

Introduction

Application Programming Interface (API) commands in Excel VBA can retrieve information from and perform actions on Rotman Interactive Trader (RIT). Before using API commands, first initialize a new RIT API object using the following code:

```
Dim API As Rotman.RITAPI  
Set API = New Rotman.RITAPI
```

In general, the syntax for an API command is made up of 3 main parts: the object, the method and the parameter(s) (optional), as demonstrated in the following sample code:

```
API.CancelOrder (Range("ORDER_ID"))
```

↑ ↙ ↑
Object Method Parameter

Here, *API* is the object that actions are performed on. The method, *CancelOrder*, is the action to perform on *API* (in this case, the action is to cancel an order). The parameter, *Range("ORDER_ID")*, specifies details of the action (here, it specifies the order ID of the order to cancel).

Depending on the action that a method performs, it may or may not require a parameter. In the example above, *API.CancelOrder* requires a parameter to specify which order to cancel. There can also be more than one parameter for a method. However, as you will see in the following section, some methods do not require parameters because they perform general actions.

Other than performing actions, methods can also return a result (called the return value). It can be stored in a variable or a cell in an Excel worksheet for later reference. The example *API.CancelOrder* does not have a return value.

Retrieving General Information

General information from RIT can be retrieved by using the *API.Get[...]* commands. All *API.Get[...]* commands have a return value and therefore need to be stored. For example, to retrieve information on cash, use the following code:

```
Dim cash as variant  
cash = API.GetCash
```

The first line declares a variable named *cash* in VBA. The second line stores the value returned by *API.GetCash* in *cash*. Note that you must declare a variable before you can store values in it. Alternatively, you can assign the value to a cell in an excel worksheet:

Cells(1, 1) = **API**.GetCash
or
Range("CASH") = **API**.GetCash

where *CASH* is the name of a cell in the worksheet. When assigning values to cells, no variable declaration is needed.

Tip: if the value returned needs to be used again in the program, store it in a variable. If the value simply needs to be displayed, assign it to a worksheet cell.

Retrieving other general information from RIT follows the same pattern. The following table shows a summary of these commands:

Account Info	Sample API Code	Description (Result)
Cash	Cell(1, 0) = API.GetCash	
Buying Power	Cell(1, 1) = API.GetBP	
Net Liquidation Value	Cell(1, 2) = API.GetNLV	
Time Remaining	Cell(1, 3) = API.GetTimeRemaining	In seconds
Total Time	Cell(1, 4) = API.GetTotalTime	In seconds
Year Time	Cell(1, 5) = API.GetYearTime	In seconds

Retrieving Ticker Information

The following command retrieves information on the stock specified by the parameter "ticker" and stores it in an array with 11 items.

General Syntax:

API.GetTickerInfo(*ticker*)

Parameter:

Parameter	Description	Possible Values
ticker	Ticker symbol of a stock	"ALGO", "CRZY", Range("TICKER"), etc.

Return Value: An 11 dimension array that contains the following data in order:

position	last	bid size	bid	ask	ask size	last size	volume	cost	unrealized P/L	realized P/L
0	1	2	3	4	5	6	7	8	9	10

Sample Code:

```
Dim stockData as variant
stockData = API.GetTickerInfo(Range("TICKER"))
```

```
For I = 0 To 10
```

```
Range("TICKERDATA").Offset(0, I + 1) = stockData(I + 1)  
Next I
```

Sample Code Description:

Fill *stockData* with an array of API data of the stock in the cell named *TICKER*. Then, use a for loop to display the 11 items contained in *stockData* in an Excel worksheet, provided that *TICKERDATA* has been defined as the name of a range of cells.

If you want to display a specific item in *stockData*, for example, the volume, use the following syntax:

```
Cell (1, 1) = stockData(7)
```

where the number 7 is the position of volume in the array. Refer to the table above for the position of other items. Note that the first item's position is 0.

Adding An Order

The following command adds an order to RIT.

General command Syntax:

```
API.addorder(ticker, bid size, bid price, buy_sell, lmt_mkt)
```

Parameters:

Parameter	Description	Possible Values
Ticker	Ticker symbol of a stock	"ALGO", "CRZY", Range("TICKER") etc.
Bid size	Bid size	500, 10, Range("SIZE"), Cells(2,3) etc.
Bid price	Bid price	10.00, 15.25, Range("PRICE"), Cells(3, 4), etc.
buy_sell	Buy or sell an order	API.BUY or API.SELL
lmt_mkt	Type of an order	API.LMK or API.MKT

Return Value: True or False

Sample Code:

```
Dim status as variant  
status = API.addorder("ALGO", Range("BIDSIZE"), Range("BIDPRICE"), API.BUY, API.LMT)
```

Sample Code Description:

Submit a limit buy order for the stock *ALGO* with the size found in the cell *BIDSIZE* at the price found in the cell *BIDPRICE*. Assign True to the variable *status* if the order is successful, assign False otherwise.

Cancelling An Order

The following command cancels an order based on the order ID specified by the parameter.

General Syntax:

API.CancelOrder (*order_id*)

Parameter:

Parameter	Description	Possible Values
order_id	Order ID	3142, 2323, Range("ORDER_ID"), etc.

Return Value: None

Sample Code:

```
API.CancelOrder (Range("ORDER_ID"))
```

Sample Code Description:

Cancel the order specified in cell *ORDER_ID*

Cancelling an Order Expression:

The following command cancels all orders that satisfy the expression specified in the parameter.

General Syntax:

API.CancelOrderExpr(*order_expr*)

Parameters:

Parameter	Description	Possible Values
order_expr	Order expression.	"price > 20.00", "volume == 4.00", "price > 20.00 and volume == 400", etc.

Return Value: None

Sample Code:

```
API.CancelOrderExpr ("price > 20.00 and volume = 400")
```

Sample Code Description:

Cancel the all orders that have a price greater than \$20.00 and a volume equal to 400.