目录

[类的线程安全定义 1](#_Toc514706490)

[怎么才能做到类的线程安全？ 2](#_Toc514706491)

[栈封闭 2](#_Toc514706492)

[无状态 2](#_Toc514706493)

[让类不可变 2](#_Toc514706494)

[volatile 2](#_Toc514706495)

[加锁和CAS 2](#_Toc514706496)

[安全的发布 2](#_Toc514706497)

[TheadLocal 2](#_Toc514706498)

[死锁 2](#_Toc514706499)

[简单的 3](#_Toc514706500)

[动态的 3](#_Toc514706501)

[其他安全问题 3](#_Toc514706502)

[活锁 3](#_Toc514706503)

[线程饥饿 3](#_Toc514706504)

[性能和思考 3](#_Toc514706505)

[影响性能的因素 3](#_Toc514706506)

[上下文切换 3](#_Toc514706507)

[内存同步 3](#_Toc514706508)

[减少锁的竞争 4](#_Toc514706509)

[缩小锁的范围 4](#_Toc514706510)

[减少锁的粒度 4](#_Toc514706511)

[锁分段 4](#_Toc514706512)

[替换独占锁 4](#_Toc514706513)

[线程安全的单例模式 4](#_Toc514706514)

[双重检查锁定 4](#_Toc514706515)

[解决之道 4](#_Toc514706516)

[懒汉式 4](#_Toc514706517)

[饿汉式 4](#_Toc514706518)

**并发安全**

# 类的线程安全定义

如果多线程下使用这个类，不过多线程如何使用和调度这个类，这个类总是表示出正确的行为，这个类就是线程安全的。

类的线程安全表现为：

* 操作的原子性
* 内存的可见性

不做正确的同步，在多个线程之间共享状态的时候，就会出现线程不安全。

# 怎么才能做到类的线程安全？

## 栈封闭

所有的变量都是在方法内部声明的，这些变量都处于栈封闭状态。

## 无状态

没有任何成员变量的类，就叫无状态的类

## 让类不可变

让状态不可变，两种方式：

1，加final关键字，对于一个类，所有的成员变量应该是私有的，同样的只要有可能，所有的成员变量应该加上final关键字，但是加上final，要注意如果成员变量又是一个对象时，这个对象所对应的类也要是不可变，才能保证整个类是不可变的。

2、根本就不提供任何可供修改成员变量的地方，同时成员变量也不作为方法的返回值

## volatile

保证类的可见性，最适合一个线程写，多个线程读的情景，

## 加锁和CAS

## 安全的发布

类中持有的成员变量，特别是对象的引用，如果这个成员对象不是线程安全的，通过get等方法发布出去，会造成这个成员对象本身持有的数据在多线程下不正确的修改，从而造成整个类线程不安全的问题。

## TheadLocal

## Servlet

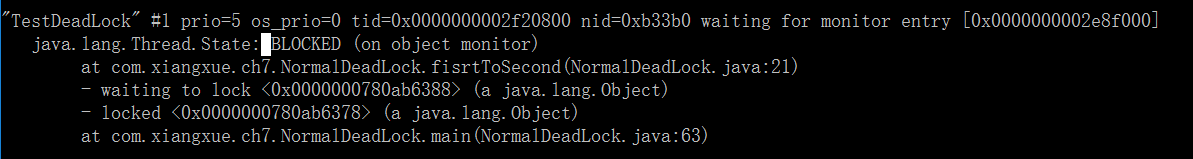
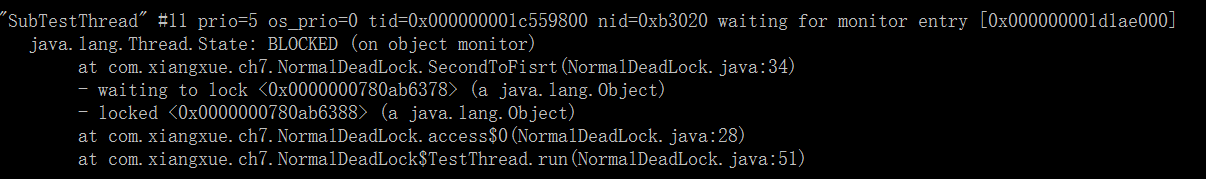
不是线程安全的类，为什么我们平时没感觉到，：2、在需求上，很少有共享的需求，第二，接收到了请求，返回应答的时候，都是由一个线程来负责的。

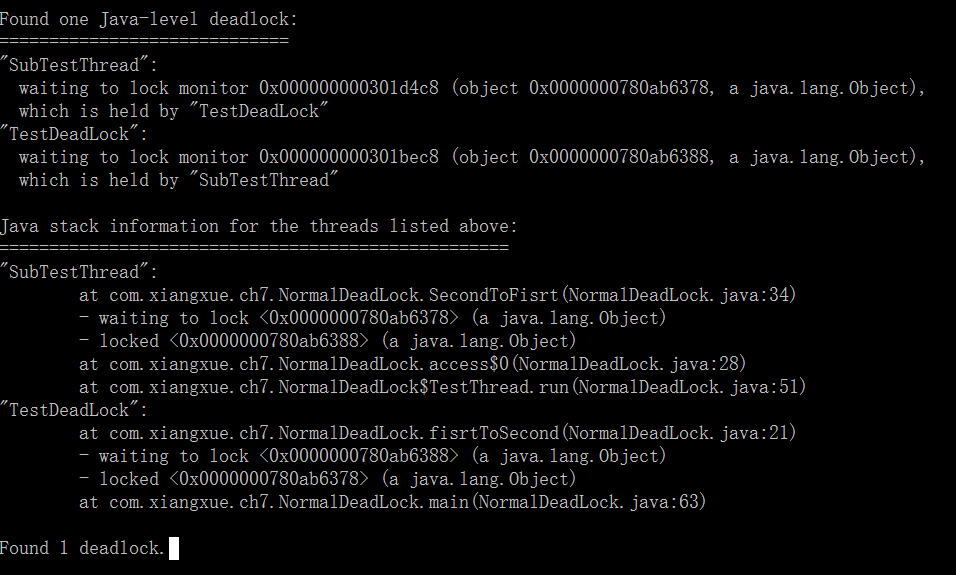
# 死锁

资源一定是多于1个，同时小于等于竞争的线程数，资源只有一个，只会产生激烈的竞争。

死锁的根本成因：**获取锁的顺序不一致导致。**

## 简单的





怀疑发送死锁：

通过jps 查询应用的 id，

再通过jstack id 查看应用的锁的持有情况

**解决办法：保证加锁的顺序性**

### 动态的

动态顺序死锁，在实现时按照某种顺序加锁了，但是因为外部调用的问题，导致无法保证加锁顺序而产生的。

解决：

1. 通过内在排序，保证加锁的顺序性
2. 通过尝试拿锁，也可以。

# 其他安全问题

## 活锁

尝试拿锁的机制中，发生多个线程之间互相谦让，不断发生拿锁，释放锁的过程。

解决办法：每个线程休眠随机数，错开拿锁的时间。

## 线程饥饿

# 性能和思考

## 影响性能的因素

### 上下文切换

## 内存同步

## 减少锁的竞争

### 缩小锁的范围

### 减少锁的粒度

### 锁分段

### 替换独占锁

# 线程安全的单例模式

## 双重检查锁定

## 解决之道

### 懒汉式

### 饿汉式