 

Вище представлений 2 результати виконання одного і того самого коду.

Суть в тому що в Simpl методах немає синхронізації в потоках. Отже методи одночасно звертаються до зміної і може получатись різний або неочікуваний результат.

А в другому методі використовуються синхронізовані функії, що означає -

клас по черзі звертається до змінної, а результат буде завжди одинаковим.

public class Main {

    static MyThread thread1;

    static MyThread thread2;

    static MyThread thread3;

    static MyThread thread4;

    public static void main(String[] args) throws InterruptedException {

        Counter counter1 = new Counter();

        CounterSync counter2 = new CounterSync();

        thread1 = new MyThread(counter1);

        thread2 = new MyThread(counter1);

        System.out.println("First case, with Simple method");

        thread1.start();

        thread2.start();

        thread1.join();

        thread2.join();

        Thread.sleep(2000);

        System.out.println("Second case, with Sync method");

        new MyThread(counter2).start();

        new MyThread(counter2).start();

    }

}

class MyThread extends Thread {

    myint counter = null;

    public MyThread(myint param){

        this.counter = param;

    }

    @Override

    public void run() {

        try {

            counter.increment();

            System.out.println(counter.value());

            System.out.println("-");

            Thread.sleep(500);

            counter.decrement();

            System.out.println(counter.value());

        } catch (InterruptedException e) {

        }

    }

}

interface myint{

    public void increment() throws InterruptedException;

    public void decrement() throws InterruptedException;

    public int value();

}

class Counter implements myint {

    private int c = 0;

    public void increment() throws InterruptedException {

        int a;

        Thread.sleep(150);

        a = c;

        a++;

        c = a;

    }

    public void decrement() throws InterruptedException {

        int a;

        Thread.sleep(100);

        a = c;

        a--;

        c = a;

    }

    public int value() {

        return c;

    }

}

class CounterSync implements myint {

    private int c = 0;

    public synchronized void increment() throws InterruptedException {

        int a;

        Thread.sleep(150);

        a = c;

        a++;

        c = a;

    }

    public synchronized void decrement() throws InterruptedException {

        int a;

        Thread.sleep(100);

        a = c;

        a--;

        c = a;

    }

    public synchronized int value() {

        return c;

    }

}