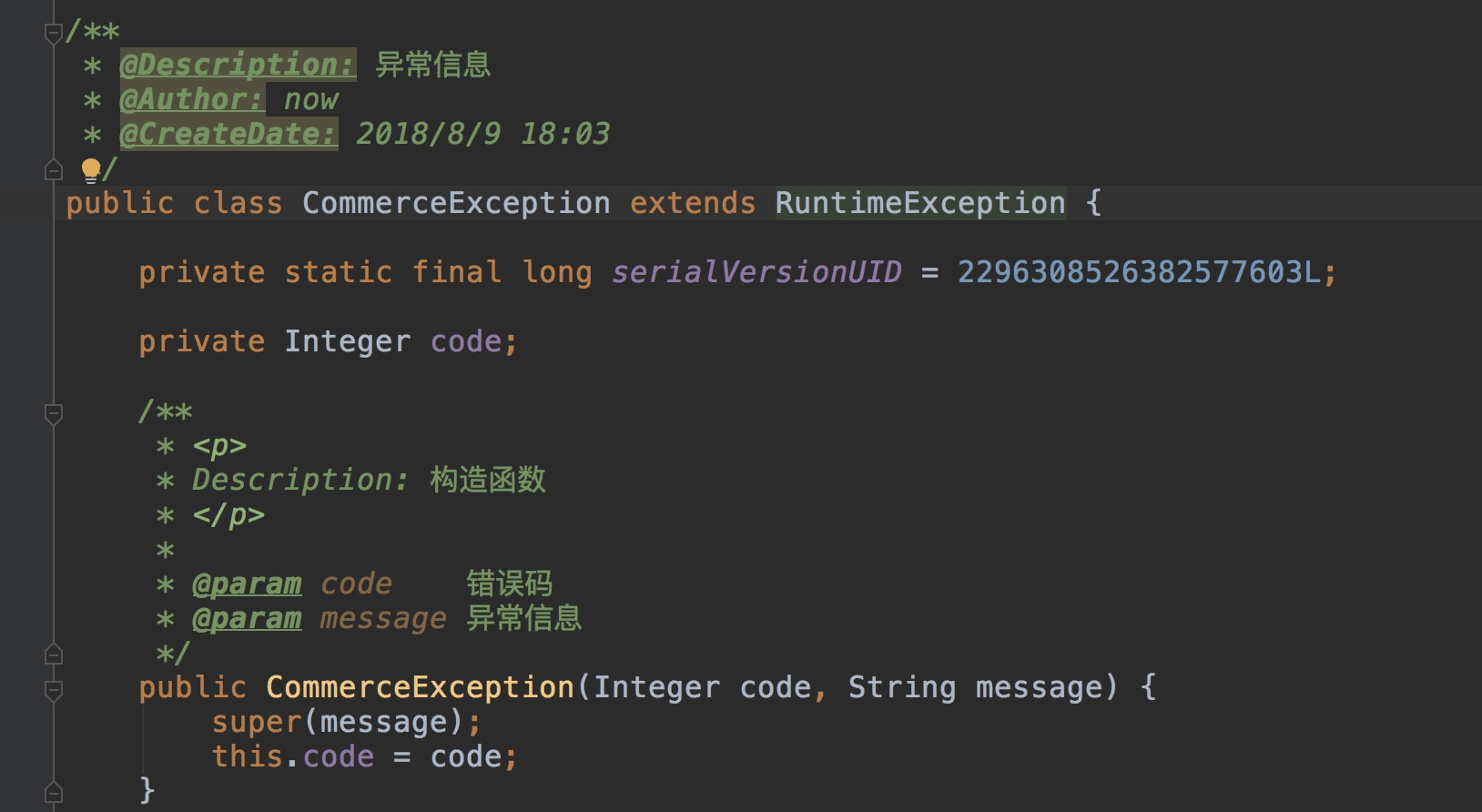
1.背景：这边变现进行服务化，建立多个独立的项目，对外提供dubbo和rest的接口；

服务的返回类型都是BaseResult和它的子类;需要在代码中组装返回值，时间点：2019年05月份

2.设计：自定异常类，添加属性错误码code字段；在接口需要返回错误时，用自定义异常抛出；同时在filter层，进行封装，我们想要的返回值；

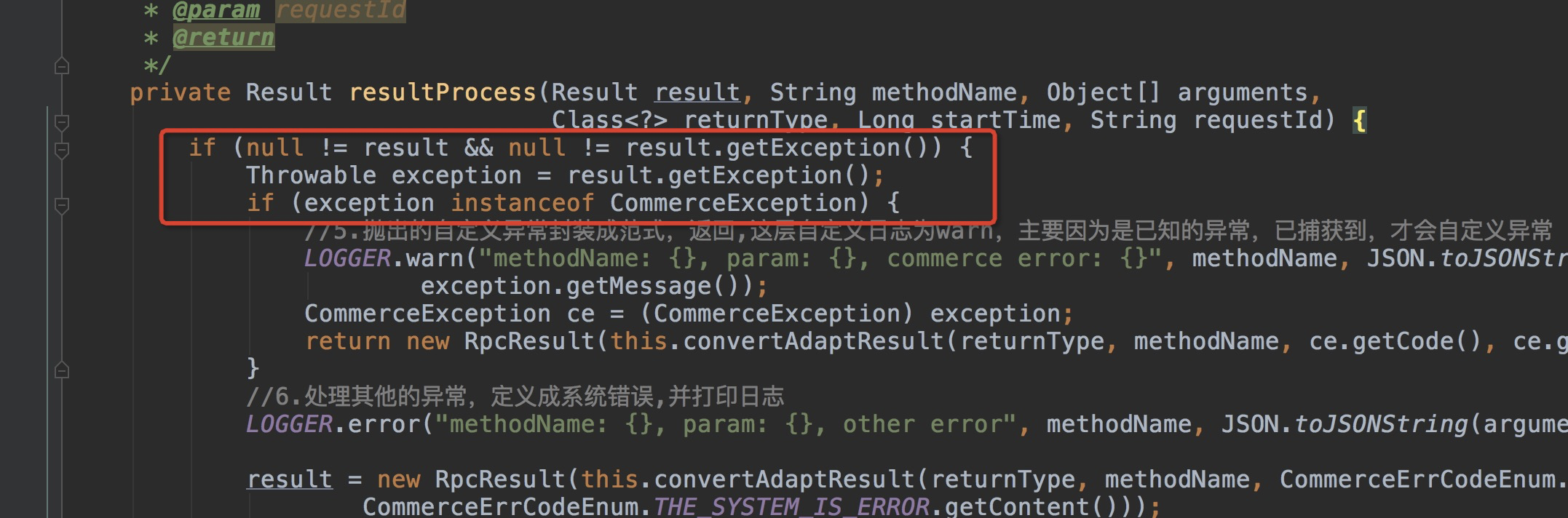
而不是直接抛出异常给调用者，使得调用者来处理你的异常；自定义异常最好能够使用 继承RuntimeException的方式，这里为啥使用RuntimeException异常(不做叙述，因该都有基本的了解）。

如下设计：



3.遇到的坑：实测一个接口抛出自定义异常；在dubbo filter中适配返回值出现问题：

如下这段代码：



错误点在：exception instanceof CommerceException 这段代码为false；所以自定义异常并没有走条件内的封装返回值的代码，和想象中不一样，导致返回值不是我们想要的。

4.解决错误：

1.原因：因为dubbo有个内置的异常拦截器叫ExceptionFilter，看下源码

如下：

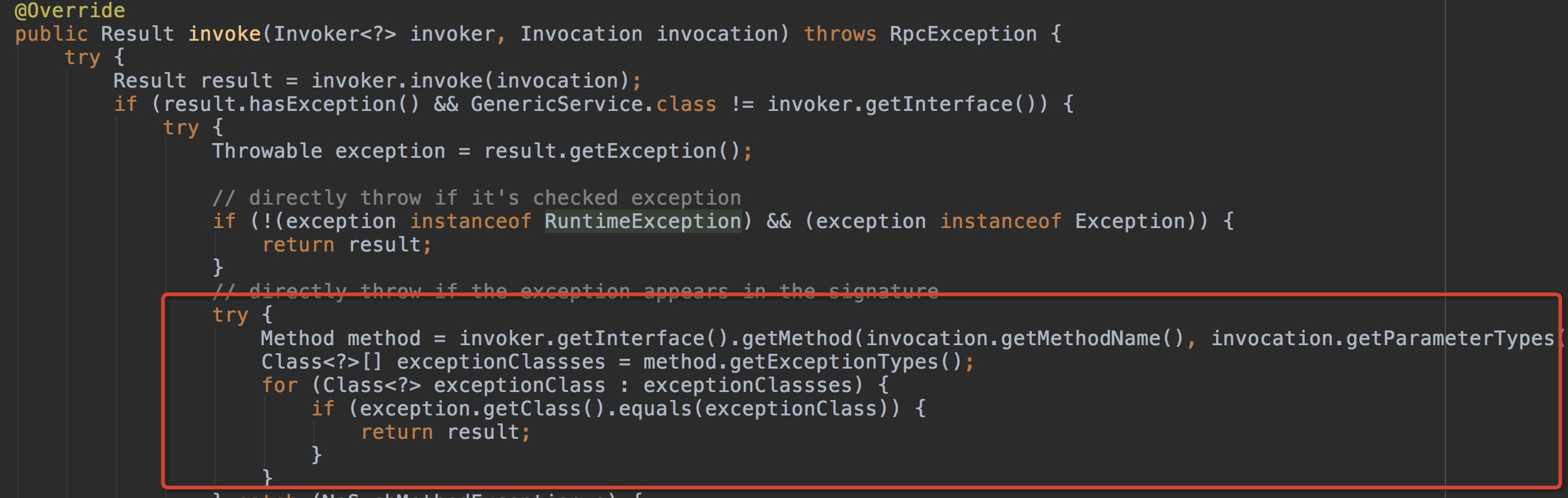


可以明显的看出，我们的自定义异常被直接toString()后在封装成RuntimeException返回，在通过我们的filter，自然就出现上面问题；好了问题找到了，那我们该如何解决呢。

方法一：在自己的对外接口上面加上你抛出的异常类，如下：



这样就解决了，为啥解决了呢？看下刚才dubbo内置拦截器中源码这一段：



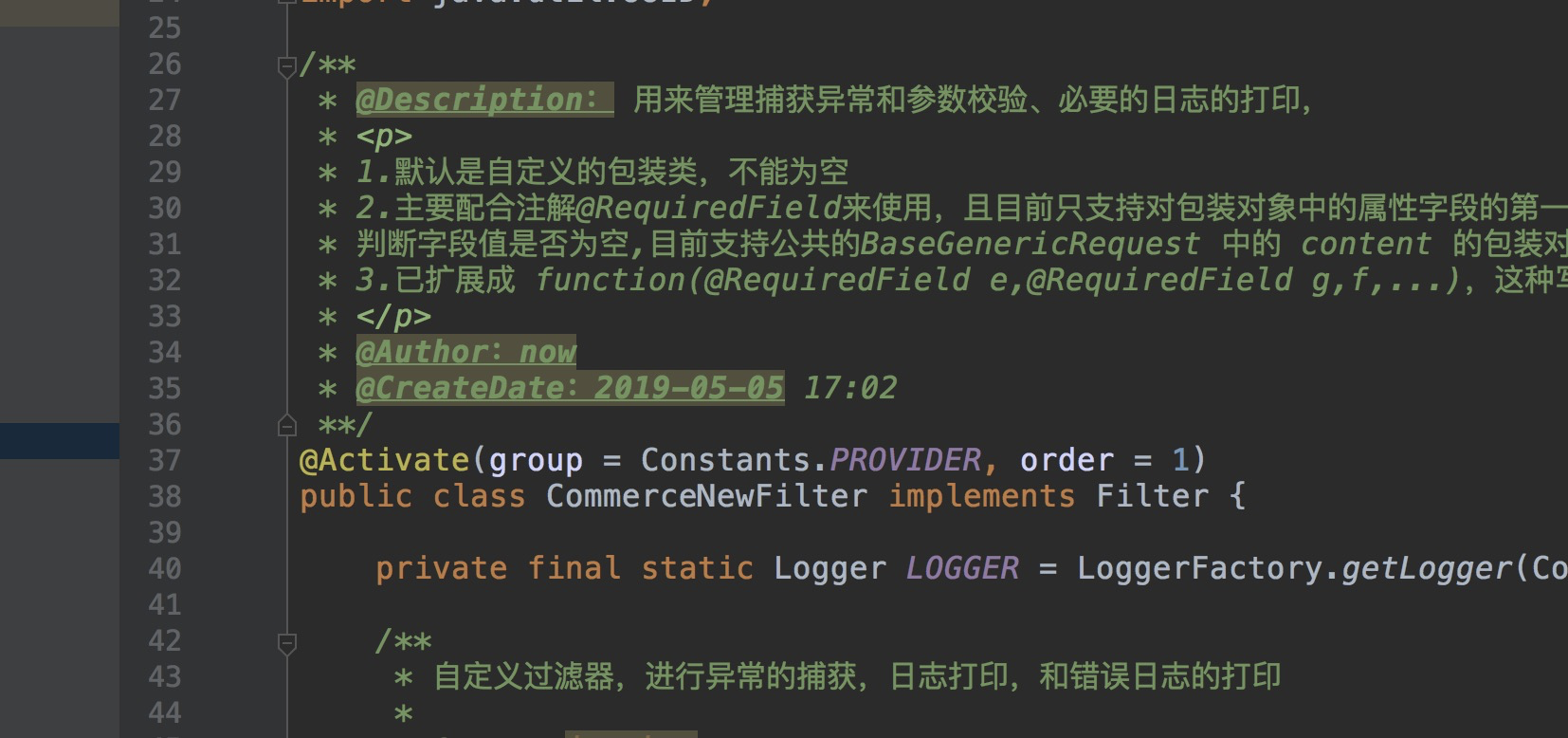
这里回调用反射接口，拿到接口对应的声名式的异常，匹配异常类型是否一致，一致则直接不改变返回结果。这样我们的filter自然就可以正常组装结果。但这种方案会需要每个接口上加 throws 自定义异常，

长远看的该方法的缺点还是比较明显：1.接口多的化，不利于维护 2.调用方在看到你接口时，不知道是否需要处理你的异常，对接上不够通畅。

方法二：目前推荐使用的，如下改动：

1.对外接口不做任何改动

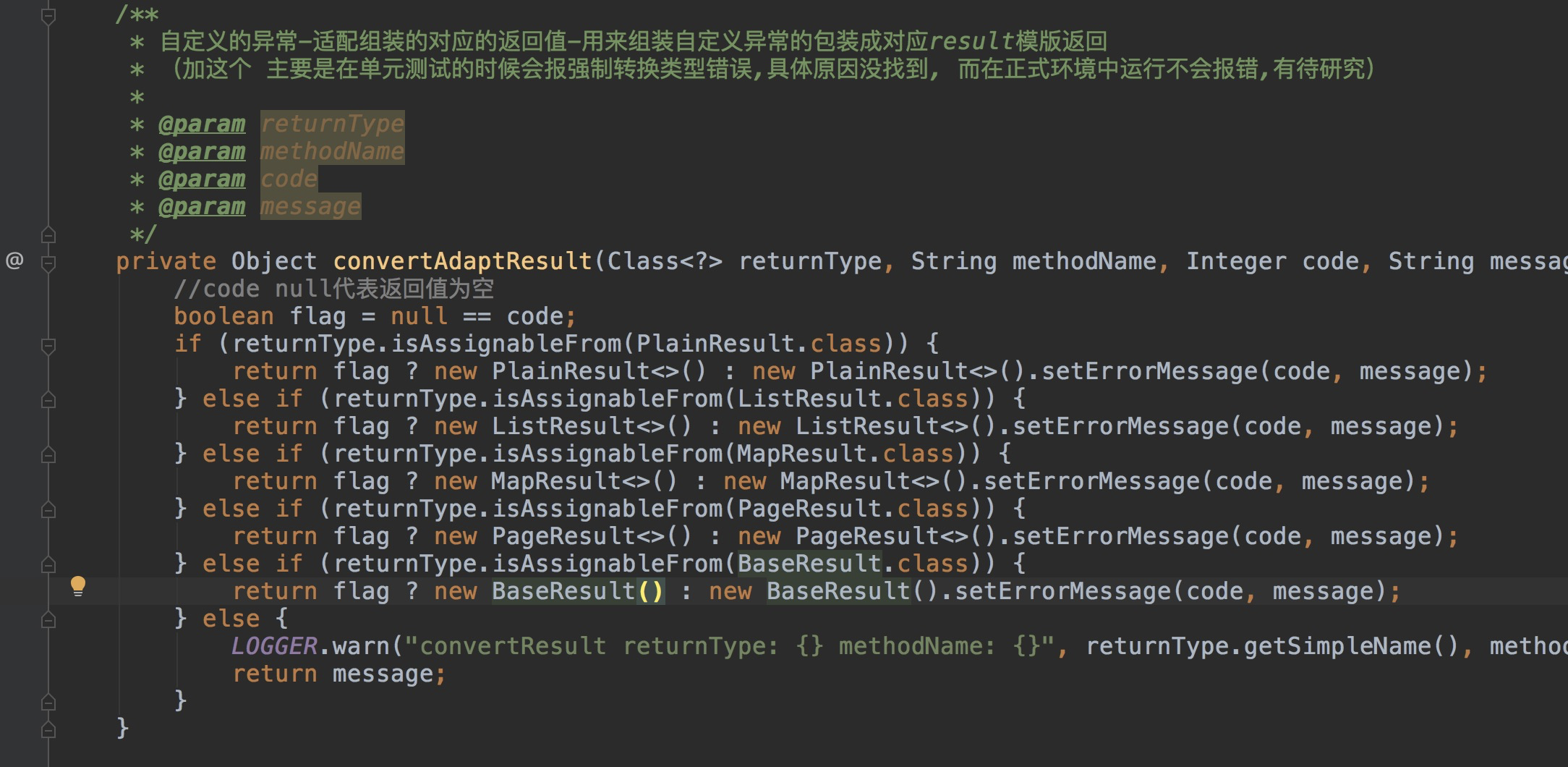
2.在 @Activate 用加上order = 1,因为ExceptionFilter 默认是order = 0;这样由于filter是链路调用，返回值就会先到达你的filter类，则解决上面的问题了。



如上，我们就返回了自己想要的结果，并解决了该问题。如果有更好的解决方案，可以加入评论。

5.注意：好了上面的那个问题解决了，于是我这边开始组装返回值，这时候要注意一点了，如果你只封装成 BaseResult类型返回，在返回类型是它的子类的时候，会出现类型转换错误。

解决方案：拿到对应的返回类型，组装返回：



参考：https://www.jianshu.com/p/c5ebe3e08161