SpringCloud之四

负载均衡Feign

一、Feign简介

Feign是一个声明式的**伪Http客户端**，它使得写Http客户端变得更简单。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，可使用Feign 注解和JAX-RS注解。Feign支持可插拔的编码器和解码器。Feign默认集成了Ribbon，并和Eureka结合，默认实现了负载均衡的效果。

简而言之：

- **Feign 采用的是基于接口的注解**
- **Feign 整合了ribbon，具有负载均衡的能力**
- **整合了Hystrix，具有熔断的能力**

官网描述如图：

官网描述是：

```
Feign is a Java to HTTP client binder inspired by Retrofit, JAXRS-2.0,
and WebSocket. Feign's first goal was reducing the complexity of binding
Denominator uniformly to HTTP APIs regardless of ReSTfulness.
```

即：Feign是受到Retrofit，JAXRS-2.0和WebSocket启发的Java到HTTP客户端绑定器。Feign的第一个目标是降低将Denominator统一绑定到HTTP API 的复杂性，无论ReSTfulness如何。

Feign官网描述运作说明如下：

```
Feign works by processing annotations into a templatized request.
Arguments are applied to these templates in a straightforward fashion
before output. Although Feign is limited to supporting text-based APIs,
it dramatically simplifies system aspects such as replaying requests.
Furthermore, Feign makes it easy to unit test your conversions knowing th
is.
```

即：Feign通过将注释处理为模板化请求来工作。在输出之前，参数以直接的方式应用于这些模板。虽然Feign仅限于支持基于文本的API，但它极大地简化了系统方面，例如重放请求。此外，Feign可以轻松地对您的转换进行单元测试。

当然有需要的，还可以查看官网https://github.com/OpenFeign/feign的描述、

官网描述的Feign的注解如下：

接口注释		
Feign注释定义了Contract接口与底层客户端应如何工作之间的关系。 Feign的默认合约定义了以下注释：		
注解	**接口目标**	**用法**
@RequestLine	方法	定义HttpMethod和UriTemplatefor请求。 Expressions，{expression}使用相应的带@Param注释的参数来解析用大括号包装的值。
@Param	参数	定义模板变量，其值将用于Expression按名称解析相应的模板。
@Headers	方法，类型	定义HeaderTemplate; 一个变种UriTemplate。使用带@Param注释的值来解析相应的Expressions。在a上使用时Type，模板将应用于每个请求。在a上使用时Method，模板仅适用于带注释的方法。
@QueryMap	参数	定义一个Map名称 - 值对或POJO，以扩展为查询字符串。
@HeaderMap	参数	定义一个Map名称 - 值对，以扩展为Http Headers
@Body	方法	定义a Template，类似于a UriTemplate和HeaderTemplate，使用带@Param注释的值来解析相应的Expressions。

总的来说就是：

- Feign是一个**声明式的REST客户端**，它的目的就是让REST调用更加简单。
- Feign提供了**HTTP请求的模板**，通过编写简单的接口和插入注解，就可以定义好HTTP请求的参数、格式、地址等信息。

- 而Feign则会完全**代理HTTP请求**，我们只需要像调用方法一样调用它就可以完成服务请求及相关处理。
- SpringCloud对Feign进行了封装，使其**支持SpringMVC标准注解和HttpMessageConverters。**
- Feign可以**与Eureka和Ribbon组合**使用以支持负载均衡。

废话够多了，相信很多人会觉得文章马上要和裹脚布有的一拼了，所以开始干货发送！

二、Feign创建

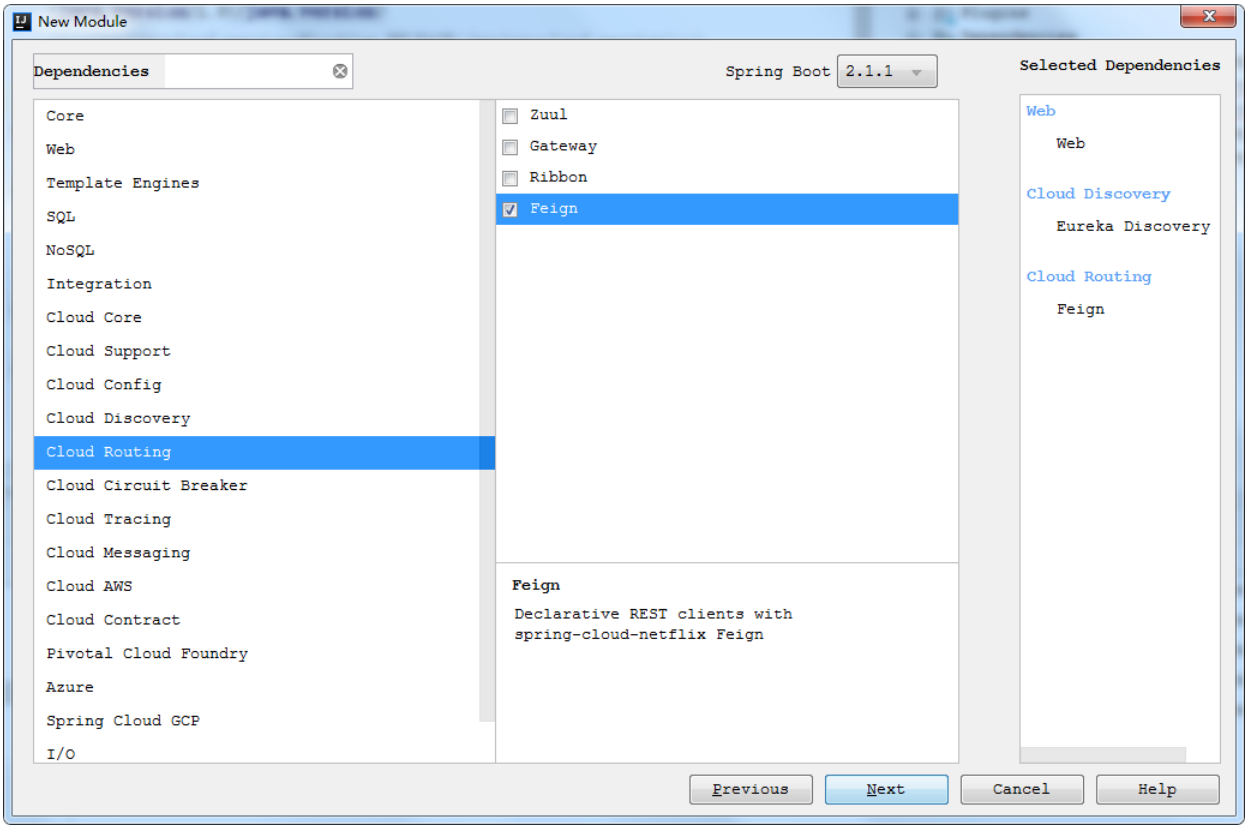
2.1、准备

启动eureka-server，端口为8761; 启动login-service两次，端口分别为8762、8773。

具体的可以参考之前的文章。

2.2、创建Feign服务

新建一个spring-boot工程，取名为feign-serice，



在它的pom文件引入Feign的起步依赖spring-cloud-starter-feign、Eureka的起步依赖spring-cloud-starter-eureka、Web的起步依赖spring-boot-starter-web，代码如下：

pom.xml

```xml
<?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>xyz.jiangnanke</groupId>
  <artifactId>feign-service</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>feign-service</name>
  <description>Demo project for Spring Boot</description>

   <parent>
   <groupId>xyz.jiangnanke</groupId>
   <artifactId>main</artifactId>
   <version>0.0.1-SNAPSHOT</version>
   </parent>

   <dependencies>
   <dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-web</artifactId>
   </dependency>
   <dependency>
   <groupId>org.springframework.cloud</groupId>
   <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
   </dependency>
   <dependency>
   <groupId>org.springframework.cloud</groupId>
   <artifactId>spring-cloud-starter-openfeign</artifactId>
   </dependency>

   </dependencies>

</project>
```

同样在main模块的pom.xml里面加入模块关联

2.3、配置项目

首先是application.yml文件，指定程序名为service-feign，端口号为8765，服务注册地址为http://localhost:8761/eureka/ ，代码如下：

```yaml
server:
 port: 8765

spring:
 application:
 name: feign-service

eureka:
 client:
  serviceUrl:
  defaultZone: http://localhost:8761/eureka/
```

然后启动类FeignServiceApplication.java的配置，加上@EnableFeignClients注解开启Feign的功能：

```java
package xyz.jiangnanke.feignservice;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.client.discovery.EnableDiscoveryClient;
import org.springframework.cloud.netflix.eureka.EnableEurekaClient;
import org.springframework.cloud.openfeign.EnableFeignClients;

@SpringBootApplication
 @EnableEurekaClient
 @EnableDiscoveryClient
 @EnableFeignClients
 public class FeignServiceApplication {

  public static void main(String[] args) {
  SpringApplication.run(FeignServiceApplication.class, args);
  }

 }

```

2.4、编写测试

首先，定义一个feign接口，通过@ FeignClient（ “服务名” ），来指定调用哪个服务。比如在代码中调用了login-service服务的 “/login” 接口，代码如下：

SchedualLoginService. java

```java
package xyz.jiangnanke.feignservice.service;

import org.springframework.cloud.openfeign.FeignClient;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;

/**
 * @Auther: zhengfeng
 * @Date: 2018\12\20 0020 15:49
 * @Description: 指定调用login-service服务
 */
@FeignClient(value = "login-service")
public interface SchedualLoginService {
 /**
 * 调用login-service服务的login接口请求
 * @param name
 * @return
 */
 @RequestMapping(value = "/login",method = RequestMethod.GET)
 String loginOne(@RequestParam(value = "name") String name);
}
```

然后，在Web层的controller层，对外暴露一个"/login"的API接口，通过上面定义的Feign客户端SchedualLoginService 来消费服务。代码如下：

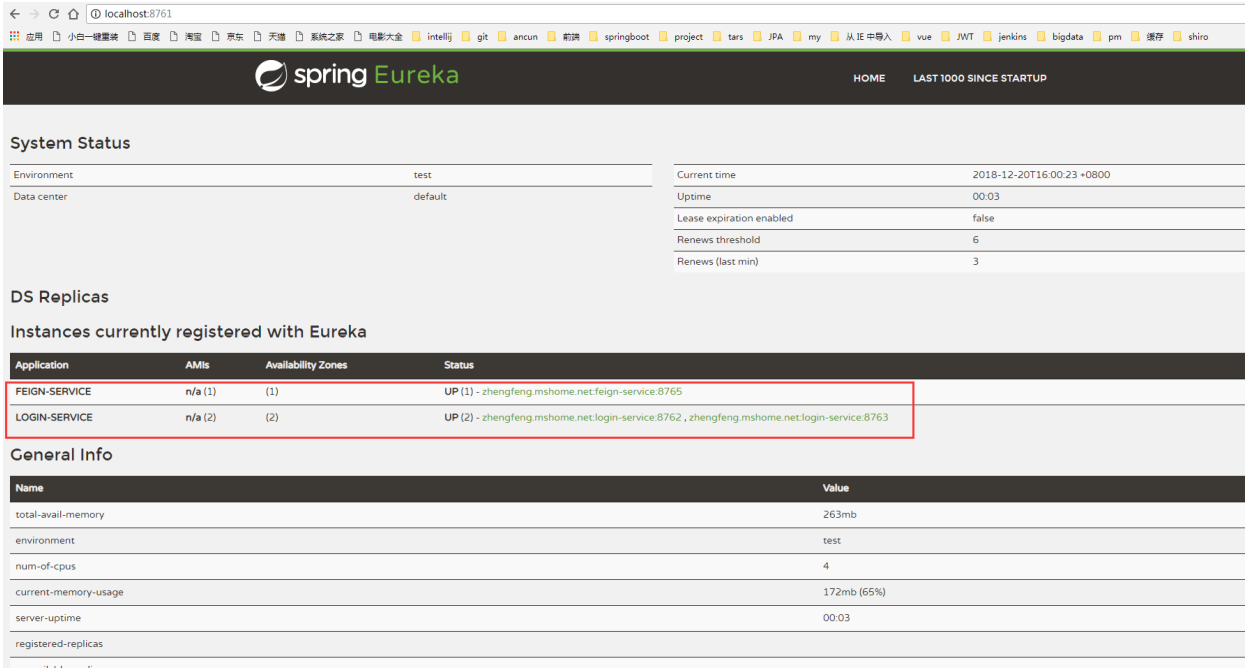
LoginController.java

```java
package xyz.jiangnanke.feignservice.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;
import xyz.jiangnanke.feignservice.service.SchedualLoginService;

/**
 * @Auther: zhengfeng
 * @Date: 2018\12\20 0020 15:52
 * @Description: 对外暴露一个"/login"的API接口，通过上面定义的Feign客户端SchedualLoginService 来消费服务
 */
@RestController
```

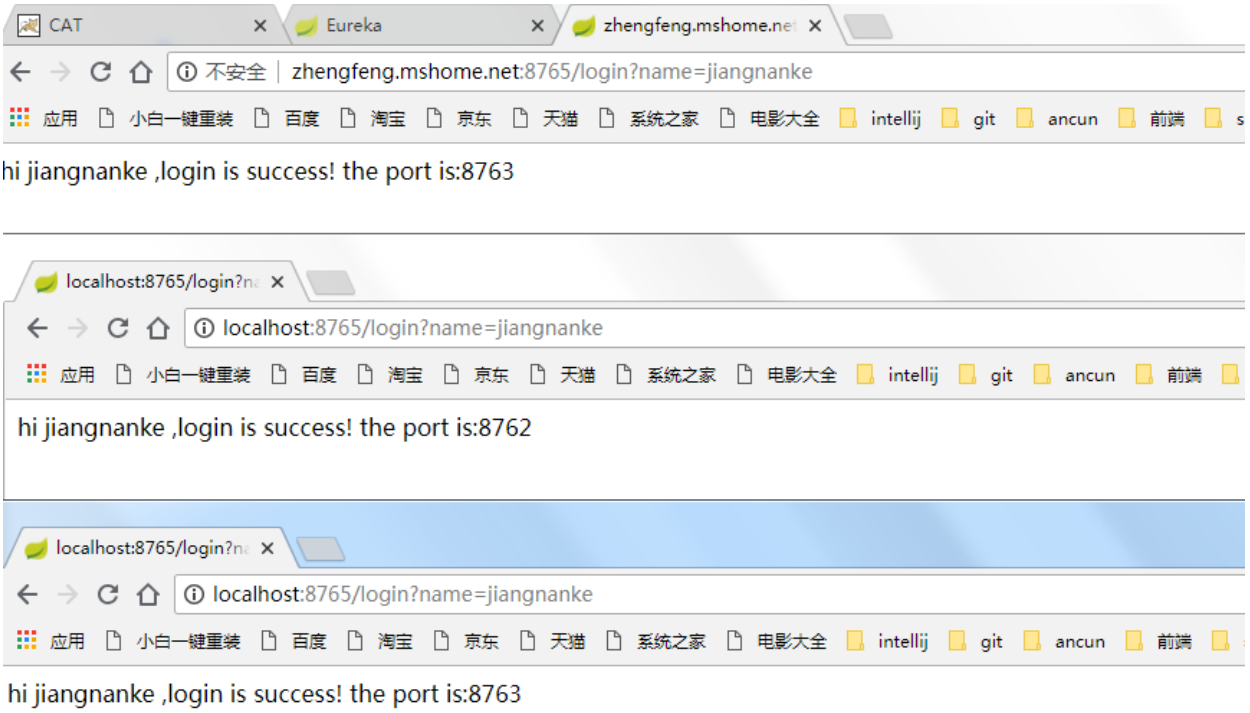
```
15 public class LoginController {
16   //编译器报错，无视。 因为这个Bean是在程序启动的时候注入的，编译器感知不到，所以报错。
17   @Autowired
18   SchedualLoginService schedualLoginService;
19
20   @GetMapping(value = "/login")
21   public String login(@RequestParam String name) {
22   return schedualLoginService.loginOne(name);
23   }
24 }
```

2.5、运行访问

可以看到在eureka里面注册了



然后多次访问http://localhost:8765/login?name=jiangnanke ,结果如下：



3、其他

3.1、JAVA 项目中接口调用怎么做 ？

1. Httpclient
2. Okhttp
3. Httpurlconnection
4. RestTemplate

参考资料

https://springcloud.cc/

https://github.com/OpenFeign/feign

https://blog.csdn.net/forezp/article/details/81040965

用Spring中的RestTemplate来调用rest接口 http://cxytiandi.com/blog/detail/6157

http://cxytiandi.com/blog/detail/12189
https://github.com/yinjihuan/spring-cloud