Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №5

По дисциплине: «Современные платформы программирования»

Выполнила:
Студентка 3 курса
Группы ПО-6
Юсковец М.А.
Проверил:
Монтик Н.С.

Брест, 2023

Цель работы: освоить приемы разработки оконных клиент-серверных приложений на Java с использованием сокетов.

Ход работы:

Задание:

Вариант 25

Игра «Быки и коровы» Первый игрок задумывает четырехзначное число, так чтобы все цифры числа были разные. Цель второго игрока – отгадать это число. Каждый ход, отгадывающий называет число, тоже четырехзначное и с разными цифрами. Если цифра из называемого числа есть в отгадываемом числе, то эта ситуация называется корова. Если цифра из называемого числа есть в отгадываемом числе и стоит на том же месте, то эта ситуация называется бык. Информация о количестве быков и коров открытая. Дается фиксированное количество попыток. Например, для загаданного числа 0475 называемое число 0251 содержит 1 быка и 1 корову.

Текст программы:

Server:

Main.java

```
package com.example.spp_lab5_server;

import java.io.IOException;
import java.net.ServerSocket;

public class Main {
    public static void main(String[] args) throws IOException {
        ServerSocket serverSocket = new ServerSocket(8080);
        new GameHandler(new PlayerThread(serverSocket.accept()), new
PlayerThread(serverSocket.accept())).start();
        System.out.println("i am here");
    }
}
```

MessageListener.java

```
package com.example.spp_lab5_server;

public interface MessageListener {
    void processMessage(String message);
}
```

GameHandler.java

```
package com.example.spp_lab5_server;

import java.util.Random;
```

```
public class GameHandler implements MessageListener {

    private PlayerThread numberGuesser;
    private PlayerThread numberPicker;
    private static final int maxGuessCount = 10;
    private short guessCount;
    private String number;

    public GameHandler(PlayerThread firstPlayer, PlayerThread secondPlayer) {
        firstPlayer.setMessageListener(this);
        secondPlayer.setMessageListener(this);
        boolean isFirstPlayerGuesser = new Random().nextBoolean();
        numberGuesser = isFirstPlayerGuesser ? firstPlayer : secondPlayer;
        numberPicker = isFirstPlayerGuesser ? secondPlayer : firstPlayer;
    }

    public void start() {
        numberGuesser.start();
        numberPicker.start();
        numberPicker.sendMessage("Выберите число");
    }

    private void restartGame() {
        guessCount = 0;
        number = null;
        PlayerThread numberGuesserTmp = numberGuesser;
        numberGuesser = numberPicker;
        numberPicker = numberGuesserTmp;
        numberPicker.sendMessage("Выберите число");
    }

    @Override
    public void processMessage(String message) {
        if (number == null) {
            number = message;
            numberGuesser.sendMessage("Угадайте число");
            return;
        }

        if (++guessCount == maxGuessCount && !message.equals(number)) {
            numberGuesser.sendMessage("Проигрыш");
            numberPicker.sendMessage("Противник проиграл");
            restartGame();
            return;
        }

        if (message.equals(number)) {
            numberGuesser.sendMessage("Победа");
            numberPicker.sendMessage("Проивник победил");
            restartGame();
            return;
        }

        int bullCount = 0;
        int cowCount = 0;

        for (int i = 0; i < message.length(); i++) {
            if (number.charAt(i) == message.charAt(i)) {
                bullCount++;
            }
            else if (number.contains(Character.toString(message.charAt(i))))
{
                cowCount++;
```

```
            }
        }

        String messageToSend = "Попытка " + guessCount + ". Быки: " +
bullCount + ". Коровы: " + cowCount;
        numberGuesser.sendMessage(messageToSend);
        numberPicker.sendMessage(messageToSend);
    }
}
```

PlayerThread.java

```
package com.example.spp_lab5_server;

import java.io.*;
import java.net.Socket;

public class PlayerThread extends Thread {

    private final BufferedReader reader;
    private final BufferedWriter writer;
    private MessageListener messageListener;

    public PlayerThread(Socket socket) throws IOException {
        reader = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
        writer = new BufferedWriter(new
OutputStreamWriter(socket.getOutputStream()));
    }

    public void setMessageListener(MessageListener messageListener) {
        this.messageListener = messageListener;
    }

    @Override
    public void run() {
        try {
            while (true) {
                messageListener.processMessage(reader.readLine());
            }
        } catch (IOException ignored) {}
    }

    public void sendMessage(String message) {
        try {
            writer.write(message + "\n");
            writer.flush();
        } catch (IOException ignored) {}
    }
}
```

Client:

GameApplication.java

```
package com.example.spp_lab5_client;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Scene;
import javafx.stage.Stage;
```

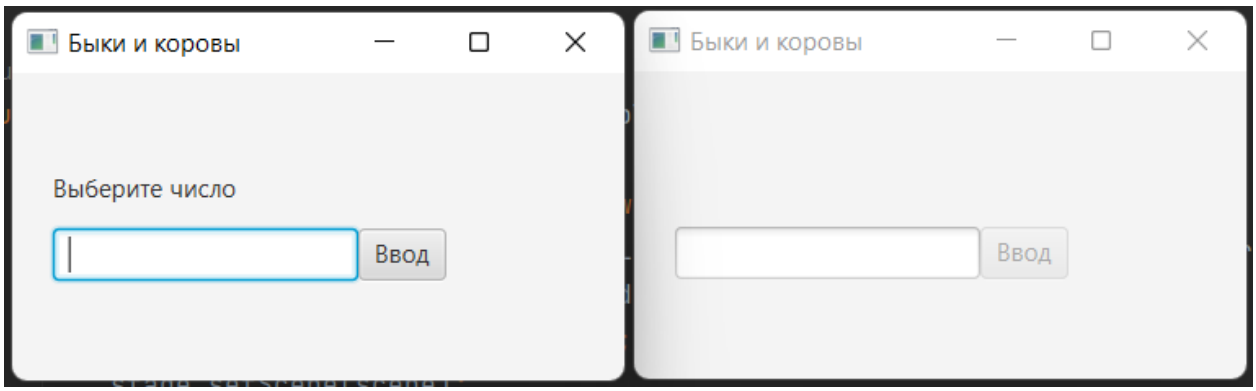
```
import java.io.IOException;

public class GameApplication extends Application {
    @Override
    public void start(Stage stage) throws IOException {
        FXMLLoader fxmlLoader = new
FXMLLoader(GameApplication.class.getResource("game-view.fxml"));
        Scene scene = new Scene(fxmlLoader.load(), 300, 150);
        stage.setTitle("Быки и коровы");
        stage.setScene(scene);
        stage.show();
    }

    public static void main(String[] args) {
        launch();
    }
}
```

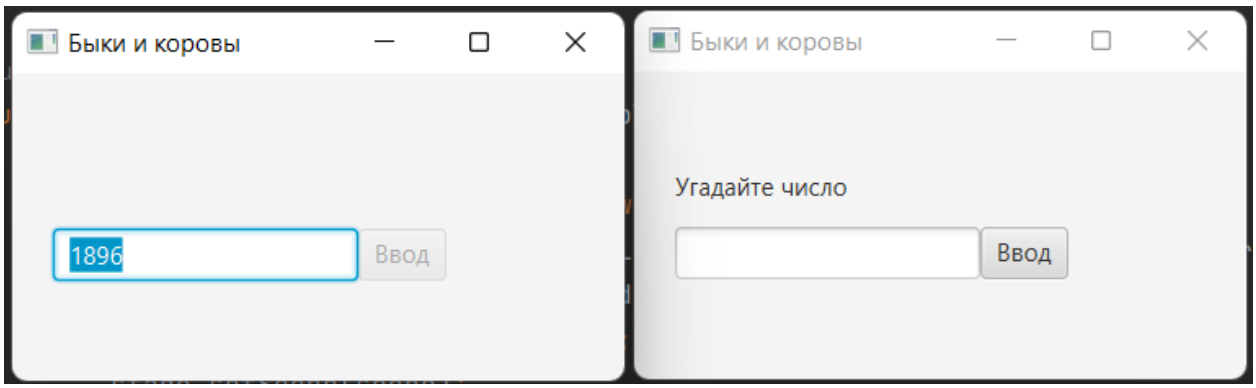
GameController.java

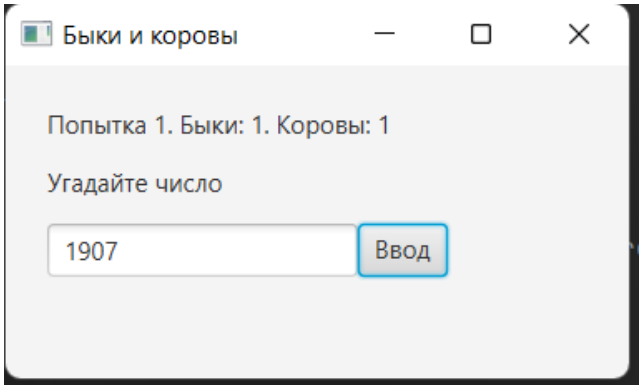
```
package com.example.spp lab5 client;

import javafx.application.Platform;
import javafx.fxml.FXML;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.TextField;

import java.io.*;
import java.net.Socket;
import java.util.HashSet;
import java.util.List;

public class GameController {
    @FXML
    private Label result;
    @FXML
    private Label command;
    @FXML
    private TextField numberInput;
    @FXML
    private Button enterButton;

    private BufferedWriter writer;
    private boolean isGuesser;

    public GameController() {
        new Thread(() -> {
            try {
                Socket socket = new Socket("localhost", 8080);
                BufferedReader reader = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
                writer = new BufferedWriter(new
OutputStreamWriter(socket.getOutputStream()));

                while (true) {
                    String str = reader.readLine();

                    if (str.equals("Угадайте число")) {
                        isGuesser = true;
                    }
```

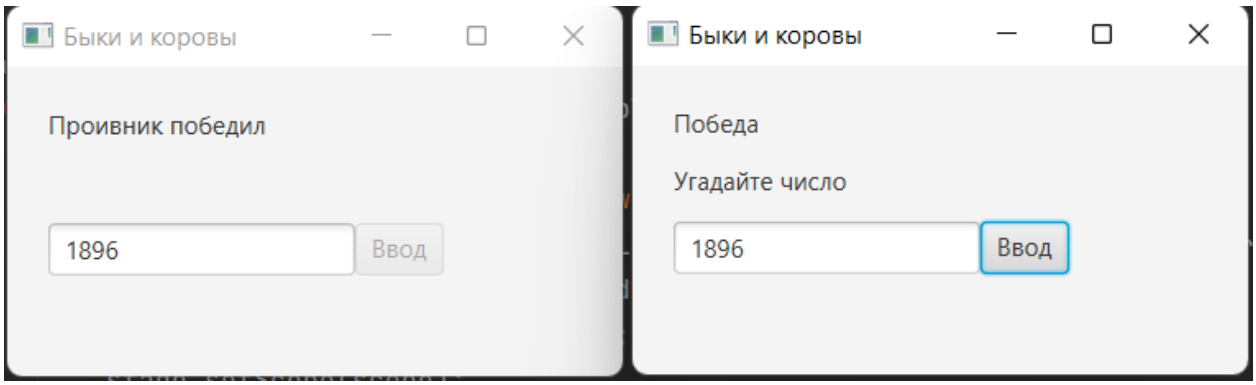
```
                    if (str.equals("Выберите число") || str.equals("Угадайте
число")) {
                        Platform.runLater(() -> command.setText(str));
                        Platform.runLater(() ->
enterButton.setDisable(false));
                        continue;
                    }

                    Platform.runLater(() -> result.setText(str));

                    if (!str.startsWith("Попытка")) {
                        Thread.sleep(5000);
                        isGuesser = false;
                        Platform.runLater(() -> result.setText(""));
                    }
                }
            } catch (IOException | InterruptedException ignored) { }
        }).start();
    }

    @FXML
    protected void onClick() throws IOException {
        List<Character> characters =
numberInput.getText().chars().mapToObj(character -> (char)
character).toList();
        if (characters.size() != 4 || new HashSet<>(characters).size() != 4
|| !characters.stream().allMatch(Character::isDigit)) {
            command.setText("Неверный ввод. Попробуйте еще раз");
            return;
        }

        writer.write(numberInput.getText() + "\n");
        writer.flush();

        if (!isGuesser) {
            command.setText("");
            enterButton.setDisable(true);
            return;
        }

        command.setText("Угадайте число");
    }
}
```

Результат программы:









Вывод: освоили приемы разработки оконных клиент-серверных приложений на Java с использованием сокетов.