1. 同步容器
2. 使用

austen-concurrency-> com.austen.syncContainer(Demo\*)

1. Queue
   1. Queue
   2. BrokingQueue

LinkedBlockingQueue

ArrayBlockingQueue

DelayQueue

SynchronousQueue

* 1. TransferQueue

LinkedTransferQueue

二、线程

1)interrupt();

参考：austen-concurrency->com.austen.threadExample.interrupt

--用来优雅停止线程的函数

--jvm不推荐直接停止一个线程，一定要让一个线程执行完

--假设现在在阻塞->解阻塞

--假设线程是while(flag)->flag=false

三、锁

1）synchronized

参考：austen-concurrency->com.austen.com.austen.lockExample.sync

I、使用（demo\*）

II、1.6之前，所有的synchronized都是重量级锁(Test1)

--偏向锁、轻量级锁、重量级锁

--证明方式。

1、修改linux源码-glibc；

java中使用public native void gettid()获取jvm的线程ID

gettid使用c语言编写

2、看对象头

--重量级锁：借助于OS函数来实现的锁。存在互斥场景中

jvm->lock()//C++

如：linux

pthread\_mutex\_t mt;//int

pthread\_mutex\_init(mt);

pthread\_mutex\_lock(mt);//1

pthread\_mutex\_unlock(mt);//0

--偏向锁：线程必须线程安全，但不一定有互斥。不会调用OS函数。存在没有资源竞争的情况下（Test2）

--轻量级锁：