

# 悠悠课程项目接口文档



作者: 上海-悠悠

联系微信: 283340479

Blog: <https://www.cnblogs.com/yoyoketang/>

(仅供内部学习使用, 勿外传, 违者必究!)

## 目录

用户注册.....	4
用户注册 /api/v1/register .....	4
功能描述.....	4
请求示例.....	5
返回示例.....	5
用户登录接口.....	6
用户登录 /api/v1/login .....	6
功能描述.....	6
请求示例.....	7
返回示例.....	8
用户登录加密 /api/v2/login .....	8
功能描述.....	8
AES/CBC/PKCS7Padding 加密规则 .....	8
请求示例: .....	12

返回示例.....	13
用户登录签名 /api/v3/login .....	13
功能描述.....	14
Sign 签名规则 .....	14
请求示例: .....	16
返回示例: .....	17
用户登录 /api/v4/login .....	17
功能描述.....	17
请求示例.....	18
返回示例.....	18
商品信息接口.....	19
查询商品/api/v1/goods/{id}.....	19
功能描述.....	19
请求示例.....	20
返回示例: .....	20
新增商品 /api/v1/goods.....	21
功能描述: .....	22
请求示例: .....	23
返回示例: .....	24
修改商品 /api/v1/goods/{id}.....	25
功能描述: .....	25
请求示例.....	27
返回示例: .....	28
删除商品 /api/v1/goods/{id}.....	29
功能描述: .....	29
请求示例.....	30
返回示例.....	30
分页查询全部商品 /api/v1/goods?page=1&size=2 .....	31
功能描述: .....	31
请求示例.....	32
返回示例: .....	32
需登录才能访问的 v2 接口.....	33
查询单个商品 /api/v2/goods/{id}.....	33
新增商品 /api/v2/goods.....	34
修改商品 /api/v2/goods/{id}.....	35
删除商品 /api/v2/goods/{id}.....	36
分页查询 /api/v2/goods?page=1&size=2 .....	36
我的购物车接口.....	36
查询我的购物车 /api/v2/carts.....	36
功能描述.....	36
请求示例.....	37
返回示例.....	37
添加购物车 /api/v2/carts.....	38

功能描述.....	39
请求示例.....	40
返回示例.....	40
订单管理.....	41
查询我的全部订单 /api/v2/orders .....	41
功能描述.....	41
请求示例.....	42
返回示例.....	42
创建订单 /api/v2/orders .....	44
功能描述.....	44
请求示例.....	46
返回示例.....	46
删除订单 /api/v2/orders .....	47
功能描述.....	47
请求示例.....	48
返回示例.....	49
收获地址.....	49
查询收获地址 /api/v2/address .....	49
功能描述: .....	49
请求示例.....	50
返回示例.....	50
添加和修改收获地址 /api/v2/address .....	51
功能描述: .....	51
请求示例.....	52
返回示例.....	53
个人资料.....	54
查询个人资料 /api/v1/userinfo .....	54
功能描述: .....	54
请求示例.....	55
返回示例.....	55
修改/添加个人资料 /api/v1/userinfo.....	56
功能描述: .....	56
请求示例.....	57
返回示例.....	57
文件上传.....	58
上传文件 /api/v1/upfile/.....	58
功能描述: .....	58
请求示例.....	59
返回示例.....	60

# 用户注册

## 用户注册 /api/v1/register

接口名称: 用户注册

接口地址: /api/v1/register

请求方式: POST

功能描述: 请求 body

参数类型: Content-Type: application/json

请求参数:

字段名	变量名	必填	类型	示例值	描述
用户名	username	是	String	test	用户账户 长度3-30 位
用户密码	password	是	String	123456	密码长度 6-16 位
邮箱	email	否	String	123@qq.com	检验邮箱合法性

返回参数:

字段名	变量名	类型	示例值	描述
返回错误码	code	Int	0	0 为成功, 其它值为失败  3003 参数不合法

				2000 用户已被注册
返回消息	msg	String	register success!	为 0 时返回 register success!
数据	data	object	{}	返回{}

## 请求示例

POST http://localhost:8000/api/v1/register

Content-Type: application/json

```
{  
  "username": "test",  
  "password": "123456"  
}
```

## 返回示例

```
{  
  "code": 0,  
  "msg": "register success!",  
  "datas": {}  
}
```

```
}

```

# 用户登录接口

## 用户登录 /api/v1/login

**接口名称：**用户账户登录

**接口地址：**/api/v1/login

**请求方式：**POST

### 功能描述：

用户账户登录，服务端返回 token 请求 body

**参数类型：**Content-Type: application/json

### 请求参数：

字段名	变量名	必填	类型	示例值	描述
用户名	username	是	String	test	用户账户 长度3-30 位
用户密码	password	是	String	123456	密码长度 6-16 位
邮箱	email	否	String	123@qq.com	检验邮箱合法性

### 返回参数:

字段名	变量名	类型	示例值	描述
返回错误码	code	Int	0	0 为成功, 其它值为失败  3003 参数不合法  2000 用户已被注册
返回消息	msg	String	success!	0 时 返回 login  success
用户名	username	String	test	返回当前登录的用户名
Token	token	String	e8329c6f25ac892  6773b0eb7313ce2  5ffa6a4a73	返回 40 位 token

### 请求示例:

POST http://localhost:8000/api/v1/login

Content-Type: application/json

```
{  
  
  "username": "test",  
  
  "password": "123456"  
}
```

返回示例:

```
{  
  "code": 0,  
  "msg": "login success!",  
  "username": "test",  
  "token": "a52a9d985723ba9af67619f4eb0d3182c2856f52"  
}
```

## 用户登录加密 **/api/v2/login**

**接口名称:** 用户账户登录

**接口地址:** /api/v2/login

**请求方式:** POST

### 功能描述

用户账户登录, 服务端返回 token

请求 body 参数类型: Content-Type: application/json

## **AES/CBC/PKCS7Padding** 加密规则

关于加密参考这篇 <https://www.cnblogs.com/yoyoketang/p/11717282.html> python

代码 AES/CBC/pkcs7padding 加解密



```

from cryptography.hazmat.primitives import padding
from cryptography.hazmat.primitives.ciphers import algorithms
from Crypto.Cipher import AES
from binascii import b2a_hex, a2b_hex
import json

'''
AES/CBC/PKCS7Padding 加密解密
环境需求:
pip3 install pycryptodome==3.9.0
pip3 install cryptography
不要安装 Crypto
'''

class PrpCrypt(object):

    def __init__(self, key='0000000000000000'):
        self.key = key.encode('utf-8')
        self.mode = AES.MODE_CBC
        self.iv = b'0102030405060708'
        # block_size 128 位

        # 加密函数, 如果 text 不足 16 位就用空格补足为 16 位,
        # 如果大于 16 但是不是 16 的倍数, 那就补足为 16 的倍数。
        def encrypt(self, text):
            cryptor = AES.new(self.key, self.mode, self.iv)
            text = text.encode('utf-8')

            # 这里密钥 key 长度必须为 16 (AES-128), 24 (AES-192), 或者 32 (AES-256) Bytes 长度
            # 目前 AES-128 足够目前使用

            text=self.pkcs7_padding(text)

            self.ciphertext = cryptor.encrypt(text)

            # 因为 AES 加密时候得到的字符串不一定是 ascii 字符集的, 输出到终端或者保存时候可能存
            # 在问题
            # 所以这里统一把加密后的字符串转化为 16 进制字符串
            return b2a_hex(self.ciphertext).decode().upper()

    @staticmethod

```

```

def pkcs7_padding(data):
    if not isinstance(data, bytes):
        data = data.encode()

    padder = padding.PKCS7(algorithms.AES.block_size).padder()

    padded_data = padder.update(data) + padder.finalize()

    return padded_data

@staticmethod
def pkcs7_unpadding(padded_data):
    unpadder = padding.PKCS7(algorithms.AES.block_size).unpadder()
    data = unpadder.update(padded_data)

    try:
        upadded_data = data + unpadder.finalize()
    except ValueError:
        raise Exception('无效的加密信息!')
    else:
        return upadded_data

# 解密后, 去掉补足的空格用 strip() 去掉
def decrypt(self, text):
    # 偏移量'iv'
    cryptor = AES.new(self.key, self.mode, self.iv)
    plain_text = cryptor.decrypt(a2b_hex(text))
    # return plain_text.rstrip('\0')
    return bytes.decode(plain_text).rstrip("\x01").\
        rstrip("\x02").rstrip("\x03").rstrip("\x04").rstrip("\x05").\
        rstrip("\x06").rstrip("\x07").rstrip("\x08").rstrip("\x09").\
        rstrip("\x0a").rstrip("\x0b").rstrip("\x0c").rstrip("\x0d").\
        rstrip("\x0e").rstrip("\x0f").rstrip("\x10")

def dict_json(self, d):
    '''python 字典转 json 字符串, 去掉一些空格'''
    j = json.dumps(d).replace(':', ' ', ':').replace(', ', ',').replace(", {",
    ", {")
    return j

# 加解密
if __name__ == '__main__':
    import json

    pc = PrpCrypt('12345678\0\0\0\0\0\0\0\0') # 初始化密钥

```

```
a = "1"

print("加密前: %s" % a)

b = pc.encrypt(a)

print("解密后: %s" % b)

print("大写变小写: %s" % b.lower())
```

请求参数 (加密前)

字段名	变量名	必填	类型	示例值	描述
用户名	username	是	String	test	用户账户 长度3-30 位
用户密码	password	是	String	123456	密码长度 6-16 位

请求参数 (加密后)

字段名	变量名	必填	类型	示例值	描述
参数	params	是	String	3DBF6BE0F8549A98AA81B62 9A028E487BA2EBE85EF1BC9 FE7105F5FE833F1DF26F022 E404EFBDAD5A5DF1A7B7FED A16C	加密请求  body

返回参数:

字段名	变量名	类型	示例值	描述
返回错误码	code	Int	0	0 为成功, 其它值为失败  3003 参数不合法  2000 用户已被注册
返回消息	msg	String	success!	0 时返回 login success

息				
数据	data	String	加密串	{ "username": "test", "token": "a003442ffc9645af181d8c768bd8758a250ba6d6" } 返回加密后的值

## 请求示例：

POST http://localhost:8000/api/v2/login

Content-Type: application/json

```
{
  "params": "3DBF6BE0F8549A98AA81B629A028E487BA2EBE85EF1BC9FE7105F5FE833F1DF26F022E404EFBDAD5A5DF1A7B7FEDA16C"
}
```

## 未加密前请求：

```
{
  "params": {
    "username": "test",
    "password": "123456"
  }
}
```

## 返回示例

```
{
  "code": 0,
  "msg": "login success!",
  "datas": "D7AD39D4AB776EA75F4B0EA1E8BFF211DB6E3AC48704A97D6
B6A3C8814EE0D43CB1F4D3E2C66C126C251840F0666D558C31BFCAF174C00F89E
B0784789A3E83C7626CC40629E109401A79FE860FFFD65"
}
```

### 解密返回数据

```
{
  "code": 0,
  "msg": "login success!",
  "datas": {
    "username": "test",
    "token": "a003442ffc9645af181d8c768bd8758a250ba6d6"
  }
}
```

## 用户登录签名 /api/v3/login

**接口名称：**用户账户登录

**接口地址：**/api/v3/login

请求方式: POST

## 功能描述

用户账户登录, 服务端返回 token 请求 body

参数类型: Content-Type: application/json

## Sign 签名规则

关于签名参考这篇 <https://www.cnblogs.com/yoyoketang/p/11742187.html>

签名参数 sign 生成的方法

- 第 1 步: 将所有参数 (注意是所有参数), 除去 sign 本身, 以及值是空 的参数, 按参数名字母升序排序。
- 第 2 步: 然后把排序后的参数按参数 1 值 1 参数 2 值 2...参数 n 值 n (这 里的参数和值必须是传输参数的原始值, 不能是经过处理的, 如不能将" 转成" 后再拼接) 的方式拼接成一个字符串。
- 第 3 步: 把分配给接入方的验证密钥 key 拼接在第 2 步得到的字符串 key。

在上一步得到的字符串后面加上验证密钥 key(这里的密钥 key 是接口提供方分配给接口接入方的), 然后计算 md5 值, 得到 32 位字符串, 然后转成大写。

- 第 4 步: 计算第 3 步字符串转小写后得到的 md5 值(32 位), 得到的字符串作为 sign 的值

假 设 传 输 的 数 据 是

[http://www.xxx.com/interface.aspx?sign=sign\\_value&p2=v2&p1=v1&method=ca](http://www.xxx.com/interface.aspx?sign=sign_value&p2=v2&p1=v1&method=ca)

ncel&p3=&pn=vn

(实际情况最好是通过 post 方式发送),

其中 sign 参数对应的 sign\_value 就是签名的值。

第一步，拼接字符串，首先去除 sign 参数本身，然后去除值是空的参数 p3，剩下  
p2=v2&p1=v1&method=cancel&pn=vn,

然后按参数名字符升序排序，method=cancel&p1=v1&p2=v2&pn=vn.

第二步，然后做参数名和值的拼接，最后得到 methodcancelp1v1p2v2pnnvn

第三步，在上面拼接得到的字符串后加上验证密钥 key，我们假设是 abc，得到新的字符串  
methodcancelp1v1p2v2pnnvnabc

第四步，然后将这个字符串换为小写进行 md5 计算，假设得到的是 abcdef，这个值即为  
sign 签名值。

注意，计算 md5 之前请确保接口与接入方的字符串编码一致，如统一使用 utf-8 编码或者  
GBK 编码，如果编码方式不一致则计算出来的签名会校验失败。

请求参数

字段名	变量名	必填	类型	示例值	描述
用户名	username	是	String	test	用户账户 长度 3-30 位
用户密码	password	是	String	123456	密码长度 6-16 位
签名	sign	是	String	1aca01806e93bb408041965 a817666af	参考 Sign 签 名规则

### 返回参数:

字段名	变量名	类型	示例值	描述
返回错误码	code	Int	0	0 为成功, 其它值为失败  3003 账号或密码错误  3005 签名不合法
返回消息	msg	String	success!	0 时返回 login success
用户名	username	String	test	返回当前登录的用户名
Token	token	String	e8329c6f25ac892  6773b0eb7313ce2  5ffa6a4a73	返回 40 位 token

### 请求示例:

POST http://localhost:8000/api/v3/login

Content-Type: application/json

```
{  
  
  "username": "test",  
  
  "password": "123456",  
  
  "sign": "1aca01806e93bb408041965a8176661af"  
}
```



返回示例：

```
{
  "code": 0,
  "msg": "login success!",
  "username": "test",
  "token": "a52a9d985723ba9af67619f4eb0d3182c2856f52"
}
```

用户登录 **/api/v4/login**

**接口名称：**用户账户登录

**接口地址：**/api/v4/login

**请求方式：**POST

功能描述

用户账户登录，服务端返回 token 请求 body

参数类型：Content-Type: application/x-www-form-urlencoded

请求参数：

字段名	变量名	必填	类型	示例值	描述
用户名	username	是	String	test	用户账户 长度3-30 位

用户密码	password	是	String	123456	密码长度 6-16 位
------	----------	---	--------	--------	-------------

返回参数:

字段名	变量名	类型	示例值	描述
返回错误码	code	Int	0	0 为成功, 其它值为失败 3003 参数不合法 2000 用户已被注册
返回消息	msg	String	success!	0 时返回 login success
用户名	username	String	test	返回当前登录的用户名
Token	token	String	e8329c6f25ac892 6773b0eb7313ce2 5ffa6a4a73	返回 40 位 token

请求示例

```
POST http://localhost:8000/api/v4/login

Content-type: application/x-www-form-urlencoded

username=test&password=123456
```

返回示例

```
{
```

```
{
  "code": 0,

  "msg": "login success!",

  "username": "test",

  "token": "a52a9d985723ba9af67619f4eb0d3182c2856f52"
}
```

## 商品信息接口

### 查询商品/`api/v1/goods/{id}`

接口名称: 查询商品信息

接口地址: `/api/v1/goods/{id}`

请求方式: GET

### 功能描述

查询单个接口的商品详情, `{id}` 是商品的 id, 在 url 上传对应的 id

请求参数:

字段名	变量名	必填	类型	示例值	描述
商品 id	id	是	String	1	商品 ID

返回参数:

字段名	变量名	类型	示例值	描述
返回错误	code	Int	0	0 为成功, 其它值为失败

码				1000 为找不到商品信息
返回消息	msg	String	success!	code 为 0 时返回详情
数据	Data	Object	参考返回示例	返回商品详情 json Object 对象 查询不到返回{}

## 请求示例

```
GET http://localhost:8000/api/v1/goods/1
```

## 返回示例：

```
GET http://localhost:8000/api/v1/goods/1
```

```
{
  "code": 0,
  "msg": "success!",
  "data": {
    "id": 1,
    "goodsname": "《pytest 入门到精通》",
    "goodscode": "sp_100002",
    "merchantid": "10001",
    "merchantname": "悠悠学堂",
    "goodsprice": 79.6,
```

```
        "stock": 1000,  
  
        "goodsgroupid": 0,  
  
        "goodsstatus": 1,  
  
        "price": 22.0,  
  
        "create_time": "2021-01-17 15:14:25",  
  
        "update_time": "2021-01-17 15:14:47"  
  
    }  
  
}
```

查询失败示例

```
{  
  
    "code": 1000,  
  
    "msg": "no info",  
  
    "data": {}  
  
}
```

## 新增商品 **/api/v1/goods**

**接口名称：**新增商品信息

**接口地址：**/api/v1/goods

**请求方式：**POST

**请求头部：**Content-Type: application/json

功能描述：

调用此接口新增商品信息

Body 传 json 类型，头部声明参数类型 Content-Type: application/json

请求参数：

字段名	变量名	必填	类型	示例值	描述
商品名称	goodsname	是	String	《selenium 入门到精通》	商品名称
商品代码	goodscode	是	String	sp_100001	商品对应的代码，唯一字段数据库不能重复 8-15 位，必须以 sp 开头
商户 ID	merchantid	否	String	10001	商户 id
商户名称	merchantname	否	String	悠悠学堂	商户名称
商品价格	goodsprice	否	Float	49.9	商品价格，浮点数，精确到分 如 49.90
库存	stock	否	Int	100	库存剩余商品数量
商品分组	goodsgroupid	否	Int	0	商品分组，整数类型 默认 0
成本价	price	否	Float	21	成本价，不展示给用户看 后台展示
商品状态	goodsstatus	否	Int	1	枚举类型： 0 下架，1 出售 中 默认 1

返回参数：

字段名	变量名	类型	示例值	描述
-----	-----	----	-----	----

返回错误码	code	Int	0	0 为成功，其它值为失败  2000 缺少必填项 3003  参数不合法 4000 不能重复添加
返回消息	msg	String	success!	状态码为 0 时返回 success!
数据	Data	Object	参考返回示例	返回商品详情 json Object 对象 查询不到返回{ } 参数不合法返回 {"goodsprice":["请填写合法的数字。"]}

请求示例：

POST http://localhost:8000/api/v1/goods

Content-Type: application/json

```
{  
  
  "goodsname": "《selenium 入门到精通》",  
  
  "goodscode": "sp_100022",  
  
  "merchantid": "10001",  
  
  "merchantname": "悠悠学堂",  
  
  "goodsprice": 49.9,  
  
  "stock": 100,  
  
  "goodsgroupid": 0,
```

```
"goodsstatus": 1,  
  
"price": 21.0  
  
}  
  
}
```

返回示例：

```
{  
  
  "code": 0,  
  
  "msg": "success!",  
  
  "data": {  
  
    "id": 1,  
  
    "goodsname": "《pytest 入门到精通》",  
  
    "goodscode": "sp_100002",  
  
    "merchantid": "10001",  
  
    "merchantname": "悠悠学堂",  
  
    "goodsprice": 79.6,  
  
    "stock": 1000,  
  
    "goodsgroupid": 0,  
  
    "goodsstatus": 1,  
  
    "price": 22.0,  
  
  }  
}
```



```
        "create_time": "2021-01-17 15:14:25",  
        "update_time": "2021-01-17 15:14:47"  
    }  
}
```

参数不合法返回

```
{"code":3003,"msg":"参数不合法","data":{"goodsprice":["请填写合法的数 字。"]}}
```

商品 id 重复返回

```
{"code":4000,"msg":"goodscode 不能重复添加","data":{}}
```

缺少必填项 goodscode

```
返回 {"code":2000,"msg":"缺少必填项 goodscode","data":{}}
```

## 修改商品 /api/v1/goods/{id}

**接口名称：**修改商品信息

**接口地址：**/api/v1/goods/{id}

**请求方式：**PUT

**请求头部：**Content-Type: application/json

## 功能描述：

根据商品 id 修改商品的详情信息，{id} 是商品的 id，在 url 上传对应的 id

Body 传 json 类型

**请求参数:**

字段名	变量名	必填	类型	示例值	描述
商品 id	id	是	String	1	商品 ID

**请求 body:**

字段名	变量名	必填	类型	示例值	描述
商品名称	goodsname	是	String	《selenium 入门到精通》	商品名称
商品代码	goodscode	是	String	sp_100001	商品对应的代码，唯一字段数据库不能重复 8-15 位，必须以 sp 开头
商户 ID	merchantid	否	String	10001	商户 id
商户名称	merchantname	否	String	悠悠学堂	商户名称
商品价格	goodsprice	否	Float	49.9	商品价格，浮点数，精确到分 如 49.90
库存	stock	否	Int	100	库存剩余商品数量
商品分组	goodsgroupid	否	Int	0	商品分组，整数类型 默认 0
成本价	price	否	Float	21	成本价，不展示给用户看 后台展示
商品状态	goodsstatus	否	Int	1	枚举类型： 0 下架，1 出售中 默认 1

**返回参数:**

字段名	变量名	类型	示例值	描述
返回错误码	code	Int	0	0 为成功，其它值为失败 2000 缺少必填项 3003

				参数不合法 4000 不能重复添加
返回消息	msg	String	success!	状态码为 0 时返回 success!
数据	data	Object	参考返回示例	返回商品详情 json Object 对象 查询不到返回{ } 参数不合法返回 {"goodsprice":["请填写合法的数字。"]}

## 请求示例

PUT http://localhost:8000/api/v1/goods/1

Content-Type: application/json

```
{
  "code": 0,
  "msg": "success!",
  "data": {
    "id": 1,
    "goodsname": "《pytest 入门到精通》",
    "goodscode": "sp_100002",
    "merchantid": "10001",
    "merchantname": "悠悠学堂",
    "goodsprice": 79.6,
    "stock": 1000,
```

```
        "goodsgroupid": 0,  
  
        "goodsstatus": 1,  
  
        "price": 22.0,  
  
        "create_time": "2021-01-17 15:14:25",  
  
        "update_time": "2021-01-17 15:14:47"  
  
    }  
  
}
```

返回示例：

```
{  
  
    "code": 0,  
  
    "msg": "success!",  
  
    "data": {  
  
        "id": 1,  
  
        "goodsname": "《pytest 入门到精通》",  
  
        "goodsgroupid": 0,  
  
        "goodscode": "sp_100002",  
  
        "merchantid": "10001",  
  
        "merchantname": "悠悠学堂",  
  
        "goodsprice": 79.6,  
  
        "stock": 1000,  
  
        "goodsgroupid": 0,  
  
    }  
  
}
```

```
        "goodsstatus": 1,

        "price": 22.0,

        "create_time": "2021-01-17 15:14:25",

        "update_time": "2021-01-17 15:14:47"

    }

}
```

## 删除商品 /api/v1/goods/{id}

**接口名称：**查询商品信息

**接口地址：**/api/v1/goods/{id}

**请求方式：**DELETE

### 功能描述：

删除商品信息，不可逆操作，会直接删掉数据库对应的数据，{id} 是商品的 id，在 url 上传 对应的 id

**请求参数：**

字段名	变量名	必填	类型	示例值	描述
商品 id	id	是	String	1	商品 ID

**返回参数：**

字段名	变量名	类型	示例值	描述
-----	-----	----	-----	----

返回错误码	code	Int	0	0 为成功，其它值为失败 1000 为找不到商品信息
返回消息	msg	String	success!	code 为 0 时返回详情
数据	Data	Object	参考返回示例	返回商品详情 json Object 对象 查询不到返回{}

## 请求示例

```
DELETE http://localhost:8000/api/v1/goods/1
```

## 返回示例

删除成功返回示例

```
{
  "code":0,
  "msg":"删除成功",
  "data":{}
}
```

查询无商品 id 返回示例

```
{
  "code":1000,
```

```
"msg": "无数据删除",  
  
"data": {}  
  
}
```

## 分页查询全部商品 /api/v1/goods?page=1&size=2

**接口名称:** 查询全部商品信息

**接口地址:** /api/v1/goods

**请求方式:** GET

### 功能描述:

查询全部商品信息，按修改时间倒叙不显示商品状态为 0（下架的商品），不显示 price 成本价字段

支持分页查询 /api/v1/goods?page=1&size=2 每页默认显示 5 条，最大 50 条

**请求参数:**

字段名	变量名	必填	类型	示例值	描述
页数	page	否	Int	1	查询第几页的数据
每页显示数量	size	否	Int	2	每页默认显示 5 条，最大 50 条 可以自己指定条数

## 请求示例

```
GET http://localhost:8000/api/v1/goods?page=1&size=2
```

## 返回示例：

```
{
  "code": 0,
  "msg": "success!",
  "data": [ {
    "id": 128,
    "goodsname": "",
    "goodscode": "sp_10019",
    "merchantid": "",
    "merchantname": "",
    "goodsprice": 0.0,
    "stock": 0,
    "goodsgroupid": 0,
    "goodsstatus": 1,
    "create_time": "2021-01-17 15:45:26.79",
    "update_time": "2021-01-17 15:45:26.79"
  },
```



```
{ "id": 127,  
  
  "goodsname": "",  
  
  "goodscode": "sp_10018",  
  
  "merchantid": "",  
  
  "merchantname": "",  
  
  "goodsprice": 0.0,  
  
  "stock": 0,  
  
  "goodsgroupid": 0,  
  
  "goodsstatus": 1,  
  
  "create_time": "2021-01-17 15:42:53.85",  
  
  "update_time": "2021-01-17 15:42:53.85"  
  
}]  
  
}
```

## 需登录才能访问的 v2 接口

v2 版本在前面 v1 版本基础上加了访问权限，需用户拿到 token 才能访问 获取 token

参考登录接口返回 token

查询单个商品 **/api/v2/goods/{id}**

请求示例

GET http://localhost:8000/api/v2/goods/1

Authorization: Token 313737f83c42e546e491b4c3888f81f1a4a88a73

## 新增商品 **/api/v2/goods**

请求示例

POST http://localhost:8000/api/v2/goods

Content-Type: application/json

Authorization: Token 313737f83c42e546e491b4c3888f81f1a4a88a73

```
{  
  
  "goodsname": "《selenium 入门到精通》",  
  
  "goodscode": "sp_100029",  
  
  "merchantid": "10001",  
  
  "merchantname": "悠悠学堂",  
  
  "goodsprice": 49.9,  
  
  "stock": 100,  
  
  "goodsgroupid": 0,  
  
  "goodsstatus": 1,  
  
  "price": 21.0
```

```
}
```

## 修改商品 `/api/v2/goods/{id}`

请求示例

```
PUT http://localhost:8000/api/v2/goods/1
```

```
Content-Type: application/json
```

```
Authorization: Token 313737f83c42e546e491b4c3888f81f1a4a88a73
```

```
{  
  "goodsname": "《selenium 入门到精通》",  
  "goodscode": "sp_100002",  
  "merchantid": "10001",  
  "merchantname": "悠悠学堂",  
  "goodsprice": 49.9,  
  "stock": 100,  
  "goodsgroupid": 0,  
  "goodsstatus": 1,  
  "price": 21.0  
}
```

## 删除商品 **/api/v2/goods/{id}**

请求示例

```
DELETE http://localhost:8000/api/v2/goods/1
```

```
Authorization: Token 313737f83c42e546e491b4c3888f81f1a4a88a73
```

## 分页查询 **/api/v2/goods?page=1&size=2**

请求示例

```
GET http://localhost:8000/api/v2/goods?page=1&size=2
```

```
Authorization: Token 313737f83c42e546e491b4c3888f81f1a4a88a73
```

# 我的购物车接口

## 查询我的购物车 **/api/v2/carts**

接口名称: 查询我的购物车

接口地址: /api/v2/carts

请求方式: GET

### 功能描述:

用户账户登录, 带着 token 查询我的购物车, 购物车商品数量(goodsnum)为 0 的不显示

请求头部带 token

Authorization: Token bf4c2a6d5895d302ff8b90a8b0f06df520caf6bf

请求参数： 无

返回参数：

字段名	变量名	类型	示例值	描述
返回错误码	code	Int	0	0 为成功，其它值为失败
返回消息	msg	String	success!	code 为 0 时返回 success!
数据	Data	Object	参考返回示例	返回商品详情 list of json Object 对象 查询不到返回[]
商品code	goodscode	String	sp_100002	每个商品对应的 code,
商品数量	goodsnum	Int	1	购物车商品数量为 0 的不显示，最小值 0，最大值 100000

请求示例

GET http://localhost:8000/api/v2/carts

Authorization: Token bf4c2a6d5895d302ff8b90a8b0f06df520caf6bf

返回示例

{

```
{
  "code": 0,
  "msg": "success!",
  "data": [{
    "id": 1,
    "create_time": "2021-01-23 10:37:51",
    "update_time": "2021-01-23 10:37:51",
    "goodscode": "cp_100002",
    "goodsnum": 1
  },
  { "id": 2,
    "create_time": "2021-01-23 10:54:45",
    "update_time": "2021-01-23 12:49:58",
    "goodscode": "sp_100008",
    "goodsnum": 2
  }
]
}
```

## 添加购物车 **/api/v2/carts**

接口名称: 添加商品到购物车

接口地址: /api/v2/carts

请求方式: POST

功能描述:

用户账户登录，带着 token 请求，添加商品到我的购物车

- 1.商品不存在，不添加到购物车
- 2.如果用户购物车无此商品，那么直接添加到购物车
- 3.如果用户购物车已存在此商品，那么更新商品的数量(goodsnum)
- 4.如果用户需把该商品从购物车删除，那么修改商品的数量(goodsnum)为 0

请求头部带 token

Authorization: Token bf4c2a6d5895d302ff8b90a8b0f06df520caf6bf

请求参数:

字段名	变量名	必填	类型	示例值	描述
商品code	goodscode	是	String	sp_100008	每个商品对应的 code
商品数量	goodsnum	是	Int	1	购物车商品数量 最小值 0, 最大值 100000

返回参数:

字段名	变量名	类型	示例值	描述
返回错误	code	Int	0	0 为成功，其它值为失败 3003 参数不合法 5000 商品 code 不存在

码				
返回消息	msg	String	success!	code 为 0 时返回 success!
数据	Data	Object	参考返回示例	返回商品详情 json Object 对象 查询开到返回{}

## 请求示例

POST http://localhost:8000/api/v2/carts

Authorization: Token bf4c2a6d5895d302ff8b90a8b0f06df520caf6bf

Content-Type: application/json

```
{
  "goodscode": "sp_100009",
  "goodsnum": 1
}
```

## 返回示例

```
{
  "code": 0,
  "msg": "success!",
  "data": {
    "id": 3,
```



```
        "create_time": "2021-01-23 13:25:50",  
        "update_time": "2021-01-23 13:31:33",  
        "goodscode": "sp_100009", "goodsnum": 1  
    }  
}
```

## 订单管理

### 查询我的全部订单 **/api/v2/orders**

接口名称: 我的全部订单

接口地址: /api/v2/orders

请求方式: GET

### 功能描述

根据当前登陆用户，查询用户个人的订单，订单状态有四种：

0-待付款的订单

1-已付款的订单

2-已取消的订单

3-订单超时，关闭状态

查询全部订单（不显示已取消的订单）

请求头部带 token

Authorization: Token bf4c2a6d5895d302ff8b90a8b0f06df520caf6bf

请求参数： 无

返回参数：

字段名	变量名	类型	示例值	描述
返回错误码	code	Int	0	0 为成功，其它值
返回消息	msg	String	success!	code 为 0 时返回 success!
数据	Data	Object	参考返回示例	返回商品详情 list of json Object 对象 查询开到返回[]

## 请求示例

GET http://localhost:8000/api/v2/orders

Authorization: Token bf4c2a6d5895d302ff8b90a8b0f06df520caf6bf

## 返回示例

```
{
  "code": 0,
  "msg": "success!",
  "data": [{
```

```
"id": 11,

"create_time": "2021-01-28 16:00:59",

"update_time": "2021-01-28 16:00:59",

"pay_status": 1,

"order_id": "yy20210128160059168",

"pay_type": "alipay",

"pay_time": null,

"payment": "149.70",

"user_id": "test1",

"buyer_msg": ""

},

{ "id": 13,

"create_time": "2021-01-28 16:18:20",

"update_time": "2021-01-28 16:18:20",

"pay_status": 0,

"order_id": "yy20210128161820119",

"pay_type": "alipay",

"pay_time": null,

"payment": "149.70",

"user_id": "test1",

"buyer_msg": "用户留言：亲，送一包辣条哦！ "

}
```

```
    ]  
}  

```

## 创建订单 `/api/v2/orders`

接口名称: 创建订单

接口地址: `/api/v2/orders`

请求方式: POST

### 功能描述

根据当前登陆用户，用户需把购买的商品先添加到购物车（数量大于等于 1）

- 1.如果购物车无商品，不生成订单，
- 2.如果购物车有多个商品，一次性把购物车商品生成一个订单 （不支持购物车部分商品生成订单）
- 3.生成订单后，购物车清空
- 4.下次生成订单，需重新加到购物车 请求头部带 token

Authorization: Token bf4c2a6d5895d302ff8b90a8b0f06df520caf6bf

请求参数:

字段名	变量名	必填	类型	示例值	描述
用户留言	buyer_msg	否	String	用户留言： 亲，送一包	生成订单时, 用户给商家的留言 最大 100 个字符

				辣条哦!	
--	--	--	--	------	--

#### 返回参数:

字段名	变量名	类型	示例值	描述
返回错误码	code	Int	0	0 为成功, 其它值为失败  5003 购物车无商品
返回消息	msg	String	success!	状态码为 0 时返回 success! 5003 请先添加商品到购物车!
数据	Data	Object	参考返回示例	返回商品详情 json Object 对象

#### 订单详情参数说明

字段名	变量名	类型	示例值	描述
支付状态	pay_status	Int	0	0-待付款的订单  1-已付款的订单  2-已取消的订单  3-订单超时 (24 小时超时, 系统更改状态为关闭), 关闭状态
订单号	order_id	String	yy202101281  71126112	系统自动生成
支付类型	pay_type	String	alipay	默认显示 alipay, 支付的时候可选择支付类型 alipay---支付宝 wechat---微信"
支付时间	pay_time	DateTime		用户支付后生成
订单金额	payment	Float	49.90	订单生成的金额, 根据购物车商品和数量 计算得出 (暂时不支持优惠

				券，抵扣券)
买家留言	buyer_msg	String	记得包邮哦!	买家留言信息
用户名	user_id	String	test	用户名

## 请求示例

POST http://localhost:8000/api/v2/orders

Authorization: Token bf4c2a6d5895d302ff8b90a8b0f06df520caf6bf

Content-Type: application/json

```
{"buyer_msg": "用户留言：亲，送一包辣条哦！"}
```

## 返回示例

```
{
  "code": 0,
  "msg": "success!",
  "data": {
    "id": 14,
    "create_time": "2021-01-28 17:11:26",
    "update_time": "2021-01-28 17:11:26",
    "pay_status": 0,
    "order_id": "yy20210128171126112",
```

```
        "pay_type": "alipay",

        "pay_time": null,

        "payment": "49.90",

        "user_id": "test1",

        "buyer_msg": "用户留言：亲，送一包辣条哦！ "

    }

}
```

## 删除订单 **/api/v2/orders**

接口名称：创建订单

接口地址: /api/v2/orders

请求方式: DELETE

## 功能描述

根据当前登陆用户，删除自己的订单，订单状态有 4 种

0-待付款的订单

1-已付款的订单

2-已取消的订单

3-订单超时（24 小时超时，系统更改状态为关闭），关闭状态 只有 0（待付款） 和 3(订单超时) 订单可以删除，删掉订单后，订单状态变成 2（已取消）

已付款的订单不能被用户删除

请求头部带 token

Authorization: Token bf4c2a6d5895d302ff8b90a8b0f06df520caf6bf

请求参数:

字段名	变量名	必填	类型	示例值	描述
订单号	order_id	是	String	yy20210128160059168	根据订单号删除 只有 0 (待付款) 和 3(订单超时) 订单可以删除, 删掉订单后, 订单状态变成 2 (已取消) 已付款的订单不能被用户删除

返回参数:

字段名	变量名	类型	示例值	描述
返回错误码	code	Int	0	0 为成功, 其它值为失败 1000 用户已经删除过, 无数据删除 1005 已支付的订单不能被删除 1006 其它未知异常 5003 order_id 不存在
返回消息	msg	String	success!	code 为 0 时返回 success!
数据	Data	Object	参考返回示例	返回商品详情 json Object 对象

请求示例

DELETE http://localhost:8000/api/v2/orders?order\_id=yy20210128160059168



Authorization: Token bf4c2a6d5895d302ff8b90a8b0f06df520caf6bf

## 返回示例

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

## 收获地址

### 查询收获地址 **/api/v2/address**

接口名称: 查询我的收获地址

接口地址: /api/v2/address

请求方式: GET

### 功能描述:

用户账户登录, 带着 token 查询我的收获地址

请求头部带 token

Authorization: Token bf4c2a6d5895d302ff8b90a8b0f06df520caf6bf

请求参数： 无

返回参数：

字段名	变量名	类型	示例值	描述
错误码	code	Int	0	0 为成功，其它值为失败
返回消息	msg	String	success!	code 为 0 时返回 success!
收获地址	data	Object	参考返回示例	返回详情 json Object 对象 查询不到返回{ }

## 请求示例

```
GET http://localhost:8000/api/v2/address
```

```
Authorization: Token bf4c2a6d5895d302ff8b90a8b0f06df520caf6bf
```

## 返回示例

```
{
  "code": 0,
  "msg": "success!",
  "data": {
    "id": 3,
    "create_time": "2021-01-23 17:32:21",
    "update_time": "2021-01-23 17:36:50",
    "userid": "test",
```

```
        "tel": "15001234002",

        "name": "张三",

        "address": null, "postal": null

    }

}
```

## 添加和修改收获地址 `/api/v2/address`

接口名称: 添加和修改收获地址

接口地址: `/api/v2/address`

请求方式: POST

### 功能描述:

用户账户登录, 带着 token 请求, 添加和修改收获地址

每个用户只能添加一个收获地址

1. 用户首次可以添加收获地址
2. 用户可以修改收获地址相关信息

请求头部带 token

Authorization: Token bf4c2a6d5895d302ff8b90a8b0f06df520caf6bf

请求参数:

字段名	变量名	必填	类型	示例值	描述
-----	-----	----	----	-----	----

手机号	tel	否	String	15001234001	收件人手机号
收件人	name	否	String	张三	收件人姓名
收件人 地址	address	否	String	上海市徐汇区 xx 路 1001 号 101	收件人地址
邮编	postal	否	String	200000	邮编

**返回参数：**

字段名	变量名	类型	示例值	描述
错误码	code	Int	0	0 为成功，其它值为失败  3003 参数不合法  5000 商品 code 不存在
返回消息	msg	String	success!	code 为 0 时返回 success!
购物车 数据	data	Object	参考返回示 例	返回详情 json Object 对象 查询不到返回{ }

**请求示例**

POST http://localhost:8000/api/v2/address

Authorization: Token bf4c2a6d5895d302ff8b90a8b0f06df520caf6bf

Content-Type: application/json

```
{  
  
  "tel": "15001234002"  
  
}
```

## 返回示例

```
{  
  
  "code": 0,  
  
  "msg": "success!",  
  
  "data": {  
  
    "id": 3,  
  
    "create_time": "2021-01-23 17:32:21",  
  
    "update_time": "2021-01-23 17:46:20",  
  
    "userid": "test",  
  
    "tel": "15001234002",  
  
    "name": null,  
  
    "address": null, "postal": null  
  
  }  
  
}
```

# 个人资料

## 查询个人资料 /api/v1/userinfo

接口名称: 查询我的个人资料

接口地址: /api/v1/userinfo

请求方式: GET

### 功能描述:

查询用户的个人资料

请求头部带 token

Authorization: Token bf4c2a6d5895d302ff8b90a8b0f06df520caf6bf

请求参数: 无

### 返回参数:

字段名	变量名	类型	示例值	描述
错误码	code	Int	0	0 为成功，其它值为失败
返回消息	msg	String	success!	code 为 0 时返回 success!
收获地址	data	Object	参考返回示例	返回详情 json Object 对象 查询不到返回{ }

## 请求示例

```
GET http://49.235.92.12:7005/api/v1/userinfo HTTP/1.1
```

```
Authorization: Token 187ed7fc443e35276b4c8ebd3f5873f590169480
```

## 返回示例

```
{
  "msg": "success!",
  "code": 0,
  "data": [
    {
      "id": 1,
      "name": "test",
      "sex": "M", "age": 20,
      "mail": "283340479@qq.com",
      "create_time": "2021-07-22"
    }
  ]
}
```

# 修改/添加个人资料 /api/v1/userinfo

接口名称: 修改/添加我的个人资料

接口地址: /api/v1/userinfo

请求方式: POST

## 功能描述:

添加/修改用户的个人资料

请求头部带 token

Authorization: Token bf4c2a6d5895d302ff8b90a8b0f06df520caf6bf

## 请求参数:

字段名	变量名	必填	类型	示例值	描述
用户名	name	是	String	test	必须和登录的用户名保持一致 不能修改别的用户名资料
性别	sex	否	String	M	枚举类型: M-男 F-女
年龄	age	否	Int	20	年龄, 整数类型
邮编	Mail	否	String	123@qq.com	邮箱格式校验

## 返回参数:

字段名	变量名	类型	示例值	描述
-----	-----	----	-----	----



错误码	code	Int	0	0 为成功，其它值为失败
返回消息	msg	String	success!	code 为 0 时返回 success!
详情	data	Object	参考返回示例	返回详情 json Object 对象 查询不到返回{ }

## 请求示例

POST http://49.235.92.12:7005/api/v1/userinfo HTTP/1.1

Authorization: Token 187ed7fc443e35276b4c8ebd3f5873f590169480

```
{
  "name": "test",
  "sex": "M",
  "age": 20,
  "mail": "283340479@qq.com"
}
```

## 返回示例

```
{
  "message": "update some data!",
  "code": 0,
  "data": {
    "name": "test",
```

```
        "sex": "M",

        "age": 20,

        "mail": "283340479@qq.com"

    }

}
```

## 文件上传

上传文件 **/api/v1/upfile/**

**接口名称：**上传文件

接口地址: /api/v1/upfile/

请求方式: POST

### 功能描述：

上传文件功能，不用登录

**请求参数：**

字段名	变量名	必填	类型	示例值	描述
文件	file	是	file		传的文件
标题	title	是	String	上海-悠悠	标题描述

### 返回参数:

字段名	变量名	类型	示例值	描述
错误码	code	Int	0	0 为成功, 其它值为失败
返回消息	msg	String	success!	code 为 0 时返回 success!
详情	data	Object	参考返回示例	返回详情 json Object 对象 查询不到返回{ }

### 请求示例

The screenshot shows a REST client interface for a file upload endpoint. The method is POST and the URL is `{{base_url}}/api/v1/upfile/`. The 'Body' tab is selected, and 'form-data' is chosen as the content type. Two parameters are defined: a file named 'file' with the value 'part\_0.png' and a text parameter 'title' with the value '上海-悠悠'.

KEY	VALUE	DESCRIPTION
file	part_0.png	
title	上海-悠悠	

POST /api/v1/upfile/ HTTP/1.1

Content-Type: multipart/form-data;

boundary=----WebKitFormBoundary7MA4YWxkTrZu0gW

Content-Disposition: form-data; name="file";

filename="/C:/Users/dell/Desktop/image/part\_0.png

```
-----WebKitFormBoundary7MA4YWxkTrZu0gW--,
Content-Disposition: form-data; name="file";
filename="/C:/Users/dell/Desktop/image/part_0.png
```

```
-----WebKitFormBoundary7MA4YWxkTrZu0gW--
Content-Disposition: form-data; name="title"
```

上海-悠悠

```
-----WebKitFormBoundary7MA4YWxkTrZu0gW-- }
```

## 返回示例

```
{
  "message":"update some data!",
  "code":0,
  "data":{
    "name":"test",
    "sex":"M",
    "age":20,
    "mail":283340479@qq.com
```

}

}