反射机制是指在运行状态中，对于任何一个类来说，都能知道这个类的所有属性和方法；对于任何一个对象来说，都能调用它的任意一个方法和属性。这种动态获取的信息和动态调用对象的方法的功能叫做反射机制。

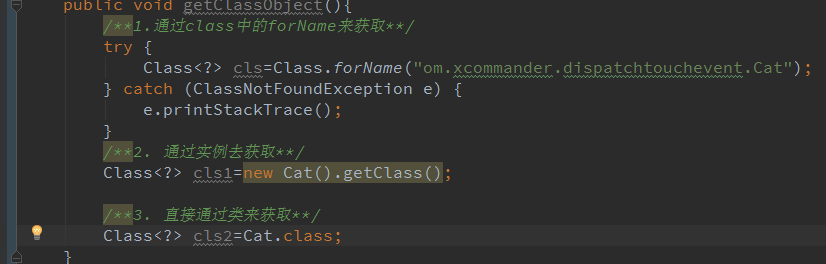
上面这个只不过表现形式而已。

在java中一切都是对象 ，可以分为实例对象和Class对象。对于实例对象，我们可以用通过new来获得，但是class对象我们获取不能通过new获得，因为它是jvm生成用来保存类的信息的。当我们定义好一个类的时候并且编译成.class字节码之后，编译器为我们创建了一个class对象并且保存在.class文件中。同时在jvm内部有个类的加载机制，在需要的时候（懒加载模式）将对用的.class文件和class对象加载到内存中来。

总结来说，A.java编译成A.class字节码之后，对应生成了一个class对象，只不过这个对象被保存在class文件中。当我们需要的时候，再加载到内存中。

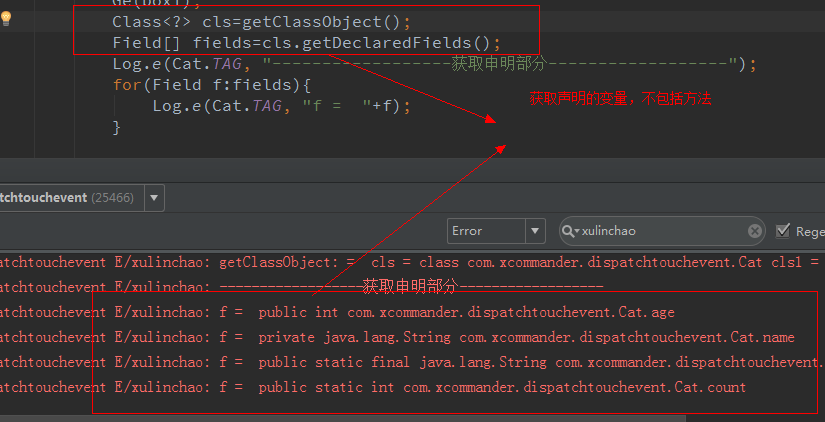
Java的反射机制，就是基于class对象来的，从class对象可以看到这个类以及实例的所有信息，就像一面镜子一样，所以叫反射。

所以说，要反射，必须获得class对象，而class对象在java文件编译成class字节码的时候，就产生了。所以现在就是如何把已经产生的class对象加载到内存中去了。

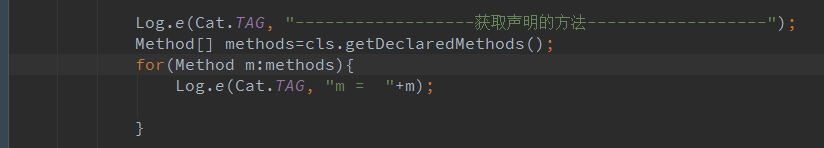


注意：一般情况下，我们按照第一种方式去获取class对象的。

（1）获取这个类声明的变量（都是子类有的，不包括父类）

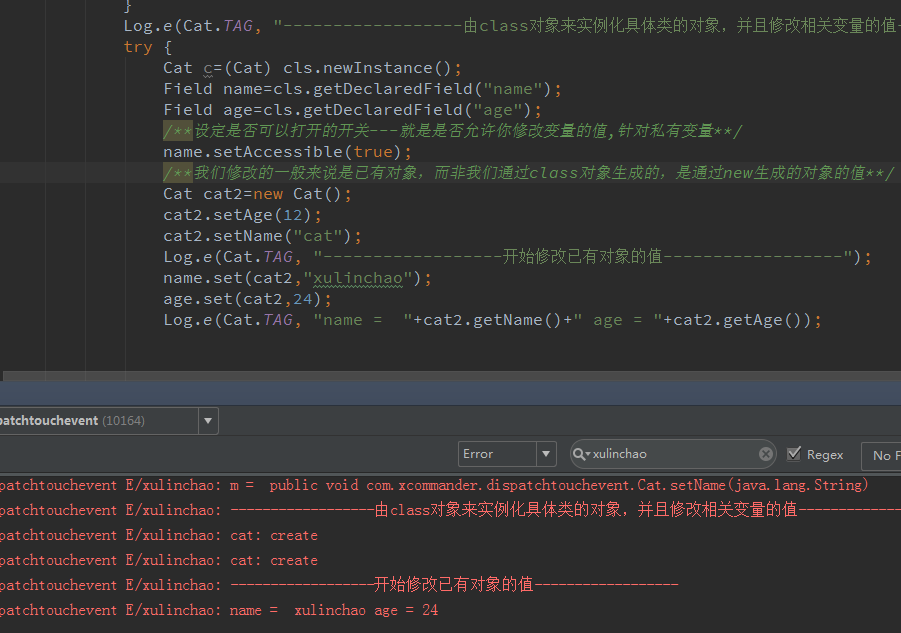


（2）获取方法

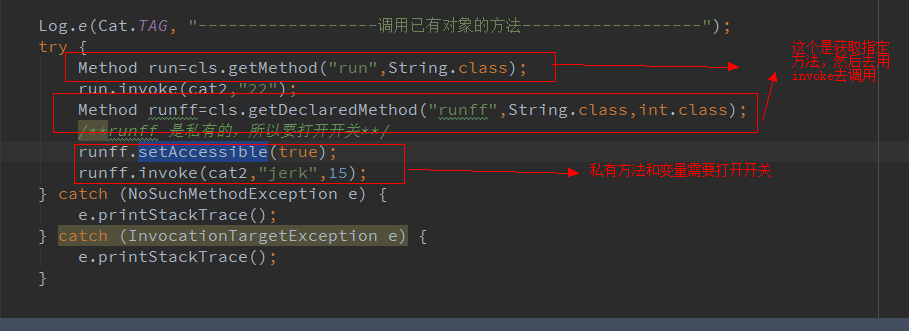


注意的是构造方法是没法获取的，只有实例化后，才可以获取。

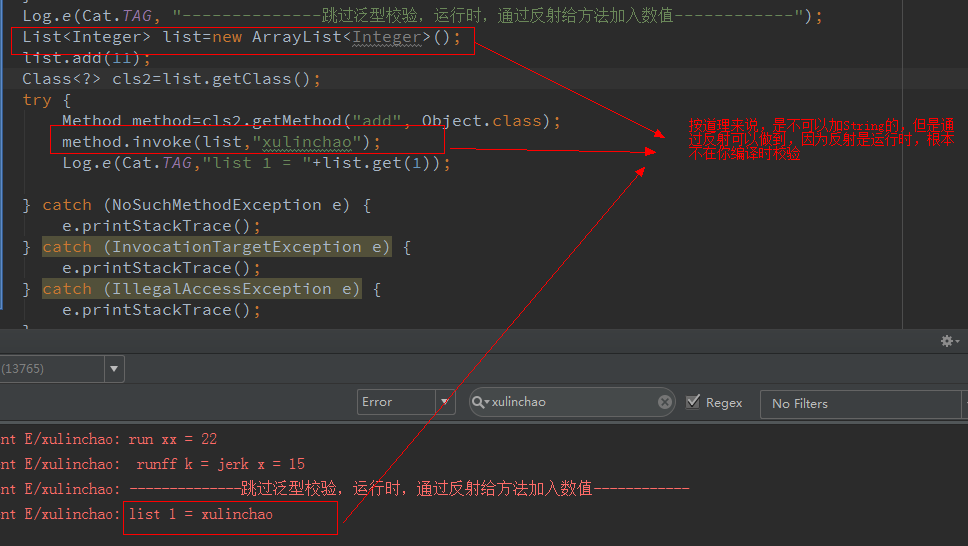
（3）修改已有对象的属性值以及从class对象中获取类的实例。



（4）调用某个对象的方法。



（5）前面不是说泛型，直接传递一个Integer类型的参数过去，就会变成Integer类型的list，然后存放String类型数据就会报错，因为在编译阶段会校验这个数据的，校验完成后，没报错，就认为数据正确，报错，则数据不正确。现在我们可以通过反射去打破这种作法。因为反射是运行时。编译校验没问题，我们可以在运行时，通过反射去加一个String数据，这样校验就会校验不到。



（6）反射修改数组信息。

