CST 8221 – JAP - Assignment #2, Part 1

Demonstration Date: prior or on April 3rd, 2020

Earnings: 5% of your total course mark

Purpose: Building the GUIs

The purpose of Assignment #2 is to build a multi-threaded client/server chat application. In Part 1 of the assignment you are to build the GUI. In Part 2 of the assignment you are to write the client and the server code.

Problem Specification:

In this part of Assignment #2 you are to build a relatively simple GUI for the client and server part application. You may use the Swing API or JavaFX API to build the GUI. Your GUI must have exactly the same appearance (including the sizes of the components) as the one shown in the figures below.
Important note: If you decide to use GUI **builders** to build the GUI, you may not be able the implement the Part 2 of the assignment.

Swing GUI

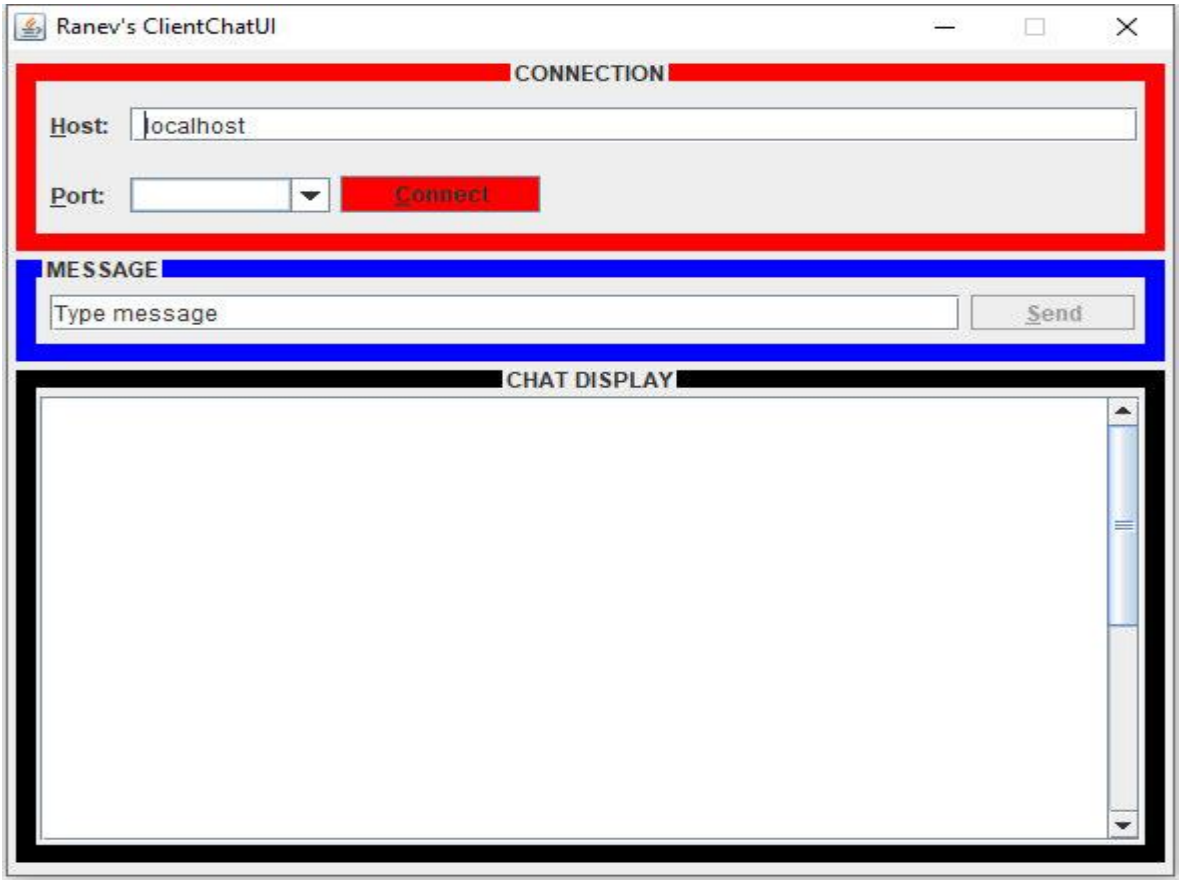

Fig. 1. The Client Swing GUI at launch.

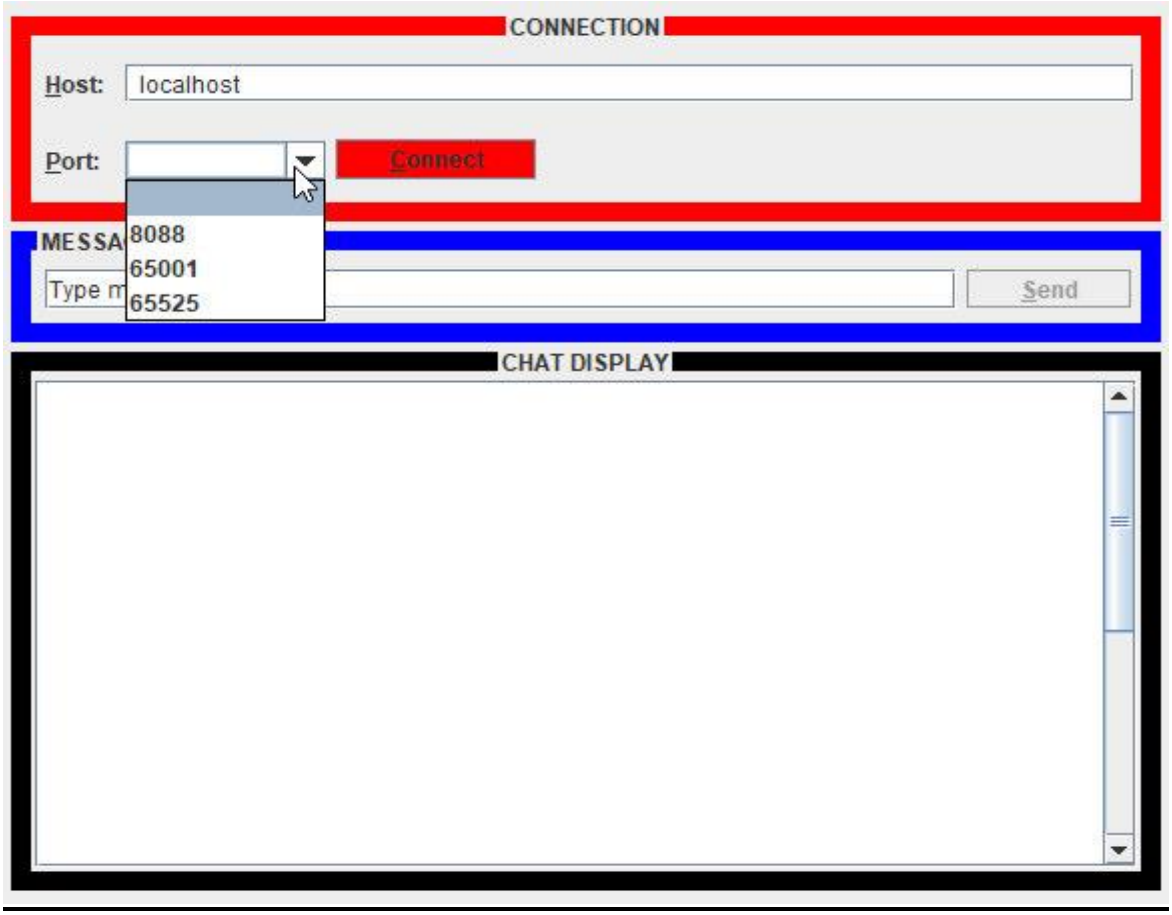

Fig. 2. The Client Swing GUI with Combo Box opened.

Requirements for the Client Swing GUI:

- The size of the non-resizable frame is (588, 500). At launch the location of the frame must be set by the platform.
- The combo box and the **Connect** button must have the same size visually.
- The text fields and the combo box must have white backgrounds and must be editable. The displayed text must be left aligned.
- The preferred size of the **Host:** label is (35,30) and the **Port:** label is (35,30).
- All buttons must have mnemonics.
- The **Send** button must be disabled at launch and must have the same height as the adjacent text field. The text field must display the specified text at launch.
- The labels must have mnemonics and when the corresponding Alt-Key is pressed the focus must be transferred to the corresponding host text field or the port combo box.
- At launch the host text field must has the focus and (the insertion caret (cursor)) must be blinking at the beginning of the text field in front of the latter **l** of the text localhost. The text (localhost) in the host text field must be 5 pixels removed from the left margin of the text field.
- The CONNECTION panel must have a 10 pixels red titled line border with centered title.
- The MESSAGE panel must have a 10 pixels blue titled line border.
- The CHAT DISPLAY panel must have a 10 pixels black titled line border with centered title.

- The CHAT DISPLAY text area must have 30 rows and 45 columns. The text area must have horizontal and vertical scroll bars. Only the vertical scroll bar must be visible at launch. The horizontal scroll bar is not visible at launch but must become visible if the displayed text exceeds the number of columns displayed. The text area must not be editable.
- The Host label, the Port label and the left margin of the message text field must be visibly vertically aligned. The right margin of the host text field and the right margin of the Send button must be visibly vertically aligned. The left margin of the combo box and left margin of the host text field must be visibly vertically aligned.
- Replace my name in the frame title with your name.
- No event handling is required for this part of the assignment.

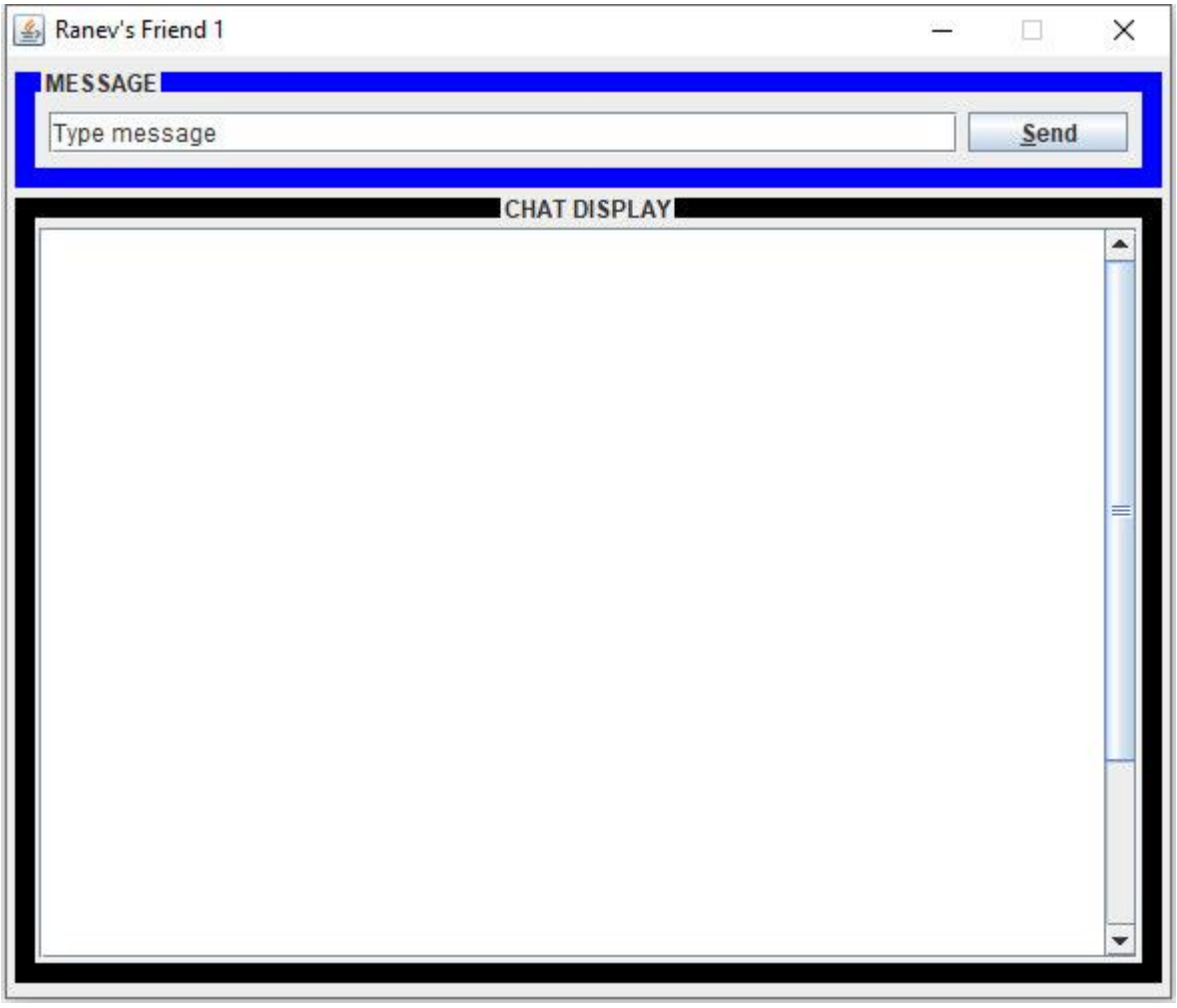

Fig. 3. The Server Swing GUI at launch.

Requirements for the Server Swing GUI:

- The size of the non-resizable frame is (588, 500). At launch the location of the frame must be in the center of the screen.
- The **Send** button must not be disabled at launch and must have the same height as the adjacent text field. It must have a mnemonic.
- The text field must display the specified text at launch. At launch the text field must has the focus and (the insertion caret (cursor) must be blinking at the beginning of the text field. The text field must be editable.
- The MESSAGE panel must have a 10 pixels blue titled line border.
- The CHAT DISPLAY panel must have a 10 pixels black titled line border with centered title.
- The CHAT DISPLAY text area must have 30 rows and 45 columns. The text area must have horizontal and vertical scroll bars. Only the vertical scroll bar must be visible at launch. The horizontal scroll bar is not visible at launch but must become visible if the displayed text exceeds the number of columns displayed. The text area must not be editable.
- Replace my name in the frame title with your name.
- No event handling is required for this part of the assignment.

Swing GUI Implementation

The Swing GUI implementation involves the following classes: ***Client, ClientChatUI, Server,*** and ***ServerChatUI***.

Class Client

The class ***Client*** must contain only a main method. In the main method you must call the ***ClientChatUI*** constructor and then sets the appropriate frame properties before making the frame visible. Do not forget that you must create the GUI in the event-dispatch thread.

Class ClientChatUI

The ***ClientChatUI*** class must inherit from ***JFrame***.

It must have a constructor that takes the frame title. In the constructor you must set the frame title and call a method *runClient()*.

The ***ClientChatUI*** class must have a private inner class ***WindowController*** which inherits from ***WindowAdapter***. You have to override the *windowClosing()* method. For this part of the assignment you must simply call *System.exit(0)* inside the method.

The ***ClientChatUI*** class must have another private inner class ***Controller*** which inherits from ***ActionListener***. The ***Controller*** will handle the user evens of the GUI. You have to override the *actionPerformed()* method. For this part of the assignment you can implement the method with an empty body (NOP).

The ***ClientChatUI*** class must have a public method *createClientUI()* which takes no parameters and returns ***JPanel***. In that method the entire GUI must be created in a ***JPanel*** and panel must be returned at the end of the method. A handler of type ***Controller*** must be added to all buttons and the combo box.

The private *runClient()* method takes no parameters and returns nothing. The method calls the *createClientUI()* and sets the content pane of the frame. It adds a window listener to the frame using an object of ***WindowController***.

In the second part of the assignment more methods will be added and some of the existing methods will be modified.

Class Server
The class ***Server*** must contain two method – the *main()* method and a static *launchClient()* method. The *launchClient()* method contains two parameters – a ***Socket*** and a String *title*. In the main method you must call *launchClient()* with a **null** argument for the socket and a title string.
The *launchClient()* method creates the GUI in the event-dispatch thread. It creates a frame calling the *ServerChatUI()* constructor with a **null** argument. The method sets the frame title and the location of the frame before making the frame visible.

Class ServerChatUI
The ***ServerChatUI*** class must inherit from ***JFrame***.

It must have a constructor that takes a ***java.net.Socket*** parameter. In the constructor you must set the class field ***socket*** and call the *setFrame()* and *runClient()* methods.

The ***ServerChatUI*** class must have a private inner class ***WindowController*** which inherits from ***WindowAdapter***. You have to override the *windowClosing()* method. For this part of the assignment you must simply dispose the frame and call *System.exit(0)* inside the method.

The ***ServerChatUI*** class must have another private inner class ***Controller*** which inherits from ***ActionListener***. The ***Controller*** will handle the **Send** button events of the GUI. You have to override the *actionPerformed()* method. For this part of the assignment you can implement the method with an empty body (NOP).

The ***ServerChatUI*** class must have a public method *createUI()* which takes no parameters and returns ***JPanel***. In that method the entire GUI must be created in a ***JPanel*** and the panel must be returned at the end of the method. It adds a handler of type ***Controller*** to the Send button.

The public ***final void setFrame()*** takes a ***JPanel*** as a parameter and adds it to the content pane of the frame. It set the size and the resizable properties of the frame. It adds a window listener to the frame using an object of ***WindowController*** and then returns.

The private *runClient()* method takes no parameters and returns nothing. For this part of the assignment you can implement the method with an empty body (NOP).

In the second part of the assignment more methods will be added and some of the existing methods will be modified.

Java FX GUI

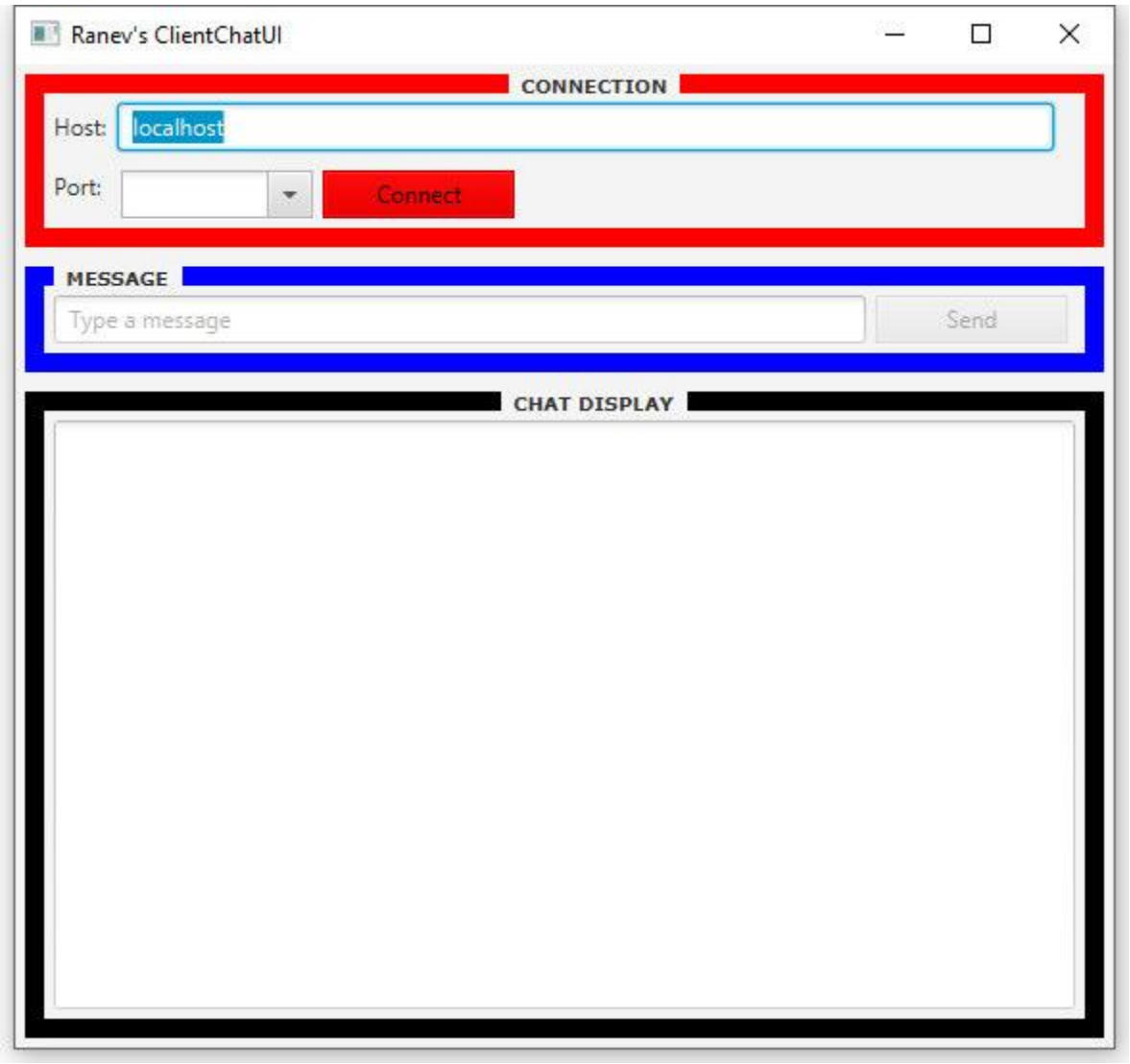

Fig. 4. The Client JavaFX GUI at launch.

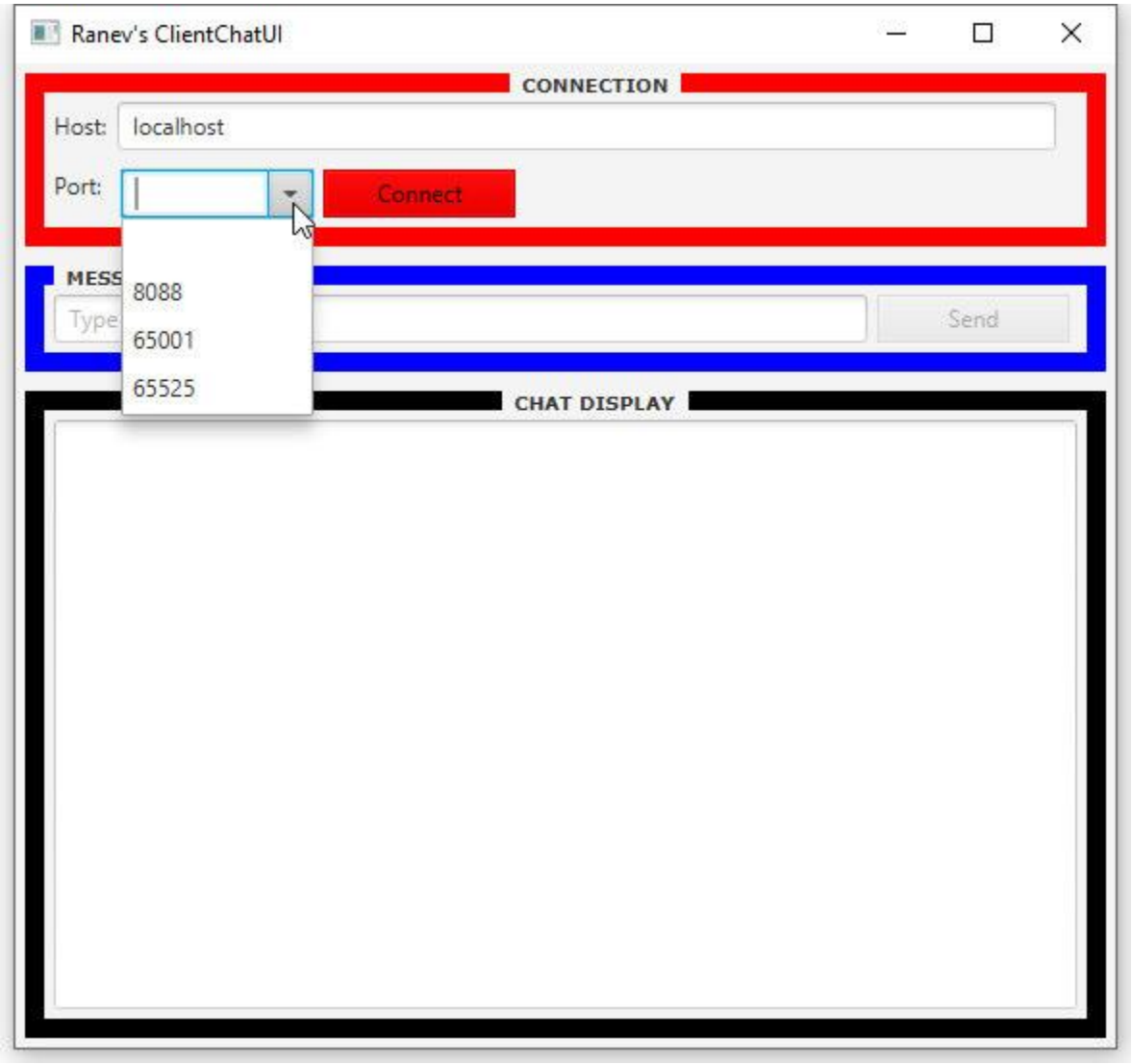

Fig. 5. The Client JavaFX GUI with Combo Box opened.

Requirements for the Client JavaFX GUI:

- The size of the non-resizable frame is (588, 500). At launch the location of the frame must be set to100 pixels from the left of the screen.
- The combo box and the **Connect** button have the same size visually.
- The text fields and the combo box must have white backgrounds and must be editable. The displayed text must be left aligned.
- All buttons must have mnemonics.
- The **Send** button must be disabled at launch and must have the same height as the adjacent text field. The text field must display the specified text at launch.
- The labels must have mnemonics and when the corresponding Alt-Key is pressed the focus must be transferred to the corresponding host text field or port combo box.
- At launch the host text field must has the focus and (the insertion caret (cursor)) must be blinking at the beginning of the text field in front of the latter **l** of the localhost text.
- The CONNECTION panel must have a 10 pixels red titled line border with centered title.
- The MESSAGE panel must have a 10 pixels blue titled line border.

- The CHAT DISPLAY panel must have a 10 pixels black titled line border with centered title.
- The CHAT DISPLAY text area must not be editable.
- The Host label, the Port label and the left margin of the message text field must be visibly vertically aligned. The right margin of the host text field and the right margin of the Send button must be visibly vertically aligned. The left margin of the combo box and left margin of the host text field must be visibly vertically aligned.
- Replace my name in the frame title with your name.
- No event handling is required for this part of the assignment.

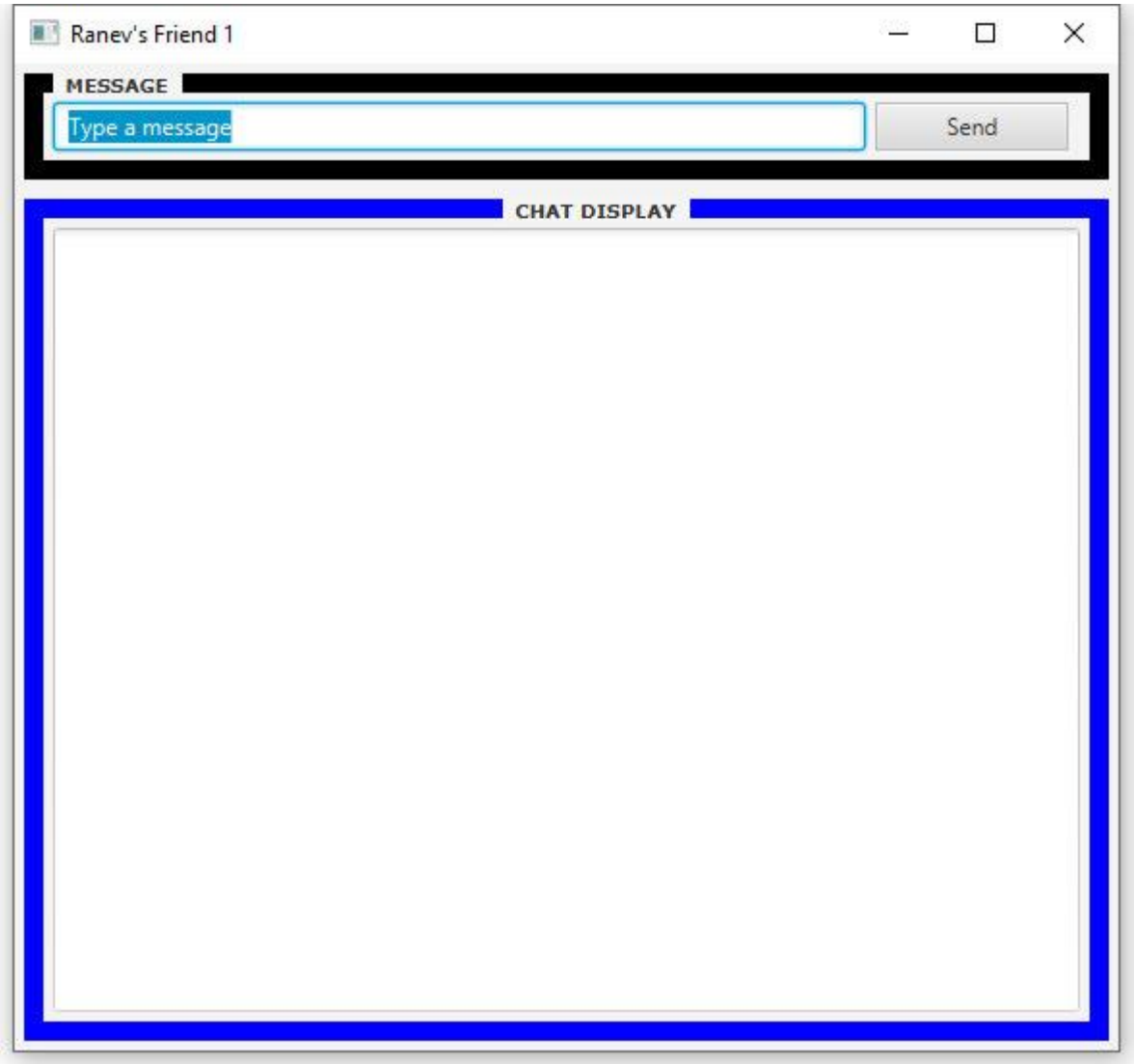

Fig. 6. The Server JavaFX GUI at launch.

Requirements for the Server Swing GUI:

- The size of the non-resizable frame is (588, 500). At launch the location of the frame must be in the center of the screen.
- The **Send** button must not be disabled at launch and must have the same height as the adjacent text field. It must have a mnemonic.
- The text field must display the specified text at launch. At launch the text field must has the focus and (the insertion caret (cursor) must be blinking at the beginning of the text field. The text field must be editable.

- The MESSAGE panel must have a 10 pixels blue titled line border.
- The CHAT DISPLAY panel must have a 10 pixels black titled line border with centered title.
- The CHAT DISPLAY text area must not be editable
- Replace my name in the frame title with your name.
- No event handling is required for this part of the assignment.

Java FX GUI Implementation

The JavaFX GUI implementation involves the following classes: ***Client, ClientChatUI, Server,*** and ***ServerChatUI***.

Class Client

The class ***Client*** must contain only a main method. In the main method launches the client GUI with the following line of code:

```
Application.launch(ClientChatUI.class);
```

Class ClientChatUI

The ***ClientChatUI*** class must inherit from ***Application***.
The ***start()*** method creates a Scene calling the *createScene()* method and sets the stage with the scene. Before showing the stage it set the size, the location, and the title of the stage.

The ***ClientChatUI*** class must have a private inner class ***Controller*** which inherits from ***EventHandler<ActionEvent>***. The ***Controller*** will handle the user evens of the GUI. You have to override the *handle()* method. For this part of the assignment you can implement the method with an empty body (NOP).

The ***ClientChatUI*** class must have a public method *createScene()* which takes no parameters and returns ***Scene***. In that method a scene must be created and returned at the end of the method. It adds a handler of type ***Controller*** to all buttons.

Since JavaFX does not provide a **TitleBorder** class like Swing, you must create an inner class called **BorderedTitledPane** which inherits from ***StackPane***. You should have a constructor that takes the title of the border, the position of the title, the color of the border, and the node (content) that is surrounded by the border. In the constructor you should set the padding of the supper class and create a ***Label*** for the title. After creating the ***Label*** set the font, the padding, the translateX, and the background of the title. Then you must set the alignment of the title (using the position parameter) in the ***StackPane***. Next, you have to create a new ***StackPane*** that will hold the content (node) surrounded by the title border. You have to set a line border, padding, and then add the content (node) to that pane. Finally, you have to add the title ***Label*** and the ***StackPane*** containing the content to the super class (***StackPane****)*.

In the second part of the assignment more methods will be added and some of the existing methods will be modified.

Class Server
The class ***Server*** must contain two method – the *main()* method and a static *launchClient()* method. The *launchClient()* method contains two parameters – a ***Socket*** and a title string. In the main method you must call *launchClient()* with a **null** argument for the socket and a title string. Here is the implementation of the *launchClient()* method:

```
private static void launchClient(Socket in, String title){
try{
    new JFXPanel();
    Platform.runLater(new Runnable() {
     @Override
       public void run(){
         new ServerChatUI(in,title).start(new Stage());
       }
     });
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

Class **ServerChatUI**
The **ServerChatUI** class must inherit from ***Application***.

It must have a constructor that takes a ***java.net.Socket*** parameter and a title string. In the constructor you must set the class field *socket* and the class field *title*.
The ***start()*** method must set a class field called *primaryStage* to the parameter of the start method. Then it creates a Scene calling the *createScene()* method and sets the stage with the scene. Before showing the stage it set the size, the location, and the title of the stage.

The **ServerChatUI** must have a private inner class ***Controller*** which inherits from ***EventHandler<ActionEvent>***. The Controller will handle the user evens of the GUI. You have to override the *handle()* method. For this part of the assignment you can implement the method with an empty body (NOP).

The **ServerChatUI** class must have a public method *createScene()* which takes no parameters and returns ***Scene***. In that method a scene must be created and returned at the end of the method. It adds a handler of type ***Controller*** to the Send button.

Since JavaFX does not provide a **TitleBorder** class like Swing, you must add the inner class **BorderedTitledPane**.

In the second part of the assignment more methods will be added and some of the existing methods will be modified.

Task:

Build the specified Swing or JavaFX GUI. The GUI screenshots have been taken under Windows 10 with the default “look and feel” with the image is captured with a screen resolution 1366X768. If you are using some other operating platform, you may have a different “look and feel” but the properties of all of the components and their relative locations must be the same. On your platform the frame title maybe left-justified.

What to Submit:

No paper submission is required for Part 1 of Assignment 2.
Code submission:
Compress in one .zip file all .java files and .class files. Upload the assignment zip file to Brightspace prior to or on the due date. The name of the zip file must have the following structure: Student’s family name followed by the last three digits of the student ID number followed by _JAP_A2P1 , and finally, followed by your lab section number (for example, s301). For example, *Ranev007_JAP_A2P1_s301.zip*.

Marking Scheme:

5 marks – The Client GUI and Server GUI must meet all requirements and must look exactly the same as the one shown in the figures.

4 marks – The Client GUI and Server GUI looks exactly the same as the one shown in the figures but does not meet two of the requirements.

3 marks – The Client GUI and Server GUI has all of the required components but they are not properly aligned and sized, or the GUI does not meet three or more requirements.

0 marks – The Client GUI is missing a component.

Enjoy the assignment. And do not forget that

“To have a server you ought to have a client first.” Business Rule #1

CST8221 – JAP, 27 February 2020, S^R