JAVA语言内存泄漏问题 PDF转换可能丢失图片或格式，建议阅读原文
https://www.100test.com/kao_ti2020/278/2021_2022_JAVA_E8_AF_AD_E8_A8_80_c104_278561.htm 1 Java的一个重要优点就是通过垃圾收集器GC （Garbage Collection）自动管理内存的回收，程序员不需要通过调用函数来释放内存。因此，很多程序员认为Java 不存在内存泄漏问题，或者认为即使有内存泄漏也不是程序的责任，而是GC 或JVM的问题。其实，这种想法是不正确的，因为Java 也存在内存泄漏，但它的表现与C 不同。如果正在开发的Java 代码要全天24 小时在服务器上运行，则内存漏洞在此处的影响就比在配置实用程序中的影响要大得多，即使最小的漏洞也会导致JVM耗尽全部可用内存。另外，在很多嵌入式系统中，内存的总量非常有限。在相反的情况下，即便程序的生存期较短，如果存在分配大量临时对象（或者若干吞噬大量内存的对象）的任何Java 代码，而且当不再需要这些对象时也没有取消对它们的引用，则仍然可能达到内存极限。 2 Java 内存回收机制 Java 的内存管理就是对象的分配和释放问题。分配内存的方式多种多样，取决于该种语言的语法结构。但不论是哪一种语言的内存分配方式，最后都要返回所分配的内存块的起始地址，即返回一个指针到内存块的首地址。在Java 中所有对象都是在堆（Heap）中分配的，对象的创建通常都是采用new或者是反射的方式，但对象释放却有直接的手段，所以对象的回收都是由Java虚拟机通过垃圾收集器去完成的。这种收支两条线的方法确实简化了程序员的工作，但同时也加重了JVM的工作，这也是Java 程序运行速度较慢的原因之一。因为，GC 为了

能够正确释放对象，GC 必须监控每一个对象的运行状态，包括对象的申请、引用、被引用、赋值等，GC 都需要进行监控。监视对象状态是为了更加准确地、及时地释放对象，而释放对象的根本原则就是该对象不再被引用。Java 使用有向图的方式进行内存管理，可以消除引用循环的问题，例如有三个对象，相互引用，只要它们和根进程不可达，那么GC 也是可以回收它们的。在Java 语言中，判断一块内存空间是否符合垃圾收集器收集标准的标准只有两个：一个是给对象赋予了空值null，以下再没有调用过，另一个是给对象赋予了新值，即重新分配了内存空间。 3 Java 中的内存泄漏 3.1 Java 中内存泄漏与C 的区别 在Java 中，内存泄漏就是存在一些被分配的对象，这些对象有下面两个特点，首先，这些对象是可达的，即在有向图中，存在通路可以与其相连；其次，这些对象是无用的，即程序以后不会再使用这些对象。如果对象满足这两个条件，这些对象就可以判定为Java 中的内存泄漏，这些对象不会被GC 所回收，然而它却占用内存。在C 中，内存泄漏的范围更大一些。有些对象被分配了内存空间，然后却不可达，由于C 中没有GC，这些内存将永远收 不回来。在Java 中，这些不可达的对象都由GC 负责回收，因此程序员不需要考虑这部分的内存泄漏。通过分析，可以得知，对于C ，程序员需要自己管理边和顶点，而对于Java 程序员只需要管理边就可以了（不需要管理顶点的释放）。通过这种方式，Java 提高了编程的效率。 3.2 内存泄漏示例 3.2.1 示例1 在这个例子中，循环申请Object 对象，并将所申请的对象放入一个Vector 中，如果仅仅释放引用本身，那么Vector 仍然引用该对象，所以这个对象对GC 来说是不可回收的。因此，

如果对象加入到Vector 后，还必须从Vector 中删除，最简单的方法就是将Vector对象设置为null。 Vector v = new Vector(10). for (int i = 1. i{Object o = new Object(). v.add(o). o = null. }// 此时，所有的Object 对象都没有被释放，因为变量v 引用这些对象。实际上无用，而还被引用的对象，GC 就无能为力了（事实上GC 认为它还有用），这一点是导致内存泄漏最重要的原因。 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