链滴

Java 并发编程基础概念

作者：DongXiaokai0819

原文链接：https://ld246.com/article/1570839966833

来源网站：链滴

Java并发编程基础概念

线程安全性

1、什么是线程安全性?

● 当多个线程访问某个状态变量并且其中有一个线程执行写入操作时，必须采用同步机制来协同这些程对变量的访问。

● 如果当多个线程访问同一个可变的状态变量时没有使用合适的同步，那么程序就会出现错误，这也是所谓的线程不安全。

怎么解决这些不安全的问题呢?

1. 不在线程之间共享该状态变量
2. 将状态变量修改为不可变的变量
3. 在访问状态变量时没有使用同步

当多个线程对同一共享变量进行操作时就有可能导致线程安全问题，那什么又是线程安全性呢?

当多个线程访问某个类时，不管运行时环境采用何种调度方式或者这些线程将如何交替执行，并且在调代码中不需要任何额外的同步或协同，这个类都能表现出正确的行为，那么就称这个类是线程安全。

2、竞态条件

如果多个线程对同一个数据进行“读取---修改---写入”的操作序列的话，如果对这个共享变量的操没有进行同步的话，在多次调用中可能会导致严重的数据完整性问题。

在这里引出竞态条件的定义:

由于不恰当的执行时序而出现不正确的结果，这种情况被称为竞态条件。

竞态条件 很容易会与 **数据竞争** 相混淆，下面给出数据竞争的定义。

数据竞争是指，如果在访问共享的非final类型的域时没有采取同步来进行协同，那么就会出现数据竞。

当某个计算的正确性取决于多个线程交替执行时序时，那么就会发生竞态条件。竞态条件又有哪些类呢?

● **先检查后执行** 基于一种可能失效的观察结果来做出判断或者执行某个计算

3、加锁机制

在介绍锁之前，先要介绍原子性相关的几个概念

● 原子操作：对于访问同一个状态的所有操作（包括该操作本身）来说，这个操作时一个以原子方式行的操作

● 复合操作：包含了一组必须以原子方式执行的操作以确保线程安全性

内置锁

同步代码块：java为支持原子性的一种内置的锁机制

同步代码块包括俩个部分，一个作为锁的对象引用，一个作为由这个锁保护的代码块

```
synchronized(lock){
//访问或修改由锁保护的共享状态
}
```