Distributed Systems
Exercise Sheet 5, Tuesday, 14:15
Klingemann, SS 2023

Deadline: 13th June 2023

2nd Assessed Exercise

1. Date-Service using Java RMI
Understand the example code and test the programs. Client and server can run on the same machine. The nameservice has to be started by means of the rmiregistry command.

2. Distributed account-management-system of a bank using Java RMI
Extend your simple account-management-system of a bank from Sheet 3 to create a client-server-based system using Java RMI. Therefore, you have to transform the objects of the classes account and accounting entry into remote objects. All account- and accounting entry-objects exist only on the server! For each accounting entry there exists a separate object. Similar to the description in Sheet 3, we require that each account has two private attributes: a name and a set of accounting entry objects. This set should represent the accounting entries that belong to the account. (Note that it is a set of objects and not just a number!) Each accounting entry has three private attributes: an amount, a date and the name of the other account involved in the transaction.
An account should have methods with the following functionality. All these methods can be invoked from the client.

- Search for an accounting entry with a particular amount. You can assume that there exists at most one. The method returns a reference to the corresponding accounting entry object.
- Add a new accounting entry. The method has three parameters: an amount, a date and an account-name. A corresponding accounting entry object is created and added to the set of accounting entries.
- Return the set of all accounting entry objects of the account. (The return value has to be a collection of references to accounting entry objects and not just a number!)
- Return the name of the account.

An accounting entry should have methods with the following functionality. All these methods can be invoked from the client.

- Return the amount
- Return the date
- Return the account-name
- Change the amount

Implement a client that is calling all methods of the objects on the server in a sensible manner. In particular, your client should be able to calculate based on the methods above the total amount of all accounting entries of an account. Your application should use two account objects. The nameservice has to be started by means of the rmiregistry command.

Organisatorical matters

- You have to solve the exercise completely on your own! (No working in groups!)
- It is necessary but not sufficient to present a working program. Moreover, you have to be able to explain all parts of your program, be able to answer questions with respect to your program and make small extensions of you program.
- Your program has to be created completely within the exercise slot.
- If you violate one of the rules above, this implies that you definitely fail in this exercise.
- You can only present solutions that correspond to the exercise slot you are assigned to.
- It is in your responsibility to present your solution in time before the deadline. The assessment of your solution can only be guaranteed if you finish your program 60 minutes before the end of the exercises.
- To take part in the exam it is required to solve at least three of five assessed exercise sheets.