Java Semaphore:
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 concept of semaphores
- Be aware of the two types of semaphores
- Note a human known use of semaphores
- Recognize the structure & functionality of Java Semaphore
- Know the key methods defined by the Java Semaphore class
Overview of Key Java Semaphore Methods
Overview of Key Java Semaphore Methods

- Its key methods acquire & release the semaphore

```java
public class Semaphore
    implements ... {
    ...
    public void acquire() { ... }
    public void acquireUninterruptibly()
    { ... }
    public boolean tryAcquire
        (long timeout,
         TimeUnit unit)
    { ... }
    public void release() { ... }
    ...
```

See [docs.oracle.com/javase/8/docs/api/java/util/concurrent/Semaphore.html](docs.oracle.com/javase/8/docs/api/java/util/concurrent/Semaphore.html)
Overview of Key Java Semaphore Methods

- Its key methods acquire & release the semaphore

```java
public class Semaphore
{
    implements ...
{
    ...

    public void acquire() { ... }

    public void
    acquireUninterruptibly()
    { ... }

    public boolean tryAcquire
    (long timeout,
     TimeUnit unit)
    { ... }

    public void release() { ... }
    ...
}
```

These methods forward to their implementor methods, which are largely inherited from the AbstractQueuedSynchronizer framework

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/locks/AbstractQueuedSynchronizer.html
Overview of Key Java Semaphore Methods

• Its key methods acquire & release the semaphore
• acquire() atomically obtains a permit from the semaphore

```java
public class Semaphore
    implements ...
{
    ...
    public void acquire()
    {
        sync.
        acquireSharedInterruptibly(1);
    }
    ...
}
```
Overview of Key Java Semaphore Methods

- Its key methods acquire & release the semaphore
  - acquire() atomically obtains a permit from the semaphore
  - Can be interrupted

```java
public class Semaphore implements ... {
    ...
    public void acquire() {
        sync.
        acquireSharedInterruptibly(1);
    }
    ...
}
```

See [docs.oracle.com/javase/tutorial/essential/concurrency/interrupt.html](https://docs.oracle.com/javase/tutorial/essential/concurrency/interrupt.html)
Overview of Key Java Semaphore Methods

- Its key methods acquire & release the semaphore
  - `acquire()` atomically obtains a permit from the semaphore
  - `acquireUninterruptibly()` also obtains a permit from the semaphore
  - Cannot be interrupted

```java
public class Semaphore implements ... {
    ...
    public void acquireUninterruptibly() {
        sync.acquireShared(1)
    }
    ...
}
```
Overview of Key Java Semaphore Methods

- Its key methods acquire & release the semaphore
- acquire() atomically obtains a permit from the semaphore
- acquireUninterruptibly() also obtains a permit from the semaphore
- tryAcquire() obtains a permit if it’s available at invocation time

```java
public class Semaphore implements ... {
    ...
    public boolean tryAcquire() {
        ...
        sync.
        nonfairTryAcquireShared(1) >= 0;
    }
    ...
```

---

9
Overview of Key Java Semaphore Methods

• Its key methods acquire & release the semaphore
  • acquire() atomically obtains a permit from the semaphore
  • acquireUninterruptibly() also obtains a permit from the semaphore
  • tryAcquire() obtains a permit if it’s available at invocation time

```java
public class Semaphore implements ... {
    ...
    public boolean tryAcquire() {
        ...
        sync.
        nonfairTryAcquireShared(1) >= 0;
    }
    ...
}
```

Untimed tryAcquire() methods will “barge”, i.e., they don’t honor the fairness setting & take any permits available
Overview of Key Java Semaphore Methods

- Its key methods acquire & release the semaphore
  - acquire() atomically obtains a permit from the semaphore
  - acquireUninterruptibly() also obtains a permit from the semaphore
  - tryAcquire() obtains a permit if it’s available at invocation time
  - release() atomically increments the permit count by 1

```java
public class Semaphore
    implements ...
{
    ...
    public void release()
    {
        sync.releaseShared(1);
    }
    ...
}
```

Recall it’s valid for the permit count to exceed the initial permit count!!
Overview of Key Java Semaphore Methods

- Its key methods acquire & release the semaphore
  - acquire() atomically obtains a permit from the semaphore
  - acquireUninterruptibly() also obtains a permit from the semaphore
  - tryAcquire() obtains a permit if it’s available at invocation time
  - release() atomically increments the permit count by 1
    - If the permit count is now > - a thread waiting to acquire the semaphore can then proceed

```java
public class Semaphore implements ...
{
    ...
    public void release() {
        sync.releaseShared(1);
    }
    ...
}
```
Overview of Other Java Semaphore Methods
Overview of Other Java Semaphore Methods

- There are many other Semaphore methods

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void acquire(int permits)</code></td>
<td>Acquires # of permits from semaphore, blocking until all are available, or thread interrupted</td>
</tr>
<tr>
<td><code>void acquireUninterruptibly(int permits)</code></td>
<td>Acquires # of permits from semaphore, blocking until all available</td>
</tr>
<tr>
<td><code>boolean tryAcquire(int permits)</code></td>
<td>Acquires given # of permits from semaphore, only if all are available at the time of invocation</td>
</tr>
<tr>
<td><code>void release(int permits)</code></td>
<td>Releases # of permits, returning them to semaphore</td>
</tr>
<tr>
<td><code>boolean tryAcquire(long timeout, TimeUnit unit)</code></td>
<td>Acquires a permit from semaphore, if one is available within given waiting time &amp; thread has not been interrupted</td>
</tr>
<tr>
<td><code>boolean tryAcquire(int permits, long timeout, TimeUnit unit)</code></td>
<td>Acquires given # of permits from semaphore, if all available within given waiting time &amp; current thread has not been interrupted</td>
</tr>
</tbody>
</table>
Overview of Other Java Semaphore Methods

- There are many other Semaphore methods
- Some methods can acquire or release multiple permits at a time

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void acquire(int permits)</td>
<td>Acquires # of permits from semaphore, blocking until all are available, or thread interrupted</td>
</tr>
<tr>
<td>void acquireUninterruptibly(int permits)</td>
<td>Acquires # of permits from semaphore, blocking until all available</td>
</tr>
<tr>
<td>boolean tryAcquire(int permits)</td>
<td>Acquires given # of permits from semaphore, only if all are available at the time of invocation</td>
</tr>
<tr>
<td>void release(int permits)</td>
<td>Releases # of permits, returning them to semaphore</td>
</tr>
<tr>
<td>boolean tryAcquire(long timeout, TimeUnit unit)</td>
<td>Acquires a permit from semaphore, if one is available within given waiting time &amp; thread has not been interrupted</td>
</tr>
<tr>
<td>boolean tryAcquire(int permits, long timeout, TimeUnit unit)</td>
<td>Acquires given # of permits from semaphore, if all available within given waiting time &amp; current thread has not been interrupted</td>
</tr>
</tbody>
</table>
Overview of Other Java Semaphore Methods

- There are many other Semaphore methods
- Some methods can acquire or release multiple permits at a time
- Likewise, some of these methods use timeouts

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void acquire(int permits)</code></td>
<td>Acquires # of permits from semaphore, blocking until all are available, or thread interrupted</td>
</tr>
<tr>
<td><code>void acquireUninterruptibly(int permits)</code></td>
<td>Acquires # of permits from semaphore, blocking until all available</td>
</tr>
<tr>
<td><code>boolean tryAcquire(int permits)</code></td>
<td>Acquires given # of permits from semaphore, only if all are available at the time of invocation</td>
</tr>
<tr>
<td><code>void release(int permits)</code></td>
<td>Releases # of permits, returning them to semaphore</td>
</tr>
<tr>
<td><code>boolean tryAcquire(long timeout, TimeUnit unit)</code></td>
<td>Acquires a permit from semaphore, if one is available within given waiting time &amp; thread has not been interrupted</td>
</tr>
<tr>
<td><code>boolean tryAcquire(int permits, long timeout, TimeUnit unit)</code></td>
<td>Acquires given # of permits from semaphore, if all available within given waiting time &amp; current thread has not been interrupted</td>
</tr>
</tbody>
</table>

Ironically, the timed tryAcquire() methods do honor the fairness setting, so they don’t “barge”