A
PROJECT REPORT ON

"JAVA NOTEPAD APPLICATION"

By

Sr. No.	NAME	ROLL NO.
1	Gayatri Anil Taware	32481
2	Sakshi Annasaheb Thorat	32482

GUIDE
MR. NILESH SHIRUDE

DEPARTMENT OF
ELECTRONICS AND TELECOMMUNICATION ENGINEERING
PUNE INSTITUTE OF COMPUTER TECHNOLOGY
PUNE – 43

A.Y. 2023-24

INDEX

1. PROBLEM STATEMENT:

To Build a Simple Java NotePad Application Using Java.

2. OBJECTIVE :

Objective: The objective is to build a simple Java notepad application that allows users to create, edit, save, and open text files.

Addressing the Problem:

Basic Text Editing: Implement functionality for basic text editing such as typing, deleting, and selecting text.

File Operations: Incorporate features for file operations like opening, saving, and creating new text files. This allows users to interact with the application effectively.

User Interface: Design a user-friendly interface to provide a seamless experience for users. This includes components such as menus, toolbars, and text areas.

Error Handling: Implement proper error handling mechanisms to notify users of any issues that may arise during file operations or application usage.

Keyboard Shortcuts: Integrate keyboard shortcuts for common operations to enhance user productivity and ease of use.

Persistence: Ensure that changes made by the user are persisted, so that they are not lost when closing or reopening the application.

By achieving these objectives, the Java notepad application effectively addresses the problem of providing users with a simple yet functional tool for creating and editing text files. It offers a familiar interface similar to standard text editors while incorporating essential features necessary for text editing and file management.

3. INTRODUCTION:

This lightweight yet powerful tool is designed to fulfill all your basic text editing needs with ease and simplicity. Whether you're a student jotting down notes, a programmer writing code snippets, or just someone who needs to keep track of thoughts and ideas, our Java Notepad has got you covered.

With a clean and intuitive user interface, you can dive right into creating, editing, saving, and opening text files effortlessly. The application offers essential features such as basic text editing functionalities, file operations including opening, saving, and creating new files, as well as keyboard shortcuts for quick navigation and operation.

Say goodbye to complicated text editors and welcome the simplicity and efficiency of our Java Notepad. Whether you're a novice or an experienced user, you'll find this application to be your go-to tool for all your text editing needs.

3.1 Background/context

The concept of text editors dates back to the early days of computing when the need arose for a digital platform to create, edit, and manage textual information. The journey of text editing applications has been intertwined with the evolution of computing technology itself.

Text editors have a rich history, starting from simple command-line tools to sophisticated graphical user interface (GUI) applications. They have played a crucial role in various domains such as software development, documentation, academic research, and personal note-taking.

The development of text editors has seen significant milestones, including the introduction of features like syntax highlighting, spell checking, version control integration, and collaborative editing. Alongside technological advancements, user experience and accessibility have also been key focus areas, leading to the creation of intuitive interfaces and streamlined workflows.

The popularity of text editors continues to grow, with users ranging from beginners seeking a simple tool for note-taking to professionals relying on advanced features for complex tasks. In today's digital age, where information is abundant and communication is essential, text editors remain indispensable tools for productivity and creativity.

3.2 Relevance

While the primary focus of the project lies in software development using Java, its relevance to the field of Electronics and Communication stems from the underlying principles of computer science and technology.

Overall, the Java Notepad application serves as a practical example of software development principles that are relevant across various domains, including electronics and communication. Through this project, learners can gain insights into essential concepts such as GUI development, file handling, event-driven programming, and potentially explore connections

to broader electronic systems and communication technologies.

3.3 Project Details

The Simple Java Notepad Application is a straightforward yet functional tool developed in Java to fulfill basic text editing needs. Its graphical user interface (GUI) offers a minimalistic layout with menus, a text area for input and editing, and intuitive toolbar options. Users can seamlessly create new files, open existing ones, and save their work using the integrated file operations. The application also supports standard text editing functionalities like cut, copy, paste, and undo, enhancing user productivity and ease of use.

Project leverages Java's Swing library to construct the UI components and manage user interactions. Event handling mechanisms ensure smooth execution of actions triggered by user inputs, such as opening and saving files.

File input/output streams facilitate reading from and writing to text files, enabling seamless file management within the application. Overall, the Simple Java Notepad Application provides a solid foundation for a lightweight text editing tool, demonstrating essential Java programming concepts in GUI development, file handling, and event-driven programming.

3.4 Scope:

1. Basic Text Editing Features
2. File Operations
3. User Interface Design
4. Error Handling and Validation
5. Keyboard Shortcuts
6. Persistence and Autosave

4.Flow Chart and SOURCE CODE:

myApp.java

```
package notepad;
public class MyApp {

   public static void main(String [] args){

   new Notepad();
}
}
```

Notepad.java

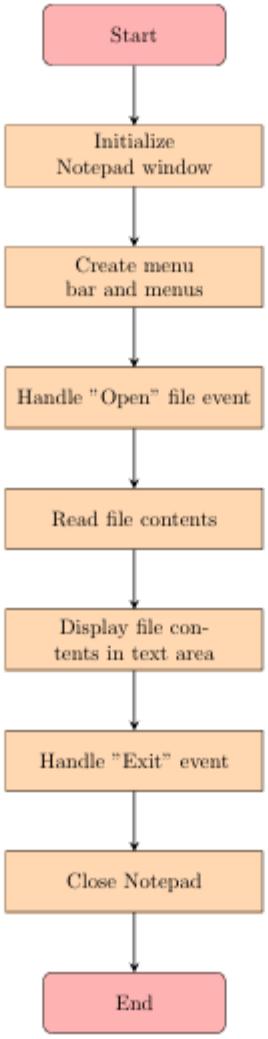
```
package notepad;
import java.io.*;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;


public class Notepad extends JFrame implements ActionListener, WindowListener{
/**
         *
         */
        private static final long serialVersionUID = 1L;
JTextArea jta=new JTextArea();
        File fnameContainer;

        public Notepad(){


                Font fnt=new Font("Arial",Font.PLAIN,15);
                Container con=getContentPane();

                JMenuBar jmb=new JMenuBar();
                JMenu jmfile = new JMenu("File");
                JMenu jmedit = new JMenu("Edit");
                JMenu jmhelp = new JMenu("Help");

                con.setLayout(new BorderLayout());
                //trying to add scrollbar
                JScrollPane sbrText = new JScrollPane(jta);
                sbrText.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_ALWAYS);
                sbrText.setVisible(true);
```

```
        jta.setFont(fnt);
        jta.setLineWrap(true);
        jta.setWrapStyleWord(true);

        con.add(sbrText);

        createMenuItem(jmfile,"New");
        createMenuItem(jmfile,"Open");
        createMenuItem(jmfile,"Save");
        jmfile.addSeparator();
        createMenuItem(jmfile,"Exit");

        createMenuItem(jmedit,"Cut");
        createMenuItem(jmedit,"Copy");
        createMenuItem(jmedit,"Paste");

        createMenuItem(jmhelp,"About Notepad");

        jmb.add(jmfile);
        jmb.add(jmedit);
        jmb.add(jmhelp);

        setJMenuBar(jmb);
        setIconImage(Toolkit.getDefaultToolkit().getImage("notepad.gif"));
        addWindowListener(this);
        setSize(500,500);
        setTitle("Untitled.txt - Notepad");

        setVisible(true);

}

public void createMenuItem(JMenu jm,String txt){
        JMenuItem jmi=new JMenuItem(txt);
        jmi.addActionListener(this);
        jm.add(jmi);
}

public void actionPerformed(ActionEvent e){
        JFileChooser jfc=new JFileChooser();

        if(e.getActionCommand().equals("New")){
                //new
```

```
            this.setTitle("Untitled.txt - Notepad");
            jta.setText("");
            fnameContainer=null;
        }else if(e.getActionCommand().equals("Open")){
            //open
            int ret=jfc.showDialog(null,"Open");

            if(ret == JFileChooser.APPROVE_OPTION)
            {
                try{
                    File fyl=jfc.getSelectedFile();
                    OpenFile(fyl.getAbsolutePath());
                    this.setTitle(fyl.getName()+ " - Notepad");
                    fnameContainer=fyl;
                }catch(IOException ers){}
            }

        }else if(e.getActionCommand().equals("Save")){
            //save
            if(fnameContainer != null){
                jfc.setCurrentDirectory(fnameContainer);
                jfc.setSelectedFile(fnameContainer);
            }
            else {
                //jfc.setCurrentDirectory(new File("."));
                jfc.setSelectedFile(new File("Untitled.txt"));
            }

            int ret=jfc.showSaveDialog(null);

            if(ret == JFileChooser.APPROVE_OPTION){
                try{

                    File fyl=jfc.getSelectedFile();
                    SaveFile(fyl.getAbsolutePath());
                    this.setTitle(fyl.getName()+ " - Notepad");
                    fnameContainer=fyl;

                }catch(Exception ers2){}
            }

        }else if(e.getActionCommand().equals("Exit")){
            //exit
            Exiting();
```

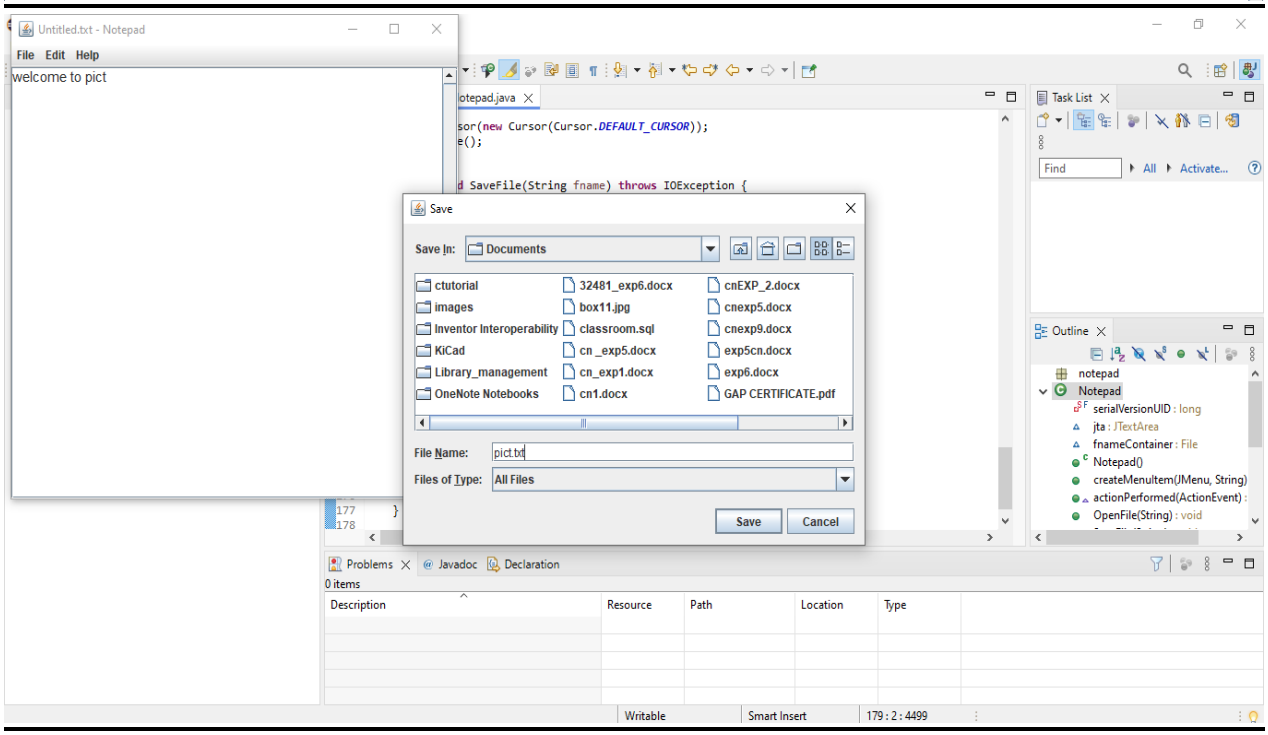
```
            }else if(e.getActionCommand().equals("Copy")){
                //copy
                jta.copy();
            }else if(e.getActionCommand().equals("Paste")){
                //paste
                jta.paste();
            }else if(e.getActionCommand().equals("About Notepad")){
                //about
                JOptionPane.showMessageDialog(this,"Created for making noted on pc","Notepad",JOptionPane.INFORMATION_MESSAGE);
            }else if(e.getActionCommand().equals("Cut")){
                jta.cut();
            }
        }

        public void OpenFile(String fname) throws IOException {
            //open file code here
            BufferedReader d=new BufferedReader(new InputStreamReader(new FileInputStream(fname)));
            String l;
            //clear the textbox
            jta.setText("");

            setCursor(new Cursor(Cursor.WAIT_CURSOR));

            while((l=d.readLine())!= null) {
                jta.setText(jta.getText()+l+"\r\n");
            }

            setCursor(new Cursor(Cursor.DEFAULT_CURSOR));
            d.close();
        }

        public void SaveFile(String fname) throws IOException {
            setCursor(new Cursor(Cursor.WAIT_CURSOR));
            DataOutputStream o=new DataOutputStream(new FileOutputStream(fname));
            o.writeBytes(jta.getText());
            o.close();
            setCursor(new Cursor(Cursor.DEFAULT_CURSOR));
        }

        public void windowDeactivated(WindowEvent e){}
        public void windowActivated(WindowEvent e){}
        public void windowDeiconified(WindowEvent e){}
```

```
	public void windowIconified(WindowEvent e){}
	public void windowClosed(WindowEvent e){}

	public void windowClosing(WindowEvent e) {
		Exiting();
	}

	public void windowOpened(WindowEvent e){}

	public void Exiting() {
		System.exit(0);
	}

}
```


Start
Initialize Notepad window
Create menu bar and menus
Handle "Open" file event
Read file contents
Display file contents in text area
Handle "Exit" event
Close Notepad
End

5.RESULT:

Screen shot of output



6. CONCLUSION:

In conclusion, the development of the Simple Java Notepad Application has provided a solid foundation for a versatile text editing tool. By implementing basic text editing functionalities, file operations, and a user-friendly interface, the application meets the immediate needs of users for creating and managing text files.

However, the project's potential extends beyond its current capabilities. With future enhancements in areas such as advanced text editing features, cloud integration, collaborative editing, and customization options, the application can evolve into a comprehensive solution for users across various domains.

Through continuous development and feedback-driven iterations, the Simple Java Notepad Application can become a go-to tool for individuals and professionals seeking a reliable and feature-rich text editing experience. Its open-ended nature allows for further innovation and adaptation to meet the evolving needs of users in the digital age.

7.APPLICATIONS:

1 .Education

Students can use the notepad application for taking lecture notes, jotting down important information, and organizing study materials.

Teachers can use it for creating lesson plans, drafting assignments, and preparing lecture notes.

2.Programming and Software Development:

Programmers and developers can utilize the notepad application for writing and editing code snippets, scripts, and configuration files.

It can serve as a lightweight code editor for quick edits and testing purposes.

3.Business and Office Work:

Professionals in offices can use the notepad application for drafting emails, writing meeting agendas, and keeping track of tasks and to-do lists.

It can be used for quick note-taking during meetings, brainstorming sessions, and interviews.

4.**Creative Writing and Content Creation:**

Writers and authors can utilize the notepad application for drafting articles, blog posts, stories, and other creative content.

It can serve as a platform for brainstorming ideas, outlining plots, and organizing thoughts.

5.Research and Academia:

Researchers and academics can use the notepad application for writing research papers, journal articles, thesis drafts, and scholarly publications.

It can be used for annotating documents, summarizing findings, and keeping track of references and citations.

8. FUTURE SCOPE:

Expand the application by incorporating advanced text editing features such as syntax highlighting, code folding, and auto-completion, catering to the needs of programmers and technical writers.
Enable users to synchronize their notes with cloud storage platforms like Google Drive, Dropbox, or OneDrive, providing seamless access to their documents across devices.
Implement real-time collaboration features, allowing multiple users to work on the same document simultaneously, with changes being synchronized in real-time.
Ensure accessibility compliance and support for multiple languages, making the application accessible to a wider audience and accommodating diverse user preferences.

9. COPY RIGHT AFFIRMATION:

We undersigned pledge and represent that the source code printed in this project report does not violate any proprietary or personal rights of others (including, without limitation, any copyrights or privacy rights); that the Work is factually accurate and contains no matter libellous or otherwise unlawful; that we have substantially participated in the creation of the Work and that it represents our original work sufficient for us to claim authorship.

Name of students **Sign**

1 Gayatri Anil Taware

2 Sakshi Annasaheb Thorat