简介

Spring Cloud Bus 将分布式的节点用轻量的消息代理连接起来。它可以用于广播配置文件的更改或者服务之间的通讯，也可以用于监控，这篇主要讲述用Spring Cloud Bus实现通知微服务架构的配置文件的更改。

准备工作

1. 安装kafka
2. 启动kafka

一、改造config-server

1.在pom文件上加上依赖spring-cloud-starter-bus-kafka，及spring-boot-starter-actuator，pom文件如下：

```
<?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>

  <groupId>com.gewdata</groupId>
  <artifactId>config-server</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>jar</packaging>

  <name>config-server</name>
  <description>Demo project for Spring Boot</description>

  <parent>
  <groupId>com.gewdata</groupId>
  <artifactId>springCloudConfigDemo</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  </parent>

  <dependencies>
  <dependency>
```

```
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-bus-kafka</artifactId>
</dependency>

<dependency>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
</dependency>

<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-web</artifactId>
</dependency>
<dependency>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-config-server</artifactId>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
</dependencies>

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


</project>
```

2.在配置文件中加上kafka的配置，并且加上spring.cloud.bus配置：

```
spring.application.name=config-server
server.port=8888

# 配置git仓库地址
spring.cloud.config.server.git.uri=http://139.159.143.78:10086/wangjunyao/SpringCloudDemo.git
```

```
# 配置仓库路径
spring.cloud.config.server.git.searchPaths=config
# 配置仓库分支
spring.cloud.config.label=master
# 访问git仓库用户名
spring.cloud.config.server.git.username=wangjunyao
# 访问git仓库用户密码
spring.cloud.config.server.git.password=12345678
# 服务注册地址
eureka.client.serviceUrl.defaultZone=http://localhost:8889/eureka/

#消息总线配置
spring.cloud.bus.enabled=true
spring.cloud.bus.trace.enabled=true
management.endpoints.web.exposure.include=*
#Kafka的服务端列表，默认localhost
spring.cloud.stream.kafka.binder.brokers=132.232.111.101:9092
#Kafka服务端的默认端口，当brokers属性中没有配置端口信息时，就会使用这个默认端口，默认9092
spring.cloud.stream.kafka.binder.defaultBrokerPort=9092
#ZooKeeper节点的默认端口，当zkNodes属性中没有配置端口信息时，就会使用这个默认端口，默认2181
spring.cloud.stream.kafka.binder.defaultZkPort=2181
```

二、改造config-client

1.修改pom文件：

```
<?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>com.gewdata</groupId>
  <artifactId>config-client</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>jar</packaging>

  <name>config-client</name>
  <description>Demo project for Spring Boot</description>
```

```
13
14  <parent>
15  <groupId>com.gewdata</groupId>
16  <artifactId>springCloudConfigDemo</artifactId>
17  <version>0.0.1-SNAPSHOT</version>
18  </parent>
19
20  <dependencies>
21  <dependency>
22  <groupId>org.springframework.cloud</groupId>
23  <artifactId>spring-cloud-starter-bus-kafka</artifactId>
24  </dependency>
25
26  <dependency>
27  <groupId>org.springframework.cloud</groupId>
28  <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
29  </dependency>
30
31  <dependency>
32  <groupId>org.springframework.boot</groupId>
33  <artifactId>spring-boot-starter-web</artifactId>
34  </dependency>
35  <dependency>
36  <groupId>org.springframework.cloud</groupId>
37  <artifactId>spring-cloud-starter-config</artifactId>
38  </dependency>
39  <dependency>
40  <groupId>org.springframework.boot</groupId>
41  <artifactId>spring-boot-starter-actuator</artifactId>
42  </dependency>
43  </dependencies>
44
45  <build>
46  <plugins>
47  <plugin>
48  <groupId>org.springframework.boot</groupId>
49  <artifactId>spring-boot-maven-plugin</artifactId>
50  </plugin>
51  </plugins>
52  </build>
53 </project>
```

2.修改配置文件bootstrap.properties：

```
# 这里的配置是和git上文件名相对应的
spring.application.name=wjy-client
# 指明分支
spring.cloud.config.label=master
spring.cloud.config.profile=dev
# 指明配置服务中心网址
#spring.cloud.config.uri= http://localhost:8888/
server.port=8881

 # 高可用
 # 指定服务注册地址
 eureka.client.serviceUrl.defaultZone=http://localhost:8889/eureka/
 # 是从配置中心读取文件
 spring.cloud.config.discovery.enabled=true
 # 配置中心的servieId，即是服务名
 spring.cloud.config.discovery.serviceId=config-server

 #消息总线配置
 spring.cloud.bus.enabled=true
 spring.cloud.bus.trace.enabled=true
 management.endpoints.web.exposure.include=bus-refresh
 #Kafka的服务端列表，默认localhost
 spring.cloud.stream.kafka.binder.brokers=132.232.111.101:9092
 #Kafka服务端的默认端口，当brokers属性中没有配置端口信息时，就会使用这个默认端口，默认9092
 spring.cloud.stream.kafka.binder.defaultBrokerPort=9092
 #ZooKeeper节点的默认端口，当zkNodes属性中没有配置端口信息时，就会使用这个默认端口，默认2181
 spring.cloud.stream.kafka.binder.defaultZkPort=2181
```

3.在调用到@Value方法的类上添加注解@RefreshScope：

```
package com.gewdata;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.context.config.annotation.RefreshScope;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

```

```java
@SpringBootApplication
@RestController
@RefreshScope // 刷新配置
public class ConfigClientApplication {

 public static void main(String[] args) {
 SpringApplication.run(ConfigClientApplication.class, args);
 }

 @Value("${msg}")
 String msg;

 @RequestMapping(value = "/hi")
 public String hi(){
 return msg;
 }
}
```

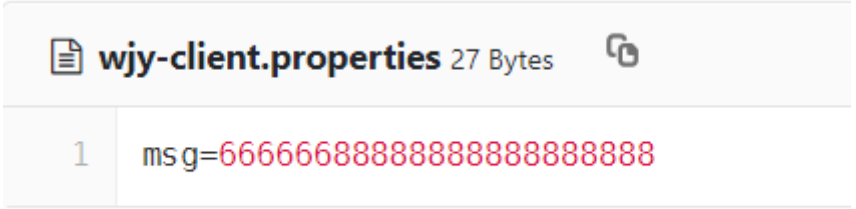
三、调试

1. 启动eureka-server
2. 启动config-server
3. 启动config-client
4. 通过查看kafka的topic命令，我们可以看到已经自定义生成了一条topic（topic也可以自定义）：

```
bin/kafka-topics.sh --describe --zookeeper localhost:2181
```

```
Topic:springCloudBus    PartitionCount:1        ReplicationFactor:1     Configs:
        Topic: springCloudBus   Partition: 0    Leader: 0       Replicas: 0     Isr: 0
```

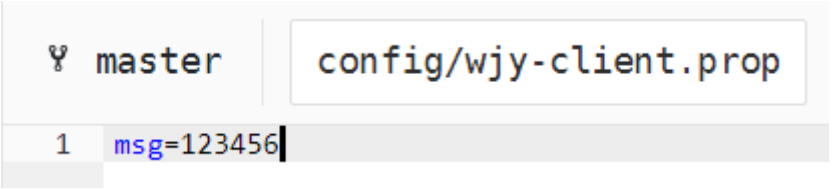
5. 此时git中配置文件内容为：



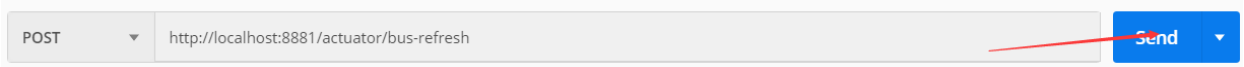
访问http://localhost:8881/hi:

```
666666888888888888888
```

6. 修改git中配置内容：



7. 使用post方式请求http://localhost:8881/actuator/bus-refresh，刷新配置：



8. 再次访问http://localhost:8881/hi:

```
123456
```

Spring Cloud2.x版本踩坑

1.必须在config-server和config-client中pom文件添加以下依赖：

```
<dependency>
 <groupId>org.springframework.boot</groupId>
 <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
<dependency>
 <groupId>org.springframework.cloud</groupId>
 <artifactId>spring-cloud-starter-bus-kafka</artifactId>
</dependency>
```

2.必须在在config-server和config-client配置文件中分别添加以下配置：

```
management:
 endpoints:
 web:
 exposure:
 include: bus-refresh
```

3.必须在调用@Value注解类上添加注解：

```
@RefreshScope
```

4.请求刷新的页面由原来1.5.x的localhost:8881/bus/refresh，变成：

http://localhost:8881/actuator/bus-refresh