关于synchronized重复造轮子的问题

笔记本:	我的第一个笔记本		
创建时间:	2020/8/13 10:50	**更新时间:**	2020/8/13 11:03
作者:	1667567382@qq.com		
URL:	file:///D:/BaiduNetdiskDownload/jeek/24%20Java%E5%B9%B6%E5%8F%91...		

关于synchronized重复造轮子的问题

既然synchronized底层就是利用管程来实现的，那么为什么又开发了Lock接口呢?

对于这个问题，肯定的是这两个工具是不一样的，不然这两个工具重复造轮子了，那么这样看来的话，是这样来理解的。
从synchronized的缺陷来看，synchronized锁是个重量级锁，获取不到这个锁的话，就会直接进入Blocked状态，进入阻塞状态，如果发生死锁，就没有机会唤醒阻塞的线程。具体的synchronized细节还是很复杂的，什么entrylist的，以后来理解。
那么如果可以给这个锁加一个功能，让其可以破坏掉不可抢占这个条件。

1. 可以被打断，响应interrupt
2. 给一个时间，设置一个响应时间，如果超时就释放掉已经获取的锁。
3. 如果尝试失败就返回，就可以释放持有的锁。
   这就是Lock的三个api

```
// 支持中断的 API
void lockInterruptibly() throws InterruptedException;
// 支持超时的API
boolean tryLock(long time, TimeUnit unit) throws InterruptedException;
// 支持非阻塞获取锁的 API
boolean tryLock()
```