Vivek Patel

CSC435

JokeServer

Header:

/\*--------------------------------------------------------

1. Vivek Patel / April 23, 2019:

2. Java build 1.8.0\_201

3. Precise command-line compilation examples / instructions:

> javac JokeServer.java

> javac JokeClient.java

> javac JokeClientAdmin.java

or

> javac \*.java

works too

4. Precise examples / instructions to run this program:

In separate shell windows:

> java JokeServer

> java JokeClient

> java JokeClientAdmin

All acceptable commands are displayed on the various consoles.

This runs across machines, in which case you have to pass the IP address of

the server to the clients. For exmaple, if the server is running at

140.192.1.22 then you would type:

> java JokeClient 140.192.1.22

> java JokeClientAdmin 140.192.1.22

5. List of files needed for running the program.

a. JokeServer.java

b. JokeClient.java

c. JokeClientAdmin.java

5. Notes:

AdminLooper and implementation in main() method comes from Prof. Elliot’s code:

http://condor.depaul.edu/elliott/435/hw/programs/joke/joke-threads.html

Joke Client Admin can shut down the server, but if you want to start it again, you

can't without re-running the program. Also, the Admin controls are posted in the console, but

it does not switch states with the <enter> button alone.

The state is stored on the Client using 2 separate Lists. It is very sloppy, but it worked for me.

The Client makes a list of integers, which is then copied to 2 separate lists: one for jokes and

one for proverbs. The state lists are then shuffled. We take the first of both lists and send it to

the Server. The Server takes these indexes and picks the one corresponding to the mode and sends it

to the appropriate output method. Then the joke/proverb that has the same index as the Client's is

sent along with the index and a large number back to the client. The Client then processes this by

printing out the joke/proverb, removing the index from the temp list that it came from (which is

determined by comparing the index and the large number) and then checking if the temp list is empty.

If the temp list is empty, then the appropriate cycle completion is annouced and the temp list is

repopulated and shuffled using the master list.

Jokes/Proverbs credits:

JA: https://www.reddit.com/r/Jokes/comments/7he12e/i\_hope\_elon\_musk\_never\_gets\_involved\_in\_a\_scandal/

JB: https://www.reddit.com/r/Jokes/comments/7v0je6/my\_girlfriend\_is\_like\_the\_square\_root\_of\_100/

JC: https://www.reddit.com/r/Jokes/comments/6zukvv/i\_invited\_my\_girlfriend\_to\_go\_to\_the\_gym\_with\_me/

JD: https://www.reddit.com/r/Jokes/comments/basb7p/as\_i\_handed\_my\_dad\_his\_50th\_birthday\_card\_he/

PA: http://www.goldenproverbs.com/au\_confucius.html

PB: http://www.goldenproverbs.com/au\_plato.html

PC: http://www.goldenproverbs.com/au\_gandhi.html

PD: http://www.goldenproverbs.com/au\_verne.html

----------------------------------------------------------\*/

JokeServer.java

import java.io.\*;

import java.net.\*;

import java.util.\*;

// Communicates with the client, recieving 2 index numbers and a name,and sends either a joke or proverb,

//depending on the mode set by the Admin, joke by default.

class ClientWorker extends Thread {

Socket sock;

ClientWorker (Socket s) {sock = s;}

public void run() { //the main code for conversation with a client.

PrintStream out = null;

BufferedReader in = null;

//initialize joke and proverb lists

ArrayList<String> jokes = new ArrayList<String>(4);

ArrayList<String> proverbs = new ArrayList<String>(4);

try { // tries to create an input reader and an output reader

in = new BufferedReader(new InputStreamReader (sock.getInputStream()));

out = new PrintStream(sock.getOutputStream());

try { // tries to read the input from the client, process it, and send the output to the client

// Initializes variables needed by the ClientWorker to process the input and prepare for output

// indexJ and indexP refer to the saved state stored on the client.

String tempID;

String name;

String mode;

tempID = in.readLine(); // the input from the client, containing a string of a name and 2 index values

String id[] = tempID.split(", "); // splits the input into a String array so the input can be set to the appropriate variables

name = id[0]; // this is the name the user entered into the Client

//Creates temporary strings from the input that are then converted to integers to be used as index

//values to reference the joke and proverb lists

String inJTemp = id[1];

int indexJ = Integer.parseInt(inJTemp);

String inPTemp = id[2];

int indexP = Integer.parseInt(inPTemp);

// Creates the lists for the jokes and proverbs. They are in 2 separate lists.

jokeBuilder(name, jokes);

proverbBuilder(name, proverbs);

// checks the current mode and then sends the corresponding list, index number, and

//PrintStream to create and send the output joke/proverb. Prints on the server as to

//what the server is sending and to whom.

mode = AdminWorker.mode;

if (mode == "joke") {

System.out.println("Sending joke to " + name);

sendJoke(jokes, indexJ, out);

}

else {

System.out.println("Sending proverb to " + name);

sendProverb(proverbs, indexP, out);

}

} catch (IOException x) { // catches an IO exception and prints an error message

System.out.println("Server read error");

x.printStackTrace();

}

sock.close(); // closes the connection to this specific client

} catch (IOException ioe) {System.out.println(ioe);} // catches an IO exception and prints an error message

}

// the list builders for each Jokes and Proverbs. Here we take the name from the client and add it to the joke output.

static void jokeBuilder(String name, ArrayList<String> jokes){

jokes.add("JA " + name + ": I hope Elon Musk never gets involved in a scandal. Elongate would be really drawn out.");

jokes.add("JB " + name + ": My girlfriend is like the square root of -100. A solid 10 but also imaginary.");

jokes.add("JC " + name + ": I invited my girlfriend to go to the gym with me and then I didn\'t show. I hope she gets the message that we\'re not working out.");

jokes.add("JD " + name + ": As I handed my dad his 50th birthday card he looked at me with tears in his eyes and said\" Y'know one would have been enough.\"");

}

static void proverbBuilder(String name, ArrayList<String> proverbs){

proverbs.add("PA " + name + ": He who learns but does not think is lost! He who thinks but does not learn is in great danger. -Confucius");

proverbs.add("PB " + name + ": Wise men speak because they have something to say; Fools because they have to say something. -Plato");

proverbs.add("PC " + name + ": An eye for an eye only ends up making the whole world blind. -Mahatma Gandhi");

proverbs.add("PD " + name + ": Science my lad is made up of mistakes but they are mistakes which it is useful to make because they lead little by little to the truth. -Jules Verne");

}

// the calls to send the joke or proverb to the Client.

static void sendJoke(ArrayList<String> list, int jDex, PrintStream out) {

String jp = list.get(jDex);

//Client input is set as (joke/proverb, joke index used, proverb index used) where the joke is printed and the indexs are

// compared to eachother to determine which index was used. The number 5 is used because it is larger than the max index number.

// The client will see this as the smaller number was used, so we should delete it from the corresponding index list.

out.println(jp + ", " + jDex + ", " + 5);

out.flush();

}

static void sendProverb(ArrayList<String> list, int pDex, PrintStream out) {

String jp = list.get(pDex);

out.println(jp + ", " + 5 + ", " + pDex);

out.flush();

}

}

// Communicates with the Admin Client to set the mode or shut the server down. Input from

// the admin is a single character. Admin controls are posted in the Admin Client console.

class AdminWorker extends Thread {

Socket sock;

AdminWorker (Socket s) {sock = s;}

public static String mode = "joke";

public void run() { // runs the Admin server functions (mode swap or shut down)

PrintStream out = null;

BufferedReader in = null;

try { // tries to create an input reader and an output reader

in = new BufferedReader(new InputStreamReader (sock.getInputStream()));

out = new PrintStream(sock.getOutputStream());

try { // takes the input from the Admin Client, processes the command, and will do 1 of 3 things

// 1) switch the mode to Joke and print a message confirming the switch, 2) same as 1), but with Proverb, 3) Shut the server down

// 4) do nothing and post a message on the Server.

String modeInput;

modeInput = in.readLine(); // takes the input

switch (modeInput) {

case "j":

modeSwitch("j");

break;

case "p":

modeSwitch("p");

break;

case "s":

System.out.println("Shutting down due to Admin command.");

System.exit(1);

break;

default:

System.out.println("Invalid command.");

break;

}

} catch (IOException x) { // catches an IO exception and prints an error message

System.out.println("Server read error");

x.printStackTrace();

}

sock.close(); // closes the connection to this specific client

} catch (IOException ioe) {System.out.println(ioe);} // catches an IO exception and prints an error message

}

// Switches the mode based on the input from the Admin Client

static void modeSwitch(String m){

if (m == "j"){

mode = "joke";

System.out.println("Currently telling jokes.");

}

else if ( m == "p"){

mode = "proverb";

System.out.println("Currently telling proverbs.");

}

}

}

// Creates the loop that connects to the Admin Client. The server will wait and listen for the

// Admin to send a command. Neither Admin Client nor JokeServer is needed to be active when starting

// the other.

class AdminLooper implements Runnable {

public static boolean adminControlSwitch = true;

public void run(){

System.out.println("In the admin looper thread");

int q\_len = 6;

int port = 5050; // the Admin Client's port, different fromt the Client port

Socket sock;

try{

ServerSocket servsock = new ServerSocket(port, q\_len);

while (adminControlSwitch) {

sock = servsock.accept();

new AdminWorker (sock).start();

}

}catch (IOException ioe) {System.out.println(ioe);}

}

}

public class JokeServer{

//Starts the server and deploys workers as needed when each Client connects

public static void main(String[] args) throws IOException {

int q\_len = 6;

int port = 4545; // the port connection the server is listening to for Clients

Socket sock;

// Initializes the loop for the Admin Client to connect to

AdminLooper AL = new AdminLooper();

Thread t = new Thread(AL);

t.start();

ServerSocket servsock = new ServerSocket(port, q\_len); //initializes the server socket and port

System.out.println ("Vivek Patel's JokeServer 1.0 starting up, listening at port 4545 (client) and 5050 (admin).\n");

while (true) { // listens for a client to connect

sock = servsock.accept(); // accepts the client

new ClientWorker(sock).start(); // creates a worker for the client

}

}

}

JokeClient.java:

import java.io.\*;

import java.net.\*;

import java.lang.\*;

import java.util.\*;

public class JokeClient {

public static void main(String[] args) { // runs the Client and connects it to the Server

String serverName;

if (args.length < 1) serverName = "localhost"; //default server

else serverName = args[0]; // directed server input from the terminal

// creates the list of integers that will be used as a master copy for the

// index lists that represent the server state.

// 0-3 was used because that is all the integers needed for the index

List<Integer> list = new ArrayList<>();

list.add(0);

list.add(1);

list.add(2);

list.add(3);

System.out.println("Vivek Patel's JokeClient, 1.0.\n");

System.out.println("Using server: " + serverName + ", Port: 4545");

BufferedReader in = new BufferedReader(new InputStreamReader(System.in)); // creates inputstream for the Client

try {

//initializes the variables necessary for the Client.

String name; // name provided to the Server by the user

String input; // the input stream from the user

// the creaton of the index lists for the Jokes and Proverbs.

// these index lists save the state of the conversation to the Server

// but are not stored in any permanent file. Both index lists are

// shuffled as to create the random order for the jokes and proverb output

List<Integer> indexListJ = new ArrayList<>(list);

Collections.shuffle(indexListJ);

List<Integer> indexListP = new ArrayList<>(list);

Collections.shuffle(indexListP);

System.out.print("What is your name?: "); // Asks the user for their name

System.out.flush(); // writes the data from the input stream from the user

name = in.readLine(); // saves the user's name

do {

// the user is prompted to press enter to recieve a joke or proverb. They can input "quit" to close the client.

// most random inputs are ignored. The only issue is if there "quit" anywhere in the input

System.out.print("Press <enter> for a joke or proverb or input (quit) to end: ");

System.out.flush(); //writes the data from the input stream from the user

input = in.readLine();

if (input.indexOf("quit") < 0) // if "quit" isn't inputted, collect the necessary variables and call getJP

//to prepare the data to be sent to the server

getJP(name, indexListJ, indexListP, list, serverName);

} while (input.indexOf("quit") < 0); //if "quit" is anywhere within the input, close the client, otherwise keep looping the above code.

System.out.println("Cancelled by user request."); // This message is printed when the user "quit"s the client

} catch (IOException x) {x.printStackTrace();}

}

// the method that takes the user's input and internal state variables and sends it to the server.

//inputs are the name the user inputted, the index lists for both jokes and proverbs, the master list, and the output.

private static void getJP(String name, List<Integer> iLJ, List<Integer> iLP, List<Integer> list, String serverName) {

Socket sock;

BufferedReader fromServer;

PrintStream toServer;

String dataFromServer;

String message; // this will house the joke or proverb sent from the server

int jDex; // jDex and pDex are the index values sent from the server back to the client.

int pDex; // They represent joke and proverb, respectively

try {

sock = new Socket(serverName, 4545); // connects to the server at port 4545

fromServer = new BufferedReader(new InputStreamReader(sock.getInputStream())); //stream recieves input from the server

toServer = new PrintStream(sock.getOutputStream()); //stream sends output to the server

//These create string versions of the index number, the first of each shuffled index list.

// this way, both jokes and proverbs will have their own randomized order

String indexJSend = iLJ.get(0).toString();

String indexPSend = iLP.get(0).toString();

// sends the data and makes sure of it. Packages the user's name and the 2 indexes for the joke or proverb

toServer.println(name + ", " + indexJSend + ", " + indexPSend); toServer.flush();

// This is the package from the server. Included is that message, the used index number that represents what

// type of message was recieved, and a dummy index number. Here you can't tell what is the real index number is

// or which list it's refering to.

// The message is printed to the Client's console.

dataFromServer = fromServer.readLine();

String[] text = dataFromServer.split(", ");

message = text[0];

System.out.println(message);

String jTemp = text[1];

jDex = Integer.parseInt(jTemp);

String pTemp = text[2];

pDex = Integer.parseInt(pTemp);

// these if statements will compare the 2 returned indexes. One of these will be a fake index of 5.

// the smaller index will reveal which type of message was printed (joke or proverb), and will then remove

// the index from the parent index list.

// if after the removal of the index from the index list, the list is empty, the Client will print a message to

// the console saying that all of that type of message has been seen. The index list is repopulated from the master

// list and then reshuffled to be reused. This way we can not only have our randomized order of jokes/proverbs, but there will be no repeats either.

if (jDex < pDex) {

iLJ.remove(0);

if (iLJ.size() == 0){

System.out.println("JOKE CYCLE COMPLETE");

iLJ.addAll(list);

Collections.shuffle(iLJ);

}

}

else {

iLP.remove(0);

if (iLP.size() == 0){

System.out.println("PROVERB CYCLE COMPLETE");

iLP.addAll(list);

Collections.shuffle(iLP);

}

}

sock.close(); // closes the connection to the server

} catch (IOException x) { // catches an IO exception and prints it

System.out.println("Socket error.");

x.printStackTrace();

}

}

}

JokeClientAdmin.java:

import java.io.\*;

import java.net.\*;

public class JokeClientAdmin {

public static void main(String[] args) { // runs the Admin Client

String serverName;

if (args.length < 1) serverName = "localhost"; //default server

else serverName = args[0]; // directed server input from the terminal

System.out.println("Vivek Patel's JokeClientAdmin, 1.0.\n");

System.out.println("Using server: " + serverName + ", Port: 5050");

BufferedReader in = new BufferedReader(new InputStreamReader(System.in)); // creates inputstream for the Admin

try {

String mode; // creates the mode variable for the Admin. This keeps the input from the Admin.

do {

// The Admin Client's console will print the list of commands for the Admin. It can change the mode based on the input,

// shut down the server, or exit the Admin Client

System.out.print("Input (j) for joke or (p) for proverb, (quit) to quit the Admin Client, or (s) to shut down the server: ");

System.out.flush(); //writes the data from the input stream from the user

mode = in.readLine(); // saves the Admin's input

if (mode.indexOf("quit") < 0) // if "quit" isn't inputted, send the Admin's command to the server

adminControl(mode, serverName);

} while (mode.indexOf("quit") < 0); //if "quit" is anywhere within the input, close the client.

System.out.println("Cancelled by user request."); // printed when the Admin closes the client.

} catch (IOException x) {x.printStackTrace();}

}

static void adminControl(String mode, String serverName) { //contact's the server with the Admin's command

Socket sock;

BufferedReader fromServer;

PrintStream toServer;

String textFromServer;

try {

sock = new Socket(serverName, 5050); // connects to the server, uses a different port than regular Clients

fromServer = new BufferedReader(new InputStreamReader(sock.getInputStream())); //stream recieves input from the server

toServer = new PrintStream(sock.getOutputStream()); //stream sends output to the server

toServer.println(mode); toServer.flush(); // sends the data and makes sure of it

sock.close(); // closes the connection to the server

} catch (IOException x) { // catches an IO exception and prints it

System.out.println("Socket error.");

x.printStackTrace();

}

}

}

JokeLog.txt

/\*

I show here that I run 8 jokes in a row, then 8 proverbs, then splitting 2/2/2/2 proverbs/jokes/proverbs/jokes. This shows 3 things:

1) jokes and proverbs are independently randomized

2) jokes and proverb sstates are independently stored

3) Admin is able to switch the mode and the server can send the current mode's message

I also have a shut-down command in the Admin client which closes the server. There is also a quit function on both Client and Admin client, but that is borrowed from InetServer code.

\*/

JokeServer:

In the admin looper thread

Vivek Patel's JokeServer 1.0 starting up, listening at port 4545 (client) and 5050 (admin).

Sending joke to Vivek

Sending joke to Vivek

Sending joke to Vivek

Sending joke to Vivek

Sending joke to Vivek

Sending joke to Vivek

Sending joke to Vivek

Sending joke to Vivek

Currently telling proverbs.

Sending proverb to Vivek

Sending proverb to Vivek

Sending proverb to Vivek

Sending proverb to Vivek

Sending proverb to Vivek

Sending proverb to Vivek

Sending proverb to Vivek

Sending proverb to Vivek

Sending proverb to Vivek

Sending proverb to Vivek

Currently telling jokes.

Sending joke to Vivek

Sending joke to Vivek

Currently telling proverbs.

Sending proverb to Vivek

Sending proverb to Vivek

Currently telling jokes.

Sending joke to Vivek

Sending joke to Vivek

Shutting down due to Admin command.

JokeClient:

Vivek Patel's JokeClient, 1.0.

Using server: localhost, Port: 4545

What is your name?: Vivek

Press <enter> for a joke or proverb or input (quit) to end:

JC Vivek: I invited my girlfriend to go to the gym with me and then I didn't show. I hope she gets the message that we're not working out.

Press <enter> for a joke or proverb or input (quit) to end:

JD Vivek: As I handed my dad his 50th birthday card he looked at me with tears in his eyes and said" Y'know one would have been enough."

Press <enter> for a joke or proverb or input (quit) to end:

JA Vivek: I hope Elon Musk never gets involved in a scandal. Elongate would be really drawn out.

Press <enter> for a joke or proverb or input (quit) to end:

JB Vivek: My girlfriend is like the square root of -100. A solid 10 but also imaginary.

JOKE CYCLE COMPLETE

Press <enter> for a joke or proverb or input (quit) to end:

JB Vivek: My girlfriend is like the square root of -100. A solid 10 but also imaginary.

Press <enter> for a joke or proverb or input (quit) to end:

JD Vivek: As I handed my dad his 50th birthday card he looked at me with tears in his eyes and said" Y'know one would have been enough."

Press <enter> for a joke or proverb or input (quit) to end:

JC Vivek: I invited my girlfriend to go to the gym with me and then I didn't show. I hope she gets the message that we're not working out.

Press <enter> for a joke or proverb or input (quit) to end:

JA Vivek: I hope Elon Musk never gets involved in a scandal. Elongate would be really drawn out.

JOKE CYCLE COMPLETE

Press <enter> for a joke or proverb or input (quit) to end:

PB Vivek: Wise men speak because they have something to say; Fools because they have to say something. -Plato

Press <enter> for a joke or proverb or input (quit) to end:

PA Vivek: He who learns but does not think is lost! He who thinks but does not learn is in great danger. -Confucius

Press <enter> for a joke or proverb or input (quit) to end:

PD Vivek: Science my lad is made up of mistakes but they are mistakes which it is useful to make because they lead little by little to the truth. -Jules Verne

Press <enter> for a joke or proverb or input (quit) to end:

PC Vivek: An eye for an eye only ends up making the whole world blind. -Mahatma Gandhi

PROVERB CYCLE COMPLETE

Press <enter> for a joke or proverb or input (quit) to end:

PB Vivek: Wise men speak because they have something to say; Fools because they have to say something. -Plato

Press <enter> for a joke or proverb or input (quit) to end:

PD Vivek: Science my lad is made up of mistakes but they are mistakes which it is useful to make because they lead little by little to the truth. -Jules Verne

Press <enter> for a joke or proverb or input (quit) to end:

PC Vivek: An eye for an eye only ends up making the whole world blind. -Mahatma Gandhi

Press <enter> for a joke or proverb or input (quit) to end:

PA Vivek: He who learns but does not think is lost! He who thinks but does not learn is in great danger. -Confucius

PROVERB CYCLE COMPLETE

Press <enter> for a joke or proverb or input (quit) to end:

PB Vivek: Wise men speak because they have something to say; Fools because they have to say something. -Plato

Press <enter> for a joke or proverb or input (quit) to end:

PD Vivek: Science my lad is made up of mistakes but they are mistakes which it is useful to make because they lead little by little to the truth. -Jules Verne

Press <enter> for a joke or proverb or input (quit) to end:

JC Vivek: I invited my girlfriend to go to the gym with me and then I didn't show. I hope she gets the message that we're not working out.

Press <enter> for a joke or proverb or input (quit) to end:

JA Vivek: I hope Elon Musk never gets involved in a scandal. Elongate would be really drawn out.

Press <enter> for a joke or proverb or input (quit) to end:

PA Vivek: He who learns but does not think is lost! He who thinks but does not learn is in great danger. -Confucius

Press <enter> for a joke or proverb or input (quit) to end:

PC Vivek: An eye for an eye only ends up making the whole world blind. -Mahatma Gandhi

PROVERB CYCLE COMPLETE

Press <enter> for a joke or proverb or input (quit) to end:

JB Vivek: My girlfriend is like the square root of -100. A solid 10 but also imaginary.

Press <enter> for a joke or proverb or input (quit) to end:

JD Vivek: As I handed my dad his 50th birthday card he looked at me with tears in his eyes and said" Y'know one would have been enough."

JOKE CYCLE COMPLETE

Press <enter> for a joke or proverb or input (quit) to end: quit

Cancelled by user request.

/\*

Output simplified:

8 jokes in a row: C, D, A, B [COMPLETE], B, D, C, A [COMPLETE]

8 proverbs in a row: B, A, D, C [COMPLETE], B, D, C, A [COMPLETE]

2 proverbs, 2 jokes twice (both cycles complete): PB, PD, JC, JA, PA, PC [COMPLETE], JB, JD [COMPLETE]

\*/

JokeClientAdmin:

Vivek Patel's JokeClientAdmin, 1.0.

Using server: localhost, Port: 5050

Input (j) for joke or (p) for proverb, (quit) to quit the Admin Client, or (s) to shut down the server: p

Input (j) for joke or (p) for proverb, (quit) to quit the Admin Client, or (s) to shut down the server: j

Input (j) for joke or (p) for proverb, (quit) to quit the Admin Client, or (s) to shut down the server: p

Input (j) for joke or (p) for proverb, (quit) to quit the Admin Client, or (s) to shut down the server: j

Input (j) for joke or (p) for proverb, (quit) to quit the Admin Client, or (s) to shut down the server: s

Input (j) for joke or (p) for proverb, (quit) to quit the Admin Client, or (s) to shut down the server: quit

Cancelled by user request.