链滴

spring cloud gateway 配置

作者：wenyl

原文链接：https://ld246.com/article/1631946022409

来源网站：链滴

nacos安装准备工作参考spring cloud整合nacos

1、pom引入依赖

主要是spring-cloud-starter-gateway

```
<?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>

    <groupId>cn.com.wenyl.alibaba</groupId>
    <artifactId>gateway</artifactId>
    <version>1.0-SNAPSHOT</version>

    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.0.5.RELEASE</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>

    <dependencyManagement>
        <dependencies>
            <dependency>
                <groupId>org.springframework.cloud</groupId>
                <artifactId>spring-cloud-dependencies</artifactId>
                <version>Finchley.SR1</version>
                <type>pom</type>
```

```
            <scope>import</scope>
        </dependency>
        <dependency>
            <groupId>org.springframework.cloud</groupId>
            <artifactId>spring-cloud-alibaba-dependencies</artifactId>
            <version>0.2.2.RELEASE</version>
            <type>pom</type>
            <scope>import</scope>
        </dependency>
    </dependencies>
</dependencyManagement>

<dependencies>
    <dependency>
        <groupId>org.springframework.cloud</groupId>
        <artifactId>spring-cloud-starter-alibaba-nacos-discovery</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.cloud</groupId>
        <artifactId>spring-cloud-starter-alibaba-nacos-config</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.cloud</groupId>
        <artifactId>spring-cloud-starter-gateway</artifactId>
    </dependency>
    <dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
        <version>1.18.2</version>
        <optional>true</optional>
    </dependency>
</dependencies>


<build>
    <plugins>
        <plugin>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-maven-plugin</artifactId>
        </plugin>
    </plugins>
</build>
</project>
```

2、配置文件

2.1、bootstrap.yml

```
spring:
  profiles:
    active: dev
  application:
    name: gateway
```

```
  cloud:
    nacos:
      discovery:
        server-addr: 127.0.0.1:8848
      config:
        server-addr: 127.0.0.1:8848
        group: DEFAULT_GROUP
        file-extension: yml
```

2.2、application.yml

```
server:
  port: 8080
management:
  endpoints:
    web:
      exposure:
        include: '*'
```

2.3、nacos配置信息


```
spring:
  cloud:
    gateway:
      discovery:
        locator:
          enabled: true
      routes:
        - id: nacos-provider
          uri: lb://nacos-provider
          predicates:
          - Path=/nacos-provider/**
          filters:
            - StripPrefix=1
        - id: service-consumer
          uri: lb://service-consumer
          predicates:
          - Path=/service-consumer/**
          filters:
            - StripPrefix=1
```

原文链接：spring cloud gateway 配置

2.4、配置规则

Spring Cloud Gateway通过路由谓词工厂来进行路径规则配置

2.4.1、After配置

after配置只有一个时间参数，用于匹配指定时间之后的请求

```
spring:
  cloud:
    gateway:
      routes:
      - id: after_route
        uri: https://example.org
        predicates:
        - After=2017-01-20T17:42:47.789-07:00[America/Denver]
```

2.4.2、Before配置

after配置只有一个时间参数，用于匹配指定时间之前的请求

```
spring:
  cloud:
    gateway:
      routes:
      - id: before_route
        uri: https://example.org
        predicates:
        - Before=2017-01-20T17:42:47.789-07:00[America/Denver]
```

2.4.3、Between配置

after配置有两个时间参数，用于匹配指定时间范围之间的请求

```
spring:
  cloud:
    gateway:
      routes:
      - id: between_route
        uri: https://example.org
        predicates:
        - Between=2017-01-20T17:42:47.789-07:00[America/Denver], 2017-01-21T17:42:47.789-
7:00[America/Denver]
```

2.4.4、Cookie配置

有两个参数，第一个为Cookie的name，第二个为匹配对应值的正则表达式，匹配符合条件的请求

```
spring:
  cloud:
    gateway:
      routes:
```

```
      - id: cookie_route
        uri: https://example.org
        predicates:
        - Cookie=chocolate, ch.p
```

2.4.5、Header配置

有两个参数，第一个为Header的name，第二个为匹配对应值的正则表达式，匹配符合条件的请求

```
spring:
  cloud:
    gateway:
      routes:
      - id: header_route
        uri: https://example.org
        predicates:
        - Header=X-Request-Id, \d+
```

2.4.6、Host配置

处理指定域名服务的请求

```
spring:
  cloud:
    gateway:
      routes:
      - id: host_route
        uri: https://example.org
        predicates:
        - Host=**.somehost.org,**.anotherhost.org
```

2.4.7、Method配置

通过HTTP方法来匹配

```
spring:
  cloud:
    gateway:
      routes:
      - id: method_route
        uri: https://example.org
        predicates:
        - Method=GET,POST
```

2.4.8、Path配置

通过请求路径匹配

```
spring:
  cloud:
    gateway:
      routes:
```

```
      - id: path_route
        uri: https://example.org
        predicates:
        - Path=/red/{segment},/blue/{segment}
```

2.4.9、Query配置

query用于包含指定参数值的请求，有两个参数，第一个是参数name，第二个是用于匹配值得正则达式

```
spring:
  cloud:
    gateway:
      routes:
      - id: query_route
        uri: https://example.org
        predicates:
        - Query=red, gree.
```

2.4.10、RemoteAddr配置

根据请求发起的IP进行拦截处理

```
spring:
  cloud:
    gateway:
      routes:
      - id: remoteaddr_route
        uri: https://example.org
        predicates:
        - RemoteAddr=192.168.1.1/24
```

2.4.11、Weight配置

根据权重进行配置，一共两个参数，第一个是组名，第二个是在改组中的占比（相加为10）

```
spring:
  cloud:
    gateway:
      routes:
      - id: weight_high
        uri: https://weighthigh.org
        predicates:
        - Weight=group1, 8
      - id: weight_low
        uri: https://weightlow.org
        predicates:
        - Weight=group1, 2
```

表示一个占比80%，一个占比20%