CHAPTER 8

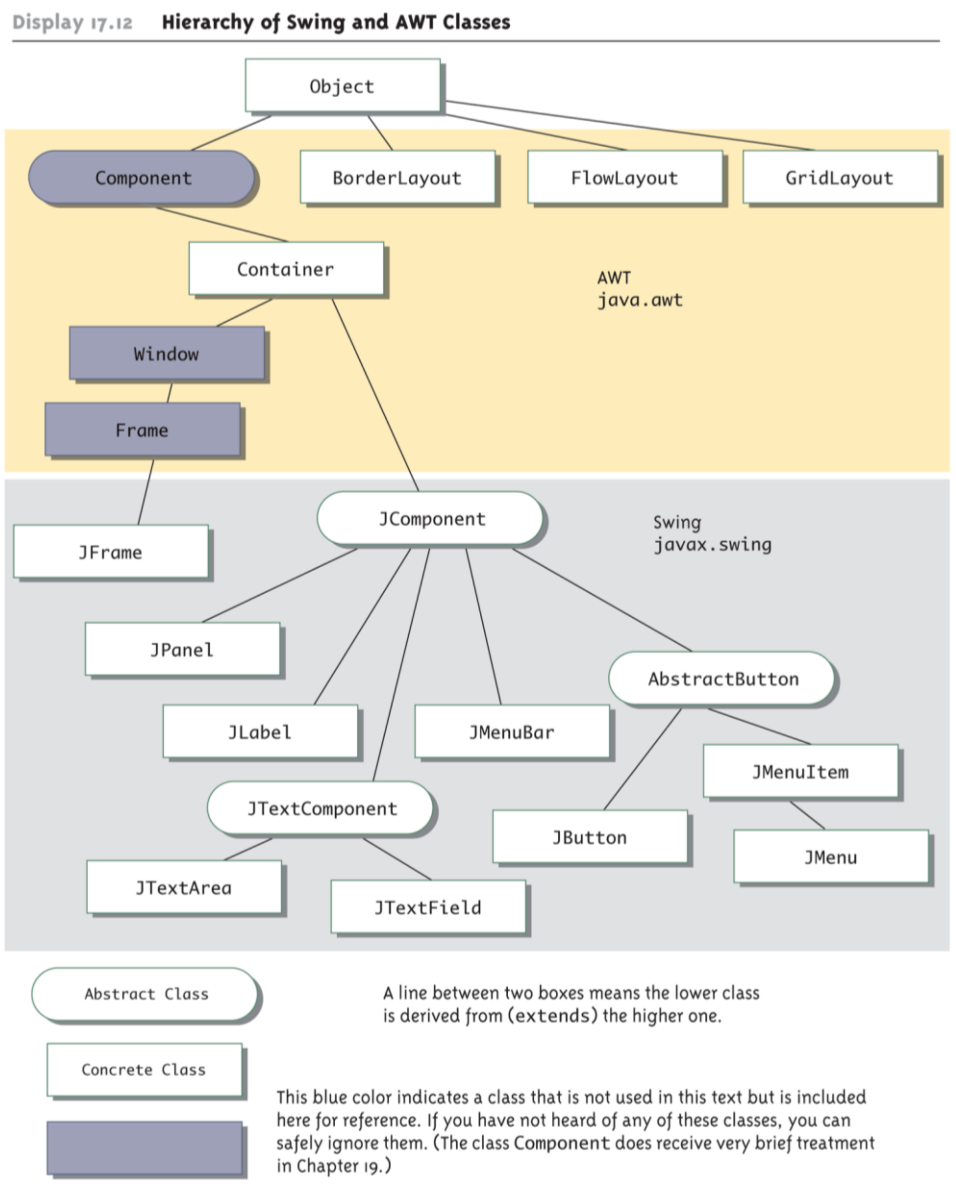
Swing

Shin-Jie Lee (李信杰)
Associate Professor
Computer and Network Center
Department of Computer Science and Information Engineering
National Cheng Kung University

Introduction to Swing

- A *GUI (graphical user interface)* is a windowing system that interacts with the user
- The Java *AWT (Abstract Window Toolkit)* package is the original Java package for creating *GUIs*
- The Swing package is an improved version of the AWT

Hierarchy of Swing and AWT Classes



A line between two boxes means the lower class is derived from (extends) the higher one.

This blue color indicates a class that is not used in this text but is included here for reference. If you have not heard of any of these classes, you can safely ignore them. (The class Component does receive very brief treatment in Chapter 19.)

A Simple Window

- A simple window can consist of an object of the `JFrame` class
  - A `JFrame` object includes a border and the usual three buttons for minimizing, changing the size of, and closing the window
  - The `JFrame` class is found in the `javax.swing` package

    ```
    JFrame firstWindow = new JFrame();
    ```
- A `JFrame` can have components added to it, such as buttons, menus, and text labels
  - These components can be programmed for action

    ```
    firstWindow.add(endButton);
    ```
  - It can be made visible using the `setVisible` method

    ```
    firstWindow.setVisible(true);
    ```

Lab

```java
import javax.swing.JFrame;

public class MyFrame {

  public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setSize(800, 600);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);
  }

}
```

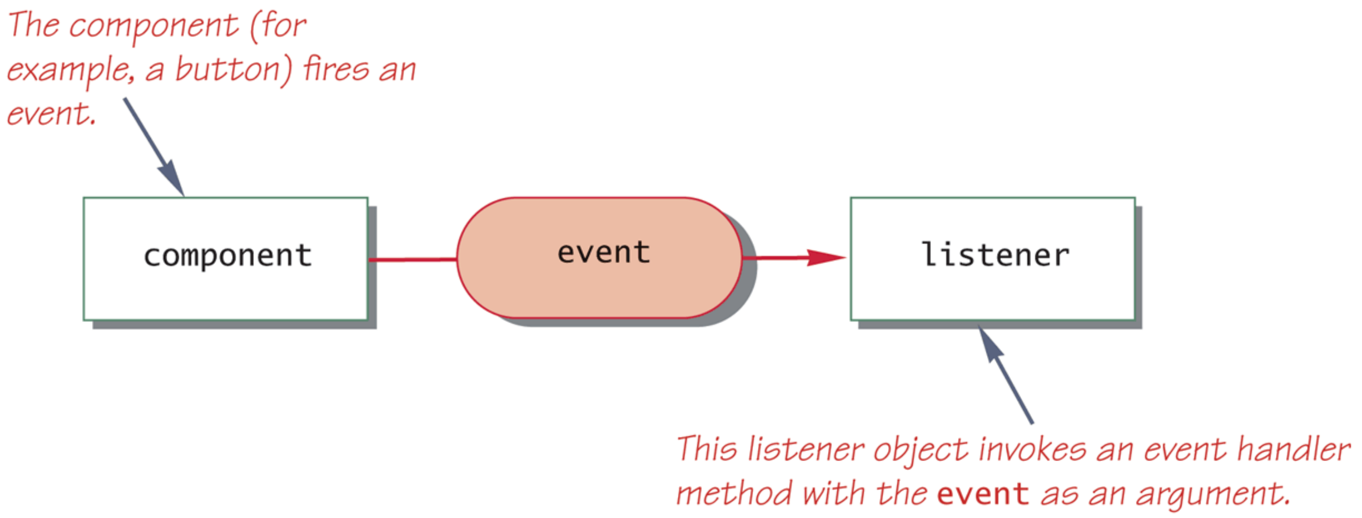
Buttons

- A *button* object is created from the class **`JButton`** and can be added to a **`JFrame`**
  - The argument to the **`JButton`** constructor is the string that appears on the button when it is displayed

```
JButton endButton = new
           JButton("Click to end program.");
firstWindow.add(endButton);
```

Event Firing and an Event Listener

Display 17.1 Event Firing and an Event Listener


The component (for example, a button) fires an event.
component
event
listener
This listener object invokes an event handler method with the event as an argument.

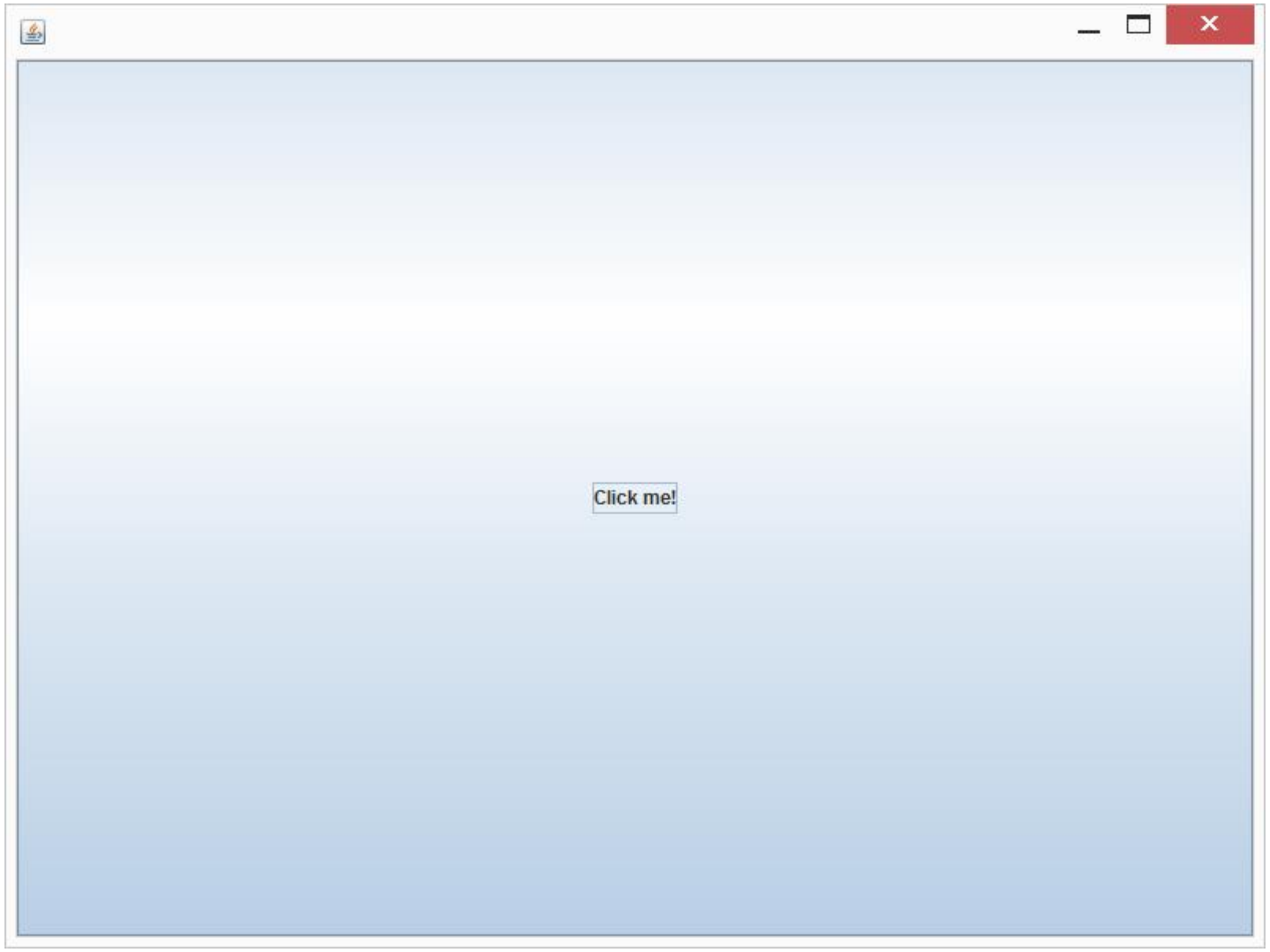
Lab

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class MyButtonListener implements ActionListener{

  public void actionPerformed(ActionEvent e){
    System.out.println(e.getActionCommand());
    System.out.println(e.getSource());
  }

}
```

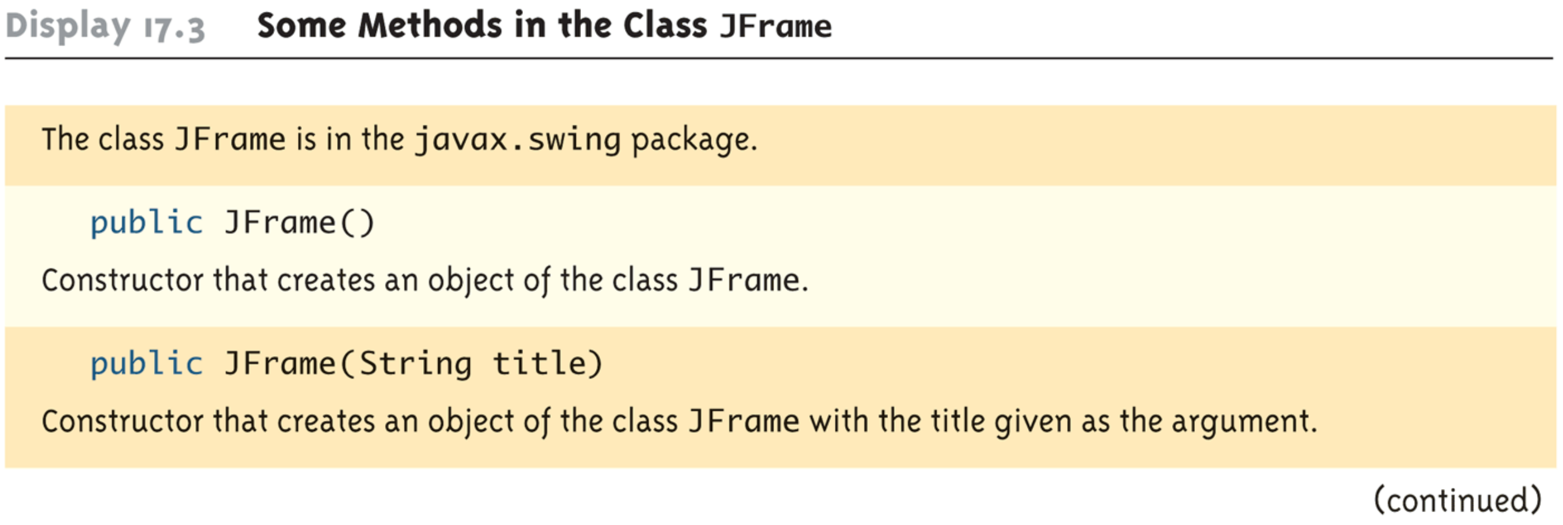
Lab

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

public class MyFrame {

  public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setSize(800, 600);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JButton btn = new JButton("Click me!");
    MyButtonListener mblistener = new MyButtonListener();
    btn.addActionListener(mblistener);
    frame.add(btn);

    frame.setVisible(true);
  }

}
```

Lab



Some Methods in the Class `JFrame` (Part 1 of 3)


Display 17.3 **Some Methods in the Class `JFrame`**

The class `JFrame` is in the `javax.swing` package.

```
public JFrame()
```

Constructor that creates an object of the class `JFrame`.

```
public JFrame(String title)
```

Constructor that creates an object of the class `JFrame` with the title given as the argument.

(continued)

Some Methods in the Class `JFrame` (Part 2 of 3)

Display 17.3 Some Methods in the Class `JFrame`

```
public void setDefaultCloseOperation(int operation)
```

Sets the action that will happen by default when the user clicks the close-window button. The argument should be one of the following defined constants:

`JFrame.DO_NOTHING_ON_CLOSE`: Do nothing. The `JFrame` does nothing, but if there are any registered window listeners, they are invoked. (Window listeners are explained in Chapter 19.)

`JFrame.HIDE_ON_CLOSE`: Hide the frame after invoking any registered `WindowListener` objects.

`JFrame.DISPOSE_ON_CLOSE`: Hide and *dispose* the frame after invoking any registered window listeners. When a window is **disposed** it is eliminated but the program does not end. To end the program, you use the next constant as an argument to `setDefaultCloseOperation`.

`JFrame.EXIT_ON_CLOSE`: Exit the application using the `System exit` method. (Do not use this for frames in applets. Applets are discussed in Chapter 18.)

If no action is specified using the method `setDefaultCloseOperation`, then the default action taken is `JFrame.HIDE_ON_CLOSE`.

Throws an `IllegalArgumentException` if the argument is not one of the values listed above.[2]

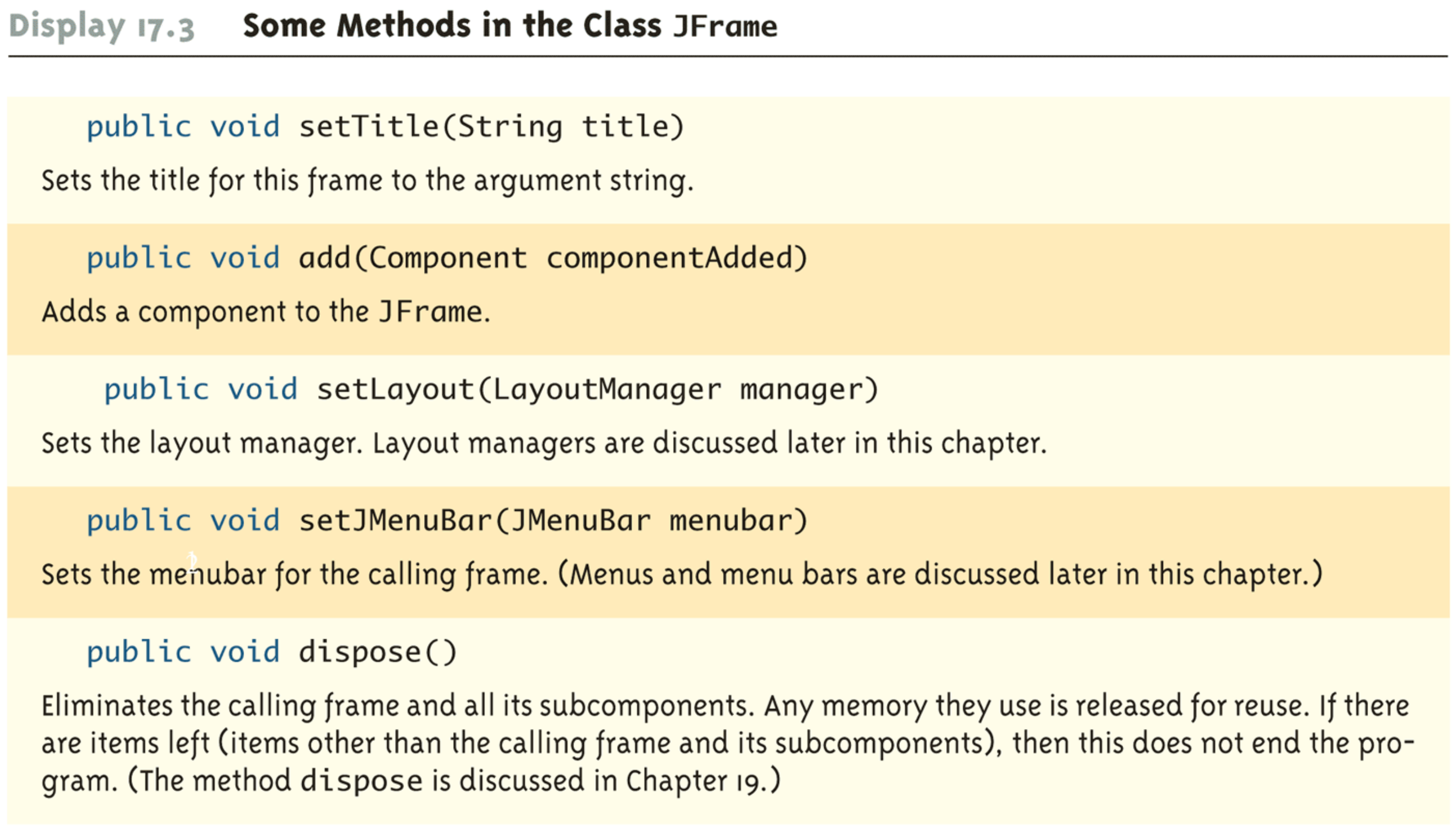
Throws a `SecurityException` if the argument is `JFrame.EXIT_ON_CLOSE` and the Security Manager will not allow the caller to invoke `System.exit`. (You are not likely to encounter this case.)

```
public void setSize(int width, int height)
```

Sets the size of the calling frame so that it has the `width` and `height` specified. Pixels are the units of length used.

(continued)

Some Methods in the Class `JFrame` (Part 3 of 3)



Display 17.3 **Some Methods in the Class `JFrame`**

```
public void setTitle(String title)
```

Sets the title for this frame to the argument string.

```
public void add(Component componentAdded)
```

Adds a component to the `JFrame`.

```
public void setLayout(LayoutManager manager)
```

Sets the layout manager. Layout managers are discussed later in this chapter.

```
public void setJMenuBar(JMenuBar menubar)
```

Sets the menubar for the calling frame. (Menus and menu bars are discussed later in this chapter.)

```
public void dispose()
```

Eliminates the calling frame and all its subcomponents. Any memory they use is released for reuse. If there are items left (items other than the calling frame and its subcomponents), then this does not end the program. (The method `dispose` is discussed in Chapter 19.)

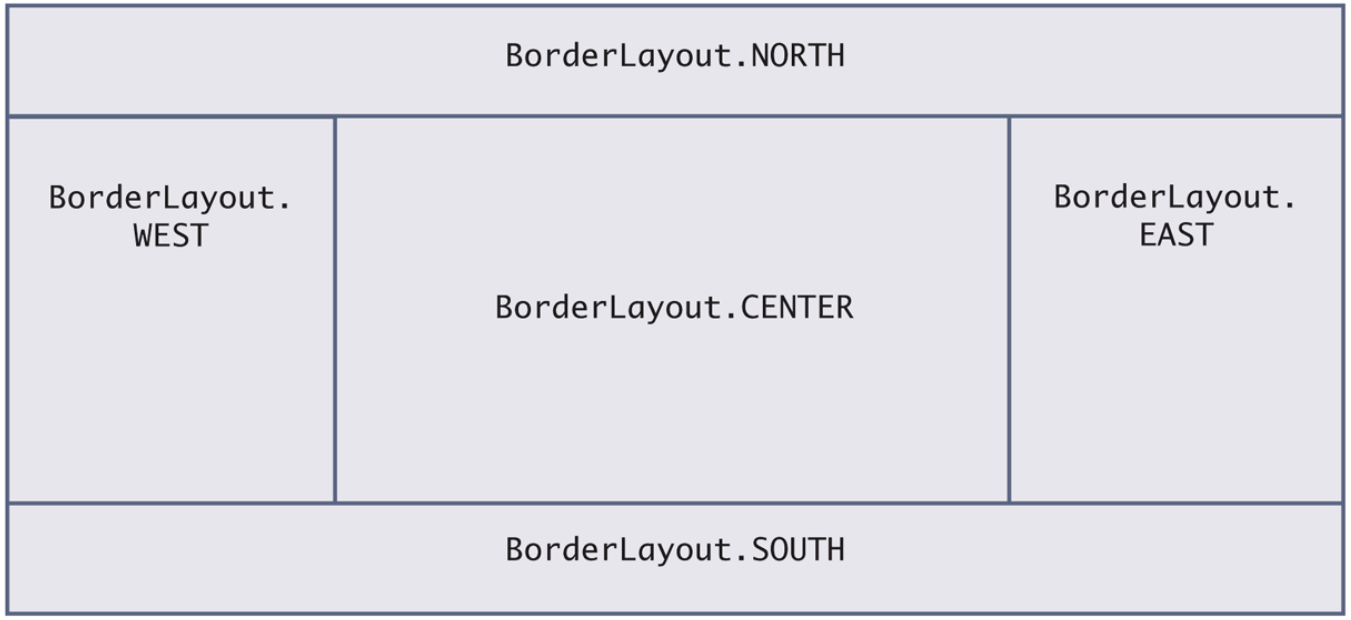
Layout Managers

- There are a number of layout manager classes such as `BorderLayout`, `FlowLayout`, and `GridLayout`
- For example:

```
setLayout(new BorderLayout());
add(label1, BorderLayout.NORTH);
```

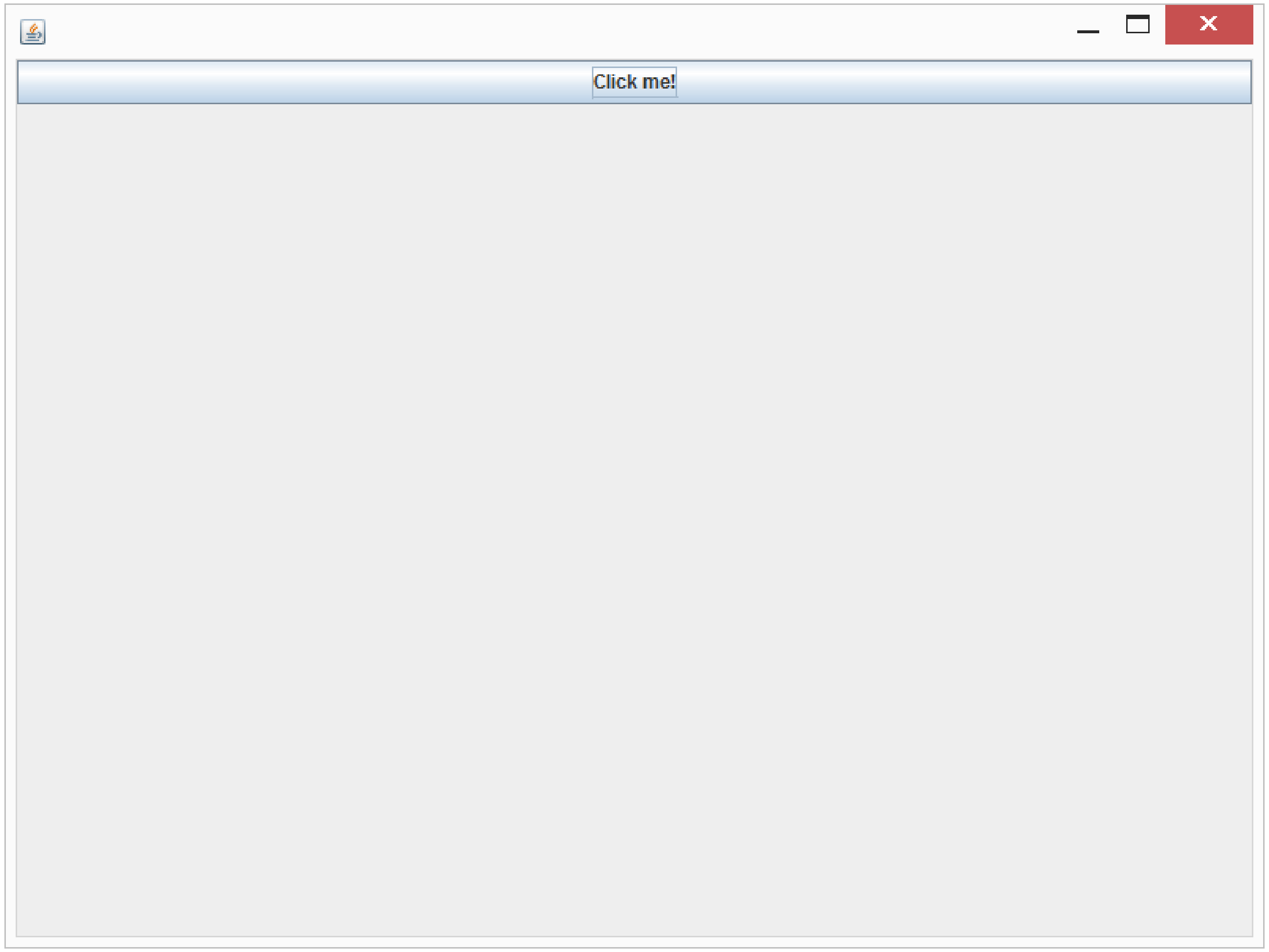
BorderLayout Regions

Display 17.8 **BorderLayout Regions**



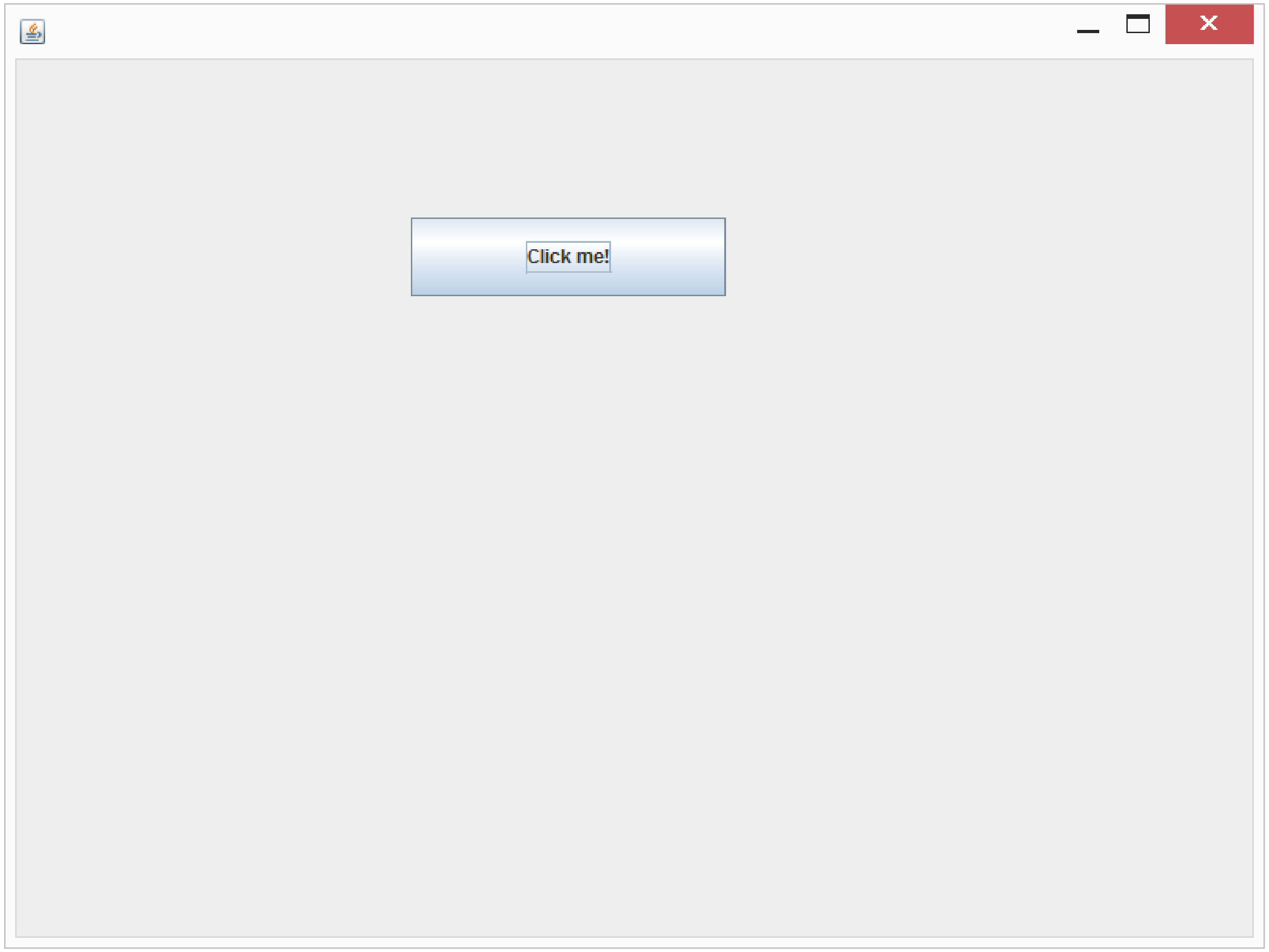
Lab

```java
import java.awt.BorderLayout;

import javax.swing.JButton;
import javax.swing.JFrame;

public class MyFrame {

  public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setSize(800, 600);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JButton btn = new JButton("Click me!");
    MyButtonListener mblistener = new MyButtonListener();
    btn.addActionListener(mblistener);

    frame.setLayout(new BorderLayout());
    frame.add(btn, BorderLayout.NORTH);

    frame.setVisible(true);
}

}
```


Click me!

Lab (No Layout)

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

public class MyFrame {

  public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setSize(800, 600);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JButton btn = new JButton("Click me!");
    MyButtonListener mblistener = new MyButtonListener();
    btn.addActionListener(mblistener);
    btn.setLocation(250, 100);
    btn.setSize(200, 50);

    frame.setLayout(null);
    frame.add(btn);

    frame.setVisible(true);
  }

}
```


Click me!

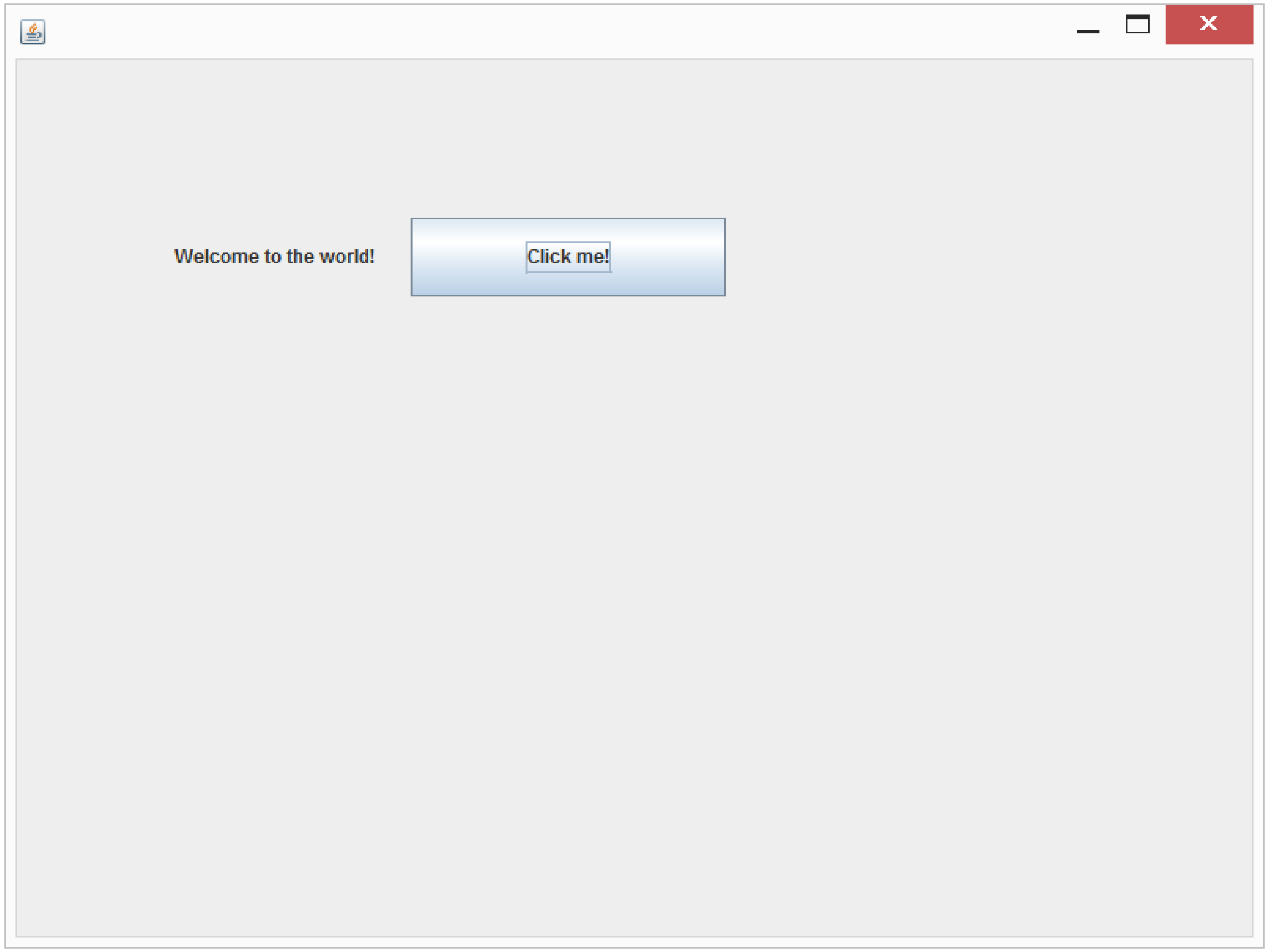
Labels

- A *label* is an object of the class **`JLabel`**

```
JLabel greeting = new JLabel("Hello");
add(greeting);
```

Lab

```
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;

public class MyFrame {

  public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setSize(800, 600);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JButton btn = new JButton("Click me!");
    MyButtonListener mblistener = new MyButtonListener();
    btn.addActionListener(mblistener);
    btn.setLocation(250, 100);
    btn.setSize(200, 50);

    JLabel lb = new JLabel("Welcome to the world!");
    lb.setLocation(100,100);
    lb.setSize(200,50);

    frame.setLayout(null);
    frame.add(btn);
    frame.add(lb);

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


Welcome to the world!
Click me!

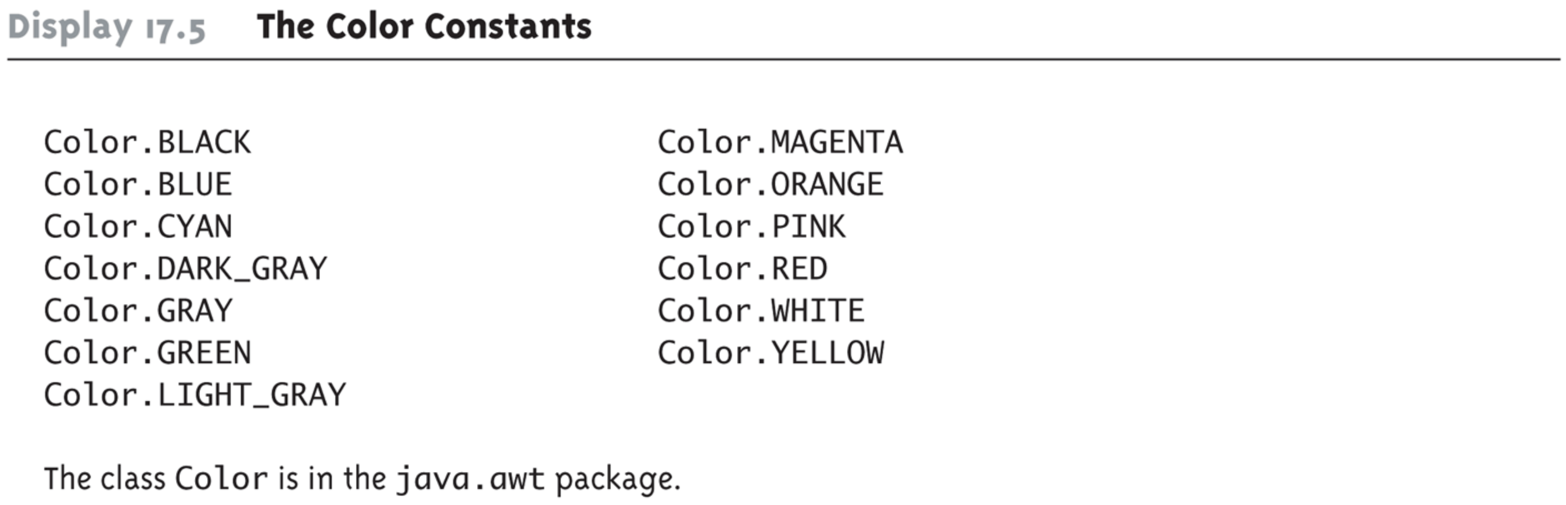
Color

- A *color* is an object of the class `java.awt.Color`

- The background color of a `JFrame` can be set using the following code:

```
getContentPane().setBackground(Color.PINK);
```

The Color Constants



Display 17.5 **The Color Constants**

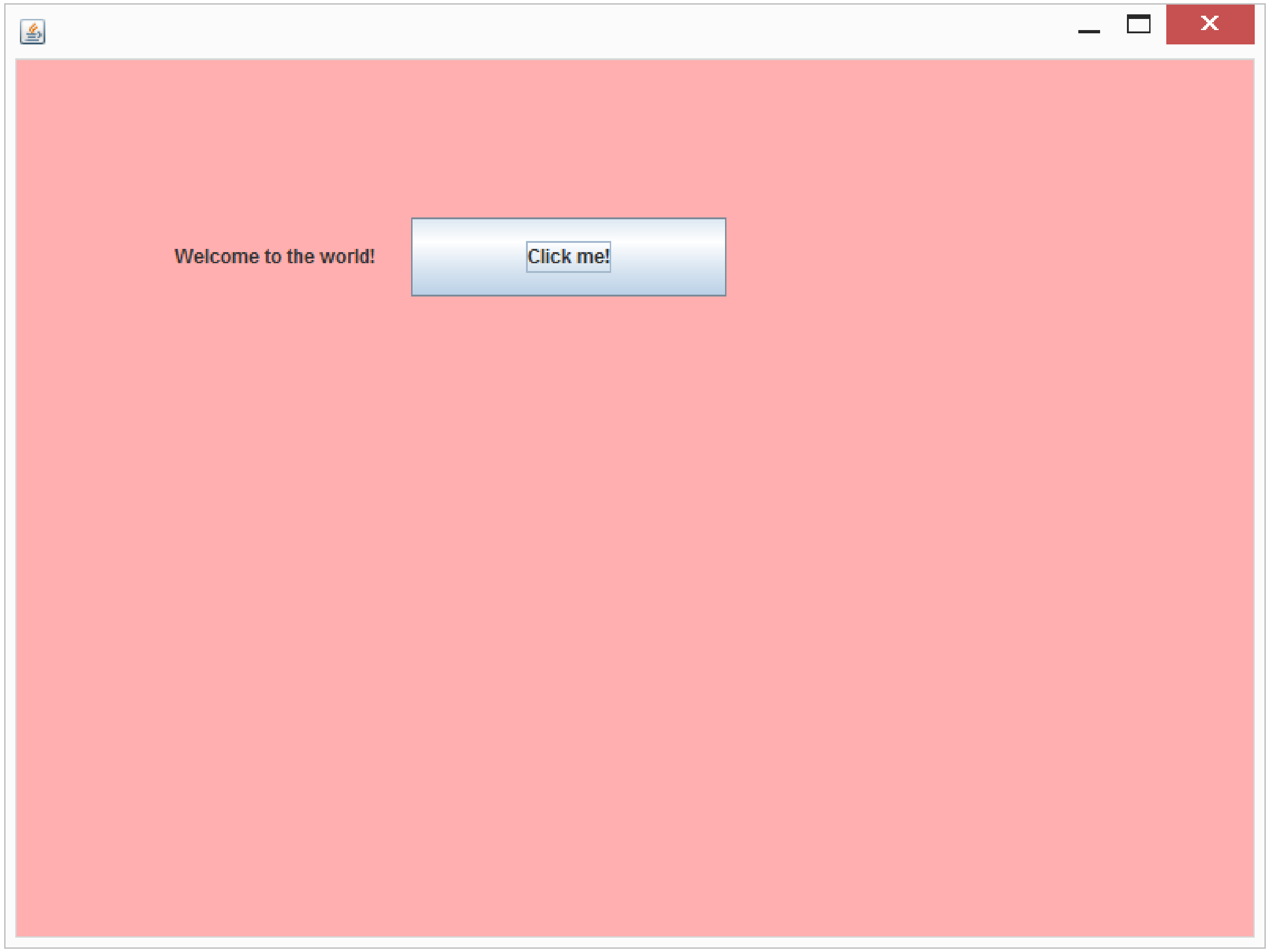
`Color.BLACK`	`Color.MAGENTA`
`Color.BLUE`	`Color.ORANGE`
`Color.CYAN`	`Color.PINK`
`Color.DARK_GRAY`	`Color.RED`
`Color.GRAY`	`Color.WHITE`
`Color.GREEN`	`Color.YELLOW`
`Color.LIGHT_GRAY`	

The class `Color` is in the `java.awt` package.

Lab

```
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;

public class MyFrame {

  public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setSize(800, 600);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JButton btn = new JButton("Click me!");
    MyButtonListener mblistener = new MyButtonListener();
    btn.addActionListener(mblistener);
    btn.setLocation(250, 100);
    btn.setSize(200, 50);

    JLabel lb = new JLabel("Welcome to the world!");
    lb.setLocation(100,100);
    lb.setSize(200,50);

    frame.setLayout(null);
    frame.add(btn);
    frame.add(lb);

    frame.getContentPane().setBackground(Color.PINK);

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

Lab



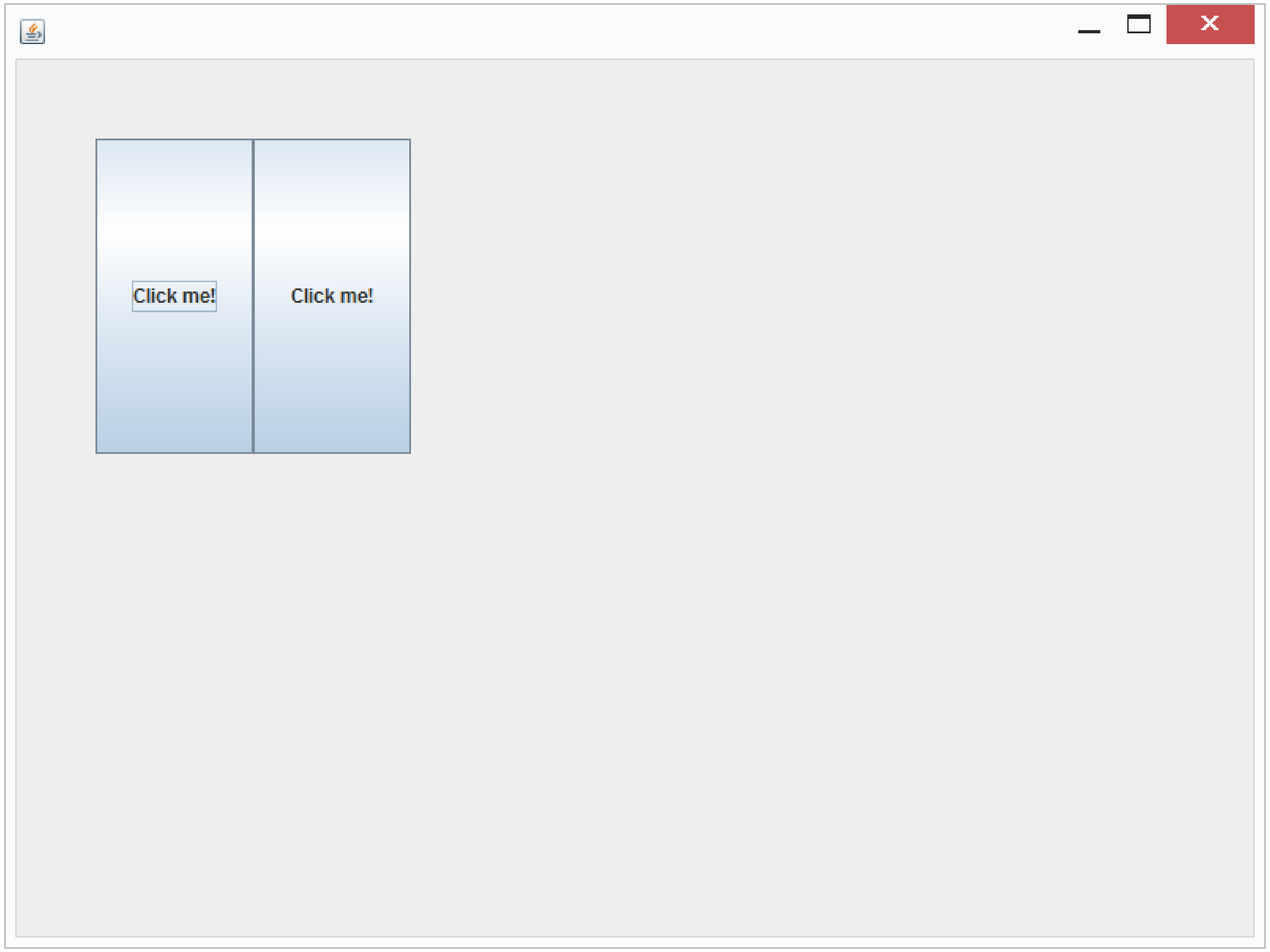
Panels

- A panel is an object of the `JPanel` class that serves as a simple container
  - It is used to **group** smaller objects into a larger component (the panel)

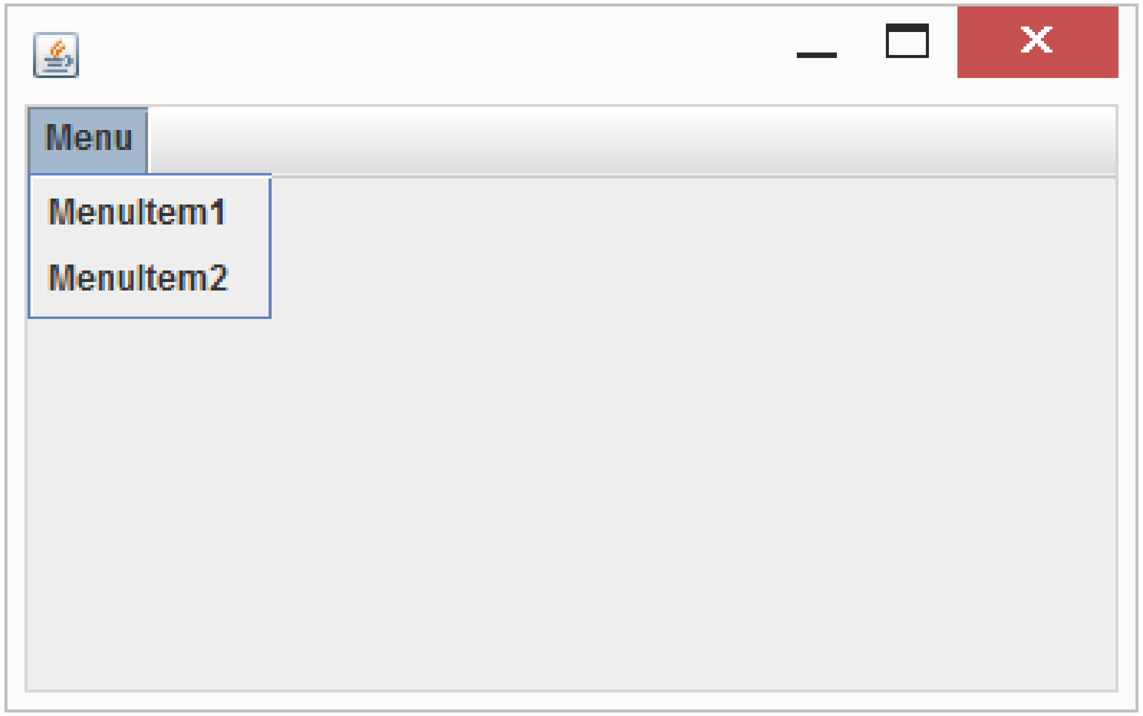
Lab

```java
import java.awt.GridLayout;
import javax.swing.*;

public class PanelTest {

  public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setSize(800, 600);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JButton btn1 = new JButton("Click me!");
    JButton btn2 = new JButton("Click me!");

    JPanel panel = new JPanel();
    panel.setSize(200,200);
    panel.setLayout(new GridLayout());
    panel.add(btn1);
    panel.add(btn2);

    frame.setLayout(null);
    frame.add(panel);

    frame.setVisible(true);

    for(int i=0;i<300;i++){
      panel.setLocation(i, i);
        try{
          Thread.sleep(100);
        }catch(Exception e){
          e.printStackTrace();
      }
    }
  }
}
```

Lab



Menu Bars, Menus, and Menu Items



Lab

```
import javax.swing.JFrame;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;

public class MenuTest {

  public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setSize(800, 600);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JMenu diner = new JMenu("Menu");
    JMenuItem item1 = new JMenuItem("MenuItem1");
    JMenuItem item2 = new JMenuItem("MenuItem2");
    item1.addActionListener(new MyButtonListener());
    item2.addActionListener(new MyButtonListener());
    diner.add(item1);
    diner.add(item2);
    JMenuBar bar = new JMenuBar();
    bar.add(diner);

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

Lab

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class MyButtonListener implements ActionListener{

  public void actionPerformed(ActionEvent e){

    String command = e.getActionCommand();
    if(command.equals("MenuItem1")){
      System.out.println("You pressed menuitem1");
    }else if(command.equals("MenuItem2")){
      System.out.println("You pressed menuitem2");
    }
  }

}
```

Text Fields

- A *text field* is an object of the class **`JTextField`**

```
JTextField name = new
  JTextField(NUMBER_OF_CHAR);
```

- A Swing GUI can read the text in a text field using the **`getText`** method

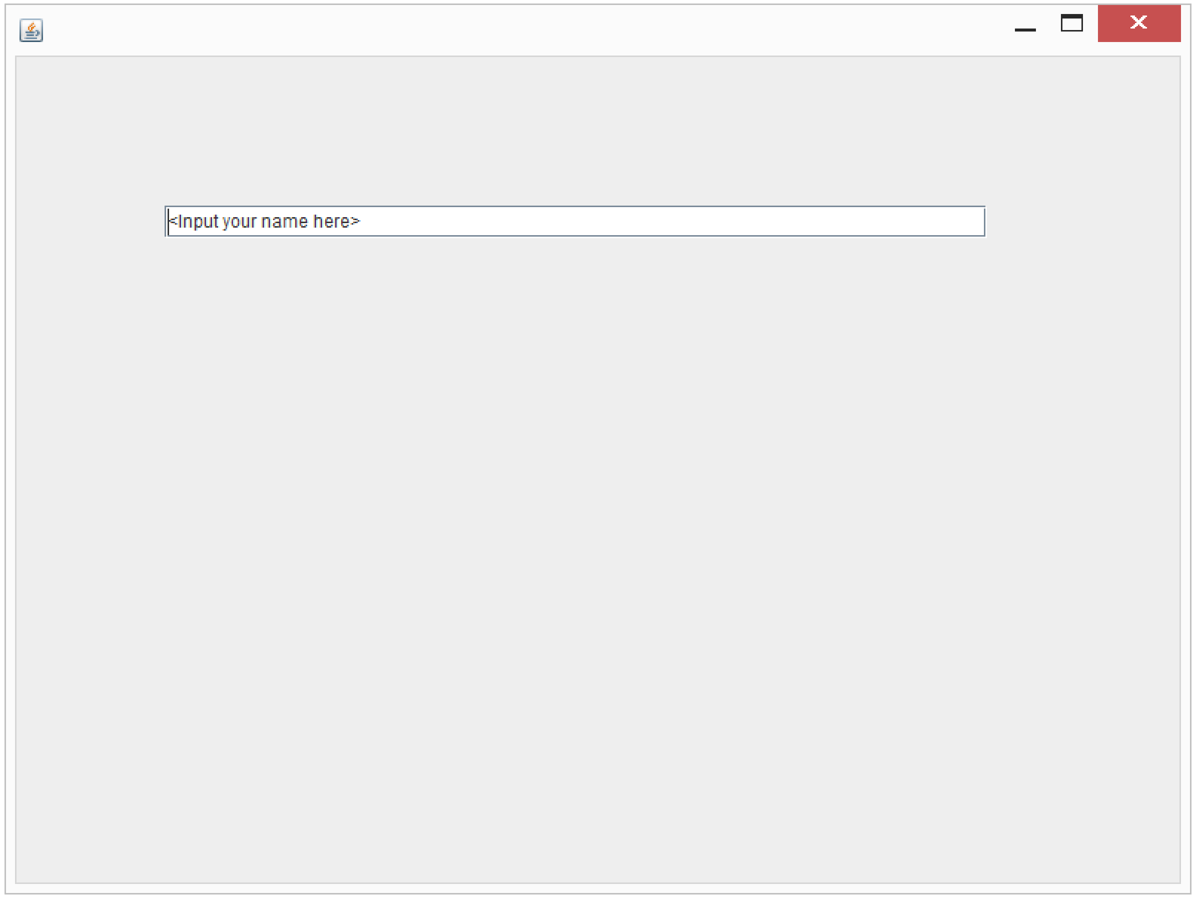
```
String inputString = name.getText();
```

- The method **`setText`** can be used to display a new text string in a text field

```
name.setText("This is some output");
```

Lab

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JFrame;
import javax.swing.JTextField;

public class MyNewFrame extends JFrame implements ActionListener{

  public static void main(String[] args) {
    MyNewFrame frame = new MyNewFrame();
    frame.setVisible(true);
  }

  public MyNewFrame(){
    setSize(800, 600);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setLayout(null);

    JTextField input = new JTextField(50);
    input.setLocation(100,100);
    input.setSize(input.getPreferredSize());
    input.setText("<Input your name here>");
    add(input);
  }

  public void actionPerformed(ActionEvent e){
  }
}
```



Text Areas

- A *text area* is an object of the class **`JTextArea`**

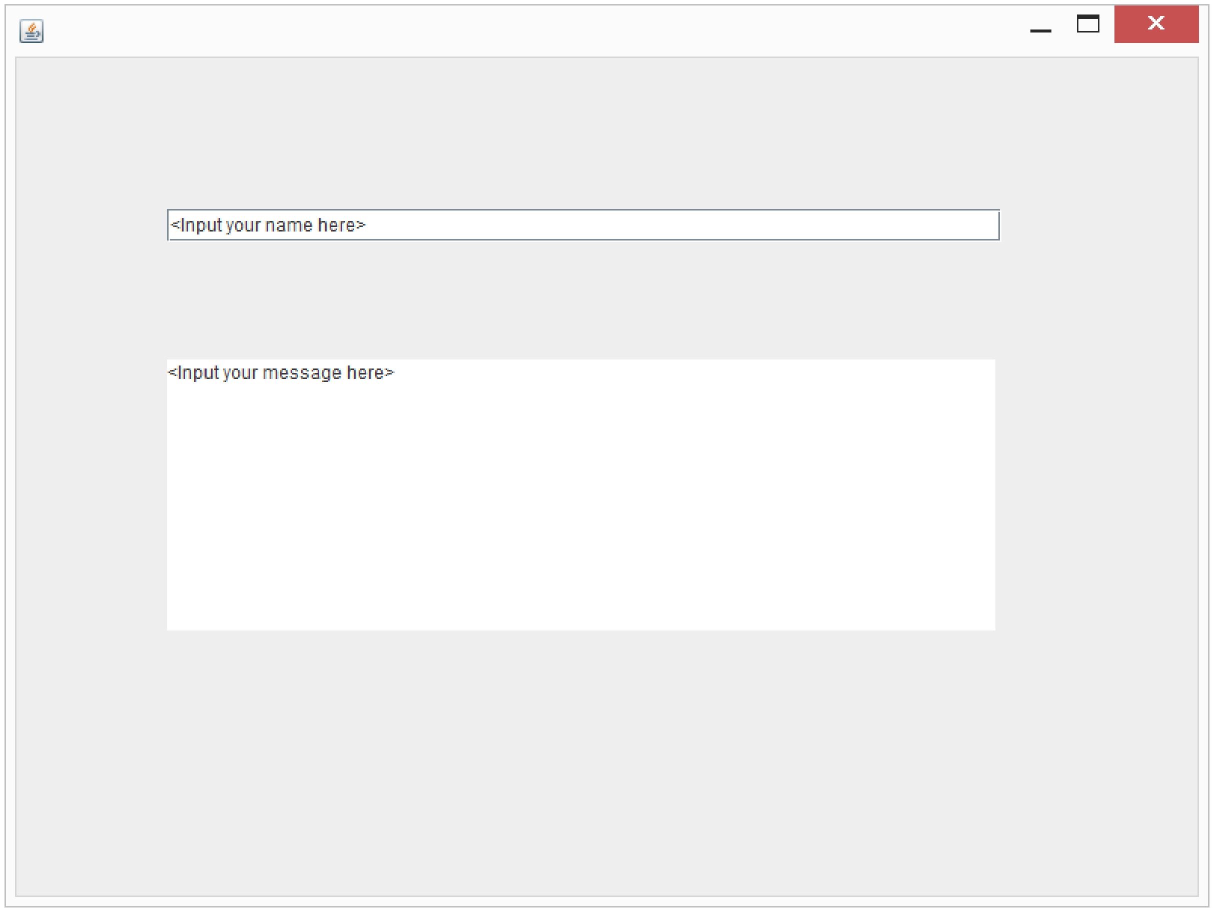
```
JTextArea theText = new JTextArea(5,20);
```

- The line-wrapping policy for a **`JTextArea`** can be set using the method **`setLineWrap`**

```
theText.setLineWrap(true);
```

Lab

```java
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JFrame;
import javax.swing.JTextArea;
import javax.swing.JTextField;

public class MyNewFrame extends JFrame implements ActionListener{
  public static void main(String[] args) {
    MyNewFrame frame = new MyNewFrame();
    frame.setVisible(true);
  }
  public MyNewFrame(){
    setSize(800, 600);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setLayout(null);

    JTextField input = new JTextField(50);
    input.setLocation(100,100);
    input.setSize(input.getPreferredSize());
    input.setText("<Input your name here>");
    add(input);

    JTextArea inputlines = new JTextArea(10,50);
    inputlines.setLocation(100,200);
    inputlines.setSize(inputlines.getPreferredSize());
    inputlines.setText("<Input your message here>");
    add(inputlines);
  }
  public void actionPerformed(ActionEvent e){}
}
```



Reference

- ❑ "Absolute Java". Walter Savitch and Kenrick Mock. Addison-Wesley; 5 edition. 2012
- ❑ "Java How to Program". Paul Deitel and Harvey Deitel. Prentice Hall; 9 edition. 2011.
- ❑ "A Programmers Guide To Java SCJP Certification: A Comprehensive Primer 3rd Edition". Khalid Mughal, Rolf Rasmussen. Addison-Wesley Professional. 2008