

JAVA 单例双重检查(double check)为什么不好用

原创 2016年04月01日 17:28:06

标签 : java (<http://so.csdn.net/so/search/s.do?q=java&t=blog>) /

线程 (<http://so.csdn.net/so/search/s.do?q=线程&t=blog>) /

并发 (<http://so.csdn.net/so/search/s.do?q=并发&t=blog>)

1242

anjxue (<http://blog.csdn.net/anjxue>)

+ 关注

(<http://blog.csdn.net/anjxue>)

码云

1

原创

粉丝

喜欢

(<https://github.com/anjxue>)

5

6

1

utm\_source

JAVA 单例双重检查(double check)为什么不好用

在阅读之前，请先了解下线程并发涉及到的三个概念：原子性、可见性、有序性，可以看下这篇文章：<http://www.cnblogs.com/dolphin0520/p/3920373.html>  
(<http://www.cnblogs.com/dolphin0520/p/3920373.html>)

我假设你已经看过几篇double check的文章，但还是一知半解。

我们先看这种双重检查，不加volatile

```
1 public static Singleton instance;
2 public static Singleton getInstance()
3 {
4     if (instance == null) //1
5     { //2
6         synchronized(Singleton.class) { //3
7             if (instance == null) //4
8                 instance = new Singleton(); //5
9         }
10    }
11    return instance;
12 }
```

这种方式存在什么问题呢？

也许有人说存在可见性问题：线程1执行完第5步，释放锁。线程2获得锁后执行到第4步，由于可见性的原因，发现instance还是null，从而初始化了两次。

但是不会存在这种情况，因为synchronized能保证线程1在释放锁之前会讲对变量的修改刷新到主存当中，线程2拿到的值是最新的。

实际存在的问题是无序性。

第5步这个new操作是无序的，它可能会被编译成：

- a. 先分配内存，让instance指向这块内存

广告

他的最新文章  
更多文章 (<http://blog.csdn.net/anjxue>)

Android游戏开发中备忘录模式的应用 (<http://blog.csdn.net/anjxue/article/details/50668309>)

原汁原味设计模式-原型prototype (<http://blog.csdn.net/anjxue/article/details/50668292>)

一次非常诡异的WebView js失效问题 (<http://blog.csdn.net/anjxue/article/details/50015847>)

Android/Java回调机制带来的内存回收问题 (<http://blog.csdn.net/anjxue/article/details/49358813>)

- 文章分类
- JVM (<http://blog.csdn.net/anjxue/article/details/51038466>)

1篇
- Android (<http://blog.csdn.net/anjxue/article/details/50668309>)

1篇
- WebView (<http://blog.csdn.net/anjxue/article/details/50015847>)

1篇
- 原汁原味设计模式系列 (<http://blog.csdn.net/anjxue/article/details/50668292>)

1篇
- 设计模式 (<http://blog.csdn.net/anjxue/article/details/49358813>)

1篇
- 展开

文章存档

- b. 在内存中创建对象

然而我们需要意识到这么一个问题，synchronized虽然是互斥的，但不代表一次就把整个过程执行完，它在中间是可能释放时间片的，时间片不是锁。（我因为这里没转过来，耽误了很多时间）  
也就是说可能在a执行完后，时间片被释放，线程2执行到1，这时他读到的instance是不是null呢？（标记1）  
基于可见性，可能是null，也可能不是null。  
**非常奇葩的是**，在这个例子中，如果读到的是null，反而没问题了，接下来会等待锁，然后再次判断时不为null，最后返回单例。  
如果读到的不是null，那么坏了，按逻辑它就直接return instance了，这个instance还没执行构造参数，去做事情的话，很可能就崩溃了。

加volatile

```
1 public volatile static Singleton instance;
2 public static Singleton getInstance()
3 {
4     if (instance == null)           //1
5     {                               //2
6         synchronized(Singleton.class) { //3
7             if (instance == null)     //4
8                 instance = new Singleton(); //5
9         }
10    }
11    return instance;
12 }
```

唯一的区别是加了volatile关键字，那么会有什么现象？  
这时要区分jdk版本了，在jdk1.4及之前，volatile并不能保证new操作的有序性，但是它能保证可见性，因此标记1处，读到的不是null，导致了问题。  
从1.5开始，加了volatile关键字的引用，它的初始化就不能是：  
- a. 先分配内存，让instance指向这块内存  
- b. 在内存中创建对象

而应该是：  
- a.在内存中创建对象  
- b.让instance指向这个对象.

这种形式，也就避免了无序性问题。

2016年4月 (http://blog.csdn....	1篇
2016年2月 (http://blog.csdn....	2篇
2015年11月 (http://blog.csdn...	1篇
2015年10月 (http://blog.csdn...	1篇

他的热门文章

- 一次非常诡异的WebView js失效问题 (http://blog.csdn.net/anjxue/article/details/50015847) 1357
- JAVA 单例双重检查(double check)为什么不好用 (http://blog.csdn.net/anjxue/article/details/51038466) 1233
- Android/Java回调机制带来的内存回收问题 (http://blog.csdn.net/anjxue/article/details/49358813) 703
- Android游戏中开发中备忘录模式的应用 (http://blog.csdn.net/anjxue/article/details/50668309) 370
- 原汁原味设计模式-原型prototype (http://blog.csdn.net/anjxue/article/details/50668292) 224

联系我们

网站客服 微博客服  
(http://wpa.qq.com/msgrd?v=3&uin=2431299880&site=qq&rv=34910153) (http://e.weibo.com/csdnsupport/pc)  
webmaster@csdn.net (mailto:webmaster@csdn.net)  
400-660-0108

京ICP证09002463号  
(http://www.miibeian.gov.cn/)  
关于 (http://www.csdn.net/company/about.html)  
招聘 (http://www.csdn.net/company/recruitment.html)  
广告服务 (http://www.csdn.net/company/marketingservice.html)  
阿里云  
Copyright © 1999-2018  
CSDN.NET, All Rights Reserved

- surpassno (/surpassno) 2017-12-12 11:20 2楼  
(/surpassno) 分析得很对！！  
回复
- surpassno (/surpassno) 2017-12-11 19:31 1楼