MQL4 COURSE

By Coders' guru
www.forex-tsd.com

-14

Váš první Expert Advisor - Část 2

Vítejte ve druhé části lekce vytváření vašeho prvního Expert Advisoru.
V předchozí části jsme převzali kód, vygenerovaný pomocníkem *new program wizard* a přidali váš vlastní kód , který si dnes vysvětlíme řádek po řádku.

Máte nasazeny *kódovací rukavice*? Pojďme tedy na to.

***Poznámka**: Musím opakovat, že náš expert advisor slouží pouze k výukovým účelům a nebude nám přinášet žádný zisk (ani k tomu není stvořen).*

Kód, který máme:

```
//+------------------------------------------------------------------+

//|                                                  My_First_EA.mq4 |
//|                                                      Coders Guru |
//|                                          http://www.forex-tsd.com |

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

#property copyright "Coders Guru"
#property link "http://www.forex-tsd.com"


//----vstupní parametry

extern double
extern double
extern double

                    TakeProfit=250.0;

                    Lots=0.1;
```

```
                              TrailingStop=35.0;
//+------------------------------------------------------------------+
//| expert – inicializační funkce |
//+------------------------------------------------------------------+
int init()

   {
//----


//----

     return(0);

   }
//+------------------------------------------------------------------+
//| expert – deinicializační funkce |
//+------------------------------------------------------------------+
int deinit()

   {
//----

//----

     return(0);
   }


int Crossed (double line1 , double line2)

     {
          static int last_direction = 0;
          static int current_direction = 0;


          if(line1>line2)current_direction = 1; //nahoru
          if(line1<line2)current_direction = 2; //dolů



          if(current_direction != last_direction) //změněno

          {
                    last_direction = current_direction;
                    return (last_direction);

          }

          else
```

```
    {
        return (0);
        }
        }

//+------------------------------------------------------------------+
//| expert – spouštěcí funkce |
//+------------------------------------------------------------------+

int start()
  {

//----


   int cnt, ticket, total;
   double shortEma, longEma;


   if(Bars<100)

      {
       Print("bars less than 100");
       return(0);

      }
   if(TakeProfit<10)

      {
       Print("TakeProfit less than 10");
       return(0); // kontrola - TakeProfit

      }


   shortEma = iMA(NULL,0,8,0,MODE_EMA,PRICE_CLOSE,0);
   longEma = iMA(NULL,0,13,0,MODE_EMA,PRICE_CLOSE,0);


   int isCrossed = Crossed (shortEma,longEma);


  total = OrdersTotal();
  if(total < 1)
  {
  if(isCrossed == 1)

        {
```

```
            ticket=OrderSend(Symbol(),OP_BUY,Lots,Ask,3,0,Ask+TakeProfit*Point,
            "My EA",12345,0,Green);
            if(ticket>0)
            {
            if(OrderSelect(ticket,SELECT_BY_TICKET,MODE_TRADES))
            Print("BUY order opened : ",OrderOpenPrice());

}

else Print("Error opening BUY order : ",GetLastError());

return(0);

             }

if(isCrossed == 2)

             {


                          ticket=OrderSend(Symbol(),OP_SELL,Lots,Bid,3,0,
            Bid-TakeProfit*Point,"My EA",12345,0,Red);
            if(ticket>0)
            {
            if(OrderSelect(ticket,SELECT_BY_TICKET,MODE_TRADES))
            Print("SELL order opened : ",OrderOpenPrice());

}

else Print("Error opening SELL order : ",GetLastError());

return(0);

    }
    return(0);
    }
    for(cnt=0;cnt<total;cnt++)

        {

OrderSelect(cnt, SELECT_BY_POS, MODE_TRADES);

if(OrderType()<=OP_SELL && OrderSymbol()==Symbol())

            {

                        if(OrderType()==OP_BUY) // long position je otevřena

                {
```

```
// má být uzavřena?
        if(isCrossed == 2)
        {
                        OrderClose(OrderTicket(),OrderLots(),Bid,3,Violet);
//uzavření pozice
return(0); // opuštění
}
// kontrola - trailing stop
                        if(TrailingStop>0)
                          {
                          if(Bid-OrderOpenPrice()>Point*TrailingStop)
                          {
                          if(OrderStopLoss()<Bid-Point*TrailingStop)
                          {
                          OrderModify(OrderTicket(),OrderOpenPrice(),Bid-
                          Point*TrailingStop,OrderTakeProfit(),0,Green);
                          return(0);
                          }
                          }
                          }
                          }
else // přechod k short position
                {
// má být uzavřena?
    if(isCrossed == 1)
    {
                        OrderClose(OrderTicket(),OrderLots(),Ask,3,Violet);
// uzavření pozice
return(0); // opuštění
}
// kontrola - trailing stop
                        if(TrailingStop>0)
                          {
```

```
                if((OrderOpenPrice()-
                Ask)>(Point*TrailingStop))
                {
                if((OrderStopLoss()>(Ask+Point*TrailingSt
                op)) ||
                (OrderStopLoss()==0))
                {


            OrderModify(OrderTicket(),Order
            OpenPrice(),Ask+Point*TrailingS
            top,
            OrderTakeProfit(),0,Red);
            return(0);
            }
            }
            }
            }
            }
            }
            return(0);
            }

//+------------------------------------------------------------------+
```

Co se skrývá za funkcí expert advisor.

Před zabořením se do objasňování našeho kódu si musíme vysvětlit, co stojí za myšlenkou funkce expert advisor. Každý expert advisor musí rozhodnout, kdy **vstoupit** na trh a kdy jej **opustit**. A podstatou každé funkce expert advisor je za jakých **podmínek** má vstupování a vystupování probíhat. Náš expert advisor patří k těm jednoduchým a jeho podstata je rovněž jednoduchá. Pojďme se tedy na ni podívat.

Používáme dva EMA indikátory, 8 denní (short EMA) a druhý 13 denní (long EMA).

Poznámka: *Používání těchto EMA či jakékoliv jiné myšlenky v této lekci neslouží jako doporučení, nýbrž pouze ke studijním účelům.*

Vstup (Open):

Náš expert advisor vstoupí na trh, když linie **short EMA** přetne linii **long EMA**, směr každé linie určí typ příkazu:
Pokud je **short EMA nad long EMA** proběhne **nákup** (long).
Pokud je **short EMA pod long EMA** proběhne **prodej** (short).

Otevřeme pouze **jeden** příkaz najednou.

Opuštění (Close):

Náš expert advisor uzavře příkaz **buy,** když **short EMA** přetne **long EMA** a **short EMA** je **pod long EMA**. A uzavře příkaz **sell,** pokud **short EMA** přetíná **long EMA** a **short EMA** je **nad long EMA**.

Náš příkaz (nákup nebo prodej) bude rovněž automaticky uzavřen, když budou dosaženy body **Take profit** nebo **Stop loss**.

Modifikace:

Kromě vstupování (opening) a opouštění (closing) trhu (positions), náš expert advisor má schopnost modifikace existujících pozic na základě bodu **Trailing stop**. Jak implementovat tuto funkci se dozvíme dále v této lekci.

Nyní budeme pokračovat v objasňování našeho kódu.

```
//----vstupní parametry

extern double       TakeProfit=250.0;
extern double       Lots=0.1;
extern double       TrailingStop=35.0;
```

Ve výše uvedených řádcích jsme pomocníka požádali o deklarování 3 **externích** proměnných (které může uživatel nastavit z okna properties). Tyto tři proměnné jsou typu double. Inicializovali jsme je jako výchozí hodnoty (uživatel může změnit tyto hodnoty v okně properties, doporučuje se však jejich ponechání jako výchozích (defaults)).

Nyní se na chvíli pozastavím, abych vám něco řekl o těchto proměnných.

Stop loss:

Je limitní bod, kterým nastavujete příkaz pro zastavení. Když je dosažena jeho hodnota, pozice bude uzavřena. To je užitečné pro minimalizaci vašich ztrát, když trh mluví proti vám. Body Stop loss jsou vždy nastaveny pod aktuální vyvolávací cenou při nákupu a nad aktuální nabízenou cenou při prodeji.

Trailing Stop

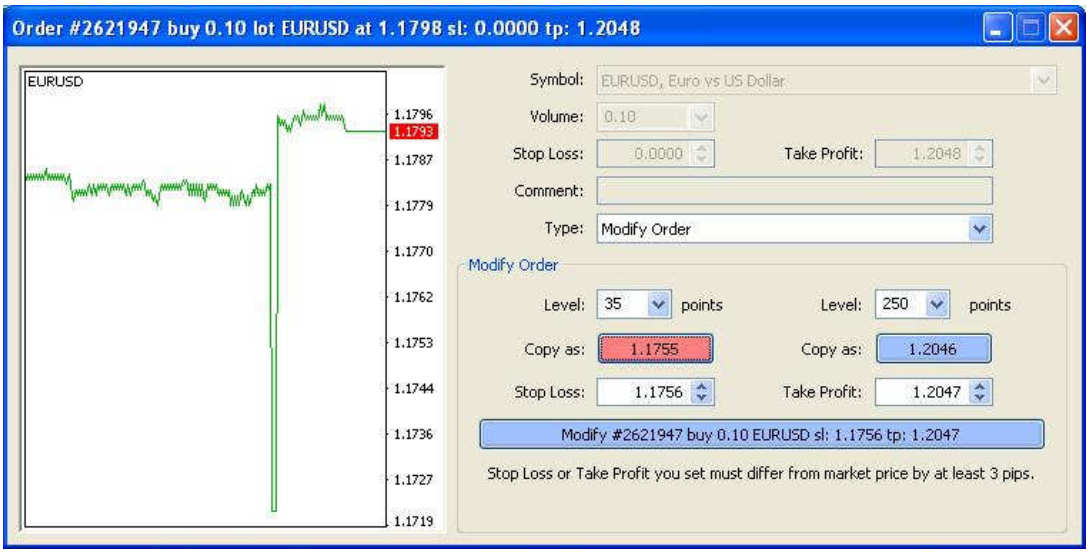
Je druhem příkazu stop loss, který je nastaven na určité procento pod (v případě long position) nebo nad (v případě short position) cenou trhu. Cena je nastavena podle její fluktuace. O tomto velmi důležitém konceptu budeme hovořit dále v této lekci.

Take profit:

Je podobný, jako příkaz stop loss, ve smyslu příkazu stanovení limitního bodu, kdy po jeho dosažení dojde k uzavření pozice.

Vyskytují se zde však dva rozdíly:

. • Nevyskytuje se zde “trailing” point.
. • Bod pro opuštění musí být nastaven nad aktuální cenou trhu, namísto pod ní.



Obr. 1 – Nastavení bodů Stop loss a Take profit

```
int Crossed (double line1 , double line2)

   {
      static int last_direction = 0;
      static int current_direction = 0;


      if(line1>line2)current_direction = 1; //nahoru
      if(line1<line2)current_direction = 2; //dolů



      if(current_direction != last_direction) //změněno

      {
            last_direction = current_direction;
            return (last_direction);

      }

      else
   {
         return (0);
         }


  }
```

Jak jsem uvedl již dříve, podstatou funkce expert advisor je monitoring křížení linií short EMA a the long EMA. A udávání směru překřížení (která linie je nad a která pod), čímž bude určen typ příkazu (buy, sell, buy-close a sell-close).

Za tímto účelem jsme vytvořili funkci *Crossed*.

Funkce *Crossed* přebírá dvě hodnoty double a vrací hodnotu integer.
První parametr je hodnota první linie, kterou si přejeme monitorovat (short EMA v našem případě) a druhý parametr je hodnota druhé linie (long EMA).

Funkce bude monitorovat obě linie pokaždé, když ji **vyvoláme** uložením směrů obou linií ve statických proměnných k zapamatování jejich stavu mezi opakovaným vyvoláním.

- Vrátí hodnotu **0,** pokud nedošlo k žádné změně v naposledy uloženém směru.
- Vrátí hodnotu **1,** pokud došlo ke změně směru (linie se vzájemně protínají) a první linie je nad druhou.
- Vrací hodnotu **2,** pokud byl směr změněn (linie se vzájemně protínají) a první linie je pod druhou.

Poznámka: *tuto funkci můžete použít v dalším expert advisoru pro monitorování jakýchkoliv dvou linií a stanovení směru překřížení.*

Podívejme se, jak jsme jej zapsali

```
int Crossed (double line1 , double line2)
```

Tento řádek je deklarování funkce, což znamená, že si přejeme vytvořit funkci *Crossed,* která přebírá dva **parametry** datového typu double a navrací integer.

Když tuto funkci **vyvoláte,** musíte jí poskytnout dva parametry typu double a vrátí vám hodnotu celého čísla.

Funkci musíte deklarovat před jejím používáním (voláním). Na umístění funkce nezáleží, já jsem ji umístil nad funkcí **start(),** vy ji však můžete umístit kdekoliv jinde.

```
static int last_direction = 0;
static int current_direction = 0;
```

Zde jsme deklarovali dvě statické hodnoty integer k udržení posledního a aktuálního směru obou linií.
Tyto proměnné použijeme (jedná se o statické proměnné, což znamená, že své hodnoty uloží mezi opakovaným voláním) pro kontrolu, zda došlo ke změně směru linií nebo nikoliv.
My jsme ji inicializovali na **0,** protože po nich nechceme, aby pracovaly při prvním vyvolání funkce (kdyby pracovaly při prvním spuštění expert advisoru, otevřely by příkaz ihned po spuštění klientského

terminálu).

```
if(current_direction != last_direction) //changed
```

V tomto řádku srovnáváme dvě statické proměnné pro kontrolu změn mezi posledním voláním naší funkce a aktuální funkcí. Pokud se *last_direction* nerovná *current_direction,* znamená to, že se ve směru nevyskytla žádná změna.

```
last_direction = current_direction;
return (last_direction);
```

V tomto případě (*last_direction* není rovno *current_direction)* musím resetovat *last_direction* pomocí přiřazení hodnoty *current_direction*. A my vrátíme hodnotu posledního směru *last_direction*. Tato hodnota bude **1,** pokud je první linie nad druhou a **2,** pokud je pod druhou linií.

```
else
  {
     return (0);
  }
```

Jinak (*last_direction* je rovno *current_direction)* nedojde k žádné změně směru linií a nám se vrátí **0**.

Náš program vyvolá tuto funkci při spuštění těla funkce **start()** a použije vrácenou hodnotu k určení odpovídající akce.

V další části lekce se dozvíme, jak jsme vyvolali funkci a dozvíme se mnohé o velmi důležitých obchodních funkcích (trading).
Pro dnešek vám přeji hodně štěstí.

Velmi uvítám jakékoliv dotazy a návrhy.

S pozdravem

Coders' Guru
01-12-2005