synchronized 用来给对象获得对象锁，当不同的对象时，对象锁也是不一样的，所以此时能够保证两个线程是互斥的，不会影响，从而也不会导致堵塞的情况。

当synchronized用来修饰静态方法或者类时，将会使得这个类的所有对象都是共享一把类锁，导致线程阻塞，所以这种写法一定要规避

* 无论synchronized关键字加在方法上还是对象上，如果它作用的对象是非静态的，则它取得的锁是对象；如果synchronized作用的对象是一个静态方法或一个类，则它取得的锁是对类，该类所有的对象同一把锁。
* 每个对象只有一个锁（lock）与之相关联，谁拿到这个锁谁就可以运行它所控制的那段代码。
* 实现同步是要很大的系统开销作为代价的，甚至可能造成死锁，所以尽量避免无谓的同步控制。

synchronized 关键字表示了 Java 的一个实现同步的锁，线程进入同步代码块，必须要先获得锁，退出时释放锁。这个锁是互斥的，同一时间只有一个线程能获得锁。获得不到锁的线程，只能等待别的线程释放锁，然后再去抢这个锁。

修饰代码块

这是最常用的方式，获得 lock 对象的线程，才能继续运行。

Object lock = new Object();

public void foo() {

synchronized (lock) {

//do things

}

}

修饰方法

public synchronized void foo()

{

// do things

}

这个等价于下面的代码，获取的锁对象就是当前类的实例。

public void foo()

{

synchronized(this) {

// do things

}

}

修饰静态的方法

public synchronized static void foo() {

// do things

}

这个等价于下面的代码，获取的锁对象就是当前类的 class 对象。

class myclass {

public void foo() {

synchronized(myclass.class) {

// do things

}

}

1.synchronized关键字不能继承。

虽然可以使用synchronized来定义方法，但synchronized并不属于方法定义的一部分，因此，synchronized关键字不能被继承。如果在父类中的某个方法使用了synchronized关键字，而在子类中覆盖了这个方法，在子类中的这个方法默认情况下并不是同步的，而必须显式地在子类的这个方法中加上synchronized关键字才可以。当然，还可以在子类方法中调用父类中相应的方法，这样虽然子类中的方法不是同步的，但子类调用了父类的同步方法，因此，子类的方法也就相当于同步了。

2.无论synchronized关键字加在方法上还是对象上，如果它作用的对象是非静态的，则它取得的锁是对象；如果synchronized作用的对象是一个静态方法或一个类，则它取得的锁是对类，该类所有的对象同一把锁。

3.每个对象只有一个锁（lock）与之相关联，谁拿到这个锁谁就可以运行它所控制的那段代码。

4.实现同步是要很大的系统开销作为代价的，甚至可能造成死锁，所以尽量避免无谓的同步控制。

|  |  |  |
| --- | --- | --- |
| 类别 | synchronized | lock |
| 存在层次 | java的关键字，在jvm层面上 | 是一个类 |
| 锁的释放 | 1、以获取锁的线程执行完同步代码，释放锁  2、线程执行发生异常，jvm会让线程释放锁 | 在finally中必须释放锁，不然容易造成线程死锁 |
| 锁的获取 | 假设A线程获得锁，B线程等待，  如果A线程阻塞，B线程会一直等待 | 分情况而定，lock有多个锁获取的方法，可以尝试获得锁，  线程可以不用功一直等待 |
| 锁状态 | 无法判断 | 可以判断 |
| 锁类型 | 可以重入，不可以中断，非公平 | 可重入 可以判断 可公平 |
| 性能 | 少量同步 | 大量同步 |