JAVA异常谜题43:异常地危险(1) PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/138/2021_2022_JAVA_E5_BC_82_E5_B8_B8_c97_138500.htm 在JDK1.2中，Thread.stop、Thread.suspend以及其他许多线程相关的方法都因为它们不安全而不推荐使用了[ThreadStop]。下面的方法展示了你用Thread.stop可以实现的可怕事情之一。 // Don’t do this - circumvents exception checking! public static void sneakyThrow(Throwable t) { Thread.currentThread().stop(t). // Deprecated!! } 这个讨厌的小方法所做的事情正是throw语句要做的事情，但是它绕过了编译器的所有异常检查操作。你可以（卑鄙地）在你的代码的任意一点上抛出任何受检查的或不受检查的异常，而编译器对此连眉头都不会皱一下。 不使用任何不推荐的方法，你也可以编写出在功能上等价于sneakyThrow的方法。事实上，至少有两种方式可以这么实现这一点，其中一种只能在5.0或更新的版本中运行。你能够编写出这样的方法吗？它必须是用Java而不是用JVM字节码编写的，你不能在其客户对它编译完之后再去修改它。你的方法不必是完美无瑕的：如果它不能抛出一两个Exception的子类，也是可以接受的。 本谜题的一种解决之道是利用Class.newInstance方法中的设计缺陷，该方法通过反射来对一个类进行实例化。引用有关该方法的文档中的话[Java-API]：“ 请注意，该方法将传播从空的[换句话说，就是无参数的]构造器所抛出的任何异常，包括受检查的异常。使用这个方法可以有效地绕开在其他情况下都会执行的编译期异常检查。 ”一旦你了解了这一点，编写一个sneakyThrow的等价方

法就不是太难了。 public class Thrower { private static Throwable t. private Thrower() throws Throwable { throw t. } public static synchronized void sneakyThrow(Throwable t) { Thrower.t = t. try { Thrower.class.newInstance(). } catch (InstantiationException e) { throw new IllegalArgumentException(). } catch (IllegalAccessException e) { throw new IllegalArgumentException(). } finally { Thrower.t = null. // Avoid memory leak } } } 在这个解决方案中将会发生许多微妙的事情。我们想要在构造器执行期间所抛出的异常不能作为一个参数传递给该构造器，因为Class.newInstance调用的是一个类的无参数构造器。因此，sneakyThrow方法将这个异常藏匿于一个静态变量中。为了使该方法是线程安全的，它必须被同步，这使得对其的并发调用将顺序地使用静态域t。 要注意的是，t这个域在从finally语句块中出来时是被赋为空的：这只是因为该方法虽然是卑鄙的，但这并不意味着它还应该是内存泄漏的。如果这个域不是被赋为空出来的，那么它阻止该异常被垃圾回收。最后，请注意，如果你让该方法抛出一个InstantiationException或是一个IllegalAccessException异常，它将以抛出一个IllegalArgumentException而失败。这是这项技术的一个内在限制。 Class.newInstance的文档继续描述道 “ Constructor.newInstance方法通过将构造器抛出的任何异常都包装在一个（受检查的）InvocationTargetException异常中而避免了这个问题。 ”很明显，Class.newInstance应该是做了相同的处理，但是纠正这个缺陷已经为时过晚，因为这么做将引入源代码级别的不兼容性，这将使许多依赖于Class.newInstance的程序崩溃。而弃用这个方法也不切实际

，因为它太常用了。当你在使用它时，一定要意识到Class.newInstance可以抛出它没有声明过的受检查异常。 被添加到5.0版本中的“ 通用类型（generics） ”可以为本谜题提供一个完全不同的解决方案。为了实现最大的兼容性，通用类型是通过类型擦除（type erasure）来实现的：通用类型信息是在编译期而非运行期检查的[JLS 4.7]。 下面的解决方案就利用了这项技术： 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