上和国际前后端联调规范

hk196829

目　录

前后端联调规范

RESTful API 规范

规范原则

1. 接口返回数据即显示：前端做渲染及部分逻辑处理；
2. 渲染逻辑禁止跨多个接口调用（3个及以上）；
3. 请求响应传输数据格式：JSON，JSON数据尽量简单轻量，尽量避免多级JSON的出现；

一、版本（Versioning）

应该将API的版本号放入URL。

```
https://api.example.com/v1/
```

另一种做法是，将版本号放在HTTP头信息中，但不如放入URL方便和直观。Github采用这种做法。

二、路径（Endpoint）

路径又称"终点"（endpoint），表示API的具体网址。

在RESTful架构中，资源一般对应服务器端领域模型中的实体类。

路径规范

- 不用大写;
- 用中杠 `-` 而不用下杠 `_` ;
- 参数列表要encode;
- URI中要用名词表示资源集合，使用复数形式;

举例来说，有一个API提供动物园（zoo）的信息，还包括各种动物和雇员的信息，则它的路径应该设计成下面这样。

```
https://api.example.com/v1/zoos
https://api.example.com/v1/animals
https://api.example.com/v1/employees
```

不要使用：

- ~/getAllZoos~
- ~/createNewZoo~

- ~/deleteAllZoos~

三、HTTP动词

对于资源的具体操作类型，由HTTP动词表示。

常用的HTTP动词有下面五个（括号里是对应的SQL命令）。

```
- GET    （SELECT）：从服务器取出资源（一项或多项）。
- POST   （CREATE）：在服务器新建一个资源。
- PUT    （UPDATE）：在服务器更新资源（客户端提供改变后的完整资源）。
- PATCH  （UPDATE）：在服务器更新资源（客户端提供改变的属性）。
- DELETE（DELETE）：从服务器删除资源。
```

还有两个不常用的HTTP动词。

```
- HEAD：获取资源的元数据。
- OPTIONS：获取信息，关于资源的哪些属性是客户端可以改变的。
```

下面是一些例子。

```
- GET    /zoos：列出所有动物园
- POST   /zoos：新建一个动物园
- GET    /zoos/ID：获取某个指定动物园的信息
- PUT    /zoos/ID：更新某个指定动物园的信息（提供该动物园的全部信息）
- PATCH  /zoos/ID：更新某个指定动物园的信息（提供该动物园的部分信息）
- DELETE /zoos/ID：删除某个动物园
- GET    /zoos/ID/animals：列出某个指定动物园的所有动物
- DELETE /zoos/ID/animals/ID：删除某个指定动物园的指定动物
```

四、过滤信息（Filtering）

如果记录数量很多，服务器不可能都将它们返回给用户。API应该提供参数，过滤返回结果。

参数的设计允许存在冗余，即允许API路径和URL参数偶尔有重复。比如，GET /zoo/ID/animals 与 GET /animals?zoo_id=ID 的含义是相同的。

五、状态码（Status Codes）

服务器向用户返回的状态码和提示信息，常见的有以下一些（方括号中是该状态码对应的HTTP动词）。

```
- 200 OK - [GET]：服务器成功返回用户请求的数据，该操作是幂等的（Idempotent）。
- 201 CREATED - [POST/PUT/PATCH]：用户新建或修改数据成功。
- 202 Accepted - [*]：表示一个请求已经进入后台排队（异步任务）
- 204 NO CONTENT - [DELETE]：用户删除数据成功。
```

- 400 INVALID REQUEST - [POST/PUT/PATCH]：用户发出的请求有错误，服务器没有进行新建或修改数据的操作，该操作是幂等的。
- 401 Unauthorized - [*]：表示用户没有权限（令牌、用户名、密码错误）。
- 403 Forbidden - [*] 表示用户得到授权（与401错误相对），但是访问是被禁止的。
- 404 NOT FOUND - [*]：用户发出的请求针对的是不存在的记录，服务器没有进行操作，该操作是幂等的。
- 406 Not Acceptable - [GET]：用户请求的格式不可得（比如用户请求JSON格式，但是只有XML格式）。
- 410 Gone -[GET]：用户请求的资源被永久删除，且不会再得到的。
- 422 Unprocesable entity - [POST/PUT/PATCH] 当创建一个对象时，发生一个验证错误。
- 500 INTERNAL SERVER ERROR - [*]：服务器发生错误，用户将无法判断发出的请求是否成功。

六、错误处理（Error handling）

如果状态码是4xx，就应该向用户返回出错信息。一般来说，返回的信息中将error作为键名，出错信息作为键值即可。

```
{
    "msg": "未知异常，请联系管理员",
    "code": 500,
}
```

七、返回结果

响应基本格式

```
{
    "msg": "success",
    "code": 0,
}
```

响应实体格式

```
{
    "msg": "success",
    "code": 200,
    "data": {
        "user": {
            "roleId": 1,
            "roleName": "管理员",
            "remark": "管理菜单",
            "createUserId": 1,
            "menuIdList": null,
            "createTime": "2018-08-25 14:35:39"
        }
    }
```

```
}
```

响应列表格式

```
{
    "msg": "success",
    "code": 200,
    "data": {
        "list": [{
            "roleId": 1,
            "roleName": "管理员",
            "remark": "管理菜单",
            "createUserId": 1,
            "menuIdList": null,
            "createTime": "2018-08-25 14:35:39"
        }]
    }
}
```

响应分页格式

```
{
    "msg": "success",
    "code": 200,
    "data": {
        "page": {
            "total": 1, // 总条目数
            "pageSize": 10,  // 每页显示个数
            "currentPage": 1, // 当前页数
            "list": [{
                "userId": 1,
                "username": "admin",
                "password": "9ec9750e709431dad22365cabc5c625482e574c74adaebba7d
d02f1129e4ce1d",
                "salt": "YzcmCZNvbXocrsz9dm8e",
                "email": "root@dhcc.com.cn",
                "mobile": "18018018018",
                "status": 1,
                "roleIdList": null,
                "createUserId": 1,
                "createTime": "2016-11-11 11:11:11"
            }]
        }
    }
}
```

特殊内容规范

下拉框、复选框、单选框

由后端接口统一逻辑判定是否选中，通过checked标示是否选中，示例如下：

```
{
    "msg": "success",
    "code": 200,
    "data": {
        "list": [{
            "id": 1,
            "name": "XXX",
            "code": "XXX",
            "checked": 1
        }, {
            "id": 1,
            "name": "XXX",
            "code": "XXX",
            "checked": 0
        }]
    }
}
```

禁止下拉框、复选框、单选框判定选中逻辑由前端来处理，统一由后端逻辑判定选中返回给前端展示；

Boolean类型

关于Boolean类型，JSON数据传输中一律使用1/0来标示，1为是/True，0为否/False；

日期类型

关于日期类型，JSON数据传输中一律使用字符串，具体日期格式因业务而定；

八、认证

1. API的身份认证可选方案
2.
    1. JWT
3.
    2. 数据签名

九、数据格式

服务器返回的数据格式，统一使用JSON，避免使用XML。

十、其他规范

1. 后端返回状态码统一为number类型

```
示例
{
    code:200
    data:{}
    msg:"成功"
}
```

2. 后端时间不做处理，前端统一处理

```
示例：
{
    date:'19092929828221'
}
```

3. 后端设置数据字典并及时更新给前端
4. 后端的id统一使用Long类型

```
---------------------------------
{
code : 200
msg::"成功"
data：{
    id：12345678987456321
    }
}
```

5. 后端返回错误信息

```
{
    code:401
    data:{}
    msg:："暂无权限访问"
}
```

6. 考虑使用黑词库

```
例如：污秽词语过滤
```

7. 后端文档采用swagger 及时更新
8. 前端请求方式统一使用async+try catch的方式

```
async getDoctorInfo () {
```

```
    try {
        const result = await this.$http.post(`/api-patient/doctor/info/${sessio
nStorage.getItem('chainId')}/${this.$route.query.id}`)
        this.detailed = result
        if (result.doctorFees[0].operatingState !== 1) {
            this.canQuestion = false
        }
    } catch (e) {
        throw e
    }
}
```

9. 文件资源下载是否是直接一个get请求（window.open）
10. 文件导入是支持xls和xlsx
11. 前端做excel表格导出
12. 前后端交互字段全部使用小驼峰方式

```
post :{
    userName:："张三"
}
```

13. 接口文档对应字段应和后端返回一致
14. 后端返回数据时不要用null

```
{
    code:200
    data:{[]}
    msg:：“成功"
}
```

15. 接口功能独立应分拆成新的接口
16. 返回字段须有msg,必须有值

```
{
    code:401
    data:{}
    msg:："暂无权限访问"
}
```

17. 返回数据中图片 URL 是完整的（后端做拼接）

```
{
    code:200
```

```
    data:{
        headImg::"http:/189.23.1.07:8080/headImg/1.jpg"
    }
    msg:："成功"
}
```

18. 后端处理电话号码以及证件信息，前端做模糊显示（比如说用*）

```
后端返回：
{
    code：200
    data:{
        phone:18908180999
    }
    msg:："成功"
}
----------------------------------------
前端显示 ：电话号码：189****0999
```

19. 特殊需求特殊处理