**ЛАБОРАТОРНА РОБОТА № 4**

з дисципліни

«Розробка програмного забезпечення на Java»

Шейко Р. О.

Групи КІ-21-2

**Завдання на лабораторну роботу**

1. У меню Правка додати і реалізувати пункт Вирізати. Додати створений елемент в контекстне меню (JPopupMenu).

2. Додати в меню Правка підменю Стиль тексту з наступними елементами: Нормальний, Жирний і Курсив. Описати обробники для цих пунктів. Дані операції застосовуються до всього вмістуJEditorPane.

3. Нижче меню додати компонент JToolBar, який буде відповідати за форматування тексту в компоненті JEditorPane. Помістити на JToolBar наступні компоненти: JComboBox, JTextField і JButton. Використовувати JComboBox для вибору одного з декількох шрифтів. Компонент JTextField використовувати для введення розміру тексту, зміна якого повинна відбутися після натискання клавіші Enter. Кнопка JButton повинна викликати компо-нент JColorChooser для вибору кольору тексту. Дані операції застосовуються до всього вмісту JEditorPane.

4. Виконати оптимізацію проекту. Застосувати компонент JSpinner, стандартний фільтр FileNameExtensionFilter.

1. У меню Правка додати і реалізувати пункт Вирізати. Додати створений елемент в контекстне меню (JPopupMenu). Реалізований інтерфейс програми показано на рис 1в і 1г.

|  |  |
| --- | --- |
|  |  |
| Рис 1a. Інтерфейс програми | Рис 1б. Інтерфейс програми |

|  |  |
| --- | --- |
|  |  |
| Рис 1в. Інтерфейс програми | Рис 1г. Інтерфейс програми |

1. Додати в меню Правка підменю Стиль тексту з наступними елементами: Нормальний, Жирний і Курсив. Описати обробники для цих пунктів. Дані операції застосовуються до всього вмісту JEditorPane.

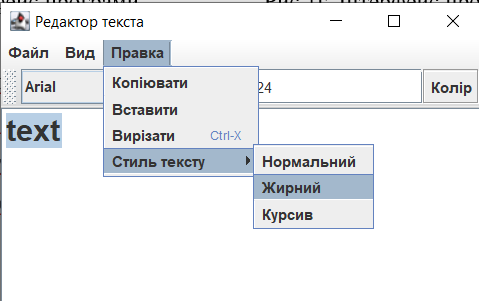


Рис 2. Ілюстрація роботи жирного

1. Нижче було додано компонент JToolBar, який буде відповідати за форматування тексту в компоненті JEditorPane. Помістити на JToolBar наступні компоненти: JComboBox, JTextField і JButton. Використовувати JComboBox для вибору одного з декількох шрифтів. Компонент JTextField використовувати для введення розміру тексту, зміна якого повинна відбутися після натискання клавіші Enter. Кнопка JButton повинна викликати компо-нент JColorChooser для вибору кольору тексту. Дані операції застосовуються до всього вмісту JEditorPane.

|  |  |
| --- | --- |
|  |  |
| Рис 3а. Ілюстрація роботи JComboBox | Рис 3б. Ілюстрація роботи JColorChooser |

1. Виконати оптимізацію проекту. Застосувати компонент JSpinner, стандартний фільтр FileNameExtensionFilter.

Використання JSpinner:

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

JSpinner fontSizeSpinner = new JSpinner(spinnerModel);

toolBar.add(fontSizeSpinner);

Використання FileNameExtensionFilter:

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

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

public static int width = 400;

public static int height = 250;

Були додані зміни до скролл панелі.

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 А і Б.

**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.\*;

import java.io.\*;

import javax.swing.JSpinner;

import javax.swing.SpinnerModel;

import javax.swing.SpinnerNumberModel;

import javax.swing.filechooser.FileNameExtensionFilter;

public class Main {

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",

};

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);

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 Б*

*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;

}

}