

API 管理平台操作手册

1 引言

1.1 编写目的

方便前端和后端同学更好的使用 API 管理平台。

1.2 项目背景

- 1、前后端同学合作时，定义接口不能规范化，增加了大量沟通时间。
- 2、前端同学调试代码时，必须依赖后端系统返回测试数据，大大降低开发效率。
- 3、测试同学在测试过程中产生 bug，不能确认是由于前端还是后端代码引发的，提 bug 时需要转发再转发。

1.3 参考资料

mock: <http://mockjs.com/>

2 软件概述

2.1 目标

能让前后端的小伙伴们能更愉快的一起写代码。

2.2 功能

- 1、后端同学编写接口文档，详细说明接口的参数信息。
- 2、前端同学查看接口文档，并通过平台自动模拟生成后端返回的测试数据。
- 3、校验前端同学请求的参数是否正确。
- 4、校验后端同学返回的结果是否正确。

3 运行环境

3.1 硬件

- 不详

3.2 软件

- linux
- nodejs
- mongodb
- mock.js

4 使用说明

4.1 用户管理

4.2 团队管理

4.3 项目管理

4.4 PRD 管理

4.5 API 管理

4.5.1 进入 API 管理平台

- 1、在我的 PRD 页面里，点击 **api**，

创建PRD

操作	版本号	主要功能	项目类型	阶段	提测时间	上线时间	产品	自测	JIRA地址	备注
<div>api详情更新删除</div>	1.2	交易进度子任务		已上线	2016-07-08	2016-07-08				

<

1

>

2、进入 api 管理平台页面。

全部团队 / 微信企业号 / 端口管理 / PRD: v2.0

当前项目URL: http://fe.iwjw.com:8888/api/fete_api/9RWVq/mock/
当前PRD URL: http://fe.iwjw.com:8888/api/fete_api/9RWVq/IS3RW2/mock/
工程名: /weixinEnt

描述	链接	方法	最后修改	返回数据预览
[端口管理]-[端口申请]-获取端口有效操作列表	/port/openway	GET	2016-08-04 13:08:12 lancui	预览
端口扣费问题反馈跳转	/portCost/getFeedbackUrl	GET	2016-08-04 11:04:39 lancui	预览
端口获得历史六个月时间	/portCost/getPortCostDates	GET	2016-08-04 11:04:35 lancui	预览
经纪人消费端口每月结算数据详情	/portCost/getPortCostMonthDetail	GET	2016-08-04 11:04:31 lancui	预览
[端口管理]-[我的端口]-获取待开通、已开通列表	/port/myport.do	GET	2016-08-04 11:04:26 lancui	预览
[端口管理]-[端口申请]-提交(充值)	/port/recharge	GET	2016-08-04 11:04:21 lancui	预览
[端口管理]-[端口申请]-提交(刷新)	/port/fresh	GET	2016-08-04 11:04:17 lancui	预览

4.5.2 新增 API 接口

1、点击 新建API 按钮，弹出新增窗口：

全部团队 / 微信企业号 / 端口管理 / PRD: v2.0

当前项目URL: http://fe.iwjw.com:8888/api/fete_api/9RWVq/mock/
当前PRD URL: http://fe.iwjw.com:8888/api/fete_api/9RWVq/IS3RW2/mock/
工程名: /weixinEnt

新建API

2、点击新增行，左侧弹出编辑页面编辑

设置返回的数据是输入的数据还是通过mock生成的数据

* 标题: [端口管理]-[端口申请]-获取端口有效操作

* URL (首字符请输入): /port/openway

method: get

* 返回output数据: ☐

* 输入数据格式

```
1 {
2   "siteId": 23232,
3   "type": 2
4 }
```

请求参数

属性	数据类型	含义	是否必须
siteId	Number		<input checked="" type="checkbox"/> 必需
type	Number		<input checked="" type="checkbox"/> 必需

* 返回数据格式

```
1 {
2   "status": 1,
3   "message": "服务器异常",
4   "data": {
5     "ports": [
6       {
7         "id": 2,
8         "type": 1,
9         "costPrice": 300,
10        "versionInfo": "满信件还却收求外先白命型再。",
11        "newOpenStatus": 2,
12        "copy": {
13          "siteId": 2
14        }
15      }
16    ]
17  }
18 }
```

返回数据格式

mock->

帮助 上一条 确定 删除 取消 下一条

属性	数据类型	含义	mock规则
status	Number		:1
message	String		:success
- data	Object		:1
versionTitle	String		:@ctitle()
versionInfo	String		:@csentence()
portAccount	Number		:139177812937
siteName	String		1:['58','安居','乐居','赶集','搜房','推99']

修改说明

接口修改说明

接口修改日志

lancui

2016-08-04

葛袁军

2016-07-13 add siteName

更多

帮助

上一条

确定

删除

取消

下一条

输入

- 标题：接口的中文说明
- URL：请求地址，不包括工程名
- Method：接口请求方式：GET, POST, PUT, DELETE
- 返回 output 数据：设置返回的数据是输入的数据，还是通过 mockjs 生成的数据
- 输入数据格式：接口的请求参数，json 格式
- 返回数据格式：返回的数据格式，json 格式
- 表格：含义：属性说明。
- 表格：mock 规则：参考 <http://mockjs.com/>, 输入 mock 规则。
 - 写入方法：从 “|” 开始写到最后，

```
Mock.mock({
  "string|1-10": "*"
})
```

- 1、字符串：写成：| 1-10: " *", 省略掉冒号左边的引号！

```
Mock.mock({
  "array|1-10": [
    {
      "name|+1": [
        "Hello",
        "Mock.js",
        "!"
      ]
    }
  ]
})
```

- 2、集合对象：写成：data: | 1-10: , (最后的 “:” 也要保留)
name: |+1: [“Hello”, “Mock.js”, “!”]

- 3、对象：Object 类型，只需要写 “:”

```
Mock.mock({
  "array|1-10": [
    "Mock.js"
  ]
})
```

- 4、数组值：Array 类型：如，写成：| 1-10: [“mock.js”]

```
Mock.mock({
  "boolean|1": true
})
```

- 5、Boolean：写成：| 1: true

如：

属性	数据类型	含义	mock规则
status 数字	Number	状态 (0: 成功, 1:失败)	[1:[0,1]
msg 字符串	String	返回结果说明	[1:['成功', '失败']
type 数组值	Array	类型	[1-2:['lcz', 'ry', 'cy']
- data 对象集合	Array	返回数据	[1-3:
taskId	String	taskId	[+1:['4', '5', '6']
taskName	String	task名称	[1:['受理', '查限购', '出押']
- user 对象	Object	用户对象	:
id	String	用户ID	:@word(32)
name	String	用户名	:@cname

➤ mock 帮助。
点击 **帮助** 按钮，显示常用的 mock 规则，点击规则，自动复制到剪贴板。

string

|2:

随机2个字符

|2-6:

随机2-6个字符

:@city()

随机城市名

:@word(1,10)

随机1-10个字符

:@cword(1,4)

随机1-6个汉字

:@cname()

姓名

:@date("yyyy-MM-dd")

日期字符

:@time()

时间

:@ip()

ip地址

:/^13[0-9][0-9][8][15][89]

手机号

Number

|123.3:

保留3位小数

|123.1-10:

随机保留1-10位小数

|1-100.1-10:

范围1-100,1-10位小数

|1:|1,2,3,4|

随机数组中的1位

:100

固定的增量

Boolean

:false

false

|1-2:

true or false

Array or Object

|4:

4次

|1-4:["m"]

随机1-4次m

帮助

上一条

➤ 修改说明：每次改动时，必须要填写修改说明。

• 修改说明

接口修改说明

接口修改日志

lancui

2016-08-04

lancui

2016-08-04

更多

点击 **确定** 按钮保存。

4. 5. 3 修改 API 接口

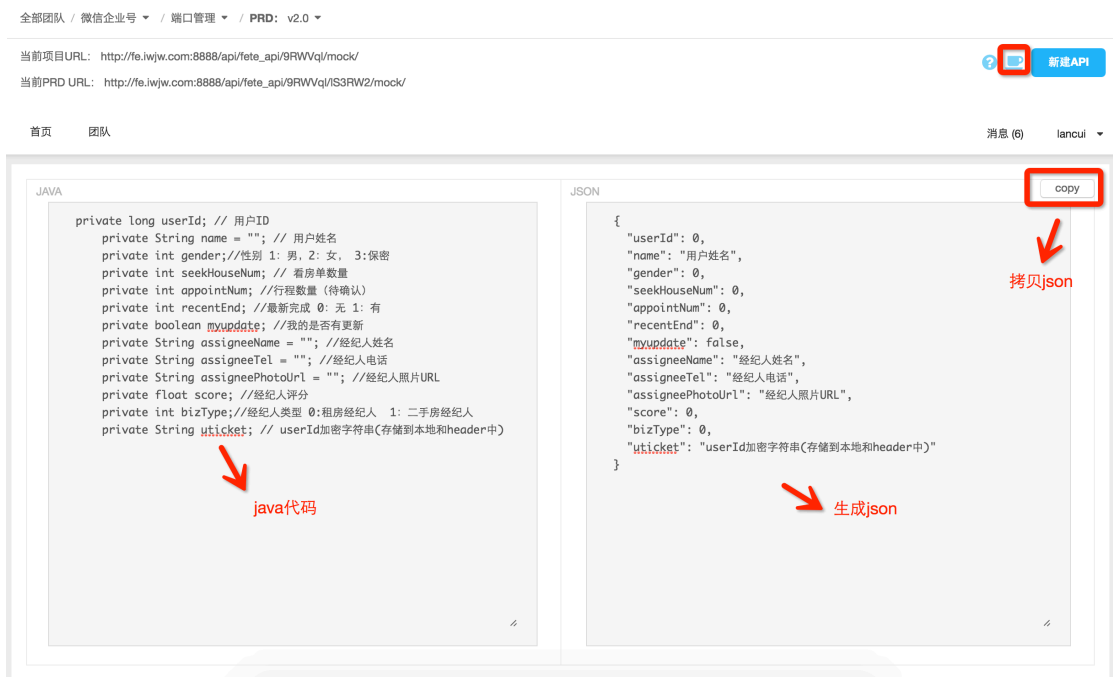
点击列表数据行，左侧弹出 API 详情，修改信息后，点击 **确定** 按钮保存。

4. 5. 4 删除 API 接口

点击列表数据行，左侧弹出 API 详情，点击 **删除** 删除。

4. 5. 5 java 类属性转 json

点击 **🔗** 按钮，进入 java 转 json 页面，粘贴 java 代码，鼠标移出，生成 json 对象。



4.5.6 查看提醒

点击右上角 **消息 (6)**，进入消息列表：

<div><div>全部已读</div><div>批量删除</div></div>						
<input type="checkbox"/> 全选	时间	用户名	平台	操作	描述	状态
<input type="checkbox"/>	2016-08-22T03:50:25.582Z	lancui	team	invited	lancui邀请你加入"房源宝企业号"	<div>接受拒绝</div>
<input type="checkbox"/>	2016-08-19T03:56:10.901Z	葛袁军	api	update	modify 1	<div>未读</div>
<input type="checkbox"/>	2016-08-19T03:46:14.266Z	葛袁军	api	update	修改返回值	<div>未读</div>
<input type="checkbox"/>	2016-08-19T03:28:31.671Z	葛袁军	api	update	modify data structure	<div>未读</div>
<input type="checkbox"/>	2016-08-19T03:21:22.257Z	葛袁军	api	update	modify status	<div>未读</div>
<input type="checkbox"/>	2016-08-19T03:20:51.546Z	葛袁军	api	update	modify status	<div>已读</div>
<input type="checkbox"/>	2016-08-19T03:19:58.904Z	葛袁军	api	update	modify status	<div>已读</div>

点击 **接受**、**拒绝**、**未读** 按钮，标记为已读。

4.6 校验请求参数和返回参数正确性。

4.6.1 用法（两种方法选一种）

```
// 1: gulp api --task=vue --mock --dev --pjid=123456
说明: --task: 必填, task 的名称
      --mock: 选填, 是否使用 fete 生成的数据
      --dev: 选填, 是否把 mock_check.js 打包到 common 或者 vue-common 里
      --pjid: 选填, 对应 fete 里的 projectId, 如果项目里设置了 pjid, 就不必填

// 2: 在 package.json 里的 script 里面新建项, 把上面的命令当值; 然后使用 npm
start [name] 来启动
```

4.6.2 以下代码加入 gulpfile.js 中:

```
gulp.task('api', function() {
  // 必要参数的检查
  if (!argv.pjid) {
    console.error('必须加 --pjid=[对应的 projectId] ')
    return false
  }

  // 如果项目目录下没有 .tmp 文件夹，下面到代码会新建一个 .tmp 文件夹。
  // 注意：需要自己手动把它添加到 .gitignore 里面
  var fs = require('fs')
  var tmpDir = fs.statSync('./.tmp')
  if (!tmpDir.isDirectory()){
    fs.mkdirSync('./.tmp')
  } else {
    fse.emptydirSync('./.tmp')
  }

  // 发 http 请求去动态下载 mock_check.js 文件
  var http = require('http')
  var file = fs.createWriteStream('./tmp/mock_check.js', { flags: 'w' })
  var request = http.get(`http://fe.superjia.com:8888/api/mock_check.js?useMockData=${argv.mock ? true : false}&projectId=${argv.pjid}`, function(response) {
    var stream = response.pipe(file)
    stream.on('finish', function() {
      // js 文件下载保存完成之后，判断是否有 --task 参数，有到话启动相应的任务
      if (argv.task) {
        gulp.start(argv.task)
      }
    })
  })
})
})
```

```

// 这里是用 --dev 参数来决定是否把 mock_check.js 文件一起打包
// 这段代码需要在 webpack 使用 config 之前
if(argv.dev) {
    config.entry.common.push('./.tmp/mock_check.js');
}

```

```

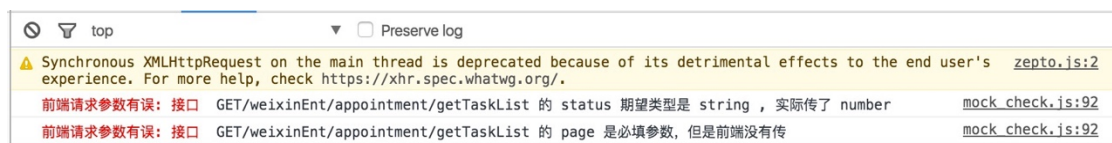
// 这里是用 --dev 参数来决定是否把 mock_check.js 文件一起打包
// 这段代码需要在 webpack 使用 config 之前
if (argv.dev) {
    config.entry.common.push('./.tmp/mock_check.js');
}
webpack(config, function (err, stats) {
    console.log(stats.toString());
});

```

在这个点上面加那个 if 语句

4.6.3 校验请求参数和返回结果数据的显示（用浏览器调试时）

➤ 请求参数有误：



➤ 返回结果校验：

