

Inhalt

| | | |
|----------|--|----|
| 1 | Einleitung | 15 |
| 1.1 | Parallelität, Nebenläufigkeit und Verteilung | 15 |
| 1.2 | Programme, Prozesse und Threads | 16 |
| 2 | Grundlegende Synchronisationskonzepte in Java | 20 |
| 2.1 | Erzeugung und Start von Java-Threads | 20 |
| 2.1.1 | Ableiten der Klasse Thread | 20 |
| 2.1.2 | Implementieren der Schnittstelle Runnable | 22 |
| 2.1.3 | Einige Beispiele | 23 |
| 2.2 | Probleme beim Zugriff auf gemeinsam genutzte Objekte | 30 |
| 2.2.1 | Erster Lösungsversuch | 33 |
| 2.2.2 | Zweiter Lösungsversuch | 34 |
| 2.3 | Synchronized und volatile | 36 |
| 2.3.1 | Synchronized-Methoden | 36 |
| 2.3.2 | Synchronized-Blöcke | 37 |
| 2.3.3 | Wirkung von synchronized | 39 |
| 2.3.4 | Notwendigkeit von synchronized | 40 |
| 2.3.5 | Volatile | 41 |
| 2.3.6 | Regel für die Nutzung von synchronized | 42 |
| 2.4 | Ende von Java-Threads | 43 |
| 2.4.1 | Asynchrone Beauftragung mit Abfragen der Ergebnisse | 44 |
| 2.4.2 | Zwangswises Beenden von Threads | 49 |
| 2.4.3 | Asynchrone Beauftragung mit befristetem Warten | 55 |
| 2.4.4 | Asynchrone Beauftragung mit Rückruf (Callback) | 56 |
| 2.4.5 | Asynchrone Beauftragung mit Rekursion | 59 |
| 2.5 | Wait und notify | 63 |
| 2.5.1 | Erster Lösungsversuch | 64 |
| 2.5.2 | Zweiter Lösungsversuch | 64 |
| 2.5.3 | Dritter Lösungsversuch | 66 |
| 2.5.4 | Korrekte und effiziente Lösung mit wait und notify | 67 |

| | | |
|----------|--|------------|
| 2.6 | NotifyAll | 74 |
| 2.6.1 | Erzeuger-Verbraucher-Problem mit wait und notify | 75 |
| 2.6.2 | Erzeuger-Verbraucher-Problem mit wait und notifyAll | 79 |
| 2.6.3 | Faires Parkhaus mit wait und notifyAll | 81 |
| 2.7 | Prioritäten von Threads | 83 |
| 2.8 | Thread-Gruppen | 90 |
| 2.9 | Vordergrund- und Hintergrund-Threads | 95 |
| 2.10 | Weitere „gute“ und „schlechte“ Thread-Methoden | 96 |
| 2.11 | Thread-lokale Daten | 98 |
| 2.12 | Zusammenfassung | 100 |
| 3 | Fortgeschrittene Synchronisationskonzepte in Java | 105 |
| 3.1 | Semaphore | 106 |
| 3.1.1 | Einfache Semaphore | 106 |
| 3.1.2 | Einfache Semaphore für den gegenseitigen Ausschluss | 107 |
| 3.1.3 | Einfache Semaphore zur Herstellung vorgegebener Ausführungsreihenfolgen | 109 |
| 3.1.4 | Additive Semaphore | 112 |
| 3.1.5 | Semaphorgruppen | 115 |
| 3.2 | Message Queues | 118 |
| 3.2.1 | Verallgemeinerung des Erzeuger-Verbraucher-Problems | 118 |
| 3.2.2 | Übertragung des erweiterten Erzeuger-Verbraucher-Problems auf Message Queues | 120 |
| 3.3 | Pipes | 122 |
| 3.4 | Philosophen-Problem | 126 |
| 3.4.1 | Lösung mit synchronized - wait - notifyAll | 127 |
| 3.4.2 | Naive Lösung mit einfachen Semaphoren | 129 |
| 3.4.3 | Einschränkende Lösung mit gegenseitigem Ausschluss | 131 |
| 3.4.4 | Gute Lösung mit einfachen Semaphoren | 132 |
| 3.4.5 | Lösung mit Semaphorgruppen | 135 |
| 3.5 | Leser-Schreiber-Problem | 137 |
| 3.5.1 | Lösung mit synchronized - wait - notifyAll | 138 |
| 3.5.2 | Lösung mit additiven Semaphoren | 141 |
| 3.6 | Schablonen zur Nutzung der Synchronisationsprimitive und Konsistenzbetrachtungen | 143 |
| 3.7 | Concurrent-Klassenbibliothek aus Java 5 | 147 |
| 3.7.1 | Executors | 148 |
| 3.7.2 | Locks und Conditions | 154 |
| 3.7.3 | Atomic-Klassen | 162 |
| 3.7.4 | Synchronisationsklassen | 164 |
| 3.7.5 | Queues | 166 |

| | | |
|----------|--|------------|
| 3.8 | Das Fork-Join-Framework von Java 7 | 168 |
| 3.8.1 | Grenzen von ThreadPoolExecutor | 168 |
| 3.8.2 | ForkJoinPool und RecursiveTask | 170 |
| 3.8.3 | Beispiel zur Nutzung des Fork-Join-Frameworks | 171 |
| 3.9 | Ursachen für Verklemmungen | 174 |
| 3.9.1 | Beispiele für Verklemmungen mit synchronized | 175 |
| 3.9.2 | Beispiele für Verklemmungen mit Semaphoren | 178 |
| 3.9.3 | Bedingungen für das Eintreten von Verklemmungen | 179 |
| 3.10 | Vermeidung von Verklemmungen | 181 |
| 3.10.1 | Anforderung von Betriebsmitteln „auf einen Schlag“ | 183 |
| 3.10.2 | Anforderung von Betriebsmitteln gemäß einer vorgegebenen Ordnung | 184 |
| 3.10.3 | Weitere Verfahren | 185 |
| 3.11 | Zusammenfassung | 187 |
| 4 | Parallelität und grafische Benutzeroberflächen | 189 |
| 4.1 | Einführung in die Programmierung grafischer Benutzeroberflächen mit Swing | 190 |
| 4.1.1 | Einige erste Beispiele | 190 |
| 4.1.2 | Ereignisbehandlung | 194 |
| 4.1.3 | Container | 197 |
| 4.1.4 | Primitive Interaktionselemente | 200 |
| 4.1.5 | Grafikprogrammierung | 202 |
| 4.1.6 | Applets | 206 |
| 4.2 | MVC | 209 |
| 4.2.1 | Prinzip von MVC | 209 |
| 4.2.2 | MVC für die Entwicklung eigener Programme | 212 |
| 4.2.3 | MVC in Swing | 219 |
| 4.3 | Threads und Swing | 221 |
| 4.3.1 | Threads für grafische Benutzeroberflächen | 221 |
| 4.3.2 | Probleme bei längeren Ereignisbehandlungen | 223 |
| 4.3.3 | Regeln für längere Ereignisbehandlungen | 224 |
| 4.3.4 | Beispiel Stoppuhr | 225 |
| 4.3.5 | Beispiel Animation | 230 |
| 4.3.6 | SwingWorker | 233 |
| 4.4 | Zusammenfassung | 237 |
| 5 | Verteilte Anwendungen mit Sockets | 238 |
| 5.1 | Einführung in das Themengebiet der Rechnernetze | 239 |
| 5.1.1 | Schichtenmodell | 239 |
| 5.1.2 | IP-Adressen und DNS-Namen | 243 |
| 5.1.3 | Das Transportprotokoll UDP | 244 |
| 5.1.4 | Das Transportprotokoll TCP | 245 |

| | | |
|----------|--|------------|
| 5.2 | Socket-Schnittstelle | 246 |
| 5.2.1 | Socket-Schnittstelle zu UDP | 247 |
| 5.2.2 | Socket-Schnittstelle zu TCP | 248 |
| 5.2.3 | Socket-Schnittstelle für Java | 250 |
| 5.3 | Kommunikation über UDP mit Java-Sockets | 252 |
| 5.4 | Multicast-Kommunikation mit Java-Sockets | 260 |
| 5.5 | Kommunikation über TCP mit Java-Sockets | 265 |
| 5.6 | Sequenzielle und parallele Server | 275 |
| 5.6.1 | TCP-Server mit dynamischer Parallelität | 276 |
| 5.6.2 | TCP-Server mit statischer Parallelität | 280 |
| 5.6.3 | Sequenzieller, „verzahnt“ arbeitender TCP-Server | 285 |
| 5.7 | Zusammenfassung | 288 |
| 6 | Verteilte Anwendungen mit RMI | 290 |
| 6.1 | Prinzip von RMI | 290 |
| 6.2 | Einführendes RMI-Beispiel | 293 |
| 6.2.1 | Basisprogramm | 293 |
| 6.2.2 | RMI-Client mit grafischer Benutzeroberfläche | 297 |
| 6.2.3 | RMI-Registry | 300 |
| 6.3 | Parallelität bei RMI-Methodenaufrufen | 304 |
| 6.4 | Wertübergabe für Parameter und Rückgabewerte | 308 |
| 6.4.1 | Serialisierung und Deserialisierung von Objekten | 309 |
| 6.4.2 | Serialisierung und Deserialisierung bei RMI | 313 |
| 6.5 | Referenzübergabe für Parameter und Rückgabewerte | 318 |
| 6.6 | Transformation lokaler in verteilte Anwendungen | 334 |
| 6.6.1 | Rechnergrenzen überschreitende Synchronisation mit RMI | 335 |
| 6.6.2 | Asynchrone Kommunikation mit RMI | 337 |
| 6.6.3 | Verteilte MVC-Anwendungen mit RMI | 338 |
| 6.7 | Dynamisches Umschalten zwischen Wert- und Referenzübergabe - Migration von Objekten | 340 |
| 6.7.1 | Das Exportieren und „Unexportieren“ von Objekten | 340 |
| 6.7.2 | Migration von Objekten | 342 |
| 6.7.3 | Eintrag eines Nicht-Stub-Objekts in die RMI-Registry | 350 |
| 6.8 | Laden von Klassen über das Netz | 350 |
| 6.9 | Realisierung von Stubs und Skeletons | 352 |
| 6.9.1 | Realisierung von Skeletons | 352 |
| 6.9.2 | Realisierung von Stubs | 353 |
| 6.10 | Verschiedenes | 355 |
| 6.11 | Zusammenfassung | 357 |

| | | |
|----------|---|------------|
| 7 | Webbasierte Anwendungen mit Servlets und JSP | 358 |
| 7.1 | HTTP | 359 |
| 7.1.1 | GET | 359 |
| 7.1.2 | Formulare | 362 |
| 7.1.3 | POST | 364 |
| 7.1.4 | Format von HTTP-Anfragen und -Antworten | 365 |
| 7.2 | Einführende Servlet-Beispiele | 365 |
| 7.2.1 | Allgemeine Vorgehensweise | 365 |
| 7.2.2 | Erstes Servlet-Beispiel | 367 |
| 7.2.3 | Zugriff auf Formulardaten | 370 |
| 7.2.4 | Zugriff auf die Daten der HTTP-Anfrage und -Antwort | 372 |
| 7.3 | Parallelität bei Servlets | 373 |
| 7.3.1 | Demonstration der Parallelität von Servlets | 373 |
| 7.3.2 | Paralleler Zugriff auf Daten | 375 |
| 7.3.3 | Anwendungsglobale Daten | 379 |
| 7.4 | Sessions und Cookies | 382 |
| 7.4.1 | Sessions | 383 |
| 7.4.2 | Realisierung von Sessions mit Cookies | 387 |
| 7.4.3 | Direkter Zugriff auf Cookies | 389 |
| 7.4.4 | Servlets mit länger dauernden Aufträgen | 390 |
| 7.5 | Asynchrone Servlets | 396 |
| 7.6 | Filter | 400 |
| 7.7 | Übertragung von Dateien mit Servlets | 401 |
| 7.7.1 | Herunterladen von Dateien | 401 |
| 7.7.2 | Hochladen von Dateien | 404 |
| 7.8 | JSP (Java Server Pages) | 407 |
| 7.8.1 | Scripting-Elemente | 407 |
| 7.8.2 | Direktiven | 410 |
| 7.8.3 | Aktionen | 410 |
| 7.9 | MVC-Prinzip mit Servlets und JSPs | 414 |
| 7.10 | MVC-Prinzip mit AJAX und GWT | 421 |
| 7.11 | MVC-Prinzip mit WebSockets | 429 |
| 7.12 | Zusammenfassung | 432 |
| | Literatur | 435 |
| | Index | 437 |