# Springcloud-Alibaba(nacos)

1. 下载地址 <https://github.com/alibaba/nacos/releases>(windos下启动双击即可)

|  |
| --- |
|  |

### 创建全局版本管理

|  |
| --- |
| <**dependencyManagement**> //dependencyManagement起着关键作用，用来配置全局的依赖管理  <**dependencies**>  <**dependency**>  <**groupId**>org.springframework.cloud</**groupId**>  <**artifactId**>spring-cloud-dependencies</**artifactId**>  <**version**>${spring-cloud.version}</**version**>  <**type**>pom</**type**>  <**scope**>import</**scope**>  </**dependency**>  </**dependencies**> </**dependencyManagement**> |

### 创建服务提供者

1. 加入maven相关依赖

|  |
| --- |
| *<!--依赖父项目的包含springboot的版本信息 -->* <**parent**>  <**groupId**>com.li</**groupId**>  <**artifactId**>myshop-dependencies</**artifactId**>  <**version**>1.0.0-SNAPSHOT</**version**> </**parent**>  *<!-- Spring Cloud Begin -->* <**dependency**>  <**groupId**>org.springframework.cloud</**groupId**>  <**artifactId**>spring-cloud-starter-alibaba-nacos-discovery</**artifactId**> </**dependency**> *<!-- Spring Cloud End -->* |

1. 在spring中增加配置

|  |
| --- |
|  |

1. 在application.yml中进行配置

|  |
| --- |
| *#服务的名称* **spring**:  **application**:  **name**: nacos-provider  **cloud**:  **nacos**:  **discovery**:  *#注册中心的注册地址* **server-addr**: 127.0.0.1:8848 *#服务器的运行端口* **server**:  **port**: 8081 |

### 创建服务消费者（同上：服务名称不同）

### 微服务之间的调用采用Fegin技术

1.增加相关的pom依赖

|  |
| --- |
| *<!--为服务之间的通讯采用fegin-->* <**dependency**>  <**groupId**>org.springframework.cloud</**groupId**>  <**artifactId**>spring-cloud-starter-openfeign</**artifactId**> </**dependency**> |

1. 在application.yml中添加相应的注解

|  |
| --- |
|  |

1. 创建所要提供的微服务接口

|  |
| --- |
|  |

1. 通常在实际开发中一般暴露的fegin接口在api模块，对于微服务的消费者而言只需要依赖提供者的api模块即可，直接调用就可以实现微服务之间的调用
2. 注意：在fegin接口调用的同时，双方都要在入口类（application）中开启@EnableFeignClients

### 微服务熔断机制Sentinel

1.引入相关的pom文件

|  |
| --- |
| *<!--阿里巴巴 熔断器sentinel-->* <**dependency**>  <**groupId**>org.springframework.cloud</**groupId**>  <**artifactId**>spring-cloud-starter-alibaba-sentinel</**artifactId**> </**dependency**> |

1. 在application.yml中开启sentinel的熔断服务

|  |
| --- |
| **feign**:  **sentinel**:  **enabled**: true |

1. 重新编写所要的降级方法的处理（继承fegin接口编写相应的降级方法）

|  |
| --- |
|  |

1. 在fegin接口中增加相应的返回类信息

|  |
| --- |
|  |

### Spring Cloud GateWay的使用

1. 引入相应的pom文件

|  |
| --- |
| *<!--getway网关 -->* <**dependency**>  <**groupId**>org.springframework.cloud</**groupId**>  <**artifactId**>spring-cloud-starter-gateway</**artifactId**> </**dependency**>  *<!--springcloud getway 需要依赖servlet -->* <**dependency**>  <**groupId**>javax.servlet</**groupId**>  <**artifactId**>javax.servlet-api</**artifactId**> </**dependency**>  <!--注意：*springcloud getway 与其冲突需要删掉* -->  <**dependency**>  <**groupId**>org.springframework.boot</**groupId**>  <**artifactId**>spring-boot-starter-web</**artifactId**> </**dependency**> |

1. 在application中增加相应的配置

|  |
| --- |
| *#服务的名称* **spring**:  **application**:  **name**: cloud-gateway  **cloud**:  **nacos**:  **discovery**:  *#注册中心的注册地址* **server-addr**: 127.0.0.1:8848  **gateway**:  *#设置与服务注册发现组件相结合，这样可以采用服务名的路由策略* **discovery**:  **locator**:  **enabled**: **true  routes**:  *#采用自定义路由id* - **id** : nacos-consumer  *#采用loadBalanceClient方式请求，以lb://开头，后面的是注册在Nacos上的服务名* **uri**: lb://nacos-consumer  *#predicates 主要作用是匹配用户的请求，有很多方法* **predicates**:  - Method=GET,POST |

1. 怎样使用网关创建一个全局过滤器

|  |
| --- |
| @Component **public class** AuthFilter **implements** GlobalFilter, Ordered {  @Override  **public** Mono<Void> filter(ServerWebExchange exchange, GatewayFilterChain chain) {  *//获取响应头信息* String token = exchange.getRequest().getQueryParams().getFirst(**"token"**);  **if**(StringUtils.*isEmpty*(token)){  *//返回状态码* exchange.getResponse().setStatusCode(HttpStatus.***NOT\_FOUND***);  **return** exchange.getResponse().setComplete();  }  **return** chain.filter(exchange);  }   @Override  **public int** getOrder() {  *//执行顺序，数字越小优先级越高* **return** 0;  } } |

### Nacos的分布式配置中心

1. 增加相关的依赖

|  |
| --- |
| *<!--增加nacos的分布式配置中心 -->* <**dependency**>  <**groupId**>org.springframework.cloud</**groupId**>  <**artifactId**>spring-cloud-starter-alibaba-nacos-config</**artifactId**> </**dependency**> |

1. 增加bootstrap.yml配置文件

|  |
| --- |
| **spring**:  **application**:  *#此处的配置文件名称为nacos配置中心的名称* **name**: nacos-consumer-config  **cloud**:  **nacos**:  **config**:  *#指定查找名为nacos-consumer-config.yaml的配置文件* **file-extension**: yaml  *#nacos server的地址* **server-addr**: 127.0.0.1:8848 |

1. 对应nacos在配置中心的使用

|  |
| --- |
|  |
|  |