**Отчёт по практической работе «Лабораторная работа 16»**

Выполнил студент

Курс: ПРИ-311

ФИО: Пикалов Е.А.

**Задание:**

Создание текстового редактора

**Программный код:**

**App.java**

public class App {

    public static void main(String[] args) throws Exception {

        TextEditor texteditor = new TextEditor();

        texteditor.setVisible(true);

    }

}

**TextEditor:**

import java.awt.\*;

import javax.swing.\*;

import java.awt.event.\*;

import java.io.\*;

import javax.swing.filechooser.FileNameExtensionFilter;

*// import java.io.File;*

*// import java.io.FileInputStream;*

*// import java.io.FileNotFoundException;*

*// import java.io.IOException;*

import java.util.List;

import java.util.Scanner;

import org.apache.poi.xwpf.usermodel.XWPFDocument;

import org.apache.poi.xwpf.usermodel.XWPFParagraph;

import org.apache.poi.xwpf.usermodel.XWPFRun;

public class TextEditor extends JFrame{

    private Container container;

    private JMenuBar menu;

    private JMenu createFile;

    private JMenuItem createTXT;

    private JMenuItem createWord;

    private JMenuItem createExcel;

    private JMenuItem createCSV;

    private JMenuItem editFile;

    private JMenu editWord;

    private JMenu editTXT;

    private JMenu editExcel;

    private JMenu editCSV;

    private JMenuItem writeTXTFile;

    private JMenuItem addTXTFile;

    private JMenuItem readTXTFile;

    private JMenuItem writeWordFile;

    private JMenuItem addWordFile;

    private JMenuItem readWordFile;

    private JMenuItem writeExcelFile;

    private JMenuItem addExcelFile;

    private JMenuItem readExcelFile;

    private JMenuItem writeCSVFile;

    private JMenuItem addCSVFile;

    private JMenuItem readCSVFile;

    private JMenu viewReports;

    private JMenu admin;

    private JMenuItem exit;

    private JTextArea editor;

    private JScrollPane scroll;

    private JFileChooser dialogRead;

    private JFileChooser dialogWrite;

    public TextEditor(){

        super("Text Editor");

        super.setBounds(200, 200, 700, 400);

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

        try {

            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");

        } catch (ClassNotFoundException | InstantiationException | IllegalAccessException | UnsupportedLookAndFeelException e) {

            e.printStackTrace();

        }

        container = super.getContentPane();

        container.setLayout(new BorderLayout());

        menu = new JMenuBar();

        createFile = new JMenu("Создание файла");

        createTXT = new JMenuItem("Формат TXT");

        createWord = new JMenuItem("Формат Word");

        createExcel = new JMenuItem("Формат Excel");

        createCSV = new JMenuItem("Формат CSV");

        editFile = new JMenu("Работа с файлами различных форматов");

        editTXT = new JMenu("Формат TXT");

        editWord = new JMenu("Формат Word");

        editExcel = new JMenu("Формат Excel");

        editCSV = new JMenu("Формат CSV");

        writeTXTFile = new JMenuItem("Записать в файл");

        addTXTFile = new JMenuItem("Дописать в файл");

        readTXTFile = new JMenuItem("Чтение их файла");

        writeWordFile = new JMenuItem("Записать в файл");

        addWordFile = new JMenuItem("Дописать в файл");

        readWordFile = new JMenuItem("Чтение их файла");

        writeExcelFile = new JMenuItem("Записать в файл");

        addExcelFile = new JMenuItem("Дописать в файл");

        readExcelFile = new JMenuItem("Чтение их файла");

        writeCSVFile = new JMenuItem("Записать в файл");

        addCSVFile = new JMenuItem("Дописать в файл");

        readCSVFile = new JMenuItem("Чтение их файла");

        viewReports = new JMenu("Просмотр отчетов");

        admin = new JMenu("Администрирование");

        exit = new JMenuItem("Выход");

        dialogRead = new JFileChooser();

        dialogWrite = new JFileChooser();

        editor = new JTextArea();

        editor.setFont(editor.getFont().deriveFont(12f));

        scroll = new JScrollPane();

        editor.setLineWrap(true);

        editor.setWrapStyleWord(true);

*// editor.setEditable(false);*

        scroll.setViewportView(editor);

        scroll.getVerticalScrollBar().setPreferredSize(new Dimension(0, 0));

        readTXTFile.addActionListener(new OpenTxtAction());

        readWordFile.addActionListener(new OpenWordAction());

        readExcelFile.addActionListener(new OpenExcelAction());

        readCSVFile.addActionListener(new OpenCSVAction());

        createTXT.addActionListener(new CreateTxtAction());

        createWord.addActionListener(new CreateWordAction());

        createExcel.addActionListener(new CreateExcelAction());

        createCSV.addActionListener(new CreateCSVAction());

        writeTXTFile.addActionListener(new WriteTxtAction());

        writeWordFile.addActionListener(new WriteWordAction());

        writeExcelFile.addActionListener(new WriteExcelAction());

        writeCSVFile.addActionListener(new WriteCSVAction());

        addTXTFile.addActionListener(new AddTxtAction());

        addWordFile.addActionListener(new AddWordAction());

        addExcelFile.addActionListener(new AddExcelAction());

        addCSVFile.addActionListener(new AddCSVAction());

        viewReports.addActionListener(new ViewReportsAction());

        admin.addActionListener(new AdminAction());

        exit.addActionListener(new ExitAction());

        editTXT.add(writeTXTFile);

        editTXT.add(addTXTFile);

        editTXT.add(readTXTFile);

        editWord.add(writeWordFile);

        editWord.add(addWordFile);

        editWord.add(readWordFile);

        editExcel.add(writeExcelFile);

        editExcel.add(addExcelFile);

        editExcel.add(readExcelFile);

        editCSV.add(writeCSVFile);

        editCSV.add(addCSVFile);

        editCSV.add(readCSVFile);

        createFile.add(createTXT);

        createFile.add(createWord);

        createFile.add(createExcel);

        createFile.add(createCSV);

        editFile.add(editTXT);

        editFile.add(editWord);

        editFile.add(editExcel);

        editFile.add(editCSV);

        menu.add(createFile);

        menu.add(editFile);

        menu.add(viewReports);

        menu.add(admin);

        menu.add(exit);

        super.setDefaultLookAndFeelDecorated(true);

        super.setJMenuBar(menu);

        container.add(scroll);

    }

    private class ExitAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            System.exit(0);

        }

    }

    private class OpenTxtAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e){

            FileNameExtensionFilter filter = new FileNameExtensionFilter("Text File(.txt)", "txt");

            dialogRead.resetChoosableFileFilters();

            dialogRead.setFileFilter(filter);

            dialogRead.showOpenDialog(container);

            String text = "";

            String wrap = "";

            try {

                File file = new File(dialogRead.getSelectedFile().getPath());

                Scanner sc = new Scanner(file);

                while (sc.hasNextLine()){

                    text += wrap + sc.nextLine();

                    wrap = "\n";

                }

                sc.close();

            } catch (FileNotFoundException | NullPointerException e1) {

                e1.printStackTrace();

            }

            editor.setText(text);

        }

    }

    private class OpenWordAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e){

            FileNameExtensionFilter filter = new FileNameExtensionFilter("Microsoft Word(.doc, .docx)", "docx", "doc");

            dialogRead.resetChoosableFileFilters();

            dialogRead.setFileFilter(filter);

            dialogRead.showOpenDialog(container);

            String text = "";

            String wrap = "";

            try {

                File file = new File(dialogRead.getSelectedFile().getPath());

                XWPFDocument document = new XWPFDocument(new FileInputStream(file));

                List<XWPFParagraph> paragraphs = document.getParagraphs();

                document.close();

                for (XWPFParagraph para : paragraphs){

                    text += wrap + para.getText();

                    wrap = "\n";

                }

            } catch (NullPointerException | IOException e1) {

                e1.printStackTrace();

            }

            editor.setText(text);

        }

    }

    private class OpenExcelAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            editor.setText("Данная функция находиться в разработке");

        }

    }

    private class OpenCSVAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            editor.setText("Данная функция находиться в разработке");

        }

    }

    private class CreateTxtAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            FileNameExtensionFilter filter = new FileNameExtensionFilter("Text File(.txt)", "txt");

            dialogWrite.resetChoosableFileFilters();

            dialogWrite.setFileFilter(filter);

            dialogWrite.setSelectedFile(new File("FileName.txt"));

            dialogWrite.showSaveDialog(container);

            File file = new File(dialogWrite.getSelectedFile().getPath());

            try {

                if (!dialogWrite.getSelectedFile().getPath().endsWith(".txt")){

                    file = new File(dialogWrite.getSelectedFile().getPath() + ".txt");

                }

                file.createNewFile();

            } catch (IOException e1) {

                e1.printStackTrace();

            }

        }

    }

    private class CreateWordAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            FileNameExtensionFilter filter = new FileNameExtensionFilter("Microsoft Word(.doc, .docx)", "docx", "doc");

            dialogWrite.resetChoosableFileFilters();

            dialogWrite.setFileFilter(filter);

            dialogWrite.setSelectedFile(new File("FileName.docx"));

            dialogWrite.showSaveDialog(container);

            File file = new File(dialogWrite.getSelectedFile().getPath());

            try {

                if (!dialogWrite.getSelectedFile().getPath().endsWith(".doc") && !dialogWrite.getSelectedFile().getPath().endsWith(".docx")){

                    file = new File(dialogWrite.getSelectedFile().getPath() + ".docx");

                }

                file.createNewFile();

            } catch (IOException e1) {

                e1.printStackTrace();

            }

        }

    }

    private class CreateExcelAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            FileNameExtensionFilter filter = new FileNameExtensionFilter("Microsoft Excel(.xlsx)", "xlsx");

            dialogWrite.resetChoosableFileFilters();

            dialogWrite.setFileFilter(filter);

            dialogWrite.setSelectedFile(new File("FileName.xlsx"));

            dialogWrite.showSaveDialog(container);

            File file = new File(dialogWrite.getSelectedFile().getPath());

            try {

                if (!dialogWrite.getSelectedFile().getPath().endsWith(".xlsx")){

                    file = new File(dialogWrite.getSelectedFile().getPath() + ".xlsx");

                }

                file.createNewFile();

            } catch (IOException e1) {

                e1.printStackTrace();

            }

        }

    }

    private class CreateCSVAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            FileNameExtensionFilter filter = new FileNameExtensionFilter("Text File(.csv)", "csv");

            dialogWrite.resetChoosableFileFilters();

            dialogWrite.setFileFilter(filter);

            dialogWrite.setSelectedFile(new File("FileName.csv"));

            dialogWrite.showSaveDialog(container);

            File file = new File(dialogWrite.getSelectedFile().getPath());

            try {

                if (!dialogWrite.getSelectedFile().getPath().endsWith(".csv")){

                    file = new File(dialogWrite.getSelectedFile().getPath() + ".csv");

                }

                file.createNewFile();

            } catch (IOException e1) {

                e1.printStackTrace();

            }

        }

    }

    private class WriteTxtAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            FileNameExtensionFilter filter = new FileNameExtensionFilter("Text File(.txt)", "txt");

            dialogRead.resetChoosableFileFilters();

            dialogRead.setFileFilter(filter);

            dialogRead.showOpenDialog(container);

            try {

                BufferedWriter fileout = new BufferedWriter(new FileWriter(dialogRead.getSelectedFile().getPath()));

                editor.write(fileout);

            } catch (IOException e1) {

                e1.printStackTrace();

            }

        }

    }

    private class WriteWordAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            FileNameExtensionFilter filter = new FileNameExtensionFilter("Microsoft Word(.doc, .docx)", "docx", "doc");

            dialogRead.resetChoosableFileFilters();

            dialogRead.setFileFilter(filter);

            dialogRead.showOpenDialog(container);

            try {

                File file = new File(dialogRead.getSelectedFile().getPath());

                XWPFDocument document = new XWPFDocument();

                String[] text = editor.getText().split("\n");

                for (String string : text) {

                    XWPFParagraph para = document.createParagraph();

                    XWPFRun run = para.createRun();

                    run.setText(string);

                }

                FileOutputStream out = new FileOutputStream(file);

                document.write(out);

                out.close();

                document.close();

            } catch (IOException e1) {

                e1.printStackTrace();

            }

        }

    }

    private class WriteExcelAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            editor.setText("Данная функция находиться в разработке");

        }

    }

    private class WriteCSVAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            editor.setText("Данная функция находиться в разработке");

        }

    }

    private class AddTxtAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            FileNameExtensionFilter filter = new FileNameExtensionFilter("Text File(.txt)", "txt");

            dialogRead.resetChoosableFileFilters();

            dialogRead.setFileFilter(filter);

            dialogRead.showOpenDialog(container);

            String text = "";

            String wrap = "";

            try {

                File file = new File(dialogRead.getSelectedFile().getPath());

                Scanner sc = new Scanner(file);

                while (sc.hasNextLine()){

                    text += wrap + sc.nextLine();

                    wrap = "\n";

                }

                sc.close();

                text += "\n" + editor.getText();

                BufferedWriter fileout = new BufferedWriter(new FileWriter(file));

                fileout.write(text);

                fileout.close();

            } catch (NullPointerException | IOException e1) {

                e1.printStackTrace();

            }

        }

    }

    private class AddWordAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            FileNameExtensionFilter filter = new FileNameExtensionFilter("Microsoft Word(.doc, .docx)", "docx", "doc");

            dialogRead.resetChoosableFileFilters();

            dialogRead.setFileFilter(filter);

            dialogRead.showOpenDialog(container);

            try {

                File file = new File(dialogRead.getSelectedFile().getPath());

                String[] text = editor.getText().split("\n");

                XWPFDocument document = new XWPFDocument(new FileInputStream(file));

                for (String string : text) {

                    XWPFParagraph para = document.createParagraph();

                    XWPFRun run = para.createRun();

                    run.setText(string);

                }

                FileOutputStream out = new FileOutputStream(file);

                document.write(out);

                out.close();

                document.close();

            } catch (NullPointerException | IOException e1) {

                e1.printStackTrace();

            }

        }

    }

    private class AddExcelAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            editor.setText("Данная функция находиться в разработке");

        }

    }

    private class AddCSVAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            editor.setText("Данная функция находиться в разработке");

        }

    }

    private class ViewReportsAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            editor.setText("Данная функция находиться в разработке");

        }

    }

    private class AdminAction implements ActionListener{

        @Override

        public void actionPerformed(ActionEvent e) {

            editor.setText("Данная функция находиться в разработке");

        }

    }

}

**Результат:**

