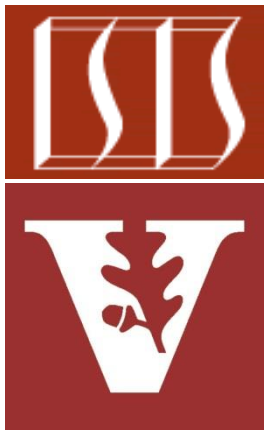
Java CountDownLatch: Key Methods

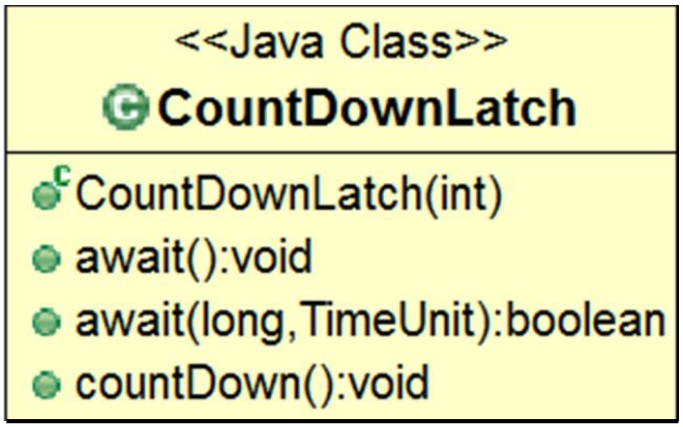


Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Institute for Software
Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Lesson

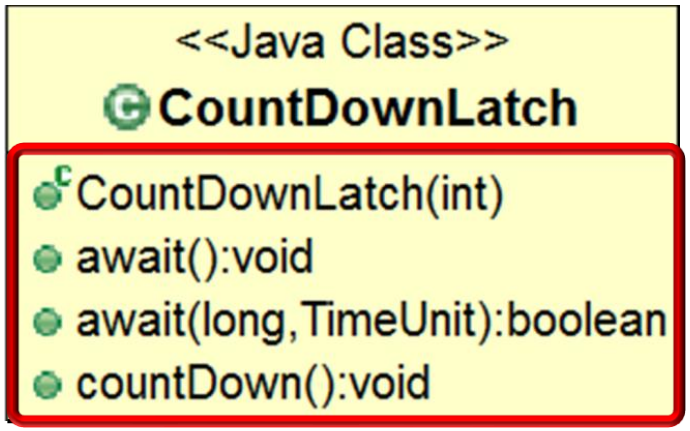
- Understand the structure & functionality of Java CountDownLatch
- Recognize the key methods in Java CountDownLatch



Key Methods in Java CountDownLatch

Key Methods in Java CountDownLatch

- CountDownLatch has a very simple API
  - i.e., only a handful of methods are commonly used



Key Methods in Java CountDownLatch

- CountDownLatch's constructor initializes the count

```
public class CountDownLatch {
  ...
  public CountDownLatch
                        (int count) {
    ...
    this.sync = new Sync(count);
  }
  ...
```

Key Methods in Java CountDownLatch

- CountDownLatch's constructor initializes the count
  - This count is simply used to create an instance of the AbstractQueuedSynchronizer

```
public class CountDownLatch {
   ...
   public CountDownLatch
                        (int count) {
      ...
      this.sync = new Sync(count);
   }
   ...
```

Key Methods in Java CountDownLatch

- CountDownLatch's constructor initializes the count
  - This count is simply used to create an instance of the AbstractQueuedSynchronizer
  - The count cannot be reset without recreating a new instance of CountDownLatch

```
public class CountDownLatch {
  ...
  public CountDownLatch
                        (int count) {
    ...
    this.sync = new Sync(count);
  }
  ...
```

See upcoming lessons on "*Java CyclicBarrier*" & "*Java Phaser*" for alternatives

Key Methods in Java CountDownLatch

- Key methods count down & wait for the count to reach 0

```
public class CountDownLatch {
  ...
  public void countDown() {
    sync.releaseShared(1);
  }

  public void await() ... {
    sync.acquireShared
         Interruptibly(1);
  }

  public boolean await
    (long timeout,
     TimeUnit unit) ... {
    return sync.
      tryAcquireSharedNanos
      (1, unit.toNanos(timeout));
  }
  ...
```

Key Methods in Java CountDownLatch

- Key methods count down & wait for the count to reach 0

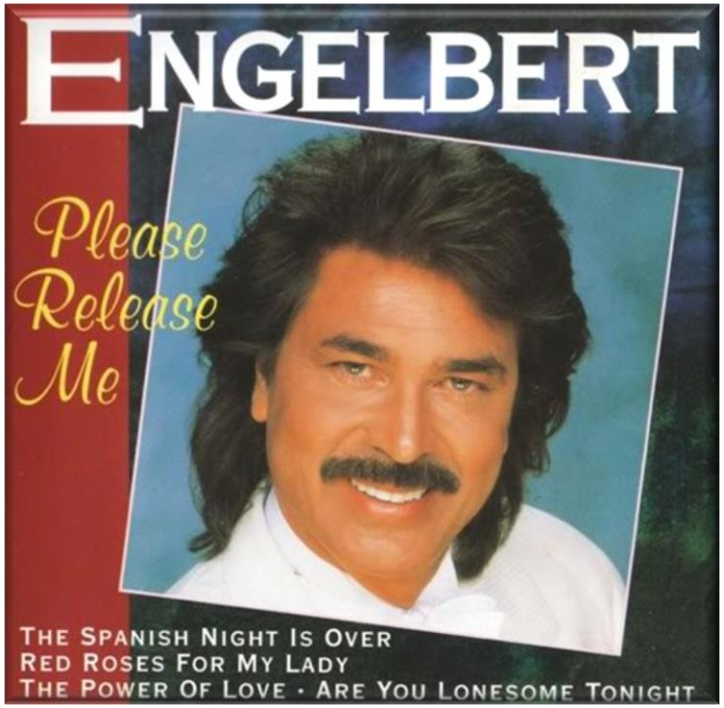
Methods forward to the underlying methods in the AbstractQueuedSynchronizer

```
public class CountDownLatch {
  ...
  public void countDown() {
    sync.releaseShared(1);
  }

  public void await() ... {
    sync.acquireShared
         Interruptibly(1);
  }

  public boolean await
    (long timeout,
     TimeUnit unit) ... {
    return sync.
      tryAcquireSharedNanos
      (1, unit.toNanos(timeout));
  }
  ...
```

See gee.cs.oswego.edu/dl/papers/aqs.pdf

Key Methods in Java CountDownLatch

- Key methods count down & wait for the count to reach 0
  - Decrements latch count by 1 & releases any threads blocked on await() when count reaches 0




```
public class CountDownLatch {
   ...
   public void countDown() {
      sync.releaseShared(1);
   }
```

Key Methods in Java CountDownLatch

- Key methods count down & wait for the count to reach 0
  - Decrements latch count by 1 & releases any threads blocked on await() when count reaches 0
    - Threads calling countDown() don't block for count to reach 0 before proceeding

```
public class CountDownLatch {
   ...
   public void countDown() {
     sync.releaseShared(1);
   }
```

Key Methods in Java CountDownLatch

- Key methods count down & wait for the count to reach 0
  - Decrements latch count by 1 & releases any threads blocked on await() when count reaches 0
  - Causes the calling thread to block until the latch's count reaches 0, at which point await() returns
    - *Unless* the thread is interrupted

```
public class CountDownLatch {
   ...
   public void await() ... {
      sync.acquire...(1);
   }
   ...
```

Key Methods in Java CountDownLatch

- Key methods count down & wait for the count to reach 0
  - Decrements latch count by 1 & releases any threads blocked on await() when count reaches 0
  - Causes the calling thread to block until the latch's count reaches 0, at which point await() returns
    - *Unless* the thread is interrupted
    - *Unless* waiting time elapses or the thread is interrupted

```
public class CountDownLatch {
  ...
  public void await() ... {
    sync.acquire...(1);
  }

  public boolean await
    (long timeout,
     TimeUnit unit) ... {
    return sync.
      tryAcquireSharedNanos
      (1, unit.toNanos(timeout));
  }
  ...
```

Key Methods in Java CountDownLatch

- Key methods count down & wait for the count to reach 0
  - Decrements latch count by 1 & releases any threads blocked on await() when count reaches 0
  - Causes the calling thread to block until the latch's count reaches 0, at which point await() returns


```
public class CountDownLatch {
  ...
  public void await() ... {
    sync.acquire...(1);
  }

  public boolean await
    (long timeout,
     TimeUnit unit) ... {
    return sync.
      tryAcquireSharedNanos
      (1, unit.toNanos(timeout));
  }
  ...
```

There is no "non-interruptible" version of await()

End of Java CountDownLatch: Key Methods