# Lecture 7: Coordination 3

Adam Hawley

February 20, 2019

## Contents



## 1 Recapping Java

In Java, there are two approaches to coordination:

- Extend Thread and override run()
- Declare a class that implements Runnable

### 1.1 Shared Variable Coordination

In Java, every object has a **lock** which enforces mutual exclusion. The lock can be used in two different ways:

- The `synchronized` method modifier – Produces a monitor
- The `synchronized` statement – Defines a critical region

Conditional Synchronisation uses methods from the top-level `Object` class:

`wait()` Always blocks when executed, but releases the lock.

`notify()` Wakes up another thread waiting on the currently held lock:

- At most one thread is woken

- Thread selection is non-deterministic
- Woken thread **waits** to obtain the lock

`notifyAll()` Wakes up all waiting threads (which access under normal mutual exclusion)

**LIVE CODE SECTION MISSED AS NOT ON RECORDING**

`Join()` is a method called by a main thread to *join* its subthread and wait for it to complete to ensure consistency throughout both threads.

## 2 Readers/Writers Problem

Different implementation strategies exist:

- We choose to give priority to waiting writers
- All (new) readers are blocked if any writer is available.