

Inhaltsverzeichnis

	Einleitung	21
I	Dateien und Streams	27
I.1	Dateien und Dateiverzeichnisse	27
	☆☆ Aufgabe 1.1: Die Konstruktoren und Methoden der Klasse File	28
	☆ Aufgabe 1.2: Auflisten von Einträgen aus Dateiverzeichnissen	29
	☆☆ Aufgabe 1.3: Die Klassen FileFilter und FilenameFilter	29
I.2	Die Definition und Klassifikation von Streams	30
	☆☆ Aufgabe 1.4: Operationen mit primitiven Datentypen	32
I.3	Die gemeinsamen Methoden zum Schreiben und Lesen der Oberklassen OutputStream und Writer bzw. InputStream und Reader	36
I.4	Die Klassen FileInputStream und FileReader bzw. FileOutputStream und FileWriter und der Zugriff auf Dateien	37
	☆ Aufgabe 1.5: Die Klassen FileOutputStream und FileInputStream	37
	☆ Aufgabe 1.6: Die Klassen FileWriter und FileReader	38
I.5	Der Zugriff auf Daten aus dem Arbeitsspeicher mithilfe der Klassen ByteArrayOutputStream und CharArrayWriter bzw. ByteArrayInputStream und CharArrayReader	38
	☆☆ Aufgabe 1.7: Die Klassen ByteArrayOutputStream und ByteArrayInputStream	39
	☆☆ Aufgabe 1.8: Die Klassen CharArrayWriter und CharArrayReader	40
I.6	Die Filterklassen FilterInputStream und FilterReader bzw. FilterOutputStream und FilterWriter	40
	☆☆☆ Aufgabe 1.9: Das Ketten von FilterOutputStreams mit FileOutputStreams	41

	☆☆	Aufgabe 1.10: Das Ketten von FilterOutputStreams mit der System.out-Instanz.	42
	☆☆	Aufgabe 1.11: Das Ketten von FilterWriter-Streams mit der System.out-Instanz.	43
	☆☆	Aufgabe 1.12: Das Ketten von FilterInputStreams mit FileInputStreams	44
	☆☆	Aufgabe 1.13: Das Ketten von FilterReader-Streams mit FileReader-Streams.	45
1.7		Das Puffern von Daten und die Klassen BufferedOutputStream und BufferedWriter bzw. BufferedInputStream und BufferedReader	45
	☆☆	Aufgabe 1.14: Das Ketten von BufferedInputStreams und BufferedOutputStreams mit FileInputStreams ..	46
	☆☆	Aufgabe 1.15: Das Ketten von BufferedReader-Streams und BufferedWriter-Streams mit FileReader-Streams	47
	☆☆	Aufgabe 1.16: Die readLine()-Methode der Klasse BufferedReader und das Selektieren von Textzeilen	47
1.8		Die Schnittstellen DataOutput und DataInput und deren Methoden zum Schreiben und Lesen von primitiven Datentypen	48
	☆☆	Aufgabe 1.17: Das Schreiben und Lesen von primitiven Datentypen	49
	☆☆	Aufgabe 1.18: Das Filtern von primitiven Datentypen.	50
1.9		Formatierte Ein-/Ausgaben	51
1.10		Standard Ein-/Ausgabe-Kanäle.	54
	☆☆	Aufgabe 1.19: Die Klasse Formatter und das Formatieren von Ausgaben.	55
	☆☆	Aufgabe 1.20: Die Klasse Scanner und das Suchen, Ersetzen und Zerlegen von Zeichenketten	56
	☆	Aufgabe 1.21: Die System.out-Instanz der Klasse PrintStream	57
	☆	Aufgabe 1.22: Andere PrintStream-Instanzen.	58
	☆	Aufgabe 1.23: Die printf()-Methode	58
	☆	Aufgabe 1.24: Die Klasse PrintWriter.	58

	☆☆	Aufgabe 1.25: Die Klasse PrintWriter als Filterklasse einsetzen	59
	☆☆☆	Aufgabe 1.26: Die Klasse StringWriter	59
	☆	Aufgabe 1.27: Die System.in-Instanz der Klasse InputStream	60
	☆	Aufgabe 1.28: Das Umlenken des Standard-Eingabe-Kanals	61
	☆	Aufgabe 1.29: Das Einlesen von Tastatureingaben in ein Array	61
	☆	Aufgabe 1.30: Das Puffern von Tastatureingaben beim Einlesen	62
	☆☆	Aufgabe 1.31: Das Schreiben auf den Fehlerkanal mit der System.err-Instanz	62
	☆☆	Aufgabe 1.32: Einlesen von primitiven Datentypen mithilfe der Klasse Scanner	63
	☆	Aufgabe 1.33: Einlesen von Tastatureingaben mithilfe der Klasse Scanner	64
	☆	Aufgabe 1.34: Die printf()-Methode der Klasse Console	65
	☆	Aufgabe 1.35: Die readLine()-Methode der Klasse Console	65
I.11		ObjectStreams und die Serialisierung von Objekten	65
I.12		Die Versionsverwaltung von Klassen	66
	☆	Aufgabe 1.36: Das Speichern und Einlesen eines Objektes	67
	☆☆	Aufgabe 1.37: Das Speichern und Einlesen von Objekten, deren Klassen Referenzfelder definieren. . .	67
	☆☆	Aufgabe 1.38: Die serialVersionUID einer Klasse . . .	68
I.13		Die Klasse RandomAccessFile	69
	☆	Aufgabe 1.39: Wahlfreier Zugriff auf eine Datei	70
I.14		Erweiterung der I/O-Funktionalität mit Java 7	71
	☆	Aufgabe 1.40: Das Interface Path und die Klasse Files von Java 7	74
	☆	Aufgabe 1.41: Die Klassen FileChannel und AsynchronousFileChannel	75
I.15		Lösungen	76
		Lösung I.1	76
		Lösung I.2	79

Lösung I.3	80
Lösung I.4	82
Lösung I.5	84
Lösung I.6	85
Lösung I.7	86
Lösung I.8	88
Lösung I.9	90
Lösung I.10	92
Lösung I.11	93
Lösung I.12	94
Lösung I.13	96
Lösung I.14	97
Lösung I.15	99
Lösung I.16	100
Lösung I.17	101
Lösung I.18	104
Lösung I.19	105
Lösung I.20	106
Lösung I.21	108
Lösung I.22	109
Lösung I.23	110
Lösung I.24	111
Lösung I.25	111
Lösung I.26	113
Lösung I.27	116
Lösung I.28	117
Lösung I.29	117
Lösung I.30	118
Lösung I.31	119
Lösung I.32	121
Lösung I.33	123
Lösung I.34	123
Lösung I.35	124
Lösung I.36	125
Lösung I.37	127
Lösung I.38	129
Lösung I.39	133
Lösung I.40	135
Lösung I.41	141

2	Multithreading	145
2.1	Programme, Prozesse und Threads	145
	☆ Aufgabe 2.1: Thread-Definition als Unterklasse der Klasse Thread	148
	☆ Aufgabe 2.2: Thread-Definition mit Hilfe der Runnable-Schnittstelle	149
	☆☆ Aufgabe 2.3: Thread-Ausführungsintervalle	149
	☆ Aufgabe 2.4: Der main-Thread und von ihm gestartete Benutzerthreads	150
2.2	Vordergrund- und Hintergrundthreads und das Setzen von Prioritäten; Threadzustände; Threads beenden und unterbrechen	151
2.3	Die Klasse ThreadGroup	154
	☆ Aufgabe 2.5: Threadprioritäten setzen, den Status von Threads abfragen und Thread-StackTraces ausgeben	155
	☆☆ Aufgabe 2.6: Das Beenden von Threads über ein vom Benutzer definiertes Interrupt-Flag.	156
	☆☆ Aufgabe 2.7: Das Beenden von Threads über ein vom System definiertes Interrupt-Flag	157
	☆ Aufgabe 2.8: Die Klasse ThreadGroup und der UncaughtException-Handler	157
	☆ Aufgabe 2.9: Vom Benutzer definierte Threadgruppen	159
2.4	Die Klassen java.util.Timer und java.util.TimerTask	159
	☆☆ Aufgabe 2.10: Timer- und TimerTask-Instanzen	160
2.5	Die Threads von grafischen Benutzeroberflächen	161
	☆ Aufgabe 2.11: Die Threads von grafischen Benutzeroberflächen anzeigen	163
	☆ Aufgabe 2.12: Der AWT-EventQueue-Thread und die Ereignisbehandlung	163
	☆ Aufgabe 2.13: Die Methode invokeAndWait()	164
	☆ Aufgabe 2.14: Die Methode invokeLater()	165
	☆ Aufgabe 2.15: Ein Timer, der periodisch ein ActionEvent-Objekt sendet	165
	☆ Aufgabe 2.16: Das Erstellen einer GUI mit dem Ereignisbehandlungs-Thread (EDT)	166

	☆	Aufgabe 2.17: Das Editieren von Swing-Komponenten und die Methode <code>invokeLater()</code>	166
	☆	Aufgabe 2.18: Das Editieren von Swing-Komponenten und die Methode <code>invokeAndWait()</code>	167
2.6		Wiederholungsaufgaben	167
	☆☆☆	Aufgabe 2.19: Wiederholungsaufgabe	167
	☆☆	Aufgabe 2.20: Wiederholungsaufgabe	168
	☆☆	Aufgabe 2.21: Wiederholungsaufgabe	169
	☆☆	Aufgabe 2.22: Wiederholungsaufgabe	169
2.7		Die Klasse <code>SwingWorker</code>	170
	☆	Aufgabe 2.23: Das Zusammenspiel der Methoden <code>doInBackground()</code> , <code>done()</code> und <code>get()</code> der Klasse <code>SwingWorker</code>	173
	☆	Aufgabe 2.24: Das Editieren von Swing-Komponenten mit den Methoden der Klasse <code>SwingWorker</code>	174
	☆	Aufgabe 2.25: Das Zusammenspiel der Methoden <code>publish()</code> und <code>process()</code> der Klasse <code>SwingWorker</code>	174
	☆☆	Aufgabe 2.26: Wiederholungsaufgabe	175
	☆☆	Aufgabe 2.27: Wiederholungsaufgabe	176
2.8		Threadsynchronisation durch implizite Locks (<code>synchronized</code> -Blöcke und -Methoden)	177
2.9		Deadlocks, Livelocks und Starvation	186
	☆	Aufgabe 2.28: Threadkommunikation über die Werte von Klassenfeldern	188
	☆	Aufgabe 2.29: Threadkommunikation über <code>PipedWriter</code> - und <code>PipedReader</code> -Streams	189
	☆☆	Aufgabe 2.30: Threadkommunikation über <code>PipedInputStream</code> - und <code>PipedOutputStream</code> -Instanzen	190
	☆	Aufgabe 2.31: Threadkommunikation und Synchronisation über die Methode <code>join()</code> der Thread-Klasse	191
	☆	Aufgabe 2.32 : Der Unterschied zwischen Klassen- und Objekt-Sperre	191
	☆	Aufgabe 2.33: »Klassen-Sperre« mit einem <code>synchronized</code> -Block setzen	192

	☆	Aufgabe 2.34: »Klassen-Sperre« mit einer synchronized-Methode setzen	193
	☆☆	Aufgabe 2.35: Das Deadlock-Problem	194
	☆☆☆	Aufgabe 2.36: Wiederholungsaufgabe	195
	☆☆	Aufgabe 2.37: Die Benutzung von volatile in der »Veröffentlichung« von Referenzen	197
2.10		Threadsynchronisation mit Signalen (die Methoden wait(), notify() und notifyAll())	197
	☆☆	Aufgabe 2.38: Die Methoden wait() und notify()	198
	☆☆	Aufgabe 2.39: Die Methoden wait() und notifyAll() ...	200
2.11		»Immutable«-Objekte	201
	☆	Aufgabe 2.40: »Immutable«-Objekte	206
	☆☆	Aufgabe 2.41: »Immutable«-Klassen, deren Instanzfelder Referenzen auf veränderliche Instanzen halten	208
2.12		Threadsynchronisation durch explizite Locks (die Interfaces Lock, Condition und BlockingQueue)	210
	☆	Aufgabe 2.42: Synchronisieren mit den Methoden des Interface Lock	213
	☆☆	Aufgabe 2.43: Producer-Consumer mit Locks und Conditions	214
	☆☆	Aufgabe 2.44: Wiederholungsaufgabe	216
	☆☆	Aufgabe 2.45: Producer-Consumer mit BlockingQueue-Interface	217
2.13		Die neuen Interfaces Callable, Future und Executor	217
	☆☆	Aufgabe 2.46: Callables versus Runnables und das Executor Framework	222
	☆☆	Aufgabe 2.47: Die Klasse ScheduledThread- PoolExecutor versus Timer	226
2.14		Das Fork/Join-Framework	227
	☆☆	Aufgabe 2.48: Die Klasse ServiceAction versus ServiceTask	228
	☆☆	Aufgabe 2.49: Wiederholungsaufgabe	230
	☆☆	Aufgabe 2.50: Wiederholungsaufgabe	231
2.15		Lösungen	231
		Lösung 2.1	231
		Lösung 2.2	232

Lösung 2.3	232
Lösung 2.4	234
Lösung 2.5	235
Lösung 2.6	237
Lösung 2.7	239
Lösung 2.8	241
Lösung 2.9	244
Lösung 2.10	246
Lösung 2.11	249
Lösung 2.12	252
Lösung 2.13	253
Lösung 2.14	255
Lösung 2.15	256
Lösung 2.16	257
Lösung 2.17	258
Lösung 2.18	260
Lösung 2.19	262
Lösung 2.20	266
Lösung 2.21	269
Lösung 2.22	271
Lösung 2.23	274
Lösung 2.24	275
Lösung 2.25	277
Lösung 2.26	279
Lösung 2.27	282
Lösung 2.28	285
Lösung 2.29	286
Lösung 2.30	288
Lösung 2.31	291
Lösung 2.32	292
Lösung 2.33	294
Lösung 2.34	296
Lösung 2.35	297
Lösung 2.36	300
Lösung 2.37	304
Lösung 2.38	305
Lösung 2.39	308
Lösung 2.40	310
Lösung 2.41	315

	Lösung 2.42.	324
	Lösung 2.43.	328
	Lösung 2.44.	333
	Lösung 2.45.	336
	Lösung 2.46.	339
	Lösung 2.47.	359
	Lösung 2.48.	362
	Lösung 2.49.	367
	Lösung 2.50.	372
3	Einführung in Java-Applets	381
3.1	Applets erzeugen und starten.	381
	☆ Aufgabe 3.1: Die Methoden init(), start(), stop() und destroy() der Applet-Klassen.	383
	☆ Aufgabe 3.2: Eine Graphik im Applet zeichnen	383
	☆ Aufgabe 3.3: Parameterübergabe an Applets mithilfe des <param>-Tags der HTML-Datei	383
	☆ Aufgabe 3.4: Die Methode getParameterInfo() der Klasse Applet.	384
	☆ Aufgabe 3.5: AWT-Komponenten zu einem Applet hinzufügen	384
	☆ Aufgabe 3.6: Swing-Komponenten zu einem Applet hinzufügen	385
	☆ Aufgabe 3.7: Mit einem Graphics2D-Objekt auf ein Applet zeichnen	386
	☆☆ Aufgabe 3.8: Die Klassen GeneralPath und AffineTransform	386
3.2	Threads und Eventbehandlung	387
	☆ Aufgabe 3.9: Die Threads von Applets anzeigen	387
	☆ Aufgabe 3.10: Das Erstellen einer GUI mit dem Ereignisbehandlungs-Thread (EDT)	388
	☆☆ Aufgabe 3.11: Die Eventbehandlung in Applets und die Methoden paint() und update() von AWT- Komponenten	388
3.3	Lösungen	390
	Lösung 3.1	390
	Lösung 3.2	391
	Lösung 3.3	392

	Lösung 3.4	393
	Lösung 3.5	395
	Lösung 3.6	396
	Lösung 3.7	398
	Lösung 3.8	400
	Lösung 3.9	402
	Lösung 3.10	404
	Lösung 3.11	405
4	Multimedia für Applets und Applikationen	411
4.1	Die Einbindung von Images und Sounds	411
	☆ Aufgabe 4.1: Bild- und Sounddateien im Applet laden	417
	☆ Aufgabe 4.2: Die Methoden getCodeBase() und getDocumentBase() der Klasse Applet	418
	☆ Aufgabe 4.3: Pakete in Applets	418
	☆ Aufgabe 4.4: Sounds in einer Applikation abspielen	419
	☆☆ Aufgabe 4.5: Die Klassen BufferedImage und TexturePaint	419
	☆☆☆ Aufgabe 4.6: Bild- und Sounddateien in einem Applet laden – eine Übersicht	420
	☆☆☆ Aufgabe 4.7: Bild- und Sounddateien in einem Fenster laden – eine Übersicht	423
	☆☆ Aufgabe 4.8: Bild- und Sounddateien für Applets aus einem JAR-Archiv laden	423
	☆☆ Aufgabe 4.9: Bild- und Sounddateien für Frames aus einem JAR-Archiv laden	424
4.2	Das Überwachen des Ladevorgangs	425
	☆ Aufgabe 4.10: Die Methoden prepareImage() und checkImage() der Klasse Toolkit	426
	☆ Aufgabe 4.11: Die Klasse MediaTracker	426
	☆☆ Aufgabe 4.12: Das Laden von Images mit dem MediaTracker überwachen	427
	☆☆ Aufgabe 4.13: Die Methode imageUpdate() von Component überschreiben	428
	☆☆ Aufgabe 4.14: Das Laden von Images mit dem ImageObserver überwachen	428

	☆☆	Aufgabe 4.15: Den ImageObserver mittels einer anonymen Klasse implementieren	429
	☆☆	Aufgabe 4.16: Ein benutzerdefinierter ImageObserver und die Methode setImageObserver() der Klasse ImageIcon	429
	☆☆	Aufgabe 4.17: Der Standard-ImageObserver und die Methode setImageObserver() der Klasse ImageIcon	430
4.3		Die RootPane-Container von Applets	431
	☆☆☆	Aufgabe 4.18: Die RootPane-Container für JApplet-Instanzen ermitteln und setzen	431
	☆☆☆	Aufgabe 4.19: Wiederholungsaufgabe	432
4.4		Lösungen	433
		Lösung 4.1	433
		Lösung 4.2	436
		Lösung 4.3	437
		Lösung 4.4	438
		Lösung 4.5	440
		Lösung 4.6	442
		Lösung 4.7	448
		Lösung 4.8	454
		Lösung 4.9	456
		Lösung 4.10	458
		Lösung 4.11	460
		Lösung 4.12	462
		Lösung 4.13	464
		Lösung 4.14	465
		Lösung 4.15	468
		Lösung 4.16	471
		Lösung 4.17	474
		Lösung 4.18	478
		Lösung 4.19	480
5		Threads und Animation	483
5.1		Applets und die Definition von Arbeitsthreads.	483
	☆☆	Aufgabe 5.1: Ein Applet, welches die Runnable-Schnittstelle implementiert.	483
	☆☆	Aufgabe 5.2: Das Beenden von Threads in der stop()-Methode eines Applets.	484

	☆☆☆ Aufgabe 5.3: SwingWorker- und Timer-Tasks für Applets	485
	☆☆☆ Aufgabe 5.4: Wiederholungsaufgabe	487
	☆☆☆ Aufgabe 5.5: Laden von mehreren Sounddateien parallel zum Abspielen von Sounds.	487
	☆☆☆ Aufgabe 5.6: Wiederholungsaufgabe	488
	☆☆☆ Aufgabe 5.7: Laden von mehreren Imagedateien parallel zum Zeichnen von Images	489
	☆☆☆ Aufgabe 5.8: Wiederholungsaufgabe	490
5.2	Synchronisation und Kommunikation von Threads in Applets	491
	☆☆☆ Aufgabe 5.9: Mehrere Threads, die über eine Monitor-Klasse synchronisiert werden, zum Laden von Sounddateien starten	491
	☆☆ Aufgabe 5.10: Eine einfache Animation	492
	☆☆ Aufgabe 5.11: Wiederholungsaufgabe	493
	☆☆ Aufgabe 5.12: Wiederholungsaufgabe	494
	☆☆☆ Aufgabe 5.13: Animation mit Timer und Image-Komponenten	495
	☆☆☆ Aufgabe 5.14: Wiederholungsaufgabe	496
	☆☆ Aufgabe 5.15: Animation mit Timer und ImageIcon-Komponenten.	497
	☆☆ Aufgabe 5.16: Wiederholungsaufgabe	497
	☆☆☆ Aufgabe 5.17: Animation mit mehreren Threads	497
	☆☆ Aufgabe 5.18: Applet mit »Klassen-Sperre« für Threads.	498
	☆☆ Aufgabe 5.19: Wiederholungsaufgabe	500
	☆☆☆ Aufgabe 5.20: Applet mit »Objekt-Sperre« für Threads.	500
	☆☆ Aufgabe 5.21: Wiederholungsaufgabe	502
	☆☆ Aufgabe 5.22: Applet mit SwingWorkerTasks.	503
	☆☆☆ Aufgabe 5.23: Threadkommunikation mit Warten auf das Threadende.	504
	☆☆☆ Aufgabe 5.24: Threadkommunikation mit Rückruf nach Ausführung der Aufträge	505
	☆☆☆ Aufgabe 5.25: Threadkommunikation mit Callables	506

5.3	Lösungen	508
	Lösung 5.1	508
	Lösung 5.2	510
	Lösung 5.3	513
	Lösung 5.4	519
	Lösung 5.5	525
	Lösung 5.6	528
	Lösung 5.7	530
	Lösung 5.8	533
	Lösung 5.9	537
	Lösung 5.10	540
	Lösung 5.11	543
	Lösung 5.12	546
	Lösung 5.13	549
	Lösung 5.14	553
	Lösung 5.15	556
	Lösung 5.16	559
	Lösung 5.17	561
	Lösung 5.18	565
	Lösung 5.19	568
	Lösung 5.20	571
	Lösung 5.21	575
	Lösung 5.22	577
	Lösung 5.23	579
	Lösung 5.24	584
	Lösung 5.25	588
6	Java-Applets und Webbrowser	595
6.1	Zugriffsrechte und Sicherheitsbestimmungen	595
6.2	Kommunikation zwischen Applets sowie zwischen Applets und Webbrowser	599
	☆☆☆ Aufgabe 6.1: Der target-Parameter der Methode showDocument()	600
	☆☆☆ Aufgabe 6.2: Zugriffsrechte für Applets vergeben	601
	☆☆☆ Aufgabe 6.3: Wiederholungsaufgabe	602
	☆☆ Aufgabe 6.4: Kommunikation zwischen Applet und Browser	603
	☆☆ Aufgabe 6.5: Benutzung von Image-Maps in der Kommunikation zwischen Applet und Browser	604

	☆☆	Aufgabe 6.6: Kommunikation zwischen Applets in derselben Webseite	605
	☆☆	Aufgabe 6.7: Benutzung von Image-Maps in der Kommunikation zwischen Applets	606
	☆☆	Aufgabe 6.8: Redirects im Webbrowser	606
	☆☆☆	Aufgabe 6.9: Die Threads von Applets synchronisieren	607
6.3		Lösungen	609
		Lösung 6.1	609
		Lösung 6.2	610
		Lösung 6.3	613
		Lösung 6.4	618
		Lösung 6.5	621
		Lösung 6.6	624
		Lösung 6.7	626
		Lösung 6.8	630
		Lösung 6.9	633
7		Netzwerkprogrammierung	641
7.1		Einführung in die Netzwerkprogrammierung	641
7.2		URL-Verbindungen	642
	☆☆	Aufgabe 7.1: Die Klasse JEditorPane und ein einfacher Webbrowser	644
	☆	Aufgabe 7.2: Die Methoden der Klasse URL	644
	☆☆	Aufgabe 7.3: Die Methoden der Klassen Network- Interface, InetAddress und InterfaceAddress	646
	☆	Aufgabe 7.4: Die openStream()-Methode der URL-Klasse	647
	☆	Aufgabe 7.5: Wiederholungsaufgabe	647
	☆	Aufgabe 7.6: Die URLConnection-Klasse und ihre Methoden	648
	☆☆	Aufgabe 7.7: E-Mails über eine URLConnection- Instanz verschicken	649
7.3		Client-Server-Kommunikation über TCP mit Java-Sockets	650
	☆☆☆	Aufgabe 7.8: Kommunikation über TCP mit Java-Sockets	651
	☆☆☆	Aufgabe 7.9: Wiederholungsaufgabe	653

7.4	Client-Server-Kommunikation über UDP mit Java-Sockets	654
	☆☆☆ Aufgabe 7.10: Kommunikation über UDP mit Java-Sockets	655
	☆☆☆ Aufgabe 7.11: Wiederholungsaufgabe	656
7.5	Webbasierte Client-Server-Kommunikation	657
	☆☆ Aufgabe 7.12: Ein einfacher Web-Client, der über TCP-Socket mit einem Webserver kommuniziert	658
	☆☆ Aufgabe 7.13: Ein einfacher Webserver	659
	☆☆ Aufgabe 7.14: Der HttpServer von Java 6	660
	☆☆ Aufgabe 7.15: Wiederholungsaufgabe	662
7.6	Sequentielle und parallele Server	664
	☆☆ Aufgabe 7.16: Ein sequentieller Echo-Server	664
	☆☆☆ Aufgabe 7.17: Ein paralleler Echo-Server mit dynamischer Parallelität	665
	☆☆☆ Aufgabe 7.18: Ein paralleler Echo-Server mit statischer Parallelität	666
	☆☆☆ Aufgabe 7.19: Wiederholungsaufgabe	667
	☆☆☆ Aufgabe 7.20: Wiederholungsaufgabe	668
7.7	Updates mit Java 6 zum Thema Netzwerkprogrammierung	668
	☆☆☆ Aufgabe 7.21: Die Klasse CookieHandler	669
	☆☆☆ Aufgabe 7.22: Die Klassen CookieManager, CookieStore und CookiePolicy	670
7.8	Neue Features von Java 7 zum Thema Netzwerkprogrammierung	671
	☆☆☆ Aufgabe 7.23: Die Klassen SocketChannel und ServerSocketChannel	674
	☆☆☆ Aufgabe 7.24: Die Klassen AsynchronousSocket- Channel und AsynchronousServerSocketChannel	675
	☆☆ Aufgabe 7.26: Die Klasse DatagramChannel	676
7.9	Lösungen	677
	Lösung 7.1	677
	Lösung 7.2	679
	Lösung 7.3	682
	Lösung 7.4	687
	Lösung 7.5	689
	Lösung 7.6	690
	Lösung 7.7	692
	Lösung 7.8	695

Lösung 7.9	700
Lösung 7.10	706
Lösung 7.11	710
Lösung 7.12	717
Lösung 7.13	720
Lösung 7.14	724
Lösung 7.15	728
Lösung 7.16	739
Lösung 7.17	743
Lösung 7.18	748
Lösung 7.19	752
Lösung 7.20	758
Lösung 7.21	763
Lösung 7.22	766
Lösung 7.23	768
Lösung 7.24	774
Lösung 7.25	780
Stichwortverzeichnis	785