# 案例1：Nacos 单例服务部署

1、下载解压安装包，执行命令启动（我们使用最近比较稳定的版本 nacos-server-1.2.0.tar.gz）

linux/mac：sh startup.sh -m standalone  
 windows：cmd startup.cmd

2、访问nacos控制台：

<http://127.0.0.1:8848/nacos/#/login> 或者

<http://127.0.0.1:8848/nacos/index.html>（默认端口8848，账号和密码 nacos/nacos）

# 案例2：微服务注册到Nacos

在父pom中引入SCA依赖

  <dependencyManagement>  
       <dependencies>  
           <dependency>  
               <groupId>org.springframework.cloud</groupId>  
               <artifactId>spring-cloud-dependencies</artifactId>  
               <version>Greenwich.RELEASE</version>  
               <type>pom</type>  
               <scope>import</scope>  
           </dependency>  
           <!--SCA -->  
           <dependency>  
               <groupId>com.alibaba.cloud</groupId>  
               <artifactId>spring-cloud-alibaba-dependencies</artifactId>  
               <version>2.1.0.RELEASE</version>  
               <type>pom</type>  
               <scope>import</scope>  
           </dependency>  
       </dependencies>  
   </dependencyManagement>

在商品服务提供者工程中引入nacos客户端依赖，必须删除eureka-client依赖

<dependency>  
 <groupId>com.alibaba.cloud</groupId>  
 <artifactId>spring-cloud-starter-alibaba-nacos-discovery</artifactId>  
</dependency>

application.yml修改，添加nacos配置信息

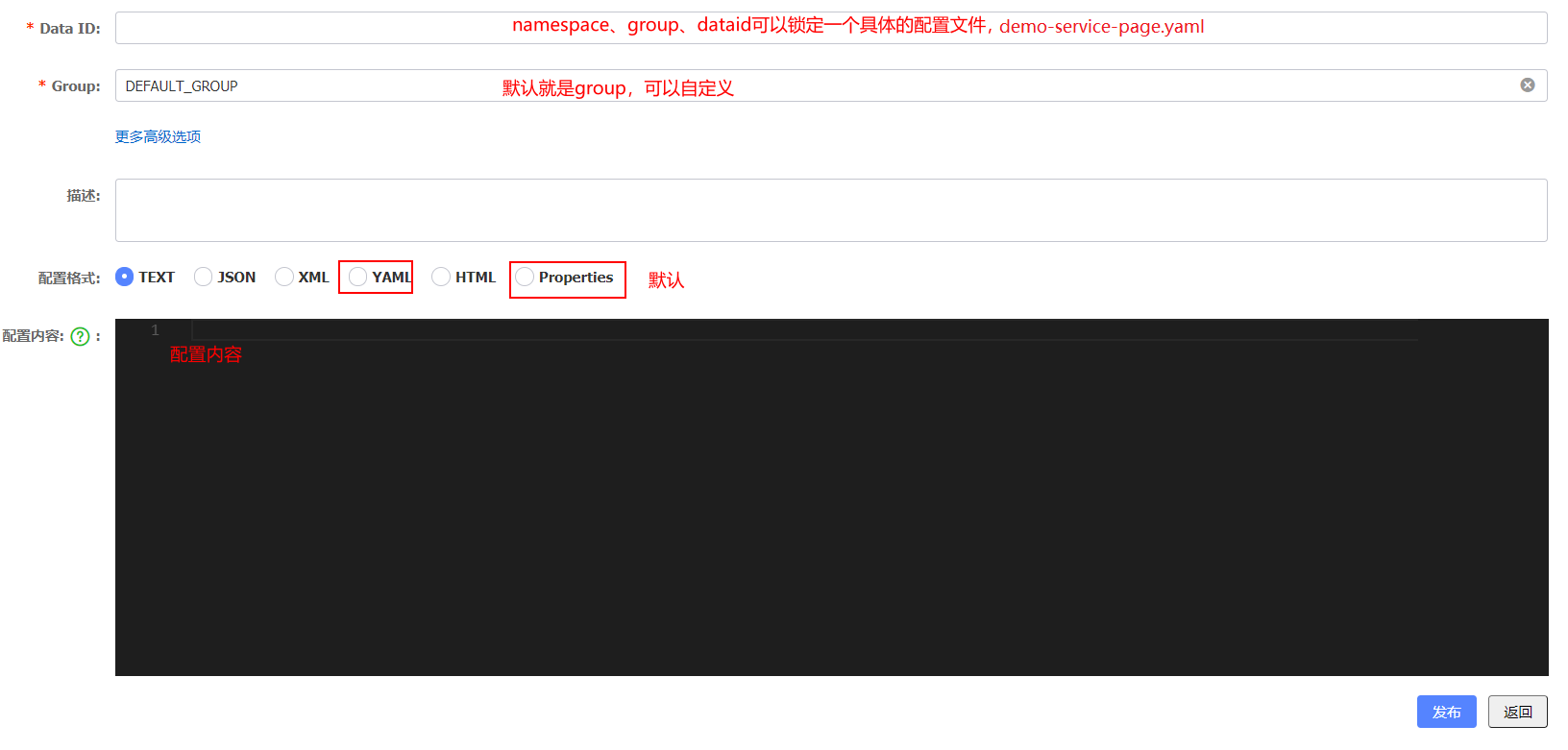
在yml文件中需要删除调用config和eureka相关的配置，否则启动失败

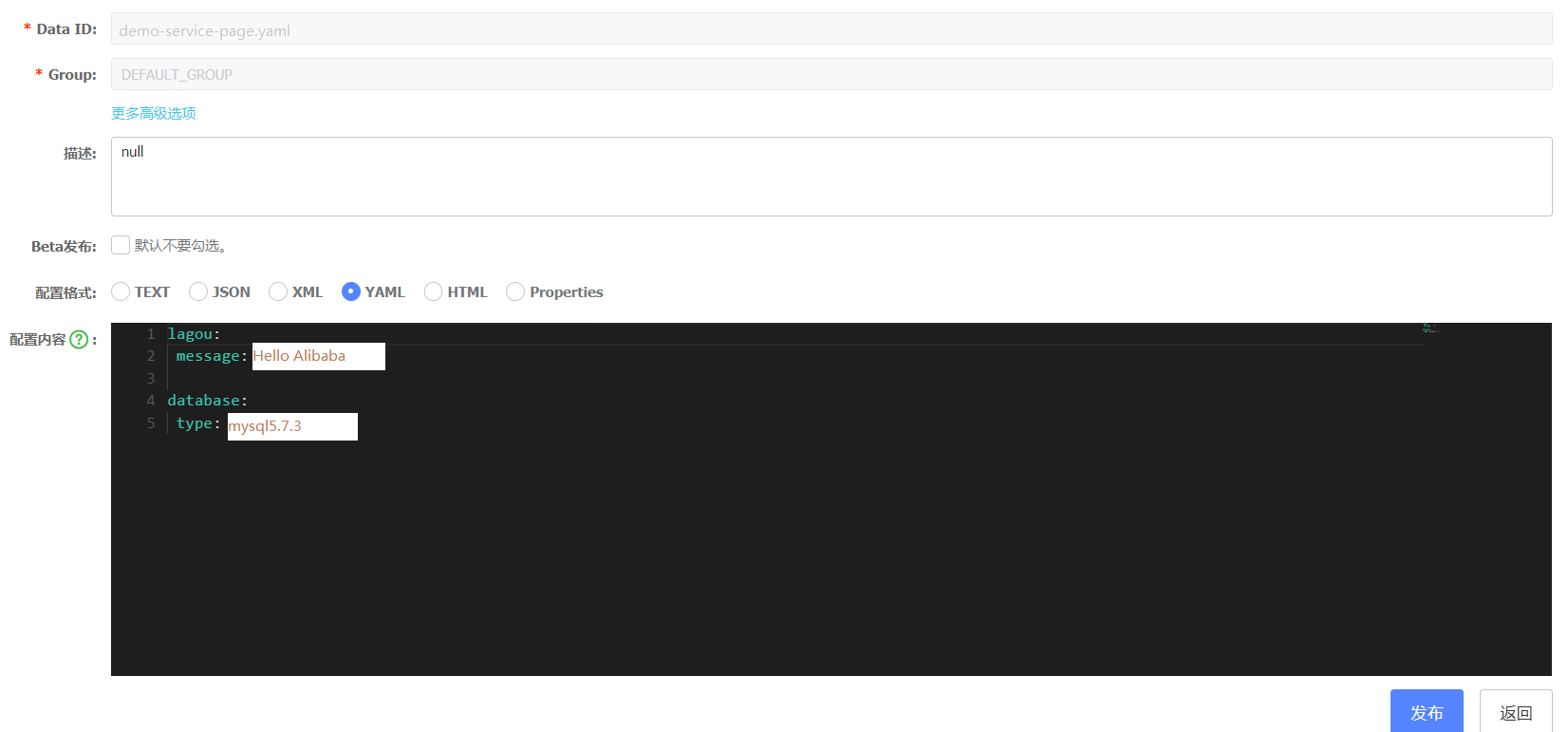
cloud:  
  nacos:  
    discovery:  
      server-addr: 127.0.0.1:8848 #nacos server 地址

启动商品微服务，观察nacos控制台



# 案例3：Nacos实现配置中心





1）添加依赖

<dependency>  
   <groupId>com.alibaba.cloud</groupId>  
   <artifactId>spring-cloud-starter-alibaba-nacos-config</artifactId>  
</dependency>

2）微服务中锁定 Nacos Server 中的配置文件（dataId）

cloud:  
  nacos:  
    discovery:  
      server-addr: 127.0.0.1:8848  
    config:  
      server-addr: 127.0.0.1:8848  
      namespace: 26ae1708-28de-4f63-8005-480c48ed6510 #命名空间的ID  
      group: DEFAULT\_GROUP #如果使用的默认分组,可以不设置  
      file-extension: yaml  
​

3）通过 Spring Cloud 原生注解 @RefreshScope 实现配置自动更新

package com.demo.page.controller;  
​  
import org.springframework.beans.factory.annotation.Value;  
import org.springframework.cloud.context.config.annotation.RefreshScope;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.RestController;  
​  
@RestController  
@RequestMapping("/config")  
@RefreshScope  //自动刷新  
public class ConfigClientController {  
​  
   @Value("${demo.message}")  
   private String message;  
​  
  @Value("${database.type}")  
   private String dbType;  
​  
   @RequestMapping("/remote")  
   public String findRemoteConfig() {  
       return message + " " + dbType;  
  }  
​  
}  
​