Java StampedLock:
Key Methods

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 StampedLock class

• Know the key methods in Java StampedLock

```
<<Java Class>>

G StampedLock

- StampedLock()
- writeLock():long
- tryWriteLock():long
- tryWriteLock(long, TimeUnit):long
- writeLockInterruptibly():long
- readLock():long
- tryReadLock():long
- tryReadLock(long, TimeUnit):long
- readLockInterruptibly():long
- tryOptimisticRead():long
- validate(long):boolean
- unlockWrite(long):void
- unlockRead(long):void
- unlock(long):void
- tryConvertToWriteLock(long):long
- tryConvertToReadLock(long):long
- tryConvertToOptimisticRead(long):long
- tryUnlockWrite():boolean
- tryUnlockRead():boolean
- isWriteLocked():boolean
- isReadLocked():boolean
- getReadLockCount():int
- toString()
- asReadLock():Lock
- asWriteLock():Lock
- asReadWriteLock():ReadWriteLock
```
Key Methods in Stamped Lock: Writing Mode
Key Methods in StampedLock: Writing Mode

- Writing mode methods, which acquire the lock exclusively

```java
public class StampedLock implements java.io.Serializable {
    ...
    public long writeLock() { ... }
    public long tryWriteLock() { ... }
    public long tryWriteLock(long time, TimeUnit unit) {...}
    ...
}
```

Only one thread at a time can acquire a lock exclusively
Key Methods in StampedLock: Writing Mode

- Writing mode methods, which acquire the lock exclusively

```java
public class StampedLock implements java.io.Serializable {
    ...
    public long writeLock() { ... }
    public long tryWriteLock() { ... }
    public long tryWriteLock(long time, TimeUnit unit) { ... }
    ...
}
```

These methods are "pessimistic", i.e., they assume contention can occur.
Key Methods in StampedLock: Writing Mode

- Writing mode methods, which acquire the lock exclusively

```
public class StampedLock
    implements java.io.Serializable {
    ...
    public long writeLock() { ... }
    public long tryWriteLock() { ... }
    public long tryWriteLock(
        long time,
        TimeUnit unit) { ... }
    ...

    All methods return a "stamp" value, which is a long that contains a version & a mode

See dzone.com/articles/a-look-at-stampedlock
```
Writing mode methods, which acquire the lock exclusively

- Acquires lock exclusively, blocking until available
- This method can’t be interrupted

There’s also a `writeLockInterruptibly()` method that can be interrupted

```java
class StampedLock implements java.io.Serializable {
    ...
    public long writeLock() {
        ...
    }
    public long tryWriteLock() {
        ...
    }
    public long tryWriteLock(long time, TimeUnit unit) {
        ...
    }
    ...
}
```
Key Methods in StampedLock: Writing Mode

- Writing mode methods, which acquire the lock exclusively
  - Acquires lock exclusively, blocking until available
  - Acquires lock exclusively if it’s immediately available
  - Otherwise, it returns 0

```java
public class StampedLock implements java.io.Serializable {
    ...
    public long writeLock() { ... }
    public long tryWriteLock() { ... }
    public long tryWriteLock(long time,
                              TimeUnit unit) { ... }
    ...
```
Key Methods in StampedLock: Writing Mode

- Writing mode methods, which acquire the lock exclusively
  - Acquires lock exclusively, blocking until available
  - Acquires lock exclusively if it’s immediately available
  - Acquires lock exclusively if available within given time
  - Otherwise, it returns 0

```
public class StampedLock implements java.io.Serializable {
    ...
    public long writeLock() { ... }
    public long tryWriteLock() { ... }
    public long tryWriteLock(long time, TimeUnit unit) { ... }
    ...
}
```
Key Methods in StampedLock: Writing Mode

- Writing mode methods, which acquire the lock exclusively
  - Acquires lock exclusively, blocking until available
  - Acquires lock exclusively if it’s immediately available
  - Acquires lock exclusively if available within given time
  - Otherwise, it returns 0
  - This method throws InterruptedException if it’s interrupted

```java
public class StampedLock implements java.io.Serializable {
    ...  
    public long writeLock() { ... }  
    public long tryWriteLock() { ... }
    public long tryWriteLock(long time, TimeUnit unit) {...}  
    ...
}
```
Key Methods in Stamped Lock: Reading Mode
Key Methods in StampedLock: Reading Mode

- Reading mode methods, which acquire the lock non-exclusively

```java
public class StampedLock implements java.io.Serializable {
    ...
    public long readLock() { ... }
    
    public long tryReadLock() { ... }
    
    public long tryReadLock(
            long time,
            TimeUnit unit) {...}
    ...
}
```

Many threads at a time can acquire a lock non-exclusively
Key Methods in StampedLock: Reading Mode

- Reading mode methods, which acquire the lock non-exclusively

```java
public class StampedLock implements java.io.Serializable {
    ...
    public long readLock() { ... }
    public long tryReadLock() { ... }
    public long tryReadLock(long time, TimeUnit unit) { ... }
    ...
}
```

These methods are “pessimistic”, i.e., they assume contention can occur.
Key Methods in StampedLock: Reading Mode

- Reading mode methods, which acquire the lock non-exclusively

  Again, all methods return a "stamp", which is a long that contains a version & a mode

  ```java
  public class StampedLock
  implements java.io.Serializable {
      ...
  
      public long readLock() { ... }
      
      public long tryReadLock() { ... }
      
      public long tryReadLock
      (long time,
       TimeUnit unit) {...}
      
      ...
  }
  ```
Key Methods in StampedLock: Reading Mode

- Reading mode methods, which acquire the lock non-exclusively
- Acquires lock non-exclusively, blocking until available
- This method can’t be interrupted

```java
public class StampedLock implements java.io.Serializable {
    ...
    public long readLock() { ... }
    public long tryReadLock() { ... }
    public long tryReadLock(long time, TimeUnit unit) {...}
    ...
```

There’s also a readLockInterruptibly() method that *can* be interrupted
Key Methods in StampedLock: Reading Mode

- Reading mode methods, which acquire the lock non-exclusively
  - Acquires lock non-exclusively, blocking until available
  - Acquires lock non-exclusively if immediately available
  - Otherwise, it returns 0

```java
public class StampedLock implements java.io.Serializable {
    ...
    public long readLock() { ... }
    public long tryReadLock() { ... }
    public long tryReadLock(
        long time,
        TimeUnit unit) {...}
    ...
```

Key Methods in StampedLock: Reading Mode

- Reading mode methods, which acquire the lock non-exclusively
  - Acquires lock non-exclusively, blocking until available
  - Acquires lock non-exclusively if immediately available
  - Acquires lock non-exclusively if it is available within given time
  - Otherwise, it returns 0

```java
public class StampedLock implements java.io.Serializable {
    ...
    public long readLock() { ... }
    public long tryReadLock() { ... }
    public long tryReadLock(
        long time,
        TimeUnit unit) {...}
    ...
```
Key Methods in StampedLock: Reading Mode

• Reading mode methods, which acquire the lock non-exclusively

  • Acquires lock non-exclusively, blocking until available
  • Acquires lock non-exclusively if immediately available
  • Acquires lock non-exclusively if it is available within given time
  • Otherwise, it returns 0
  • This method throws InterruptedException if it’s interrupted

```java
public class StampedLock implements java.io.Serializable {
    ...
    public long readLock() { ... }
    public long tryReadLock() { ... }
    public long tryReadLock(long time, TimeUnit unit) {...}
    ...
}
```
Key Methods in Stamped Lock: Optimistic Mode
Key Methods in StampedLock: Optimistic Mode

- Optimistic reading mode methods, which acquire the lock non-exclusively

public class StampedLock
    implements java.io.Serializable {
    ...
    public long tryOptimisticRead() {
        ...
    }

    public boolean validate (long stamp) {
        ...
    }

    ...

Many threads at a time can acquire a lock non-exclusively
Key Methods in StampedLock: Optimistic Mode

- Optimistic reading mode methods, which acquire the lock non-exclusively

```java
public class StampedLock
    implements java.io.Serializable {
    ...
    public long tryOptimisticRead() {
        ...
    }
    public boolean validate
        (long stamp) {
        ...
    }
    ...
}
```

These methods are “optimistic”, i.e., they assume contention may not occur.
Key Methods in StampedLock: Optimistic Mode

- Optimistic reading mode methods, which acquire the lock non-exclusively
- Returns an "observation stamp" for later validation or 0 if the lock is currently held exclusively

```java
public class StampedLock
    implements java.io.Serializable {
    ...
    public long tryOptimisticRead() {
        ...
    }
    public boolean validate (long stamp) {
        ...
    }
    ...
```
Key Methods in StampedLock: Optimistic Mode

- Optimistic reading mode methods, which acquire the lock non-exclusively
- Returns an “observation stamp” for later validation or 0 if the lock is currently held exclusively
- Code using this mode reads the value of fields & holds them in local variables for use after they are “validated”

```java
public class StampedLock implements java.io.Serializable {
    ...

    public long tryOptimisticRead() {
        ...
    }

    public boolean validate(
        long stamp) { ...
    }

    ...
```

See upcoming part of this lesson on “Java StampedLock: Example Application”
Key Methods in StampedLock: Optimistic Mode

- Optimistic reading mode methods, which acquire the lock non-exclusively
- Returns an “observation stamp” for later validation or 0 if the lock is currently held exclusively
- Code using this mode reads the value of fields & holds them in local variables for use after they are “validated”
- tryOptimisticRead() internally does a volatile read on a field inside of StampedLock to ensure “fresh” values of fields are observed

```java
public class StampedLock implements java.io.Serializable {
    ... 
    public long tryOptimisticRead() {
        ... }
    public boolean validate (long stamp) { ... }
    ... 
}
```

See [www.javaspecialists.eu/archive/Issue242.html](http://www.javaspecialists.eu/archive/Issue242.html)
Key Methods in StampedLock: Optimistic Mode

- Optimistic reading mode methods, which acquire the lock non-exclusively
  - Returns an “observation stamp” for later validation or 0 if the lock is currently held exclusively
- Returns true if lock hasn’t been acquired exclusively since stamp was issued, else false

```java
public class StampedLock implements java.io.Serializable {
    ...
    public long tryOptimisticRead() {
        ...
    }
    public boolean validate (long stamp) {
        ...
    }
    ...
```
Key Methods in StampedLock: Optimistic Mode

- Optimistic reading mode methods, which acquire the lock non-exclusively
  - Returns an “observation stamp” for later validation or 0 if the lock is currently held exclusively
  - Returns true if lock hasn’t been acquired exclusively since stamp was issued, else false
  - If validate() succeeds (i.e., returns true) synchronization overhead is very low & there’s no need to unlock the “lock”

```java
public class StampedLock implements java.io.Serializable {
    ...
    public long tryOptimisticRead() {
        ...
    }

    public boolean validate(long stamp) {
        ...
    }
    ...
```
Key Methods in Stamped Lock: Conversions
Key Methods in StampedLock: Conversions

- Conditionally convert a stamp across lock modes

```java
public class StampedLock implements java.io.Serializable {
    ...
    
    public long tryToConvertToWriteLock (long stamp) { ... }

    public long tryToConvertToReadLock (long stamp) { ... }

    public long tryToConvertToOptimisticRead (long stamp) { ... }
    ...
}
```

These calls perform work *atomically* (despite lack of documentation ;-))
Key Methods in StampedLock: Conversions

- Conditionally convert a stamp to a write lock

```java
class StampedLock implements java.io.Serializable {
    ...
    public long tryToConvertToWriteLock (long stamp) { ... }
    ...
    public long tryToConvertToReadLock (long stamp) { ... }
    ...
    public long tryToConvertToOptimisticRead (long stamp) { ... }
    ...
}
```
Key Methods in StampedLock: Conversions

- Conditionally convert a stamp to a write lock
- If lock state matches stamp, performs one following action

```java
class StampedLock implements java.io.Serializable {
    ...
    public long tryToConvertToWriteLock(long stamp) { ... }

    public long tryToConvertToReadLock(long stamp) { ... }

    public long tryToConvertToOptimisticRead(long stamp) { ... }
    ...
}
```
Key Methods in StampedLock: Conversions

- Conditionally convert a stamp to a write lock
  - If lock state matches stamp, performs one following action
  - If stamp represents holding a write lock, return it

```java
public class StampedLock implements java.io.Serializable {
    ...
    public long tryToConvertToWriteLock (long stamp) { ... }

    public long tryToConvertToReadLock (long stamp) { ... }

    public long tryToConvertToOptimisticRead (long stamp) { ... }

    ...
}
```
Key Methods in StampedLock: Conversions

- Conditionally convert a stamp to a write lock
  - If lock state matches stamp, performs one following action
    - If stamp represents holding a write lock, return it
    - If stamp represents holding a read lock—and a write lock is available—atomically release read lock & return write stamp

```java
public class StampedLock
    implements java.io.Serializable {

    public long tryToConvertToWriteLock(
        long stamp) {
        ... }

    public long tryToConvertToReadLock(
        long stamp) {
        ... }

    public long tryToConvertToOptimisticRead(
        long stamp) {
        ... }

    ...
```

This represents a “lock upgrade”
Key Methods in StampedLock: Conversions

- Conditionally convert a stamp to a write lock
  - If lock state matches stamp, performs one following action
    - If stamp represents holding a write lock, return it
    - If stamp represents holding a read lock—and a write lock is available—atomically release read lock & return write stamp
    - If stamp represents a read that’s optimistic, return a write stamp if immediately available

```java
public class StampedLock implements java.io.Serializable {
    ...
    public long tryToConvertToWriteLock (long stamp) { ... }
    public long tryToConvertToReadLock (long stamp) { ... }
    public long tryToConvertToOptimisticRead (long stamp) { ... }
    ...
```
• Conditionally convert a stamp to a write lock
  • If lock state matches stamp, performs one following action
    • If stamp represents holding a write lock, return it
    • If stamp represents holding a read lock—and a write lock is available—atomically release read lock & return write stamp
    • If stamp represents a read that’s optimistic, return a write stamp if immediately available
  • Else return zero

public class StampedLock
  implements java.io.Serializable {
    ...
    public long tryToConvertToWriteLock
      (long stamp) { ... }

    public long tryToConvertToReadLock
      (long stamp) { ... }

    public long tryToConvertToOptimisticRead
      (long stamp) { ... }
    ...
  }
Key Methods in StampedLock: Conversions

- Conditionally convert a stamp to a read lock

```java
class StampedLock
    implements java.io.Serializable {

    public long tryToConvertToWriteLock(long stamp) {
        ...
    }

    public long tryToConvertToReadLock(long stamp) {
        ...
    }

    public long tryToConvertToOptimisticRead(long stamp) {
        ...
    }

    ...
```
Key Methods in StampedLock: Conversions

- Conditionally convert a stamp to a read lock
- If lock state matches stamp, performs one following action

```java
public class StampedLock
    implements java.io.Serializable {
    ...
    public long tryToConvertToWriteLock
        (long stamp) { ... }

    public long tryToConvertToReadLock
        (long stamp) { ... }

    public long tryToConvertToOptimisticRead
        (long stamp) { ... }
    ...
```
Key Methods in StampedLock: Conversions

- Conditionally convert a stamp to a read lock
- If lock state matches stamp, performs one following action
- If stamp represents holding a write lock atomically release it & obtain read lock

```java
class StampedLock implements java.io.Serializable {
    ... public long tryToConvertToWriteLock (long stamp) { ... }
    public long tryToConvertToReadLock (long stamp) { ... }
    public long tryToConvertToOptimisticRead (long stamp) { ... }
    ...
}
```

This represents a "lock downgrade"
Key Methods in StampedLock: Conversions

• Conditionally convert a stamp to a read lock
  • If lock state matches stamp, performs one following action
    • If stamp represents holding a write lock atomically release it & obtain read lock
  • If stamp represents holding a read lock, return it

public class StampedLock implements java.io.Serializable {
  ...
  public long tryToConvertToWriteLock (long stamp) { ... }
  ...
  public long tryToConvertToReadLock (long stamp) { ... }
  ...
  public long tryToConvertToOptimisticRead (long stamp) { ... }
  ...

• Conditionally convert a stamp to a read lock
  • If lock state matches stamp, performs one following action
    • If stamp represents holding a write lock atomically release it & obtain read lock
    • If stamp represents holding a read lock, return it
    • If stamp represents holding an optimistic read, return read stamp only if available

public class StampedLock
    implements java.io.Serializable {
    ...
    public long
        tryToConvertToWriteLock
            (long stamp) { ... }

    public long
        tryToConvertToReadLock
            (long stamp) { ... }

    public long
        tryToConvertToOptimisticRead
            (long stamp) { ... }
    ...

Key Methods in StampedLock: Conversions

- Conditionally convert a stamp to a read lock
  - If lock state matches stamp, performs one following action
    - If stamp represents holding a write lock atomically release it & obtain read lock
    - If stamp represents holding a read lock, return it
    - If stamp represents holding an optimistic read, return read stamp only if available
  - Else return zero

```java
public class StampedLock
    implements java.io.Serializable {
    ...
    public long
        tryToConvertToWriteLock
            (long stamp) { ... }

    public long
        tryToConvertToReadLock
            (long stamp) { ... }

    public long
        tryToConvertToOptimisticRead
            (long stamp) { ... }
    ...
```
Key Methods in StampedLock: Conversions

• Conditionally convert a stamp to an optimistic read lock

```java
public class StampedLock
    implements java.io.Serializable {
    ...
    public long
        tryToConvertToWriteLock
            (long stamp) { ... }

    public long
        tryToConvertToReadLock
            (long stamp) { ... }

    public long
        tryToConvertToOptimisticRead
            (long stamp) { ... }
    ...
```
Key Methods in StampedLock: Conversions

• Conditionally convert a stamp to an optimistic read lock
• If lock state matches stamp, performs one following action

```java
public class StampedLock
    implements java.io.Serializable {
    ...
    public long tryToConvertToWriteLock
        (long stamp) { ... }

    public long tryToConvertToReadLock
        (long stamp) { ... }

    public long tryToConvertToOptimisticRead
        (long stamp) { ... }
    ...
```
Key Methods in StampedLock: Conversions

- Conditionally convert a stamp to an optimistic read lock
- If lock state matches stamp, performs one following action
- If stamp represents holding a lock release it & return an observation stamp

```
public class StampedLock
    implements java.io.Serializable {
    ...
    public long
        tryToConvertToWriteLock
            (long stamp) { ... }

    public long
        tryToConvertToReadLock
            (long stamp) { ... }

    public long
        tryToConvertToOptimisticRead
            (long stamp) { ... }
    ...
```

This represents a “lock downgrade”
• Conditionally convert a stamp to an optimistic read lock
  • If lock state matches stamp, performs one following action
    • If stamp represents holding a lock release it & return an observation stamp
  • If stamp represents holding an optimistic read, return it if it’s validated

public class StampedLock
  implements java.io.Serializable {

  public long tryToConvertToWriteLock (long stamp) { ... }

  public long tryToConvertToReadLock (long stamp) { ... }

  public long tryToConvertToOptimisticRead (long stamp) { ... }

  ...

  ...
• Conditionally convert a stamp to an optimistic read lock
  • If lock state matches stamp, performs one following action
    • If stamp represents holding a lock release it & return an observation stamp
    • If stamp represents holding an optimistic read, return it if it’s validated
  • Else return zero

public class StampedLock
    implements java.io.Serializable {
    ...
    public long
        tryToConvertToWriteLock
            (long stamp) { ... }

    public long
        tryToConvertToReadLock
            (long stamp) { ... }

    public long
        tryToConvertToOptimisticRead
            (long stamp) { ... }
    ...

Key Methods in Stamped Lock: Unlocking
Key Methods in StampedLock: Unlocking

- There are several ways to unlock a StampedLock

```java
public class StampedLock
    implements java.io.Serializable {
    ...
    public void unlockWrite
        (long stamp) { ... }

    public void unlockRead
        (long stamp) { ... }

    public void unlock
        (long stamp) { ... }

    ...
```
Key Methods in StampedLock: Unlocking

- There are several ways to unlock a StampedLock
- Releases exclusive lock if the state matches given stamp

```java
public class StampedLock implements java.io.Serializable {
    ...
    public void unlockWrite (long stamp) { ... }
    ...
    public void unlockRead (long stamp) { ... }
    ...
    public void unlock (long stamp) { ... }
    ...
}
```
Key Methods in StampedLock: Unlocking

- There are several ways to unlock a StampedLock
  - Releases exclusive lock if the state matches given stamp
  - Releases non-exclusive lock if the state matches given stamp

```java
public class StampedLock implements java.io.Serializable {
    ...
    public void unlockWrite (long stamp) { ... }

    public void unlockRead (long stamp) { ... }

    public void unlock (long stamp) { ... }

    ...
}```
Key Methods in StampedLock: Unlocking

- There are several ways to unlock a StampedLock
  - Releases exclusive lock if the state matches given stamp
  - Releases non-exclusive lock if the state matches given stamp
  - Releases lock if the lock state matches given stamp

```java
class StampedLock implements java.io.Serializable {

  public void unlockWrite(long stamp) {
    ...}

  public void unlockRead(long stamp) {
    ...}

  public void unlock(long stamp) {
    ...}

  ...
}
```

`unlock()` is slightly less efficient than `unlockWrite()` & `unlockRead()`
End of Java StampedLock: Key Methods