# nacos简介

Nacos 致力于帮助您发现、配置和管理微服务。Nacos 提供了一组简单易用的特性集，帮助您快速实现动态服务发现、服务配置、服务元数据及流量管理。

从上面的介绍就可以看出，**nacos的作用就是一个注册中心**，用来管理注册上来的各个微服务。

# nacos实战入门

接下来，我们就在现有的环境中加入nacos，并将我们的两个微服务注册上去。

## 搭建nacos环境

第1步: 安装nacos

下载地址: https://github.com/alibaba/nacos/releases

下载zip格式的安装包，然后进行解压缩操作

第2步: 启动nacos

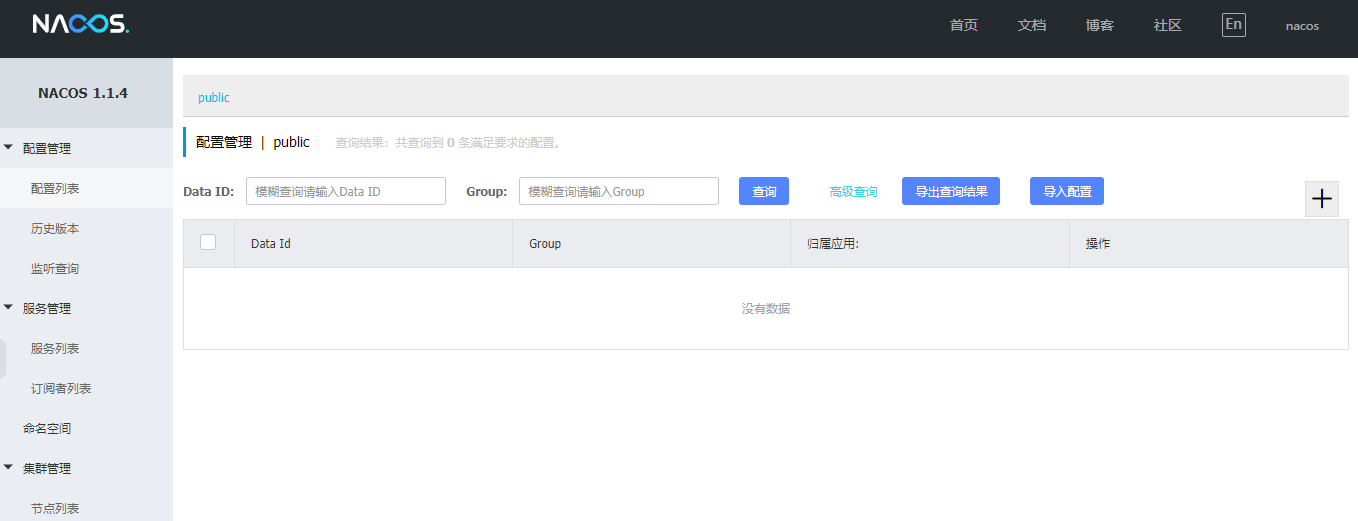
#切换目录

cd nacos/bin #命令启动

startup.cmd -m standalone

第3步: 访问nacos

打开浏览器输入http://localhost:8848/nacos，即可访问服务， 默认密码是nacos/nacos



## 将商品微服务注册到nacos

接下来开始修改shop-product 模块的代码， 将其注册到nacos服务上

1. 在pom.xml中添加nacos的依赖

<!--nacos客户端-->

<dependency>

<groupId>com.alibaba.cloud</groupId>

<artifactId>spring-cloud-starter-alibaba-nacos-discovery</artifactId>

</dependency>

1. 在主类上添加**@EnableDiscoveryClient**注解

@SpringBootApplication @EnableDiscoveryClient

public class ProductApplication

1. 在application.yml中添加nacos服务的地址

spring: cloud:

nacos:

discovery:

server-addr: 127.0.0.1:8848

1. 启动服务， 观察nacos的控制面板中是否有注册上来的商品微服务



## 将订单微服务注册到nacos

接下来开始修改shop\_order 模块的代码， 将其注册到nacos服务上

1. 在pom.xml中添加nacos的依赖

<!--nacos客户端-->

<dependency>

<groupId>com.alibaba.cloud</groupId>

<artifactId>spring-cloud-starter-alibaba-nacos-discovery</artifactId>

</dependency>

1. 在主类上添加**@EnableDiscoveryClient**注解

@SpringBootApplication @EnableDiscoveryClient

public class OrderApplication

1. 在application.yml中添加nacos服务的地址

spring: cloud:

nacos:

discovery:

server-addr: 127.0.0.1:8848

1. 修改OrderController， 实现微服务调用

@RestController @Slf4j

public class OrderController {

@Autowired

private RestTemplate restTemplate;

@Autowired

private OrderService orderService;

@Autowired

private DiscoveryClient discoveryClient;

//准备买1件商品@GetMapping("/order/prod/{pid}")

public Order order(@PathVariable("pid") Integer pid) { log.info(">>客户下单，这时候要调用商品微服务查询商品信息");

//从nacos中获取服务地址ServiceInstance serviceInstance =

discoveryClient.getInstances("service-product").get(0); String url = serviceInstance.getHost() + ":" +

serviceInstance.getPort();

log.info(">>从nacos中获取到的微服务地址为:" + url);

//通过restTemplate调用商品微服务

Product product = restTemplate.getForObject(

"http://" + url + "/product/" + pid, Product.class); log.info(">>商品信息,查询结果:" + JSON.toJSONString(product));

Order order = new Order(); order.setUid(1); order.setUsername("测试用户"); order.setPid(product.getPid());



order.setPname(product.getPname()); order.setPprice(product.getPprice()); order.setNumber(1);

orderService.save(order);

return order;

}

}

DiscoveryClient是专门负责服务注册和发现的，我们可以通过它获取到注册到注册中心的所有服 务

1. 启动服务， 观察nacos的控制面板中是否有注册上来的订单微服务，然后通过访问消费者服务验证调用是否成功

