**JAVA Lab**

**Lab Experiment No 11**

**Name:** Aamir Ansari **Batch:** A **Roll no:** 01

**Aim**: To write a Java program to illustrate the usage of built in layout manager.

**Problem statement:**

Write a Java program to create a simple calculator using java AWT elements. Use a grid layout to arrange buttons for the digits and basic operation +, -, /, \*. Add a text field to display the results.

**Theory**:

Explain grid layout manager in detail.

Grid Layout class represents a layout manager with a specified number of rows and columns in a rectangular grid. The Grid Layout container is divided into an equal-sized of rectangles, and one of the components is placed in each rectangle. Every rectangle cell has the same size therefore, they contain a component, which fills the entire cell. When the user changes or adjusts the size of the container, the size of each rectangles changes accordingly.

Constructors of the class:

GridLayout(): It Creates a grid layout with a default of one column per component, in a single row.

GridLayout(int rw, int cl): It creates a grid layout with the specified number of rows and columns.

GridLayout(int rw, int cl, int hgap, int vgap): It creates a grid layout with the specified number of rows and columns with horizontal and vertical gap.

Commonly Used Methods:

addLayoutComponent(String str, Component cmp): Adds the specified component with the specified name to the layout.

setColumns(int cl): Sets the number of columns in this layout to the specified value.

layoutContainer(Container pr): Lays out the specified container using this layout.

toString(): Returns the string representation of this grid layout’s values.

How to add buttons and handle the event of clicking the buttons i.e. explain listeners.

To add button in the java we follow the syntax

Button name\_button = new Button(“Label\_on\_button”);

The event listener is the feature of java that handles the several events for the several objects. Classes for helping in implementing event listeners are present in the java.awt.event.\*; package. All the operations are controlled by the events generated by these buttons.

To handle the events generated by these buttons you add action listeners

e.g. button\_name.addActionListener(this);.

When the action event occurs, that object's actionPerformed method is invoked.

actionPerformed(ActionEvent e)

{

//code

}

**Program:**

import java.awt.\*;

import java.awt.event.\*;

public class calculator implements ActionListener

{

int c,n;

String s1,s2,s3,s4,s5;

Frame f;

Button b1,b2,b3,b4,b5,b6,b7,b8,b9,b10,b11,b12,b13,b14,b15,b16,b17;

Panel p;

TextField tf;

GridLayout g;

calculator()

{

f = new Frame("My calculator");

f.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

f.setLayout(new FlowLayout());

p = new Panel();

//Assigning buttons

b1 = new Button("0");

b1.addActionListener(this);

b2 = new Button("1");

b2.addActionListener(this);

b3 = new Button("2");

b3.addActionListener(this);

b4 = new Button("3");

b4.addActionListener(this);

b5 = new Button("4");

b5.addActionListener(this);

b6 = new Button("5");

b6.addActionListener(this);

b7 = new Button("6");

b7.addActionListener(this);

b8 = new Button("7");

b8.addActionListener(this);

b9 = new Button("8");

b9.addActionListener(this);

b10 = new Button("9");

b10.addActionListener(this);

b11 = new Button("+");

b11.addActionListener(this);

b12 = new Button("-");

b12.addActionListener(this);

b13 = new Button("\*");

b13.addActionListener(this);

b14 = new Button("/");

b14.addActionListener(this);

b15 = new Button("=");

b15.addActionListener(this);

b16 = new Button("C");

b16.addActionListener(this);

//Text field to display

tf = new TextField(20);

f.add(tf);

//Setting the layout

g = new GridLayout(4,4,10,20);

p.setLayout(g);

//Adding buttons to it

p.add(b1);p.add(b2);p.add(b3);p.add(b4);p.add(b5);p.add(b6);p.add(b7);p.add(b8);p.add(b9);

p.add(b10);p.add(b11);p.add(b12);p.add(b13);p.add(b14);p.add(b15);p.add(b16);

f.add(p); f.setSize(300,300); f.setVisible(true);

}

public void actionPerformed(ActionEvent e)

{

//Performing calculations

if(e.getSource()==b1)

{

s3 = tf.getText();

s4 = "0";

s5 = s3+s4;

tf.setText(s5);

}

if(e.getSource()==b2)

{

s3 = tf.getText();

s4 = "1";

s5 = s3+s4;

tf.setText(s5);

}

if(e.getSource()==b3)

{

s3 = tf.getText();

s4 = "2";

s5 = s3+s4;

tf.setText(s5);

}

if(e.getSource()==b4)

{

s3 = tf.getText();

s4 = "3";

s5 = s3+s4;

tf.setText(s5);

}

if(e.getSource()==b5)

{

s3 = tf.getText();

s4 = "4";

s5 = s3+s4;

tf.setText(s5);

}

if(e.getSource()==b6)

{

s3 = tf.getText();

s4 = "5";

s5 = s3+s4;

tf.setText(s5);

}

if(e.getSource()==b7)

{

s3 = tf.getText();

s4 = "6";

s5 = s3+s4;

tf.setText(s5);

}

if(e.getSource()==b8)

{

s3 = tf.getText();

s4 = "7";

s5 = s3+s4;

tf.setText(s5);

}

if(e.getSource()==b9)

{

s3 = tf.getText();

s4 = "8";

s5 = s3+s4;

tf.setText(s5);

}

if(e.getSource()==b10)

{

s3 = tf.getText();

s4 = "9";

s5 = s3+s4;

tf.setText(s5);

}

if(e.getSource()==b11)

{

s1 = tf.getText();

tf.setText("");

c=1;

}

if(e.getSource()==b12)

{

s1 = tf.getText();

tf.setText("");

c=2;

}

if(e.getSource()==b13)

{

s1 = tf.getText();

tf.setText("");

c=3;

}

if(e.getSource()==b14)

{

s1 = tf.getText();

tf.setText("");

c=4;

}

if(e.getSource()==b15)

{

s2 = tf.getText();

if(c==1)

{

n = Integer.parseInt(s1)+Integer.parseInt(s2);

tf.setText(String.valueOf(n));

}

else

if(c==2)

{

n = Integer.parseInt(s1)-Integer.parseInt(s2);

tf.setText(String.valueOf(n));

}

else

if(c==3)

{

n = Integer.parseInt(s1)\*Integer.parseInt(s2);

tf.setText(String.valueOf(n));

}

if(c==4)

{

try

{

int p=Integer.parseInt(s2);

if(p!=0)

{

n = Integer.parseInt(s1)/Integer.parseInt(s2);

tf.setText(String.valueOf(n));

}

else

tf.setText("infinite");

}

catch(Exception i){}

}

if(c==5)

{

n = Integer.parseInt(s1)%Integer.parseInt(s2);

tf.setText(String.valueOf(n));

}

}

if(e.getSource()==b16)

{

tf.setText("");

}

}

public static void main(String[] abc)

{

calculator v = new calculator();

}

}

**Output:**







