**PROGRAM CODING:**

import javax.swing.\*;

import javax.swing.text.\*;

import java.awt.\*;

import java.io.\*;

import java.awt.event.\*;

import java.util.Hashtable;

public class SimpleEditor extends JFrame {

public static void main(String[] args) {

SimpleEditor editor = new SimpleEditor( );

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

editor.setVisible(true);

}

public SimpleEditor( ) {

super("Swing Editor");

textComp = createTextComponent( );

makeActionsPretty( );

Container content = getContentPane( );

content.add(textComp, BorderLayout.CENTER);

content.add(createToolBar( ), BorderLayout.NORTH);

setJMenuBar(createMenuBar( ));

setSize(320, 240);

}

protected JTextComponent createTextComponent( ) {

JTextArea ta = new JTextArea( );

ta.setLineWrap(true);

return ta;

}

protected void makeActionsPretty( ) {

Action a;

a = textComp.getActionMap( ).get(DefaultEditorKit.cutAction);

a.putValue(Action.SMALL\_ICON, new ImageIcon("4.png"));

a.putValue(Action.NAME, "Cut");

a = textComp.getActionMap( ).get(DefaultEditorKit.copyAction);

a.putValue(Action.SMALL\_ICON, new ImageIcon("7.png"));

a.putValue(Action.NAME, "Copy");

a = textComp.getActionMap( ).get(DefaultEditorKit.pasteAction);

a.putValue(Action.SMALL\_ICON, new ImageIcon("8.png"));

a.putValue(Action.NAME, "Paste");

a = textComp.getActionMap( ).get(DefaultEditorKit.selectAllAction);

a.putValue(Action.NAME, "Select All");

}

protected JToolBar createToolBar( ) {

JToolBar bar = new JToolBar( );

bar.add(getOpenAction( )).setText("");

bar.add(getSaveAction( )).setText("");

bar.addSeparator( );

bar.add(textComp.getActionMap( ).get(DefaultEditorKit.cutAction)).setText("");

bar.add(textComp.getActionMap( ).get(DefaultEditorKit.copyAction)).setText("");

bar.add(textComp.getActionMap( ).get(DefaultEditorKit.pasteAction)).setText("");

return bar;

}

protected JMenuBar createMenuBar( ) {

JMenuBar menubar = new JMenuBar( );

JMenu file = new JMenu("File");

JMenu edit = new JMenu("Edit");

menubar.add(file);

menubar.add(edit);

file.add(getOpenAction( ));

file.add(getSaveAction( ));

file.add(new ExitAction( ));

edit.add(textComp.getActionMap( ).get(DefaultEditorKit.cutAction));

edit.add(textComp.getActionMap( ).get(DefaultEditorKit.copyAction));

edit.add(textComp.getActionMap( ).get(DefaultEditorKit.pasteAction));

edit.add(textComp.getActionMap( ).get(DefaultEditorKit.selectAllAction));

return menubar;

}

protected Action getOpenAction( ) { return openAction; }

protected Action getSaveAction( ) { return saveAction; }

protected JTextComponent getTextComponent( ) { return textComp; }

private Action openAction = new OpenAction( );

private Action saveAction = new SaveAction( );

private JTextComponent textComp;

private Hashtable actionHash = new Hashtable( );

public class ExitAction extends AbstractAction {

public ExitAction( ) { super("Exit"); }

public void actionPerformed(ActionEvent ev) { System.exit(0); }

}

class OpenAction extends AbstractAction {

public OpenAction( ) {

super("Open", new ImageIcon("3.png"));

}

public void actionPerformed(ActionEvent ev) {

JFileChooser chooser = new JFileChooser( );

if (chooser.showOpenDialog(SimpleEditor.this) !=

JFileChooser.APPROVE\_OPTION)

return;

File file = chooser.getSelectedFile( );

if (file == null)

return;

FileReader reader = null;

try {

reader = new FileReader(file);

textComp.read(reader, null);

}

catch (IOException ex) {

JOptionPane.showMessageDialog(SimpleEditor.this,

"File Not Found", "ERROR", JOptionPane.ERROR\_MESSAGE);

}

finally {

if (reader != null) {

try {

reader.close( );

} catch (IOException x) {}

}

}

}

}

class SaveAction extends AbstractAction {

public SaveAction( ) {

super("Save", new ImageIcon("1.png"));

}

public void actionPerformed(ActionEvent ev) {

JFileChooser chooser = new JFileChooser( );

if (chooser.showSaveDialog(SimpleEditor.this) !=

JFileChooser.APPROVE\_OPTION)

return;

File file = chooser.getSelectedFile( );

if (file == null)

return;

FileWriter writer = null;

try {

writer = new FileWriter(file);

textComp.write(writer);

}

catch (IOException ex) {

JOptionPane.showMessageDialog(SimpleEditor.this,

"File Not Saved", "ERROR", JOptionPane.ERROR\_MESSAGE);

}

finally {

if (writer != null) {

try {

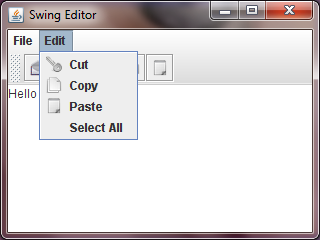
writer.close( );

} catch (IOException x) {}

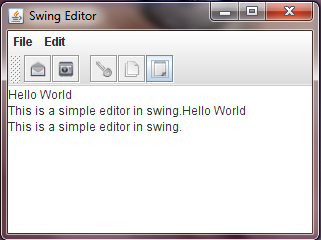
} } } }}

**OUTPUT:**

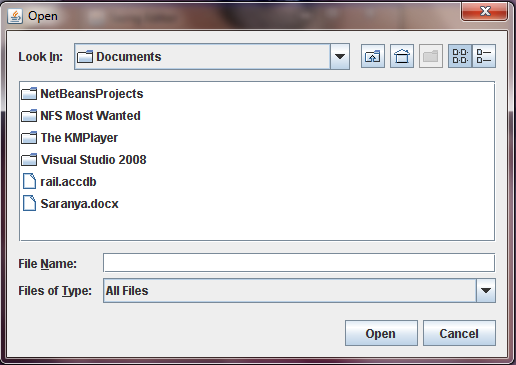
/\* Edit \*/



/\* Copy Action\*/



/\* Open Action \*/



/\* Exit \*/

