### Feign简介

Feign是一个声明式的web服务客户端，它使得写web服务变得更简单。使用Feign,只需要创建一个接口并注解。它具有可插拔的注解特性，包括Feign 注解和JAX-RS注解。Feign同时支持可插拔的编码器和解码器。**[spring](http://lib.csdn.net/base/javaee" \o "Java EE知识库" \t "http://blog.csdn.net/forezp/article/details/_blank)** cloud对Spring mvc添加了支持，同时在spring web中次用相同的HttpMessageConverter。当我们使用feign的时候，spring cloud 整和了Ribbon和eureka去提供负载均衡。

简而言之：

* feign采用的是接口加注解
* feign 整合了ribbon

### 二、准备工作

继续用上一节的工程： 启动eureka-server，端口为8761; 启动service-hi 两次，端口分别为8762 、8773.

### 三、创建一个feign的服务

创建一个spring-boot工程，取名为：serice-feign,它的pom文件为：

<?xml version="1.0" encoding="UTF-8"?><project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.forezp</groupId>

<artifactId>service-feign</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>service-feign</name>

<description>Demo project for Spring Boot</description>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>1.5.2.RELEASE</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-eureka</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-feign</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<dependencyManagement>

<dependencies>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-dependencies</artifactId>

<version>Dalston.RC1</version>

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

<repositories>

<repository>

<id>spring-milestones</id>

<name>Spring Milestones</name>

<url>https://repo.spring.io/milestone</url>

<snapshots>

<enabled>false</enabled>

</snapshots>

</repository>

</repositories>

</project>

向服务注册中心注册它自己，这时service-feign既是服务提供者，也是服务消费者,配置文件application.yml

eureka:

client:

serviceUrl:

defaultZone: <http://localhost:8761/eureka/>

server:

port: 8765

spring:

application:

name: service-feign

在程序的入口类，需要通过注解@EnableFeignClients来开启feign:

@SpringBootApplication

@EnableDiscoveryClient

@EnableFeignClients

public class ServiceFeignApplication {

public static void main(String[] args) {

SpringApplication.run(ServiceFeignApplication.class, args);

}

}

定义一个feign接口类,通过@ FeignClient（“服务名”），来指定调用哪个服务：

/\*\*

\* Created by fangzhipeng on 2017/4/6.

\*/

@FeignClient(value = "service-hi")

public interface SchedualServiceHi {

@RequestMapping(value = "/hi",method = RequestMethod.GET)

String sayHiFromClientOne(@RequestParam(value = "name") String name);

}

在web层的controllrt:

@RestController

public class HiController {

@Autowired

SchedualServiceHi schedualServiceHi;

@RequestMapping(value = "/hi",method = RequestMethod.GET)

public String sayHi(@RequestParam String name){

return schedualServiceHi.sayHiFromClientOne(name);

}

}

访问[http://localhost:8765/hi?name=forezp](http://localhost:8765/hi?name=forezp" \t "http://blog.csdn.net/forezp/article/details/_blank),浏览器交替显示：

hi forezp,i am from port:8762

hi forezp,i am from port:8763

#### 四、更改feign的配置

在声明feignclient的时候，不仅要指定服务名，同时需要制定服务配置类：

@FeignClient(name = "stores", configuration = FooConfiguration.class)

public interface StoreClient {

//..

}

重写配置，需要加@Configuration注解，并重写下面的两个bean,栗子：

@Configuration

public class FooConfiguration {

@Bean

public Contract feignContractg() {

return new feign.Contract.Default();

}

@Bean

public BasicAuthRequestInterceptor basicAuthRequestInterceptor() {

return new BasicAuthRequestInterceptor("user", "password");

}

}