# **10.Feign客户端通过Gateway访问其他服务的方法**

## **1. 常规的feign接口(不经过网关)**

* 一般情况下，我们使用feign客户端调用其他服务时是这样定义的

@FeignClient(name="xxx",fallback=xxx.class)

public interface Hello(){

.......

}

* 这种方式的请求不会经过网关
* 有人会说，你直接把name属性设置成网关的服务名不就行了吗
* 当有多个这样的接口时，如果都把name设置成网关，spring会报bean不唯一的错误。因此，那种解决方案是不可行的
* 那么我们希望服务间调用也能够进入网关处理逻辑，比如验证，token验证或者需要经过网关过滤器处理，该如何实现呢？答案是contextId

## **2. contextId 解决(经过网关)**

@FeignClient(contextId = "[唯一的名称]",name="网关服务名",fallback =xxx .class)

public interface Hello {

@GetMapping("[其他服务名]/[具体的url]")

String Hello(){

.......

}

}

## **3. 分析**

### **3.1 FeignClient注解的使用介绍**

转载地址：<http://www.imooc.com/article/details/id/299213>

****value, name****：value和name的作用一样，如果没有配置url那么配置的值将作为服务名称，用于服务发现。反之只是一个名称。

****serviceId****：serviceId已经废弃了，直接使用name即可。

****contextId**** ：比如我们有个user服务，但user服务中有很多个接口，我们不想将所有的调用接口都定义在一个类中，比如

@FeignClient(name = "optimization-user")

public interface UserRemoteClient {

@GetMapping("/user/get")

public User getUser(@RequestParam("id") int id);

}

--------------

@FeignClient(name = "optimization-user")

public interface UserRemoteClient2 {

@GetMapping("/user2/get")

public User getUser(@RequestParam("id") int id);

}

* 这种情况下启动就会报错了，因为Bean的名称冲突了，具体错误如下：

Description:

The bean 'optimization-user.FeignClientSpecification', defined in null, could not be registered. A bean with that name has already been defined in null and overriding is disabled.

Action:

Consider renaming one of the beans or enabling overriding by setting spring.main.allow-bean-definition-overriding=true

解决方案可以增加下面的配置，作用是允许出现beanName一样的BeanDefinition。  
spring.main.allow-bean-definition-overriding=true

另一种解决方案就是为每个Client手动指定不同的contextId，这样就不会冲突了。

* 上面给出了Bean名称冲突后的解决方案，下面来分析下contextId在Feign Client的作用，在注册Feign Client Configuration的时候需要一个名称，名称是通过getClientName方法获取的：
* 如果配置了contextId就会用contextId，如果没有配置就会去value然后是name最后是serviceId。默认都没有配置，当出现一个服务有多个Feign Client的时候就会报错了。
* 其次的作用是在注册FeignClient中，contextId会作为Client 别名的一部分，如果配置了qualifier优先用qualifier作为别名

****url****：url用于配置指定服务的地址，相当于直接请求这个服务，不经过Ribbon的服务选择。像调试等场景可以使用。

@FeignClient(name = "optimization-user", url = "http://localhost:8085")

public interface UserRemoteClient {

@GetMapping("/user/get")

public User getUser(@RequestParam("id") int id);

}