Java认证辅导:Java反射机制深入研究Java认证考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/644/2021_2022_Java_E8_AE_A4_E8_AF_81_c104_644677.htm Java 反射是Java语言的一个很重要的特征，它使得Java具体了“ 动态性 ”。 在Java运行时环境中，对于任意一个类，能否知道这个类有哪些属性和方法？对于任意一个对象，能否调用它的任意一个方法？答案是肯定的。这种动态获取类的信息以及动态调用对象的方法的功能来自于Java 语言的反射（Reflection）机制。 Java 反射机制主要提供了以下功能： 在运行时判断任意一个对象所属的类。 在运行时构造任意一个类的对象。 在运行时判断任意一个类所具有的成员变量和方法。 在运行时调用任意一个对象的方法。 Reflection 是Java被视为动态（或准动态）语言的一个关键性质。这个机制允许程序在运行时透过Reflection APIs取得任何一个已知名称的class的内部信息，包括其modifiers（诸如public, static 等等）、superclass（例如Object）、实现之interfaces（例如Serializable），也包括fields和methods 的所有信息，并可于运行时改变fields内容或调用methods。 一般而言，开发者社群说到动态语言，大致认同的一个定义是：“ 程序运行时，允许改变程序结构或变量类型，这种语言称为动态语言 ”。从这个观点看，Perl，Python，Ruby是动态语言，C，Java，C#不是动态语言。 尽管在这样的定义与分类下Java不是动态语言，它却有着一个非常突出的动态相关机制：Reflection。这个字的意思是“ 反射、映象、倒影 ”，用在Java身上指的是我们可以于运行时加载、探知、使用编译期间完全未知的classes。换句话说，Java程序可以加载一个运行

时才得知名称的class，获悉其完整构造（但不包括methods定义），并生成其对象实体、或对其fields设值、或唤起其methods。这种“ 看透 class ”的能力（the ability of the program to examine itself）被称为introspection（内省、内观、反省）。Reflection和introspection是常被并提的两个术语。 在JDK中，主要由以下类来实现Java反射机制，这些类都位于java.lang.reflect包中： Class类：代表一个类。 Field 类：代表类的成员变量（成员变量也称为类的属性）。 Method类：代表类的方法。 Constructor 类：代表类的构造方法。 Array类：提供了动态创建数组，以及访问数组的元素的静态方法。 下面给出几个例子看看Reflection API的实际运用： 一、通过Class类获取成员变量、成员方法、接口、超类、构造方法等 在java.lang.Object 类中定义了getClass()方法，因此对于任意一个Java对象，都可以通过此方法获得对象的类型。Class类是Reflection API 中的核心类，它有以下方法 getName()：获得类的完整名字。 getFields()：获得类的public类型的属性。 getDeclaredFields()：获得类的所有属性。 getMethods()：获得类的public类型的方法。 getDeclaredMethods()：获得类的所有方法。 getMethod(String name, Class[] parameterTypes)：获得类的特定方法，name参数指定方法的名字，parameterTypes 参数指定方法的参数类型。 getConstructors()：获得类的public类型的构造方法。 getConstructor(Class[] parameterTypes)：获得类的特定构造方法，parameterTypes 参数指定构造方法的参数类型。 newInstance()：通过类的不带参数的构造方法创建这个类的一个对象。 下面给出一个综合运用的例子： public class RefConstructor { public static void main(String args[]) throws

Exception { RefConstructor ref = new RefConstructor(). ref.getConstructor(). } public void getConstructor() throws Exception { Class c = null. c = Class.forName("java.lang.Long"). Class cs[] = {java.lang.String.class}. System.out.println("\n----------\n"). Constructor cst1 = c.getConstructor(cs). System.out.println("1、通过参数获取指定Class对象的构造方法："). System.out.println(cst1.toString()). Constructor cst2 = c.getDeclaredConstructor(cs). System.out.println("2、通过参数获取指定Class对象所表示的类或接口的构造方法："). System.out.println(cst2.toString()). Constructor cst3 = c.getEnclosingConstructor(). System.out.println("3、获取本地或匿名类Constructor 对象，它表示基础类的立即封闭构造方法。"). if (cst3 != null) System.out.println(cst3.toString()). else System.out.println("-- 没有获取到任何构造方法！"). Constructor[] csts = c.getConstructors(). System.out.println("4、获取指定Class对象的所有构造方法："). for (int i = 0. i 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