ÖZYEĞİN UNIVERSITY

CS 102

Object Oriented Programming

Graphical User Interfaces (GUI)

Reyyan Yeniterzi

reyyan.yeniterzi@ozyegin.edu.tr

Sources:
http://www3.ntu.edu.sg/home/ehchua/programming/java/j4a_gui.html

Java GUI

- Java APIs for GUI programming
  - AWT (**A**bstract **W**indowing **T**oolkit)
    - Sun's first attempt to create a set of cross-platform GUI classes that can be used to implement GUIs.
    - The standart API for implementing Java GUIs
    - Cons: It is limited.
  - Swing
    - This second attempt enhances the earlier AWT
    - Contains much more compherensive and powerful set of graphics libraries

Java GUI

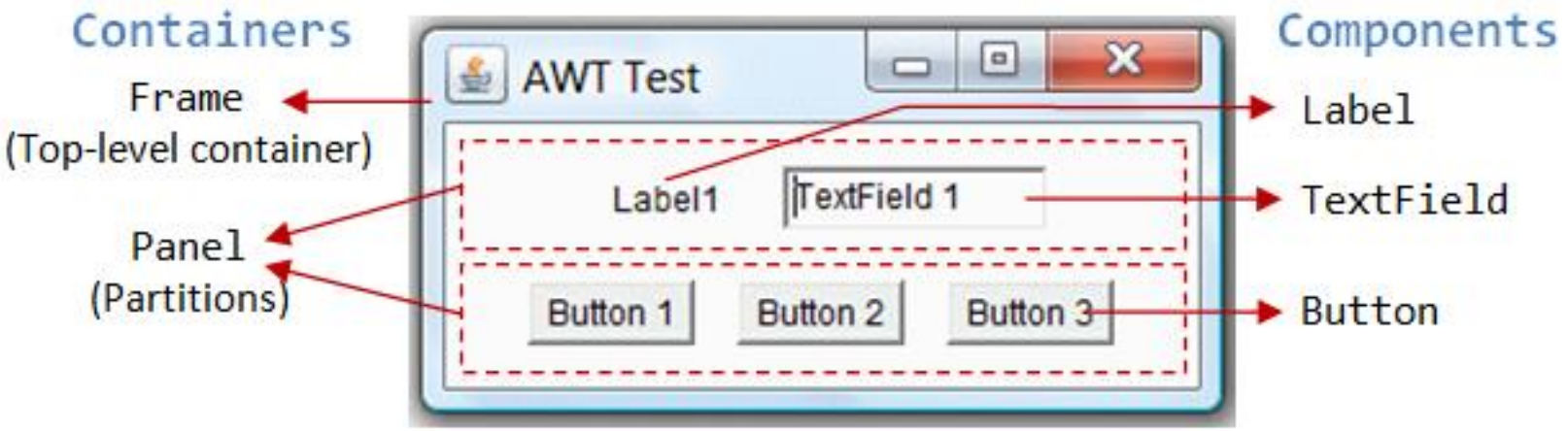
- Java APIs for GUI programming
  - AWT (**A**bstract **W**indowing **T**oolkit)
  - Swing
- They still exist in Java and have to be used together in some cases.
- We will be mixing and using components from both.

4

Components and Containers

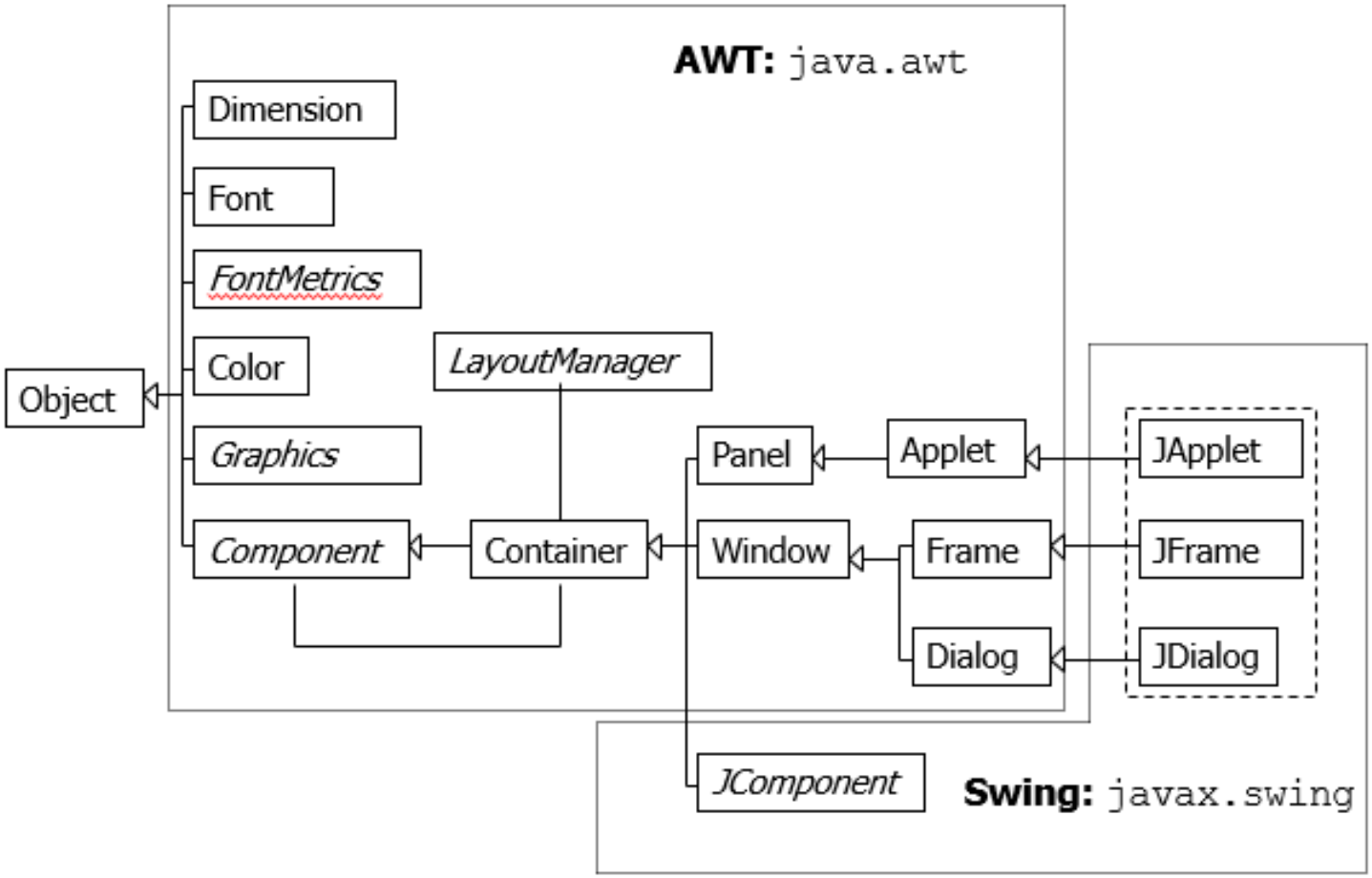
Components and Containers

- Containers are used to hold components together.
- Components (aka. controls) are entities that reside inside containers.



Source:http://www3.ntu.edu.sg/home/ehchua/programming/java/j4a_gui.html

Java GUI


AWT: java.awt
Dimension
Font
FontMetrics
Color
Object
Graphics
Component
LayoutManager
Container
Panel
Applet
JApplet
Window
Frame
JFrame
Dialog
JDialog
JComponent
Swing: javax.swing

GUI Components

- Components: GUI entities like button, label etc.

Source:http://www3.ntu.edu.sg/home/ehchua/programming/java/j4a_gui.html

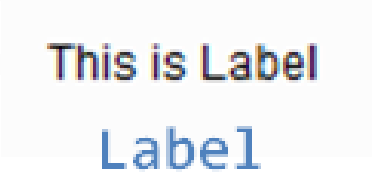
GUI Components

- Swing components starts with a prefix 'J'
  - JComponent
    - JPanel (partition)
    - JScrollPane
    - JTable
    - JButton
    - JLabel
    - JTextField
    - ....

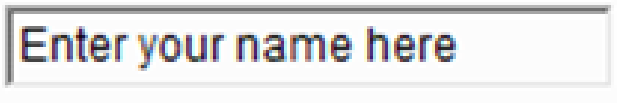
Common GUI Components

- JButton: Clickable component to perform a particular action
- JLabel: Used to display text
- JTextField: Used to get input from user
- JTextArea: Multiline text field

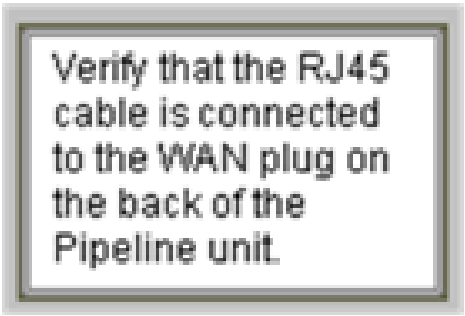
Button



Label



TextField



Text Area

Source:http://www3.ntu.edu.sg/home/ehchua/programming/java/j4a_gui.html

Attributes of Components

name	type	description
enabled	boolean	whether it can be interacted with
focusable	boolean	whether key text can be typed on it
visible	boolean	whether component can be seen
background	Color	background color behind component
foreground	Color	foreground color of component
font	Font	font used for text in component
tooltip text	String	text shown when hovering mouse
border	Border	border line around component
height, width	int	component's current size in pixels
size, minimum / maximum / preferred size	Dimension	various sizes, size limits, or desired sizes that the component may take

Source: GUI slides from http://www.cs.washington.edu/331/

Attributes of Components

name	type	description
enabled	boolean	whether it can be interacted with
f		n it
		ent
height, width	int	component's current size in pixels
size, minimum / maximum / preferred size	Dimension	various sizes, size limits, or desired sizes that the component may take

Each attribute has a `get` (or `is`) accessor and a `set` modifier method.

examples:

```
getColor(), isVisible()
setFont(Arial), setEnabled(true),
```


Source: GUI slides from http://www.cs.washington.edu/331/

Color

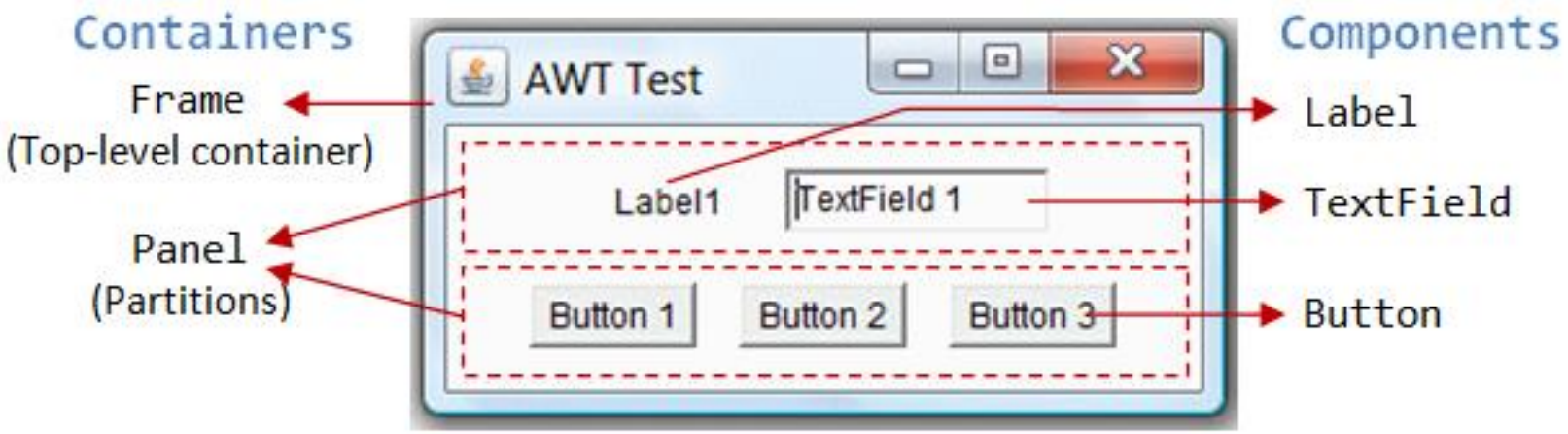
- The color of GUI components can be set using the `java.awt.Color` class
- Colors are made of red, green and blue components which range from 0 (darkest shade) to 255 (lightest shade)
- Each UI component has a background and foreground:

```
Color color = new Color(128, 0, 0);
JButton button = new JButton();
button.setBackground(color); //reddish
button.setForeground(new Color(0, 0, 128));//blueish
```

Colored Buttons

Container Classes

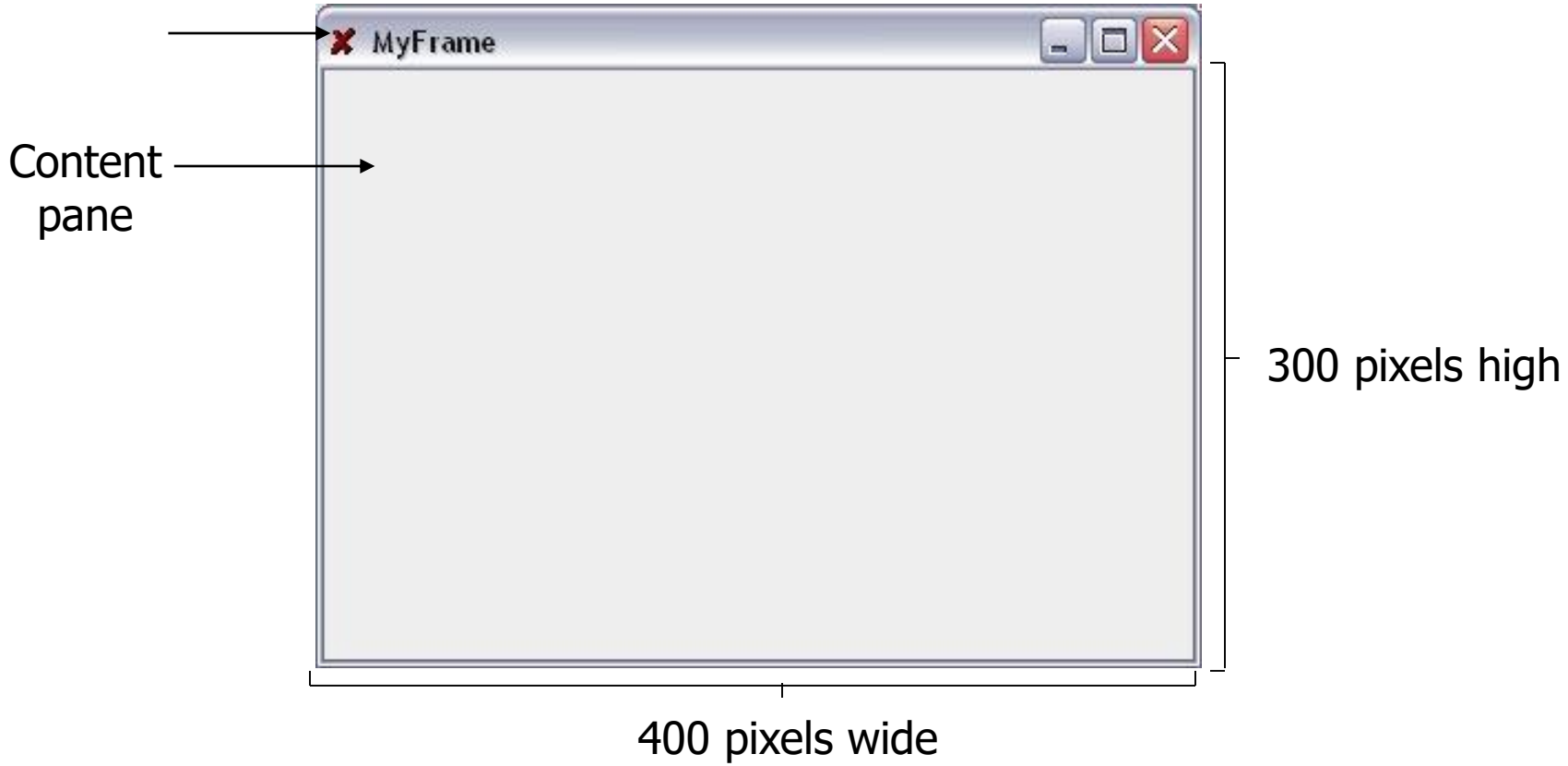
- Containers are used to hold other GUI components in a specific layout.
  - For example: Frames, panels etc.



Source:http://www3.ntu.edu.sg/home/ehchua/programming/java/j4a_gui.html

Frame

- A graphical window used to hold the components.



Frames

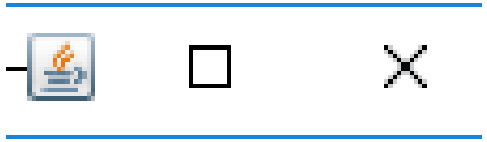

```
import javax.swing.JFrame;

public class example1 {
    public static void main(String[] args) {
        JFrame frame = new JFrame();
    }
}
```

- We have created the frame
- But nothing will be displayed when we run this.

Frames


```
public class example1 {
    public static void main(String[] args) {
        JFrame frame = new JFrame();

        frame.setVisible(true);
    }
}
```

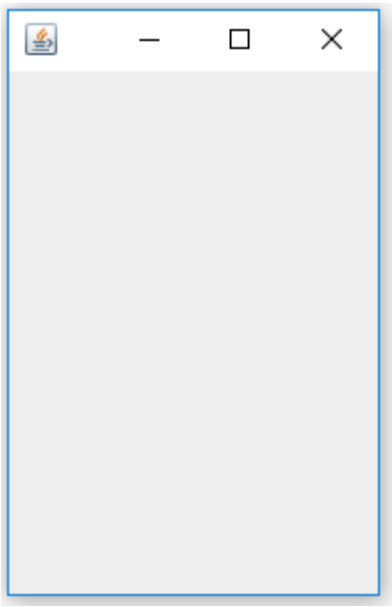
- `setVisible(true)` function makes the frame visible on the screen.



- Now the frame is visible.

Frames


```
public class example1 {
    public static void main(String[] args) {
        JFrame frame = new JFrame();

        int width = 200;
        int height = 300;
        frame.setSize(width, height);

        frame.setVisible(true);
    }
}
```

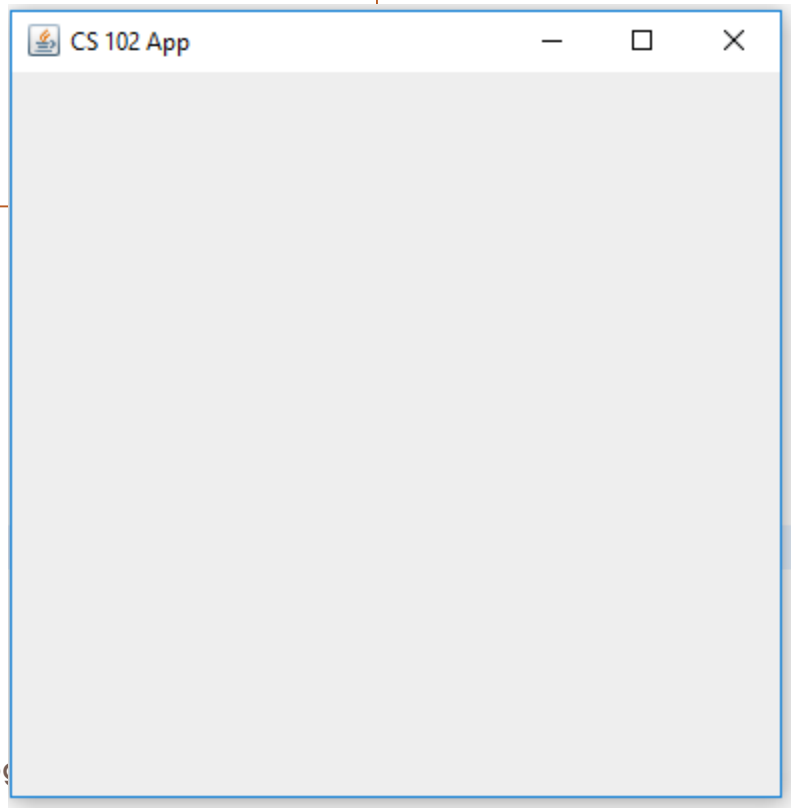


- Sets the frame size to the given pixels.

```
setSize(width, height)
```

Frames


```
public class example1 {
    public static void main(String[] args) {
        JFrame frame = new JFrame("CS 102 App");
        frame.setSize(400, 400);

        frame.setVisible(true);
    }
}
```



- We can give a title to the frame.

Frames


```
public class AbsoluteLayoutExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame();
        frame.setSize(400,400);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        frame.setVisible(true);
    }
}
```

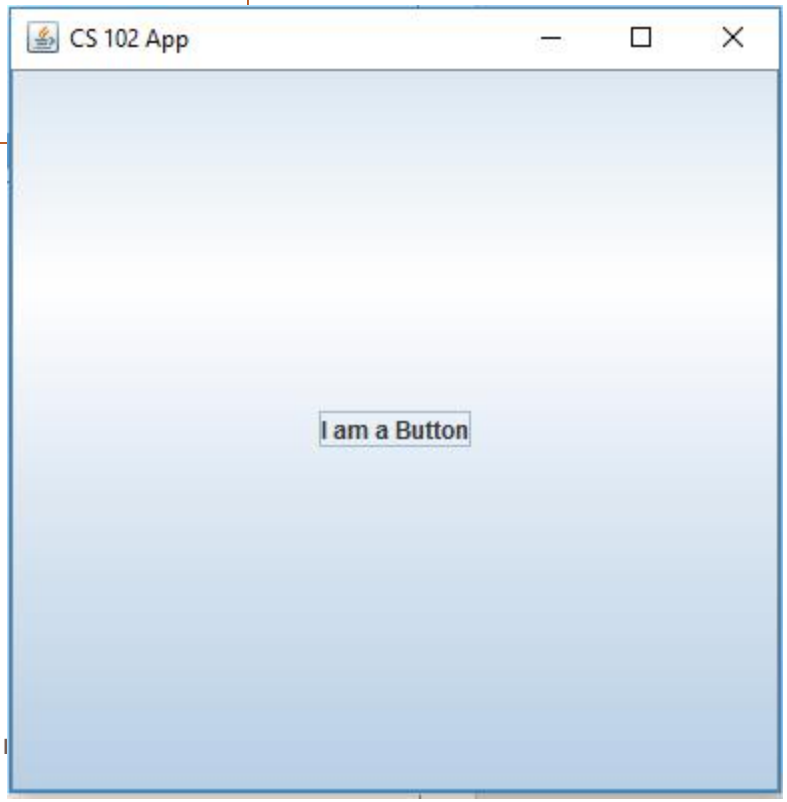
- Optional code specifying what happens when user closes the frame.
- In default it is `frame.setDefaultCloseOperation(JFrame.HIDE_ON_CLOSE);` it looks like you have "killed" the program, but it keeps on running, and you don't see any frame.

Adding Components

- Components can be `add`'ed to the content pane after they are created.
- Containers have `add(component)` method to add the component to the container.

Adding Components


```
public static void main(String[] args) {
    JFrame frame = new JFrame("CS 102 App");
    frame.setSize(400, 400);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JButton button = new JButton("I am a Button");
    frame.add(button);

    frame.setVisible(true);
}
```



- The button is centered in the frame and occupies the whole frame, no matter how it is resized.

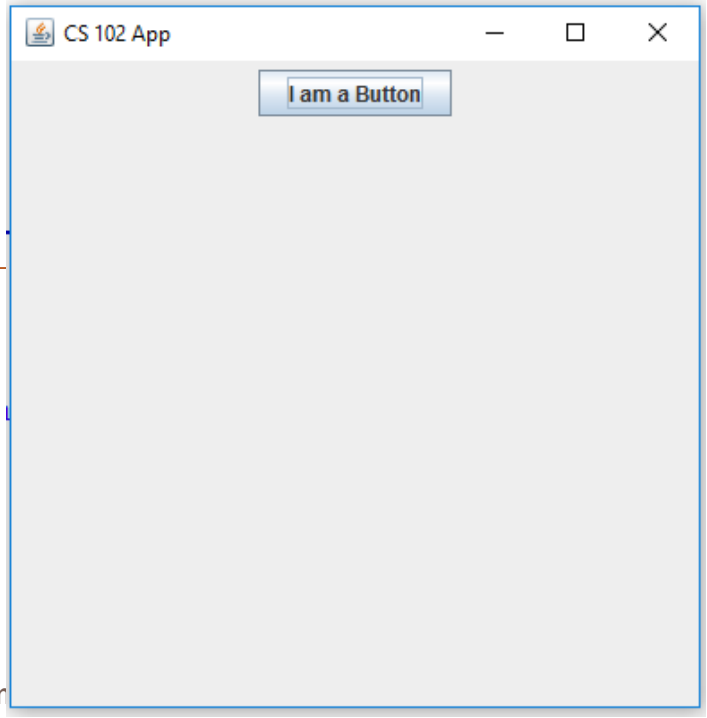
Adding Containers

- `add` method can be also used to add containers to the frame.

- Swing Container Classes
  - **JFrame** is a window not contained inside another window (top-level)
  - **JPanel** is an invisible, nest-able container used to hold UI components or canvases to draw graphics

Adding Containers


```
public static void main(String[] args) {
    JFrame frame = new JFrame("CS 102 App");
    frame.setSize(400, 400);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);

    JButton button = new JButton("I am a Button");
    mainPanel.add(button);

    frame.setVisible(true);
}
```



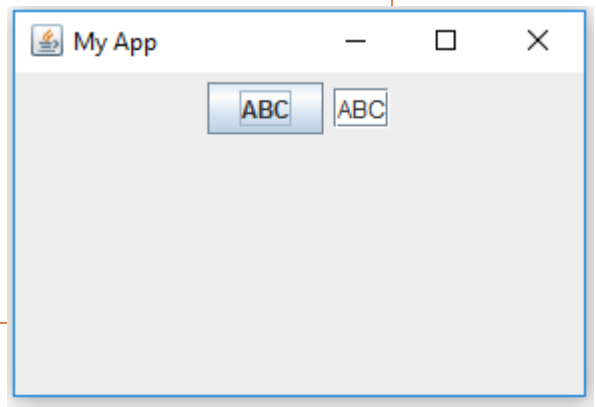
- Even though panel is not visible, it is there.

Example


```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setSize(400, 400);
    frame.setTitle("My App");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);

    JButton button = new JButton("ABC");
    mainPanel.add(button);

    JTextField field = new JTextField();
    field.setText(button.getText());
    mainPanel.add(field);

    frame.setVisible(true);
}
```

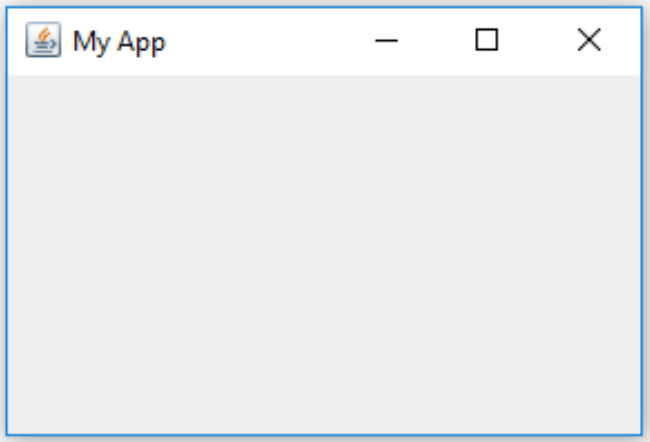
Example


```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setSize(400, 400);
    frame.setTitle("My App");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);

    JButton button = new JButton("ABC");
    mainPanel.add(button);

    JTextField field = new JTextField();
    field.setText(button.getText());
    mainPanel.add(field);

    frame.setVisible(true);
}
```



Example

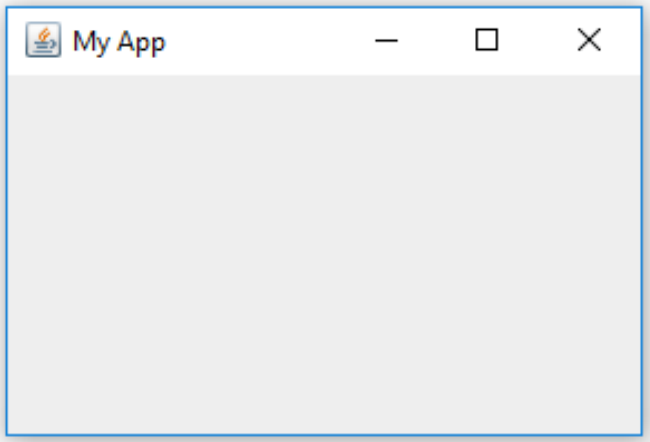

```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setSize(300, 200);
    frame.setTitle("My App");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);

    JButton button = new JButton("ABC");
    mainPanel.add(button);

    JTextField field = new JTextField();
    field.setText(button.getText());
    mainPanel.add(field);
}
```



- setVisible method should be called after all components are added

Example

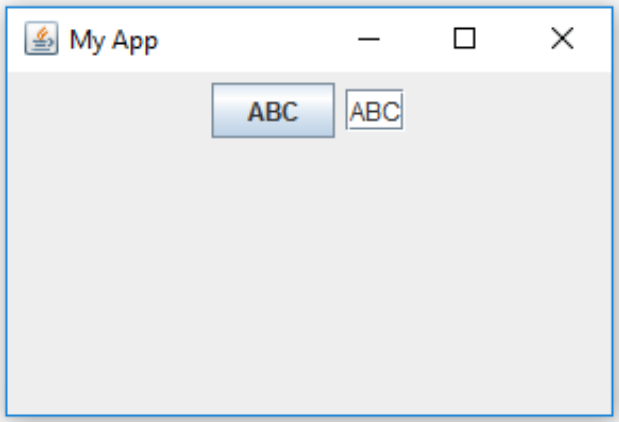

```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setSize(300, 200);
    frame.setTitle("My App");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);

    JButton button = new JButton("ABC");
    mainPanel.add(button);

    JTextField field = new JTextField();
    field.setText(button.getText());
    mainPanel.add(field);
}
```



- setVisible method should be called after all components are added

Example

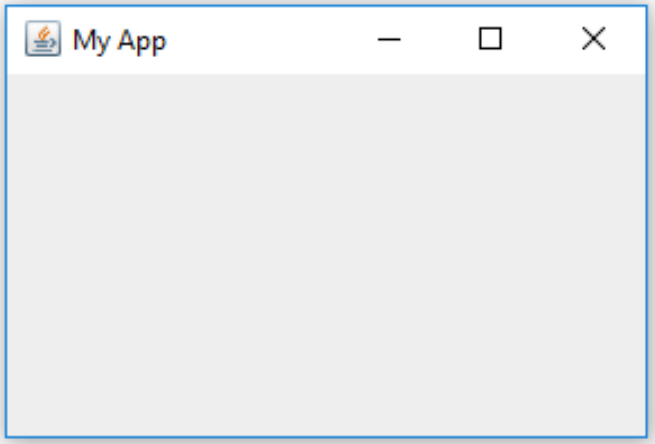

```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setSize(300, 200);
    frame.setTitle("My App");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);

    JButton button = new JButton("ABC");
    mainPanel.add(button);

    JTextField field = new JTextField();
    field.setText(button.getText());
    mainPanel.add(field);

    frame.validate();
}
```



- `validate` method refreshes the frame

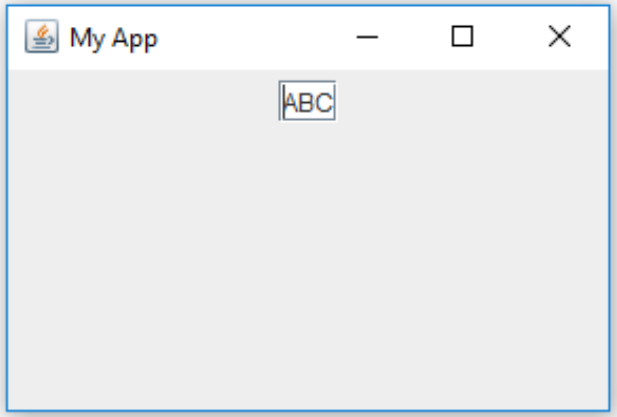
Example


```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setSize(300, 200);
    frame.setTitle("My App");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);

    JButton button = new JButton("ABC");

    JTextField field = new JTextField();
    field.setText(button.getText());

    frame.setVisible(true);
}
```



Example


```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setSize(300, 200);
    frame.setTitle("My App");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);

    JButton button = new JButton("ABC");
    mainPanel.add(button);

    JTextField field = new JTextField();
    field.setText(button.getText());
    mainPanel.add(field);

    mainPanel.remove(button);

    frame.setVisible(true);
}
```



Layout Managers

Layout Managers

- A **layout manager** is used to position and place components in a container
- There are three basic layout managers which control how components are organized on the frame
  - Absolute or Null Layout
  - FlowLayout
  - GridLayout
  - BorderLayout

Layout Managers

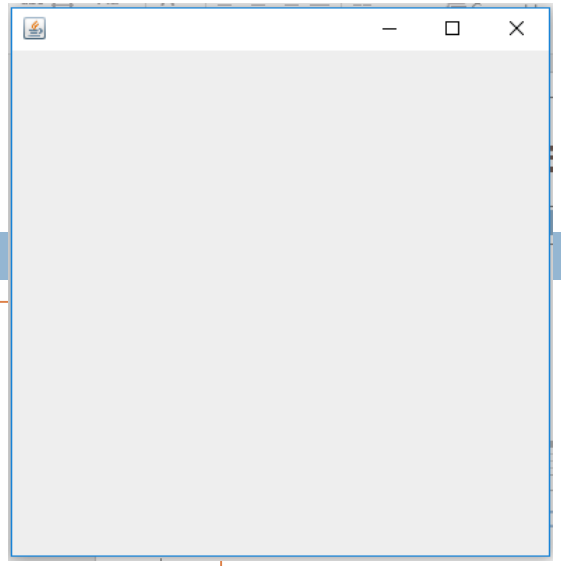
- Once created, the layout can be set in the content pane using `setLayout`.

- As the window is resized, the components reorganize themselves based on the rules of the layout.

Absolute Layout

- **Absolute layout,** enable us to specify the exact location (x,y coordinates) of the components.

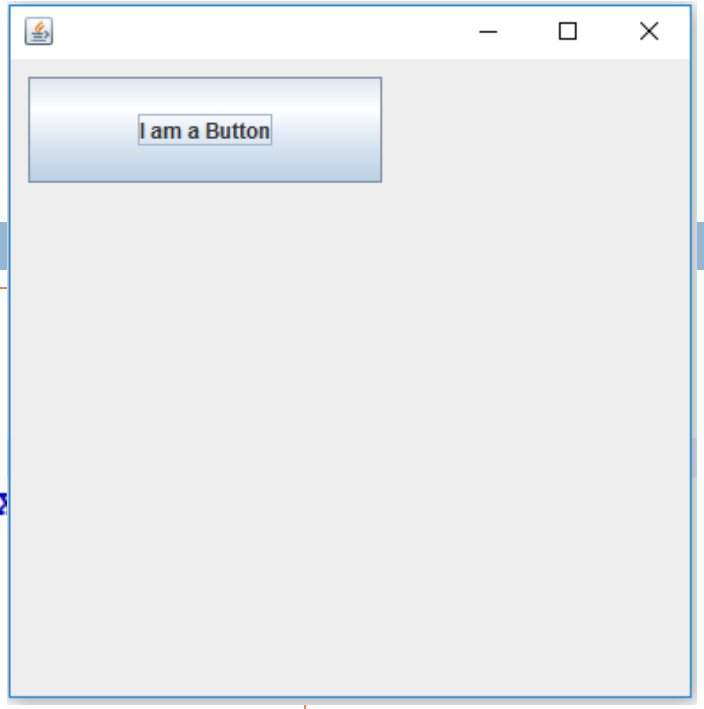
Absolute Layout



```
import javax.swing.JFrame;
import javax.swing.JPanel;

public class AbsoluteLayoutExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame();
        frame.setSize(400,400);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JPanel mainPanel = new JPanel();
        frame.add(mainPanel);
        mainPanel.setLayout(null);

        frame.setVisible(true);
    }
}
```

- Creating a new panel, adding it to the frame and setting its layout to absolute layout.

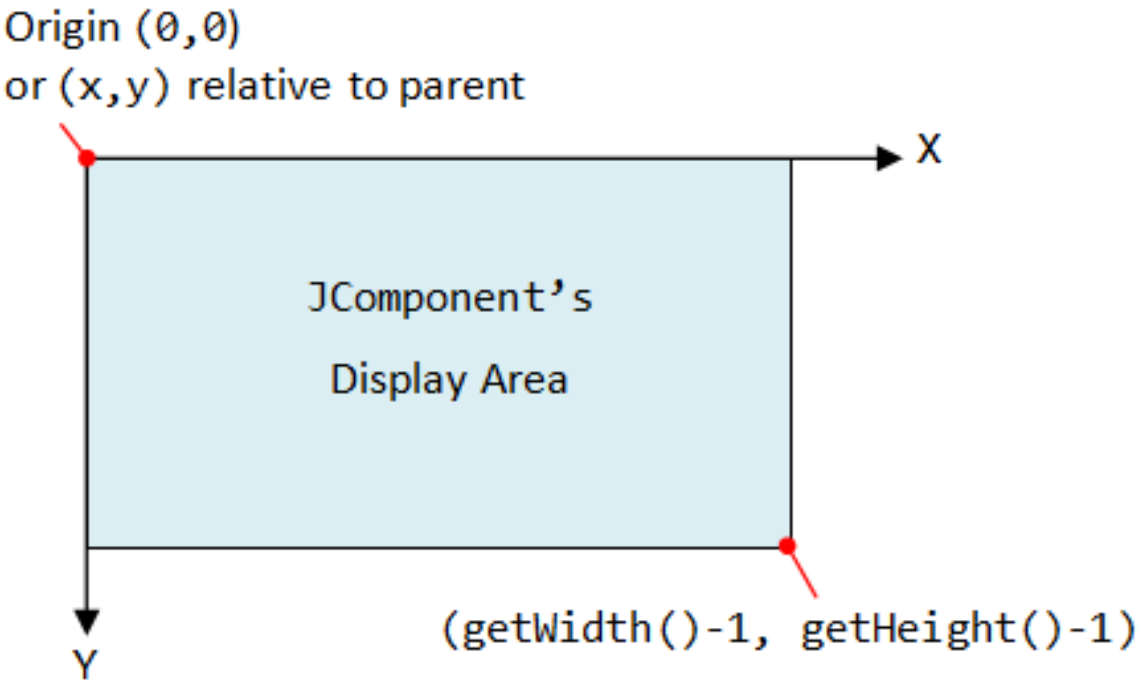
Absolute Layout




```
public class AbsoluteLayoutExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame();
        frame.setSize(400,400);
        frame.setDefaultCloseOperation(JFrame.EX

        JPanel mainPanel = new JPanel();
        frame.add(mainPanel);
        mainPanel.setLayout(null);

        JButton button = new JButton("I am a Button");
        button.setBounds(10, 10, 200, 60);
        mainPanel.add(button);

        frame.setVisible(true);
    }
}
```

- Creating a button and adding it to the panel.
- setBounds(x, y, width, height)

Graphics Coordinate System



- Y axis is inverted

Source:http://www3.ntu.edu.sg/home/ehchua/programming/java/j8b_game_2dgraphics.html

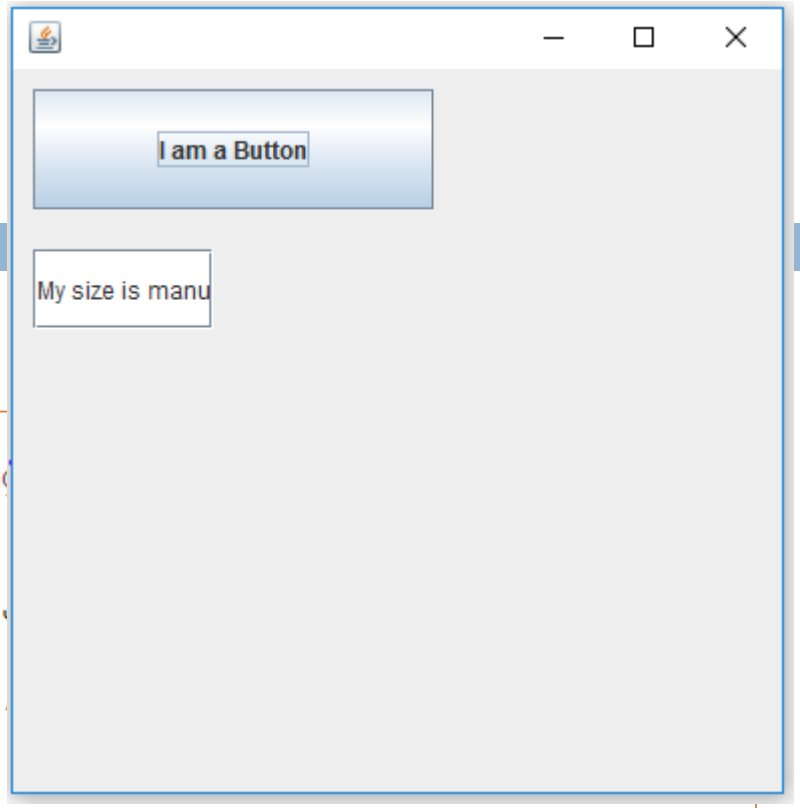
Absolute Layout

- Lets add more components.

```
public class AbsoluteLayoutExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame();
        frame.setSize(400,400);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JPanel mainPanel = new JPanel();
        frame.add(mainPanel);
        mainPanel.setLayout(null);

        JButton button = new JButton("I am a Button");
        button.setBounds(10, 10, 200, 60);
        mainPanel.add(button);

        JTextField field = new JTextField("My size is manually set...");
        field.setBounds(10, 90, 90, 40);
        mainPanel.add(field);

        frame.setVisible(true);
    }
}
```

Absolute Layout

- Lets add more components.




```
public class AbsoluteLayoutExample {
    public static void main(String[] ar
        JFrame frame = new JFrame();
        frame.setSize(400,400);
        frame.setDefaultCloseOperation(

        JPanel mainPanel = new JPanel()
        frame.add(mainPanel);
        mainPanel.setLayout(null);

        JButton button = new JButton("I am a Button");
        button.setBounds(10, 10, 200, 60);
        mainPanel.add(button);

        JTextField field = new JTextField("My size is manually set...");
        field.setBounds(10, 90, 90, 40);
        mainPanel.add(field);

        frame.setVisible(true);
    }
}
```

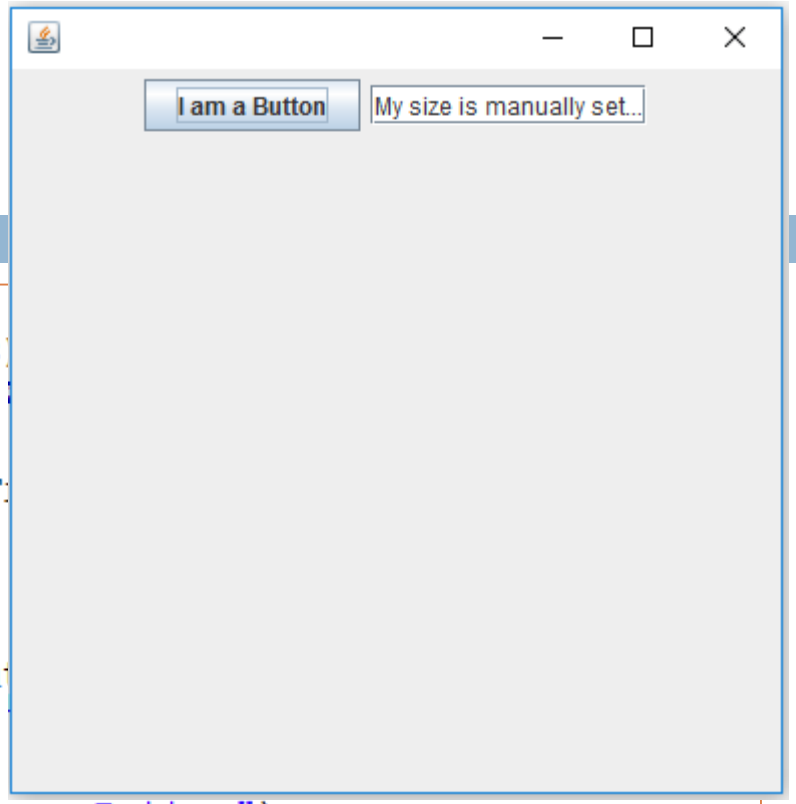
Flow Layout

- With **flow layout,** the components arrange themselves from left to right and top to bottom in the order they were added.

Flow Layout


```
public class FlowLayoutExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame();
        frame.setSize(400,400);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JPanel mainPanel = new JPanel();
        frame.add(mainPanel);
        mainPanel.setLayout(new FlowLayout());


        JButton button = new JButton("I am a Button");
        mainPanel.add(button);

        JTextField field = new JTextField("My size is manually set...");
        mainPanel.add(field);

        frame.setVisible(true);
    }
}
```

Flow Layout



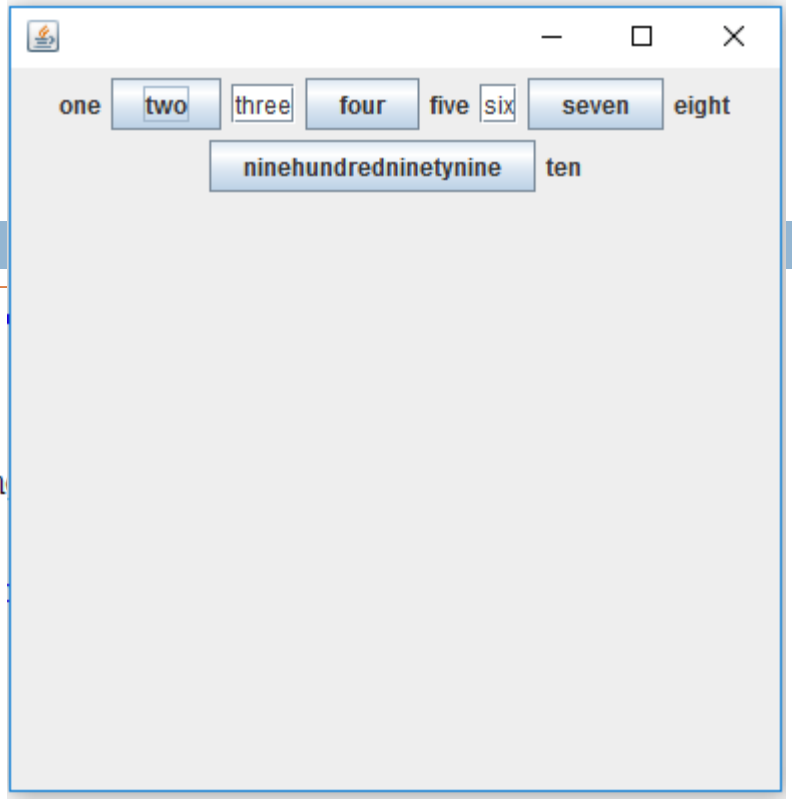

```
public class FlowLayoutExample {
    public static void main(String[] args
        JFrame frame = new JFrame();
        frame.setSize(400,400);
        frame.setDefaultCloseOperation(JF

        JPanel mainPanel = new JPanel();
        frame.add(mainPanel);
        mainPanel.setLayout(new FlowLayou


        JButton button = new JButton("I am a Button");
        mainPanel.add(button);

        JTextField field = new JTextField("My size is manually set...");
        mainPanel.add(field);

        frame.setVi
    }
}
```

The components are being positioned based on the rules of the FlowLayout.
The components are also automatically sized.

Flow Layout


```
public class FlowLayoutExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame();
        frame.setSize(400,400);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JPanel mainPanel = new JPanel();
        frame.add(mainPanel);
        mainPanel.setLayout(new FlowLayout());

        mainPanel.add(new JLabel("one"));
        mainPanel.add(new JButton("two"));
        mainPanel.add(new JTextField("three"));
        mainPanel.add(new JButton("four"));
        mainPanel.add(new JLabel("five"));
        mainPanel.add(new JTextField("six"));
        mainPanel.add(new JButton("seven"));
        mainPanel.add(new JLabel("eight"));
        mainPanel.add(new JButton("ninehundredninetynine"));
        mainPanel.add(new JLabel("ten"));

        frame.setVisible(true);
    }
}
```

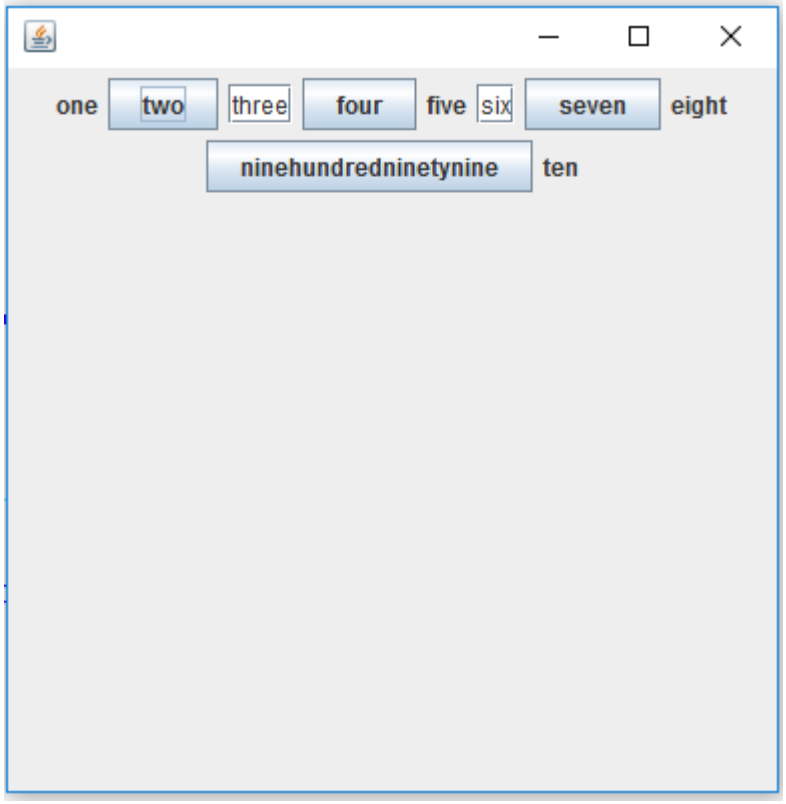
Flow Layout




```
public class FlowLayoutExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame();
        frame.setSize(400,400);
        frame.setDefaultCloseOperation(JFram

        JPanel mainPanel = new JPanel();
        frame.add(mainPanel);
        mainPanel.setLayout(new FlowLayout()

        mainPanel.add(new JLabel("one"));
        mainPanel.add(new JButton("two"));
        mainPanel.add(new JTextField("three"));
        mainPanel.add(new JButton("four"));
        mainPanel.add(new JLabel("five"));
        mainPanel.add(new JTextField("six"));
        mainPanel.add(new JButton("seven"));
        mainPanel.add(new JLabel("eight"));
        mainPanel.add(new JButton("ninehundredninetynine"));
        mainPanel.add(new JLabel("ten"));

        frame.setVisible(true);
    }
}
```

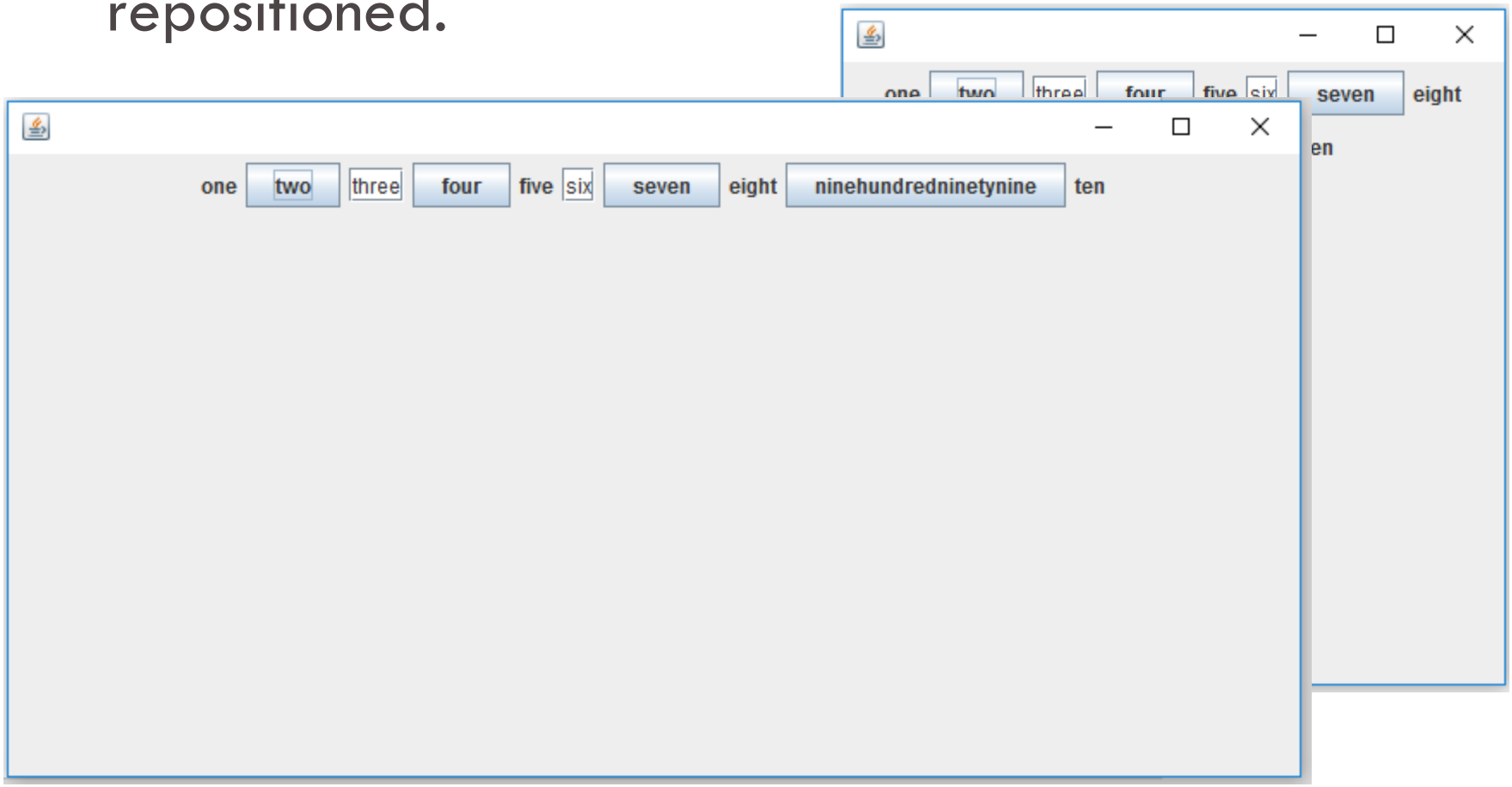
Flow Layout

- When you resize the frame they will be repositioned.



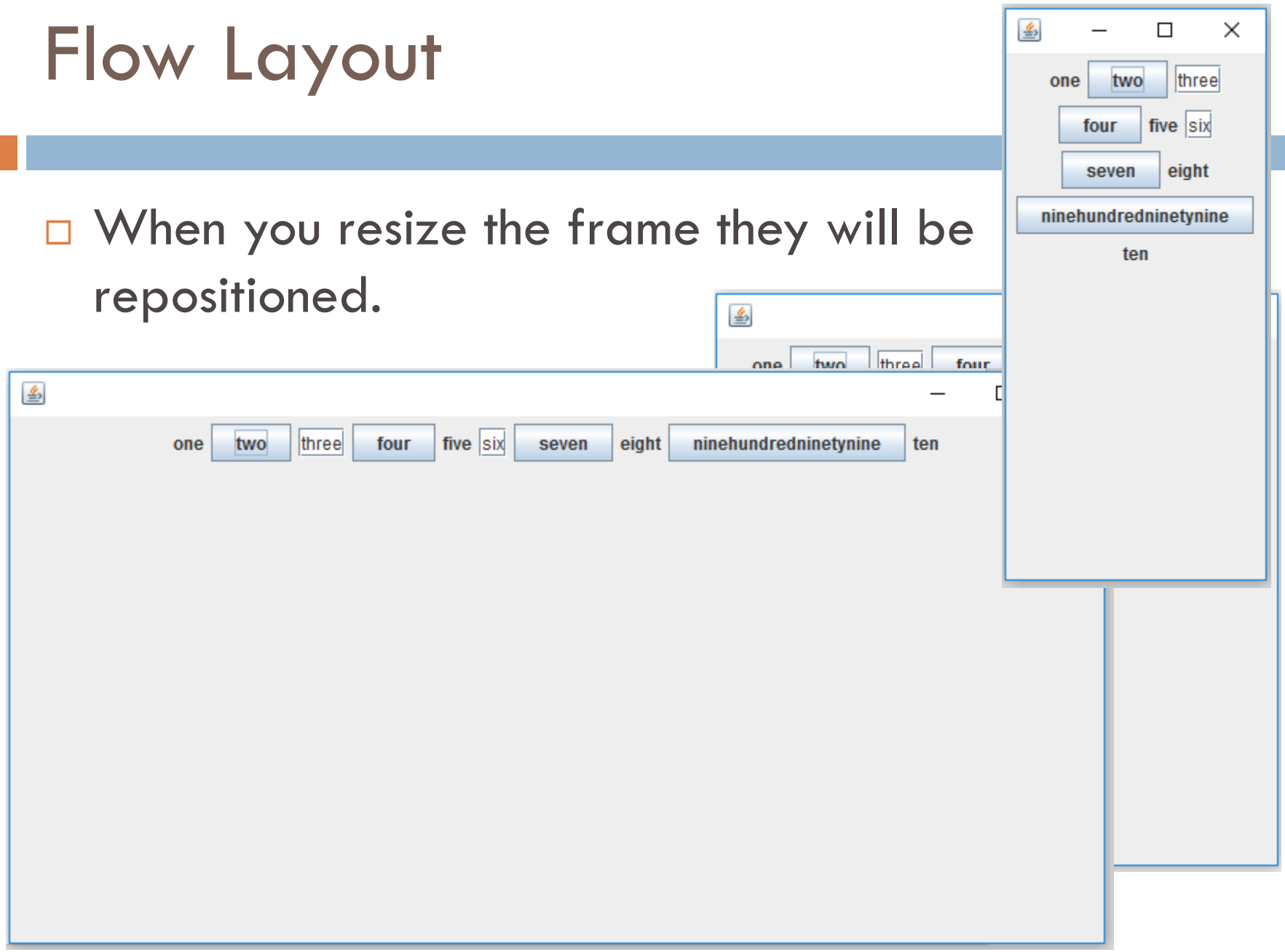
Flow Layout

- When you resize the frame they will be repositioned.



Flow Layout

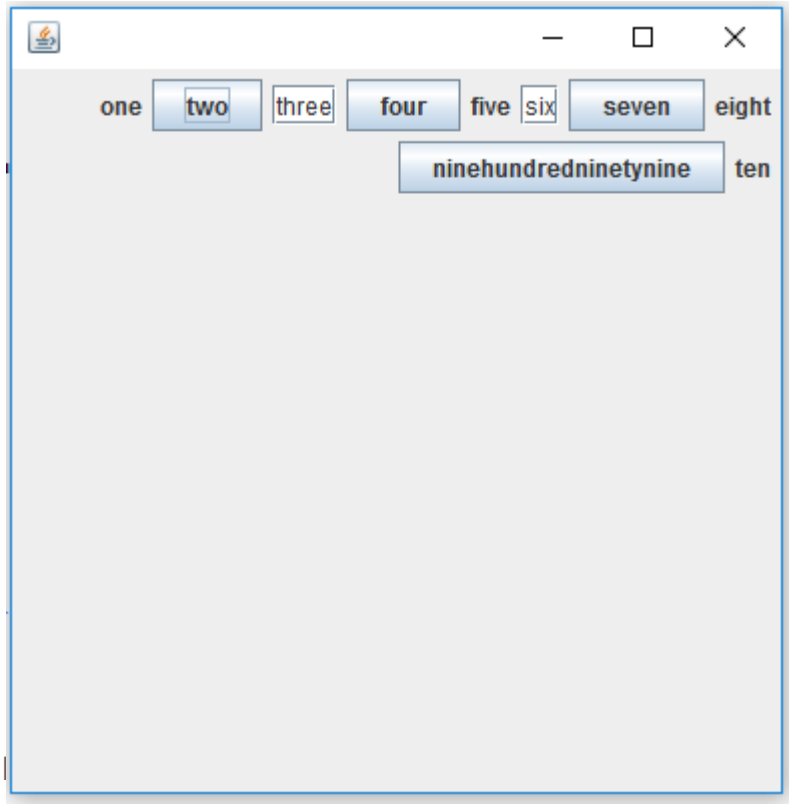
- When you resize the frame they will be repositioned.



Flow Layout

- FlowLayout by default centers the components.

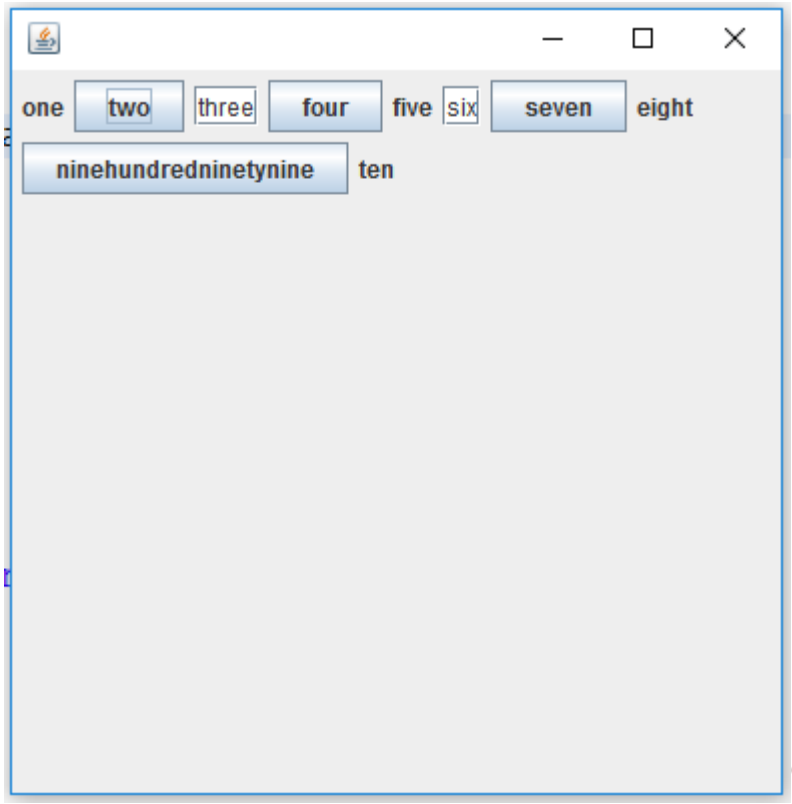
```
mainPanel.setLayout(new FlowLayout(FlowLayout.RIGHT));
```



Flow Layout

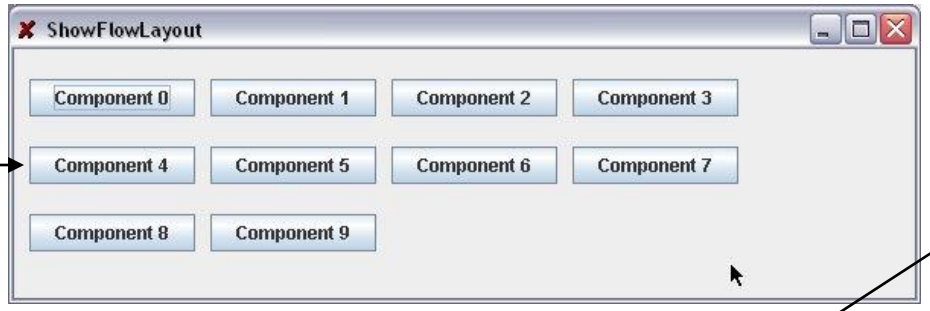
- FlowLayout by default centers the components.

```
mainPanel.setLayout(new FlowLayout(FlowLayout.LEFT));
```



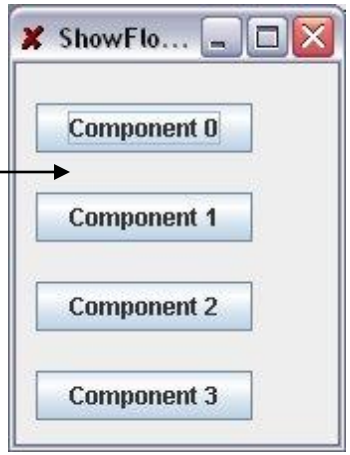
Flow Layout

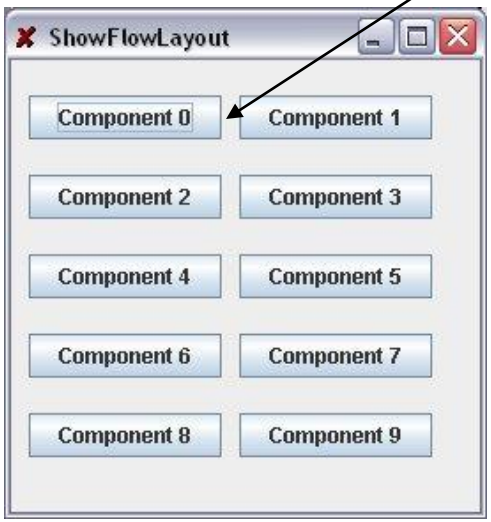
Rows/buttons are left aligned using `FlowLayout.LEFT`



Horizontal gap of 10 pixels

Vertical gap of 20 pixels






```
new FlowLayout(FlowLayout.LEFT,20,10)
```

Grid Layout

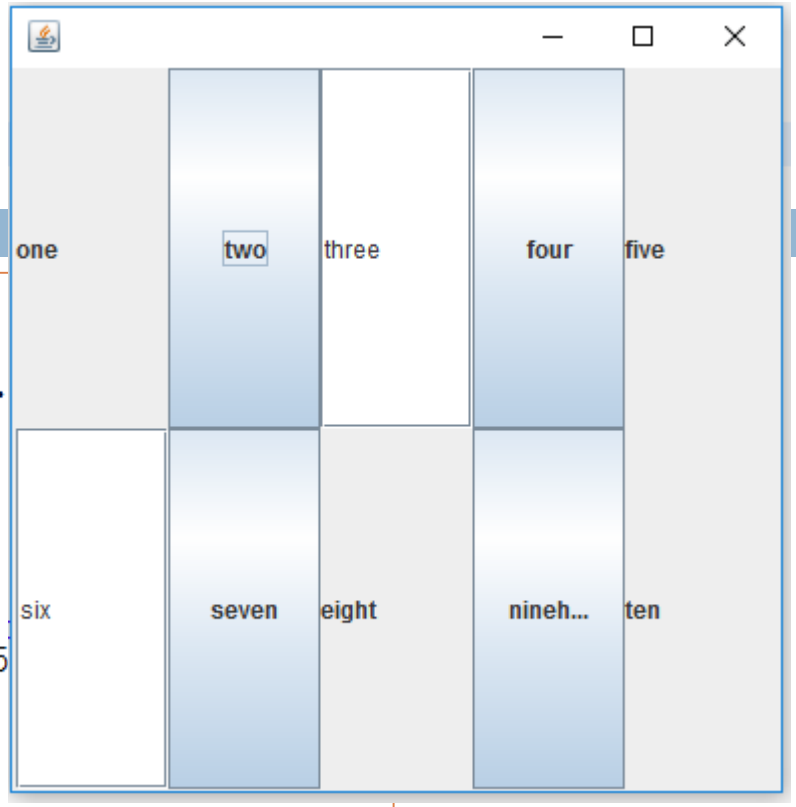
- With **grid layout**, the components arrange themselves in a matrix formation (rows, columns)
- Either the row or column must be non-zero
- The non-zero dimension is fixed and the zero dimension is determined dynamically
- The dominating parameter is the rows

Grid Layout


```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(400,400);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);

    mainPanel.setLayout(new GridLayout(2,5));

    mainPanel.add(new JLabel("one"));
    mainPanel.add(new JButton("two"));
    mainPanel.add(new JTextField("three"));
    mainPanel.add(new JButton("four"));
    mainPanel.add(new JLabel("five"));
    mainPanel.add(new JTextField("six"));
    mainPanel.add(new JButton("seven"));
    mainPanel.add(new JLabel("eight"));
    mainPanel.add(new JButton("ninehundredninetynine"));
    mainPanel.add(new JLabel("ten"));

    frame.setVisible(true);
}
```

Grid Layout


```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setDefaultCloseOperation(JFrame.
    frame.setSize(400,400);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);

    mainPanel.setLayout(new GridLayout(2,5

    mainPanel.add(new JLabel("one"));
    mainPanel.add(new JButton("two"));
    mainPanel.add(new JTextField("three"));
    mainPanel.add(new JButton("four"));
    mainPanel.add(new JLabel("five"));
    mainPanel.add(new JTextField("six"));
    mainPanel.add(new JButton("seven"));
    mainPanel.add(new JLabel("eight"));
    mainPanel.add(new JButton("ninehundredninetynine"));
    mainPanel.add(new JLabel("ten"));

    frame.setVisible(true);
}
```

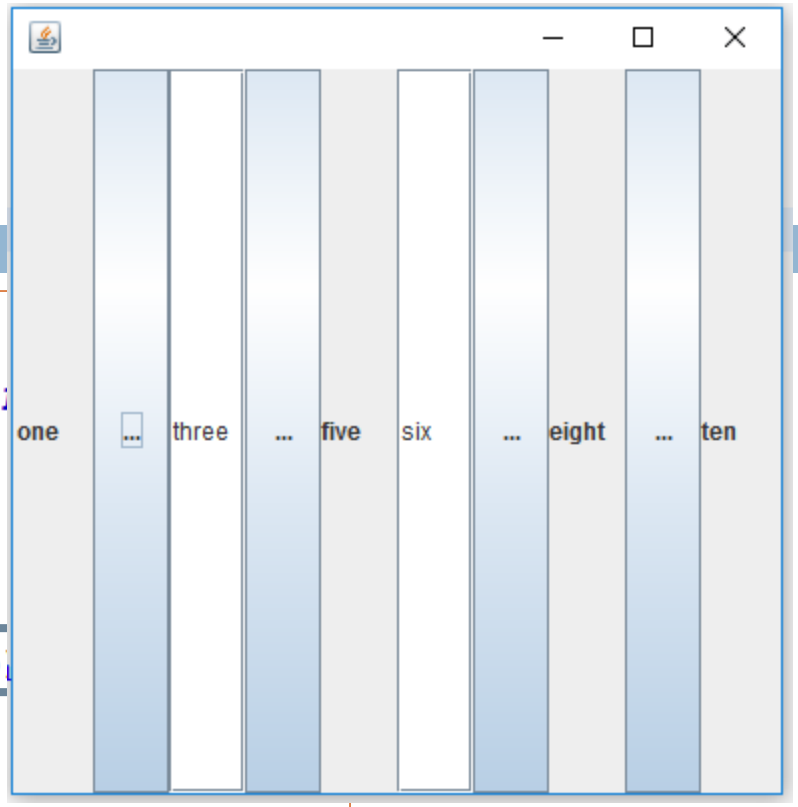


Grid Layout


```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(400,400);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);

    mainPanel.setLayout(new GridLayout(1,5));

    mainPanel.add(new JLabel("one"));
    mainPanel.add(new JButton("two"));
    mainPanel.add(new JTextField("three"));
    mainPanel.add(new JButton("four"));
    mainPanel.add(new JLabel("five"));
    mainPanel.add(new JTextField("six"));
    mainPanel.add(new JButton("seven"));
    mainPanel.add(new JLabel("eight"));
    mainPanel.add(new JButton("ninehundredninetynine"));
    mainPanel.add(new JLabel("ten"));

    frame.setVisible(true);
}
```

Grid Layout

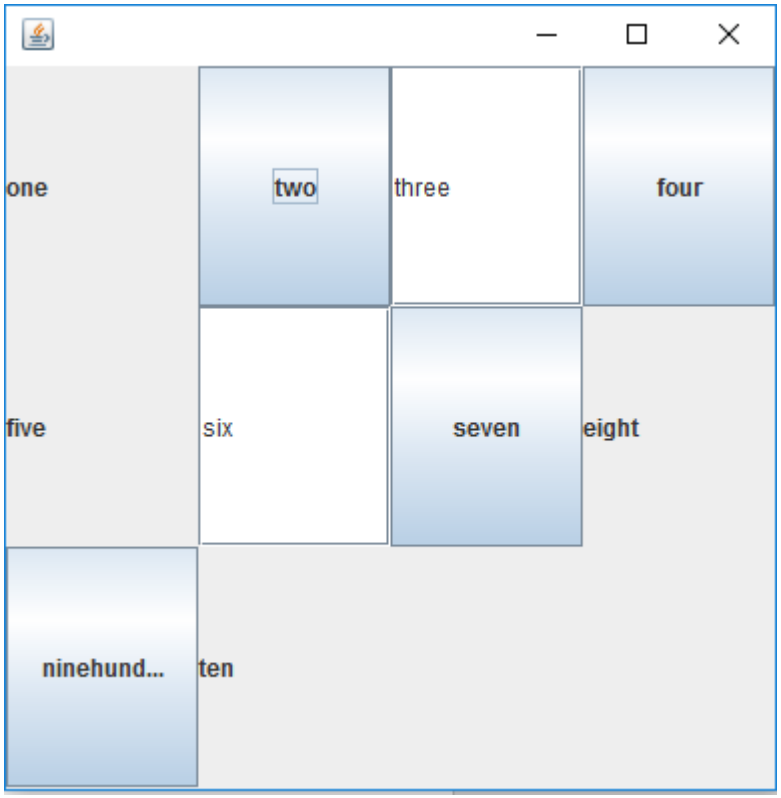

```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setDefaultCloseOperation(JFrame.
    frame.setSize(400,400);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);

    mainPanel.setLayout(new GridLayout(1,5

    mainPanel.add(new JLabel("one"));
    mainPanel.add(new JButton("two"));
    mainPanel.add(new JTextField("three"));
    mainPanel.add(new JButton("four
    mainPanel.add(new JLabel("five"
    mainPanel.add(new JTextField("s
    mainPanel.add(new JButton("seve
    mainPanel.add(new JLabel("eight"));
    mainPanel.add(new JButton("ninehundredninetynine"));
    mainPanel.add(new JLabel("ten"));

    frame.setVisible(true);
}
```



The row dominates the column.

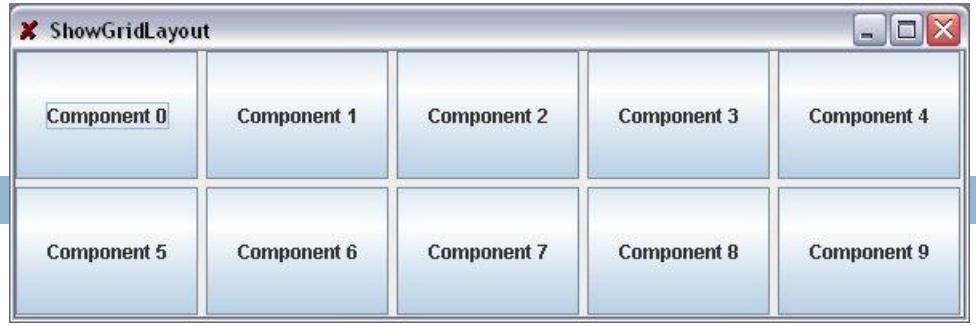
Grid Layout


```
mainPanel.setLayout(new GridLayout(3,5));
```



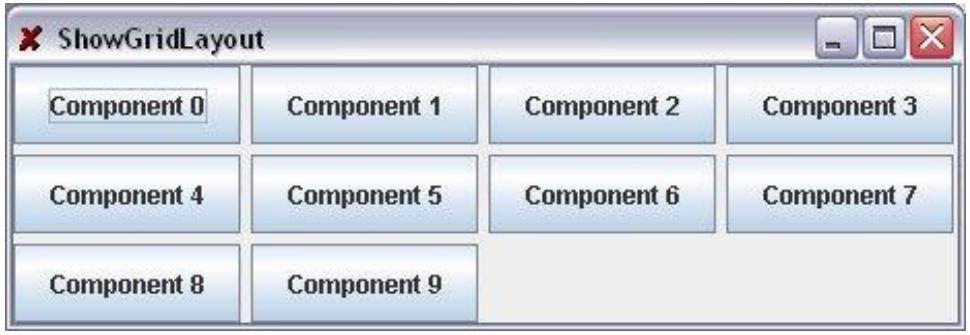
Grid Layout

2,4

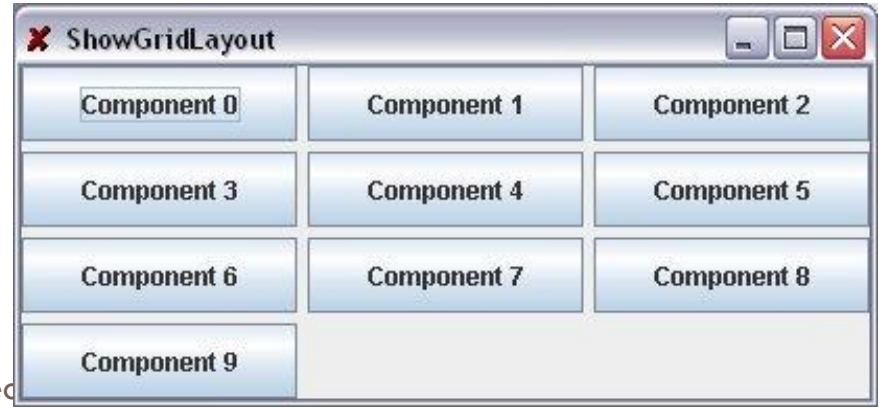


10, 10

0,4

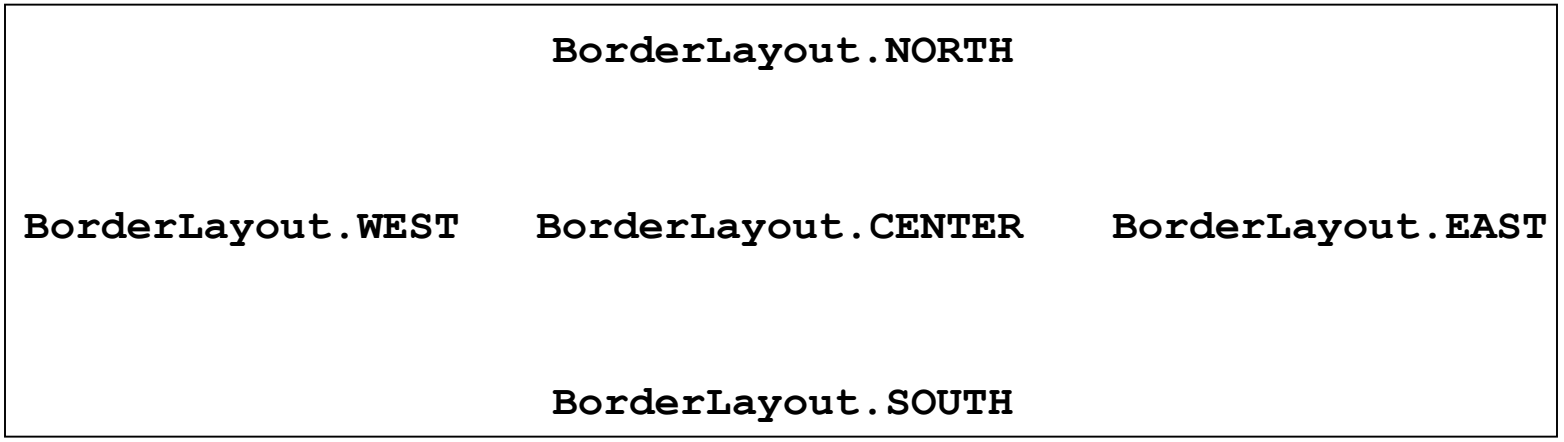


4,4



Border Layout

- With **border layout**, the window is divided into five areas:



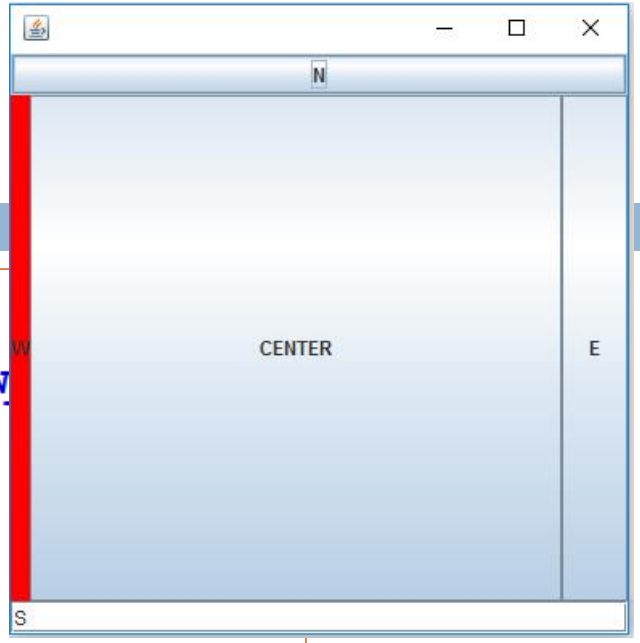
- Components are added to the frame using a specified index:

```
container.add(new JButton("East"),
  BorderLayout.EAST);
```

Border Layout

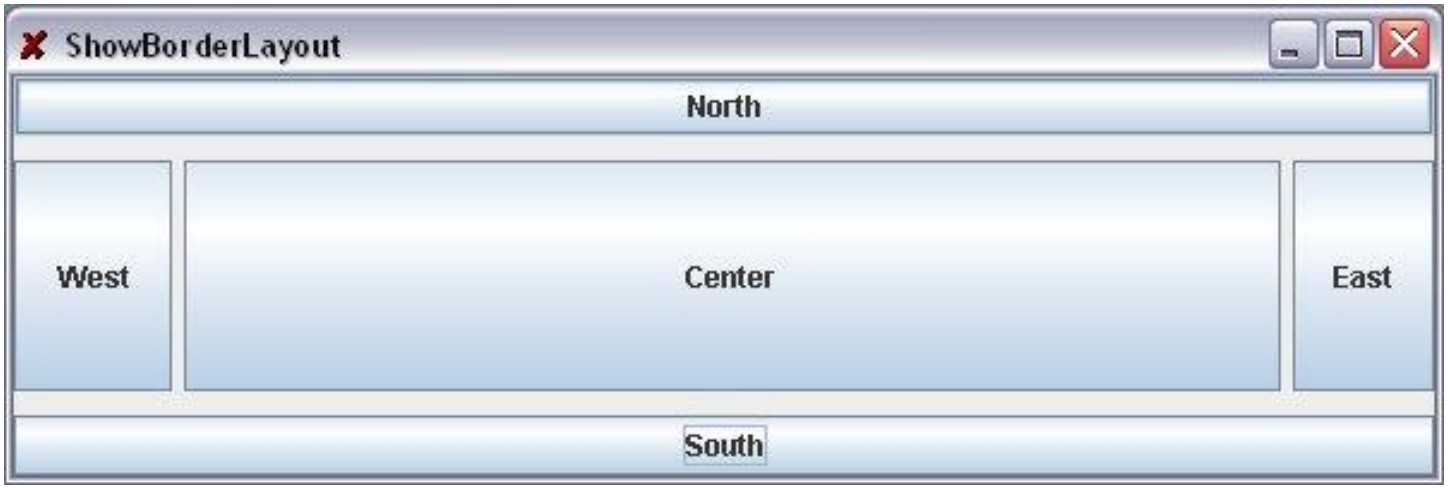

```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(400,400);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);

    mainPanel.setLayout(new BorderLayout());
    mainPanel.setBackground(Color.RED);

    mainPanel.add(new JButton("CENTER"), BorderLayout.CENTER);
    mainPanel.add(new JTextField("S"), BorderLayout.SOUTH);
    mainPanel.add(new JButton("N"), BorderLayout.NORTH);
    mainPanel.add(new JLabel("W"), BorderLayout.WEST);
    mainPanel.add(new JButton("E"), BorderLayout.EAST);

    frame.setVisible(true);
}
```

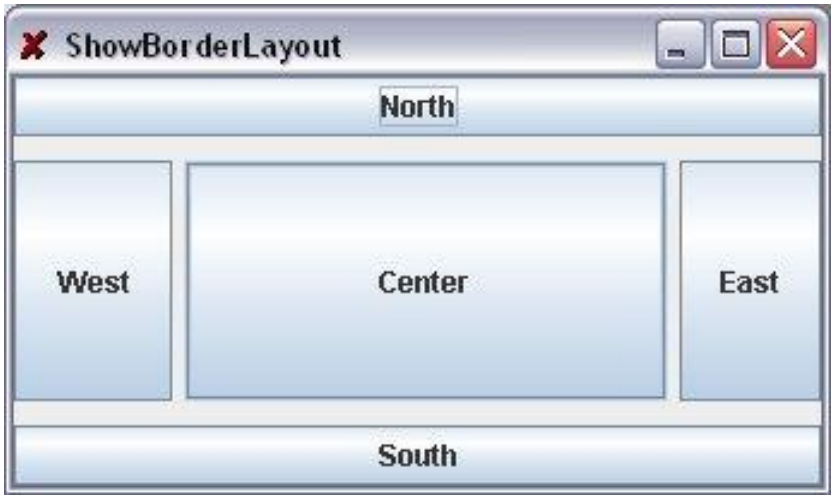
Border Layout

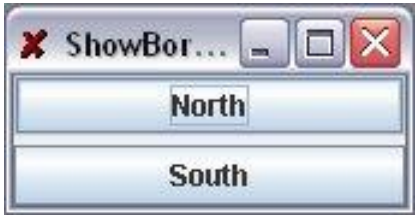



```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setDefaultCloseOperation(JFrame.EXIT_ON
    frame.setSize(400,400);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);

    mainPanel.setLayout(new BorderLayout());
    mainPanel.setBackground(Color.RED);

    mainPanel.add(new JButton("CENTER"), BorderLayout.CENTER);
    mainPanel.add(new JTextField("S"), BorderLayout.SOUTH);
    mainPanel.add(new JButton("N"), BorderLayout.NORTH);
    mainPanel.add(new JLabel("W"), BorderLayout.WEST);
    mainPanel.add(new JButton("E"), BorderLayout.EAST);

    frame.setVisible(true);
}
```

Border Layout







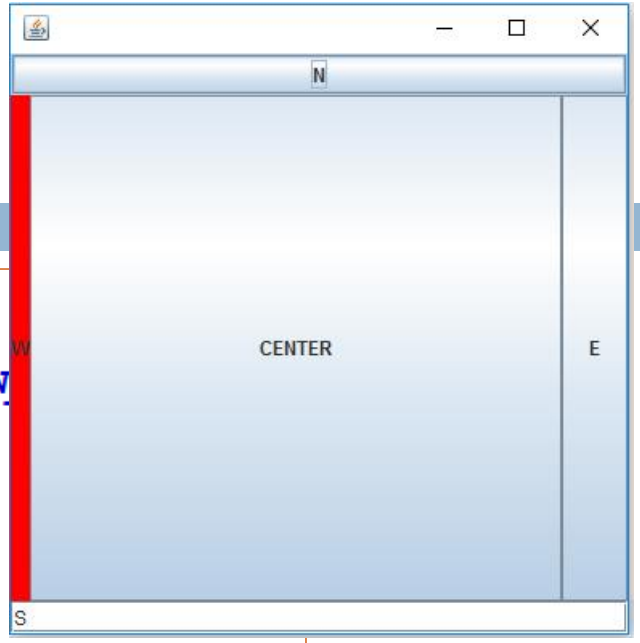
Border Layout

- The components stretch in this manner:
  - North and South stretch horizontally
  - East and West stretch vertically
  - Center can stretch in both directions to fill space
- The default location for a component is `BorderLayout.CENTER`
- If you add two components to the same location, only the last one will be displayed
- It is unnecessary to place components to occupy all areas

Deafult Layouts

- Border layout is the default layout for JFrame
- Flow layout is the default layout for JPanel

Border Layout

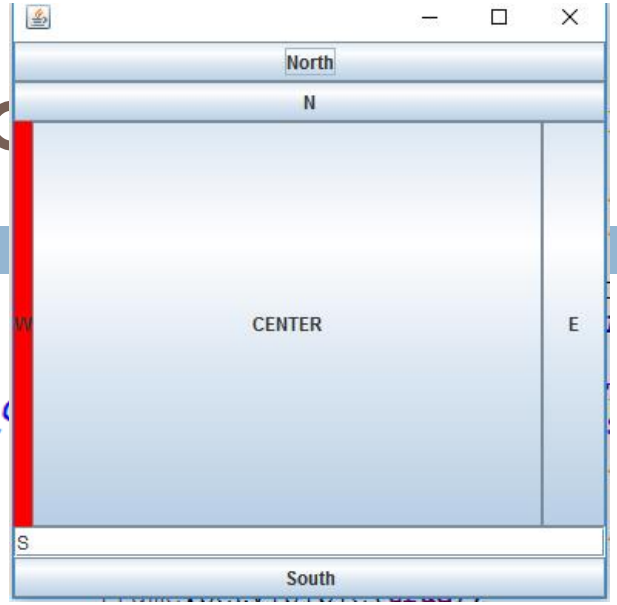



```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setDefaultCloseOperation(JFrame.EXIT_ON
    frame.setSize(400,400);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);

    mainPanel.setLayout(new BorderLayout());
    mainPanel.setBackground(Color.RED);

    mainPanel.add(new JButton("CENTER"), BorderLayout.CENTER);
    mainPanel.add(new JTextField("S"), BorderLayout.SOUTH);
    mainPanel.add(new JButton("N"), BorderLayout.NORTH);
    mainPanel.add(new JLabel("W"), BorderLayout.WEST);
    mainPanel.add(new JButton("E"), BorderLayout.EAST);

    frame.setVisible(true);
}
```

What does the output look like?


```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(400,400);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);
    frame.add(new JButton("South"), BorderLayout.SOUTH);
    frame.add(new JButton("North"), BorderLayout.NORTH);

    mainPanel.setLayout(new BorderLayout());
    mainPanel.setBackground(Color.RED);

    mainPanel.add(new JButton("CENTER"), BorderLayout.CENTER);
    mainPanel.add(new JTextField("S"), BorderLayout.SOUTH);
    mainPanel.add(new JButton("N"), BorderLayout.NORTH);
    mainPanel.add(new JLabel("W"), BorderLayout.WEST);
    mainPanel.add(new JButton("E"), BorderLayout.EAST);

    frame.setVisible(true);
}
```

What does the output lo

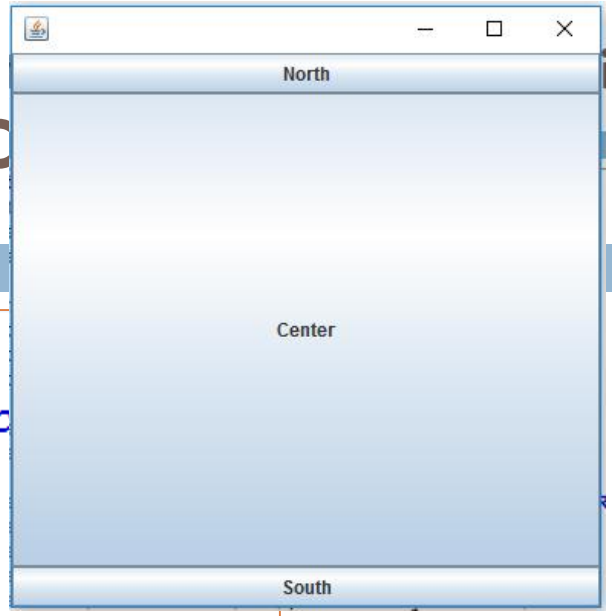



```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_
    frame.setSize(400,400);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);
    frame.add(new JButton("South"), BorderLayout.SOUTH);
    frame.add(new JButton("North"), BorderLayout.NORTH);

    mainPanel.setLayout(new BorderLayout());
    mainPanel.setBackground(Color.RED);

    mainPanel.add(new JButton("CENTER"), BorderLayout.CENTER);
    mainPanel.add(new JTextField("S"), BorderLayout.SOUTH);
    mainPanel.add(new JButton("N"), BorderLayout.NORTH);
    mainPanel.add(new JLabel("W"), BorderLayout.WEST);
    mainPanel.add(new JButton("E"), BorderLayout.EAST);

    frame.setVisible(true);
}
```

What does the output look like?


```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(400,400);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);
    frame.add(new JButton("South"), BorderLayout.SOUTH);
    frame.add(new JButton("North"), BorderLayout.NORTH);

    mainPanel.setLayout(new BorderLayout());
    mainPanel.setBackground(Color.RED);

    mainPanel.add(new JButton("CENTER"), BorderLayout.CENTER);
    mainPanel.add(new JTextField("S"), BorderLayout.SOUTH);
    mainPanel.add(new JButton("N"), BorderLayout.NORTH);
    mainPanel.add(new JLabel("W"), BorderLayout.WEST);
    mainPanel.add(new JButton("E"), BorderLayout.EAST);

    frame.add(new JButton("Center"), BorderLayout.CENTER);
    frame.setVisible(true);
}
```

What does the output lo



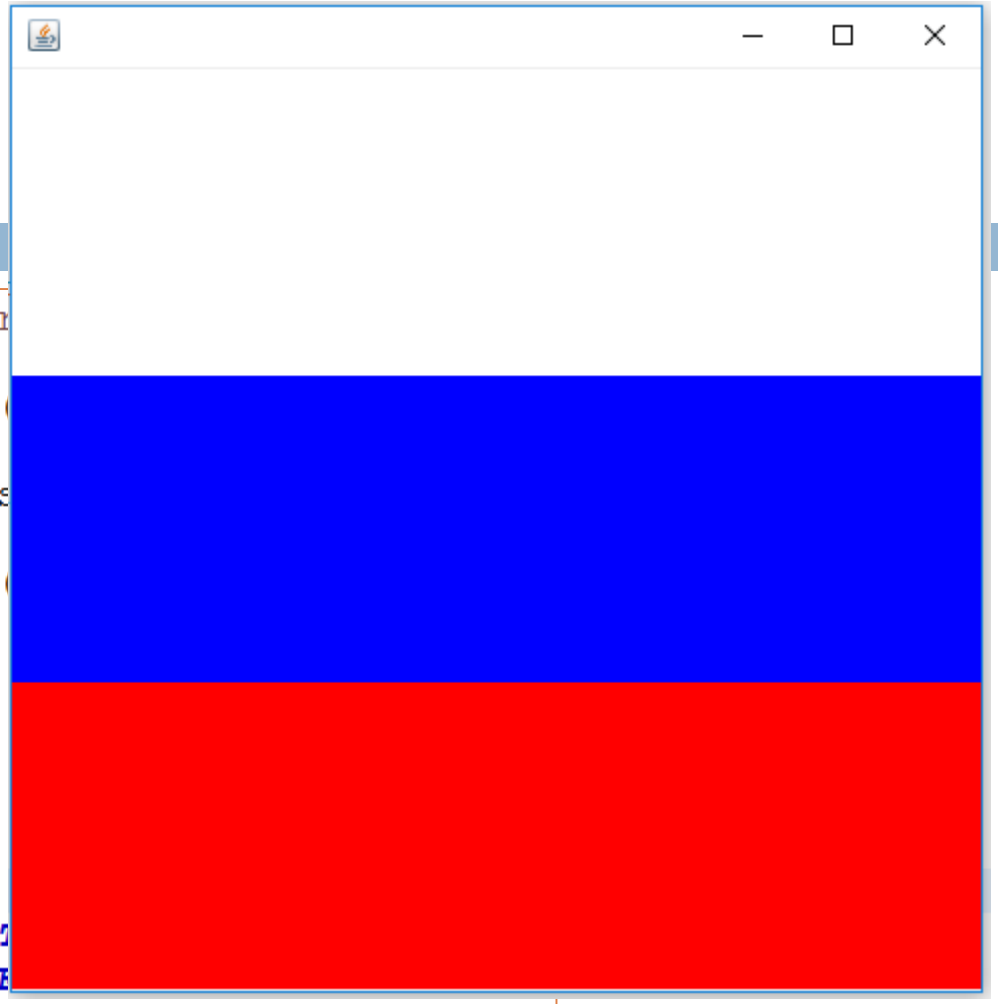

```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_C
    frame.setSize(400,400);

    JPanel mainPanel = new JPanel();
    frame.add(mainPanel);
    frame.add(new JButton("South"), BorderLayout.SOUTH);
    frame.add(new JButton("North"), BorderLayout.NORTH);

    mainPanel.setLayout(new BorderLayout());
    mainPanel.setBackground(Color.RED);

    mainPanel.add(new JButton("CENTER"), BorderLayout.CENTER);
    mainPanel.add(new JTextField("S"), BorderLayout.SOUTH);
    mainPanel.add(new JButton("N"), BorderLayout.NORTH);
    mainPanel.add(new JLabel("W"), BorderLayout.WEST);
    mainPanel.add(new JButton("E"), BorderLayout.EAST);

    frame.add(new JButton("Center"), BorderLayout.CENTER);
    frame.setVisible(true);
}
```

Example – Part 1


```
public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(500,500);
    frame.setMinimumSize(new Dimension(400, 400));

    frame.setLayout(new GridLayout(3,1));
    JPanel panel1 = new JPanel();
    JPanel panel2 = new JPanel();
    JPanel panel3 = new JPanel();
    frame.add(panel1);
    frame.add(panel2);
    frame.add(panel3);

    panel1.setBackground(Color.WHITE);
    panel2.setBackground(Color.BLUE);
    panel3.setBackground(Color.RED);

    panel1.setLayout(new BorderLayout());
    panel2.setLayout(new FlowLayout());
    panel3.setLayout(new GridLayout(5,2));
```

Example – Part


```
public static void main(String[] ar
    JFrame frame = new JFrame();
    frame.setDefaultCloseOperation(
    frame.setSize(500,500);
    frame.setMinimumSize(new Dimens

    frame.setLayout(new GridLayout(
    JPanel panel1 = new JPanel();
    JPanel panel2 = new JPanel();
    JPanel panel3 = new JPanel();
    frame.add(panel1);
    frame.add(panel2);
    frame.add(panel3);

    panel1.setBackground(Color.WHI
    panel2.setBackground(Color.BLU
    panel3.setBackground(Color.RED);

    panel1.setLayout(new BorderLayout());
    panel2.setLayout(new FlowLayout());
    panel3.setLayout(new GridLayout(5,2));
```

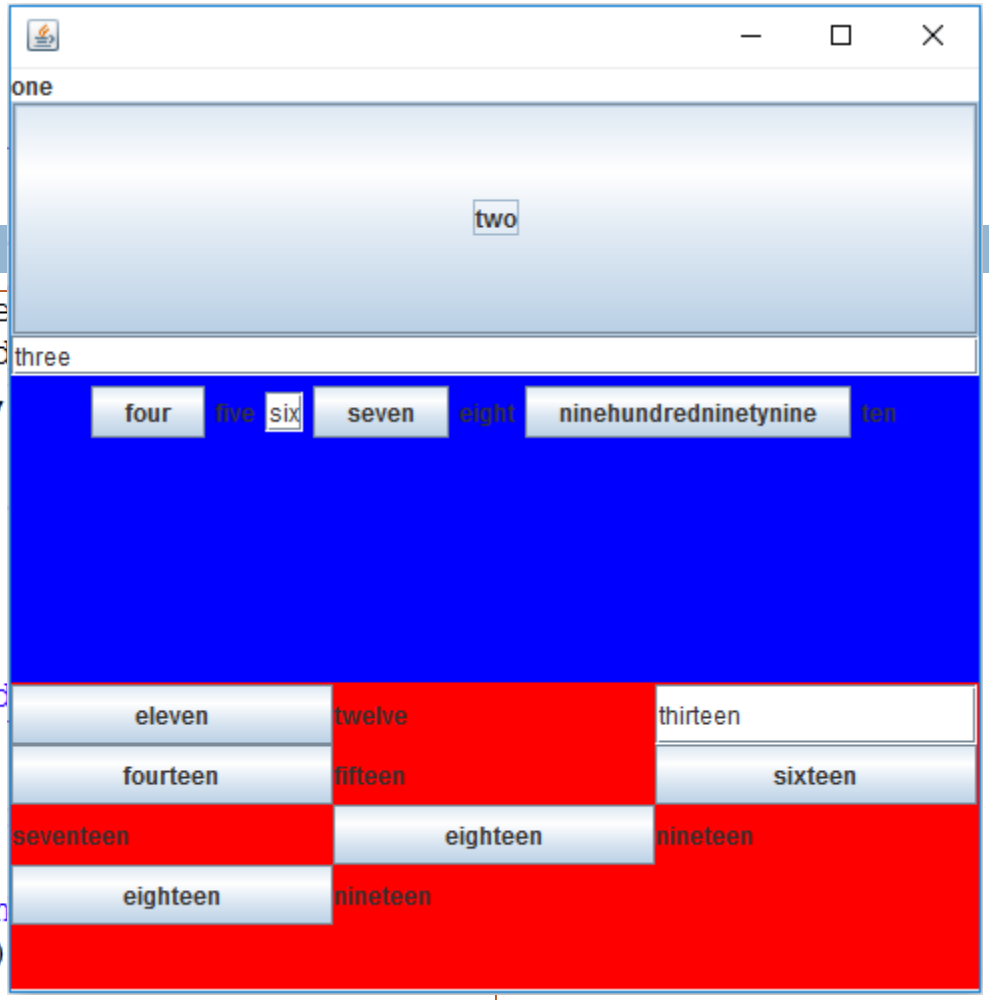


Example – Part 2


```
panel1.add(new JLabel("one"), BorderLayout.NORTH);
panel1.add(new JButton("two"), BorderLayout.CENTER);
panel1.add(new JTextField("three"), BorderLayout.SOUTH);

panel2.add(new JButton("four"));
panel2.add(new JLabel("five"));
panel2.add(new JTextField("six"));
panel2.add(new JButton("seven"));
panel2.add(new JLabel("eight"));
panel2.add(new JButton("ninehundredninetynine"));
panel2.add(new JLabel("ten"));

panel3.add(new JButton("eleven"));
panel3.add(new JLabel("twelve"));
panel3.add(new JTextField("thirteen"));
panel3.add(new JButton("fourteen"));
panel3.add(new JLabel("fifteen"));
panel3.add(new JButton("sixteen"));
panel3.add(new JLabel("seventeen"));
panel3.add(new JButton("eighteen"));
panel3.add(new JLabel("nineteen"));
panel3.add(new JButton("eighteen"));
panel3.add(new JLabel("nineteen"));

frame.setVisible(true);
```

Example – Part


```
panel1.add(new JLabel("one"), Borde
panel1.add(new JButton("two"), Bord
panel1.add(new JTextField("three"),

panel2.add(new JButton("four"));
panel2.add(new JLabel("five"));
panel2.add(new JTextField("six"));
panel2.add(new JButton("seven"));
panel2.add(new JLabel("eight"));
panel2.add(new JButton("ninehundred
panel2.add(new JLabel("ten"));

panel3.add(new JButton("eleven"));
panel3.add(new JLabel("twelve"));
panel3.add(new JTextField("thirteen
panel3.add(new JButton("fourteen"))
panel3.add(new JLabel("fifteen"));
panel3.add(new JButton("sixteen"));
panel3.add(new JLabel("seventeen"));
panel3.add(new JButton("eighteen"));
panel3.add(new JLabel("nineteen"));
panel3.add(new JButton("eighteen"));
panel3.add(new JLabel("nineteen"));

frame.setVisible(true);
```



Any Questions ?