

# Inhalt

Vorwort .....	29
---------------	----

## 1 Java ist auch eine Sprache

---

<b>1.1 Historischer Hintergrund .....</b>	<b>47</b>
<b>1.2 Warum Java gut ist: die zentralen Eigenschaften .....</b>	<b>50</b>
1.2.1 Bytecode .....	50
1.2.2 Ausführung des Bytecodes durch eine virtuelle Maschine .....	50
1.2.3 Plattformunabhängigkeit .....	52
1.2.4 Java als Sprache, Laufzeitumgebung und Standardbibliothek .....	53
1.2.5 Objektorientierung in Java .....	54
1.2.6 Java ist verbreitet und bekannt .....	55
1.2.7 Java ist schnell: Optimierung und Just-in-Time Compilation .....	55
1.2.8 Das Java-Security-Modell .....	57
1.2.9 Zeiger und Referenzen .....	57
1.2.10 Bring den Müll raus, Garbage-Collector! .....	59
1.2.11 Ausnahmebehandlung .....	59
1.2.12 Einfache Syntax der Programmiersprache Java .....	60
1.2.13 Java ist Open Source .....	62
1.2.14 Wofür sich Java weniger eignet .....	64
1.2.15 Java im Vergleich zu anderen Sprachen .....	65
1.2.16 Java und das Web, Applets und JavaFX .....	68
1.2.17 Features, Enhancements (Erweiterungen) und ein JSR .....	71
1.2.18 Die Entwicklung von Java und seine Zukunftsaussichten .....	72
<b>1.3 Java-Plattformen: Java SE, Java EE und Java ME .....</b>	<b>73</b>
1.3.1 Die Java SE-Plattform .....	73
1.3.2 Java für die Kleinen .....	75
1.3.3 Java für die ganz, ganz Kleinen .....	76
1.3.4 Java für die Großen .....	76
1.3.5 Echtzeit-Java (Real-time Java) .....	77
<b>1.4 Die Installation der Java Platform Standard Edition (Java SE) .....</b>	<b>78</b>
1.4.1 Die Java SE von Oracle .....	78

1.4.2	Download des JDK .....	79
1.4.3	Java SE unter Windows installieren .....	80
<b>1.5</b>	<b>Das erste Programm compilieren und testen .....</b>	<b>84</b>
1.5.1	Ein Quadratzahlen-Programm .....	84
1.5.2	Der Compilerlauf .....	86
1.5.3	Die Laufzeitumgebung .....	88
1.5.4	Häufige Compiler- und Interpreterprobleme .....	88
<b>1.6</b>	<b>Entwicklungsumgebungen im Allgemeinen .....</b>	<b>89</b>
1.6.1	Die Entwicklungsumgebung Eclipse .....	89
1.6.2	NetBeans von Oracle .....	90
1.6.3	IntelliJ IDEA .....	91
1.6.4	Ein Wort zu Microsoft, Java und zu J++, J# .....	91
<b>1.7</b>	<b>Eclipse im Speziellen .....</b>	<b>92</b>
1.7.1	Eclipse starten .....	94
1.7.2	Das erste Projekt anlegen .....	96
1.7.3	Eine Klasse hinzufügen .....	100
1.7.4	Übersetzen und ausführen .....	102
1.7.5	JDK statt JRE * .....	103
1.7.6	Start eines Programms ohne Speicheraufforderung .....	103
1.7.7	Projekt einfügen, Workspace für die Programme wechseln .....	104
1.7.8	Plugins für Eclipse .....	106
<b>1.8</b>	<b>NetBeans im Speziellen .....</b>	<b>106</b>
1.8.1	NetBeans-Bundles .....	106
1.8.2	NetBeans installieren .....	107
1.8.3	NetBeans starten .....	108
1.8.4	Ein neues NetBeans-Projekt anlegen .....	108
1.8.5	Ein Java-Programm starten .....	110
1.8.6	Einstellungen .....	110
<b>1.9</b>	<b>Zum Weiterlesen .....</b>	<b>111</b>

## **2 Imperative Sprachkonzepte**

---

<b>2.1</b>	<b>Elemente der Programmiersprache Java .....</b>	<b>113</b>
2.1.1	Token .....	113
2.1.2	Textkodierung durch Unicode-Zeichen .....	115
2.1.3	Bezeichner .....	115

2.1.4	Literale .....	117
2.1.5	Reservierte Schlüsselwörter .....	118
2.1.6	Zusammenfassung der lexikalischen Analyse .....	119
2.1.7	Kommentare .....	120
<b>2.2</b>	<b>Von der Klasse zur Anweisung .....</b>	<b>122</b>
2.2.1	Was sind Anweisungen? .....	122
2.2.2	Klassendeklaration .....	123
2.2.3	Die Reise beginnt am main() .....	124
2.2.4	Der erste Methodenaufruf: println() .....	125
2.2.5	Atomare Anweisungen und Anweisungssequenzen .....	126
2.2.6	Mehr zu print(), println() und printf() für Bildschirmausgaben .....	126
2.2.7	Die API-Dokumentation .....	128
2.2.8	Ausdrücke .....	130
2.2.9	Ausdrucksanweisung .....	131
2.2.10	Erste Idee der Objektorientierung .....	132
2.2.11	Modifizierer .....	133
2.2.12	Gruppieren von Anweisungen mit Blöcken .....	134
<b>2.3</b>	<b>Datentypen, Typisierung, Variablen und Zuweisungen .....</b>	<b>135</b>
2.3.1	Primitive Datentypen im Überblick .....	136
2.3.2	Variablendeklarationen .....	139
2.3.3	Konsoleneingaben .....	142
2.3.4	Fließkommazahlen mit den Datentypen float und double .....	144
2.3.5	Ganzzahlige Datentypen .....	146
2.3.6	Wahrheitswerte .....	148
2.3.7	Unterstriche in Zahlen * .....	148
2.3.8	Alphanumerische Zeichen .....	149
2.3.9	Gute Namen, schlechte Namen .....	150
2.3.10	Initialisierung von lokalen Variablen .....	151
<b>2.4</b>	<b>Ausdrücke, Operanden und Operatoren .....</b>	<b>152</b>
2.4.1	Zuweisungsoperator .....	152
2.4.2	Arithmetische Operatoren .....	154
2.4.3	Unäres Minus und Plus .....	158
2.4.4	Zuweisung mit Operation .....	159
2.4.5	Präfix- oder Postfix-Inkrement und -Dekrement .....	160
2.4.6	Die relationalen Operatoren und die Gleichheitsoperatoren .....	162
2.4.7	Logische Operatoren: Nicht, Und, Oder, Xor .....	164
2.4.8	Kurzschluss-Operatoren .....	166

2.4.9	Der Rang der Operatoren in der Auswertungsreihenfolge .....	167
2.4.10	Die Typanpassung (das Casting) .....	170
2.4.11	Überladenes Plus für Strings .....	175
2.4.12	Operator vermisst * .....	176
<b>2.5</b>	<b>Bedingte Anweisungen oder Fallunterscheidungen .....</b>	<b>177</b>
2.5.1	Die if-Anweisung .....	177
2.5.2	Die Alternative mit einer if-else-Anweisung wählen .....	180
2.5.3	Der Bedingungsoperator .....	184
2.5.4	Die switch-Anweisung bietet die Alternative .....	187
<b>2.6</b>	<b>Schleifen .....</b>	<b>192</b>
2.6.1	Die while-Schleife .....	193
2.6.2	Die do-while-Schleife .....	195
2.6.3	Die for-Schleife .....	197
2.6.4	Schleifenbedingungen und Vergleiche mit == .....	201
2.6.5	Ausbruch planen mit break und Wiedereinstieg mit continue .....	204
2.6.6	break und continue mit Marken * .....	208
<b>2.7</b>	<b>Methoden einer Klasse .....</b>	<b>212</b>
2.7.1	Bestandteil einer Methode .....	213
2.7.2	Signatur-Beschreibung in der Java-API .....	215
2.7.3	Aufruf einer Methode .....	216
2.7.4	Methoden ohne Parameter deklarieren .....	217
2.7.5	Statische Methoden (Klassenmethoden) .....	218
2.7.6	Parameter, Argument und Wertübergabe .....	219
2.7.7	Methoden vorzeitig mit return beenden .....	221
2.7.8	Nicht erreichbarer Quellcode bei Methoden * .....	222
2.7.9	Methoden mit Rückgaben .....	223
2.7.10	Methoden überladen .....	227
2.7.11	Sichtbarkeit und Gültigkeitsbereich .....	230
2.7.12	Vorgegebener Wert für nicht aufgeführte Argumente * .....	232
2.7.13	Finale lokale Variablen .....	232
2.7.14	Rekursive Methoden * .....	233
2.7.15	Die Türme von Hanoi * .....	237
<b>2.8</b>	<b>Zum Weiterlesen .....</b>	<b>240</b>

## 3 Klassen und Objekte

---

<b>3.1 Objektorientierte Programmierung (OOP)</b> .....	241
3.1.1 Warum überhaupt OOP? .....	241
3.1.2 Denk ich an Java, denk ich an Wiederverwendbarkeit .....	242
<b>3.2 Eigenschaften einer Klasse</b> .....	243
3.2.1 Die Klasse Point .....	244
<b>3.3 Die UML (Unified Modeling Language) *</b> .....	244
3.3.1 Hintergrund und Geschichte der UML .....	245
3.3.2 Wichtige Diagrammtypen der UML .....	246
3.3.3 UML-Werkzeuge .....	247
<b>3.4 Neue Objekte erzeugen</b> .....	249
3.4.1 Ein Exemplar einer Klasse mit dem new-Operator anlegen .....	249
3.4.2 Garbage-Collector (GC) – Es ist dann mal weg .....	251
3.4.3 Deklarieren von Referenzvariablen .....	251
3.4.4 Zugriff auf Objektattribute und -methoden mit dem ».« .....	252
3.4.5 Überblick über Point-Methoden .....	257
3.4.6 Konstruktoren nutzen .....	261
<b>3.5 ZZZZZnake</b> .....	262
<b>3.6 Kompilationseinheiten, Imports und Pakete schnüren</b> .....	265
3.6.1 Volle Qualifizierung und import-Deklaration .....	266
3.6.2 Mit import p1.p2.* alle Typen eines Pakets erreichen .....	267
3.6.3 Hierarchische Strukturen über Pakete .....	268
3.6.4 Die package-Deklaration .....	269
3.6.5 Unbenanntes Paket (default package) .....	270
3.6.6 Klassen mit gleichen Namen in unterschiedlichen Paketen * .....	271
3.6.7 Compilationseinheit (Compilation Unit) .....	272
3.6.8 Statischer Import * .....	272
3.6.9 Eine Verzeichnisstruktur für eigene Projekte * .....	274
<b>3.7 Mit Referenzen arbeiten, Identität und Gleichheit</b> .....	274
3.7.1 Die null-Referenz .....	274
3.7.2 null-Referenzen testen .....	276
3.7.3 Zuweisungen bei Referenzen .....	278
3.7.4 Methoden mit nicht-primitiven Parametern .....	279
3.7.5 Identität von Objekten .....	284
3.7.6 Gleichheit und die Methode equals() .....	285

<b>3.8 Arrays</b> .....	287
3.8.1 Grundbestandteile .....	287
3.8.2 Deklaration von Arrays .....	288
3.8.3 Arrays mit Inhalt .....	289
3.8.4 Die Länge eines Arrays über das Attribut length auslesen .....	289
3.8.5 Zugriff auf die Elemente über den Index .....	290
3.8.6 Array-Objekte mit new erzeugen .....	292
3.8.7 Typische Feldfehler .....	293
3.8.8 Feld-Objekte als Parametertyp .....	294
3.8.9 Vorinitialisierte Arrays .....	295
3.8.10 Die erweiterte for-Schleife .....	296
3.8.11 Arrays mit nicht-primitiven Elementen .....	298
3.8.12 Mehrdimensionale Arrays * .....	301
3.8.13 Nichtrechteckige Arrays * .....	306
3.8.14 Die Wahrheit über die Array-Initialisierung * .....	309
3.8.15 Mehrere Rückgabewerte * .....	310
3.8.16 Methode mit variabler Argumentanzahl (Vararg) .....	311
3.8.17 Klonen kann sich lohnen – Arrays vermehren * .....	313
3.8.18 Feldinhalte kopieren * .....	314
3.8.19 Die Klasse Arrays zum Vergleichen, Füllen, Suchen, Sortieren nutzen .....	315
3.8.20 Eine lange Schlange .....	325
<b>3.9 Der Einstiegspunkt für das Laufzeitsystem: main()</b> .....	328
3.9.1 Korrekte Deklaration der Startmethode .....	328
3.9.2 Kommandozeilenargumente verarbeiten .....	329
3.9.3 Der Rückgabotyp von main() und System.exit() * .....	330
<b>3.10 Annotationen und Generics</b> .....	332
3.10.1 Generics .....	332
3.10.2 Annotationen .....	333
3.10.3 Eigene Metadaten setzen .....	333
3.10.4 Annotationstypen @Override, @Deprecated, @SuppressWarnings .....	334
<b>3.11 Zum Weiterlesen</b> .....	339

## 4 Der Umgang mit Zeichenketten

---

<b>4.1 Von ASCII über ISO-8859-1 zu Unicode</b> .....	341
4.1.1 ASCII .....	341

4.1.2	ISO/IEC 8859-1 .....	342
4.1.3	Unicode .....	343
4.1.4	Unicode-Zeichenkodierung .....	345
4.1.5	Escape-Sequenzen/Fluchtsymbole .....	346
4.1.6	Schreibweise für Unicode-Zeichen und Unicode-Escapes .....	347
4.1.7	Unicode 4.0 und Java * .....	349
<b>4.2</b>	<b>Die Character-Klasse .....</b>	<b>350</b>
4.2.1	Ist das so? .....	350
4.2.2	Zeichen in Großbuchstaben/Kleinbuchstaben konvertieren .....	353
4.2.3	Ziffern einer Basis * .....	355
<b>4.3</b>	<b>Zeichenfolgen .....</b>	<b>357</b>
<b>4.4</b>	<b>Die Klasse String und ihre Methoden .....</b>	<b>359</b>
4.4.1	String-Literale als String-Objekte für konstante Zeichenketten .....	359
4.4.2	Konkatenation mit + .....	361
4.4.3	String-Länge und Test auf Leerstring .....	361
4.4.4	Zugriff auf ein bestimmtes Zeichen mit charAt() .....	362
4.4.5	Nach enthaltenen Zeichen und Zeichenfolgen suchen .....	363
4.4.6	Das Hangman-Spiel .....	367
4.4.7	Gut, dass wir verglichen haben .....	369
4.4.8	Phonetische Vergleiche * .....	373
4.4.9	String-Teile extrahieren .....	374
4.4.10	Strings anhängen, Groß-/Kleinschreibung und Leerraum .....	378
4.4.11	Suchen und ersetzen .....	381
4.4.12	String-Objekte mit Konstruktoren neu anlegen * .....	383
<b>4.5</b>	<b>Konvertieren zwischen Primitiven und Strings .....</b>	<b>389</b>
4.5.1	Unterschiedliche Typen in String-Repräsentationen konvertieren .....	389
4.5.2	Stringinhalt in einen primitiven Wert konvertieren .....	391
4.5.3	String-Repräsentation im Format Binär, Hex, Oktal * .....	393
<b>4.6</b>	<b>Veränderbare Zeichenketten mit StringBuilder und StringBuffer .....</b>	<b>397</b>
4.6.1	Anlegen von StringBuilder/StringBuffer-Objekten .....	398
4.6.2	StringBuilder/StringBuffer in andere Zeichenkettenformate konvertieren .....	400
4.6.3	Zeichen(folgen) erfragen .....	400
4.6.4	Daten anhängen .....	400
4.6.5	Zeichen(folgen) setzen, löschen und umdrehen .....	402
4.6.6	Länge und Kapazität eines StringBuilder/StringBuffer-Objekts * .....	404

4.6.7	Vergleichen von String mit StringBuilder und StringBuffer .....	405
4.6.8	hashCode() bei StringBuilder/StringBuffer * .....	407
<b>4.7</b>	<b>CharSequence als Basistyp *</b> .....	407
<b>4.8</b>	<b>Reguläre Ausdrücke</b> .....	409
4.8.1	Pattern.matches() bzw. String#matches() .....	410
4.8.2	Die Klassen Pattern und Matcher .....	413
4.8.3	Finden und nicht matchen .....	419
4.8.4	Gierige und nicht gierige Operatoren * .....	420
4.8.5	Mit MatchResult alle Ergebnisse einsammeln * .....	421
4.8.6	Suchen und Ersetzen mit Mustern .....	423
4.8.7	Hangman Version 2 .....	425
<b>4.9</b>	<b>Zerlegen von Zeichenketten</b> .....	427
4.9.1	Splitten von Zeichenketten mit split() .....	427
4.9.2	Die Klasse Scanner .....	429
4.9.3	Die Klasse StringTokenizer * .....	436
4.9.4	BreakIterator als Zeichen-, Wort-, Zeilen- und Satztrenner * .....	438
<b>4.10</b>	<b>Zeichenkodierungen, XML/HTML-Entitys, Base64 *</b> .....	442
4.10.1	Unicode und 8-Bit-Abbildungen .....	442
4.10.2	Das Paket java.nio.charset und der Typ Charset .....	443
4.10.3	Konvertieren mit OutputStreamWriter/InputStreamReader-Klassen * ....	445
4.10.4	XML/HTML-Entitys ausmaskieren .....	446
4.10.5	Base64-Kodierung .....	447
<b>4.11</b>	<b>Ausgaben formatieren</b> .....	449
4.11.1	Formatieren und Ausgeben mit format() .....	449
4.11.2	Die Formatter-Klasse * .....	455
4.11.3	Formatieren mit Masken * .....	457
4.11.4	Format-Klassen .....	459
4.11.5	Zahlen, Prozente und Währungen mit NumberFormat und DecimalFormat formatieren * .....	462
4.11.6	MessageFormat und Pluralbildung mit ChoiceFormat .....	465
<b>4.12</b>	<b>Sprachabhängiges Vergleichen und Normalisierung *</b> .....	467
4.12.1	Die Klasse Collator .....	467
4.12.2	Effiziente interne Speicherung für die Sortierung .....	471
4.12.3	Normalisierung .....	473
<b>4.13</b>	<b>Zum Weiterlesen</b> .....	474

## 5 Eigene Klassen schreiben

---

<b>5.1</b>	<b>Eigene Klassen mit Eigenschaften deklarieren</b> .....	475
5.1.1	Attribute deklarieren .....	476
5.1.2	Methoden deklarieren .....	478
5.1.3	Die this-Referenz .....	483
<b>5.2</b>	<b>Privatsphäre und Sichtbarkeit</b> .....	487
5.2.1	Für die Öffentlichkeit: public .....	487
5.2.2	Kein Public Viewing – Passwörter sind privat .....	487
5.2.3	Wieso nicht freie Methoden und Variablen für alle? .....	489
5.2.4	Privat ist nicht ganz privat: Es kommt darauf an, wer's sieht * .....	490
5.2.5	Zugriffsmethoden für Attribute deklarieren .....	491
5.2.6	Setter und Getter nach der JavaBeans-Spezifikation .....	491
5.2.7	Paketsichtbar .....	494
5.2.8	Zusammenfassung zur Sichtbarkeit .....	496
<b>5.3</b>	<b>Statische Methoden und statische Attribute</b> .....	499
5.3.1	Warum statische Eigenschaften sinnvoll sind .....	499
5.3.2	Statische Eigenschaften mit static .....	500
5.3.3	Statische Eigenschaften über Referenzen nutzen? * .....	501
5.3.4	Warum die Groß- und Kleinschreibung wichtig ist * .....	502
5.3.5	Statische Variablen zum Datenaustausch * .....	503
5.3.6	Statische Eigenschaften und Objekteigenschaften * .....	505
<b>5.4</b>	<b>Konstanten und Aufzählungen</b> .....	505
5.4.1	Konstanten über öffentliche statische finale Variablen .....	506
5.4.2	Typ(un)sichere Aufzählungen * .....	507
5.4.3	Aufzählungen mit enum .....	509
<b>5.5</b>	<b>Objekte anlegen und zerstören</b> .....	513
5.5.1	Konstruktoren schreiben .....	513
5.5.2	Der vorgegebene Konstruktor (default constructor) .....	515
5.5.3	Parametrisierte und überladene Konstruktoren .....	517
5.5.4	Copy-Konstruktor .....	519
5.5.5	Einen anderen Konstruktor der gleichen Klasse mit this() aufrufen .....	521
5.5.6	Ihr fehlt uns nicht – der Garbage-Collector .....	525
5.5.7	Private Konstruktoren, Utility-Klassen, Singleton, Fabriken .....	527
<b>5.6</b>	<b>Klassen- und Objektinitialisierung *</b> .....	529
5.6.1	Initialisierung von Objektvariablen .....	530
5.6.2	Statische Blöcke als Klasseninitialisierer .....	532

5.6.3	Initialisierung von Klassenvariablen .....	534
5.6.4	Eincompilierte Belegungen der Klassenvariablen .....	534
5.6.5	Exemplarinitialisierer (Instanzinitialisierer) .....	535
5.6.6	Finale Werte im Konstruktor und in statischen Blöcken setzen .....	539
<b>5.7</b>	<b>Assoziationen zwischen Objekten .....</b>	<b>541</b>
5.7.1	Unidirektionale 1:1-Beziehung .....	542
5.7.2	Bidirektionale 1:1-Beziehungen .....	543
5.7.3	Unidirektionale 1:n-Beziehung .....	545
<b>5.8</b>	<b>Vererbung .....</b>	<b>547</b>
5.8.1	Vererbung in Java .....	548
5.8.2	Spielobjekte modellieren .....	548
5.8.3	Die implizite Basisklasse java.lang.Object .....	550
5.8.4	Einfach- und Mehrfachvererbung * .....	551
5.8.5	Die Sichtbarkeit protected .....	551
5.8.6	Konstruktoren in der Vererbung und super() .....	552
<b>5.9</b>	<b>Typen in Hierarchien .....</b>	<b>559</b>
5.9.1	Automatische und explizite Typanpassung .....	559
5.9.2	Das Substitutionsprinzip .....	561
5.9.3	Typen mit dem instanceof-Operator testen .....	563
<b>5.10</b>	<b>Methoden überschreiben .....</b>	<b>566</b>
5.10.1	Methoden in Unterklassen mit neuem Verhalten ausstatten .....	566
5.10.2	Mit super an die Eltern .....	570
5.10.3	Finale Klassen und finale Methoden .....	573
5.10.4	Kovariante Rückgabetypen .....	575
5.10.5	Array-Typen und Kovarianz * .....	576
<b>5.11</b>	<b>Drum prüfe, wer sich ewig dynamisch bindet .....</b>	<b>577</b>
5.11.1	Gebunden an toString() .....	578
5.11.2	Implementierung von System.out.println(Object) .....	580
5.11.3	Nicht dynamisch gebunden bei privaten, statischen und finalen Methoden .....	581
5.11.4	Dynamisch gebunden auch bei Konstruktoraufrufen * .....	583
5.11.5	Eine letzte Spielerei mit Javas dynamischer Bindung und überschatteten Attributen * .....	585
<b>5.12</b>	<b>Abstrakte Klassen und abstrakte Methoden .....</b>	<b>587</b>
5.12.1	Abstrakte Klassen .....	587
5.12.2	Abstrakte Methoden .....	589

<b>5.13 Schnittstellen</b> .....	593
5.13.1 Schnittstellen deklarieren .....	594
5.13.2 Implementieren von Schnittstellen .....	595
5.13.3 Markierungsschnittstellen * .....	597
5.13.4 Ein Polymorphie-Beispiel mit Schnittstellen .....	598
5.13.5 Die Mehrfachvererbung bei Schnittstellen * .....	599
5.13.6 Keine Kollisionsgefahr bei Mehrfachvererbung * .....	604
5.13.7 Erweitern von Interfaces – Subinterfaces .....	605
5.13.8 Konstantendeklarationen bei Schnittstellen .....	606
5.13.9 Initialisierung von Schnittstellenkonstanten * .....	609
5.13.10 Abstrakte Klassen und Schnittstellen im Vergleich .....	613
<b>5.14 Zum Weiterlesen</b> .....	614

## 6 Exceptions

---

<b>6.1 Problembereiche einzäunen</b> .....	615
6.1.1 Exceptions in Java mit try und catch .....	616
6.1.2 Eine NumberFormatException auffangen .....	617
6.1.3 Ablauf einer Ausnahmesituation .....	620
6.1.4 Eigenschaften vom Exception-Objekt .....	620
6.1.5 Wiederholung abgebrochener Bereiche * .....	622
6.1.6 Mehrere Ausnahmen auffangen .....	623
6.1.7 throws im Methodenkopf angeben .....	626
6.1.8 Abschlussbehandlung mit finally .....	627
<b>6.2 RuntimeException muss nicht aufgefangen werden</b> .....	633
6.2.1 Beispiele für RuntimeException-Klassen .....	634
6.2.2 Kann man abfangen, muss man aber nicht .....	635
<b>6.3 Die Klassenhierarchie der Fehler</b> .....	635
6.3.1 Die Exception-Hierarchie .....	636
6.3.2 Oberausnahmen auffangen .....	636
6.3.3 Schon gefangen? .....	638
6.3.4 Alles geht als Exception durch .....	639
6.3.5 Zusammenfassen gleicher catch-Blöcke mit dem multi-catch .....	641
<b>6.4 Harte Fehler: Error *</b> .....	645
<b>6.5 Auslösen eigener Exceptions</b> .....	646
6.5.1 Mit throw Ausnahmen auslösen .....	646

6.5.2	Vorhandene Runtime-Fehlertypen kennen und nutzen .....	649
6.5.3	Parameter testen und gute Fehlermeldungen .....	651
6.5.4	Neue Exception-Klassen deklarieren .....	653
6.5.5	Eigene Ausnahmen als Unterklassen von Exception oder RuntimeException? .....	655
6.5.6	Ausnahmen abfangen und weiterleiten * .....	657
6.5.7	Aufrufstack von Ausnahmen verändern * .....	659
6.5.8	Präzises rethrow * .....	660
6.5.9	Geschachtelte Ausnahmen * .....	664
<b>6.6</b>	<b>Automatisches Ressourcen-Management (try mit Ressourcen)</b> .....	<b>667</b>
6.6.1	try mit Ressourcen .....	668
6.6.2	Ausnahmen vom close() bleiben bestehen .....	669
6.6.3	Die Schnittstelle AutoCloseable .....	670
6.6.4	Mehrere Ressourcen nutzen .....	671
6.6.5	Unterdrückte Ausnahmen * .....	673
<b>6.7</b>	<b>Besonderheiten bei der Ausnahmebehandlung *</b> .....	<b>676</b>
6.7.1	Rückgabewerte bei ausgelösten Ausnahmen .....	677
6.7.2	Ausnahmen und Rückgaben verschwinden: Das Duo »return« und »finally« .....	677
6.7.3	throws bei überschriebenen Methoden .....	679
6.7.4	Nicht erreichbare catch-Klauseln .....	681
<b>6.8</b>	<b>Den Stack-Trace erfragen *</b> .....	<b>682</b>
6.8.1	StackTraceElement .....	683
6.8.2	printStackTrace() .....	684
6.8.3	StackTraceElement vom Thread erfragen .....	685
<b>6.9</b>	<b>Assertions *</b> .....	<b>686</b>
6.9.1	Assertions in eigenen Programmen nutzen .....	687
6.9.2	Assertions aktivieren .....	688
<b>6.10</b>	<b>Zum Weiterlesen</b> .....	<b>690</b>

## 7 Äußere.innere Klassen

---

<b>7.1</b>	<b>Geschachtelte (innere) Klassen, Schnittstellen, Aufzählungen</b> .....	<b>691</b>
<b>7.2</b>	<b>Statische innere Klassen und Schnittstellen</b> .....	<b>692</b>

<b>7.3</b>	<b>Mitglieds- oder Elementklassen</b> .....	694
7.3.1	Exemplare innerer Klassen erzeugen .....	695
7.3.2	Die this-Referenz .....	696
7.3.3	Vom Compiler generierte Klassendateien * .....	697
7.3.4	Erlaubte Modifizierer bei äußeren und inneren Klassen .....	698
7.3.5	Innere Klassen greifen auf private Eigenschaften zu .....	698
<b>7.4</b>	<b>Lokale Klassen</b> .....	700
<b>7.5</b>	<b>Anonyme innere Klassen</b> .....	701
7.5.1	Umsetzung innerer anonymer Klassen * .....	703
7.5.2	Nutzung innerer Klassen für Threads * .....	703
7.5.3	Konstruktoren innerer anonymer Klassen * .....	704
<b>7.6</b>	<b>Zugriff auf lokale Variablen aus lokalen inneren und anonymen Klassen *</b> .....	705
<b>7.7</b>	<b>this in Unterklassen *</b> .....	707

## 8 Besondere Klassen der Java SE

---

<b>8.1</b>	<b>Vergleichen von Objekten</b> .....	709
8.1.1	Natürlich geordnet oder nicht? .....	709
8.1.2	Die Schnittstelle Comparable .....	710
8.1.3	Die Schnittstelle Comparator .....	711
8.1.4	Rückgabewerte kodieren die Ordnung .....	711
8.1.5	Aneinanderreihung von Comparatoren * .....	713
<b>8.2</b>	<b>Wrapper-Klassen und Autoboxing</b> .....	718
8.2.1	Wrapper-Objekte erzeugen .....	719
8.2.2	Konvertierungen in eine String-Repräsentation .....	721
8.2.3	Die Basisklasse Number für numerische Wrapper-Objekte .....	722
8.2.4	Vergleiche durchführen mit compare(), compareTo(), equals() .....	724
8.2.5	Die Klasse Integer .....	727
8.2.6	Die Klassen Double und Float für Fließkommazahlen .....	729
8.2.7	Die Long-Klasse .....	730
8.2.8	Die Boolean-Klasse .....	731
8.2.9	Autoboxing: Boxing und Unboxing .....	732
<b>8.3</b>	<b>Object ist die Mutter aller Klassen</b> .....	737
8.3.1	Klassenobjekte .....	737
8.3.2	Objektidentifikation mit toString() .....	738

8.3.3	Objektgleichheit mit equals() und Identität .....	740
8.3.4	Klonen eines Objekts mit clone() * .....	745
8.3.5	Hashcodes über hashCode() liefern * .....	751
8.3.6	System.identityHashCode() und das Problem der nicht-eindeutigen Objektverweise * .....	757
8.3.7	Aufräumen mit finalize() * .....	761
8.3.8	Synchronisation * .....	764
<b>8.4</b>	<b>Die Utility-Klasse java.util.Objects .....</b>	<b>764</b>
<b>8.5</b>	<b>Die Spezial-Oberklasse Enum .....</b>	<b>767</b>
8.5.1	Methoden auf Enum-Objekten .....	767
8.5.2	enum mit eigenen Konstruktoren und Methoden * .....	771
<b>8.6</b>	<b>Erweitertes for und Iterable .....</b>	<b>777</b>
8.6.1	Die Schnittstelle Iterable .....	777
8.6.2	Einen eigenen Iterable implementieren * .....	778
<b>8.7</b>	<b>Zum Weiterlesen .....</b>	<b>780</b>

## 9 Generics<T>

---

<b>9.1</b>	<b>Einführung in Java Generics .....</b>	<b>781</b>
9.1.1	Mensch versus Maschine: Typprüfung des Compilers und der Laufzeitumgebung .....	781
9.1.2	Taschen .....	782
9.1.3	Generische Typen deklarieren .....	785
9.1.4	Generics nutzen .....	786
9.1.5	Diamonds are forever .....	789
9.1.6	Generische Schnittstellen .....	792
9.1.7	Generische Methoden/Konstruktoren und Typ-Inferenz .....	794
<b>9.2</b>	<b>Umsetzen der Generics, Typlöschung und Raw-Types .....</b>	<b>799</b>
9.2.1	Realisierungsmöglichkeiten .....	799
9.2.2	Typlöschung (Type Erasure) .....	800
9.2.3	Probleme aus der Typlöschung .....	802
9.2.4	Raw-Type .....	807
<b>9.3</b>	<b>Einschränken der Typen über Bounds .....</b>	<b>810</b>
9.3.1	Einfache Einschränkungen mit extends .....	810
9.3.2	Weitere Obertypen mit & .....	813

<b>9.4</b>	<b>Typparameter in der throws-Klausel *</b> .....	814
9.4.1	Deklaration einer Klasse mit Typvariable <E extends Exception> .....	814
9.4.2	Parametrisierter Typ bei Typvariable <E extends Exception> .....	814
<b>9.5</b>	<b>Generics und Vererbung, Invarianz</b> .....	818
9.5.1	Arrays sind invariant .....	818
9.5.2	Generics sind kovariant .....	819
9.5.3	Wildcardcards mit ? .....	820
9.5.4	Bounded Wildcards .....	822
9.5.5	Bounded-Wildcard-Typen und Bounded-Typvariablen .....	826
9.5.6	Das LESS-Prinzip .....	829
9.5.7	Enum<E extends Enum<E>> * .....	832
<b>9.6</b>	<b>Konsequenzen der Typlöschung: Typ-Token, Arrays und Brücken *</b> .....	834
9.6.1	Typ-Token .....	834
9.6.2	Super-Type-Token .....	836
9.6.3	Generics und Arrays .....	837
9.6.4	Brückenmethoden .....	839

## **10 Architektur, Design und angewandte Objektorientierung**

---

<b>10.1</b>	<b>Architektur, Design und Implementierung</b> .....	847
<b>10.2</b>	<b>Design-Pattern (Entwurfsmuster)</b> .....	848
10.2.1	Motivation für Design-Pattern .....	849
10.2.2	Das Beobachter-Pattern (Observer/Observable) .....	849
10.2.3	Ereignisse über Listener .....	856
<b>10.3</b>	<b>JavaBean</b> .....	861
10.3.1	Properties (Eigenschaften) .....	862
10.3.2	Einfache Eigenschaften .....	863
10.3.3	Indizierte Eigenschaften .....	863
10.3.4	Gebundene Eigenschaften und PropertyChangeListener .....	864
10.3.5	Veto-Eigenschaften – dagegen! .....	867
<b>10.4</b>	<b>Zum Weiterlesen</b> .....	872

## 11 Die Klassenbibliothek

---

<b>11.1 Die Java-Klassenphilosophie</b> .....	873
11.1.1 Übersicht über die Pakete der Standardbibliothek .....	873
<b>11.2 Sprachen der Länder</b> .....	876
11.2.1 Sprachen und Regionen über Locale-Objekte .....	876
<b>11.3 Die Klasse Date</b> .....	880
11.3.1 Objekte erzeugen und Methoden nutzen .....	880
11.3.2 Date-Objekte sind nicht immutable .....	882
<b>11.4 Calendar und GregorianCalendar</b> .....	883
11.4.1 Die abstrakte Klasse Calendar .....	883
11.4.2 Der gregorianische Kalender .....	885
11.4.3 Calendar nach Date und Millisekunden fragen .....	887
11.4.4 Abfragen und Setzen von Datumselementen über Feldbezeichner .....	888
<b>11.5 Klassenlader (Class Loader)</b> .....	892
11.5.1 Woher die kleinen Klassen kommen .....	892
11.5.2 Setzen des Klassenpfades .....	894
11.5.3 Die wichtigsten drei Typen von Klassenladern .....	895
11.5.4 Die Klasse java.lang.ClassLoader * .....	896
11.5.5 Hot Deployment mit dem URL-Classloader * .....	897
11.5.6 Das Verzeichnis jre/lib/endorsed * .....	901
<b>11.6 Die Utility-Klasse System und Properties</b> .....	902
11.6.1 Systemeigenschaften der Java-Umgebung .....	903
11.6.2 line.separator .....	905
11.6.3 Eigene Properties von der Konsole aus setzen * .....	905
11.6.4 Umgebungsvariablen des Betriebssystems * .....	908
11.6.5 Einfache Zeitmessung und Profiling * .....	910
<b>11.7 Einfache Benutzereingaben</b> .....	913
11.7.1 Grafischer Eingabedialog über JOptionPane .....	913
11.7.2 Geschützte Passwort-Eingaben mit der Klasse Console * .....	915
<b>11.8 Ausführen externer Programme *</b> .....	916
11.8.1 ProcessBuilder und Prozesskontrolle mit Process .....	917
11.8.2 Einen Browser, E-Mail-Client oder Editor aufrufen .....	922
<b>11.9 Benutzereinstellungen *</b> .....	924
11.9.1 Benutzereinstellungen mit der Preferences-API .....	924
11.9.2 Einträge einfügen, auslesen und löschen .....	926

11.9.3	Auslesen der Daten und Schreiben in einem anderen Format .....	929
11.9.4	Auf Ereignisse horchen .....	929
11.9.5	Zugriff auf die gesamte Windows-Registry .....	931
<b>11.10</b>	<b>Zum Weiterlesen .....</b>	<b>932</b>

## **12 Einführung in die nebenläufige Programmierung**

---

<b>12.1</b>	<b>Nebenläufigkeit .....</b>	<b>933</b>
12.1.1	Threads und Prozesse .....	934
12.1.2	Wie parallele Programme die Geschwindigkeit steigern können .....	935
12.1.3	Was Java für Nebenläufigkeit alles bietet .....	937
<b>12.2</b>	<b>Threads erzeugen .....</b>	<b>937</b>
12.2.1	Threads über die Schnittstelle Runnable implementieren .....	938
12.2.2	Thread mit Runnable starten .....	939
12.2.3	Die Klasse Thread erweitern .....	941
<b>12.3</b>	<b>Thread-Eigenschaften und -Zustände .....</b>	<b>944</b>
12.3.1	Der Name eines Threads .....	944
12.3.2	Wer bin ich? .....	944
12.3.3	Schläfer gesucht .....	945
12.3.4	Mit yield() auf Rechenzeit verzichten .....	947
12.3.5	Der Thread als Dämon .....	948
12.3.6	Das Ende eines Threads .....	950
12.3.7	Einen Thread höflich mit Interrupt beenden .....	951
12.3.8	UncaughtExceptionHandler für unbehandelte Ausnahmen .....	953
<b>12.4</b>	<b>Der Ausführer (Executor) kommt .....</b>	<b>954</b>
12.4.1	Die Schnittstelle Executor .....	955
12.4.2	Die Thread-Pools .....	957
<b>12.5</b>	<b>Synchronisation über kritische Abschnitte .....</b>	<b>958</b>
12.5.1	Gemeinsam genutzte Daten .....	959
12.5.2	Probleme beim gemeinsamen Zugriff und kritische Abschnitte .....	959
12.5.3	Punkte parallel initialisieren .....	961
12.5.4	Kritische Abschnitte schützen .....	963
12.5.5	Kritische Abschnitte mit ReentrantLock schützen .....	966
<b>12.6</b>	<b>Zum Weiterlesen .....</b>	<b>969</b>

## 13 Einführung in Datenstrukturen und Algorithmen

---

<b>13.1</b>	<b>Datenstrukturen und die Collection-API</b> .....	971
13.1.1	Designprinzip mit Schnittstellen, abstrakten und konkreten Klassen ....	972
13.1.2	Die Basis-Schnittstellen Collection und Map .....	972
13.1.3	Die Utility-Klassen Collections und Arrays .....	973
13.1.4	Das erste Programm mit Container-Klassen .....	973
13.1.5	Die Schnittstelle Collection und Kernkonzepte .....	975
13.1.6	Schnittstellen, die Collection erweitern und Map .....	978
13.1.7	Konkrete Container-Klassen .....	981
13.1.8	Generische Datentypen in der Collection-API .....	982
13.1.9	Die Schnittstelle Iterable und das erweiterte for .....	983
<b>13.2</b>	<b>Listen</b> .....	983
13.2.1	Erstes Listen-Beispiel .....	984
<b>13.3</b>	<b>Mengen (Sets)</b> .....	987
13.3.1	Ein erstes Mengen-Beispiel .....	988
13.3.2	Methoden der Schnittstelle Set .....	991
<b>13.4</b>	<b>Assoziative Speicher</b> .....	992
13.4.1	Die Klassen HashMap und TreeMap .....	992
13.4.2	Einfügen und Abfragen der Datenstruktur .....	995
13.4.3	Über die Bedeutung von equals() und hashCode() .....	998
<b>13.5</b>	<b>Mit einem Iterator durch die Daten wandern</b> .....	998
13.5.1	Die Schnittstelle Iterator .....	999
13.5.2	Der Iterator kann (eventuell auch) löschen .....	1001
<b>13.6</b>	<b>Algorithmen in Collections</b> .....	1002
13.6.1	Die Bedeutung von Ordnung mit Comparator und Comparable .....	1004
13.6.2	Sortieren .....	1005
13.6.3	Den größten und kleinsten Wert einer Collection finden .....	1008
<b>13.7</b>	<b>Zum Weiterlesen</b> .....	1011

## 14 Einführung in grafische Oberflächen

---

<b>14.1</b>	<b>Das Abstract Window Toolkit und Swing</b> .....	1013
14.1.1	SwingSet-Demos .....	1013
14.1.2	Abstract Window Toolkit (AWT) .....	1014

14.1.3	Java Foundation Classes .....	1015
14.1.4	Was Swing von AWT unterscheidet .....	1018
14.1.5	GUI-Builder für AWT und Swing .....	1019
<b>14.2</b>	<b>Mit NetBeans zur ersten Oberfläche .....</b>	<b>1020</b>
14.2.1	Projekt anlegen .....	1020
14.2.2	Eine GUI-Klasse hinzufügen .....	1022
14.2.3	Programm starten .....	1024
14.2.4	Grafische Oberfläche aufbauen .....	1025
14.2.5	Swing-Komponenten-Klassen .....	1028
14.2.6	Funktionalität geben .....	1030
<b>14.3</b>	<b>Fenster zur Welt .....</b>	<b>1033</b>
14.3.1	Swing-Fenster mit javax.swing.JFrame darstellen .....	1034
14.3.2	Fenster schließbar machen – setDefaultCloseOperation() .....	1035
14.3.3	Sichtbarkeit des Fensters .....	1036
<b>14.4</b>	<b>Beschriftungen (JLabel) .....</b>	<b>1037</b>
<b>14.5</b>	<b>Es tut sich was – Ereignisse beim AWT .....</b>	<b>1039</b>
14.5.1	Die Ereignisquellen und Horcher (Listener) von Swing .....	1039
14.5.2	Listener implementieren .....	1041
14.5.3	Listener bei dem Ereignisauslöser anmelden/abmelden .....	1044
14.5.4	Adapterklassen nutzen .....	1046
14.5.5	Innere Mitgliedsklassen und innere anonyme Klassen .....	1048
14.5.6	Aufrufen der Listener im AWT-Event-Thread .....	1051
<b>14.6</b>	<b>Schaltflächen .....</b>	<b>1051</b>
14.6.1	Normale Schaltflächen (JButton) .....	1051
14.6.2	Der aufmerksame ActionListener .....	1054
<b>14.7</b>	<b>Alles Auslegungssache: die Layoutmanager .....</b>	<b>1055</b>
14.7.1	Übersicht über Layoutmanager .....	1055
14.7.2	Zuweisen eines Layoutmanagers .....	1056
14.7.3	Im Fluss mit FlowLayout .....	1057
14.7.4	BoxLayout .....	1059
14.7.5	Mit BorderLayout in alle Himmelsrichtungen .....	1060
14.7.6	Rasteranordnung mit GridLayout .....	1063
<b>14.8</b>	<b>Textkomponenten .....</b>	<b>1065</b>
14.8.1	Text in einer Eingabezeile .....	1066
14.8.2	Die Oberklasse der Text-Komponenten (JTextComponent) .....	1067

<b>14.9</b>	<b>Zeichnen von grafischen Primitiven</b> .....	1068
14.9.1	Die paint()-Methode für das AWT-Frame .....	1068
14.9.2	Die ereignisorientierte Programmierung ändert Fensterinhalte .....	1070
14.9.3	Zeichnen von Inhalten auf ein JFrame .....	1072
14.9.4	Linien .....	1073
14.9.5	Rechtecke .....	1074
14.9.6	Zeichenfolgen schreiben .....	1075
14.9.7	Die Font-Klasse .....	1076
14.9.8	Farben mit der Klasse Color .....	1079
<b>14.10</b>	<b>Zum Weiterlesen</b> .....	1083

## **15 Einführung in Dateien und Datenströme**

---

<b>15.1</b>	<b>Datei und Verzeichnis</b> .....	1085
15.1.1	Dateien und Verzeichnisse mit der Klasse File .....	1086
15.1.2	Verzeichnis oder Datei? Existiert es? .....	1089
15.1.3	Verzeichnis- und Dateieigenschaften/-attribute .....	1090
15.1.4	Umbenennen und Verzeichnisse anlegen .....	1091
15.1.5	Verzeichnisse auflisten und Dateien filtern .....	1092
15.1.6	Dateien und Verzeichnisse löschen .....	1093
<b>15.2</b>	<b>Dateien mit wahlfreiem Zugriff</b> .....	1094
15.2.1	Ein RandomAccessFile zum Lesen und Schreiben öffnen .....	1095
15.2.2	Aus dem RandomAccessFile lesen .....	1096
15.2.3	Schreiben mit RandomAccessFile .....	1097
15.2.4	Die Länge des RandomAccessFile .....	1098
15.2.5	Hin und her in der Datei .....	1098
<b>15.3</b>	<b>Dateisysteme unter NIO.2</b> .....	1100
15.3.1	FileSystem und Path .....	1100
15.3.2	Die Utility-Klasse Files .....	1102
<b>15.4</b>	<b>Stream-Klassen und Reader/Writer am Beispiel von Dateien</b> .....	1104
15.4.1	Mit dem FileWriter Texte in Dateien schreiben .....	1105
15.4.2	Zeichen mit der Klasse FileReader lesen .....	1107
15.4.3	Kopieren mit FileOutputStream und FileInputStream .....	1108
15.4.4	Datenströme über Files mit NIO.2 beziehen .....	1110
<b>15.5</b>	<b>Basisklassen für die Ein-/Ausgabe</b> .....	1113
15.5.1	Die abstrakten Basisklassen .....	1113

15.5.2	Übersicht über Ein-/Ausgabeklassen .....	1113
15.5.3	Die abstrakte Basisklasse OutputStream .....	1116
15.5.4	Die Schnittstellen Closeable, AutoCloseable und Flushable .....	1117
15.5.5	Die abstrakte Basisklasse InputStream .....	1118
15.5.6	Ressourcen aus dem Klassenpfad und aus Jar-Archiven laden .....	1120
15.5.7	Die abstrakte Basisklasse Writer .....	1120
15.5.8	Die abstrakte Basisklasse Reader .....	1122
<b>15.6</b>	<b>Datenströme filtern und verketteten</b> .....	<b>1125</b>
15.6.1	Streams als Filter verketteten (verschachteln) .....	1125
15.6.2	Gepufferte Ausgaben mit BufferedWriter und BufferedOutputStream ...	1126
15.6.3	Gepufferte Eingaben mit BufferedReader/BufferedInputStream .....	1128
<b>15.7</b>	<b>Vermittler zwischen Byte-Streams und Unicode-Strömen</b> .....	<b>1131</b>
15.7.1	Datenkonvertierung durch den OutputStreamWriter .....	1131
15.7.2	Automatische Konvertierungen mit dem InputStreamReader .....	1132

## 16 Einführung in die <XML>-Verarbeitung mit Java

---

<b>16.1</b>	<b>Auszeichnungssprachen</b> .....	<b>1135</b>
16.1.1	Die Standard Generalized Markup Language (SGML) .....	1136
16.1.2	Extensible Markup Language (XML) .....	1136
<b>16.2</b>	<b>Eigenschaften von XML-Dokumenten</b> .....	<b>1137</b>
16.2.1	Elemente und Attribute .....	1137
16.2.2	Beschreibungssprache für den Aufbau von XML-Dokumenten .....	1140
16.2.3	Schema – eine Alternative zu DTD .....	1144
16.2.4	Namensraum (Namespace) .....	1147
16.2.5	XML-Applikationen * .....	1148
<b>16.3</b>	<b>Die Java-APIs für XML</b> .....	<b>1149</b>
16.3.1	Das Document Object Model (DOM) .....	1150
16.3.2	Simple API for XML Parsing (SAX) .....	1150
16.3.3	Pull-API StAX .....	1150
16.3.4	Java Document Object Model (JDOM) .....	1150
16.3.5	JAXP als Java-Schnittstelle zu XML .....	1151
16.3.6	DOM-Bäume einlesen mit JAXP * .....	1152
<b>16.4</b>	<b>Java Architecture for XML Binding (JAXB)</b> .....	<b>1153</b>
16.4.1	Bean für JAXB aufbauen .....	1153

16.4.2	JAXBContext und die Marshaller .....	1154
16.4.3	Ganze Objektgraphen schreiben und lesen .....	1156
<b>16.5</b>	<b>XML-Dateien mit JDOM verarbeiten .....</b>	<b>1158</b>
16.5.1	JDOM beziehen .....	1159
16.5.2	Paketübersicht * .....	1159
16.5.3	Die Document-Klasse .....	1161
16.5.4	Eingaben aus der Datei lesen .....	1162
16.5.5	Das Dokument im XML-Format ausgeben .....	1163
16.5.6	Elemente .....	1164
16.5.7	Zugriff auf Elementinhalte .....	1167
16.5.8	Attributinhalte lesen und ändern .....	1170
<b>16.6</b>	<b>Zum Weiterlesen .....</b>	<b>1173</b>

## **17 Einführung ins Datenbankmanagement mit JDBC**

---

<b>17.1</b>	<b>Relationale Datenbanken .....</b>	<b>1175</b>
17.1.1	Das relationale Modell .....	1175
<b>17.2</b>	<b>Einführung in SQL .....</b>	<b>1176</b>
17.2.1	Ein Rundgang durch SQL-Abfragen .....	1177
17.2.2	Datenabfrage mit der Data Query Language (DQL) .....	1179
17.2.3	Tabellen mit der Data Definition Language (DDL) anlegen .....	1181
<b>17.3</b>	<b>Datenbanken und Tools .....</b>	<b>1182</b>
17.3.1	HSQLDB .....	1182
17.3.2	Eclipse-Plugins zum Durchschauen von Datenbanken .....	1184
<b>17.4</b>	<b>JDBC und Datenbanktreiber .....</b>	<b>1188</b>
<b>17.5</b>	<b>Eine Beispielabfrage .....</b>	<b>1190</b>
17.5.1	Schritte zur Datenbankabfrage .....	1190
17.5.2	Ein Client für die HSQLDB-Datenbank .....	1190
17.5.3	Datenbankbrowser und eine Beispielabfrage unter NetBeans .....	1192

## **18 Bits und Bytes und Mathematisches**

---

<b>18.1</b>	<b>Bits und Bytes * .....</b>	<b>1199</b>
18.1.1	Die Bit-Operatoren Komplement, Und, Oder und Xor .....	1200

18.1.2	Repräsentation ganzer Zahlen in Java – das Zweierkomplement .....	1202
18.1.3	Das binäre (Basis 2), oktale (Basis 8), hexadezimale (Basis 16) Stellenwertsystem .....	1203
18.1.4	Auswirkung der Typanpassung auf die Bitmuster .....	1204
18.1.5	byte als vorzeichenlosen Datentyp nutzen .....	1207
18.1.6	Die Verschiebeoperatoren .....	1208
18.1.7	Ein Bit setzen, löschen, umdrehen und testen .....	1210
18.1.8	Bit-Methoden der Integer- und Long-Klasse .....	1211
<b>18.2</b>	<b>Fließkommaarithmetik in Java .....</b>	<b>1213</b>
18.2.1	Spezialwerte für Unendlich, Null, NaN .....	1213
18.2.2	Standard-Notation und wissenschaftliche Notation bei Fließkommazahlen * .....	1216
18.2.3	Mantisse und Exponent * .....	1217
<b>18.3</b>	<b>Die Eigenschaften der Klasse Math .....</b>	<b>1218</b>
18.3.1	Attribute .....	1220
18.3.2	Absolutwerte und Vorzeichen .....	1220
18.3.3	Maximum/Minimum .....	1221
18.3.4	Runden von Werten .....	1222
18.3.5	Wurzel- und Exponentialmethoden .....	1225
18.3.6	Der Logarithmus * .....	1226
18.3.7	Rest der ganzzahligen Division * .....	1227
18.3.8	Winkelmethoden * .....	1228
18.3.9	Zufallszahlen .....	1229
<b>18.4</b>	<b>Genauigkeit, Wertebereich eines Typs und Überlaufkontrolle * .....</b>	<b>1230</b>
18.4.1	Behandlung des Überlaufs .....	1230
18.4.2	Was bitte macht ein ulp? .....	1233
<b>18.5</b>	<b>Mathe bitte strikt * .....</b>	<b>1234</b>
18.5.1	Strikte Fließkommaberechnungen mit strictfp .....	1235
18.5.2	Die Klassen Math und StrictMath .....	1235
<b>18.6</b>	<b>Die Random-Klasse .....</b>	<b>1236</b>
18.6.1	Objekte mit dem Samen aufbauen .....	1236
18.6.2	Zufallszahlen erzeugen .....	1237
18.6.3	Pseudo-Zufallszahlen in der Normalverteilung * .....	1238
<b>18.7</b>	<b>Große Zahlen * .....</b>	<b>1238</b>
18.7.1	Die Klasse BigInteger .....	1239
18.7.2	Methoden von BigInteger .....	1241
18.7.3	Ganz lange Fakultäten .....	1244

18.7.4	Große Fließkommazahlen mit BigDecimal .....	1246
18.7.5	Mit MathContext komfortabel die Rechengenauigkeit setzen .....	1248
<b>18.8</b>	<b>Zum Weiterlesen .....</b>	<b>1250</b>

## **19 Die Werkzeuge des JDK**

---

<b>19.1</b>	<b>Java-Quellen übersetzen .....</b>	<b>1252</b>
19.1.1	Java-Compiler vom JDK .....	1252
19.1.2	Native Compiler .....	1253
19.1.3	Java-Programme in ein natives ausführbares Programm einpacken .....	1253
<b>19.2</b>	<b>Die Java-Laufzeitumgebung .....</b>	<b>1254</b>
<b>19.3</b>	<b>Dokumentationskommentare mit JavaDoc .....</b>	<b>1259</b>
19.3.1	Einen Dokumentationskommentar setzen .....	1259
19.3.2	Mit dem Werkzeug javadoc eine Dokumentation erstellen .....	1262
19.3.3	HTML-Tags in Dokumentationskommentaren * .....	1263
19.3.4	Generierte Dateien .....	1263
19.3.5	Dokumentationskommentare im Überblick * .....	1264
19.3.6	JavaDoc und Doclets * .....	1266
19.3.7	Veraltete (deprecated) Typen und Eigenschaften .....	1266
<b>19.4</b>	<b>Das Archivformat Jar .....</b>	<b>1269</b>
19.4.1	Das Dienstprogramm jar benutzen .....	1270
19.4.2	Das Manifest .....	1273
19.4.3	Applikationen in Jar-Archiven starten .....	1273

## **A Die Klassenbibliothek**

---

<b>A.1</b>	<b>java.lang-Paket .....</b>	<b>1285</b>
A.1.1	Schnittstellen .....	1285
A.1.2	Klassen .....	1286
<b>Index .....</b>		<b>1289</b>