**3.开发规范**

# 目录规范

## 启动文件

app.py

## 配置文件

config.py

## 控制层

controllers

## 模型层

models

## 业务逻辑层

services

## 工具包

tools

## 校验文件

validators

## 虚拟环境

venv

# 命名规范

## 项目名称

全小写，或者使用-分割

如：flaskapp、flask-app

## 模块名和包名

全部小写+下划线驼峰  
module\_name package\_name

## 类名称，异常

首字母大写+大写式驼峰，  
class ClassName: ，ExceptionName

## 全局变量、常量

全部使用大写字母+下划线驼峰  
GLOBAL\_VAR\_NAME，CONSTANT\_NAME

## 方法名，函数名，其余变量，参数，实例

全部小写+下划线驼峰  
method\_name，function\_name，instance\_var\_name, function\_parameter\_name, local\_var\_name

# 接口规范

前端与后端人员约定的一个数据交换的一种规范。

## 数据格式-JSON

{  
 "id": 1,  
 "userName": "admin",  
 "userType": 1  
}

## 后端统一返回结构规范

{  
 "code": 0,  
 "msg": "登录成功",  
 "data": {}  
}

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 描述 |
| code | int | 状态码，0为成功，其他为失败 |
| msg | String | 消息描述 |
| data | ｏｂｊｅｃｔ | 业务数据 |

## 返回规范

### 提交类

{  
 "code": 0,  
 "msg": "登录成功",  
 "data": {}  
}

### 单条记录

{  
 "code": 0,  
 "msg": "登录成功",  
 "data": {  
 "id": 1,  
 "userName": "admin",  
 "userType": 1  
 }  
}

### 多条记录

{  
 "code": 0,  
 "msg": "登录成功",  
 "data": [  
 {  
 "id": 1,  
 "userName": "admin1",  
 "userType": 1  
 },  
 {  
 "id": 2,  
 "userName": "admin2",  
 "userType": 1  
 }  
 ]  
}

### 分页记录

{  
 "code": 0,  
 "msg": "登录成功",  
 "data": {  
 "recordCount": 100,  
 "totalPage": 10,  
 "pageSize": 10,  
 "pageNum": 1,  
 "rows": [  
 {  
 "id": 1,  
 "userName": "admin1",  
 "userType": 1  
 },  
 {  
 "id": 2,  
 "userName": "admin2",  
 "userType": 1  
 }  
 ]   
 }  
}

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 描述 |
| recordCount | long | 总记录数 |
| totalPage | int | 总页数 |
| pageSize | int | 每页的条数 |
| pageNum | int | 当前页，默认值为1 |

## 提交的规范

### id提交类

请求正文JSON

{  
 "id": 1  
}

### ids提交类

{  
 "ids": [1,2,3]  
}

### 审核提交类

{  
 "ids": [1,2],  
 "rejectReason":"驳回原因",  
 "pass": 0  
}

### 实体提交类

{  
 "id": 1,  
 "userName": "admin",  
 "userType": 1,  
 "password": "123456"  
}

### 通用查询类

{  
 "pageNum": 1,  
 "pageSize": 15,  
 "m\_LIKE\_userName": "admin",  
 "m\_BT\_createTime": ["2020-01-01","2020-06-06"]  
}

无表别名：m\_EQ\_userName

有表别名：m\_t\_EQ\_userName

**操作符说明**

|  |  |  |
| --- | --- | --- |
| 操作名 | 说明 | 传参类型 |
| EQ | 等于= | string |
| NE | 不等于<> | string |
| GT | 大于> | string |
| GE | 大于等于>= | string |
| LT | 小于< | string |
| LE | 小于等于<= | string |
| BT | between 值1 and 值2 | array |
| NBT | not between 值1 and 值2 | array |
| LIKE | like '%值%' | string |
| NLIKE | not like '%值%' | string |
| LLIKE | like '%abc' | string |
| RLIKE | like 'abc%' | string |
| IN | in(值1,值2) | array |
| NIN | not in(值1,值2) | array |

# 数据库规范

## 关于表名

表名使用下划线。例：模块名\_表名。主表中必要字段（id/create\_time/update\_time/is\_deleted,如为父子关系表,新增字段(parent\_id/sort/name)

如果系统比较小，可以直接使用类似这样子的命名规范。

t\_user->用户表

t\_role->角色表

t\_category->文章分类表

t\_article->文章表

如果系统略大，可以对表再进行功能分类。

* sys->系统管理

1. sys\_user->用户表
2. sys\_role->角色表

* cms->内容管理

1. cms\_category->文章分类表

2. cms\_article->文章表