INSTITUT TEKNOLOGI DEL

MATERI PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK

Topik	:	Swing dan Event Handling
Tujuan	:	Mahasiswa dapat membuat aplikasi Java berbasis desktop.

I. Membuat aplikasi desktop dari Nol

1. Berikut ini adalah kode program (lihat slide 16 dari materi *event handling*) yang telah dimodifikasi sehingga Mouse Adapter di-extend pada outer class.

```
1   import java.awt.FlowLayout;
2   import java.awt.event.MouseAdapter;
3   import java.awt.event.MouseEvent;
4   import java.awt.Color;
5   import javax.swing.JFrame;
6   import javax.swing.JLabel;
7   import javax.swing.JButton;
8
9   public class DemoEventHandlingAdapterOuter extends MouseAdapter
10  {
11        private JFrame myFrame;
12        private JLabel myLabel;
13        private JButton myButton1;
14        private JButton myButton2;
15
16        //constructor
17        public DemoEventHandlingAdapterOuter(String title){
18              myFrame = new JFrame(title);
19              myLabel = new JLabel("Default Text");
20              myLabel.setOpaque(true);
21              myLabel.setBackground(Color.lightGray);
22              myButton1 = new JButton("Button1");
23              myButton2 = new JButton("Button2");
24
25              System.out.println(myFrame.getLayout());
26
27              myFrame.setLocationRelativeTo(null);
28              myFrame.setSize(240,100);
29              myFrame.setVisible(true);
30              myFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
31        }
32        public void setLayout(){
33              myFrame.setLayout(new FlowLayout());
34        }
35        //method untuk add komponen pada kontainer
36        private void addComponent(){
37              myFrame.add(myButton1);
38              myFrame.add(myButton2);
39              myFrame.add(myLabel);
40        }
```

```
41
42          //method untuk meregister objek pada objek listener
43          private void addListener(){
44                myButton1.addMouseListener(this);
45                myButton2.addMouseListener(this);
46          }
47
48          //override method
49          public void mouseClicked(MouseEvent e) {
50                if (e.getSource() == myButton1)
51                myLabel.setText("Button 1 clicked");
52              else
53                myLabel.setText("Button 2 clicked");
54          }
55
56          public static void main(String[] args){
57                DemoEventHandlingAdapterOuter deho = new
58    DemoEventHandlingAdapterOuter("Demo Event Handling");
59                deho.setLayout();
60                deho.addComponent();
61                deho.addListener();
62          }
63    }
```

Catatan: sebelum menggunakan method **`setBackground()`**(baris 21) atau **`setForeground()`** pada objek **`JLabel`**, anda harus set true pada method **`setOpaque()`.**

Tuliskan perbedaan kode program di atas dengan kode program pada slide 16. Kenapa pada baris 44 dan 45 method **`addMouseListener()`** menerima parameter **`this?`**

2. Berikut ini adalah kode program dari slide 13 yang telah dimodifikasi; event listener MouseListener (Low Level events) diganti dengan ActionListener (Semantic events).

```
import java.awt.FlowLayout;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.awt.Color;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JButton;
public class DemoEventHandlingActionListener implements ActionListener{
      private JFrame myFrame;
      private JLabel myLabel;
      private JButton myButton1;
      private JButton myButton2;
      public DemoEventHandlingActionListener(String title){
            myFrame = new JFrame(title);
            myLabel = new JLabel("Default Text");
            myLabel.setOpaque(true);
            myLabel.setBackground(Color.lightGray);
```

```
            myButton1 = new JButton("Button1");
            myButton2 = new JButton("Button2");

            myFrame.setLocationRelativeTo(null);
            myFrame.setSize(240,100);
            myFrame.setVisible(true);
            myFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      }
      public void setLayout(){
            myFrame.setLayout(new FlowLayout());
      }
      private void addComponent(){
            myFrame.add(myButton1);
            myFrame.add(myButton2);
            myFrame.add(myLabel);
      }
      private void addListener(){
            myButton1.addActionListener(this);
            myButton2.addActionListener(this);
      }
      public void actionPerformed(ActionEvent e){
            if (e.getSource() == myButton1)
                  myLabel.setText("Button 1 clicked");
            else
                  myLabel.setText("Button 2 clicked");
      }
      public static void main(String[] args){
            DemoEventHandlingActionListener deho = new
DemoEventHandlingActionListener("Demo Event Handling");
            deho.setLayout();
            deho.addComponent();
            deho.addListener();

}
```

Catatan:

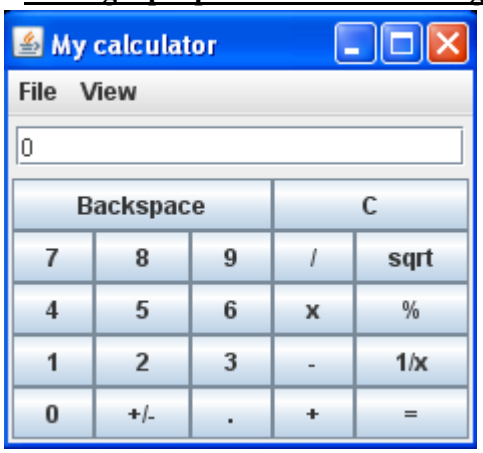
- Bandingkan hasil eksekusi dari program di atas ketika menggunakan MouseListener (MouseAdapter) dengan ketika menggunakan ActionListener sebagai event listener.
- Apakah ActionLister memiliki kelas adapter seperti MouseAdapter pada MouseListener?

3. Berikut ini adalah modifikasi terhadap program DemoEventHandlingInner.java dengan menggunakan Anonymous class (kelas tak bernama).

```
import java.awt.FlowLayout;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.Color;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JButton;
public class DemoEventHandlingInnerAnonymous extends JFrame{
```

```
	private JLabel myLabel;
	private JButton myButton1;
	private JButton myButton2;

	public DemoEventHandlingInnerAnonymous(String title){
		super(title);
		myLabel = new JLabel("Default Text");
		myLabel.setOpaque(true);
		myLabel.setBackground(Color.lightGray);
		myButton1 = new JButton("Button1");
		myButton2 = new JButton("Button2");

		setLocationRelativeTo(null);
		setSize(240,100);
		setVisible(true);
		setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
	}
	public void setLayout(){
		this.setLayout(new FlowLayout());
	}
	private void addComponent(){
		add(myButton1);
		add(myButton2);
		add(myLabel);
	}
	private void addListener(){
		myButton1.addMouseListener(new MouseAdapter(){
			public void mouseClicked(MouseEvent e) {
				if (e.getSource() == myButton1)
					myLabel.setText("Button 1 clicked");
				else
					myLabel.setText("Button 2 clicked");
			}
		});

		myButton2.addMouseListener(new MouseAdapter(){
			public void mouseClicked(MouseEvent e) {
				if (e.getSource() == myButton1)
					myLabel.setText("Button 1 clicked");
				else
					myLabel.setText("Button 2 clicked");
			}
		});
	}

	public static void main(String[] args){
		DemoEventHandlingInnerAnonymous deho = new
DemoEventHandlingInnerAnonymous("Demo Event Handling");
		deho.setLayout();
		deho.addComponent();
		deho.addListener();
	}
}
```

II. Melengkapi aplikasi kalkulator dengan event handling



Latihan: Lengkapi aplikasi kalkulator di atas sehingga bisa melakukan operasi matematika, minimal 2 operand.