МИНИТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра «Вычислительная техника»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

К курсовой работе

По курсу «Программирование на языке Java»

На тему «Разработка многомодульного приложения на языке Java»

Выполнил:

Студент группы 19ВВ2

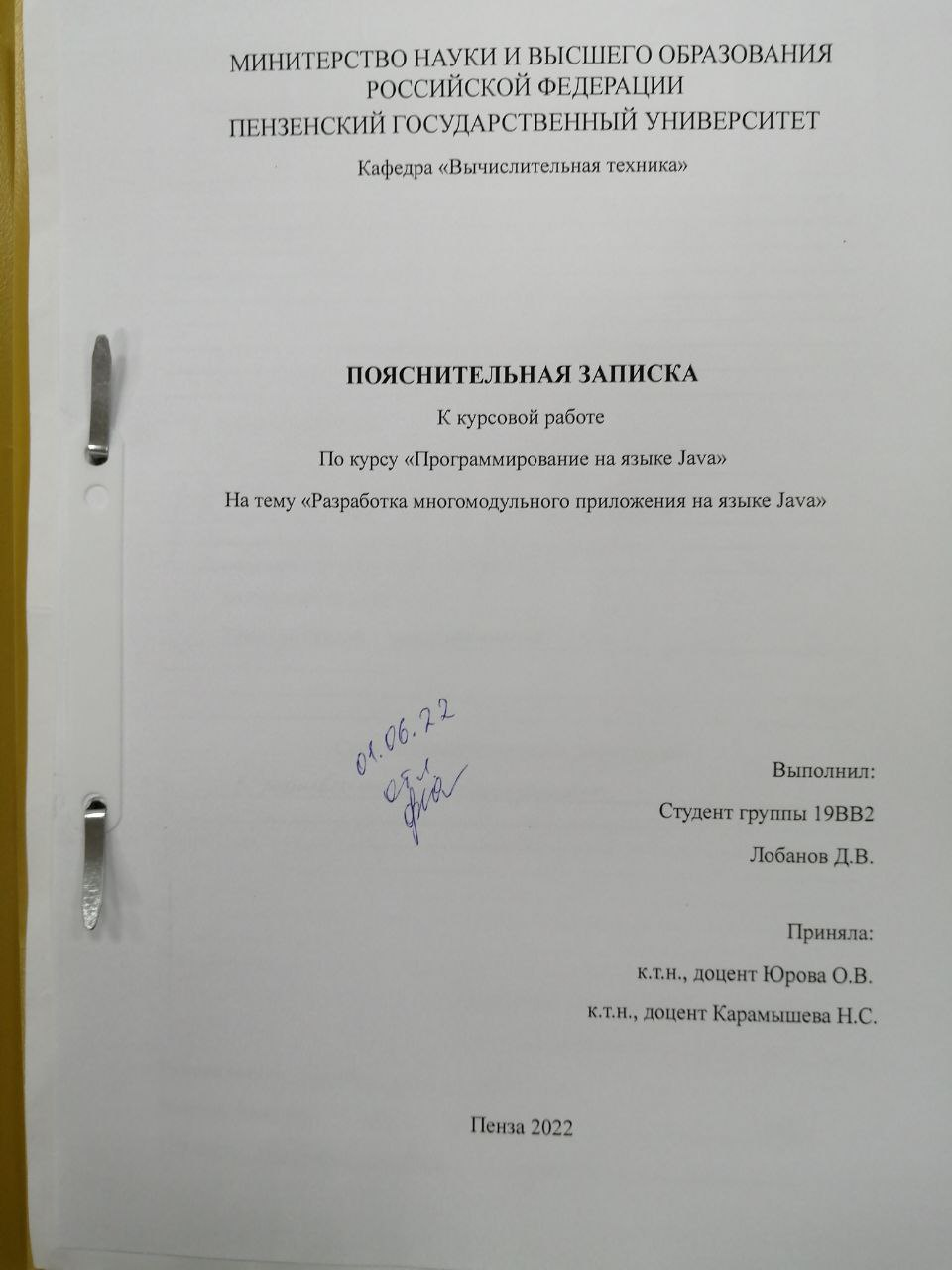
Лобанов Д.В.

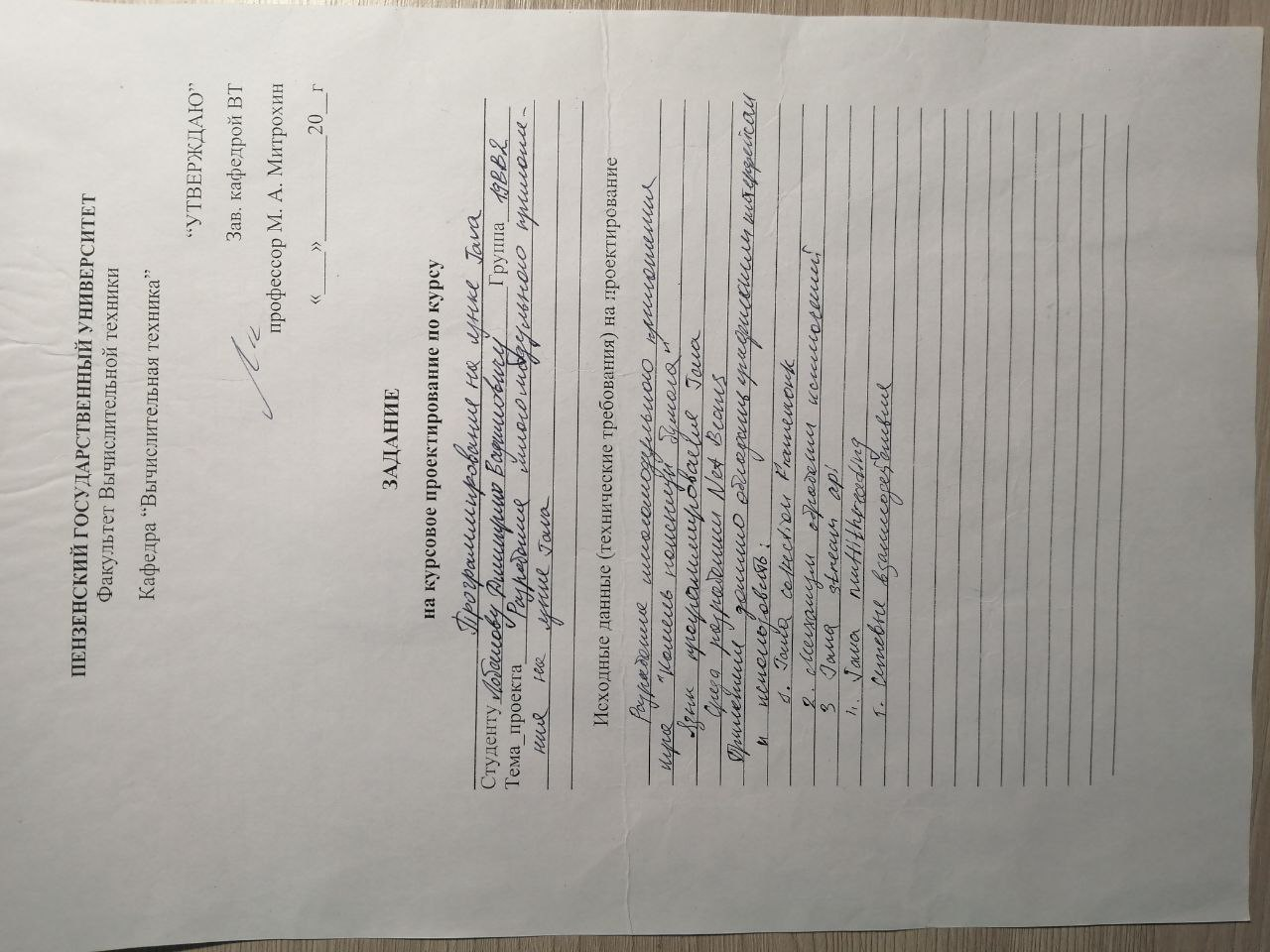
Приняла:

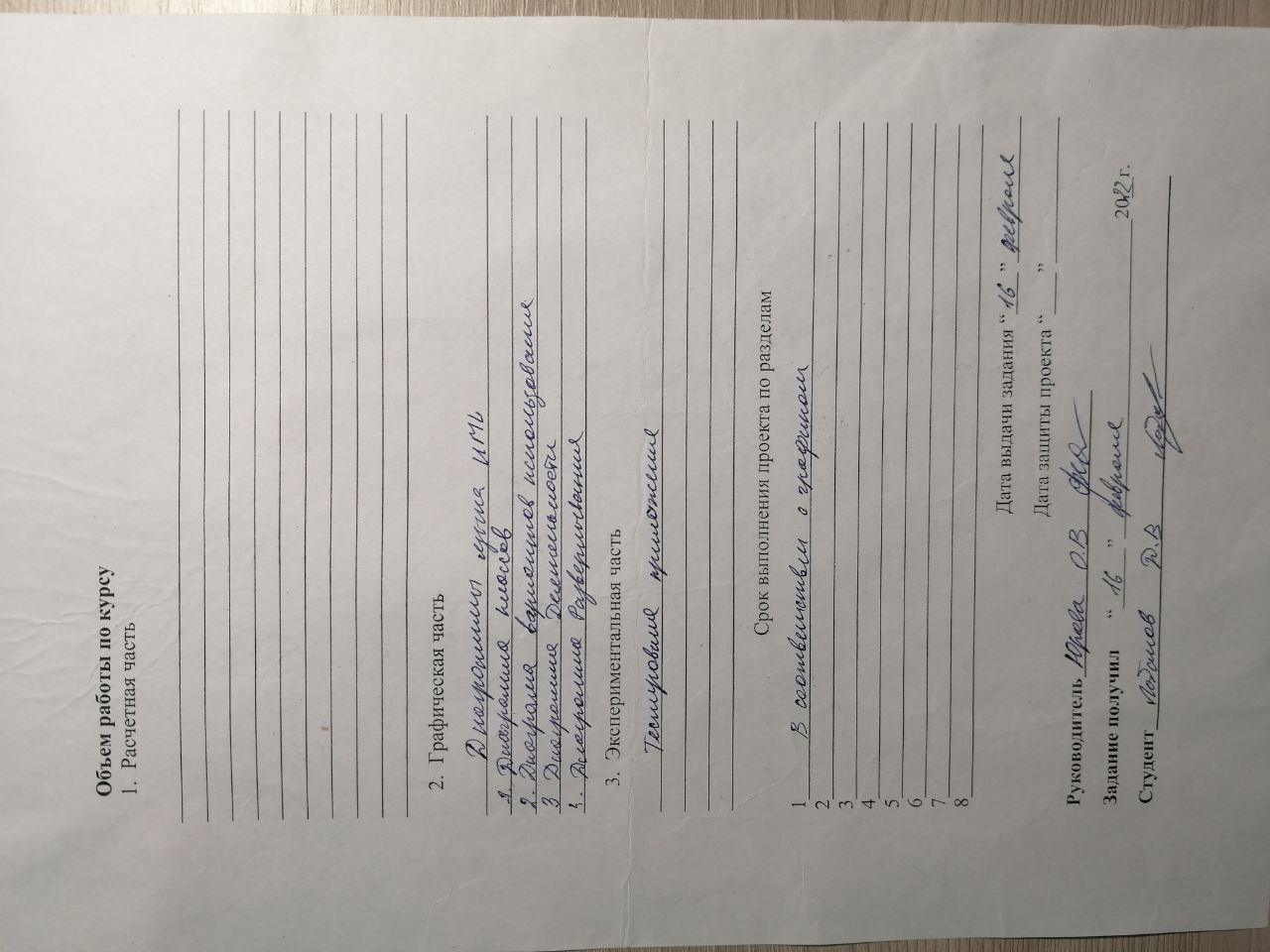
к.т.н., доцент Юрова О.В.

к.т.н., доцент Карамышева Н.С.

Пенза 2022







**Оглавление**

[Введение 3](#_Toc105008732)

[1. Постановка задачи 4](#_Toc105008733)

[2. Выбор решения 5](#_Toc105008734)

[3. Описание программы 6](#_Toc105008735)

[4. Экспериментальная часть 8](#_Toc105008736)

[5. Заключение 11](#_Toc105008737)

[Список используемой литературы. 12](#_Toc105008738)

[Приложение Б. Исходный код клиентской части 20](#_Toc105008739)

[Приложение Б.1 – MainframeClient.java 20](#_Toc105008740)

[Приложение В. UML-диаграммы приложения 31](#_Toc105008741)

[Приложение В.1 – UML-диаграмма вариантов использования 31](#_Toc105008742)

[Приложение В.2 – UML-диаграмма классов сервера 32](#_Toc105008743)

[Приложение В.3 – UML-диаграмма классов клиента 33](#_Toc105008744)

[Приложение В.4 – UML-диаграмма деятельности 34](#_Toc105008745)

[Приложение В.5 – UML-диаграмма развёртывания 35](#_Toc105008746)

[Приложение В.6 – UML-диаграмма последовательности 36](#_Toc105008747)

# Введение

Жизнь современного человека невозможно представить без ежедневного взаимодействия с компьютером. Сегодня люди с легкостью по всему миру могут обмениваться не только сообщениями, но и графикой и звуковой и другими видами информации.

Java – один из самых популярных языков программирования, с большим количеством достоинств:

1) Язык Java хорош для изучения объектно-ориентированного программирования.

2) Java имеет богатый API, и вы можете делать очень много в Java, например, работать с графикой, звуком и, что всего вероятнее, писать игры.

3) Java имеет сильную поддержку сообщества.

Такая простая игра, как «Камень, ножницы, бумага», отлично подходит для закрепления изученного в ходе курса материала.

# Постановка задачи

Разработать клиент-серверное приложение, реализующее игру «Камень, ножницы, бумага»

Функции сервера:

1) Обработка данных от пользователей.

2) Хранение результатов во время игры.

3) Отправка результата в соответствии с принятыми данными

4) Своевременное окончание игровой сессии.

Функции клиента:

1. Реализация интерфейса, и его изменение в соответствии с ответами сервера.
2. Отправка данных серверу

Приложение должно обладать графическим интерфейсом и использовать следующие технологии:

1. Java Collections Framework
2. Механизм обработки исключительных ситуаций
3. Java Stream API
4. Java Multithreading
5. Сетевое взаимодействие.

ОС – Windows. Язык программирования: Java. Среда разработки NetBeans.

# 2. Выбор решения

Важной частью курсовой работы является передача сообщений от клиента к серверу. Для реализации этого используются TCP-сокеты.

Сокет— это один конец двустороннего канала связи между двумя программами, работающими в сети. Соединив вместе два сокета, можно передавать данные между разными процессами (локальными или удаленными). Для сокетов жизнь дескриптора можно разделить на три фазы: открыть (создать) сокет, получить из сокета или отправить сокету и в конце концов закрыть сокет.

Обычно приложение клиент-сервер, использующее сокеты, состоит из двух разных приложений - клиента, инициирующего соединение с целью (сервером), и сервера, ожидающего соединения от клиента.

Клиент соединяется с сервером с помощью библиотеки *java.net.Socket*.

Интерфейс пользователя реализован с помощью библиотеки Java Swing.

# 3. Описание программы

Программа, состоит из клиентской и серверной части. Сначала запускается сервер, который входит в состояние ожидания подключения клиентов.

На других устройствах запускается клиентская часть, которая по нажатию определенной кнопки, пытается подключиться к серверу используя сокет.

**3.1 Серверная часть**

MainFrame.java – главный файл серверного процесса, котором находится интерфейс и точка входа в программу, функция Main, а также класс для реализации многопоточности ServerSomething.

run() – метод, через который происходит взаимодействие с клиентами. В зависимости от того, что сервер получает от клиентов (их выбор камень, ножницы или бумага), он определяет победителя и отправляет соответствующий ответ. Если количество побед одного становится равным трем, игра заканчивается, клиенты об этом оповещаются. Затем закрывается нить взаимодействия.

Вначале сервер ожидает подключения двух клиентов через сокет, когда это происходит он отправляет сообщение «found», только после этого начинается игра. Он принимает сообщение «rock», «paper» или «scissor», и отправляет в ответ «win» или «lose», когда переменная Count1 или Count2 принимает значение равное трем, то сервер отправляет сообщение «endgame», после этого игра заканчивается.

**3.2 Клиентская часть.**

MainframeClient.java – главный файл пользовательского процесса, содержащий функцию Main. Также реализует интерфейс.

gameRockChooserMouseClicked(java.awt.event.MouseEvent evt) – обеспечивает корректное отображение выбора клиента. Картинки Rock.png.

gamePaperChooserMouseClicked(java.awt.event.MouseEvent evt) – обеспечивает корректное отображение выбора клиента. Картинки Paper.png.

gameScissorsChooserMouseClicked(java.awt.event.MouseEvent evt) – обеспечивает корректное отображение выбора клиента. Картинки Scissors.png.

mainMenuStartButtonActionPerformed(java.awt.event.ActionEvent evt) – функция нажатия кнопки «Старт», которая пытается присоединиться к серверу. В случае успеха меняет панель главного меню на игровую панель.

gameSendButtonActionPerformed(java.awt.event.ActionEvent evt) – функция, которая отправляет на сервер выбор клиента, и которая осуществляет изменение интерфейса в зависимости от результата раунда.

RulesButtonMouseClicked(java.awt.event.MouseEvent evt) – кнопка вызова всплывающего окна с правилами.

jButton1ActionPerformed(java.awt.event.ActionEvent evt) – кнопка выхода из программы.

Клиенты отправляют на сервер «rock», «paper» или «scissor», а получают в ответ «win» или «lose», в зависимости от этого изменяется интерфейс. Очки начисляют либо себе либо противнику. Как только клиент получает сообщение «endgame» это означает что один из игроков набрал 3 очка, а следовательно победил. После этого игра заканчивается и панель игра сменяется обратно на главное меню.

# 4. Экспериментальная часть

Для начала запускаем сервер.

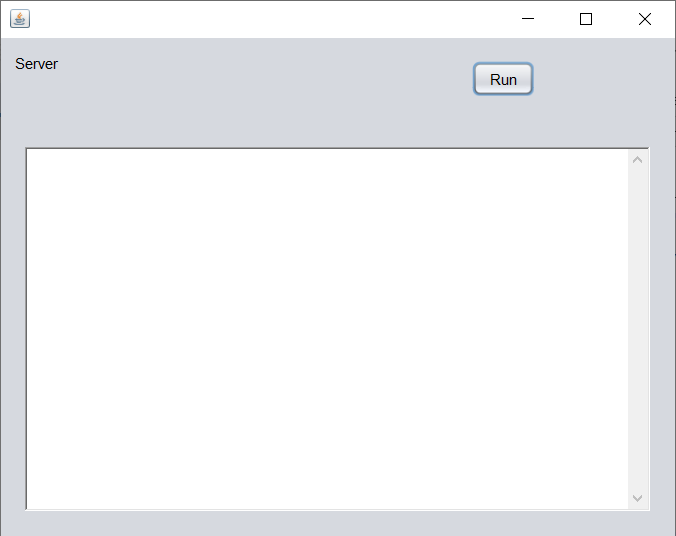


Рисунок – 1 Запуск сервера.

Далее запускаем клиента.

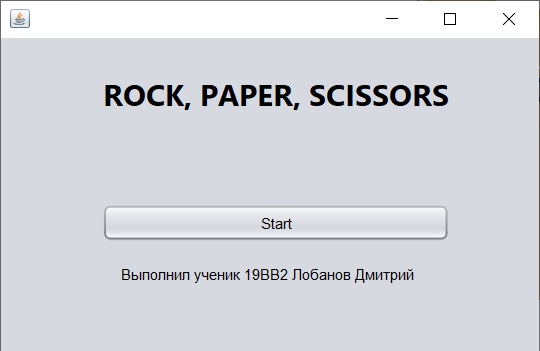


Рисунок – 2 Запуск Клиента.

После нажатия кнопки «Старт», происходит успешное подключение к серверу, значит второй клиент тоже смог подсоединиться. Открывается игровое поле.

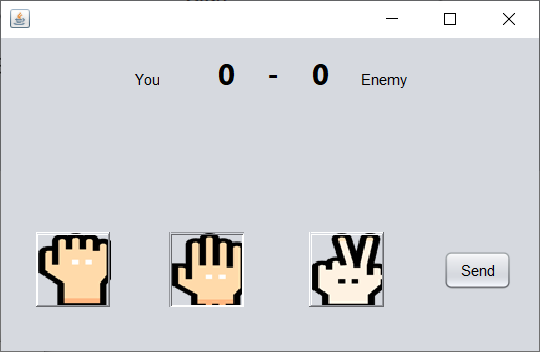


Рисунок – 3 Игровое поле.

Отправляем наш выбор с помощью кнопки «Send», до тех пор пока один из клиентов не наберет три очка. После этого игра заканчивается.

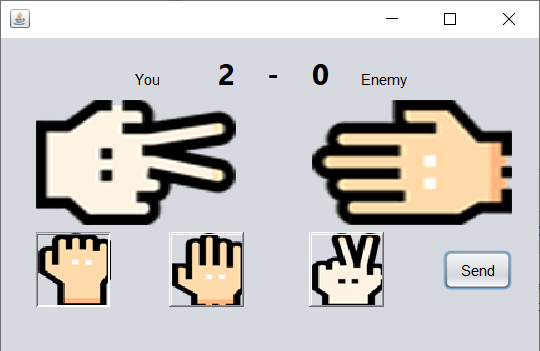


Рисунок – 4 Игра.

# 5. Заключение

При выполнении данной курсовой работы были получены навыки разработки программ на языке высокого уровня Java. Были освоены навыки разработки графического интерфейса с помощью библиотеки Swing. Изучены возможности среды разработки NetBeans. Освоены навыки разработки клиент-серверного приложения.

В рамках данной работы было написано приложение «Камень, ножницы, бумага».

Данная реализация не лишена недочетов в визуальной части и оптимизации кода, поэтому ее можно улучшить. Также есть возможность расширения функционала, например, добавление истории игр, рейтинга, результатов игры и отделение графической от логической частей.

# Список используемой литературы.

1. Берд, Барри Java для чайников / Барри Берд. - М.: Диалектика / Вильямс, 2013. - 521 c.

2. Гурвиц Г. Разработка реального приложения в среде клиент-сервер –

ДВГУПС, 2005, 206 с.

3. Дубаков А.А. Сетевое программирование: учебное пособие / А.А. Дубаков – СП: НИУ ИТМО, 2013. – 248 с.

4. Мейер Б. Объектно-ориентированное программирование и программная инженерия / Мейер Б.— Электрон. текстовые данные.— М.: Интернет Университет Информационных Технологий (ИНТУИТ), 2016.— 285с.

##### **Приложение А. Исходный код серверной части.**

**Приложение А.1 - Mainframe.java**

import java.net.\*;

import java.util.\*;

import java.io.\*;

import java.nio.\*;

import java.nio.channels.\*;

import java.nio.charset.\*;

import java.util.logging.Level;

import java.util.logging.Logger;

public class Mainframe extends javax.swing.JFrame {

public Mainframe() {

initComponents();

}

// Выбираем порт вне пределов 1-1024:

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

panel1 = new java.awt.Panel();

textArea1 = new java.awt.TextArea();

label1 = new java.awt.Label();

jButton1 = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

javax.swing.GroupLayout panel1Layout = new javax.swing.GroupLayout(panel1);

panel1.setLayout(panel1Layout);

panel1Layout.setHorizontalGroup(

panel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(panel1Layout.createSequentialGroup()

.addContainerGap()

.addComponent(textArea1, javax.swing.GroupLayout.DEFAULT\_SIZE, 460, Short.MAX\_VALUE)

.addContainerGap())

);

panel1Layout.setVerticalGroup(

panel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(panel1Layout.createSequentialGroup()

.addContainerGap()

.addComponent(textArea1, javax.swing.GroupLayout.DEFAULT\_SIZE, 291, Short.MAX\_VALUE)

.addContainerGap())

);

label1.setText("Server");

jButton1.setText("Run");

jButton1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

try {

jButton1ActionPerformed(evt);

} catch (IOException e) {

throw new RuntimeException(e);

}

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(panel1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addContainerGap())

.addGroup(layout.createSequentialGroup()

.addComponent(label1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton1)

.addGap(113, 113, 113))))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(label1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGap(46, 46, 46))

.addGroup(layout.createSequentialGroup()

.addGap(19, 19, 19)

.addComponent(jButton1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)))

.addComponent(panel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap())

);

panel1.getAccessibleContext().setAccessibleName("mainPanel");

label1.getAccessibleContext().setAccessibleName("Name, label");

pack();

}// </editor-fold>

public static final int PORT = 8080;

public static LinkedList<ServerSomthing> serverList = new LinkedList<>(); // список всех нитей

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) throws IOException {

ServerSocket server = null;

int ConnectionCount;

server = new ServerSocket(8080);

try

{

while (true)

{

// Блокируется до возникновения нового соединения:

Socket socket1 = server.accept();

Socket socket2 = server.accept();

try

{

serverList.add(new ServerSomthing(socket1, socket2)); // добавить новое соединенние в список

}

catch (IOException e)

{

socket1.close();

socket2.close();

}

}

}

finally

{

server.close();

}

}

public static void main(String args[]) throws IOException

{

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Mainframe.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Mainframe.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Mainframe.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Mainframe.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable()

{

public void run() {

new Mainframe().setVisible(true);

}

});

}

class ServerSomthing extends Thread {

private Socket socket1;

private Socket socket2;

// сокет, через который сервер общается с клиентом,

// кроме него - клиент и сервер никак не связаны

private BufferedReader in1; // поток чтения из сокета

private BufferedWriter out1; // поток записи в сокет

private BufferedReader in2; // поток чтения из сокета

private BufferedWriter out2; // поток записи в сокет

public ServerSomthing(Socket socket1, Socket socket2) throws IOException {

this.socket1 = socket1;

this.socket2 = socket2;

// если потоку ввода/вывода приведут к генерированию исключения, оно проброситься дальше

in1 = new BufferedReader(new InputStreamReader(socket1.getInputStream()));

out1 = new BufferedWriter(new OutputStreamWriter(socket1.getOutputStream()));

in2 = new BufferedReader(new InputStreamReader(socket2.getInputStream()));

out2 = new BufferedWriter(new OutputStreamWriter(socket2.getOutputStream()));

start(); // вызываем run()

}

@Override

public void run() {

String word;

try

{

int Count1 = 0;

int Count2 = 0;

out1.write("found\n");

out2.write("found\n");

out1.flush();

out2.flush();

while (true)

{

String choose1 = in1.readLine();

String choose2 = in2.readLine();

if (Objects.equals(choose1, choose2))

{

out1.write("draw\n");

out2.write("draw\n");

out1.flush();

out2.flush();

}

if ((Objects.equals(choose1, "rock")) && (Objects.equals(choose2, "paper")))

{

out1.write("lose\n");

out2.write("win\n");

Count2++;

out1.write(Count2+ "\n");

out2.write(Count2+ "\n");

out1.flush();

out2.flush();

}

if ((Objects.equals(choose1, "rock")) && (Objects.equals(choose2, "scissors")))

{

out1.write("win\n");

out2.write("lose\n");

Count1++;

out1.write(Count1 + "\n");

out2.write(Count1+ "\n");

out1.flush();

out2.flush();

}

if ((Objects.equals(choose1, "paper")) && (Objects.equals(choose2, "rock")))

{

out1.write("win\n");

out2.write("lose\n");

Count1++;

out1.write(Count1 + "\n");

out2.write(Count1+ "\n");

out1.flush();

out2.flush();

}

if ((Objects.equals(choose1, "paper")) && (Objects.equals(choose2, "scissors")))

{

out1.write("lose\n");

out2.write("win\n");

Count2++;

out1.write(Count2+ "\n");

out2.write(Count2+ "\n");

out1.flush();

out2.flush();

}

if ((Objects.equals(choose1, "scissors")) && (Objects.equals(choose2, "rock")))

{

out1.write("lose\n");

out2.write("win\n");

Count2++;

out1.write(Count2+ "\n");

out2.write(Count2+ "\n");

out1.flush();

out2.flush();

}

if ((Objects.equals(choose1, "scissors")) && (Objects.equals(choose2, "paper")))

{

out1.write("win\n");

out2.write("lose\n");

Count1++;

out1.write(Count1 + "\n");

out2.write(Count1+ "\n");

out1.flush();

out2.flush();

}

if((Count1 != 3) && (Count2 != 3))

{

out1.write("continue\n");

out2.write("continue\n");

out1.flush();

out2.flush();

}

else

{

out1.write("endgame\n");

out2.write("endgame\n");

out1.flush();

out2.flush();

socket1.close();

socket2.close();

return;

}

}

}

catch (IOException e)

{

throw new RuntimeException(e);

}

}

}

// Variables declaration - do not modify

private javax.swing.JButton jButton1;

private java.awt.Label label1;

private java.awt.Panel panel1;

private java.awt.TextArea textArea1;

// End of variables declaration

}

# Приложение Б. Исходный код клиентской части

## Приложение Б.1 – MainframeClient.java

import java.awt.\*;

import java.net.\*;

import java.io.\*;

import java.util.Objects;

import java.util.concurrent.TimeUnit;

import javax.swing.BorderFactory;

import javax.swing.border.Border;

import javax.swing.border.EtchedBorder;

public class MainframeClient extends javax.swing.JFrame

{

public MainframeClient()

{

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jLabel2 = new javax.swing.JLabel();

mainPanel = new javax.swing.JPanel();

mainMenuPanel = new javax.swing.JPanel();

mainMenuSignature = new javax.swing.JLabel();

mainMenuStartButton = new javax.swing.JButton();

mainMenuLabel = new javax.swing.JLabel();

connectingPanel = new javax.swing.JPanel();

connectingLabel = new javax.swing.JLabel();

gamePanel = new javax.swing.JPanel();

gameYourPick = new javax.swing.JLabel();

gameRockChooser = new javax.swing.JLabel();

gamePaperChooser = new javax.swing.JLabel();

gameScissorsChooser = new javax.swing.JLabel();

gameEnemyPick = new javax.swing.JLabel();

gameYouLabel = new javax.swing.JLabel();

gameEnemyLabel = new javax.swing.JLabel();

gameYourPoints = new javax.swing.JLabel();

gameDashLabel = new javax.swing.JLabel();

gameEnemyPoints = new javax.swing.JLabel();

gameSendButton = new javax.swing.JButton();

jLabel2.setText("jLabel2");

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setSize(new java.awt.Dimension(0, 0));

mainPanel.setLayout(new java.awt.CardLayout());

mainMenuSignature.setText("Выполнил ученик 19ВВ2 Лобанов Дмитрий");

mainMenuStartButton.setText("Start");

mainMenuStartButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

mainMenuStartButtonActionPerformed(evt);

}

});

mainMenuLabel.setFont(new java.awt.Font("Segoe UI", 1, 24)); // NOI18N

mainMenuLabel.setText("ROCK, PAPER, SCISSORS");

javax.swing.GroupLayout mainMenuPanelLayout = new javax.swing.GroupLayout(mainMenuPanel);

mainMenuPanel.setLayout(mainMenuPanelLayout);

mainMenuPanelLayout.setHorizontalGroup(

mainMenuPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(mainMenuPanelLayout.createSequentialGroup()

.addGroup(mainMenuPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(mainMenuLabel)

.addGroup(mainMenuPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(mainMenuPanelLayout.createSequentialGroup()

.addGap(90, 90, 90)

.addComponent(mainMenuSignature, javax.swing.GroupLayout.PREFERRED\_SIZE, 254, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(mainMenuPanelLayout.createSequentialGroup()

.addGap(75, 75, 75)

.addComponent(mainMenuStartButton, javax.swing.GroupLayout.PREFERRED\_SIZE, 278, javax.swing.GroupLayout.PREFERRED\_SIZE))))

.addContainerGap(66, Short.MAX\_VALUE))

);

mainMenuPanelLayout.setVerticalGroup(

mainMenuPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(mainMenuPanelLayout.createSequentialGroup()

.addGap(23, 23, 23)

.addComponent(mainMenuLabel)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 72, Short.MAX\_VALUE)

.addComponent(mainMenuStartButton, javax.swing.GroupLayout.PREFERRED\_SIZE, 30, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(18, 18, 18)

.addComponent(mainMenuSignature)

.addGap(48, 48, 48))

);

mainPanel.add(mainMenuPanel, "card2");

connectingLabel.setFont(new java.awt.Font("Segoe UI", 1, 18)); // NOI18N

connectingLabel.setText("Connecting...");

javax.swing.GroupLayout connectingPanelLayout = new javax.swing.GroupLayout(connectingPanel);

connectingPanel.setLayout(connectingPanelLayout);

connectingPanelLayout.setHorizontalGroup(

connectingPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, connectingPanelLayout.createSequentialGroup()

.addContainerGap(161, Short.MAX\_VALUE)

.addComponent(connectingLabel, javax.swing.GroupLayout.PREFERRED\_SIZE, 147, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(111, 111, 111))

);

connectingPanelLayout.setVerticalGroup(

connectingPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(connectingPanelLayout.createSequentialGroup()

.addGap(87, 87, 87)

.addComponent(connectingLabel, javax.swing.GroupLayout.PREFERRED\_SIZE, 25, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap(127, Short.MAX\_VALUE))

);

mainPanel.add(connectingPanel, "card3");

gamePanel.setPreferredSize(new java.awt.Dimension(419, 239));

gameYourPick.setToolTipText("");

gameRockChooser.setText("Rock");

gameRockChooser.setBorder(javax.swing.BorderFactory.createBevelBorder(javax.swing.border.BevelBorder.RAISED));

gameRockChooser.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND\_CURSOR));

gameRockChooser.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

gameRockChooserMouseClicked(evt);

}

});

gamePaperChooser.setText("Paper");

gamePaperChooser.setToolTipText("");

gamePaperChooser.setBorder(javax.swing.BorderFactory.createBevelBorder(javax.swing.border.BevelBorder.RAISED));

gamePaperChooser.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND\_CURSOR));

gamePaperChooser.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

gamePaperChooserMouseClicked(evt);

}

});

gameScissorsChooser.setText("Scissors");

gameScissorsChooser.setToolTipText("");

gameScissorsChooser.setBorder(javax.swing.BorderFactory.createBevelBorder(javax.swing.border.BevelBorder.RAISED));

gameScissorsChooser.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND\_CURSOR));

gameScissorsChooser.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

gameScissorsChooserMouseClicked(evt);

}

});

gameEnemyPick.setToolTipText("");

gameYouLabel.setText("You");

gameEnemyLabel.setText("Enemy");

gameYourPoints.setFont(new java.awt.Font("Segoe UI", 1, 24)); // NOI18N

gameYourPoints.setText("0");

gameDashLabel.setFont(new java.awt.Font("Segoe UI", 1, 24)); // NOI18N

gameDashLabel.setText("-");

gameEnemyPoints.setFont(new java.awt.Font("Segoe UI", 1, 24)); // NOI18N

gameEnemyPoints.setText("0");

gameSendButton.setText("Send");

gameSendButton.setActionCommand("");

gameSendButton.setEnabled(false);

gameSendButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

gameSendButtonActionPerformed(evt);

}

});

javax.swing.GroupLayout gamePanelLayout = new javax.swing.GroupLayout(gamePanel);

gamePanel.setLayout(gamePanelLayout);

gamePanelLayout.setHorizontalGroup(

gamePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, gamePanelLayout.createSequentialGroup()

.addGroup(gamePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(gamePanelLayout.createSequentialGroup()

.addGap(101, 101, 101)

.addComponent(gameYouLabel))

.addGroup(gamePanelLayout.createSequentialGroup()

.addGap(22, 22, 22)

.addGroup(gamePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(gamePanelLayout.createSequentialGroup()

.addComponent(gameRockChooser, javax.swing.GroupLayout.PREFERRED\_SIZE, 60, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(47, 47, 47)

.addComponent(gamePaperChooser, javax.swing.GroupLayout.PREFERRED\_SIZE, 60, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(gamePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(gameYourPoints)

.addComponent(gameYourPick, javax.swing.GroupLayout.PREFERRED\_SIZE, 160, javax.swing.GroupLayout.PREFERRED\_SIZE)))))

.addGap(18, 18, 18)

.addComponent(gameDashLabel)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 21, Short.MAX\_VALUE)

.addGroup(gamePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addGroup(gamePanelLayout.createSequentialGroup()

.addComponent(gameScissorsChooser, javax.swing.GroupLayout.PREFERRED\_SIZE, 60, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 33, Short.MAX\_VALUE)

.addComponent(gameSendButton))

.addGroup(gamePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(gamePanelLayout.createSequentialGroup()

.addComponent(gameEnemyPoints)

.addGap(25, 25, 25)

.addComponent(gameEnemyLabel))

.addComponent(gameEnemyPick, javax.swing.GroupLayout.PREFERRED\_SIZE, 160, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addGap(16, 16, 16))

);

gamePanelLayout.setVerticalGroup(

gamePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, gamePanelLayout.createSequentialGroup()

.addContainerGap()

.addGroup(gamePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(gameYourPoints)

.addComponent(gameDashLabel)

.addComponent(gameEnemyPoints)

.addComponent(gameYouLabel)

.addComponent(gameEnemyLabel))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(gamePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(gameYourPick, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED\_SIZE, 100, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(gameEnemyPick, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED\_SIZE, 100, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(gamePanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(gamePaperChooser, javax.swing.GroupLayout.PREFERRED\_SIZE, 60, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(gameScissorsChooser, javax.swing.GroupLayout.PREFERRED\_SIZE, 60, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(gameRockChooser, javax.swing.GroupLayout.PREFERRED\_SIZE, 60, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(gameSendButton, javax.swing.GroupLayout.PREFERRED\_SIZE, 32, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(110, 110, 110))

);

gameYourPick.getAccessibleContext().setAccessibleName("gameYourPick");

mainPanel.add(gamePanel, "card4");

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(mainPanel, javax.swing.GroupLayout.PREFERRED\_SIZE, 419, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(mainPanel, javax.swing.GroupLayout.PREFERRED\_SIZE, 239, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

pack();

}// </editor-fold>

private static Socket clientSocket; //сокет для общения

private static BufferedReader in; // поток чтения из сокета

private static BufferedWriter out; // поток записи в сокет

boolean isRockChoosed = false;

boolean isPaperChoosed = false;

boolean isScissorsChoosed = false;

javax.swing.ImageIcon YourPaper = new javax.swing.ImageIcon("src/main/java/icons/paper\_2\_Your.png");

javax.swing.ImageIcon YourRock = new javax.swing.ImageIcon("src/main/java/icons/rock\_2\_Your.png");

javax.swing.ImageIcon YourScissors = new javax.swing.ImageIcon("src/main/java/icons/scissors\_2\_Your.png");

javax.swing.ImageIcon ChoosePaperLabel = new javax.swing.ImageIcon("src/main/java/icons/paper.png");

javax.swing.ImageIcon ChooseRockLabel = new javax.swing.ImageIcon("src/main/java/icons/rock.png");

javax.swing.ImageIcon ChooseScissorsLabel = new javax.swing.ImageIcon("src/main/java/icons/scissors.png");

javax.swing.ImageIcon EnemyPaper = new javax.swing.ImageIcon("src/main/java/icons/paper\_2\_Enemy.png");

javax.swing.ImageIcon EnemyRock = new javax.swing.ImageIcon("src/main/java/icons/rock\_2\_Enemy.png");

javax.swing.ImageIcon EnemyScissors = new javax.swing.ImageIcon("src/main/java/icons/scissors\_2\_Enemy.png");

private void mainMenuStartButtonActionPerformed(java.awt.event.ActionEvent evt) {

mainMenuPanel.setVisible(false);

mainMenuPanel.setEnabled(false);

connectingPanel.setVisible(true);

connectingPanel.setEnabled(true);

gameRockChooser.setIcon(ChooseRockLabel);

gamePaperChooser.setIcon(ChoosePaperLabel);

gameScissorsChooser.setIcon(ChooseScissorsLabel);

mainMenuPanel.updateUI();

connectingPanel.updateUI();

try

{

clientSocket = new Socket("localhost", 8080);

// читать соообщения с сервера

in = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));

// писать туда же

out = new BufferedWriter(new OutputStreamWriter(clientSocket.getOutputStream()));

String str = in.readLine();

if (Objects.equals(str, "found"))

{

connectingPanel.setVisible(false);

connectingPanel.setEnabled(false);

gamePanel.setVisible(true);

gamePanel.setEnabled(true);

}

else

{

connectingLabel.setText("No one came...");

mainMenuPanel.setEnabled(true);

mainMenuPanel.setVisible(true);

connectingPanel.setVisible(false);

connectingPanel.setEnabled(false);

}

}

catch (IOException e)

{

connectingLabel.setText("Error!");

}

}

private void gameRockChooserMouseClicked(java.awt.event.MouseEvent evt) {

if(gameRockChooser.isEnabled())

{

Border loweredBorder, raisedBorder;

raisedBorder = BorderFactory.createRaisedBevelBorder();

loweredBorder = BorderFactory.createLoweredBevelBorder();

gameRockChooser.setBorder(loweredBorder);

gamePaperChooser.setBorder(raisedBorder);

gameScissorsChooser.setBorder(raisedBorder);

isRockChoosed = true;

isPaperChoosed = false;

isScissorsChoosed = false;

gameSendButton.setEnabled(true);

}

}

private void gamePaperChooserMouseClicked(java.awt.event.MouseEvent evt) {

if(gamePaperChooser.isEnabled()) {

Border loweredBorder, raisedBorder;

raisedBorder = BorderFactory.createRaisedBevelBorder();

loweredBorder = BorderFactory.createLoweredBevelBorder();

gameRockChooser.setBorder(raisedBorder);

gamePaperChooser.setBorder(loweredBorder);

gameScissorsChooser.setBorder(raisedBorder);

isRockChoosed = false;

isPaperChoosed = true;

isScissorsChoosed = false;

gameSendButton.setEnabled(true);

}

}

private void gameScissorsChooserMouseClicked(java.awt.event.MouseEvent evt) {

if(gameScissorsChooser.isEnabled())

{

Border loweredBorder, raisedBorder;

raisedBorder = BorderFactory.createRaisedBevelBorder();

loweredBorder = BorderFactory.createLoweredBevelBorder();

gameRockChooser.setBorder(raisedBorder);

gamePaperChooser.setBorder(raisedBorder);

gameScissorsChooser.setBorder(loweredBorder);

isRockChoosed = false;

isPaperChoosed = false;

isScissorsChoosed = true;

gameSendButton.setEnabled(true);

}

}

private void gameSendButtonActionPerformed(java.awt.event.ActionEvent evt) {

try {

if (isPaperChoosed == true)

{

out.write("paper" + "\n");

gameYourPick.setIcon(YourPaper);

} else if (isRockChoosed == true)

{

out.write("rock" + "\n");

gameYourPick.setIcon(YourRock);

} else {

out.write("scissors" + "\n");

gameYourPick.setIcon(YourScissors);

}

out.flush();

gameSendButton.setEnabled(false);

gameRockChooser.setEnabled(false);

gamePaperChooser.setEnabled(false);

gameScissorsChooser.setEnabled(false);

String result = in.readLine();

if (Objects.equals(result, "win"))

{

gameYourPoints.setText(in.readLine());

if (isPaperChoosed == true)

{

gameEnemyPick.setIcon(EnemyRock);

}

else if (isRockChoosed == true)

{

gameEnemyPick.setIcon(EnemyScissors);

}

else if (isScissorsChoosed == true)

{

gameEnemyPick.setIcon(EnemyPaper);

}

}

if (Objects.equals(result, "lose"))

{

gameEnemyPoints.setText(in.readLine());

if (isPaperChoosed == true)

{

gameEnemyPick.setIcon(EnemyScissors);

}

else if (isRockChoosed == true)

{

gameEnemyPick.setIcon(EnemyPaper);

}

else if (isScissorsChoosed == true)

{

gameEnemyPick.setIcon(EnemyRock);

}

}

if (Objects.equals(result, "draw"))

{

if (isPaperChoosed == true)

{

gameEnemyPick.setIcon(EnemyPaper);

}

else if (isRockChoosed == true)

{

gameEnemyPick.setIcon(EnemyRock);

}

else if (isScissorsChoosed == true)

{

gameEnemyPick.setIcon(EnemyScissors);

}

}

Thread.sleep(100);

String gameState = in.readLine();

if (Objects.equals(gameState, "endgame"))

{

gameEnemyPick.setIcon(null);

gameYourPick.setIcon(null);

gameEnemyPoints.setText("0");

gameYourPoints.setText("0");

gameSendButton.setEnabled(true);

gameRockChooser.setEnabled(true);

gamePaperChooser.setEnabled(true);

gameScissorsChooser.setEnabled(true);

clientSocket.close();

in.close();

out.close();

gamePanel.setVisible(false);

gamePanel.setEnabled(false);

mainMenuPanel.setVisible(true);

mainMenuPanel.setEnabled(true);

return;

}

gameSendButton.setEnabled(true);

gameRockChooser.setEnabled(true);

//gameRockChooser.setVisible(false);

gamePaperChooser.setEnabled(true);

//gamePaperChooser.setVisible(false);

gameScissorsChooser.setEnabled(true);

//gameScissorsChooser.setVisible(false);

}

catch (IOException | InterruptedException e)

{

gameSendButton.setText("Error!");

}

}

public static void main(String args[]) throws IOException

{

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(MainframeClient.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(MainframeClient.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(MainframeClient.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(MainframeClient.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable()

{

public void run() {

new MainframeClient().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JLabel connectingLabel;

private javax.swing.JPanel connectingPanel;

private javax.swing.JLabel gameDashLabel;

private javax.swing.JLabel gameEnemyLabel;

private javax.swing.JLabel gameEnemyPick;

private javax.swing.JLabel gameEnemyPoints;

private javax.swing.JPanel gamePanel;

private javax.swing.JLabel gamePaperChooser;

private javax.swing.JLabel gameRockChooser;

private javax.swing.JLabel gameScissorsChooser;

private javax.swing.JButton gameSendButton;

private javax.swing.JLabel gameYouLabel;

private javax.swing.JLabel gameYourPick;

private javax.swing.JLabel gameYourPoints;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel mainMenuLabel;

private javax.swing.JPanel mainMenuPanel;

private javax.swing.JLabel mainMenuSignature;

private javax.swing.JButton mainMenuStartButton;

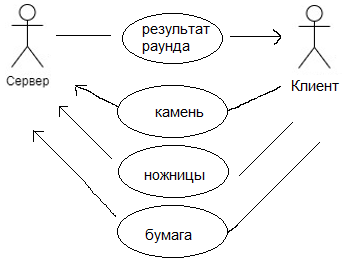
private javax.swing.JPanel mainPanel;

// End of variables declaration

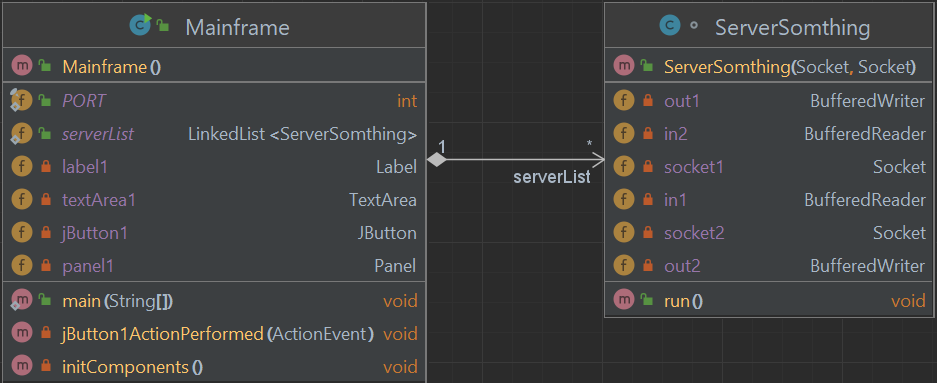
}

# Приложение В. UML-диаграммы приложения

# Приложение В.1 – UML-диаграмма вариантов использования



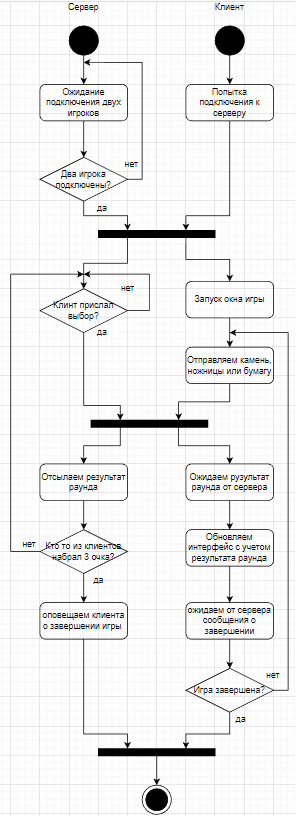
## Приложение В.2 – UML-диаграмма классов сервера



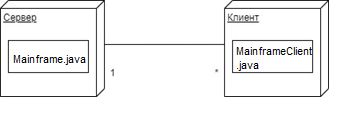
## Приложение В.3 – UML-диаграмма классов клиента



# Приложение В.4 – UML-диаграмма деятельности



# Приложение В.5 – UML-диаграмма развёртывания



# Приложение В.6 – UML-диаграмма последовательности

