Hedging Addendum

In March 2016, MetaQuotes updated the MT5 platform to allow MT4-style hedging of positions. The changes to the platform and the language are described in this MQL5.com article:

> https://www.mql5.com/en/articles/2299

The source code as described in the book will not work on hedging accounts, as it was written for "netting" accounts. We've added a new include file, `TradeHedge.mqh`, that contains trading functions that will work on hedging accounts.

To use the hedging include file, add this include directive to the top of your MQ5 file:

```
#include <Mql5Book\TradeHedge.mqh>
CTradeHedge Trade;
```

Remove any references to the `Mql5Book\Trade.mqh` file, as it is already included.

The `TradeHedge` class extends the `Trade` class included in the `Trade.mqh` file. All of the functions in the `Trade` class are still accessible to your program. The only change to the `Trade.mqh` file was to change the `private` keyword on line 33 to `protected`, to allow us to inherit these functions in our `TradeHedge` class.

`TradeHedge` contains five public functions for managing trades on hedging accounts: `BuyHedge(), SellHedge(), ModifyHedge(), CloseHedge()` and `CloseByHedge()`. Use these functions instead of the trade functions in `Trade.mqh` if you are trading on a hedging account.

Hedging Trade Functions

```
ulong BuyHedge(string pSymbol, double pVolume, double pStop = 0,
      double pProfit = 0, string pComment = NULL);

ulong SellHedge(string pSymbol, double pVolume, double pStop = 0,
      double pProfit = 0, string pComment = NULL);
```

Use these functions to open a buy or sell position on a hedging account. Both functions return a ticket number, which can be stored for later use.

```
bool ModifyHedge(ulong pTicket, double pStop, double pProfit = 0);
```

Modifies the stop loss and take profit of a hedging position. You must specify the ticket number of the order to modify.

```
bool CloseHedge(ulong pTicket, double pVolume = 0, string pComment = NULL);
```

Closes a position on a hedging account. You must specify the ticket number of the order to close.

```
bool CloseByHedge(ulong pTicket, ulong pOppositeTicket, double pVolume = 0,
      string pComment = NULL);
```

This function uses MT5s new *Close By* functionality, which allows you to close two opposing positions at once and pay only the spread on one position. You must specify the ticket numbers of both orders to close.

Position Information Functions

The position information functions in `Trade.mqh` (`PositionType()`, `PositionStopLoss()`, etc.) have been overloaded to accept a ticket number. To use these functions on hedging accounts, pass a ticket number instead of a symbol:

```
long posType = PositionType(ticket);
```

When working with a hedging position, use the `PositionSelectByTicket()` or `PositionGetTicket()` functions in MQL5 to select a position and work with it using MQL5's built-in `PositionGet...()` functions. The position information functions described above demonstrate this concept.

Examples

The code in the `TradeHedge.mqh` file has been only slightly modified from the classes and functions in `Trade.mqh`, so if you've read the relevant sections of the book, the code should be easy to understand. An example expert advisor, `Moving Average Hedging.mq5`, has been included in the `Experts` folder to demonstrate how to use the hedging functions in a trading system.

The process of managing orders on hedging accounts is similar to the process of managing orders in MT4 expert advisors. Some considerations to keep in mind when writing trading systems for a hedging account:

You will need to set the deviation and the magic number in the `OnInit()` function. These can be done using the `CTrade::Deviation()` and `CTrade::MagicNumber()` functions, as demonstrated in the example EA above. The magic number is used when trading with multiple expert advisors at once. It allows you to select only the trades that match the magic number set by the user.

It is necessary to manage ticket numbers when using a hedging account. You can store the ticket numbers returned by the `BuyHedge()` and `SellHedge()` functions into static or global variables, or use the `PositionGetTicket()` function in MQL5 to retrieve your order tickets. Here's an example of how to loop through all open positions:

```
for(int i = 0; i < PositionsTotal(); i++)
{
      int ticket = PositionGetTicket(i);
      if(PositionMagicNumber(ticket) != MagicNumber) continue;
      else break;
}
```

This code returns the ticket number of each order to the `ticket` variable, and then checks the magic number of that order using our `PositionMagicNumber()` function. If the order matches our magic number, then we exit the loop, and the `ticket` variable will contain the ticket number of the oldest order that matches our magic number.

Disclaimer