# Spring Cloud分布式配置中心Config

在实际工作中，我们经常会遇到不同环境下处理配置文件的问题，最常见的可能就是数据库连接了。开发、测试、生产环境使用不同的口令访问数据库不同的地址。

我在工作这些年，待的一些公司大概是两种方式来处理：

1. 开发人员一肩挑，开发测试生产的配置都有，打包的时候先配好。

比如我们经常在配置文件里这么写：

#dev

db.ip:192.168.100.111

#test

#db.ip:192.168.100.112

#prod

#db.ip:192.168.100.113

打包之前改一改，扔到相应的环境里。

1. 开发、测试、运维各负其责。

那时候是开发发版，我们要写好部署文档，明确告知配置文件哪里要改。测试、运维人员拿到之后还要写个部署文档，第一步停服务，命令是什么，第二步备份，命令是什么……

写完之后我们还要审核，看有没有漏掉或错误的。

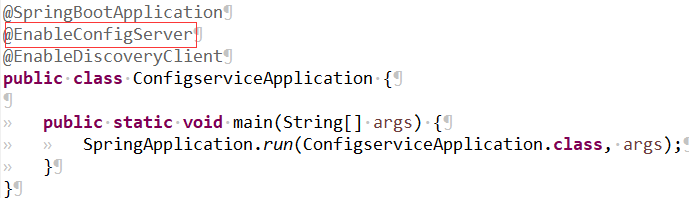
配置文件要么先从war包里拉出来，修改好了再放进去。要么是在Web容器里先运行解压，修改配置之后再启动容器。

对单体应用来说，这么做没啥问题，但是微服务这种，几十上百上千的，每个微服务再分开发、测试、生产环境，再像以前那么搞就要命了，可以使用Spring Cloud Config统一配置。

## Config Server

参考以前的文章构建服务端，依赖是Web、Eureka Discovery、Config Server。

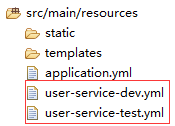
代码如下图：



只增加一个@EnableConfigServer。

config内容支持从git、svn、本地获取，本例采用本地文件。

在src-main-resources目录下增加如下两个配置文件：



这个配置文件的命名是有规则的，它决定了Http访问这些配置内容的URL，本文使用的是/**{application}-{profile}.yml**这种方式，对应的URL就是<http://IP:PORT/application/profile>。

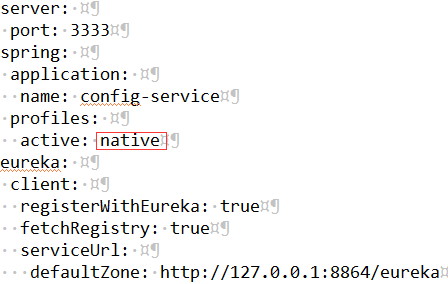
详细的规则网上一搜全是，有想用其他方式的自己找一下。

dev的内容如下：



test里面改成test。记住这个配置项的Key是**userservice.env**。

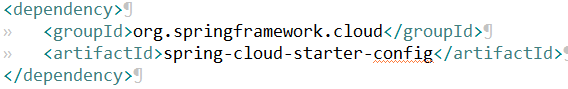
application.yml：



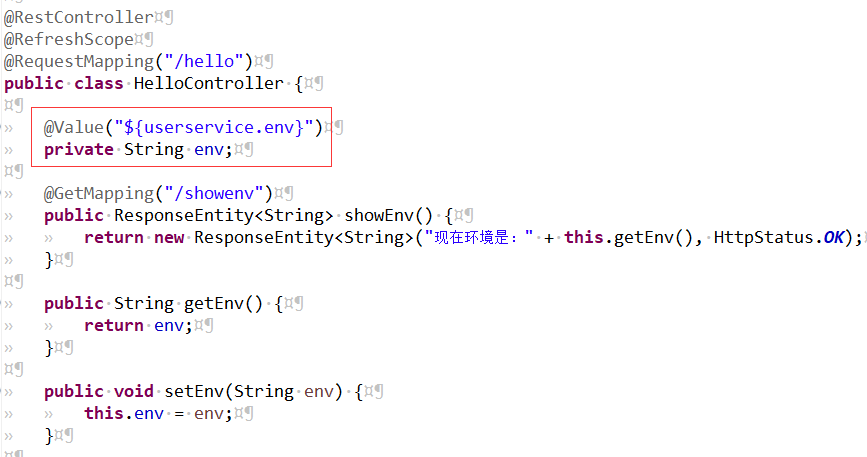
红框代表告知Config Server从本地文件获取配置内容。从SVN、GIT上获取请查阅官方文档或相关教程。

## Config Client

使用我之前文章写User Service工程，pom添加以下依赖：



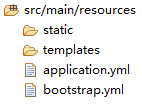
在Controller中增加如下注解和代码：



@Refresh这个注解后文讲。

红框就是我们要取得配置项的Key。

增加bootstrap.yml，因为配置项要先于application.yml加载，所以内容要放在这个文件里。

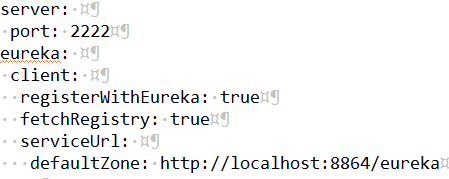


bootstrap.yml内容如下：



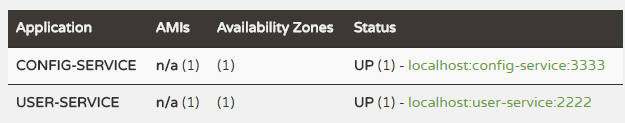
注意两个红框，决定了我们要从Config Server里获取哪个配置文件。

application.yml：

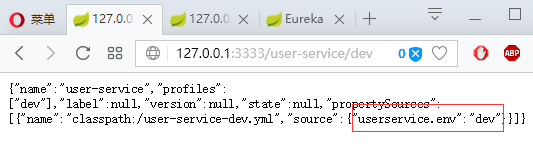


## 结果

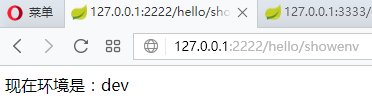
依次启动Eureka注册中心、Config Server、User Service：



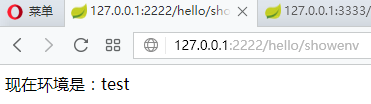
先查看Config Server里的配置：



访问User Service的URL结果如下：



将bootstrap.yml里的dev改成test，重启服务，配置项的内容变了：



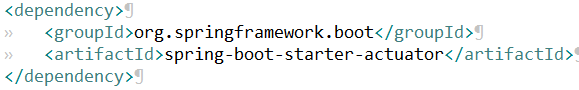
## 配置重载

先说明一下，为实现这一功能我费了好长时间，因为我用的版本是2.0.0.M3，而现在官方代码、文档以及网上你能搜到的参考资料都是1.5.X的，2.0发生了一些变化，下文我会说一下。

我们都知道，配置项一般是项目启动时加载到内存里，如果运行中配置项发生了变化，我们一般是重启服务，这就太不好了，所以我们需要重加载功能。

这一功能实现在客户端。

pom.xml增加以下依赖：



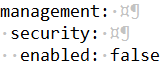
actuator是个好东西，抽空我写一篇讲讲。

还记得在Controller加的注解@RefreshScope吗？这个注解功能就是说这个Bean会被重新加载。所以在实际开发中，配置项都要有个专门的类来存放，这个类就可以加上这个注解。本例为了方便就直接写在Controller了。

如何触发这个重加载动作呢？官方文档以及网上资料都说：POST请求<http://127.0.0.1:2222/refresh> 即可，这是1.5.X版本！2.0版本变成了<http://127.0.0.1:2222/application/refresh> 切记！

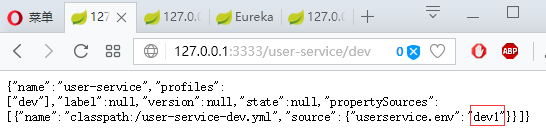
这个URL是需要权限的，我们先关闭校验。

userservice的application.yml增加：

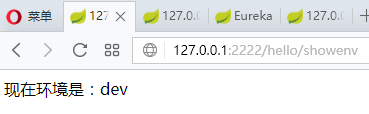


这时候修改Config Server里面的user-service-dev.yml，dev改成dev1。

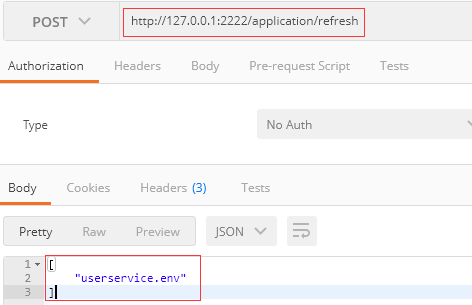
Config Server已经变了：



但是User Service还没有变：



使用Postman发送请求：



如果没有关闭校验，这里会显示401。

再看一下User Service：

