[**第八篇: 消息总线(Spring Cloud Bus)**](http://blog.csdn.net/forezp/article/details/70148235)

参考：<http://blog.csdn.net/forezp/article/details/70148235>

转载请标明出处：   
<http://blog.csdn.net/forezp/article/details/70148235>   
本文出自[方志朋的博客](http://blog.csdn.net/forezp)

转载请标明出处：

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。本文要讲述的是用Spring Cloud Bus实现通知微服务架构的配置文件的更改。

一、准备工作

本文还是基于上一篇文章来实现。按照官方文档，我们只需要在配置文件中配置 spring-cloud-starter-bus-amqp ；这就是说我们需要装rabbitMq，点击[rabbitmq](http://www.rabbitmq.com/)下载。至于怎么使用 rabbitmq，搜索引擎下。

二、改造config-client

在pom文件加上起步依赖spring-cloud-starter-bus-amqp，完整的配置文件如下：

<?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.forezp</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>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>1.5.2.RELEASE</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.retry</groupId>

<artifactId>spring-retry</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-config</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-aop</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-eureka</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-bus-amqp</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-actuator</artifactId>

</dependency>

</dependencies>

<dependencyManagement>

<dependencies>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-dependencies</artifactId>

<version>Dalston.RC1</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>

<repositories>

<repository>

<id>spring-milestones</id>

<name>Spring Milestones</name>

<url>https://repo.spring.io/milestone</url>

<snapshots>

<enabled>false</enabled>

</snapshots>

</repository>

</repositories>

</project>

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54
* 55
* 56
* 57
* 58
* 59
* 60
* 61
* 62
* 63
* 64
* 65
* 66
* 67
* 68
* 69
* 70
* 71
* 72
* 73
* 74
* 75
* 76
* 77
* 78
* 79
* 80
* 81
* 82
* 83
* 84
* 85
* 86
* 87
* 88
* 89
* 90
* 91
* 92
* 93
* 94
* 95
* 96
* 97
* 98
* 99
* 100
* 101
* 102
* 103

在配置文件application.properties中加上RabbitMq的配置，包括RabbitMq的地址、端口，用户名、密码，代码如下：

spring.rabbitmq.host=localhost

spring.rabbitmq.port=5672

# spring.rabbitmq.username=

# spring.rabbitmq.password=

* 1
* 2
* 3
* 4
* 5
* 6

如果rabbitmq有用户名密码，输入即可。

依次启动eureka-server、confg-cserver,启动两个config-client，端口为：8881、8882。

访问<http://localhost:8881/hi> 或者<http://localhost:8882/hi> 浏览器显示：

foo version 3

这时我们去[代码仓库](https://github.com/forezp/SpringcloudConfig/blob/master/respo/config-client-dev.properties)将foo的值改为“foo version 4”，即改变配置文件foo的值。如果是传统的做法，需要重启服务，才能达到配置文件的更新。此时，我们只需要发送post请求：<http://localhost:8881/bus/refresh>，你会发现config-client会重现肚脐配置文件

重新读取配置文件：

这时我们再访问<http://localhost:8881/hi> 或者<http://localhost:8882/hi> 浏览器显示：

foo version 4

另外，/bus/refresh接口可以指定服务，即使用”destination”参数，比如 “/bus/refresh?destination=customers:\*\*” 即刷新服务名为customers的所有服务，不管ip。

三、分析

此时的架构图： 

当git文件更改的时候，通过pc端用post 向端口为8882的config-client发送请求/bus/refresh／；此时8882端口会发送一个消息，由消息总线向其他服务传递，从而使整个微服务集群都达到更新配置文件。

四、其他扩展（可忽视）

可以用作自定义的Message Broker,只需要spring-cloud-starter-bus-amqp, 然后再配置文件写上配置即可，同上。

Tracing Bus Events：   
需要设置：spring.cloud.bus.trace.enabled=true，如果那样做的话，那么Spring Boot TraceRepository（如果存在）将显示每个服务实例发送的所有事件和所有的ack,比如：（来自官网）

{

"timestamp": "2015-11-26T10:24:44.411+0000",

"info": {

"signal": "spring.cloud.bus.ack",

"type": "RefreshRemoteApplicationEvent",

"id": "c4d374b7-58ea-4928-a312-31984def293b",

"origin": "stores:8081",

"destination": "\*:\*\*"

}

},

{

"timestamp": "2015-11-26T10:24:41.864+0000",

"info": {

"signal": "spring.cloud.bus.sent",

"type": "RefreshRemoteApplicationEvent",

"id": "c4d374b7-58ea-4928-a312-31984def293b",

"origin": "customers:9000",

"destination": "\*:\*\*"

}

},

{

"timestamp": "2015-11-26T10:24:41.862+0000",

"info": {

"signal": "spring.cloud.bus.ack",

"type": "RefreshRemoteApplicationEvent",

"id": "c4d374b7-58ea-4928-a312-31984def293b",

"origin": "customers:9000",

"destination": "\*:\*\*"

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31

本文源码下载：   
<https://github.com/forezp/SpringCloudLearning/tree/master/chapter8>

五、参考资料

[spring\_cloud\_bus](http://projects.spring.io/spring-cloud/spring-cloud.html#_spring_cloud_bus)