AtomicInteger源码解读

1. **AtomicInteger源码解读**

## 变量必须用volatile修饰：保证可见性

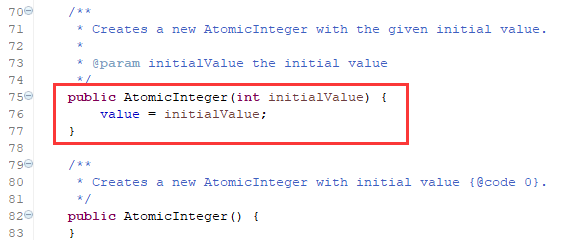
public class **AtomicInteger** extends **Number** implements java.io.Serializable

拿出**AtomicInteger**来研究在**没有锁的情况下**是如何做到数据正确性的。

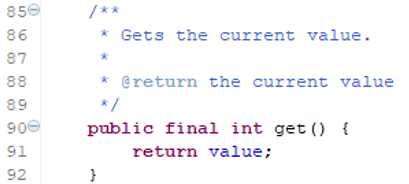
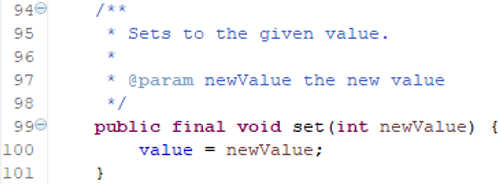
利用volatile修饰变量value：保证可见性，但是没有原子性。



## 构造方法：

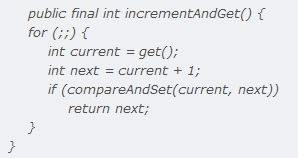


## get和set方法：

重点是++i是如何做到的：

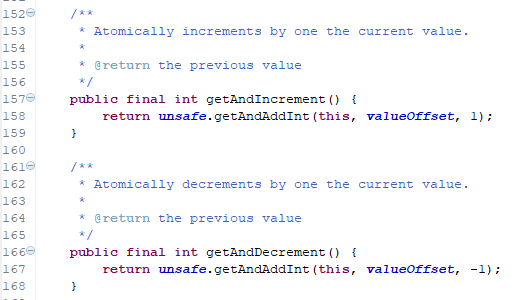
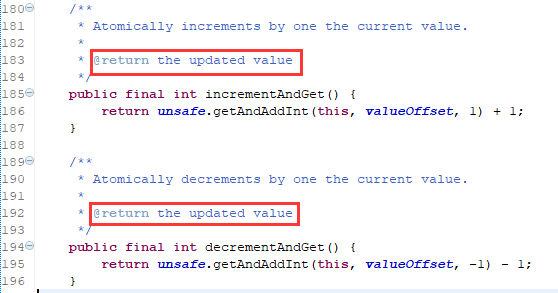
JDK1.5的源代码：



## getAndIncrement与incrementAndGet

**JDK1.8的源代码**：**getAndIncrement**返回原来的，**incrementAndGet**返回更新之后的值。

getAndIncrement与incrementAndGet、getAndDecrement与decrementAndGet。

利用了**compareAndSet**方法，这个方法就是**Native方法**。只有返回true，才返回值，否则一直循环。

在这里采用了CAS操作，每次从内存中读取数据然后将此数据和+1后的结果进行CAS操作，如果成功就返回结果，否则重试直到成功为止。

## compareAndSet

而compareAndSet利用JNI来完成CPU指令的操作。

public final boolean **compareAndSet**(int expect, int update) {

return **unsafe.compareAndSwapInt(this, valueOffset, expect, update);**

}

## sun.misc.Unsafe类

JDK1.5以后，CAS操作由1) sun.misc.Unsafe类的compareAndSwapInt()和compareAndSwapLong等方法包装提供。