JavaReflection(JAVA反射)详解 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/462/2021_2022_JavaReflec_c104_462417.htm Reflection是Java 程序开发语言的特征之一，它允许运行中的 Java 程序对自身进行检查，或者说"自审"，并能直接操作程序的内部属性。例如，使用它能获得 Java 类中各成员的名称并显示出来。 Java 的这一能力在实际应用中也许用得不是很多，但是在其它的程序设计语言中根本就不存在这一特性。例如，Pascal、C 或者 C 中就没有办法在程序中获得函数定义相关的信息。 JavaBean 是 reflection 的实际应用之一，它能让一些工具可视化的操作软件组件。这些工具通过 reflection 动态的载入并取得 Java 组件（类） 的属性。 1. 一个简单的例子 考虑下面这个简单的例子，让我们看看 reflection 是如何工作的。import java.lang.reflect.*.public class DumpMethods {public static void main(String args[]) {try {Class c = Class.forName(args[0]).Method m[] = c.getDeclaredMethods().for (int i = 0. i System.out.println(m[i].toString()).} catch (Throwable e) {System.err.println(e).}}} 按如下语句执行： java DumpMethods java.util.Stack 它的结果输出为：public java.lang.Object java.util.Stack.push(java.lang.Object)public synchronized java.lang.Object java.util.Stack.pop()public synchronized java.lang.Object java.util.Stack.peek()public boolean java.util.Stack.empty()public synchronized int java.util.Stack.search(java.lang.Object) 这样就列出了java.util.Stack 类的各方法名以及它们的限制符和返回类型。 这个程序使用 Class.forName 载入指定的类，然后调用 getDeclaredMethods 来

获取这个类中定义了的方法列表。java.lang.reflect.Methods 是用来描述某个类中单个方法的一个类。 2.开始使用 Reflection 用于 reflection 的类，如 Method，可以在 java.lang.relfect 包中找到。使用这些类的时候必须要遵循三个步骤：第一步是获得你想操作的类的 java.lang.Class 对象。在运行中的 Java 程序中，用 java.lang.Class 类来描述类和接口等。 下面就是获得一个 Class 对象的方法之一： Class c = Class.forName（"java.lang.String"）；这条语句得到一个 String 类的类对象。还有另一种方法，如下面的语句： Class c = int.class；或者 Class c = Integer.TYPE；它们可获得基本类型的类信息。其中后一种方法中访问的是基本类型的封装类（如 Integer）中预先定义好的 TYPE 字段。 第二步是调用诸如 getDeclaredMethods 的方法，以取得该类中定义的所有方法的列表。 一旦取得这个信息，就可以进行第三步了使用 reflection API 来操作这些信息，如下面这段代码： Class c = Class.forName（"java.lang.String"）； Method m[] = c.getDeclaredMethods（）；System.out.println（m[0].toString（））；它将以文本方式打印出 String 中定义的第一个方法的原型。 在下面的例子中，这三个步骤将为使用 reflection 处理特殊应用程序提供例证。 模拟 instanceof 操作符 得到类信息之后，通常下一个步骤就是解决关于 Class 对象的一些基本的问题。例如，Class.isInstance 方法可以用于模拟 instanceof 操作符：class A {}public class instance1 {public static void main(String args[]) {try {Class cls = Class.forName("A").boolean b1 = cls.isInstance(new Integer(37)).System.out.println(b1).boolean b2 = cls.isInstance(new A()).System.out.println(b2).} catch

(Throwable e) {System.err.println(e).}}} 在这个例子中创建了一个 A 类的 Class 对象，然后检查一些对象是否是 A 的实例。Integer（37） 不是，但 new A（）是。 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