Nathan Gaffney

Lab 8

28-Oct-14

/\*------------------------------  
Edited by: Nathan Gaffney  
28-OCT-2014  
CST 183 JAva  
--------------------------------\*/  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.\*;  
  
public class NoteTaker extends JFrame   
{  
 //constants for set up of note taking area  
 public static final int WIDTH = 600;  
 public static final int HEIGHT = 300;  
 public static final int LINES = 13;  
 public static final int CHAR\_PER\_LINE = 45;  
  
 //objects in GUI  
 private JTextArea theText; //area to take notes  
 private JMenuBar mBar; //horizontal menu bar  
 private JPanel textPanel; //panel to hold scrolling text area   
 private JMenu notesMenu; //vertical menu with choices for notes  
  
 //\*\*\*\*THESE ITEMS ARE NOT YET USED. YOU WILL BE CREATING THEM IN THIS LAB  
 private JMenu viewMenu; //vertical menu with choices for views   
 private JMenu lafMenu; //vertical menu with look and feel  
 private JMenu sbMenu; //vertical menu with scroll bar option  
 private JScrollPane scrolledText;//scroll bars   
  
 //default notes  
 private String note1 = "No Note 1.";  
 private String note2 = "No Note 2.";  
  
 /\*\*constructor\*/   
 public NoteTaker()  
 {  
 //create a closeable JFrame with a specific size  
 super("Note Taker");   
 setSize(WIDTH, HEIGHT);  
 setDefaultCloseOperation (JFrame.EXIT\_ON\_CLOSE);  
   
 //get contentPane and set layout of the window  
 Container contentPane = getContentPane();  
 contentPane.setLayout(new BorderLayout());  
  
 //creates the vertical menus   
 createNotes();  
 createViews();  
   
 //creates horizontal menu bar and   
 //adds vertical menus to it  
 mBar = new JMenuBar();  
 mBar.add(notesMenu);  
 mBar.add(viewMenu);  
 //\*\*\*\*ADD THE viewMenu TO THE MENU BAR HERE  
 setJMenuBar(mBar);  
  
 //creates a panel to take notes on  
 textPanel = new JPanel();  
 textPanel.setBackground(Color.blue);  
 theText = new JTextArea(LINES, CHAR\_PER\_LINE);  
 theText.setBackground(Color.white);  
 //\*\*\*\*CREATE A JScrollPane OBJECT HERE CALLED scrolledText  
 scrolledText = new JScrollPane(theText);  
 //\*\*\*\*AND PASS IN theText, THEN   
 //\*\*\*\*CHANGE THE LINE BELOW BY PASSING IN scrolledText  
 textPanel.add(scrolledText);  
 contentPane.add(textPanel, BorderLayout.CENTER);  
 }  
  
 /\*\*creates vertical menu associated with Notes   
 menu item on menu bar\*/  
 public void createNotes()  
 {  
 notesMenu = new JMenu("Notes");  
 JMenuItem item;  
   
 item = new JMenuItem("Save Note 1");  
 item.addActionListener(new MenuListener());  
 notesMenu.add(item);  
  
 item = new JMenuItem("Save Note 2");  
 item.addActionListener(new MenuListener());  
 notesMenu.add(item);  
  
 item = new JMenuItem("Open Note 1");  
 item.addActionListener(new MenuListener());  
 notesMenu.add(item);  
  
 item = new JMenuItem("Open Note 2");  
 item.addActionListener(new MenuListener());  
 notesMenu.add(item);  
  
 item = new JMenuItem("Clear");  
 item.addActionListener(new MenuListener());  
 notesMenu.add(item);  
  
 item = new JMenuItem("Exit");  
 item.addActionListener(new MenuListener());  
 notesMenu.add(item);  
 }  
   
 /\*\*creates vertical menu associated with Views   
 menu item on the menu bar\*/  
 public void createViews()  
 {  
 viewMenu = new JMenu ("Views");  
 createLookAndFeel();  
 createScrollBars();  
 viewMenu.add(lafMenu);  
 lafMenu.addActionListener(new MenuListener());  
 viewMenu.add(sbMenu);  
 sbMenu.addActionListener(new MenuListener());  
 }  
  
 /\*\*creates the look and feel submenu \*/  
 public void createLookAndFeel()  
 {  
 lafMenu = new JMenu("Look and Feel");  
 JMenuItem item;  
   
 item = new JMenuItem("Metal");  
 item.addActionListener(new MenuListener());  
 lafMenu.add(item);  
   
 item = new JMenuItem("Motif");  
 item.addActionListener(new MenuListener());  
 lafMenu.add(item);  
   
 item = new JMenuItem("Windows");  
 item.addActionListener(new MenuListener());  
 lafMenu.add(item);  
 }  
  
 /\*\*creates the scroll bars submenu\*/  
 public void createScrollBars()  
 {  
 sbMenu = new JMenu("Scroll Bars");  
 JMenuItem item;  
   
 item = new JMenuItem("Never");  
 item.addActionListener(new MenuListener());  
 sbMenu.add(item);  
   
 item = new JMenuItem("Always");  
 item.addActionListener(new MenuListener());  
 sbMenu.add(item);  
   
 item = new JMenuItem("As Needed");  
 item.addActionListener(new MenuListener());  
 sbMenu.add(item);  
 }  
  
 private class MenuListener implements ActionListener  
 {  
   
 public void actionPerformed(ActionEvent e)  
 {  
 String actionCommand = e.getActionCommand();  
 if (actionCommand.equals("Save Note 1"))  
 note1 = theText.getText();  
 else if (actionCommand.equals("Save Note 2"))  
 note2 = theText.getText();  
 else if (actionCommand.equals("Clear"))  
 theText.setText("");  
 else if (actionCommand.equals("Open Note 1"))  
 theText.setText(note1);  
 else if (actionCommand.equals("Open Note 2"))  
 theText.setText(note2);  
 else if (actionCommand.equals("Exit"))  
 System.exit(0);  
 else if (actionCommand.equals("Metal"))  
 {  
 try  
 {  
 UIManager.setLookAndFeel(  
 "javax.swing.plaf.metal.MetalLookAndFeel");  
 SwingUtilities.updateComponentTreeUI(getContentPane());  
 }  
 catch (Exception a)  
 {  
 JOptionPane.showMessageDialog(null,  
 "Error setting the look and feel.");  
 System.exit(0);  
 }  
   
 }  
 else if (actionCommand.equals("Motif"))  
 {  
 try  
 {  
 UIManager.setLookAndFeel(  
 "com.sun.java.swing.plaf.motif.MotifLookAndFeel");  
 SwingUtilities.updateComponentTreeUI(getContentPane());  
 }  
 catch (Exception b)  
 {  
 JOptionPane.showMessageDialog(null,  
 "Error setting the look and feel.");  
 System.exit(0);  
 }  
   
 }  
 else if (actionCommand.equals("Windows"))  
 {  
 try  
 {  
 UIManager.setLookAndFeel(  
 "com.sun.java.swing.plaf.windows.WindowsLookAndFeel");  
 SwingUtilities.updateComponentTreeUI(getContentPane());  
 }  
 catch (Exception c)  
 {  
 JOptionPane.showMessageDialog(null,  
 "Error setting the look and feel.");  
 System.exit(0);  
 }  
   
 }   
 else if (actionCommand.equals("Never"))  
 {  
 scrolledText.setHorizontalScrollBarPolicy(  
 JScrollPane.HORIZONTAL\_SCROLLBAR\_NEVER);  
 scrolledText.setVerticalScrollBarPolicy(  
 JScrollPane.VERTICAL\_SCROLLBAR\_NEVER);  
   
 }  
 else if (actionCommand.equals("Always"))  
 {  
 scrolledText.setHorizontalScrollBarPolicy(  
 JScrollPane.HORIZONTAL\_SCROLLBAR\_ALWAYS);  
 scrolledText.setVerticalScrollBarPolicy(  
 JScrollPane.VERTICAL\_SCROLLBAR\_ALWAYS);   
 }  
 else if (actionCommand.equals("As Needed"))  
 {  
 scrolledText.setHorizontalScrollBarPolicy(  
 JScrollPane.HORIZONTAL\_SCROLLBAR\_AS\_NEEDED);  
 scrolledText.setVerticalScrollBarPolicy(  
 JScrollPane.VERTICAL\_SCROLLBAR\_AS\_NEEDED);  
   
 }  
 else  
 theText.setText("Error in memo interface");  
 }  
 }  
  
 public static void main(String[] args)  
 {  
 NoteTaker gui = new NoteTaker();  
 gui.setVisible(true);  
 }  
}

