XTP极速交易系统TraderAPI

制作者 中泰证券股份有限公司

2016年 三月 11日 星期五 13:23:41

Contents

Chapter 1

XTP 极速行情交易系统 Trader API 1.0.0

本项目是XTP项目中的交易类接口

(1) XTP的交易接口和响应类 xtp_trader_api.h

(2) 程序化交易接口测试Demo demo_test_trade_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

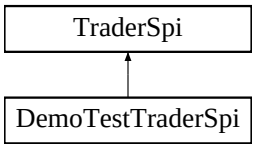
结构体说明

5.1 DemoTestTraderSpi类 参考

Demo自定义交易接口响应类

`#include <demo_test_trade_spi.h>`

类 DemoTestTraderSpi 继承关系图:



Public 成员函数

- virtual void OnError (XTPRI *error_info)
  错误应答
- virtual void OnOrderEvent (XTPOrderInfo *order_info, XTPRI *error_info)
  报单通知
- virtual void OnTradeEvent (XTPTradeReport *trade_info)
  成交通知
- virtual void OnQueryOrder (XTPQueryOrderRsp *order_info, XTPRI *error_info, int request_id, bool is_last)
- virtual void OnQueryTrade (XTPQueryTradeRsp *trade_info, XTPRI *error_info, int request_id, bool is_last)
- virtual void OnQueryPosition (XTPQueryStkPositionRsp *position, XTPRI *error_info, int request_id, bool is_last)
- virtual void OnQueryAsset (XTPQueryAssetRsp *asset, XTPRI *error_info, int request_id, bool is_last)
  请求查询资金账户响应

5.1.1 详细描述

Demo自定义交易接口响应类

作者

中泰证券股份有限公司

日期

十月 2015

5.1.2 成员函数说明

5.1.2.1 virtual void OnQueryOrder (XTPQueryOrderRsp * *order_info*, XTPRI * *error_info*, int *request_id*, bool *is_last*) [virtual]

请求查询报单响应

参数

order_info	查询到的报单
error_info	错误信息
request_id	此消息响应函数对应的请求ID
is_last	此消息响应函数是否为request_id这条请求所对应的最后一个响应

备注

由于支持分时段查询，一个查询请求可能对应多个响应

重载 TraderSpi .

5.1.2.2 virtual void OnQueryPosition (XTPQueryStkPositionRsp * *position*, XTPRI * *error_info*, int *request_id*, bool *is_last*) [virtual]

请求查询投资者持仓响应

参数

position	查询到的持仓情况
error_info	错误信息
request_id	此消息响应函数对应的请求ID
is_last	此消息响应函数是否为request_id这条请求所对应的最后一个响应

备注

由于用户可能持有多个股票，一个查询请求可能对应多个响应

重载 TraderSpi .

5.1.2.3 virtual void OnQueryTrade (XTPQueryTradeRsp * *trade_info*, XTPRI * *error_info*, int *request_id*, bool *is_last*) [virtual]

请求查询成交响应

参数

trade_info	查询到的成交回报
error_info	错误信息
request_id	此消息响应函数对应的请求ID
is_last	此消息响应函数是否为request_id这条请求所对应的最后一个响应

备注

由于支持分时段查询，一个查询请求可能对应多个响应

重载 TraderSpi .

该类的文档由以下文件生成:

- demo_test_trade_spi.h

5.2 TraderApi类 参考

交易接口类

`#include <xtp_trader_api.h>`

Public 成员函数

- virtual void Release ()=0
- virtual const char * GetTradingDay ()=0
- virtual void RegisterSpi (TraderSpi *spi)=0
- virtual int Login (const char *ip, int port, const char *user, const char *password, XTP_PROTOCOL_TYPE sock_type, int client_id=0)=0
- virtual int Logout ()=0
- virtual uint64_t InsertOrder (XTPOrderInsertInfo *order)=0
- virtual uint64_t CancelOrder (const uint64_t order_xtp_id)=0
- virtual int QueryOrderByXTPID (const uint64_t order_xtp_id, int request_id)=0
- virtual int QueryOrders (const XTPQueryOrderReq *query_param, int request_id)=0
- virtual int QueryTradesByXTPID (const uint64_t order_xtp_id, int request_id)=0
- virtual int QueryTrades (XTPQueryTraderReq *query_param, int request_id)=0
- virtual int QueryPosition (const char *ticker, int request_id)=0
- virtual int QueryAsset (int request_id)=0

静态 Public 成员函数

- static TraderApi * CreateTraderApi (const char *save_file_path="")

5.2.1 详细描述

交易接口类

作者

中泰证券股份有限公司

日期

十月 2015

5.2.2 成员函数说明

5.2.2.1 virtual uint64_t CancelOrder (const uint64_t *order_xtp_id*) [pure virtual]

报单操作请求

返回

撤单单在XTP系统中的ID,如果为'0'表示撤单失败，用户需要记录下返回的order_cancel_xtp_id

参数

order_xtp_id	需要撤销的委托单在XTP系统中的ID

5.2.2.2 static TraderApi* CreateTraderApi (const char * *save_file_path* = "") [static]

创建TraderApi

参数

save_file_path	（保留字段）存贮订阅信息文件的目录，默认为当前目录

返回

创建出的UserApi

5.2.2.3 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

5.2.2.4 virtual uint64_t InsertOrder (XTPOrderInsertInfo * *order*) [pure virtual]

报单录入请求

返回

报单在XTP系统中的ID,如果为'0'表示报单失败，用户需要记录下返回的order_xtp_id

参数

order	报单录入信息

5.2.2.5 virtual int Login (const char * *ip,* int *port,* const char * *user,* const char * *password,* XTP_PROTOCOL_TYPE *sock_type,* int *client_id* = 0) [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，"-2"表示已存在连接，不允许重复登录，如果需要重连，请先logout

参数

ip	服务器地址
port	服务器端口号
user	登录用户名
password	登录密码
sock_type	“1”代表TCP，“2”代表UDP，目前暂时只支持TCP
client_id	（保留字段）客户端id，用于区分同一用户的不同连接

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

5.2.2.6 virtual int Logout ( ) [pure virtual]

登出请求

返回

登出是否成功，“0”表示登出成功，“-1”表示登出失败

5.2.2.7 virtual int QueryAsset (int *request_id*) [pure virtual]

请求查询资产

返回

查询是否成功，“0”表示成功，非“0”表示出错

参数

request_id	用于用户定位查询响应的ID，由用户自定义

5.2.2.8 virtual int QueryOrderByXTPID (const uint64_t *order_xtp_id,* int *request_id*) [pure virtual]

根据报单ID请求查询报单

返回

查询是否成功，“0”表示成功，非“0”表示出错

参数

order_xtp_id	需要查询的报单在xtp系统中的ID
request_id	用于用户定位查询响应的ID，由用户自定义

5.2.2.9 virtual int QueryOrders (const XTPQueryOrderReq * *query_param,* int *request_id*) [pure virtual]

请求查询报单

返回

查询是否成功，“0”表示成功，非“0”表示出错

参数

query_param	需要查询的订单相关筛选条件
request_id	用于用户定位查询响应的ID，由用户自定义

备注

该方法支持分时段查询，如果股票代码为空，则默认查询时间段内的所有报单，否则查询时间段内所有跟股票代码相关的报单

5.2.2.10 virtual int QueryPosition (const char * *ticker,* int *request_id*) [pure virtual]

请求查询投资者持仓

返回

查询是否成功，“0”表示成功，非“0”表示出错

参数

ticker	需要查询的持仓股票，可以为空
request_id	用于用户定位查询响应的ID，由用户自定义

备注

该方法如果用户提供了股票代码，则会查询此股票的持仓信息，如果股票代码为空，则默认查询所有持仓信息

5.2.2.11 virtual int QueryTrades (XTPQueryTraderReq * *query_param,* int *request_id*) [pure virtual]

请求查询已成交

返回

查询是否成功，“0”表示成功，非“0”表示出错

参数

query_param	需要查询的成交回报筛选条件
request_id	用于用户定位查询响应的ID，由用户自定义

备注

该方法支持分时段查询，如果股票代码为空，则默认查询时间段内的所有成交回报，否则查询时间段内所有跟股票代码相关的成交回报

5.2.2.12 virtual int QueryTradesByXTPID (const uint64_t *order_xtp_id,* int *request_id*) [pure virtual]

根据委托编号请求查询相关成交

返回

查询是否成功，“0”表示成功，非“0”表示出错

参数

order_xtp_id	需要查询的委托编号
request_id	用于用户定位查询响应的ID，由用户自定义

备注

此函数查询出的结果可能对应多个响应

5.2.2.13 virtual void RegisterSpi (TraderSpi * *spi*) [pure virtual]

注册回调接口

参数

spi	派生自回调接口类的实例

5.2.2.14 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

该类的文档由以下文件生成:

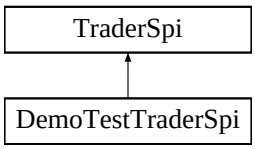
- xtp_trader_api.h

5.3 TraderSpi类 参考

交易接口响应类

`#include <xtp_trader_api.h>`

类 TraderSpi 继承关系图:



Public 成员函数

- virtual void OnDisconnected (int reason)
- virtual void OnError (XTPRI *error_info)
- virtual void OnOrderEvent (XTPOrderInfo *order_info, XTPRI *error_info)
- virtual void OnTradeEvent (XTPTradeReport *trade_info)
- virtual void OnCancelOrderError (XTPOrderCancelInfo *cancel_info, XTPRI *error_info)
- virtual void OnQueryOrder (XTPQueryOrderRsp *order_info, XTPRI *error_info, int request_id, bool is_last)
- virtual void OnQueryTrade (XTPQueryTradeRsp *trade_info, XTPRI *error_info, int request_id, bool is_last)
- virtual void OnQueryPosition (XTPQueryStkPositionRsp *position, XTPRI *error_info, int request_id, bool is_last)
- virtual void OnQueryAsset (XTPQueryAssetRsp *asset, XTPRI *error_info, int request_id, bool is_last)

  请求查询资金账户响应

5.3.1 详细描述

交易接口响应类

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 virtual void OnCancelOrderError (XTPOrderCancelInfo * *cancel_info*, XTPRI * *error_info*) [inline], [virtual]

撤单出错响应

参数

cancel_info	撤单具体信息，包括撤单单和待撤单的order_xtp_id
error_info	撤单被拒绝或者发生错误时错误代码和错误信息

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

当客户端与交易后台通信连接断开时，该方法被调用。

备注

保留函数，暂未支持

参数

reason	错误原因

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

错误应答

参数

error_info	具体的错误代码和错误信息

被 DemoTestTraderSpi 重载.

5.3.2.4 virtual void OnOrderEvent (XTPOrderInfo * *order_info*, XTPRI * *error_info*) [inline], [virtual]

报单通知

参数

order_info	订单响应具体信息
error_info	订单被拒绝或者发生错误时错误代码和错误信息

备注

每次订单状态更新时，都会被调用

被 DemoTestTraderSpi 重载.

5.3.2.5 virtual void OnQueryOrder (XTPQueryOrderRsp * *order_info*, XTPRI * *error_info*, int *request_id*, bool *is_last*) [inline], [virtual]

请求查询报单响应

参数

order_info	查询到的报单
error_info	错误信息
request_id	此消息响应函数对应的请求ID
is_last	此消息响应函数是否为request_id这条请求所对应的最后一个响应

备注

由于支持分时段查询，一个查询请求可能对应多个响应

被 DemoTestTraderSpi 重载.

5.3.2.6 virtual void OnQueryPosition (XTPQueryStkPositionRsp * *position*, XTPRI * *error_info*, int *request_id*, bool *is_last*) [inline], [virtual]

请求查询投资者持仓响应

参数

position	查询到的持仓情况
error_info	错误信息
request_id	此消息响应函数对应的请求ID
is_last	此消息响应函数是否为request_id这条请求所对应的最后一个响应

备注

由于用户可能持有多个股票，一个查询请求可能对应多个响应

被 DemoTestTraderSpi 重载.

5.3.2.7 virtual void OnQueryTrade (XTPQueryTradeRsp * *trade_info*, XTPRI * *error_info*, int *request_id*, bool *is_last*) [inline], [virtual]

请求查询成交响应

参数

trade_info	查询到的成交回报

error_info	错误信息
request_id	此消息响应函数对应的请求ID
is_last	此消息响应函数是否为request_id这条请求所对应的最后一个响应

备注

由于支持分时段查询，一个查询请求可能对应多个响应

被 DemoTestTraderSpi 重载.

5.3.2.8 virtual void OnTradeEvent (XTPTradeReport * *trade_info*) [inline], [virtual]

成交通知

参数

trade_info	成交回报的具体信息

备注

订单有成交发生的时候，会被调用

被 DemoTestTraderSpi 重载.

该类的文档由以下文件生成:

- xtp_trader_api.h

5.4 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）
- 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
- double pre_open_interest

昨持仓量（目前未填写）
- double open_interest
  持仓量（目前未填写）
- double pre_settlement_price
  上次结算价（目前未填写）
- double settlement_price
  本次结算价（目前未填写）
- double upper_limit_price
  涨停板价（目前未填写）
- double lower_limit_price
  跌停板价（目前未填写）
- double pre_delta
  昨虚实度（目前未填写）
- double curr_delta
  今虚实度（目前未填写）
- int64_t data_time
  时间类
- int32_t qty
  数量
- double turnover
  成交金额
- double avg_price
  当日均价
- double bid [10]
  十档申买价
- double ask [10]
  十档申卖价
- int32_t bid_qty [10]
  十档申买量
- int32_t ask_qty [10]
  十档申卖量
- int32_t trades_count
  成交笔数
- char ticker_status [8]
  当前交易状态说明
- int32_t total_bid_qty
  委托买入总量
- int32_t total_ask_qty
  委托卖出总量
- double ma_bid_price
  加权平均委买价格
- double ma_ask_price
  加权平均委卖价格
- double ma_bond_bid_price
  债券加权平均委买价格
- double ma_bond_ask_price
  债券加权平均委卖价格
- double yield_to_maturity
  债券到期收益率
- double iopv
  *ETF*净值估值

- int32_t etf_buy_count
  *ETF*申购笔数
- int32_t etf_sell_count
  *ETF*赎回笔数
- double etf_buy_qty
  *ETF*申购数量
- double etf_buy_money
  *ETF*申购金额
- double etf_sell_qty
  *ETF*赎回数量
- double etf_sell_money
  *ETF*赎回金额
- double total_warrant_exec_qty
  权证执行的总数量
- double warrant_lower_price
  权证跌停价格（元）
- double warrant_upper_price
  权证涨停价格（元）
- int32_t cancel_buy_count
  买入撤单笔数
- int32_t cancel_sell_count
  卖出撤单笔数
- double cancel_buy_qty
  买入撤单数量
- double cancel_sell_qty
  卖出撤单数量
- double cancel_buy_money
  买入撤单金额
- double cancel_sell_money
  卖出撤单金额
- int32_t total_buy_count
  买入总笔数
- int32_t total_sell_count
  卖出总笔数
- int32_t duration_after_buy
  买入委托成交最大等待时间
- int32_t duration_after_sell
  卖出委托成交最大等待时间
- int32_t num_bid_orders
  买方委托价位数
- int32_t num_ask_orders
  卖方委托价位数
- int32_t exec_time
  成交时间（*UA3113*）
- char is_market_closed [4]
  闭市标志（*UA103/UA104*）
- double total_position
  合约持仓量（*UA103*）
- double pe_ratio1
  市盈率*1*（*UA103*）
- double pe_ratio2
  市盈率*2*（*UA103*）

5.4.1 详细描述

行情

该结构体的文档由以下文件生成:

- xquote_api_struct.h

5.5 XTPOrderCancel结构体 参考

撤单

`#include <xoms_api_struct.h>`

成员变量

- uint64_t order_cancel_xtp_id
  *XTP*系统订单*ID.*
- uint32_t order_cancel_client_id
  报单操作引用
- char ticker [XTP_API_TICKER_LEN]
  合约代码
- XTP_MARKET_TYPE market
  交易市场
- int64_t action_time
  操作时间
- uint32_t order_client_id
  报单引用
- uint64_t order_xtp_id
  操作对象订单的序号

5.5.1 详细描述

撤单

该结构体的文档由以下文件生成:

- xoms_api_struct.h

5.6 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*

5.6.1 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

5.7 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*.
- 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
- uint64_t order_cancel_xtp_id
  撤单在*XTP*系统中的*id*
- char ticker [XTP_API_TICKER_LEN]
  合约代码
- XTP_MARKET_TYPE market
  交易市场
- double price
  价格
- int64_t quantity
  数量
- XTP_PRICE_TYPE price_type
  报单价格条件
- XTP_SIDE_TYPE side
  买卖方向
- int64_t qty_traded
  今成交数量
- int64_t qty_left
  剩余数量
- int64_t insert_time
  委托时间
- int64_t update_time
  最后修改时间
- int64_t cancel_time
  撤销时间
- double trade_amount
  成交金额
- char order_local_id [XTP_LOCAL_ORDER_LEN]
  本地报单编号 *OMS*生成的单号
- XTP_ORDER_STATUS_TYPE order_status

报单状态

- XTP_ORDER_SUBMIT_STATUS_TYPE order_submit_status
  报单提交状态
- TXTPOrderTypeType order_type
  报单类型

5.7.1 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

5.8 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*.
- uint32_t order_client_id
  报单引用,由客户自定义
- char ticker [XTP_API_TICKER_LEN]
  合约代码 客户端请求不带空格
- 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
- XTP_PRICE_TYPE price_type
  报单价格
- XTP_SIDE_TYPE side
  买卖方向

5.8.1 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

5.9 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
- double buying_power
  可用资金
- double security_asset
  证券资产
- double fund_buy_amount
  累计买入成交证券占用资金（保留字段）
- double fund_buy_fee
  累计买入成交交易费用（保留字段）
- double fund_sell_amount
  累计卖出成交证券所得资金（保留字段）
- double fund_sell_fee
  累计卖出成交交易费用（保留字段）

5.9.1 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

5.10 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_API_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

5.10.1 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

5.11 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*.
- uint64_t exec_id
  成交执行编号

5.11.1 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

5.12 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_API_TICKER_LEN]
  证券代码
- char ticker_name [XTP_API_TICKER_NAME_LEN]
  证券名称
- int64_t total_qty
  当前持仓
- int64_t sellable_qty
  可用股份数
- double avg_price
  持仓成本
- double unrealized_pnl
  浮动盈亏

5.12.1 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

5.13 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_API_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

5.13.1 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

5.14 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

5.14.1 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

5.15 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 "*

5.15.1 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

5.16 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.*
- uint32_t order_client_id
  报单引用
- char ticker [XTP_API_TICKER_LEN]
  合约代码
- XTP_MARKET_TYPE market
  交易市场
- uint64_t local_order_id
  订单号
- uint64_t exec_id
  成交编号
- double price
  价格
- int64_t quantity
  数量
- int64_t trade_time
  成交时间
- double trade_amount
  成交金额
- uint64_t report_index
  成交序号 –回报记录号
- char order_exch_id [XTP_ORDER_EXCH_LEN]
  报单编号 –交易所单号
- TXTPTradeTypeType trade_type
  成交类型 –成交回报中的执行类型
- XTP_SIDE_TYPE side
  买卖方向
- char branch_pbu [XTP_BRANCH_PBU_LEN]
  交易所交易员代码

5.16.1 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_trade_api.cpp 文件参考

定义控制台测试应用程序的入口点

```
#include "xtp_trader_api.h"
#include <string>
#include <map>
#include <iostream>
#include <unistd.h>
#include "xtp_trader_api_compatible.h"
#include "demo_test_trade_spi.h"
```

函数

- int main ()

  int main() {

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

```
XTP::API::TraderApi* pUserApi = XTP::API::TraderApi::CreateTraderApi("");            // 创建UserApi

DemoTestTraderSpi* pUserSpi = new DemoTestTraderSpi();// 创建响应类实例

pUserApi->RegisterSpi(pUserSpi);                          // 注册事件类
```

```
int loginResult = pUserApi->Login(server_ip.c_str(), server_port, username.c_str(), password.c_str(), XTP_PROT
陆交易服务器

if (loginResult == 0)

{
    int client_id = 1;//用户自定义用于标识本地订单的编号，可以任意

    //下单

    XTPOrderInsertInfo orderInsert;

    orderInsert.order_client_id = client_id++;

    std::string ticker("000002");

    strcpy(orderInsert.ticker, ticker.c_str());

    orderInsert.exchange_id = (XTP_EXCHANGE_TYPE)2;
    orderInsert.price = 17.5;

    orderInsert.quantity = 200;

    orderInsert.side = (XTP_SIDE_TYPE)1;

    orderInsert.price_type = (XTP_PRICE_TYPE)3;

    //返回的xtp_id需要记录下来，与服务器交互的时候，所有对订单的操作由xtp_id唯一确定

    int64_t insert_xtp_id = pUserApi->InsertOrder(&orderInsert);

    //如果需要撤单
    //返回的xtp_id需要记录下来，与服务器交互的时候，所有对订单的操作由xtp_id唯一确定

    int64_t cancel_xtp_id = pUserApi->CancelOrder(&orderCancel);

}

return 0;


}
```

6.2 demo_test_trade_spi.h 文件参考

Demo自定义客户端交易响应接口类

```
#include "xtp_trader_api.h"
```

结构体

- class DemoTestTraderSpi

  *Demo*自定义交易接口响应类

6.2.1 详细描述

Demo自定义客户端交易响应接口类

作者

中泰证券股份有限公司

6.3 xoms_api_struct.h 文件参考

定义订单管理系统外部交互接口数据结构

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

结构体

- struct XTPOrderInsertInfo
  新订单请求
- struct XTPOrderCancel
  撤单
- 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

宏定义

- #define **XTP_API_TICKER_LEN** 13
- #define **XTP_API_TICKER_NAME_LEN** 48
- #define **XTP_LOCAL_ORDER_LEN** 11
- #define **XTP_ORDER_EXCH_LEN** 17
- #define **XTP_ORDER_RES_LEN** 4
- #define **XTP_SUBPBUID_LEN** 7
- #define **XTP_BRANCH_PBU_LEN** 7
- #define **XTP_SYS_INSTRUMENT_LEN** 24

类型定义

- typedef XTPOrderInfo XTPQueryOrderRsp
  报单查询响应结构体
- typedef XTPTradeReport XTPQueryTradeRsp
  成交回报查询响应结构体

6.3.1 详细描述

定义订单管理系统外部交互接口数据结构

日期

2015/10/23 16:45

作者

howellxu Contact: user@company.com

TODO: long description

注解

6.4 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 XTPMarketDataStruct
  行情

类型定义

- typedef struct XTPSpecificTickerStruct XTPST
  指定的合约
- typedef struct XTPMarketDataStruct XTPMD
  行情

6.4.1 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 xtp_api_data_type.h 文件参考

定义兼容数据基本类型

宏定义

- #define XTP_TICKER_LEN 16
  存放证券代码的字符串长度
- #define XTP_EXP_Normal '0'
  正常
- #define XTP_EXP_GenOrderByTrade '1'
  根据成交生成报单
- #define XTP_ECS_NoConnection '1'
  没有任何连接
- #define XTP_ECS_QryInstrumentSent '2'
  已经发出合约查询请求
- #define XTP_ECS_GotInformation '9'
  已经获取信息
- #define XTP_PC_Futures '1'
  期货
- #define XTP_PC_Options '2'
  期权
- #define XTP_PC_Combination '3'
  组合
- #define XTP_PC_Spot '4'
  即期
- #define XTP_PC_EFP '5'
  期转现
- #define XTP_PC_StockA '6'
  证券*A*股
- #define XTP_PC_StockB '7'
  证券*B*股
- #define XTP_PC_ETF '8'
  ETF
- #define XTP_PC_ETFPurRed '9'
  *ETF*申赎
- #define XTP_PT_Net '1'
  净持仓
- #define XTP_PT_Gross '2'
  综合持仓
- #define XTP_PDT_UseHistory '1'
  使用历史持仓
- #define XTP_PDT_NoUseHistory '2'
  不使用历史持仓
- #define XTP_IP_NotStart '0'
  未上市
- #define XTP_IP_Started '1'
  上市
- #define XTP_IP_Pause '2'
  停牌
- #define XTP_IP_Expired '3'
  到期
- #define XTP_PTT_CanSelTodayPos '1'
  今日新增持仓能卖出
- #define XTP_PTT_CannotSellTodayPos '2'

今日新增持仓不能卖出

- #define XTP_ICT_EID '0'
  组织机构代码
- #define XTP_ICT_IDCard '1'
  身份证
- #define XTP_ICT_OfficerIDCard '2'
  军官证
- #define XTP_ICT_PoliceIDCard '3'
  警官证
- #define XTP_ICT_SoldierIDCard '4'
  士兵证
- #define XTP_ICT_HouseholdRegister '5'
  户口簿
- #define XTP_ICT_Passport '6'
  护照
- #define XTP_ICT_TaiwanCompatriotIDCard '7'
  台胞证
- #define XTP_ICT_HomeComingCard '8'
  回乡证
- #define XTP_ICT_LicenseNo '9'
  营业执照号
- #define XTP_ICT_TaxNo 'A'
  税务登记号
- #define XTP_ICT_OtherCard 'x'
  其他证件
- #define XTP_CLT_Normal '1'
  普通
- #define XTP_CLT_Credit '2'
  信用交易
- #define XTP_CLT_Derive '3'
  衍生品账户
- #define XTP_CLT_Other '4'
  其他类型
- #define XTP_FC_ForceUserLogout '2'
  强制用户登出
- #define XTP_FC_UserPasswordUpdate '3'
  变更管理用户口令
- #define XTP_FC_BrokerPasswordUpdate '4'
  变更经纪公司口令
- #define XTP_FC_InvestorPasswordUpdate '5'
  变更投资者口令
- #define XTP_FC_OrderInsert '6'
  报单插入
- #define XTP_FC_OrderAction '7'
  报单操作
- #define XTP_FC_SyncSystemData '8'
  同步系统数据
- #define XTP_FC_SyncBrokerData '9'
  同步经纪公司数据
- #define XTP_FC_SuperQuery 'B'
  超级查询

- #define XTP_FC_ParkedOrderInsert 'C'
  报单插入
- #define XTP_FC_ParkedOrderAction 'D'
  报单操作
- #define XTP_FC_SyncOTP 'E'
  同步动态令牌
- #define XTP_FC_UnkownOrderAction 'F'
  未知单操作
- #define XTP_FC_DepositoryTransfer 'G'
  转托管
- #define XTP_FC_ExcessStockTransfer 'H'
  余券划转
- #define XTP_UT_Investor '0'
  投资者
- #define XTP_UT_Operator '1'
  操作员
- #define XTP_UT_SuperUser '2'
  管理员
- #define XTP_BFC_ForceUserLogout '1'
  强制用户登出
- #define XTP_BFC_UserPasswordUpdate '2'
  变更用户口令
- #define XTP_BFC_SyncBrokerData '3'
  同步经纪公司数据
- #define XTP_BFC_OrderInsert '5'
  报单插入
- #define XTP_BFC_OrderAction '6'
  报单操作
- #define XTP_BFC_AllQuery '7'
  全部查询
- #define XTP_BFC_UnkownOrderAction '8'
  未知单操作
- #define XTP_BFC_DepositoryTransfer '9'
  转托管
- #define XTP_BFC_ExcessStockTransfer 'A'
  余券划转
- #define XTP_BFC_FundInterTransfer 'B'
  资金内转
- #define XTP_BFC_log 'a'
  系统功能：登入/登出/修改密码等
- #define XTP_BFC_BaseQry 'b'
  基本查询：查询基础数据，如合约，交易所等常量
- #define XTP_BFC_TradeQry 'c'
  交易查询：如查成交，委托
- #define XTP_BFC_Trade 'd'
  交易功能：报单，撤单
- #define XTP_BFC_Virement 'e'
  转账
- #define XTP_BFC_Session 'g'
  查询/管理：查询会话，踢人等
- #define XTP_BFC_SyncOTP 'E'

同步动态令牌

- #define XTP_AcT_Normal '1'
  普通账户
- #define XTP_AcT_Credit '2'
  信用账户
- #define XTP_AcT_Derive '3'
  衍生品账户
- #define XTP_AcT_Other '4'
  其他类型
- #define XTP_DR_All '1'
  所有
- #define XTP_DR_Group '2'
  组织架构
- #define XTP_DR_Single '3'
  单一投资者
- #define XTP_URT_Logon '1'
  登录
- #define XTP_URT_Transfer '2'
  银期转帐
- #define XTP_URT_EMail '3'
  邮寄结算单
- #define XTP_URT_Fax '4'
  传真结算单
- #define XTP_URT_ConditionOrder '5'
  条件单
- #define XTP_HF_Speculation '1'
  投机
- #define XTP_HF_Hedge '3'
  套保
- #define XTP_D_Buy '0'
  买
- #define XTP_D_Sell '1'
  卖
- #define XTP_TRDT_Common '0'
  普通成交
- #define XTP_TRDT_OptionsExecution '1'
  期权执行
- #define XTP_TRDT_OTC '2'
  *OTC*成交
- #define XTP_TRDT_EFPDerived '3'
  期转现衍生成交
- #define XTP_TRDT_CombinationDerived '4'
  组合衍生成交
- #define XTP_TRDT_EFTPurchase '5'
  *ETF*申购
- #define XTP_TRDT_EFTRedem '6'
  *ETF*赎回
- #define XTP_CDS_Forbidden '0'
  不允许申购赎回
- #define XTP_CDS_Allow '1'
  表示允许申购和赎回

- #define XTP_CDS_OnlyPurchase '2'
  允许申购、不允许赎回
- #define XTP_CDS_OnlyRedeem '3'
  不允许申购、允许赎回
- #define XTP_ETFCRS_Forbidden '0'
  禁止现金替代
- #define XTP_ETFCRS_Allow '1'
  可以现金替代
- #define XTP_ETFCRS_Force '2'
  必须现金替代
- #define XTP_ETFCRS_CrossMarketComp '3'
  跨市场股票退补现金替代
- #define XTP_ETFCRS_CrossMarketFroce '4'
  跨市场必须现金替代
- #define XTP_CPTSTOCK_TOTALSTOCK '1'
  总通股本
- #define XTP_CPTSTOCK_CIRCULATION '2'
  流通股本
- #define XTP_MPT_PreSettlementPrice '1'
  昨结算价
- #define XTP_MPT_SettlementPrice '2'
  最新价
- #define XTP_MPT_AveragePrice '3'
  成交均价
- #define XTP_MPT_OpenPrice '4'
  开仓价
- #define XTP_AG_All '1'
  浮盈浮亏都计算
- #define XTP_AG_OnlyLost '2'
  浮盈不计，浮亏计
- #define XTP_AG_OnlyGain '3'
  浮盈计，浮亏不计
- #define XTP_AG_None '4'
  浮盈浮亏都不计算
- #define XTP_ICP_Include '0'
  包含平仓盈利
- #define XTP_ICP_NotInclude '2'
  不包含平仓盈利
- #define XTP_AWT_Enable '0'
  不受可提比例限制
- #define XTP_AWT_Disable '2'
  受可提比例限制
- #define XTP_AWT_NoHoldEnable '3'
  无仓不受可提比例限制
- #define XTP_HPA_Base '1'
  基本
- #define XTP_HPA_NoneTrade '4'
  非交易
- #define XTP_HPA_Stock '5'
  证券
- #define XTP_TPID_EncryptionStandard 'E'

系统加密算法

- #define XTP_TPID_SingleUserSessionMaxNum 'S'
  用户最大会话数
- #define XTP_TPID_LoginFailMaxNum 'L'
  最大连续登录失败数
- #define XTP_TPID_IsAuthForce 'A'
  是否强制认证
- #define XTP_TPID_GenUserEvent 'G'
  是否生成用户事件
- #define XTP_TPID_StartOrderLocalID 'O'
  起始报单本地编号
- #define XTP_TPID_RepayStockAlgo 'R'
  融资融券买券还券算法
- #define XTP_TPID_DeriveWithdrawRatio 'D'
  衍生品账户资金提取线
- #define XTP_IR_All '1'
  所有
- #define XTP_IR_Group '2'
  投资者组
- #define XTP_IR_Single '3'
  单一投资者
- #define XTP_DS_Asynchronous '1'
  未同步
- #define XTP_DS_Synchronizing '2'
  同步中
- #define XTP_DS_Synchronized '3'
  已同步
- #define XTP_TCS_NotConnected '1'
  没有任何连接
- #define XTP_TCS_Connected '2'
  已经连接
- #define XTP_TCS_QryInstrumentSent '3'
  已经发出合约查询请求
- #define XTP_TCS_SubPrivateFlow '4'
  订阅私有流
- #define XTP_OAS_Submitted 'a'
  已经提交
- #define XTP_OAS_Accepted 'b'
  已经接受
- #define XTP_OAS_Rejected 'c'
  已经被拒绝
- #define XTP_OST_AllTraded '0'
  全部成交
- #define XTP_OST_PartTradedQueueing '1'
  部分成交还在队列中
- #define XTP_OST_PartTradedNotQueueing '2'
  部分成交不在队列中
- #define XTP_OST_NoTradeQueueing '3'
  未成交还在队列中
- #define XTP_OST_NoTradeNotQueueing '4'
  未成交不在队列中

- #define XTP_OST_Canceled '5'
  撤单
- #define XTP_OST_Unknown 'a'
  未知
- #define XTP_OST_NotTouched 'b'
  尚未触发
- #define XTP_OST_Touched 'c'
  已触发
- #define XTP_OSS_InsertSubmitted '0'
  已经提交
- #define XTP_OSS_CancelSubmitted '1'
  撤单已经提交
- #define XTP_OSS_ModifySubmitted '2'
  修改已经提交
- #define XTP_OSS_Accepted '3'
  已经接受
- #define XTP_OSS_InsertRejected '4'
  报单已经被拒绝
- #define XTP_OSS_CancelRejected '5'
  撤单已经被拒绝
- #define XTP_OSS_ModifyRejected '6'
  改单已经被拒绝
- #define XTP_PSD_Today '1'
  今日持仓
- #define XTP_PSD_History '2'
  历史持仓
- #define XTP_ER_Broker '1'
  代理
- #define XTP_ER_Host '2'
  自营
- #define XTP_ER_Maker '3'
  做市商
- #define XTP_PD_Net '1'
  净
- #define XTP_PD_Long '2'
  多头
- #define XTP_PD_Short '3'
  空头
- #define XTP_PD_Covered '4'
  备兑
- #define XTP_OPT_AnyPrice '1'
  即时成交剩余撤销市价单
- #define XTP_OPT_LimitPrice '2'
  限价
- #define XTP_OPT_BestPrice '3'
  最优五档即时成交剩余撤销市价单
- #define XTP_OPT_BestLimitPrice '4'
  最优五档即时成交剩余转限价市价单
- #define XTP_OPT_AllPrice '5'
  全部成交或撤销市价单
- #define XTP_OPT_ForwardBestPrice '6'

本方最优价格市价单

- #define XTP_OPT_ReverseBestPrice '7'
  对方最优价格市价单
- #define XTP_OPT_Any2LimitPrice '8'
  即时成交剩余转限价市价单
- #define XTP_OPT_AllLimitPrice '9'
  全部成交或撤销限价单
- #define XTP_OF_Open '0'
  开仓
- #define XTP_OF_Close '1'
  平仓
- #define XTP_OF_ForceClose '2'
  强平
- #define XTP_OF_CloseToday '3'
  平今
- #define XTP_OF_CloseYesterday '4'
  平昨
- #define XTP_OF_ForceOff '5'
  强减
- #define XTP_OF_LocalForceClose '6'
  本地强平
- #define XTP_FCC_NotForceClose '0'
  非强平
- #define XTP_FCC_LackDeposit '1'
  资金不足
- #define XTP_FCC_ClientOverPositionLimit '2'
  客户超仓
- #define XTP_FCC_MemberOverPositionLimit '3'
  会员超仓
- #define XTP_FCC_NotMultiple '4'
  持仓非整数倍
- #define XTP_FCC_Violation '5'
  违规
- #define XTP_FCC_Other '6'
  其它
- #define XTP_FCC_PersonDeliv '7'
  自然人临近交割
- #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
- #define XTP_TC_IOC '1'
  立即完成，否则撤销

- #define XTP_TC_GFS '2'
  本节有效
- #define XTP_TC_GFD '3'
  当日有效
- #define XTP_TC_GTD '4'
  指定日期前有效
- #define XTP_TC_GTC '5'
  撤销前有效
- #define XTP_TC_GFA '6'
  集合竞价有效
- #define XTP_VC_AV '1'
  任何数量
- #define XTP_VC_MV '2'
  最小数量
- #define XTP_VC_CV '3'
  全部数量
- #define XTP_CC_Immediately '1'
  立即
- #define XTP_CC_Touch '2'
  止损
- #define XTP_CC_TouchProfit '3'
  止赢
- #define XTP_CC_ParkedOrder '4'
  预埋单
- #define XTP_CC_LastPriceGreaterThanStopPrice '5'
  最新价大于条件价
- #define XTP_CC_LastPriceGreaterEqualStopPrice '6'
  最新价大于等于条件价
- #define XTP_CC_LastPriceLesserThanStopPrice '7'
  最新价小于条件价
- #define XTP_CC_LastPriceLesserEqualStopPrice '8'
  最新价小于等于条件价
- #define XTP_CC_AskPriceGreaterThanStopPrice '9'
  卖一价大于条件价
- #define XTP_CC_AskPriceGreaterEqualStopPrice 'A'
  卖一价大于等于条件价
- #define XTP_CC_AskPriceLesserThanStopPrice 'B'
  卖一价小于条件价
- #define XTP_CC_AskPriceLesserEqualStopPrice 'C'
  卖一价小于等于条件价
- #define XTP_CC_BidPriceGreaterThanStopPrice 'D'
  买一价大于条件价
- #define XTP_CC_BidPriceGreaterEqualStopPrice 'E'
  买一价大于等于条件价
- #define XTP_CC_BidPriceLesserThanStopPrice 'F'
  买一价小于条件价
- #define XTP_CC_BidPriceLesserEqualStopPrice 'H'
  买一价小于等于条件价
- #define XTP_AF_Delete '0'
  删除
- #define XTP_AF_Modify '3'

修改

- #define XTP_TR_Allow '0'
  可以交易
- #define XTP_TR_Forbidden '2'
  不能交易
- #define XTP_OSRC_Participant '0'
  来自参与者
- #define XTP_OSRC_Administrator '1'
  来自管理员
- #define XTP_PSRC_LastPrice '0'
  前成交价
- #define XTP_PSRC_Buy '1'
  买委托价
- #define XTP_PSRC_Sell '2'
  卖委托价
- #define XTP_UET_Login '1'
  登录
- #define XTP_UET_Logout '2'
  登出
- #define XTP_UET_Trading '3'
  交易成功
- #define XTP_UET_TradingError '4'
  交易失败
- #define XTP_UET_UpdatePassword '5'
  修改密码
- #define XTP_UET_Authenticate '6'
  客户端认证
- #define XTP_UET_Other '9'
  其他
- #define XTP_OTP_NONE '0'
  无动态令牌
- #define XTP_OTP_TOTP '1'
  时间令牌
- #define XTP_TSRC_NORMAL '0'
  来自交易所普通回报
- #define XTP_TSRC_QUERY '1'
  来自查询
- #define XTP_INR_All '1'
  所有
- #define XTP_INR_Product '2'
  产品
- #define XTP_INR_Model '3'
  股票权限模版
- #define XTP_INR_Stock '4'
  股票
- #define XTP_INR_Market '5'
  市场
- #define XTP_STT_Stock '0'
  可交易证券
- #define XTP_STT_BuyNetService '1'
  买入网络密码服务

- #define XTP_STT_CancelRepurchase '2'
  回购注销
- #define XTP_STT_CancelRegister '3'
  指定撤销
- #define XTP_STT_Register '4'
  指定登记
- #define XTP_STT_PurchaseIssue '5'
  买入发行申购
- #define XTP_STT_Allotment '6'
  卖出配股
- #define XTP_STT_SellTender '7'
  卖出要约收购
- #define XTP_STT_BuyTender '8'
  买入要约收购
- #define XTP_STT_NetVote '9'
  网上投票
- #define XTP_STT_SellConvertibleBonds 'a'
  卖出可转债回售
- #define XTP_STT_OptionExecute 'b'
  权证行权代码
- #define XTP_STT_PurchaseOF 'c'
  开放式基金申购
- #define XTP_STT_RedeemOF 'd'
  开放式基金赎回
- #define XTP_STT_SubscribeOF 'e'
  开放式基金认购
- #define XTP_STT_OFCustodianTranfer 'f'
  开放式基金转托管转出
- #define XTP_STT_OFDividendConfig 'g'
  开放式基金分红设置
- #define XTP_STT_OFTransfer 'h'
  开放式基金转成其他基金
- #define XTP_STT_BondsIn 'i'
  债券入库
- #define XTP_STT_BondsOut 'j'
  债券出库
- #define XTP_STT_PurchaseETF 'k'
  *EFT*申购
- #define XTP_STT_RedeemETF 'l'
  *EFT*赎回
- #define XTP_STT_ConvertibleRegister 'm'
  可转债回售登记
- #define XTP_HTAA_Base '1'
  基本
- #define XTP_FIOT_FundIO '1'
  出入金
- #define XTP_FIOT_Transfer '2'
  银期转帐
- #define XTP_FT_Deposite '1'
  银行存款
- #define XTP_FT_ItemFund '2'

分项资金

- #define XTP_FT_Company '3'
  公司调整
- #define XTP_FD_In '1'
  入金
- #define XTP_FD_Out '2'
  出金
- #define XTP_BF_ICBC '1'
  工商银行
- #define XTP_BF_ABC '2'
  农业银行
- #define XTP_BF_BC '3'
  中国银行
- #define XTP_BF_CBC '4'
  建设银行
- #define XTP_BF_BOC '5'
  交通银行
- #define XTP_BF_Other 'Z'
  其他银行
- #define XTP_FS_Record '1'
  已录入
- #define XTP_FS_Check '2'
  已复核
- #define XTP_FS_Charge '3'
  已冲销
- #define XTP_LF_Yes '0'
  是最后分片
- #define XTP_LF_No '1'
  不是最后分片
- #define XTP_CUSTT_Person '0'
  自然人
- #define XTP_CUSTT_Institution '1'
  机构户
- #define XTP_YNI_Yes '0'
  是
- #define XTP_YNI_No '1'
  否
- #define XTP_FPF_BEN '0'
  由受益方支付费用
- #define XTP_FPF_OUR '1'
  由发送方支付费用
- #define XTP_FPF_SHA '2'
  由发送方支付发起的费用，受益方支付接受的费用
- #define XTP_BAT_BankBook '1'
  银行存折
- #define XTP_BAT_SavingCard '2'
  储蓄卡
- #define XTP_BAT_CreditCard '3'
  信用卡
- #define XTP_BPWDF_NoCheck '0'
  不核对

- #define XTP_BPWDF_BlankCheck '1'
  明文核对
- #define XTP_BPWDF_EncryptCheck '2'
  密文核对
- #define XTP_TRFS_Normal '0'
  正常
- #define XTP_TRFS_Repealed '1'
  被冲正
- #define XTP_AVAF_Invalid '0'
  未确认
- #define XTP_AVAF_Valid '1'
  有效
- #define XTP_AVAF_Repeal '2'
  冲正
- #define XTP_RSA_Original '0'
  默认算法
- #define XTP_RSA_Ratio '1'
  按还券比例计算
- #define XTP_RSA_Min '2'
  Min[1,2].
- #define XTP_TS_Common '1'
  普通业务
- #define XTP_TS_Options '2'
  个股期权
- #define XTP_SST_Aboss '1'
  顶点系统
- #define XTP_SST_HS '2'
  恒生系统
- #define XTP_SMS_Allow '0'
  表示允许拆分和合并
- #define XTP_SMS_OnlySplit '1'
  允许拆分、不允许合并
- #define XTP_SMS_OnlyMerge '2'
  不允许拆分、允许合并
- #define XTP_SMS_Forbidden '3'
  不允许拆分和合并
- #define XTP_FITT_TransferIn '0'
  转入
- #define XTP_FITT_TransferOut '1'
  转出
- #define XTP_IT_Normal '0'
  普通
- #define XTP_IT_CallOptions '1'
  看涨期权
- #define XTP_IT_PutOptions '2'
  看跌期权
- #define XTP_IT_Normal_STEP '3'
  普通*(STEP)*
- #define XTP_IL_Level_1 '0'
  一级
- #define XTP_IL_Level_2 '1'

二级

- #define XTP_IL_Level_3 '2'
  三级
- #define XTP_CD_CloseBuy '!'
  买平仓
- #define XTP_CD_CloseSell '@'
  卖平仓
- #define XTP_CD_CloseCover '#'
  备兑平仓
- #define XTP_DT_ExecCallOptions '0'
  看涨期权执行
- #define XTP_DT_ExecPutOptions '1'
  看跌期权执行
- #define XTP_DT_UnavailStock '2'
  在途证券

类型定义

- typedef int TXTPErrorIDType
  *TXTPErrorIDType*是一个错误代码类型
- typedef char TXTPErrorMsgType[81]
  *TXTPErrorMsgType*是一个错误信息类型
- typedef XTP_EXCHANGE_TYPE TXTPExchangeIDType
  *TXTPExchangeIDType*是一个交易所代码类型
- typedef char TXTPExchangeNameType[31]
  *TXTPExchangeNameType*是一个交易所名称类型
- typedef char TXTPExchangePropertyType
  *TXTPExchangePropertyType*是一个交易所属性类型
- typedef char TXTPExchangeConnectStatusType
  *TXTPExchangeConnectStatusType*是一个交易所连接状态类型
- typedef char TXTPDateType[9]
  *TXTPDateType*是一个日期类型
- typedef char TXTPTimeType[9]
  *TXTPTimeType*是一个时间类型
- typedef char TXTPInstrumentIDType[31]
  *TXTPInstrumentIDType*是一个合约代码类型
- typedef char TXTPProductNameType[21]
  *TXTPProductNameType*是一个产品名称类型
- typedef char TXTPProductClassType
  *TXTPProductClassType*是一个产品类型类型
- typedef int TXTPVolumeMultipleType
  *TXTPVolumeMultipleType*是一个合约数量乘数类型
- typedef double TXTPPriceType
  *TXTPPriceType*是一个价格类型
- typedef int TXTPVolumeType
  *TXTPVolumeType*是一个数量类型
- typedef char TXTPPositionTypeType
  *TXTPPositionTypeType*是一个持仓类型类型
- typedef char TXTPPositionDateTypeType
  *TXTPPositionDateTypeType*是一个持仓日期类型类型

- typedef char TXTPExchangeInstIDType[31]
  *TXTPExchangeInstIDType*是一个合约在交易所的代码类型
- typedef int TXTPYearType
  *TXTPYearType*是一个年份类型
- typedef int TXTPMonthType
  *TXTPMonthType*是一个月份类型
- typedef char TXTPInstLifePhaseType
  *TXTPInstLifePhaseType*是一个合约生命周期状态类型
- typedef int TXTPBoolType
  *TXTPBoolType*是一个布尔型类型
- typedef char TXTPRightModelIDType[31]
  *TXTPRightModelIDType*是一个股票权限模版代码类型
- typedef char TXTPRightModelNameType[161]
  *TXTPRightModelNameType*是一个股票权限模版名称类型
- typedef char TXTPPosTradeTypeType
  *TXTPPosTradeTypeType*是一个持仓交易类型类型
- typedef char TXTPTraderIDType[21]
  *TXTPTraderIDType*是一个交易所交易员代码类型
- typedef char TXTPParticipantIDType[11]
  *TXTPParticipantIDType*是一个会员代码类型
- typedef char TXTPPasswordType[41]
  *TXTPPasswordType*是一个密码类型
- typedef char TXTPBrokerIDType[11]
  *TXTPBrokerIDType*是一个经纪公司代码类型
- typedef char TXTPOrderLocalIDType[13]
  *TXTPOrderLocalIDType*是一个本地报单编号类型
- typedef char TXTPBrokerAbbrType[9]
  *TXTPBrokerAbbrType*是一个经纪公司简称类型
- typedef char TXTPBrokerNameType[81]
  *TXTPBrokerNameType*是一个经纪公司名称类型
- typedef char TXTPInvestorIDType[15]
  *TXTPInvestorIDType*是一个投资者代码类型
- typedef char TXTPPartyNameType[81]
  *TXTPPartyNameType*是一个参与人名称类型
- typedef char TXTPIdCardTypeType
  *TXTPIdCardTypeType*是一个证件类型类型
- typedef char TXTPIdentifiedCardNoType[51]
  *TXTPIdentifiedCardNoType*是一个证件号码类型
- typedef char TXTPClientIDType[21]
  *TXTPClientIDType*是一个交易编码类型
- typedef char TXTPAccountIDType[15]
  *TXTPAccountIDType*是一个投资者帐号类型
- typedef char TXTPClientTypeType
  *TXTPClientTypeType*是一个交易编码类型类型
- typedef char TXTPInvestorGroupNameType[41]
  *TXTPInvestorGroupNameType*是一个投资者分组名称类型
- typedef char TXTPUserIDType[16]
  *TXTPUserIDType*是一个用户代码类型
- typedef char TXTPUserNameType[81]
  *TXTPUserNameType*是一个用户名称类型
- typedef char TXTPFunctionCodeType

*TXTPFunctionCodeType*是一个功能代码类型

- typedef char TXTPUserTypeType

  *TXTPUserTypeType*是一个用户类型类型
- typedef char TXTPBrokerFunctionCodeType

  *TXTPBrokerFunctionCodeType*是一个经纪公司功能代码类型
- typedef char TXTPCurrencyCodeType[4]

  *TXTPCurrencyCodeType*是一个币种类型
- typedef double TXTPMoneyType

  *TXTPMoneyType*是一个资金类型
- typedef double TXTPRatioType

  *TXTPRatioType*是一个比率类型
- typedef char TXTPAccountTypeType

  *TXTPAccountTypeType*是一个账户类型类型
- typedef char TXTPDepartmentRangeType

  *TXTPDepartmentRangeType*是一个投资者范围类型
- typedef char TXTPUserRightTypeType

  *TXTPUserRightTypeType*是一个客户权限类型类型
- typedef char TXTPProductInfoType[11]

  *TXTPProductInfoType*是一个产品信息类型
- typedef char TXTPAuthCodeType[17]

  *TXTPAuthCodeType*是一个客户端认证码类型
- typedef double TXTPLargeVolumeType

  *TXTPLargeVolumeType*是一个大额数量类型
- typedef int TXTPMillisecType

  *TXTPMillisecType*是一个时间（毫秒）类型
- typedef char TXTPHedgeFlagType

  *TXTPHedgeFlagType*是一个投机套保标志类型
- typedef char TXTPDirectionType

  *TXTPDirectionType*是一个买卖方向类型
- typedef char TXTPTradeIDType[21]

  *TXTPTradeIDType*是一个成交编号类型
- typedef char TXTPTradeTypeType

  *TXTPTradeTypeType*是一个成交类型类型
- typedef char TXTPCreationredemptionStatusType

  *TXTPCreationredemptionStatusType*是一个基金当天申购赎回状态类型
- typedef char TXTPETFCurrenceReplaceStatusType

  *TXTPETFCurrenceReplaceStatusType*是一个*ETF*现金替代标志类型
- typedef double TXTPInterestType

  *TXTPInterestType*是一个利息类型
- typedef double TXTPRepurchaseMaxTimesType

  *TXTPRepurchaseMaxTimesType*是一个正回购放大倍数类型
- typedef char TXTPCapitalStockTypeType

  *TXTPCapitalStockTypeType*是一个股本类型类型
- typedef char TXTPMarginPriceTypeType

  *TXTPMarginPriceTypeType*是一个保证金价格类型类型
- typedef char TXTPAlgorithmType

  *TXTPAlgorithmType*是一个盈亏算法类型
- typedef char TXTPIncludeCloseProfitType

  *TXTPIncludeCloseProfitType*是一个是否包含平仓盈利类型
- typedef char TXTPAllWithoutTradeType

  *TXTPAllWithoutTradeType*是一个是否受可提比例限制类型

- typedef char TXTPHandlePositionAlgoIDType
  *TXTPHandlePositionAlgoIDType*是一个持仓处理算法编号类型
- typedef char TXTPTradeParamIDType
  *TXTPTradeParamIDType*是一个交易系统参数代码类型
- typedef char TXTPSettlementParamValueType[256]
  *TXTPSettlementParamValueType*是一个参数代码值类型
- typedef char TXTPMemoType[161]
  *TXTPMemoType*是一个备注类型
- typedef int TXTPPriorityType
  *TXTPPriorityType*是一个优先级类型
- typedef char TXTPOrderRefType[13]
  *TXTPOrderRefType*是一个报单引用类型
- typedef char TXTPMarketIDType[31]
  *TXTPMarketIDType*是一个市场代码类型
- typedef char TXTPMacAddressType[21]
  *TXTPMacAddressType*是一个*Mac*地址类型
- typedef char TXTPInstrumentNameType[21]
  *TXTPInstrumentNameType*是一个合约名称类型
- typedef char TXTPOrderSysIDType[21]
  *TXTPOrderSysIDType*是一个报单编号类型
- typedef char TXTPIPAddressType[16]
  *TXTPIPAddressType*是一个*IP*地址类型
- typedef int TXTPIPPortType
  *TXTPIPPortType*是一个*IP*端口类型
- typedef char TXTPProtocolInfoType[11]
  *TXTPProtocolInfoType*是一个协议信息类型
- typedef char TXTPDepositSeqNoType[15]
  *TXTPDepositSeqNoType*是一个出入金流水号类型
- typedef char TXTPSystemNameType[41]
  *TXTPSystemNameType*是一个系统名称类型
- typedef char TXTPInvestorRangeType
  *TXTPInvestorRangeType*是一个投资者范围类型
- typedef char TXTPDataSyncStatusType
  *TXTPDataSyncStatusType*是一个数据同步状态类型
- typedef char TXTPTraderConnectStatusType
  *TXTPTraderConnectStatusType*是一个交易所交易员连接状态类型
- typedef char TXTPOrderActionStatusType
  *TXTPOrderActionStatusType*是一个报单操作状态类型
- typedef char TXTPOrderStatusType
  *TXTPOrderStatusType*是一个报单状态类型
- typedef char TXTPOrderSubmitStatusType
  *TXTPOrderSubmitStatusType*是一个报单提交状态类型
- typedef char TXTPPositionDateType
  *TXTPPositionDateType*是一个持仓日期类型
- typedef char TXTPTradingRoleType
  *TXTPTradingRoleType*是一个交易角色类型
- typedef char TXTPPosiDirectionType
  *TXTPPosiDirectionType*是一个持仓多空方向类型
- typedef char TXTPOrderPriceTypeType
  *TXTPOrderPriceTypeType*是一个报单价格条件类型
- typedef char TXTPOffsetFlagType

*TXTPOffsetFlagType*是一个开平标志类型

- typedef char TXTPForceCloseReasonType

  *TXTPForceCloseReasonType*是一个强平原因类型
- typedef char TXTPOrderTypeType

  *TXTPOrderTypeType*是一个报单类型类型
- typedef char TXTPTimeConditionType

  *TXTPTimeConditionType*是一个有效期类型类型
- typedef char TXTPVolumeConditionType

  *TXTPVolumeConditionType*是一个成交量类型类型
- typedef char TXTPContingentConditionType

  *TXTPContingentConditionType*是一个触发条件类型
- typedef char TXTPActionFlagType

  *TXTPActionFlagType*是一个操作标志类型
- typedef char TXTPTradingRightType

  *TXTPTradingRightType*是一个交易权限类型
- typedef char TXTPOrderSourceType

  *TXTPOrderSourceType*是一个报单来源类型
- typedef char TXTPPriceSourceType

  *TXTPPriceSourceType*是一个成交价来源类型
- typedef int TXTPOrderActionRefType

  *TXTPOrderActionRefType*是一个报单操作引用类型
- typedef int TXTPFrontIDType

  *TXTPFrontIDType*是一个前置编号类型
- typedef int TXTPSessionIDType

  *TXTPSessionIDType*是一个会话编号类型
- typedef int TXTPInstallIDType

  *TXTPInstallIDType*是一个安装编号类型
- typedef int TXTPSequenceNoType

  *TXTPSequenceNoType*是一个序号类型
- typedef int TXTPRequestIDType

  *TXTPRequestIDType*是一个请求编号类型
- typedef char TXTPCombOffsetFlagType[5]

  *TXTPCombOffsetFlagType*是一个组合开平标志类型
- typedef char TXTPCombHedgeFlagType[5]

  *TXTPCombHedgeFlagType*是一个组合投机套保标志类型
- typedef short TXTPSequenceSeriesType

  *TXTPSequenceSeriesType*是一个序列系列号类型
- typedef short TXTPCommPhaseNoType

  *TXTPCommPhaseNoType*是一个通讯时段编号类型
- typedef char TXTPUserEventTypeType

  *TXTPUserEventTypeType*是一个用户事件类型类型
- typedef char TXTPUserEventInfoType[1025]

  *TXTPUserEventInfoType*是一个用户事件信息类型
- typedef char TXTPOTPTypeType

  *TXTPOTPTypeType*是一个动态令牌类型类型
- typedef char TXTPTradeSourceType

  *TXTPTradeSourceType*是一个成交来源类型
- typedef char TXTPBranchIDType[9]

  *TXTPBranchIDType*是一个营业部编号类型
- typedef char TXTPStockPriceType[16]

  *TXTPStockPriceType*是一个证券交易价格类型

- typedef char TXTPSerialNumberType[17]
  *TXTPSerialNumberType*是一个序列号类型
- typedef char TXTPInstrumentRangeType
  *TXTPInstrumentRangeType*是一个股票权限分类类型
- typedef char TXTPBusinessUnitType[21]
  *TXTPBusinessUnitType*是一个业务单元类型
- typedef char TXTPOTPVendorsIDType[2]
  *TXTPOTPVendorsIDType*是一个动态令牌提供商类型
- typedef int TXTPLastDriftType
  *TXTPLastDriftType*是一个上次*OTP*漂移值类型
- typedef int TXTPLastSuccessType
  *TXTPLastSuccessType*是一个上次*OTP*成功值类型
- typedef char TXTPAuthKeyType[41]
  *TXTPAuthKeyType*是一个令牌密钥类型
- typedef int TXTPUserSessionHashType
  *TXTPUserSessionHashType*是一个用户会话*Hash*值类型
- typedef char TXTPStockTradeTypeType
  *TXTPStockTradeTypeType*是一个证券交易类型类型
- typedef char TXTPHandleTradingAccountAlgoIDType
  *TXTPHandleTradingAccountAlgoIDType*是一个资金处理算法编号类型
- typedef int TXTPStockWthType
  *TXTPStockWthType*是一个股票使用流水号类型
- typedef char TXTPStockSeqType[17]
  *TXTPStockSeqType*是一个股票使用流水号类型
- typedef int TXTPWTFSType
  *TXTPWTFSType*是一个委托方式类型
- typedef int TXTPWTLBType
  *TXTPWTLBType*是一个委托类别类型
- typedef int TXTPWTRQType
  *TXTPWTRQType*是一个委托日期类型
- typedef int TXTPINTEGERType
  *TXTPINTEGERType*是一个一般整型类型
- typedef int TXTPINT3Type
  *TXTPINT3Type*是一个三位数整型类型
- typedef int TXTPINT6Type
  *TXTPINT6Type*是一个六位数整型类型
- typedef int TXTPINT12Type
  *TXTPINT12Type*是一个十二位数整型类型
- typedef char TXTPCHAR1Type[2]
  *TXTPCHAR1Type*是一个一字节*CHAR*类型
- typedef char TXTPCHAR2Type[3]
  *TXTPCHAR2Type*是一个二字节*CHAR*类型
- typedef char TXTPCHAR3Type[4]
  *TXTPCHAR3Type*是一个三字节*CHAR*类型
- typedef char TXTPCHAR4Type[5]
  *TXTPCHAR4Type*是一个四字节*CHAR*类型
- typedef char TXTPCHAR5Type[6]
  *TXTPCHAR5Type*是一个五字节*CHAR*类型
- typedef char TXTPCHAR6Type[7]
  *TXTPCHAR6Type*是一个六字节*CHAR*类型
- typedef char TXTPCHAR8Type[9]

*TXTPCHAR8Type*是一个八字节*CHAR*类型

- typedef char TXTPCHAR10Type[11]

  *TXTPCHAR10Type*是一个十字节*CHAR*类型
- typedef char TXTPCHAR11Type[12]

  *TXTPCHAR11Type*是一个十一字节*CHAR*类型
- typedef char TXTPCHAR12Type[13]

  *TXTPCHAR12Type*是一个十二字节*CHAR*类型
- typedef char TXTPCHAR13Type[14]

  *TXTPCHAR13Type*是一个十三字节*CHAR*类型
- typedef char TXTPCHAR14Type[15]

  *TXTPCHAR14Type*是一个十四字节*CHAR*类型
- typedef char TXTPCHAR16Type[17]

  *TXTPCHAR16Type*是一个十六字节*CHAR*类型
- typedef char TXTPCHAR19Type[20]

  *TXTPCHAR19Type*是一个十九字节*CHAR*类型
- typedef char TXTPCHAR20Type[21]

  *TXTPCHAR20Type*是一个二十字节*CHAR*类型
- typedef char TXTPCHAR21Type[22]

  *TXTPCHAR21Type*是一个二十一字节*CHAR*类型
- typedef char TXTPCHAR23Type[24]

  *TXTPCHAR23Type*是一个二十三字节*CHAR*类型
- typedef char TXTPCHAR30Type[31]

  *TXTPCHAR30Type*是一个三十字节*CHAR*类型
- typedef char TXTPCHAR32Type[33]

  *TXTPCHAR32Type*是一个三十二字节*CHAR*类型
- typedef char TXTPCHAR50Type[51]

  *TXTPCHAR50Type*是一个五十字节*CHAR*类型
- typedef char TXTPCHAR64Type[65]

  *TXTPCHAR64Type*是一个六十四字节*CHAR*类型
- typedef char TXTPCHAR65Type[66]

  *TXTPCHAR65Type*是一个六十五字节*CHAR*类型
- typedef char TXTPVCHAR4Type[5]

  *TXTPVCHAR4Type*是一个四字节*VCHAR*类型
- typedef char TXTPVCHAR6Type[7]

  *TXTPVCHAR6Type*是一个六字节*VCHAR*类型
- typedef char TXTPVCHAR8Type[9]

  *TXTPVCHAR8Type*是一个八字节*VCHAR*类型
- typedef char TXTPVCHAR10Type[11]

  *TXTPVCHAR10Type*是一个十字节*VCHAR*类型
- typedef char TXTPVCHAR12Type[13]

  *TXTPVCHAR12Type*是一个十二字节*VCHAR*类型
- typedef char TXTPVCHAR16Type[17]

  *TXTPVCHAR16Type*是一个十六字节*VCHAR*类型
- typedef char TXTPVCHAR20Type[21]

  *TXTPVCHAR20Type*是一个二十字节*VCHAR*类型
- typedef char TXTPVCHAR30Type[31]

  *TXTPVCHAR30Type*是一个三十字节*VCHAR*类型
- typedef char TXTPVCHAR50Type[51]

  *TXTPVCHAR50Type*是一个五十字节*VCHAR*类型
- typedef char TXTPVCHAR60Type[61]

  *TXTPVCHAR60Type*是一个六十字节*VCHAR*类型

- typedef char TXTPVCHAR65Type[66]
  *TXTPVCHAR65Type*是一个六十五字节*VCHAR*类型
- typedef char TXTPVCHAR80Type[81]
  *TXTPVCHAR80Type*是一个八十字节*VCHAR*类型
- typedef char TXTPVCHAR84Type[85]
  *TXTPVCHAR84Type*是一个八十四字节*VCHAR*类型
- typedef char TXTPVCHAR255Type[256]
  *TXTPVCHAR255Type*是一个二五五字节*VCHAR*类型
- typedef char TXTPVCHAR1024Type[1025]
  *TXTPVCHAR1024Type*是一个一零二四字节*VCHAR*类型
- typedef double TXTPREAL8P3Type
  *TXTPREAL8P3Type*是一个八点三实型类型
- typedef double TXTPREAL9P3Type
  *TXTPREAL9P3Type*是一个九点三实型类型
- typedef double TXTPREAL9P6Type
  *TXTPREAL9P6Type*是一个九点六实型类型
- typedef double TXTPREAL10P4Type
  *TXTPREAL10P4Type*是一个十点四实型类型
- typedef double TXTPREAL16P2Type
  *TXTPREAL16P2Type*是一个十六点二实型类型
- typedef double TXTPREAL16P8Type
  *TXTPREAL16P8Type*是一个十六点八实型类型
- typedef double TXTPREAL22P2Type
  *TXTPREAL22P2Type*是一个二十二点二实型类型
- typedef int TXTPCommandNoType
  *TXTPCommandNoType*是一个*DB*命令序号类型
- typedef char TXTPCommandTypeType[65]
  *TXTPCommandTypeType*是一个*DB*命令类型类型
- typedef char TXTPSettlementGroupIDType[9]
  *TXTPSettlementGroupIDType*是一个结算组代码类型
- typedef char TXTPFieldNameType[2049]
  *TXTPFieldNameType*是一个字段名类型
- typedef char TXTPFieldContentType[2049]
  *TXTPFieldContentType*是一个字段内容类型
- typedef char TXTPBankIDType[4]
  *TXTPBankIDType*是一个银行代码类型
- typedef char TXTPBankNameType[101]
  *TXTPBankNameType*是一个银行名称类型
- typedef char TXTPBankBrchIDType[5]
  *TXTPBankBrchIDType*是一个银行分中心代码类型
- typedef int TXTPLiberSerialType
  *TXTPLiberSerialType*是一个*Liber*系统流水号类型
- typedef char TXTPRoleIDType[11]
  *TXTPRoleIDType*是一个角色编号类型
- typedef char TXTPRoleNameType[41]
  *TXTPRoleNameType*是一个角色名称类型
- typedef char TXTPDescriptionType[401]
  *TXTPDescriptionType*是一个描述类型
- typedef char TXTPFunctionIDType[25]
  *TXTPFunctionIDType*是一个功能代码类型
- typedef char TXTPBillNoType[15]

*TXTPBillNoType*是一个票据号类型

- typedef char TXTPFundIOTypeType

  *TXTPFundIOTypeType*是一个出入金类型类型
- typedef char TXTPFundTypeType

  *TXTPFundTypeType*是一个资金类型类型
- typedef char TXTPFundDirectionType

  *TXTPFundDirectionType*是一个出入金方向类型
- typedef char TXTPBankFlagType

  *TXTPBankFlagType*是一个银行统一标识类型类型
- typedef char TXTPOperationMemoType[1025]

  *TXTPOperationMemoType*是一个操作摘要类型
- typedef char TXTPFundStatusType

  *TXTPFundStatusType*是一个资金状态类型
- typedef char TXTPFundProjectIDType[5]

  *TXTPFundProjectIDType*是一个资金项目编号类型
- typedef char TXTPOperatorIDType[65]

  *TXTPOperatorIDType*是一个操作员代码类型
- typedef char TXTPCounterIDType[33]

  *TXTPCounterIDType*是一个计数器代码类型
- typedef char TXTPFunctionNameType[65]

  *TXTPFunctionNameType*是一个功能名称类型
- typedef char TXTPTradeCodeType[7]

  *TXTPTradeCodeType*是一个交易代码类型
- typedef char TXTPBrokerBranchIDType[31]

  *TXTPBrokerBranchIDType*是一个经纪公司分支机构代码类型
- typedef char TXTPTradeDateType[9]

  *TXTPTradeDateType*是一个交易日期类型
- typedef char TXTPTradeTimeType[9]

  *TXTPTradeTimeType*是一个交易时间类型
- typedef char TXTPBankSerialType[13]

  *TXTPBankSerialType*是一个银行流水号类型
- typedef int TXTPSerialType

  *TXTPSerialType*是一个流水号类型
- typedef char TXTPLastFragmentType

  *TXTPLastFragmentType*是一个最后分片标志类型
- typedef char TXTPIndividualNameType[51]

  *TXTPIndividualNameType*是一个个人姓名类型
- typedef char TXTPCustTypeType

  *TXTPCustTypeType*是一个客户类型类型
- typedef char TXTPBankAccountType[41]

  *TXTPBankAccountType*是一个银行账户类型
- typedef char TXTPYesNoIndicatorType

  *TXTPYesNoIndicatorType*是一个是或否标识类型
- typedef double TXTPTradeAmountType

  *TXTPTradeAmountType*是一个交易金额（元）类型
- typedef double TXTPCustFeeType

  *TXTPCustFeeType*是一个应收客户费用（元）类型
- typedef double TXTPBrokerFeeType

  *TXTPBrokerFeeType*是一个应收经纪公司费用（元）类型
- typedef char TXTPFeePayFlagType

  *TXTPFeePayFlagType*是一个费用支付标志类型

- typedef char TXTPAddInfoType[129]
  *TXTPAddInfoType*是一个附加信息类型
- typedef char TXTPDigestType[36]
  *TXTPDigestType*是一个摘要类型
- typedef char TXTPBankAccTypeType
  *TXTPBankAccTypeType*是一个银行帐号类型类型
- typedef char TXTPDeviceIDType[3]
  *TXTPDeviceIDType*是一个渠道标志类型
- typedef char TXTPPwdFlagType
  *TXTPPwdFlagType*是一个密码核对标志类型
- typedef char TXTPBankCodingForBrokerType[33]
  *TXTPBankCodingForBrokerType*是一个银行对经纪公司的编码类型
- typedef char TXTPOperNoType[17]
  *TXTPOperNoType*是一个交易柜员类型
- typedef int TXTPTIDType
  *TXTPTIDType*是一个交易*ID*类型
- typedef char TXTPTransferStatusType
  *TXTPTransferStatusType*是一个转账交易状态类型
- typedef int TXTPPlateSerialType
  *TXTPPlateSerialType*是一个平台流水号类型
- typedef char TXTPAvailabilityFlagType
  *TXTPAvailabilityFlagType*是一个有效标志类型
- typedef char TXTPOperatorCodeType[17]
  *TXTPOperatorCodeType*是一个操作员类型
- typedef char TXTPRepayStockAlgoType
  *TXTPRepayStockAlgoType*是一个买券还券算法类型
- typedef char TXTPTradeSpanType
  *TXTPTradeSpanType*是一个交易时间段类型类型
- typedef char TXTPSettleSystemTypeType
  *TXTPSettleSystemTypeType*是一个所属结算系统类型类型
- typedef char TXTPLogLevelType[33]
  *TXTPLogLevelType*是一个日志级别类型
- typedef char TXTPProcessNameType[257]
  *TXTPProcessNameType*是一个存储过程名称类型
- typedef char TXTPTemplateIDType[9]
  *TXTPTemplateIDType*是一个模板代码类型
- typedef int TXTPTradeIndexType
  *TXTPTradeIndexType*是一个成交序号类型
- typedef char TXTPSplitMergeStatusType
  *TXTPSplitMergeStatusType*是一个基金当天拆分合并状态类型
- typedef char TXTPFundInterTransferTypeType
  *TXTPFundInterTransferTypeType*是一个资金内转类型类型
- typedef char TXTPInstrumentTypeType
  *TXTPInstrumentTypeType*是一个合约类型类型
- typedef char TXTPInvestorLevelType
  *TXTPInvestorLevelType*是一个投资者期权交易等级类型
- typedef char TXTPCloseDirectionType
  *TXTPCloseDirectionType*是一个平仓方向类型
- typedef char TXTPDelivTypeType
  *TXTPDelivTypeType*是一个交割类型类型

枚举

- enum XTP_PROTOCOL_TYPE { XTP_PROTOCOL_TCP = 1, XTP_PROTOCOL_UDP }

  是一个通讯传输协议方式
- enum XTP_EXCHANGE_TYPE { XTP_EXCHANGE_SH = 1, XTP_EXCHANGE_SZ, XTP_EXCHANGE_INVALID }

  是交易所类型
- enum XTP_MARKET_TYPE {
  XTP_MKT_SZ_A = 0, XTP_MKT_SH_A, XTP_MKT_SZ_B, XTP_MKT_SH_B,
  XTP_MKT_MAX }

  市场类型
- enum XTP_PRICE_TYPE {
  XTP_PRICE_LIMIT = 1, XTP_PRICE_BEST_OR_CANCEL, XTP_PRICE_BEST5_OR_LIMIT, XTP_PRICE_BEST5_OR_CANCEL,
  XTP_PRICE_ALL_OR_CANCEL, XTP_PRICE_FORWARD_BEST, XTP_PRICE_REVERSE_BEST_LIMIT,
  XTP_PRICE_TYPE_MAX }

  是一个价格类型
- enum XTP_SIDE_TYPE {
  XTP_SIDE_BUY = 1, XTP_SIDE_SELL, XTP_SIDE_BUY_OPEN, XTP_SIDE_SELL_OPEN,
  XTP_SIDE_BUY_CLOSE, XTP_SIDE_SELL_CLOSE }

  是一个买卖方向类型
- enum XTP_ORDER_ACTION_STATUS_TYPE { XTP_ORDER_ACTION_STATUS_SUBMITTED = 1, XTP_ORDER_ACTION_STATUS_ACCEPTED, XTP_ORDER_ACTION_STATUS_REJECTED }

  是一个报单操作状态类型
- enum XTP_ORDER_STATUS_TYPE {
  XTP_ORDER_STATUS_INIT = 0, XTP_ORDER_STATUS_ALLTRADED = 1, XTP_ORDER_STATUS_PARTTRADEDQUEUEING, XTP_ORDER_STATUS_PARTTRADEDNOTQUEUEING,
  XTP_ORDER_STATUS_NOTRADEQUEUEING, XTP_ORDER_STATUS_CANCELED, XTP_ORDER_STATUS_REJECTED, XTP_ORDER_STATUS_UNKNOWN }

  *XTP_ORDER_STATUS_TYPE*是一个报单状态类型
- 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_STATUS_CANCEL_SUBMITTED,
  XTP_ORDER_SUBMIT_STATUS_CANCEL_REJECTED, XTP_ORDER_SUBMIT_STATUS_CANCEL_ACCEPTED }

  *XTP_ORDER_SUBMIT_STATUS_TYPE*是一个报单提交状态类型
- enum XTP_TE_RESUME_TYPE { XTP_TERT_RESTART = 0, XTP_TERT_RESUME, XTP_TERT_QUICK }

  是一个私有流重传方式

6.5.1 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

6.5.2 枚举类型说明

6.5.2.1 enum XTP_EXCHANGE_TYPE

是交易所类型

枚举值

XTP_EXCHANGE_SH 上证

XTP_EXCHANGE_SZ 深证

XTP_EXCHANGE_INVALID 不存在的交易所类型

6.5.2.2 enum XTP_MARKET_TYPE

市场类型

枚举值

XTP_MKT_SZ_A 深圳A股

XTP_MKT_SH_A 上海A股

XTP_MKT_SZ_B 深圳B股

XTP_MKT_SH_B 上海B股

XTP_MKT_MAX 市场类型个数

6.5.2.3 enum XTP_ORDER_ACTION_STATUS_TYPE

是一个报单操作状态类型

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

6.5.2.4 enum XTP_ORDER_STATUS_TYPE

XTP_ORDER_STATUS_TYPE是一个报单状态类型

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

6.5.2.5 enum XTP_ORDER_SUBMIT_STATUS_TYPE

XTP_ORDER_SUBMIT_STATUS_TYPE是一个报单提交状态类型

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

6.5.2.6 enum XTP_PRICE_TYPE

是一个价格类型

枚举值

XTP_PRICE_LIMIT 限价单
XTP_PRICE_BEST_OR_CANCEL 即时成交剩余转撤销，市价单
XTP_PRICE_BEST5_OR_LIMIT 最优五档即时成交剩余转限价，市价单
XTP_PRICE_BEST5_OR_CANCEL 最优5档即时成交剩余转撤销，市价单
XTP_PRICE_ALL_OR_CANCEL 全部成交或撤销,市价单
XTP_PRICE_FORWARD_BEST 本方最优，市价单
XTP_PRICE_REVERSE_BEST_LIMIT 对方最优剩余转限价，市价单
XTP_PRICE_TYPE_MAX 价格类型个数

6.5.2.7 enum XTP_PROTOCOL_TYPE

是一个通讯传输协议方式

枚举值

XTP_PROTOCOL_TCP 采用TCP方式传输
XTP_PROTOCOL_UDP 采用UDP方式传输

6.5.2.8 enum XTP_SIDE_TYPE

是一个买卖方向类型

枚举值

XTP_SIDE_BUY 买
XTP_SIDE_SELL 卖
XTP_SIDE_BUY_OPEN 买开
XTP_SIDE_SELL_OPEN 卖开
XTP_SIDE_BUY_CLOSE 买平
XTP_SIDE_SELL_CLOSE 卖平

6.5.2.9 enum XTP_TE_RESUME_TYPE

是一个私有流重传方式

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

XTP_TERT_QUICK 只传送登录后私有流的内容

6.6 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"
```

6.6.1 详细描述

定义业务数据结构

作者

中泰证券股份有限公司

6.7 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 76
  错误字串长度

类型定义

- typedef struct XTPRspInfoStruct XTPRI
  响应信息

6.7.1 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

6.8 xtp_trader_api.h 文件参考

定义客户端交易接口

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

结构体

- class TraderSpi
  交易接口响应类
- class TraderApi
  交易接口类

6.8.1 详细描述

定义客户端交易接口

作者

中泰证券股份有限公司

Index