Computer Science 2B

Practical Assignment 04

2016-08-30

Deadline: 2016-09-06 12h00 Marks: 100

This practical assignment must be uploaded to eve.uj.ac.za **before** 2016-09-06 12h00. Late or incorrect submissions **will not be accepted**, and will therefore not be marked. You are **not allowed to collaborate** with any other student.

The Java Development Kit (JDK) has been installed on the laboratory computers along with the Eclipse Integrated Development Environment (IDE).

This practical will focus on using **UDP** sockets to implement a custom protocol.

Create a GUI based Java application which will implement a custom protocol that allows users to communicate with one another called **Text A Lot Kommunikation**. For this practical you will be creating both server and client which will have graphical front-ends. In the command structure, all commands are prefixed with a backslash (\). The Server will respond to the following commands received from clients:

- HOWZIT NAME - Clients send the HOWZIT command as initiation of communication. The NAME is the name of the client. The client is added to the list of online users. The server will then send an acknowledge response as follows:

```
AWE NAME, NUMBER_OF_CLIENTS
USERNAME ADDRESS PORT
USERNAME ADDRESS PORT
...
...
...
USERNAME ADDRESS PORT
USERNAME ADDRESS PORT
```

- CHINA - Server responds with current online users (as above).
- TUNE MESSAGE - Sends MESSAGE to all online users.
- SHARP - Client indicates *log off*.

Servers can also send commands to clients.

- EISH MESSAGE - Error has occurred. MESSAGE describes the error.

Clients must keep track of online users received from the server after the HOWZIT command is sent. Clients can also process commands.

- EKSE NAME MESSAGE - Send a message to NAME directly without server interaction. If the recipient is not available then the client must request a new list from the server and try to send the message again.
- EISH MESSAGE - Display the error message.
- SHARP - Client indicates *log off*.

Example of communication.

```
S   - indicates server.
Ca - indicates client A.
Cb - indicates client B.

Server is running
Ca starts up
Ca: \HOWZIT JOHN
S:  AWE JOHN, 0
Cb Starts up
Cb: \HOWZIT JANE
S:  AWE JANE, 1
    JOHN 127.0.0.1 553
Cb: \TUNE I AM AMAZING!
Server broadcasts message to all connected clients namely Ca and Cb.
Ca: \CHINA
S:  AWE JOHN, 2
    JOHN 127.0.0.1 553
    JANE 127.0.0.1 24578
Ca: \EKSE JANE You are so cool!
Ca: \SHARP
Cb: \EKSE JOHN Thanks!
Cb does not get a response.
Cb: \CHINA
S:  AWE JANE, 1
    JANE 127.0.0.1 24578
Cb: \SHARP
```

This practical must make use of Datagram sockets only! If you use other sockets you will be awarded ZERO!

Marksheet

1. Server - Create server UDP socket. [5]
2. Server - Handle HOWZIT and CHINA. [10]
3. Server - Handle TUNE and SHARP. [10]
4. Server - Handle EISH. [5]
5. Client - Create client UDP socket. [5]
6. Client - Handle HOWZIT and CHINA. [10]
7. Client - Handle TUNE and SHARP. [10]
8. Client - Handle EISH. [5]
9. Client - Handle EKSE. [10]
10. Graphical User Interface [10]
11. Compilation and Correct execution. [20]