**8.13 Menu Components :-**

A Menu Bar is a set of option that allow the user to choose from any one of the  saving option. A list of other option component  a user can choose in a menu bar. In this Tutorial we want to describe you how to create a Menu Bar in your application, First we need to import list of packages that required for creating a menu bar in an application. We defined a class name 'MenuBarExample' extends JFrame.A JFrame is a window working in a swing environment. We define a constructor that is used to create  a JFrame with a specified size of 200\*350.The java  code also describe you to exit when the JFrame when it is closed and show us the window. The JMenuBar is used to control the list of bar at the top of window, this includes File,Edit,View,etc.JMenuBar is positioned at the menubar in the JFrame.The MenuBar is set at the JMenuBar.We define and add three dropdown menu in the menubar i.e

1. **File**
2. **Edit**
3. **View**

After we have done with the list of option in menuBar,then we go for the sub menuBar options i.e. we create and add simple menu item to the list of dropdown menu present in menuBar.The various submenu that we have used in this menu Bar are-

|  |  |  |
| --- | --- | --- |
| File | Edit | View |
| New | Copy | Toolbars |
| Open | Paste | - |
| Exit | Cut | - |

The class implement Action Listener, that is used to add  action listener to submenu item. Action listener is a set of event  generated when a user perform certain action on a component.Example,when a user click or perform a certain operation on button and textfield,as a result a certain action event is generated on the clicking of a button or enter in a text field. The result come out is action performed that is given to all action listener registered with respective component.

**Let us go through list of method  to understand  example program code**

1.newAction.addActionListener(new Action Listener())-This is used to register an instance of the event handler class as a listener on  components

2.public void actionPerformed(ActionEvent arg0)  - The program must register this object as an action listener the component source, using the addActionListener method. When the user clicks the onscreen component, the component fires an action event. This results in the invocation of the action listener's action Performed method. The single argument to the method is an Action Event object that gives information about the event and its source. The System.out.println is used to display the "you have clicked on the open action" as long as you clicked on open submenu.

A menu is a way to arrange buttons. There are several types.

Traditional dropdown menus are positioned across the top of a window in a menu bar, and display below the menu name.

Popup menus appear when the user clicks, eg with the right mouse button, on a component that can handle a popup request.

Dropdown menus: JMenuBar, JMenu, and JMenuItem

A menu bar can be added to the top of a top-level container, eg, JFrame, JApplet, or JDialog. Note that a menu bar can not be added to JPanel.

Dropdown menus have three parts:

JMenuBar is positioned across the top of a container (eg a JFrame, JPanel, or JApplet). It's placed above the content pane, so does not use the container's layout. Add menus to the menubar.

JMenu has a name and contains a number of menu itemsl which are displayed is a vertical list of menu items.

JMenuItems and Separators are added to each menu. Menu items are usually text "buttons", but can also have icons, checkboxes, radio buttons, or be hierarchical submenus.

Menus and Menu Items are Buttons!

It is easy to see how menu items are buttons that appear when a menu appears. But the menu names in the menu bar are also buttons. When you press on these "buttons", they create a popup menu that you see as a dropdown menu.

Keyboard Mnemonics and Accelerators

You can associated characters with menus and menu items so that the user can invoke them from the keyboard:

Menu mnemonics can be used to open a menu by typing a single character associated with a menu along with an operating system defined key for this action. For example, on MS Windows, The ALT key with F will typically open the File menu. You can then select the relevant menu item with either the mnemonic key for that item, or with the arrow keys and Enter. The corresponding letter (char) in the menu will be underlined. For example

fileMenu.setMnemonic('F');

Menu item mnemonics are used to select a menu item when its menu is already open. Typically the character corresponds to the first, or a significant, letter in the menu item name. That letter in the menu item will be underlined.

openItem.setMnemonic('O');

Accelerator key combinations are used to directly invoke a menu item without opening the menu, for example the common CTRL-C (Copy) execute the copy menu action. Accelerator key options are displayed to the right of the menu item name. Adding accelerator key requires using KeyStroke codes. There are several ways to get these codes, but the model below shows one of the easiest.

openItem.setAccelerator(KeyStroke.getKeyStroke("control O"));

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JFrame;

import javax.swing.JMenu;

import javax.swing.JMenuBar;

import javax.swing.JMenuItem;

public class MenuBarExample extends JFrame

{

public MenuBarExample()

{

setTitle("MenuBarExample");

setSize(600, 650);

JMenuBar menuBar = new JMenuBar();

setJMenuBar(menuBar);

JMenu fileMenu = new JMenu("File");

JMenu editMenu = new JMenu("Edit");

JMenu viewMenu = new JMenu("View");

menuBar.add(fileMenu);

menuBar.add(editMenu);

menuBar.add(viewMenu);

JMenuItem newAction = new JMenuItem("New");

JMenuItem openAction = new JMenuItem("Open");

JMenuItem exitAction = new JMenuItem("Exit");

JMenuItem cutAction = new JMenuItem("Cut");

JMenuItem copyAction = new JMenuItem("Copy");

JMenuItem pasteAction = new JMenuItem("Paste");

JMenuItem toolbarsAction= new JMenuItem("toolbarsAction");

fileMenu.add(newAction);

fileMenu.add(openAction);

fileMenu.addSeparator();

fileMenu.add(exitAction);

editMenu.add(cutAction);

editMenu.add(copyAction);

editMenu.add(pasteAction);

editMenu.addSeparator();

viewMenu.add(toolbarsAction);

newAction.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent arg0)

{

System.out.println("You clicked on the new action");

}

}

);

}

public static void main(String[] args)

{

MenuBarExample mba = new MenuBarExample();

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

mba.setVisible(true);

}

}