链滴

# 多线程之回环栅栏 CyclicBarrier 的使用

作者：MaidongAndYida

原文链接：https://ld246.com/article/1554887892540

来源网站：链滴

# 场景定义

假设有一个场景，有一场百米赛跑运动会，在三名选手都跑完之后，要求算出所有人的平均成绩。那就可以用到CyclicBarrier了。

## 要求1.所有人跑完之后才能计算平均值。

## 要求2.计算平均值用多线程运算(因为换成三亿人进行赛跑时也适用)

```
public class CyclicBarrier1 implements Runnable{

    //创建初始化3个线程的线程池
    private ExecutorService threadPool =Executors.newFixedThreadPool(3);

    //创建3个CyclicBarrier对象,执行完后执行当前类的run方法
    CyclicBarrier cb = new CyclicBarrier(3,this);

    //保存三组随机数的平均值
    private ConcurrentHashMap<String, Integer> map=new ConcurrentHashMap<String,Integ
r>();

    //计算三组随机数的平均值
    public void count(){
        for (int i = 0; i < 3; i++) {
            threadPool.execute(() -> {
                //生成三个数，取值范围1-10
                int random1 = (int)(Math.random()*10)+1;
                int random2 = (int)(Math.random()*10)+1;
                int random3 = (int)(Math.random()*10)+1;
                int average = (random1+random2+random3)/3;
```

```
                map.put(Thread.currentThread().getName(),average);
                System.out.println(Thread.currentThread().getName()+"的平均数为"+average);
                try {
                    cb.await();
                } catch (InterruptedException e) {
                    e.printStackTrace();
                } catch (BrokenBarrierException e) {
                    e.printStackTrace();
                }
            });
        }
    }

    @Override
    public void run() {
        Iterator<Map.Entry<String, Integer>> iterator = map.entrySet().iterator();
        double add = 0;
        while(iterator.hasNext()) {
            Map.Entry<String, Integer> next = iterator.next();
            String key=next.getKey();
            int value=next.getValue();
            add += value;
            System.out.println(key+" "+value);
        }
        add /=3;
        System.out.println("三组平均数为：  "+add);
        threadPool.shutdownNow();
    }

    public static void main(String[] args) {
        CyclicBarrier1 cb=new CyclicBarrier1();
        cb.count();
    }
}
```

## 方法简介

CyclicBarrier有两个构造函数:

CyclicBarrier(int parties): int类型的参数表示有几个线程来参与这个屏障拦截

CyclicBarrier(int parties,Runnable barrierAction): 当所有线程到达一个屏障点时,优先执行arrierAction这个线程的run().

await(): 每个线程调用await(),表示我已经到达屏障点,然后当前线程被阻塞