Blöcke als Klasseninitialisierer	368
6.6.3	Initialisierung von Klassenvariablen	368
6.6.4	Exemplarinitialisierer (Instanzinitialisierer)	369
6.6.5	Finale Werte im Konstruktor und in statischen Blöcken setzen ...	371
6.7	Assoziationen zwischen Objekten	373
6.7.1	Unidirektionale 1:1-Beziehung	373
6.7.2	Bidirektionale 1:1-Beziehungen	374
6.7.3	Unidirektionale 1:n-Beziehung	375
6.8	Vererbung	377
6.8.1	Vererbung in Java	378
6.8.2	Spielobjekte modelliert	378
6.8.3	Implizite Basisklasse java.lang.Object	380
6.8.4	Einfach- und Mehrfachvererbung	380
6.8.5	Sichtbarkeit protected	380
6.8.6	Konstruktoren in der Vererbung und super	381
6.9	Typen in Hierarchien	385
6.9.1	Automatische und explizite Typanpassung	385
6.9.2	Das Substitutionsprinzip	387
6.9.3	Typen mit dem binären Operator instanceof testen	389
6.10	Methoden überschreiben	390
6.10.1	Methoden in Unterklassen mit neuem Verhalten ausstatten	390
6.10.2	Mit super an die Eltern	393
6.10.3	Finale Klassen und finale Methoden	395
6.10.4	Kovariante Rückgabetypen	396
6.10.5	Array-Typen und Kovarianz	397
6.11	Dynamisches Binden/Polymorphie	398
6.11.1	Unpolymorph bei privaten, statischen und finalen Methoden	401
6.11.2	Polymorphie bei Konstruktoraufrufen	403

6.12	Abstrakte Klassen und abstrakte Methoden	404
6.12.1	Abstrakte Klassen	405
6.12.2	Abstrakte Methoden	406
6.13	Schnittstellen	409
6.13.1	Deklarieren von Schnittstellen	409
6.13.2	Implementieren von Schnittstellen	410
6.13.3	Markierungsschnittstellen	411
6.13.4	Ein Polymorphie-Beispiel mit Schnittstellen	412
6.13.5	Die Mehrfachvererbung bei Schnittstellen	413
6.13.6	Keine Kollisionsgefahr bei Mehrfachvererbung	416
6.13.7	Erweitern von Interfaces – Subinterfaces	417
6.13.8	Vererbte Konstanten bei Schnittstellen	418
6.13.9	Abstrakte Klassen und Schnittstellen im Vergleich	419
6.14	Dokumentationskommentare mit JavaDoc	420
6.14.1	Einen Dokumentationskommentar setzen	421
6.14.2	Mit javadoc eine Dokumentation erstellen	423
6.14.3	HTML-Tags in Dokumentationskommentaren	423
6.14.4	Generierte Dateien	423
6.14.5	Dokumentationskommentare im Überblick	424
6.14.6	JavaDoc und Doclets	425
6.14.7	Veraltete (deprecated) Typen und Eigenschaften	426

7 Angewandte Objektorientierung 429

7.1	Schnittstellen in der Anwendung	429
7.1.1	CharSequence als Beispiel einer Schnittstelle	429
7.1.2	Die Schnittstelle Iterable	431
7.1.3	Funktionszeiger	433
7.2	Design-Pattern (Entwurfsmuster)	435
7.2.1	Motivation für Design-Pattern	435
7.2.2	Das Beobachter-Pattern (Observer/Observable)	436
7.2.3	Ereignisse über Listener	439
7.3	Service-Factory	443
7.3.1	Arbeiten mit dem ServiceLoader	443
7.3.2	Utility-Klasse Lookup als ServiceLoader-Fassade	444
7.4	JavaBean	445
7.4.1	Properties (Eigenschaften)	446
7.4.2	Einfache Eigenschaften	446
7.4.3	Indizierte Eigenschaften	447
7.4.4	Gebundene Eigenschaften	447
7.4.5	Veto-Eigenschaften – dagegen!	450

8	Exceptions	455
8.1	Problembereiche einzäunen	455
8.1.1	Exceptions in Java mit try und catch	455
8.1.2	Eine Datei mit RandomAccessFile auslesen	456
8.1.3	Ablauf einer Ausnahmesituation	458
8.1.4	Wiederholung abgebrochener Bereiche	458
8.1.5	throws im Methodenkopf angeben	459
8.1.6	Abschlussbehandlung mit finally	462
8.1.7	Nicht erreichbare catch-Klauseln	465
8.2	Die Klassenhierarchie der Fehler	466
8.2.1	Die Exception-Hierarchie	466
8.2.2	Oberausnahmen auffangen	467
8.2.3	Alles geht als Exception durch	468
8.2.4	RuntimeException muss nicht aufgefangen werden	469
8.2.5	Harte Fehler: Error	470
8.3	Auslösen eigener Exceptions	471
8.3.1	Mit throw Ausnahmen auslösen	471
8.3.2	Neue Exception-Klassen deklarieren	473
8.3.3	Abfangen und Weiterleiten	474
8.3.4	Geschachtelte Ausnahmen	475
8.4	Rückgabewerte bei ausgelösten Ausnahmen	476
8.5	Der Stack-Trace	476
8.5.1	Stack-Trace erfragen	477
8.6	Assertions	478
8.6.1	Assertions in eigenen Programmen nutzen	479
8.6.2	Assertions aktivieren	480
9	Generics, innere Klassen	481
9.1	Generische Datentypen	481
9.1.1	Einfache Klassenschablonen	482
9.1.2	Einfache Methodenschablonen	483
9.1.3	Umsetzen der Generics, Typlöschung und Raw-Types	484
9.1.4	Generics und Arrays	486
9.1.5	Einschränken der Typen	487
9.1.6	Generics und Vererbung, Invarianz	488
9.1.7	Wildcards	489
9.2	Geschachtelte (innere) Klassen, Schnittstellen, Aufzählungen	490
9.2.1	Statische innere Klassen und Schnittstellen	491
9.2.2	Mitglieds- oder Elementklassen	492
9.2.3	Lokale Klassen	496

9.2.4	Anonyme innere Klassen	496
9.2.5	this und Vererbung	499

10 Die Klassenbibliothek 501

10.1	Die Java-Klassenphilosophie	501
10.1.1	Übersicht über die Pakete der Standardbibliothek	501
10.2	Object ist die Mutter aller Klassen	508
10.2.1	Klassenobjekte	508
10.2.2	Objektidentifikation mit toString()	509
10.2.3	Objektgleichheit mit equals() und Identität	510
10.2.4	Klonen eines Objekts mit clone()	514
10.2.5	Hashcodes über hashCode() liefern	517
10.2.6	Aufräumen mit finalize()	520
10.2.7	Synchronisation	522
10.3	Die Spezial-Oberklasse Enum	522
10.3.1	Methoden auf Enum-Objekten	523
10.3.2	enum mit eigenen Konstruktoren und Methoden	525
10.4	Klassenlader (Class Loader)	528
10.4.1	Woher die kleinen Klassen kommen	528
10.4.2	Setzen des Klassenpfades	529
10.4.3	Die wichtigsten drei Typen von Klassenladern	530
10.4.4	Der java.lang.ClassLoader	531
10.4.5	Hot Deployment mit dem URL-Classloader	532
10.4.6	Das jre/lib/endorsed-Verzeichnis	535
10.5	Die Utility-Klasse System und Properties	535
10.5.1	Systemeigenschaften der Java-Umgebung	536
10.5.2	line.separator	537
10.5.3	Property von der Konsole aus setzen	538
10.5.4	Umgebungsvariablen des Betriebssystems	539
10.5.5	Einfache Zeitmessung und Profiling	540
10.6	Ausführen externer Programme und Skripte	540
10.6.1	ProcessBuilder und Prozesskontrolle mit Process	541
10.6.2	Einen Browser/E-Mail-Client/Editor aufrufen	544
10.6.3	Ausführen von Skripten	545
10.7	Benutzereinstellungen	547
10.7.1	Benutzereinstellungen speichern	547
10.7.2	Einträge einfügen, auslesen und löschen	548
10.7.3	Auslesen der Daten und Schreiben in anderem Format	550
10.7.4	Auf Ereignisse horchen	550
10.7.5	Zugriff auf die gesamte Windows-Registry	551
10.8	Musik abspielen	552
10.8.1	Die Arbeit mit AudioClip	552

10.8.2	Java Sound API	552
10.9	Annotationen	554
10.9.1	Annotationstypen @Override, @Deprecated, @SuppressWarnings	554
10.9.2	Annotationen für Web-Services	556
10.10	Zum Weiterlesen	556

11 Threads und nebenläufige Programmierung 557

11.1	Nebenläufigkeit	557
11.1.1	Threads und Prozesse	557
11.1.2	Wie parallele Programme die Geschwindigkeit steigern können .	559
11.1.3	Was Java für Nebenläufigkeit alles bietet	560
11.2	Threads erzeugen	561
11.2.1	Threads über die Schnittstelle Runnable implementieren	561
11.2.2	Thread mit Runnable starten	562
11.2.3	Der Name eines Threads	563
11.2.4	Die Klasse Thread erweitern	563
11.2.5	Wer bin ich?	566
11.3	Die Zustände eines Threads	566
11.3.1	Threads schlafen	567
11.3.2	Mit yield() auf Rechenzeit verzichten	569
11.3.3	Das Ende eines Threads	569
11.3.4	UncaughtExceptionHandler für unbehandelte Ausnahmen	570
11.3.5	Einen Thread höflich mit Interrupt beenden	571
11.3.6	Der stop() von außen und die Rettung mit ThreadDeath	573
11.3.7	Ein Rendezvous mit join()	574
11.3.8	Barrier und Austausch mit Exchanger	576
11.3.9	Arbeit niederlegen und wieder aufnehmen	577
11.3.10	Priorität	577
11.3.11	Der Thread als Dämon	578
11.4	Der Ausführer (Executor) kommt	580
11.4.1	Die Schnittstelle Executor	580
11.4.2	Die Thread-Pools	581
11.4.3	Threads mit Rückgabe über Callable	583
11.4.4	Mehrere Callable abarbeiten	585
11.4.5	Mit ScheduledExecutorService wiederholende Ausgaben und Zeitsteuerungen	586
11.5	Synchronisation über kritische Abschnitte	586
11.5.1	Gemeinsam genutzte Daten	586
11.5.2	Probleme beim gemeinsamen Zugriff und kritische Abschnitte ...	587
11.5.3	Punkte parallel initialisieren	588
11.5.4	i++ sieht atomar aus, ist es aber nicht	589

11.5.5	Kritische Abschnitte schützen	590
11.5.6	Schützen mit ReentrantLock	591
11.5.7	Synchronisieren mit synchronized	596
11.5.8	Synchronized-Methoden der Klasse StringBuffer	597
11.5.9	Mit synchronized synchronisierte Blöcke	598
11.5.10	Dann machen wir doch gleich alles synchronisiert!	599
11.5.11	Lock-Freigabe im Fall von Exceptions	600
11.5.12	Mit synchronized nachträglich synchronisieren	600
11.5.13	Monitore sind reentrant – gut für die Geschwindigkeit	601
11.5.14	Synchronisierte Methodenaufrufe zusammenfassen	602
11.5.15	Deadlocks	603
11.6	Synchronisation über Warten und Benachrichtigen	605
11.6.1	Die Schnittstelle Condition	606
11.6.2	It's Disco-Time	609
11.6.3	Warten mit wait() und Aufwecken mit notify()	613
11.6.4	Falls der Lock fehlt: IllegalMonitorStateException	615
11.6.5	Semaphor	616
11.7	Atomare Operationen und frische Werte mit volatile	619
11.7.1	Der Modifizierer volatile bei Objekt-/Klassenvariablen	619
11.7.2	Das Paket java.util.concurrent.atomic	620
11.8	Mit dem Thread verbundene Variablen	621
11.8.1	ThreadLocal	622
11.8.2	InheritableThreadLocal	624
11.9	Gruppen von Threads in einer Thread-Gruppe	625
11.9.1	Aktive Threads in der Umgebung	626
11.9.2	Etwas über die aktuelle Thread-Gruppe herausfinden	626
11.9.3	Threads in einer Thread-Gruppe anlegen	628
11.9.4	Methoden von Thread und ThreadGroup im Vergleich	631
11.10	Zeitgesteuerte Abläufe	632
11.10.1	Die Klassen Timer und TimerTask	633
11.10.2	Job-Scheduler Quartz	634
11.11	Einen Abbruch der virtuellen Maschine erkennen	635
11.12	Zum Weiterlesen	636

12 Datenstrukturen und Algorithmen 637

12.1	Datenstrukturen und die Collection-API	637
12.1.1	Designprinzip mit Schnittstellen, abstrakten und konkreten Klassen	638
12.1.2	Die Basis-Schnittstellen Collection und Map	639
12.1.3	Das erste Programm mit Container-Klassen	639
12.1.4	Die Schnittstelle Collection	640
12.1.5	Schnittstellen, die Collection erweitern, und Map	642

12.1.6	Konkrete Container-Klassen	644
12.1.7	Welche Klasse nehmen?	645
12.1.8	Generische Datentypen in der Collection-API	645
12.1.9	Die Schnittstelle Iterable und das erweiterte for	646
12.2	Mit einem Iterator durch die Daten wandern	647
12.2.1	Die Schnittstellen Enumeration und Iterator	647
12.2.2	Iteratoren von Sammlungen und das erweiterte for	649
12.2.3	Fail-Fast Iterator und die ConcurrentModificationException	651
12.3	Listen	651
12.3.1	Auswahlkriterium ArrayList oder LinkedList	652
12.3.2	Die Schnittstelle List	652
12.3.3	ArrayList	659
12.3.4	LinkedList	661
12.3.5	Der Feld-Adapter Arrays.asList()	662
12.3.6	toArray() von Collection verstehen – die Gefahr einer Falle erkennen	663
12.3.7	Primitive Elemente in den Collection-Datenstrukturen	666
12.4	Vergleichen von Objekten	666
12.4.1	Die Schnittstellen Comparator und Comparable	666
12.4.2	Algorithmen mit Such- und Sortiermöglichkeiten	669
12.4.3	Den größten und kleinsten Wert einer Collection finden	669
12.4.4	Sortieren	671
12.5	Mengen (Sets)	674
12.5.1	HashSet	676
12.5.2	TreeSet – die Menge durch Bäume	677
12.5.3	LinkedHashSet	680
12.6	Stack (Kellerspeicher, Stapel)	680
12.6.1	Die Methoden von Stack	680
12.6.2	Ein Stack ist ein Vector – aha!	681
12.7	Queues (Schlangen) und Deques	682
12.7.1	Die Schnittstelle Queue	682
12.7.2	Blockierende Queues und Prioritätswarteschlangen	683
12.7.3	Deque-Klassen	684
12.8	Assoziative Speicher	684
12.8.1	Die Klassen HashMap und TreeMap	684
12.8.2	Einfügen und Abfragen der Datenstruktur	686
12.8.3	equals(), hashCode() und IdentityHashMap	689
12.8.4	Das Problem von veränderbaren Elementen	691
12.8.5	Aufzählungen und Sichten auf den Assoziativspeicher	691
12.8.6	Der Gleichheitstest, Hash-Wert und Klon einer Hash-Tabelle	694
12.8.7	Die Arbeitsweise einer Hash-Tabelle	694
12.8.8	Multi-Maps	697

12.9	Die Properties-Klasse	697
12.9.1	Properties setzen und lesen	697
12.9.2	Properties verketteten	697
12.9.3	Eigenschaften ausgeben	699
12.9.4	Hierarchische Eigenschaften	699
12.9.5	Properties speichern	699
12.9.6	Klassenbeziehungen: Properties und Hashtable	701
12.10	Algorithmen in Collections	701
12.10.1	Ersetzen, Kopieren, Füllen, Umdrehen, Rotieren, Durchmischen	702
12.10.2	Mit der Halbierungssuche nach Elementen fahnden	703
12.10.3	Nicht-änderbare Datenstrukturen	705
12.10.4	Häufigkeit eines Elements	705
12.10.5	nCopies()	705
12.10.6	Singletons	706
12.11	Synchronisation der Datenstrukturen	707
12.11.1	Lock-free-Algorithmen aus java.util.concurrent	707
12.11.2	Wrapper zur Synchronisation	708
12.11.3	CopyOnWriteArrayList und CopyOnWriteArraySet	709
12.12	Die Klasse BitSet für Bitmengen	709
12.12.1	Ein BitSet anlegen, füllen und erfragen	709
12.12.2	Mengenorientierte Operationen	710
12.12.3	Methodenübersicht	711
12.12.4	Primzahlen in einem BitSet verwalten	712

13 Raum und Zeit 713

13.1	Weltzeit	713
13.2	Wichtige Datum-Klassen im Überblick	714
13.3	Sprachen der Länder	714
13.3.1	Sprachen und Regionen über Locale-Objekte	715
13.4	Internationalisierung und Lokalisierung	718
13.4.1	ResourceBundle-Objekte und Ressource-Dateien	719
13.4.2	Ressource-Dateien zur Lokalisierung	719
13.4.3	Die Klasse ResourceBundle	719
13.4.4	Ladestrategie für ResourceBundle-Objekte	720
13.5	Die Klasse Date	722
13.5.1	Objekte erzeugen und Methoden nutzen	722
13.5.2	Date-Objekte nicht immutable	723
13.6	Calendar und GregorianCalendar	724
13.6.1	Die abstrakte Klasse Calendar	724
13.6.2	Der gregorianische Kalender	726
13.6.3	Ostertage	728
13.6.4	Abfragen und Setzen von Datumselementen	729

13.7	Formatieren und Parsen von Datumsangaben	735
13.7.1	Ausgaben mit printf()	735
13.7.2	Mit DateFormat und SimpleDateFormat formatieren	735
13.7.3	Parsen von Datumswerten	740
13.8	Zum Weiterlesen	742

14 Dateien und Datenströme 743

14.1	Datei und Verzeichnis	744
14.1.1	Dateien und Verzeichnisse mit der Klasse File	744
14.1.2	Verzeichnis oder Datei? Existiert es?	746
14.1.3	Verzeichnis- und Dateieigenschaften/-attribute	747
14.1.4	Wurzelverzeichnis, Laufwerksnamen, Plattenspeicher	749
14.1.5	Umbenennen und Verzeichnisse anlegen	752
14.1.6	Verzeichnisse listen und Dateien filtern	752
14.1.7	Dateien berühren, neue Dateien anlegen, temporäre Dateien	755
14.1.8	Dateien und Verzeichnisse löschen	756
14.1.9	Verzeichnisse nach Dateien iterativ durchsuchen	758
14.1.10	URL- und URI-Objekte aus einem File-Objekt ableiten	759
14.1.11	Mit Locking Dateien sperren	759
14.1.12	Sicherheitsprüfung	760
14.2	Dateien mit wahlfreiem Zugriff	761
14.2.1	Ein RandomAccessFile zum Lesen und Schreiben öffnen	762
14.2.2	Aus dem RandomAccessFile lesen	762
14.2.3	Schreiben mit RandomAccessFile	764
14.2.4	Die Länge des RandomAccessFile	765
14.2.5	Hin und her in der Datei	765
14.3	Stream-Klassen und Reader/Writer am Beispiel von Dateien	766
14.3.1	Mit dem FileWriter Texte in Dateien schreiben	767
14.3.2	Zeichen mit der Klasse FileReader lesen	768
14.3.3	Kopieren mit FileOutputStream und FileInputStream	769
14.3.4	Das FileDescriptor-Objekt	772
14.4	Basisklassen für die Ein-/Ausgabe	773
14.4.1	Die abstrakten Basisklassen	773
14.4.2	Übersicht über Ein-/Ausgabeklassen	773
14.4.3	Die abstrakte Basisklasse OutputStream	775
14.4.4	Die Schnittstellen Closeable und Flushable	776
14.4.5	Ein Datenschlucker	777
14.4.6	Die abstrakte Basisklasse InputStream	777
14.4.7	Ressourcen aus dem Klassenpfad und aus Jar-Archiven laden	778
14.4.8	Ströme mit SequenceInputStream zusammensetzen	779
14.4.9	Die abstrakte Basisklasse Writer	781
14.4.10	Die Schnittstelle Appendable	782

14.4.11	Die abstrakte Basisklasse Reader	783
14.5	Formatierte Textausgaben	785
14.5.1	Die Klassen PrintWriter und PrintStream	785
14.5.2	System.out, System.err und System.in	790
14.5.3	Geschützte Passwort-Eingaben mit der Klasse Console	792
14.6	Schreiben und Lesen aus Strings und Byte-Feldern	792
14.6.1	Mit dem StringWriter ein String-Objekt füllen	793
14.6.2	CharArrayWriter	794
14.6.3	StringReader und CharArrayReader	795
14.6.4	Mit ByteArrayOutputStream in ein Byte-Feld schreiben	796
14.6.5	Mit ByteArrayInputStream aus einem Byte-Feld lesen	797
14.7	Datenströme filtern und verketteten	797
14.7.1	Streams als Filter verketteten	798
14.7.2	Gepufferte Ausgaben mit BufferedWriter/BufferedOutputStream	798
14.7.3	Gepufferte Eingaben mit BufferedReader/BufferedReader	800
14.7.4	LineNumberReader zählt automatisch Zeilen mit	802
14.7.5	Daten mit der Klasse PushbackReader zurücklegen	803
14.7.6	DataOutputStream/DataInputStream	805
14.7.7	Basisklassen für Filter	805
14.7.8	Die Basisklasse FilterWriter	806
14.7.9	Ein LowerCaseWriter	807
14.7.10	Eingaben mit der Klasse FilterReader filtern	808
14.8	Vermittler zwischen Byte-Streams und Unicode-Strömen	809
14.8.1	Datenkonvertierung durch den OutputStreamWriter	809
14.8.2	Automatische Konvertierungen mit dem InputStreamReader	810
14.9	Kommunikation zwischen Threads mit Pipes	812
14.9.1	PipedOutputStream und PipedInputStream	812
14.9.2	PipedWriter und PipedReader	814
14.10	Datenkompression	815
14.10.1	Java-Unterstützung beim Komprimieren	816
14.10.2	Datenströme komprimieren	817
14.10.3	Zip-Archive	821
14.10.4	Jar-Archive	827
14.11	Prüfsummen	827
14.11.1	Die Schnittstelle Checksum	827
14.11.2	Die Klasse CRC32	828
14.11.3	Die Adler32-Klasse	830
14.12	Persistente Objekte und Serialisierung	830
14.12.1	Objekte mit der Standard-Serialisierung speichern und lesen	831
14.12.2	Zwei einfache Anwendungen der Serialisierung	834
14.12.3	Die Schnittstelle Serializable	835
14.12.4	Nicht serialisierbare Attribute aussparen	836

14.12.5	Das Abspeichern selbst in die Hand nehmen	838
14.12.6	Tiefe Objektkopien	841
14.12.7	Versionenverwaltung und die SUID	843
14.12.8	Wie die ArrayList serialisiert	845
14.12.9	Probleme mit der Serialisierung	845
14.12.10	Serialisieren in XML-Dateien	846
14.12.11	JavaBeans Persistence	846
14.12.12	XStream	848
14.13	Tokenizer	849
14.13.1	StreamTokenizer	849
14.13.2	CSV-Dateien verarbeiten	852
14.14	Zum Weiterlesen	853

15 Die eXtensible Markup Language (XML) 855

15.1	Auszeichnungssprachen	855
15.1.1	Die Standard Generalized Markup Language (SGML)	855
15.1.2	Extensible Markup Language (XML)	856
15.2	Eigenschaften von XML-Dokumenten	856
15.2.1	Elemente und Attribute	856
15.2.2	Beschreibungssprache für den Aufbau von XML-Dokumenten ...	858
15.2.3	Schema – eine Alternative zu DTD	861
15.2.4	Namensraum (Namespace)	863
15.2.5	XML-Applikationen	865
15.3	Die Java-APIs für XML	865
15.3.1	Das Document Object Model (DOM)	866
15.3.2	Simple API for XML Parsing (SAX)	866
15.3.3	Pull-API StAX	866
15.3.4	Java Document Object Model (JDOM)	866
15.3.5	JAXP als Java-Schnittstelle zu XML	867
15.3.6	DOM-Bäume einlesen mit JAXP	867
15.4	Serielle Verarbeitung mit StAX	868
15.4.1	Unterschiede der Verarbeitungsmodelle	868
15.4.2	XML-Dateien mit dem Cursor-Verfahren lesen	870
15.4.3	XML-Dateien mit dem Iterator-Verfahren verarbeiten	872
15.4.4	Mit Filtern arbeiten	873
15.4.5	XML-Dokumente schreiben	874
15.5	Serielle Verarbeitung von XML mit SAX	876
15.5.1	Schnittstellen von SAX	877
15.5.2	SAX-Parser erzeugen	877
15.5.3	Operationen der Schnittstelle ContentHandler	878
15.5.4	ErrorHandler und EntityResolver	880

15.6	XML-Dateien mit JDOM verarbeiten	881
15.6.1	JDOM beziehen	881
15.6.2	Paketübersicht	882
15.6.3	Die Document-Klasse	883
15.6.4	Eingaben aus der Datei lesen	884
15.6.5	Das Dokument im XML-Format ausgeben	885
15.6.6	Der Dokumenttyp	885
15.6.7	Elemente	886
15.6.8	Zugriff auf Elementinhalte	889
15.6.9	Liste mit Unterelementen erzeugen	891
15.6.10	Neue Elemente einfügen und ändern	891
15.6.11	Attributinhalte lesen und ändern	894
15.6.12	XPath	896
15.7	Transformationen mit XSLT	899
15.7.1	Templates und XPath als Kernelemente von XSLT	899
15.7.2	Umwandlung von XML-Dateien mit JDOM und JAXP	901
15.8	Java Architecture for XML Binding (JAXB)	902
15.8.1	Beans für JAXB aufbauen	902
15.8.2	JAXBContext und die Marshaller/Unmarshaller	904
15.8.3	Weitere JAXB-Annotationen	905
15.9	HTML-Dokumente einlesen	910
15.10	Zum Weiterlesen	912

16 Grafische Oberflächen mit Swing 913

16.1	Das Abstract Window Toolkit und Swing	913
16.1.1	SwingSet-Demos	913
16.1.2	Abstract Window Toolkit (AWT)	913
16.1.3	Java Foundation Classes	914
16.1.4	Was Swing von AWT unterscheidet	917
16.1.5	Die Klasse Toolkit	918
16.2	Fenster unter grafischen Oberflächen	918
16.2.1	Swing-Fenster darstellen	919
16.2.2	Fenster schließbar machen – setDefaultCloseOperation()	920
16.2.3	AWT-Fenster darstellen	920
16.2.4	Sichtbarkeit des Fensters	921
16.2.5	Größe und Position des Fensters verändern	921
16.2.6	Unterklassen der Fenster-Klassen bilden	923
16.2.7	Fenster- und Dialog-Dekoration, Transparenz	923
16.2.8	Dynamisches Layout während einer Größenänderung	924
16.3	Beschriftungen (JLabel)	924
16.3.1	Mehrzeiliger Text, HTML in der Darstellung	928
16.4	Icon und ImageIcon für Bilder auf Swing-Komponenten	928

16.5	Es tut sich was – Ereignisse beim AWT	930
16.5.1	Die Klasse AWTEvent	930
16.5.2	Events auf verschiedenen Ebenen	931
16.5.3	Swings Ereignisquellen und Horcher (Listener)	933
16.5.4	Listener implementieren	934
16.5.5	Listener bei dem Ereignisauslöser anmelden/abmelden	936
16.5.6	Aufrufen der Listener im AWT-Event-Thread	937
16.5.7	Adapterklassen nutzen	937
16.5.8	Innere Mitgliedsklassen und innere anonyme Klassen	939
16.6	Schaltflächen	940
16.6.1	Normale Schaltflächen (JButton)	941
16.6.2	Der aufmerksame ActionListener	942
16.6.3	Basisklasse AbstractButton	944
16.6.4	Wechselknopf (JToggleButton)	946
16.7	Swing Action	947
16.8	JComponent und Component als Basis aller Komponenten	948
16.8.1	Tooltips	949
16.8.2	Rahmen (Border)	949
16.8.3	Fokus und Navigation	951
16.8.4	Ereignisse jeder Komponente	952
16.8.5	Die Größe und Position einer Komponente	955
16.8.6	Komponenten-Ereignisse	956
16.8.7	Hinzufügen von Komponenten	957
16.8.8	UI-Delegate – der wahre Zeichner	957
16.8.9	Undurchsichtige (opaque) Komponente	960
16.8.10	Properties und Listener für Änderungen	960
16.9	Container	960
16.9.1	Standardcontainer (JPanel)	961
16.9.2	Bereich mit automatischen Rollbalken (JScrollPane)	961
16.9.3	Reiter (JTabbedPane)	962
16.9.4	Teilungs-Komponente (JSplitPane)	964
16.10	Alles Auslegungssache: die Layoutmanager	964
16.10.1	Übersicht über Layoutmanager	964
16.10.2	Zuweisen eines Layoutmanagers	965
16.10.3	Im Fluss mit FlowLayout	966
16.10.4	Mit BorderLayout in allen Himmelsrichtungen	968
16.10.5	Rasteranordnung mit GridLayout	970
16.10.6	Der GridBagLayout-Manager	972
16.10.7	Null-Layout	976
16.10.8	BoxLayout	977
16.10.9	Weitere Layoutmanager	978

16.11	Rollbalken und Schieberegler	978
16.11.1	Schieberegler (JSlider)	978
16.11.2	Rollbalken (JScrollBar)	980
16.12	Kontrollfelder, Optionsfelder, Kontrollfeldgruppen	984
16.12.1	Kontrollfelder (JCheckBox)	985
16.12.2	ItemSelectable, ItemListener und das ItemEvent	986
16.12.3	Sich gegenseitig ausschließende Optionen (JRadioButton)	987
16.13	Fortschritte bei Operationen überwachen	989
16.13.1	Fortschrittsbalken (JProgressBar)	989
16.13.2	Dialog mit Fortschrittsanzeige (ProgressMonitor)	991
16.14	Menüs und Symbolleisten	991
16.14.1	Die Menüleisten und die Einträge	991
16.14.2	Menüeinträge definieren	993
16.14.3	Einträge durch Action-Objekte beschreiben	994
16.14.4	Mit der Tastatur: Mnemonics und Shortcut	995
16.14.5	Der Tastatur-Shortcut (Accelerator)	996
16.14.6	Tastenkürzel (Mnemonics)	997
16.14.7	Symbolleisten alias Toolbars	998
16.14.8	Popup-Menüs	1000
16.14.9	System-Tray nutzen	1004
16.15	Das Model-View-Controller-Konzept	1005
16.16	Auswahlmenüs, Listen und Spinner	1007
16.16.1	Auswahlmenü (JComboBox)	1007
16.16.2	Zuordnung einer Taste mit einem Eintrag	1010
16.16.3	Datumsauswahl	1011
16.16.4	Listen (JList)	1011
16.16.5	Drehfeld (JSpinner)	1016
16.17	Texteingabefelder	1017
16.17.1	Text in einer Eingabezeile	1018
16.17.2	Die Oberklasse der Text-Komponenten (JTextComponent)	1019
16.17.3	Geschützte Eingaben (JPasswordField)	1020
16.17.4	Validierende Eingabefelder (JFormattedTextField)	1021
16.17.5	Einfache mehrzeilige Textfelder (JTextArea)	1022
16.17.6	Editor-Klasse (JEditorPane)	1024
16.18	Tabellen (JTable)	1027
16.18.1	Ein eigenes Tabellen-Model	1028
16.18.2	Basisklasse für eigene Modelle (AbstractTableModel)	1029
16.18.3	Vorgefertigtes Standard-Modell (DefaultTableModel)	1032
16.18.4	Ein eigener Renderer für Tabellen	1033
16.18.5	Zell-Editoren	1036
16.18.6	Größe und Umrandung der Zellen	1037
16.18.7	Spalteninformationen	1038

16.18.8	Tabellenkopf von Swing-Tabellen	1039
16.18.9	Selektionen einer Tabelle	1039
16.18.10	Automatisches Sortieren und Filtern mit RowSorter	1040
16.19	Bäume (JTree)	1041
16.19.1	JTree und sein TreeModel und TreeNode	1041
16.19.2	Selektionen bemerken	1043
16.19.3	Das TreeModel von JTree	1043
16.20	JRootPane und JDesktopPane	1045
16.20.1	Wurzelkomponente der Top-Level-Komponenten (JRootPane)	1046
16.20.2	JDesktopPane und die Kinder JInternalFrame	1046
16.21	Dialoge und Window-Objekte	1048
16.21.1	JWindow und JDialog	1048
16.21.2	Modal oder nicht-modal	1048
16.21.3	Standarddialoge mit JOptionPane	1049
16.21.4	Der Dateiauswahldialog	1051
16.22	Flexibles Java-Look-and-Feel	1054
16.22.1	Look and Feel global setzen	1054
16.22.2	UIManager	1055
16.22.3	Windowsoptik mit JGoodies Looks verbessern	1056
16.23	Die Zwischenablage (Clipboard)	1056
16.23.1	Clipboard-Objekte	1056
16.23.2	Auf den Inhalt zugreifen mit Transferable	1057
16.23.3	DataFlavor ist das Format der Daten in der Zwischenablage	1058
16.23.4	Einfügungen in der Zwischenablage erkennen	1060
16.23.5	Drag & Drop	1060
16.24	Undo durchführen	1061
16.25	AWT, Swing und die Threads	1063
16.25.1	Ereignisschlange (EventQueue) und AWT-Event-Thread	1063
16.25.2	Swing ist nicht Thread-sicher	1064
16.25.3	invokeLater() und invokeAndWait()	1065
16.25.4	SwingWorker	1066
16.25.5	Eigene Ereignisse in die Queue setzen	1068
16.25.6	Auf alle Ereignisse hören	1069
16.26	Barrierefreiheit mit der Java Accessibility API	1069
16.27	Zeitliches Ausführen mit dem javax.swing.Timer	1070
16.28	Alternativen zu AWT und Swing	1071
16.28.1	XML-Beschreibungen der Oberfläche: Swixml, XUL/Luxor	1071
16.28.2	SWT (Standard Widget Toolkit)	1072
16.29	Zum Weiterlesen	1073

17	Grafikprogrammierung	1075
17.1	Grundlegendes zum Zeichnen	1075
17.1.1	Die paint()-Methode für das AWT-Frame	1075
17.1.2	Zeichnen von Inhalten mit JFrame	1077
17.1.3	Auffordern zum Neuzeichnen mit repaint()	1078
17.1.4	Grundbegriffe: Koordinaten, Punkte, Pixel	1078
17.1.5	Die ereignisorientierte Programmierung ändert Fensterinhalte ...	1079
17.1.6	Java 2D-API	1080
17.2	Einfache Zeichenmethoden	1081
17.2.1	Linien	1081
17.2.2	Rechtecke	1082
17.2.3	Ovale und Kreisbögen	1083
17.2.4	Polygone und Polylines	1084
17.3	Zeichenketten schreiben und Fonts	1085
17.3.1	Zeichenfolgen schreiben	1085
17.3.2	Die Font-Klasse	1086
17.3.3	Einen neuen Font aus einem gegebenen Font ableiten	1088
17.3.4	Zeichensätze des Systems ermitteln	1089
17.3.5	Neue TrueType-Fonts in Java nutzen	1089
17.3.6	Font-Metadaten durch FontMetrics	1090
17.4	Geometrische Objekte	1094
17.4.1	Die Schnittstelle Shape	1095
17.4.2	Kreisförmiges	1096
17.4.3	Kurviges	1097
17.4.4	Area und die konstruktive Flächengeometrie	1097
17.4.5	Pfade	1097
17.4.6	Punkt in einer Form, Schnitt von Linien, Abstand Punkt/Linie ...	1100
17.5	Das Innere und Äußere einer Form	1102
17.5.1	Farben und die Paint-Schnittstelle	1102
17.5.2	Farben mit der Klasse Color	1103
17.5.3	Die Farben des Systems über SystemColor	1107
17.5.4	Composite und Xor	1111
17.5.5	Dicke und Art der Linien von Formen bestimmen über Stroke ...	1111
17.6	Bilder	1116
17.6.1	Eine Übersicht über die Bilder-Bibliotheken	1116
17.6.2	Bilder mit ImageIO lesen	1117
17.6.3	Ein Bild zeichnen	1119
17.6.4	Programm-Icon/Fenster-Icon setzen	1122
17.6.5	Splash-Screen	1123
17.6.6	Bilder im Speicher erzeugen	1123
17.6.7	Pixel für Pixel auslesen und schreiben	1125
17.6.8	Bilder skalieren	1127

17.6.9	Schreiben mit ImageIO	1129
17.6.10	Asynchrones Laden mit getImage() und dem MediaTracker	1133
17.7	Weitere Eigenschaften von Graphics	1134
17.7.1	Eine Kopie von Graphics erstellen	1134
17.7.2	Koordinatensystem verschieben	1135
17.7.3	Beschnitt (Clipping)	1135
17.7.4	Zeichenhinweise durch RenderingHints	1138
17.7.5	Transformationen mit einem AffineTransform-Objekt	1139
17.8	Drucken	1140
17.8.1	Drucken der Inhalte	1140
17.8.2	Bekannte Drucker	1143
17.9	Zum Weiterlesen	1144

18 Netzwerkprogrammierung 1145

18.1	Grundlegende Begriffe	1145
18.1.1	Internet-Standards und RFC	1145
18.2	URI und URL	1146
18.2.1	URI	1146
18.2.2	Die Klasse URL	1146
18.2.3	Informationen über eine URL	1149
18.2.4	Der Zugriff auf die Daten über die Klasse URL	1150
18.2.5	Verbindungen durch einen Proxy-Server	1152
18.3	Die Klasse URLConnection	1153
18.3.1	Methoden und Anwendung von URLConnection	1153
18.3.2	Protokoll- und Content-Handler	1155
18.3.3	Im Detail: vom URL zur URLConnection	1156
18.3.4	Der Protokoll-Handler für Jar-Dateien	1157
18.3.5	Basic Authentication/Proxy-Authentifizierung	1159
18.4	Mit GET und POST Daten übergeben	1160
18.4.1	Kodieren der Parameter für Serverprogramme	1161
18.4.2	Eine Suchmaschine ansprechen	1162
18.5	Host- und IP-Adressen	1163
18.5.1	Lebt der Rechner?	1165
18.5.2	IP-Adresse des lokalen Hosts	1166
18.5.3	Das Netz ist Klasse	1167
18.6	NetworkInterface	1167
18.7	Mit dem Socket zum Server	1168
18.7.1	Das Netzwerk ist der Computer	1168
18.7.2	Sockets	1168
18.7.3	Eine Verbindung zum Server aufbauen	1169
18.7.4	Server unter Spannung: die Ströme	1170
18.7.5	Die Verbindung wieder abbauen	1171

18.7.6	Informationen über den Socket	1171
18.7.7	Reine Verbindungsdaten über SocketAddress	1172
18.8	Client-Server-Kommunikation	1173
18.8.1	Warten auf Verbindungen	1174
18.8.2	Ein Multiplikationsserver	1175
18.8.3	Blockierendes Lesen	1178
18.8.4	Von außen erreichbar sein	1179
18.9	Apache Jakarta Commons HttpClient und Net	1179
18.9.1	Jakarta Commons HttpClient	1180
18.9.2	Jakarta Commons Net	1180
18.10	Arbeitsweise eines Webserver	1181
18.10.1	Das Hypertext Transfer Protocol (HTTP)	1181
18.10.2	Anfragen an den Server	1181
18.10.3	Die Antworten vom Server	1184
18.10.4	Webserver mit com.sun.net.httpserver.HttpServer	1187
18.11	Datagram-Sockets	1189
18.11.1	Die Klasse DatagramSocket	1190
18.11.2	Datagramme und die Klasse DatagramPacket	1192
18.11.3	Auf ein hereinkommendes Paket warten	1192
18.11.4	Ein Paket zum Senden vorbereiten	1193
18.11.5	Methoden der Klasse DatagramPacket	1194
18.11.6	Das Paket senden	1195
18.12	E-Mail	1196
18.12.1	Wie eine E-Mail um die Welt geht	1196
18.12.2	Das Simple Mail Transfer Protocol und RFC 822	1196
18.12.3	POP (Post Office Protocol)	1197
18.12.4	Die JavaMail API	1197
18.12.5	E-Mails mittels POP3 abrufen	1198
18.12.6	E-Mails versenden	1201
18.12.7	Ereignisse und Suchen	1203
18.13	Tiefer liegende Netzwerkeigenschaften	1204
18.13.1	Internet Control Message Protocol (ICMP)	1204
18.13.2	MAC-Adresse	1204
18.14	Zum Weiterlesen	1205

19 Verteilte Programmierung mit RMI und Web-Services 1207

19.1	Entfernte Objekte und Methoden	1207
19.1.1	Stellvertreter helfen bei entfernten Methodenaufrufen	1207
19.1.2	Standards für entfernte Objekte	1209
19.2	Java Remote Method Invocation	1209
19.2.1	Zusammenspiel von Server, Registry und Client	1209
19.2.2	Wie die Stellvertreter die Daten übertragen	1209

19.2.3	Probleme mit entfernten Methoden	1210
19.2.4	Nutzen von RMI bei Middleware-Lösungen	1211
19.2.5	Zentrale Klassen und Schnittstellen	1212
19.2.6	Entfernte und lokale Objekte im Vergleich	1213
19.3	Auf der Serverseite	1213
19.3.1	Entfernte Schnittstelle deklarieren	1213
19.3.2	Remote-Objekt-Implementierung	1214
19.3.3	Stellvertreterobjekte	1215
19.3.4	Der Namensdienst (Registry)	1215
19.3.5	Remote-Objekt-Implementierung exportieren und beim Namensdienst anmelden	1217
19.3.6	Einfaches Logging	1219
19.3.7	Aufräumen mit dem DGC	1220
19.4	Auf der Clientseite	1220
19.5	Entfernte Objekte übergeben und laden	1221
19.5.1	Klassen vom RMI-Klassenlader nachladen	1222
19.6	Weitere Eigenschaften von RMI	1222
19.6.1	RMI und CORBA	1222
19.6.2	RMI über HTTP getunnelt	1223
19.6.3	Automatische Remote-Objekt-Aktivierung	1223
19.7	Daily Soap	1224
19.7.1	SOAP-Protokoll	1224
19.7.2	Die technische Realisierung	1225
19.7.3	SOAP-Implementierungen	1225
19.7.4	@WebService in Java 6	1226
19.7.5	Einen Web-Service definieren	1226
19.7.6	Web-Services veröffentlichen	1227
19.7.7	Einen JAX-WS-Client implementieren	1228
19.8	Java Message Service (JMS)	1229
19.9	Zum Weiterlesen	1230

20 JavaServer Pages und Servlets 1231

20.1	Dynamisch generierte Webseiten	1231
20.1.1	Was sind Servlets?	1231
20.1.2	Was sind JavaServer Pages?	1232
20.2	Servlets und JSPs mit Tomcat entwickeln	1233
20.2.1	Servlet-Container	1233
20.2.2	Entwicklung der Servlet-/JSP-Spezifikationen	1234
20.2.3	Webserver mit Servlet-Funktionalität	1234
20.2.4	Tomcat installieren	1234
20.2.5	Ablageort für eigene JSPs	1235
20.2.6	Web-Applikationen	1236

20.2.7	Zuordnung von Web-Applikationen zu physikalischen Verzeichnissen	1237
20.2.8	Web-Projekt mit Eclipse IDE for Java EE Developers	1237
20.3	Statisches und Dynamisches	1238
20.3.1	Statischer Template-Code	1238
20.3.2	Dynamische Inhalte	1239
20.3.3	Kommentare	1239
20.4	Die Expression Language (EL)	1240
20.4.1	Operatoren der EL	1240
20.4.2	Literale	1241
20.4.3	Implizite EL-Objekte	1241
20.5	Formulardaten	1242
20.5.1	Einen Parameter auslesen	1242
20.5.2	HTML-Formulare	1242
20.6	Auf Beans zurückgreifen	1244
20.6.1	Beans in JSPs anlegen	1244
20.6.2	Properties einer Bean im EL-Ausdruck erfragen	1244
20.6.3	Properties mit <code><jsp:setProperty></code> setzen	1244
20.6.4	Bean-Klasse zum Testen von E-Mail-Adressen	1245
20.6.5	Parameterwerte in Bean übertragen	1246
20.7	JSP-Tag-Libraries	1247
20.7.1	Standard Tag Library (JSTL)	1247
20.8	Einbinden und Weiterleiten	1251
20.8.1	Einbinden von Inhalten	1251
20.8.2	Forward und Redirect	1252
20.8.3	Applets einbinden	1253
20.9	Scripting-Elemente in JSPs	1253
20.9.1	Scriptlets	1254
20.9.2	JSP-Ausdrücke	1254
20.9.3	JSP-Deklarationen	1254
20.9.4	Quoting	1255
20.9.5	Entsprechende XML-Tags	1255
20.9.6	Implizite Objekte für Scriptlets und JSP-Ausdrücke	1255
20.10	JSP-Direktiven	1256
20.10.1	page-Direktiven im Überblick	1256
20.10.2	Mit JSPs Bilder generieren	1257
20.11	Sitzungsverfolgung (Session Tracking)	1258
20.11.1	Lösungen für Sitzungsverfolgung	1259
20.11.2	Auf Session-Dateien zurückgreifen	1260
20.12	Servlets	1260
20.12.1	Servlets compilieren	1261
20.12.2	Servlet-Mapping	1262

20.12.3	Der Lebenszyklus eines Servlets	1263
20.12.4	Mehrere Anfragen beim Servlet und die Thread-Sicherheit	1263
20.12.5	Servlets und Sessions	1263
20.12.6	Weiterleiten und Einbinden von Servlet-Inhalten	1264
20.13	Zum Weiterlesen	1265

21 Applets 1267

21.1	Applets in der Wiege von Java	1267
21.1.1	(J)Applet und Applikationen	1268
21.1.2	Das erste Hallo-Applet	1268
21.1.3	HTML Converter	1270
21.1.4	Fehler in Applets finden	1272
21.2	Die Applet-API	1272
21.2.1	Die Zyklen eines Applets	1272
21.2.2	Parameter an das Applet übergeben	1272
21.2.3	Wie das Applet den Browser-Inhalt ändern kann	1274
21.2.4	Den Ursprung des Applets erfragen	1274
21.2.5	Datenaustausch zwischen Applets	1275
21.2.6	Was ein Applet alles darf	1278
21.2.7	Ist Java im Browser aktiviert?	1279
21.2.8	Applet unter Netscape oder Microsoft Internet Explorer?	1279
21.3	Webstart	1280

22 Midlets und die Java ME 1283

22.1	Java Platform, Micro Edition (Java ME)	1283
22.2	Konfigurationen	1283
22.2.1	Connected Limited Device Configuration (CLDC)	1284
22.2.2	Connected Device Configuration (CDC)	1284
22.3	Profile	1284
22.3.1	Mobile Information Device Profile (MIDP)	1285
22.3.2	Weitere Profile	1285
22.4	Wireless Toolkits	1285
22.4.1	Sun Java Wireless Toolkit for CLDC	1285
22.4.2	Eclipse-Plugin	1286
22.5	Die Midlet-API	1289
22.5.1	Paketstruktur Mobile Information Device Profile (2.0)	1289
22.6	Zum Weiterlesen	1290

23 Datenbankmanagement mit JDBC 1291

23.1	Das relationale Modell	1291
------	------------------------------	------

23.2	Datenbanken und Tools	1292
23.2.1	HSQLDB	1292
23.2.2	Weitere Datenbanken	1293
23.2.3	Eclipse-Plugins zum Durchschauen von Datenbanken	1295
23.3	JDBC und Datenbanktreiber	1297
23.3.1	Treibertypen	1298
23.3.2	JDBC-Versionen	1299
23.4	Eine Beispielabfrage	1300
23.4.1	Schritte zur Datenbankabfrage	1300
23.4.2	Client für HSQLDB-Datenbank	1301
23.5	Mit Java an eine Datenbank andocken	1302
23.5.1	Der Treiber-Manager	1303
23.5.2	Den Treiber laden	1303
23.5.3	Eine Aufzählung aller Treiber	1304
23.5.4	Log-Informationen	1305
23.5.5	Verbindung zur Datenbank auf- und abbauen	1305
23.5.6	DataSource	1308
23.5.7	Gepoolte Verbindungen	1310
23.6	Datenbankabfragen	1311
23.6.1	Abfragen über das Statement-Objekt	1311
23.6.2	Ergebnisse einer Abfrage in ResultSet	1313
23.6.3	Java und SQL-Datentypen	1314
23.6.4	Unicode in der Spalte korrekt auslesen	1317
23.6.5	Eine SQL-NULL und wasNull() bei ResultSet	1318
23.6.6	Wie viele Zeilen hat ein ResultSet?	1318
23.7	Die Ausnahmen bei JDBC	1319
23.8	Elemente einer Datenbank hinzufügen und aktualisieren	1320
23.8.1	Batch-Updates	1320
23.9	ResultSets in Bohnen durch RowSet	1322
23.9.1	Die Schnittstelle RowSet	1322
23.9.2	Implementierungen von RowSet	1322
23.9.3	Der Typ CachedRowSet	1323
23.9.4	Der Typ WebRowSet	1324
23.10	Vorbereitete Anweisungen (Prepared Statements)	1326
23.10.1	PreparedStatement-Objekte vorbereiten	1326
23.10.2	Werte für die Platzhalter eines PreparedStatement	1327
23.11	Transaktionen	1328
23.12	Metadaten	1329
23.12.1	Metadaten über die Tabelle	1329
23.12.2	Informationen über die Datenbank	1332
23.13	Einführung in SQL	1333
23.13.1	Ein Rundgang durch SQL-Abfragen	1334

23.13.2	Datenabfrage mit der Data Query Language (DQL)	1335
23.13.3	Tabellen mit der Data Definition Language (DDL) anlegen	1337
23.14	Zum Weiterlesen	1337

24 Reflection und Annotationen 1339

24.1	Metadaten	1339
24.1.1	Metadaten durch JavaDoc-Tags	1339
24.2	Metadaten der Klassen mit dem Class-Objekt	1340
24.2.1	An ein Class-Objekt kommen	1340
24.2.2	Was das Class-Objekt beschreibt	1342
24.2.3	Der Name der Klasse	1344
24.2.4	instanceof mit Class-Objekten	1346
24.2.5	Oberklassen finden	1347
24.2.6	Implementierte Interfaces einer Klasse oder eines Interfaces	1348
24.2.7	Modifizierer und die Klasse Modifier	1348
24.2.8	Die Arbeit auf dem Feld	1350
24.3	Attribute, Methoden und Konstruktoren	1351
24.3.1	Reflections – Gespür für Attribute einer Klasse	1351
24.3.2	Methoden einer Klasse erfragen	1354
24.3.3	Properties einer Bean erfragen	1358
24.3.4	Konstruktoren einer Klasse	1358
24.3.5	Annotationen	1360
24.4	Objekte erzeugen und manipulieren	1361
24.4.1	Objekte erzeugen	1361
24.4.2	Die Belegung der Variablen erfragen	1362
24.4.3	Eine generische toString()-Funktion	1364
24.4.4	Variablen setzen	1366
24.4.5	Private Attribute ändern	1368
24.5	Methoden aufrufen	1368
24.5.1	Statische Methoden aufrufen	1369
24.5.2	Dynamische Methodenaufrufe bei festen Methoden beschleunigen	1370
24.6	Annotationen	1372
24.6.1	Neue Annotationen definieren	1372
24.6.2	Annotationen mit genau einem Element	1372
24.6.3	Beliebige Schlüssel-Werte-Paare	1374
24.6.4	Vorbelegte Elemente	1377
24.6.5	Annotieren von Annotationstypen	1378
24.6.6	Annotationen zur Laufzeit auslesen	1380
24.6.7	Mögliche Nachteile von Annotationen	1382
24.6.8	XDoclet	1383
24.7	Zum Weiterlesen	1384

25 Logging und Monitoring 1385

25.1	Logging mit Java	1385
25.1.1	Logging-APIs	1385
25.1.2	Logging mit log4j	1386
25.2	Überwachen von Systemzuständen	1388
25.3	MBean-Typen, MBean-Server und weitere Begriffe	1389
25.3.1	MXBeans des Systems	1390
25.4	Geschwätzige Programme und JConsole	1392
25.4.1	JConsole	1392
25.5	Der MBeanServer	1393
25.6	Eine eigene Standard-MBean	1395
25.6.1	Management-Schnittstelle	1395
25.6.2	Implementierung der Managed-Ressource	1395
25.6.3	Anmeldung beim Server	1396
25.6.4	Eigene Bean in JConsole	1396
25.7	Zum Weiterlesen	1399

26 Sicherheitskonzepte 1401

26.1	Zentrale Elemente der Java-Sicherheit	1401
26.1.1	Security-API der Java SE	1401
26.1.2	Cryptographic Service Providers	1402
26.2	Der Sandkasten (Sandbox)	1403
26.3	Sicherheitsmanager (Security Manager)	1404
26.3.1	Der Sicherheitsmanager bei Applets	1405
26.3.2	Sicherheitsmanager aktivieren	1406
26.3.3	Rechte durch Policy-Dateien vergeben	1408
26.3.4	Erstellen von Rechedateien mit dem grafischen Policy-Tool	1410
26.3.5	Kritik an den Policies	1410
26.4	Signierung	1412
26.4.1	Warum signieren?	1412
26.4.2	Digitale Ausweise und die Zertifizierungsstelle	1412
26.4.3	Mit keytool Schlüssel erzeugen	1413
26.4.4	Signieren mit jarsigner	1414
26.5	Digitale Unterschriften	1414
26.5.1	Die MDx-Reihe	1415
26.5.2	Secure Hash Algorithm (SHA)	1415
26.5.3	Mit der Security-API einen Fingerabdruck berechnen	1416
26.5.4	Die Klasse MessageDigest	1416
26.6	Verschlüsseln von Daten(-strömen)	1418
26.6.1	Den Schlüssel bitte	1418

26.6.2	Verschlüsseln mit Cipher	1420
26.6.3	Verschlüsseln von Datenströmen	1420
26.7	Zum Weiterlesen	1421

27 Java Native Interface (JNI) 1423

27.1	Java Native Interface und Invocation-API	1423
27.2	Einbinden einer C-Funktion in ein Java-Programm	1424
27.2.1	Schreiben des Java-Codes	1424
27.2.2	Compilieren des Java-Programms	1425
27.2.3	Erzeugen der Header-Datei	1425
27.2.4	Implementierung der Funktion in C	1426
27.2.5	Übersetzen der C-Programme und Erzeugen der dynamischen Bibliothek	1427
27.2.6	Suchort der dynamischen Bibliothek	1429
27.3	Nativ die Stringlänge ermitteln	1429
27.4	Erweiterte JNI-Eigenschaften	1430
27.4.1	Klassendefinitionen	1430
27.4.2	Zugriff auf Attribute	1431
27.5	Einfache Anbindung von existierenden Bibliotheken	1433
27.5.1	C++ Klassen ansprechen	1433
27.5.2	COM-Schnittstellen anzapfen	1433
27.6	Zum Weiterlesen	1434

28 Dienstprogramme für die Java-Umgebung 1435

28.1	Die Werkzeuge im Überblick	1435
28.2	Java-Compiler	1435
28.2.1	Bytecode-Compiler javac	1435
28.2.2	Native Compiler	1436
28.2.3	Java-Programme in ein natives ausführbares Programm einpacken	1437
28.3	Der Java-Interpreter java	1437
28.3.1	Der Unterschied zwischen java.exe und javaw.exe	1439
28.4	Das Archivformat Jar	1439
28.4.1	Das Dienstprogramm Jar benutzen	1440
28.4.2	Das Manifest	1442
28.4.3	Applikationen in Jar-Archiven starten	1442
28.4.4	Applets in Jar-Archiven	1443
28.5	Monitoringprogramme	1444
28.5.1	jps	1444
28.5.2	jstat	1444
28.5.3	jmap	1445
28.5.4	jstack	1445

28.6	Ant	1446
28.6.1	Bezug und Installation von Ant	1446
28.6.2	Properties	1448
28.6.3	Externe und vordefinierte Properties	1449
28.6.4	Weitere Ant-Tasks	1449
28.6.5	Das Obfuscator-Programm ProGuard	1450
28.7	Weitere Dienstprogramme	1451
28.7.1	Sourcecode Beautifier	1451
28.7.2	Java-Programme als Systemdienst ausführen	1452
28.8	Zum Weiterlesen	1453
Index	1455