api_cn

亚交所API中文文档

前言

本文为亚交所提供的程序化交易接口API的说明文档，我们希望通过该文档的说明，开发者可用自行实现对于平台API申请、校验、调用、调试等功能，最终在链星科技的平台上实现程序化交易。

Platform Url

- platform Url:https://www.cdcbk.com

baseUrl

- baseUrl:https://www.cdcbk.com/unique/
- 其他平台需要先进行邮件申请，并开通接口，之后才会开发程序化接口API

查询

开发者可用通过查询接口获取到币种交易对、实时行情、交易深度、交易历史等信息。目前，查询功能不需要登录认证并传递token，如果以后查询的请求压力增大可用考虑增加token，并限制每日查询次数。

获取币种交易对

- request_url：baseUrl + coin/allCurrencyRelations
- method：GET
- parameter：null
- response_data:

item	description	type
id	编号	int
baseCurrencyId	基础币id	int
baseCurrencyName	基础币名称	string
baseCurrencyNameEn	基础币英文名称	string
tradeCurrencyId	交易币id	int
tradeCurrencyName	交易币名称	string
tradeCurrencyNameEn	交易币英文名称	string
amountLowLimit	最低买入数量限制	decimal
amountHighLimit	最高买入数量限制	decimal
sellAmountLowLimit	最低卖出数量限制	decimal
sellAmountHighLimit	最高卖出数量限制	decimal

- response description：当接口返回的status 为200时，则attachment包含以下数据，如果status参数不为200 ，则出现异常。
- example：


```
{
    "attachment": [
    {
    "id": 1,
    "baseCurrencyId": "1",
        "tradeCurrencyId": "2",
        "baseCurrencyName": "Bitcoin",
        "baseCurrencyNameEn": "BTC",
        "tradeCurrencyName": "Litecoin",
        "tradeCurrencyNameEn": "LTC",
        "amountLowLimit": 0.0,
        "amountHighLimit": 9999999.0,
        "sellAmountLowLimit": 0.0,
        "sellAmountHighLimit": 9999999.0
    },
    {
    "id": 2,
    "baseCurrencyId": "1",
        "tradeCurrencyId": "3",
        "baseCurrencyName": "Bitcoin",
        "baseCurrencyNameEn": "BTC",
        "tradeCurrencyName": "Ethereum",
        "tradeCurrencyNameEn": "ETH",
```

```
        "amountLowLimit": 0.01,
        "amountHighLimit": 999999.0,
        "sellAmountLowLimit": 0.01,
        "sellAmountHighLimit": 999999.0
    },
    {
    "id": 5,
    "baseCurrencyId": "1",
        "tradeCurrencyId": "6",
        "baseCurrencyName": "Bitcoin",
        "baseCurrencyNameEn": "BTC",
        "tradeCurrencyName": "Lisk",
        "tradeCurrencyNameEn": "LSK",
        "amountLowLimit": 1.0E-4,
        "amountHighLimit": 999999.0,
        "sellAmountLowLimit": 1.0E-4,
        "sellAmountHighLimit": 999999.0
    }
],
"status": 200,
"message": null
}
```


获取实时行情(第一版，即将废弃)

- request_url：baseUrl + quote/realTime
- method：GET
- parameter：

item	description	type
baseCurrencyId	基础币id	string
tradeCurrencyId	交易币id	string

- response_data:

item	description	type
buy	买一价	int
high	最高价	decimal
last	最新成交价	decimal
low	最低价	decimal
sell	卖一价	decimal
vol	交易币成交量	decimal
currencyId	交易币id	int
baseCurrencyId	基础币id	int

- response description：当接口返回的status 为200时，则attachment包含以下数据，如果status参数不为200 ，则出现异常。
- example：


```
{
    "attachment": {
    "buy": 0.0174565,
    "high": 0.01805645,
    "last": 0.0174565,
    "low": 0.01646356,
    "sell": 0.0174565,
    "vol": 391.896,
    "currencyId": 2,
    "baseCurrencyId": 1
    },
    "message": "",
    "status": 200
}
```


获取实时行情(第二版)

- request_url：baseUrl + quote/v2/realTime
- method：GET
- parameter：

Item	description	type
coins	(基础币英文+ '_' + 交易币种英文)，大小写不限，如code_chex	string

- response_data:

Item	description	type
buy	买一价	decimal
high	最高价	decimal
last	最新成交价	decimal
low	最低价	decimal
sell	卖一价	decimal
vol	交易币成交量	decimal
currencyId	交易币id	int
baseCurrencyId	基础币id	int
changeRate	涨跌幅(涨为正数，跌为负数)	decimal
changeAmount	涨跌额(涨为正数，跌为负数)	decimal

- response description：当接口返回的status 为200时，则attachment包含以下数据，如果status参数不为200 ，则出现异常。
- example：

```
{
    "attachment": {
    "buy": 0.2721,
    "high": 0.2815,
    "last": 0.2765,
    "low": 0.2669,
    "sell": 0.2815,
    "vol": 1.2449481758E8,
    "currencyId": 70,
    "baseCurrencyId": 22,
    "changeRate": 0.47,
    "changeAmount": 0.0013,
    },
    "message": null,
    "status": 200
}
```

获取交易深度

- request_url：baseUrl + quote/tradeDeepin
- method：GET
- parameter：

item	description	type
baseCurrencyId	基础币id	string
tradeCurrencyId	交易币id	string
limit	获得的深度的档数	int

- response_data:

item	description	type
asks	卖方委托单数组	Array
bids	买方委托单数组	Array

> 注意：每个数组对象的第一个元素为**价格**，第二个元素为**数量**

- response description：当接口返回的status 为200时，则attachment包含以下数据，如果status参数不为200 ，则出现异常。 返回结果中的asks按价格升序排序，bids按照价格降序排序
- example：

```
{
    "attachment": {
    "asks": [
            [
                "0.01751456",
                "0.0071000000000000"
            ]
        ],
    "bids": [
            [
                "0.01745650",
                "0.0109000000000000"
            ]
        ]
    },
    "message": "",
    "status": 200
}
```

获取所有在线交易对实时行情

- request_url：baseUrl + quote/v2/allRealTime
- method：GET
- parameter：null
- response_data:

Item	description	type
buy	买一价	decimal
high	最高价	decimal
last	最新成交价	decimal
low	最低价	decimal
sell	卖一价	decimal
vol	交易币成交量	decimal
currencyId	交易币名称	int
tradeCurrencyName	交易币id	string
baseCurrencyId	基础币id	int
baseCurrencyName	基础币名称	string
changeRate	涨跌幅(涨为正数，跌为负数)	decimal
changeAmount	涨跌额(涨为正数，跌为负数)	decimal

- response description：当接口返回的status 为200时，则attachment包含以下数据，如果status参数不为200 ，则出现异常。
- example：


```
{
    {
    "attachment": [
      {
        "buy": 0.008025,
        "high": 0.0,
        "last": 0.008025,
        "low": 0.0,
        "sell": null,
        "vol": 0.0,
        "currencyId": 2,
        "baseCurrencyId": 1,
        "tradeCurrencyName": "LTC",
        "baseCurrencyName": "BTC",
        "changeRate": 0.0,
        "changeAmount": 0.0,
        "rmbScale": 0.0
      },
      {
        "buy": 0.0278,
        "high": 0.0,
```

```
        "last": 0.028,
        "low": 0.0,
        "sell": 0.028,
        "vol": 0.0,
        "currencyId": 3,
        "baseCurrencyId": 1,
        "tradeCurrencyName": "ETH",
        "baseCurrencyName": "BTC",
        "changeRate": 0.0,
        "changeAmount": 0.0,
        "rmbScale": 0.0
      }
    ],
    "message": null,
    "status": 200
}
```


获取所有在线交易对实时行情

- request_url：baseUrl + quote/v2/allRealTime
- method：GET
- parameter：null
- response_data:

Item	description	type
buy	买一价	decimal
high	最高价	decimal
last	最新成交价	decimal
low	最低价	decimal
sell	卖一价	decimal
vol	交易币成交量	decimal
currencyId	交易币名称	int
tradeCurrencyName	交易币id	string
baseCurrencyId	基础币id	int
baseCurrencyName	基础币名称	string
changeRate	涨跌幅(涨为正数，跌为负数)	decimal
changeAmount	涨跌额(涨为正数，跌为负数)	decimal

- response description：当接口返回的status 为200时，则attachment包含以下数据，如果status

参数不为200 ，则出现异常。

- example：


```
{
    {
    "attachment": [
      {
        "buy": 0.008025,
        "high": 0.0,
        "last": 0.008025,
        "low": 0.0,
        "sell": null,
        "vol": 0.0,
        "currencyId": 2,
        "baseCurrencyId": 1,
        "tradeCurrencyName": "LTC",
        "baseCurrencyName": "BTC",
        "changeRate": 0.0,
        "changeAmount": 0.0,
        "rmbScale": 0.0
      },
      {
        "buy": 0.0278,
        "high": 0.0,
        "last": 0.028,
        "low": 0.0,
        "sell": 0.028,
        "vol": 0.0,
        "currencyId": 3,
        "baseCurrencyId": 1,
        "tradeCurrencyName": "ETH",
        "baseCurrencyName": "BTC",
        "changeRate": 0.0,
        "changeAmount": 0.0,
        "rmbScale": 0.0
      }
    ],
    "message": null,
    "status": 200
}
```


获取交易历史

- request_url：baseUrl + quote/tradeHistory
- method：GET
- parameter：

item	description	type
baseCurrencyId	基础币id	string
tradeCurrencyId	交易币id	string
limit	获得的深度的档数	int

- response_data:

item	description	type
date	交易时间	string
price	交易价格	decimal
amount	交易金额	decimal
number	交易数量	decimal
coinCode	交易币id	int
baseCurrencyId	基础币id	int
tid	订单号	string
type	交易类型，值为buy，或者sell	string
buyOrSellTrade	交易类型id，1是buy，2是sell	int

- response description：当接口返回的status 为200时，则attachment包含以下数据，如果status参数不为200 ，则出现异常。返回数据 按照时间升序排序
- example：


```
{
    "attachment": [
        {
    "date": "1524561232806",
    "price": 0.0174565,
    "amount": 0.00002618,
    "number": 0.0015,
    "coinCode": 2,
    "baseCurrencyId": 1,
    "tid": "15245612328058682391221000295251",
    "type": "sell",
    "buyOrSellTrade": 2
        }
    ],
    "message": "",
    "status": 200
}
```

权限申请

邮件申请

邮件申请的格式：

item	description	
公司名	公司申请请写企业全称	
电话	手机号码	
公司营业执照号	公司营业执照号	
平台账号	请写对应的平台账号	
public-key	您的公钥将会保存在平台，下文会详细说明公钥的生成和公钥的使用	

> 注意：由于存在一些不可抗因素，目前对于登录、下单、撤单等接口API权限的申请还不能提供独立的页面来实现，因此，目前采用邮件申请的方式来支持API调用权限的申请。

加密签名

1.请您根据**RSA**生成自己的KeyPair（公钥/私钥）

```
# 步骤1  生成rsa私钥  （注：该步骤生成的私钥只为供第二步使用，并无实际用处）
openssl genrsa -out rsa_private_key_1024.pem 1024

# 步骤2  将上一步生成的rsa私钥转换成PKCS#8编码（注：该步骤生成的私钥构成实际密钥对的私钥）
openssl pkcs8 -topk8 -in rsa_private_key_1024.pem -out
pkcs8_rsa_private_key_1024.pem -nocrypt

# 步骤3  导出rsa公钥  （注：该步骤生成的公钥构成实际密钥对的公钥）
openssl rsa -in rsa_private_key_1024.pem -out rsa_public_key_1024.pem -pubout

//私钥文件
pkcs8_rsa_private_key_1024.pem
//公钥文件
rsa_public_key_1024.pem
```

2.然后，将您的public-key（也就是rsa_public_key_1024.pem）、用户名、用户uid发送到**，进行申请。 3.每次申请调用登录、下单、撤单等接口API权限的时候，请您使用sha256算法并通过私钥将参数签名，我在这里给出一个Javascript的实现。 4.timestamp为最新时间

```
//timestamp格式为：2018-05-02T18:30:00Z （注：Javascript使用的是这样的格式 java使用的是13位毫秒级时间戳）

const signature = crypto.createSign('sha256');
signature.write(timestamp.toString());
signature.end();

console.log(signature.sign(APISECRET, 'hex'));

//签名样式如下：
739deeac15a533508d679c6bc7f67d29e92fce7f05359a500abd8fd9844134d97d60e76495850ba392694dd8a50fc6202d8b004f9b9d61422ae698d2e9fc07aeaf29feb02e0019dfa6f30fd2de03be8f2e4cf5c8442d1b56b560427dd1637934750e5c071e8c9928fa81bfb7de83001e0e70b2c774be4bf43fdaa5e932849c9e
```

> 登录时采用用户/密码/签名的方式获取token，在使用下单、撤单、委托单查询等功能时，直接通过token进行。

登录

开发者通过登录接口获取平台token。只有获得了平台token之后，才能操作委托下单和委托撤单等功能。

描述

- request_url：baseUrl + user/signLogin
- method：POST
- Content-Type:application/x-www-form-urlencoded
- Accept:application/json, text/plain,
- parameter：

item	description	type
email	平台登录名（手机号或邮箱）	string
pwd	平台登录密码（经过加盐加密之后的pwd)	string
timestamp	时间戳 (13位毫秒级时间戳)	string
sign	签名 (使用私钥对timestamp进行签名,SHA256withRSA算法)	string

前端密码加密规则

```
const salt = 'dig?F*ckDang5PaSsWOrd&%(12lian0160630).'
let pwd = md5(pwd + salt)
```

- response_data:

返回值在data.attachment中，其中开发者需要的值有：

item	description	type
token	操作请求token	string
uid	用户id	int
uname	用户姓名，目前为空	string
isShow	是否展示高级选项，0是不可以，1是可以	int

- response description：当接口返回的status 为200时，则attachment包含以下数据，如果status参数不为200 ，则出现异常。


```
{
    "attachment": {
        "uid": ****,
        "token": "************************",
        "point": null,
        "uname": null,
        "isShow": 0
    },
    "message": null,
    "status": 200
}
```


signLogin接口对接最为麻烦,在此给出一个java实现的Demo

```
import java.util.HashMap;
import java.util.Map;

public class SignLoginDemo {

  public static void main(String[] args) {
    // 访问url
    String url = "https://www.cdcbk.com/unique/user/signLogin";
    // 登陆账号
    String email = "123456@163.com";
    // 登陆密码  详见MD5Util工具类
    String pwd = MD5Util.encryptPwd("Abc123456");
    // 时间戳  获取13位毫秒级时间戳
    String timestamp = System.currentTimeMillis()+"";
    // 私钥  建议用pkcs8
```

```
    String secretPrivate =
"MIICdwIBADANBgkqhkiG9w0BAQEFAASCAmEwggJdAgEAAoGBAK......";
    // sign签名  详见RSAUtil工具类
    String sign = RSAUtil.sign(secretPrivate, timestamp);

    //以下是调用方式,可以用自己习惯的方式。  demo选择的方式是HttpClient,不做详细说明。
    Map<String, String> params = new HashMap<String,String>();
    params.put("email", email);
    params.put("pwd", pwd);
    params.put("timestamp", timestamp);
    params.put("sign", sign);
    String post = HttpUtil.post(url, params);
    System.out.println(post);

    /**
     *  需要注意:
     *  pwd: 必须经过加盐加密。
     *  timestamp: 发送接口的值和进行sign加密的值 必须是同一个。
     *  如果请求发生403情况,请在Header加User-Agent。否则CDN会认为该请求在攻击,会进行拦
截。
     *
     */
  }

}
```

MD5Util工具类

```
import java.security.MessageDigest;

public class MD5Util {

    public MD5Util() {
    }

    public static final String encrypt(String s) {
        char[] hexDigits = new char[]{'0', '1', '2', '3', '4', '5', '6', '7',
'8', '9', 'A', 'B', 'C', 'D', 'E', 'F'};

        try {
            byte[] btInput = s.getBytes();
            // 获得MD5摘要算法的 MessageDigest 对象
            MessageDigest mdInst = MessageDigest.getInstance("MD5");
            // 使用指定的字节更新摘要
            mdInst.update(btInput);
            // 获得密文
            byte[] md = mdInst.digest();
            // 把密文转换成十六进制的字符串形式
```

```
            int j = md.length;
            char[] str = new char[j * 2];
            int k = 0;

            for(int i = 0; i < j; ++i) {
                byte byte0 = md[i];
                str[k++] = hexDigits[byte0 >>> 4 & 15];
                str[k++] = hexDigits[byte0 & 15];
            }

            return new String(str);
        } catch (Exception var10) {
            return null;
        }
    }

    public static final String encryptPwd(String pwd) {
        String afterFormat = null;
        String afterEncrypt = null;

        try {
            // 盐值
            String salt = "dig?F*ckDang5PaSsWOrd&%(12lian0160630).";
            // pwd+盐值
            afterFormat = pwd + salt;
            // 去进行MD5加密
            afterEncrypt = encrypt(afterFormat);
        } catch (Exception var4) {
        }

        return afterEncrypt;
    }

}
```

RSAUtil工具类

```
import java.security.*;
import java.security.spec.PKCS8EncodedKeySpec;
import java.text.SimpleDateFormat;

import sun.misc.BASE64Decoder;

public class RSAUtil {

  public static final String KEY_ALGORITHM = "RSA";
  //加密算法
  public static final String SIGNATURE_ALGORITHM = "SHA256withRSA";
```

```
  public static final SimpleDateFormat df = new SimpleDateFormat("yyyy-MM-
dd'T'HH:mm:ss'Z'");

  public static String sign(String secretPrivate, String timestamp) {
      String out = null;
      try {
          out = sign(timestamp.getBytes(), secretPrivate);
      } catch (Exception e) {
          e.printStackTrace();
      }

      return out;
  }

  public static String sign(byte[] data, String privateKey) throws Exception {
      // 解密由base64编码的私钥
      byte[] keyBytes = decryptBASE64(privateKey);

      // 构造PKCS8EncodedKeySpec对象
      PKCS8EncodedKeySpec pkcs8KeySpec = new PKCS8EncodedKeySpec(keyBytes);

      // KEY_ALGORITHM 指定的加密算法
      KeyFactory keyFactory = KeyFactory.getInstance(KEY_ALGORITHM);

      // 取私钥匙对象
      PrivateKey priKey = keyFactory.generatePrivate(pkcs8KeySpec);

      // 用私钥对信息生成数字签名
      Signature signature = Signature.getInstance(SIGNATURE_ALGORITHM);
      signature.initSign(priKey);
      signature.update(data);

      return bytes2Hex(signature.sign());

  }

  public static byte[] decryptBASE64(String key) throws Exception {
      return (new BASE64Decoder()).decodeBuffer(key);
  }

  public static String bytes2Hex(byte[] bytes) {
      StringBuilder stringBuilder = new StringBuilder("");
      if (bytes == null || bytes.length <= 0) {
          return null;
      }
      for (int i = 0; i < bytes.length; i++) {
          int v = bytes[i] & 0xFF;
          String hv = Integer.toHexString(v);
          if (hv.length() < 2) {
```

```
                stringBuilder.append(0);
            }
            stringBuilder.append(hv);
        }
        return stringBuilder.toString();
    }


}
```

委托下单

通过调用该接口，可以使用平台的生成委托单功能，该功能可以生成限价买单和限价卖单

描述

- request_url：baseUrl + order/order
- method：POST
- Content-Type:application/x-www-form-urlencoded
- Accept:application/json, text/plain, /
- parameter：

item	description	type
buyOrSell	买卖方向，1是buy，2是sell	int
currencyId	交易币种	int
baseCurrencyId	基础币种	int
fdPassword	交易密码 (可以为空,但必须传fdPassword="")	string
num	数量，保留小数点后最多8位小数	decimal
price	价格，保留小数点后最多8位小数	decimal
source	程序化交易对接类型，该值目前为5	int
type	交易类型，该值目前为1	int
token	请求认证，如果token过期后需要重新获取	string
uid	用户id	int
local	语种，繁体中文,可选zh_TW、en_US	string
timestamp	时间戳	string

- response description：当接口返回的status 为200时，则attachment包含以下数据，如果status参数不为200 ，则出现异常。
- response_data:

item	description	type
attachment	单号	string

```
{
    "attachment": "152472056360800300103411002618 23"
    , "message": ""
    , "status": 200
}
```

委托撤单

通过调用该接口，可以使用平台的撤销委托单功能

描述

- request_url：baseUrl + order/cancel
- method：POST
- Content-Type:application/x-www-form-urlencoded
- Accept:application/json, text/plain, /
- parameter：

item	description	type
currencyId	基础币种id	int
orderNo	委托单号	string
fdPassword	交易密码	string
source	程序化交易对接类型，该值目前为5	int
token	请求认证，如果token过期后需要重新获取	string
uid	用户id	int
local	语种，繁体中文,可选zh_TW、en_US	string
timestamp	时间戳	string

- response description：当接口返回的status 为200时，则attachment包含以下数据，如果status参数不为200 ，则出现异常。
- response_data:

```
{
    "attachment":200
    ,"message":""
    ,"status":200
}
```


账户资产查询

平台为用户提供账户资产查询功能。

描述

- request_url：baseUrl + coin/customerCoinAccount
- method：POST
- Content-Type:application/x-www-form-urlencoded
- Accept:application/json, text/plain, /
- parameter：

item	description	type
token	请求认证，如果token过期后需要重新获取	string
uid	用户id	int
local	语种，繁体中文,可选zh_TW、en_US	string
timestamp	时间戳	string

- response description：当接口返回的status 为200时，则attachment包含以下数据，如果status参数不为200 ，则出现异常。
- response_data:

item	description	type
allMoney	总资产合计	decimal
coinList	用户个人持仓明细	object

coinList的内容

item	description	type
amount	持仓数量	decimal
baseCurrencyId	基础币id	int
btc_value	大约折合多少BTC	int
cashAmount	现金价值	decimal
currencyId	交易币id	int
currencyName	数字货币全称	string
currencyNameEn	数字货币简称	string
freezeAmount	冻结数量	decimal
icoUrl	币种缩略图	string


```
{
  "attachment": {
    "allMoney": 11.0,
    "takeBtcMax": 2.0,
    "takeBtcNum": 0.0,
    "coinList": [{
      "currencyNameEn": "LTC",
      "amount": "100.00000000",
      "freezeAmount": "0.00000000",
      "currencyName": "Litecoin",
      "cashAmount": "100.00000000",
      "icoUrl": "bestex/ltc.png",
      "baseCurrencyId": 2,
      "currencyId": 2,
      "btc_value": 10.0
    }, {
      "currencyNameEn": "BTC",
      "amount": "1.00000000",
      "freezeAmount": "0.00000000",
      "currencyName": "BitCoin",
      "cashAmount": "1.00000000",
      "icoUrl": "bestex/btc.png",
      "baseCurrencyId": 1,
      "currencyId": 1,
      "btc_value": 1.0
    }, {
      "currencyNameEn": "ETH",
      "amount": "0.00000000",
      "freezeAmount": "0.00000000",
      "currencyName": "Ethereum",
      "cashAmount": "0.00000000",
```

```
      "icoUrl": "bestex/eth.png",
      "baseCurrencyId": 3,
      "currencyId": 3,
      "btc_value": 0.0
    }, {
      "currencyNameEn": "BCH",
      "amount": "0.00000000",
      "freezeAmount": "0.00000000",
      "currencyName": "Bitcoin Cash",
      "cashAmount": "0.00000000",
      "icoUrl": "bestex/bch.png",
      "baseCurrencyId": 14,
      "currencyId": 14,
      "btc_value": 0.0
    }, {
      "currencyNameEn": "ETC",
      "amount": "0.00000000",
      "freezeAmount": "0.00000000",
      "currencyName": "Ethereum Classic",
      "cashAmount": "0.00000000",
      "icoUrl": "coinimg/etc.png",
      "baseCurrencyId": 15,
      "currencyId": 15,
      "btc_value": 0.0
    }, {
      "currencyNameEn": "BEB",
      "amount": "0.00000000",
      "freezeAmount": "0.00000000",
      "currencyName": "Be best",
      "cashAmount": "0.00000000",
      "icoUrl": "bestex/beb.png",
      "baseCurrencyId": 23,
      "currencyId": 23,
      "btc_value": 0.0
    }]
  },
  "message": null,
  "status": 200
}
```


用户委托单查询

通过该接口，可以查询用户的历史委托信息。

描述

- request_url：baseUrl + user/trOrderListByCustomer
- method：POST

- Content-Type:application/x-www-form-urlencoded
- Accept:application/json, text/plain, /
- parameter：

item	description	type
beginTime	查询开始日期，格式：2018-04-25	string
endTime	查询结束日期，格式：2018-04-26	string
start	查询起点，默认为1，最大为999 (start=第几页)	int
size	查询数量，最小值1，最大值20 (size=每页多少条数据)	int
status	委托单状态，未成交=0、部分成交=1、全部成交=2、撤单=4、未成交以及部分成交=11、全部状态=10（默认值10）	int
buyOrSell	买卖方向，0是全部方向，1是buy，2是sell	int
currencyId	交易币种id	int
baseCurrencyId	基础币种id	int
priceType	价格类型，默认为0，表示保留0位小数	
token	请求认证，如果token过期后需要重新获取	string
uid	用户id	int
local	语种，繁体中文,可选zh_TW、en_US	string
timestamp	时间戳	string

- response description：当接口返回的status 为200时，则attachment包含以下数据，如果status参数不为200 ，则出现异常。 返回数据按照orderTime降序排序
- response_data:

item	description	type
total	查询出来的条数	int
list	查询内容	object

list的内容

item	description	type
averagePrice	平均委托金额	decimal
baseCurrencyId	基础币id	int
baseCurrencyName	基础币名称	string
baseCurrencyNameEn	基础币简称	string
buyOrSell	买卖方向，0是全部方向，1是buy，2是sell	int
currencyName	数字货币全称	string
currencyNameEn	数字货币简称	string
dealAmount	成交金额	decimal
fee	交易费用	decimal
num	委托数量	decimal
orderNo	委托单号	string
orderTime	委托时间	string
price	委托价格	decimal
remainNum	剩余数量	decimal
status	委托单状态，未成交=0、部分成交=1、全部成交=2、撤单=4、全部状态=10	int
tradeNum	成交数量	decimal


```
{
    "attachment": {
        "total": 5,
        "list": [
            {
                "currencyNameEn": "LTC",
                "remainNum": "0.00000000",
                "orderNo": "152472243500400500103511002235O5",
                "dealAmount": "0.00000000",
                "fee": "0.00000000",
                "num": "0.01000000",
                "tradeNum": "0.00000000",
                "baseCurrencyId": 1,
                "baseCurrencyName": "BitCoin",
                "baseCurrencyNameEn": "BTC",
                "buyOrSell": 1,
                "orderTime": "2018-04-26 14:00:35",
```

```
            "currencyName": "Litecoin",
            "price": "10.00000000",
            "averagePrice": "0.00000000",
            "status": 4
        },
        {
            "currencyNameEn": "LTC",
            "remainNum": "0.00000000",
            "orderNo": "152472078131300400103511002998 23",
            "dealAmount": "0.00000000",
            "fee": "0.00000000",
            "num": "0.01000000",
            "tradeNum": "0.00000000",
            "baseCurrencyId": 1,
            "baseCurrencyName": "BitCoin",
            "baseCurrencyNameEn": "BTC",
            "buyOrSell": 1,
            "orderTime": "2018-04-26 13:33:01",
            "currencyName": "Litecoin",
            "price": "10.00000000",
            "averagePrice": "0.00000000",
            "status": 4
        },
        {
            "currencyNameEn": "LTC",
            "remainNum": "0.00000000",
            "orderNo": "152472056360800300103411002618 23",
            "dealAmount": "0.00000000",
            "fee": "0.00000000",
            "num": "0.01000000",
            "tradeNum": "0.00000000",
            "baseCurrencyId": 1,
            "baseCurrencyName": "BitCoin",
            "baseCurrencyNameEn": "BTC",
            "buyOrSell": 1,
            "orderTime": "2018-04-26 13:29:23",
            "currencyName": "Litecoin",
            "price": "1.00000000",
            "averagePrice": "0.00000000",
            "status": 4
        },
        {
            "currencyNameEn": "LTC",
            "remainNum": "0.00000000",
            "orderNo": "152472053301700200103411002929 77",
            "dealAmount": "0.00000000",
            "fee": "0.00000000",
            "num": "0.01000000",
            "tradeNum": "0.00000000",
```

```
                "baseCurrencyId": 1,
                "baseCurrencyName": "BitCoin",
                "baseCurrencyNameEn": "BTC",
                "buyOrSell": 1,
                "orderTime": "2018-04-26 13:28:53",
                "currencyName": "Litecoin",
                "price": "1.00000000",
                "averagePrice": "0.00000000",
                "status": 4
            },
            {
                "currencyNameEn": "LTC",
                "remainNum": "0.00000000",
                "orderNo": "15247205167340010010381100231945",
                "dealAmount": "0.00000000",
                "fee": "0.00000000",
                "num": "1.00000000",
                "tradeNum": "0.00000000",
                "baseCurrencyId": 1,
                "baseCurrencyName": "BitCoin",
                "baseCurrencyNameEn": "BTC",
                "buyOrSell": 2,
                "orderTime": "2018-04-26 13:28:36",
                "currencyName": "Litecoin",
                "price": "1.00000000",
                "averagePrice": "0.00000000",
                "status": 4
            }
        ]
    },
    "message": null,
    "status": 200
}
```


委托单查询-通过订单号

通过订单号和交易币id和计价币id,查询订单信息。

描述

- request_url：baseUrl + user/trOrderByOrderIds
- method：POST
- Content-Type:application/x-www-form-urlencoded
- Accept:application/json, text/plain, /
- parameter：

item	description	type
orderIds	订单号 下方有详细说明	string
currencyId	交易币种id	int
baseCurrencyId	计价币种id	int
uid	用户id	int

- orderIds参数详细说明:
- 1.单个订单号查询：'订单号A'
- 2.批量订单号查询：'订单号A','订单号B','订单号C'
- 注意：
- 1.该参数首尾都有单引号' 'xxxxx' 。
- 2.如果需要批量查询中间以逗号隔开 'xxxx','xxxx','xxxx' 。
- response description：当接口返回的status 为200时，则attachment包含以下数据，如果status参数不为200 ，则出现异常。
- response_data:

item	description	type
attachment	订单信息	int
message	信息	object
status	状态	int

attachment

item	description	type
averagePrice	平均委托金额	decimal
baseCurrencyId	基础币id	int
baseCurrencyName	基础币名称	string
baseCurrencyNameEn	基础币简称	string
buyOrSell	买卖方向，0是全部方向，1是buy，2是sell	int
currencyName	数字货币全称	string
currencyNameEn	数字货币简称	string
dealAmount	成交金额	decimal
fee	交易费用	decimal
num	委托数量	decimal
orderNo	委托单号	string
orderTime	委托时间	string
price	委托价格	decimal
remainNum	剩余数量	decimal
status	委托单状态，未成交=0、部分成交=1、全部成交=2、全部撤单=4、部分成交后撤单=5、	int
tradeNum	成交数量	decimal


```
{
    "attachment": [
        {
            "start": 1,
            "size": 20,
            "total": 0,
            "orderBy": "desc",
            "ctbOrderId": 1,
            "orderNo": "150607278347800101904311006544 70",
            "customerUuid": "ead60e1231904f4fa5733b6cc13c15c9",
            "currencyId": 6,
            "baseCurrencyId": 1,
            "buyOrSell": 1,
            "type": 1,
            "num": 12,
            "tradeNum": 12,
            "cancelNum": 0,
```

```
            "remainNum": 0,
            "price": 0.0015602,
            "source": 1,
            "orderTime": "2017-09-22 17:33:03",
            "fee": 0,
            "freezeFee": 0,
            "freezeAmount": 0,
            "riskFlag": 1,
            "cancelTime": "2017-09-22 17:33:03",
            "averagePrice": 0.0015602,
            "status": 2,
            "createTime": "2017-09-22 17:33:03",
            "beginTime": null,
            "endTime": null,
            "createBy": null,
            "lastEditTime": "2017-09-22 18:03:47",
            "lastEditBy": null,
            "dealAmount": null,
            "currencyNameEn": "LSK",
            "currencyName": "Lisk",
            "baseCurrencyNameEn": "BTC",
            "baseCurrencyName": "Bitcoin"
        }
    ],
    "message": null,
    "status": 200
}
```