https://www.garot.com/trading/

01 – Introduction

This is the informal video blog.

This sixth video in this VLOG series is called: **Show me some statistics (and other cool code)**

I haven't made a video in a while because my life got side-tracked as life often does. Coronavirus this, and Coronavirus that. Also, I had planned to stop this series after episode 05, but I had some other ideas that I thought might be useful to you to help you create your own EA's and strategies.

(page 2)

In this VLOG we will talk about:

- Tracking both counts and dollar amounts of wins/losses
- How to add metrics to the Strategy Tester Visualization window (to show us, at a glance, how well the strategy works under current test conditions.)
- Optimization using "Custom max criterion" (to see what that's all about)
- And then the addition of CTRL+Q to quickly exit the Expert Advisor with a few keystrokes.

Let's first take a look at code changes. Then we'll see the changes in action. Note that the code for this video is only available on my website. You can also find a transcript of this video there as well. Now let's do an overview of the code changes in WinMerge.

02 – Code View in WinMerge

To show the differences in the code between versions 05 and 06, we will do a quick walk-thru summary using WinMerge. After this summary, we will go through the added code in more detail in MetaEditor.

Remember that in WinMerge changed code is highlighted yellow on both sides, and additions (or new code) is gray on the left side and yellow on the right side.

- The first thing we note is the addition of a keycode for the letter Q. This is a compiler definition rather than a constant because a switch statement demands a "constant," but not a variable constant. Yes, I don't fully understand the logic behind that myself.
- Scrolling down, we see the OnChartEvent() event.
  - This event detects when events outside the script occur. This can be a keydown event, as we see here. Events can also include
    - mouse movements
    - chart clicked
    - chart object clicked
    - drag
    - window resize
    - custom events
    - and more . . .

- Scrolling down, we see the OnTester() event.
  - This event is only triggered from the Strategy Tester at the end of a test. In this event we calculate and display statistical metrics that are important to us on the Strategy Tester Visualization window.
  - Toward the bottom we see where the customMaxCriterion is set and returned from the OnTester() event.
- Scrolling down, we see the OnTradeTransaction() event.
  - The IN section now dumps some interesting information to the system log file.
  - The OUT section detects when a trade is completed via SL or TP. It runs the TradeCompleted() function, which we will see in a moment.
- The next section of change shows the DrawLabel() function.
  - This simply outputs text to the visualization window at coordinates x and y, which are measured in pixels.
- The next section shows the Expectancy calculation.
  - Note that this gives the same value as does MT5. It is here as a convenience for displaying on the Strategy Tester Visualization window as well as for adding to our log file.
- The final addition is the TradeCompleted() function.
  - This function is run from the OnTradeTransaction() event when either a SL or TP occurs. It is also run once, and only once, from the OnTester() event to capture any ongoing trade that hasn't yet closed at the end time of the test.
  - I also have "placeholder code," in a comment, for tracking wins & losses, both counts and dollar amounts. I will discuss why this is here when we hit the code.

03 – Code View in MetaEditor

I'm back in MetaEditor, and we're looking at the code for Rapid Doji EA 06.

Keycode and OnChartEvent()

As was just pointed out, we added a keycode for the letter Q. This is for a bit of code that I add to every EA and Indicator that I write, although it is slightly different for Indicators.[(1)] It defines a quick method to unload the running program from MT5. Let's take a look at the OnChartEvent() event where this is used.

In OnChartEvent(), we check if the id argument is of type CHARTEVENT_KEYDOWN. This is to differentiate from other event types, e.g. CHARTEVENT_OBJECT_DRAG. The lparam argument holds the keycode. If we only wanted the letter Q, we would be done; but what we want is to know if the CTRL key is currently pressed. Once we've determined that both conditions are met, that is the letter Q and the CTRL key, we run ExpertRemove() to exit the Expert Advisor.

OnTester(), customMaxCriterion, and GetExpectancy()

Now lets jump down to the OnTester() event. First we define some variables to be used to position labels

on the Strategy Tester Visualization window. Then we capture certain tester statistics into shorthand variables. The next section of code handles any trade left over at the end time of the current strategy test. The value it gives is the mark to market value—as though we closed the trade at the very end time that the test ended. Next we get the width of Strategy Tester Visualization window so that we can specify where our statistical metrics will be shown. I find the upper right hand corner to be useful. Now we can create the text to be displayed and determine where to position the labels.

Each of these "Values of Interest" are straight forward and exist on the MT5 [Backtest] tab, so I won't describe each one—with one exception: the winLossRatio, which we set as the customMaxCriterion value returned from this event.

Let's jump to the DrawLabel() function. We see that it creates an "OBJ_LABEL" visualization object. This is not unlike the DrawArrow() or DrawLine() functions we already have, so we won't go into too much detail. It simply places text upon the window at the specified x and y pixel coordinates. At present the color is hard coded to "Plum."

Now let's take a look at the calculation for the GetExpectancy() function.

The GetExpectancy() function calculates this double value. Note that MT5 already gives this value in the [Backtest] tab—it is called "Expected Payoff." I find it convenient to see it on the Strategy Tester Visualization window when it pops up. The formula is straightforward, so I won't discuss it here.

OnTradeTransaction(), IN/OUT, and SL/TP detection

Let's go to the OnTradeTransaction() event.

The IN section simply outputs useful information to the system log. This can be invaluable for debugging.

The OUT section runs the TradeCompleted() function whenever a trade is closed via SL, TP, or by the EXPERT itself, e.g. a signal from a moving average crossover in the opposite direction of the trade. Let's look at that code now.

The TradeCompleted() function first assigns shorthand variables based upon the ticket number such as profit, swap, and commission. The sum of these values becomes the total transaction Amount.

This next section is commented code. This code can be used to track important values real-time rather than use the TesterStatistics() function that we used in the OnTester() event. Why do this? In case we want to use these values for live calculations. For example, if we want to calculate Z-score, or if we want to display live values on control panel we could create, or to dump to a CSV file for use in Excel or plotting.

The final section of this function outputs a useful line to the system log file. This can be invaluable for debugging your code. We will look at the log file shortly.

04 – Code Changes in Action

Visual (At-a-glance) Statistics

Let's first run the code in the Strategy Tester to see the statistics on the right hand side of the Strategy Tester Visualization Window.

I'm in MetaTrader 5 on the [Settings] tab of the Strategy Tester panel (CTRL+R). We will use the same

parameters as in previous videos.

Hitting the [Start] button starts the test. When the visualization window pops up, we see our statistics at the right hand side of the window. Excellent!

Note that these statistics will not show up under normal use of the EA because they are generated in the OnTester() event. Of course, you could move the code to some other event if you want to create a control panel of sorts.[2]

Let's verify these values against the [Backtest] tab in MT5:

- first column – Recovery factor – the value we expect
- middle column – Expected Payoff – the value we expect
- right column – OnTester result – this is the winLossRatio, and it has the value we expect
- Even the Total Net Profit is correct.

So, to sum up, we have made certain statistics that interest us on the Strategy Tester Visualization window so that we don't have to refer back to the [Backtest] tab every time we run a strategy test.

Show The Log Files

Now let's take a look at the log file to see the IN and OUT *detail* lines that we added in the code. Click on the [Journal] tab in the Strategy Tester Visualization Window.

We can scroll back through the log to see inputs that the system added as well as the inputs we added. I find the [Journal] tab view rather cumbersome, so I look at the log file using my preferred editor[3] because I developed a template to color code keywords and values.

Right-click → Open to open the directory to the log file. Double-click to edit.

Now, if I want to see all reasons for exits, I can search for the keyword "Reason," which I added in the OUT *detail* line. Now, if I only want to see only those lines, I can delete all lines that don't contain the word Reason. We see that all the closed trades, except the final one, was done through a SL. That makes sense since our EXIT is a trailing stop. The final close of CLIENT is the Mark to Market value at the end of the test session. This is exactly what we expect for an ongoing trade.

Now let's see how customMaxCriterion can be used when optimizing.

Running to Optimizer using customMaxCriterion

Let's go back to MT5 at the Strategy Tester [Settings] tab.

Here I change the optimization drop down to "Slow complete algorithm." I generally use this option when optimizing, even though it is slower, because I want to see all of the results—every time. For a really long optimization where the precision of the values doesn't matter, I might use the "Fast generic based algorithm." In general, however, I recommend you use "Slow complete algorithm" until you really have a feel for the Strategy Tester.

A *sorting* pull-down has now been enabled. Let's choose Custom Max.

Now switch to the [Inputs] tab. Let's vary the "Max points to keep SL from prices while trailing." Check the box and set the Start to 2000, the Step to 100 points, and the Stop to 3000. Keep in mind that this value is in points.[4]

One final setting should be changed. Back in the [Settings] tab, let's set the Modeling to 1-Minute

OHLC. This tells the Strategy Tester to use more precision of prices when calculating results. The reason for this is because I got rather strange results from the optimizer using “Open Prices Only.” While 1-Minute OHLC takes longer, we are at a point where meaningful results matter more than quick results. In point of fact, this optimization takes 10 seconds on my slow computer.

Now that the optimization has concluded, we see the [Optimization Results] tab. The first thing to note is the Result column. This contains the value of Custom Max Criterion, which in our case is the winLossRatio. We want this value to be as large as possible, which in this case is 1.36. However, we also want the Profit to be as large as possible, which in this case is not the same test pass as the best winLossRatio. Thus, we can see that optimizing an EA becomes somewhat an art-form, i.e. there is no one best answer for all parameters.

While we’re in the [Optimization Results] tab, we should show other important statistics that will interest us. To do this, right click and choose each column from the list.

Now we have even more methods to choose the best input parameters for our EA. This underscores what I just said about optimizing being an art-form.

05 – Wrapping up

(read from slide)

Final Notes (not in video)

1. Note that we don’t have many trades, and we’re at 50% chance. This is **not** a great strategy! But it wasn’t meant to be. This strategy was intended for instruction only.

Footnotes to This Document

(1) ChartIndicatorDelete() is used from an indicator instead of ExpertRemove().

(2) The OnTimer() event firing every second is what I use for control panels.

(3) VIM is my preferred editor for log files. It’s extremely powerful and extensible, but it does have a learning curve. I use Genie for general coding of MQL.

(4) Also, keep in mind that having a trailing stop more than 2 to 3 ATR is somewhat meaningless.