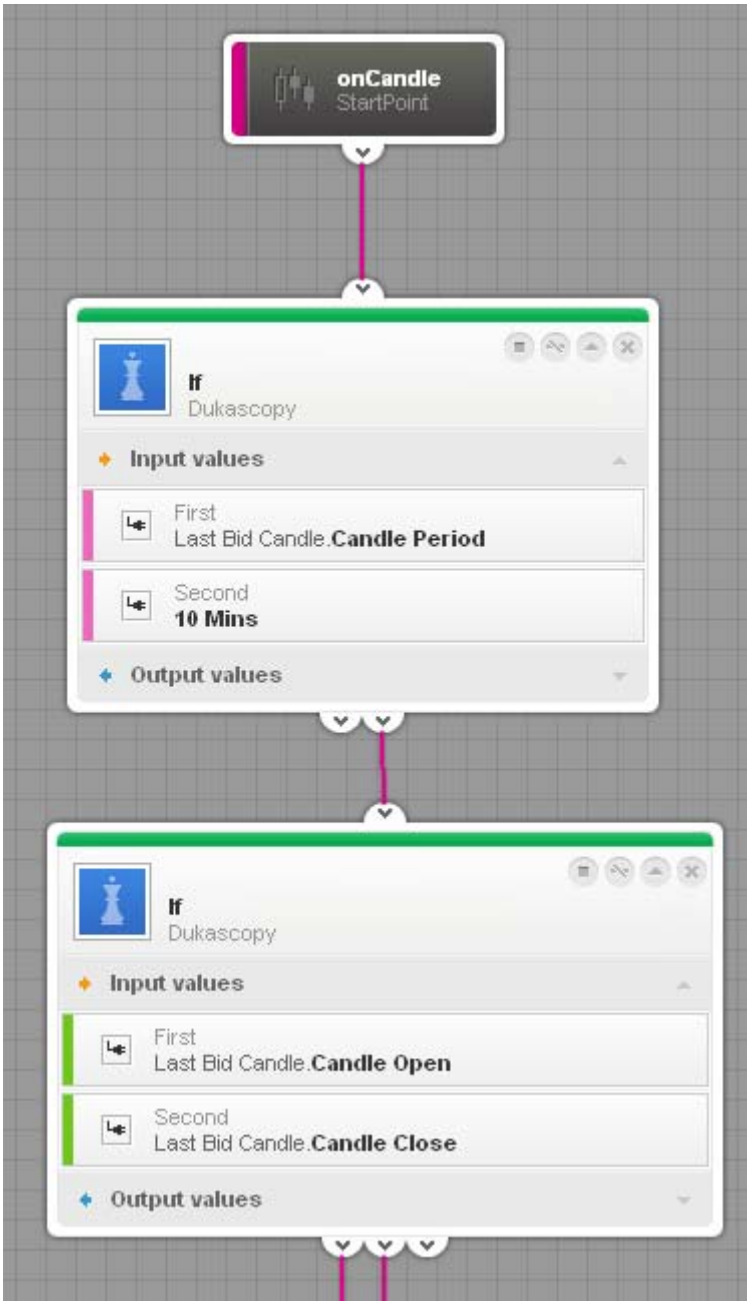
Начальные точки.

onCandle - Описание

onCandle - это начальная точка события, которое генерируется системой, когда Свеча готова. На это событие потока идут все свечи с определенными систему агрегации периоды, см. период для детали. onCandle - генерирует значения последний ASK Свечи и последний Bid свечи, так для каждого события onCandle значения этой системы переменной будет новой.

На следующем примере вы можете увидеть пример, как фильтр 10 минутной свечи, и как проверить, бычья или медвежья последняя свеча:

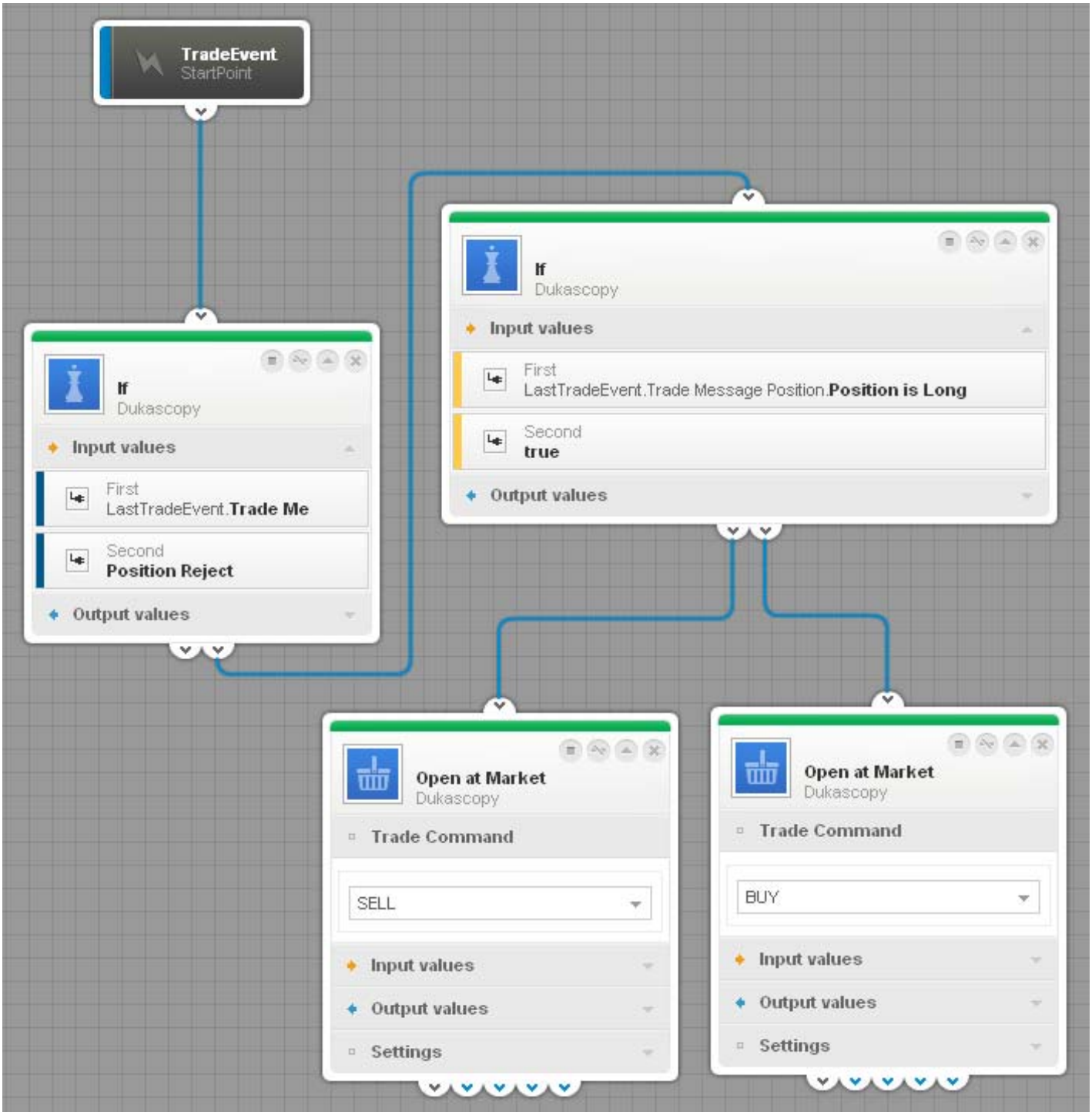
Советы: щелчок на левой вертикальной линии красочным блока, позволяет установить соответствующий цветовой поток цвета. В случае смешивания начальной потоков, поток становится черного цвета.



TradeEvent - Описание

TradeEvent - это начальная точка события, которое генерируется системой, когда новые торговые события, поступают от сервера. Результат **TradeEvent** – последняя переменная TradeEvent которая обновляет данные.
На следующем простом примере, вы можете увидеть, как проверка отвергается на сервере и повтор входа в рынок:

Советы: щелчок на левой вертикальной линии красочным блока, позволяет установить соответствующий цветовой поток цвета. В случае смешивания начальной потоков, поток становится черного цвета.



Component (Компоненты).

Info – информационные блоки.

Get Historical Candle (получить историческую свечу).

Определение блока:

"**Get Historical Candle**" - возвращает пользователя к свече, которая была.

Описание параметров:

Параметр **Instrument** - выберите необходимую валютную пару. Заполнить это поле обязательно.

Параметр **Candle Period** - Выбирите необходимый период времени, за который вы хотите получить свечу. Заполнить это поле обязательно.

Параметр **OfferSide** - определяет сторону Свечи. Заполнить это поле обязательно.

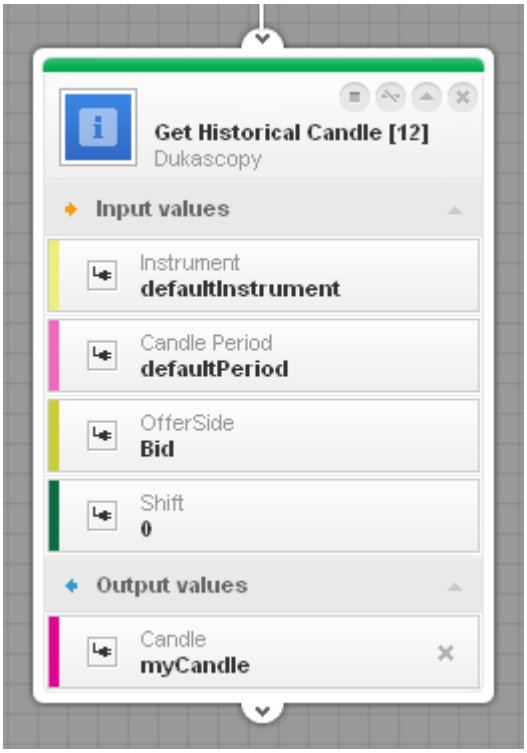
Параметр **Shift** (Сдвиг) Целочисленный параметр - определяет расстояние в целых числах, как далеко свеча сместилась от текущего. Текущая Свеча индекса всегда 0. Заполнить это поле обязательно.

Результат работы блока - результат на блок работы вывод на свечу, которую пользователь получает.

Поток: - данный блок закончит свою работу и поток закончится. Блок имеет только один поток.

Основные Определения:

Блок возвращает необходимые Свеча/бар на выходе. Блок получает Candle Period, Instrument и OfferSide и Shift в качестве входных параметров. Вход параметра сдвига работает в следующем режиме: 0 - текущая свеча, 1 - предыдущая свеча е.т.АР.



Get Historical Candles (получить исторические свечи).

Определение блока:

"**Get Historical Candles**" - возвращает пользователя в массив свечей.

Описание параметров:

Параметр **Instrument** - выберите необходимую валютную пару. Заполнить это поле обязательно.

Параметр **Period** - выбрать необходимый период времени за который вы хотите получить свечу. Заполнить это поле обязательно.

Параметр **OfferSide** - определяет сторону свечи Bid или Ask. Заполнить это поле обязательно.

Параметр **Shift** (Сдвиг) целочисленный параметр - определяет расстояние в целых числах, как далеко Свеча сместилась от текущей. Текущая Свеча всегда 0. Заполнить это поле обязательно.

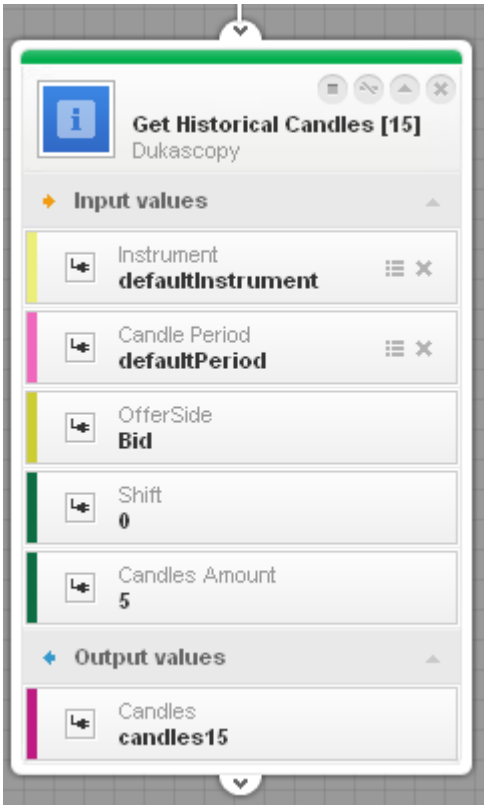
Параметр **Candles Amount** - целочисленный параметра, он определяет количество свечей в целых числах. Сколько свечей должно быть получено с помощью блока. Заполнить это поле обязательно.

Candles в исходных значениях - результат работы блока - вывод массива свечей, которые получает пользователь.

Поток: - данный блок закончит свою работу и поток закончится. Блок имеет только один поток.

Основные Определения:

На выходе блок возвращает необходимое множество свечей/баров. Блок получает **Candle Period** (период свечи), **Instrument** (инструмент) и **OfferSide** (сторона свечи Bid или Ask) , **Shift** (сдвиг) и **Candles Amount** (размер свечи) в качестве входных параметров. Ввод параметра сдвига работает в следующем режиме: 0 - текущая свеча, 1 - предыдущая свеча е.т.АР. Этот блок обычно сочетают с **Loop Viewer**, из-за того, что **Loop Viewer** может управлять множеством свечей.



Logical - Логические блоки.

IF

if - используется для логического сравнения одного значения с другим значением, в случае комплексных значений результат сравнения может быть "равен" или "не равно", в случае числовых значений результатов сравнения может быть "равно", "меньше" или "больше".

Основные определения: типы сравниваемых значений должны быть одни и те же типы в случае нечисловых типов, а в случае числовых типов, типы могут быть любыми из **Double** (вещественный), **Date and Time(Long)** (Дата и время) и/или **Integer** (целое число).

Positions Viewer

Positions Viewer - этот блок используется для выполнения возможности пройти через множество позиций (ордеров)

Описание параметров:

Параметр **Position** - определяет множество позиций, используемых, чтобы выполнить итерацию (повторное применение математической операции в серии аналогичных операций, производимых для получения результата) внутри "**Positions Viewer**". Заполните это поле обязательно.

Параметр **Current Position** - переменная position, в котором блок сохраняет текущую итерацию позиции. Заполнить это поле обязательно.

Поток: **CurrentPosition** - поток, который активируется каждый раз, когда блок выполняет итерации.

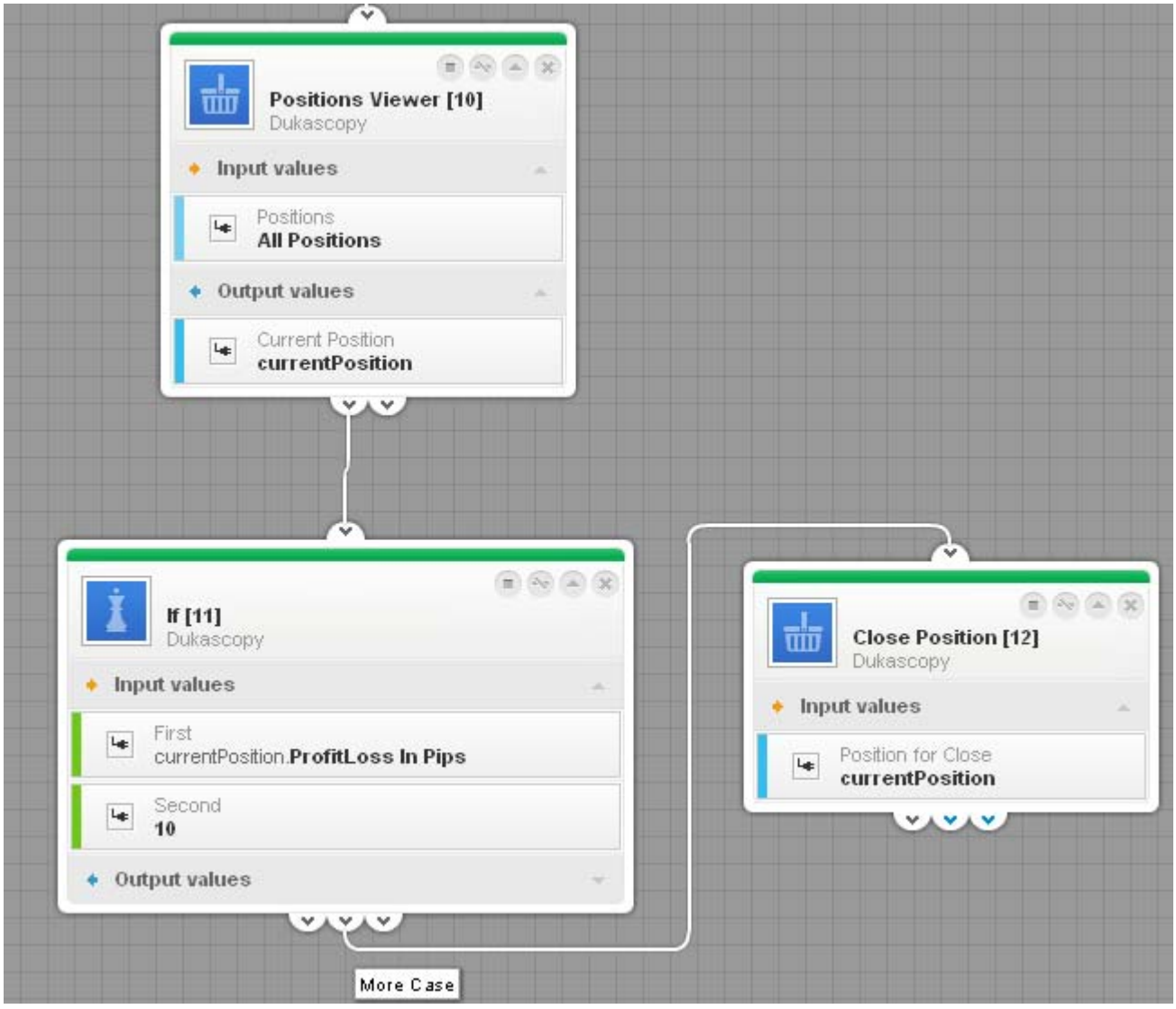
CycleEnd - этот поток происходит только тогда, когда блок проходит все позиции коллекции.

Основные определения:

"**Positions Viewer**" используется для обработки позиций(ордеров) в коллекции (массива) позиций. Блок имеет обязательное поле - переменная массив позиций и одна исходящая переменная - текущая позиция.

"**Positions Viewer**" имеет 2 исходящих потока, первый из них используется для текущей позиции, и второй-когда коллекция закончится.

Для того, чтобы понять, зачем этот блок нужен - приведем пример. Например, мы имеем следующую стратегию, где мы должны закрыть все позиции, где прибыль больше чем 10 пунктов. Для этого случая - " Positions Viewer " будет лучший вариант. Мы перетащим "все позиции" переменную требуемого входного блока. Эта переменная содержит все позиции в настоящее время. В Выходное значение - мы создаем "currentPosition" переменной, которой оно будет принимает значение текущей итерации из "все позиции". На первом потоке мы создаем блок "если" - которые проверяют "currentPosition" прибыль и сравнивает её с 10 пунктов, в случае большей прибыли мы закрываем "currentPosition" с "Close Position". Затем " Positions Viewer" переходит на следующую позицию, и она продолжается, а если проверка будет идти не по всем позициям. См. изображение:



Loop Viewer

"**Loop Viewer**" - этот блок используется для описания условия прохода через множество любого массива или коллекции, это может быть позиций, свечи и сигналы.

Описание параметров:

Массив данных параметра - определяет набор любого массива, например, позиции (ордера), свечи или сигналы, используемых для повторения внутри "Loop Viewer". Заполните это поле обязательно.

Параметр текущий указатель - используется для сохранения текущего указателя сбор, может быть позиция, Свеча или уровень сигнала. Заполните это поле обязательное.

Текущий параметр Position - переменная позицию, какой блок сохраняет текущий итерированным установки. Заполните это поле обязательное.

Текущий параметр Свеча - переменная Свеча, к которому блок сохраняет текущий итерационный свечи. Заполните это поле обязательное.

Текущий параметр Сигнал - переменная позицию, какой блок сохраняет текущий повторный сигнал. Заполните это поле обязательное.

Потоки: CurrentPointer - поток, который активирует каждый раз, когда блок итераций.

CycleEnd - этот поток происходит только тогда, когда блок перебирает все элементы коллекции.

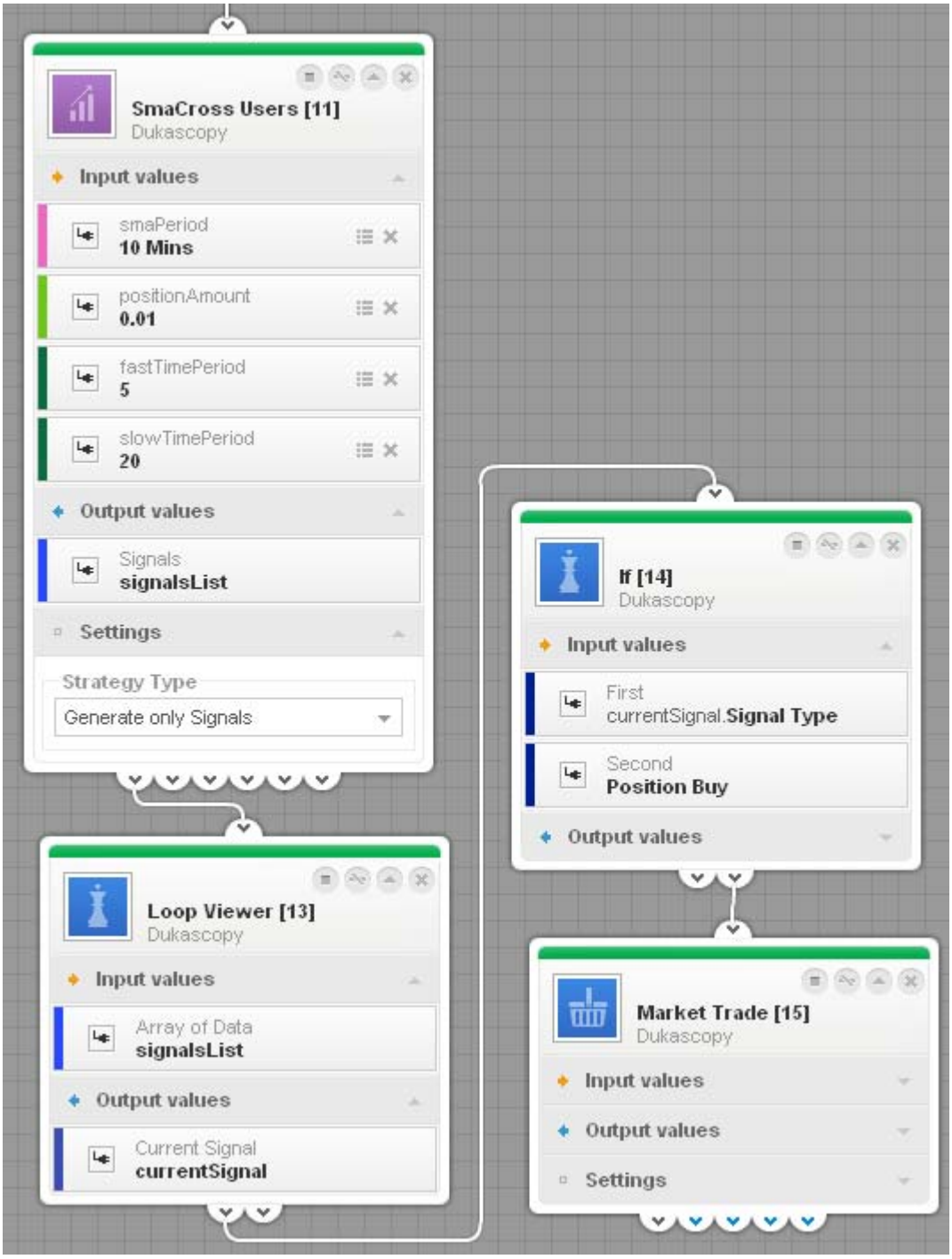
Главное Определение:

"Loop Viewer" используется для обработки коллекцию (массив) позиции, свечей и сигналов. Блок обязательное поле - переменный элемент массива и один выходной переменной - сигнал, свеча или позиция.

"Loop Viewer" имеет 2 выходных потоков, первый используется для текущего элемента и второй, когда массив ордеров закончился.

Лучший способ понять, зачем этот блок нужен - рассмотрим пример. Например, у нас есть следующая стратегия, где нам нужно проверить все торговые сигналы, которые мы получаем от

компонента стратегии. В этом случае - "**Loop Viewer**" будет лучшим выходом. Мы перетаскиваем переменную "signalsList», созданный для хранения сигналов по стратегии SmaCross его к нужному входу блока. Эта переменная содержит все сигналы в настоящее время, возвращаемые внутренней стратегии. В выходные величины - мы создаем переменную "currentSignal", который это занимает значение текущей итерации от "signalsList". На первом потоке мы создаем блок "if" - что проверить тип "currentSignal" и сравнить его с "Position Купить», в случае равенства, мы откроем Открыть на рынке (рыночной торговли). Тогда "Loop Viewer" идет к следующему сигналу (если она существует), и она будет продолжать, пока не проверит все сигналы. См изображение.



Multiple Action

" Multiple Action " - основной функцией этого блока разделена на несколько потоков алгоритмов.

Описание параметров:

Первый поток в порядок.

Второй поток в порядок.

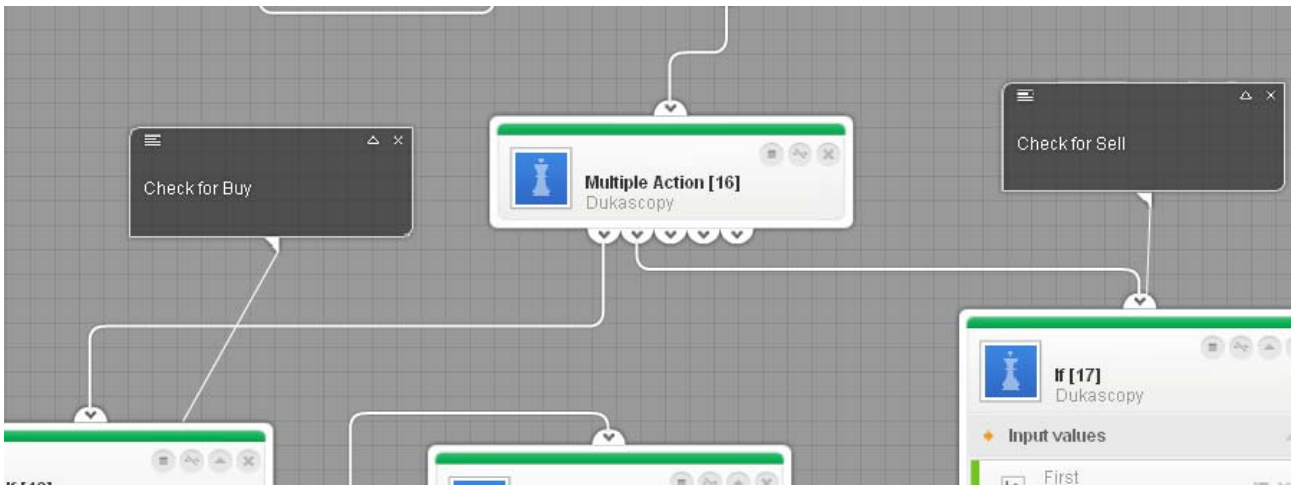
Третий поток в порядок.

Четвертый поток в порядок.

Пятый поток в порядок.

Основные Определения:

"Несколько действий" распределить 5 параллельных потоков, от 1 входного потока. Выходной поток порядок запуска, зависит от потока номер заказа. Первый поток будет обрабатываться первым, затем второй и так далее. Это очень полезно, если вам нужно совершить несколько самостоятельных действий, см. рисунок:



Компонент Mathematical – Математические блоки.

Assign (назначить)

Определение блока:

Назначение - используется, чтобы сделать простые задания из переменной/значение переменной.

Описание параметров:

Первый параметр - определяет **Assigned Value** (присвоить значение). Заполнить это поле обязательно. Это может любое значение или переменная, но она должна быть того же типа, что выходное значение.

Assigned Variable (присвоенная переменная) - здесь определить переменную, которая принимает новое присвоенное значение. Заполнить это поле обязательным.

Потоки: - данный блок закончит свою работу и поток закончится. Блок имеет только один поток.

Calculation (расчет)

Определение блока:

Calculation - используется, чтобы сделать примитивные расчеты, такие как сумма, вычитание, деление и умножение, между двумя входными аргументами.

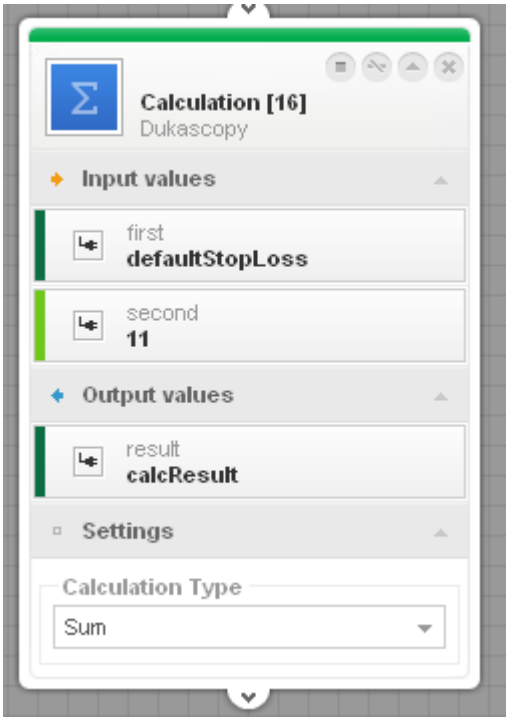
Описание параметров:

Первый параметр - определяет числовое (целое число, вещественное число или дату и время(давно)) (**Integer**/целое число, **Double**/ вещественное или **Date and Time**/дата и время) или значение числовой переменной. Заполнить это поле обязательно.

Второй параметр - определяет числовое (целое число, вещественное число или дату и время(давно)) или значение числовой переменной. Заполнить это поле обязательно.

VariableResult расчета блок возвращает - результат на блок работы вывод переменной числовой (целое число, вещественное число или дату и время(давно)) типа. Заполнить это поле обязательным.

Потоки: - данный блок закончит свою работу и поток закончится. Блок имеет только один поток.



Calculation Expression (выражение для расчета)

Определение блока: выражение для расчета - используется для выполнения более сложных арифметических вычислений и формул, например, это можно сделать в Excel. Каждый блок имеет результат поле, где пользователь определяет конечную формулу расчета в таблицу/строку в Excel именования.

Описание параметров:

Результат вычисления блока возвращает - результат на блок работы вывод переменной числовой (целое число, вещественное число или дату и время(давно)) типа. Заполнить это поле обязательным.

Потоки: - данный блок закончит свою работу и поток закончится. Блок имеет только один поток.

Основные Определения:

Блок возвращает результат вычислений на выход. Блок получает математическую формулу в "результат" поле.

Блок поддерживает следующие операции:

"-" Substration.

"+" Дополнение.

"/" Деление.

"*" Умножение.

"мощность (x, Y)" возвращает значение x В степени e.

"%"- Модуль.

"грех(x)" возвращает синус инструкциям expr, где expr-это считается в радианах.

"потому что(x)" возвращает косинус инструкциям expr, где expr-это в радианах.

"Тан(x)" возвращает тангенс инструкциям expr, где expr-это считается в CA.

"ЛН(x)" возвращает логарифм по основанию e выражение из. Если выражение не в домене, произойдет ошибка.

"атан(x)' возвращает угол в радианах между-пи/2 и пи/2, тангенс которого является выражение.

"Атос(x)" возвращает угол в радианах между 0 и Pi, косинус которого равен выражение.

"асин(x)" возвращает угол в радианах между-пи/2 и пи/2, синус которого равен выражение.

"exp(x)" возвращает число Эйлера e, возведенное в степень числа двойной точности.

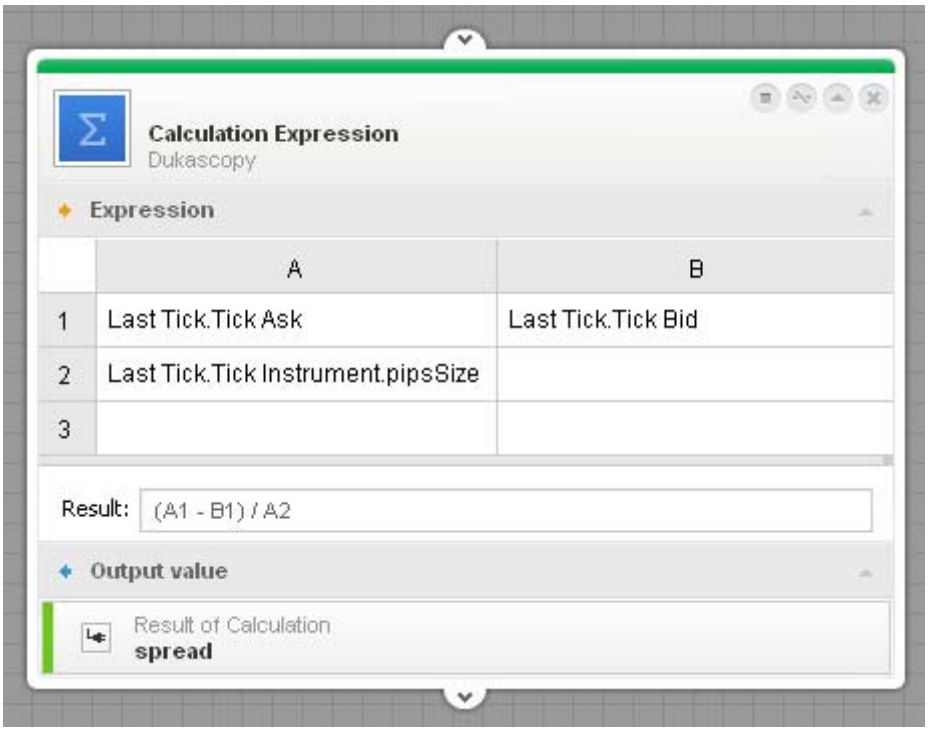
"мин(X, X)" возвращает минимальное между выражение1 и выражение2.

"Макс(X, X)" возвращает максимальное между выражение1 и выражение2.

"функция sqrt(x)" возвращает квадратный корень выражение. Если выражение не в домене, произойдет ошибка.

Нажатие на столбец или строку можно добавить новый столбец/строку или удалить его.

Смотрите следующий пример, где в текущий спред от последнего тика рассчитывается:



Компоненты Trading

Close and Cancel Position (закрыть и удалить позицию).

" **Close and Cancel Position** " - осуществляет торговую позицию или заказ будет отменен или закрыт.

Описание параметров:

Position - позиция/ордер переменных, которые должны быть закрыты или отменены. Обязательное поле.

Сумма - не обязательное поле, используется в случае частичного закрытия. Если Вы не хотите частично закрыть, установите его пустым.

Размер лота не должен быть меньше, чем минимальный размер лота для позиции документа.

Поток: выполнение - это блок завершил свою работу и поток закончится.

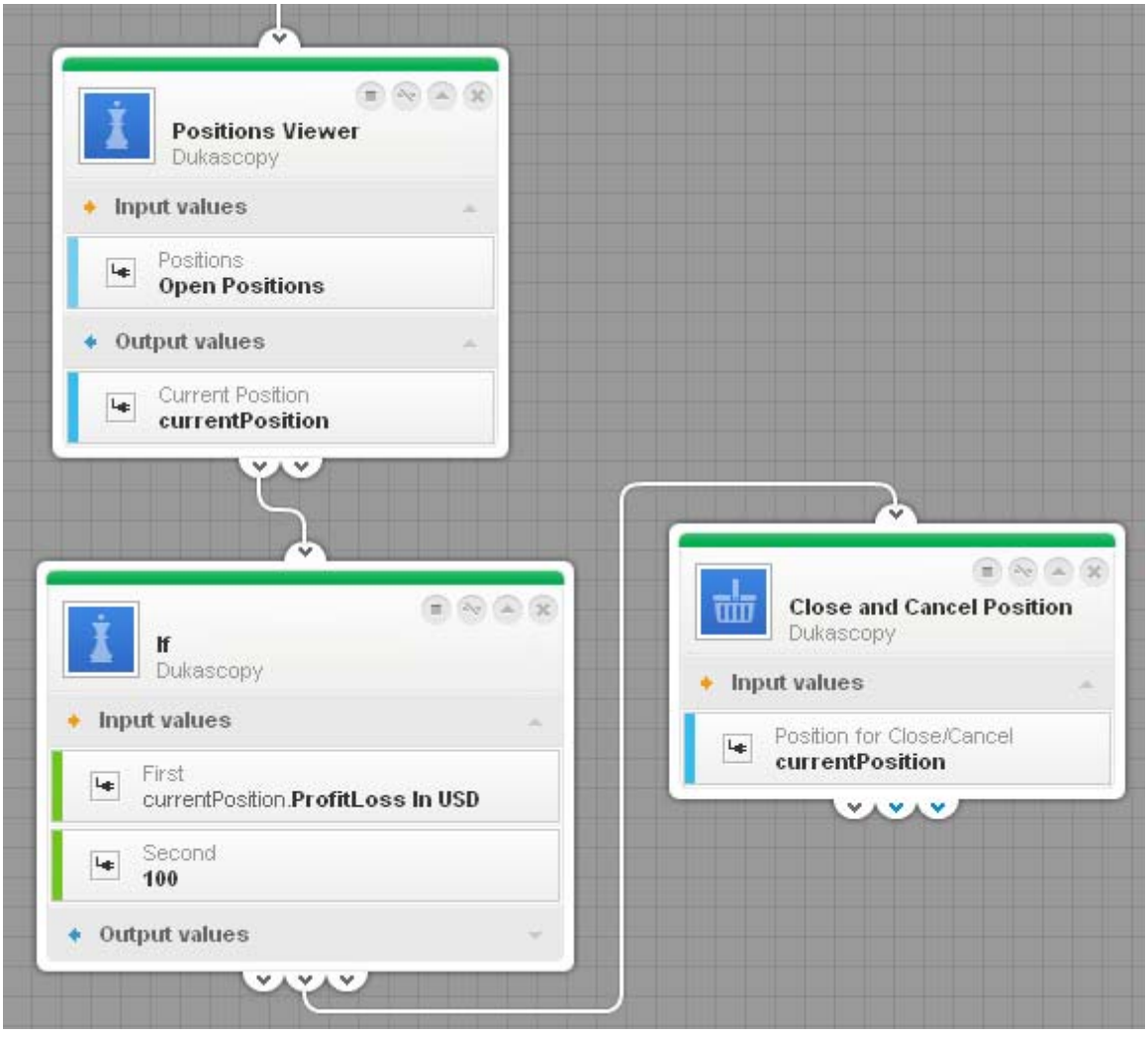
Позицию закрыть ок - этот поток бывает только в случае установки/заказ успешно закрыть

Позиция отвергать - это наплыв бывает только в случае установки/отклонение заказа на следующих

Основные Определения:

" **Close and Cancel Position** " используется для закрытия позиции или отменить отложенный ордер. Обязательно для этого блока поле " **Position for Close** " - пользователь должен заполнить его с переменной позиции. Эта переменная не должна быть "нулем". Обычно это блоки используются с **Position Viewer** и **Loop Viewer**, которая позволяет пройти все текущие позиции. " **Close and Cancel Position** " - есть 2 события потоков - по успешной позиции закрыть и отклонить мероприятие. Мероприятие будет обработан Visual jforex от только один раз. Потоки события выделяются синим цветом.

В следующем примере показано, как закрыть все прибыльные позиции, прибыль которых составляет более 100$:



Custom TrailingStop (Настраиваемый трейлинг-стоп).

" **Custom TrailingStop** " - выполняет позицию Трейлинг, иначе как в Дукаскопи платформы. Стоп лосс трейлинг выполняет на каждом рынке движение цены к прибыли.

Описание параметров:

Position - позиция/ордер переменной, что стоп-лосс должен быть прицепной. Обязательное поле

Скользящий шаг - Целочисленное значение (в пунктах) - шаг трейлинга значение, должно быть больше, чем 10. Обязательное поле

Стоп-лосс - Целочисленное значение - количество пунктов, на которое должен быть стоп лосс итерированных от цены открытия. Обязательное поле

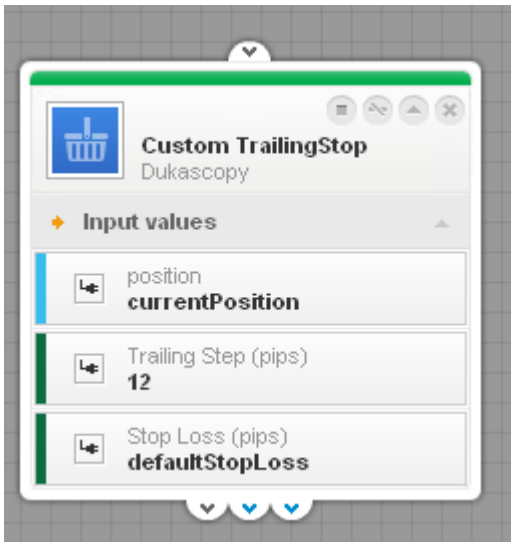
Потоки: выполнение - это блок завершил свою работу и поток закончится.

Изменение позиции ок - этот поток происходит только в случае расположения/порядок успешного изменения

Изменение позиции отклонение - это наплыв бывает только в случае установки/отказа в выполнении распоряжения на изменение позиции

Основные Определения:

" **Custom TrailingStop** " используется для установки стоп-лосса и подтягивать стоп лосс установленными во входных параметрах позиция. Трейлинг функциональность используется для всякий раз, когда блок активен. Это изменение стоп-лосса с периодом Трейлинг шаг. Шаг трейлинга значение-целое число, и это поле обязательное. В зависимости от набора параметров Цена/пунктов - пользователь может определить стоп-лосс: либо в цене (Double значение типа) или в пунктах (int значение типа)". "Установить стоп-лосс" - имеет 2 входа, который он/она должен заполнить обязательные это - позиция переменной и стоп-лосса. Позиция переменная не должна быть "нулем". По умолчанию значение стоп-лосса устанавливается с "defaultStopLoss" переменной. Иногда это блоки используются с позиции зрителя и зрителя петли, которая позволяет пройти все текущие позиции. "Установить стоп-лосс" - имеет 2 события потоков - по успешной позиции изменения и отклонить мероприятие. Мероприятие будет обработан Visual jforex от только один раз. Потоки события выделяются синим цветом. Например, если Трейлинг-стоп устанавливается 5, что означает, что уровень стопа будет меняться каждые 5 пипсов.



Open at Market (Открыть ордер по рыночной цене).

"**Open at Market**" - выступает на рынке открыть подчиняться системе.

Описание параметров:

Параметр **Instrument** - выберите необходимую валютную пару. Заполнить это поле обязательно.

Параметр **Lot Amount** - определяет размер лота в валюте счета, Тип Double. 1 - это 1 млн., минимальное значение = 0.001. Заполнить это поле обязательно.

Параметр **Slippage** - определяет проскальзывание в целых числах. Заполнить это поле обязательно. Значение проскальзывания означает следующее:

* если отрицательная, то значение по умолчанию равно 5 пунктам

* если двойной.isNaN (проскальзывание) = True тогда нет проскальзывания

* в противном случае, проскальзывание задается в пунктах, вы должны пройти 1, не 0.0001

Параметр **Stop Loss** - определяет количество пунктов стоп-лосса от цены открытия.

Параметр **Take Profit** - определяет количество пунктов тейк профит от цены открытия.

Комментарии - строковый параметр определяет текст, комментарий, который будет сохранен в ордере.

Параметр **Stop Loss Double** - определяет цену стоп-лосса. Цена должна быть кратна 0.1 пунктов или ордер будет отклонен.

Параметр **Take Profit Double** - определяет цену тейк-профита. Цена должна быть кратна 0.1 пунктов или ордер будет отклонен.

Позиция блок возвращает - результат на блок работы вывод должности пользователя получает.

Поток: выполнение - это блок завершил свою работу и поток закончится.

Отправить " ок " - это наплыв бывает только в случае установки представить событие.

Отвергать - это наплыв бывает только в случае установки отклонить мероприятие.

Заполнить ОК - этот поток происходит только в случае заполнения позиции событие.

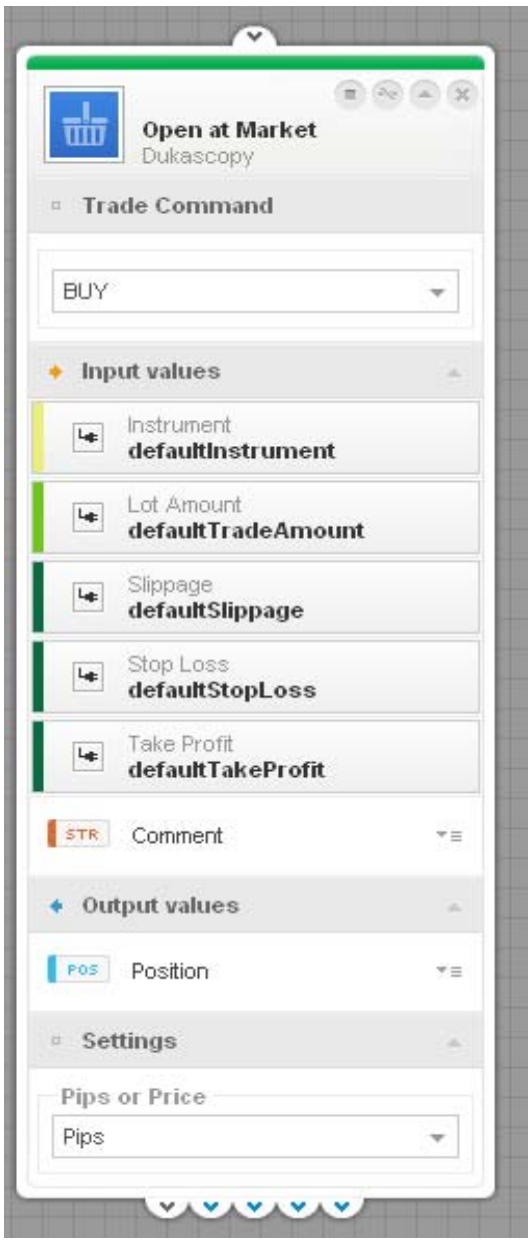
Заполнить отвергнуть - этого потока происходит только в случае заполнения позиции отклонить мероприятие.

Основные Определения:

" **Open at Market** " используется для торговли с рыночных ордеров(позиций). Самый важный параметр " **Open at Market** " - это " **Trade Command** " - где пользователь может выбрать торговую действие: "купить" или "продать". Параметр "Индикатор цен" - позволяет установить стоп лосс и тейк Профит либо в Целочисленные значения пипсов от текущей цены или в абсолютной цене Double значения. Комментарий – строковый тип не является обязательным, просто для информации пользователя.

" **Open at Market** " позволит сохранить позицию в переменную, но этот исходящее поле вывода необязательно.

Из " **Open at Market** " можно обрабатывать будущие позиции обработки событий. Событие будет обработано JForexVisual только один раз. Потоки события выделяются синим цветом.



Pending Open (В ожидании открытия).

"**Pending Open**" - выполняет ожидание открытия подчиняться системе.

Описание параметров:

Параметр **Price** (Цена входа) - определяет запись Цена отложенного ордера, Тип Double. Заполнить это поле обязательно.

Параметр **Instrument** - выберите необходимую валютную пару. Заполнить это поле обязательно.

Параметр **Lot Amount** - определяет размер лота в валюте счета, Тип Double. 1 - это 1 млн., минимальное значение = 0.001. Заполнить это поле обязательно.

Параметр **Slippage** - определяет проскальзывание в целых числах. Заполнить это поле обязательно. Значение проскальзывания означает следующее:

* если отрицательная, то значение проскальзывания по умолчанию 5 пунктов.

* если двойной.isNaN (проскальзывание) == True тогда нет проскальзывания.

* в противном случае, проскальзывание задается в пунктах, вы должны задать 1, не 0.0001

Параметр **Stop Loss** - определяет количество пунктов стоп-лосса от цены открытия.

Параметр **Take Profit** - определяет количество пунктов тейк профит от цены открытия.

Параметр **Comment** (комментарии) Строковый параметр определяет текст, комментарий, который будет сохранен в ордере.

Параметр **Stop Loss Double** параметр - определяет цену стоп-лосса. В случае цена должна быть кратна 0.1 пипсов или ордер будет отклонен.

Параметр **Take Profit Double** параметр - определяет цену тейк-профита. В случае цена должна быть кратна 0.1 пипсов или ордер будет отклонен.

Параметр **GoodTillTime** (Дата и время) - определяет дату, сколько времени ордер должен быть открыт, если не казнен. Только если > 0, то orderCommand должны быть PLACE_BID или PLACE_OFFER.

Position (Позиция)- результат работы блока – ордер который пользователь получает.

Поток: выполнение - это блок завершил свою работу и поток закончится.

Отправить " ок " - это поток бывает только в случае установки представить событие.

Отвергать - это поток бывает только в случае установки отклонить мероприятие.

Заполнить ОК - этот поток происходит только в случае заполнения позиции событие.

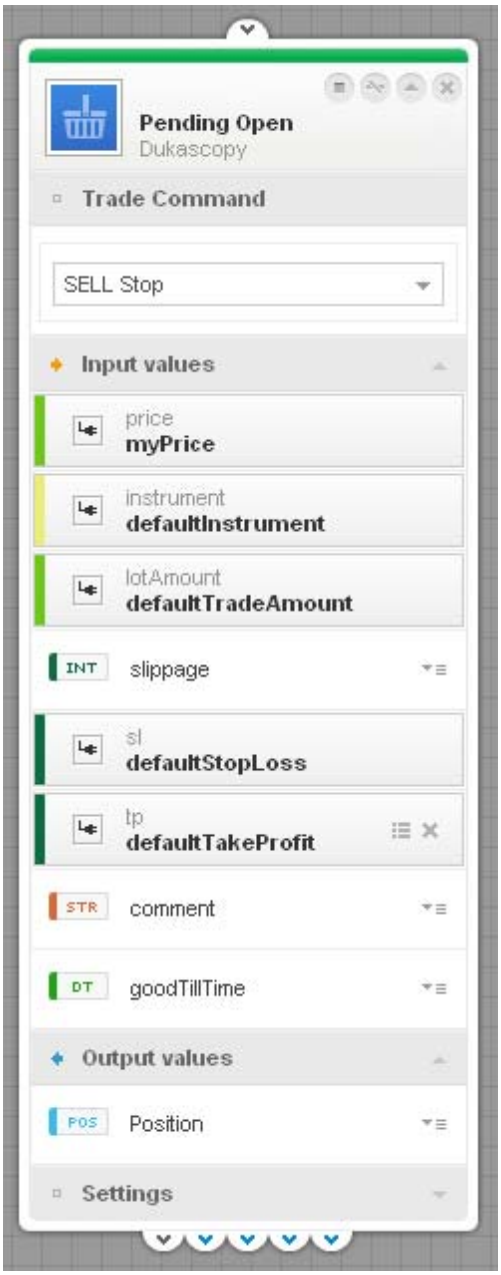
Заполнить отвергнуть - этого потока происходит только в случае заполнения позиции отклонить мероприятие.

Основные Определения:

" **Pending Open**" используется, чтобы установить новые отложенные торговые ордера. Наиболее важный параметр из " **Pending Open**" - это " **Trade Command** " - где пользователь может выбрать торговые действий: "Sell Stop", "Sell Stop по цене Ask" "Buy Stop", "Sell Limit", "Buy Limit", "Place Bid", "Place Offer". Параметр "Pips of Price" - позволяет установить стоп лосс и тейк Профит либо в Целочисленные значения пунктов от текущей цены или в абсолютной цене Double значения.

"**Pending Open**" позволяет сохранить позицию в переменную, но этот выход поля не надо.

Из " **Pending Trade** " можно обрабатывать будущую позицию обработки событий. Мероприятие будет обработан Visual jforex от только один раз. Потоки события выделяются синим цветом.



Set Stop Loss (установить стоп-лосс).

"**Set Stop Loss**" – устанавливает стоп-лосс на ордер.

Описание параметров:

Position - позиция/ордер переменной, чей стоп-лосс должен быть изменен. Обязательное поле.

Stop Loss - значения типа Double, цена на который должен быть уровень стоп-лосс. Устанавливается от цены открытия. Обязательное поле.

Stop Loss - целочисленное значение - количество пунктов, на которое должен быть уровень стоп-лосс устанавливается от цены открытия. Обязательное поле.

Поток: выполнение - это блок завершил свою работу и поток закончится.

Изменение позиции ок - этот поток происходит только в случае расположения/порядок успешного изменения

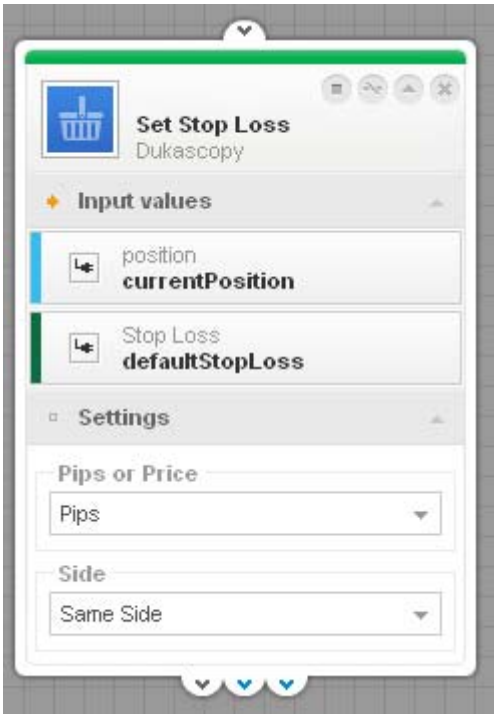
Изменение позиции отклонение - это наплыв бывает только в случае установки/отказа в выполнении распоряжения на изменение позиции

Основные Определения:

"Set Stop Loss" используется для установки стоп-лосса уровень установленными во входных параметрах позиция. В зависимости от набора параметров Цена/пунктов - пользователь может определить стоп-лосс: либо в цене (Double значение типа) или в пунктах (int значение типа)". "Set Stop Loss" - имеет 2 входа, который он/она должен заполнить обязательные это - позиция переменной и стоп-лосса. Позиция переменная не должна быть "нулем". По умолчанию значение стоп-лосса устанавливается с "defaultStopLoss" переменной. Второй параметр определяет Сторона: одной стороне или противоположной стороне. Же сторону - значит по умолчанию задается установка стоп-лосса, напротив, означает, что она будет установлена на противоположной bid или ask, в зависимости от типа должности.

Иногда это блоки используются с позиции зрителя и зрителя петли, которая позволяет пройти все текущие позиции. "Set Stop Loss" - имеет 2 события потоков - по успешной позиции изменения и

отклонить мероприятие. Мероприятие будет обработан Visual jforex от только один раз. Потоки события выделяются синим цветом.



Set Take Profit (Установить тейк-Профит).

"**Set Take Profit**" - устанавливает/изменят тейк-профит на позицию/ордер.

Описание параметров:

Параметр **Position** (позиция/ордер), для которой должен быть изменен тейк-профит. Обязательное поле.

Параметр **Take Profit** - значение типа Double, цена на который должен быть установлен тейк профит от цены открытия. Обязательное поле.

Take Profit - Целочисленное значение - количество пунктов, на которые следует установить уровень тейк Профит от цены открытия. Обязательное поле.

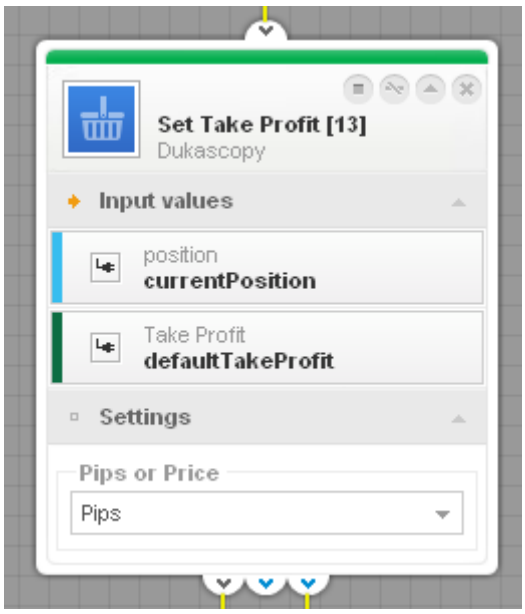
Поток: выполнение - это блок завершил свою работу и поток закончится.

Изменение позиции ок - этот поток происходит только в случае расположения/порядок успешного изменения

Изменение позиции отклонение - это наплыв бывает только в случае установки/отказа в выполнении распоряжения на изменение позиции

Основные Определения:

"**Set Take Profit**" используется, чтобы установить уровень тейк Профит для определенных во входных параметрах позиция. В зависимости от набора параметров Цена/пунктов - пользователь может определить тейк-Профит: либо в цене (Double значение типа) или в пунктах (int значение типа)". "**Set Take Profit**" - имеет 2 входа, который он/она должен заполнить обязательные это - позиция переменной и тейк-профита. Позиция переменная не должна быть "нулем". По умолчанию значение тейк-профита устанавливается с "defaultTakeProfit" переменной. Иногда это блоки используются с позиции зрителя и зрителя петли, которая позволяет пройти все текущие позиции. "**Set Take Profit**" - имеет 2 события потоков - по успешной позиции изменения и отклонить мероприятие. Мероприятие будет обработан Visual jforex от только один раз. Потоки события выделяются синим цветом.



Trailing Stop (Трейлинг-стоп).

"**Trailing Stop**" - выполняет положение скользящего шага, такой же, как на платформе Дукаскопи.

Описание параметров:

Параметр **Position** - позиция/ордер переменной, что стоп-лосс должен быть прицепной. Обязательное поле

Параметр **Stop Loss** - целочисленное значение - количество пунктов, на которое должен быть установлен стоп лосс итерированных от цены открытия. Обязательное поле.

Параметр **Trailing Step** (скользящий шаг) - целочисленное значение (в пунктах) - шаг значения трейлинга, должно быть больше, чем 10. Обязательное поле.

Stop Loss - значения типа Double, цена на который должен быть уровень стоп-лосс устанавливается от цены открытия. Обязательное поле.

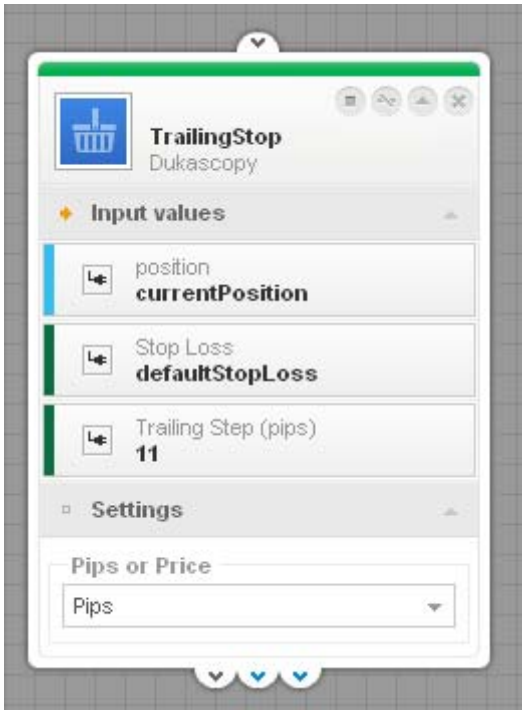
Поток: выполнение - это блок завершил свою работу и поток закончится.

Изменение позиции ок - этот поток происходит только в случае расположения/порядок успешного изменения.

Изменение позиции отклонение - это наплыв бывает только в случае установки/отказа в выполнении распоряжения на изменение позиции.

Основные Определения:

"**Trailing Stop**" используется для установки стоп-лосса и подтягивать стоп лосс установленными во входных параметрах позиция. Трейлинг функциональность используется трейлинг Дукаскопи, как в платформе. Шаг трейлинга должно быть определено больше, чем 10, это значение является целым числом и это поле обязательное. В зависимости от набора параметров Цена/пунктов - пользователь может определить стоп-лосс: либо в цене (Double значение типа) или в пунктах (int значение типа)". "Установить стоп-лосс" - имеет 2 входа, который он/она должен заполнить обязательные это - позиция переменной и стоп-лосса. Позиция переменная не должна быть "нулем". По умолчанию значение стоп-лосса устанавливается с "defaultStopLoss" переменной. Иногда это блоки используются с позиции зрителя и зрителя петли, которая позволяет пройти все текущие позиции. "Установить стоп-лосс" - имеет 2 события потоков - по успешной позиции изменения и отклонить мероприятие. Мероприятие будет обработан Visual jforex от только один раз. Потоки события выделяются синим цветом. Например, если шаг Трейлинга задается 10, что означает, что уровень стопа будет меняться каждые 10 пунктов.



Компоненты Utils

Get Time Unit (Получить единицу времени)

"**Get Time Unit**" - позволяют получить от Дата и время (продолжительность) значение, числовое число, день месяца, день недели, час, минуту или секунду.

Описание параметров:

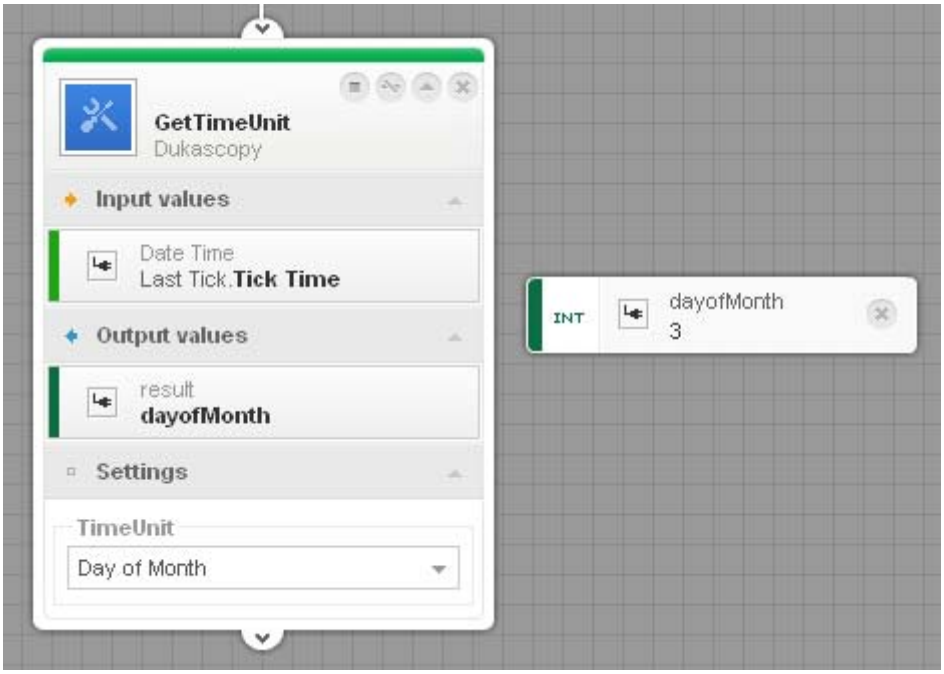
Параметр **Date Time** - дата и время (продолжительность) значение, которое должно быть отформатировано. Обязательное поле.

Исходный параметр **Result** Результат - результат переменной, которая может быть целое число, вещественное число, Дата и время (продолжительность) или Строковых типов. Обязательное поле.

Потоки: выполнение - это блок завершил свою работу и поток закончится.

Основные Определения:

"Получить единицу времени" - позволяют получить от Дата и время (продолжительность) значение, числовое число, день месяца, день недели, час, минуту или секунду. Выходной переменной, которая может быть целое число, вещественное число, Дата и время или Строковых типов. Дополнительная информация: по Дню недели, в воскресенье-1, и в субботу в 7.



Sound Alert (Звуковой сигнал)

Результат блока - звуковой сигнал.

Stop Strategy (Остановить стратегию)

Результат блока - остановка работы стратегии.