(디지털컨버전스)스마트 콘텐츠와 웹 융합응용SW개발자 양성과정

강사 - Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 –Joongyeon Kim(김중연)
jjjr69@naver.com

2021년 6월 3일 질문노트

[김중연]

Mutex vs Spinlock의 차이점

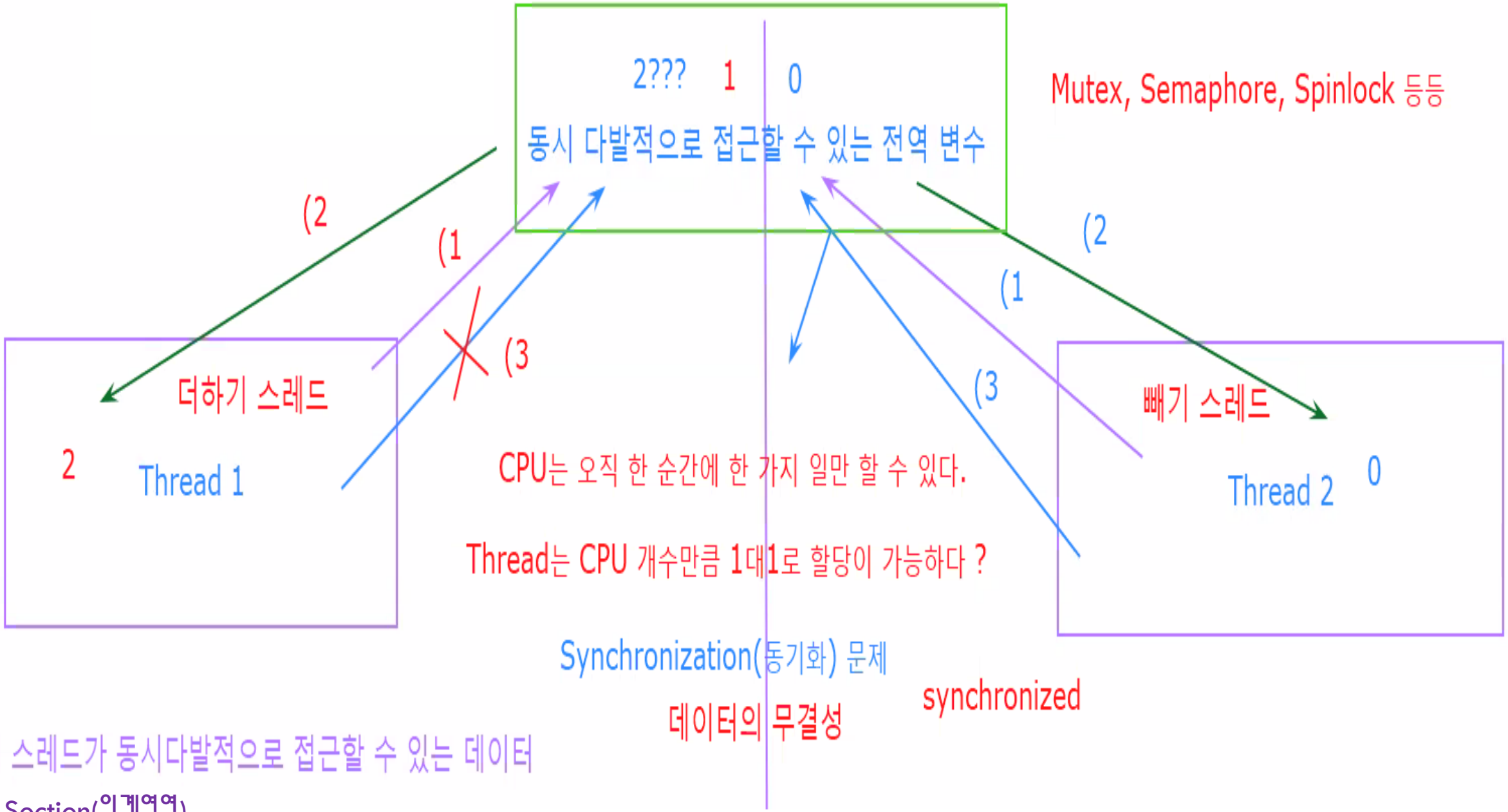
Mutex는 콘텍스트 스위칭을 사용해서 여러 정보들을 많이 한꺼번에 처리할수있지만 적은 데이터를 처리할때는 오히려 스위칭하는게 손해가된다

Spinlock은 하나의 정보를 빠르게 처리할수있지만 처리할 데이터가 크면 나머지는 처리가 될때까지 기다려야하는 불상사가 생긴다

요약

복잡한 작업을 할 때는 mutex를 사용하고

단순한 작업에는 Spinlock을 사용하는 것이 효율적이다



여러개의 스레드가 동시다발적으로 접근할 수 있는 데이터

Critical Section(임계영역)

Multi-Tasking
스타 선수들을 보면 화면 전환이 오지게 빠르고
읶닉 컨트롤 하면서 생산하고 일꾼찍고 견제도 하고 난리가 납니다.
(엄밀하게 말해서 아주 빠르게 순차적으로 일어날 뿐 실제로는 정말 동시에 일어나지 않습니다)
컴퓨터의 Multi-Tasking 개념도 동일한데
아주 빠르게 순차적으로 동작하기 때문에 우리가 느끼기엔 동시에 발생하는것 처럼 느껴질 뿐

최고 운동 신경을 가진 운동 선수가 육안으로 식별할 수 있는 가장 짧은 시간은 몇 초일까요 ?
0.3x 초 정도라고 합니다.
1초 -> 20억, 0.1초 -> 2억 ===> 0.3초 ===> 6억, 0.3x초 ===> 6 ~ 7억 사이의 명령어를 처리할 수
있음을 의미함
현재 작업표시줄에 나타나는 프로세스가 대략 100개 정도 됩니다.
100개의 프로세스가 모두 명령 2000개씩을 실행한다고 해봐야 200000 <<< 20만개 밖에 안되죠.
만약 모든 프로세스가 0.0001초씩 동작을 한다고 가정해보자!
대략 1000개 프로세스가 모두 한 바퀴 순회하는데 얼마나 걸리나요 ? 0.1초
(아주 빠르게 순차적으로 일어날 뿐 ...)
지금까지 진행했던 이 과정 자체를 우리는 Multi-Tasking 이라고 부릅니다.

컨텍스트 스위칭이란?

프로세스 p1, 프로세스 p2가 존재할 때
p1이 실행중일 때 p2가 대기하고 p2가 실행중이면 p1이 대기하면 지금까지 작업해온 내용을 모두 어딘가에 저장해야하는데 그것이 PCB라는 곳이다
즉 p1은 PCB에 저장해야하고 p2가 기지고 있던 데이터는 PCB에서 가져온다

이렇게 p1과 p2가 서로 대기<->실행을 번갈아가며하는 것을 컨텍스트 스위칭(Context Switching)이라고 한다

```java
import java.util.Random;

class Car implements Runnable{

    String name;
    private int sleepTime;
    private final static Random generator = new Random();

    public Car(String name){
        this.name=name;
        //Random 클래스로 만든 객체 nextInt() 메소드를 통해서도 랜덤값을 생성할 수 있다
        sleepTime=generator.nextInt( bound: 3000) + 3000;
    }

    // 스레드를 돌릴땐 무조건 이 run()부분을 구동시키게 되어있다
    // 매우 중요하니 이 run()을 반드시 기억해두자!
    @Override
    public void run() {
        //try라는 키워드를 적는 경우는 I/O나 특정한이벤트,
        //인터럽트 등등이 발생하는 경우에 작성하게 됨
        //이 녀석은 에러를 유발할 수도 있다! 를 암시함
        try{
            Thread.sleep(sleepTime);
        } catch (InterruptedException e){
            //정말 에러가 발생했다면 여기로 온다
            //물론 Thread.sleep()에서 에러가 발생할 일은 99.99999999%없다

        }
        System.out.println(name + "이 경주를 완료하였습니다!");
    }
}
public class ThreadTest {
    public static void main(String[] args) {
        Thread t1 = new Thread(new Car( name: "Ferrari"));
        Thread t2 = new Thread(new Car( name: "Spyder 918"));
        Thread t3 = new Thread(new Car( name: "Maserati MC20"));

        t1.start();
        t2.start();
        t3.start();

    }
}
```

1. Runnable (준비상태)

스레드가 실행되기 위한 준비단계. CPU를 점유하고 있지 않으며 실행(Running 상태)을 하기 위해 대기하고 있는 상태입니다.
코딩 상에서 start( ) 메소드를 호출하면 run( ) 메소드에 설정된 스레드가 Runnable 상태로 진입함.
"Ready" 상태라고도 한다.

스레드를 돌릴때에는 run()부분을 구동시킨다(중요)

Try 키워드는 I/O나 특정이벤트,인터럽트 등등이 발생하는 경우에 작성한다

이 코드는 에러를 유발할 수 있다! 라는 의미이다