XTP极速交易系统QuoteAPI

制作者 中泰证券股份有限公司

2018年 六月 28日 星期四 17:37:59

Contents

Chapter 1

XTP 极速行情交易系统 Quote API 1.1.18.13

本项目是XTP项目中的行情类接口

(1) XTP的行情订阅接口和响应类 xtp_quote_api.h

(2) 行情订阅测试Demo demo_test_quote_api.cpp

Chapter 2

继承关系索引

2.1 类继承关系

此继承关系列表按字典顺序粗略的排序:

Chapter 3

结构体索引

3.1 结构体

这里列出了所有结构体，并附带简要说明:

Chapter 4

文件索引

4.1 文件列表

这里列出了所有文档化的文件，并附带简要说明:

Chapter 5

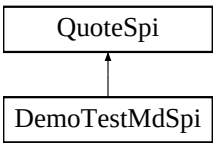
结构体说明

5.1 DemoTestMdSpi类 参考

Demo自定义行情订阅接口响应类

`#include <demo_test_quote_spi.h>`

类 DemoTestMdSpi 继承关系图:



Public 成员函数

- virtual void OnDisconnected (int reason)
  当客户端与行情后台通信连接断开
- virtual void OnError (XTPRI *error_info)
  错误应答
- virtual void OnSubMarketData (XTPST *ticker, XTPRI *error_info, bool is_last)
  重写订阅行情应答
- virtual void OnUnSubMarketData (XTPST *ticker, XTPRI *error_info, bool is_last)
  重写取消订阅行情应答
- virtual void OnDepthMarketData (XTPMD *market_data, int64_t bid1_qty[], int32_t bid1_count, int32_t max_bid1_count, int64_t ask1_qty[], int32_t ask1_count, int32_t max_ask1_count)
  重写行情通知
- virtual void OnSubOrderBook (XTPST *ticker, XTPRI *error_info, bool is_last)
  重写订阅行情订单簿应答
- virtual void OnUnSubOrderBook (XTPST *ticker, XTPRI *error_info, bool is_last)
  重写退订行情订单簿应答
- virtual void OnSubTickByTick (XTPST *ticker, XTPRI *error_info, bool is_last)
  订阅逐笔行情应答
- virtual void OnUnSubTickByTick (XTPST *ticker, XTPRI *error_info, bool is_last)
  退订逐笔行情应答
- virtual void OnOrderBook (XTPOB *order_book)
  行情订单簿通知
- virtual void OnTickByTick (XTPTBT *tbt_data)

逐笔行情通知，包括股票、指数和期权

- virtual void OnQueryAllTickers (XTPQSI *ticker_info, XTPRI *error_info, bool is_last)
  查询可交易合约的应答
- virtual void OnQueryTickersPriceInfo (XTPTPI *ticker_info, XTPRI *error_info, bool is_last)
  查询合约的最新价格信息应答
- virtual void OnSubscribeAllMarketData (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
  订阅全市场的股票行情应答
- virtual void OnUnSubscribeAllMarketData (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
  退订全市场的股票行情应答
- virtual void OnSubscribeAllOrderBook (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
  订阅全市场的股票行情订单簿应答
- virtual void OnUnSubscribeAllOrderBook (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
  退订全市场的股票行情订单簿应答
- virtual void OnSubscribeAllTickByTick (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
  订阅全市场的股票逐笔行情应答
- virtual void OnUnSubscribeAllTickByTick (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
  退订全市场的股票逐笔行情应答
- virtual void OnSubscribeAllOptionMarketData (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
  订阅全市场的期权行情应答
- virtual void OnUnSubscribeAllOptionMarketData (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
  退订全市场的期权行情应答
- virtual void OnSubscribeAllOptionOrderBook (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
  订阅全市场的期权行情订单簿应答
- virtual void OnUnSubscribeAllOptionOrderBook (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
  退订全市场的期权行情订单簿应答
- virtual void OnSubscribeAllOptionTickByTick (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
  订阅全市场的期权逐笔行情应答
- virtual void OnUnSubscribeAllOptionTickByTick (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
  退订全市场的期权逐笔行情应答

5.1.1 详细描述

Demo自定义行情订阅接口响应类

作者

中泰证券股份有限公司

日期

十月 2015

该类的文档由以下文件生成:

- demo_test_quote_spi.h

5.2 OrderBookStruct结构体 参考

定单薄

```
#include <xquote_api_struct.h>
```

成员变量

- XTP_EXCHANGE_TYPE exchange_id
  交易所代码
- char ticker [XTP_TICKER_LEN]
  合约代码（不包含交易所信息），不带空格，以'\0'结尾
- double last_price
  最新价
- int64_t qty
  数量，为总成交量
- double turnover
  成交金额，为总成交金额
- int64_t trades_count
  成交笔数
- double bid [10]
  十档申买价
- double ask [10]
  十档申卖价
- int64_t bid_qty [10]
  十档申买量
- int64_t ask_qty [10]
  十档申卖量
- int64_t data_time
  时间类

5.2.1 详细描述

定单薄

该结构体的文档由以下文件生成:

- xquote_api_struct.h

5.3 QuoteApi类 参考

行情订阅接口类

```
#include <xtp_quote_api.h>
```

Public 成员函数

- virtual void Release ()=0
- virtual const char * GetTradingDay ()=0
- virtual const char * GetApiVersion ()=0
- virtual XTPRI * GetApiLastError ()=0
- virtual void SetUDPBufferSize (uint32_t buff_size)=0
- virtual void RegisterSpi (QuoteSpi *spi)=0
- virtual void SetHeartBeatInterval (uint32_t interval)=0
- virtual int SubscribeMarketData (char *ticker[ ], int count, XTP_EXCHANGE_TYPE exchange_id)=0
- virtual int UnSubscribeMarketData (char *ticker[ ], int count, XTP_EXCHANGE_TYPE exchange_id)=0
- virtual int SubscribeOrderBook (char *ticker[ ], int count, XTP_EXCHANGE_TYPE exchange_id)=0
- virtual int UnSubscribeOrderBook (char *ticker[ ], int count, XTP_EXCHANGE_TYPE exchange_id)=0

- virtual int SubscribeTickByTick (char *ticker[ ], int count, XTP_EXCHANGE_TYPE exchange_id)=0
- virtual int UnSubscribeTickByTick (char *ticker[ ], int count, XTP_EXCHANGE_TYPE exchange_id)=0
- virtual int SubscribeAllMarketData (XTP_EXCHANGE_TYPE exchange_id=XTP_EXCHANGE_UNKNOWN)=0
- virtual int UnSubscribeAllMarketData (XTP_EXCHANGE_TYPE exchange_id=XTP_EXCHANGE_UNKNOWN)=0
- virtual int SubscribeAllOrderBook (XTP_EXCHANGE_TYPE exchange_id=XTP_EXCHANGE_UNKNOWN)=0
- virtual int UnSubscribeAllOrderBook (XTP_EXCHANGE_TYPE exchange_id=XTP_EXCHANGE_UNKNOWN)=0
- virtual int SubscribeAllTickByTick (XTP_EXCHANGE_TYPE exchange_id=XTP_EXCHANGE_UNKNOWN)=0
- virtual int UnSubscribeAllTickByTick (XTP_EXCHANGE_TYPE exchange_id=XTP_EXCHANGE_UNKNOWN)=0
- virtual int Login (const char *ip, int port, const char *user, const char *password, XTP_PROTOCOL_TYPE sock_type)=0
- virtual int Logout ()=0
- virtual int QueryAllTickers (XTP_EXCHANGE_TYPE exchange_id)=0
- virtual int QueryTickersPriceInfo (char *ticker[ ], int count, XTP_EXCHANGE_TYPE exchange_id)=0
- virtual int QueryAllTickersPriceInfo ()=0
- virtual int SubscribeAllOptionMarketData (XTP_EXCHANGE_TYPE exchange_id=XTP_EXCHANGE_UNKNOWN)=0
- virtual int UnSubscribeAllOptionMarketData (XTP_EXCHANGE_TYPE exchange_id=XTP_EXCHANGE_UNKNOWN)=0
- virtual int SubscribeAllOptionOrderBook (XTP_EXCHANGE_TYPE exchange_id=XTP_EXCHANGE_UNKNOWN)=0
- virtual int UnSubscribeAllOptionOrderBook (XTP_EXCHANGE_TYPE exchange_id=XTP_EXCHANGE_UNKNOWN)=0
- virtual int SubscribeAllOptionTickByTick (XTP_EXCHANGE_TYPE exchange_id=XTP_EXCHANGE_UNKNOWN)=0
- virtual int UnSubscribeAllOptionTickByTick (XTP_EXCHANGE_TYPE exchange_id=XTP_EXCHANGE_UNKNOWN)=0

静态 Public 成员函数

- static QuoteApi * CreateQuoteApi (uint8_t client_id, const char *save_file_path, XTP_LOG_LEVEL log_level=XTP_LOG_LEVEL_DEBUG)

5.3.1 详细描述

行情订阅接口类

作者

中泰证券股份有限公司

日期

十月 2015

5.3.2 成员函数说明

5.3.2.1 static QuoteApi* CreateQuoteApi (uint8_t *client_id,* const char * *save_file_path,* XTP_LOG_LEVEL *log_level =* XTP_LOG_LEVEL_DEBUG) [static]

创建QuoteApi

参数

client_id	（必须输入）用于区分同一用户的不同客户端，由用户自定义
save_file_path	（必须输入）存贮订阅信息文件的目录，请设定一个有可写权限的真实存在的路径
log_level	日志输出级别

返回

创建出的UserApi

备注

如果一个账户需要在多个客户端登录，请使用不同的client_id，系统允许一个账户同时登录多个客户端，但是对于同一账户，相同的client_id只能保持一个session连接，后面的登录在前一个session存续期间，无法连接

5.3.2.2 virtual XTPRI* GetApiLastError ( ) [pure virtual]

获取API的系统错误

返回

返回的错误信息，可以在Login、Logout、订阅、取消订阅失败时调用，获取失败的原因

备注

可以在调用api接口失败时调用，例如login失败时

5.3.2.3 virtual const char* GetApiVersion ( ) [pure virtual]

获取API的发行版本号

返回

返回api发行版本号

5.3.2.4 virtual const char* GetTradingDay ( ) [pure virtual]

获取当前交易日

返回

获取到的交易日

备注

只有登录成功后,才能得到正确的交易日

5.3.2.5 virtual int Login (const char * *ip*, int *port*, const char * *user*, const char * *password*, XTP_PROTOCOL_TYPE *sock_type*) [pure virtual]

用户登录请求

返回

登录是否成功，“0”表示登录成功，“-1”表示连接服务器出错，此时用户可以调用GetApiLastError()来获取错误代码，“-2”表示已存在连接，不允许重复登录，如果需要重连，请先logout，“-3”表示输入有错误

参数

ip	服务器ip地址，类似“127.0.0.1”
port	服务器端口号
user	登陆用户名
password	登陆密码
sock_type	“1”代表TCP，“2”代表UDP

备注

此函数为同步阻塞式，不需要异步等待登录成功，当函数返回即可进行后续操作，此api只能有一个连接

5.3.2.6 virtual int Logout ( ) [pure virtual]

登出请求

返回

登出是否成功，“0”表示登出成功，非“0”表示登出出错，此时用户可以调用GetApiLastError()来获取错误代码

备注

此函数为同步阻塞式，不需要异步等待登出，当函数返回即可进行后续操作

5.3.2.7 virtual int QueryAllTickers (XTP_EXCHANGE_TYPE *exchange_id*) [pure virtual]

获取当前交易日可交易合约

返回

查询是否成功，“0”表示查询成功，非“0”表示查询不成功

参数

exchange_id	交易所代码

5.3.2.8 virtual int QueryAllTickersPriceInfo ( ) [pure virtual]

获取所有合约的最新价格信息

返回

查询是否成功，“0”表示查询成功，非“0”表示查询不成功

5.3.2.9 virtual int QueryTickersPriceInfo (char * *ticker[ ],* int *count,* XTP_EXCHANGE_TYPE *exchange_id*) [pure virtual]

获取合约的最新价格信息

返回

查询是否成功，“0”表示查询成功，非“0”表示查询不成功

参数

ticker	合约ID数组，注意合约代码必须以'\0'结尾，不包含空格
count	要查询的最新价的合约个数
exchange_id	交易所代码

5.3.2.10 virtual void RegisterSpi (QuoteSpi * *spi*) [pure virtual]

注册回调接口

参数

spi	派生自回调接口类的实例，请在登录之前设定

5.3.2.11 virtual void Release ( ) [pure virtual]

删除接口对象本身

备注

不再使用本接口对象时,调用该函数删除接口对象

5.3.2.12 virtual void SetHeartBeatInterval (uint32_t *interval*) [pure virtual]

设置心跳检测时间间隔，单位为秒

参数

interval	心跳检测时间间隔，单位为秒

备注

此函数必须在Login之前调用

5.3.2.13 virtual void SetUDPBufferSize (uint32_t *buff_size*) [pure virtual]

设置采用UDP方式连接时的接收缓冲区大小

备注

需要在Login之前调用，默认大小和最小设置均为64MB。此缓存大小单位为MB，请输入2的次方数，例如128MB请输入128。

5.3.2.14 virtual int SubscribeAllMarketData (XTP_EXCHANGE_TYPE *exchange_id* = XTP_EXCHANGE_UNKNOWN) [pure virtual]

订阅全市场的股票行情

返回

订阅全市场行情接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

exchange_id	表示当前全订阅的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，↩ XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场

备注

需要与全市场退订行情接口配套使用

5.3.2.15 virtual int SubscribeAllOptionMarketData (XTP_EXCHANGE_TYPE *exchange_id* = XTP_EXCHANGE_UNKNOWN) [pure virtual]

订阅全市场的期权行情

返回

订阅全市期权场行情接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

exchange_id	表示当前全订阅的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，↩ XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场

备注

需要与全市场退订期权行情接口配套使用

5.3.2.16 virtual int SubscribeAllOptionOrderBook (XTP_EXCHANGE_TYPE *exchange_id* = XTP_EXCHANGE_UNKNOWN) [pure virtual]

订阅全市场的期权行情订单簿

返回

订阅全市场期权行情订单簿接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

exchange_id	表示当前全订阅的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，↩ XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场

备注

需要与全市场退订期权行情订单簿接口配套使用

5.3.2.17 virtual int SubscribeAllOptionTickByTick (XTP_EXCHANGE_TYPE *exchange_id* = XTP_EXCHANGE_UNKNOWN) [pure virtual]

订阅全市场的期权逐笔行情

返回

订阅全市场期权逐笔行情接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

exchange_id	表示当前全订阅的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，↩ XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场

备注

需要与全市场退订期权逐笔行情接口配套使用

5.3.2.18 virtual int SubscribeAllOrderBook (XTP_EXCHANGE_TYPE *exchange_id* = XTP_EXCHANGE_UNKNOWN) [pure virtual]

订阅全市场的股票行情订单簿

返回

订阅全市场行情订单簿接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

exchange_id	表示当前全订阅的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，↩ XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场

备注

需要与全市场退订行情订单簿接口配套使用

5.3.2.19 virtual int SubscribeAllTickByTick (XTP_EXCHANGE_TYPE *exchange_id* = XTP_EXCHANGE_UNKNOWN) [pure virtual]

订阅全市场的股票逐笔行情

返回

订阅全市场逐笔行情接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

exchange_id	表示当前全订阅的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，↩ XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场

备注

需要与全市场退订逐笔行情接口配套使用

5.3.2.20 virtual int SubscribeMarketData (char * *ticker[ ],* int *count,* XTP_EXCHANGE_TYPE *exchange_id*) [pure virtual]

订阅行情，包括股票、指数和期权。

返回

订阅接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

ticker	合约ID数组，注意合约代码必须以'\0'结尾，不包含空格
count	要订阅/退订行情的合约个数
exchange_id	交易所代码

备注

可以一次性订阅同一证券交易所的多个合约，无论用户因为何种问题需要重新登录行情服务器，都需要重新订阅行情

5.3.2.21 virtual int SubscribeOrderBook (char * *ticker[ ],* int *count,* XTP_EXCHANGE_TYPE *exchange_id*) [pure virtual]

订阅行情订单簿，包括股票、指数和期权。

返回

订阅行情订单簿接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

ticker	合约ID数组，注意合约代码必须以'\0'结尾，不包含空格
count	要订阅/退订行情订单簿的合约个数
exchange_id	交易所代码

备注

可以一次性订阅同一证券交易所的多个合约，无论用户因为何种问题需要重新登录行情服务器，都需要重新订阅行情(仅支持深交所)

5.3.2.22 virtual int SubscribeTickByTick (char * *ticker[ ],* int *count,* XTP_EXCHANGE_TYPE *exchange_id*) [pure virtual]

订阅逐笔行情，包括股票、指数和期权。

返回

订阅逐笔行情接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

ticker	合约ID数组，注意合约代码必须以'\0'结尾，不包含空格
count	要订阅/退订行情订单簿的合约个数
exchange_id	交易所代码

备注

可以一次性订阅同一证券交易所的多个合约，无论用户因为何种问题需要重新登录行情服务器，都需要重新订阅行情

5.3.2.23 virtual int UnSubscribeAllMarketData (XTP_EXCHANGE_TYPE *exchange_id* = XTP_EXCHANGE_UNKNOWN) [pure virtual]

退订全市场的股票行情

返回

退订全市场行情接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

exchange_id	表示当前退订的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场

备注

需要与订阅全市场行情接口配套使用

5.3.2.24 virtual int UnSubscribeAllOptionMarketData (XTP_EXCHANGE_TYPE *exchange_id* = XTP_EXCHANGE_UNKNOWN) [pure virtual]

退订全市场的期权行情

返回

退订全市场期权行情接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

exchange_id	表示当前退订的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场

备注

需要与订阅全市场期权行情接口配套使用

5.3.2.25 virtual int UnSubscribeAllOptionOrderBook (XTP_EXCHANGE_TYPE *exchange_id* = XTP_EXCHANGE_UNKNOWN) [pure virtual]

退订全市场的期权行情订单簿

返回

退订全市场期权行情订单簿接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

exchange_id	表示当前退订的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场

备注

需要与订阅全市场期权行情订单簿接口配套使用

5.3.2.26 virtual int UnSubscribeAllOptionTickByTick (XTP_EXCHANGE_TYPE *exchange_id* = XTP_EXCHANGE_UNKNOWN) [pure virtual]

退订全市场的期权逐笔行情

返回

退订全市场期权逐笔行情接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

exchange_id	表示当前退订的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场

备注

需要与订阅全市场期权逐笔行情接口配套使用

5.3.2.27 virtual int UnSubscribeAllOrderBook (XTP_EXCHANGE_TYPE *exchange_id* = XTP_EXCHANGE_UNKNOWN) [pure virtual]

退订全市场的股票行情订单簿

返回

退订全市场行情订单簿接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

exchange_id	表示当前退订的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场

备注

需要与订阅全市场行情订单簿接口配套使用

5.3.2.28 virtual int UnSubscribeAllTickByTick (XTP_EXCHANGE_TYPE *exchange_id* = XTP_EXCHANGE_UNKNOWN) [pure virtual]

退订全市场的股票逐笔行情

返回

退订全市场逐笔行情接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

exchange_id	表示当前退订的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场

备注

需要与订阅全市场逐笔行情接口配套使用

5.3.2.29 virtual int UnSubscribeMarketData (char * *ticker[ ],* int *count,* XTP_EXCHANGE_TYPE *exchange_id*) [pure virtual]

退订行情，包括股票、指数和期权。

返回

取消订阅接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

ticker	合约ID数组，注意合约代码必须以'\0'结尾，不包含空格
count	要订阅/退订行情的合约个数
exchange_id	交易所代码

备注

可以一次性取消订阅同一证券交易所的多个合约，需要与订阅行情接口配套使用

5.3.2.30 virtual int UnSubscribeOrderBook (char * *ticker[ ],* int *count,* XTP_EXCHANGE_TYPE *exchange_id*) [pure virtual]

退订行情订单簿，包括股票、指数和期权。

返回

取消订阅行情订单簿接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

ticker	合约ID数组，注意合约代码必须以'\0'结尾，不包含空格
count	要订阅/退订行情订单簿的合约个数
exchange_id	交易所代码

备注

可以一次性取消订阅同一证券交易所的多个合约，需要与订阅行情订单簿接口配套使用

5.3.2.31 virtual int UnSubscribeTickByTick (char * *ticker[ ],* int *count,* XTP_EXCHANGE_TYPE *exchange_id*) [pure virtual]

退订逐笔行情，包括股票、指数和期权。

返回

取消订阅逐笔行情接口调用是否成功，“0”表示接口调用成功，非“0”表示接口调用出错

参数

ticker	合约ID数组，注意合约代码必须以'\0'结尾，不包含空格
count	要订阅/退订行情订单簿的合约个数
exchange_id	交易所代码

备注

可以一次性取消订阅同一证券交易所的多个合约，需要与订阅逐笔行情接口配套使用

该类的文档由以下文件生成:

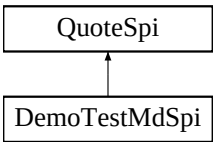
- xtp_quote_api.h

5.4 QuoteSpi类 参考

行情回调类

`#include <xtp_quote_api.h>`

类 QuoteSpi 继承关系图:



Public 成员函数

- virtual void OnDisconnected (int reason)
- virtual void OnError (XTPRI *error_info)
- virtual void OnSubMarketData (XTPST *ticker, XTPRI *error_info, bool is_last)
- virtual void OnUnSubMarketData (XTPST *ticker, XTPRI *error_info, bool is_last)
- virtual void OnDepthMarketData (XTPMD *market_data, int64_t bid1_qty[], int32_t bid1_count, int32_t max_bid1_count, int64_t ask1_qty[], int32_t ask1_count, int32_t max_ask1_count)
- virtual void OnSubOrderBook (XTPST *ticker, XTPRI *error_info, bool is_last)
- virtual void OnUnSubOrderBook (XTPST *ticker, XTPRI *error_info, bool is_last)
- virtual void OnOrderBook (XTPOB *order_book)
- virtual void OnSubTickByTick (XTPST *ticker, XTPRI *error_info, bool is_last)
- virtual void OnUnSubTickByTick (XTPST *ticker, XTPRI *error_info, bool is_last)
- virtual void OnTickByTick (XTPTBT *tbt_data)
- virtual void OnSubscribeAllMarketData (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
- virtual void OnUnSubscribeAllMarketData (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
- virtual void OnSubscribeAllOrderBook (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
- virtual void OnUnSubscribeAllOrderBook (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
- virtual void OnSubscribeAllTickByTick (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
- virtual void OnUnSubscribeAllTickByTick (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
- virtual void OnQueryAllTickers (XTPQSI *ticker_info, XTPRI *error_info, bool is_last)
- virtual void OnQueryTickersPriceInfo (XTPTPI *ticker_info, XTPRI *error_info, bool is_last)
- virtual void OnSubscribeAllOptionMarketData (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
- virtual void OnUnSubscribeAllOptionMarketData (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
- virtual void OnSubscribeAllOptionOrderBook (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
- virtual void OnUnSubscribeAllOptionOrderBook (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
- virtual void OnSubscribeAllOptionTickByTick (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)
- virtual void OnUnSubscribeAllOptionTickByTick (XTP_EXCHANGE_TYPE exchange_id, XTPRI *error_info)

5.4.1 详细描述

行情回调类

作者

中泰证券股份有限公司

日期

十月 2015

5.4.2 成员函数说明

5.4.2.1 virtual void OnDepthMarketData (XTPMD * *market_data*, int64_t *bid1_qty[]*, int32_t *bid1_count*, int32_t *max_bid1_count*, int64_t *ask1_qty[]*, int32_t *ask1_count*, int32_t *max_ask1_count*) [inline], [virtual]

深度行情通知，包含买一卖一队列

参数

market_data	行情数据
bid1_qty	买一队列数据
bid1_count	买一队列的有效委托笔数
max_bid1_count	买一队列总委托笔数
ask1_qty	卖一队列数据
ask1_count	卖一队列的有效委托笔数
max_ask1_count	卖一队列总委托笔数

备注

需要快速返回，否则会堵塞后续消息，当堵塞严重时，会触发断线

被 DemoTestMdSpi 重载.

5.4.2.2 virtual void OnDisconnected (int *reason*) [inline], [virtual]

当客户端与行情后台通信连接断开时，该方法被调用。

参数

reason	错误原因，请与错误代码表对应

备注

api不会自动重连，当断线发生时，请用户自行选择后续操作。可以在此函数中调用Login重新登录。注意用户重新登录后，需要重新订阅行情

被 DemoTestMdSpi 重载.

5.4.2.3 virtual void OnError (XTPRI * *error_info*) [inline], [virtual]

错误应答

参数

error_info	当服务器响应发生错误时的具体的错误代码和错误信息，当error_info为空，或者error_info.error_id为0时，表明没有错误

备注

此函数只有在服务器发生错误时才会调用，一般无需用户处理

被 DemoTestMdSpi 重载.

5.4.2.4 virtual void OnOrderBook (XTPOB * *order_book*) [inline], [virtual]

行情订单簿通知，包括股票、指数和期权

参数

order_book	行情订单簿数据，需要快速返回，否则会堵塞后续消息，当堵塞严重时，会触发断线

被 DemoTestMdSpi 重载.

5.4.2.5 virtual void OnQueryAllTickers (XTPQSI * *ticker_info*, XTPRI * *error_info*, bool *is_last*) [inline], [virtual]

查询可交易合约的应答

参数

ticker_info	可交易合约信息
error_info	查询可交易合约时发生错误时返回的错误信息，当error_info为空，或者error_info.↩ error_id为0时，表明没有错误
is_last	是否此次查询可交易合约的最后一个应答，当为最后一个的时候为true，如果为false，表示还有其他后续消息响应

被 DemoTestMdSpi 重载.

5.4.2.6 virtual void OnQueryTickersPriceInfo (XTPTPI * *ticker_info*, XTPRI * *error_info*, bool *is_last*) [inline], [virtual]

查询合约的最新价格信息应答

参数

ticker_info	合约的最新价格信息
error_info	查询合约的最新价格信息时发生错误时返回的错误信息，当error_info为空，或者error_info.error_id为0时，表明没有错误
is_last	是否此次查询的最后一个应答，当为最后一个的时候为true，如果为false，表示还有其他后续消息响应

被 DemoTestMdSpi 重载.

5.4.2.7 virtual void OnSubMarketData (XTPST * *ticker*, XTPRI * *error_info*, bool *is_last*) [inline], [virtual]

订阅行情应答，包括股票、指数和期权

参数

ticker	详细的合约订阅情况
error_info	订阅合约发生错误时的错误信息，当error_info为空，或者error_info.error_id为0时，表明没有错误
is_last	是否此次订阅的最后一个应答，当为最后一个的时候为true，如果为false，表示还有其他后续消息响应

备注

每条订阅的合约均对应一条订阅应答，需要快速返回，否则会堵塞后续消息，当堵塞严重时，会触发断线

被 DemoTestMdSpi 重载.

5.4.2.8 virtual void OnSubOrderBook (XTPST * *ticker*, XTPRI * *error_info*, bool *is_last*) [inline], [virtual]

订阅行情订单簿应答，包括股票、指数和期权

参数

ticker	详细的合约订阅情况
error_info	订阅合约发生错误时的错误信息，当error_info为空，或者error_info.error_id为0时，表明没有错误
is_last	是否此次订阅的最后一个应答，当为最后一个的时候为true，如果为false，表示还有其他后续消息响应

备注

每条订阅的合约均对应一条订阅应答，需要快速返回，否则会堵塞后续消息，当堵塞严重时，会触发断线

被 DemoTestMdSpi 重载.

5.4.2.9 virtual void OnSubscribeAllMarketData (XTP_EXCHANGE_TYPE *exchange_id*, XTPRI * *error_info*) [inline], [virtual]

订阅全市场的股票行情应答

参数

exchange_id	表示当前全订阅的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场
error_info	取消订阅合约时发生错误时返回的错误信息，当error_info为空，或者error_info.error_id为0时，表明没有错误

备注

需要快速返回

被 DemoTestMdSpi 重载.

5.4.2.10 virtual void OnSubscribeAllOptionMarketData (XTP_EXCHANGE_TYPE *exchange_id*, XTPRI * *error_info*) [inline], [virtual]

订阅全市场的期权行情应答

参数

exchange_id	表示当前全订阅的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，↩ XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场
error_info	取消订阅合约时发生错误时返回的错误信息，当error_info为空，或者error_info.↩ error_id为0时，表明没有错误

备注

需要快速返回

被 DemoTestMdSpi 重载.

5.4.2.11 virtual void OnSubscribeAllOptionOrderBook (XTP_EXCHANGE_TYPE *exchange_id,* XTPRI * *error_info*)

`[inline],[virtual]`

订阅全市场的期权行情订单簿应答

参数

exchange_id	表示当前全订阅的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，↩ XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场
error_info	取消订阅合约时发生错误时返回的错误信息，当error_info为空，或者error_info.↩ error_id为0时，表明没有错误

备注

需要快速返回

被 DemoTestMdSpi 重载.

5.4.2.12 virtual void OnSubscribeAllOptionTickByTick (XTP_EXCHANGE_TYPE *exchange_id,* XTPRI * *error_info*)

`[inline],[virtual]`

订阅全市场的期权逐笔行情应答

参数

exchange_id	表示当前全订阅的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，↩ XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场
error_info	取消订阅合约时发生错误时返回的错误信息，当error_info为空，或者error_info.↩ error_id为0时，表明没有错误

备注

需要快速返回

被 DemoTestMdSpi 重载.

5.4.2.13 virtual void OnSubscribeAllOrderBook (XTP_EXCHANGE_TYPE *exchange_id,* XTPRI * *error_info*)

`[inline],[virtual]`

订阅全市场的股票行情订单簿应答

参数

exchange_id	表示当前全订阅的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，↩ XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场
error_info	取消订阅合约时发生错误时返回的错误信息，当error_info为空，或者error_info.↩ error_id为0时，表明没有错误

备注

需要快速返回

被 DemoTestMdSpi 重载.

5.4.2.14 virtual void OnSubscribeAllTickByTick (XTP_EXCHANGE_TYPE *exchange_id*, XTPRI * *error_info*) [inline], [virtual]

订阅全市场的股票逐笔行情应答

参数

exchange_id	表示当前全订阅的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，↩ XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场
error_info	取消订阅合约时发生错误时返回的错误信息，当error_info为空，或者error_info.↩ error_id为0时，表明没有错误

备注

需要快速返回

被 DemoTestMdSpi 重载.

5.4.2.15 virtual void OnSubTickByTick (XTPST * *ticker*, XTPRI * *error_info*, bool *is_last*) [inline], [virtual]

订阅逐笔行情应答，包括股票、指数和期权

参数

ticker	详细的合约订阅情况
error_info	订阅合约发生错误时的错误信息，当error_info为空，或者error_info.error_id为0时，表明没有错误
is_last	是否此次订阅的最后一个应答，当为最后一个的时候为true，如果为false，表示还有其他后续消息响应

备注

每条订阅的合约均对应一条订阅应答，需要快速返回，否则会堵塞后续消息，当堵塞严重时，会触发断线

被 DemoTestMdSpi 重载.

5.4.2.16 virtual void OnTickByTick (XTPTBT * *tbt_data*) [inline], [virtual]

逐笔行情通知，包括股票、指数和期权

参数

tbt_data	逐笔行情数据，包括逐笔委托和逐笔成交，此为共用结构体，需要根据type来区分是逐笔委托还是逐笔成交，需要快速返回，否则会堵塞后续消息，当堵塞严重时，会触发断线

被 DemoTestMdSpi 重载.

5.4.2.17 virtual void OnUnSubMarketData (XTPST * *ticker*, XTPRI * *error_info*, bool *is_last*) [inline], [virtual]

退订行情应答，包括股票、指数和期权

参数

ticker	详细的合约取消订阅情况
error_info	取消订阅合约时发生错误时返回的错误信息，当error_info为空，或者error_info.↩ error_id为0时，表明没有错误
is_last	是否此次取消订阅的最后一个应答，当为最后一个的时候为true，如果为false，表示还有其他后续消息响应

备注

每条取消订阅的合约均对应一条取消订阅应答，需要快速返回，否则会堵塞后续消息，当堵塞严重时，会触发断线

被 DemoTestMdSpi 重载.

5.4.2.18 virtual void OnUnSubOrderBook (XTPST * *ticker*, XTPRI * *error_info*, bool *is_last*) [inline], [virtual]

退订行情订单簿应答，包括股票、指数和期权

参数

ticker	详细的合约取消订阅情况
error_info	取消订阅合约时发生错误时返回的错误信息，当error_info为空，或者error_info.↩ error_id为0时，表明没有错误
is_last	是否此次取消订阅的最后一个应答，当为最后一个的时候为true，如果为false，表示还有其他后续消息响应

备注

每条取消订阅的合约均对应一条取消订阅应答，需要快速返回，否则会堵塞后续消息，当堵塞严重时，会触发断线

被 DemoTestMdSpi 重载.

5.4.2.19 virtual void OnUnSubscribeAllMarketData (XTP_EXCHANGE_TYPE *exchange_id*, XTPRI * *error_info*) [inline], [virtual]

退订全市场的股票行情应答

参数

exchange_id	表示当前退订的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场
error_info	取消订阅合约时发生错误时返回的错误信息，当error_info为空，或者error_info.error_id为0时，表明没有错误

备注

需要快速返回

被 DemoTestMdSpi 重载.

5.4.2.20 virtual void OnUnSubscribeAllOptionMarketData (XTP_EXCHANGE_TYPE *exchange_id*, XTPRI * *error_info*) [inline], [virtual]

退订全市场的期权行情应答

参数

exchange_id	表示当前退订的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场
error_info	取消订阅合约时发生错误时返回的错误信息，当error_info为空，或者error_info.error_id为0时，表明没有错误

备注

需要快速返回

被 DemoTestMdSpi 重载.

5.4.2.21 virtual void OnUnSubscribeAllOptionOrderBook (XTP_EXCHANGE_TYPE *exchange_id*, XTPRI * *error_info*) [inline], [virtual]

退订全市场的期权行情订单簿应答

参数

exchange_id	表示当前退订的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场
error_info	取消订阅合约时发生错误时返回的错误信息，当error_info为空，或者error_info.error_id为0时，表明没有错误

备注

需要快速返回

被 DemoTestMdSpi 重载.

5.4.2.22 virtual void OnUnSubscribeAllOptionTickByTick (XTP_EXCHANGE_TYPE *exchange_id*, XTPRI * *error_info*) [inline], [virtual]

退订全市场的期权逐笔行情应答

参数

exchange_id	表示当前退订的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场
error_info	取消订阅合约时发生错误时返回的错误信息，当error_info为空，或者error_info.error_id为0时，表明没有错误

备注

需要快速返回

被 DemoTestMdSpi 重载.

5.4.2.23 virtual void OnUnSubscribeAllOrderBook (XTP_EXCHANGE_TYPE *exchange_id*, XTPRI * *error_info*) [inline], [virtual]

退订全市场的股票行情订单簿应答

参数

exchange_id	表示当前退订的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场
error_info	取消订阅合约时发生错误时返回的错误信息，当error_info为空，或者error_info.error_id为0时，表明没有错误

备注

需要快速返回

被 DemoTestMdSpi 重载.

5.4.2.24 virtual void OnUnSubscribeAllTickByTick (XTP_EXCHANGE_TYPE *exchange_id*, XTPRI * *error_info*) [inline], [virtual]

退订全市场的股票逐笔行情应答

参数

exchange_id	表示当前退订的市场，如果为XTP_EXCHANGE_UNKNOWN，表示沪深全市场，XTP_EXCHANGE_SH表示为上海全市场，XTP_EXCHANGE_SZ表示为深圳全市场
error_info	取消订阅合约时发生错误时返回的错误信息，当error_info为空，或者error_info.error_id为0时，表明没有错误

备注

需要快速返回

被 DemoTestMdSpi 重载.

5.4.2.25 virtual void OnUnSubTickByTick (XTPST * *ticker*, XTPRI * *error_info*, bool *is_last*) [inline], [virtual]

退订逐笔行情应答，包括股票、指数和期权

参数

ticker	详细的合约取消订阅情况
error_info	取消订阅合约时发生错误时返回的错误信息，当error_info为空，或者error_info.↩ error_id为0时，表明没有错误
is_last	是否此次取消订阅的最后一个应答，当为最后一个的时候为true，如果为false，表示还有其他后续消息响应

备注

每条取消订阅的合约均对应一条取消订阅应答，需要快速返回，否则会堵塞后续消息，当堵塞严重时，会触发断线

被 DemoTestMdSpi 重载.

该类的文档由以下文件生成:

- xtp_quote_api.h

5.5 XTPFundTransferNotice结构体 参考

资金内转流水通知

```
#include <xoms_api_struct.h>
```

成员变量

- uint64_t serial_id
  资金内转编号
- XTP_FUND_TRANSFER_TYPE transfer_type
  内转类型
- double amount
  金额
- XTP_FUND_OPER_STATUS oper_status
  操作结果
- uint64_t transfer_time
  操作时间

5.5.1 详细描述

资金内转流水通知

该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.6 XTPFundTransferReq结构体 参考

用户资金请求

```
#include <xoms_api_fund_struct.h>
```

成员变量

- uint64_t serial_id
  资金内转编号，无需用户填写，类似于*xtp_id*
- char fund_account [XTP_ACCOUNT_NAME_LEN]
  资金账户代码
- char password [XTP_ACCOUNT_PASSWORD_LEN]
  资金账户密码
- double amount
  金额
- XTP_FUND_TRANSFER_TYPE transfer_type
  内转类型

5.6.1 详细描述

用户资金请求

该结构体的文档由以下文件生成:

- xoms_api_fund_struct.h

5.7 XTPMarketDataOptionExData结构体 参考

期权额外数据

```
#include <xquote_api_struct.h>
```

成员变量

- double auction_price
  波段性中断参考价*(SH)*
- int64_t auction_qty
  波段性中断集合竞价虚拟匹配量*(SH)*
- int64_t last_enquiry_time
  最近询价时间*(SH)*

5.7.1 详细描述

期权额外数据

该结构体的文档由以下文件生成:

- xquote_api_struct.h

5.8 XTPMarketDataStockExData结构体 参考

股票、基金、债券等额外数据

```
#include <xquote_api_struct.h>
```

成员变量

- int64_t total_bid_qty
  委托买入总量*(SH,SZ)*
- int64_t total_ask_qty
  委托卖出总量*(SH,SZ)*
- double ma_bid_price
  加权平均委买价格*(SH,SZ)*
- double ma_ask_price
  加权平均委卖价格*(SH,SZ)*
- double ma_bond_bid_price
  债券加权平均委买价格*(SH)*
- double ma_bond_ask_price
  债券加权平均委卖价格*(SH)*
- double yield_to_maturity
  债券到期收益率*(SH)*
- double iopv
  基金实时参考净值*(SH,SZ)*
- int32_t etf_buy_count
  *ETF*申购笔数*(SH)*
- int32_t etf_sell_count
  *ETF*赎回笔数*(SH)*
- double etf_buy_qty
  *ETF*申购数量*(SH)*
- double etf_buy_money
  *ETF*申购金额*(SH)*
- double etf_sell_qty
  *ETF*赎回数量*(SH)*
- double etf_sell_money
  *ETF*赎回金额*(SH)*
- double total_warrant_exec_qty
  权证执行的总数量*(SH)*
- double warrant_lower_price
  权证跌停价格（元）*(SH)*
- double warrant_upper_price
  权证涨停价格（元）*(SH)*
- int32_t cancel_buy_count
  买入撤单笔数*(SH)*
- int32_t cancel_sell_count
  卖出撤单笔数*(SH)*
- double cancel_buy_qty
  买入撤单数量*(SH)*
- double cancel_sell_qty
  卖出撤单数量*(SH)*
- double cancel_buy_money
  买入撤单金额*(SH)*
- double cancel_sell_money
  卖出撤单金额*(SH)*
- int64_t total_buy_count
  买入总笔数*(SH)*
- int64_t total_sell_count

卖出总笔数*(SH)*

- int32_t duration_after_buy
  买入委托成交最大等待时间*(SH)*
- int32_t duration_after_sell
  卖出委托成交最大等待时间*(SH)*
- int32_t num_bid_orders
  买方委托价位数*(SH)*
- int32_t num_ask_orders
  卖方委托价位数*(SH)*
- double pre_iopv
  基金*T-1*日净值*(SZ)*
- int64_t r1
  预留
- int64_t r2
  预留

5.8.1 详细描述

股票、基金、债券等额外数据

该结构体的文档由以下文件生成:

- xquote_api_struct.h

5.9 XTPMarketDataStruct结构体 参考

行情

```
#include <xquote_api_struct.h>
```

成员变量

- XTP_EXCHANGE_TYPE exchange_id
  交易所代码
- char ticker [XTP_TICKER_LEN]
  合约代码（不包含交易所信息），不带空格，以*'\0'*结尾
- double last_price
  最新价
- double pre_close_price
  昨收盘
- double open_price
  今开盘
- double high_price
  最高价
- double low_price
  最低价
- double close_price
  今收盘
- int64_t pre_total_long_positon
  昨日持仓量*(*张*)(*目前未填写*)*
- int64_t total_long_positon

持仓量*(张)*

- double pre_settl_price
  昨日结算价
- double settl_price
  今日结算价
- double upper_limit_price
  涨停价
- double lower_limit_price
  跌停价
- double pre_delta
  预留
- double curr_delta
  预留
- int64_t data_time
  时间类，格式为*YYYYMMDDHHMMSSsss*
- int64_t qty
  数量，为总成交量（单位股，与交易所一致）
- double turnover
  成交金额，为总成交金额（单位元，与交易所一致）
- double avg_price
  当日均价*=(turnover/qty)*
- double bid [10]
  十档申买价
- double ask [10]
  十档申卖价
- int64_t bid_qty [10]
  十档申买量
- int64_t ask_qty [10]
  十档申卖量
- int64_t trades_count
  成交笔数
- char ticker_status [8]
  当前交易状态说明
- union {
  XTPMarketDataStockExData **stk**
  XTPMarketDataOptionExData **opt**
  };

  数据
- XTP_MARKETDATA_TYPE data_type
  决定了*union*是哪种数据类型
- int32_t r4
  预留

5.9.1 详细描述

行情

该结构体的文档由以下文件生成:

- xquote_api_struct.h

5.10 XTPOrderCancelInfo结构体 参考

撤单失败响应消息

`#include <xoms_api_struct.h>`

成员变量

- uint64_t order_cancel_xtp_id
  撤单*XTPID*
- uint64_t order_xtp_id
  原始订单*XTPID*

5.10.1 详细描述

撤单失败响应消息

该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.11 XTPOrderInfo结构体 参考

报单响应结构体

`#include <xoms_api_struct.h>`

成员变量

- uint64_t order_xtp_id
  *XTP*系统订单*ID*，在*XTP*系统中唯一
- uint32_t order_client_id
  报单引用，用户自定义
- uint32_t order_cancel_client_id
  报单操作引用，用户自定义（暂未使用）
- uint64_t order_cancel_xtp_id
  撤单在*XTP*系统中的*id*，在*XTP*系统中唯一
- char ticker [XTP_TICKER_LEN]
  合约代码
- XTP_MARKET_TYPE market
  交易市场
- double price
  价格
- int64_t quantity
  数量，此订单的报单数量
- XTP_PRICE_TYPE price_type
  报单价格条件
- union {
  uint32_t **u32**
  struct {
  XTP_SIDE_TYPE side
  买卖方向

```
    XTP_POSITION_EFFECT_TYPE position_effect
      开平标志
    uint8_t reserved1
      预留字段1
                uint8_t reserved2
                预留字段2
                }
              };
```

- XTP_BUSINESS_TYPE business_type
  业务类型
- int64_t qty_traded
  今成交数量，为此订单累计成交数量
- int64_t qty_left
  剩余数量，当撤单成功时，表示撤单数量
- int64_t insert_time
  委托时间，格式为*YYYYMMDDHHMMSSsss*
- int64_t update_time
  最后修改时间，格式为*YYYYMMDDHHMMSSsss*
- int64_t cancel_time
  撤销时间，格式为*YYYYMMDDHHMMSSsss*
- double trade_amount
  成交金额，为此订单的成交总金额
- char order_local_id [XTP_LOCAL_ORDER_LEN]
  本地报单编号 *OMS*生成的单号，不等同于*order_xtp_id*，为服务器传到报盘的单号
- XTP_ORDER_STATUS_TYPE order_status
  报单状态，订单响应中没有部分成交状态的推送，在查询订单结果中，会有部分成交状态
- XTP_ORDER_SUBMIT_STATUS_TYPE order_submit_status
  报单提交状态，*OMS*内部使用，用户无需关心
- TXTPOrderTypeType order_type
  报单类型

5.11.1 详细描述

报单响应结构体

该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.12 XTPOrderInsertInfo结构体 参考

新订单请求

```
#include <xoms_api_struct.h>
```

成员变量

- uint64_t order_xtp_id
  *XTP*系统订单*ID*，无需用户填写，在*XTP*系统中唯一
- uint32_t order_client_id
  报单引用，由客户自定义

- char ticker [XTP_TICKER_LEN]
  合约代码 客户端请求不带空格，以'\0'结尾
- XTP_MARKET_TYPE market
  交易市场
- double price
  价格
- double stop_price
  止损价（保留字段）
- int64_t quantity
  数量(股票单位为股，逆回购单位为张)
- XTP_PRICE_TYPE price_type
  报单价格
- union {
  uint32_t **u32**
  struct {
  XTP_SIDE_TYPE side
  买卖方向
  XTP_POSITION_EFFECT_TYPE position_effect
  开平标志
  uint8_t reserved1
  预留字段1
  uint8_t reserved2
  预留字段2
  }
  };
- XTP_BUSINESS_TYPE business_type
  业务类型

5.12.1 详细描述

新订单请求

该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.13 XTPQueryAssetRsp结构体 参考

账户资金查询响应结构体

```
#include <xoms_api_struct.h>
```

成员变量

- double total_asset
  总资产(=可用资金 + 证券资产（目前为0） + 预扣的资金)
- double buying_power
  可用资金
- double security_asset
  证券资产（保留字段，目前为0）
- double fund_buy_amount

累计买入成交证券占用资金

- double fund_buy_fee
  累计买入成交交易费用
- double fund_sell_amount
  累计卖出成交证券所得资金
- double fund_sell_fee
  累计卖出成交交易费用
- double withholding_amount
  *XTP*系统预扣的资金（包括购买卖股票时预扣的交易资金+预扣手续费）
- XTP_ACCOUNT_TYPE account_type
  账户类型
- double frozen_margin
  冻结的保证金
- double frozen_exec_cash
  行权冻结资金
- double frozen_exec_fee
  行权费用
- double pay_later
  垫付资金
- double preadva_pay
  预垫付资金
- double orig_banlance
  昨日余额
- double banlance
  当前余额
- double deposit_withdraw
  当天出入金
- double trade_netting
  当日交易资金轧差
- double captial_asset
  资金资产
- double force_freeze_amount
  强锁资金
- double preferred_amount
  可取资金
- uint64_t unknown [43-12]
  *(*保留字段*)*

5.13.1 详细描述

账户资金查询响应结构体

该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.14 XTPQueryETFBaseReq结构体 参考

```
#include <xoms_api_struct.h>
```

成员变量

- XTP_MARKET_TYPE market
  交易市场
- char ticker [XTP_TICKER_LEN]
  *ETF*买卖代码

5.14.1 详细描述

查询股票ETF合约基本情况–请求结构体, 请求参数为多条件参数:1,不填则返回所有市场的ETF合约信息。2,只填写market,返回该交易市场下结果 3,填写market及ticker参数,只返回该etf信息。

该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.15 XTPQueryETFBaseRsp结构体 参考

查询股票ETF合约基本情况–响应结构体

```
#include <xoms_api_struct.h>
```

成员变量

- XTP_MARKET_TYPE market
  交易市场
- char etf [XTP_TICKER_LEN]
  *etf*代码,买卖,申赎统一使用该代码
- char subscribe_redemption_ticker [XTP_TICKER_LEN]
  *etf*申购赎回代码
- int32_t unit
  最小申购赎回单位对应的*ETF*份数,例如上证*"50ETF"*就是*900000*
- int32_t subscribe_status
  是否允许申购,*1*-允许,*0*-禁止
- int32_t redemption_status
  是否允许赎回,*1*-允许,*0*-禁止
- double max_cash_ratio
  最大现金替代比例,小于*1*的数值 *TODO* 是否采用*double*
- double estimate_amount
  *T*日预估金额
- double cash_component
  *T-X*日现金差额
- double net_value
  基金单位净值
- double total_amount
  最小申赎单位净值总金额*=net_value∗unit*

5.15.1 详细描述

查询股票ETF合约基本情况–响应结构体

该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.16 XTPQueryETFComponentReq结构体 参考

查询股票ETF合约成分股信息–请求结构体,请求参数为:交易市场+ETF买卖代码

`#include <xoms_api_struct.h>`

成员变量

- XTP_MARKET_TYPE market
  交易市场
- char ticker [XTP_TICKER_LEN]
  *ETF*买卖代码

5.16.1 详细描述

查询股票ETF合约成分股信息–请求结构体,请求参数为:交易市场+ETF买卖代码
该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.17 XTPQueryETFComponentRsp结构体 参考

查询股票ETF合约成分股信息–响应结构体

`#include <xoms_api_struct.h>`

成员变量

- XTP_MARKET_TYPE market
  交易市场
- char ticker [XTP_TICKER_LEN]
  *ETF*代码
- char component_ticker [XTP_TICKER_LEN]
  成份股代码
- char component_name [XTP_TICKER_NAME_LEN]
  成份股名称
- int64_t quantity
  成份股数量
- XTP_MARKET_TYPE component_market
  成份股交易市场
- ETF_REPLACE_TYPE replace_type
  成份股替代标识
- double premium_ratio
  溢价比例
- double amount
  成分股替代标识为必须现金替代时候的总金额

5.17.1 详细描述

查询股票ETF合约成分股信息–响应结构体

该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.18 XTPQueryFundTransferLogReq结构体 参考

资金内转流水查询请求与响应

`#include <xoms_api_struct.h>`

成员变量

- uint64_t serial_id
  资金内转编号

5.18.1 详细描述

资金内转流水查询请求与响应

该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.19 XTPQueryIPOQuotaRsp结构体 参考

查询用户申购额度

`#include <xoms_api_struct.h>`

成员变量

- XTP_MARKET_TYPE market
  交易市场
- int32_t quantity
  可申购额度

5.19.1 详细描述

查询用户申购额度

该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.20 XTPQueryIPOTickerRsp结构体 参考

查询当日可申购新股信息

`#include <xoms_api_struct.h>`

成员变量

- XTP_MARKET_TYPE market
  交易市场
- char ticker [XTP_TICKER_LEN]
  申购代码
- char ticker_name [XTP_TICKER_NAME_LEN]
  申购股票名称
- double price
  申购价格
- int32_t unit
  申购单元
- int32_t qty_upper_limit
  最大允许申购数量

5.20.1 详细描述

查询当日可申购新股信息
该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.21 XTPQueryOptionAuctionInfoReq结构体 参考

查询期权竞价交易业务参考信息–请求结构体,请求参数为:交易市场+8位期权代码

```
#include <xoms_api_struct.h>
```

成员变量

- XTP_MARKET_TYPE market
  交易市场
- char ticker [XTP_TICKER_LEN]
  *8*位期权合约代码

5.21.1 详细描述

查询期权竞价交易业务参考信息–请求结构体,请求参数为:交易市场+8位期权代码
该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.22 XTPQueryOptionAuctionInfoRsp结构体 参考

查询期权竞价交易业务参考信息

```
#include <xoms_api_struct.h>
```

成员变量

- char ticker [XTP_TICKER_LEN]
  合约编码，报单*ticker*采用本字段
- XTP_MARKET_TYPE security_id_source
  证券代码源
- char symbol [XTP_TICKER_NAME_LEN]
  合约简称
- char contract_id [XTP_TICKER_NAME_LEN]
  合约交易代码
- char underlying_security_id [XTP_TICKER_LEN]
  基础证券代码
- XTP_MARKET_TYPE underlying_security_id_source
  基础证券代码源
- uint32_t list_date
  上市日期，格式为*YYYYMMDD*
- uint32_t last_trade_date
  最后交易日，格式为*YYYYMMDD*
- XTP_TICKER_TYPE ticker_type
  证券类别
- int32_t day_trading
  是否支持当日回转交易，*1*-允许，*0*-不允许
- XTP_OPT_CALL_OR_PUT_TYPE call_or_put
  认购或认沽
- uint32_t delivery_day
  行权交割日，格式为*YYYYMMDD*
- uint32_t delivery_month
  交割月份，格式为*YYYYMM*
- XTP_OPT_EXERCISE_TYPE_TYPE exercise_type
  行权方式
- uint32_t exercise_begin_date
  行权起始日期，格式为*YYYYMMDD*
- uint32_t exercise_end_date
  行权结束日期，格式为*YYYYMMDD*
- double exercise_price
  行权价格
- int64_t qty_unit
  数量单位，对于某一证券申报的委托，其委托数量字段必须为该证券数量单位的整数倍
- int64_t contract_unit
  合约单位
- int64_t contract_position
  合约持仓量
- double prev_close_price
  合约前收盘价
- double prev_clearing_price
  合约前结算价
- int64_t lmt_buy_max_qty
  限价买最大量
- int64_t lmt_buy_min_qty
  限价买最小量
- int64_t lmt_sell_max_qty

限价卖最大量
- int64_t lmt_sell_min_qty
  限价卖最小量
- int64_t mkt_buy_max_qty
  市价买最大量
- int64_t mkt_buy_min_qty
  市价买最小量
- int64_t mkt_sell_max_qty
  市价卖最大量
- int64_t mkt_sell_min_qty
  市价卖最小量
- double price_tick
  最小报价单位
- double upper_limit_price
  涨停价
- double lower_limit_price
  跌停价
- double sell_margin
  今卖开每张保证金
- double margin_ratio_param1
  交易所保证金比例计算参数一
- double margin_ratio_param2
  交易所保证金比例计算参数二
- uint64_t unknown [20]
  （保留字段）

5.22.1 详细描述

查询期权竞价交易业务参考信息
该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.23 XTPQueryOrderReq结构体 参考

报单查询 // 报单查询请求-条件查询

```
#include <xoms_api_struct.h>
```

成员变量

- char ticker [XTP_TICKER_LEN]
  证券代码，可以为空，如果为空，则默认查询时间段内的所有成交回报
- int64_t begin_time
  格式为*YYYYMMDDHHMMSSsss*，为*0*则默认当前交易日*0*点
- int64_t end_time
  格式为*YYYYMMDDHHMMSSsss*，为*0*则默认当前时间

5.23.1 详细描述

报单查询 // 报单查询请求-条件查询

该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.24 XTPQueryReportByExecIdReq结构体 参考

成交回报查询 // 查询成交报告请求-根据执行编号查询（保留字段）

`#include <xoms_api_struct.h>`

成员变量

- uint64_t order_xtp_id
  *XTP*订单系统*ID.*
- char exec_id [XTP_EXEC_ID_LEN]
  成交执行编号

5.24.1 详细描述

成交回报查询 // 查询成交报告请求-根据执行编号查询（保留字段）

该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.25 XTPQueryStkPositionRsp结构体 参考

查询股票持仓情况

`#include <xoms_api_struct.h>`

成员变量

- char ticker [XTP_TICKER_LEN]
  证券代码
- char ticker_name [XTP_TICKER_NAME_LEN]
  证券名称
- XTP_MARKET_TYPE market
  交易市场
- int64_t total_qty
  总持仓
- int64_t sellable_qty
  可卖持仓
- double avg_price
  持仓成本
- double unrealized_pnl

浮动盈亏（保留字段）

- int64_t yesterday_position
  昨日持仓
- int64_t purchase_redeemable_qty
  今日申购赎回数量（申购和赎回数量不可能同时存在，因此可以共用一个字段）
- XTP_POSITION_DIRECTION_TYPE position_direction
  持仓方向
- uint32_t reserved1
  保留字段*1*
- int64_t executable_option
  可行权合约
- int64_t lockable_position
  可锁定标的
- int64_t executable_underlying
  可行权标的
- int64_t locked_position
  已锁定标的
- int64_t usable_locked_position
  可用已锁定标的
- uint64_t unknown [50-6]
  *(*保留字段*)*

5.25.1 详细描述

查询股票持仓情况

该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.26 XTPQueryStructuredFundInfoReq结构体 参考

查询分级基金信息结构体

`#include <xoms_api_struct.h>`

成员变量

- XTP_EXCHANGE_TYPE exchange_id
  交易所代码，不可为空
- char sf_ticker [XTP_TICKER_LEN]
  分级基金母基金代码，可以为空，如果为空，则默认查询所有的分级基金

5.26.1 详细描述

查询分级基金信息结构体

该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.27 XTPQueryTraderReq结构体 参考

查询成交回报请求-查询条件

`#include <xoms_api_struct.h>`

成员变量

- char ticker [XTP_TICKER_LEN]
  证券代码，可以为空，如果为空，则默认查询时间段内的所有成交回报
- int64_t begin_time
  开始时间，格式为*YYYYMMDDHHMMSSsss*，为*0*则默认当前交易日*0*点
- int64_t end_time
  结束时间，格式为*YYYYMMDDHHMMSSsss*，为*0*则默认当前时间

5.27.1 详细描述

查询成交回报请求-查询条件

该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.28 XTPQuoteStaticInfo结构体 参考

股票行情静态信息

`#include <xquote_api_struct.h>`

成员变量

- XTP_EXCHANGE_TYPE exchange_id
  交易所代码
- char ticker [XTP_TICKER_LEN]
  合约代码（不包含交易所信息），不带空格，以*'\0'*结尾
- char ticker_name [XTP_TICKER_NAME_LEN]
  合约名称
- XTP_TICKER_TYPE ticker_type
  合约类型
- double pre_close_price
  昨收盘
- double upper_limit_price
  涨停板价
- double lower_limit_price
  跌停板价
- double price_tick
  最小变动价位
- int32_t buy_qty_unit
  合约最小交易量*(买)*
- int32_t sell_qty_unit
  合约最小交易量*(卖)*

5.28.1 详细描述

股票行情静态信息

该结构体的文档由以下文件生成:

- xquote_api_struct.h

5.29 XTPRspInfoStruct结构体 参考

响应信息

`#include <xtp_api_struct_common.h>`

成员变量

- int32_t error_id
  错误代码
- char error_msg [XTP_ERR_MSG_LEN]
  错误信息

5.29.1 详细描述

响应信息

该结构体的文档由以下文件生成:

- xtp_api_struct_common.h

5.30 XTPSpecificTickerStruct结构体 参考

指定的合约

`#include <xquote_api_struct.h>`

成员变量

- XTP_EXCHANGE_TYPE exchange_id
  交易所代码
- char ticker [XTP_TICKER_LEN]
  合约代码（不包含交易所信息）例如*"600000"*，不带空格，以*'\0'*结尾

5.30.1 详细描述

指定的合约

该结构体的文档由以下文件生成:

- xquote_api_struct.h

5.31 XTPStructuredFundInfo结构体 参考

查询分级基金信息响应结构体

`#include <xoms_api_struct.h>`

成员变量

- XTP_EXCHANGE_TYPE exchange_id
  交易所代码
- char sf_ticker [XTP_TICKER_LEN]
  分级基金母基金代码
- char sf_ticker_name [XTP_TICKER_NAME_LEN]
  分级基金母基金名称
- char ticker [XTP_TICKER_LEN]
  分级基金子基金代码
- char ticker_name [XTP_TICKER_NAME_LEN]
  分级基金子基金名称
- XTP_SPLIT_MERGE_STATUS split_merge_status
  基金允许拆分合并状态
- uint32_t ratio
  拆分合并比例
- uint32_t min_split_qty
  最小拆分数量
- uint32_t min_merge_qty
  最小合并数量
- double net_price
  基金净值

5.31.1 详细描述

查询分级基金信息响应结构体
该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.32 XTPTickByTickEntrust结构体 参考

逐笔委托(仅适用深交所)

`#include <xquote_api_struct.h>`

成员变量

- int32_t channel_no
  频道代码
- int64_t seq
  委托序号*(*在同一个*channel_no*内唯一，从*1*开始连续*)*
- double price
  委托价格

- int64_t qty
  委托数量
- char side
  '1':买; *'2'*:卖; *'G'*:借入; *'F'*:出借
- char ord_type
  订单类别: *'1'*: 市价; *'2'*: 限价; *'U'*: 本方最优

5.32.1 详细描述

逐笔委托(仅适用深交所)

该结构体的文档由以下文件生成:

- xquote_api_struct.h

5.33 XTPTickByTickStruct结构体 参考

逐笔数据信息

```
#include <xquote_api_struct.h>
```

成员变量

- XTP_EXCHANGE_TYPE exchange_id
  交易所代码
- char ticker [XTP_TICKER_LEN]
  合约代码（不包含交易所信息），不带空格，以*'\0'*结尾
- int64_t seq
  预留
- int64_t data_time
  委托时间 *or* 成交时间
- XTP_TBT_TYPE type
  委托 *or* 成交
- union {
  XTPTickByTickEntrust **entrust**
  XTPTickByTickTrade **trade**
  };

5.33.1 详细描述

逐笔数据信息

该结构体的文档由以下文件生成:

- xquote_api_struct.h

5.34 XTPTickByTickTrade结构体 参考

逐笔成交

```
#include <xquote_api_struct.h>
```

成员变量

- int32_t channel_no
  频道代码
- int64_t seq
  委托序号*(*在同一个*channel_no*内唯一，从*1*开始连续*)*
- double price
  成交价格
- int64_t qty
  成交量
- double money
  成交金额*(*仅适用上交所*)*
- int64_t bid_no
  买方订单号
- int64_t ask_no
  卖方订单号
- char trade_flag

5.34.1 详细描述

逐笔成交

5.34.2 结构体成员变量说明

5.34.2.1 char trade_flag

SH: 内外盘标识('B':主动买; 'S':主动卖; 'N':未知) SZ: 成交标识('4':撤; 'F':成交)

该结构体的文档由以下文件生成:

- xquote_api_struct.h

5.35 XTPTickerPriceInfo结构体 参考

供查询的最新信息

`#include <xquote_api_struct.h>`

成员变量

- XTP_EXCHANGE_TYPE exchange_id
  交易所代码
- char ticker [XTP_TICKER_LEN]
  合约代码（不包含交易所信息），不带空格，以*'\0'*结尾
- double last_price
  最新价

5.35.1 详细描述

供查询的最新信息

该结构体的文档由以下文件生成:

- xquote_api_struct.h

5.36 XTPTradeReport结构体 参考

报单成交结构体

`#include <xoms_api_struct.h>`

成员变量

- uint64_t order_xtp_id
  *XTP*系统订单*ID*，此成交回报相关的订单*ID*，在*XTP*系统中唯一
- uint32_t order_client_id
  报单引用
- char ticker [XTP_TICKER_LEN]
  合约代码
- XTP_MARKET_TYPE market
  交易市场
- uint64_t local_order_id
  订单号，引入*XTPID*后，该字段实际和*order_xtp_id*重复。接口中暂时保留。
- char exec_id [XTP_EXEC_ID_LEN]
  成交编号，深交所唯一，上交所每笔交易唯一，当发现*2*笔成交回报拥有相同的*exec_id*，则可以认为此笔交易自成交
- double price
  价格，此次成交的价格
- int64_t quantity
  数量，此次成交的数量，不是累计数量
- int64_t trade_time
  成交时间，格式为*YYYYMMDDHHMMSSsss*
- double trade_amount
  成交金额，此次成交的总金额 = *price*∗*quantity*
- uint64_t report_index
  成交序号 –回报记录号，每个交易所唯一,*report_index+market*字段可以组成唯一标识表示成交回报
- char order_exch_id [XTP_ORDER_EXCH_LEN]
  报单编号 –交易所单号，上交所为空，深交所有此字段
- TXTPTradeTypeType trade_type
  成交类型 –成交回报中的执行类型
- union {
  uint32_t **u32**
  struct {
  XTP_SIDE_TYPE side
  买卖方向
  XTP_POSITION_EFFECT_TYPE position_effect
  开平标志
  uint8_t reserved1
  预留字段*1*
  uint8_t reserved2
  预留字段*2*
  }
  };
- XTP_BUSINESS_TYPE business_type
  业务类型
- char branch_pbu [XTP_BRANCH_PBU_LEN]
  交易所交易员代码

5.36.1 详细描述

报单成交结构体

该结构体的文档由以下文件生成:

- xoms_api_struct.h

Chapter 6

文件说明

6.1 demo_test_quote_api.cpp 文件参考

定义控制台测试应用程序的入口点

```
#include <string>
#include <map>
#include <iostream>
#include "xtp_quote_api.h"
#include "demo_test_quote_spi.h"
```

函数

- int main ()

  int main() {

变量

- XTP::API::QuoteApi * user_api_pointer

  *UserApi*对象
- char username [ ] = "00092"

  投资者代码
- char password [ ] = "888888"

  用户密码
- char * ppTicker [ ] = { "000001" }

  行情订阅列表
- int ticker_count = 1
- int client_id = 1

  客户端标识
- char filepath [ ] = "c:\\log\\"

  可读写存储路径

6.1.1 详细描述

定义控制台测试应用程序的入口点

作者

中泰证券股份有限公司

6.1.2 函数说明

6.1.2.1 int main ( )

int main() {

测试Demo入口函数
// 初始化UserApi

user_api_pointer = XTP::API::QuoteApi::CreateQuoteApi(client_id,filepath); // 创建UserApi

user_spi_pointer->SetHeartBeatInterval(15);//设置心跳超时时间间隔，单位为秒

user_api_pointer->SetUDPBufferSize(128);//设置UDP接收缓冲区大小，单位为MB

DemoTestMdSpi* user_spi_pointer = new DemoTestMdSpi(); //创建响应类实例

user_api_pointer->RegisterSpi(user_spi_pointer); // 注册事件类

int loginResult = user_api_pointer->Login("127.0.0.1", 6666, username, password, XTP_PROTOCOL_UDP);//采用UDP方式登陆行情订阅服务器

if (loginResult == 0)

{ //登录成功

user_api_pointer->SubscribeMarketData(ppTicker, ticker_count, XTP_EXCHANGE_SZ);//订阅股票行情

}

return 0;

}

6.1.3 变量说明

6.1.3.1 int ticker_count = 1

行情订阅数量

6.2 demo_test_quote_spi.h 文件参考

Demo自定义客户端行情订阅响应接口类

```
#include "xtp_quote_api.h"
```

结构体

- class DemoTestMdSpi

  *Demo*自定义行情订阅接口响应类

6.2.1 详细描述

Demo自定义客户端行情订阅响应接口类

作者

中泰证券股份有限公司

6.3 xoms_api_fund_struct.h 文件参考

定义资金划拨相关结构体类型

```
#include "xtp_api_data_type.h"
#include "xoms_api_struct.h"
#include "xtp_api_struct_common.h"
```

结构体

- struct XTPFundTransferReq
  用户资金请求

宏定义

- #define XTP_ACCOUNT_PASSWORD_LEN 64
  用户资金账户的密码字符串长度

类型定义

- typedef struct XTPFundTransferNotice XTPFundTransferAck
  用户资金划转请求的响应-复用资金通知结构体

6.3.1 详细描述

定义资金划拨相关结构体类型

作者

中泰证券股份有限公司

6.4 xoms_api_struct.h 文件参考

定义交易类相关数据结构

```
#include "xtp_api_data_type.h"
#include "stddef.h"
```

结构体

- struct XTPOrderInsertInfo
  新订单请求
- struct XTPOrderCancelInfo
  撤单失败响应消息
- struct XTPOrderInfo

报单响应结构体

- struct XTPTradeReport
  报单成交结构体
- struct XTPQueryOrderReq
  报单查询 // 报单查询请求-条件查询
- struct XTPQueryReportByExecIdReq
  成交回报查询 // 查询成交报告请求-根据执行编号查询（保留字段）
- struct XTPQueryTraderReq
  查询成交回报请求-查询条件
- struct XTPQueryAssetRsp
  账户资金查询响应结构体
- struct XTPQueryStkPositionRsp
  查询股票持仓情况
- struct XTPFundTransferNotice
  资金内转流水通知
- struct XTPQueryFundTransferLogReq
  资金内转流水查询请求与响应
- struct XTPQueryStructuredFundInfoReq
  查询分级基金信息结构体
- struct XTPStructuredFundInfo
  查询分级基金信息响应结构体
- struct XTPQueryETFBaseReq
- struct XTPQueryETFBaseRsp
  查询股票*ETF*合约基本情况–响应结构体
- struct XTPQueryETFComponentReq
  查询股票*ETF*合约成分股信息–请求结构体,请求参数为:交易市场*+ETF*买卖代码
- struct XTPQueryETFComponentRsp
  查询股票*ETF*合约成分股信息–响应结构体
- struct XTPQueryIPOTickerRsp
  查询当日可申购新股信息
- struct XTPQueryIPOQuotaRsp
  查询用户申购额度
- struct XTPQueryOptionAuctionInfoReq
  查询期权竞价交易业务参考信息–请求结构体,请求参数为:交易市场*+8*位期权代码
- struct XTPQueryOptionAuctionInfoRsp
  查询期权竞价交易业务参考信息

类型定义

- typedef struct XTPOrderInfo XTPQueryOrderRsp
  报单查询响应结构体
- typedef struct XTPTradeReport XTPQueryTradeRsp
  成交回报查询响应结构体
- typedef struct XTPFundTransferNotice XTPFundTransferLog
  资金内转流水记录结构体
- typedef struct XTPQueryETFBaseRsp XTPQueryETFBaseRsp
  查询股票*ETF*合约基本情况–响应结构体
- typedef struct XTPQueryETFComponentReq XTPQueryETFComponentReq
  查询股票*ETF*合约成分股信息–请求结构体,请求参数为:交易市场*+ETF*买卖代码

6.4.1 详细描述

定义交易类相关数据结构

作者

中泰证券股份有限公司

6.5 xquote_api_struct.h 文件参考

定义行情类相关数据结构

```
#include <stdint.h>
#include "xtp_api_data_type.h"
```

结构体

- struct XTPSpecificTickerStruct
  指定的合约
- struct XTPMarketDataStockExData
  股票、基金、债券等额外数据
- struct XTPMarketDataOptionExData
  期权额外数据
- struct XTPMarketDataStruct
  行情
- struct XTPQuoteStaticInfo
  股票行情静态信息
- struct OrderBookStruct
  定单薄
- struct XTPTickByTickEntrust
  逐笔委托*(*仅适用深交所*)*
- struct XTPTickByTickTrade
  逐笔成交
- struct XTPTickByTickStruct
  逐笔数据信息
- struct XTPTickerPriceInfo
  供查询的最新信息

类型定义

- typedef struct XTPSpecificTickerStruct XTPST
  指定的合约
- typedef struct XTPMarketDataStruct XTPMD
  行情
- typedef struct XTPQuoteStaticInfo XTPQSI
  股票行情静态信息
- typedef struct OrderBookStruct XTPOB
  定单薄
- typedef struct XTPTickByTickStruct XTPTBT
  逐笔数据信息
- typedef struct XTPTickerPriceInfo XTPTPI
  供查询的最新信息

枚举

- enum XTP_MARKETDATA_TYPE { XTP_MARKETDATA_ACTUAL = 0, XTP_MARKETDATA_OPTION = 1 }
  *XTP_MARKETDATA_TYPE*是行情快照数据类型

6.5.1 详细描述

定义行情类相关数据结构

作者

中泰证券股份有限公司

6.5.2 枚举类型说明

6.5.2.1 enum XTP_MARKETDATA_TYPE

XTP_MARKETDATA_TYPE是行情快照数据类型

枚举值

XTP_MARKETDATA_ACTUAL 现货(股票/基金/债券等)
XTP_MARKETDATA_OPTION 期权

6.6 xtp_api_data_type.h 文件参考

定义兼容数据基本类型

宏定义

- #define XTP_VERSION_LEN 16
  存放版本号的字符串长度
- #define XTP_TRADING_DAY_LEN 9
  可交易日字符串长度
- #define XTP_TICKER_LEN 16
  存放证券代码的字符串长度
- #define XTP_TICKER_NAME_LEN 64
  存放证券名称的字符串长度
- #define XTP_LOCAL_ORDER_LEN 11
  本地报单编号的字符串长度
- #define XTP_ORDER_EXCH_LEN 17
  交易所单号的字符串长度
- #define XTP_EXEC_ID_LEN 18
  成交执行编号的字符串长度
- #define XTP_BRANCH_PBU_LEN 7
  交易所交易员代码字符串长度
- #define XTP_ACCOUNT_NAME_LEN 16
  用户资金账户的字符串长度
- #define XTP_SIDE_BUY 1
  买（新股申购，*ETF*买，配股，信用交易中担保品买）

- #define XTP_SIDE_SELL 2
  卖（逆回购，*ETF*卖，信用交易中担保品卖）
- #define XTP_SIDE_PURCHASE 7
  申购
- #define XTP_SIDE_REDEMPTION 8
  赎回
- #define XTP_SIDE_SPLIT 9
  拆分
- #define XTP_SIDE_MERGE 10
  合并
- #define XTP_SIDE_COVER 11
  改版之后的*side*的备兑，暂不支持
- #define XTP_SIDE_FREEZE 12
  改版之后的*side*锁定（对应开平标识为开）/解锁（对应开平标识为平）
- #define XTP_SIDE_MARGIN_TRADE 21
  融资买入
- #define XTP_SIDE_SHORT_SELL 22
  融券卖出
- #define XTP_SIDE_REPAY_MARGIN 23
  卖券还款
- #define XTP_SIDE_REPAY_STOCK 24
  买券还券
- #define XTP_SIDE_CASH_REPAY_MARGIN 25
  现金还款
- #define XTP_SIDE_STOCK_REPAY_STOCK 26
  现券还券
- #define XTP_SIDE_UNKNOWN 27
  未知或者无效买卖方向
- #define XTP_POSITION_EFFECT_INIT 0
  初始值或未知值开平标识，现货适用
- #define XTP_POSITION_EFFECT_OPEN 1
  开
- #define XTP_POSITION_EFFECT_CLOSE 2
  平
- #define XTP_POSITION_EFFECT_FORCECLOSE 3
  强平
- #define XTP_POSITION_EFFECT_CLOSETODAY 4
  平今
- #define XTP_POSITION_EFFECT_CLOSEYESTERDAY 5
  平昨
- #define XTP_POSITION_EFFECT_FORCEOFF 6
  强减
- #define XTP_POSITION_EFFECT_LOCALFORCECLOSE 7
  本地强平
- #define XTP_POSITION_EFFECT_UNKNOWN 8
  未知的开平标识类型
- #define XTP_TRDT_COMMON '0'
  普通成交
- #define XTP_TRDT_CASH '1'
  现金替代
- #define XTP_TRDT_PRIMARY '2'

一级市场成交

- #define XTP_ORDT_Normal '0'
  正常
- #define XTP_ORDT_DeriveFromQuote '1'
  报价衍生
- #define XTP_ORDT_DeriveFromCombination '2'
  组合衍生
- #define XTP_ORDT_Combination '3'
  组合报单
- #define XTP_ORDT_ConditionalOrder '4'
  条件单
- #define XTP_ORDT_Swap '5'
  互换单

类型定义

- typedef char XTPVersionType[XTP_VERSION_LEN]
  版本号类型
- typedef enum XTP_LOG_LEVEL XTP_LOG_LEVEL
  *XTP_LOG_LEVEL*是日志输出级别类型
- typedef enum XTP_PROTOCOL_TYPE XTP_PROTOCOL_TYPE
  *XTP_PROTOCOL_TYPE*是通讯传输协议方式
- typedef enum XTP_EXCHANGE_TYPE XTP_EXCHANGE_TYPE
  *XTP_EXCHANGE_TYPE*是交易所类型
- typedef enum XTP_MARKET_TYPE XTP_MARKET_TYPE
  *XTP_MARKET_TYPE*市场类型
- typedef enum XTP_PRICE_TYPE XTP_PRICE_TYPE
  *XTP_PRICE_TYPE*是价格类型
- typedef uint8_t XTP_SIDE_TYPE
  *XTP_SIDE_TYPE*是买卖方向类型
- typedef uint8_t XTP_POSITION_EFFECT_TYPE
  *XTP_POSITION_EFFECT_TYPE*是开平标识类型
- typedef enum XTP_ORDER_ACTION_STATUS_TYPE XTP_ORDER_ACTION_STATUS_TYPE
  *XTP_ORDER_ACTION_STATUS_TYPE*是报单操作状态类型
- typedef enum XTP_ORDER_STATUS_TYPE XTP_ORDER_STATUS_TYPE
  *XTP_ORDER_STATUS_TYPE*是报单状态类型
- typedef enum XTP_ORDER_SUBMIT_STATUS_TYPE XTP_ORDER_SUBMIT_STATUS_TYPE
  *XTP_ORDER_SUBMIT_STATUS_TYPE*是报单提交状态类型
- typedef enum XTP_TE_RESUME_TYPE XTP_TE_RESUME_TYPE
  *XTP_TE_RESUME_TYPE*是公有流（订单响应、成交回报）重传方式
- typedef enum ETF_REPLACE_TYPE ETF_REPLACE_TYPE
  *ETF_REPLACE_TYPE*现金替代标识定义
- typedef enum XTP_TICKER_TYPE XTP_TICKER_TYPE
  *XTP_TICKER_TYPE*证券类型
- typedef enum XTP_BUSINESS_TYPE XTP_BUSINESS_TYPE
  *XTP_BUSINESS_TYPE*证券业务类型
- typedef enum XTP_ACCOUNT_TYPE XTP_ACCOUNT_TYPE
  *XTP_ACCOUNT_TYPE*账户类型
- typedef enum XTP_FUND_TRANSFER_TYPE XTP_FUND_TRANSFER_TYPE
  *XTP_FUND_TRANSFER_TYPE*是资金流转方向类型

- typedef enum XTP_FUND_OPER_STATUS XTP_FUND_OPER_STATUS
  *XTP_FUND_OPER_STATUS*柜台资金操作结果
- typedef enum XTP_SPLIT_MERGE_STATUS XTP_SPLIT_MERGE_STATUS
  *XTP_SPLIT_MERGE_STATUS*是一个基金当天拆分合并状态类型
- typedef enum XTP_TBT_TYPE XTP_TBT_TYPE
  *XTP_TBT_TYPE*是一个逐笔回报类型
- typedef enum XTP_OPT_CALL_OR_PUT_TYPE XTP_OPT_CALL_OR_PUT_TYPE
  *XTP_OPT_CALL_OR_PUT_TYPE*是一个认沽或认购类型
- typedef enum XTP_OPT_EXERCISE_TYPE_TYPE XTP_OPT_EXERCISE_TYPE_TYPE
  *XTP_OPT_EXERCISE_TYPE_TYPE*是一个行权方式类型
- typedef enum XTP_POSITION_DIRECTION_TYPE XTP_POSITION_DIRECTION_TYPE
  *XTP_POSITION_DIRECTION_TYPE*是一个持仓方向类型
- typedef char TXTPTradeTypeType
  *TXTPTradeTypeType*是成交类型类型
- typedef char TXTPOrderTypeType
  *TXTPOrderTypeType*是报单类型类型

枚举

- enum XTP_LOG_LEVEL {
  XTP_LOG_LEVEL_FATAL, XTP_LOG_LEVEL_ERROR, XTP_LOG_LEVEL_WARNING, XTP_LOG_LEV↩
  EL_INFO,
  XTP_LOG_LEVEL_DEBUG, XTP_LOG_LEVEL_TRACE }
  *XTP_LOG_LEVEL*是日志输出级别类型
- enum XTP_PROTOCOL_TYPE { XTP_PROTOCOL_TCP = 1, XTP_PROTOCOL_UDP }
  *XTP_PROTOCOL_TYPE*是通讯传输协议方式
- enum XTP_EXCHANGE_TYPE { XTP_EXCHANGE_SH = 1, XTP_EXCHANGE_SZ, XTP_EXCHANGE_↩
  UNKNOWN }
  *XTP_EXCHANGE_TYPE*是交易所类型
- enum XTP_MARKET_TYPE { XTP_MKT_INIT = 0, XTP_MKT_SZ_A = 1, XTP_MKT_SH_A, XTP_MKT_U↩
  NKNOWN }
  *XTP_MARKET_TYPE*市场类型
- enum XTP_PRICE_TYPE {
  XTP_PRICE_LIMIT = 1, XTP_PRICE_BEST_OR_CANCEL, XTP_PRICE_BEST5_OR_LIMIT, XTP_PRIC↩
  E_BEST5_OR_CANCEL,
  XTP_PRICE_ALL_OR_CANCEL, XTP_PRICE_FORWARD_BEST, XTP_PRICE_REVERSE_BEST_LIMIT,
  XTP_PRICE_LIMIT_OR_CANCEL,
  XTP_PRICE_TYPE_UNKNOWN }
  *XTP_PRICE_TYPE*是价格类型
- enum XTP_ORDER_ACTION_STATUS_TYPE { XTP_ORDER_ACTION_STATUS_SUBMITTED = 1, XT↩
  P_ORDER_ACTION_STATUS_ACCEPTED, XTP_ORDER_ACTION_STATUS_REJECTED }
  *XTP_ORDER_ACTION_STATUS_TYPE*是报单操作状态类型
- enum XTP_ORDER_STATUS_TYPE {
  XTP_ORDER_STATUS_INIT = 0, XTP_ORDER_STATUS_ALLTRADED = 1, XTP_ORDER_STATUS_P↩
  ARTTRADEDQUEUEING, XTP_ORDER_STATUS_PARTTRADEDNOTQUEUEING,
  XTP_ORDER_STATUS_NOTRADEQUEUEING, XTP_ORDER_STATUS_CANCELED, XTP_ORDER_S↩
  TATUS_REJECTED, XTP_ORDER_STATUS_UNKNOWN }
  *XTP_ORDER_STATUS_TYPE*是报单状态类型
- enum XTP_ORDER_SUBMIT_STATUS_TYPE {
  XTP_ORDER_SUBMIT_STATUS_INSERT_SUBMITTED = 1, XTP_ORDER_SUBMIT_STATUS_INSERT↩
  _ACCEPTED, XTP_ORDER_SUBMIT_STATUS_INSERT_REJECTED, XTP_ORDER_SUBMIT_STATU↩
  S_CANCEL_SUBMITTED,
  XTP_ORDER_SUBMIT_STATUS_CANCEL_REJECTED, XTP_ORDER_SUBMIT_STATUS_CANCEL_A↩
  CCEPTED }

*XTP_ORDER_SUBMIT_STATUS_TYPE*是报单提交状态类型

- enum XTP_TE_RESUME_TYPE { XTP_TERT_RESTART = 0, XTP_TERT_RESUME, XTP_TERT_QUICK }

  *XTP_TE_RESUME_TYPE*是公有流（订单响应、成交回报）重传方式

- enum ETF_REPLACE_TYPE {
  ERT_CASH_FORBIDDEN = 0, ERT_CASH_OPTIONAL, ERT_CASH_MUST, ERT_CASH_RECOMPUT↩
  E_INTER_SZ,
  ERT_CASH_MUST_INTER_SZ, ERT_CASH_RECOMPUTE_INTER_OTHER, ERT_CASH_MUST_INTE↩
  R_OTHER, EPT_INVALID }

  *ETF_REPLACE_TYPE*现金替代标识定义

- enum XTP_TICKER_TYPE {
  XTP_TICKER_TYPE_STOCK = 0, XTP_TICKER_TYPE_INDEX, XTP_TICKER_TYPE_FUND, XTP_TIC↩
  KER_TYPE_BOND,
  XTP_TICKER_TYPE_OPTION, XTP_TICKER_TYPE_UNKNOWN }

  *XTP_TICKER_TYPE*证券类型

- enum XTP_BUSINESS_TYPE {
  XTP_BUSINESS_TYPE_CASH = 0, XTP_BUSINESS_TYPE_IPOS, XTP_BUSINESS_TYPE_REPO, XT↩
  P_BUSINESS_TYPE_ETF,
  XTP_BUSINESS_TYPE_MARGIN, XTP_BUSINESS_TYPE_DESIGNATION, XTP_BUSINESS_TYPE_A↩
  LLOTMENT, XTP_BUSINESS_TYPE_STRUCTURED_FUND_PURCHASE_REDEMPTION,
  XTP_BUSINESS_TYPE_STRUCTURED_FUND_SPLIT_MERGE, XTP_BUSINESS_TYPE_MONEY_FU↩
  ND, XTP_BUSINESS_TYPE_OPTION, XTP_BUSINESS_TYPE_EXECUTE,
  XTP_BUSINESS_TYPE_FREEZE, XTP_BUSINESS_TYPE_UNKNOWN }

  *XTP_BUSINESS_TYPE*证券业务类型

- enum XTP_ACCOUNT_TYPE { XTP_ACCOUNT_NORMAL = 0, XTP_ACCOUNT_CREDIT, XTP_ACCO↩
  UNT_DERIVE, XTP_ACCOUNT_UNKNOWN }

  *XTP_ACCOUNT_TYPE*账户类型

- enum XTP_FUND_TRANSFER_TYPE { XTP_FUND_TRANSFER_OUT = 0, XTP_FUND_TRANSFER_IN,
  XTP_FUND_TRANSFER_UNKNOWN }

  *XTP_FUND_TRANSFER_TYPE*是资金流转方向类型

- enum XTP_FUND_OPER_STATUS {
  XTP_FUND_OPER_PROCESSING = 0, XTP_FUND_OPER_SUCCESS, XTP_FUND_OPER_FAILED, X↩
  TP_FUND_OPER_SUBMITTED,
  XTP_FUND_OPER_UNKNOWN }

  *XTP_FUND_OPER_STATUS*柜台资金操作结果

- enum XTP_SPLIT_MERGE_STATUS { XTP_SPLIT_MERGE_STATUS_ALLOW = 0, XTP_SPLIT_MER↩
  GE_STATUS_ONLY_SPLIT, XTP_SPLIT_MERGE_STATUS_ONLY_MERGE, XTP_SPLIT_MERGE_ST↩
  ATUS_FORBIDDEN }

  *XTP_SPLIT_MERGE_STATUS*是一个基金当天拆分合并状态类型

- enum XTP_TBT_TYPE { XTP_TBT_ENTRUST = 1, XTP_TBT_TRADE = 2 }

  *XTP_TBT_TYPE*是一个逐笔回报类型

- enum XTP_OPT_CALL_OR_PUT_TYPE { XTP_OPT_CALL = 1, XTP_OPT_PUT = 2 }

  *XTP_OPT_CALL_OR_PUT_TYPE*是一个认沽或认购类型

- enum XTP_OPT_EXERCISE_TYPE_TYPE { XTP_OPT_EXERCISE_TYPE_EUR = 1, XTP_OPT_EXERC↩
  ISE_TYPE_AME = 2 }

  *XTP_OPT_EXERCISE_TYPE_TYPE*是一个行权方式类型

- enum XTP_POSITION_DIRECTION_TYPE { XTP_POSITION_DIRECTION_NET = 0, XTP_POSITION_D↩
  IRECTION_LONG, XTP_POSITION_DIRECTION_SHORT, XTP_POSITION_DIRECTION_COVERED }

  *XTP_POSITION_DIRECTION_TYPE*是一个持仓方向类型

6.6.1 详细描述

定义兼容数据基本类型

作者

中泰证券股份有限公司

6.6.2 枚举类型说明

6.6.2.1 enum ETF_REPLACE_TYPE

ETF_REPLACE_TYPE现金替代标识定义

枚举值

ERT_CASH_FORBIDDEN 禁止现金替代
ERT_CASH_OPTIONAL 可以现金替代
ERT_CASH_MUST 必须现金替代
ERT_CASH_RECOMPUTE_INTER_SZ 深市退补现金替代
ERT_CASH_MUST_INTER_SZ 深市必须现金替代
ERT_CASH_RECOMPUTE_INTER_OTHER 非沪深市场成分证券退补现金替代
ERT_CASH_MUST_INTER_OTHER 表示非沪深市场成份证券必须现金替代
EPT_INVALID 无效值

6.6.2.2 enum XTP_ACCOUNT_TYPE

XTP_ACCOUNT_TYPE账户类型

枚举值

XTP_ACCOUNT_NORMAL 普通账户
XTP_ACCOUNT_CREDIT 信用账户
XTP_ACCOUNT_DERIVE 衍生品账户
XTP_ACCOUNT_UNKNOWN 未知账户类型

6.6.2.3 enum XTP_BUSINESS_TYPE

XTP_BUSINESS_TYPE证券业务类型

枚举值

XTP_BUSINESS_TYPE_CASH 普通股票业务（股票买卖，ETF买卖等）
XTP_BUSINESS_TYPE_IPOS 新股申购业务（对应的price type需选择限价类型）
XTP_BUSINESS_TYPE_REPO 回购业务 (对应的price type填为限价，side填为卖)
XTP_BUSINESS_TYPE_ETF ETF申赎业务
XTP_BUSINESS_TYPE_MARGIN 融资融券业务（暂未支持）
XTP_BUSINESS_TYPE_DESIGNATION 转托管（未支持）
XTP_BUSINESS_TYPE_ALLOTMENT 配股业务（对应的price type需选择限价类型,side填为买）
XTP_BUSINESS_TYPE_STRUCTURED_FUND_PURCHASE_REDEMPTION 分级基金申赎业务
XTP_BUSINESS_TYPE_STRUCTURED_FUND_SPLIT_MERGE 分级基金拆分合并业务
XTP_BUSINESS_TYPE_MONEY_FUND 货币基金业务（暂未支持）
XTP_BUSINESS_TYPE_OPTION 期权业务
XTP_BUSINESS_TYPE_EXECUTE 行权
XTP_BUSINESS_TYPE_FREEZE 锁定解锁，暂不支持
XTP_BUSINESS_TYPE_UNKNOWN 未知类型

6.6.2.4 enum XTP_EXCHANGE_TYPE

XTP_EXCHANGE_TYPE是交易所类型

枚举值

XTP_EXCHANGE_SH 上证

XTP_EXCHANGE_SZ 深证

XTP_EXCHANGE_UNKNOWN 不存在的交易所类型

6.6.2.5 enum XTP_FUND_OPER_STATUS

XTP_FUND_OPER_STATUS柜台资金操作结果

枚举值

XTP_FUND_OPER_PROCESSING XOMS已收到，正在处理中

XTP_FUND_OPER_SUCCESS 成功

XTP_FUND_OPER_FAILED 失败

XTP_FUND_OPER_SUBMITTED 已提交到集中柜台处理

XTP_FUND_OPER_UNKNOWN 未知

6.6.2.6 enum XTP_FUND_TRANSFER_TYPE

XTP_FUND_TRANSFER_TYPE是资金流转方向类型

枚举值

XTP_FUND_TRANSFER_OUT 转出 从XTP转出到柜台

XTP_FUND_TRANSFER_IN 转入 从柜台转入XTP

XTP_FUND_TRANSFER_UNKNOWN 未知类型

6.6.2.7 enum XTP_LOG_LEVEL

XTP_LOG_LEVEL是日志输出级别类型

枚举值

XTP_LOG_LEVEL_FATAL 严重错误级别

XTP_LOG_LEVEL_ERROR 错误级别

XTP_LOG_LEVEL_WARNING 警告级别

XTP_LOG_LEVEL_INFO info级别

XTP_LOG_LEVEL_DEBUG debug级别

XTP_LOG_LEVEL_TRACE trace级别

6.6.2.8 enum XTP_MARKET_TYPE

XTP_MARKET_TYPE市场类型

枚举值

XTP_MKT_INIT 初始化值或者未知
XTP_MKT_SZ_A 深圳A股
XTP_MKT_SH_A 上海A股
XTP_MKT_UNKNOWN 未知交易市场类型

6.6.2.9 enum XTP_OPT_CALL_OR_PUT_TYPE

XTP_OPT_CALL_OR_PUT_TYPE是一个认沽或认购类型

枚举值

XTP_OPT_CALL 认购
XTP_OPT_PUT 认沽

6.6.2.10 enum XTP_OPT_EXERCISE_TYPE_TYPE

XTP_OPT_EXERCISE_TYPE_TYPE是一个行权方式类型

枚举值

XTP_OPT_EXERCISE_TYPE_EUR 欧式
XTP_OPT_EXERCISE_TYPE_AME 美式

6.6.2.11 enum XTP_ORDER_ACTION_STATUS_TYPE

XTP_ORDER_ACTION_STATUS_TYPE是报单操作状态类型

枚举值

XTP_ORDER_ACTION_STATUS_SUBMITTED 已经提交
XTP_ORDER_ACTION_STATUS_ACCEPTED 已经接受
XTP_ORDER_ACTION_STATUS_REJECTED 已经被拒绝

6.6.2.12 enum XTP_ORDER_STATUS_TYPE

XTP_ORDER_STATUS_TYPE是报单状态类型

枚举值

XTP_ORDER_STATUS_INIT 初始化
XTP_ORDER_STATUS_ALLTRADED 全部成交
XTP_ORDER_STATUS_PARTTRADEDQUEUEING 部分成交
XTP_ORDER_STATUS_PARTTRADEDNOTQUEUEING 部分撤单
XTP_ORDER_STATUS_NOTRADEQUEUEING 未成交
XTP_ORDER_STATUS_CANCELED 已撤单
XTP_ORDER_STATUS_REJECTED 已拒绝
XTP_ORDER_STATUS_UNKNOWN 未知订单状态

6.6.2.13 enum XTP_ORDER_SUBMIT_STATUS_TYPE

XTP_ORDER_SUBMIT_STATUS_TYPE是报单提交状态类型

枚举值

XTP_ORDER_SUBMIT_STATUS_INSERT_SUBMITTED 订单已经提交
XTP_ORDER_SUBMIT_STATUS_INSERT_ACCEPTED 订单已经被接受
XTP_ORDER_SUBMIT_STATUS_INSERT_REJECTED 订单已经被拒绝
XTP_ORDER_SUBMIT_STATUS_CANCEL_SUBMITTED 撤单已经提交
XTP_ORDER_SUBMIT_STATUS_CANCEL_REJECTED 撤单已经被拒绝
XTP_ORDER_SUBMIT_STATUS_CANCEL_ACCEPTED 撤单已经被接受

6.6.2.14 enum XTP_POSITION_DIRECTION_TYPE

XTP_POSITION_DIRECTION_TYPE是一个持仓方向类型

枚举值

XTP_POSITION_DIRECTION_NET 净
XTP_POSITION_DIRECTION_LONG 多（期权则为权利方）
XTP_POSITION_DIRECTION_SHORT 空（期权则为义务方）
XTP_POSITION_DIRECTION_COVERED 备兑（期权则为备兑义务方）

6.6.2.15 enum XTP_PRICE_TYPE

XTP_PRICE_TYPE是价格类型

枚举值

XTP_PRICE_LIMIT 限价单-沪 / 深 / 沪期权（除普通股票业务外，其余业务均使用此种类型）
XTP_PRICE_BEST_OR_CANCEL 即时成交剩余转撤销，市价单-深 / 沪期权
XTP_PRICE_BEST5_OR_LIMIT 最优五档即时成交剩余转限价，市价单-沪
XTP_PRICE_BEST5_OR_CANCEL 最优5档即时成交剩余转撤销，市价单-沪深
XTP_PRICE_ALL_OR_CANCEL 全部成交或撤销,市价单-深 / 沪期权
XTP_PRICE_FORWARD_BEST 本方最优，市价单-深
XTP_PRICE_REVERSE_BEST_LIMIT 对方最优剩余转限价，市价单-深 / 沪期权
XTP_PRICE_LIMIT_OR_CANCEL 期权限价申报FOK
XTP_PRICE_TYPE_UNKNOWN 未知或者无效价格类型

6.6.2.16 enum XTP_PROTOCOL_TYPE

XTP_PROTOCOL_TYPE是通讯传输协议方式

枚举值

XTP_PROTOCOL_TCP 采用TCP方式传输
XTP_PROTOCOL_UDP 采用UDP方式传输(仅行情接口支持)

6.6.2.17 enum XTP_SPLIT_MERGE_STATUS

XTP_SPLIT_MERGE_STATUS是一个基金当天拆分合并状态类型

枚举值

XTP_SPLIT_MERGE_STATUS_ALLOW 允许拆分和合并
XTP_SPLIT_MERGE_STATUS_ONLY_SPLIT 只允许拆分，不允许合并
XTP_SPLIT_MERGE_STATUS_ONLY_MERGE 只允许合并，不允许拆分
XTP_SPLIT_MERGE_STATUS_FORBIDDEN 不允许拆分合并

6.6.2.18 enum XTP_TBT_TYPE

XTP_TBT_TYPE是一个逐笔回报类型

枚举值

XTP_TBT_ENTRUST 逐笔委托
XTP_TBT_TRADE 逐笔成交

6.6.2.19 enum XTP_TE_RESUME_TYPE

XTP_TE_RESUME_TYPE是公有流（订单响应、成交回报）重传方式

枚举值

XTP_TERT_RESTART 从本交易日开始重传
XTP_TERT_RESUME 从从上次收到的续传（暂未支持）
XTP_TERT_QUICK 只传送登录后公有流（订单响应、成交回报）的内容

6.6.2.20 enum XTP_TICKER_TYPE

XTP_TICKER_TYPE证券类型

枚举值

XTP_TICKER_TYPE_STOCK 普通股票
XTP_TICKER_TYPE_INDEX 指数
XTP_TICKER_TYPE_FUND 基金
XTP_TICKER_TYPE_BOND 债券
XTP_TICKER_TYPE_OPTION 期权
XTP_TICKER_TYPE_UNKNOWN 未知类型

6.7 xtp_api_struct.h 文件参考

定义业务数据结构

```
#include "xtp_api_struct_common.h"
#include "xquote_api_struct.h"
#include "xoms_api_struct.h"
#include "xoms_api_fund_struct.h"
```

6.7.1 详细描述

定义业务数据结构

作者

中泰证券股份有限公司

6.8 xtp_api_struct_common.h 文件参考

定义业务公共数据结构

```
#include <stdint.h>
#include "xtp_api_data_type.h"
```

结构体

- struct XTPRspInfoStruct
  响应信息

宏定义

- #define XTP_ERR_MSG_LEN 124
  错误信息的字符串长度

类型定义

- typedef struct XTPRspInfoStruct XTPRI
  响应信息

6.8.1 详细描述

定义业务公共数据结构

作者

中泰证券股份有限公司

6.9 xtp_quote_api.h 文件参考

定义行情订阅客户端接口

```
#include "xtp_api_struct.h"
```

结构体

- class QuoteSpi
  行情回调类
- class QuoteApi
  行情订阅接口类

6.9.1 详细描述

定义行情订阅客户端接口

作者

中泰证券股份有限公司

Index