JDK1.5编译中的一个奇怪问题 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/252/2021_2022_JDK15_E7_BC_96_E8_c104_252484.htm 学员冯伟立今天中午问了我一个有趣的Java编译问题，我也无法给他解释，不知道有没有路过的高人能够解释清楚原因，望不吝赐教！ 下面程序的main方法中的第二行代码和注释中的两行代码表达的意思完全相同，注释中的两行代码不能通过编译（这很容易理解），而第二行（采用方法调用链）却可以顺利通过编译（这就很难理解了）。public class Test{ public void func() { System.out.println("func"). } public static void main(String args[]) throws Exception { Object obj = new Test(). //下面这行可以成功编译 ((Test)obj).getClass().newInstance().func(). //下面这两行无法通过编译 /*Class c = ((Test)obj).getClass(). c.newInstance().func(). */ }} 感谢paulex先生的帮助，在paulex先生的提示下，我基本上明白了上述问题的原因。下面是paulex先生的解答： 因为Generic, 编译器可以在编译期获得类型信息所以可以编译这类代码。你将下面那两行改成 Class c = ((Test)obj).getClass(). c.newInstance().func(). 应该就能通过编译了。 下面是我在paulex先生解答的基础上，对问题的进一步解释： 在JDK 1.5中引入范型后，Object.getClass()方法的定义如下：public final Class getClass()Returns the runtime class of an object. That Class object is the object that is locked by static synchronized methods of the represented class.Returns: The Java.lang.Class object that represents the runtime class of the object. The result is of type Class where X is the erasure of the static type of

the expression on which getClass is called. 这说明((Test)obj).getClass()语句返回的对象类型为Class，而Class的newInstance()方法的定义如下：public T newInstance() throws InstantiationException,IllegalACCESSException 即对于编译器看来，Class的newInstance()方法的对象类型为Test，而((Test)obj).getClass()返回的为对象类型为Class，所以，编译器认为((Test)obj).getClass().newInstance()返回的对象类型为Test。 下面这两行代码之所以无法通过编译Class c = ((Test)obj).getClass().c.newInstance().func(). 是因为((Test)obj).getClass()返回的为对象类型为Class，但是我们在第一行将结果强制转换成了Class，然后再去调用Class的newInstance方法，而不是去调用Class的newInstance方法，编译器当然不再认为Class的newInstance方法返回的对象为Test了。 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