**[学习微服务-第8天] ServiceComb内置负载均衡组件handler-loadbalance**

在上两篇 [微服务]ServiceComb + SpringCloud Ribbon：使用篇 和 [微服务]ServiceComb + SpringCloud Ribbon：源码解读篇 中介绍了负载均衡的概念和ServiceComb结合SpringCloud Ribbon的使用, 本篇将介绍ServiceComb内置的负载均衡组件handler-loadbalance。

本文参考于官方手册：

<https://docs.servicecomb.io/java-chassis/zh_CN/references-handlers/loadbalance.html>

**简介**

ServiceComb提供了非常强大的负载均衡能力。它的核心包括两部分，第一部分是DiscoveryTree，通过将微服务实例根据接口兼容性、数据中心、实例状态等分组，DiscoveryFilter是其主要组成部分；第二部分是基于Ribbon的负载均衡方案，支持随机、顺序、基于响应时间的权值等多种负载均衡路由策略IRule，以及可以支持Invocation状态的ServerListFilterExt。

**代码示例**

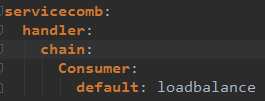
以下代码请参考官方示例：

https://github.com/apache/servicecomb-java-chassis/tree/master/samples/springmvc-sample

注意该示例中并未添加重试策略，读者可自行添加验证。

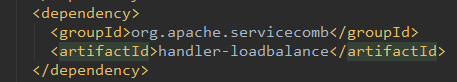
**1. 启动负载均衡**

在配置文件microservice.yaml中加入以下配置

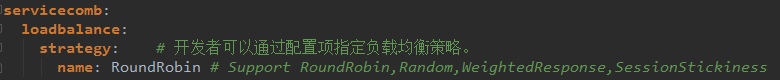


注意Consumer是首字母大写

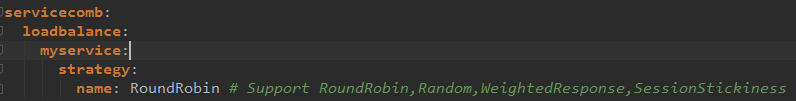
在项目的pom文件中加入以下依赖提供负载均衡支持



**2. 配置负载均衡策略**



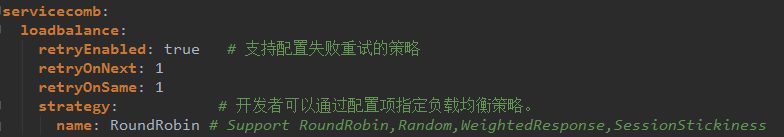
开发者还可以针对不同的微服务配置不一样的策略，只需要给配置项增加服务名，例如：



其中myservice为微服务名

**3. 设置重试策略**

负载均衡模块还支持配置失败重试的策略。默认未启用重试。开启只需加如下配置

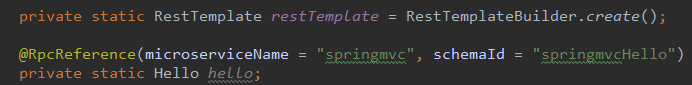


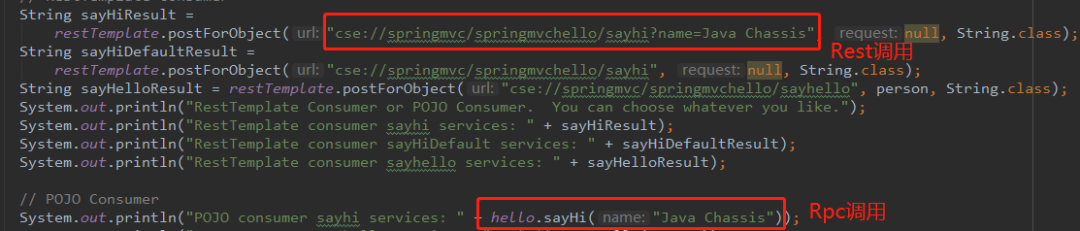
•retryOnNext : 表示失败以后，根据负载均衡策略，重新选择一个实例重试（可能选择到同一个实例）。

•retryOnSame : 表示仍然使用上次失败的实例进行重试。

**4. 代码调用**

支持Restful和Rpc调用





对于Restful调用的url形式如下：

cse://微服务名/资源路径?参数

**自定义**

ServiceComb的负载均衡模块提供了强大的扩展能力，包括DiscoveryFilter、ServerListFilterExt、ExtensionsFactory（扩展IRule，RetryHandler等）。loadbalance模块本身包含了每一个扩展的实现，这里不详细描述如何扩展。

开发者可参考官方手册：

https://docs.servicecomb.io/java-chassis/zh\_CN/references-handlers/loadbalance.html