Miscellaneous Concepts

- → Locking - Reentrant lock
- → Read Write lock
- → Code: wait(), notify(), notifyall()

→ Thread lifecycle.

↓

Drive link

6 States.

① Synchronized

```
public void method() {

    synchronised (lockobj) {
        ≡ CS       shared Resource.doSomething();
    }
}
```

② lock lock = new ReentrantLock()

```
public void method() {

        lock.lock();
        CS ≡      sharedRes.doSomething()
        lock.unlock();
}
```

$\underbrace{\text{doSomething()}}$ → throwed an exception

lock.unlock(); ↓ Never execute if CS throws an Exception.

↳ other threads will wait infinitely to get this resource

3

```
lock. lock();
try {
        CS ≡

}
catch {


}
finally {
        lock.unlock();
}
```

// Always execute irr of exception is thrown or not

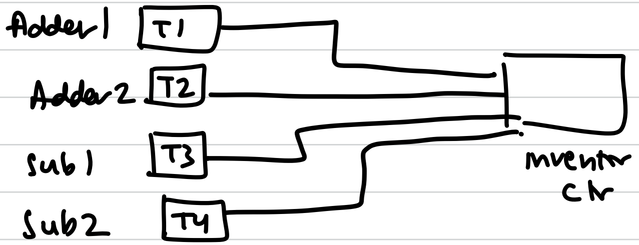
2

Fairness of the lock

additional CPU cost.

lock lock = new ReentrantLock(true);

All threads get a fair chance to acquire lock on this obj.



```
add() {
    ctr++
}

sub() {
    ctr--
}
```

$\underbrace{\text{T1 T2 T1 T1 T2 T3 ....}}$

- T1 T2 T3 T4 T1 T2 T3 T4 ....

→ No diff. if lock is available.

→

③ Try Lock

Ⓘ

```
· lock. lock();   ✓ → waiting stage.
                      until lock
                      becomes
                      available
                      again.
  try {
        CS   ✓
  }
  catch {

  }
  finally {
        lock.unlock()   ✓
  }
  --- ↓
```

Ⅱ

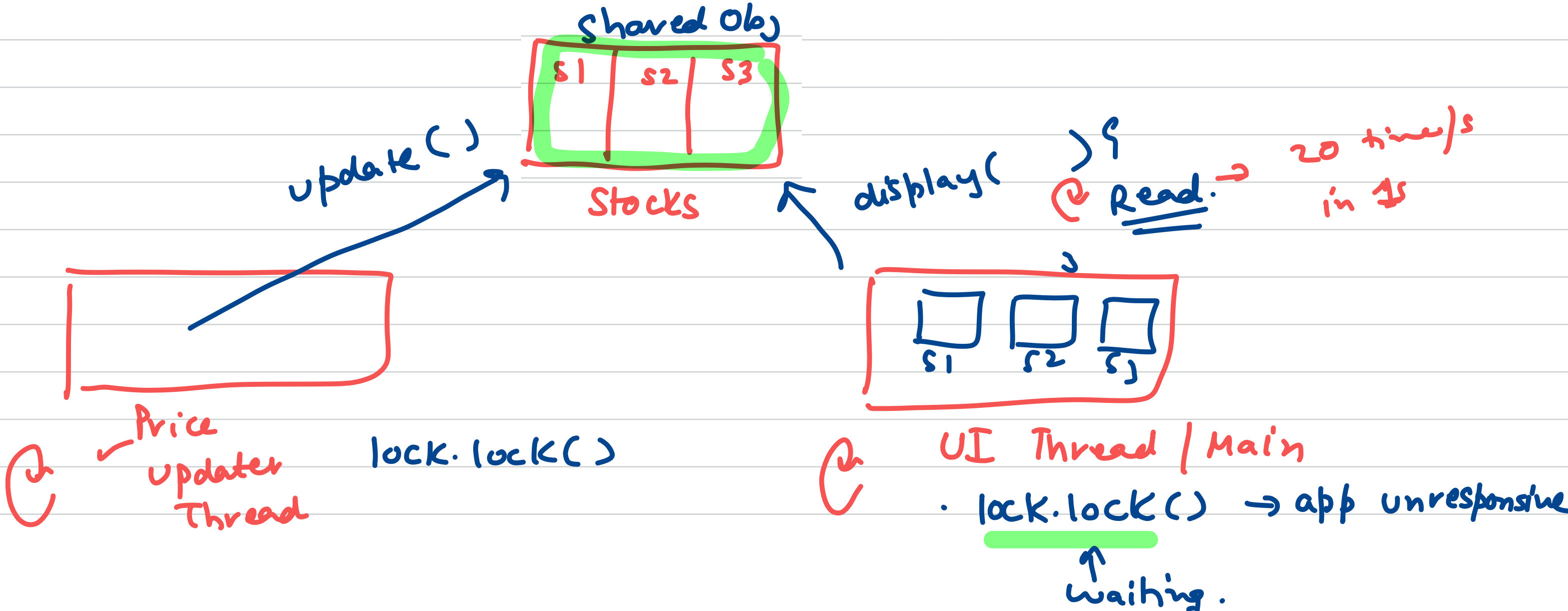
```
if( lock. try Lock() ){        false
    try {
        CS      ↓
    }
    catch {

    }
    finally {
        lock.unlock()   ↓
    }
}
else {
    -----
    -----      } else block
    -----        will
                 still
                 execute and
                 thread is
                 not blocked,
                 it can do
                 some other
                 work.
}
--- ↓
```

Real Use cases:

1) Video / image Processing

2) high speed / low latency → trading applications

3) User Interface.



UI ?

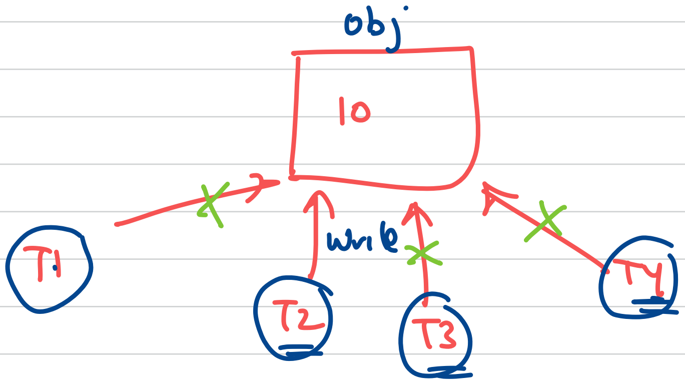
- lock.trylock(){

→ immediately T/F

}

↓ ≡

4 Read Write Locks

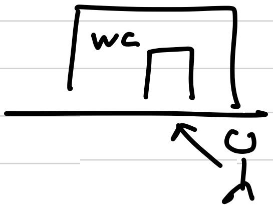


All of them can acquired Read lock given no thread has req. for a write lock

Read Lock – If no thread acquired the write lock or requested for it, multiple threads can acquire the read lock.

Write Lock – If no threads are reading or writing, only one thread can acquire the write lock.

5 Busy - waiting



knock knock .....

Wasting lot of CPU cycles

wait() , notify()

→ Produce
→ wait()
→ Do produce another T-shirt

→ Buy
→ notify()
→ ...