MT4 Server API manual

Exception handling

We recommend any API calls surround with try-catch. Examples below could be without try-catch but in real applications be attentive to exception handling. See next section for try-catch example.

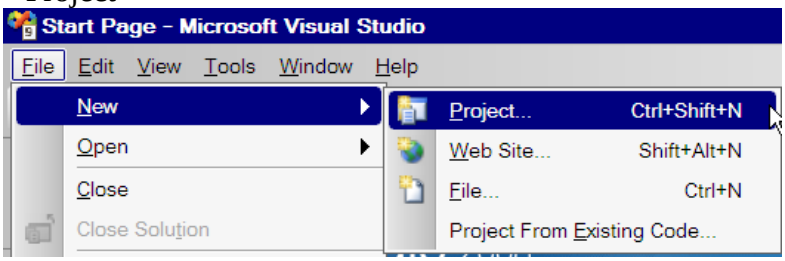
Main classes

`QuoteClient` – real time quotes, bar history, account information, order history, monitoring of trading activity.
`OrderClient` – order send, close, delete, modify.

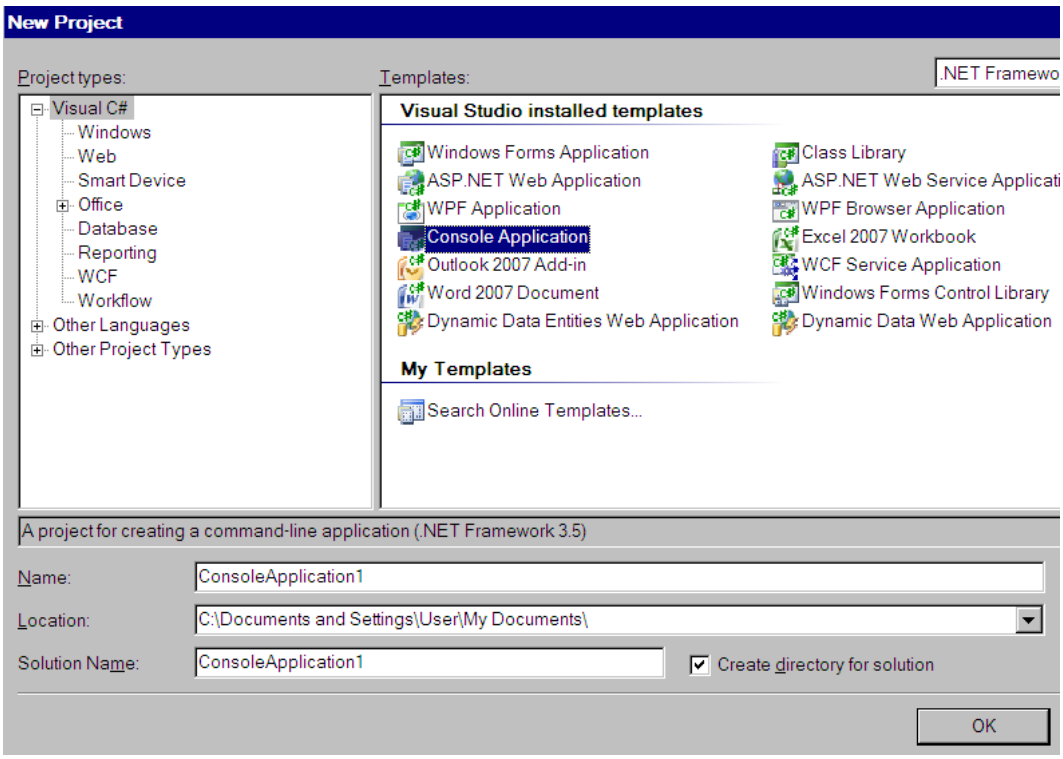
Project in Visual Studio step by step

1. Open Visual Studio

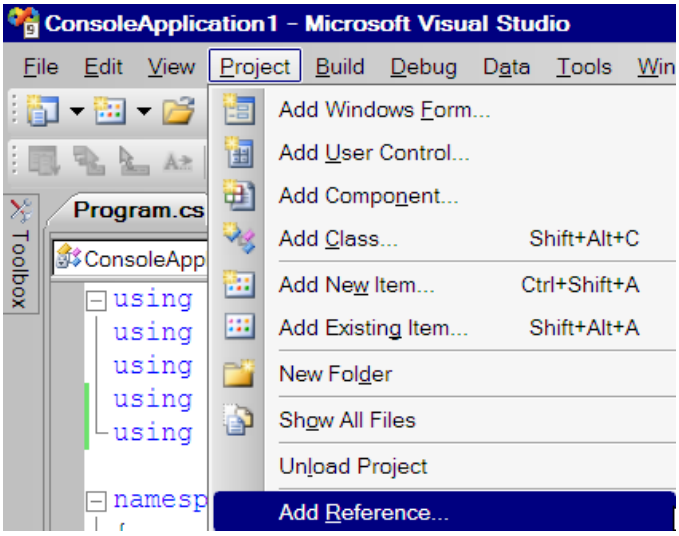
2. Menu > File -> New > Project



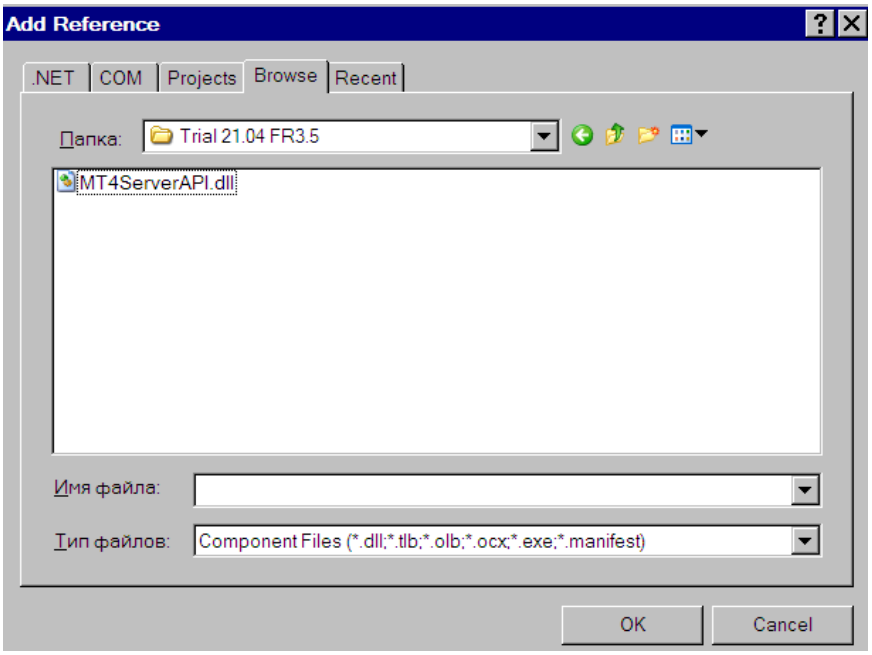
3. Visual C# > Console Application -> OK



4. Menu > Project > Add Reference



5. Choose MT4ServerAPI.dll and press OK.



Connecting to server

MT4 contains server details (address and port) in *.srv files. For simple application is enough to get just primary server as shown below.

```
using System;
using TradingAPI.MT4Server;

namespace Help
{
  class Program
  {
    static void Main(string[] args)
    {
      new Program().Run();
    }

    void Run()
    {
      try
      {
        MainServer srv = QuoteClient.LoadSrv(
          @"C:\Program Files\MetaTrader 4\config\MetaQuotes-Demo.srv");
        QuoteClient qc = new QuoteClient(1718059, "p2ripnt", srv.Host, srv.Port);
        Console.WriteLine("Connecting...");
        qc.Connect();
        Console.WriteLine("Connected to server");
        Console.WriteLine("Press any key...");
        Console.ReadKey();
        qc.Disconnect();

      }
      catch (Exception ex)
      {
        Console.WriteLine(ex.Message);
      }
    }
  }
}
```

In real applications you need to get also slave servers and connect to them if primary server not available.

```
void Run()
{
  try
  {

    Server[] slaves;
    MainServer primary = QuoteClient.LoadSrv(
      @"C:\Program Files\MetaTrader 4\config\MetaQuotes-Demo.srv", out slaves);
    QuoteClient qc = Connect(primary, slaves, 1718059, "p2ripnt");
    Console.WriteLine("Connected to server");
  }
  catch (Exception ex)
  {
    Console.WriteLine(ex.Message);
  }
  Console.WriteLine("Press any key...");
  Console.ReadKey();
}

QuoteClient Connect(MainServer primary, Server[] slaves, int user, string password)
{
  Console.WriteLine("Connecting...");
  QuoteClient qc = new QuoteClient(user, password, primary.Host, primary.Port);
  try
  {
    qc.Connect();
    return qc;
  }
  catch (Exception)
  {
    Console.WriteLine("Cannot connect to main server");
    return ConnectSlaves(slaves, user, password);
  }
}

QuoteClient ConnectSlaves(Server[] slaves, int user, string password)
{
  Console.WriteLine("Connecting to slaves...");
  foreach (var server in slaves)
  {
    QuoteClient qc = new QuoteClient(user, password, server.Host, server.Port);
    try
    {
      qc.Connect();
      return qc;
    }
    catch (Exception)
    {
    }
  }
  throw new Exception("Cannot connect to slaves");
}
```

Broker could periodically change .srv file, if you want to get such updates use PathForSavingSrv field. If you set this field before Connect() method it would be updated during connection.

```
MainServer srv = QuoteClient.LoadSrv(
  @"C:\Program Files\MetaTrader 4\config\MetaQuotes-Demo.srv");
QuoteClient qc = new QuoteClient(1718059, "p2ripnt", srv.Host, srv.Port);
```

```
qc.PathForSavingSrv = @"C:\Program Files\MetaTrader 4\config\";
qc.Connect();
```

Real time quotes

```
void Run()
{
  try
  {
    MainServer srv = QuoteClient.LoadSrv(
      @"C:\Program Files\MetaTrader 4\config\MetaQuotes-Demo.srv");
    QuoteClient qc = new QuoteClient(1718059, "p2ripnt", srv.Host, srv.Port);
    Console.WriteLine("Connecting...");
    qc.Connect();
    Console.WriteLine("Connected to server");
    qc.OnQuote += new QuoteEventHandler(qc_OnQuote);
    qc.Subscribe("EURUSD");
  }
  catch (Exception ex)
  {
    Console.WriteLine(ex.Message);
  }
  Console.WriteLine("Press any key...");
  Console.ReadKey();
}

void qc_OnQuote(object sender, QuoteEventArgs args)
{
  Console.WriteLine(args.Symbol + " " + args.Bid + " " + args.Ask);
}
```

Market order

```
MainServer srv = QuoteClient.LoadSrv(
  @"C:\Program Files\MetaTrader 4\config\MetaQuotes-Demo.srv");
QuoteClient qc = new QuoteClient(1718059, "p2ripnt", srv.Host, srv.Port);
Console.WriteLine("Connecting...");
qc.Connect();
OrderClient oc = new OrderClient(qc);
Console.WriteLine("Connected to server");
while (qc.GetQuote("EURUSD") == null)
  Thread.Sleep(10);
double ask = qc.GetQuote("EURUSD").Ask;
Order order = oc.OrderSend("EURUSD", Op.Buy, 0.1, ask, 0, 0, 0, null, 0, new
DateTime());
Console.WriteLine("Order " + order.Ticket + " opened");
Console.WriteLine("Press any key...");
Console.ReadKey();
qc.Disconnect();
```

Requote handling and slippage

To get symbol execution type of the symbol use following:

```
Execution execution = qc.GetSymbolInfo("EURUSD").Execution;
```

If symbol has Market execution slippage parameter doesn’t work, it also doesn’t depend what open price you put during order sending. Market execution means that position would be opened at any comfortable for broker price.

Instant execution means that you can specify maximum slippage and if price will go against you for more pips than specified in slippage parameter requote exception would be thrown.

```
double ask = qc.GetQuote("EURUSD").Ask;
Order order;
try
{
  order = oc.OrderSend("EURUSD", Op.Buy, 0.1, ask, 0, 0, 0, null, 0, new DateTime());
}
catch (RequoteException ex)
{
   ask = ex.Ask; // new price returned by server
}
```

Pending order

```
MainServer srv = QuoteClient.LoadSrv(
  @"C:\Program Files\MetaTrader 4\config\MetaQuotes-Demo.srv");
QuoteClient qc = new QuoteClient(1718059, "p2ripnt", srv.Host, srv.Port);
Console.WriteLine("Connecting...");
qc.Connect();
OrderClient oc = new OrderClient(qc);
Console.WriteLine("Connected to server");
Order order = oc.OrderSend("EURUSD", Op.BuyStop, 0.1, 1.4, 0, 0, 0, null, 0, new
DateTime());
Console.WriteLine(order.Type + "order " + order.Ticket + " opened");
Console.WriteLine("Press any key...");
Console.ReadKey();
qc.Disconnect();
```

Monitoring account trading activity

With API you could monitor any trading activity making by all clients connected to this account. This functionality should be useful for trade copiers.

```
void Run()
{
  try
  {
    MainServer srv = QuoteClient.LoadSrv(
      @"C:\Program Files\MetaTrader 4\config\MetaQuotes-Demo.srv");
    QuoteClient qc = new QuoteClient(1718059, "p2ripnt", srv.Host, srv.Port);
    qc.OnOrderUpdate += new OrderUpdateEventHandler(qc_OnOrderUpdate);
    Console.WriteLine("Connecting...");
    qc.Connect();
    Console.WriteLine("Try to open/close trades on this account in MT4 terminal to see
updates here. Press any key...");
    Console.ReadKey();
    qc.Disconnect();
  }
  catch (Exception ex)
  {
    Console.WriteLine(ex.Message);
    Console.WriteLine("Press any key...");
    Console.ReadKey();
  }
}

void qc_OnOrderUpdate(object sender, OrderUpdateEventArgs update)
{
  Console.WriteLine(update.Action + " " + update.Order.Ticket);
}
```

Asynchronous functions

All trading functions(OrderSend, OrderClose, OrderModify, OrderDelete) and Connect function have asynchronous variants that have Async postfix. Execution of asynchronous function goes in another thread so you need to receive results with using events OnConnect(for ConnectAsync) and OnOrderProgress(for trading functions). Example below shows how to send two orders simultaneously. This also should be useful for trade copiers because after getting order update from master account you can send that update to all slaves simultaneously.

```
void Run()
{
  try
  {
    MainServer srv = QuoteClient.LoadSrv(
      @"C:\Program Files\MetaTrader 4\config\MetaQuotes-Demo.srv");
    QuoteClient qc = new QuoteClient(1718059, "p2ripnt", srv.Host, srv.Port);
    Console.WriteLine("Connecting...");
    qc.Connect();
    OrderClient oc1 = new OrderClient(qc);
    oc1.OnOrderProgress += new OrderProgressEventHandler(oc_OnOrderProgress);
    OrderClient oc2 = new OrderClient(qc);
    oc2.OnOrderProgress += new OrderProgressEventHandler(oc_OnOrderProgress);
    Console.WriteLine("Connected to server");
    while (qc.GetQuote("EURUSD") == null)
      Thread.Sleep(10);
    double ask = qc.GetQuote("EURUSD").Ask;
    int id1 = oc1.OrderSendAsync("EURUSD", Op.Buy, 0.1, ask, 0, 0, 0, null, 0, new
DateTime());
    Console.WriteLine("Order with RequestID = " + id1 + " sent");
    int id2 = oc2.OrderSendAsync("EURUSD", Op.Buy, 0.1, ask, 0, 0, 0, null, 0, new
DateTime());
    Console.WriteLine("Order with RequestID = " + id2 + " sent");
    Console.WriteLine("Press any key...");
    Console.ReadKey();
    qc.Disconnect();
  }
  catch (Exception ex)
  {
    Console.WriteLine(ex.Message);
    Console.WriteLine("Press any key...");
    Console.ReadKey();
  }

}

void oc_OnOrderProgress(object sender, OrderProgressEventArgs args)
{
  if (args.Exception == null)
  {
    Console.WriteLine("ReuestID = " + args.TempID + ", State = " + args.Type);
    if (args.Type == ProgressType.Opened)
      Console.WriteLine("ReuestID = " + args.TempID + ", Ticket = " +
args.Order.Ticket);
  }
  else
    Console.WriteLine(args.Exception.Message);
}
```

Opened orders

```
MainServer srv = QuoteClient.LoadSrv(
  @"C:\Program Files\MetaTrader 4\config\MetaQuotes-Demo.srv");
QuoteClient qc = new QuoteClient(1718059, "p2ripnt", srv.Host, srv.Port);
Console.WriteLine("Connecting...");
qc.Connect();
Console.WriteLine("Connected to server");
foreach (Order order in qc.GetOpenedOrders())
  Console.WriteLine(order);
Console.WriteLine("Press any key...");
Console.ReadKey();
qc.Disconnect();
```

History orders

```
MainServer srv = QuoteClient.LoadSrv(
  @"C:\Program Files\MetaTrader 4\config\MetaQuotes-Demo.srv");
QuoteClient qc = new QuoteClient(1718059, "p2ripnt", srv.Host, srv.Port);
Console.WriteLine("Connecting...");
qc.Connect();
Console.WriteLine("Connected to server");
DateTime from = DateTime.Now.AddDays(-1);
DateTime to = DateTime.Now;
foreach (Order order in qc.DownloadOrderHistory(from, to))
  Console.WriteLine(order);
Console.WriteLine("Press any key...");
Console.ReadKey();
qc.Disconnect();
```

From and To parameters it's server time, not local time. Some servers doesn't interpret From and To time precisely so we recommend to subsctruct several hours from From time and add several hours to To time to get orders from required range with guarantee.

Server time

Server time updates with every quote so you need to get at least one quote to use it.

```
MainServer srv = QuoteClient.LoadSrv(
  @"C:\Program Files\MetaTrader 4\config\MetaQuotes-Demo.srv");
QuoteClient qc = new QuoteClient(1718059, "p2ripnt", srv.Host, srv.Port);
Console.WriteLine("Connecting...");
qc.Connect();
Console.WriteLine("Connected to server");
qc.Subscribe("EURUSD");
while (qc.ServerTime == DateTime.MinValue)
  Thread.Sleep(10);
Console.WriteLine(qc.ServerTime);
Console.WriteLine("Press any key...");
Console.ReadKey();
qc.Disconnect();
```

Quote history

```
void Run()
{
  try
  {
    MainServer srv = QuoteClient.LoadSrv(
      @"C:\Program Files\MetaTrader 4\config\MetaQuotes-Demo.srv");
```

```
    QuoteClient qc = new QuoteClient(1718059, "p2ripnt", srv.Host, srv.Port);
    qc.OnQuoteHistory += new QuoteHistoryEventHandler(qc_OnQuoteHistory);
    Console.WriteLine("Connecting...");
    qc.Connect();
    Console.WriteLine("Connected to server");
    qc.Subscribe("EURUSD");
    // ServerTime update goes with quotes, wait for first quote
    while (qc.ServerTime == new DateTime())
      Thread.Sleep(10);
    //request 3 previuos bars
    Timeframe tf = Timeframe.M5;
    qc.DownloadQuoteHistory("EURUSD", tf, qc.ServerTime.AddMinutes(-3 * (int)tf), 0);
    Console.WriteLine("Press any key...");
    Console.ReadKey();
    qc.Disconnect();
  }
  catch (Exception ex)
  {
    Console.WriteLine(ex.Message);
  }
}

void qc_OnQuoteHistory(object sender, QuoteHistoryEventArgs args)
{
  if (args.Bars.Length == 1 && args.Bars[0].Time == new DateTime(1970, 1, 1))
    return; //null message means that it's the last message
  foreach (Bar bar in args.Bars)
    Console.WriteLine(bar);
}
```