##### 后台与移动端及前端接口交互规范v1.1

[一 背景 2](#_Toc8912565)

[二 后台接口规范 3](#_Toc8912566)

[1 入参约定 3](#_Toc8912567)

[2 入参传递方式 3](#_Toc8912568)

[3 请求方式及controller请求注解 4](#_Toc8912569)

[4 resultfull接口约定 4](#_Toc8912570)

[5 接口相应格式 8](#_Toc8912571)

[6 数据缓存 9](#_Toc8912572)

[7接口安全 9](#_Toc8912573)

[8 接口聚合处理,减少多次请求 10](#_Toc8912574)

[9 接口默认值及数据格式问题 10](#_Toc8912575)

[10 YAPI上接口必须添加好Mock数据 11](#_Toc8912576)

[11 接口入参参数校验 12](#_Toc8912577)

[12 YAPI文档完善 12](#_Toc8912578)

[13 YAPI系统上接口完成状态设置 12](#_Toc8912579)

[14接口名称要有版本号 12](#_Toc8912580)

[15接口传参需要说明下必填项和非必填项 13](#_Toc8912581)

[16 接口传参若是枚举值，需要把各个枚举的定义写一下 13](#_Toc8912582)

[17 其他建议 14](#_Toc8912583)

[附1:版本历史 14](#_Toc8912584)

# 一 背景

1. 目的

对不同项目后台接口与移动端,前端交互时,接口规范不统一的情况,进行规范.减少沟通成本,提高开发联调效率.

# 二 后台接口规范

## 1 入参约定

#### 入参请求头

1.1.1 请求头中存放token信息 ,其中

head名称:Authorization

对应的值: eyJzdWIiOiJVU0VSMTgw

值的生成由前端提交登陆信息到服务器端,服务器端验证通过后,根据算法生成后返回给前端,前端后面每次请求都带上这个请求头信息.

1.1.2 Content-Type的值分别有如下三种:

application/x-www-form-urlencoded

application/json

multipart/form-data



## 2 入参传递方式

*keyValue方式:*

Content-Type为application/x-www-form-urlencoded, 数据被编码为名称/值对, 会将表单内的数据转换为键值对，如:name=lulei&address=hangzhou

*json方式:*

Content-Type为application/json,数据以json格式提交, postman软件里标的是RAW

*文件流方式:*

Content-Type为multipart/form-data, 它会将表单的数据处理为一条消息, 既可以上传键值对，也可以上传文件,一般用于上传文件时使用

## 3 请求方式及controller请求注解

get请求: 请求获取数据

使用@GetMapping 注解(等同于@RequestMapping(method = RequestMethod.GET))

post请求: 新增数据

使用@PostMapping 注解(等同于@RequestMapping(method = RequestMethod.POST) )

put请求: 更新数据

使用@PutMapping 注解(等同于@RequestMapping(method = RequestMethod.PUT) )

delete请求: 删除数据

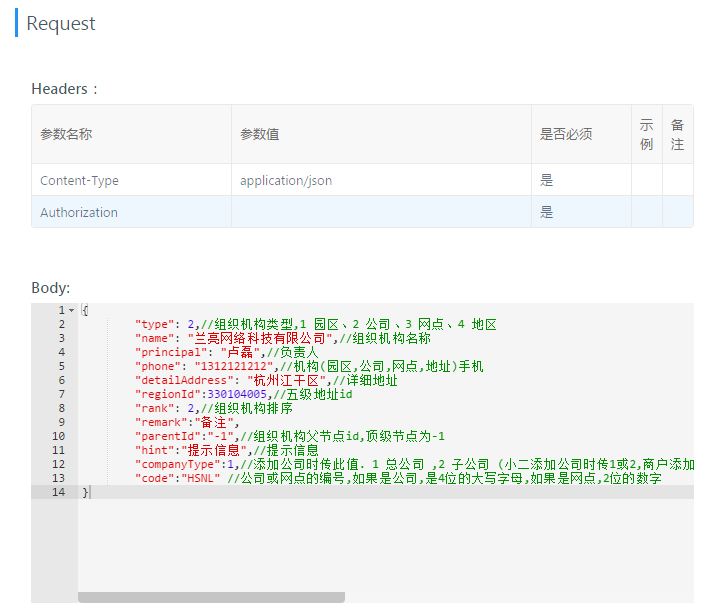
使用@DeleteMapping注解(等同于@RequestMapping(method = RequestMethod.DELETE) )

## 4 resultfull接口约定

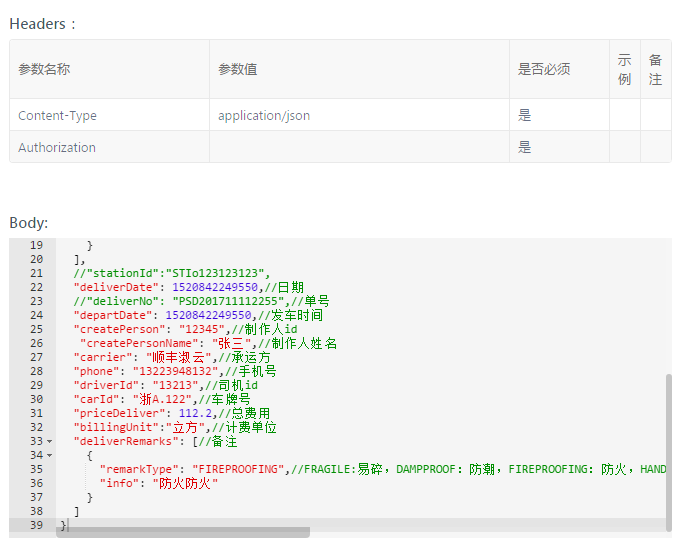
post方法

新增一条XXX  
比如 ……/products 则代表新增一条产品

**入参大于三个的,请使用示例1的json格式,否则使用示例2的键值对格式**入参示例1:json格式1



或者:



java 代码controller层

    @PostMapping (value = "/organization")  
    public ResultObject addOrganization (@RequestBody Organization organization) {  
        //具体逻辑。  
    }

入参示例2:键值对格式



java 代码controller层

    @PostMapping (value = "/organization")  
    public ResultObject addOrganization (String groupId,String menuId,String buttonIdList) {  
        //具体逻辑。  
    }

或者:

    @PostMapping (value = "/organization")  
    public ResultObject addOrganization (Organization organization) {  
        //具体逻辑。  
    }

get方法

查询获得数据,参数形式如下:  
/getProducts?name=xx&pageNum=1&pageSize=50

一个或多个参数,都按如上写法



java代码controller层  
    @GetMapping(value = "/products")  
    public ResultObject getProducts(String sepatateNumber,Integer pageNum) {  
        //具体逻辑。  
    }

或者:

    @GetMapping(value = "/products")  
    public ResultObject getProducts(Products products) {  
        //具体逻辑。  
    }

put方法

修改数据  
/products

入参与POST方式相同

java代码controller层  
    @PutMapping(value = "products")  
    public ResultObject putProducts(@RequestBody Products products) {  
        //具体逻辑。  
    }

delete方法

删除记录,根据id或其他条件进行删除  
/products?name=xx&age=12

或者

/products?id=123

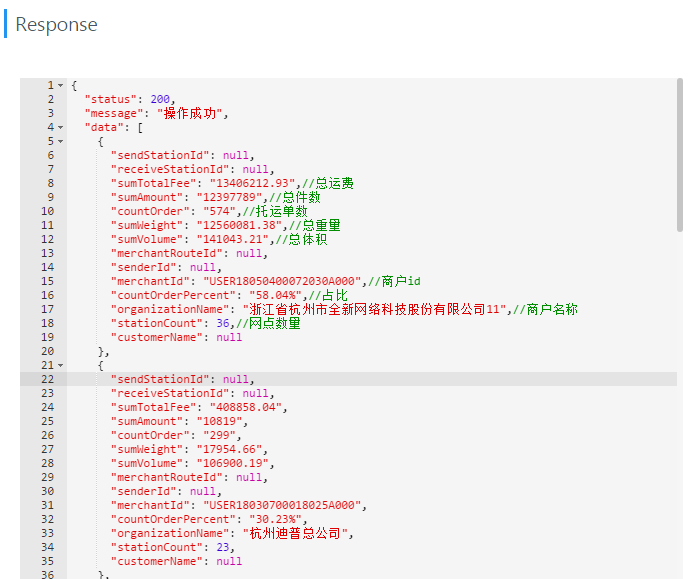
  
java代码controller层  
    @DeleteMapping(value = "/products")  
    public ResultObject deleteProducts(String id) {  
        //具体逻辑。  
    }

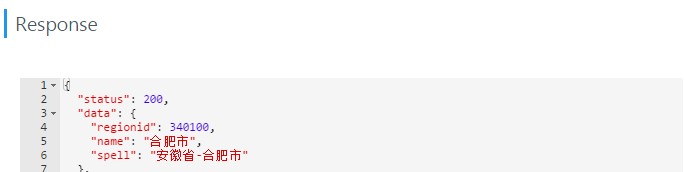
## 5 接口相应格式

所有接口以json格式相应,响应格式如下:

|  |
| --- |
| {  "status": 200, //前端或移动端同学先判断此字段值,200表示正常,其他异常  "message": "ok",//如果status不是200,获取此值显示出来  "data": {//接口返回的数据字段  }  } |

示例:







## 6 数据缓存

为提高接口相应速度，对于不经常变化的数据，必须做数据缓存,请把相关数据缓存到redis中,接口数据从redis中查询后直接返回,不要查询数据库.

## 7接口安全

7.1 token

请使用统一公用token生产器,不要自行实现或使用其他第三方,要使用请先提交架构组评审.

统一token使用请查看wiki文档:

<http://192.168.173.206:8090/pages/viewpage.action?pageId=328865>

7.2 接口鉴权

请使用统一公用api鉴权服务

7.3 外部项目及金融相关项目务必使用https

## 8 接口聚合处理,减少多次请求

首页等或初始化数据页面的接口比较多,虽然不同模块有不同人负责开发，但是有些信息需要合并至一个接口，**由开发经理进行评审后决定是否要精简聚合,如果需要接口聚合的,需要与前端或移动端沟通后再执行.**

## 9 接口默认值及数据格式问题

关于接口默认值，接口有些字段没有数据，返回null非标准数据会导致APP解析崩溃。

规范如下:

如果是字符串类型,字段没有值时, null返回””,如 “name”:””,不要返回”name”:null

如果数组没有值时,返回[],如”list”:[],不要返回”list”:null

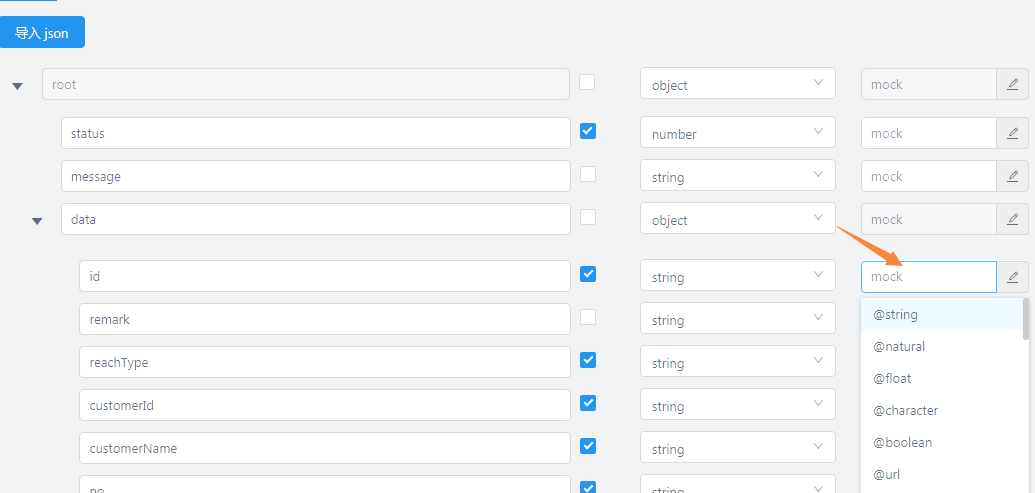
(说明:上面红色这段,已经写了统一处理方法,对json响应进行处理,后台开发人员不需要单独处理,还是按以前的方式写)

各开发经理把统一处理方法添加到自己项目,代码如下:

|  |
| --- |
| import com.alibaba.fastjson.serializer.SerializerFeature;  import com.alibaba.fastjson.support.config.FastJsonConfig;  import com.alibaba.fastjson.support.spring.FastJsonHttpMessageConverter;  import org.springframework.context.annotation.Configuration;  import org.springframework.http.converter.HttpMessageConverter;  import org.springframework.web.servlet.config.annotation.WebMvcConfigurerAdapter;  import java.util.List;  /\*\*  \* json返回null统一处理  \* @author caorui  \* @version v1.0  \* @date 2019年05月16日 14:07:47  \*/  @Configuration  public class WebMvcJsonParseConfig extends WebMvcConfigurerAdapter {  @Override  public void configureMessageConverters(List<HttpMessageConverter<?>> converters) {  FastJsonHttpMessageConverter fastConvert = new FastJsonHttpMessageConverter();  FastJsonConfig fastJsonConfig = new FastJsonConfig();  fastJsonConfig.setSerializerFeatures(  SerializerFeature.WriteNullStringAsEmpty,  SerializerFeature.WriteNullListAsEmpty  );  fastConvert.setFastJsonConfig(fastJsonConfig);  converters.add(fastConvert);  super.configureMessageConverters(converters);  }  } |

如果是数组,按[{},{},{}]这种格式返回,不要返回这种["key1"：{}，“key2”:{}]  
即java数组返回别用map的形式，用面向对象的方式

## 10 YAPI上接口必须添加好Mock数据



接口设计完成,录入YAPI后,必须设置响应的mock数据,保证移动端及前端可以通过先调用mock数据提取联调,提早发现问题,提高效率.(不会使用的请找开发经理指导)

## 11 接口入参参数校验

接口的入参参数必须在后台有业务校验,包括非空及格式校验，接口如果让前端去做判断，容易出现遗漏，也增加前端/移动端的工作量。

## 12 YAPI文档完善

接口在yapi上必须带上一些说明，比如首页获取用户信息，方便前端查找.

有些字段是通用的字段最好统一，尽量不要存在一值多名的情况，方便维护，还有些实现逻辑一样，但却由于不同人负责写出了两种标准，前端都需要适配

## 13 YAPI系统上接口完成状态设置

如果已经完成,并且自测没问题,请设置为”已完成”,表示前端/移动端可以调用,否则表示这个接口还未完成,前端/移动端不要去调用.

## 14接口名称要有版本号



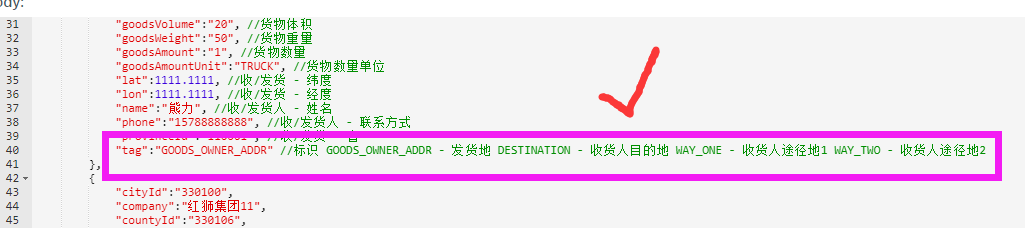
## 15接口传参需要说明下必填项和非必填项





## 16 接口传参若是枚举值，需要把各个枚举的定义写一下





## 17 其他建议

后端开发人员在接口开发完成后，必须根据设计图及测试用例，测试接口是否满足要求.按标准开发流程,先做接口设计及接口设计评审,评审通过后再去写代码实现,开发经理要做好评审流程把控及整体思考,考虑前后关联.

后台开发人员对接口的改动,必须要通知到使用相关接口的前端/移动端人员,未通知而被投诉或发现的,后台开发人员相应的绩效要被扣分.

## 附1:版本历史

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 修改人 | 更新日期 | 备注 |
| v1.0 | 研发中心 | 2019-02-11 | 重新整理形成初稿 |
| v1.1 | 卢磊 | 2019-05-08 | 补充完善 |
|  |  |  |  |