A. Kompetensi

Setelah mengikuti praktikum ini, mahasiswa diharapkan mampu menggunakan Sliders serta Change Events.

B. Dasar Teori

The `Jslider` class represents a graphical slider. Like scrollbars, sliders can have either a horizontal or vertical orientation. With sliders, however, you can enhance their appearance with tick marks and labels. In most instances, a slider is preferable to a standalone scrollbar. Sliders represent a selection of one value from a bounded range. Scrollbars represent a range of values within a bounded range and are best used in things like the `JscrollPane`. The `Jslider` class allows you to set the spacing of two types of tick marks: major and minor. Major tick marks are longer than minor tick marks and are generally used at wider intervals. Figure 1 shows various sliders that can be composed in Swing.

Figure 1. Various sliders in Swing

Figure 2. `Jslider` component with horizontal orientation

`Jslider` enable the user to select from a range of integer values. Class `Jslider` inherits from `JComponent`. Figure 2 shows a horizontal `Jslider` with **tick marks** and the **thumb** that allows the user to select a value. `Jslider` can be customized to display major tick marks, minor tick marks and labels for the tick marks. They also support **snap-to ticks**, which cause the thumb to snap to the closest tick mark when it is positioned between two tick marks.

Constructors

public JSlider()

Creates a horizontal slider with the range 0 to 100 and an initial value of 50.

public JSlider(int orientation)

Creates a slider using the specified orientation with the range 0 to 100 and an initial value of 50.

public JSlider(int min, int max)

Creates a horizontal slider using the specified min and max with an initial value equal to the average of the min plus max.

public JSlider(int min, int max, int value)

Creates a horizontal slider using the specified min, max and value.

publi c JSlider(int orientation, int minimum, int maximum, int value)

Creates a slider with the specified orientation and the specified minimum, maximum, and initial values.

public JSlider(BoundedRangeModel brm)

Creates a horizontal slider using the specified BoundedRangeModel.

Set the initial values of the slider. The orientation must be either `JSlider.HORIZONTAL` or `JSlider.VERTICAL`. If anything else is passed in, the `JSlider` object throws a runtime `IllegalArgumentException`. The remaining parameters are used to initialize the slider's bounded-range model. If the parameters are not given, they are initialized to the default values.

The `setPaintTicks()` method sets a `boolean`, which is used to activate or deactivate the slider's tick marks. In some L&Fs, the slider changes from a rectangular shape to a pointer when tick marks are activated. This is often done to give the user a more accurate representation of where the slider falls.

`JSlider` triggers a `ChangeEvent` whenever the user modifies any of its properties. It also generates a `PropertyChangeEvent` whenever any of its properties change.

public void addChangeListener(ChangeListener l)
public void removeChangeListener(ChangeListener l)

Add or remove a specific listener from receiving property change events generated by the `JSlider` object.

Listing 1, for drawing circles of a specified diameter.

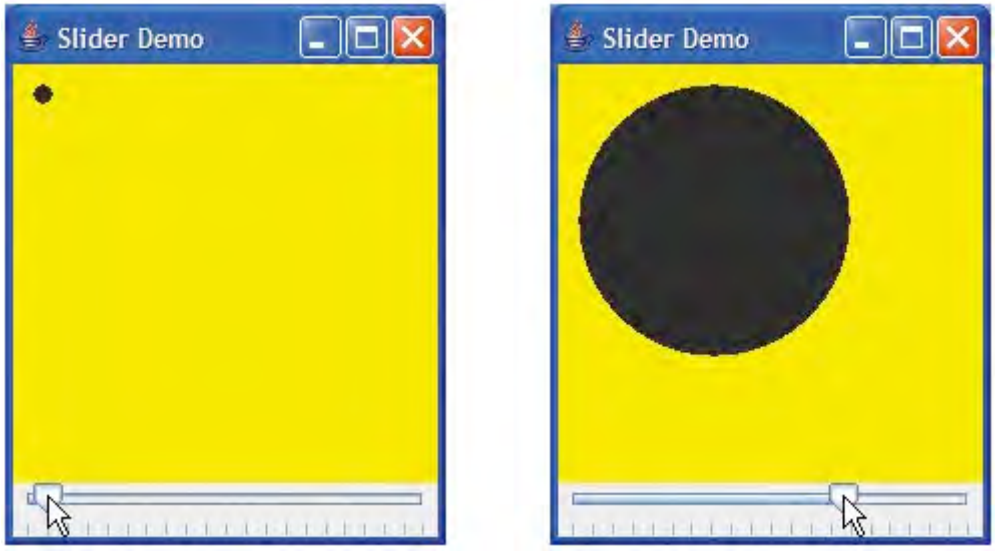
```
import java.awt.Graphics;
import java.awt.Dimension;
import javax.swing.JPanel;

public class OvalPanel extends JPanel  {
   private int diameter = 10 ;   // default diameter of 10

   // draw an oval of the specified diameter
```

```
public void paintComponent(Graphics g)    {
    super .paintComponent(g);
    g.fillOval( 10, 10, diameter, diameter ); // draw circle
   } // end method paintComponent

     // validate and set diameter, then repaint
public void setDiameter(int newDiameter)  {
       // if diameter invalid, default to 10
       diameter = ( newDiameter >= 0 ? newDiameter : 10 );
       repaint(); // repaint panel
    } // end method setDiameter

    // used by layout manager to determine preferred size
public Dimension getPreferredSize()  {
       return new Dimension( 200, 200 );
    } // end method getPreferredSize

    // used by layout manager to determine minimum size
public Dimension getMinimumSize() {
       return getPreferredSize();
    } // end method getMinimumSize
} // end class OvalPanel
```

Listing 2, `JSlider` value used to determine the diameter of a circle.

```
import java.awt.BorderLayout;
import java.awt.Color;
import javax.swing.JFrame;
import javax.swing.JSlider;
import javax.swing.SwingConstants;
import javax.swing.event.ChangeListener;
import javax.swing.event.ChangeEvent;

public class SliderFrame extends JFrame  {
    private JSlider diameterJSlider; // slider to select diameter
    private OvalPanel myPanel; // panel to draw circle

    // no-argument constructor
public SliderFrame()    {
       super ( "Slider Demo" );
       myPanel = new OvalPanel(); // create panel to draw circle
       myPanel.setBackground( Color.YELLOW ); // set background to yellow
```

```
      // set up JSlider to control diameter value
      diameterJSlider =
        new JSlider( SwingConstants.HORIZONTAL, 0, 200, 10 );
      diameterJSlider.setMajorTickSpacing( 10 ); // create tick every 10
      diameterJSlider.setPaintTicks( true ); // paint ticks on slider

      // register JSlider event listener
      diameterJSlider.addChangeListener(

        new ChangeListener() // anonymous inner class
        {
          // handle change in slider value
          public void stateChanged( ChangeEvent e )
          {
            myPanel.setDiameter( diameterJSlider.getValue() );
          } // end method stateChanged
        } // end anonymous inner class
      );  // end call to addChangeListener
      add( diameterJSlider, BorderLayout.SOUTH ); // add slider to frame
      add( myPanel, BorderLayout.CENTER ); // add panel to frame
    } // end SliderFrame constructor

 public static void main( String args[] )  {
      SliderFrame sliderFrame = new SliderFrame();
      sliderFrame.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      sliderFrame.setSize( 220, 270 );  // set frame size
      sliderFrame.setVisible( true );  // display frame1
  } // end main
} // end class SliderFrame
```



C. Alat/ Bahan

1. Lab. Sheet Pemrograman Java 12
2. PC / Laptop with OS installed
3. JDK 1.5 or latest
4. J-Creator or text editor

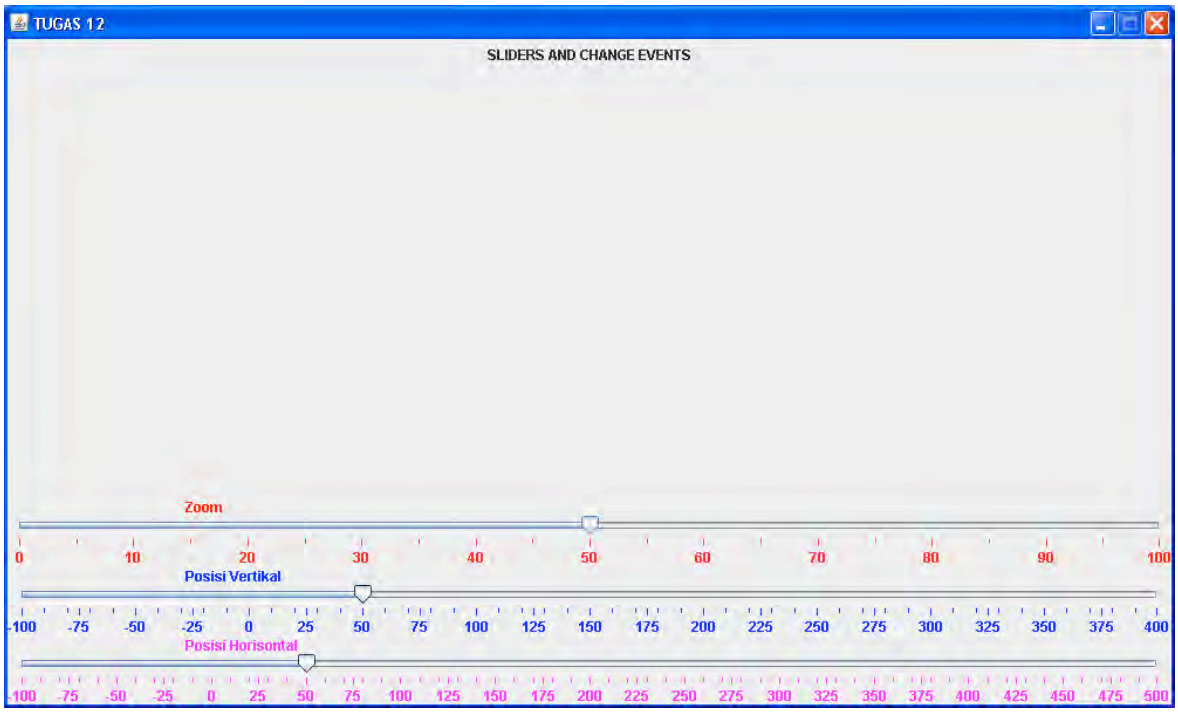
D. Langkah Kerja

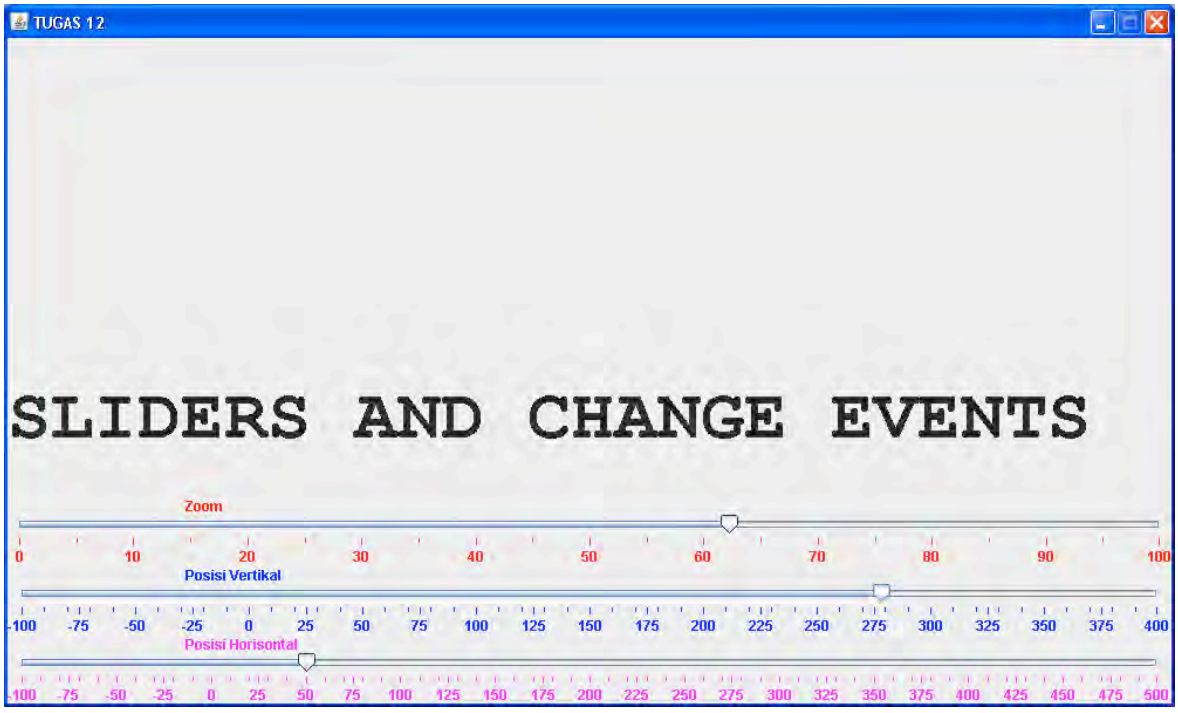
1. Baca dan pahami dasar teori di atas.
2. Lakukan kompilasi dan eksekusi terhadap contoh-contoh source code atau program yang ada di dasar teori dan LAMPIRAN.
3. Kerjakan tugas individu di bawah.

E. Tugas Individu

Buatlah sebuah aplikasi yang menggunakan Slider serta component lain, seperti contoh di bawah ini.

- ✓ Slider Zoom untuk mengatur ukuran label “SLIDERS AND CHANGE EVENTS”.
- ✓ Slider Posisi Vertikal untuk mengatur ukuran label “SLIDERS AND CHANGE EVENTS”.
- ✓ Slider Posisi Horisontal untuk mengatur ukuran label “SLIDERS AND CHANGE EVENTS”.





F. Lampiran

- **CH 64** (Introduction to Computer Science using Java, Java 5.0 version, January 2006, Bradley Kjell, Central Connecticut State University http://chortle.ccsu.edu/CS151/cs151java.html)

- **Java™ Swing, 2nd Edition,** Brian Cole, Robert Eckstein, James Elliott, Marc Loy, David Wood