**QUIZ UP**

**Introduction**

Our project is a multiplayer quiz game. Two players will be connected in a network. The game will put forth the same question to both the players and wait for them to reply. Once both have answered, the game will match it with the correct answer and give five points for the correct answer and zero points for the wrong answer. The winner will be declared after the end of question set.

**Abstract**

To implement this, there will be a server and concurrent clients running. Server will have a set of questions stored in it, each with four alternatives. One of the questions will be randomly selected and sent to both the clients. The server waits for the response from both the clients. It matches their response for the answer of the sent question, and accordingly points will be allotted to them.

This uses Socket Programming to implement the functionality of this project. We have used TCP socket servers for the same. For the client GUI, we have used JavaFX. The GUI will have the question displayed with a radio button next to each options. The Client will have to select an answer and submits it. She will be told if his answer is right or wrong and the scores will be updated accordingly.

**Working**

The quizup application consists of 3 components : Server,Client and GUI. The server program communicates to the clients using tcp/ip. It uses 2 ports 3000,3001

There are 2 instances of the same client program. One client connects on port 3000 and the second client connects on 3001.

The server program sends a question to he client programs and waits for their response. Once the responses from both the clients are received the server informs the client whether the response is correct or not and sends the next question. For each correct answer the client adds 5 to the score.

At the end of the quiz the server indicates which client application has won.

The GUI uses the client Tcp/ip to communicate with the server. The GUI shows the question and the various options to the user. Once user submits the answer the response is sent to the server at the send of each question the GUI informs the user whether the answer is correct or wrong and updates the score on the screen.

At the end of the quiz the GUI indicates who has won

**Source Code**:

**Server**

import java.io.BufferedInputStream;

import java.io.DataInputStream;

import java.io.DataOutputStream;

import java.io.IOException;

import java.net.ServerSocket;

import java.net.Socket;

public class ChatServer implements Runnable

{

private Socket socket = null;

private ServerSocket server = null;

private Thread thread = null;

private DataOutputStream streamOut = null;

private DataInputStream streamIn = null;

public static boolean answered[] = new boolean[2];

public static boolean scoresent[] = new boolean[2];

private int port\_number;

public static int score[] = new int[2];

private static srvquestion q0;

private static srvquestion q1;

private static srvquestion q2;

private static srvquestion q3;

private static srvquestion q4;

public static srvquestion[] q=new srvquestion[5];

private static int i=0,flag=0;

public ChatServer(int port)

{

try

{

port\_number=port;

System.out.println("Binding to port " + port + ", please wait ...");

server = new ServerSocket(port);

System.out.println("Server started: " + server);

answered[0]=false;

answered[1]=false;

start();

}

catch(IOException ioe)

{

System.out.println(ioe);

}

}

public void run()

{

String str;

while (thread != null)

{

try

{

System.out.println("Waiting for a client ...");

socket = server.accept();

System.out.println("Client accepted: " + socket);

open();

boolean done = false;

while (i!=5)

{

try

{

//Get Question and options

str = q[i].qandop();

//Broadcast it

streamOut.writeUTF(str);

//receive ans

String line = streamIn.readUTF();

System.out.println(line);

//broadcast correct or not

if (check(line,q[i]))

{

streamOut.writeUTF("true");

}

else

{

streamOut.writeUTF("false");

}

if(this.port\_number==3000)

{

System.out.println("3000 answered");

answered[0]=true;

while(!answered[1])

{

try {

Thread.sleep(10);

} catch (InterruptedException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

else if(this.port\_number==3001)

{

System.out.println("3001 answered");

answered[1]=true;

while(!answered[0])

{

try {

Thread.sleep(10);

} catch (InterruptedException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

if(answered[0] && answered[1])

{

if(flag==0)

{

i++;

flag=1;

}

else

{

flag=0;

}

try {

Thread.sleep(1000);

catch (InterruptedException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

answered[0]=false;

answered[1]=false;

}

try {

Thread.sleep(100);

}

catch (InterruptedException e) {

// TODO Auto-generated catch bloc e.printStackTrace();

}

}

catch(IOException ioe)

{ done = true; }

}

//end of while loop

//String line = streamIn.readUTF();

//System.out.println(line);

//asking for score

streamOut.writeUTF("send score");

String sc= streamIn.readUTF();

System.out.println(sc);

if(this.port\_number==3000)

{score[0]=Integer.parseInt(sc);

scoresent[0]=true;

}

else

{score[1]=Integer.parseInt(sc);

scoresent[1]=true;

}

if(this.port\_number==3000 && scoresent[0]==true)

{

while(!scoresent[1])

{

try {

Thread.sleep(10);

} catch (InterruptedException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

else if(this.port\_number==3001 && scoresent[1]==true)

{

while(!scoresent[0])

{

try {

Thread.sleep(10);

} catch (InterruptedException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

if(score[1]>score[0])

streamOut.writeUTF("Winner is 3001 and Loser is 3000");

else

streamOut.writeUTF("Winner is 3000 and Loser is 3001");

streamOut.writeUTF("bye");

close();

break;

}

catch(IOException ie)

{ System.out.println("Acceptance Error: " + ie); }

}

}

public void start()

{ if (thread == null)

{ thread = new Thread(this);

q0 = new srvquestion("Who is the president of India?","Pranab Mukherjee","Donald Trump","Barack Obama","Manmohan Singh",1);

q1 = new srvquestion("Who is the PM of India?","Pranab Mukherjee","Donald Trump","Narendra Modi","Manmohan Singh",3);

q2 = new srvquestion("Who was in the movie NH10?","Alia Bhatt","Aishwariya Rai","kareena Kapoor","Anushka Sharma",4);

q3 = new srvquestion("Who bulit the Taj Mahal?","Shah Jahan","Akbar","Jahangir","Aurangzeb",1);

q4 = new srvquestion("Who is the founder of Reliance Jio?","Bill Gates","Mukesh Ambani","Anil Ambani","Vijay Mallya",2);

q[0]=q0;

q[1]=q1;

q[2]=q2;

q[3]=q3;

q[4]=q4;

thread.start();

}

}

public boolean check(String n,srvquestion obj)

{

int num=Integer.parseInt(n);

if(num==obj.ans)

return true;

else

return false;

}

public void open() throws IOException

{ streamIn = new DataInputStream(new BufferedInputStream(socket.getInputStream()));

streamOut = new DataOutputStream(socket.getOutputStream());

}

public void close() throws IOException

{ if (socket != null) socket.close();

if (streamIn != null) streamIn.close();

}

public static void main(String args[])

{

ChatServer server1 = null;

ChatServer server2 = null;

// if (args.length != 1)

// System.out.println("Usage: java ChatServer port");

// else

server1 = new ChatServer(3000);

server2 = new ChatServer(3001);

}

}

class srvquestion{

String quest,op1,op2,op3,op4;

int ans;

srvquestion(String quest, String op1, String op2, String op3, String op4, int ans)

{

this.quest = quest;

this.op1 = op1;

this.op2 = op2;

this.op3 = op3;

this.op4 = op4;

this.ans=ans;

}

public String getquestion()

{

return quest;

}

public String qandop()

{

return quest+"\n"+op1+"\n"+op2+"\n"+op3+"\n"+op4+"\n";

}

}

**Client**

package application;

import java.io.BufferedInputStream;

import java.io.BufferedReader;

import java.io.DataInputStream;

import java.io.DataOutputStream;

import java.io.IOException;

import java.io.InputStreamReader;

import java.net.Socket;

import java.net.UnknownHostException;

public class ChatClient

{ private Socket socket = null;

private BufferedReader console = null;

private DataOutputStream streamOut = null;

private DataInputStream streamIn = null;

public static int score=0;

public static int correctanswercount = 0;

public static int wronganswercount = 0;

String line = "",qnop="",result="",end="";

public ChatClient(String serverName, int serverPort)

{

System.out.println("Establishing connection. Please wait ...");

try

{ socket = new Socket(serverName, serverPort);

System.out.println("Connected: " + socket);

start();

}

catch(UnknownHostException uhe)

{ System.out.println("Host unknown: " + uhe.getMessage());

}

catch(IOException ioe)

{ System.out.println("Unexpected exception: " + ioe.getMessage());

}

}

public void start() throws IOException

{

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

streamOut = new DataOutputStream(socket.getOutputStream());

streamIn = new DataInputStream(new BufferedInputStream(socket.getInputStream()));

}

public void stop()

{ try

{ if (console != null) console.close();

if (streamOut != null) streamOut.close();

if (socket != null) socket.close();

}

catch(IOException ioe)

{ System.out.println("Error closing ...");

}

}

public String getquestion ()

{

try {

///getting q from server

qnop = streamIn.readUTF();

if(qnop.equals("send score"))

{

streamOut.writeUTF(score+"");

streamOut.flush();

qnop = streamIn.readUTF();

return qnop;

}

}

catch(IOException ioe)

{ System.out.println("Sending error: " + ioe.getMessage());

}

return qnop;

}

public void stopchat ()

{

stop();

}

/\*public static void main(String args[])

{

String str;

ChatClient client = null;

client = new ChatClient("localhost", 3001);

str = client.getquestion ();

client.stopchat ();

}\*/

public String checkAnswer(String ans)

{

try

{

streamOut.writeUTF(ans);

streamOut.flush();

//checking is myans is true or false

result = streamIn.readUTF();

if(result.equals("true"))

{

score+=5;

correctanswercount++;

return "Last answer: Correct!!";

}

else

{

wronganswercount++;

if(ans.equalsIgnoreCase("0"))

return "Time up";

return "Last answer: Wrong!!";

}

//System.out.println("Current Score:"+score);

}

catch(IOException ioe)

{ System.out.println("Sending error: " + ioe.getMessage());

return "";

}

}

}

**User Interface**

**Main,java**

package application;

import javafx.application.Application;

import javafx.event.ActionEvent;

import javafx.event.EventHandler;

import javafx.fxml.FXMLLoader;

import javafx.scene.Parent;

import javafx.scene.Scene;

import javafx.stage.Stage;

import javafx.scene.layout.BorderPane;

import javafx.scene.control.\*;

import javafx.scene.layout.StackPane;

public class Main extends Application {

public static int portno;

@Override

public void start(Stage primaryStage) {

try

{

Parent root = FXMLLoader.load(getClass().getResource("/application/quizfx.fxml"));

Scene scene = new Scene(root);

scene.getStylesheets().add(getClass().getResource("application.css").toExternalForm());

primaryStage.setScene(scene);

primaryStage.show();

}

catch(Exception e){

e.printStackTrace();

}

}

public static void main(String[] args) {

System.out.println ("arguments " + args[0]);

portno = Integer.parseInt(args[0]);

launch(args);

}

}

**Quixfx.fxml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<?import javafx.scene.control.Button?>

<?import javafx.scene.control.Label?>

<?import javafx.scene.control.RadioButton?>

<?import javafx.scene.control.ToggleGroup?>

<?import javafx.scene.layout.AnchorPane?>

<AnchorPane fx:id=*"anchor"* prefHeight=*"416.0"* prefWidth=*"585.0"* xmlns=*"http://javafx.com/javafx/8.0.111"* xmlns:fx=*"http://javafx.com/fxml/1"* fx:controller=*"application.controller"*>

<children>

<Button fx:id=*"submit"* layoutX=*"422.0"* layoutY=*"328.0"* mnemonicParsing=*"false"* onAction=*"#quizClient"* prefHeight=*"31.0"* prefWidth=*"105.0"* text=*"Submit"* />

<Label fx:id=*"message"* layoutX=*"35.0"* layoutY=*"104.0"* prefHeight=*"68.0"* prefWidth=*"539.0"* text=*" which city is called pink city?"* />

<RadioButton fx:id=*"rb1"* layoutX=*"29.0"* layoutY=*"220.0"* mnemonicParsing=*"false"* text=*"A. Jaipur"*>

<toggleGroup>

<ToggleGroup fx:id=*"tgroup"* />

</toggleGroup></RadioButton>

<RadioButton fx:id=*"rb2"* layoutX=*"321.0"* layoutY=*"220.0"* mnemonicParsing=*"false"* text=*"B. Udaipur"* toggleGroup=*"$tgroup"* />

<RadioButton fx:id=*"rb3"* layoutX=*"29.0"* layoutY=*"282.0"* mnemonicParsing=*"false"* text=*"C. Kota"* toggleGroup=*"$tgroup"* />

<RadioButton fx:id=*"rb4"* layoutX=*"321.0"* layoutY=*"282.0"* mnemonicParsing=*"false"* text=*"D. Raipur"* toggleGroup=*"$tgroup"* />

<Label fx:id=*"message1"* layoutX=*"146.0"* layoutY=*"344.0"* />

<Label fx:id=*"score1"* layoutX=*"35.0"* layoutY=*"52.0"* prefHeight=*"21.0"* prefWidth=*"491.0"* />

<Label fx:id=*"title"* layoutX=*"156.0"* layoutY=*"14.0"* prefHeight=*"31.0"* prefWidth=*"297.0"* text=*" WELCOME TO QUIZUP"* />

<Label fx:id=*"portdis"* layoutX=*"35.0"* layoutY=*"83.0"* prefHeight=*"21.0"* prefWidth=*"191.0"* text=*"Connecting to port"* />

<Label fx:id=*"optionslabel"* layoutX=*"35.0"* layoutY=*"175.0"* prefHeight=*"21.0"* prefWidth=*"96.0"* text=*"Options:"* />

</children>

</AnchorPane>

**Controller.java**

**package** application;

**import** javafx.animation.Animation;

**import** javafx.animation.KeyFrame;

**import** javafx.animation.KeyValue;

**import** javafx.animation.Timeline;

**import** javafx.event.ActionEvent;

**import** javafx.fxml.\*;

**import** javafx.scene.control.Label;

**import** javafx.scene.control.Button;

**import** javafx.scene.control.Slider;

**import** javafx.scene.control.ToggleGroup;

**import** javafx.scene.paint.Color;

**import** javafx.scene.shape.Rectangle;

**import** javafx.util.Duration;

**import** javafx.scene.control.RadioButton;

**import** javafx.animation.Timeline;

**import** java.util.\*;

**public** **class** controller {

@FXML **private** Label message,message1,score1,portdis,optionslabel;

@FXML **private** Slider move;

@FXML **private** RadioButton rb1,rb2,rb3,rb4;

@FXML **private** Button submit;

String op[] = **new** String[4];

String question;

String questionClient;

String result;

question q;

**int** score;

**int** noQdone = 0;

**int** i,j;

**boolean** firsttime = **true**;

String ansno = "";

questionBank qb;

ChatClient client;

@FXML ToggleGroup tgroup;

Timeline timeline;

**public** **void** initialize() {

rb1.setToggleGroup(tgroup);

rb2.setToggleGroup(tgroup);

rb3.setToggleGroup(tgroup);

rb4.setToggleGroup(tgroup);

rb1.setSelected(**false**);

rb2.setSelected(**false**);

rb3.setSelected(**false**);

rb4.setSelected(**false**);

message1.setText("");

client = **new** ChatClient("localhost",Main.*portno*);

Integer port = **new** Integer(Main.*portno*);

portdis.setText("Connected on port: "+ port.toString());

System.***out***.println("Connected on port " + Main.*portno*);

qb = **new** questionBank();

qb.makeQuestion();

q = qb.getQuestion(client);

setQuestion(q);

timeline = **new** Timeline(**new** KeyFrame(

Duration.*millis*(30000),

ae -> noAnswer()));

//timeline.setCycleCount(Animation.INDEFINITE);

timeline.play();

/\*for(i=0;i<50000;i++)

for(j=0;j<50000;j++)

;

if(ansno.equalsIgnoreCase(""))

{

ansno = "5";

result = client.checkAnswer(ansno);

score = ChatClient.score;

message1.setText(result);

Integer scorestr = new Integer(score);

Integer correctstr = new Integer (ChatClient.correctanswercount);

Integer wrongstr = new Integer (ChatClient.wronganswercount);

score1.setText("Correct: " + correctstr.toString() + ", Wrong: " + wrongstr.toString() + ", Score: "+ scorestr.toString());

q = qb.getQuestion(client);

setQuestion(q);

}\*/

//questionClient = client.getquestion();

//message1.setText(questionClient);

//quizTimeline();

}

**public** **void** generateRandom(ActionEvent event )

{

Random r = **new** Random();

**int** no = r.nextInt(50) + 1;

message.setText(Integer.*toString*(no));

}

**public** **void** quizTimeline ()

{

**final** Rectangle rectBasicTimeline = **new** Rectangle(100, 50, 100, 50);

rectBasicTimeline.setFill(Color.***RED***);

**final** Timeline timeline = **new** Timeline();

timeline.setCycleCount(Timeline.***INDEFINITE***);

timeline.setAutoReverse(**true**);

**final** KeyValue kv = **new** KeyValue(rectBasicTimeline.xProperty(), 300);

**final** KeyFrame kf = **new** KeyFrame(Duration.*millis*(500), kv);

timeline.getKeyFrames().add(kf);

timeline.play();

}

**public** **void** noAnswer()

{

timeline.stop();

System.***out***.println("Time up");

ansno = "0";

result = client.checkAnswer(ansno);

score = ChatClient.*score*;

message1.setText(result);

Integer scorestr = **new** Integer(score);

Integer correctstr = **new** Integer (ChatClient.*correctanswercount*);

Integer wrongstr = **new** Integer (ChatClient.*wronganswercount*);

score1.setText("Correct: " + correctstr.toString() + ", Wrong: " + wrongstr.toString() + ", Score: "+ scorestr.toString());

q = qb.getQuestion(client);

**if**(q.finalResult.equalsIgnoreCase(""))

{

setQuestion(q);

timeline.playFromStart();

/\*for(i=0;i<50000;i++)

for(j=0;j<50000;j++)

;

if(ansno.equalsIgnoreCase(""))

{

ansno = "5";

result = client.checkAnswer(ansno);

score = ChatClient.score;

message1.setText(result);

scorestr = new Integer(score);

correctstr = new Integer (ChatClient.correctanswercount);

wrongstr = new Integer (ChatClient.wronganswercount);

score1.setText("Correct: " + correctstr.toString() + ", Wrong: " + wrongstr.toString() + ", Score: "+ scorestr.toString());

q = qb.getQuestion(client);

setQuestion(q);

}\*/

//wait1();

//wait1();

//message1.setText("");

}

**else**

{

message1.setText(q.finalResult);

//wait1();

message.setVisible(**false**);

rb1.setVisible(**false**);

rb2.setVisible(**false**);

rb3.setVisible(**false**);

rb4.setVisible(**false**);

submit.setVisible(**false**);

message.setVisible(**false**);

optionslabel.setVisible(**false**);

client.stop();

}

rb1.setSelected(**false**);

rb2.setSelected(**false**);

rb3.setSelected(**false**);

rb4.setSelected(**false**);

}

**public** **void** setQuestion(question q)

{

question = q.question;

op[0] = q.options[0];

op[1] = q.options[1];

op[2] = q.options[2];

op[3] = q.options[3];

message.setText("Question: "+question);

rb1.setText(op[0]);

rb2.setText(op[1]);

rb3.setText(op[2]);

rb4.setText(op[3]);

}

**public** **void** quizClient(ActionEvent event)

{

**if**(rb1.isSelected())

ansno = "1";

**else** **if**(rb2.isSelected())

ansno = "2";

**else** **if**(rb3.isSelected())

ansno = "3";

**else** **if**(rb4.isSelected())

ansno = "4";

**else**

message1.setText("Please select an answer");

timeline.stop();

//noQdone++;

//int i,j;

//wait1();

result = client.checkAnswer(ansno);

score = ChatClient.*score*;

message1.setText(result);

Integer scorestr = **new** Integer(score);

Integer correctstr = **new** Integer (ChatClient.*correctanswercount*);

Integer wrongstr = **new** Integer (ChatClient.*wronganswercount*);

score1.setText("Correct: " + correctstr.toString() + ", Wrong: " + wrongstr.toString() + ", Score: "+ scorestr.toString());

q = qb.getQuestion(client);

**if**(q.finalResult.equalsIgnoreCase(""))

{

setQuestion(q);

timeline.playFromStart();

/\*for(i=0;i<50000;i++)

for(j=0;j<50000;j++)

;

if(ansno.equalsIgnoreCase(""))

{

ansno = "5";

result = client.checkAnswer(ansno);

score = ChatClient.score;

message1.setText(result);

scorestr = new Integer(score);

correctstr = new Integer (ChatClient.correctanswercount);

wrongstr = new Integer (ChatClient.wronganswercount);

score1.setText("Correct: " + correctstr.toString() + ", Wrong: " + wrongstr.toString() + ", Score: "+ scorestr.toString());

q = qb.getQuestion(client);

setQuestion(q);

}\*/

//wait1();

//wait1();

//message1.setText("");

}

**else**

{

message1.setText(q.finalResult);

//wait1();

message.setVisible(**false**);

rb1.setVisible(**false**);

rb2.setVisible(**false**);

rb3.setVisible(**false**);

rb4.setVisible(**false**);

submit.setVisible(**false**);

message.setVisible(**false**);

optionslabel.setVisible(**false**);

client.stop();

}

rb1.setSelected(**false**);

rb2.setSelected(**false**);

rb3.setSelected(**false**);

rb4.setSelected(**false**);

}

**public** **void** quiz(ActionEvent event)

{

**if**((rb1.isSelected()) && (op[q.ansno].equals(rb1.getText())))

message1.setText("Last Answer: Correct");

**else** **if**((rb2.isSelected()) && (op[q.ansno].equals(rb2.getText())))

message1.setText("Last Answer: Correct");

**else** **if**((rb3.isSelected()) && (op[q.ansno].equals(rb3.getText())))

message1.setText("Last Answer: Correct");

**else** **if**((rb4.isSelected()) && (op[q.ansno].equals(rb4.getText())))

message1.setText("Last Answer: Correct");

**else**

message1.setText("Last Answer: Wrong");

noQdone++;

**int** i,j;

**for**(i=0;i<50000;i++)

**for**(j=0;j<50000;j++)

;

**if**(noQdone < qb.totq)

{//qb = new questionBank();

q = qb.getQuestion(client);

setQuestion(q);

}

rb1.setSelected(**false**);

rb2.setSelected(**false**);

rb3.setSelected(**false**);

rb4.setSelected(**false**);

}

**void** wait1()

{

**int** i,j;

**for**(i=0;i<10000;i++)

**for**(j=0;j<10000;j++)

;

}

}

**Questionbank.java**

**package** application;

**public** **class** questionBank {

question q[];

**static** **int** *currqno* = 0;

**int** curr;

**int** totq = 3;

String qnop;

**public** **void** makeQuestion()

{

q = **new** question[5];

q[0] = **new** question("Entomology is the science that studies","Behaviour of human beings","Insects","Origin of scientific terms","Formation of rocks",1);

q[1] = **new** question("Java language was developed by which company","Microsoft","IBM","Apple","Sun Microsystem",3);

q[2] = **new** question("First China war was fought between","China and Britain","China and France","China and Egypt","China and Greek",0);

}

**public** question makeQuestionClient(String qnop)

{

**int** i=0,j=0,begin;

i = qnop.indexOf('\n');

**if**(i != -1)

q[0].question = qnop.substring(0,i);

**else**

{

q[0].finalResult = qnop;

**return** q[0];

}

System.***out***.println(q[0].question);

begin = i+1;

i = i+1;

**while**((i=qnop.indexOf('\n',i)) > -1)

{

System.***out***.println(i);

q[0].options[j] = qnop.substring(begin,i);

System.***out***.println(q[0].options[j]);

j++;

i++;

begin = i;

}

**return** q[0];

}

question getQuestion()

{

**if**(*currqno* == 3)

{

**return** **null**;

}

**else**

{

curr = *currqno*;

*currqno*++;

**return** q[curr];

}

}

question getQuestion(ChatClient client)

{

qnop = client.getquestion();

System.***out***.println("Question from client: "+qnop);

**return** makeQuestionClient(qnop);

}

}

**Question.java**

**package** application;

**public** **class** question {

String question="";

String options[] = **new** String[4];

**int** ansno;

String finalResult="";

question(String qstr,String op1,String op2,String op3,String op4,**int** ans)

{

question = qstr;

options[0] = op1;

options[1] = op2;

options[2] = op3;

options[3] = op4;

ansno = ans;

}

}

**Output**





