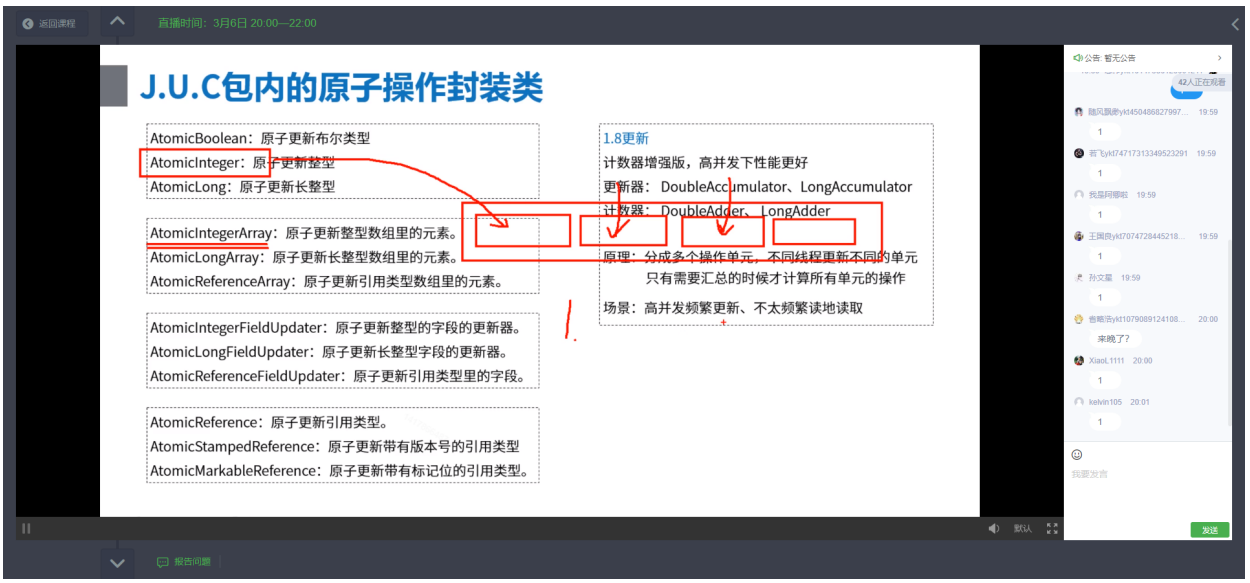
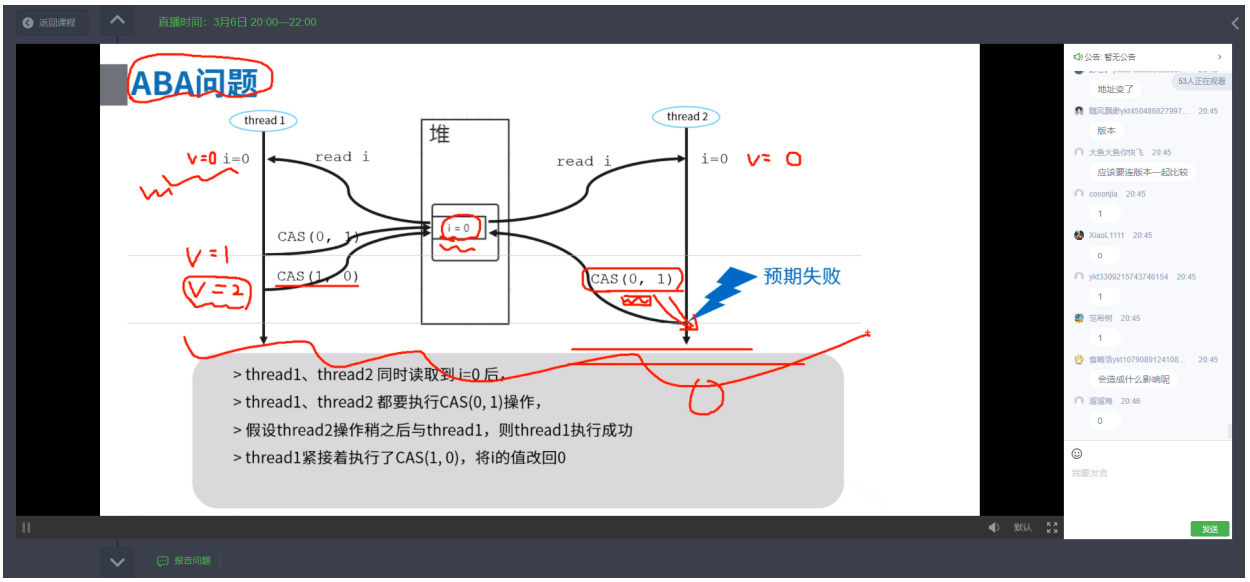
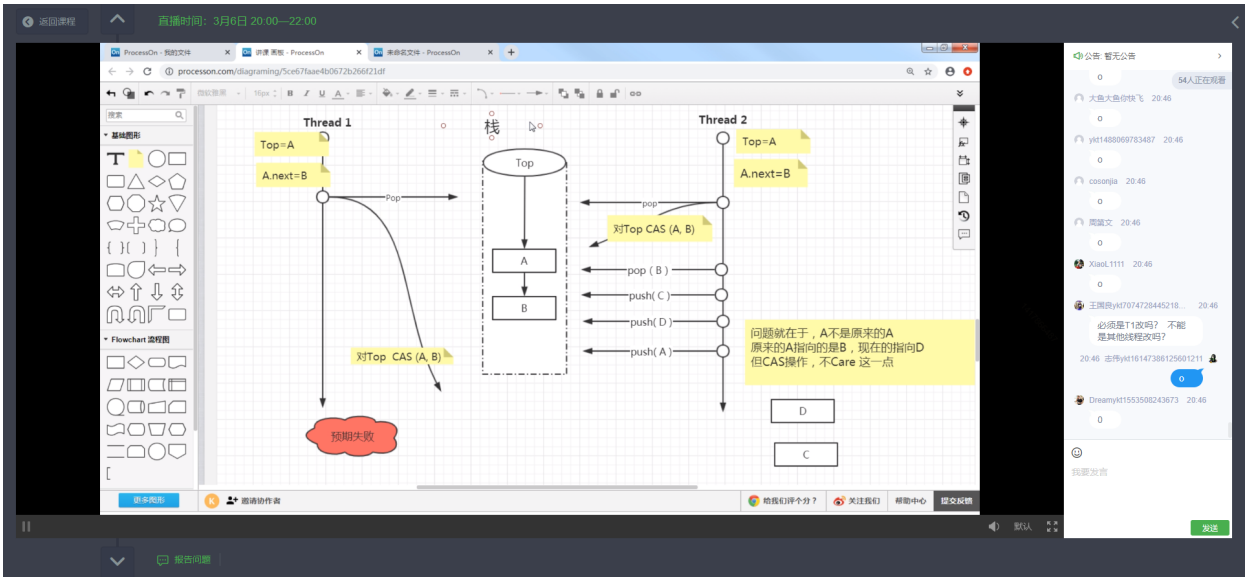

直播时间：3月6日 20:00—22:00
J.U.C包内的原子操作封装类
AtomicBoolean：原子更新布尔类型
AtomicInteger：原子更新整型
AtomicLong：原子更新长整型
AtomicIntegerArray：原子更新整型数组里的元素。
AtomicLongArray：原子更新长整型数组里的元素。
AtomicReferenceArray：原子更新引用类型数组里的元素。
AtomicIntegerFieldUpdater：原子更新整型的字段的更新器。
AtomicLongFieldUpdater：原子更新长整型字段的更新器。
AtomicReferenceFieldUpdater：原子更新引用类型里的字段。
AtomicReference：原子更新引用类型。
AtomicStampedReference：原子更新带有版本号的引用类型
AtomicMarkableReference：原子更新带有标记位的引用类型。
1.8更新
计数器增强版，高并发下性能更好
更新器： DoubleAccumulator、LongAccumulator
计数器： DoubleAdder、LongAdder
原理：分成多个操作单元，不同线程更新不同的单元
只有需要汇总的时候才计算所有单元的操作
场景：高并发频繁更新、不太频繁读地读取


直播时间：3月6日 20:00—22:00
CAS的三个问题
1. 循环+CAS，自旋的实现让所有线程都处于高频运行，争抢CPU执行时间的状态 。如果操作长时间不成功，会带来很大的CPU资源消耗。
2. 仅针对单个变量的操作，不能用于多个变量来实现原子操作。
3. ABA问题。


直播时间：3月6日 20:00—22:00
ABA问题
thread 1
thread 2
堆
read i
read i
i=0
CAS(0, 1)
CAS(1, 0)
CAS(0, 1)
预期失败
> thread1、thread2 同时读取到i=0 后，
> thread1、thread2 都要执行CAS(0, 1)操作，
> 假设thread2操作稍之后与thread1，则thread1执行成功
> thread1紧接着执行了CAS(1, 0)，将i的值改回0



Synchronized使用到原理


返回课程
直播时间：3月6日 20:00—22:00
思考一下？？
锁住 this 对象
synchronized(this){
i++;
}
加锁的状态如何记录？
公告: 暂无公告
52人正在观看
kelvin105 21:40
随风飘的yk450486827997... 21:40
yxt1693892089954540 9 21:40
yxt1488069783487 21:40
渥渥海 21:41
加了volatile 也能被粗化吗
渥渥海 21:41
不是不能被重排吗
yxt3309215743746154 21:43
有锁细化吗
B先u 21:43
我要发言
发送
流畅
报告问题