Java ReentrantReadWriteLock: Usage Considerations

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Institute for Software Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA
Learning Objectives in this Part of the Lesson

• Understand the structure & functionality of the Java ReentrantReadWriteLock class
• Know the key methods in Java ReentrantReadWriteLock
• Recognize how to apply Java ReentrantReadWriteLock in practice
• Appreciate Java ReentrantReadWriteLock usage considerations
ReentrantReadWriteLock
Usage Considerations
ReentrantReadWriteLock Usage Considerations

- ReentrantReadWriteLock enables higher levels of concurrency when accessing shared “read-only” data compared with a ReentrantLock

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/locks/ReadWriteLock.html
ReentrantReadWriteLock Usage Considerations

- ReentrantReadWriteLock enables higher levels of concurrency when accessing shared “read-only” data compared with a ReentrantLock.

- May improve performance if data are read from much more often than written to on multi-core systems.
• However, ReentrantReadWriteLock has several limitations

See javaspecialists.eu/talks/jfokus13/PhaserAndStampedLock.pdf
ReentrantReadWriteLock Usage Considerations

- However, ReentrantReadWriteLock has several limitations
  - Both read & write locks are “pessimistic” & thus assume contention will always occur
ReentrantReadWriteLock Usage Considerations

- However, ReentrantReadWriteLock has several limitations
- Both read & write locks are “pessimistic” & thus assume contention will always occur
- In contrast, StampedLock has an “optimistic” read mode & generally performs better

See upcoming lesson on “Java StampedLock”
ReentrantReadWriteLock Usage Considerations

- However, ReentrantReadWriteLock has several limitations
  - Both read & write locks are “pessimistic”
  - Can starve readers or writers, depending on their priority

See en.wikipedia.org/wiki/Readers-writer_lock
ReentrantReadWriteLock Usage Considerations

- However, ReentrantReadWriteLock has several limitations
  - Both read & write locks are “pessimistic”
  - Can starve readers or writers, depending on their priority
  - Java 5 (readers priority) & 6+ (writers priority) semantics differ

See [www.javaspecialists.eu/archive/Issue165.html](http://www.javaspecialists.eu/archive/Issue165.html)
ReentrantReadWriteLock Usage Considerations

- However, ReentrantReadWriteLock has several limitations
  - Both read & write locks are “pessimistic”
  - Can starve readers or writers, depending on their priority
  - Can be tedious & error-prone to program
• However, ReentrantReadWriteLock has several limitations
  • Both read & write locks are “pessimistic”
  • Can starve readers or writers, depending on their priority
  • Can be tedious & error-prone to program

See earlier lessons on “Java ReentrantLock” & “Java Semaphore”
Profiling is essential to see if a ReentrantReadWriteLock is suited for a particular use-case.
ReentrantReadWriteLock Usage Considerations

- Profiling is essential to see if a ReentrantReadWriteLock is suited for a particular use-case.
- ReentrantReadWriteLock’s overhead is nearly always greater than any benefits it provides.

End of Java Reentrant ReadWriteLock: Usage Considerations