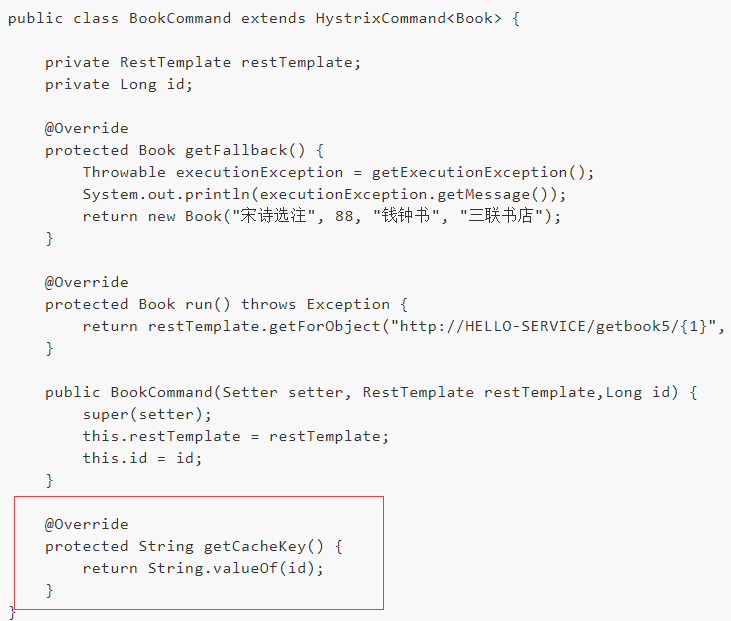
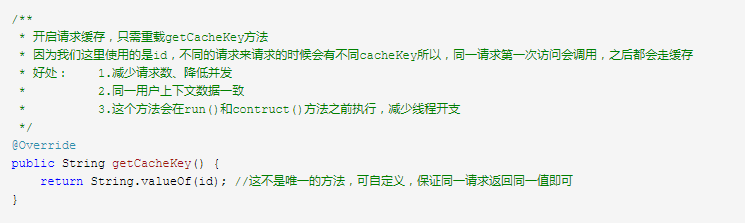
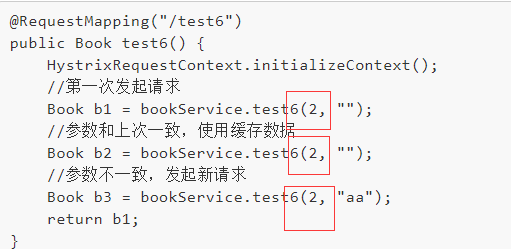
1. 使用了自定义Hystrix请求命令的方式来使用Hystrix，只需要重写getCacheKey方法即可实现请求缓存







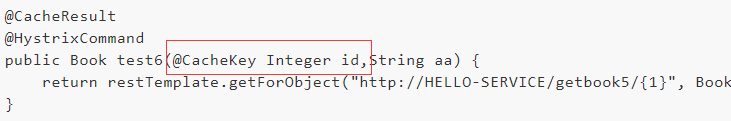


系统在运行时会根据getCacheKey方法的返回值来判断这个请求是否和之前执行过的请求一样，即被缓存，如果被缓存，则直接使用缓存数据而不去请求服务提供者，那么很明显，getCacheKey方法将在run方法之前执行。

1. 通过注解来开启缓存，和缓存相关的注解一共有三个，分别是@CacheResult、@CacheKey和@CacheRemove

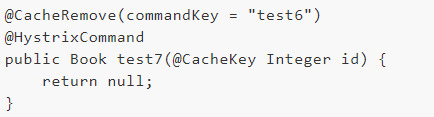
@CacheResult方法可以用在我们之前的Service方法上，表示给该方法开启缓存，默认情况下方法的所有参数都将作为缓存的key





@CacheKey注解指明了缓存的key为id，和aa这个参数无关，此时只要id相同就认为是同一个请求，而aa参数的值则不会作为判断缓存的依据（这里只是举例子，实际开发中我们的调用条件可能都要作为key，否则可能会获取到错误的数据）。如果我们即使用了@CacheResult中cacheKeyMethod属性来指定key，又使用了@CacheKey注解来指定key，则后者失效。

让缓存失效的注解:





网上找了一些解释，如下：





