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

Introduction

This is the informal video blog.

In part 2 of this VLOG series, we will build upon the Expert Advisor (EA) code from part 1.

I call this part: Show me the Doji's!

In this VLOG we will add the OnTick() event, add code to check when a new bar is created, then place an arrow on the chart under each bar that is a Doji.

(page 2)

In this video we will add the OnTick() event and a few functions.

Now let's turn to the code.

Open MetaEditor

I have added the additional event and functions directly into the code. You can quickly jump to any function or event by pressing ALT+M. Notice on the left-hand side of the popup menu that events have a little yellow arrow next to them.

Let's jump there now.

First note that the OnTick() event runs or "fires" every single tick that comes in from your broker to your MT5 platform. You can get many ticks in a single second during volatile trading times, e.g. at the New York open.

Because we are interested only in bars, we must detect when a new bar is formed. That is the purpose of the IsNewBar() function. Let's take a look at this function. Remember, we can jump quickly to this function by pressing ALT+M.

The IsNewBar() function detects the very tick at which a new bar begins forming. Note that the new bar is not complete—it only has a single tick at this point. So what IsNewBar() also tells us is that the prior bar has completed. Invariably, this is the bar we wish to know.

The input arguments to IsNewBar() are the symbol and period. Note that both input arguments are constants. I like to make any variable a constant that I can because the compiler will alert me quickly if I try to assign a value to a constant. This stringent coding practice alerts me to potential issues early in my code development.

The variable priorBarOpenTime needs some explanation because it has a static scope. The documentation says this about Static variables:

> Local variables declared with the static keyword retain their values throughout the function lifetime. With each next function call, such local variables contain the values that they had during the previous call.

So, priorBarOpenTime acts rather like a global variable in that it is not reinitialized each time the IsNewBar() function is called.

The next line pulls a value from the MT5 platform called SERIES_LASTBAR_DATE. The documentation says:

Open time of the last bar of the symbol-period

. . . and that is exactly what we wanted. So, when the prior bar's open time differs from the current bar's open time, we know we have a new bar.

This next bit of code suppresses a true value when the EA first starts. There is a shorthand way to write this using the ternary operator, which is a little confusing to look at, so I wrote it out completely for this VLOG episode.

Finally, I update the "hold" value, which is the prior bar time to the current bar time.

Let's return to the OnTick() event by pressing ALT+M.

We know we have a new bar, so it's time to pull in fresh data for rates and indicators. Let's take a look at the RefreshData() function.

For this EA, I want to access rates data and the ATR indicator data. This function pulls these data into global arrays. Let's take a look at these definitions.

We're at the top of the code again.

First I want to point out the macro NUM_RATES which I define to be 3. You will soon see how this is used in the code. What it means is that I pull in the last 3 rates and indicator values. I prefer to have this as a definable value in one location rather than hard code the number "3" in the code.

Next we see the g_handleATR variable. This is a handle to the ATR indicator function. It acts much like a file handle when opening a file.

Here we have the g_rates and g_atr dimensionless arrays. g_rates will hold an array of type MqlRates, and g_atr will hold an array of type double.

Finally, I define an "input" variable called InpAtrPeriod. An input variable can be changed by the user at run time; also, you can have MetaTester automatically adjust this value through optimization. 14 is a typical value of ATR.

Now let's jump to the OnInit() event to see how these variables are wired-up.

This new addition to the code shows how to open a handle to the internal ATR indicator.

Here is where I set the size of the arrays and specify that the arrays will be used as a series. This simply means that I am indexing the arrays such that the zeroth bar is the current bar, bar 1 is the first fully formed bar, and bar 2 is the one that precedes bar 1 in time.

Now we can make sense of RefreshData(). Let's jump back to it using ALT+M.

Here we see where NUM_RATES rates are pulled into the g_rates array, and here is where NUM_RATES values are pulled into the g_atr array.

Let's jump back to the OnTick() event.

The last piece of the puzzle is where we detect if the fully formed bar is a Doji. If it is a Doji, I want to draw an arrow under it. Note that I pass the first fully formed bar, which has an index of 1. Let's take a look at the IsDoji() function.

The IsDoji() function is simple. The size of the body and the size of the bar are computed. Finally, I ensure that the size of the body is less than or equal to 3% of the size of the bar.

Going back to the OnTick() event, when a Doji is fully formed, the DrawArrow() function is called. This draws a yellow arrow under each Doji bar. Note that I pass the time from the first element of the g_rates array, which means the first fully formed bar. For positioning under the bar, I pass the low value, but I

subtract roughly one-third of the ATR value. This gives a little space under the bar rather than touch it.

Before jumping to the DrawArrow() code, let's scroll to the top to see a few defined constants.

For the purpose of this EA, I only care about the current chart, and we won't have any sub-windows. Thus, I set these constants to their default values. Now press ALT+M to jump to the DrawArrow() function.

Any object drawn on a chart must have a unique name, so I have created a sequence number using a static integer, out of which I build the name variable.

After I create the arrow at the specified time and price, I set the attributes of color, code (which corresponds to one of these two constants) and width (which determines the size).

Running In The Tester

Now I am back MetaTrader 5.

For the purposes of this VLOG series, this strategy uses the Daily time frame. That means that in order to see an arrow under each Doji bar, the current bar must finish, and it must be a Doji. Looking at the pound-dollar, Daily chart, the current bar is not fully formed, nor will it be a Doji. Thus, it could take days for the right conditions to occur.

Therefore, I will use *historical data* to ensure that our EA is detecting Dojis and displays the arrows properly. To do this, I open the strategy tester by choose the menu View → Strategy Tester, which has a shortcut of CTRL+R. I recommend you memorize this shortcut combination because you will use it frequently in your software development.

If we go to the Overview tab, we can choose *Visualize*. Now we can choose the expert Rapid EA 02.ex5.

For the purpose of this VLOG, choose pound-dollar (GBPUSD), and ensure the Daily pull-down menu is selected.

I will choose a Custom period of roughly five years, spanning from 2015 to 2020—January 1st of each year.

For modeling, it's OK to use "Open prices only," because it's far quicker than specifying each tick. Plus, this EA's code detects a new bar internally, so modeling every tick isn't useful at this stage of the development.

For backtesting, I typically use a Deposit of 10000 USD and leverage of 1:50.

Finally, ensure the "visual mode" is checked, which is should be because we chose Visualize on the Overview tab. This is what will display the chart with the drawn arrows.

Now press the Start button.

The chart that opens is exactly what we expect. The yellow arrows do seem to point at Doji bars. Excellent.

Wrapping up

(read from slide)

Final Notes (not in video)

- If you have any suggestions for improvement to this script or this video series, do let me know. Programming is a bit of an art form, and each programmer will do it their own way.