MQL4 COURSE

By Coders' guru
www.forex-tsd.com

-12

Váš první indikátor – Část 3

Vítejte ve třetí části lekce **"Váš první indikátor"**.
V předchozí lekci jsme studovali kódy našeho prvního indikátoru řádek po řádku a dostali jsme se k funkci **deinit()**.

Doufám, že jste v minulých lekcích získali jasný obraz o tom, co jsme prováděli.

Dnes budeme studovat funkci **start()** a její obsah. A –nakonec – zkompilujeme a spustíme náš první indikátor.

Jste připraveni? Pojďme tedy rozřezat kód řádek po řádku:

Náš kód:

```
//+------------------------------------------------------------------
+
//| My_First_Indicator.mq4 |
//| Codersguru |
//| http://www.forex-tsd.com |
//+------------------------------------------------------------------
+

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


#property indicator_separate_window
#property indicator_buffers 1
#property indicator_color1 Red


//----vyrovnávací paměti

double ExtMapBuffer1[];
```

```
//+------------------------------------------------------------------+
//| Custom indicator - inicializační funkce                          |
//+------------------------------------------------------------------+
int init()

  {
//----indikátory

   SetIndexStyle(0,DRAW_LINE);

   SetIndexBuffer(0,ExtMapBuffer1);

   string short_name = "Váš první indikátor běží!";

   IndicatorShortName(short_name);
//----

   return(1);

  }
//+------------------------------------------------------------------+
//| Custor indicator - deinicializační funkce                        |
//+------------------------------------------------------------------+
int deinit()

  {
//----


//----

   return(0);

  }
//+------------------------------------------------------------------+
//| Custom indicator - opakovací funkce |
//+------------------------------------------------------------------+
int start()

  {

   int counted_bars=IndicatorCounted();


//----kontrola možných chyb

   if (counted_bars<0) return(-1);

//----naposledy počítaná svíce bude přepočtena

   if (counted_bars>0) counted_bars--;
```

```
   int pos=Bars-counted_bars;

   double dHigh , dLow , dResult;

   Comment("Hi! I'm here on the main chart window!");

//----hlavní kalkulační smyčka

   while(pos>=0)

     {

         dHigh = High[pos];

         dLow = Low[pos];

         dResult = dHigh -dLow;

         ExtMapBuffer1[pos]= dResult ;

         pos--;

     }

//----

   return(0);

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

```
 int start()
  {...
  return(0);
  }
```

Jak jsem vám již říkal dříve, 90% našeho programovacího života strávíme uvnitř závorek funkce **start()**. To proto, že se jedná o nejdůležitější zvláštní funkci programu MQL4.

Na rozdíl od funkcí **init()** a **deinit**, funkce **start()** nebude vyvolána pouze jednou (klientským terminálem), ale při každé nové cenové nabídce, kterou klientský terminál MetaTrader obdrží. Funkce **start()** vrací hodnotu celého čísla, jak je tomu u zvláštních funkcí MQL4, kde **0** znamená, že se nevyskytla žádná chyba, v každém jiném případě se chyba vyskytla.

```
int counted_bars=IndicatorCounted();
```

Zde jsme definovali proměnné **counted_bars** jako typy integer a přiřadili jsme k nim vratnou hodnotu funkce **IndicatorCounted()**.

int IndicatorCounted()

Tato funkce vrací hodnotu typu integer a obsahuje součet **svící**, který propočítal náš indikátor.

Při prvním spuštění vašeho indikátoru bude hodnota **0,** protože indikátor zatím žádné svíce v grafu –**1** nepropočítával. (Viz. níže uvedené funkce **Bars**).

```
if (counted_bars<0) return(-1);
```

Součet **counted_bars** jsme dostali v předchozím řádku kódu funkce **IndicatorCounted()**. V případě žádného výskytu chyb musí být číslo 0 nebo vyšší. Pokud je hodnota nižší než 0, znamená to, že se vyskytla chyba a funkce **start()** je zrušena příkazem return.

```
if (counted_bars>0) counted_bars--;
```

Zde kontrolujeme, zda je hodnota **counted_bars** vyšší než 0. Pokud je tomu tak, snížíme tento počet odečtením hodnoty 1. To proto, že poslední lištu budeme chtít přepočítat.

Použijeme operátor pro odpočet (viz. Lesson 4 - Operace & Výrazy) pro snížení hodnoty **counted_bars** o **1**.

Poznámka: Výraz counted_bars můžeme zapsat takto:

```
counted_bars = counted_bars-1;
```

```
int pos=Bars-counted_bars;
```

Zde deklarujeme proměnnou **pos** k podržení počtu opakování, po které bude naše kalkulační smyčka pracovat. (viz. smyčka „while“ později v této lekci). To probíhá odečtem **counted_bars** od celkového počtu svící v grafu, čímž dosáhneme celkové hodnoty počtu svící pomocí funkce **Bars()**. Nyní je pravý čas prodiskutovat funkci **Bars()** a jejího bratra.

Předdefinované MQL4 proměnné:

Ask, Bid, Bars, Close, Open, High, Low, Time a Volume jsou funkce nazývané v MQL4 „předdefinované hodnoty“. A já vám ukážu, proč jsou vlastně funkcemi.

Proměnná znamená místo v paměti a datový typ dle vaší specifikace.
Funkce znamená něco udělat a vrátit nějakou hodnotu, např. **Bars** shromažďuje a vrací počet svíc v grafu.
Takže, jedná se o proměnnou?

Další příklad, dokazující, že se nejedná o proměnné:
Když napíšete a zkompilujete tento řádek kódu:

```
Bars=1;
```

Vyobrazí se vám tato chyba: ***'Bars' -unexpected token*** To proto, že se nejedná o proměnné, nemůžete k nim tedy přiřadit hodnotu.

Dalším důkazem je, že další řádek kódu je platným řádkem a nebude generovat chybu v kompilaci:

```
Alert(Bars(1));
```

Nemůžete předat parametry proměnné, předány mohou být pouze funkcím. Omlouvám se za protahování, věnujme se tedy jednotlivým funkcím.

int Bars

Tato funkce, která vrací hodnotu typu celého čísla, uchovává celkový počet svící aktuálního grafu.

double Ask

Tato funkce (používána funkcemi Expert Advisors) vrací hodnotu typu double a uchovává cenu kupujícího v měnovém páru.

double Bid

Tato funkce (používaná funkcemi Expert Advisor) vrací hodnotu typu double a uchovává cenu kupujícího v měnovém páru. Poznámka: Např., USD/JPY = 133.27/133.32 levá část je nazývána „**bid** price“ (což je cena, za kterou obchodník prodává), druhá (pravá část) se nazývá **ask** (cena, při které obchodník kupuje měnu).

double Open[]

Tato funkce vrací hodnotu typu double, uchovává otevírací cenu referenční svíce, kde otevírací cena je cenou na začátku obchodní periody (rok, měsíc, den, týden, hodina, atd.)
Např.: Open[0] vrací otevírací cenu aktuální svíce.

double Close[]

Tato funkce vrací hodnotu typu double, uchovává uzavírací cenu referenční svíce, kde uzavírací cena je cenou na konci obchodní periody.
Např.: Close[0] vrací uzavírací cenu aktuální svíce.

double High[]

Tato funkce vrací hodnotu typu double nejvyšší ceny referenční svíce, která je nejvyšší z cen hlídaných během obchodní periody.
Např.: High [0] vrací nejvyšší cenu aktuální svíce.

double Low[]

Tato funkce vrací hodnotu typu double nejnižší ceny referenční svíce, která je nejnižší z cen hlídaných během obchodní periody. Např.: Low [0] vrací nejnižší cenu aktuální svíce

double Volume[]

Tato funkce vrací hodnotu typu double, uchovává celkový průměr měn obchodovaných v rámci určité časové periody, většinou jednoho dne.
Např.: Volume [0] vrací průměr aktuální svíce.

int Digits

Tato funkce vrací hodnotu celého čísla, uchovává počet číslic po decimálním bodu (většinou 4).

double Point

Tato funkce vrací hodnotu typu double, bodovou hodnotu aktuální svíce (obvykle 0.0001).

datetime Time[]

Tato funkce vrací hodnotu typu datetime, uchovává otevírací dobu referenční svíce. Např.: Time [0] vrátí hodnotu otevírací doby aktuální svíce.

```
double dHigh,dLow,dResult;
```

Deklarovali jsme tři typy proměnných double, které použijeme později. Všimněte si způsobu, jakým jsme

deklarovali tři z nich na stejném řádku, oddělením čárkami.

```
Comment("Hi! I'm here on the main chart windows!");
```

Tento řádek kódů se používá ke **komentování** funkce prostřednictvím vyobrazení textu "Hi! I'm here on the mainchart windows!" v levé straně hlavního grafu (obr. 1).
Jsou zde dvě obdobné funkce:

void Comment(...)

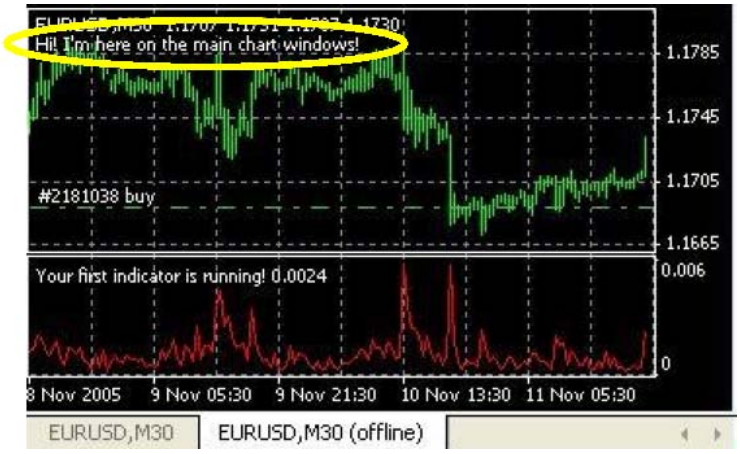
Tato funkce přebírá hodnoty, které jí byly poskytnuty (mohou být jakéhokoliv typu) a vyobrazí je v levém horním rohu grafu (obr. 1).

void Print (...)

Tato funkce přebírá hodnoty, které jí byly poskytnuty (mohou být jakéhokoliv typu) a vyobrazí je v protokolu Expertu. (obr. 2).

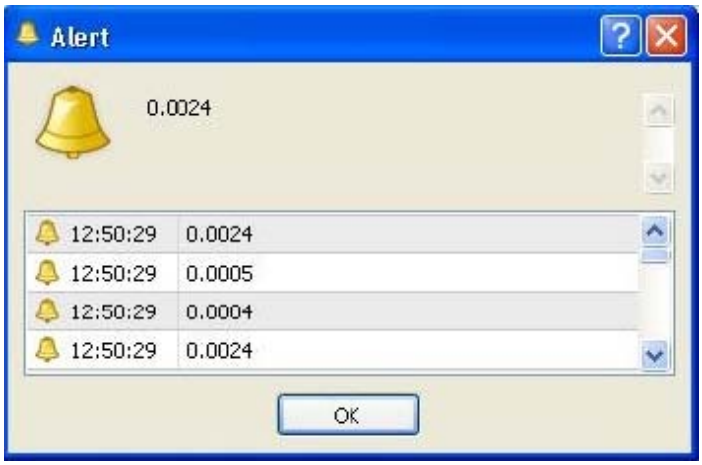
void Alert(...)

Tato funkce přebírá hodnoty, které jí byly poskytnuty (mohou být jakéhokoliv typu) a vyobrazí je v dialogovém boxu (obr. 3)



Obr. 1 – Komentář

Obr. 2-Protokol Expertu



Obr. 3 -Upozornění

```
while(pos>=0)

     {
         dHigh = High[pos];
         dLow = Low[pos];
         dResult = dHigh-dLow;
         ExtMapBuffer1[pos]= dResult ;
         pos--;

     }
```

Nyní je čas ke vstupu do smyčky pro výpočet bodů indikátoru pro vykreslení. Jakákoliv hodnota, kterou jsme přiřadili do pole ExtMapBuffer1[] bude vykreslena v grafu. (protože jsme toto pole přiřadili pro vykreslení použitím funkce vyrovnávací paměti **SetIndexBuffer**).

Před vstupem do smyčky máme k dispozici počet časů, smyčka bude pracovat odpočtem **counted_bars** z celkového počtu svící v grafu. Počet časů smyčky bude pracovat vyvoláním proměnné ***Loop variable,*** v našem případě jde o proměnnou **pos**.

Smyčku variable používáme jako aktuální svíci kalkulace, např. **High[pos]** vrací vyšší cenu svíce pos.

V těle smyčky jsme přiřadili proměnné **dHigh,** což je hodnota nejvyšší ceny aktuální dostupné smyčky. A k proměnné **dLow** jsme přiřadili hodnotu nejnižší ceny aktuální dostupné proměnné smyčky. Výsledek odpočtu **dLow** od **dHigh** bude přiřazena proměnné **dResult**.

Poté použijeme pro vykreslení řádku indikátoru proměnnou **dResult** jejím přiřazením k poli vyrovnávací

paměti **ExtMapBuffer1[]**.

Poslední řádek výrazu je sestupný výraz, který sníží proměnnou smyčky pos o 1 pokaždé, když smyčka běží. A když tato proměnná dosáhne hodnoty **-1** smyčka bude ukončena.

Nakonec můžeme náš indikátor zkompilovat. Stiskněte klávesu **F5** nebo volbu **Compile** z file menu. Tím vygenerujeme prováděcí soubor "My_First_indicator.ex4", který můžete nahrát do vašeho terminálového klienta.

Pro nahrání vašeho indikátoru klikněte na klávesu **F4,** čímž klienta terminálu vyvoláte. Poté v okně **Navigator** vyhledejte **My_First_indicator** a přiřaďte jej ke grafu (obr. 4).

Poznámka: Indikátor toho mnoho neudělá, Odpočty nejvyšších a nejnižších cen nám však poskytnou obraz o vrtkavosti trhu.

Obr. 4 – Můj první indikátor

Doufám, že vás váš první indikátor bavil. Velmi uvítám jakékoliv dotazy a připomínky.

S pozdravem

Coders' Guru
13-11-2005