知识点

第二章 线程安全性

2.1 线程安全

1.*无状态对象一定是线程安全的。*线程访问无状态对象的行为不会影响其他线程操作的正确性，例如大多数servlet就是无状态的，所以servlet一般就是线程安全的，只有处理请求时servlet需要保存信息，才回引发线程安全问题。

2.2 原子性

1．原子性。确保一系列的操作是原子操作，才能保证线程安全。例如对于计数类操作，多使用AtomicLong等原子变量，就可以保证线程安全

2.3 加锁机制

1. java内置锁：同步代码块（Synchronized Block）包括两部分：一个是锁的对象引用，一个是锁保护的代码块。用synchronized修饰的方法就是锁保护起来的代码块，同步代码块的锁就是方法调用的对象。静态synchronized方法以class对象作为锁。

2.重入：一个线程试图获取自己持有的锁的时候，会发生重入，所以请求会成功，不会死锁。

2.5 活跃性与性能

当执行时间较长的计算或者无法完成快速操作的时候，例如网络io或者控制台io，一定不要持有锁。

第三章 对象的共享

3.1 可见性

1. java内存模型要求，变量的读取和写入都必须是原子操作。对于非volatile修饰的double和long类型，jvm将64位的读写分成两个32位的原子操作，所以是线程不安全的，除非volatile声明，或者锁保护起来。

加锁机制既可以保证可见性又可以保证原子性，volatile只能保证可见性

2.volatile使用条件：

1.对变量的写入操作不依赖于变量的当前值

2.该变量不会与其他变量纳入不变形条件中

3.在访问变量时不需要加锁

3.2 发布和逸出

发布：让对象可以在当前作用域之外的代码中访问到。例如将对象的引用保存到其他代码可以访问的地方，或非私有方法中返回该引用，或将引用传递到其他类

逸出：某个不应该发布的类被发布了。

注意：不要在构造函数中使this引用逸出

3.3 线程封闭

1.维持线程封闭的更规范方法是ThreadLocal。

3.4 不变性

1.不可变对象一定是线程安全的。不可变对象需要满足的条件：

1.对象创建以后就不能再修改

2.对象的所有域都是final类型

3.对象是正确创建的（在对象的创建期间，this引用没有逸出）