MQL4 COURSE

By Coders' guru
www.forex-tsd.com

-17-

Your first script

It was a long path for you to reach this lesson, Congratulations!
You learnt the basics of MQL4 language then you wrote your first indicator and your first expert advisor.
I hope you enjoyed the journey and interested to continue the road with me.
Today we will create our first script which will simply set the ***stoploss*** and ***takeprofit*** values for all the already opened orders on the chart.
It's useful if you are lazy to set manually your ***stoploss*** and ***takeprofit*** values from the ***Modify or Delete Orders*** window for every order.

What are we waiting for? Let's script!

What's a MQL4 script?

The scrip is a program you run once to execute an action and it will be removed immediately after the execution.
Unlike the expert advisors which will be called every time in a new tick arrival, the script will be executed only for once.
And unlike the indicators which have the ability to draw lines on the chart, the script has no access to indicator functions.
In short the script is an expert advisor which you can run only once.

Let's create our first script!

Wizards again!

With the help of the ***New Program*** wizard we will create the backbone of our scrip.
Bring the ***New Program*** wizard by choosing ***New*** from ***File*** menu or by clicking CTRL+N hot keys (Figure 1).

We are going to create a script, so choose ***Scrip program*** in the wizard and click ***next.***
That will bring the second step wizard (Figure 2).
Fill the Name, Author and Link fields with what you see in the figure 2 (or whatever you want). Then click finish.

Figure 1 – New Program wizard

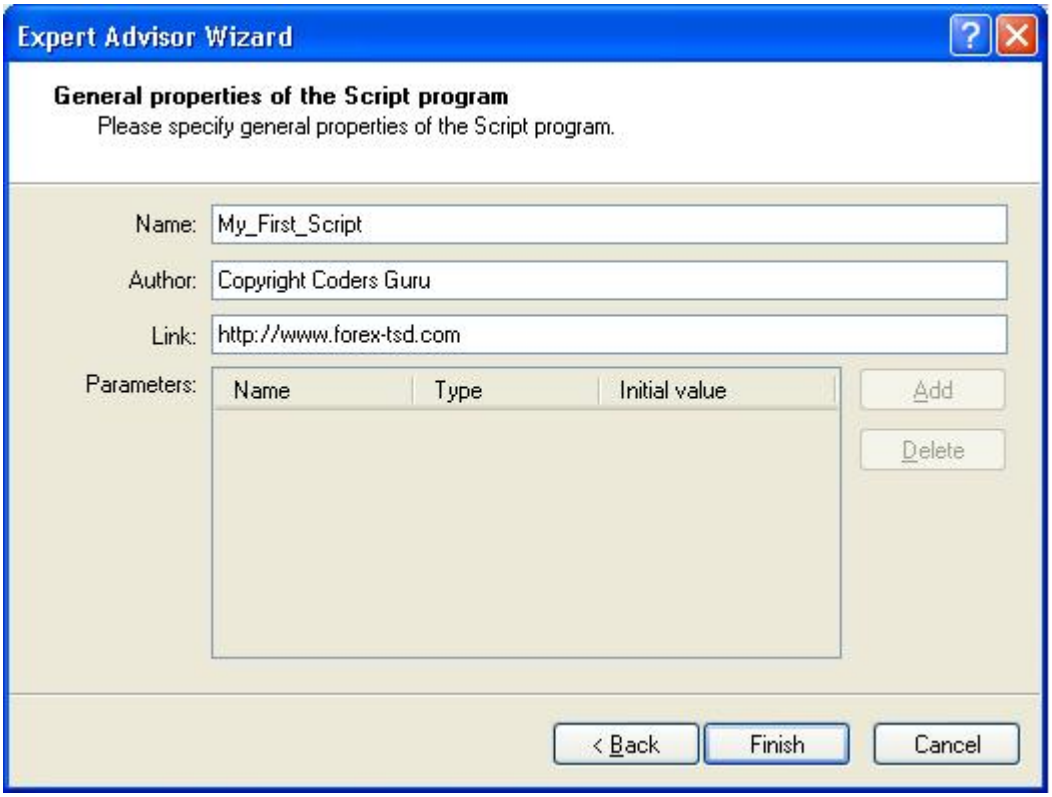

Figure 2 – Step 2

By clicking ***Finish*** button the wizard will write some code for you and left the places to write your own code, this is the code we have got from the wizard.

```
//+------------------------------------------------------------------+
//|                                                My_First_Script.mq4 |
//|                                             Copyright Coders Guru |
//|                                           http://www.forex-tsd.com |
//+------------------------------------------------------------------+
#property copyright "Copyright Coders Guru"
#property link      "http://www.forex-tsd.com"

//+------------------------------------------------------------------+
//| script program start function                                    |
//+------------------------------------------------------------------+
int start()
  {
//----

//----
   return(0);
  }
//+------------------------------------------------------------------+
```

Note: As you can easily notice from the above code that the wizard hasn't added the ***init()*** and ***deinit()*** functions and only added the ***start()*** function.
That's because it's rare to find a script needs to execute code in the program initialization and de-initialization because the ***start()*** function itself will be executed for once.
But that's not mean you can't add ***init()*** and ***deinit()*** functions in a script. You can add them if you want.

Now, we have to add our code to make our script more useful.

This is the code we have added to the above wizard's generated code (our added code marked by the **bold** font):

```
//+------------------------------------------------------------------+
//|                                                My_First_Script.mq4 |
//|                                             Copyright Coders Guru |
//|                                           http://www.forex-tsd.com |
//+------------------------------------------------------------------+
#property copyright "Copyright Coders Guru"
#property link      "http://www.forex-tsd.com"

#property show_inputs

#include <stdlib.mqh>

extern double    TakeProfit=250;
extern double    StopLoss=35;

//+------------------------------------------------------------------+
//| script program start function                                    |
//+------------------------------------------------------------------+
int start()
  {
//----
   int total,cnt,err;
```

```
   total = OrdersTotal();

   for(cnt=0;cnt<total;cnt++)
     {
        OrderSelect(cnt, SELECT_BY_POS, MODE_TRADES);

        if(OrderType()==OP_BUY)   // long position is opened
          {
               OrderModify(OrderTicket(),OrderOpenPrice(),
Bid-Point*StopLoss,Bid+Point*TakeProfit,0,Green);
               err=GetLastError();
               Print("error(",err,"): ",ErrorDescription(err));
               Sleep(1000);
          }

        if(OrderType()==OP_SELL)   // short position is opened
          {

OrderModify(OrderTicket(),OrderOpenPrice(),Ask+Point*StopLoss,
Ask-Point*TakeProfit,0,Red);
               err=GetLastError();
               Print("error(",err,"): ",ErrorDescription(err));
               Sleep(1000);
          }
     }
//----
   return(0);
  }
//+------------------------------------------------------------------+
```

Let's crack the code line by line as we used to do.

```
//+------------------------------------------------------------------+
//|                                                My_First_Script.mq4 |
//|                                             Copyright Coders Guru |
//|                                           http://www.forex-tsd.com |
//+------------------------------------------------------------------+
#property copyright "Copyright Coders Guru"
#property link      "http://www.forex-tsd.com"
```

Here the wizard has written the comment block which contains the script name we have been chosen, the copyright and link we have been typed.

Then two directive properties have been added according to the date we have entered, these are the copyright and link properties.

```
#property show_inputs
```

We have added the ***show_inputs*** directive property to our script.
Without this property the script will not accept inputs from the user therefore will not prompt an input window to the user as the one showed in figure 3.

In our script the user can set the ***TakeProfit*** and ***StopLoss*** values from the inputs window or he can leave the default values we have set in our code.

Note: If you intend to write a script which doesn't need inputs from the user, it's preferred to remove ***show_inputs*** directive property.

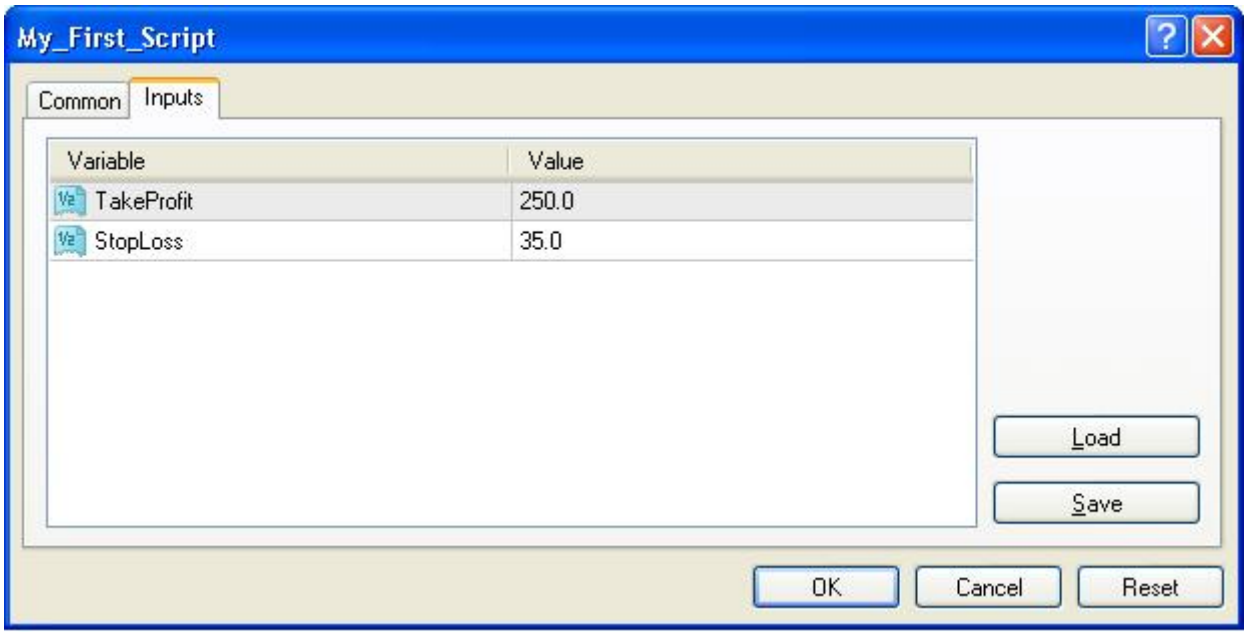

Figure 3 – Script Input window

```
#include <stdlib.mqh>
```

Later in our code we are going to use the function ***ErrorDescription*** which returns a string description of the passed error number. This function is included in the file ***"stdlib.mqh"***, so we have included this file to our program using the ***include*** directive. That means the content of the ***"stdlib.mqh"*** is a part of our code now and we can freely use the ***ErrorDescription*** function.

```
extern double    TakeProfit=250;
extern double    StopLoss=35;
```

Here we have declared two extern variables which the user can set them from the script inputs window (Figure 2) or he can leave the default values.
We will use these variables in our ***start()*** function to set the ***takeprofit*** and ***stoploss*** values of all the already opened order.

```
int start()
  {
//----
   int total,cnt,err;
   total = OrdersTotal();
   ....
  }
```

We are inside the ***start()*** function which will be called only one time when you attach the script to you chart.
We have declared three integer variables using a single line declaration method.
Then we assigned the return value of the ***OrdersTotal*** function to the ***total*** variable.
As you remember ***OrdersTotal*** function returns the number of opened and pending orders.

```
for(cnt=0;cnt<total;cnt++)
     {
      ....
     }
```

Here we enter a ***for*** loop starts from **0** and ends to the ***total*** of already opened and pending orders. We will use the loop variable ***cnt*** with the ***OrderSelect*** function to select every opened order by its index. In every cycle of the loop we increase the ***cnt*** variable by **1**.

```
OrderSelect(cnt, SELECT_BY_POS, MODE_TRADES);
```

We have selected the order by its index using the function ***OrderSelect*** and the loop variable ***cnt***.
We have to use the ***OrderSelect*** function before calling ***OrderType*** in the coming line. (Please review Appendix 2 – Trading functions).

```
if(OrderType()==OP_BUY)   // long position is opened
   {
    ....
   }
```

The ***OrderType*** function returns the type of selected order that will be one of: ***OP_BUY, OP_SELL, OP_BUYLIMIT, OP_BUYSTOP, OP_SELLLIMIT*** or ***OP_SELLSTOP***.
In the above ***if*** block we have checked the type of order to find is it a ***Buy*** order or not. If it's a ***Buy*** order, the code inside the block will be executed.

```
OrderModify(OrderTicket(),OrderOpenPrice(),
Bid-Point*StopLoss,Bid+Point*TakeProfit,0,Green);
```

It's a ***Buy*** order type, so we want to modify the values of ***stoploss*** and ***takeprofit*** to the values of ***StopLoss*** and ***TakeProfit*** variables the user has been set.

We use the ***OrderModify*** function which modifies the properties of a specific opened order or pending order with the new values you pass to the function.
We used these parameters:

ticket: We've got the current selected order ticket with ***OrderTicket*** function.

price: We've got the open price of the current selected order with ***OrderOpenPrice*** function.

stoploss: Here we have set the ***stoploss*** value of the current selected order to the value of the ***subtraction*** of the current ***bid*** price and the ***StopLoss*** value the user has been set.

takeprofit: Here we have set the ***takeprofit*** value of the current selected order to the value of the ***addition*** of the current ***bid*** price and the ***TakeProfit*** value the user has been set.

expiration: We haven't set an ***expiration*** date to our order, so we used **0**.

arrow_color: We have set the color of the arrow to ***Green***.

```
err=GetLastError();
Print("error(",err,"): ",ErrorDescription(err));
Sleep(1000);
```

We have assigned the returned value of the ***GetLastError*** function to the ***err*** variable.
The ***GetLastError*** returns the number of the last error occurred after an operation (***OrderModify*** in our case).
But we want to ***print*** not only the error number but the description of the error, so we have used the ***ErrorDescription*** function to get the string description of the error number and we have used the print function to output this message to the expert log.

Then we have used the ***Sleep*** function to give the terminal and our script the time to take their breath for one second (1000 milliseconds).

Note: ***Sleep*** function suspends the execution of the program for a specified interval, this interval passed to the function in milliseconds.

```
if(OrderType()==OP_SELL)   // short position is opened
   {
    ....
   }
```

In the above ***if*** block we have checked the type of order to find is it a ***Sell*** order or not.
If it's a ***Sell*** order, the code inside the block will be executed.

```
OrderModify(OrderTicket(),OrderOpenPrice(),Ask+Point*StopLoss,
Ask-Point*TakeProfit,0,Red);
```

It's a ***Sell*** order type, so we want to modify the values of ***stoploss*** and ***takeprofit*** to the values of ***StopLoss*** and ***TakeProfit*** variables the user has been set.
We use the ***OrderModify*** function again with these parameters:

ticket: We've got the current selected order ticket with ***OrderTicket*** function.

price: We've got the open price of the current selected order with ***OrderOpenPrice*** function.

stoploss: Here we have set the ***stoploss*** value of the current selected order to the value of the ***addition*** of the current ***ask*** price and the ***StopLoss*** value the user has been set.

takeprofit: Here we have set the ***takeprofit*** value of the current selected order to the value of the ***subtraction*** of the current ***ask*** price and the ***TakeProfit*** value the user has been set.

expiration: We haven't set an ***expiration*** date to our order, so we used **0**.

arrow_color: We have set the color of the arrow to ***Red***.

```
err=GetLastError();
Print("error(",err,"): ",ErrorDescription(err));
Sleep(1000);
```

The same as above explained lines.

```
return(0);
```

It's the time to terminate our script by using ***return*** function.

Let's compile our script by pressing **F5** hot key, as soon as you compile the script it will appear in the navigator window in the script category (Figure 4)

Note: The wizard has created the file ***"My_First_Script.mq4"*** and has placed it in the ***"experts\scripts"*** folder for us.
You have to put all of you scripts in this folder and compile them before using them in the terminal.

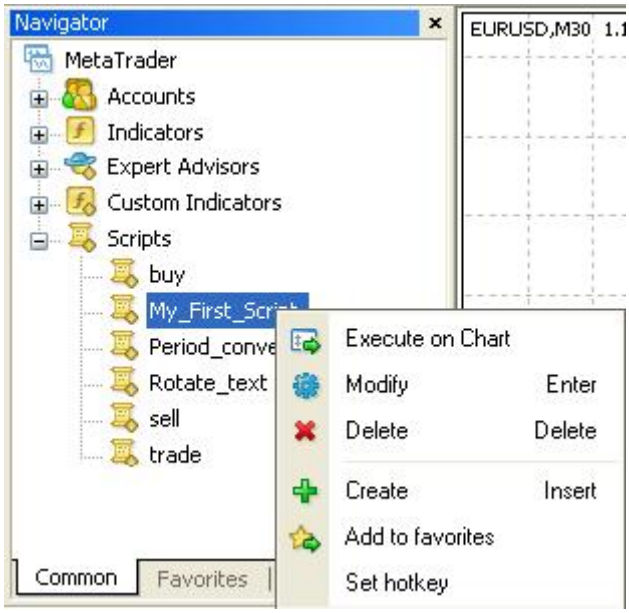

Figure 4 – Execute your script

To execute the script simply point it with your mouse and double click it, or click the left mouse button and a menu as showed in figure 4 will appear then choose ***Execute On Chart***.

The inputs window (Figure 3) will appear and you can set new values for ***StopLoss*** and ***TakeProfit*** or leave the default values.

I hope you enjoyed the lesson and found the first script we have created a useful one.

I welcome very much your questions and suggestions.

Coders' Guru
05-01-2005