1. 摘要
2. Spring Cloud OpenFeign 是声明式的服务调用工具，它整合了Ribbon和Hystrix，拥有负载均衡和服务容错功能。

二、Feign简介

1、Feign是声明式的服务调用工具，我们只需创建一个接口并用注解的方式来配置它，就可以实现对某个服务接口的调用，简化了直接使用RestTemplate来调用服务接口的开发量。Feign具备可插拔的注解支持，同时支持Feign注解、JAX-RS注解及SpringMvc注解。当使用Feign时，Spring Cloud集成了Ribbon和Eureka以提供负载均衡的服务调用及基于Hystrix的服务容错保护功能。

三、创建一个feign-service模块

1、在pom.xml中添加相关依赖：

<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-openfeign</artifactId>  
</dependency>  
  
<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
</dependency>

2、在application.yml中进行配置

server:  
 port: 8701  
spring:  
 application:  
 name: feign-service  
eureka:  
 client:  
 register-with-eureka: true #注册到Eureka注册中心中  
 fetch-registry: true #获取服务列表  
 service-url:  
 defaultZone: <http://localhost:8001/eureka/> #指定注册中心地址

3、在启动类上添加@EnableFeignClients注解来启用Feign的客户端功能

4、添加UserService接口完成对user-service服务的接口绑定

注意：我们通过@FeignClient注解实现了一个Feign客户端，其中的value为user-service表示这是对user-service服务的接口调用客户端。我们可以回想下user-service中的UserController，只需将其改为接口，保留原来的SpringMvc注释即可。

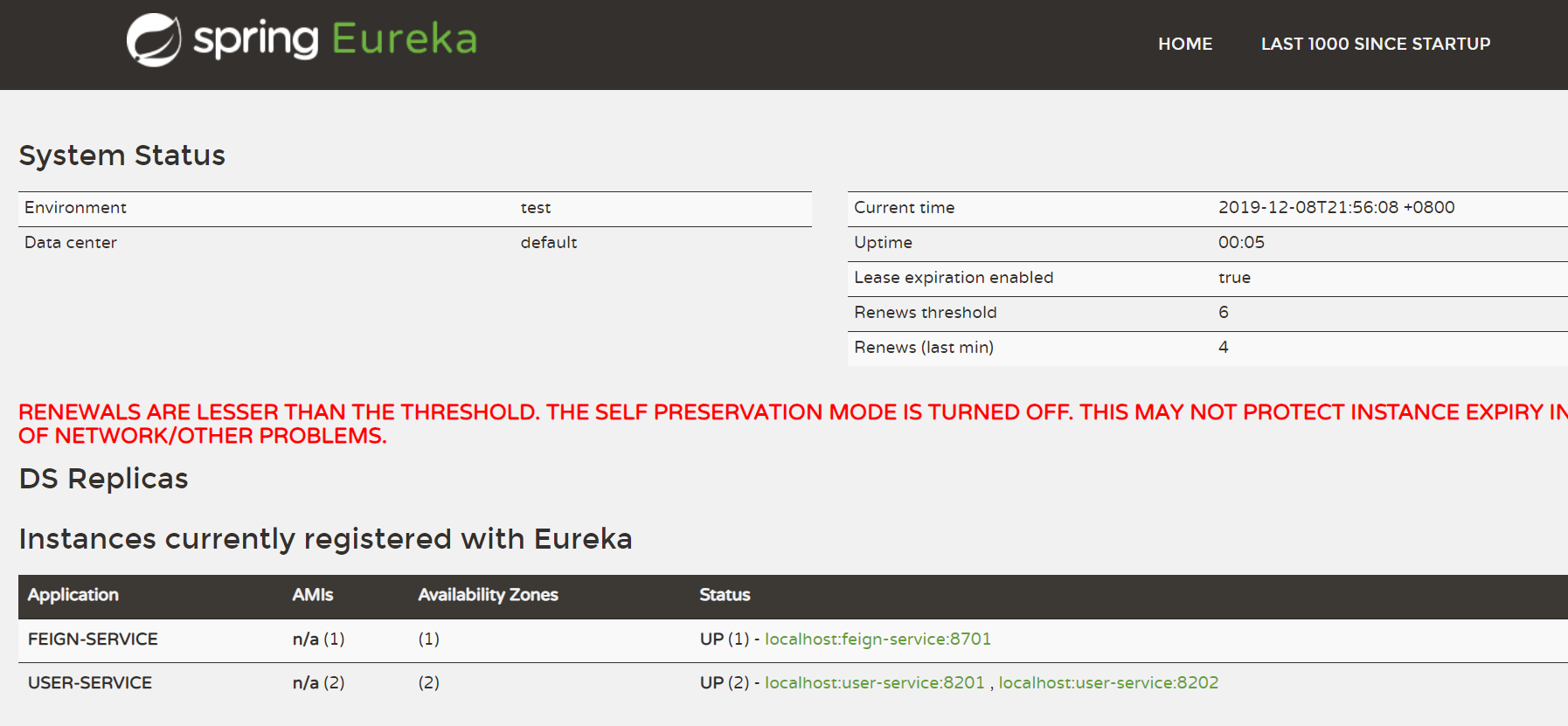
*/\*\*  
 \** ***@author*** *lihao  
 \** ***@ClassName*** *UserService  
 \** ***@Since*** *2019/12/8  
 \*/*//对user-service服务的接口调用客户端  
@FeignClient(value = "user-service", fallback = UserFallbackService.class)  
public interface UserService {  
 @PostMapping("/user/create")  
 CommonResult create(@RequestBody User user);  
  
 @GetMapping("/user/{id}")  
 CommonResult<User> getUser(@PathVariable Long id);  
  
 @GetMapping("/user/getByUsername")  
 CommonResult<User> getByUsername(@RequestParam String username);  
  
 @PostMapping("/user/update")  
 CommonResult update(@RequestBody User user);  
  
 @PostMapping("/user/delete/{id}")  
 CommonResult delete(@PathVariable Long id);  
}

5、添加UserFeignController调用UserService实现服务调用

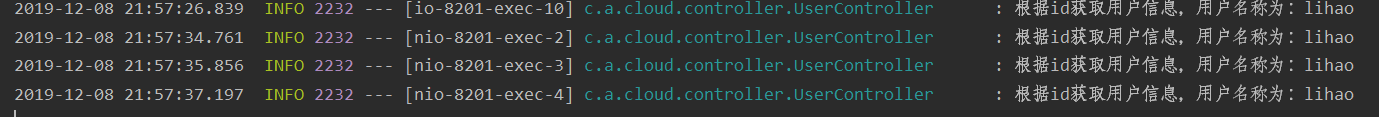
*/\*\*  
 \** ***@author*** *lihao  
 \** ***@ClassName*** *UserFeignController  
 \** ***@Since*** *2019/12/8  
 \*/*@RestController  
@RequestMapping("/user")  
public class UserFeignController {  
 @Autowired  
 private UserService userService;  
  
 @GetMapping("/{id}")  
 public CommonResult getUser(@PathVariable Long id) {  
 return userService.getUser(id);  
 }  
  
 @GetMapping("/getByUsername")  
 public CommonResult getByUsername(@RequestParam String username) {  
 return userService.getByUsername(username);  
 }  
  
 @PostMapping("/create")  
 public CommonResult create(@RequestBody User user) {  
 return userService.create(user);  
 }  
  
 @PostMapping("/update")  
 public CommonResult update(@RequestBody User user) {  
 return userService.update(user);  
 }  
  
 @PostMapping("/delete/{id}")  
 public CommonResult delete(@PathVariable Long id) {  
 return userService.delete(id);  
 }  
}

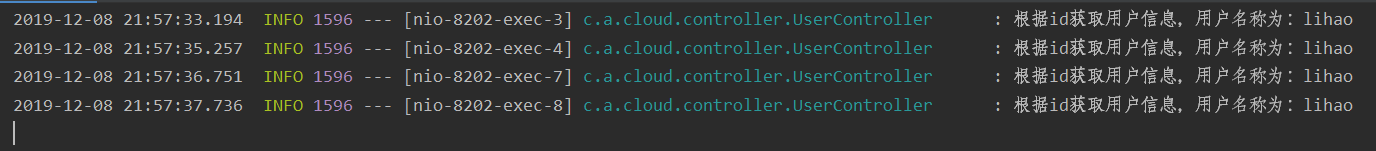
四、负载均衡功能演示

1、启动eureka-service，两个user-service，feign-service服务，启动后注册中心显示如下：



2、多次调用[http://localhost:8701/user/1](http://localhost:8701/user/1" \t "https://juejin.im/post/_blank)进行测试，可以发现运行在8201和8202的user-service服务交替打印如下信息：





五、Feign中的服务降级

注意：Feign中的服务降级使用起来非常方便，只需要为Feign客户端定义的接口添加一个服务降级处理的实现类即可，下面我们为UserService接口添加一个服务降级实现类。

1、添加服务降级实现类UserFallbackService

注意：需要注意的是它实现了UserService接口，并且对接口中的每个实现方法进行了服务降级逻辑的实现。

*/\*\*  
 \* 服务降级实现类  
 \*  
 \** ***@author*** *lihao  
 \** ***@ClassName*** *UserFallbackService  
 \** ***@Since*** *2019/12/8  
 \*/*@Component  
public class UserFallbackService implements UserService {  
 @Override  
 public CommonResult create(User user) {  
 User defaultUser = new User(-1L, "defaultUser", "123456");  
 return new CommonResult<>(defaultUser);  
 }  
  
 @Override  
 public CommonResult<User> getUser(Long id) {  
 User defaultUser = new User(-1L, "defaultUser", "123456");  
 return new CommonResult<>(defaultUser);  
 }  
  
 @Override  
 public CommonResult<User> getByUsername(String username) {  
 User defaultUser = new User(-1L, "defaultUser", "123456");  
 return new CommonResult<>(defaultUser);  
 }  
  
 @Override  
 public CommonResult update(User user) {  
 return new CommonResult("调用失败，服务被降级", 500);  
 }  
  
 @Override  
 public CommonResult delete(Long id) {  
 return new CommonResult("调用失败，服务被降级", 500);  
 }  
}

2、修改UserService接口，设置服务降级处理类为UserFallbackService

注意：修改@FeignClient注解中的参数，设置fallback为UserFallbackService.class即可。

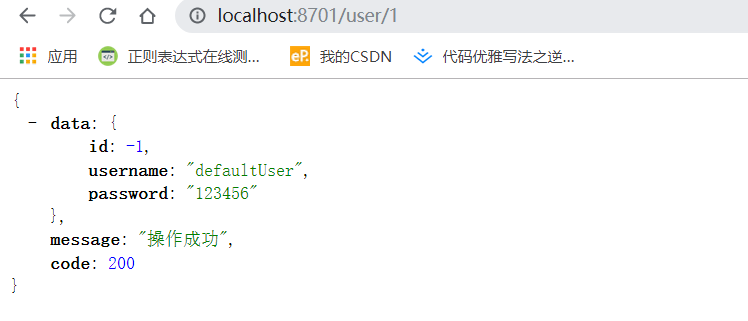
@FeignClient(value = "user-service", fallback = UserFallbackService.class)  
public interface UserService {

}

3、修改application.yml，开启Hystrix功能

feign:  
 hystrix:  
 enabled: true *#在Feign中开启Hystrix*

1. 服务降级功能演示
2. 关闭两个user-service服务，重新启动feign-service;
3. 调用[http://localhost:8701/user/1](http://localhost:8701/user/1" \t "https://juejin.im/post/_blank)进行测试，可以发现返回了服务降级信息。



七、日志打印功能

注意：Feign提供了日志打印功能，我们可以通过配置来调整日志级别，从而了解Feign中Http请求的细节。

1、日志级别

①NONE：默认的，不显示任何日志；

②BASIC：仅记录请求方法、URL、响应状态码及执行时间；

③HEADERS：除了BASIC中定义的信息之外，还有请求和响应的头信息；

④FULL：除了HEADERS中定义的信息之外，还有请求和响应的正文及元数据。

2、通过配置开启更为详细的日志

注意：通过java配置来使Feign打印最详细的Http请求日志信息。

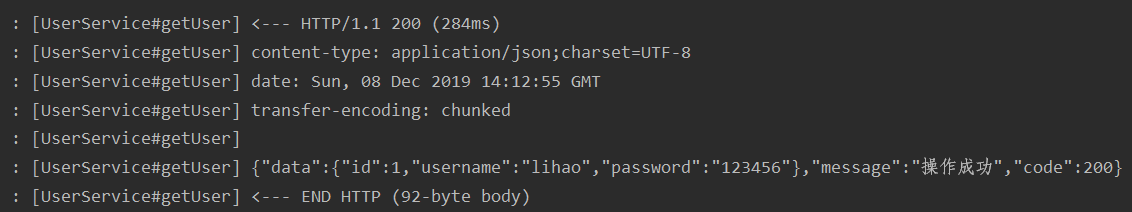
*/\*\*  
 \* 设置日志打印级别  
 \** ***@author*** *lihao  
 \** ***@ClassName*** *FeignConfig  
 \** ***@Since*** *2019/12/8  
 \*/*@Configuration  
public class FeignConfig {  
 @Bean  
 Logger.Level feignLoggerLevel() {  
 return Logger.Level.*FULL*;  
 }  
}

3、在application.yml中配置需要开启日志的Feign客户端

注意：配置UserService的日志级别为debug。

4、查看日志

调用[http://localhost:8701/user/1](http://localhost:8701/user/1" \t "https://juejin.im/post/_blank)进行测试，可以看到以下日志。



八、Feign的常用配置

1、Feign自己的配置

2、Feign中的Ribbon配置

注意：在Feign中配置Ribbon可以直接使用Ribbon的配置。

3、Feign中的Hystrix配置

注意：在Feign中配置Hystrix可以直接使用Hystrix的配置。

九、使用到的模块

springcloud-learning

├── eureka-server -- eureka注册中心

├── user-service -- 提供User对象CRUD接口的服务

├── feign-service -- feign服务调用测试服务