COMP90015: Distributed Systems

Assignment 2: Distributed Shared White Board

Author: RAO YUNHUI

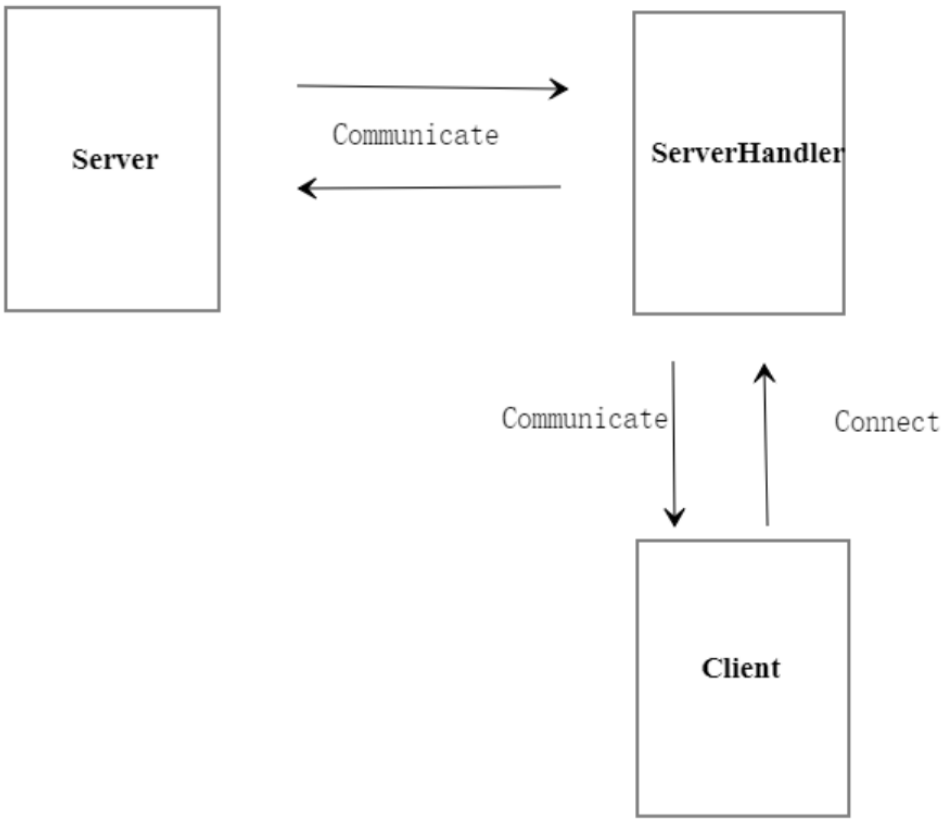
Student number: 1316834

1 System Architecture

In this assignment, I choose a client-server architecture, because the client-server architecture perfectly fits the requirements: the server can regulate clients and send instructions to all clients.

Also, the data is stored distributedly, in case that client would like to draw on his own after the server exits and stops shared white board. In this case, the user will only be disconnected and is left with the option to continue drawing without others' participation.

The overall architecture is like the figure below. The server listens for connection, create one handler for each client and communicate using the handler.



2 Message Format

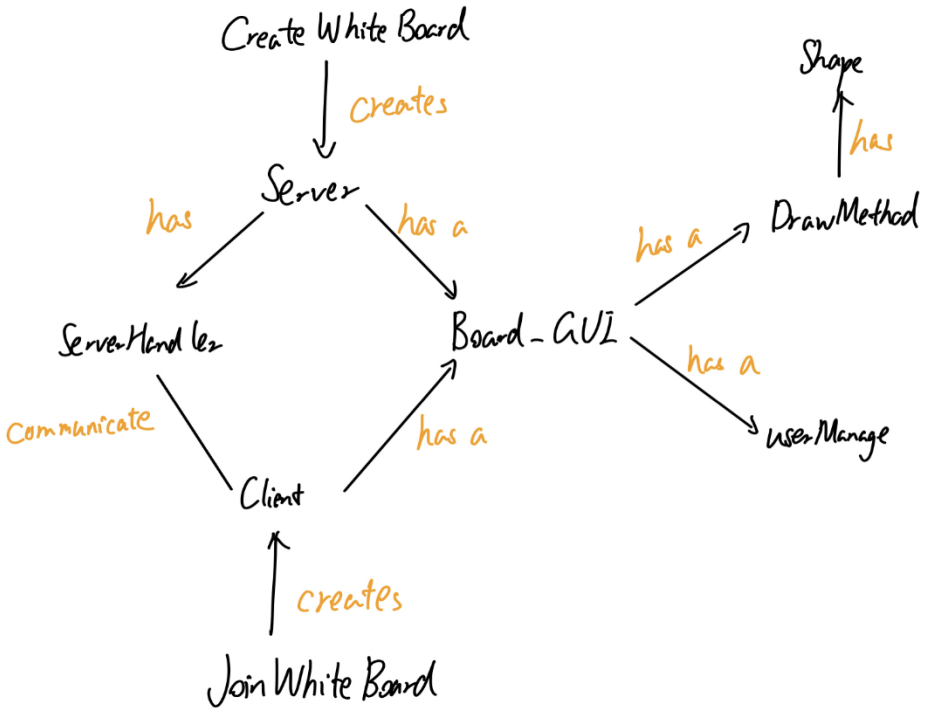
In this project, JSON is used for message exchange. The message exchange is basically for command deliver, so command field is necessary for all messages while others are optional. The fields of JSON objects are shown in the table below.

command	The name of command Command options: draw, exit, closefile, disconnect, newfile, join, newboard, clientNum, userlist, close
shape	The name of shape drawn Shape options: line, oval, triangle, rectangle, text
userlist	The string of user list
drawtext	The text being draw on canvas
colorR	The red value of the color of shape drawn
colorG	The green value of the color of shape drawn
colorB	The blue value of the color of shape drawn
x1	The initial x value of the shape drawn
x2	The end x value of the shape drawn
y1	The initial y value of the shape drawn
y2	The end y value of the shape drawn
clientNum	The allocated client number for clients

In the message exchange, TCP protocol is used, which enables data integrity. This ensures that the canvases of all users are the same at all times.

3 Design Diagrams

This is the interaction diagram for classes:



These are classes' properties for main classes

Server:

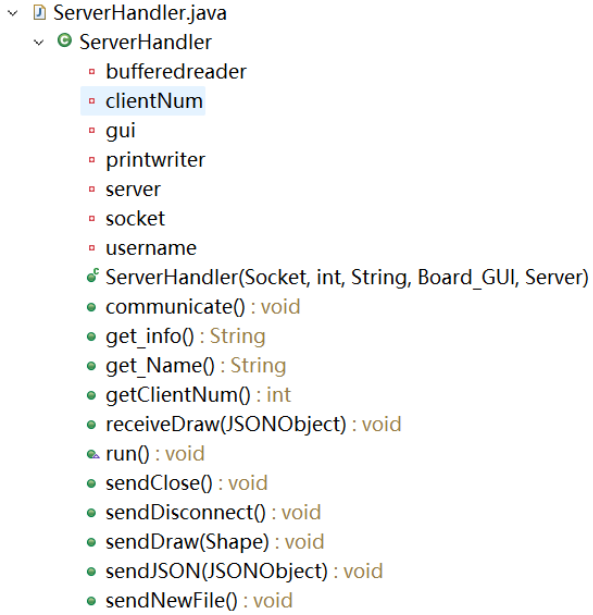
- Server.java
  - Server
    - gui
    - ip
    - port
    - threadPool
    - username
    - Server(String[], Board_GUI)
    - connect() : void
    - disconnectWithClient(int) : void
    - passChat(JSONObject) : void
    - removeThread(int) : void
    - sendChat(String) : void
    - sendClose() : void
    - sendDrawToAll(Shape, int) : void
    - sendNewBoard() : void
    - sendUserList() : void
    - updateClients() : void

Client:

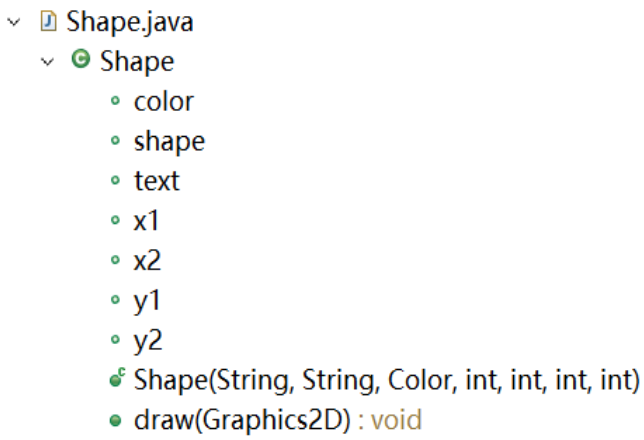
- Client.java
  - Client
    - bufferedreader
    - clientNum
    - gui
    - ip
    - port
    - printwriter
    - socket
    - username
    - Client(String[], Board_GUI)
    - communicate() : void
    - connect() : void
    - join() : void
    - receiveDraw(JSONObject) : void
    - sendChat(String) : void
    - sendDisconnect() : void
    - sendDraw(Shape) : void

DrawMethod:

ServerHandler:



Shape:



4 Implementation details

Figure below is the GUI of main page, including a canvas, chat window, user list showing area and text input area. There are also tool and color selector on the left. They can be used by simple click.

On the upper left, there are menu items. In the file column, there are New, Open, Save, Save as only for managers, and Close for everyone. In the Manage column, there is a user manage item for showing the user management page. In the user management page, the server can kick out a client.

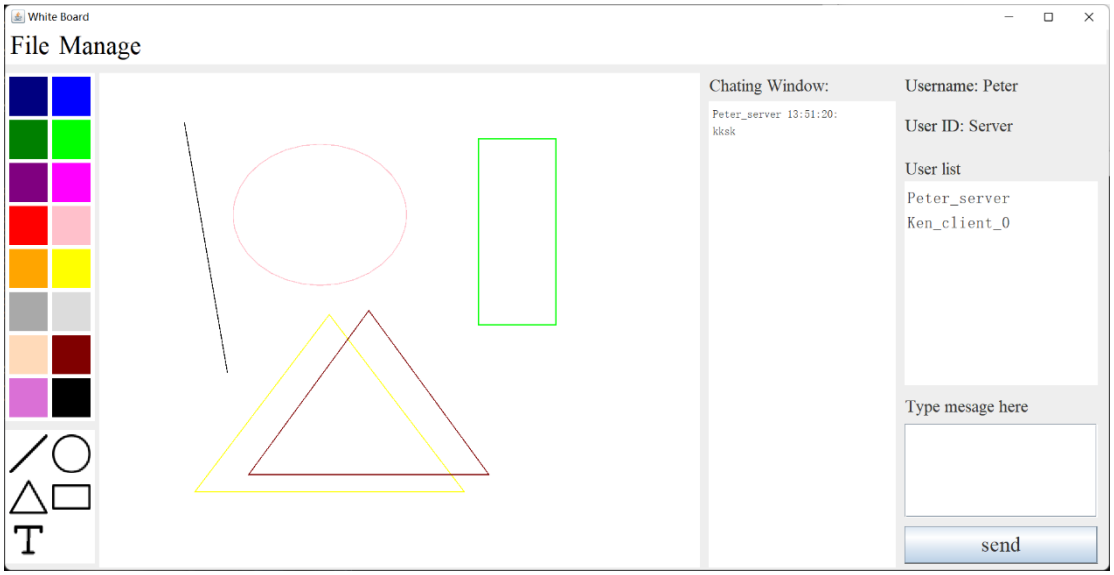


Figure GUI

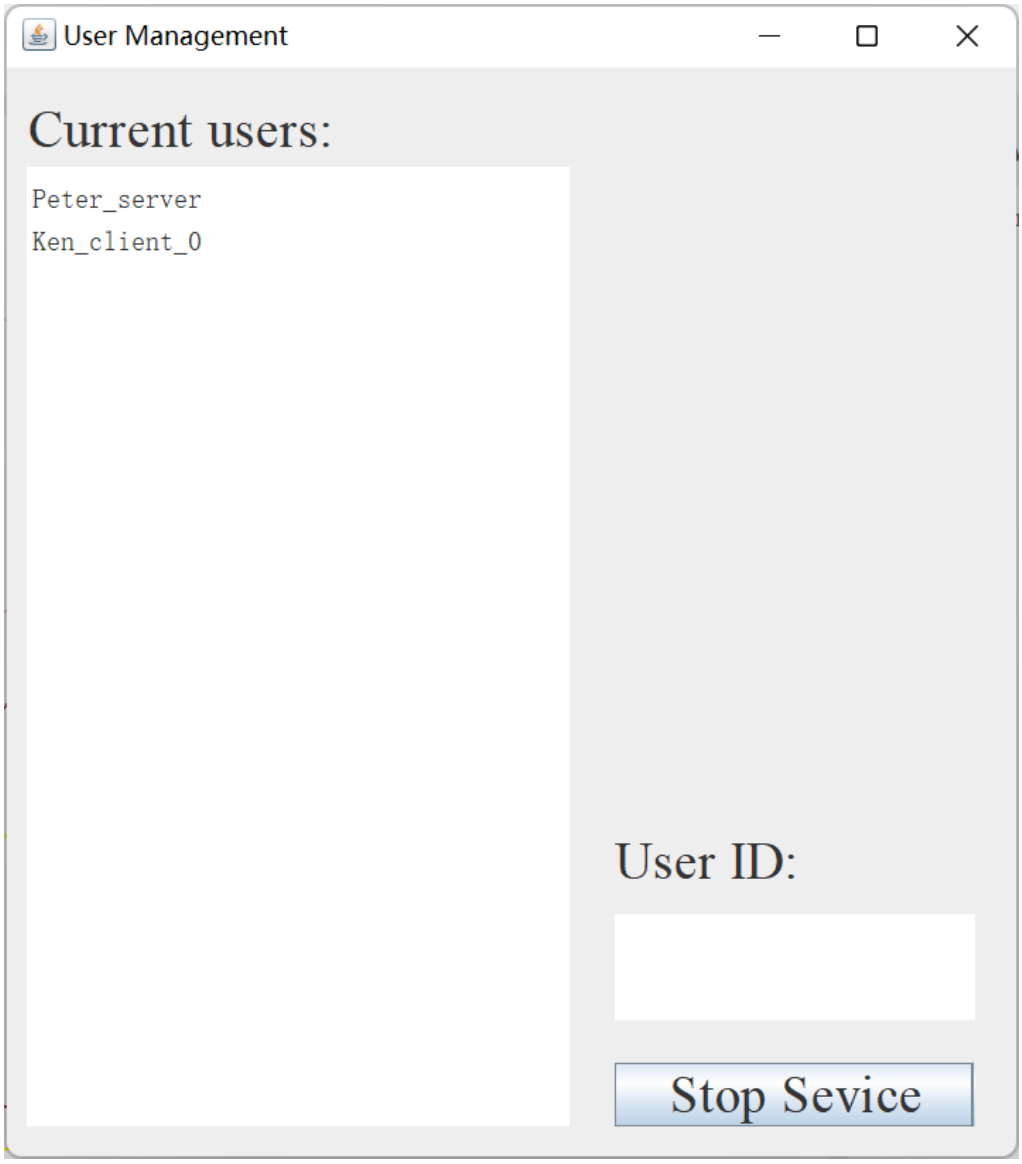


Figure user management page

If you are a client, you do not have permission to the items except Close. If you click it, a notification dialog will pop up.

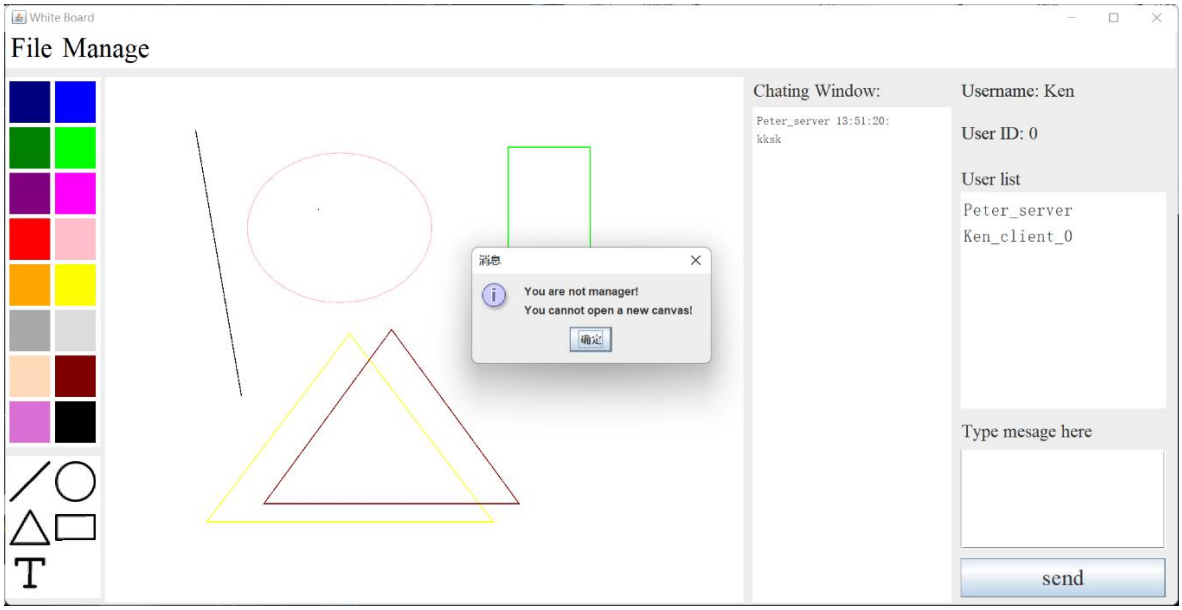
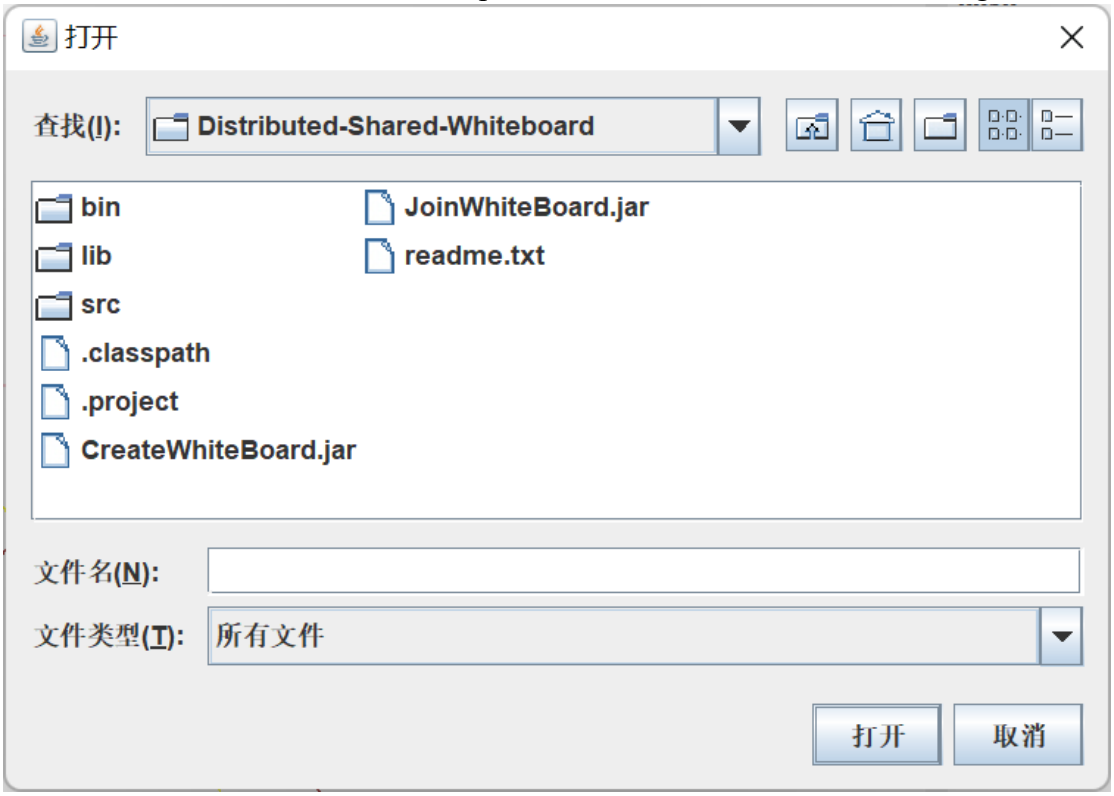


Figure No permission dialog

Also, the client needs to be permitted in order to join the shared whiteboard. When a client wants to join, a confirmation will be shown to the server. The server can decide whether to let the client in.

Figure Permission

The server can use a file chooser to open a saved file or save current drawing.



5 Innovation

The client can still draw on his own without connection with others after the server exits so that the client's pleasure will not be interrupted.