# Java 中的线程：对象锁 V

**切坦·库达尔卡**

**评价我：** 5.00/5 (1 票)

2021 年 6 月 18 日 警察

一个线程可以同时获取多个锁吗？

这与我上一篇关于"Java 中的线程：对象锁 - IV"的文章是连续的。一个线程可以同时获取多个锁吗？是我们将在今天的文章中回答的问题。

## 介绍

一个线程可以同时获取多个锁吗？ . 而这个问题的答案是肯定的！Java 中的一个线程可以处理多个锁。采取以下程序。

缩小▲ 复制代码

```
locks.java
-----------
class processor implements Runnable{
    Object objLock;
    String name;
    public processor(String name, Object objLock){
        this.objLock = objLock;
        this.name = name;
    }
    public void run() {
        display();
    }
    public void display() {
        synchronized(this){
          synchronized(objLock) {
            for (int i = 0 ; i<= 5; i++) {
                System.out.println("i = " + i + " In thread" + Thread.currentThread() +
"acquired lock on this i.e. " + name + " and objLock");
            }
          }
        }
    }
}

class locks {
    public static void main(String[] args) {
        Object objLock = new Object();
        processor p1 = new processor("p1",objLock);
        processor p2 = new processor("p2",objLock);
        Thread t1 = new Thread(p1, "t1");
        Thread t2 = new Thread(p1, "t2");
```

```
        Thread t3 = new Thread(p2, "t3");
        Thread t4 = new Thread(p2, "t4");

        t1.start();
        t2.start();
        t3.start();
        t4.start();
    }
}


Output:
-------
i = 0 In threadThread[t1,5,main]acquired lock on this i.e. p1 and objLock
i = 1 In threadThread[t1,5,main]acquired lock on this i.e. p1 and objLock
i = 2 In threadThread[t1,5,main]acquired lock on this i.e. p1 and objLock
i = 3 In threadThread[t1,5,main]acquired lock on this i.e. p1 and objLock
i = 4 In threadThread[t1,5,main]acquired lock on this i.e. p1 and objLock
i = 5 In threadThread[t1,5,main]acquired lock on this i.e. p1 and objLock
i = 0 In threadThread[t3,5,main]acquired lock on this i.e. p2 and objLock
i = 1 In threadThread[t3,5,main]acquired lock on this i.e. p2 and objLock
i = 2 In threadThread[t3,5,main]acquired lock on this i.e. p2 and objLock
i = 3 In threadThread[t3,5,main]acquired lock on this i.e. p2 and objLock
i = 4 In threadThread[t3,5,main]acquired lock on this i.e. p2 and objLock
i = 5 In threadThread[t3,5,main]acquired lock on this i.e. p2 and objLock
i = 0 In threadThread[t2,5,main]acquired lock on this i.e. p1 and objLock
i = 1 In threadThread[t2,5,main]acquired lock on this i.e. p1 and objLock
i = 2 In threadThread[t2,5,main]acquired lock on this i.e. p1 and objLock
i = 3 In threadThread[t2,5,main]acquired lock on this i.e. p1 and objLock
i = 4 In threadThread[t2,5,main]acquired lock on this i.e. p1 and objLock
i = 5 In threadThread[t2,5,main]acquired lock on this i.e. p1 and objLock
i = 0 In threadThread[t4,5,main]acquired lock on this i.e. p2 and objLock
i = 1 In threadThread[t4,5,main]acquired lock on this i.e. p2 and objLock
i = 2 In threadThread[t4,5,main]acquired lock on this i.e. p2 and objLock
i = 3 In threadThread[t4,5,main]acquired lock on this i.e. p2 and objLock
i = 4 In threadThread[t4,5,main]acquired lock on this i.e. p2 and objLock
i = 5 In threadThread[t4,5,main]acquired lock on this i.e. p2 and objLock
```

复制代码

上面的程序编译并成功运行，为我们提供了所需的预期输出。如果您仔细观察输出，您会发现我们在程序中创建的每个线程都能够根据上下文同时获取对“this”对象即 p1 或 p2 以及 objLock 的锁。

在 main() 中，线程 t1 和 t2 是从处理器对象 p1 创建的，名称 p1 和从 Object 类创建的 objLock 被传递给它的构造函数。接下来，线程 t3 和 t4 从处理器对象 p2 创建，名称 p2 和从 Object 类创建的 objLock 被传递给它的构造函数。因此 objLock 可以被所有线程看到。run() 中的线程表示函数是 display() 函数，它具有在“this”和“objLock”中同步的 for 循环。因此，任何线程都必须获得对自身的锁以及每个线程都可以看到的锁 objLock。

当所有线程都启动时，t1 和 t2 竞争获取 p1 上的锁，而 t3 和 t4 竞争获取 p2 上的锁，因为它们分别是从 p1 和 p2 创建的。假设 t1 获取 p1 上的锁，t3 获取 p2 上的锁，然后 t1 和 t3 都竞争 objLock。假设 t1 获得了 objLock 的锁并赢得了比赛。现在，t1 将在同时拥有两个锁的同时执行 for 循环。完成后，t1 将释放两个 te 锁，t3 有机会执行 for 循环。现在类似的竞争条件发生在 t2 和 t4 之间，我们得到了一个清晰的顺序输出。

# 兴趣点

希望这个例子清楚地表明一个线程可以获取和拥有多个锁。这是一个好习惯吗？我们稍后再讨论。

# 历史

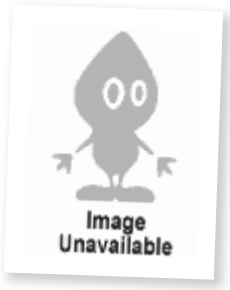
初始修订：2021 年 6 月 18 日

# 执照



# 分享

# 关于作者




**切坦·库达尔卡**
软件开发人员（高级）
印度
手表
该会员

我是一名软件工程师，拥有大约 7 年以上的经验。我的大部分经验是在存储技术方面。

# 评论和讨论

添加评论或问题 电子邮件提醒 Search Comments

第一 页上一页 下一页

**将零件放在一起**
**Klaus Luedenscheidt 18-Jun-21 11:49**

回复：把零件放在一起
**Chetan Kudalkar** 19-Jun-21 18:04

回复：把零件放在一起
**Klaus Luedenscheidt** 20-Jun-21 11:50

刷新 1

一般 新闻 建议 问题 错误 答案 笑话 赞美 咆哮 管理员

使用Ctrl+Left/Right 切换消息，Ctrl+Up/Down 切换主题，Ctrl+Shift+Left/Right 切换页面。