EJB的最佳实践：工业强度的JNDI优化 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/252/2021_2022_EJB_E7_9A_84_E6_9C_80_E4_c104_252537.htm 在这篇技巧文章中，我们将研究一些最常用的 JNDI 优化。特别地，我们将向您展示如何将高速缓存和通用助手类组合使用，以创建针对 JNDI 开销的工厂风格的解决方案。 减少上下文实例 清单 1 显示了一段典型的 EJB 代码，它需要多次 JNDI 查找。请花一点时间研究代码，然后我们将对它进行优化以获得更佳性能。 清单 1. 典型的 EJB 查找 public boolean buyItems(PaymentInfo paymentInfo, String storeName, List items) { // Load up the initial context Context ctx = new InitialContext(). // Look up a bean ' s home interface Object obj = ctx.lookup("java:comp/env/ejb/PurchaseHome"). PurchaseHome purchaseHome = (PurchaseHome)PortableRemoteObject.narrow(obj, PurchaseHome.class). Purchase purchase = purchaseHome.create(paymentInfo). // Work on the bean for (Iterator i = items.iterator(). i.hasNext(). ) { purchase.addItem((Item)i.next()). } // Look up another bean Object obj = ctx.lookup("java:comp/env/ejb/InventoryHome"). InventoryHome inventoryHome = (InventoryHome)PortableRemoteObject.narrow(obj, InventoryHome.class). Inventory inventory = inventoryHome.findByStoreName(storeName). // Work on the bean for (Iterator i = items.iterator(). i.hasNext(). ) inventory.markAsSold((Item)i.next()). } // Do some other stuff } 尽

管这个示例多少有点刻意，但它确实揭示了使用 JNDI 时的一些最明显的问题。对于初学者，您应该问问自己，新建 InitialContext 对象是否必需。很可能在应用程序代码的其它地方已经装入了这个上下文，而我们又在这里创建了一个新的。高速缓存 InitialContext 实例会立即促使性能提高，如清单 2 所示： 清单 2. 高速缓存 InitialContext 实例 public static Context getInitialContext() { if (initialContext == null) { initialContext = new InitialContext(). } return initialContext. } 通过对 getInitialContext() 使用助手类，而不是为每个操作都实例化一个新的 InitialContext，我们将遍布在应用程序中的上下文数量减少为一个。 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