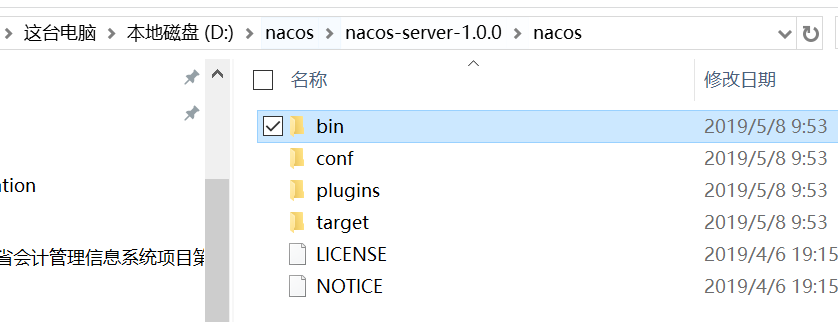
# Nacos服务安装及访问

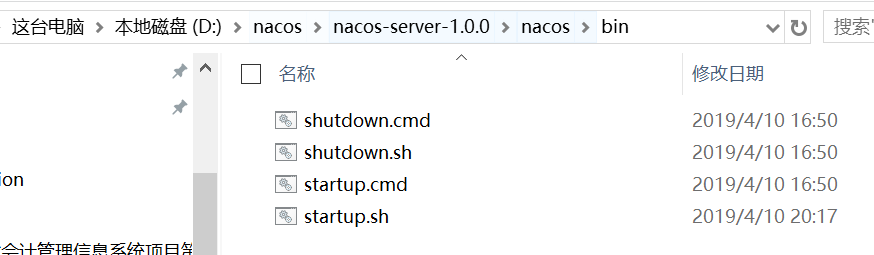
## 1.1、下载及解压查看

地址：<https://github.com/alibaba/nacos/releases>



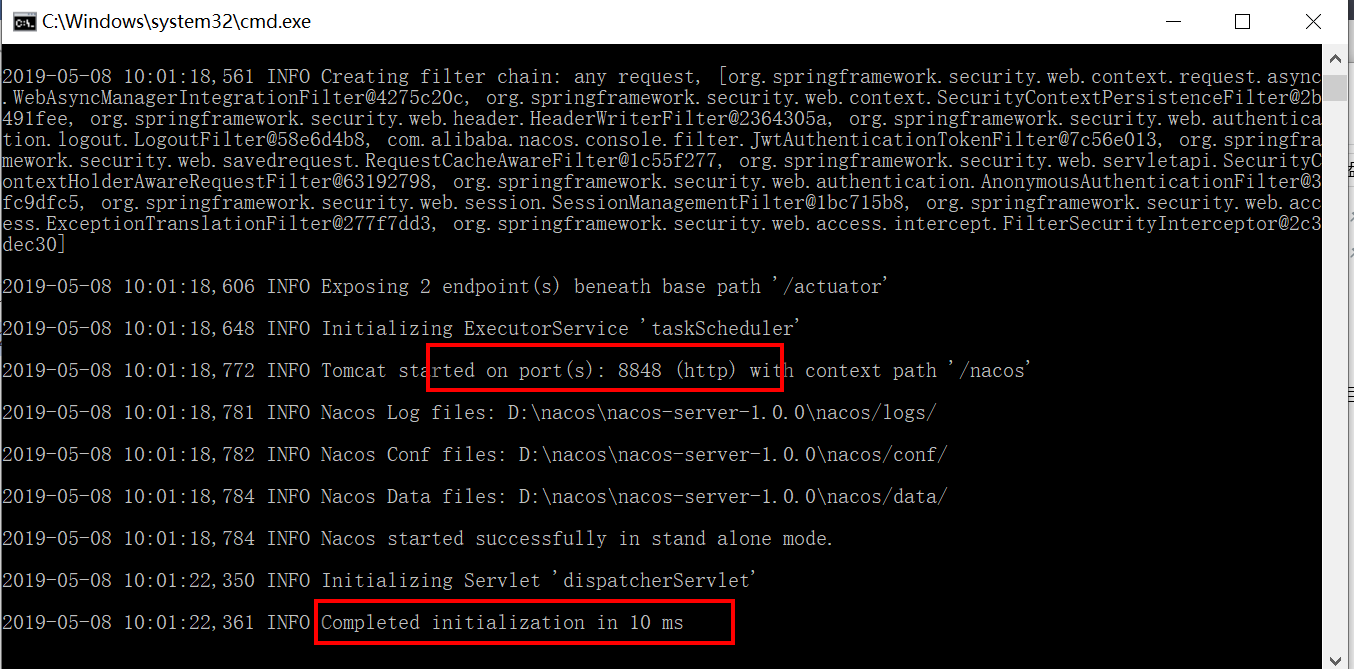
## 1.2、启动（windows下）

进入解压目录：D:\nacos\nacos-server-1.0.0\nacos\bin

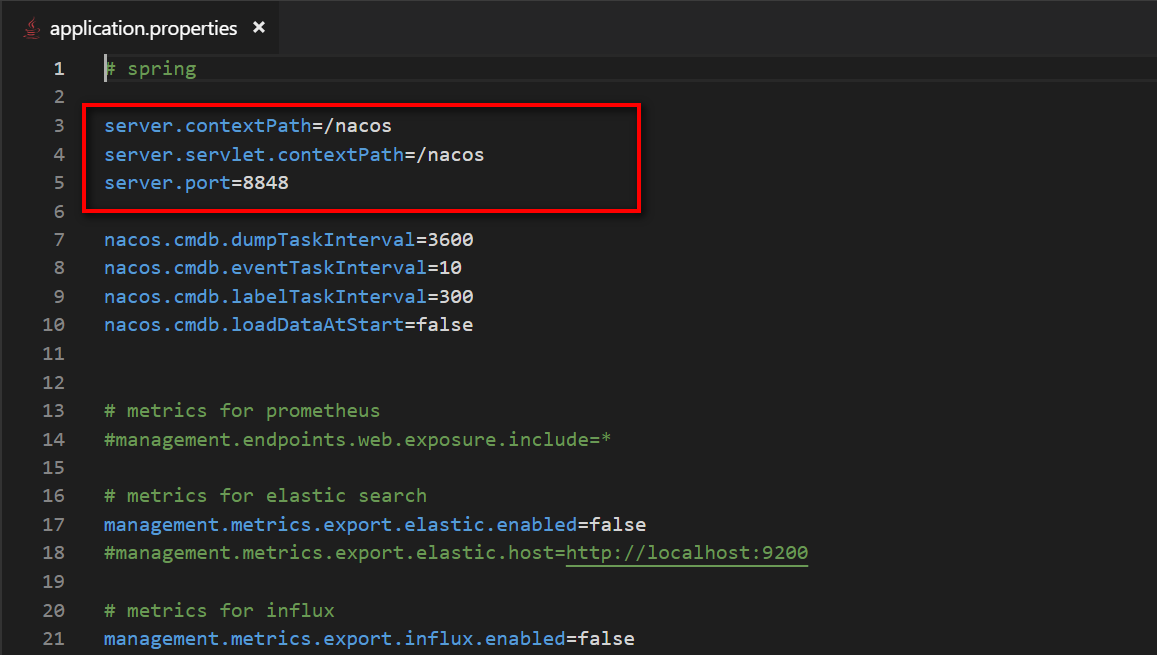


双击startup.cmd即可启动（linux下在命令行进入bin目录，然后sh startup.sh即可启动）

启动之后的界面，默认是8848端口（可以在D:\nacos\nacos-server-1.0.0\nacos\conf\application.properties中查看）



下面给出application.properties的文件内容查看

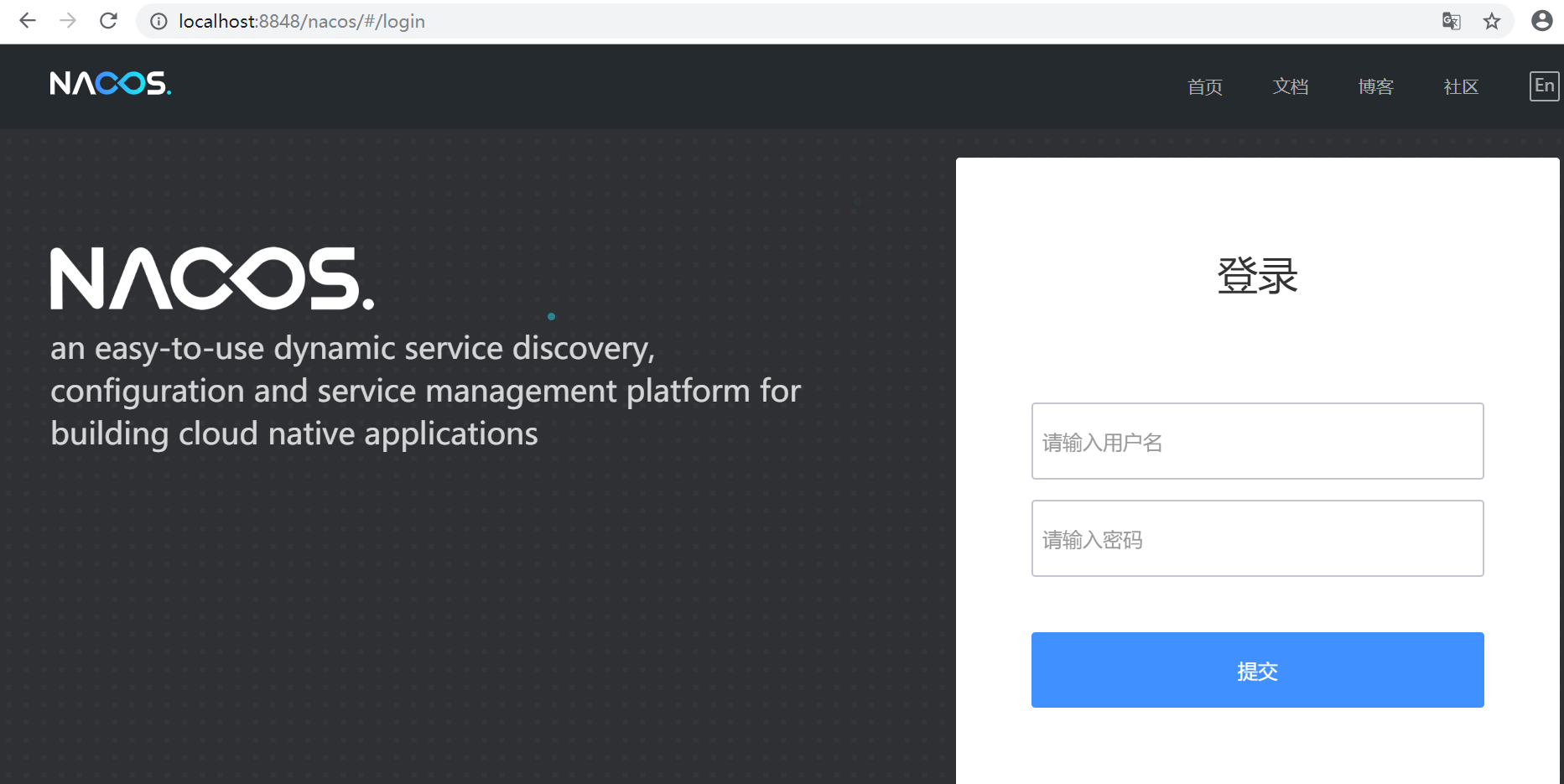


## 1.3、访问

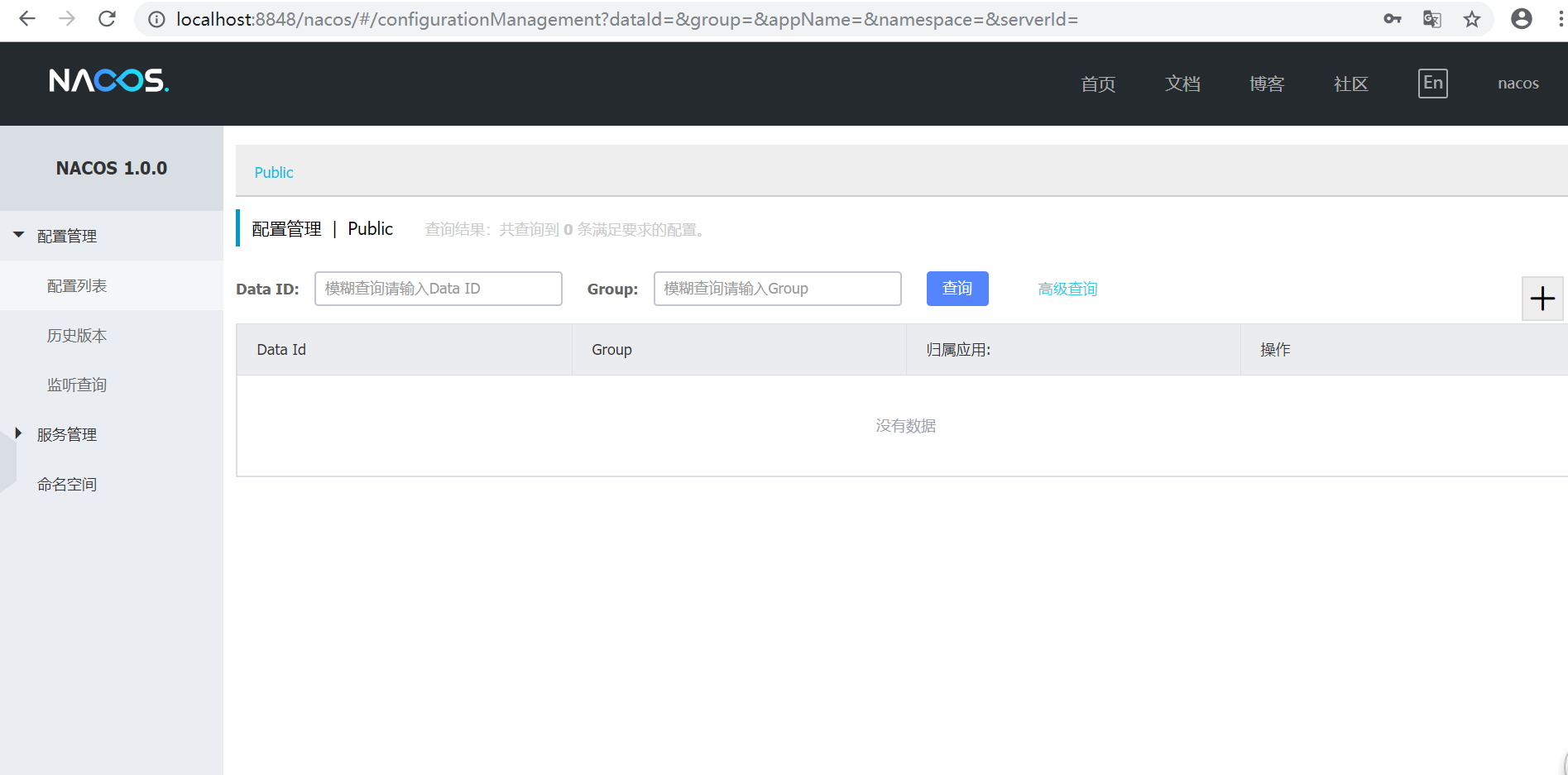
根据application.properties的配置可以知道访问地址为：

<http://localhost:8848/nacos>

访问后的跳转界面，此时会提示登录，默认登录账号密码nacos/nacos



登录成功之后的界面



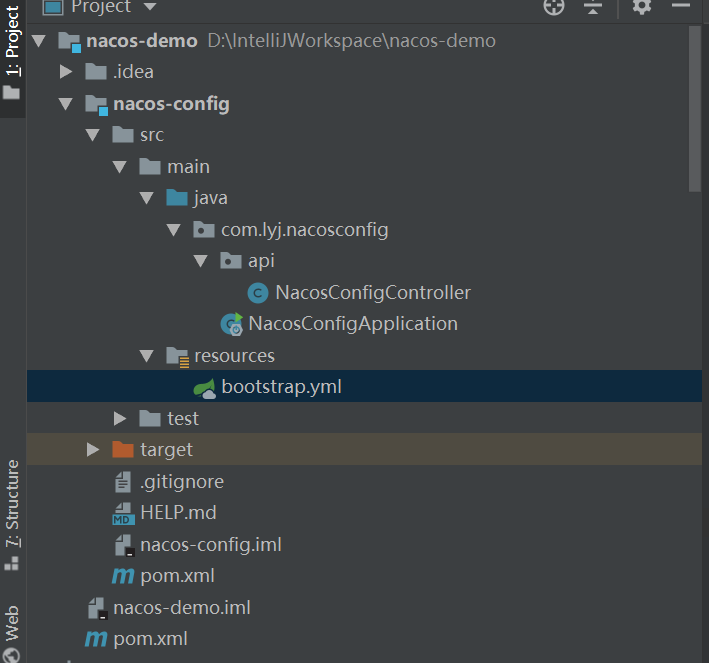
# Nacos结合spring cloud使用

以下项目是基于spring-boot的2.0.4.RELEASE版本开发

## 2.1、nacos配置中心

### 2.1.1、建一个spring cloud项目（查看nacos-config模块即可）

项目结构



parent的pom.xml如下

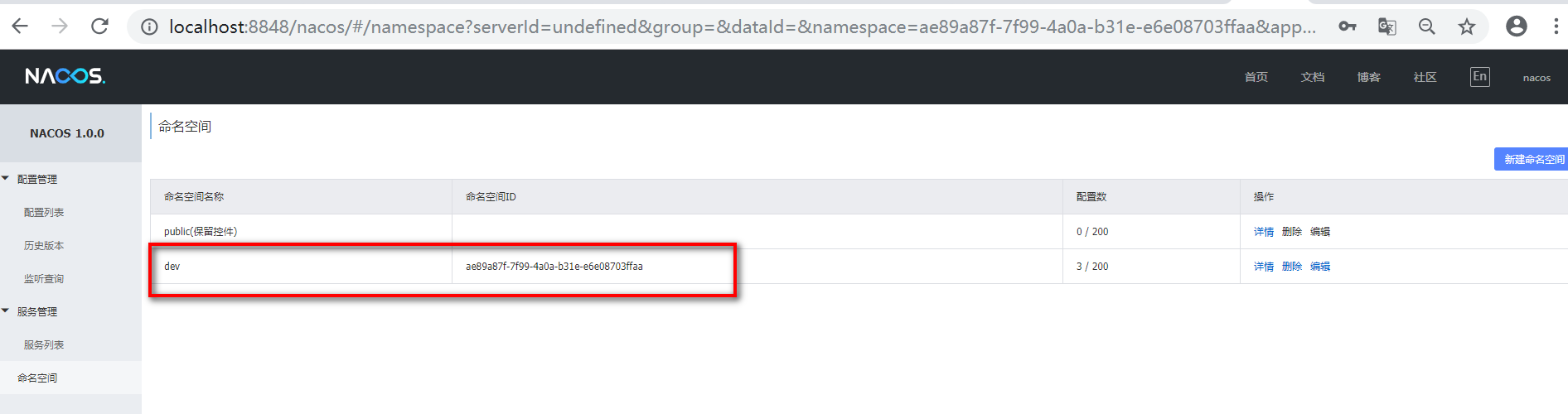
<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <packaging>pom</packaging>  
 <modules>  
 <module>nacos-config</module>  
 </modules>  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>2.0.4.RELEASE</version>  
 <relativePath/> <!-- lookup parent from repository -->  
 </parent>  
 <groupId>com.lyj</groupId>  
 <artifactId>nacos-demo</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>nacos-demo</name>  
 <description>Demo project for Spring Boot</description>  
  
 <properties>  
 <spring-cloud.version>Finchley.SR2</spring-cloud.version>  
 <spring-cloud-alibaba.version>0.2.0.RELEASE</spring-cloud-alibaba.version>  
 <java.version>1.8</java.version>  
 </properties>  
  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
  
 <!-- nacos配置中心 -->  
 <dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-alibaba-nacos-config</artifactId>  
 </dependency>  
 </dependencies>  
  
 <dependencyManagement>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-alibaba-dependencies</artifactId>  
 <version>${spring-cloud-alibaba.version}</version>  
 <type>pom</type>  
 <scope>import</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-dependencies</artifactId>  
 <version>${spring-cloud.version}</version>  
 <type>pom</type>  
 <scope>import</scope>  
 </dependency>  
 </dependencies>  
 </dependencyManagement>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
  
</project>

#### 2.1.1.1、pom.xml

<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <parent>  
 <groupId>com.lyj</groupId>  
 <artifactId>nacos-demo</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <relativePath/> <!-- lookup parent from repository -->  
 </parent>  
 <groupId>com.lyj</groupId>  
 <artifactId>nacos-config</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>nacos-config</name>  
 <description>Demo project for Spring Boot</description>  
 <packaging>jar</packaging>  
  
 <properties>  
 <java.version>1.8</java.version>  
 </properties>  
  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
</project>

#### 2.1.2、bootstrap.xml

这里需要注意的就是使用到了nacos的命名空间配置，需要对应的新建的命名空间id



server:  
 port: 28080  
  
nacos:  
 server-addr: 127.0.0.1  
 namespace: ae89a87f-7f99-4a0a-b31e-e6e08703ffaa  
  
  
*#此处读取的dataId为nacos-config.yml，格式是${spring.cloud.nacos.config.prefix}-{spring.profile.active}.{spring.cloud.nacos.config.file-extension}  
#默认读取的${spring.application.name}-{spring.profile.active}.{spring.cloud.nacos.config.file-extension}*spring:  
 application:  
 name: nacos-config  
 cloud:  
 nacos:  
 config:  
 server-addr: ${nacos.server-addr}  
 file-extension: yml  
 namespace: ${nacos.namespace}  
*# spring.cloud.nacos.config.prefix此属性可以配置nacos访问的配置文件前缀，默认是spring.applicaiton.name的值  
# prefix: nacos-config*

#### 2.1.3、ConfigController

记得增加@RefreshScope注解，这个是为了配置中心修改对应的文件会自动刷新配置，以下的configValue会用到

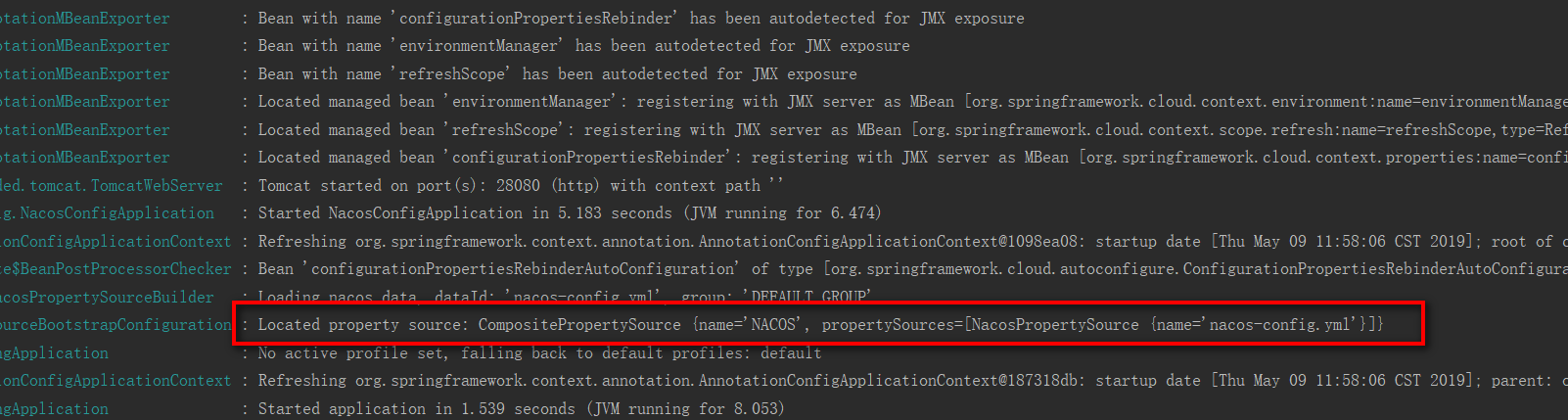
package com.lyj.nacosconfig.api;  
  
import org.springframework.beans.factory.annotation.Value;  
import org.springframework.cloud.context.config.annotation.RefreshScope;  
import org.springframework.http.MediaType;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.RestController;  
  
@RestController  
@RefreshScope  
public class NacosConfigController {  
 @Value("${configValue: true}")  
 private boolean configValue;  
  
 @GetMapping(value = "getConfigValue", produces = MediaType.*APPLICATION\_JSON\_VALUE*)  
 public String getConfigValue(){  
 return String.*valueOf*(configValue);  
 }  
}

#### 2.1.4、NacosConfigApplication

package com.lyj.nacosconfig;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class NacosConfigApplication {  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(NacosConfigApplication.class, args);  
 }  
  
}

#### 2.1.5、启动日志查看是否成功（需要在2.2之后方能启动）

如果出现以下内容就证明已经添加到nacos管理了

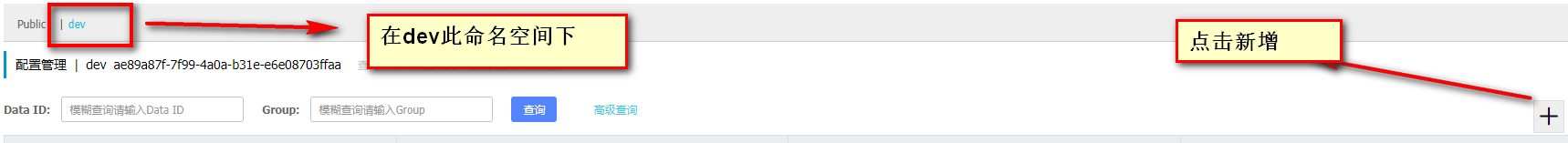


### 2.1.2、在nacos上添加dataId步骤：

1. 新增一个命名空间（这里可以不新增，默认是public命名空间）

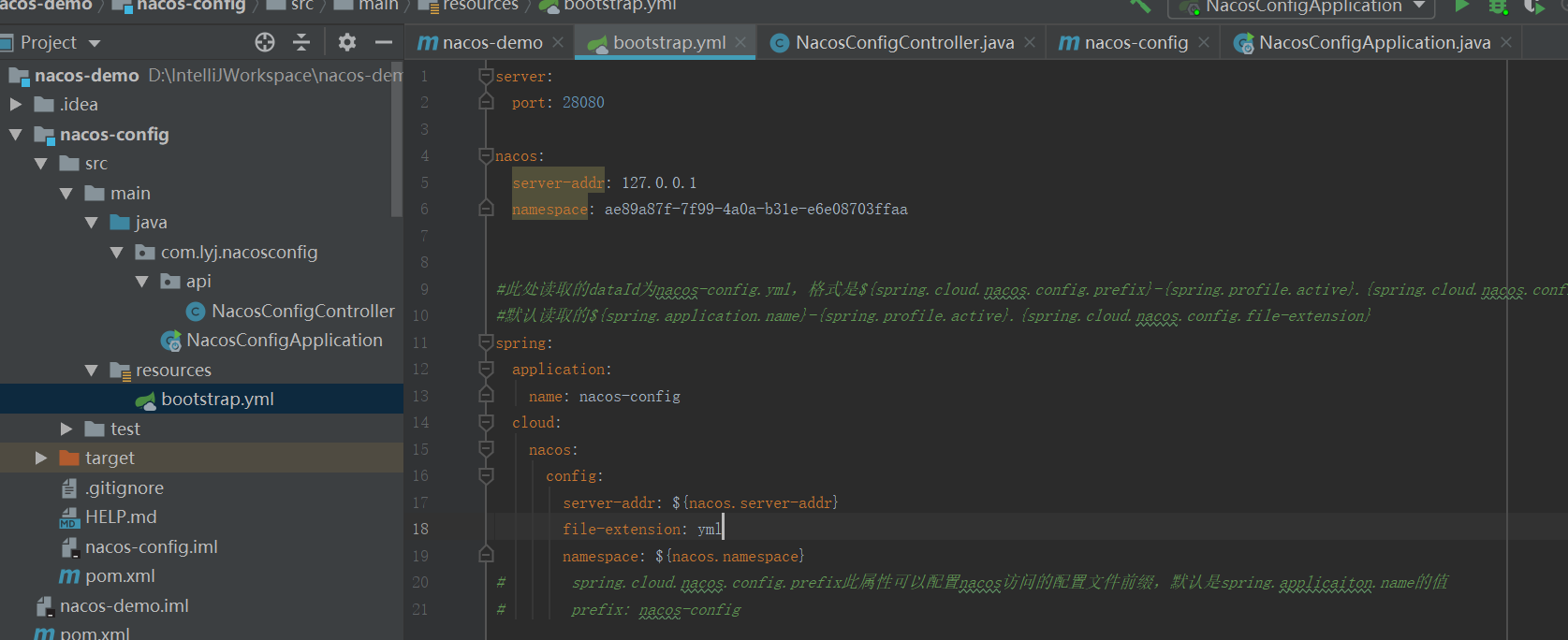


1. 点击+



1. 配置（这里需要配置对应的命名空间，使用属性是spring.cloud.nacos.config.namespace）

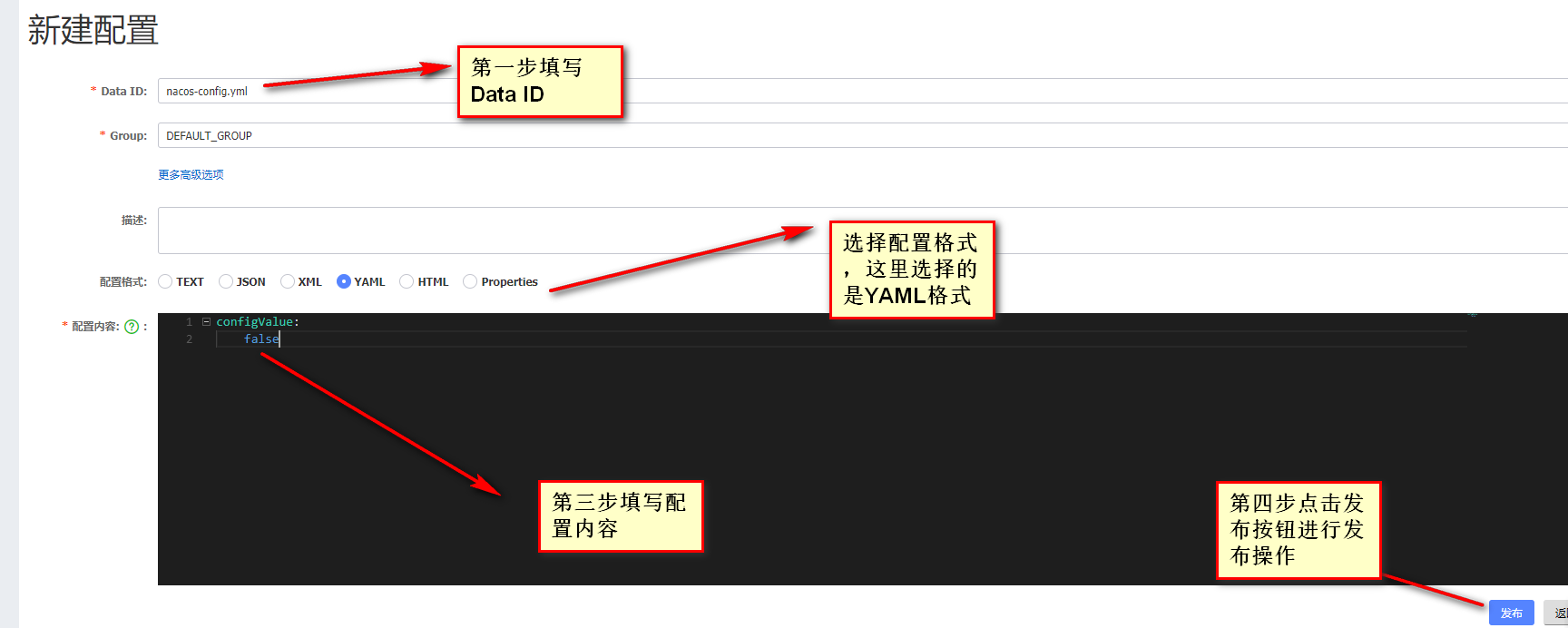
DataId的完整格式为，这个是在项目中的bootstrap.xml文件里面配置



DataId的完整格式为

格式是${spring.cloud.nacos.config.prefix}-{spring.profile.active}.{spring.cloud.nacos.config.file-extension}

如果spring.cloud.nacos.config.prefix为空的时候，默认读取的${spring.application.name}-{spring.profile.active}.{spring.cloud.nacos.config.file-extension}



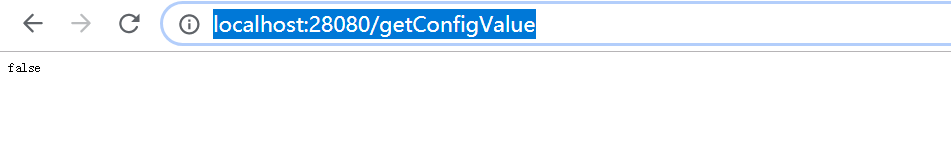
1. 发布完成查看配置（在dev命名空间下的）



### 2.1.3、访问

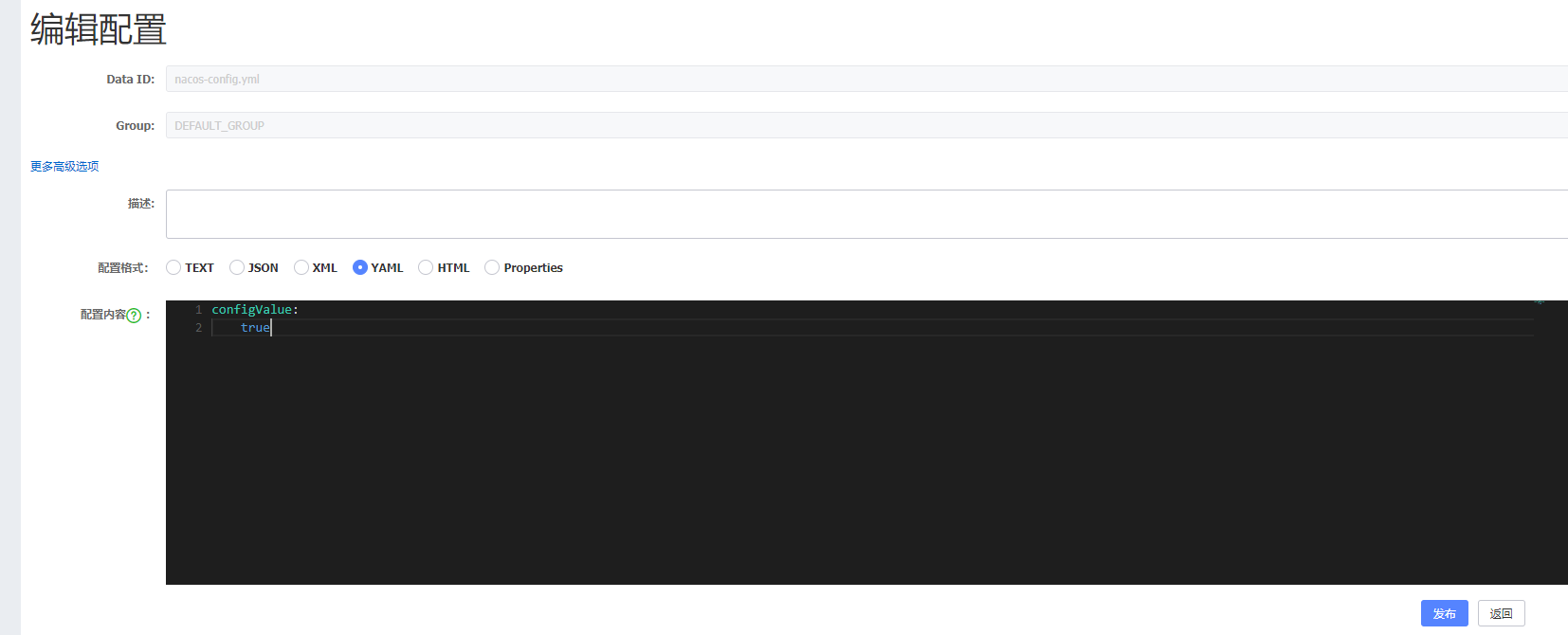
1）、会默认优先读取nacos的配置nacos.yml，而不是controller中的配置

<http://localhost:28080/getConfigValue>

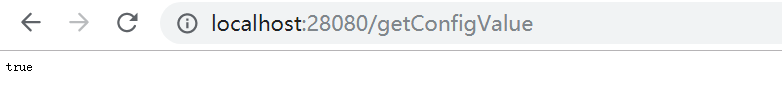


1. 、此时修改nacos.yml的配置为以下（这是的需要在ConfigController中加上@RefreshScope才能看到效果）

修改之后点击“发布”



重新访问：

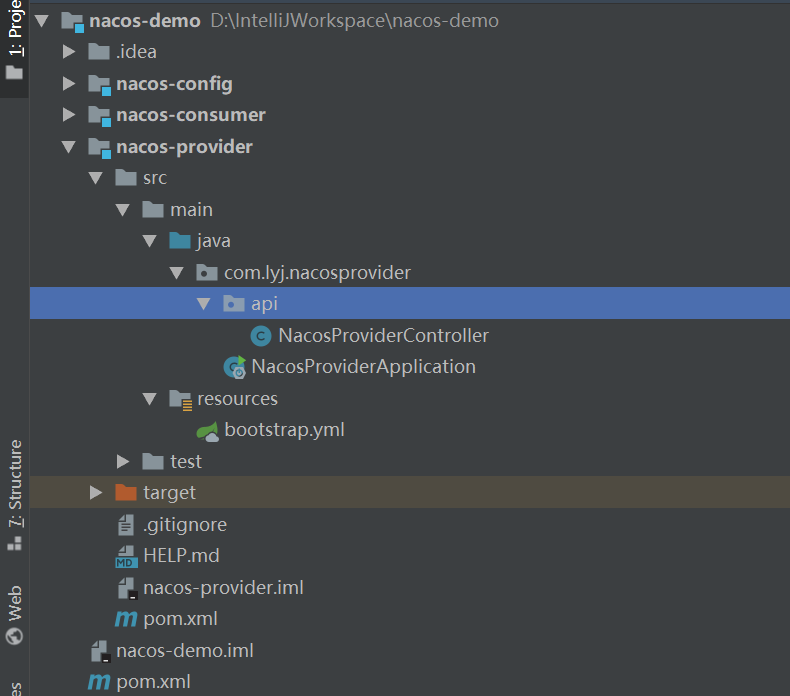


## 2.2、nacos服务注册与发现

父pom.xml

### 2.2.1、服务提供者nacos-provider（查看nacos-provider模块即可）

项目结构



#### 2.2.1.1、pom.xml

<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <parent>  
 <groupId>com.lyj</groupId>  
 <artifactId>nacos-demo</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <relativePath/> <!-- lookup parent from repository -->  
 </parent>  
 <groupId>com.lyj</groupId>  
 <artifactId>nacos-provider</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>nacos-provider</name>  
 <description>Demo project for Spring Boot</description>  
 <packaging>jar</packaging>  
  
 <properties>  
 <java.version>1.8</java.version>  
 </properties>  
  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
  
 <!-- nacos服务发现与注册 -->  
 <dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-alibaba-nacos-discovery</artifactId>  
 </dependency>  
 </dependencies>  
  
</project>

#### 2.2.1.2、bootstrap.yml

server:  
 port: 28081  
  
nacos:  
 server-addr: 127.0.0.1  
 namespace: ae89a87f-7f99-4a0a-b31e-e6e08703ffaa  
  
*#此处读取的dataId为nacos-config.yml，格式是${spring.cloud.nacos.config.prefix}-{spring.profile.active}.{spring.cloud.nacos.config.file-extension}  
#默认读取的${spring.application.name}-{spring.profile.active}.{spring.cloud.nacos.config.file-extension}*spring:  
 application:  
 name: nacos-provider  
 cloud:  
 nacos:  
 config:  
 server-addr: ${nacos.server-addr}  
 file-extension: yml  
 namespace: ${nacos.namespace}  
*# spring.cloud.nacos.config.prefix此属性可以配置nacos访问的配置文件前缀，默认是spring.applicaiton.name的值  
# prefix: nacos-provider* prefix: nacos-provider  
 discovery:  
 server-addr: ${nacos.server-addr}  
 namespace: ${nacos.namespace}

#### 2.2.1.3、NacosProviderController

需要加上@RefreshScope

package com.lyj.nacosprovider.api;  
  
import com.alibaba.fastjson.JSON;  
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;  
  
import java.util.Map;  
  
@RestController  
@RefreshScope  
public class NacosProviderController {  
 @Value("${spring.application.name}")  
 private String springApplicationName;  
  
 @RequestMapping(value = "provider")  
 public String provider(Map<String, Object> params){  
 return "当前服务为：" + springApplicationName + "->params:"+ JSON.*toJSONString*(params);  
 }  
}

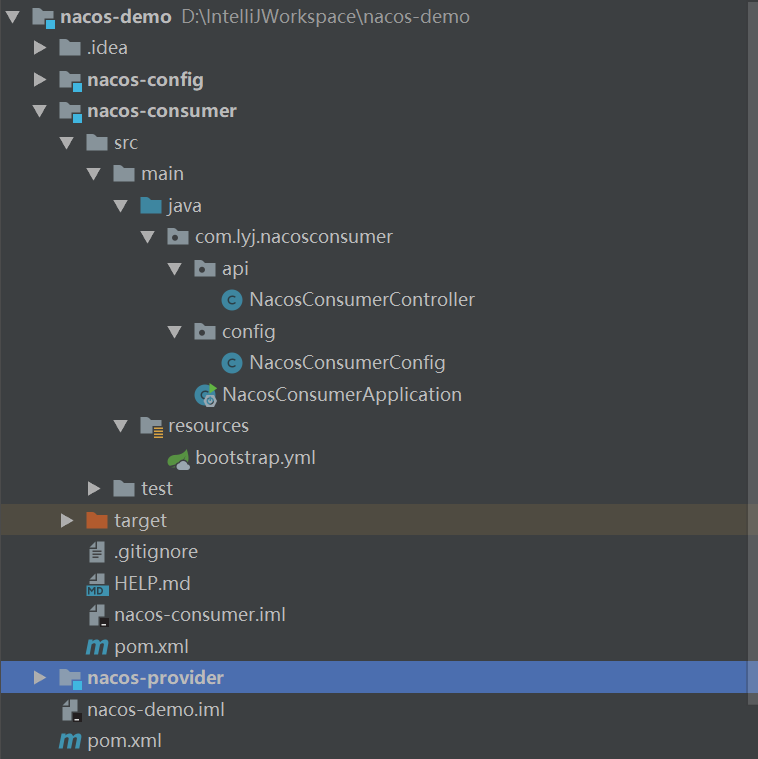
#### 2.2.1.4、NacosProviderApplication

记得加上@EnableDiscoveryClient注解，将此服务注册到nacos注册中心

package com.lyj.nacosprovider;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.cloud.client.discovery.EnableDiscoveryClient;  
  
@SpringBootApplication  
@EnableDiscoveryClient //开启注册中心发现  
public class NacosProviderApplication {  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(NacosProviderApplication.class, args);  
 }  
  
}

### 2.2.2、服务消费者nacos-consumer（查看nacos-consumer模块即可）

项目结构



#### 2.2.2.1、pom.xml

<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <parent>  
 <groupId>com.lyj</groupId>  
 <artifactId>nacos-demo</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <relativePath/> <!-- lookup parent from repository -->  
 </parent>  
 <groupId>com.lyj</groupId>  
 <artifactId>nacos-consumer</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>nacos-consumer</name>  
 <description>Demo project for Spring Boot</description>  
 <packaging>jar</packaging>  
  
 <properties>  
 <java.version>1.8</java.version>  
 </properties>  
  
 <dependencies>  
 <!-- nacos服务发现与注册 -->  
 <dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-alibaba-nacos-discovery</artifactId>  
 </dependency>  
 </dependencies>  
  
  
</project>

#### 2.2.2.2、bootstrap.yml

server:  
 port: 28082  
  
nacos:  
 server-addr: 127.0.0.1  
 namespace: ae89a87f-7f99-4a0a-b31e-e6e08703ffaa  
  
  
*#此处读取的dataId为nacos-config.yml，格式是${spring.cloud.nacos.config.prefix}-{spring.profile.active}.{spring.cloud.nacos.config.file-extension}  
#默认读取的${spring.application.name}-{spring.profile.active}.{spring.cloud.nacos.config.file-extension}*spring:  
 application:  
 name: nacos-consumer  
 cloud:  
 nacos:  
 config:  
 server-addr: ${nacos.server-addr}  
 file-extension: yml  
 namespace: ${nacos.namespace}  
 *# spring.cloud.nacos.config.prefix此属性可以配置nacos访问的配置文件前缀，默认是spring.applicaiton.name的值  
 # prefix: nacos-provider* prefix: nacos-consumer  
 discovery:  
 server-addr: ${nacos.server-addr}  
 namespace: ${nacos.namespace}

#### 2.2.2.3、NacosConsumerConfig

package com.lyj.nacosconsumer.config;  
  
import org.springframework.cloud.client.loadbalancer.LoadBalanced;  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
import org.springframework.web.client.RestTemplate;  
  
@Configuration  
public class NacosConsumerConfig {  
  
 @Bean  
 @LoadBalanced //ribbon使用  
 public RestTemplate restTemplate(){  
 return new RestTemplate();  
 }  
}

#### 2.2.2.4、NacosConsumerController

package com.lyj.nacosconsumer.api;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.cloud.context.config.annotation.RefreshScope;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.RestController;  
import org.springframework.web.client.RestTemplate;  
  
import java.util.HashMap;  
import java.util.Map;  
  
@RestController  
@RefreshScope  
public class NacosConsumerController {  
  
  
  
 @Autowired  
 private RestTemplate restTemplate;  
  
 @RequestMapping("consumer")  
 public String consumer(){  
  
 Map<String,Object> params = new HashMap<>();  
 params.put("username", "lyj");  
 params.put("password", "lyj");  
  
 String result = restTemplate.getForObject("http://nacos-provider/provider", String.class, params);  
  
 return result;  
 }  
}

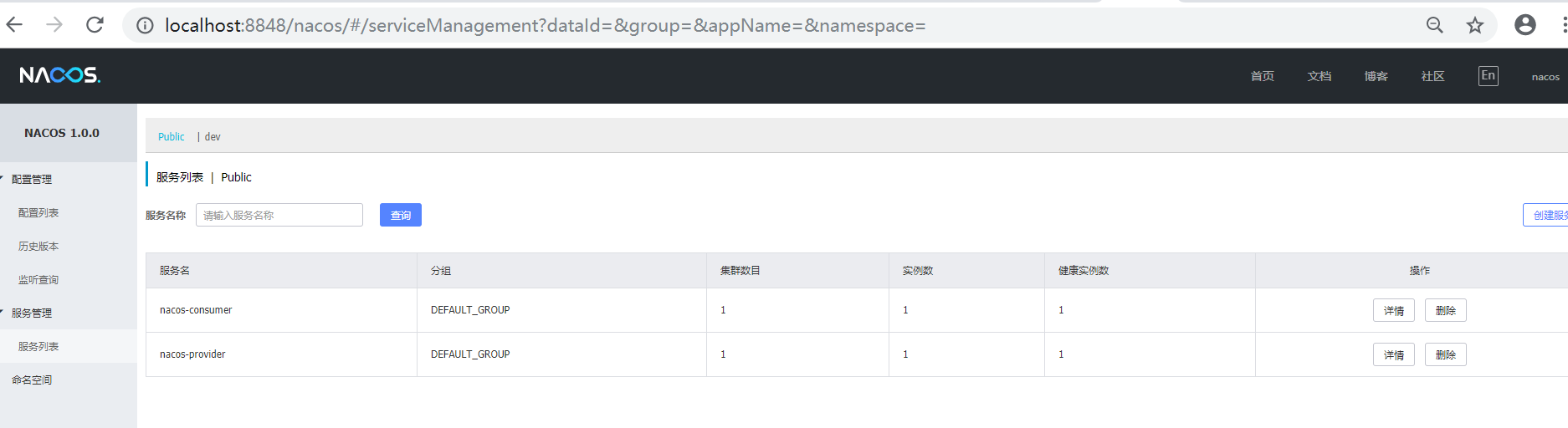
#### 2.2.2.5、NacosConsumerApplication

记得加上@EnableDiscoveryClient注解，将此服务注册到nacos注册中心

package com.lyj.nacosconsumer;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.cloud.client.discovery.EnableDiscoveryClient;  
  
@SpringBootApplication  
@EnableDiscoveryClient  
public class NacosConsumerApplication {  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(NacosConsumerApplication.class, args);  
 }  
  
}

### 2.2.3、nacos上查看是否已经注册服务上去

记得在项目中加上记得加上@EnableDiscoveryClient注解，才能将此服务注册到nacos注册中心



这里有个疑惑地方就是这里服务为什么在public命名空间中，明明配置在dev命名空间，有待探讨

### 2.2.4、访问

<http://localhost:28082/consumer>