2017年6月12日 星期一

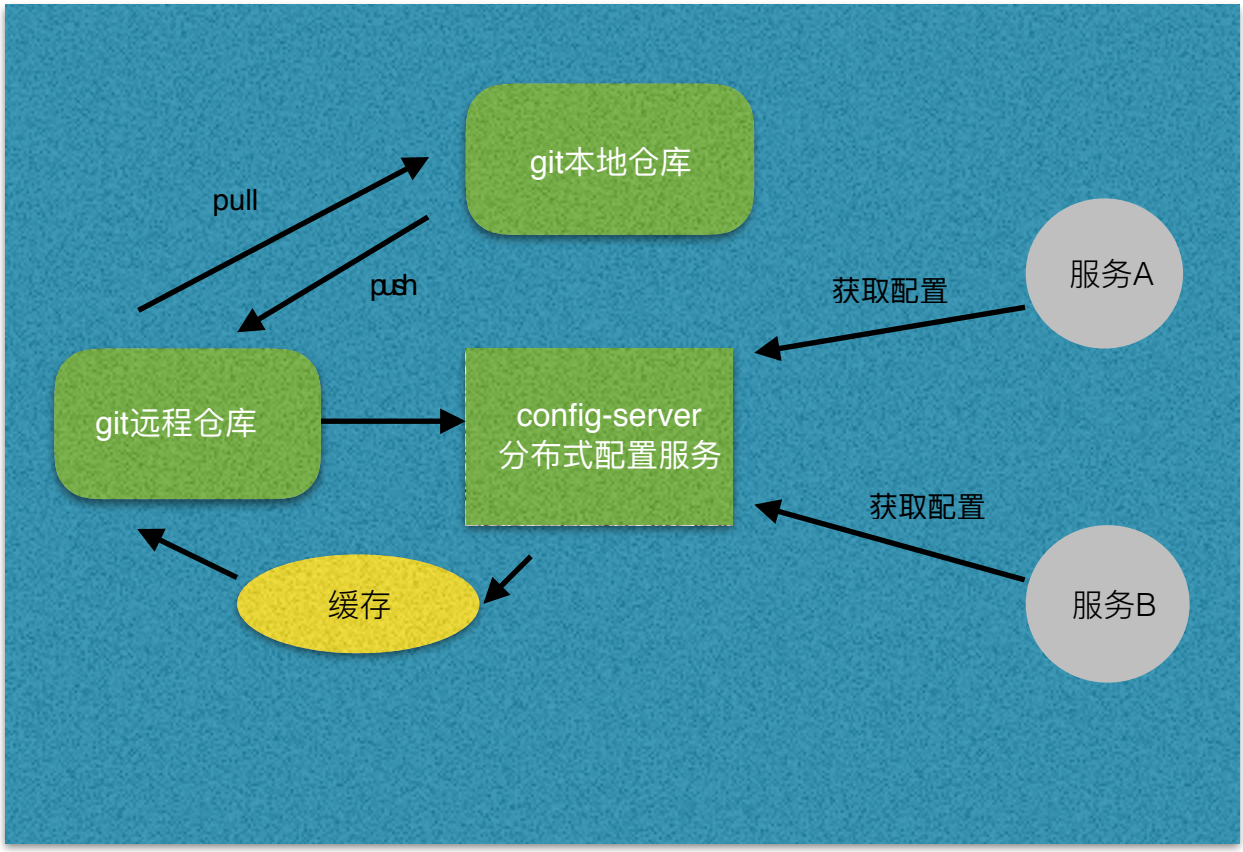
分布式配置中心之SpringCloud-Config

author： hushuang
email ： hd1611756908@gmail.com
adress： www.dajiangdahe.com

说说分布式配置中心？没有做过分布式开发的最好是先了解一下分布式开发，要不然学习springcloud会很麻烦，要是用过Dubbo框架的人，入门SpringCloud会很简单，SpringCloud就是将一个重量级的分布式框架（类似于Dubbo）的，将其中的各个组件拆分开了，然后在这个基础上添加一些没有的组件，让分布式开发更简单快捷而又不让框架十分的笨重。

这章要说的就是分布式配置中心spring cloud config：

说一下分布式配置的实现原理，我这里通过一个图示来说明：



通过上图来说明分布式配置中心的工作原理，这里介绍使用github远程仓库存储配置信息

创建一个分布式配置中心的步骤：这里介绍单点配置，本章不介绍高可用配置中心的搭建，想构建一个高可用的配置中心，请参考Eureka自行搭建。

一、构建项目前准备工作。

1.创建本地GIT仓库：在自己的电脑上创建一个本地的GIT仓库

>>$ mkdir config-repo　　—>创建文件夹

>>$ git init　　—>初始化成git仓库

2.创建一个文件夹放置配置文件

>>$ mkdir con_repo

3.将配置文件夹添加进本地缓存区

>>$ git add con_repo

4.将本地缓存区文件提交到本地仓库

在这里要首先创建配置文件，按照格式{aaaa}-{bbbb}.properties将配置文件放到con_repo文件夹下。配置文件里面添加几个自定义属性

config-dev.properties

```
from=git-dev-1.0
key=value-dev-1.0
```

config-prod.properties

```
from=git-prod-1.0
key=value-prod-1.0
```

config-test.properties

```
from=git-test-1.0
key=value-test-1.0
```

config.properties

```
from=git-default-1.0
key=value-default-1.0
```

>>$ git commit config*.properties -m “配置文件提交”

5.将本地git库推送到远程的github上，我这里用github作为远程仓库，所以你得有github账号。

>>$ git push [remote] [branch]

remote：远程仓库地址　　branch：远程仓库分支

以上为创建分布式配置服务的准备工作。接下来才是构建分布式服务配置中心项目。

二、分布式配置服务分时服务端和客户端两部分。

1.分布式配置服务服务端构建步骤：

—>创建config-server项目，并添加依赖 spring-cloud-config-server

—>创建分布式服务入口启动类SpringCloudConfigServer

```
package com.im.sc;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.config.server.EnableConfigServer;

@EnableConfigServer    //启动分布式配置服务注解
@SpringBootApplication
public class SpringCloudConfigServer {
        public static void main(String[] args) {
                SpringApplication.run(SpringCloudConfigServer.class, args);
        }

}
```

—>分布式配置服务配置项application.yml

```
server:
  port: 7001  #服务端口号
spring:
  application:
    name: config-server  #应用名称
  cloud:
    config:
      label: master #git分支
      server:
        git:
          uri: https://github.com/hd1611756908/config-repo #git仓库地址
          search-paths: con_repo    #git仓库下面的文件夹，可以是多个用逗号分隔
          username: admin@163.com    #github 用户名
          password: 111111         #github密码
```

2.启动config-server服务

通过curl在命令行访问config-server服务

curl http://localhost:7001/config/dev/master

解释一下这个url访问路径：

http://localhost:7001 ： 这块我就不说了就是分布式配置服务的地址

config：这个是什么？还记不记得上面我们自定义的几个提交到github上的配置文件

还记得我说的配置文件的定义格式为{application}-{env}.properties,一般都会定义为服务名称-环境标志,比如a服务的dev环境 a-dev.properties

所以http://localhost:7001/config/dev/地址里面的/config/dev/分别为下面配置文件的 - 左右两边的标志。最后一个master是什么呢？这就更简单了，我们使用git一般都会选择分支，这个就是看你git仓库文件选择放在哪个分支了，我的放在默认分支，所以是master

—> config-dev.properties

—> config-prod.properties

—> config-test.properties

—> config.properties

三、客户端验证：config-client

1.创建config-client项目并且添加依赖

```
spring-boot-starter-web
spring-cloud-starter-config
```

2.添加配置信息bootstrap.properties

```
spring.application.name=config
spring.cloud.config.profile=dev
spring.cloud.config.label=master
spring.cloud.config.uri=http://localhost:7001/
server.port=7002
```

为什么配置文件名称为bootstrap.properties，因为bootstrap.properties配置文件比

application.properties文件先被加载并被初始化，要问我为什么，我只能说自己看源码去，spring这么定的，为什么要放在bootstrap.properties中，因为这些数据要比较早的被初始化，所以放在这个配置文件中。

解说bootstrap.properties配置信息

`spring.application.name=config` ： 首先就是这个名称不是随便写的，这个名称是什么呢？？这就得回忆了，还记得不记得上面`config-server`服务端定义的几个配置文件

—> config-dev.properties

—> config-prod.properties

—> config-test.properties

—> config.properties

对就是这几个配置文件，spring.application.name=的名称就是上面配置文件的开头属性

```
spring.cloud.config.profile=dev   : dev和上面的配置文件一一对应(config-dev.properties)
spring.cloud.config.label=master
spring.cloud.config.uri=http://localhost:7001/
server.port=7002
```

这几个就不用说了，一个是分支，一个是分布式服务`config-server`的请求地址还有一个就是分布式服务客户端的端口号

以上就是分布式配合的配置中心，以及配置的获取。