# Assignment Module 3 Knock Knock Application

*By Michael Floerchinger*

08/27/2017

Java IV Programming

Assignment #3

Assignment Description.

Using the multi-threaded version of the Knock Knock example as the basis for this assignment:

Create a simple Swing GUI to start and stop the clients and server and display the jokes. All of the interaction does not have to occur in this one GUI. You may open other simple GUIs to replicate the behavior of the client cmd window. I expect you will need one application GUI that starts and stops the server. It should not be possible to start a client unless the server has been started. When the server is stopped, it is not necessary or even desirable to try to stop the clients. The application UI should be capable of starting multiple clients and displaying them simultaneously.

Handle the exceptions thrown by the stream and socket classes.

Read the jokes from a resource such as a file

Successive clients should get different initial knock-knock jokes. You do not have to guarantee the difference, just the randomness of the initial joke. Consider a load time solution that populates the clues and answers arrays randomly. Also consider a run-time

solution that has a random start location and operates on the lists as circular lists.

Add appropriate Javadoc comments to the source code.

Make any other changes you think should be made to improve the "OO-ness" of the design (at least one). The example code cries out for moving code out of the main methods and into appropriate class methods. Please explain what changes you made in the comments while submitting your assignment.

# Requirements and Deliverables:

## **Images of Application Execution and Functionality**

### **1. Create a simple Swing GUI to start/stop the clients and to display the jokes**

The KnockKnock main application launches with a single central location for starting and stopping the application components consisting of:

* The server service listening on a TCP port
* The client application connecting to the server's TCP port

Main Knock Knock Application – start server and client applications

Server service – start and stop

Client Application – start multiple clients

Client cannot be started without server running

2. It should not be possible to start a client unless the server is running

3. "When the server is stopped, it is not necessary or even desirable to try to stop the clients." NOT DONE

4. handling the exceptions thrown by the stream and socket classes

5. read jokes from are resource file

6. successive clients get different jokes

7. consider a load time solution that populates the clues and answers arrays randomly

8. consider a run-time solutions that has a random start location and operates on the lists as circular lists => ring buffer

9. add Javadoc comments to the source code

10. make other changes to improve the "OO-ness" of the app

### **2. Create a simple Swing GUI to sta**

### **1. Create a simple Swing**

Assignment 3 demonstrates the replacement and implementation of a database layer. This aims to replace only the previously

OO-ness suggestions

create extra methods

empty out the main

add interfaces / client server interfaces

server class, and have one extending the class

create an abstract server, that defines the header

Requirements:

1. create a simple Swing GUI to start/stop the clients and server and display the jokes.

- Can be multiple GUI's,

- should be capable of starting many clients

2. It should not be possible to start a client unless the server is running

3. "When the server is stopped, it is not necessary or even desirable to try to stop the clients." Should stopping also firstly stop the clients?

4. Handle the exceptions thrown by the classes:

- stream

- sockets

5. read jokes from a resource such as a file

6. successive clients should get different initial jokes, do not need to guarantee complete difference, just randomness of initial joke

7. consider a load time solution that populates the clues and answers arrays randomly

8. consider a run-time solutions that has a random start location and operates on the lists as circular lists => ring buffer

9. add Javadoc comments to the source code

10. make other changes to improve the "OO-ness" of the app

Research Notes:

http://cs.lmu.edu/~ray/notes/javanetexamples/

Client:

http://pirate.shu.edu/~wachsmut/Teaching/CSAS3211/2010-03/KnockClient.java

So Far

1. server start => Done

2. client connect => Done

- need multiple clients => Done

- should not start clients without server running => Done

- client notification => Done

3. automatically stop clients when shutting down the server => not done

4. I think so

5. read jokes from a file => DONE

6. startup with different jokes => DONE

7. Randomize array using Collections.shuffle(Arrays.asList(a)); => DONE

8. Ring buffer => https://stackoverflow.com/questions/590069/how-would-you-code-an-efficient-circular-buffer-in-java-or-c-sharp

better\*\*\*\*http://www.museful.net/2012/software-development/circulararraylist-for-java

Initialization vector for randomization as seed

start clients with rpc's remote procedure calls

reading for shutdown error:

http://www.dreamincode.net/forums/topic/259777-a-simple-chat-program-with-clientserver-gui-optional/

https://edn.embarcadero.com/article/31995

TODO: server needs disable client stop/start when it changes state

Replace the “Model” component of your application with a new model. The new model will make use of JDBC to connect to an

underlying relational database.