Text Editor

A report submitted for the course named Project I (CS-200)

By

RAJESH KUMAR

Bachelor of Technology, IV Semester

Roll No. 15010105

Department of Computer Science and Engineering

Indian Institute of Information Technology Manipur

May, 2017

Abstract

In this work, i attempt to first read about text editors that we have available in markets . now categorise them in different sub groups like some one are best for commercial/official and someone are best for programming . We subdivide our problem in different parts to complete it .

In this work , i firstly analyse different types of Text Editors , then i came to know that they have some of the common features in them . Like opening a Text file and Saving a Text File , Editing parts like cut,copy,paste,select all etc . According to my skill i'm trying to make a new text editor who have same features like any other have . I made it in netbeans .For doing this firstly i divide my planning in two stages first one is gui for my text editor and next one is coding .

For this i choose four lebels (File,Edit,Format,About),one Text Area (for accepting text in it),and one Text Field(which shows the current status like ,New file or File is saved),and four Combo Box, to perform different functions .

Keywords -

jCombo Box,frames,labels,Model property,Editable, Word Wrap,LineWrap,Text,Editor,Cut,Copy, Paste,Select all,Delete,Case, Upper case,Lower case ,Length,New ,Open,Save as .

Declaration

I declare that this submission represents my idea in my own words and where others' idea or words have been included, I have adequately cited and referenced the original source. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/sources in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and can also evoke penal action from the sources which have thus not been properly cited or from proper permission has not been taken when needed.

(*Signature*)

(RAJESH KUMAR)

Date: (15010105)

Department of Computer Science & Engineering
Indian Institute of Information Technology Manipur

Dr. Nongmeikapam Kishorjit Singh Email: kishorjit@iiitmanipur.ac.in
Assistant Professor

To Whome It May Concern

This is to certify that the report entitled **"Text Editor"** submitted to by "RAJESH KUMAR", has been carried out under my supervision and that this work has not been submitted elsewhere for a degree,diploma or a course.

Signature of Supervisor

(Dr. Nongmeikapam Kishorjit Singh)

Acknowledgement

Dr. Kabita Thaoroijam (Ph D)
Assistant Professor (CSE)

Dr. Nongmeikapam Kishorjit Singh (Ph D)
Assistant Professor (CSE)

Dr Navanath Saharia (Ph D)
Assistant Professor (CSE)

Dr Thoudam Doren Singh (Ph D)
Assistant Professor (CSE)

- Rajesh Kumar

Contents

List of Tables

List of Figures

List of abbreviations

	J
jCombo Box	Java Combo Box

Chapter 1

Introduction

A QUOTATION RELATED TO YOUR PROBLEM STATEMENT

"······Anyone and everyone taking a writing class knows that the secret of good writing is to cut it back, pare it down, winnow, chop, hack, prune, and trim, remove every superfluous word, compress, compress, compress...

Actually, when you think about it, not many novels in the Spare tradition are terribly cheerful. Jokes you can usually pluck out whole, by the roots, so if you're doing some heavy-duty prose-weeding, they're the first to go. And there's some stuff about the whole winnowing process I just don't get. Why does it always stop when the work in question has been reduced to sixty or seventy thousand words--entirely coincidentally, I'm sure, the minimum length for a publishable novel? I'm sure you could get it down to twenty or thirty if you tried hard enough. In fact, why stop at twenty or thirty? Why write at all? Why not just jot the plot and a couple of themes down on the back of an envelope and leave it at that? The truth is, there's nothing very utilitarian about fiction or its creation, and I suspect that people are desperate to make it sound manly, back-breaking labor because it's such a wussy thing to do in the first place. The obsession with austerity is an attempt to compensate, to make writing resemble a real job, like farming, or logging. (It's also why people who work in advertising put in twenty-hour days.) Go on, young writers--treat yourself to a joke, or an adverb! Spoil yourself! Readers won't mind!”

– Nick Hornby, The Polysyllabic Spree ······

1.1 A brief study on text editors

There are lots of text editors. Some of these aspire to just be Notepad replacements, while others are full-on programming text editors.
If you are looking for a text editor with programming or source code editing features, such as Notepad++, please take a look at Best Free Programming Editor. Or if you are searching for a text editor that is good for editing html codes, you might want to check out Best Free HTML Editor.
This review will only focus on text editors used for general purpose, saving files in plain text format. What's more? It includes some free programs which give more useful features, such as tabbed interface, than those available to Notepad, and they still remain reasonably lightweight, fast and reliable.

1.2 Refereneces

In order to make this text editor, I explore about all the text editors and thier working strategies. As this project is based on the text informatics, I gothrough the following text book as my reference.

A Text Book on INFORMATICS PRACTICES [1]

1.3 Other Text Editors

AEdit SX - ultra-fast text editor, with spell checker, print preview and setup.
DocPad - a free alternative to Notepad for plain text editing, offering an arsenal of time-saving text editing tools.
Writer’s D’Lite, a lightweight text editor good for writing short stories and articles without any bells and whistles to distract you.
QJot, a small USB portable alternative rich text editor that reads and writes .doc files, inserts images and more.
Jovial Notepad, a lightweight Microsoft Notepad replacement program that

offers some handy and useful features that are used every day.

1.4 Goal

my goal is to design a text editor which will be looking as like any other text editors are.

Chapter 2

Existing System Study

Outline: This chapter presents the following:

1. **A brief introduction .....**
2. **A brief ....**

2.1 Introduction

we have lots of text editors in the market , I got interest from them and i decided to know how they works . So i planned to do it , there is different types of text editors like Microsoft Text Editor , vi Text Editor , simple Notepad, Notepad+,Notepad++,LibreOffice writer,TextEditor,and much more , so firstly i analyse the overview of each and functionalities of them , then i decided to built this in netbeans ehich have similiar functions like any other have .like cut, copy,paste,delete,select all,case changing,length finding,and the basis for when to save and when to open with proper error checking conditions .

Description of Figure 2.1. .... this is my own made text editor , i have given name it to RajEditor .

2.2 study on text editor

Some text editors are small and simple, while others offer broad and complex functions. For example, Unix and Unix-like operating systems have the pico editor (or a variant), but many also include the vi and Emacs editors. Microsoft Windows systems come with the simple Notepad, though many people—especially programmers—prefer other editors with more features. Under Apple Macintosh's classic Mac OS there was the native SimpleText, which was replaced in Mac OS X by TextEdit, which combines features of a text editor with those typical of a word processor such as rulers, margins and multiple font selection. These features are not available simultaneously, but must be switched by user command, or through the program automatically determining the file type.

2.2.1 Typical features

Find and replace – Text editors provide extensive facilities for searching and replacing text, either on groups of files or interactively. Advanced editors can use regular expressions to search and edit text or code. Cut, copy, and paste – most text editors provide methods to duplicate and move text within the file, or between files. Ability to handle UTF-8 encoded text. Text formatting – Text editors often provide basic formatting features like line wrap, auto-indentation, bullet list formatting using ASCII characters, comment formatting, syntax highlighting and so on. Undo and redo – As with word processors, text editors provide a way to undo and redo the last edit. Often—especially with older text editors—there is only one level of edit history remembered and successively issuing the undo command will only ”toggle” the last change.

Modern or more complex editors usually provide a multiple level history such that issuing the undo command repeatedly will revert the document to successively older edits. A separate redo command will cycle the edits ”forward” toward the most recent changes. The number of changes remembered depends upon the editor and is often configurable by the user.

2.3 Summary

In this chapter, we describe basically different types of text editors with there specific functionalities. different text editors are made for different editing purpose .like
Source code editors are text editors with additional functionality to facilitate the production of source code. These often feature user-programmable syntax highlighting and code navigation functions as well as coding tools or keyboard macros similar to an HTML editor (see below). Folding editors. This subclass includes so-called ”orthodox editors” that are derivatives of Xedit. Editors that implement folding without programing-specific features are usually called outliners (see below).
IDEs (integrated development environments) are designed to manage and streamline large programming projects. They are usually only used for programming as they contain many features unnecessary for simple text editing. World Wide Web authors are offered a variety of HTML editors dedicated to the task of creating web pages. These include: Dreamweaver, KompoZer and E Text Editor. Many offer the option of viewing a work in progress on a built-in HTML rendering engine or standard web browser. Most web development is done in a dynamic programming language such as Ruby or PHP using a source code editor or IDE. The HTML delivered by all but the simplest static web sites is stored as individual template files that are assembled by the software controlling the site and do not compose a complete HTML document.
Mathematicians, physicists, and computer scientists often produce articles and books using TeX or LaTeX in plain text files. Such documents are often produced by a standard text editor, but some people use specialized TeX editors. Outliners. Also called tree-based editors, because they combine a hierarchical outline tree with a text editor. Folding (see above) can be considered a specialized form of outlining.
Collaborative editors allow multiple users to work on the same document simultaneously from remote locations over a network. The changes made by individual users are tracked and merged into the document automatically to eliminate the possibility of conflicting edits. These editors also typically include an online chat component for discussion among editors. Simultaneous editing is a technique in End-user development research to edit all items in a multiple selection. It allows the user to manipulate all the selected items at once through direct manipulation. The Lapis text editor and the multi

edit plugin for gedit are examples of this technique. The Lapis editor can also create an automatic multiple selection based on an example item. Distraction-free editors provide a minimalistic interface with the purpose of isolating the writer from the rest of the applications and operating system, thus being able to focus on the writing without distractions from interface elements like a toolbar or notification area.

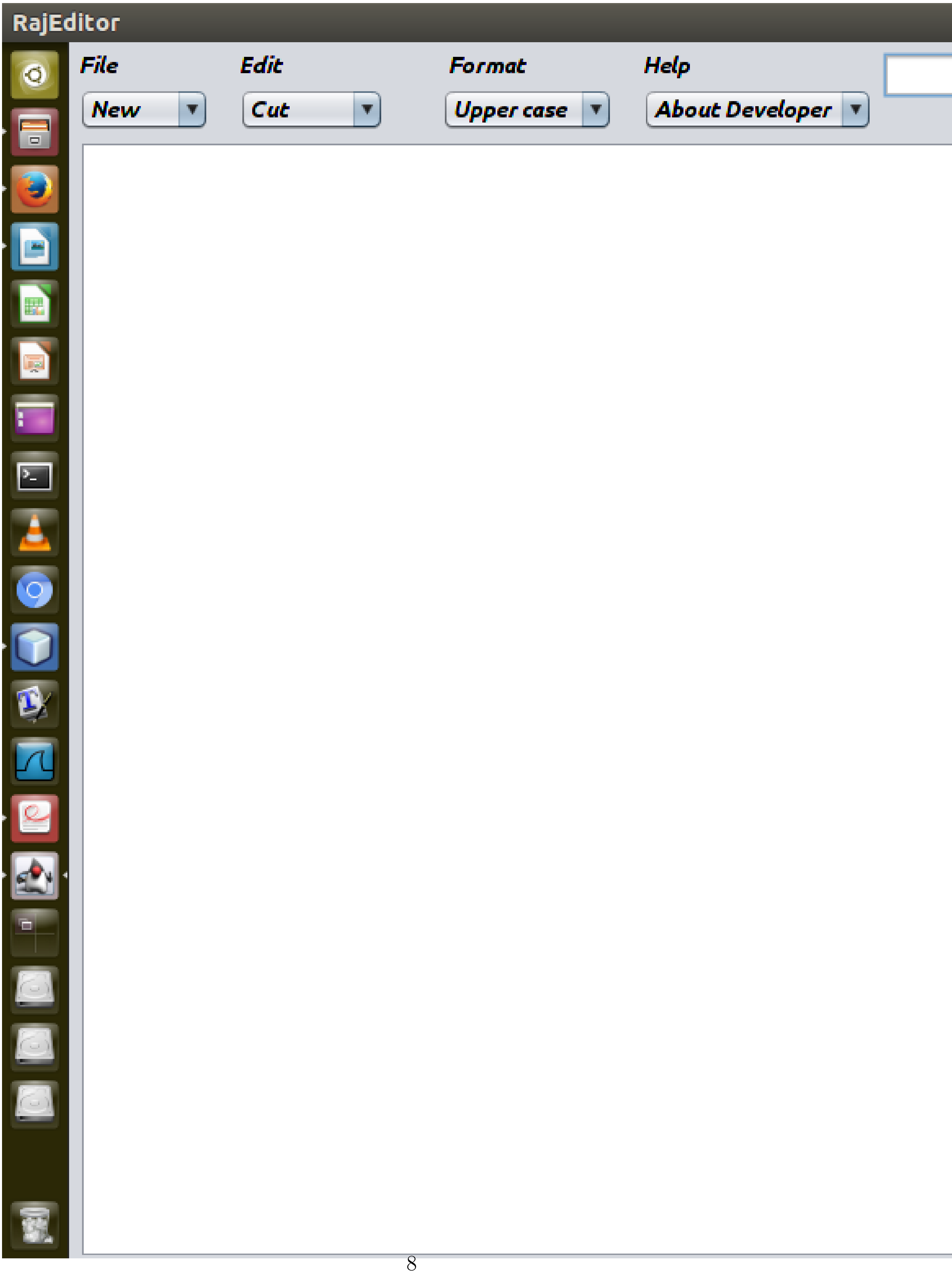


Figure 2.1: RajEditor

Chapter 3

System Analysis, Design & Implementation

Outline: **This chapter presents the following:**

1. **A brief ....**
2. **A brief ....**

3.1 Introduction

Basically any kind of text editor is a kind of application in which can open a text , and we can save any text file at any location .
so for doing that i know that text area is better for accepting the text for my text editor , and to show the current status text field is good.
now to perform Edit ,File ,Formating and help menu bar i choose j combo boxes because it gives a new look from any other text editors . and for guiding this i choose four labels .
after completing my design write there source code according to the functions they have to perform .
after complete design i make a exe file for this jar by using launch 4j .

3.2 Summary

Text Editors is good to write and keep track of every day work digitally . and it is easier , user friendly and it requires less space .
my text editor is ready to perform all task whatevere i have provided , with complete error handling .

Chapter 4

Conclusion

my application is working good with provided functions . And ready to handle logical error like , while writing user unfortunately clicks new or exit , then there is confirm dialog for it , it ask whether you really want to perform this action .
(a) So finally it is easy to use . (b) new design from other text editors . (c) User friendly . (d) open source software . (e) easy to share . (f) light weigth (less memory) . (g) works smoother .

4.1 Future direction

In future i want to add more features in it like , automatic spell check . And desire to add more functions in it like replace any word , and more user friendly .

Appendix A

Screenshot and Description of the Implemented System

A.1 setup file and main screen

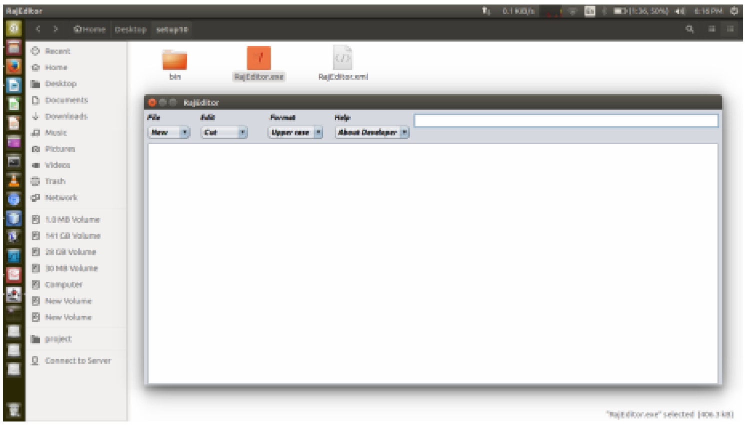
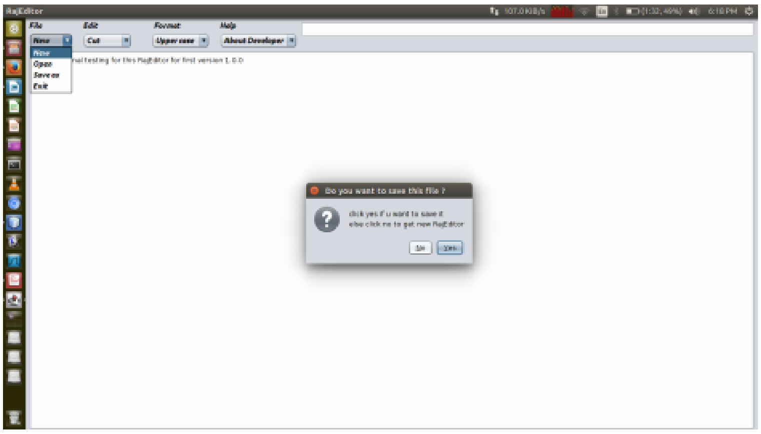


Figure A.1: Image caption

Descriptoin of Figure A.1 setup folder and main screen .

A.2 File menu bar(New)

Descriptoin of Figure A.2 confirming whether you wants to save before opens a new one .



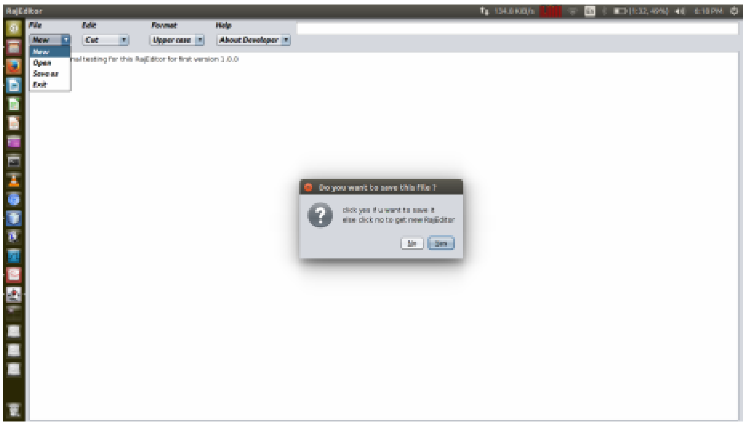
nal testing for this RajEditor for first version 1.0.0

Do you want to save this file ?

click yes if u want to save it
else click no to get new RajEditor

Figure A.2: New Tab

A.3 File menu bar(Save as)



nal testing for this RajEditor for first version 1.0.0

Do you want to save this file ?

click yes if u want to save it
else click no to get new RajEditor

Figure A.3: Save tab

Descriptoin of Figure A.3 Save at specified location .

A.4 File menu bar(Open)

Descriptoin of Figure A.4 Opening a text file from specified location .

A.5 Edit menu bar(Cut,Copy,Paste)

Descriptoin of Figure A.5 performing cut,copy,paste operations .

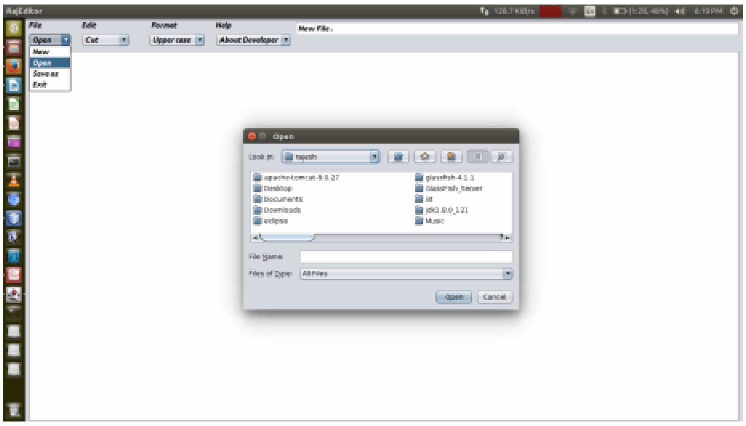
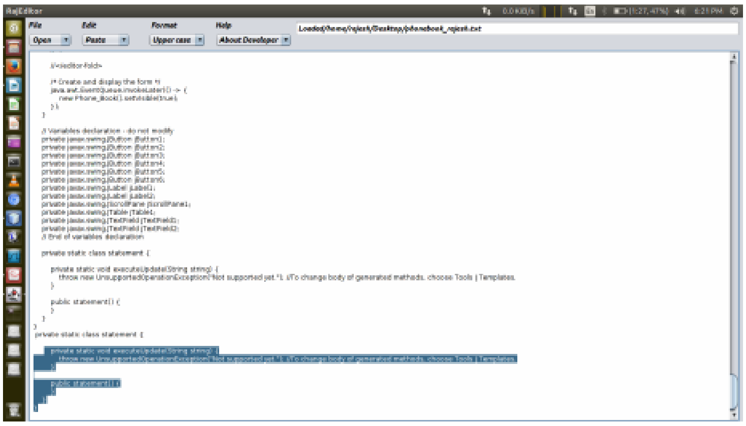


Figure A.4: Opening a text file from specified location .

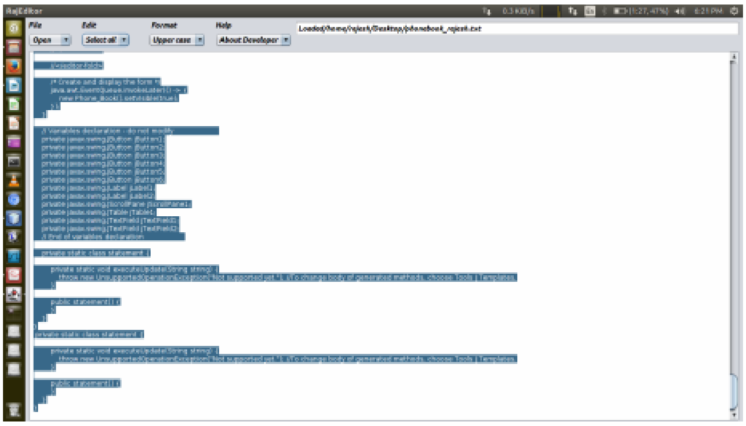


Loaded/home/rajesh/Desktop/phonebook_rajesh.txt

```
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(() -> {
            new Phone_Book().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.JButton jButton5;
    private javax.swing.JButton jButton6;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTable jTable1;
    private javax.swing.JTextField jTextField1;
    private javax.swing.JTextField jTextField2;
    // End of variables declaration

    private static class statement {

        private static void executeUpdate(String string) {
            throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.
        }

        public statement() {
        }
    }
}
private static class statement {

        private static void executeUpdate(String string) {
            throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.
        }

        public statement() {
        }
    }
}
```

Figure A.5: Performing cut and paste

A.6 Edit menu bar(Select all)



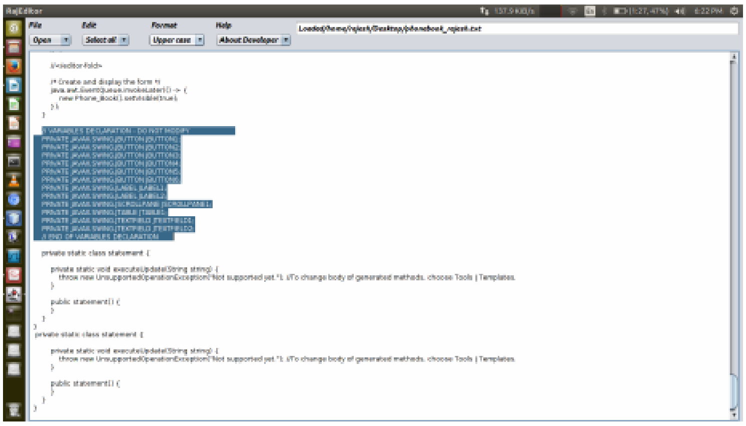
Loaded/home/rajesh/Desktop/phonebook_rajesh.txt

```
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(() -> {
            new Phone_Book().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.JButton jButton5;
    private javax.swing.JButton jButton6;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTable jTable1;
    private javax.swing.JTextField jTextField1;
    private javax.swing.JTextField jTextField2;
    // End of variables declaration

    private static class statement {

        private static void executeUpdate(String string) {
            throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.
        }

        public statement() {
        }
    }
}
private static class statement {

        private static void executeUpdate(String string) {
            throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.
        }

        public statement() {
        }
    }
}
```

Figure A.6: Performing select all operation

Descriptoin of Figure A.6 selects all the texts from text area.

A.7 Format menu bar(Upper case)



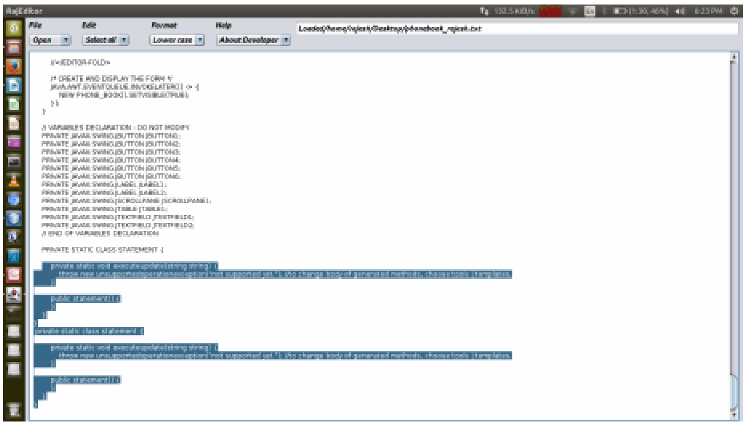
Loaded/home/rajesh/Desktop/phonebook_rajesh.txt

```
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(() -> {
            new Phone_Book().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.JBUTTON JBUTTON5;
    PRIVATE JAVAX.SWING.JBUTTON JBUTTON6;
    PRIVATE JAVAX.SWING.JLABEL JLABEL1;
    PRIVATE JAVAX.SWING.JLABEL JLABEL2;
    PRIVATE JAVAX.SWING.JSCROLLPANE JSCROLLPANE1;
    PRIVATE JAVAX.SWING.JTABLE JTABLE1;
    PRIVATE JAVAX.SWING.JTEXTFIELD JTEXTFIELD1;
    PRIVATE JAVAX.SWING.JTEXTFIELD JTEXTFIELD2;
    // END OF VARIABLES DECLARATION

    private static class statement {

        private static void executeUpdate(String string) {
            throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.
        }

        public statement() {
        }
    }
}
private static class statement {

        private static void executeUpdate(String string) {
            throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.
        }

        public statement() {
        }
    }
}
```

Figure A.7: Converting selected text to upper case.

Descriptoin of Figure A.7 converts selected text to upper case .

A.8 Format menu bar(Lower case)



Loaded/home/rajesh/Desktop/phonebook_rajesh.txt

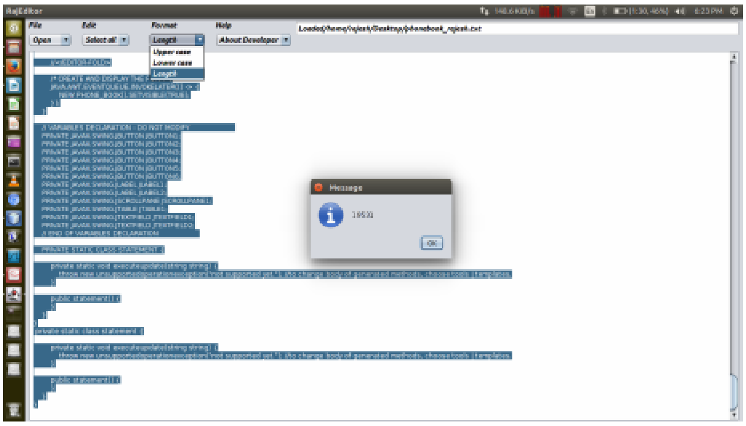
```
        //</EDITOR-FOLD>

        /* CREATE AND DISPLAY THE FORM */
        JAVA.AWT.EVENTQUEUE.INVOKELATER(() -> {
            NEW PHONE_BOOK().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.JBUTTON JBUTTON5;
    PRIVATE JAVAX.SWING.JBUTTON JBUTTON6;
    PRIVATE JAVAX.SWING.JLABEL JLABEL1;
    PRIVATE JAVAX.SWING.JLABEL JLABEL2;
    PRIVATE JAVAX.SWING.JSCROLLPANE JSCROLLPANE1;
    PRIVATE JAVAX.SWING.JTABLE JTABLE1;
    PRIVATE JAVAX.SWING.JTEXTFIELD JTEXTFIELD1;
    PRIVATE JAVAX.SWING.JTEXTFIELD JTEXTFIELD2;
    // END OF VARIABLES DECLARATION

    PRIVATE STATIC CLASS STATEMENT {

        private static void executeupdate(string string) {
            throw new unsupportedoperationexception("not supported yet."); //to change body of generated methods, choose tools | templates.
        }

        public statement() {
        }
    }
}
private static class statement {

        private static void executeupdate(string string) {
            throw new unsupportedoperationexception("not supported yet."); //to change body of generated methods, choose tools | templates.
        }

        public statement() {
        }
    }
}
```

Figure A.8: Converting selected text to lower case.

Descriptoin of Figure A.8 convert selected text to lower case.

A.9 Format menu bar(Length)

Descriptoin of Figure A.9 finding the length of selected text including space .



Loaded/home/rajesh/Desktop/phonebook_rajesh.txt

```
        }//GEN-LAST:EVENT_JBUTTON6ACTIONPERFORMED

            /* CREATE AND DISPLAY THE FORM */
            JAVA.AWT.EVENTQUEUE.INVOKELATER(() -> {
                NEW PHONE_BOOK().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.JBUTTON JBUTTON5;
    PRIVATE JAVAX.SWING.JBUTTON JBUTTON6;
    PRIVATE JAVAX.SWING.JLABEL JLABEL1;
    PRIVATE JAVAX.SWING.JLABEL JLABEL2;
    PRIVATE JAVAX.SWING.JSCROLLPANE JSCROLLPANE1;
    PRIVATE JAVAX.SWING.JTABLE JTABLE1;
    PRIVATE JAVAX.SWING.JTEXTFIELD JTEXTFIELD1;
    PRIVATE JAVAX.SWING.JTEXTFIELD JTEXTFIELD2;
    // END OF VARIABLES DECLARATION

    PRIVATE STATIC CLASS STATEMENT {

        private static void executeupdate(string string) {
            throw new unsupportedoperationexception("not supported yet."); //to change body of generated methods, choose tools | templates.
        }

        public statement() {
        }
    }
}
private static class statement {

        private static void executeupdate(string string) {
            throw new unsupportedoperationexception("not supported yet."); //to change body of generated methods, choose tools | templates.
        }

        public statement() {
        }
    }
}
```

18531

Figure A.9: finding the length of selected text

Appendix B

User manual

B.1 Introduction

Description of the system.
my application contains four labels(File,Edit,Format,Help).under each label everyone performs according to there names specified .like
New – it will open new file , if user is currently writing something if he unfortunately clicks New then there's a confirm Dialog box came for asking whether you are confirmed to open New file .
Open – it will open a JFileChooser chooser which will ask to open from which location you want to choose the text file .

Save as – it will open JfileChooser box which will ask where to save the current working file .

Exit – it will ask whether you want to exit or not , if no is clicked on confirmed dialog button then it will still be on the same file , else clicked on yes it will terminate the application .

Cut – it will cut the selected text .

Copy – it will copy the selected text .

Paste – it will paste the text .

Delete – it will delete the selected text .

Select all – it will simply select all the text present in text area .

Upper case – it will convert the selected text to upper case .

lower case – it will convert the selected text to lower case . Length – it will show the length of selected text (including space) .

About Developer – this will show the details about developer .

Send feedback – for enhancement of app in future , user can send there feedback at this details , which came from send feedback list of combo box .

About RajEditor – this will show the details of making this text editor . And what it includes .

B.2 Step to install my implemented system

This is my system geneted output to install this text editor.

When you build an Java application project that has a main class, the IDE automatically copies all of the JAR files on the projects classpath to your projects dist/lib folder. The IDE also adds each of the JAR files to the Class-Path element in the application JAR files manifest file (MANIFEST.MF).

To run the project from the command line, go to the dist folder and type the following:

```
java -jar ”tedit.jar”
```

To distribute this project, zip up the dist folder (including the lib folder) and distribute the ZIP file.

Notes:

(a)If two JAR files on the project classpath have the same name, only the first JAR file is copied to the lib folder.
(b)Only JAR files are copied to the lib folder. If the classpath contains other types of files or folders, these files (folders) are not copied.
(c)If a library on the projects classpath also has a Class-Path element specified in the manifest,the content of the Class-Path element has to be on the projects runtime path.
(d)To set a main class in a standard Java project, right-click the project node in the Projects window and choose Properties. Then click Run and enter the class name in the Main Class field. Alternatively, you can manually type the class name in the manifest Main-Class element.

Step to install implemented system

1. As i provided both jar and exe file for my text editor ,
so for both these are the steps :
1.1 steps for jar file :
(a) for jar file : copy the provided jar on any location on your system . For example : /home/rajesh/Desktop/setup10/bin under this you copied it .
(b) run it by terminal : first go to bin directory then type

```
java -jar tedit.jar
```

here "tedit" is the file name during creating this application .

1.2 steps for installation of exe file
(a) copy the exe file on your system .
(b) open it with "Open JDK java 8 runtime".

Note For this application you can use any version of jdk above then 1.4.0

Bibliography

[1] C. XI and H. HOMEWORK, “Informatics practices.”