Lab 09: Auction Server

CO225: Software Construction

September 13, 2016

1 Introduction

In this lab you will implement a server which can be used by clients to bid for items in an auction.

2 Objective

Objective of this lab is to use

- Threads
- Synchronization primitives
- Sockets
- Collections

for developing somewhat complex applications.

3 Task

Implement a server which can be used by clients to bid for items in an auction. Specification for each component in the software is given below.

- Each item has, among other things a *Symbol*, *Security Name*, and a *Price*. This data is given in a CSV file. Once the system starts to run, the price of the item will vary and what is given in the CSV is the initial price. (see *stocks.csv*)
- Read the CSV file and store the information about the items in a suitable data structure (i.e. suitable collection). Before selecting the data structure read how the data items will be accessed.

Server side

The server should be able to handle more than one connection at a time. It should display the current price of stocks via a GUI. You may assume that stock prices do not change in 500ms.

- Server will be listing to incoming connections on port 2000. It should be able to handle more than one connection at a time. Therefore should use threads for handling the connections.
- You should be able to connect to the server using a common communication tool such as *nc* or *telnet*.
- Once a client is connected, the server should except the first message to be the name of the client. For now we will not authenticate the client but use this as the name for all the bids. Once the name is given the client is expected to provide the Symbol of the security he/she is willing to bid on. If the provided Symbol is found the server should reply back with the current cost of the security or -1 to indicate that the Symbol is invalid.
- Once that is done the clients are not allowed to change neither their names (obviously) nor the security that they are bidding on.
- Server should be able to locate a given stock item, update its price. Furthermore it should be able to track all the changes done to the stock item, how the offers varied with time and who made the offers.

- Server should be able to list the stock items (Symbol, name) together with the current price in a GUI. The GUI should display the price of following Symbols: FB, VRTU, MSFT, GOOGL, YHOO, XLNX, TSLA and TXN.
- For the display, you may assume that updates do not happen in 500ms.

4 Things to consider

This lab involves a fair bit of coding. So first consider each of the operations needed and how that can be implemented before you start coding. You should be able to re-use most of the code from the lectures/labs.

5 What to turn in?

Submit the Java file(s) *AuctionServer.java* (this file should contain the main method to run the class) to FEeLS before the deadline. Do not submit any form of archives such as .zip etc. If you have more than one Java Class submit them without making an archive. You should also add a README file which would explain how to use the application. We will test your system on a Linux machine using *nc* as client.