Functional Specification

The implemented Shared White Board meets the basic and advanced requirements, and has realized the following functionalities:

1. The Share White Board support multiple users to draw on a canvas simultaneously from different clients.

2. The new joined user will acquire the latest image of the current white board canvas.

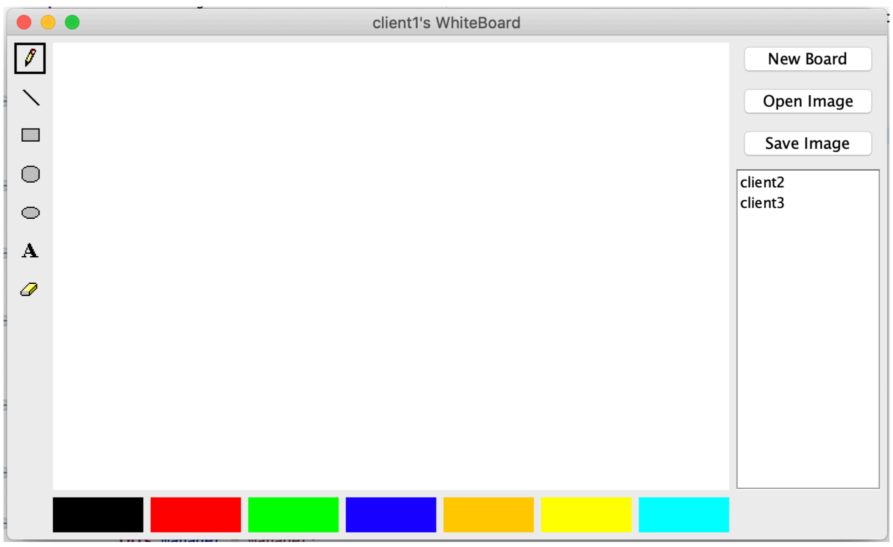
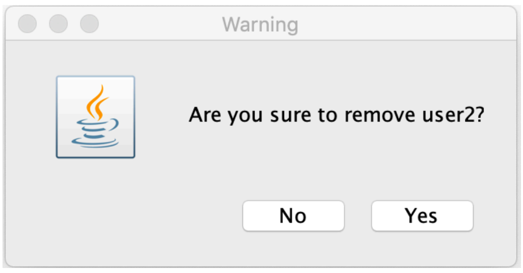


Figure 1 The White Board GUI

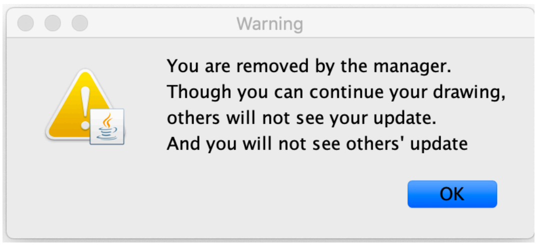
3. The White Board GUI (Figure 1) supports the user for the following activities
    a. Draw line, straight line, rectangle, circle, oval on the canvas
    b. Insert text anywhere inside the white board
    c. Provides an eraser to erase the drawings on specific area
    d. Provide 7 colour selections
    e. Display a list of active usernames from other clients
    f. Save/Open Image (Manager only)
    g. Clear Board (Manager only)
    h. Remove the user (Manager only)

4. A manager will be assigned to the first joined client and has the privilege to save/open image, clear board, remove any user.
5. The manager can remove any user by double click the username from his white board window. A confirmation dialog (Figure 2) will be provided to the manager for removing the user. The removed user will be notified and can continue drawing locally (Figure 3).


Warning

Are you sure to remove user2?

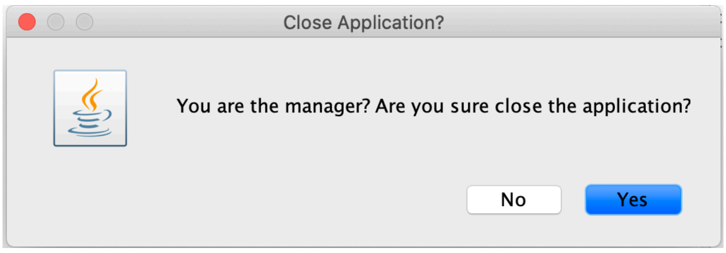
Figure 2 Manage confirm to remove a user


Warning

You are removed by the manager.
Though you can continue your drawing,
others will not see your update.
And you will not see others' update

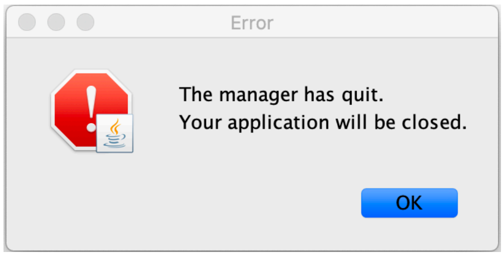
Figure 3 Removed user gets the notification

6. If the manager quit the shared white board, the other active clients will be notified, and force quit. See Figure 4 and Figure 5.


Close Application?

You are the manager? Are you sure close the application?

Figure 4 Confirmation Dialog when Manager close the window


Error

The manager has quit.
Your application will be closed.

Figure 5 Error dialog for other clients when manage closed the window

System Architecture

The Shared White Board consist of 3 components: server, remote interface and client. The communication is implemented via Java RMI application. The creation of the GUI and the drawing of the canvas are implemented by the java *awt* and *swing* library.

The Component of the Share White Board:

1. Server

   The server is responsible for the following tasks:

   a. Create a main method to register the white board service in the RMI Name Registry

   b. Receive the client join request, assign the manager and add/manage/remove the clients
   c. Receive the update from the client and publish the update to the other active client
2. Remove Interface
   a. Define the interface of the RMI remote method between the server and client
3. Client
   a. Create the white board HMI GUI
   b. Initialize the white board canvas image to synchronize with the other client
   c. Handle the user interaction for local activities, e.g., change colour, change shape, draw on the canvas
   d. Send the white board update to the server
   e. Receive the other client update from the server and synchronize the update in the local GUI

Overall Class Design

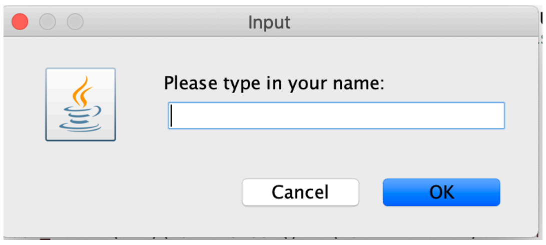
The server side (Server Package) contains 3 classes:

1. StartServer class:
   This class is the main() entry of the program. It is responsible for registering the WhiteBoardServer in the RMI Name Registry and handle the registry exceptions.

2. WhiteBoardServer class:
   This class implements the WhiteBoardServerinterface which implements the methods to process the client message and publish the update to other client, which including:
   a. Register the new client. If the client has duplicated name and assign a random number to this client.
   b. Assign the first client as manager. The manager has the privilege of save image, open image, clear the white board, remove user.
   c. Publish the new joined client name to all other client for display.
   d. Receive the white board update from each client and publish the update to all other active clients
3. ClientManager class:
   This class stores the list of clients and contains the methods for managing the clients which are used by the WhiteBoardServer class.

The Client side (Client package) contains 4 classes:

1. StartClient class:
   This class implements the WhiteBoardBlientInterface and provides the main() entry of the client and provides the following methods:
   a. Look up the WhiteBoardInterface server from the RMI Name Registry.
   b. On start-up, presents the user a “Input Username” dialog (Figure 6) to register the user in the server. If no username inserted, it will assign a random unique name to this user.


Input

Please type in your name:

Figure 6 Input Username Dialog

c. Build and launch the White Board GUI and enable the user for drawing (PaintBoard class) after user successful registered in the server.
d. Display the list of other users. If the user is a manager, display a “*” next to the username to indicate it. See Figure 7

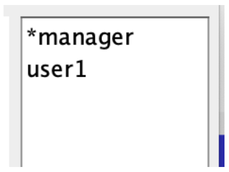

*manager
user1

Figure 7 Manage name is flagged with a ""*

e. Handles the manager function, if the user is registered as the manager.
f. Display the dialog window for specific conditions
g. Acquire the update from the server and request to update the local white board canvas
h. Force close the window when the manager is quit

2. PaintBoard
   This class mainly handle the mouse movement on the canvas and realize drawing on the canvas.
   a. Listen to the mouse motion on the white board
   b. Draw the selected shape on the white board
   c. Send the shape information including username, point coordinates, colour, shape and etc to the server
   d. Initialize the white board to synchronize with the manager’s image when the client joining the shared white board
3. CreateShape
   This class defines the methods that are used by the PaintBoard class for drawing the shapes.
4. DrawItemWrapper
   This class implements the WhiteBoardMsgInterface. It wraps the drawing information messages that are exchanged between the server and the client. A internal defined communication protocol.

The Remote interface (Remote class) side contains 3 Interfaces:

1. WhiteBoardServerInterface
   Extends the remote method and registers the WhiteBoardServer methods
2. WhiteBoardClientInterface
   Extends the remote method and registers the StartClient methods
3. WhiteBoardMsgInterface

Extends the remote method and registers the DrawItemWrapper methos

Innovations

The following innovation points have been added in this system:

- Display user friendly icon and colour palette on the whiteboard GUI.
- Provide a visual feedback for the selected drawing mode and colour. A border with the corresponding pen colour will be displayed around the icon for the current drawing mode. See Figure 8 below.

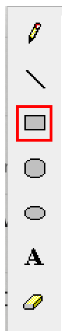


Figure 8 The border shows the current mode is rectangle with red pen

- All clients can see the other active clients and know who is the manager. See Figure 7.
- The whiteboard supports an eraser