前面的说下 服务注册中心可以用ZK或者Consul来代替

Ribbon 未来有可能被 Alibaba的loadbalance替换掉

Eureka有自带Ribbon 调用方式就是80端口加一个@LoadBalanced 赋予了RestTemplate有负载均衡的能力，其实就是自带Ribbon、Ribbon是配合RestTemplate一起实现调用的

Ribbon

提供客服端负载均衡和服务调用，可以进行自动以的负载均衡，也可以采用简单轮询和随机连接，调用时也包括一系列的连接超时和重试

LB负载均衡

将用户的请求平摊到多个服务器上，达到系统的高可用 一般有Nginx和LVS

Nginx属于软件集中式LB F5属于硬件集中式 而Ribbon属于进程内LB

Ribbon本地负载均衡 会利用RPC远程调用从注册服务中心那里拿到信息存储在JVM上，实现本地负载均衡调用

Ribbon的使用

可以新建一个config进行自定义LB，然后在主启动类添加@RibbonClient注解

Ribbon的算法

第几次请求%服务器集群总数量=调用服务器下标（轮询），下标从1开始，也就是第二台开始

1%2=1

2%2=0

。。。。。。

接下来是有差不多相同功能的OpenFeign，可以和Ribbon和Eureka进行组合使用

Feign是一个声明式的web服务客户端，让编写web服务客户端变得非常容易，只需创建一个接口并在接口上添加注解即可

微服务中往往一个接口会被多处调用，仅仅靠RestTemplate+Ribbon往往是不够的，这时候就需要就一步对微服务进行封装，由Feign来定义和优化接口，用法就是在每个微服务的dao层的接口上除了mapper注解，添加一个Feign注解，和Ribbon不同的是，他属于声明式的

方法，简单的实现服务调用。

OpenFeign相比于Feign基础上支持了springMvc注解，而且@FeignClient可以解析SpringMvc的@requestMapping注解下的接口，通过代理实现负载均衡

用法

主启动类加@EnableFeignClients 最早之前加的是@EnableEurekaClients加@loadbalance（Eureka自带）实现调用的负载均衡

业务类接口上加上@FeignClient即可

一般情况下 8001端口如果暂停了 配好Feign 80的客户端会等待一秒就直接返回错误，解决办法就只有在yaml设置超时控制时间 其中 OpenFeign也是默认支持Ribbon负载均衡的

还有最后一点，OpenFeign也是支持日志打印功能的 通过配置日志config然后在yaml进行开启日志查看。

接下来是豪猪哥

分布式系统面临的问题，应用程序依赖关系之间将不可避免失败，失败了怎么办

，如果其中一个服务超时了，可能会导致服务雪崩

这时候就有了豪猪哥

豪猪哥保证了整体服务不会出现级联故障 保证了分布式系统的高可用

本身就是断路器，当某个服务器发生故障时，不会长时间等待或者直接抛出异常，会给出一个符合的可预期的备选响应（服务降级 限流 熔断 实时监控）

降级就是（服务器忙 请稍后再试，返回友好提示）

程序运行异常 超时 熔断前的降级提示 线程池满也会进行降级提示

熔断就是类似保险丝 达到最大服务访问后 直接拒绝访问 先调用降级提示 然后进行熔断

最后发生心跳连接测试 恢复调用链路

限流 可以理解为阻塞队列 严禁高并发 保证有序排队进行

如何配置 主要还是引入pom包 然后在yml进行配置

定义两个service接口 一个是正常 另外一个是超时

用Jmeter来测试 向8001打满20000个线程 会出现卡顿 tomcat打满

此时80 再进行调用 添乱 80会傻傻的等待 体验感非常差

服务端超时 down机 包括客户端自己发生错误

都可以用降级进行处理

降级除了前面在yml配置完后 要在service层接口上写出兜底方法，在方法上加上@HystrixCommand注解 然后进行配置（该注解除了定义降级还可以定义熔断） 下方还要写上兜底方法，进行友好提示

接下来就是主启动类激活@EnableCircuitBreaker

客户端也可以这样添加 只是主启动类添加的是@EnableHystrix

@DefaultProperties(defaultFallback = "") 可以用这个添加全局兜底方案

这样就不用每个方法都写兜底 是在controller层上进行全局定义的

熔断（5s内20次调用失败会启动熔断机制）

先降级 进而熔断 检测节点响应正常后恢复（涉及到熔断半开）

在service层 @HystrixCommand配置一些熔断参数

是否开启断路器 请求次数 时间范围 失败率达到多少后跳闸

判断开启与关闭也是根据这三个条件设置的值来判断的

熔断类型 打开 关闭 半开

限流由Alibabab Sentinel配置

执行步骤

graphic

豪猪哥也是可以提供服务监控

接下来是新的一章 以前网关用zuul现在用的是Gateway

网关 作为微服务架构提供一种简单有效的统一的API路由管理方式、

底层框架是WebFlux框架（基于Reactor模式Netty框架）

作用 可以提供安全 监控和限流（

方向代理 鉴权 流量控制 熔断 日志监控

）

一般前面还有一个负载均衡nginx（nginx需要通过网关才可以），后面就是连接着各个微服务

graphicgateway比zuul的优势在于 zuul是传统的tomcat容器 用的是servelt IO处理

不适用高并发的今天 gateway采用netty底层 支持长连接websocket 而且建立在springboot2.0以上 支持高并发

这里说下 WebFlux 是一个异步非阻塞的框架 基于Reactor来实现 可以运行在Netty上

Gateway的三大核心

路由 ：由ID 目标URI 一些列的断言和过滤器组成 如果断言为true则匹配该路由

断言 ：例如请求头 请求参数等http请求内容 进行匹配路由

过滤：可以对路由请求前后进行修改

graphic

Eureka是7001 网关是9527

添加网关前 http://localhost:8001/payment/get/31

添加网关后 <http://localhost:9527/payment/get/31>

网关

Gateway zuul已经过期了