Trading API in C++

Class Reference

Version 1.0

Contents

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

Module Documentation

3.1 General functionality of Questrade API

Classes

- struct QuestradeAPI::AuthenticationInfo

  Authentication Information.
- struct QuestradeAPI::DateTime

  Structure for date time.

Functions

- QUESTRADELIBRARYAPI
  AuthenticationInfo APIENTRY QuestradeAPI::Authenticate (const std::string &refreshToken, bool isDemo)

  Authenticates with provided parameters.
- QUESTRADELIBRARYAPI void APIENTRY QuestradeAPI::Init ()

  Initialize library.
- QUESTRADELIBRARYAPI void APIENTRY QuestradeAPI::UnInit ()

  Uninitialize library.

3.1.1 Detailed Description

3.1.2 Function Documentation

3.1.2.1 QUESTRADELIBRARYAPI AuthenticationInfo APIENTRY QuestradeAPI::Authenticate (const std::string & *refreshToken,* bool *isDemo*)

Authenticates with provided parameters.

clientId - Client ID of the Application using DLL

refreshToken - Refresh token which user of the application got from Questrade site

isDemo - flag whether this is Live token or Demo (Practice) token

3.2 Questrade API Enums

Classes

- struct QuestradeAPI::AccountStatus

  AccountStatus enumeration.
- struct QuestradeAPI::ClientAccountType

  Types of clients on account.
- struct QuestradeAPI::OrderSide

  ClientOrderSide aka Action.
- struct QuestradeAPI::CurrencyType

  CurrencyType enumeration.
- struct QuestradeAPI::ExerciseType

  Option exercise type.
- struct QuestradeAPI::CandlesGranularity

  Possible granularities of historical data, required for charting.
- struct QuestradeAPI::OptionDurationType

  Duration types of options.
- struct QuestradeAPI::OptionType

  Option types.
- struct QuestradeAPI::OrderClass

  For bracket order describes the type of component.
- struct QuestradeAPI::OrderAction

  Action for inserted order.
- struct QuestradeAPI::OrderSource

  Platform from which order was placed.
- struct QuestradeAPI::OrderState

  State of the order.
- struct QuestradeAPI::OrderStateFilterTypes

  Types used for filtering orders by state.
- struct QuestradeAPI::OrderTimeInForce

  Time in force aka Duration of the order.
- struct QuestradeAPI::OrderType

  Type of the order.
- struct QuestradeAPI::SecurityType

  Basic types of securities.
- struct QuestradeAPI::StrategyType

  Strategy type for multi-leg order.
- struct QuestradeAPI::TickType

  Tick types for last trade of the quotes.
- struct QuestradeAPI::UserAccountType

  Types of accounts.

Enumerations

- enum QuestradeAPI::AccountStatus::Value {
  Undefined = -1, **UnAllocated** = 0, **Active** = 1, **SuspendedClosed** = 2,
  SuspendedViewOnly = 3, **LiquidateOnly** = 4, **Closed** = 5, **Count** }

  Values for AccountStatus.

- enum QuestradeAPI::ClientAccountType::Value {
  Undefined = 0, **Individual** = 1, **Joint** = 2, **InformalTrust** = 3,
  Corporation = 4, **InvestmentClub** = 5, **FormalTrust** = 6, **Partnership** = 7,
  SoleProprietorship = 8, **Family** = 9, **JointAndInformalTrust** = 10, **Institution** = 11,
  Count }

  Values for ClientAccountType.
- enum QuestradeAPI::OrderSide::Value {
  Undefined = 0, **Buy** = 1, **Sell** = 2, **Short** = 3,
  Cov = 4, **BTO** = 5, **STC** = 6, **STO** = 7,
  BTC = 8, **Count** }

  Values for OrderSide.
- enum QuestradeAPI::CurrencyType::Value { **Undefined** = 0, **CAD** = 22, **USD** = 99, **Count** }

  Values for CurrencyType.
- enum QuestradeAPI::ExerciseType::Value { **Undefined** = 0, **American** = 1, **European** = 2, **Count** }

  Values for ExerciseType.
- enum QuestradeAPI::CandlesGranularity::Value {
  Undefined = 0, **OneMinute** = 1, **TwoMinutes** = 2, **ThreeMinutes** = 3,
  FourMinutes = 4, **FiveMinutes** = 5, **TenMinutes** = 6, **FifteenMinutes** = 7,
  TwentyMinutes = 8, **HalfHour** = 9, **OneHour** = 10, **TwoHours** = 11,
  FourHours = 12, **OneDay** = 13, **OneWeek** = 14, **OneMonth** = 15,
  ThreeMonths = 16, **OneYear** = 17, **Count** }

  Values for CandlesGranularity.
- enum QuestradeAPI::OptionDurationType::Value {
  Undefined = 0, **Weekly** = 1, **Monthly** = 2, **Quarterly** = 3,
  LEAP = 4, **Count** }

  Values for OptionDurationType.
- enum QuestradeAPI::OptionType::Value { **Undefined** = 0, **Call** = 1, **Put** = 2, **Count** }

  Values for OptionType.
- enum QuestradeAPI::OrderClass::Value {
  Undefined = 0, **Primary** = 1, **Limit** = 2, **StopLoss** = 3,
  Count }

  Values for OrderClass.
- enum QuestradeAPI::OrderAction::Value { **Undefined** = 0, **Buy** = 1, **Sell** = 2, **Count** }

  Values for OrderAction.
- enum QuestradeAPI::OrderSource::Value {
  Undefined = 0, **Automatic** = 1, **BackOfficeAgent** = 2, **QuestradeIQ** = 3,
  QuestradeIQEdge = 4, **QuestradeIQMobile** = 5, **QuestradeIQEssential** = 6, **TradeDesk** = 7,
  QuestradeIQMobileWeb = 8, **TradingAPI** = 9, **Count** }

  Values for OrderSource.
- enum QuestradeAPI::OrderState::Value {
  Undefined = 0, **Failed** = 1, **Pending** = 2, **Accepted** = 3,
  Rejected = 4, **CancelPending** = 5, **Canceled** = 6, **PartialCanceled** = 7,
  Partial = 8, **Executed** = 9, **ReplacePending** = 10, **Replaced** = 11,
  Stopped = 12, **Suspended** = 13, **Expired** = 14, **Queued** = 15,
  Triggered = 16, **Activated** = 17, **PendingRiskReview** = 18, **ContingentOrder** = 19,
  Count }

  Values for OrderState.
- enum QuestradeAPI::OrderStateFilterTypes::Value {
  Undefined = 0, **All** = 1, **Open** = 2, **Closed** = 3,
  Count }

  Values for OrderStateFilterTypes.
- enum QuestradeAPI::OrderTimeInForce::Value {
  Undefined = 0, **Day** = 1, **GoodTillCanceled** = 2, **GoodTillExtendedDay** = 3,
  GoodTillDate = 4, **ImmediateOrCancel** = 5, **FillOrKill** = 6, **Count** }

Values for OrderTimeInForce.

- enum QuestradeAPI::OrderType::Value {
  Undefined = 0, **Market** = 1, **Limit** = 2, **Stop** = 3,
  StopLimit = 4, **TrailStopInPercentage** = 5, **TrailStopInDollar** = 6, **TrailStopLimitInPercentage** = 7,
  TrailStopLimitInDollar = 8, **LimitOnOpen** = 9, **LimitOnClose** = 10, **Count** }

  Values for OrderType.
- enum QuestradeAPI::SecurityType::Value {
  Undefined = 0, **Stock** = 1, **Option** = 2, **Bond** = 3,
  Right = 4, **Commodity** = 5, **MutualFund** = 6, **Index** = 7,
  Currency = 8, **Count** }

  Values for SecurityType.
- enum QuestradeAPI::StrategyType::Value {
  Undefined = 0, **CallsPuts** = 1, **Calls** = 2, **Puts** = 3,
  CoveredCall = 4, **MarriedPuts** = 5, **VerticalCallSpread** = 6, **VerticalPutSpread** = 7,
  CalendarCallSpread = 8, **CalendarPutSpread** = 9, **DiagonalCallSpread** = 10, **DiagonalPutSpread** = 11,
  Collar = 12, **Straddle** = 13, **Strangle** = 14, **ButterflyCall** = 15,
  ButterflyPut = 16, **IronButterfly** = 17, **CondorCall** = 18, **CondorPut** = 19,
  IronCondor = 20, **Custom** = 21, **SingleLeg** = 22, **Count** }

  Values for StrategyType.
- enum QuestradeAPI::TickType::Value {
  Undefined = 0, **Up** = 1, **Equal** = 2, **Down** = 3,
  Count }

  Values for TickType.
- enum QuestradeAPI::UserAccountType::Value {
  Undefined = 0, **Cash** = 1, **Margin** = 2, **TFSA** = 3,
  RRSP = 4, **SRRSP** = 5, **LRRSP** = 6, **LIRA** = 7,
  LIF = 8, **RIF** = 9, **SRIF** = 10, **LRIF** = 11,
  RRIF = 12, **PRIF** = 13, **RESP** = 14, **FRESP** = 15,
  FX = 16, **FXD** = 17, **Count** }

  Values for UserAccountType.

Functions

- QUESTRADELIBRARYAPI
  OrderAction::Value APIENTRY QuestradeAPI::convertSide (OrderSide::Value side)

  Converts OrderSide to corresponding OrderAction.

3.2.1 Detailed Description

3.3 Questrade API Structures

Classes

- struct QuestradeAPI::Level1DataItem

  Level 1 quote data.
- struct QuestradeAPI::EquitySymbol

  Data for equities returned by symbol search.
- struct QuestradeAPI::CandleData

  Historical Candle Data.
- struct QuestradeAPI::InsertOrderRequest

  Structure that used as input in some services.
- struct QuestradeAPI::OrderLegData

  Describes leg for multi-leg order.
- struct QuestradeAPI::OrderData

  Describes order details.
- struct QuestradeAPI::ReplaceOrderRequest

  Structure that used as input in some services.
- struct QuestradeAPI::PositionData

  Position Data.
- struct QuestradeAPI::AccountData

  Account Data.
- struct QuestradeAPI::ChainPerStrikePrice

  Pair of call and put options with same expiry date, root and strike.
- struct QuestradeAPI::ChainPerRoot

  Option chain grouped per option root symbol for same expiry date.
- struct QuestradeAPI::ChainPerExpiryDate

  Option chain grouped per expiry date.
- struct QuestradeAPI::ExecutionData

  Execution Data.
- struct QuestradeAPI::BalanceData

  Balance Data.
- struct QuestradeAPI::Market

  Listing exchange information.
- struct QuestradeAPI::UnderlyingMultiplierPair

  Underlying information for the option.
- struct QuestradeAPI::ContractDeliverables

  Deliverables of option contract.
- struct QuestradeAPI::MinTickData

  Min tick rules data.
- struct QuestradeAPI::SymbolData

  Symbol Data.

Functions

- QUESTRADELIBRARYAPI InsertOrderRequest APIENTRY QuestradeAPI::createCopyOfOrderRequest (const std::string &account, OrderData order)

  Converts OrderData to make a copy of the order.
- QUESTRADELIBRARYAPI ReplaceOrderRequest APIENTRY QuestradeAPI::createReplaceRequest (const std::string &account, OrderData origOrder)

  Converts OrderData to make a replace request for the order.

3.3.1 Detailed Description

3.4 Questrade API Services

Classes

- struct QuestradeAPI::CancelOrderResponse

  CancelOrder Response structure.
- struct QuestradeAPI::GetAccountsResponse

  GetAccounts Response structure.
- struct QuestradeAPI::GetBalancesResponse

  GetBalances Response structure.
- struct QuestradeAPI::GetCandlesResponse

  GetCandles Response structure.
- struct QuestradeAPI::GetExecutionsResponse

  GetExecutions Response structure.
- struct QuestradeAPI::GetMarketsResponse

  GetMarkets Response structure.
- struct QuestradeAPI::GetOptionsResponse

  GetOptions Response structure.
- struct QuestradeAPI::GetOrdersResponse

  GetOrders Response structure.
- struct QuestradeAPI::GetPositionsResponse

  GetPositions Response structure.
- struct QuestradeAPI::GetQuoteResponse

  GetQuote Response structure.
- struct QuestradeAPI::GetServerTimeResponse

  GetServerTime Response structure.
- struct QuestradeAPI::GetSymbolsResponse

  GetSymbols Response structure.
- struct QuestradeAPI::InsertOrderResponse

  InsertOrder Response structure.
- struct QuestradeAPI::InsertOrderImpactResponse

  InsertOrderImpact Response structure.
- struct QuestradeAPI::ReplaceOrderResponse

  ReplaceOrder Response structure.
- struct QuestradeAPI::ReplaceOrderImpactResponse

  ReplaceOrderImpact Response structure.
- struct QuestradeAPI::SearchSymbolsResponse

  SearchSymbols Response structure.

Functions

- QUESTRADELIBRARYAPI CancelOrderResponse APIENTRY QuestradeAPI::CancelOrder (AuthenticationInfo &authInfo, const std::string &accountNumber, uint64 orderId)

  Cancel order.
- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::CancelOrderAsync (AuthenticationInfo &authInfo, std::function< void(CancelOrderResponse)> callback, int32 requestId, const std::string &accountNumber, uint64 orderId)

  Asynchronous execution of CancelOrder.
- QUESTRADELIBRARYAPI GetAccountsResponse APIENTRY QuestradeAPI::GetAccounts (AuthenticationInfo &authInfo)

Returns user accounts.

- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::GetAccountsAsync (AuthenticationInfo &authInfo, std::function< void(GetAccountsResponse)> callback, int32 requestId)

  Asynchronous execution of GetAccounts.
- QUESTRADELIBRARYAPI GetBalancesResponse APIENTRY QuestradeAPI::GetBalances (AuthenticationInfo &authInfo, const std::string &accountNumber)

  GetBalance.
- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::GetBalancesAsync (AuthenticationInfo &authInfo, std::function< void(GetBalancesResponse)> callback, int32 requestId, const std::string &accountNumber)

  Asynchronous execution of GetBalances.
- QUESTRADELIBRARYAPI GetCandlesResponse APIENTRY QuestradeAPI::GetCandles (AuthenticationInfo &authInfo, uint64 symbolId, const DateTime &startTime, const DateTime &endTime, CandlesGranularity::Value interval)

  Get historical candles for the symbol.
- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::GetCandlesAsync (AuthenticationInfo &authInfo, std::function< void(GetCandlesResponse)> callback, int32 requestId, uint64 symbolId, const DateTime &startTime, const DateTime &endTime, CandlesGranularity::Value interval)

  Asynchronous execution of GetCandles.
- QUESTRADELIBRARYAPI GetExecutionsResponse APIENTRY QuestradeAPI::GetExecutions (AuthenticationInfo &authInfo, const std::string &accountNumber, const DateTime &startTime=DateTime::startOfDay(), const DateTime &endTime=DateTime())

  Returns account executions.
- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::GetExecutionsAsync (AuthenticationInfo &authInfo, std::function< void(GetExecutionsResponse)> callback, int32 requestId, const std::string &accountNumber, const DateTime &startTime=DateTime::startOfDay(), const DateTime &endTime=DateTime())

  Asynchronous execution of GetExecutions.
- QUESTRADELIBRARYAPI GetMarketsResponse APIENTRY QuestradeAPI::GetMarkets (AuthenticationInfo &authInfo)

  Returns listings exchange info.
- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::GetMarketsAsync (AuthenticationInfo &authInfo, std::function< void(GetMarketsResponse)> callback, int32 requestId)

  Asynchronous execution of GetMarkets.
- QUESTRADELIBRARYAPI GetOptionsResponse APIENTRY QuestradeAPI::GetOptions (AuthenticationInfo &authInfo, uint64 symbolId)

  Returns option chain for the underlying symbol.
- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::GetOptionsAsync (AuthenticationInfo &authInfo, std::function< void(GetOptionsResponse)> callback, int32 requestId, uint64 symbolId)

  Asynchronous execution of GetOptions.
- QUESTRADELIBRARYAPI GetOrdersResponse APIENTRY QuestradeAPI::GetOrders (AuthenticationInfo &authInfo, const std::string &accountNumber, OrderStateFilterTypes::Value stateFilter=OrderStateFilterTypes::All, const DateTime &startTime=DateTime::startOfDay(), const DateTime &endTime=DateTime())

  Returns account orders.
- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::GetOrdersAsync (AuthenticationInfo &authInfo, std::function< void(GetOrdersResponse)> callback, int32 requestId, const std::string &accountNumber, OrderStateFilterTypes::Value stateFilter=OrderStateFilterTypes::All, const DateTime &startTime=DateTime::startOfDay(), const DateTime &endTime=DateTime())

  Asynchronous execution of GetOrders.
- QUESTRADELIBRARYAPI GetOrdersResponse APIENTRY QuestradeAPI::GetOrdersByID (AuthenticationInfo &authInfo, const std::string &accountNumber, const std::vector< uint64 > &ids)

  Returns orders by ids.

- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::GetOrdersByIDAsync (AuthenticationInfo &auth↩ Info, std::function< void(GetOrdersResponse)> callback, int32 requestId, const std::string &accountNumber, const std::vector< uint64 > &ids)

  Asynchronous execution of GetOrdersByID.
- QUESTRADELIBRARYAPI GetPositionsResponse APIENTRY QuestradeAPI::GetPositions (AuthenticationInfo &authInfo, const std↩ ::string &accountNumber)

  Returns account positions.
- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::GetPositionsAsync (AuthenticationInfo &authInfo, std::function< void(GetPositionsResponse)> callback, int32 requestId, const std::string &accountNumber)

  Asynchronous execution of GetPositions.
- QUESTRADELIBRARYAPI GetQuoteResponse APIENTRY QuestradeAPI::GetQuote (AuthenticationInfo &authInfo, const std::vector< uint64 > &ids)

  Returns L1 data.
- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::GetQuoteAsync (AuthenticationInfo &authInfo, std::function< void(GetQuoteResponse)> callback, int32 requestId, const std::vector< uint64 > &ids)

  Asynchronous execution of GetQuote.
- QUESTRADELIBRARYAPI GetServerTimeResponse APIENTRY QuestradeAPI::GetServerTime (AuthenticationInfo &authInfo)

  Returns server time.
- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::GetServerTimeAsync (AuthenticationInfo &auth↩ Info, std::function< void(GetServerTimeResponse)> callback, int32 requestId)

  Asynchronous execution of GetServerTime.
- QUESTRADELIBRARYAPI GetSymbolsResponse APIENTRY QuestradeAPI::GetSymbols (AuthenticationInfo &authInfo, const std↩ ::vector< uint64 > &ids)

  Returns symbols data.
- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::GetSymbolsAsync (AuthenticationInfo &authInfo, std::function< void(GetSymbolsResponse)> callback, int32 requestId, const std::vector< uint64 > &ids)

  Asynchronous execution of GetSymbols.
- QUESTRADELIBRARYAPI InsertOrderResponse APIENTRY QuestradeAPI::InsertOrder (AuthenticationInfo &authInfo, InsertOrder↩ Request request)

  Insert order.
- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::InsertOrderAsync (AuthenticationInfo &authInfo, std::function< void(InsertOrderResponse)> callback, int32 requestId, InsertOrderRequest request)

  Asynchronous execution of InsertOrder.
- QUESTRADELIBRARYAPI InsertOrderImpactResponse APIENTRY QuestradeAPI::InsertOrderImpact (AuthenticationInfo &authInfo, InsertOrderRequest request)

  Impact of inserting the order.
- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::InsertOrderImpactAsync (AuthenticationInfo &authInfo, std::function< void(InsertOrderImpactResponse)> callback, int32 requestId, InsertOrderRequest request)

  Asynchronous execution of InsertOrderImpact.
- QUESTRADELIBRARYAPI ReplaceOrderResponse APIENTRY QuestradeAPI::ReplaceOrder (AuthenticationInfo &authInfo, Replace↩ OrderRequest request)

  Replace order.
- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::ReplaceOrderAsync (AuthenticationInfo &authInfo, std::function< void(ReplaceOrderResponse)> callback, int32 requestId, ReplaceOrderRequest request)

  Asynchronous execution of ReplaceOrder.

- QUESTRADELIBRARYAPI
  ReplaceOrderImpactResponse
  APIENTRY QuestradeAPI::ReplaceOrderImpact (AuthenticationInfo &authInfo, ReplaceOrderRequest request)

  Impact of replacing the order.
- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::ReplaceOrderImpactAsync (AuthenticationInfo &authInfo, std::function< void(ReplaceOrderImpactResponse)> callback, int32 requestId, ReplaceOrderRequest request)

  Asynchronous execution of ReplaceOrderImpact.
- QUESTRADELIBRARYAPI
  SearchSymbolsResponse APIENTRY QuestradeAPI::SearchSymbols (AuthenticationInfo &authInfo, const std::string &prefix, uint64 offset=0, uint64 limit=20)

  Search symbols by prefix.
- QUESTRADELIBRARYAPI int APIENTRY QuestradeAPI::SearchSymbolsAsync (AuthenticationInfo &authInfo, std::function< void(SearchSymbolsResponse)> callback, int32 requestId, const std::string &prefix, uint64 offset=0, uint64 limit=20)

  Asynchronous execution of SearchSymbols.

3.4.1 Detailed Description

3.4.2 Function Documentation

3.4.2.1 QUESTRADELIBRARYAPI CancelOrderResponse APIENTRY QuestradeAPI::CancelOrder (AuthenticationInfo & *authInfo,* const std::string & *accountNumber,* uint64 *orderId*)

Cancel order.

authInfo - AuthenticationInfo object with which this service should be called.

accountNumber - Account number

orderId - Order ID

3.4.2.2 QUESTRADELIBRARYAPI GetAccountsResponse APIENTRY QuestradeAPI::GetAccounts (AuthenticationInfo & *authInfo*)

Returns user accounts.

authInfo - AuthenticationInfo object with which this service should be called.

3.4.2.3 QUESTRADELIBRARYAPI GetBalancesResponse APIENTRY QuestradeAPI::GetBalances (AuthenticationInfo & *authInfo,* const std::string & *accountNumber*)

GetBalance.

authInfo - AuthenticationInfo object with which this service should be called.

accountNumber - Account number

3.4.2.4 QUESTRADELIBRARYAPI GetCandlesResponse APIENTRY QuestradeAPI::GetCandles (AuthenticationInfo & *authInfo,* uint64 *symbolId,* const DateTime & *startTime,* const DateTime & *endTime,* CandlesGranularity::Value *interval*)

Get historical candles for the symbol.

authInfo - AuthenticationInfo object with which this service should be called.

symbolId - The symbol ID

startTime - The start time

endTime - The end time

interval - The interval

3.4.2.5 QUESTRADELIBRARYAPI GetExecutionsResponse APIENTRY QuestradeAPI::GetExecutions (AuthenticationInfo & *authInfo,* const std::string & *accountNumber,* const DateTime & *startTime* = `DateTime::startOfDay()`, const DateTime & *endTime* = `DateTime()`)

Returns account executions.

authInfo - AuthenticationInfo object with which this service should be called.

accountNumber - Account number

startTime - Start time

endTime - End time

3.4.2.6 QUESTRADELIBRARYAPI GetMarketsResponse APIENTRY QuestradeAPI::GetMarkets (AuthenticationInfo & *authInfo*)

Returns listings exchange info.

authInfo - AuthenticationInfo object with which this service should be called.

3.4.2.7 QUESTRADELIBRARYAPI GetOptionsResponse APIENTRY QuestradeAPI::GetOptions (AuthenticationInfo & *authInfo,* uint64 *symbolId*)

Returns option chain for the underlying symbol.

authInfo - AuthenticationInfo object with which this service should be called.

symbolId - The symbol ID

3.4.2.8 QUESTRADELIBRARYAPI GetOrdersResponse APIENTRY QuestradeAPI::GetOrders (AuthenticationInfo & *authInfo,* const std::string & *accountNumber,* OrderStateFilterTypes::Value *stateFilter* = `OrderStateFilterTypes::All`, const DateTime & *startTime* = `DateTime::startOfDay()`, const DateTime & *endTime* = `DateTime()`)

Returns account orders.

authInfo - AuthenticationInfo object with which this service should be called.

accountNumber - The account number

stateFilter - The state of the orders to return

startTime - The start time of the interval which to return orders for

endTime - The end time of the interval which to return orders for

3.4.2.9 QUESTRADELIBRARYAPI GetOrdersResponse APIENTRY QuestradeAPI::GetOrdersByID (AuthenticationInfo & *authInfo,* const std::string & *accountNumber,* const std::vector< uint64 > & *ids*)

Returns orders by ids.

authInfo - AuthenticationInfo object with which this service should be called.

accountNumber - The account number

ids - Input array of order IDs

3.4.2.10 QUESTRADELIBRARYAPI GetPositionsResponse APIENTRY QuestradeAPI::GetPositions (AuthenticationInfo & *authInfo,* const std::string & *accountNumber*)

Returns account positions.

authInfo - AuthenticationInfo object with which this service should be called.

accountNumber - Account number

3.4.2.11 QUESTRADELIBRARYAPI GetQuoteResponse APIENTRY QuestradeAPI::GetQuote (AuthenticationInfo & *authInfo,* const std::vector< uint64 > & *ids*)

Returns L1 data.

authInfo - AuthenticationInfo object with which this service should be called.

ids - Input array of symbol IDs

3.4.2.12 QUESTRADELIBRARYAPI GetServerTimeResponse APIENTRY QuestradeAPI::GetServerTime (AuthenticationInfo & *authInfo*)

Returns server time.

authInfo - AuthenticationInfo object with which this service should be called.

3.4.2.13 QUESTRADELIBRARYAPI GetSymbolsResponse APIENTRY QuestradeAPI::GetSymbols (AuthenticationInfo & *authInfo,* const std::vector< uint64 > & *ids*)

Returns symbols data.

authInfo - AuthenticationInfo object with which this service should be called.

ids - Input array of symbol IDs

3.4.2.14 QUESTRADELIBRARYAPI InsertOrderResponse APIENTRY QuestradeAPI::InsertOrder (AuthenticationInfo & *authInfo,* InsertOrderRequest *request*)

Insert order.

authInfo - AuthenticationInfo object with which this service should be called.

3.4.2.15 QUESTRADELIBRARYAPI InsertOrderImpactResponse APIENTRY QuestradeAPI::InsertOrderImpact (AuthenticationInfo & *authInfo,* InsertOrderRequest *request*)

Impact of inserting the order.

authInfo - AuthenticationInfo object with which this service should be called.

3.4.2.16 QUESTRADELIBRARYAPI ReplaceOrderResponse APIENTRY QuestradeAPI::ReplaceOrder (AuthenticationInfo & *authInfo,* ReplaceOrderRequest *request*)

Replace order.

authInfo - AuthenticationInfo object with which this service should be called.

3.4.2.17 QUESTRADELIBRARYAPI ReplaceOrderImpactResponse APIENTRY QuestradeAPI::ReplaceOrderImpact (AuthenticationInfo & *authInfo,* ReplaceOrderRequest *request*)

Impact of replacing the order.

authInfo - AuthenticationInfo object with which this service should be called.

3.4.2.18 QUESTRADELIBRARYAPI SearchSymbolsResponse APIENTRY QuestradeAPI::SearchSymbols (AuthenticationInfo & *authInfo,* const std::string & *prefix,* uint64 *offset* = `0`, uint64 *limit* = `20`)

Search symbols by prefix.

authInfo - AuthenticationInfo object with which this service should be called.

prefix - The prefix

offset - The offset

limit - The max number of symbols to return

Chapter 4

Class Documentation

4.1 QuestradeAPI::AccountData Struct Reference

Account Data.

`#include <CommonStructures.h>`

Public Member Functions

- UserAccountType::Value getType () const
  Type.
- std::string getNumber () const
  Account number.
- AccountStatus::Value getStatus () const
  Status.
- bool isPrimary () const
  Whether or not the account is primary.
- bool isBilling () const
  Whether or not the account is a billing account.
- ClientAccountType::Value getClientAccountType () const
  Client account type.
- std::string dumpToJson () const
  Dumps data in json format (use for debug purposes only, as json is generated)
- AccountData ()
  Creates invalid structure (added for use with STL containers)

Friends

- class **AccountDataImplementation**

4.1.1 Detailed Description

Account Data.

Definition at line 502 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.2 QuestradeAPI::AccountStatus Struct Reference

AccountStatus enumeration.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value {
  Undefined = -1, **UnAllocated** = 0, **Active** = 1, **SuspendedClosed** = 2,
  SuspendedViewOnly = 3, **LiquidateOnly** = 4, **Closed** = 5, **Count** }

  Values for AccountStatus.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of AccountStatus to String.
- static Value fromString (const std::string &name)

  Converts value of AccountStatus from String.

4.2.1 Detailed Description

AccountStatus enumeration.

Definition at line 26 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.3 QuestradeAPI::AuthenticationInfo Struct Reference

Authentication Information.

`#include <Authentication.h>`

Public Member Functions

- bool isAuthenticated () const

  Returns whether authentication was successful and not interrupted.
- bool isValid () const

  Returns whether authentication code provided was valid code.
- int32 getErrorCode () const

  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const

  Returns error message text in case of unsuccessful request.
- std::string getRefreshToken () const

  Returns refresh token with which this session was created.
- std::string getAccessToken () const

  Returns access token of current session. (may change over time if requests are rare)
- std::string getProxyURL () const

  Returns URL of proxy server used by current session. (may change over time if requests are rare)

- bool isDemo () const

  Retuns true if it is a demo AuthenticationInfo.
- AuthenticationInfo ()

  Creates invalid object (added for use with STL libraries)

Friends

- class **AuthenticationInfoImplementation**
- QUESTRADELIBRARYAPI
  AuthenticationInfo APIENTRY Authenticate (const std::string &refreshToken, bool isDemo)

  Authenticates with provided parameters.

4.3.1 Detailed Description

Authentication Information.

Definition at line 25 of file Authentication.h.

4.3.2 Friends And Related Function Documentation

4.3.2.1 QUESTRADELIBRARYAPI AuthenticationInfo APIENTRY Authenticate (const std::string & *refreshToken,* bool *isDemo*) [friend]

Authenticates with provided parameters.

clientId - Client ID of the Application using DLL

refreshToken - Refresh token which user of the application got from Questrade site

isDemo - flag whether this is Live token or Demo (Practice) token

The documentation for this struct was generated from the following file:

- Authentication.h

4.4 QuestradeAPI::BalanceData Struct Reference

Balance Data.

`#include <CommonStructures.h>`

Public Member Functions

- CurrencyType::Value getCurrency () const

  BOCurrency.
- double getCash () const

  Cash.
- double getMarketValue () const

  Market value.
- double getTotalEquity () const

  Total equity.
- double getBuyingPower () const

  Buying power.
- double getMaintenanceExcess () const

Maintenance Excess.

- bool isRealTime () const

  Is data real-time.

- std::string dumpToJson () const

  Dumps data in json format (use for debug purposes only, as json is generated)

- BalanceData ()

  Creates invalid structure (added for use with STL containers)

Friends

- class **BalanceDataImplementation**

4.4.1 Detailed Description

Balance Data.

Definition at line 721 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.5 QuestradeAPI::CancelOrderResponse Struct Reference

CancelOrder Response structure.

`#include <CancelOrder.h>`

Public Member Functions

- uint64 getOrderId () const

  Resulting Order ID.

- bool isValid () const

  Returns whether request was successful.

- int32 getErrorCode () const

  Returns particular error code in case of unsuccessful request.

- std::string getErrorMessage () const

  Returns error message text in case of unsuccessful request.

- DateTime getRateLimitReset () const

  Returns rate limit reset time after this request.

- int32 getRateLimitRemaining () const

  Returns rate limit remaining after this request.

- int32 getAsyncRequestId () const

  Return async request Id that was passed to BeginCancelOrder function.

- std::string dumpToJson () const

  Dumps message in json format (use for debug purposes only, as json is generated)

- CancelOrderResponse ()

  Creates invalid response (added for use with STL containers)

Friends

- class **CancelOrderResponseImplementation**

4.5.1 Detailed Description

CancelOrder Response structure.

Definition at line 26 of file CancelOrder.h.

The documentation for this struct was generated from the following file:

- CancelOrder.h

4.6 QuestradeAPI::CandleData Struct Reference

Historical Candle Data.

`#include <CommonStructures.h>`

Public Member Functions

- DateTime getStart () const

  Start time.
- DateTime getEnd () const

  End time.
- double getLow () const

  Low price.
- double getHigh () const

  High price.
- double getOpen () const

  Open price.
- double getClose () const

  Closing price.
- double getVolume () const

  Volume.
- std::string dumpToJson () const

  Dumps data in json format (use for debug purposes only, as json is generated)
- CandleData ()

  Creates invalid structure (added for use with STL containers)

Friends

- class **CandleDataImplementation**

4.6.1 Detailed Description

Historical Candle Data.

Definition at line 142 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.7 QuestradeAPI::CandlesGranularity Struct Reference

Possible granularities of historical data, required for charting.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value {
  Undefined = 0, **OneMinute** = 1, **TwoMinutes** = 2, **ThreeMinutes** = 3,
  FourMinutes = 4, **FiveMinutes** = 5, **TenMinutes** = 6, **FifteenMinutes** = 7,
  TwentyMinutes = 8, **HalfHour** = 9, **OneHour** = 10, **TwoHours** = 11,
  FourHours = 12, **OneDay** = 13, **OneWeek** = 14, **OneMonth** = 15,
  ThreeMonths = 16, **OneYear** = 17, **T133** = 18, **T144** = 19,
  T233 = 20, **T333** = 21, **T512** = 22, **Count** }

  Values for CandlesGranularity.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of CandlesGranularity to String.
- static Value fromString (const std::string &name)

  Converts value of CandlesGranularity from String.

4.7.1 Detailed Description

Possible granularities of historical data, required for charting.

Definition at line 140 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.8 QuestradeAPI::ChainPerExpiryDate Struct Reference

Option chain grouped per expiry date.

`#include <CommonStructures.h>`

Public Member Functions

- DateTime getExpiryDate () const

  Expiry date.
- std::string getDescription () const

  Description.
- std::string getListingExchange () const

  Listing exchange.
- ExerciseType::Value getOptionExerciseType () const

  Exercise type.
- std::vector< ChainPerRoot > getChainPerRoot () const

  Sub-chains grouped by option root symbol.

- std::string dumpToJson () const

  Dumps data in json format (use for debug purposes only, as json is generated)
- ChainPerExpiryDate ()

  Creates invalid structure (added for use with STL containers)

Friends

- class **ChainPerExpiryDateImplementation**

4.8.1 Detailed Description

Option chain grouped per expiry date.

Definition at line 613 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.9 QuestradeAPI::ChainPerRoot Struct Reference

Option chain grouped per option root symbol for same expiry date.

`#include <CommonStructures.h>`

Public Member Functions

- std::string getOptionRoot () const

  OCC root.
- std::vector< ChainPerStrikePrice > getChainPerStrikePrice () const

  Array of pairs of options per strike price.
- double getMultiplier () const

  Multiplier.
- std::string dumpToJson () const

  Dumps data in json format (use for debug purposes only, as json is generated)
- ChainPerRoot ()

  Creates invalid structure (added for use with STL containers)

Friends

- class **ChainPerRootImplementation**

4.9.1 Detailed Description

Option chain grouped per option root symbol for same expiry date.

Definition at line 578 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.10 QuestradeAPI::ChainPerStrikePrice Struct Reference

Pair of call and put options with same expiry date, root and strike.

`#include <CommonStructures.h>`

Public Member Functions

- double getStrikePrice () const

  Strike price.
- uint64 getCallSymbolId () const

  Call symbol ID.
- uint64 getPutSymbolId () const

  Put symbol ID.
- std::string dumpToJson () const

  Dumps data in json format (use for debug purposes only, as json is generated)
- ChainPerStrikePrice ()

  Creates invalid structure (added for use with STL containers)

Friends

- class **ChainPerStrikePriceImplementation**

4.10.1 Detailed Description

Pair of call and put options with same expiry date, root and strike.

Definition at line 543 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.11 QuestradeAPI::ClientAccountType Struct Reference

Types of clients on account.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value {
  Undefined = 0, **Individual** = 1, **Joint** = 2, **InformalTrust** = 3,
  Corporation = 4, **InvestmentClub** = 5, **FormalTrust** = 6, **Partnership** = 7,
  SoleProprietorship = 8, **Family** = 9, **JointAndInformalTrust** = 10, **Institution** = 11,
  Count }

  Values for ClientAccountType.

Static Public Member Functions

- static const char * toString (Value value)
  Converts value of ClientAccountType to String.
- static Value fromString (const std::string &name)
  Converts value of ClientAccountType from String.

4.11.1 Detailed Description

Types of clients on account.

Definition at line 49 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.12 QuestradeAPI::ContractDeliverables Struct Reference

Deliverables of option contract.

`#include <CommonStructures.h>`

Public Member Functions

- std::vector
  < UnderlyingMultiplierPair > getUnderlyings () const
  Array of underlyings for option.
- double getCashInLieu () const
  Cash in lieu amount.
- std::string dumpToJson () const
  Dumps data in json format (use for debug purposes only, as json is generated)
- ContractDeliverables ()
  Creates invalid structure (added for use with STL containers)

Friends

- class **ContractDeliverablesImplementation**

4.12.1 Detailed Description

Deliverables of option contract.

Definition at line 852 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.13 QuestradeAPI::CurrencyType Struct Reference

CurrencyType enumeration.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value { **Undefined** = 0, **CAD** = 22, **USD** = 99, **Count** }

  Values for CurrencyType.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of CurrencyType to String.
- static Value fromString (const std::string &name)

  Converts value of CurrencyType from String.

4.13.1 Detailed Description

CurrencyType enumeration.

Definition at line 102 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.14 QuestradeAPI::DateTime Struct Reference

Structure for date time.

`#include <CommonDefines.h>`

Public Member Functions

- DateTime (const std::string &isoString)

  Builds DateTime from ISO 8601 (e.g. 2014-10-21T10:56:55.557000-04:00)
- DateTime (int64 epochTime)

  Build time from epoch time (number of seconds from January 1st 1970 00:00 UTC)
- DateTime ()

  Generates invalid DateTime.
- std::string toIsoString () const

  Converts DateTime to ISO 8601 format in EST timezone.
- int64 toEpochTime () const

  Converts to EpochTime (number of seconds from January 1st 1970 00:00 UTC)
- bool isValid () const

  Is this DateTime object valid.

Static Public Member Functions

- static DateTime startOfDay ()

  Creates DateTime object coressponding to start of day in EST/EDT timezone.

Friends

- class **DateTimeImplementation**

4.14.1 Detailed Description

Structure for date time.

Definition at line 110 of file CommonDefines.h.

The documentation for this struct was generated from the following file:

- CommonDefines.h

4.15 QuestradeAPI::EquitySymbol Struct Reference

Data for equities returned by symbol search.

`#include <CommonStructures.h>`

Public Member Functions

- std::string getSymbol () const
  Symbol name.
- uint64 getSymbolId () const
  Symbol ID.
- std::string getDescription () const
  Description of the symbol.
- SecurityType::Value getSecurityType () const
  Security Type.
- std::string getListingExchange () const
  Listing exchange.
- bool isTradable () const
  Whether the symbol is tradable.
- bool isQuotable () const
  Whether the symbol is quotable.
- CurrencyType::Value getCurrency () const
  BOCurrency.
- std::string dumpToJson () const
  Dumps data in json format (use for debug purposes only, as json is generated)
- EquitySymbol ()
  Creates invalid structure (added for use with STL containers)

Friends

- class **EquitySymbolImplementation**

4.15.1 Detailed Description

Data for equities returned by symbol search.

Definition at line 97 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.16 QuestradeAPI::ExecutionData Struct Reference

Execution Data.

`#include <CommonStructures.h>`

Public Member Functions

- std::string getSymbol () const
  Symbol name.
- uint64 getSymbolId () const
  Symbol ID.
- double getQuantity () const
  Execution quantity.
- OrderSide::Value getSide () const
  BOSide.
- double getPrice () const
  Execution fill price.
- uint64 getId () const
  Execution ID.
- uint64 getOrderId () const
  Order ID.
- uint64 getOrderChainId () const
  Chain ID.
- std::string getExchangeExecId () const
  Exchange execution ID.
- DateTime getTimestamp () const
  Execution time.
- std::string getNotes () const
  Notes.
- std::string getVenue () const
  Execution venue.
- double getTotalCost () const
  Total execution cost.
- double getOrderPlacementCommission () const
  Order placement commission.
- double getCommission () const
  Questrade commission for this execution.
- double getExecutionFee () const
  ECN fee for this execution.
- double getSecFee () const
  SEC fee for this execution.
- uint64 getLegId () const
  Leg ID for part of multileg execution.
- double getCanadianExecutionFee () const
  Canadian security fee for this execution.
- uint64 getParentId () const
  Parent execution ID for part of multi-leg execution.
- std::string dumpToJson () const
  Dumps data in json format (use for debug purposes only, as json is generated)
- ExecutionData ()
  Creates invalid structure (added for use with STL containers)

Friends

- class **ExecutionDataImplementation**

4.16.1 Detailed Description

Execution Data.

Definition at line 652 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.17 QuestradeAPI::ExerciseType Struct Reference

Option exercise type.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value { **Undefined** = 0, **American** = 1, **European** = 2, **Count** }

  Values for ExerciseType.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of ExerciseType to String.
- static Value fromString (const std::string &name)

  Converts value of ExerciseType from String.

4.17.1 Detailed Description

Option exercise type.

Definition at line 121 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.18 QuestradeAPI::GetAccountsResponse Struct Reference

GetAccounts Response structure.

`#include <GetAccounts.h>`

Public Member Functions

- std::vector< AccountData > getAccounts () const

  Array of account data.

- uint64 getUserId () const

  ID of the active user.
- bool isValid () const

  Returns whether request was successful.
- int32 getErrorCode () const

  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const

  Returns error message text in case of unsuccessful request.
- DateTime getRateLimitReset () const

  Returns rate limit reset time after this request.
- int32 getRateLimitRemaining () const

  Returns rate limit remaining after this request.
- int32 getAsyncRequestId () const

  Return async request Id that was passed to BeginGetAccounts function.
- std::string dumpToJson () const

  Dumps message in json format (use for debug purposes only, as json is generated)
- GetAccountsResponse ()

  Creates invalid response (added for use with STL containers)

Friends

- class **GetAccountsResponseImplementation**

4.18.1 Detailed Description

GetAccounts Response structure.

Definition at line 26 of file GetAccounts.h.

4.18.2 Member Function Documentation

4.18.2.1 std::vector<AccountData> QuestradeAPI::GetAccountsResponse::getAccounts ( ) const

Array of account data.

Returns empty vector in case of not valid response, check isValid() to know whether the request went through correctly.

The documentation for this struct was generated from the following file:

- GetAccounts.h

4.19 QuestradeAPI::GetBalancesResponse Struct Reference

GetBalances Response structure.

`#include <GetBalances.h>`

Public Member Functions

- std::vector< BalanceData > getPerCurrencyBalances () const

  Per-currency balances.
- std::vector< BalanceData > getCombinedBalances () const

  Combined balances.
- std::vector< BalanceData > getSodPerCurrencyBalances () const

  Start of Day per-currency balances.
- std::vector< BalanceData > getSodCombinedBalances () const

  Start of Day combined balances.
- bool isValid () const

  Returns whether request was successful.
- int32 getErrorCode () const

  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const

  Returns error message text in case of unsuccessful request.
- DateTime getRateLimitReset () const

  Returns rate limit reset time after this request.
- int32 getRateLimitRemaining () const

  Returns rate limit remaining after this request.
- int32 getAsyncRequestId () const

  Return async request Id that was passed to BeginGetBalances function.
- std::string dumpToJson () const

  Dumps message in json format (use for debug purposes only, as json is generated)
- GetBalancesResponse ()

  Creates invalid response (added for use with STL containers)

Friends

- class **GetBalancesResponseImplementation**

4.19.1 Detailed Description

GetBalances Response structure.

Definition at line 26 of file GetBalances.h.

4.19.2 Member Function Documentation

4.19.2.1 std::vector<BalanceData> QuestradeAPI::GetBalancesResponse::getCombinedBalances ( ) const

Combined balances.

Returns empty vector in case of not valid response, check isValid() to know whether the request went through correctly.

4.19.2.2 std::vector<BalanceData> QuestradeAPI::GetBalancesResponse::getPerCurrencyBalances ( ) const

Per-currency balances.

Returns empty vector in case of not valid response, check isValid() to know whether the request went through correctly.

4.19.2.3 std::vector<BalanceData> QuestradeAPI::GetBalancesResponse::getSodCombinedBalances ( ) const

Start of Day combined balances.

Returns empty vector in case of not valid response, check isValid() to know whether the request went through correctly.

4.19.2.4 std::vector<BalanceData> QuestradeAPI::GetBalancesResponse::getSodPerCurrencyBalances ( ) const

Start of Day per-currency balances.

Returns empty vector in case of not valid response, check isValid() to know whether the request went through correctly.

The documentation for this struct was generated from the following file:

- GetBalances.h

4.20 QuestradeAPI::GetCandlesResponse Struct Reference

GetCandles Response structure.

`#include <GetCandles.h>`

Public Member Functions

- std::vector< CandleData > getCandles () const
  Array of candles.
- bool isValid () const
  Returns whether request was successful.
- int32 getErrorCode () const
  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const
  Returns error message text in case of unsuccessful request.
- DateTime getRateLimitReset () const
  Returns rate limit reset time after this request.
- int32 getRateLimitRemaining () const
  Returns rate limit remaining after this request.
- int32 getAsyncRequestId () const
  Return async request Id that was passed to BeginGetCandles function.
- std::string dumpToJson () const
  Dumps message in json format (use for debug purposes only, as json is generated)
- GetCandlesResponse ()
  Creates invalid response (added for use with STL containers)

Friends

- class **GetCandlesResponseImplementation**

4.20.1 Detailed Description

GetCandles Response structure.

Definition at line 26 of file GetCandles.h.

4.20.2 Member Function Documentation

4.20.2.1 std::vector<CandleData> QuestradeAPI::GetCandlesResponse::getCandles ( ) const

Array of candles.

Returns empty vector in case of not valid response, check isValid() to know whether the request went through correctly.

The documentation for this struct was generated from the following file:

- GetCandles.h

4.21 QuestradeAPI::GetExecutionsResponse Struct Reference

GetExecutions Response structure.

`#include <GetExecutions.h>`

Public Member Functions

- std::vector< ExecutionData > getExecutions () const

  Array of executions.
- bool isValid () const

  Returns whether request was successful.
- int32 getErrorCode () const

  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const

  Returns error message text in case of unsuccessful request.
- DateTime getRateLimitReset () const

  Returns rate limit reset time after this request.
- int32 getRateLimitRemaining () const

  Returns rate limit remaining after this request.
- int32 getAsyncRequestId () const

  Return async request Id that was passed to BeginGetExecutions function.
- std::string dumpToJson () const

  Dumps message in json format (use for debug purposes only, as json is generated)
- GetExecutionsResponse ()

  Creates invalid response (added for use with STL containers)

Friends

- class **GetExecutionsResponseImplementation**

4.21.1 Detailed Description

GetExecutions Response structure.

Definition at line 26 of file GetExecutions.h.

4.21.2 Member Function Documentation

4.21.2.1 std::vector<ExecutionData> QuestradeAPI::GetExecutionsResponse::getExecutions ( ) const

Array of executions.

Returns empty vector in case of not valid response, check isValid() to know whether the request went through correctly.

The documentation for this struct was generated from the following file:

- GetExecutions.h

4.22 QuestradeAPI::GetMarketsResponse Struct Reference

GetMarkets Response structure.

`#include <GetMarkets.h>`

Public Member Functions

- std::vector< Market > getMarkets () const

  Array of listing exchanges.
- bool isValid () const

  Returns whether request was successful.
- int32 getErrorCode () const

  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const

  Returns error message text in case of unsuccessful request.
- DateTime getRateLimitReset () const

  Returns rate limit reset time after this request.
- int32 getRateLimitRemaining () const

  Returns rate limit remaining after this request.
- int32 getAsyncRequestId () const

  Return async request Id that was passed to BeginGetMarkets function.
- std::string dumpToJson () const

  Dumps message in json format (use for debug purposes only, as json is generated)
- GetMarketsResponse ()

  Creates invalid response (added for use with STL containers)

Friends

- class **GetMarketsResponseImplementation**

4.22.1 Detailed Description

GetMarkets Response structure.

Definition at line 26 of file GetMarkets.h.

4.22.2 Member Function Documentation

4.22.2.1 std::vector<Market> QuestradeAPI::GetMarketsResponse::getMarkets ( ) const

Array of listing exchanges.

Returns empty vector in case of not valid response, check isValid() to know whether the request went through correctly.

The documentation for this struct was generated from the following file:

- GetMarkets.h

4.23 QuestradeAPI::GetOptionsResponse Struct Reference

GetOptions Response structure.

`#include <GetOptions.h>`

Public Member Functions

- std::vector< ChainPerExpiryDate > getOptionChain () const

  Option chain for the underlying symbol.
- bool isValid () const

  Returns whether request was successful.
- int32 getErrorCode () const

  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const

  Returns error message text in case of unsuccessful request.
- DateTime getRateLimitReset () const

  Returns rate limit reset time after this request.
- int32 getRateLimitRemaining () const

  Returns rate limit remaining after this request.
- int32 getAsyncRequestId () const

  Return async request Id that was passed to BeginGetOptions function.
- std::string dumpToJson () const

  Dumps message in json format (use for debug purposes only, as json is generated)
- GetOptionsResponse ()

  Creates invalid response (added for use with STL containers)

Friends

- class **GetOptionsResponseImplementation**

4.23.1 Detailed Description

GetOptions Response structure.

Definition at line 26 of file GetOptions.h.

4.23.2 Member Function Documentation

4.23.2.1 std::vector<ChainPerExpiryDate> QuestradeAPI::GetOptionsResponse::getOptionChain ( ) const

Option chain for the underlying symbol.

Returns empty vector in case of not valid response, check isValid() to know whether the request went through correctly.

The documentation for this struct was generated from the following file:

- GetOptions.h

4.24 QuestradeAPI::GetOrdersResponse Struct Reference

GetOrders Response structure.

`#include <GetOrders.h>`

Public Member Functions

- std::vector< OrderData > getOrders () const

  Array of Order Data.
- bool isValid () const

  Returns whether request was successful.
- int32 getErrorCode () const

  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const

  Returns error message text in case of unsuccessful request.
- DateTime getRateLimitReset () const

  Returns rate limit reset time after this request.
- int32 getRateLimitRemaining () const

  Returns rate limit remaining after this request.
- int32 getAsyncRequestId () const

  Return async request Id that was passed to BeginGetOrders function.
- std::string dumpToJson () const

  Dumps message in json format (use for debug purposes only, as json is generated)
- GetOrdersResponse ()

  Creates invalid response (added for use with STL containers)

Friends

- class **GetOrdersResponseImplementation**

4.24.1 Detailed Description

GetOrders Response structure.

Definition at line 26 of file GetOrders.h.

4.24.2 Member Function Documentation

4.24.2.1 std::vector<OrderData> QuestradeAPI::GetOrdersResponse::getOrders ( ) const

Array of Order Data.

Returns empty vector in case of not valid response, check isValid() to know whether the request went through correctly.

The documentation for this struct was generated from the following file:

- GetOrders.h

4.25 QuestradeAPI::GetPositionsResponse Struct Reference

GetPositions Response structure.

`#include <GetPositions.h>`

Public Member Functions

- std::vector< PositionData > getPositions () const

  Resulting array of position data.
- bool isValid () const

  Returns whether request was successful.
- int32 getErrorCode () const

  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const

  Returns error message text in case of unsuccessful request.
- DateTime getRateLimitReset () const

  Returns rate limit reset time after this request.
- int32 getRateLimitRemaining () const

  Returns rate limit remaining after this request.
- int32 getAsyncRequestId () const

  Return async request Id that was passed to BeginGetPositions function.
- std::string dumpToJson () const

  Dumps message in json format (use for debug purposes only, as json is generated)
- GetPositionsResponse ()

  Creates invalid response (added for use with STL containers)

Friends

- class **GetPositionsResponseImplementation**

4.25.1 Detailed Description

GetPositions Response structure.

Definition at line 26 of file GetPositions.h.

4.25.2 Member Function Documentation

4.25.2.1 std::vector<PositionData> QuestradeAPI::GetPositionsResponse::getPositions ( ) const

Resulting array of position data.

Returns empty vector in case of not valid response, check isValid() to know whether the request went through correctly.

The documentation for this struct was generated from the following file:

- GetPositions.h

4.26 QuestradeAPI::GetQuoteResponse Struct Reference

GetQuote Response structure.

`#include <GetQuote.h>`

Public Member Functions

- std::vector< Level1DataItem > getQuotes () const

  Array of quotes.
- bool isValid () const

  Returns whether request was successful.
- int32 getErrorCode () const

  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const

  Returns error message text in case of unsuccessful request.
- DateTime getRateLimitReset () const

  Returns rate limit reset time after this request.
- int32 getRateLimitRemaining () const

  Returns rate limit remaining after this request.
- int32 getAsyncRequestId () const

  Return async request Id that was passed to BeginGetQuote function.
- std::string dumpToJson () const

  Dumps message in json format (use for debug purposes only, as json is generated)
- GetQuoteResponse ()

  Creates invalid response (added for use with STL containers)

Friends

- class **GetQuoteResponseImplementation**

4.26.1 Detailed Description

GetQuote Response structure.

Definition at line 26 of file GetQuote.h.

4.26.2 Member Function Documentation

4.26.2.1 std::vector<Level1DataItem> QuestradeAPI::GetQuoteResponse::getQuotes ( ) const

Array of quotes.

Returns empty vector in case of not valid response, check isValid() to know whether the request went through correctly.

The documentation for this struct was generated from the following file:

- GetQuote.h

4.27 QuestradeAPI::GetServerTimeResponse Struct Reference

GetServerTime Response structure.

`#include <GetServerTime.h>`

Public Member Functions

- DateTime getTime () const
  Current server time.
- bool isValid () const
  Returns whether request was successful.
- int32 getErrorCode () const
  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const
  Returns error message text in case of unsuccessful request.
- DateTime getRateLimitReset () const
  Returns rate limit reset time after this request.
- int32 getRateLimitRemaining () const
  Returns rate limit remaining after this request.
- int32 getAsyncRequestId () const
  Return async request Id that was passed to BeginGetServerTime function.
- std::string dumpToJson () const
  Dumps message in json format (use for debug purposes only, as json is generated)
- GetServerTimeResponse ()
  Creates invalid response (added for use with STL containers)

Friends

- class **GetServerTimeResponseImplementation**

4.27.1 Detailed Description

GetServerTime Response structure.

Definition at line 26 of file GetServerTime.h.

The documentation for this struct was generated from the following file:

- GetServerTime.h

4.28 QuestradeAPI::GetSymbolsResponse Struct Reference

GetSymbols Response structure.

`#include <GetSymbols.h>`

Public Member Functions

- std::vector< SymbolData > getSymbols () const

  Resulting array of symbol data.
- bool isValid () const

  Returns whether request was successful.
- int32 getErrorCode () const

  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const

  Returns error message text in case of unsuccessful request.
- DateTime getRateLimitReset () const

  Returns rate limit reset time after this request.
- int32 getRateLimitRemaining () const

  Returns rate limit remaining after this request.
- int32 getAsyncRequestId () const

  Return async request Id that was passed to BeginGetSymbols function.
- std::string dumpToJson () const

  Dumps message in json format (use for debug purposes only, as json is generated)
- GetSymbolsResponse ()

  Creates invalid response (added for use with STL containers)

Friends

- class **GetSymbolsResponseImplementation**

4.28.1 Detailed Description

GetSymbols Response structure.

Definition at line 26 of file GetSymbols.h.

4.28.2 Member Function Documentation

4.28.2.1 std::vector<SymbolData> QuestradeAPI::GetSymbolsResponse::getSymbols ( ) const

Resulting array of symbol data.

Returns empty vector in case of not valid response, check isValid() to know whether the request went through correctly.

The documentation for this struct was generated from the following file:

- GetSymbols.h

4.29 QuestradeAPI::InsertOrderImpactResponse Struct Reference

InsertOrderImpact Response structure.

`#include <InsertOrderImpact.h>`

Public Member Functions

- double getEstimatedCommissions () const
  The estimated commissions.
- double getBuyingPowerEffect () const
  The buying power effect.
- double getBuyingPowerResult () const
  The buying power result.
- double getMaintExcessEffect () const
  The maintenance excess effect.
- double getMaintExcessResult () const
  The maintenance excess result.
- OrderSide::Value getSide () const
  Resulting side of the order.
- std::string getTradeValueCalculation () const
  The trade value calculation.
- double getPrice () const
  Estimated price of execution.
- bool isValid () const
  Returns whether request was successful.
- int32 getErrorCode () const
  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const
  Returns error message text in case of unsuccessful request.
- DateTime getRateLimitReset () const
  Returns rate limit reset time after this request.
- int32 getRateLimitRemaining () const
  Returns rate limit remaining after this request.
- int32 getAsyncRequestId () const
  Return async request Id that was passed to BeginInsertOrderImpact function.
- std::string dumpToJson () const
  Dumps message in json format (use for debug purposes only, as json is generated)
- InsertOrderImpactResponse ()
  Creates invalid response (added for use with STL containers)

Friends

- class **InsertOrderImpactResponseImplementation**

4.29.1 Detailed Description

InsertOrderImpact Response structure.

Definition at line 26 of file InsertOrderImpact.h.

The documentation for this struct was generated from the following file:

- InsertOrderImpact.h

4.30 QuestradeAPI::InsertOrderRequest Struct Reference

Structure that used as input in some services.

`#include <CommonStructures.h>`

Public Member Functions

- InsertOrderRequest (const std::string &accountNumber, uint64 symbolId, double quantity, OrderType::Value orderType, OrderTimeInForce::Value timeInForce, OrderAction::Value action, const std::string &primaryRoute)

  Creates a request with required fields filled out.

Public Attributes

- std::string m_accountNumber

  The account number.
- uint64 m_symbolId

  The symbol ID.
- double m_quantity

  Quantity of the order.
- double m_icebergQuantity

  Iceberg quantity.
- double m_minQuantity

  Minimum quantity.
- double m_limitPrice

  The limit price.
- double m_stopPrice

  The stop price.
- bool m_isAllOrNone

  Is All-or-none flag.
- bool m_isAnonymous

  Is anonymous flag.
- OrderType::Value m_orderType

  The type.
- OrderTimeInForce::Value m_timeInForce

  The time in force.
- DateTime m_gtdDate

  The good-till date.
- OrderAction::Value m_action

  The side.
- std::string m_primaryRoute

  The primary route.
- std::string m_secondaryRoute

  The secondary route.
- bool m_isLimitOffsetInDollar

  Is limit offset in dollar flag for synthetic orders.

4.30.1 Detailed Description

Structure that used as input in some services.

Definition at line 186 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.31 QuestradeAPI::InsertOrderResponse Struct Reference

InsertOrder Response structure.

`#include <InsertOrder.h>`

Public Member Functions

- uint64 getOrderId () const

  The ID of the new order.
- std::vector< OrderData > getOrders () const

  Orders that were created after request.
- bool isValid () const

  Returns whether request was successful.
- int32 getErrorCode () const

  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const

  Returns error message text in case of unsuccessful request.
- DateTime getRateLimitReset () const

  Returns rate limit reset time after this request.
- int32 getRateLimitRemaining () const

  Returns rate limit remaining after this request.
- int32 getAsyncRequestId () const

  Return async request Id that was passed to BeginInsertOrder function.
- std::string dumpToJson () const

  Dumps message in json format (use for debug purposes only, as json is generated)
- InsertOrderResponse ()

  Creates invalid response (added for use with STL containers)

Friends

- class **InsertOrderResponseImplementation**

4.31.1 Detailed Description

InsertOrder Response structure.

Definition at line 26 of file InsertOrder.h.

4.31.2 Member Function Documentation

4.31.2.1 std::vector<OrderData> QuestradeAPI::InsertOrderResponse::getOrders ( ) const

Orders that were created after request.

Returns empty vector in case of not valid response, check isValid() to know whether the request went through correctly.

The documentation for this struct was generated from the following file:

- InsertOrder.h

4.32 QuestradeAPI::Level1DataItem Struct Reference

Level 1 quote data.

`#include <CommonStructures.h>`

Public Member Functions

- std::string getSymbol () const
  Symbol name.
- uint64 getSymbolId () const
  Symbol ID.
- std::string getTier () const
  Tier of Pink sheet if applicable.
- double getBidPrice () const
  Bid price.
- uint64 getBidSize () const
  Bid size.
- double getAskPrice () const
  Ask price.
- uint64 getAskSize () const
  Ask size.
- double getLastTradePriceTrHrs () const
  Last trade price during trading hours.
- double getLastTradePrice () const
  Last trade price.
- uint64 getLastTradeSize () const
  Last trade size.
- TickType::Value getLastTradeTick () const
  Last trade tick.
- DateTime getLastTradeTime () const
  Last trade time.
- uint64 getVolume () const
  Volume.
- double getOpenPrice () const
  Open price.
- double getHighPrice () const
  High price.
- double getLowPrice () const

Low price.

- uint64 getDelay () const

  How much is data delayed.

- bool isHalted () const

  Whether or not the symbol was halted.

- std::string dumpToJson () const

  Dumps data in json format (use for debug purposes only, as json is generated)

- Level1DataItem ()

  Creates invalid structure (added for use with STL containers)

Friends

- class **Level1DataItemImplementation**

4.32.1 Detailed Description

Level 1 quote data.

Definition at line 32 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.33 QuestradeAPI::Market Struct Reference

Listing exchange information.

`#include <CommonStructures.h>`

Public Member Functions

- std::string getName () const

  Listing exchange name.

- std::vector< std::string > getTradingVenues () const

  Array of trading venues.

- std::string getDefaultTradingVenue () const

  Default trading venue.

- std::vector< std::string > getPrimaryOrderRoutes () const

  Array of primary order routes.

- std::vector< std::string > getSecondaryOrderRoutes () const

  Array of secondary order routes.

- std::vector< std::string > getLevel1Feeds () const

  Array of level 1 feeds.

- std::vector< std::string > getLevel2Feeds () const

  Array of level 2 feeds.

- DateTime getExtendedStartTime () const

  Extended market start time.

- DateTime getStartTime () const

  Trading hours start time.

- DateTime getEndTime () const

Trading hours end time.

- DateTime getExtendedEndTime () const

  Extended market end time.

- uint64 getSnapQuotesLimit () const

  Snap quote limit.

- std::string dumpToJson () const

  Dumps data in json format (use for debug purposes only, as json is generated)

- Market ()

  Creates invalid structure (added for use with STL containers)

Friends

- class **MarketImplementation**

4.33.1 Detailed Description

Listing exchange information.

Definition at line 764 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.34 QuestradeAPI::MinTickData Struct Reference

Min tick rules data.

`#include <CommonStructures.h>`

Public Member Functions

- double getPivot () const

  Price above which this min tick applies.

- double getMinTick () const

  Min tick.

- std::string dumpToJson () const

  Dumps data in json format (use for debug purposes only, as json is generated)

- MinTickData ()

  Creates invalid structure (added for use with STL containers)

Friends

- class **MinTickDataImplementation**

4.34.1 Detailed Description

Min tick rules data.

Definition at line 885 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.35 QuestradeAPI::OptionDurationType Struct Reference

Duration types of options.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value {
  Undefined = 0, **Weekly** = 1, **Monthly** = 2, **Quarterly** = 3,
  LEAP = 4, **Count** }

  Values for OptionDurationType.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of OptionDurationType to String.
- static Value fromString (const std::string &name)

  Converts value of OptionDurationType from String.

4.35.1 Detailed Description

Duration types of options.

Definition at line 179 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.36 QuestradeAPI::OptionType Struct Reference

Option types.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value { **Undefined** = 0, **Call** = 1, **Put** = 2, **Count** }

  Values for OptionType.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of OptionType to String.
- static Value fromString (const std::string &name)

  Converts value of OptionType from String.

4.36.1 Detailed Description

Option types.

Definition at line 200 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.37 QuestradeAPI::OrderAction Struct Reference

Action for inserted order.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value { **Undefined** = 0, **Buy** = 1, **Sell** = 2, **Count** }

  Values for OrderAction.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of OrderAction to String.
- static Value fromString (const std::string &name)

  Converts value of OrderAction from String.

4.37.1 Detailed Description

Action for inserted order.

Definition at line 239 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.38 QuestradeAPI::OrderClass Struct Reference

For bracket order describes the type of component.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value {
  Undefined = 0, **Primary** = 1, **Limit** = 2, **StopLoss** = 3,
  Count }

  Values for OrderClass.

Static Public Member Functions

- static const char * toString (Value value)
  Converts value of OrderClass to String.
- static Value fromString (const std::string &name)
  Converts value of OrderClass from String.

4.38.1 Detailed Description

For bracket order describes the type of component.

Definition at line 219 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.39 QuestradeAPI::OrderData Struct Reference

Describes order details.

`#include <CommonStructures.h>`

Public Member Functions

- uint64 getId () const
  Order ID.
- std::string getSymbol () const
  Symbol name.
- uint64 getSymbolId () const
  Symbol ID.
- double getTotalQuantity () const
  Total order quantity.
- double getOpenQuantity () const
  Open order quantity.
- double getFilledQuantity () const
  Filled quantity.
- double getCanceledQuantity () const
  Canceled quantity.
- OrderSide::Value getSide () const
  BOSide.
- OrderType::Value getOrderType () const
  Type.
- double getLimitPrice () const
  Limit price.
- double getStopPrice () const
  Stop price.
- bool isAllOrNone () const
  Is all or none.
- bool isAnonymous () const
  Is anonymous.
- double getIcebergQuantity () const

Iceberg quantity.

- double getMinQuantity () const

  Min quantity.
- double getAvgExecPrice () const

  Average execution price.
- double getLastExecPrice () const

  Last execution price.
- OrderSource::Value getSource () const

  Platform from which order was placed.
- OrderTimeInForce::Value getTimeInForce () const

  Time in force.
- DateTime getGtdDate () const

  Good-till date.
- OrderState::Value getState () const

  Order state.
- std::string getRejectionReason () const

  Rejection reason string.
- uint64 getChainId () const

  Chain ID.
- DateTime getCreationTime () const

  Order record creation time.
- DateTime getUpdateTime () const

  Order record update time.
- std::string getNotes () const

  Notes.
- std::string getPrimaryRoute () const

  Primary route.
- std::string getSecondaryRoute () const

  Secondary route.
- std::string getOrderRoute () const

  Actual route to which order was sent.
- std::string getVenueHoldingOrder () const

  Venue holding order.
- double getComissionCharged () const

  Commission charged for the order.
- std::string getExchangeOrderId () const

  Exchange order ID.
- bool isSignificantShareHolder () const

  Whether or not the order was placed by a significant shareholder.
- bool isInsider () const

  Whether or not the order was placed by an insider.
- bool isLimitOffsetInDollar () const

  Whether or not the synthetic order's limit price offset is in dollars.
- uint64 getUserId () const

  User ID placing the order.
- double getPlacementCommission () const

  Placement commission.
- std::vector< OrderLegData > getLegs () const

  Array of legs for multi-leg orders.
- StrategyType::Value getStrategyType () const

  Strategy type for multi-leg orders.

- double getTriggerStopPrice () const

  The trigger stop price for the order.
- uint64 getOrderGroupId () const

  Order Group ID of the order, if its bracket component.
- OrderClass::Value getOrderClass () const

  For bracket order describes the type of component.
- std::string dumpToJson () const

  Dumps data in json format (use for debug purposes only, as json is generated)
- OrderData ()

  Creates invalid structure (added for use with STL containers)

Friends

- class **OrderDataImplementation**

4.39.1 Detailed Description

Describes order details.

Definition at line 286 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.40 QuestradeAPI::OrderLegData Struct Reference

Describes leg for multi-leg order.

`#include <CommonStructures.h>`

Public Member Functions

- uint64 getLegId () const

  Leg ID.
- std::string getSymbol () const

  Symbol name.
- uint64 getSymbolId () const

  Symbol ID.
- double getLegRatioQuantity () const

  Leg ratio quantity.
- OrderSide::Value getSide () const

  Order side.
- double getAvgExecPrice () const

  Average execution price.
- double getLastExecPrice () const

  Last execution price.
- std::string dumpToJson () const

  Dumps data in json format (use for debug purposes only, as json is generated)
- OrderLegData ()

  Creates invalid structure (added for use with STL containers)

Friends

- class **OrderLegDataImplementation**

4.40.1 Detailed Description

Describes leg for multi-leg order.

Definition at line 243 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.41 QuestradeAPI::OrderSide Struct Reference

ClientOrderSide aka Action.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value {
  Undefined = 0, **Buy** = 1, **Sell** = 2, **Short** = 3,
  Cov = 4, **BTO** = 5, **STC** = 6, **STO** = 7,
  BTC = 8, **Count** }

  Values for OrderSide.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of OrderSide to String.
- static Value fromString (const std::string &name)

  Converts value of OrderSide from String.

4.41.1 Detailed Description

ClientOrderSide aka Action.

Definition at line 77 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.42 QuestradeAPI::OrderSource Struct Reference

Platform from which order was placed.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value {
  Undefined = 0, **Automatic** = 1, **BackOfficeAgent** = 2, **QuestradeIQ** = 3,
  QuestradeIQEdge = 4, **QuestradeIQMobile** = 5, **QuestradeIQEssential** = 6, **TradeDesk** = 7,
  QuestradeIQMobileWeb = 8, **TradingAPI** = 9, **AutomatedCorrection** = 10, **Count** }

  Values for OrderSource.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of OrderSource to String.
- static Value fromString (const std::string &name)

  Converts value of OrderSource from String.

4.42.1 Detailed Description

Platform from which order was placed.

Definition at line 258 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.43 QuestradeAPI::OrderState Struct Reference

State of the order.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value {
  Undefined = 0, **Failed** = 1, **Pending** = 2, **Accepted** = 3,
  Rejected = 4, **CancelPending** = 5, **Canceled** = 6, **PartialCanceled** = 7,
  Partial = 8, **Executed** = 9, **ReplacePending** = 10, **Replaced** = 11,
  Stopped = 12, **Suspended** = 13, **Expired** = 14, **Queued** = 15,
  Triggered = 16, **Activated** = 17, **PendingRiskReview** = 18, **ContingentOrder** = 19,
  Count }

  Values for OrderState.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of OrderState to String.
- static Value fromString (const std::string &name)

  Converts value of OrderState from String.

4.43.1 Detailed Description

State of the order.

Definition at line 285 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.44 QuestradeAPI::OrderStateFilterTypes Struct Reference

Types used for filtering orders by state.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value {
  Undefined = 0, **All** = 1, **Open** = 2, **Closed** = 3,
  Count }

  Values for OrderStateFilterTypes.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of OrderStateFilterTypes to String.
- static Value fromString (const std::string &name)

  Converts value of OrderStateFilterTypes from String.

4.44.1 Detailed Description

Types used for filtering orders by state.

Definition at line 321 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.45 QuestradeAPI::OrderTimeInForce Struct Reference

Time in force aka Duration of the order.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value {
  Undefined = 0, **Day** = 1, **GoodTillCanceled** = 2, **GoodTillExtendedDay** = 3,
  GoodTillDate = 4, **ImmediateOrCancel** = 5, **FillOrKill** = 6, **Count** }

  Values for OrderTimeInForce.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of OrderTimeInForce to String.
- static Value fromString (const std::string &name)

  Converts value of OrderTimeInForce from String.

4.45.1 Detailed Description

Time in force aka Duration of the order.

Definition at line 341 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.46 QuestradeAPI::OrderType Struct Reference

Type of the order.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value {
  Undefined = 0, **Market** = 1, **Limit** = 2, **Stop** = 3,
  StopLimit = 4, **TrailStopInPercentage** = 5, **TrailStopInDollar** = 6, **TrailStopLimitInPercentage** = 7,
  TrailStopLimitInDollar = 8, **LimitOnOpen** = 9, **LimitOnClose** = 10, **Count** }

  Values for OrderType.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of OrderType to String.
- static Value fromString (const std::string &name)

  Converts value of OrderType from String.

4.46.1 Detailed Description

Type of the order.

Definition at line 364 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.47 QuestradeAPI::PositionData Struct Reference

Position Data.

`#include <CommonStructures.h>`

Public Member Functions

- std::string getSymbol () const

  Symbol name.
- uint64 getSymbolId () const

  Symbol ID.
- double getOpenQuantity () const

  Open quantity.
- double getClosedQuantity () const

  Closed quantity.
- double getCurrentMarketValue () const

  Current market value.
- double getCurrentPrice () const

  Current price.
- double getAverageEntryPrice () const

  Average entry price.
- double getClosedPnl () const

  Closed profit and loss.
- double getOpenPnl () const

  Open profit and loss.
- double getTotalCost () const

  Total cost.
- bool isRealTime () const

  Whether or not real-time prices are used.
- bool isUnderReorg () const

  Whether or not the position is under corporate action.
- std::string dumpToJson () const

  Dumps data in json format (use for debug purposes only, as json is generated)
- PositionData ()

  Creates invalid structure (added for use with STL containers)

Friends

- class **PositionDataImplementation**

4.47.1 Detailed Description

Position Data.

Definition at line 449 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.48 QuestradeAPI::ReplaceOrderImpactResponse Struct Reference

ReplaceOrderImpact Response structure.

`#include <ReplaceOrderImpact.h>`

Public Member Functions

- double getEstimatedCommissions () const
  Resulting estimated commissions.
- double getBuyingPowerEffect () const
  Resulting buying power effect.
- double getBuyingPowerResult () const
  Resulting buying power result.
- double getMaintExcessEffect () const
  Resulting maintenance excess effect.
- double getMaintExcessResult () const
  Resulting maintenance excess result.
- std::string getTradeValueCalculation () const
  Resulting trade value calculation.
- double getPrice () const
  Estimated price of execution.
- bool isValid () const
  Returns whether request was successful.
- int32 getErrorCode () const
  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const
  Returns error message text in case of unsuccessful request.
- DateTime getRateLimitReset () const
  Returns rate limit reset time after this request.
- int32 getRateLimitRemaining () const
  Returns rate limit remaining after this request.
- int32 getAsyncRequestId () const
  Return async request Id that was passed to BeginReplaceOrderImpact function.
- std::string dumpToJson () const
  Dumps message in json format (use for debug purposes only, as json is generated)
- ReplaceOrderImpactResponse ()
  Creates invalid response (added for use with STL containers)

Friends

- class **ReplaceOrderImpactResponseImplementation**

4.48.1 Detailed Description

ReplaceOrderImpact Response structure.

Definition at line 26 of file ReplaceOrderImpact.h.

The documentation for this struct was generated from the following file:

- ReplaceOrderImpact.h

4.49 QuestradeAPI::ReplaceOrderRequest Struct Reference

Structure that used as input in some services.

`#include <CommonStructures.h>`

Public Member Functions

- ReplaceOrderRequest (const std::string &accountNumber, uint64 orderId, double quantity, OrderType::Value orderType, OrderTimeInForce::Value timeInForce)

  Creates a request with required fields filled out.

Public Attributes

- std::string m_accountNumber

  Account number.
- uint64 m_orderId

  ID of the order to replace.
- double m_quantity

  New quantity.
- double m_icebergQuantity

  New iceberg quantity.
- double m_minQuantity

  New min quantity.
- double m_limitPrice

  New limit price.
- double m_stopPrice

  New stop price.
- OrderType::Value m_orderType

  New type.
- OrderTimeInForce::Value m_timeInForce

  New time in force.
- DateTime m_gtdDate

  New good-till date.
- bool m_isLimitOffsetInDollar

  New value for is limit offset in dollar for synthetic orders.
- bool m_isAllOrNone

  New value for is all or none.
- bool m_isAnonymous

  New value for is anonymous.

4.49.1 Detailed Description

Structure that used as input in some services.

Definition at line 400 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.50 QuestradeAPI::ReplaceOrderResponse Struct Reference

ReplaceOrder Response structure.

`#include <ReplaceOrder.h>`

Public Member Functions

- uint64 getOrderId () const

  Resulting order ID.
- std::vector< OrderData > getOrders () const

  Orders that were changed and created after request.
- bool isValid () const

  Returns whether request was successful.
- int32 getErrorCode () const

  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const

  Returns error message text in case of unsuccessful request.
- DateTime getRateLimitReset () const

  Returns rate limit reset time after this request.
- int32 getRateLimitRemaining () const

  Returns rate limit remaining after this request.
- int32 getAsyncRequestId () const

  Return async request Id that was passed to BeginReplaceOrder function.
- std::string dumpToJson () const

  Dumps message in json format (use for debug purposes only, as json is generated)
- ReplaceOrderResponse ()

  Creates invalid response (added for use with STL containers)

Friends

- class **ReplaceOrderResponseImplementation**

4.50.1 Detailed Description

ReplaceOrder Response structure.

Definition at line 26 of file ReplaceOrder.h.

4.50.2 Member Function Documentation

4.50.2.1 std::vector<OrderData> QuestradeAPI::ReplaceOrderResponse::getOrders ( ) const

Orders that were changed and created after request.

Returns empty vector in case of not valid response, check isValid() to know whether the request went through correctly.

The documentation for this struct was generated from the following file:

- ReplaceOrder.h

4.51 QuestradeAPI::SearchSymbolsResponse Struct Reference

SearchSymbols Response structure.

`#include <SearchSymbols.h>`

Public Member Functions

- std::vector< EquitySymbol > getSymbols () const

  Resulting symbols matching the search criteria.
- bool isValid () const

  Returns whether request was successful.
- int32 getErrorCode () const

  Returns particular error code in case of unsuccessful request.
- std::string getErrorMessage () const

  Returns error message text in case of unsuccessful request.
- DateTime getRateLimitReset () const

  Returns rate limit reset time after this request.
- int32 getRateLimitRemaining () const

  Returns rate limit remaining after this request.
- int32 getAsyncRequestId () const

  Return async request Id that was passed to BeginSearchSymbols function.
- std::string dumpToJson () const

  Dumps message in json format (use for debug purposes only, as json is generated)
- SearchSymbolsResponse ()

  Creates invalid response (added for use with STL containers)

Friends

- class **SearchSymbolsResponseImplementation**

4.51.1 Detailed Description

SearchSymbols Response structure.

Definition at line 26 of file SearchSymbols.h.

4.51.2 Member Function Documentation

4.51.2.1 std::vector<EquitySymbol> QuestradeAPI::SearchSymbolsResponse::getSymbols ( ) const

Resulting symbols matching the search criteria.

Returns empty vector in case of not valid response, check isValid() to know whether the request went through correctly.

The documentation for this struct was generated from the following file:

- SearchSymbols.h

4.52 QuestradeAPI::SecurityType Struct Reference

Basic types of securities.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value {
  Undefined = 0, **Stock** = 1, **Option** = 2, **Bond** = 3,
  Right = 4, **Commodity** = 5, **MutualFund** = 6, **Index** = 7,
  Currency = 8, **Count** }

  Values for SecurityType.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of SecurityType to String.
- static Value fromString (const std::string &name)

  Converts value of SecurityType from String.

4.52.1 Detailed Description

Basic types of securities.

Definition at line 391 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.53 QuestradeAPI::StrategyType Struct Reference

Strategy type for multi-leg order.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value {
  Undefined = 0, **CallsPuts** = 1, **Calls** = 2, **Puts** = 3,
  CoveredCall = 4, **MarriedPuts** = 5, **VerticalCallSpread** = 6, **VerticalPutSpread** = 7,
  CalendarCallSpread = 8, **CalendarPutSpread** = 9, **DiagonalCallSpread** = 10, **DiagonalPutSpread** = 11,
  Collar = 12, **Straddle** = 13, **Strangle** = 14, **ButterflyCall** = 15,
  ButterflyPut = 16, **IronButterfly** = 17, **CondorCall** = 18, **CondorPut** = 19,
  IronCondor = 20, **Custom** = 21, **SingleLeg** = 22, **Count** }

  Values for StrategyType.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of StrategyType to String.
- static Value fromString (const std::string &name)

  Converts value of StrategyType from String.

4.53.1 Detailed Description

Strategy type for multi-leg order.

Definition at line 416 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.54 QuestradeAPI::SymbolData Struct Reference

Symbol Data.

`#include <CommonStructures.h>`

Public Member Functions

- std::string getSymbol () const
  Symbol name.
- uint64 getSymbolId () const
  Symbol ID.
- double getPrevDayClosePrice () const
  Previous day close price.
- double getHighPrice52 () const
  52-week high price
- double getLowPrice52 () const
  52-week low price
- uint64 getAverageVol3Months () const
  Average 3-month volume.
- uint64 getAverageVol20Days () const
  Average 20-day volume.
- uint64 getOutstandingShares () const
  Number of outstanding shares.
- double getEps () const
  The EPS.
- double getPe () const
  The PE ratio.
- double getDividend () const
  The dividend.
- double getYield () const
  The yield.
- DateTime getExDate () const
  Expiration date.
- uint64 getMarketCap () const
  Market capitalization.
- uint64 getTradeUnit () const
  Multiplier.
- OptionType::Value getOptionType () const
  Option type.
- OptionDurationType::Value getOptionDurationType () const
  Option duration type.

- std::string getOptionRoot () const

  OCC root.
- ContractDeliverables getOptionContractDeliverables () const

  Contract deliverables.
- ExerciseType::Value getOptionExerciseType () const

  Exercise type.
- std::string getListingExchange () const

  Listing exchange.
- std::string getDescription () const

  Description of the symbol.
- SecurityType::Value getSecurityType () const

  Security type.
- DateTime getOptionExpiryDate () const

  Expiration date.
- DateTime getDividendDate () const

  Dividend date.
- double getOptionStrikePrice () const

  Strike price.
- bool isTradable () const

  Whether or not the symbol is tradable.
- bool isQuotable () const

  Whether or not the symbol is quotable.
- bool isHasOptions () const

  Whether or not the symbol has options.
- std::string getCurrency () const

  Currency string.
- std::vector< MinTickData > getMinTicks () const

  Array of MinTickData.
- std::string dumpToJson () const

  Dumps data in json format (use for debug purposes only, as json is generated)
- SymbolData ()

  Creates invalid structure (added for use with STL containers)

Friends

- class **SymbolDataImplementation**

4.54.1 Detailed Description

Symbol Data.

Definition at line 918 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.55 QuestradeAPI::TickType Struct Reference

Tick types for last trade of the quotes.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value {
  Undefined = 0, **Up** = 1, **Equal** = 2, **Down** = 3,
  Count }

  Values for TickType.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of TickType to String.
- static Value fromString (const std::string &name)

  Converts value of TickType from String.

4.55.1 Detailed Description

Tick types for last trade of the quotes.

Definition at line 455 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h

4.56 QuestradeAPI::UnderlyingMultiplierPair Struct Reference

Underlying information for the option.

`#include <CommonStructures.h>`

Public Member Functions

- uint64 getMultiplier () const

  Multiplier.
- std::string getUnderlyingSymbol () const

  Underlying symbol name.
- uint64 getUnderlyingSymbolId () const

  Underlying symbol ID.
- std::string dumpToJson () const

  Dumps data in json format (use for debug purposes only, as json is generated)
- UnderlyingMultiplierPair ()

  Creates invalid structure (added for use with STL containers)

Friends

- class **UnderlyingMultiplierPairImplementation**

4.56.1 Detailed Description

Underlying information for the option.

Definition at line 817 of file CommonStructures.h.

The documentation for this struct was generated from the following file:

- CommonStructures.h

4.57 QuestradeAPI::UserAccountType Struct Reference

Types of accounts.

`#include <CommonEnumsDefines.h>`

Public Types

- enum Value {
  Undefined = 0, **Cash** = 1, **Margin** = 2, **TFSA** = 3,
  RRSP = 4, **SRRSP** = 5, **LRRSP** = 6, **LIRA** = 7,
  LIF = 8, **RIF** = 9, **SRIF** = 10, **LRIF** = 11,
  RRIF = 12, **PRIF** = 13, **RESP** = 14, **FRESP** = 15,
  FX = 16, **FXD** = 17, **Count** }

  Values for UserAccountType.

Static Public Member Functions

- static const char * toString (Value value)

  Converts value of UserAccountType to String.
- static Value fromString (const std::string &name)

  Converts value of UserAccountType from String.

4.57.1 Detailed Description

Types of accounts.

Definition at line 475 of file CommonEnumsDefines.h.

The documentation for this struct was generated from the following file:

- CommonEnumsDefines.h