МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Пензенский государственный университет

Кафедра «Вычислительная техника»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

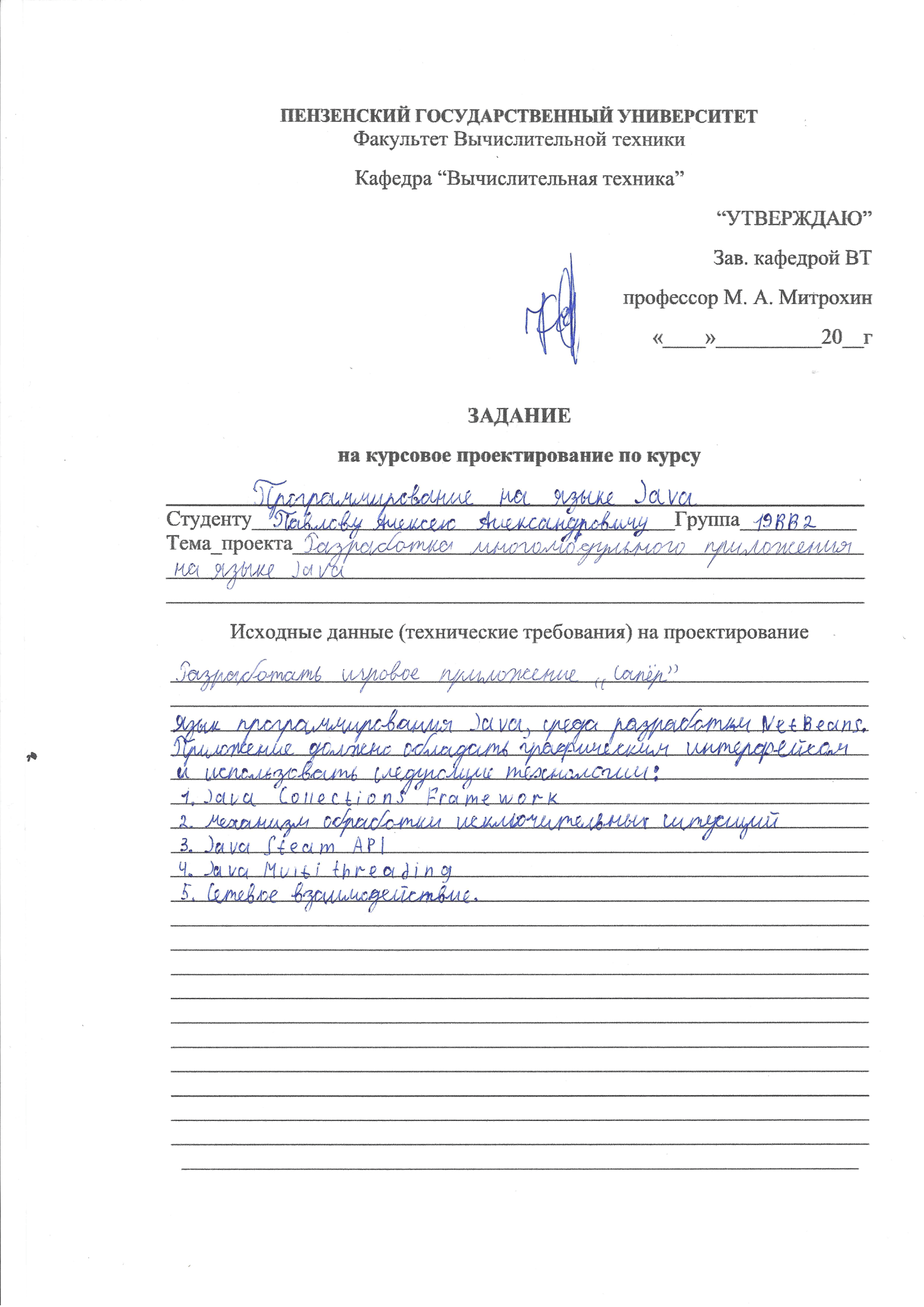
к курсовому проекту по дисциплине

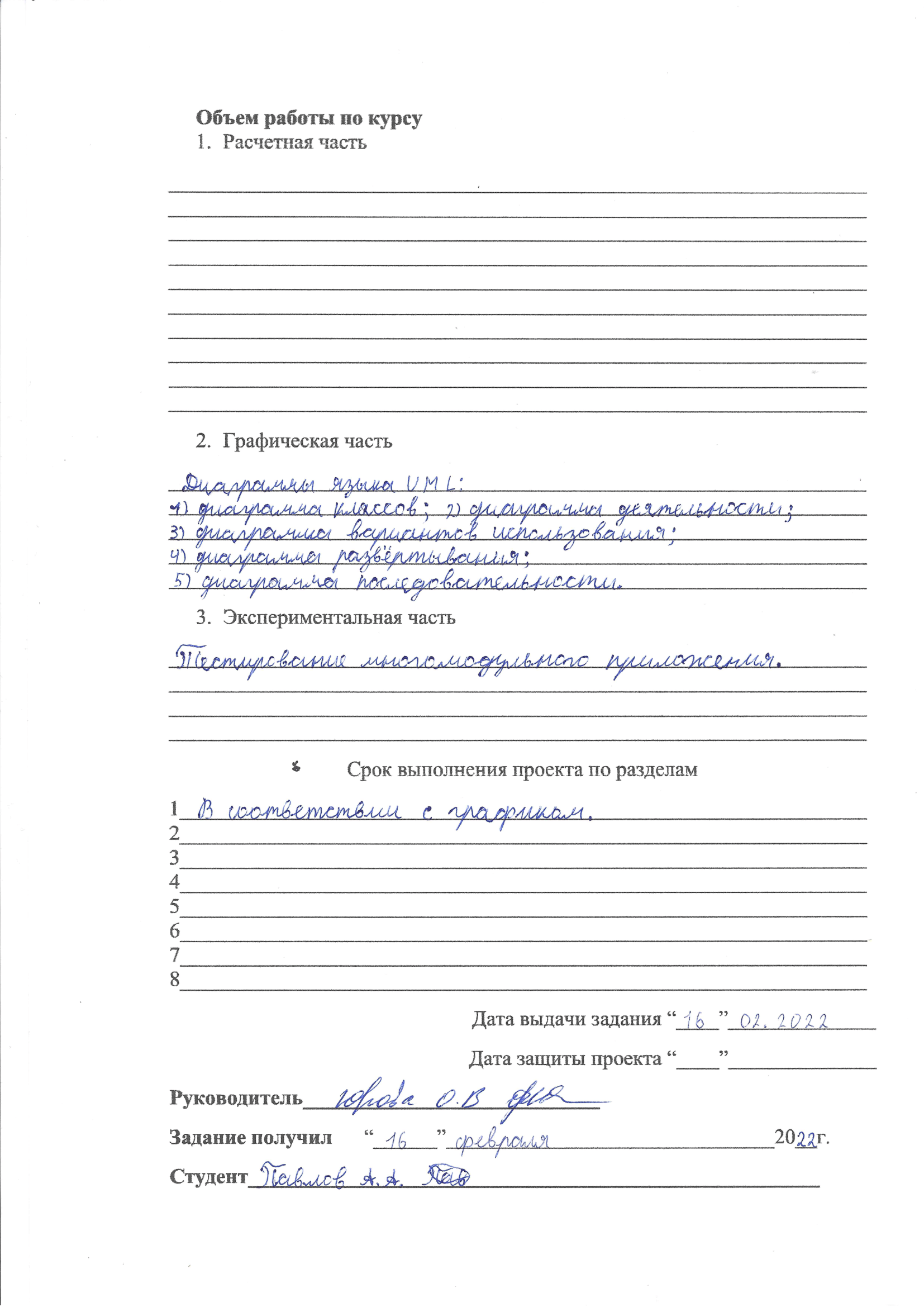
«Программирование на языке Java»

на тему: «Разработка многомодульного приложения на языке Java»

|  |
| --- |
| Выполнил: |
| Студент группы 19ВВ2 |
| Павлов А.А. |
| Принял: |
| Юрова О.В. |

Пенза 2022





Содержание

[Введение 5](#_Toc104996471)

[1 Постановка задачи 6](#_Toc104996472)

[2 Выбор решения 7](#_Toc104996473)

[3 Описание программы 8](#_Toc104996474)

[4 Описание способа организации пользовательского интерфейса 13](#_Toc104996475)

[5 Описание результатов работы программы 14](#_Toc104996476)

[Заключение 18](#_Toc104996477)

[Литература 19](#_Toc104996478)

[Приложение А. Листинг программы 20](#_Toc104996479)

[Приложение Б. UML-диаграммы 87](#_Toc104996480)

# Введение

Java разработала компания Sun Microsystems в начале 90-х годов XX века. Ведущую роль в создании языка сыграл канадский инженер Джеймс Гослинг (James Gosling). На ранних этапах разработки язык назывался Oak. Затем его переименовали в честь сорта кофе Java. Связь языка с напитком отражается в логотипе.

Джеймс Гослинг и его единомышленники хотели создать язык с си-подобным синтаксисом. В то же время он должен быть более простым по сравнению с C/C++. Создатели планировали использовать Java для программирования бытовой электроники. Однако практически сразу после выпуска версии 1.0 в 1995 язык стали использовать разработчики серверного и клиентского ПО.

Java — язык программирования общего назначения. Относится к объектно-ориентированным языкам программирования, к языкам с сильной типизацией. Создатели реализовали принцип WORA: write once, run anywhere или «пиши один раз, запускай везде». Это значит, что написанное на Java приложение можно запустить на любой платформе, если на ней установлена среда исполнения Java (JRE, Java Runtime Environment).

Эта задача решается благодаря компиляции написанного на Java кода в байт-код. Этот формат исполняет JVM или виртуальная машина Java. JVM — часть среды исполнения Java (JRE). Виртуальная машина не зависит от платформы.

Java является объектно-ориентированным языком. Он поддерживает полиморфизм, наследование, статическую типизацию. Объектно-ориентированный подход позволяет решить задачи по построению крупных, но в тоже время гибких, масштабируемых и расширяемых приложений.

# 1 Постановка задачи

Основной задачей данного курсового проекта является программная реализация многомодульного игрового приложения «Сапёр» на языке Java, среда разработки NetBeans.

Программа должна включать сетевое взаимодействие, то есть чтобы пользователь мог через локальную сеть подключаться к серверу и осуществлять кооперативную игру с другими пользователями. В проекте предусмотрен графический интерфейс и используются следующие технологии:

1. Java Collections Framework

2. Механизм обработки исключительных ситуаций

3. Java Stream API

4. Java Multithreading

5. Сетевое взаимодействие

# 2 Выбор решения

Для написания данной программы использован язык программирования Java.

«Сапёр» (англ. Minesweeper) — популярная компьютерная игра-головоломка, входящая в состав практически всех версий операционной системы Windows.

Для реализации проекта используется библиотека Swing; интерфейс Runnable; пакеты, содержащие файлы ресурсов, пользовательский интерфейс и сетевое взаимодействие, и игровую логику; унаследованные от класса Thread классы, обработка исключительных ситуаций; сетевое взаимодействие; Java Stream API и тд.

# 3 Описание программы

Программа реализует кооперативное прохождение игры «Сапёр» для пользователей, подключенных к серверу по локальной сети.

Для корректной работы необходимо запустить сначала проект «SaperKursServer». Появится окно с кнопкой «Запустить сервер», при нажатии на которую появится диалоговое окно с полем ввода порта сервера и тремя кнопками выбора уровня сложности, при нажатии на любую из которых сервер будет запущен под указанным раннее в поле портом.

После запуска сервера запускается проект «SaperKursClient». Появится диалоговое окно с двумя полями ввода: «Адрес» и «Порт», и кнопкой «Подключиться». При корректно введённых данных, после нажатия на кнопку происходит подключение к серверу и на экран выводится окно с полученным от сервера игровым полем.

Результатом программы является кооперативное прохождение игры «Сапёр», т.е. подключение к серверу и обмен действиями между участниками.

Запуск сервера происходит следующим образом. В зависимости от выбранной кнопки уровня сложности выбирается количество строк и столбцов ячеек игрового поля, а также бомб: «Легко – 9 столбцов, 9 строк, 10 бомб», «Норм – 12 столбцов, 12 строк, 20 бомб», «Saper – 16 столбцов, 16 строк, 40 бомб», - после чего создаётся серверный сокет под указанным при запуске портом. Далее запускается новый поток, который будет отслеживать подключение новых клиентов к серверу и вместе с тем записывать их в коллекцию ArrayList. Каждое подключение клиента в дальнейшем также будет обрабатываться в новом потоке.

При обнаружении запроса на подключение от клиента при условии, что серверный сокет корректно запущен, создаётся объект унаследованного от класса «Thread» класса «SocketThread», в конструктор которого передаётся сокет клиента, его уникальный номер и экземпляр созданной игры.

После запускается новый поток, который записывает в выходной поток (OutputStream) размер игрового поля и экземпляр созданной игры, после чего инициализирует класс координат и входит в бесконечный цикл в ожидании запросов от клиента. Запросы представляют из себя простые слова класса String.

Запрос «Left», «Right» - клиент оповещает, что была нажата левая или правая кнопка мыши, в этом случае входной поток (InputStream) считывает координаты, на которые было произведено нажатие кнопки мыши, и передаёт их статическому методу класса «Server» «PressButton» вместе со своим уникальным номером и полученным сообщением. Метод перебирает коллекцию клиентских сокетов и передаёт им полученные данные при условии, если клиентский сокет в рабочем состоянии, и он не является адресатом (то есть не от того же самого сокета был произведён запрос). После чего конкретно связанный со своим клиентом сокет записывает в выходной поток сообщение, обозначающее нажатую кнопку, и координаты.

Запрос «NgPls» - клиент оповещает о завершении игрового процесса и просит новый экземпляр игры. Сервер инициализирует объект класса Game и записывает его в выходные потоки клиентских сокетов по указанному ранее принципу.

Запрос «Exit» - клиент оповещает о завершении своей работы. В этом случае при условии, что подключение с клиентом всё ещё установлено (*эта проверка нужна на случай, если разрыв соединения был произведён самим сервером, в этом случае поток принимает сообщение о завершении работы клиента и просто прекращает свою работу*), поток отдельного клиентского сокета закрывает входной и выходной поток и прекращает свою работу.

При корректном подключении к серверу клиент считывает из входного потока размеры игрового поля и экземпляр созданной игры. Дальше клиент запускает новый поток, который будет отслеживать запросы других клиентов, переданные через сервер.

При получении одного из таких, клиент будет обновлять игровое поле в соответствии с выполненным другим клиентом действием.

Пакет «res.img» содержит картинки типа png, которые отображаются на панели игрового поля.

В пакете «sweeper», что является общим и для сервера, и для клиента, находятся классы и перечисления, реализующие игровую логику.

Перечисление (enum) «Box» содержит название картинок в специальном порядке, объект «image» для отображения картинок, метод «GetNumber» для получения числа, который эквивалентен номеру картинки в перечислении; и метод «GetNextNumberBox» для получения следующего числа относительно текущего.

Класс «Coord» хранит координаты клетки поля и сравнивает координаты друг с другом.

Класс «Ranges» хранит в себе размеры поля (статическое поле Coord size), статическая коллекция ArrayList «allCoords» типа «Coord», где хранятся все игровые координаты; и объект класса «Random» для создания генератора случайных чисел, который будет генерировать случайные координаты. Также реализован метод «GetCoordsAround», который возвращает коллекцию ArrayList координат, находящихся вокруг переданной в параметр координаты. Метод «InRange» проверяет координату на то, находится ли она в пределах игрового поля.

Класс Matrix хранит матрицу объектов перечисления «Box». Конструктор заполняет всю матрицу указанным элементом «Box». Метод «Get» для получения по координатам конкретного объекта «Box». И метод «Set» для установки объекта «Box» по координатам.

Класс «Bomb» хранит в себе матрицу «bombMap» класса «Matrix» с расположением всех бомб и чисел вокруг них. Поле «totalBombs» хранит число общего количества бомб. Метод «Start» инициализирует матрицу картинками пустых клеток и вызывает метод для размещения бомб. Метод «Get» для получения конкретного объекта «Box» по координатам. Метод «FixBombsCount» для ограничения максимального количества бомб относительно размера поля. Метод «PlaceBomb» размещает «Box» с бомбой относительно случайной координаты. Метод «IncNumbersAroundBomb» для увеличения значения «Box» вокруг бомбы.

Класс «Flag» хранит в себе матрицу «flagMap» класса «Matrix» для работы с верхним слоем поля: закрытые «Box», на которые можно ставить флаги. Поле «countOfClosedBoxes» для отслеживания числа закрытых объектов «Box». Метод «Start» инициализирует матрицу картинками пустых закрытых клеток и устанавливает их общее количество. Метод «Get» для получения конкретного объекта «Box» по координатам. Методы «SetOpenedToBox», «SetFlagedToBox», «SetClosedToBox», «SetBombedToBox» для установки обозначенных «Box» по координатам. Метод «ToggleFlagedToBox» для переключения между закрытым «Box» и «Box» с флагом. Метод «IncNumbersAroundBomb» для увеличения значения «Box» вокруг бомбы. Метод «SetNoBombToFlagedSafeBox» для установки «Box» с отсутствием бомбы в месте, где неверно был размещён «Box» с флагом. Метод «SetOpenedToClosedBombBox» для установки «Box» с бомбой в месте, где она была размещена случайным образом. Метод «GetCountOfFlagedBoxesAround» для получения числа «Box» с флагом вокруг указанной координаты.

Перечисление «GameState», где записаны возможные состояния игры.

Класс «Game» для управления процессами игры. Объект «bomb» класса «Bomb» хранит нижний слой поля, где расположены бомбы и цифры. Объект «flag» класса «Flag» хранит верхний слой поля, ячейки которого можно открывать и ставить на них флаги. Метод «StartSolo» инициализирует матрицы нижнего и верхнего слоя поля и устанавливает состояние игры, используется только на стороне сервера. Метод «StartCoop» только устанавливает состояние игры, используется на стороне клиента. Методы «PressLeftButton» и «PressRightButton» для обработки действий при нажатии кнопки мыши. Метод «GetBox» для получения конкретного «Box» в указанных координатах. Метод «OpenBombs» открывает всё игровое поле в случае поражения. Метод «OpenBoxesAround» для создания рекурсии открытия пустых «Box». Метод «GameOver» для проверки текущего состояния игры. Метод «SetOpenedToClosedBoxesAroundNumber» для открытия закрытых «Box» вокруг открытой клетки при условии клика на неё, если количество «Box» с флагами равно количеству «Box» с бомбами.

# 4 Описание способа организации пользовательского интерфейса

Для реализации пользовательского интерфейса была использована библиотека Swing.

В приложении «SaperKursServer» основной JFrame содержит кнопку «jButton4» для запуска сервера и панель «jPanel2» с меткой «jLabel1», которая после попытки включения сервера будут отображать его состояние.

Нажатие «jButton4» вызывает диалоговое окно «jDialog1» с двумя панелями. jPanel1 включает в себя две текстовые метки и текстовое поле «jTextField1», откуда в процессе работы будет считываться значение порта. «jPanel3» содержит 3 кнопки «jButton1-3», являющиеся выбором уровня сложности игры.

При запуске приложения «SaperKursClient» пользователя сначала встречает диалоговое окно с двумя панелями и кнопкой «jButton1». Панель «jPanel3» содержит два текстовых поля «jTextField1» и «jTextField2», которые нужны для ввода адреса сервера и его порта, и две пояснительные текстовые метки. Панель «jPanel4» содержит только одну текстовую метку.

При корректно введённых значениях адреса и порта, после нажатия «jButton1», диалоговое окно закрывается и открывается основной JFrame приложения, на панели «jPanel1» которого будет размещаться игровое поле, представляющее из себя матрицу картинок размером 50x50 пикселей. Панель «jPanel2» содержит одну текстовую метку «jLabel1», которая в процессе работы приложения отображает состояние игры и сервера.

# 5 Описание результатов работы программы

Среда разработки NetBeans предоставляет все средства, необходимые при разработке и отладке многомодульной программы.

Тестирование проводилось в рабочем порядке, в процессе разработки, после завершения написания программы. В ходе тестирования было выявлено и исправлено множество проблем, связанных с вводом данных, изменением дизайна клиентской части, алгоритмом программы.

Ниже продемонстрирован результат тестирования программы сначала при работе с двумя клиентами, а после и с подключением третьего.

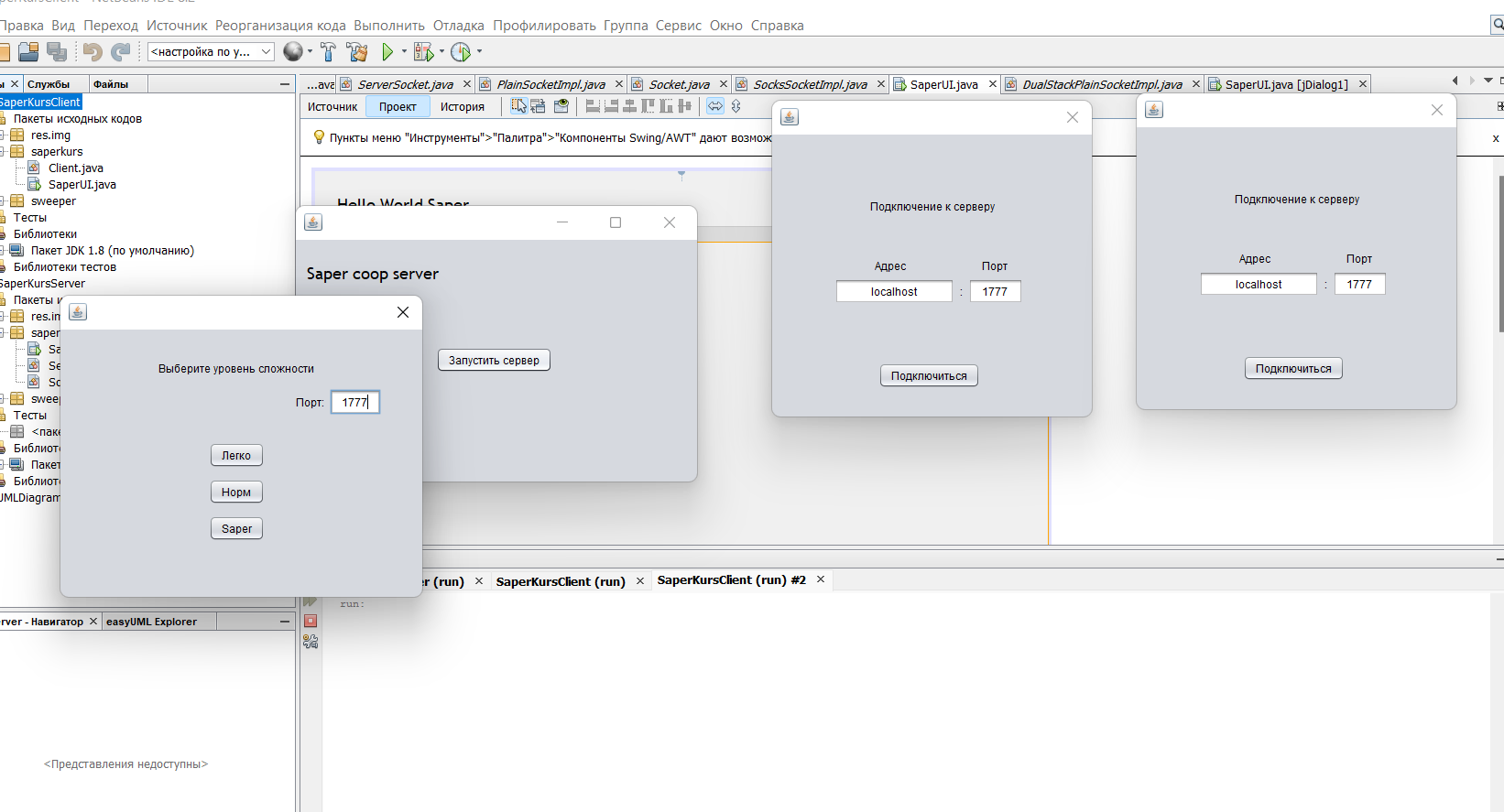


Рисунок 1 – Запуск приложений

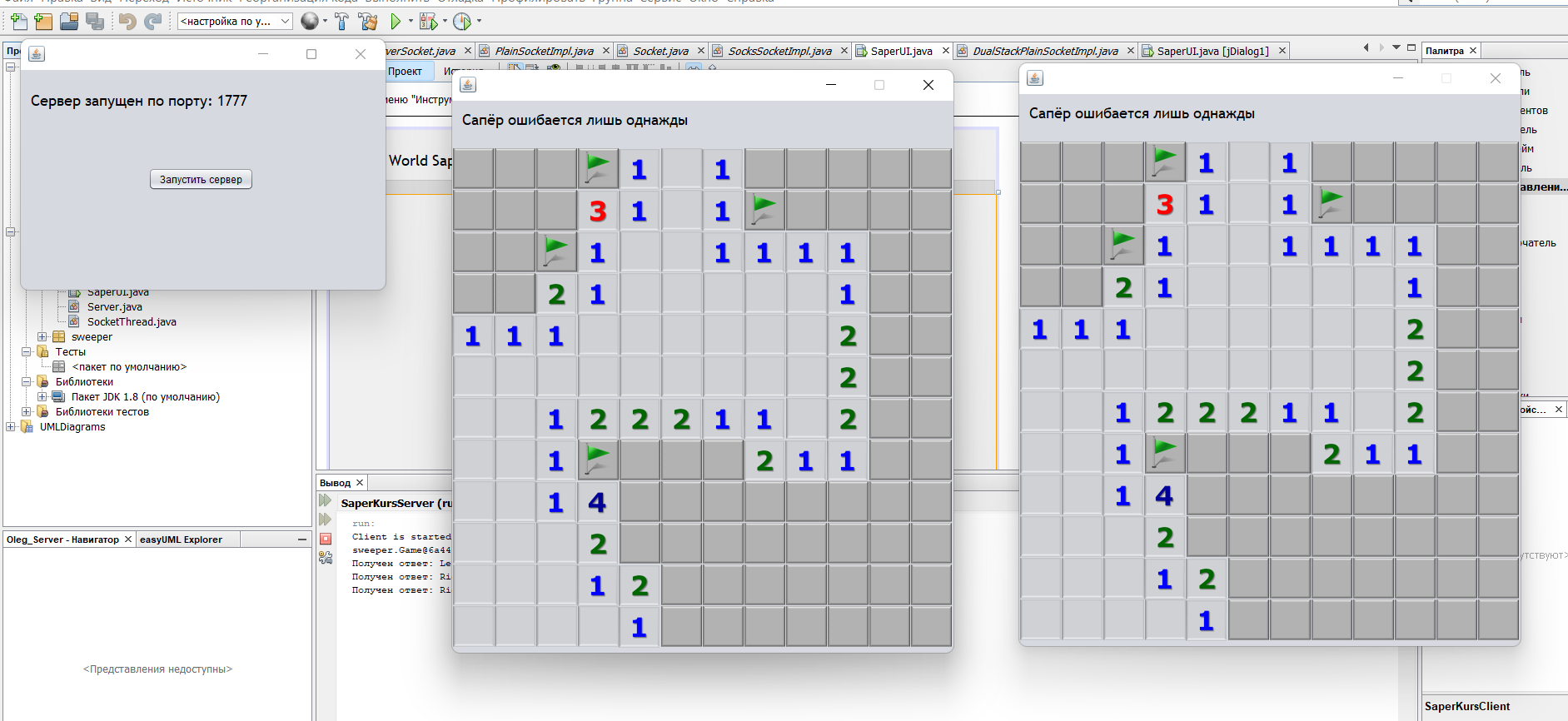
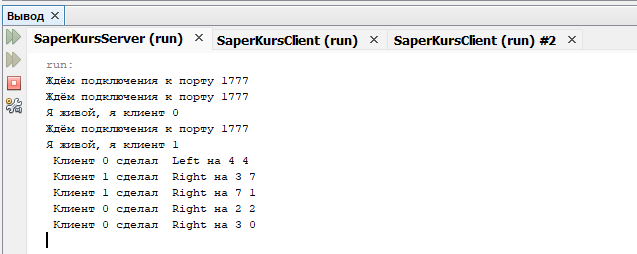
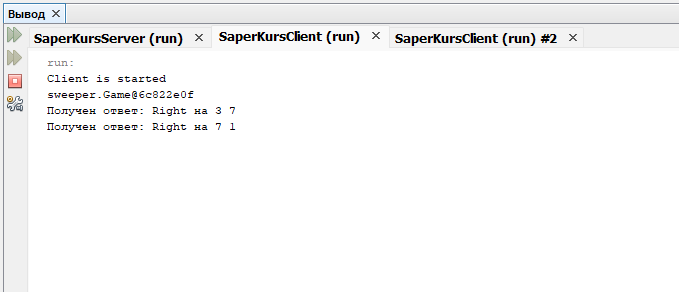
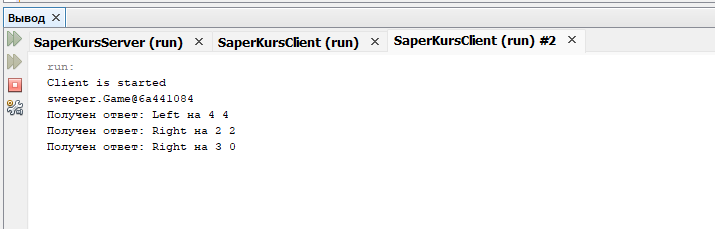


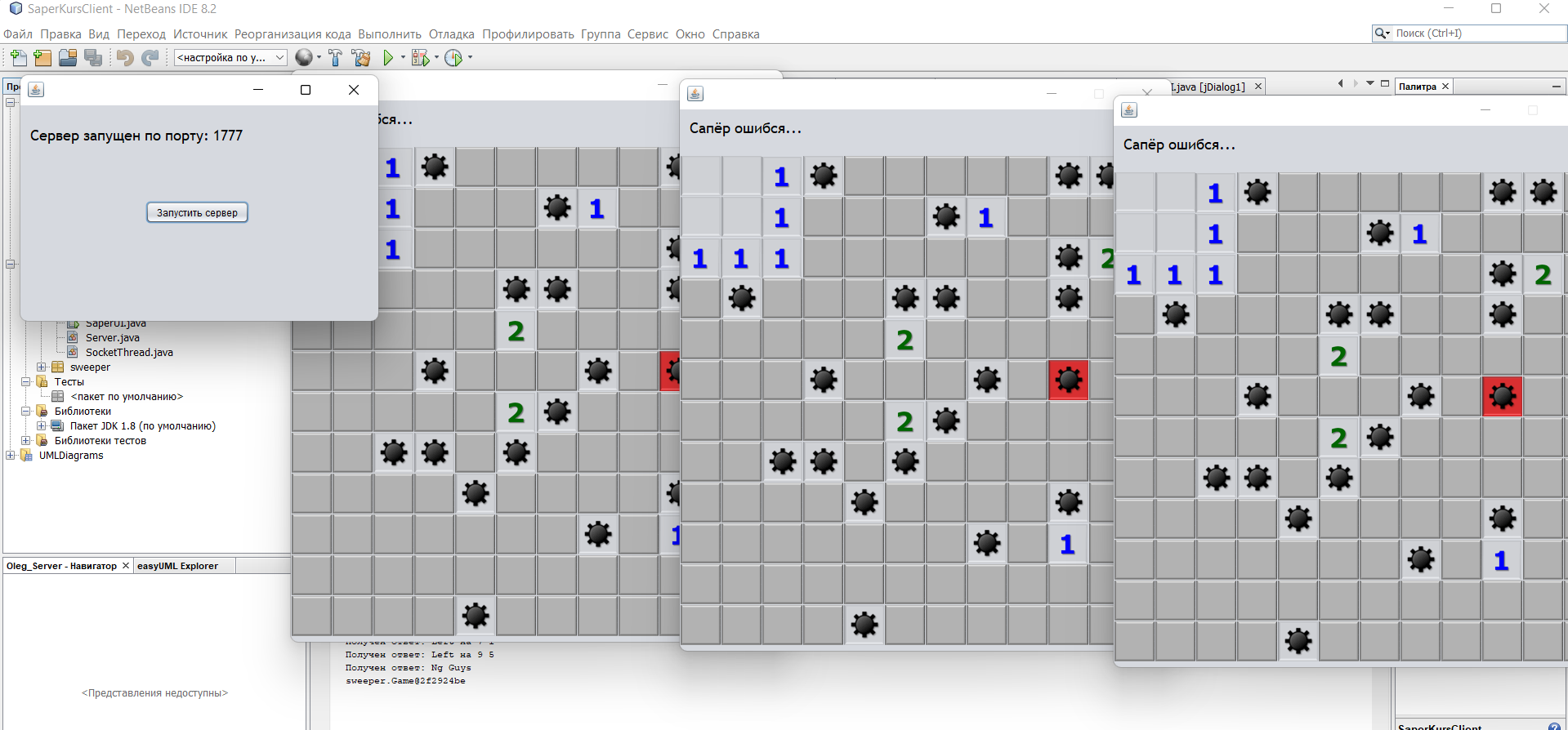
Рисунок 2 – Результаты работы графического интерфейса

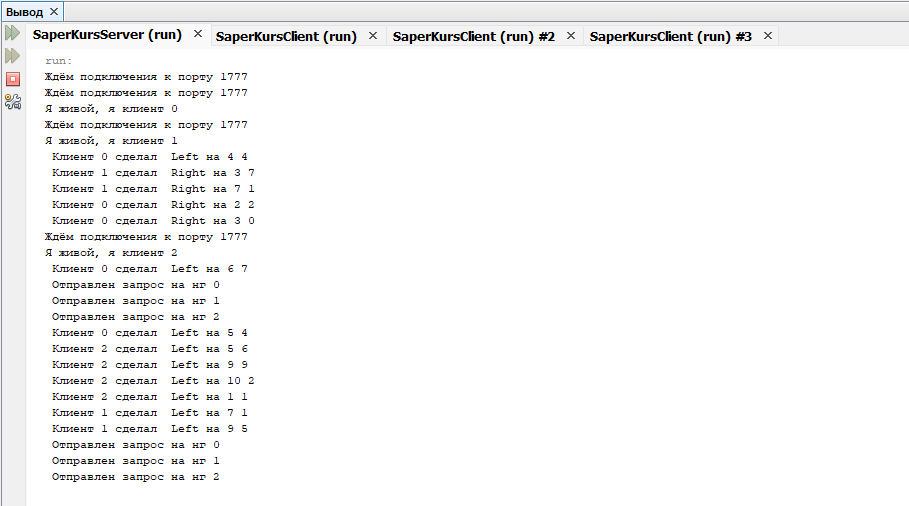




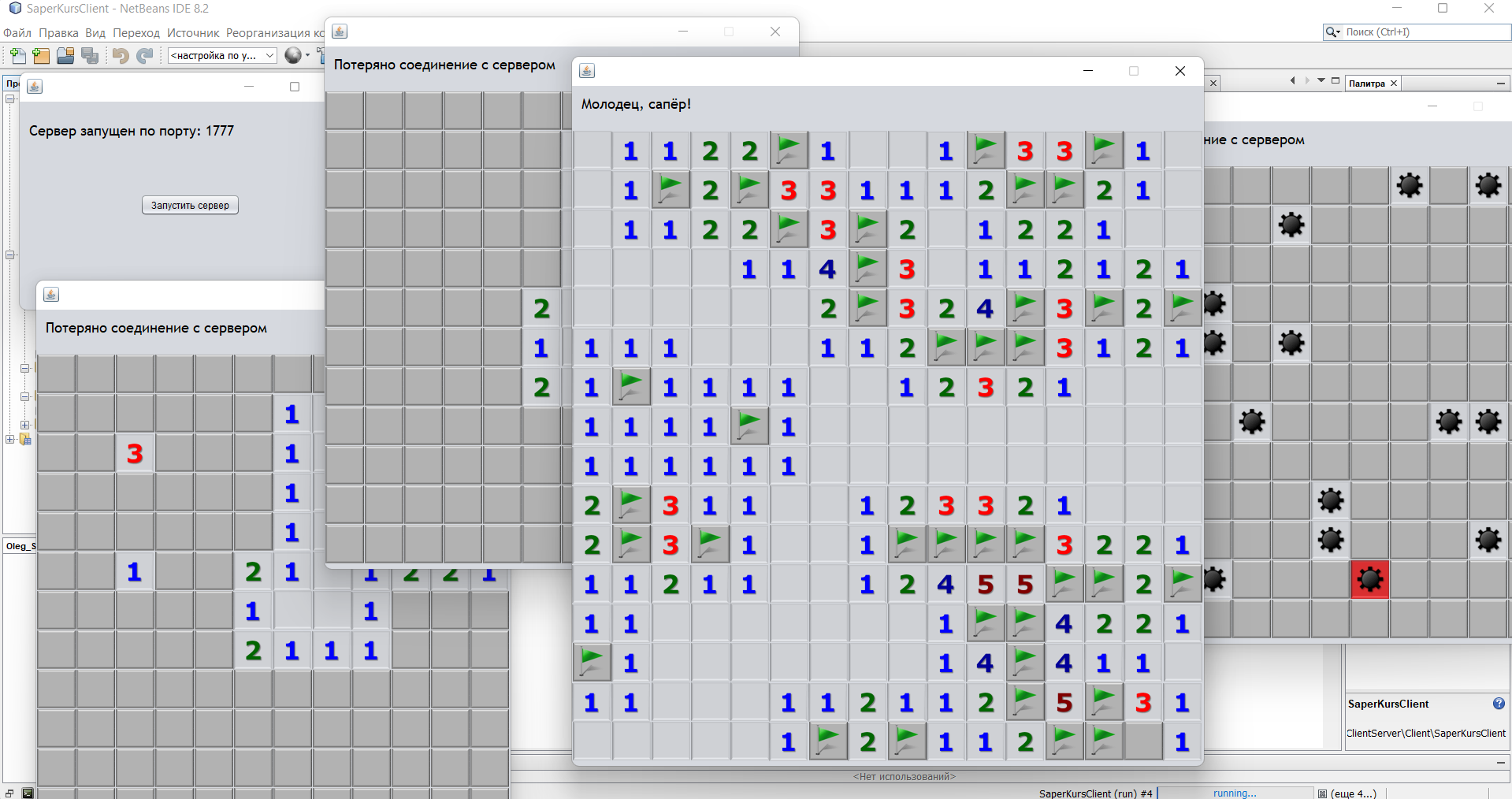


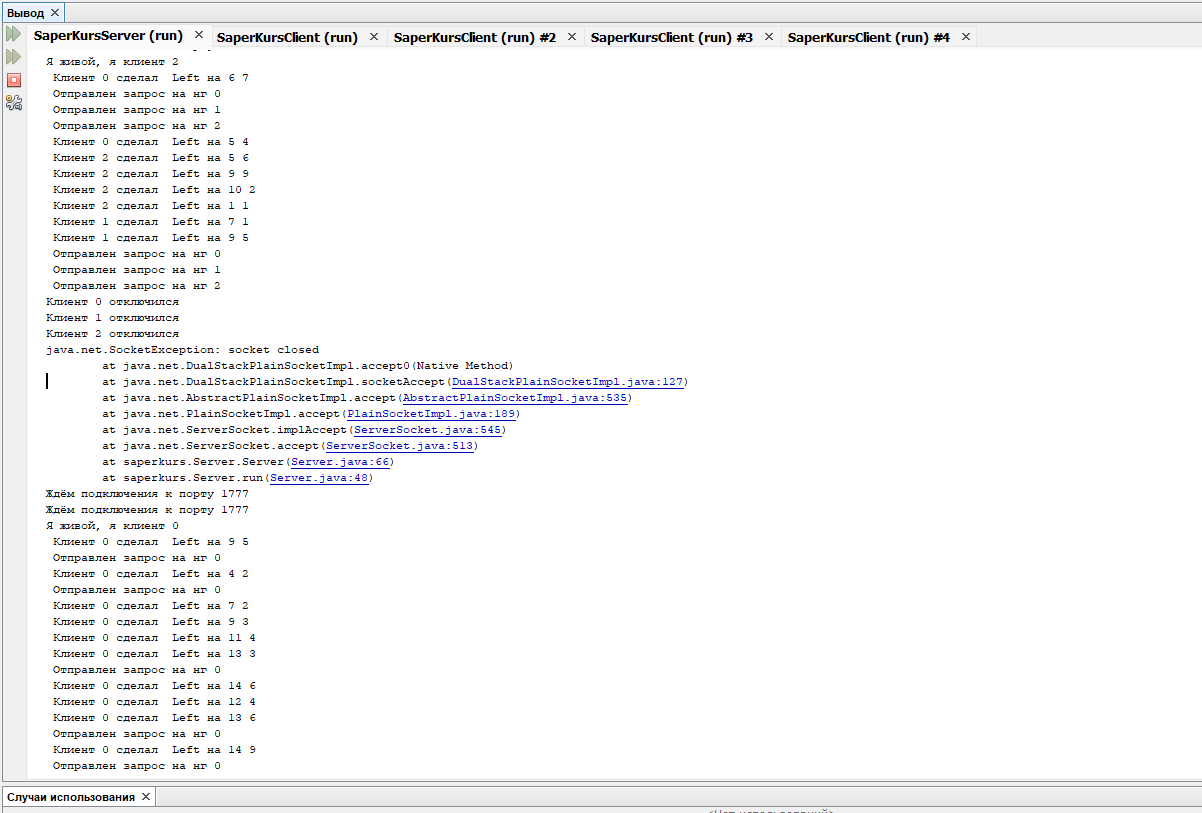
Рисунки 3-5 – Результаты работы консоли





Рисунки 6-7 – Подключение третьего клиента





Рисунки 8-9 – Перезапуск сервера с выбором другого уровня сложности и подключением нового клиента

По итогам тестирования было выявлено, что многомодульное клиент-серверное игровое приложение «Сапёр» успешно выполняет свою работу.

.

# Заключение

Целью курсового проектирования являлось разработка многомодульного клиент-серверного игрового приложения «Сапёр» на языке Java с графическим пользовательским интерфейсом.

При выполнении данной курсовой работы были закреплены знания, полученные при прослушании курса, и приобретены практические навыки разработки многомодульных приложений на языке Java.

# Литература

1. Шилдт, Г. Swing. Руководство для начинающих. – М.: Вильямс, 2007. – 704 с.: ил.
2. Шилдт, Г. Java 8. Полное руководство. – 9-е издание – М.: Вильямс, 2015. – 1376 с.: ил.
3. Эккель, Б. Философия Java. – 4-е полное издание – СПб.: Питер, 2016. – 1168 с.: ил.
4. Справочная система компании Oracle. Материалы сайта https://docs.oracle.com/ – Режим доступа: <https://docs.oracle.com/>

# Приложение А. Листинг программы

Приложение **SaperKursServer**

Форма «JFrame» SaperUI

package saperkurs;

import java.awt.event.WindowAdapter;

import java.awt.event.WindowEvent;

import java.util.logging.Level;

import java.util.logging.Logger;

public class SaperUI extends javax.swing.JFrame {

private int bombs;

private int cols;

private int rows;

private Server serv;

public SaperUI() {

initComponents();

this.addWindowListener(new WindowAdapter()

{

@Override

public void windowClosing(WindowEvent e)

{

if(serv != null){

if(serv.IsWorking()){

serv.StopServer();

}

}

System.exit(0);

}

});

}

@SuppressWarnings("unchecked")

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

private void initComponents() {

jDialog1 = new javax.swing.JDialog((java.awt.Dialog)null);

jPanel1 = new javax.swing.JPanel();

jLabel2 = new javax.swing.JLabel();

jTextField1 = new javax.swing.JTextField();

jLabel3 = new javax.swing.JLabel();

jPanel3 = new javax.swing.JPanel();

jButton1 = new javax.swing.JButton();

jButton2 = new javax.swing.JButton();

jButton3 = new javax.swing.JButton();

jPanel2 = new javax.swing.JPanel();

jLabel1 = new javax.swing.JLabel();

jButton4 = new javax.swing.JButton();

jLabel2.setText("Выберите уровень сложности");

jTextField1.setHorizontalAlignment(javax.swing.JTextField.CENTER);

jTextField1.setText("1777");

jTextField1.addActionListener(new java.awt.event.ActionListener() {

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

jTextField1ActionPerformed(evt);

}

});

jLabel3.setText("Порт:");

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel1Layout.createSequentialGroup()

.addContainerGap(64, Short.MAX\_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED\_SIZE, 181, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel3))

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

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, 56, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap())

);

jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addGap(26, 26, 26)

.addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED\_SIZE, 33, javax.swing.GroupLayout.PREFERRED\_SIZE)

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

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel3))

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

);

jButton1.setText("Легко");

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

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

jButton1ActionPerformed(evt);

}

});

jButton2.setText("Норм");

jButton2.addActionListener(new java.awt.event.ActionListener() {

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

jButton2ActionPerformed(evt);

}

});

jButton3.setText("Saper");

jButton3.addActionListener(new java.awt.event.ActionListener() {

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

jButton3ActionPerformed(evt);

}

});

javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);

jPanel3.setLayout(jPanel3Layout);

jPanel3Layout.setHorizontalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel3Layout.createSequentialGroup()

.addContainerGap(29, Short.MAX\_VALUE)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(jButton3, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton2, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addGap(27, 27, 27))

);

jPanel3Layout.setVerticalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel3Layout.createSequentialGroup()

.addContainerGap()

.addComponent(jButton1)

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

.addComponent(jButton2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 12, Short.MAX\_VALUE)

.addComponent(jButton3)

.addContainerGap())

);

javax.swing.GroupLayout jDialog1Layout = new javax.swing.GroupLayout(jDialog1.getContentPane());

jDialog1.getContentPane().setLayout(jDialog1Layout);

jDialog1Layout.setHorizontalGroup(

jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jDialog1Layout.createSequentialGroup()

.addContainerGap(43, Short.MAX\_VALUE)

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(39, 39, 39))

.addGroup(jDialog1Layout.createSequentialGroup()

.addGap(133, 133, 133)

.addComponent(jPanel3, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

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

);

jDialog1Layout.setVerticalGroup(

jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jDialog1Layout.createSequentialGroup()

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(18, 18, 18)

.addComponent(jPanel3, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap(55, Short.MAX\_VALUE))

);

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

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

jLabel1.setFont(new java.awt.Font("Trebuchet MS", 0, 18)); // NOI18N

jLabel1.setText("Saper coop server");

javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);

jPanel2.setLayout(jPanel2Layout);

jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel2Layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 389, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap(31, Short.MAX\_VALUE))

);

jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel2Layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 48, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap(25, Short.MAX\_VALUE))

);

jButton4.setText("Запустить сервер");

jButton4.addActionListener(new java.awt.event.ActionListener() {

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

jButton4ActionPerformed(evt);

}

});

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(jPanel2, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addContainerGap())

.addGroup(layout.createSequentialGroup()

.addGap(153, 153, 153)

.addComponent(jButton4)

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

);

layout.setVerticalGroup(

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

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jPanel2, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(32, 32, 32)

.addComponent(jButton4)

.addContainerGap(119, Short.MAX\_VALUE))

);

pack();

setLocationRelativeTo(null);

}// </editor-fold>

private void StartServer(){

if(serv != null){

if(serv.IsWorking()){

serv.StopServer();

}

}

serv = new Server(cols, rows, bombs, Integer.parseInt(jTextField1.getText()));

serv.start();

try {

Thread.sleep(55);

} catch (InterruptedException ex) {

Logger.getLogger(SaperUI.class.getName()).log(Level.SEVERE, null, ex);

}

if(serv.IsWorking()){

jLabel1.setText("Сервер запущен по порту: " + jTextField1.getText() );

jLabel1.repaint();

}

else

{

jLabel1.setText("Ошибка при запуске сервера");

jLabel1.repaint();

}

}

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

cols = 9;

rows = 9;

bombs = 10;

StartServer();

if(serv.IsWorking()){

jDialog1.dispose();

repaint();

}

}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

cols = 12;

rows = 12;

bombs = 20;

StartServer();

if(serv.IsWorking()){

jDialog1.dispose();

repaint();

}

}

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

cols = 16;

rows = 16;

bombs = 40;

StartServer();

if(serv.IsWorking()){

jDialog1.dispose();

repaint();

}

}

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {

jDialog1.pack();

jDialog1.setLocationRelativeTo(null);

jDialog1.setVisible (true);

}

private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* 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(SaperUI.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

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

} catch (IllegalAccessException ex) {

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

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(SaperUI.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 SaperUI().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton jButton1;

private javax.swing.JButton jButton2;

private javax.swing.JButton jButton3;

private javax.swing.JButton jButton4;

private javax.swing.JDialog jDialog1;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JPanel jPanel1;

private javax.swing.JPanel jPanel2;

private javax.swing.JPanel jPanel3;

private javax.swing.JTextField jTextField1;

// End of variables declaration

}

Класс Server.java

package saperkurs;

import java.io.IOException;

import java.net.ServerSocket;

import java.net.Socket;

import java.util.ArrayList;

import java.util.logging.Level;

import java.util.logging.Logger;

import sweeper.Coord;

import sweeper.Game;

public class Server extends Thread {

static ArrayList<SocketThread> sockets;

ServerSocket servSocket;

private static Game game;

private int bombs;

private int cols;

private int rows;

int port;

private String sIp;

public Server(int cols, int rows, int bombs, int port) {

this.cols = cols;

this.rows = rows;

this.bombs = bombs;

// sIp = ip;

this.port = port;

}

public void run() {

Server();

}

public void Server() {

// port = 1777;

try {

servSocket = new ServerSocket(port);

sockets = new ArrayList<SocketThread>();

NewGame(cols, rows, bombs);

int countSocket = 0;

while (true) {

System.out.println("Ждём подключения к порту " + port);

Socket fromClientSocket = servSocket.accept();

if(IsWorking()){

sockets.add(new SocketThread(fromClientSocket, countSocket, game));

sockets.get(countSocket).start();

countSocket++;

}

else{

return;

}

}

} catch (IOException ex) {

ex.printStackTrace(System.out);

}

}

public Game NewGame(int cols, int rows, int bombs){

game = new Game(cols, rows, bombs);

game.StartSolo();

return game;

}

public static void NewGameRequest(Game game) {

Server.game = game;

for(int i = 0; i < sockets.size(); i++){

if(sockets.get(i).IsStatus() == true){

System.out.println(" Отправлен запрос на нг " + i);

sockets.get(i).SendNewGame(game);

}

}

}

public static void PressButton(int number, String button, Coord coord) {

System.out.println(" Клиент " + number + " сделал " + " " + button + " на " + coord.x + " " + coord.y);

for(int i = 0; i < sockets.size(); i++){

if(i != number){

if(sockets.get(i).IsStatus() == true){

// System.out.println(" Обнаружен клиент " + i);

sockets.get(i).PressButton(number, button, coord);

}

}

}

}

public void StopServer() {

if(IsWorking()){

for(int i = 0; i < sockets.size(); i++){

if(sockets.get(i).IsStatus() == true){

sockets.get(i).StopConnection();

}

}

}

try {

servSocket.close();

} catch (IOException ex) {

Logger.getLogger(Server.class.getName()).log(Level.SEVERE, null, ex);

}

}

public boolean IsWorking(){

if((servSocket != null))

return !servSocket.isClosed();

else

return false;

}

}

Класс SocketThread.java

package saperkurs;

import java.io.IOException;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.net.Socket;

import java.util.logging.Level;

import java.util.logging.Logger;

import sweeper.Coord;

import sweeper.Game;

import sweeper.Ranges;

class SocketThread extends Thread

{

private Socket fromClientSocket;

private int number;

private ObjectOutputStream oos;

private ObjectInputStream ois;

private boolean status;

Game game;

public boolean IsStatus() {

return status;

}

public SocketThread(Socket fromClientSocket, int number, Game game) {

this.fromClientSocket = fromClientSocket;

this.number = number;

this.game = game;

status = true;

}

@Override

public void run() {

InteractionWithTheClient(game);

}

public void PressButton(int number, String button, Coord coord) {

try {

oos.writeObject(button);

oos.writeObject(coord);

} catch (IOException ex) {

Logger.getLogger(Server.class.getName()).log(Level.SEVERE, null, ex);

}

}

public void InteractionWithTheClient(Game game) {

try {

oos = new ObjectOutputStream(fromClientSocket.getOutputStream());

ois = new ObjectInputStream(fromClientSocket.getInputStream());

oos.writeInt(Ranges.GetSize().x);

oos.writeInt(Ranges.GetSize().y);

oos.writeObject(game);

String str;

Coord coord = new Coord(0,0);

System.out.println("Я живой, я клиент " + number);

while ((str = (String) ois.readObject()) != null) {

switch(str){

case "Left":

coord = (Coord) ois.readObject();

Server.PressButton(number, str, coord);

break;

case "Right":

coord = (Coord) ois.readObject();

Server.PressButton(number, str, coord);

break;

case "NgPls":

int cols = Ranges.GetSize().x;

int rows = Ranges.GetSize().y;

int bombs = game.GetBomb().GetTotalBombs();

game = new Game(cols, rows, bombs);

game.StartSolo();

Server.NewGameRequest(game);

break;

case "Exit":

if(status)

StopConnection();

return;

default:

System.out.println(" Получен необработанный ответ: " + str);

break;

}

}

} catch (IOException ex) {

Logger.getLogger(SocketThread.class.getName()).log(Level.SEVERE, null, ex);

}

catch (ClassNotFoundException ex) {

Logger.getLogger(SocketThread.class.getName()).log(Level.SEVERE, null, ex);

}

}

public void StopConnection(){

if(status == true){

try {

status = false;

oos.writeObject("Exit");

try {

Thread.sleep(120);

} catch (InterruptedException ex) {

Logger.getLogger(SocketThread.class.getName()).log(Level.SEVERE, null, ex);

}

oos.close();

ois.close();

fromClientSocket.close();

System.out.println("Клиент " + number + " отключился");

} catch (IOException ex) {

Logger.getLogger(SocketThread.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

void SendNewGame(Game game) {

try {

oos.writeObject("Ng Guys");

oos.writeObject(game);

} catch (IOException ex) {

Logger.getLogger(SocketThread.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

Приложение **SaperKursClient**

Форма «JFrame» SaperUI.java

package saperkurs;

import javax.swing.ImageIcon;

import java.awt.Image;

import java.awt.Graphics;

import java.awt.event.MouseAdapter;

import java.awt.event.MouseEvent;

import java.awt.event.WindowAdapter;

import java.awt.event.WindowEvent;

import sweeper.Box;

import sweeper.Coord;

import sweeper.Game;

import sweeper.Ranges;

import sweeper.GameState;

public class SaperUI extends javax.swing.JFrame {

private static Game game;

private int bombs = 10;

private int cols = 9;

private int rows = 9;

private final int IMAGE\_SIZE = 50;

private Client client;

public SaperUI() {

InitDialog();

jDialog1.pack();

jDialog1.setLocationRelativeTo(null);

jDialog1.setVisible(true);

jDialog1.addWindowListener(new WindowAdapter()

{

@Override

public void windowClosing(WindowEvent e)

{

//e.getWindow().dispose();

System.exit(0);

}

}

);

}

/\*\*

\* 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.

\*/

public void InitDialog(){

jDialog1 = new javax.swing.JDialog((java.awt.Dialog)null);

jPanel3 = new javax.swing.JPanel();

jTextField1 = new javax.swing.JTextField();

jLabel3 = new javax.swing.JLabel();

jTextField2 = new javax.swing.JTextField();

jLabel4 = new javax.swing.JLabel();

jLabel5 = new javax.swing.JLabel();

jPanel4 = new javax.swing.JPanel();

jLabel2 = new javax.swing.JLabel();

jButton1 = new javax.swing.JButton();

jPanel1 = new javax.swing.JPanel(){

@Override

protected void paintComponent(Graphics g){

super.paintComponent(g);

for(Coord coord : Ranges.GetAllCoords()){

g.drawImage((Image)game.GetBox(coord).image,

coord.x \* IMAGE\_SIZE, coord.y \* IMAGE\_SIZE, this);

}

}

};

jPanel2 = new javax.swing.JPanel();

jLabel1 = new javax.swing.JLabel();

jTextField1.setHorizontalAlignment(javax.swing.JTextField.CENTER);

jTextField1.setText("localhost");

jLabel3.setText(":");

jTextField2.setHorizontalAlignment(javax.swing.JTextField.CENTER);

jTextField2.setText("1777");

jTextField2.addActionListener(new java.awt.event.ActionListener() {

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

jTextField2ActionPerformed(evt);

}

});

jLabel4.setText("Адрес");

jLabel5.setText("Порт");

javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);

jPanel3.setLayout(jPanel3Layout);

jPanel3Layout.setHorizontalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel3Layout.createSequentialGroup()

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

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel3Layout.createSequentialGroup()

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, 131, javax.swing.GroupLayout.PREFERRED\_SIZE)

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

.addComponent(jLabel3)

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

.addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED\_SIZE, 60, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(jPanel3Layout.createSequentialGroup()

.addGap(44, 44, 44)

.addComponent(jLabel4)

.addGap(83, 83, 83)

.addComponent(jLabel5)))

.addGap(30, 30, 30))

);

jPanel3Layout.setVerticalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel3Layout.createSequentialGroup()

.addGap(21, 21, 21)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel4)

.addComponent(jLabel5))

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

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel3)

.addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addContainerGap(34, Short.MAX\_VALUE))

);

jLabel2.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);

jLabel2.setText("Подключение к серверу");

jLabel2.setToolTipText("");

javax.swing.GroupLayout jPanel4Layout = new javax.swing.GroupLayout(jPanel4);

jPanel4.setLayout(jPanel4Layout);

jPanel4Layout.setHorizontalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel4Layout.createSequentialGroup()

.addContainerGap(31, Short.MAX\_VALUE)

.addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED\_SIZE, 196, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(31, 31, 31))

);

jPanel4Layout.setVerticalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel4Layout.createSequentialGroup()

.addGap(0, 14, Short.MAX\_VALUE)

.addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED\_SIZE, 36, javax.swing.GroupLayout.PREFERRED\_SIZE))

);

jButton1.setText("Подключиться");

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

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

jButton1ActionPerformed(evt);

}

});

javax.swing.GroupLayout jDialog1Layout = new javax.swing.GroupLayout(jDialog1.getContentPane());

jDialog1.getContentPane().setLayout(jDialog1Layout);

jDialog1Layout.setHorizontalGroup(

jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jDialog1Layout.createSequentialGroup()

.addContainerGap(46, Short.MAX\_VALUE)

.addGroup(jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)

.addComponent(jPanel3, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jPanel4, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addGap(45, 45, 45))

.addGroup(jDialog1Layout.createSequentialGroup()

.addGap(116, 116, 116)

.addComponent(jButton1)

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

);

jDialog1Layout.setVerticalGroup(

jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jDialog1Layout.createSequentialGroup()

.addContainerGap(46, Short.MAX\_VALUE)

.addComponent(jPanel4, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(18, 18, 18)

.addComponent(jPanel3, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(30, 30, 30)

.addComponent(jButton1)

.addGap(31, 31, 31))

);

}

@SuppressWarnings("unchecked")

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

private void initComponents() {

jDialog1 = new javax.swing.JDialog();

jPanel3 = new javax.swing.JPanel();

jTextField1 = new javax.swing.JTextField();

jLabel3 = new javax.swing.JLabel();

jTextField2 = new javax.swing.JTextField();

jLabel4 = new javax.swing.JLabel();

jLabel5 = new javax.swing.JLabel();

jPanel4 = new javax.swing.JPanel();

jLabel2 = new javax.swing.JLabel();

jButton1 = new javax.swing.JButton();

jPanel1 = new javax.swing.JPanel(){

@Override

protected void paintComponent(Graphics g){

super.paintComponent(g);

for(Coord coord : Ranges.GetAllCoords()){

g.drawImage((Image)game.GetBox(coord).image,

coord.x \* IMAGE\_SIZE, coord.y \* IMAGE\_SIZE, this);

}

}

};

jPanel2 = new javax.swing.JPanel();

jLabel1 = new javax.swing.JLabel();

jTextField1.setHorizontalAlignment(javax.swing.JTextField.CENTER);

jTextField1.setText("localhost");

jLabel3.setText(":");

jTextField2.setHorizontalAlignment(javax.swing.JTextField.CENTER);

jTextField2.setText("1777");

jTextField2.addActionListener(new java.awt.event.ActionListener() {

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

jTextField2ActionPerformed(evt);

}

});

jLabel4.setText("Адрес");

jLabel5.setText("Порт");

javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);

jPanel3.setLayout(jPanel3Layout);

jPanel3Layout.setHorizontalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel3Layout.createSequentialGroup()

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

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel3Layout.createSequentialGroup()

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, 131, javax.swing.GroupLayout.PREFERRED\_SIZE)

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

.addComponent(jLabel3)

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

.addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED\_SIZE, 60, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(jPanel3Layout.createSequentialGroup()

.addGap(44, 44, 44)

.addComponent(jLabel4)

.addGap(83, 83, 83)

.addComponent(jLabel5)))

.addGap(30, 30, 30))

);

jPanel3Layout.setVerticalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel3Layout.createSequentialGroup()

.addGap(21, 21, 21)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel4)

.addComponent(jLabel5))

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

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel3)

.addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addContainerGap(34, Short.MAX\_VALUE))

);

jLabel2.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);

jLabel2.setText("Подключение к серверу");

jLabel2.setToolTipText("");

javax.swing.GroupLayout jPanel4Layout = new javax.swing.GroupLayout(jPanel4);

jPanel4.setLayout(jPanel4Layout);

jPanel4Layout.setHorizontalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel4Layout.createSequentialGroup()

.addContainerGap(31, Short.MAX\_VALUE)

.addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED\_SIZE, 196, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(31, 31, 31))

);

jPanel4Layout.setVerticalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel4Layout.createSequentialGroup()

.addGap(0, 14, Short.MAX\_VALUE)

.addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED\_SIZE, 36, javax.swing.GroupLayout.PREFERRED\_SIZE))

);

jButton1.setText("Подключиться");

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

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

jButton1ActionPerformed(evt);

}

});

javax.swing.GroupLayout jDialog1Layout = new javax.swing.GroupLayout(jDialog1.getContentPane());

jDialog1.getContentPane().setLayout(jDialog1Layout);

jDialog1Layout.setHorizontalGroup(

jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jDialog1Layout.createSequentialGroup()

.addContainerGap(46, Short.MAX\_VALUE)

.addGroup(jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)

.addComponent(jPanel3, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jPanel4, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addGap(45, 45, 45))

.addGroup(jDialog1Layout.createSequentialGroup()

.addGap(116, 116, 116)

.addComponent(jButton1)

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

);

jDialog1Layout.setVerticalGroup(

jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jDialog1Layout.createSequentialGroup()

.addContainerGap(46, Short.MAX\_VALUE)

.addComponent(jPanel4, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(18, 18, 18)

.addComponent(jPanel3, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(30, 30, 30)

.addComponent(jButton1)

.addGap(31, 31, 31))

);

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

setMaximumSize(new java.awt.Dimension(800, 910));

setMinimumSize(new java.awt.Dimension(450, 560));

setPreferredSize(new java.awt.Dimension((Ranges.GetSize().x \* IMAGE\_SIZE) + 20,

(Ranges.GetSize().y \* IMAGE\_SIZE) + 110));

setResizable(false);

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

jPanel1.addMouseListener(new MouseAdapter(){

@Override

public void mousePressed(MouseEvent event){

int x = event.getX() / IMAGE\_SIZE;

int y = event.getY() / IMAGE\_SIZE;

Coord coord = new Coord(x, y);

if(game.GetState() == GameState.PLAYED){

if (event.getButton() == MouseEvent.BUTTON1){

if(client.IsConnected())

client.PressButton("Left", coord);

game.PressLeftButton (coord);

}

if (event.getButton() == MouseEvent.BUTTON3){

if(client.IsConnected())

client.PressButton("Right", coord);

game.PressRightButton (coord);

}

jPanel1.repaint();

if(game.GetState() == GameState.BOMBED){

if(client.IsConnected()){

client.NewGameRequest();

jLabel1.setText(GetMessage(client.IsConnected(), game));

game.SetState(GameState.READY);

return;

}

}

if(game.GetState() == GameState.WINNER){

if(client.IsConnected()){

client.NewGameRequest();

jLabel1.setText(GetMessage(client.IsConnected(), game));

game.SetState(GameState.READY);

return;

}

}

}

jLabel1.setText(GetMessage(client.IsConnected(), game));

if (event.getButton() == MouseEvent.BUTTON2){

if(game.GetState() == GameState.READY){

if(client.IsConnected()){

game = client.GetGame();

game.StartCoop();

jPanel1.repaint();

jLabel1.setText(GetMessage(client.IsConnected(), game));

}

}

}

}

});

jPanel1.setMaximumSize(new java.awt.Dimension(800, 800));

jPanel1.setMinimumSize(new java.awt.Dimension(450, 450));

jPanel1.setName("panel"); // NOI18N

jPanel1.setPreferredSize(new java.awt.Dimension(

Ranges.GetSize().x \* IMAGE\_SIZE,

Ranges.GetSize().y \* IMAGE\_SIZE));

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGap(0, 800, Short.MAX\_VALUE)

);

jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGap(0, 800, Short.MAX\_VALUE)

);

jLabel1.setFont(new java.awt.Font("Trebuchet MS", 0, 18)); // NOI18N

jLabel1.setText("Hello World Saper");

javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);

jPanel2.setLayout(jPanel2Layout);

jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel2Layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel1)

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

);

jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel2Layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel1)

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

);

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(jPanel2, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addContainerGap())

.addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT\_SIZE, 800, Short.MAX\_VALUE)

);

layout.setVerticalGroup(

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

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jPanel2, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(18, 18, 18)

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 800, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap(32, Short.MAX\_VALUE))

);

pack();

setLocationRelativeTo(null);

}// </editor-fold>

private void jTextField2ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

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

client = new Client(jTextField1.getText(), Integer.parseInt(jTextField2.getText()));

client.ClientStart();

if(client.IsConnected())

{

jDialog1.dispose();

game = client.GetGame();

game.StartCoop();

client.start();

SetImages();

initComponents();

setVisible(true);

client.SetPanel(jPanel1);

client.SetLabel(jLabel1);

this.addWindowListener(new WindowAdapter()

{

@Override

public void windowClosing(WindowEvent e)

{

client.StopConnection();

// e.getWindow().dispose();

System.exit(0);

}

});

setVisible(true);

}

else{

jLabel2.setText("Ошибка подключения");

}

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* 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(SaperUI.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

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

} catch (IllegalAccessException ex) {

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

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(SaperUI.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 SaperUI();

}

});

}

public static String GetMessage(boolean isConnected, Game game){

if(isConnected){

switch(game.GetState()){

case PLAYED : return "Сапёр ошибается лишь однажды";

case BOMBED : return "Сапёр ошибся...";

case WINNER : return "Молодец, сапёр!";

case READY : return "Новая игра готова. Нажмите СКМ.";

default : return "Hello World";

}

}

else

return "Потеряно соединение с сервером";

}

private Image GetImage (String name)

{

String filename = "/res/img/" + name + ".png";

// System.out.println(filename);

ImageIcon icon = new ImageIcon(getClass().getResource(filename));

return icon.getImage();

}

public javax.swing.JPanel GetPanel(){

return jPanel1;

}

private void SetImages(){

for(Box box : Box.values())

box.image = GetImage(box.name().toLowerCase());

}

// Variables declaration - do not modify

private javax.swing.JButton jButton1;

private javax.swing.JDialog jDialog1;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JLabel jLabel5;

private javax.swing.JPanel jPanel1;

private javax.swing.JPanel jPanel2;

private javax.swing.JPanel jPanel3;

private javax.swing.JPanel jPanel4;

private javax.swing.JTextField jTextField1;

private javax.swing.JTextField jTextField2;

// End of variables declaration

}

Класс Client.java

package saperkurs;

import java.io.IOException;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.net.InetSocketAddress;

import java.net.Socket;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.JLabel;

import javax.swing.JPanel;

import javax.swing.SwingUtilities;

import sweeper.Coord;

import sweeper.Game;

import sweeper.GameState;

public class Client extends Thread {

private ObjectOutputStream oos;

private ObjectInputStream ois;

private Game game;

private JLabel label;

private JPanel panel;

private Socket socket;

private boolean status;

private int cols;

private int rows;

private String sIp;

private int port;

public Client(String ip, int port) {

sIp = ip;

this.port = port;

}

public void SetGame(Game game) {

this.game = game;

}

public Game GetGame() {

return game;

}

public void SetLabel(JLabel label) {

this.label = label;

}

public void SetPanel(JPanel panel) {

this.panel = panel;

}

public void run() {

try {

WaitingForAction();

} catch (Exception ex) {

Logger.getLogger(Client.class.getName()).log(Level.SEVERE, null, ex);

}

}

public boolean IsConnected() {

if((socket != null) && (!socket.isClosed()))

return socket.isConnected();

else

return false;

}

public void ClientStart() {

try {

// socket = new Socket(sIp, port);

socket = new Socket();

socket.connect(new InetSocketAddress(sIp, port), 666);

if(IsConnected()){

oos = new ObjectOutputStream(socket.getOutputStream());

ois = new ObjectInputStream(socket.getInputStream());

try {

System.out.println("Client is started");

cols = ois.readInt();

rows = ois.readInt();

game = new Game(cols, rows);

game = (Game)ois.readObject();

System.out.println(game.toString());

status = true;

} catch (ClassNotFoundException ex) {

Logger.getLogger(Client.class.getName()).log(Level.SEVERE, null, ex);

}

}

} catch (IOException ex) {

Logger.getLogger(Client.class.getName()).log(Level.SEVERE, null, ex);

}

}

public void NewGameRequest(){

if(IsConnected()){

try {

oos.writeObject("NgPls");

} catch (IOException ex) {

Logger.getLogger(Client.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

private void WaitingForAction() throws Exception {

Coord coord = new Coord(0, 0);

String str = "123";

while ((str = (String)ois.readObject()) != null) {

switch(str){

case "Left":

coord = (Coord) ois.readObject();

System.out.println("Получен ответ: " + str + " на " + coord.x + " " + coord.y);

game.PressLeftButton(coord);

RepaintPanel();

break;

case "Right":

coord = (Coord) ois.readObject();

System.out.println("Получен ответ: " + str + " на " + coord.x + " " + coord.y);

game.PressRightButton(coord);

RepaintPanel();

break;

case "Ng Guys":

System.out.println("Получен ответ: " + str);

game.SetState(GameState.READY);

game = (Game)ois.readObject();

System.out.println(game.toString());

break;

case "Exit":

if(status)

StopConnection();

return;

default:

System.out.println(" Получен необработанный ответ: " + str);

break;

}

}

}

void RepaintPanel(){

SwingUtilities.invokeLater(new Runnable() {

public void run() {

label.setText(SaperUI.GetMessage(IsConnected(), game));

panel.repaint();

}

});

}

void PressButton(String string, Coord coord) {

if(IsConnected()){

try {

oos.writeObject(string);

oos.writeObject(coord);

} catch (IOException ex) {

Logger.getLogger(Client.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

void StopConnection() {

if(IsConnected()){

try {

status = false;

oos.writeObject("Exit");

try {

Thread.sleep(120);

} catch (InterruptedException ex) {

Logger.getLogger(Client.class.getName()).log(Level.SEVERE, null, ex);

}

oos.close();

ois.close();

socket.close();

} catch (IOException ex) {

Logger.getLogger(Client.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

}

Общий пакет **sweeper**

Класс Bomb.java

package sweeper;

import java.io.Serializable;

public class Bomb implements Serializable {

private Matrix bombMap;

private int totalBombs;

Bomb(int totalBombs){

this.totalBombs = totalBombs;

FixBombsCount();

}

void Start(){

bombMap = new Matrix(Box.ZERO);

for (int i = 0; i < totalBombs; i++) {

PlaceBomb();

}

}

public int GetTotalBombs() {

return totalBombs;

}

Box Get (Coord coord){

return bombMap.Get(coord);

}

private void FixBombsCount (){

int maxBombs = Ranges.GetSize().x \* Ranges.GetSize().y / 2;

if(totalBombs > maxBombs)

totalBombs = maxBombs;

}

private void PlaceBomb(){

while(true){

Coord rCoord = Ranges.GetRandomCoord();

if(bombMap.Get(rCoord) == Box.BOMB)

continue;

bombMap.Set(rCoord, Box.BOMB);

IncNumbersAroundBomb(rCoord);

break;

}

}

private void IncNumbersAroundBomb(Coord coord){

/\*for (Coord around : Ranges.GetCoordsAround(coord))

if(Box.BOMB != bombMap.Get(around))

bombMap.Set(around, bombMap.Get(around).GetNextNumberBox()); \*/

Ranges.GetCoordsAround(coord).stream().filter((around) -> (Box.BOMB != bombMap.Get(around))).forEachOrdered((around) -> {

bombMap.Set(around, bombMap.Get(around).GetNextNumberBox());

});

}

}

Перечисление Box.java

package sweeper;

import java.io.Serializable;

public enum Box implements Serializable {

ZERO,

NUM1,

NUM2,

NUM3,

NUM4,

NUM5,

NUM6,

NUM7,

NUM8,

BOMB,

OPENED,

CLOSED,

FLAGED,

BOMBED,

NOBOMB;

public Object image;

Box GetNextNumberBox (){

return Box.values()[this.ordinal() + 1];

}

int GetNumber(){

return this.ordinal();

}

}

Класс Coord.java

package sweeper;

import java.io.Serializable;

/\*\*

\*

\* @author User

\*/

public class Coord implements Serializable {

public int x;

public int y;

public Coord(int x, int y){

this.x = x;

this.y = y;

}

@Override

public boolean equals(Object obj){

if (obj instanceof Coord){

Coord to = (Coord)obj;

return to.x == x && to.y == y;

}

return super.equals(obj);

}

}

Класс Flag.java

package sweeper;

import java.io.Serializable;

class Flag implements Serializable {

private Matrix flagMap;

private int countOfClosedBoxes;

void Start(){

flagMap = new Matrix(Box.CLOSED);

countOfClosedBoxes = Ranges.GetSize().x \* Ranges.GetSize().y;

}

Box Get (Coord coord){

return flagMap.Get(coord);

}

void SetOpenedToBox(Coord coord) {

flagMap.Set(coord, Box.OPENED);

countOfClosedBoxes--;

}

void ToggleFlagedToBox (Coord coord){

switch(flagMap.Get(coord)){

case FLAGED : SetClosedToBox(coord); break;

case CLOSED : SetFlagedToBox(coord); break;

}

}

void SetFlagedToBox(Coord coord) {

flagMap.Set(coord, Box.FLAGED);

}

void SetClosedToBox(Coord coord) {

flagMap.Set(coord, Box.CLOSED);

}

int GetCountOfClosedBoxes() {

return countOfClosedBoxes;

}

void SetBombedToBox(Coord coord) {

flagMap.Set(coord, Box.BOMBED);

}

void SetOpenedToClosedBombBox(Coord coord) {

if(flagMap.Get(coord) == Box.CLOSED)

flagMap.Set(coord, Box.OPENED);

}

void SetNoBombToFlagedSafeBox(Coord coord) {

if(flagMap.Get(coord) == Box.FLAGED)

flagMap.Set(coord, Box.NOBOMB);

}

int GetCountOfFlagedBoxesAround(Coord coord) {

int count = 0;

for(Coord around : Ranges.GetCoordsAround(coord))

if(flagMap.Get(around) == Box.FLAGED)

count++;

return count;

}

}

Класс Game.java

package sweeper;

import java.io.Serializable;

public class Game implements Serializable {

private Bomb bomb;

private Flag flag;

private GameState gState;

public Game(int cols, int rows){

Ranges.SetSize(new Coord(cols, rows));

}

public Game(int cols, int rows, int bombs){

Ranges.SetSize(new Coord(cols, rows));

bomb = new Bomb(bombs);

flag = new Flag();

}

public Game() {

}

public void StartSolo(){

bomb.Start();

flag.Start();

gState = GameState.PLAYED;

}

public void StartCoop(){

gState = GameState.PLAYED;

}

public Bomb GetBomb() {

return bomb;

}

public Flag GetFlag() {

return flag;

}

public void SetState(GameState gState) {

this.gState = gState;

}

public void SetBomb(Bomb bomb) {

this.bomb = bomb;

}

public void SetFlag(Flag flag) {

this.flag = flag;

}

public GameState GetState() {

return gState;

}

public Box GetBox(Coord coord){

if(flag.Get(coord) == Box.OPENED)

return bomb.Get(coord);

else

return flag.Get(coord);

}

public void PressLeftButton(Coord coord){

if(GameOver())

return;

OpenBox(coord);

CheckWinner();

/// flag.SetOpenedToBox(coord);

}

public void PressRightButton(Coord coord){

if(GameOver())

return;

flag.ToggleFlagedToBox(coord);

}

public void CheckWinner(){

if (gState == GameState.PLAYED)

if (flag.GetCountOfClosedBoxes() == bomb.GetTotalBombs())

gState = GameState.WINNER;

}

public void OpenBox(Coord coord){

switch(flag.Get(coord)){

case OPENED: SetOpenedToClosedBoxesAroundNumber(coord); return;

case FLAGED: return;

case CLOSED:

switch(bomb.Get(coord)){

case ZERO : OpenBoxesAround(coord); return;

case BOMB : OpenBombs(coord); return;

default : flag.SetOpenedToBox(coord); return;

}

}

}

private void OpenBombs(Coord coordBombed) {

gState = GameState.BOMBED;

flag.SetBombedToBox(coordBombed);

for(Coord coord : Ranges.GetAllCoords())

if(bomb.Get(coord) == Box.BOMB)

flag.SetOpenedToClosedBombBox(coord);

else

flag.SetNoBombToFlagedSafeBox(coord);

}

private void OpenBoxesAround(Coord coord) {

flag.SetOpenedToBox(coord);

for(Coord around : Ranges.GetCoordsAround(coord))

OpenBox(around);

}

private boolean GameOver() {

if(gState == GameState.PLAYED)

return false;

System.out.println("Я подорвался");

return true;

}

void SetOpenedToClosedBoxesAroundNumber(Coord coord) {

if(bomb.Get(coord) != Box.BOMB)

if(flag.GetCountOfFlagedBoxesAround(coord) == bomb.Get(coord).GetNumber())

for(Coord around : Ranges.GetCoordsAround(coord))

if (flag.Get(around) == Box.CLOSED)

OpenBox(around);

}

}

Перечисление GameState.java

package sweeper;

import java.io.Serializable;

public enum GameState implements Serializable {

PLAYED,

BOMBED,

WINNER,

READY

}

Класс Matrix.java

package sweeper;

import java.io.Serializable;

class Matrix implements Serializable {

private Box[][] matrix;

Matrix(Box defaultBox){

matrix = new Box[Ranges.GetSize().x][Ranges.GetSize().y];

for(Coord coord : Ranges.GetAllCoords())

matrix [coord.x] [coord.y] = defaultBox;

}

Box Get(Coord coord)

{

if(Ranges.InRange(coord))

return matrix[coord.x][coord.y];

else

return null;

}

void Set(Coord coord, Box box){

if(Ranges.InRange(coord))

matrix[coord.x][coord.y] = box;

}

}

Класс Ranges.java

package sweeper;

import java.io.Serializable;

import java.util.ArrayList;

import java.util.Random;

public class Ranges implements Serializable {

private static Coord size;

private static ArrayList<Coord> allCoords;

private static Random random = new Random();

static void SetSize(Coord oSize){

size = oSize;

allCoords = new ArrayList<Coord>();

for(int y = 0; y < size.y; y++)

for (int x = 0; x < size.x; x++) {

allCoords.add(new Coord(x, y));

}

}

public static Coord GetSize() {

return size;

}

public static ArrayList<Coord> GetAllCoords(){

return allCoords;

}

static boolean InRange(Coord coord){

return coord.x >= 0 && coord.x < size.x &&

coord.y >= 0 && coord.y < size.y;

}

static Coord GetRandomCoord(){

return new Coord(random.nextInt(size.x), random.nextInt(size.y));

}

static ArrayList<Coord> GetCoordsAround(Coord coord){ // для перебора единиц

Coord around;

ArrayList<Coord> coordsList = new ArrayList<Coord>();

for (int x = coord.x - 1; x <= coord.x + 1; x++)

for (int y = coord.y - 1; y <= coord.y + 1; y++) {

if(InRange(around = new Coord(x, y)))

if(!around.equals(coord))

coordsList.add(around);

}

return coordsList;

}

}

# Приложение Б. UML-диаграммы

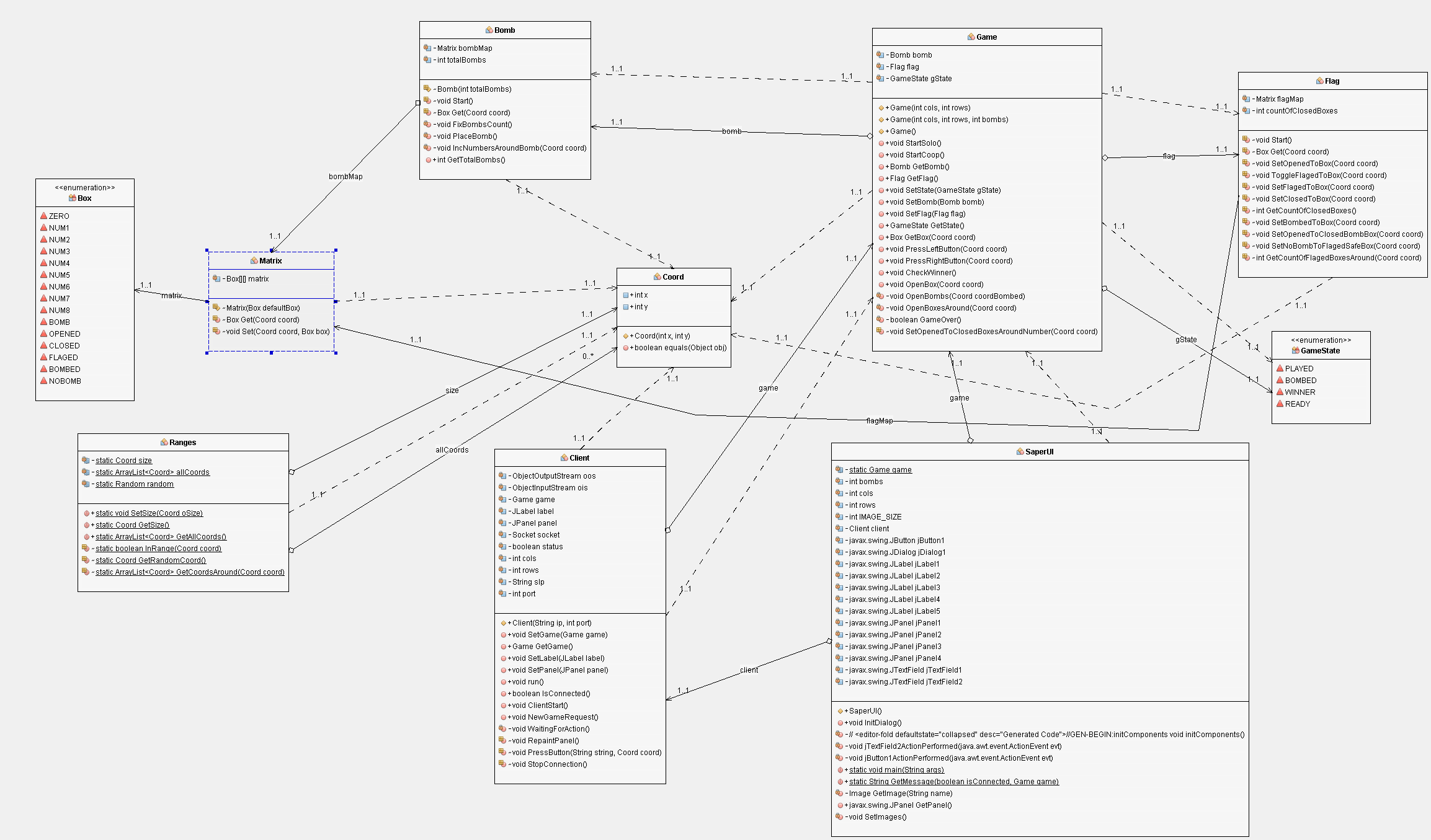


Рисунок 10 – Диаграмма классов клиента

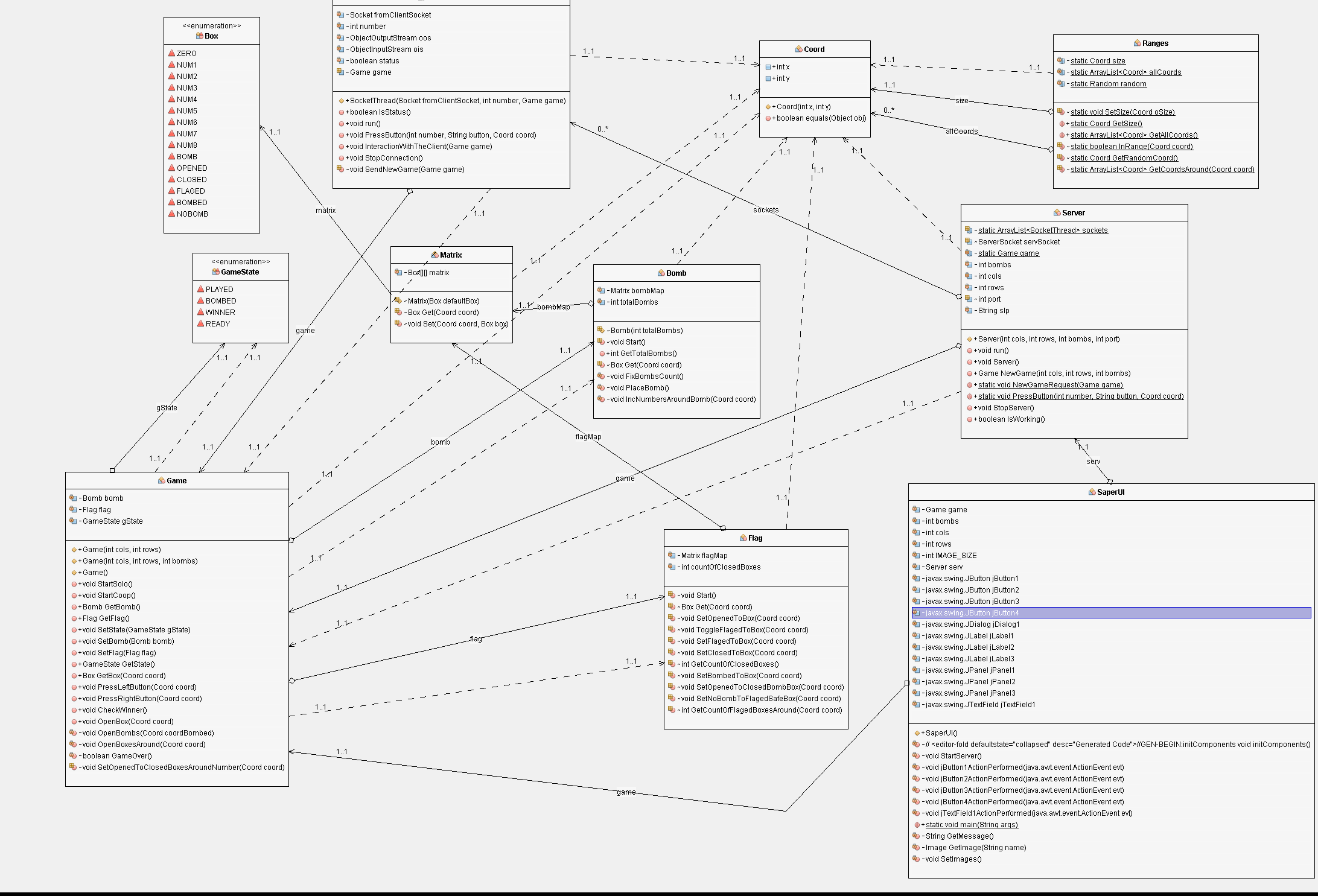


Рисунок 11 – Диаграмма классов сервера

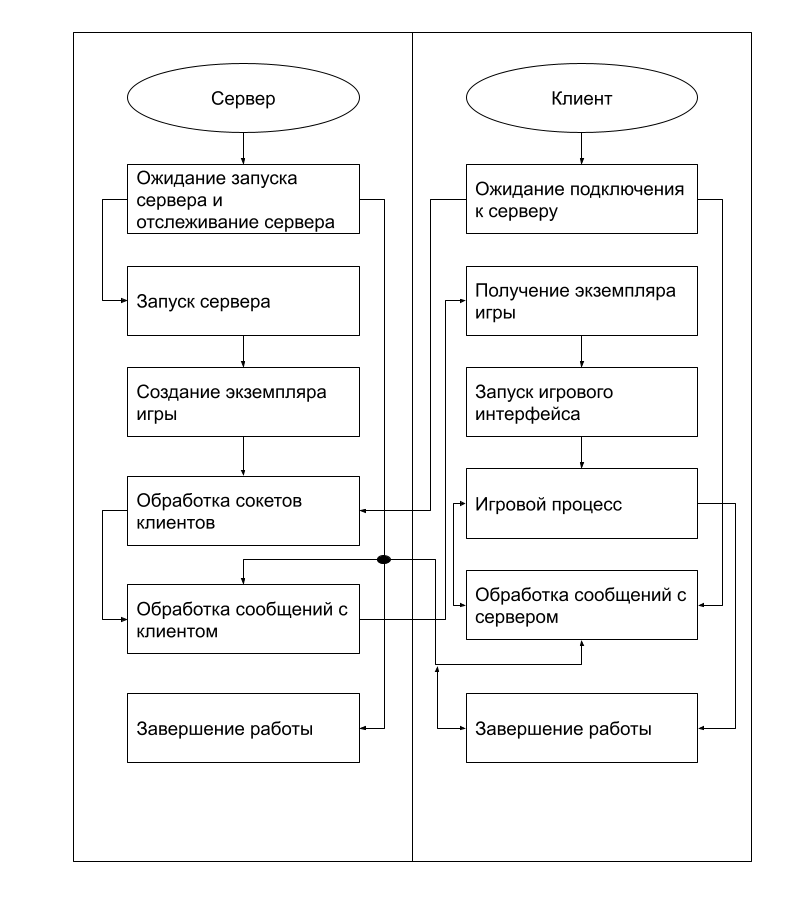


Рисунок 12 – Диаграмма деятельности

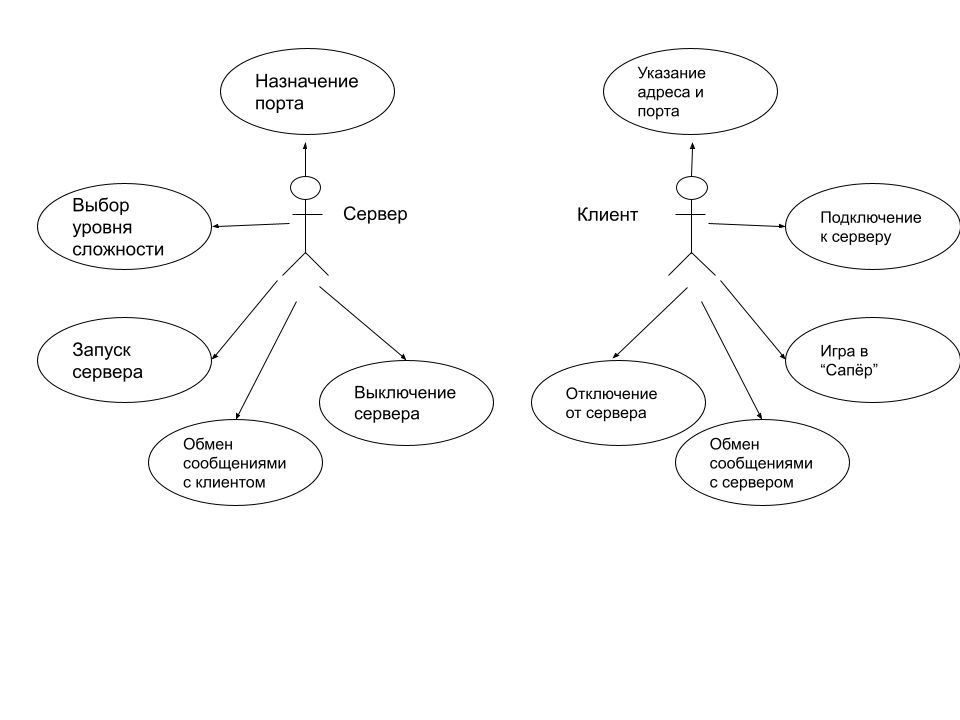


Рисунок 13 – Диаграмма вариантов использования

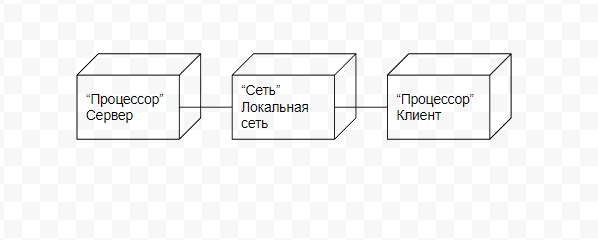


Рисунок 14 – Диаграмма развертывания

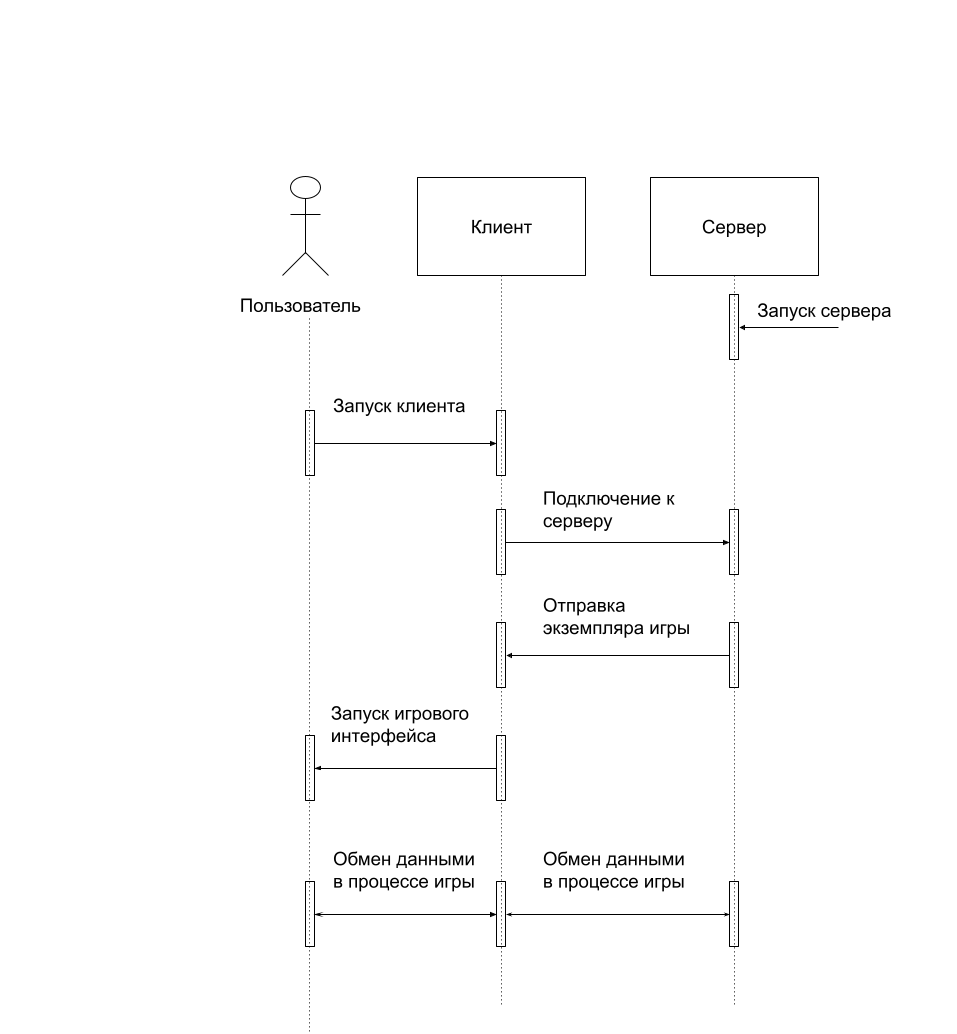


Рисунок 15 – Диаграмма последовательности