Java单例对象同步问题探讨 PDF转换可能丢失图片或格式，建议阅读原文
https://www.100test.com/kao_ti2020/244/2021_2022_Java_E5_8D_95_E4_BE_8B_c104_244686.htm 单例对象（Singleton）是一种常用的设计模式。在Java应用中，单例对象能保证在一个JVM中，该对象只有一个实例存在。正是由于这个特点，单例对象通常作为程序中的存放配置信息的载体，因为它能保证其他对象读到一致的信息。例如在某个服务器程序中，该服务器的配置信息可能存放在数据库或文件中，这些配置数据由某个单例对象统一读取，服务进程中的其他对象如果要获取这些配置信息，只需访问该单例对象即可。这种方式极大地简化了在复杂环境下，尤其是多线程环境下的配置管理，但是随着应用场景的不同，也可能带来一些同步问题。 本文将探讨一下在多线程环境下，使用单例对象作配置信息管理时可能会带来的几个同步问题，并针对每个问题给出可选的解决办法。 问题描述 在多线程环境下，单例对象的同步问题主要体现在两个方面，单例对象的初始化和单例对象的属性更新。 本文描述的方法有如下假设： 1. 单例对象的属性（或成员变量）的获取，是通过单例对象的初始化实现的。也就是说，在单例对象初始化时，会从文件或数据库中读取最新的配置信息。 2. 其他对象不能直接改变单例对象的属性，单例对象属性的变化来源于配置文件或配置数据库数据的变化。 1.1 单例对象的初始化 首先，讨论一下单例对象的初始化同步。单例模式的通常处理方式是，在对象中有一个静态成员变量，其类型就是单例类型本身；如果该变量为null，则创建该单例类型的对象，并将该变量指向这个对象；如果该变量不

为null，则直接使用该变量。 其过程如下面代码所示：public class GlobalConfig { private static GlobalConfig instance = null. private Vector properties = null. private GlobalConfig() { //Load configuration information from DB or file //Set values for properties } public static GlobalConfig getInstance() { if (instance == null) { instance = new GlobalConfig(). } return instance. } public Vector getProperties() { return properties. } } 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com