Algo-trading market emulator LLD

## **Project's General Structure**

We designed our project according to 3 tier architecture - we separated our project into 3 layers . The first layer, the Presentation layer's role is to interact with the user through a fixed communication module where he can type in his request in a specified format. The second layer, the Business layer receives the input string from the PL and analyzes it, then sends it to suitable method in the Data Access Layer which is the third layer - this layer is responsible for communication with the server, and returns the relevant information that the server returns in response which will be printed to the user via the console.

## **Presentation Layer (PL)**

#### Functionality

As mentioned before the Presentation Layer (PL) is responsible for communication and interaction with the consumers/users. While the user use the system (and haven't chose to exit the system), the system uses the next methods:

#### AsciSilverTongue Class - Methods

public AsciiSilverTongue()- Default Constructor

public void PrintWelcome()- Just a title for designing the home page.

public void PrintGoodbye() – Print a message for exiting from the system.

public void PrintMenu()- Show a menu for the visitors with the possible using system options.

public static void printServerData(Object obj)- Prints information from the server.

public string ReadLine()-Returns the input of the user with the wanted request.

#### Program Class

This is the default class of the project - no use at all

#### Presentation Class

This class starts the main loop of the communication module which runs the methods of AsciSilverTongue Class.

The division between the 2 last classes was created for the good order, and comfortable reading.

## **Business Layer (BL)**

#### Functionality

This layer is responsible for committing the right method, depends of the user's request, notification of wrong typing, converting user's input to understandable data for the server, and updating of the information in the system.

#### Socket Class - Methods

public Socket()- a constructor of the class.

public static void printNoValidCommandError() – prints an exception when the user types a wrong input.

public static int generalStringToInt(String str,int errorVal, string errorMsg) –convert an input from string to int.

public static int idStringToInt(String str,int errorVal, string errorMsg)-Responsible for converting input of id for requests from string to int.

public static void sell(String str) – This method gets an input of the required commodity for selling, and the method commits sell activity.

public static void buy(String str) – This method gets an input of the required commodity for purchasing, and the method buy the wanted product.

public static void cancel(String str) – This method gets an input of the required commodity for purchasing, and the method buy the wanted product.

public static void findInfo(String str)- The method interprets the input and redirect to the suit query. If there is an incorrect input, error message will be sent.

public static bool userInfo – The method returns details about the current client.

#### Program Class – a default class.

#### Parser Class - Methods

The main purpose of this class is to analyze the input, and committing the suit method.

#### Method

public static void parser(String str)- Gets an input of String from the user with his requirments. The method interprets the input and redirecting to the suit code.

## **Data Access Layer (DAL)**

The following classes represents instances of request/query's requests which delivered for the server by JSON tool.

#### SellRequest Class - Methods

public SellRequest(int commodity, int amount, int price)- Gets some details from BL and initializes new Sell Request object for the server.

#### QueryUserRequest Class - Methods

public QueryUserRequest()- initializes a new user query.

#### QueryMarketRequest Class - Methods

public QueryMarketRequest(int commodity)- Gets id of commodity from BL and creates an object which describes a market query.

#### QueryBuySellRequest Class - Methods

public QueryBuySellRequest(int id)- Initializes an object which describes a buy/sell request.

#### CancelBuySellRequest Class - Methods

public QueryCancelBuySellRequest(int id)- Initializes a new object of cancel request with a given id of the order.

#### BuyRequest Class - Methods

public BuyRequest(int commodity, int amount, int price)

Gets some details from BL and initializes new Buy Request object for the server.

#### MarketCommodityOffer Class - Methods

public override string ToString()- The method prints information about the commodity's offers which received from the server.

* This class implements IMarketCommodityOffer interface.

#### QueryUserRequest Class – Method

#### public QueryUserRequest()- The method initializes a new object which describes the type of the commodity.

#### QueryBuySellRequest Class – Method

#### public QueryrBuySellRequest(int id)- The method initializes a new object which describes details about an order. This object is supposed to be sent to the server.

MarketUserData Class- Method

public override string ToString()- This method initializes a new object with received data from the server.

* This class implements IMarketUserData interface.

MarketItemQuery Class- Method

public override string ToString()- The method prints received data from the server about the items.

* This class implements IMarketItemQuery interface.

The next class coordinates ways of communicating with the server:

MarketClient Class- Methods

public MarketClient()-initializes new object which contains details of client, and information about his authentication.

private void printError(string error)- prints an error.

private string SendRequest<T>(T data)-

private T2 SendRequest<T1, T2>(T1 data) where T2 : class- This method sends request to the server with Object with type of T1, and receives from the server an object with type of T2 with the relevant information.

public int SendBuyRequest(int price, int commodity, int amount)- Sending a buy request and returns the answer from the server.

public int SendSellRequest(int price, int commodity, int amount)- Sending a buy request and returns the answer from the server.

public IMarketItemQuery SendQueryBuySellRequest(int id)

public IMarketUserData SendQueryUserRequest()- The method asks from the server information about current user.

public IMarketCommodityOffer SendQueryMarketRequest(int commodity)- The method asks from the server information about received commodity.

public bool SendCancelBuySellRequest(int id)- The method sends a request to the server for cancelling deal number id.

* This class implements IMarketClient interface.