

# trade-order

## 说明

- 项目编码：UTF-8
- JDK:1.8
- IDE：不限
- 项目开发方式
  - 分支开发，发布时合并主干
  - 环境隔离 (profile: local、dev、uat、prod)

## 模块

- biz：公共业务模块（供其它模块调用）
- biz-base:基础业务
- biz-req-trace:请求链路跟踪
- biz-glue:
- trade-order-api：业务代码
- trade-order-gateway：路由服务

## 功能一 实现类似于Nginx请求转发功能

1.通过applications.yml读取负载均衡的URL列表，actions如下

```
actions: http://campus.query1.ksyun.com:8089,http://campus.query2.ksyun.com:8089
```

2.通过随机负载均衡算法将trade-order-gateway的请求打至trade-order-api

3.采用postman发送http请求。



## 4.结果

系统报的日志如下，可以看到两次请求发送去了两个不同的url

```
2023-07-24 15:48:24.990 | [请求的请求Id为:lbnb] | INFO | GatewayService | http-nio-8088-exec-5 | 负载均衡，转发去了http://campus.query2.ksyun.com:8089/online/trade_order/44
responseBody:{code:200,msg:ok,requestId:lbnb,desc:0.00毫秒,data:{upsteam:http://campus.query1.ksyun.com,user_id:3835.0,price_value:9339.8398,user:{username:user44,email:
2023-07-24 15:48:24.997 | [请求的请求Id为:lbnb] | INFO | webLogger | http-nio-8088-exec-5 | 方法:GatewayController_queryOrderInfo 执行耗时:6.779 ms |
2023-07-24 15:49:03.012 | [请求的请求Id为:lbnb] | INFO | webLogger | http-nio-8088-exec-6 | 方法:GatewayController_queryOrderInfo,请求参数:{url=http://localhost:8088/online/queryOrderInfo, method=GE
headerValue:lbnb
2023-07-24 15:49:03.012 | [请求的请求Id为:lbnb] | INFO | GatewayService | http-nio-8088-exec-6 | 负载均衡，转发去了http://campus.query2.ksyun.com:8089/online/trade_order/44
responseBody:{code:200,msg:ok,requestId:lbnb,desc:1.00毫秒,data:{upsteam:http://campus.query1.ksyun.com,user_id:3835.0,price_value:9339.8398,user:{username:user44,email:
2023-07-24 15:49:03.017 | [请求的请求Id为:lbnb] | INFO | webLogger | http-nio-8088-exec-6 | 方法:GatewayController_queryOrderInfo 执行耗时:5.110 ms |
```

# 功能二 查询订单详情

## 1.通过postman发送请求

GET

http://localhost:8088/online/queryOrderInfo?id=66

发送

## 2.通过随机负载均衡算法去打请求

3.用到了多级缓存，先查本地缓存，再查redis缓存，没有就最后再去查数据库

第一次查询结果如下，本地和redis都没有，去查数据库和第三方接口共用了6.14s:

2023-07-24 15:58:22.684 | [[请求的requestId为:lbxNB]] INFO | TradeOrderController | http-nio-8089-exec-9 | | 本地和redis都没有，去查数据库!!!

GET http://localhost:8088/online/queryOrderInfo?id=66 发送

参数 授权 Header (7) Body 预请求脚本 测试 设置 Cookie

Header 6个已隐藏

键	值	描述	...	批量修改	预设
<input checked="" type="checkbox"/> X-KSY-REQUEST-ID	lbxNB				

Body Cookie Header (3) 测试结果 状态: 200 OK 时间: 6.15 s 大小: 514 B 保存响应

美化 原 预览 可视化 JSON

```
1 {
2   "code": 200,
3   "msg": "ok",
4   "requestId": "lbxNB",
5   "descr": "6.14秒",
6   "data": {
7     "upsteam": "http://campus.query1.ksyun.com",
8     "user_id": 3477,
9     "price_value": 9865.0957,
10    "user": {
11      "username": "user66",
12      "email": "user66@example.com",
13      "phone": "10000737870",
14      "address": "重庆市建设路保利城44号"
15    },
16    "region": {
17      "code": "Weinan",
18      "name": "渭南"
19    },
20    "config": [
21      {
22        "item_no": "cpu",
23        "item_name": "CPU",
24        "unit": "个",
25        "value": 367
26      }
27    ]
28  }
29 }
```

第二次结果如下，只用了2ms，用到了本地缓存

2023-07-24 15:58:22.684 | [[请求的requestId为:lbxNB]] INFO | TradeOrderController | http-nio-8089-exec-9 | | 本地和redis都没有，去查数据库!!!

GET http://localhost:8088/online/queryOrderInfo?id=66 发送

参数 授权 Header (7) Body 预请求脚本 测试 设置 Cookie

Header 6个已隐藏

键	值	描述	...	批量修改	预设
<input checked="" type="checkbox"/> X-KSY-REQUEST-ID	lbxNB				

Body Cookie Header (3) 测试结果 状态: 200 OK 时间: 17 ms 大小: 521 B 保存响应

美化 原 预览 可视化 JSON

```
1 {
2   "code": 200,
3   "msg": "ok",
4   "requestId": "lbxNB",
5   "descr": "2.00毫秒",
6   "data": {
7     "upsteam": "http://campus.query1.ksyun.com",
8     "user_id": 3477.0,
9     "price_value": 9865.0957,
10    "user": {
11      "username": "user66",
12      "email": "user66@example.com",
13      "phone": "10000737870",
14      "address": "重庆市建设路保利城44号"
15    },
16    "region": {
17      "code": "Weinan",
18      "name": "渭南"
19    },
20    "config": [
21      {
22        "item_no": "cpu",
23        "item_name": "CPU",
24        "unit": "个",
25        "value": 367.0
26      }
27    ]
28  }
29 }
```

重启程序，会用到redis缓存

2023-07-24 16:03:12.902 | [[请求的requestId为:lbxBN]] INFO | TradeOrderController | http-nio-8089-exec-1 | | 用到了redis缓存! :  
2023-07-24 16:03:12.906 | [[请求的requestId为:lbxBN]] INFO | webLogger | http-nio-8089-exec-1 | | 方法:TradeOrderController\_query 执行耗时:41.18 ms |

功能三 根据机房Id查询机房名称

本功能用到的知识点：重试，我用的guava-retrying（第三方接口不稳定，我这里重试次数设置为10次）\*\*

1.postman获得的结果如下

GET

▼

http://127.0.0.1:8088/online/queryRegionName?regionId=33

发送

▼

参数●授权Header (7)Body预请求脚本测试设置

Cookie

查询参数

	键	值	描述	...	批量修改
☰	<input checked="" type="checkbox"/>	regionId	33		×
	键	值	描述		

BodyCookieHeader (3)测试结果

200 OK3.52 s191 B保存响应

▼

美化原预览可视化JSON

▼

≡

1

{

2

"code": 200.0,

3

"msg": "ok",

4

"requestId": "lbxBN",

5

"data": "中山"

6

}

这次运气不好，一共重试了五次才获取成功，哈哈。

```
2023-07-24 16:10:04.001 | [[请求的requestId为:]] INFO | DispatcherServlet | http-nio-8088-exec-1 | Completed initialization in 0 ms  
2023-07-24 16:10:04.684 | [[请求的requestId为:lbxBN]] INFO | webLogger | http-nio-8088-exec-1 | 方法:GatewayController_queryRegionName,请求参数:{url=http://127.0.0.1:8088/online/queryRegionName, method=  
2023-07-24 16:10:04.693 | [[请求的requestId为:lbxBN]] INFO | GatewayService | http-nio-8088-exec-1 | http://campus.meta.ksyun.com:8099/online/region/name/ 重试第1次  
2023-07-24 16:10:05.840 | [[请求的requestId为:lbxBN]] INFO | GatewayService | http-nio-8088-exec-1 | 从第三方接口获取到的数据为:{"code":500,"msg":"接口请求异常, 请稍后重试!", "requestId":"bfad01f1-e09b-4883-9952  
2023-07-24 16:10:05.844 | [[请求的requestId为:lbxBN]] INFO | GatewayService | http-nio-8088-exec-1 | 输出内容为:{"code":500.0, msg=接口请求异常, 请稍后重试!, requestId=lbxBN, data=null}  
2023-07-24 16:10:06.346 | [[请求的requestId为:lbxBN]] INFO | GatewayService | http-nio-8088-exec-1 | http://campus.meta.ksyun.com:8099/online/region/name/ 重试第2次  
2023-07-24 16:10:06.401 | [[请求的requestId为:lbxBN]] INFO | GatewayService | http-nio-8088-exec-1 | 从第三方接口获取到的数据为:{"code":500,"msg":"接口请求异常, 请稍后重试!", "requestId":"8efa5578-6941-4758-8052  
2023-07-24 16:10:06.401 | [[请求的requestId为:lbxBN]] INFO | GatewayService | http-nio-8088-exec-1 | 输出内容为:{"code":500.0, msg=接口请求异常, 请稍后重试!, requestId=lbxBN, data=null}  
2023-07-24 16:10:06.902 | [[请求的requestId为:lbxBN]] INFO | GatewayService | http-nio-8088-exec-1 | http://campus.meta.ksyun.com:8099/online/region/name/ 重试第3次  
2023-07-24 16:10:06.957 | [[请求的requestId为:lbxBN]] INFO | GatewayService | http-nio-8088-exec-1 | 从第三方接口获取到的数据为:{"code":500,"msg":"接口请求异常, 请稍后重试!", "requestId":"f40fd8b8-9d5c-4983-8510  
2023-07-24 16:10:06.957 | [[请求的requestId为:lbxBN]] INFO | GatewayService | http-nio-8088-exec-1 | 输出内容为:{"code":500.0, msg=接口请求异常, 请稍后重试!, requestId=lbxBN, data=null}  
2023-07-24 16:10:07.525 | [[请求的requestId为:lbxBN]] INFO | GatewayService | http-nio-8088-exec-1 | http://campus.meta.ksyun.com:8099/online/region/name/ 重试第4次  
2023-07-24 16:10:07.526 | [[请求的requestId为:lbxBN]] INFO | GatewayService | http-nio-8088-exec-1 | 从第三方接口获取到的数据为:{"code":500,"msg":"接口请求异常, 请稍后重试!", "requestId":"3aabcba8-9d71-42d0-80ef  
2023-07-24 16:10:07.526 | [[请求的requestId为:lbxBN]] INFO | GatewayService | http-nio-8088-exec-1 | 输出内容为:{"code":500.0, msg=接口请求异常, 请稍后重试!, requestId=lbxBN, data=null}  
2023-07-24 16:10:08.032 | [[请求的requestId为:lbxBN]] INFO | GatewayService | http-nio-8088-exec-1 | http://campus.meta.ksyun.com:8099/online/region/name/ 重试第5次  
2023-07-24 16:10:08.086 | [[请求的requestId为:lbxBN]] INFO | GatewayService | http-nio-8088-exec-1 | 从第三方接口获取到的数据为:{"code":200,"msg":"ok","requestId":"2aaba729-3da0-44ed-a9d1-7232648e5cb0","de  
2023-07-24 16:10:08.086 | [[请求的requestId为:lbxBN]] INFO | GatewayService | http-nio-8088-exec-1 | 输出内容为:{"code":200.0, msg=ok, requestId=lbxBN, data=中山}  
2023-07-24 16:10:08.086 | [[请求的requestId为:lbxBN]] INFO | webLogger | http-nio-8088-exec-1 | 方法:GatewayController_queryRegionName 执行耗时:3.425 s |
```

功能四 订单优惠券抵扣公摊

用postman的http的post请求发送，， 结果如下，可以抵扣多次

POST http://127.0.0.1:8088/online/voucher/deduct

参数 授权 Header (8) Body 预请求脚本 测试 设置

Cookie

none form-data x-www-form-urlencoded raw binary GraphQL JSON 美化

```
1 {
2   "orderId": 888,
3   "voucherNo": "TEST123456",
4   "amount": 300.0000
5 }
```

Body Cookie Header (3) 测试结果 200 OK 401 ms 217 B 保存响应

美化 原 预览 可视化 JSON

```
1 {
2   "code": 200,
3   "RequestId": "e90154d4-3720-4bed-8584-587a859107a2",
4   "msg": "ok",
5   "descr": "0.00毫秒"
6 }
```

对象	ksc_voucher_deduct @test_trade (lb...						
开始事务	文本	筛选	排序	导入	导出		
id	order_id	voucher_no	amount	before_deduct	after_deduct	create_time	update_time
1	1	TEST123456	10.0000	2000.0000	1990.0000	2023-07-23 04:02	2023-07-23 04:02:08
65	1	TEST123456	20.0000	1990.0000	1970.0000	2023-07-23 23:45	2023-07-23 23:45:36
66	1	TEST123456	0.0000.0000	0.0000	0.0000	2023-07-23 23:45	2023-07-23 23:45:48
67	1	TEST123456	2.0000	0.0000	0.0000	2023-07-23 23:45	2023-07-23 23:45:57
68	1	TEST123456	2.0000	0.0000	0.0000	2023-07-23 23:47	2023-07-23 23:47:50
69	1	TEST123456	2.0000	0.0000	0.0000	2023-07-23 23:47	2023-07-23 23:47:51
70	10	TEST123456	2.0000	9406.5013	9404.5013	2023-07-24 00:59	2023-07-24 00:59:11
71	39	TEST123456	2.0000	9842.3585	9840.3585	2023-07-24 00:59	2023-07-24 00:59:47
72	39	TEST123456	2.0000	9840.3585	9838.3585	2023-07-24 01:00	2023-07-24 01:00:00
73	39	TEST123456	10.0000	9838.3585	9828.3585	2023-07-24 01:00	2023-07-24 01:00:12
74	500	TEST123456	0.0000.0000	4901.4077	0.0000	2023-07-24 01:01	2023-07-24 01:01:30
75	5000	TEST123456	0.0000.0000	5890.0230	0.0000	2023-07-24 01:04	2023-07-24 01:04:16
76	5001	TEST123456	0.0000.0000	11861.4906	0.0000	2023-07-24 01:06	2023-07-24 01:06:06
77	5001	TEST123456	0.0000.0000	0.0000	0.0000	2023-07-24 01:07	2023-07-24 01:07:16
78	777	TEST123456	300.0000	17179.8338	16879.8338	2023-07-24 15:38	2023-07-24 15:38:52
79	888	TEST123456	300.0000	12151.7954	11851.7954	2023-07-24 17:01	2023-07-24 17:01:42
80	888	TEST123456	300.0000	11851.7954	11551.7954	2023-07-24 17:02	2023-07-24 17:02:44

## 功能五 基于Redis实现漏桶限流算法，并在API调用上体现

### 1.实现思路

1. 写了一漏桶限流类 `RedisLeakyBucketService`,
2. 定义了 `INTERVAL_TIME = 1000`;表示一秒
3. 定义了jedis的list变量，往里面存储时间戳，通过计算1s内时间戳的数量来判断当前有多少给http请求，如果请求数量大于设定的阈值(5),就返回false，表示现在http请求已经满了，不可访问了

用postman发送请求，我一次发了100个，从下图中可以看出在数量为6时就不可以访问了

```
2023-07-24 17:31:34.730 | [请求的requestId为:LBX666]] | INFO | RedisLeakyBucketService | http-nio-8088-exec-5 | timestampCount:5
2023-07-24 17:31:34.730 | [请求的requestId为:LBX666]] | INFO | RedisLeakyBucketService | http-nio-8088-exec-5 | 是否可以访问:true
2023-07-24 17:31:34.730 | [请求的requestId为:LBX666]] | INFO | webLogger | http-nio-8088-exec-5 | 方法:GatewayController_listUpstreamInfo 执行耗时:1.661 ms !
2023-07-24 17:31:34.821 | [请求的requestId为:LBX666]] | INFO | webLogger | http-nio-8088-exec-6 | 方法:GatewayController_listUpstreamInfo,请求参数:{url=http://127.0.0.1:8088/online/listUpstreamInfo, met
2023-07-24 17:31:34.822 | [请求的requestId为:LBX666]] | INFO | RedisLeakyBucketService | http-nio-8088-exec-6 | timestampCount:6
2023-07-24 17:31:34.822 | [请求的requestId为:LBX666]] | INFO | RedisLeakyBucketService | http-nio-8088-exec-6 | 是否可以访问:false
```

postman发送批量请求的，所以我在浏览器访问结果如下，所以是访问不了的



## 简单的链路跟踪实现

1. 在postman发送http请求的时候带X-KSY-REQUEST-ID及requestId
2. 通过HttpServletRequest获取请求头的requestId，并放入MDC.put(traceId,requestId)放入日志中
3. 再通过把traceId放入logback的配置文件中就可以了

在日志中体现如下图：

```
2023-07-24 17:47:15.536 [请求的requestId为:LBX666] INFO | RedisLeakyBucketService | http-nio-8088-exec-9 | timestampCount:14
2023-07-24 17:47:15.536 [请求的requestId为:LBX666] INFO | RedisLeakyBucketService | http-nio-8088-exec-9 | 是否可以访问:false
2023-07-24 17:47:15.536 [请求的requestId为:LBX666] INFO | webLogger | http-nio-8088-exec-9 | 方法:GatewayController_ListUpstreamInfo 执行耗时:1.195 ms !
2023-07-24 17:47:15.630 [请求的requestId为:LBX666] INFO | webLogger | http-nio-8088-exec-10 | 方法:GatewayController_ListUpstreamInfo,请求参数:{url=http://127.0.0.1:8088/online/listUpstreamInfo, met
2023-07-24 17:47:15.630 [请求的requestId为:LBX666] INFO | RedisLeakyBucketService | http-nio-8088-exec-10 | timestampCount:13
2023-07-24 17:47:15.630 [请求的requestId为:LBX666] INFO | RedisLeakyBucketService | http-nio-8088-exec-10 | 是否可以访问:false
2023-07-24 17:47:15.630 [请求的requestId为:LBX666] INFO | webLogger | http-nio-8088-exec-10 | 方法:GatewayController_ListUpstreamInfo 执行耗时:1.277 ms !
2023-07-24 17:47:15.708 [请求的requestId为:LBX666] INFO | webLogger | http-nio-8088-exec-1 | 方法:GatewayController_ListUpstreamInfo,请求参数:{url=http://127.0.0.1:8088/online/listUpstreamInfo, meth
2023-07-24 17:47:26.343 [请求的requestId为:lbxBNB] INFO | webLogger | http-nio-8088-exec-3 | 方法:GatewayController_queryOrderInfo,请求参数:{url=http://localhost:8088/online/queryOrderInfo, method=GE
headerValue:lbxBNB
2023-07-24 17:47:26.344 [请求的requestId为:lbxBNB] INFO | GatewayService | http-nio-8088-exec-3 | 负载均衡，转发去了http://campus.query2.ksyun.com:8089/online/trade_order/74
responseBody:{"code":200
2023-07-24 17:47:27.562 [请求的requestId为:lbxBNB] INFO | webLogger | http-nio-8088-exec-3 | 方法:GatewayController_queryOrderInfo 执行耗时:1.228 s !
2023-07-24 17:47:29.571 [请求的requestId为:lbxBNB] INFO | webLogger | http-nio-8088-exec-4 | 方法:GatewayController_queryRegionName,请求参数:{url=http://127.0.0.1:8088/online/queryRegionName, method=
2023-07-24 17:47:29.571 [请求的requestId为:lbxBNB] INFO | GatewayService | http-nio-8088-exec-4 | http://campus.meta.ksyun.com:8090/online/region/name/id重试第1次
2023-07-24 17:47:29.640 [请求的requestId为:lbxBNB] INFO | GatewayService | http-nio-8088-exec-4 | 从第三方接口获取到的数据为:{"code":200,"msg":"ok","requestId":"da2e0e51-812a-464a-ab54-1baf06c1dee4","de
2023-07-24 17:47:29.643 [请求的requestId为:lbxBNB] INFO | GatewayService | http-nio-8088-exec-4 | 输出内容为:{code=200.0, msg=ok, requestId=lbxBNB, data=黄山}
2023-07-24 17:47:29.644 [请求的requestId为:lbxBNB] INFO | webLogger | http-nio-8088-exec-4 | 方法:GatewayController_queryRegionName 执行耗时:72.80 ms !
2023-07-24 17:47:31.570 [请求的requestId为:lbX666] INFO | webLogger | http-nio-8088-exec-5 | 方法:GatewayController_ListUpstreamInfo,请求参数:{url=http://127.0.0.1:8088/online/listUpstreamInfo, meth
```