iExpertAdvisor™

Signal Aggregator Plug-in

Requires VTS-Connect minimum version **4.0.0.54**

The ***Signal Aggregator Plug-in*** allows a large group of trading signals to be evaluated using Fuzzy Logic by assigning a custom weight to each trade signal.

What is a Plug-in?
VTS stands for Visual Traders Studio.

The VTS Expert Advisor Builder is a Windows graphical application that enables non-programmers to build complex Expert Advisors by dragging, dropping and connecting logical elements.

The VTS application contains basic functionality to build almost any Expert Advisor.

A VTS Plug-in allows traders to easily implement advanced trading techniques using an add-on user interface.

Contents

Enable the *Signal Aggregator* Plug-in 5

What is the Signal Aggregator? 6

Signal Aggregator in the Toolbox 9

Creating a *Signal Aggregator* Element 10

Adding a Signal Type Element to a *Signal Aggregator* 12

Adding a Signal Type MQL to a *Signal Aggregator* 16

Adding a Signal Type Logic Condition to a *Signal Aggregator* 18

Configuring the *Weight* and *Threshold* 20

Using a *Signal Aggregator* on a Drawing 23

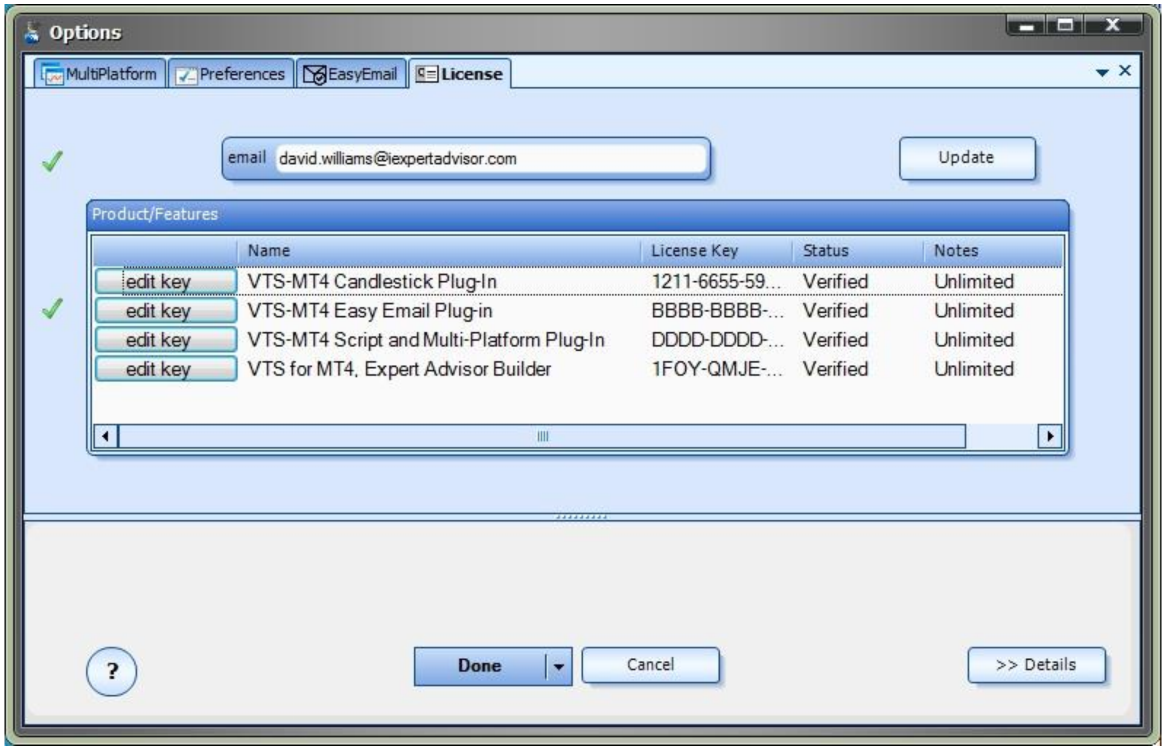
Enable the *Signal Aggregator* Plug-in

You must enter your License key to enable the *Signal Aggregator Plug-in*. Your license key for all of your VTS products can be found in the Members Area.

License keys are entered in Visual Traders Studio (VTS) from the License entry tab.

- The email address is the email address used to purchase VTS.
- The License Key is the key listed in the Members Area.
- The Update button is used to verify the email address and license key.
- The edit key button is used edit the key value.

NOTE: After entering the license key, VTS must be restarted to build and enable the Signal Aggregator features.



email david.williams@iexpertadvisor.com

	Name	License Key	Status	Notes
edit key	VTS-MT4 Candlestick Plug-In	1211-6655-59...	Verified	Unlimited
edit key	VTS-MT4 Easy Email Plug-in	BBBB-BBBB-...	Verified	Unlimited
edit key	VTS-MT4 Script and Multi-Platform Plug-In	DDDD-DDDD-...	Verified	Unlimited
edit key	VTS for MT4, Expert Advisor Builder	1FOY-QMJE-...	Verified	Unlimited

What is the Signal Aggregator?

Most trading systems evaluate a *trading signal* and execute actions if the signal is *true* or *false*.

Some trading systems will use multiple signals. Generally, all of the signals must evaluate to *true* for an action to be executed.

A *trade signal* is any logic derived from the market that can be evaluate to *true* or *false*. A *trade signal* is usually based on technical analysis: indicators values, prices, candle-stick patterns, etc.

An example of a simple *trade signal* is:

RSI > 75.0

This *trade signals* reads "RSI greater than 75.0". If the current RSI value is greater than 75.0, then this *trade signal* will evaluate to *true*.

This kind of signal can be very powerful and is the basis for virtually all automated trading systems.

The *Signal Aggregator* brings technical analysis to another level. It combines an unlimited number of trade signals and applies an individual weight to each signal.

The *Signal Aggregator* works likes this:

Trade Signals are added to the *Signal Aggregator.*

Each Signal returns a value of *true* or *false.*

Each Signal is assigned a weight.

If a Signal evaluates to *true*, its weight is added to the *Signal Aggregator's* total weight.

If the *Signal Aggregator's* total weight is greater than a threshold value, a trading action is executed.

This aggregation allows some signals to be assigned a greater or lesser weight than other signals. The result is that a single *trade signal* does not control the entire trade action.

This is an example of six *trade signals*, each with its own assigned weight:

Trade Signal	**Weight**	**Notes**
ADX > 35.0	20	ADX is greater than 35.0
Close[1] > Close[2]	20	The Close price of the previous candle is greater than the Close price before the previous candle
RSI < 75.0	20	The RSI is less than 75
Hour > 5 and Hour < 11	10	The time is between 5 and 11 o'clock
High[1] > iHighest(24,...)	25	The current High price is greater than the highest high of the last 24 candles
Daily ATR > Weekly ATR	5	The daily ATR is greater than the weekly ATR
Total*	100	Total weight of all signals

*The sum of all weights is 100.

For this *Signal Aggregator* a *threshold* value is chosen, for example: threshold = 75.

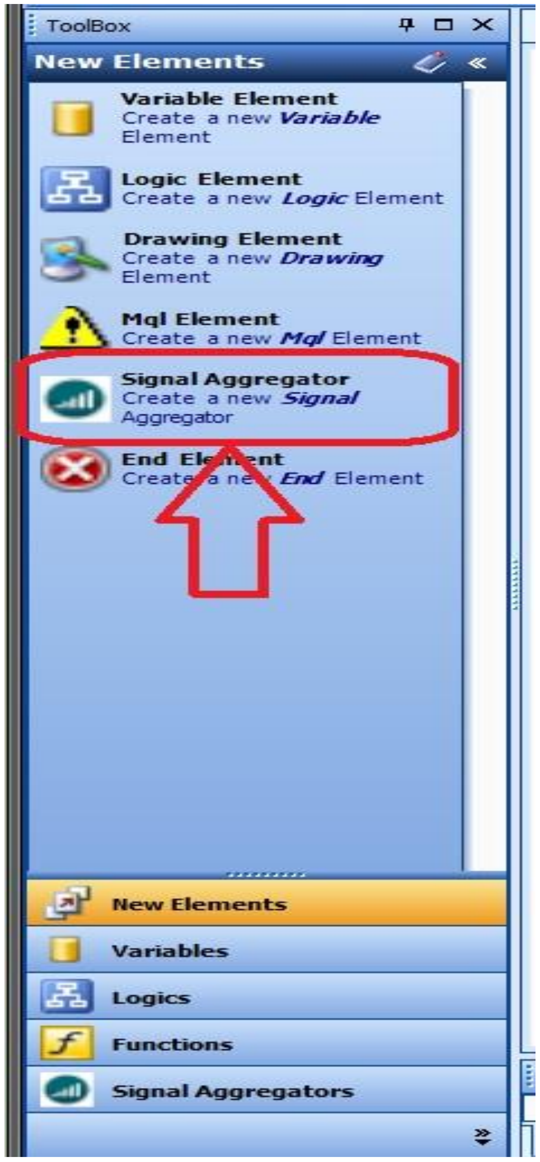
If a Trade Signal evaluates to *true*, its weight is added to the *Signal Aggregator's* total weight.

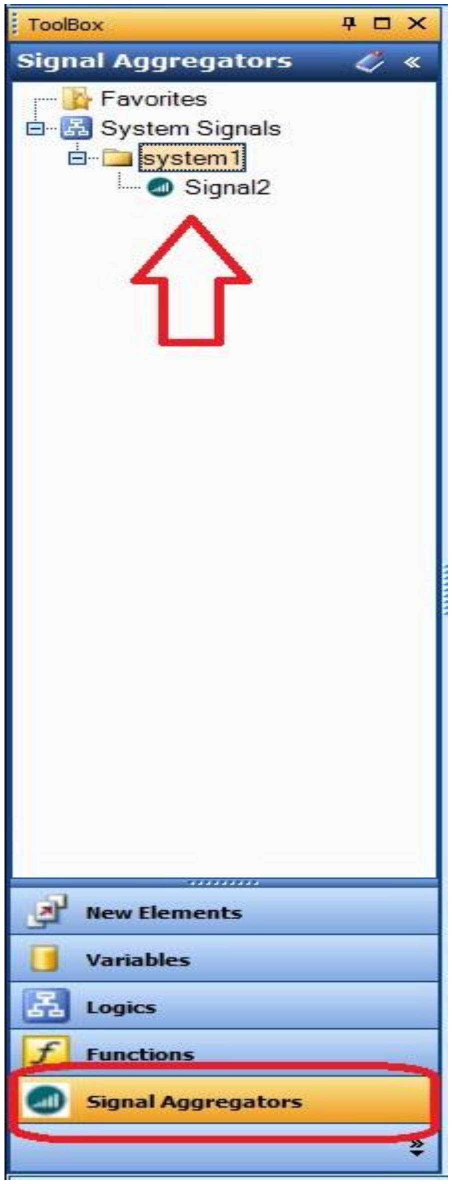
If the *Signal Aggregator's* total weight is greater than 75, a trade action is executed.

There are several combinations of *true Trade Signals* that can lead to a threshold value greater than 75, but no one signal can drive the trade action.

Signal Aggregator in the Toolbox

The *Signal Aggregator* appears in the Toolbox in the New Elements pane and in its own pane for previously defined *Signal Aggregators.*

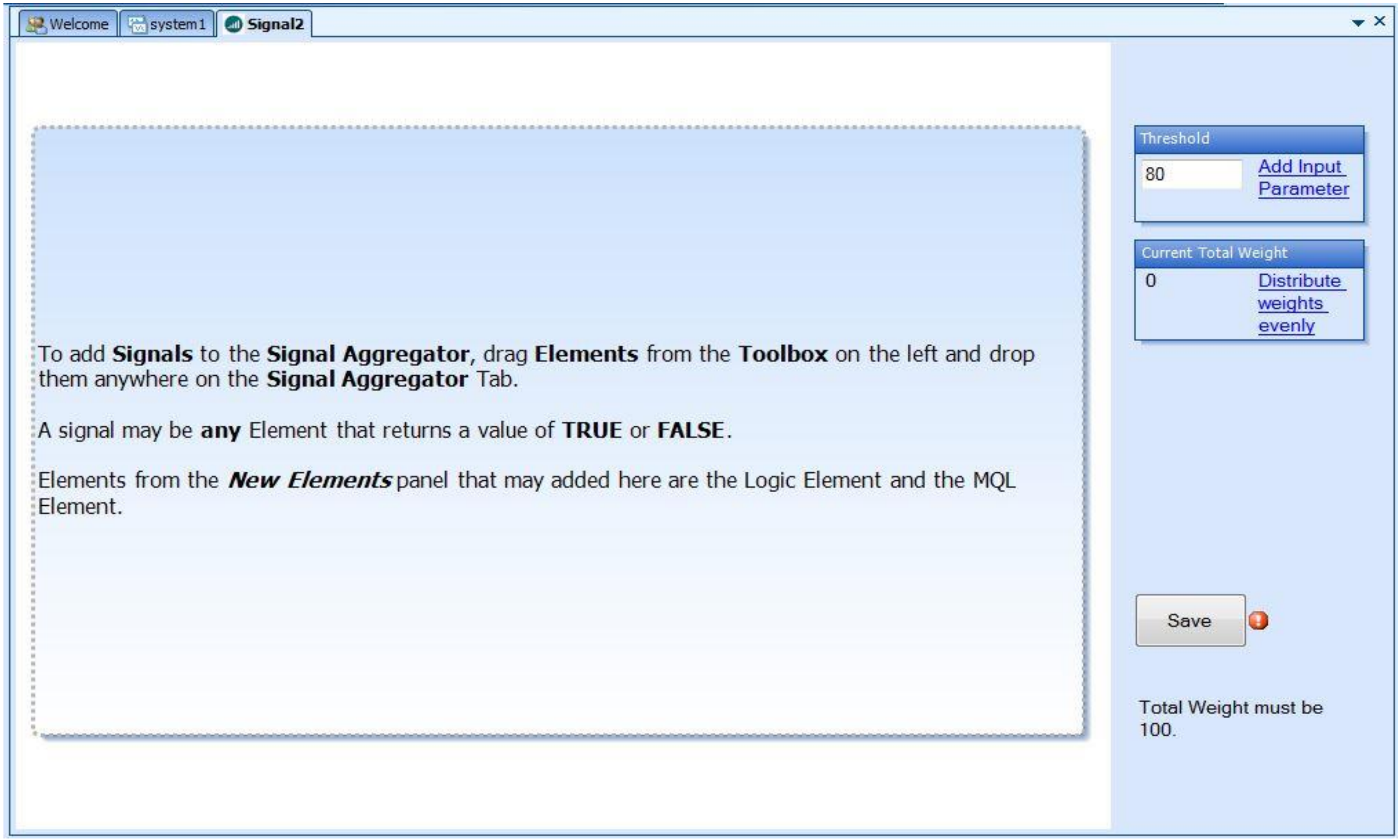




Creating a *Signal Aggregator* Element

To begin creating a *Signal Aggregator*, drag a *Signal Aggregator* from the Toolbox onto the Drawing Pad of an open VTS system.

The *Signal Aggregator* instruction screen is shown:



To add **Signals** to the **Signal Aggregator**, drag **Elements** from the **Toolbox** on the left and drop them anywhere on the **Signal Aggregator** Tab.

A signal may be **any** Element that returns a value of **TRUE** or **FALSE**.

Elements from the ***New Elements*** panel that may added here are the Logic Element and the MQL Element.

Total Weight must be 100.

The following Elements can be dragged onto the *Signal Aggregator*:

From the *New Element* pane: Logic Element

From the *New Element* pane: MQL Element

From the *Logics* pane: any previously defined Logic Element

From the *Functions* pane: Any function that returns a value of *true* or *false*.

Adding a Signal Type Element to a *Signal Aggregator*

A signal type Element is a previously defined Logic or Function Element.

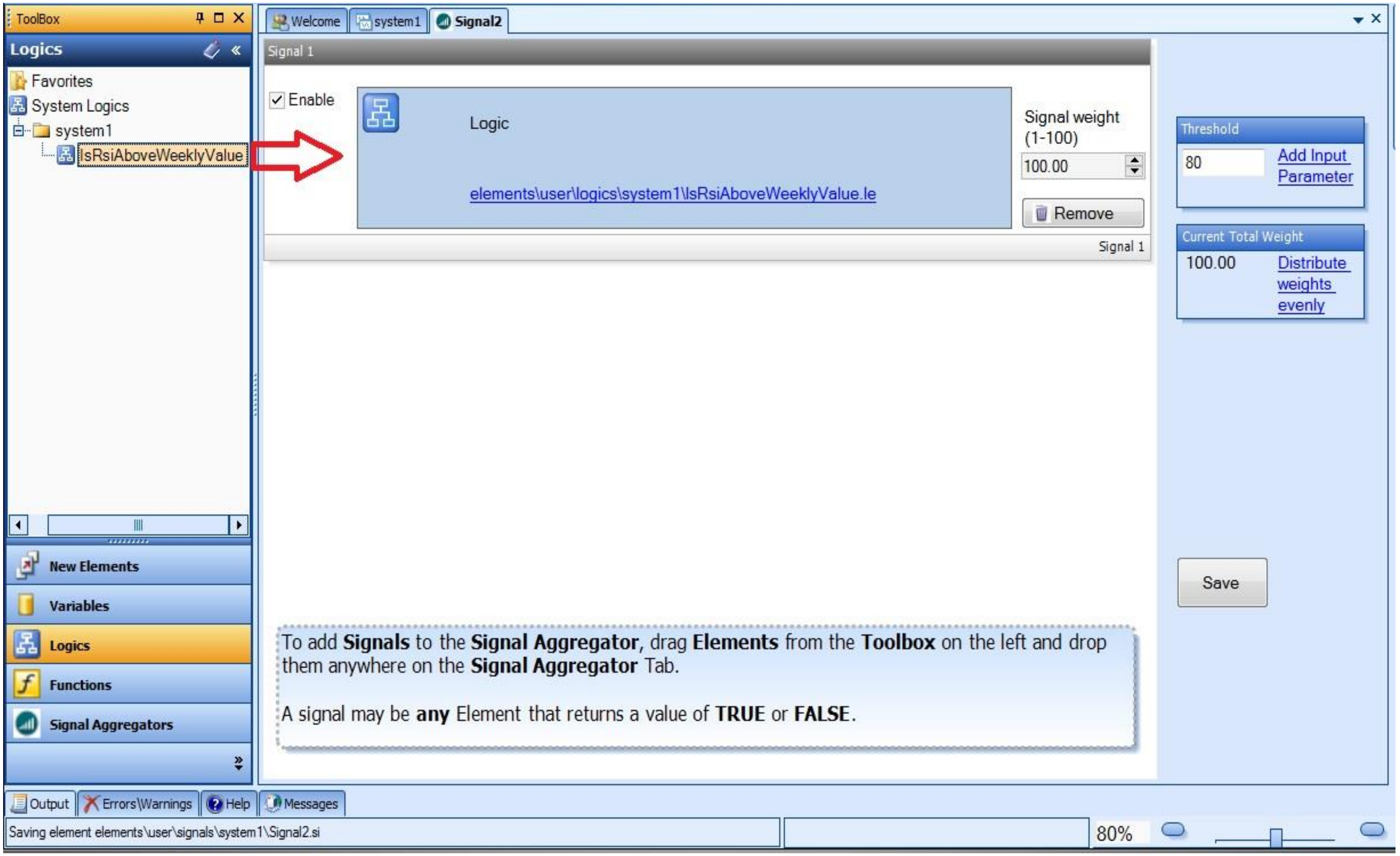
- Any Logic Element can be added.
- Any Function Element that returns true or false can be added.

Note: When a Function Drawing is added to a *Signal Aggregator*, the Function Drawing's Element must be present and connected on the same drawing as the *Signal Aggregator*.

For example, follow these steps to add the CandleStick Pattern Drawing Function *Black_Body* to the *Signal Aggregator*:

- Drag, drop and connect the *Black_Body* Element from the Toolbox CandleSticks menu onto the Drawing Pad in front of the *Signal Aggregator* Element.
- Configure and save the *Black_Body* CandleStick Pattern function, for example save it as *"Black_Body1".*
- Drag the *"Black_Body1"* Element from the Toolbox System Functions menu onto the *Signal Aggregator*.

This image shows a user-define Logic named *IsRsiAboveWeeklyValue* added to the *Signal Aggregator*:



elements\user\logics\system1\IsRsiAboveWeeklyValue.le

Signal weight (1-100): 100.00

Threshold: 80

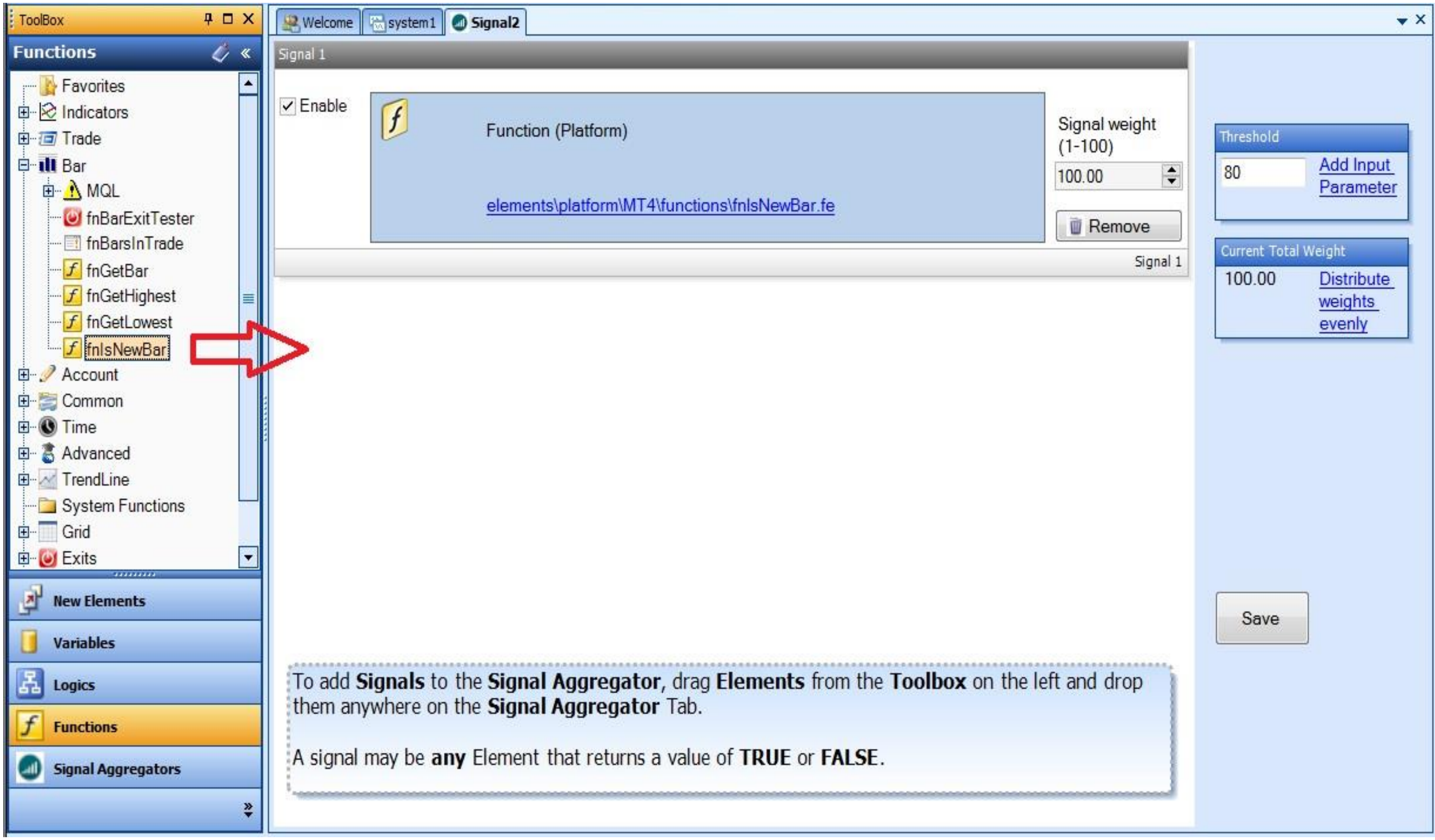
Current Total Weight: 100.00

To add **Signals** to the **Signal Aggregator**, drag **Elements** from the **Toolbox** on the left and drop them anywhere on the **Signal Aggregator** Tab.

A signal may be **any** Element that returns a value of **TRUE** or **FALSE**.

Saving element elements\user\signals\system1\Signal2.si

This image shows the VTS built-in Function *fnIsNewBar* added to the Signal Aggregator:



Enable [✓]

Function (Platform)

elements\platform\MT4\functions\fnIsNewBar.fe

Signal weight (1-100)

100.00

Threshold

80

Add Input Parameter

Current Total Weight

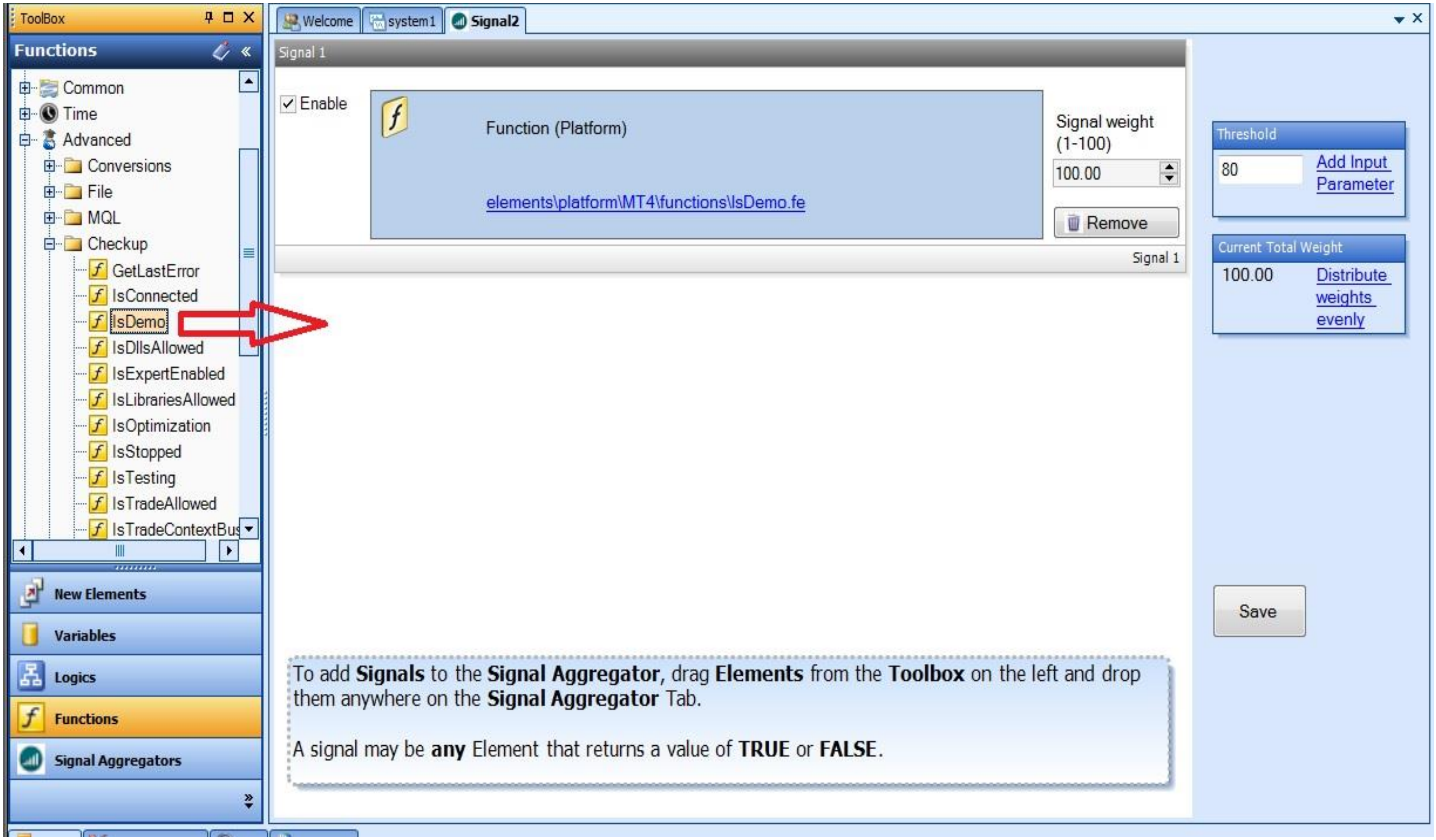
100.00

Distribute weights evenly

To add **Signals** to the **Signal Aggregator**, drag **Elements** from the **Toolbox** on the left and drop them anywhere on the **Signal Aggregator** Tab.

A signal may be **any** Element that returns a value of **TRUE** or **FALSE**.

This image shows the MetaTrader Platform Function *IsDemo* added to the *Signal Aggregator*:



Enable ✓

Function (Platform)

elements\platform\MT4\functions\IsDemo.fe

Signal weight (1-100): 100.00

Threshold: 80

Current Total Weight: 100.00

To add **Signals** to the **Signal Aggregator**, drag **Elements** from the **Toolbox** on the left and drop them anywhere on the **Signal Aggregator** Tab.

A signal may be **any** Element that returns a value of **TRUE** or **FALSE**.

Adding a Signal Type MQL to a *Signal Aggregator*

A signal type MQL is a used to add native MQL code to a Signal Aggregator.

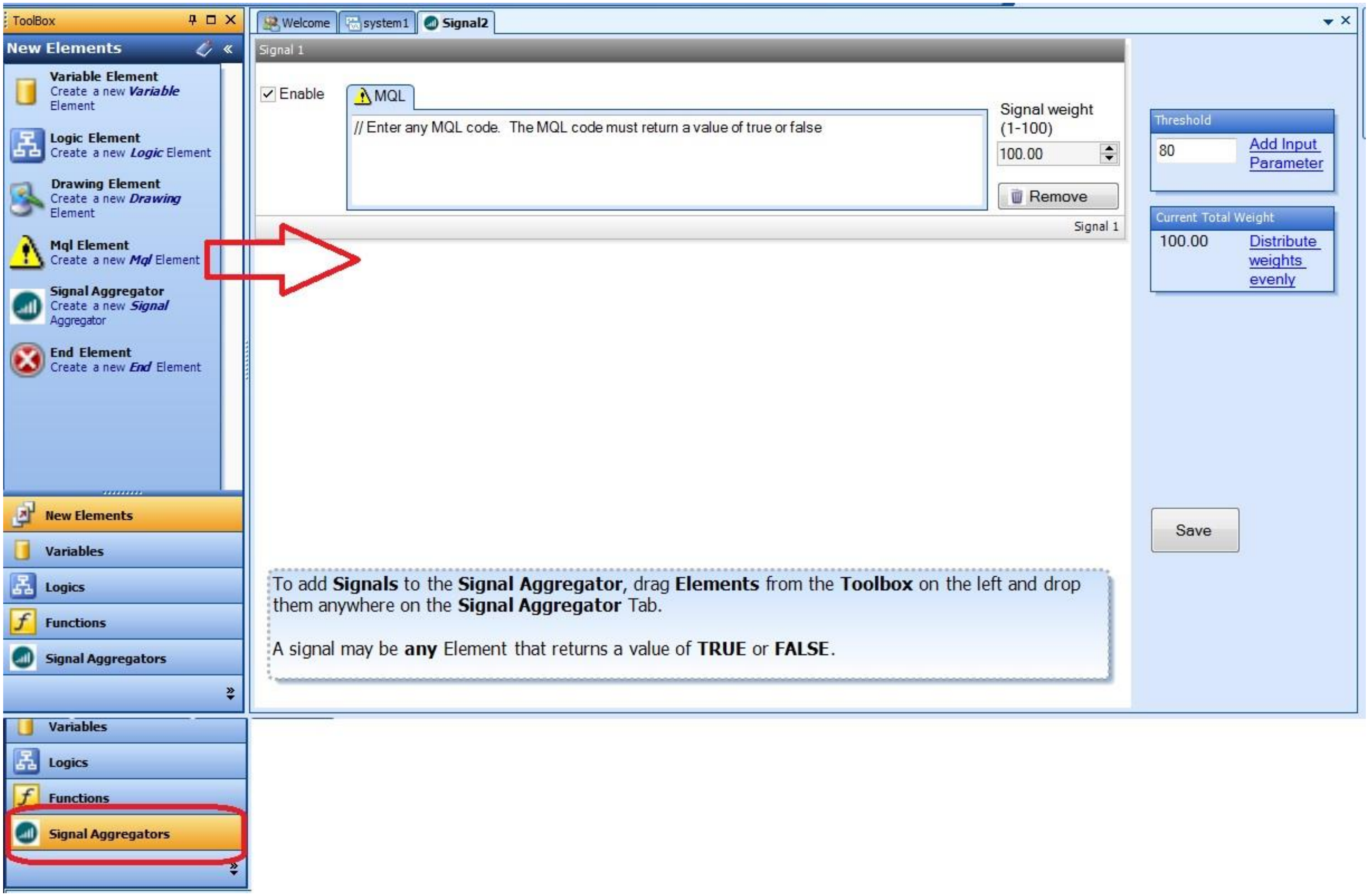
The MQL code must be an expression that can be evaluated to *true* or *false*. For example:

Ask > Bid

Close < Open

Warning: Invalid MQL code added to the MQL Element will prevent the Expert Advisor from building. Use this feature carefully.

This image shows a MQL Element added to a *Signal Aggregator*:



// Enter any MQL code. The MQL code must return a value of true or false

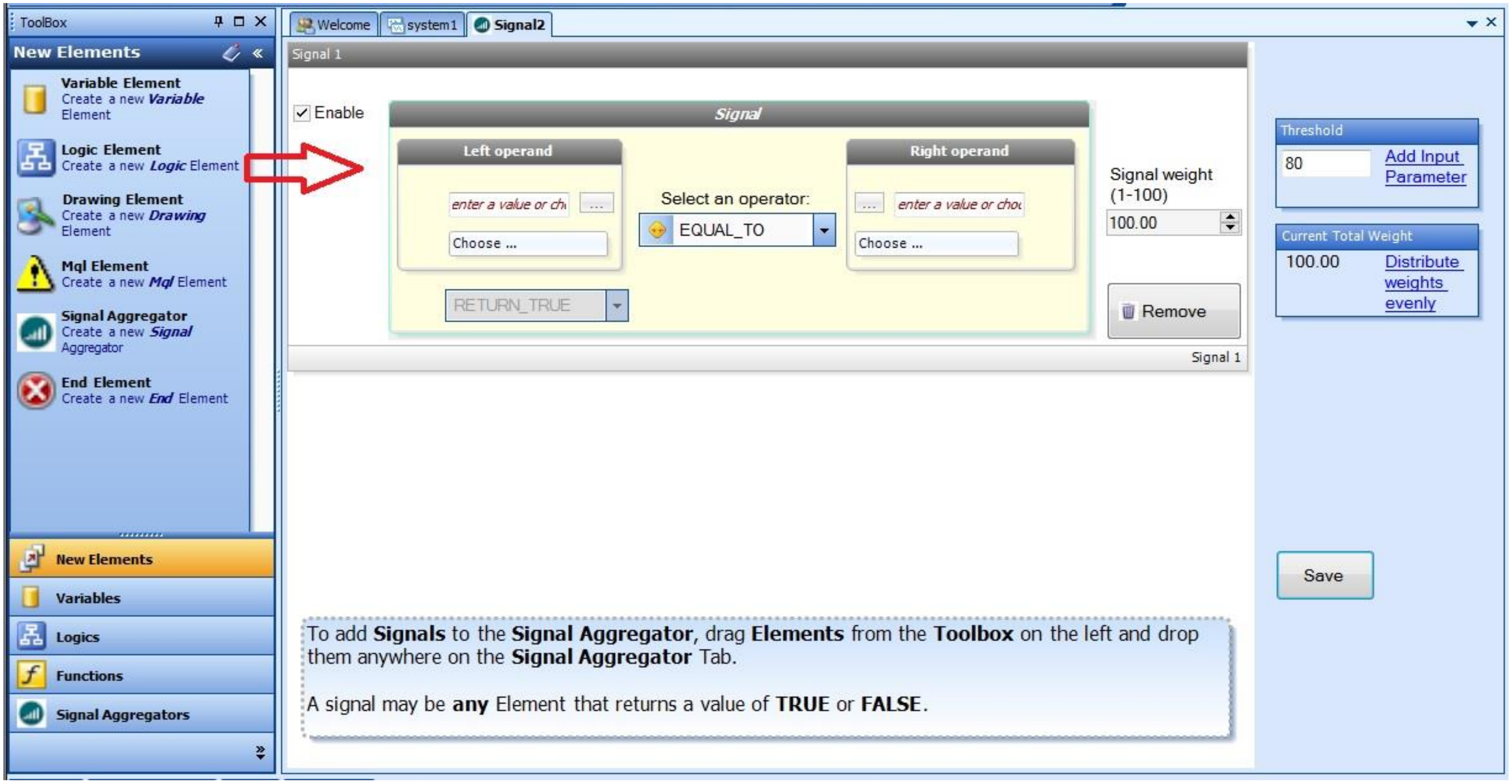
To add **Signals** to the **Signal Aggregator**, drag **Elements** from the **Toolbox** on the left and drop them anywhere on the **Signal Aggregator** Tab.

A signal may be **any** Element that returns a value of **TRUE** or **FALSE**.

Adding a Signal Type Logic Condition to a *Signal Aggregator*

A logical condition can be defined within a *Signal Aggregator* by dragging a New Logic Element into the *Signal Aggregator*.

This image shows a Logic Condition added to a *Signal Aggregator*:



To add **Signals** to the **Signal Aggregator**, drag **Elements** from the **Toolbox** on the left and drop them anywhere on the **Signal Aggregator** Tab.

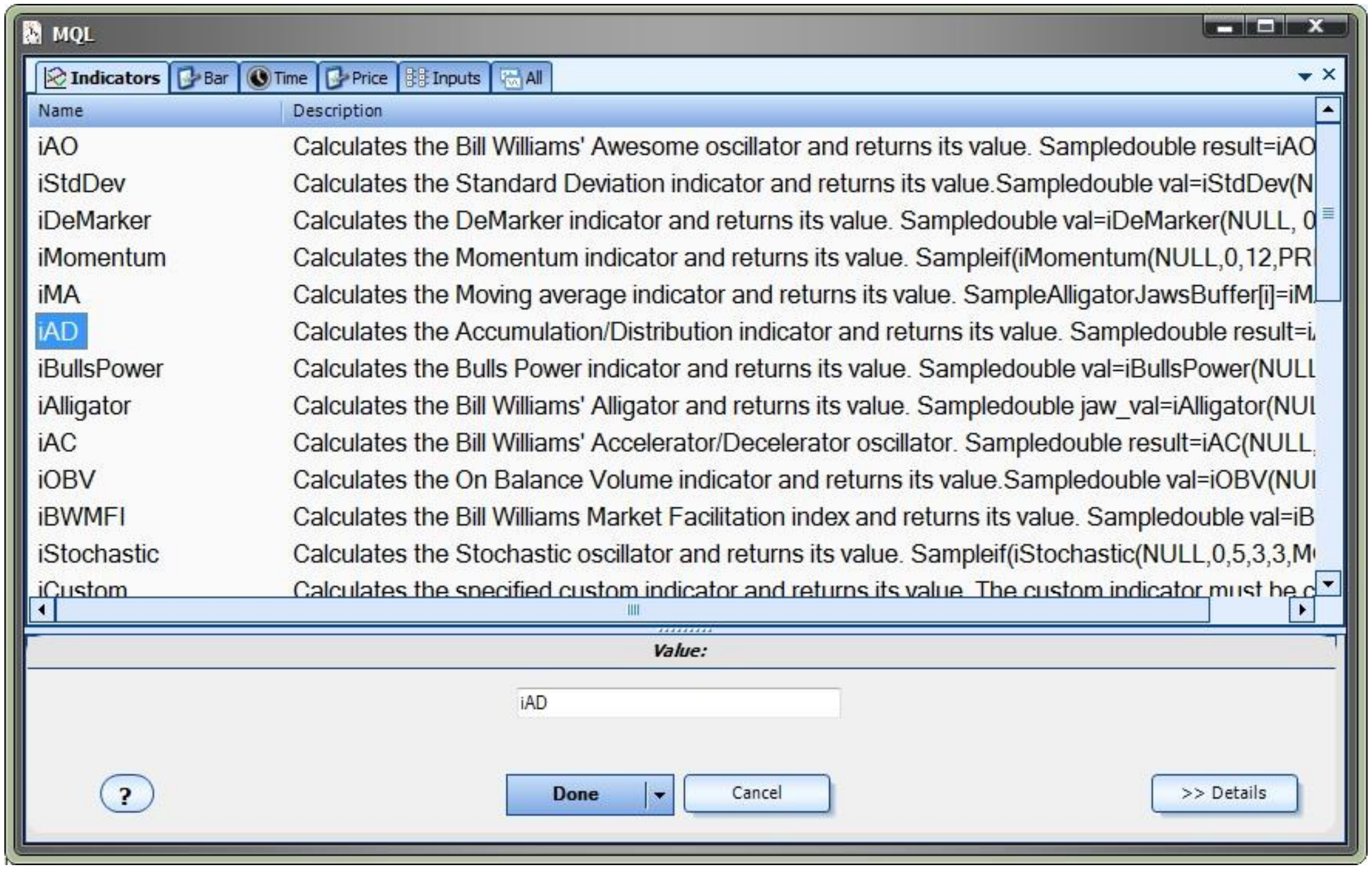
A signal may be **any** Element that returns a value of **TRUE** or **FALSE**.

The logic condition of the *Signal Aggregator* is configured similar to how a condition is configured in a Logic Element, except:

A logic condition of a *Signal Aggregator* always returns *true*.

A logic condition of the *Signal Aggregator* is limited to a single condition.

MetaTrader Indicator functions can be directly added to the logical condition by clicking the *Choose* button and selecting the *Indicator* tab:



Name	Description
iAO	Calculates the Bill Williams' Awesome oscillator and returns its value. Sampledouble result=iAO
iStdDev	Calculates the Standard Deviation indicator and returns its value.Sampledouble val=iStdDev(N
iDeMarker	Calculates the DeMarker indicator and returns its value. Sampledouble val=iDeMarker(NULL, 0
iMomentum	Calculates the Momentum indicator and returns its value. Sampleif(iMomentum(NULL,0,12,PR
iMA	Calculates the Moving average indicator and returns its value. SampleAlligatorJawsBuffer[i]=iM
iAD	Calculates the Accumulation/Distribution indicator and returns its value. Sampledouble result=i
iBullsPower	Calculates the Bulls Power indicator and returns its value. Sampledouble val=iBullsPower(NULL
iAlligator	Calculates the Bill Williams' Alligator and returns its value. Sampledouble jaw_val=iAlligator(NUL
iAC	Calculates the Bill Williams' Accelerator/Decelerator oscillator. Sampledouble result=iAC(NULL,
iOBV	Calculates the On Balance Volume indicator and returns its value.Sampledouble val=iOBV(NUI
iBWMFI	Calculates the Bill Williams Market Facilitation index and returns its value. Sampledouble val=iB
iStochastic	Calculates the Stochastic oscillator and returns its value. Sampleif(iStochastic(NULL,0,5,3,3,M
iCustom	Calculates the specified custom indicator and returns its value. The custom indicator must be c

Value:

iAD

Configuring the *Weight* and *Threshold*

After adding Signals to the *Signal Aggregator*, the *Signal weight* and *Threshold* values are adjusted.

The *Threshold* is the value that the sum of the active signals must exceed for the *Signal Aggregator* to generate a value of *true*.

For example, if the *Threshold* value is 80.0, when the sum of the *Signal weights* of the Signals that return true exceeds 80.0, the *Signal Aggregator* will return a value of *true*.

- The *Threshold* value can be defined as an input parameter to the EA by clicking the link *Add Input Parameter.*
- The *Signal weight* is defined for each Signal.
- The *Signal weight* is a value between 1-100.
- The sum of all active *Signal weights* should be equal to 100.

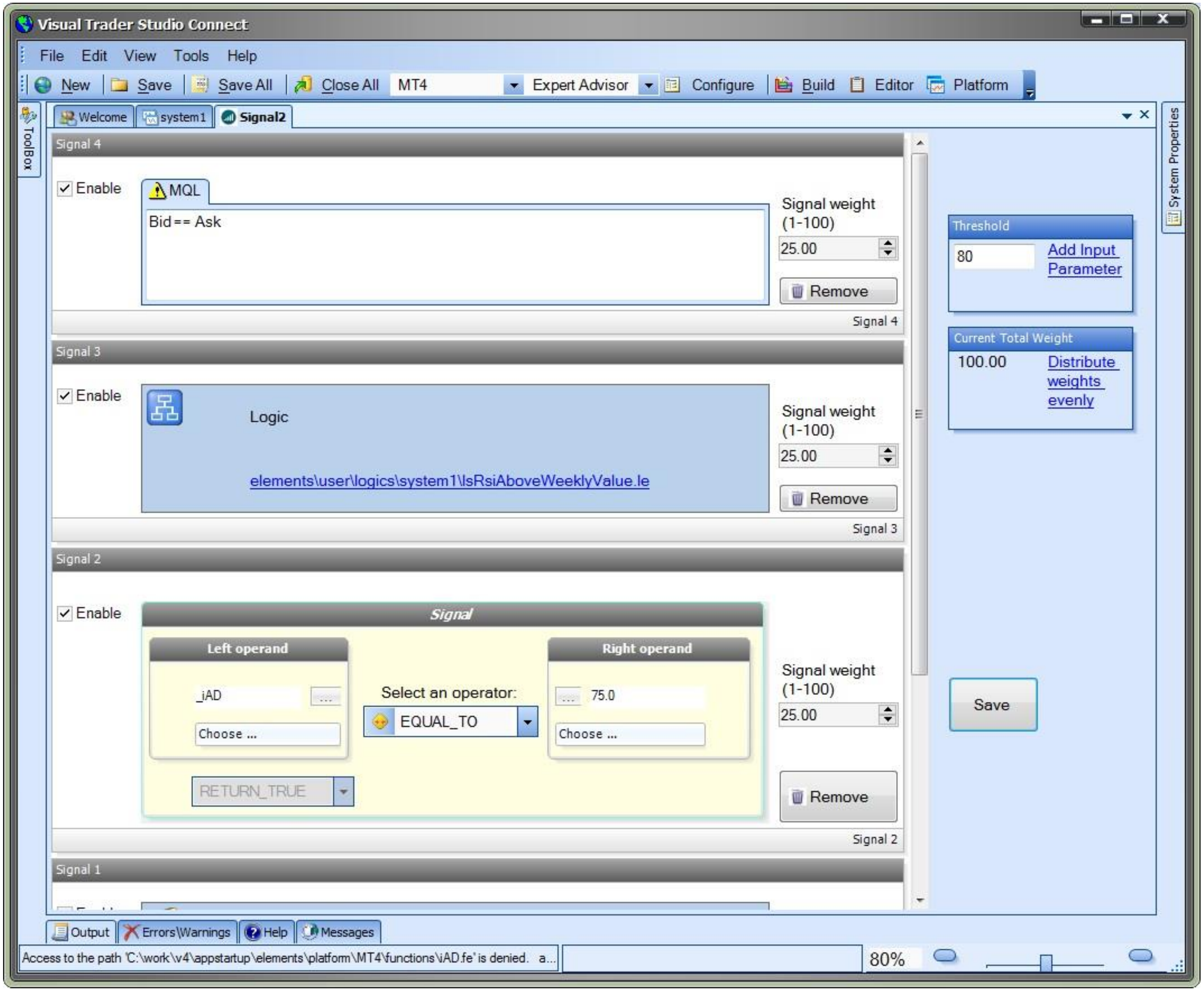
Click the link *Distribute weights evenly* to assign the same weight to all active Signals.

The *Current Total Weight* is displayed. This is a read-only value that calculates the current sum of all active Signal's weights.

The Save button is used to save the *Signal Aggregator*.

Each Signal has an *Enable* button.   When the *Enable* button is unchecked:

- The Signal's weight is not added to the *Current Total Weight.*
- The Signal is not included in the generated MQL code.


Visual Trader Studio Connect
File Edit View Tools Help
New Save Save All Close All MT4 Expert Advisor Configure Build Editor Platform
ToolBox
Welcome system1 Signal2
System Properties
Output Errors\Warnings Help Messages

Signal 4

Enable: ✓

MQL

```
Bid == Ask
```

Signal weight (1-100): 25.00

Remove

Signal 3

Enable: ✓

Logic

elements\user\logics\system1\IsRsiAboveWeeklyValue.le

Signal weight (1-100): 25.00

Remove

Signal 2

Enable: ✓

Signal

Left operand: _iAD

Choose ...

Select an operator: EQUAL_TO

Right operand: 75.0

Choose ...

RETURN_TRUE

Signal weight (1-100): 25.00

Remove

Signal 1

Threshold: 80 Add Input Parameter

Current Total Weight: 100.00 Distribute weights evenly

Save

Access to the path 'C:\work\v4\appstartup\elements\platform\MT4\functions\iAD.fe' is denied. a...

80%

Using a *Signal Aggregator* on a Drawing

After a Signal Aggregator has been saved it is available in the Toolbox to be dragged onto the Drawing Pad.

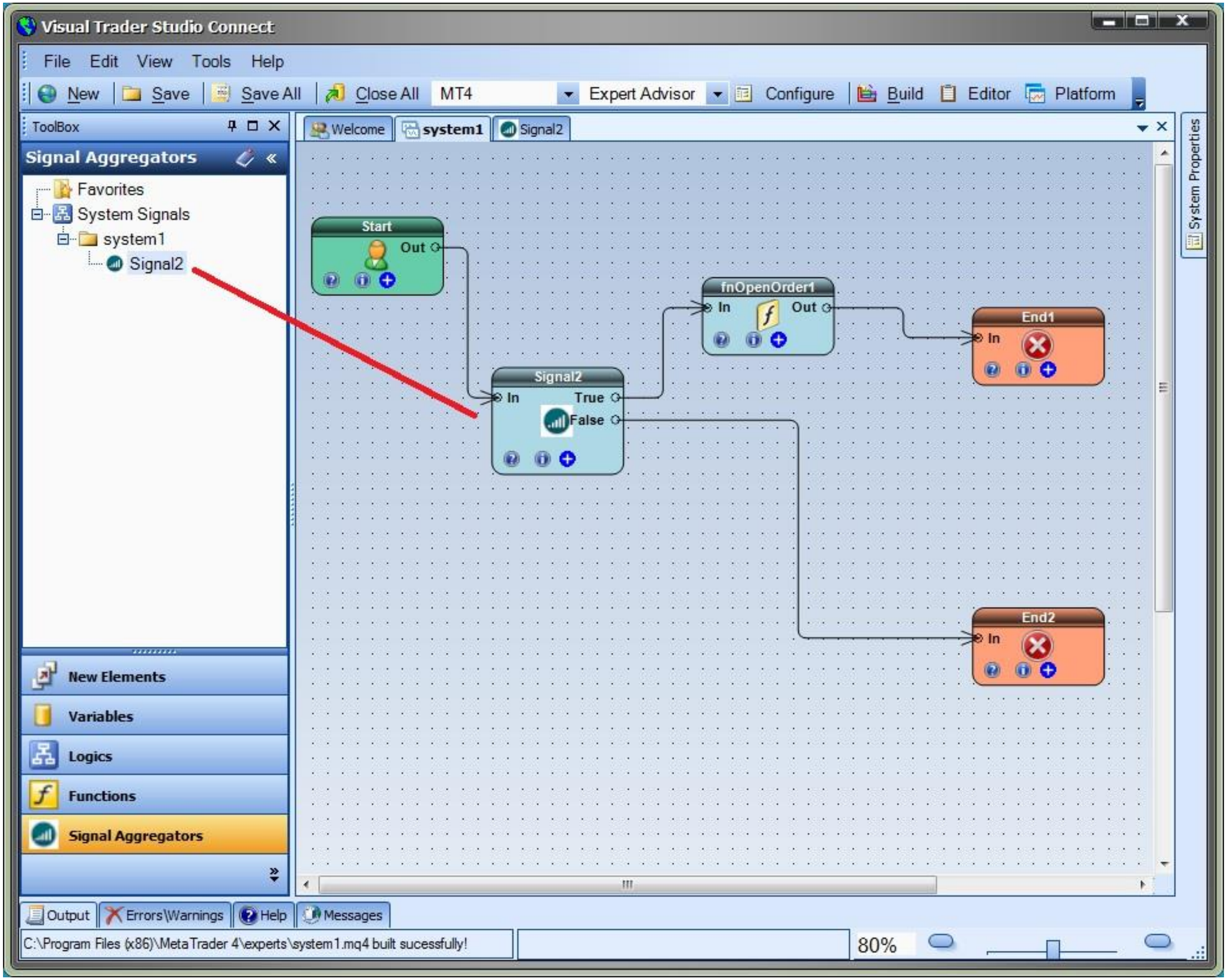
A Signal Aggregator is used on a Drawing similar to a Logic Element.

- A *Signal Aggregator* has a single input.
- A *Signal Aggregator* has two outputs: *true* and *false*.
- Execution will follow the *true* link when the sum of the active signals exceeds the *Threshold* value.
- Execution will follow the *false* link when the sum of the active signals is equal to or below the *Threshold* value.

A Signal Aggregator is opened for configuration:

  - Clicking the (+) on the *Signal Aggregator* Element
  - Right-clicking the *Signal Aggregator* in the Toolbox and selecting *Configure*.

In the Drawing below, when the sum of the active signals exceeds the *Threshold* value of the *Signal2 Signal Aggregator*, execution will follow the *true* output, and the fnOpenOrder1 function is called to open a trade.



C:\Program Files (x86)\MetaTrader 4\experts\system1.mq4 built sucessfully!