

Getting started

About TickTrader Algo Studio

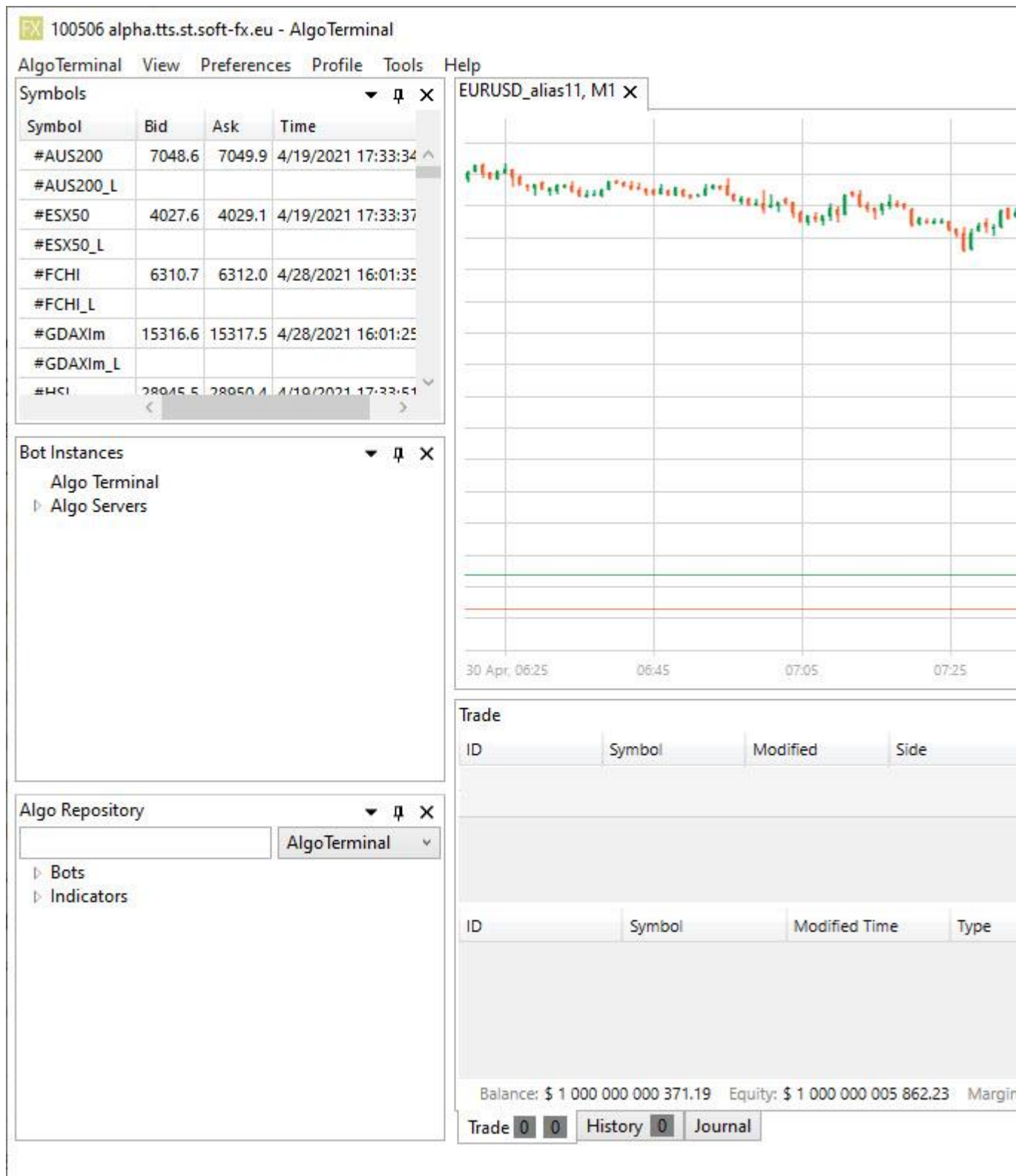
TickTrader Algo Studio application is developed for automated trading. TickTrader Algo Studio includes products that allow developing and running algorithmic trading systems.

Overview

The application consists of several related modules and their submodules:

- **TickTrader Algo Terminal:** a terminal that allows a user to view charts of current quotes, storage of bots and indicators, set custom symbols, test developed bots and find optimal settings for bot parameters. TickTrader Algo Terminal submodules:
 - Terminal: terminal with quotes charts, storage of bots and trading indicators, a module for managing connected algo servers;
 - Optimizator: a tool that allows you to run multiple bots and find the optimal parameters for launching bots . It has functionality for collecting statistics on all launches, saves launch results and provides better bot customization.
 - Symbol Manager: a tool that allows you to download historical quotes from the server, create and configure custom symbols for testing strategies.
- **Algo Server:** a module that allows a user to setup the Algo Studio process on a remote server to launch and operate trading bots. Submodules:
 - Algo Server: a service launched on a remote server to manage the constant bots' execution;
 - TT Algo Web Console: a page on the Internet, necessary to visualize the performance of bots and simplify their management;
 - Algo Configurator: application for configuring TickTrader Algo Server and TickTrader Algo Web Console.

TickTrader Algo Studio can be operated under MS Server 12/16 for the server part and MS Win 7/10 for the client part. Hardware requirements are limited by operating system requirements.



How do I get started?

1. [Install TickTrader Algo Studio.](#)
2. [Log into the Terminal.](#)
3. [Register TT Algo Server Service in the TickTrader Algo Terminal.](#)
4. [Create a bot/indicator.](#)

5. [Launch a bot.](#)

What's new in TickTrader Algo Studio

This section describes new features in **TickTrader Algo Studio**.

Trading

Account types

The following types of trading accounts are available on the TickTrader platform.

Type of trading	Trading accounts
Margin	Gross
	Net
Exchange	Cash

Margin accounts: Gross, Net

Trading on Margin accounts is performed using leverage. Leverage allows you to trade a larger amount of funds than is available on the account.

Order execution. Orders are automatically matched with opposite orders available in the order book. An order can be executed in one block or broken into multiple transactions.

Swap and commission. The Swap value on Gross and Net accounts is presented in points but is charged in profit currency according to the algorithm of profit calculation. Swap and commission are charged in proportion to the volume of a closed order. If an order remains open overnight and is partially executed the next day, swap applies only to the filled order volume. Commission is charged at position opening and immediately recorded to History.

Gross account

On Gross accounts, all positions opened on a financial instrument are separate operations. Each position is a separate record. You can open multiple positions, including opposite ones, on one financial instrument.

- Stop Loss / Take Profit orders and Trailing Stop are available.
- P/L is calculated separately for each position.
- You can close positions using the *Close position* command.
- When you reverse a position, the current position is closed and the opposite position of the same volume is opened.

Net account

On Net accounts, all positions opened on a financial instrument are aggregated into one Net position. There can be only one entry for each financial instrument in the list of positions.

The Net position volume changes as you open new positions on the same instrument. You cannot open opposite positions on the same financial instrument. To close a position, open an opposite position on the same instrument.

Stop Loss / Take Profit orders and Trailing Stop are not available. To profit from a position, place an opposite Limit order. Similarly, to limit the possible loss from a position, place an opposite Stop order.

The volume-weighted open price of an aggregated Net position is calculated as follows:

$$\text{Total Open Price} = ((\text{Open price of order 1} \times \text{Volume order 1}) + (\text{Open price of order 2} \times \text{Volume order 2})) / (\text{Volume order 1} + \text{Volume order 2})$$

Positions on Gross and Net accounts

Criteria	Gross account	Net account
Number of positions on the same financial instrument	Many	One
Opposite positions (Buy and Sell) on the same financial instrument	✓	✗
Increasing position volume	✗	✓
Decreasing position volume	✗ Partial closing of a position decreases its volume. In the Order Modification window, specify a smaller volume and click Close by Market . The specified <i>Trade Volume</i> is closed, and the position remains open with the <i>Remaining Volume</i> .	✓ To decrease the Net position, open an opposite position of a larger volume.
Position reversal	To reverse a position, use the Reverse Trade Position command. The current position is closed and the opposite position of the same volume is opened.	To reverse a Net position, open an opposite position of a larger volume.
Closing position	Position is closed using the Close position command.	Net position is closed by opening an opposite position of the same volume.

Cash account

On Cash accounts, one asset is exchanged for another at the currently available exchange rate.

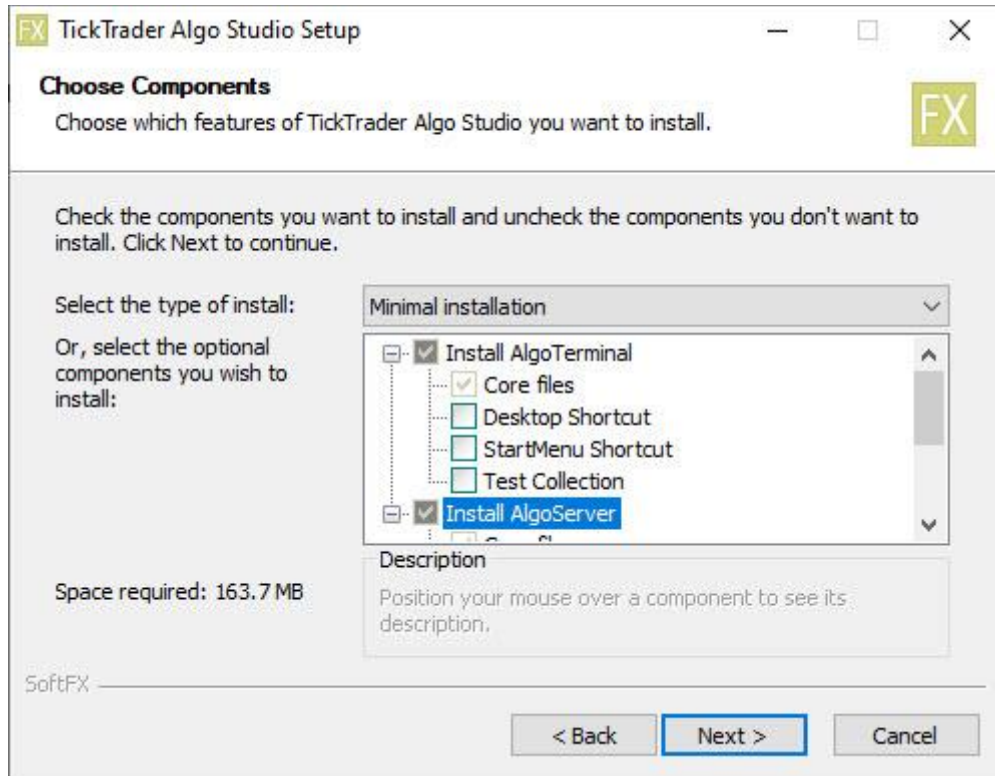
Orders are executed as follows: asset X is exchanged for asset Y at the exchange rate X/Y available at the time of the transaction. As a result, the values of balances expressed in assets X and Y on the trading account change. The amount of assets on Cash account is recalculated after each exchange or balance operation.

Only Limit orders are available on Cash accounts. You can always expect pending orders to be executed at the requested or better price.

[Install TickTrader Algo Studio](#)

How to install TickTrader Algo Studio

1. Download the installation file of **TickTrader Algo Studio**.
2. Run the downloaded .exe file and follow the Installation Wizard steps. Click **Next**.
3. Read the License Agreement, select **I accept the agreement**, and then click **Next** to proceed to the next installation step.
4. Select which components and features of **TickTrader Algo** you want to install. Click **Next**.



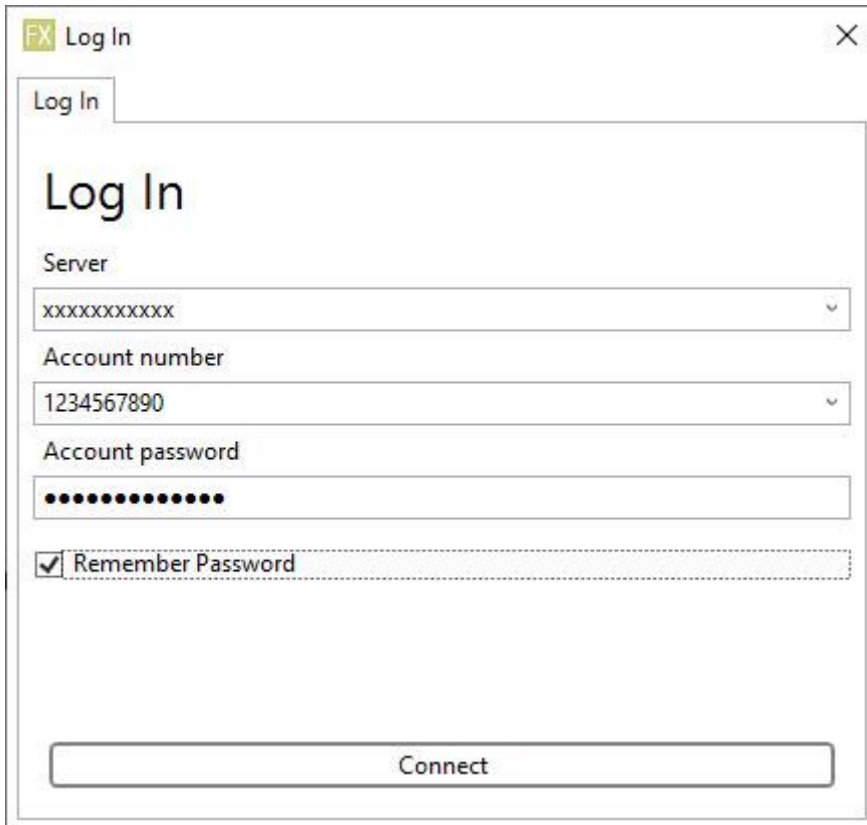
5. Select the destination folders and click **Next**.
6. Wait until Setup downloads the necessary installation files and configures the program.
7. When the installation process is complete, click **Next**.
8. Click **Finish** to exit Setup.
9. In case you selected AlgoServer as a component to install then after the finishing of the application installation process the AlgoServer Configurator window appears.
10. Register the Algo Server in the Algo Terminal.

Update the application

The application can be updated by launching the TickTrader Algo Studio .exe file.

Terminal

Log into TickTrader Algo Terminal



FX Log In

Log In

Log In

Server
xxxxxxxxxx

Account number
1234567890

Account password
●●●●●●●●●●

☒ Remember Password

Connect

To log into the Terminal:

1. Enter the server name or select it from the list.
2. Enter account details: Login and Password.
3. The **Remember Password** option allows you to save the entered password so that you do not have to enter it every time you start the Terminal or switch between accounts. If you disable this option, the Terminal will not save the password for this account, and you will have to enter it the next time you log in.

After successful login, the Terminal will connect to a TickTrader Server.

The account number and server name are indicated in the title bar of the window.

FX 100015 beta.tts.st.soft-fx.eu - AlgoTerminal

Switch between trading accounts

To switch between trading accounts, select the account number in the **Log In** window.

FX Log In

Log In

Log In

Server

Soft-Fx Staging Beta

Account number

100015

- Beta STG 100015
- Beta STG 100017

☒ Remember Password


Connect

Log out the Terminal

To log out the Terminal click **Log Out** on the **Algo Terminal** menu.

Exit the Terminal

You can exit TickTrader Algo Terminal in one of the following ways:

- Click the **Close** button .
- On the **Algo Terminal** menu, click **Exit**.

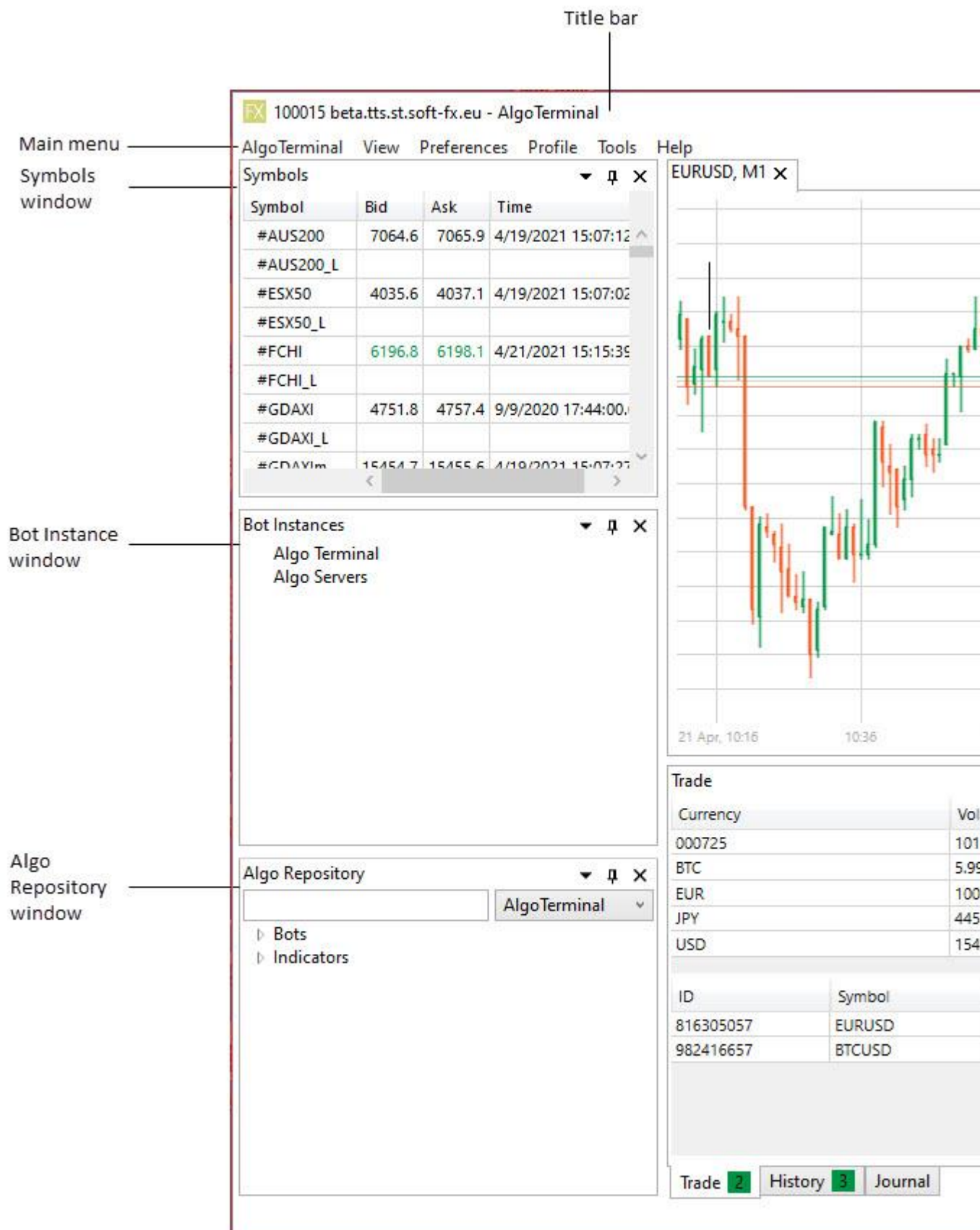
User Interface

This section includes topics that describe the main features of the user interface of the Terminal and its elements.

Main application window

The main application window is your working space where you can access the main menu and toolbar, arrange windows in various ways, close, add, or detach them.

When you open the Terminal after installation, the application appears as shown in the screenshot.



Title bar: The title bar of the window contains the program icon and the application name, account login, username and server name (if you are logged in), the active chart name (if there is an open chart), and the **Minimise**, **Maximise/Restore Down** and **Close** buttons.



Main menu: The main menu gives you quick access to the major features of TickTrader Algo Terminal.

Status bar: The status bar is located at the bottom of the application window.



The status bar displays the following information:

- **Connection status:** There is a visual indication of your connection status to the trading server.
 - **Online:** TickTrader Algo Terminal is connected to the trading server.
 - **Connecting:** TickTrader Algo Terminal is connecting to the trading server.
 - **Offline:** TickTrader Algo Terminal is not connected to the trading server.
- **Trading server:** Trading server name.
- **Protocol type:** Protocol type.

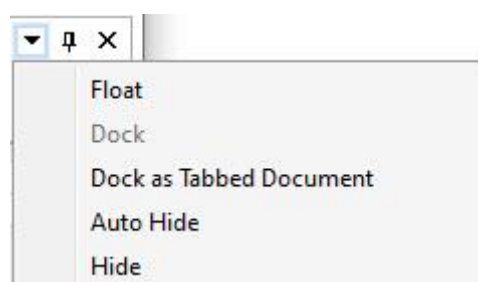
Manage windows

You can customize the appearance and behavior of application windows to suit your needs. In the Main workspace, you can:

- Dock windows to any location within the Main workspace.
- Float windows, that is, drag them out of the main application window.
- Automatically hide dockable windows.
- Hide and show windows.

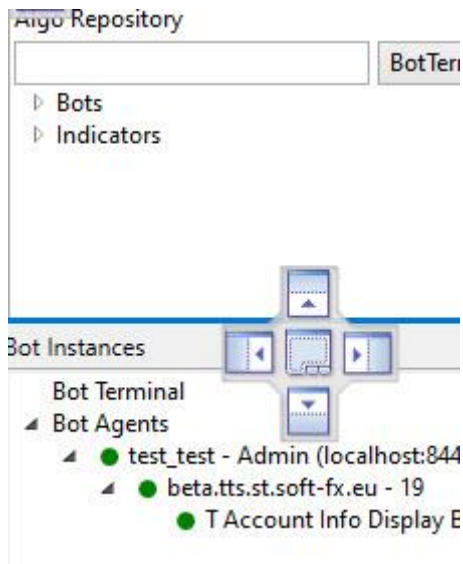
You can view available **window states** and switch between them by right-clicking the title bar of the window or by clicking ▼. The state of windows affects how you can arrange them in the Main workspace.

Tabs and windows inside the main **TickTrader Algo Studio** window may be opened in various display modes. Right-click the title bar of the tab/window to see available display mode options: **Float**, **Dock**, **Dock as Tabbed Document**, **Auto Hide**, **Hide**.




Dock windows

TickTrader Algo Studio uses the docking panel technology. A dock window is a window that can be floating or attached to the edge of the parent window within the workspace. To dock a window that is now in a floating state, drag it toward the control button until you see the highlighted docking pane. Release the mouse button to dock the window to the selected pane. The window changes its width and height depending on the size of the container. You can drag the window out of the docking pane to make it floating or change its position.




Tip: To quickly dock or undock a window, double-click its title bar.

Automatically hide dockable windows

Docked windows are attached, or “pinned” to the edge of the parent window. To automatically hide a dockable window, click the **Auto Hide** pushpin button  on the title bar or select Auto Hide.



When you “unpin” a window, it automatically slides into the main window frame and becomes a tab. The tab attaches to the nearest side of the window. The window name is displayed on the tab. To show the hidden window, point to the tab. When you move the pointer out of the window, it slides back, that is, it automatically hides.

To dock a hidden window to its original docking pane, click  or disable the **Auto Hide** option.

Float windows

To float (undock) a window, select **Floating**. You can drag a floating window out of the main application window.

Hide and show windows

To hide a window, select the **Hide** option or click . To show a window, use the **View** menu.

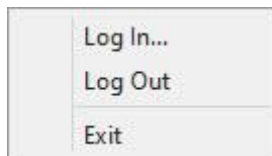
Main menu

The main menu includes all of the Terminal's key features.

Menu	Commands
AlgoTerminal	Login window and terminal closing.
View	Alerts and application windows: Symbols, Bot Instance, Algo Repository, Trade, History and Journal.
Preferences	Managing sounds, Restart Bots on startup command.
Profile	Loading and saving profiles.
Tools	Installing or updating Visual Studio plug-in.
Help	User Manual and app information.

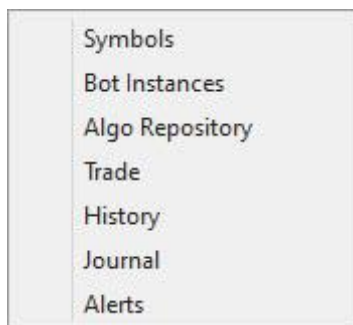
AlgoTerminal menu

The following commands are available on the **AlgoTerminal** menu:



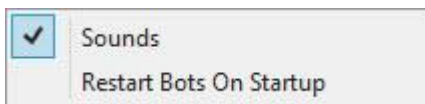
- **Log In:** This command allows you to log in to the application.
- **Log Out:** This command allows you to log out of the application.
- **Exit:** Click **Exit** to quit the application.

View menu



- **Symbols:** Shows/hides the [Symbols](#) window.
- **Bot Instances:** Shows/hides the [Bot Instances](#) window.
- **Algo Repository:** Shows/hides the [Algo Repository](#) window.
- **History:** Shows/hides the [History](#) window.
- **Journal:** Shows/hides the [Journal](#) window.
- **Alerts:** Shows/hides the [Alerts](#) window.

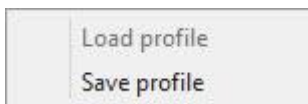
Preferences



- **Sounds:** If the command is selected the sounds will alert you when particular actions take place, or when prompted for input.
- **Restart Bots On Startup:** If this command is selected, then running bots in the Terminal should start after reopening the application.

Profile

You can adapt the Terminal to the way you work by creating your own profiles. Any arrangement of chart windows and with their properties is called a profile. You can manipulate chart windows, change their properties, and then save the layout in a custom profile. If you rearrange chart windows or modify their properties, all changes will be saved to the current profile.

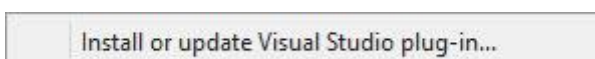


- **Load profile:** To load the file with the application settings, select the file stored locally (depending on the chosen location during the Saving profile procedure).
- **Save profile:** Select this option to save the file with the application settings to your device or to export it.

To save a profile:

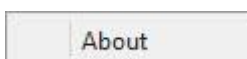
1. On the **Profile** menu, point to **Save profile** and click **Save As**.
2. In the window that opens, enter a profile name and click **Save**.

Tools



Install or update Visual Studio plug-in: Used to write a plugin with the help of which it will be possible then to create a bot or an indicator.

Help



About: Displays information about the program.

In the application, the U.S. format is used for date and time notation and numerical values (for thousands and decimal separators).

Symbols window

The Symbols window displays real-time quotes incoming from the trading server.

Symbols			
Symbol	Bid	Ask	Time
XTIUSD_L			
XTIUSD	62.97	63.03	4/19/2021 15:07:26.389
XRPUSD_L	NaN	0.57128	2/22/2021 15:58:28.175
XRPUSD	1.33370	1.34018	4/22/2021 15:43:51.539
XRPRUB_L			
XRPRUB			
XRPIPY_L			
XRPIPY			
XRPGBP_L			
XRPGBP			
XRPEUR_L	0.47134	NaN	2/22/2021 15:58:02.542
XRPEUR	1.11002	1.11370	4/22/2021 15:43:51.458
XRPCNH_L			

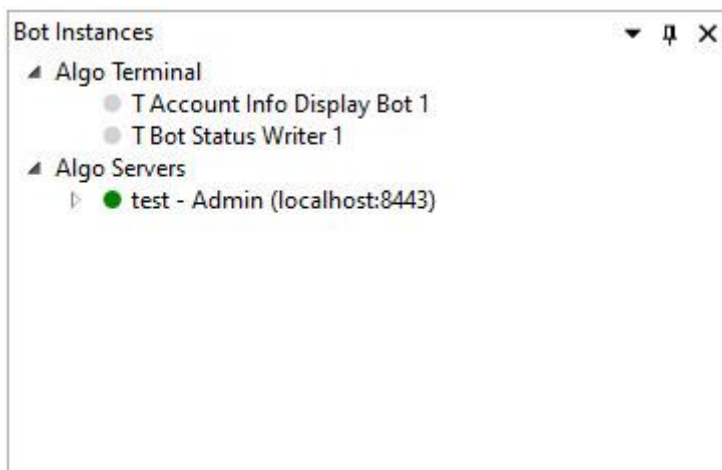
- **Symbol:** Symbol name.
- **Bid:** Bid price.
- **Ask:** Ask price.
- **Time:** Time when the quote was received from the server.

Bot Instances

Algo Terminal and Algo servers if they are added are displayed here.

Bot Instances	
Algo Terminal	
Algo Servers	

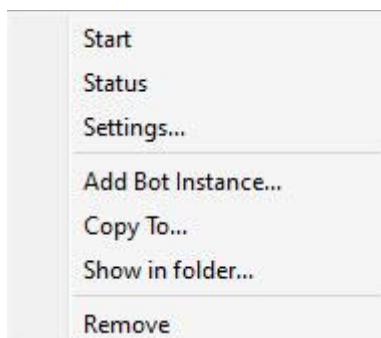
If bots were added to the Terminal they will be shown under the Algo Terminal.



Bot status indicators:

- Light Gray: stopped.
- Gold: starting.
- Green: running.
- Orange: stopping.
- Sky Blue: reconnecting.
- Red: broken/faulted.

Right-click the added bot to perform the following actions:



1. [Start/Stop.](#)
2. [Status.](#)
3. [Settings.](#)
4. [Add Bot Instance.](#)
5. [Copy to.](#)
6. [Show in folder.](#)
7. [Remove.](#)

Start/Stop command

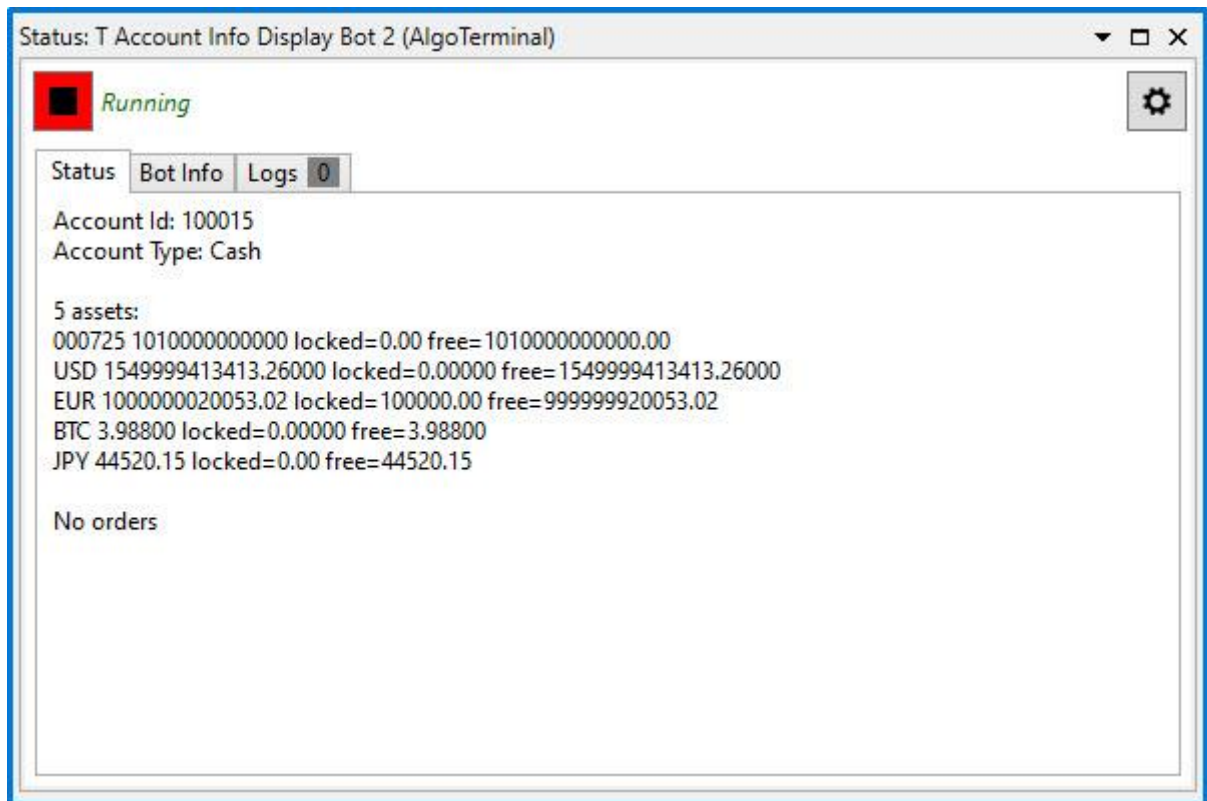


The command allows to start or stop the bot. You can start/stop the bot in the Status window. Select the Status command from the right-click menu or double-click the bot.

Status

After selecting this command, the **Status** window will be displayed.

Here you can see the bot status (Running/Stopped), bot info, logs and configure the bot settings.

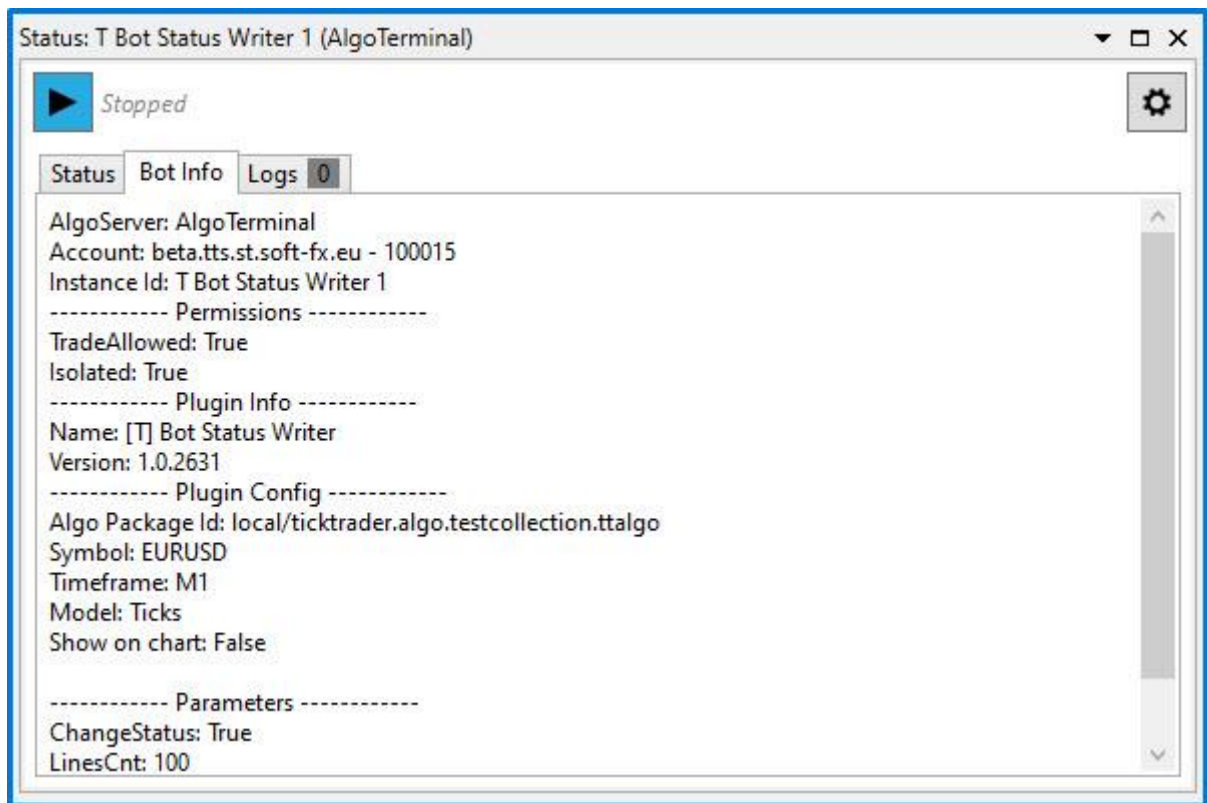


Status tab

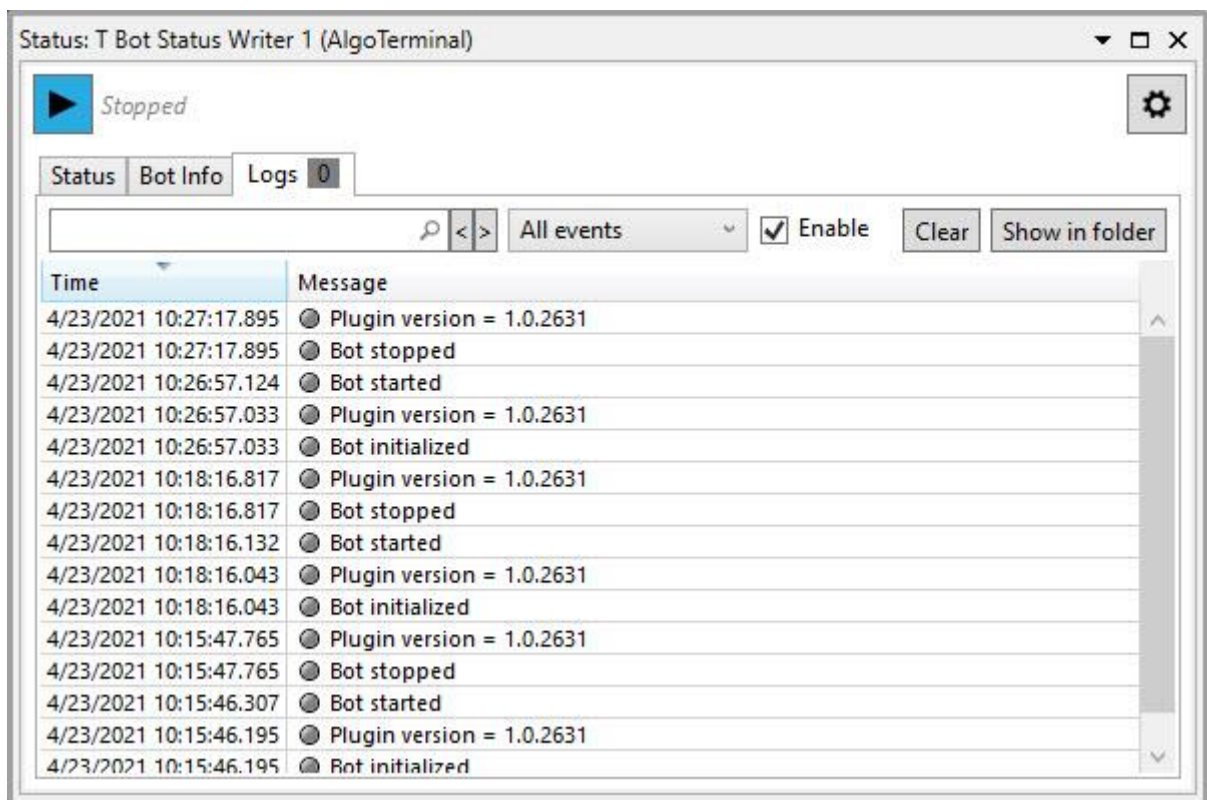
Shows the bot parameters and trade operations. The information on this tab depends on how the bot is written.

Bot info

Here you can view more details related to the bot package and the bot itself.



Logs



On the **Logs** tab, the events related to bots are recorded in real time.

Digits near the tab label indicate the number of errors in a log Logs 176.

The information about events over a particular period can be requested at the top of the tab. Use the **Search** field to find logs and using the filter select an event by which you want the search to be performed.

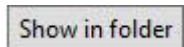
To navigate between the search terms, click the arrow buttons as shown below.


Enable or disable filter. To enable or disable the filter, select or clear the **Enable** check box.



- **Time:** The event timestamp (in dd/mm/yyyy hh:min:sec format) 2/20/2021 21:42:23.382
- **Message:** Event description.

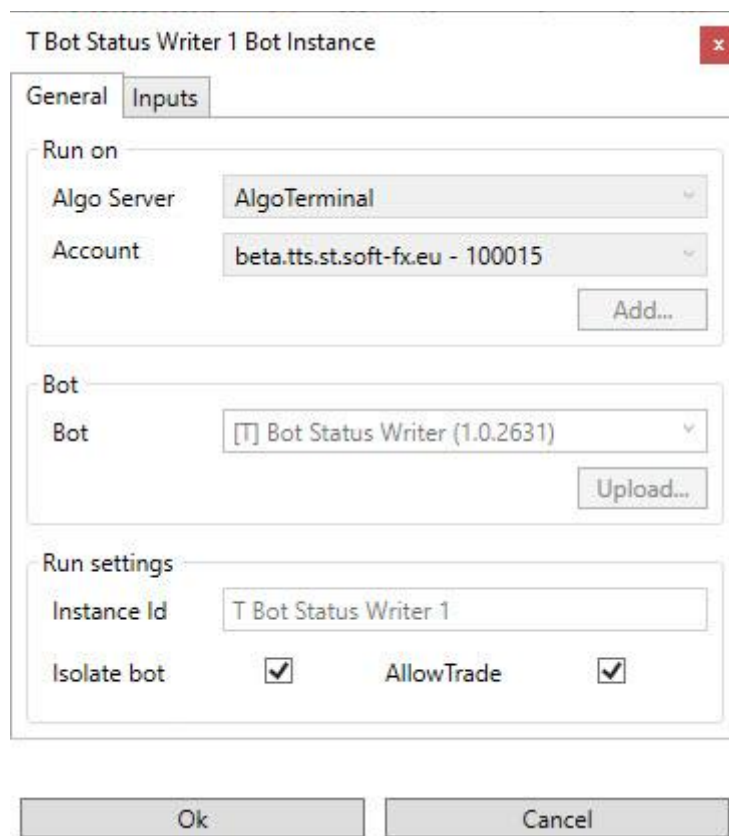
To clear events history, click **Clear**  on the toolbar.

To open the folder containing log files click the **Show in folder**  button. By default logs are stored: Local Disk (C:) > Program Files > TickTrader > AlgoTerminal > Bot logs.

Click **Settings**  to configure the bot settings.

Settings

The **Bot Instance** window is displayed.



General tab

Run on

- **Algo Server:** Contains “Bot Terminal” item and bot services names associated with this Bot Terminal.

- **Account:** Account to be selected for bot instance run. The list is formed based on [Algo Server](#) selection.

Click the **Add** button to add a new account (available only when the Algo Server service is selected).

Bot

- **Bot:** Combo box that contains the bots from the packages loaded to the selected Algo Server.

Click **Upload** to upload an Algo package (available only when the Algo Server connection is selected).

Run settings

- **Instance Id:** Bot Instance Name.
- **Isolate bot:** Set the flag on if you don't want to allow the bot to operate with all the orders available on the account but allow it to see just orders created by this bot.
- **AllowTrade:** Set the flag off if you don't want to allow the real orders opening, but want to check the order open requests.
- **Start immediately:** Set the flag on to start the bot right after creating or modifying.

Inputs tab

Information in this tab varies and depends on the way the bot is written.

Common info:

- **Load settings:** Allows loading previously saved bot settings.
- **Save settings:** Allows to save current bot settings.

Most bots contain the **Main symbol** section:

- **Main symbol:** The base symbol that bot requires to start. This is the default symbol that the bot works with unless otherwise specified in the bot settings. When writing a bot, you can set a flag "Set up Main Symbol = False" in this case default Main symbol is irrelevant. If the bot is drawing something on the chart, then the number of points on the chart will correspond to the number of bars by the Main Symbol.

Select the symbol's side on the right: Ask or Bid.

- **TimeFrame:** The periodicity selected for the execution.
- **Model**
 - From MN to S1: a configurable parameter that indicates the refresh rate, calls when a quote comes from a new bar (for example, if M5, then every 5 minutes).
 - Ticks: a configurable parameter that specifies the refresh rate, it's called on every tick.

Add Bot Instance

Select this command to add a new bot instance. Information **New Bot Instance** window is similar to the [Bot Instance](#) window.

New Bot Instance

General Inputs

Run on

Algo Server: AlgoTerminal

Account: beta.tts.st.soft-fx.eu - 100015

Bot

Bot: [T] Bot Status Writer (1.0.2631)

Run settings

Instance Id: T Bot Status Writer 2

Isolate bot: ☒ AllowTrade: ☒

Start immediately: ☒

Ok Cancel

Copy to

The command allows to copy the selected bot with its package and settings to another account (local or remote).

Copy Bot Instance

From: AlgoTerminal/T Bot Status Writer 1

To: AlgoTerminal - beta.tts.st.soft-fx.eu - 100015 T Bot Status Writer 1

Ok Cancel

- **From:** Displays the current bot location and the bot's name.
- **To:** Select the account where you want to run the bot from the list. Bot Instance ID must be specified in the field on the right.

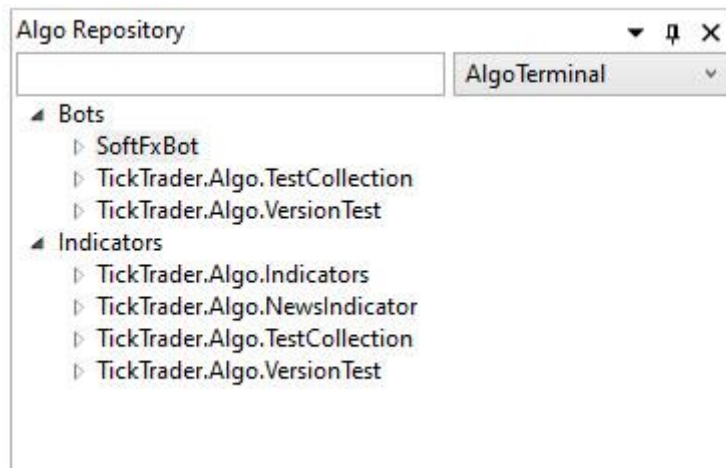
Show in folder

The command opens the folder where the bot logs are stored.

Remove

The command allows removing the bot from the Terminal and Algo Server.

Algo Repository



In this window, you can manage **bots** and **indicators**.

Bots: Algorithmic trading systems that allow complete automation of analytical and trading activities.

Indicators: These programs are used for analyzing prices and identifying patterns in price changes. Indicators can be used directly in trading bots forming a complete automated trading system.

Bots and indicators are grouped in packages.

To find a bot or an indicator, type their name in the **Search** field. You can search for symbols by name or part of the name.

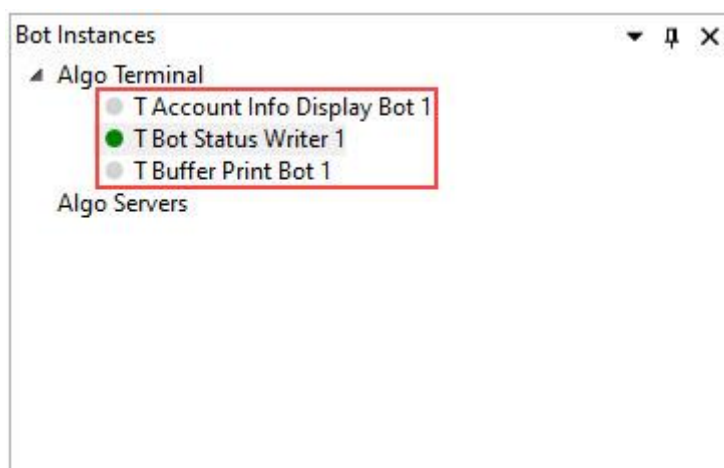
Hover the mouse over the package to view the path where its stored and the time of last modification.

Adding bots to the Terminal

There are several ways to add bots to the Terminal:

1. In the package select a bot and drag it to the chart.
2. In the package select a bot and drag it to the **Bot Instances** window > **Algo Terminal**.
3. Right-click on the Chart area and choose the Add Bot command, select a bot from the list.

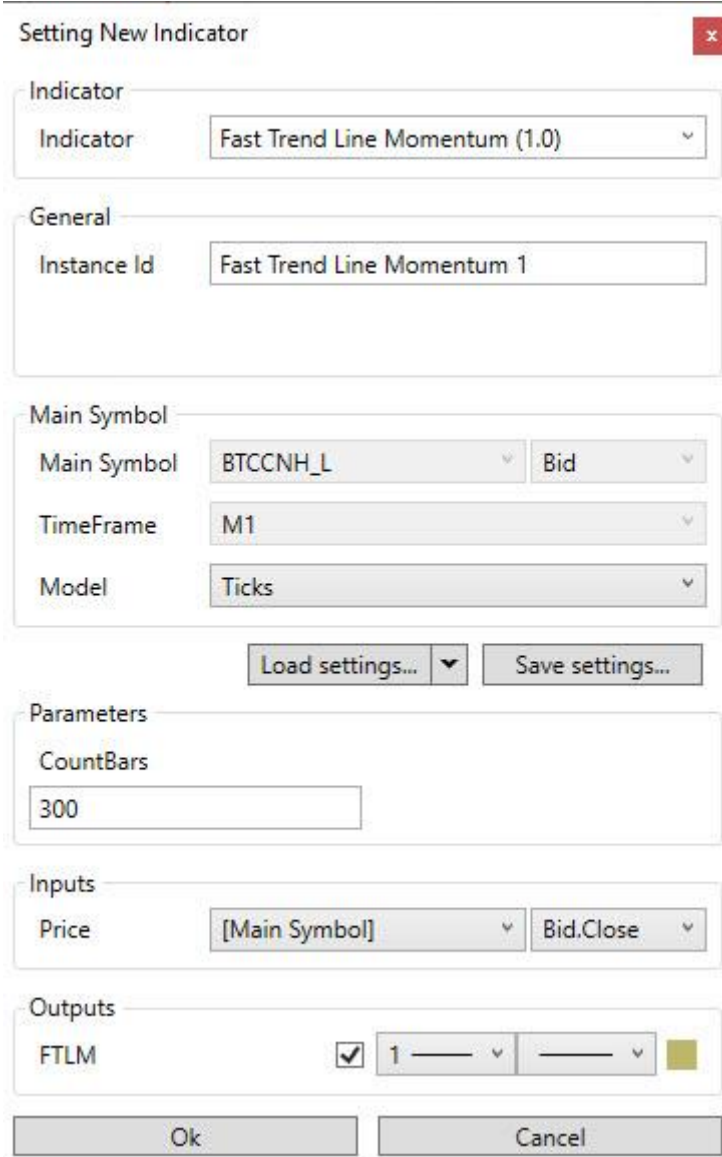
The added bots will be displayed in the **Bot Instances** window under the **Algo Terminal**.



Adding Indicators

The indicators can be added to the **Chart** only.

Select an indicator and drag it to the chart area or right-click on the Chart area and choose the Add Indicator command and then select an indicator from the list. The **Setting New Indicator** window will be displayed:



The 'Setting New Indicator' dialog box is shown with the following sections and controls:

- Indicator**: A dropdown menu showing 'Fast Trend Line Momentum (1.0)'.
- General**: An 'Instance Id' text field containing 'Fast Trend Line Momentum 1'.
- Main Symbol**:
 - 'Main Symbol' dropdown: 'BTCNH_L'.
 - 'Bid' dropdown: 'Bid'.
 - 'TimeFrame' dropdown: 'M1'.
 - 'Model' dropdown: 'Ticks'.
- Buttons: 'Load settings...' and 'Save settings...'.
- Parameters**: A 'CountBars' text field containing '300'.
- Inputs**:
 - 'Price' dropdown: '[Main Symbol]'.
 - 'Bid.Close' dropdown: 'Bid.Close'.
- Outputs**:
 - 'FTLM' checkbox: checked.
 - '1' dropdown: '1'.
 - 'FTLM' dropdown: empty.
 - Color swatch: yellow.
- Buttons: 'Ok' and 'Cancel'.

The sections in this window may vary depending on the indicator's settings.

The Common sections and description for all the indicators:

Indicator

- **Indicator:** The indicator name.

General


- **Instance Id:** The unique indicator name which is used for bot isolation.

Main Symbol

- **Main Symbol:** The base symbol that indicator requires to start. Select the symbol's side on the right: Ask or BidS .
- **TimeFrame:** The periodicity selected for the execution.
- **Model:**
 - From MN to S1: a configurable parameter that indicates the refresh rate, calls when a quote comes from a new bar (for example, if M5, then every 5 minutes).
 - Ticks: a configurable parameter that specifies the refresh rate, it's called on every tick.

Load and **Save:** The buttons that allow loading saved and saving current bot settings.

After configuring the indicator setting and clicking the **Ok** button the indicator will be displayed in the **Chart** window.

To remove the indicator, right-click in the chart area point to **Indicators** and click the cross  near the indicator you want to delete.

Chart

The Chart reflects the changes in the price of a financial instrument over time. It is used for graphical analysis, study of indicators and forecasting of the market price. You can add bots and indicators to the chart, change the chart type and its parameters.

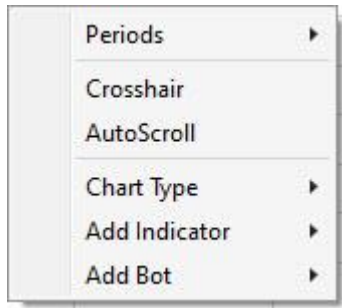


The x-axis shows the date and time values, while the y-axis displays price values of the financial instrument.

Chart data is refreshed automatically.

The current symbol is displayed at the top of the Chart window.

Right-click the chart area to view the context menu. The following actions and commands are available:



1. [Periods](#).
2. [Crosshair](#).
3. [AutoScroll](#).
4. [Chart Type](#).
5. [Add Indicator/Add Bot](#).

Periods

Periodicity is the chart time frame. It affects how often pricing information is refreshed on the chart. Once the trading period is over, the chart updates the latest data based on the selected interval time frame. The Terminal allows you to choose periodicity from one second (S1) to one month (MN1).

Crosshair

A crosshair consists of two perpendicular lines (horizontal and vertical) that move as you move the pointer. It is a handy tool that allows you to focus on the price (bar) of interest and compare price levels in different periods without adding graphic objects to the chart. The crosshair adds labels with exact price and time on the X and Y axes of the chart.



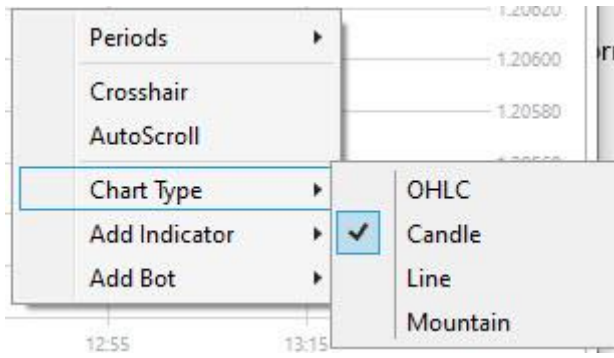
Move the pointer (Cursor or Crosshair) over the chart to see the detailed price information for the selected period. The OHLC data is displayed in the upper-left corner of the chart window and on the tooltip:

- **Time:** The start of the selected period.
- **Open:** The price of the first tick that came in the selected period.
- **High:** The highest price of the selected period.
- **Low:** The lowest price of the selected period.
- **Close:** The price of the last tick that came in the selected period.

Autoscroll

The **Auto Scroll** option enables an automatic scroll of the chart in the window area, allowing you to see the latest quotes all the time.

Chart Type



The following chart types are available:

- **OHLC:** A type of bar chart that shows open, high, low, and closing prices for each period.
- **Candle:** The chart is a sequence of Japanese candlesticks.
- **Line:** The chart is a line from one closing price to the next closing price.
- **Mountain:** A mountain chart is the same as a line chart, except the area under the line is shaded. The result is a chart that looks like a mountain, hence the name.

Add Indicator/Add Bot

This command allows to select and add indicators and bots to the Chart.

Trade

The **Trade** window contains trade positions and pending orders that have not yet been activated.

[illegible]

Screenshot for Gross account

Gross Account

Gross position parameters:

- **ID:** ID of the trade operation.
- **Parent Order Id:** ID of the parent trade operation.
- **Time:** Position open time.

- **Modified:** Position last update time.
- **Symbol:** Financial instrument.
- **Side:** Sell or Buy order.
- **Volume:** Order remaining volume.
- **Req Open Price:** Requested open price.
- **Open Price:** Position price.
- **Current Price:** Last Quote. For the *Buy* side – Bid, for *Sell* – Ask.
- **Price Deviation:** For the *Buy* side is calculated as: $Price\ Deviation = Bid - Open\ Price$.
For the *Sell* side is calculated as: $Price\ Deviation = Open\ Price - Ask$.
- **T/P:** Position's TP price.
- **S/L:** Position's SL price.
- **Swap:** Position's swap value.
- **Commission:** Position's commission value.
- **Net P/L:** Position's profit or loss including commission and swap values for the symbol.
- **Tag:** Order Tag.
- **Comment:** Order comment.

Hover the mouse over the data in the grid to see the following parameters:

- **Instance ID:** Bot instance ID that created the order.
- **Initial Volume:** Original order volume. The initial amount of volume before open execution.
- **Initial Type:** Initial order type.
- **Max Visible Open Volume:** Max visible volume, which appears in the Market Depth (If it was defined).
- **Order Open Execution Option:** IoC/Hidden/Iceberg.
- **Slippage:** The slippage value set by the user at the opening order step.

Gross pending orders parameters:

- **ID:** ID of the trade operation.
- **Symbol:** Financial instrument.
- **Time:** Order placement time.
- **Modified Time:** Time when the order was changed/modified.
- **Type:** Type of operation (*Buy/Sell Limit, Buy/Sell Stop, Buy/Sell Stop Limit*)
- **Volume:** Order volume remaining/Original order volume.
- **Open Price:** Order activation price.
- **Current Price:** Last Quote. For the *Buy* side Ask. For *Sell* – Bid.
- **Price Deviation:** For the *Buy* side is calculated as $Price\ Deviation = Ask - Open\ Price$.
For the *Sell* side is calculated as $Price\ Deviation = Open\ Price - Bid$.

- **T/P:** Take Profit level.
- **S/L:** Stop Loss level.
- **Tag:** Order Tag.
- **Comment:** Order comment.

Hover the mouse over the data in the grid to see the following parameters:

- **Instance ID:** Bot instance ID that created the order.
- **Limit/Stop Price:** The *Stop Price* and *Limit Price* fields for *StopLimits*. The *Limit Price* field for *Buy/Sell Limit* orders, the *Stop Price* for *Stop* orders.
- **Max Visible Volume:** Max visible volume, which appears in the Market Depth.
- **Expiration:** Order expiration date (for *GoodTillDate* orders).
- **Slippage:** The slippage value set by the user at the opening order step.

Current Balance

- ❓ **Balance:** The total financial result of all completed transactions and *Deposit/Withdrawal* operations on the trading account.
- ❓ **Equity:** Result of the following calculation: *Balance + Floating*.
- ❓ **Margin Used:** Margin required to cover an open position.
- ❓ **Free Margin:** The amount of available Margin.
- ❓ **Margin Level:** The percentage of Equity to Margin. Calculated as $(Equity/Margin) * 100\%$.
- ❓ **Swap:** Account Swap value is calculated as a summary of rounded* Swap values for each symbol in real time.
- ❓ **Floating:** Current profit/loss on open positions calculated at the current quotes.

Net Account

Net position parameters:

- **ID:** ID of the trade operation.
- **Symbol:** Financial instrument.
- **Modified:** Last modified date.
- **Side:** *Sell* or *Buy*.
- **Volume:** Order volume.
- **Open Price:** Position price.
- **Current Price:** Last Quote. For the *Buy* side – Bid, for *Sell* – Ask.
- **Price Deviation:** For the *Buy* side is calculated as $Price\ Deviation = Bid - Open\ Price$.
For the *Sell* side is calculated as $Price\ Deviation = Open\ Price - Ask$.
- **Swap:** Position's swap value.
- **Net P/L:** Position's profit or loss including commission and swap values for the symbol.

Net pending orders parameters:

- **ID:** ID of the trade operation.
- **Symbol:** Financial instrument.
- **Time:** Order placement time.
- **Modified Time:** Time when the order was changed/modified.
- **Type:** Type of operation (Buy/Sell Limit, Buy/Sell Stop, Buy/Sell StopLimit).
- **Volume:** Order volume remaining/Original order volume.
- **Open Price:** Order activation price.
- **Current Price:** Last Quote. For the *Buy* side Ask. For Sell – Bid.
- **Price Deviation:** For the *Buy* side is calculated as $Price\ Deviation = Ask - Open\ Price$.
For the *Sell* side is calculated as $Price\ Deviation = Open\ Price - Bid$.
- **Tag:** Order Tag.
- **Comment:** Order comment.

Hover the mouse over the data in the grid to see the following parameters:

- **Instance ID:** Bot instance ID that created the order.
- **Limit/Stop Price:** The *Stop Price* and *Limit Price* fields for *StopLimits*. The *Limit Price* field for *Buy/Sell Limit* orders. The *Stop Price* field for *Stop* orders.
- **Max Visible Volume:** Max visible volume, which appears in the Market Depth.
- **Expiration:** Order expiration date (for *GoodTillDate* orders).
- **Slippage:** The slippage value set by the user at the opening order step.

Current Balance

- ❓ **Balance:** The total financial result of all completed transactions and deposit/withdrawal operations on the trading account.
- ❓ **Equity:** Equity is calculated as Balance plus Floating.
- ❓ **Margin Used:** Margin required to cover an open position.
- ❓ **Free Margin:** The amount of available Margin.
- ❓ **Margin Level:** The percentage of *Equity* to *Margin*. Calculated as $(Equity/Margin) * 100\%$.
- ❓ **Swap:** Account Swap value is calculated as a summary of rounded* Swap values for each symbol in real time.
- ❓ **Floating:** Current profit/loss on open positions calculated at the current quotes.

Cash Account

The **Trade** tab contains information about the current state of assets on the trading account. Here you can monitor the status of your assets in real time.

- **Currency:** Asset (currency) name.

- **Volume:** Total currency volume (the sum of free and locked volume for the indicated currency).
- **Locked Volume:** The volume of margin currency required to fill the order.
- **Available Volume:** The volume of available currency.

Cash pending orders parameters:

- **ID:** ID of the trade operation.
- **Symbol:** Financial instrument.
- **Modified Time:** Time when the order was changed/modified.
- **Type:** Type of operation (Buy/Sell Limit, Buy/Sell Stop, Buy/Sell StopLimit).
- **Volume:** Order volume remaining/Original order volume.
- **Open Price:** Order activation price.
- **Price Deviation:** For the *Buy* side is calculated as $Price\ Deviation = Ask - Open\ Price$.
- For the *Sell* side is calculated as $Price\ Deviation = Open\ Price - Bid$.
- **Time:** Order placement time.
- **Current Price:** Last Quote. Last Quote. For the *Buy* side Ask. For Sell – Bid.
- **Tag:** Order Tag.
- **Comment:** Order comment.

History

The **History** window contains the history of operations on the trading account.

Gross Account

Parameters:

- **ID:** Consists of two parts and divided by #: Order ID # Fill ID (Fill ID – order's sequence number in case of partial execution).
- **Parent Order ID:** ID of the parent trade operation.
- **Open Time:** Order open execution (fill) time.
- **Type:** The column shows values depending on the order Type and Reason. See more information in the Type/Reason column table.
- **Symbol:** Financial instrument. Note: The column remains empty for balance operations.
- **Req Open Volume:** Open order volume requested before the execution.
- **Open Volume:** Order volume that was closed on this execution step. The amount for Deposit, Withdrawal or Dividends + currency sign.
- **Req Open Price:** Open order price requested before the execution.
- **Open Price:** Open order price that refers to the selected trade record.

- **Open Slippage:** For the Buy orders is calculated as: $\text{Slippage} = \text{Price} - \text{ReqPrice}$. For the Sell: $\text{Slippage} = \text{ReqPrice} - \text{Price}$.
- **Close Time:** Position close execution time.
- **Req Close Volume:** Close order volume requested before the execution.
- **Close Volume:** Close order volume that refers to the selected trade record.
- **TP:** Position's TP price.
- **SL:** Position's SL price.
- **Req Close Price:** Close order volume requested before the execution.
- **Close Price:** Close order price that refers to selected trade record.
- **Close Slippage:** For the Buy orders is calculated as: $\text{Slippage} = \text{Price} - \text{ReqPrice}$. For the Sell orders: $\text{Slippage} = \text{ReqPrice} - \text{Price}$.
- **Remaining Volume:** Remaining position volume that stays open after the close.
- **Commission:** Order commission value.
- **Swap:** Order swap value.
- **Net P/L:** Position's profit or loss including commission and swap values for the symbol.
- **Taxes:** Dividend's taxes.
- **Fees:** Dividend's fees.
- **Tag:** User tag.
- **Comment:** User comment.

Hover the mouse over the data in the grid to see the following parameters:

- **Instance ID:** Bot instance ID that created the order.
- **Reason:** See information in the [Type/Reason](#) column table.
- **Initial Volume:** Original (pending order) volume. The initial amount of volume before execution.
- **Initial Type:** Initial (pending) order type.
- **Max Visible Open Volume:** Max visible volume, which appears in the Market Depth (If it was defined).
- **Order Open Execution:** IoC/HiddenIceberg.
- **Req Slippage:** The slippage value set by the user at the opening order step.

Type/Reason column

- **Order Type:** Order Type field in the Trade report.
- **Reason:** Trade Transaction Reason in the Trade Report.
- **Type:** The final value in Trade History -> Type column.

TradeTransReportType	TradeTransReason	Type Column Value in History	Reason (in to
Order Filled	Client request	Buy or Sell	
Order Filled	DealerDecision	Buy or Sell	DealerDecisi
Order Filled	StopOut	Buy or Sell	StopOut
Order Filled	Pending order activation + OrdType: Stop	Buy or Sell	
Order Filled	Pending order activation + OrdType: Limit	Buy or Sell	
Order Activated	DealerDecision + OrdType: StopLimit	Buy or Sell StopLimit	Activated
Position Closed	StopOut	Buy or Sell	StopOut
Order Canceled	Client request	Buy/Sell Limit Canceled Buy/Sell Stop Canceled Buy/Sell StopLimit Canceled	
Order Canceled	DealerDecision	Buy/Sell for position Buy/Sell Limit Canceled Buy/Sell Stop Canceled Buy/Sell StopLimit Canceled	Canceled By

Order Canceled	StopOut	Buy/Sell Limit Canceled Buy/Sell Stop Canceled Buy/Sell StopLimit Canceled	StopOut
Order Expired	Expired	Buy/Sell Limit Canceled Buy/Sell Stop Canceled Buy/Sell StopLimit Canceled	Expired
Balance Transaction	Dividend	Dividend	
Balance Transaction	Transfer Money	Transfer Funds	
Balance Transaction	Split	Split	
Balance Transaction		Deposit/Withdrawal	
Trade Modified	Split	Split Buy/Split Sell	

Net Account

Parameters

- **ID:** Consists of two parts divided by #: Order ID # Fill ID (Fill ID – order's sequence number in case of partial execution).
- **Time:** Order execution (fill) time.
- **Type:** This column should show values depending on the order Type and Reason. See the specification in the [Type/Reason](#) column table.
- **Symbol:** Financial instrument. Currency for Deposit and Withdrawal.
- **Volume:** Order volume executed on this execution step. Amount for Deposit or Withdrawal.

- **Price:** Execution price that refers to selected trade record.
- **Slippage:** For Buy orders is calculated as $Slippage = Price - ReqPrice$, for Sell orders: $Slippage = ReqPrice - Price$.
- **Req Volume:** Order requested volume before the execution.
- **Pos Close Price:** Price that position had before this trade. This price is used for calculating P/L.
- **Pos Close Volume:** Position Volume that was closed during this execution. If $Order Side = Position Side$ then $Pos Close Volume = 0$.
- **Commission:** Order commission value.
- **Swap:** Order swap value.
- **Net P/L:** Position's profit or loss including commission and swap values for the symbol.
- **Taxes:** Dividend's taxes.
- **Fees:** Dividend's fees.
- **Tag:** User tag.
- **Comment:** User comment.

Hover the mouse over the data in the grid to see the following parameters:

- **Reason:** See the specification in the Type/Reason column table.
- **Initial Volume:** Original order volume. Initial amount of volume before execution.
- **Initial Type:** Initial type of order.
- **Max Visible Volume:** Max visible volume, which appears in the Market Depth (If it was defined).
- **Order Execution Option:** IoC/HiddenIceberg.
- **Pos Price:** Final position price that was applied after this execution step.
- **Pos Volume:** Final position volume that was applied after this execution step.
- **Req Slippage:** The slippage value set by the user at the opening order step.

Type/Reason column

Order Type: Order Type field in Trade report.

Reason: Trade Transaction Reason in Trade Report.

Type: Final value in Trade History ->Type column.

TradeTransReportType	TradeTransReason	Type Column Value in History	Reason (in tooltip)
Order Filled	Client request	Buy or Sell	

Order Filled	DealerDecision	Buy or Sell	DealerDecision
Order Filled	StopOut	Buy or Sell	StopOut
Order Filled	Pending order activation + OrdType: Stop	Buy or Sell	
Order Filled	Pending order activation+ OrdType: Limit	Buy or Sell	
Order Activated	DealerDecision + OrdType: StopLimit	Buy or Sell StopLimit	Activated
Order Canceled	Client request	Buy/Sell Limit Canceled Buy/Sell Stop Canceled Buy/Sell StopLimit Canceled	
Order Canceled	DealerDecision	Buy/Sell Limit Canceled Buy/Sell Stop Canceled Buy/Sell StopLimit Canceled	Canceled By Dealer
Order Canceled	StopOut	Buy/Sell Limit Canceled Buy/Sell Stop Canceled Buy/Sell StopLimit Canceled	StopOut
Order Expired	Expired	Buy/Sell Limit Canceled Buy/Sell Stop Canceled	Expired

		Buy/Sell StopLimit Canceled	
Deposit/Withdrawal		Deposit/Withdrawal	
Balance Transaction	Dividend	Dividend	
Balance Transaction	Transfer Money	Transfer Funds	
Balance Transaction	Split	Split	
Trade Modified	Split	Split Buy/Split Sell	

Cash Account

Parameters:

- **Order:** Unique transaction identification number.
- **Open Time:** Time when the order (position) was opened.
- **Type:** Operation type.
- **Action:** Action number.
- **Symbol:** Trading instrument or currency.
- **Requested Quantity:** Order requested quantity.
- **Requested Rate:** Requested order execution rate.
- **Exec Time:** Order execution (fill) time.
- **Exec Quantity:** Execution Amount.
- **Exec Rate:** Execution Price.
- **Remaining Quantity:** The remaining volume after the pending order execution operation.
- **Commission:** Order commission value.
- **Asset I:** Change in the balance of asset I resulting from the transaction.
- **Asset II:** Change in the balance of asset II resulting from the transaction.
- **Comment:** Comment to the operation.

- **Max Visible Volume:** Maximum order volume visible in the Market Depth.

Journal

On the **Journal** tab, the application events are recorded in real time (UTC): connection status to the server, check for updates, events occurring during the operation of the application, actions related to trade operations.

Journal	
<input type="text"/> <input type="button" value="🔍"/> <input type="button" value="◀"/> <input type="button" value="▶"/> <input checked="" type="checkbox"/> Enable	
Time	Message
2/20/2021 21:42:23.382	12: login on Soft-Fx Staging Beta
2/20/2021 21:42:18.847	Microsoft Windows 10 Enterprise (x64-based PC), Intel(R) Core(TM) i5-4430 CPU @ 3.00GHz, RAM: 6941 /
2/20/2021 21:42:18.764	BotTrader started

- **Time:** The event timestamp (in dd/mm/yyyy hh:min:sec format) 2/20/2021 21:42:23.382
- **Message:** Event description. Icons in this column allow to select different types of records easily:
 - **Gray:** Info messages.
 - **Blue:** Trade messages.
 - **Green:** TradeSuccess messages.
 - **DarkOrange:** TradeFail messages.
 - **Red:** Error messages.
 - **Violet:** Custom messages.
 - **Khaki:** Alert messages.

To clear events history, click **Clear** on the toolbar. So, you can see the new records coming.

Click **Show in folder** to open the folder containing the files of the events.

Use the **Search** field to find logs in Logs history. To navigate between the search terms, click the arrow buttons as shown below.

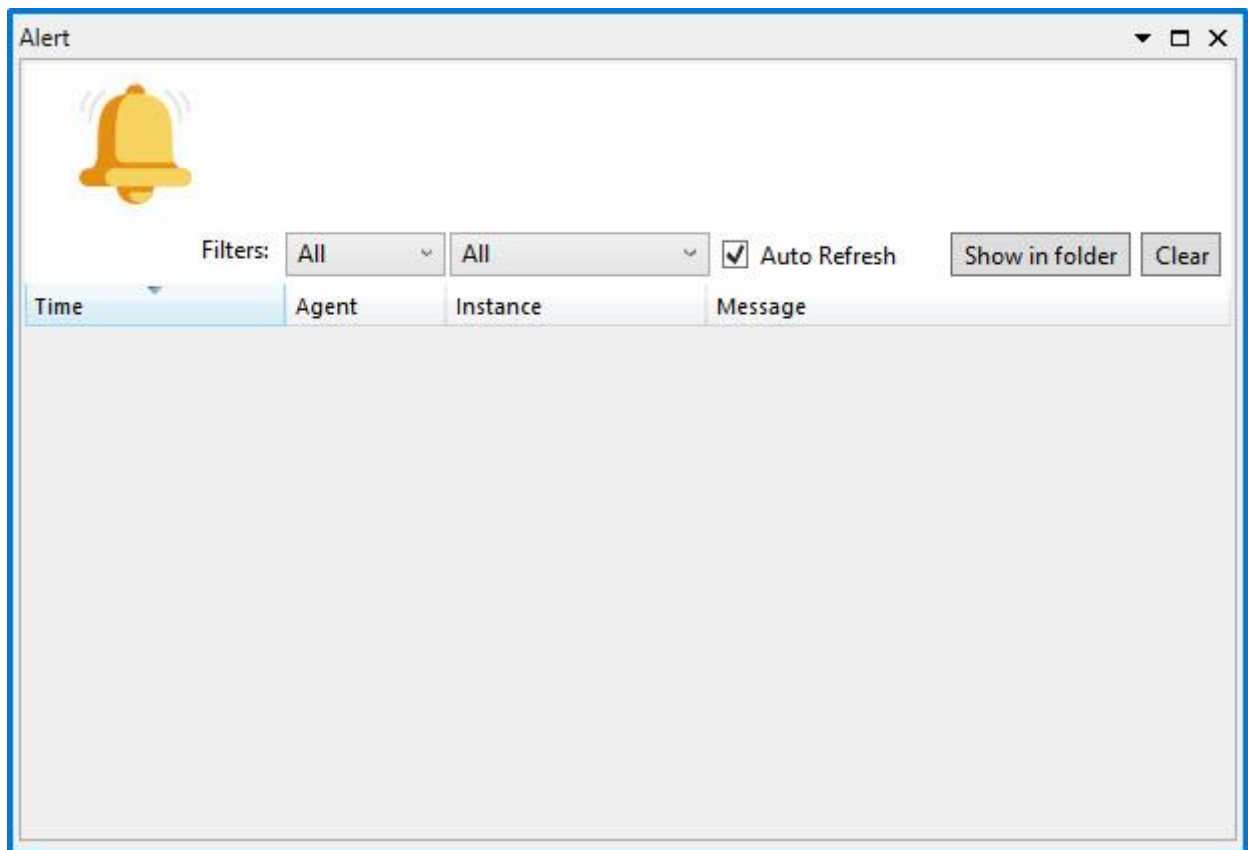
Enable or disable filter. To enable or disable the filter, select or clear the **Enable** check box.

<input type="text"/> <input type="button" value="🔍"/> <input type="button" value="◀"/> <input type="button" value="▶"/>	<input checked="" type="checkbox"/> Enable
---	--

Alerts

Alerts are the highest priority messages.

The **Alert** window contains alerts created by the user, bots or indicators.

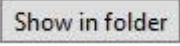



All created alerts are represented as a table with the following fields:

- **Time:** Time when the event occurred.
- **Agent:** The application where the alert was created: Terminal/Algo Server.
- **Instance:** The Instance Id of a bot or indicator.
- **Message:** Description of the event.

You can filter alerts by the **application** (Tick trader Algo Terminal/Tick trader Algo Server) or by **Instance Id**.

Information in the table is refreshed automatically. If you don't want it to be refreshed clear the **Auto Refresh** check box.

Click the **Show in folder**  button to go to the folder where the alert logs are stored.

To remove the information in the table, click the **Clear**  button.

Algo Server

Before starting to use the Algo Server service check that the application is [installed](#) on your device.






In case you selected TickTrader Algo Server service as a component to install then after the finishing of the application installation process the [AlgoServer Configurator](#) window appears.

AlgoServer Configurator window

In the AlgoServer Configurator window you can configure the Algo server settings.

Toolbar commands

The following commands are available on the toolbar:

- **Reset**  : Click to undo changes.
- **Save**  : Click if you want to save the changes.
- **Restart**  : Click to restart the server if the parameters in the Terminal connection, WEB Admin and Advanced settings tabs were modified.
- **Run**  : Click to run the server.
- **Stop**  : Click to stop the server.

AlgoServer Configurator: 1.18.2631.0 (2021.04.01)

Reset Save Restart Run Stop

User credentials

Terminal connection

WEB Admin

Advanced settings

Admin

User that has full rights in browser and AlgoServer

Login: Admin Generate

Password: Admin Generate

Dealer

User that has same rights as Admin. Available only in AlgoServer

Login: Dealer Generate

Password: Dealer Generate

Viewer

User that has read-only access to packages, accounts, AlgoData and download. Available only in AlgoServer

Login: Viewer Generate

Password: Viewer Generate

Logs

2021-04-01 19:12:01.6502| Info|ThreadPoolManager|Global queue ping back took 0 ms. Status=RanToCompletion

2021-04-01 19:11:00.6172| Info|ThreadPoolManager|Global queue ping back took 0 ms. Status=RanToCompletion

2021-04-01 19:09:59.5823| Info|ThreadPoolManager|Global queue ping back took 1.0003 ms. Status=RanToCompletion

2021-04-01 19:08:58.5409| Info|ThreadPoolManager|Global queue ping back took 0 ms. Status=RanToCompletion

2021-04-01 19:07:57.5064| Info|ThreadPoolManager|Global queue ping back took 0 ms. Status=RanToCompletion

2021-04-01 19:06:56.4733| Info|ThreadPoolManager|Global queue ping back took 0 ms. Status=RanToCompletion

User credentials tab

There are 3 roles in the Service:

- **Viewer:** Has readonly access to AlgoServer state. Can download logs and see packages, accounts, logs, bots and their status. Can't see AlgoData and downloaded packages. Available only in AlgoTerminal.
- **Dealer:** User that has the same access as Viewer, but can start and stop bots. Available only in AlgoTerminal.
- **Admin:** User that has full rights to modify AlgoServer state. Available both in browser and AlgoTerminal. Select the role and the set the Login and Password.

Note: At the moment in the Algo server application there can be only one login for a user.

Terminal connection tab

Here you can set Terminal connection settings:

- **Listening Port:** Port that is used for AlgoTerminal connection. It should be different from web server ports.
- **Directory Name:** Directory where AlgoTerminal connection logs will be stored.
- **Log Message:** Enables writing messages sent over AlgoTerminal connection to logs. Only for troubleshooting. Has a severe impact on performance.

WEB Admin tab

Here you configure the settings for browser connection.

- **Secret key:** is used for signing auth tokens. Its changing makes all existing tokens invalid, all users will have to relogin. It is not recommended to change the secret key unless it becomes compromised.
- **Listening Urls:** Web server will listen to certain ports and protocols. Server has auto https redirect so http(ex.80) and https(ex.443) port setting are recommended for easier browser access.

Advanced settings tab


Some extra settings are available on this tab:

- **Folder:** Selected AlgoServer installation folder.
- **Enable server connection logs:** Enables writing messages sent over trade server connection to logs. Only for troubleshooting, has severe impact on performance.
- **Save AlgoServer settings:** Allows you to transfer bot and account settings between serves. Click **Save** and select the folder to save the current settings. To load the settings, click **Load** and select a previously saved ZIP file from the directory.

Logs

Logs	
2021-04-01 23:30:18.8541	Info ThreadPoolManager Global queue ping back took 0 ms. Status=RanToCompletion
2021-04-01 23:29:17.8180	Info ThreadPoolManager Global queue ping back took 0 ms. Status=RanToCompletion
2021-04-01 23:28:16.7849	Info ThreadPoolManager Global queue ping back took 0 ms. Status=RanToCompletion
2021-04-01 23:27:15.7466	Info ThreadPoolManager Global queue ping back took 0 ms. Status=RanToCompletion
2021-04-01 23:26:14.7136	Info ThreadPoolManager Global queue ping back took 0 ms. Status=RanToCompletion
2021-04-01 23:25:13.6757	Info ThreadPoolManager Global queue ping back took 0 ms. Status=RanToCompletion

In the **Logs** section you can see information about the application start and all events during its running are reflected in the section. Only the latest messages are shown in the section. To view the directory

that contains all the log files, click the **Show in folder**  button.

Status bar

Server has been started! ● _TTAlgoServer is Running

The following information is displayed in the status bar:

- The status of program commands execution.
- The indicator of server connection:

● Green: running.

● Yellow: starting.

● Green: stopped.

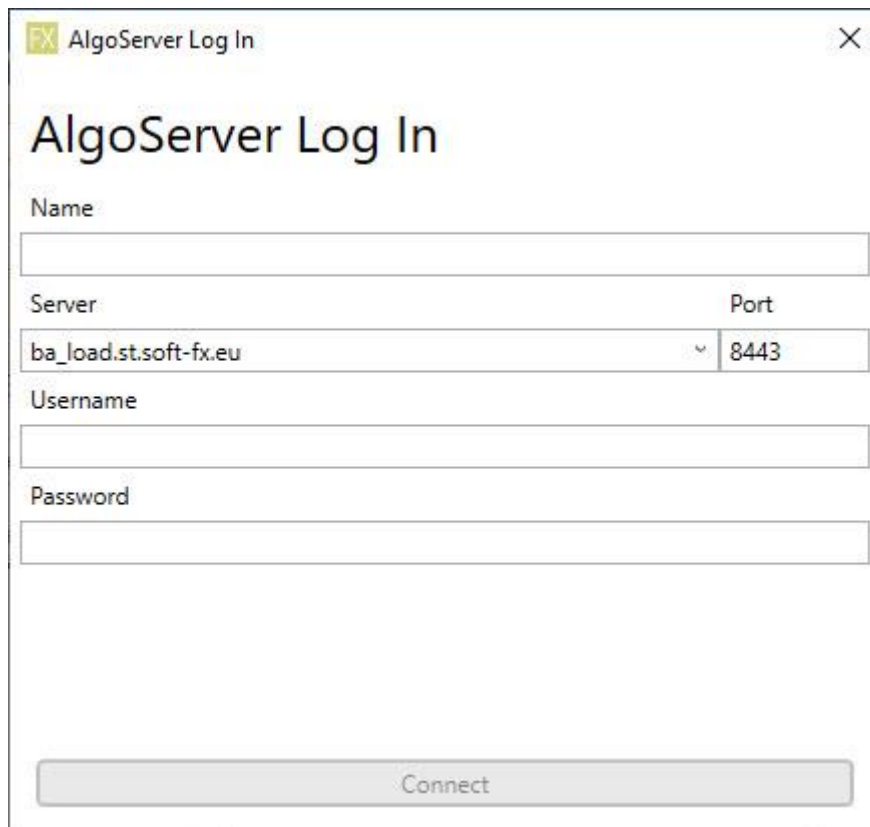
Registration of TickTrader AlgoServer in Algo Terminal.

1. Open the TT Algo Terminal.
2. In the Bot Instances section right-click **Algo Servers** and select the **Register AlgoServer** command.

The **AlgoServer Log In** window appears.

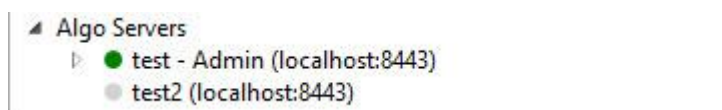
3. Complete the fields in the **AlgoServer Log In** window:
 - **Name:** Display name.
 - **Server:** Address of the server where the AlgoServer service machine is installed.
 - **Port, Username, Password:** fill in the parameters as they were set in the [AlgoServer Configurator](#) window.

Click **Connect**.



The image shows a 'AlgoServer Log In' dialog box. It has a title bar with a close button. The main area contains the following fields: a 'Name' text box, a 'Server' dropdown menu with 'ba_load.st.soft-fx.eu' selected, a 'Port' dropdown menu with '8443' selected, a 'Username' text box, and a 'Password' text box. At the bottom is a 'Connect' button.

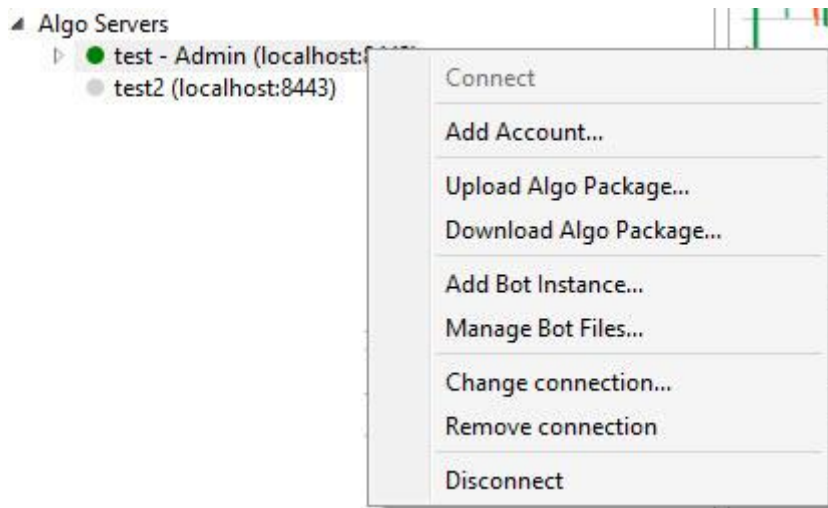
Algo Server Service status indicators:



- Light Gray: offline.
- Gold: connecting.
- Green: online.
- Orange: disconnecting.
- Sky Blue: waiting for reconnecting.

Context Menu

Right-click the previously added Algo Server to view its context menu. The following options and commands are available:



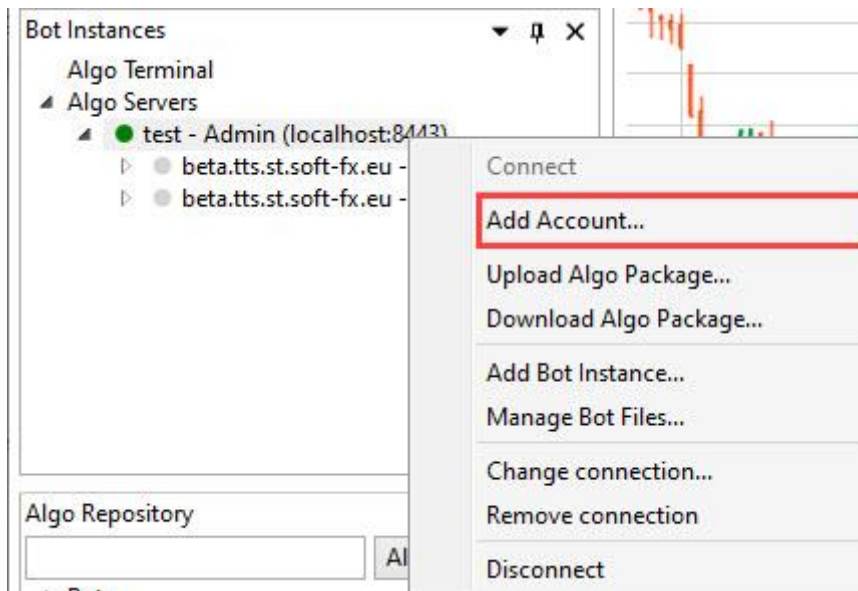
1. **Connect:** Select the command to connect to the Algo server.
2. [Add Account](#) to the Algo Server.
3. [Upload Algo Package](#).
4. [Download Algo Package](#).
5. **Add Bot Instance:** The command allows you to add a new **Bot Instance** to the server. After selecting this command, the **New Bot Instance** window appears. It is similar to the [Bot Instance](#) window.
6. **Manage Bot Files:** This command allows you to manage logs and configs on the remote server.
7. **Change connection:** The command allows editing an existing connection.
8. **Remove connection:** Use this command to delete the server.
9. **Disconnect:** The command closes the current connection to the Algo server.

Add Account

Select this command to add an account to the Algo Server.

How to add an account to the Algo Server

1. In the **Bot Instances > Algo Servers** right-click an available **Algo server** and select the **Add Account** command.



2. Complete the fields in the **Add account** window:

- **Algo Server:** Select an Algo server to which a bot should be added.
- **Server:** Server, to which you are going to connect.
- **Account:** Account number;
- **Password:** Account password.

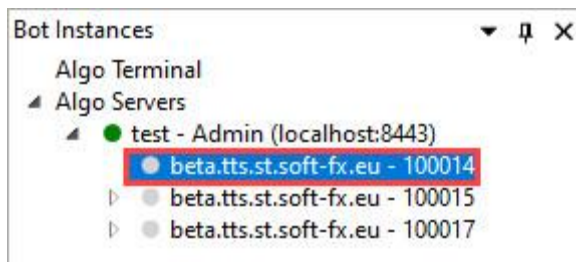
The 'Add account' dialog box is shown with the following fields and values:

- Algo Server:** test (dropdown menu)
- Server:** beta.tts.st.soft-fx.eu (dropdown menu)
- Account:** 100014 (dropdown menu)
- Password:** [masked with dots]

At the bottom, there is a 'Test' button (highlighted with a yellow background), an 'Ok' button, and a 'Cancel' button.

Click **Test** to check the credentials.

Click **Ok**. After that the account will be available in the **Bot Instances** section.

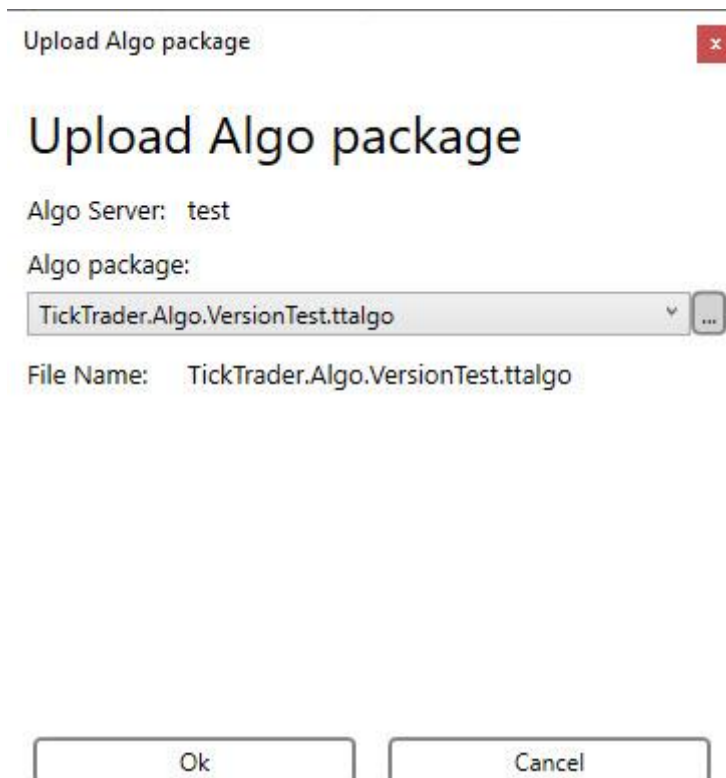


Account status indicators:

- Red: offline.
- Gold: connecting.
- Green: online.
- Orange: disconnecting.

Upload Algo Package

Use this command to upload algo packages.



- **Algo Server:** The name of the Algo Server service.
- **Algo Package:** You can upload the packages from the folder where the Terminal is installed or from the Documents > Algo Terminal > Algo Repository.
- **File Name:** The name of the file to be uploaded. If the package with that name already exists the *Copy*, *Copy1*, *Copy2*, etc. will be added to the file name.

Algo package:

softfxbot.ttalgo

File Name: softfxbot - Copy.ttalgo

Download Algo Package

Use this command to download the packages from the remote Algo server to your PC.

Download Algo package

Algo Server: test

Algo package:

softfxbot - copy.ttalgo

File Name: softfxbot - copy.ttalgo

Ok Cancel

- **Algo Server:** The name of the Algo Server service where the packages are stored.
- **Algo Package:** The list of the packages which exist on the server.
- **File Name:** The name of the file to be downloaded.

Click the **More** button to select the location where the package will be downloaded.

Launch a bot

To launch a bot, you need to [add an account](#) and [add a package with a bot](#), which should be launched.

How to add a Bot package

1. In the **Algo Repository** section select the **Algo Server service**.

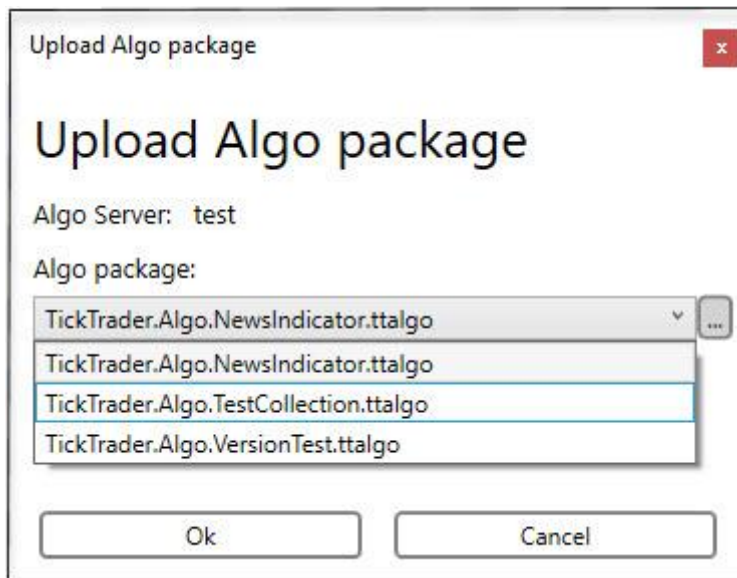
Algo Repository

AlgoTerminal

AlgoTerminal

test

2. In the **Bot Instance** section right-click the service and select the **Upload Algo Package** command and then select the package from the list.



Bot and **Indicators** will be displayed in the **Algo Repository** section.



3. In the Bot Instance section select a service or an account and right-click **Add Bot Instance**. The [New Bot Instance](#) window will be displayed.

New Bot Instance

General Inputs

Run on

Algo Server test

Account beta.tts.st.soft-fx.eu - 100014

Add...

Bot

Bot [T] Account History Bot (1.0.2631)

Upload...

Run settings

Instance Id T Account History Bot 1

Isolate bot ☒ AllowTrade ☒

Start immediately ☒

Ok Cancel

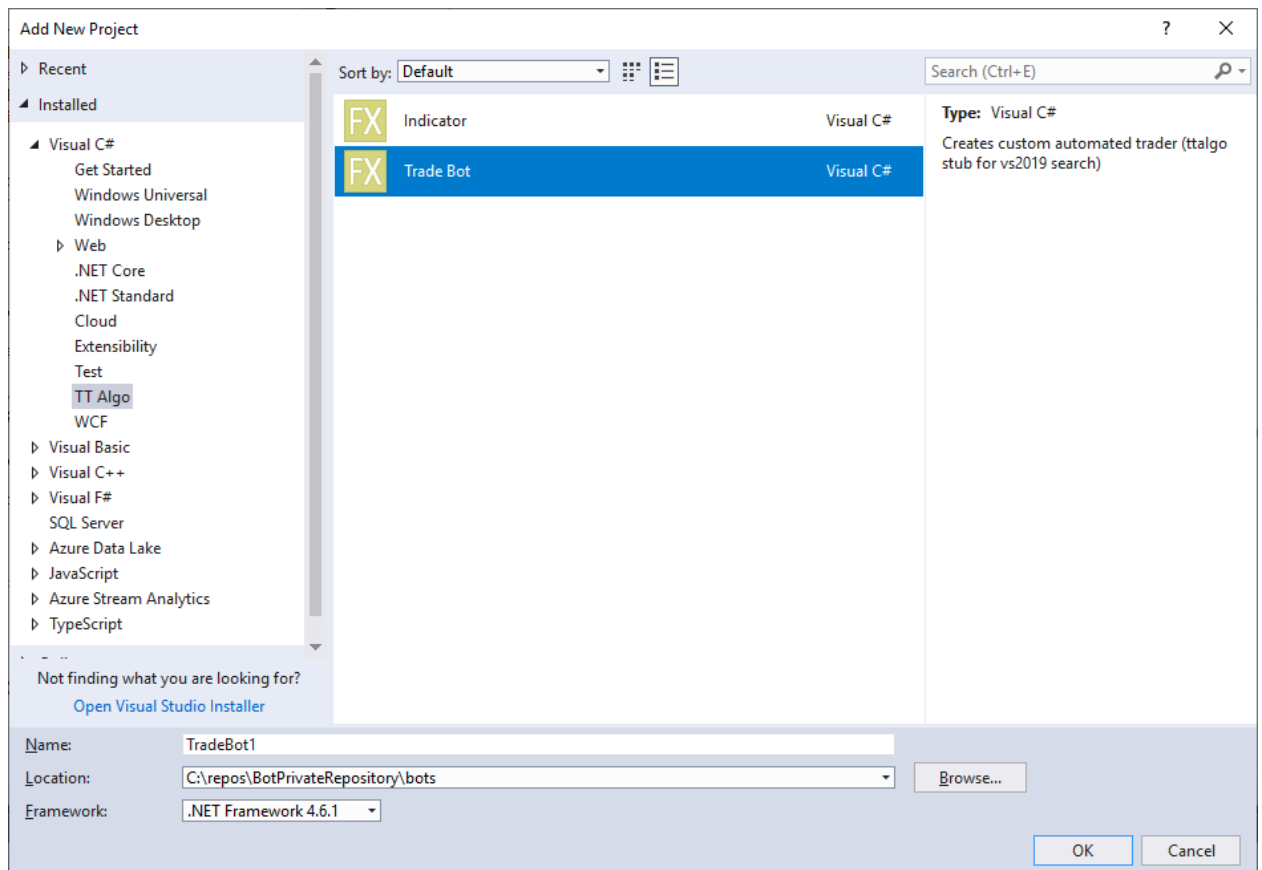
Creating a bot

Tutorial on creating a basic Algo Studio bot using Visual Studio and the C# programming language.

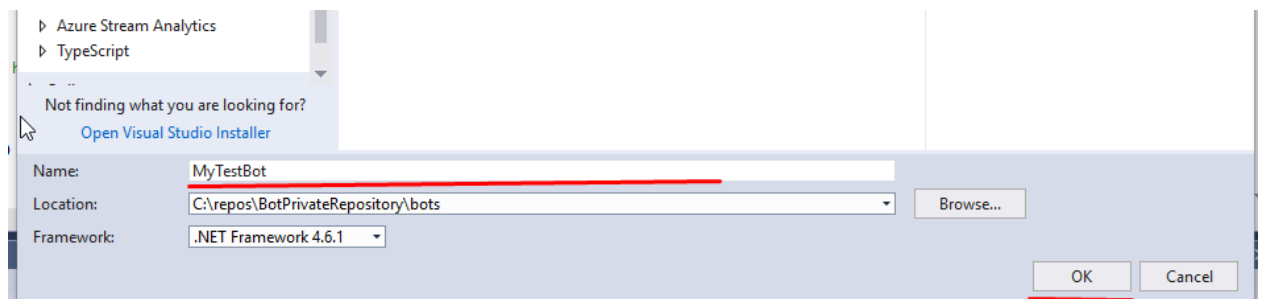
Steps:

1. Create a new project in **Visual Studio**.

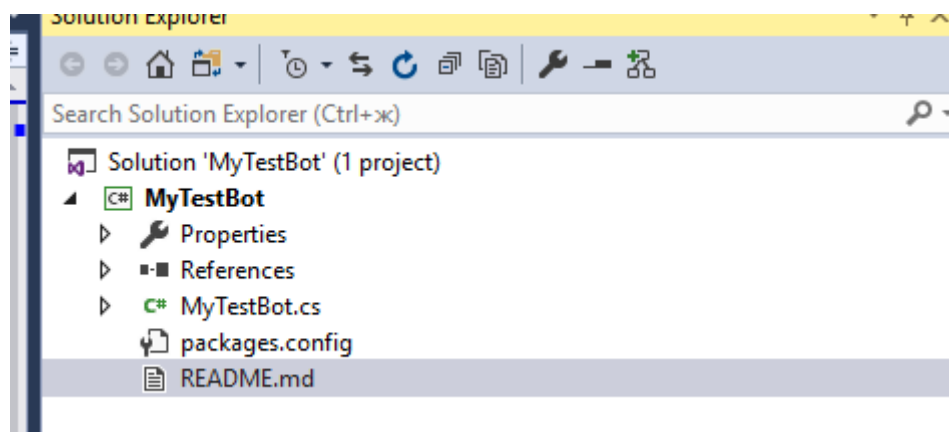
For the convenience of the user, a project template was developed. Run the **TickTrader.Algo.VS.Package.vsix** file to install it. The file will find studios on the computer and install in them.



2. After installing the extension, 2 new projects will appear: **Indicator** and **Trade Bot**. Select the **Trade Bot**.
3. Set the name and click **OK**.



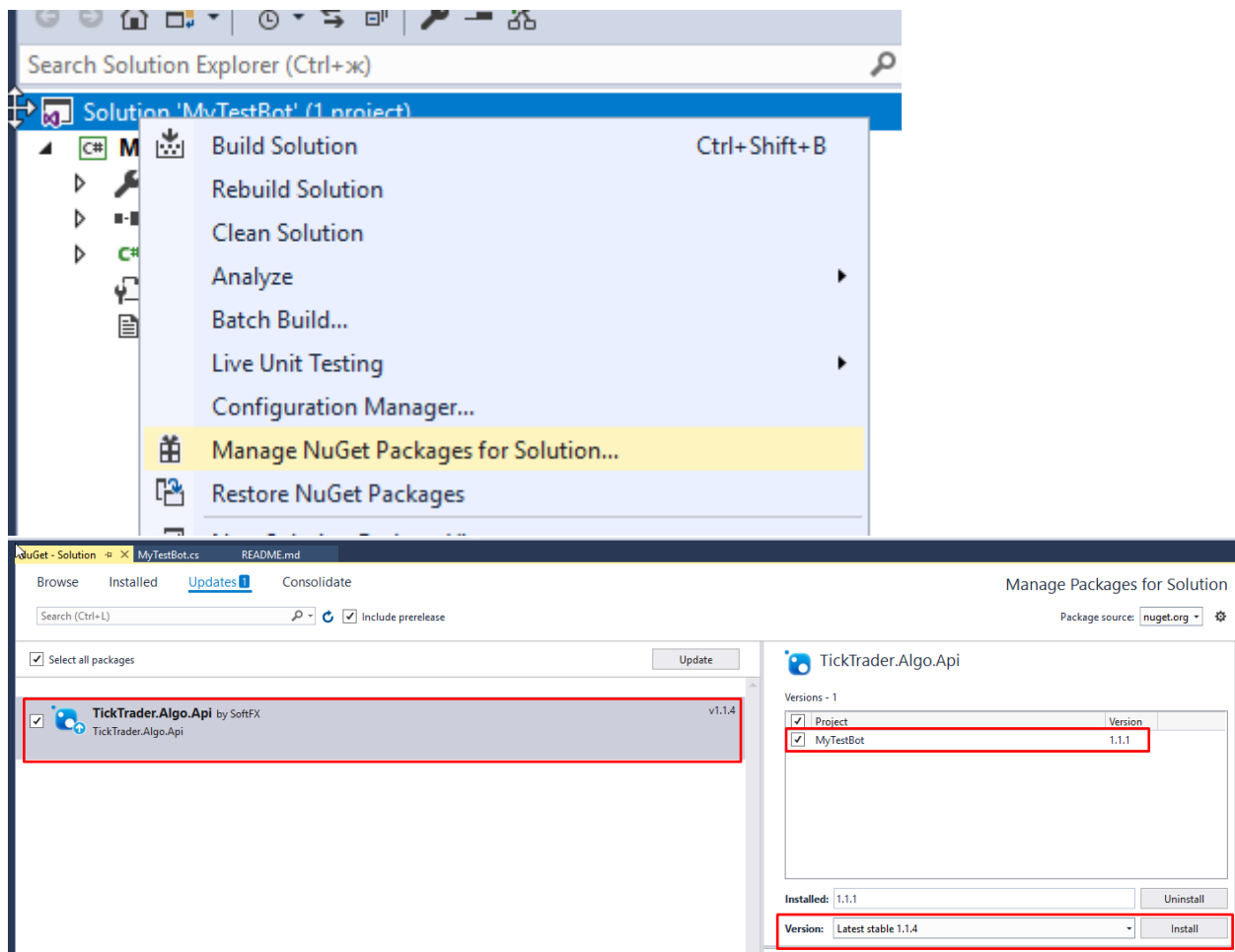
4. In the **Solution Explorer** section, you'll see the following structure:



where

- **MyTestBot.cs** – a file with the code of the created bot.

- **README.md** – you can add a description of your bot in Markdown format.
5. If you want to use the latest version of the **API**, check for updates in **NuGet explorer**.



Select the **TickTrader.Algo.Api** package -> Select the **project** (MyTestProject) -> Select the **version** -> Click **Install**.

6. After creating the **TradeBot** project you will get the following template:

```

using System.Linq;
using TickTrader.Algo.Api;

namespace MyTestBot
{
    [TradeBot(Category = "Misc", DisplayName = "MyTestBot", Version = "1.0")]
    0 references
    public class MyTestBot : TradeBot
    {
        [Parameter(DisplayName = "Param 1", DefaultValue = 2)]
        0 references
        public int IntParam { get; set; }

        [Parameter(DisplayName = "Param 2", DefaultValue = 1.2)]
        0 references
        public double DoubleParam { get; set; }

        2 references
        public enum Variants { Varitan1, Variant2, Variant3 }
        [Parameter(DisplayName = "Param 3", DefaultValue = Variants.Variant3)]
        0 references
        public Variants EnumParam { get; set; }

        [Input]
        0 references
        public DataSeries PriceInput { get; set; }

        0 references
        protected override void Init()
        {
            // TO DO: Put your initialization logic here...
        }

        0 references
        protected override void OnQuote(Quote quote)
        {
            // TO DO:
        }
    }
}

```

It describes test parameters so that a user can see how they are set.

Select **TradeBot** and press **F12** to see the trading bots functionality.

```

TradeBot(Category = "Misc", DisplayName = "MyTestBot", Version =
0 references
public class MyTestBot : TradeBot
{
    [Parameter(DisplayName = "P
0 references
class TickTrader.Algo.Api.TradeBot

```

```

namespace TickTrader.Algo.Api
{
    public class TradeBot : AlgoPlugin
    {
        public TradeBot();

        public DateTime UtcNow { get; }
        public StatusApi Status { get; }
        public DateTime Now { get; }
        protected bool IsStopped { get; }

        public OrderCmdResult CancelOrder(string orderId);
        public Task<OrderCmdResult> CancelOrderAsync(string orderId);
        public OrderCmdResult CloseOrder(string orderId, double? volume = null);
        public Task<OrderCmdResult> CloseOrderAsync(string orderId, double? volume = null);
        public OrderCmdResult CloseOrderBy(string orderId, string byOrderId);
        public Task<OrderCmdResult> CloseOrderByAsync(string orderId, string byOrderId);
        public Timer CreateTimer(TimeSpan period, Action<Timer> callback);
        public Timer CreateTimer(int periodMs, Action<Timer> callback);
        public Task Delay(TimeSpan period);
        public Task Delay(int periodMs);
        public OrderCmdResult LimitBuy(string symbol, double volume, double price, double? maxVisibleVolume = null, double? sl = null, double? tp = null, string comment = "", string tag = null);
        public OrderCmdResult LimitSell(string symbol, double volume, double price, double? maxVisibleVolume = null, double? sl = null, double? tp = null, string comment = "", string tag = null);
        public OrderCmdResult LimitSell(double volume, double price, double? maxVisibleVolume = null, double? sl = null, double? tp = null, string comment = "", string tag = null);
        public OrderCmdResult MarketBuy(double volume, double? tp = null, double? sl = null, string comment = "", string tag = null);
        public OrderCmdResult MarketBuy(string symbol, double volume, double? tp = null, double? sl = null, string comment = "", string tag = null);
        public OrderCmdResult MarketSell(double volume, double? sl = null, double? tp = null, string comment = "", string tag = null);
        public OrderCmdResult MarketSell(string symbol, double volume, double? sl = null, double? tp = null, string comment = "", string tag = null);
        public OrderCmdResult ModifyOrder(string orderId, double price, double? sl = null, double? tp = null, string comment = "");
        public OrderCmdResult ModifyOrder(string orderId, double? price, double? stopPrice, double? maxVisibleVolume = null, double? sl = null, double? tp = null, string comment = "");
        public Task<OrderCmdResult> ModifyOrderAsync(string orderId, double price, double? sl = null, double? tp = null, string comment = "");
        public Task<OrderCmdResult> ModifyOrderAsync(string orderId, double? price, double? stopPrice, double? maxVisibleVolume = null, double? sl = null, double? tp = null, string comment = "");
        public OrderCmdResult OpenLimitIoC(string symbol, OrderSide side, double volume, double price, double? sl = null, double? tp = null, string comment = "");
        public Task<OrderCmdResult> OpenLimitIoCAsync(string symbol, OrderSide side, double volume, double price, double? sl = null, double? tp = null, string comment = "");
        public OrderCmdResult OpenLimitOrder(OrderSide side, string symbol, double volume, double price, double? maxVisibleVolume = null, double? sl = null, double? tp = null, string comment = "");
        public OrderCmdResult OpenMarketOrder(OrderSide side, double volume, double? tp = null, double? sl = null, string comment = "");
        public OrderCmdResult OpenMarketOrder(string symbol, OrderSide side, double volume, double? tp = null, double? sl = null, string comment = "");
        public OrderCmdResult OpenOrder(string symbol, OrderType type, OrderSide side, double volume, double price, double? sl = null, double? tp = null, string comment = "");
        public Task<OrderCmdResult> OpenOrderAsync(string symbol, OrderType type, OrderSide side, double volume, double price, double? sl = null, double? tp = null, string comment = "");
        public Task<OrderCmdResult> OpenOrderAsync(string symbol, OrderType type, OrderSide side, double volume, double? maxVisibleVolume = null, double? sl = null, double? tp = null, string comment = "");
        public OrderCmdResult OpenStopLimitOrder(OrderSide side, string symbol, double volume, double price, double stopPrice, double? maxVisibleVolume = null, double? sl = null, double? tp = null, string comment = "");
    }
}

```

7. Bots have the following methods:

```

References
protected override void Init()
{
    // TO DO: Put your initialization logic here...
}

References
protected override void OnQuote(Quote quote)
{
    // TO DO:
}

References
protected override void OnStart()
{
    base.OnStart();
}

References
protected override void OnStop()
{
    base.OnStop();
}

```

```

References
protected override void OnRateUpdate(RateUpdate update)
{
    base.OnRateUpdate(update);
}

```

- **Init** – the method will be invoked, after running the bot in Algo Studio.
- **OnStart** – the method will be invoked right before the bot start.
- **OnQuote** – the method will be invoked with a new incoming quote.
- **OnStop** – the method will be invoked with the bot disabling.
- **OnRateUpdate** – the method will be invoked with bars updating.

8. The example of a basic bot:

Condition: The bot works on a **Cash** account, and places Limit orders. It has 3 parameters:

- *CountOrders* – number of orders to be placed (int).
- *TradeVolume* – volume value (double).
- *TradeSide* – side (enum).

The code example is given below:

using TickTrader.Algo.Api;

namespace MyTestBot

```

{
    [TradeBot(Category = "Misc", DisplayName = "MyTestBot", Version = "1.0", Description = "I test bot")]
    public class MyTestBot : TradeBot
    {
        private int _countOrders = 0;

        [Parameter(DisplayName = "Count", DefaultValue = 5)]
        public int CountOrders { get; set; }

        [Parameter(DisplayName = "Volume", DefaultValue = 0.1)]
        public double TradeVolume { get; set; }

        [Parameter(DisplayName = "Side", DefaultValue = OrderSide.Buy)]
        public OrderSide TradeSide { get; set; }

        protected override void Init()
        {
            if (Account.Type != AccountTypes.Cash)
            {
                PrintError("Invalid account type");
                Exit();
                return;
            }
        }
    }
}

```

```

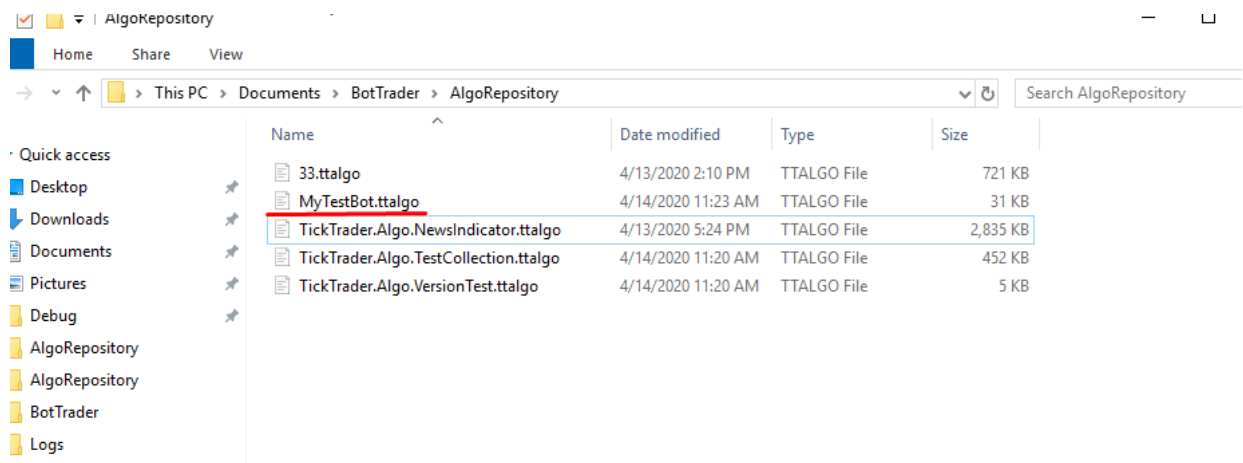
protected override void OnQuote(Quote quote)
{
    if (_countOrders++ < CountOrders)
    {
        var price = TradeSide == OrderSide.Buy ? Symbol.Bid : Symbol.Ask;
        OpenLimitOrder(TradeSide, Symbol.Name, TradeVolume, price);
    }
    else
        Exit();
}

protected override void OnStart()
{
    base.OnStart();
}

protected override void OnStop()
{
    Print("Bye!");
}
}

```

After building, the package with the project will automatically appear in the Algo Studio Repository and will be available in the application.



You can drag the package to the chart and view the bot's settings.

New Bot Instance

General Inputs

Description
I test bot

Main Symbol
Main Symbol EURUSD Bid
TimeFrame S10

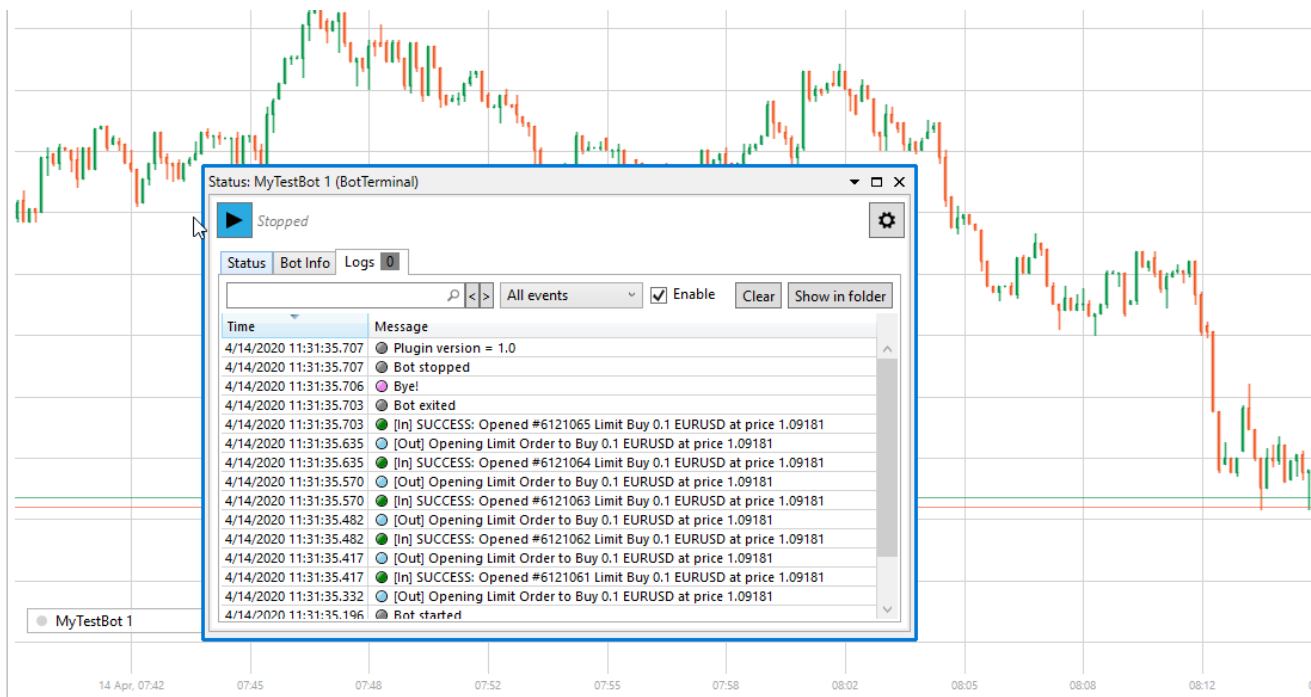
Load settings... Save settings...

Parameters
Count Volume
5 0.1
Side
Buy

Ok Cancel

Volume

9. Run the bot.



Trade						
Currency		Volume		Locked Volume		
EUR		592 312.02000		0.00000		5
GBP		123 214 412 409.30		0.00		1
USD		2 322 501 545.65		54 590.50		2

ID	Symbol	Modified Time	Type	Volume	Open Price	P
6121061	EURUSD	4/14/2020 08:31:35.331	Buy Limit	0.1/0.1	1.09181	0
6121062	EURUSD	4/14/2020 08:31:35.414	Buy Limit	0.1/0.1	1.09181	0
6121063	EURUSD	4/14/2020 08:31:35.479	Buy Limit	0.1/0.1	1.09181	0
6121064	EURUSD	4/14/2020 08:31:35.566	Buy Limit	0.1/0.1	1.09181	0
6121065	EURUSD	4/14/2020 08:31:35.631	Buy Limit	0.1/0.1	1.09181	0

