2017年6月8日 星期四

Netflix-Ribbon客户端负载均衡

1.官网地址：https://spring.io/guides/gs/client-side-load-balancing/

2.Ribbon+Spring Cloud的客户端负载均衡

3.环境

—>JDK8

—>Maven3.x or Gradel2.3+

4.实现的功能

构建一个Netflix Ribbon客户端服务应用，去调用其他的微服务实现客户端的负载均衡。

为了实现测试，我写两个服务分别为服务端ribbon-server,客户端ribbon-client

——>ribbon-server : 服务端为了测试，一共Ribbon客户端调用实现负载均衡

——>ribbon-client : ribbon客户端，负责均衡实现

5.功能实现步骤：

①首先得写一个服务器，供Ribbon客户端调用。ribbon-server

—>创建ribbon-server项目，并添加依赖，建一个简单的web项目即可添加依赖+插件

```
spring-boot-starter-web
spring-boot-maven-plugin
```

②配置文件application.yml

```yaml
spring:
  application:
    name: ribbon-server    #应用名称为ribbon-server

server:
  port: 8090          #配置端口号8090
```

③项目启动入口类SayHelloApplication

```
package hello;

import java.util.Arrays;
import java.util.List;
import java.util.Random;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
@SpringBootApplication
public class SayHelloApplication {
	private static Logger logger = LoggerFactory.getLogger(SayHelloApplication.class);
	@RequestMapping(value = "/greeting")
	public String greet() {
		logger.info("Access /greeting");
		List<String> greetings = Arrays.asList("Hi there", "Greetings","Salutations");
		Random rand = new Random();

		int randomNum = rand.nextInt(greetings.size());
		return greetings.get(randomNum);
	}
	@RequestMapping(value = "/")
	public String home() {
		logger.info("Access /");
		return "Hi!";
	}
	public static void main(String[] args) {
		SpringApplication.run(SayHelloApplication.class, args);
	}

}
```

④启动项目，为了测试负载均衡所以启多个服务，我这里启动两个，进入项目pom.xml文件下，采用maven命令启动

>>$ SERVER_PORT=8090 spring-boot:run

>>$ SERVER_PORT=9999 spring-boot:run

启动两个节点，供Ribbon客户端调用。

6.写一个Ribbon客户端服务器

①创建ribbon-client项目，并添加依赖和插件

```
spring-cloud-starter-ribbon
spring-boot-starter-web
spring-boot-maven-plugin
```

②添加配置文件application.yml

```yaml
spring:
  application:
    name: ribbon-client

server:
  port: 8888

myService:              #不采用Eureka方式，官网提供这中自定义前缀的方式设置myServer为自定义前缀
  ribbon:
    eureka:
      enabled: false    #不采用Eureka注册
    listOfServers: localhost:8090,localhost:9999     #服务列表，要调用的服务，上面已经启动的两个节点
    ServerListRefreshInterval: 15000     #服务列表刷新时间
```

③对客户端的额外配置类RibbonConfiguration

```
package hello;

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

import com.netflix.client.config.IClientConfig;
import com.netflix.loadbalancer.IPing;
import com.netflix.loadbalancer.IRule;
import com.netflix.loadbalancer.PingUrl;
import com.netflix.loadbalancer.WeightedResponseTimeRule;

/**
 * 客户端额外配置，设置之后会覆盖客户端默认配置
 * @author hushuang
 * @email hd1611756908@gmail.com
 * @Time  2017年6月9日 下午7:53:23
 * @Description:
 */
public class RibbonConfiguration {
        @Autowired
        IClientConfig ribbonClientConfig;
        @Bean
        public IPing ribbonPing(IClientConfig config) {
                return new PingUrl();//检测服务器状态状态
        }
        //负载均衡策略设置（权重，随机，轮询等等）
        @Bean
        public IRule ribbonRule(IClientConfig config) {
                return new WeightedResponseTimeRule();
        }

}
```

④客户端实现类UserApplication

```
package hello;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.client.loadbalancer.LoadBalanced;
import org.springframework.cloud.netflix.ribbon.RibbonClient;
import org.springframework.context.annotation.Bean;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.client.RestTemplate;

@SpringBootApplication
@RestController
//name：配置文件中myService为前缀,RibbonConfiguration为客户端额外配置，不能进行全局扫描，只适用于此客户端
@RibbonClient(name="myService",configuration=RibbonConfiguration.class)
public class UserApplication {
	@LoadBalanced
	@Bean
	RestTemplate restTemplate() {
		return new RestTemplate();
	}
	@Autowired
	RestTemplate restTemplate;
	@RequestMapping("/hi")
	public String hi(@RequestParam(value = "name", defaultValue = "Artaban") String name) {
		//http://myService/greeting ： SpringCloud的一个特性将url地址换成应用名称，此工程因为没有使用注册中心但是myService前缀设置了地址。所以此处设置myService
		String greeting = this.restTemplate.getForObject("http://myService/greeting", String.class);
		return String.format("%s, %s!", greeting, name);
	}
	public static void main(String[] args) {
		SpringApplication.run(UserApplication.class, args);
	}
}
```

⑤启动ribbon客户端打开浏览器输入http://localhost:8888/hi 多请求几次看看ribbon-server后台日志。调用的哪个服务。