

Auf einen Blick

Vorwort	39
1 Java ist auch eine Sprache	53
2 Sprachbeschreibung	83
3 Klassen und Objekte	175
4 Der Umgang mit Zeichenketten	221
5 Mathematisches	275
6 Eigene Klassen schreiben	301
7 Exceptions	433
8 Die Funktionsbibliothek	459
9 Threads und nebenläufige Programmierung	515
10 Raum und Zeit	595
11 Datenstrukturen und Algorithmen	627
12 Dateien und Datenströme	693
13 Die eXtensible Markup Language (XML)	803
14 Grafische Oberflächen mit Swing	855
15 Grafikprogrammierung	1021
16 Das Netz	1093
17 JavaServer Pages und Servlets	1155
18 Verteilte Programmierung mit RMI und Web-Services	1211
19 Applets, Midlets und Sound	1235
20 Datenbankmanagement mit JDBC	1253
21 Reflection und Annotationen	1301
22 Komponenten durch Bohnen	1345
23 Logging und Monitoring	1359
24 Sicherheitskonzepte	1371
25 Java Native Interface (JNI)	1393
26 Dienstprogramme für die Java-Umgebung	1405
A Die Begleit-DVD	1427
Index	1429

Inhalt

Vorwort	39
---------------	----

1 Java ist auch eine Sprache	53
1.1 Der erste Kontakt	53
1.2 Historischer Hintergrund	53
1.3 Eigenschaften von Java	55
1.3.1 Bytecode und die virtuelle Maschine	55
1.3.2 Objektorientierung in Java	56
1.3.3 Java-Security-Modell	57
1.3.4 Zeiger und Referenzen	57
1.3.5 Bring den Müll raus, Garbage-Collector!	58
1.3.6 Ausnahmebehandlung	58
1.3.7 Kein Präprozessor für Textersetzung	59
1.3.8 Keine überladenen Operatoren	60
1.3.9 Java als Sprache, Laufzeitumgebung und Bibliothek	60
1.3.10 Wofür sich Java nicht eignet	61
1.3.11 Java im Vergleich zu anderen Sprachen	62
1.4 Die Rolle von Java im Web	62
1.4.1 Vollwertige Applikationen statt Applets	63
1.5 Die Java Platform Standard Edition (Java SE)	63
1.5.1 JDK und JRE	63
1.5.2 Java-Versionen	64
1.5.3 Java für die Kleinen	65
1.5.4 Java für die Großen	65
1.5.5 Installationsanleitung für Java SE	66
1.6 Das erste Programm compilieren und testen	67
1.6.1 Ein Quadratzahlen-Programm	67
1.6.2 Der Compilerlauf	68
1.6.3 Die Laufzeitumgebung	70
1.6.4 Häufige Compiler- und Interpreterprobleme	70
1.7 Entwicklungsumgebungen im Allgemeinen	71
1.7.1 Java Studio und NetBeans von Sun	71
1.7.2 Die Entwicklungsumgebung Eclipse	71
1.7.3 Together	72
1.7.4 Ein Wort zu Microsoft, Java und zu J++	72
1.8 Eclipse im Speziellen	73
1.8.1 Eclipse starten	74
1.8.2 Das erste Projekt anlegen	75
1.8.3 Eine Klasse hinzufügen	77

1.8.4	Übersetzen und Ausführen	78
1.8.5	JDK statt JRE	78
1.8.6	Start eines Programms ohne Speicheraufforderung	79
1.8.7	Projekt einfügen oder Workspace für die Aufgaben wechseln	79
1.8.8	Plugins für Eclipse	80
1.8.9	Eclipse Web Tools Platform (WTP)	81
1.9	Zum Weiterlesen	81

2 Sprachbeschreibung 83

2.1	Elemente der Programmiersprache Java	83
2.1.1	Textkodierung durch Unicode-Zeichen	83
2.1.2	Literale	86
2.1.3	Bezeichner	86
2.1.4	Reservierte Schlüsselwörter	87
2.1.5	Token	88
2.1.6	Kommentare	89
2.2	Anweisungen und Programme	90
2.2.1	Eine Klasse bildet den Rahmen	91
2.2.2	Die Reise beginnt am main()	92
2.2.3	Programme übersetzen und starten	93
2.2.4	Funktionsaufrufe als Ausdrücke und Anweisungen	93
2.2.5	Modifizierer	96
2.2.6	Anweisungen und Blöcke	96
2.3	Datentypen	97
2.3.1	Primitive Datentypen im Überblick	98
2.3.2	Wahrheitswerte	100
2.3.3	Variablendeklarationen	100
2.3.4	Ganzzahlige Datentypen	104
2.3.5	Die Fließkommazahlen float und double	107
2.3.6	Alphanumerische Zeichen	108
2.3.7	Die Typanpassung (das Casting)	109
2.4	Ausdrücke, Operanden und Operatoren	113
2.4.1	Zuweisungsoperator	114
2.4.2	Arithmetische Operatoren	115
2.4.3	Unäres Minus und Plus	117
2.4.4	Zuweisung mit Operation	118
2.4.5	Präfix- oder Postfix-Inkrement und -Dekrement	119
2.4.6	Die relationalen Operatoren und die Gleichheitsoperatoren	120
2.4.7	Logische Operatoren Und, Oder, Xor, Nicht	121
2.4.8	Rang der Operatoren in der Auswertungsreihenfolge	122
2.4.9	Überladenes Plus für Strings	124
2.4.10	Was C(++)-Programmierer vermissen könnten	125

2.5	Bedingte Anweisungen oder Fallunterscheidungen	126
2.5.1	Die if-Anweisung	126
2.5.2	Die Alternative mit einer if/else-Anweisung wählen	128
2.5.3	Die switch-Anweisung bietet die Alternative	130
2.6	Schleifen	133
2.6.1	Die while-Schleife	133
2.6.2	Schleifenbedingungen und Vergleiche mit ==	134
2.6.3	Die do/while-Schleife	136
2.6.4	Die for-Schleife	137
2.6.5	Ausbruch planen mit break und Wiedereinstieg mit continue	142
2.6.6	break und continue mit Sprungmarken	144
2.7	Methoden einer Klasse	145
2.7.1	Bestandteil einer Funktion	146
2.7.2	Beschreibungen in der Java-API	146
2.7.3	Aufruf einer Methode	148
2.7.4	Methoden ohne Parameter	148
2.7.5	Statische Funktionen (Klassenmethoden)	149
2.7.6	Parameter, Argument und Wertübergabe	150
2.7.7	Methoden vorzeitig mit return beenden	151
2.7.8	Nicht erreichbarer Quellcode bei Funktionen	152
2.7.9	Rückgabewerte	152
2.7.10	Methoden überladen	156
2.7.11	Vorgegebener Wert für nicht aufgeführte Argumente	158
2.7.12	Finale lokale Variablen	158
2.7.13	Rekursive Funktionen	160
2.7.14	Die Türme von Hanoi	163
2.8	Weitere Operatoren	164
2.8.1	Operationen auf Bit-Ebene	165
2.8.2	Die Verschiebeoperatoren	167
2.8.3	Ein Bit setzen, löschen, umdrehen und testen	169
2.8.4	Bit-Funktionen der Integer- und Long-Klasse	170
2.8.5	Der Bedingungsoperator	171
2.9	Einfache Benutzereingaben	172
2.10	Zum Weiterlesen	173

3 Klassen und Objekte

3.1	Objektorientierte Programmierung	175
3.1.1	Warum überhaupt OOP?	175
3.1.2	Wiederverwertbarkeit	176
3.2	Eigenschaften einer Klasse	177
3.2.1	Die Klasse Point	177

3.3	Die UML (Unified Modeling Language)	178
3.3.1	Hintergrund und Geschichte zur UML	178
3.3.2	Wichtige Diagrammtypen der UML	179
3.4	Anlegen und Nutzen eines Punktes	180
3.4.1	Deklarieren von Referenz-Variablen	180
3.4.2	Anlegen eines Exemplars einer Klasse mit dem new-Operator	180
3.4.3	Zugriff auf Variablen und Methoden mit dem »«	181
3.4.4	Konstruktoren nutzen	184
3.5	Import und Pakete	184
3.6	Die API-Dokumentation	185
3.7	Mit Referenzen arbeiten	186
3.7.1	Die null-Referenz	186
3.7.2	Zuweisungen bei Referenzen	188
3.7.3	Funktionen mit nicht-primitiven Parametern	188
3.8	Identität und Gleichheit	190
3.8.1	Identität von Objekten	190
3.8.2	Gleichheit und die Methode equals()	191
3.9	Arrays	193
3.9.1	Deklaration von Arrays	193
3.9.2	Arrays mit Inhalt	194
3.9.3	Die Länge eines Arrays über das Attribut length	195
3.9.4	Zugriff auf die Elemente über den Index	195
3.9.5	Array-Objekte erzeugen	196
3.9.6	Fehler bei Arrays	197
3.9.7	Arrays mit nicht-primitiven Elementen	198
3.9.8	Vorinitialisierte Arrays	199
3.9.9	Die erweiterte for-Schleife	199
3.9.10	Mehrdimensionale Arrays	201
3.9.11	Die Wahrheit über die Array-Initialisierung	203
3.9.12	Mehrere Rückgabewerte	204
3.9.13	Methode mit variabler Argumentanzahl (Vararg)	204
3.9.14	Arrays klonen	205
3.9.15	Feldinhalte kopieren	206
3.9.16	Die Klasse Arrays zum Vergleichen, Füllen und Suchen	207
3.10	Der Einstiegspunkt für das Laufzeitsystem main()	213
3.10.1	Kommandozeilen-Argumente verarbeiten	213
3.10.2	Der Rückgabewert von main() und System.exit()	214
3.10.3	Parser der Kommandozeilenargumente – Apache CLI	214
3.11	Eigene Pakete schnüren	216
3.11.1	Die package-Anweisung	216
3.11.2	Importieren von Klassen mit import	216
3.11.3	Paketnamen	217

3.11.4	Hierarchische Strukturen und das Default-Package	217
3.11.5	Klassen mit gleichen Namen in unterschiedlichen Paketen	218
3.11.6	Statisches Import	218
3.11.7	Eine Verzeichnisstruktur für eigene Projekte	219
3.12	Zum Weiterlesen	220

4 Der Umgang mit Zeichenketten 221

4.1	Einzelne Zeichen mit der Character-Klasse behandeln	221
4.2	Strings und deren Anwendung	222
4.2.1	String-Literale als String-Objekte für konstante Zeichenketten ...	224
4.2.2	String-Länge und Test auf Leerstring	226
4.2.3	Nach enthaltenen Zeichen und Zeichenfolgen suchen	226
4.2.4	Gut, dass wir verglichen haben	227
4.2.5	String-Teile extrahieren	229
4.2.6	Strings anhängen, Groß-/Kleinschreibung und Leerraum	231
4.2.7	Suchen und ersetzen	233
4.2.8	String-Objekte mit Konstruktoren neu anlegen	235
4.3	Konvertieren zwischen Primitiven und Strings	237
4.3.1	Unterschiedliche Typen in Zeichenketten konvertieren	237
4.3.2	String in primitives Element konvertieren	238
4.4	Veränderbare Zeichenketten mit StringBuffer/StringBuilder	239
4.4.1	Anlegen von StringBuffer/StringBuilder-Objekten	239
4.4.2	Länge eines StringBuffer/-Builder-Objekts lesen und setzen	240
4.4.3	Daten anhängen	241
4.4.4	Zeichen(folgen) setzen, erfragen, löschen und umdrehen	241
4.4.5	Vergleichen von String/StringBuffer/StringBuilder	242
4.4.6	hashCode() bei StringBuffer/StringBuilder	244
4.5	Sprachabhängiges Vergleichen mit der Collator-Klasse	244
4.5.1	Die Klasse Collator	244
4.5.2	Effiziente interne Speicherung für die Sortierung	247
4.6	Reguläre Ausdrücke	248
4.6.1	Die Klassen Pattern und Matcher	248
4.6.2	Mit MatchResult alle Ergebnisse einsammeln	250
4.7	Zerlegen von Zeichenketten	252
4.7.1	Splitten von Zeichenketten mit split()	252
4.7.2	split() in Pattern	253
4.7.3	Die Klasse Scanner	254
4.7.4	StringTokenizer	258
4.7.5	BreakIterator als Zeichen-, Wort-, Zeilen- und Satztrenner	260
4.8	Zeichenkodierungen und Base64	263
4.8.1	Über die Klasse String Kodierungen vornehmen	263
4.8.2	Konvertieren mit OutputStreamWriter-Klassen	264

4.8.3	Das Paket <code>java.nio.charset</code>	264
4.8.4	Base64-Kodierung	265
4.9	Formatieren von Ausgaben	266
4.9.1	Formatieren mit <code>format()</code> aus <code>String</code>	266
4.9.2	Die Format-Klassen im Überblick	269
4.9.3	Zahlen, Prozente und Währungen mit <code>NumberFormat</code> und <code>DecimalFormat</code> formatieren	270
4.9.4	Ausgaben mit <code>MessageFormat</code> formatieren	272
4.10	Zum Weiterlesen	274

5 Mathematisches 275

5.1	Arithmetik in Java	275
5.1.1	Mantisse und Exponent	275
5.1.2	Spezialwerte Unendlich, Null, NaN	276
5.2	Wertebereich eines Typs und Überlaufkontrolle	278
5.2.1	Behandlung des Überlaufs	278
5.3	Die Eigenschaften der Klasse <code>Math</code>	280
5.3.1	Attribute	281
5.3.2	Winkelfunktionen	281
5.3.3	Runden von Werten	282
5.3.4	Wurzel und Exponentialfunktionen	284
5.3.5	Der Logarithmus	284
5.3.6	Rest der ganzzahligen Division	285
5.3.7	Absolutwerte und Maximum/Minimum	286
5.3.8	Zufallszahlen	286
5.3.9	Der Nächste, bitte	287
5.4	Mathe bitte strikt	287
5.4.1	Strikt Fließkomma mit <code>strictfp</code>	287
5.4.2	Die Klassen <code>Math</code> und <code>StrictMath</code>	287
5.5	Die Random-Klasse	288
5.6	Große Zahlen	290
5.6.1	Die Klasse <code>BigInteger</code>	290
5.6.2	Funktionen von <code>BigInteger</code>	292
5.6.3	Ganz lange Fakultäten	294
5.6.4	Große Fließkommazahlen mit <code>BigDecimal</code>	295
5.6.5	Mit <code>MathContext</code> komfortabel die Rechengenauigkeit setzen	296
5.7	Rechnen mit Einheiten: Java Units Specification	297
5.8	Zum Weiterlesen	299

6 Eigene Klassen schreiben 301

6.1	Eigene Klassen deklarieren	301
6.1.1	Methodenaufrufe und Nebeneffekte	304

6.1.2	Argumentübergabe mit Referenzen	304
6.1.3	Die this-Referenz	305
6.1.4	Überdeckte Objektvariablen nutzen	306
6.2	Privatsphäre und Sichtbarkeit	307
6.2.1	Wieso nicht freie Methoden und Variablen für alle?	309
6.2.2	Privat ist nicht ganz privat: Es kommt darauf an, wer's sieht	309
6.2.3	Zugriffsmethoden für Attribute deklarieren	310
6.3	Statische Methoden und statische Attribute	313
6.3.1	Warum statische Eigenschaften sinnvoll sind	314
6.3.2	Statische Eigenschaften mit static	314
6.3.3	Statische Eigenschaften über Referenzen nutzen?	315
6.3.4	Warum die Groß- und Kleinschreibung wichtig ist	315
6.3.5	Statische Eigenschaften und Objekteigenschaften	316
6.3.6	Statische Variablen zum Datenaustausch	317
6.3.7	Statische Blöcke als Klasseninitialisierer	318
6.4	Konstanten und Aufzählungen	319
6.4.1	Konstanten über öffentliche statische final-Variablen	319
6.4.2	Eincompilierte Belegungen der Klassenvariablen	320
6.4.3	Typsicherere Konstanten	321
6.4.4	Aufzählungen mit enum	323
6.4.5	enum-Konstanten in switch	324
6.4.6	Enum-Objekte in der Weitergabe	324
6.4.7	Statische Imports von Aufzählungen	325
6.5	Objekte anlegen und zerstören	325
6.5.1	Konstruktoren schreiben	325
6.5.2	Konstruktor nimmt ein Objekt vom Typ der eigenen Klasse (Copy-Konstruktor)	328
6.5.3	Einen anderen Konstruktor der gleichen Klasse aufrufen	329
6.5.4	Initialisierung der Objekt- und Klassenvariablen	332
6.5.5	Finale Werte im Konstruktor und in statischen Blöcken setzen ...	334
6.5.6	Exemplarinitialisierer (Instanzinitialisierer)	335
6.5.7	Ihr fehlt uns nicht – der Garbage-Collector	337
6.5.8	Implizit erzeugte String-Objekte	338
6.5.9	Private Konstruktoren, Utility-Klassen, Singleton, Fabriken	339
6.6	Assoziationen zwischen Objekten	341
6.6.1	Gegenseitige Abhängigkeiten von Klassen	342
6.7	Vererbung	342
6.7.1	Vererbung in Java	343
6.7.2	Einfach- und Mehrfachvererbung	343
6.7.3	Gebäude modelliert	344
6.7.4	Konstruktoren in der Vererbung	345
6.7.5	Sichtbarkeit protected	347

6.7.6	Das Substitutionsprinzip	348
6.7.7	Automatische und explizite Typanpassung	349
6.7.8	Typen mit dem binären Operator instanceof testen	350
6.7.9	Array-Typen und Kovarianz	351
6.7.10	Methoden überschreiben	352
6.7.11	Mit super eine Methode der Oberklasse aufrufen	354
6.7.12	Kovariante Rückgabetypen	356
6.7.13	Finale Klassen	356
6.7.14	Nicht überschreibbare Funktionen	357
6.7.15	Zusammenfassung zur Sichtbarkeit	357
6.7.16	Sichtbarkeit in der UML	358
6.7.17	Zusammenfassung: Konstruktoren und Methoden	359
6.8	Die Oberklasse gibt Funktionalität vor	360
6.8.1	Spätes dynamisches Binden als Beispiel für Polymorphie	362
6.8.2	Unpolymorph bei privaten, statischen und finalen Methoden	363
6.8.3	Polymorphie bei Konstruktoraufrufen	365
6.9	Abstrakte Klassen und abstrakte Methoden	367
6.9.1	Abstrakte Klassen	367
6.9.2	Abstrakte Methoden	368
6.10	Schnittstellen	370
6.10.1	Ein Polymorphie-Beispiel mit Schnittstellen	374
6.10.2	Die Mehrfachvererbung bei Schnittstellen	375
6.10.3	Erweitern von Interfaces – Subinterfaces	378
6.10.4	Vererbte Konstanten bei Schnittstellen	378
6.10.5	Schnittstellenmethoden, die nicht implementiert werden müssen	380
6.10.6	Abstrakte Klassen und Schnittstellen im Vergleich	381
6.10.7	CharSequence als Beispiel einer Schnittstelle	381
6.10.8	Die Schnittstelle Iterable	383
6.11	Object ist die Mutter aller Oberklassen	386
6.11.1	Klassenobjekte	386
6.11.2	Objektidentifikation mit <code>toString()</code>	386
6.11.3	Objektgleichheit mit <code>equals()</code> und Identität	388
6.11.4	Klonen eines Objekts mit <code>clone()</code>	392
6.11.5	Hashcodes über <code>hashCode()</code> liefern	395
6.11.6	Aufräumen mit <code>finalize()</code>	397
6.11.7	Synchronisation	398
6.12	Innere Klassen	399
6.12.1	Statische innere Klassen und Schnittstellen	399
6.12.2	Mitglieds- oder Elementklassen	400
6.12.3	Lokale Klassen	403
6.12.4	Anonyme innere Klassen	404
6.12.5	<code>this</code> und Vererbung	407

6.12.6	Implementierung einer verketteten Liste	408
6.12.7	Funktionszeiger	410
6.13	Generische Datentypen	412
6.13.1	Einfache Klassenschablonen	413
6.13.2	Einfache Methodenschablonen	414
6.13.3	Umsetzen der Generics, Typlösung und Raw-Types	415
6.13.4	Einschränken der Typen	416
6.13.5	Generics und Vererbung, Invarianz	418
6.13.6	Wildcards	418
6.14	Die Spezial-Oberklasse Enum	419
6.14.1	Methoden auf Enum-Objekten	420
6.14.2	enum mit eigenen Konstruktoren und Methoden	422
6.15	Dokumentationskommentare mit JavaDoc	424
6.15.1	Einen Dokumentationskommentar setzen	425
6.15.2	Mit javadoc eine Dokumentation erstellen	426
6.15.3	HTML-Tags in Dokumentationskommentaren	426
6.15.4	Generierte Dateien	427
6.15.5	Dokumentationskommentare im Überblick	428
6.15.6	JavaDoc und Doclets	428
6.15.7	Veraltete (deprecated) Klassen, Konstruktoren und Methoden	429

7 Exceptions 433

7.1	Problembereiche einzäunen	433
7.1.1	Exceptions in Java mit try und catch	433
7.1.2	Eine Datei mit RandomAccessFile auslesen	434
7.1.3	Ablauf einer Ausnahmesituation	435
7.1.4	Wiederholung abgebrochener Bereiche	436
7.1.5	throws im Methodenkopf angeben	437
7.1.6	Abschlussbehandlung mit finally	440
7.1.7	Nicht erreichbare catch-Klauseln	443
7.2	Die Klassenhierarchie der Fehler	443
7.2.1	Die Exception-Hierarchie	444
7.2.2	Oberausnahmen auffangen	445
7.2.3	Alles geht als Exception durch	446
7.2.4	RuntimeException muss nicht aufgefangen werden	447
7.2.5	Harte Fehler: Error	448
7.3	Auslösen eigener Exceptions	449
7.3.1	Mit throw Ausnahmen auslösen	449
7.3.2	Neue Exception-Klassen deklarieren	450
7.3.3	Abfangen und Weiterleiten	452
7.3.4	Geschachtelte Ausnahmen	453

7.4	Rückgabewerte bei ausgelösten Ausnahmen	454
7.5	Der Stack Trace	455
7.5.1	Stack Trace aus Throwable	455
7.5.2	Stack Trace aus Thread	456
7.6	Assertions	457
7.6.1	Assertions in Java	457
7.6.2	Assertions in eigenen Programmen nutzen	457
7.6.3	Assertions aktivieren	458
8	Die Funktionsbibliothek	459
8.1	Die Java-Klassenphilosophie	459
8.1.1	Übersicht über die Pakete der Standardbibliothek	459
8.2	Wrapper-Klassen	465
8.2.1	Die Basisklasse Number für numerische Wrapper-Objekte	467
8.2.2	Die Klasse Integer	468
8.2.3	Unterschiedliche Ausgabeformate	470
8.2.4	Autoboxing: Boxing und Unboxing	470
8.2.5	Die Boolean-Klasse	473
8.2.6	Die Klassen Double und Float für Fließkommazahlen	475
8.3	Die Utility-Klasse System und Properties	475
8.3.1	Systemeigenschaften der Java-Umgebung	476
8.3.2	line.separator	477
8.3.3	Browser-Version abfragen	478
8.3.4	Property von der Konsole aus setzen	478
8.3.5	Umgebungsvariablen des Betriebssystems	479
8.3.6	Einfache Zeitmessung und Profiling	480
8.4	Benutzereinstellungen	483
8.4.1	Eine zentrale Registry	483
8.4.2	Einträge einfügen, auslesen und löschen	484
8.4.3	Auslesen der Daten und Schreiben in anderem Format	486
8.4.4	Auf Ereignisse horchen	487
8.5	Klassenlader (Class Loader)	487
8.5.1	Woher die kleinen Klassen kommen	487
8.5.2	Setzen des Klassenpfades	488
8.5.3	Die wichtigsten drei Typen von Klassenladern	489
8.5.4	Der java.lang.ClassLoader	490
8.5.5	Hot Deployment mit dem URL-ClassLoader	491
8.5.6	Das jre/lib/endorsed-Verzeichnis	494
8.5.7	getContextClassLoader() vom Thread	494
8.5.8	Wie heißt die Klasse mit der Methode main()?	495
8.6	Design-Pattern und das Beobachten von Änderungen	496
8.6.1	Design-Pattern	497

8.6.2	Das Beobachter-Pattern (Observer/Observable)	497
8.7	Ausführen externer Programme, Compiler und Skripten	501
8.7.1	ProcessBuilder und Prozesskontrolle mit Process	501
8.7.2	Die Windows-Registry verwenden	504
8.7.3	Einen Browser/E-Mail-Client/Editor aufrufen	505
8.7.4	Ausführen von Skripten	506
8.7.5	Programme mit der Compiler-API übersetzen	508
8.8	Annotationen	510
8.8.1	Annotationstypen @Override, @Deprecated, @SuppressWarnings	510
8.8.2	Common Annotations	512
8.8.3	Annotationen für Web-Services	512
8.8.4	Annotationen für XML-Mapping	512
8.9	Zum Weiterlesen	512

9 Threads und nebenläufige Programmierung 515

9.1	Nebenläufigkeit	515
9.1.1	Threads und Prozesse	515
9.1.2	Wie parallele Programme die Geschwindigkeit steigern können	516
9.2	Threads erzeugen	518
9.2.1	Threads über die Schnittstelle Runnable implementieren	518
9.2.2	Thread mit Runnable starten	519
9.2.3	Der Name eines Threads	521
9.2.4	Die Klasse Thread erweitern	521
9.2.5	Wer bin ich?	524
9.3	Der Ausführer (Executor) kommt	524
9.3.1	Die Schnittstelle Executor	524
9.3.2	Die Thread-Pools	526
9.3.3	Threads mit Rückgabe über Callable	527
9.3.4	Mehrere Callable abarbeiten	529
9.3.5	Mit ScheduledExecutorService wiederholende Ausgaben und Zeitsteuerungen	530
9.4	Die Zustände eines Threads	530
9.4.1	Threads schlafen	531
9.4.2	Das Ende eines Threads	532
9.4.3	UncaughtExceptionHandler für unbehandelte Ausnahmen	533
9.4.4	Einen Thread höflich mit Interrupt beenden	534
9.4.5	Der stop() von außen und die Rettung mit ThreadDeath	536
9.4.6	Ein Rendezvous mit join()	537
9.4.7	Barrier und Austausch mit Exchanger	539
9.4.8	Mit yield() auf Rechenzeit verzichten	540

9.4.9	Arbeit niederlegen und wieder aufnehmen	540
9.4.10	Priorität	540
9.4.11	Der Thread ist ein Dämon	542
9.5	Synchronisation über kritische Abschnitte	544
9.5.1	Gemeinsam genutzte Daten	544
9.5.2	Probleme beim gemeinsamen Zugriff und kritische Abschnitte ...	544
9.5.3	Punkte parallel initialisieren	545
9.5.4	i++ sieht atomar aus, ist es aber nicht	547
9.5.5	Kritische Abschnitte schützen	548
9.5.6	Schützen mit ReentrantLock	549
9.5.7	Synchronisieren mit synchronized	553
9.5.8	Synchronized-Methoden der Klasse StringBuffer	554
9.5.9	Mit synchronized synchronisierte Blöcke	555
9.5.10	Dann machen wir doch gleich alles synchronisiert	556
9.5.11	Look-Freigabe im Fall von Exceptions	557
9.5.12	Mit synchronized nachträglich synchronisieren	558
9.5.13	Monitore sind reentrant – gut für die Geschwindigkeit	559
9.5.14	Synchronisierte Methodenaufrufe zusammenfassen	560
9.5.15	Deadlocks	561
9.5.16	Erkennen von Deadlocks	562
9.6	Synchronisation über Warten und Benachrichtigen	563
9.6.1	Die Schnittstelle Condition	564
9.6.2	Beispiel Erzeuger-Verbraucher-Programm	567
9.6.3	Warten mit wait() und Aufwecken mit notify()	571
9.6.4	Falls der Lock fehlt: IllegalMonitorStateException	573
9.6.5	Semaphore	574
9.7	Atomare Operationen und frische Werte mit volatile	577
9.7.1	Der Modifizierer volatile bei Objekt-/Klassenvariablen	577
9.7.2	Das Paket java.util.concurrent.atomic	578
9.8	Mit dem Thread verbundene Variablen	579
9.8.1	ThreadLocal	580
9.8.2	InheritableThreadLocal	581
9.9	Gruppen von Threads in einer Thread-Gruppe	583
9.9.1	Aktive Threads in der Umgebung	583
9.9.2	Etwas über die aktuelle Thread-Gruppe herausfinden	584
9.9.3	Threads in einer Thread-Gruppe anlegen	586
9.9.4	Methoden von Thread und ThreadGroup im Vergleich	588
9.10	Zeitgesteuerte Abläufe	590
9.10.1	Die Klassen Timer und TimerTask	590
9.10.2	Job-Scheduler Quartz	591
9.11	Einen Abbruch der virtuellen Maschine erkennen	592
9.12	Zum Weiterlesen	593

10 Raum und Zeit	595
10.1 Weltzeit	595
10.2 Wichtige Datum-Klassen im Überblick	596
10.3 Sprachen der Länder	596
10.3.1 Sprachen und Regionen über Locale-Objekte	597
10.4 Übersetzung durch ResourceBundle-Objekte	600
10.4.1 Ressource-Dateien	600
10.4.2 Die Klasse ResourceBundle	601
10.4.3 Ladestrategie für ResourceBundle-Objekte	602
10.5 Zeitzonen	603
10.5.1 Zeitzonen durch die Klasse TimeZone repräsentieren	604
10.6 Die Klasse Date	605
10.6.1 Objekte erzeugen und Methoden nutzen	605
10.7 Calendar und GregorianCalendar	607
10.7.1 Die abstrakte Klasse Calendar	607
10.7.2 Der gregorianische Kalender	608
10.7.3 Ostertage	611
10.7.4 Abfragen und Setzen von Datumselementen	612
10.8 Formatieren der Datumsangaben	617
10.8.1 Mit DateFormat und SimpleDateFormat formatieren	617
10.8.2 Parsen von Datumswerten	623
10.8.3 Parsen und Formatieren ab bestimmten Positionen	624
10.9 Zum Weiterlesen	625
11 Datenstrukturen und Algorithmen	627
11.1 Datenstrukturen und die Collection-API	627
11.1.1 Die Schnittstelle Collection	628
11.1.2 Das erste Programm mit Container-Klassen	629
11.1.3 Die Schnittstelle Iterable und das erweiterte for	630
11.1.4 Generische Datentypen in der Collection-API	631
11.1.5 Generischer Typ bei Iterable und konkreter Typ beim erweiterten for	632
11.1.6 Schnittstellen, die Collection erweitern, und Map	632
11.1.7 Konkrete Container-Klassen	634
11.2 Mit einem Iterator durch die Daten wandern	634
11.2.1 Die Schnittstellen Enumeration und Iterator	634
11.2.2 Der typisierte Iterator	636
11.3 Listen	638
11.3.1 ArrayList oder LinkedList? Speicherung im Feld oder in einer verketteten Liste	638
11.3.2 Die Schnittstelle List	639

11.3.3	ListIterator	641
11.3.4	Beispiel mit List-Methoden	642
11.3.5	ArrayList	645
11.3.6	LinkedList	647
11.3.7	Arrays.asList() und die »echten« Listen	649
11.3.8	toArray() von Collection verstehen – die Gefahr einer Falle erkennen	649
11.4	Vergleichen von Objekten	652
11.4.1	Die Schnittstellen Comparator und Comparable	652
11.4.2	Algorithmen mit Such- und Sortiermöglichkeiten	654
11.4.3	Den größten und kleinsten Wert einer Collection finden	655
11.4.4	Sortieren	657
11.5	Mengen (Sets)	659
11.5.1	HashSet	662
11.5.2	TreeSet – die Menge durch Bäume	662
11.5.3	LinkedHashSet	664
11.6	Stack (Kellerspeicher, Stapel)	664
11.6.1	Die Methoden von Stack	665
11.6.2	Ein Stack ist ein Vector – aha!	665
11.7	Queues (Schlangen)	666
11.7.1	Blockierende Queues und Prioritätswarteschlangen	667
11.8	Assoziative Speicher HashMap und TreeMap	668
11.8.1	Ein Objekt der Klasse HashMap erzeugen	668
11.8.2	Einfügen und Abfragen der Datenstruktur	669
11.8.3	Die Bedeutung von equals(), hashCode() und IdentityHashMap	671
11.8.4	Elemente im Assoziativspeicher müssen unveränderbar bleiben	672
11.8.5	Aufzählen der Elemente	672
11.8.6	Der Gleichheitstest, Hash-Wert und Klon einer Hash-Tabelle	674
11.8.7	Die Arbeitsweise einer Hash-Tabelle	675
11.9	Die Properties-Klasse	677
11.9.1	Properties setzen und lesen	677
11.9.2	Properties verketten	677
11.9.3	Eigenschaften ausgeben	678
11.9.4	Hierarchische Eigenschaften	679
11.9.5	Properties speichern	679
11.9.6	Über die Beziehung zwischen den Klassen Properties und Hashtable	681
11.10	Algorithmen in Collections	681
11.10.1	Datenmanipulation: Umdrehen, Füllen, Kopieren	682
11.10.2	Mit der Halbierungssuche nach Elementen fahnden	683

11.10.3	Nicht-änderbare Datenstrukturen	684
11.10.4	Häufigkeit eines Elements	684
11.10.5	nCopies()	684
11.10.6	Singletons	685
11.11	Synchronisation der Datenstrukturen	686
11.11.1	Lock-Free-Algorithmen aus java.util.concurrent	686
11.11.2	Wrapper zur Synchronisation	686
11.11.3	CopyOnWriteArrayList und CopyOnWriteArraySet	687
11.12	Die abstrakten Basisklassen für Container	687
11.12.1	Optionale Methoden	688
11.13	Die Klasse BitSet für Bitmengen	689
11.13.1	Ein BitSet anlegen, füllen und erfragen	689
11.13.2	Mengenorientierte Operationen	690
11.13.3	Funktionsübersicht	691
11.13.4	Primzahlen in einem BitSet verwalten	692
12	Dateien und Datenströme	693
12.1	Datei und Verzeichnis	694
12.1.1	Dateien und Verzeichnisse mit der Klasse File	694
12.1.2	Verzeichnis oder Datei? Existiert es?	696
12.1.3	Verzeichnis- und Dateieigenschaften/-attribute	697
12.1.4	Wurzelverzeichnis, Laufwerknamen, Plattenspeicher	699
12.1.5	Umbenennen und Verzeichnisse anlegen	701
12.1.6	Verzeichnisse listen und Dateien filtern	702
12.1.7	Dateien berühren, neue Dateien anlegen, temporäre Dateien	704
12.1.8	Dateien und Verzeichnisse löschen	706
12.1.9	Verzeichnisse nach Dateien rekursiv durchsuchen	707
12.1.10	URL- und URI-Objekte aus einem File-Objekt ableiten	708
12.1.11	Mit Locking Dateien sperren	709
12.1.12	Sicherheitsprüfung	709
12.1.13	Mime-Typen mit dem JavaBeans Activation Framework (JAF)	709
12.1.14	Zugriff auf SMB-Server mit jcIFS	710
12.2	Dateien mit wahlfreiem Zugriff	711
12.2.1	Ein RandomAccessFile zum Lesen und Schreiben öffnen	712
12.2.2	Aus dem RandomAccessFile lesen	712
12.2.3	Schreiben	714
12.2.4	Die Länge des RandomAccessFile	715
12.2.5	Hin und her in der Datei	715
12.2.6	Wahlfreier Zugriff und Pufferung mit Unified I/O	716
12.3	Stream-Klassen und Reader/Writer am Beispiel von Dateien	717
12.3.1	Mit dem FileWriter Texte in Dateien schreiben	718
12.3.2	Zeichen mit der Klasse FileReader lesen	719

12.3.3	Kopieren mit FileOutputStream und FileInputStream	720
12.3.4	Das FileDescriptor-Objekt	723
12.4	Basisklassen für die Ein-/Ausgabe	724
12.4.1	Die abstrakten Basisklassen	724
12.4.2	Übersicht über Ein-/Ausgabeklassen	724
12.4.3	Die abstrakte Basisklasse OutputStream	725
12.4.4	Die Schnittstellen Closeable und Flushable	727
12.4.5	Ein Datenschlucke	728
12.4.6	Die abstrakte Basisklasse InputStream	728
12.4.7	Ressourcen wie Grafiken aus dem Klassenpfad und aus Jar-Archiven laden	729
12.4.8	Ströme mit SequenceInputStream zusammensetzen	730
12.4.9	Die abstrakte Basisklasse Writer	732
12.4.10	Die Schnittstelle Appendable	733
12.4.11	Die abstrakte Basisklasse Reader	734
12.5	Formatierte Textausgaben	736
12.5.1	Die Klassen PrintWriter und PrintStream	736
12.5.2	System.out, System.err und System.in	740
12.5.3	Geschützte Passwort-Eingaben mit der Klasse Console	743
12.6	Schreiben und Lesen aus Strings und Byte-Feldern	743
12.6.1	Mit dem StringWriter ein String-Objekt füllen	744
12.6.2	CharArrayWriter	744
12.6.3	StringReader und CharArrayReader	745
12.6.4	Mit ByteArrayOutputStream in ein Byte-Feld schreiben	746
12.6.5	Mit ByteArrayInputStream aus einem Byte-Feld lesen	747
12.7	Datenströme filtern und verketten	747
12.7.1	Writer als Filter verketten	748
12.7.2	Gepufferte Unicode-Ausgabe mit BufferedWriter	748
12.7.3	Gepufferte Unicode-Eingabe mit BufferedReader	750
12.7.4	LineNumberReader zählt automatisch Zeilen mit	752
12.7.5	Daten mit der Klasse PushbackReader zurücklegen	752
12.7.6	DataOutputStream/DataInputStream	755
12.7.7	Basisklassen für Filter	755
12.7.8	Die Basisklasse FilterWriter	756
12.7.9	Ein LowerCaseWriter	757
12.7.10	Eingaben mit der Klasse FilterReader filtern	758
12.8	Vermittler zwischen Byte-Streams und Unicode-Strömen	759
12.8.1	Datenkonvertierung durch den OutputStreamWriter	759
12.8.2	Automatische Konvertierungen mit dem InputStreamReader	760
12.9	Kommunikation zwischen Threads mit Pipes	761
12.9.1	PipedOutputStream und PipedInputStream	762
12.9.2	PipedWriter und PipedReader	763

12.10	Datenkompression	765
12.10.1	Java-Umgebung beim Komprimieren und Zusammenpacken	766
12.10.2	Datenströme komprimieren	766
12.10.3	Zip-Archive	770
12.10.4	Jar-Archive	777
12.11	Prüfsummen	777
12.11.1	Die Schnittstelle Checksum	777
12.11.2	Die Klasse CRC32	778
12.11.3	Die Adler32-Klasse	780
12.12	Persistente Objekte und Serialisierung	780
12.12.1	Objekte mit der Standard-Serialisierung speichern	781
12.12.2	Objekte über die Standard-Serialisierung lesen	784
12.12.3	Die Schnittstelle Serializable	785
12.12.4	Nicht serialisierbare Attribute aussparen	786
12.12.5	Das Abspeichern selbst in die Hand nehmen	788
12.12.6	Tiefe Objektkopien	790
12.12.7	Versionenverwaltung und die SUID	792
12.12.8	Wie die ArrayList serialisiert	794
12.12.9	Probleme mit der Serialisierung	795
12.12.10	Serialisieren in XML-Dateien	795
12.12.11	JavaBeans Persistence	796
12.12.12	XStream	798
12.13	Tokenizer	798
12.13.1	StreamTokenizer	798
12.13.2	CSV-(Comma Separated Values-)Dateien verarbeiten	801

13	Die eXtensible Markup Language (XML)	803
13.1	Auszeichnungssprachen	803
13.1.1	Die Standard Generalized Markup Language (SGML)	803
13.1.2	Extensible Markup Language (XML)	804
13.2	Eigenschaften von XML-Dokumenten	804
13.2.1	Elemente und Attribute	804
13.2.2	Beschreibungssprache für den Aufbau von XML-Dokumenten ...	806
13.2.3	Schema – eine Alternative zu DTD	809
13.2.4	Namensraum (Namespace)	811
13.2.5	XML-Applikationen	813
13.3	Die Java-APIs für XML	813
13.3.1	Das Document Object Model (DOM)	814
13.3.2	Simple API for XML Parsing (SAX)	814
13.3.3	Pull-API StAX	814
13.3.4	Java Document Object Model (JDOM)	814

13.3.5	JAXP als Java-Schnittstelle zu XML	815
13.3.6	DOM-Bäume einlesen mit JAXP	815
13.4	Serielle Verarbeitung mit StAX	816
13.4.1	Unterschiede der Verarbeitungsmodelle	816
13.4.2	XML-Dateien mit dem Cursor-Verfahren lesen	818
13.4.3	XML-Dateien mit dem Iterator-Verfahren verarbeiten	820
13.4.4	Mit Filtern arbeiten	821
13.4.5	XML-Dokumente schreiben	822
13.5	Serielle Verarbeitung von XML mit SAX	824
13.5.1	Schnittstellen von SAX	825
13.5.2	SAX-Parser erzeugen	826
13.5.3	Die wichtigsten Methoden der Schnittstelle ContentHandler	826
13.5.4	ErrorHandler und EntityResolver	828
13.6	XML-Dateien mit JDOM verarbeiten	829
13.6.1	JDOM beziehen	829
13.6.2	Paketübersicht	830
13.6.3	Die Document-Klasse	831
13.6.4	Eingaben aus der Datei lesen	832
13.6.5	Das Dokument im XML-Format ausgeben	833
13.6.6	Der Dokumenttyp	833
13.6.7	Elemente	834
13.6.8	Zugriff auf Elementinhalte	836
13.6.9	Liste mit Unterelementen erzeugen	838
13.6.10	Neue Elemente einfügen und ändern	839
13.6.11	Attributinhalte lesen und ändern	841
13.6.12	XPath	844
13.7	Transformationen mit XSLT	847
13.7.1	Templates und XPath als Kernelemente von XSLT	847
13.7.2	Umwandlung von XML-Dateien mit JDOM und JAXP	849
13.8	Java Architecture for XML Binding (JAXB)	850
13.9	HTML-Dokumente einlesen	852
13.10	Zum Weiterlesen	853

14	Grafische Oberflächen mit Swing	855
14.1	Das Abstract-Window-Toolkit und Swing	855
14.1.1	Java Foundation Classes	855
14.1.2	Was Swing vom AWT unterscheidet	857
14.1.3	Peer-Klassen und Lightweight-Komponenten	857
14.1.4	Die Klasse Toolkit	858
14.2	Fenster unter grafischen Oberflächen	859
14.2.1	Swing-Fenster darstellen	859
14.2.2	Fenster schließbar machen – setDefaultCloseOperation()	860

14.2.3	AWT-Fenster darstellen	861
14.2.4	Sichtbarkeit des Fensters	861
14.2.5	Größe und Position des Fensters verändern	862
14.2.6	Unterklassen der Fenster-Klassen bilden	863
14.2.7	Fenster- und Dialog-Dekoration	864
14.2.8	Dynamisches Layout während einer Größenänderung	864
14.3	Beschriftungen über die Klasse JLabel	864
14.3.1	Mehrzeiliger Text, HTML in der Darstellung	868
14.4	Es tut sich was – Ereignisse beim AWT	868
14.4.1	Die Klasse AWTEvent	868
14.4.2	Events auf verschiedenen Ebenen	869
14.4.3	Ereignisquellen und Horcher (Listener)	871
14.4.4	Listener implementieren	872
14.4.5	Listener bei dem Ereignisauslöser anmelden/abmelden	874
14.4.6	Aufrufen der Listener im AWT-Event-Thread	875
14.4.7	Adapterklassen nutzen	875
14.4.8	Innere Mitgliedsklassen und innere anonyme Klassen	877
14.4.9	Generic Listener	878
14.5	Schaltfläche	879
14.5.1	Der JButton	879
14.5.2	Der aufmerksame ActionListener	880
14.5.3	AbstractButton	882
14.5.4	JToggleButton	885
14.6	Icon und ImageIcon für Bilder auf Swing-Komponenten	885
14.6.1	Die Schnittstelle Icon	887
14.6.2	Was Icon und Image verbindet	888
14.7	JComponent und Component als Basis aller Komponenten	889
14.7.1	Ereignisse jeder Komponente	889
14.7.2	Die Größe einer Komponente	892
14.7.3	Die Position einer Komponente	892
14.7.4	Properties	893
14.7.5	Fokus	893
14.7.6	Komponenten-Ereignisse	894
14.7.7	Hinzufügen von Komponenten	896
14.7.8	Zeichnen von Komponenten und die Undurchsichtigkeit	896
14.7.9	Tooltips	897
14.7.10	Rahmen (Border)	898
14.8	Container	900
14.8.1	JPanel	901
14.8.2	JScrollPane	901
14.8.3	JTabbedPane	902
14.8.4	JSplitPane	903

14.9	Alles Auslegungssache: die Layoutmanager	904
14.9.1	Übersicht über Layoutmanager	904
14.9.2	Zuweisen eines Layoutmanagers	905
14.9.3	Im Fluss mit FlowLayout	906
14.9.4	Mit BorderLayout in allen Himmelsrichtungen	908
14.9.5	Rasteranordnung mit GridLayout	910
14.9.6	Der GridBagLayout-Manager	912
14.9.7	Null-Layout	916
14.9.8	BoxLayout	917
14.9.9	Weitere Layoutmanager	918
14.10	Rollbalken und Slider	918
14.10.1	Der Schieberegler JSlider	918
14.10.2	Der Rollbalken JScrollPane	920
14.11	Kontrollfelder, Optionsfelder, Kontrollfeldgruppen	924
14.11.1	Kontrollfelder (JCheckBox)	924
14.11.2	Ereignisse über ItemListener	926
14.11.3	Sich gegenseitig ausschließende Optionen (JRadioButton)	928
14.12	Der Fortschrittsbalken JProgressBar	930
14.13	Menüs und Symbolleisten	932
14.13.1	Die Menüleisten und die Einträge	932
14.13.2	Menüeinträge definieren	934
14.13.3	Einträge durch Action-Objekte beschreiben	935
14.13.4	Mnemonics und Shortcuts (Accelerator)	937
14.13.5	Symbolleisten alias Toolbars	939
14.13.6	Popup-Menüs	941
14.13.7	System-Tray nutzen	945
14.14	Das Konzept des Model-View-Controllers	946
14.15	Auswahlmenüs, Listen und Spinner	948
14.15.1	Das Auswahlmenü JComboBox	948
14.15.2	Zuordnung einer Taste mit einem Eintrag	951
14.15.3	Datumsauswahl	952
14.15.4	Die JList	953
14.15.5	JSpinner	956
14.16	Texteingabefelder	957
14.16.1	Text in einer Eingabezeile	958
14.16.2	Die Oberklasse der JText-Komponenten: JTextComponent	959
14.16.3	JPasswordField	960
14.16.4	Validierende Eingabefelder	960
14.16.5	Mehrzeilige Textfelder	961
14.16.6	Die Editor-Klasse JEditorPane	964
14.17	Bäume mit JTree-Objekten	966
14.17.1	Selektionen bemerkern	968

14.18	Tabellen mit JTable	969
14.18.1	Ein eigenes Tabellen-Model	970
14.18.2	AbstractTableModel	971
14.18.3	DefaultTableModel	974
14.18.4	Ein eigener Renderer für Tabellen	975
14.18.5	Zell-Editoren	979
14.18.6	Größe und Umrandung der Zellen	980
14.18.7	Spalteninformationen	980
14.18.8	Tabellenkopf von Swing-Tabellen	981
14.18.9	Selektionen einer Tabelle	981
14.18.10	Automatisches Sortieren und Filtern mit RowSorter	982
14.18.11	Ein professionelles Tabellenlayout mit JGrid	983
14.19	JRootPane, JLayeredPane und JDesktopPane	983
14.19.1	JRootPane	984
14.19.2	JLayeredPane	984
14.19.3	JDesktopPane und die Kinder JInternalFrame	984
14.20	Dialoge und Window-Objekte	986
14.20.1	JWindow und JDialog	987
14.20.2	Modal oder nicht-modal	987
14.20.3	Standarddialoge mit JOptionPane	987
14.20.4	Der Farbauswahldialog JColorChooser	989
14.20.5	Der Dateiauswahldialog	991
14.21	Flexibles Java-Look & Feel	995
14.21.1	L&F global setzen	995
14.21.2	UIManager	995
14.21.3	Verbessern des Aussehens unter Windows mit JGoodies Looks	997
14.21.4	Swing-Beschriftungen eine andere Sprache geben	997
14.22	Die Zwischenablage (Clipboard)	998
14.22.1	Clipboard-Objekte	998
14.22.2	Auf den Inhalt zugreifen mit Transferable	999
14.22.3	DataFlavor ist das Format der Daten in der Zwischenablage	1000
14.22.4	Einfügungen in der Zwischenablage erkennen	1002
14.22.5	Drag & Drop	1002
14.23	Undo durchführen	1003
14.24	AWT, Swing und die Threads	1005
14.24.1	Swing ist nicht thread-sicher	1006
14.24.2	Swing-Elemente mit invokeLater() und invokeAndWait() bedienen	1008
14.24.3	SwingWorker	1009
14.24.4	Eigene Ereignisse in die Queue setzen	1011
14.24.5	Auf alle Ereignisse hören	1012

14.25	Selbst definierte Cursor	1012
14.26	Benutzerinteraktionen automatisieren	1014
14.26.1	Automatisch in die Tasten hauen	1014
14.26.2	Mausoperationen	1015
14.26.3	Methoden zur Zeitsteuerung	1016
14.26.4	Screenshots	1016
14.26.5	Funktionsweise und Beschränkungen	1017
14.26.6	MouseInfo und PointerInfo	1017
14.27	Zeitliches Ausführen mit dem javax.swing.Timer	1017
14.28	Alternativen zu AWT und Swing	1018
14.28.1	XML-Beschreibungen der Oberfläche: Swixml, XUL/Luxor	1018
14.28.2	SWT	1019
14.29	Zum Weiterlesen	1019

15 Grafikprogrammierung 1021

15.1	Grundlegendes zum Zeichnen	1021
15.1.1	Die paint()-Methode für das AWT-Frame	1021
15.1.2	Zeichen von Inhalten mit JFrame	1023
15.1.3	Auffordern zum Neuzeichnen mit repaint()	1024
15.1.4	Grundbegriffe: Koordinaten, Punkte, Pixel	1024
15.1.5	Die ereignisorientierte Programmierung ändert Fensterinhalte ...	1025
15.1.6	Java 2D-API	1026
15.2	Einfache Zeichenfunktionen	1027
15.2.1	Linien	1027
15.2.2	Rechtecke	1028
15.2.3	Ovale und Kreisbögen	1028
15.2.4	Polygone und Polylines	1029
15.3	Zeichenketten schreiben und Fonts	1032
15.3.1	Zeichenfolgen schreiben	1032
15.3.2	Die Font-Klasse	1032
15.3.3	Einen neuen Font aus einem gegebenen Font ableiten	1034
15.3.4	Zeichensätze des Systems ermitteln	1035
15.3.5	Neue TrueType-Fonts in Java nutzen	1036
15.3.6	Font-Metadaten durch FontMetrics	1036
15.4	Geometrische Objekte	1039
15.4.1	Die Schnittstelle Shape	1041
15.4.2	Kreisförmiges	1042
15.4.3	Kurviges	1042
15.4.4	Area und die konstruktive Flächengeometrie	1042
15.4.5	Pfade	1043
15.5	Farben und die Paint-Schnittstelle, Linientypen	1046
15.5.1	Zufällige Farbblöcke zeichnen	1046

15.5.2	Farben mit der Klasse Color	1048
15.5.3	Die Farben des Systems über SystemColor	1051
15.5.4	Composite und Xor	1054
15.5.5	Dicke und Art der Linien von Formen bestimmen	1055
15.6	Bilder	1058
15.6.1	Eine Übersicht über die Bilder-Bibliotheken	1059
15.6.2	Bilder mit ImageIO lesen	1060
15.6.3	Ein Bild zeichnen	1062
15.6.4	Programm-Icon/Fenster-Icon setzen	1065
15.6.5	Splash-Screen	1065
15.6.6	Bilder im Speicher erzeugen	1066
15.6.7	Kein Flackern durch Double-Buffering	1067
15.6.8	Bilder skalieren	1069
15.6.9	Schreiben mit ImageIO	1070
15.6.10	Java Image Management Interface (JIMI)	1074
15.6.11	Pixel für Pixel auslesen und schreiben	1074
15.6.12	Asynchrones Laden mit getImage() und dem MediaTracker	1076
15.6.13	VolatileImage	1077
15.7	Clipping-Operationen	1078
15.8	Zeichenhinweise durch RenderingHints	1081
15.9	Transformationen mit einem AffineTransform-Objekt	1082
15.10	Drucken	1084
15.10.1	Drucken mit dem einfachen Ansatz	1084
15.10.2	Ein PrintJob	1085
15.10.3	Drucken der Inhalte	1086
15.10.4	Komponenten drucken	1088
15.10.5	Den Drucker am Parallelport ansprechen	1089
15.10.6	Bekannte Drucker	1090
15.11	Grafikverarbeitung ohne grafische Oberfläche	1090
15.11.1	Xvfb-Server	1091
15.11.2	Pure Java AWT Toolkit (PJA)	1091
15.12	Zum Weiterlesen	1091
16	Das Netz	1093
16.1	Grundlegende Begriffe	1093
16.1.1	Internet-Standards und RFC	1094
16.2	URI und URL	1094
16.2.1	URI	1094
16.2.2	Die Klasse URL	1095
16.2.3	Informationen über eine URL	1097
16.2.4	Der Zugriff auf die Daten über die Klasse URL	1099
16.2.5	Verbindungen durch einen Proxy-Server	1101

16.3	Die Klasse URLConnection	1101
16.3.1	Methoden und Anwendung von URLConnection	1102
16.3.2	Protokoll- und Content-Handler	1104
16.3.3	Im Detail: vom URL zu URLConnection	1105
16.3.4	Der Protokoll-Handler für Jar-Dateien	1106
16.3.5	Verbindungen und Proxy-Authentifizierung mit Basic Authentication	1107
16.4	Mit GET und POST Daten übergeben	1109
16.4.1	Kodieren der Parameter für Serverprogramme	1109
16.4.2	Eine Suchmaschine ansprechen	1111
16.5	Host- und IP-Adressen	1111
16.5.1	Lebt der Rechner?	1113
16.5.2	Das Netz ist Klasse	1114
16.5.3	IP-Adresse des lokalen Hosts	1114
16.6	NetworkInterface	1115
16.7	Mit dem Socket zum Server	1116
16.7.1	Das Netzwerk ist der Computer	1116
16.7.2	Sockets	1117
16.7.3	Eine Verbindung zum Server aufbauen	1117
16.7.4	Server unter Spannung: die Ströme	1118
16.7.5	Die Verbindung wieder abbauen	1119
16.7.6	Informationen über den Socket	1119
16.7.7	Reine Verbindungsdaten über SocketAddress	1121
16.8	Client/Server-Kommunikation	1122
16.8.1	Warten auf Verbindungen	1122
16.8.2	Ein Multiplikationsserver	1123
16.8.3	Blockierendes Lesen	1126
16.8.4	Von außen erreichbar sein	1127
16.9	Apache Jakarta Commons HttpClient und Net	1128
16.9.1	Jakarta Commons HttpClient	1128
16.9.2	Jakarta Commons Net	1129
16.10	Arbeitsweise eines Webservers	1129
16.10.1	Das Hypertext Transfer Protocol (HTTP)	1129
16.10.2	Anfragen an den Server	1130
16.10.3	Die Antworten vom Server	1133
16.10.4	Webserver mit com.sun.net.httpserver.HttpServer	1136
16.11	Datagram-Sockets	1137
16.11.1	Die Klasse DatagramSocket	1139
16.11.2	Datagramme und die Klasse DatagramPacket	1140
16.11.3	Auf ein hereinkommendes Paket warten	1141
16.11.4	Ein Paket zum Senden vorbereiten	1142
16.11.5	Methoden der Klasse DatagramPacket	1143
16.11.6	Das Paket senden	1144

16.12	E-Mail	1145
16.12.1	Wie eine E-Mail um die Welt geht	1145
16.12.2	Das Simple Mail Transfer Protocol und RFC 822	1145
16.12.3	POP (Post Office Protocol)	1146
16.12.4	Die JavaMail API	1146
16.12.5	E-Mails versenden	1147
16.12.6	MimeMultipart-Nachrichten schicken	1148
16.12.7	E-Mails mittels POP3 abrufen	1149
16.12.8	Ereignisse und Suchen	1150
16.13	Tiefer liegende Netzwerkeigenschaften	1151
16.13.1	Internet Control Message Protocol (ICMP)	1151
16.13.2	MAC-Adresse	1152
16.14	Zum Weiterlesen	1153

17 JavaServer Pages und Servlets 1155

17.1	Dynamisch generierte Webseiten	1155
17.1.1	Was sind Servlets?	1156
17.1.2	Was sind JavaServer Pages?	1156
17.2	Servlets und JSPs mit Tomcat entwickeln	1157
17.2.1	Servlet-Container	1157
17.2.2	Entwicklung der Servlet/JSP-Spezifikationen	1158
17.2.3	Webserver mit Servlet-Funktionalität	1158
17.2.4	Tomcat	1159
17.2.5	Ablageort für eigene JSP-Seiten	1159
17.2.6	Webapplikationen	1160
17.2.7	Zuordnung von Webapplikationen zu physikalischen Verzeichnissen	1161
17.2.8	Mit dem WTP ein Web-Projekt entwickeln	1161
17.3	JSP-Konstrukte	1163
17.3.1	Scriptlets	1164
17.3.2	Ausdrücke	1164
17.3.3	Deklarationen	1165
17.3.4	Kommentare	1165
17.3.5	Quoting	1166
17.3.6	Entsprechende XML-Tags	1166
17.4	Implizite Objekte	1166
17.5	Formulardaten	1167
17.6	Auf Beans zurückgreifen	1169
17.6.1	Die Bean-Klasse	1169
17.6.2	Beans in JSP-Seiten anlegen	1170
17.6.3	Attribute setzen und erfragen	1170
17.6.4	Der schnelle Zugriff auf Parameter	1171

17.7	Mit HttpServletRequest an die geschickten Browser-Daten	1171
17.7.1	Verarbeiten der Header	1172
17.7.2	Hilfsfunktion im Umgang mit Headern	1172
17.7.3	Übersicht der Browser-Header	1173
17.8	Das HttpServletResponse-Objekt	1174
17.8.1	Automatisches Neuladen	1174
17.8.2	Seiten umlenken	1174
17.9	JSP-Direktiven	1175
17.9.1	page-Direktiven im Überblick	1176
17.9.2	Die include-Direktive	1177
17.9.3	Mit JSPs Bilder generieren	1178
17.10	Aktionen	1179
17.10.1	Die Aktion include	1179
17.10.2	Die Aktion forward	1179
17.10.3	Die Aktion plugin	1179
17.11	Kleine Kekse: die Klasse Cookies	1180
17.11.1	Cookies erzeugen und setzen	1181
17.11.2	Cookies vom Servlet einlesen	1182
17.11.3	Cookie-Status ändern	1183
17.11.4	Langlebige Cookies	1184
17.12	Sitzungsverfolgung (Session Tracking)	1184
17.12.1	Das mit einer Sitzung verbundene Objekt HttpSession	1185
17.12.2	Werte mit einer Sitzung assoziieren und auslesen	1185
17.12.3	URL-Rewriting	1186
17.12.4	Zusätzliche Informationen	1187
17.13	JSTL und weitere Tag-Libraries	1189
17.13.1	Standard Tag Library (JSTL)	1189
17.13.2	Jakarta Taglibs Project	1191
17.14	Servlets	1191
17.14.1	Servlets compilieren	1192
17.14.2	Die Servlets in das classes-Verzeichnis bringen	1192
17.14.3	Servlet-Mapping	1193
17.15	Der Lebenszyklus eines Servlets	1195
17.15.1	Abfragen bei service()	1196
17.15.2	Mehrere Anfragen beim Servlet und die Thread-Sicherheit	1197
17.15.3	Das Ende eines Servlets	1197
17.16	Das HttpServletResponse-Objekt	1197
17.16.1	Wir generieren eine Webseite	1197
17.16.2	Binärdaten senden	1199
17.16.3	Noch mehr über Header, die der Server setzt	1200
17.17	Objekte und Dateien per POST verschicken	1201
17.17.1	Datei-Upload	1202

17.18	Servlets und Sessions	1203
17.19	Weiterleiten und Einbinden von Servlet-Inhalten	1204
17.20	Inter-Servlet-Kommunikation	1205
17.20.1	Daten zwischen Servlets teilen	1205
17.21	Internationalisierung	1205
17.21.1	Die Länderkennung des Anfragers auslesen	1206
17.21.2	Länderkennung für die Ausgabe setzen	1206
17.21.3	Westeuropäische Texte senden	1206
17.22	Tomcat: Spezielles	1208
17.22.1	Tomcat als Service unter Windows NT ausführen	1208
17.23	Zum Weiterlesen	1208

18 Verteilte Programmierung mit RMI und Web-Services 1211

18.1	Entfernte Objekte und Methoden	1211
18.1.1	Stellvertreter helfen bei entfernten Methodenaufrufen	1211
18.1.2	Standards für entfernte Objekte	1213
18.2	Java Remote Method Invocation	1213
18.2.1	Zusammenspiel von Server, Registry und Client	1213
18.2.2	Wie die Stellvertreter die Daten übertragen	1213
18.2.3	Probleme mit entfernten Methoden	1214
18.2.4	Nutzen von RMI bei Middleware-Lösungen	1216
18.2.5	Zentrale Klassen und Schnittstellen	1216
18.2.6	Entfernte und lokale Objekte im Vergleich	1217
18.3	Auf der Serverseite	1217
18.3.1	Entfernte Schnittstelle deklarieren	1217
18.3.2	Remote-Objekt-Implementierung	1218
18.3.3	Stellvertreterobjekte erzeugen	1219
18.3.4	Der Namensdienst (Registry)	1219
18.3.5	Remote-Objekt-Implementierung exportieren und beim Namensdienst anmelden	1220
18.3.6	Einfaches Logging	1222
18.3.7	Aufräumen mit dem DGC	1222
18.4	Auf der Clientseite	1223
18.5	Entfernte Objekte übergeben und laden	1224
18.5.1	Klassen vom RMI-Klassenlader nachladen	1224
18.6	Der Server startet die Registry selbst	1225
18.7	Weitere Eigenschaften von RMI	1226
18.7.1	RMI und CORBA	1226
18.7.2	RMI über HTTP getunnelt	1226
18.7.3	Automatische Remote-Objekt-Aktivierung	1227
18.8	Daily Soap	1227
18.8.1	SOAP-Protokoll	1228

18.8.2	Die technische Realisierung	1228
18.8.3	SOAP-Implementierungen	1229
18.8.4	@WebService in Java 6	1229
18.8.5	Einen Web-Service definieren	1229
18.8.6	Web-Services veröffentlichen	1230
18.8.7	Einen JAX-WS-Client implementieren	1231
18.9	Java Message Service (JMS)	1233
18.10	Zum Weiterlesen	1233

19 Applets, Midlets und Sound 1235

19.1	Applets in der Wiege von Java	1235
19.1.1	(J)Applet und Applikationen	1235
19.1.2	Das erste Hallo-Applet	1235
19.1.3	Die Zyklen eines Applets	1237
19.1.4	Parameter an das Applet übergeben	1237
19.1.5	Wie das Applet den Browser-Inhalt ändern kann	1239
19.1.6	Den Ursprung des Applets erfragen	1239
19.1.7	Datenaustausch zwischen Applets	1240
19.1.8	Was ein Applet alles darf	1243
19.2	Fehler in Applets finden	1244
19.2.1	Ist Java im Browser aktiviert?	1244
19.2.2	Läuft das Applet unter Netscape oder Microsoft Explorer?	1244
19.2.3	Datenaustausch zwischen Applets und JavaScript	1246
19.3	Musik in einem Applet und in Applikationen	1246
19.3.1	Die Arbeit mit AudioClip	1247
19.3.2	Java Sound API	1247
19.4	Webstart	1248
19.5	Java Micro Edition	1249
19.5.1	Konfigurationen	1249
19.5.2	Profile	1250
19.6	Zum Weiterlesen	1252

20 Datenbankmanagement mit JDBC 1253

20.1	Das relationale Modell	1253
20.2	Datenbanken und Tools	1254
20.2.1	HSQLDB	1254
20.2.2	Weitere Datenbanken	1255
20.2.3	Eclipse-Plugins zum Durchschauen von Datenbanken	1256
20.3	JDBC und Datenbanktreiber	1258
20.3.1	Treibertypen	1259
20.3.2	JDBC-Versionen	1260



20.4	Eine Beispielabfrage	1261
20.4.1	Schritte zur Datenbankabfrage	1261
20.4.2	Client für HSQLDB-Datenbank	1262
20.5	Mit Java an eine Datenbank andocken	1263
20.5.1	Der Treiber-Manager	1264
20.5.2	Den Treiber laden	1264
20.5.3	Eine Aufzählung aller Treiber	1265
20.5.4	Log-Informationen	1266
20.5.5	Verbindung zur Datenbank auf- und abbauen	1267
20.5.6	DataSource	1270
20.5.7	Gepoolte Verbindungen	1273
20.6	Datenbankabfragen	1273
20.6.1	Abfragen über das Statement-Objekt	1273
20.6.2	Ergebnisse einer Abfrage in ResultSet	1275
20.6.3	Java und SQL-Datentypen	1276
20.6.4	Unicode in der Spalte korrekt auslesen	1279
20.6.5	wasNull() bei ResultSet	1279
20.6.6	Wie viele Zeilen hat ein ResultSet?	1280
20.7	Die Ausnahmen bei JDBC	1280
20.8	Elemente einer Datenbank hinzufügen und aktualisieren	1281
20.8.1	Batch-Updates	1282
20.9	ResultSets in Bohnen durch RowSet	1283
20.9.1	Die Schnittstelle RowSet	1283
20.9.2	Implementierungen von RowSet	1284
20.9.3	Der Typ CachedRowSet	1284
20.9.4	Der Typ WebRowSet	1285
20.10	Vorbereitete Anweisungen (Prepared Statements)	1287
20.10.1	PreparedStatement-Objekte vorbereiten	1288
20.10.2	Werte für die Platzhalter eines PreparedStatement	1289
20.11	Transaktionen	1290
20.12	Die LOBs (Large Objects)	1290
20.12.1	Einen BLOB besorgen	1291
20.13	Metadaten	1291
20.13.1	Metadaten über die Tabelle	1291
20.13.2	Informationen über die Datenbank	1294
20.14	Einführung in SQL	1295
20.14.1	Ein Rundgang durch SQL-Anfragen	1296
20.14.2	Datenabfrage mit der Data Query Language (DQL)	1297
20.14.3	Tabellen mit der Data Definition Language (DDL) anlegen	1299
20.15	Zum Weiterlesen	1300

21 Reflection und Annotationen	1301
21.1 Metadaten	1301
21.1.1 Metadaten durch Java-Doc Tags	1301
21.1.2 XDoclet	1302
21.2 Metadaten der Klassen mit dem Class-Objekt	1302
21.2.1 An ein Class-Objekt kommen	1303
21.2.2 Was das Class-Objekt beschreibt	1305
21.2.3 Der Name der Klasse	1306
21.2.4 Die Arbeit auf dem Feld	1308
21.2.5 instanceof mit Class-Objekten	1309
21.2.6 Oberklassen finden	1310
21.2.7 Implementierte Interfaces einer Klasse oder eines Interfaces	1310
21.2.8 Modifizierer und die Klasse Modifier	1311
21.2.9 Die Attribute einer Klasse	1313
21.2.10 Methoden einer Klasse erfragen	1315
21.2.11 Konstruktoren einer Klasse	1319
21.2.12 Annotationen	1321
21.3 Objekte manipulieren	1321
21.3.1 Objekte erzeugen	1321
21.3.2 Die Belegung der Variablen erfragen	1322
21.3.3 Eine generische <code>toString()</code> -Funktion	1324
21.3.4 Variablen setzen	1326
21.3.5 Private Attribute ändern	1328
21.4 Methoden aufrufen	1328
21.4.1 Statische Methoden aufrufen	1330
21.4.2 Dynamische Methodenaufrufe bei festen Methoden beschleunigen	1330
21.5 Informationen und Identifizierung von Paketen	1332
21.5.1 Geladene Pakete	1332
21.6 Annotationen	1332
21.6.1 Neue Annotationen definieren	1332
21.6.2 Annotationen mit genau einem Element	1333
21.6.3 Beliebige Schlüssel-Werte-Paare	1334
21.6.4 Vorbelegte Elemente	1337
21.6.5 Annotieren von Annotationstypen	1338
21.6.6 Annotationen zur Laufzeit ausgelesen	1340
21.6.7 Mögliche Nachteile von Annotationen	1342
22 Komponenten durch Bohnen	1345
22.1 Grundlagen der Komponententechnik	1345
22.1.1 Brauchen wir überhaupt Komponenten?	1345

22.1.2	Visuelle und nichtvisuelle Komponenten	1346
22.1.3	Komponenten-Technologien von Microsoft	1346
22.2	Worauf JavaBeans basieren	1347
22.3	Eigenschaften	1348
22.3.1	Einfache Eigenschaften	1349
22.3.2	Boolesche Eigenschaften	1349
22.3.3	Indizierte Eigenschaften	1349
22.4	Ereignisse	1351
22.4.1	Multicast und Unicast	1351
22.4.2	Namenskonvention	1351
22.5	Weitere Eigenschaften	1354
22.5.1	Gebundene Eigenschaften	1354
22.5.2	Anwendung von PropertyChange bei AWT-Komponenten	1357
22.5.3	Veto-Eigenschaften – dagegen!	1357
23	Logging und Monitoring	1359
23.1	Die Logging-API	1359
23.2	Überwachen von Systemzuständen	1361
23.3	MBean-Typen, MBean-Server und weitere Begriffe	1362
23.3.1	MXBeans des Systems	1363
23.4	Geschwätzige Programme und JConsole	1365
23.5	Der MBeanServer	1367
23.6	Eine eigene Standard-MBean	1368
23.7	Zum Weiterlesen	1370
24	Sicherheitskonzepte	1371
24.1	Zentrale Elemente der Java-Sicherheit	1371
24.1.1	Java Cryptography Architecture und Extension	1371
24.2	Der Sandkasten (Sandbox)	1372
24.3	Sicherheitsmanager (Security Manager)	1372
24.3.1	Der Sicherheitsmanager bei Applets	1374
24.3.2	Sicherheitsmanager aktivieren	1375
24.3.3	Wie nutzen die Java-Bibliotheken den Sicherheitsmanager?	1376
24.3.4	Rechte durch Policy-Dateien vergeben	1377
24.3.5	Erstellen von Rechte-Dateien mit dem grafischen Policy-Tool	1379
24.3.6	Kritik an den Policies	1380
24.4	Signierung	1381
24.4.1	Warum signieren?	1381
24.4.2	Digitale Ausweise und die Zertifizierungsstelle	1382
24.4.3	Mit keytool Schlüssel erzeugen	1382
24.4.4	Signieren mit jarsigner	1383

24.5	Digitale Unterschriften	1383
24.5.1	Die MDx-Reihe	1384
24.5.2	Secure Hash Algorithm (SHA)	1384
24.5.3	Mit der Security-API einen Fingerabdruck berechnen	1385
24.5.4	Die Klasse MessageDigest	1385
24.5.5	Unix-Crypt	1387
24.6	Verschlüsseln von Daten(-strömen)	1387
24.6.1	Den Schlüssel bitte	1388
24.6.2	Verschlüsseln mit Cipher	1389
24.6.3	Verschlüsseln von Datenströmen	1389
24.7	Zum Weiterlesen	1391

25 Java Native Interface (JNI) 1393

25.1	Java Native Interface und Invocation-API	1393
25.2	Einbinden einer C-Funktion in ein Java-Programm	1394
25.2.1	Schreiben des Java-Codes	1394
25.2.2	Compilieren des Java-Programms	1395
25.2.3	Erzeugen der Header-Datei	1395
25.2.4	Implementierung der Methode in C	1396
25.2.5	Übersetzen der C-Programme und Erzeugen der dynamischen Bibliothek	1397
25.2.6	Suchort der dynamischen Bibliothek	1399
25.3	Nativ die Stringlänge ermitteln	1399
25.4	Erweiterte JNI-Eigenschaften	1400
25.4.1	Klassendefinitionen	1400
25.4.2	Zugriff auf Attribute	1401
25.5	Einfache Anbindung von existierenden Bibliotheken	1403
25.5.1	C++ Klassen ansprechen	1403
25.5.2	COM-Schnittstellen anzapfen	1403
25.6	Zum Weiterlesen	1403

26 Dienstprogramme für die Java-Umgebung 1405

26.1	Die Werkzeuge im Überblick	1405
26.2	Java-Compiler	1405
26.2.1	Bytecode Compiler javac	1405
26.2.2	Native Compiler	1406
26.2.3	Java-Programme in ein natives ausführbares Programm einpacken	1407
26.3	Der Java-Interpreter java	1407
26.3.1	Der Unterschied zwischen java.exe und javaw.exe	1408

26.4	Das Archivformat Jar	1409
26.4.1	Das Dienstprogramm Jar benutzen	1409
26.4.2	Das Manifest	1412
26.4.3	Applikationen in Jar-Archiven starten	1412
26.4.4	Applets in Jar-Archiven	1413
26.5	Monitoringprogramme	1414
26.5.1	jps	1414
26.5.2	jstat	1414
26.5.3	jmap	1414
26.5.4	jstack	1415
26.6	Ant	1416
26.6.1	Bezug und Installation von Ant	1416
26.6.2	Properties	1418
26.6.3	Externe und vordefinierte Properties	1419
26.6.4	Weitere Ant-Tasks	1420
26.7	Decompiler und Obfuscatoren	1420
26.7.1	Der Decompiler Jad	1421
26.7.2	Das Decompilieren erschweren	1423
26.7.3	Das Obfuscator-Programm ProGuard	1425
26.8	Sourcecode Beautifier	1425
26.9	Zum Weiterlesen	1426
Anhang	1427
A	Die Begleit-DVD	1427
Index	1429