**史上最简单的SpringCloud教程 | 第六篇: 分布式配置中心(Spring Cloud Config)**

2017年04月10日 21:18:06

阅读数：304050

转载请标明出处：   
<http://blog.csdn.net/forezp/article/details/70037291>   
本文出自[方志朋的博客](http://blog.csdn.net/forezp)

* 最新版本：
* [史上最简单的SpringCloud教程 | 第六篇: 分布式配置中心(Spring Cloud Config)(Finchley版本)](http://blog.csdn.net/forezp/article/details/81041028)

在上一篇文章讲述zuul的时候，已经提到过，使用配置服务来保存各个服务的配置文件。它就是Spring Cloud Config。

**一、简介**

在分布式系统中，由于服务数量巨多，为了方便服务配置文件统一管理，实时更新，所以需要分布式配置中心组件。在Spring Cloud中，有分布式配置中心组件spring cloud config ，它支持配置服务放在配置服务的内存中（即本地），也支持放在远程Git仓库中。在spring cloud config 组件中，分两个角色，一是config server，二是config client。

**二、构建Config Server**

创建一个spring-boot项目，取名为config-server,其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>

<groupId>com.forezp</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>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.cloud</groupId>

<artifactId>spring-cloud-config-server</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-eureka</artifactId>

</dependency>

</dependencies>

<dependencyManagement>

<dependencies>

<dependency>

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

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

<version>Camden.SR6</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>

在程序的入口Application类加上@EnableConfigServer注解开启配置服务器的功能，代码如下：

@SpringBootApplication

@EnableConfigServer

public class ConfigServerApplication {

public static void main(String[] args) {

SpringApplication.run(ConfigServerApplication.class, args);

}

}

需要在程序的配置文件application.properties文件配置以下：

spring.application.name=config-server

server.port=8888

spring.cloud.config.server.git.uri=https://github.com/forezp/SpringcloudConfig/

spring.cloud.config.server.git.searchPaths=respo

spring.cloud.config.label=master

spring.cloud.config.server.git.username=your username

spring.cloud.config.server.git.password=your password

* spring.cloud.config.server.git.uri：配置git仓库地址
* spring.cloud.config.server.git.searchPaths：配置仓库路径
* spring.cloud.config.label：配置仓库的分支
* spring.cloud.config.server.git.username：访问git仓库的用户名
* spring.cloud.config.server.git.password：访问git仓库的用户密码

如果Git仓库为公开仓库，可以不填写用户名和密码，如果是私有仓库需要填写，本例子是公开仓库，放心使用。

远程仓库<https://github.com/forezp/SpringcloudConfig/> 中有个文件config-client-dev.properties文件中有一个属性：

foo = foo version 3

启动程序：访问<http://localhost:8888/foo/dev>

{"name":"foo","profiles":["dev"],"label":"master",

"version":"792ffc77c03f4b138d28e89b576900ac5e01a44b","state":null,"propertySources":[]}

证明配置服务中心可以从远程程序获取配置信息。

http请求地址和资源文件映射如下:

* /{application}/{profile}[/{label}]
* /{application}-{profile}.yml
* /{label}/{application}-{profile}.yml
* /{application}-{profile}.properties
* /{label}/{application}-{profile}.properties

**三、构建一个config client**

重新创建一个springboot项目，取名为config-client,其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.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.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-test</artifactId>

<scope>test</scope>

</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>

其配置文件**bootstrap.properties**：

spring.application.name=config-client

spring.cloud.config.label=master

spring.cloud.config.profile=dev

spring.cloud.config.uri= http://localhost:8888/

server.port=8881

* spring.cloud.config.label 指明远程仓库的分支
* spring.cloud.config.profile
  + dev开发环境配置文件
  + test测试环境
  + pro正式环境
* spring.cloud.config.uri= <http://localhost:8888/> 指明配置服务中心的网址。

程序的入口类，写一个API接口“／hi”，返回从配置中心读取的foo变量的值，代码如下：

@SpringBootApplication

@RestController

public class ConfigClientApplication {

public static void main(String[] args) {

SpringApplication.run(ConfigClientApplication.class, args);

}

@Value("${foo}")

String foo;

@RequestMapping(value = "/hi")

public String hi(){

return foo;

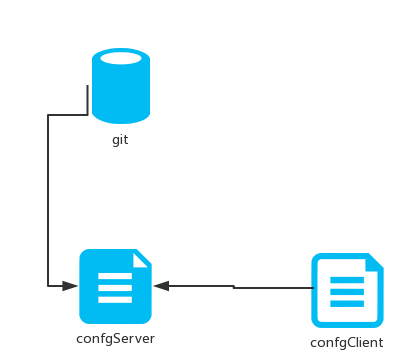
}

}

打开网址访问：<http://localhost:8881/hi>，网页显示：

foo version 3

这就说明，config-client从config-server获取了foo的属性，而config-server是从git仓库读取的,如图：



本文源码下载：   
<https://github.com/forezp/SpringCloudLearning/tree/master/chapter6>

**四、参考资料**

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