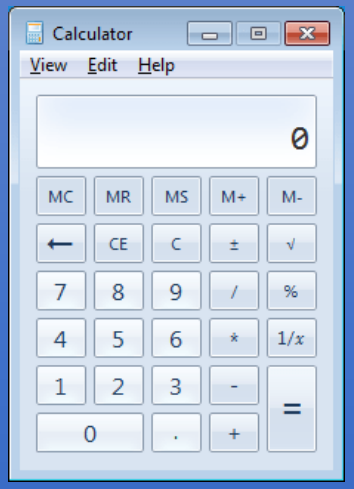
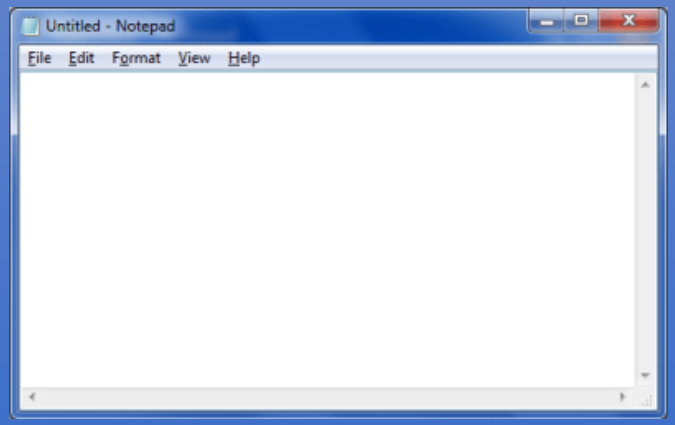
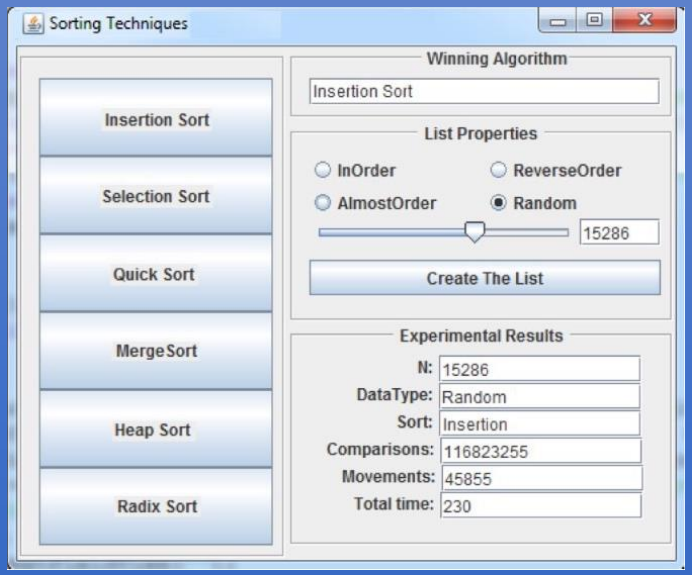






Graphical User Interface (GUI)

Java's AWT and Swing APIs

Lecture 8

Dr. Tamer ABUHMED
Java Programming Course (SWE2023)
College of Computing

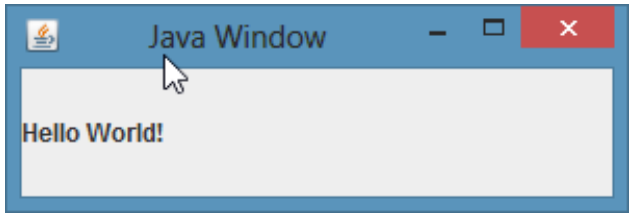
SUNG KYUN KWAN UNIVERSITY
1398

Outline

- More on Java GUI
- MVC model
  - Full Example
- WINDOWBUILDER
  - Example

1. “Hello World” example


```
import java.awt.Dimension;
import javax.swing.JFrame;
import javax.swing.JLabel;

/**
 * Simple demo of Java Swing GUI toolkit.
*/
public class HelloWorldGUI {
  public static void main(String[] args) {
      // set up the window
      JFrame frame = new JFrame();
      frame.setTitle("Java Window");
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      // add a label
      JLabel label = new JLabel("Hello World!");
      frame.add(label);

      // show the window
      frame.setSize(300, 100);
      frame.setVisible(true);
  }
}
```



Set up the window:

- JFrame acts as window component
- Set title for the window
- Assign action to close button/operation

Add a label:

- JLabel contains our message
- Add label to frame's content pane

Show the window:

- Set size for the frame
- Make frame visible

JButton Example

Buttons events handling by ActionListener

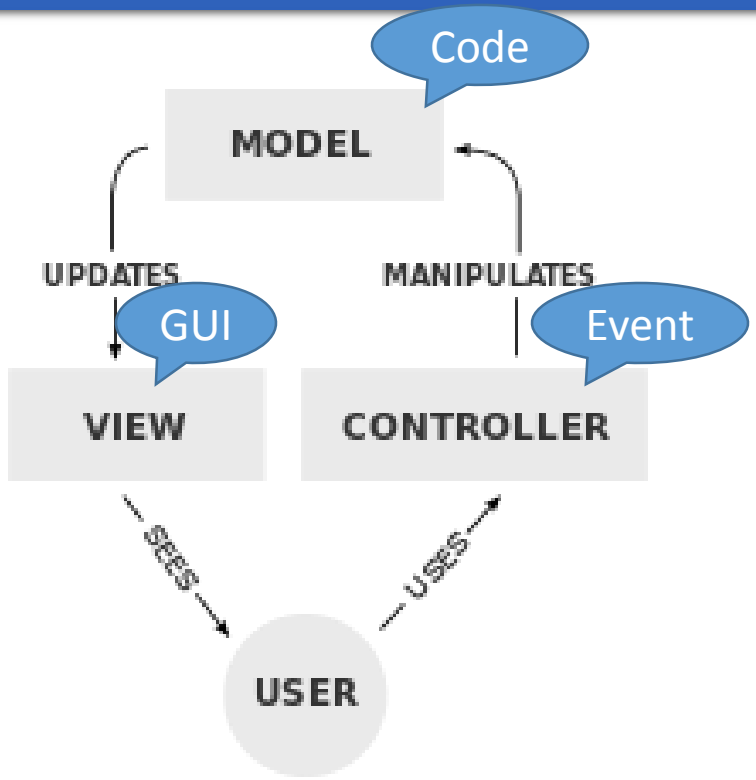
```
import java.awt.FlowLayout;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.Icon;
import javax.swing.ImageIcon;
import javax.swing.JOptionPane;

public class ButtonFrame extends JFrame
{
   private JButton plainJButton; // button with just text
   private JButton fancyJButton; // button with icons

   // ButtonFrame adds JButtons to JFrame
   public ButtonFrame()
   {
      super( "Testing Buttons" );
   setLayout( new FlowLayout() ); // set frame layout

   plainJButton = new JButton( "Plain Button" ); // button with text
      add( plainJButton ); // add plainJButton to JFrame

   Icon bug1 = new ImageIcon( getClass().getResource( "bug1.gif" ) );
   Icon bug2 = new ImageIcon( getClass().getResource( "bug2.gif" ) );
   fancyJButton = new JButton( "Fancy Button", bug1 ); // set image
   fancyJButton.setRolloverIcon( bug2 ); // set rollover image
   add( fancyJButton ); // add fancyJButton to JFrame

      // create new ButtonHandler for button event handling
      ButtonHandler handler = new ButtonHandler();
      fancyJButton.addActionListener( handler );
      plainJButton.addActionListener( handler );
   } // end ButtonFrame constructor
```

```
// inner class for button event handling
   private class ButtonHandler implements ActionListener
   {
      // handle button event
      public void actionPerformed( ActionEvent event )
      {
        JOptionPane.showMessageDialog( ButtonFrame.this,
String.format("You pressed: %s", event.getActionCommand() ) );
      } // end method actionPerformed
   } // end private inner class ButtonHandler
} // end class ButtonFrame
```

```
import javax.swing.JFrame;

public class ButtonTest

{

   public static void main( String[] args )

   {

      ButtonFrame buttonFrame = new ButtonFrame(); // create Frame

      buttonFrame.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );

      buttonFrame.setSize( 300, 200 ); // set frame size

      buttonFrame.setVisible( true ); // display frame

   } // end main

} // end class ButtonTest
```

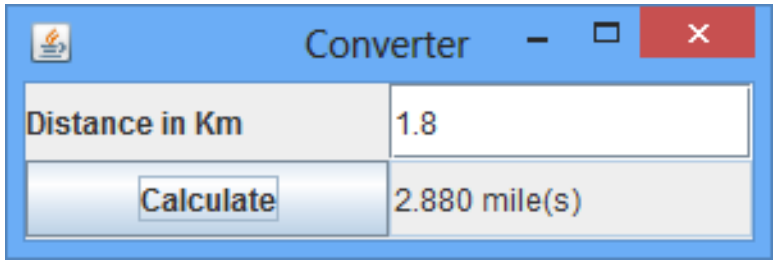
2. MVC model

- Model-View-Controller design pattern
- Isolate data model from user interface from application logic
  - **Model**: data model classes
  - **View**: user interface (i.e. GUI, console, etc.)
  - **Controller**: interacts with View, manipulates Model



MVC model SWING example (Video)

3. Converter application



- **Problem:**
  - Create a GUI program to convert the distance from kilometer to mile (1 km = 0.621 mile)
- **Model (Data):**
  - We don't have to store any data so no need
- **View (GUI):**
  - 1 label
  - 1 input text
  - 1 button
  - 1 display text
- **Controller (User interactive)**
  - Click the button

To control more about the window and can be reusable, we create our own object extending JFrame

3. Converter – Code (1)

- Begin by defining the class & attributes
  - `public class ConverterFrame extends JFrame {`

```
private JLabel lblDistance;
private JTextField txtKmDistance;
private JButton btnCalculate;
private JTextField txtMileDistance;
```

- Start by setting up the window
  - `public void showWindow(){`

```
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setSize(300,100);
setVisible(true);
}
```

3. Converter – Code (2)

- Initialize GUI window by adding components
  - Next, set up the individual components

```
public ConverterFrame(){
// call super constructor to set the name
   super("Converter");
   lblDistance = new JLabel("Distance in Km");
   txtKmDistance = new JTextField(10);
   btnCalculate = new JButton("Calculate");
   txtMileDistance = new JTextField(50);
   txtMileDistance.setEditable(false);
```

  - Then, initialize window layout & add components

```
   setLayout( new GridLayout( 2, 2 ));
   this.add(lblDistance);
   this.add(txtKmDistance);
   this.add(btnCalculate);
   this.add(txtMileDistance);
```

3. Converter – Code (3)

- Finally, create action listener for button

```
public class CalculateHandler implements ActionListener {
    @Override
    public void actionPerformed(ActionEvent e) {
        double km = Double.parseDouble(txtKmDistance.getText());
        String ret = String.format("%.3f mile(s)", km * 0.621);
        txtMileDistance.setText(ret);
    }
}
```

- and assign to button

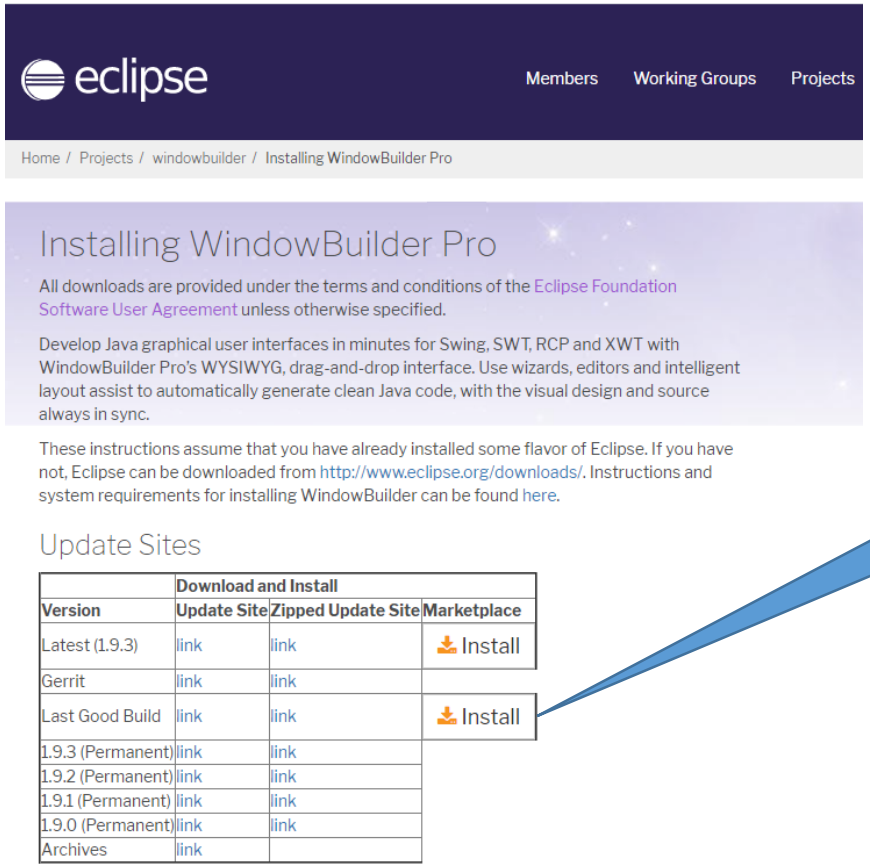
```
btnCalculate.addActionListener(new CalculateHandler());
```

WINDOWBUILDER

Tool to create Java GUI applications

1. Install WindowBuilder

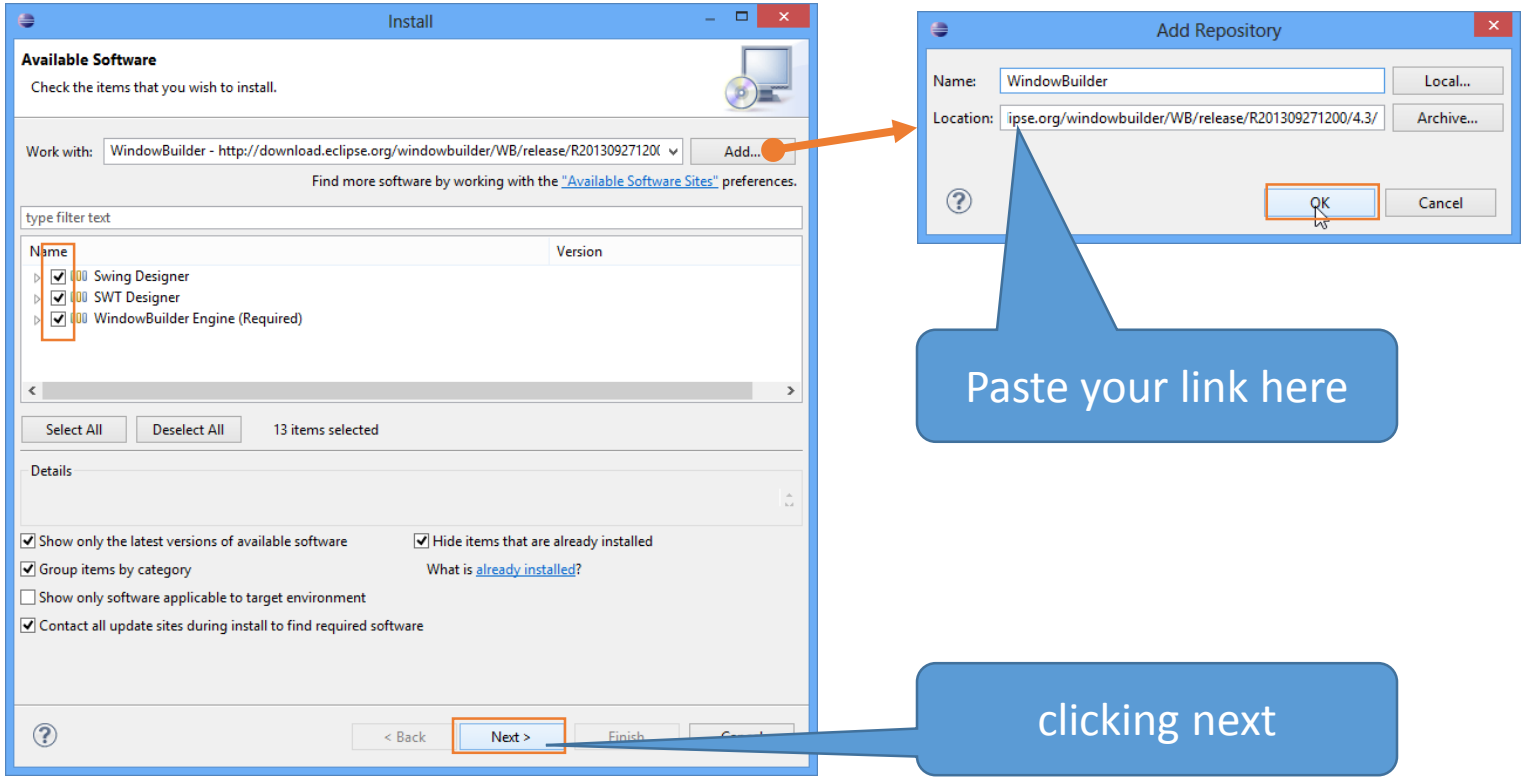
- Go to this website and copy the link based on your Eclipse version: http://www.eclipse.org/windowbuilder/download.php



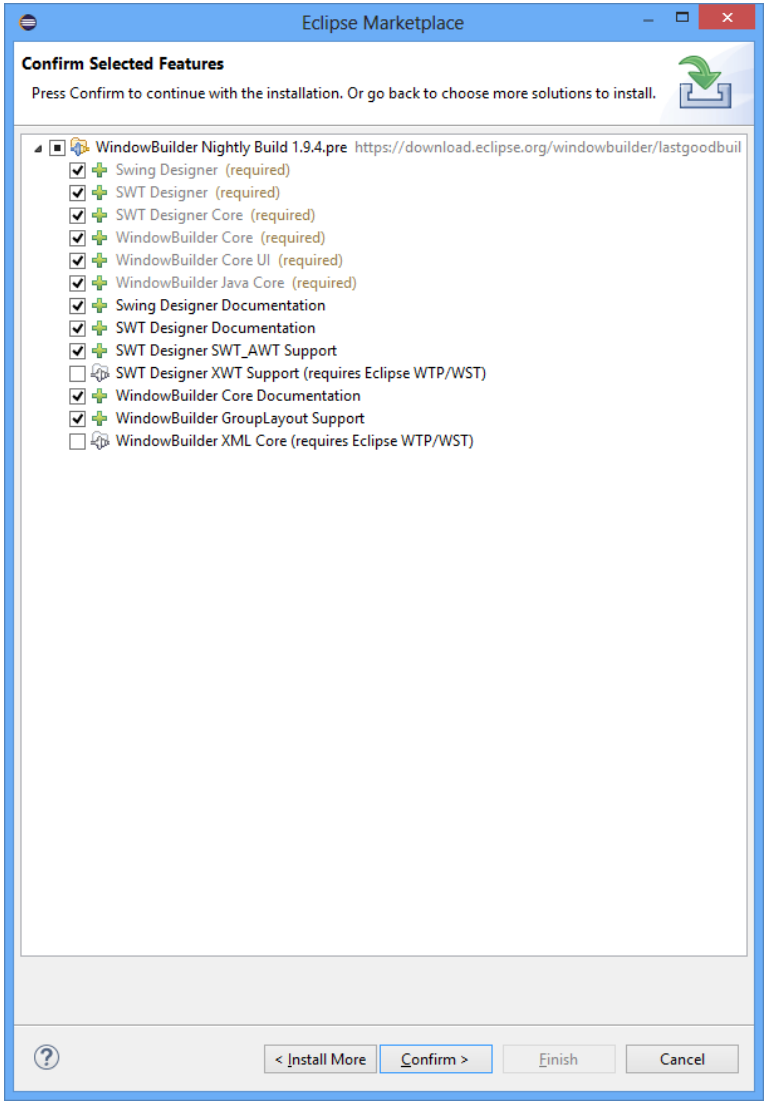
Right click on this link and copy the link address or drag and drop install to eclipse marketplace

1. Install WindowBuilder

- Go to Eclipse, Menu Help -> Install New Software

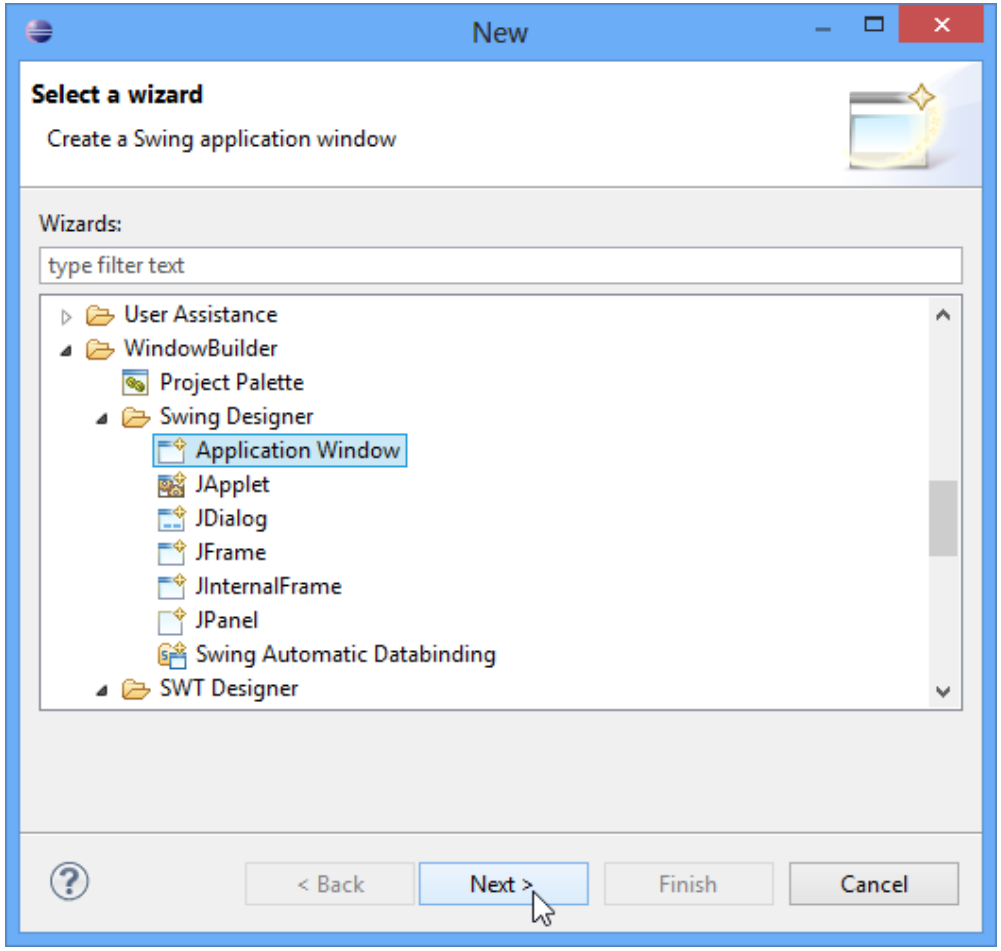


WindowBuilder components

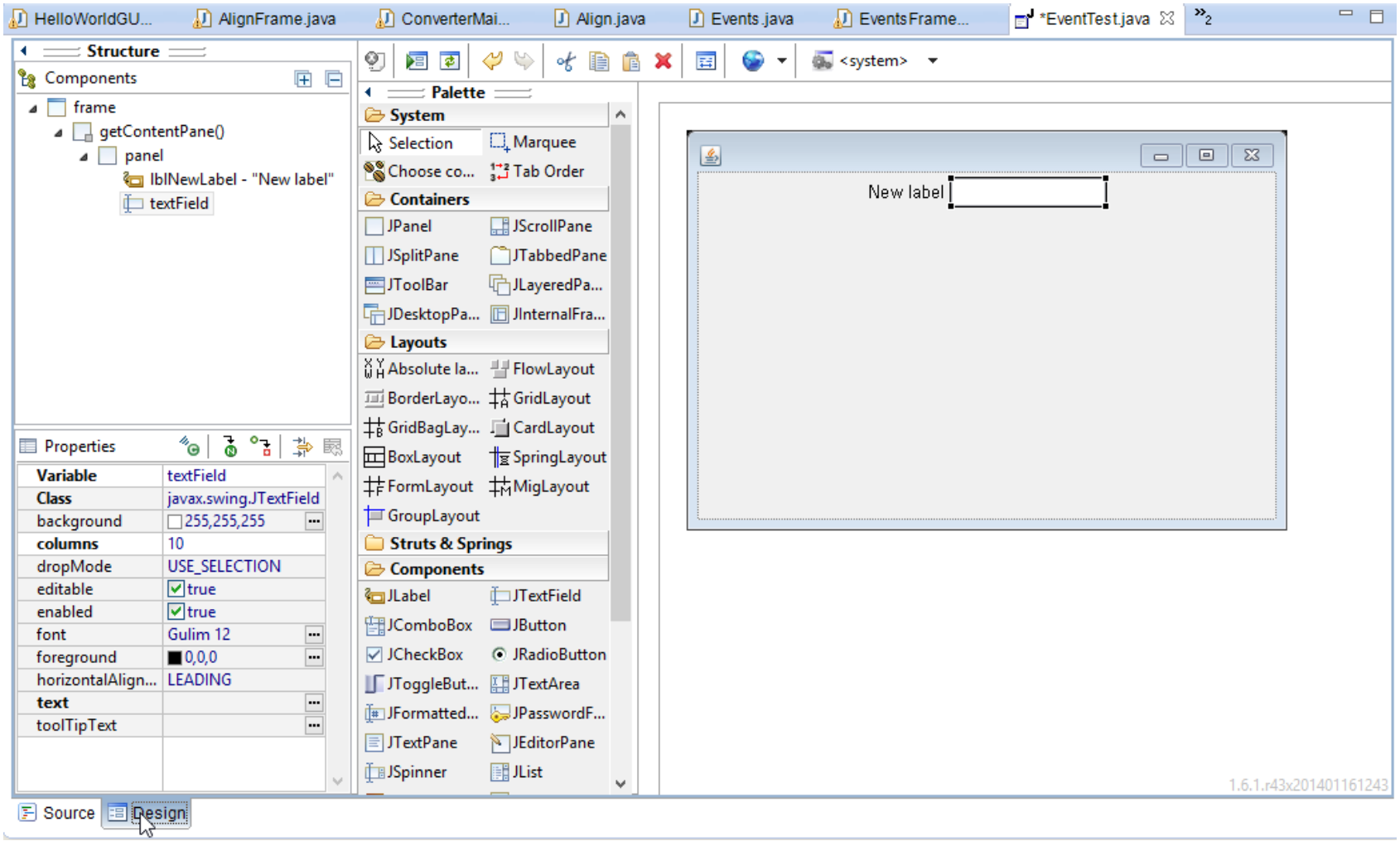


2. Using WindowBuilder
New Application

- File → New → Other... → WindowBuilder → Swing Designer → Application Window.



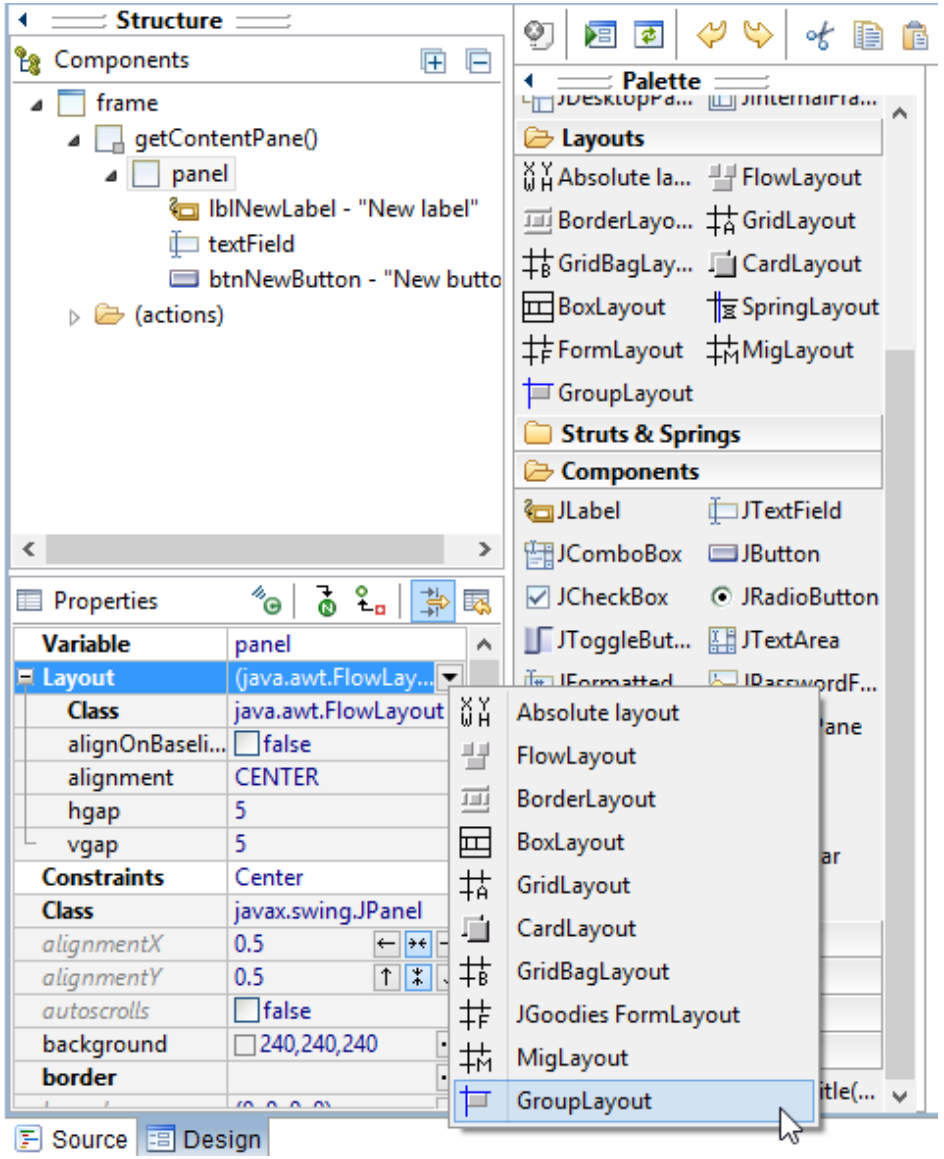
2. Using WindowBuilder
Design Tab



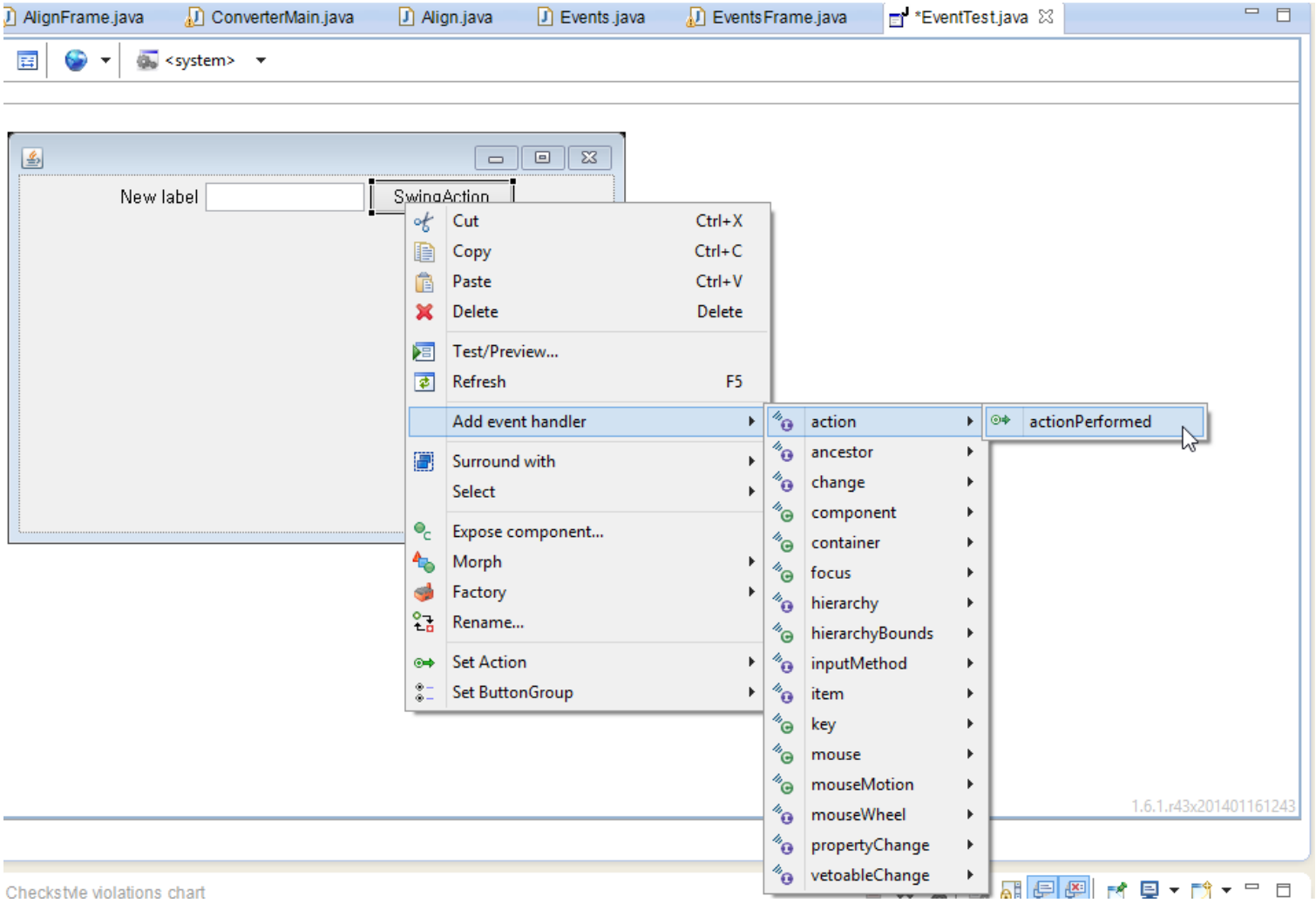
2. Using WindowBuilder

Layout

- Change the layout in the Properties window
- Or right click on the panel object -> Set Layout

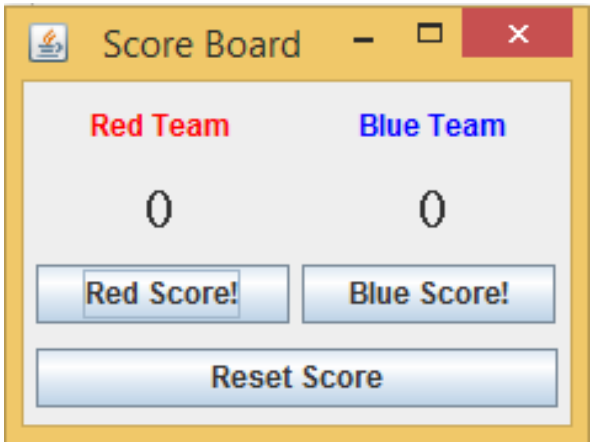


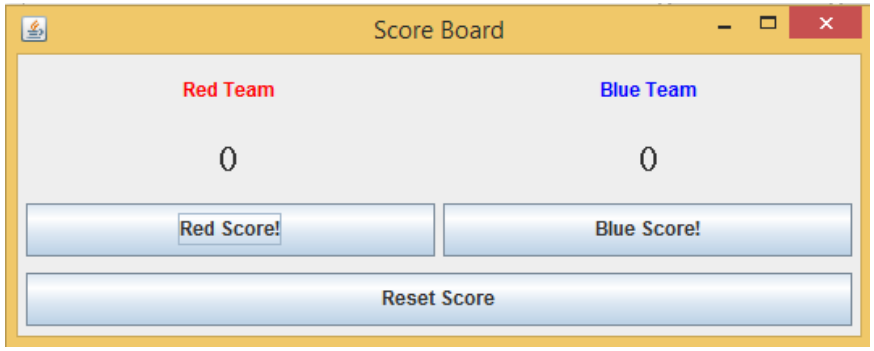
2. Using WindowBuilder
Event handling



Practice: Scoreboard

- Create a scoreboard Java application displaying score of 2 teams. User can increase the score of each team and reset the score of 2 teams.
- Challenge (you don't have to do this): Make sure the components still look good when you resize the window.





Summary

- Introduction to Java GUI
- AWT and Swing
- GUI Terminology
  - Swing Components
  - GUI Window: JFrame
  - GUI Component: Jbutton
- GUI Sizing and positioning
  - Containers and Layout
- Graphical Events
- Event Listeners
  - Action Events
  - Implementing a listener