# 22.12.22

## 🎯Java의 동기화

### 0) 동기화의 기능 2가지

- 배타적 실행 제어 (쓰기)
- 스레드 사이의 안정적 통신(읽기)

  *원자적 연산: 중단이 불가능한 연산

  *Java언어에서 스레드가 (원자적 데이터 값을 가지더라도) 필드를 읽을 때 수정이 완전히 반영된 값을 얻는다고 보장하지 않는다. - 다른 스레드에서 값이 변경되었을 수 있다는 말.

### 1) Volatile

- 항상 스레드가 캐시 없이 메인 메모리 영역에서 값을 참조. → 가시성 문제 해결
- 원자적 연산에서의 동기화 보장 (ex a=1 읽기, b=false 저장, b=a -a의 입장에서는 읽기 연산)
- 비원자적 연산에 대해서는 동기화를 보장할 수 없다. (ex a = a+1  - 레지스터에서 총 3개의 연산)

  → 배타적 실행 제어를 위해서는 synchronized를 사용해야 한다. 혹은 Atomic 클래스 사용

### 2) Synchronized

- Object 클래스와 이를 상속하는 클래스들 및 인스턴스, 즉 자바의 모든 객체는 Monitor 방식을 통해 동기화될 수 있다.
- Monitor구성 요소
  - Mutex Lock
  - 스레드가 대기할 수 있는 자료구조 - wait set(Wait Pool), Entry set(Entry Set)
  - Condition variable - Object의 wait(), notify(), notifyAll()
- synchronized 키워드를 통해 컴파일러는 moniterenter, moniterexit 를 해당 구문의 앞뒤에 추가하여 모니터가 시작하고 종료될 수 있도록 한다.

- monitor를 통해 자바는 상호 배제, 협력(스레드 간의 접근 순서 제어)가 가능하다.
- Java의 동기화가 필요한 대부분의 클래스에는 범위의 차이는 있으나 synchronized가 사용됨

  ex) Concurrenthashmap - 버킷 삽입 부분에 synchronized 사용
- 비동기적으로 원자적 연산이 필요할 경우 Atomic 관련 클래스를 사용할 수 있다. (내부적으로 Compare And Swap, 즉 비교 후 교체 작업이 이루어진다.)

## 4) Monitor 작동 방식

1) 한 스레드가 동기화 코드 영역(Synchronized method or block)에 접근하기 위해 Entry Set에 진입

(Runnable → Blocked)

2) 모니터 락을 보유한 스레드가 있으면 반환할 때까지 Entry Set에서 대기

(Blocked)

3) 모니터 락을 보유한 스레드 없으면 Entry Set 중 하나의 스레드가 선택되어 모니터 락을 획득, 실행

(획득 스레드 Runnable / 나머지 스레드 Blocked)

4) 모니터 락 획득한 스레드가 로직 상의 조건을 만족하지 않을 경우, wait()를 호출하여 해당 스레드를 wait set에 넣을 수 있다. 모니터 락을 획득한 스레드만이 wait() 호출이 가능하다.

(Runnable → Blocked)

*wait set에 들어간 스레드는 모니터 락을 획득한 스레드의 notify() 혹은 notifyAll() 호출을 통해서만 Entry Set으로 이동시킬 수 있다. 그러나 wait set에서 Entry Set으로의 이동일 뿐 lock을 획득하는 것과의 직접적인 관련은 없다. 따라서 모니터 락은 뮤텍스와 시그널 매커니즘이 결합된 방식 이라고 볼 수 있다.

참고: https://happy-coding-day.tistory.com/entry/JAVA-모니터란-무엇인가, https://jyami.tistory.com/112 , https://bestugi.tistory.com/40 , https://about-myeong.tistory.com/34