**ЛАБОРАТОРНА РОБОТА № 5**

з дисципліни

«Розробка програмного забезпечення на Java»

Шейко Р. О.

Групи КІ-21-2

**Завдання на лабораторну роботу**

1. Реалізувати обробники подій для пунктів меню Вид. Підменю Змінити розміри вікна реалізувати таким чином, щоб при його виборі викликалося діалогове вікно, в якому можна ввести нові розміри вікна.

2. У меню Програми додати додатки з попередніх робіт. Усі попередження і повідомлення про помилки виводити методами класу JOptionPane.

3. Виконати оптимізацію проекту.

1. Реалізувано було обробники подій для пунктів меню Вид. Підменю Змінено розміри вікна реалізувати таким чином, щоб при його виборі викликалося діалогове вікно, в якому можна ввести нові розміри вікна.

|  |  |
| --- | --- |
|  |  |
| Рис 1a. Інтерфейс програми | Рис 1б. Інтерфейс програми. Змінений фон та один із JOptionPane |

2. У меню Програми додати додатки з попередніх робіт. Усі попередження і повідомлення про помилки виводити методами класу JOptionPane.

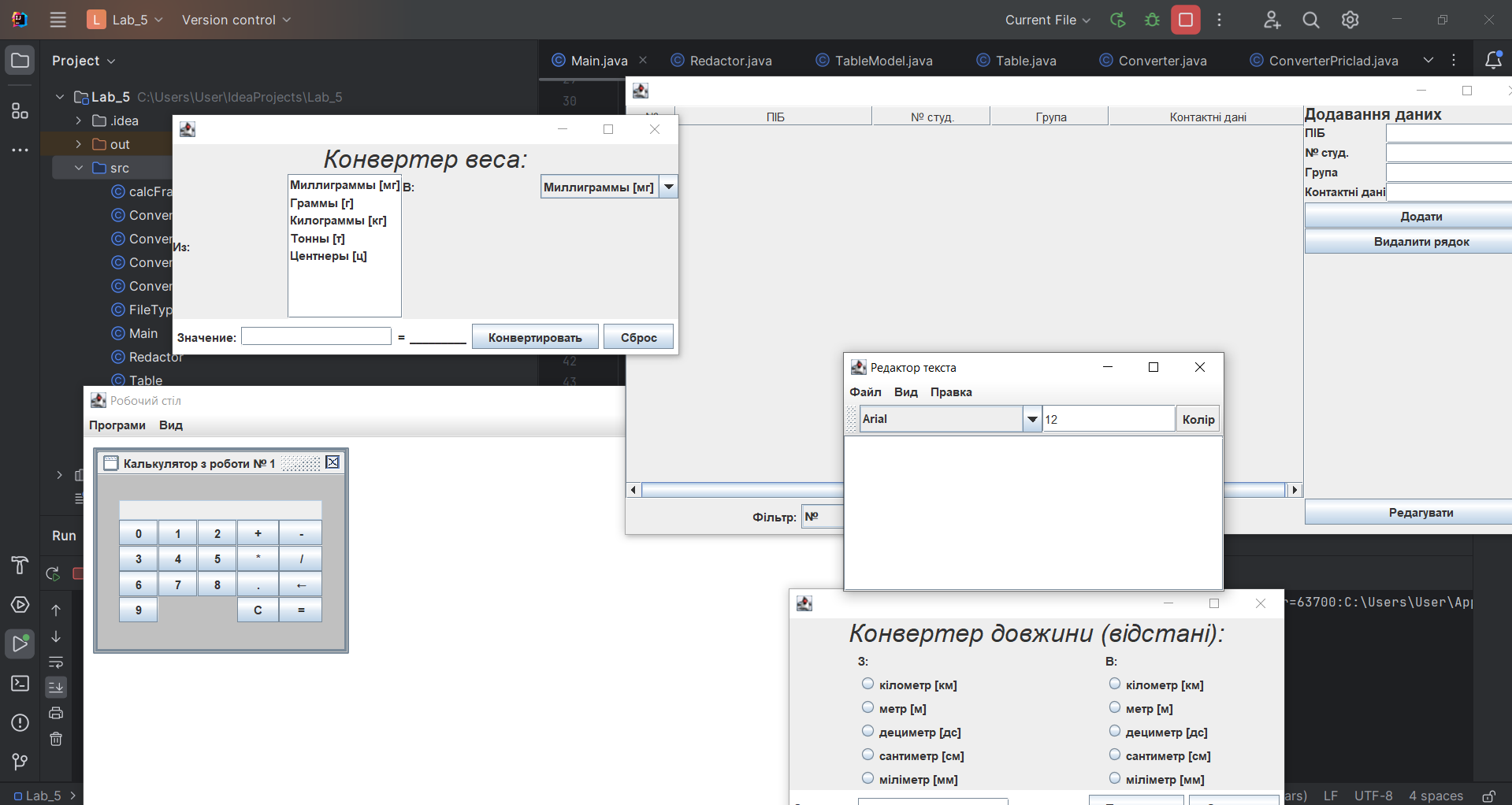


Рис 2. Ілюстрація роботи вcіх робіт

1. Виконати оптимізацію проекту

Для початку в проекті були винесені головні налаштування у клас Main. Зміни зображені нижче.

public static int width = 800;

public static int height = 600;

Також була додана змінна localization, що дозволяє зробити редагування інтефейсу більш швидким, та зміну мови більш лаконічною для програміста.

Код до всієї роботи знаходиться у додатках.

**Висновок**

У цій лабораторній роботі були виконані наступні завдання:

1. Додано пункт "Вирізати" у меню "Правка". Реалізовано вирізання виділеного тексту в JEditorPane та його збереження в буфер обміну.

2. Додано підменю "Стиль тексту" у меню "Правка" з пунктами "Нормальний", "Жирний" і "Курсив". Реалізовано зміну стилю виділеного тексту в JEditorPane відповідно до вибору користувача.

3. Додано JToolBar з компонентами JComboBox для вибору шрифту, JTextField для введення розміру тексту і кнопкою для вибору кольору тексту. Застосовано вибір шрифту, розміру тексту та кольору тексту для всього вмісту JEditorPane.

4. Використано компонент JSpinner для вибору розміру тексту зі значеннями від 1 до 100. Додано стандартний фільтр FileNameExtensionFilter для відкриття та збереження файлів з розширенням .txt.

У результаті виконаних завдань, програма стала більш функціональною, користувач може зручно працювати з текстом, вибирати шрифти, розміри тексту і кольори за допомогою інтуїтивного інтерфейсу, а також зберігати та відкривати файли з розширенням .txt.

Додаток А

*Лістинг файлу «Main.java»*

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class Main {

public static int width = 800;

public static int height = 600;

public static String[] localization = {

"Робочий стіл",

"Програми",

"Вид",

"Калькулятор",//3

"Конвертер приклад",

"Конвертер",

"Таблиця",//6

"Редактор тексту",

"Змінити розміри вікна",

"Введіть нову ширину:",//9

"Введіть нову висоту:",

"Невірний формат введених даних",

"Помилка",//12

"На весь єкран",

"Змінити колір фону",

"Вибір фону",//15

};

public static void main(String[] args) {

JFrame frame = new JFrame(localization[0]);

frame.setSize(width, height);

frame.setLocationRelativeTo(null);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JDesktopPane dp = new JDesktopPane();

frame.add(dp);

JMenuBar mb = new JMenuBar();

JMenu program = new JMenu(localization[1]);

JMenu view = new JMenu(localization[2]);

JMenuItem calcul = new JMenuItem(localization[3]);

JMenuItem converterPr = new JMenuItem(localization[4]);

JMenuItem converter = new JMenuItem(localization[5]);

JMenuItem table = new JMenuItem(localization[6]);

JMenuItem redactor = new JMenuItem(localization[7]);

program.add(calcul);

program.add(converterPr);

program.add(converter);

program.add(table);

program.add(redactor);

// Пункт меню для зміни розмірів вікна

JMenuItem resize = new JMenuItem(localization[8]);

resize.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

String widthInput = JOptionPane.showInputDialog(frame, localization[9]);

String heightInput = JOptionPane.showInputDialog(frame, localization[10]);

try {

int newWidth = Integer.parseInt(widthInput);

int newHeight = Integer.parseInt(heightInput);

frame.setSize(newWidth, newHeight);

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog(frame, localization[11], localization[12], JOptionPane.ERROR\_MESSAGE);

}

}

});

view.add(resize);

JCheckBoxMenuItem fullScreen = new JCheckBoxMenuItem(localization[13]);

JMenuItem background = new JMenuItem(localization[14]);

view.add(fullScreen);

view.add(background);

mb.add(program);

mb.add(view);

frame.setJMenuBar(mb);

frame.setVisible(true);

background.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

Color newColor = JColorChooser.showDialog(null, localization[15], Color.yellow);

dp.setBackground(newColor);

}

});

calcul.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

calcFrame cc = new calcFrame();

dp.add(cc);

cc.setVisible(true);

}

});

converterPr.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

ConverterPriclad cv = new ConverterPriclad();

dp.add(cv);

cv.setVisible(true);

}

});

converter.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

Converter cv = new Converter();

dp.add(cv);

cv.setVisible(true);

}

});

table.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

Table tb = new Table();

dp.add(tb);

tb.setVisible(true);

}

});

redactor.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

Redactor rd = new Redactor();

dp.add(rd);

rd.setVisible(true);

}

});

}

}

*Додаток Б*

*Лістинг файлу «calcFrame.java»*

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class calcFrame extends JInternalFrame {

JTextField tf1;

String currentInput = "";

double currentResult = 0.0;

String operation = "";

calcFrame() {

setBackground(Color.LIGHT\_GRAY);

setTitle("Калькулятор з роботи № 1");

setBounds(10, 10, 260, 210);

setClosable(true);

setResizable(false);

setLayout(new GridBagLayout());

GridBagConstraints gc = new GridBagConstraints();

tf1 = new JTextField(20);

tf1.setEditable(false);

// Создаем кнопки цифр

JButton[] digitButtons = new JButton[10];

for (int i = 0; i < 10; i++) {

digitButtons[i] = new JButton(Integer.toString(i));

}

// Создаем кнопки операций

JButton badd = new JButton("+");

JButton bsub = new JButton("-");

JButton bmult = new JButton("\*");

JButton bdiv = new JButton("/");

JButton beq = new JButton("=");

JButton bdot = new JButton(".");

JButton bcl = new JButton("C");

JButton bbackspace = new JButton("←");

// Размещаем элементы на панели

gc.fill = GridBagConstraints.BOTH;

gc.gridwidth = 5;

gc.gridx = 0;

gc.gridy = 0;

add(tf1, gc);

gc.gridwidth = 1;

for (int i = 0; i < 10; i++) {

int row = 1 + i / 3;

int col = i % 3;

gc.gridx = col;

gc.gridy = row;

add(digitButtons[i], gc);

final int digit = i;

digitButtons[i].addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

currentInput += digit;

tf1.setText(currentInput);

}

});

}

gc.gridx = 3;

gc.gridy = 1;

add(badd, gc);

gc.gridx = 4;

gc.gridy = 1;

add(bsub, gc);

gc.gridx = 3;

gc.gridy = 2;

add(bmult, gc);

gc.gridx = 4;

gc.gridy = 2;

add(bdiv, gc);

gc.gridx = 3;

gc.gridy = 3;

add(bdot, gc);

gc.gridx = 4;

gc.gridy = 3;

add(bbackspace, gc);

gc.gridx = 3;

gc.gridy = 4;

add(bcl, gc);

gc.gridwidth = 2;

gc.gridx = 4;

gc.gridy = 4;

add(beq, gc);

// Добавляем обработчики для кнопок операций

badd.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

performOperation();

operation = "+";

}

});

bsub.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

performOperation();

operation = "-";

}

});

bmult.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

performOperation();

operation = "\*";

}

});

bdiv.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

performOperation();

operation = "/";

}

});

beq.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

performOperation();

operation = "";

}

});

bdot.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if (!currentInput.contains(".")) {

currentInput += ".";

tf1.setText(currentInput);

}

}

});

bcl.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

currentInput = "";

operation = "";

currentResult = 0.0;

tf1.setText("");

}

});

bbackspace.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if (!currentInput.isEmpty()) {

currentInput = currentInput.substring(0, currentInput.length() - 1);

tf1.setText(currentInput);

}

}

});

}

// Метод для выполнения операции

private void performOperation() {

if (!currentInput.isEmpty()) {

double operand = Double.parseDouble(currentInput);

if (operation.equals("+")) {currentResult += operand;

} else if (operation.equals("-")) {currentResult -= operand;

} else if (operation.equals("\*")) {

currentResult \*= operand;

} else if (operation.equals("/")) {

if (operand != 0) {

currentResult /= operand;

} else {

tf1.setText("Ошибка");

return;

}

}

tf1.setText(Double.toString(currentResult));

currentInput = "";

}

}

}

Додаток В

*Лістинг файлу «Conversion.java»*

public class Conversion {

public static double convert(String inputType, String outputType, double value) {

switch (inputType) {

case "mg":

return convertFromMilligrams(value, outputType);

case "g":

return convertFromGrams(value, outputType);

case "kg":

return convertFromKilograms(value, outputType);

case "t":

return convertFromTonnes(value, outputType);

case "c":

return convertFromCentners(value, outputType);

default:

throw new IllegalArgumentException("Неподдерживаемый тип входных данных");

}

}

private static double convertFromMilligrams(double value, String outputType) {

switch (outputType) {

case "mg":

return value;

case "g":

return value / 1000;

case "kg":

return value / 1\_000\_000;

case "t":

return value / 1\_000\_000\_000;

case "c":

return value / 10\_000\_000;

default:

throw new IllegalArgumentException("Неподдерживаемый тип выходных данных");

}

}

private static double convertFromGrams(double value, String outputType) {

switch (outputType) {

case "mg":

return value \* 1000;

case "g":

return value;

case "kg":

return value / 1000;

case "t":

return value / 1\_000\_000;

case "c":

return value / 10\_000;

default:

throw new IllegalArgumentException("Неподдерживаемый тип выходных данных");

}

}

private static double convertFromKilograms(double value, String outputType) {

switch (outputType) {

case "mg":

return value \* 1\_000\_000;

case "g":

return value \* 1000;

case "kg":

return value;

case "t":

return value / 1000;

case "c":

return value / 10;

default:

throw new IllegalArgumentException("Неподдерживаемый тип выходных данных");

}

}

private static double convertFromTonnes(double value, String outputType) {

switch (outputType) {

case "mg":

return value \* 1\_000\_000\_000;

case "g":

return value \* 1\_000\_000;

case "kg":

return value \* 1\_000;

case "t":

return value;

case "c":

return value \* 10;

default:

throw new IllegalArgumentException("Неподдерживаемый тип выходных данных");

}

}

private static double convertFromCentners(double value, String outputType) {

switch (outputType) {

case "mg":

return value \* 10\_000\_000;

case "g":

return value \* 10\_000;

case "kg":

return value \* 100;

case "t":

return value / 10;

case "c":

return value;

default:

throw new IllegalArgumentException("Неподдерживаемый тип выходных данных");

}

}

}

Додаток Г

*Лістинг файлу «ConversionPriclad.java»*

public class ConversionPriclad {

private float km\_conv(float value, String output\_type){

switch(output\_type){

case "km": break;

case "m" : {value \*= 1000; break;}

case "sm": {value \*= 100000; break;}

case "mm": {value \*= 1000000; break;}

case "dc": {value \*= 10000; break;}

}

return value;

}

private float m\_conv(float value, String output\_type){

switch(output\_type){

case "km": {value /= 1000; break;}

case "m" : { break;}

case "dc": {value \*= 10; break;}

case "sm": {value \*= 100; break;}

case "mm": {value \*= 1000; break;}

}

return value;

}

private float dc\_conv(float value, String output\_type){

switch(output\_type){

case "km": {value \*= 0.0001; break;}

case "m" : { value \*= 0.1; break;}

case "dc": { break; }

case "sm": {value \*= 10; break;}

case "mm": {value \*= 100; break;}

}

return value;

}

private float sm\_conv(float value, String output\_type){

switch(output\_type){

case "km": { value \*= 0.00001 ; break;}

case "m" : { value \*= 0.01; break;}

case "dc": { value \*= 0.1; break; }

case "sm": { break;}

case "mm": { value \*= 10; break;}

}

return value;

}

private float mm\_conv(float value, String output\_type){

switch(output\_type) {

case "km": { value \*= 0.000001 ; break;}

case "m" : { value \*= 0.001; break;}

case "dc": { value \*= 0.01; break; }

case "sm": { value \*= 0.1; break;}

case "mm": { break;}

}

return value;

}

public float convert(String input\_type, String output\_type,

float value){

float res = 0;

switch(input\_type){

case "km": {res = km\_conv(value, output\_type); break;}

case "m": {res = m\_conv(value, output\_type); break;}

case "dc": {res = dc\_conv(value, output\_type); break;}

case "sm": {res = sm\_conv(value, output\_type); break;}

case "mm": {res = mm\_conv(value, output\_type); break;}

}

return res; }

}

Додаток Д

*Лістинг файлу «Conversion.java»*

import javax.swing.\*;

import javax.swing.event.ListSelectionEvent;

import javax.swing.event.ListSelectionListener;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class Converter extends JInternalFrame {

public static int width = 600;

public static int height = 450;

public static Font font = new Font("Arial", Font.ITALIC, 26);

public static final String[] Names = {

"Миллиграммы [мг]",

"Граммы [г]",

"Килограммы [кг]",

"Тонны [т]",

"Центнеры [ц]"};

Converter() {

JFrame frame = new JFrame();

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setSize(width, height);

frame.setLocationRelativeTo(null);

frame.setLayout(new GridBagLayout());

GridBagConstraints c = new GridBagConstraints();

JLabel l1 = new JLabel("Конвертер веса:");

l1.setFont(font);

c.gridx = 0;

c.gridy = 0;

c.anchor = GridBagConstraints.NORTH;

c.gridheight = 1;

c.gridwidth = 2;

c.weightx = 1.0;

c.weighty = 0.0;

frame.add(l1, c);

JPanel pan1 = new JPanel();

c.fill = GridBagConstraints.EAST;

c.gridx = 0;

c.gridy = 1;

c.gridheight = 1;

c.gridwidth = 1;

c.weightx = 1.0;

c.weighty = 0.5;

pan1.setLayout(new GridLayout(1, 1));

JLabel l\_in = new JLabel("Из:");

pan1.add(l\_in);

DefaultListModel<String> listModel1 = new DefaultListModel<>();

for (int i = 0; i < Names.length; i++)

listModel1.addElement(Names[i]);

JList<String> list1 = new JList<>(listModel1);

list1.setSelectionMode(ListSelectionModel.SINGLE\_SELECTION);

pan1.add(new JScrollPane(list1));

frame.add(pan1, c);

JPanel pan2 = new JPanel();

c.gridx = 1;

c.gridy = 1;

pan2.setLayout(new GridLayout(1, 1));

JLabel l\_out = new JLabel("В:");

pan2.add(l\_out);

JComboBox<String> comboBox = new JComboBox<>(Names);

pan2.add(comboBox);

frame.add(pan2, c);

JPanel pan3 = new JPanel();

pan3.setBackground(Color.WHITE);

c.fill = GridBagConstraints.BOTH;

c.gridx = 0;

c.gridy = 2;

c.gridwidth = 2;

c.weightx = 1.0;

c.weighty = 0.0;

JLabel l\_value = new JLabel("Значение:");

pan3.add(l\_value);

final JTextField tf\_value = new JTextField(15);

pan3.add(tf\_value);

JLabel l\_equal = new JLabel("=");

pan3.add(l\_equal);

final JLabel l\_result = new JLabel("\_\_\_\_\_\_\_\_\_");

pan3.add(l\_result);

JButton result = new JButton("Конвертировать");

pan3.add(result);

JButton reset = new JButton("Сброс");

pan3.add(reset);

Conversion convert = new Conversion();

list1.addListSelectionListener(new ListSelectionListener() {

@Override

public void valueChanged(ListSelectionEvent e) {

if (!e.getValueIsAdjusting()) {

String selectedUnit1 = list1.getSelectedValue();

// Здесь обработайте выбор единицы веса из первой группы

}

}

});

comboBox.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

String selectedUnit2 = comboBox.getSelectedItem().toString();

// Здесь обработайте выбор единицы веса из второй группы

}

});

result.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

double result = 0;

try {

String inputUnit = getType(list1.getSelectedValue());

String outputUnit = getType(comboBox.getSelectedItem().toString());

result = convert.convert(inputUnit, outputUnit, Double.parseDouble(tf\_value.getText()));

l\_result.setText(Double.toString(result));

} catch (Exception ex) {

l\_result.setText(ex.getMessage());

}

}

});

reset.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

list1.clearSelection();

comboBox.setSelectedIndex(0);

tf\_value.setText("");

l\_result.setText("\_\_\_\_\_\_\_\_\_");

}

});

frame.add(pan3, c);

frame.pack();

frame.setVisible(true);

}

public static String getType(String type){

switch (type)

{

case "Миллиграммы [мг]": return "mg";

case "Граммы [г]": return "g";

case "Килограммы [кг]": return "kg";

case "Тонны [т]": return "t";

case "Центнеры [ц]": return "c";

}

return "error";

}

}

Додаток [Ґ](https://uk.wikipedia.org/wiki/%D2%90)

*Лістинг файлу «ConverterPriclad.java»*

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

// Press Shift twice to open the Search Everywhere dialog and type `show whitespaces`,

// then press Enter. You can now see whitespace characters in your code.

public class ConverterPriclad extends JInternalFrame {

ConverterPriclad() {

JFrame frame = new JFrame();

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setSize(600, 450);

frame.setLocationRelativeTo(null);

frame.setLayout(new GridBagLayout());

GridBagConstraints c = new GridBagConstraints();

JLabel l1 = new JLabel("Конвертер довжини (відстані):");

l1.setFont(new Font("Arial", Font.ITALIC, 26));

c.gridx = 0; c.gridy = 0;

c.anchor = GridBagConstraints.NORTH;

c.gridheight = 1; c.gridwidth = 2;

c.weightx = 1.0; c.weighty = 0.0;

frame.add(l1, c);

JPanel pan1 = new JPanel();

c.fill = GridBagConstraints.EAST;

c.gridx = 0; c.gridy = 1;

c.gridheight = 1; c.gridwidth = 1;

c.weightx = 1.0; c.weighty = 0.5;

pan1.setLayout(new GridLayout(6,1));

JLabel l\_in = new JLabel("З:");

pan1.add(l\_in);

final ButtonGroup group1 = new ButtonGroup();

JRadioButton rb1 = new JRadioButton("кілометр [км]");

rb1.setActionCommand("km");

JRadioButton rb2 = new JRadioButton("метр [м]");

rb2.setActionCommand("m");

JRadioButton rb3 = new JRadioButton("дециметр [дс]");

rb3.setActionCommand("dc");

JRadioButton rb4 = new JRadioButton("сантиметр [см]");

rb4.setActionCommand("sm");

JRadioButton rb5 = new JRadioButton("міліметр [мм]");

rb5.setActionCommand("mm");

group1.add(rb1);

group1.add(rb2);

group1.add(rb3);

group1.add(rb4);

group1.add(rb5);

pan1.add(rb1);

pan1.add(rb2);

pan1.add(rb3);

pan1.add(rb4);

pan1.add(rb5);

frame.add(pan1, c);

JPanel pan2 = new JPanel();

c.gridx = 1;

c.gridy = 1;

pan2.setLayout(new GridLayout(6,1));

JLabel l\_out = new JLabel("B:");

pan2.add(l\_out);

final ButtonGroup group2 = new ButtonGroup();

JRadioButton rb6 = new JRadioButton("кілометр [км]");

rb6.setActionCommand("km");

JRadioButton rb7 = new JRadioButton("метр [м]");

rb7.setActionCommand("m");

JRadioButton rb8 = new JRadioButton("дециметр [дс]");

rb8.setActionCommand("dc");

JRadioButton rb9 = new JRadioButton("сантиметр [см]");

rb9.setActionCommand("sm");

JRadioButton rb10 = new JRadioButton("міліметр [мм]");

rb10.setActionCommand("mm");

group2.add(rb6);

group2.add(rb7);

group2.add(rb8);

group2.add(rb9);

group2.add(rb10);

pan2.add(rb6);

pan2.add(rb7);

pan2.add(rb8);

pan2.add(rb9);

pan2.add(rb10);

frame.add(pan2, c);

JPanel pan3 = new JPanel();

pan3.setBackground(Color.WHITE);

c.fill = GridBagConstraints.BOTH;

c.gridx = 0; c.gridy = 2;

c.gridwidth = 2;

c.weightx = 1.0; c.weighty = 0.0;

JLabel l\_value = new JLabel("Значення:");

pan3.add(l\_value);

final JTextField tf\_value = new JTextField(15);

pan3.add(tf\_value);

JLabel l\_equal = new JLabel("=");

pan3.add(l\_equal);

final JLabel l\_result = new JLabel("\_\_\_\_\_\_\_\_\_");

pan3.add(l\_result);

JButton result = new JButton("Перевести");

pan3.add(result);

JButton reset = new JButton("Скидання");

pan3.add(reset);

ConversionPriclad convert = new ConversionPriclad();

result.addActionListener(new ActionListener(){

@Override

public void actionPerformed(ActionEvent e) {

float result = 0;

try{

result =

convert.convert(group1.getSelection().getActionCommand(),

group2.getSelection().getActionCommand(),

Float.parseFloat(tf\_value.getText()) );

l\_result.setText(Float.toString(result));

}

catch (Exception NullPointerException){

l\_result.setText("Помилка!!!");

}

}});

reset.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

group1.clearSelection();

group2.clearSelection();

tf\_value.setText("");

l\_result.setText("\_\_\_\_\_\_\_\_\_");

}});

frame.add(pan3, c);

frame.pack();

frame.setVisible(true);

}

}

Додаток Е

*Лістинг файлу «FileTypeFilter.java»*

import javax.swing.filechooser.FileFilter;

import java.io.File;

public class FileTypeFilter extends FileFilter {

private final String extension;

private final String description;

public FileTypeFilter(String extension, String

description) {

this.extension = extension;

this.description = description;

}

@Override

public boolean accept(File f) {

if (f.isDirectory()) {

return true;

}

return f.getName().endsWith(extension);

}

@Override

public String getDescription() {

return extension;

}

}

Додаток І

*Лістинг файлу «Redactor.java»*

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.io.\*;

import javax.swing.JSpinner;

import javax.swing.SpinnerModel;

import javax.swing.SpinnerNumberModel;

import javax.swing.filechooser.FileNameExtensionFilter;

public class Redactor extends JMenuItem{

public static int width = 400;

public static int height = 250;

public static String[] localization = {

"Редактор текста",

"Файл",

"Відкрити",

"Зберегти",//3

"Закрити",

"Вид",

"Зафіксувати розмір вікна",//6

"Повний екран",

"Правка",

"Копіювати",//9

"Вставити",

"Вирізати",

"Стиль тексту",//12

"Нормальний",

"Жирний",

"Курсив",//15

"Колір",

"Зберігти файл",

"Вибір кольору тексту",//18

"Відкрити файл",

};

Redactor() {

JFrame frame = new JFrame(localization[0]);

frame.setSize(width, height);

frame.setLocationRelativeTo(null);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JMenuBar menubar = new JMenuBar();

JMenu file = new JMenu(localization[1]);

JMenuItem open = new JMenuItem(localization[2]);

file.add(open);

open.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_O,

Event.CTRL\_MASK));

JMenuItem save = new JMenuItem(localization[3]);

file.add(save);

save.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_S,

Event.CTRL\_MASK));

file.addSeparator();

JMenuItem close = new JMenuItem(localization[4]);

file.add(close);

close.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_C,

Event.CTRL\_MASK));

JMenu view = new JMenu(localization[5]);

JCheckBoxMenuItem resizable = new JCheckBoxMenuItem(localization[6]);

view.add(resizable);

JCheckBoxMenuItem fullScreen = new JCheckBoxMenuItem(localization[7]);

view.add(fullScreen);

fullScreen.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_ENTER, Event.ALT\_MASK));

JMenu edit = new JMenu(localization[8]);

JMenuItem copy = new JMenuItem(localization[9]);

edit.add(copy);

JMenuItem paste = new JMenuItem(localization[10]);

edit.add(paste);

JMenuItem cut = new JMenuItem(localization[11]);

edit.add(cut);

cut.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_X, Event.CTRL\_MASK));

JMenu textStyle = new JMenu(localization[12]);

edit.add(textStyle);

JMenuItem normalStyle = new JMenuItem(localization[13]);

textStyle.add(normalStyle);

JMenuItem boldStyle = new JMenuItem(localization[14]);

textStyle.add(boldStyle);

JMenuItem italicStyle = new JMenuItem(localization[15]);

textStyle.add(italicStyle);

JToolBar toolBar = new JToolBar();

// Додайте JComboBox для вибору шрифту

String[] fontNames = GraphicsEnvironment.getLocalGraphicsEnvironment().getAvailableFontFamilyNames();

JComboBox<String> fontComboBox = new JComboBox<>(fontNames);

toolBar.add(fontComboBox);

// Додайте JTextField для введення розміру тексту

JTextField fontSizeField = new JTextField("12", 4);

toolBar.add(fontSizeField);

// Додайте кнопку для вибору кольору тексту

JButton colorButton = new JButton(localization[16]);

toolBar.add(colorButton);

menubar.add(file);

menubar.add(view);

menubar.add(edit);

JEditorPane ep = new JEditorPane();

JScrollPane scrollPane = new JScrollPane(ep);

frame.add(scrollPane);

frame.setJMenuBar(menubar);

frame.setVisible(true);

JFileChooser fc\_save = new JFileChooser(new File("d:\\"));

fc\_save.setDialogTitle(localization[17]);

FileTypeFilter fileFilter1 = new FileTypeFilter(".txt", "Текстовий файл");

FileTypeFilter fileFilter2 = new FileTypeFilter(".log", "Log-файл");

fc\_save.addChoosableFileFilter(fileFilter1);

fc\_save.addChoosableFileFilter(fileFilter2);

fontComboBox.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

String selectedFont = (String) fontComboBox.getSelectedItem();

ep.setFont(new Font(selectedFont, Font.PLAIN, Integer.parseInt(fontSizeField.getText())));

}

});

fontSizeField.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

String selectedFont = (String) fontComboBox.getSelectedItem();

ep.setFont(new Font(selectedFont, Font.PLAIN, Integer.parseInt(fontSizeField.getText())));

}

});

colorButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

Color textColor = JColorChooser.showDialog(frame, localization[18], Color.BLACK);

ep.setForeground(textColor);

}

});

save.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

// ȼɢɤɨɪɢɫɬɚɬɢ ɧɚɜɟɞɟɧɿ ɜɢɳɟ ɮɪɚɝɦɟɧɬɢ ɤɨɞɭ:

// 1. ɋɬɜɨɪɢɬɢ ɨɛ'єɤɬ ɤɥɚɫɭ JFileChooser ɿ ɜɤɚɡɚɬɢ ɡɚɝɨɥɨɜɨɤ

// 2. Ⱦɨɞɚɬɢ ɮɿɥьɬɪɢ

int result = fc\_save.showSaveDialog(null);

if (result == JFileChooser.APPROVE\_OPTION){

String content = ep.getText();

File fi = fc\_save.getSelectedFile();

try {

FileWriter fw = new FileWriter(fi.getPath() +

fc\_save.getFileFilter().getDescription());

fw.write(content);

fw.flush();

fw.close();

} catch (IOException ex) {

JOptionPane.showMessageDialog(null,

ex.getMessage());}}}});

open.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

// 1. ɋɬɜɨɪɢɬɢ ɨɛ'єɤɬ ɤɥɚɫɭ JFileChooser ɿ ɜɤɚɡɚɬɢ ɡɚɝɨɥɨɜɨɤ

// 2. Ⱦɨɞɚɬɢ ɮɿɥьɬɪɢ

JFileChooser fc\_open = new JFileChooser(new File("d:\\"));

fc\_open.setDialogTitle(localization[19]);

fc\_open.addChoosableFileFilter(fileFilter1);

fc\_open.addChoosableFileFilter(fileFilter2);

// Створіть модель для JSpinner зі значеннями від 1 до 100 та початковим значенням 12

SpinnerModel spinnerModel = new SpinnerNumberModel(12, 1, 100, 1);

JSpinner fontSizeSpinner = new JSpinner(spinnerModel);

// Додайте JSpinner до JToolBar

toolBar.add(fontSizeSpinner);

// Створіть фільтр для відкриття та збереження файлів з розширенням .txt

FileNameExtensionFilter txtFilter = new FileNameExtensionFilter("Текстові файли (\*.txt)", "txt");

fc\_save.setFileFilter(txtFilter);

JEditorPane ep = new JEditorPane();

JScrollPane scrollPane = new JScrollPane(ep);

frame.add(scrollPane);

// Встановіть стандартний фільтр для відкриття файлів

fc\_open.setFileFilter(txtFilter);

int result = fc\_open.showOpenDialog(null);

if (result == JFileChooser.APPROVE\_OPTION) {

try {File fi = fc\_open.getSelectedFile();

BufferedReader br = new BufferedReader(new

FileReader(fi.getPath()));

String line = "";

StringBuilder cont = new StringBuilder();

while ((line = br.readLine()) != null){

cont.append(line);

}

ep.setText(cont.toString());

if (br != null) br.close();

} catch (Exception ex) {

JOptionPane.showMessageDialog(null, ex.getMessage());

}}}});

close.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

System.exit(0);

}});

resizable.addItemListener(new ItemListener(){

public void itemStateChanged(ItemEvent e){

if (e.getStateChange() == ItemEvent.SELECTED) {

frame.setResizable(false);

fullScreen.setVisible(false);

}else {

frame.setResizable(true);

fullScreen.setVisible(true);

}

}});

fullScreen.addItemListener(new ItemListener(){

public void itemStateChanged(ItemEvent e){

if (e.getStateChange() == ItemEvent.SELECTED)

frame.setExtendedState(JFrame.MAXIMIZED\_BOTH);

else frame.setExtendedState(JFrame.NORMAL);

}});

copy.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

ep.copy();}});

paste.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

ep.paste();

}});

JPopupMenu jp\_menu = new JPopupMenu();

JMenuItem copyContext = new JMenuItem(localization[9]);

jp\_menu.add(copyContext);

JMenuItem pasteContext = new JMenuItem(localization[10]);

jp\_menu.add(pasteContext);

JMenuItem cutContext = new JMenuItem(localization[11]);

jp\_menu.add(cutContext);

ep.setComponentPopupMenu(jp\_menu);

cutContext.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

ep.cut();

}

});

ep.addMouseListener(new MouseAdapter() {

@Override

public void mouseReleased(MouseEvent evt) {

if (evt.isPopupTrigger()) {

jp\_menu.show(evt.getComponent(), evt.getX(), evt.getY());

}}});

frame.setVisible(true);

copyContext.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

ep.copy();

}

});

pasteContext.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

ep.paste();

}

});

cut.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

ep.cut();

}

});

normalStyle.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

// Застосовуйте стиль "Нормальний" до тексту

ep.setFont(ep.getFont().deriveFont(Font.PLAIN));

}

});

boldStyle.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

// Застосовуйте стиль "Жирний" до тексту

ep.setFont(ep.getFont().deriveFont(Font.BOLD));

}

});

italicStyle.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

// Застосовуйте стиль "Курсив" до тексту

ep.setFont(ep.getFont().deriveFont(Font.ITALIC));

}

});

frame.add(toolBar, BorderLayout.NORTH);

}

}

Додаток Ї

*Лістинг файлу «Table.java»*

import javax.swing.\*;

import javax.swing.JFormattedTextField;

import javax.swing.text.NumberFormatter;

import javax.swing.table.TableRowSorter;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.MouseAdapter;

import java.awt.event.MouseEvent;

import java.text.NumberFormat;

public class Table extends JMenuItem{

public static int width = 800;

public static int height = 600;

public static Font font = new Font("Arial", Font.BOLD, 16);

public static int columnNumber = 15;

public static int[] columnWidth = {

50, 200, 120, 120, 200

};

public static String[] localizationUA = {

"Додавання даних",

"Додати",

"ПІБ",

"№ студ.", //3

"Контактні дані",

"Група",

"Фільтр:", //6

"Фільтрувати",

"Видалити рядок",

"Редагувати", //9

"Закінчити редагування",

};

Table() {

JFrame frame = new JFrame();

frame.setSize(width, height);

frame.setLocationRelativeTo(null);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setLayout(new GridBagLayout());

GridBagConstraints c = new GridBagConstraints();

TableModel tm = new TableModel();

JTable t1 = new JTable(tm);

DefaultCellEditor cellEditor = new DefaultCellEditor(new JTextField());

t1.setDefaultEditor(Object.class, cellEditor);

t1.setAutoResizeMode(JTable.AUTO\_RESIZE\_OFF);

for(int i = 0; i < columnWidth.length; i++)

t1.getColumnModel().getColumn(i).setPreferredWidth(columnWidth[i]);

JScrollPane scrollPane = new JScrollPane(t1);

scrollPane.setPreferredSize(new Dimension(t1.getPreferredSize().width, 400));

c.gridx = 0;

c.gridy = 0;

c.gridwidth = 1;

c.gridheight = 2;

c.fill = GridBagConstraints.BOTH;

frame.add(scrollPane, c);

JPanel panel1 = new JPanel();

JLabel panel\_name = new JLabel(localizationUA[0]);

panel\_name.setFont(font);

panel1.setLayout(new GridBagLayout());

JButton add\_data = new JButton(localizationUA[1]);

JLabel fio\_lab = new JLabel(localizationUA[2]);

final JTextField fio\_tf = new JTextField(columnNumber);

JLabel id\_lab = new JLabel(localizationUA[3]);

NumberFormatter formatter = new NumberFormatter(NumberFormat.getIntegerInstance());

formatter.setValueClass(Integer.class); // Укажите класс значений (в данном случае Integer)

formatter.setMinimum(0); // Минимальное значение

formatter.setMaximum(Integer.MAX\_VALUE); // Максимальное значение

JFormattedTextField id\_tf = new JFormattedTextField(formatter);

id\_tf.setColumns(columnNumber);

JLabel number\_lab = new JLabel(localizationUA[4]);

final JTextField number\_tf = new JTextField(columnNumber);

JLabel group\_lab = new JLabel(localizationUA[5]);

final JTextField group\_tf = new JTextField(columnNumber);

// Додайте фільтраційну панель

JPanel filterPanel = new JPanel();

JLabel filterLabel = new JLabel(localizationUA[6]);

JComboBox<String> filterColumn = new JComboBox<>(tm.getColumnNames());

JTextField filterText = new JTextField(columnNumber);

JButton filterButton = new JButton(localizationUA[7]);

filterPanel.add(filterLabel);

filterPanel.add(filterColumn);

filterPanel.add(filterText);

filterPanel.add(filterButton);

c.gridx = 0;

c.gridy = 2;

c.gridwidth = 1;

frame.add(filterPanel, c);

// Створіть фільтрувальник для таблиці

DefaultRowSorter<TableModel, Integer> sorter = new TableRowSorter<>(tm);

t1.setRowSorter(sorter);

filterButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

String selectedColumn = filterColumn.getSelectedItem().toString();

String filterValue = filterText.getText();

sorter.setRowFilter(RowFilter.regexFilter(filterValue, tm.getColumnIndexByName(selectedColumn)));

}

});

c.gridwidth = 1;

c.gridheight = 1;

c.fill = GridBagConstraints.HORIZONTAL;

c.gridwidth = 2;

c.gridx = 0;

c.gridy = 0;

panel1.add(panel\_name, c);

c.gridwidth = 1;

c.gridx = 0;

c.gridy = 1;

c.anchor = GridBagConstraints.NORTH;

panel1.add(fio\_lab, c);

c.gridx = 1;

c.gridy = 1;

panel1.add(fio\_tf, c);

c.gridx = 0;

c.gridy = 2;

panel1.add(id\_lab, c);

c.gridx = 1;

c.gridy = 2;

panel1.add(id\_tf, c);

c.gridx = 0;

c.gridy = 3;

panel1.add(group\_lab, c);

c.gridx = 1;

c.gridy = 3;

panel1.add(group\_tf, c);

c.gridx = 0;

c.gridy = 4;

panel1.add(number\_lab, c);

c.gridx = 1;

c.gridy = 4;

panel1.add(number\_tf, c);

c.gridx = 0;

c.gridy = 5;

c.gridwidth = 2;

panel1.add(add\_data, c);

c.gridx = 1;

c.gridy = 0;

c.gridwidth = 1;

frame.add(panel1, c);

JButton select\_remove = new JButton(localizationUA[8]);

c.gridx = 1;

c.gridy = 1;

c.gridwidth = 1;

c.gridheight = 1;

frame.add(select\_remove, c);

add\_data.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

String[] st = {Integer.toString(tm.getRowCount() + 1),

fio\_tf.getText(), id\_tf.getText(), group\_tf.getText(), number\_tf.getText()};

tm.addData(st);

fio\_tf.setText("");

id\_tf.setText("");

group\_tf.setText("");

number\_tf.setText("");

t1.revalidate();

t1.repaint();

}

});

final boolean[] editMode = {false}; // Declare as an array to make it effectively final

JButton editButton = new JButton(localizationUA[9]);

c.gridx = 1;

c.gridy = 2;

frame.add(editButton, c);

editButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

editMode[0] = !editMode[0];

if (editMode[0]) {

t1.setEnabled(true);

editButton.setText(localizationUA[10]);

} else {

t1.setEnabled(false);

editButton.setText(localizationUA[9]);

}

}

});

// Rest of the code remains the same

// Добавьте слушатель для редактирования данных в таблице

t1.addMouseListener(new MouseAdapter() {

@Override

public void mouseClicked(MouseEvent e) {

if (editMode[0] && e.getClickCount() == 2) {

int selectedRow = t1.getSelectedRow();

int selectedColumn = t1.getSelectedColumn();

if (selectedRow != -1 && selectedColumn != -1) {

t1.editCellAt(selectedRow, selectedColumn);

t1.getEditorComponent().requestFocus();

}

}

}

});

select\_remove.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

int selectedRow = t1.getSelectedRow();

if (selectedRow != -1) {

int modelRow = t1.convertRowIndexToModel(selectedRow);

tm.removeRow(modelRow);

tm.fireTableDataChanged(); // Обновить таблицу

}

}

});

t1.setRowSelectionAllowed(false);

t1.setColumnSelectionAllowed(false);

t1.setCellSelectionEnabled(true);

frame.pack();

frame.setVisible(true);

}

}

Додаток Ж

*Лістинг файлу «TableModel.java»*

import javax.swing.table.AbstractTableModel;

import java.util.ArrayList;

import javax.swing.RowFilter;

public class TableModel extends AbstractTableModel {

protected ArrayList<String[]> data;

protected int columnCount = 5;

public TableModel() {

data = new ArrayList<String[]>();

}

@Override

public int getRowCount() {

return data.size();

}

@Override

public int getColumnCount() {

return columnCount;

}

@Override

public Object getValueAt(int rowIndex, int columnIndex) {

String[] row = data.get(rowIndex);

return row[columnIndex];

}

// Добавьте следующий метод для разрешения редактирования ячейки

@Override

public boolean isCellEditable(int rowIndex, int columnIndex) {

// Возвращайте true только для тех столбцов, которые вы хотите разрешить редактировать.

return columnIndex != 0; // Например, не разрешайте редактировать столбец "№"

}

// Добавьте следующий метод для сохранения изменений в ячейке

@Override

public void setValueAt(Object value, int rowIndex, int columnIndex) {

String[] row = data.get(rowIndex);

row[columnIndex] = value.toString();

fireTableCellUpdated(rowIndex, columnIndex);

}

public void removeRow(int rowIndex) {

if (rowIndex >= 0 && rowIndex < data.size()) {

data.remove(rowIndex);

fireTableRowsDeleted(rowIndex, rowIndex);

changeNumbers(rowIndex);

}

}

public void addData(String[] row) {

data.add(row);

fireTableDataChanged();

}

public String getColumnName(int columnIndex) {

switch (columnIndex) {

case 0:

return "№";

case 1:

return Table.localizationUA[2];

case 2:

return Table.localizationUA[3];

case 3:

return Table.localizationUA[5];

case 4:

return Table.localizationUA[4];

}

return "";

}

public void changeNumbers(int rowDeleteIndex) {

for (int i = rowDeleteIndex; i < data.size(); i++) {

String currentId = data.get(i)[0];

data.get(i)[0] = Integer.toString(i + 1);

}

}

public int getColumnIndexByName(String columnName) {

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

if (getColumnName(i).equals(columnName)) {

return i;

}

}

return -1;

}

public String[] getColumnNames() {

String[] columnNames = new String[getColumnCount()];

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

columnNames[i] = getColumnName(i);

}

return columnNames;

}

}