MultiConnect

Документация к торговому роботу

«MultiConnect»

Содержание

История изменений

Версия 1.0.1, 24.05.2016

1. Обновлено описание алгоритма примера

Версия 1.0.2, 10.09.2016

1. Добавлен раздел *Подсчет скорости транзакций*
2. Добавлено ограничение на количество портфелей и финансовых инструментов

Версия 1.1.2, 28.03.2017

1. Добавлен параметр *Save/load trades* в настройки приложения, позволяющий сохранять/-загружать таблицу сделок при закрытии/открытии приложения
2. Добавлен экспорт таблиц в *CSV* файл
3. Ограничен размер таблиц лога и сделок до 100000 строк, при переполнении "старые" записи будут автоматически удаляться

Версия 1.1.3, 11.08.2017

1. Добавлена торговля "в файл": добавлено значение *Client code*, равное *To file*
2. Добавлен фильтр некоторых сообщений в логе, такие сообщения будут отображаться в логе 1 раз в 10 секунд и будут отмечены в конце сообщения символом xN, где N – количество непоказанных сообщений

Руководство пользователя

Соглашение об ответственности

«Финансовая Компания «Викинг» не несет ответственности за умышленные попытки пользователя дестабилизировать работу торгового робота.

Архитектура

Торговый робот *«MultiConnect»* состоит из двух частей: серверной и клиентской. Серверная часть робота – это то место, где непосредственно выполняется сам торговый алгоритм. Клиентская часть робота – это графический интерфейс пользователя (GUI). Отделение графического интерфейса от торгового алгоритма позволяет разнести по разным физическим устройствам торговую часть и интерфейс пользователя, так торговый алгоритм может быть запущен на физическом сервере или виртуальной машине под управлением ОС семейства Linux в зоне колокации Московской биржи, в то время как графический интерфейс пользователя устанавливается на персональный компьютер клиента.

Принцип совместной работы двух частей торгового робота (серверной и клиентской) состоит в следующем: все настройки всегда хранятся на клиентской части, если вы запускаете серверную часть то она запускается не инициализированной настройками (т.е. торговых портфелей в ней нет и торговать она не будет). При первом подключении клиентской части робота к серверной, серверная часть получает настройки торговых портфелей и хранит их до своего (серверной части) выключения.

Следует четко понимать следующие моменты, связанные с работой робота:

- *если вы отключаете клиентскую часть от серверной, но не выключаете при этом серверную часть, то все настройки в ней остаются и если торговля не остановлена, то она будет продолжаться при выключенной клиентской части;*
- *если вы подключаетесь клиентской частью к серверной, которая не перезапускалась и уже заинициализирована настройками, то настройки загрузятся уже из серверной части в клиентскую, именно поэтому все настройки портфелей клиентской части рекомендуется проводить будучи подключенным к серверной части робота.*

Далее речь пойдет только о настройке клиентской части, при этом подразумевается, что серверная часть запущена, и подключение ко всем требуемым торговым площадкам установлено.

Важно: ЗАПРЕЩЕНО два и более одновременных подключения клиентской части к серверной, одновременное подключение приведет к потере данных из-за конфликта графических частей при подключении к серверной.

Настройка подключения

Для начала работы с торговым роботом необходимо запустить клиентскую часть. Открывшийся графический интерфейс будет иметь вид, как показано на рисунке 1.

Если это первый запуск GUI, то необходимо настроить подключение графической части к роботу, для этого нужно выбрать следующий  пункт главного меню , в открывшемся окне указать требуемые данные (Рис. 2). Обычно это *Host* и *Robot port*. Если вы подключаетесь с использованием SSL (например, вам выдали SSL сертификат, т.е. файл с расширением *.crt*) то поставьте галку *Use SSL* и укажите путь к SSL сертификату в поле *Cert path*. Далее следует нажать кнопку *Apply* для применения изменений. При последующих запусках робота настройку соединения выполнять не требуется.

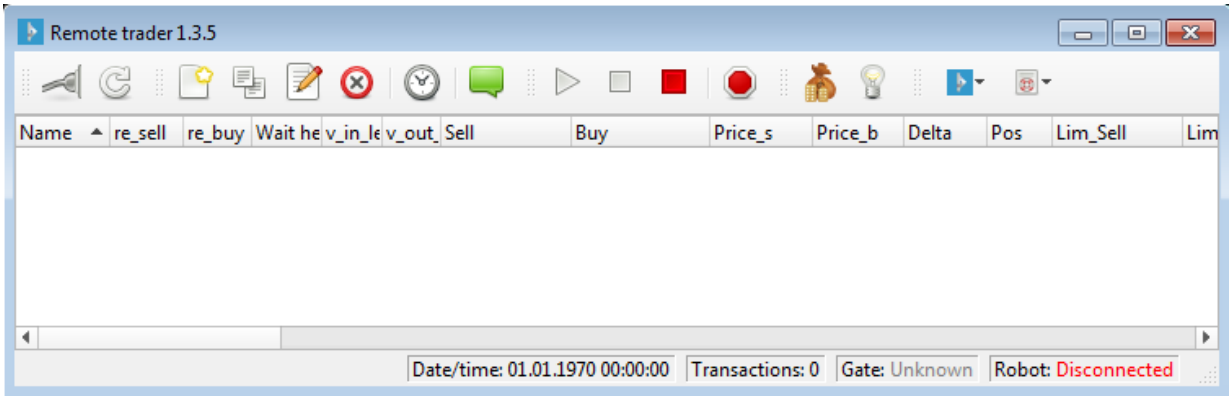

Рис. 1: Графический интерфейс пользователя до подключения

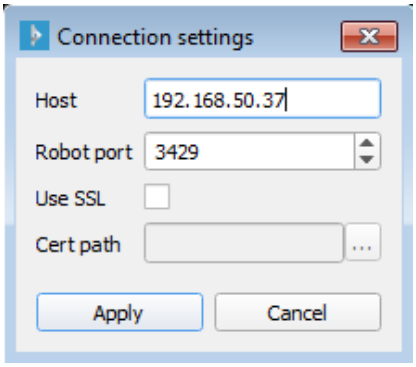

Рис. 2: Настройка подключения к серверной части робота

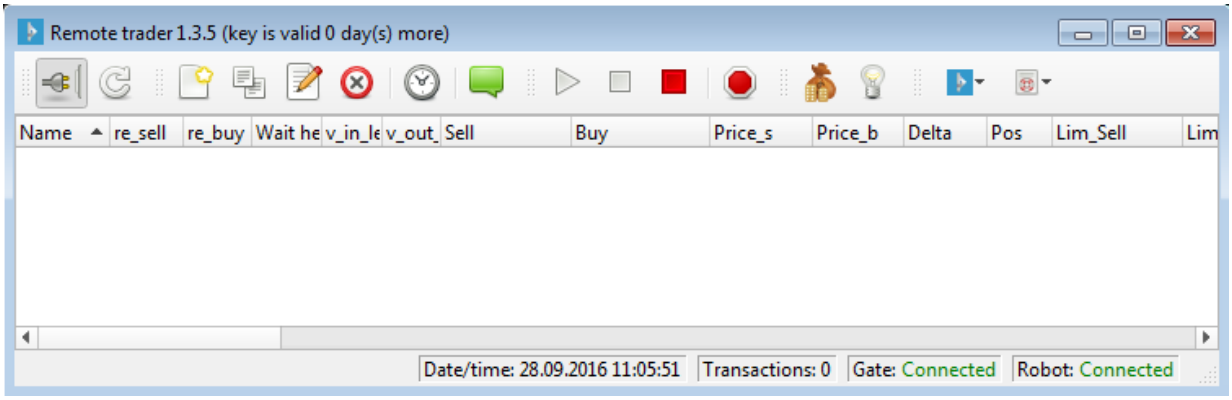

Рис. 3: Графический интерфейс пользователя, соединение установлено

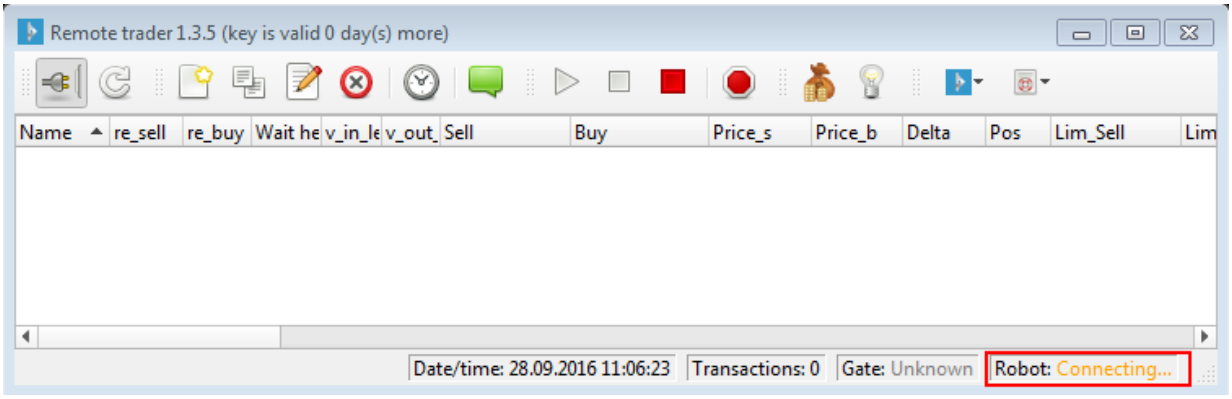

Рис. 4: Графический интерфейс пользователя, попытка установки соединения

Для подключения к серверной части робота необходимо нажать на кнопку подключения  , после чего GUI примет вид как представлено на рисунке 3, если вместо этого GUI имеет вид как представлено на рисунке 4, то следует проверить указанные настройки подключения (Рис. 2), если настройки верны, необходимо обратиться к контактному лицу. Если подключение выполнено успешно, то следует выполнить обновление списка финансовых инструментов, нажав кнопку  .

Важно: Изменение всех торговых настроек следует проводить ТОЛЬКО при наличии подключения к серверной части робота.

Настройка портфелей и торговые функции

После того, как выполнено подключение к серверной части робота, и робот имеет вид как показано на рисунке 3, можно приступать к настройке нового торгового портфеля. Если это не первый запуск робота, то в таблице будут отображены уже существующие портфели. Для этого необходимо нажать кнопку добавления портфеля  , в открывшемся окне нужно указать необходимые настройки (Рис. 5). В каждом торговом портфеле может быть несколько инструментов, один из которых является главным – по нему рассчитывается позиция всего портфеля. Кроме создания нового портфеля доступны функции редактирования существующего портфеля, создания его копии, а так же удаления портфеля, для этого служат кнопки  ,  и  соответственно. Чтобы отредактировать портфель, создать его копию или удалить его, необходимо чтобы нужный портфель был выбран в таблице. Так же существует возможность экспортировать настройки портфелей ( в главном меню) и импортировать настройки портфелей из имеющегося файла настроек ( в главном меню ). При импорте будут добавлены портфели с уникальными именами, имеющиеся портфели изменены или удалены НЕ будут.

Для просмотра сделок, совершенных роботом, предусмотрена кнопка  , в открывшейся таблице отображаются раздвижки по совершенным сделкам (если строк в таблице становится более 100000 ”старые” строки будут автоматически удаляться).

При закрытии GUI все настройки сохраняются, и при следующем включении будут доступны все созданные портфели. После добавления портфелей основной экран робота будет выглядеть как показано на рисунке 6. Изменение всех торговых настроек следует проводить только при наличии подключения к серверной части робота.

Для включения и остановки торговли по всем портфелям одновременно служат кнопки  и  соответственно. Если необходимо задать расписание работы робота, т.е. отрезки времени, когда будет производиться торговля, то нажатием на кнопку  нужно вызвать окно настройки расписания. В расписании следует указывать время для того же часового пояса, который выбран на машине, где запущена серверная часть торгового робота (при торговле на Московской бирже, предполагается использование московского времени), в не зависимости от того, какое время задано на клиентской машине.

В строке состояния в нижней части главного окна программы расположен суммарный счетчик транзакций (то есть попыток выставления и снятия заявок) по всем транзакционным подключениям к бирже (*Transactions*). Также в строке состояния расположен индикатор состояния подключения всех подключений торгового робота к бирже (*Gate*), при наведении курсора мыши на индикатор появится всплывающая подсказка с подробной расшифровкой состояния всех подключений.

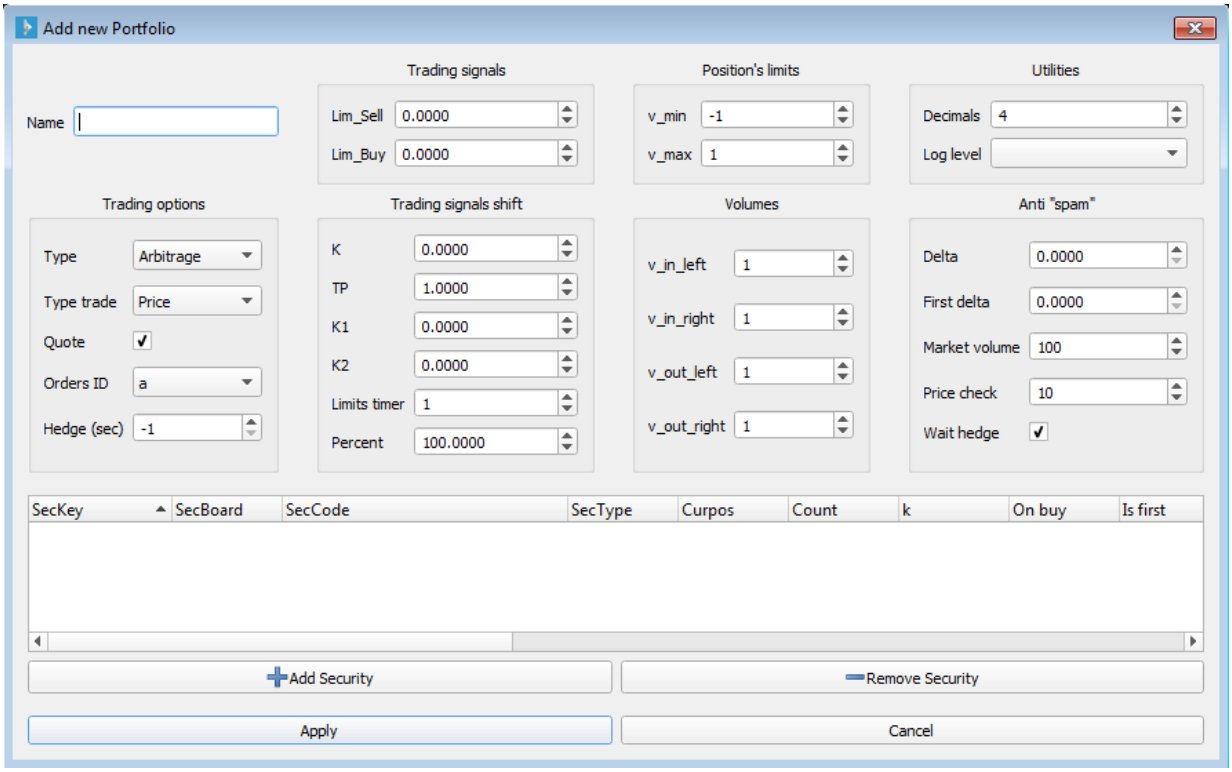

Рис. 5: Добавление нового портфеля

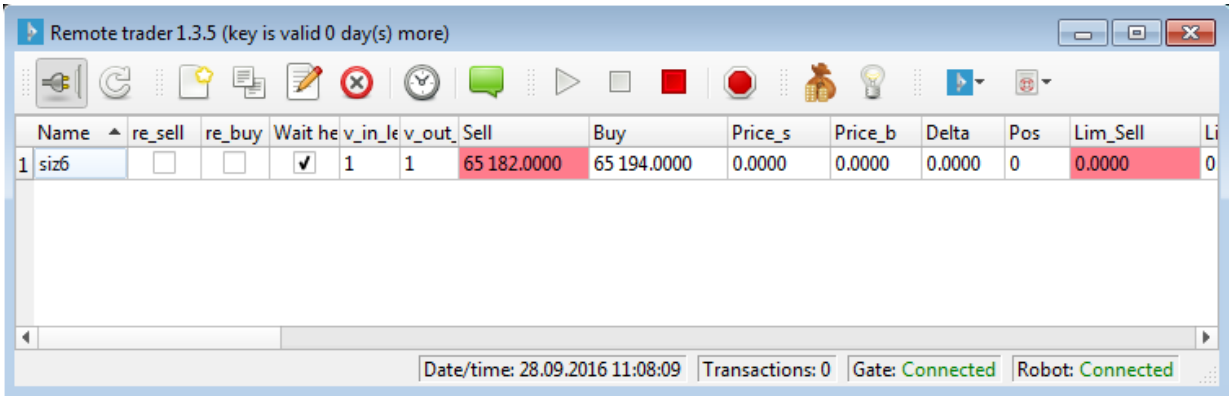

Рис. 6: Графический интерфейс пользователя

Важно: По умолчанию вы можете добавить не более 50-ти портфелей и при этом не более 100-а финансовых инструментов суммарно во все портфели, при превышении указанных чисел, превысивший ограничения портфель будет автоматически отключен.

Сброс статусов заявок робота

В торговом роботе есть ”механизм отслеживания заявок”, бывают ситуации, например, когда происходят сбои в работе биржи, что заявка выставляется роботом, а потом снимается биржей, никак не информируя робот о снятии. В таком случае заявка ”зависает” в своем текущем, внутреннем для робота, статусе. Для сброса статусов всех заявок портфеля есть кнопка . Пользоваться это кнопкой можно ТОЛЬКО в КРАЙНИХ случаях, когда торговля по портфелю отключена и Вы уверены что нет ни одной активной заявки по данному портфелю, в противном случае робот потеряет активные заявки, что приведет к неправильной позиции по бумагам в роботе.

Неторговые функции

Во время работы робота могут возникать различные ошибки при выставлении и удалении заявок, такие ситуации не являются нештатной работой робота и могут быть вызваны причинами, не связанными с некорректной работой торгового робота, например, отсутствием денег на счете клиента. Для настройки уведомлений об ошибках, информационных сообщениях и поведения главного окна программы выберите следующий пункт главного меню . Все ошибки и еще некоторую информацию, связанную с работой программы, можно посмотреть в логе, нажав на кнопку (если строк в таблице становится более 100000 "старые" строки будут автоматически удаляться). Некоторые слишком часто приходящие сообщения будут отображаться в логе не все, а 1 раз в 10 секунд и будут отмечены в конце сообщения символом xN, где N – количество непоказанных сообщений. Время в логе – это время прихода сообщения из серверной части в клиентскую, это локальное время машины, на которой запущена клиентская часть. По этому времени НЕЛЬЗЯ судить о скорости работы робота.

Стоит заметить, что все таблицы робота настраиваемые. Можно менять местами столбцы, а так же отключать ненужные. Для перетаскивания столбца достаточно зажать левую кнопку мыши на его заголовке и перетащить столбец в нужное место таблицы. Для отключения лишних столбцов нужно сделать правый клик мышью на заголовке таблицы, после этого откроется контекстное меню, как показано на рисунке 7. В этом же контекстном меню есть пункты применения фильтра к таблице и копирования/экспорта таблицы в *csv* формате. Для таблицы сделок, совершенных роботом, и таблицы лога в контекстном меню есть возможность очистить таблицу.

Обновление программы и документация

Клиентская часть робота автоматически проверяет наличие обновлений при запуске и далее 1 раз в час (убедитесь, что адрес http://fkviking.ru/downloads НЕ добавлен в черный список вашего фаервола). При наличии обновлений на панели появится иконка (вместо иконки меню помощи), нажав на которую вы сможете открыть меню помощи и, выбрав соответствующий пункт меню, скачать инсталлятор новой версии программы. Для проверки наличия обновлений программы в ручном режиме необходимо выбрать следующий пункт меню помощи.

Для того чтобы открыть файл с документацией, используйте следующий пункт меню помощи, документация откроется в просмотрщике *pdf* файлов, установленном в системе по-умолчанию.

Важно: При наличии обновлений клиентской части рекомендуется ВСЕГДА обновлять программу, во избежании конфликта версий и невозможности подключения "старой" клиентской части к более новой (обновляемой без Вашего участия) серверной части. Серверная часть робота обновляется независимо от клиентской и обычно ее обновляют ночью, после окончания торговых сессий на всех рынках, поэтому рекомендуется НЕ оставлять активных заявок после окончания торгов, так как если серверная часть будет перезапущена, то она уже не "увидит" оставленных заявок, и эти заявки будут потеряны роботом и не будут корректно обработаны, что приведет к несоответствию позиции в роботе и реальной позиции на счете (если эти заявки исполнятся).

Подсчет скорости транзакций

При наведении курсора мыши на надпись, отображающую число транзакций, совершенных роботом (*Transactions*), появится всплывающая подсказка, в которой отображена статистика ско-

Рис. 7: Отключение столбцов таблицы

рости транзакций для каждого транзакционного подключения. Данные отображены в таблице, строки таблицы соответствуют имеющимся в роботе транзакционным подключениям, столбцы таблицы:

1. *connect* – название подключения;
2. *all* – суммарное количество транзакций по подключению;
3. *> 0.5* – количество транзакций с round-trip-ом больше 0.5 мс по подключению;
4. *> 1* – количество транзакций с round-trip-ом больше 1 мс по подключению;
5. *> 2* – количество транзакций с round-trip-ом больше 2 мс по подключению;
6. *> 4* – количество транзакций с round-trip-ом больше 4 мс по подключению;
7. *> 8* – количество транзакций с round-trip-ом больше 8 мс по подключению;
8. *> 16* – количество транзакций с round-trip-ом больше 16 мс по подключению;
9. *adds* – суммарное количество успешных транзакций на выставление заявки по подключению;

10. *dels* – суммарное количество успешных транзакций на удаление заявки по подключению;
11. *moves* – суммарное количество успешных транзакций на перемещение заявки по подключению;
12. *rejects* – суммарное количество отвергнутых транзакций по подключению;
13. *avg* – среднее время round-trip-а транзакций (в микросекундах) по подключению;
14. *avga* – среднее время round-trip-а транзакций на выставление заявки (в микросекундах) по подключению;
15. *avgd* – среднее время round-trip-а транзакций на удаление заявки (в микросекундах) по подключению;
16. *avgm* – среднее время round-trip-а транзакций на перемещение заявки (в микросекундах) по подключению;
17. *avgr* – среднее время round-trip-а отвергнутых транзакций (в микросекундах) по подключению.

Важно: Время замеряется именно для round-trip-а, т.е. это время между моментом отправки транзакции и моментом прихода ответного сообщения на данную транзакцию.

Возможные проблемы и их решение

Описание наиболее распространенных ошибок в работе робота и способов их устранения.

- **Проблема**

  При подключении к серверной части меняются настройки торговых портфелей.

  Решение

  Все правильно, так и должно быть. Все изменения настроек портфелей рекомендуется производить только когда выполнено подключение к серверной части. В противном случае после подключения загрузятся те настройки, которые были переданы на сервер ранее.

- **Проблема**

  Торговля включается и выключается самопроизвольно.

  Решение

  Проверьте, не задано ли расписание торговли.

- **Проблема**

  Отсутствует нужный инструмент в списке инструментов.

  Решение

  В таком случае следует обновить список инструментов, нажав кнопку  .

- **Проблема**

  Робот ставит заявки по второй ноге в "непонятном", неправильном объеме.

  Решение

  Данная ситуация может возникать после какого-либо сбоя на бирже, когда биржа не прислала необходимую информацию по заявкам робота и в роботе "зависли" внутренние статусы заявок. В данной ситуации необходимо остановить торговлю по проблемному портфелю, убедиться что по данному портфелю нет активных заявок в рынке и сбросить статусы заявок, нажав кнопку  .

Наиболее распространенные ошибки, возникающие при работе программы, и способы их устранения

Важно: большинство возникающих ошибок в работе робота, это не ошибки самого робота, а проблемы со связью, между клиентской и серверной частями, вызванные плохим качеством интернет соединения, и ошибки выставления заявок, например, из-за нехватки денег на счете.

Торговые ошибки:

- **Ошибка**

  *Order adding error on *, error: REASON_NO_MONEY*

  или

  *order adding error on *, order exceeds limit*

  Описание и способ устранения

  Нехватка денег на счете. Проверьте хватает ли у вас денег на счете для выставления заявки по данной бумаге данного объема. Торговля будет остановлена автоматически.

- **Ошибка**

  *order adding error on *, continue trading, error: REASON_FLOOD*

  Описание и способ устранения

  Флуд-контроль, превышен лимит максимального разрешенного количества транзакций (удалений/снятий заявок) в секунду для данного логина. Для данного робота 30 транзакций в секунду (единица транзакционной активности биржи, принятая на текущий момент) – это элементарно.

Неторговые ошибки:

- **Ошибка**

  Error on sendMsg (<class 'ConnectionRefusedError'>, ConnectionRefusedError(10061, 'Подключение не установлено, т.к. конечный компьютер отверг запрос на подключение', None, 10061, None))

  или

  Error on sendMsg (<class 'ConnectionRefusedError'>, ConnectionRefusedError(111, 'Connection refused'))

  Описание и способ устранения

  Нет соединения с роботом, скорее всего робот просто не запущен в данный момент. Проверьте запущена ли серверная часть робота, если у вас нет на это прав, то обратитесь к администратору за информацией о состоянии робота. Если данная ошибка продолжалась порядка нескольких минут и после этого клиентская часть снова подключилась к серверной, то, возможно, серверная часть по какой-то причине была перезапущена.

- **Ошибка**

  Error on sendMsg (<class 'socket.timeout'>, timeout('timed out',))

  или

  Error on sendMsg (<class 'OSError'>, OSError(113, 'No route to host'))

или

Error on sendMsg (<class 'OSError'>, OSError(101, 'Network is unreachable'))

Описание и способ устранения

Нет соединения с компьютером на котором запущена серверная часть. Вероятнее всего указан не верный IP-адрес серверной части в настройках подключения, либо на удаленной машине запущен фаервол, который запрещает входящие подключения, либо отсутствует соединение с интернетом (или с локальной сетью) на машине где запущена клиентская часть, либо отсутствует соединение с интернетом (или с локальной сетью) на машине где запущена серверная часть. Если отсутствует соединения с сетью на машине, на которой запущена серверная часть, то сообщение всегда *socket.timeout*.

- **Ошибка**

  Cann't parse message from server (<class 'xml.etree.ElementTree.ParseError'>, ParseError('no element found: line 1, column 0',))

  Описание и способ устранения

  Данная ошибка сопутствует предыдущей ошибке, так как программа не получила валидное сообщение от серверной части и не смогла его корректно обработать.

- **Ошибка**

  Multiple client connection is not allowed!!!

  Описание и способ устранения

  Запрещено одновременное подключение нескольких клиентских частей к серверной. Убедитесь что у Вас нигде не запущена вторая клиентская часть уже подключенная к серверной части. Подождите 10 секунд, и попробуйте подключиться снова.

- **Ошибка**

  Error on sendMsg (<class 'OSError'>, OSError(9, 'Bad file descriptor'))

  или

  Error on sendMsg (<class 'OSError'>, OSError(10038, 'Сделана попытка выполнить операцию на объекте, не являющемся сокетом', None, 10038, None))

  Описание и способ устранения

  Попытка подключения используя уже закрытый сокет. Скорее всего сокет был принудительно закрыт серверной частью из-за попытки подключиться одновременно второй клиентской частью

- **Ошибка**

  Error reading application version file ".version"from site, during update check: (<class 'urllib.error.URLError'>, URLError(TimeoutError(10060, 'Попытка установить соединение была безуспешной, т.к. от другого компьютера за требуемое время не получен нужный отклик, или было разорвано уже установленное соединение из-за неверного отклика уже подключенного компьютера', None, 10060, None)))*

  или

  Error reading application version file ".version"from site, during update check: (<class 'urllib.error.URLError'>, URLError(gaierror(-5, 'No address associated with hostname')))*

  Описание и способ устранения

  Программа не смогла проверить обновление на сервере из-за проблем со связью.

Описание параметров программы

Алгоритм примера

Пример робота реализует алгоритм соревнования на скорость. 2 инструмента в стратегии. За одним следим, по второму выставляемся. 1 параметр "K". Следим за бидом первого. Если остаток от деления бида, умноженного на K, на 10 равен 0 ($bid \times K \equiv 0 \pmod{10}$) и последняя кратная цена не равна новой, то кидаем заявки на покупку всех инструментов, кроме *Is first*. Цены заявок, по которым они точно не сведутся в сделки. После прихода подтверждения о выставлении каждая заявка снимается. Другими словами, когда цена первого удовлетворяет условию, то кидаем заявки по остальным, но по такой цене, чтобы не свелись в сделки. Когда выставятся, снимаем.

Архитектура примера

Сначала определимся с терминологией. Будем называть портфелем набор биржевых инструментов, которыми производится торговля. В роботе одновременно может осуществляться торговля несколькими портфелями. Главным или первым инструментом портфеля назовем инструмент, отмеченный в настройках галкой *Is first*. В реализованном тестовом алгоритме портфель может состоять из произвольного числа инструментов, больше или равного единице.

В примере используются три основных класса: *Portfolio<T>*, *SecurityFields<T>*, *Order*. Класс *Portfolio<T>* содержит параметры алгоритма, общие для всех инструментов портфеля, и методы, этот алгоритм реализующие. Этот класс является наследником класса *BasePortfolio<T>*. Два основных метода базового класса, один из которых в зависимости от стратегии должен быть переписан, это методы *eval()* и *eval_ob()*. Метод *eval()* вызывается при любом изменении лучшей цены на покупку или продажу любого из инструментов портфеля. Метод *eval_ob()* вызывается при любом изменении стакана любого из инструментов портфеля. Важно: Консистентность стаканов по всем инструментам портфеля гарантируется только на момент вызова метода *eval_ob()*.

Класс *SecurityFields<T>* содержит параметры алгоритма специфичные для каждого инструмента портфеля, а так же массив или пул заявок, которые будут выставляться по данному инструменту. Одновременно робот может отслеживать несколько заявок по каждому инструменту портфеля.

Класс *Order* содержит параметры заявки и статус ее исполнения, а так же наследует класс *OrderWrapper* и реализует методы *orderAdded(long long oNo, int code)*, *orderDeleted(long long oNo, int amount, int code)*, *orderDeal(long long oNo, int amount, int amount_rest, double price, int dir, double moment, char * key_for_swap)*, где реализовано отслеживание заявки. Следует понимать, отслеживание, реализованное в примере, рассчитано на торговлю одним лотом.
Важно: Методы *orderAdded(long long oNo, int code)* и *orderDeleted(long long oNo, int amount, int code)* для одной и той же заявки могут вызываться несколько раз, это связано с тем, что в некоторых биржевых подключениях данные о выставлении и удалении заявок могут приходить в двух независимых несинхронизированных биржевых потоках, а для увеличения скорости реакции робота производится обработка обоих потоков.

Замечания

1. Библиотека *MultiConnect* может работать как в однопоточном, так и многопоточном режиме. Поэтому рекомендуется воздержаться от использования механизмов синхронизации. В однопоточном режиме все используемые биржевые подключения опрашиваются последовательно в одном цикле. В многопоточной версии привязкой биржевых подключений к

ядрам управляет пользователь. Так же можно выбирать ядра, на которых будет выполняться алгоритм. На одном и том же ядре позволительно и опрашивать подключения, и исполнять алгоритм. Не рекомендуется реализовывать алгоритмы, выполнение которых при каждом изменении цен или стакана занимает дольше десяти микросекунд (10 мкс). Чем быстрее будет алгоритм, тем лучше.

2. При выставлении заявки следует округлить цену, указываемую в заявке, до шага цены по данному инструменту. В алгоритме примера округление цен производится при помощи макроса *ROUND_PRICE*, рекомендуется использовать его.

MultiConnect GUI

Настройка столбцов таблицы

Для добавления/удаления столбцов в таблицы необходимо редактировать файл в папке с установленной программой *«MultiConnect» config/extra.py*.

В файле *config/extra.py* есть объявление трех списков, которые отвечают за содержимое столбцов таблицы главного окна программы, за сохраняемые параметры (они же настраиваются в окне настроек портфеля) и за столбцы таблицы настройки финансовых инструментов портфеля, соответственно: *MAIN_TABLE_*, *ADD_PORTFOLIO_*, *ADD_PORTFOLIO_TABLE_*. В данные списки вы можете добавлять необходимые вам элементы как объекты класса *dict*. Примеры добавления столбцов в соответствии с необходимым типом данных так же описаны в файле *config/extra.py*. Содержимое по-умолчанию для файла *config/extra.py* представлено ниже (будьте внимательны с лишними переносами строк при копировании):

```
"""You can add your custom table columns in this file

Examples of cell types (value of 't' dict key):
Combobox, %combo, value MUST be int:
{'name':'on_buy', 'title':'On buy', 't':'%combo', 'combolist':[{'v':1, 't':'Buy'}, {'
    v':2, 't':'Sell'}], 'tt':'Contract order direction on portfolio buy', 'edit':RW, '
    default':1},

Combobox, %combo, with images:
{'name':'level', 'title':'Level', 't':'%combo', 'sub_t':'%i', 'edit':R, 'tt':'Message
     level', 'default':0, 'combolist':[{'v':0, 't':'DEBUG', 'image':'images/bug.png'},
     {'v':1, 't':'INFO', 'image':'images/information.png'}, {'v':2, 't':'WARNING', '
    image':'images/warning.png'}, {'v':3, 't':'ERROR', 'image':'images/error.png'}, {'
    v':4, 't':'CRITICAL', 'image':'images/error.png'}, {'v':5, 't':'ORDER', 'image':'
    images/error.png'}, {'v':6, 't':'MESSAGE', 'image':'images/message.png'}],},

Combobox with check boxes, %comboc, value MUST be int and power of two (it will be '
    and' with each other value):
{'name':'log_level', 'title':'Log level', 't':'%comboc', 'combolist':[{'v':1, 't':'
    Order state', 'tt':'Log state of orders'}, {'v':2, 't':'Option info', 'tt':'Log
    information about options'}, {'v':4, 't':'Price info', 'tt':'Log all price updates
    '}, {'v':8, 't':'Adding order reason', 'tt':'Log reasons for adding/not adding
    orders'}, {'v':16, 't':'Pos info', 'tt':'Log position information (real and
    theoretical)'}, {'v':32, 't':'Limits info', 'tt':'Log information about moving
    limits'}], 'edit':RW, 'tt':'Log level', 'default':0},

Combobox, %combos, value MUST be string:
{'name':'client_code', 'title':'Client code', 't':'%combos', 'combolist':[{'v':'', 't
    ':''}, ], 'tt':'Trade from this client code', 'edit':RW, 'default':''},

Int, %d:
{'name':'pos', 'title':'Pos', 't':'%d', 'edit':RW, 'tt':'Current portfolio position',
     'colorfill':posColorfill, 'range':(-1e9, 1e9), 'default':0},

Float (double in C++), %f:
```

{'name':'k', 'title':'k', 't':'%f', 'range':(−1e9, 1e9), 'tt':'Price indent of order', 'edit':RW, 'default':0},

String, %s:
{'name':'name', 'title':'Name', 't':'%s', 'edit':R, 'tt':'Portfolio name', 'default':''},

String, %s, with image:
{'name':'add_buy', 'title':'Buy clicker', 't':'%s', 'edit':R, 'tt':'Add buy order', 'default':'', 'sub_t':'%i', 'image':'images/buy.png'},

Bool, %b:
{'name':'re_sell', 'title':'re_sell', 't':'%b', 'edit':RW, 'tt':'Enable/disable sell', 'default':False},

Date time msec, %datetmsec:
{'name':'time', 'title':'Time', 't':'%datetmsec', 'f':'hh:mm:ss.zzz', 'edit':R, 'tt':'Message time', 'default':−1},

Date, %date:
as for %datetmsec with another format 'f'

Time, %t:
as for %datetmsec with another format 'f'

Time msec, %tmsec:
as for %datetmsec with another format 'f'

Date time, %datet:
as for %datetmsec with another format 'f'

Table row necessary keys:
- *name − field name in python code*
- *title − field title in GUI*
- *t − field type (types are described above)*
- *edit − is field editable (R) or not (RW)*
- *default − field default value (must be of 't' type)*

Table row extra keys:
- *tt − field tooltip*
- *hidden − field will be hidden*
- *combolist − combobox choice list, only for %combo, %comboc, %combos*
- *range − permitted value range, ony for %d, %f*
- *sub_t − sub type, only %i is available now*
- *image − image file path*
- *f − date time format (as Qt time format)*
- *colorfill − function for change cell color (there are examples below)*
- *f_disable − function for cell disable (there are examples below)*

```
Combolist keys:
        v - value, int for %combo and %comboc, string for %combos
        t - text
        tt - tooltip
        image - image file path
"""

"""Disable buy clicker"""
BUY_CLICKER_ = False

"""Disable sell clicker"""
SELL_CLICKER_ = False

"""Disable to_market clicker"""
TO_MARKET_CLICKER_ = False

"""Disable re_sell and remove it from table, use it if you need only one flag to
    enable both buy and sell
You must call remove_field("re_sell") in robot in portfolio's init_fields!!!"""
NO_RE_SELL_ = False

"""Disable re_buy and remove it from table, use it if you need only one flag to
    enable both buy and sell
You must call remove_field("re_buy") in robot in portfolio's init_fields!!!"""
NO_RE_BUY_ = False

"""Disable on_buy and remove it from table, use it if you need on_buy always BUY
You must call remove_field("on_buy") in robot in securities's init_fields!!!"""
NO_ON_BUY_ = False

"""Disable count and remove it from table, use it if you need count always 1
You must call remove_field("count") in robot in securities's init_fields!!!"""
NO_COUNT_ = False

"""Disable client_code and remove it from table, use it if you need always empty
    client_code
You must call remove_field("client_code") in robot in securities's init_fields!!!"""
NO_CLIENT_CODE_ = False

"""Maximum number of contracts in each portfolio"""
MAX_CONTRACTS_ = 10

"""Settings columns count"""
SETTINGS_COLS_COUNT_ = 2

"""Add close flag to timetable, you must add "close" attribute in timetable XML in
    portfolio's toString"""
TT_WITH_CLOSE_ = False

"""Add to_market flag to timetable, you must add "to_market" attribute in timetable
```

```
    XML in portfolio's toString"""
TT_WITH_TO_MARKET_ = False

"""Add to0 flag to timetable, you must add "to0" attribute in timetable XML in
    portfolio's toString"""
TT_WITH_TO0_ = False

"""It is timetable XML example
res << "\n<timetable begin=\"" << this->startStop[i].first << "\" end=\"" << this->
    startStop[i].second << "\" close=\"" << this->startStop[i].close << "\" to_market
    =\"" << this->startStop[i].to_market << "\" to0=\"" << this->startStop[i].to0 <<
    "\"/>";"""

"""Functions for table cells color change:
row is a table row with all it's fields
option is a QStyleOptionViewItem object
should return pair of QBrush-es: first is a main color, second is a selection color
these functions are called in initStyleOption method of table cell delegate
"""

def sellColorfill(row, option=None):
        if row.get('sell', 0) >= row.get('lim_s', 0) and not row.get('disabled',
            False):
                return (BRUSH_RED, BRUSH_RED_SELECT)
        return None

def buyColorfill(row, option=None):
        if row.get('buy', 0) <= row.get('lim_b', 0) and not row.get('disabled', False
            ):
                return (BRUSH_GREEN, BRUSH_GREEN_SELECT)
        return None

def resColorfill(row, option=None):
        if row.get('fin_res', 0) > 0 and not row.get('disabled', False):
                return (BRUSH_GREEN, BRUSH_GREEN_SELECT)
        if row.get('fin_res', 0) < 0 and not row.get('disabled', False):
                return (BRUSH_RED, BRUSH_RED_SELECT)
        return None

"""ALL fields from MAIN_TABLE_ table will be shown in main table, but they will NOT
    be saved if they are not in ADD_PORTFOLIO_

Predefined fields are:
{'name':'name', 'title':'Name', 't':'%s', 'edit':R, 'tt':'Portfolio name', 'default
    ':''},
{'name':'re_sell', 'title':'re_sell', 't':'%b', 'edit':RW, 'tt':'Enable/disable sell
    ', 'default':False},
{'name':'re_buy', 'title':'re_buy', 't':'%b', 'edit':RW, 'tt':'Enable/disable buy', '
    default':False},
{'name':'pos', 'title':'Pos', 't':'%d', 'edit':RW, 'tt':'Current portfolio position',
```

'colorfill':posColorfill, 'range':(-1e9, 1e9), 'default':0},
{'name':'start_nearest', 'title':'Start nearest', 't':'%t', 'edit':R, 'tt':'Nearest trading start', 'colorfill':startColorfill, 'default':-1},
{'name':'stop_nearest', 'title':'Stop nearest', 't':'%t', 'edit':R, 'tt':'Nearest trading stop', 'colorfill':stopColorfill, 'default':-1},
{'name':'use_tt', 'title':'Use timetable', 't':'%b', 'edit':RW, 'tt':'Use trading timetable', 'default':False},
{'name':'decimals', 'title':'Decimals', 't':'%d', 'edit':RW, 'tt':'Fraction digits in float numbers', 'range':(0, 10), 'default':4},
{'name':'disabled', 'title':'Disabled', 't':'%b', 'edit':RW, 'tt':'Disable/enable portfolio (stop\nrobot and disable timetable,\ncancel visual price updates from\nremote robot if checked)', 'default':False},
{'name':'log_level', 'title':'Log level', 't':'%comboc', 'combolist':[{'v':1, 't':'Order state', 'tt':'Log state of orders'}, {'v':2, 't':'Option info', 'tt':'Log information about options'}, {'v':4, 't':'Price info', 'tt':'Log all price updates'}, {'v':8, 't':'Adding order reason', 'tt':'Log reasons for adding/not adding orders'}, {'v':16, 't':'Pos info', 'tt':'Log position information (real and theoretical)'}, {'v':32, 't':'Limits info', 'tt':'Log information about moving limits'}], 'edit':RW, 'tt':'Log level', 'default':0},
{'name':'portfolio_num', 'title':'Orders ID', 't':'%combo', 'edit':RW, 'combolist':[{'v':i - ord('a'), 't':chr(i)} for i in range(ord('a'), ord('z') + 1)], 'tt':'Orders ID, it is recommended to be unique', 'default':0},
{'name':'all_free', 'title':'all_free', 't':'%b', 'edit':R, 'default':True, 'hidden':True},

It is strongly recommended NOT to remove re_sell, re_buy and pos from server side, because they are used by buttons in toolbar

f_disable example:
{'name':'sell', 'title':'Sell', 't':'%f', 'edit':R, 'tt':'Sell price', 'colorfill':sellColorfill, 'range':(-1e9, 1e9), 'default':0, 'f_disable': lambda r: r.get('lim_s', 0) < 0},
disable 'sell' if 'lim_s' is negative, you have access ONLY TO CURRENT TABLE ROW
"""

```
MAIN_TABLE_ = [
        {'name':'sell', 'title':'Sell', 't':'%f', 'edit':R, 'tt':'Sell price', '
            colorfill':sellColorfill, 'range':(-1e9, 1e9), 'default':0},
        {'name':'buy', 'title':'Buy', 't':'%f', 'edit':R, 'tt':'Buy price', '
            colorfill':buyColorfill, 'range':(-1e9, 1e9), 'default':0},
        {'name':'lim_s', 'title':'Lim_Sell', 't':'%f', 'edit':RW, 'tt':'Limit sell',
            'colorfill':sellColorfill, 'range':(-1e9, 1e9), 'default':0},
        {'name':'lim_b', 'title':'Lim_Buy', 't':'%f', 'edit':RW, 'tt':'Limit buy', '
            colorfill':buyColorfill, 'range':(-1e9, 1e9), 'default':0},
        ]

def byName_(name, default=None, base=None):
        if base is None:
                global MAIN_TABLE_
                base = MAIN_TABLE_
        for i, v in enumerate(base):
```

```
                if v['name'] == name:
                        res = {key:val for key, val in v.items()}
                        if 'default' not in res.keys() and not (default is None):
                                res['default'] = default
                        return res
        return None
```

"""ALL fields from ADD_PORTFOLIO_ table will be saved on app close, loaded on app start and can be edited in portfolio editor

*byName function gets rows with specified name from MAIN_TABLE_*

Predefined fields are:
{'name':'name', 'title':'Name', 't':'%s', 'edit':R, 'tt':'Portfolio name, should starts from latin letter\nand can only consist of latin letters and numbers.\nIt should differs from any contract key', 'default':''},
byName('log_level', 0),
byName('decimals', 4),
byName('portfolio_num', 0),
"""

```
ADD_PORTFOLIO_ = [
        byName_('lim_s', 0),
        byName_('lim_b', 0),
        ]
```

"""ALL fields from ADD_PORTFOLIO_TABLE_ table will be saved on app close, loaded on app start and can be edited in portfolio editor's securities table

Predefined fields are:
{'name':'sec_key', 'title':'SecKey', 't':'%s', 'tt':'Unique contract key', 'default':''},
{'name':'sec_type', 'title':u'SecType', 't':'%combo', 'combolist':[{'v':i[1], 't':i[0]} for i in SecType.values()], 'tt':u'Contract type', 'default':SecType.FUT},
{'name':'pos', 'title':'Curpos', 't':'%d', 'range':(-1e9, 1e9), 'tt':'Current position on contract', 'edit':RW, 'default':0},
{'name':'is_first', 'title':'Is first', 't':'%b', 'tt':'Set main contract', 'unique':True, 'edit':RW, 'default':False},
{'name':'on_buy', 'title':'On buy', 't':'%combo', 'combolist':[{'v':1, 't':'Buy'}, {'v':2, 't':'Sell'}], 'tt':'Contract order direction on portfolio buy', 'edit':RW, 'default':1},
{'name':'count', 'title':'Count', 't':'%d', 'range':(0, 1e9), 'tt':'Volume of contract in one portfolio', 'edit':RW, 'default':1},
byName('decimals'),
{'name':'d_pg', 't':'%f', 'default':0, 'hidden':True},
byName('all_free', True),

It is strongly recommended NOT to remove sec_board and sec_code fields

It is strongly recommended NOT to remove is_first, on_buy and count from server side, because they are used in computation of portfolio position in GUI

```
f_disable example:
{'name':'sec_board', 'title':'SecBoard', 't':'%s', 'tt':'Contract board', 'default
    ':'', 'f_disable': lambda r: r.get('k', 0) < 0},
disable 'sec_board' if 'k' is negative, you have access ONLY TO CURRENT TABLE ROW (
   not to all portfolio if it is ADD_PORTFOLIO_TABLE_)
"""
ADD_PORTFOLIO_TABLE_ = [
        {'name':'sec_board', 'title':'SecBoard', 't':'%s', 'tt':'Contract board', '
           default':''},
        {'name':'sec_code', 'title':'SecCode', 't':'%s', 'tt':'Contract code', '
           default':''},
        {'name':'k', 'title':'k', 't':'%f', 'range':(-1e9, 1e9), 'tt':'Price indent
           of order', 'edit':RW, 'default':0},
        ]

"""Check some fields values before value apply
Example (you can uncomment this row in FILTERS_ list):
{'name':'lim_s', 'title':'Limit is negative', 't':'%b', 'formula':lambda r: r.get('
   lim_s', 0) >= 0, 'fields':['lim_s', ]},
this filter asks question when 'lim_s' value is negative
where:
        name - unique key ('pos_ok' name is reserved)
        title - warning message
        t - type, always '%b'
        formula - function which returns True if filter is passed and False if it is
           failed
        fileds - list of fileds affected by this filter
"""
FILTERS_ = [
        #{'name':'lim_s', 'title':'Limit is negative', 't':'%b', 'formula':lambda r:
           r.get('lim_s', 0) >= 0, 'fields':['lim_s', ]},
        ]
```