

用户端小程序接口列表

接口统一前缀地址: <https://domain/api/user>

请求统一参数:

token (wxLogin接口除外)

统一返回结果:

```
{
  code: 状态码 (0表示成功, 其他为失败),
  message: 返回信息(成功时一般为请求成功, 失败时为失败信息),
  data: 返回数据, 始终是一个对象(成功时存在, 失败时一般不存在, 除非特殊情况需要数据时才存在)
}
```

全局状态码:

- 0 成功
- 500 未知错误
- 10000 接口不存在
- 10001 参数不合法
- 10003 未登录
- 10004 账号不存在
- 10005 密码错误
- 10006 账号已存在
- 10007 token无效或不存在
- 70010 微信openId不存在(此时需重新调用 `wx.login` 并调用/wxLogin接口以获取微信openId)
- 70011 微信appid不存在

用户端登陆机制 (即 token , openId, 用户账号之间的关系):

步骤:

1. token获取: 调用 `wx.login`, 获取code, 然后请求 `/wxLogin` 接口 (通过user agent 获取appid), 获取token, 并缓存在本地

2. 绑定openId 到账户: 通过 `/login` 接口, 将用户手机号(即用户实际账号)与openId相关联

流程:

首次进入小程序后, 首先 获取token, 然后 登陆时绑定openId 到账户

每次进入小程序, 首先获取本地token缓存, 若不存在则 获取token , 获取后若不存在用户信息,则需要登录(即步骤2)

后续操作:

若收到后台返回的状态码为 10007 和 70010 , 则需要执行步骤1

若收到后台返回的状态码为 10003 , 则需要执行步骤2

登陆相关

☐ 微信登陆接口

接口地址: POST /wxLogin

参数:

code wx.login 获取到的code

返回值:

```
data: {
  token: 用户令牌,
  userInfo: 用户信息, 微信openId已绑定用户的情况下返回, 测试时code传值为user时返回用户信息
}
```

☐ 登陆接口(绑定手机号)

接口地址: POST /login

参数:

mobile 手机号
verify_code 验证码

返回值

```
data: {
  userInfo: 用户信息
}
```

☐ 短信接口

接口地址: POST /sms/verify_code

参数:

mobile 手机号

返回值:

```
data: {
  verify_code: 验证码, 测试时存在
}
```

☒ 获取地区列表(树形结构)

接口地址: GET /area/tree

参数: tier 要获取的层级数 1-3 (省/市/县), 默认为3

返回:

```
data: {
  list: [
    {
      id: id,
      area_id: 地区ID,
      name: 地区名,
      parent_id: 父级地区ID
      sub: [
        子地区列表, 有下级时存在, 结构同父级结构
      ]
    }
  ]
}
```

- ☐ 根据经纬度获取所在城市

//

商家模块

- ☐ 获取商家类别列表

接口地址: GET `/merchant/categories/tree`

返回:

```
data: {
  list: [
    {
      id: id,
      pid: 父id,
      name: 分类名,
      icon: 图标 url,
      status: 状态 (只返回状态正常的分类),
      sub: [
        子分类列表
      ]
    }
  ]
}
```

- ☐ 获取商家列表 (关键字搜索, 附近商家等)

接口地址: GET `/merchants`

参数:

city_id: 城市ID
keyword: 搜索关键字
merchant_category_id: 商家类别ID,
lng: 用户当前经度
lat: 用户当前纬度
radius: 范围, 范围参数只有在经纬度信息存在时才有效, 当传递范围参数时, 会获取用户位置所指定范围内的商家并根据距离排序
page: 获取的页数

返回

```
data: {
  total: 总记录数,
  list: [
    {
      id: 商家id,
      oper_id: 运营中心ID
      merchant_category_id: 商家分类ID,
      name: 商家名,
      status: 状态(只返回状态正常的商家),
      lng: 商家所在位置经度,
      lat: 商家所在位置纬度,
      address: 详细地址,
      province: 所在省份,
      province_id: 省份ID,
      city: 城市,
      city_id: 城市ID,
      area: 县区,
      area_id: 县区ID,
      distance: 距离, 当传递经纬度信息时才存在
    }
  ]
}
```

☐ 商品列表

接口地址: GET `/goods`

参数: `merchant_id` 商家ID

返回

```
data: {
  list: [
    {
      id: 商品ID,
      oper_id: 运营中心ID
      merchant_id: 商家ID,
      name: 商品名,

      desc: 商品描述,
```

```
    price: 商品价格,  
    pic: 商品默认图,  
    pic_list: 商品小图列表, 数组  
    status: 状态 1-上架 2-下架  
  }  
]  
}
```

☐ 商品详情

接口地址: GET `/goods/detail`

参数: id 商品ID

返回

同商品列表中的每一项

订单模块

☐ 订单列表

接口地址: GET `/orders`

参数:

status: 状态

返回

```
data: {  
  total: 总订单数,  
  list: [  
    {  
      id: 订单ID,  
      oper_id: 运营中心ID,  
      order_no: 订单号,  
      user_id: 用户ID,  
      user_name: 用户名,  
      merchant_id: 商家ID,  
      merchant_name: 商家名,  
      goods_id: 商品ID,  
      goods_name: 商品名,  
      goods_pic: 商品图片,  
      price: 商家价格,  
      status: 状态 1-未支付 2-已取消 3-已关闭 (超时自动关闭) 4-已付款 6-已退款 7-已完成  
(不可退款),  
    }  
  ]  
}
```

```
]
}
```

☐ 订单详情

地址: GET `/order/detail` 参数

order_no 订单号

返回

同订单列表中的每一项

☐ 下单接口

地址: POST `/order/add`

参数

goods_id: 商品ID
number: 数量
notify_mobile: 通知手机号(可为空)

返回

```
data: {
  orderInfo: {
    同订单列表中的每一项
  },
  wxPayParams: {
    微信下单参数 -- 暂时还没有
  }
}
```

☐ 退款接口

地址: POST `/order/refund`

参数

order_no 订单号

返回

同订单列表中的每一项