CSC 205 Lab 8 : GUI Programming with the AWT

Goals

After completing this lab, you should be able to:

- Write GUI applications using AWT components.
- Know how to implement action listeners to handle events.

Lab Startup

Change into your `Labs` directory, and let's create and change into a `Lab8` directory. Now, let's copy over some files by typing : `cp /pub/digh/CSC205/Lab8/*` .

My First Interactive GUI

1. Compile and execute the GUI application (`MyGUI.java`). Play with it and make sure that the button and text field will not respond to mouse clicks. Check the source file, and determine which Java statements are being used to generate the GUI layout. Which layout manager class is being used? How does it set up the components within the window? What lines of code are needed to set it up?

setLayout
(new FlowLayout())

2. Remove the comment signs to bring the code segments for the `ButtonListener` and `WindowCloser` classes to life. Make additional changes as follows:

   a. Create a `ButtonListener` object and attach it to the button `b`. You can use one line of code as follows.

   ```
   b.addActionListener(new ButtonListener());
   ```

   b. Create a `WindowCloser` object and attach it to the `MyGUIFrame`. You can use the `addWindowListener()` method of the Window class, which is the direct superclass of the `Frame` class.

   ```
   addWindowListener(new WindowCloser());
   ```

   c. What types of components are going to be added by the `MyGUIFrame` constructor?

   Action Listener Component

   d. What steps are being performed by the `ButtonListener` object?

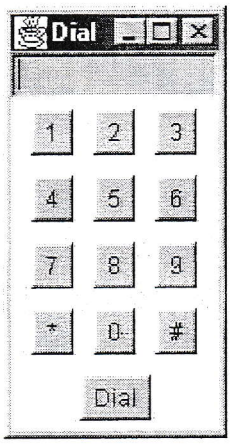
   Close frame and clear string

3. Compile and execute the GUI application again. Test the enter button and the close window button (X) on the frame. Verify your answers from above.

4. Replace the `f.setSize()` method in the driver class (`MyGUI`) with the `f.pack()` method. Compile and execute the GUI application again. Do you notice any differences from the previous frame? Write down your explanation below.

Yes, for f.pack() the "Enter" button and text field was right next to each other, while f.setSize(), the "Enter" button was below the text field

A Simple Phone Dial Pad



We've learned how to add `Component` objects to a `Container` such as a `Frame`. In addition, we've learned how to add an `ActionListener` to catch and respond to events. Now, let's play with a `Frame` with more buttons to simulate a phone dial pad. This frame contains a text field, twelve buttons (corresponding to the twelve buttons on a Touchtone phone), and a "Dial" button. When the user presses a numbered button, the corresponding digit will be added to the text field, so that the number being dialed is gradually spelled out. (Pressing the "*" and "#" buttons has no effect.) When the user presses the "Dial" button, the text field will display the message "Dialing..." for two seconds. The text field will then go blank, and the user can enter another phone number.

1. Take a look at the source code for `Dial.java`. Answer the following three questions.
   a. What `LayoutManager` is used in the `buttonPanel` to make sure the numbered keys are displayed in a 4 by 3 mesh?

      Grid Layout

   b. A `for` loop was used to generate and add keys 1 through 9 into the pad. Why were the numbered keys displayed row-wise, instead of column-wise?

      We don't have column, so row by row instanstiation

   c. What `LayoutManager` is used to put the `phoneNumber`, `centerPanel`, and `bottomPanel` into desired places?

      Border Layout

2. To make the pad work, complete the following eight exercises. Each is labeled within your source code (e.g., ** 1**).
   a. Create a `DigitListener` object for all numbered keys (i.e., right after the line with `**1**`.) That is, `DigitListener dl = new DigitListener();` Why is it possible that only one listener is enough for 10 different digits?

   b. Add the listener reference created in part a. to each button.
      `buttonPanel.add(b);`

   c. Add the listener reference for the button '0'. `b.addActionListener(dl);`

   d. Create and add a `DialListener` object to the dial button.
      `b.addActionListener(new DialListener());`

   e. Implement the `DigitListener` class to modify the phone number when a new digit is pressed. Include the previous numerical string with the current button label.
      `phoneNumber.setText(phoneNumber.getText()+buttonLabel);`

   f. Display the "`Dialing...`" message.
      `phoneNumber.setText("Dialing...");`