14장 멀티스레드 예제

경쟁 조건 (Race Condition)

다음 코드는 두 개의 스레드가 공유 변수 **count**를 동시에 증가시키는 작업을 수행합니다.

이로 인해 경쟁 조건이 발생할 수 있습니다.

다음 코드를 완성하여 경쟁 조건을 해결하는 코드를 작성하세요.

위 코드에서 **synchronized (lock)** 블록 내에서 **count** 변수를 증가시키는 것은 한 번에 하나의 스레드만 **lock** 객체를 획득하여 해당 블록을 실행하도록 보장합니다.

이를 통해 여러 스레드가 동시에 **count** 변수를 수정하는 것을 방지하여 경쟁 조건을 해결할 수 있습니다.

```
public class RaceConditionFixed {
    private static int count = 0;
    private static final Object lock = new Object();

    public static void main(String[] args) throws
InterruptedException {
        Thread thread1 = new Thread(() -> {
            for (int i = 0; i < 1000; i++) {
                synchronized (lock) {
                    count++;
                }
            }
        });

        Thread thread2 = new Thread(() -> {
            for (int i = 0; i < 1000; i++) {
                synchronized (lock) {
                    count++;
                }

            }
        });

        thread1.start();
        thread2.start();

        thread1.join();
        thread2.join();

        System.out.println("Count: " + count);
    }
}
```

2 참고-자바스레드 생성 방식

1. Thread 클래스 상속

이 방식은 Thread 클래스를 직접 상속받아서 새로운 스레드를 정의하는 방식입니다. 이 방식은 Thread 클래스의 run() 메서드를 오버라이딩하여 스레드가 실행할 코드를 정의합니다.

```
public class MyThread extends Thread {
    public void run() {
        // 스레드가 실행할 코드
        for (int i = 0; i < 5; i++) {
            System.out.println("Hello from MyThread " + i);
            try {
                Thread.sleep(1000); // 1초간 일시 정지
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }

    public static void main(String[] args) {
        // 새로운 스레드 객체 생성
        MyThread myThread = new MyThread();

        // 스레드 시작
        myThread.start();
    }
}
```

MyThread 클래스는 Thread 클래스를 상속하고 있습니다. 따라서 MyThread 클래스는 Thread 클래스의 모든 기능을 상속받습니다.

run() 메서드는 Thread 클래스의 run() 메서드를 오버라이드하여 스레드가 실행할 코드를 정의합니다. 위의 코드에서는 단순히 0부터 4까지의 숫자를 출력하고 1초간 일시 정지하는 작업을 반복하도록 작성되어 있습니다.

main() 메서드에서는 MyThread 클래스의 객체를 생성하고, 이 객체를 통해 start() 메서드를 호출하여 스레드를 시작합니다. start() 메서드를 호출하면 새로운 스레드가 생성되고 run() 메서드가 호출되어 스레드가 실행됩니다.

이러한 방식으로 Thread 클래스를 상속하여 스레드를 생성하면 run() 메서드를 오버라이드하여 스레드가 실행할 코드를 직접 정의할 수 있습니다. 하지만 자바는 단일 상속만을 지원하므로 다른 클래스를 상속받을 수 없는 단점이 있습니다. 따라서 보다 유연한 구현을 위해서는 Runnable 인터페이스를 구현하는 방식을 사용하는 것이 권장됩니다.

참고-자바스레드 생성 방식

2. Runnable 인터페이스 구현

```
public class MyRunnable implements Runnable {
    public void run() {
        // 스레드가 실행할 코드
        for (int i = 0; i < 5; i++) {
            System.out.println("Hello from MyRunnable " + i);
            try {
                Thread.sleep(1000); // 1초간 일시 정지
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }

    public static void main(String[] args) {
        // Runnable 인터페이스를 구현한 객체 생성
        MyRunnable myRunnable = new MyRunnable();

        // Thread 객체 생성 및 Runnable 객체 전달
        Thread thread = new Thread(myRunnable);

        // 스레드 시작
        thread.start();
    }
}
```

MyRunnable 클래스는 Runnable 인터페이스를 구현하고 있으며, run() 메서드를 오버라이드하여 스레드가 실행할 코드를 정의하고 있습니다.

main() 메서드에서는 MyRunnable 객체를 생성하고, 이 객체를 Thread 클래스의 생성자에 전달하여 Thread 객체를 생성합니다.

그리고 이후에 start() 메서드를 호출하여 스레드를 시작합니다.

감사합니다