**Experiment No-2**

**Title:** Write a program to demonstrate different layouts.

**Aim:** Able to apply different layouts to Applet and frame.

Able to demonstrate the use of different types of Layout manager.

**Theory:**

### **Java Layout Managers**

The LayoutManagers are used to arrange components in a particular manner. The **Java LayoutManagers** facilitates us to control the positioning and size of the components in GUI forms. LayoutManager is an interface that is implemented by all the classes of layout managers. There are the following classes that represent the layout managers:

1. java.awt.BorderLayout
2. java.awt.FlowLayout
3. java.awt.GridLayout
4. java.awt.CardLayout
5. java.awt.GridBagLayout

### **Java BorderLayout**

The BorderLayout is used to arrange the components in five regions: north, south, east, west, and center. Each region (area) may contain one component only. It is the default layout of a frame or window. The BorderLayout provides five constants for each region:

1. **public static final int NORTH**
2. **public static final int SOUTH**
3. **public static final int EAST**
4. **public static final int WEST**
5. **public static final int CENTER**

### **Constructors of BorderLayout class:**

* **BorderLayout():**creates a border layout but with no gaps between the components.
* **BorderLayout(int hgap, int vgap):**creates a border layout with the given horizontal and vertical gaps between the components.

# **Java GridLayout**

The Java GridLayout class is used to arrange the components in a rectangular grid. One component is displayed in each rectangle.

### **Constructors of GridLayout class**

1. **GridLayout():** creates a grid layout with one column per component in a row.
2. **GridLayout(int rows, int columns):** creates a grid layout with the given rows and columns but no gaps between the components.
3. **GridLayout(int rows, int columns, int hgap, int vgap):** creates a grid layout with the given rows and columns along with given horizontal and vertical gaps.

# **Java FlowLayout**

The Java FlowLayout class is used to arrange the components in a line, one after another (in a flow). It is the default layout of the applet or panel.

### **Fields of FlowLayout class**

1. **public static final int LEFT**
2. **public static final int RIGHT**
3. **public static final int CENTER**
4. **public static final int LEADING**
5. **public static final int TRAILING**

### **Constructors of FlowLayout class**

1. **FlowLayout():** creates a flow layout with centered alignment and a default 5 unit horizontal and vertical gap.
2. **FlowLayout(int align):** creates a flow layout with the given alignment and a default 5 unit horizontal and vertical gap.
3. **FlowLayout(int align, int hgap, int vgap):** creates a flow layout with the given alignment and the given horizontal and vertical gap.

# **Java CardLayout**

The **Java CardLayout** class manages the components in such a manner that only one component is visible at a time. It treats each component as a card that is why it is known as CardLayout.

### **Constructors of CardLayout Class**

1. **CardLayout():** creates a card layout with zero horizontal and vertical gap.
2. **CardLayout(int hgap, int vgap):** creates a card layout with the given horizontal and vertical gap.

### **Commonly Used Methods of CardLayout Class**

* **public void next(Container parent):** is used to flip to the next card of the given container.
* **public void previous(Container parent):** is used to flip to the previous card of the given container.
* **public void first(Container parent):** is used to flip to the first card of the given container.
* **public void last(Container parent):** is used to flip to the last card of the given container.
* **public void show(Container parent, String name):** is used to flip to the specified card with the given name.

# **Java GridBagLayout**

The Java GridBagLayout class is used to align components vertically, horizontally or along their baseline.

The components may not be of the same size. Each GridBagLayout object maintains a dynamic, rectangular grid of cells. Each component occupies one or more cells known as its display area. Each component associates an instance of GridBagConstraints. With the help of the constraints object, we arrange the component's display area on the grid. The GridBagLayout manages each component's minimum and preferred sizes in order to determine the component's size. GridBagLayout components are also arranged in the rectangular grid but can have many different sizes and can occupy multiple rows or columns.

### **Constructor**

**GridBagLayout():** The parameterless constructor is used to create a grid bag layout manager.

**Conclusion:** In this experiment we have learnt to demonstrate use of different layout managers used in JAVA.

**Exercise:**

