Chapter. 15

자바 Thread 프로그래밍

# multi-thread 프로그래밍

FAST CAMPUS ONLINE

자바 기초

강사. 박은종

Chapter. 15

# 03 multi-thread 프로그래밍

# I 임계 영역(critical section)

두 개 이상의 thread가 동시에 접근하게 되는 리소스

critical section에 동시에 thread 가 접근하게 되면 실행 결과를 보장할 수 없음

thread간의 순서를 맞추는 동기화(synchronization) 이 필요

# 동기화 (synchronization)



임계 영역에 여러 thread가 접근 하는 경우 한 thread가 수행 하는 동안 공유 자원을 lock 하려 다른 thread의 접근을 막음

동기화를 잘못 구현하면 deadlock에 빠질 수 있음

# I critical section과 동기화

# I 자바에서 동기화 구현

## synchronized 수행문과 synchronized 메서드를 이용

### synchronized 수행문

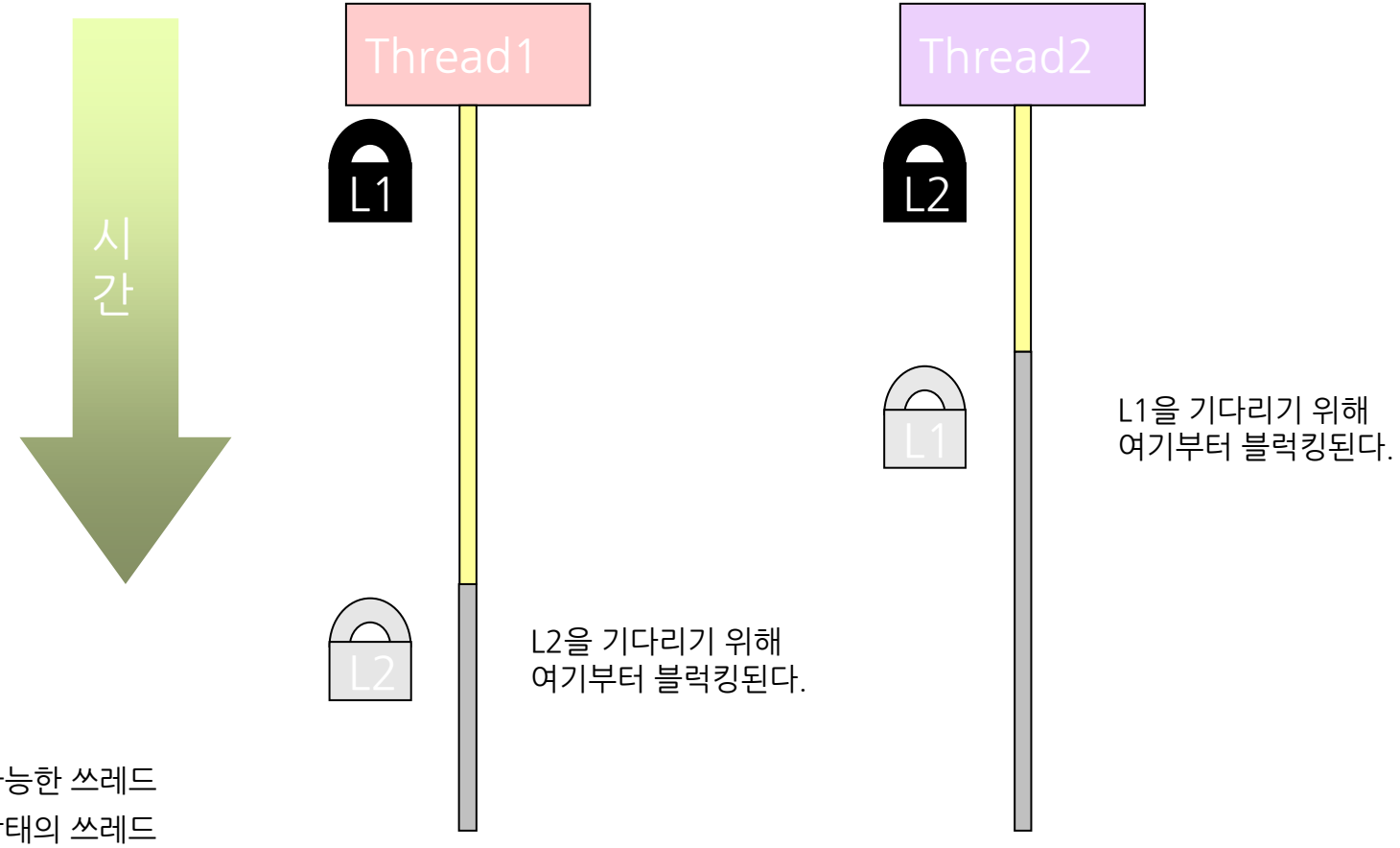
```
synchronized(참조형 수식) {

}
```

참조형 수식에 해당되는 객체에 lock을 건다.

### synchronized 메서드

현재 이 메서드가 속해 있는 객체에 lock을 건다.
synchronized 메서드 내에서 다른 synchronized 메서드를 호출하지 않는다.
(deadlock 방지위해)

# deadlock



# I wait()/notify()

wait() : 리소스가 더 이상 유효하지 않은 경우 리소스가 사용 가능할 때 까지 위해
thread를 non-runnable 상태로 전환
wait() 상태가 된 thread은 notify() 가 호출 될 때까지 기다린다.

notify(): wait() 하고 있는 thread 중 한 thread를 runnable 한 상태로 깨움

notifyAll() : wait() 하고 있는 모든 thread가 runnable 한 상태가 되도록 함
notify() 보다 notifyAll()을 사용하기를 권장
특정 thread가 통지를 받도록 제어 할 수 없으므로 모두 깨운 후
scheduler에 CPU를 점유하는 것이 좀더 공평하다고 함