Module 14

GUI-Based Applications

Objectives

Upon completion of this module, you should be able to:

- Describe how to construct a menu bar, menu, and menu items in a Java GUI
- Understand how to change the color and font of a component

Relevance

Discussion – The following questions are relevant to the material presented in this module:

- You now know how to set up a Java GUI for both graphic output and interactive user input. However, only a few of the components from which GUIs can be built have been described. What other components would be useful in a GUI?

  _____

  _____

  _____

- How can you create a menu for your GUI frame?

  _____

  _____

  _____

How to Create a Menu

A `JMenu` is different from other components because you cannot add a `JMenu` component to ordinary containers and have them laid out by the layout manager. You can add menus only to a *menu container*. This section describes the following procedure to create a menu:

1. Create a `JMenuBar` object, and set it into a menu container, such as a `JFrame`.
2. Create one or more `JMenu` objects, and add them to the menu bar object.
3. Create one or more `JMenuItem` objects, and add them to the menu object.

Note – Pop-up menus are an exception to this rule because they appear as floating windows and, therefore, do not require layout.

Note – With AWT, you could designate one menu to be the Help menu. This was specified by using the `setHelpMenu` method. This method is not implemented in Swing and Help menus are added just like any other menu item.

Creating a JMenuBar

A JMenuBar component is a horizontal menu. You can add it to a JFrame object only, and it forms the root of all menu trees. A JFrame displays one JMenuBar component at a time. However, you can change the JMenuBar based on the state of the program so that different menus appear at various points.

Code 14-1 JMenuBar

```
10   f = new JFrame("JMenuBar");
11   mb = new JMenuBar();
12   f.setJMenuBar(mb);
```

The following is what the code produces in Microsoft Windows.

Figure 14-1 The JMenuBar Component

Note – The appearance of the window varies slightly from operating system to operating system.

The JMenuBar does not support listeners. All the events that might arise in a menu bar are processed by the menus that are added to the menu bar.

Creating a JMenu

The JMenu component provides a basic pull-down menu. You add it either to a JMenuBar or to another JMenu.

Code 14-2 JMenuBar with Top Level JMenu Items

```
13   f = new JFrame("Menu");
14   mb = new JMenuBar();
15   m1 = new JMenu("File");
16   m2 = new JMenu("Edit");
17   m3 = new JMenu("Help");
18   mb.add(m1);
19   mb.add(m2);
20   mb.add(m3);
21   f.setJMenuBar(mb);
```

The code produces the following:

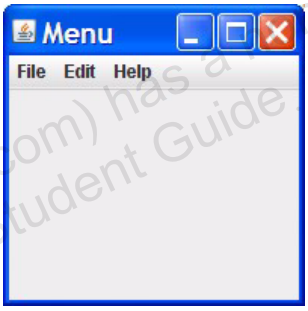

Figure 14-2 The JMenu Component

You *can* add an ActionListener to a JMenu object, but this is unusual. Normally, you use menus to display and control menu items that are described in the following section.

Creating a JMenuItem

The JMenuItem components are the text *leaf* nodes of a menu tree. They are added to a menu to complete it, as shown in the following example.

Code 14-3 JMenuItem Added to a Menu

```
28   mi1 = new JMenuItem("New");
29   mi2 = new JMenuItem("Save");
30   mi3 = new JMenuItem("Load");
```

```
31  mi4 = new JMenuItem("Quit");
32  mi1.addActionListener(this);
33  mi2.addActionListener(this);
34  mi3.addActionListener(this);
35  mi4.addActionListener(this);
36  m1.add(mi1);
37  m1.add(mi2);
38  m1.add(mi3);
39  m1.addSeparator();
40  m1.add(mi4);
```

This code produces the following:

Figure 14-3 The `JMenuItem` Component

Usually, you add an `ActionListener` to a `JMenuItem` object to provide behavior for the menus.

Creating a `JCheckboxMenuItem`

The `JCheckboxMenuItem` is a checkable menu item, so you can have selections (*on* or *off* choices) listed in menus, as shown in the following example.

Demonstration – `JCheckboxMenuItem`

```
19  f = new JFrame("CheckboxMenuItem");
20  mb = new JMenuBar();
21  m1 = new JMenu("File");
22  m2 = new JMenu("Edit");
23  m3 = new JMenu("Help");
24  mb.add(m1);
25  mb.add(m2);
26  mb.add(m3);
27  f.setJMenuBar(mb);
```

```
...........
43   mi5 = new JCheckBoxMenuItem("Persistent");
44   mi5.addItemListener(this);
45   m1.add(mi5);
```

The code produces the following:

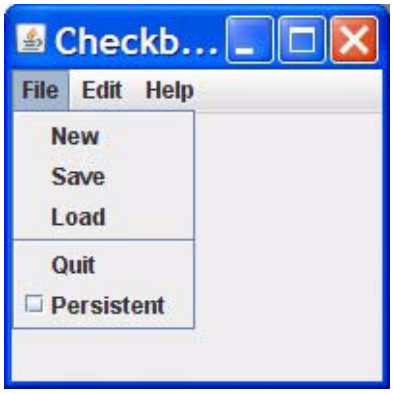

Figure 14-4 The `JCheckboxMenuItem` Component

You should monitor the `JCheckboxMenuItem` using the `ItemListener` interface.

Controlling Visual Aspects

You can control the colors used for the foreground and the background of AWT components.

Colors

You use two methods to set the colors of a component:

```
setForeground()
setBackground()
```

Both of these methods take an argument that is an instance of the `java.awt.Color` class. You can use constant colors referred to as `Color.red`, `Color.blue`, and so on. The full range of predefined colors is listed in the documentation page for the `Color` class.

You can also construct a specific `Color` object by specifying the color by a combination of three byte-sized integers (0–255), one for each primary color: red, blue, and green. For example:

```
Color purple = new Color(255, 0, 255);
JButton b = new JButton("Purple");
b.setBackground(purple);
```