



Java™

Object Oriented Design and Programming (OODP)

Week-10 GUI Programming using Swing(Frontend)

LET'S GET STARTED WITH LECTURE 10

Revision Topics

- **Collection Framework**
 - **List**
 - **Queue**
 - **Set**
 - **Map**

Java Swing

- Java Swing is a part of Java Foundation Classes (JFC) that is used to create window-based applications.
- It is platform-independent and light weight components.
- The javax.swing package provides classes for java swing API such as JButton, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser etc.

1.Java Swing

2.Java Swing and AWT Differences

3.Hierarchy of Java Swing

4.Method of component class

5.Ways of frame creation

5.1Jframe inside main method

5.2By association

5.3By inheritance

6.Window Builder

7. Window Builder Installation

8. Creation of empty JFrame

9.Components of Window Builder

9.1 Source View

9.2 Design View

9.2.1 Component Tree

9.2.2 Property Pane

9.2.3 Palette

9.2.3.1 Types of palette

10. Layout Selection

10.1 Absolute Layout

10.2 Flow Layout

10.3 Border Layout

10.4 Card Layout

11. Struts and Spring

12. Components

13.Menu

14.AWT Components

15.Event Handler

15.1 Adding Event Handler

15.2 Mouse Listener

15.3 Mouse Listener API

15.4 Mouse Event Class

16.Example of Login Page

Java Swing and AWT Differences

AWT	Swing
AWT components are heavyweight components	Swing components are lightweight components
AWT doesn't support pluggable look and feel	Swing supports pluggable look and feel
AWT programs are not portable	Swing programs are portable
AWT is old framework for creating GUIs	Swing is new framework for creating GUIs
AWT components require java.awt package	Swing components require javax.swing package
AWT supports limited number of GUI controls	Swing provides advanced GUI controls like Jtable, JTabbedPane etc
More code is needed to implement AWT controls functionality	Less code is needed to implement swing controls functionality
AWT doesn't follow MVC	Swing follows MVC

1.Java Swing

2.Java Swing and AWT Differences

3.Hierarchy of Java Swing

4.Method of component class

5.Ways of frame creation

5.1Jframe inside main method

5.2By association

5.3By inheritance

6.Window Builder

7. Window Builder Installation

8. Creation of empty JFrame

9.Components of Window Builder

9.1 Source View

9.2 Design View

9.2.1 Component Tree

9.2.2 Property Pane

9.2.3 Palette

9.2.3.1 Types of palette

10. Layout Selection

10.1 Absolute Layout

10.2 Flow Layout

10.3 Border Layout

10.4 Card Layout

11. Struts and Spring

12. Components

13.Menu

14.AWT Components

15.Event Handler

15.1 Adding Event Handler

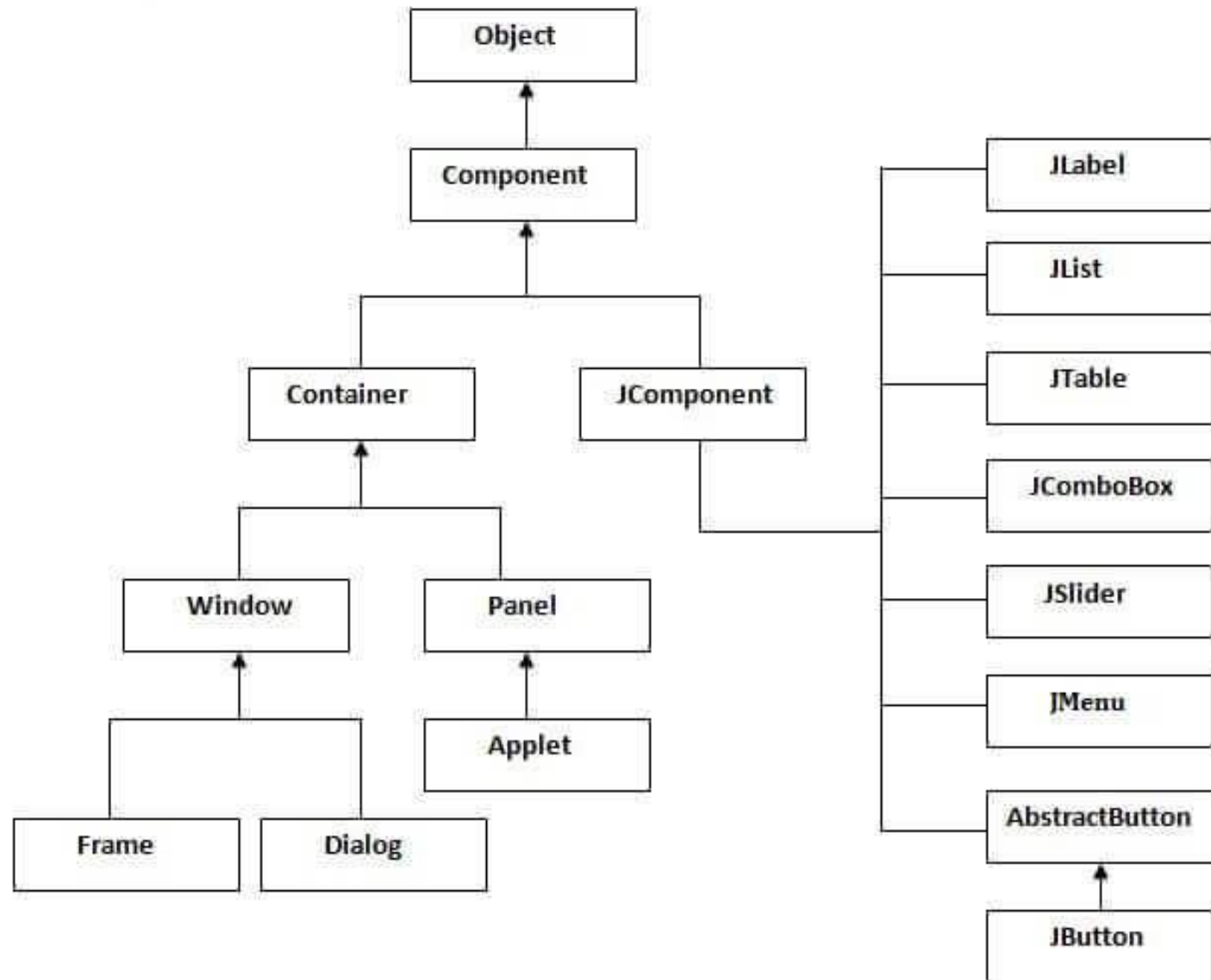
15.2 Mouse Listener

15.3 Mouse Listener API

15.4 Mouse Event Class

16.Example of Login Page

Hierarchy of Java Swing



1. Java Swing
2. Java Swing and AWT Differences
3. Hierarchy of Java Swing
4. Method of component class
5. Ways of frame creation
 - 5.1 JFrame inside main method
 - 5.2 By association
 - 5.3 By inheritance
6. Window Builder
7. Window Builder Installation
8. Creation of empty JFrame
9. Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
11. Grids and Spring
12. Components
13. Menu
14. AWT Components
15. Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class
16. Example of Login Page

Method of component class

Method	Description
<i>public void add(Component c)</i>	<i>add a component on another component.</i>
<i>public void setSize(int width,int height)</i>	<i>sets size of the component.</i>
<i>public void setLayout(LayoutManager m)</i>	<i>sets the layout manager for the component.</i>
<i>public void setVisible(boolean b)</i>	<i>sets the visibility of the component. It is by default false.</i>

- 1.Java Swing
- 2.Java Swing and AWT Differences
- 3.Hierarchy of Java Swing
- 4.Method of component class**
- 5.Ways of frame creation
 - 5.1Jframe inside main method
 - 5.2By association
 - 5.3By inheritance
- 6.Window Builder
- 7. Window Builder Installation
- 8. Creation of empty jframe
- 9.Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13.Menu
- 14.AWT Components
- 15.Event Handler
 - 15.1 Adding Event HAndler
 - 15.2 Mouse Listener**
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class
- 16.Example of Login Page

Ways of frame creation

There are two ways to create a frame:

- By creating the object of Frame class (association).
- By extending Frame class (inheritance).

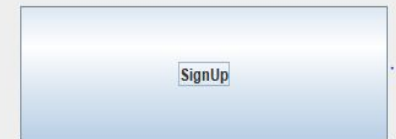
We can write the code of swing inside the main(), constructor or any other method.

By Creating the object of frame class inside main method

- 1. Java Swing
- 2. Java Swing and AWT Differences
- 3. Hierarchy of Java Swing
- 4. Method of component class
- 5. Ways of frame creation
 - 5.1 JFrame inside main method

```
1 package collectionFrameWork;  
2 import javax.swing.*;  
3 public class SignUp {  
4     public static void main(String[] args) {  
5         JFrame f=new JFrame();//creating object of JFrame  
6         JButton b=new JButton("SignUp");//creating object of JButton  
7         b.setBounds(130,100,300,100);//x axis, y axis, width, height  
8         f.add(b);//adding button in JFrame  
9         f.setSize(400,500);//400 width and 500 height  
10        f.setLayout(null);//using no layout manager  
11        f.setVisible(true);//making the frame visible  
12    }  
13 }  
14  
15
```

Output



- 5.2 By association
- 5.3 By inheritance
- 6. Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9. Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Grids and Spring
- 12. Components
- 13. Menu
- 14. AWT Components
- 15. Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class
- 16. Example of Login Page

By Association inside constructor

1. Java Swing
2. Java Swing and AWT Differences
3. Hierarchy of Java Swing
4. Method of component class
5. Ways of frame creation

5.1 JFrame inside main method

5.2 By association

5.3 By inheritance

6. Window Builder

7. Window Builder Installation

8. Creation of empty JFrame

9. Components of Window Builder

9.1 Source View

9.2 Design View

9.2.1 Component Tree

9.2.2 Property Pane

9.2.3 Palette

9.2.3.1 Types of palette

10. Layout Selection

10.1 Absolute Layout

10.2 Flow Layout

10.3 Border Layout

10.4 Card Layout

11. Struts and Spring

12. Components

13. Menu

14. AWT Components

15. Event Handler

15.1 Adding Event Handler

15.2 Mouse Listener

15.3 Mouse Listener API

15.4 Mouse Event Class

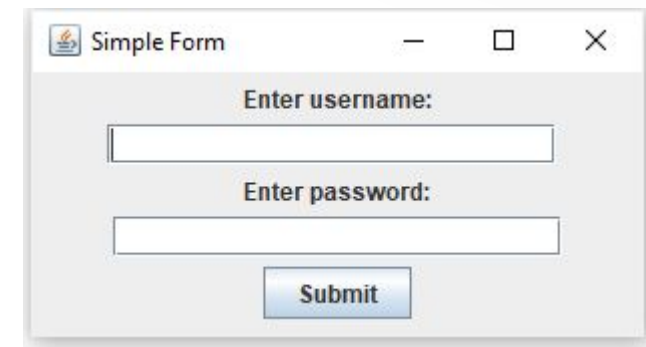
16. Example of Login Page

```

20 import javax.swing.*;
6 public class SimpleForm extends JFrame{
7 public static void main(String[] args) {
8     SimpleForm sf = new SimpleForm();
9     SwingUtilities.invokeLater(new Runnable() {
10         @Override
11         public void run() {
12             new SimpleForm().setVisible(true);
13         }
14     });
15 }
16 private JTextField textField;
17 private JTextField textField1;
18 private JButton submitButton;
19 public SimpleForm() {
20     // Set the title of the frame
21     super("Simple Form");
22     // Create components
23     textField = new JTextField(20);
24     textField1 = new JTextField(20);
25     submitButton = new JButton("Submit");
26     // Set layout manager
27     setLayout(new FlowLayout());
28     // Add components to the frame
29     add(new JLabel("Enter username:"));
30     add(textField);
31     add(new JLabel());
32     add(new JLabel("Enter password:"));
33     add(textField1);
34     add(submitButton);
35     // Add ActionListener to the button
36     submitButton.addActionListener(new ActionListener() {
37         @Override
38         public void actionPerformed(ActionEvent e) {
39             // Handle button click event
40             String inputText = textField.getText();
41             JOptionPane.showMessageDialog(SimpleForm.this, "You entered: " + inputText);
42         }
43     });
44     // Set default close operation and size
45     setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
46     setSize(320, 170);
47     setLocationRelativeTo(null); // Center the frame on the screen
48 }
49 }

```

Output



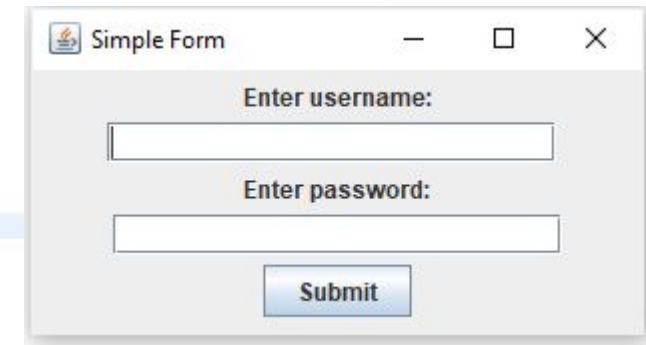
By Inheritance

```

6 public class SimpleForm extends JFrame{
7 public static void main(String[] args) {
15     private JTextField textField;
16     private JTextField textField1;
17     private JButton submitButton;
18
19     public SimpleForm() {
20         // Set the title of the frame
21         super("Simple Form");
22
23         // Create components
24         textField = new JTextField(20);
25         textField1 = new JTextField(20);
26         submitButton = new JButton("Submit");
27
28         // Set layout manager
29         setLayout(new FlowLayout());
30
31         // Add components to the frame
32         add(new JLabel("Enter username:"));
33         add(textField);
34         add(new JLabel());
35         add(new JLabel("Enter password:"));
36         add(textField1);
37         add(submitButton);
38
39         // Add ActionListener to the button
40         submitButton.addActionListener(new ActionListener() {
41             @Override
42             public void actionPerformed(ActionEvent e) {
43                 // Handle button click event
44                 String inputText = textField.getText();
45                 JOptionPane.showMessageDialog(SimpleForm.this, "You entered: " + inputText);
46             }
47         });
48
49         // Set default close operation and size
50         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
51         setSize(320, 170);
52         setLocationRelativeTo(null); // Center the frame on the screen
53     }
54 }
55

```

Output



- 1. Java Swing
- 2. Java Swing and AWT Differences
- 3. Hierarchy of Java Swing
- 4. Method of component class
- 5. Ways of frame creation
 - 5.1 JFrame inside main method
 - 5.2 By association
 - 5.3 By inheritance
- 6. Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9. Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13. Menu
- 14. AWT Components
- 15. Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class
- 16. Example of Login Page

Window Builder

- WindowBuilder is a powerful and easy to use bi-directional Java GUI designer that makes it very easy to create Java GUI applications without spending a lot of time writing code to display simple forms.
- With WindowBuilder we can create complicated windows in minutes.

Window Builder

- WindowBuilder is built as a plug-in to **Eclipse** and the various Eclipse-based IDEs (RAD, RSA, MyEclipse, JBuilder, etc.).
- The plug-in builds an abstract syntax tree (AST) to navigate the source code and uses **Graphical Editing Form (GEF)** to display and manage the visual presentation.

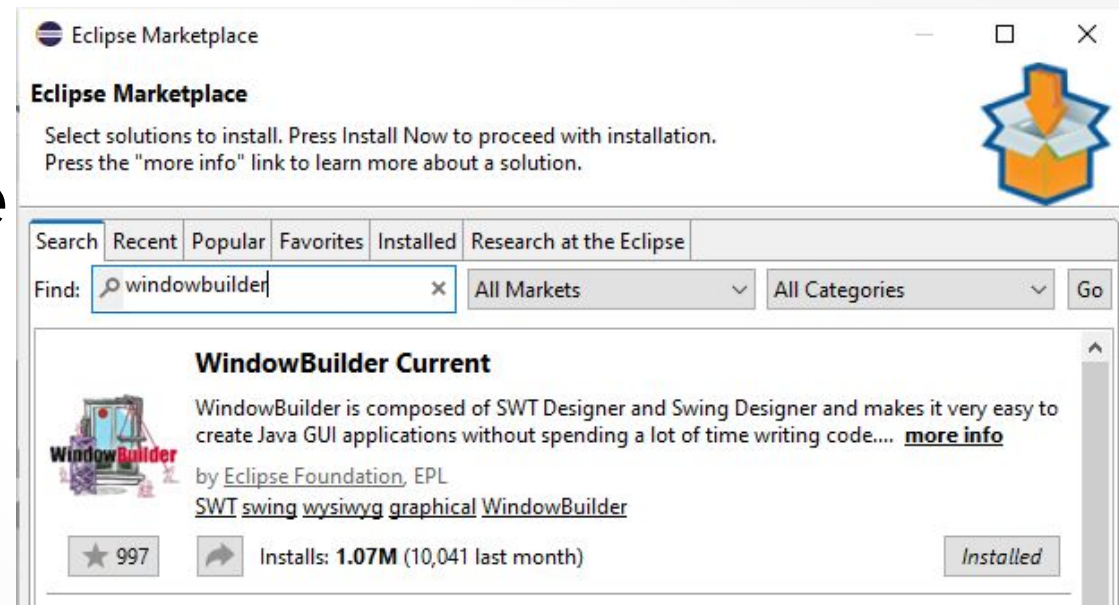
- 1.Java Swing
- 2.Java Swing and AWT Differences
- 3.Hierarchy of Java Swing
- 4.Method of component class
- 5.Ways of frame creation
 - 5.1Jframe inside main method
 - 5.2By association
 - 5.3By inheritance
- 6.Window Builder**
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9.Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Statics and Spring
- 12. Components
- 13.Menu
- 14.AWT Components
- 15.Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener**
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class
- 16.Example of Login Page

Window Builder Installation steps

1. Go to **help** option in eclipse.

2. Go to the **Eclipse Marketplace**

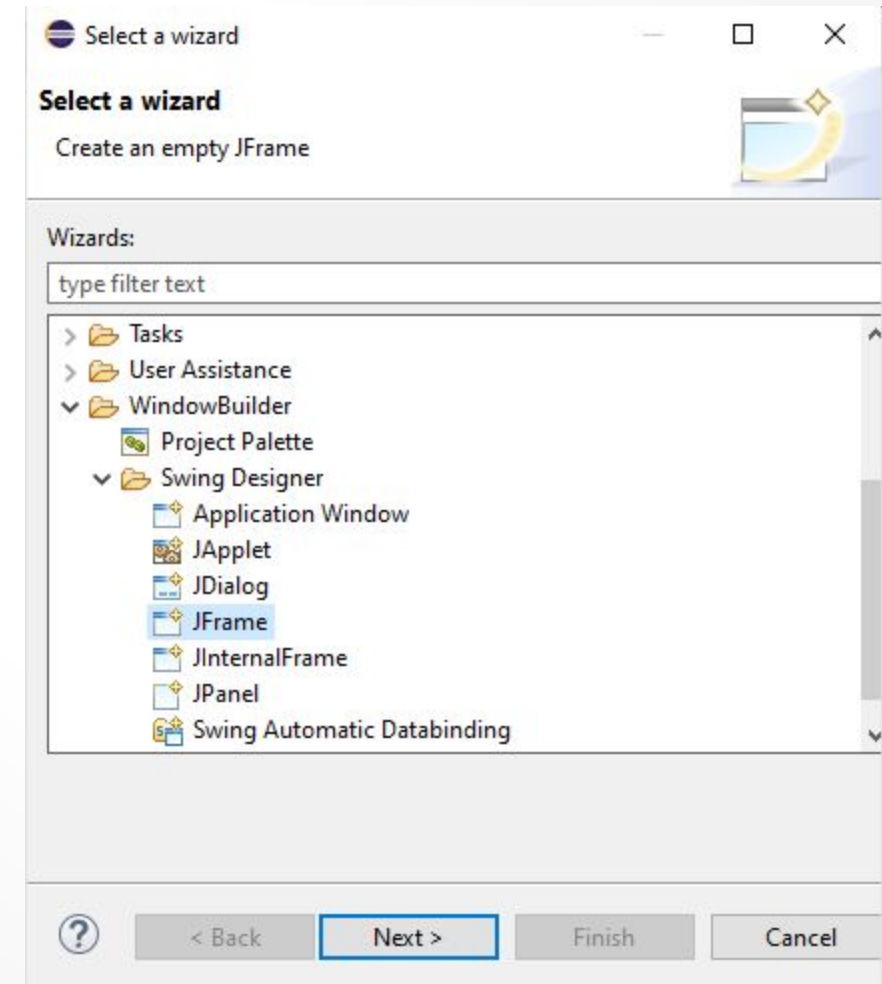
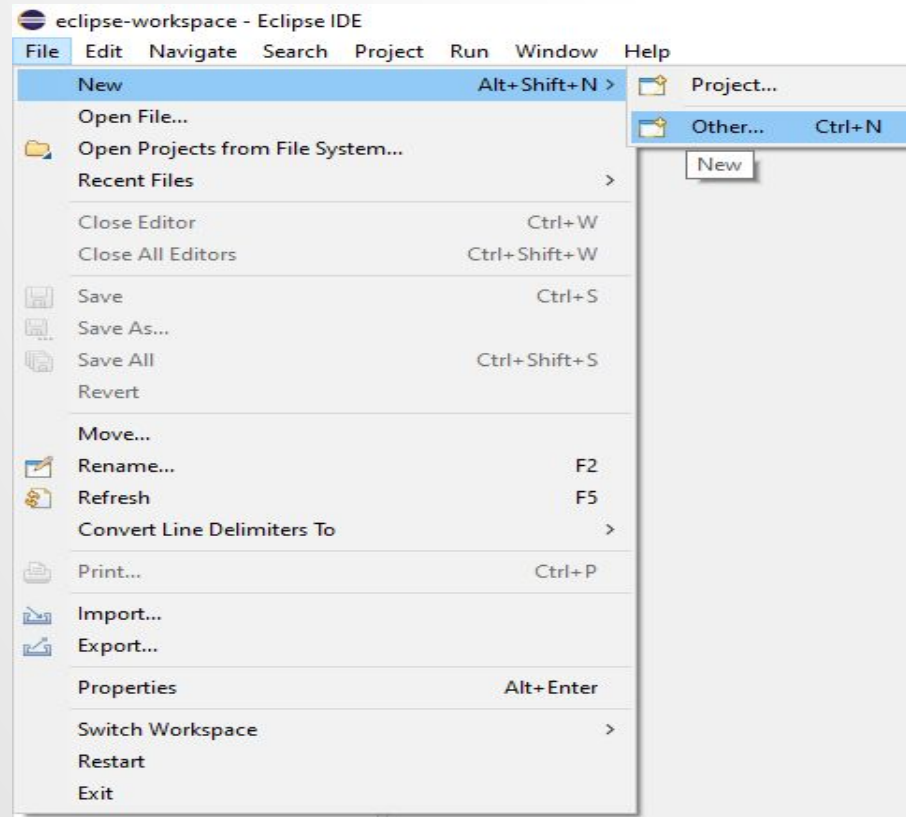
3. Search **windowBuilder** and install.



- 1.Java Swing
- 2.Java Swing and AWT Differences
- 3.Hierarchy of Java Swing
- 4.Method of component class
- 5.Ways of frame creation
 - 5.1Jframe inside main method
 - 5.2By association
 - 5.3By inheritance
- 6.Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9.Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13.Menu
- 14.AWT Components
- 15.Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class
- 16.Example of Login Page

1. Create an empty JFrame

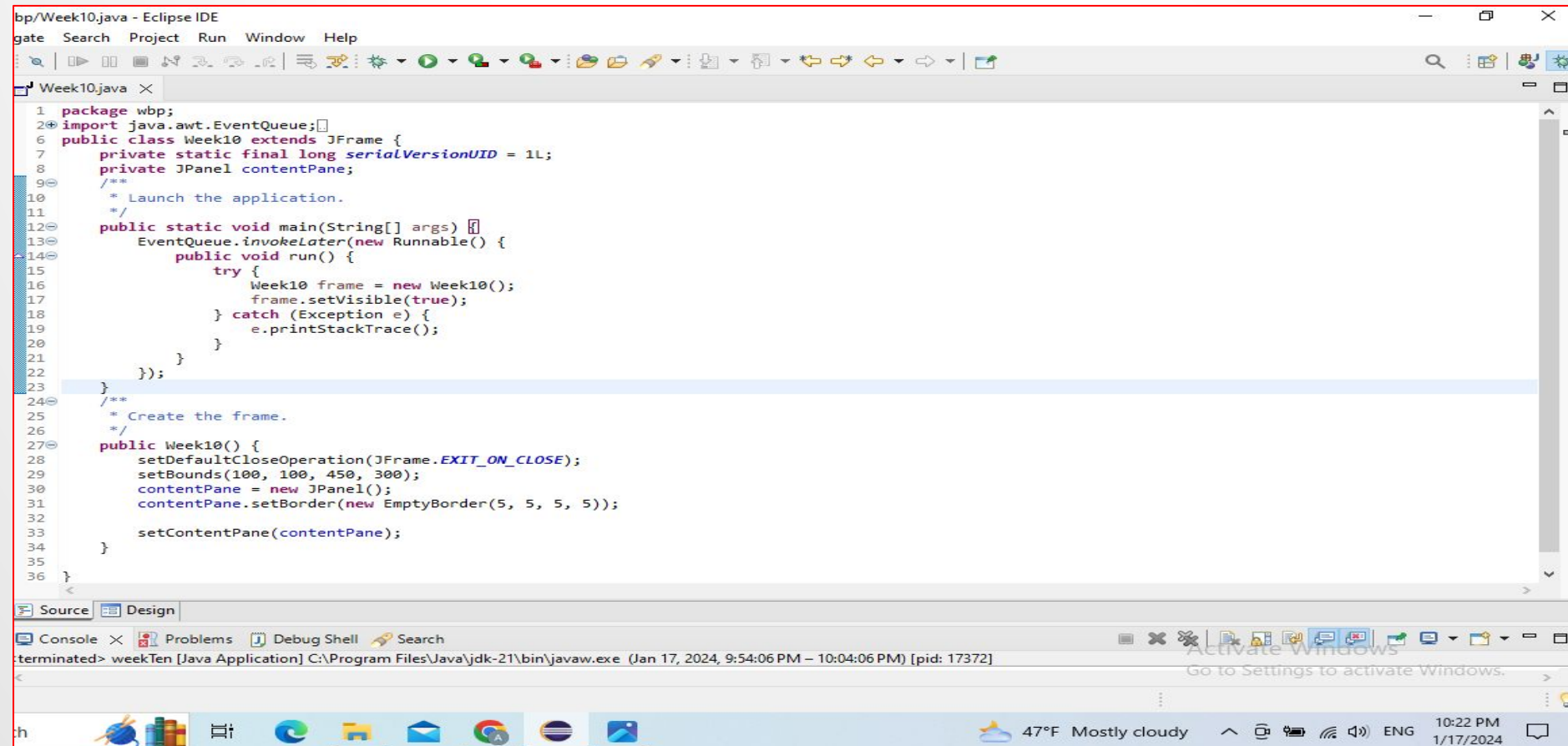
- Go to **File->New->Other->Swing Designer->JFrame**



- 1. Java Swing
- 2. Java Swing and AWT Differences
- 3. Hierarchy of Java Swing
- 4. Method of component class
- 5. Ways of frame creation
 - 5.1 JFrame inside main method
 - 5.2 By association
 - 5.3 By inheritance
- 6. Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9. Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13. Menu
- 14. AWT Components
- 15. Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class
- 16. Example of Login Page

Components of window builder

- **Source View:** Embedded version of Eclipse Java editor provides all of the services we normally find in the Java editor.



```

bp/Week10.java - Eclipse IDE
gate Search Project Run Window Help
Week10.java
1 package wbp;
2 import java.awt.EventQueue;
3
4 public class Week10 extends JFrame {
5     private static final long serialVersionUID = 1L;
6     private JPanel contentPane;
7
8     /**
9      * Launch the application.
10     */
11     public static void main(String[] args) {
12         EventQueue.invokeLater(new Runnable() {
13             public void run() {
14                 try {
15                     Week10 frame = new Week10();
16                     frame.setVisible(true);
17                 } catch (Exception e) {
18                     e.printStackTrace();
19                 }
20             }
21         });
22     }
23
24     /**
25      * Create the frame.
26      */
27     public Week10() {
28         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
29         setBounds(100, 100, 450, 300);
30         contentPane = new JPanel();
31         contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
32
33         setContentPane(contentPane);
34     }
35 }
36

```

Source Design

Console Problems Debug Shell Search

terminated> weekTen [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (Jan 17, 2024, 9:54:06 PM – 10:04:06 PM) [pid: 17372]

47°F Mostly cloudy 10:22 PM 1/17/2024

- 1. Java Swing
- 2. Java Swing and AWT Differences
- 3. Hierarchy of Java Swing
- 4. Method of component class
- 5. Ways of frame creation
 - 5.1 JFrame inside main method
 - 5.2 By association
 - 5.3 By inheritance
- 6. Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9. Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
 - 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
 - 11. Struts and Spring
 - 12. Components
 - 13. Menu
 - 14. AWT Components
 - 15. Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class
- 16. Example of Login Page

Components of window builder

- **Design View**

The Design View is our virtual sketchpad.

We can add or remove components, edit layout properties, direct edit labels, and see what your overall design looks like as it evolves.

- Component Tree
- Property Pane
- Palette
- Wizards
- Toolbars & Context Menus

1.Java Swing

2.Java Swing and AWT Differences

3.Hierarchy of Java Swing

4.Method of component class

5.Ways of frame creation

5.1Jframe inside main method

5.2By association

5.3By inheritance

6.Window Builder

7. Window Builder Installation

8. Creation of empty JFrame

9.Components of Window Builder

9.1 Source View

9.2 Design View

9.2.1 Component Tree

9.2.2 Property Pane

9.2.3 Palette

9.2.3.1 Types of palette

10. Layout Selection

10.1 Absolute Layout

10.2 Flow Layout

10.3 Border Layout

10.4 Card Layout

11. Grids and Spring

12. Components

13.Menu

14.AWT Components

15.Event Handler

15.1 Adding Event Handler

15.2 Mouse Listener

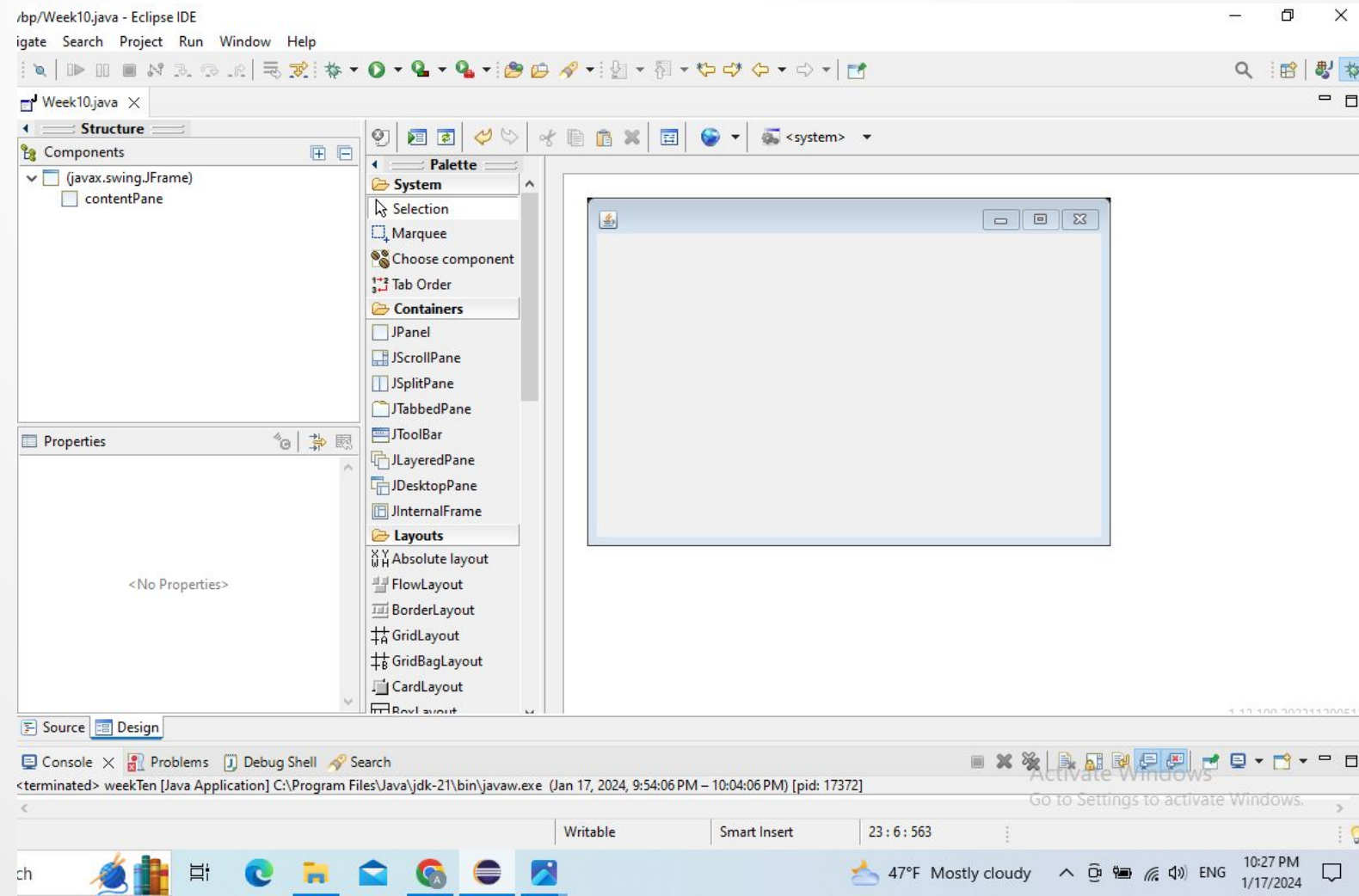
15.3 Mouse Listener API

15.4 Mouse Event Class

16.Example of Login Page

Design View

- Design View



1.Java Swing

2.Java Swing and AWT Differences

3.Hierarchy of Java Swing

4.Method of component class

5.Ways of frame creation

5.1Jframe inside main method

5.2By association

5.3By inheritance

6.Window Builder

7. Window Builder Installation

8. Creation of empty JFrame

9.Components of Window Builder

9.1 Source View

9.2 Design View

9.2.1 Component Tree

9.2.2 Property Pane

9.2.3 Palette

9.2.3.1 Types of palette

10. Layout Selection

10.1 Absolute Layout

10.2 Flow Layout

10.3 Border Layout

10.4 Card Layout

11. Structs and Spring

12. Components

13.Menu

14.AWT Components

15.Event Handler

15.1 Adding Event Handler

15.2 Mouse Listener

15.3 Mouse Listener API

15.4 Mouse Event Class

16.Example of Login Page

Design View

1. Java Swing
2. Java Swing and AWT Differences
3. Hierarchy of Java Swing
4. Method of component class
5. Ways of frame creation

- 5.1 Jframe inside main method
- 5.2 By association

5.3 By inheritance

6. Window Builder

7. Window Builder Installation

8. Creation of empty JFrame

9. Components of Window Builder

9.1 Source View

9.2 Design View

- 9.2.1 Component Tree
- 9.2.2 Property Pane
- 9.2.3 Palette
 - 9.2.3.1 Types of palette

10. Layout Selection

- 10.1 Absolute Layout
- 10.2 Flow Layout
- 10.3 Border Layout
- 10.4 Card Layout

11. Struts and Spring

12. Components

13. Menu

14. AWT Components

15. Event Handler

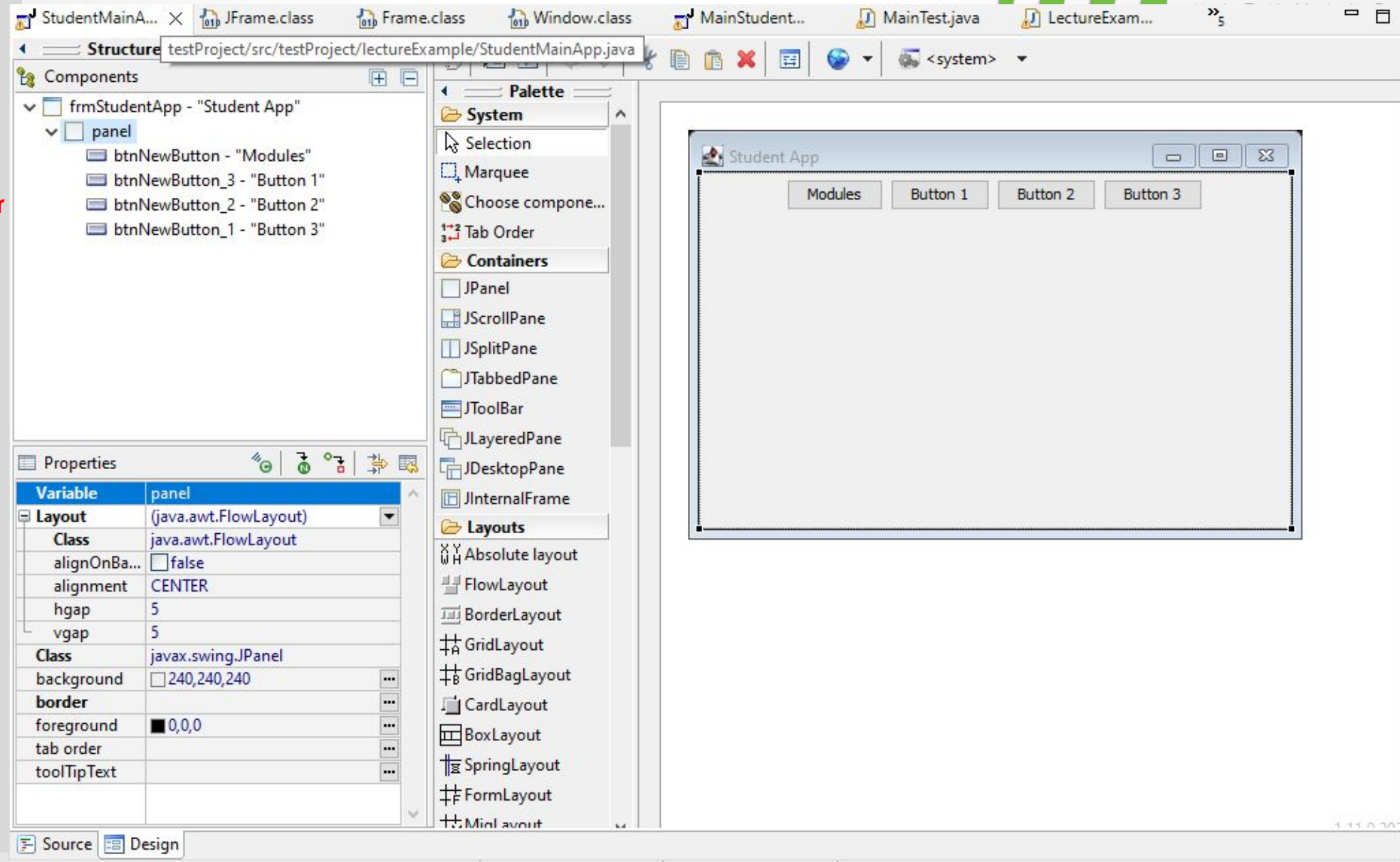
15.1 Adding Event HANDler

15.2 Mouse Listener

15.3 Mouse Listener API

15.4 Mouse Event Class

16. Example of Login Page



(0,0) – your laptop screen top left corner..

Design View

(100,100)



Student App

iconImage

*height of the
Jframe = 300*

y-axis

width of the Jframe = 450

```
JFrame frmStudentApp = new JFrame();  
frmStudentApp.setTitle("Student App");  
frmStudentApp.setBounds(100, 100, 450, 300);  
frmStudentApp.setVisible(true);  
frmStudentApp.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

No Application name

There is nothing in this app (it is empty)

Now we will make this app visually appealing by setting the properties(think this as CSS) of JFrame object.

All GUI Swing Elements have this similar kind of properties.

Properties	
Variable	frame
Class	javax.swing.JFrame
alwaysOnTop	<input type="checkbox"/> false
autoRequest...	<input checked="" type="checkbox"/> true
background	<input type="checkbox"/> 240,240,240 ...
defaultClose...	EXIT_ON_CLOSE
enabled	<input checked="" type="checkbox"/> true
font	...
foreground	...
iconImage	...
modalExclusi...	NO_EXCLUDE
opacity	1.0
resizable	<input checked="" type="checkbox"/> true
tab order	...
title	Student App ...
type	NORMAL

JFrame object's data = Properties

- 1.Java Swing
- 2.Java Swing and AWT Differences
- 3.Hierarchy of Java Swing
- 4.Method of component class
- 5.Ways of frame creation

- 5.1Jframe inside main method
- 5.2By association

5.3By inheritance

6.Window Builder

7. Window Builder Installation

8. Creation of empty JFrame

9.Components of Window Builder

9.1 Source View

9.2 Design View

- 9.2.1 Component Tree
- 9.2.2 Property Pane
- 9.2.3 Palette
 - 9.2.3.1 Types of palette

10. Layout Selection

- 10.1 Absolute Layout
- 10.2 Flow Layout
- 10.3 Border Layout
- 10.4 Card Layout

11. Struts and Spring

12. Components

13.Menu

14.AWT Components

15.Event Handler

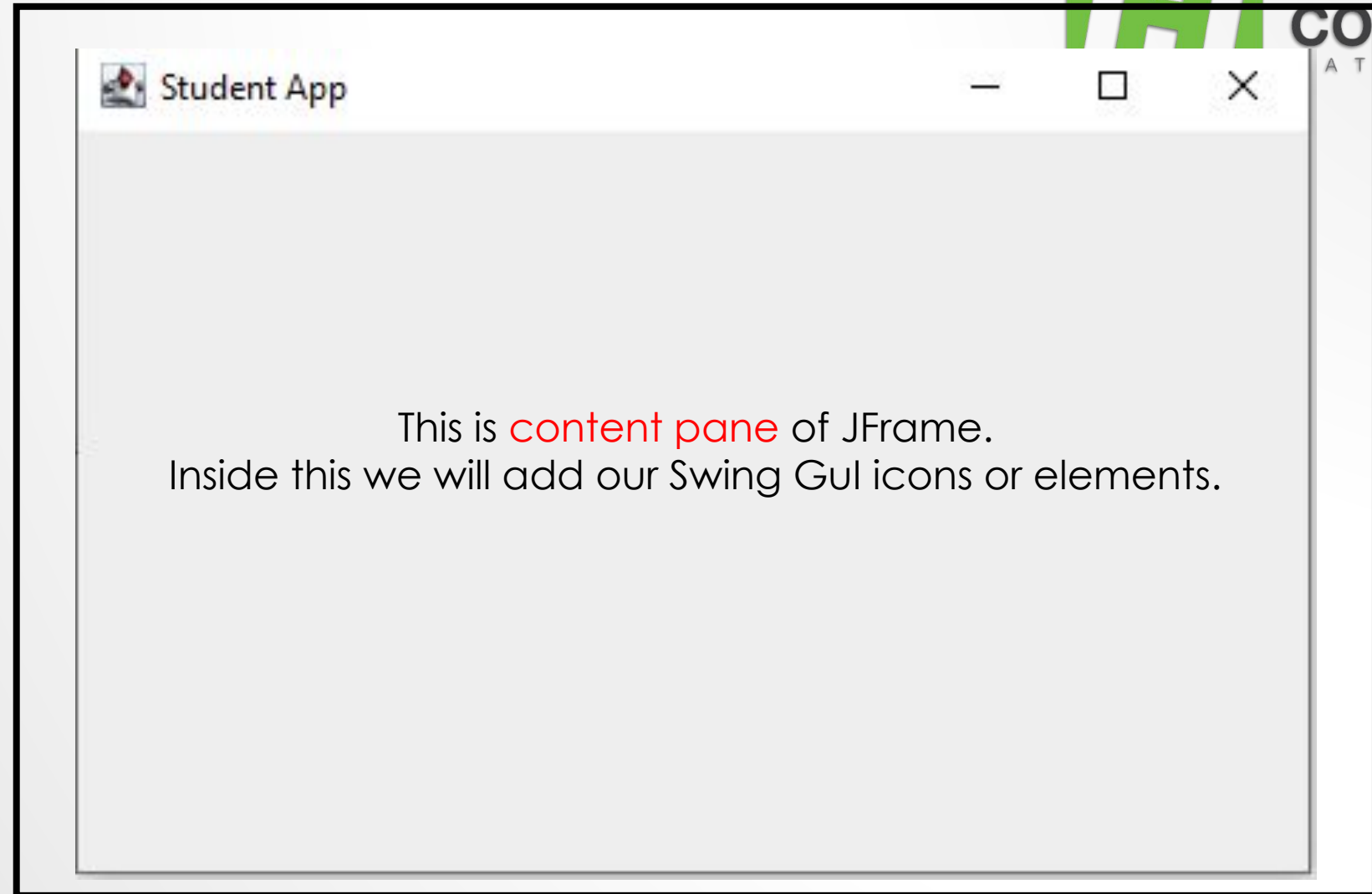
15.1 Adding Event HANdler

15.2 Mouse Listener

15.3 Mouse Listener API

15.4 Mouse Event Class

16.Example of Login Page

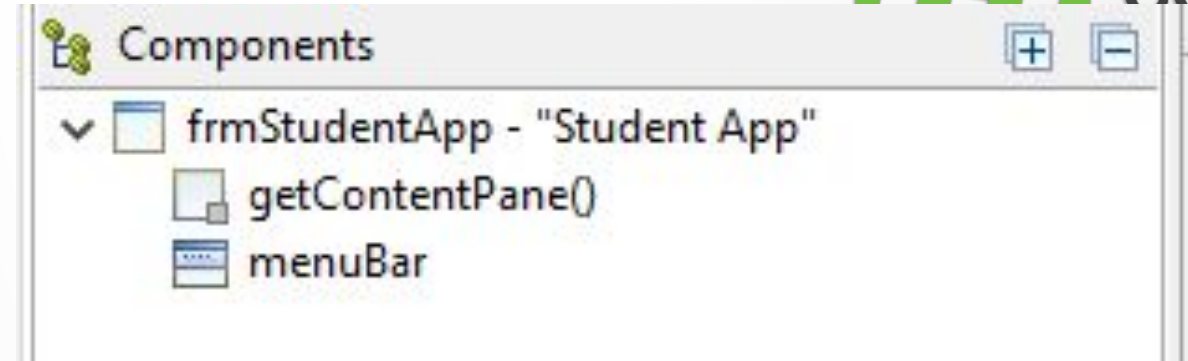


JFrame

Containment Hierarchy it's just like DOM tree

Design View

Containment Hierarchy



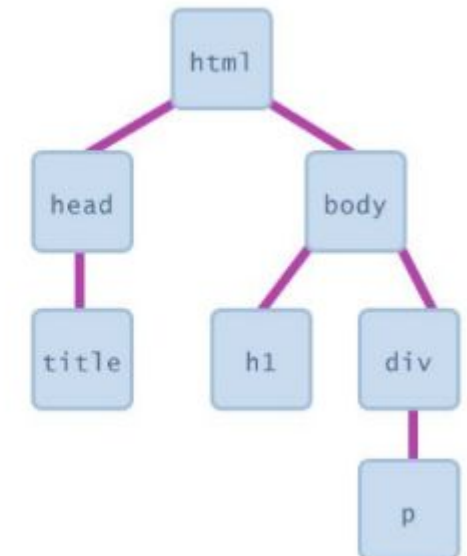
JFrame

Content Pane

Content pane we can treat
this as `<body>`
Element.

What about `<head>` i.e JMenuBar

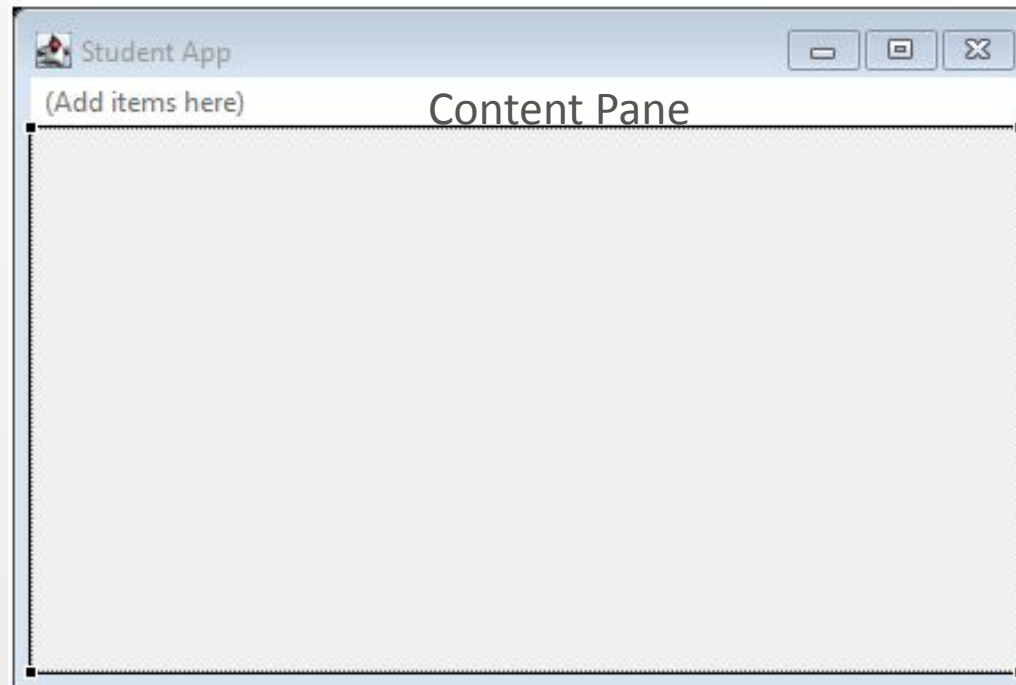
```
<html>
  <head>
    <title></title>
  </head>
  <body>
    <h1></h1>
    <div>
      <p></p>
    </div>
  </body>
</html>
```



Inside the contentPane we will another Swing GUI elements or Components.

Steps to follow.....

1. Get the contentPane or select the content pane



- 1. Java Swing
- 2. Java Swing and AWT Differences
- 3. Hierarchy of Java Swing
- 4. Method of component class
- 5. Ways of frame creation

- 5.1 JFrame inside main method
- 5.2 By association

5.3 By inheritance

6. Window Builder

7. Window Builder Installation

8. Creation of empty JFrame

9. Components of Window Builder

9.1 Source View

9.2 Design View

- 9.2.1 Component Tree
- 9.2.2 Property Pane
- 9.2.3 Palette
 - 9.2.3.1 Types of palette

10. Layout Selection

- 10.1 Absolute Layout
- 10.2 Flow Layout
- 10.3 Border Layout
- 10.4 Card Layout

11. Grids and Spring

12. Components

13. Menu

14. AWT Components

15. Event Handler

15.1 Adding Event Handler

15.2 Mouse Listener

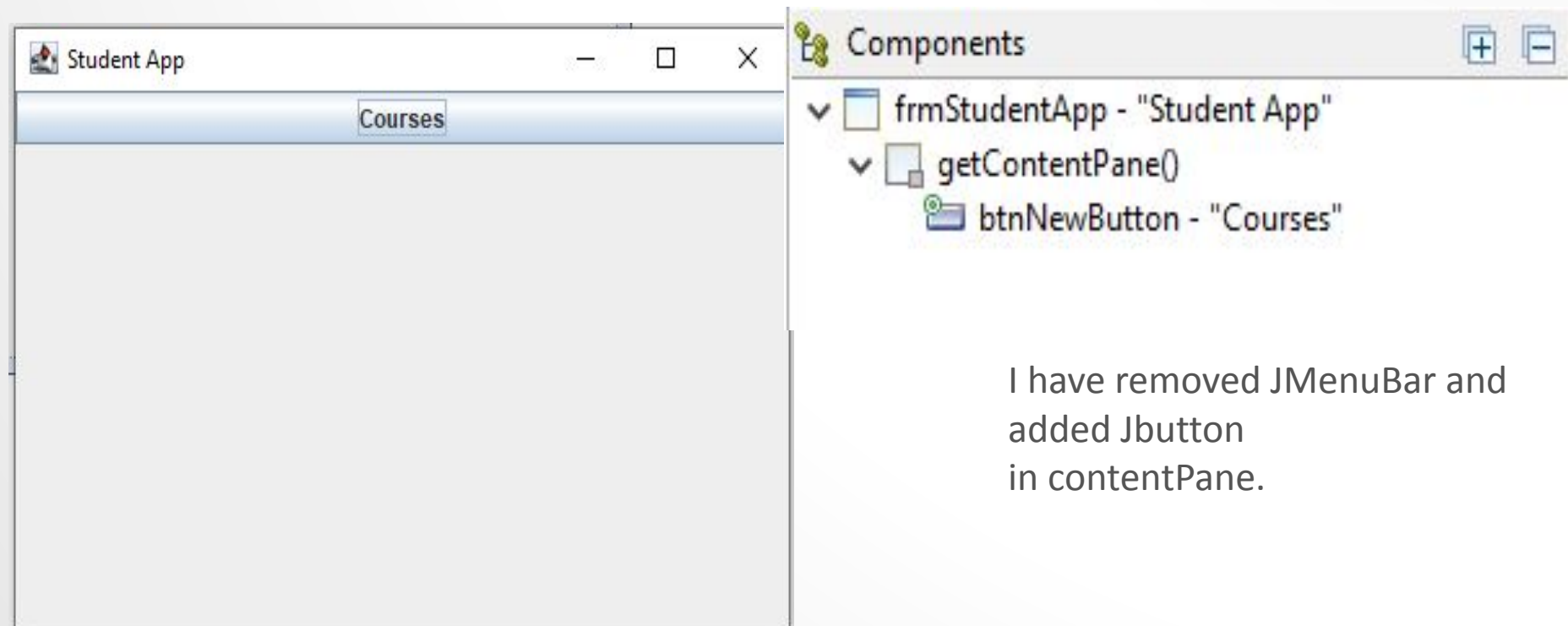
15.3 Mouse Listener API

15.4 Mouse Event Class

16. Example of Login Page

Inside the contentPane we will add another Swing GUI elements or Components.

2. Add JButton(treat this as a <button> element) inside the content pane.



I have removed JMenuBar and added JButton in contentPane.

1. Java Swing
2. Java Swing and AWT Differences
3. Hierarchy of Java Swing
4. Method of component class
5. Ways of frame creation

- 5.1 JFrame inside main method
- 5.2 By association
- 5.3 By inheritance

6. Window Builder

7. Window Builder Installation

8. Creation of empty JFrame

9. Components of Window Builder

9.1 Source View

9.2 Design View

- 9.2.1 Component Tree
- 9.2.2 Property Pane
- 9.2.3 Palette
 - 9.2.3.1 Types of palette

10. Layout Selection

- 10.1 Absolute Layout
- 10.2 Flow Layout
- 10.3 Border Layout
- 10.4 Card Layout

11. Structs and Spring

12. Components

13. Menu

14. AWT Components

15. Event Handler

15.1 Adding Event Handler

15.2 Mouse Listener

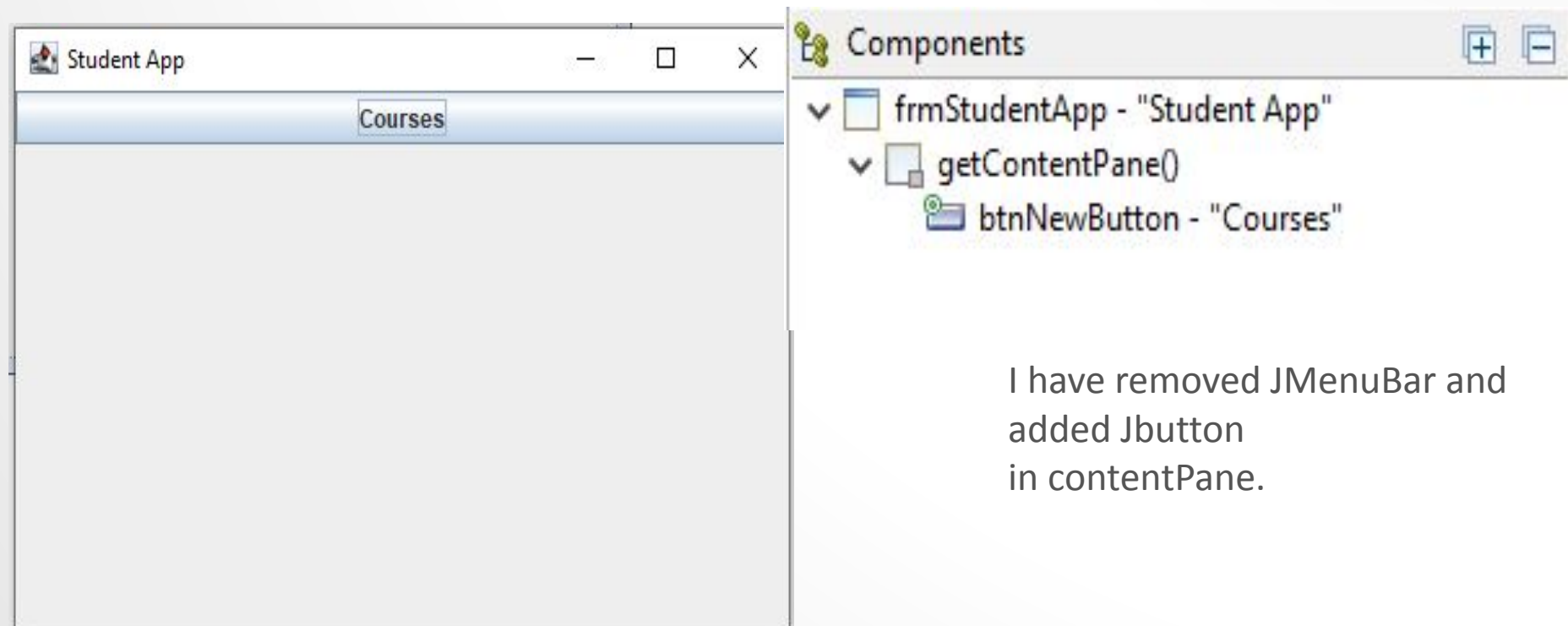
15.3 Mouse Listener API

15.4 Mouse Event Class

16. Example of Login Page

Inside the contentPane we will add another Swing GUI elements or Components.

2. Add JButton(treat this as a <button> element) inside the content pane.



I have removed JMenuBar and added JButton in contentPane.

1. Java Swing
2. Java Swing and AWT Differences
3. Hierarchy of Java Swing
4. Method of component class
5. Ways of frame creation

- 5.1 JFrame inside main method
- 5.2 By association
- 5.3 By inheritance

6. Window Builder

7. Window Builder Installation

8. Creation of empty JFrame

9. Components of Window Builder

9.1 Source View

9.2 Design View

- 9.2.1 Component Tree
- 9.2.2 Property Pane
- 9.2.3 Palette
 - 9.2.3.1 Types of palette

10. Layout Selection

- 10.1 Absolute Layout
- 10.2 Flow Layout
- 10.3 Border Layout
- 10.4 Card Layout

11. Structs and Spring

12. Components

13. Menu

14. AWT Components

15. Event Handler

15.1 Adding Event Handler

15.2 Mouse Listener

15.3 Mouse Listener API

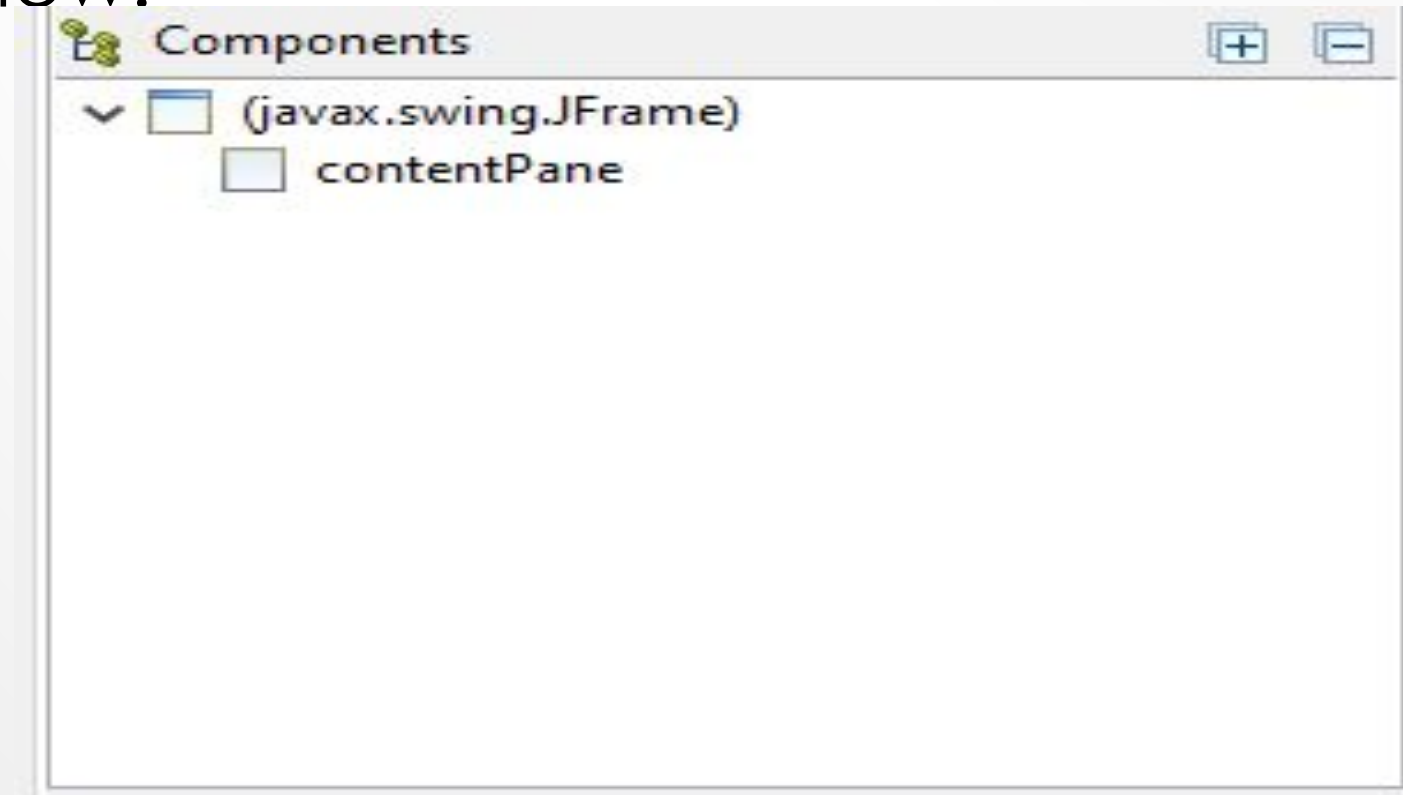
15.4 Mouse Event Class

16. Example of Login Page

Design View(Component Tree)

- **Component Tree:**

The Component Tree shows the hierarchical relationship between all of the components in the Design View.

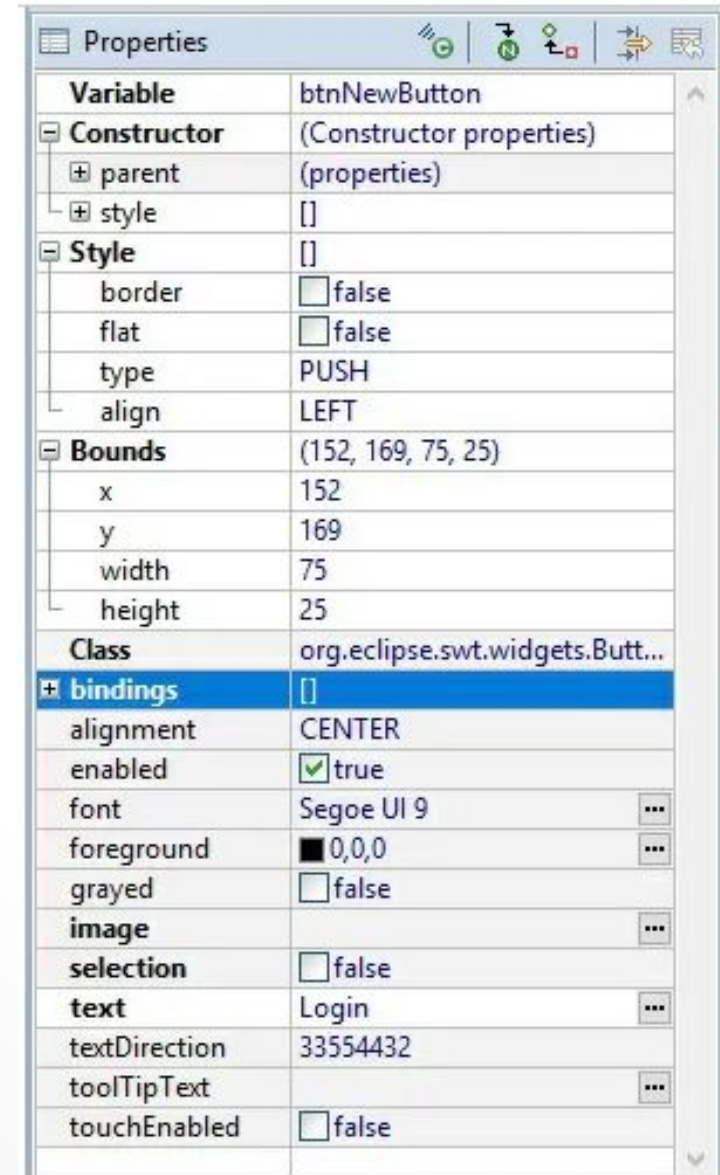


- 1.Java Swing
- 2.Java Swing and AWT Differences
- 3.Hierarchy of Java Swing
- 4.Method of component class
- 5.Ways of frame creation
 - 5.1Jframe inside main method
 - 5.2By association
 - 5.3By inheritance
- 6.Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9.Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13.Menu
- 14.AWT Components
- 15.Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class
- 16. Example of Login Page

Design View(Property Pane)

● Property Pane:

The Property Pane displays properties and events of the selected components and provides editable text fields, lists and other controls to allow you to edit properties and events.



1.Java Swing

2.Java Swing and AWT Differences

3.Hierarchy of Java Swing

4.Method of component class

5.Ways of frame creation

5.1Jframe inside main method

5.2By association

5.3By inheritance

6.Window Builder

7. Window Builder Installation

8. Creation of empty JFrame

9.Components of Window Builder

9.1 Source View

9.2 Design View

9.2.1 Component Tree

9.2.2 Property Pane

9.2.3 Palette

9.2.3.1 Types of palette

10. Layout Selection

10.1 Absolute Layout

10.2 Flow Layout

10.3 Border Layout

10.4 Card Layout

11. Struts and Spring

12. Components

13.Menu

14.AWT Components

15.Event Handler

15.1 Adding Event Handler

15.2 Mouse Listener

15.3 Mouse Listener API

15.4 Mouse Event Class

Design View(Palette)

- **Palette:** The Palette provides quick access to toolkit-specific components as well as any custom components installed by the user. The Palette is organized into categories which may be expanded, collapsed or hidden. To add a component to the Design View, we can:
 - Select it in the palette and drop it on the Design View or Component Tree by clicking in the desired location.
 - Use the Choose Component command to select the widget type from Choose Component dialog.

1.Java Swing

2.Java Swing and AWT Differences

3.Hierarchy of Java Swing

4.Method of component class

5.Ways of frame creation

5.1Jframe inside main method

5.2By association

5.3By inheritance

6.Window Builder

7. Window Builder Installation

8. Creation of empty JFrame

9.Components of Window Builder

9.1 Source View

9.2 Design View

9.2.1 Component Tree

9.2.2 Property Pane

9.2.3 Palette

9.2.3.1 Types of palette

10. Layout Selection

10.1 Absolute Layout

10.2 Flow Layout

10.3 Border Layout

10.4 Card Layout

11. Struts and Spring

12. Components

13.Menu

14.AWT Components

15.Event Handler

15.1 Adding Event Handler

15.2 Mouse Listener

15.3 Mouse Listener API

15.4 Mouse Event Class

Types of Palette

JPanel – A generic lightweight container.

JScrollPane – Provides a scrollable view of a lightweight component. A JScrollPane manages a viewport, optional vertical and horizontal scroll bars, and optional row and column heading viewports.

JSplitPane – JSplitPane is used to divide two (and only two) Components. The two Components are graphically divided based on the look and feel implementation, and the two Components can then be interactively resized by the user.

- 1.Java Swing
- 2.Java Swing and AWT Differences
- 3.Hierarchy of Java Swing
- 4.Method of component class
- 5.Ways of frame creation
 - 5.1Jframe inside main method
 - 5.2By association
 - 5.3By inheritance
- 6.Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9.Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13.Menu
- 14.AWT Components
- 15.Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class

Types of Palette

JTabbedPane – A component that lets the user switch between a group of components by clicking on a tab with a given title and/or icon.

JToolBar – A component that is useful for displaying commonly used Action's or controls.

JDesktopPane – A container used to create a multiple-document interface or a virtual desktop.

InternalFrame – A lightweight object that provides many of the features of a native frame, including dragging, closing, becoming an icon, resizing, title display, and support for a menu bar.

- 1. Java Swing
- 2. Java Swing and AWT Differences
- 3. Hierarchy of Java Swing
- 4. Method of component class
- 5. Ways of frame creation
 - 5.1 JFrame inside main method
 - 5.2 By association
 - 5.3 By inheritance
- 6. Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9. Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13. Menu
- 14. AWT Components
- 15. Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API

2.Layout Selection

- **What is layout?**

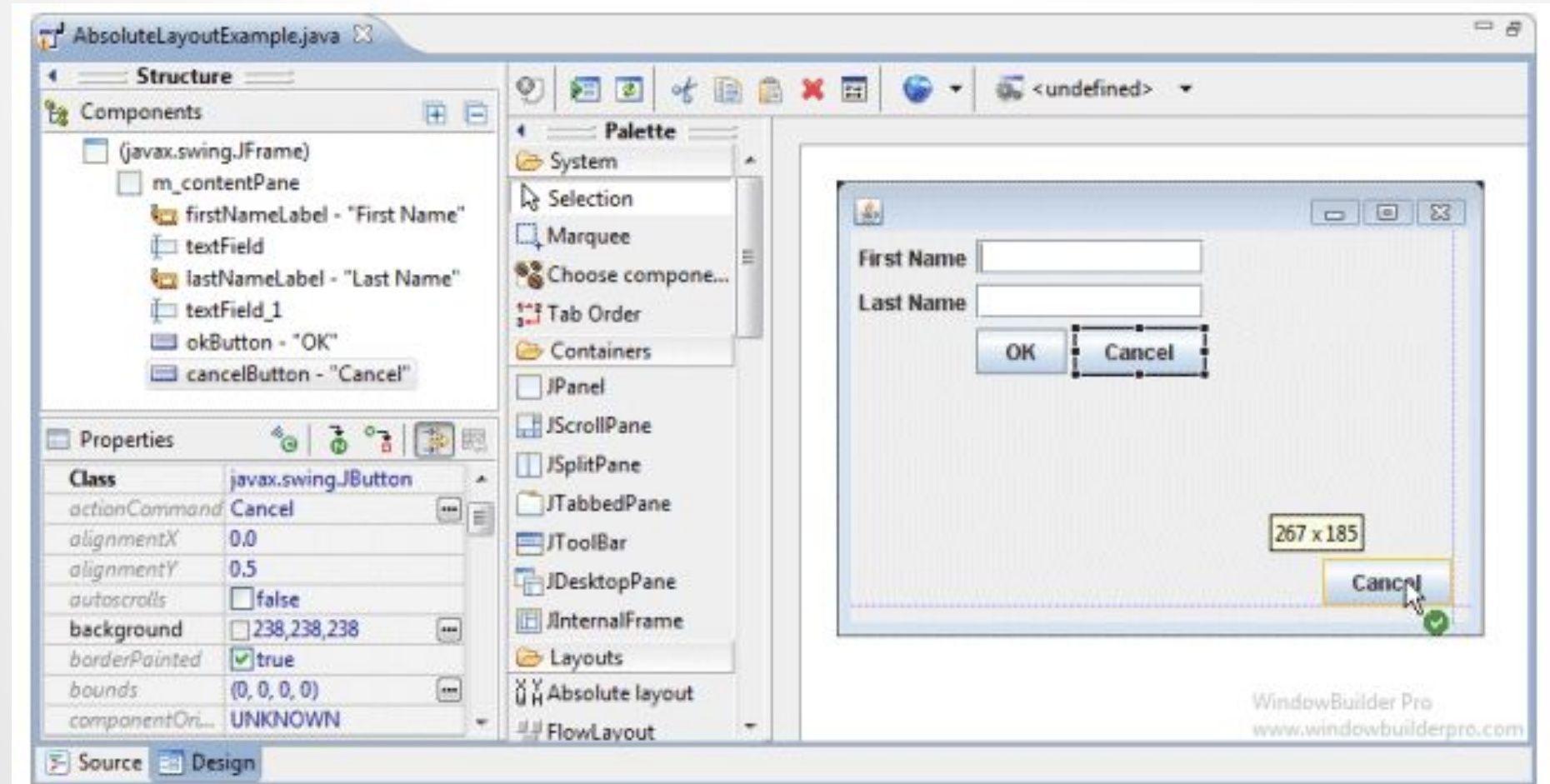
->Layout refers to the arrangement of components within the container.

Types

2.1 Absolute Layout or Null Layout: is a simple x,y oriented layout. During layout, the live component is shown moving with a tooltip showing its current location or size.

- 1.Java Swing
- 2.Java Swing and AWT Differences
- 3.Hierarchy of Java Swing
- 4.Method of component class
- 5.Ways of frame creation
 - 5.1Jframe inside main method
 - 5.2By association
 - 5.3By inheritance
- 6.Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9.Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection**
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13.Menu
- 14.AWT Components
- 15.Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener**
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class

2.1.Absolute Layout

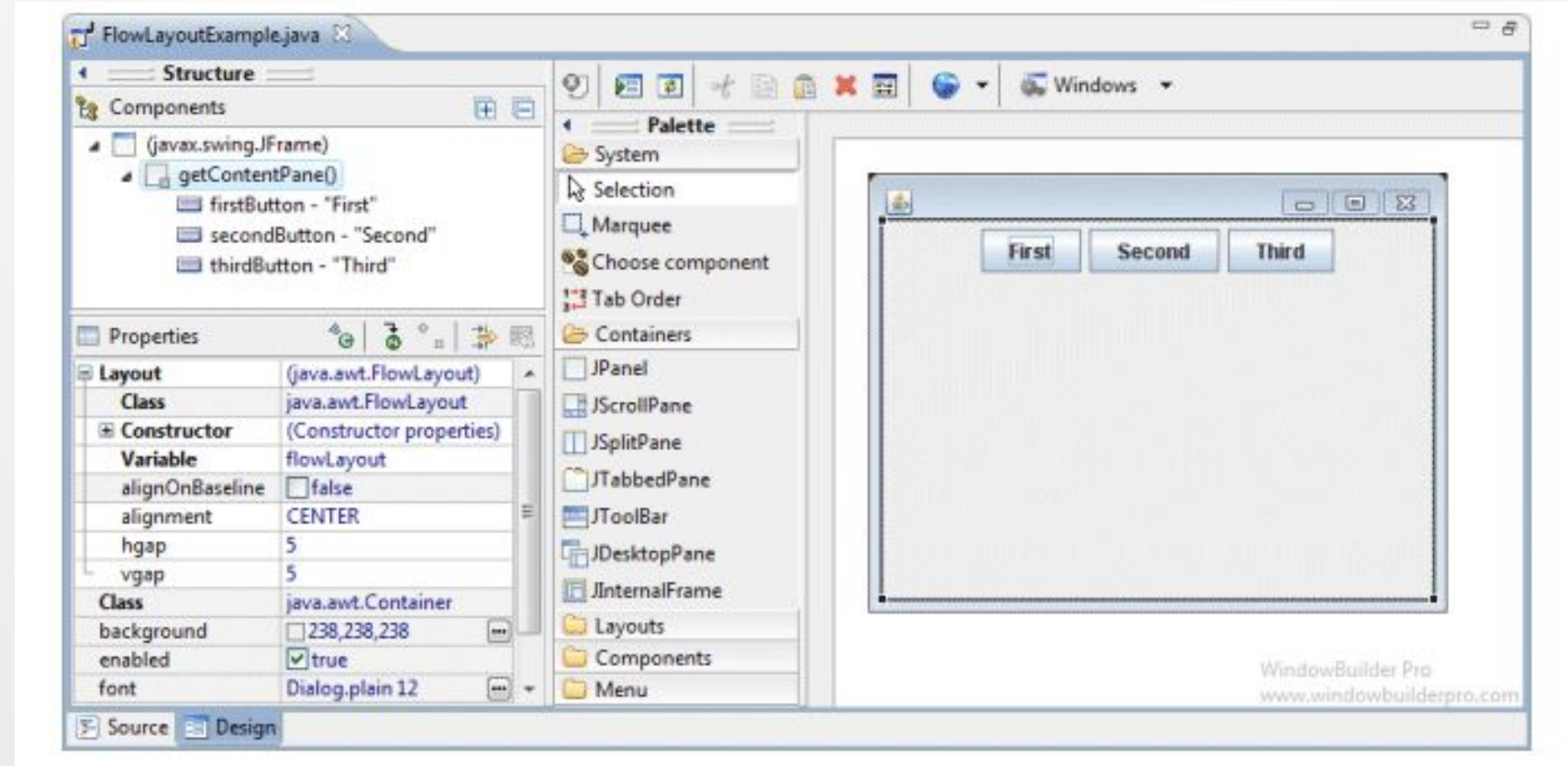


- 1.Java Swing
- 2.Java Swing and AWT Differences
- 3.Hierarchy of Java Swing
- 4.Method of component class
- 5.Ways of frame creation
 - 5.1Jframe inside main method
 - 5.2By association
 - 5.3By inheritance
- 6.Window Builder
- 7. Window Builder Installation
- 8. Creation of empty jframe
- 9.Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13.Menu
- 14.AWT Components
- 15.Event Handler
 - 15.1 Adding Event HANDler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class

2.2.Flow Layout

- A **FlowLayout** arranges components in a directional flow, much like lines of text in a paragraph. **FlowLayout** are typically used to arrange buttons in a panel. It arranges buttons horizontally until no more buttons fit on the same line.

2.2 Flow Layout

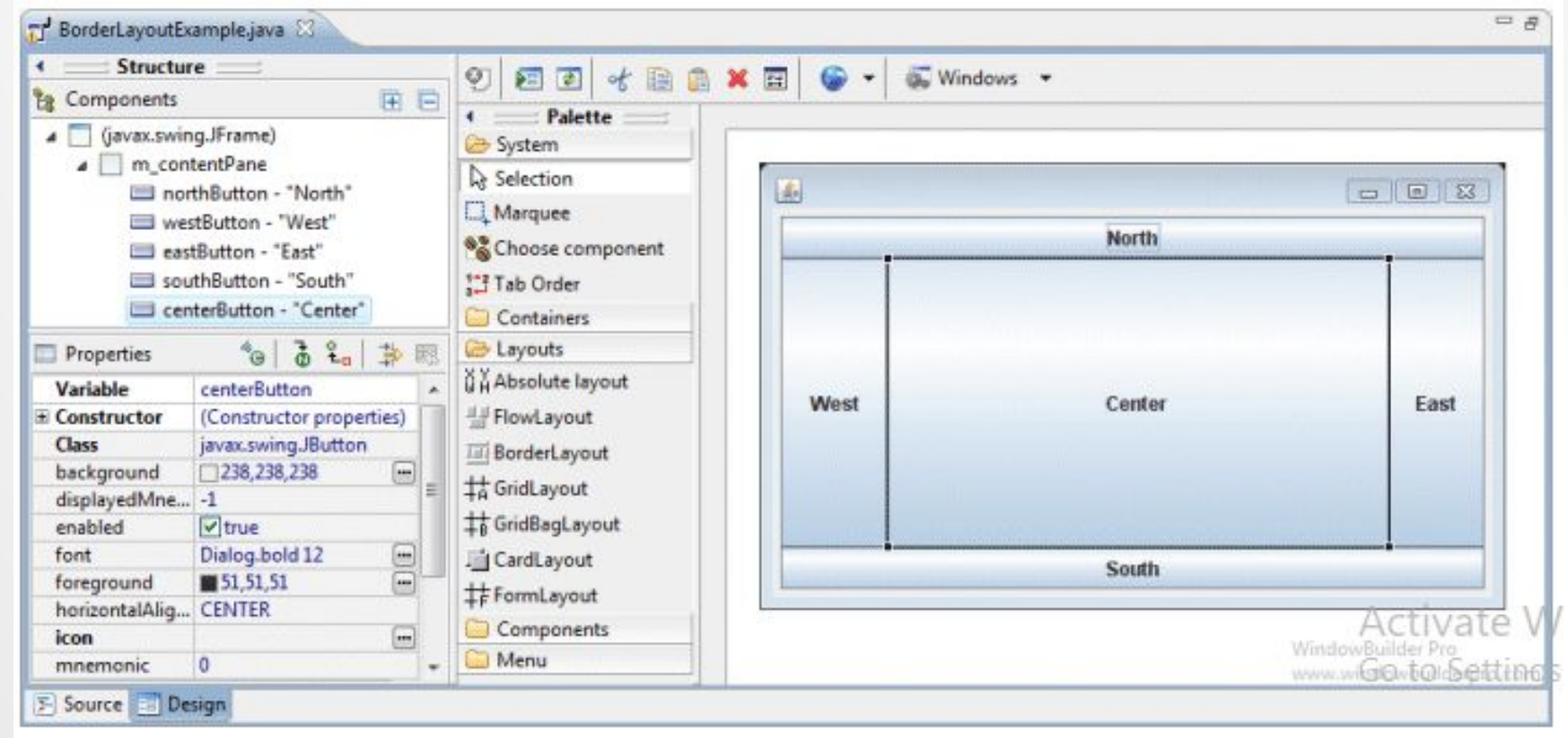


- 1. Java Swing
- 2. Java Swing and AWT Differences
- 3. Hierarchy of Java Swing
- 4. Method of component class
- 5. Ways of frame creation
 - 5.1 JFrame inside main method
 - 5.2 By association
 - 5.3 By inheritance
- 6. Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9. Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13. Menu
- 14. AWT Components
- 15. Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class

2.3.Border Layout

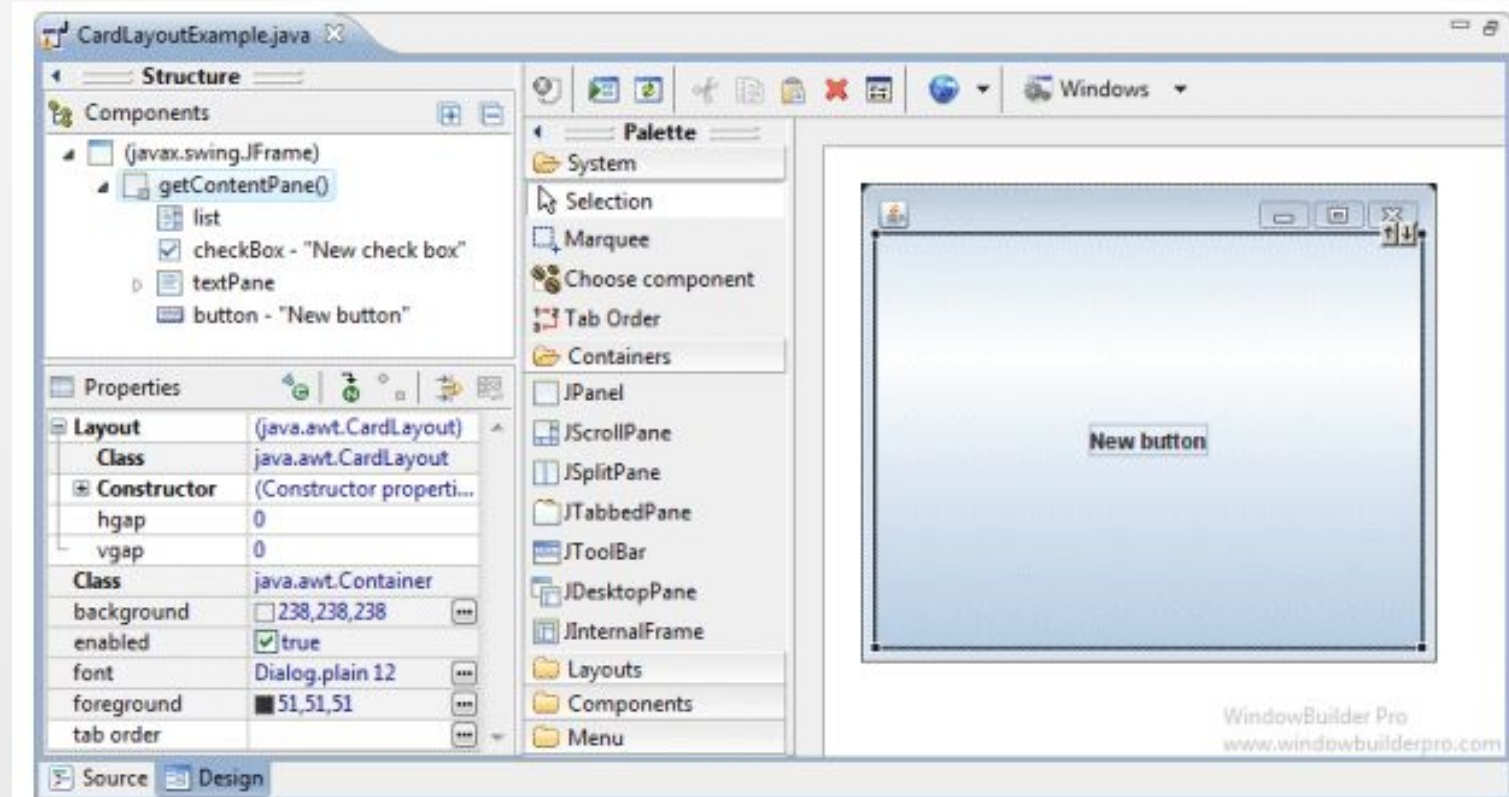
- **BorderLayout** lays out a container, arranging and resizing its components to fit in five regions: **north**, **south**, **east**, **west**, and **center**. Each region may contain no more than one component, and is identified by a corresponding constant: **NORTH**, **SOUTH**, **EAST**, **WEST**, and **CENTER**.

2.3.Border Layout



2.4.Card Layout

- A **CardLayout** object is a layout manager for a container. It treats each component in the container as a card.



Structs and Spring

Horizontal Box – Creates a panel that uses an implicit horizontal BoxLayout.

Vertical Box – Creates a panel that uses an implicit vertical BoxLayout.

Horizontal strut – Creates an invisible, fixed-width component .

Vertical strut – Creates an invisible, fixed-height component .

- 1.Java Swing
- 2.Java Swing and AWT Differences
- 3.Hierarchy of Java Swing
- 4.Method of component class
- 5.Ways of frame creation
 - 5.1Jframe inside main method
 - 5.2By association
 - 5.3By inheritance
- 6.Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9.Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Structs and Spring**
- 12. Components
- 13.Menu
- 14.AWT Components
- 15.Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class

Structs and Spring

Horizontal glue – Creates a horizontal glue component .

Vertical glue – Creates a vertical glue component .
Rigid area – Creates an invisible component that's always the specified size .

Glue – Creates an invisible glue component.

- 1.Java Swing
- 2.Java Swing and AWT Differences
- 3.Hierarchy of Java Swing
- 4.Method of component class
- 5.Ways of frame creation
 - 5.1Jframe inside main method
 - 5.2By association
 - 5.3By inheritance
- 6.Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9.Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Structs and Spring**
- 12. Components
- 13.Menu
- 14.AWT Components
- 15.Event Handler
 - 15.1 Adding Event HANdler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class

Components

JLabel – A display area for a short text string or an image, or both.

TextField – A lightweight component that allows the editing of a single line of text.

JComboBox – A component that combines a button or editable field and a drop-down list. The user can select a value from the drop-down list, which appears at the user's request.

Button – An implementation of a “push” button.

JCheckBox – An implementation of a check box — an item that can be selected or deselected, and which displays its state to the user.

- 1. Java Swing
- 2. Java Swing and AWT Differences
- 3. Hierarchy of Java Swing
- 4. Method of component class
- 5. Ways of frame creation
 - 5.1 JFrame inside main method
 - 5.2 By association
 - 5.3 By inheritance
- 6. Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9. Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Grids and Spring
- 12. Components
- 13. Menu
- 14. AWT Components
- 15. Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class

Components

JRadioButton – An implementation of a radio button — an item that can be selected or deselected, and which displays its state to the user.

JToggleButton – An implementation of a two-state button — an item that can be selected or deselected, and which displays its state to the user.

JTextArea – A JTextArea is a multi-line area that displays plain text.

- 1.Java Swing
- 2.Java Swing and AWT Differences
- 3.Hierarchy of Java Swing
- 4.Method of component class
- 5.Ways of frame creation
 - 5.1Jframe inside main method
 - 5.2By association
 - 5.3By inheritance
- 6.Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9.Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Structs and Spring
- 12. Components
- 13.Menu
- 14.AWT Components
- 15.Event Handler
 - 15.1 Adding Event HAndler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class

Components

JList – A component that allows the user to select one or more objects from a list.

JTable – The JTable is used to display and edit regular two-dimensional tables of cells.

JSeparator – A horizontal or vertical line used to separate other controls.

JSlider – A component that lets the user graphically select a value by sliding a knob within a bounded interval.

JProgressBar – A component that, by default, displays an integer value within a bounded interval.

Menu

MenuBar – A menu bar.

JPopupMenu – A popup menu.

JMenu – Either a pull-down or cascaded menu.

JMenuItem – A simple menu item.

JCheckBoxMenuItem – A checkbox menu item.

JRadioButtonMenuItem – A radio button menu item.

- 1. Java Swing
- 2. Java Swing and AWT Differences
- 3. Hierarchy of Java Swing
- 4. Method of component class
- 5. Ways of frame creation
 - 5.1 JFrame inside main method
 - 5.2 By association
 - 5.3 By inheritance
- 6. Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9. Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13. Menu**
- 14. AWT Components
- 15. Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class

AWT Components

Panel – Panel is the simplest container class. A panel provides space in which an application can attach any other component, including other panels.

Canvas – A Canvas component represents a blank rectangular area of the screen onto which the application can draw or from which the application can trap input events from the user.

ScrollPane – A container class which implements automatic horizontal and/or vertical scrolling for a single child component.

Button – This class creates a labeled button. The application can cause some action to happen when the button is pushed.

- 1. Java Swing
- 2. Java Swing and AWT Differences
- 3. Hierarchy of Java Swing
- 4. Method of component class
- 5. Ways of frame creation
 - 5.1 JFrame inside main method
 - 5.2 By association
 - 5.3 By inheritance
- 6. Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9. Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13. Menu
- 14. AWT Components
- 15. Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class

AWT Components

Checkbox – A check box is a graphical component that can be in either an “on” (true) or “off” (false) state. Clicking on a check box changes its state from “on” to “off”, or from “off” to “on”.

Choice – The Choice class presents a pop-up menu of choices. The current choice is displayed as the title of the menu.

List – The List component presents the user with a scrolling list of text items. The list can be set up so that the user can choose either one item or multiple items.

- 1. Java Swing
- 2. Java Swing and AWT Differences
- 3. Hierarchy of Java Swing
- 4. Method of component class
- 5. Ways of frame creation
 - 5.1 JFrame inside main method
 - 5.2 By association
 - 5.3 By inheritance
- 6. Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9. Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13. Menu
- 14. AWT Components**
- 15. Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class

AWT Components

Scrollbar – The Scrollbar class embodies a scrollbar, a familiar user-interface object. A scrollbar provides a convenient means for allowing a user to select from a range of values.

TextField – A TextField object is a text component that allows for the editing of a single line of text.

TextArea – A TextArea object is a multi-line region that displays text. It can be set to allow editing or to be read-only.

- 1. Java Swing
- 2. Java Swing and AWT Differences
- 3. Hierarchy of Java Swing
- 4. Method of component class
- 5. Ways of frame creation
 - 5.1 JFrame inside main method
 - 5.2 By association
 - 5.3 By inheritance
- 6. Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9. Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13. Menu
- 14. AWT Components
- 15. Event Handler
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class

Event Handler

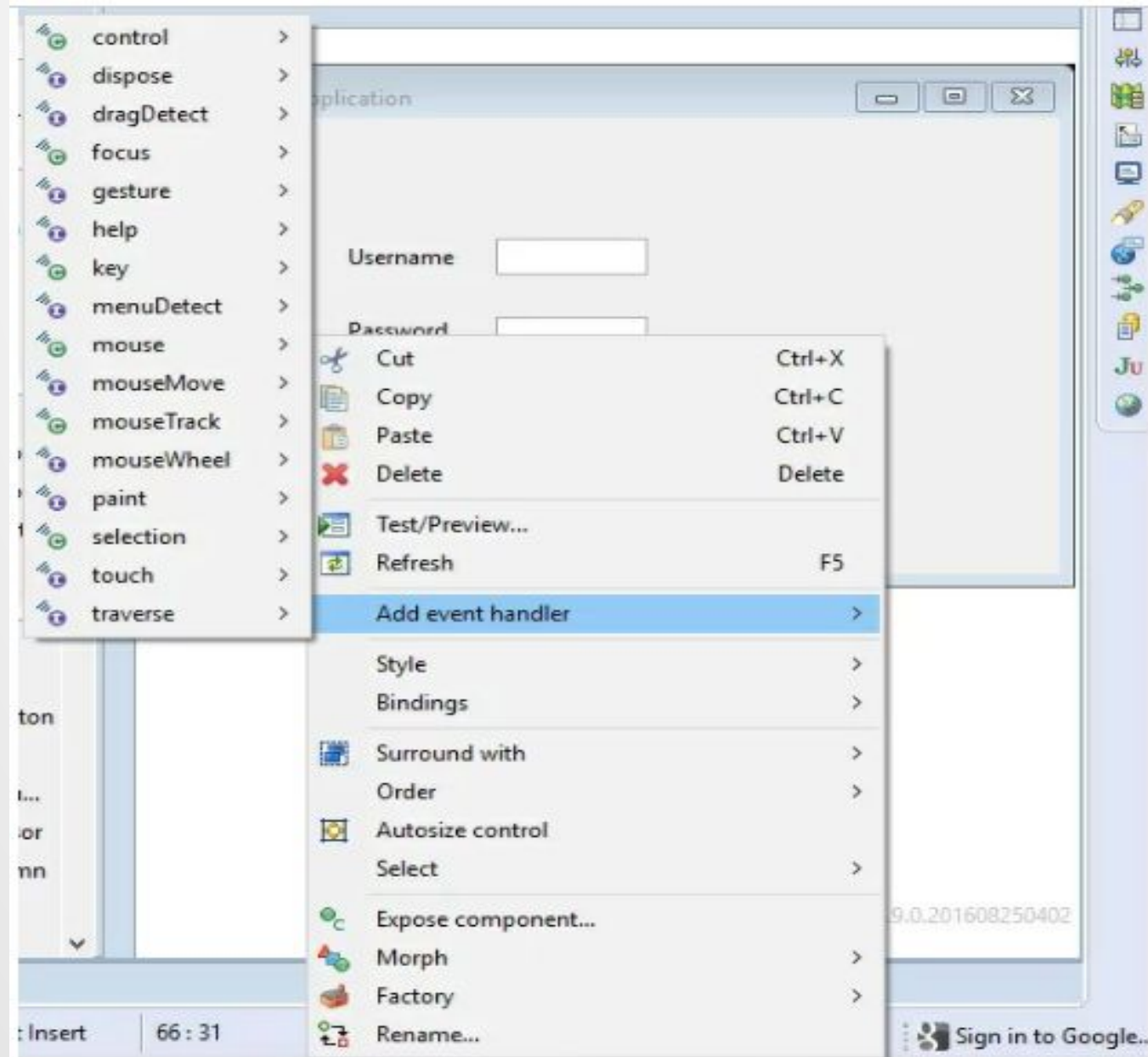
- Events occur when the user interacts with the UI.
- The appropriate event-handling code is then executed.
- In order to know when events occur, event handlers must first be added to your components.
- The tool makes it very easy to add and remove event listeners to your components.

Event Handler

- Way to add an event handler is to simply right-click on a component (either in the Design View or in the Component Tree), select Add event handler > [name of the event] > [name of the event handler to implement].

- 1.Java Swing
- 2.Java Swing and AWT Differences
- 3.Hierarchy of Java Swing
- 4.Method of component class
- 5.Ways of frame creation
 - 5.1Jframe inside main method
 - 5.2By association
 - 5.3By inheritance
- 6.Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9.Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13.Menu
- 14.AWT Components
- 15.Event Handler**
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener**
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class
- 16.Example of Login Page

Adding Event Handler



1. Java Swing
2. Java Swing and AWT Differences
3. Hierarchy of Java Swing
4. Method of component class
5. Ways of frame creation
 - 5.1 JFrame inside main method
 - 5.2 By association
 - 5.3 By inheritance
6. Window Builder
7. Window Builder Installation
8. Creation of empty JFrame
9. Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
11. Structs and Spring
12. Components
13. Menu
14. AWT Components
- 15. Event Handler**
 - 15.1 Adding Event Handler**
 - 15.2 Mouse Listener**
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class
16. Example of Login Page

Mouse Listener

- Mouse events notify when the user uses the mouse (or similar input device) to interact with a component. Mouse events occur when the cursor enters or exits a component's onscreen area and when the user presses or releases one of the mouse buttons.

- 1.Java Swing
- 2.Java Swing and AWT Differences
- 3.Hierarchy of Java Swing
- 4.Method of component class
- 5.Ways of frame creation
 - 5.1Jframe inside main method
 - 5.2By association
 - 5.3By inheritance
- 6.Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9.Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13.Menu
- 14.AWT Components
- 15.Event Handler**
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener**
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class
- 16.Example of Login Page

Mouse Listener API

- The **MouseAdapter** class (the AWT adapter class) is abstract. All its methods have an empty body. So a developer can define methods for events specific to the application. You can also use the **MouseInputAdapter** class, which has all the methods available from `MouseListener` and `MouseMotionListener`.

- 1.Java Swing
- 2.Java Swing and AWT Differences
- 3.Hierarchy of Java Swing
- 4.Method of component class
- 5.Ways of frame creation
 - 5.1Jframe inside main method
 - 5.2By association
 - 5.3By inheritance
- 6.Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9.Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Grids and Spring
- 12. Components
- 13.Menu
- 14.AWT Components
- 15.Event Handler**
 - 15.1 Adding Event Handler
 - 15.2 Mouse Listener**
 - 15.3 Mouse Listener API**
 - 15.4 Mouse Event Class
- 16.Example of Login Page

Mouse Listener API

Method	Purpose
<code>mouseClicked(MouseEvent)</code>	Called just after the user clicks the listened-to component.
<code>mouseEntered(MouseEvent)</code>	Called just after the cursor enters the bounds of the listened-to component.
<code>mouseExited(MouseEvent)</code>	Called just after the cursor exits the bounds of the listened-to component.
<code>mousePressed(MouseEvent)</code>	Called just after the user presses a mouse button while the cursor is over the listened-to component.
<code>mouseReleased(MouseEvent)</code>	Called just after the user releases a mouse button after a mouse press over the listened-to component.

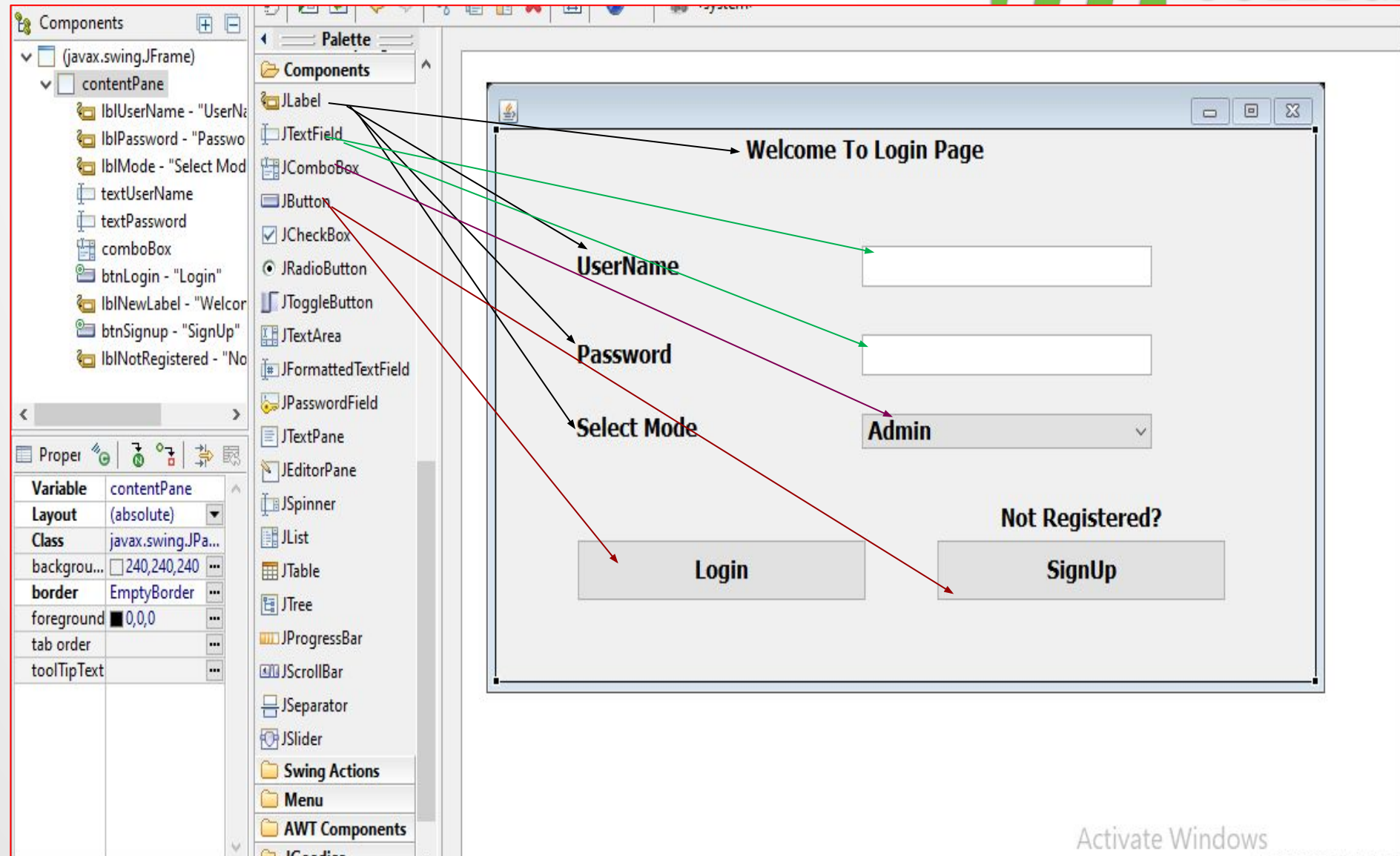
- 1.Java Swing
- 2.Java Swing and AWT Differences
- 3.Hierarchy of Java Swing
- 4.Method of component class
- 5.Ways of frame creation
 - 5.1Jframe inside main method
 - 5.2By association
 - 5.3By inheritance
- 6.Window Builder
- 7. Window Builder Installation
- 8. Creation of empty jframe
- 9.Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Struts and Spring
- 12. Components
- 13.Menu
- 14.AWT Components
- 15.Event Handler**
 - 15.1 Adding Event HANdler
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API**
 - 15.4 Mouse Event Class
- 16.Example of Login Page

Mouse Event Class

Method	Purpose
<code>int getClickCount()</code>	Returns the number of quick, consecutive clicks the user has made (including this event). For example, returns 2 for a double click.
<code>int getX()</code> <code>int getY()</code> <code>Point getPoint()</code>	Return the (x,y) position at which the event occurred, relative to the component that fired the event.
<code>int getXOnScreen()</code> <code>int getYOnScreen()</code> <code>int getLocationOnScreen()</code>	Return the absolute (x,y) position of the event. These coordinates are relative to the virtual coordinate system for the multi-screen environment. Otherwise, these coordinates are relative to the coordinate system associated with the Component's Graphics Configuration.
<code>int getButton()</code>	Returns which mouse button, if any, has a changed state. One of the following constants is returned: NOBUTTON, BUTTON1, BUTTON2, or BUTTON3.
<code>boolean isPopupTrigger()</code>	Returns true if the mouse event should cause a popup menu to appear. Because popup triggers are platform dependent, if your program uses popup menus, you should call <code>isPopupTrigger</code> for all mouse-pressed and mouse-released events fired by components over which the popup can appear. See Bringing Up a Popup Menu for more information about popup menus.
<code>String getMouseModifiersText(int)</code>	Returns a String describing the modifier keys and mouse buttons that were active during the event, such as "Shift", or "Ctrl+Shift". These strings can be localized using the <code>awt.properties</code> file.

- 1.Java Swing
- 2.Java Swing and AWT Differences
- 3.Hierarchy of Java Swing
- 4.Method of component class
- 5.Ways of frame creation
 - 5.1Jframe inside main method
 - 5.2By association
 - 5.3By inheritance
- 6.Window Builder
- 7. Window Builder Installation
- 8. Creation of empty jframe
- 9.Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Structs and Spring
- 12. Components
- 13.Menu
- 14.AWT Components
- 15.Event Handler**
 - 15.1 Adding Event HANDler
 - 15.2 Mouse Listener**
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class**
- 16.Example of Login Page

Example of Login Page



- 1. Java Swing
- 2. Java Swing and AWT Differences
- 3. Hierarchy of Java Swing
- 4. Method of component class
- 5. Ways of frame creation
 - 5.1 JFrame inside main method
 - 5.2 By association
 - 5.3 By inheritance
- 6. Window Builder
- 7. Window Builder Installation
- 8. Creation of empty JFrame
- 9. Components of Window Builder
 - 9.1 Source View
 - 9.2 Design View
 - 9.2.1 Component Tree
 - 9.2.2 Property Pane
 - 9.2.3 Palette
 - 9.2.3.1 Types of palette
- 10. Layout Selection
 - 10.1 Absolute Layout
 - 10.2 Flow Layout
 - 10.3 Border Layout
 - 10.4 Card Layout
- 11. Structs and Spring
- 12. Components
- 13. Menu
- 14. AWT Components
- 15. Event Handler
 - 15.1 Adding Event HANDLER
 - 15.2 Mouse Listener
 - 15.3 Mouse Listener API
 - 15.4 Mouse Event Class
- 16. Example of Login Page

JDBC in Java

Java JDBC

JDBC API is a Java API that can access any kind of tabular data, especially data stored in a Relational Database. JDBC works with Java on a variety of platforms, such as Windows, Mac OS, and the various versions of UNIX.

1. Java JDBC

2. Why to Learn JDBC

3. Applications of JDBC

4. JDBC Architecture

5. Common JDBC

Components

6. Creating JDBC

Applications

7. Example for Creating
Database

8. Example for Selecting
Database

9. Example for Creating Table

10. Example for Inserting
Record

11. Example for Selecting
Record

12. Example for Updating
Record

13. Example for Deleting
Record

Why to Learn JDBC

- JDBC stands for Java Database Connectivity, which is a standard Java API for database independent connectivity between the Java programming language and a wide range of databases.
- The JDBC library includes APIs for each of the tasks mentioned below that are commonly associated with database usage.
 - Making a connection to a database.
 - Creating SQL or MySQL statements.
 - Executing SQL or MySQL queries in the database.
 - Viewing & Modifying the resulting records.

1. Java JDBC
- 2. Why to Learn JDBC**
3. Applications of JDBC
4. JDBC Architecture
5. Common JDBC Components
6. Creating JDBC Applications
7. Example for Creating Database
8. Example for Selecting Database
9. Example for Creating Table
10. Example for Inserting Record
11. Example for Selecting Record
12. Example for Updating Record
13. Example for Deleting Record

Applications of JDBC

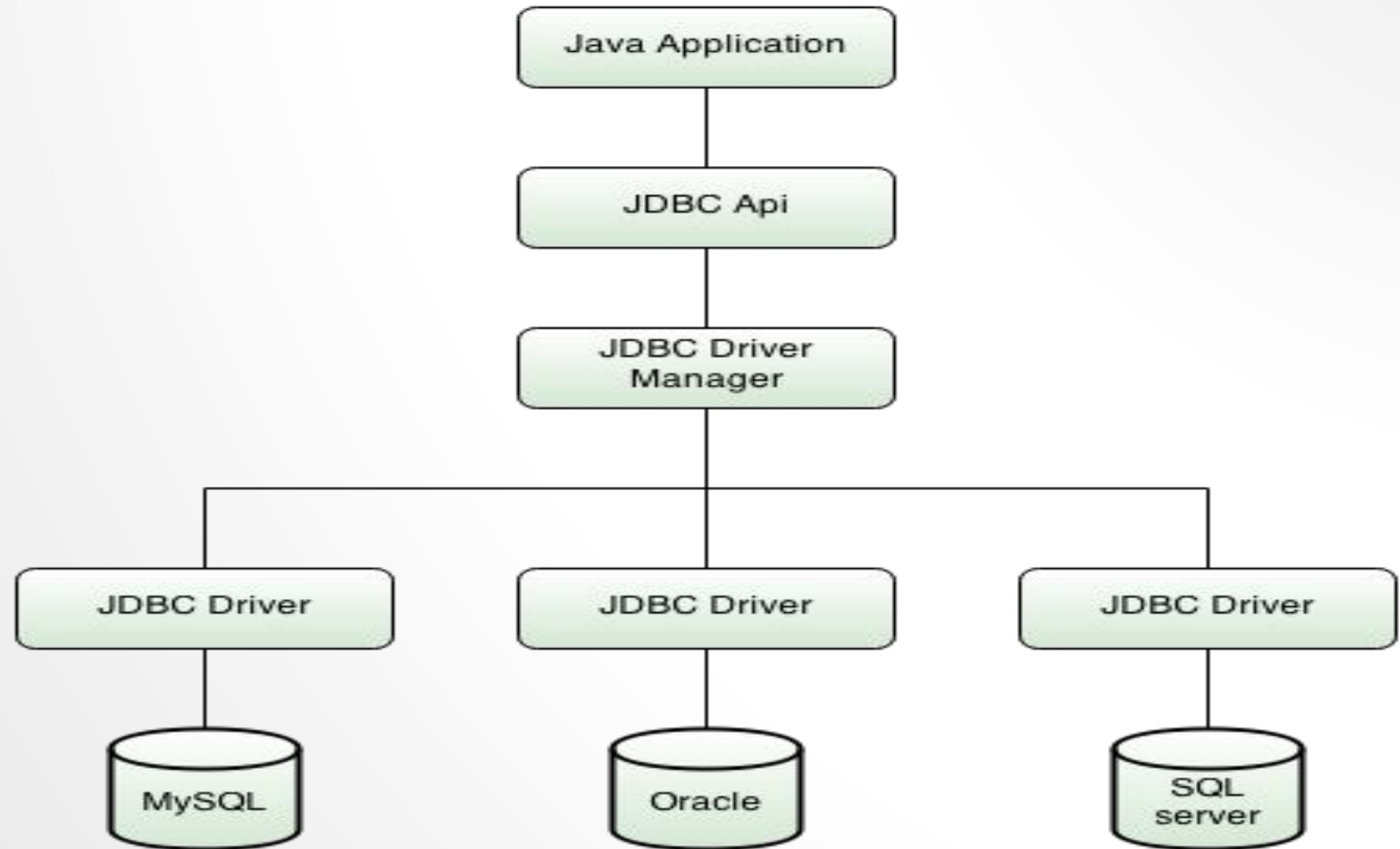
- Fundamentally, JDBC is a specification that provides a complete set of interfaces that allows for portable access to an underlying database. Java can be used to write different types of executables, such as
 - Java Applications
 - Java Applets
 - Java Servlets
 - Java ServerPages (JSPs)
 - Enterprise JavaBeans (EJBs).
- All of these different executables are able to use a JDBC driver to access a database, and take advantage of the stored data.
- JDBC provides the same capabilities as ODBC, allowing Java programs to contain database independent code.

1. Java JDBC
2. Why to Learn JDBC
- 3. Applications of JDBC**
4. JDBC Architecture
5. Common JDBC Components
6. Creating JDBC Applications
7. Example for Creating Database
8. Example for Selecting Database
9. Example for Creating Table
10. Example for Inserting Record
11. Example for Selecting Record
12. Example for Updating Record
13. Example for Deleting Record

JDBC Architecture

- The JDBC API supports both two-tier and three-tier processing models for database access but in general, JDBC Architecture consists of two layers
 - JDBC API – This provides the application-to-JDBC Manager connection.
 - JDBC Driver API – This supports the JDBC Manager-to-Driver Connection.
- The JDBC API uses a driver manager and database-specific drivers to provide transparent connectivity to heterogeneous databases.
- The JDBC driver manager ensures that the correct driver is used to access each data source. The driver manager is capable of supporting multiple concurrent drivers connected to multiple heterogeneous databases.

JDBC Architecture



1. Java JDBC
2. Why to Learn JDBC
3. Applications of JDBC
- 4. JDBC Architecture**
5. Common JDBC Components
6. Creating JDBC Applications
7. Example for Creating Database
8. Example for Selecting Database
9. Example for Creating Table
10. Example for Inserting Record
11. Example for Selecting Record
12. Example for Updating Record
13. Example for Deleting Record

Common JDBC Components

- **Driver Manager** – This class manages a list of database drivers. Matches connection requests from the java application with the proper database driver using communication sub protocol. The first driver that recognizes a certain subprotocol under JDBC will be used to establish a database Connection.
- **Driver** – This interface handles the communications with the database server. You will interact directly with Driver objects very rarely. Instead, you use Driver Manager objects, which manages objects of this type. It also abstracts the details associated with working with Driver objects.

1. Java JDBC
2. Why to Learn JDBC
3. Applications of JDBC
4. JDBC Architecture
- 5. Common JDBC Components**
6. Creating JDBC Applications
7. Example for Creating Database
8. Example for Selecting Database
9. Example for Creating Table
10. Example for Inserting Record
11. Example for Selecting Record
12. Example for Updating Record
13. Example for Deleting Record

Common JDBC Components

- **Connection** - This interface with all methods for contacting a database. The connection object represents communication context, i.e., all communication with database is through connection object only.
- **Statement** - You use objects created from this interface to submit the SQL statements to the database. Some derived interfaces accept parameters in addition to executing stored procedures.
- **Result Set** - These objects hold data retrieved from a database after you execute an SQL query using Statement objects. It acts as an iterator to allow you to move through its data.
- **SQL Exception** - This class handles any errors that occur in a database application.

1. Java JDBC
2. Why to Learn JDBC
3. Applications of JDBC
4. JDBC Architecture
- 5. Common JDBC Components**
6. Creating JDBC Applications
7. Example for Creating Database
8. Example for Selecting Database
9. Example for Creating Table
10. Example for Inserting Record
11. Example for Selecting Record
12. Example for Updating Record
13. Example for Deleting Record

Creating JDBC Applications

- **Import the packages** - Requires that you include the packages containing the JDBC classes needed for database programming. Most often, using `import java.sql.*` will suffice.
- **Open a connection** - Requires using the `DriverManager.getConnection()` method to create a `Connection` object, which represents a physical connection with the database.
- **Execute a Query** - Requires using an object of type `Statement` for building and submitting an SQL statement to the database.
- **Extract data from Result set** - Requires that you use the appropriate `ResultSet.getXXX()` method to retrieve the data from the result set.
- **Clean up the environment** - Requires explicitly closing all database resources versus relying on the JVM's garbage collection.

1. Java JDBC
2. Why to Learn JDBC
3. Applications of JDBC
4. JDBC Architecture
5. Common JDBC Components
- 6. Creating JDBC Applications**
7. Example for Creating Database
8. Example for Selecting Database
9. Example for Creating Table
10. Example for Inserting Record
11. Example for Selecting Record
12. Example for Updating Record
13. Example for Deleting Record

Example for Creating Database

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;

public class JDBCExample {
    static final String DB_URL = "jdbc:mysql://localhost/";
    static final String USER = "guest";
    static final String PASS = "guest123";

    public static void main(String[] args) {
        // Open a connection
        try(Connection conn = DriverManager.getConnection(DB_URL, USER, PASS);
            Statement stmt = conn.createStatement();
        ) {
            String sql = "CREATE DATABASE STUDENTS";
            stmt.executeUpdate(sql);
            System.out.println("Database created successfully...");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

1. Java JDBC
2. Why to Learn JDBC
3. Applications of JDBC
4. JDBC Architecture
5. Common JDBC Components
6. Creating JDBC Applications
- 7. Example for Creating Database**
8. Example for Selecting Database
9. Example for Creating Table
10. Example for Inserting Record
11. Example for Selecting Record
12. Example for Updating Record
13. Example for Deleting Record

Example for Selecting Database

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;

public class JDBCExample {
    static final String DB_URL = "jdbc:mysql://localhost/Student";
    static final String USER = "guest";
    static final String PASS = "guest123";

    public static void main(String[] args) {
        System.out.println("Connecting to a selected database...");
        // Open a connection
        try(Connection conn = DriverManager.getConnection(DB_URL, USER, PASS);) {
            System.out.println("Connected database successfully...");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

1. Java JDBC
2. Why to Learn JDBC
3. Applications of JDBC
4. JDBC Architecture
5. Common JDBC Components
6. Creating JDBC Applications
7. Example for Creating Database
- 8. Example for Selecting Database**
9. Example for Creating Table
10. Example for Inserting Record
11. Example for Selecting Record
12. Example for Updating Record
13. Example for Deleting Record

Example for Creating Table

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;

public class TestApplication {
    static final String DB_URL = "jdbc:mysql://localhost/Student";
    static final String USER = "guest";
    static final String PASS = "guest123";

    public static void main(String[] args) {
        // Open a connection
        try(Connection conn = DriverManager.getConnection(DB_URL, USER, PASS);
            Statement stmt = conn.createStatement();
        ) {
            String sql = "CREATE TABLE REGISTRATION " +
                "(id INTEGER not NULL, " +
                " first VARCHAR(255), " +
                " last VARCHAR(255), " +
                " age INTEGER, " +
                " PRIMARY KEY ( id ))";

            stmt.executeUpdate(sql);
            System.out.println("Created table in given database...");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

1. Java JDBC
2. Why to Learn JDBC
3. Applications of JDBC
4. JDBC Architecture
5. Common JDBC Components
6. Creating JDBC Applications
7. Example for Creating Database
8. Example for Selecting Database
- 9. Example for Creating Table**
10. Example for Inserting Record
11. Example for Selecting Record
12. Example for Updating Record
13. Example for Deleting Record

Example for Inserting Record

```
public class JDBCExample {  
    static final String DB_URL = "jdbc:mysql://localhost/Student";  
    static final String USER = "guest";  
    static final String PASS = "guest123";  
  
    public static void main(String[] args) {  
        // Open a connection  
        try(Connection conn = DriverManager.getConnection(DB_URL, USER, PASS);  
            Statement stmt = conn.createStatement();  
        ) {  
            // Execute a query  
            System.out.println("Inserting records into the table...");  
            String sql = "INSERT INTO Registration VALUES (100, 'Zara', 'Ali', 18)";  
            stmt.executeUpdate(sql);  
            sql = "INSERT INTO Registration VALUES (101, 'Mahnaz', 'Fatma', 25)";  
            stmt.executeUpdate(sql);  
            sql = "INSERT INTO Registration VALUES (102, 'Zaid', 'Khan', 30)";  
            stmt.executeUpdate(sql);  
            sql = "INSERT INTO Registration VALUES (103, 'Sumit', 'Mittal', 28)";  
            stmt.executeUpdate(sql);  
            System.out.println("Inserted records into the table...");  
        } catch (SQLException e) {  
            e.printStackTrace();  
        }  
    }  
}
```

1. Java JDBC
2. Why to Learn JDBC
3. Applications of JDBC
4. JDBC Architecture
5. Common JDBC Components
6. Creating JDBC Applications
7. Example for Creating Database
8. Example for Selecting Database
9. Example for Creating Table
- 10. Example for Inserting Record**
11. Example for Selecting Record
12. Example for Updating Record
13. Example for Deleting Record

Example for Selecting Record

```
public class JDBCExample {  
    static final String DB_URL = "jdbc:mysql://localhost/Student";  
    static final String USER = "guest";  
    static final String PASS = "guest123";  
    static final String QUERY = "SELECT id, first, last, age FROM Registration";  
  
    public static void main(String[] args) {  
        // Open a connection  
        try(Connection conn = DriverManager.getConnection(DB_URL, USER, PASS);  
            Statement stmt = conn.createStatement();  
            ResultSet rs = stmt.executeQuery(QUERY);  
        ) {  
            while(rs.next()){  
                //Display values  
                System.out.print("ID: " + rs.getInt("id"));  
                System.out.print(", Age: " + rs.getInt("age"));  
                System.out.print(", First: " + rs.getString("first"));  
                System.out.println(", Last: " + rs.getString("last"));  
            }  
        } catch (SQLException e) {  
            e.printStackTrace();  
        }  
    }  
}
```

1. Java JDBC
2. Why to Learn JDBC
3. Applications of JDBC
4. JDBC Architecture
5. Common JDBC Components
6. Creating JDBC Applications
7. Example for Creating Database
8. Example for Selecting Database
9. Example for Creating Table
10. Example for Inserting Record
- 11. Example for Selecting Record**
12. Example for Updating Record
13. Example for Deleting Record

Example for Updating Record

```
public class JDBCExample {  
    static final String DB_URL = "jdbc:mysql://localhost/Student";  
    static final String USER = "guest";  
    static final String PASS = "guest123";  
    static final String QUERY = "SELECT id, first, last, age FROM Registration";  
  
    public static void main(String[] args) {  
        // Open a connection  
        try(Connection conn = DriverManager.getConnection(DB_URL, USER, PASS);  
            Statement stmt = conn.createStatement();  
        ) {  
            String sql = "UPDATE Registration " +  
                "SET age = 30 WHERE id in (100, 101)";  
            stmt.executeUpdate(sql);  
            ResultSet rs = stmt.executeQuery(QUERY);  
            while(rs.next()){  
                //Display values  
                System.out.print("ID: " + rs.getInt("id"));  
                System.out.print(", Age: " + rs.getInt("age"));  
                System.out.print(", First: " + rs.getString("first"));  
                System.out.println(", Last: " + rs.getString("last"));  
            }  
            rs.close();  
        } catch (SQLException e) {  
            e.printStackTrace();  
        }  
    }  
}
```

1. Java JDBC
2. Why to Learn JDBC
3. Applications of JDBC
4. JDBC Architecture
5. Common JDBC Components
6. Creating JDBC Applications
7. Example for Creating Database
8. Example for Selecting Database
9. Example for Creating Table
10. Example for Inserting Record
11. Example for Selecting Record
- 12. Example for Updating Record**
13. Example for Deleting Record

Example for Deleting Record

```
public class JDBCExample {
    static final String DB_URL = "jdbc:mysql://localhost/Student";
    static final String USER = "guest";
    static final String PASS = "guest123";
    static final String QUERY = "SELECT id, first, last, age FROM Registration";

    public static void main(String[] args) {
        // Open a connection
        try(Connection conn = DriverManager.getConnection(DB_URL, USER, PASS);
            Statement stmt = conn.createStatement();
        ) {
            String sql = "DELETE FROM Registration " +
                "WHERE id = 101";
            stmt.executeUpdate(sql);
            ResultSet rs = stmt.executeQuery(QUERY);
            while(rs.next()){
                //Display values
                System.out.print("ID: " + rs.getInt("id"));
                System.out.print(", Age: " + rs.getInt("age"));
                System.out.print(", First: " + rs.getString("first"));
                System.out.println(", Last: " + rs.getString("last"));
            }
            rs.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

1. Java JDBC
2. Why to Learn JDBC
3. Applications of JDBC
4. JDBC Architecture
5. Common JDBC Components
6. Creating JDBC Applications
7. Example for Creating Database
8. Example for Selecting Database
9. Example for Creating Table
10. Example for Inserting Record
11. Example for Selecting Record
12. Example for Updating Record
- 13. Example for Deleting Record**

THANK YOU