为你的应用程序添加动态Java代码（三） PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/145/2021_2022__E4_B8_BA_E4_BD_A0_E7_9A_84_E5_c104_145074.htm 创造你自己的类装载器时注意parentLoader。基本上，父类装载器必须提供所有的子类装载器所需要的（依赖的）相关内容。在例子代码中，动态类PostmanImpl依赖接口Postman；这就是我们利用Postman的类装载器作父类装载器的原因。 我们离完成动态代码还差一步。回想早先介绍的例子。那里，动态类再装载对它的调用者是透明的。但在上述例子代码中，当代码改变时，我们仍必须改变从postman1到postman2的服务实例。第四步即最后一步将移除这种手动改变的需要。连接最新的类到它的调用者对于一个静态引用，我们如何访问最新的动态类呢？显然，一个对动态类的对象直接(通常都是这样)的引用是不会成功的。我们需要介于客户和动态类之间的东西代理。（关于代理模式请参阅著名的《设计模式》一书。）这里，代理是一个作为动态类的访问接口的功能类。客户不能直接调用动态类；要用代理代替。然后代理指向后端动态类的调用。图2显示了这种协作。Figure 2. Collaboration of the proxy当动态类重新装载时，我们仅需更新代理和动态类之间的链接即可，客户继续用同样的代理实例来访问被重新转载的类。图3显示了这种协作。Figure 3. Collaboration of the proxy with reloaded dynamic class这样，动态类的改变对它的调用者就是透明的了。Java反射API包括了一个创建代理的方便的工具。类java.lang.reflect.Proxy提供了静态的方法，让你为任何java接口创建代理实例。下面的例子为接口Postman创建了

一个代理。（如果你对java.lang.reflect.Proxy不熟悉，请在继续往下看之前看看javadoc。） InvocationHandler handler = new DynaCodeInvocationHandler(...). Postman proxy = (Postman) Proxy.newProxyInstance(Postman.class.getClassLoader(), new Class[] { Postman.class }, handler).返回的proxy是匿名类的一个对象，它与Postman接口共享了同一个类装载器（newProxyInstance()方法的第一个参数）并执行Postman接口（第二个参数）。Proxy实例的方法调用被分配给handler的invoke()方法（第三个参数）。Handler的执行程序可能看起来如下：public class DynaCodeInvocationHandler implements InvocationHandler { public Object invoke(Object proxy, Method method, Object[] args) throws Throwable { // Get an instance of the up-to-date dynamic class Object dynacode = getUpToDateInstance(). // Forward the invocation return method.invoke(dynacode, args). }}invoke()方法获得最新的动态类实例并且调用它。如果动态类的源码文件被修改了的话,这可能导致源代码的编辑和类的重新装载。现在我们有了完整的Postman服务的动态代码。客户创建一个服务的代理并通过该代理调用deliverMessage()方法。代理的每次调用被分配到DynaCodeInvocationHandler类的invoke()方法。在那个方法里，将会得到可能需要源码编译和类的重新装载的最新的服务实现，然后，调用服务实现进行实际的处理。把它们放到一起我们讲述了动态java代码需要的所有窍门。是时候把它们放到一起来建立一些可重用的东西了。我们可以通过建立一个工具,从而压缩以上四个步骤，使得采用动态代码变得更简单。 Postman例子会依赖于这个名叫DynaCode的工具。记

得PostmanApp应用程序和它的省略方法getPostman()吧？是时候展示它了：public class PostmanApp { public static void main(String[] args) throws Exception { // ..... } private static Postman getPostman() { DynaCode dynacode = new DynaCode(). dynacode.addSourceDir(new File("dynacode")). return (Postman) dynacode.newProxyInstance(Postman.class, "sample.PostmanImpl"). }} 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