

National Institute of Technology,Raipur

Department of Computer Science & Engineering



Buzzer System

A Term Project on Network Programming

GitHub Project Link: github.com/asamitraja/Buzzer

Submitted By:

Roll no: 14115001

Roll no: 14115008

Roll no: 14115016

Roll no: 14115006

Roll no: 14115052

Name: Abhik Sarkar

Name: Amit Singh

Name: Anurag Thakur

Name: Akshansh Sharma

Name: Mayank Singhal

Table of Contents

Abstract	3
Introduction (Motivation / Problem / Project Outline)	4
Detailed description of the solution to the problem.	4
Screenshots	6
Conclusion	11
References	11

Abstract

To make quiz easy and interesting we developed this software to help quizzers to answer more easily and help organizers to find who pressed button first. For this we have developed a software in java using netbeans platform and concept of client-server programming and multithreading. To make this application user friendly and interactive we have used javaFX. This project is consist of one server and many clients i.e. more than one client can connect to the one server. In this application many clients can connect to the one server at the same time using their IP addresses and whenever a contestant presses his/her buzzer his/her pop up in the organizers screen.

Introduction (Motivation / Problem / Project Outline)

When we have quizzes in any institution and we have a fastest answer first, we need buzzers but to wire those buzzers properly and detection of who pressed the buzzer first was an issue.

Motivation : To ease the process of finding out which buzzer or client has to answer first in a concurrent structure, an application had to be developed to correctly identify that user using its IP address as the identifier and hence laptops(with java) could be used to be clients in that system and a server could be used to identify the first pressed buzzer.

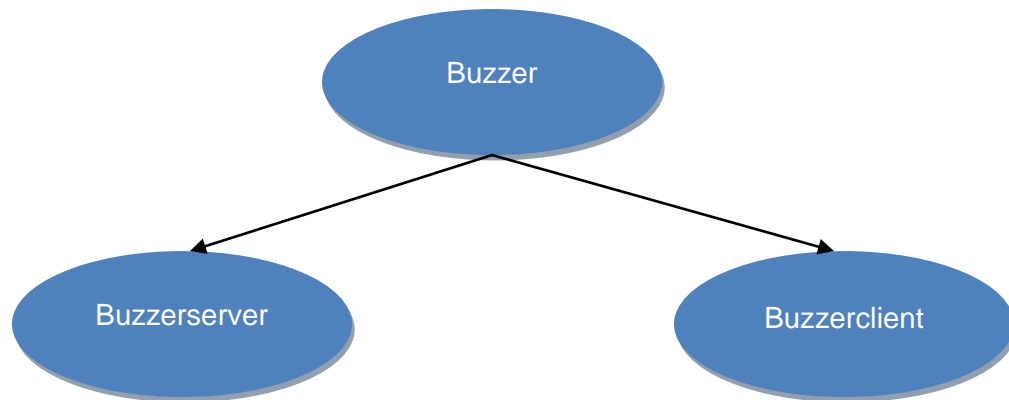
Detailed description of the solution to the problem.

JavaFX: JavaFX is a Java library used to build Rich Internet Applications. The applications written using this library can run consistently across multiple platforms. The applications developed using JavaFX can run on various devices such as Desktop Computers, Mobile Phones, TVs, Tablets, etc. JavaFX provides a rich set of graphics and media API's and it leverages the modern Graphical Processing Unit through hardware accelerated graphics. JavaFX also provides interfaces using which developers can combine graphics animation and UI control.

We used JavaFX extensively in our project to add the front-end part of the project, of both client and server side of the project.

Multithreading: A program can be divided into a number of small processes. Each small process can be addressed as a single thread (a lightweight process). You can think of a lightweight process as a virtual CPU that executes code or system calls. You usually do not need to concern yourself with lightweight processes to program with threads. Multithreaded programs contain two or more threads that can run concurrently and each thread defines a separate path of execution. This means that a single program can perform two or more tasks simultaneously. For example, one thread is writing content on a file at the same time another thread is performing spelling check.

Java Socket Programming: Java Socket programming is used for communication between the applications running on different JRE. Java Socket programming can be connection-oriented or connectionless. A socket is one end-point of a two-way communication link between two programs running on the network. Socket classes are used to represent the connection between a client program and a server program. The java.net package provides two classes--Socket and ServerSocket--that implement the client side of the connection and the server side of the connection.



Representation of flow of work in the Project

In this project we are having 3 Java Packages Buzzer, Buzzerserver and Buzzerclient. Buzzer server contains the java classes that are common to both buzzerserver and buzzerclient.

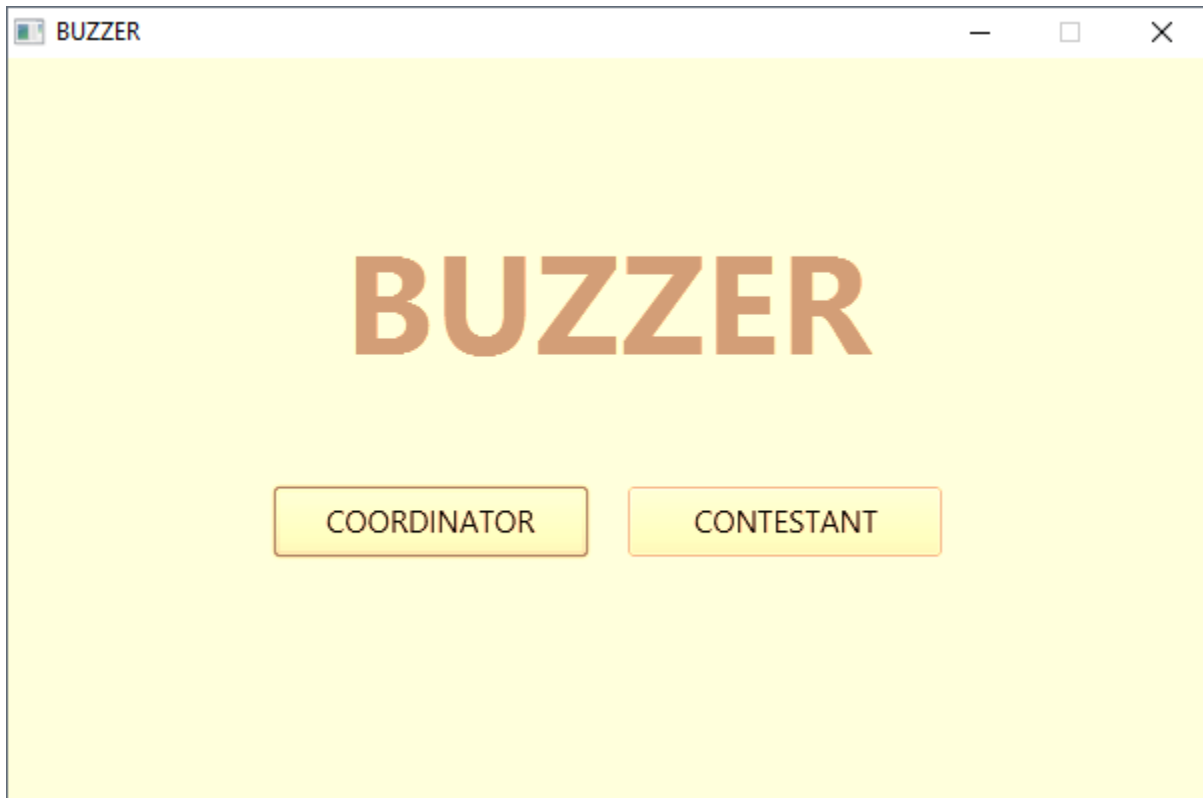
Buzzer contains the Buzzer.java whose main work is to load the file. It has two controllers DisplayMessageController and FrontPageController which are needed for the two fxml file for the front end namely displayMessage.fxml and frontPage.fxml. It also contains Message.java which helps to pop up the message.

Buzzerserver basically deals with the server portion of the buzzer. It has java class BuzzerServer.java which has all the logic related to the program. It stores all the client data in the form of ArrayList. It also encapsulates functions such as resetApplication which is used to reset the application. It also contains the ConnectionThread.java which is a class that contains all the connection related files. It takes the number of clients which are to be connected with the buzzers and waits for everyone them to get connected to the server. It's only after all the users get connected to the server then the server starts working. It also contains 3 fxml files namely ClientUI, frontPage and Messages which are needed for the better management of the JavaFX files.

Buzzerclient basically deals with the client portion of the buzzer. It has java class BuzzerClient.java which server as the basic backbone of the client. ConnectionThread.java takes care of the connection at client level and connects to the server. Suppose we are have 5 contestants these contestants connect to the server. Server waits for all the clients to join the network to start the buzzer. It contains frontPage.fxml , message.fxml , removeContestants.fxml and serverUI.fxml which are having front end files and the corresponding frontPagecontroller , messagecontroller , removeContestantscontroller and serverUIcontroller.

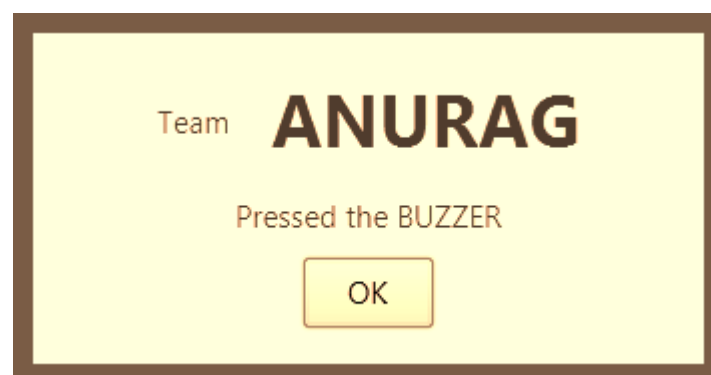
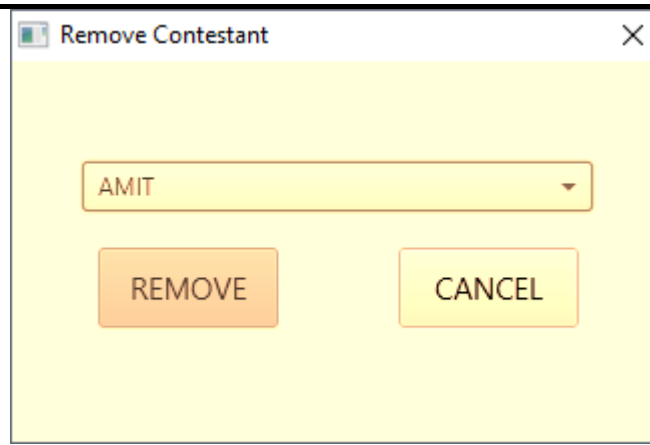
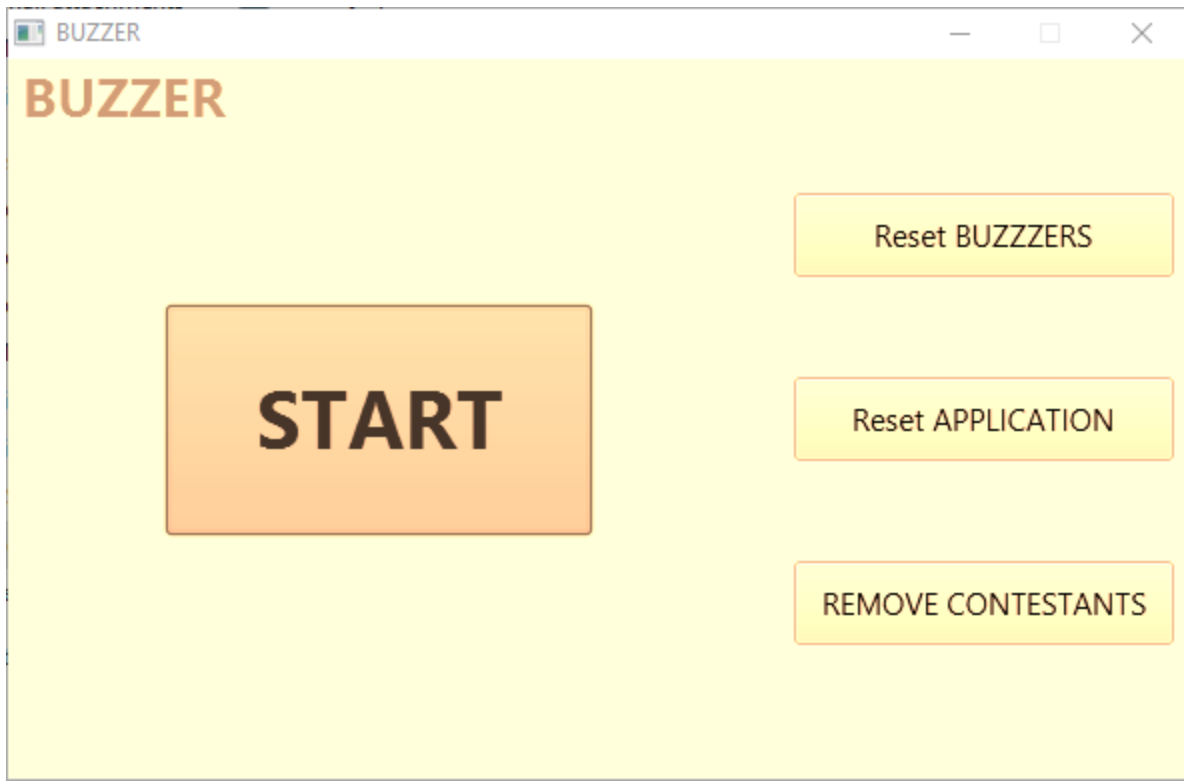
Screenshots

Home Screen:

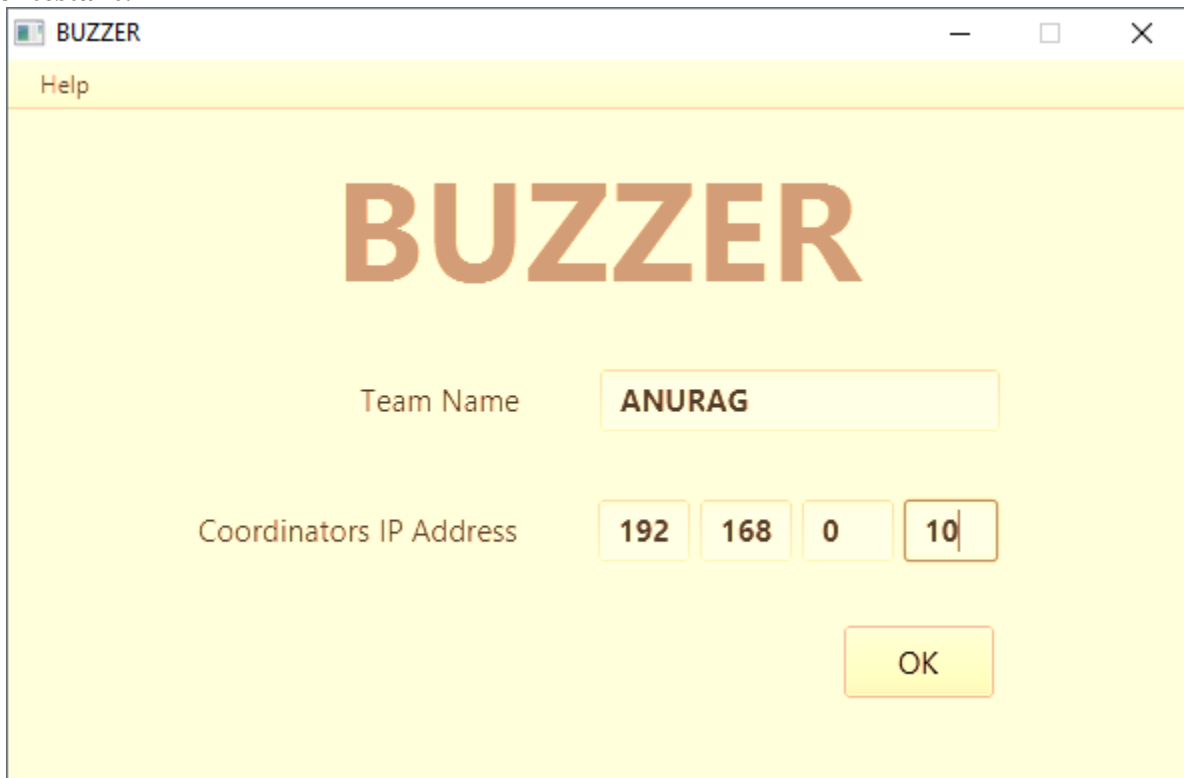


Coordinator:





Contestant:



A screenshot of a web application window titled "BUZZER". The window has a yellow background and a white header bar with a "Help" link. The main content area displays the word "BUZZER" in large, bold, orange letters. Below this, there are two input fields: "Team Name" with the value "ANURAG" and "Coordinators IP Address" with the value "192.168.0.10". An "OK" button is located at the bottom right of the form.

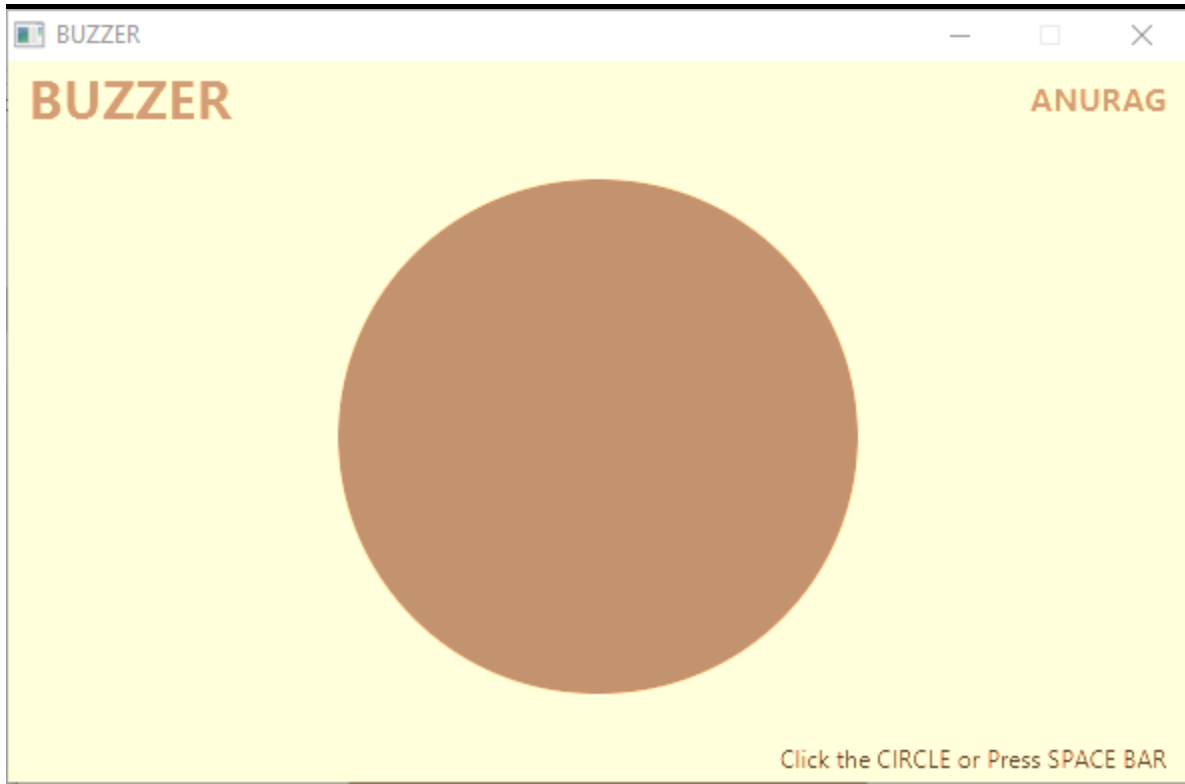
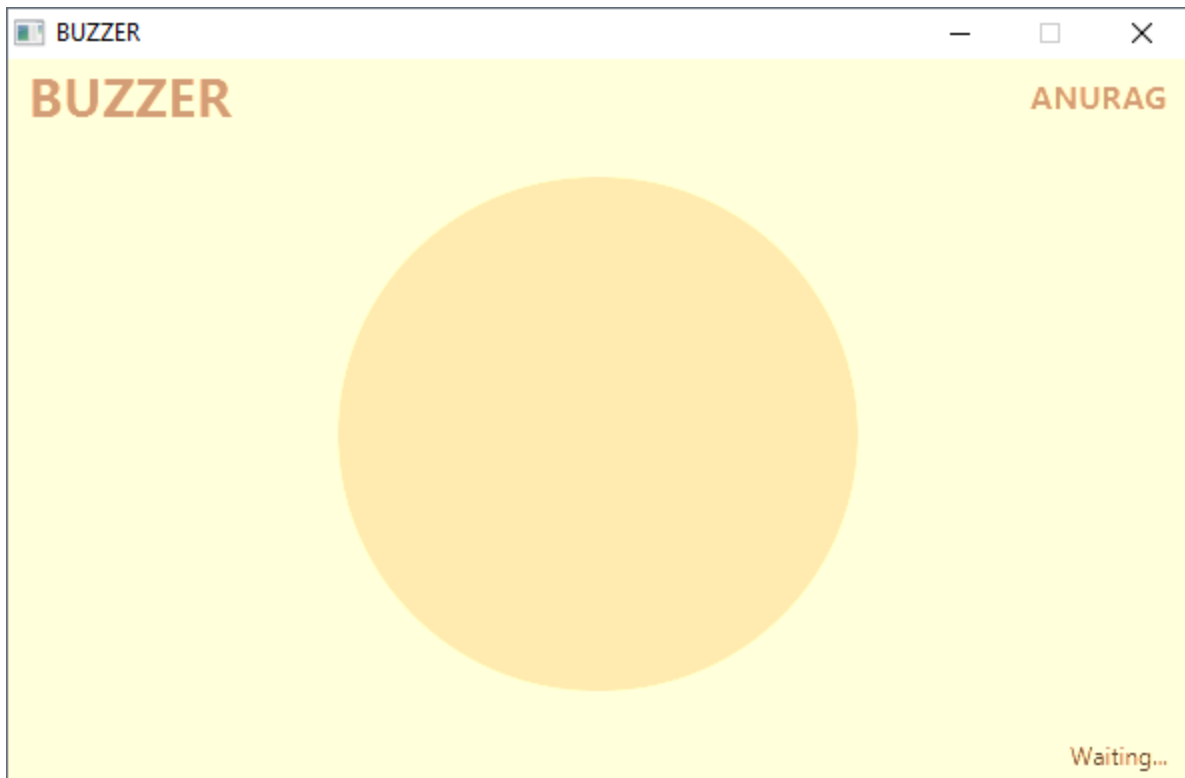
BUZZER

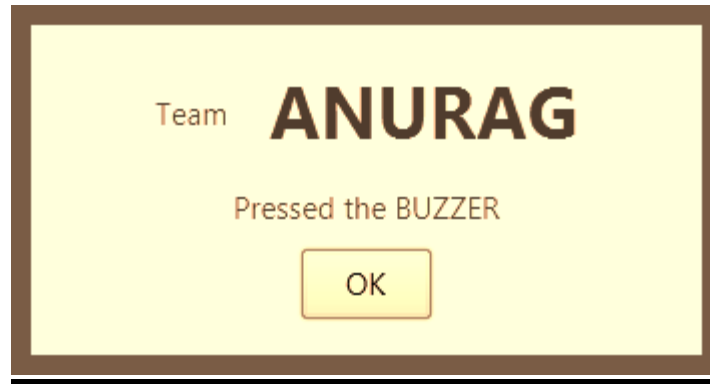
Help

BUZZER

Team Name

Coordinators IP Address





Conclusion

Hence we conclude that our project was working fine without any known bugs and we came with a working solution of the problem successfully. During implementation of this project we learned how to use the concept of server programming and multithreading. This project helped us to build our concept of network programming.

References

- Wikipedia.
- Tutorialspoint.
- Stackoverflow.
- javatpoint.