通过上节课程，我们已经成功地将服务提供者：eureka-client或consul-client注册到了Eureka服务注册中心或Consul服务端上了，同时我们也通过DiscoveryClient接口的getServices获取了当前客户端缓存的所有服务清单，那么接下来我们要学习的就是：如何去消费服务提供者的接口？

使用LoadBalancerClient

在Spring Cloud Commons中提供了大量的与服务治理相关的抽象接口，包括DiscoveryClient、这里我们即将介绍的LoadBalancerClient等。对于这些接口的定义我们在上一篇介绍服务注册与发现时已经说过，Spring Cloud做这一层抽象，很好的解耦了服务治理体系，使得我们可以轻易的替换不同的服务治理设施。

从LoadBalancerClient接口的命名中，我们就知道这是一个负载均衡客户端的抽象定义，下面我们就看看如何使用Spring Cloud提供的负载均衡器客户端接口来实现服务的消费。

下面的例子，我们将利用上一篇中构建的eureka-server作为服务注册中心、eureka-client作为服务提供者作为基础。

- 我们先来创建一个服务消费者工程，命名为：eureka-consumer。并在pom.xml中引入依赖（这里省略了parent和dependencyManagement的配置）：

```
<dependencies>
    <dependency>
        <groupId>org.springframework.cloud</groupId>
        <artifactId>spring-cloud-starter-eureka</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-actuator</artifactId>
    </dependency>
</dependencies>
```

- 配置application.properties，指定eureka注册中心的地址：

```
spring.application.name=eureka-consumer
server.port=2101
eureka.client.serviceUrl.defaultZone=http://localhost:1001/eureka/
```

- 创建应用主类。初始化RestTemplate，用来真正发起REST请求。@EnableDiscoveryClient注解用来将当前应用加入到服务治理体系中。

```
package lizzy.springcloud;

import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.builder.SpringApplicationBuilder;
import org.springframework.cloud.client.discovery.EnableDiscoveryClient;
```

```
import org.springframework.context.annotation.Bean;
import org.springframework.web.client.RestTemplate;

@EnableDiscoveryClient
@SpringBootApplication
public class EurekaConsumerApplication {
    @Bean
    public RestTemplate restTemplate() {
        return new RestTemplate();
    }
    public static void main(String[] args) {
        new
SpringApplicationBuilder(EurekaConsumerApplication.class).web(true).run(args);
    }
}
```

- 创建一个接口用来消费eureka-client提供的接口：

```
package lizzy.springcloud.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.cloud.client.ServiceInstance;
import org.springframework.cloud.client.loadbalancer.LoadBalancerClient;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.client.RestTemplate;

@RestController
public class DcController {

    @Autowired
    LoadBalancerClient loadBalancerClient;
    @Autowired
    RestTemplate restTemplate;

    @GetMapping("/consumer")
    public String dc() {
        ServiceInstance serviceInstance = loadBalancerClient.choose("eureka-
client");
        String url = "http://" + serviceInstance.getHost() + ":" +
serviceInstance.getPort() + "/dc";
        System.out.println(url);
        return restTemplate.getForObject(url, String.class);
```

```
    }
}
```

可以看到这里，我们注入了`LoadBalancerClient`和`RestTemplate`，并在`/consumer`接口的实现中，先通过`loadBalancerClient`的`choose`函数来负载均衡的选出一个`eureka-client`的服务实例，这个服务实例的基本信息存储在`ServiceInstance`中，然后通过这些对象中的信息拼接出访问`/dc`接口的详细地址，最后再利用`RestTemplate`对象实现对服务提供者接口的调用。

在完成了上面你的代码编写之后，读者可以将eureka-server、eureka-client、eureka-consumer都启动起来，然后访问http://localhost:2101/consumer ，来跟踪观察eureka-consumer服务是如何消费eureka-client服务的`/dc`接口的。

具体工程说明如下：

- eureka的服务注册中心：eureka-server
- eureka的服务提供方：eureka-client
- eureka的服务消费者：eureka-consumer