cmc markets

Excel RTD

1. About the Excel RTD app

The Excel RTD app lets you do two things:

- Put real-time data into Excel using only Excel's RTD() function. No macros; no programming; no XLL add-ins
- Send simple trading commands from VBA code in Excel

You can run multiple copies of the Excel RTD app for different accounts, and then combine the data for those accounts in a single spreadsheet.

The app is supplied with an example spreadsheet which lets you enter up to 5 account numbers, and then automatically displays a dashboard of equity and balance etc; symbol prices; and a consolidated list of open positions.

2 Displaying real-time data in Excel

2.1 Running the RTD app

In order to put real-time data into Excel you need to run the RTD app. The Excel formulas listed below will give blank values if the app is not running.

2.1.1 Available symbols

Information about the following symbols will be available in the Excel RTD app:

- **MT4/5**: the app will report all the symbols which are included in the MT4/5 market watch
- **Tradable**: the app will report all available symbols in the platform
- **All other platforms**: you configure which symbols the app reports using the app's Symbols menu.

2.2 Excel formula

Once the RTD app is running, you can use the following formula in Excel to insert a real-time feed of account, ticket, or price data. You simply need to fill in the account number, and the "property" which you want to display:

=RTD("FXBlueLabs.ExcelRTD", , "**account number**", "**property**")

For example, if your account number is 156734 and you want to display the account's balance, or the bid price of GBPUSD:

=RTD("FXBlueLabs.ExcelRTD", , "156734", "balance")
=RTD("FXBlueLabs.ExcelRTD", , "156734", "bidGBPUSD")

Please note: with some language settings – for example, Polish – Excel may want the sections of the formula to be separate by semi-colons instead of commas. For example:

=RTD("FXBlueLabs.ExcelRTD"; ; "**account number**"; "**property**")

2.3 Property names

The RTD app supplies data about the account (e.g. equity and balance), symbol prices, "tickets", bar history, and technical indicator values. The list of tickets includes both open positions and pending orders.

2.3.1 Account data properties

Property	Meaning
currency	The deposit currency of the account
balance	Account balance
equity	Account equity
pl	Floating profit/loss
usedmargin	Margin in use
freemargin	Free margin
tickets	Number of "tickets": open positions and pending orders

2.3.2 Price and symbol data properties

The app supplies the current ask and bid prices for all symbols configured in the app. For example, if the symbol name you are interested in is EURUSD, then the property name for its ask price is askEURUSD. For example:

```
=RTD("FXBlueLabs.ExcelRTD", , "156734", "askEURUSD")
```

Property	Meaning
bid*Symbol*	Bid price of symbol
ask*Symbol*	Ask price of symbol
high*Symbol*	Daily high of the symbol. Not available on all platforms. The definition of the day's start (e.g. GMT, or some other time zone) depends on the broker/platform.
low*Symbol*	Daily low of the symbol. Not available on all platforms. The definition of the day's start (e.g. GMT, or some other time zone) depends on the broker/platform.

The app also provides a count and a list of all configured symbols. For example, the following formulas return the number of available symbols and the name of the 5th symbol on the list (which can be in any order):

```
=RTD("FXBlueLabs.ExcelRTD", , "156734", "symbols")
=RTD("FXBlueLabs.ExcelRTD", , "156734", "s5")
```

Property	Meaning
symbols	Number of symbols
s*N*	Name of the nth symbol, e.g. EURUSD. The *N* value is an index between 1 and the total number of symbols

2.3.3 Ticket data properties

The app supplies the following information about each "ticket", i.e. each open position and pending order. The *N* value in each property name is an index between 1 and the total number of tickets (reported by the tickets property).

For example, you can get the symbol name and net profit of the 2nd ticket (if there is one) using the following formulas:

```
=RTD("FXBlueLabs.ExcelRTD", , "156734", "t2s")
=RTD("FXBlueLabs.ExcelRTD", , "156734", "t2npl")
```

Property	Meaning
t*N*t	Ticket number, i.e. the ID of the open position or pending order
t*N*a	Action: BUY, SELL, BUYLIMIT, SELLLIMIT, BUYSTOP, SELLSTOP
t*N*s	Symbol name
t*N*v	Volume
t*N*npl	Net profit (gross profit + commission + swap). Not applicable on pending orders, and reported as zero.
t*N*pl	Gross profit. Not applicable on pending orders, and reported as zero.
t*N*swap	Swap. Not applicable on pending orders, and reported as zero.

t/comm	Commission. Not applicable on pending orders, and reported as zero.
t/sl	Stop-loss price
t/tp	Take-profit price
t/op	Open/entry price
t/cp	Current price of symbol
t/cm	Order comment
t/mg	Order magic number (MetaTrader 4 only)
t/ot	Open time (as number of seconds since 1/1/1970)

2.3.4 Bar history

You can use the Excel RTD app to request recent price history from the platform. All values are bid prices. (Please note that this price history is not available on the tradable platform.)

The property name for bar history is as follows: @bh,*symbol*,*timeframe*,*data*,*shift*. For example, the following formula shows the high of the current EUR/USD H1 bar:

=RTD("FXBlueLabs.ExcelRTD", ,"156734", "@bh,EURUSD,H1,high,0")

2.3.4.1 Timeframe value

The bar timeframe can either be specified as a number of minutes – e.g. 60 for hourly bars – or you can use standard notations such as H1 or M3. The available timeframes are as follows:

Period	Timeframe value
M1	1
M2	2
M3	3
M4	4
M5	5
M6	6
M10	10
M12	12
M15	15

M30	30
H1	60
H2	120
H3	180
H4	240
H6	360
H8	480
D1	1440
D2	2880
W1	7200

2.3.4.2 Price data

You can request the following information about each bar:

Data	Meaning
time	Start time of the bar (in the format yyyy/mm/dd hh:mm:ss)
open	Open price
high	High price
low	Low price
close	Close price
range	Range from high to low
median	Average of high and low
typical	"Typical" price: average of high, low, and close
weighted	"Weighted" price: average of high, low, close, and close – i.e. double-weighting on the close value
change	Change in bar: close minus open, therefore negative for down bars and positive for up bars.
abschange	Absolute change value, i.e. change converted to a positive number if negative

2.3.4.3 Bar shift

The final part of the price history formula is the bar "shift", i.e. which bar to get information about. Bars are numbered with the newest at zero, and increasing in order of age. In other words, bar 0 is the current in-progress bar; bar 1 is the last complete bar etc.

Therefore, the close price on bar 0 (for any timeframe) is the current bid price. In effect, the following two formulas are identical:

```
=RTD("FXBlueLabs.ExcelRTD", , "156734", "bidGBPUSD")
=RTD("FXBlueLabs.ExcelRTD", , "156734", "@bh,GBPUSD,60,close,0")
```

The amount of data available on each timeframe depends on the underlying platform, but will typically be around 250 bars.

2.3.5 Technical indicators

The Excel RTD app has some built-in indicator calculations which you can request using formulas. For example, the following formula will show 14-bar Relative Strength Index for GBP/USD M5:

```
=RTD("FXBlueLabs.ExcelRTD", , "156734", "@rsi,GBPUSD,M5,14,0")
```

Please note that the technical indicators are not available on the tradable platform.

The property name for a technical indicator starts with an indicator name such as @rsi or @ema, and is then followed by a list of parameters separated by commas.

The first two parameters for an indicator are always the symbol name and the timeframe, which can be specified either as H2 or as the equivalent number of minutes such as 120.

The last parameter is always the bar "shift". You will normally want to use a value of 0 for the shift, in order to get the current indicator value, but you can also use a shift of e.g. 1 to get the value of the indicator at the end of the previous bar. (The only exception are the swing-point indicators, which always return the latest swing point and do not use a shift parameter.)

Many indicators can be applied to different data values from each bar, e.g. the high price or even the bar range instead of the close price.

Please bear in mind that exponential moving averages and similar calculations are affected by the amount of available bars. For convenience, everyone always refers to the N value in such calculations as "N bars" (e.g. "21-bar EMA"), but this is not what it truly means. The N is a weighting factor, and a calculation such as an EMA always looks at the entire bar history which it has collected, but giving increased weight to the most recent N bars. Two calculations of an EMA can be different – though only usually by small amounts – if they are using different amounts of bar history.

2.3.5.1 @sma – Simple Moving Average

You can calculate a simple moving average (i.e. arithmetic mean) using the @sma indicator. For example, the following formula does an average of the median prices for the last 10 bars on GBP/USD M5:

=RTD("FXBlueLabs.ExcelRTD", , "156734", "@sma,GBPUSD,M5,median,10,0")

The indicator's parameters are as follows:

Parameter	Meaning
symbol	Symbol name, e.g. GBPUSD
timeframe	Bar timeframe, as a number of minutes or a notation such as H1 or M3
data	Data to use from each bar, e.g. close or high
period	Number of bars to calculate the average over
shift	Bar shift, e.g. zero in order to get the current value of the indicator

2.3.5.2 @ema – Exponential Moving Average

You can calculate an exponential moving average using the @ema indicator. For example, the following formula does an average of the ranges of the last 21 bars on GBP/USD D1:

=RTD("FXBlueLabs.ExcelRTD", , "156734", "@ema,GBPUSD,1440,range,21,0")

The indicator's parameters are as follows:

Parameter	Meaning
symbol	Symbol name, e.g. GBPUSD
timeframe	Bar timeframe, as a number of minutes or a notation such as H1 or M3
data	Data to use from each bar, e.g. close or high
period	Number of bars to calculate the average over
shift	Bar shift, e.g. zero in order to get the current value of the indicator

2.3.5.3 @smma – Smoothed Moving Average

You can calculate a smoothed moving average using the @smma indicator. (A smoothed average with period N is the same as an exponential moving average with period 2N-1).

For example, the following formula does an average of the close prices of the last 21 bars on GBP/USD H1:

=RTD("FXBlueLabs.ExcelRTD", , "156734", "@smma,GBPUSD,H1,close,21,0")

The indicator's parameters are as follows:

Parameter	Meaning
symbol	Symbol name, e.g. GBPUSD
timeframe	Bar timeframe, as a number of minutes or a notation such as H1 or M3
data	Data to use from each bar, e.g. close or high
period	Number of bars to calculate the average over
shift	Bar shift, e.g. zero in order to get the current value of the indicator

2.3.5.4 @lwma – Linear-Weighted Moving Average

You can calculate a linear-weighted moving average using the @lwma indicator. For example, the following formula does an average of the ranges of the last 21 bars on GBP/USD D1:

=RTD("FXBlueLabs.ExcelRTD", , "156734", "@lwma,GBPUSD,1440,range,21,0")

The indicator's parameters are as follows:

Parameter	Meaning
symbol	Symbol name, e.g. GBPUSD
timeframe	Bar timeframe, as a number of minutes or a notation such as H1 or M3
data	Data to use from each bar, e.g. close or high
period	Number of bars to calculate the average over
shift	Bar shift, e.g. zero in order to get the current value of the indicator

2.3.5.5 @macd and @macdsig – MACD

You can calculate MACD (the difference between a "fast" EMA and a "slow" EMA) using the @macd indicator. You can also use @macdsig to get the smoothed "signal" value of the MACD indicator.

For example, the following formula calculates MACD for GBP/USD M30, using the standard 12-bar fast EMA and a 26-bar slow EMA, and applying the calculation to the high price of each bar:

=RTD("FXBlueLabs.ExcelRTD", , "156734", "@macd,GBPUSD,30,high,12,26,9,0")

The indicator's parameters are as follows:

Parameter	Meaning
symbol	Symbol name, e.g. GBPUSD
timeframe	Bar timeframe, as a number of minutes or a notation such as H1 or M3
data	Data to use from each bar, e.g. close or high
fast	Number of bars for the fast EMA
slow	Number of bars for the slow EMA
slowing	Smoothing period for the signal value
shift	Bar shift, e.g. zero in order to get the current value of the indicator

2.3.5.6 @atr – Average True Range

You can calculate average true range using the @atr indicator. For example, the following formula calculates the average true range of the last 21 bars on GBP/USD D1:

=RTD("FXBlueLabs.ExcelRTD", , "156734", "@atr,GBPUSD,D1,21,0")

The indicator's parameters are as follows:

Parameter	Meaning
symbol	Symbol name, e.g. GBPUSD
timeframe	Bar timeframe, as a number of minutes or a notation such as H1 or M3
period	Number of bars to calculate the average over
shift	Bar shift, e.g. zero in order to get the current value of the indicator

2.3.5.7 @rsi – Relative Strength Index

You can calculate Relative Strength Index using the @rsi indicator. For example, the following formula calculates 14-bar RSI on USD/JPY M3:

=RTD("FXBlueLabs.ExcelRTD", , "156734", "@atr,USDJPY,3,14,0")

The indicator's parameters are as follows:

Parameter	Meaning
symbol	Symbol name, e.g. GBPUSD
timeframe	Bar timeframe, as a number of minutes or a notation such as H1 or M3
period	Number of bars to calculate the indicator over
shift	Bar shift, e.g. zero in order to get the current value of the indicator

2.3.5.8 @stoch and @stochslow – Stochastic Oscillator

You can calculate the stochastic oscillator using the @stoch indicator. You can also calculate the slowed "signal" value for the indicator using @stochslow.

For example, the following formula calculates the oscillator on GBP/USD H2 bars, using standard parameters of (5,3,3) – i.e. K period of 5, D period of 5, slowing value of 3.

=RTD("FXBlueLabs.ExcelRTD", , "156734", "@stoch,GBPUSD,120,5,3,3,0")

The indicator's parameters are as follows:

Parameter	Meaning
symbol	Symbol name, e.g. GBPUSD
timeframe	Bar timeframe, as a number of minutes or a notation such as H1 or M3
k	K period for the calculation
d	D period for the calculation
slowing	Slowing period (moving average of D values)
shift	Bar shift, e.g. zero in order to get the current value of the indicator

2.3.5.9 @bbupper and @bblower – Bollinger bands

You can calculate "Bollinger" bands – a simple moving average plus/minus a number of standard deviations – using the @bbupper and @bblower indicators.

For example, the following formula calculates the upper band on GBP/USD M10, using an average of the close prices on the last 30 bars, and 2 standard deviations:

=RTD("FXBlueLabs.ExcelRTD", , "156734", "@bbupper,GBPUSD,10,close,30,2,0")

The indicator's parameters are as follows:

Parameter	Meaning
symbol	Symbol name, e.g. GBPUSD

timeframe	Bar timeframe, as a number of minutes or a notation such as H1 or M3
data	Data to use from each bar, e.g. close or high
period	Number of bars to calculate the indicator over
deviations	Number of standard deviations to calculate (e.g. 2)
shift	Bar shift, e.g. zero in order to get the current value of the indicator

2.3.5.10 @vol – Volatility (standard deviation)

You can calculate volatility – i.e. 1 standard deviation – using the @vol indicator.

For example, the following formula calculates the volatility of the last 21 bar-ranges on GBP/USD M10:

=RTD("FXBlueLabs.ExcelRTD", , "156734", "@vol,GBPUSD,10,range,21,0")

The indicator's parameters are as follows:

Parameter	Meaning
symbol	Symbol name, e.g. GBPUSD
timeframe	Bar timeframe, as a number of minutes or a notation such as H1 or M3
data	Data to use from each bar, e.g. close or high
period	Number of bars to calculate the indicator over
shift	Bar shift, e.g. zero in order to get the current value of the indicator

2.3.5.11 @cci – Commodity Channel Index

You can calculate the Commodity Channel Index using the @cci indicator.

For example, the following formula calculates CCI using the typical bar price for the last 14 bars on EUR/USD H1:

=RTD("FXBlueLabs.ExcelRTD", , "156734", "@cci,EURUSD,60,typical,14,0")

The indicator's parameters are as follows:

Parameter	Meaning
symbol	Symbol name, e.g. GBPUSD
timeframe	Bar timeframe, as a number of minutes or a notation such as H1 or M3
data	Data to use from each bar. CCI is usually calculated on the "typical" bar price
period	Number of bars to calculate the indicator over
shift	Bar shift, e.g. zero in order to get the current value of the indicator

2.3.5.12 @high – Highest bar value

You can calculate the highest of a series of bar values using the @high indicator. For example, the following formula calculates the highest high during the last 20 GBP/USD D1 bars:

=RTD("FXBlueLabs.ExcelRTD", , "156734", "@high,GBPUSD,D1,high,20,0")

The indicator can be applied to any bar data. For example, you can find the highest low as well as the highest high. You can also use it to find the bar with the largest range or change.

The indicator's parameters are as follows:

Parameter	Meaning
symbol	Symbol name, e.g. GBPUSD
timeframe	Bar timeframe, as a number of minutes or a notation such as H1 or M3
data	Data to use from each bar, e.g. close or high
period	Number of bars to calculate the indicator over
shift	Bar shift, e.g. zero in order to get the current value of the indicator

2.3.5.13 @low – Lowest bar value

You can calculate the lowest of a series of bar values using the @low indicator. The indicator can be applied to any bar data. For example, you can find the lowest high as well as the lowest low. You can also use it to find the bar with the smallest range or change.

The following formula calculates the smallest D1 bar range during the last 20 GBP/USD D1 bars:

=RTD("FXBlueLabs.ExcelRTD", , "156734", "@low,GBPUSD,D1,range,20,0")

The indicator's parameters are as follows:

Parameter	Meaning
symbol	Symbol name, e.g. GBPUSD
timeframe	Bar timeframe, as a number of minutes or a notation such as H1 or M3
data	Data to use from each bar, e.g. close or high
period	Number of bars to calculate the indicator over
shift	Bar shift, e.g. zero in order to get the current value of the indicator

2.3.5.14 @swingh and @swingl – Swing points ("fractals")

You can calculate the most recent swing points using the @swingh and @swingl indicators. A swing point is defined as a bar with lower highs either side of it (or higher lows, for @swingl). These swing points are similar to the MT4 "Fractals" indicator.

For example, the following formulas finds the most recent swing-high and swing-low prices on GBP/USD M5, using a 5-bar swing (2 bars either side of swing point) and not allowing an "unconfirmed" swing involving the current bar:

=RTD("FXBlueLabs.ExcelRTD", , "156734", "@swingh,GBPUSD,M5,high,2,0")
=RTD("FXBlueLabs.ExcelRTD", , "156734", "@swingl,GBPUSD,M5,low,2,0")

The indicator's parameters are as follows. Please note that the @swingh and @swingl do not have a bar "shift" parameter; they only return the most recent swing price.

Parameter	Meaning
symbol	Symbol name, e.g. GBPUSD
timeframe	Bar timeframe, as a number of minutes or a notation such as H1 or M3
data	Data to use from each bar. You normally use "high" with @swingh, and "low" with @swingl. However, you can use any value; for example, you can use "high" with @swingl to find a bar which has a lower high than the bars around it.
swingbars	Number of higher/lower bars required either side of the swing bar. The usual value is 2, for a five-bar swing consisting of two lower highs/higher lows either side of the swing bar, but you can use any value from 1 (i.e. three-bar swing) upwards.
unconfirmed	Either 0 or 1. Zero ignores the current bar and only allows "confirmed" swings. 1 includes the current bar, and allows "unconfirmed" swings which can change depending on price movements during the current bar.

2.3.5.15 @keltupper and @keltlower – Keltner channels

You can calculate Keltner channels using the @kelternupper and @keltnerlower indicators. A Keltner channel is an exponential moving average plus/minus average true range.

For example, the following formula calculates the lower Keltner channel on GBP/USD H1, using a 20-bar EMA minus half of 10-bar ATR:

=RTD("FXBlueLabs.ExcelRTD", , "156734", "@keltlower,GBPUSD,60,close,20,10,0.5,0")

Because Keltner channels are simply a combination of an EMA and ATR, the formula above is equivalent to the following:

=RTD("FXBlueLabs.ExcelRTD", , "156734", "@ema,GBPUSD,60,close,20,0") – (RTD("FXBlueLabs.ExcelRTD", , "156734", "@atr,GBPUSD,60,10,0") * 0.5)

The indicator's parameters are as follows:

Parameter	Meaning
symbol	Symbol name, e.g. GBPUSD
timeframe	Bar timeframe, as a number of minutes or a notation such as H1 or M3
data	Data to use from each bar, e.g. close or high, for calculating the EMA
emaPeriod	Number of bars to use for the EMA
atrPeriod	Number of bars to use for the ATR
atrMultiple	Multiples of ATR to add to/subtract from the EMA
shift	Bar shift, e.g. zero in order to get the current value of the indicator

2.3.6 Other properties

Other miscellaneous data items provided by the Excel RTD app are as follows:

Property	Meaning
LastUpdateTime	Time of the last update from the RTD app. Will report 1/1/2000 if the RTD app is not running for the account.

2.4 Symbol names and standardisation

By default the RTD app uses standardised symbol names. These may be different to the symbol names which your broker uses in your trading platform. For example, your broker's symbol names may have a suffix such as cx or mn, e.g. EURUSDcx or EURUSDmn.

By default, all forex symbols are converted to the form AAABBB. For example, a name such as EURUSDnm or EUR/USD will be converted by default to EURUSD. You can turn off this standardisation by un-ticking the option "Use standardised symbol names" in the app.

This setting is intended for spreadsheets where you are collecting data from multiple accounts on different brokers/platforms (by running multiple copies of the RTD app), and the brokers/platforms use different symbol names.

For example, you might have something like the following spreadsheet where there are account numbers in columns B onwards, and symbol names in rows 2 onwards. You can then have a formula which uses the symbol names in column A without having to adjust for one account using EUR/USD and the other using EURUSDfx etc.

	A	B	C
1	Symbol/Account	12376522	265823654
2	EURUSD	[see below]	
3	GBPUSD		

In cell B2: =RTD("FXBlueLabs.ExcelRTD", , B$1, CONCATENATE("bid", $A2))

You could then fill the formula from cell B2 into B3, C2 etc and the cell references would automatically adjust.

(The CONCATENATE function in Excel simply joins two pieces of together. In the above example it is joining the text "bid" with the symbol name in column A, to produce the property name bidEURUSD or bidGBPUSD.)

2.5 Ticket volumes

The RTD app reports the volumes on tickets as the nominal volume, not as a lot size. For example, a size of 0.20 lots will be reported as a volume of 20000.

(Unless you are using something like an MT4 mini account with a lot size of 10K instead of 100K, in which case 0.20 lots would be 2000 instead of 20000.)

3. Sending trading commands from Excel

The RTD app can also be used to send simple trading commands from VBA code in Excel. You can also programmatically read the same data which is available via the RTD function.

The following features can in fact be used from any programming language which supports COM, not just from VBA in Excel.

3.1 Reading data in VBA code

You can read data programmatically using the FXBlueLabs.ExcelReader object. For example:

```
Set reader = CreateObject("FXBlueLabs.ExcelReader")
reader.Connect ("156734")
MsgBox reader.Read("balance")
```

In other words: you create an instance of the FXBlueLabs.ExcelReader object; you use the Connect() function to link it to a specific account number; and then you can use the Read() function to get data about the account.

The property names for the Read() function are the same as the property names for use with Excel's RTD function.

3.1.1 Checking if a reader is successfully connected

You can successfully create the ExcelReader object and call the Connect() function even if no RTD app is currently running for that account.

In order to check whether data is actually available you can use Read() to make sure that properties such as balance are not blank, or you can read the LastUpdateTime property and check that the time is later than 1/1/2000.

3.1.2 Data consistency across multiple reads

If you are querying multiple pieces of data, particularly multiple pieces of ticket data, then you need to be careful about updates and data consistency. For example, consider the following code which loops through the ticket list:

```
For i = 1 To reader.Read("tickets")
   strSymbol = reader.Read("t" & i & "s")
   vVolume = reader.Read("t" & i & "v")
Next
```

It is possible for the following to happen:

- At outset there are 2 open tickets
- Between the two uses of Read(), i.e. between the execution of lines 2 and 3, one of the tickets is closed.
- Therefore, what used to be ticket 2 becomes ticket 1.
- As a result, at the end of the first loop, strSymbol will hold the symbol of the ticket which is now closed, and vVolume will hold the volume of the ticket which is still open.

To ensure consistency while reading multiple pieces of data, use Reader.ReaderLock(). This will suspend any changes to the data until you then use Reader.ReaderUnlock(). For example:

```
Reader.ReaderLock()
For i = 1 To reader.Read("tickets")
   strSymbol = reader.Read("t" & i & "s")
   vVolume = reader.Read("t" & i & "v")
Next
Reader.ReaderUnlock()
```

Don't forget to call ReaderUnlock() after using ReaderLock()...

3.2 Sending trading commands from Excel

As a security measure, commands are **turned off** by default. You must turn on the "Accept commands" setting in the RTD app in order to send commands

successfully. If this option is turned off then all commands will return "ERR:Commands not allowed".

You can send simple commands from Excel using the FXBlueLabs.ExcelCommand object. For example:

```
Set cmd = CreateObject("FXBlueLabs.ExcelCommand")
strResult = cmd.SendCommand("156734", "BUY", "s=EURUSD|v=10000", 5)
```

The SendCommand() function has four parameters:

- The account number (e.g. 156734)
- The command, e.g. BUY
- Parameters for the command, e.g. symbol and volume to buy
- The number of seconds to wait for a response

SendCommand() is synchronous. It returns either when the RTD app completes the command, or when the timeout period expires. (Timeout **does not** mean that the request such as a market order has been withdrawn/cancelled. It only means that the broker/platform has not responded within the acceptable time.)

The return value from SendCommand() is a string, beginning either with ERR: to indicate that an error occurred, or with OKAY:. The only exception to this is the TEST command, which returns the text HELLO.

3.2.1 Differences between trading platforms

There are some minor differences in the trading features which are currently supported on different platforms:

- "Magic numbers" are only valid for MT4 and MT5, and these parameters will be ignored on other platforms.
- Order comments are only available on some platforms.
- Stop-losses and take-profits are not currently supported on tradable

3.2.2 Commands and parameters

The parameters for a command are sent as a pipe-delimited string, consisting of a number of settings in the format name=value. The parameters can be listed in any order, and some parameters are optional.

```
cmd.SendCommand("156734", "BUY", "s=EURUSD|v=10000", 5)
```

Trading volumes are always specified as cash amounts, not as lot sizes. The format of symbol names depends on whether the "Use standardised symbol names" option is turned on in the RTD app.

3.2.2.1 TEST command

Simply returns the text HELLO if successful.

3.2.2.2 BUY and SELL commands

Submits buy or sell market orders. If successful, it returns the ID of the new ticket in the form OKAY:ticket-number

Parameter	Optional?	Meaning
s	Compulsory	Symbol name for the buy order
v	Compulsory	Trading volume
sl	Optional	Stop-loss price for the new position
tp	Optional	Take-profit price for the new position
comment	Optional	Comment for the new position
magic	Optional	Magic number for the new position

3.2.2.3 BUYLIMIT, SELLLIMIT, BUYSTOP, and SELLSTOP commands

Submits a new pending order. If successful, it returns the ID of the new ticket in the form OKAY:ticket-number

Parameter	Optional?	Meaning

S	Compulsory	Symbol name for the buy order
V	Compulsory	Trading volume
price	Compulsory	Entry price for the pending stop/limit order
sl	Optional	Stop-loss price for the new position
tp	Optional	Take-profit price for the new position
comment	Optional	Comment for the new position
magic	Optional	Magic number for the new position

3.2.2.4 CLOSE command

Closes an open position or deletes a pending order. Returns OKAY:okay if successful.

Parameter	Optional?	Meaning
t	Compulsory	ID of the position to be closed, or the pending order to be deleted.

3.2.2.5 PARTIALCLOSE command

Does a partial-close of an open position. Returns OKAY:okay if successful. Volumes larger than the position size are simply treated as a full close (not as a close plus a reverse for the remaining amount). Cannot be used on pending orders.

Parameter	Optional?	Meaning
t	Compulsory	ID of the position to be partially closed.
v	Compulsory	Volume to be closed, e.g. 20000

3.2.2.6 REVERSE command

Reverses an open position, e.g. closing an open sell and replacing it with a buy. Returns OKAY:okay if successful.

Parameter	Optional?	Meaning
t	Compulsory	ID of the position to be reversed
v	Optional	Volume for the new reversed position. If omitted, the volume of the existing position is used (i.e. symmetrical reverse)
sl	Optional	Stop-loss price for the new position
tp	Optional	Take-profit price for the new position
comment	Optional	Comment for the new position
magic	Optional	Magic number for the new position

3.2.2.7 CLOSESYMBOL command

Closes all open positions and pending orders for a specific symbol. Returns OKAY:okay if successful.

Parameter	Optional?	Meaning
s	Compulsory	Symbol name to close

3.2.2.8 CLOSEALL command

Closes all open positions and pending orders for all symbols. Returns OKAY:okay if successful. Please note that closing everything can require a substantial timeout.

Parameter	Optional?	Meaning
(none)		

For example:

```
cmd.SendCommand("156734", "CLOSEALL", "", 20)    ' 20-second timeout
```

3.2.2.9 ORDERSL command

Changes the stop-loss on an open trade or pending order. Returns OKAY:okay if successful.

Parameter	Optional?	Meaning
t	Compulsory	ID of the trade or pending order to be modified
sl	Compulsory	New stop-loss price, or 0 to remove any existing stop-loss

3.2.2.10 ORDERTP command

Changes the take-profit on an open trade or pending order. Returns OKAY:okay if successful.

Parameter	Optional?	Meaning
t	Compulsory	ID of the trade or pending order to be modified
tp	Compulsory	New take-profit price, or 0 to remove any existing take-profit

3.2.2.11 ORDERMODIFY command

Changes both the stop-loss and take-profit on an open trade or pending order. For pending orders, you can also alter the entry price.

Parameter	Optional?	Meaning
t	Compulsory	ID of the trade or pending order to be modified
p	Compulsory for pending orders	For pending orders, the new entry price for the order. Ignored and not required on open trades.
sl	Compulsory	New stop-loss price, or 0 to remove any existing stop-loss
tp	Compulsory	New take-profit price, or 0 to remove any existing take-profit

3.2.3 Standard error messages

Property	Meaning
ERR:Need account	Account value for SendCommand() is blank
ERR:Need command	Command value for SendCommand() is blank
ERR:No listening app	Cannot find an running instance of the RTD app for the specified account
ERR:No response within timeout	No response from the broker/platform within the specified number of seconds
ERR:Commands not allowed	The "Allow commands" option is not turned on in the RTD app
ERR:Unrecognised command	The command value for SendCommand() is not understood by the RTD app
ERR:Missing parameters	The command was missing one or more compulsory parameters

3.3 Asynchronous commands

It is also possible to send commands asynchronously rather than blocking execution of the VBA code until the command completes or times out. This works as follows:

- You use SendCommandAsync() instead of SendCommand().
- You periodically check the result of the asynchronous action using CheckAsyncResult().
- When finished (or when you have decided to give up) you free up the command memory using FreeAsyncCommand()

For example:

```
Set cmd = CreateObject("FXBlueLabs.ExcelCommand")
lCommandId = cmd.SendCommandAsync("10915", "BUY",
        "s=EURUSD|v=10000", 60)
strResult = ""
While strResult = ""
   strResult = cmd.CheckAsyncResult(lCommandId)
   If strResult = "" Then MsgBox "Still waiting..."
Wend
```

cmd.FreeAsyncCommand (ICommandId)

SendCommandAsync uses the same four parameters as SendCommand(), but returns a "command ID" for subsequent use with CheckAsyncResult() and FreeAsyncCommand(), instead of returning the command result. Please note that SendCommandAsync() still has a timeout value.

You must eventually call FreeAsyncCommand() after SendCommandAsync(), or else your code will leak memory, albeit in small amounts.

CheckAsyncResult() either returns a blank string if the command is still executing and has not reached its specified timeout or, if complete, it returns the same string response as SendCommand().


cmc markets