链滴

volatile 的是与非

作者：eddy

原文链接：https://ld246.com/article/1485219733352

来源网站：链滴

volatile关键字所做的事情

被volatile修饰的属性，在修改属性的值的时候，会增加一个内存屏障，将包括该属性在内的前面所修改的属性刷新CPU缓存，做到属性的变化对其他线程可见。

volatile关键字的用处

保证属性的安全，即属性的任何变化都能够做到及时对其他线程可见.

volatile不等同于线程安全

由于一个线程会将临界变量的值压入线程栈，类似在线程栈中对临界属性做了备份，所以即使临界属时时保证为最新的值，但无法更改线程栈中的备份，所以也就无法做到线程安全。

下面来看一个具体示例，用来解释线程是如何备份变量的值。

java源码:

```
public class T2 {


    private Integer i = 1;

    public void test3() {
        i = i + 2;
    }
}
```

反汇编:

```
Compiled from "T2.java"
public class test.T2 {
  public test.T2();
    Code:
       0: aload_0
       1: invokespecial #1                  // Method java/lang/Object."<init>":()V
       4: aload_0
       5: iconst_1
       6: invokestatic  #2                  // Method java/lang/Integer.valueOf:(I)Ljava/lang/Integer;
       9: putfield      #3                  // Field i:Ljava/lang/Integer;
      12: return

  public void test3();
    Code:
       0: aload_0
       1: aload_0
       2: getfield      #3                  // Field i:Ljava/lang/Integer; -- 将属性i的引用压入栈顶
       5: invokevirtual #4                  // Method java/lang/Integer.intValue:()I --i的引用出栈并调用i
tValue()方法，并将结果压入栈顶
       8: iconst_2
       9: iadd
      10: invokestatic  #2                  // Method java/lang/Integer.valueOf:(I)Ljava/lang/Integer;
```

```
      13: putfield      #3                  // Field i:Ljava/lang/Integer;
      16: return
}
```

可以看出，在test3方法中, 虚拟机会拿到临界变量i的值，并且将i的值压入栈顶，后续计算操作使用是线程栈中的值进行操作。

结论

所以，volatile可以保证属性安全，但并非保证线程安全。

说明

博主是个水货，如果写的不对的地方欢迎大家指出来。

原文地址