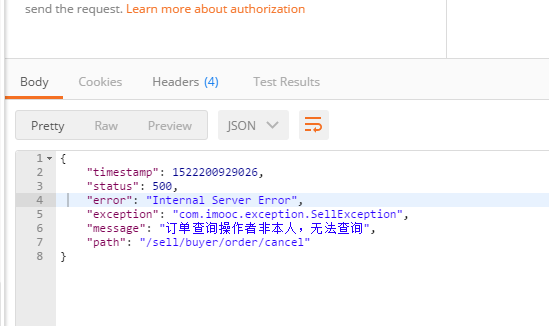
具体做法，就是利用SpringAOP，拦截所有的需要拦截的方法（连接点）行成一个切入点（Pointcut），然后在其前面加入通知（advice就是一些操作），比如我要登录，我的before操作就是查看是否有登录。

如果有，就继续执行方法。

如果没有，就抛出异常。然后统一拦截异常，进行处理。

下面的例子是，把异常统一拦截后，在后台显示。



**我想把异常给统一处理，不让它在前端显示。**

**异常统一处理（ControllerAdvice注解）**

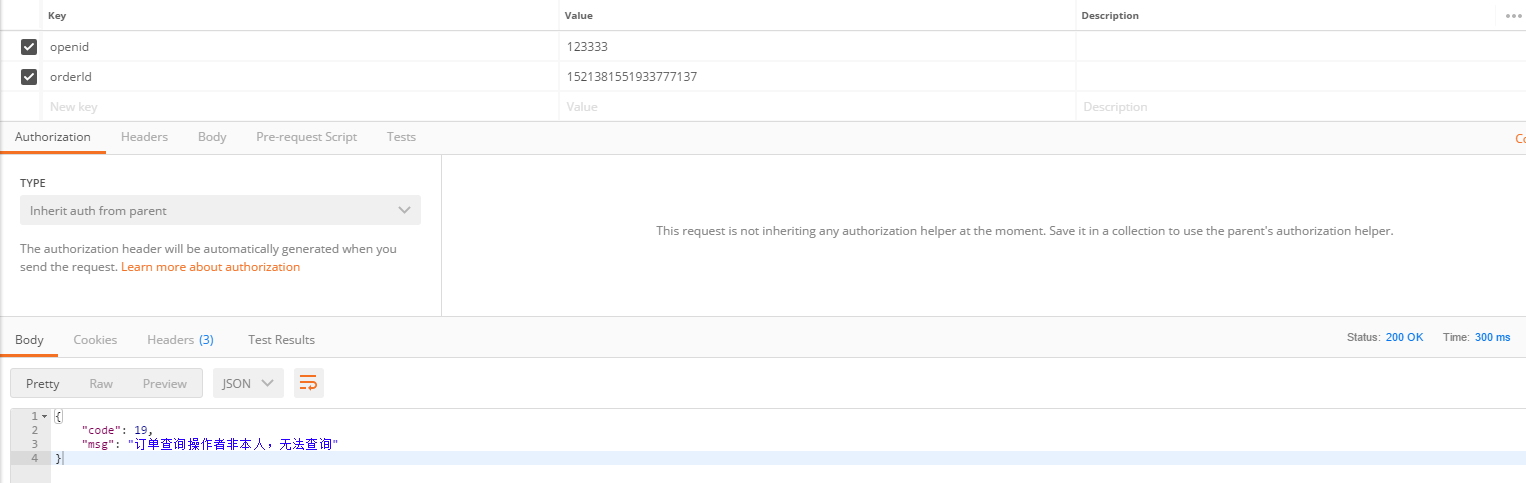
#### 在spring 3.2中，[新增了@ControllerAdvice](mailto:%E6%96%B0%E5%A2%9E%E4%BA%86@controlleradvice) 注解，[可以用于定义@ExceptionHandler](mailto:%E5%8F%AF%E4%BB%A5%E7%94%A8%E4%BA%8E%E5%AE%9A%E4%B9%89@exceptionhandler)、@InitBinder、@ModelAttribute，[并应用到所有@RequestMapping中](mailto:%E5%B9%B6%E5%BA%94%E7%94%A8%E5%88%B0%E6%89%80%E6%9C%89@requestmapping%E4%B8%AD)。参考：[@ControllerAdvice 文档](http://docs.spring.io/spring-framework/docs/5.0.0.M1/javadoc-api/org/springframework/web/bind/annotation/ControllerAdvice.html)

首先[加入@ControllerAdvice](mailto:%E6%96%B0%E5%A2%9E%E4%BA%86@controlleradvice) 注解

#### 

其次要声明拦截抛出的异常类，这里我是自定义一个类SellException。然后方法返回JSON格式。@ResponseBody注解。

#### 启动应用后，被 @ExceptionHandler、@InitBinder、@ModelAttribute等 注解的方法，都会作用在 被 @RequestMapping 注解的方法上。



大家有没有发现异常已经被捕获统一处理了呢。

其实这就涉及到，利用Spring AOP统一处理异常了，这就是面向横向切面的编程啦哈。