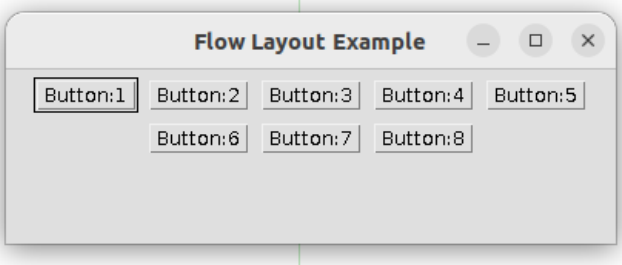
Layout Management

- To add components to a container for e.g. a Frame instance,
  - we need to specify the type of component layout scheme to be used i.e.
    - how components within that container should be arranged.
- Layout managers implements a layout policy that defines the spatial relationship between components in a container.
- Layout managers are provided to arrange GUI components on a container for presentation purposes.
- The layout managers provide basic layout capabilities that are easier to use. This enables the programmer to concentrate on the basic “look and feel” and lets the layout managers process most of the layout details.

- Layout managers deals with:
  - location and size of the components.
  - the way to update the interface when user re-sizes the window.

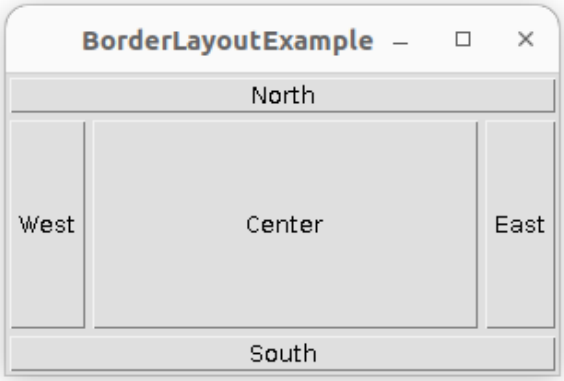
1. Flowlayout

```java
// It arranges the components in a line, one after another (in a flow). It is the default layoutof panel.

import java.awt.*;
public class FlowLayoutExample extends Frame{
    // constructor
    public FlowLayoutExample(String title)
    {
        //It creates  the Frame by calling the constructor of Frame class.
        super(title);

        //Setting up Flow Layout
        setLayout(new FlowLayout());

        //Creating a button and adding it to the frame
        Button b1 = new Button("Button:1");
        add(b1);

        //Adding other components to the Frame
        Button b2 = new Button("Button:2");
        add(b2);

        Button b3 = new Button("Button:3");
        add(b3);

        Button b4 = new Button("Button:4");
        add(b4);

        Button b5 = new Button("Button:5");
        add(b5);

        Button b6 = new Button("Button:6");
        add(b6);

        Button b7 = new Button("Button:7");
        add(b7);

        Button b8 = new Button("Button:8");
        add(b8);

    }
    public static void main(String[] args)
    {
        FlowLayoutExample screen =  new FlowLayoutExample("Flow Layout Example");
        screen.setSize(400,150);//sets the width and height of the frame
        screen.setVisible(true);
    }
}
```



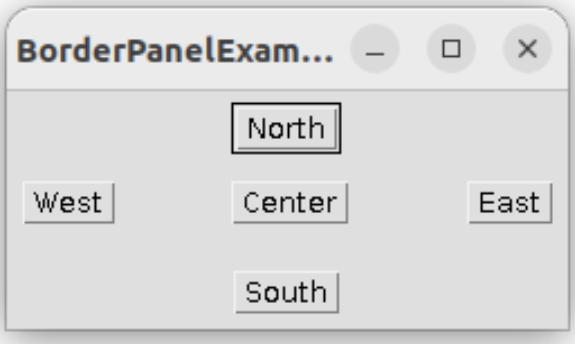
2. BorderLayout

```java
import java.awt.*;
public class BorderLayoutExample extends Frame {
    public BorderLayoutExample(String title) {
      super(title);
      add("North", new Button("North"));
      add("South", new Button("South"));
      add("East", new Button("East"));
      add("West", new Button("West"));
      add("Center", new Button("Center"));
    }
    public static void main(String[] args) {
        BorderLayoutExample ble = new BorderLayoutExample("BorderLayoutExample");
      ble.setSize(300, 200);
      ble.setVisible(true);
    }
}
```



3. BorderLayout with Panel

```
1  import java.awt.*;
2  public class BorderPanelExample extends Frame {
3      public BorderPanelExample(String title) {
4        super(title);
5        addComponent("North", new Button("North"));
6        addComponent("South", new Button("South"));
7        addComponent("East", new Button("East"));
8        addComponent("West", new Button("West"));
9        addComponent("Center", new Button("Center"));
10     }
11     public void addComponent(String region, Component component) {
12       Panel panel = new Panel();
13       panel.add(component);
14       add(region, panel); // make sure you add the panel!
15     }
16     public static void main(String[] args) {
17         BorderPanelExample bpe =
18             new BorderPanelExample("BorderPanelExample");
19     bpe.setSize(200, 150);
20     bpe.setVisible(true);
21     }
22 }
23
24
```

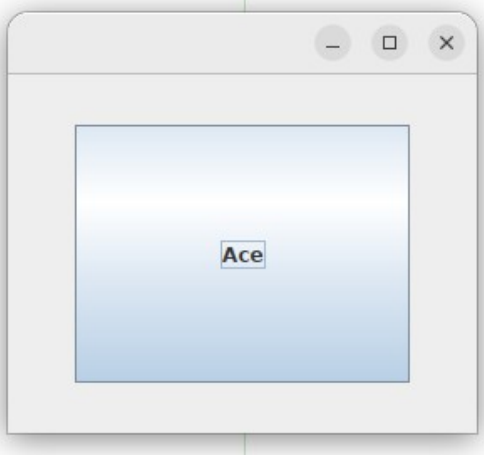


4. GridLayout

```java
//  arrange the components in a rectangular grid/table

import java.awt.*;

public class GridLayoutBasicExample extends Frame {
     public  GridLayoutBasicExample( String title) {
         setLayout(new GridLayout(3,2));
         add(new Button("1"));
         add(new Button("2"));
         add(new Button("3"));
         add(new Button("4"));
         add(new Button("5"));
         add(new Button("6"));
     }

     public static void main(String args[]){

        GridLayoutBasicExample gl = new GridLayoutBasicExample("GridLayout Basic Example");
        gl.setSize(300, 200);
        gl.setVisible(true);
     }
 }
```

5. CardLayout

```java
import java.awt.*;
import java.awt.event.*;

import javax.swing.*;

public class CardLayoutBasicExample extends JFrame implements ActionListener{
    CardLayout card;
    JButton b1,b2,b3,b4;
    Container c;
    CardLayoutBasicExample(){

        c=getContentPane(); // Gets the content pane layer so that we can add elements to it
        card=new CardLayout(40,30);
        //create CardLayout object with 40 hor space and 30 ver space
        c.setLayout(card);

        b1=new JButton("Ace");
        b2=new JButton("King");
        b3=new JButton("Queen");
        b4=new JButton("Jocker");

        b1.addActionListener(this);
        b2.addActionListener(this);
        b3.addActionListener(this);
        b4.addActionListener(this);

        c.add(b1);c.add(b2);c.add(b3); c.add(b4);

    }

    public void actionPerformed(ActionEvent e) {
        card.next(c);
    }

    public static void main(String[] args) {
        CardLayoutBasicExample cl=new CardLayoutBasicExample();
        cl.setSize(500,400);
        cl.setVisible(true);
        cl.setDefaultCloseOperation(EXIT_ON_CLOSE);
    }
}
```



6. GridBagLayout

```java
// can arrange 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.
import java.awt.*;
import javax.swing.*;
public class GridBagBasicExample {

  public static void main(String[] args) {
    JFrame f = new JFrame();
    JPanel p = new JPanel();

    p.setLayout(new GridBagLayout());
    p.add(new JLabel("Java"));
    p.add(new JLabel("Programming"));
    p.add(new JLabel("Year 1"));
    p.add(new JButton("Button1"));
    p.add(new JButton("Button2"));
    p.add(new JTextField("Pokhara Gandaki"));
    p.add(new JTextField("2"));
    p.add(new JTextField("3"));

    f.getContentPane().add(p);
    f.setSize(600, 200);
    f.setVisible(true);
  }
}
```

7. BoxLayout

```java
//  arrange the components in a rectangular grid/table

import java.awt.*;
import javax.swing.*; // for BoxLayout


public class BoxLayoutBasic extends Frame {

     public  BoxLayoutBasic( String title) {

         setLayout (new BoxLayout (this, BoxLayout.X_AXIS));  // Set Box Layout
         // BoxLayout.X_AXIS : horizontally add buttons
        // BoxLayout.Y_AXIS : vertically add buttons

         add(new Button("1"));
         add(new Button("2"));
         add(new Button("3"));
         add(new Button("4"));
         add(new Button("5"));
         add(new Button("6"));
     }

     public static void main(String args[]){

        BoxLayoutBasic blb = new BoxLayoutBasic("BoxLayout Basic");
        blb.setSize(300, 200);
        blb.setVisible(true);
     }
 }
```