# Spring Cloud Feign重试

有朋友问到Feign的重试如何做，这里写一下。

先参照我之前写的Feign、Hystrix等文章搭建工程，Eureka（基本上搞Spring Cloud这个是必备的），FeignService，OrderServiceX2。

## 原理说明

在之前的文章也提到过，Feign是通过内置的Ribbon进行负载均衡，并通过HTTP去访问被调用方（在本例中是OrderService）。

所以我们所说的Feign的重试，其实就是Ribbon的重试。

另外，我们在实际工作中使用Feign，必有Hystrix，在这里说一下Hystrix的原理。

其实这一段应该写在Hystrix那篇文章里。

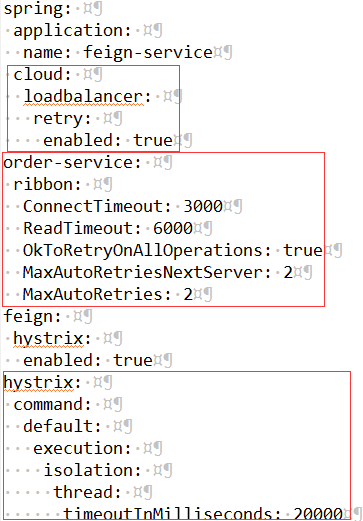
以我在讲Hystrix文章中举的例子说明，没有Hystrix的情况下，用户一个下单操作需要调用订单、支付、物流三个Service，如果其中物流Service因为种种问题不能提供服务，即便另外两个服务都好好的，整个下单操作也会因为物流Service而阻塞住，导致下单服务崩溃。

而Hystrix为每个依赖的服务配置独立的线程池进行隔离，假设下单服务有90个线程，没有Hystrix，则因为物流Service不能提供服务，不停地重试直到90个线程全卡死。而有了Hystrix，三个依赖的服务，每个30个线程，即便物流的全部阻塞了，也不影响另外两个服务的线程。

Hystrix运行机制。Hystrix采用命令模式，简单地说，就是先开启命令，然后执行要调用的方法，在本例中其实就是Feign通过Ribbon调用OrderService服务，如果顺利就返回结果。如果出现异常，比如Hystrix未开启，线程池已满，执行命令失败，执行命令超时，就执行Fallback处理。

## 实现方案

在FeignService的application.yml添加配置：



解释一下，第一个红框没啥好说的，开启负载均衡——就是Ribbon的重试。

第二个红框，order-service是我们要重试的依赖服务。

ConnectTimeout：Http建立连接的超时时间。

ReadTimeout：Http请求读取响应的超时时间。

MaxAutoRetriesNextServer：重试其他实例的最大次数，不包括第一次调用的实例。

MaxAutoRetries：当前实例最大重试次数，不含第一次调用。

第一次调用失败后，对本实例进行MaxAutoRetries次调用。依然失败则调用另一个实例，如果还不行则执行MaxAutoRetriesNextServer次切换，最后还是失败则返回失败信息。

OkToRetryOnAllOperations：对所有操作请求都进行重试。这个得额外说明一下，如果是查询操作，配置成true是可行的，但是如果是非幂等性的写操作，如果被调用方某个实例实际中执行了写操作，仅仅是因为网络或其他原因造成了异常，调用方依然进行重试，这结果就大发了。所以要慎用，而被调用的接口要保证幂等性——这个有机会写一篇文章聊聊。

另外通过红框内order-service的配置项，大家应该会想到一个实际问题：如果调用方有N多个被调用方，那就要写N多个第二个红框的配置啊！所以我之前的文章中说内部调用也可以用网关Zuul，很多朋友留言说没必要，现在看其实是蛮有必要的。

大家有另外的解决办法么，请指教。

第三个红框，之前在讲原理时着重讲了Hystrix，就是怕大家不明白这个参数设置的意义。这个是断路器线程池超时时间，它一定要比Ribbon的超时时间长，否则还没等调用方重试完毕，断路器自己的超时时间到了，就直接进入Fallback了。

顺便说一下，一开始整合Feign和Hystrix时，第一次调用总是会失败，就是因为这个值，默认是1秒，但是我们实际使用的时候，Spring是懒加载的，第一次实例化总是很耗时的，所以很容易就超过1秒报异常。

**PS：以上我设置的数值仅仅是本例测试用，工作中要按照实际情况进行配置。**

## 测试

测试就比较简单了，启动Eureka、OrderServiceX2、FeignService，关闭OrderService，刷新一下看看，可以掐着秒表看一下时间。