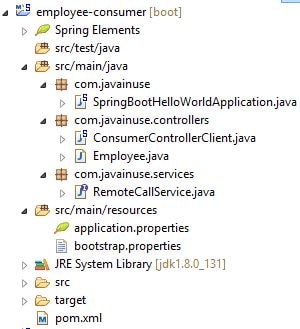
Lets Begin-

**Feign as a declarative REST client**

Feign is a Spring Cloud Netflix library for providing a higher level of abstraction over REST-based service calls. Spring Cloud Feign works on a declarative principle. When using Feign, we write declarative REST service interfaces at the client, and use those interfaces to program the client.

RestTemplate is used for making the synchronous call. When using RestTemplate, the URL parameter is constructed programmatically, and data is sent across to the other service.

As mentioned earlier the source code shared in the previous [netflix ribbon tutorial](https://www.javainuse.com/spring/spring_ribbon) will be the starting point. And the employee-consumer model we will be making the changes.  


We first add the netflix feign dependency in the pom as follows-

<?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.javainuse</groupId>

<artifactId>employee-consumer</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<parent>

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

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

<version>1.4.1.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.boot</groupId>

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

</dependency>

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

**<dependency>**

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

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

**</dependency>**

</dependencies>

<dependencyManagement>

<dependencies>

<dependency>

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

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

<version>Camden.SR6</version>

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>

</project>

We next define a Feign Client by creating an interface with @FeignClient annotation. We also specify the name value as "employee-producer". This value is the name of the service registered using Eureka for discovery. We define the method call to be made to consume the REST service exposed by the employee-producer module.

package com.javainuse.services;

import org.springframework.cloud.netflix.feign.FeignClient;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import com.javainuse.controllers.Employee;

**@FeignClient(name="employee-producer")**

public interface RemoteCallService {

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

public Employee getData();

}

Next we autowire the RemoteCallService in the ConsumerControllerClient class. Then using it make the REST call. Load Balancing is automatically taken care by Feign Client.

package com.javainuse.controllers;

import java.io.IOException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.client.RestClientException;

import com.javainuse.services.RemoteCallService;

@Controller

public class ConsumerControllerClient {

@Autowired

private RemoteCallService loadBalancer;

public void getEmployee() throws RestClientException, IOException {

try {

**Employee emp = loadBalancer.getData();**

System.out.println(emp.getEmpId());

} catch (Exception ex) {

System.out.println(ex);

}

}

}

Finally we annotate the Spring Boot Main class with **@EnableFeignClients**.

package com.javainuse;

import java.io.IOException;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.cloud.netflix.feign.EnableFeignClients;

import org.springframework.context.ApplicationContext;

import org.springframework.context.annotation.Bean;

import org.springframework.web.client.RestClientException;

import com.javainuse.controllers.ConsumerControllerClient;

@SpringBootApplication

**@EnableFeignClients**

public class SpringBootHelloWorldApplication {

public static void main(String[] args) throws RestClientException, IOException {

ApplicationContext ctx = SpringApplication.run(SpringBootHelloWorldApplication.class, args);

ConsumerControllerClient consumerControllerClient = ctx.getBean(ConsumerControllerClient.class);

System.out.println(consumerControllerClient);

consumerControllerClient.getEmployee();

}

@Bean

public ConsumerControllerClient consumerControllerClient() {

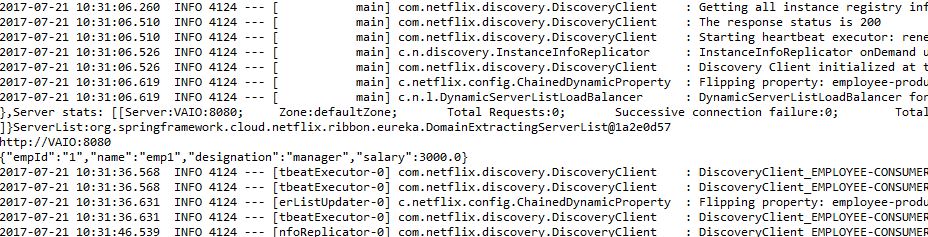
return new ConsumerControllerClient();

}

}

As we had done in previous posts- Start the following Spring Boot Applications-

* eureka-server
* employee-producer
* employee-consumer

On running the employee-consumer we get the output as follows-  


s