



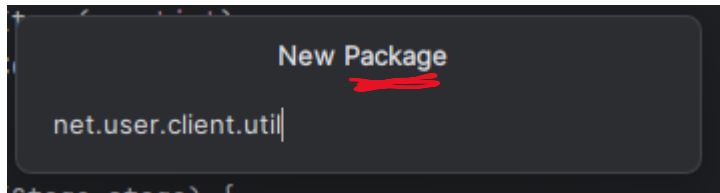
Мій Telegram

№3

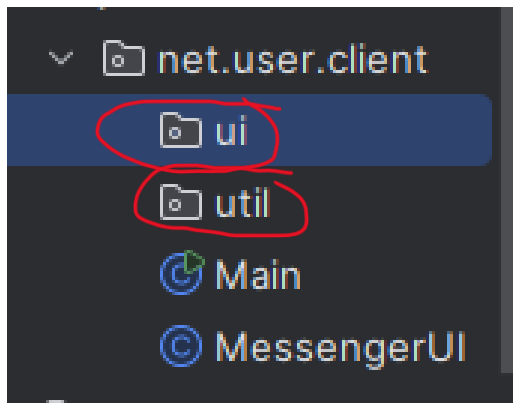
Інтерфейс Сервера та оформлення чату

Винесення в окремі програми

- Іноді програм стає надто багато, тому краще винести все в окремі класи – в нашому інтерфейсі буде багато.

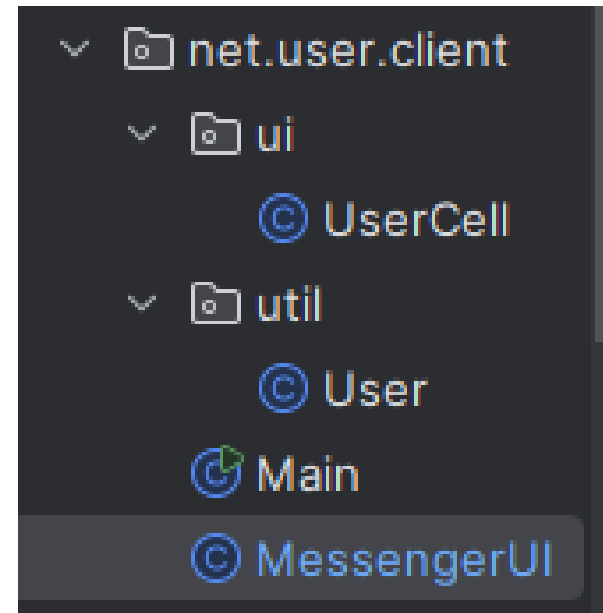


Відразу почнемо сортувати по пакетам



Завдання

Самостійно перенесіть наші статичні класи (підкласи) в окремі файли, виправіть імпорти в файлах



Виправлення

Приберемо зайву змінну, тому що ListView автоматично створює собі список.

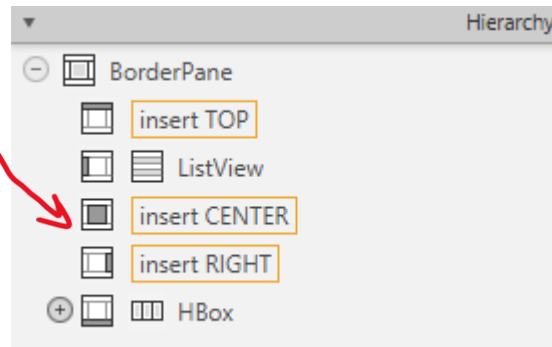
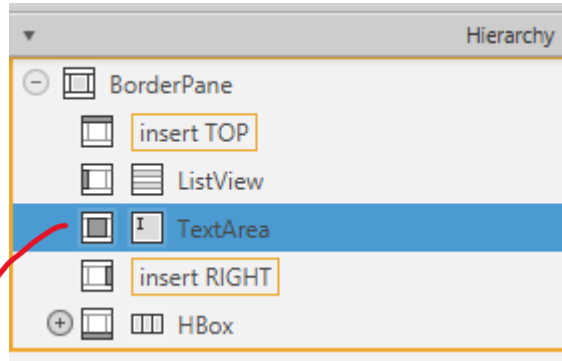
```
</>    @FXML public TextArea textArea;  
</>    @FXML public TextField messageArea;  
</>    @FXML public Button sendBtn;  
  
private final ObservableList<User> userList =  
    FXCollections.observableArrayList();  
  
public void show(Stage stage) {  
    FXMLLoader loader = new FXMLLoader(getClass().ge
```

```
@FXML  
public void initialize() {  
    listView.setCellFactory(info -> new UserCell());  
    listView.getItems().addAll(  
        new User("Serhii"),  
        new User("Andrii"),  
        new User("Taras"),  
        new User("Shush")  
    );  
}
```

Доступ до нього через **.getItems()**

Оформлення чату

- Наш чат на зараз це лише список з символами, а нам треба зробити красиві повідомлення, картинку для користувача і тд.
- Спочатку поміняємо в SceneBuilder TextArea на панель для розташування – ScrollPane, яка і дозволить листати повідомлення.
- Додавати будемо через програму, тому лишаємо пустою середину головної панелі.**



```
public void sendMessage(ActionEvent actionEvent) {
    User user = listView.getSelectionModel().getSelectedItem();
    if (user != null) {
        user.addMessage("[ " + user.getName() + " ] " + messageArea.getText());
        textArea.setText(user.getMessages());
        messageArea.setText("");
    }
}

public void selectUser(MouseEvent mouseEvent) {
    User selected = listView.getSelectionModel().getSelectedItem();
    if (selected != null) {
        textArea.setText(selected.getMessages());
    }
}
```

Не забуваємо стерти використання в коді

```
public static final String MAIN_UI = "/message.fxml";

@FXML public ListView<User> listView;
@FXML public TextField messageArea;
@FXML public Button sendBtn;

public void show(Stage stage) {
```

```
public void sendMessage(ActionEvent actionEvent) {
    User user = listView.getSelectionModel().getSelectedItem();
    if (user != null) {
        user.addMessage("[ " + user.getName() + " ] " + messageArea.getText());
        messageArea.setText("");
    }
}

public void selectUser(MouseEvent mouseEvent) {
    User selected = listView.getSelectionModel().getSelectedItem();
    if (selected != null) {
        (
    }
}
```

Оформлення чату

```
public class ChatMessage {

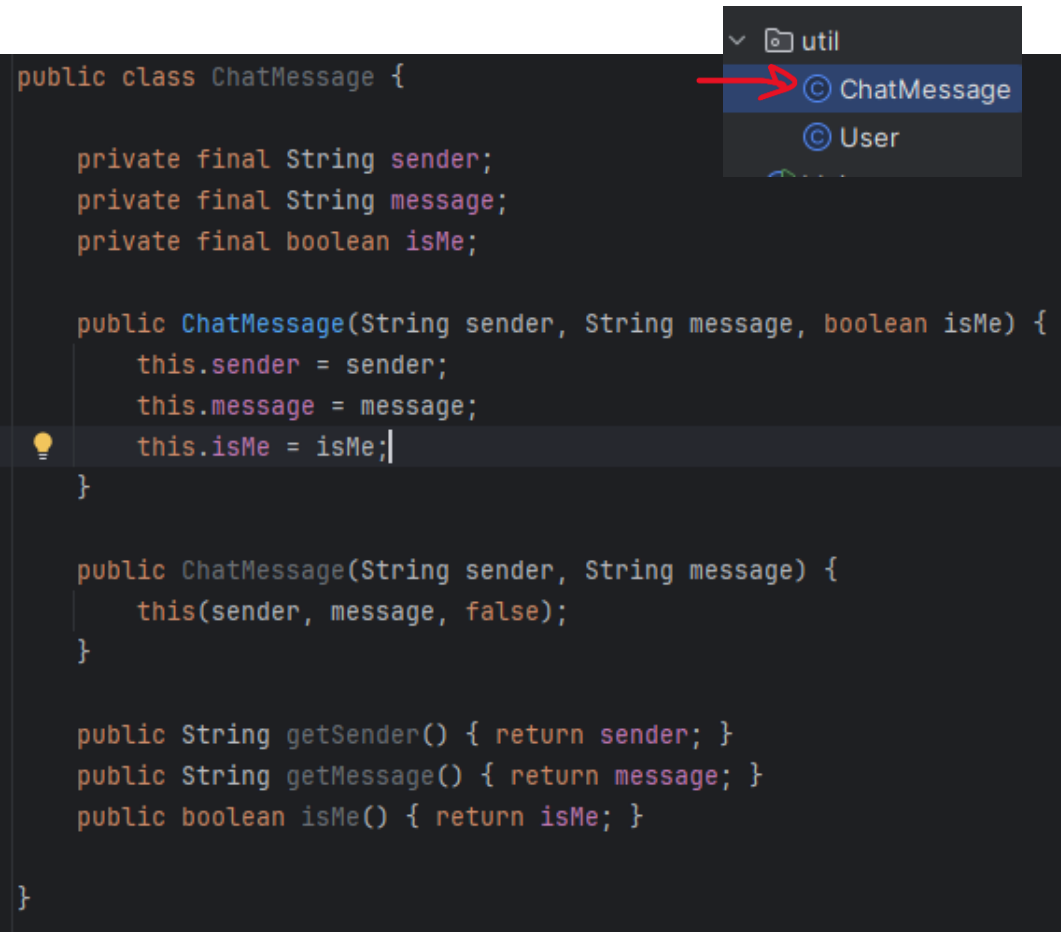
    private final String sender;
    private final String message;
    private final boolean isMe;

    public ChatMessage(String sender, String message, boolean isMe) {
        this.sender = sender;
        this.message = message;
        this.isMe = isMe;
    }

    public ChatMessage(String sender, String message) {
        this(sender, message, false);
    }

    public String getSender() { return sender; }
    public String getMessage() { return message; }
    public boolean isMe() { return isMe; }

}
```



Додамо повідомлення. У нього буде відправник (якого доробимо пізніше), саме повідомлення, та вказівник чи це ми відправили.

```
public class MessageBubble extends VBox {

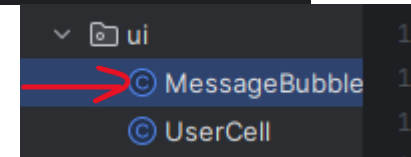
    public MessageBubble(ChatMessage msg) {
        // Відправник
        Text senderText = new Text(msg.getSender() + "\n");
        senderText.setStyle(
            "-fx-fill: black;" +
            "-fx-font-size: 12;" +
            "-fx-font-weight: bold;"
        );

        // Відступ від відправника
        Text spacer = new Text("\n");
        spacer.setStyle("-fx-font-size: 4;");

        // Повідомлення
        Text messageText = new Text(msg.getMessage());
    }

}
```

Саме повідомлення, яке буде у панелі
Коду багато, показати файл



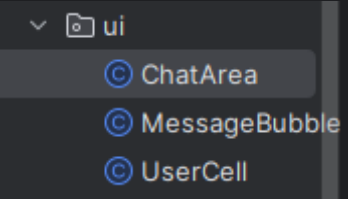
Оформлення чату

```
public class ChatArea extends ScrollPane {

    private final VBox container;

    public ChatArea() {
        container = new VBox(5); // 5 - піксельні відступи між елементами
        container.setPadding(new Insets(10)); // внутрішні відступи (всі по 10 px)
        this.setContent(container); // встановити основною панелькою VBox
        this.setFitToWidth(true); // розтягувати на всю ширину
        this.setVbarPolicy(ScrollBarPolicy.ALWAYS); // завжди є смуга прокручення
    }

    public void addMessage(ChatMessage msg) {
        MessageBubble bubble = new MessageBubble(msg);
        bubble.maxWidthProperty().bind( // зменшення ширини через смугу прокручення
            container.widthProperty().subtract(20)
        );
        container.getChildren().add(bubble); // додати в список повідомлення
        this.layout(); // змусити оновитися панельку
        this.setVvalue(1.0); // прокрутити в самий низ
    }
}
```



- Сама панель, яку будемо додавати в коді.

```
</> @FXML public ListView<User> listView;
</> @FXML public TextField messageArea;
</> @FXML public Button sendBtn;
public ChatArea chatArea;
```

```
@FXML
public void initialize() {
    listView.setCellFactory(info -> new UserCell());
    listView.getItems().addAll(
        new User("Serhii"),
        new User("Andrii"),
        new User("Taras"),
        new User("Shush")
    );

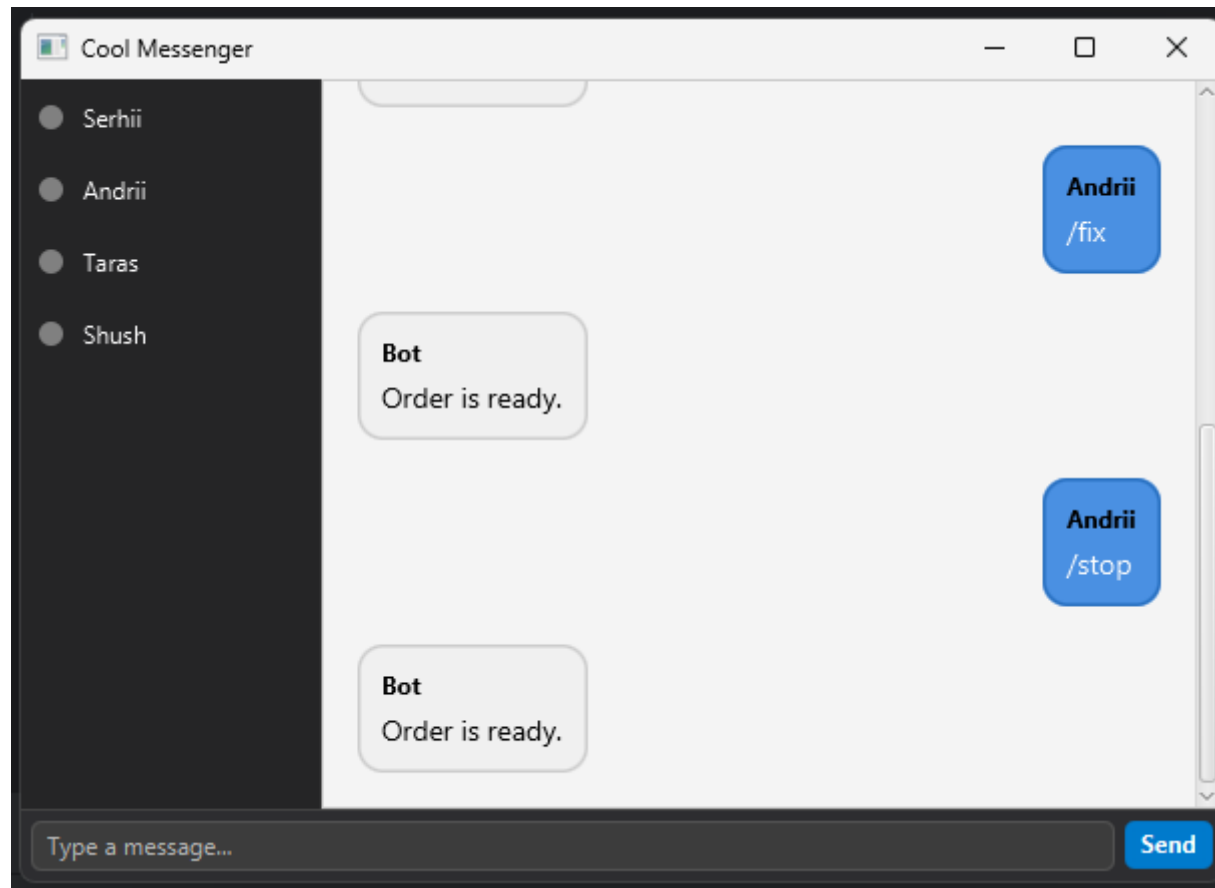
    chatArea = new ChatArea();
    if (listView.getParent() instanceof BorderPane pane) {
        pane.setCenter(chatArea);
    }

    chatArea.addMessage(new ChatMessage("Andrii", "Hello!", true));
    chatArea.addMessage(new ChatMessage("Bot", "Order is ready."));

    chatArea.addMessage(new ChatMessage("Andrii", "Thanks!", true));
    chatArea.addMessage(new ChatMessage("Andrii", "/fix", true));
}
```

Тестові повідомлення

Стилізація



Як бачимо, повідомлення чудово видно
Але вони не міняються від чату до чату(

Зміна чатів

Щоб змінювати чати, додамо до кожного користувача замість String з повідомленнями – список з повідомленнями.

```
public class User {
    private String name;
    private List<MessageBubble> messages;
    private boolean isOnline;

    public User(String name) {
        this.name = name;
        this.messages = new ArrayList<>();
        this.isOnline = false;
    }

    public String getName() { return name; }
    public void setName(String name) { this.name = name; }

    public List<MessageBubble> getMessages() { return messages; }
    public void addMessage(MessageBubble bubble) { messages.add(bubble); }
    public void addMessage(ChatMessage msg) { messages.add(new MessageBubble(msg)); }

    public void setOnline(boolean isOnline) { this.isOnline = isOnline; }
    public boolean isOnline() { return this.isOnline; }
}
```

Тепер в метод кліку по користувачу додамо завантаження повідомлень. Але спочатку – зручні методи для **ChatArea** (на майбутнє).

Уважно подивіться кожен рядок, всюди зміни!

```
public void addMessage(MessageBubble bubble) {
    container.getChildren().add(bubble); // додати в список повідомлень
    this.layout(); // змусити оновитися панельку
    this.setVvalue(1.0); // прокрутити в самий низ
}

public MessageBubble genMessage(ChatMessage msg) {
    MessageBubble bubble = new MessageBubble(msg);
    bubble.maxWidthProperty().bind( // зменшення ширини через смугу прокрутки
        container.widthProperty().subtract(20)
    );
    return bubble;
}

public void loadAllMessages(User user) {
    container.getChildren().clear();
    for (MessageBubble bubble: user.getMessages()) {
        bubble.maxWidthProperty().bind(
            container.widthProperty().subtract(20)
        );
        container.getChildren().add(bubble);
    }
    this.layout();
    this.setVvalue(1.0);
}
```


Зміна чатів

```
@FXML
public void initialize() {
    listView.setCellFactory(info -> new UserCell());
    listView.getItems().addAll(
        new User("Serhii"),
        new User("Andrii"),
        new User("Taras"),
        new User("Shush")
    );

    chatArea = new ChatArea();
    if (listView.getParent() instanceof BorderPane pane) {
        pane.setCenter(chatArea);
    }
}
```

messengerUI.java

```
public void selectUser(MouseEvent mouseEvent) {
    User selected = listView.getSelectionModel().getSelectedItem();
    if (selected != null) {
        chatArea.loadAllMessages(selected);
    }
}
```

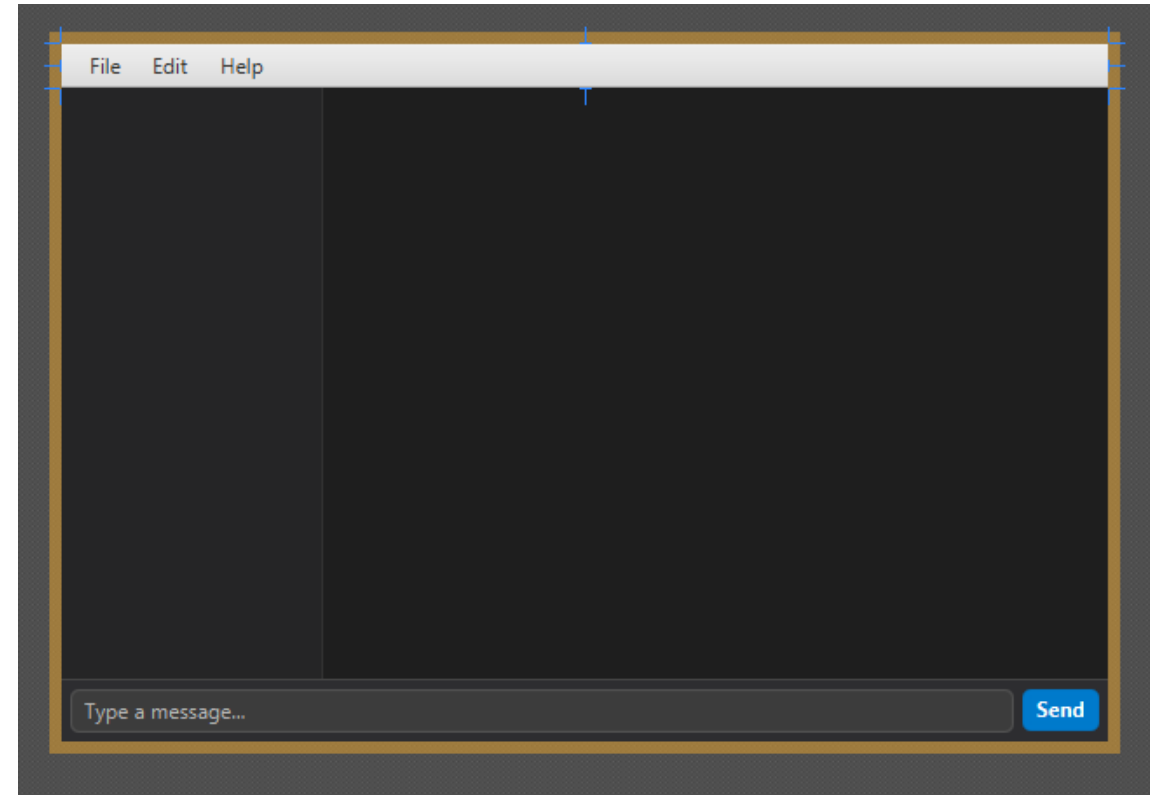
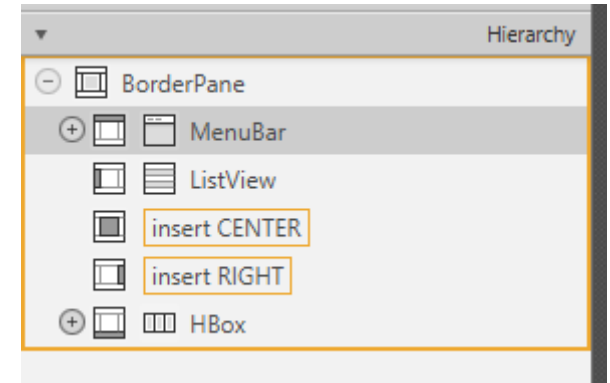
```
public void sendMessage(ActionEvent actionEvent) {
    User user = listView.getSelectionModel().getSelectedItem();
    if (user != null) {
        ChatMessage msg = new ChatMessage("Me", messageArea.getText(), true);
        MessageBubble bubble = chatArea.genMessage(msg);
        user.addMessage(bubble);
        chatArea.addMessage(bubble);
    }
}
```

Операції з юзерами

Тепер повідомлення можемо писати кому захочемо, але немає збереження цих самих повідомлень, додавання користувачів та власного імені (а може ще й тегу @).

Все це буде відбуватися на сервері і потім синхронізуватися. Ну крім додавання користувачів (кнопку треба все ж додати).

Зробити можна кількома шляхами – додати зверху меню, додати першим у списку користувача «+» (який відкриватиме вікно). Краще звичайно ж перший, бо можна потім додати збереження чатів локально (але виникатиме контраверсія з сервером).



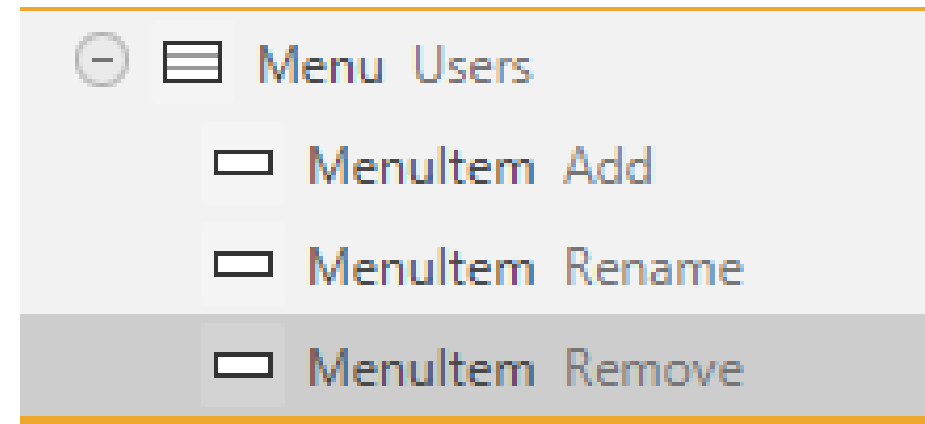
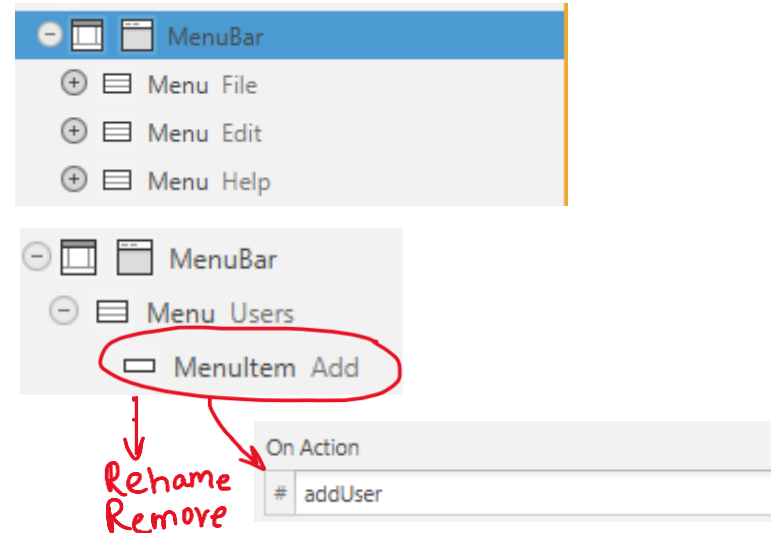
MenuBar

Це стрічка для меню, такі є в стандартних програмах Windows. Свою ж можна налаштувати, видати стилі та інші пункти.

Ми залишаємо в меню лише 1 компонент меню, перейменовуємо в Users, а всередині у нас буде кнопка Add, Rename, Remove.

Уважно дивіться, як розкрито на кнопку «+» компоненти!

У кожну кнопку додати onAction.



MenuBar в програмі

Додаємо відгуки на кнопки, а після цього можливість зчитати того, кого хочемо додати.

Це відбувається через готове модальне вікно `TextInputDialog`. Він дозволяє просто вказати що саме ввести, назву вікна і тд. Функція `.showAndWait()` чекає доки щось введеться.

Іноді інформації може й не бути – для цього існує клас `Optional<>`, який зберігає результат і має функції для зрозумілішої роботи.

```
<MenuItem mnemonicParsing="false" onAction="#addUser" text="Add" />
<MenuItem mnemonicParsing="false" onAction="#renameUser" text="Rename" />
<MenuItem mnemonicParsing="false" onAction="#removeUser" text="Remove" />
```

```
public void addUser(ActionEvent actionEvent) {

}

public void renameUser(ActionEvent actionEvent) {

}

public void removeUser(ActionEvent actionEvent) {

}
```

```
public static String getInput(String prompt) {
    TextInputDialog dialog = new TextInputDialog();
    dialog.setTitle("Input");
    dialog.setHeaderText(null);
    dialog.setContentText(prompt);

    Optional<String> result = dialog.showAndWait();
    return result.orElse(null);
}
```

```
public void addUser(ActionEvent actionEvent) {
    String input = getInput("Enter username");
    if (input == null) return;

    // System.out.println(input);
}
```

```
public void renameUser(ActionEvent actionEvent) {
    String input = getInput("Enter username");
    if (input == null) return;

    // System.out.println(input);
}
```

```
public void removeUser(ActionEvent actionEvent) {
    String input = getInput("Enter username");
    if (input == null) return;

    // System.out.println(input);
}
```

Додавання нікнейму

```
8 public class User {
9     private String name;
10    private String nickname;
11    private List<MessageBubble> messages;
12    private boolean isOnline;

    public User(String name, String nickname) {
        this.name = name;
        this.nickname = nickname;
        this.messages = new ArrayList<>();
        this.isOnline = false;
    }
```

```
22
23 public String getNickname() { return nickname; }
24 public void setNickname(String nickname) { this.nickname = nickname; }
25
```

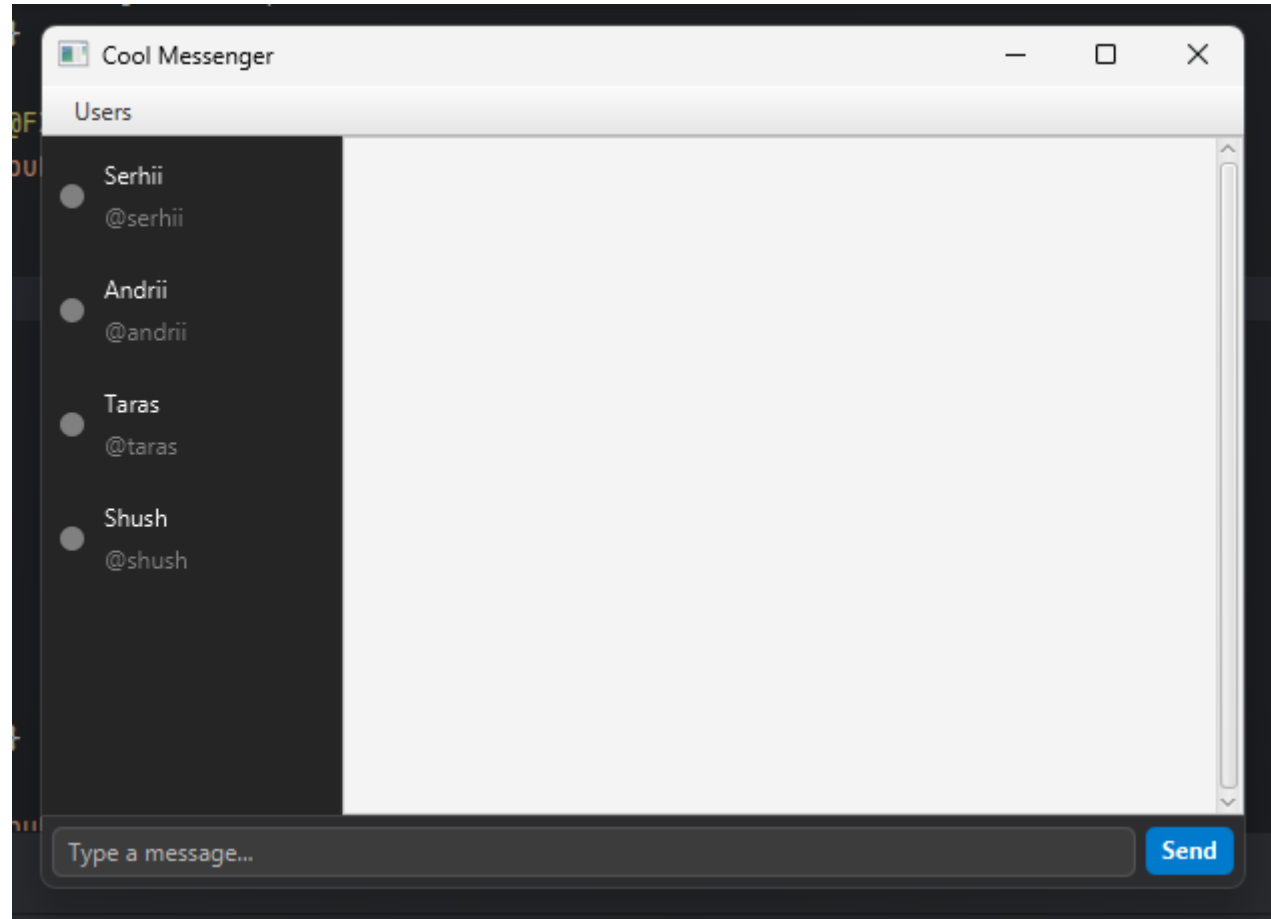
```
@FXML
public void initialize() {
    listView.setCellFactory(info -> new UserCell());
    listView.getItems().addAll(
        new User("Serhii", "@serhii"),
        new User("Andrii", "@andrii"),
        new User("Taras", "@taras"),
        new User("Shush", "@shush")
    );
}
```

```
33
34 @Override
35 protected void updateItem(User user, boolean empty) {
36     super.updateItem(user, empty);
37     if (empty || user == null) {
38         setGraphic(null);
39     } else {
40         nameText.setText(user.getName());
41         nicknameText.setText(user.getNickname());
42         statusDot.setFill(user.isOnline() ? Color.LIMEGREEN : Color.GRAY);
43         setGraphic(content);
    }
```

```
12 public class UserCell extends ListCell<User> {
13
14     private final HBox content;
15     private final Circle statusDot;
16     private final VBox textContainer;
17     private final Text nameText;
18     private final Text nicknameText;
19
20     public UserCell() {
21         statusDot = new Circle(6);
22
23         nameText = new Text();
24         nameText.setFill(Color.WHITE);
25         nicknameText = new Text();
26         nicknameText.setFill(Color.GRAY);
27         textContainer = new VBox(5, nameText, nicknameText);
28
29         content = new HBox(10, statusDot, textContainer);
30         content.setAlignment(Pos.CENTER_LEFT);
31     }
```

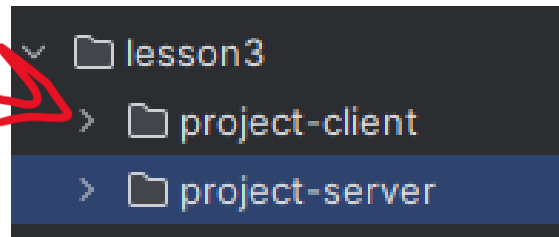
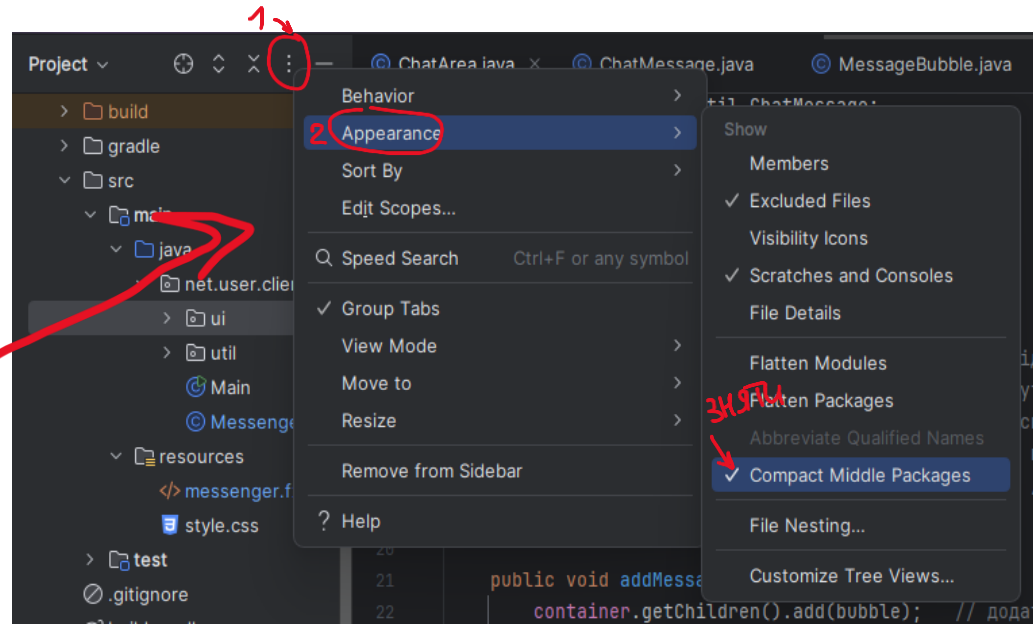
Результат

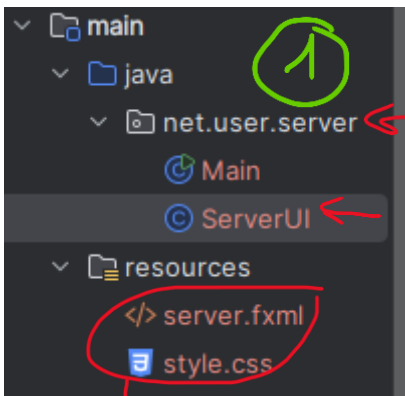
- Тепер у нас є все, щоб продовжувати серверну частину.



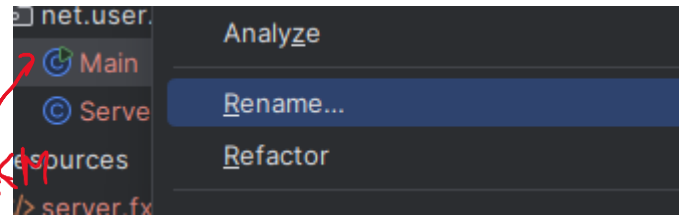
Сервер

- Поки що почнемо робити сам сервер – додамо йому просте вікно, можливості робити команди і список людей які підключилися.
- Налаштуємо в ІДЕ, щоб нам показувалися проміжні пакети, а не їх скорочена версія.
- **Копіюємо проект клієнта в project-server.** Тобто в нас вийде 2 окремих, але схожих проєкти.
- Переіменовуємо та додаємо необхідні пакети та класи.





• Перейменування



• Це попросити у викладача

• Програми

```
public class Main extends Application {  
  
    @Override  
    public void start(Stage primaryStage) {  
        ServerUI ui = new ServerUI();  
        ui.show(primaryStage);  
    }  
  
    public static void main(String[] args) {  
        launch(args);  
    }  
}
```

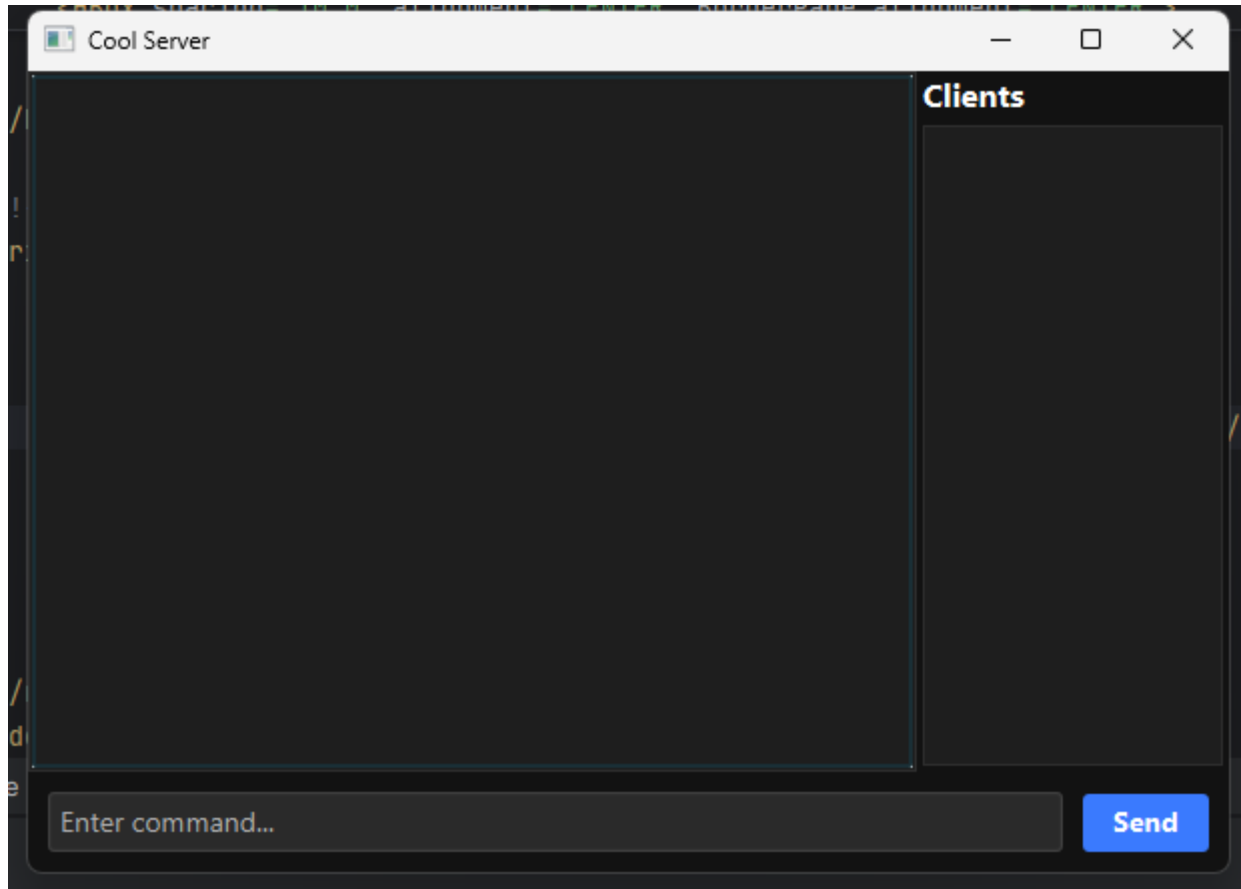
```
public class ServerUI {  
  
    public static final String MAIN_UI = "/server.fxml";  
  
    public void show(Stage stage) {  
        FXMLLoader loader = new FXMLLoader(getClass().getResource(MAIN_UI));  
        Parent root;  
        try {  
            root = loader.load();  
        } catch (IOException e) {  
            throw new RuntimeException(e);  
        }  
  
        // Scene and stage setup  
        Scene scene = new Scene(root, 600, 400);  
        stage.setTitle("Cool Server");  
        stage.setScene(scene);  
        stage.setMinWidth(400);  
        stage.setMinHeight(300);  
        stage.show();  
    }  
  
    @FXML  
    public void initialize() {  
  
    }  
}
```

```
application {  
    mainClass = 'net.user.server.Main' // Клас, який запустить пр  
}
```

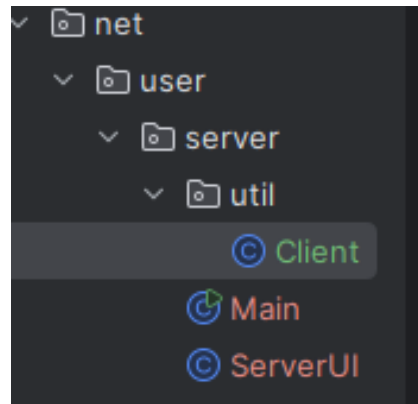
gradle.build +
перезавантажити

3

- Ось таке має вийти в результаті



- І ще додамо клас для підключеного користувача.
- Самий простий пробний варіант – далі буде БД.



```
public class Client {

    private final String ipAddress;
    private final int port;
    private String nickname;
    private String password;

    public Client(String ipAddress, int port, String nickname, String password) {
        this.ipAddress = ipAddress;
        this.port = port;
        this.nickname = nickname;
        this.password = password;
    }

    public String getIpAddress() { return ipAddress; }
    public int getPort() { return port; }
    public String getNickname() { return nickname; }
    public String getPassword() { return password; }

    public void setNickname(String nickname) { this.nickname = nickname; }
    public void setPassword(String password) { this.password = password; }

}
```

дописати

```
<BorderPane xmlns="http://javafx.com/javafx/23.0.1"
    xmlns:fx="http://javafx.com/fxml/1"
    fx:controller="net.user.server.ServerUI"
    stylesheets="@style.css">
```

.fxml

```
public class ServerUI {  
  
    public static final String MAIN_UI = "/server.fxml";
```

```
@FXML public TextArea consoleArea;  
@FXML public TextField commandField;  
@FXML public Button sendButton;  
@FXML public ListView<String> clientList;
```

```
    public void show(Stage stage) {  
        FXMLLoader loader = new FXMLLoader(getCl  
I));
```

```
@@ -32,7 +41,11 @@ public class ServerUI {
```

```
@FXML  
    public void initialize() {
```

```
        clientList.getItems().addAll(  
            "@illias",  
            "@archasmiel",  
            "@hubris"  
        );
```

```
    }
```

