

Inhalt

1 Einleitung	13
1.1 Parallelität, Nebenläufigkeit und Verteilung.....	13
1.2 Programme, Prozesse und Threads.....	14
2 Grundlegende Synchronisationskonzepte in Java	19
2.1 Erzeugung und Start von Java-Threads	19
2.1.1 Ableiten der Klasse Thread	19
2.1.2 Implementieren der Schnittstelle Runnable.....	21
2.1.3 Einige Beispiele	22
2.2 Probleme beim Zugriff auf gemeinsam genutzte Objekte.....	28
2.2.1 Erster Lösungsversuch	31
2.2.2 Zweiter Lösungsversuch.....	33
2.3 Synchronized und volatile	35
2.3.1 Synchronized-Methoden	35
2.3.2 Synchronized-Blöcke	36
2.3.3 Wirkung von synchronized	37
2.3.4 Notwendigkeit von synchronized	39
2.3.5 Volatile.....	40
2.3.6 Regel für die Nutzung von synchronized	40
2.4 Ende von Java-Threads.....	42
2.4.1 Asynchrone Beauftragung mit Abfragen der Ergebnisse	43
2.4.2 Zwangsweises Beenden von Threads	48
2.4.3 Asynchrone Beauftragung mit befristetem Warten	53
2.4.4 Asynchrone Beauftragung mit Rückruf (Callback).....	54
2.5 Wait und notify	57
2.5.1 Erster Lösungsversuch	57
2.5.2 Zweiter Lösungsversuch.....	58
2.5.3 Korrekte Lösung mit wait und notify	59
2.6 NotifyAll	67
2.6.1 Erzeuger-Verbraucher-Problem mit wait und notify	68
2.6.2 Erzeuger-Verbraucher-Problem mit wait und notifyAll	71
2.6.3 Faires Parkhaus mit wait und notifyAll.....	74
2.7 Prioritäten von Threads.....	75

2.8 Thread-Gruppen	82
2.9 Vordergrund- und Hintergrund-Threads.....	86
2.10 Weitere „gute“ und „schlechte“ Thread-Methoden	88
2.11 Thread-lokale Daten	89
2.12 Zusammenfassung	92
3 Fortgeschrittene Synchronisationskonzepte in Java	97
3.1 Semaphore.....	98
3.1.1 Einfache Semaphore	98
3.1.2 Einfache Semaphore für den gegenseitigen Ausschluss.....	99
3.1.3 Einfache Semaphore zur Herstellung vorgegebener Ausführungsreihenfolgen	101
3.1.4 Additive Semaphore.....	104
3.1.5 Semaphorgruppen	106
3.2 Message Queues	108
3.2.1 Verallgemeinerung des Erzeuger-Verbraucher-Problems	108
3.2.2 Übertragung des erweiterten Erzeuger-Verbraucher-Problems auf Message Queues ..	111
3.3 Pipes	113
3.4 Philosophen-Problem	116
3.4.1 Lösung mit synchronized – wait – notifyAll.....	117
3.4.2 Naive Lösung mit einfachen Semaphoren.....	119
3.4.3 Einschränkende Lösung mit gegenseitigem Ausschluss	120
3.4.4 Gute Lösung mit einfachen Semaphoren	121
3.4.5 Lösung mit Semaphorgruppen	124
3.5 Leser-Schreiber-Problem.....	126
3.5.1 Lösung mit synchronized – wait – notifyAll.....	127
3.5.2 Lösung mit additiven Semaphoren.....	130
3.6 Schablonen zur Nutzung der Synchronisationsprimitive und Konsistenzbetrachtungen	131
3.7 Concurrent-Klassenbibliothek aus Java 5.....	135
3.7.1 Executors	135
3.7.2 Locks und Conditions	141
3.7.3 Atomic-Klassen.....	148
3.7.4 Synchronisationsklassen	149
3.7.5 Queues	151
3.8 Ursachen für Verklemmungen.....	153
3.8.1 Beispiele für Verklemmungen mit synchronized	154
3.8.2 Beispiele für Verklemmungen mit Semaphoren	157
3.8.3 Bedingungen für das Eintreten von Verklemmungen	158
3.9 Vermeidung von Verklemmungen	159
3.9.1 Anforderung von Betriebsmitteln „auf einen Schlag“.....	162
3.9.2 Anforderung von Betriebsmitteln gemäß einer vorgegebenen Ordnung	163
3.9.3 Anforderung von Betriebsmitteln mit Bedarfsanalyse	164
3.10 Modellierung mit Petri-Netzen	171
3.10.1 Petri-Netze	171
3.10.2 Modellierung der Nutzung von Synchronized-Methoden	173
3.10.3 Modellierung von wait, notify und notifyAll	176
3.11 Zusammenfassung	179

4 Parallelität und grafische Benutzeroberflächen	181
4.1 Einführung in die Programmierung grafischer Benutzeroberflächen mit Swing	182
4.1.1 Einige erste Beispiele	182
4.1.2 Ereignisbehandlung	186
4.1.3 Container	190
4.1.4 Primitive Interaktionselemente	193
4.1.5 Grafikprogrammierung	194
4.1.6 Applets	199
4.2 MVC	201
4.2.1 Prinzip von MVC	202
4.2.2 MVC für die Entwicklung eigener Programme	205
4.2.3 MVC in Swing	210
4.3 Threads und Swing	212
4.4 Zusammenfassung	223
5 Verteilte Anwendungen mit Sockets	225
5.1 Einführung in das Themengebiet der Rechnernetze	226
5.1.1 Schichtenmodell	226
5.1.2 IP-Adressen und DNS-Namen	230
5.1.3 Das Transportprotokoll UDP	231
5.1.4 Das Transportprotokoll TCP	233
5.2 Socket-Schnittstelle	234
5.2.1 Socket-Schnittstelle zu UDP	234
5.2.2 Socket-Schnittstelle zu TCP	235
5.2.3 Socket-Schnittstelle für Java	238
5.3 Kommunikation über UDP mit Java-Sockets	239
5.4 Multicast-Kommunikation mit Java-Sockets	246
5.5 Kommunikation über TCP mit Java-Sockets	250
5.6 Sequenzielle und parallele Server	260
5.6.1 Server mit dynamischer Parallelität	262
5.6.2 Server mit statischer Parallelität	266
5.7 Zusammenfassung	271
6 Verteilte Anwendungen mit RMI	273
6.1 Prinzip von RMI	273
6.2 Einführendes RMI-Beispiel	276
6.2.1 Basisprogramm	276
6.2.2 RMI-Client mit grafischer Benutzeroberfläche	280
6.2.3 RMI-Registry	283
6.3 Parallelität bei RMI-Methodenaufrufen	287
6.4 Wertübergabe für Parameter und Rückgabewerte	291
6.4.1 Serialisierung und Deserialisierung von Objekten	292
6.4.2 Serialisierung und Deserialisierung bei RMI	296
6.5 Referenzübergabe für Parameter und Rückgabewerte	300
6.6 Transformation lokaler in verteilte Anwendungen	316
6.6.1 Rechnergrenzen überschreitende Synchronisation mit RMI	317

Inhalt

6.6.2 Asynchrone Kommunikation mit RMI.....	319
6.6.3 Verteilte MVC-Anwendungen mit RMI	319
6.7 Dynamisches Umschalten zwischen Wert- und Referenzübergabe – Migration von Objekten..	321
6.7.1 Das Exportieren und „Unexportieren“ von Objekten.....	321
6.7.2 Migration von Objekten.....	324
6.7.3 Eintrag eines Nicht-Stub-Objekts in die RMI-Registry.....	331
6.8 Laden von Klassen über das Netz.....	332
6.9 Realisierung von Stubs und Skeletons.....	333
6.9.1 Realisierung von Skeletons	333
6.9.2 Realisierung von Stubs.....	334
6.10 Verschiedenes.....	336
6.11 Zusammenfassung	337
7 Webbasierte Anwendungen mit Servlets und JSP.....	339
7.1 HTTP.....	340
7.1.1 GET.....	340
7.1.2 Formulare.....	343
7.1.3 POST.....	345
7.1.4 Format von HTTP-Anfragen und -Antworten.....	346
7.2 Einführende Servlet-Beispiele	347
7.2.1 Allgemeine Vorgehensweise.....	347
7.2.2 Erstes Servlet-Beispiel	348
7.2.3 Zugriff auf Formulardaten.....	355
7.2.4 Zugriff auf die Daten der HTTP-Anfrage und -Antwort.....	356
7.3 Parallelität bei Servlets	357
7.3.1 Demonstration der Parallelität von Servlets.....	357
7.3.2 Paralleler Zugriff auf Daten	359
7.3.3 Anwendungsglobale Daten	362
7.4 Sessions und Cookies	365
7.4.1 Sessions.....	366
7.4.2 Realisierung von Sessions mit Cookies.....	371
7.4.3 Direkter Zugriff auf Cookies.....	372
7.4.4 Servlets mit länger dauernden Aufträgen	374
7.5 Übertragung von Dateien mit Servlets.....	378
7.5.1 Herunterladen von Dateien	378
7.5.2 Hochladen von Dateien.....	381
7.6 JSP (Java Server Pages).....	384
7.6.1 Scripting-Elemente.....	384
7.6.2 Direktiven	386
7.6.3 Aktionen.....	387
7.7 MVC-Prinzip mit Servlets und JSPs.....	390
7.8 MVC-Prinzip mit AJAX und GWT.....	397
7.9 Zusammenfassung	405
Literatur	407
Register.....	409