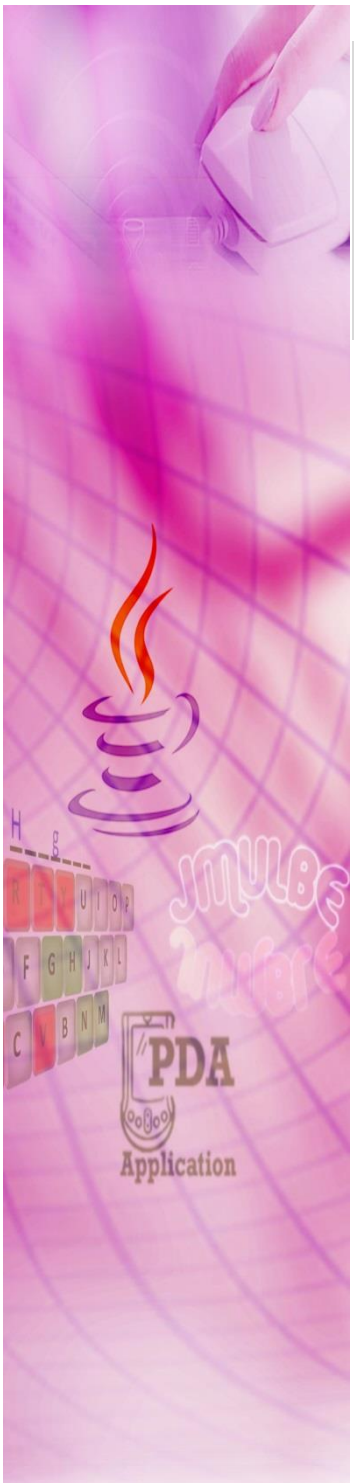


Instructor Inputs

Session 10

Session Overview

This session covers Chapter 7 of the book, Introduction to Java – SG. In this session, the students will learn to work with the various Swing components. In addition, the session will describe how to lay the components on the frame or on the window with the help of layout managers.

Exploring UI Components

Handling Tips

Start the discussion by asking the students what they understand by the term, User Interface (UI). Elicit the responses, and then start explaining the importance of UI with respect to the application development. Thereafter, explain the students that there are mainly two types of UI, Character User Interface (CUI) and Graphical User Interface (GUI). Further, explain the difference between both the interfaces with the help of the example of the Classic Jumble Word game.

Next, explain the students the Swing class hierarchy with the help of the figure given in the SG for the menu of the Classic Jumble Word game. Thereafter, discuss the components of each category one by one with the help of figures of the respective components. Further, inform the students about some more classes of the Swing class hierarchy with the information given in the Additional Inputs topic.

Thereafter, discuss all the listed containers and components in the SG with their constructors and most commonly used methods. In addition, explain the syntax and code snippet to create the same. Further, demonstrate the example of `JTable` given in the Additional Examples topic.

Additional Inputs

The following list describes the UI components of the Swing hierarchy:

- `JColorChooser`: Provides a color pane that enables a user to select the color.
- `JProgressBar`: Provides a progress bar that graphically displays the progress of the task.
- `JTable`: Provides a table structure that is used to display a two-dimensional table of cells.
- `JToolBar`: Provides a tool bar that is used to display the most commonly used actions in the form of commands.

Additional Examples

Example of JTable:

```
import javax.swing.JFrame;
import javax.swing.JTable;

public class TableDemo extends JFrame
{
    JTable employee;
    String[] columnNames = {"ID", "First Name", "Last Name", "Department"};
    Object[][] data = {{"T101", "Peter", "Parker", "Admin"}, {"T102",
"Kathy", "White", "IT"}, {"T103", "John", "Smith", "Accounts"}};
```

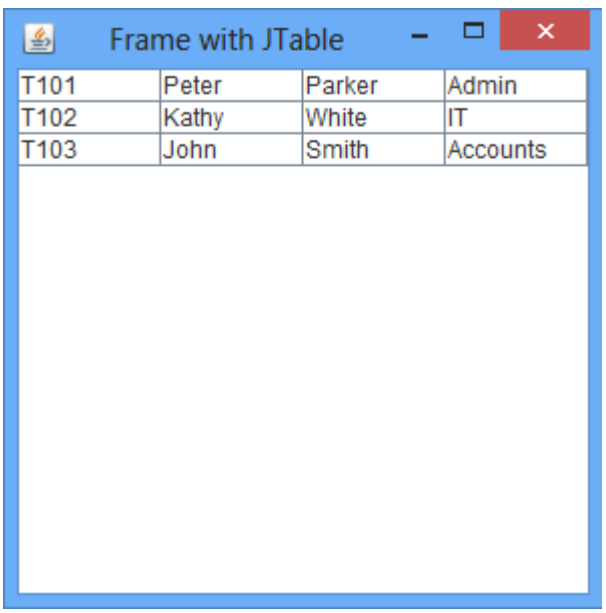
```
        public TableDemo()
        {
            employee = new JTable(data, columnNames);

            setTitle("Frame with JTable");
            setSize(300,300);
            setVisible(true);

            add(employee);
        }
        public static void main(String[] args)
        {
             TableDemo obj = new TableDemo();
        }

    }
```

The output of the preceding code is:



Managing Layouts

Start the discussion by explaining the importance of layouts in the GUI application to enhance the look and feel of the interface. Thereafter, tell the students that Java provides various layouts, and then explain the layouts given in the SG. Further, explain every layout along with the code snippet given in the SG.

Next, while explaining `FlowLayout`, discuss the alignment properties for the controls on the container with the information given in the Additional Inputs topic.

Thereafter, while explaining `GridBagLayout`, discuss the instance variable of the `GridBagConstraints` class with the information given in the Additional Inputs topic.

Additional Inputs

`FlowLayout` arranges controls in a container horizontally until no more controls fit on the same line. However, the alignment of the line is determined by the `align` property. `FlowLayout` provides the following alignment properties:

- `LEFT`: Specifies that each row of component should be left justified.
- `RIGHT`: Specifies that each row of component should be right justified.
- `CENTER`: Specifies that each row of component should be centered.
- `LEADING`: Specifies that each row of component should be justified to the leading edge of the container's orientation. For example, to the left in case of left to right orientation.
- `TRAILING`: Specifies that each row of component should be justified to the trailing edge of the container's orientation. For example, to the right in case of left to right orientation.

The following table lists the instance variables of the `GridBagConstraints` class.

Instance Variable	***Description***
`Ipadx`	*Is used to specify the additional width of the component.*
`Ipady`	*Is used to specify the additional height of the component.*
`Insets`	*Is used to specify the minimum space between the component and the edges of its display area.*

The Instance Variables of the GridBagConstraints Class

Activity 7.1: Managing Layouts

Handling Tips

Discuss the problem statement with the students.

The solution files, **Hangman.java**, **Menu.java**, and **GameWindow.java**, for this activity are provided at the following location in the TIRM CD:

- **Datafiles For Faculty\Activities\Chapter 07\Activity 7.1\Solution**

FAQs

- *Can a layout be assigned to any component?*

  Ans: Yes, layouts can be assigned to any component. However, it is best to use only with the container component.

- *What will happen if we try to add more than one component in a particular region by using the* `BorderLayout` *manager?*

  Ans: Only a single component can be added in each regions of the `BorderLayout` manager. If you try to add more than one component, only the last component added will be visible in that region.

- *How can we identify the layout that is already applied to the container?*

  Ans: You can identify the layout by using the `getLayout()` method of that container.

- *How the* `JFrame` *container lays the components by default?*

  Ans: The `JFrame` container lays the components according to the `BorderLayout` manager.