**Ribbon配置**

直接在application.properties中配置

# 设置连接超时时间

ribbon.ConnectTimeout=600

# 设置读取超时时间

ribbon.ReadTimeout=6000

# 对所有操作请求都进行重试

ribbon.OkToRetryOnAllOperations=true

# 切换实例的重试次数

ribbon.MaxAutoRetriesNextServer=2

# 对当前实例的重试次数

ribbon.MaxAutoRetries=1

这个参数的测试方式很简单，可以在服务提供者的hello方法中睡眠5s，然后调节这个参数就能看到效果。下面的参数是配置的超时重试参数，超时之后，首先会继续尝试访问当前实例1次，如果还是失败，则会切换实例访问，切换实例一共可以切换两次，两次之后如果还是没有拿到访问结果，则会报Read timed out executing GET http://hello-service/hello。  
但是这种配置是一种全局配置，就是是对所有的请求生效的，如果想针对不同的服务配置不同的连接超时和读取超时，那么可以在属性的前面加上服务的名字

# 设置针对hello-service服务的连接超时时间

hello-service.ribbon.ConnectTimeout=600

# 设置针对hello-service服务的读取超时时间

hello-service.ribbon.ReadTimeout=6000

# 设置针对hello-service服务所有操作请求都进行重试

hello-service.ribbon.OkToRetryOnAllOperations=true

# 设置针对hello-service服务切换实例的重试次数

hello-service.ribbon.MaxAutoRetriesNextServer=2

# 设置针对hello-service服务的当前实例的重试次数

hello-service.ribbon.MaxAutoRetries=1

## Hystrix配置

# 设置熔断超时时间

hystrix.command.default.execution.isolation.thread.timeoutInMilliseconds=10000

# 关闭Hystrix功能（不要和上面的配置一起使用）

feign.hystrix.enabled=false

# 关闭熔断功能

hystrix.command.default.execution.timeout.enabled=false

这种配置也是全局配置，如果想针对某一个接口配置，比如/hello接口

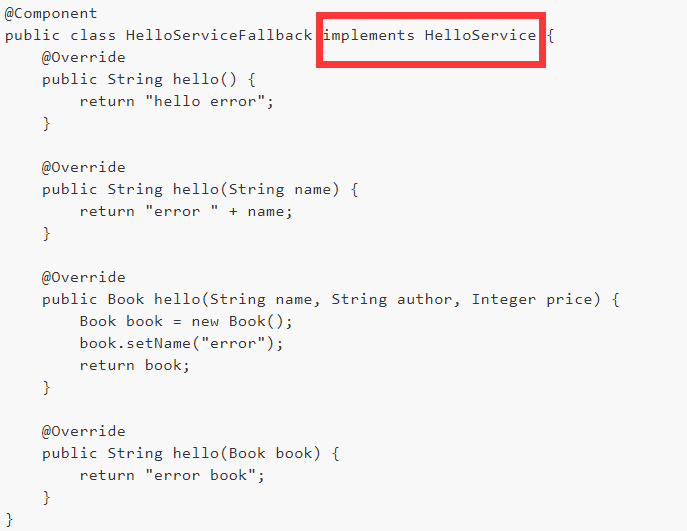
# 设置熔断超时时间

hystrix.command.hello.execution.isolation.thread.timeoutInMilliseconds=10000

# 关闭熔断功能

hystrix.command.hello.execution.timeout.enabled=false

在Feign中配置Hystrix的服务降级

1. 新建一个类
2. 在@FeignClient注解中指定服务降级处理类即可

启动eureka-server和feign-consumer，然后访问相应的接口，可以看到如下结果（**注意这里需要在application.properties中配置feign.hystrix.enabled=true，新版本(Dalston.SR3)的Spring Cloud Feign默认是关闭了Hystrix功能的**）

## 其他配置

1.Spring Cloud Feign支持对请求和响应进行GZIP压缩，以提高通信效率

# 配置请求GZIP压缩

feign.compression.request.enabled=true

# 配置响应GZIP压缩

feign.compression.response.enabled=true

# 配置压缩支持的MIME TYPE

feign.compression.request.mime-types=text/xml,application/xml,application/json

# 配置压缩数据大小的下限

feign.compression.request.min-request-size=2048

2.Feign为每一个FeignClient都提供了一个feign.Logger实例，我们可以在配置中开启日志

**第一步：application.properties中配置日志输出**

# 开启日志 格式为logging.level.+Feign客户端路径

logging.level.org.sang.HelloService=debug

**第二步：入口类中配置日志Bean**

@Bean

Logger.Level feignLoggerLevel() {

   return Logger.Level.FULL;

}