链滴

springcloud(三)：服务提供与调用

作者：911708498

原文链接：https://ld246.com/article/1543310621979

来源网站：链滴

上一篇文章我们介绍了eureka服务注册中心的搭建，这篇文章介绍一下如何使用eureka服务注册中，搭建一个简单的服务端注册服务，客户端去调用服务使用的案例。

案例中有三个角色：服务注册中心、服务提供者、服务消费者，其中服务注册中心就是我们上一篇的eureka单机版启动既可，流程是首先启动注册中心，服务提供者生产服务并注册到服务中心中，消费者服务中心中获取服务并执行。

服务提供

我们假设服务提供者有一个hello方法，可以根据传入的参数，提供输出“hello ，this is first messg ”的服务

1、pom包配置

创建一个springboot项目，pom.xml中添加如下配置：

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

2、配置文件

application.properties配置如下：

```
spring.application.name=spring-cloud-producer
server.port=9000
eureka.client.serviceUrl.defaultZone=http://localhost:8000/eureka/
```

3、启动类

启动类中添加@EnableDiscoveryClient注解

```
@SpringBootApplication
@EnableDiscoveryClient
public class ProducerApplication {

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

4、controller

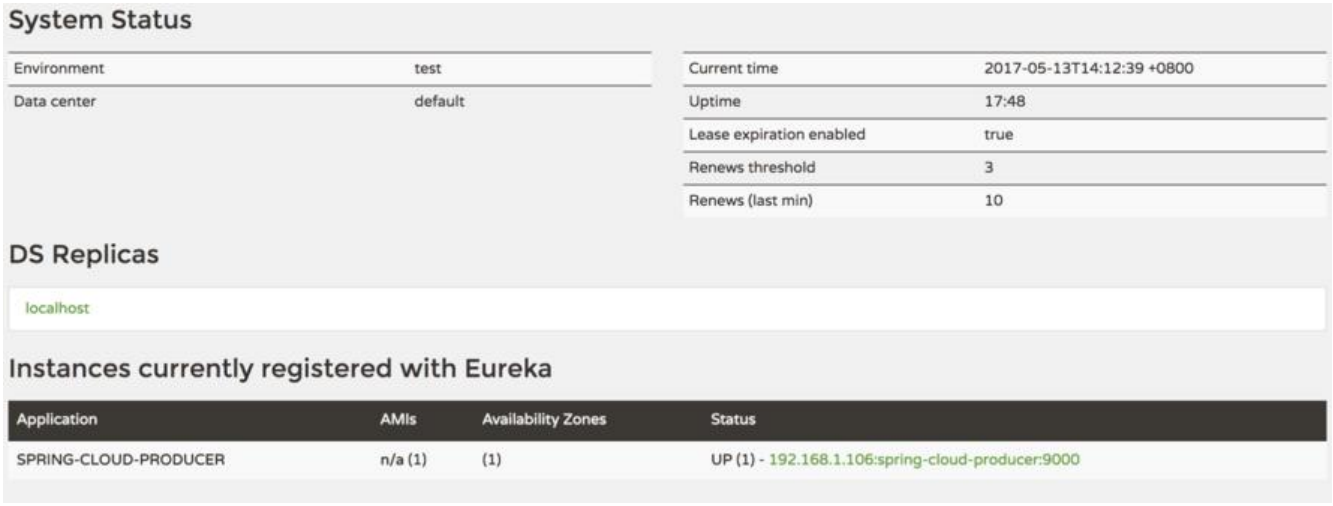
提供hello服务

```
@RestController
public class HelloController {

    @RequestMapping("/hello")
```

```
    public String index(@RequestParam String name) {
        return "hello "+name+", this is first messge";
    }
}
```

添加@EnableDiscoveryClient注解后，项目就具有了服务注册的功能。启动工程后，就可以在注册心的页面看到SPRING-CLOUD-PRODUCER服务。



到此服务提供者配置就完成了。

服务调用

1、pom包配置

和服务提供者一致

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

2、配置文件

application.properties配置如下：

```
spring.application.name=spring-cloud-consumer
server.port=9001
eureka.client.serviceUrl.defaultZone=http://localhost:8000/eureka/
```

3、启动类

启动类添加@EnableDiscoveryClient和@EnableFeignClients注解。

```
@SpringBootApplication
@EnableDiscoveryClient
```

```
@EnableFeignClients
public class ConsumerApplication {

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

}
```

@EnableDiscoveryClient :启用服务注册与发现

@EnableFeignClients：启用feign进行远程调用

Feign是一个声明式Web Service客户端。使用Feign能让编写Web Service客户端更加简单, 它的使方法是定义一个接口，然后在上面添加注解，同时也支持JAX-RS标准的注解。Feign也支持可拔插式编码器和解码器。Spring Cloud对Feign进行了封装，使其支持了Spring MVC标准注解和HttpMessageConverters。Feign可以与Eureka和Ribbon组合使用以支持负载均衡。

4、feign调用实现

```
@FeignClient(name= "spring-cloud-producer")
public interface HelloRemote {
    @RequestMapping(value = "/hello")
    public String hello(@RequestParam(value = "name") String name);
}
```

name:远程服务名，及spring.application.name配置的名称

此类中的方法和远程服务中contoller中的方法名和参数需保持一致。

5、web层调用远程服务

将HelloRemote注入到controller层，像普通方法一样去调用即可。

```
@RestController
public class ConsumerController {

    @Autowired
    HelloRemote HelloRemote;

    @RequestMapping("/hello/{name}")
    public String index(@PathVariable("name") String name) {
        return HelloRemote.hello(name);
    }

}
```

到此，最简单的一个服务注册与调用的例子就完成了。

整体代码结构如下：

原文链接：springcloud(三)：服务提供与调用