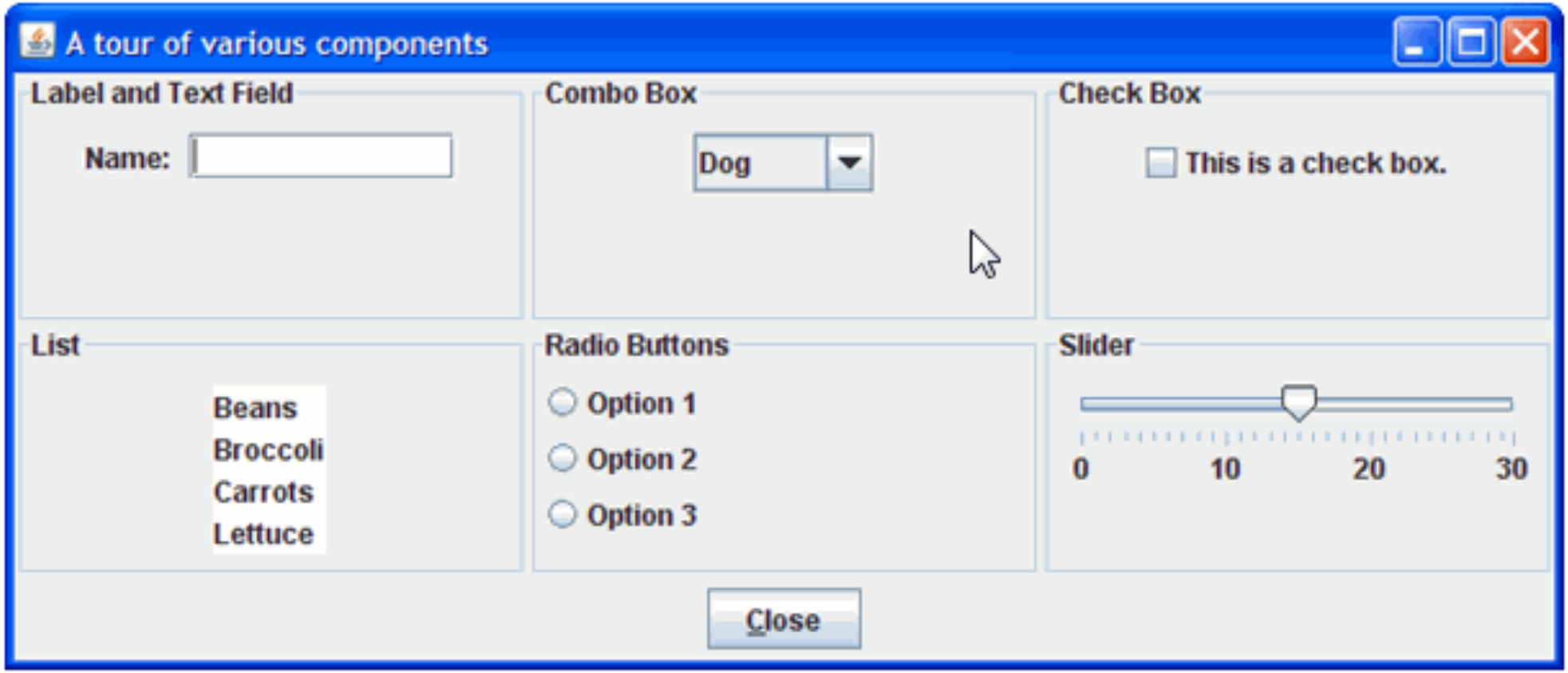
# Advanced Programming
# (CPIT 305)

## 2-Graphical User Interface (GUI)

# GUI Components in Java

- There are two types of GUI components in Java
  - AWT
    - Frame
    - Button
  - Swing
    - JFrame
    - JButton
- Both these types have almost all the components.
- We can use both and for this course we will use swing components.
- Swing components have better performance and work well from one system to another.



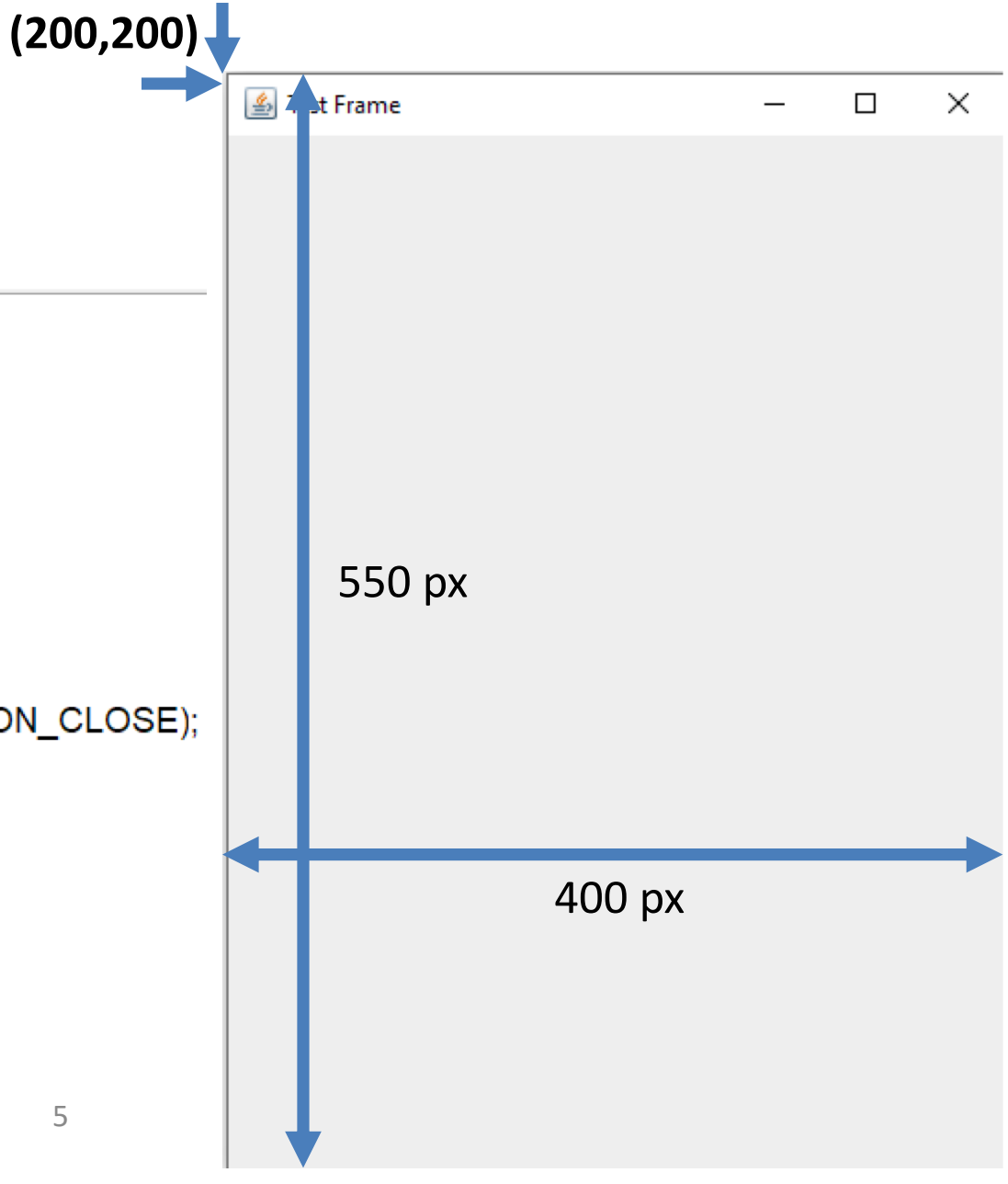

Frame or JFrame

# Frames

- Frame is a window that is not contained inside another window. Frame is the basis to contain other user interface components in Java GUI applications.
- For Swing GUI programs, use JFrame class to create windows.

# Creating Frames by Making JFrame Object

```
1  import javax.swing.JFrame;
2
3  public class MyFrame
4  {
5     public static void main(String []args)
6     {
7        JFrame frame = new JFrame("Test Frame");
8        frame.setBounds(200,200,400,550);
9        frame.setVisible(true);
10
11       frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
12    }
13 }
```

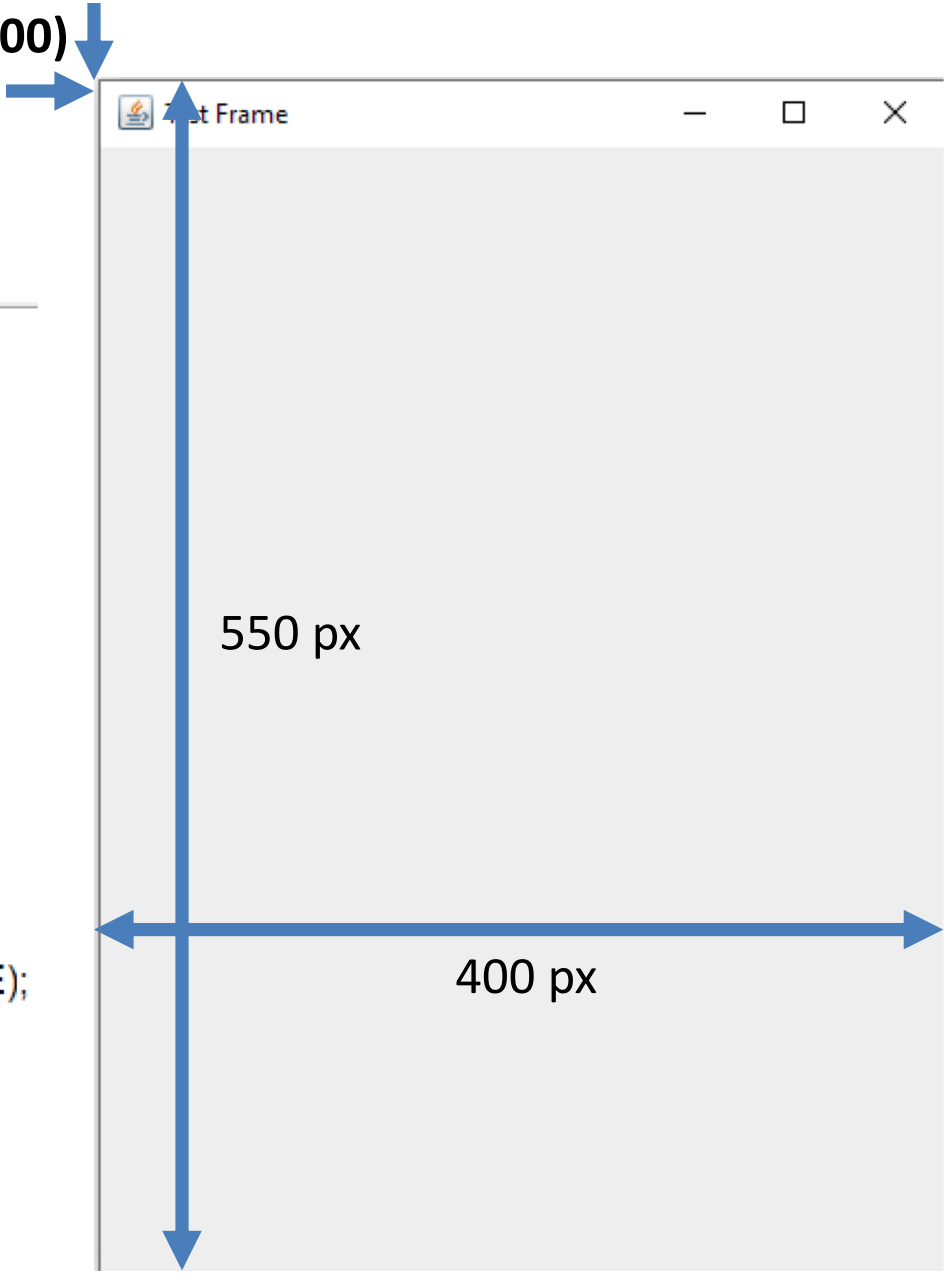


# Description of the Program

- JFrame is a built-in class in Java
- For every class we need to use,
  - We need to include the corresponding import statement. An import statement tells java where to find this class which we are using.
  - We need to make its object and call its functions.
- The value in JFrame("Test Frame") is what is passed to the constructor to initialize something commonly used. In this case this is the title of the frame.
- setBounds(x, y, Width, Height) is used to set the position and size of the frame. All these values are in pixels.
- In Java, y increases from top to bottom while x increases from left to right. That means the top left corner of the screen is (0,0).
- We are calling all these functions using object of class JFrame.
- setVisible(true) makes the frame visible.
- setDefaultCloseOperation() terminates the program when we close the frame.

# Creating Frames Using Inheritance

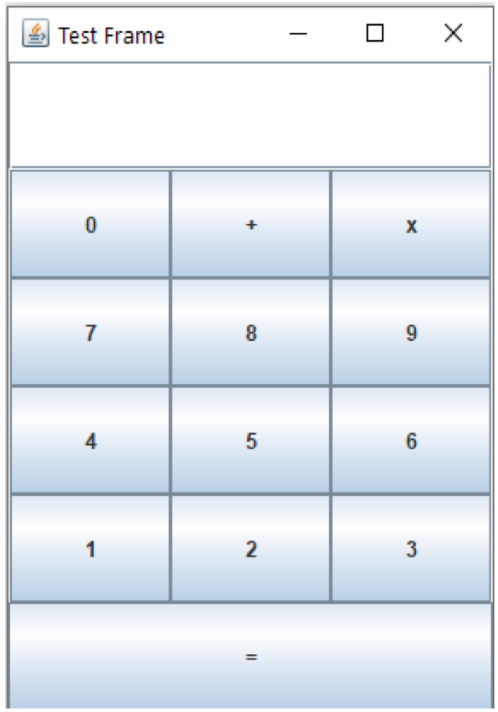
```
1  import javax.swing.JFrame;
2
3  public class MyFrame extends JFrame
4  {
5     public static void main(String []args)
6     {
7        MyFrame frame = new MyFrame("Test Frame");
8     }
9
10    MyFrame(String title)
11    {
12       setTitle(title);
13       setBounds(200,200,400,550);
14       setVisible(true);
15       setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
16    }
17 }
```



# Description of the Program

- If we inherit a class AA from another class BB, class AA can use the functions and data of class BB.
  - In java we use the word extends to inherit from a class.
- Using this same principle we inherit our class from JFrame.
- We make an object of our class in main function.
  - This will be like making an object of JFrame as now our class inherits from JFrame
- We set the properties of this frame inside the constructor of MyFrame.
  - setTitle
  - setBounds
  - setVisible
  - setDefaultCloseOperation

# Adding User Interface Components to the Frame

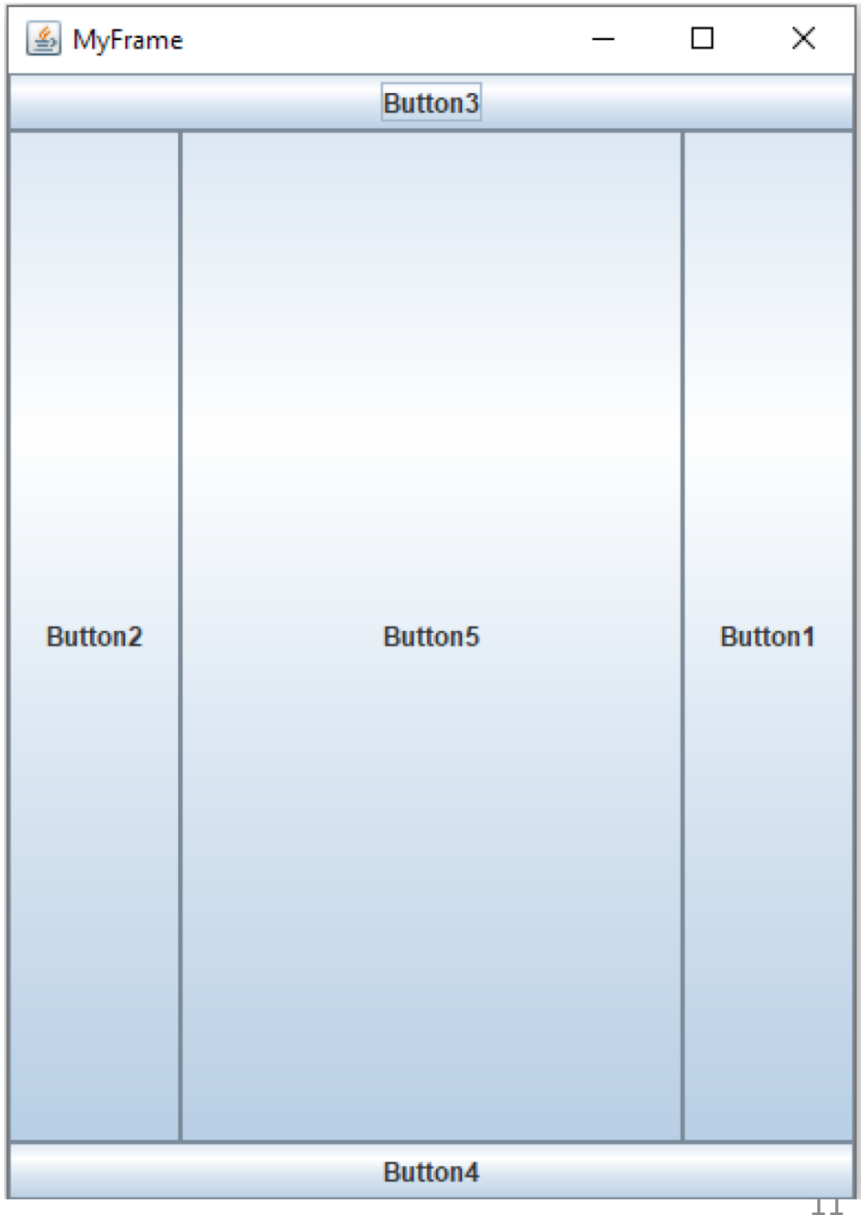


# Where will the new Components be Placed? What is their size?

- Depends on the chosen layout manager
- A layout manager describes where the components will be placed and what is their size.
- There are 4 layout managers in Java
  - Border Layout (This is the default layout for Frames)
  - Flow Layout
  - Grid Layout
  - Null Layout

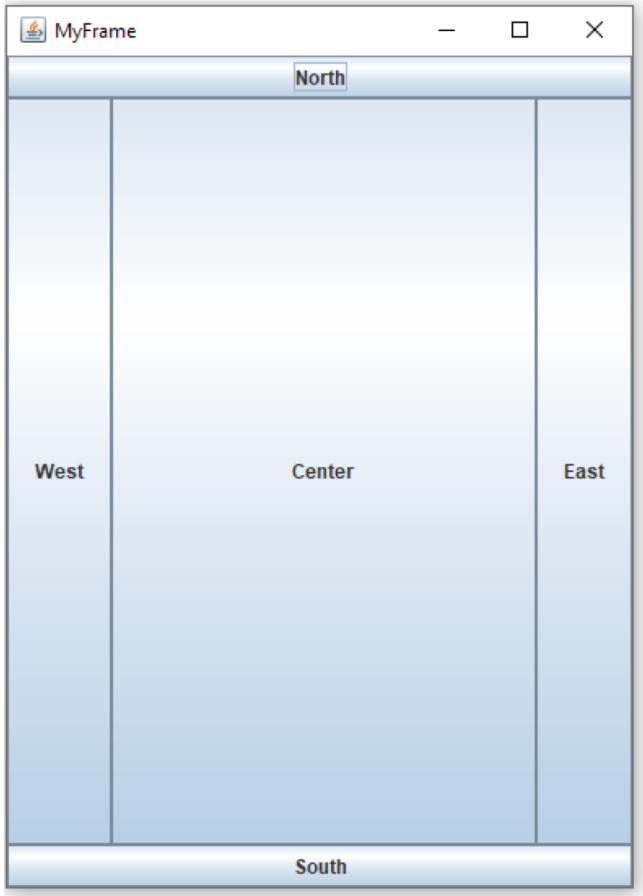
# Border Layout

```
1  import javax.swing.JFrame;
2  import javax.swing.JButton;
3
4  public class MyFrame extends JFrame
5  {
6     public static void main(String []args)
7     {
8        MyFrame frame= new MyFrame("MyFrame");
9     }
10
11    MyFrame(String title)
12    {
13       JButton b1=new JButton("Button1");
14       JButton b2=new JButton("Button2");
15       JButton b3=new JButton("Button3");
16       JButton b4=new JButton("Button4");
17       JButton b5=new JButton("Button5");
18
19       add(b1, "East");
20       add(b2, "West");
21       add(b3, "North");
22       add(b4, "South");
23       add(b5, "Center");
24
25
26
27       setTitle(title);
28       setBounds(200,200,400,550);
29       setVisible(true);
30       setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
31    }
32 }
```



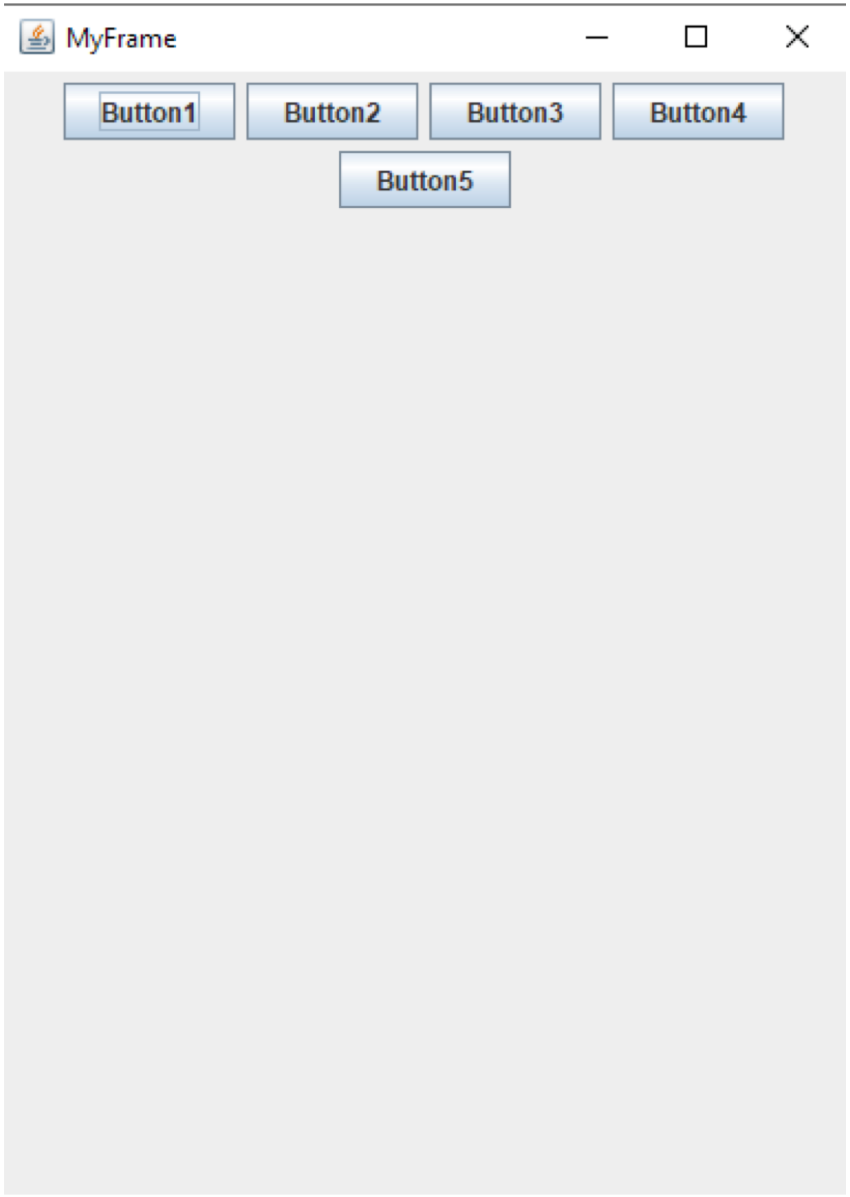
# Description

- Border Layout Manager divides the frame into 5 sections, East, West, North, South, Center.
- This is default layout manager for JFrame.
- To add a component we
  - Make an object for the component to add. For example,
    - JButton b=new JButton("OK");
  - Add the component to the Frame specifying its location
    - add(b, "North");
- The size of the component is automatically determined by the layout manager



# Flow Layout

```
1  import javax.swing.JFrame;
2  import javax.swing.JButton;
3  import java.awt.FlowLayout;
4
5  public class MyFrame extends JFrame
6  {
7     public static void main(String []args)
8     {
9        MyFrame frame= new MyFrame("MyFrame");
10    }
11
12    MyFrame(String title)
13    {
14       FlowLayout fl=new FlowLayout();
15       setLayout(fl);
16
17       JButton b1=new JButton("Button1");
18       JButton b2=new JButton("Button2");
19       JButton b3=new JButton("Button3");
20       JButton b4=new JButton("Button4");
21       JButton b5=new JButton("Button5");
22
23       add(b1);
24       add(b2);
25       add(b3);
26       add(b4);
27       add(b5);
28
29       setTitle(title);
30       setBounds(200,200,400,550);
31       setVisible(true);
32       setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
33    }
34 }
```

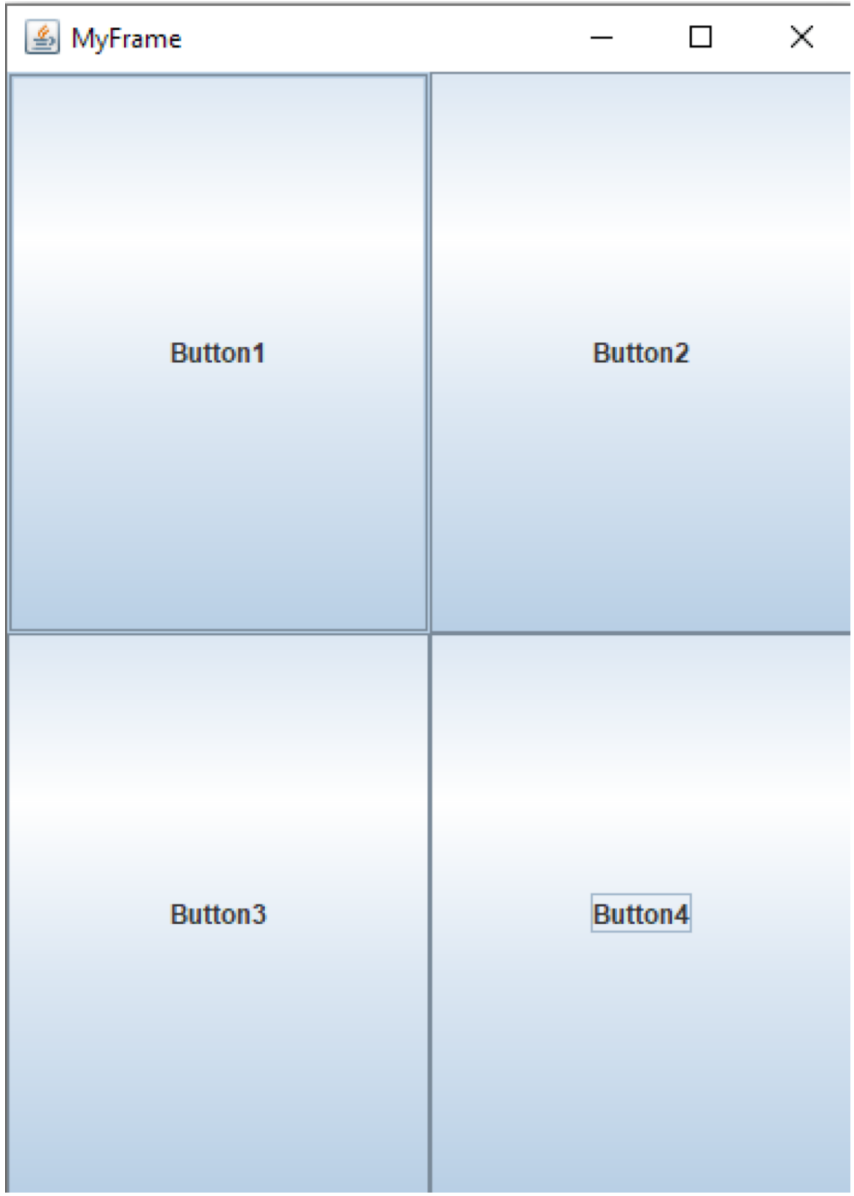


# Description

- Because FlowLayout is not the default layout for JFrame, we need to change it first.
- Line 14 and 15 change the layout of the frame to “FlowLayout”.
  - Line 14 makes a new object of FlowLayout
  - Line 15 sets the layout for the frame to this object
- Line 3 is required if we are using FlowLayout.
- To add a component we
  - Make an object for the component to add. For example,
    - JButton b=new JButton(“OK”);
  - Add the component to the Frame
    - add(b);
- The size and position of the component is automatically determined by the layout manager

# Grid Layout (2x2)

```java
import javax.swing.JFrame;
import javax.swing.JButton;
import java.awt.GridLayout;

public class MyFrame extends JFrame
{
   public static void main(String []args)
   {
      MyFrame frame= new MyFrame("MyFrame");
   }

   MyFrame(String title)
   {
      GridLayout gl=new GridLayout(2,2); //(rows, columns)
      setLayout(gl);

      JButton b1=new JButton("Button1");
      JButton b2=new JButton("Button2");
      JButton b3=new JButton("Button3");
      JButton b4=new JButton("Button4");

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

      setTitle(title);
      setBounds(200,200,400,550);
      setVisible(true);
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   }
}
```



# Description

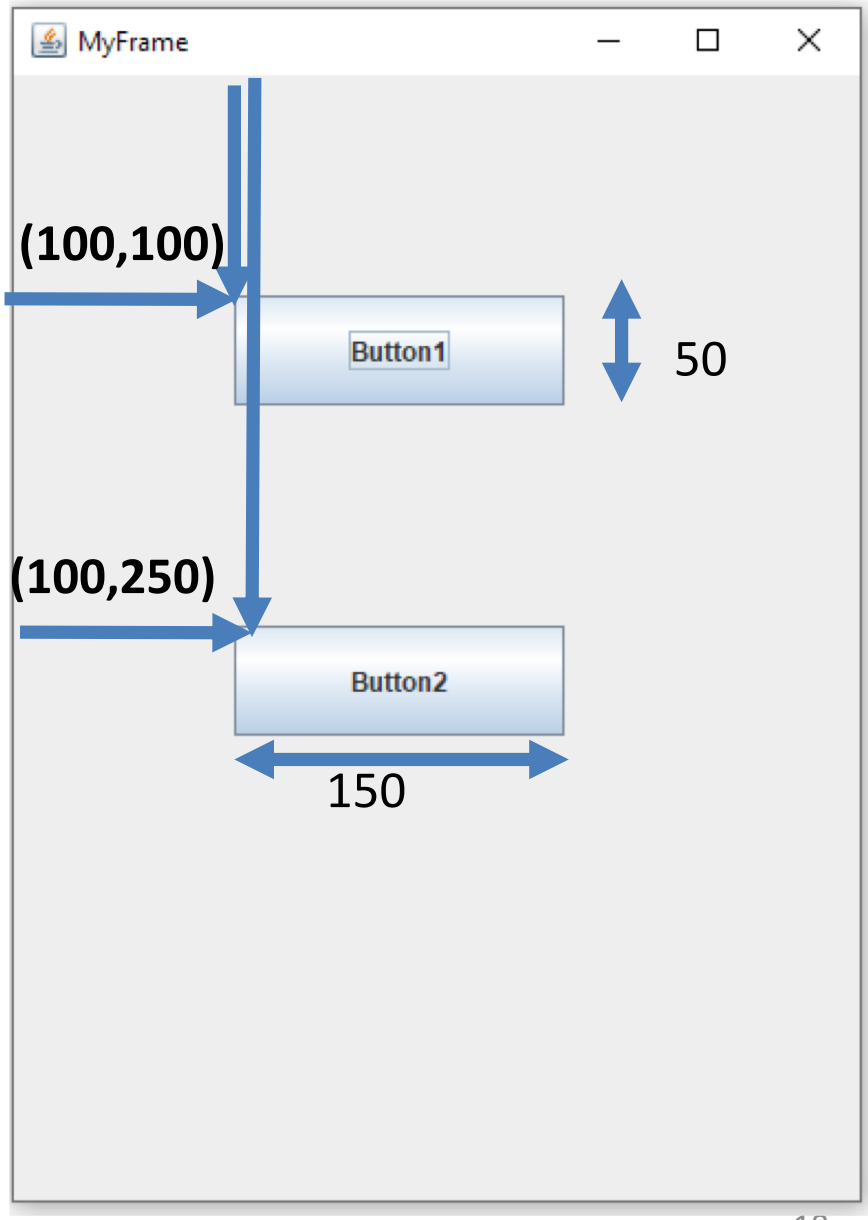
- Because GridLayout is not the default layout for JFrame, we need to change it first.
- Line 14 and 15 change the layout of the frame to GridLayout.
  - Line 14 makes a new GridLayout object with 2 rows and 2 columns. (rows, columns)
  - Line 15 sets the layout of the frame to this object
- Line 3 is required when using GridLayout
- To add a component we
  - Make an object for the component to add. For example,
    - JButton b=new JButton("OK");
  - Add the component to the Frame
    - add(b);
- The size of the component is automatically determined by the layout manager
- The components will be placed in the same order as we add them

# Null Layout Manager

- If using Null Layout manager, we can specify both the position and the size of the component and can place it wherever we like.
- Adding a component now is a 3 step process
  - Make an object for the component to add
    - JButton b=new JButton("OK");
  - Set its position and size using **setBounds(x,y,width,height)**
    - b.setBounds(200,150,100,40);
  - Add the component to the Frame using
    - add(b);

# Null Layout Example

```
1  import javax.swing.JFrame;
2  import javax.swing.JButton;
3
4  public class MyFrame extends JFrame
5  {
6     public static void main(String []args)
7     {
8        MyFrame frame= new MyFrame("MyFrame");
9     }
10
11    MyFrame(String title)
12    {
13       setLayout(null);
14
15       JButton b1=new JButton("Button1");
16       b1.setBounds(100,100,150,50);
17       add(b1);
18
19       JButton b2=new JButton("Button2");
20       b2.setBounds(100,250,150,50);
21       add(b2);
22
23       setTitle(title);
24       setBounds(200,200,400,550);
25       setVisible(true);
26       setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
27    }
28 }
```



# Description

- Line 13 sets the layout of the frame to null.
- Now we need to set the position and size of the components as well.
- Line 16 sets the position of b1 to (100,100), width to 150 and height to 50.
- Line 20 sets the position of b2 to (100,250), width to 150 and height to 50.
- If we want to place the second button below the first, we use same x but bigger y.

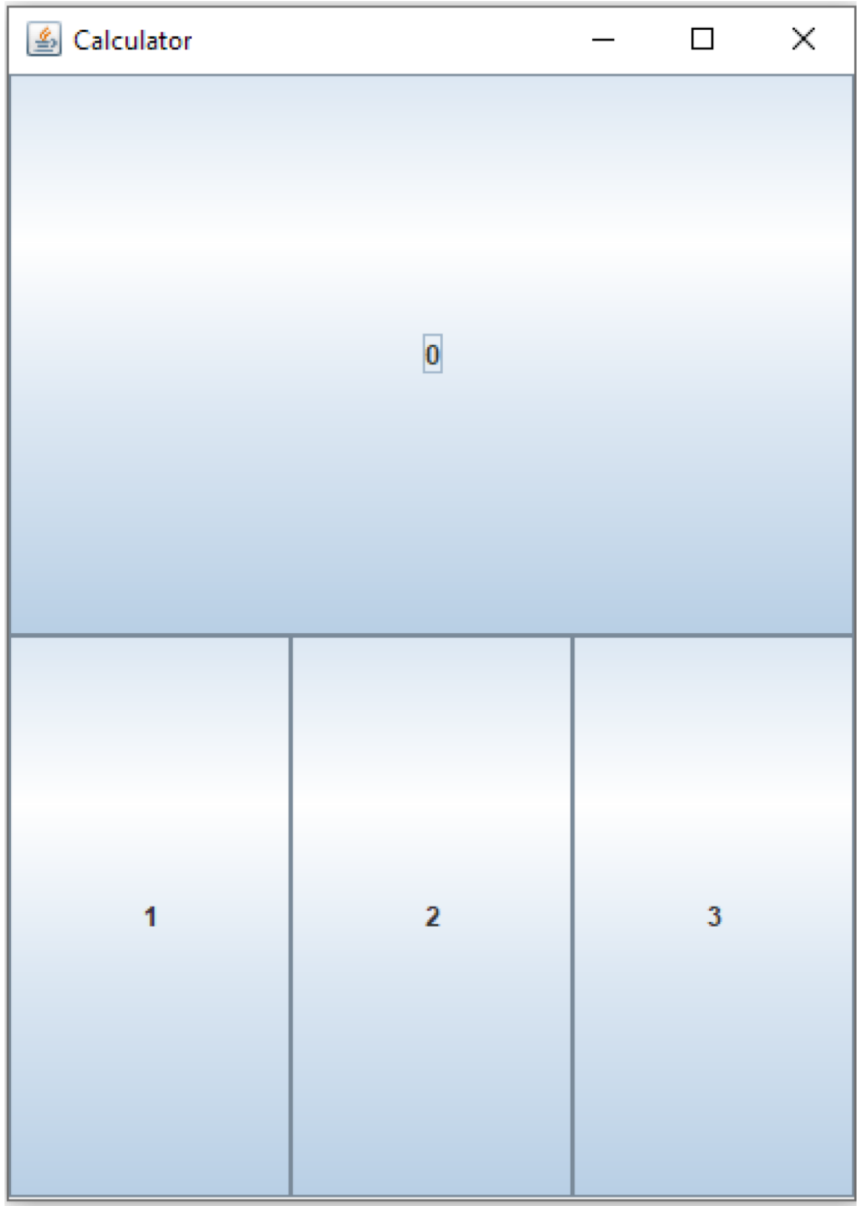
# Which Layout Manager to Use?

- Border, Flow and Grid Layout managers provide size and positioning of the elements relative to the size of the frame.
- Null Layout Manager fixes the size and the position which may not be very useful.
- Normally we use a combination of Border, Flow and Grid Layout managers.

# Combining Multiple Layout Managers

- In the example on the right, the interface is divided into 11 rows.
  - Grid Layout with 1 column and 11 rows
- Different rows have different number of components
  - This row (cell) has Grid Layout with 1 row and 5 columns.
- How to set layout of a cell?
- Add JPanel to the cell and set its layout. Then add components to it.

```java
public class JPanelExample extends JFrame
{
   public static void main(String []args)
   {
      JPanelExample obj=new JPanelExample("Calculator");
   }

   JPanelExample(String title)
   {
      JButton b0=new JButton("0");
      JButton b1=new JButton("1");
      JButton b2=new JButton("2");
      JButton b3=new JButton("3");

      GridLayout gl=new GridLayout(2,1);
      setLayout(gl);

      JPanel jp1=new JPanel();
      jp1.setLayout(new GridLayout(1,3));

      add(b0);
      add(jp1);

      jp1.add(b1);
      jp1.add(b2);
      jp1.add(b3);

      setTitle(title);
      setBounds(200,200,400,550);
      setVisible(true);
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   }
}
```



# Description

- The frame has GridLayout with 2 rows and 1 column.
  - Line 20-21
- We then make a JPanel with 1 row and 3 columns
  - Line 23-24
- We add b0 and jp1 in order to the frame
  - Line 26-27
- Now we can add 3 components to the JPanel.
  - Line 29-31

# Other User Interface Components

- We will make an object of the corresponding class and will add it in exactly the same way as we did for JButton.
- Other classes for User Interface Components are
  - JTextField
  - JRadioButton
  - JLabel
  - ……