**ซอสโค้ดระบบซื้อขายอัตโนมัติหลายสกุลเงิน**

ในส่วนนี้จะเป็นส่วนของซอสโค้ดของโปรแกรมที่ใช้ในโปรแกรม Metatrader 5 ของกลุ่ม ระบบซื้อขายอัตโนมัติหลายสกุลเงิน

#property copyright "Copyright 2020, MetaQuotes Software Corp."

#property link "https://www.mql5.com"

#property version "1.01" // check PositionsTotal()==0 before open order

#property version "1.02" // close buy at taget

#property version "1.03" // input lot p1-p2-p3

#property version "1.04" // edit close order by target

#property version "1.05" // add funtion open\_order and close\_order with POSITION\_TICKET

#property version "1.06" // add StopLoss

#property version "1.07" // add funtion LineNotify

#property version "1.08" // add status ea and remove Stoploss

#include <Trade\Trade.mqh>

#include <Math\Stat\Math.mqh> // class for libraries calculations mathematics.

CTrade trade ;

input string Step1 = "===== Currency pairs Setting =================";// • Step 1

int MG\_B = 1 ; //Magic Number

input string P1 = "EURUSDm" ; //Symbol P1

input double Lot\_P1 = 0.01 ; //Lot P1

input string P2 = "GBPUSDm" ; //Symbol P2

input double Lot\_P2 = 0.02 ; //Lot P2

input string P3 = "USDCHFm" ; //Symbol P3

input double Lot\_P3 = 0.03 ; //Lot P3

input string Step4 = "===== Indicator Bollinger bands Setting ================";//• Step 2

input double BB\_Period = 50 ; //BB Period

input double STD = 2.0 ; //BB STD

input double TP\_Target = 15.0; // TP\_Target(USD)

string version = "1.08";

input string Step5 = "===== Line Notify Setting ===================";//• Step 3

input bool Use\_LineNotify = false;//• Use LineNotify

string message = "",endl="\n";

string message\_2 = "";

input string token="gAQrlXr9ifJ1jHPdjk2djZ17GWtglfqENQLAeXzB7nI"; // Token

input string api\_url="https://notify-api.line.me/api/notify"; // URL API

// status EA Send to LineNotify

string status\_1 = "BUY";

string status\_2 = "SELL";

string status\_3 = "CLOSE";

string status\_4 = "EA Runing";

string status\_5 = "EA Stop";

//+------------------------------------------------------------------+

//| Expert initialization function |

//+------------------------------------------------------------------+

int OnInit()

{

//---

Call\_LineNotify(status\_4);

//---

return(INIT\_SUCCEEDED);

}

//+------------------------------------------------------------------+

//| Expert deinitialization function |

//+------------------------------------------------------------------+

void OnDeinit(const int reason)

{

//---

Call\_LineNotify(status\_5);

}

//+------------------------------------------------------------------+

//| Expert tick function |

//+------------------------------------------------------------------+

int MG\_S = MG\_B + 987 ;

void OnTick() {

MqlTradeRequest myRequest ;

MqlTradeResult myResult ;

MqlTick Tick ;

//---------| BUY

if(EntrySignal() == "Buy" && (PositionsTotal()==0)) {

BUY();

}//end if

//-------| SELL

if(EntrySignal() == "Sell" && (PositionsTotal()==0)) {

SELL();

}//end if

//-------| CLOSE

if (AccountInfoDouble(ACCOUNT\_PROFIT)>=+MathAbs(TP\_Target))

{ Call\_LineNotify(status\_3);

for(int i=PositionsTotal()-1;i>=0;i--)

{

if(PositionSelectByTicket(PositionGetTicket(i)))

{

trade.PositionClose(PositionGetInteger(POSITION\_TICKET));

message\_2 += "Close TP Target by Ticket = "+IntegerToString(PositionGetTicket(i)) + endl;

}

}

}

}

//end function

//+------------------------------------------------------------------+

// funtion OpenOrder Position BUY

void BUY(){

trade.Buy( Lot\_P1,

P1,

NULL, // execution price

NULL, // stop loss price

NULL, // take profit price

NULL // comment

);

trade.Buy( Lot\_P2,

P2,

NULL, // execution price

NULL, // stop loss price

NULL, // take profit price

NULL // comment

);

trade.Buy( Lot\_P3,

P3,

NULL, // execution price

NULL, // stop loss price

NULL, // take profit price

NULL // comment

);

message\_2 += "Buy :";

message\_2 += " Lot="+DoubleToString(Lot\_P1,2);

message\_2 += " Symbol="+P1;

message\_2 += endl;

message\_2 += "Buy :";

message\_2 += " Lot="+DoubleToString(Lot\_P2,2);

message\_2 += " Symbol="+P2;

message\_2 += endl;

message\_2 += "Buy :";

message\_2 += " Lot="+DoubleToString(Lot\_P3,2);

message\_2 += " Symbol="+P3;

message\_2 += endl;

Call\_LineNotify(status\_1);

}

//funtion OpenOrder Position SELL

void SELL(){

trade.Sell( Lot\_P1,

P1,

NULL, // execution price

NULL, // stop loss price

NULL, // take profit price

NULL // comment

);

trade.Sell( Lot\_P2,

P2,

NULL, // execution price

NULL, // stop loss price

NULL, // take profit price

NULL // comment

);

trade.Sell( Lot\_P3,

P3,

NULL, // execution price

NULL, // stop loss price

NULL, // take profit price

NULL // comment

);

message\_2 += "Sell :";

message\_2 += " Lot="+DoubleToString(Lot\_P1,2);

message\_2 += " Symbol="+P1;

message\_2 += endl;

message\_2 += "Sell :";

message\_2 += " Lot="+DoubleToString(Lot\_P2,2);

message\_2 += " Symbol="+P2;

message\_2 += endl;

message\_2 += "Sell :";

message\_2 += " Lot="+DoubleToString(Lot\_P3,2);

message\_2 += " Symbol="+P3;

message\_2 += endl;

Call\_LineNotify(status\_2);

}

//funtion CloseOrder

void CloseAll\_MG( int MG ) {

}//end function

string EntrySignal(){

string Output = "" ;

double C1 = 0, C2 = 0, C3 = 0, Price = 0 ;

int i = 0 ;

double Sum = 0, AVG = 0, UBand = 0, LBand = 0, SD = 0 ;

for( i = 0 ; i < BB\_Period ; i++ ) {

C1 = iClose( P1, PERIOD\_CURRENT, i ) ; C2 = iClose( P2, PERIOD\_CURRENT, i ) ; C3 = iClose( P3, PERIOD\_CURRENT, i ) ;

Price = C1 \* C2 / C3 ;

Sum += Price ;

}//end for

AVG = Sum / BB\_Period ;

double hold = 0 ;

for( i = 0 ; i < BB\_Period ; i++ ) {

C1 = iClose( P1, PERIOD\_CURRENT, i ) ; C2 = iClose( P2, PERIOD\_CURRENT, i ) ; C3 = iClose( P3, PERIOD\_CURRENT, i ) ;

Price = C1 \* C2 / C3 ;

hold += MathPow( ( Price - AVG ), 2 ) ;

}//end for

SD = MathSqrt( hold / ( BB\_Period - 1 ) ) ;

UBand = AVG + SD \* STD ;

LBand = AVG - SD \* STD ;

i = 1 ;

C1 = iClose( P1, PERIOD\_CURRENT, i ) ; C2 = iClose( P2, PERIOD\_CURRENT, i ) ; C3 = iClose( P3, PERIOD\_CURRENT, i ) ;

double PriceNext = C1 \* C2 / C3 ;

double BBW = ( ( PriceNext - LBand ) / ( UBand - LBand ) ) \* 100 ;

if( BBW > 100 ) Output = "Sell" ;

else if( BBW < 0 ) Output = "Buy" ;

Comment(

"\n Order Buy = " + DoubleToString( OrdersTotalMG( MG\_B ), 0 )

+ "\n Order Sell = " + DoubleToString( OrdersTotalMG( MG\_S ), 0 )

+ "\n\nPrice = " + DoubleToString( Price, (int)Digits() )

+ "\n\n---------------\nMA = " + DoubleToString( AVG, 5 )

+ "\nUBand = " + DoubleToString( UBand, 5 )

+ "\nLBand = " + DoubleToString( LBand, 5 )

+ "\n\nBBW = " + DoubleToString( BBW, 2 ) + " %"

) ;

return Output ;

}//end function

string ExitSignal(){

string Output = "" ;

double C1 = 0 ;

double C2 = 0 ;

double C3 = 0 ;

double Price = 0 ;

int i = 0 ;

double Sum = 0 ;

double AVG = 0 ;

double UBand = 0 ;

double LBand = 0 ;

double SD = 0 ;

for( i = 0 ; i < BB\_Period ; i++ ) {

C1 = iClose( P1, PERIOD\_CURRENT, i ) ;

C2 = iClose( P2, PERIOD\_CURRENT, i ) ;

C3 = iClose( P3, PERIOD\_CURRENT, i ) ;

Price = C1 \* C2 / C3 ;

Sum += Price ;

}//end for

AVG = Sum / BB\_Period ;

double hold = 0 ;

for( i = 0 ; i < BB\_Period ; i++ ) {

C1 = iClose( P1, PERIOD\_CURRENT, i ) ;

C2 = iClose( P2, PERIOD\_CURRENT, i ) ;

C3 = iClose( P3, PERIOD\_CURRENT, i ) ;

Price = C1 \* C2 / C3 ;

hold += MathPow( ( Price - AVG ), 2 ) ;

}//end for

SD = MathSqrt( hold / ( BB\_Period - 1 ) ) ;

UBand = AVG + SD \* STD ;

LBand = AVG - SD \* STD ;

//---------|

i = 0 ;

C1 = iClose( P1, PERIOD\_CURRENT, i ) ;

C2 = iClose( P2, PERIOD\_CURRENT, i ) ;

C3 = iClose( P3, PERIOD\_CURRENT, i ) ;

Price = C1 \* C2 / C3 ;

//double BBW = ( ( UBand - LBand ) / AVG ) \* 100 ;

double BBW = ( ( Price - LBand ) / ( UBand - LBand ) ) \* 100 ;

if( OrdersTotalMG( MG\_B ) > 0 && BBW >= 50 ) {

Output = "ExitBuy" ;

}//end if

if( OrdersTotalMG( MG\_S ) > 0 && BBW <= 50 ) {

Output = "ExitSell" ;

}//end if

return Output ;

}//end function

int OrdersTotalMG( int MG ) {

int Output = 0 ;

ulong t = 0 ;

for( int i = 0 ; i < (int)PositionsTotal() ; i++ ) {

t = PositionGetTicket( i ) ;

if( PositionSelectByTicket( t ) ) {

if( PositionGetInteger( POSITION\_MAGIC ) == MG ) {

Output++ ;

}//end if

}//end if

}//end for

return Output ;

}//end function

//funtion LineNotify

void LineNotify(string Massage)

{

string headers;

char post[],result[];

headers="Authorization: Bearer "+token+"\r\n";

headers+="Content-Type: application/x-www-form-urlencoded\r\n";

ArrayResize(post,StringToCharArray("message="+Massage,post,0,WHOLE\_ARRAY,CP\_UTF8)-1);

int res = WebRequest("POST", "https://notify-api.line.me/api/notify", headers, 10000, post, result, headers);

Print("Status code: " , res, ", error: ", GetLastError());

Print("Server response: ", CharArrayToString(result));

}

//funtion Call LineNotify

void Call\_LineNotify(string \_status)

{

if(!Use\_LineNotify) return;

message = "";

message += "สถานะ: "+\_status+".";

message += " \n แจ้งเตือนรายละเอียดดังนี้\n";

message += "AccountNumber : "+AccountInfoString(ACCOUNT\_NAME)+endl;

message += "Balance : "+DoubleToString(AccountInfoDouble(ACCOUNT\_BALANCE),2)+endl;

message += "Equity : "+DoubleToString(AccountInfoDouble(ACCOUNT\_EQUITY),2)+endl;

message += "Profit : "+DoubleToString(AccountInfoDouble(ACCOUNT\_PROFIT),2)+endl;

message += message\_2 +endl;

message\_2 = "";

LineNotify(message);

}