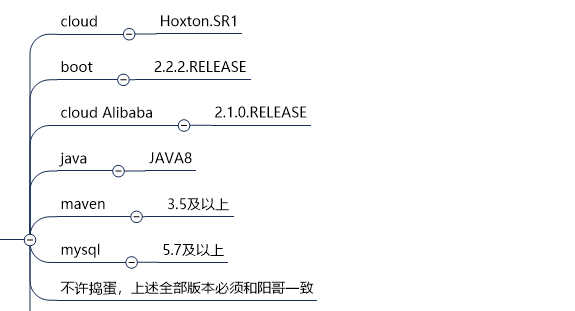
准备事项

Springboot

版本依赖必须一致



graphic

graphic

几个不能用了

Eureka 以前的注册服务中心直接不能用了 改用Nacos

服务调用继续用Ribbon（---唯一不需要换的）

Feign改换成OpenFeign

服务降级熔断 豪猪哥也不能用了 我们用阿里巴巴的sentienl 和哨兵机制一个类型

网关Zuul不能用了 用gateway

还有服务配置的Config也不能用了 用Nacos

服务总线bus也不能用了 用Nacos代替

微服务是需要一个父工程的

具体的步骤

选Maven建项目 各种提前准备 和名字 然后引pom 父工程里面带着各种版本的依赖

子项目

消费者 8001

建微服务module 改pom 写yml 主启动

实体类（SQL 建entity 还有建一个JSON返回值实体类 用于接收Json）

Dao层

因为是Mybaits 所以需要dao接口 PaymentDao 然后写mybaits.xml文件

写service层

一个是接口（方便调用） 另一个是实现类

最后就是controller层

开启DashBoard 方便管理多个微服务

开启热部署

接下来 是订单80接口

前面几部都是一样的 在业务类的时候还是有点不一样的

这边说下 RestTemplate 即可以调用多种Http服务的方法 是spring提供用于访问Restful模板的方法 一般用config配置类表示

graphic

Controller层

直接在controller层加载这个config 用restTemplate利用URL调用8001接口 简单粗暴

但是开发一般不是这样调用 最简单的调用而已

Eureka

服务注册中心

采用CS架构 作为服务注册功能服务器 微服务连接到Eureka Server上 维持心跳连接 维护人员可以通过Eureka Server来监控系统中的各个微服务是否正常运行 是用RPC远程进行调用

graphic

服务注册端用@EnableEurekaServer 客户端用@EnableEurekaServer 服务端提供注册

客户端方便和服务端交互 而已自带负载均衡 默认轮询 保持心跳连接一般是30s

一旦断开超过90s 自动剔除，但是当网络分区故障发生时，微服务与Eureka Server之间无法正常通信，而微服务本身是正常运行的，此时不应该移除这个微服务，所以引入了自我保护机制。

自我保护机制的工作机制是：**如果在15分钟内超过85%的客户端节点都没有正常的心跳，那么Eureka就认为客户端与注册中心出现了网络故障，Eureka Server自动进入自我保护机制**，此时会出现以下几种情况：

Eureka Server不再从注册列表中移除因为长时间没收到心跳而应该过期的服务。

Eureka Server仍然能够接受新服务的注册和查询请求，但是不会被同步到其它节点上，保证当前节点依然可用。

当网络稳定时，当前Eureka Server新的注册信息会被同步到其它节点中。

因此Eureka Server可以很好的应对因网络故障导致部分节点失联的情况，而不会像ZK那样如果有一半不可用的情况会导致整个集群不可用而变成瘫痪。

服务间的调用是用http技术 可是往eureka注册维持心跳用的是rpc

具体流程是支付8001 把自身信息比如URL提供给Eureka 订单80 也注册进Eureka并利用rpc 获取8001的信息 底层用httpclient调用8001接口，获取到的信息会保存在jvm里 每90s更新一次