## XTP极速交易系统TraderAPI

制作者 中泰证券股份有限公司

## **Contents**

1	XTP	极速交	易系统 T	rader API 2.2.33.5	1
2	继承	关系索	:引		3
	2.1	类继承	关系		3
3	结构	体索引			5
	3.1	结构体	ž		5
4	文件	索引			9
	4.1	文件列	表		9
5	结构	体说明			11
	5.1	Demo1	estTrader	Spi类 参考	11
		5.1.1	详细描述	፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟፟	11
		5.1.2	成员函数	女说明	12
			5.1.2.1	OnAlgoConnected()	12
			5.1.2.2	OnAlgoDisconnected()	12
			5.1.2.3	OnALGOUserEstablishChannel()	12
			5.1.2.4	OnCancelAlgoOrder()	13
			5.1.2.5	OnDisconnected()	14
			5.1.2.6	OnInsertAlgoOrder()	14
			5.1.2.7	OnQueryStrategy()	15
			5.1.2.8	OnStrategyStateReport()	15
	5.2	OrderE	BookStruct	结构体 参考	16
		5.2.1	详细描述	₺	16

ii CONTENTS

5.3	Trader	Api类 参考		17
	5.3.1	详细描述	<u> </u>	19
	5.3.2	成员函数	y说明	19
		5.3.2.1	ALGOUserEstablishChannel()	19
		5.3.2.2	CancelAlgoOrder()	20
		5.3.2.3	CancelOptionCombinedOrder()	21
		5.3.2.4	CancelOrder()	21
		5.3.2.5	CreateTraderApi()	22
		5.3.2.6	CreditCashRepay()	22
		5.3.2.7	CreditCashRepayDebtInterestFee()	23
		5.3.2.8	CreditExtendDebtDate()	23
		5.3.2.9	CreditSellStockRepayDebtInterestFee()	24
		5.3.2.10	FundTransfer()	24
		5.3.2.11	GetAccountByXTPID()	25
		5.3.2.12	GetAlgorithmIDByOrder()	25
		5.3.2.13	GetANewOrderXTPID()	26
		5.3.2.14	GetApiLastError()	26
		5.3.2.15	GetApiVersion()	27
		5.3.2.16	GetClientIDByXTPID()	27
		5.3.2.17	GetTradingDay()	27
		5.3.2.18	InsertAlgoOrder()	28
		5.3.2.19	InsertOptionCombinedOrder()	28
		5.3.2.20	InsertOptionCombinedOrderExtra()	29
		5.3.2.21	InsertOrder()	29
		5.3.2.22	InsertOrderExtra()	30
		5.3.2.23	IsServerRestart()	30
		5.3.2.24	Login()	31
		5.3.2.25	LoginALGO()	32
		5.3.2.26	Logout()	32
		5.3.2.27	ModifyUserTerminalInfo()	33

CONTENTS

5.3.2.28	QueryAccountTradeMarket()	33
5.3.2.29	QueryAsset()	34
5.3.2.30	QueryCreditAssetDebtInfo()	34
5.3.2.31	QueryCreditCashRepayInfo()	35
5.3.2.32	QueryCreditDebtInfo()	35
5.3.2.33	QueryCreditExcessStock()	36
5.3.2.34	QueryCreditExtendDebtDateOrders()	36
5.3.2.35	QueryCreditFundExtraInfo()	37
5.3.2.36	QueryCreditFundInfo()	37
5.3.2.37	QueryCreditPositionExtraInfo()	37
5.3.2.38	QueryCreditTickerAssignInfo()	38
5.3.2.39	QueryCreditTickerDebtInfo()	38
5.3.2.40	QueryETF()	40
5.3.2.41	QueryETFTickerBasket()	40
5.3.2.42	QueryFundTransfer()	41
5.3.2.43	QueryIPOInfoList()	41
5.3.2.44	QueryIPOQuotaInfo()	41
5.3.2.45	QueryMulCreditExcessStock()	42
5.3.2.46	QueryOptionAuctionInfo()	42
5.3.2.47	QueryOptionCombinedExecPosition()	43
5.3.2.48	QueryOptionCombinedOrderByXTPID()	43
5.3.2.49	QueryOptionCombinedOrderByXTPIDEx()	44
5.3.2.50	QueryOptionCombinedOrders()	44
5.3.2.51	QueryOptionCombinedOrdersByPage()	45
5.3.2.52	QueryOptionCombinedOrdersByPageEx()	45
5.3.2.53	QueryOptionCombinedOrdersEx()	46
5.3.2.54	QueryOptionCombinedPosition()	47
5.3.2.55	QueryOptionCombinedStrategyInfo()	47
5.3.2.56	QueryOptionCombinedTrades()	48
5.3.2.57	QueryOptionCombinedTradesByPage()	48

iv CONTENTS

		5.3.2.58	QueryOptionCombinedTradesByXTPID()	49
		5.3.2.59	QueryOptionCombinedUnfinishedOrders()	49
		5.3.2.60	QueryOptionCombinedUnfinishedOrdersEx()	50
		5.3.2.61	QueryOrderByXTPID()	50
		5.3.2.62	QueryOrderByXTPIDEx()	51
		5.3.2.63	QueryOrders()	51
		5.3.2.64	QueryOrdersByPage()	52
		5.3.2.65	QueryOrdersByPageEx()	52
		5.3.2.66	QueryOrdersEx()	53
		5.3.2.67	QueryOtherServerFund()	53
		5.3.2.68	QueryPosition()	54
		5.3.2.69	QueryStrategy()	55
		5.3.2.70	QueryStructuredFund()	55
		5.3.2.71	QueryTrades()	56
		5.3.2.72	QueryTradesByPage()	56
		5.3.2.73	QueryTradesByXTPID()	57
		5.3.2.74	QueryUnfinishedOrders()	57
		5.3.2.75	QueryUnfinishedOrdersEx()	58
		5.3.2.76	RegisterSpi()	58
		5.3.2.77	Release()	59
		5.3.2.78	SetHeartBeatInterval()	59
		5.3.2.79	SetSoftwareKey()	59
		5.3.2.80	SetSoftwareVersion()	60
		5.3.2.81	SubscribePublicTopic()	60
5.4	Trader	Spi类 参考	<del>,</del>	60
	5.4.1	详细描述	₾	63
	5.4.2	成员函数	女说明	63
		5.4.2.1	OnAlgoDisconnected()	63
		5.4.2.2	OnALGOUserEstablishChannel()	63
		5.4.2.3	OnCancelAlgoOrder()	64

CONTENTS

5.4.2.4	OnCancelOptionCombinedOrderError()	64
5.4.2.5	OnCancelOrderError()	65
5.4.2.6	OnCreditCashRepay()	65
5.4.2.7	OnCreditCashRepayDebtInterestFee()	66
5.4.2.8	OnCreditExtendDebtDate()	66
5.4.2.9	OnDisconnected()	67
5.4.2.10	OnError()	67
5.4.2.11	OnFundTransfer()	68
5.4.2.12	OnInsertAlgoOrder()	68
5.4.2.13	OnOptionCombinedOrderEvent()	69
5.4.2.14	OnOptionCombinedTradeEvent()	69
5.4.2.15	OnOrderEvent()	71
5.4.2.16	OnQueryAccountTradeMarket()	72
5.4.2.17	OnQueryAsset()	72
5.4.2.18	OnQueryCreditAssetDebtInfo()	73
5.4.2.19	OnQueryCreditCashRepayInfo()	73
5.4.2.20	OnQueryCreditDebtInfo()	74
5.4.2.21	OnQueryCreditExcessStock()	74
5.4.2.22	OnQueryCreditExtendDebtDateOrders()	75
5.4.2.23	OnQueryCreditFundExtraInfo()	75
5.4.2.24	OnQueryCreditFundInfo()	77
5.4.2.25	OnQueryCreditPositionExtraInfo()	77
5.4.2.26	OnQueryCreditTickerAssignInfo()	78
5.4.2.27	OnQueryCreditTickerDebtInfo()	79
5.4.2.28	OnQueryETF()	79
5.4.2.29	OnQueryETFBasket()	80
5.4.2.30	OnQueryFundTransfer()	80
5.4.2.31	OnQueryIPOInfoList()	81
5.4.2.32	OnQueryIPOQuotaInfo()	81
5.4.2.33	OnQueryMulCreditExcessStock()	82

vi CONTENTS

	5.4.2.34	OnQueryOptionAuctionInfo()	83
	5.4.2.35	OnQueryOptionCombinedExecPosition()	83
	5.4.2.36	OnQueryOptionCombinedOrders()	84
	5.4.2.37	OnQueryOptionCombinedOrdersByPage()	84
	5.4.2.38	OnQueryOptionCombinedOrdersByPageEx()	85
	5.4.2.39	OnQueryOptionCombinedOrdersEx()	86
	5.4.2.40	OnQueryOptionCombinedPosition()	86
	5.4.2.41	OnQueryOptionCombinedStrategyInfo()	87
	5.4.2.42	OnQueryOptionCombinedTrades()	87
	5.4.2.43	OnQueryOptionCombinedTradesByPage()	88
	5.4.2.44	OnQueryOrder()	89
	5.4.2.45	OnQueryOrderByPage()	89
	5.4.2.46	OnQueryOrderByPageEx()	90
	5.4.2.47	OnQueryOrderEx()	91
	5.4.2.48	OnQueryOtherServerFund()	91
	5.4.2.49	OnQueryPosition()	92
	5.4.2.50	OnQueryStrategy()	92
	5.4.2.51	OnQueryStructuredFund()	93
	5.4.2.52	OnQueryTrade()	94
	5.4.2.53	OnQueryTradeByPage()	94
	5.4.2.54	OnStrategyStateReport()	95
	5.4.2.55	OnStrategySymbolStateReport()	95
	5.4.2.56	OnTradeEvent()	96
5.5	XTPClientQuery	CrdDebtStockReq结构体 参考	96
	5.5.1 详细描述	ᡮ	97
5.6	XTPClientQuery	CrdPositionStkInfo结构体 参考	97
	5.6.1 详细描述	盘	97
5.7	XTPClientQuery	CrdPositionStockReq结构体参考	98
	5.7.1 详细描述	术	98
5.8	XTPClientQuery	CrdSurplusStkReqInfo结构体 参考	98

CONTENTS vii

	5.8.1	详细描述	98
5.9	XTPCli	entQueryCrdSurplusStkRspInfo结构体 参考......................	99
	5.9.1	详细描述	99
5.10	XTPCo	mbLegStrategy结构体 参考	99
	5.10.1	详细描述	100
5.11	XTPCro	dCashRepayDebtInterestFeeRsp结构体参考	100
	5.11.1	详细描述	100
5.12	XTPCro	dCashRepayInfo结构体 参考	100
	5.12.1	详细描述	101
5.13	XTPCro	dCashRepayRsp结构体 参考	101
	5.13.1	详细描述	101
5.14	XTPCro	dDebtInfo结构体 参考....................................	102
	5.14.1	详细描述	102
5.15	XTPCro	dDebtStockInfo结构体 参考	103
	5.15.1	详细描述	103
5.16	XTPCro	dFundExtraInfo结构体 参考	103
	5.16.1	详细描述	103
5.17	XTPCro	dFundInfo结构体 参考	104
	5.17.1	详细描述	104
5.18	XTPCro	dPositionExtraInfo结构体 参考	104
	5.18.1	详细描述	105
5.19	XTPCre	editDebtExtendNotice结构体 参考	105
	5.19.1	详细描述	105
5.20	XTPCre	editDebtExtendReq结构体 参考	105
	5.20.1	详细描述	106
5.21	XTPFu	ndQueryReq结构体 参考 ...................................	106
	5.21.1	详细描述	106
5.22	XTPFu	ndQueryRsp结构体 参考	107
	5.22.1	详细描述	107
5.23	XTPFu	ndTransferNotice结构体参考....................................	107

VIII CONTENTS

	5.23.1 详细描述	108
5.24	XTPFundTransferReq结构体 参考	108
	5.24.1 详细描述	108
5.25	XTPMarketDataBondExData结构体 参考	108
	5.25.1 详细描述	110
5.26	XTPMarketDataOptionExData结构体 参考	110
	5.26.1 详细描述	110
5.27	XTPMarketDataStockExData结构体 参考	110
	5.27.1 详细描述	112
5.28	XTPMarketDataStruct结构体 参考	112
	5.28.1 详细描述	114
5.29	XTPOptCombLegInfo结构体 参考	114
	5.29.1 详细描述	114
5.30	XTPOptCombOrderInfo结构体 参考	114
	5.30.1 详细描述	115
	5.30.2 结构体成员变量说明	115
	5.30.2.1 market	116
5.31	XTPOptCombOrderInfoEx结构体 参考	116
	5.31.1 详细描述	117
	5.31.2 结构体成员变量说明	117
	5.31.2.1 market	117
5.32	XTPOptCombOrderInsertInfo结构体 参考	117
	5.32.1 详细描述	118
5.33	XTPOptCombPlugin结构体 参考	118
	5.33.1 详细描述	118
5.34	XTPOptCombTradeReport结构体 参考	118
	5.34.1 详细描述	119
5.35	XTPOrderCancelInfo结构体 参考	119
	5.35.1 详细描述	120
5.36	XTPOrderInfo结构体 参考	120

CONTENTS

	5.36.1 详细描述	121
5.37	XTPOrderInfoEx结构体 参考	121
	5.37.1 详细描述	123
5.38	XTPOrderInsertInfo结构体 参考	123
	5.38.1 详细描述	124
5.39	XTPQueryAssetRsp结构体 参考	124
	5.39.1 详细描述	125
5.40	XTPQueryCombineStrategyInfoRsp结构体 参考	126
	5.40.1 详细描述	126
5.41	XTPQueryETFBaseReq结构体 参考	126
	5.41.1 详细描述	127
5.42	XTPQueryETFBaseRsp结构体 参考	127
	5.42.1 详细描述	127
5.43	XTPQueryETFComponentReq结构体 参考	128
	5.43.1 详细描述	128
5.44	XTPQueryETFComponentRsp结构体 参考	128
	5.44.1 详细描述	129
5.45	XTPQueryETFComponentRspV1结构体 参考	129
	5.45.1 详细描述	129
5.46	XTPQueryFundTransferLogReq结构体 参考	129
	5.46.1 详细描述	130
5.47	XTPQueryIPOQuotaRsp结构体 参考	130
	5.47.1 详细描述	130
5.48	XTPQueryIPOQuotaRspV1结构体 参考	130
	5.48.1 详细描述	131
5.49	XTPQueryIPOTickerRsp结构体 参考	131
	5.49.1 详细描述	131
5.50	XTPQueryOptCombExecPosReq结构体 参考	132
	5.50.1 详细描述	132
5.51	XTPQueryOptCombExecPosRsp结构体 参考	132

X CONTENTS

	5.51.1 详细描述	133
5.52	XTPQueryOptCombOrderByPageReq结构体 参考	133
	5.52.1 详细描述	134
5.53	XTPQueryOptCombOrderReq结构体 参考	134
	5.53.1 详细描述	
5.54	XTPQueryOptCombPositionReg结构体 参考	
	5.54.1 详细描述	135
5.55	XTPQueryOptCombPositionRsp结构体 参考	135
	5.55.1 详细描述	135
5.56	XTPQueryOptCombReportByExecIdReg结构体 参考	135
	5.56.1 详细描述	136
5.57	XTPQueryOptCombTraderByPageReg结构体 参考	136
	5.57.1 详细描述	136
5.58	XTPQueryOptCombTraderReq结构体 参考	136
0.00	5.58.1 详细描述	137
5 59	XTPQueryOptExecInfoRsp结构体 参考	137
0.00	5.59.1 详细描述	138
5.60	XTPQueryOptionAuctionInfoReg结构体参考	138
3.00	5.60.1 详细描述	
5.61	XTPQueryOptionAuctionInfoRsp结构体参考	
5.01	5.61.1 详细描述	
E 60	XTPQueryOrderByPageReq结构体 参考	
3.02	5.62.1 详细描述	
F 00		
5.63	XTPQueryOrderReq结构体 参考	
	5.63.1 详细描述	
5.64	XTPQueryReportByExecIdReq结构体 参考	
	5.64.1 详细描述	
5.65	XTPQueryStkPositionReq结构体 参考	
	5.65.1 详细描述	
5.66	XTPQueryStkPositionRsp结构体 参考	142

CONTENTS xi

	5.66.1 详细描述	143
5.67	XTPQueryStructuredFundInfoReq结构体 参考	144
	5.67.1 详细描述	144
5.68	XTPQueryTraderByPageReq结构体 参考	144
	5.68.1 详细描述	144
5.69	XTPQueryTraderReq结构体 参考	145
	5.69.1 详细描述	145
5.70	XTPQuoteFullInfo结构体 参考	145
	5.70.1 详细描述	146
5.71	XTPQuoteRebuildReq结构体 参考	147
	5.71.1 详细描述	147
5.72	XTPQuoteRebuildResultRsp结构体 参考	147
	5.72.1 详细描述	148
5.73	XTPQuoteStaticInfo结构体 参考	148
	5.73.1 详细描述	149
5.74	XTPRspInfoStruct结构体 参考	149
	5.74.1 详细描述	149
5.75	XTPSpecificTickerStruct结构体 参考	149
	5.75.1 详细描述	150
5.76	XTPStrategyInfoStruct结构体 参考	150
	5.76.1 详细描述	150
5.77	XTPStrategyStateReportStruct结构体 参考	150
	5.77.1 详细描述	151
5.78	XTPStrategySymbolInfoStruct结构体 参考	151
	5.78.1 详细描述	152
5.79	XTPStrategySymbolReqStruct结构体 参考	152
	5.79.1 详细描述	152
5.80	XTPStrategySymbolStateReportStruct结构体 参考	152
	5.80.1 详细描述	154
5.81	XTPStructuredFundInfo结构体 参考	154

xii CONTENTS

5.81.1 详细描述	154
5.82 XTPTickByTickEntrust结构体 参考	155
5.82.1 详细描述	155
5.82.2 结构体成员变量说明	155
5.82.2.1 ord_type	155
5.82.2.2 order_no	155
5.82.2.3 qty	155
5.82.2.4 seq	156
5.82.2.5 side	156
5.83 XTPTickByTickStatus结构体 参考	156
5.83.1 详细描述	156
5.84 XTPTickByTickStruct结构体 参考	156
5.84.1 详细描述	157
5.84.2 结构体成员变量说明	157
5.84.2.1 seq	157
5.85 XTPTickByTickTrade结构体参考	157
5.85.1 详细描述	158
5.85.2 结构体成员变量说明	158
5.85.2.1 seq	158
5.85.2.2 trade_flag	158
5.86 XTPTickerPriceInfo结构体 参考	158
5.86.1 详细描述	159
5.87 XTPTradeReport结构体 参考	159
5.87.1 详细描述	160
5.88 XTPUserTerminalInfoReq结构体参考	160
5.88.1 详细描述	161

CONTENTS xiii

6	文件	说明	163
	6.1	algo_api_struct.h 文件参考	163
		6.1.1 详细描述	164
	6.2	algo_data_type.h 文件参考	164
		6.2.1 详细描述	164
	6.3	demo_test_trade_api.cpp 文件参考	165
		6.3.1 详细描述	165
		6.3.2 函数说明	165
		6.3.2.1 main()	165
	6.4	demo_test_trade_spi.cpp 文件参考	167
		6.4.1 详细描述	167
	6.5	demo_test_trade_spi.h 文件参考	167
		6.5.1 详细描述	167
	6.6	xoms_api_fund_struct.h 文件参考	168
		6.6.1 详细描述	168
	6.7	xoms_api_struct.h 文件参考	168
		6.7.1 详细描述	172
	6.8	xquote_api_rebuild_tbt_struct.h 文件参考	172
		6.8.1 详细描述	173
		6.8.2 类型定义说明	173
		6.8.2.1 XTPQuoteRebuildReq	173
	6.9	xquote_api_struct.h 文件参考	173
		6.9.1 详细描述	175
	6.10	xtp_api_data_type.h 文件参考	175
		6.10.1 详细描述	182
		6.10.2 宏定义说明	182
		6.10.2.1 XTP_SIDE_STOCK_REPAY_STOCK	182
		6.10.3 枚举类型说明	183
		6.10.3.1 ETF_REPLACE_TYPE	183
		6.10.3.2 XTP_ACCOUNT_TYPE	183

xiv CONTENTS

	6.10.3.3 XTP_AUTO_SPLIT_TYPE	183
	6.10.3.4 XTP_BUSINESS_TYPE	184
	6.10.3.5 XTP_CRD_CR_STATUS	184
	6.10.3.6 XTP_DEBT_EXTEND_OPER_STATUS	186
	6.10.3.7 XTP_EXCHANGE_TYPE	186
	6.10.3.8 XTP_EXPIRE_DATE_TYPE	186
	6.10.3.9 XTP_FUND_OPER_STATUS	187
	6.10.3.10 XTP_FUND_QUERY_TYPE	187
	6.10.3.11 XTP_FUND_TRANSFER_TYPE	187
	6.10.3.12 XTP_LOG_LEVEL	188
	6.10.3.13 XTP_MARKET_TYPE	188
	6.10.3.14 XTP_OPT_CALL_OR_PUT_TYPE	189
	6.10.3.15 XTP_OPT_COVERED_OR_UNCOVERED	189
	6.10.3.16 XTP_OPT_EXERCISE_TYPE_TYPE	189
	6.10.3.17 XTP_OPT_POSITION_TYPE	189
	6.10.3.18 XTP_ORDER_ACTION_STATUS_TYPE	190
	6.10.3.19 XTP_ORDER_DETAIL_TYPE	190
	6.10.3.20 XTP_ORDER_STATUS_TYPE	190
	6.10.3.21 XTP_ORDER_SUBMIT_STATUS_TYPE	191
	6.10.3.22 XTP_POSITION_DIRECTION_TYPE	191
	6.10.3.23 XTP_POSITION_SECURITY_TYPE	192
	6.10.3.24 XTP_PRICE_TYPE	192
	6.10.3.25 XTP_PROTOCOL_TYPE	192
	6.10.3.26 XTP_QUALIFICATION_TYPE	193
	6.10.3.27 XTP_QUOTE_REBUILD_DATA_TYPE	193
	6.10.3.28 XTP_REBUILD_RET_TYPE	193
	6.10.3.29 XTP_SECURITY_STATUS	194
	6.10.3.30 XTP_SECURITY_TYPE	194
	6.10.3.31 XTP_SPLIT_MERGE_STATUS	195
	6.10.3.32 XTP_TBT_TYPE	195
	6.10.3.33 XTP_TE_RESUME_TYPE	195
	6.10.3.34 XTP_TICKER_TYPE	196
	6.10.3.35 XTP_UNDERLYING_TYPE	196
	6.10.3.36 XTPTerminalType	196
6.11	xtp_api_struct.h 文件参考	197
	6.11.1 详细描述	
6.12	xtp_api_struct_common.h 文件参考	
	6.12.1 详细描述	198
6.13	xtp_trader_api.h 文件参考	
	6.13.1 详细描述	198
索引		199

# XTP 极速交易系统 Trader API 2.2.33.5

本项目是XTP项目中的交易类+算法单接口

- (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
- (3) 程序化交易接口响应类Demo demo\_test\_trade\_spi.cpp

# 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:

OrderBookStruct	6
TraderApi	7
TraderSpi	80
DemoTestTraderSpi	1
KTPClientQueryCrdDebtStockReq	96
KTPClientQueryCrdPositionStkInfo	97
KTPClientQueryCrdPositionStockReq	8
KTPClientQueryCrdSurplusStkReqInfo	8
KTPClientQueryCrdSurplusStkRspInfo	99
KTPCombLegStrategy9	99
KTPCrdCashRepayDebtInterestFeeRsp	)0
KTPCrdCashRepayInfo	)0
KTPCrdCashRepayRsp	)1
KTPCrdDebtInfo	)2
KTPCrdDebtStockInfo	)3
KTPCrdFundExtraInfo	)3
KTPCrdFundInfo	)4
KTPCrdPositionExtraInfo	)4
KTPCreditDebtExtendNotice	)5
KTPCreditDebtExtendReq	)5
KTPFundQueryReq	)6
KTPFundQueryRsp	)7
KTPFundTransferNotice	)7
KTPFundTransferReq	8(
KTPMarketDataBondExData	8(
KTPMarketDataOptionExData	10
KTPMarketDataStockExData	10
KTPMarketDataStruct	12
KTPOptCombLegInfo	14
KTPOptCombOrderInfo	14
KTPOptCombOrderInfoEx	16
KTPOptCombOrderInsertInfo	17
KTPOptCombPlugin	18
KTPOptCombTradeReport	18
CTPOrderCancelInfo	19

XTPOrderInfo
XTPOrderInfoEx
XTPOrderInsertInfo
XTPQueryAssetRsp
XTPQueryCombineStrategyInfoRsp
XTPQueryETFBaseReq
XTPQueryETFBaseRsp
XTPQueryETFComponentReq
XTPQueryETFComponentRsp
XTPQueryETFComponentRspV1
XTPQueryFundTransferLogReq
XTPQueryIPOQuotaRsp
XTPQueryIPOQuotaRspV1
XTPQueryIPOTickerRsp
XTPQueryOptCombExecPosReq
XTPQueryOptCombExecPosRsp
XTPQueryOptCombOrderByPageReq
XTPQueryOptCombOrderReq
XTPQueryOptCombPositionReq
XTPQueryOptCombPositionRsp
XTPQueryOptCombReportByExecIdReq
XTPQueryOptCombTraderByPageReq
XTPQueryOptCombTraderReq
XTPQueryOptExecInfoRsp
XTPQueryOptionAuctionInfoReq
XTPQueryOptionAuctionInfoRsp
XTPQueryOrderByPageReq
XTPQueryOrderReq
XTPQueryReportByExecIdReq
XTPQueryStkPositionReq
XTPQueryStkPositionRsp
XTPQueryStructuredFundInfoReq
XTPQueryTraderByPageReq
XTPQueryTraderReq
XTPQuoteFullInfo
XTPQuoteRebuildReq
XTPQuoteRebuildResultRsp
XTPQuoteStaticInfo
XTPRspInfoStruct
XTPSpecificTickerStruct
XTPStrategyInfoStruct
XTPStrategyStateReportStruct
XTPStrategySymbolInfoStruct
XTPStrategySymbolReqStruct
XTPStrategySymbolStateReportStruct
XTPStructuredFundInfo
XTPTickByTickEntrust
XTPTickByTickStatus
XTPTickByTickStruct
XTPTickByTickTrade
XTPTickerPriceInfo
XTPTradeReport
XTPUserTerminalInfoReg
73.11 0001.101111111011111111111111111111

# 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:

DemoTestTraderSpi	
Demo自定义交易接口响应类	11
OrderBookStruct	
订单薄	16
TraderApi	
	17
TraderSpi	
	60
XTPClientQueryCrdDebtStockReq 融资融券指定证券上的负债未还数量请求结构体	96
XTPClientQueryCrdPositionStkInfo	
10.000	97
XTPClientQueryCrdPositionStockReq	
	98
XTPClientQueryCrdSurplusStkReqInfo	
	98
XTPClientQueryCrdSurplusStkRspInfo 信用业务余券信息	99
XTPCombLegStrategy	
期权组合策略的成分合约信息	99
XTPCrdCashRepayDebtInterestFeeRsp	
融资融券现金还息费响应信息	00
XTPCrdCashRepayInfo	
单条融资融券直接还款记录信息1	00
XTPCrdCashRepayRsp	
	01
XTPCrdDebtInfo	
	02
XTPCrdDebtStockInfo	
	03
XTPCrdFundExtraInfo	
	03
XTPCrdFundInfo	_
	04
XTPCrdPositionExtraInfo	n۷
MAR THEMRATERIA I STOT IN DRIVING ENGLISHED IN	114

6 结构体索引

XTPCreditDebtExtendNotice 用户展期请求的通知	105
XTPCreditDebtExtendReq 用户展期请求	105
XTPFundQueryReq         用户资金查询请求结构体	106
XTPFundQueryRsp         用户资金查询响应结构体	107
XTPFundTransferNotice 资金内转流水通知	107
XTPFundTransferReq 用户资金请求	108
XTPMarketDataBondExData 债券额外数据	108
XTPMarketDataOptionExData 期权额外数据	110
XTPMarketDataStockExData 股票、基金等额外数据	110
XTPMarketDataStruct	112
XTPOptCombLegInfo 组合策略腿合约信息结构体	114
XTPOptCombOrderInfo 期权组合策略报单响应结构体	114
XTPOptCombOrderInfoEx 期权组合策略报单响应结构体,新版本	116
XTPOptCombOrderInsertInfo 期权组合策略新订单请求	117
XTPOptCombPlugin 期权组合策略报单附加信息结构体	118
XTPOptCombTradeReport 期权组合策略报单成交结构体	118
XTPOrderCancelInfo 撤单失败响应消息	119
XTPOrderInfo 报单响应结构体	120
XTPOrderInfoEx 报单响应结构体,新版本	121
XTPOrderInsertInfo 新订单请求	123
XTPQueryAssetRsp 账户资金查询响应结构体	124
XTPQueryCombineStrategyInfoRsp 查询期权组合策略信息的响应	126
XTPQueryETFBaseReq	126
XTPQueryETFBaseRsp 查询股票ETF合约基本情况—响应结构体	127
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	128
XTPQueryETFComponentRsp         查询股票ETF成分股信息—响应结构体	128
XTPQueryETFComponentRspV1         查询股票ETF成分股信息—响应结构体,旧版本。	129
XTPQueryFundTransferLogReq 资金内转流水查询请求与响应	129
XTPQueryIPOQuotaRsp 查询用户申购额度-包含创业板额度	130

3.1 结构体 7

XTPQueryIPOQuotaRspV1	
查询用户申购额度-旧版	130
XTPQueryIPOTickerRsp查询当日可申购新股信息	131
XTPQueryOptCombExecPosReq         查询期权行权合并头寸请求结构体	132
XTPQueryOptCombExecPosRsp         查询期权行权合并头寸的响应	132
XTPQueryOptCombOrderByPageReq 查询期权组合策略订单请求-分页查询	133
XTPQueryOptCombOrderReq         期权组合策略报单查询 ////////////////////////////////////	134
XTPQueryOptCombPositionReq 查询期权组合策略持仓情况请求结构体	134
XTPQueryOptCombPositionRsp         查询期权组合策略持仓信息的响应	135
XTPQueryOptCombReportByExecIdReq         期权组合策略成交回报查询 ////////////////////////////////////	135
XTPQueryOptCombTraderByPageReq 查询期权组合策略成交回报请求-分页查询	136
XTPQueryOptCombTraderReq 查询期权组合策略成交回报请求-查询条件	136
XTPQueryOptExecInfoRsp 查询期权合约行权信息的响应	137
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	138
XTPQueryOptionAuctionInfoRsp         查询期权竞价交易业务参考信息	138
XTPQueryOrderByPageReq 查询订单请求-分页查询	140
XTPQueryOrderReq 报单查询 ////////////////////////////////////	141
XTPQueryReportByExecIdReq           成交回报查询 ////////////////////////////////////	141
XTPQueryStkPositionReq 查询股票持仓情况请求结构体	142
XTPQueryStkPositionRsp 查询股票持仓情况	142
XTPQueryStructuredFundInfoReq 查询分级基金信息结构体	144
XTPQueryTraderByPageReq 查询成交回报请求-分页查询	144
XTPQueryTraderReq 查询成交回报请求-查询条件	145
XTPQuoteFullInfo  股票行情全量静态信息	145
XTPQuoteRebuildReq 实时行情回补查询	147
XTPQuoteRebuildResultRsp 实时行情回补响应结构体	147
XTPQuoteStaticInfo 股票行情静态信息	148
XTPRspInfoStruct 响应信息	149

KTPSpecificTickerStruct	
· · · · · · · · · · · · · · · · · · ·	49
CTPStrategyInfoStruct	
策略信息结构体	50
(TPStrategyStateReportStruct	
	50
(TPStrategySymbolInfoStruct	
., .,	51
KTPStrategySymbolReqStruct	
18/9// 8/8/9 = 24 // 18 4 18/41	52
KTPStrategySymbolStateReportStruct	
	52
KTPStructuredFundInfo 查询分级基金信息响应结构体	- 4
	54
KTPTickByTickEntrust 逐笔委托 ....................................	55
少毛安化 · · · · · · · · · · · · · · · · · · ·	55
and the state of t	56
(TPTickByTickStruct	50
to the state of th	56
(TPTickByTickTrade	
	57
(TPTickerPriceInfo	
供查询的最新信息 1	58
KTPTradeReport CTPTradeReport CTPTra	
报 <sup>单</sup> 成交结构体	59
CTPUserTerminalInfoReq	
申报用户的ip和mac等信息、仅限授权用户使用	60

# 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:

algo_api_struct.h	
	163
algo_data_type.h	
定义业务公共数据结构	164
demo_test_trade_api.cpp	
定义控制台测试应用程序的入口点	165
demo_test_trade_spi.cpp	
Demo自定义客户端交易响应接口类	167
demo_test_trade_spi.h	
Demo自定义客户端交易响应接口类	167
xoms_api_fund_struct.h	
定义资金划拨相关结构体类型	168
xoms_api_struct.h	
定义交易类相关数据结构	168
xquote_api_rebuild_tbt_struct.h	
定义行情类相关数据结构	172
xquote_api_struct.h	
定义行情类相关数据结构	173
xtp_api_data_type.h	
定义兼容数据基本类型	175
xtp_api_struct.h	
定义业务数据结构	197
xtp_api_struct_common.h	
定义业务公共数据结构	197
xtp_trader_api.h	
定义客户端交易接口	198

文件索引

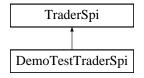
## 结构体说明

### 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 OnDisconnected (uint64\_t session\_id, int reason)
   当客户端个某个连接与交易后台通信连接断开
- virtual void OnAlgoDisconnected (int reason)
- virtual void OnAlgoConnected () 当客户端与*AlgoBus*断线后重新连接时,该方法被调用,仅在断线重连成功后会被调用。
- virtual void OnQueryStrategy (XTPStrategyInfoStruct \*strategy\_info, char \*strategy\_param, XTPRI \*error
   — info, int32\_t request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnStrategyStateReport (XTPStrategyStateReportStruct \*strategy\_state, uint64\_t session\_id)
- virtual void OnALGOUserEstablishChannel (char \*user, XTPRI \*error\_info, uint64\_t session\_id)
- virtual void OnInsertAlgoOrder (XTPStrategyInfoStruct \*strategy\_info, XTPRI \*error\_info, uint64\_t session 
   id)
- virtual void OnCancelAlgoOrder (XTPStrategyInfoStruct \*strategy\_info, XTPRI \*error\_info, uint64\_← t session\_id)

#### 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

12 结构体说明

#### 5.1.2 成员函数说明

#### 5.1.2.1 OnAlgoConnected()

```
void OnAlgoConnected ( ) [virtual]
```

当客户端与AlgoBus断线后重新连接时,该方法被调用,仅在断线重连成功后会被调用。

与算法服务器重新建立起连接,仅在断连重连成功后通知

重载 TraderSpi.

#### 5.1.2.2 OnAlgoDisconnected()

```
void OnAlgoDisconnected ( int \ reason \ ) \quad [virtual]
```

当客户端与AlgoBus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

请不要堵塞此线程,否则会影响algo的登录,与Algo之间的连接,断线后会自动重连,用户无需做其他操作

与算法服务器断线,此时api会自动与算法服务器重连,无需用户重连

重载 TraderSpi.

#### 5.1.2.3 OnALGOUserEstablishChannel()

algo业务中用户建立算法通道的消息响应

#### 参数

user	用户名
error_info	建立算法通道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,即算法通道成功
session← _id	资金账户对应的session_id,登录时得到

#### 备注

算法通道建立成功后,才能对用户创建策略等操作,一个用户只能拥有一个算法通道,如果之前已经建立,则无需重复建立

建立算法通道的异步通知

建立算法通道成功后, 可以下算法母单

读取算法单的参数

发送算法单

算法单发送成功

算法单发送失败

重载 TraderSpi.

#### 5.1.2.4 OnCancelAlgoOrder()

#### algo业务中撤销策略单的响应

#### 参数

strategy_info	用户撤销的策略单的具体信息
error_info	撤销策略单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
session_id	资金账户对应的session_id,登录时得到

#### 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

#### 算法单撤销结果通知

14 结构体说明

算法单撤销失败

算法单撤销成功

重载 TraderSpi.

#### 5.1.2.5 OnDisconnected()

```
void OnDisconnected (
          uint64_t session_id,
          int reason ) [virtual]
```

当客户端个某个连接与交易后台通信连接断开

用户与交易服务器断连,需要用户重新建立连接,并更新session id

重载 TraderSpi.

#### 5.1.2.6 OnInsertAlgoOrder()

algo业务中报送策略单的响应

#### 参数

strategy_info	用户报送的策略单的具体信息
error_info	报送策略单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
session_id	资金账户对应的session_id,登录时得到

#### 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

发送算法单后的异步通知

算法单建立成功

需要撤销算法单的时候, 可以发送撤销算法单消息

撤销算法单发送成功

撤销算法单发送失败

重载 TraderSpi.

#### 5.1.2.7 OnQueryStrategy()

#### algo业务中查询策略列表的响应

#### 参数

strategy_info	策略具体信息
strategy_param	此策略中包含的参数,如果error_info.error_id为0时,有意义
error_info	查询查询策略列表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
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,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
session_id	资金账户对应的session_id,登录时得到

#### 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

算法单查询结果通知

重载 TraderSpi.

#### 5.1.2.8 OnStrategyStateReport()

algo业务中策略运行时策略状态通知

#### 参数

strategy_state	用户策略运行情况的状态通	i知
session_id	资金账户对应的session_id,	登录时得到

#### 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

16 结构体说明

#### 重载 TraderSpi.

该类的文档由以下文件生成:

- demo\_test\_trade\_spi.h
- demo\_test\_trade\_spi.cpp

### 5.2 OrderBookStruct结构体参考

#### 订单薄

#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10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

#### 订单薄

该结构体的文档由以下文件生成:

xquote\_api\_struct.h

5.3 TraderApi类 参考 17

### 5.3 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 XTPRI \* GetApiLastError ()=0
- virtual const char \* GetApiVersion ()=0
- virtual uint8 t GetClientIDByXTPID (uint64 t order xtp id)=0
- virtual const char \* GetAccountByXTPID (uint64\_t order\_xtp\_id)=0
- virtual void SubscribePublicTopic (XTP\_TE\_RESUME\_TYPE resume\_type)=0
- virtual void SetSoftwareVersion (const char \*version)=0
- virtual void SetSoftwareKey (const char \*key)=0
- virtual void SetHeartBeatInterval (uint32 t interval)=0
- virtual uint64\_t Login (const char \*ip, int port, const char \*user, const char \*password, XTP\_PROTOCOL\_TYPE sock\_type, const char \*local\_ip=NULL)=0
- virtual int Logout (uint64\_t session\_id)=0
- virtual bool IsServerRestart (uint64 t session id)=0
- virtual int ModifyUserTerminalInfo (const XTPUserTerminalInfoReq \*info, uint64 t session id)=0
- virtual int QueryAccountTradeMarket (uint64 t session id, int request id)=0
- virtual uint64\_t GetANewOrderXTPID (uint64\_t session\_id)=0
- virtual uint64 t InsertOrder (XTPOrderInsertInfo \*order, uint64 t session id)=0
- virtual uint64 t InsertOrderExtra (XTPOrderInsertInfo \*order, uint64 t session id)=0
- virtual uint64\_t CancelOrder (const uint64\_t order\_xtp\_id, uint64\_t session\_id)=0
- virtual int QueryOrderByXTPID (const uint64 t order xtp id, uint64 t session id, int request id)=0
- virtual int QueryOrders (const XTPQueryOrderReg \*query param, uint64 t session id, int request id)=0
- virtual int QueryUnfinishedOrders (uint64\_t session\_id, int request\_id)=0
- virtual int QueryOrdersByPage (const XTPQueryOrderByPageReq \*query\_param, uint64\_t session\_id, int request\_id)=0
- virtual int QueryOrderByXTPIDEx (const uint64\_t order\_xtp\_id, uint64\_t session\_id, int request\_id)=0
- virtual int QueryOrdersEx (const XTPQueryOrderReq \*query\_param, uint64\_t session\_id, int request\_id)=0
- virtual int QueryUnfinishedOrdersEx (uint64 t session id, int request id)=0
- virtual int QueryOrdersByPageEx (const XTPQueryOrderByPageReq \*query\_param, uint64\_t session\_id, int request id)=0
- virtual int QueryTradesByXTPID (const uint64\_t order\_xtp\_id, uint64\_t session\_id, int request\_id)=0
- virtual int QueryTrades (XTPQueryTraderReq \*query\_param, uint64\_t session\_id, int request\_id)=0
- virtual int QueryTradesByPage (const XTPQueryTraderByPageReq \*query\_param, uint64\_t session\_id, int request\_id)=0
- virtual int QueryPosition (const char \*ticker, uint64\_t session\_id, int request\_id, XTP\_MARKET\_TYPE market=XTP MKT INIT)=0
- virtual int QueryAsset (uint64\_t session\_id, int request\_id)=0
- virtual int QueryStructuredFund (XTPQueryStructuredFundInfoReq \*query\_param, uint64\_t session\_id, int request\_id)=0
- virtual uint64 t FundTransfer (XTPFundTransferReg \*fund transfer, uint64 t session id)=0
- virtual int QueryFundTransfer (XTPQueryFundTransferLogReq \*query\_param, uint64\_t session\_id, int request id)=0
- virtual int QueryOtherServerFund (XTPFundQueryReq \*query\_param, uint64\_t session\_id, int request\_id)=0
- virtual int QueryETF (XTPQueryETFBaseReq \*query\_param, uint64\_t session\_id, int request\_id)=0

18 结构体说明

 virtual int QueryETFTickerBasket (XTPQueryETFComponentReq \*query\_param, uint64\_t session\_id, int request\_id)=0

- virtual int QueryIPOInfoList (uint64\_t session\_id, int request\_id)=0
- virtual int QueryIPOQuotaInfo (uint64\_t session\_id, int request\_id)=0
- virtual int QueryOptionAuctionInfo (XTPQueryOptionAuctionInfoReq \*query\_param, uint64\_t session\_id, int request\_id)=0
- virtual uint64\_t CreditCashRepay (double amount, uint64\_t session\_id)=0
- virtual uint64\_t CreditCashRepayDebtInterestFee (const char \*debt\_id, double amount, uint64\_t session\_
   id)=0
- virtual uint64\_t CreditSellStockRepayDebtInterestFee (XTPOrderInsertInfo \*order, const char \*debt\_id, uint64\_t session\_id)=0
- virtual int QueryCreditCashRepayInfo (uint64\_t session\_id, int request\_id)=0
- virtual int QueryCreditFundInfo (uint64\_t session\_id, int request\_id)=0
- virtual int QueryCreditDebtInfo (uint64\_t session\_id, int request\_id)=0
- virtual int QueryCreditTickerDebtInfo (XTPClientQueryCrdDebtStockReq \*query\_param, uint64\_t session ← id, int request id)=0
- virtual int QueryCreditAssetDebtInfo (uint64 t session id, int request id)=0
- virtual int QueryCreditTickerAssignInfo (XTPClientQueryCrdPositionStockReq \*query\_param, uint64\_←
   t session\_id, int request\_id)=0
- virtual int QueryCreditExcessStock (XTPClientQueryCrdSurplusStkReqInfo \*query\_param, uint64\_←
  t session id, int request id)=0
- virtual int QueryMulCreditExcessStock (XTPClientQueryCrdSurplusStkReqInfo \*query\_param, uint64\_←
  t session id, int request id)=0
- virtual uint64 t CreditExtendDebtDate (XTPCreditDebtExtendReg \*debt extend, uint64 t session id)=0
- virtual int QueryCreditExtendDebtDateOrders (uint64\_t xtp\_id, uint64\_t session\_id, int request\_id)=0
- virtual int QueryCreditFundExtraInfo (uint64 t session id, int request id)=0
- virtual int QueryCreditPositionExtraInfo (XTPClientQueryCrdPositionStockReq \*query\_param, uint64\_← t session\_id, int request\_id)=0
- virtual uint64 t InsertOptionCombinedOrder (XTPOptCombOrderInsertInfo \*order, uint64 t session id)=0
- virtual uint64\_t InsertOptionCombinedOrderExtra (XTPOptCombOrderInsertInfo \*order, uint64\_t session\_← id)=0
- virtual uint64 t CancelOptionCombinedOrder (const uint64 t order xtp id, uint64 t session id)=0
- virtual int QueryOptionCombinedUnfinishedOrders (uint64 t session id, int request id)=0
- virtual int QueryOptionCombinedOrderByXTPID (const uint64\_t order\_xtp\_id, uint64\_t session\_id, int request\_id)=0
- virtual int QueryOptionCombinedOrders (const XTPQueryOptCombOrderReq \*query\_param, uint64\_← t session\_id, int request\_id)=0
- virtual int QueryOptionCombinedOrdersByPage (const XTPQueryOptCombOrderByPageReq \*query\_← param, uint64\_t session\_id, int request\_id)=0
- virtual int QueryOptionCombinedUnfinishedOrdersEx (uint64 t session id, int request id)=0
- virtual int QueryOptionCombinedOrderByXTPIDEx (const uint64\_t order\_xtp\_id, uint64\_t session\_id, int request\_id)=0
- virtual int QueryOptionCombinedOrdersEx (const XTPQueryOptCombOrderReq \*query\_param, uint64\_←
   t session\_id, int request\_id)=0
- virtual int QueryOptionCombinedOrdersByPageEx (const XTPQueryOptCombOrderByPageReq \*query\_
   param, uint64\_t session\_id, int request\_id)=0
- virtual int QueryOptionCombinedTradesByXTPID (const uint64\_t order\_xtp\_id, uint64\_t session\_id, int request\_id)=0
- virtual int QueryOptionCombinedTrades (const XTPQueryOptCombTraderReq \*query\_param, uint64\_←
   t session\_id, int request\_id)=0
- virtual int QueryOptionCombinedTradesByPage (const XTPQueryOptCombTraderByPageReq \*query\_← param, uint64 t session id, int request id)=0
- virtual int QueryOptionCombinedPosition (const XTPQueryOptCombPositionReq \*query\_param, uint64\_←
  t session\_id, int request\_id)=0
- virtual int QueryOptionCombinedStrategyInfo (uint64\_t session\_id, int request\_id)=0

5.3 TraderApi类 参考 19

- virtual int QueryOptionCombinedExecPosition (const XTPQueryOptCombExecPosReq \*query\_param, uint64\_t session\_id, int request\_id)=0
- virtual int LoginALGO (const char \*ip, int port, const char \*user, const char \*password, XTP\_PROTOCOL\_TYPE sock\_type, const char \*local\_ip=NULL)=0
- virtual int QueryStrategy (uint32\_t strategy\_type, uint64\_t client\_strategy\_id, uint64\_t xtp\_strategy\_id, uint64\_t session\_id, int32\_t request\_id)=0
- virtual int ALGOUserEstablishChannel (const char \*oms\_ip, int oms\_port, const char \*user, const char \*password, uint64\_t session\_id)=0
- virtual int InsertAlgoOrder (uint32\_t strategy\_type, uint64\_t client\_strategy\_id, char \*strategy\_param, uint64\_t session\_id)=0
- virtual int CancelAlgoOrder (bool cancel\_flag, uint64\_t xtp\_strategy\_id, uint64\_t session\_id)=0
- virtual uint64\_t GetAlgorithmIDByOrder (uint64\_t order\_xtp\_id, uint32\_t order\_client\_id)=0

#### 静态 Public 成员函数

#### 5.3.1 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

#### 5.3.2 成员函数说明

#### 5.3.2.1 ALGOUserEstablishChannel()

用户请求使用algo服务器建立算法通道

返回

表明此资金账号建立算法通道请求消息发送是否成功,非"0"表示发送失败,可以调用GetApiLast←Error()来获取错误代码,"0"表示发送成功

20 结构体说明

#### 参数

oms_ip	oms服务器地址,类似"127.0.0.1",非algo服务器地址
oms_port	oms服务器端口号,非algo服务器端口号
user	登录用户名
password	登录密码
session← _id	资金账户对应的session_id,登录时得到

#### 备注

此函数为异步方式,一个用户只能拥有一个算法通道,如果之前已经建立,则无需重复建立,在使用算法前,请先建立算法通道

#### 5.3.2.2 CancelAlgoOrder()

```
virtual int CancelAlgoOrder (
                bool cancel_flag,
                 uint64_t xtp_strategy_id,
                      uint64_t session_id ) [pure virtual]
```

#### algo业务中用户撤销算法单请求

#### 返回

请求发送是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

#### 参数

cancel_flag	是否需要算法去处理已下的算法子单标志,true-交给算法自行处理,包括撤单、平仓等,算法处理完成后会通知客户;false-立即停止算法母单的执行,此时算法平台会对已下的子单做撤单操作,其余的平仓等操作需要客户自己处理
xtp_strategy↔ _id	需要撤销的算法单在xtp algobus系统中的id
session_id	资金账户对应的session_id,登录时得到

#### 备注

仅能在用户建立算法通道后调用

#### 5.3.2.3 CancelOptionCombinedOrder()

## 期权组合策略报单撤单请求

## 返回

撤单在XTP系统中的ID,如果为'0'表示撤单发送失败,此时用户可以调用GetApiLastError()来获取错误代码,非"0"表示撤单发送成功,用户需要记录下返回的order\_cancel\_xtp\_id,它保证一个交易日内唯一,不同的交易日不保证唯一性

### 参数

order_xtp↔ _id	需要撤销的期权组合策略委托单在XTP系统中的ID
session_id	资金账户对应的session_id,登录时得到

#### 备注

如果撤单成功,会在报单响应函数OnOptionCombinedOrderEvent()里返回原单部撤或者全撤的消息,如果不成功,会在OnCancelOrderError()响应函数中返回错误原因

#### 5.3.2.4 CancelOrder()

### 报单操作请求

### 返回

撤单在XTP系统中的ID,如果为'0'表示撤单发送失败,此时用户可以调用GetApiLastError()来获取错误代码,非"0"表示撤单发送成功,用户需要记录下返回的order\_cancel\_xtp\_id,它保证一个交易日内唯一,不同的交易日不保证唯一性

order_xtp↔ _id	需要撤销的委托单在XTP系统中的ID
session_id	资金账户对应的session_id,登录时得到

#### 备注

如果撤单成功,会在报单响应函数OnOrderEvent()里返回原单部撤或者全撤的消息,如果不成功,会在OnCancelOrderError()响应函数中返回错误原因

#### 5.3.2.5 CreateTraderApi()

# 创建TraderApi

### 参数

client_id	(必须输入)客户端id,用于区分同一用户的不同客户端,由用户自定义,普通用户 必须使用1-99之间的数值	
save_file_path	(必须输入)存贮订阅信息文件的目录,请设定一个真实存在的有可写权限的路径	
log_level	日志输出级别	

### 返回

创建出的UserApi

## 备注

只能创建一次,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。系统不支持过夜,请确保每天开盘前重新启动

# 5.3.2.6 CreditCashRepay()

## 融资融券业务中现金直接还款请求

#### 返回

现金直接还款订单在XTP系统中的ID,如果为'0'表示消息发送失败,此时用户可以调用GetApiLast←Error()来获取错误代码,非"0"表示消息发送成功,用户需要记录下返回的xtp\_id,它保证一个交易日内唯一,不同的交易日不保证唯一性

### 参数

amount	现金还款的金额
session←	资金账户对应的session_id,登录时得到
_id	

### 5.3.2.7 CreditCashRepayDebtInterestFee()

融资融券业务中现金还指定负债合约息费请求

#### 返回

现金还息订单在XTP系统中的ID,如果为'0'表示消息发送失败,此时用户可以调用GetApiLastError()来获取错误代码,非"0"表示消息发送成功,用户需要记录下返回的xtp\_id,它保证一个交易日内唯一,不同的交易日不保证唯一性

### 参数

debt_id	指定的负债合约编号
amount	现金还息的金额
session⊷	资金账户对应的session_id,登录时得到
_id	

#### 5.3.2.8 CreditExtendDebtDate()

融资融券业务中请求负债合约展期

## 返回

负债合约展期订单在XTP系统中的ID,如果为'0'表示消息发送失败,此时用户可以调用GetApiLast←Error()来获取错误代码,非"0"表示消息发送成功,用户需要记录下返回的xtp\_id,它保证一个交易日内唯一,不同的交易日不保证唯一性

#### 参数

debt_extend	d 负债合约展期的请求信息	
session_id	资金账户对应的session_id,登录时得到	

### 5.3.2.9 CreditSellStockRepayDebtInterestFee()

融资融券业务中卖券还指定负债合约息费请求

### 返回

卖券还息订单在XTP系统中的ID,如果为'0'表示消息发送失败,此时用户可以调用GetApiLastError()来获取错误代码,非"0"表示消息发送成功,用户需要记录下返回的xtp\_id,它保证一个交易日内唯一,不同的交易日不保证唯一性

## 参数

order	卖券的报单录入信息,其中order.order_client_id字段是用户自定义字段,用户输入什么值,订单响应OnOrderEvent()返回时就会带回什么值,类似于备注,方便用户自己定位订单。当然,如果你什么都不填,也是可以的。order.order_xtp_id字段无需用户填写,order.ticker必须不带空格,以'\0'结尾	
debt_id	指定的负债合约编号	
session← _id	资金账户对应的session_id,登录时得到	

#### 5.3.2.10 FundTransfer()

## 资金划拨请求

# 返回

资金划拨订单在XTP系统中的ID,如果为'0'表示消息发送失败,此时用户可以调用GetApiLastError()来获取错误代码,非"0"表示消息发送成功,用户需要记录下返回的serial\_id,它保证一个交易日内唯一,不同的交易日不保证唯一性

### 参数

fund_transfer	资金划拨的请求信息
session_id	资金账户对应的session_id,登录时得到

# 备注

此函数支持一号两中心节点之间的资金划拨,注意资金划拨的方向。

### 5.3.2.11 GetAccountByXTPID()

通过报单在xtp系统中的ID获取相关资金账户名

返回

返回资金账户名

#### 参数

order_xtp↔	报单在xtp系统中的ID
_id	

# 备注

只有资金账户登录成功后,才能得到正确的信息

## 5.3.2.12 GetAlgorithmIDByOrder()

## 获取算法单的母单ID

返回

返回算法单的母单ID,如果返回为0表示不是算法单

### 参数

order_xtp_id	算法单对应的xtp id
order_client← id	算法单对应的自定义ID,不可随意填写

#### 备注

返回为0表示,不是算法单,如果传入的参数不对的话,可能会得不到正确结果,此函数调用不依赖于是否登录

#### 5.3.2.13 GetANewOrderXTPID()

为用户获取一个新的订单XTPID, 用于报单

### 返回

生成的订单XTPID,非"0"表示获取成功,"0"表示获取失败,此时用户可以调用GetApiLastError()来获取错误代码

### 参数

session⊷	资金账户对应的session_id,登录时得到
_id	

# 备注

此函数必须在Login之后调用,通过这个函数获取的order\_xtp\_id仅用于对应的用户报单,如果设置错误,将会导致下单失败

## 5.3.2.14 GetApiLastError()

```
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、InsertOrder、CancelOrder返回值为0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

### 5.3.2.15 GetApiVersion()

```
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

## 5.3.2.16 GetClientIDByXTPID()

通过报单在xtp系统中的ID获取下单的客户端id

返回

返回客户端id,可以用此方法过滤自己下的订单

## 参数

order_xtp⇔	报单在xtp系统中的ID
_id	

备注

由于系统允许同一用户在不同客户端上登录操作,每个客户端通过不同的client\_id进行区分

# 5.3.2.17 GetTradingDay()

```
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

#### 5.3.2.18 InsertAlgoOrder()

```
virtual int InsertAlgoOrder (
            uint32_t strategy_type,
            uint64_t client_strategy_id,
             char * strategy_param,
            uint64_t session_id ) [pure virtual]
```

# algo业务中用户报算法单请求

#### 返回

算法报单请求发送是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获 取错误代码

#### 参数

strategy_type	需要创建的策略类型
client_strategy↔ _id	用户自定义id,帮助用户定位
strategy_param	策略参数
session_id	资金账户对应的session_id,登录时得到

## 备注

仅能在用户建立算法通道后使用, 算法单的异步通知得

# 5.3.2.19 InsertOptionCombinedOrder()

```
virtual uint64_t InsertOptionCombinedOrder (
             XTPOptCombOrderInsertInfo * order,
            uint64_t session_id ) [pure virtual]
```

# 期权组合策略报单录入请求

# 返回

报单在XTP系统中的ID,如果为'0'表示报单发送失败,此时用户可以调用GetApiLastError()来获取错误 代码,非"0"表示报单发送成功,用户需要记录下返回的order\_xtp\_id,它保证一个交易日内唯一,不 同的交易日不保证唯一性

order	报单录入信息,其中order.order_client_id字段是用户自定义字段,用户输入什么值,订单响应OnOptionCombinedOrderEvent()返回时就会带回什么值,类似于备注,方便用户自己定位订单。当然,如果你什么都不填,也是可以的。order.order_xtp_id字段无需用户填写,order.ticker必须不带空格,以\0'结尾
session← id	资金账户对应的session_id,登录时得到

#### 备注

交易所接收订单后,会在报单响应函数OnOptionCombinedOrderEvent()中返回报单未成交的状态,之后所有的订单状态改变(除了部成状态)都会通过报单响应函数返回

#### 5.3.2.20 InsertOptionCombinedOrderExtra()

已经提前设置order xtp id的期权组合策略报单录入请求,与GetANewOrderXTPID()配合使用

### 返回

报单在XTP系统中的ID,如果为'0'表示报单发送失败,此时用户可以调用GetApiLastError()来获取错误代码,非"0"表示报单发送成功,此时等同与传入的order xtp id

#### 参数

order	报单录入信息,其中order.order_client_id字段是用户自定义字段,用户输入什么值,订单响应OnOrderEvent()返回时就会带回什么值,类似于备注,方便用户自己定位订单,也可以什么都不填。order.order_xtp_id字段必须是通过GetANewOrderXTPID()获得的值,order.ticker必须不带空格,以\\0'结尾
session← _id	资金账户对应的session_id,登录时得到

#### 备注

使用设置好的order\_xtp\_id(通过GetANewOrderXTPID()获得)进行报单,注意此处如果order\_xtp\_↔ id设置不对,将导致报单失败。交易所接收订单后,会在报单响应函数OnOptionCombinedOrder↔ Event()中返回报单未成交的状态,之后所有的订单状态改变(除了部成状态)都会通过报单响应函数返回

#### 5.3.2.21 InsertOrder()

#### 报单录入请求

#### 返回

报单在XTP系统中的ID,如果为'0'表示报单发送失败,此时用户可以调用GetApiLastError()来获取错误代码,非"0"表示报单发送成功,用户需要记录下返回的order\_xtp\_id,它保证一个交易日内唯一,不同的交易日不保证唯一性

#### 参数

order	报单录入信息,其中order.order_client_id字段是用户自定义字段,用户输入什么值,订单响应OnOrderEvent()返回时就会带回什么值,类似于备注,方便用户自己定位订单。当然,如果你什么都不填,也是可以的。order.order_xtp_id字段无需用户填写,order.ticker必须不带空格,以'\0'结尾
session← _id	资金账户对应的session_id,登录时得到

## 备注

交易所接收订单后,会在报单响应函数OnOrderEvent()中返回报单未成交的状态,之后所有的订单状态改变(除了部成状态)都会通过报单响应函数返回

### 5.3.2.22 InsertOrderExtra()

已经提前设置order xtp id的报单录入请求,与GetANewOrderXTPID()配合使用

## 返回

报单在XTP系统中的ID,如果为'0'表示报单发送失败,此时用户可以调用GetApiLastError()来获取错误代码,非"0"表示报单发送成功,此时等同与传入的order\_xtp\_id

### 参数

order	报单录入信息,其中order.order_client_id字段是用户自定义字段,用户输入什么值,订单响应OnOrderEvent()返回时就会带回什么值,类似于备注,方便用户自己定位订单,也可以什么都不填。order.order_xtp_id字段必须是通过GetANewOrderXTPID()获得的值,order.ticker必须不带空格,以\0'结尾
session← _id	资金账户对应的session_id,登录时得到

### 备注

使用设置好的order\_xtp\_id(通过GetANewOrderXTPID()获得)进行报单,注意此处如果order\_xtp\_→id设置不对,将导致报单失败。交易所接收订单后,会在报单响应函数OnOrderEvent()中返回报单未成交的状态,之后所有的订单状态改变(除了部成状态)都会通过报单响应函数返回

#### 5.3.2.23 IsServerRestart()

## 服务器是否重启过

### 返回

"true"表示重启过, "false"表示没有重启过

## 参数

session←	资金账户对应的session_id,登录时得到
_id	

## 备注

此函数必须在Login之后调用

#### 5.3.2.24 Login()

# 用户登录请求

## 返回

session\_id表明此资金账号登录是否成功,"0"表示登录失败,可以调用GetApiLastError()来获取错误代码,非"0"表示登录成功,此时需要记录下这个返回值session\_id,与登录的资金账户对应

## 参数

ip	服务器地址,类似"127.0.0.1"
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
local_ip	本地网卡地址,类似"127.0.0.1"

# 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可支持多个账户连接,但是同一个账户同一个client\_id只能有一个session连接,后面的登录在前一个session存续期间,无法连接

## 5.3.2.25 LoginALGO()

## 用户登录algo服务器请求

## 返回

表明此资金账号登录是否成功,非"0"表示登录失败,可以调用GetApiLastError()来获取错误代码, "0"表示登录成功

#### 参数

ip	algo服务器地址,类似"127.0.0.1"
port	algo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
local_ip	本地网卡地址,类似"127.0.0.1"

# 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只需调用一次,所有用户共用即可

#### 5.3.2.26 Logout()

# 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

### 参数

session←	资金账户对应的session_id,登录时得到
_id	

#### 5.3.2.27 ModifyUserTerminalInfo()

修改已登录用户的硬件信息, 仅限授权系统使用

返回

发送消息是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

## 参数

info	需要修改成的用户硬件信息
session⊷	资金账户对应的session_id,登录时得到
_id	

# 备注

此函数必须在Login之后调用,且仅限授权系统使用,一般客户无需使用

## 5.3.2.28 QueryAccountTradeMarket()

查询用户在本节点上的可交易市场类型

返回

### 参数

session← _id	资金账户对应的session_id,登录时得到
request⊷ _id	用于用户定位查询响应的ID,由用户自定义

### 备注

此函数必须在Login之后调用,对应的响应函数是OnQueryAccountTradeMarket()

## 5.3.2.29 QueryAsset()

### 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,此时用户可以调用GetApiLastError()来获取错误代码

## 参数

session←	资金账户对应的session_id,登录时得到
_id	
request←	用于用户定位查询响应的ID,由用户自定义
_id	

### 5.3.2.30 QueryCreditAssetDebtInfo()

### 请求查询信用账户待还资金信息

## 返回

### 参数

session→ 资金账户对应的session_id,登录时得到	
_id	
request⇔	用于用户定位查询响应的ID,由用户自定义
_id	

#### 5.3.2.31 QueryCreditCashRepayInfo()

请求查询融资融券业务中的现金直接还款报单

返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

#### 参数

session← _id	资金账户对应的session_id,登录时得到
request⊷ _id	用于用户定位查询响应的ID,由用户自定义

# 5.3.2.32 QueryCreditDebtInfo()

请求查询信用账户负债合约信息

返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

session← _id	资金账户对应的session_id,登录时得到
request <i>←</i> _id	用于用户定位查询响应的ID,由用户自定义

## 5.3.2.33 QueryCreditExcessStock()

### 融资融券业务中请求查询指定证券的余券

### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

## 参数

query_param	需要查询的余券信息,不可以为空,需要明确指定		
session_id	资金账户对应的session_id,登录时得到		
request_id	用于用户定位查询响应的ID,由用户自定义		

### 备注

该方法中用户必须提供了证券代码和所在市场

## 5.3.2.34 QueryCreditExtendDebtDateOrders()

# 融资融券业务中请求查询负债合约展期

# 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

xtp_id	需要查询的负债合约展期订单筛选条件,xtp_id可以为0,则默认所有负债合约展期订单,如果不为0,则请求特定的负债合约展期订单
session⊷	资金账户对应的session_id,登录时得到
_id	
request⊷	用于用户定位查询响应的ID,由用户自定义
_id	

## 5.3.2.35 QueryCreditFundExtraInfo()

请求查询融资融券业务中账的附加信息

返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

## 参数

session← _id	资金账户对应的session_id,登录时得到
request <i>←</i> _id	用于用户定位查询响应的ID,由用户自定义

# 5.3.2.36 QueryCreditFundInfo()

请求查询信用账户特有信息,除资金账户以外的信息

返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

## 参数

session⊷ id	资金账户对应的session_id,登录时得到
 request←	   用于用户定位查询响应的ID,由用户自定义
_id	

### 5.3.2.37 QueryCreditPositionExtraInfo()

```
\label{thm:continuous} \mbox{virtual int QueryCreditPositionExtraInfo (} $$ \mbox{XTPClientQueryCrdPositionStockReq} * query\_param, $$
```

```
uint64_t session_id,
int request_id ) [pure virtual]
```

请求查询融资融券业务中账指定证券的附加信息

# 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

### 参数

query_param	需要指定的证券,筛选条件中ticker可以全填0,如果不为0,请不带空格,并以'\0'结尾	
session_id	资金账户对应的session_id,登录时得到	
request_id	用于用户定位查询响应的ID,由用户自定义	

### 5.3.2.38 QueryCreditTickerAssignInfo()

请求查询信用账户可融券头寸信息

### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

# 参数

query_param	需要查询的证券,筛选条件中ticker可以全填0,如果不为0,请不带空格,并以'\0'结尾	
session_id	资金账户对应的session_id,登录时得到	
request_id	用于用户定位查询响应的ID,由用户自定义	

# 5.3.2.39 QueryCreditTickerDebtInfo()

请求查询指定证券负债未还信息

返回

### 参数

query_param	需要查询的指定证券,筛选条件中ticker可以全填0,如果不为0,请不带空格,并以'\0'结尾
session_id 资金账户对应的session_id,登录时得到	
request_id	用于用户定位查询响应的ID,由用户自定义

### 5.3.2.40 QueryETF()

## 请求查询ETF清单文件

# 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

## 参数

query_param	需要查询的ETF清单文件的筛选条件,其中合约代码可以为空,则默认所有存在的ETF合约代码,market字段也可以为初始值,则默认所有市场的ETF合约
session_id	资金账户对应的session_id,登录时得到
request_id	用于用户定位查询响应的ID,由用户自定义

## 5.3.2.41 QueryETFTickerBasket()

## 请求查询ETF股票篮

# 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

query_param	需要查询股票篮的的ETF合约,其中合约代码不可以为空,market字段也必须指定	
session_id	资金账户对应的session_id,登录时得到	
request_id	用于用户定位查询响应的ID,由用户自定义制作者	Doxygen

## 5.3.2.42 QueryFundTransfer()

### 请求查询资金划拨

#### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

## 参数

query_param	需要查询的资金划拨订单筛选条件,其中serial_id可以为0,则默认所有资金划拨订单,如果不为0,则请求特定的资金划拨订单
session_id	资金账户对应的session_id,登录时得到
request_id	用于用户定位查询响应的ID,由用户自定义

## 5.3.2.43 QuerylPOInfoList()

### 请求查询今日新股申购信息列表

# 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

## 参数

session← _id	资金账户对应的session_id,登录时得到
request← _id	用于用户定位查询响应的ID,由用户自定义

## 5.3.2.44 QuerylPOQuotaInfo()

```
virtual int QueryIPOQuotaInfo (
```

```
uint64_t session_id,
int request_id ) [pure virtual]
```

## 请求查询用户新股申购额度信息

# 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

### 参数

session← _id	资金账户对应的session_id,登录时得到
request← _id	用于用户定位查询响应的ID,由用户自定义

### 5.3.2.45 QueryMulCreditExcessStock()

#### 融资融券业务中请求查询余券

### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

# 参数

query_param	需要查询的余券信息。若填入市场和股票代码,返回单支股票信息;若市场代码为空,股票代码非空,是无效查询,会在SPI中返回错误;若市场和股票代码均为空,返回全市场信息;若市场代码非空,股票代码为空,返回单市场信息。
session_id	资金账户对应的session_id,登录时得到
request_id	用于用户定位查询响应的ID,由用户自定义

# 5.3.2.46 QueryOptionAuctionInfo()

## 请求查询期权合约

#### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

# 参数

query_param	需要查询的期权合约的筛选条件,可以为NULL(为NULL表示查询所有的期权合约)
session_id	资金账户对应的session_id,登录时得到
request_id	用于用户定位查询响应的ID,由用户自定义

#### 5.3.2.47 QueryOptionCombinedExecPosition()

请求查询期权行权合并头寸

### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

### 参数

query_param	需要查询的行权合并的筛选条件,其中market为0会默认查询全市场,成分合约代码可以初始化为空,如果不为空,请不带空格,并以'\0'结尾,注意所有填写的条件都会进行匹配
session_id	资金账户对应的session_id,登录时得到
request_id	用于用户定位查询响应的ID,由用户自定义

## 备注

该方法可能对应多条响应消息

### 5.3.2.48 QueryOptionCombinedOrderByXTPID()

根据报单ID请求查询期权组合策略报单-旧版本接口

### 返回

#### 参数

order_xtp↔ _id	需要查询的报单在xtp系统中的ID,即InsertOrder()成功时返回的order_xtp_id
session_id	资金账户对应的session_id,登录时得到
request_id	用于用户定位查询响应的ID,由用户自定义

## 5.3.2.49 QueryOptionCombinedOrderByXTPIDEx()

根据报单ID请求查询期权组合策略报单-新版本接口

### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

# 参数

order_xtp↔ _id	需要查询的报单在xtp系统中的ID,即InsertOrder()成功时返回的order_xtp_id
session_id	资金账户对应的session_id,登录时得到
request_id	用于用户定位查询响应的ID,由用户自定义

# 5.3.2.50 QueryOptionCombinedOrders()

请求查询期权组合策略报单-旧版本接口

### 返回

#### 参数

query_param	需要查询的订单相关筛选条件,其中合约代码可以为空,则默认所有存在的合约代码,如果不为空,请不带空格,并以\0'结尾,其中起始时间格式为YYYYMMDDHHMMSSsss,为0则默认当前交易日0点,结束时间格式为YYYYMMDDHHMMSSsss,为0则默认当前时间
session_id	资金账户对应的session_id,登录时得到
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,此函数查询出的结果可能对应多个查询结果响应。此函数不建议轮询使用,当报单量过多时,容易造成用户线路拥堵,导致api断线

#### 5.3.2.51 QueryOptionCombinedOrdersByPage()

分页请求查询期权组合策略报单-旧版本接口

返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

## 参数

query_param	需要分页查询订单的条件,如果第一次查询,那么query_param.reference填0
session_id	资金账户对应的session_id,登录时得到
request_id	用于用户定位查询响应的ID,由用户自定义

### 备注

该方法支持分页查询、注意用户需要记录下最后一笔查询结果的reference以便用户下次查询使用

## 5.3.2.52 QueryOptionCombinedOrdersByPageEx()

分页请求查询期权组合策略报单-新版本接口

#### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

## 参数

query_param	需要分页查询订单的条件,如果第一次查询,那么query_param.reference填0
session_id	资金账户对应的session_id,登录时得到
request_id	用于用户定位查询响应的ID,由用户自定义

## 备注

该方法支持分页查询,注意用户需要记录下最后一笔查询结果的reference以便用户下次查询使用

### 5.3.2.53 QueryOptionCombinedOrdersEx()

请求查询期权组合策略报单-新版本接口

# 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

### 参数

query_param	需要查询的订单相关筛选条件,其中合约代码可以为空,则默认所有存在的合约代码,如果不为空,请不带空格,并以'\0'结尾,其中起始时间格式为YYYYMMDDHHMMSSsss,为0则默认当前交易日0点,结束时间格式为YYYYMMDDHHMMSSsss,为0则默认当前时间
session_id	资金账户对应的session_id,登录时得到
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,此函数查询出的结果可能对应多个查询结果响应。此函数不建议轮询使用,当报单量过多时,容易造成用户线路拥堵,导致api断线

#### 5.3.2.54 QueryOptionCombinedPosition()

#### 请求查询投资者期权组合策略持仓

### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

# 参数

query_param	需要查询持仓的筛选条件,其中组合策略代码可以初始化为空,表示查询所有,如果不为空,请不带空格,并以'\0'结尾,注意需与market匹配,不匹配的话,可能导致查询不到所需的持仓
session_id	资金账户对应的session_id,登录时得到
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合约代码,则会查询此合约的持仓信息(注意请指定market,如果market为0,可能会查询到2个市场的持仓,如果market为其他非有效值,则查询结果会返回找不到持仓),如果合约代码为空,则默认查询所有持仓信息。

### 5.3.2.55 QueryOptionCombinedStrategyInfo()

#### 请求查询期权组合策略信息

### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

session⊷ _id	资金账户对应的session_id,登录时得到
request← id	用于用户定位查询响应的ID,由用户自定义

### 备注

该方法仅支持精确查询, 不支持模糊查询

#### 5.3.2.56 QueryOptionCombinedTrades()

## 请求查询期权组合策略的成交回报

### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

### 参数

query_param	需要查询的成交回报筛选条件,其中合约代码可以为空,则默认所有存在的合约代码,如果不为空,请不带空格,并以'\0'结尾,其中起始时间格式为YYYYMMDDHHMMSSsss,为0则默认当前交易日0点,结束时间格式为YYYYMMDDHHMMSSsss,为0则默认当前时间
session_id	资金账户对应的session_id,登录时得到
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,此函数查询出的结果可能对应多个查询结果响应。此函数不 建议轮询使用,当报单量过多时,容易造成用户线路拥堵,导致api断线

#### 5.3.2.57 QueryOptionCombinedTradesByPage()

### 分页请求查询期权组合策略成交回报

## 返回

### 参数

query_param	需要分页查询成交回报的条件,如果第一次查询,那么reference填0
session_id	资金账户对应的session_id,登录时得到
request_id	用于用户定位查询响应的ID,由用户自定义

## 备注

该方法支持分页查询,注意用户需要记录下最后一笔查询结果的reference以便用户下次查询使用

### 5.3.2.58 QueryOptionCombinedTradesByXTPID()

根据期权组合策略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,此时用户可以调用GetApiLastError()来获取错误代码

# 参数

order_xtp↔ _id	需要查询的委托编号,即InsertOrder()成功时返回 的order_xtp_id
session_id	资金账户对应的session_id,登录时得到
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查询结果响应

# 5.3.2.59 QueryOptionCombinedUnfinishedOrders()

请求查询期权组合策略未完结报单-旧版本接口

# 返回

### 参数

session← _id	资金账户对应的session_id,登录时得到
request← _id	用于用户定位查询响应的ID,由用户自定义

#### 5.3.2.60 QueryOptionCombinedUnfinishedOrdersEx()

请求查询期权组合策略未完结报单-新版本接口

返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

#### 参数

session← _id	资金账户对应的session_id,登录时得到
request⊷ _id	用于用户定位查询响应的ID,由用户自定义

## 5.3.2.61 QueryOrderByXTPID()

# 根据报单ID请求查询报单-旧版本接口

返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

order_xtp↔ _id	需要查询的报单在xtp系统中的ID,即InsertOrder()成功时返回 的order_xtp_id
session_id	资金账户对应的session_id,登录时得到
request_id	用于用户定位查询响应的ID,由用户自定义

## 5.3.2.62 QueryOrderByXTPIDEx()

### 根据报单ID请求查询报单-新版本接口

### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

## 参数

order_xtp↔ _id	需要查询的报单在xtp系统中的ID,即InsertOrder()成功时返回的order_xtp_id
session_id	资金账户对应的session_id,登录时得到
request_id	用于用户定位查询响应的ID,由用户自定义

## 5.3.2.63 QueryOrders()

# 请求查询报单-旧版本接口

## 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

query_param	需要查询的订单相关筛选条件,其中合约代码可以为空,则默认所有存在的合约代码,如果不为空,请不带空格,并以'\0'结尾,其中起始时间格式为YYYYMMDDHHMMSSsss,为0则默认当前交易日0点,结束时间格式为YYYYMMDDHHMMSSsss,为0则默认当前时间
session_id	资金账户对应的session_id,登录时得到
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,此函数查询出的结果可能对应多个查询结果响应。此函数不建议轮询使用,当报单量过多时,容易造成用户线路拥堵,导致api断线

### 5.3.2.64 QueryOrdersByPage()

#### 分页请求查询报单-旧版本接口

### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

#### 参数

query_param	需要分页查询订单的条件,如果第一次查询,那么query_param.reference填0
session_id	资金账户对应的session_id,登录时得到
request_id	用于用户定位查询响应的ID,由用户自定义

### 备注

该方法支持分页查询、注意用户需要记录下最后一笔查询结果的reference以便用户下次查询使用

#### 5.3.2.65 QueryOrdersByPageEx()

## 分页请求查询报单-新版本接口

# 返回

### 参数

query_param	需要分页查询订单的条件,如果第一次查询,那么query_param.reference填0
session_id	资金账户对应的session_id,登录时得到
request_id	用于用户定位查询响应的ID,由用户自定义

## 备注

该方法支持分页查询,注意用户需要记录下最后一笔查询结果的reference以便用户下次查询使用

#### 5.3.2.66 QueryOrdersEx()

#### 请求查询报单-新版本接口

### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

## 参数

query_param	需要查询的订单相关筛选条件,其中合约代码可以为空,则默认所有存在的合约代码,如果不为空,请不带空格,并以'\0'结尾,其中起始时间格式为YYYYMMDDHHMMSSsss,为0则默认当前交易日0点,结束时间格式为YYYYMMDDHHMMSSsss,为0则默认当前时间
session_id	资金账户对应的session_id,登录时得到
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,此函数查询出的结果可能对应多个查询结果响应。此函数不建议轮询使用,当报单量过多时,容易造成用户线路拥堵,导致api断线

# 5.3.2.67 QueryOtherServerFund()

请求查询其他节点可用资金

### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

## 参数

query_param	查询时需要提供的信息
session_id	资金账户对应的session_id,登录时得到
request_id	用于用户定位查询响应的ID,由用户自定义

## 5.3.2.68 QueryPosition()

## 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,此时用户可以调用GetApiLastError()来获取错误代码

## 参数

ticker	需要查询持仓的合约代码,可以为NULL,表示查询全市场,如果不为NULL,请不带空格,并以'\0'结尾,注意需与market匹配,不匹配的话,可能由于证券代码沪深2个市场有重复,而导致查询不到所需的持仓
market	需要查询持仓的合约所在市场,默认为0,仅在合约代码不为NULL的时候,才会使用。market不指定或者为非0的其他非有效值情况下,可能由于证券代码沪深2个市场有重复,而导致查询不到所需的持仓。如果想正确查询指定持仓,请指定market
session← _id	资金账户对应的session_id,登录时得到
request <i>←</i> _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合约代码,则会查询此合约的持仓信息(注意请指定market,如果market为0,可能会查询到2个市场的持仓,如果market为其他非有效值,则查询结果会返回找不到持仓),如果合约代码为空,则默认查询所有持仓信息。

#### 5.3.2.69 QueryStrategy()

```
virtual int QueryStrategy (
          uint32_t strategy_type,
          uint64_t client_strategy_id,
          uint64_t xtp_strategy_id,
          uint64_t session_id,
          int32_t request_id ) [pure virtual]
```

## algo业务中查询用户策略请求

## 返回

请求发送是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

### 参数

strategy_type	需要查询的策略类型,可填0
client_strategy⊷ _id	需要查询的策略用户自定义id,可填0
xtp_strategy_id	需要查询的策略在xtp系统中的id,如果指定,就一定按指定查询,如果填0,则按 其他筛选条件查询
session_id	资金账户对应的session_id,登录时得到
request_id	用于用户定位查询响应的ID,由用户自定义

## 备注

xtp\_strategy\_id条件的优先级最高,只有当xtp\_strategy\_id为0时,其他条件才生效,此条请求可能对应多条回应消息

## 5.3.2.70 QueryStructuredFund()

### 请求查询分级基金

#### 返回

### 参数

query_param	需要查询的分级基金筛选条件,其中母基金代码可以为空,则默认所有存在的母基 金,如果不为空,请不带空格,并以\0'结尾,其中交易市场不能为空
session_id	资金账户对应的session_id,登录时得到
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查询结果响应

## 5.3.2.71 QueryTrades()

#### 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,此时用户可以调用GetApiLastError()来获取错误代码

### 参数

query_param	需要查询的成交回报筛选条件,其中合约代码可以为空,则默认所有存在的合约代码,如果不为空,请不带空格,并以\0'结尾,其中起始时间格式为YYYYMMDDHHMMSSsss,为0则默认当前交易日0点,结束时间格式为YYYYMMDDHHMMSSsss,为0则默认当前时间
session_id	资金账户对应的session_id,登录时得到
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,此函数查询出的结果可能对应多个查询结果响应。此函数不 建议轮询使用,当报单量过多时,容易造成用户线路拥堵,导致api断线

## 5.3.2.72 QueryTradesByPage()

### 分页请求查询成交回报

5.3 TraderApi类 参考 57

#### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

### 参数

query_param	需要分页查询成交回报的条件,如果第一次查询,那么reference填0	
session_id	资金账户对应的session_id,登录时得到	
request_id	用于用户定位查询响应的ID,由用户自定义	

### 备注

该方法支持分页查询,注意用户需要记录下最后一笔查询结果的reference以便用户下次查询使用

### 5.3.2.73 QueryTradesByXTPID()

### 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,此时用户可以调用GetApiLastError()来获取错误代码

#### 参数

order_xtp↔ _id	需要查询的委托编号,即InsertOrder()成功时返回 的order_xtp_id
session_id	资金账户对应的session_id,登录时得到
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查询结果响应

#### 5.3.2.74 QueryUnfinishedOrders()

请求查询未完结报单-旧版本接口

### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

### 参数

session←	资金账户对应的session_id,登录时得到
_id	
request←	用于用户定位查询响应的ID,由用户自定义
_id	

#### 5.3.2.75 QueryUnfinishedOrdersEx()

### 请求查询未完结报单-新版本接口

#### 返回

查询是否成功,"0"表示成功,非"0"表示出错,此时用户可以调用GetApiLastError()来获取错误代码

#### 参数

session←	资金账户对应的session_id,登录时得到
_id	
request←	用于用户定位查询响应的ID,由用户自定义
_id	

#### 5.3.2.76 RegisterSpi()

# 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 TraderApi类 参考 59

#### 5.3.2.77 Release()

```
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

### 5.3.2.78 SetHeartBeatInterval()

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

### 5.3.2.79 SetSoftwareKey()

# 设置软件开发Key

参数

key 用户开发软件Key,用户申请开户时给予,以'\0'结尾

备注

此函数必须在Login之前调用

#### 5.3.2.80 SetSoftwareVersion()

### 设置软件开发版本号

#### 参数

version 用户开发软件版本号,非api发行版本号,长度不超过15位,以'\0'结尾

备注

此函数必须在Login之前调用,标识的是客户端版本号,而不是API的版本号,由用户自定义

### 5.3.2.81 SubscribePublicTopic()

订阅公共流。

#### 参数

resume\_type

公共流(订单响应、成交回报) 重传方式

XTP\_TERT\_RESTART:从本交易日开始重传 XTP\_TERT\_RESUME:(保留字段,此方式暂未支持)从上次收到的续传 XTP\_TERT\_QUICK:只传送登录后公共流的内容

备注

该方法要在Login方法前调用。若不调用则不会收到公共流的数据。注意在用户断线后,如果不登出就login(),公共流订阅方式不会起作用。用户只会收到断线后的所有消息。如果先logout()再login(),那么公共流订阅方式会起作用,用户收到的数据会根据用户的选择方式而定。

该类的文档由以下文件生成:

• xtp\_trader\_api.h

# 5.4 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 (uint64 t session id, int reason)
- virtual void OnError (XTPRI \*error\_info)
- virtual void OnQueryAccountTradeMarket (int trade\_location, XTPRI \*error\_info, int request\_id, uint64\_←
  t session id)
- virtual void OnOrderEvent (XTPOrderInfo \*order info, XTPRI \*error info, uint64 t session id)
- virtual void OnTradeEvent (XTPTradeReport \*trade info, uint64 t session id)
- virtual void OnCancelOrderError (XTPOrderCancelInfo \*cancel info, XTPRI \*error info, uint64 t session id)
- virtual void OnQueryOrder (XTPQueryOrderRsp \*order\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64 t session id)
- virtual void OnQueryOrderEx (XTPOrderInfoEx \*order\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64 t session id)

- virtual void OnQueryTrade (XTPQueryTradeRsp \*trade\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnQueryPosition (XTPQueryStkPositionRsp \*position, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnQueryAsset (XTPQueryAssetRsp \*asset, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnQueryStructuredFund (XTPStructuredFundInfo \*fund\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnQueryFundTransfer (XTPFundTransferNotice \*fund\_transfer\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnFundTransfer (XTPFundTransferNotice \*fund\_transfer\_info, XTPRI \*error\_info, uint64\_←
  t session id)
- virtual void OnQueryOtherServerFund (XTPFundQueryRsp \*fund\_info, XTPRI \*error\_info, int request\_id, uint64\_t session\_id)
- virtual void OnQueryETF (XTPQueryETFBaseRsp \*etf\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnQueryETFBasket (XTPQueryETFComponentRsp \*etf\_component\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnQueryIPOInfoList (XTPQueryIPOTickerRsp \*ipo\_info, XTPRI \*error\_info, int request\_id, bool is last, uint64 t session id)
- virtual void OnQueryIPOQuotaInfo (XTPQueryIPOQuotaRsp \*quota\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnQueryOptionAuctionInfo (XTPQueryOptionAuctionInfoRsp \*option\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnCreditCashRepay (XTPCrdCashRepayRsp \*cash\_repay\_info, XTPRI \*error\_info, uint64\_←
  t session\_id)
- virtual void OnCreditCashRepayDebtInterestFee (XTPCrdCashRepayDebtInterestFeeRsp \*cash\_repay\_info, XTPRI \*error\_info, uint64\_t session\_id)
- virtual void OnQueryCreditCashRepayInfo (XTPCrdCashRepayInfo \*cash\_repay\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)

virtual void OnQueryCreditFundInfo (XTPCrdFundInfo \*fund\_info, XTPRI \*error\_info, int request\_id, uint64
 \_t session\_id)

- virtual void OnQueryCreditDebtInfo (XTPCrdDebtInfo \*debt\_info, XTPRI \*error\_info, int request\_id, bool is
   last, uint64 t session id)
- virtual void OnQueryCreditTickerDebtInfo (XTPCrdDebtStockInfo \*debt\_info, XTPRI \*error\_info, int request
   —id, bool is\_last, uint64\_t session\_id)
- virtual void OnQueryCreditAssetDebtInfo (double remain\_amount, XTPRI \*error\_info, int request\_id, uint64← t session id)
- virtual void OnQueryCreditTickerAssignInfo (XTPClientQueryCrdPositionStkInfo \*assign\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnQueryCreditExcessStock (XTPClientQueryCrdSurplusStkRspInfo \*stock\_info, XTPRI \*error
  info, int request id, uint64 t session id)
- virtual void OnQueryMulCreditExcessStock (XTPClientQueryCrdSurplusStkRspInfo \*stock\_info, XTPRI \*error\_info, int request\_id, uint64\_t session\_id, bool is\_last)
- virtual void OnCreditExtendDebtDate (XTPCreditDebtExtendNotice \*debt\_extend\_info, XTPRI \*error\_info, uint64\_t session\_id)
- virtual void OnQueryCreditExtendDebtDateOrders (XTPCreditDebtExtendNotice \*debt\_extend\_info, XTPRI \*error info, int request id, bool is last, uint64 t session id)
- virtual void OnQueryCreditFundExtraInfo (XTPCrdFundExtraInfo \*fund\_info, XTPRI \*error\_info, int request
   —id, uint64\_t session\_id)
- virtual void OnQueryCreditPositionExtraInfo (XTPCrdPositionExtraInfo \*fund\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnOptionCombinedOrderEvent (XTPOptCombOrderInfo \*order\_info, XTPRI \*error\_info, uint64←
   \_t session\_id)
- virtual void OnOptionCombinedTradeEvent (XTPOptCombTradeReport \*trade info, uint64 t session id)
- virtual void OnCancelOptionCombinedOrderError (XTPOptCombOrderCancelInfo \*cancel\_info, XTPRI \*error info, uint64 t session id)
- virtual void OnQueryOptionCombinedOrders (XTPQueryOptCombOrderRsp \*order\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnQueryOptionCombinedOrdersEx (XTPOptCombOrderInfoEx \*order\_info, XTPRI \*error\_info, int request id, bool is last, uint64 t session id)
- virtual void OnQueryOptionCombinedOrdersByPage (XTPQueryOptCombOrderRsp \*order\_info, int64\_

  t req\_count, int64\_t order\_sequence, int64\_t query\_reference, int request\_id, bool is\_last, uint64\_t session

  \_id)
- virtual void OnQueryOptionCombinedOrdersByPageEx (XTPOptCombOrderInfoEx \*order\_info, int64\_t req
   —count, int64\_t order\_sequence, int64\_t query\_reference, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnQueryOptionCombinedTrades (XTPQueryOptCombTradeRsp \*trade\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnQueryOptionCombinedTradesByPage (XTPQueryOptCombTradeRsp \*trade\_info, int64\_
   t req\_count, int64\_t trade\_sequence, int64\_t query\_reference, int request\_id, bool is\_last, uint64\_t session
   \_id)
- virtual void OnQueryOptionCombinedPosition (XTPQueryOptCombPositionRsp \*position\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnQueryOptionCombinedStrategyInfo (XTPQueryCombineStrategyInfoRsp \*strategy\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnQueryOptionCombinedExecPosition (XTPQueryOptCombExecPosRsp \*position\_info, XTPRI \*error\_info, int request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnQueryStrategy (XTPStrategyInfoStruct \*strategy\_info, char \*strategy\_param, XTPRI \*error
   — info, int32\_t request\_id, bool is\_last, uint64\_t session\_id)
- virtual void OnStrategyStateReport (XTPStrategyStateReportStruct \*strategy\_state, uint64\_t session\_id)
- virtual void OnALGOUserEstablishChannel (char \*user, XTPRI \*error info, uint64 t session id)
- virtual void OnInsertAlgoOrder (XTPStrategyInfoStruct \*strategy\_info, XTPRI \*error\_info, uint64\_t session ← id)
- virtual void OnCancelAlgoOrder (XTPStrategyInfoStruct \*strategy\_info, XTPRI \*error\_info, uint64\_←
   t session\_id)
- virtual void OnAlgoDisconnected (int reason)

- virtual void OnAlgoConnected () 当客户端与*AlgoBus*断线后重新连接时,该方法被调用,仅在断线重连成功后会被调用。
- virtual void OnStrategySymbolStateReport (XTPStrategySymbolStateReport \*strategy\_symbol\_state, uint64\_t session\_id)

# 5.4.1 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

# 5.4.2 成员函数说明

#### 5.4.2.1 OnAlgoDisconnected()

当客户端与AlgoBus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

请不要堵塞此线程,否则会影响algo的登录,与Algo之间的连接,断线后会自动重连,用户无需做其他操作

被 DemoTestTraderSpi 重载.

### 5.4.2.2 OnALGOUserEstablishChannel()

algo业务中用户建立算法通道的消息响应

#### 参数

user	用户名
error_info	建立算法通道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,即算法通道成功
session← _id	资金账户对应的session_id,登录时得到

### 备注

算法通道建立成功后,才能对用户创建策略等操作,一个用户只能拥有一个算法通道,如果之前已经建立,则无需重复建立

被 DemoTestTraderSpi 重载.

#### 5.4.2.3 OnCancelAlgoOrder()

algo业务中撤销策略单的响应

#### 参数

strategy_info	用户撤销的策略单的具体信息
error_info	撤销策略单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
session_id	资金账户对应的session_id,登录时得到

### 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 DemoTestTraderSpi 重载.

### 5.4.2.4 OnCancelOptionCombinedOrderError()

期权组合策略撤单出错响应

### 参数

cancel_info	撤单具体信息,包括撤单的order_cancel_xtp_id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,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,当error_info为空,或者error_info.error_id为0时,表明没有错误
session_id	资金账户对应的session_id,登录时得到

### 备注

此响应只会在撤单发生错误时被回调

### 5.4.2.5 OnCancelOrderError()

#### 撤单出错响应

# 参数

cancel_info	撤单具体信息,包括撤单的order_cancel_xtp_id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,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,当error_info为空,或者error_info.error_id为0时,表明没有错误
session_id	资金账户对应的session_id,登录时得到

### 备注

此响应只会在撤单发生错误时被回调

#### 5.4.2.6 OnCreditCashRepay()

融资融券业务中现金直接还款的响应

#### 参数

cash_repay_info	现金直接还款通知的具体信息,用户可以通过cash_repay_info.xtp_id来管理订单,通过GetClientIDByXTPID() == client_id来过滤自己的订单。
error_info	现金还款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
session_id	资金账户对应的session_id,登录时得到

### 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

### 5.4.2.7 OnCreditCashRepayDebtInterestFee()

融资融券业务中现金还息的响应

# 参数

cash_repay_info	现金还息通知的具体信息,用户可以通过cash_repay_info.xtp_id来管理订单,通过GetClientIDByXTPID() == client_id来过滤自己的订单。
error_info	现金还息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
session_id	资金账户对应的session_id,登录时得到

### 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

# 5.4.2.8 OnCreditExtendDebtDate()

融资融券业务中负债合约展期的通知

#### 参数

debt_extend_info	负债合约展期通知的具体信息,用户可以通过debt_extend_info.xtpid来管理订单,通过GetClientIDByXTPID() == client_id来过滤自己的订单。
error_info	负债合约展期订单被拒绝或者发生错误时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。
session_id	资金账户对应的session_id,登录时得到

### 备注

当负债合约展期订单有状态变化的时候,会被调用,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。所有登录了此用户的客户端都将收到此用户的负债合约展期通知。

#### 5.4.2.9 OnDisconnected()

当客户端的某个连接与交易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
session← _id	资金账户对应的session_id,登录时得到

### 备注

用户主动调用logout导致的断线,不会触发此函数。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,并更新session\_id,此时用户收到的数据跟断线之前是连续的

被 DemoTestTraderSpi 重载.

#### 5.4.2.10 OnError()

#### 错误应答

#### 参数

error\_info | 当

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

#### 5.4.2.11 OnFundTransfer()

### 资金划拨通知

#### 参数

fund_transfer_info	资金划拨通知的具体信息,用户可以通过fund_transfer_info.serial_id来管理订单,通过GetClientIDByXTPID() == client_id来过滤自己的订单。
error_info	资金划拨订单被拒绝或者发生错误时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。当资金划拨方向为一号两中心节点之间划拨,且error_info.error_id=11000384时,error_info.error_msg中含有对方结点中可用于划拨的资金(以整数为准),用户需解析后进行stringToInt的转化,可据此填写合适的资金,再次发起划拨请求
session_id	资金账户对应的session_id,登录时得到

# 备注

当资金划拨订单有状态变化的时候,会被调用,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。所有登录了此用户的客户端都将收到此用户的资金划拨通知。

# 5.4.2.12 OnInsertAlgoOrder()

#### algo业务中报送策略单的响应

#### 参数

strategy_info	用户报送的策略单的具体信息
error_info	报送策略单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
session_id	资金账户对应的session_id,登录时得到

### 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 DemoTestTraderSpi 重载.

#### 5.4.2.13 OnOptionCombinedOrderEvent()

### 期权组合策略报单通知

#### 参数

order_info	订单响应具体信息,用户可以通过order_info.order_xtp_id来管理订单,通过GetClientIDByXTPID() == client_id来过滤自己的订单,order_info.qty_left字段在订单为未成交、部成、全成、废单状态时,表示此订单还没有成交的数量,在部撤、全撤状态时,表示此订单被撤的数量。order_info.order_cancel_xtp_id为其所对应的撤单ID,不为0时表示此单被撤成功
error_info	订单被拒绝或者发生错误时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
session⊷ _id	资金账户对应的session_id,登录时得到

#### 备注

每次订单状态更新时,都会被调用,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,在订单未成交、全部成交、全部撤单、部分撤单、已拒绝这些状态时会有响应,对于部分成交的情况,请由订单的成交回报来自行确认。所有登录了此用户的客户端都将收到此用户的订单响应

#### 5.4.2.14 OnOptionCombinedTradeEvent()

期权组合策略成交通知

#### 参数

trade_info	成交回报的具体信息,用户可以通过trade_info.order_xtp_id来管理订单,通过GetClientIDByXTPID() == client_id来过滤自己的订单。对于上交所,exec_id可以唯一标识一笔成交。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了。对于深交所,exec_id是唯一的,暂时无此判断机制。report_index+market字段可以组成唯一标识表示成交回报。
session← _id	资金账户对应的session_id,登录时得到

### 备注

订单有成交发生的时候,会被调用,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。所有登录了此用户的客户端都将收到此用户的成交回报。相关订单为部成状态,需要用户通过成交回报的成交数量来确定,OnOrderEvent()不会推送部成状态。

#### 5.4.2.15 OnOrderEvent()

#### 报单通知

### 参数

order_info	订单响应具体信息,用户可以通过order_info.order_xtp_id来管理订单,通过GetClientIDByXTPID() == client_id来过滤自己的订单,order_info.qty_left字段在订单为未成交、部成、全成、废单状态时,表示此订单还没有成交的数量,在部撤、全撤状态时,表示此订单被撤的数量。order_info.order_cancel_xtp_id为其所对应的撤单ID,不为0时表示此单被撤成功
error_info	订单被拒绝或者发生错误时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
session⊷ _id	资金账户对应的session_id,登录时得到

#### 备注

每次订单状态更新时,都会被调用,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,在订单未成交、全部成交、全部撤单、部分撤单、已拒绝这些状态时会有响应,对于部分成交的情况,请由订单的成交回报来自行确认。所有登录了此用户的客户端都将收到此用户的订单响应

#### 5.4.2.16 OnQueryAccountTradeMarket()

```
virtual void OnQueryAccountTradeMarket (
    int trade_location,
    XTPRI * error_info,
    int request_id,
    uint64_t session_id ) [inline], [virtual]
```

请求查询用户在本节点上可交易市场的响应

# 参数

trade_location	查询到的交易市场信息,按位来看,从低位开始数,第0位表示沪市,即如果(trade_location&0x01) == 0x01,代表可交易沪市,第1位表示深市,即如果(trade_location&0x02) == 0x02,表示可交易深市,如果第0位和第1位均是1,即(trade_location&(0x01 0x02)) == 0x03,就表示可交易沪深2个市场
error_info	查询可交易市场发生错误时,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
request_id	此消息响应函数对应的请求ID
session_id	资金账户对应的session_id,登录时得到

### 备注

此查询只会有一个结果

# 5.4.2.17 OnQueryAsset()

请求查询资金账户响应,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

asset	查询到的资金账户情况
error_info	查询资金账户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
request <i>⇔</i> _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,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
session← _id	资金账户对应的session_id,登录时得到

5.4 TraderSpi类 参考 73

#### 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

#### 5.4.2.18 OnQueryCreditAssetDebtInfo()

请求查询信用账户待还资金的响应,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

### 参数

remain_amount	查询到的信用账户待还资金
error_info	查询信用账户待还资金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
request_id	此消息响应函数对应的请求ID
session_id	资金账户对应的session_id,登录时得到

### 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

#### 5.4.2.19 OnQueryCreditCashRepayInfo()

请求查询融资融券业务中的现金直接还款报单的响应

cash_repay_info	查询到的某一笔现金直接还款通知的具体信息
error_info	查询现金直接报单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
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,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
session_id	资金账户对应的session_id,登录时得到

#### 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

#### 5.4.2.20 OnQueryCreditDebtInfo()

请求查询信用账户负债信息的响应,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

### 参数

debt_info	查询到的信用账户合约负债情况
error_info	查询信用账户负债信息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
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,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
session← _id	资金账户对应的session_id,登录时得到

# 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

### 5.4.2.21 OnQueryCreditExcessStock()

融资融券业务中请求查询指定余券信息的响应,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

stock_info	查询到的余券信息
error_info	查询信用账户余券信息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

5.4 TraderSpi类 参考 75

#### 参数

request←	此消息响应函数对应的请求ID
_id	
session⊷	资金账户对应的session_id,登录时得到
_id	

#### 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

### 5.4.2.22 OnQueryCreditExtendDebtDateOrders()

查询融资融券业务中负债合约展期订单响应,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

### 参数

debt_extend_info	查询到的负债合约展期情况
error_info	查询负债合约展期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。当error_info.error_id=11000350时,表明 没有记录,当为其他非0值时,表明合约发生拒单时的错误原因
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,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
session_id	资金账户对应的session_id,登录时得到

# 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

### 5.4.2.23 OnQueryCreditFundExtraInfo()

```
int request_id,
uint64_t session_id ) [inline], [virtual]
```

查询融资融券业务中信用账户附加信息的响应,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

5.4 TraderSpi类 参考 77

#### 参数

fund_info	信用账户附加信息
error_info	查询信用账户附加信息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
request <i>←</i> _id	此消息响应函数对应的请求ID
session← _id	资金账户对应的session_id,登录时得到

### 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

### 5.4.2.24 OnQueryCreditFundInfo()

请求查询信用账户额外信息的响应,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

# 参数

fund_info	查询到的信用账户额外信息情况
error_info	查询信用账户额外信息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
request <i>⇔</i> _id	此消息响应函数对应的请求ID
session← _id	资金账户对应的session_id,登录时得到

# 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

#### 5.4.2.25 OnQueryCreditPositionExtraInfo()

```
int request_id,
bool is_last,
uint64_t session_id ) [inline], [virtual]
```

查询融资融券业务中信用账户指定证券的附加信息的响应,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

#### 参数

fund_info	信用账户指定证券的附加信息
error_info	查询信用账户附加信息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
request <i>←</i> _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,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
session← _id	资金账户对应的session_id,登录时得到

### 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

### 5.4.2.26 OnQueryCreditTickerAssignInfo()

请求查询信用账户可融券头寸信息的响应,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

### 参数

assign_info	查询到的信用账户可融券头寸信息
error_info	查询信用账户可融券头寸信息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
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,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
session_id	资金账户对应的session_id,登录时得到

# 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

5.4 TraderSpi类 参考 79

### 5.4.2.27 OnQueryCreditTickerDebtInfo()

请求查询信用账户指定证券负债未还信息响应,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

### 参数

debt_info	查询到的信用账户指定证券负债未还信息情况
error_info	查询信用账户指定证券负债未还信息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
request <i>⇔</i> _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,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
session← _id	资金账户对应的session_id,登录时得到

### 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

#### 5.4.2.28 OnQueryETF()

请求查询ETF清单文件的响应,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

etf_info	查询到的ETF清单文件情况
error_info	查询ETF清单文件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
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,当为最后一个的时候
ー 制作者 Doxygen	为true,如果为false,表示还有其他后续消息响应
session←	资金账户对应的session_id,登录时得到
_id	

### 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

#### 5.4.2.29 OnQueryETFBasket()

请求查询ETF股票篮的响应,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

### 参数

etf_component_info	查询到的ETF合约的相关成分股信息
error_info	查询ETF股票篮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
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,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
session_id	资金账户对应的session_id,登录时得到

### 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

### 5.4.2.30 OnQueryFundTransfer()

请求查询资金划拨订单响应、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

fund_transfer_info	查询到的资金账户情况
error_info	查询资金账户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
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,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
session_id	资金账户对应的session_id,登录时得到

5.4 TraderSpi类 参考 81

#### 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

#### 5.4.2.31 OnQueryIPOInfoList()

请求查询今日新股申购信息列表的响应,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

#### 参数

ipo_info	查询到的今日新股申购的一只股票信息
error_info	查询今日新股申购信息列表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
request <i>←</i> _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,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
session← _id	资金账户对应的session_id,登录时得到

### 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

#### 5.4.2.32 OnQueryIPOQuotaInfo()

请求查询用户新股申购额度信息的响应,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

#### 参数

quota_info	查询到的用户某个市场的今日新股申购额度信息
error_info	查查询用户新股申购额度信息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
request <i>⇔</i> _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,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
session← _id	资金账户对应的session_id,登录时得到

### 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

#### 5.4.2.33 OnQueryMulCreditExcessStock()

融资融券业务中请求查询余券信息的响应,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

#### 参数

stock_info	查询到的余券信息
error_info	查询信用账户余券信息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
request <i>←</i> _id	此消息响应函数对应的请求ID
session← _id	资金账户对应的session_id,登录时得到
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,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

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

5.4 TraderSpi类 参考 83

#### 5.4.2.34 OnQueryOptionAuctionInfo()

请求查询期权合约的响应,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

#### 参数

option_info	查询到的期权合约情况
error_info	查询期权合约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
request <i>←</i> _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,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
session⊷ _id	资金账户对应的session_id,登录时得到

#### 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

### 5.4.2.35 OnQueryOptionCombinedExecPosition()

查询期权行权合并头寸的响应

position_info	查询到的一个行权合并头寸信息
error_info	查询持仓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
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,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
session_id	资金账户对应的session_id,登录时得到

#### 备注

一个查询请求可能对应多个响应,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。

#### 5.4.2.36 OnQueryOptionCombinedOrders()

请求查询期权组合策略报单响应-旧版本接口

#### 参数

order_info	查询到的一个报单
error_info	查询报单时发生错误时,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
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,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
session← _id	资金账户对应的session_id,登录时得到

### 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,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。此对应的请求函数不建议轮询使用,当报单量过多时,容易造成用户线路拥堵,导致**api**断线

#### 5.4.2.37 OnQueryOptionCombinedOrdersByPage()

分页请求查询期权组合策略报单响应-旧版本接口

#### 参数

order_info	查询到的一个报单
req_count	分页请求的最大数量
order_sequence	分页请求的当前回报数量
query_reference	当前报单信息所对应的查询索引,需要记录下来,在进行下一次分页查询的时候需 要用到
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,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
session_id	资金账户对应的session_id,登录时得到

### 备注

当order\_sequence为0,表明当次查询没有查到任何记录,当is\_last为true时,如果order\_sequence等于req\_count,那么表示还有报单,可以进行下一次分页查询,如果不等,表示所有报单已经查询完毕。一个查询请求可能对应多个响应,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。

#### 5.4.2.38 OnQueryOptionCombinedOrdersByPageEx()

### 分页请求查询期权组合策略报单响应-新版本接口

order_info	查询到的一个报单
req_count	分页请求的最大数量
order_sequence	分页请求的当前回报数量
query_reference	当前报单信息所对应的查询索引,需要记录下来,在进行下一次分页查询的时候需 要用到
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,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
session_id	资金账户对应的session_id,登录时得到

#### 备注

当order\_sequence为0,表明当次查询没有查到任何记录,当is\_last为true时,如果order\_sequence等于req\_count,那么表示还有报单,可以进行下一次分页查询,如果不等,表示所有报单已经查询完毕。一个查询请求可能对应多个响应,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。

#### 5.4.2.39 OnQueryOptionCombinedOrdersEx()

请求查询期权组合策略报单响应-新版本接口

#### 参数

order_info	查询到的一个报单
error_info	查询报单时发生错误时,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
request <i>←</i> _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,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
session← _id	资金账户对应的session_id,登录时得到

### 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,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。此对应的请求函数不建议轮询使用,当报单量过多时,容易造成用户线路拥堵,导致api断线

# 5.4.2.40 OnQueryOptionCombinedPosition()

# 请求查询期权组合策略持仓响应

#### 参数

position_info	查询到的一个持仓信息
error_info	查询持仓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
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,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
session_id	资金账户对应的session_id,登录时得到

#### 备注

一个查询请求可能对应多个响应,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。

#### 5.4.2.41 OnQueryOptionCombinedStrategyInfo()

#### 请求查询期权组合策略信息响应

### 参数

strategy_info	查询到的一个组合策略信息
error_info	查询成交回报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
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,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
session_id	资金账户对应的session_id,登录时得到

### 备注

一个查询请求可能对应多个响应,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。

### 5.4.2.42 OnQueryOptionCombinedTrades()

```
\label{thm:condition} \mbox{virtual void OnQueryOptionCombinedTrades (} $$ \mbox{XTPQueryOptCombTradeRsp} * trade_info, $$
```

```
XTPRI * error_info,
int request_id,
bool is_last,
uint64_t session_id ) [inline], [virtual]
```

### 请求查询期权组合策略成交响应

#### 参数

trade_info	查询到的一个成交回报
error_info	查询成交回报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
request <i>⊷</i> _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,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
session← _id	资金账户对应的session_id,登录时得到

### 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,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。此对应的请求函数不建议轮询使用,当报单量过多时,容易造成用户线路拥堵,导致**api**断线

# 5.4.2.43 OnQueryOptionCombinedTradesByPage()

#### 分页请求查询期权组合策略成交响应

trade_info	查询到的一个成交信息
req_count	分页请求的最大数量
trade_sequence	分页请求的当前回报数量
query_reference	当前报单信息所对应的查询索引,需要记录下来,在进行下一次分页查询的时候需 要用到
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,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
session_id	资金账户对应的session_id,登录时得到

#### 备注

当trade\_sequence为0,表明当次查询没有查到任何记录,当is\_last为true时,如果trade\_sequence等于req\_count,那么表示还有回报,可以进行下一次分页查询,如果不等,表示所有回报已经查询完毕。一个查询请求可能对应多个响应,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。

#### 5.4.2.44 OnQueryOrder()

#### 请求查询报单响应-旧版本接口

#### 参数

order_info	查询到的一个报单
error_info	查询报单时发生错误时,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
request <i>⇔</i> _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,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
session← _id	资金账户对应的session_id,登录时得到

#### 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,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。此对应的请求函数不建议轮询使用,当报单量过多时,容易造成用户线路拥堵,导致api断线

### 5.4.2.45 OnQueryOrderByPage()

分页请求查询报单响应-旧版本接口

#### 参数

order_info	查询到的一个报单
req_count	分页请求的最大数量
order_sequence	分页请求的当前回报数量
query_reference	当前报单信息所对应的查询索引,需要记录下来,在进行下一次分页查询的时候需 要用到
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,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
session_id	资金账户对应的session_id,登录时得到

### 备注

当order\_sequence为0,表明当次查询没有查到任何记录,当is\_last为true时,如果order\_sequence等于req\_count,那么表示还有报单,可以进行下一次分页查询,如果不等,表示所有报单已经查询完毕。一个查询请求可能对应多个响应,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。

#### 5.4.2.46 OnQueryOrderByPageEx()

### 分页请求查询报单响应-新版本接口

order_info	查询到的一个报单
req_count	分页请求的最大数量
order_sequence	分页请求的当前回报数量
query_reference	当前报单信息所对应的查询索引,需要记录下来,在进行下一次分页查询的时候需 要用到
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,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
session_id	资金账户对应的session_id,登录时得到

#### 备注

当order\_sequence为0,表明当次查询没有查到任何记录,当is\_last为true时,如果order\_sequence等于req\_count,那么表示还有报单,可以进行下一次分页查询,如果不等,表示所有报单已经查询完毕。一个查询请求可能对应多个响应,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。

### 5.4.2.47 OnQueryOrderEx()

#### 请求查询报单响应-新版本接口

#### 参数

order_info	查询到的一个报单信息
error_info	查询报单时发生错误时,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
request <i>⇔</i> _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,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
session← _id	资金账户对应的session_id,登录时得到

### 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,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

#### 5.4.2.48 OnQueryOtherServerFund()

请求查询其他节点可用资金的响应,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

### 参数

fund_info	查询到的其他节点可用资金情况
error_info	查询其他节点可用资金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
request <i>←</i> _id	此消息响应函数对应的请求ID
session⊷ _id	资金账户对应的session_id,登录时得到

### 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

#### 5.4.2.49 OnQueryPosition()

# 请求查询投资者持仓响应

# 参数

position	查询到的一只股票的持仓情况
error_info	查询账户持仓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
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,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
session← _id	资金账户对应的session_id,登录时得到

### 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,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

#### 5.4.2.50 OnQueryStrategy()

5.4 TraderSpi类 参考 93

```
char * strategy_param,
XTPRI * error_info,
int32_t request_id,
bool is_last,
uint64_t session_id ) [inline], [virtual]
```

# algo业务中查询策略列表的响应

### 参数

strategy_info	策略具体信息	
strategy_param	此策略中包含的参数,如果error_info.error_id为0时,有意义	
error_info	查询查询策略列表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	
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,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	
session_id	资金账户对应的session_id,登录时得到	

### 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 DemoTestTraderSpi 重载.

#### 5.4.2.51 OnQueryStructuredFund()

```
virtual void OnQueryStructuredFund (
    XTPStructuredFundInfo * fund_info,
    XTPRI * error_info,
    int request_id,
    bool is_last,
    uint64_t session_id ) [inline], [virtual]
```

请求查询分级基金信息响应,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

#### 参数

fund_info	查询到的分级基金情况
error_info	查询分级基金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
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,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
session← _id	资金账户对应的session_id,登录时得到

#### 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

#### 5.4.2.52 OnQueryTrade()

#### 请求查询成交响应

#### 参数

trade_info	查询到的一个成交回报
error_info	查询成交回报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
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,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
session← _id	资金账户对应的session_id,登录时得到

### 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,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。此对应的请求函数不建议轮询使用,当报单量过多时,容易造成用户线路拥堵,导致api断线

#### 5.4.2.53 OnQueryTradeByPage()

```
virtual void OnQueryTradeByPage (
    XTPQueryTradeRsp * trade_info,
    int64_t req_count,
    int64_t trade_sequence,
    int64_t query_reference,
    int request_id,
    bool is_last,
    uint64_t session_id ) [inline], [virtual]
```

#### 分页请求查询成交响应

#### 参数

trade_info	查询到的一个成交信息	
req_count	分页请求的最大数量	
trade_sequence	分页请求的当前回报数量	
query_reference	当前报单信息所对应的查询索引,需要记录下来,在进行下一次分页查询的时候需要用到	
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,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	
session_id	资金账户对应的session_id,登录时得到	

#### 备注

当trade\_sequence为0,表明当次查询没有查到任何记录,当is\_last为true时,如果trade\_sequence等于req\_count,那么表示还有回报,可以进行下一次分页查询,如果不等,表示所有回报已经查询完毕。一个查询请求可能对应多个响应,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。

#### 5.4.2.54 OnStrategyStateReport()

algo业务中策略运行时策略状态通知

#### 参数

strategy_state	用户策略运行情况的状态通知
session_id	资金账户对应的session_id,登录时得到

# 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 DemoTestTraderSpi 重载.

#### 5.4.2.55 OnStrategySymbolStateReport()

algo业务中策略运行时策略指定证券执行状态通知

#### 参数

strategy_symbol_state	用户策略指定证券运行情况的状态通知
session_id	资金账户对应的session_id,登录时得到

#### 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

#### 5.4.2.56 OnTradeEvent()

#### 成交通知

#### 参数

trade_info	成交回报的具体信息,用户可以通过trade_info.order_xtp_id来管理订单,通过GetClientIDByXTPID() == client_id来过滤自己的订单。对于上交所,exec_id可以唯一标识一笔成交。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了。对于深交所,exec_id是唯一的,暂时无此判断机制。report_index+market字段可以组成唯一标识表示成交回报。
session← _id	资金账户对应的session_id,登录时得到

#### 备注

订单有成交发生的时候,会被调用,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。所有登录了此用户的客户端都将收到此用户的成交回报。相关订单为部成状态,需要用户通过成交回报的成交数量来确定,OnOrderEvent()不会推送部成状态。

该类的文档由以下文件生成:

• xtp\_trader\_api.h

# 5.5 XTPClientQueryCrdDebtStockReq结构体参考

融资融券指定证券上的负债未还数量请求结构体

#include <xoms\_api\_struct.h>

# 成员变量

• XTP\_MARKET\_TYPE market 市场

char ticker [XTP\_TICKER\_LEN]
 证券代码

#### 5.5.1 详细描述

融资融券指定证券上的负债未还数量请求结构体

该结构体的文档由以下文件生成:

• xoms\_api\_struct.h

# 5.6 XTPClientQueryCrdPositionStkInfo结构体参考

融券头寸证券信息

#include <xoms\_api\_struct.h>

### 成员变量

• XTP\_MARKET\_TYPE market

证券市场

• char ticker [XTP\_TICKER\_LEN] 证券代码

int64\_t limit\_qty

融券限量(保留字段)

int64\_t yesterday\_qty
 昨日日融券数量(保留字段)

int64\_t left\_qty

剩余可融券数量

int64\_t frozen\_qty

冻结融券数量(保留字段)

# 5.6.1 详细描述

融券头寸证券信息

该结构体的文档由以下文件生成:

# 5.7 XTPClientQueryCrdPositionStockReq结构体参考

#### 融券头寸证券查询请求结构体

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

### 成员变量

XTP\_MARKET\_TYPE market

证券市场

char ticker [XTP\_TICKER\_LEN]
 证券代码

### 5.7.1 详细描述

融券头寸证券查询请求结构体

该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

# 5.8 XTPClientQueryCrdSurplusStkReqInfo结构体参考

信用业务余券查询请求结构体

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

### 成员变量

- XTP\_MARKET\_TYPE market
  - 证券市场
- char ticker [XTP\_TICKER\_LEN] 证券代码

# 5.8.1 详细描述

信用业务余券查询请求结构体

该结构体的文档由以下文件生成:

# 5.9 XTPClientQueryCrdSurplusStkRspInfo结构体参考

#### 信用业务余券信息

#include <xoms\_api\_struct.h>

### 成员变量

XTP MARKET TYPE market

证券市场

char ticker [XTP\_TICKER\_LEN]

证券代码

• int64\_t transferable\_quantity

可划转数量

• int64\_t transferred\_quantity

已划转数量

### 5.9.1 详细描述

信用业务余券信息

该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

# 5.10 XTPCombLegStrategy结构体参考

期权组合策略的成分合约信息

#include <xoms\_api\_struct.h>

### 成员变量

• XTP\_OPT\_CALL\_OR\_PUT\_TYPE call\_or\_put

合约类型, 认沽或认购

XTP\_POSITION\_DIRECTION\_TYPE position\_side

权利仓或者义务仓或备兑义务仓

• TXTPExerciseSeqType exercise\_price\_seq

行权价顺序

• int32\_t expire\_date\_seq

到期日顺序

int64\_t leg\_qty

单份组合策略中包含的此合约张数

### 5.10.1 详细描述

期权组合策略的成分合约信息

该结构体的文档由以下文件生成:

• xoms\_api\_struct.h

# 5.11 XTPCrdCashRepayDebtInterestFeeRsp结构体参考

融资融券现金还息费响应信息

#include <xoms\_api\_struct.h>

# 成员变量

int64\_t xtp\_id

直接还款操作的XTPID

· double request\_amount

直接还款的申请金额

• double cash\_repay\_amount

实际还款使用金额

char debt\_compact\_id [XTP\_CREDIT\_DEBT\_ID\_LEN]

指定的负债合约编号

• char unknow [32]

保留字段

### 5.11.1 详细描述

融资融券现金还息费响应信息

该结构体的文档由以下文件生成:

• xoms\_api\_struct.h

# 5.12 XTPCrdCashRepayInfo结构体参考

单条融资融券直接还款记录信息

#include <xoms\_api\_struct.h>

# 成员变量

int64\_t xtp\_id

直接还款操作的XTPID

XTP\_CRD\_CR\_STATUS status

直接还款处理状态

· double request\_amount

直接还款的申请金额

double cash\_repay\_amount

实际还款使用金额

XTP\_POSITION\_EFFECT\_TYPE position\_effect

强平标志
• XTPRI error\_info

直接还款发生错误时的错误信息

## 5.12.1 详细描述

单条融资融券直接还款记录信息

该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

# 5.13 XTPCrdCashRepayRsp结构体参考

融资融券直接还款响应信息

#include <xoms\_api\_struct.h>

### 成员变量

int64\_t xtp\_id

直接还款操作的XTPID

double request\_amount

直接还款的申请金额

double cash\_repay\_amount

实际还款使用金额

### 5.13.1 详细描述

融资融券直接还款响应信息

该结构体的文档由以下文件生成:

# 5.14 XTPCrdDebtInfo结构体参考

单条融资融券负债记录信息

#include <xoms\_api\_struct.h>

# 成员变量

int32\_t debt\_type

负债合约类型: 0为融资, 1为融券, 2未知

• char debt\_id [33]

负债合约编号

int64\_t position\_id

负债对应两融头寸编号

uint64\_t order\_xtp\_id

生成负债的订单编号, 非当日负债无此项

int32\_t debt\_status

负债合约状态: 0为未偿还或部分偿还, 1为已偿还, 2为过期未平仓, 3未知

XTP\_MARKET\_TYPE market

市场

char ticker [XTP\_TICKER\_LEN]

证券代码

· uint64\_t order\_date

委托日期

uint64\_t end\_date

负债截止日期

· uint64\_t orig\_end\_date

负债原始截止日期

· bool is\_extended

当日是否接收到展期请求: false为没收到, true为收到

• double remain\_amt

未偿还金额

int64\_t remain\_qty

未偿还融券数量

double remain\_principal

未偿还本金金额

int64\_t due\_right\_qty

应偿还权益数量

int64\_t unknown [2]

保留字段

### 5.14.1 详细描述

单条融资融券负债记录信息

该结构体的文档由以下文件生成:

# 5.15 XTPCrdDebtStockInfo结构体参考

融资融券指定证券的融券负债相关信息

#include <xoms\_api\_struct.h>

### 成员变量

- char ticker [XTP\_TICKER\_LEN]
   证券代码
- int64\_t stock\_repay\_quantity
   融券负债可还券数量
- int64\_t stock\_total\_quantity 融券负债未还总数量

# 5.15.1 详细描述

融资融券指定证券的融券负债相关信息 该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

# 5.16 XTPCrdFundExtraInfo结构体参考

融资融券帐户附加信息

#include <xoms\_api\_struct.h>

#### 成员变量

- double mf\_rs\_avl\_used 当前资金账户购买货币基金使用的融券卖出所得资金占用
- char reserve [64] 预留空间

融资融券帐户附加信息

5.16.1 详细描述

该结构体的文档由以下文件生成:

# 5.17 XTPCrdFundInfo结构体参考

#### 融资融券特有帐户数据

#include <xoms\_api\_struct.h>

### 成员变量

· double maintenance\_ratio

维持担保品比例

• double all\_asset

总资产(包含证券资产)

double all\_debt

总负债

• double line\_of\_credit

两融授信额度

double guaranty

两融保证金可用数

· double reserved

保留字段

#### 5.17.1 详细描述

融资融券特有帐户数据

该结构体的文档由以下文件生成:

xoms\_api\_struct.h

# 5.18 XTPCrdPositionExtraInfo结构体参考

融资融券帐户持仓附加信息

#include <xoms\_api\_struct.h>

### 成员变量

XTP\_MARKET\_TYPE market

证券市场

• char ticker [XTP\_TICKER\_LEN]

证券代码

• double mf\_rs\_avl\_used

购买货币基金使用的融券卖出所得资金占用

• char reserve [64]

预留空间

### 5.18.1 详细描述

融资融券帐户持仓附加信息

该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

# 5.19 XTPCreditDebtExtendNotice结构体参考

用户展期请求的通知

#include <xoms\_api\_struct.h>

# 成员变量

uint64\_t xtpid

XTP系统订单ID, 无需用户填写, 在XTP系统中唯一

- char debt\_id [XTP\_CREDIT\_DEBT\_ID\_LEN]
   负债合约编号
- XTP\_DEBT\_EXTEND\_OPER\_STATUS oper\_status 展期请求操作状态
- uint64\_t oper\_time
   操作时间

### 5.19.1 详细描述

用户展期请求的通知

该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

# 5.20 XTPCreditDebtExtendReq结构体参考

用户展期请求

#include <xoms\_api\_struct.h>

# 成员变量

uint64\_t xtpid

xtpid

char debt\_id [XTP\_CREDIT\_DEBT\_ID\_LEN]
 负债合约编号

uint32\_t defer\_days

展期天数

char fund\_account [XTP\_ACCOUNT\_NAME\_LEN]
 资金账号

char password [XTP\_ACCOUNT\_PASSWORD\_LEN]
 资金账号密码

### 5.20.1 详细描述

用户展期请求

该结构体的文档由以下文件生成:

• xoms\_api\_struct.h

# 5.21 XTPFundQueryReq结构体参考

用户资金查询请求结构体

#include <xoms\_api\_fund\_struct.h>

### 成员变量

- 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
- XTP\_FUND\_QUERY\_TYPE query\_type 查询类型
- uint64\_t unknown [4]
   预留字段,用户无需填写

### 5.21.1 详细描述

用户资金查询请求结构体

该结构体的文档由以下文件生成:

xoms\_api\_fund\_struct.h

# 5.22 XTPFundQueryRsp结构体参考

用户资金查询响应结构体

#include <xoms\_api\_fund\_struct.h>

# 成员变量

· double amount

金额

 XTP\_FUND\_QUERY\_TYPE query\_type 查询类型

uint64\_t unknown [4]
 预留字段,用户无需填写

#### 5.22.1 详细描述

用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\_fund\_struct.h

# 5.23 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

全新

XTP\_FUND\_OPER\_STATUS oper\_status

操作结果

• uint64\_t transfer\_time

操作时间

### 5.23.1 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

# 5.24 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

### 5.24.1 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

# 5.25 XTPMarketDataBondExData结构体参考

债券额外数据

#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 match\_lastpx

匹配成交最近价(SZ)

double ma\_bond\_price

债券加权平均价格(SH)

double r2

预留

• double r3

预留

double r4

预留

• double r5

预留

• double r6

预留

double r7

预留

• double r8

预留

• 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)

• char instrument\_status [8]

时段(SHL2), L1快照数据没有此字段, 具体字段说明参阅《上海新债券Level2行情说明.doc》文档

#### 5.25.1 详细描述

债券额外数据

该结构体的文档由以下文件生成:

xquote\_api\_struct.h

# 5.26 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)

#### 5.26.1 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

# 5.27 XTPMarketDataStockExData结构体参考

股票、基金 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

### 5.27.1 详细描述

股票、基金 等额外数据

该结构体的文档由以下文件生成:

• xquote\_api\_struct.h

# 5.28 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10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 (SH)
• double settl_price
    今日结算价 (SH)
• 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
    预留(无意义)
• 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]
```

```
union {
    XTPMarketDataStockExData stk
    XTPMarketDataOptionExData opt
    XTPMarketDataBondExData bond
};
```

当前交易状态说明,参阅《XTP API常见问题.doc》文档

数据

XTP\_MARKETDATA\_TYPE data\_type

决定了union是哪种数据类型 (2.2.32版本以前所用字段,仅为了保持兼容,不建议使用该字段)

XTP\_MARKETDATA\_TYPE\_V2 data\_type\_v2

决定了union是哪种数据类型(2.2.32版本新增字段,更详细区分了行情快照数据类型)

### 5.28.1 详细描述

行情

该结构体的文档由以下文件生成:

xquote\_api\_struct.h

# 5.29 XTPOptCombLegInfo结构体参考

组合策略腿合约信息结构体

#include <xoms\_api\_struct.h>

# 成员变量

- char leg\_security\_id [XTP\_TICKER\_LEN]
   成分合约代码
- XTP\_OPT\_CALL\_OR\_PUT\_TYPE leg\_cntr\_type 合约类型,认沽或认购。
- XTP\_POSITION\_DIRECTION\_TYPE leg\_side
   特仓方向,权利方或义务方。
- XTP\_OPT\_COVERED\_OR\_UNCOVERED leg\_covered 备兑标签
- int32\_t leg\_qty 成分合约数量(张)

### 5.29.1 详细描述

组合策略腿合约信息结构体

该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

# 5.30 XTPOptCombOrderInfo结构体参考

期权组合策略报单响应结构体

#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

- XTP\_MARKET\_TYPE market
- int64\_t quantity

数量,此订单的报单数量

· XTP SIDE TYPE side

组合方向

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 tinsert 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,组合拆分涉及的保证金(保留字段)

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,用户可用此字段来区分撤单和报单

• TXTPOrderTypeType order\_type

报单类型

• XTPOptCombPlugin opt\_comb\_info

期权组合策略信息

#### 5.30.1 详细描述

期权组合策略报单响应结构体

#### 5.30.2 结构体成员变量说明

#### 5.30.2.1 market

XTP\_MARKET\_TYPE market

证券代码 char ticker[XTP\_TICKER\_LEN]; 交易市场

该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

# 5.31 XTPOptCombOrderInfoEx结构体参考

期权组合策略报单响应结构体,新版本

#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

- XTP\_MARKET\_TYPE market
- int64\_t quantity

数量,此订单的报单数量

• XTP\_SIDE\_TYPE side

组合方向

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,组合拆分涉及的保证金

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

· XTPOptCombPlugin opt comb info

期权组合策略信息

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

XTPRI order\_err\_t

订单的错误信息

• uint64\_t unknown [2]

保留字段

### 5.31.1 详细描述

期权组合策略报单响应结构体, 新版本

#### 5.31.2 结构体成员变量说明

#### 5.31.2.1 market

XTP\_MARKET\_TYPE market

证券代码 char ticker[XTP\_TICKER\_LEN]; 交易市场 该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

# 5.32 XTPOptCombOrderInsertInfo结构体参考

期权组合策略新订单请求

#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

XTP\_MARKET\_TYPE market

交易市场

int64 t quantity

数量(单位为份)

• XTP\_SIDE\_TYPE side

组合方向

• XTP\_BUSINESS\_TYPE business\_type

业务类型

XTPOptCombPlugin opt\_comb\_info

期权组合策略信息

# 5.32.1 详细描述

期权组合策略新订单请求

该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

# 5.33 XTPOptCombPlugin结构体参考

期权组合策略报单附加信息结构体

#include <xoms\_api\_struct.h>

# 成员变量

- char strategy\_id [XTP\_STRATEGY\_ID\_LEN] 组合策略代码,比如*CNSJC*认购牛市价差策略等。合并行权时,此字段可为空
- char comb\_num [XTP\_SECONDARY\_ORDER\_ID\_LEN] 组合编码,组合申报时,该字段为空;拆分申报时,填写拟拆分组合的组合编码。
- int32\_t num\_legs 成分合约数
- XTPOptCombLegInfo leg\_detail [XTP\_STRATEGE\_LEG\_NUM] 成分合约数组、最多四条限。

#### 5.33.1 详细描述

期权组合策略报单附加信息结构体

该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

# 5.34 XTPOptCombTradeReport结构体参考

期权组合策略报单成交结构体

#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

· 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

· 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,组合拆分涉及的保证金

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(保留字段)

• TXTPTradeTypeType trade\_type

成交类型 -成交回报中的执行类型

• XTP\_SIDE\_TYPE side

组合方向

• XTP\_BUSINESS\_TYPE business\_type

业务类型

• char branch\_pbu [XTP\_BRANCH\_PBU\_LEN]

交易所交易员代码

XTPOptCombPlugin opt\_comb\_info

期权组合策略信息

#### 5.34.1 详细描述

期权组合策略报单成交结构体

该结构体的文档由以下文件生成:

xoms\_api\_struct.h

# 5.35 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

### 5.35.1 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

# 5.36 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
    32位字段,用来兼容老版本api,用户无需关心
    XTP SIDE TYPE side
      买卖方向
    XTP_POSITION_EFFECT_TYPE position_effect
      开平标志,期权用户关注字段,其余用户填0即可
    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,用户可用此字段来区分撤单和报单

    TXTPOrderTypeType order_type

   报单类型
```

#### 5.36.1 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

# 5.37 XTPOrderInfoEx结构体参考

```
报单响应结构体, 新版本
```

#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

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

XTPRI order\_err\_t

订单的错误信息

• uint64\_t unknown [2]

保留字段

#### 5.37.1 详细描述

报单响应结构体, 新版本

该结构体的文档由以下文件生成:

· xoms api struct.h

# 5.38 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10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

.

5.38.1 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

# 5.39 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(现货账户/期权账户参考公式:总资产 = 可用资金 + 证券资产(目前为0) + 预扣的资金),(信用账户参考公式:总资产 = 可用资金 + 融券卖出所得资金余额 + 证券资产 + 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(仅限现货账户/期权账户,信用账户暂不可用)

· double fund buy fee

累计买入成交交易费用(仅限现货账户/期权账户,信用账户暂不可用)

double fund\_sell\_amount

累计卖出成交证券所得资金(仅限现货账户/期权账户,信用账户暂不可用)

• double fund\_sell\_fee

累计卖出成交交易费用(仅限现货账户/期权账户,信用账户暂不可用)

double withholding\_amount

XTP系统预扣的资金(包括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 (仅限期权账户)

double frozen\_exec\_cash

行权冻结资金 (仅限期权账户)

· double frozen exec fee

行权费用(仅限期权账户)

double pay\_later

垫付资金(仅限期权账户)

double preadva\_pay

预垫付资金(仅限期权账户)

· double orig\_banlance

昨日余额(仅限期权账户)

· double banlance

当前余额(仅限期权账户)

double deposit\_withdraw

当天出入金 (仅限期权账户)

double trade\_netting

当日交易资金轧差 (仅限期权账户)

· double captial asset

资金资产 (仅限期权账户)

· double force\_freeze\_amount

强锁资金 (仅限期权账户)

· double preferred\_amount

可取资金 (仅限期权账户)

double repay\_stock\_aval\_banlance

融券卖出所得资金余额(仅限信用账户,只能用于买券还券)

· double fund\_order\_data\_charges

累计订单流量费

double fund\_cancel\_data\_charges

累计撤单流量费

• double exchange\_cur\_risk\_degree

交易所实时风险度(仅限期权账户,后续服务器版本支持,目前为0)

· double company\_cur\_risk\_degree

公司实时风险度(仅限期权账户,后续服务器版本支持,目前为0)

• uint64\_t unknown [43 - 12 - 1 - 2 - 2] (保留字段)

,

#### 5.39.1 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:

# 5.40 XTPQueryCombineStrategyInfoRsp结构体参考

查询期权组合策略信息的响应

#include <xoms\_api\_struct.h>

#### 成员变量

• char strategy\_id [XTP\_STRATEGY\_ID\_LEN] 组合策略代码,*CNSJC、PXSJC、PNSJC、CXSJC、KS、KKS* 

• char strategy\_name [XTP\_STRATEGY\_NAME\_LEN]

组合策略名称,认购牛市价差策略、认沽熊市价差策略、认沽牛市价差策略、认购熊市价差策略、跨式空头、宽跨式空头

XTP\_MARKET\_TYPE market

交易市场

• int32\_t leg\_num

成分合约个数, 1-4个, 即下面数组的实际大小

XTPCombLegStrategy leg\_strategy [XTP\_STRATEGE\_LEG\_NUM]

成分合约信息,最多四条腿

XTP\_EXPIRE\_DATE\_TYPE expire\_date\_type

到期日要求。枚举值为: 同到期日, 不同到期日, 无到期日要求

XTP\_UNDERLYING\_TYPE underlying\_type

标的要求。枚举值为:相同标的,不同标的,无标的要求

XTP\_AUTO\_SPLIT\_TYPE auto\_sep\_type

自动解除类型。枚举值为:-1:不适用; 0:到期日自动解除; 1: E-1日自动解除,依次类推

uint64 t reserved [10]

预留的字段

#### 5.40.1 详细描述

查询期权组合策略信息的响应

该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

# 5.41 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

#### 5.41.1 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

# 5.42 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

7日预估金额差额

· 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

#### 5.42.1 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:

# 5.43 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

#### 5.43.1 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

# 5.44 XTPQueryETFComponentRsp结构体参考

查询股票ETF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

• double creation\_premium\_ratio

申购溢价比例

· double redemption\_discount\_ratio

赎回溢价比例

· double creation amount

申购时,成分股替代标识为必须现金替代时候的总金额

· double redemption\_amount

赎回时,成分股替代标识为必须现金替代时候的总金额

### 5.44.1 详细描述

查询股票ETF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

## 5.45 XTPQueryETFComponentRspV1结构体参考

查询股票ETF成分股信息-响应结构体,旧版本。

#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

### 5.45.1 详细描述

查询股票ETF成分股信息-响应结构体, 旧版本。

该结构体的文档由以下文件生成:

xoms\_api\_struct.h

## 5.46 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

### 5.46.1 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

## 5.47 XTPQueryIPOQuotaRsp结构体参考

查询用户申购额度-包含创业板额度

#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
- int32\_t tech\_quantity
   上海科创板额度
- int32\_t unused 保留

### 5.47.1 详细描述

查询用户申购额度-包含创业板额度

该结构体的文档由以下文件生成:

• xoms\_api\_struct.h

## 5.48 XTPQueryIPOQuotaRspV1结构体参考

查询用户申购额度-旧版

#include <xoms\_api\_struct.h>

XTP\_MARKET\_TYPE market

交易市场

int32\_t quantity

可申购额度

### 5.48.1 详细描述

查询用户申购额度-旧版

该结构体的文档由以下文件生成:

• xoms\_api\_struct.h

## 5.49 XTPQuerylPOTickerRsp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

XTP\_TICKER\_TYPE ticker\_type

证券类别

· 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

### 5.49.1 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

## 5.50 XTPQueryOptCombExecPosReq结构体参考

查询期权行权合并头寸请求结构体

#include <xoms\_api\_struct.h>

### 成员变量

• XTP\_MARKET\_TYPE market

市场

 char cntrt\_code\_1 [XTP\_TICKER\_LEN] 成分合约1代码

 char cntrt\_code\_2 [XTP\_TICKER\_LEN] 成分合约2代码

### 5.50.1 详细描述

查询期权行权合并头寸请求结构体

该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

### 5.51 XTPQueryOptCombExecPosRsp结构体参考

查询期权行权合并头寸的响应

#include <xoms\_api\_struct.h>

### 成员变量

XTP\_MARKET\_TYPE market

市场

char cntrt\_code\_1 [XTP\_TICKER\_LEN]

成分合约1代码
• char cntrt\_name\_1 [XTP\_TICKER\_NAME\_LEN]

成分合约1名称
• XTP\_POSITION\_DIRECTION\_TYPE position\_side\_1

成分合约1持仓方向
• XTP\_OPT\_CALL\_OR\_PUT\_TYPE call\_or\_put\_1

成分合约1类型

int64\_t avl\_qty\_1

成分合约1可用持仓数量

• int64\_t orig\_own\_qty\_1 成分合约1昨日持仓数量

int64\_t own\_qty\_1

成分合约1当前持仓数量

char cntrt\_code\_2 [XTP\_TICKER\_LEN]

成分合约2代码

char cntrt\_name\_2 [XTP\_TICKER\_NAME\_LEN]

成分合约2名称

XTP\_POSITION\_DIRECTION\_TYPE position\_side\_2

成分合约2持仓方向

XTP\_OPT\_CALL\_OR\_PUT\_TYPE call\_or\_put\_2

成分合约2类型

int64\_t avl\_qty\_2

成分合约2可用持仓数量

• int64\_t orig\_own\_qty\_2

成分合约2昨日持仓数量

int64\_t own\_qty\_2

成分合约2当前持仓数量

int64\_t net\_qty

权利仓净头寸

int64\_t order\_qty

行权合并委托数量,不含已拒单已撤单。

int64\_t confirm\_qty

行权合并已确认数量

int64 t avl qty

可行权合并数量

• uint64\_t reserved [49]

保留字段

### 5.51.1 详细描述

查询期权行权合并头寸的响应

该结构体的文档由以下文件生成:

· xoms api struct.h

## 5.52 XTPQueryOptCombOrderByPageReq结构体参考

查询期权组合策略订单请求-分页查询

#include <xoms\_api\_struct.h>

### 成员变量

int64\_t req\_count

需要查询的订单条数

· int64 t reference

上一次收到的查询订单结果中带回来的索引,如果是从头查询,请置0

· int64\_t reserved

保留字段

### 5.52.1 详细描述

查询期权组合策略订单请求-分页查询

该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

## 5.53 XTPQueryOptCombOrderReq结构体参考

期权组合策略报单查询 /////////////// 期权组合策略报单查询请求-条件查询 #include <xoms\_api\_struct.h>

### 成员变量

- char comb\_num [XTP\_SECONDARY\_ORDER\_ID\_LEN] 组合编码(流水号),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

### 5.53.1 详细描述

期权组合策略报单查询 /////////////// 期权组合策略报单查询请求-条件查询该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

### 5.54 XTPQueryOptCombPositionReq结构体参考

查询期权组合策略持仓情况请求结构体

#include <xoms\_api\_struct.h>

### 成员变量

- char comb\_num [XTP\_SECONDARY\_ORDER\_ID\_LEN]
   组合编码
- XTP\_MARKET\_TYPE market 交易市场

### 5.54.1 详细描述

查询期权组合策略持仓情况请求结构体

该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

### 5.55 XTPQueryOptCombPositionRsp结构体参考

查询期权组合策略持仓信息的响应

#include <xoms\_api\_struct.h>

### 成员变量

- char strategy\_id [XTP\_STRATEGY\_ID\_LEN]
   组合策略代码
- char strategy\_name [XTP\_STRATEGY\_NAME\_LEN]
   组合策略名称
- XTP\_MARKET\_TYPE market

交易市场

int64\_t total\_qty

总持仓

- int64\_t available\_qty
  - 可拆分持仓
- int64\_t yesterday\_position

昨日持仓

- · XTPOptCombPlugin opt comb info
  - 期权组合策略信息
- uint64\_t reserved [50]

保留字段

### 5.55.1 详细描述

查询期权组合策略持仓信息的响应

该结构体的文档由以下文件生成:

xoms\_api\_struct.h

## 5.56 XTPQueryOptCombReportByExecIdReq结构体参考

期权组合策略成交回报查询 //////////////////// 查询期权组合策略成交报告请求-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

### 5.56.1 详细描述

期权组合策略成交回报查询 /////////////////////// 查询期权组合策略成交报告请求-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

## 5.57 XTPQueryOptCombTraderByPageReq结构体参考

查询期权组合策略成交回报请求-分页查询

#include <xoms\_api\_struct.h>

### 成员变量

- int64\_t req\_count 需要查询的成交回报条数
- int64\_t reference 上一次收到的查询成交回报结果中带回来的索引,如果是从头查询,请置*0*
- int64\_t reserved 保留字段

### 5.57.1 详细描述

查询期权组合策略成交回报请求-分页查询

该结构体的文档由以下文件生成:

· xoms api struct.h

## 5.58 XTPQueryOptCombTraderReq结构体参考

查询期权组合策略成交回报请求-查询条件

#include <xoms\_api\_struct.h>

• char comb\_num [XTP\_SECONDARY\_ORDER\_ID\_LEN]

组合编码(流水号),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

#### 5.58.1 详细描述

查询期权组合策略成交回报请求-查询条件

该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

### 5.59 XTPQueryOptExecInfoRsp结构体参考

查询期权合约行权信息的响应

#include <xoms\_api\_struct.h>

### 成员变量

XTP\_MARKET\_TYPE market

市场

char cntrt\_code [XTP\_TICKER\_LEN]

合约代码

• int64\_t own\_qty\_long

权利仓数量

int64\_t own\_qty\_short

义务仓数量

int64\_t own\_qty\_short\_cover

备兑义务仓数量

int64\_t net\_qty

净头寸

int64\_t combed\_qty\_long

权利仓已组合数量

• int64\_t combed\_qty\_short

义务仓已组合数量

• int64 t combed qty short cover

备兑义务仓已组合数量

int64\_t total\_execute\_gene\_order\_qty

累计普通行权委托数量

int64\_t total\_execute\_gene\_confirm\_qty

累计普通行权确认数量

int64\_t total\_execute\_comb\_order\_qty

累计行权合并委托数量

• int64\_t total\_execute\_comb\_confirm\_qty

累计行权合并确认数量

uint64\_t reserved [50]

保留字段

### 5.59.1 详细描述

查询期权合约行权信息的响应

该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

## 5.60 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

### 5.60.1 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

## 5.61 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>

• 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)

### 5.61.1 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

### 5.62 XTPQueryOrderByPageReq结构体参考

查询订单请求-分页查询

#include <xoms\_api\_struct.h>

### 成员变量

• int64\_t req\_count 需要查询的订单条数

· int64 t reference

上一次收到的查询订单结果中带回来的索引,如果是从头查询,请置0

int64\_t reserved 保留字段

### 5.62.1 详细描述

查询订单请求-分页查询

该结构体的文档由以下文件生成:

· xoms\_api\_struct.h

## 5.63 XTPQueryOrderReg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

### 5.63.1 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

### 5.64 XTPQueryReportByExecldReq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

### 5.64.1 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

## 5.65 XTPQueryStkPositionReq结构体参考

查询股票持仓情况请求结构体

#include <xoms\_api\_struct.h>

### 成员变量

- char ticker [XTP\_TICKER\_LEN]
   证券代码
- XTP\_MARKET\_TYPE market 交易市场

### 5.65.1 详细描述

查询股票持仓情况请求结构体

该结构体的文档由以下文件生成:

• xoms\_api\_struct.h

## 5.66 XTPQueryStkPositionRsp结构体参考

查询股票持仓情况

#include <xoms\_api\_struct.h>

• 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

XTP\_POSITION\_SECURITY\_TYPE position\_security\_type

持仓类型/此字段所有账户都可能用到,可以用来区分股份是否为配售)

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

double profit\_price

盈亏成本价

double buy\_cost

买入成本

double profit\_cost

盈亏成本

• double market\_value

持仓市值(此字段目前只有期权账户有值,其他类型账户为0)

• uint64\_t unknown [50 - 10]

(保留字段)

### 5.66.1 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

## 5.67 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

### 5.67.1 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

## 5.68 XTPQueryTraderByPageReq结构体参考

查询成交回报请求-分页查询

#include <xoms\_api\_struct.h>

### 成员变量

- int64\_t req\_count 需要查询的成交回报条数
- · int64 t reference

上一次收到的查询成交回报结果中带回来的索引,如果是从头查询,请置0

int64\_t reserved 保留字段

### 5.68.1 详细描述

查询成交回报请求-分页查询

该结构体的文档由以下文件生成:

• xoms\_api\_struct.h

### 5.69 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

### 5.69.1 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

### 5.70 XTPQuoteFullInfo结构体参考

股票行情全量静态信息

#include <xquote\_api\_struct.h>

### 成员变量

• XTP\_EXCHANGE\_TYPE exchange\_id

交易所代码

char ticker [XTP\_TICKER\_LEN]

证券代码

• char ticker\_name [XTP\_TICKER\_NAME\_LEN]

证券名称

XTP\_SECURITY\_TYPE security\_type

合约详细类型

• XTP\_QUALIFICATION\_TYPE ticker\_qualification\_class

合约适当性类别

bool is\_registration

是否注册制(仅适用创业板股票,创新企业股票及存托凭证)

• bool is VIE

是否具有协议控制架构(仅适用创业板股票,创新企业股票及存托凭证)

bool is\_noprofit

是否尚未盈利/仅适用创业板股票,创新企业股票及存托凭证)

• bool is\_weighted\_voting\_rights

是否存在投票权差异(仅适用创业板股票,创新企业股票及存托凭证)

• bool is\_have\_price\_limit

是否有涨跌幅限制(注:不提供具体幅度,可通过涨跌停价和昨收价来计算幅度)

double upper\_limit\_price

涨停价(仅在有涨跌幅限制时有效)

· double lower\_limit\_price

跌停价(仅在有涨跌幅限制时有效)

double pre\_close\_price

昨收价

· double price tick

价格最小变动价位

• int32\_t bid\_qty\_upper\_limit

限价买委托数量上限

· int32\_t bid\_qty\_lower\_limit

限价买委托数量下限

int32\_t bid\_qty\_unit

限价买数量单位

int32\_t ask\_qty\_upper\_limit

限价卖委托数量上限

int32\_t ask\_qty\_lower\_limit

限价卖委托数量下限

· int32\_t ask\_qty\_unit

限价卖数量单位

• int32\_t market\_bid\_qty\_upper\_limit

市价买委托数量上限

• int32\_t market\_bid\_qty\_lower\_limit

市价买委托数量下限

• int32\_t market\_bid\_qty\_unit

市价买数量单位

int32\_t market\_ask\_qty\_upper\_limit

市价卖委托数量上限

· int32\_t market\_ask\_qty\_lower\_limit

市价卖委托数量上限

int32\_t market\_ask\_qty\_unit

市价卖数量单位

• XTP\_SECURITY\_STATUS security\_status

证券状态

• uint32 t unknown1

保留字段

• uint64\_t unknown [3]

保留字段

### 5.70.1 详细描述

股票行情全量静态信息

该结构体的文档由以下文件生成:

xquote\_api\_struct.h

### 5.71 XTPQuoteRebuildReq结构体参考

实时行情回补查询

#include <xquote\_api\_rebuild\_tbt\_struct.h>

### 成员变量

· int32\_t request\_id

请求id 请求端自行管理定义

XTP\_QUOTE\_REBUILD\_DATA\_TYPE data\_type

请求数据类型 1-快照 2-逐笔

XTP\_EXCHANGE\_TYPE exchange\_id

请求市场 1-上海 2-深圳

• char ticker [16]

合约代码以10′结尾沪深A股6位期权8位

· int16\_t channel\_number

data\_type=逐笔 表示逐笔通道号

int64 t begin

data\_type=逐笔表示序列号begin; =快照表示时间begin(格式为YYYYMMDDHHMMSSsss)

• int64\_t end

data\_type=逐笔 表示序列号end; =快照 表示时间end(格式为YYYYMMDDHHMMSSsss) 逐笔区间: [begin, end]前后闭区间 快照区间: [begin, end) 前闭后开区间

### 5.71.1 详细描述

实时行情回补查询

实时行情回补请求结构体

该结构体的文档由以下文件生成:

• xquote\_api\_rebuild\_tbt\_struct.h

## 5.72 XTPQuoteRebuildResultRsp结构体参考

实时行情回补响应结构体

#include <xquote\_api\_rebuild\_tbt\_struct.h>

### 成员变量

· int32 t request id

请求id 请求包带过来的id

XTP\_EXCHANGE\_TYPE exchange\_id

市场类型 上海 深圳

uint32\_t size

总共返回的数据条数

· int16 t channel number

逐笔数据 通道号

• int64\_t begin

逐笔表示序列号begin;快照表示时间begin(格式为YYYYMMDDHHMMSSsss)

· int64 t end

逐笔表示序列号end;快照表示时间end(格式为YYYYMMDDHHMMSSsss)

XTP\_REBUILD\_RET\_TYPE result\_code

结果类型码

• char msg [64]

结果信息文本

### 5.72.1 详细描述

实时行情回补响应结构体 该结构体的文档由以下文件生成:

• xquote\_api\_rebuild\_tbt\_struct.h

### 5.73 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10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)

### 5.73.1 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

## 5.74 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

### 5.74.1 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

## 5.75 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

### 5.75.1 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

## 5.76 XTPStrategyInfoStruct结构体参考

策略信息结构体

#include <algo\_api\_struct.h>

### 成员变量

uint16\_t m\_strategy\_type
 策略类型

• XTPStrategyStateType m\_strategy\_state 策略状态

• uint64\_t m\_client\_strategy\_id 客户策略*id* 

uint64\_t m\_xtp\_strategy\_id
 xtp策略id

### 5.76.1 详细描述

策略信息结构体

该结构体的文档由以下文件生成:

· algo\_api\_struct.h

## 5.77 XTPStrategyStateReportStruct结构体参考

策略状态结构体

#include <algo\_api\_struct.h>

XTPStrategyInfoStruct m\_strategy\_info

策略信息

• int64\_t m\_strategy\_qty

策略总量

int64\_t m\_strategy\_ordered\_qty

策略已委托数量

• int64\_t m\_strategy\_cancelled\_qty

策略已撤单数量

int64\_t m\_strategy\_execution\_qty

策略已成交数量

int64\_t m\_strategy\_unclosed\_qty

策略未平仓数量(T0卖出数量-买入数量)

• double m\_strategy\_asset

策略总金额

• double m\_strategy\_ordered\_asset

策略已委托金额

double m\_strategy\_execution\_asset

策略已成交金额

• double m\_strategy\_execution\_price

策略执行价格

• double m\_strategy\_market\_price

策略市场价

• double m\_strategy\_price\_diff

策略执行价差

• double m\_strategy\_asset\_diff

策略执行绩效(T0资金预净收入)

• XTPRI m\_error\_info

错误信息

### 5.77.1 详细描述

策略状态结构体

该结构体的文档由以下文件生成:

• algo\_api\_struct.h

## 5.78 XTPStrategySymbolInfoStruct结构体参考

策略中指定证券信息结构体

#include <algo\_api\_struct.h>

### 成员变量

 XTPStrategyInfoStruct m\_strategy\_info 策略信息

• char m\_ticker [XTP\_TICKER\_LEN] 证券代码

• XTP\_MARKET\_TYPE m\_market 市场

### 5.78.1 详细描述

策略中指定证券信息结构体

该结构体的文档由以下文件生成:

· algo\_api\_struct.h

### 5.79 XTPStrategySymbolRegStruct结构体参考

指定策略指定证券的请求结构体

#include <algo\_api\_struct.h>

### 成员变量

- uint64\_t m\_xtp\_strategy\_id xtp策略id
- char m\_ticker [XTP\_TICKER\_LEN] 证券代码
- XTP\_MARKET\_TYPE m\_market 市场

### 5.79.1 详细描述

指定策略指定证券的请求结构体

该结构体的文档由以下文件生成:

· algo\_api\_struct.h

## 5.80 XTPStrategySymbolStateReportStruct结构体参考

策略中指定证券的算法执行状态结构体

#include <algo\_api\_struct.h>

XTPStrategyInfoStruct m\_strategy\_info

策略信息

char m\_ticker [XTP\_TICKER\_LEN]

证券代码

XTP\_MARKET\_TYPE m\_market

市场

• XTP\_SIDE\_TYPE m\_side

买卖方向, =0时为T0单

• int64\_t m\_strategy\_qty

策略总量

• int64\_t m\_strategy\_ordered\_qty

策略已委托数量

int64\_t m\_strategy\_cancelled\_qty

策略已撤单数量

int64\_t m\_strategy\_execution\_qty

策略已成交数量

int64\_t m\_strategy\_buy\_qty

策略已买入数量(TO)

int64\_t m\_strategy\_sell\_qty

策略已卖出数量(T0)

int64\_t m\_strategy\_unclosed\_qty

策略未平仓数量(T0卖出数量-买入数量)

double m\_strategy\_asset

策略总金额

• double m\_strategy\_ordered\_asset

策略已委托金额

• double m\_strategy\_execution\_asset

策略已成交金额

double m\_strategy\_buy\_asset

策略买入金额(T0)

double m\_strategy\_sell\_asset

策略卖出金额(TO)

• double m\_strategy\_unclosed\_asset

策略未平仓金额(T0)

• double m\_strategy\_asset\_diff

策略毛收益增强金额(T0)

• double m\_strategy\_execution\_price

策略执行价格

• double m\_strategy\_market\_price

策略市场价

• double m\_strategy\_price\_diff

策略执行价差(TO时为毛增强收益率)

• XTPRI m\_error\_info

错误信息

### 5.80.1 详细描述

策略中指定证券的算法执行状态结构体

该结构体的文档由以下文件生成:

· algo api struct.h

### 5.81 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

### 5.81.1 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

## 5.82 XTPTickByTickEntrust结构体参考

### 逐笔委托

#include <xquote\_api\_struct.h>

### 成员变量

- int32\_t channel\_no
  - 频道代码
- int64\_t seq
- · double price
  - 委托价格
- int64\_t qty
- char side
- char ord\_type
- int64\_t order\_no

### 5.82.1 详细描述

逐笔委托

5.82.2 结构体成员变量说明

5.82.2.1 ord\_type

char ord\_type

SH: 'A': 增加; 'D': 删除 SZ: 订单类别: '1': 市价; '2': 限价; 'U': 本方最优

5.82.2.2 order\_no

int64\_t order\_no

SH: 原始订单号 SZ: 无意义

5.82.2.3 qty

int64\_t qty

SH: 剩余委托数量(balance) SZ: 委托数量

#### 5.82.2.4 seq

int64\_t seq

SH: 委托序号(委托单独编号, 同一channel\_no内连续) SZ: 委托序号(委托成交统一编号, 同一channel\_no内连续)

#### 5.82.2.5 side

char side

SH: 'B':买; 'S':卖 SZ: '1':买; '2':卖; 'G':借入; 'F':出借

该结构体的文档由以下文件生成:

• xquote\_api\_struct.h

## 5.83 XTPTickByTickStatus结构体参考

#### 逐笔状态订单

#include <xquote\_api\_struct.h>

### 成员变量

- int32\_t channel\_no
   频道代码
- int64\_t seq

同一channel\_no内连续

• char flag [8] 状态信息

### 5.83.1 详细描述

逐笔状态订单

该结构体的文档由以下文件生成:

xquote\_api\_struct.h

## 5.84 XTPTickByTickStruct结构体参考

### 逐笔数据信息

#include <xquote\_api\_struct.h>

### 5.84.1 详细描述

逐笔数据信息

### 5.84.2 结构体成员变量说明

```
5.84.2.1 seq
```

```
int64_t seq
```

SH: 业务序号(委托成交统一编号,同一个channel\_no内连续,此seq区别于联合体内的seq,channel\_↔ no等同于联合体内的channel no) SZ: 无意义

该结构体的文档由以下文件生成:

• xquote\_api\_struct.h

## 5.85 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
- · 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

### 5.85.1 详细描述

逐笔成交

### 5.85.2 结构体成员变量说明

5.85.2.1 seq

int64\_t seq

SH: 成交序号(成交单独编号, 同一channel\_no内连续) SZ: 成交序号(委托成交统一编号, 同一channel\_no内连续)

5.85.2.2 trade\_flag

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

## 5.86 XTPTickerPriceInfo结构体参考

供查询的最新信息

#include <xquote\_api\_struct.h>

• 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10结尾

· double last price

最新价

### 5.86.1 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

### 5.87 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,对于单个账户来说,深交所每个平台(不同交易品种)唯一,上交所唯一,对于 多账户来说,不唯一

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
        32位字段,用来兼容老版本api,用户无需关心
    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

### 5.87.1 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

## 5.88 XTPUserTerminalInfoReq结构体参考

```
申报用户的ip和mac等信息,仅限授权用户使用
```

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

### 成员变量

- char local\_ip [XTP\_INET\_ADDRESS\_STR\_LEN] 本地*IP*地址
- char mac\_addr [XTP\_MAC\_ADDRESS\_LEN]

  MAC地址
- char hd [XTP\_HARDDISK\_SN\_LEN]
   硬盘序列号

• XTPTerminalType term\_type

终端类型

• char internet\_ip [XTP\_INET\_ADDRESS\_STR\_LEN]

公网IP地址

int32\_t internet\_port

公网端口号

• XTPVersionType client\_version

客户端版本号

• char macos\_sno [XTP\_MACOS\_SNO\_LEN]

MacOS系统的序列号,仅为MacOS系统需要填写

• char unused [27]

预留

### 5.88.1 详细描述

申报用户的ip和mac等信息,仅限授权用户使用

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 algo\_api\_struct.h 文件参考

定义业务公共数据结构

```
#include "algo_data_type.h"
#include "xtp_api_struct_common.h"
```

### 结构体

- struct XTPStrategyInfoStruct 策略信息结构体
- struct XTPStrategySymbolInfoStruct 策略中指定证券信息结构体
- struct XTPStrategyStateReportStruct 策略状态结构体
- struct XTPStrategySymbolReqStruct
- 指定策略指定证券的请求结构体
   struct XTPStrategySymbolStateReportStruct

策略中指定证券的算法执行状态结构体

### 类型定义

- typedef struct XTPStrategyInfoStruct XTPStrategyInfoStruct 策略信息结构体
- typedef struct XTPStrategySymbolInfoStruct XTPStrategySymbolInfo 策略中指定证券信息结构体
- typedef struct XTPStrategyStateReportStruct XTPStrategyStateReport 策略状态结构体
- typedef struct XTPStrategySymbolReqStruct XTPStrategySymbolReq 指定策略指定证券的请求结构体
- typedef struct XTPStrategySymbolStateReportStruct XTPStrategySymbolStateReport 策略中指定证券的算法执行状态结构体

文件说明

### 6.1.1 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

### 6.2 algo\_data\_type.h 文件参考

定义业务公共数据结构

### 宏定义

- #define XTP\_STRATEGY\_STATE\_CREATING 0 创建中
- #define XTP\_STRATEGY\_STATE\_CREATED 1 已创建
- #define XTP\_STRATEGY\_STATE\_STARTING 2
   开始执行中
- #define XTP\_STRATEGY\_STATE\_STARTED 3
   已执行
- #define XTP\_STRATEGY\_STATE\_STOPPING 4 停止中
- #define XTP\_STRATEGY\_STATE\_STOPPED 5 已停止
- #define XTP\_STRATEGY\_STATE\_DESTROYING 6
   销毁中
- #define XTP\_STRATEGY\_STATE\_DESTROYED 7 已销毁
- #define XTP\_STRATEGY\_STATE\_ERROR 8 发生错误

### 类型定义

typedef uint8\_t XTPStrategyStateType
 XTPStrategyStateType策略状态类型

### 6.2.1 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

# 6.3 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 "demo_test_trade_spi.h"
```

### 函数

• int main ()

//API的全局变量指针 XTP::API::TraderApi\* user\_api\_pointer;

#### 变量

XTP::API::TraderApi \* user\_api\_pointer
 API的全局变量指针

#### 6.3.1 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

#### 6.3.2 函数说明

```
6.3.2.1 main()
```

```
int main ( )
```

//API的全局变量指针 XTP::API::TraderApi\* user\_api\_pointer;

```
测试Demo入口函数 int main() {
    int client_id = 1;//客户端标识    char filepath[] = "c:\\log\\";//真实存在的可读写路径    //初始化UserApi    user_api_pointer = XTP::API::TraderApi::CreateTraderApi(client_id, filepath, XTP_LOG_LEVEL_DEBUG); // 创
```

```
建UserApi
user_api_pointer->SubscribePublicTopic(XTP_TERT_QUICK);//设定公共流传输方式
予,以'\0'结尾
user api pointer->SetSoftwareVersion("1.1.0");//设定软件的开发版本号,非api版本号
user api pointer->SetHeartBeatInterval(15);//设置心跳超时时间间隔,单位为秒
DemoTestTraderSpi* user spi pointer = new DemoTestTraderSpi(); // 创建响应类实例
user api pointer->RegisterSpi(user spi pointer); // 注册响应事件类
uint64 t temp session = user api pointer->Login(server ip oms.c str(), server port oms, account name ←
oms.c_str(), account_pw_oms.c_str(), XTP_PROTOCOL_TCP);//登陆交易服务器
if (temp session != 0)
{
//登录AlgoBus算法服务器
int login_ret = user_api_pointer->LoginALGO(server_ip_algo.c_str(), server_port_algo, account_name_algo.c_
str(), account_pw_algo.c_str(), XTP_PROTOCOL_TCP);
if (login ret != 0)
//登录算法服务器失败
std::cout << account_name_algo << " login to AlgoBus error!!!!!!!!!!" << std::endl;
XTPRI* error info = user api pointer->GetApiLastError(); //登录算法服务器失败时的错误原因代码
}
else
{
std::cout << account_name_algo << " login to AlgoBus success." << std::endl;
///在用户成功登录算法服务器后,算法用户建立算法通道
std::cout << account_name_oms << " begin to establish channel." << std::endl;
int user ret = user api pointer->ALGOUserEstablishChannel(server ip oms.c str(), server port oms, account ←
name oms.c str(), account pw oms.c str(), temp session );
if (user ret != 0)
std::cout << account name oms << " establish channel send error!!!!!!!!!!" << std::endl;
XTPRI* error info = user api pointer->GetApiLastError(): //建立算法通道消息发送失败时的错误原因代码
}
else
{
std::cout << account name oms << " establish channel send success." << std::endl;
}
else
//登录交易服务器失败
XTPRI* error info = user api pointer->GetApiLastError();
std::cout << "Login to server error, " << error_info->error_id << " : " << error_info->error_msg << std::endl;
//保持主线程不退出
while (true)
#ifdef WIN32
Sleep(1000);
#else
sleep(1);
#endif // WIN32
}
return 0;
在用户成功登录算法服务器后,算法用户建立算法通道
```

# 6.4 demo\_test\_trade\_spi.cpp 文件参考

Demo自定义客户端交易响应接口类

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

# 变量

XTP::API::TraderApi \* user\_api\_pointer
 API的全局变量指针

# 6.4.1 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

# 6.5 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

### 6.5.1 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

# 6.6 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
- struct XTPFundQueryReq 用户资金查询请求结构体
- struct XTPFundQueryRsp
   用户资金查询响应结构体

# 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

### 6.6.1 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

# 6.7 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"
#include "xtp_api_struct_common.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 XTPOrderInfoEx

报单响应结构体, 新版本

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 XTPQueryOrderByPageReq

查询订单请求-分页查询

struct XTPQueryReportByExecIdReq

struct XTPQueryTraderReq

查询成交回报请求-查询条件

struct XTPQueryTraderByPageReq

查询成交回报请求-分页查询

struct XTPQueryAssetRsp

账户资金查询响应结构体

struct XTPQueryStkPositionReq

查询股票持仓情况请求结构体

struct XTPQueryStkPositionRsp

查询股票持仓情况

struct XTPCreditDebtExtendNotice

用户展期请求的通知

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 XTPQueryETFComponentRspV1

查询股票ETF成分股信息--响应结构体,旧版本。

struct XTPQueryETFComponentRsp

查询股票ETF成分股信息--响应结构体

struct XTPQueryIPOTickerRsp

查询当日可申购新股信息

struct XTPQueryIPOQuotaRspV1

查询用户申购额度-旧版

struct XTPQueryIPOQuotaRsp

查询用户申购额度-包含创业板额度

struct XTPUserTerminalInfoReq

申报用户的ip和mac等信息,仅限授权用户使用

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

struct XTPCombLegStrategy

期权组合策略的成分合约信息

struct XTPQueryCombineStrategyInfoRsp

查询期权组合策略信息的响应

struct XTPOptCombLegInfo

组合策略腿合约信息结构体

• struct XTPOptCombPlugin

期权组合策略报单附加信息结构体

struct XTPQueryOptCombPositionReq

查询期权组合策略持仓情况请求结构体

struct XTPQueryOptCombPositionRsp

查询期权组合策略持仓信息的响应

struct XTPQueryOptExecInfoRsp

查询期权合约行权信息的响应

struct XTPQueryOptCombExecPosReq

查询期权行权合并头寸请求结构体

struct XTPQueryOptCombExecPosRsp

查询期权行权合并头寸的响应

struct XTPCrdCashRepayRsp

融资融券直接还款响应信息

struct XTPCrdCashRepayDebtInterestFeeRsp

融资融券现金还息费响应信息

struct XTPCrdCashRepayInfo

单条融资融券直接还款记录信息

struct XTPCrdDebtInfo

单条融资融券负债记录信息

• struct XTPCrdFundInfo

融资融券特有帐户数据

struct XTPClientQueryCrdDebtStockReq

融资融券指定证券上的负债未还数量请求结构体

struct XTPCrdDebtStockInfo

融资融券指定证券的融券负债相关信息

struct XTPClientQueryCrdPositionStockReq

融券头寸证券查询请求结构体

struct XTPClientQueryCrdPositionStkInfo

融券头寸证券信息

struct XTPClientQueryCrdSurplusStkReqInfo

信用业务余券查询请求结构体

• struct XTPClientQueryCrdSurplusStkRspInfo

信用业务余券信息

struct XTPCreditDebtExtendReq

用户展期请求

struct XTPCrdFundExtraInfo

融资融券帐户附加信息

struct XTPCrdPositionExtraInfo

融资融券帐户持仓附加信息

struct XTPOptCombOrderInsertInfo

期权组合策略新订单请求

struct XTPOptCombOrderInfo

期权组合策略报单响应结构体

struct XTPOptCombOrderInfoEx

期权组合策略报单响应结构体, 新版本

struct XTPOptCombTradeReport

期权组合策略报单成交结构体

struct XTPQueryOptCombOrderReq

struct XTPQueryOptCombOrderByPageReq

查询期权组合策略订单请求-分页查询

struct XTPQueryOptCombReportByExecIdReq

struct XTPQueryOptCombTraderReq

查询期权组合策略成交回报请求-查询条件

struct XTPQueryOptCombTraderByPageReq

查询期权组合策略成交回报请求-分页查询

#### 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 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

typedef struct XTPOrderCancelInfo XTPOptCombOrderCancelInfo

期权组合策略撤单错误响应结构体

typedef struct XTPOptCombLegInfo XTPOptCombLegInfo

组合策略腿合约信息结构体

typedef struct XTPOptCombPlugin XTPOptCombPlugin

期权组合策略报单附加信息结构体

typedef struct XTPCrdDebtInfo XTPCrdDebtInfo

单条融资融券负债记录信息

typedef struct XTPCrdFundInfo XTPCrdFundInfo

融资融券特有帐户数据

 typedef struct XTPClientQueryCrdDebtStockReq XTPClientQueryCrdDebtStockReq 融资融券指定证券上的负债未还数量请求结构体

typedef struct XTPCrdDebtStockInfo

融资融券指定证券的融券负债相关信息

typedef struct XTPClientQueryCrdPositionStockReq XTPClientQueryCrdPositionStockReq 融券头寸证券查询请求结构体

 typedef struct XTPClientQueryCrdPositionStkInfo XTPClientQueryCrdPositionStkInfo 融券头寸证券信息

- typedef struct XTPClientQueryCrdSurplusStkReqInfo XTPClientQueryCrdSurplusStkReqInfo
   信用业务余券查询请求结构体
- typedef struct XTPClientQueryCrdSurplusStkRspInfo XTPClientQueryCrdSurplusStkRspInfo 信用业务余券信息
- typedef struct XTPCreditDebtExtendNotice XTPCreditDebtExtendAck

用户展期请求的响应结构

 typedef struct XTPCrdFundExtraInfo XTPCrdFundExtraInfo 融资融券帐户附加信息

typedef struct XTPCrdPositionExtraInfo XTPCrdPositionExtraInfo 融资融券帐户持仓附加信息

 typedef struct XTPOptCombOrderInfo XTPQueryOptCombOrderRsp 期权组合策略报单查询响应结构体

 typedef struct XTPOptCombTradeReport XTPQueryOptCombTradeRsp 成交回报查询响应结构体

#### 6.7.1 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

# 6.8 xquote\_api\_rebuild\_tbt\_struct.h 文件参考

定义行情类相关数据结构

#include "xtp\_api\_data\_type.h"

#### 结构体

struct XTPQuoteRebuildReq

实时行情回补查询

struct XTPQuoteRebuildResultRsp

实时行情回补响应结构体

# 类型定义

- typedef struct XTPQuoteRebuildReq XTPQuoteRebuildReq 实时行情回补查询
- typedef struct XTPQuoteRebuildResultRsp XTPQuoteRebuildResultRsp 实时行情回补响应结构体

### 6.8.1 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

6.8.2 类型定义说明

#### 6.8.2.1 XTPQuoteRebuildReq

 ${\tt typedef \ struct \ XTPQuoteRebuildReq \ XTPQuoteRebuildReq}$ 

实时行情回补查询

实时行情回补请求结构体

# 6.9 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 等额外数据

• struct XTPMarketDataBondExData

债券额外数据

struct XTPMarketDataOptionExData

期权额外数据

struct XTPMarketDataStruct

行情

· struct XTPQuoteStaticInfo

股票行情静态信息

struct OrderBookStruct

订单薄

struct XTPTickByTickEntrust

逐笔委托

• struct XTPTickByTickTrade

逐笔成交

• struct XTPTickByTickStatus

逐笔状态订单

struct XTPTickByTickStruct

逐笔数据信息

• struct XTPTickerPriceInfo

供查询的最新信息

• struct XTPQuoteFullInfo

股票行情全量静态信息

# 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

订单薄

typedef struct XTPTickByTickStruct XTPTBT

逐笔数据信息

• typedef struct XTPTickerPriceInfo XTPTPI

供查询的最新信息

typedef struct XTPQuoteFullInfo XTPQFI

股票行情全量静态信息

### 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, 2.2.32以前版本所用

 enum XTP\_MARKETDATA\_TYPE\_V2 { XTP\_MARKETDATA\_V2\_INDEX = 0, XTP\_MARKETDATA\_V2. \_OPTION = 1, XTP\_MARKETDATA\_V2\_ACTUAL = 2, XTP\_MARKETDATA\_V2\_BOND = 3 }

XTP\_MARKETDATA\_TYPE\_V2是行情快照数据类型, 2.2.32版本新增字段

# 6.9.1 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

# 6.10 xtp\_api\_data\_type.h 文件参考

定义兼容数据基本类型

#### 宏定义

- #define MAX\_TGW\_CNT\_PER\_PBU 10 每个PBU最多被10个TGW使用。
- #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\_CREDIT\_DEBT\_ID\_LEN 33 信用业务合约负债编号长度
- #define XTP\_INET\_ADDRESS\_STR\_LEN 64 IP地址的字符串长度
- #define XTP\_MAC\_ADDRESS\_LEN 16

MAC地址的字符串长度

#define XTP\_HARDDISK\_SN\_LEN 24

硬盘序列号的字符串长度

#define XTP\_MACOS\_SNO\_LEN 21

MacOS系统序列号的字符串长度

• #define XTP\_STRATEGE\_LEG\_NUM 4

期权组合策略最多腿数

• #define XTP\_STRATEGY\_ID\_LEN 10

期权组合策略代码字符串长度

#define XTP\_STRATEGY\_NAME\_LEN 32

期权组合策略名称字符串长度

• #define XTP\_SECONDARY\_ORDER\_ID\_LEN 18

期权组合策略组合编码字符串长度

#define XTP\_CNTRT\_COMB\_STRA\_LIST\_LEN 2048

期权合约可支持的组合策略列表字符串长度

• #define XTP\_COMBINED\_EXECUTION\_LEG\_NUM 2

期权行权合并最多成分合约数量

• #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

申贴

#define XTP\_SIDE\_REDEMPTION 8

陸回

• #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\_STOCK\_REPAY\_STOCK 26

现金还款(不放在普通订单协议,另加请求和查询协议)

#define XTP\_SIDE\_SURSTK\_TRANS 27

余券划转

• #define XTP SIDE GRTSTK TRANSIN 28

担保品转λ

#define XTP\_SIDE\_GRTSTK\_TRANSOUT 29

担保品转出

• #define XTP\_SIDE\_OPT\_COMBINE 31

组合策略的组合

#define XTP\_SIDE\_OPT\_SPLIT 32
 组合策略的拆分

• #define XTP SIDE OPT SPLIT FORCE 33

组合策略的管理员强制拆分

#define XTP\_SIDE\_OPT\_SPLIT\_FORCE\_EXCH 34

组合策略的交易所强制拆分

#define XTP\_SIDE\_UNKNOWN 50

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, 除期权外, 均使用此值

#define XTP\_POSITION\_EFFECT\_OPEN 1

开

#define XTP\_POSITION\_EFFECT\_CLOSE 2

平

• #define XTP\_POSITION\_EFFECT\_FORCECLOSE 3

#define XTP\_POSITION\_EFFECT\_CLOSETODAY 4

平今

• #define XTP\_POSITION\_EFFECT\_CLOSEYESTERDAY 5

平的

#define XTP\_POSITION\_EFFECT\_FORCEOFF 6

强减

#define XTP\_POSITION\_EFFECT\_LOCALFORCECLOSE 7

本地强平

#define XTP\_POSITION\_EFFECT\_CREDIT\_FORCE\_COVER 8

信用业务追保强平

#define XTP\_POSITION\_EFFECT\_CREDIT\_FORCE\_CLEAR 9

信用业务清偿强平

#define XTP\_POSITION\_EFFECT\_CREDIT\_FORCE\_DEBT 10

信用业务合约到期强平

#define XTP\_POSITION\_EFFECT\_CREDIT\_FORCE\_UNCOND 11

信用业务无条件强平

#define XTP\_POSITION\_EFFECT\_UNKNOWN 12

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 TRDT CROSS MKT CASH '3'

跨市场资金成交

#define XTP\_TRDT\_HK\_MKT\_CASH '4'

港市资金成交

#define XTP\_TRDT\_NON\_SHSZ\_MKT\_CASH '5'

非沪深资金成交

#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, 行情里使用

typedef enum XTP\_MARKET\_TYPE XTP\_MARKET\_TYPE

XTP\_MARKET\_TYPE市场类型,交易里使用

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

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 QUERY TYPE XTP FUND QUERY TYPE

XTP\_FUND\_QUERY\_TYPE是柜台资金查询类型

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\_DEBT\_EXTEND\_OPER\_STATUS XTP\_DEBT\_EXTEND\_OPER\_STATUS

XTP\_DEBT\_EXTEND\_OPER\_STATUS柜台负债展期操作状态

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\_QUOTE\_REBUILD\_DATA\_TYPE XTP\_QUOTE\_REBUILD\_DATA\_TYPE

XTP\_QUOTE\_DATA\_TYPE是行情数据类型逐笔,快照等

typedef enum XTP\_REBUILD\_RET\_TYPE XTP\_REBUILD\_RET\_TYPE

XTP\_REBUILD\_RET\_TYPE 实时行情回补返回结果类型

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 enum XTP OPT COVERED OR UNCOVERED XTP OPT COVERED OR UNCOVERED

XTP\_OPT\_COVERED\_OR\_UNCOVERED是否备兑的标签

typedef enum XTP CRD CR STATUS XTP CRD CR STATUS

XTP\_CRD\_CASH\_REPAY\_STATUS是一个融资融券直接还款状态类型

typedef enum XTP\_OPT\_POSITION\_TYPE XTP\_OPT\_POSITION\_TYPE

XTP\_OPT\_POSITION\_TYPE是一个期权持仓类型

typedef char TXTPTradeTypeType

TXTPTradeTypeType是成交类型类型

typedef char TXTPOrderTypeType

TXTPOrderTypeType是报单类型类型

typedef enum XTP\_EXPIRE\_DATE\_TYPE XTP\_EXPIRE\_DATE\_TYPE

XTP\_EXPIRE\_DATE\_TYPE是一个期权组合策略合约到期日要求类型

typedef enum XTP\_UNDERLYING\_TYPE XTP\_UNDERLYING\_TYPE

XTP\_UNDERLYING\_TYPE是一个期权组合策略标的要求类型

typedef enum XTP\_AUTO\_SPLIT\_TYPE XTP\_AUTO\_SPLIT\_TYPE

XTP\_AUTO\_SPLIT\_TYPE是一个期权组合策略自动解除枚举类型

typedef char TXTPExerciseSeqType

行权价顺序类型,从1开始,1表示行权价最高,2次之。如果行权价相同,则填写相同数字,用A表示行权价大于等于B,B大于等于C依次类推(C、D)

typedef enum XTP\_QUALIFICATION\_TYPE XTP\_QUALIFICATION\_TYPE

XTP QUALIFICATION TYPE是一个证券适当性枚举类型

typedef enum XTP\_SECURITY\_TYPE XTP\_SECURITY\_TYPE

XTP\_SECURITY\_TYPE是一个证券详细分类枚举类型

typedef enum XTP\_POSITION\_SECURITY\_TYPE XTP\_POSITION\_SECURITY\_TYPE

XTP POSITION SECURITY TYPE是一个持仓证券枚举类型

typedef enum XTP SECURITY STATUS XTP SECURITY STATUS

XTP\_SECURITY\_STATUS是一个证券状态枚举类型

#### 枚举

enum XTP\_LOG\_LEVEL {
 XTP\_LOG\_LEVEL\_FATAL, XTP\_LOG\_LEVEL\_ERROR, XTP\_LOG\_LEVEL\_WARNING, XTP\_LOG\_LEVEL\_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,行情里使用

    enum XTP MARKET TYPE { XTP MKT INIT = 0, XTP MKT SZ A = 1, XTP MKT SH A, XTP MKT UNKNOWN

   XTP_MARKET_TYPE市场类型,交易里使用
enum XTP PRICE TYPE {
 XTP PRICE LIMIT = 1, XTP PRICE BEST OR CANCEL, XTP PRICE BEST5 OR LIMIT, XTP PRICE BEST5 OR CAN
 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,
 XTP 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_PARTTRADEDQUEUE
 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 RECOMPUTE INTER SZ,
 ERT_CASH_MUST_INTER_SZ, ERT_CASH_RECOMPUTE_INTER_OTHER, ERT_CASH_MUST_INTER_OTHER,
 ERT CASH RECOMPUTE INTER HK,
 ERT CASH MUST INTER HK, 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 TICKER TYPE BOND,
 XTP_TICKER_TYPE_OPTION, XTP_TICKER_TYPE_TECH_STOCK, 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,
 XTP BUSINESS TYPE ETF,
 XTP BUSINESS TYPE MARGIN, XTP BUSINESS TYPE DESIGNATION, XTP BUSINESS TYPE ALLOTMENT,
 XTP BUSINESS TYPE STRUCTURED FUND PURCHASE REDEMPTION.
 XTP BUSINESS TYPE STRUCTURED FUND SPLIT MERGE, XTP BUSINESS TYPE MONEY FUND,
 XTP BUSINESS TYPE OPTION, XTP BUSINESS TYPE EXECUTE,
 XTP BUSINESS TYPE FREEZE, XTP BUSINESS TYPE OPTION COMBINE, XTP BUSINESS TYPE EXECUTE COMI
 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_ACCOUNT_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_INTER_TRANSFER_OUT,
 XTP_FUND_INTER_TRANSFER_IN,
 XTP FUND_TRANSFER_UNKNOWN }
```

```
XTP FUND TRANSFER TYPE是资金流转方向类型

    enum XTP_FUND_QUERY_TYPE { XTP_FUND_QUERY_JZ = 0, XTP_FUND_QUERY_INTERNAL,

 XTP_FUND_QUERY_UNKNOWN }
   XTP FUND QUERY TYPE是柜台资金查询类型
enum XTP FUND OPER STATUS {
 XTP FUND OPER PROCESSING = 0, XTP FUND OPER SUCCESS, XTP FUND OPER FAILED,
 XTP_FUND_OPER_SUBMITTED,
 XTP_FUND_OPER_UNKNOWN }
   XTP_FUND_OPER_STATUS柜台资金操作结果

    enum XTP DEBT EXTEND OPER STATUS {

 XTP DEBT EXTEND OPER PROCESSING = 0, XTP DEBT EXTEND OPER SUBMITTED, XTP DEBT EXTEND OPER
 XTP DEBT EXTEND OPER FAILED,
 XTP_DEBT_EXTEND_OPER_UNKNOWN }
   XTP_DEBT_EXTEND_OPER_STATUS柜台负债展期操作状态

    enum XTP SPLIT MERGE STATUS {XTP SPLIT MERGE STATUS ALLOW = 0, XTP SPLIT MERGE STATUS ONLY $

 XTP SPLIT MERGE STATUS ONLY MERGE, XTP SPLIT MERGE STATUS 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_STATE = 3 }
   XTP TBT TYPE是一个逐笔回报类型

    enum XTP QUOTE REBUILD DATA TYPE { XTP QUOTE REBUILD UNKNOW = 0, XTP QUOTE REBUILD MD

 = 1, XTP_QUOTE_REBUILD_TBT = 2}
   XTP_QUOTE_DATA_TYPE是行情数据类型逐笔,快照等

    enum XTP REBUILD RET TYPE {

 XTP_REBUILD_RET_COMPLETE = 1, XTP_REBUILD_RET_PARTLY = 2, XTP_REBUILD_RET_NO_DATA
 = 3, XTP_REBUILD_RET_PARAM_ERR = 4,
 XTP REBUILD RET FREQUENTLY = 5 }
    XTP_REBUILD_RET_TYPE 实时行情回补返回结果类型
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 EXERCISE TYPE AME
   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_DIRECTION_LONG,
 XTP_POSITION_DIRECTION_SHORT, XTP_POSITION_DIRECTION_COVERED }
   XTP_POSITION_DIRECTION_TYPE是一个持仓方向类型
enum XTP_OPT_COVERED_OR_UNCOVERED { XTP_POSITION_UNCOVERED = 0, XTP_POSITION_COVERED
 }
   XTP_OPT_COVERED_OR_UNCOVERED是否备兑的标签

    enum XTP CRD CR STATUS XTP CRD CR INIT = 0, XTP CRD CR SUCCESS, XTP CRD CR FAILED

 }
   XTP_CRD_CASH_REPAY_STATUS是一个融资融券直接还款状态类型

    enum XTP OPT POSITION TYPE { XTP OPT POSITION TYPE CONTRACT = 0, XTP OPT POSITION TYPE COMBINE

 = 1 }
   XTP_OPT_POSITION_TYPE是一个期权持仓类型

    enum XTP ORDER DETAIL TYPE {XTP ORDER DETAIL TYPE NEW ORDER = 0, XTP ORDER DETAIL TYPE CANC

 = 1, XTP ORDER DETAIL TYPE OPT COMB NEW ORDER = 2, XTP ORDER DETAIL TYPE OPT COMB CANCEL O
 = 3 }
   XTP_ORDER_TYPE是一个订单的类型
• enum XTPTerminalType {
 XTP TERMINAL PC = 1, XTP TERMINAL ANDROID, XTP TERMINAL IOS, XTP TERMINAL WP,
 XTP TERMINAL STATION, XTP TERMINAL TEL, XTP TERMINAL PC LINUX }
    XTPTerminalType是一种终端类型枚举,仅供授权系统使用
enum XTP_EXPIRE_DATE_TYPE { XTP_EXP_DATE_SAME = 0, XTP_EXP_DATE_DIFF, XTP_EXP_DATE_NON
 }
```

XTP\_EXPIRE\_DATE\_TYPE是一个期权组合策略合约到期日要求类型

enum XTP\_UNDERLYING\_TYPE { XTP\_UNDERLYING\_SAME = 0, XTP\_UNDERLYING\_DIFF, XTP\_UNDERLYING\_NON }

XTP\_UNDERLYING\_TYPE是一个期权组合策略标的要求类型

enum XTP\_AUTO\_SPLIT\_TYPE { XTP\_AUTO\_SPLIT\_EXPDAY = 0, XTP\_AUTO\_SPLIT\_PREDAY, XTP AUTO SPLIT PRE2DAY, XTP AUTO SPLIT NON}

XTP\_AUTO\_SPLIT\_TYPE是一个期权组合策略自动解除枚举类型

enum XTP\_QUALIFICATION\_TYPE { XTP\_QUALIFICATION\_PUBLIC = 0, XTP\_QUALIFICATION\_COMMON = 1, XTP\_QUALIFICATION\_ORGANIZATION = 2, XTP\_QUALIFICATION\_UNKNOWN = 3 }

XTP QUALIFICATION TYPE是一个证券适当性枚举类型

enum XTP SECURITY TYPE {

XTP\_SECURITY\_MAIN\_BOARD = 0, XTP\_SECURITY\_SECOND\_BOARD, XTP\_SECURITY\_STARTUP\_BOARD, XTP SECURITY INDEX,

XTP\_SECURITY\_TECH\_BOARD = 4, XTP\_SECURITY\_STATE\_BOND = 5, XTP\_SECURITY\_ENTERPRICE\_BOND = 6, XTP\_SECURITY\_COMPANEY\_BOND = 7,

XTP\_SECURITY\_CONVERTABLE\_BOND = 8, XTP\_SECURITY\_NATIONAL\_BOND\_REVERSE\_REPO = 12, XTP\_SECURITY\_ETF\_SINGLE\_MARKET\_STOCK = 14, XTP\_SECURITY\_ETF\_INTER\_MARKET\_STOCK,

XTP\_SECURITY\_ETF\_CROSS\_BORDER\_STOCK = 16, XTP\_SECURITY\_ETF\_SINGLE\_MARKET\_BOND = 17, XTP\_SECURITY\_ETF\_GOLD = 19, XTP\_SECURITY\_STRUCTURED\_FUND\_CHILD = 24,

XTP\_SECURITY\_SZSE\_RECREATION\_FUND = 26, XTP\_SECURITY\_STOCK\_OPTION = 29, XTP\_SECURITY\_ETF\_OPTIC = 30, XTP\_SECURITY\_ALLOTMENT = 100,

XTP\_SECURITY\_MONETARY\_FUND\_SHCR = 110, XTP\_SECURITY\_MONETARY\_FUND\_SHTR = 111, XTP\_SECURITY\_MONETARY\_FUND\_SZ = 112, XTP\_SECURITY\_OTHERS = 255 }

XTP SECURITY TYPE是一个证券详细分类枚举类型

enum XTP\_POSITION\_SECURITY\_TYPE { XTP\_POSITION\_SECURITY\_NORMAL = 0, XTP\_POSITION\_SECURITY\_PLACE = 1, XTP\_POSITION\_SECURITY\_UNKNOWN = 2 }

XTP POSITION SECURITY TYPE是一个持仓证券枚举类型

enum XTP\_SECURITY\_STATUS {

XTP\_SECURITY\_STATUS\_ST = 0, XTP\_SECURITY\_STATUS\_N\_IPO, XTP\_SECURITY\_STATUS\_COMMON, XTP\_SECURITY\_STATUS\_RESUME,

XTP SECURITY STATUS DELISTING = 10, XTP SECURITY STATUS OTHERS = 255 }

XTP\_SECURITY\_STATUS是一个证券状态枚举类型

#### 6.10.1 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

6.10.2 宏定义说明

#### 6.10.2.1 XTP\_SIDE\_STOCK\_REPAY\_STOCK

#define XTP\_SIDE\_STOCK\_REPAY\_STOCK 26

现金还款 (不放在普通订单协议, 另加请求和查询协议)

现券还券

# 6.10.3 枚举类型说明

# 6.10.3.1 ETF\_REPLACE\_TYPE

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(不适用于跨沪深 港ETF产品)	
ERT_CASH_MUST_INTER_OTHER	表示非沪深市场成份证券必须现金替代(不适用于跨沪 深港ETF产品)	
ERT_CASH_MUST_INTER_HK	港市退补现金替代(仅适用于跨沪深港ETF产品)	
EPT_INVALID	港市必须现金替代(仅适用于跨沪深港ETF产品) 无效 值	

#### 6.10.3.2 XTP\_ACCOUNT\_TYPE

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

#### 6.10.3.3 XTP\_AUTO\_SPLIT\_TYPE

enum XTP\_AUTO\_SPLIT\_TYPE

XTP\_AUTO\_SPLIT\_TYPE是一个期权组合策略自动解除枚举类型

# 枚举值

XTP_AUTO_SPLIT_EXPDAY	到期日自动解除
XTP_AUTO_SPLIT_PREDAY	E-1日自动解除
XTP_AUTO_SPLIT_PRE2DAY	E-2日自动解除
XTP_AUTO_SPLIT_NON	无效值

### 6.10.3.4 XTP\_BUSINESS\_TYPE

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,沪市交易型货币基金等)
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(国债逆回购业务对应的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
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	货币基金申赎业务(暂未支持,沪市交易型货币 基金的买卖请使用普通股票业务)
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_OPTION_COMBINE	期权组合策略 组合和拆分业务
XTP_BUSINESS_TYPE_EXECUTE_COMBINE	期权行权合并业务
XTP_BUSINESS_TYPE_UNKNOWN	未知类型

6.10.3.5 XTP\_CRD\_CR\_STATUS

enum XTP\_CRD\_CR\_STATUS

XTP\_CRD\_CASH\_REPAY\_STATUS是一个融资融券直接还款状态类型

#### 枚举值

XTP_CRD_CR_INIT	初始、未处理状态
XTP_CRD_CR_SUCCESS	已成功处理状态
XTP_CRD_CR_FAILED	处理失败状态

6.10.3.6 XTP\_DEBT\_EXTEND\_OPER\_STATUS

enum XTP\_DEBT\_EXTEND\_OPER\_STATUS

XTP\_DEBT\_EXTEND\_OPER\_STATUS柜台负债展期操作状态

### 枚举值

XTP_DEBT_EXTEND_OPER_PROCESSING	XTP已收到,正在处理中
XTP_DEBT_EXTEND_OPER_SUBMITTED	已提交到集中柜台处理
XTP_DEBT_EXTEND_OPER_SUCCESS	成功
XTP_DEBT_EXTEND_OPER_FAILED	失败
XTP_DEBT_EXTEND_OPER_UNKNOWN	未知

6.10.3.7 XTP\_EXCHANGE\_TYPE

enum XTP\_EXCHANGE\_TYPE

XTP\_EXCHANGE\_TYPE是交易所类型,行情里使用

### 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

6.10.3.8 XTP\_EXPIRE\_DATE\_TYPE

enum XTP\_EXPIRE\_DATE\_TYPE

XTP\_EXPIRE\_DATE\_TYPE是一个期权组合策略合约到期日要求类型

#### 枚举值

XTP_EXP_DATE_SAME	相同到期日
XTP_EXP_DATE_DIFF	不同到期日
XTP_EXP_DATE_NON	无到期日要求

6.10.3.9 XTP\_FUND\_OPER\_STATUS

enum XTP\_FUND\_OPER\_STATUS

XTP\_FUND\_OPER\_STATUS柜台资金操作结果

### 枚举值

XTP_FUND_OPER_PROCESSING	XTP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

6.10.3.10 XTP\_FUND\_QUERY\_TYPE

enum XTP\_FUND\_QUERY\_TYPE

XTP\_FUND\_QUERY\_TYPE是柜台资金查询类型

### 枚举值

XTP_FUND_QUERY_JZ	查询金证主柜台可转资金
XTP_FUND_QUERY_INTERNAL	查询一账号两中心设置时,对方节点的资金
XTP_FUND_QUERY_UNKNOWN	未知类型

6.10.3.11 XTP\_FUND\_TRANSFER\_TYPE

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_INTER_TRANSFER_OUT	跨节点转出 从本XTP节点1,转出到对端XTP节点2,XTP服务器之间划拨,只能"一账号两中心"跨节点用户使用
XTP_FUND_INTER_TRANSFER_IN	跨节点转入 从对端XTP节点2,转入到本XTP节点1,XTP服务器之间划拨,只能"一账号两中心"跨节点用户使用
XTP_FUND_TRANSFER_UNKNOWN	未知类型

6.10.3.12 XTP\_LOG\_LEVEL

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

6.10.3.13 XTP\_MARKET\_TYPE

enum XTP\_MARKET\_TYPE

XTP\_MARKET\_TYPE市场类型,交易里使用

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

6.10.3.14 XTP\_OPT\_CALL\_OR\_PUT\_TYPE

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

6.10.3.15 XTP\_OPT\_COVERED\_OR\_UNCOVERED

enum XTP\_OPT\_COVERED\_OR\_UNCOVERED

XTP\_OPT\_COVERED\_OR\_UNCOVERED是否备兑的标签

枚举值

XTP_POSITION_UNCOVERED	非备兑
XTP_POSITION_COVERED	备兑

6.10.3.16 XTP\_OPT\_EXERCISE\_TYPE\_TYPE

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

6.10.3.17 XTP\_OPT\_POSITION\_TYPE

enum XTP\_OPT\_POSITION\_TYPE

XTP\_OPT\_POSITION\_TYPE是一个期权持仓类型

### 枚举值

XTP_OPT_POSITION_TYPE_CONTRACT	单合约持仓
XTP_OPT_POSITION_TYPE_COMBINED	组合策略持仓

### 6.10.3.18 XTP\_ORDER\_ACTION\_STATUS\_TYPE

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

### 6.10.3.19 XTP\_ORDER\_DETAIL\_TYPE

enum XTP\_ORDER\_DETAIL\_TYPE

XTP\_ORDER\_TYPE是一个订单的类型

# 枚举值

XTP_ORDER_DETAIL_TYPE_NEW_ORDER	新订单
XTP_ORDER_DETAIL_TYPE_CANCEL_ORDER	新订单撤单
XTP_ORDER_DETAIL_TYPE_OPT_COMB_NEW_ORDER	组合订单
XTP_ORDER_DETAIL_TYPE_OPT_COMB_CANCEL_ORDER	组合订单撤单

### 6.10.3.20 XTP\_ORDER\_STATUS\_TYPE

enum XTP\_ORDER\_STATUS\_TYPE

XTP\_ORDER\_STATUS\_TYPE是报单状态类型

XTP_ORDER_STATUS_INIT	初始化
-----------------------	-----

### 枚举值

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

### 6.10.3.21 XTP\_ORDER\_SUBMIT\_STATUS\_TYPE

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

### 6.10.3.22 XTP\_POSITION\_DIRECTION\_TYPE

enum XTP\_POSITION\_DIRECTION\_TYPE

# XTP\_POSITION\_DIRECTION\_TYPE是一个持仓方向类型

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)

#### 6.10.3.23 XTP\_POSITION\_SECURITY\_TYPE

enum XTP\_POSITION\_SECURITY\_TYPE

XTP\_POSITION\_SECURITY\_TYPE是一个持仓证券枚举类型

#### 枚举值

XTP_POSITION_SECURITY_NORMAL	普通持仓
XTP_POSITION_SECURITY_PLACEMENT	配售类型的持仓,包含配股、配债等
XTP_POSITION_SECURITY_UNKNOWN	未知类型

6.10.3.24 XTP\_PRICE\_TYPE

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/深期权 (除普通股票业务外, 其余未 特指的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/深期权
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/深期权
XTP_PRICE_ALL_OR_CANCEL	全部成交或撤销,市价单-深/沪期权/深期权
XTP_PRICE_FORWARD_BEST	本方最优,市价单-深/深期权/沪科创板
	对方最优剩余转限价,市价单-深/沪期权/深期权/沪科创板
XTP_PRICE_REVERSE_BEST_LIMIT	
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

6.10.3.25 XTP\_PROTOCOL\_TYPE

enum XTP\_PROTOCOL\_TYPE

XTP\_PROTOCOL\_TYPE是通讯传输协议方式

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)

#### 6.10.3.26 XTP\_QUALIFICATION\_TYPE

enum XTP\_QUALIFICATION\_TYPE

### XTP\_QUALIFICATION\_TYPE是一个证券适当性枚举类型

#### 枚举值

XTP_QUALIFICATION_PUBLIC	公众投资者,合格投资者与机构投资者均可	
XTP_QUALIFICATION_COMMON	仅合格投资者与公众投资者	
XTP_QUALIFICATION_ORGANIZATION	仅限机构投资者	
XTP_QUALIFICATION_UNKNOWN	N 未知,期权等可能为此种类型	

#### 6.10.3.27 XTP\_QUOTE\_REBUILD\_DATA\_TYPE

enum XTP\_QUOTE\_REBUILD\_DATA\_TYPE

XTP\_QUOTE\_DATA\_TYPE是行情数据类型逐笔,快照等

### 枚举值

XTP_QUOTE_REBUILD_UNKNOW	未知类型
XTP_QUOTE_REBUILD_MD	快照类型
XTP_QUOTE_REBUILD_TBT	逐笔类型

#### 6.10.3.28 XTP\_REBUILD\_RET\_TYPE

enum XTP\_REBUILD\_RET\_TYPE

XTP\_REBUILD\_RET\_TYPE 实时行情回补返回结果类型

XTP_REBUILD_RET_COMPLETE	全部数据
XTP_REBUILD_RET_PARTLY	部分数据
XTP_REBUILD_RET_NO_DATA	没有数据
XTP_REBUILD_RET_PARAM_ERR	参数错误
XTP_REBUILD_RET_FREQUENTLY	请求频繁

#### 6.10.3.29 XTP\_SECURITY\_STATUS

enum XTP\_SECURITY\_STATUS

XTP\_SECURITY\_STATUS是一个证券状态枚举类型

### 枚举值

XTP_SECURITY_STATUS_ST	风险警示板
XTP_SECURITY_STATUS_N_IPO	首日上市
XTP_SECURITY_STATUS_COMMON	普通
XTP_SECURITY_STATUS_RESUME	恢复上市
XTP_SECURITY_STATUS_DELISTING	退市整理期
XTP_SECURITY_STATUS_OTHERS	其他

# 6.10.3.30 XTP\_SECURITY\_TYPE

enum XTP\_SECURITY\_TYPE

XTP\_SECURITY\_TYPE是一个证券详细分类枚举类型

XTP_SECURITY_MAIN_BOARD	主板股票
XTP_SECURITY_SECOND_BOARD	中小板股票
XTP_SECURITY_STARTUP_BOARD	创业板股票
XTP_SECURITY_INDEX	指数
XTP_SECURITY_TECH_BOARD	科创板股票(上海)
XTP_SECURITY_STATE_BOND	国债
XTP_SECURITY_ENTERPRICE_BOND	企业债
XTP_SECURITY_COMPANEY_BOND	公司债
XTP_SECURITY_CONVERTABLE_BOND	转换债券
XTP_SECURITY_NATIONAL_BOND_REVERSE_REPO	国债逆回购
XTP_SECURITY_ETF_SINGLE_MARKET_STOCK	本市场股票 ETF
XTP_SECURITY_ETF_INTER_MARKET_STOCK	跨市场股票 ETF
XTP_SECURITY_ETF_SINGLE_MARKET_BOND	本市场实物债券 ETF
XTP_SECURITY_ETF_GOLD	黄金 ETF
XTP_SECURITY_STRUCTURED_FUND_CHILD	分级基金子基金
XTP_SECURITY_SZSE_RECREATION_FUND	深交所仅申赎基金
XTP_SECURITY_STOCK_OPTION	个股期权
XTP_SECURITY_ETF_OPTION	ETF期权
XTP_SECURITY_ALLOTMENT	配股

#### 枚举值

XTP_SECURITY_MONETARY_FUND_SHCR	上交所申赎型货币基金
XTP_SECURITY_MONETARY_FUND_SHTR	上交所交易型货币基金
XTP_SECURITY_MONETARY_FUND_SZ	深交所货币基金
XTP_SECURITY_OTHERS	其他

### 6.10.3.31 XTP\_SPLIT\_MERGE\_STATUS

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

#### 6.10.3.32 XTP\_TBT\_TYPE

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
XTP TBT STATE	逐笔状态订单, 2.2.32版本新增字段, 为上海新债券Level2行情中独有

#### 6.10.3.33 XTP\_TE\_RESUME\_TYPE

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

6.10.3.34 XTP\_TICKER\_TYPE

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_TECH_STOCK	科创板股票 (上海)
XTP_TICKER_TYPE_UNKNOWN	未知类型

6.10.3.35 XTP\_UNDERLYING\_TYPE

enum XTP\_UNDERLYING\_TYPE

XTP\_UNDERLYING\_TYPE是一个期权组合策略标的要求类型

# 枚举值

XTP_UNDERLYING_SAME	相同标的
XTP_UNDERLYING_DIFF	不同标的
XTP_UNDERLYING_NON	无标的要求

6.10.3.36 XTPTerminalType

enum XTPTerminalType

XTPTerminalType是一种终端类型枚举,仅供授权系统使用

#### 枚举值

XTP_TERMINAL_PC	"PC",PC-windows及MacOS
XTP_TERMINAL_ANDROID	"MA",Mobile-Android
XTP_TERMINAL_IOS	"MI",Mobile-los
XTP_TERMINAL_WP	"MW",Mobile-Windows Phone
XTP_TERMINAL_STATION	"WP",无盘站
XTP_TERMINAL_TEL	"TO",电话委托
XTP_TERMINAL_PC_LINUX	"OH",PC-linux及其他终端

# 6.11 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"
#include "xquote_api_rebuild_tbt_struct.h"
```

### 6.11.1 详细描述

定义业务数据结构

作者

中泰证券股份有限公司

# 6.12 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

### 6.12.1 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

# 6.13 xtp\_trader\_api.h 文件参考

定义客户端交易接口

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

# 结构体

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

# 6.13.1 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

ALGOUserEstablishChannel	XTP::API::TraderApi, 26
XTP::API::TraderApi, 19	GetClientIDByXTPID
algo_api_struct.h, 163	XTP::API::TraderApi, 27
algo_data_type.h, 164	GetTradingDay
	XTP::API::TraderApi, 27
CancelAlgoOrder	
XTP::API::TraderApi, 20	InsertAlgoOrder
CancelOptionCombinedOrder	XTP::API::TraderApi, 27
XTP::API::TraderApi, 20	InsertOptionCombinedOrder
CancelOrder	XTP::API::TraderApi, 28
XTP::API::TraderApi, 21	InsertOptionCombinedOrderExtra
CreateTraderApi	XTP::API::TraderApi, 29
XTP::API::TraderApi, 22	InsertOrder
CreditCashRepay	XTP::API::TraderApi, 29
XTP::API::TraderApi, 22	InsertOrderExtra
CreditCashRepayDebtInterestFee	XTP::API::TraderApi, 30
XTP::API::TraderApi, 23	IsServerRestart
CreditExtendDebtDate	XTP::API::TraderApi, 30
XTP::API::TraderApi, 23	μ,,,,,
CreditSellStockRepayDebtInterestFee	Login
XTP::API::TraderApi, 24	XTP::API::TraderApi, 31
	LoginALGO
demo_test_trade_api.cpp, 165	XTP::API::TraderApi, 32
main, 165	Logout
demo_test_trade_spi.cpp, 167	XTP::API::TraderApi, 32
demo_test_trade_spi.h, 167	, , , , , , , , , , , , , , , , , , ,
DemoTestTraderSpi, 11	main
OnALGOUserEstablishChannel, 12	demo_test_trade_api.cpp, 165
OnAlgoConnected, 12	market
OnAlgoDisconnected, 12	XTPOptCombOrderInfo, 115
OnCancelAlgoOrder, 13	XTPOptCombOrderInfoEx, 117
OnDisconnected, 14	ModifyUserTerminalInfo
OnInsertAlgoOrder, 14	XTP::API::TraderApi, 33
OnQueryStrategy, 14	7011 III III III III III III III III III
OnStrategyStateReport, 15	OnALGOUserEstablishChannel
	DemoTestTraderSpi, 12
ETF_REPLACE_TYPE	XTP::API::TraderSpi, 63
xtp_api_data_type.h, 183	OnAlgoConnected
EundTranafar	DemoTestTraderSpi, 12
FundTransfer	OnAlgoDisconnected
XTP::API::TraderApi, 24	DemoTestTraderSpi, 12
GetANewOrderXTPID	XTP::API::TraderSpi, 63
XTP::API::TraderApi, 26	OnCancelAlgoOrder
GetAccountByXTPID	DemoTestTraderSpi, 13
XTP::API::TraderApi, 25	XTP::API::TraderSpi, 64
GetAlgorithmIDByOrder	OnCancelOptionCombinedOrderError
XTP::API::TraderApi, 25	XTP::API::TraderSpi, 64
•	OnCancelOrderError
GetApiLastError	XTP::API::TraderSpi, 65
XTP::API::TraderApi, 26	•
GetApiVersion	OnCreditCashRepay

XTP::API::TraderSpi, 65	XTP::API::TraderSpi, 82
OnCreditCashRepayDebtInterestFee	OnQueryOptionCombinedExecPosition
XTP::API::TraderSpi, 66	XTP::API::TraderSpi, 83
OnCreditExtendDebtDate	OnQueryOptionCombinedOrders
XTP::API::TraderSpi, 66	XTP::API::TraderSpi, 84
OnDisconnected	OnQueryOptionCombinedOrdersByPage
DemoTestTraderSpi, 14	XTP::API::TraderSpi, 84
XTP::API::TraderSpi, 67	OnQueryOptionCombinedOrdersByPageEx
OnError	XTP::API::TraderSpi, 85
XTP::API::TraderSpi, 67	OnQueryOptionCombinedOrdersEx
OnFundTransfer	XTP::API::TraderSpi, 86
XTP::API::TraderSpi, 68	OnQueryOptionCombinedPosition
OnInsertAlgoOrder	XTP::API::TraderSpi, 86
DemoTestTraderSpi, 14	OnQueryOptionCombinedStrategyInfo
XTP::API::TraderSpi, 68	XTP::API::TraderSpi, 87
OnOptionCombinedOrderEvent	OnQueryOptionCombinedTrades
XTP::API::TraderSpi, 69	XTP::API::TraderSpi, 87
OnOptionCombinedTradeEvent	OnQueryOptionCombinedTradesByPage
XTP::API::TraderSpi, 69	XTP::API::TraderSpi, 88
OnOrderEvent	OnQueryOrder
XTP::API::TraderSpi, 71	XTP::API::TraderSpi, 89
OnQueryAccountTradeMarket	OnQueryOrderByPage
XTP::API::TraderSpi, 71	XTP::API::TraderSpi, 89
OnQueryAsset	OnQueryOrderByPageEx
XTP::API::TraderSpi, 72	XTP::API::TraderSpi, 90
OnQueryCreditAssetDebtInfo	OnQueryOrderEx
XTP::API::TraderSpi, 73	XTP::API::TraderSpi, 91
OnQueryCreditCashRepayInfo	OnQueryOtherServerFund
XTP::API::TraderSpi, 73	XTP::API::TraderSpi, 91
OnQueryCreditDebtInfo	OnQueryPosition
XTP::API::TraderSpi, 74	XTP::API::TraderSpi, 92
OnQueryCreditExcessStock	OnQueryStrategy
XTP::API::TraderSpi, 74	DemoTestTraderSpi, 14
OnQueryCreditExtendDebtDateOrders	XTP::API::TraderSpi, 92
XTP::API::TraderSpi, 75	OnQueryStructuredFund
OnQueryCreditFundExtraInfo	XTP::API::TraderSpi, 93
XTP::API::TraderSpi, 75	OnQueryTrade
OnQueryCreditFundInfo	XTP::API::TraderSpi, 94
XTP::API::TraderSpi, 77	OnQueryTradeByPage
OnQueryCreditPositionExtraInfo	XTP::API::TraderSpi, 94
XTP::API::TraderSpi, 77	OnStrategyStateReport
OnQueryCreditTickerAssignInfo	DemoTestTraderSpi, 15
XTP::API::TraderSpi, 78	XTP::API::TraderSpi, 95
OnQueryCreditTickerDebtInfo	OnStrategySymbolStateReport
XTP::API::TraderSpi, 79	XTP::API::TraderSpi, 95
OnQueryETFBasket	OnTradeEvent
XTP::API::TraderSpi, 80	XTP::API::TraderSpi, 96
OnQueryETF	ord_type
XTP::API::TraderSpi, 79	XTPTickByTickEntrust, 155
OnQueryFundTransfer	order_no
XTP::API::TraderSpi, 80	XTPTickByTickEntrust, 155
OnQueryIPOInfoList	OrderBookStruct, 16
XTP::API::TraderSpi, 81	atu
OnQueryIPOQuotaInfo	qty
XTP::API::TraderSpi, 81	XTPTickByTickEntrust, 155
OnQueryMulCreditExcessStock	QueryAccountTradeMarket
XTP::API::TraderSpi, 82	XTP::API::TraderApi, 33 QueryAsset
OnQueryOptionAuctionInfo	XTP::API::TraderApi, 34
	ATTALT ITAUCIAPI, 34

QueryCreditAssetDebtInfo	QueryOptionCombinedUnfinishedOrders
XTP::API::TraderApi, 34	XTP::API::TraderApi, 49
QueryCreditCashRepayInfo	QueryOptionCombinedUnfinishedOrdersEx
XTP::API::TraderApi, 35	XTP::API::TraderApi, 50
QueryCreditDebtInfo	QueryOrderByXTPIDEx
XTP::API::TraderApi, 35	XTP::API::TraderApi, 51
QueryCreditExcessStock	QueryOrderByXTPID
	XTP::API::TraderApi, 50
XTP::API::TraderApi, 36	·
QueryCreditExtendDebtDateOrders	QueryOrders
XTP::API::TraderApi, 36	XTP::API::TraderApi, 51
QueryCreditFundExtraInfo	QueryOrdersByPage
XTP::API::TraderApi, 37	XTP::API::TraderApi, 52
QueryCreditFundInfo	QueryOrdersByPageEx
XTP::API::TraderApi, 37	XTP::API::TraderApi, 52
QueryCreditPositionExtraInfo	QueryOrdersEx
XTP::API::TraderApi, 37	XTP::API::TraderApi, 53
QueryCreditTickerAssignInfo	QueryOtherServerFund
XTP::API::TraderApi, 38	XTP::API::TraderApi, 53
QueryCreditTickerDebtInfo	QueryPosition
XTP::API::TraderApi, 38	XTP::API::TraderApi, 54
QueryETFTickerBasket	QueryStrategy
XTP::API::TraderApi, 40	XTP::API::TraderApi, 54
QueryETF	QueryStructuredFund
XTP::API::TraderApi, 40	XTP::API::TraderApi, 55
QueryFundTransfer	QueryTrades
	XTP::API::TraderApi, 56
XTP::API::TraderApi, 41	QueryTradesByPage
QueryIPOInfoList	XTP::API::TraderApi, 56
XTP::API::TraderApi, 41	QueryTradesByXTPID
QueryIPOQuotaInfo	XTP::API::TraderApi, 57
XTP::API::TraderApi, 41	QueryUnfinishedOrders
QueryMulCreditExcessStock	-
XTP::API::TraderApi, 42	XTP::API::TraderApi, 57
QueryOptionAuctionInfo	QueryUnfinishedOrdersEx
XTP::API::TraderApi, 42	XTP::API::TraderApi, 58
QueryOptionCombinedExecPosition	Pogistor Spi
XTP::API::TraderApi, 43	RegisterSpi
QueryOptionCombinedOrderByXTPIDEx	XTP::API::TraderApi, 58
XTP::API::TraderApi, 44	Release
QueryOptionCombinedOrderByXTPID	XTP::API::TraderApi, 58
XTP::API::TraderApi, 43	000
QueryOptionCombinedOrders	SEQ VIDIal Division 155
XTP::API::TraderApi, 44	XTPTickByTickEntrust, 155
QueryOptionCombinedOrdersByPage	XTPTickByTickStruct, 157
XTP::API::TraderApi, 45	XTPTickByTickTrade, 158
QueryOptionCombinedOrdersByPageEx	SetHeartBeatInterval
XTP::API::TraderApi, 45	XTP::API::TraderApi, 59
QueryOptionCombinedOrdersEx	SetSoftwareKey
- •	XTP::API::TraderApi, 59
XTP::API::TraderApi, 46	SetSoftwareVersion
QueryOptionCombinedPosition	XTP::API::TraderApi, 59
XTP::API::TraderApi, 46	side
QueryOptionCombinedStrategyInfo	XTPTickByTickEntrust, 156
XTP::API::TraderApi, 47	SubscribePublicTopic
QueryOptionCombinedTrades	XTP::API::TraderApi, 60
XTP::API::TraderApi, 48	
QueryOptionCombinedTradesByPage	trade_flag
XTP::API::TraderApi, 48	XTPTickByTickTrade, 158
QueryOptionCombinedTradesByXTPID	TraderApi, 17
XTP::API::TraderApi, 49	TraderSpi, 60

XTP::API::TraderApi	QueryOptionCombinedTradesByXTPID, 49
ALGOUserEstablishChannel, 19	QueryOptionCombinedUnfinishedOrders, 49
CancelAlgoOrder, 20	QueryOptionCombinedUnfinishedOrdersEx, 50
CancelOptionCombinedOrder, 20	QueryOrderByXTPIDEx, 51
CancelOrder, 21	QueryOrderByXTPID, 50
CreateTraderApi, 22	QueryOrders, 51
CreditCashRepay, 22	QueryOrdersByPage, 52
CreditCashRepayDebtInterestFee, 23	QueryOrdersByPageEx, 52
CreditExtendDebtDate, 23	QueryOrdersEx, 53
CreditSellStockRepayDebtInterestFee, 24	
• •	QueryOtherServerFund, 53
FundTransfer, 24	QueryPosition, 54
GetANewOrderXTPID, 26	QueryStrategy, 54
GetAccountByXTPID, 25	QueryStructuredFund, 55
GetAlgorithmIDByOrder, 25	QueryTrades, 56
GetApiLastError, 26	QueryTradesByPage, 56
GetApiVersion, 26	QueryTradesByXTPID, 57
GetClientIDByXTPID, 27	QueryUnfinishedOrders, 57
GetTradingDay, 27	QueryUnfinishedOrdersEx, 58
InsertAlgoOrder, 27	RegisterSpi, 58
InsertOptionCombinedOrder, 28	Release, 58
InsertOptionCombinedOrderExtra, 29	SetHeartBeatInterval, 59
InsertOrder, 29	SetSoftwareKey, 59
InsertOrderExtra, 30	SetSoftwareVersion, 59
IsServerRestart, 30	SubscribePublicTopic, 60
Login, 31	XTP::API::TraderSpi
LoginALGO, 32	OnALGOUserEstablishChannel, 63
Logout, 32	OnAlgoDisconnected, 63
ModifyUserTerminalInfo, 33	OnCancelAlgoOrder, 64
QueryAccountTradeMarket, 33	OnCancelOptionCombinedOrderError, 64
QueryAsset, 34	OnCancelOrderError, 65
QueryCreditAssetDebtInfo, 34	OnCreditCashRepay, 65
QueryCreditCashRepayInfo, 35	OnCreditCashRepayDebtInterestFee, 66
QueryCreditDebtInfo, 35	OnCreditExtendDebtDate, 66
QueryCreditExcessStock, 36	OnDisconnected, 67
QueryCreditExtendDebtDateOrders, 36	OnError, 67
QueryCreditFundExtraInfo, 37	OnFundTransfer, 68
QueryCreditFundInfo, 37	OnInsertAlgoOrder, 68
QueryCreditPositionExtraInfo, 37	OnOptionCombinedOrderEvent, 69
QueryCreditTickerAssignInfo, 38	OnOptionCombinedTradeEvent, 69
QueryCreditTickerDebtInfo, 38	OnOrderEvent, 71
QueryETFTickerBasket, 40	OnQueryAccountTradeMarket, 71
QueryETF, 40	OnQueryAsset, 72
QueryFundTransfer, 41	•
·	OnQueryCreditAssetDebtInfo, 73 OnQueryCreditCashRepayInfo, 73
QueryIPOCuetalafa 41	• • • • • • • • • • • • • • • • • • • •
QueryIPOQuotaInfo, 41	OnQueryCreditDebtInfo, 74
QueryMulCreditExcessStock, 42	OnQueryCreditExcessStock, 74
QueryOptionAuctionInfo, 42	OnQueryCreditExtendDebtDateOrders, 75
QueryOptionCombinedExecPosition, 43	OnQueryCreditFundExtraInfo, 75
QueryOptionCombinedOrderByXTPIDEx, 44	OnQueryCreditFundInfo, 77
QueryOptionCombinedOrderByXTPID, 43	OnQueryCreditPositionExtraInfo, 77
QueryOptionCombinedOrders, 44	OnQueryCreditTickerAssignInfo, 78
QueryOptionCombinedOrdersByPage, 45	OnQueryCreditTickerDebtInfo, 79
QueryOptionCombinedOrdersByPageEx, 45	OnQueryETFBasket, 80
QueryOptionCombinedOrdersEx, 46	OnQueryETF, 79
QueryOptionCombinedPosition, 46	OnQueryFundTransfer, 80
QueryOptionCombinedStrategyInfo, 47	OnQueryIPOInfoList, 81
QueryOptionCombinedTrades, 48	OnQueryIPOQuotaInfo, 81
QueryOptionCombinedTradesByPage, 48	OnQueryMulCreditExcessStock, 82

	OnQueryOptionAuctionInfo, 82	xtp_api_data_type.h, 190
	OnQueryOptionCombinedExecPosition, 83	XTP_ORDER_STATUS_TYPE
	OnQueryOptionCombinedOrders, 84	xtp_api_data_type.h, 190
	OnQueryOptionCombinedOrdersByPage, 84	XTP_ORDER_SUBMIT_STATUS_TYPE
	OnQueryOptionCombinedOrdersByPageEx, 85	xtp_api_data_type.h, 191
	OnQueryOptionCombinedOrdersEx, 86	XTP_POSITION_DIRECTION_TYPE
	OnQueryOptionCombinedPosition, 86	xtp_api_data_type.h, 191
	OnQueryOptionCombinedStrategyInfo, 87	XTP_POSITION_SECURITY_TYPE
	OnQueryOptionCombinedTrades, 87	xtp_api_data_type.h, 191
	OnQueryOptionCombinedTradesByPage, 88	XTP PRICE TYPE
	OnQueryOrder, 89	xtp_api_data_type.h, 192
	OnQueryOrderByPage, 89	XTP_PROTOCOL_TYPE
	OnQueryOrderByPageEx, 90	xtp_api_data_type.h, 192
	OnQueryOrderEx, 91	XTP_QUALIFICATION_TYPE
	OnQueryOtherServerFund, 91	xtp_api_data_type.h, 193
	OnQueryPosition, 92	XTP_QUOTE_REBUILD_DATA_TYPE
	OnQueryStrategy, 92	xtp_api_data_type.h, 193
	OnQueryStructuredFund, 93	XTP_REBUILD_RET_TYPE
	OnQueryTrade, 94	xtp_api_data_type.h, 193
	OnQueryTradeByPage, 94	XTP_SECURITY_STATUS
	· · · · ·	
	OnStrategyStateReport, 95	xtp_api_data_type.h, 194
	OnStrategySymbolStateReport, 95	XTP_SECURITY_TYPE
/TD	OnTradeEvent, 96	xtp_api_data_type.h, 194
KIP	_ACCOUNT_TYPE	XTP_SIDE_STOCK_REPAY_STOCK
/TD	xtp_api_data_type.h, 183	xtp_api_data_type.h, 182
KIP	_AUTO_SPLIT_TYPE	XTP_SPLIT_MERGE_STATUS
(TD	xtp_api_data_type.h, 183	xtp_api_data_type.h, 195
KIP	_BUSINESS_TYPE	XTP_TBT_TYPE
	xtp_api_data_type.h, 184	xtp_api_data_type.h, 195
KTP	_CRD_CR_STATUS	XTP_TE_RESUME_TYPE
	xtp_api_data_type.h, 184	xtp_api_data_type.h, 195
KTP	_DEBT_EXTEND_OPER_STATUS	XTP_TICKER_TYPE
	xtp_api_data_type.h, 186	xtp_api_data_type.h, 196
KTP	_EXCHANGE_TYPE	XTP_UNDERLYING_TYPE
	xtp_api_data_type.h, 186	xtp_api_data_type.h, 196
KTP	_EXPIRE_DATE_TYPE	XTPClientQueryCrdDebtStockReq, 96
	xtp_api_data_type.h, 186	XTPClientQueryCrdPositionStkInfo, 97
KTP	_FUND_OPER_STATUS	XTPClientQueryCrdPositionStockReq, 98
	xtp_api_data_type.h, 187	XTPClientQueryCrdSurplusStkReqInfo, 98
KTP	_FUND_QUERY_TYPE	XTPClientQueryCrdSurplusStkRspInfo, 99
	xtp_api_data_type.h, 187	XTPCombLegStrategy, 99
KTP	_FUND_TRANSFER_TYPE	XTPCrdCashRepayDebtInterestFeeRsp, 100
	xtp_api_data_type.h, 187	XTPCrdCashRepayInfo, 100
KTP	_LOG_LEVEL	XTPCrdCashRepayRsp, 101
	xtp_api_data_type.h, 188	XTPCrdDebtInfo, 102
KTP	_MARKET_TYPE	XTPCrdDebtStockInfo, 103
	xtp_api_data_type.h, 188	XTPCrdFundExtraInfo, 103
KTP	_OPT_CALL_OR_PUT_TYPE	XTPCrdFundInfo, 104
	xtp_api_data_type.h, 188	XTPCrdPositionExtraInfo, 104
KTP	OPT_COVERED_OR_UNCOVERED	XTPCreditDebtExtendNotice, 105
	xtp_api_data_type.h, 189	XTPCreditDebtExtendReq, 105
KTP	OPT EXERCISE TYPE TYPE	XTPFundQueryReq, 106
	xtp_api_data_type.h, 189	XTPFundQueryRsp, 107
ΚΤР	_OPT_POSITION_TYPE	XTPFundTransferNotice, 107
	xtp_api_data_type.h, 189	XTPFundTransferReg, 108
(TP	ORDER_ACTION_STATUS_TYPE	XTPMarketDataBondExData, 108
•••	xtp_api_data_type.h, 190	XTPMarketDataOptionExData, 110
(TP	ORDER DETAIL TYPE	XTPMarketDataStockExData, 110
VIII.		ATT MarketbalactionExbala, 110

XTPMarketDataStruct, 112	xtp_api_data_type.h, 196
XTPOptCombLegInfo, 114	XTPTickByTickEntrust, 155
XTPOptCombOrderInfo, 114	ord_type, 155
market, 115	order_no, 155
XTPOptCombOrderInfoEx, 116	qty, 155
market, 117	seq, 155
XTPOptCombOrderInsertInfo, 117	side, 156
XTPOptCombPlugin, 118	XTPTickByTickStatus, 156
XTPOptCombTradeReport, 118	XTPTickByTickStruct, 156
XTPOrderCancelInfo, 119	seq, 157
XTPOrderInfo, 120	XTPTickByTickTrade, 157
XTPOrderInfoEx, 121	seq, 158
XTPOrderInsertInfo, 123	trade_flag, 158
XTPQueryAssetRsp, 124	XTPTickerPriceInfo, 158
XTPQueryCombineStrategyInfoRsp, 126	XTPTradeReport, 159
XTPQueryETFBaseReq, 126	XTPUserTerminalInfoReq, 160
XTPQueryETFBaseRsp, 127	xoms_api_fund_struct.h, 168
XTPQueryETFComponentReq, 128	xoms_api_struct.h, 168
XTPQueryETFComponentRsp, 128	xquote_api_rebuild_tbt_struct.h, 172
XTPQueryETFComponentRspV1, 129	XTPQuoteRebuildReq, 173
XTPQueryFundTransferLogReq, 129	xquote_api_struct.h, 173
XTPQueryIPOQuotaRsp, 130	xtp_api_data_type.h, 175
XTPQueryIPOQuotaRspV1, 130	ETF_REPLACE_TYPE, 183
XTPQueryIPOTickerRsp, 131	XTP ACCOUNT TYPE, 183
XTPQueryOptCombExecPosReq, 132	XTP_AUTO_SPLIT_TYPE, 183
XTPQueryOptCombExecPosRsp, 132	XTP_BUSINESS_TYPE, 184
XTPQueryOptCombOrderByPageReq, 133	XTP_CRD_CR_STATUS, 184
XTPQueryOptCombOrderReq, 134	XTP_DEBT_EXTEND_OPER_STATUS, 186
XTPQueryOptCombPositionReq, 134	XTP_EXCHANGE_TYPE, 186
XTPQueryOptCombPositionRsp, 135	XTP_EXPIRE_DATE_TYPE, 186
XTPQueryOptCombReportByExecldReq, 135	XTP_FUND_OPER_STATUS, 187
XTPQueryOptCombTraderByPageReq, 136	XTP FUND QUERY TYPE, 187
XTPQueryOptCombTraderReq, 136	XTP FUND TRANSFER TYPE, 187
XTPQueryOptExecInfoRsp, 137	XTP_LOG_LEVEL, 188
XTPQueryOptionAuctionInfoReq, 138	XTP MARKET TYPE, 188
XTPQueryOptionAuctionInfoRsp, 138	XTP_OPT_CALL_OR_PUT_TYPE, 188
XTPQueryOrderByPageReq, 140	XTP_OPT_COVERED_OR_UNCOVERED, 189
XTPQueryOrderReq, 141	XTP_OPT_EXERCISE_TYPE_TYPE, 189
XTPQueryReportByExecIdReq, 141	XTP_OPT_POSITION_TYPE, 189
XTPQueryStkPositionReq, 142	XTP ORDER ACTION STATUS TYPE, 190
XTPQueryStkPositionRsp, 142	XTP ORDER DETAIL TYPE, 190
XTPQueryStructuredFundInfoReq, 144	XTP_ORDER_STATUS_TYPE, 190
XTPQueryTraderByPageReq, 144	XTP_ORDER_SUBMIT_STATUS_TYPE, 191
XTPQueryTraderReg, 145	XTP_POSITION_DIRECTION_TYPE, 191
XTPQuoteFullInfo, 145	XTP POSITION SECURITY TYPE, 191
XTPQuoteRebuildReq, 147	XTP PRICE TYPE, 192
xquote_api_rebuild_tbt_struct.h, 173	XTP_PROTOCOL_TYPE, 192
XTPQuoteRebuildResultRsp, 147	XTP QUALIFICATION TYPE, 193
XTPQuoteStaticInfo, 148	XTP_QUOTE_REBUILD_DATA_TYPE, 193
XTPRspInfoStruct, 149	XTP_REBUILD_RET_TYPE, 193
XTPSpecificTickerStruct, 149	XTP_SECURITY_STATUS, 194
XTPStrategyInfoStruct, 150	XTP SECURITY TYPE, 194
XTPStrategyStateReportStruct, 150	XTP_SIDE_STOCK_REPAY_STOCK, 182
XTPStrategySymbolInfoStruct, 151	XTP_SPLIT_MERGE_STATUS, 195
XTPStrategySymbolReqStruct, 152	XTP_TBT_TYPE, 195
XTPStrategySymbolStateReportStruct, 152	XTP_TE_RESUME_TYPE, 195
XTPStructuredFundInfo, 154	XTP_TICKER_TYPE, 196
XTPTerminalType	XTP_UNDERLYING_TYPE, 196

```
XTPTerminalType, 196
xtp_api_struct.h, 197
xtp_api_struct_common.h, 197
xtp_trader_api.h, 198
```