# MQL4 COURSE

By Coders' guru

www.forex-tsd.com

(Appendix 2)

**Trading Functions**

---

V tomto dodatku je obsažen popis **25** obchodních funkcí jazyka MQL4. Rozhodl jsem se napsat tento dodatek před uvedením třetí části návodu "**Your First Expert Advisor**" (lekce 15) protože je to nezbytné pro pochopení zbývající části programového kódu.

---

## OrderSend:

### ***Syntaxe:***

```
int OrderSend(string symbol, int cmd, double volume, double price, int slippage, double
stoploss, double takeprofit, string comment=NULL, int magic=0, datetime expiration=0,
color arrow_color=CLR_NONE)
```

### ***Popis:***

Funkce *OrderSend* se používá k zadání příkazů buy/sell (nákup/prodej) nebo pending order (podmíněný vstup do pozice) .
Vrací číslo štítku příkazu, v případě chyby vrací hodnotu **-1**. K získání podrobností o chybě se používá funkce *GetLastError*.

**Poznámka:** Číslo štítku, vrácené funkcí *OrderSend* je jedinečné a používá se jako reference (návěští) zadaného příkazu. (například můžeme použít funkci *OrderClose* k uzavření tohoto příkazu).

**Poznámka:** Funkce *GetLastError* vrací předdefinované číslo poslední chyby. (Například pokud voláme funkci *GetLastError* po provedení *OrderSend,* dostaneme číslo chyby, která nastala při provádění funkce *OrderSend*).

Po provedení funkce *GetLastError* je číslo poslední chyby nastaveno na hodnotu 0.

Úplný seznam chyb jazyka MQL4 je uložen v souboru *stderror.mqh*. Popis konkrétní chyby můžeme získat pomocí funkce *ErrorDescription,* která je definována v souboru *stdlib.mqh*.

***Parametry:***
Tato funkce má **11** Parametrů:

**String symbol**:
Symbol měnového páru, který obchodujeme (Příklad: EURUSD and USDJPY).

**Poznámka:** Funkce *Symbol()* se používá k načtení symbolu aktuálního měnového páru a funkce *OrderSymbol* k načtení symbolu měnového páru zadaného příkazu.

**int cmd**:
Celé číslo, určuje typ operace, kterou chceme provést; může nabývat následujících hodnot:

| Konstanta | Hodnota | Popis |
|---|---|---|
| OP_BUY | 0 | Buying position. |
| OP_SELL | 1 | Selling position. |
| OP_BUYLIMIT | 2 | Buy limit pending position. |
| OP_SELLLIMIT | 3 | Sell limit pending position. |
| OP_BUYSTOP | 4 | Buy stop pending position. |
| OP_SELLSTOP | 5 | Sell stop pending position. |

**Poznámka:** Můžeme použít celočíselné vyjádření hodnoty nebo název konstanty.

Příklad:
*OrderSend(Symbol(),0,…)* je ekvivalentní *OrderSend(Symbol(),OP_BUY,…)* .
Obvykle se z důvodu přehlednosti programového kódu používají názvy konstant.

**double volume**:
Počet lotů, které chceme obchodovat.

**double price**:
Cena, na které chceme zadat příkaz. K získání aktuální ceny se používají funkce *Bid* a *Ask*.

**int slippage**:
Hodnota slippage kterou v příkazu povolíme.

**Poznámka:** *slippage* je rozdíl mezi předpokládanou cenou transakce a skutečnou cenou.

**double stoploss**:
Cena, na které chceme příkaz uzavřít příkaz v případě, že jde do ztráty.

**double takeprofit**:
Cena, na které chceme příkaz uzavřít příkaz v případě, že jde do zisku.

**string comment**:
Komentář, který chceme přiřadit zadanému příkazu (*Obrázek 1*).
Implicitní hodnota je *NULL* , to znamená bez komentáře.

**Poznámka:** Implicitní hodnota parametru znamená, že nezadáme žádnou hodnotu a MQL4 použije předdefinivanou hodnotu parametru.
Například můžeme zapsat funkci *OrderSend* s parametrem takto:
*OrderSend(Symbol(),OP_BUY,Lots,Ask,3,Ask-25*Point,Ask+25*Point,**"My order comment"**,12345,0,Green);*

nebo bez parametru takto:
*OrderSend(Symbol(),OP_BUY,Lots,Ask,3,Ask-25*Point,Ask+25*Point,12345,0,Green);*

**int magic**:
magické číslo, přiřazené zadanému příkazu.

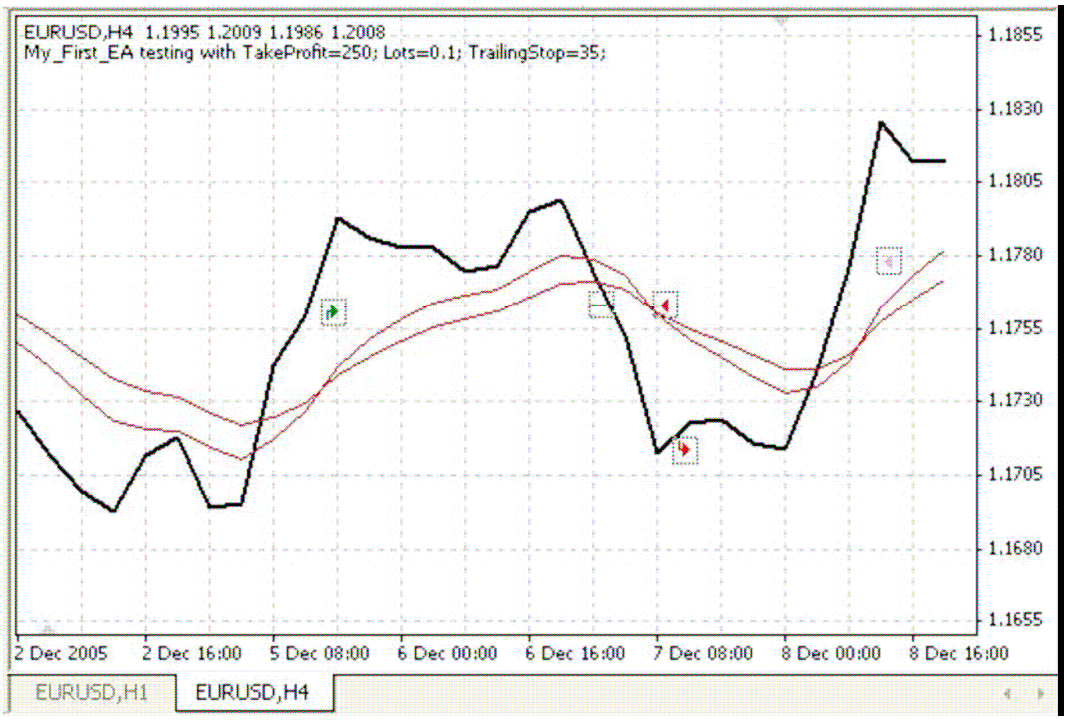
**Poznámka:** Magické číslo je číslo, které přiřadíme zadanému příkazu jako referenci (návěští), která nám umožní rozlišovat mezi různými příkazy. Například mezi příkazy, které zadal náš expert advisor a příkazy, které jsme zadali manuálně.


*Obrázek 1 – parametr* **comment**

**datetime expiration**:
Datum a čas, kdy vyprší platnost příkazu pending order (podmíněný vstup do pozice). Implicitní nastavení je **0,** to znamená bez doby expirace.

**Poznámka:** čas, zobrazený zde, je čas serveru, nikoli náš místní čas; k načtení aktuálního času serveru se používá funkce *CurTime* a k načtení místního času funkce *LocalTime*.

**color arrow_color**:
Barva šipky (*Obrázek 2*), implicitní nastavení je *CLR_NONE* , to znamená bez šipky.




*Obrázek 2 – parametr* **arrow_color**

## ***Příklad:***

```
int ticket;
if(iRSI(NULL,0,14,PRICE_CLOSE,0)<25)
  {
   ticket=OrderSend(Symbol(),OP_BUY,1,Ask,3,Ask-25*Point,Ask+25*Point,"My
   order #2",16384,0,Green);
   if(ticket<0)
     {
     Print("OrderSend failed with error #",GetLastError());
     return(0);
     }
  }
```

# OrderModify:

## *Syntaxe:*

**bool OrderModify(int ticket, double price, double stoploss, double takeprofit, datetime expiration, color arrow_color=**CLR_NONE)

## *Popis:*

Funkce *OrderModify* se používá k modifikaci vlastností zadaného příkazu zadáním nových hodnot. Vrací hodnotu true (pravda) v případě úspěšné modifikace a hodnotu false (nepravda) v případě chyby. K získání podrobností o chybě se používá funkce *GetLastError*.

## *Parametry:*

Tato funkce má **6** parametrů:

**int ticket**:
Číslo štítku příkazu, který chceme modifikovat.

**Poznámka**: Toto číslo jsme přiřadili příkazu pomocí funkce *OrderSend*. Pro získaní čísla štítku aktuálního příkazu můžeme použít funkci *OrderTicket*.

**double price**:
Cena, na které chceme zadat příkaz.

**Poznámka:** K získání open price (otvírací ceny) zadaného příkazu můžeme použít funkci *OrderOpenPrice.*

**double stoploss**:
Cena, na které chceme uzavřít příkaz v případě, že jde do ztráty.

**double takeprofit**:
Cena, na které chceme uzavřít příkaz v případě, že jde do zisku.

**Poznámka**: Obvykle používáme funkci *OrderModify* ke změnám hodnot parametrů *trailing stoploss* a *trailing takeprofit*. (posuvný stoploss a takeprofit)

**datetime expiration**:
Čas, kdy vyprší doba platnosti příkazu pending order (podmíněný vstup do pozice). Pokud nechceme nastavit dobu platnosti příkazu, zadáme hodnotu **0.**

**color arrow_color**:
Barva šipky, implicitní nastavení je *CLR_NONE* , to znamená bez šipky.

***Příklad:***

```
if(TrailingStop>0)
  {
  SelectOrder(12345,SELECT_BY_TICKET);
  if(Bid-OrderOpenPrice()>Point*TrailingStop)
    {
    if(OrderStopLoss()<Bid-Point*TrailingStop)
      {
      OrderModify(OrderTicket(),Ask-10*Point,Ask-35*Point,OrderTakeProfit(),0,Blue);
      return(0);
      }
    }
  }
```

---

## OrderClose:

***Syntaxe:***

**bool OrderClose(int ticket, double lots, double price, int slippage, color Color**=CLR_NONE)

***Popis:***

Funkce *OrderClose* se používá k uzavření příkazu (dle čísla jeho štítku).
Vrací hodnotu true (pravda) pokud je příkaz úspěšně uzavřen, v případě chyby vrací hodnotu false (nepravda). K získání podrobností o chybě se používá funkce *GetLastError*.

***Parametry:***

Tato funkce má **5** parametrů:

**int ticket**:
Číslo štítku příkazu, který chceme uzavřít.

**double lots:**
Počet lotů, se kterými příkaz pracuje.

**Poznámka:** K načtení počtu lotů aktuálního příkazu se používá funkce *OrderLots*.

**double price:**
Cena, na které chceme příkaz uzavřít. K načtení aktuální ceny se používají funkce *Bid* and *Ask*.

**int slippage:**
Hodnota *slippage* zadaného příkazu.

**color Color:**
Barva šipky, implicitní hodnota je ***CLR_NONE*** , to znamená bez šipky.

## ***Příklad:***

```
if(iRSI(NULL,0,14,PRICE_CLOSE,0)>75)
  {
  OrderClose(order_id,1,Ask,3,Red);
  return(0);
  }
```

---

# OrderSelect:

## ***Syntaxe:***

```
bool OrderSelect(int index, int select, int pool=MODE_TRADES)
```

## ***Popis:***
Funkce *OrderSelect* se používá k výběru zadaného příkazu dle čísla štítku nebo dle indexu. Vrací hodnotu true (pravda) v případě úspěšného výběru, hodnotu false (nepravda) v případě chyby. K získání podrobností o chybě se používá funkce *GetLastError*.

**Poznámka:** Funkci *OrderSelect* je nutné použít vždy před voláním funkcí bez parametrů: *OrderMagicNumber, OrderClosePrice, OrderCloseTime, OrderOpenPrice, OrderOpenTime, OrderComment, OrderCommission, OrderExpiration, OrderLots, OrderPrint, OrderProfit, OrderStopLoss, OrderSwap, OrderSymbol, OrderTakeProfit, OrderTicket a OrderType*

## ***Parametry:***
Tato funkce má **3** parametry:

**int index:**
Index nebo číslo štítku příkazu, který chceme vybrat. Závisí na druhém parametru (selecting type).

**int select**:
Typ výběrové operace (dle indexu nebo dle čísla štítku).
Může nabývat dvou hodnot:
*SELECT_BY_POS*: výběr dle pozice (indexu) zadaného příkazu.
*SELECT_BY_TICKET* : výběr dle čísla štítku zadaného příkazu.

**int pool:**
Použijeme-li *SELECT_BY_POS* (výběr dle pozice), musíme určit databázi, ze které budeme potřebný údaj vybírat:
*MODE_TRADES*: výběr z aktuálně zadaných příkazů . Tato hodnota je nastavena jako implicitní.
*MODE_HISTORY*: výběr z historie účtu (uzavřené a zrušené příkazy).

### ***Příklad:***

```
if(OrderSelect(12470, SELECT_BY_TICKET)==true)
   {
   Print("order #12470 open price is ", OrderOpenPrice());
   Print("order #12470 close price is ", OrderClosePrice());
   }
else
   Print("OrderSelect failed error code is",GetLastError());
```

---

## OrderDelete:

### ***Syntaxe:***

```
bool OrderDelete(int ticket)
```

### ***Popis:***
Funkce *OrderDelete* se používá k vymazání příkazu pending order (podmíněný vstup do pozice). Vrací hodnotu true (pravda) v případě úspěšného vymazání, v případě chyby vrací hodnotu false (nepravda). K získání podrobností o chybě se používá funkce *GetLastError*.

### ***Parametry:***
Tato funkce má **1** parametr:

**int ticket:**
Číslo štítku příkazu, který chceme vymazat.

### ***Příklad:***

```
if(Ask>var1)
   {
   OrderDelete(order_ticket);
   return(0);
   }
```

---

## OrderCloseBy:

### ***Syntaxe:***

```
bool OrderCloseBy(int ticket, int opposite, color Color=CLR_NONE)
```

### ***Popis:***
Funkce *OrderCloseBy* se používá k uzavření příkazu zadáním příkazu v opačném směru.
Vrací hodnotu true (pravda) v případě úspěšného uzavření, v případě chyby vrací hodnotu false (nepravda).
K získání podrobností o chybě se používá funkce *GetLastError*.

### ***Parametry:***
Tato funkce má **3** parametry:

**int ticket:**
Č9slo štítku příkazu, který chceme uzavřít.

**int opposite:**
Číslo štítku příkazu, který chceme uzavřít v opačném směru.

**color Color:**
Barva šipky, implicitní hodnota je CLR_NONE , to znamená bez šipky

### ***Příklad:***
```
if(iRSI(NULL,0,14,PRICE_CLOSE,0)>75)
  {
  OrderCloseBy(order_id,opposite_id);
  return(0);
  }
```

---

## OrderType:

### ***Syntaxe:***

```
int OrderType( )
```

### ***Popis:***
Funkce *OrderType* vrací typ vybraného příkazu, může nabývat hodnot:

*OP_BUY, OP_SELL, OP_BUYLIMIT, OP_BUYSTOP, OP_SELLLIMIT* nebo *OP_SELLSTOP* (viz funkce *OrderSend*)
Před voláním funkce *OrderType* musí být příkaz vybrán pomocí funkce *OrderSelect.*

### ***Parametry:***
Tato funkce nemá parametry a vrací celočíselnou hodnotu typu **int** (typ vybraného příkazu).

### ***Příklad:***

```
int order_type;
  if(OrderSelect(12, SELECT_BY_POS)==true)
    {
    order_type=OrderType();
    // ...
    }
  else
Print("OrderSelect returned error - ",GetLastError());
```

---

## HistoryTotal:

### ***Syntaxe:***

```
int HistoryTotal( )
```

### ***Popis:***
Funkce *HistoryTotal* prohledává načtenou historii účtu a vrací počet uzavřených příkazů.

**Poznámka:** Obvykle se tato funkce používá společně s funkcí ***OrderSelect*** k získaní informací o určitém příkazu z historie účtu.

### ***Parametry:***
Tato funkce nemá parametry a vrací celočíselnou hodnotu typu **int** (počet uzavřených příkazů v historii účtu).
K získání podrobností o chybě se používá funkce *GetLastError*.

***Příklad:***

```
// retrieving info from trade history
int i,hstTotal=HistoryTotal();
for(i=0;i<hstTotal;i++)
  {
  //---- check selection result
  if(OrderSelect(i,SELECT_BY_POS,MODE_HISTORY)==false)
    {
    Print("Access to history failed with error (",GetLastError(),")");
    break;
    }
  // some work with order
  }
```

---

## OrderClosePrice:

***Syntaxe:***

```
double OrderClosePrice( )
```

***Popis:***
Funkce *OrderClosePrice* vrací close price (uzavírací cenu) vybraného příkazu.
Před voláním funkce *OrderClosePrice* musí být příkaz vybrán pomocí funkce *OrderSelect.*

***Parametry:***
Tato funkce nemá parametry a vrací hodnotu typu **double** (close price vybraného příkazu).

***Příklad:***

```
if(OrderSelect(ticket,SELECT_BY_POS)==true)
    Print("Close price for the order ",ticket," = ",OrderClosePrice());
  else
    Print("OrderSelect failed error code is",GetLastError());
```

---

## OrderCloseTime:

### *Syntaxe:*

```
datetime OrderCloseTime( )
```

### *Popis:*
Funkce *OrderCloseTime* vrací close time (uzavírací čas) vybraného příkazu.
Pokud vrátí hodnotu **0** znamená to, že příkaz nebyl dosud uzavřen, nebo byl uzavřen a později znovu vyvolán z historie.
Před voláním funkce *OrderCloseTime* musí být příkaz vybrán pomocí funkce *OrderSelect.*

### *Parametry:*
Tato funkce nemá parametry a vrací hodnotu typu **datetime** (uzavírací čas vybraného příkazu).

### *Příklad:*

```
if(OrderSelect(10,SELECT_BY_POS,MODE_HISTORY)==true)
    {
    datetime ctm=OrderOpenTime();
    if(ctm>0) Print("Open time for the order 10 ", ctm);
    ctm=OrderCloseTime();
    if(ctm>0) Print("Close time for the order 10 ", ctm);
    }
  else
    Print("OrderSelect failed error code is",GetLastError());
```

---

## OrderComment:

### *Syntaxe:*

```
string OrderComment( )
```

### *Popis:*
Funkce *OrderComment* vrací komentář k vybranému příkazu.

**Poznámka:** Tento komentář byl přiřazen při zadání příkazu pomocí funkce *OrderSend* nebo jej přiřadil server. Někdy server přidává svůj komentář na konec námi zadaného komentáře.
Před voláním funkce *OrderComment* musí být příkaz vybrán pomocí funkce *OrderSelect.*

***Parametry:***
Tato funkce nemá parametry a vrací řetězec typu **string** (komentář k vybranému příkazu).

***Příklad:***

```
string comment;
  if(OrderSelect(10,SELECT_BY_TICKET)==false)
    {
    Print("OrderSelect failed error code is",GetLastError());
    return(0);
    }
  comment = OrderComment();
  // ...
```

---

## OrderCommission:

***Syntaxe:***

```
double OrderCommission( )
```

***Popis:***
Funkce *OrderCommission* vrací výši commission (poplatku) u vybraného příkazu.
Před voláním funkce *OrderCommission* musí být příkaz vybrán pomocí funkce *OrderSelect.*

***Parametry:***
Tato funkce nemá parametry a vrací hodnotu typu double (výše poplatku u vybraného příkazu).

***Příklad:***

```
if(OrderSelect(10,SELECT_BY_POS)==true)
   Print("Commission for the order 10 ",OrderCommission());
 else
   Print("OrderSelect failed error code is",GetLastError());
```

---

## OrderExpiration:

***Syntaxe:***

```
datetime OrderExpiration( )
```

***Popis:***
Funkce *OrderExpiration* vrací datum a čas, kdy vyprší platnost příkazu pending order (podmíněný vstup do pozice); datum a čas jsme zadali pomocí funkce *OrderSend.*
Před voláním funkce *OrderExpiration* musí být příkaz vybrán pomocí funkce *OrderSelect.*

***Parametry:***
Tato funkce nemá parametry a vrací hodnotu typu **datetime** (datum a čas, kdy vyprší platnost příkazu pending order).

***Příklad:***

```
if(OrderSelect(10, SELECT_BY_TICKET)==true)
    Print("Order expiration for the order #10 is ",OrderExpiration());
  else
    Print("OrderSelect failed error code is",GetLastError());
```

---

## OrderLots:

***Syntaxe:***

```
double OrderLots( )
```

***Popis:***
Funkce *OrderLots* vrací počet lotů vybraného příkazu; tuto hodnotu jsme nastavili pomocí funkce *OrderSend* (parametr *volume*).
Před voláním funkce *OrderLots* musí být příkaz vybrán pomocí funkce *OrderSelect.*

***Parametry:***
Tato funkce nemá parametry a vrací hodnotu typu **double** (počet lotů vybraného příkazu).

***Příklad:***

```
if(OrderSelect(10,SELECT_BY_POS)==true)
    Print("lots for the order 10 ",OrderLots());
  else
    Print("OrderSelect failed error code is",GetLastError());
```

---

## OrderMagicNumber:

### *Syntaxe:*

```
int OrderMagicNumber( )
```

### *Popis:*
Funkce *OrderMagicNumber* vrací magické číslo vybraného příkazu, zadané pomocí funkce *OrderSend*.
Před voláním funkce *OrderMagicNumber* musí být příkaz vybrán pomocí funkce *OrderSelect.*

### *Parametry:*
Tato funkce nemá parametry a vrací hodnotu typu **int** (magické číslo vybraného příkazu).

### *Příklad:*

```
if(OrderSelect(10,SELECT_BY_POS)==true)
    Print("Magic number for the order 10 ", OrderMagicNumber());
  else
    Print("OrderSelect failed error code is",GetLastError());
```

---

## OrderOpenPrice:

### *Syntaxe:*

```
double OrderOpenPrice( )
```

### *Popis:*
Funkce *OrderOpenPrice* vrací open price (otevírací cenu) vybraného příkazu.
Před voláním funkce *OrderOpenPrice* musí být příkaz vybrán pomocí funkce *OrderSelect.*

### *Parametry:*
Tato funkce nemá parametry a vrací hodnotu typu **double** (open price vybraného příkazu).

### *Příklad:*

```
if(OrderSelect(10, SELECT_BY_POS)==true)
    Print("open price for the order 10 ",OrderOpenPrice());
  else
    Print("OrderSelect failed error code is",GetLastError());
```

---

## OrderOpenTime:

### ***Syntaxe:***

```
datetime OrderOpenTime( )
```

### ***Popis:***
Funkce *OrderOpenTime* vrací datum a čas zadání vybraného příkazu .
Před voláním funkce *OrderOpenTime* musí být příkaz vybrán pomocí funkce *OrderSelect.*

### ***Parametry:***
Tato funkce nemá parametry a vrací hodnotu typu **datetime** (čas zadání vybraného příkazu).

### ***Příklad:***

```
if(OrderSelect(10, SELECT_BY_POS)==true)
    Print("open time for the order 10 ",OrderOpenTime());
  else
    Print("OrderSelect failed error code is",GetLastError());
```

---

## OrderPrint:

### ***Syntaxe:***

```
void OrderPrint( )
```

### ***Popis:***
Funkce *OrderPrint* zapíše vybraná data příkazu do expert log file (protokolu experta).
Před voláním funkce *OrderPrint* musí být příkaz vybrán pomocí funkce *OrderSelect.*

### ***Parametry:***
Tato funkce nemá parametry a nevrací žádnou hodnotu (void).

**Poznámka:** *void* znamená, že funkce nevrací žádnou hodnotu, takže není možné ji přiřadit proměnné, například takto:
**int i = OrderPrint();** //tento řádek nemá smysl, přestože překladač jazyka nebude hlásit chybu.

***Příklad:***

```
if(OrderSelect(10, SELECT_BY_TICKET)==true)
    OrderPrint();
  else
    Print("OrderSelect failed error code is",GetLastError());
```

---

## OrderProfit:

***Syntaxe:***

```
double OrderProfit( )
```

***Popis:***
Funkce *OrderProfit* vrací výši profitu vybraného příkazu.
Před voláním funkce *OrderProfit* musí být příkaz vybrán pomocí funkce *OrderSelect.*

***Parametry:***
Tato funkce nemá parametry a vrací hodnotu typu **double** (výši profitu vybraného příkazu).

***Příklad:***

```
if(OrderSelect(10, SELECT_BY_POS)==true)
    Print("Profit for the order 10 ",OrderProfit());
  else
    Print("OrderSelect failed error code is",GetLastError());
```

---

## OrderStopLoss:

***Syntaxe:***

```
double OrderStopLoss( )
```

***Popis:***
Funkce *OrderStopLoss* vrací hodnotu *stoploss* vybraného příkazu, kterou jsme nastavili pomocí funkce *OrderSend* nebo modifikovali pomocí funkce *OrderModify*.
Před voláním funkce *OrderStopLoss* musí být příkaz vybrán pomocí funkce *OrderSelect.*

***Parametry:***
Tato funkce nemá parametry a vrací hodnotu typu **double** (*stoploss* vybraného příkazu).

***Příklad:***

```
if(OrderSelect(ticket,SELECT_BY_POS)==true)
    Print("Stop loss Hodnota for the order 10 ", OrderStopLoss());
  else
    Print("OrderSelect failed error code is",GetLastError());
```

---

## OrdersTotal:

***Syntaxe:***

```
int OrdersTotal( )
```

***Popis:***
Funkce *OrdersTotal* vrací počet zadaných příkazů (tj. součet otevřených pozic a příkazů typu pending order). Jestliže vrátí hodnotu **0** znamená to, že není zadán žádný příkaz.

***Parametry:***
Tato funkce nemá parametry a vrací hodnotu typu **int** (počet zadaných příkazů).

***Příklad:***

```
int handle=FileOpen("OrdersReport.csv",FILE_WRITE|FILE_CSV,"\t");
  if(handle<0) return(0);
  // write header
    FileWrite(handle,"#","open price","open time","symbol","lots");
    int total=OrdersTotal();
    // write open orders
    for(int pos=0;pos<total;pos++)
       {
        if(OrderSelect(pos,SELECT_BY_POS)==false) continue;
       FileWrite(handle,OrderTicket(),OrderOpenPrice(),OrderOpenTime(),
       OrderSymbol(),OrderLots());
       }
FileClose(handle);
```

---

## OrderSwap:

### ***Syntaxe:***

```
double OrderSwap( )
```

### ***Popis:***
Funkce *OrderSwap* vrací hodnotu swap (přerolování) vybraného příkazu.
Před voláním funkce *OrderSwap* musí být příkaz vybrán pomocí funkce *OrderSelect.*

### ***Parametry:***
Tato funkce nemá parametry a vrací hodnotu typu **double** (hodnotu swap vybraného příkazu).

### ***Příklad:***

```
if(OrderSelect(order_id, SELECT_BY_TICKET)==true)
    Print("Swap for the order #", order_id, " ",OrderSwap());
  else
    Print("OrderSelect failed error code is",GetLastError());
```

---

## OrderSymbol:

### ***Syntaxe:***

```
string OrderSymbol( )
```

### ***Popis:***
Funkce *OrderSymbol* vrací symbol měnového páru vybraného příkazu (Příklad: EURUSD and USDJPY).
Před voláním funkce *OrderSymbol* musí být příkaz vybrán pomocí funkce *OrderSelect.*

### ***Parametry:***
Tato funkce nemá parametry a vrací hodnotu typu **string** (symbol měnového páru vybraného příkazu).

***Příklad:***

```
if(OrderSelect(12, SELECT_BY_POS)==true)
    Print("symbol of order #", OrderTicket(), " is ", OrderSymbol());
  else
    Print("OrderSelect failed error code is",GetLastError());
```

---

## OrderTakeProfit:

***Syntaxe:***

```
double OrderTakeProfit( )
```

***Popis:***
Funkce *OrderTakeProfit* vrací hodnotu *takeprofit* vybraného příkazu, kterou jsme zadali pomocí funkce *OrderSend* nebo modifikovali pomocí funkce *OrderModify*.
Před voláním funkce *OrderTakeProfit* musí být příkaz vybrán pomocí funkce *OrderSelect.*

***Parametry:***
Tato funkce nemá parametry a vrací hodnotu typu **double** (hodnotu *takeprofit* vybraného příkazu).

***Příklad:***

```
if(OrderSelect(12, SELECT_BY_POS)==true)
     Print("Order #",OrderTicket()," profit: ", OrderTakeProfit());
  else
    Print("OrderSelect failed error code is ",GetLastError());
```

---

## OrderTicket:

***Syntaxe:***

```
int OrderTicket( )
```

***Popis:***
Funkce *OrderTicket* vrací číslo štítku vybraného příkazu. Před voláním funkce *OrderTicket* musí být příkaz vybrán pomocí funkce *OrderSelect.*

***Parametry:***
Tato funkce nemá parametry a vrací celočíselnou hodnotu typu **int** (číslo štítku vybraného příkazu).

***Příklad:***

```
if(OrderSelect(12, SELECT_BY_POS)==true)
    order=OrderTicket();
  else
    Print("OrderSelect failed error code is",GetLastError());
```

**Coders' Guru**
20-12-2005