Visual JF

Visual JForex

Руководство пользователя

Оглавление

Уведомление

Данное руководство предназначено для точного описания и толкования платформы Visual JForex. Разработчики Visual JForex стремятся обновлять руководство как можно чаще, чтобы оно охватывало нововведения в визуальном конструкторе.

Несмотря на то, что руководство обновляется регулярно, пользователи должны быть полностью осведомлены о том, что оно может не содержать описания всех изначальных процессов и функций, измененных со временем, а также других функций, добавляемых, удаляемых или изменяемых в процессе разработки. Поэтому некоторые описания, объяснения и другие уведомления могут быть устаревшими и, как следствие, не полностью совпадать или отражать текущие процессы и процедуры.

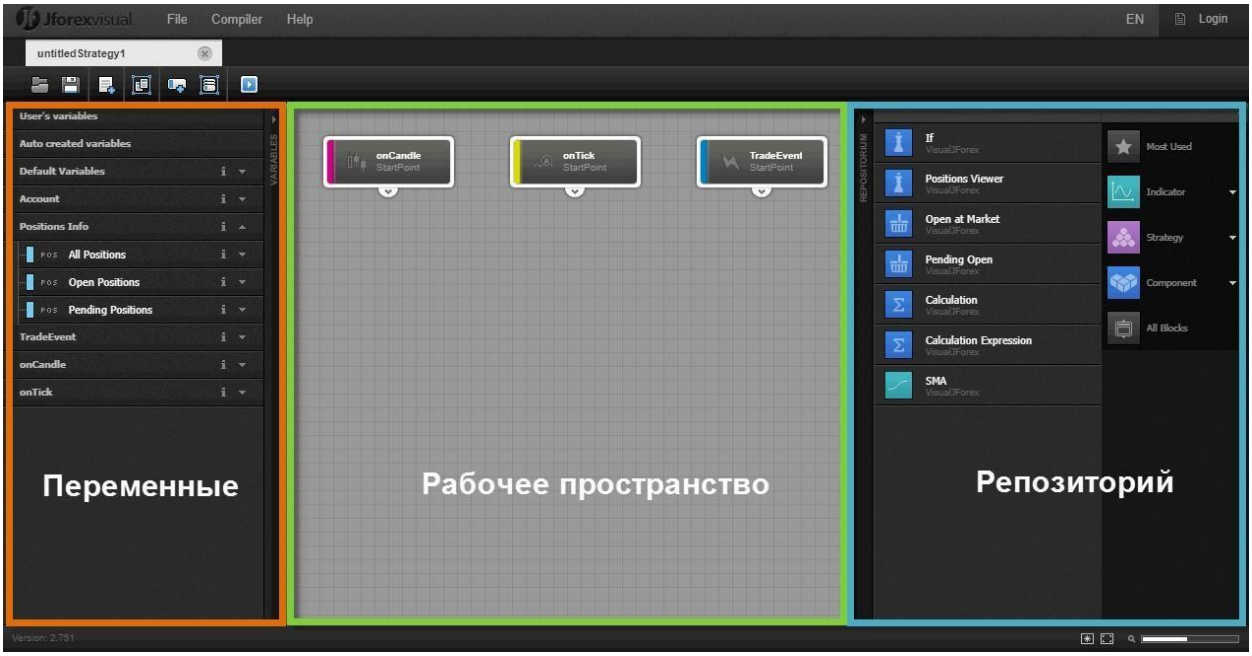
Поэтому клиентам/пользователям настоятельно рекомендуется обращаться к этому документу во время испытаний функций платформы на демо-счетах. Условия торговли, характеристики и функционал платформы, а также различные сопутствующие вопросы также должны быть проверены клиентами/пользователями. Все возможные оставшиеся вопросы должны быть разъяснены у представителей Dukascopy до того, как клиент/пользователь начнет управлять настоящим капиталом на реальном торговом счету.

Dukascopy не несет ответственности за любой ущерб, включая, но не ограничиваясь, убытками или потерями прибыли от использования, произошедшими вследствие неполной, неточной и (или) устаревшей информации, представленной в руководстве пользователя. Также данное руководство не должно толковаться как торговый совет, инвестиционный совет или непосредственные рекомендации торговать или не торговать по конкретным стратегиям на платформах Dukascopy.

Данное руководство пользователя не предназначено для освещения каких бы то ни было аспектов деловых отношений между Dukascopy с одной стороны и потенциальными и действующими клиентами и (или) другими партнерами с другой стороны. Для получения информации о политике Dukascopy по работе на рынках, по работе с клиентами, потенциальными клиентами и другими партнерами читателю настоящего руководства рекомендуется посетить веб-сайт Dukascopy и (или) связаться непосредственно с представителями Dukascopy.

Что такое Visual JForex?

Visual JForex представляет собой комплексное решение для разработки, тестирования и применения торговых стратегий на основе принципа drag-and-drop (перетаскивание мышью). Платформа дает уникальные возможности разработки полностью автоматизированных стратегий. Пользователь компонует стратегию из блоков-операторов, соединяемых между собой в визуальном конструкторе. На выходе пользователь получает полностью готовую к использованию форекс-стратегию. Visual JForex является браузерной платформой, созданной в Java и использующей flash-технологии.



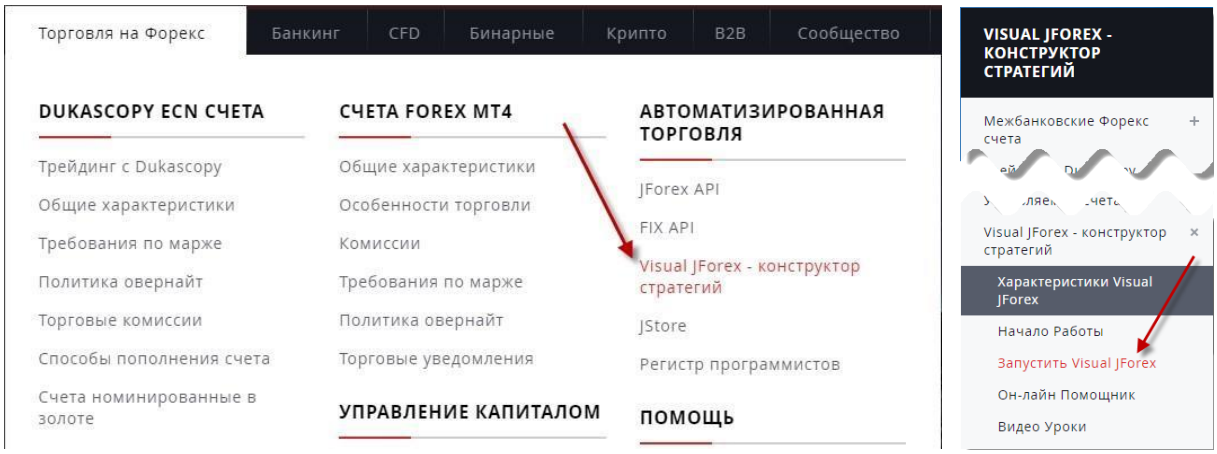
Требования Visual JForex к ПО и оборудованию:

- Операционные системы: Windows, Linux, Apple OS x.
- Интернет-браузер: Mozilla Firefox, Google Chrome, MS Explorer или Safari.
- Java 1.7 и выше.
- Flash Player.

1. Запуск платформы Visual JForex

1.1. Вход в Visual JForex

Для запуска платформы Visual JForex перейдите на сайт www.dukascopy.com и кликните "Торговля на Форекс" – "Visual JForex - конструктор стратегий" – "Запустить Visual JForex".



Большинство интернет-браузеров поддерживают встроенные flash-плагины. Мы рекомендуем использовать Mozilla Firefox, Google Chrome, MS Explorer (включая Edge) или Safari.

После запуска платформы пользователь может залогиниться по кнопке в правом верхнем углу. Для этого потребуются логин и пароль для входа в сообщество Dukascopy. Однако

пользоваться платформой можно и без логина. Учетную запись участника сообщества можно создать по ссылке.

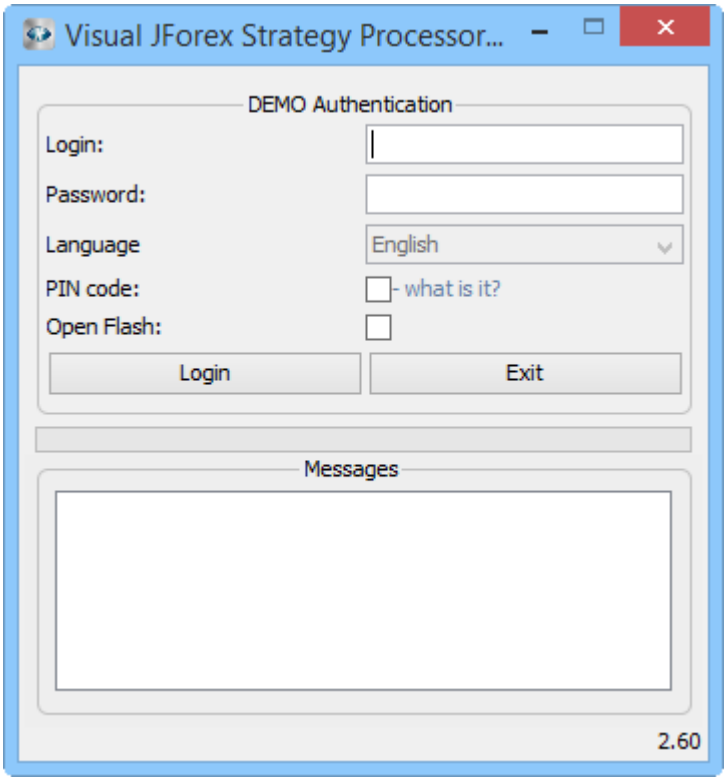
1.2. Запуск и вход в процессор стратегий

Visual JForex подключается к торговому счету с использованием моста, который связывает платформу с учетной записью JForex. Этот мост называется VJF-процессором и состоит из Java-процесса, который, в свою очередь, соединяет обе платформы (конструктор Visual JForex + торговая платформа JForex) и передает необходимые данные на историческое тестирование.

Процессор стратегий не может быть запущен при отсутствии загруженной или построенной стратегии.

После создания стратегии нажмите в меню "Compiler" ("Компилировать") и "Run" ("Выполнить") или нажмите кнопку , чтобы запустить процессор стратегий. При первом запуске платформа попытается подключиться к процессору стратегий и в итоге выдаст предупреждение о том, что процессор недоступен. Это нормально для первого запуска.

Нажмите "Download" ("Загрузить"), чтобы загрузить файл процессора стратегий. После чего будет загружен Java-файл и выполнен с использованием Java-программы (java webstart process), затем появится окно входа в систему.



Введите данные вашей демонстрационной учетной записи (демо-счета) JForex DEMO, чтобы установить соединение между конструктором Visual JForex и торговой учетной записью JForex.

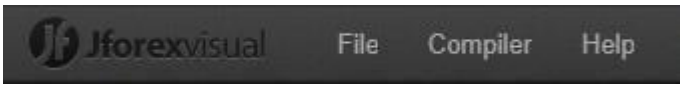
Это соединение позволяет платформе запускать стратегии как в режиме реального времени, так и в режиме исторического тестера.

Процесс подключения выполняется только один раз за сеанс. Процедуру нужно повторить, если процессор стратегий закрыт. В случае разъединения он автоматически подключится снова, а пользователь получит соответствующее сообщение.

Процессор стратегий также можно скачать напрямую по этой ссылке.

2. Меню, рабочее пространство и панель инструментов

2.1. Меню



2.1.1. Файл

Для создания новой стратегии пройдите в "File" > "New", в появившемся окне введите название стратегии. В итоге появится новая вкладка с названием стратегии.

Чтобы открыть ранее сохраненный файл, пройдите в "File" > "Open Draft". Для получения доступа к файлу необходимо быть подключенным к конструктору Visual JForex. Нажмите "Save Draft" ("Сохранить проект"), чтобы сохранить свою работу.

На серверах Dukascopy для хранения стратегий Visual JForex выделяется свободное пространство.

Для сохранения/загрузки стратегий в определенном формате файла, который может быть прочитан только из платформы Visual JForex, используются функции импорта/экспорта.
Данный файл имеет расширение ".vfs". Пользователь также может экспортировать файл стратегии в формате Java.

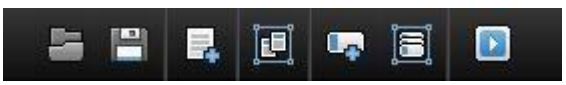
2.1.2. Компилятор

Меню компилятора позволяет запускать стратегию в режиме реального времени или в режиме исторического теста – теста на исторических котировках ("бэктест"). Также есть возможность экспорта исходного кода стратегии, то есть такая ее компиляция, которая выдает полный Javaкод, который можно использовать напрямую в платформе JForex. Для просмотра исходного кода в режиме реального времени воспользуйтесь опцией "View source" ("Просмотреть исходный код"). Опция "Contest" ("Конкурс") – это возможность отправить стратегию на конкурс стратегий Dukascopy.

2.1.3. Помощь

Меню "Help" ("Помощь") содержит ссылки на онлайн документацию и поддержку в чате.

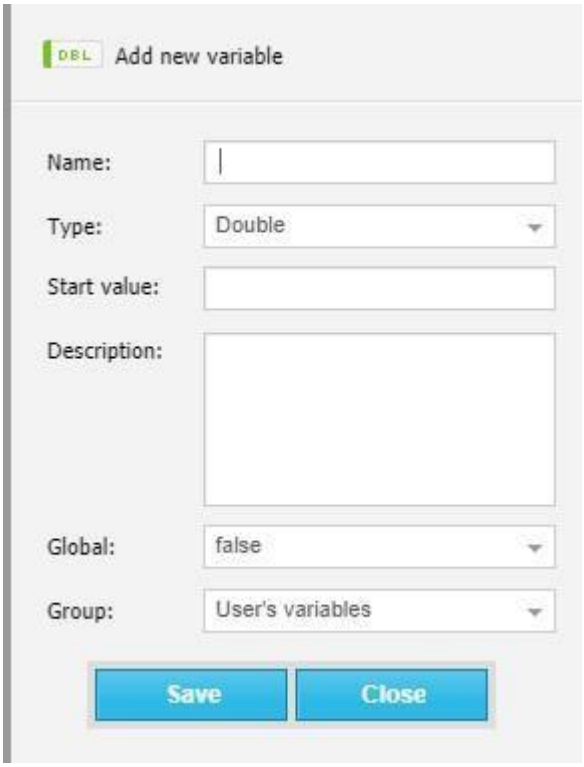
2.2. Панель инструментов



Панель инструментов содержит ярлыки для открытия, сохранения файлов в один клик, а также кнопку запуска стратегии. Имеются три дополнительные функции:

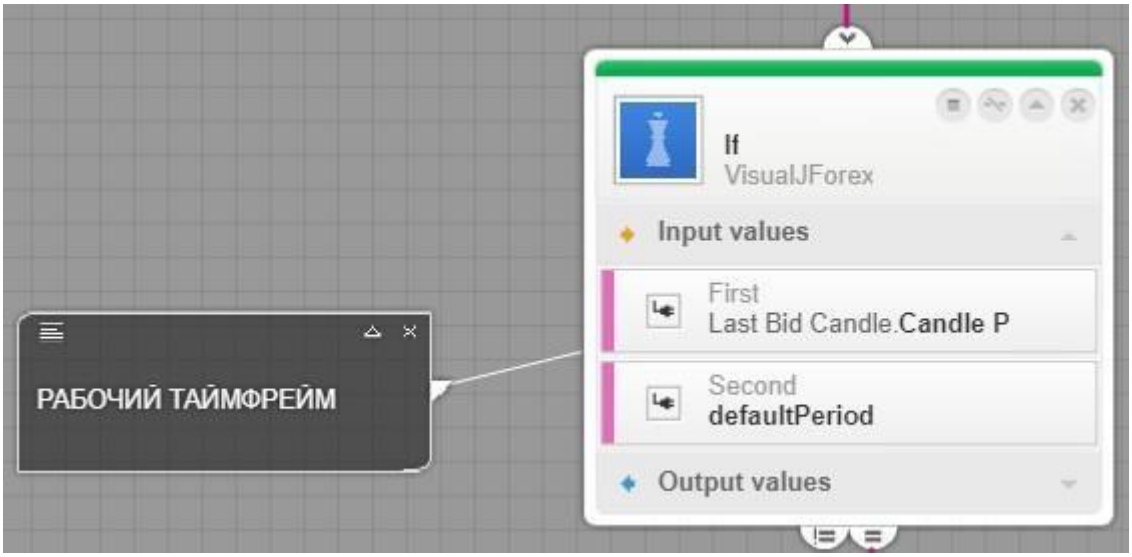
Create new group ("Создать новую группу"). Используется для добавления новой группы, в которую объединяются пользовательские переменные. Новая группа будет отображаться в виде подраздела в левой части экрана (в панели "Переменные" – "The variables").

Create new variable ("Создать новую переменную"). Нажмите эту кнопку, чтобы добавить новую переменную.

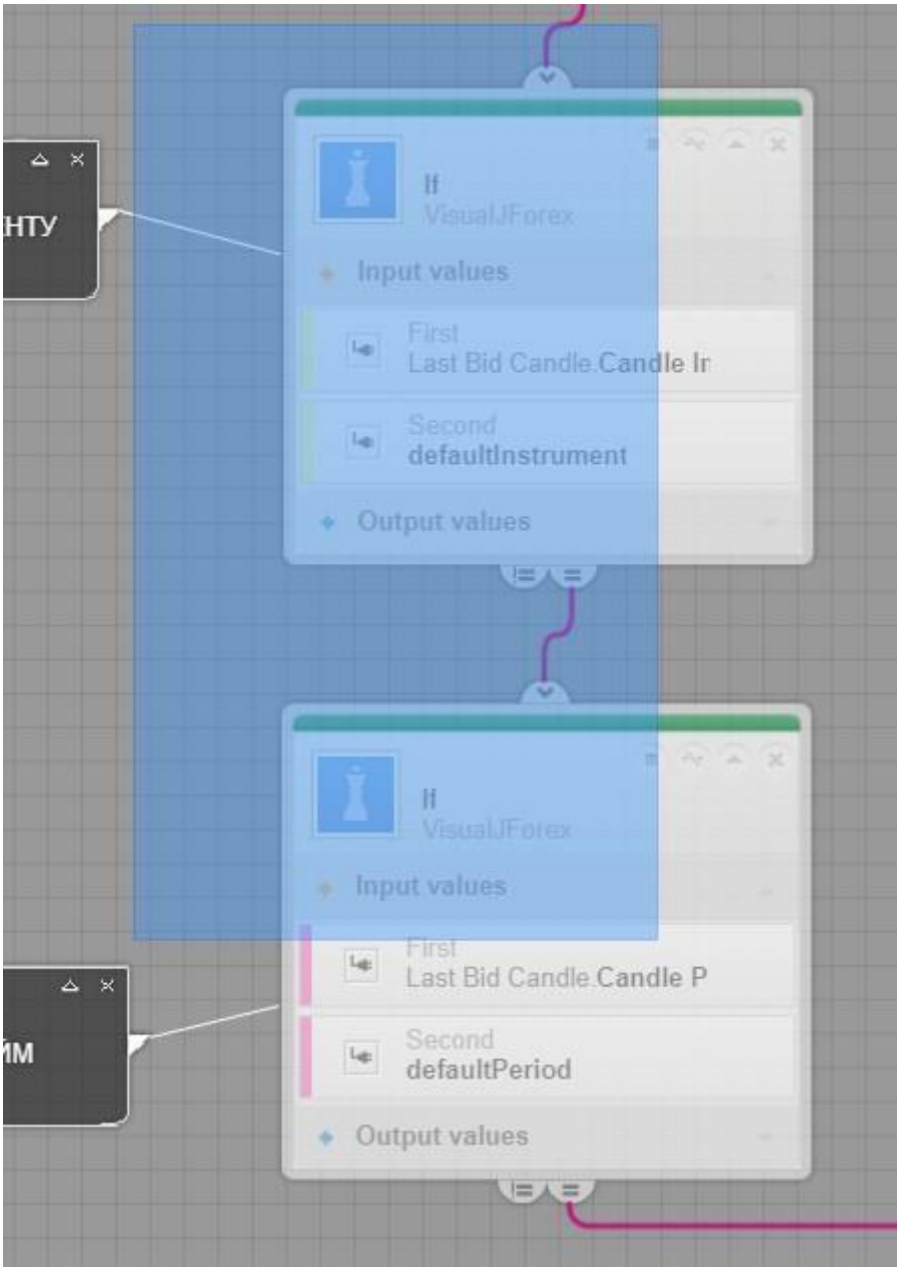


Появится новое окно, в котором нужно задать атрибуты переменной (см. раздел 6).

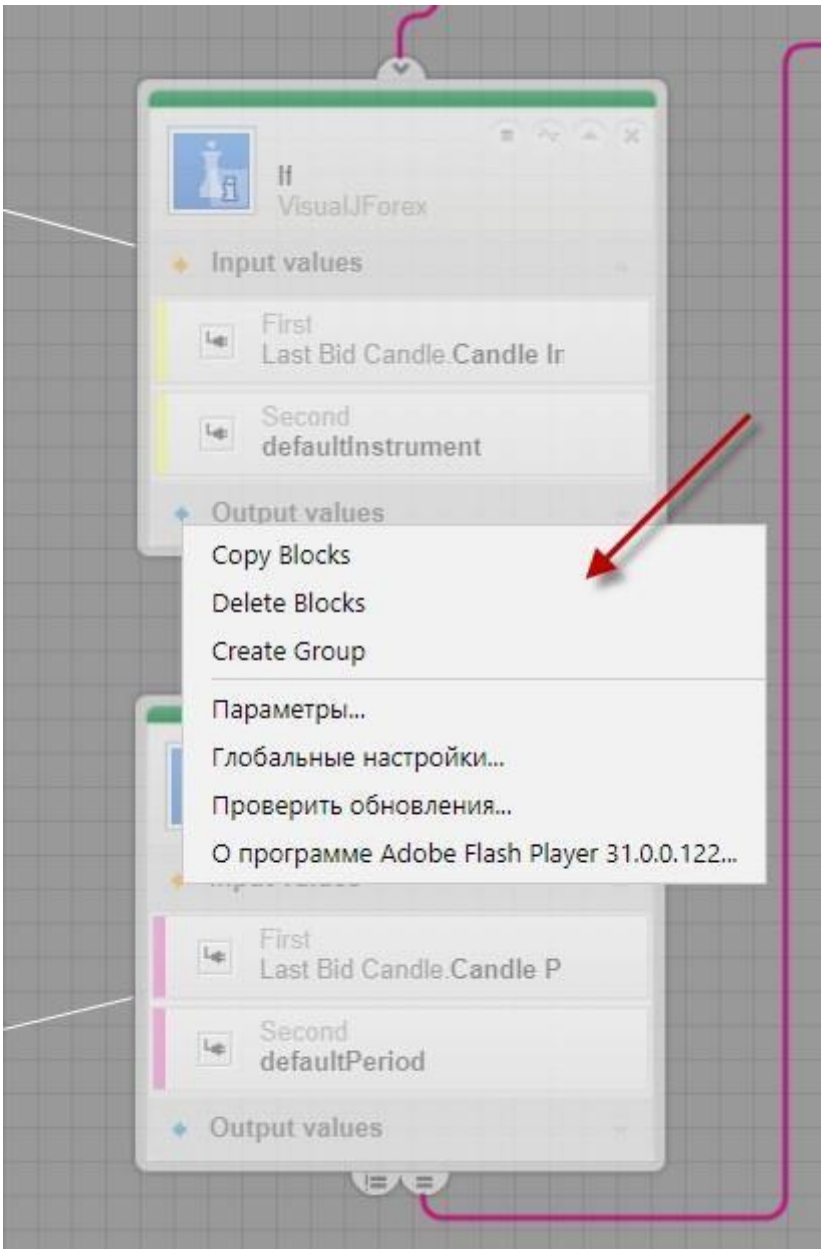
Add Note ("Вставить комментарий"). Нажмите эту кнопку, чтобы вставить комментарий в рабочую область; комментарий может быть связан с блоком или оставаться независимым от него.



2.3. Прочие опции



Нажмите Ctrl и выделите несколько блоков. Теперь можно перемещать набор выбранных блоков внутри рабочей области. Также при нажатии "Delete" ("удалить") на клавиатуре может быть выполнено удаление целой группы выбранных блоков.



Щелкните правой кнопкой мыши, чтобы отобразить контекстное меню, позволяющее скопировать группу блоков, удалить их или создать группу из выбранных блоков.

3. Стартовые точки

"Стартовые точки" – это три блока, отображающиеся по умолчанию в рабочей области при открытии платформы. Они представляют собой торговые методы, с которых должна начинаться любая стратегия. После активации от них линий логических связей (цепочек), связывающих их с другими блоками стратегии, эти стартовые точки активируются автоматически. При этом стартовая точка (точки), которая не активирована с помощью логических связей к другим блокам стратегии, остается неактивной, а поэтому ее метод не будет использоваться. Стартовые точки нельзя удалить из рабочей области.

Стратегия – в зависимости от ее логики и требований – может использовать одну либо две или три стартовые точки одновременно.

Всего стартовых точек 3 – onCandle, onTick и TradeEvent. См. рис.

- **onCandle ("По закрытию свечи").** Этот метод основан на таймфреймах свечей, и он автоматически обрабатывает периоды 10 сек, 1 мин, 5 мин, 10 мин, 15 мин, 30 мин, 1 час, 4 часа, 1 день, 1 неделя и 1 месяц. Любой из перечисленных таймфреймов можно выбрать из списка вариантов в типе переменной "Default Period" ("Таймфрейм по умолчанию", подробнее в разделе 6.2).

  Стартовая точка "onCandle" создана для работы по указанным выше таймфреймам, она активируется в конце каждого таймфрейма – после закрытия каждой свечи. При этом в режиме тестирования в процессоре стратегий вокруг блока мигает красная рамка. Другими словами, по умолчанию эта стартовая точка активируется в конце каждого из перечисленных периодов, т.е. в процессоре стратегий она мигает каждые 10 секунд (минимальный период свечи по умолчанию), если пользователь не задал фильтрацию по периодам.

  Переменные из группы "onCandle" в панели переменных (раздел 6.1.7) относятся к этой стартовой точке, поэтому их значения обновляются динамически, когда стартовая точка активируется.

- **onTick ("По тикам").** Этот метод основан на тиковых ценах по выбранному инструменту. Тик – это минимальное изменение цены. По умолчанию метод "onTick" работает с EUR/USD, но это можно изменить в группе переменных "Default Variables" (раздел 6.1.3).

Группа переменных "onTick" также обновляется всякий раз, когда активируется стартовая точка "onTick". По аналогии с "onCandle" в процессоре стратегий стартовая точка "onTick" также подсвечивается красной рамкой в момент активации.

- **TradeEvent ("Торговое событие").** Этот метод реагирует на сообщения, полученные торговой платформой, и активируется каждый раз при получении определенного сообщения.

4. Блок

Блок – это элемент конструктора, который используется для выполнения логических и математических функций, вставки индикаторов либо для выполнения торговых команд. Каждый блок обладает уникальными функциями, например: выполнение логического сравнения, выполнение торговой команды, получение значений индикатора, выполнение вычислений.

1. **Вход.** Это точка входа в блок, соединяющая его с другими блоками.
2. **Индикатор состояния.** Если индикация зеленая, блок готов к исполнению, если красная, то требуется заполнить блок входными и/или выходными данными либо создать соответствующие переменные.
3. **Панель инструментов** (слева направо).

a. *Breakpoint ("Точка останова").* Данная опция останавливает стратегию по достижении этого блока. В основном она используется для устранения неполадок, чтобы остановить стратегию и проверить функции и значения переменных. После этого работа стратегии может быть возобновлена.
b. *Remove all connections ("Отсоединить блок").* Удаление всех связей, идущих к и от данного блока, одним кликом мыши.
c. *Minimize ("Свернуть").* Свернуть блок, чтобы уменьшить его размер. Служит для улучшения эргономики рабочего пространства.
d. *Remove block ("Удалить").* Удалить блок из рабочего пространства.

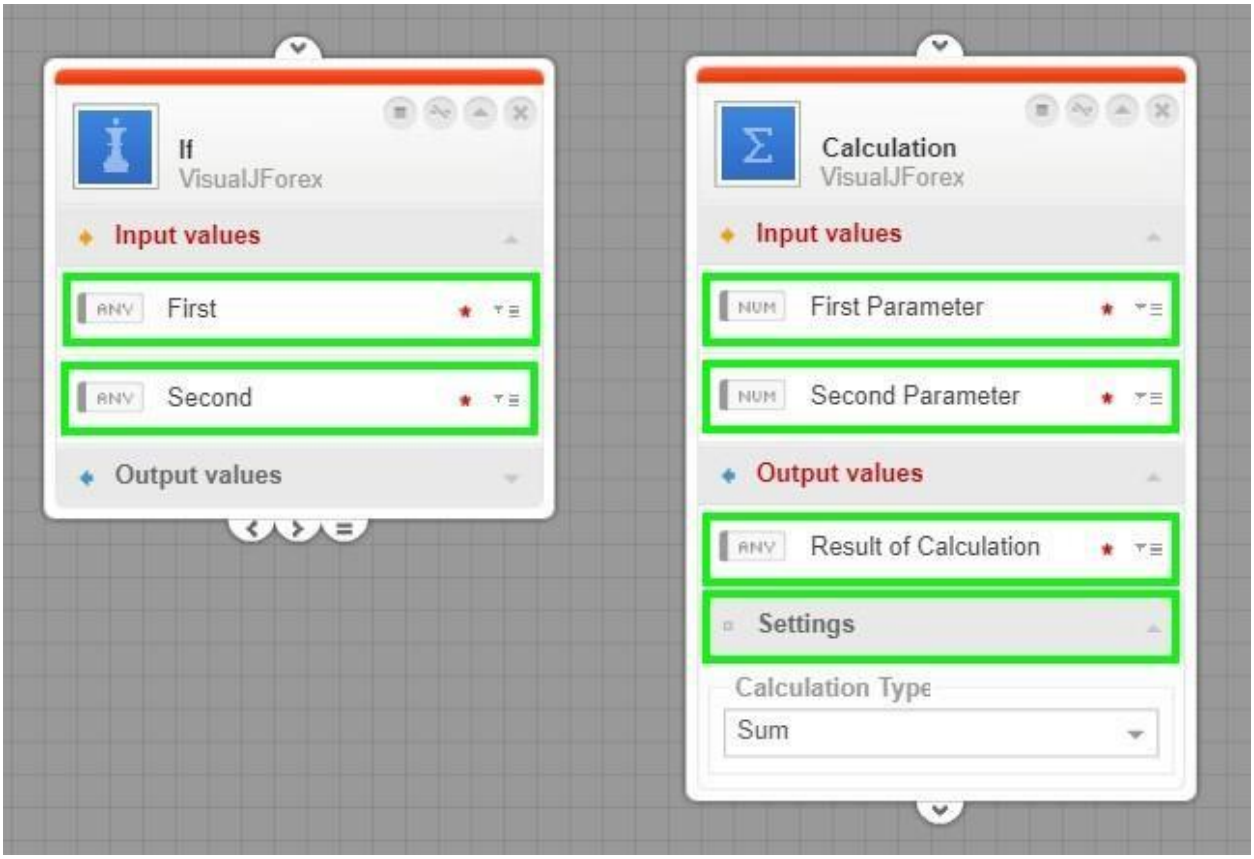
4. **Название блока.**
5. **Торговая команда.** Элемент имеется в торговых блоках, содержит доступные типы ордеров.
6. **Входные значения ("Input values").** Переменные, используемые в качестве входных данных для выполнения функции блока.
7. **Быстрый доступ к переменным.** Просмотр переменных, подходящих на данное место в блоке.
8. **Выходные данные ("Output values").** Поле для вставки выходной переменной или ее создания вручную.
9. **Настройки ("Settings").** Список параметров блока.
10. **Выход(ы).** Точки выхода из блока, здесь выведены выходные соединения. Блок может иметь как одну, так и несколько выходных связей. Для работы блок должен быть подключен к другим блокам стратегии.

Выходы синего цвета относятся к сообщениям о статусе ордера, которые также используются для стартовой точки "TradeEvent". Выходы серого цвета используются в логике стратегии для связывания блока с другими блоками, поэтому они не относятся к какому-либо статусу ордера или к подтверждающему сообщению.

Чтобы выполнить свою функцию, блок должен быть подключен на входе и на выходе к другим блокам, иначе система проигнорирует этот блок.

Блоки организованы в категории и находятся в репозитории, см. раздел 5.

Блок всегда имеет входные и выходные данные, которые должны быть заполнены. Для некоторых типов блоков доступна панель настроек.



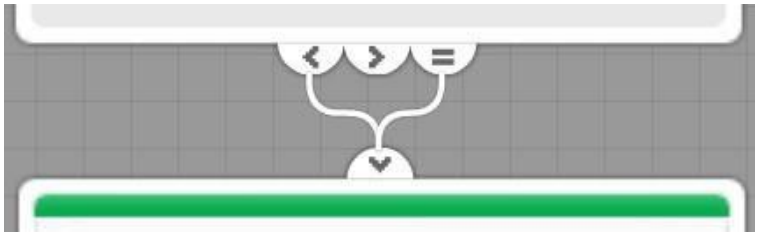
Для выполнения условия или операции, вызываемой блоком, нужно заполнить входные параметры. Выходная переменная также является обязательной. Она доступна по умолчанию или создается вручную.

Когда блок подключен к выходу с другим блоком, он становится активным, и тогда система сможет пройти через него для выполнения запрошенной команды.

Выход 1. Первая переменная меньше второй.

Выход 2. Первая переменная больше второй.

Выход 3. Первая переменная равна второй.



При наведении мыши на какой-либо выход блока или его область можно получить всплывающую подсказку.

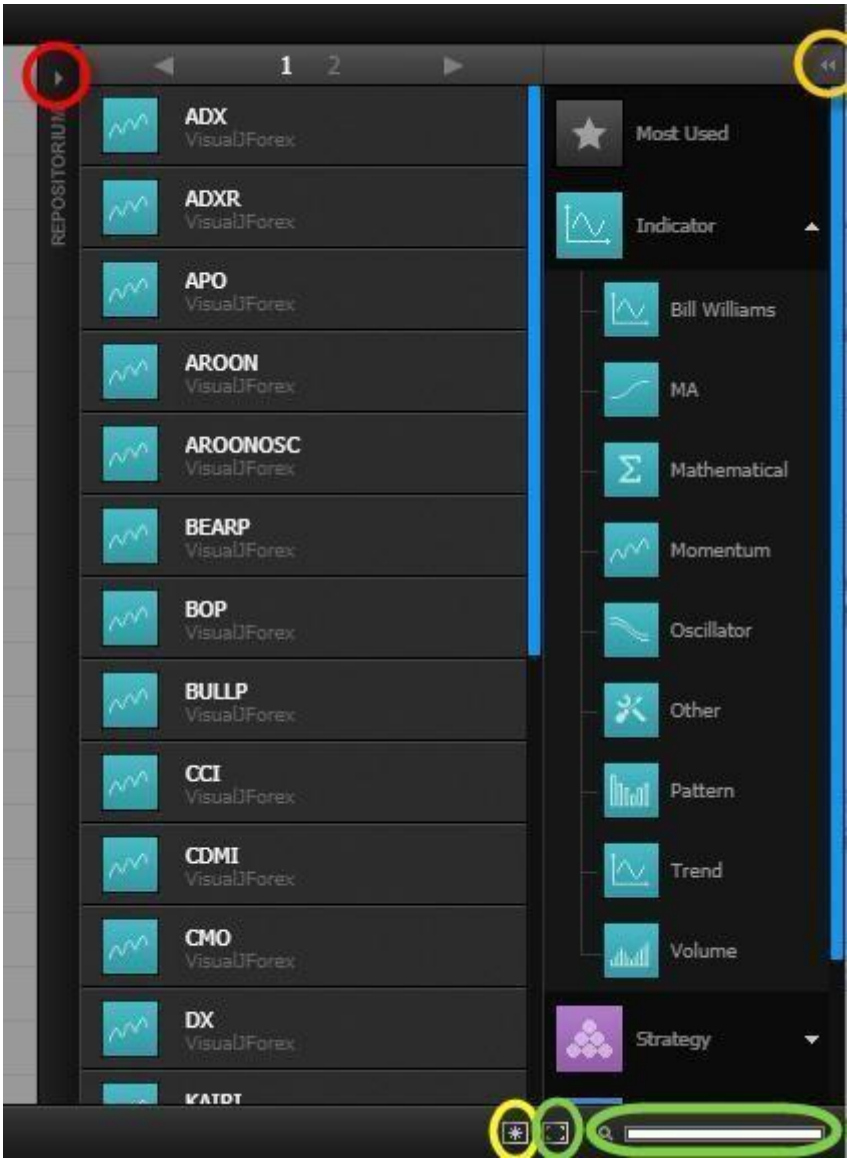
Торговую логику можно продолжить из нескольких выходов блока.

5. Репозиторий

Репозиторий – это сворачивающаяся панель в правой стороне рабочего экрана. Она содержит функции и компоненты. Данные элементы отображаются в виде значков, которые можно захватывать и перетаскивать в рабочую область в виде блока. Репозиторий состоит из трех основных категорий:

- Список индикаторов по типам.
- Стратегии, в том числе стратегии пользователя в подгруппе "My Strategies".
- Компоненты – здесь хранится весь перечень блоков, разбитый по категориям.

Панель "Репозиторий" сворачивается по нажатии на кнопку, обведенную красным кругом на рис. Если кликнуть на кнопку, обведенную оранжевым, список групп компонентов будет сокращен до пиктограмм.



Секция индикаторов содержит готовые к использованию индикаторы, которые для использования также перетаскиваются в рабочее пространство.

Функция "Navigation" (значок "Навигация" – обведен желтым) позволяет отобразить минисхему рабочей области (превью). Это замечательный инструмент для поиска блоков и разделов стратегии, особенно для сложных стратегий, требующих большого количества блоков и, соответственно, места под них на рабочем пространстве. Кнопка "Zoom" (масштабирование) и масштаб (обведены зеленым) позволяют настроить масштабирование. Увеличение и уменьшение масштаба производится с помощью колеса прокрутки мыши, а также с помощью шкалы в правой нижней части экрана.

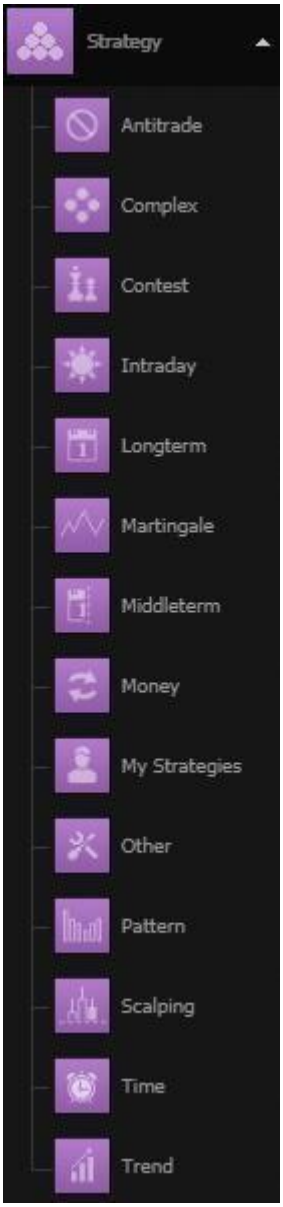
5.1. Индикаторы

Visual JForex предлагает большой выбор индикаторов, разделенных на 9 категорий: индикаторы Билла Уильямса ("Bill Williams"); скользящие средние ("MA"); математические индикаторы ("Mathematical"); индикаторы момента, или импульса ("Momentum"); осцилляторы ("Oscillator"); индикаторы паттернов ("Pattern"); трендовые индикаторы ("Trend"); индикаторы объема ("Volume"), а также прочие индикаторы ("Other").

Каждая категория имеет свой набор индикаторов, которые отображаются с коротким названием. Полное название индикатора отображается при нажатии на значок "i" ("информация") левее наименования инструмента в верхнем левом углу блока. Если это описание недоступно, можно отыскать информацию об индикаторе напрямую в библиотеке платформы JForex.

Почти все индикаторы содержат раздел настроек с двумя параметрами: "OfferSide" (сторона котирования), в которой пользователь выбирает, по какой цене производить расчет – спроса (цена "бид" – bid price) или предложения (цена "аск" – ask price). Аск отображает котировки, применяемые для открытия длинной позиции (long, или buy) и закрытия короткой позиции (short, или sell). Второй параметр – это метод, применяемый для расчета индикатора и показанный как "AppliedPrice" (используемая цена). Доступны несколько методов использования цены: простые (одна из четырех цен – открытия, максимума, минимума, закрытия) и агрегированные (средняя, типичная, взвешенная цена и т.п.).

5.2. Стратегия



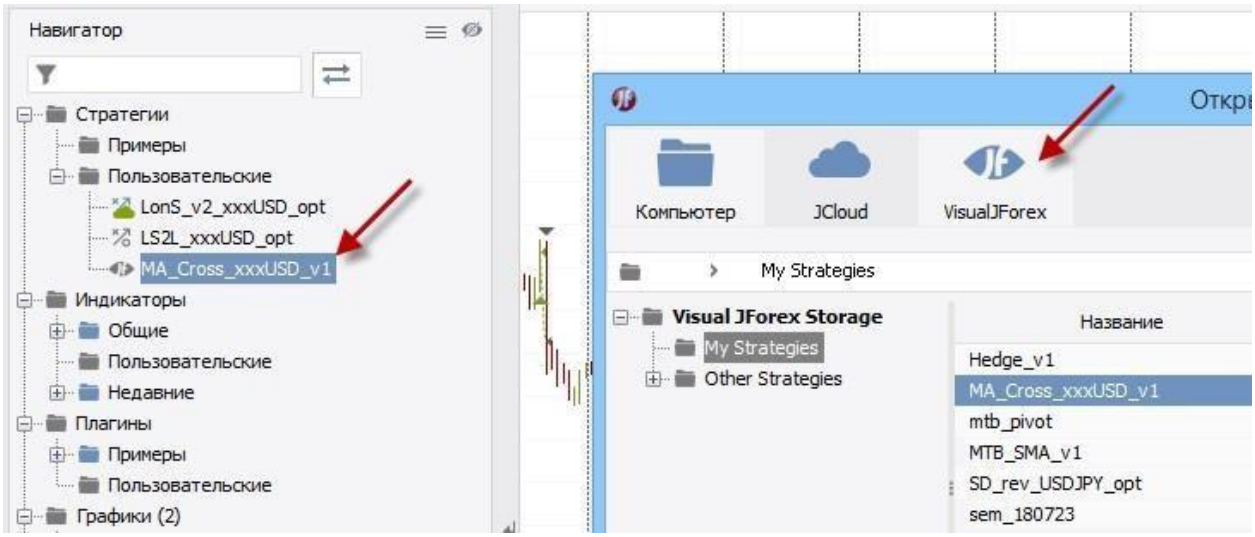
Раздел " Strategy" ("Стратегия") имеет фиолетовые пиктограммы и содержит группы стратегий, объединенных общим торговым подходом.

В данном разделе стратегии делятся на контртрендовые (" Antitrade"), комплексные (" Complex"), конкурсные (" Contest"), внутридневные ("Intraday") и т.д. Все они общедоступны. Только раздел "My Strategies" ("Мои стратегии ") является закрытым и доступен только при входе соответствующего пользователя в систему.

Наиболее используемым элементом этого списка является раздел "Мои стратегии", где хранятся скомпилированные стратегии, которые могут быть загружены следующим образом: кликните правой кнопкой мыши по значку стратегии и выберите " Load VFS" ("Загрузить VFS"), чтобы открыть стратегию в рабочей области, или "View Source" ("Просмотреть исходный код") для отображения Java-кода во всплывающем окне. Из того же контекстного мен ю стратегию можно удалить.

Когда стратегия скомпилирована, она сохраняется в разделе " My Strategies" и может быть загружена с платформы JForex, а также с мобильного приложения SWFX Trader.

Стратегии доступны под вкладкой " Visual JForex", и при загрузке е е можно идентифицировать по ее пиктограмме в списке стратегий (например, MA_Cross_xxxUSD_v1).

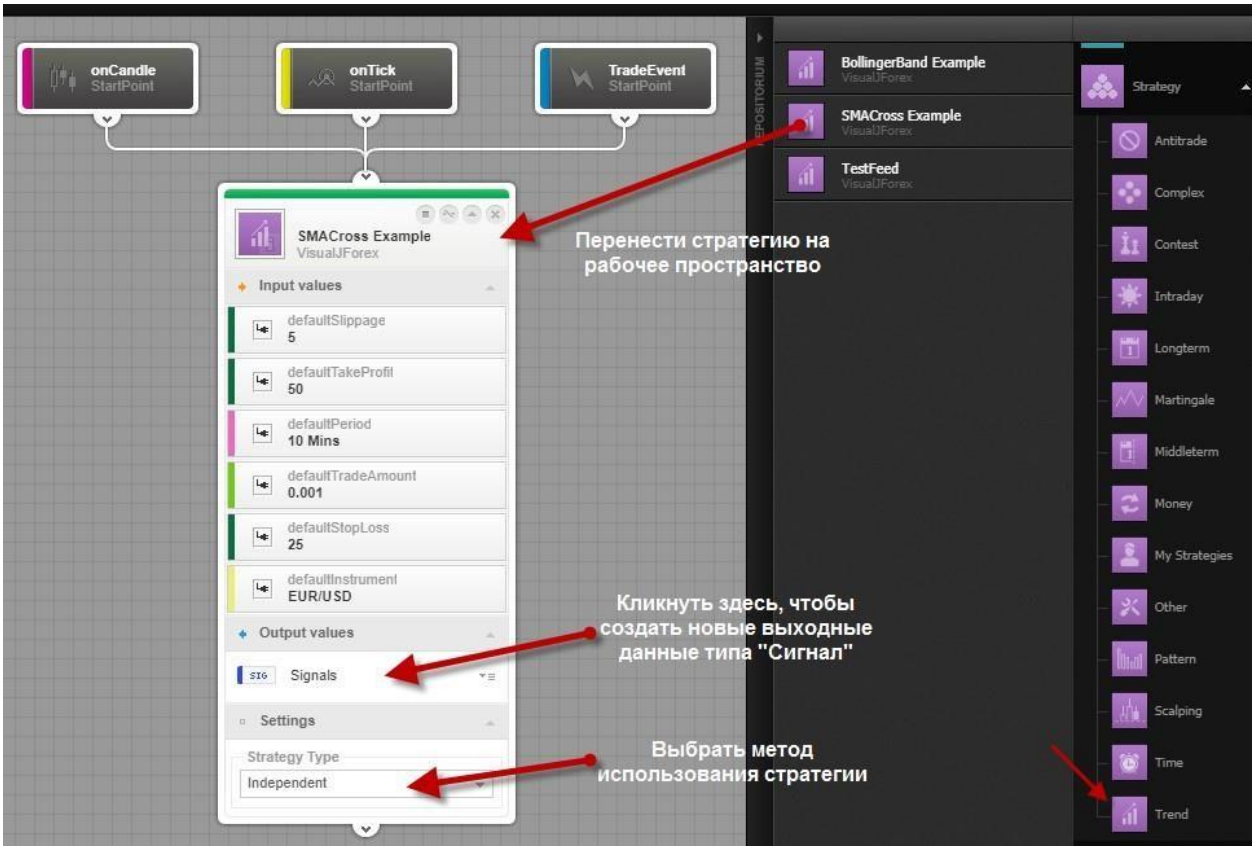


5.2.1. Использование стратегии как блока

Это упрощенный способ оптимизации рабочего пространства и отображения всей стратегии в виде единого блока для генерации сигналов или даже для нормальной работы. Сигналом является захват торговых инструкций, инициированных стратегией, а также информации, связанной с позициями и созданными ордерами.

Такой подход рационален, когда речь идет о запуске готовых стратегий с использованием их торговых сигналов и добавлением дополнительных условий и улучшений. Также облегчается изменение параметров стратегии одним щелчком, поскольку они доступны в качестве входных переменных в блоке стратегии.

Чтобы использовать стратегию в качестве блока и генерировать сигналы, просто захватите и перетащите иконку с названием стратегии в рабочую область и создайте новую выходную переменную типа "Signals" ("Сигналы"). Затем выберите параметр "Generate only Signals" ("Только генерировать сигналы") в настройках блока.



Чтобы стратегия-блок работала корректно, необходимо подключить ее к каждой стартовой точке (onCandle, onTick, TradeEvent). Выходная переменная относится к типу переменных "Signals" ("Сигналы", см. раздел 6.2.2). Это массив данных, содержащих информацию, связанную с этим сигналом. Такая переменная является "закрытым" массивом, чтобы его "раскрыть", нужно использовать блок "Loop Viewer" ("Цикл"). Для этого добавьте на рабочее пространство блок цикла "Loop Viewer" и в нем в качестве входной переменной используйте

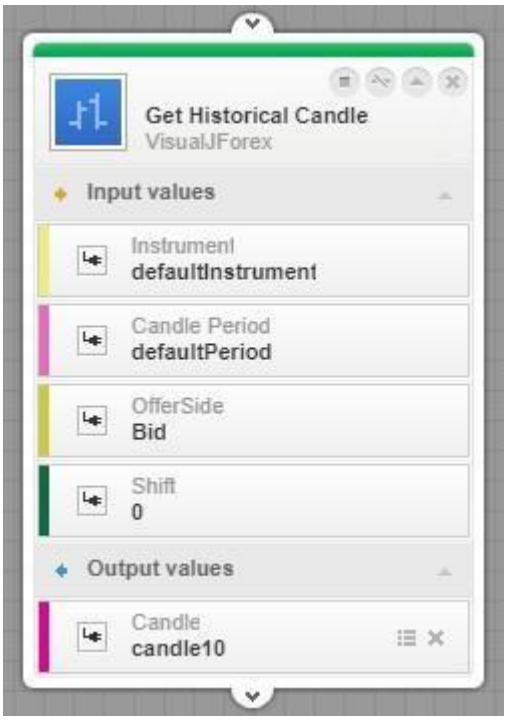
выходную переменную типа "Signals" из блока стратегии. Далее в блоке цикла создайте новую выходную переменную типа "Signal", она "вскроет" выходную переменную "Signals" из блока стратегии и будет содержать информацию о сигнале (подробнее о работе циклов см. раздел 7.6).

При запуске и выполнении стратегии в качестве блока пользователь может выбрать один из двух режимов: "Независимый" режим ("Independent") либо режим "Только генерировать сигналы" ("Generate only Signals"). Первый режим ("Independent") аналогичен нормальному запуску стратегии со всеми ее блоками и функционалом. Второй режим ("Generate only Signals") генерирует только торговые сигналы, но не отдает торговые команды. Эти сигналы являются как бы фоновым уведомлением о торговой команде. Пользователь может использовать данные сигналы в качестве руководства к действию (триггера) при дальнейшем возникновении соответствующих торговых возможностей.

5.3. Компоненты

Последняя секция репозитория с пиктограммами синего цвета содержит компоненты, разбитые по категориям. "Компоненты" – это функциональные и технические блоки, используемые для построения логических условий и выражений, а также математических и торговых команд и операций.

5.3.1. Информационные – Info



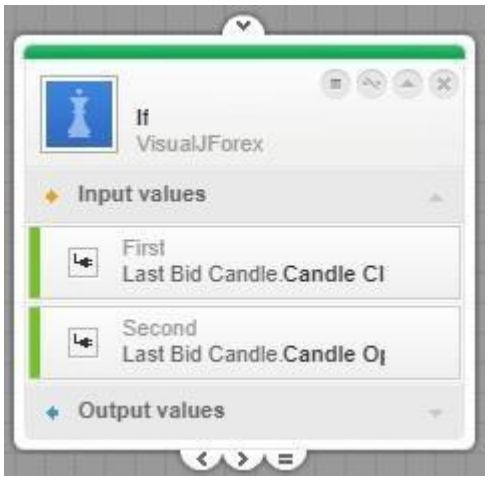
Get Historical Candle ("Получить бар из истории"). Позволяет получить информацию, связанную с текущей и еще не закрывшейся свечой или же информацию по предыдущим закрывшимся свечам. Порядковый номер бара в истории (справа налево) задается в переменной "Shift" ("Смещение"). По умолчанию он равен 0, т.е. относится к текущей, не закрывшейся свече. Пользователь может задать инструмент, таймфрейм свечи, сторону котирования. Выходные данные сохраняются в виде массива, который создается автоматически и имеет тип "Candle" ("Свеча") (подробнее в разделе 6.2.2).

Полученные выходные данные будут доступны в группе переменных "Автосозданные переменные" ("Auto created variables"). В них входят следующие: цены открытия, максимума, минимума и закрытия (Open, High, Low, Close), а также объем, таймфрейм свечи и время начала ее формирования.

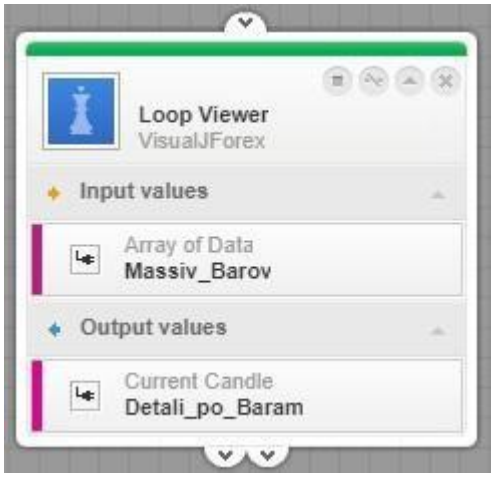
Get Historical Candles ("Получить бары из истории"). По аналогии с предыдущим блоком, который выдает информацию лишь об одной свече, этот блок позволяет извлекать данные о нескольких свечах. Их количество задается в переменной "Candles Amount" ("Количество свечей") – указанные свечи будут просмотрены на истории, а данные по ним сохранены. На выходе получается массив с информацией, которую можно "раскрыть", вставив его в блок "Loop Viewer".

Блок используется в случае, когда требуется получить данные по большому количеству свечей, что было бы неудобно выполнить с помощью множества отдельных блоков "Get Historical Candle".

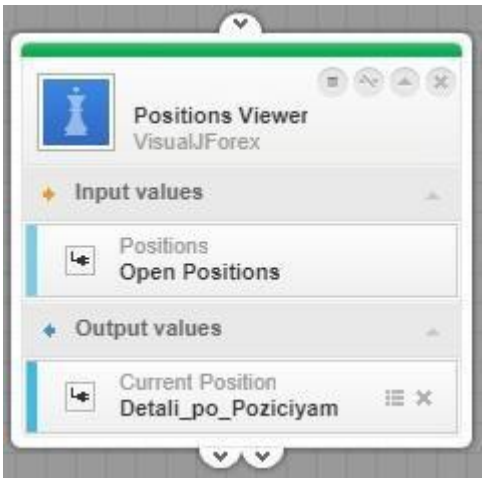
5.3.2. Логические – Logical



IF ("Если"). Широко используется в программировании для решения логических задач. Блок оперирует двумя переменными и поддерживает множество типов переменных. В случае несовпадения типов переменных блок "IF" выдает предупреждение и предлагает изменить тип переменных. Если в блоке используется переменная типа "Instrument", то блок автоматически меняет точки выхода с трех ("меньше", "больше" и "равно") на две ("равно" и "не равно"). На месте второй переменной появляется выпадающий список всех торгуемых инструментов. Это относится к переменным следующего типа: "Boolean" ("Булева"), "Instrument" ("Инструмент"), "String" ("Строка"), "Offer Side" ("Сторона котирования"), "State" ("Состояние") и "Command" ("Команда"). Подробнее см. в разделе 6.2.

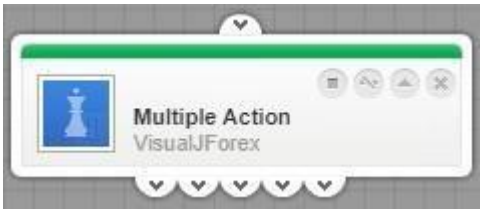


Loop Viewer ("Цикл"). Цикл – это основополагающее понятие в программировании. Этот блок выполняет цикл, или последовательность инструкций, которые непрерывно повторяются до тех пор, пока не будет выполнено определенное условие. Он в основном используется для "вскрытия" массивов переменных, имеющих тип "Candles" ("Свечи"), "Positions" ("Позиции"), "Signals" ("Сигналы"). При использовании в "Loop Viewer" выходной переменной из "Get Historical Candles" получается информация по набору запрошенных исторических свечей. При использовании стратегии как блока (см. раздел 5.5.2) выходная переменная типа "Signals" перетаскивается в "Loop Viewer" на место входной переменной, а результирующий массив сохраняется как выходная переменная блока "Loop Viewer". Если в качестве входной использовать переменную типа "Positions", то блок "Loop Viewer" выполнит такую же функцию, как и "Positions Viewer" ("Просмотрщик позиций"). Цикл состоит из блоков, выходящих из первой точки выхода блока "Loop Viewer" (слева), а после окончания цикла логика торговой стратегии продолжается из второй точки выхода блока (справа).



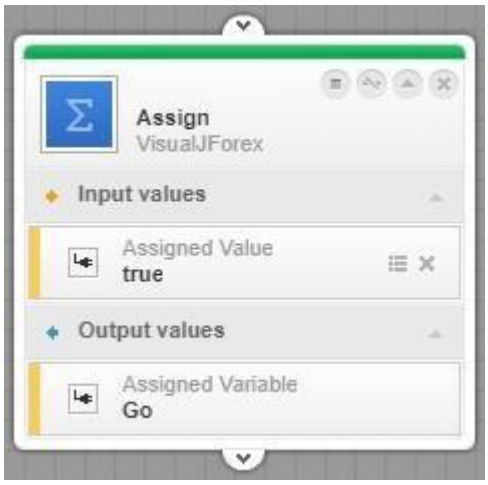
Positions Viewer ("Просмотрщик позиций"). Блок выполняет последовательность действий, которые повторяются до тех пор, пока не будет достигнуто заданное условие. Блок будет выполнять ту же работу, что и "Loop Viewer", только он предназначен для просмотра информации по позициям. Разница между "Loop Viewer" и "Positions Viewer" в том, что "Positions Viewer" предназначен для переменных типа "Positions" ("Позиции"). Как правило, входная переменная берется из группы "Positions Info" ("Позиции") в панели переменных.

Это дает возможность работать с позициями определенного состояния (открытыми позициями, отложенными ордерами или всеми вместе). Для "вскрытия" входной переменной нужно создать выходную переменную. Результат выполнения цикла просмотра позиций будет сохранен в этой новой переменной, которая будет помещена в группу "User's variables" ("Пользовательские переменные") в панели переменных.

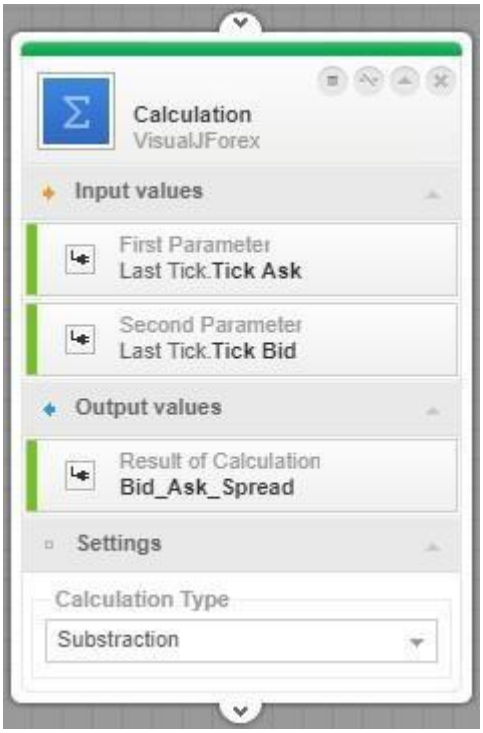


Multiple Action ("Множественное действие"). Это простой блок, который помогает разбивать основной поток на несколько потоков (до пяти отдельных выходов) по принципу: 1 вход – 5 выходов. Его можно повторить, чтобы еще раз увеличить разветвление.

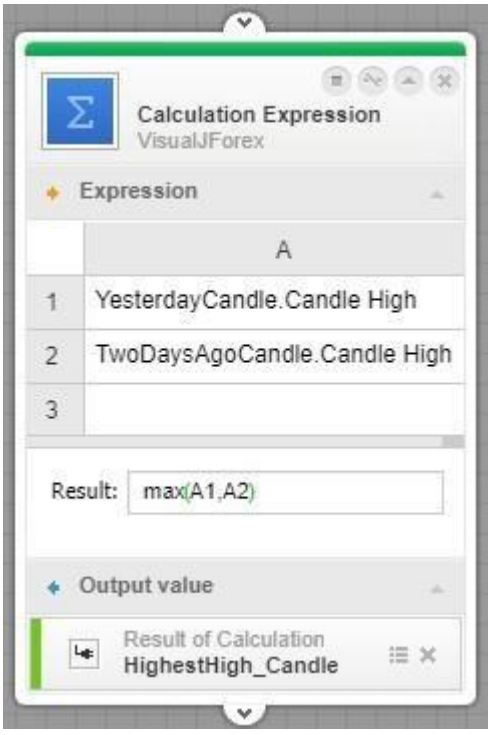
5.3.3. Математические – Mathematical



Assign ("Назначение"). Присвоить выбранное значение переменной с учетом ее типа. Когда вставляется входная переменная, для выходной автоматически подбирается соответствующий тип переменной.



Calculation ("Вычисление"). Относится к основным математическим вычислениям, таким как сложение, вычитание, умножение и деление в случаях, когда результат вычисления необходим для дальнейшего использования. Поддерживаемые типы переменных: "Double" (вещественный), "Integer" (целочисленный), "Date and Time" (дата и время). Подробнее см. 6.2.1.

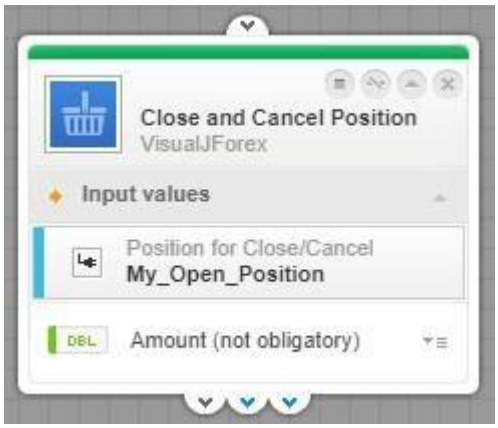


Calculation Expression ("Сложное вычисление"). Предназначен для сложных вычислений и математических функций. Этот блок имеет сходство с таблицей Excel. Пользователь может добавить строку или столбец. Для этого нужно кликнуть правой кнопкой мыши по заголовку строки или столбца. Поддерживаемые математические операции:

- "-" – вычитание.

- "+" – сложение.
- "/" – деление.
- "*" – умножение
- "power(X, Y)" – возведение X в степень Y.
- "%" – модуль.
- "min(X, Y)" – возвращает минимальное между X и Y.
- "max(X, Y)" – возвращает максимальное между X и Y.
- "sqrt(X)" – возвращает квадратный корень из X.
- "sin(X)" – возвращает синус X, при этом предполагается, что X выражается в радианах.
- "cos(X)" – возвращает косинус X, где X выражается в радианах.
- "tan(X)" – возвращает тангенс X, где X выражается в радианах.
- "ln(X)" – возвращает логарифм X по основанию эйлерова числа (e).
- "atan(X)" – возвращает угол в радианах между -π/2 и π/2, тангенс которого равен X.
- "acos(X)" – возвращает угол в радианах между 0 и π, косинус которого равен X.
- "asin(X)" – возвращает угол в радианах между -π/2 и π/2, синус которого равен X.
- "exp(X)" – возвращает эйлерово число (e), возведенное в степень X.

5.3.4. Торговые – Trading



Close and Cancel Position ("Закрыть и отменить позицию"). В зависимости от типа входной переменной этот блок либо закрывает открытые позиции, либо отменяет размещенный ордер. Чтобы выполнить частичное закрытие, во втором поле ("Amount" – объем) вводится объем позиции, подлежащий закрытию.

Выходы из блока:

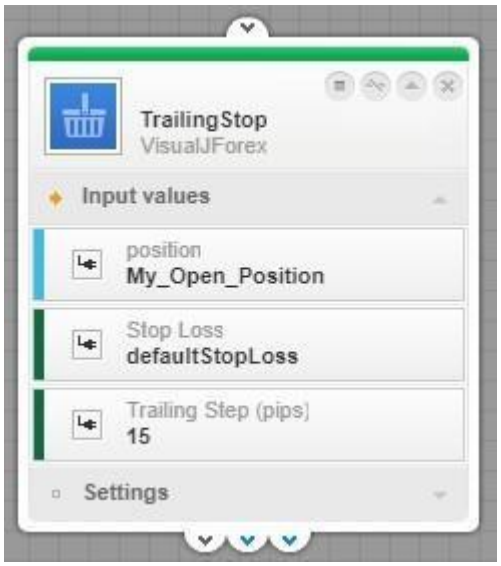
- Простой выход независимо от подтверждения ордера.
- Позиция закрыта или ордер отменен (статус OK).
- Отмена ордера отклонена или закрытие позиции отклонено.

Set Stop Loss ("Задать стоп-лосс"). Задает стоп-лосс для выбранной позиции. Позиция определяется входной переменной типа "Position" ("Позиция").

Уровень стоп-лосса задается либо в пунктах целым числом – тип переменной "Integer", либо конкретной котировкой – тип переменной "Double" (вещественная). Данный выбор выполняется в настройках блока ("Settings"). Также можно выбрать котировку реагирования (бид или аск).

Выходы из блока:

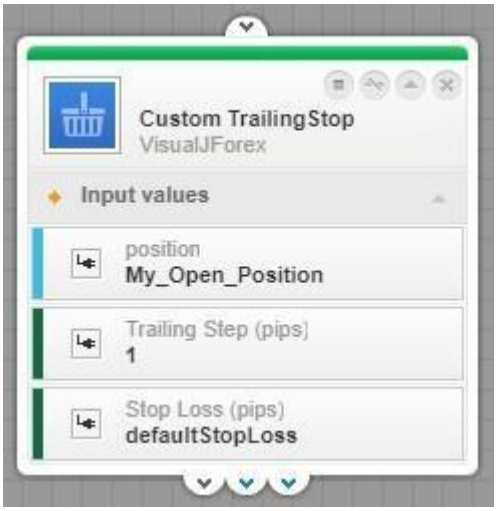
- Простой выход независимо от подтверждения ордера.
- Отправка ордера подтверждена (ордер размещен). Отказано в отправке ордера.



TrailingStop ("Трейлинг стоп"). Блок добавляет движущийся стоп-лосс (трейлинг стоп) к выбранной позиции. Первоначальный стоп-лосс задается в поле "Stop Loss". Шаг отслеживания ("Trailing Step (pips)") должен быть равным или превышать 10 пунктов. По аналогии с предыдущим блоком "Set Stop Loss" имеется возможность задать стоп-лосс в пунктах – целочисленной переменной ("Integer"), либо в виде конкретной котировки – вещественной переменной ("Double").

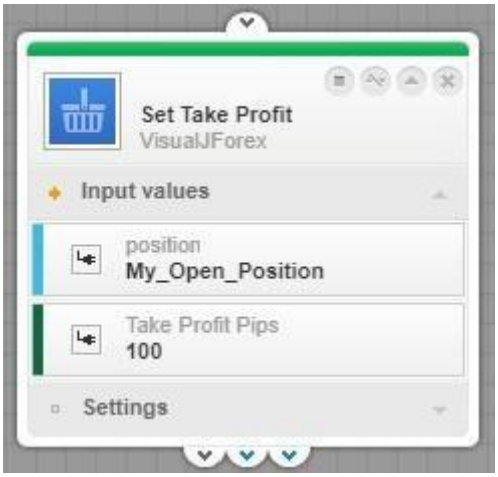
Выходы из блока:

- Простой выход независимо от подтверждения ордера.
- Отправка ордера подтверждена (ордер размещен). Отказано в отправке ордера.



Custom TrailingStop ("Пользовательский трейлинг стоп"). Блок предназначен для реализации трейлинг стопа с шагом отслеживания менее 10-ти пунктов. В блок добавляется переменная необходимой позиции, а также ее первоначальный стоп-лосс. Выходы блока идентичны предыдущим торговым блокам.

Блоки "Set Stop Loss" и "TrailingStop" должны полностью управляться стратегией. Процесс отслеживания стоп-лосса (трейлинг) выполняется самой стратегией, а не на стороне сервера. Стратегия должна иметь возможность выполнить эти блоки после достижения определенных ценовых уровней и условий.

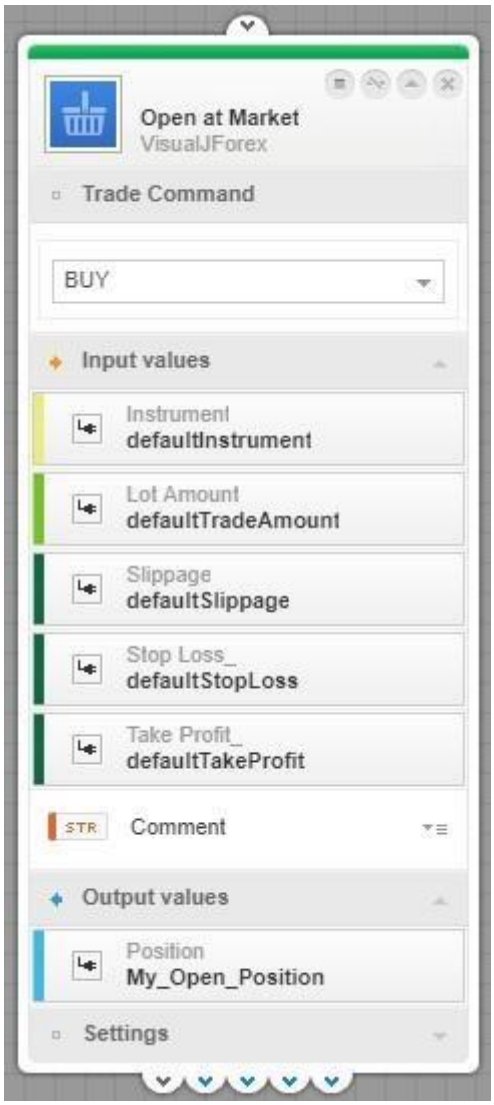


Set Take Profit ("Задать тейк профит"). Блок задает тейк профит для выбранной позиции. Позиция определяется входной переменной типа "Position" ("Позиция").

Уровень тейк профита задается либо в пунктах целым числом – тип переменной "Integer", либо конкретной котировкой – тип переменной "Double" (вещественная). Данный выбор выполняется в настройках блока ("Settings"). Также можно выбрать котировку реагирования (бид или аск).

Выходы из блока: • Простой выход независимо от подтверждения ордера.

- Отправка ордера подтверждена (ордер размещен).
- Отказано в отправке ордера.



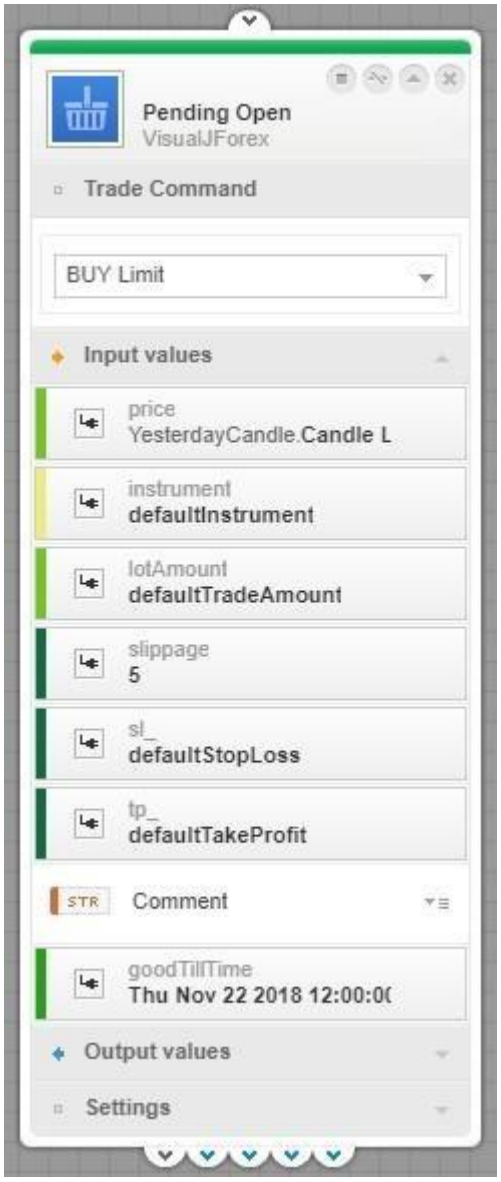
Open at Market ("Открыть по рынку"). Блок используется для отправки рыночных ордеров. При добавлении блока на рабочее пространство можно увидеть, что он содержит набор базовых переменных, однако входные и выходные переменные можно добавлять по своему усмотрению. Вид ордера выбирается из раскрывающегося списка торговых команд ("Trade Command"). Раздел настроек ("Settings") позволяет задать стоп-лосс и тейк профит в пунктах либо котировкой. Если ордера стоп-лосс или тейк профит не требуются для стратегии, пользователь может просто удалить соответствующую переменную, нажав на крестик в правом верхнем углу переменной. Это же касается и переменной "Slippage" ("Проскальзывание").

По умолчанию объем сделки ("Lot Amount") считается в миллионах (1 – значит один миллион; 0.001 – значит одна тысяча единиц базового актива или инструмента).

При добавлении блока на рабочее пространство поле комментария не заполнено, поскольку это не обязательная входная переменная. Пользователь сам решает – заполнять его или нет. Комментарий ("Comment") будет связан с позицией, а потому является ее идентификатором.

Выходы из блока: Простой выход независимо от подтверждения ордера.

- Отправка ордера подтверждена (ордер размещен; рыночные ордера мгновенно отправляются на рынок к исполнению).
- Отказано в отправке ордера.
- Ордер исполнен.
- Отказано в исполнении ордера (такие случаи являются довольно редкими).

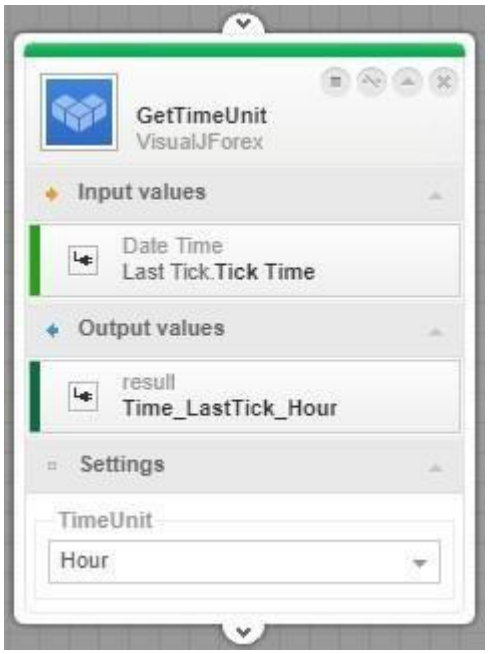


Pending Open ("Отложенный ордер"). Блок используется для размещения отложенных ордеров на открытие позиции. Перечень торговых команд (типов ордеров) представлен в выпадающем списке вверху блока: лимитные ордера, стоп-ордера (с опцией выбора стороны

реагирования – бид или аск), размещение офера и бида. В дополнение к общим входным переменным этот блок потребует ввода цены. Также он имеет поле "goodTillTime" ("Срок действия"), которое содержит время, до которого ордер остается действительным. Выходы из блока:

- Простой выход независимо от подтверждения ордера.
- Отправка ордера подтверждена (ордер размещен). Отказано в отправке ордера.
- Ордер исполнен: когда цена достигла заданного уровня и ордер исполнен. В платформе это можно проследить с помощью сообщения "Fill" ("Исполнен").
- Отказано в исполнении ордера (такие случаи являются довольно редкими).

5.3.5. Прочие – Miscellaneous



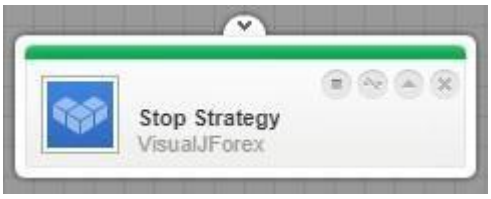
Get Time Unit ("Получить единицу времени"). Блок используется для получения метки времени либо из времени открытия свечи, либо из времени тика – обе переменные доступны в левой панели. Принцип работы блока в разбиении даты и времени на отдельные переменные: часы, минуты, секунды, дни недели и дни месяца. Полученные переменные сохраняются в целочисленном ("Integer") формате. Для получения полного формата времени используйте несколько блоков. См. раздел 7.10.

Что касается дней недели, то они нумеруются в западном формате, порядок следующий: воскресенье – 1, понедельник – 2, вторник – 3, среда – 4, четверг – 5, пятница – 6, суббота – 7.

Информация о дате и времени соответствует свечам или тиковым событиям. Возможно получить время тика или открытия свечи, однако текущее время в GMT получить невозможно.

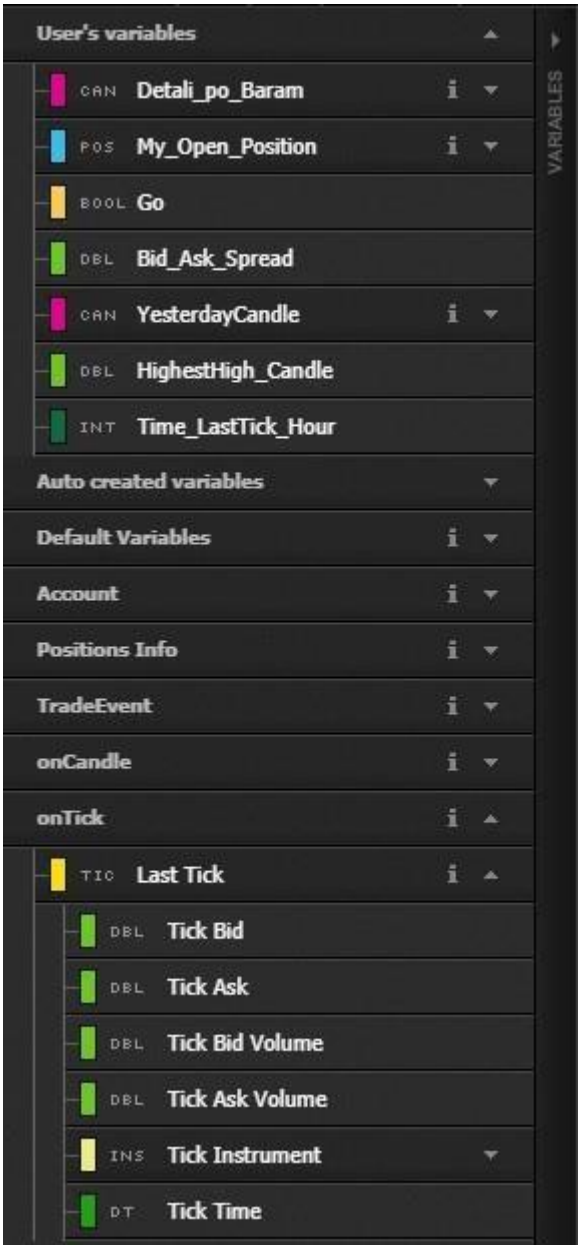
Sound Alert ("Звуковой алерт"). Для добавления звукового уведомления вставьте этот блок и выберите звук предупреждения для воспроизведения. По умолчанию для параметра "Beep on BackTesting" ("Давать сигнал на бэктестах") установлено значение "No" ("Нет"). Если сигналы нужны на бэктестах, переключите его на "Yes" ("Да").

Все оповещения из конструктора Visual JForex поддерживаются торговой платформой JForex. Поэтому, когда Java-код стратегии экспортируется в терминал JForex, эти оповещения также воспроизводятся.



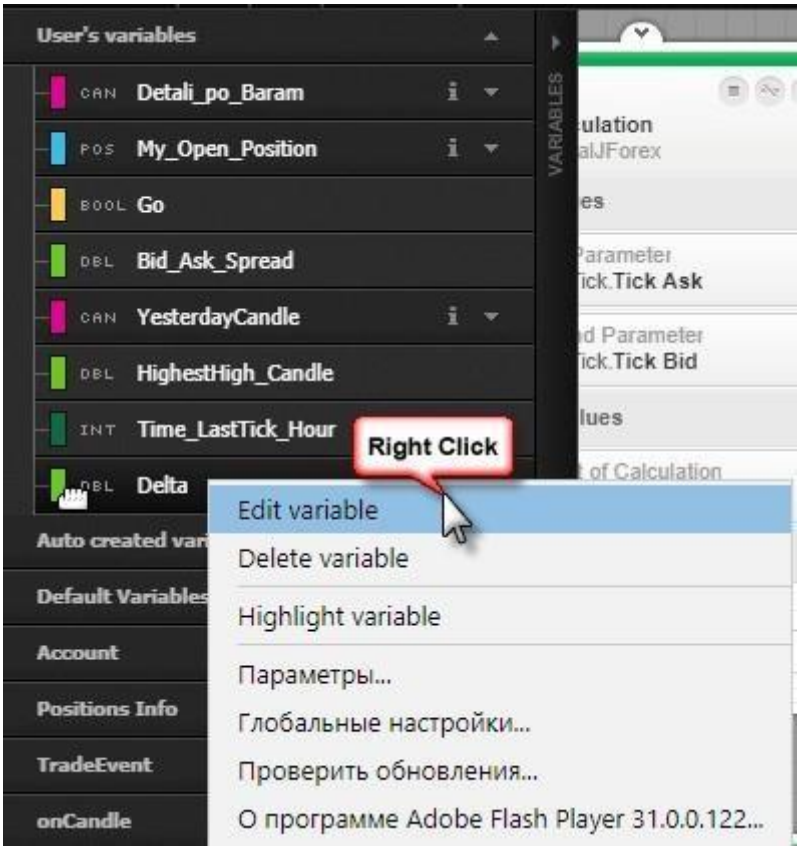
Stop Strategy ("Остановка стратегии"). Данный блок полностью останавливает стратегию.

6. Переменные

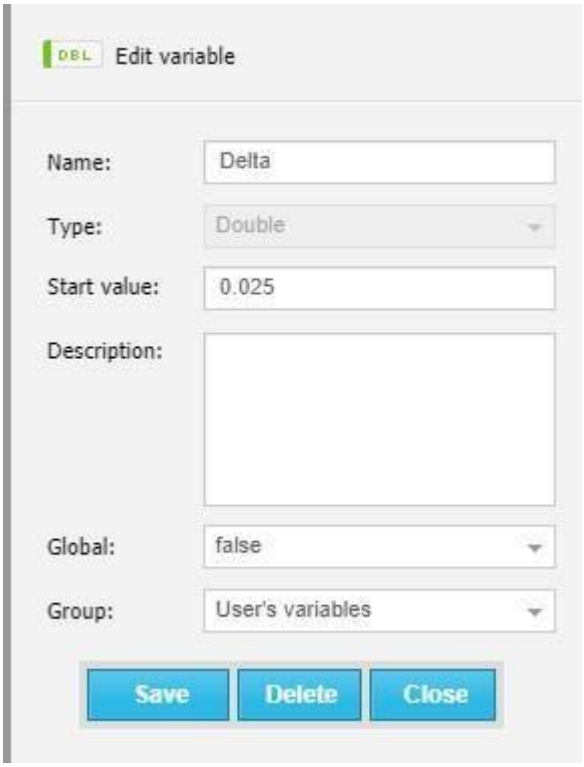


Список переменных доступен в левой части платформы и сгруппирован по категориям.

Контекстное меню и опции работы с переменными



- **Edit variable ("Редактировать переменную").** Редактирование имени переменной, ее стартового значения, описания, настройки глобальности и группы (подробнее см. ниже).
- **Delete variable ("Удалить переменную").** Удаление переменной.
- **Highlight variable ("Выделить переменную").** Выбранная переменная начинает мигать в рабочей области, чтобы ее было легко заметить. Чтобы привлечь внимание пользователя, после активации цветовой маркер переменной будет мигать в левой части, а также в соответствующем блоке.



Атрибуты переменной можно редактировать из окна на рис. выше. Тип переменной изменить нельзя. Если пользователю это необходимо, то нужно создать новую переменную.

Настройка глобальности ("Global"), если она имеет значение "true" ("истина"), позволяет впоследствии изменять переменную перед запуском стратегии в окне параметров стратегии.

6.1. Панель переменных

Панель переменных находится в левой части рабочего экрана. Она содержит переменные, которые используются для заполнения блоков. Переменные сортируются по категориям следующим образом:

6.1.1. Пользовательские переменные – User's variables

Эта подгруппа содержит переменные, созданные самим пользователем. Любая переменная, созданная пользователем, автоматически сохраняется в этой подгруппе. После создания переменной ее можно отредактировать или удалить, кликнув правой кнопкой мыши и выбрав действие из контекстного меню.

6.1.2. Автосозданные переменные – Auto created variables

Некоторые блоки автоматически создают свои выходные переменные, которые сохраняются в этой подгруппе. Например, блок индикатора автоматически создает переменную вещественного типа ("Double"), которую именует случайным образом. Пользователь может переименовать эту переменную, которая будет затем сохранена в разделе "Пользовательские переменные".

6.1.3. Базовые переменные – Default Variables

Эта подгруппа содержит переменные, которые считаются базовыми:

- defaultInstrument – инструмент по умолчанию.
- defaultTradeAmount – объем сделки по умолчанию.
- defaultSlippage – проскальзывание по умолчанию.
- defaultStopLoss – стоп-лосс по умолчанию.
- defaultTakeProfit – тейк профит по умолчанию.
- defaultPeriod – таймфрейм по умолчанию.

6.1.4. Аккаунт – Account

В этой подгруппе поместились переменные, относящиеся к торговому счету:

- **Account ID – ID аккаунта.** Идентификатор торгового счета.
- **Account Currency – валюта счета.** Базовая валюта торгового счета.
- **Equity – собственный капитал.** Капитал, доступный на торговом счете. Когда стратегия запускается в режиме реального времени, она использует текущий размер капитала. На бэктестах стратегии размер капитала определяется пользователем. По умолчанию собственный капитал равен 50,000 долларов США.
- **Leverage – кредитное плечо.** Настройки параметров кредитного плеча для торгового счета.
- **Margin Cut level – уровень "маржин кат".** Уровень использования кредитного плеча, при котором банк запускает процедуру "маржин кат". Обычно устанавливается на уровне 1:200.
- **Over Weekend End Leverage – кредитное плечо выходного дня.** Уровень кредитного плеча торгового счета, устанавливаемый за 3 часа до наступления выходных. Как правило, устанавливается в размере 1:30, но может быть изменен по запросу пользователя.
- **Use of Leverage – использование кредитного плеча.** Использование кредитного плеча в режиме реального времени.
- **Global Account – глобальная учетная запись.** Переменная указывает, работает ли торговый счет в режиме "хеджирование" или "без хеджирования" (глобальный режим). По умолчанию торговый счет настроен на режим "хеджирование" (не глобальный режим).

6.1.5. Информация о позициях – Positions Info

Информация о позициях – это раздел, динамически обрабатывающий состояние созданных позиций. Такие переменные при использовании с конкретным типом блоков, например, "Positions Viewer", могут показать состояние позиций, созданных до запуска стратегии. Под состоянием подразумевается состояние ордера или позиции на момент прочтения ее блоком "Positions Viewer". Также в переменную "Positions amount" ("Количество позиций") в режиме реального времени заносится информация о количестве позиций с выбранным состоянием. • **All Positions – все позиции.** Эта переменная часто используется с блоком "Positions Viewer" и служит для получения информации по всем позициям и ордерам на данном торговом счете независимо от состояния позиций или ордеров.

- **Open Positions – открытые позиции.** Эта переменная часто используется с блоком "Positions Viewer" и возвращает детальную информацию об открытой позиции (позициях).
- **Pending Positions – отложенные ордера.** Эта переменная часто используется с блоком "Positions Viewer" и возвращает информацию об отложенных ордерах следующих типов: Stop, Limit, MIT.

6.1.6. Группа переменных "TradeEvent" ("Торговое событие")

Эта группа переменных обрабатывает торговые сообщения, полученные платформой. Прошлые данные можно сохранять вручную в пользовательских переменных. Однако эта группа переменных весьма полезна, когда нужно работать с информацией, связанной с предыдущими сделками и ордерами. Переменные группы "Торговое событие" созданы для использования с одноименной стартовой точкой "TradeEvent". Эта стартовая точка особенная, поскольку она активируется только при получении торгового сообщения.

Переменные "Торговое событие" сортируются в многоуровневом режиме. На первом уровне стоит "LastTradeEvent" ("Последнее торговое событие"), содержащее самое свежее торговое сообщение по первой позиции.

Если стратегия не выполнялась, то предшествующие моменту запуска стратегии торговые сообщения не могут быть извлечены или использованы.

6.1.7. Группа переменных "onCandle" ("По свечам")

В этом подразделе хранятся переменные, относящиеся к текущим свечам. При этом рассматриваются обе стороны котирования (бид и аск) и по каждой из них сохраняется информация: цены открытия, максимума, минимума, закрытия, а также таймфрейм свечи и ее объем, торгуемый инструмент, время открытия и закрытия свечи.

В подгруппах "Last Bid Candle" и "Last Ask Candle" содержатся данные по свече, которая еще не закрылась. Единственные неизменные данные для такой свечи на этот момент – это цена ее открытия. Цены максимума и минимума, равно как и цена ее закрытия, не могут быть определены точно, пока свеча не будет закрыта.

6.1.8. Группа переменных "onTick" ("По тикам")

По аналогии с вышеописанными переменными из группы "onCandle" переменные в группе "onTick" – это: тики по бид и аск, проторгованный объем, а также наименование торгуемого инструмента и время формирования тика.

Время тика ("Tick Time") – это не текущее время по GMT, а метка времени, соответствующая этому тику по GMT.

6.2. Типы переменных

Далее следует описание переменных с той точки зрения, как их может применять пользователь конструктора Visual JForex.

6.2.1. Общие переменные

В платформе используется пять стандартных переменных:

Double – вещественная. Используется для отображения цен, объема сделки и показаний индикаторов. Котировка валютной пары сохраняется как вещественная переменная: например, расстояние в 5.2 пункта нужно сохранить в виде 0.00052 в переменной типа "Double".

Integer – целочисленная. Стоп-лосс и тейк профит, выраженные в пунктах, сохраняются как целочисленные переменные. Данный тип переменных не поддерживает десятичные знаки. Например, если пользователю потребуется установить стоп-лосс в 5.2 пункта, то ему нужно сначала преобразовать эту величину в тип "Double", а в торговом блоке в поле для стоп-лосса указать значение цены, а не количество пунктов (см. настройки блока в разделе 4).

Boolean – булева. Логическая переменная, принимающая только два значения – "true" или "false" ("истина" или "ложь"). Она широко используется для реализации логических выражений. Пример: позиция может быть длинной или короткой, поэтому переменная "Position is Long" может принять значение либо "true" ("истина"), либо "false" ("ложь").

String – строка. Поддерживает ввод данных в виде текста. Комментарий к позиции является строкой. Кроме того, такой тип принимает идентификатор позиции ("Position ID"), который может быть комбинацией текста и цифр.

Date and Time – дата и время. Переменная "Date and Time" в интерфейсе конструктора отображается в европейском (американском) формате даты и времени, однако в прикладной части платформы она выражается в формате эпохи Linux. Формат эпохи – это время, выраженное в миллисекундах. Например, 1.5 часа – это 90 * 60 * 1000 = 5,400,000 миллисекунд.

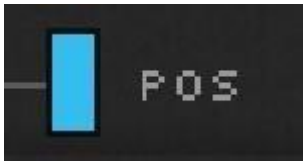
6.2.2. Специальные переменные Visual JForex

Перечисленные ниже переменные используются в JForex API – библиотеке, используемой при программировании торговых стратегий для торговой платформы Dukascopy JForex.

Массив – это диапазон данных, описывающих некий элемент. Переменные "Tick" ("Тик"), "Candle" ("Свеча"), "Position" ("Позиция"), "Signals" ("Сигналы"), описанные ниже, являются массивами, т.к. они включают в себя набор переменных, содержащих соответствующие названию массива данные.

Tick – тик. Это массив, содержащий тиковые цены обеих сторон котирования (бид и аск), проторгованный на данном тике объем, наименование инструмента, а также дату и время тика.

Candle – свеча. Это массив, содержащий цены открытия, максимума, минимума, закрытия свечи (OHLC), таймфрейм свечи, наименование инструмента, а также дату и время открытия свечи.



Position – позиция. Массив "Position" содержит информацию, относящуюся к конкретной позиции. Массив включает: идентификатор позиции ("Position ID"), объем сделки по этой позиции ("Position Amount"), цену и время закрытия позиции, время создания и исполнения позиции, цену открытия позиции и т.д. Полный список можно получить, создав переменную

типа "Position". Для "вскрытия" информации из этой переменной и получения данных по предыдущим или текущим позициям в основном используется блок "Positions Viewer".

Instrument – инструмент. Эта переменная относится к торгуемому инструменту. При создании новой переменной с этим типом пользователю будет предложено выбрать инструмент из выпадающего списка.

State – состояние. С помощью этой переменной можно получить состояние ордера или позиции. Обычно он доступен в массиве "Position" и всегда меняется динамически. Возможные значения состояния ордера или позиции:

- *Created – создан.* Ордер только что создан пользователем и отправлен в систему.
- *Opened – открыт для исполнения.* Ордер попадает в систему и регистрируется в ней. Ордер далее дожидается очереди исполнения.
- *Filled – исполнен.* Ордер исполняется брокером и становится открытой позицией.
- *Closed – закрыт.* Позиция закрыта.
- *Cancelled – отменен.* Ордер либо отвергается системой, либо отменяется пользователем.

Period – таймфрейм. Эта переменная принадлежит массиву "Candle" ("Свеча"). В ней используются стандартные временные периоды: 10 сек, 1 мин, 5 мин, 10 мин, 15 мин, 30 мин, 1 ч, 4 ч, 1 д, 1 нед и 1 мес. Любой другой таймфрейм, отличный от предыдущего списка, не поддерживается. Присвоение других пользовательских значений для таймфреймов требует обходных решений.

Command – команда. Переменная содержит типы ордеров, а именно: Buy, Sell, Buy Limit, Sell Limit, Buy Stop, Sell Stop, Buy Limit by Bid, Sell Limit by Ask, Buy Stop by Bid, Sell Stop by Ask, Place Bid, Place Offer.

Message – сообщение. Переменная работает с сообщениями, полученными платформой. Их классификация доступна при работе с переменной "Message Type" ("Тип сообщения"). Это массив, размещенный в группе переменных "TradeEvent".

Message Type – тип сообщения. Массив, содержащий виды торговых сообщений, получаемых платформой, а также некоторые дополнительные типы сообщений, которые еще не полностью реализованы в визуальном конструкторе. Переменную этого типа можно найти в группе переменных "TradeEvent".

Поддерживаемые виды сообщений: *Position Reject – ордер отклонен.*

Ордер не принят системой.

- *Position Submit – ордер отправлен.* Ордер отправлен в систему и получен сервером.
- *Position Fill – ордер исполнен.* Платформа получает сообщение "Fill" ("Исполнено").
- *Position Fill Reject – отказано в исполнении ордера.* Система не в состоянии исполнить ордер.
- *Position Close Reject – отказано в закрытии позиции.* Запрос о закрытии позиции отклоняется сервером или рынком.
- *Position Close – позиция закрыта.* Позиция успешно закрыта.
- *Position Change – ордер изменен.* Успешное изменение (редактирование) ордера.
- *Position Change Reject – отказано в изменении ордера.* Запрос на изменение (редактирование) ордера отклоняется сервером.

В настоящее время не поддерживаются следующие виды сообщений:

- *Positions Merge – объединение позиций.* Объединение 2 или более позиций выполнено успешно.
- *Positions Merge Reject – отказано в объединении позиций.* Запрос на объединение позиций отклоняется сервером.
- *Mail – почта.* Обработка запросов отправки электронной почты, предусмотренных стратегией (когда отправляется запрос на отправку электронной почты).
- *News – новости.* Рыночные новости, полученные платформой.
- *Calendar – календарь.* Получено событие из экономического календаря.
- *Notification – уведомление.* Получены уведомления платформы.
- *Instrument Status – состояние инструмента.* Состояние инструмента (если изменилось).

- *Connection Status – состояние соединения.* Текущее состояние соединения для учетной записи (если изменилось).

Offer Side – сторона котирования. Переменная связана с методом расчета индикатора в зависимости от стороны котирования – либо бид, либо аск.

Signal – сигнал. Сигнал представляет собой массив переменных, используемых в тех случаях, когда одна стратегия используется в другой в виде блока. Подробнее см. "Использование стратегии как блока" в разделе 5.2. Данная переменная воспроизводит торговые сигналы без отправки торговых команд как таковых. Она создается для особых случаев использования торговых сигналов. Также данный подход упрощает работу в Visual JForex в деле реализации сложных идей.

Массив "Signal" содержит подмассив переменных "Signal Type" ("Тип сигнала"), а информация по рассматриваемой позиции (сгенерированной сигналом) отражена в очередном массиве переменных "Signal Position" ("Позиция по сигналу"), в котором содержатся такие же переменные, как в массиве переменных типа "Position".

Signals – сигналы. В большинстве случаев эта переменная создается для хранения торговых сигналов, когда стратегия используется как блок. Этот массив необходимо "вскрывать" с помощью блока "Loop Viewer". Полученный с помощью "Loop Viewer" массив переменных имеет тип "Signal" (см. выше).

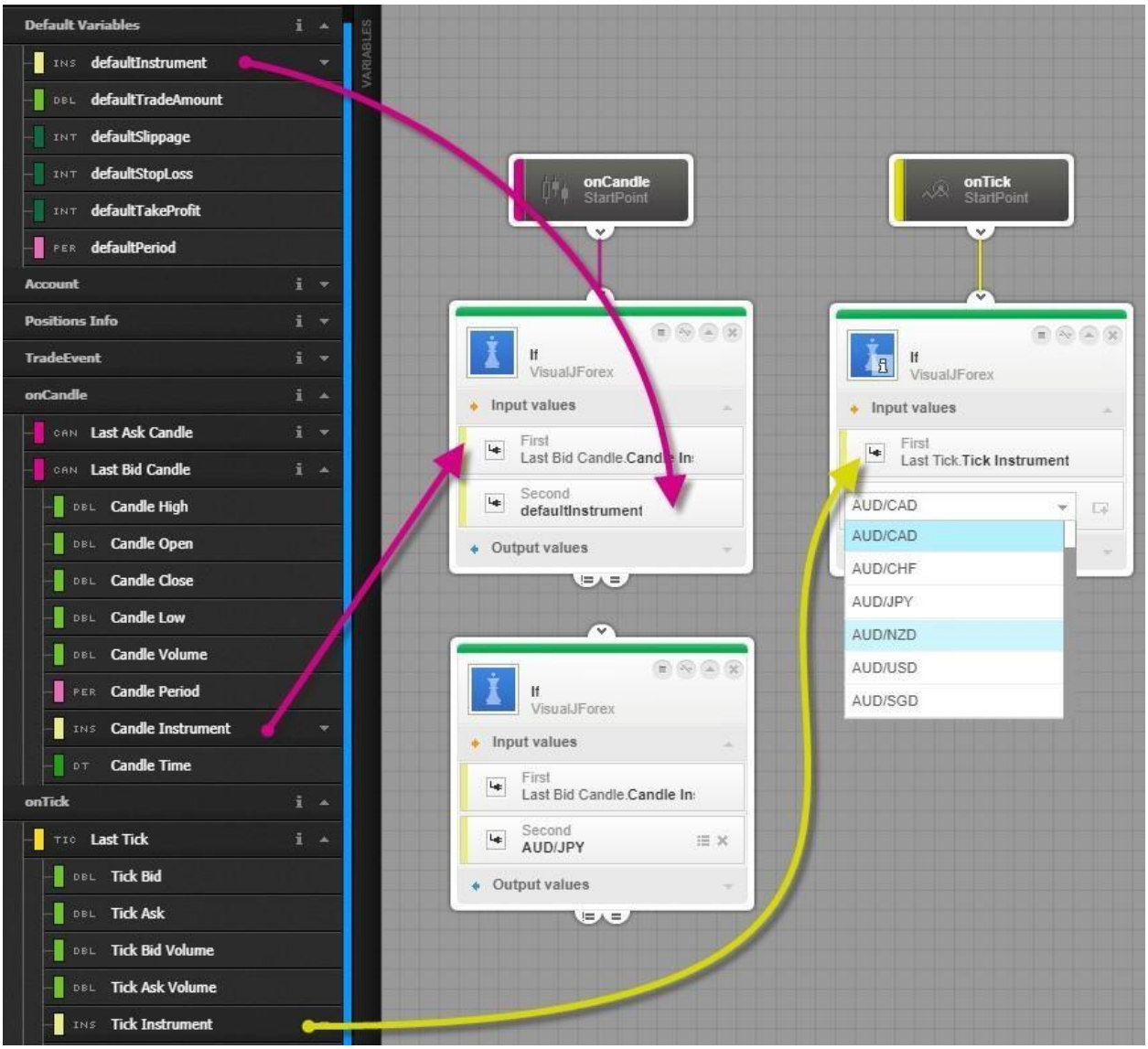
Signal Type – тип сигнала. Являясь частью массива "Signal", "Signal Type" ("Тип сигнала") содержит торговые команды, генерируемые стратегией-блоком. Существуют следующие типы сигналов: "Position Buy" ("Позиция на покупку"), "Position Sell" ("Позиция на продажу"), "Position Close" ("Закрыть позицию"), "Position Modify" ("Редактировать позицию"), "Position Merge" ("Объединить позиции").

Moving Average Type (maType) – тип скользящей средней. Это особая переменная, которая содержит тип скользящей средней. Она принимает одно из заданных значений из выпадающего списка из 12-ти типов скользящих.

Applied Price – используемая цена. По аналогии с предыдущей переменной "Applied Price" также используется в сочетании с индикаторами и обозначает сторону котирования, используемую в формуле индикатора (цена закрытия, открытия, средняя, взвешенная и т.д.). Эта переменная помогает динамически изменять метод расчета индикаторов.

7. Приемы работы в конструкторе Visual JForex

7.1. Базовые настройки стратегии



7.1.1. Как задать рабочий инструмент

При использовании стартовой точки "onCandle" в Visual JForex рабочим инструментом по умолчанию становится EURUSD. Однако, если планируется использовать стратегию в платформе JForex, необходимо конкретно указывать рабочий инструмент. Если же в стратегии не задан рабочий инструмент, то во время реального применения она будет беспорядочно торговать на различных инструментах. Технически это связано с особенностями работы стартовых точек "onCandle" и "onTick" и их настройками по умолчанию, согласно которым эти стартовые точки сканируют сразу все инструменты.

Необходимость выбора рабочего инструмента состоит в том, чтобы отфильтровать неиспользуемые инструменты и сохранить только те, которые нужны для работы. На приведенном выше скриншоте первый верхний левый блок фильтрует инструмент, используя переменную "defaultInstrument" ("Инструмент по умолчанию") и "Last Bid Candle.Candle Instrument" ("Последняя свеча по биду. Инструмент свечи"). Инструмент будет отфильтрован,

если для выхода из блока "IF" использовать знак "=". Второй блок "IF" ниже также является примером фильтрации неиспользуемых инструментов, но уже без переменной "defaultInstrument". Переменная "Last Bid Candle.Candle Instrument" попросту заносится в первое поле блока "IF", а при нажатии на второе (Second) поле рабочий инструмент можно выбрать из раскрывающегося списка. Этот метод может использоваться в частных случаях, когда в стратегии используется более одного инструмента.

Когда стратегия начинается от стартовой точки "onTick", то применяется тот же способ, но вместо переменной "Candle Instrument" нужно использовать переменную "Tick Instrument" ("Инструмент тика"), как показано в верхнем правом блоке.

7.1.2. Как задать рабочий таймфрейм

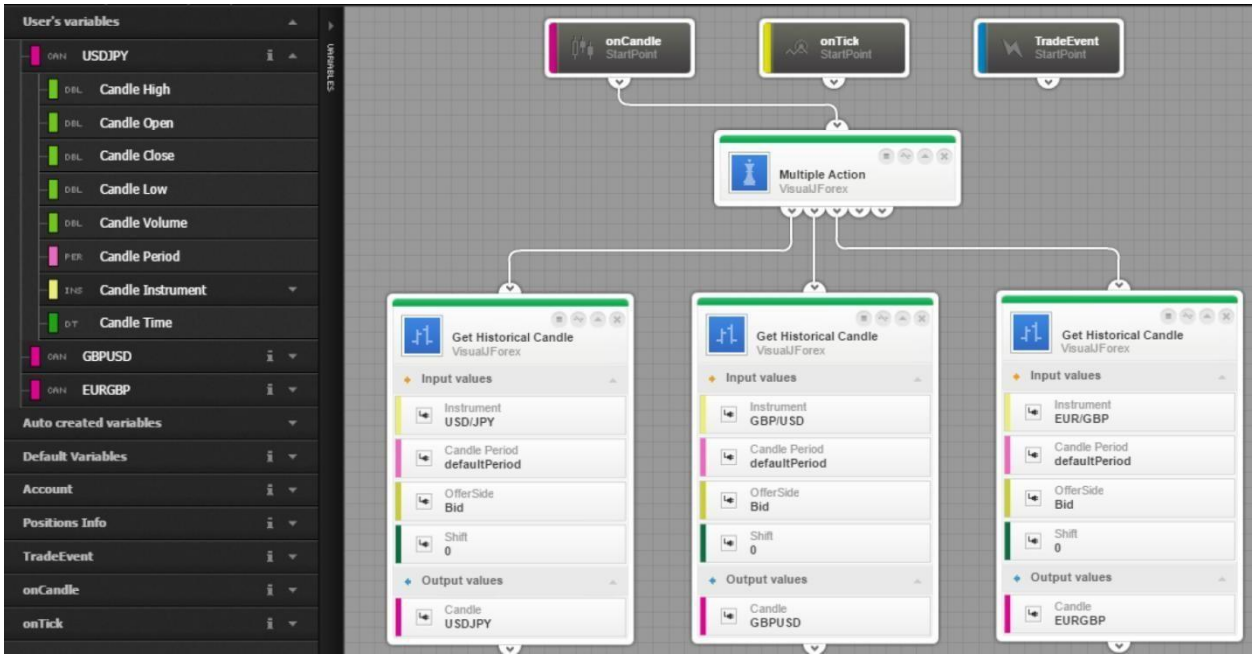
Определение рабочего таймфрейма проходит примерно так же, как и рабочего инструмента. Отличие лишь в использовании переменных "Last Bid Candle.Candle Period" и "defaultPeriod", которые отфильтруют ненужные таймфреймы. Причина в том, что метод "onCandle" автоматически мониторит все таймфреймы, а потому активируется каждые 10 секунд, поскольку это самый короткий период, доступный по умолчанию. Любой другой период свечей, отличный от стандартных, не поддерживается, присвоение других пользовательских значений для таймфреймов требует обходных решений.

7.2. Как использовать несколько инструментов

Разработка стратегии, рассчитанной на работу сразу с несколькими инструментами, возможна, но тестирование ее сложнее, так как это требует экспорта стратегии в платформу JForex для запуска тестера, одновременно тестирующего несколько инструментов.

Технически создание такой стратегии осуществляется с помощью блока "Get Historical Candle" ("Получить бар из истории"), который обращается к историческим свечным данным для каждого из инструментов.

Используйте для каждого из тестируемых инструментов свой отдельный блок "Get Historical Candle", предварительно удалив из него переменную "defaultInstrument", а затем выберите нужный инструмент из раскрывающегося списка. Вместо автосозданных выходных переменных в следующем примере используется константа, что впоследствии облегчит идентификацию переменных для инструмента.



Перед указанными выше блоками можно добавить фильтр таймфрейма.

На скриншоте выше переменные были вынесены в рабочую область, чтобы отлаживать их значения на бэктесте в процессоре стратегий. Как видно, котировки OHLC отображаются для каждого инструмента и могут использоваться в следующих блоках стратегии.

Показанный способ получения свечных данных часто используется, когда в стратегии реализуется кросс-инструментальная логика.

Другими словами, если условие A выполняется на паре EURUSD, то тогда переходят к открытию позиции, например, по паре USDJPY.

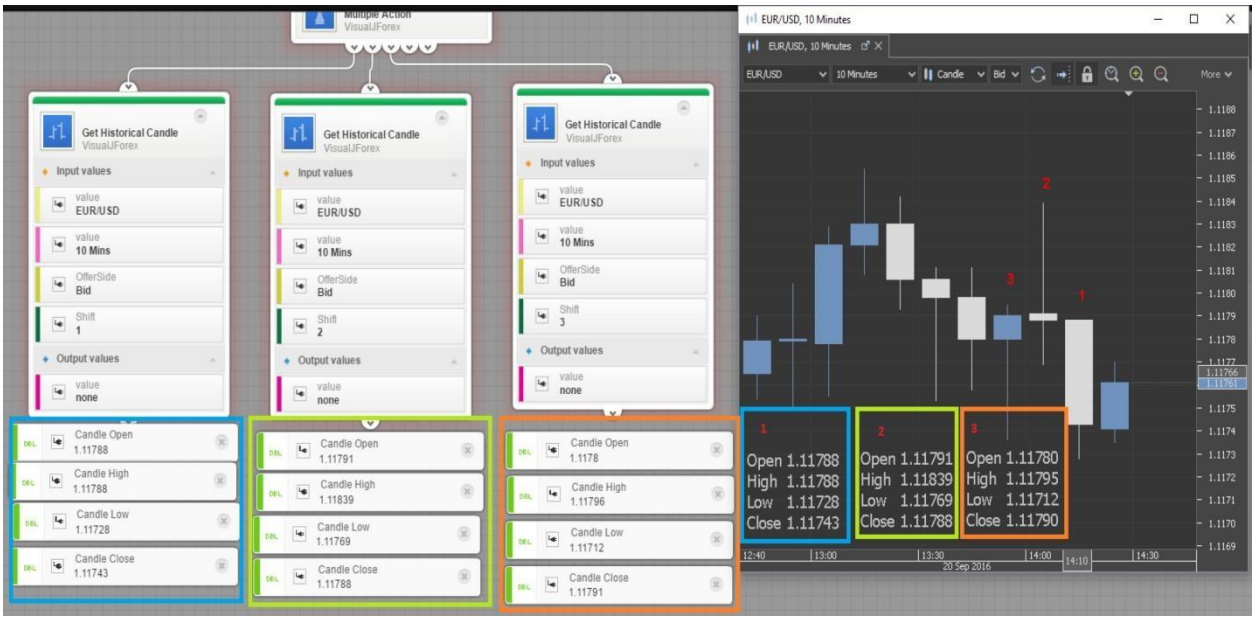
Таким образом, чтобы торговать данным инструментом, условие для входа сначала должно быть проверено на другом инструменте. Для стратегий, реализующих ту же логику, применяемую к нескольким инструментам, проще создать дополнительный файл стратегии и изменить инструмент на уровне Visual JForex. Либо можно запустить или протестировать стратегию в торговой платформе JForex и выбрать необходимые инструменты там.

В процессоре стратегий Visual JForex невозможно тестирование мультивалютных стратегий, поскольку он по умолчанию рассчитывает только один инструмент. Такие стратегии можно протестировать только в торговой платформе JForex.

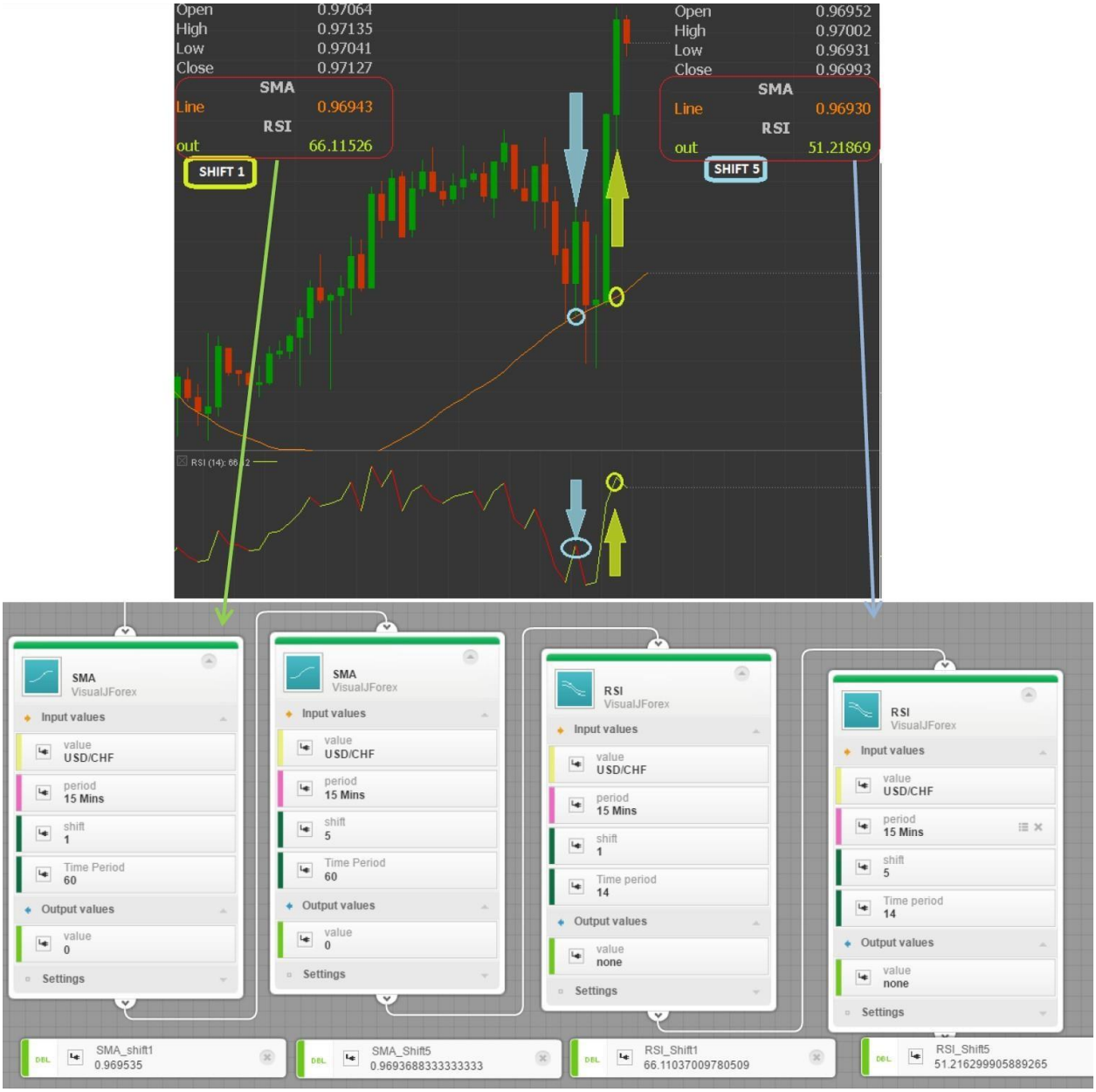
7.3. Как работать с параметром "Shift"

Параметр "Shift" ("Смещение") доступен практически для всех индикаторов, а также для блоков "Get Historical Candle(s)". Этот параметр особенно полезен, когда нужно получить данные по конкретным свечам в истории.

Для текущей – не закрывшейся – свечи значение "Shift" равно 0. Поэтому единственной известной на текущий момент информацией о ней является цена открытия и время открытия, а остальное будет доступно только когда свеча будет закрыта. Таким образом, сразу после открытия новой свечи рассматривавшаяся нами ранее свеча станет предыдущей свечой по отношению к следующей за ней новой свечой, только что начавшей процесс своего формирования. Поэтому, чтобы получить данные о закрывшейся свече, используется значение "Shift" = 1. "Shift" = 2, соответственно, связан со второй свечой влево от формирующейся, 3 – с третьей и т.д.



Значения смещения часто используются в индикаторах, особенно когда пользователю требуется определить тренд по линейному индикатору. Задача выполняется путем сравнения последнего значения индикатора (например, "Shift" = 1) с предшествующим (например, "Shift" = 2). С помощью такого сравнения можно выяснить, является тренд бычьим, медвежьим или же это флэт.

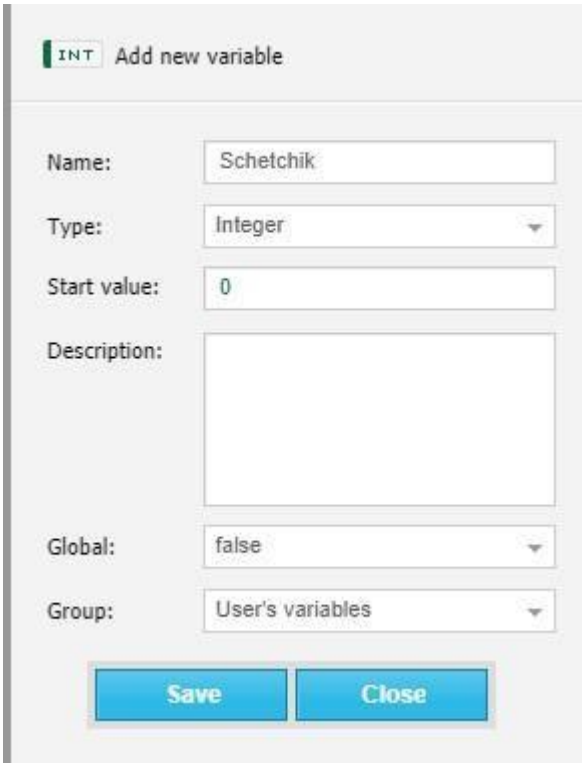


На приведенном выше скриншоте индикаторы SMA и RSI используются со смещениями 1 и 5, для этого требуется всего 4 блока. Когда стратегия запускается в режиме реального времени, выходные переменные Visual JForex получают свои значения в реальном времени и соответствуют показателям индикаторов в торговой платформе JForex. Иногда наблюдается разница между значениями Visual JForex и JForex, что объясняется правилами округления, применяемым в вычислениях показаний индикаторов в платформе JForex.

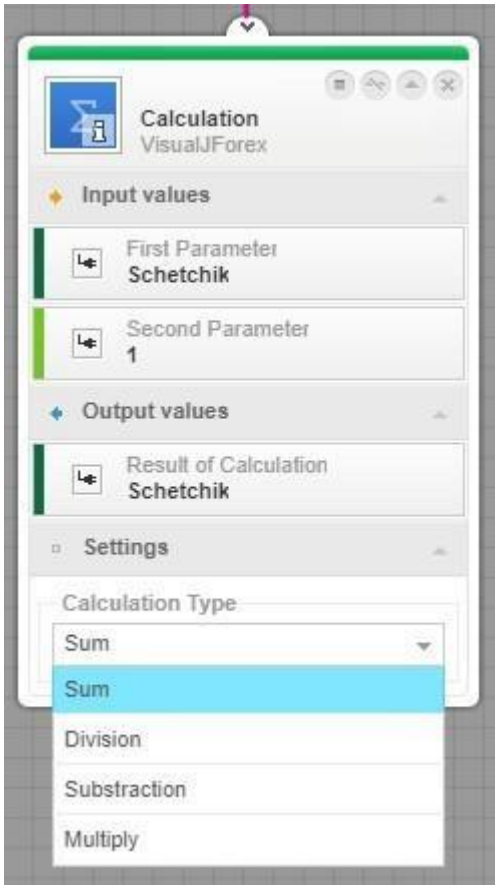
7.4. Как реализовать счетчик

Счетчик – это простой инструмент, помогающий в реализации последовательности команд. Базовая логика счетчика заключается в увеличении переменной при возникновении ситуации A до тех пор, пока не произойдет ситуация B, а затем происходит сброс счетчика назад на значение 0. В большинстве случаев он реализуется с использованием блока "Calculation" следующим образом:

Создайте новую переменную вещественного ("Double") или целочисленного ("Integer") типа и задайте ей значение = 0. В этом примере имя новой переменной "Schetchik".



Вставьте новую переменную "Schetchik" в блок "Calculation", создайте во втором поле переменную и установите ее значение = 1.

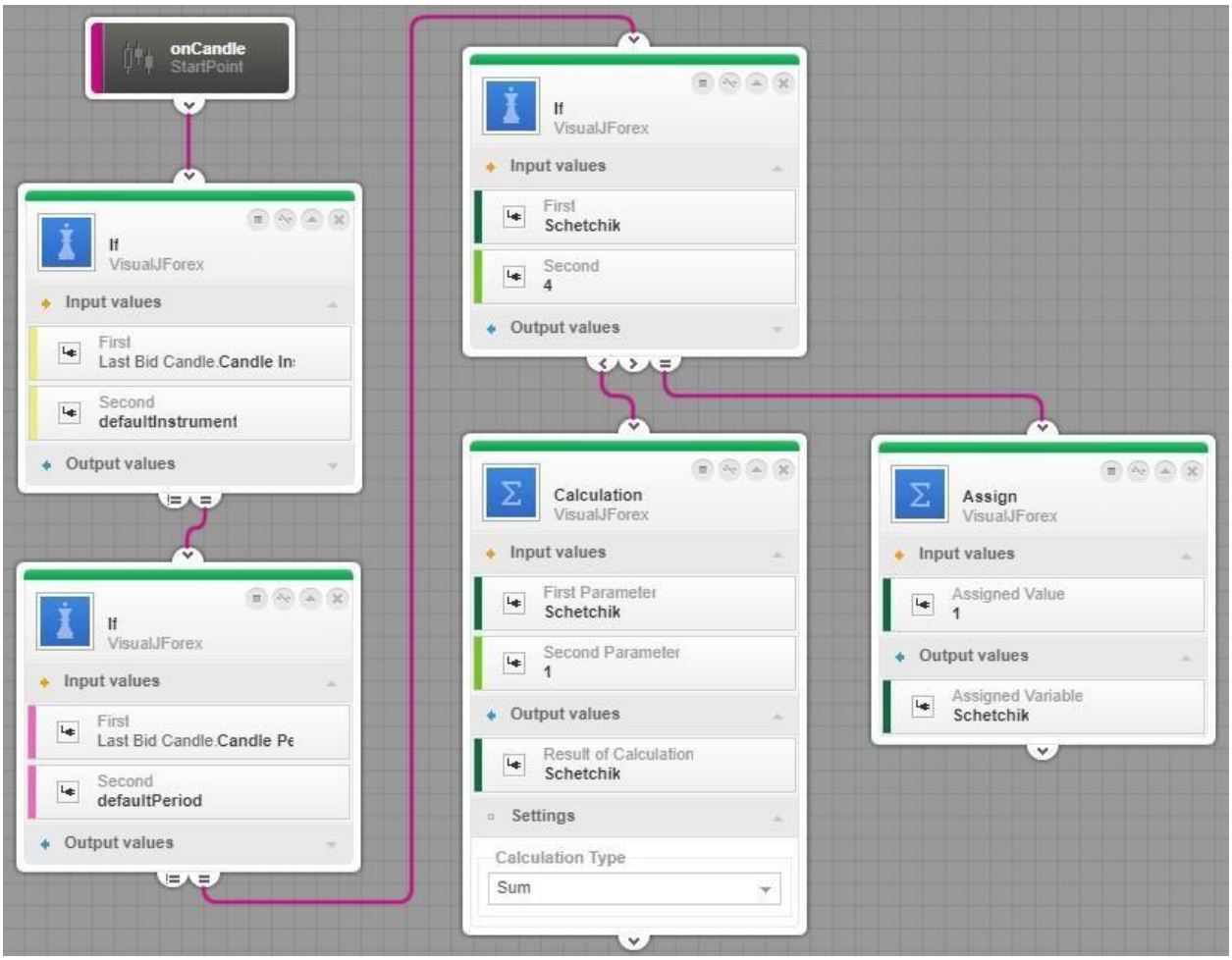


В качестве выходной переменной используйте ту же – "Schetchik", а в настройках ("Settings") и выберите "Sum" ("Сумма") в качестве математического действия.

Счетчик готов к использованию.

Способ подключения этого блока будет определять логику счетчика. **Пример**

1

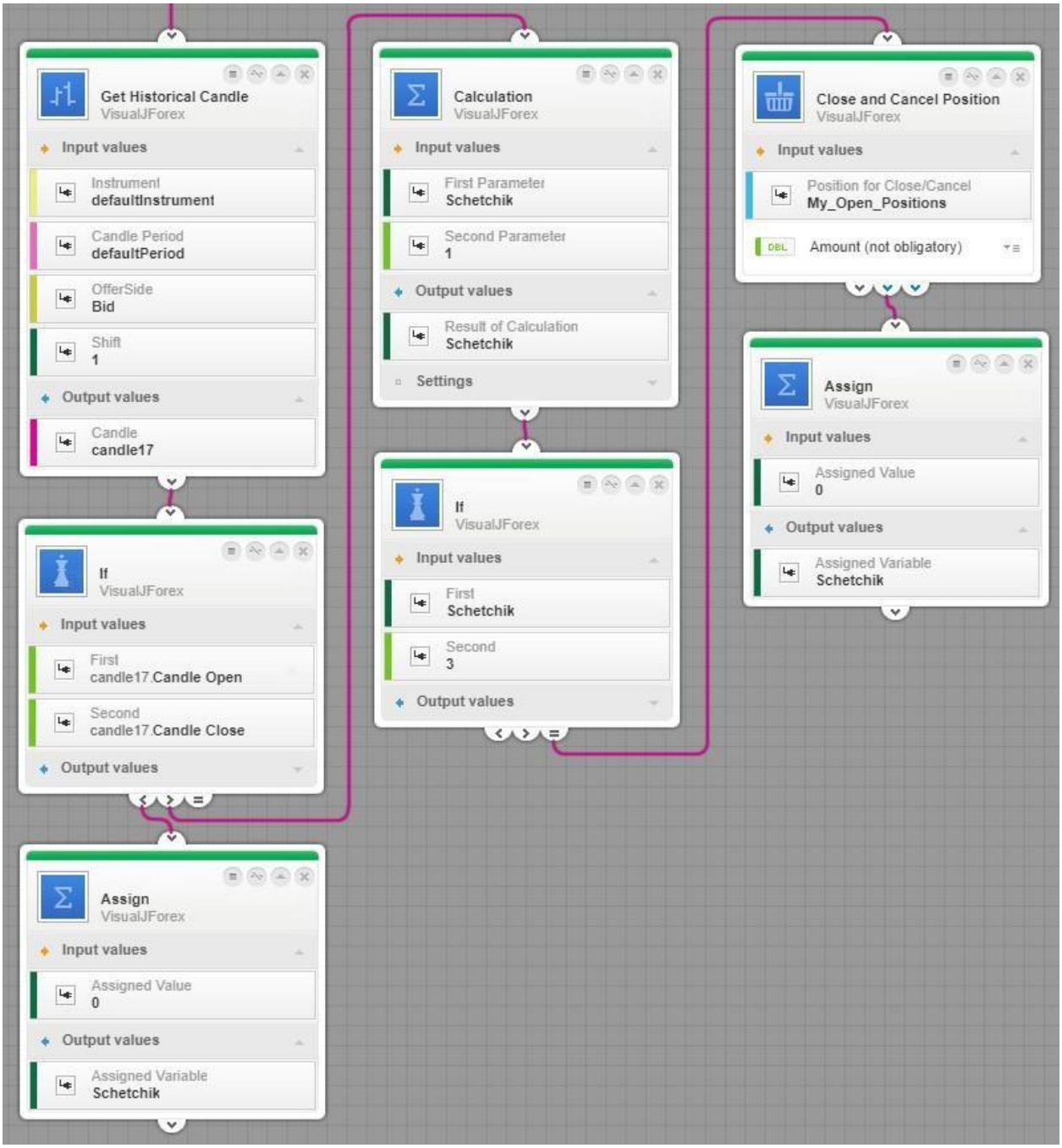


Задача: допустим, нужно построить свечи с таймфреймом 40 секунд.

1. Задать таймфрейм 10 сек.
2. Построить счетчик, как описано выше.
3. 40 сек – это 4 раза по 10 сек, поэтому счетчик должен быть установлен на увеличение до 4, а после сброшен на 1.

Условие в блоке "IF" важно, так как оно проверяет значение счетчика: достигло ли оно уровня 4. По достижении значения 4 переменная счетчика будет сброшена на 1, и цикл повторится снова.

Пример 2



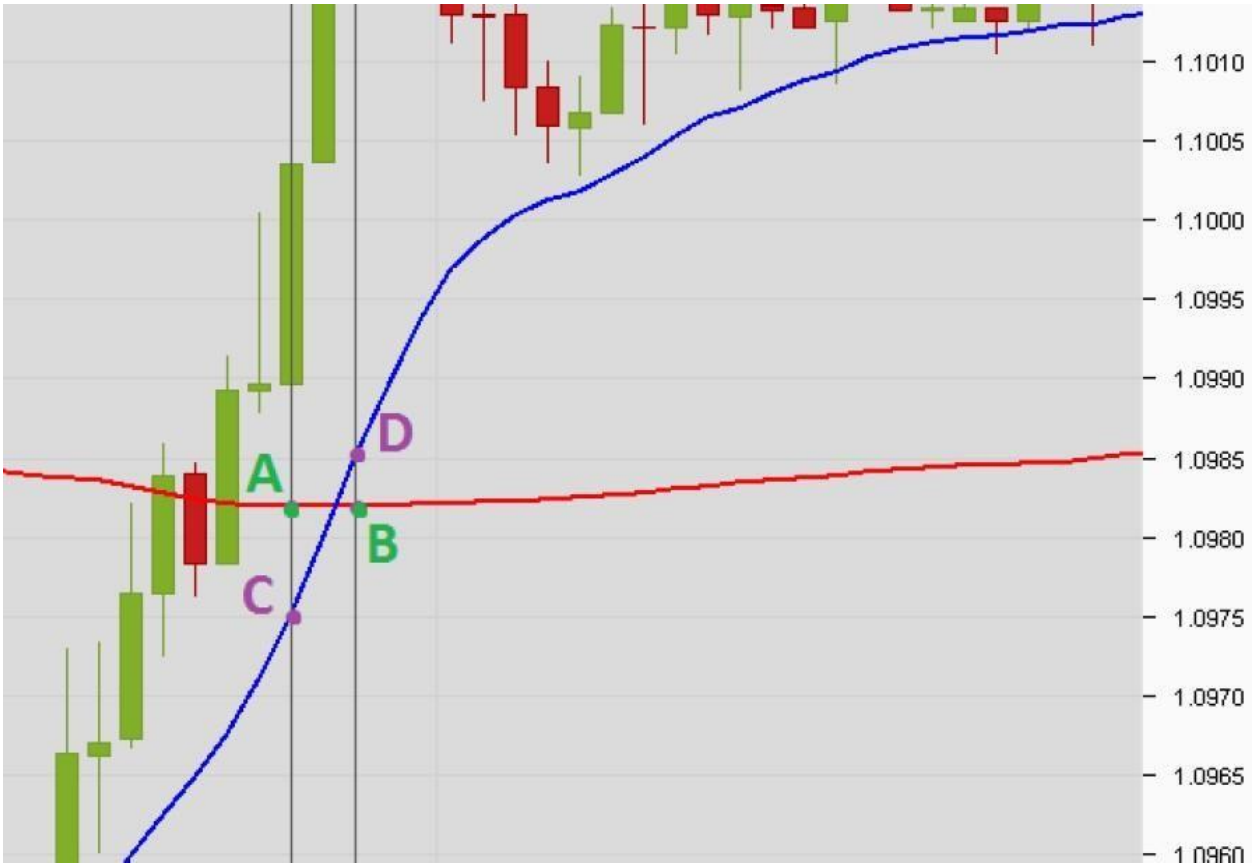
Задача: Закрыть позицию на покупку, если после ее открытия появились три последовательных медвежьих (красных) свечи.

1. Проверить состояние позиции, оно должно равняться "Filled" ("Ордер исполнен").
2. Получить данные о предыдущей свече и проверить, какая она: красная (медвежья) или зеленая (бычья).
3. Внедрить счетчик, чтобы убедиться в наличии трех красных (медвежьих) свечей подряд.

Показанные выше блоки обрабатывают часть стратегии, которая отвечает за нахождение трех красных свечей подряд. Блок "Get Historical Candle" следует за блоком "IF", который проверяет состояние позиции (должно быть равно "Filled"). Как только алгоритм видит, что предыдущая свеча – медвежья (цена открытия больше цены закрытия; Candle Open > Candle Close), счетчик

увеличится на 1, и это повторится вплоть до возникновения трех аналогичных ситуаций (красных свечей) подряд. Частота обновления счетчика соответствует таймфрейму, заданному как рабочий в стратегии. Как сказано выше, счетчик будет наращивать значение только в том случае, если предыдущая свеча была красной (медвежьей). В случае же, если она зеленая (бычья), счетчик сбрасывается на 0, потому что в этом случае условие будет нарушено. Достижение счетчиком значения 3 подтвердит, что предыдущие 3 свечи были медвежьи, поэтому позиция на покупку будет закрыта, а счетчик снова обнулится.

7.5. Как реализовать пересечение двух индикаторов

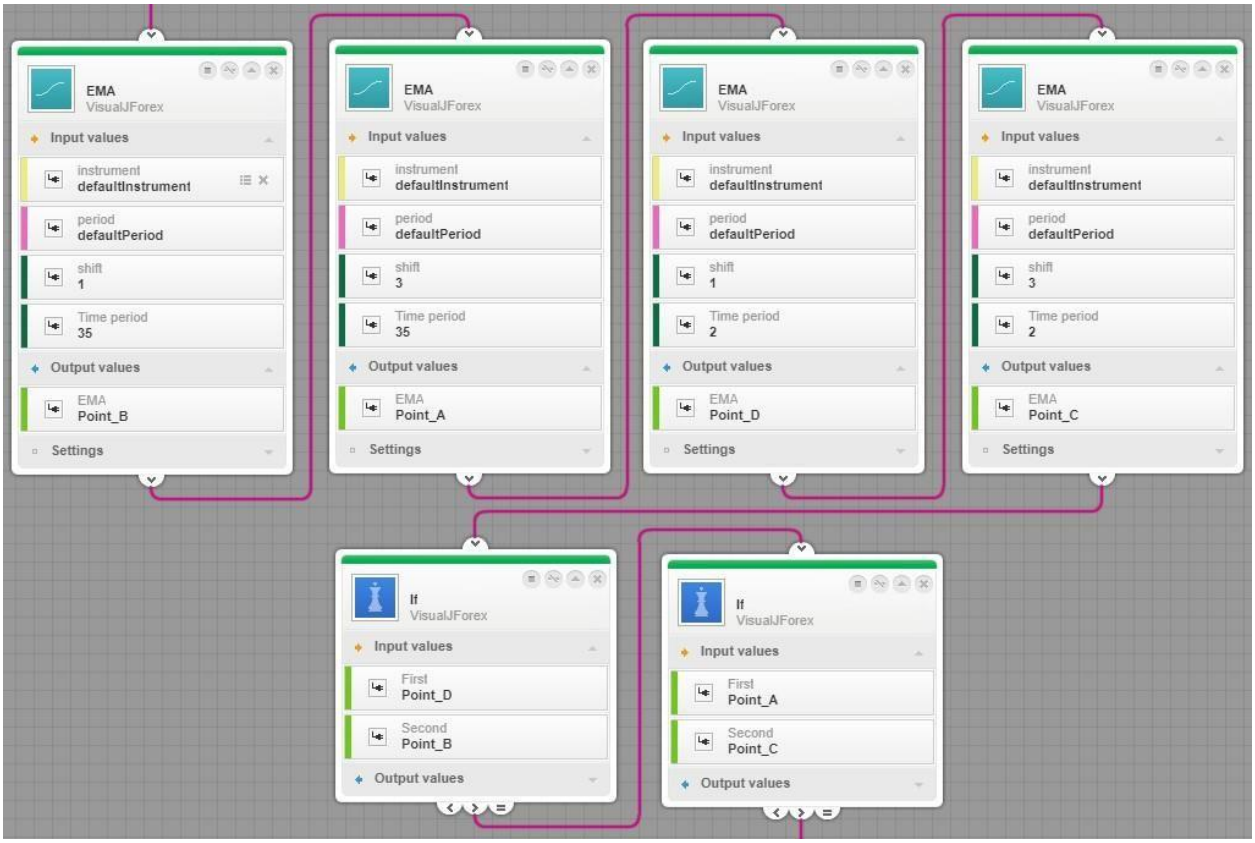


Пересечение между двумя линиями определяется с помощью четырех точек, например: A, B, C и D. Идея состоит в том, чтобы вызвать значения этих точек и далее проверить их на предмет пересечения. Это делается с помощью параметров "Shift" в блоках индикаторов.

Пересечение определяется как: A>C и D>B – это справедливо для случая, когда быстрый индикатор (синий мувинг) пересек медленный индикатор (красный мувинг) в восходящем направлении. Чтобы определить тренд каждой линии, можно добавить дополнительную проверку: A<B и C<D для восходящего (бычьего) пересечения или A>B и C>D для нисходящего (медвежьего) пересечения – т.н. "мертвого креста".

Первый шаг – добавление индикатора с соответствующими смещениями. Важно отметить, что обнаружение этого паттерна зависит от того, как выбраны значения "Shift": "Shift" = 1 относительно "Shift" = 2 более чувствителен, чем "Shift" = 1 относительно "Shift" = 3. Поэтому больше шансов получить правильное пересечение, когда между введенными значения смещения большее расстояние.

Сравнение выполняется с помощью блока "IF" следующим образом:

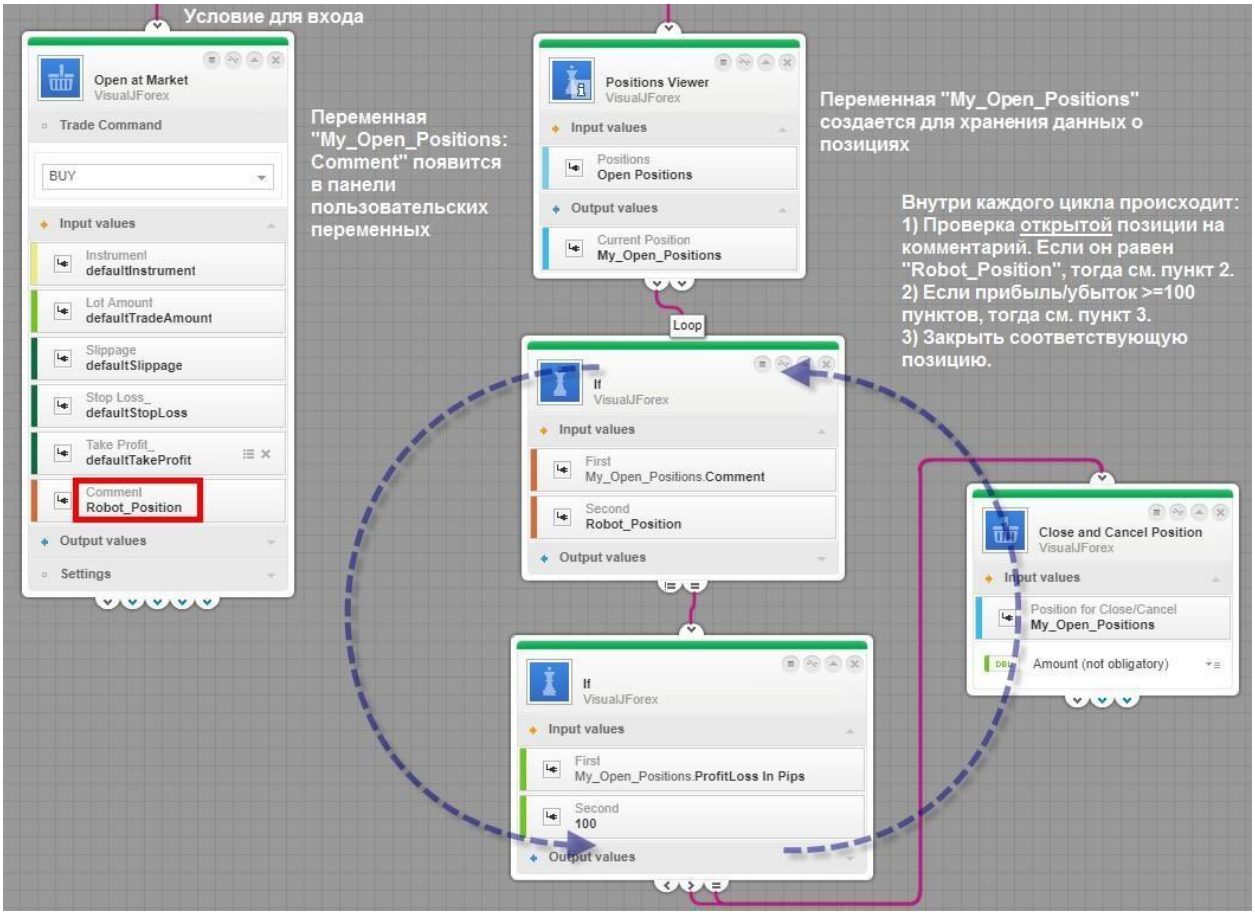


7.6. Как идентифицировать и работать с конкретными позициями

Позиции, созданные вручную или стратегией, можно прочитать с помощью блока "Positions Viewer". Этот блок работает как датчик с использованием функции цикла, внутри которого считывается вся информация по каждой позиции. С помощью такого цикла легко получить информацию по позициям: их направление, соответствующие стоп-лосс и тейк профит, текущий незафиксированный профит/убыток и т.д.

В примере ниже показано, как управлять открытыми позициями, находить позиции, открытые роботом, и закрывать те, где незафиксированная прибыль достигла 100 пунктов.

Блок "Positions Viewer" можно связать напрямую со стартовой точкой. Если есть промежуточные условия, то пользователь должен организовать логику стратегии так, чтобы эти условия не мешали работе блока "Positions Viewer". Частота работы цикла обусловлена заданным рабочим таймфреймом. Блок "Positions Viewer" проверяет, нет ли открытых позиций с заполненным комментарием (переменная "Comment" в торговом блоке). Для успешного нахождения позиций робота их необходимо помечать комментарием в каждом торговом блоке (на рис. комментарий имеет значение "Robot_Position"). Если комментарий совпадает, то стратегия закрывает открытую позицию (позиции) с незафиксированной прибылью 100 и более пунктов.



Кроме того, пользователь может работать с позициями с использованием другого метода: создать для позиции выделенную переменную, которая привязана к каждому блоку "Open at Market" в качестве выходной переменной. Тогда проверка будет производиться отдельно по каждой позиции. Такой метод подходит для несложных стратегий, однако блок "Positions Viewer" все же более эффективен в большинстве случаев.

С точки зрения метода и способа работы в цикле, логика блока "Positions Viewer" точно такая же, как у блока "Loop Viewer". Единственное различие в том, что "Loop Viewer" поддерживает больше типов массивов в качестве входных данных.

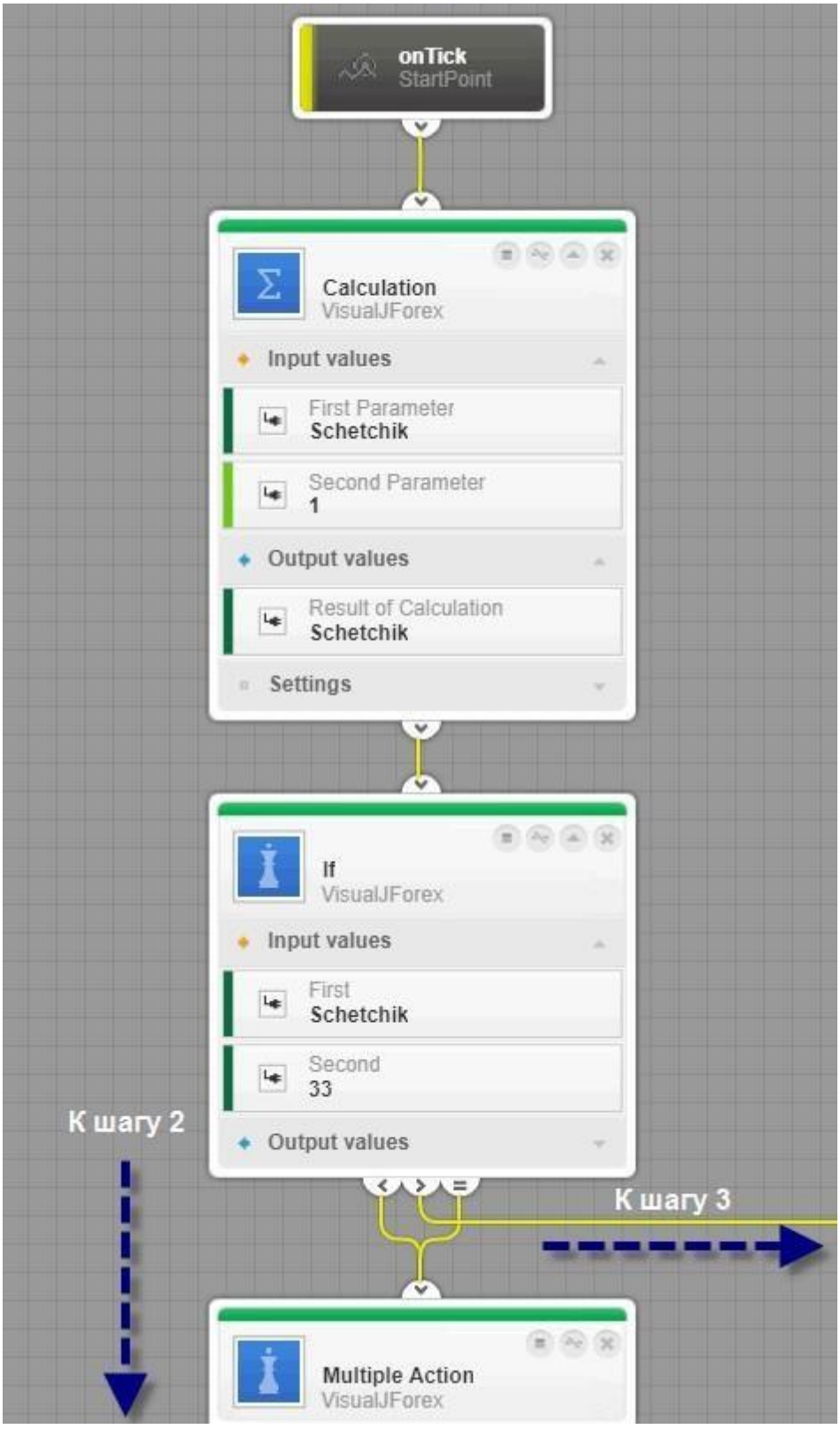
7.7. Как построить бары из тиков (со счетчиком)

Тиковые бары состоят из заданного количества тиков. Их легко построить на уровне торговой платформы JForex через меню настроек, но при программировании их в конструкторе Visual JForex применяется другой метод. Поскольку в конструкторе по умолчанию предоставлены таймфреймы от 10 сек до 1 мес, тиковые бары необходимо строить с помощью счетчика, который увеличивается по мере поступления новых тиков и обнуляется, когда заданное количество тиков достигнуто. Далее добавляется очередной набор условий для фиксации четырех цен бара: открытия, максимума, минимума, закрытия (OHLC).

Пример решения данной задачи:

1. Внедрение и настройка счетчика

В этом примере построим тиковый бар из 33-х тиков.



Для работы счетчика добавляем блок "Calculation" ("Вычисление"). Обратите внимание, что в качестве входной ("First Parameter") и выходной ("Result of Calculation") переменной используется та же – "Schetchik".

Очень важное условие содержится в блоке "IF", оно направляет поток в соответствии со значением счетчика. После того, как значение счетчика сравнялось с 33, стратегия переходит к шагу 2, а как только это значение превысит 33, стратегия переходит к выполнению шага 3.

2. Фиксация цен OHLC

На шаге 2 происходит вычленение цен OHLC. Для хранения OHLC используются, соответственно, четыре переменные.

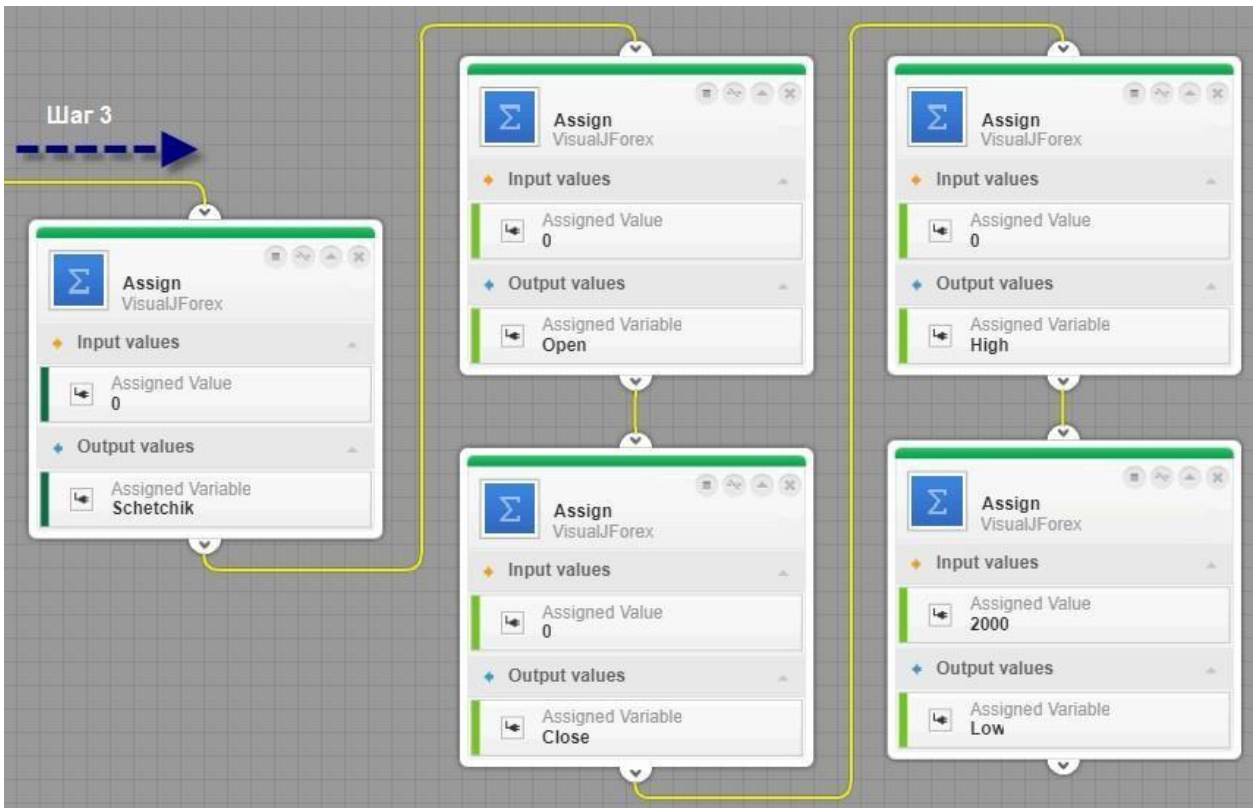
Уровень 1. Новым переменным Open, High, Low и Close еще не присвоены значения, но они будут использованы в блоках "IF" на уровне 2, чтобы поймать соответствующие цены по мере поступления тиков.

Уровень 2. При выполнении определенных условий каждая из переменных получает свою цену в таком порядке: Если счетчик равен 1, соответствующая цена тика заносится в переменную "Open". Если счетчик равен 33, соответствующая цена тика цена заносится в переменную "Close".

- Если текущая цена тика больше, чем значение переменной "High", то эта цена присваивается переменной "High". Если текущая цена тика меньше, чем значение переменной "Low", то эта цена присваивается переменной "Low". ○ Переменные "High" и "Low" будут обновляться с каждым новым полученным тиком до тех пор, пока счетчик не достигнет 33.

В этом примере после окончания цикла из 33 тиков стратегия запишет новые OHLC поверх старых, однако пользователь может сохранить предыдущие OHLC в отдельные переменные.

3. Конец цикла и сброс переменных OHLC



На 33-м тике переменные "Open", "High", "Close" сбрасываются на 0, "Low" сбрасывается на 2000 из-за условия "Last Tick.Tick Bid" < "Low". Если же сбросить "Low" на 0, то условие "Last Tick.Tick Bid" < "Low" лишится смысла, и определить минимальную цену не получится.

4. Использование OHLC в стратегии

Когда счетчик сравнивается с 33, текущие цены OHLC сохраняются в переменные "Open", "High", "Low" и "Close" с приставкой "Pred_" (предыдущий), так что в следующем цикле пользователь не теряет предыдущие данные.

Остальные условия стратегии можно подключить через блок "Multiple Action" ("Множественное действие").

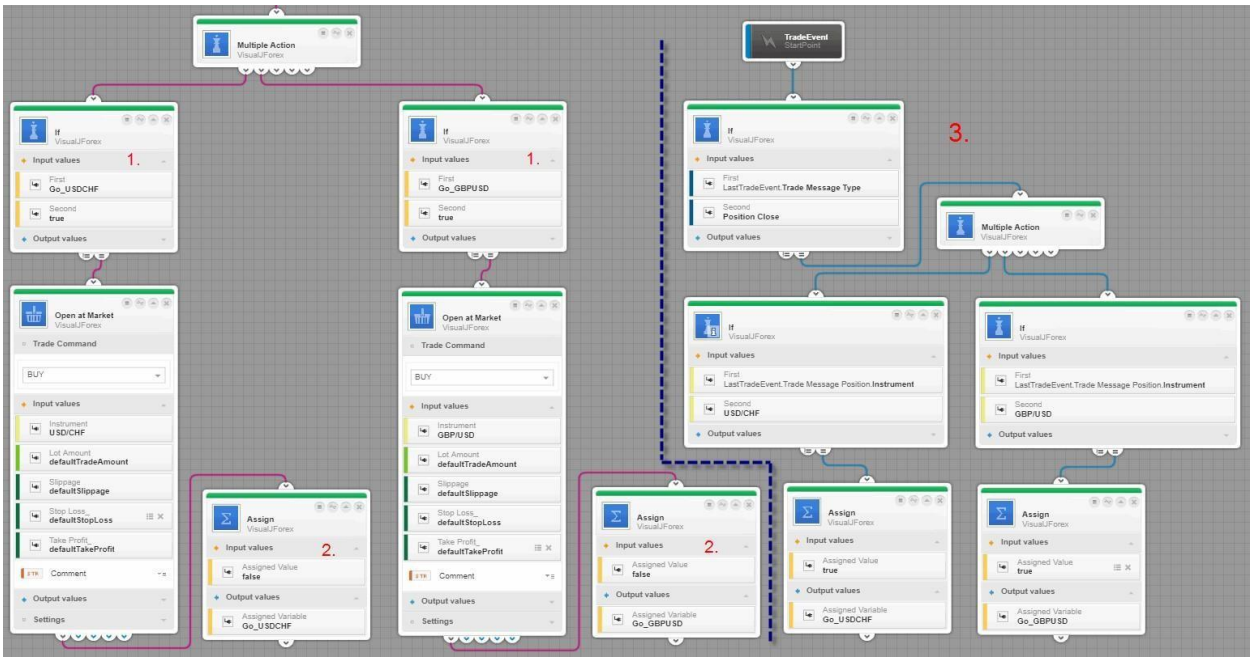
7.8. Как работать с логическими триггерами

Логический триггер – это переменная, созданная пользователем, которая принимает заданное значение при возникновении условия A и переключается на другое значение, когда наступает условие B. Предположим следующий сценарий:

В стратегии используются 2 инструмента, и пользователю необходимо действовать исходя из количества открытых позиций по каждому инструменту. Один из способов решения задачи – создание двух переменных, которые активируют или блокируют команду открытия позиции.

1. Стартовое значение такой переменной должно разрешать стратегии открыть позицию.
2. После открытия позиции переменная немедленно меняет свое значение, чтобы заблокировать следующую команду на открытие позиции.
3. Как только позиция закрывается, эта же переменная должна быть сброшена до начального значения.

Описанный процесс завязан всего на двух условиях, поэтому мы можем выбрать логическую переменную (типа "Boolean" – булева), которой присваивается одно из двух значений – "true" (истина) или "false" (ложь).



Слева: Переменным "Go_USDCHF" и "Go_GBPUSD" присвоены стартовые значения "true" (истина). При первом прохождении выполняются условия в первых блоках "IF", после чего открывается одна позиция по каждому инструменту. Далее переменным "Go_USDCHF" и "Go_GBPUSD" присваивается значение "false" (ложь) – №2 на рис. Когда стратегия заходит на следующий круг, условия в блоках "IF" (№1 на рис.) блокируют дальнейшее открытие позиций. В результате максимальное количество открытых позиций по каждой валютной паре не будет превышать 1.

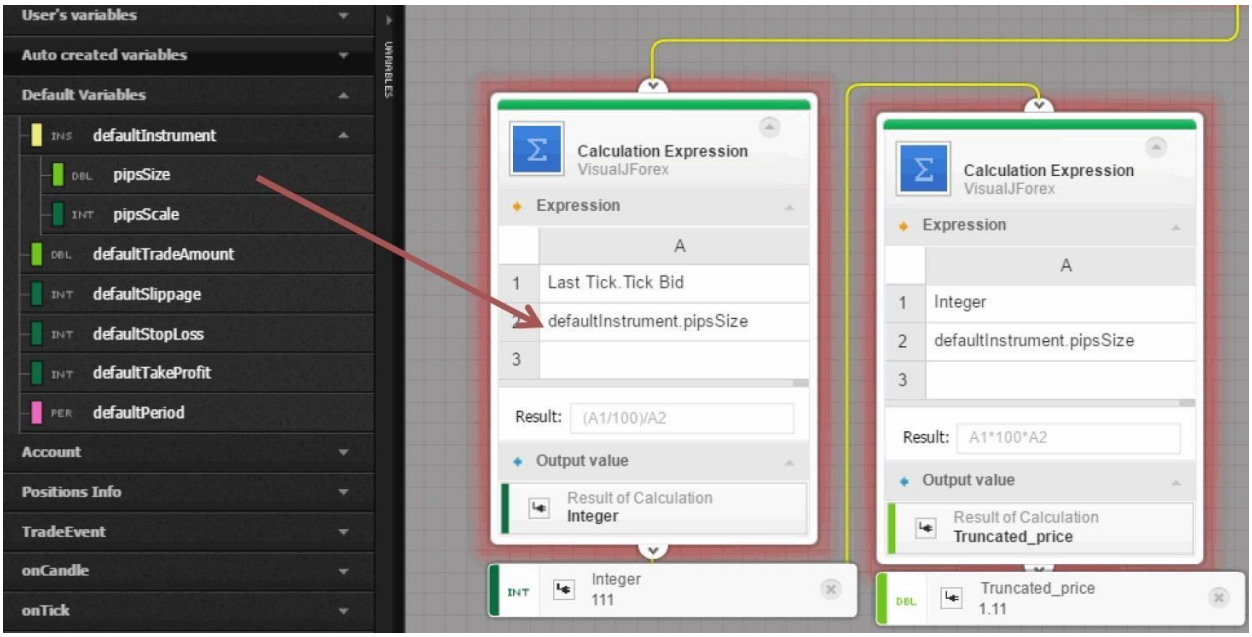
Справа: В данном случае используется стартовая точка "TradeEvent". Как только поступает сообщение "Position Close" ("Позиция закрыта"), стратегия проверяет валютную пару и присваивает триггеру значение "true" (истина) – так на следующем круге по этому инструменту может быть открыта позиция.

Логические триггеры уместно применять с условиями наподобие "старт-пауза". Пользователь должен точно определить, когда триггер включается и когда он сбрасывается к исходному значению. Более сложные процессы с тремя или четырьмя последовательностями можно реализовать с помощью вещественных ("Double") или целочисленных ("Integer") переменных.

7.9. Конверсия чисел в целые

Конверсия вещественного числа (напр., 2.56) в целое (напр., 2) означает отсечение дробной части. Это бывает необходимо для соответствия чисел правилам котирования в платформе либо для упрощения результата вычислений. Конверсия в целое не есть округление.

Для данной операции в Visual JForex специального блока нет, поэтому можно использовать следующий способ:

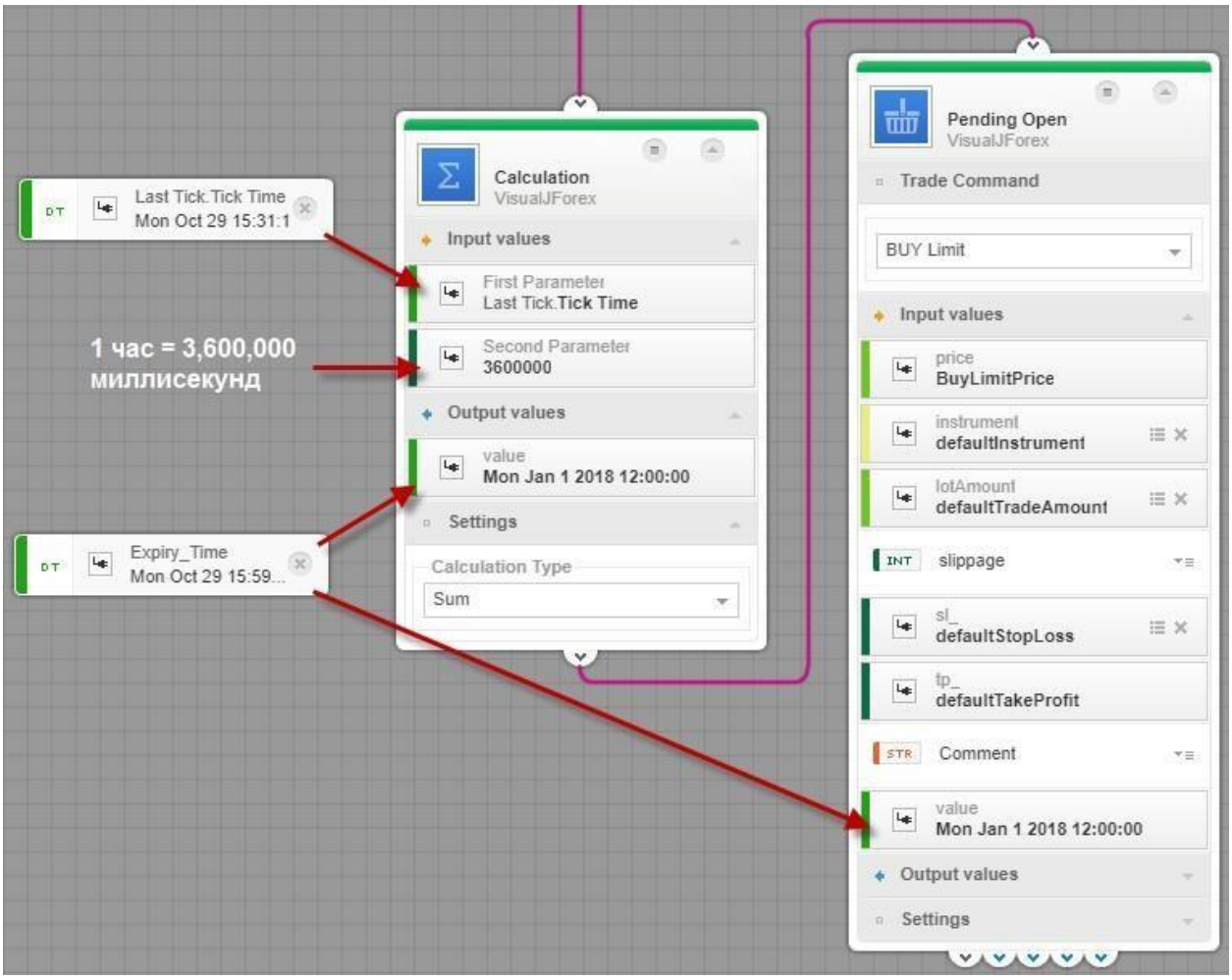


В этом примере цена последнего тика будет сокращена до двух знаков после точки. Первым делом тиковую цену умножим на 100 и сохраним в новой переменной типа "Integer" ("Целочисленная") и название ей тоже дадим "Integer". Сохранив результат в переменной типа "Integer", мы сразу же отсечем десятичные знаки. Второй шаг – разделить полученное целое число на 100, чтобы результат содержал два десятичных знака. Чтобы это решение работало на любом инструменте, используется переменная "pipsSize" ("Размер пункта"). Например, для EUR/USD "pipsSize" равен 0.0001, для USD/JPY – 0.01, для акций и индексов – 0.01.

7.10. Как использовать дату и время

На рис. ниже показано, как добавить время истечения срока лимитного ордера прямо перед его отправкой в платформу. После того, как условия входа выполняются, блок "Calculation" зафиксирует время тика и определит время истечения ордера, а затем отправит его в систему. Пока стратегия работает, время тика в блоке "Calculation" отображается в формате эпохи

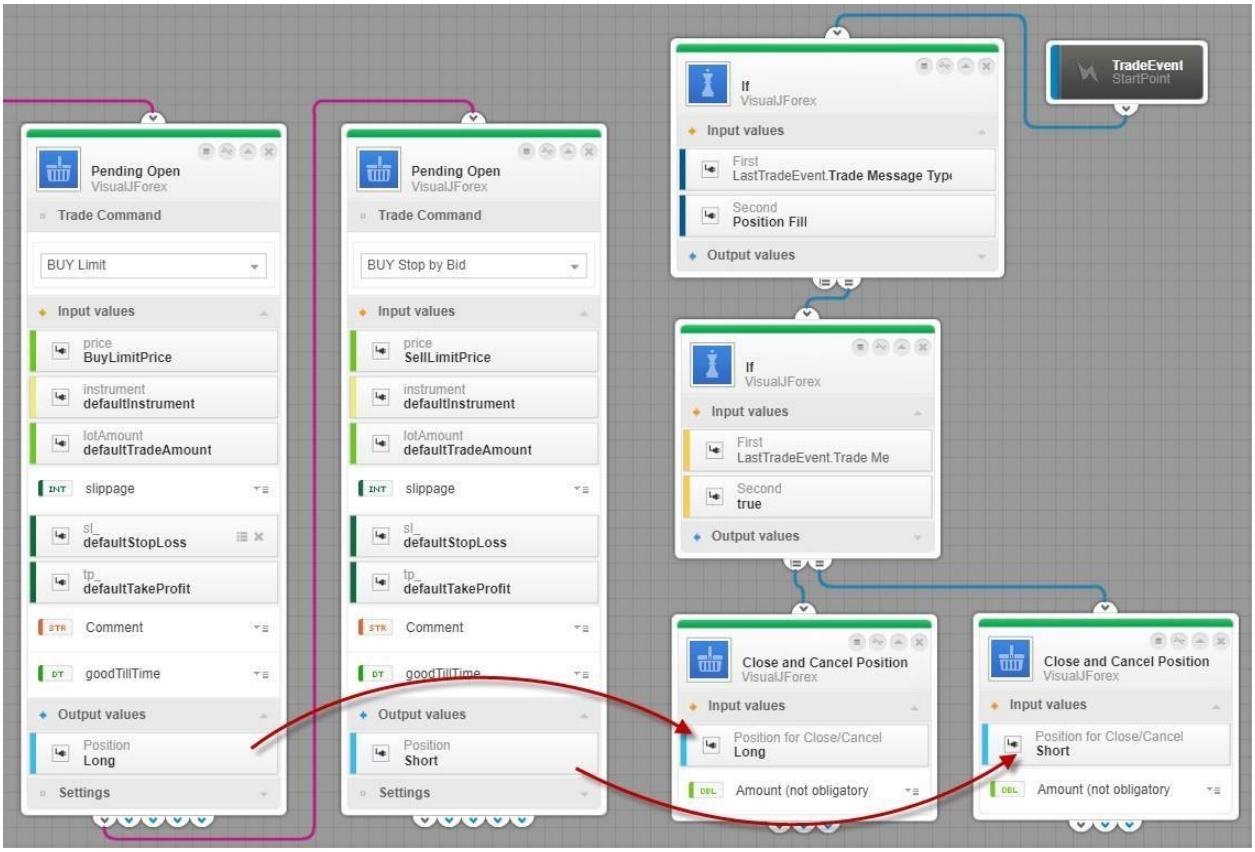
Linux. Если поместить переменную в рабочей области, то она преобразуется в обычный формат "Date and Time" ("Дата и время").



7.11. Как удалить отложенные ордера после исполнения другого

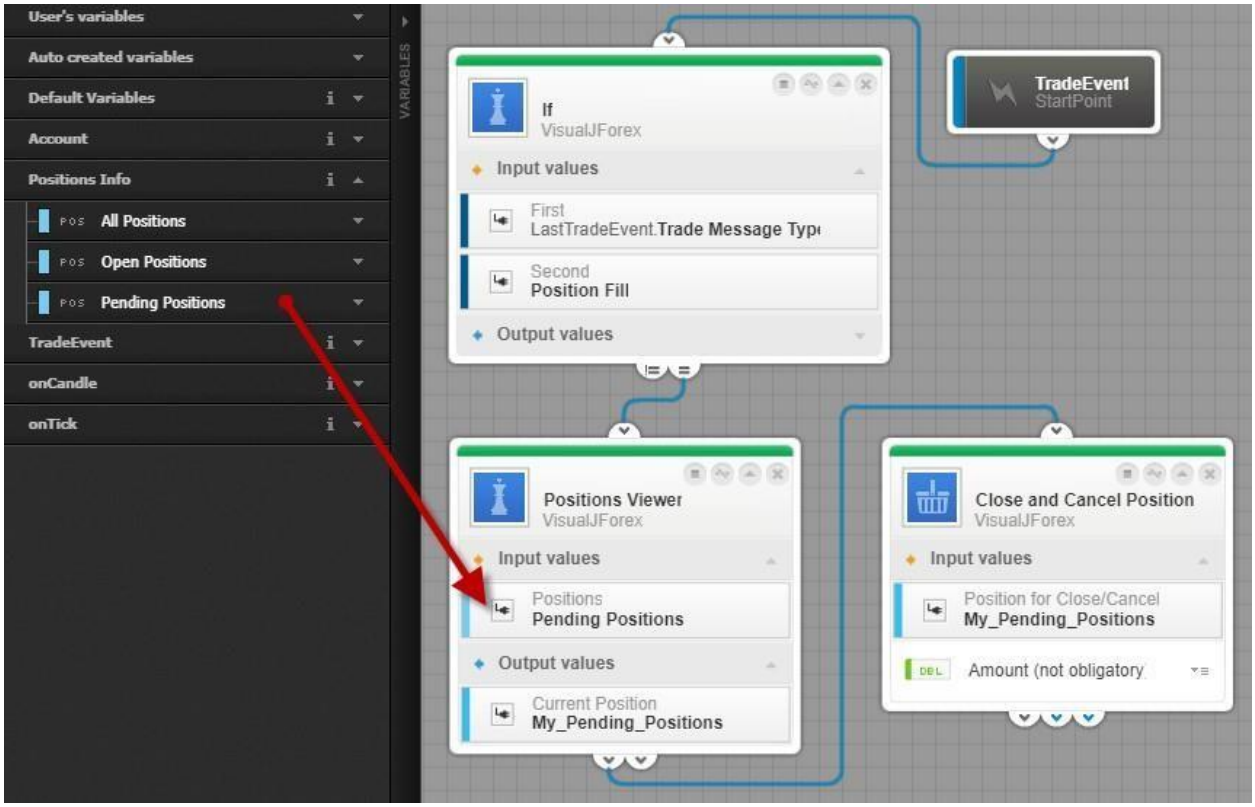
Если в стратегии используются отложенные ордера и после исполнения одного из них необходимо отменить остальные, то лучшее решение – использовать стартовую точку "TradeEvent", которая поможет зацепиться за событие "Fill" ("Исполнен") и сделать запрос на отмену остальных отложенных ордеров.

Если стратегия оперирует ограниченным количеством ордеров, то определить нужный ордер можно с помощью новой выходной переменной, добавляемой в торговый блок. Тогда, как только один из ордеров исполняется, стратегия переходит к отмене другого ордера, как показано на рис. в левой части внизу:



Пример из правой части рис. – выход из стартовой точки "TradeEvent": как только стратегия получает сообщение "Position Fill" ("Ордер исполнен"), происходит проверка направления открытой позиции (по выходной переменной из торгового блока), после чего противоположный ордер отменяется. В данном случае в блоке "IF" задается условие: если последняя сделка – это покупка ("LastTradeEvent.Trade Message Position.Position is Long" – буквально "Последнее торговое событие. Позиция из торгового сообщения. Позиция лонг"), то нужно закрыть ордер на продажу. Либо наоборот – открыта продажа, закрыть покупку.

Если же стратегия оперирует несколькими ордерами, то гораздо эффективнее работать с блоком "Positions Viewer", с его помощью моментально находятся позиции с состоянием "Opened" ("Открыт к исполнению").

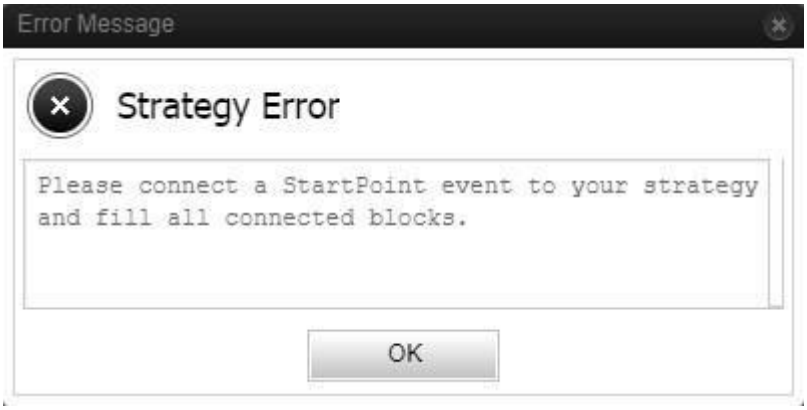


Обратите внимание, в блоке "Positions Viewer" в качестве входной переменной стоит "Pending Positions" ("Отложенные ордера").

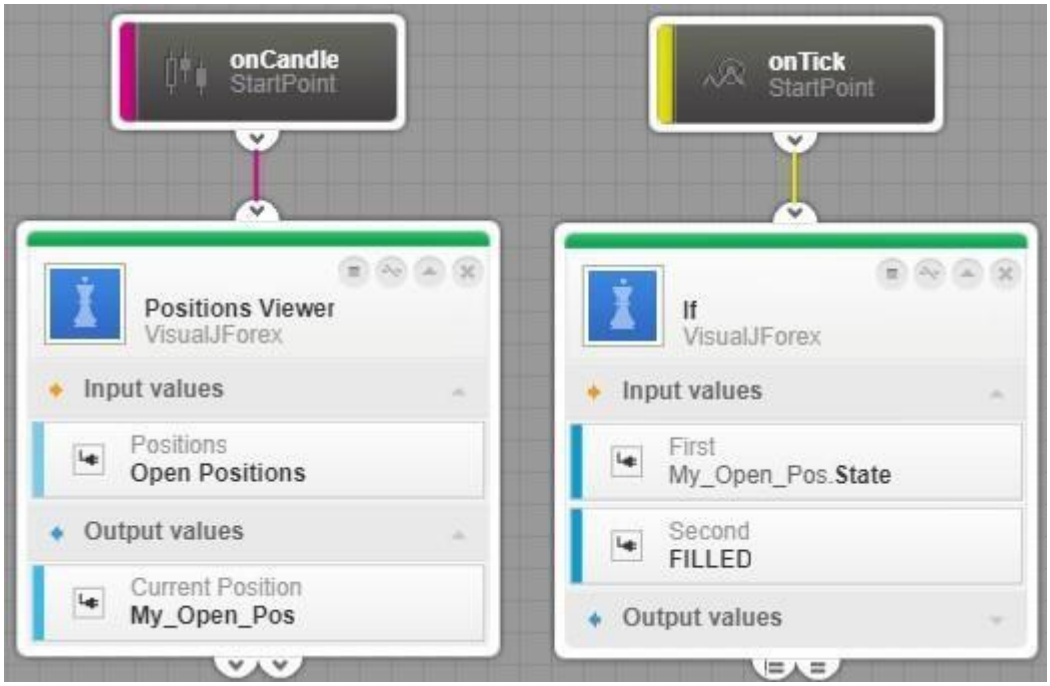
8. Сообщения об ошибках и устранение неполадок

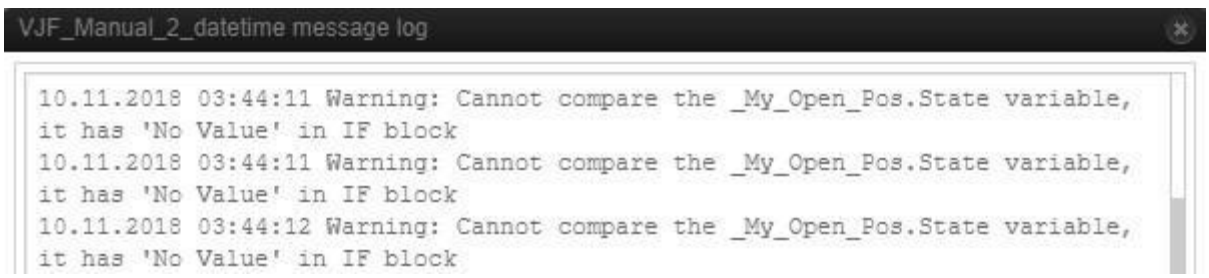
8.1. Ошибка подключения или неполный блок

Эта ошибка возникает, если пользователь пытается запустить стратегию, когда в блоке не хватает какого-либо параметра или когда не установлено соединение со стартовой точкой. Это же сообщение возникает при нажатии кнопки "Run" ("Выполнить"), когда файл стратегии еще не открыт.



8.2. Ошибки "No Value"

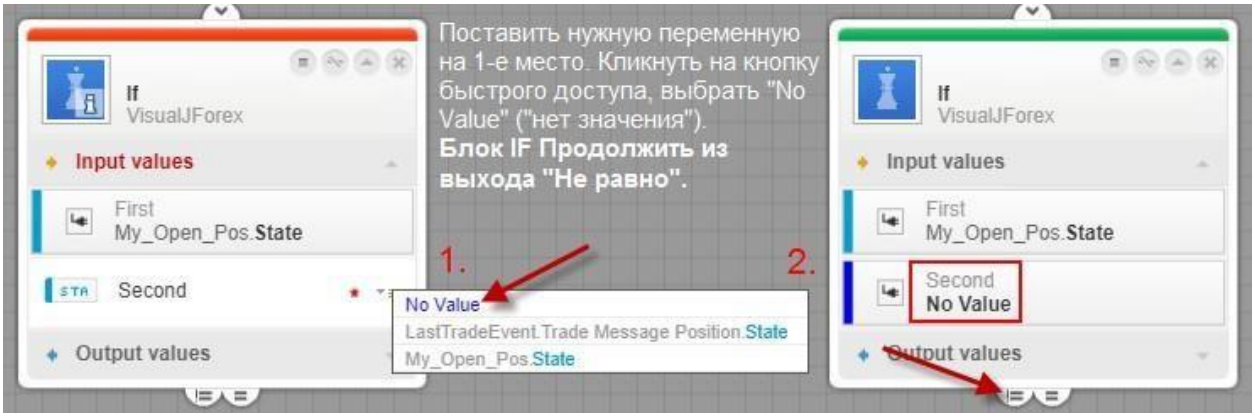




Ошибка типа "No Value" ("Нет значения") часто возникает, когда стратегия обращается к некой переменной до того, как той присваивается значение. В приведенном выше сообщении

об ошибке переменная "My_Open_Pos.State" ссылается на текущее состояние открытых позиций независимо от того, имеет ли переменная состояния некое значение или нет. Блок "IF" пытается проверить состояние еще не созданных позиций, возвращая таким образом этот тип ошибки. Во избежание подобной ситуации необходимо добавить еще один уровень проверки, где переменная "My_Open_Pos" получит некое значение и только потом будет проверено ее состояние.

Пример решения ниже.



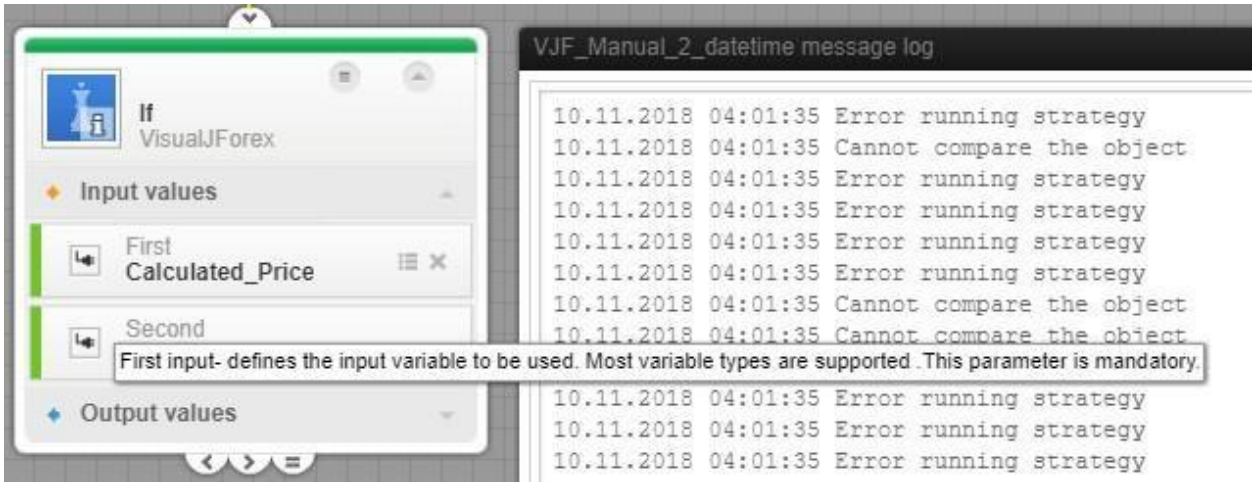
8.3. Правила именования переменных

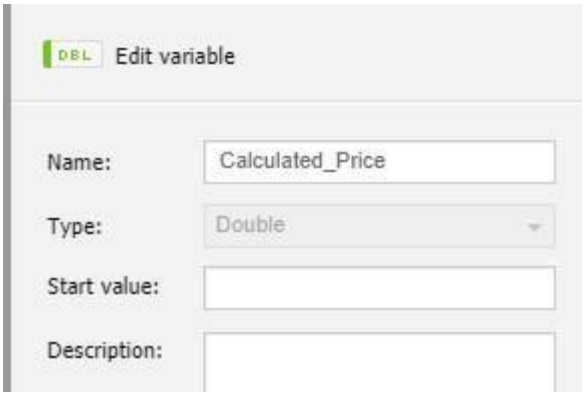
Данная ошибка возникает по той причине, что правилами именования переменных запрещены пробелы и символы, а также нельзя начинать имя переменной с цифры.

8.4. Отсутствует стартовое значение переменной

Данная ошибка возникает из-за отсутствия необходимого для работы переменной атрибута – ее стартового значения ("Start value"). Обычно он упускается на этапе создания переменной.

Подробнее о создании и настройке переменных см. раздел 6.





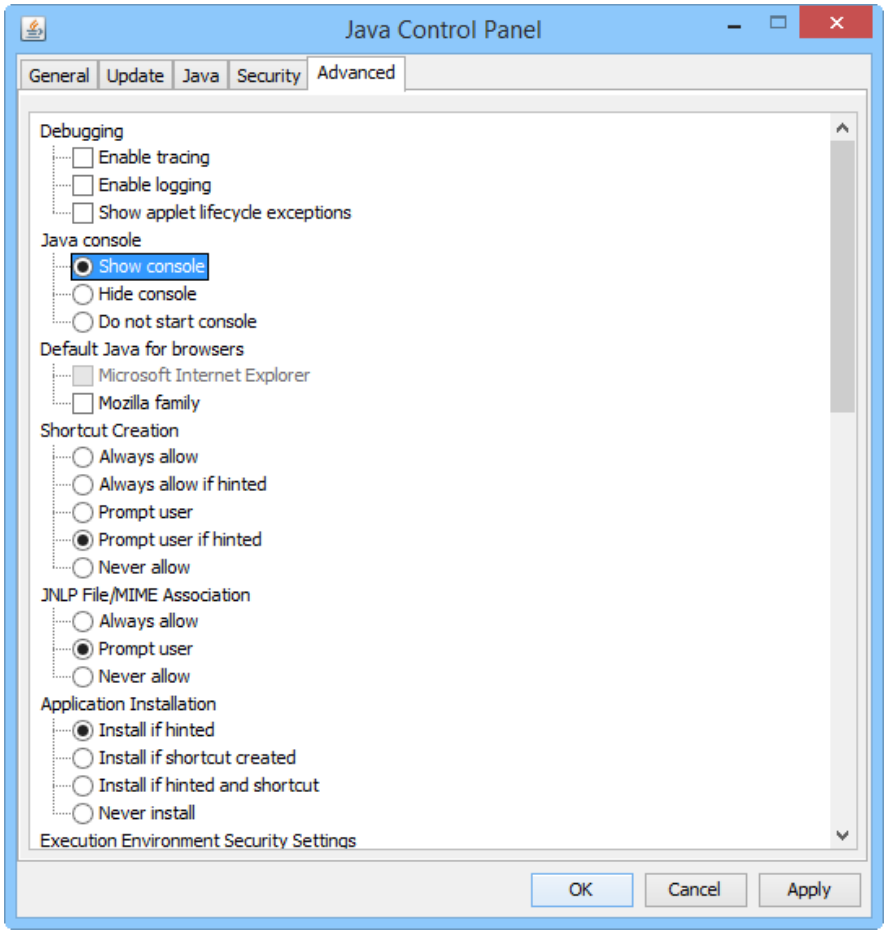
Стартовое значение – это первое значение, которое переменная получает при ее создании. Переменная "Calculated_price" создана без стартового значения, поэтому система будет считать, что у нее нет значения ("No Value"), следовательно – любая логическая или математическая операция становится невозможной.

В некоторых случаях (по аналогии с разделом 8.2) динамическая переменная может генерировать такую ошибку до момента получения некоего значения. Но после ее автоматического обновления (например, состояние позиции) ошибка исчезнет.

8.5. Продвинутые приемы отладки

Для отображения сообщений об ошибках Java пользователь может запустить Java-консоль.

Это поможет определить причины ошибок. Данная опция активируется из панели управления Java.



Когда процессор Visual JForex запущен, появится окно журнала, в котором регистрируются все сообщения о процессе в зависимости от выбранного уровня отслеживания.

Ошибки с заголовком "java.lang.NullPointerException" могут влиять на работу стратегии, а потому их стоит тщательно изучить.

Поддержка пользователей и дополнительная информация доступны на форуме Visual JForex.