Assignment 6: Remote Method Invocation (100 Punkte)

In Assignment 6 you should use Java RMI technology for building a client-server application managing and handling auctions. Clients can connect to an auction server, they can offer items and bid for items.

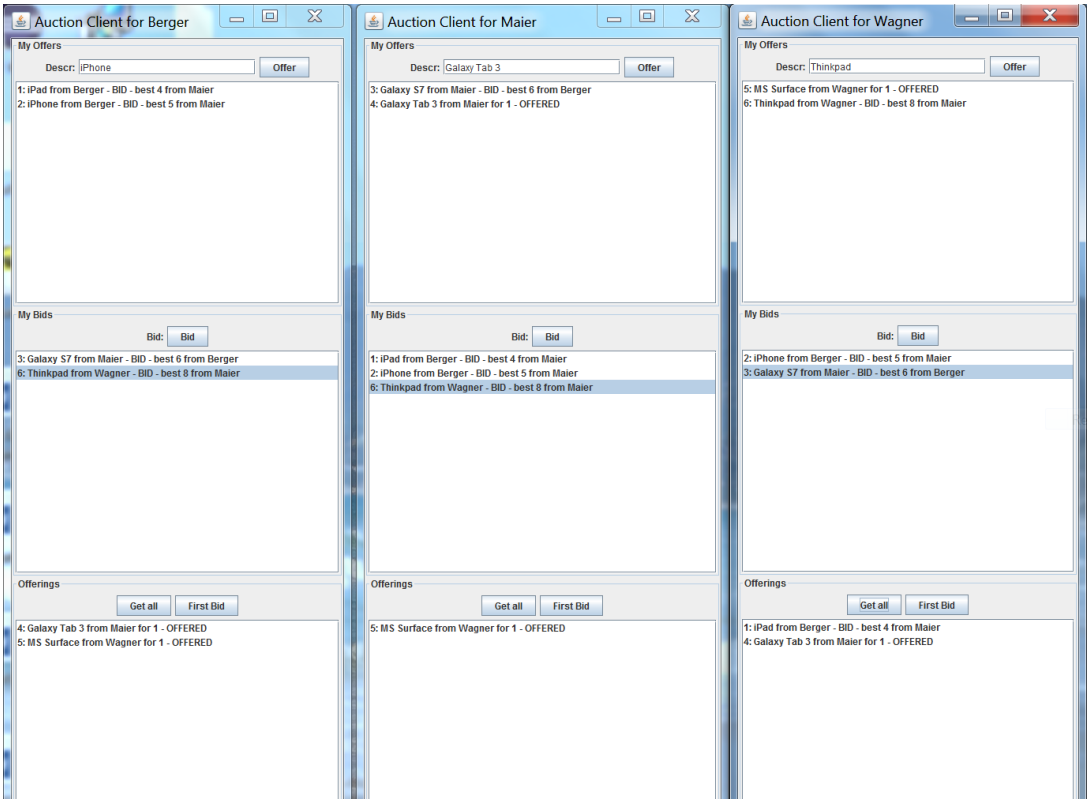
Client program:

The client can connect to the remote server by its name.

Remark: No registration and login procedure is required (omitted for keeping the assignment simple).

When connecting, a remote object is returned from the server and then serves as an interface for this client. It allows the client program to the perform the respective actions. The client can offer items, see the offerings of others, and bid for items. Further, the remote object allows the client to register remote listeners so that it can be notified about changes of items. The client is notified of changes of items he has offered or has bid for.

The following figure shows screenshots of three instances of the interactive client programs.



In the top section of the UI the client can offer items by providing a description (its initial price is 1 EURO). The list shows all the items offered by this client.

The middle section allows the client to bid for items. The list shows all the items he has given bids already. The client can select an item and give a further bid, i.e., he offers 1 EURO more than the current bid.

The bottom section allows displaying all items offered by others. Those items are displayed on demand by clicking button “Get all”. The client then can select one item in the list and give a first bid for it. This item is then added to the bid list.

A Swing frame for the client program without functionality is provided to you in the download. You can use it but it is not a requirement.

Server program:

The server maintains a list of all items and a list of clients. That means, for each client the server maintains one remote object which is used as an interface to the client (see above).

The remote object for the client should maintain a list of items offered by that client and a list of items that client has bid for already. Further, it maintains a list of remote listeners for notifying clients of changes of observed or bid items.

Auctions for items are only active for a specific time. When this time has elapsed, the item is sold to the client with the highest bid or, if the item has no bid, it is not sold but the auction is over.

To summarize, the server program should

- Allow clients to connect by their name; names are unique
- Maintain a remote object for each client which represents the interface for the client. This means that the client will interact with his remote object after connecting.
- With this remote object the client can
    - offer items with a description (with initial price 1 EURO)
    - bid for items, i.e., offer 1 EURO more than the current best bid
    - get notified of any changes in the items offered and items bid
    - can access all the other items and select one to give a first bid for it

Hints:

- Implement a `Serializable` object for auction items.
- Use a `ScheduledExecutorService` for scheduling end of auction events.
- Be aware that remote method calls are executed in separate threads and thus access to shared data has to be synchronized.

Optional task with 30 extra points

Implement a WebService class for accessing the auction server by WebMethods

Implement WebMethods for

- connecting a client
- offering items
- bidding for an item
- getting all offerings
- getting the items offered by a client
- getting the items bid by a client

No listener callback required!

Only one WebService class should be implemented. WebMethods use the client name as parameter for identifying the client.