**[学习微服务-第1天] ServiceComb+SpringCloud Zuul网关**

在微服务架构模式中后端服务的实例数一般是动态的，于客户端而言很难发现动态改变的服务实例的访问地址信息，服务网关能对用户提供统一的入口。

ServiceComb Java-Chassis 内置了网关服务EdgeService，开发者可以非常简单的搭建一个EdgeService服务。

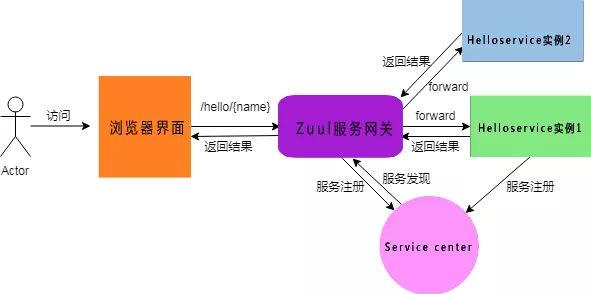
具体可参考：https://docs.servicecomb.io/java-chassis/zh\_CN/edge/by-servicecomb-sdk.html

本文将介绍ServiceComb与SpringCloud的Zuul网关组件协同工作，以构建微服务应用。ServiceComb在自身的处理链HandlerTrain中已完成Zuul的对接，用户用极简单的方法配置后即可使微服务应用具备网关服务的能力。

为使读者更好地理解，本文将编写一个简单的Hello微服务，并启动2个实例来进行演示。

Hello微服务提供hello/{name}接口，只需从前端输入参数name就可从后端微服务获取到程序员百看不厌的Hello world结果。

**微服务模式下的Hello应用模型**



**技术准备**

ServiceComb 作为后端微服务核心框架  
ServiceCenter 作为服务发现与注册中心  
SpringCloud Zuul 组件做服务网关

|  |  |
| --- | --- |
| **环境准备** |  |

以下环境为Windows 64位系统

●安装git，详情可参考git安装教程

https://git-scm.com/book/zh/v2/%E8%B5%B7%E6%AD%A5-%E5%AE%89%E8%A3%85-Git

●安装JDK 1.8，详情可参考JDK安装教程。

https://docs.oracle.com/javase/8/docs/technotes/guides/install/install\_overview.html

●安装Maven 3.x，详情可参考Maven安装教程

<https://maven.apache.org/install.html>

**ServiceCenter安装**

下载地址：

http://mirrors.hust.edu.cn/apache/servicecomb/servicecomb-service-center/1.1.0/

下载后解压如下



在该目录下双击service-center.exe即可启动，命令窗口中出现如下信息基本代表ServiceCenter启动成功，从这个信息也可以得知ServiceCenter监听的是30100端口，等下配置文件要用到。

https://mmbiz.qpic.cn/mmbiz_png/jaoydu4O9YfqHqOYYzV9zcufibGBBkBlVD0b8bEklReN1YaGV4QEib1vLK7pymhLa0F7EJhc8Fqx5IVicAyMSZSgw/640?wx_fmt=png&tp5&wx_lazy=1&wx_co=1

问题点：有可能会有如下信息，这个一般是端口被占用，很可能你打开了两个ServiceCenter，都关闭后再打开就可以了。

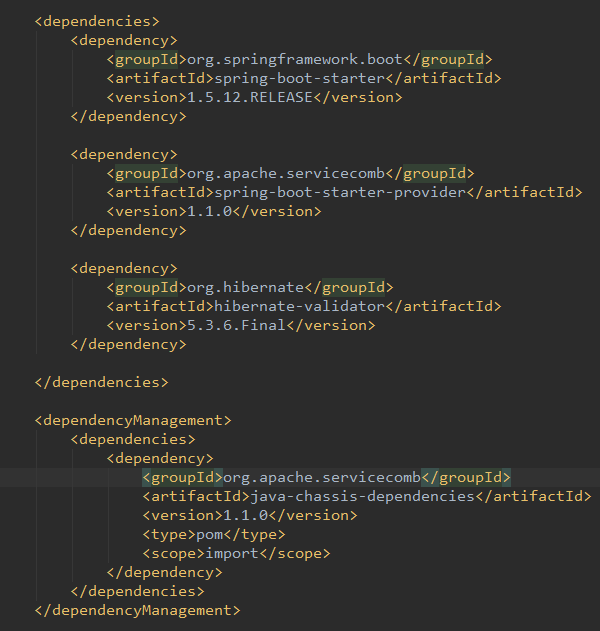


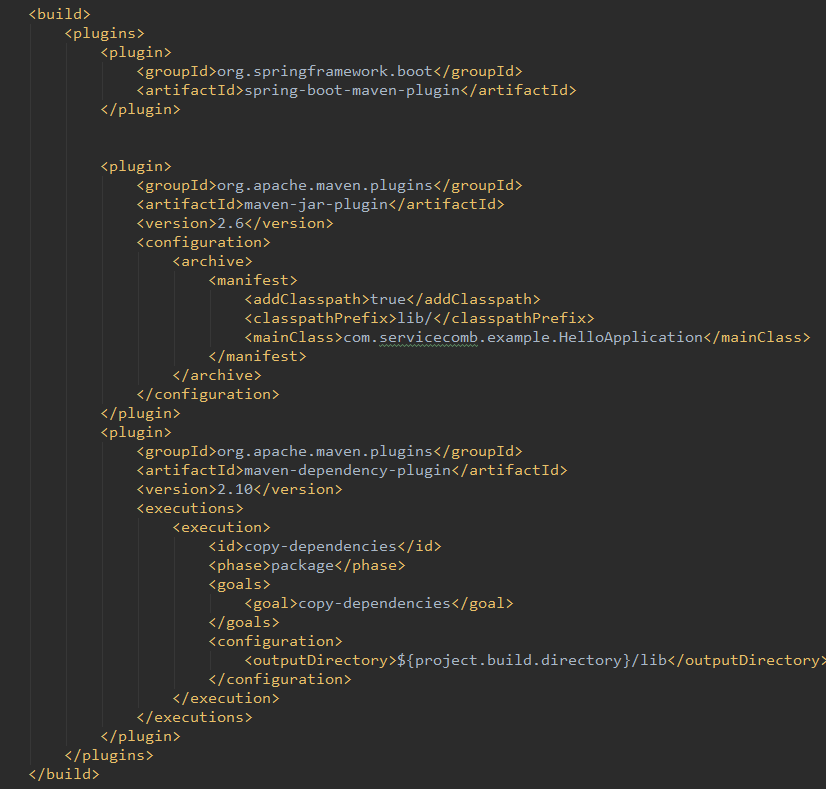
**示例**

**一、后端微服务 HelloService**

01．添加依赖

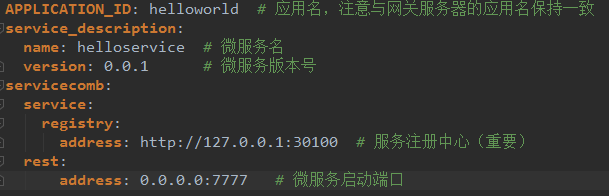
新建maven项目HelloService，pom文件如下





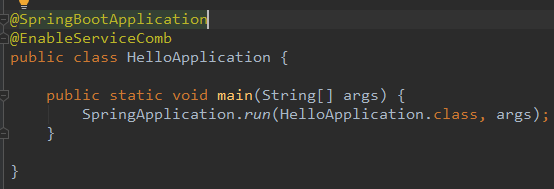
02．配置

新建ServiceComb配置文件src/main/resources/microservice.yaml，内容如下

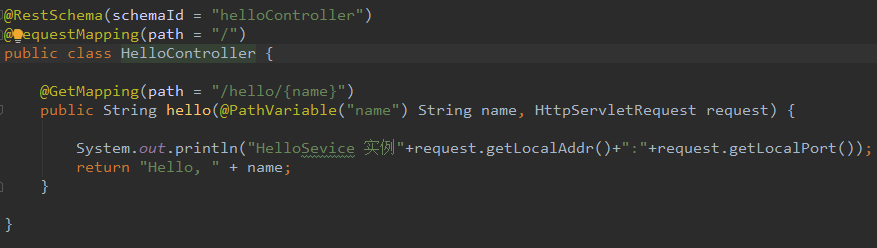


03．项目入口

新建启动类HelloApplication.java，内容如下



新建Controller类HelloController.java （这里我习惯SpringMvc的叫法，重点看注解）



04．启动

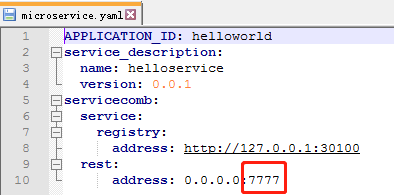
到此，Hello微服务就写完了。这里要启动2个实例。

1.先打包，执行mvn clean package

2.在项目根目录下target目录下将lib目录和生成的jar包复制到另一个目录中



3.复制jar包，修改其中一个jar包微服务启动端口号为8888（通过WinRar解压缩软件打开jar包并打开microservice.yaml文件修改里面的rest端口号）

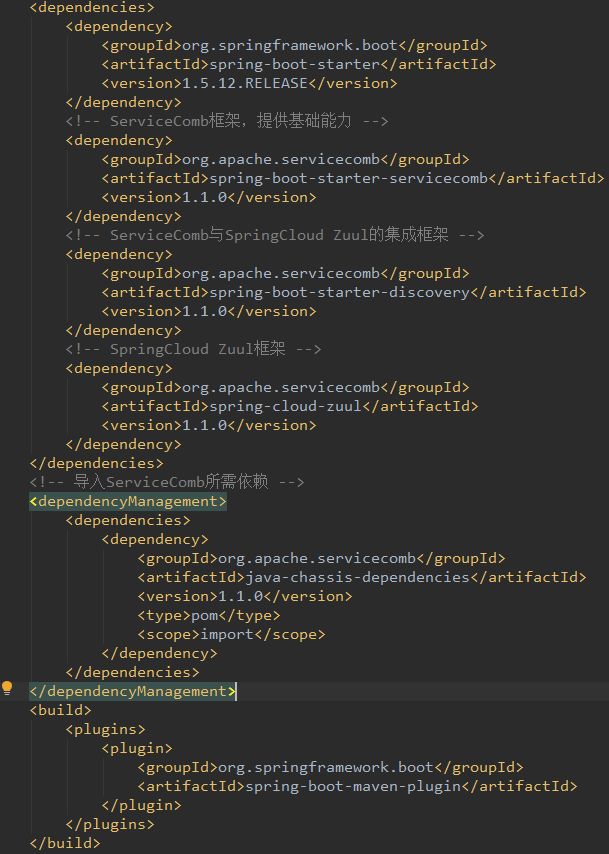


4.在当前目录打开两个cmd命令窗口，分别执行命令java -jar HelloService-7777.jar和java -jar HelloService-8888.jar 。此时已经启动了2个微服务实例。

**二、编写Zuul网关服务**

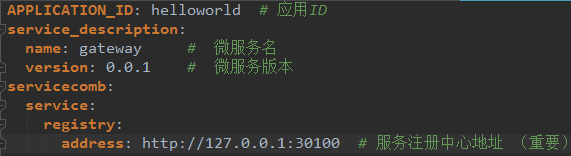
01．添加依赖

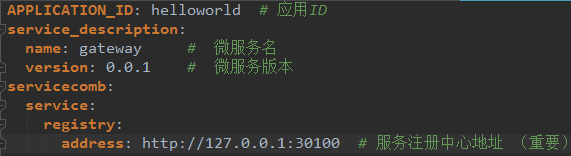
新建maven项目 zuulserver，pom文件如下



02．配置文件

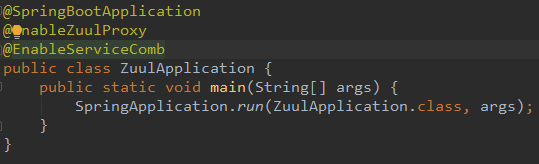
新建springboot配置文件src/main/resources/application.yaml，内容如下  
新建ServiceComb配置文件src/main/resources/microservice.yaml，内容如下



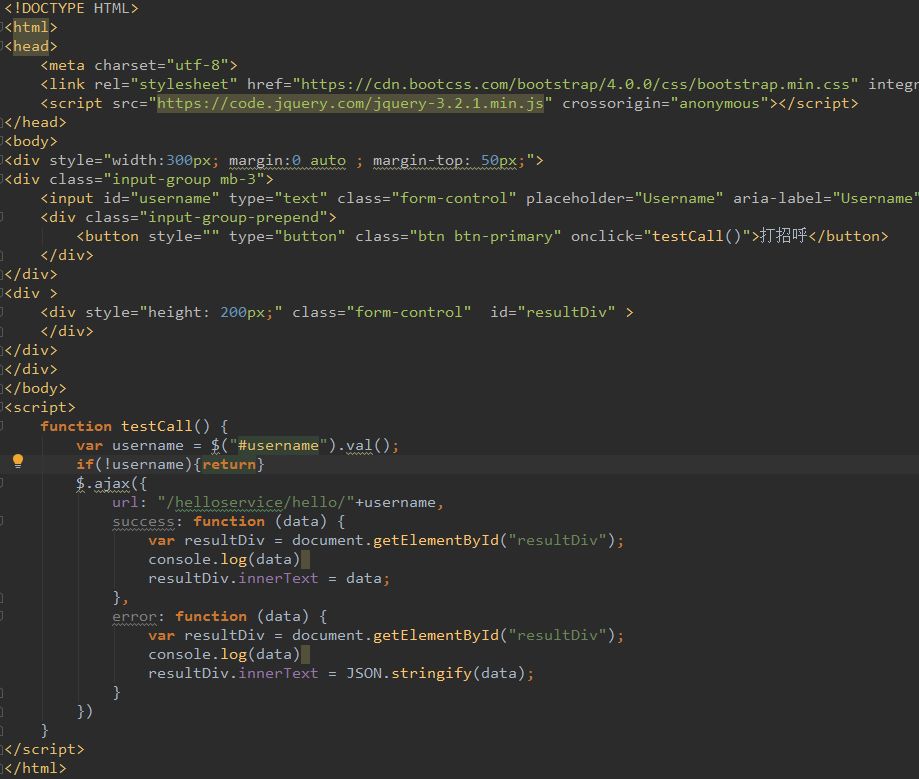


03．项目入口

新建启动类 ZuulApplication.java

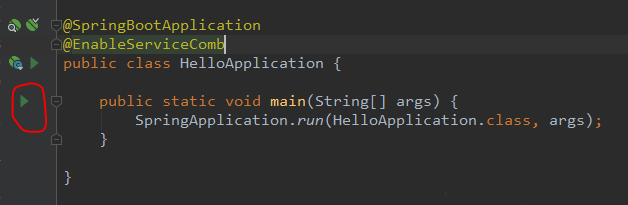


新建静态文件 src/main/resources/static/index.html，内容如下 （static目录是按springboot项目规范，属于应用静态文件根目录）



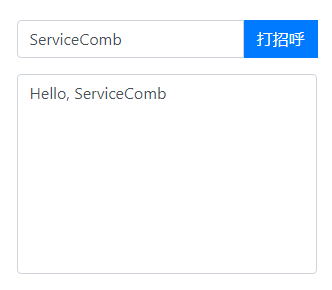
04．启动

到此，网关服务器写完，如下在IDEA里面直接启动应用



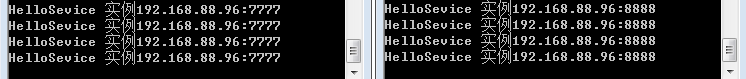
**三. 演示效果**

浏览器访问http://localhost:8080/ ，如下图。 在输入框中输入姓名，就可以在下面看到打招呼的信息



连续点击几次打招呼按钮，可以在启动的2个Hello微服务实例的控制台中看到被调用的信息，如下图（由于这里使用的ServiceComb默认的负载均衡策略 轮询，可见到两个实例都均被调用，关于负载均衡，我们将在后续的文章中解读）。

此时用户并不需要关心具体哪个实例被调用了，访问的是哪个后端的实例地址，它只要访问网关就可以了：）



总结

从以上的示例可以看出，通过 ServiceComb 结合SpringCloud Zuul 实现服务网关功能只需一些简单的配置。后续可以在网关服务上实现统一的鉴权，日志记录，和自定义过滤器等。