Spring Cloud Ribbon是一个基于HTTP和TCP的客服端负载均衡工具，它是基于Netflix Ribbon实现的。它是一个工具类框架，不像服务注册中心、配置中心、API网关那样独立部署，但是它几乎存在于每个微服务的基础设施中。微服务间的调用，API网关的请求转发等，都是通过Ribbon来实现的，包括后面要介绍的Feign，也是基于Ribbon实现的工具。

如何使用Ribbon来实现客户端的负载均衡。

#### 1、创建项目

新建一个springboot项目ribbon-server1。

#### 2、引入依赖

ribbon依赖和eureka依赖

<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-netflix-ribbon</artifactId>

</dependency>

springcloud依赖，在标签外面：

<!--spring-cloud依赖-->

<dependencyManagement>

<dependencies>

<dependency>

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

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

<version>${spring-cloud.version}</version>

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>

properties标签里设置spring-cloud版本：

<!--spring-cloud版本-->

<spring-cloud.version>Finchley.SR2</spring-cloud.version>

#### 3、修改配置文件

**server**:

**port**: 8016

**spring**:

**application**:

**name**: service-ribbon

**eureka**:

**client**:

**service-url**: http://localhost:8761/eureka/

#### 4、添加启动注解

启动类中，照例要给注册为一个eurekaClient，添加@EnableEurekaClient

#### 5、注入模板类

在启动类中，注入RestTemplate模板到容器中，该对象会使用Ribbon的自动化配置，当开启了@LoadBalance后，可以开启客户端负载均衡。

@EnableEurekaClient

@SpringBootApplication

public class RibbonServer1Application {

@Bean

@LoadBalanced

RestTemplate restTemplate() {

return new RestTemplate();

}

public static void main(String[] args) {

SpringApplication.*run*(RibbonServer1Application.class, args);

}

}

#### 6、跨服务调用

写一套简单的业务代码，然后调用前面创建的company-server服务提供的接口。

controller：

@RestController

public class RibbonController {

@Autowired

private RibbonService ribbonService;

@GetMapping(value = "getCompany")

public String getCompany(String id) {

String company = ribbonService.getCompany(id);

System.*out*.println(company);

return company;

}

}

service

public interface RibbonService {

String getCompany(String id);

}

serviceImpl：

在serviceImpl实现类中，引入前面的RestTemplate模板类，来跨服调用，并开启负载均衡功能。

@Service

public class RibbonServiceImpl implements RibbonService {

*/\*\**

*\* 启动类中注入了此模板，并开启了负载均衡功能*

*\*/*

@Autowired

RestTemplate restTemplate;

*/\*\**

*\* 程序名替代服务地址，ribbon会根据服务名自动选择服务实例*

*\** ***@param*** *id*

*\** ***@return***

*\*/*

@Override

public String getCompany(String id) {

String company = restTemplate.getForObject(

"http://company-server/company/get?id=" + id, String.class);

return company;

}

}

#### 7、启动项目，测试

启动项目，访问这个项目的接口<http://localhost:8016/getCompany?id=1234>，会发现这个服务调用了company-server服务提供的接口，并且，接口在两个相同的服务之间自动切换，也就是，实现了负载均衡的功能。

不断刷新接口，可以看到服务方发生了切换，如下：





当然，restTemplate提供了多种方法，get，post，delete，put等方法，这里只是示例了一种。