链滴

Spring-Cloud 系列第 6 篇：spring-cloud-bus

作者：xjtushilei

原文链接：https://ld246.com/article/1497580049008

来源网站：链滴

自学spring-cloud系列，越来越感觉spring-cloud很强大！

主要分为以下几篇：

1. spring-cloud-config: 分布式配置管理
2. spring-cloud-eureka: 服务注册与发现
3. spring-cloud-eureka-consumer: 远程服务调用和及其负载均衡
4. spring-cloud-Hystrix: 熔断器保证服务高可用
5. spring-cloud-config-eureka-ribbon: 分布式配置管理的高可用
6. spring-cloud-bus: 配置信息的实时更新
7. spring-cloud-zuul: 网关基础服务

介绍

在之前的Spring Cloud Config的介绍中，我们还留了一个悬念：如何实现对配置信息的实时更新。

通过/refresh接口和Git仓库的Web Hook来实现Git仓库中的内容修改触发应用程序的属性更新这是个好的方法，但是我没讲，因为：若所有触发操作均需要我们手工去维护Web Hook中的应用位置的，这随着系统的不断扩张，会变的越来越难以维护，而消息代理中间件是解决该问题最为合适的方案是否还记得我们在介绍消息代理中的特点时有提到过这样一个功能：消息代理中间件可以将消息路由一个或多个目的地。利用这个功能，我们就能完美的解决该问题，下面我们来说说Spring Cloud Bus的具体实现方案。

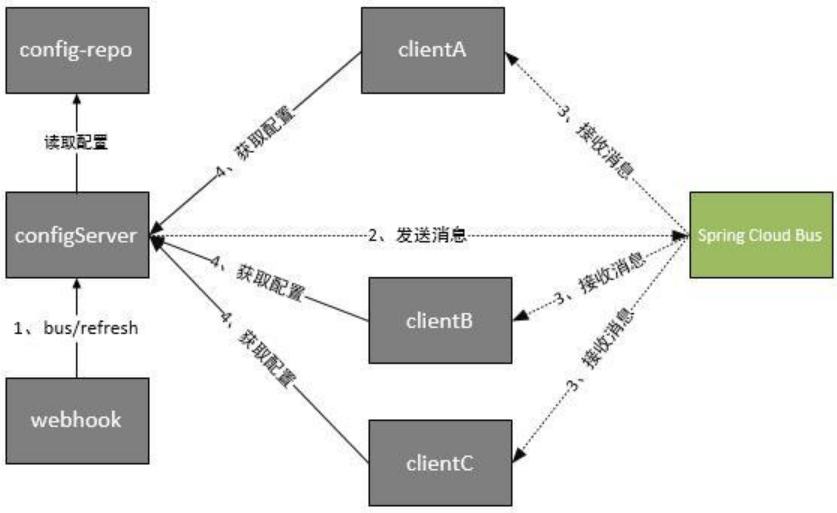
本次使用 Rabbit MQ 来进行实验，没有使用其他消息队列。

安装 Rabbit MQ

可以参考我的文章 RabbitMQ 的安装和使用

其用户名和密码是： guest/guest

架构



这时Spring Cloud Bus做配置更新步骤如下:

- 1、提交代码,利用 git 的 webhook 触发post请求给 bus/refresh
- 2、server端接收到请求并发送给Spring Cloud Bus
- 3、Spring Cloud bus接到消息并通知给其它客户端
- 4、其它客户端接收到通知，请求Server端获取最新配置
- 5、全部客户端均获取到最新的配置

这样的话我们在server端的代码做一些改动，来支持bus/refresh

开始之前:

我想说，项目源码使用模块编程，因为之前的版本原因，client端注册到了eureka服务端，所以启动目应该也要启动eureka的服务端。然后在启动config服务端，最后启动我们的config-client端，启动onfig-client要使用profile方法启动，这个之前我们已经使用了好多次。

修改 server 端

pom 增加如下:

```
<!--消息总线-->
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-bus-amqp</artifactId>
</dependency>
<!--config-->
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-config-monitor</artifactId>
</dependency>
```

其中bus-amqp就是消息总线，用来利用 rabbitmq 来进行消息通讯。monnitor就是在server 端设一个监听，来接收我们发送的post信息，告诉它我们更新了配置，你可以开始刷新已经注入的值了。

配置文件:

```
management.security.enabled=false

spring.rabbitmq.host=localhost
spring.rabbitmq.port=5672
spring.rabbitmq.username=guest
spring.rabbitmq.password=guest
```

management.security.enabled 可以将刷新设置密码取消，因为我们是在测验，所以不需要密码。果是你们公司内网的话，有物理隔离，应该也不需要密码。

client 端

讲道理，client 端不需要做什么修改

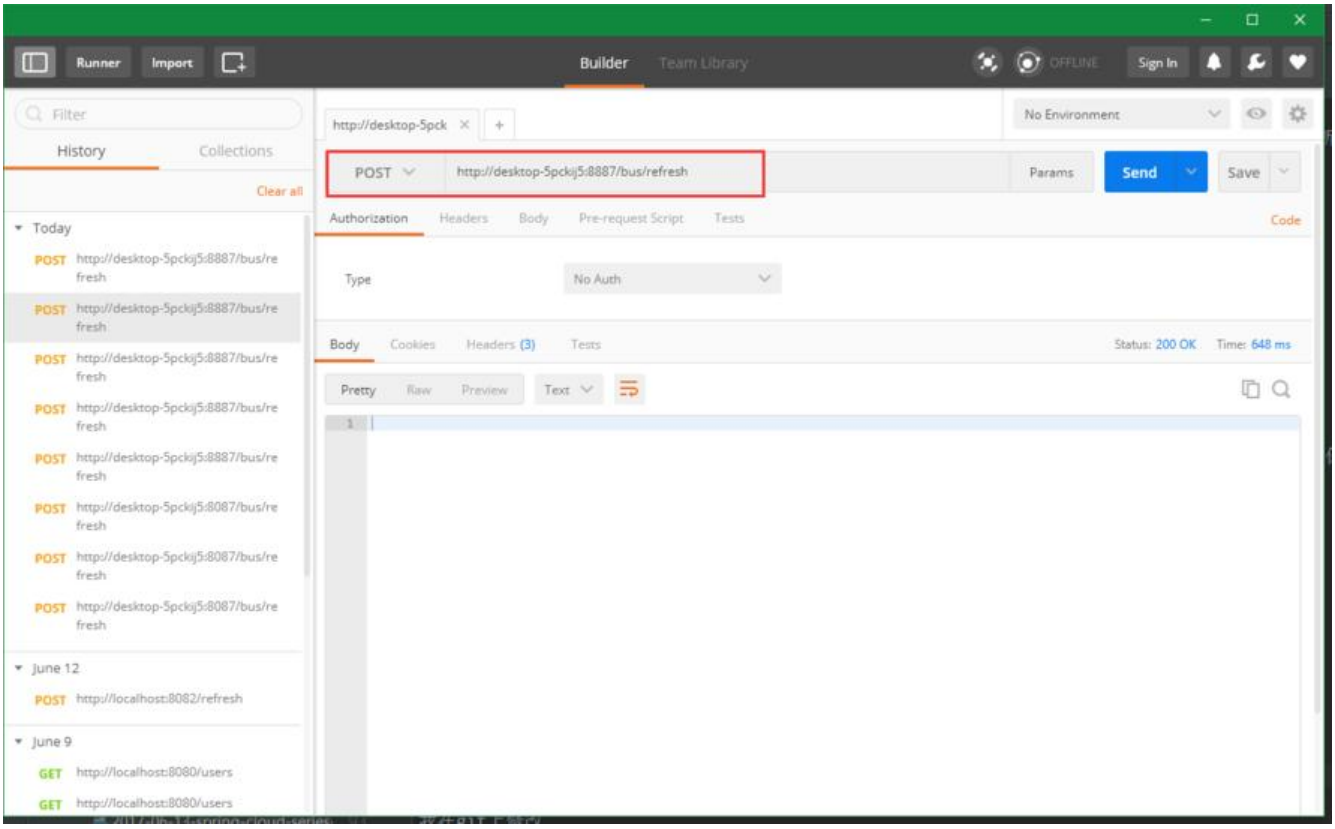
management.security.enabled=false 如果真的需要的话，加上这个，防止不能够刷新。

测试

访问 http://localhost:8087/hi 得到 123

修改 git 里文件的值为2222222222

github是有webhook的，你们公司私人的git应该也有webhook，但是github的webhook不能访问我ocalhost呀，所以这里利用postman 进行刷新操作来模拟webhook



重新访问http://localhost:8087/hi 得到 2222222222

同时，查看日志，发现值已经刷新，并且进行重新注册到了服务中心。

感觉，很酷哦！100 酷！

示例源码

所有源码在我的github仓库里，传送门：https://github.com/xjtushilei/spring-cloud-simples.git

支持

如果你喜欢~ 给个星