AlgoTrader

AlgoTrader 6.4

Institutional trading technology for digital and traditional assets

We are constantly adding new features to extend the AlgoTrader platform and provide the best trading experience for banks, crypto funds, brokers, hedge funds and other financial institution. Find enclosed the instructions on how to upgrade your AlgoTrader installation.

algotrader.com

Upgrade: AlgoTrader 6.4

Below are the instructions on how to upgrade your AlgoTrader 6.3 installation to the 6.4 release.

Market Data Subscriptions as Ephemeral Subscriptions

Market data subscriptions are now ephemeral subscriptions by default. This change carries several implications on how the subscription will be managed by the system now:

- Each API client (REST/WS, InboundFIX, Strategy service) becomes a sole owner of individually created MD subscription

- It is possible to request multiple subscriptions of the same logical key (e.g. SecurityId + exchange AccountId + Portfolio) by various "actors" and manage them individually - e.g. closing a market data Subscription A of: SecurityId 1, Account 2, Portfolio 3 will not take effect for the Subscription B with the same underlying data source (Security 1, Account 2, Portfolio 3)

- All successful market data requests will now return a String subscriptionId that can be used by client to manage the subscription individually

- AT system has several policies in place to clean up unused subscriptions (unless they were created as 'durable' - described in next points):

  – For REST/WS subscriptions, Market data subscriptions will be cleared after WS subscribers leaves and do not come back after (preconfigured) timeout period

  – For InboundFIX subscriptions, Market data subscriptions will be cleared after FIX session between the client and AT is logged out / disconnected

  – For UI subscriptions, Market data subscriptions will be cleared after the browser window is closed, or user has disconnected (after a preconfigured grace period)

- Subscription entity will not be tracked in the database by default

- It is possible, however, to set a durability option (durable=true) on a market data request

  – All Subscriptions marked as durable will be persisted into DB and recreated upon system restart

  – All Subscriptions marked as durable will not be affected by AT's subscription cleanup policies

- For the UI users, it is possible to mark the requested subscription as 'durable' in the Market data tab. Note: Such subscription will remain durable only for the given browser instance (i.e. entering AT UI via a different browser or different workstation will require a manual re-subscription to Market data streams)

- For API users, there are additional parameters that should be used to mark Market data subscriptions as 'durable':

  – REST/WS: set additional request parameter boolean 'durable' to true/false

  – InboundFIX: use additional custom tag 20062 on MarketDataRequest (V) and set it to 0 (false) or 1 (true)

- For StrategyService users it is not possible to set the requested subscription as durable. However, it should be enough to simply setup a subscription requesting logic as part of the strategy code. Placing such logic in the StrategyService onStart() phase will make the subscription effectively durable

Subscriptions in the DB

With above changes for Ephemeral subscription, the subscription model in the DB will be changed in a way that does not allow a migration from AT v 6.3. This means that all Market data subscriptions will be deleted from the DB during the upgrade and will have to be resubscribed again by the user after starting AT 6.4. It is up to the user to mark the new subscriptions as 'durable' or not (if set to durable, they will be stored in the DB, more details to be found in the "Market data subscriptions as Ephemeral subscriptions" section above).

Bar Aggregation Changes

AT now enables more granular and flexible bar aggregation possibilities.

- It is possible to subscribe to a bar stream per given Security@Exchange directly (as opposed to turning bar aggregation on/off globally, on a system level)

- It is possible to subscribe to different bar aggregation sizes at the same time and select which Securities should be aggregated into which bar size

- In order to subscribe to bars, use new Subscription endpoints:

  – **REST/WS**: /rest/subscription/marketdata/bar

  – **InboundFIX**: MarketDataRequest msg (V), tag 20051 with value "BAR", (optional) tag 20060 describes the bar size in seconds

  – **StrategyService**: SubscriptionService available there now has 4 new methods available:

  ```
  /**
   * Subscribes to market data for given securityId and accountId that will be aggregated into Bars.
   * This method looks at the global config for bar settings for Bar aggregation:
   * <li>
   *    <ul>barAggregation.barSize</ul>
   *    <ul>barAggregation.frontFillEnabled</ul>
   *    <ul>barAggregation.aggregationPriceType</ul>
   * </li>
   */
  void subscribeBar(String portfolioName, long securityId, long accountId);
  /**
   * Subscribes to market data for given securityId and accountId that will be aggregated into Bars.
   * This method looks at the global config for bar settings for Bar aggregation:
   * <li>
   *    <ul>barAggregation.frontFillEnabled</ul>
   *    <ul>barAggregation.aggregationPriceType</ul>
   * </li>
  ```

```
*/

void subscribeBar(String portfolioName, Duration duration, long securityId, long accountId);

/**

 * Unsubscribes previously subscribed market data subscription.

 */

void unsubscribeBar(String portfolioName, long securityId, long accountId);

/**

 * Unsubscribes previously subscribed market data subscription.

 */

void unsubscribeBar(String portfolioName, Duration duration, long securityId, long accountId);
```

GET /rest/historicaldata/bidAsksByMinDate - now also accepts a `symbol` and an `assetClass` (if you want to refer to a security not through its ID)

PUT /rest/subscription/marketdata/bar - now also accepts a `symbol` and an `assetClass`

PUT /rest/subscription/marketdata/subscribe - now also accepts a `symbol` and an `assetClass`

PUT /rest/subscription/marketdata/orderBook - now also accepts a `symbol` and an `assetClass`

Code Changes

All methods in public interface related to strategy (for example getPositionsByStrategy(String)) and marked as @Deprecated in Algotrader release 6.3.0 have now been removed. Please use methods that have Portfolio-based naming convention (for example getPositionsPortfolio(String)).

Generic events with portfolio values will no longer be broadcasted. They were generated cyclically as described in portfolio.savePortfolioValuesIntervalSeconds property. To find latest portfolio value, refer directly to ch.algotrader.service.PortfolioValueService. Portfolio values are still saved to influxdb if configured in portfolio.savePortfolioValuesIntervalSeconds - no changes here.

Removed internal and downstream Tick support so onTick(TickVO) is no longer supported in the strategy or in internal parts of AlgoTrader for custom or main release versions.

It can be replaced by a mix of onTrade(TradeVO) and onQuote(QuoteVO) usage as those reflect all the necessary data and timing of the previous tick publishing as ticks were originally generated from those in most cases anyway. Loading and feeding TickVO events/ticks to AlgoTrader is still possible using Influx and CSV however they will be translated into trades and quotes internally and those will be propagated to downstream instead. In addition to that sample data and historical data support for ticks was also removed and where possible got replaced by more differentiated historical data support such as bids, asks, bidAskQuotes and trades on the connector level.

The Portfolio Value Table was removed from the DB

Database table portfolio_value has been removed from the DB. Portfolio values are still available in InfluxDB (measurement: portfolio_value). To access portfolio value records, use PortfolioValueDao or LookupService, for example:

LookupService#findMany(java.lang.Class<T>, com.hazelcast.query.Predicate<K,T>)

LookupService is also available from strategy context, by using:

StrategyService.getLookupService().findMany(PortfolioValue.class, Predicates.alwaysTrue());

FIX API Changes

Tag **Account** (Tag = 1, Type: String) now becomes **PortfolioID** (foremly **AccountID**)

Tag **PartyID** (Tag = 448: Type: String) now becomes **AccountID**

Please refer to our FIX API reference for details: FIX API

REST API Changes

The following endpoint have been removed (it was marked as deprecated since Algotrader release 6.3.0)

GET /rest/strategy - use GET /rest/portfolio/all instead

 GET /rest/portfolio - moved to GET /rest/portfolio/all

 GET /rest/lookup/portfolioById/{portfolioId} - use GET /rest/portfolio/{id} instead

 GET /rest/lookup/portfolioByName/{name} - use GET /rest/portfolio?name={name} instead

GET /rest/marketData/orderBook/{securityId} - use GET /rest/marketData/currentValue/{securityId}/{fullClassName} instead

STOMP Event Topic Changes

 /topic/bid.<security_id> - use /topic/bid.<account_id>-<security_id> instead

 /topic/ask.<security_id> - use /topic/ask.<account_id>-<security_id> instead

 /topic/bid-ask-quote.<security_id> - use /topic/bid-ask-quote.<account_id>-<security_id> instead

 /topic/trade.<security_id> - use /topic/trade.<account_id>-<security_id> instead

 /topic/order-book.<SecurityId> - use /topic/order-book.<account_id>-<security_id> instead

 /topic/bar.<SecurityId> - use /topic/bar.<account_id>-<security_id> instead

 /topic/tick.<security_id> - decomissioned, use bid/ask/bid-ask-quote instead

Reference Data Changes

A new column on the security table: TRADING_VIEW_ID, is populated during the initial Reference Data Download.

If you are migrating from an existing Reference Data, this column will be populated with a null value which may result in Trading View Chart, not displaying. Re-download reference data to populate the values.

Portfolio Values and Booking Changes

Please note that from version 6.4.0 all margin FOREX trades will become positions. Any trades made on previous versions of AlgoTrader will remain in cash balances.

We have also changed the way margin trades are booked, they no longer affect cash balances until the position is (partially) closed, at which time the realzied P&L is reflected in the cash balance.

The new portfolio value definitions are:

🖊 cashBalance: The sum of cash balances amount available to the portfolio

🖊 marketValue: Market value of all open positions (including Forex positions)

🖊 netLiqValue: Cash balance + market value

🖊 realizedPL: Realized P&L of all positions

🖊 unrealizedPL: Unrealized P&L of all positions

Configuration Changes

The configuration property deribit.fix.session.targetCompID was removed as the target name is defined in the Deribit documentation and should be constant - "DERIBITSERVER".

The default value of configuration property deribit.fix.session.senderCompID was also removed.

Users now have to define the SenderCompID themselves, creating an arbitrary name for the session.

Side note: If this is left empty SenderCompID name is not replaced in the FIX cfg file and the wildcard name is used literally "${deribit.fix.session.senderCompID}".

Pre-trade checks / post-trade actions no longer configurable by CSV files

CSV files are no longer supported. Risk UI / JSON files should be used instead.

** LINK TO DOC? **

Misc

Improving the reference data download for BitMEX accounts, in order to offer all tradable securities. The changes will take place and missing securities will be added after the download is triggered for each BitMEX account (e.g. RDM tab -> Download Reference Data -> pick BitMEX account -> Download).

Algotrader's Jetty server now binds by default only to localhost, use jetty.host property to bind to other hosts (0.0.0.0 bind to all).

For inquiries, please contact

Support Team

T +41 44 291 14 85

support@algotrader.com

AlgoTrader

Head Office
Letzigraben 89
8003 Zurich
Switzerland

Contact
info@algotrader.com
+41 44 291 14 85
algotrader.com

About AlgoTrader
AlgoTrader is the global leader in institutional trading technology for both digital and traditional assets.