Python based MT5 to Interactive Brokers TWS sync.

Background

There are existing tools to do synchronisation between MT and TWS, but I've found them to be unreliable or to be a heavy footprint (slowing down the terminal or using too much resource) so wish to build one that is standalone, clean, fast and robust in Python.

Features:

Settings file to specify port for TWS, MT5 path, Email from, to, server host name/ip, server port, Telegram ID, Hash and Channel to send alerts to, refresh interval (MT5), refresh interval (TWS) from 0.5 to 10 seconds (to 60 seconds for TWS).

Mapping table. Symbol names. Lot size in MT5 to quantity in TWS, Max Lots. (Max Lots works in either Long or Short position and is a safeguard to prevent unacceptable orders)

Pulls current trades data from MT5 to database table (including price, lot size, TP and SL levels). Looks at the sum of buy/sell positions for each pair that exists in the terminal.

Pulls current positions from TWS to database table.

Compares with internal database with TWS table. If any change then make adjustment in TWS (buy/sell order). If an order if placed and filled then update in the table for the order filled priced. This is used later to determine slippage/variation from MT5 and TWS prices.

Any new TWS orders are labelled 'FX Sync' (Order Ref field in TWS)

When a trade in MT5 is closed then it updates a 'closed' field in the Orders table as well as the price (and other relevant data reported from system for closed order).

Audio, Telegram and E-mail Alert if MT5 terminal or TWS is unavailable (except for weekend hours) or in case of any error when placing orders. Telegram notice of placed or closed orders.

Display in a window (datagrid control) the total lots for each position of MT5 and the total quantity of the current positions for each symbol of TWS. (Note, ignore to show any other symbols that positions may exist for from TWS – i.e. we only show MT5 matched positions).

Symbol (MT5)	Lots (+/-)	Symbol (TWS)	Quantity	MT5 Avg. Price	TWS Avg. Price

Below this Datagrid show a similar grid for closed positions (that is scrollable to view back for last 14 days)

Symbol (MT5)	Buy / Sell Lots (+/-)	Symbol (TWS)	Quantity (+/-)	MT5 Profit (+/-)	TWS Profit (+/-)

At top of window show on-line status of each program as indicator (green for online or red indicator). Also show the date/time of the last refresh for each program.

Have a simple way to make a report of the order filled price in MT5 and TWS and show difference +/- $ and % for each order. Also to show total profit/loss per order on MT5 and TWS. (Later will also include broker fees).

Notes:

Program needs to be robust (can run for weeks). Have some degree of error handling / trapping.

Limit total number of TWS orders per hour to 5. This is really a safety to prevent anything 'funky' happening.

In case of loss of connectivity to MT5 then don't close the current TWS positions until MT5 connectivity is restored.

Use Logging to record all change in MT5 positions and order placement and outcome in TWS.

It's possible buy and sell positions of a symbol are opened simultaneously in MT5. For this we would calculate the total lots based on buy – sell before passing quantity amount to TWS.

If a symbol from MT5 doesn't exist in the mappings table then no trade is passed through to TWS.

Order placement potential problems

Hard failures may include (but not limited to):

- Instrument not available for trading
- No shares available for shorting (not apply to ForEx)
- Insufficient funds
- Error in order input

Soft failures may include (but not limited to):

- Exchange unavailable
- TWS or connection to broker offline
- Outside of normal trading hours

Database tables

- Portfolio MT5 (Current open positions)
- Portfolio TWS
- Orders (Orders waiting to be executed/completed). Details of orders (prices etc).

In the case of a hard failure we report the result in the log / telegram / email and the order's status is marked as 'Failed' with the returned error reason.

In the case of a soft failure we keep the order in the queue but change status to 'retrying' and keep re-trying every 5 seconds (in the first hour of trading) or less often at other times.

General:

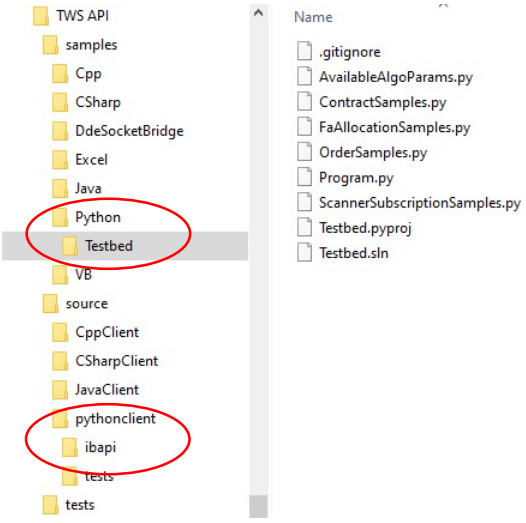
Code should be clean and easy to read. Include comments and good use of variable/function names.

Use version control (show v x.xx or similar) on each revision being shared.

Programming Help

TWS API:

Interactive Brokers TWS supports a comprehensive API (https://www.interactivebrokers.com/en/index.php?f=5041) which allows to custom develop tools to simplify or automate actions. The free API download from this site includes example code, fully compiled examples in Python, VB and Excel VBA and is well documented.



Further, this articles explains very well with many examples of API use with Python: https://algotrading101.com/learn/interactive-brokers-python-api-native-guide/

These pages also provide some good examples and information relating to interacting with TWS.

https://tradersacademy.online/trading-topics/trader-workstation-tws/accessing-the-tws-python-api-source-code

https://python-trading.com/python-trading-2-how-to-connect-to-interactive-brokers-tws-with-pycharm-and-the-api/

This is a mini course for learning about Python based interaction with TWS: https://tradersacademy.online/trading-courses/python-tws-api (recommend 1,2,3,4 and 6)

This is the developer's API reference guide: https://interactivebrokers.github.io/tws-api/introduction.html (click Python at the top of the page for Python specific examples)

Code snippets:

TWS Portfolio pulling:

With reference to http://interactivebrokers.github.io/tws-api/account_updates.html#acct_update_request and http://interactivebrokers.github.io/tws-api/positions.html#position_request

Placing an order

Place the order (assume all orders are at Market for sync / copy trading): https://interactivebrokers.github.io/tws-api/order_submission.html

Attach any bracket orders *(if relevant to include TP/SL to the order)*: https://interactivebrokers.github.io/tws-api/bracket_order.html

Use: The openOrder callback (https://interactivebrokers.github.io/tws-api/order_submission.html#open_order) to check the order status (i.e. the order was successfully received). If success/confirmed, warning or error then record in log, telegram (and optional email).

If error, record error in Log + Telegram (optional email) if 'Instrument not available for trading' or any other error that may be returned when placing the order.