Exercise01: Server/Client+Threads

Objectives:

- To learn to use Threads
- To learn to write server client code

Work with your group (or by yourself). Each group should upload only one submission.

First, open blackboard, go to Course Contents, and then download exercise01.zip file into your workspace (U:\workspace or something like that!). Then, unzip.

1 PLAYING WITH Threads & Server/Client

1.1 Threads

Play with the programs in "threadExamples".

There are four examples.

1. The first one shows how to create Threads by extending the Thread class.
2. The second example shows an alternative using implementations of the Runnable interface.
3. The third example shows problems when sharing modifiable data between threads.
4. The fourth example shows how such problems can be handled in Java by using the synchronized keyword that makes a thread that is currently doing the method to complete it before letting other methods start the method.
5. There is a lot more to threading – but that is outside the scope of this class. This info should be enough to get you started.

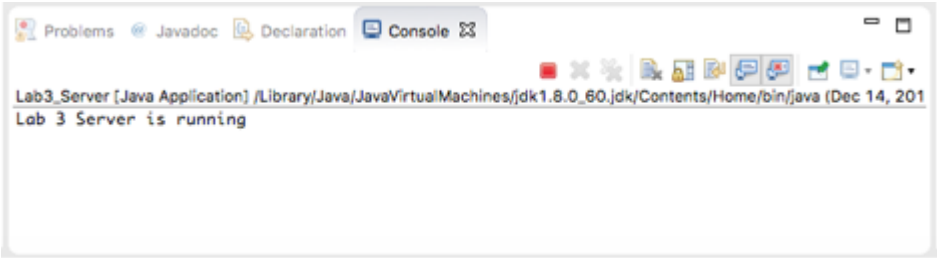
1.2 Server Client

Read serverClient.pptx file which located in the main root.

Play with the programs in "serverClientExamples".

There are two programs.

1. “MyServer.java” shows code for a sample Server program. Run it on Eclipse. If you run it multiple times, the later runs will all fail because port 4444 is already being reserved by the first run.
2. “MyClient.java” shows code for a sample client program. Run it several times. Each time a client program is run, it sends data to the server - which prints out the information on the screen.
3. Don’t forget to kill the server program after you are done playing with the code. For killing the server you need to press the red square in the Eclipse.



2 Part 2: Server Client/Thread

Create a chat application using Server Client.

2.1 Connect to Server

a) When you start the client, it should come up with a prompt (NOTE: you can either make it entirely text based – or you can use the provided GUI template in exercise01.zip to create a GUI frontend).

Enter you Name: (Type in your name, then press Enter)

b) After entering name, client should be connected to server.

c) Now, client will be shown a menu:

1. Send a text message to the server

2. Send an image file to the server

What happens next depends on which of the menu items is selected.

2.2 Send text message to server

The text message entered by user must be encrypted

Method of Encryption:

For every byte B_n in a String,
encrypted Byte = B_n XOR 11110000
e.g. B_n = 10101000, then B_n Xor 11110000 = 01011000

Method of Decryption

For every byte B_n in the received string
Let B_n = B_n Xor 11110000
e.g. B_n = 01011000, then B_n Xor 11110000= 10101000

2.3 Send image file to server

Method of encryption:

Suppose a image is stored by m bytes
For every 3 bytes, we have 24 bits.
Divide 24 into four 6 bits fragment
Add two bits of 00 to the right of each 6 bits fragment.(Then we can get four 8 bits fragments)
Convert the 8 bits fragment into charactors and combine them to make a string

Method of Decryption

For every charactor C_n received in image m
Convert C_n to a its binary representation;
Delete the two zeros on the leftmost side
Combine all the fragment into a string
Parse the string for every eight digits to make a image.

2.4 Send the image data using java's object stream.

a. https://docs.oracle.com/javase/tutorial/essential/io/objectstreams.html
b. https://docs.oracle.com/javase/7/docs/api/java/io/ObjectOutputStream.html

2.5 The server should keep history of chat in "chat.txt".

2.6 Admin facilities

When you type in your name as "admin", you should provide following menu items.

1. Broadcast message to all clients

2. List messages so far (from chat.txt)

3. Delete a selected message (from chat.txt) – give a message number.

HINT-1: In the GUI implementation, to have the CLIENT not "hang" waiting for the server to send message, on the client side code you will need to create a "server handling" thread whose sole purpose is to listen for messages from the server.

3 Submission:

Zip your Eclipse project and submit on black board. Remember there is only one submission per group. Make sure to include all the files that are needed in order to run your program(s).