**Minimal requirements**

* Windows 7 (or higher) or Windows Server 2008 (or higher);
* RAM 8 GB;
* Microsoft .Net 4.6.1;
* CPU x86 or x64;
* HDD 50 MB.

**How to install TradingServer**

We need git client to pull the latest version of TradingServer from repository. Follow this steps to get latest version:

* Create folder for TradingServer
* Run cmd.exe
* Change folder to the one created in previous step (“cd {folder path}”)
* Clone repository using command “git clone {url}”
* Build solution using Visual Studio IDE or cmd “net build” command

**Database**

TradingServer uses MS SQL Server to store data. Please create database (default name “TradingServer”) and then open script from file **script.sql** and execute it. This file contains a SQL Script that will create a database structure. Database connection settings located in **DataBaseConnection.set** file (image 1). This file is located in “Server” project and copied to execute directory when the project compiled. By default there no user in database. To be able login need to add users to **Users** table in database.

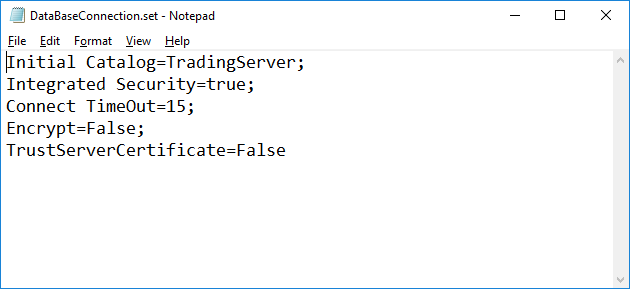


Image 1 - DataBaseConnection.set file.

**RabbitMQ Server Installation**

Download and install RabbitMQ server from <https://www.rabbitmq.com/> and Erlang from <http://www.erlang.org/> for Scripting Service <> TradingServer messages exchange.

**NLog**

TradingServer uses NLog library (see <https://github.com/NLog/NLog/wiki>) to log all kinds of info: error, info, debug, trace. ‘.\Log\YYYYMMDD’ subfolder contains all log files per day. NLog.config file may be changed to customize format of logs.

**Server Configuration**

Server is main application. Server communicate with ScriptingService via WCF and RabbitMQ. RabbitMQ configuration is location in server **app.config** file. This is how configuration file look:

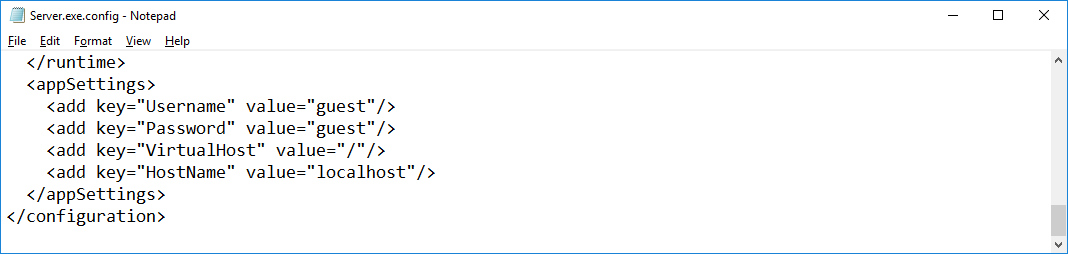


Image 2 - Server configuration.

**Services configuration**

TradingServer supports 3 connection services: WCFServiceHost, WebSocketServiceHost and RabbitMQServiceHost. It allows to connect client application to server via any of these connection services. Services projects located in ServiceHosts directory. Services get configuration data from personal **app.config** file located inside project. Services libraries and all necessary files copied to “ServiceHosts“ directory in execute folder when the project compiled. **app.config** files copied to “ServiceHosts“ directory and name changed to LibraryName + “.config” (for example WCFServiceHost.dll - WCFServiceHost.dll.config). This is how configuration files look for each services:

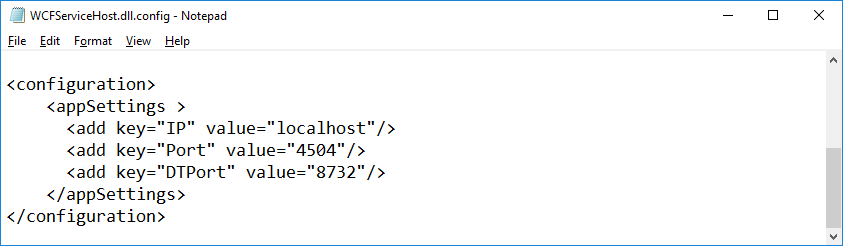


Image 3 - WCF service configuration.

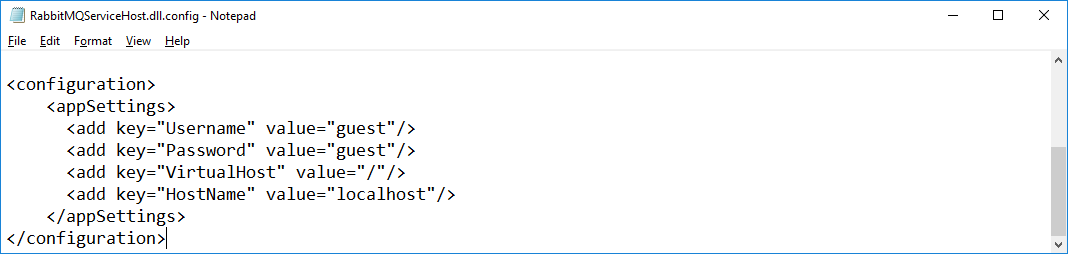


Image 4 – RabbitMQ service configuration.

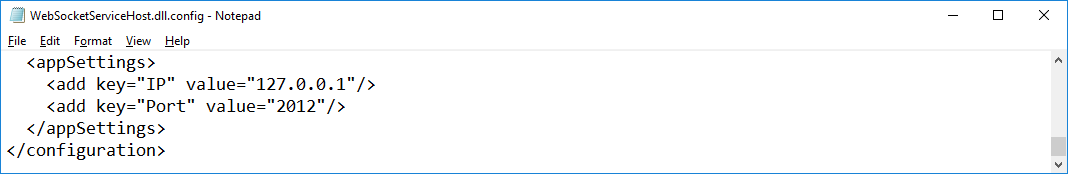


Image 5 – WebSocket service configuration.

**Datafeeds configuration**

TradingServer supports 3 DataFeeds: SimulatedDataFeed, LmaxDataFeed and PoloniexDataFeed. These projects located in DataFeeds directory. LMAX and Poloniex DataFeeds get configuration data from personal **app.config** file located inside project. DataFeeds libraries and all necessary files copied to “DataFeeds“ directory in execute folder when the project compiled. **app.config** files copied to “DataFeeds“ directory and name changed to LibraryName + “.config” (for example LmaxDataFeed.dll - LmaxDataFeed.dll.config). This is how configuration files look for each DataFeed:

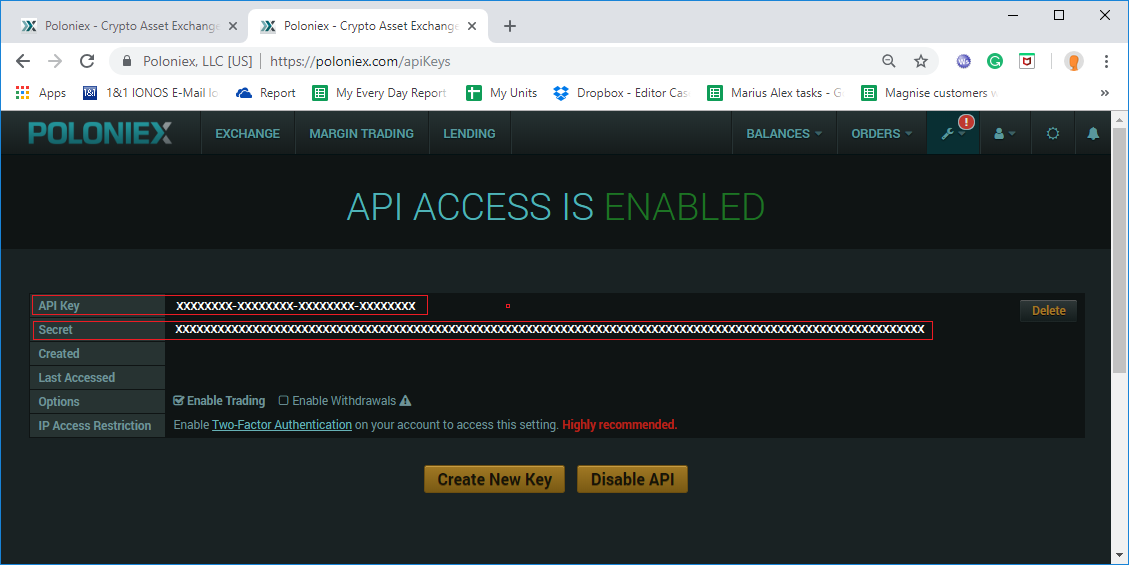
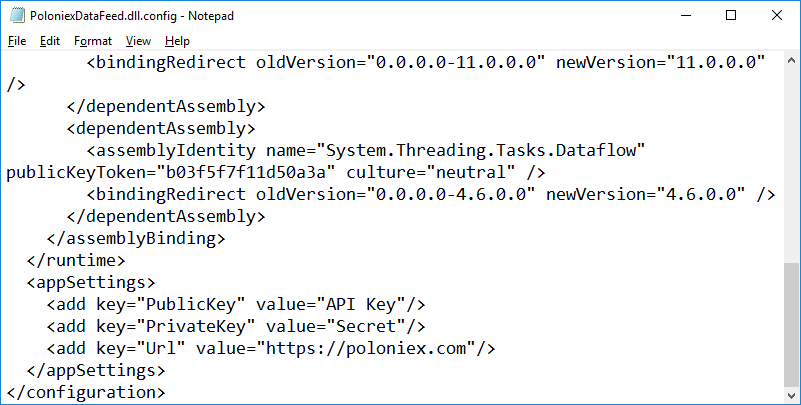


Image 6 - Poloniex configuration.

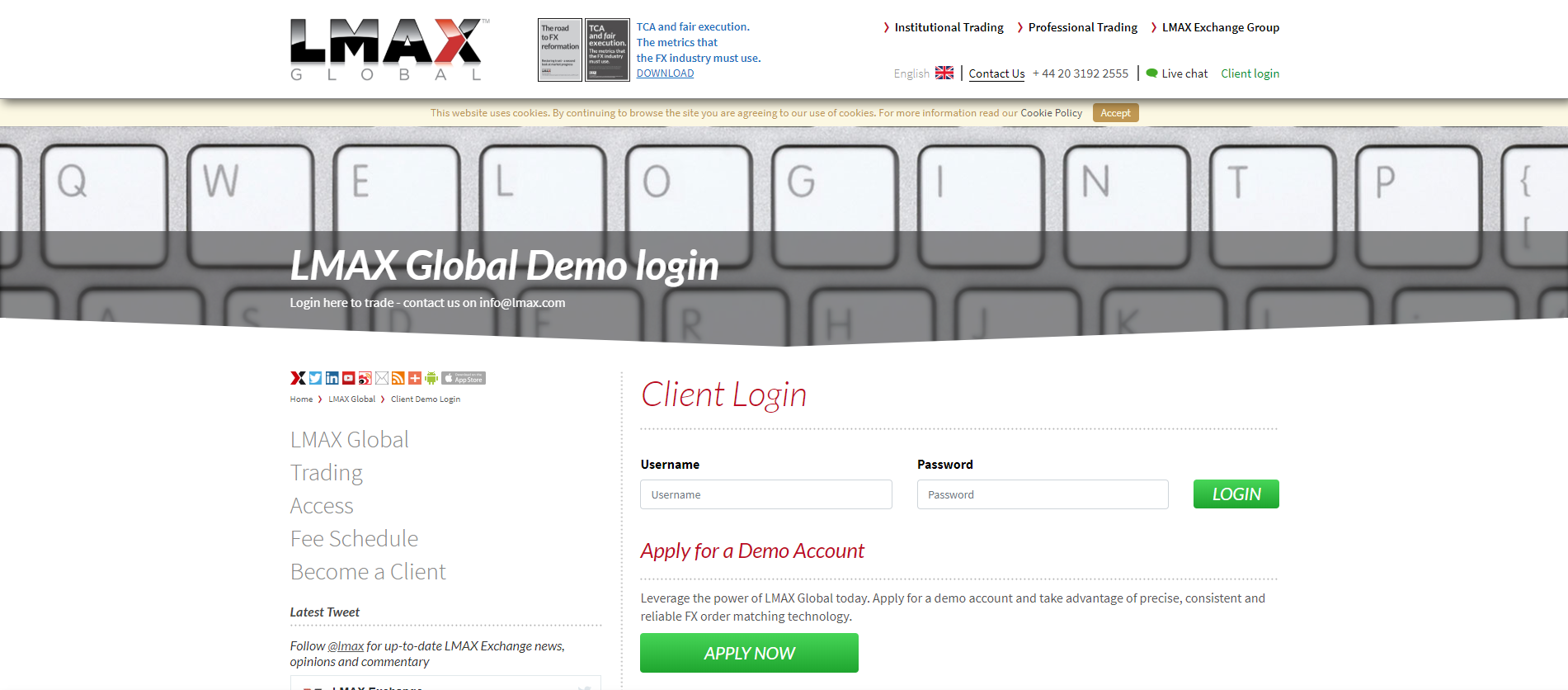
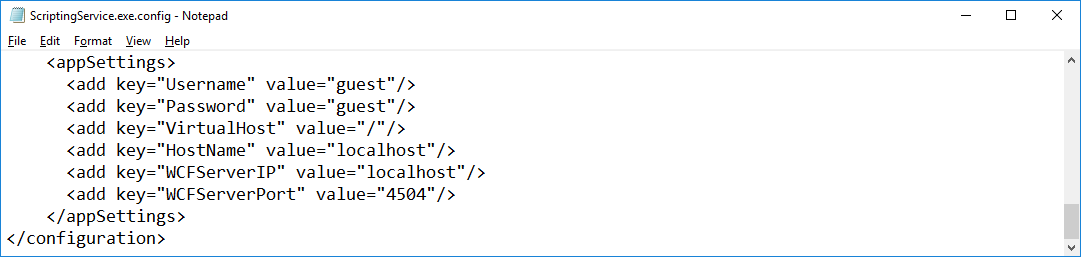
 

Image 7 - LMAX configuration.

**Scripting Services Configuration**

This project located in Scripting directory. Scripting Service communicate with Server via WCF and RabbitMQ. Server WCF and RabbitMQ configuration described above - image 3 and image 2. Scripting Service supports many instances on one or many ma­­chines, so you can parallel your signals execution. (For example: open solution from Visual Studio, start server, then right click on ScriptingService project and use “Debug->Start new instance” command from context menu) . This is how ScriptingService configuration file look:

 Image 8 - Scripting Service configuration.