# Contents

*Preface* ..... xiv

*Chapter 1 The Architecture of Threads* ..... 1

*The Problems with Threads* ..... 1
All Nontrivial Java Programs Are Multithreaded ..... 2
Java's Thread Support Is Not Platform Independent ..... 4
*Threads and Processes* ..... 5
*Thread Safety and Synchronization* ..... 6
Synchronization Is Expensive ..... 7
Avoiding Synchronization ..... 11
*Concurrency, or How Can You Be Two Places at Once (When You're Really Nowhere at All)* ..... 16
*Get Your Priorities Straight* ..... 17
*Cooperate!* ..... 19
The Cooperative Multithreading Model ..... 19
The Preemptive Multithreading Model ..... 19
Mapping Kernel Threads to User Processes ..... 20
*Wrapping Up* ..... 22

*Chapter 2 The Perils of Multithreaded Programming* ..... 25

*Monitors and Exclusion Semaphores (Mutex)* ..... 25
*Race Conditions and Spin Locks* ..... 27
The `Spin_lock` Class ..... 32
*Threads Are Not Objects* ..... 36
*Deadlock* ..... 38
*Get out the Magnifying Glass* ..... 39
*Nested-Monitor Lockout* ..... 42
Synchronization Wrappers ..... 43
*Time Out!* ..... 44
A Digression on Style ..... 47
*Why Is **suspend()** Deprecated?* ..... 49
*Deadlock on a Blocking I/O Operation* ..... 50
*Stopping Threads* ..... 53

*Starvation and Synchronizing* ***run()*** ..... 54
*The* ***volatile*** *Keyword* ..... 56
*Exceptions and Threads* ..... 58
*Conclusion* ..... 60

*Chapter 3 The Mutex and Lock Management* ..... 61

*When* **synchronized** *Isn't Good Enough* ..... 61
*Handling Granularity with* ***synchronized*** ..... 63
*Roll Your Own Semaphores: The* ***Semaphore*** *Interface* ..... 64
*Managing Semaphores and Deadlock-Resistant Locking* ..... 67
A Digression: Booch Utilities and Strategy ..... 72
*Implementing a Manageable Mutex Class* ..... 74

*Chapter 4 Condition Variables and Counting Semaphores* ..... 81

*Condition Variables* ..... 81
Waiting for the Electrician (or Somebody Like Him): Condition Variables vs. `wait()` ..... 81
Send in the Cavalry: Using a Condition Variable ..... 84
Implementing a Condition Variable ..... 91
*Condition Sets: Waiting for Multiple Conditions* ..... 99
*Counting Semaphores for Managing Resource Pools* ..... 106
*Wrapping Up* ..... 114

*Chapter 5 Timers, Alarms, and Swing Thread Safety* ..... 115

*Why a Timer?* ..... 115
*Swingin' Threads: Swing Isn't Thread Safe* ..... 116
The `invokeLater()` and `invokeAndWait()` Methods ..... 118
*Using the Swing* ***Timer*** ..... 119
So, How Does It Work? ..... 121
Why Use a Swing Timer (or Not) ..... 122
*Roll Your Own Timer: Implementing the* ***Alarm*** *Class* ..... 122
The Static Structure of an `Alarm` ..... 125
Dissecting a Notification ..... 127
Restarting an `Alarm` (Stopping a Thread) ..... 130
Suspending the Clock ..... 131

Notifier Problems ..... 132
Unit Tests ..... 144
*Summing Up* ..... 144

Chapter 6 Observers and Multicasters ..... 145

*Implementing Observer in a Multithreaded World* ..... 145
*Observer-side Problems: Inner-class Synchronization* ..... 147
*Notifier-side Problems:*
*Notifications in a Multithreaded World* ..... 149
*Mysteries of the **AWTEventMulticaster*** ..... 157
Immutable Objects and Blank Finals ..... 158
Using the Multicaster ..... 160
*Building a Multicaster* ..... 161

Chapter 7 Singletons, Critical Sections, and Reader/Writer Locks ..... 169

*Critical Sections, Singletons, and the "Class Object"* ..... 169
Static Members ..... 170
*Singletons* ..... 174
Critical Sections, Doubled-checked Locking,
and Cache-related Problems in Multiple-CPU Machines ..... 176
The `Std` Class: An Example of Singleton ..... 178
Closing Singletons ..... 186
*Reader/Writer Locks* ..... 187
*It's a Wrap* ..... 198

Chapter 8 Threads in an Object-Oriented World ..... 201

*Modeling Threads in Object-Oriented Systems* ..... 201
Synchronous vs. Asynchronous Messages ..... 202
Implementing Asynchronous Messages
Using Thread-per-Method ..... 203
An Exceptional Problem ..... 208
*Thread Pools and Blocking Queues* ..... 209
Blocking Queues ..... 210
Pooling Threads ..... 220
Passing Arguments to the Operation ..... 222

Using Introspection for Runnable Objects
that Pass Arguments ........ 223
Implementing the `Thread_pool` ........ 228
Putting the Pool to Work ........ 237
*Sockets and Thread Pools* ........ 238
*Conclusion* ........ 249

**Chapter 9 Object-Oriented Threading Architectures** ........ 251

*Reactors and Active Objects* ........ 251
*Synchronous Dispatching and Round-Robin Scheduling:
Reactors and Proactors* ........ 253
*Asynchronous Dispatching: Active Objects* ........ 262
A General Solution ........ 264
Detangling Console Output ........ 268
*That's It* ........ 274

**Chapter 10 If I Were King: Fixing Java's
Threading Problems** ........ 275

*The Task* ........ 275
*Improvements to **synchronized*** ........ 279
*Improvements to **wait()** and **notify()*** ........ 280
*Fixing the **Thread** Class* ........ 281
*Inter-Thread Coordination* ........ 282
*Internal Support for Reader/Writer Locks* ........ 282
*Access to Partially Constructed Objects
Should Be Illegal* ........ 283
*Volatile Should Always Work as Expected* ........ 284
*Access Issues* ........ 285
Immutability ........ 286
Instance-Level Access of Class-Level Fields ........ 287
*Singleton Destruction* ........ 288
*Abrupt Shut Down of Daemon Threads* ........ 289
*Bring Back the **stop(), suspend(),** and **resume()** Methods* ........ 289
*Blocking I/O Should Work Correctly* ........ 290
*The **ThreadGroup** Class* ........ 290
*Wrapping Up* ........ 290

***Index*** ........ 293