High-Performance
Java
Platform Computing

THOMAS W. CHRISTOPHER • GEORGE K. THIRUVATHUKAL

Prentice Hall PTR, Upper Saddle River, NJ 07458
www.phptr.com

Sun Microsystems Press
A Prentice Hall Title
Preface, xv
Acknowledgments, xxi

Chapter 1

Foundations, 1
   The von Neumann Machine, 2
   Flynn’s Taxonomy, 2
   Control-Memory Taxonomy, 4
   Speedup and Efficiency, 5
   Amdahl’s Law, 7
   Scalability, 8
   Problems of Parallelism, 10
       Grain Size, 10
       Starvation, 11
       Deadlock, 11
Chapter 2

Threads, 19

Why is Multithreading Useful? 20

Overview of the Java Threads Classes, 21

Thread: The Class Responsible for Thread Creation and Execution, 21

Runnable: Making Threads Without Extending the Thread Class, 23

Hello World: A First Example of Multithreading, 25

Implementing Interfaces and Extending Classes: What's the Deal? 27

Concepts, 28

A Little Bit of History, 29

Some Definitions, 30

Multiprogramming vs. Multitasking, 31

Concurrency vs. Parallelism vs. Distribution, 32

Threads vs. Processes, 33

Kernel-level vs. User-level Threads, 36

Quick Tour of Java Threads Functions, 37

Construction, 37

Daemon Threads, 38

Thread Control, 38

Very Useful Static Methods, 39

Thread Synchronization, 39

Cooperative Multitasking, 40

Prioritized Scheduling, 40
Chapter 3

Race Conditions and Mutual Exclusion, 43

Know Your Enemy! 44

Race Conditions, 44

Egoist: A First Study in Race Conditions, 44

A Word on Priorities, 46

What is a Race Condition? 47

Race0 Class, 48

Critical Sections and Object Locking, 52

Race1 Class—Fixing Race0 with Synchronization, 53

Conditions, 55

Motivating the Need for Conditions, 55

Key Object Methods Needed to Work with Conditions in Java, 57

File Copying: A Producer–Consumer Example, 58

Locks–Binary Semaphores: An Example of Using Conditions, 68

Race2: Reworked Race1 Using Locks, 71

Classic Synchronization Mechanisms, 73

Counting Semaphore, 73

Barrier, 75

Futures, 76

Deadlock, 78

What is Deadlock? 78

How to Know When Deadlock Has Hit You? 78

Four Conditions of Deadlock, 78

A Classic Example: Dining Philosophers, 79

Chapter Wrap-up, 85

Exercises, 87
Chapter 4

Monitors, 89

Real Monitors and Java Monitors, 89

Class Monitor in the Thread Package, 92

Monitor's Methods, 94

Interface MonitorCondition's Methods, 95

Interface MonitorLock, 96

Examples using Monitor Objects, 97

SimpleFuture, 97

SharedTableOfQueues, 98

Implementation of Monitor, Condition, and MonitorLock, 101

Monitor Entry and Exit, 102

Monitor.Condition, 105

Monitor.Lock, 109

The Multiple Reader-Writer Monitors, 110

Policies, 111

Entering and Leaving the Critical Sections, 112

The Single-Reader-Writer Monitor, 112

The Readers-Preferred Monitor, 113

The Writers-Preferred Monitor, 115

The Alternating Readers-Writers Monitor, 117

The Take-a-Number Monitor, 119

Chapter Wrap-up, 120

Exercises, 121

Chapter 5

Parallel Execution of Subroutines in Shared Memory, 123

Creating and Joining, 124
Example: Trapezoidal Numeric Integration, 125

**RunQueue**, 129

- **RunQueue Methods**, 129
- **RunQueue Implementation**, 133

Recursive Shell Sort: **RunQueues and SimpleFutures**, 138

**Accumulator**, 141

- **Accumulator Operations**, 141
- **Patterns of Use of Accumulators**, 142

Using Accumulators, 144

- **Numeric Integration**, 144

**TerminationGroup**, 147

Combinatorial Search, 149

- **The 0–1 Knapsack Problem**, 149
- **Parallel Depth-first Search for the Knapsack Problem**, 152

**Knapsack2**, 154

**PriorityRunQueue**, 159

Branch-and-Bound with Priority Run Queues, 161

- **Branch and Bound for 0–1 Knapsack**, 161
- **A Purer Branch-and-Bound 0–1 Knapsack**, 167

Chapter Wrap-up, 167

Exercises, 170

---

**Chapter 6**

---

**Parallelizing Loops**, 171

Chore Graphs, 172

- **Gathering Chores into Threads**, 174

Example: Warshall’s Algorithm, 174

- **Static Scheduling**, 177
- **Dynamic Scheduling**, 177

Example: Longest Common Subsequence, 184
Chapter 7

Chores, 197

The RunDelayed Interface, 199
Futures, 199
FutureFactory, 201
Chore-programming Techniques, 201
  Job Jars, 202
  Chore Graphs, 202
  Macro Dataflow, 202
  Flow of Control, 206
Macro Dataflow Examples: Vector Addition, 209
  Continuations to Separate Operand Fetch from Operation, 210
  Static Dataflow Style and Storing Operands in Instructions, 213
  Fetching in Continuations, 217
Chore examples: Warshall’s Algorithm, 218
  WarshallDF1 and Warshall’s Algorithm in Dataflow, 219
  WarshallCl through WarshallC3: Chores, But Not Dataflow, 223
Chapter Wrap-up, 236
Exercises, 237

Chapter 8

Thread and Chore Synchronization, 239
  TerminationGroup, 239
  Barrier, 241
Chapter 9

Shared Tables of Queues, 257

Shared Tables of Queues, 258

Methods, 258

Implementing Synchronizations Using a Shared Table of Queues, 260

Indexed Keys, 262

Implementing More Synchronizations and Shared Structures, 265

Reactive Objects, 268

Communicating through a Shared Table of Queues, 271

Future Queues, 277

Methods, 278

Implementation of FutureQueue, 280

Example of FutureQueue: The Queued Readers–Writers Monitor, 285

Future Tables, 287

Chapter Wrap-up, 287

Exercises, 288

Chapter 10

Streams Essentials, 289

The Streams Abstraction, 290

Building Your Own Stream, 291
Memo: A Remote Interface to **SharedTableOfQueues**, 372

*Design, 372*
*MemoServer, 376*
*MemoClient, 377*

Vector Inner Product, 379
Trapezoidal Integration, 386
Warshall’s Algorithm, 391
Chapter Wrap-up, 398
Exercises, 398
Index, 401