掌中云渠道开放接口 v1.0

1. 概述

1.1 鉴权机制

请先申请 API Key 和 Secret，如果您有 VIP 帐号，建议使用 VIP 帐号申请，这样便可以使用同一个 Key 访问子渠道的数据。

发起 API 请求时，需要先计算签到 sign，再将 sign 放到 URL 中发起 API 调用（参数名`sign`）。 签名计算方法如下：

- 将 API Key 加入到请求参数中，参数名为`key`
- 将 URL 参数按参数名字典序排序，再拼接成 URL 键值对形式的字符串A
- 将 API Secret 拼接到字符串A的前面得到字符串B
- 字符串B的 md5 值即签名 sign
- 将 URL 参数中的非空参数按字典序排序，再使用 URL 键值对的形式 (即 key1=value1&key2=value2...) 拼装成字符串A
- 对字符串A做 md5 即可得到签名 sign

示例:

API Key: your_key

API Secret: your_secret

请求参数: channel_id=1024&status=1

sign = md5('your_secretchannel_id=1024&key=your_key&status=1')

最终请求参数: channel_id=1024&status=1&key=your_key&sign=c7490364d7059f63c1ad0173e2e3a841

1.2 响应格式

所有接口都将通过 JSON 返回，并通过 HTTP 状态码来表示成功或者失败。

HTTP 状态码

- 请求成功: 2XX 系列
- 请求失败: 4XX 或 5XX 系列

响应体

```
{
    "message": "参数错误",    // 错误信息，仅失败时返回
    "data": {                // 业务数据
```

```
        ...
    }
}
```

日期字段都按 ISO8601 的格式返回，例如: 2020-02-18T12:59:00+08:00。

2. 接口列表

接口地址: `https://openapi.818tu.com`

调用频率: 1000 次 / 天

简洁起见，下面的参数列表将省略公共参数 key 和 sign。

2.1 获取 Access Token

`GET /partners/channel/mp/access_token`

获取微信公众号 Access Token，如果您的 API Key 是使用 VIP 帐号申请的，则需要带上`channel_id`参数。

Query 参数:

参数	是否必填	说明
channel_id	VIP 帐号必填	要获取 Access Token 的渠道 ID

返回结果示例

```
{
  "data": {
    "token":
"30_a9qfR9sEI18Zbi0NX8C7XrwXQtGz5GCoVPgKB7nxQksmG1P3ztJZyBzvmV9URqEJeWfIeD
sjXhGahm7n8ilhICdlut55FAhqtqQRBryNYUlXqz_2s45soAPUy09pRRnRGZk4LGcMD73w2Xd2
ZTKcAFAHBS",
    "expires_in": 1800
  }
}
```

2.2 获取子渠道列表

`GET /partners/channel/channels/list`

Query 参数:

参数	是否必填	说明
page	N	页码，默认第1页
per_page	N	每页显示数量，默认 100 条

返回结果示例

```
{
  "data": {
    "count": 12,
    "items": [
      {
        "id": 1001,
        "username": "lilei",
        "nickname": "李雷小说",
        "created_at": "2020-02-18T12:59:00+08:00"
      }
    ]
  }
}
```


2.3 获取订单列表

`GET /partners/channel/orders/list`

Query 参数:

参数	是否必填	说明
page	N	页码，默认第1页
per_page	N	每页显示数量，默认 100 条
channel_id	VIP 帐号必填	订单所属的渠道 ID
status	N	订单状态，多个状态可用英文逗号分隔，不传则获取所有订单。0: 未支付，1: 已支付，3: 已关闭，4: 退款中，5: 已退款
order_by	N	排序。例如: `created_at desc`表示按订单时间倒序，`created_at asc`表示按订单时间正序
created_at[op]	N	订单创建时间过滤条件，op 可以是 lt, lte, gt, gte, eq 中的一种

请求示例:

获取 2020-02-18 的订单

`GET /partners/channel/orders/list?created_at[gte]=2020-02-18T00:00:00+08:00&created_at[lte]=2020-02-18T23:59:59+08:00`

返回结果示例:


```
{
  "data": {
    "count": 120,
    "items": [
```

```
      {
        "id": 1001, // 订单数字ID
        "border_id": "202002081404051004750l", // 订单编号
        "member": {
          "id": 5001, // 用户ID
          "openid": "oWtA4t2zpVF73orpTLklQtlXXXX", // 用户 OpenID
          "created_at": "2020-02-08T14:03:05+08:00", // 用户创建时间
          "subscribed_at": "2020-02-08T14:03:05+08:00" // 用户关注时间，重复关
注时会更新
        },
        "price": 5000, // 订单实际支付价格，单位为分
        "status": 1, // 订单状态
        "created_at": "2020-02-08T14:04:05+08:00",  // 订单创建时间。ISO8601
格式
        "paid_at": "2020-02-08T14:06:09+08:00",     // 订单到帐时间。ISO8601
格式
        "from_novel_id": 907,   // 来源小说ID，直接充值时为 null
        "referral_link_id": 11021, // 来源推广链接ID，无来源链接时为 null
      }
    ]
  }
}
```

如果是通过，比如推广链接派单引流的，`created_at`即点击链接进入书城的时间。 而如果是引流直接关注公众号（即关注前没有先进入书城），则`created_at`基本等于关注时间。

此接口在调用时建议始终传入时间范围 (`created_at`参数)，且截止时间至少已经过去几分钟， 否则可能多次调用同一页码返回的数据会不一样（因为在调用的过程中可能产生新订单，且系统可能会有延迟）。 时间范围建议不要超过1天。