#### springcloud server config的git文件访问地址

http请求地址和资源文件映射如下:

/{application}/{profile}[/{label}]

/{application}-{profile}.yml

/{label}/{application}-{profile}.yml

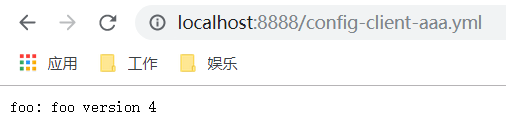
/{application}-{profile}.properties

/{label}/{application}-{profile}.properties

假设现在config-server的配置如下：



那么访问<https://github.com/qq1623299667/springcloudconfig/tree/master/config-client>/ config-client-bbb.yml的路径的http请求就是<http://localhost:8888/config-client-aaa.yml，server>会自动将文件进行解析成{application}-{profile}.yml，然后根据searchPaths: '{application}'的规则去到对应的应用名文件夹，然后去找到对应的文件。如果有打印出文件内容，那么说明server已经启动成功了

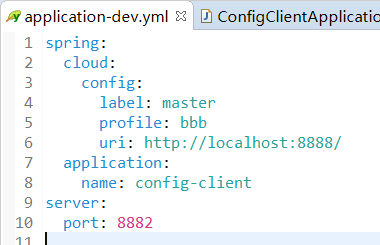


#### spring cloud config的使用感受

假设现在spring cloud config server的配置如下：



那么，一个指定了如下配置的spring cloud config client：



会自动去找服务器的文件为

<https://github.com/qq1623299667/springcloudconfig/tree/master/config-client>/ config-client-bbb.yml

1.前面的config-client是因为配置了searchPaths: '{application}'，所以不在根目录，而是去找叫spring.application.name的文件夹

2.后面的config-client-bbb.yml是springcloud config默认的查找路径：{application}-{profile}.yml

3. @RefreshScope注解写在任何位置，其他类的配置都会跟着改变，作用范围是全局

4.spring cloud config会并且只能在开启的时候拉起配置中心的配置，无法直接手动或动态刷新

并且从服务器拉取的配置会覆盖本地的端口号和参数，然后再启动

#### springcloud bus+springcloud config无需重启更新配置

先更新配置到git上，然后post调用<http://localhost:8882/actuator/bus-refresh>，等调用成功后再调用原来的接口<http://localhost:8882/hi>发现配置文件已经成功更新过来了。无需重启服务。另外，/actuator/bus-refresh接口可以指定服务，即使用"destination"参数，比如 “/actuator/bus-refresh?destination=customers:\*\*” 即刷新服务名为customers的所有服务。