**Python交易API接口开发规范**

目录

[一、 概览 4](#_Toc185002813)

[1.1. 背景 4](#_Toc185002814)

[1.2. API接口概要及架构 5](#_Toc185002815)

[1.3. 全局的错误代码 6](#_Toc185002816)

[二、 股票交易API接口参考 8](#_Toc185002817)

[1. 接口工作过程 9](#_Toc185002818)

[2. 接口模式 9](#_Toc185002819)

[3. 函数接口说明 9](#_Toc185002820)

[1) createStockApi 方法 9](#_Toc185002821)

[2) onFrontConnected 方法 10](#_Toc185002822)

[3) onFrontDisconnected方法 10](#_Toc185002823)

[4) onHeartBeatWarning方法 11](#_Toc185002824)

[5) 登录reqUserLogin/onRspUserLogin/onRspNeedVerify/ onRspAccount 12](#_Toc185002825)

[6) 登出 reqUserLogout/onRspUserLogout方法 14](#_Toc185002826)

[7) 修改密码reqPasswordUpdate/onRspPasswordUpdate 15](#_Toc185002827)

[8) 报单reqOrderInsert/onRspOrderInsert 16](#_Toc185002828)

[9) 订单推送onRtnOrder 20](#_Toc185002829)

[10) 成交推送onRtnTrade 22](#_Toc185002830)

[11) 资金信息推送onRtnCapital 24](#_Toc185002831)

[12) 改单请求/返回reqOrderModify/onRspOrderModify方法 27](#_Toc185002832)

[13) 撤单请求/返回reqOrderCancel/onRspOrderCancel 29](#_Toc185002833)

[14) 委托单查询/返回reqQryOrder/onRspQryOrder 31](#_Toc185002834)

[15) 成交查询请求/返回reqQryTrade/onRspQryTrade 33](#_Toc185002835)

[16) 持仓明细查询请求/返回reqQryPosition/onRspQryPosition 34](#_Toc185002836)

[17) 资金查询请求/返回reqQryCapital/onRspQryCapital 36](#_Toc185002837)

[18) 经纪商查询请求/返回reqQryBroker/onRspQryBroker 39](#_Toc185002838)

[19) 版本号查询请求/返回reqQryVersion/onRspQryQryVersion 40](#_Toc185002839)

[20) 币种查询请求/返回reqQryCurrency/ onRspQryCurrency 41](#_Toc185002840)

[21) 交易时间查询请求/返回reqQryTradeTime/ onRspQryTradeTime 42](#_Toc185002841)

[22) 交易所查询请求/返回reqQryExchange /onRspQryExchange 43](#_Toc185002842)

[23) 合约查询请求/返回reqQryInstrument/onRspQryInstrument 44](#_Toc185002843)

[24) 跳点查询请求/返回reqQryTick/ onRspQryTick 47](#_Toc185002844)

[25) 订单类型查询请求/返回reqQryOrderType/ onRspQryOrderType 48](#_Toc185002845)

[26) 设置双重认证密码/返回reqSetVerifyQA/ onRspSetVerifyQA 49](#_Toc185002846)

[27) 绑定双重认证需要的MAC地址/返回reqSafeVerify/ onRspSafeVerify 50](#_Toc185002847)

[三、 期货交易API接口参考 51](#_Toc185002848)

[1. 接口工作过程 53](#_Toc185002849)

[2. 接口模式 53](#_Toc185002850)

[3. 函数接口说明 54](#_Toc185002851)

[1) CreateFutureApi方法 54](#_Toc185002852)

[2) OnFrontConnected 方法 54](#_Toc185002853)

[3) OnFrontDisconnected方法 54](#_Toc185002854)

[4) OnHeartBeatWarning方法 55](#_Toc185002855)

[5) 登录ReqUserLogin/OnRspUserLogin/OnRspNeedVerify/OnRspAccount 56](#_Toc185002856)

[6) 登出 ReqUserLogout/OnRspUserLogout方法 58](#_Toc185002857)

[7) 修改密码ReqPasswordUpdate/OnRspPasswordUpdate 59](#_Toc185002858)

[8) 报单ReqOrderInsert/OnRspOrderInsert 60](#_Toc185002859)

[9) 订单推送OnRtnOrder 66](#_Toc185002860)

[10) 成交推送OnRtnTrade 68](#_Toc185002861)

[11) 持仓信息推送OnRtnPosition 70](#_Toc185002862)

[12) 资金信息推送](#_Toc185002863)~~[OnRtnAccount](#_Toc185002863)~~ [OnRtnCapital 71](#_Toc185002863)

[13) 改单请求/返回ReqOrderModify/OnRspOrderModify方法 74](#_Toc185002864)

[14) 撤单请求/返回ReqOrderCancel/OnRspOrderCancel 76](#_Toc185002865)

[15) 委托单查询/返回ReqQryOrder/OnRspQryOrder 79](#_Toc185002866)

[16) 成交查询请求/返回ReqQryTrade/OnRspQryTrade 82](#_Toc185002867)

[17) 持仓明细查询请求/返回ReqQryPosition/OnRspQryPosition 84](#_Toc185002868)

[18) 资金查询请求/返回ReqQryCapital/OnRspQryCapital 86](#_Toc185002869)

[19) 版本号查询请求/返回ReqQryVersion/OnRspQryVersion 88](#_Toc185002870)

[20) 币种查询请求/返回ReqQryCurrency/ OnRspQryCurrency 89](#_Toc185002871)

[21) 交易时间查询请求/返回ReqQryTradeTime/ OnRspQryTradeTime 90](#_Toc185002872)

[22) 交易所查询请求/返回ReqQryExchange /OnRspQryExchange 91](#_Toc185002873)

[23) 合约查询请求/返回ReqQryInstrument/OnRspQryInstrument 92](#_Toc185002874)

[24) 策略查询请求/返回ReqQryStrategy/ OnRspQryStrategy 95](#_Toc185002875)

[25) 策略明细查询请求/返回ReqQryStrategyDetail/ OnRspQryStrategyDetail 98](#_Toc185002876)

[25) 设置双重认证密码/返回ReqSetVerifyQA/ OnRspSetVerifyQA 100](#_Toc185002877)

[26) 绑定双重认证需要的MAC地址请求/返回ReqSafeVerify/ OnRspSafeVerify 101](#_Toc185002878)

[27) 查询品种列表请求/返回ReqQryCommodity/ OnRspQryCommodity 102](#_Toc185002879)

[28) 持仓总结查询请求/返回ReqQryTotalPosition/ OnRspQryTotalPosition 104](#_Toc185002880)

[29) 交易所冬夏令开盘时间查询请求/返回ReqQryExchangeTime/ OnRspQryExchangeTime 106](#_Toc185002881)

[30) 品种开收盘时间查询请求/返回ReqQryExchangeTime/ OnRspQryCommodityTime 107](#_Toc185002882)

[四、 常见问题-期货交易相关 108](#_Toc185002883)

[五、 常见问题-股票交易相关 110](#_Toc185002884)

# 概览

## 背景

本API可用来交易中国境外期货和股票，并且用来获取相应市场的行情数据。

目前支持的交易所/市场有以下(但不限于)这些：

期货类

|  |  |  |
| --- | --- | --- |
| No | 交易所 | 说明 |
| 1 | CME | 包括CME group旗下的CME,COMEX,NYMEX,CBOT  官方网站：  https://www.cmegroup.com/ |
| 2 | HKEX | 香港交易所  官方网站：  http://www.hkex.com.hk |
| 3 | SGX | 新加坡交易所  官方网站：  http://www.sgx.com |
| 4 | ICE | 洲际交易所  官方网站：  https://www.theice.com/index |
| 5 | LME | 伦敦金属交易所  官方网站：  https://www.lme.com/ |

股票类

|  |  |  |
| --- | --- | --- |
| No | 市场 | 说明 |
| 1 | NASD | 美国市场的股票。  需要注意的是，2008年金融危机后，美国通过法案，允许上市公司在美国所有的交易所都可以交易。比如，在NYSE挂牌的股票可以在NASDQ交易所交易，反之亦然。 |
| 2 | HKEX | 香港交易所的股票 |
| 3 | KRX | 韩国的股票 |

## API接口概要及架构

本API是一个基于Python的类库，通过使用和扩展类库提供的接口来实现交易所需的功能。鉴于国内期货界CTP接口的广泛使用，本API接口仿照CTP接口的方式，使得有CTP接口使用经验得用户能够快速上手开发。

同时由于股票和期货的某些不同，股票的交易接口和期货的交易接口被独立开来，但是API库是同一个库，敬请用户留意。下面两个小节按期货和股票分别加以说明。

**期货接口使用方式**

期货交易客户端

(使用DAApi.dll)

期货前置服务器

期货行情客户端

(使用DAApi.dll)

期货行情服务器

**股票接口使用方式**

股票交易客户端

(使用DAApi.dll)

股票前置服务器

股票行情客户端

(使用DAApi.dll)

股票行情服务器

本API中无论任何地方，对于以下的错误码，值为0时表示成功。

// 错误码

ErrorID[int]

## 全局的错误代码

1、业务函数中通用错误

//期货接口中的错误信息结构体

dict error

{

ErrorID[int]:错误码

ErrorMsg[str]:错误描述

};

//股票接口中的错误信息结构体

dict error

{

ErrorID[int]:错误码

ErrorMsg[str]:错误描述

};

|  |  |  |
| --- | --- | --- |
| 错误代码  (ErrorID) | 错误描述 | 说明 |
| 0 | 成功 | 各种函数返回结果成功 |
| 90001 | 解析协议头失败 | 直达系统内部错误 |
| 90002 | 解压缩数据失败 | 直达系统内部错误 |
| 90003 | 未压缩数据体是空的 | 直达系统内部错误；通常指查询的结果为空，服务器端有应答，但是没有业务数据 |
| 90004 | 压缩数据体是空的 | 直达系统内部错误；通常指查询的结果为空，服务器端有应答，但是没有业务数据 |
| 90005 | 解密失败 | 直达系统内部错误 |
| 90006 | 服务器返回数组是空的 | 直达系统内部错误；通常指查询的结果为空，服务器端有应答，但是没有业务数据 |
|  |  |  |
|  |  |  |

2、连接断开通用错误

onFrontDisconnected (nReason: int)

|  |  |  |
| --- | --- | --- |
| 错误代码  (nReason) | 错误描述 | 说明 |
| 4001 | 设置阻塞失败 | Socket底层错误 |
| 4002 | 设置非阻塞失败 | Socket底层错误 |
| 4003 | 接收数据失败 | Socket底层错误 |
| 4004 | 接收0长度数据，已断开 | Socket底层错误 |
| 4005 | 发送数据失败 | Socket底层错误 |
| 4006 | 发送0长度数据，已断开 | Socket底层错误 |
| 4007 | 选择失败 | Socket底层错误 |
| 4008 | 服务器无响应 | Socket底层错误 |
|  |  |  |
|  |  |  |

# 股票交易API接口参考

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 业务类型 | 业务 | 请求接口 | 说明 | 响应接口 |
| 连接 | 连接 | init | 连接回报 | onFrontConnected |
| 断开 |  |  | 连接断开回报 | onFrontDisconnected |
| 心跳 |  |  | 心跳警告 | onHeartBeatWarning |
| 登录 | 登录 | reqUserLogin | 登录回报 | onRspUserLogin/onRspNeedVerify |
| 登出 | reqUserLogout | 登出回报 | onRspUserLogout |
| 修改密码 | reqPasswordUpdate | 修改密码回报 | onRspPasswordUpdate |
| 双重认证step1 | reqSetVerifyQA | 设置双重认证密码 | onRspSetVerifyQA |
| 双重认证step2 | reqSafeVerify | 绑定MAC地址 | onRspSafeVerify |
| 交易 | 报单 | reqOrderInsert | 报单回报 | onRspOrderInsert |
| 改单 | reqOrderModify | 改单回报 | onRspOrderModify |
| 撤单 | reqOrderCancel | 撤单回报 | onRspOrderCancel |
| 推送 |  |  | 报单状态推送 | onRtnOrder |
|  |  | 成交信息推送 | onRtnTrade |
|  |  | *持仓信息推送* | *onRtnPosition* |
|  |  | 资金信息推送 | onRtnAccount |
| 查询 | 委托单查询 | reqQryOrder | 委托单查询回报 | onRspQryOrder |
| 成交查询 | reqQryTrade | 成交查询回报 | onRspQryTrade |
| 持仓明细查询 | reqQryPosition | 持仓明细查询回报 | onRspQryPosition |
| 资金查询 | reqQryCapital | 资金查询回报 | onRspQryCapital |
| 经纪商查询  (港股专有) | reqQryBroker | 经纪商查询回报 | onRspQryBroker |
| 版本号查询 | reqQryVersion | 版本号查询回报 | onRspQryQryVersion |
| 币种查询 | reqQryCurrency | 币种查询回报 | onRspQryCurrency |
| 交易时间查询 | reqQryTradeTime | 交易时间查询回报 | onRspQryTradeTime |
| 交易所查询 | reqQryExchange | 交易所查询回报 | onRspQryExchange |
| 合约查询 | reqQryInstrument | 合约查询回报 | onRspQryInstrument |
| 跳点查询 | reqQryTick | 跳点查询回报 | onRspQryTick |
| *订单类型查询* | *ReqQryOrderType* | *订单类型查询回报* | *onRspQryOrderType* |

注：以上斜体字的接口，当前API版本未实现。

## 接口工作过程



## 接口模式

stock\_demo.py文件中提供了一个StockApi类，通过继承StockApi类可以发出操作请求，也可以重载回调函数来处理后台服务的响应。

## 函数接口说明

### createStockApi 方法

创建实例方法。

**函数原形**：

createStockApi(bRecordLog: bool, lpszLogFileName: str = "Stock.log", lpszLogFileDir: str = "");

bRecordLog: true表示记录日志，false表示不记录日志。

lpszLogFileName：日志文件。

lpszLogFileDir： 日志文件文件夹路径。

### onFrontConnected 方法

当客户端与前置建立起通信连接时（还未登录前），该方法被调用。

**函数原形：**

onFrontConnected()

本方法在完成初始化后调用，可以在其中完成用户登录任务。

### onFrontDisconnected方法

当客户端与前置通信连接断开时，该方法被调用。当发生这个情况

后，API会自动重新连接，客户端可不做处理。自动重连地址，可能是原来注册

的地址，也可能是系统支持的其它可用的通信地址，它由程序自动选择。

**函数原形：**

onFrontDisconnected (nReason: int)

**参数：**

nReason：连接断开原因

### onHeartBeatWarning方法

心跳超时警告。当长时间未收到报文时，该方法被调用。

**函数原形：**

onHeartBeatWarning(nTimeLapse: int)

**参数：**

nTimeLapse：距离上次接收报文的时间

### 登录reqUserLogin/onRspUserLogin/onRspNeedVerify/ onRspAccount

从API v1.5引入双重认证，具体交互过程请参阅**“国际期货股票API\_双重认证交互说明.xlsx”**

函数原型：

|  |
| --- |
| reqUserLogin(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户ID  UserId[str]  // 用户密码  UserPwd[str]  // 用户类型  UserType[str]  // 软件名称  SoftwareName[str]  // 软件版本号  SoftwareVersion[str]  // 授权码  AuthorCode[str]  //登录机器网卡MAC地址(since API v1.5,双重认证使用)  MacAddress[str]  // 错误信息  ErrorDescription[str] |

1. 函数原型：

|  |
| --- |
| onRspUserLogin(error: dict, reqid: int, last: bool) |

参数：error

错误码：

|  |  |
| --- | --- |
| **登录错误码** | **错误内容** |
| 10001 | 用户名不正确 |
| 10002 | 登录密码错误 |
| 10003 | 密码错误次数超限，用户已冻结，请联系客服解冻！ |
| 10004 | 用户已被冻结，请联系客服解冻！ |
| 10013 | 授权码信息不正确，请确认UserType，SoftwareName，SoftwareVersion，AuthorCode的值 |

1. 函数原型：

服务器端在判断到需要做双重认证时，会指示API回调此函数

|  |
| --- |
| onRspNeedVerify(bFirstLogin: bool, bHasSetQA: bool); |

参数：bFirstLogin

|  |
| --- |
| 当设备(根据login消息中的MacAddress字段识别)第一次登录交易前置时， onRspNeedVerify函数被回调，且bFirstLogin 的值为true,用户在回调函数中需调用reqSetVerifyQA函数设置双重认证信息。  当设备(根据login消息中的MacAddress字段识别)第一次以后登录交易前置时，onRspNeedVerify()函数不会被回调。 |

1. 函数原型：

在onRspUserLogin指示成功登录回调后，该函数被回调，data字典中返回每个币种对应的资金账号

|  |
| --- |
| onRspAccount(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 帐户币种  CurrencyNo[str]  // 资金账号  AccountNo[str] |

### 登出 reqUserLogout/onRspUserLogout方法

函数原型：

|  |
| --- |
| reqUserLogout(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户ID  UserId[str]      // 资金账号  AccountNo[str]      // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspUserLogout(error: dict, reqid: int, last: bool) |

参数：error

### 修改密码reqPasswordUpdate/onRspPasswordUpdate

函数原型：

|  |
| --- |
| reqPasswordUpdate(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户代码  UserId[str]  // 用户原密码  OldPassword[str]  // 用户新密码  NewPassword[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspPasswordUpdate(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 用户代码  UserId[str]  // 用户原密码  OldPassword[str]  // 用户新密码  NewPassword[str]  // 错误信息  ErrorDescription[str] |

参数：error

### 报单reqOrderInsert/onRspOrderInsert

说明：

SystemNo[str] : 系统编号

LocalNo[str]: 本地编号

OrderNo[str]: 订单号

LocalNo是使用API的程序给的订单号,相当于CTP里面的orderref，当日内最好不要重复

SystemNo是直达系统给的订单号；

OrderNo是交易所给的订单号，相当于CTP里面的OrderSysID

函数原型：

|  |
| --- |
| reqOrderInsert(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户ID  UserId[str]  // 资金账号(一个用户可以有多个币种的资金，所以会有多个资金账号，下单需指定)  // 请参照登录成功后的回调函数OnRspAccount的说明  AccountNo[str]  // 本地编号  LocalNo[str]  // 交易所代码  ExchangeCode[str]  // 合约代码  ContractCode[str]  // 买还是卖：1=buy 2=sell  BidAskFlag[str]  //卖空时使用,1=开仓;2=平仓,同时和订单的买卖方向相反。非卖空时，0或不设置值  OpenCloseFlag[str]  // 下单数  OrderQty[str]  // 下单价格  OrderPrice[str]  // 定单类型：(from v1.12)  // 美股：1=限价单  // 港股：5=竞价限价盘; 6=竞价盘; 7=增强限价;8=特别限价盘  OrderType[str]  // 触发价格(股票中未使用)  TriggerPrice[str]  // 有效日期（1=当日有效）  TIF[str]  // 策略ID(股票中未使用)  StrategyId[str]  // 显示委托量，必须小于委托量(股票中未使用)  MaxShow[str]  // 最小成交量(股票中未使用)  MinQty[str]  // 错误信息  ErrorDescription[str] |

说明：

API用户下单后，onRspOrderInsert函数会被回调两次：

1. 第一次回调返回直达系统给这笔订单指定的系统号SystemNo，表明直达系统收到并确认这笔订单。
2. 第二次回调返回交易所的反馈的订单号OrderNo(如果一切正常)。通常资金不足等情况发生时，不会有订单号，因为订单指令没有向交易所发送。

函数原型：

|  |
| --- |
| onRspOrderInsert(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 用户ID  UserId[str]  // 资金账号  AccountNo[str]  // 系统编号  SystemNo[str]  // 本地编号  LocalNo[str]  // 定单号  OrderNo[str]  // 交易所代码  ExchangeCode[str]  // 合约代码  ContractCode[str]  // 买还是卖：1=buy 2=sell  BidAskFlag[str]  // 下单数  OrderQty[str]  // 下单价格  OrderPrice[str]  // 定单类型：(from v1.12)  // 美股：1=限价单  // 港股：5=竞价限价盘; 6=竞价盘; 7=增强限价;8=特别限价盘  OrderType[str]  // 下单日期  OrderDate[str]  // 下单时间  OrderTime[str]  // 错误代码(保留，未使用)  ErrorCode[str]  // 订单状态（2：已排队；3：部分成交；4：完全成交；5：已撤余单；6：已撤单；7：指令失败；8：待送出；9：待更改；A：待撤单）  OrderState[str]  // 撤单的用户ID  CancelUserId[str]  // 触发价格  TriggerPrice[str]  // 有效日期（1=当日有效）  TIF[str]  //卖空时使用,1=开仓;2=平仓,同时和订单的买卖方向相反。非卖空时，0或不设置值  OpenCloseFlag[str]  // 策略ID(股票中未使用)  StrategyId[str]  // 显示委托量，必须小于委托量(股票中未使用)  MaxShow[str]  // 最小成交量(股票中未使用)  MinQty[str]  // 交易所返回时间  ExchangeTime[str] |

参数：error

|  |
| --- |
| // 错误码(用来判断报单成功或失败)  ErrorID[int]  // 错误描述  ErrorMsg[str] |

错误码

|  |  |
| --- | --- |
| **系统号错误码** | **错误内容** |
| 20005 | 系统号生成失败 |

下单错误码

|  |  |
| --- | --- |
| **下单错误码** | **错误内容** |
| 20000 | 下单失败 |
| 20001 | 资金不足 |
| 20002 | 交易服务器未连接 |
| 20003 | 您已被禁止交易，请联系客服解禁 |
| 20004 | 下单被拒绝 |
| 20005 | 系统号生成失败 |
| 20006 | 您的资金账户中没有该合约交易所需的币种 |
| 20007 | 该合约已经到期，不能交易 |
| 20008 | 该交易市场未开市，不能交易 |
| 20009 | 该交易市场已闭市，不能交易 |
| 20010 | 下单价格超出限定范围 |
| 20011 | 下单数量过大 |
| 20012 | 您被禁止交易该合约，请联系客服 |
| 20013 | 您的资金账户风险率过高，已被禁止交易 |
| 20015 | 您的账户交易数据出现异常，暂停交易 |
| 20019 | 下单数量超过客户持仓限量 |
| 20020 | 下单数量超过客户可下单买量 |
| 20021 | 下单数量超过客户可下单卖量 |
| 20022 | 下单数量超过公司持仓限量 |
| 20023 | 下单数量超过公司可下单买量 |
| 20024 | 下单数量超过公司可下单卖量 |
| 20025 | Dealer下单数量超过最大可下单数量 |
| 20026 | 合约临近交割，只可平仓，不能开仓 |
| 20027 | 该合约已过最后交易日时间，禁止电子盘交易 |
| 20028 | 商品未开放，禁止交易 |
| 20029 | 单笔下单数量超过该商品单笔最大可下单数 |
| 20032 | 持仓不足，平仓失败 |
| 20033 | 您不能交易该商品,请联系客服开通 |

### 订单推送onRtnOrder

函数原型：

|  |
| --- |
| onRtnOrder(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 交易所代码      ExchangeNo[str]      // 合约代码      TreatyCode[str]      // 定单号      OrderNo[str]      // 委托数量      OrderNumber[int]      // 已成交数量      FilledNumber[int]      // 成交均价      FilledAdvPrice[float]      // 持买数量      BuyHoldNumber[int]      // 持买开仓均价      BuyHoldOpenPrice[float]      // 持买均价      BuyHoldPrice[float]      // 持卖数量      SaleHoldNumber[int]      // 持卖开仓均价      SaleHoldOpenPrice[float]      // 持卖均价      SaleHoldPrice[float]      // 是否已经撤单（0：没有；1：已撤单）      IsCanceled[str]      // 成交总的手续费      FilledTotalFee[float]      // 顺序号      Status[int]      // 资金帐号      AccountNo[str]      // 持仓类型（0：昨仓；1：今仓）      HoldType[str]      // 持买保证金      HoldMarginBuy[float]      // 持卖保证金  HoldMarginSale[float]  // 最新价  CurrPrice[float]  // 浮动盈亏  FloatProfit[float] |

参数：error

### 成交推送onRtnTrade

函数原型：

|  |
| --- |
| onRtnTrade(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 用户ID  UserId[str]  // 资金账号  AccountNo[str]  // 成交编号（要包括7位的订单编号，一共11位）  FilledNo[str]  // 定单号  OrderNo[str]  // 系统编号  SystemNo[str]  // 本地编号  LocalNo[str]  // 交易所代码  ExchangeCode[str]  // 合约代码  TreatyCode[str]  // 买还是卖：1=buy 2=sell  BuySale[str]  // 成交数  FilledNumber[str]  // 成交价格  FilledPrice[str]  // 成交日期(yyyy-MM-dd)  FilledDate[str]  // 成交时间(hh:mm:ss)  FilledTime[str]  // 成交手续费  Commsion[str]  // 委托数量  HTSCode[str]  // 委托价格  ErrorCode[str]  // 合约交割日期(yyyyMMdd)  DeliveryDate[str]  // 成交类别(N：普通下单成交；C、T：调期模拟成交)  FilledType[str]  // 定单类型（1=限价单, 2=市价单，3=限价止损（stop to limit），4=止损（stop to market）OrderType[str]  // 有效日期（1=当日有效, 2=永久有效）  ValidDate[str]  // 开仓还是平仓：1=开仓 2=平仓，3=平今，4=平昨  AddReduce[str]  // 错误信息  ErrorDescription[str] |

参数：error

### 资金信息推送onRtnCapital

函数原型：

|  |
| --- |
| onRtnCapital(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 客户号  ClientNo[str]  // 资金帐号  AccountNo[str]  //币种  CurrencyNo[str]  //今可用  Available[float]  //昨可用  YAvailable[float]  //今可出  CanCashOut[float]  //今结存  Money[float]  //平仓盈亏  ExpiredProfit[float]  //冻结资金  FrozenDeposit[float]  //手续费  Fee[float]  //保证金  Deposit[float]  //维持保证金  KeepDeposit[float]  //状态  Status[int]  //入金  InMoney[float]  //出金  OutMoney[float]  //未到期平盈  UnexpiredProfit[float]  //今权益  TodayTotal[float]  //未结平盈  UnaccountProfit[float]  //期权权利金  Royalty[float]  //交易所代码  ExchangeNo[str]  //合约代码  TreatyCode[str]  //定单号  OrderNo[str]  //委托数量  OrderNumber[int]  //已成交数量  FilledNumber[int]  //成交均价  FilledAdvPrice[float]  //持买数量  BuyHoldNumber[int]  //持买开仓均价  BuyHoldOpenPrice[float]  //持买均价  BuyHoldPrice[float]  //持卖数量  SaleHoldNumber[int]  //持卖开仓均价  SaleHoldOpenPrice[float]  //持卖均价  SaleHoldPrice[float]  //是否已经撤单（0：没有；1：已撤单）  IsCanceled[str]  //成交总的手续费  FilledTotalFee[float]  //信任额度  Credit[float]  //孖展上限额度 20150727 added for 港股  MarginLimit[float]  //借货价值 20150727 added for 港股  BorrowValue[float]  //按揭价值 20150727 added for 港股  FMortgageMoney[float]  // T1 20160219 added for 港股  T1[float]  // T2 20160219 added for 港股  T2[float]  //T3 20160219 added for 港股  T3[float]  // Tn 20160219 added for 港股  TN[float]  //交易限额  TradeLimit[float]  //跨市场资金限额  FCrossMax[float]  //卖空冻结资金  SellFreezenMoney[float]  //卖空利息  SellInterest[float]  //需补按金  SellNeedAddMargin[float] |

### 改单请求/返回reqOrderModify/onRspOrderModify方法

函数原型：

|  |
| --- |
| reqOrderModify(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 系统编号  SystemNo[str]  // 用户ID  UserId[str]  // 本地编号  LocalNo[str]  // 资金账号  AccountNo[str]  // 定单号  OrderNo[str]  // 交易所代码  ExchangeCode[str]  // 合约代码  ContractCode[str]  // 买还是卖：1=buy 2=sell  BidAskFlag[str]  // 下单数  OrderQty[str]  // 下单价格  OrderPrice[str]  // 改单数  ModifyQty[str]  // 改单价格  ModifyPrice[str]  // 美股：1=限价单  // 港股：5=竞价限价盘; 6=竞价盘; 7=增强限价;8=特别限价盘  OrderType[str]  // 触发价格(股票中未使用)  TriggerPrice[str]  // 改单触发价格(股票中未使用)  ModifyTriggerPrice[str]  // 有效日期（1：当日有效）  TIF[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspOrderModify(data: dict, error: dict, reqid: int, last: bool) |

参数：error

|  |
| --- |
| // 错误码(用来判断改单成功或失败)  ErrorID[int]  // 错误描述  ErrorMsg[str] |

### 撤单请求/返回reqOrderCancel/onRspOrderCancel

函数原型：

|  |
| --- |
| reqOrderCancel(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户ID      UserId[str]      // 本地编号      LocalNo[str]      // 资金账号      AccountNo[str]      // 系统编号      SystemNo[str]      // 定单号      OrderNo[str]      // 交易所代码      ExchangeCode[str]      // 合约代码      ContractCode[str]      // 买还是卖：1=buy 2=sell      BidAskFlag[str]      // 下单数      OrderQty[str]      // 下单价格      OrderPrice[str]      // 订单类型      OrderType[str]      // 错误信息      ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspOrderCancel(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 用户ID      UserId[str]      // 资金账号      AccountNo[str]      // 系统编号      SystemNo[str]  // 本地号  LocalNo[str]  // 定单号      OrderNo[str]      // 撤单号      CancelNo[str]      // 交易所代码      ExchangeCode[str]  // 合约代码  ContractCode[str]  // 买还是卖：1=buy 2=sell  BidAskFlag[str]  // 下单数  OrderQty[str]  // 下单价格  OrderPrice[str]  // 已成交数  FilledQty[str]  // 已撤单数  CancelledQty[str]  // 订单类型  OrderType[str]  // 撤单日期  CancelledDate[str]  // 撤单时间  CancelledTime[str]  // 错误代码(保留，未使用)  ErrorCode[str] |
|  |

参数：error

|  |
| --- |
| // 错误码(用来判断撤单成功或失败)  ErrorID[int]  // 错误描述  ErrorMsg[str] |

### 委托单查询/返回reqQryOrder/onRspQryOrder

两个委托单查询之间最小间隔时间为1秒，否则查询失败。(Since API v1.5)

函数原型：

|  |
| --- |
| reqQryOrder(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户ID      UserId[str]      // 资金账号      AccountNo[str]      // 是否模拟用户：1：是；0 or other：不是      IsSimulation[str]      // 取得指定订单号以后的定单      OrderNo[str]      // 取得指定订单时间以后的定单（格式：yyyy-MM-dd hh:mm:ss）      OrderDateTime[str]      // 错误信息      ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryOrder(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 用户ID      UserId[str]      // 资金账号      AccountNo[str]      // 系统编号      SystemNo[str]      // 本地编号      LocalNo[str]      // 定单号      OrderNo[str]      // 交易所代码      ExchangeCode[str]  // 合约代码  ContractCode[str]  // 买还是卖：1=buy 2=sell  BidAskFlag[str]  // 下单数  OrderQty[str]  // 下单价格  OrderPrice[str]  // 已成交数  FilledQty[str]  // 成交均价  FilledPrice[str]  // 定单类型  OrderType[str]  // 下单日期  OrderDate[str]  // 下单时间  OrderTime[str]  // 错误代码  ErrorCode[str]  // 订单状态（1：已请求；2：已排队；3：部分成交；4：完全成交；5：已撤余单；6：已撤单；7：指令失败；8：待送出；9：待更改；A：待撤单）  OrderState[str]  // 撤单的用户ID  CancelUserId[str]  // 触发价格(股票未使用)  TriggerPrice[str]  // 有效日期（1=当日有效）  TIF[str]  // 卖空单使用  OpenCloseFlag[str]  // 策略ID (股票未使用)  StrategyId[str]  // 显示委托量，必须小于委托量(股票未使用)  MaxShow[str]  // 最小成交量(股票未使用)  MinQty[str]  // 交易所返回时间  ExchangeTime[str]  // 撤单时间  CancelTime[str] |

### 成交查询请求/返回reqQryTrade/onRspQryTrade

两个成交查询之间最小间隔时间为1秒，否则查询失败。(Since API v1.5)

函数原型：

|  |
| --- |
| reqQryTrade(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户  UserId[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryTrade(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| 请参照onRtnTrade数据返回字段 |

### 持仓明细查询请求/返回reqQryPosition/onRspQryPosition

两个持仓查询之间最小间隔时间为1秒，否则查询失败。(Since API v1.5)

函数原型：

|  |
| --- |
| reqQryPosition(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryPosition(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 用户ID  ClientNo[str]  // 交易所  ExchangeNo[str]  // 证券代码  ProductCode[str]  // 持仓方向（1：持买；2：持卖）  LongShortPosFlag[str]  // 持仓成本价  PosCostPrice[str]  // 可卖数量  CanSellShares[str]  // 今买数量  TodayBuyShares[str]  // 冻结数量  FrosenShares[str]  // 持有期内买入金额总和  TotalBuyMoney[float]  // 持有期内卖出金额总和  TotalSellMoney[float]  // 持有期内买入数量  TotalBuyShares[int]  // 持有期内卖出数量  TotalSellShares[int]  // 首次开仓日期(yyyy-MM-dd)  FirstPosDate[str]  // 平仓盈亏  ClosePosPL[float]  // T1日交割股数，正入负出  T1DeliveryShares[int]  // T2日交割股数，正入负出  T2DeliveryShares[int]  // T3日交割股数，正入负出  T3DeliveryShares[int]  // 未交割股数=T1+T2  NotDeliveryShares[int]  // 已交割股数  DeliveredShares[int]  // 卖空股数  SellShortShares[int]  // 卖空冻结资金  SellShortMoney[float]  // 卖空仓位的开仓均价(开仓均价)  SSPosAvgCostPrice[float] |

### 资金查询请求/返回reqQryCapital/onRspQryCapital

两个资金查询之间最小间隔时间为1秒，否则查询失败。(Since API v1.5)

函数原型：

|  |
| --- |
| reqQryCapital(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 不使用  Unused[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRsqQryCapital(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 用户ID  UserId[str]  // 入金  Deposit[str]  // 出金  Withdraw[str]  // 今日可用于交易的资金量(今可用)  TodayTradableFund[str]  // 当日期初权益(今结存)  TodayInitialBalance[str]  // 当日实时浮动权益(今权益)  TodayRealtimeBalance[str]  // 冻结资金  FrozenFund[str]  // 佣金  Commission[str]  // 初始保证金  InitialMargin[str]  // 昨日可用于交易的资金量(昨可用)  YdTradableFund[str]  // 昨日期初权益(昨结存)  YdInitialBalance[str]  // 昨日期末权益(昨权益)  YdFinalBalance[str]  // 浮动盈亏  ProfitLoss[str]  // 币种编号  CurrencyNo[str]  // 货币与基本的汇率  CurrencyRate[float]  // LME未到期平盈(未到期平盈) (股票中不使用)  LMEUnexpiredPL[float]  // LME未结平盈(未结平盈) (股票中不使用)  LMEUnaccountPL[float]  // 维持保证金  MaintenanceMargin[float]  // 期权权利金  Premium[float]  // 信任额度  CreditAmount[float]  // 初始资金  IntialFund[float]  // 资金帐号 1  FundAccountNo[str]  // 按揭价值  MortgageInMoney[str]  // 孖展上限额度  MarginLimit[float]  // 借货价值  BorrowInMoney[float]  // T1日交割的资金，正入负出  T1DeliveryMoney[float]  // T2日交割的资金，正入负出  T2DeliveryMoney[float]  // T3日交割的资金，正入负出  T3DeliveryMoney[float]  // TN日交割的资金，正入负出  TNDeliveryMoney[float]  //交易限额  TradeLimit[float]  // 可取资金==min(T0,T0+T1,T0+T1+T2)-冻结资金(挂单)  CanCashOutMoneyAmount[float]  // 月存款利息  DepositInterest[float]  // 月欠款利息  LoanInterest[float]  // 跨市场资金限额  CrossCurrencyMaxMoneyAmt[float]  // 卖空冻结资金  SellShortFrozenMoney[float]  // 卖空利息  SellShortInterest[float]  // 需补按金(卖空持仓时)  ShortPosAddtionalMargin[float]  // 错误信息  ErrorDescription[str] |

### 经纪商查询请求/返回reqQryBroker/onRspQryBroker

|  |
| --- |
|  |

### 版本号查询请求/返回reqQryVersion/onRspQryQryVersion

函数原型：

|  |
| --- |
| reqQryVersion(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户ID      UserId[str]      // 用户密码      UserPwd[str]      // 错误信息      ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryVersion(data: dict, error: dict, reqid: int, last: bool) |

参数：dat

|  |
| --- |
| // 版本号  Version[str]  // 是否必须更新到新版本才能正常使用（0：可以不更新；1：必须更新）  MustUpdate[str]  // 必须要更到的版本号  MustVersion[str]  // 版本内容中文  VersionContent\_CN[str]  // 版本内容英文  VersionContent\_US[str] |

参数：error

### 币种查询请求/返回reqQryCurrency/ onRspQryCurrency

函数原型：

|  |
| --- |
| reqQryCurrency(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 不使用  Unused[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryCurrency(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 货币编号  CurrencyNo[str]  // 基币货币编号  IsBase[int]  // 与基币的换算汇率  ChangeRate[float]  // 货币名称  CurrencyName[str]  // 货币名称（英文）  CurrencyNameEN[str] |

### 交易时间查询请求/返回reqQryTradeTime/ onRspQryTradeTime

函数原型：

|  |
| --- |
| reqQryTradeTime(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 不使用  Unused[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryTradeTime(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 当前交易日期（yyyyMMdd）      TradeDate[str]      // 开盘时间（HH:mm:ss）      OpenTime[str]      // 收盘时间（HH:mm:ss      CloseTime[str] |

### 交易所查询请求/返回reqQryExchange /onRspQryExchange

函数原型：

|  |
| --- |
| reqQryExchange(req: dict, reqid: int) |

参数：req

|  |
| --- |
| ProductGroupID[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryExchange(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 交易所编号  ExchangeNo[str]  // 交易所名称  ExchangeName[str]  // 交易所类型（HK：港股；US：美股）  SettleType[str]  // 交易所名称（英文）  NameEN[str] |

参数：error

### 合约查询请求/返回reqQryInstrument/onRspQryInstrument

函数原型：

|  |
| --- |
| reqQryInstrument(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 不使用  Unused[str]  // 查询的数量  Index[int]  // 修改时间，增量查询  ModifyDay[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryInstrument(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 交易所编号      ExchangeNo[str]      // 交易所名称      ExchangeName[str]      // 合约NO      CommodityNo[str]      // 商品名      CommodityName[str]      // 商品类别，    0：股票，1：窝轮及牛熊证，2：认股权，3：ETF，4：债券，5：信托，6：其他，8：指数      CommodityType[str]      // 货币编号      CurrencyNo[str]      // 货币名称      CurrencyName[str]  // 点值（一个最小跳点的价值）  ProductDot[float]  // 最小变动单位  UpperTick[float]  // 上日结算价  SettlePrice[float]  // 交易月 (yyyyMM)/交割日 (yyyyMMdd)  TradeMonth[str]  // 行情小数点位数  DotNum[int]  // 进阶单位  LowerTick[int]  // 调期小数点位数  DotNum\_Carry[int]  // 调期最小变动单位  UpperTick\_Carry[float]  // 首次通知日 (yyyyMMdd)  FirstNoticeDay[str]  // 冻结保证金百分比  FreezenPercent[float]  // 冻结保证金固定值  FreezenMoney[float]  // 固定手续费  FeeMoney[float]  // 百分比手续费  FeePercent[float]  // 现货商品昨结算价  PriceStrike[float]  // 现货商品点值  ProductDotStrike[float]  // 现货商品最小变动单位  UpperTickStrike[float]  // 最后交易日 (yyyyMMdd)(期货专用)  LastTradeDay[str]  // 最后更新日 (yyyyMMdd)(期货专用)  LastUpdateDay[str]  // 期权临界价格 (期货专用)  CriticalPrice[float]  // 期权临界价格以下的最小跳点 (期货专用)  CriticalMinChangedPrice[float]  // 实际交易所(CME细分成3个交易所:CME,CME\_COMEX,CME\_NYMEX)(期货专用)      Exchange2[str]      // 期权类型(R：看涨；F：看跌)(期货专用)      OptionType[str]      // 期权年月(yyyyMM)(期货专用)      OptionMonth[str]      // 期权执行价格(期货专用)      OptionStrikePrice[str]      // 期权对应期货商品编号(期货专用)      OptionCommodityNo[str]  // 期权对应期货合约编号(期货专用)  OptionContractNo[str]  // 按揭价值百分比      MortgagePercent[str]      // 最小变动单位代码      UpperTickCode[str]      // 最小批量      LotSize[str]      // 平仓时间标志(控制T+1：0代表T+0，可以当天卖出平仓；1代表T+1，必须隔天卖出平仓；N代表T+N)      FlatTime[str]      // 商品名（英文）      CommodityFNameEN[str]      // 是否支持卖空（0或空：不可以；1：可以）      CanSell[str]  // 卖空时的按金比例（例：0.5）  SellRate[float]  // 卖空持仓上限额  SellMax[float]  // 换股比率，表示一份窝轮牛熊証，相当于多少正股  StrikeRate[float]  // 行使价  StrikePrice[float]  // 收回价  ReceivePrice[float]  // 到期日（yyyy-MM-dd）  ExpireDate[str]  // 卖空时的维持按金比例（例：0.5）  SellRateKeep[float]  // 涡轮牛熊证对应正股编号  StrikeCommodityNo[str]  // 涡轮牛熊证的涨跌标志(C:涨；P:跌)  CallPutFlag[str]  // 涡轮牛熊证的发行人  Publisher[str] |

### 跳点查询请求/返回reqQryTick/ onRspQryTick

函数原型：

|  |
| --- |
| reqQryTick(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 不使用  Unused[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryTick(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 最小变动单位代码      UpperTickCode[str]      // 最小变动单位开始行情价格      PriceFrom[str]      // 行情最小变动单位      UpperTick[str]      // 最小变动单位对应的点值      ProductDot[str]      // 行情小数位数      DotNum[str]      // 进价单位      LowerTick[str] |

### 订单类型查询请求/返回reqQryOrderType/ onRspQryOrderType

|  |
| --- |
|  |

### 设置双重认证密码/返回reqSetVerifyQA/ onRspSetVerifyQA

函数原型：

|  |
| --- |
| reqSetVerifyQA(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户帐号  UserId[str]  // 交易密码  UserPwd[str]  // 固定值，“1”  Type[str]  // 固定值，“1”  Question[str]  // 双重认证密码，英数字  Answer[str]  // 固定值，“1”  SaveMac[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspSetVerifyQA(error: dict, reqid: int, last: bool) |

参数：error

回调函数错误码指示成功时，调用reqSafeVerify()函数绑定网卡地址。

### 绑定双重认证需要的MAC地址/返回reqSafeVerify/ onRspSafeVerify

函数原型：

|  |
| --- |
| reqSafeVerify(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户帐号  UserId[str]  // 交易密码  UserPwd[str]  // 固定值，“1”  Type[str]  // 固定值，“1”  Question[str]  // 双重认证密码，英数字  Answer[str]  // 固定值，“1”  SaveMac[str]  // MAC地址  MacAddress[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspSafeVerify(error: dict, reqid: int, last: bool) |

参数：error

# 期货交易API接口参考

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 业务类型 | 业务 | 请求接口 |  | 响应接口 |
| 连接 | 连接 | init | 连接回报 | onFrontConnected |
| 断开 |  |  | 连接断开回报 | onFrontDisconnected |
| 心跳 |  |  | 心跳警告 | onHeartBeatWarning |
| 登录 | 登录 | reqUserLogin | 登录回报 | onRspUserLogin/OnRspNeedVerify |
| 登出 | reqUserLogout | 登出回报 | onRspUserLogout |
| 修改密码 | reqUserPasswordUpdate | 修改密码回报 | onRspUserPasswordUpdate |
| 双重认证step1 | reqSetVerifyQA | 设置双重认证密码 | onRspSetVerifyQA |
| 双重认证step2 | reqSafeVerify | 绑定MAC地址 | onRspSafeVerify |
| 交易 | 报单 | reqOrderInsert | 报单回报 | onRspOrderInsert |
| 改单 | reqOrderModify | 改单回报 | onRspOrderModify |
| 撤单 | reqOrderCancel | 撤单回报 | onRspOrderCancel |
| 推送 |  |  | 报单状态推送 | onRtnOrder |
|  |  | 成交信息推送 | onRtnTrade |
|  |  | *持仓信息推送* | *onRtnPosition* |
|  |  | 资金信息推送 | onRtnAccount |
| 查询 | 委托单查询 | reqQryOrder | 委托单查询回报 | onRspQryOrder |
| 成交查询 | reqQryTrade | 成交查询回报 | onRspQryTrade |
| 持仓明细查询 | reqQryPosition | 持仓明细查询回报 | onRspQryPosition |
| 持仓总结查询 | reqQryTotalPosition | 持仓总结查询回报 | onRspQryTotalPosition |
| 资金查询 | reqQryCapital | 资金查询回报 | onRspQryCapital |
| 版本号查询 | reqQryVersion | 版本号查询回报 | onRspQryQryVersion |
| 币种查询 | reqQryCurrency | 币种查询回报 | onRspQryCurrency |
| 交易时间查询 | reqQryTradeTime | 交易时间查询回报 | onRspQryTradeTime |
| 策略查询 | reqQryStrategy | 策略查询回报 | onRspQryStrategy |
| 交易所查询 | reqQryExchange | 交易所查询回报 | onRspQryExchange |
| 合约查询 | reqQryInstrument | 合约查询回报 | onRspQryInstrument |
| 策略明细查询 | reqQryStrategyDetail | 策略明细查询回报 | onRspQryStrategyDetail |
|  | 交易所冬夏令开盘时间查询(from v1.10) | reqQryExchangeTime | 查询回报 | onRspQryExchangeTime |
|  | 交易品种开收盘时间查询(from v1.10) | reqQryCommodityTime | 查询回报 | onRspQryCommodityTime |
|  | 查询品种列表(from v1.10) | reqQryCommodityTime | 查询回报 | onRspQryCommodity |

注：以上斜体字的接口，当前API版本未实现。

## 接口工作过程



## 接口模式

future\_demo.py文件中提供了一个FutureApi类，通过继承FutureApi类可以发出操作请求，也可以重载回调函数来处理后台服务的响应。

。

## 函数接口说明

### createFutureApi方法

创建实例方法。

**函数原形**：

createFutureApi(bRecordLog: bool, lpszLogFileName: str = "Future.log", lpszLogFileDir: str = "");

bRecordLog: true表示记录日志，false表示不记录日志。

lpszLogFileName：日志文件。

lpszLogFileDir： 日志文件文件夹路径。

### onFrontConnected 方法

当客户端与前置建立起通信连接时（还未登录前），该方法被调用。

**函数原形：**

onFrontConnected()

本方法在完成初始化后调用，可以在其中完成用户登录任务。

### onFrontDisconnected方法

当客户端与前置通信连接断开时，该方法被调用。当发生这个情况

后，API会自动重新连接，客户端可不做处理。自动重连地址，可能是原来注册

的地址，也可能是系统支持的其它可用的通信地址，它由程序自动选择。

**函数原形：**

onFrontDisconnected (int nReason)

**参数：**

nReason：连接断开原因

### onHeartBeatWarning方法

心跳超时警告。当长时间未收到报文时，该方法被调用。

**函数原形：**

onHeartBeatWarning(nTimeLapse: int)

**参数：**

nTimeLapse：距离上次接收报文的时间

### 登录reqUserLogin/onRspUserLogin/onRspNeedVerify/onRspAccount

从API v1.5引入双重认证，具体交互过程请参阅**“国际期货股票API\_双重认证交互说明.xlsx”**

函数原型：

|  |
| --- |
| reqUserLogin(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户ID  UserId[str]  // 用户密码  UserPwd[str]  // 用户类型  UserType[str]  // 软件名称  SoftwareName[str]  // 软件版本号  SoftwareVersion[str]  // 授权码  AuthorCode[str]  //登录机器网卡MAC地址(since API v1.5,双重认证使用)  MacAddress[str]  // 错误信息  TDAStringType[str] |

1) 函数原型：

|  |
| --- |
| onRspUserLogin(error: dict, reqid: int, last: bool) |

参数：error

错误码：

|  |  |
| --- | --- |
| **登录错误码** | **错误内容** |
| 10001 | 用户名不正确 |
| 10002 | 登录密码错误 |
| 10003 | 密码错误次数超限，用户已冻结，请联系客服解冻！ |
| 10004 | 用户已被冻结，请联系客服解冻！ |
| 10005 | 授权码信息不正确，请确认UserType，SoftwareName，SoftwareVersion，AuthorCode的值 |

2) 函数原型：

服务器端在判断到需要做双重认证时，会指示API回调此函数

|  |
| --- |
| onRspNeedVerify(bFirstLogin: bool, bHasSetQA: bool); |

参数：bFirstLogin

|  |
| --- |
| 当设备(根据login消息中的MacAddress字段识别)第一次登录交易前置时， onRspNeedVerify()函数被回调，且bFirstLogin 的值为true,用户在回调函数中需调用reqSetVerifyQA()函数设置双重认证信息。  当设备(根据login消息中的MacAddress字段识别)第一次以后登录交易前置时，onRspNeedVerify()函数不会被回调。 |

1. 函数原型：

在onRspUserLogin指示成功登录回调后，该函数被回调，data字典中返回每个币种对应的资金账号

|  |
| --- |
| onRspAccount(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 帐户币种  CurrencyNo[str]  // 资金账号  AccountNo[str] |

### 登出 reqUserLogout/onRspUserLogout方法

函数原型：

|  |
| --- |
| reqUserLogout(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户ID  UserId[str]  // 资金账号  AccountNo[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspUserLogout(error: dict, reqid:int, last: bool) |

参数：error

### 修改密码reqPasswordUpdate/onRspPasswordUpdate

函数原型：

|  |
| --- |
| reqPasswordUpdate (req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户代码  UserId[str]  // 用户原密码  OldPassword[str]  // 用户新密码  NewPassword[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspPasswordUpdate (data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 用户代码  UserId[str]  // 用户原密码  OldPassword[str]  // 用户新密码  NewPassword[str] |

参数：error

### 报单reqOrderInsert/onRspOrderInsert

说明：

SystemNo[str]: 系统编号

LocalNo[str] : 本地编号

OrderNo[str]: 定单号

LocalNo是使用API的程序给的订单号,相当于CTP里面的orderref，当日内最好不要重复

SystemNo是直达系统给的订单号；

OrderNo是交易所给的订单号，相当于CTP里面的OrderSysID

函数原型：

|  |
| --- |
| reqOrderInsert(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户ID  UserId[str]  // 资金账号(一个用户可以有多个币种的资金，所以会有多个资金账号，下单需指定)  // 请参照登录成功后的回调函数onRspAccount的说明  AccountNo[str]  // 本地编号  LocalNo[str]  // 交易所代码  ExchangeCode[str]  // 合约代码  ContractCode[str]  // 买还是卖：1=buy 2=sell  BidAskFlag[str]  // 开仓还是平仓：1=开仓 2=平仓，3=平今  OpenCloseFlag[str]  // 下单数  OrderQty[str]  // 下单价格  OrderPrice[str]  // 定单类型：1=限价单, 2=市价单，3=限价止损（stop to limit），4=止损（stop to market)  OrderType[str]  // 触发价格  TriggerPrice[str]  // 有效日期（1=当日有效, 2=永久有效（GTC），4=IOC）  TIF[str]  // 策略ID  StrategyId[str]  //冰山单使用：显示委托量 必须小于委托量  MaxShow[str]  //IOC单使用： 最小成交量  //CME交易所：有效日期=4IOC时, MinQty >=1且小于委托量时是FAK，MinQty=委托量时是FOK  //港交所，新交所：MinQty =1时FAK单，MinQty=委托量时FOK单，不能为其它值  MinQty[str]  // Tag50(订单受益人/所有者)  Tag50[str]  // 错误信息  ErrorDescription[str] |
| // 程序化单(0)或人工单(1)  IsProgram[int]  // (\*4) ISO(3166-1)标准中定义的国家/地区两个字母代码  OrgOrderLocationID[str]  // 合约类型(跨期时候必填'1'-DERIVATIVE\_CONTRACT\_CATEGORY\_SPREAD, 默认其他类型非必填）  ContractCategory[str] |

说明：

1：API用户下单后，onRspOrderInsert函数会被回调两次：

1. 第一次回调返回直达系统给这笔订单指定的系统号SystemNo，表明直达系统收到并确认这笔订单。
2. 第二次回调返回交易所的反馈的订单号OrderNo(如果一切正常)。通常资金不足等情况发生时，不会有订单号，因为订单指令没有向交易所发送。

2：tag50(订单操作人/所有者)字段说明，下改撤单中tag50类同

1. tag50字段适用于cme旗下交易所,非CME交易所的下单无需设置tag50.(关于tag50下单规则方面有任何疑问,欢迎联系经纪商业务人员咨询.)
2. 如客户长时间交易，例如超过12小时，需设置多轮管理、操作和监控（至少设置2 个班次）。当负责管理、操作和监控的人员发生变化时，新一班次人员需分配不同于前一班次的Tag50，并在订单中传输。
3. Tag50(订单操作人/所有者) 生产环境中，分两种情况：
   1. API不设置或空值的情况下，默认使用客户号作为Tag50（客户号作为Tag50无需向经纪公司申报）。
   2. 上述a)以外，须提前向经纪公司申报Tag50信息，由经纪公司设置Tag50之后，才可在API中使用；API使用任何未经申报的Tag50下单将会被拒绝。客户需确保实际下单传输Tag50与API申请时申报Tag50信息一致。  
      Tag50值的分配逻辑是客户号+‘\_001‘，客户号+’\_002’，依次类推；

3: IsProgram：区分手工单或者程序化单。

API中默认为程序化单，客户需按照实际情况设置0（程序化单）或1（手工单）。

4: OrgOrderLocationID

只有该API账号下还有下层多个不同客户交易时(API程序用作订单路由网关)，才需要填写，分两种填写方式：

国际/地区代码为加拿大时，后面紧跟逗号，再跟省代码，总共5个ASCII字符。

例：加南大魁北克省对应CA,QC

国际/地区地区代码为加拿大以外时，仅填写国际/地区代码，总共2个ASCII字符。

例：Japan对应JP, China对应CN

函数原型：

|  |
| --- |
| onRspOrderInsert(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 用户ID  UserId[str]  // 资金账号  AccountNo[str]  // 系统编号  SystemNo[str]  // 本地编号  LocalNo[str]  // 定单号  OrderNo[str]  // 交易所代码  ExchangeCode[str]  // 合约代码  ContractCode[str]  // 买还是卖：1=buy 2=sell  BidAskFlag[str]  // 下单数  OrderQty[str]  // 下单价格  OrderPrice[str]  // 定单类型：1=限价单, 2=市价单，3=限价止损（stop to limit）  OrderType[str]  // 下单日期  OrderDate[str]  // 下单时间  OrderTime[str]  // 错误代码(保留，未使用)  ErrorCode[str]  // 订单状态（2：已排队；3：部分成交；4：完全成交；5：已撤余单；6：已撤单；7：指令失败）  OrderState[str]  // 触发价格  TriggerPrice[str]  // 有效日期（1=当日有效, 2=永久有效（GTC），4=IOC）  TIF[str]  // 开仓还是平仓：1=开仓 2=平仓，3=平今，4=平昨  OpenCloseFlag[str]  // 策略ID  StrategyId[str]  // 冰山单：显示委托量 必须小于委托量  MaxShow[str]  // 最小成交量  MinQty[str]  // 交易所返回时间  ExchangeTime[str]  // 终端类型(CME交易所特定)  OrdSourceType[str]  // Tag50(订单受益人/所有者),格式为“机构通ID\_子账户”  Tag50[str] |

参数：error

|  |
| --- |
| // 错误码(用来判断报单成功或失败)  ErrorID[int]  // 错误描述  ErrorMsg[str] |

错误码

|  |  |
| --- | --- |
| **系统号错误码** | **错误内容** |
| 20005 | 系统号生成失败 |

下单错误码

|  |  |
| --- | --- |
| **下单错误码** | **错误内容** |
| 20000 | 下单失败 |
| 20001 | 资金不足 |
| 20002 | 交易服务器未连接 |
| 20003 | 您已被禁止交易，请联系客服解禁 |
| 20004 | 下单被拒绝 |
| 20005 | 系统号生成失败 |
| 20006 | 您的资金账户中没有该合约交易所需的币种 |
| 20007 | 该合约已经到期，不能交易 |
| 20008 | 该交易市场未开市，不能交易 |
| 20009 | 该交易市场已闭市，不能交易 |
| 20010 | 下单价格超出限定范围 |
| 20011 | 下单数量过大 |
| 20012 | 您被禁止交易该合约，请联系客服 |
| 20013 | 您的资金账户风险率过高，已被禁止交易 |
| 20015 | 您的账户交易数据出现异常，暂停交易 |
| 20019 | 下单数量超过客户持仓限量 |
| 20020 | 下单数量超过客户可下单买量 |
| 20021 | 下单数量超过客户可下单卖量 |
| 20022 | 下单数量超过公司持仓限量 |
| 20023 | 下单数量超过公司可下单买量 |
| 20024 | 下单数量超过公司可下单卖量 |
| 20025 | Dealer下单数量超过最大可下单数量 |
| 20026 | 合约临近交割，只可平仓，不能开仓 |
| 20027 | 该合约已过最后交易日时间，禁止电子盘交易 |
| 20028 | 商品未开放，禁止交易 |
| 20029 | 单笔下单数量超过该商品单笔最大可下单数 |
| 20032 | 持仓不足，平仓失败 |
| 20033 | 您不能交易该商品,请联系客服开通 |

### 订单推送onRtnOrder

函数原型：

|  |
| --- |
| onRtnOrder(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 交易所代码  ExchangeNo[str]  // 合约代码  TreatyCode[str]  // 定单号  OrderNo[str]  // 委托数量  OrderNumber[int]  // 已成交数量  FilledNumber[int]  // 成交均价  FilledAdvPrice[float]  // 持买数量  BuyHoldNumber[int]  // 持买开仓均价  BuyHoldOpenPrice[float]  // 持买均价  BuyHoldPrice[float]  // 持卖数量  SaleHoldNumber[int]  // 持卖开仓均价  SaleHoldOpenPrice[float]  // 持卖均价  SaleHoldPrice[float]  // 是否已经撤单（0：没有；1：已撤单）  TDAStringType IsCanceled[float]  // 成交总的手续费  FilledTotalFee[float]  // 顺序号  Status[int]  // 资金帐号  AccountNo[str]  // 持仓类型（0：昨仓；1：今仓）  HoldType[str]  // 持买保证金  HoldMarginBuy[float]  // 持卖保证金  HoldMarginSale[float]  // 最新价  CurrPrice[float]  // 浮动盈亏  FloatProfit[float] |

参数：error

### 成交推送onRtnTrade

函数原型：

|  |
| --- |
| onRtnTrade(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 用户ID  UserId[str]  // 资金账号  AccountNo[str]  // 成交编号（要包括7位的订单编号，一共11位）  FilledNo[str]  // 定单号  OrderNo[str]  // 系统编号  SystemNo[str]  // 本地编号  LocalNo[str]  // 交易所代码  ExchangeCode[str]  // 合约代码  TreatyCode[str]  // 买还是卖：1=buy 2=sell  BuySale[str]  // 成交数  FilledNumber[str]  // 成交价格  FilledPrice[str]  // 成交日期(yyyy-MM-dd)  FilledDate[str]  // 成交时间(hh:mm:ss)  FilledTime[str]  // 成交手续费  Commsion[str]  // 委托数量  OrderNumber[str]  // 委托价格  OrderPrice[str]  // 合约交割日期(yyyyMMdd)  DeliveryDate[str]  // 成交类别(N：普通下单成交；C、T：调期模拟成交)  FilledType[str]  // 定单类型（1=限价单, 2=市价单，3=限价止损（stop to limit），4=止损（stop to market）  OrderType[str]  // 有效日期（1=当日有效, 2=永久有效）  ValidDate[str]  // 开仓还是平仓：1=开仓 2=平仓，3=平今，4=平昨  AddReduce[str] |

### 持仓信息推送onRtnPosition

### 资金信息推送onRtnCapital

函数原型：

|  |
| --- |
| onRtnCapital(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 客户号  ClientNo[str]  // 资金帐号  AccountNo[str]  //币种  CurrencyNo[str]  //今可用  Available[float]  //昨可用  YAvailable[float]  //今可出  CanCashOut[float]  //今结存  Money[float]  //平仓盈亏  ExpiredProfit[float]  //冻结资金  FrozenDeposit[float]  //手续费  Fee[float]  //保证金  Deposit[float]  //维持保证金  KeepDeposit[float]  //状态  Status[int]  //入金  InMoney[float]  //出金  OutMoney[float]  //未到期平盈  UnexpiredProfit[float]  //今权益  TodayTotal[float]  //未结平盈  UnaccountProfit[float]  //期权权利金  Royalty[float]  //交易所代码  ExchangeNo[str]  //合约代码  TreatyCode[str]  //定单号  OrderNo[str]  //委托数量  OrderNumber[int]  //已成交数量  FilledNumber[int]  //成交均价  FilledAdvPrice[float]  //持买数量  BuyHoldNumber[int]  //持买开仓均价  BuyHoldOpenPrice[float]  //持买均价  BuyHoldPrice[float]  //持卖数量  SaleHoldNumber[int]  //持卖开仓均价  SaleHoldOpenPrice[float]  //持卖均价  SaleHoldPrice[float]  //是否已经撤单（0：没有；1：已撤单）  IsCanceled[str]  //成交总的手续费  FilledTotalFee[float]  //信任额度  Credit[float]  //孖展上限额度 for 港股  MarginLimit[float]  //借货价值 for 港股  BorrowValue[float]  //按揭价值 for 港股  FMortgageMoney[float]  // T1 for 港股  T1[float]  // T2 for 港股  T2[float]  //T3 for 港股  T3[float]  // Tn for 港股  TN[float]  //交易限额  TradeLimit[float]  //跨市场资金限额  FCrossMax[float]  //卖空冻结资金  SellFreezenMoney[float]  //卖空利息  SellInterest[float]  //需补按金  SellNeedAddMargin[float] |

注意：回调数据字典中有与股票相关的字段，使用时请无视。

### 改单请求/返回reqOrderModify/onRspOrderModify方法

函数原型：

|  |
| --- |
| reqOrderModify(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 系统编号  SystemNo[str]  // 用户ID  UserId[str]  // 本地编号  LocalNo[str]  // 资金账号  AccountNo[str]  // 定单号  OrderNo[str]  // 交易所代码  ExchangeCode[str]  // 合约代码  ContractCode[str]  // 买还是卖：1=buy 2=sell  BidAskFlag[str]  // 下单数  OrderQty[str]  // 下单价格  OrderPrice[str]  // 改单数  ModifyQty[str]  // 改单价格  ModifyPrice[str]  // 价格类型：1=限价单, 2=市价单，3=限价止损（stop to limit），4=止损（stop to market）  OrderType[str]  // 触发价格  TriggerPrice[str]  // 改单触发价格  ModifyTriggerPrice[str]  // 有效日期（1：当日有效；2：永久有效）  TIF[str]  //下单人ID  Tag50[str]  // 错误信息  ErrorDescription[str]  // ISO(3166-1)标准中定义的国家/地区两个字母代码  OrgOrderLocationID[str] |

函数原型：

|  |
| --- |
| onRspOrderModify(data: dict, error: dict, reqid: int, last: bool) |

参数：error

|  |
| --- |
| // 错误码(用来判断改单成功或失败)  ErrorID[int]  // 错误描述  ErrorMsg[str] |

### 撤单请求/返回reqOrderCancel/onRspOrderCancel

函数原型：

|  |
| --- |
| reqOrderCancel(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户ID  UserId[str]  // 用户类型：1：一般用户；2：机构通用户；  UserType[str]  // 本地编号  LocalNo[str]  // 资金账号  AccountNo[str]  // 系统编号  SystemNo[str]  // 定单号  OrderNo[str]  // 交易所代码  ExchangeCode[str]  // 合约代码  ContractCode[str]  // 买还是卖：1=buy 2=sell  BidAskFlag[str]  // 下单数  OrderQty[str]  // 下单价格  OrderPrice[str]  // 价格类型：1=limit order, 2=market order  OrderType[str]  // 错误信息  ErrorDescription[str]  // ISO(3166-1)标准中定义的国家/地区两个字母代码  OrgOrderLocationID[str]  // 合约类型(跨期时候必填'1'-DERIVATIVE\_CONTRACT\_CATEGORY\_SPREAD, 默认其他类型非必填）  ContractCategory[str] |

函数原型：

|  |
| --- |
| onRspOrderCancel data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 用户ID  UserId[str]  // 资金账号  AccountNo[str]  // 系统编号  SystemNo[str]  // 本地号  LocalNo[str]  // 定单号  OrderNo[str]  // 撤单号  CancelNo[str]  // 交易所代码  ExchangeCode[str]  // 合约代码  ContractCode[str]  // 买还是卖：1=buy 2=sell  BidAskFlag[str]  // 下单数  OrderQty[str]  // 下单价格  OrderPrice[str]  // 已成交数  FilledQty[str]  // 已撤单数  CancelledQty[str]  // 价格类型：1=limit order, 2=market order  OrderType[str]  // 撤单日期  CancelledDate[str]  // 撤单时间  CancelledTime[str]  // 错误代码(保留，未使用)  ErrorCode[str] |

参数：error

|  |
| --- |
| // 错误码(用来判断撤单成功或失败)  ErrorID[int]  // 错误描述  ErrorMsg[str] |

### 委托单查询/返回reqQryOrder/onRspQryOrder

两个委托单查询之间最小间隔时间为1秒，否则查询失败。(Since API v1.5)

函数原型：

|  |
| --- |
| reqQryOrder( req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户ID  UserId[str]  // 资金账号  AccountNo[str]  // 是否模拟用户：1：是；0 or other：不是  IsSimulation[str]  // 取得指定订单号以后的定单  OrderNo[str]  // 取得指定订单时间以后的定单（格式：yyyy-MM-dd hh:mm:ss）  OrderDateTime[str]  // 错误信息  ErrorDescription[str]  // 订单过滤条件 (Since API v1.18.2.2)  //值: DERIVATIVE\_ORD\_FILTER\_L，返回存活订单(未撤销等待成交的订单) //DERIVATIVE\_ORD\_FILTER\_A(缺省值)，返回所有订单，包括已全部成交和已撤订单  FilterFlag[str] |
|  |

函数原型：

|  |
| --- |
| onRspQryOrder(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 用户ID  UserId[str]  // 资金账号  AccountNo[str]  // 系统编号  SystemNo[str]  // 本地编号  LocalNo[str]  // 定单号  OrderNo[str]  // 交易所代码  ExchangeCode[str]  // 合约代码  ContractCode[str]  // 买还是卖：1=buy 2=sell  BidAskFlag[str]  // 下单数  OrderQty[str]  // 下单价格  OrderPrice[str]  // 已成交数  FilledQty[str]  // 成交均价  FilledPrice[str]  // 定单类型：1=限价单, 2=市价单，3=限价止损（stop to limit），4=止损（stop to market）  OrderType[str]  // 下单日期  OrderDate[str]  // 下单时间  OrderTime[str]  // 错误代码  ErrorCode[str]  // 订单状态（1：已请求；2：已排队；3：部分成交；4：完全成交；5：已撤余单；6：已撤单；7：指令失败；）  OrderState[str]  // 撤单的用户ID  CancelUserId[str]  // 触发价格  TriggerPrice[str]  // 有效日期（1=当日有效, 2=永久有效（GTC），4=IOC）  TIF[str]  // 开仓还是平仓：1=开仓 2=平仓，3=平今，4=平昨  OpenCloseFlag[str]  // 策略ID  StrategyId[str]  // 显示委托量 必须小于委托量  MaxShow[str]  // 最小成交量  MinQty[str]  // 交易所返回时间  ExchangeTime[str]  // 撤单时间  CancelTime[str] |

参数：error

### 成交查询请求/返回reqQryTrade/onRspQryTrade

两个成交查询之间最小间隔时间为1秒，否则查询失败。(Since API v1.5)

函数原型：

|  |
| --- |
| reqQryTrade(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户  UserId[str]  // 错误信息  ErrorDescription[str] |

由于成交数据可能很多，增加分页查询方式。具体参照demo中的代码。

(Since API v1.15)

函数原型：

|  |
| --- |
| onRspQryTrade(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 用户ID  UserId[str]  // 资金账号  AccountNo[str]  // 成交编号（要包括7位的订单编号，一共11位）  FilledNo[str]  // 定单号  OrderNo[str]  // 系统编号  SystemNo[str]  // 本地编号  LocalNo[str]  // 交易所代码  ExchangeCode[str]  // 合约代码  TreatyCode[str]  // 买还是卖：1=buy 2=sell  BuySale[str]  // 成交数  FilledNumber[str]  // 成交价格  FilledPrice[str]  // 成交日期(yyyy-MM-dd)  FilledDate[str]  // 成交时间(hh:mm:ss)  FilledTime[str]  // 成交手续费  Commsion[str]  // 委托数量  OrderNumber[str]  // 委托价格  OrderPrice[str]  // 合约交割日期(yyyyMMdd)  DeliveryDate[str]  // 成交类别(N：普通下单成交；C、T：调期模拟成交)  FilledType[str]  // 定单类型（1=限价单, 2=市价单，3=限价止损（stop to limit），4=止损（stop to market）  OrderType[str]  // 有效日期（1=当日有效, 2=永久有效）  ValidDate[str]  // 开仓还是平仓：1=开仓 2=平仓，3=平今，4=平昨  AddReduce[str]  // 错误信息  ErrorDescription[str] |

参数：error

|  |
| --- |
| // 错误码  ErrorID[int]  // 错误描述  ErrorMsg[str] |

### 持仓明细查询请求/返回reqQryPosition/onRspQryPosition

两个持仓明细查询之间最小间隔时间为1秒，否则查询失败。(Since API v1.5)

函数原型：

|  |
| --- |
| reqQryPosition(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryPosition(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 用户ID  ClientNo[str]  // 交易所  ExchangeNo[str]  // 证券代码  CommodityNo[str]  // 持仓方向（1：持买；2：持卖）  Direct[str]  // 持仓成本价  HoldPrice[float]  // 可卖数量  CanTradeVol[int]  // 今买数量  TodayBuyVol[int]  // 冻结数量  FrozenVol[int]  // 持有期内买入金额总和  TotalBuyMoney[float]  // 持有期内卖出金额总和  TotalSellMoney[float]  // 持有期内买入数量  TotalBuyVol[int]  // 持有期内卖出数量  TotalSellVol[int]  // 首次开仓日期(yyyy-MM-dd)  OpenDate[str]  // 平仓盈利  FlatProfit[float]  // 港股T+1数量  HkexT1[int]  // 港股T+2数量  HkexT2[int]  // 美股T+3数量  HkexT3[int]  // 港股未交收数量=T1+T2  UnsettleVol[int]  // 港股已交收数量  SettledVol[int]  // 持仓数量  HoldVol[int]  // 今卖数量  TodaySaleVol[int]  // 卖空冻结资金  SellFrozenMoney[float]  // 开仓均价  OpenPrice[float] |

参数：error

### 资金查询请求/返回reqQryCapital/onRspQryCapital

两个资金查询之间最小间隔时间为1秒，否则查询失败。(Since API v1.5)

函数原型：

|  |
| --- |
| reqQryCapital( req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 不使用  Unused[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryCapital(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 用户ID  UserId[str]  // 入金  Deposit[str]  // 出金  Withdraw[str]  // 今日可用于交易的资金量(今可用)  TodayTradableFund[str]  // 当日期初权益(今结存)  TodayInitialBalance[str]  // 当日实时浮动权益(今权益)  TodayRealtimeBalance[str]  // 冻结资金  FrozenFund[str]  // 佣金  Commission[str]  // 初始保证金  InitialMargin[str]  // 昨日可用于交易的资金量(昨可用)  YdTradableFund[str]  // 昨日期初权益(昨结存)  YdInitialBalance[str]  // 昨日期末权益(昨权益)  YdFinalBalance[str]  // 浮动盈亏  ProfitLoss[str]  // 币种编号  CurrencyNo[str]  // 货币与基本的汇率  CurrencyRate[float]  // LME未到期平盈(未到期平盈)  LMEUnexpiredPL[float]  // LME未结平盈(未结平盈)  LMEUnaccountPL[float]  // 维持保证金  MaintenanceMargin[float]  // 期权权利金  Premium[float]  // 信任额度  CreditAmount[float]  // 初始资金  IntialFund[float]  // 资金帐号 1  FundAccountNo[str]  // 交易限额  TradeLimit[float]  // 可取资金==min(T+0,T+0+T+1,T+0+T+1+T+2)-冻结资金(挂单)  CanCashOutMoneyAmount[float]  // 月存款利息  DepositInterest[float]  // 月欠款利息  LoanInterest[float]  // 错误信息  ErrorDescription[str] |

参数：error

### 版本号查询请求/返回reqQryVersion/onRspQryVersion

函数原型：

|  |
| --- |
| reqQryVersion(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户ID  UserId[str]  // 用户密码  UserPwd[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryVersion(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 版本号  Version[str]  // 是否必须更新到新版本才能正常使用（0：可以不更新；1：必须更新）  MustUpdate[str]  // 必须要更到的版本号  MustVersion[str]  // 版本内容中文  VersionContent\_CN[str]  // 版本内容英文  VersionContent\_US[str] |

参数：error

### 币种查询请求/返回reqQryCurrency/ onRspQryCurrency

函数原型：

|  |
| --- |
| reqQryCurrency(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 不使用  Unused[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryCurrency(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 货币编号  CurrencyNo[str]  // 基币货币编号  IsBase[int]  // 与基币的换算汇率  ChangeRate[float]  // 货币名称  CurrencyName[str]  // 货币名称（英文）  CurrencyNameEN[str] |

参数：error

### 交易时间查询请求/返回reqQryTradeTime/ onRspQryTradeTime

函数原型：

|  |
| --- |
| reqQryTradeTime(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 不使用  Unused[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryTradeTime(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 当前交易日期（yyyyMMdd）  TradeDate[str]  // 开盘时间（HH:mm:ss）  OpenTime[str]  // 收盘时间（HH:mm:ss  CloseTime[str] |

参数：error

### 交易所查询请求/返回reqQryExchange /onRspQryExchange

函数原型：

|  |
| --- |
| reqQryExchange(req: dict, reqid: int) |

参数：data

|  |
| --- |
| ProductGroupID[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryExchange(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 交易所编号  ExchangeNo[str]  // 交易所名称  ExchangeName[str]  // 交易所类型（HK：港股；US：美股）  SettleType[str]  // 交易所名称（英文）  NameEN[str] |

参数：error

### 合约查询请求/返回reqQryInstrument/onRspQryInstrument

函数原型：

|  |
| --- |
| reqQryInstrument(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 查询多少条,每次返回的条数  PageIndex[int]  // 交易所代码，如果填写值，查询一个交易所的合约  ExchangeNo[str]  // 合约代码。查询单个合约  CommodityNo[str]  // 商品类别  CommodityType[str]  // 合约NO  ContractNo[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryInstrument(data: dict, error: dict, reqid: int, last: bool) |

参数：CFutureRspInstrumentField

|  |
| --- |
| // 交易所编号  ExchangeNo[str]  // 交易所名称  ExchangeName[str]  // 合约NO  CommodityNo[str]  // 商品名  CommodityName[str]  // 商品类别，O：期权，F：期货，S：同商品跨期套利  CommodityType[str]  // 货币编号  CurrencyNo[str]  // 货币名称  CurrencyName[str]  // 点值（一个最小跳点的价值）  ProductDot[float]  // 最小变动单位  UpperTick[float]  // 上日结算价  SettlePrice[float]  // 交易月 (yyyyMM)/交割日 (yyyyMMdd)  TradeMonth[str]  // 行情小数点位数  DotNum[int]  // 进阶单位  LowerTick[int]  // 调期小数点位数  DotNum\_Carry[int]  // 调期最小变动单位  UpperTick\_Carry[float]  // 首次通知日 (yyyyMMdd)  FirstNoticeDay[str]  // 冻结保证金百分比  FreezenPercent[float]  // 冻结保证金固定值  FreezenMoney[float]  // 固定手续费  FeeMoney[float]  // 百分比手续费  FeePercent[float]  // 现货商品昨结算价  PriceStrike[float]  // 现货商品点值  ProductDotStrike[float]  // 现货商品最小变动单位  UpperTickStrike[float]  // 最后交易日 (yyyyMMdd)(期货专用)  LastTradeDay[str]  // 最后更新日 (yyyyMMdd)(期货专用)  LastUpdateDay[str]  // 期权临界价格 (期货专用)  CriticalPrice[float]  // 期权临界价格以下的最小跳点 (期货专用)  CriticalMinChangedPrice[float]  // 实际交易所(CME细分成3个交易所:CME,CME\_COMEX,CME\_NYMEX)(期货专用)  Exchange2[str]  // 期权类型(R：看涨；F：看跌)(期货专用)  OptionType[str]  // 期权年月(yyyyMM)(期货专用)  OptionMonth[str]  // 期权执行价格(期货专用)  OptionStrikePrice[str]  // 期权对应期货商品编号(期货专用)  OptionCommodityNo[str]  // 期权对应期货合约编号(期货专用)  OptionContractNo[str]  // 按揭价值百分比  MortgagePercent[str]  // 最小变动单位代码  UpperTickCode[str]  // 最小批量  LotSize[str]  // 平仓时间标志(控制T+1：0代表T+0，可以当天卖出平仓；1代表T+1，必须隔天卖出平仓；N代表T+N)  FlatTime[str]  // 商品名（英文）  CommodityFNameEN[str]  // 是否支持卖空（0或空：不可以；1：可以）  CanSell[str]  // 卖空时的按金比例（例：0.5）  SellRate[float]  // 卖空持仓上限额  SellMax[float]  // 换股比率，表示一份窝轮牛熊証，相当于多少正股  StrikeRate[float]  // 行使价  StrikePrice[float]  // 收回价  ReceivePrice[float]  // 到期日（yyyy-MM-dd）  ExpireDate[str]  // 卖空时的维持按金比例（例：0.5）  SellRateKeep[float]  // 涡轮牛熊证对应正股编号  StrikeCommodityNo[str]  // 涡轮牛熊证的涨跌标志(C:涨；P:跌)  CallPutFlag[str]  // 涡轮牛熊证的发行人  Publisher[str] |

### 策略查询请求/返回reqQryStrategy/ onRspQryStrategy

函数原型：

|  |
| --- |
| reqQryStrategy(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 合约代码。查询单个合约  InstrumentID[str]  // 交易所代码，如果填写值，查询一个交易所的合约  ExchangeID[str]  // 合约在交易所的代码  ExchangeInstID[str]  // 产品代码 ，如果填写值，查询一个产品的合约  ProductID[str]  // 开始时间,如果填写，是这个时间以后新增的  InsertTimeStart[str]  // 查询多少条,每次返回的条数  Index[int]  // 查询合约的类别 直达  ProductType[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryStrategy(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 策略所属用户ID  UserId[str]  // 策略Id  FKeyId[str]  // 策略名称  FName[str]  // 策略代码  FCode[str]  // 策略价格类型  PriceType[int]  // 策略价格详细类型  PriceTypeDetailType[int]  // 策略价格公式  PriceFormula[str]  // 触发模式  TriggerMethod[int]  // 是否启用内盘保护  InnerProtect[int]  // 是否被动腿下单失败立即平对应主动腿  PassiveFailCloseMainLeg[int]  // 触发模式为2：成交优先时设置的滑点数  SlipPoint[int]  // 追加保护策略  RecoverPriceMethod[int]  // 被动腿挂单多少秒后以市价追单  RecoverPriceSeconds[int]  // 策略设置类型  SetType[str]  // 策略行情价格最小变动单位  MinChangePrice[float]  // 主动腿单次最大下单量  MaxNum[int]  // 被动腿合约最小对盘挂单量  SuportQuantity[int]  // 安全深度  SafeDeep[int]  // 主动区域  MainRange[int]  // 单腿手动追被动设置  ManualZhuiDanBeyondNum[str]  // 单腿手动平主动设置  ManualPingZhuDongBeyondNum[str]  // 单腿自动定时设置，挂单秒数  AutoGuaDanSeconds[int]  // 单腿自动定时追被动设置  AutoZhuiBeiDongDots[int]  // 单腿自动定时平主动设置  AutoPingZhuDongDots[int]  // 单腿自动止损设置  AutoZhiSunDot[int]  // 单腿自动止损追被动设置，追价滑点数：  AutoZhiSunZhuiBeiDongDots[int]  // 单腿自动止损平主动设置，追价滑点数：  AutoZhiSunPingZhuDongDots[int]  // 策略行情价格小数位数  DotLen[int]  // 设置交易的时间段  TradeTime[str] |

参数：error

### 策略明细查询请求/返回reqQryStrategyDetail/ onRspQryStrategyDetail

函数原型：

|  |
| --- |
| reqQryStrategyDetail(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 策略编号  StrategyContractNo[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| void onRspQryStrategyDetail(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 策略商品编号  StartegyCommodityNo[str]  // 策略合约组合编号  StartegyContractNo[str]  // 价格  Price[str]  // 腿数（2个合约套利为2腿）  LegNum[int]  // 最后交易日（yyyy-MM-dd）  LastTradeDate[str]  // 策略中合约顺序号  SeqId[int]  // 策略中合约的商品编号  CommodityNo[str]  // 策略中合约的合约编号  ContractNo[str]  // 买卖方向（1表示买，2表示卖）  Direct[str]  // 合约下单数量  Quantity[int]  // 合约下单数量  Model[str]  // 最后更新日期（yyyy-MM-dd）  ModifyDate[str]  // 下单类型（限价，市价）  OrderMethod[str]  // 价格容忍度  PriceTolerance[float]  // 价格乘数参数  PriceParam[float]  // 合约最小对盘挂单量  SuportQuantity[int]  // 合约最小移动  MinMove[int]  // 合约定时追单  TimerOrder[int]  // 定时追单滑点数  TimerOrderBeyondNum[str]  // 合约定时追价  TimerOrderPrice[int]  // 定时追价滑点数  TimerOrderPriceBeyondNum[str]  // 定时追价触发次数  TimerOrderPriceTriggerNum[str]  // 止损追单  ZhiSunOrder[int]  // 止损追单滑点数  ZhiSunOrderBeyondNum[str]  // 合约定时平仓  TimerPingCang[int]  // 定时平仓滑点数  TimerPingCangBeyondNum[str]  // 安全深度  SafeDeep[int]  // 主动区域  MainRange[int] |

参数：error

### 设置双重认证密码/返回reqSetVerifyQA/ onRspSetVerifyQA

函数原型：

|  |
| --- |
| reqSetVerifyQA(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户帐号  UserId[str]  // 交易密码  UserPwd[str]  // 固定值，“1”  Type[str]  // 固定值，“1”  Question[str]  // 双重认证密码，英数字  Answer[str]  // 固定值，“1”  SaveMac[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspSetVerifyQA(error: dict, reqid: int, last: bool) |

参数：error

回调函数错误码指示成功时，调用reqSafeVerify函数绑定网卡地址。

### 绑定双重认证需要的MAC地址请求/返回reqSafeVerify/ onRspSafeVerify

函数原型：

|  |
| --- |
| reqSafeVerify(req: dict, reqid: int) |

参数：req

|  |
| --- |
| // 用户帐号  UserId[str]  // 交易密码  UserPwd[str]  // 固定值，“1”  Type[str]  // 固定值，“1”  Question[str]  // 双重认证密码，英数字  Answer[str]  // 固定值，“1”  SaveMac[str]  // MAC地址  MacAddress[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspSafeVerify(error: dict, reqid: int, last: bool) |

参数：error

### 查询品种列表请求/返回reqQryCommodity/ onRspQryCommodity

函数原型：

|  |
| --- |
| reqQryCommodity(req: dict, reqid: int) |

参数：req

|  |
| --- |
| //更新日期  UpdateDate[str]  //交易所编号  ExchangeNo[str]  // 错误信息  ErrorDescription[str] |

说明：UpdateDate不填值时，返回所有支持的品种。填入某个日期时，返回那个日期之后变化过的品种。

函数原型：

|  |
| --- |
| onRspQryCommodity(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| //商品编号，如GC，CL  CommodityNo[str]  // 交易所代码，如CME，HKEX  ExchangeNo[str]  //商品类型  CommodityType[str]  //商品名称  Name[str]  //交易状态  Enabled[int]  //更新日  RegDate[int]  //币种编号  CurrencyNo[float]  //账面跳点值  ProductDot[float]  //进阶单位  LowerTick[float]  //最小变动单位  UpperTick[float]  //行情小数点位数  DotNum[int]  //策略合约ID  StrikeCommodityId[int]  //期权类型  OptionStyle[str]  //交易所编号2 (在ExchangeNo下的交易所比如CME\_COMEX,CME\_ NYMEX CME\_CBT, CME)  ExchangeNo2[str]  //是否是个股期货  IsSFuture[float] |

说明：

ExchangeNo2/交易所编号2，这个字段是行情缴费使用，以CME为例，CME可以让投资者选择CME\_COMEX, CME\_ NYMEX,CME\_CBT, CME 其中的一个或多个付费，相应的，当API用户按交易所订阅行情时，应该填写这个交易所代码。

### 持仓总结查询请求/返回reqQryTotalPosition/ onRspQryTotalPosition

函数原型：

|  |
| --- |
| reqQryTotalPosition(req: dict, reqid: int) |

参数：req

|  |
| --- |
| //资金账号  AccountNo[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryTotalPosition(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| // 本地订单号  LocalOrderNo[str]  // 交易所代码  ExchangeNo[str]  // 合约代码  ContractCode[str]  // 定单号  OrderNo[str]  // 委托数量  OrderQty[int]  // 已成交数量  FilledQty[int]  // 成交均价  FilledAvgPrice[float]  // 多仓/持买数量  LongPositionQty[int]  //多仓/持买开仓均价  LongPosAveragePrx[float]  //中国品种(比如INE/sc)盯市用的平均价(持买均价)  CNLongPosAveragePrx[float]  // 空仓/持卖数量  ShortPositionQty[int]  //空仓/持卖开仓均价  ShortPosAveragePrx[float]  //中国品种(比如INE/sc)盯市用的平均价（持卖均价）  CNShortPosAveragePrx[float]  // 是否已经撤单（0：没有；1：已撤单）  IsCanceled[str]  // 成交总的手续费  FilledTotalFee[float]  // 顺序号  SequenceNo[int]  // 资金帐号  AccountNo[str]  // 持仓类型（0：昨仓；1：今仓）  PositionType[str]  // 多仓/持买保证金  LongPosMargin[float]  // 空仓/持卖保证金  ShortPosMargin[float]  // 最新价  CurrPrice[float]  // 浮动盈亏  ProfitLoss[float] |

### 交易所冬夏令开盘时间查询请求/返回reqQryExchangeTime/ onRspQryExchangeTime

函数原型：

|  |
| --- |
| reqQryExchangeTime(req: dict, reqid: int) |

参数：req

|  |
| --- |
| //不使用  Unused[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryExchangeTime(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| //当前年份  Year[str]  //夏令时开始时间  SummerBegin[str]  //冬令时开始时间  WinterBegin[str]  //交易所代码  ExchangeNo[str]  //交易所名称  Name[int] |

### 品种开收盘时间查询请求/返回reqQryExchangeTime/ onRspQryCommodityTime

函数原型：

|  |
| --- |
| reqQryCommodityTime(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| //交易所代码  ExchangeNo[str]  //品种代码  CommodityNo[str]  // 错误信息  ErrorDescription[str] |

函数原型：

|  |
| --- |
| onRspQryCommodityTime(data: dict, error: dict, reqid: int, last: bool) |

参数：data

|  |
| --- |
| //是否跨日  CrossTrade[str]  //交易段  Stage[str]  //2冬令 1夏令  Summer[str]  //产品开市时间  Opendate[str]  //产品收市时间  Closingdate[str]  //品种代码  CommodityNo[str]  //品种名称  ComName[str]  //交易所代码  ExchangeNo[str]  //交易所名称  ExName[str] |

# 常见问题-期货交易相关

1、合约乘数

合约乘数 = (点值/最小变动单位)\*(进阶单位/10的（行情小数点位数）次方)

说明：

如果不做美债

可以只看 点值/最小变动单位

因为大部分品种都是10进制 后面算出来都是1

2.、保证金

保证金 是下单时验证的保证金， 维持保证金 是其他时候计算的保证金

一个大点的点值=合约点值\*进价单位,除以10的小数点位数次方,再除以最小跳点

3、盈亏

持仓浮盈=持仓数\*（开仓价-最新价）\*一个大点的点值

开仓价和最新价都要转成十进制的价格

十进制的价格=价格整数+小数\*10的小数位数次方除以进价单位

合约点值=336 进价单位=532 小数点位数次方=531 最小跳点=337

4、持仓及权益

持仓：通过请求的持仓明细来计算，按时间先后，买卖直接平仓。

盯市盈亏：（昨结算价-行情最新价）\*持仓数量\*合约点值；

今权益：今结存+浮盈+未结平盈+未到期平盈

今权益= Tags.todayAccount +浮盈+Tags. LMEUnaccountPL+Tags. LMEUnexpiredPL;

今可用：今权益-保证金-冻结资金

今可用=今权益-Tags. Deposit-Tags. frozenDeposit;

持过夜仓，开盘价就是昨天的结算价，浮盈使用昨天的结算价计算。

5、期权

期权：

权利金：期权合约开仓价格\*最小跳点价值 ╱最小跳点 \*手数；

期权价值：期权合约最新价\*最小跳点价值 ╱ 最小跳点 \*手数；

期权净盈亏：期权价值-权利金；

6、撤单

需要填系统号 订单号，商品 交易所和下单方向

7、tag50字段说明，下改撤单中tag50雷同

1. 须使用已在经纪公司登记过的操作员识别码（Operator ID，旧称Tag50）下单，否则订单将会被拒绝。申请开通API 的客户可通过《尽职调查暨操作人授权书》登记申报Tag50；
2. 2.如客户长时间交易，例如超过12小时，需设置多轮管理、操作和监控（至少设置2 个班次）。当负责管理、操作和监控的人员发生变化时，新一班次人员需分配不同于前一班次的Tag50，并在订单中传输。

# 常见问题-股票交易相关

1、 get cash balance (USD)

查资金余额

可取资金: CanCashOutMoneyAmount

可用资金:FBalance (DB) 结构体中(TodayTradableFund)

可用资金包括直达借给客户的钱。

可取资金是客户自己的钱。

所以可用资金可能大于可取资金，分成两个字段。

2,、get total asset value (USD)

查总资产，就是可取资金+持仓市值

市值->没有直接的结果，需要客户自己算。

(最新价\*总持仓量(CanSellShares+FrosenShares))

总持仓量由两部分组成：可卖数量(CanSellShares) + 冻结数量(FrosenShares)

冻结数量： 当客户有排队单未成交时，排队单的数量就是冻结数量

3、get positions (should return list of symbols and its holding)

查持仓(结果中要有股票代码和对应股数)

CanSellShares可卖量，需要考虑排队单的冻结数量(FrosenShares)

4、place mkt order (buy and sell)

下市价买单 和 市价卖单

5、撤单

需要填系统号 订单号，商品 交易所和下单方向