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

1. Name: Kaijun He / Date: 01/20/2018

2. Java version used, if not the official version for the class:

java version "1.8.0\_144"

3. Precise command-line compilation examples / instructions:

> javac JokeClient.java

> javac JokeServer.java

> javac JokeClientAdmin.java

4. Precise examples / instructions to run this program:

In separate shell windows:

> java JokeClient

> java JokeServer

> java JokeClientAdmin

5. List of files needed for running the program.

a. JokeClient.java

b. JokeServer.java

c. JokeClientAdmin.java

5. Notes:

i have some problem in jokeclientAdmin, if you want quit the client, press q when u

see Press enter to see state of server or q to quit: and afte pop out another message just press enter

then you could leave. for the jokeserver file, if you want to quit the file, you have to use control c to

kill it by yourself.

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

import java.io.\*;

import java.net.\*;

import java.util.\*;

public class JokeClient {

// define joke index, proverb index to represent which joke or proverb will be printout right now

private static int userID = 0;

private static int jokeIndex = 0;

private static int proverbIndex = 0;

private static int port = 4545;

private static int jokeServerMode;

public static String username;

public static void main(String[] args) {

String serverName;

String username;

// try to check the args input from command line, if no arguement recieved we suppose it's localhost

if (args.length == 0) {

serverName = "localhost";

} else {

serverName = args[0];

}

// use scanner to get input from keyboard type in

Scanner input = new Scanner(System.in);

System.out.println("Please type in your username: ");

username = input.nextLine(); //read input from keyboard for username,

System.out.println("\n This Is " + username + "'s Joke Client\n");

System.out.println("Server: " + serverName + ", Port: " + port); //printout the server name and port number for this client

BufferedReader in = new BufferedReader(new InputStreamReader(System.in)); // using buffer to read system input

try { // try and catch to throw exceptions

String name;

do { // do while loop to consider the quit condition

System.out.println("Press enter of keyboard to print out jokes and proverbs or type q to quit : "); //print out to client window

System.out.flush();

name = in.readLine();

if (name.indexOf("q") < 0) { // when readed text is not q, then call getRemotedAddress function

getRemoteAddress(username, serverName);

}

} while (name.indexOf("q") < 0);

System.out.println("user cancelled the request by himself."); //print out text to indicate that user canceled the operation in client

}

catch (IOException e) { //throw an exception

e.printStackTrace(); //print stacktrace

}

}

static void getRemoteAddress (String username, String serverName) {

Socket socket;

BufferedReader fromServer;

PrintStream toServer;

try {

//create a new type socket object socket

socket = new Socket(serverName, port);

//Create filter input and outpu streams for the socket

fromServer = new BufferedReader(new InputStreamReader(socket.getInputStream())); //get input stream from server by calling getInputstream()

toServer = new PrintStream(socket.getOutputStream()); //sending out output form stream to server by calling getOutputStream()

int maintain = fromServer.read(); // set a integer flag for maintanance mode

if(maintain > 0){ // if it's not in maintainance mode, then

toServer.println(username); // send username to server

toServer.flush();

toServer.write(userID); //write userid for my user in client

userID = fromServer.read(); //wait for server give back userId

toServer.write(jokeIndex); // send the index of jokes and proverb to system

toServer.write(proverbIndex);

System.out.println(fromServer.readLine()); // pritnout what recieved from server

//get jokes and proverb index from server side

jokeIndex = fromServer.read();

proverbIndex = fromServer.read();

//get joke server mode

jokeServerMode = fromServer.read();

}

else{// when in maintannace mode, printout system message to notify that what joke server mode is

System.out.println("system is in maintanance mode");

}

// joke index is from 0 to 3 for 4 jokes, so when joke index go back to 0 again, printout we finish a joke cycle

if (jokeIndex == 0 && jokeServerMode == 1) {

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

}

// proverb index is from 0 to 3 for 4 proverbs, so when proverb index go back to 0 again, printout we finish a proverb cycle

if (proverbIndex == 0 && jokeServerMode == 2) {

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

}

socket.close(); //close socket connection

}

catch (IOException x) { //throw out input and output exception and printout socket error message

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

x.printStackTrace();

}

}

}

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

1. Name: Kaijun He / Date: 01/20/2018

2. Java version used, if not the official version for the class:

java version "1.8.0\_144"

3. Precise command-line compilation examples / instructions:

> javac JokeClient.java

> javac JokeServer.java

> javac JokeClientAdmin.java

4. Precise examples / instructions to run this program:

In separate shell windows:

> java JokeClient

> java JokeServer

> java JokeClientAdmin

5. List of files needed for running the program.

a. JokeClient.java

b. JokeServer.java

c. JokeClientAdmin.java

5. Notes:

i have some problem in jokeclientAdmin, if you want quit the client, press q when u

see Press enter to see state of server or q to quit: and afte pop out another message just press enter

then you could leave. for the jokeserver file, if you want to quit the file, you have to use control c to

kill it by yourself.

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

import java.io.\*;

import java.net.\*;

import java.util.\*;

class Worker extends Thread { //define a class worker to creat thread with member socket

Socket socket;

Worker(Socket s) //consturctor assign s to socket

{

socket = s;

}

public void run() {

//intinial the input and output stream to null value

PrintStream out = null;

BufferedReader in = null;

try {

// sign value to in and out by calling getInputStream() and getOutputStream()

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

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

if(JokeServer.controlSwitch != true){ // set a boolean flag controlswitch to printout message when server is shutting down

System.out.println("server is now shutting down");

}

else{// condition when server is on

try {

String username;

String nextOutput;

int userID;

// check whether the jokeserver state is mantenance mode or not if yes ten write 0, if not write 1 to client

if(JokeServer.serverState == JokeServer.ServerState.MAINTENANCEMODE){

out.write(0);

}

else{

out.write(1);

}

// if it's maintenance mode, then start conversation between client and server

if(JokeServer.serverState != JokeServer.ServerState.MAINTENANCEMODE){

username = in.readLine(); //recieve username from Client

userID = in.read(); //get userID from Client

if (userID == 0) { //if userId is 0 then we increase id number by 1 by calling function incrementUserId() function and record down userId

userID = JokeServer.incrementUserId();

}

out.write(userID);

//get joke index from JokeClient

int jokeIndex = in.read();

//get proverb index index from JokeClient

int proverbIndex = in.read();

if (JokeServer.serverState == JokeServer.ServerState.JOKEMODE) { //if serverState is in JOKEMODE

// then replace the placeholder in jokes list with current username

out.println(JokeServer.Jokes[jokeIndex].replaceAll("Placeholder", username));

} else{ //if serverState is in PROVERBMODE, then replace the placeholder in proverbs list with current username

out.println(JokeServer.Proverbs[proverbIndex].replaceAll("Placeholder", username));

}

// printout the current mode is

System.out.println("mode is :" + JokeServer.serverState);

int mode = 0;

// create a new integer flag mode

//set jokemode to value 1 and proverb mode to value 2,

if(JokeServer.serverState == JokeServer.ServerState.JOKEMODE){

mode = 1;

}

else if (JokeServer.serverState == JokeServer.ServerState.PROVERBMODE){

mode = 2;

}

//update joke index and proverb index by calling nexjoke and nextProverb function.

int newJokeIndex = JokeServer.nextJoke(jokeIndex, mode);

int newProverbIndex = JokeServer.nextProverb(proverbIndex, mode);

// using shuffle method to ramdomize the order of jokes after one cycle

if (jokeIndex == JokeServer.Jokes.length - 1) {

List<String> jokesList = Arrays.asList(JokeServer.Jokes);

Collections.shuffle(jokesList);

JokeServer.Jokes = jokesList.toArray(new String[jokesList.size()]);

}

// using shuffle method to ramdomize the order of proverbs after one cycle

if (proverbIndex == JokeServer.Proverbs.length - 1) {

List<String> proverbsList = Arrays.asList(JokeServer.Proverbs);

Collections.shuffle(proverbsList);

JokeServer.Proverbs = proverbsList.toArray(new String[proverbsList.size()]);

}

//update joke index and proverb index in client

out.write(newJokeIndex);

out.write(newProverbIndex);

out.write(mode);

// if it's shutdown mode,we switch shutdown flag controlswitch to false

if(username.indexOf("shutdown") >= 0){

JokeServer.controlSwitch = false;

System.out.println("we are shutting down the server");

}

else {

// printout which user is sending request recently

System.out.println("we recieve request from user: " + username + " with an userID(" + userID + ").");

}

}

} catch (IOException e) { // throw exceptions and printout server error message

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

e.printStackTrace(); //print a stack trace exception

}

}

socket.close(); //close this socket connection

} catch (IOException e) {

System.out.println(e); //throw out input output exception.

}

}

}

public class JokeServer {

public enum ServerState { //define three server states jokemode, proverbmode, and maintenancemode

JOKEMODE, PROVERBMODE, MAINTENANCEMODE

}

public static boolean controlSwitch = true; // flag for shuting down condition

//4 samples jokes from online resources

public static String[] Jokes = {"JA Placeholder: Can a kangaroo jump higher than a house? Of course, a house doesn’t jump at all.",

"JB Placeholder: Anton, do you think I’m a bad mother? My name is Paul.",

"JC Placeholder: What is the difference between a snowman and a snowwoman? Snowballs",

"JD Placeholder: My dog used to chase people on a bike a lot. It got so bad, finally I had to take his bike away."};

// 4 popular proverbs from online resources

public static String[] Proverbs = {"PA Placeholder: Two wrongs don't make a right.",

"PB Placeholder: The pen is mightier than the sword.",

"PC Placeholder: When in Rome, do as the Romans.",

"PD Placeholder: The squeaky wheel gets the grease"};

public static ServerState serverState = ServerState.JOKEMODE; //serverstate default mode is jokemode

//define a nextuserId

private static int nextUserID = 0;

//increase userid to make sure each user has uique id number

public static synchronized int incrementUserId() {

return ++nextUserID;

}

//return next joke index by given mode, if mode changes then save index for current user

public static int nextJoke(int index, int mode) {

if(mode == 1){

if (index == JokeServer.Jokes.length - 1) //if jokeNumber is last joke index

return 0; //return 0 for jokeNumber

else

return index += 1; //else, add 1 to jokeNumber

}

else{

return index;

}

}

//return next proverb index by given mode, if mode changes then save proverb index for current user

public static int nextProverb(int index, int mode) {

if(mode == 2){

if (index == JokeServer.Proverbs.length - 1) //if proverbNumber is last proverb index

return 0; //return 0 for proverbNumber

else

return index += 1; //else, add 1 to proverbNumber

}

else{

return index;

}

}

//Change server state by admin client

public static void changeState(int state) { // define changeState method

System.out.println("Recieved state is: " + state);

if (state == 1) //state is 1 then set to jokemode

JokeServer.serverState = ServerState.JOKEMODE;

else if (state == 2) //if state is 2 then set to proverbmode

JokeServer.serverState = ServerState.PROVERBMODE;

else if(state == 3) // if state is 3 then set to maintenancemode

JokeServer.serverState = serverState.MAINTENANCEMODE;

System.out.println("Server state: " + JokeServer.serverState);

}

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

int q\_len = 6;

int port = 4545; //default port number

AdminLoop AL = new AdminLoop(); // create new threat for admin client

Thread t = new Thread(AL);

t.start(); //start threat t for admin client

Socket socket;

ServerSocket servsock = new ServerSocket(port, q\_len); //creat a new servsock using two parameter port number and length

System.out.println("kaijun's Joke Server starting up, listening at port number:" + port); //print out message for portnumber

while (controlSwitch) {

socket = servsock.accept(); //wait for the next client connection

if(controlSwitch){

new Worker(socket).start(); //start a new work thread

}

}

}

}

class AdminLoop implements Runnable {

public static boolean adminControlSwitch = true;

public void run() {

int q\_len = 6;

int port = 5050; //Listening at a different port number which is gievn 5050 for Admin client

Socket socket;

try {

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

while (adminControlSwitch) {

//wait for the the connection and accept

socket = servsock.accept();

new AdminWorker(socket).start(); //start a new admin worker thread

}

} catch (IOException e) { //throw input and output exceptions

System.out.println(e);

}

}

}

class AdminWorker extends Thread // create admin worker class

{

private Socket socket;

AdminWorker(Socket s)

{

socket = s;

}

public void run()

{

PrintStream out = null;

BufferedReader in = null;

try

{

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

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

//Note that this branch might not execute when expected

System.out.println("Joke Admin Client connected.");

//send serverstate to the server

out.println(JokeServer.serverState);

//read the number of the mode from the client

int modeNumber = in.read();

//call changestate() function to switch from defined three mode

JokeServer.changeState(modeNumber);

this.socket.close(); // close socket connection

}

catch(IOException x)

{

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

x.printStackTrace();

}

}

}

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

1. Name: Kaijun He / Date: 01/20/2018

2. Java version used, if not the official version for the class:

java version "1.8.0\_144"

3. Precise command-line compilation examples / instructions:

> javac JokeClient.java

> javac JokeServer.java

> javac JokeClientAdmin.java

4. Precise examples / instructions to run this program:

In separate shell windows:

> java JokeClient

> java JokeServer

> java JokeClientAdmin

5. List of files needed for running the program.

a. JokeClient.java

b. JokeServer.java

c. JokeClientAdmin.java

5. Notes:

i have some problem in jokeclientAdmin, if you want quit the client, press q when u

see Press enter to see state of server or q to quit: and afte pop out another message just press enter

then you could leave. for the jokeserver file, if you want to quit the file, you have to use control c to

kill it by yourself.

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

import java.io.\*;

import java.net.\*;

public class JokeClientAdmin

{

private final static int port = 5050; // use given port number from handout

public static void main(String[] args)

{

String serverName;

if(args.length == 0) //if there is no arguement, then suppose to use local host

serverName = "localhost";

else {// if not then use args[0] instead

serverName = args[0];

}

// printout admin client servername and port number

System.out.println("Joke Admin Client");

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

BufferedReader in = new BufferedReader(new InputStreamReader(System.in));

try

{

String name;

do

{

System.out.println("Press enter to see state of server or q to quit: ");

name = in.readLine(); //get what user type in, either check the mode or quit the client

Socket socket;

BufferedReader fromServer;

PrintStream toServer;

try

{

socket = new Socket(serverName, port); // create new socket for this connection

fromServer = new BufferedReader(new InputStreamReader(socket.getInputStream()));

toServer = new PrintStream(socket.getOutputStream()); //send output stream to server

BufferedReader input = new BufferedReader(new InputStreamReader(System.in)); // read input to bufferreader

String state = fromServer.readLine();

System.out.println( "Server mode: " + state); //printout and let user to decide three 3 mode

System.out.println("press 1 for JokeMODE, press 2 for proverbMode, press 3 for maintenanceMode");

int mode = input.read();

mode -= 48;

toServer.write(mode);

toServer.flush();

}

catch(IOException e) //throw input and output exception

{

System.out.println("Socket read error"); //print socket read error and exceptions

e.printStackTrace();

}

} while(name.indexOf("q") < 0); //if user type in q as quit to cancel request

System.out.println("user has cancelled the request by user himself"); //print message when quit

}

catch(IOException e)

{

e.printStackTrace();

}

}

}