Chapter 14
Building Custom Synchronizers

Alfred Theorin, Ph.D. student

Department of Automatic Control

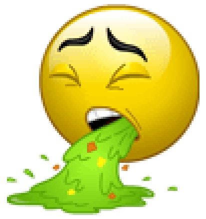
Lund University

Presentation Outline

- Why synchronizers
- Intrinsic condition queues
- Explicit conditions
- AbstractQueuedSynchronizer

Example: No Synchronizer

```
public class GrumpyBoundedBuffer {
  ...
  public synchronized void put(V v)
                          throws BufferFullException {
    if (isFull) {
      throw new BufferFullException();
    }
    doPut(v);
  }
}
```

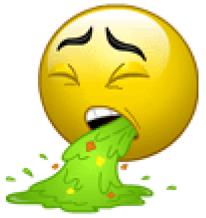


Makes caller code really messy.

Don't do this.

Example: No Synchronizer

```
public class SleepyBoundedBuffer {
  ...
  public void put(V v) throws InterruptedException {
    while (true) {
      synchronized (this) {
        if (!isFull) {
          doPut(v);
          return;
        }
      }
      Thread.sleep(...); // Busy wait vs unreponsive
    }
  }}
```



Messy and inefficient. *Don't do this.*

Why Synchronizers

Efficient encapsulation of state-based preconditions

Intrinsic Condition Queues: Precondition support for intrinsic locks

```
public class Object {
  // Temporarily release lock, suspends thread
  public void wait() { ... }

  // Wake up one thread suspended on the lock
  public void notify() { ... } // Use with care!

  // Wake up all threads suspended on the lock
  public void notifyAll() { ... }

  ...
}
```

Caller must hold the intrinsic lock.

Example: Intrinsic Condition Queues

```
public class BoundedBuffer {
  ...
  public synchronized void put(V v)
                         throws InterruptedException {
    while (isFull) {
      wait();
    }
    doPut(v);
    notifyAll();
  }
}
```

Intrinsic Condition Queues

Drawbacks

- Easy to make errors
- One queue, possibly many preconditions
- Tricky to encapsulate
  - Inheritance
  - Intrinsic lock

Big Advantage

- Easy to use

Explicit Conditions: Conditions for an Explicit Lock

```
public interface Condition {
  // Release lock temporarily, suspend thread
  public void await() { ... } // Careful, not wait

  // Wake up one thread suspended on this condition
  public void signal() { ... }

  // Wake up all threads suspended on this condition
  public void signalAll() { ... }

  ...
}
```

Caller "must" hold the explicit lock.

Example: Explicit Conditions

```
public class BoundedBuffer {
  private Condition nonFull  = lock.newCondition();
  private Condition nonEmpty = lock.newCondition();
  public void put(V v) throws InterruptedException {
    lock.lock();
    try {
      while (isFull) {
        nonFull.await();
      }
      doPut(v);
      nonEmpty.signal();
    } finally {
      lock.unlock();
    }} ... }
```

AbstractQueuedSynchronizer

- Framework to build synchronizers
- Used by many built-in synchronizers
- Encapsulates the locking and the blocking

Example: AbstractQueuedSynchronizer

```
public class Latch {
  public void signal() {
    sync.releaseShared(0);
  }

  public void await() {
    sync.acquireShared(0);
  }

  private AQS sync = new AbstractQueuedSynchronizer() {
    public boolean tryReleaseShared(int ignored) {
      setState(1);
      return true;
    }
    public int tryAcquireShared(int ignored) {
      return (getState() == 1) ? 1 : -1;
    }
  };
}
```

Summary

- Why synchronizers
- Intrinsic condition queues
- Explicit conditions
- AbstractQueuedSynchronizer