TradingMate Documentation

Release 0.1.0

Alberto Cardellini

May 09, 2019

CONTENTS

CHAPTER

ONE

INTRODUCTION

TradingMate is an autonomous trading system that uses customised strategies to trade in the London Stock Exchange market. This documentation provides an overview of the system, explaining how to create new trading strategies and how to integrate them with TradingMate. Explore the next sections for a detailed documentation of each module too.

1.1 System Overview

TradingMate is a portfolio manager with the goal to help traders to record their trades and deals in the stock market providing useful statistics and metrics to help backtest and review trade performance. It offers a simple user interface that provide information about the current asset value, the overall profit/loss indicator and the trade history.

1.1.1 TradingMate

TradingMate is the main entiy used to initialised all the components. It is the link between the user interface and the data.

TODO

1.2 Modules

Each section of this document provides the source code documentation of each component of TradingMate.

1.2.1 TradingMate

class TradingMate.**TradingMate**
Main class that handles the interaction between the User Interface and the underlying business logic of the whole application

on_close_view_event()
Callback function to handle close event of the user interface

on_delete_last_trade_event()
Callback function to handle delete of last trade request

on_manual_refresh_event()
Callback function to handle refresh data request

on_new_trade_event(*new_trade*)
Callback function to handle new trade event

on_open_portfolio_event(*filepath*)
Callback function to handle request to open a new portfolio file

on_save_portfolio_event(*filepath*)
Callback function to handle request to save/export the portfolio

on_save_settings_event(*config*)
Callback to save edited settings

on_set_auto_refresh(*enabled*)
Callback function to handle set/unset of auto refresh data

on_show_settings_event()
Callback to handle request to show the settings panel

on_update_live_price()
Callback function to handle update of stock prices data

register_callbacks()
Register all the callback functions

setup_logging()
Setup the global logging settings

start()
Start the application

1.2.2 Model

The `Model` module contains the business logic and the data management of TradingMate.

Holding

class Model.Holding.**Holding**(*symbol*, *quantity*, *open_price=None*)

add_quantity(*value*)
Add or subtract (if value is negative) the value to the holding quantity

Portfolio

class Model.Portfolio.**Portfolio**(*name*, *config*)

clear()
Reset the Portfolio clearing all data

compute_avg_holding_open_price(*symbol*, *trades_list*)
Return the average price paid to open the current positon of the requested stock. Starting from the end of the history log, find the BUY transaction that led to to have the current quantity, compute then the average price of these transactions

get_cash_available()
Return the available cash quantity in the portfolio [int]

get_cash_deposited()
Return the amount of cash deposited in the portfolio [int]

get_holding_last_price(*symbol*)
Return the last price for the given symbol

get_holding_list()
Return a list of Holding instances held in the portfolio sorted alphabetically

get_holding_open_price(*symbol*)
Return the last price for the given symbol

get_holding_quantity(*symbol*)
Return the quantity held for the given symbol

get_holding_symbols()
Return a list containing the holding symbols as [string] sorted alphabetically

get_holdings_value()
Return the value of the holdings held in the portfolio

get_name()
Return the portfolio name [string]

get_open_positions_pl()
Return the sum profit/loss in £ of the current open positions

get_open_positions_pl_perc()
Return the sum profit/loss in % of the current open positions

get_portfolio_pl()
Return the profit/loss in £ of the portfolio over the deposited cash

get_portfolio_pl_perc()
Return the profit/loss in % of the portfolio over deposited cash

get_total_value()
Return the value of the whole portfolio as cash + holdings

is_trade_valid(*newTrade*)
Validate the new Trade request against the current Portfolio

reload(*trades_list*)
Load the portfolio from the given trade list

DatabaseHandler

class Model.DatabaseHandler.**DatabaseHandler**(*config*)
Handles the IO operation with the database to handle persistent data

add_trade(*trade*)
Add a trade to the database

get_db_filepath()
Return the database filepath

get_trades_list()
Return the list of trades stored in the db

read_data(*filepath=None*)
Read the trade history from the json database and return the list of trades

- **filepath**: optional, if not set the configured path will be used

remove_last_trade()
Remove the last trade from the trade history

write_data(*filepath=None*)
Write the trade history to the database

StockPriceGetter

class Model.StockPriceGetter.**StockPriceGetter**(*config*, *onNewPriceDataCallback*)

convert_market_to_alphavantage(*symbol*)
Convert the market (LSE, etc.) into the alphavantage market compatible string i.e.: the LSE needs to be converted to LON

task()
The task done by this thread - override in subclasses

1.2.3 UI

The UI module contains the components that compose the User Interface of TradingMate.

View

class UI.View.**View**

ShareTradingFrame

class UI.ShareTradingFrame.**ShareTradingFrame**(*parent*)

AddTradeDialogWindow

class UI.AddTradeDialogWindow.**AddTradeDialogWindow**(*master*, *confirmCallback*)

WarningWindow

class UI.WarningWindow.**WarningWindow**(*master*, *title*, *message*)

Widgets

class UI.Widgets.**DatePicker**(*master*, *dateSelected*)

focus_set()
Direct input focus to this widget.

If the application currently does not have the focus this widget will get the focus if the application gets the focus through the window manager.

1.2.4 Utils

The Utils module contains all the utlity components.

1.2.5 ConfigurationManager

class Utils.ConfigurationManager.**ConfigurationManager**
Class that loads the configuration and credentials json files exposing static methods to provide the configurable parameters

get_alpha_vantage_api_key()
Get the alphavantage api key

get_alpha_vantage_base_url()
Get the alphavantage API base URI

get_alpha_vantage_polling_period()
Get the alphavantage configured polling period

get_credentials_path()
Get the filepath of the credentials file

get_editable_config()
Get a dictionary containing the editable configuration parameters

get_trading_database_path()
Get the filepath of the trading log file

load_credentials()
Load the credentials file

save_settings(*config*)
Save the edited configuration settings

1.2.6 TaskThread

class Utils.TaskThread.**TaskThread**
Thread that executes a task every N seconds

cancel_timeout()
Cancel the timeout and run the task

enable(*enabled*)
Disable/enable this thread

run()
Method representing the thread's activity.

You may override this method in a subclass. The standard run() method invokes the callable object passed to the object's constructor as the target argument, if any, with sequential and keyword arguments taken from the args and kwargs arguments, respectively.

setInterval(*interval*)
Set the number of seconds we sleep between executing our task

shutdown()
Stop this thread

task()
The task done by this thread - override in subclasses

1.2.7 Trade

class Utils.Trade.**Trade**(*date_string*, *action*, *quantity*, *symbol*, *price*, *fee*, *sdr*)

1.2.8 Utils

class Utils.Utils.**Callbacks**
An enumeration.

class Utils.Utils.**Actions**
An enumeration.

class Utils.Utils.**Messages**
An enumeration.

class Utils.Utils.**Markets**
An enumeration.

1.3 Changelog

All notable changes to this project will be documented in this file.

The format is based on Keep a Changelog, and this project adheres to Semantic Versioning.

1.3.1 [1.0.0] 2019-05-03

Added

- Initial release

CHAPTER

TWO

TRADINGMATE

TradingMate is a portfolio manager for stocks traders. It lets you record all your trades with a simple and basic interface, showing the current status of your assets and the overall profit (or loss!)

CHAPTER

THREE

DEPENDENCIES

- Python 3.4+
- Tkinter: https://docs.python.org/3/library/tk.html
- AlphaVantage: https://www.alphavantage.co/

View file `requirements.txt` for the full list of python dependencies.

CHAPTER

FOUR

INSTALL

After cloning this repo, to install TradingMate simply run:

```
./trading_mate_ctrl install
```

(This will require super-user access)

The required dependencies will be installed and all necessary files installed in /opt/TradingMate by default. It is recommended to add this path to your PATH environment variable.

CHAPTER

FIVE

SETUP

TradingMate uses AlphaVantage to fetch markets data online:

- Visit AlphaVantage website: `https://www.alphavantage.co`
- Request a free api key
- Insert these info in a file called `.credentials` This must be in json format .. code-block:: guess

  { “av_api_key”: “apiKey”

  }
- Copy the `.credentials` file in the `$HOME/.TradingMate/data` folder
- Revoke permissions to read the file by others .. code-block:: guess

  cd $HOME/.TradingMate/data sudo chmod 600 .credentials

Configuration file

The `config.json` file is in the `$HOME/.TradingMate/config` folder and it contains several parameters to personalise how TradingMate works. These are the descriptions of each parameter:

- **general/trading_log_path**: The absolute path of the trading log where the history of your trades are saved
- **general/credentials_filepath**: File path of the .credentials file
- **alpha_vantage/api_base_uri**: Base URI of AlphaVantage API
- **alpha_vantage/polling_period_sec**: The polling period to query AlphaVantage for stock prices

CHAPTER

SIX

RUN

TradingMate can be controlled by the `trading_mate_ctrl` shell script. The script provides commands to perform different actions:

6.1 Start TradingMate

```
./trading_mate_ctrl start
```

6.2 Stop TradingMate

Closing the main window will stop the whole application. You can also use the command:

```
./trading_mate_ctrl stop
```

CHAPTER

SEVEN

TEST

Test can't run with the installed script. You can run the test from a "workspace" environment with:

```
./trading_mate_ctrl test
```

You can run the test in Docker containers against different python versions:

```
./trading_mate_ctrl test_docker
```

CHAPTER

EIGHT

DOCUMENTATION

The Sphinx documentation contains further details about each TradingMate module with source code documentation of each component.

Read the documentation at:

https://tradingmate.readthedocs.io

You can build it locally from the "workspace" root folder:

```
./trading_mate_ctrl docs
```

The generated html files will be under `doc/_build/html`.

CHAPTER

NINE

CONTRIBUTING

I appreciate any help so if you have suggestions or issues open an Issue for discussion. If you can contribute please just open a pull request with your changes. Thanks for all the support!

PYTHON MODULE INDEX

m

t

u

INDEX

H

I

L

M

O

P

R

S

T

U

V

W