DISTRIBUTED SYSTEMS ASSIGNMENT [CMPU4021]

MULTITHREADED CLIENT-SERVER ONLINE AUCTION SYSTEM

OLEG PETCOV (C14399846)

DT228/4 Computer Science

23 November 2017

For this assignment you are required to design and implement a distributed application that is a simulator of an on-line auction system.

There will be multiple clients and a server. The server will offer items for sale. The client module will allow the user to bid for items. An item is sold to the highest bidder. Your code will enable users only to bid for items – it will not conduct actual credit card transactions.

Client Specification

- Connects to the server. The item currently being offered for sale and the current bid or a (or reserve price) are displayed.
- Enter the bid. The amount entered should be greater than the current highest bid.
- After a new bid is placed, the amount of the new bid must be displayed on the client's window/console.

Server Specification

- Receive connections from multiple clients.
- After a client connects, notify the client which item is currently on sale and the highest bid (or reserve price).
- Specify the bid period. Max allowed 1 minute. When a new bid is raised, the bid period is reset back.
- When a new bid is placed, all clients are notified immediately. Clients should be notified about the time left for bidding (when appropriate).
- If the bid period elapses without a new bid, then the auction for this item closes. The successful bidder (if any) is chosen and all clients are notified.
- When an auction for one item finishes, another item auctioning should start. Minimum of 5 items should be auctioned, one after another. Only one item at a time.
- Any item not sold should be auctioned again (automatically).

Arguments for Client and Server

Client.bat:

Java Client <host address> <port number> [username]

java -classpath . Client localhost 10000 Tom (username is optional)

Server.bat:

Java Server <port number>

java -classpath . Server 10000

COMPILE.bat: javac -classpath . *.java

The Client (Client.java)

- The client will attempt to connect to the Server application.
    - Will display errors if the command arguments are not correct.
- The user will be prompted to enter a username.
    - Only if It wasn't entered in the initial command

- The Server will return a 'welcome' message to the user, and will output on of the following messages to the user:
    1) Waiting for more users
    2) There is currently an auction ongoing (Starting a new auction)
    3) The item currently being sold

    (*namecommand.png, entername.png*)

- When there are 2 or more users, the user may enter bids for the current item.

    (*twoClientBid.png*)

- The first bid entry may match the reserve price being asked for
    - Every other bid must be higher than the previous bid.

- The user will receive Prompts if their entry is not a valid number (any letters, dashes, etc)
    - Decimal values are allowed
- The user may enter 'help' or 'QUIT' for additional functionality

The ServerListener Thread (ServerListener.java)

- This thread will listen for any messages coming from the Server. This can be:
    1) Setting user ID (IDCONF::ID)
    2) Getting user ID after they are the highest bidder (GETUID::ID)
    3) Sending new highest bid info
    4) Time left for bidding on an item
    5) Server disconnects
    6) Incorrect bid entry

The Server (Server.java)

The Server handles computing bids and outputting messages to all Clients.

- The Server will continually wait for a Client to connect to it.

- When a Client does connect to it, it will add them to a Client list.
    - If there is only one user, It will wait for another Client before starting a new auction process
    - Will not move onto another item until the minimum two bidders requirement is met

- After accepting a connection will set an ID to this Client
    - Because there can be multiple "*Dave's"* connecting to the Server, but only one with ID 1 and another with ID 12

- Will then proceed to send messages dependent on status of Clients list size, and item list size
        - If either list is too short, will not start / continue the auctioning process

- *offerItems()* will start the auctioning process form here on out
    - Will add a new item for users to bid on

- *broadcastAll()* will broadcast a message to all users
- *broadcastBid()* will broadcast the new highest bid to all users besides the highest bidder
    - They get their own message
- *broadcastWin()* will send message with ID of user, item price and item name to all users

- *bidFinished()* This will write the bid to a file (winning_bids.txt) if the item was bid on. Otherwise it will add the unbid on item to the end of the list for later bidding

The ClientHandler Thread (ClientHandler.java)

- Used to give each Client its own Thread for entry
    - Many clients can input a bid at the same time
- It is used to both send messages to the Server, and for the Server to have a way to message the Client .
- Contains logic to get Client ID, and to enter Bid amount
- Can gracefully disconnect from the Server

I Declare that this work, which is submitted as part of my coursework, is entirely my own, except where clearly and explicitly stated.