D3 Teknik Informatika (TI)
Jurusan Teknik Informatika
Politeknik Negeri Indramayu

Event-Driven Programming

Fachrul Pralienka Bani Muhamad, S.ST., M.Kom.
fachrul.pbm@polindra.ac.id

2025

Tujuan

Setelah materi ini disampaikan, mahasiswa diharapkan mampu:

- **Menginteraksikan** tampilan UI aplikasi menggunakan event-listener
- **Menangani** (handling) event yang terjadi pada UI aplikasi

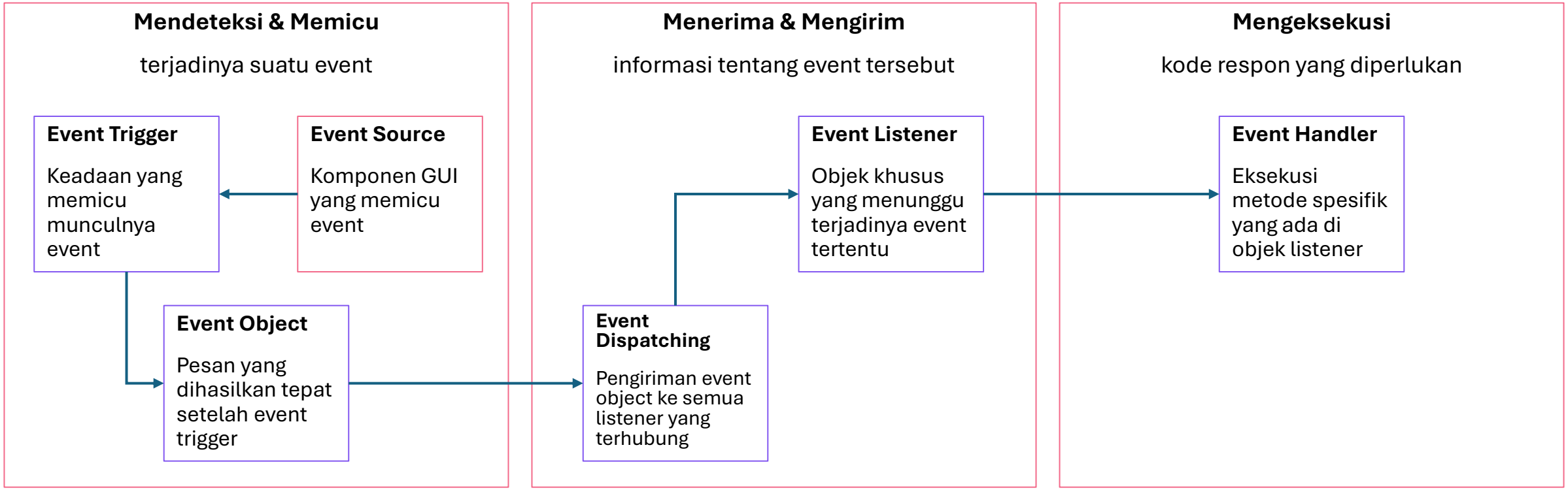
Konsep Event-Driven Programming

- Paradigma desain perangkat lunak di mana alur eksekusi program utamanya ditentukan oleh peristiwa (events) yang terjadi
- Alur eksekusi program dikendalikan oleh suatu event, tidak sekuensial (baris per baris)
- Pemrograman sekuensial tradisional dieksekusi dari atas ke bawah, sedangkan event-driven programming menunggu input atau aksi terhadapnya

Event Handling

- Mekanisme spesifik dari paradigma event-driven programming, yang digunakan untuk merespon suatu peristiwa (event) yang terjadi, atau perubahan status internal sistem
- Rangkaian proses dalam merespon event mencakup proses mendeteksi event, menerima event, dan mengeksekusi respon

Event Handling



Kategori Event Handling pada Swing

Kategori	Deskripsi	Event Object	Keterangan	Event Listener (Adapter)	Event Source
Semantic	Interaksi pengguna tingkat tinggi yang bermakna (fokus pada tujuan / aksi pengguna)	ActionEvent	Aksi selesai dilakukan	ActionListener	JButton, JTextField, JMenuItem, JComboBox
		ItemEvent	Status pilihan item berubah	ItemListener	JCheckBox, JRadioButton, JToggleButton, JComboBox
		ChangeEvent	Status komponen berubah secara umum	ChangeListener	JSlider, JSpinner, JTabbedPane, JProgressBar
		AdjustmentEvent	Nilai komponen yang dapat disesuaikan	AdjustmentListener	JScrollBar, JScrollPane
Low-level	Interaksi fisik langsung dengan hardware (mouse, keyboard, window)	MouseEvent	Detail gerakan atau klik mouse	MouseListener **(MouseAdapter)**, MouseMotionListener **(MouseMotionAdapter)**	Semua komponen GUI (misalnya, JPanel, JFrame, JButton)
		KeyEvent	Detail penekanan tombol fisik pada keyboard	KeyListener **(KeyAdapter)**	Komponen yang sedang fokus (JTextField, JTextArea, JFrame)
		WindowEvent	Perubahan status jendela	WindowListener **(WindowAdapter)**	JFrame, JDialog
		FocusEvent	Fokus input keyboard berpindah dari satu komponen ke komponen lain	FocusListener **(FocusAdapter)**	Semua komponen GUI yang bisa mendapat fokus

Kategori Event Handling pada Swing

Kategori	Deskripsi	Event Object	Keterangan	Event Listener (Adapter)	Event Source
Specific	Perubahan status internal komponen atau struktur GUI	ComponentEvent	Komponen dipindahkan, diubah ukurannya, disembunyikan, atau ditampilkan	ComponentListener **(ComponentAdapter)**	Semua komponen GUI
		ContainerEvent	Komponen ditambahkan ke atau dihapus dari container	ContainerListener **(ContainerAdapter)**	JPanel, JFrame, dan komponen container lainnya
		DocumentEvent	Isi teks pada komponen Swing berubah	DocumentListener	JTextField, JTextArea
		ListSelectionEvent	Pilihan item dalam daftar berubah	ListSelectionListener	JList, JTable

Implementasi Event Handling

Penerapan event handling pada Java Swing dapat dilakukan dengan 2 cara, yaitu anonymous class dan lambda expression

Event Handling (Anonymous class)

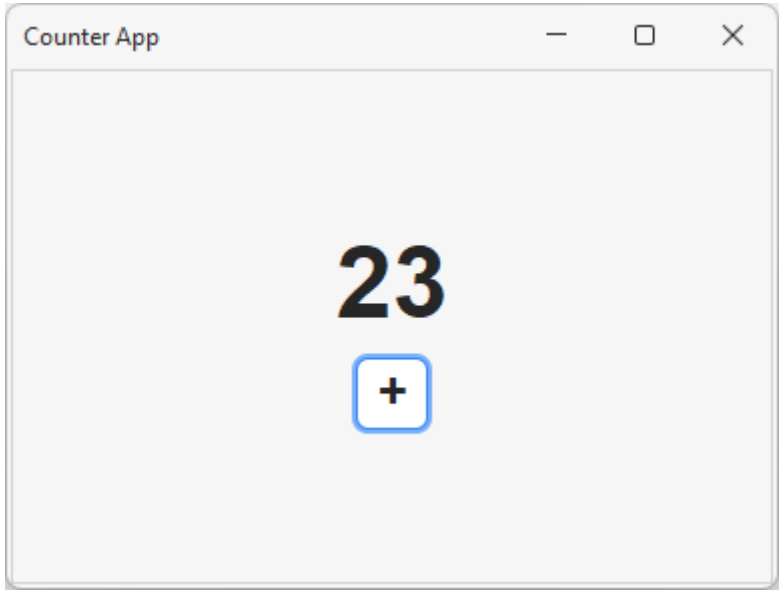
```
tombol.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {

    }
});
```

Event Handling (Lambda expression)

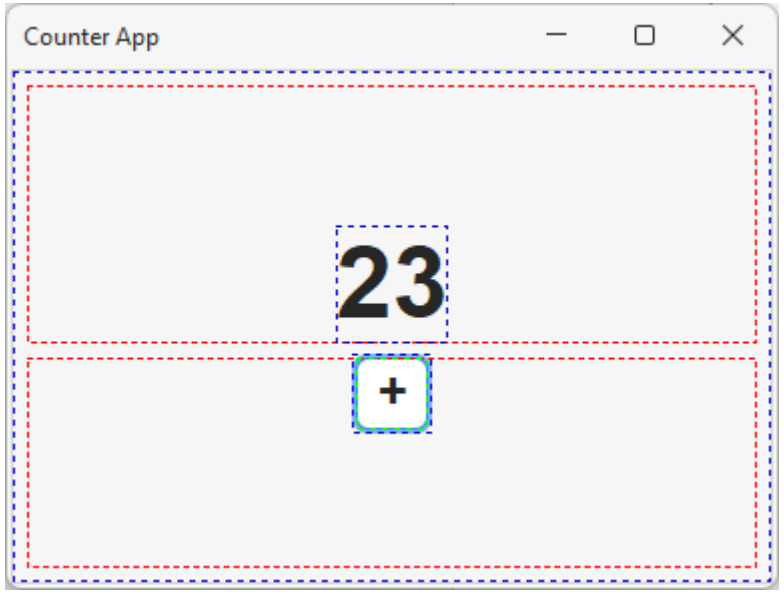
```
tombol.addActionListener(e -> {

});
```

* *Lambda expression **hanya dapat digunakan** pada Interface Listener yang hanya memiliki 1 method abstract, misalnya ActionListener. Lambda expression **tidak dapat digunakan** pada Abstract Class Adapter, atau Interface Listener yang memiliki lebih dari 1 method abstract, misalnya MouseListener.*

Latihan



Latihan



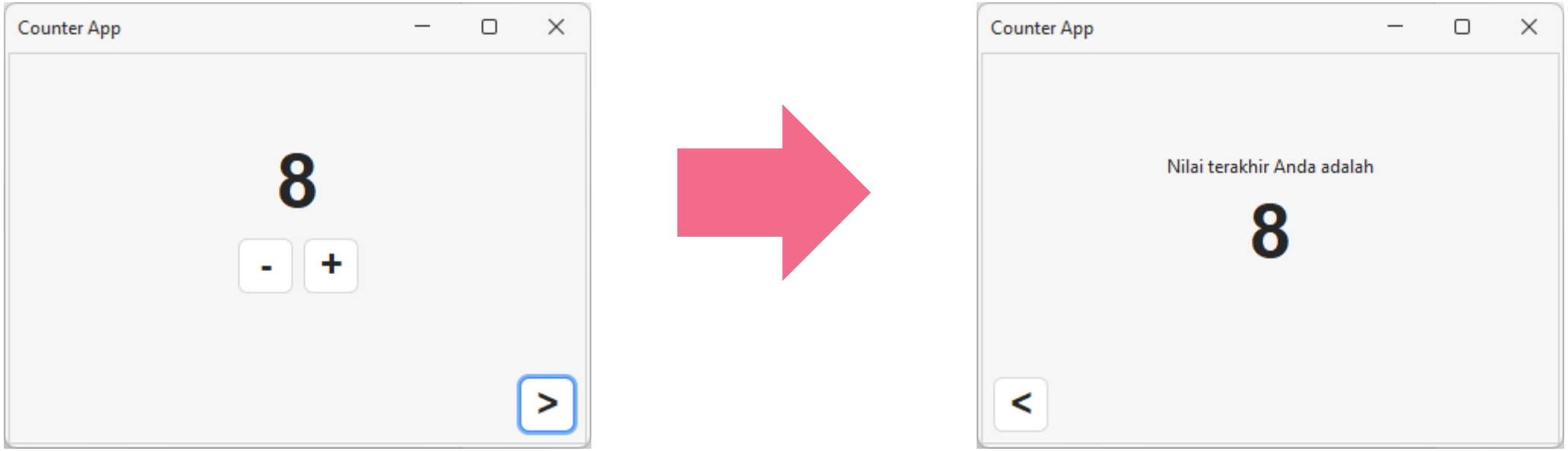
Latihan

```
public class CounterApp {

    private static int nilaiCounter = 0;

    public static void main(String[] args) {

        try {
            UIManager.setLookAndFeel(new FlatMacLightLaf());
        } catch (UnsupportedLookAndFeelException e) {
            e.printStackTrace();
        }

        SwingUtilities.invokeLater(() -> {

            JFrame frame = new JFrame("Counter App");
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            frame.setPreferredSize(new Dimension(400, 300));
            frame.setMinimumSize(new Dimension(200, 150));

            JPanel panelCounter = new JPanel(new MigLayout("fill, wrap 1", "[grow, center]", "[grow][grow]"));

            JLabel labelCounter = new JLabel(nilaiCounter + "");
            labelCounter.setFont(new Font("Arial", Font.BOLD, 50));
            panelCounter.add(labelCounter, "bottom");
```

Latihan

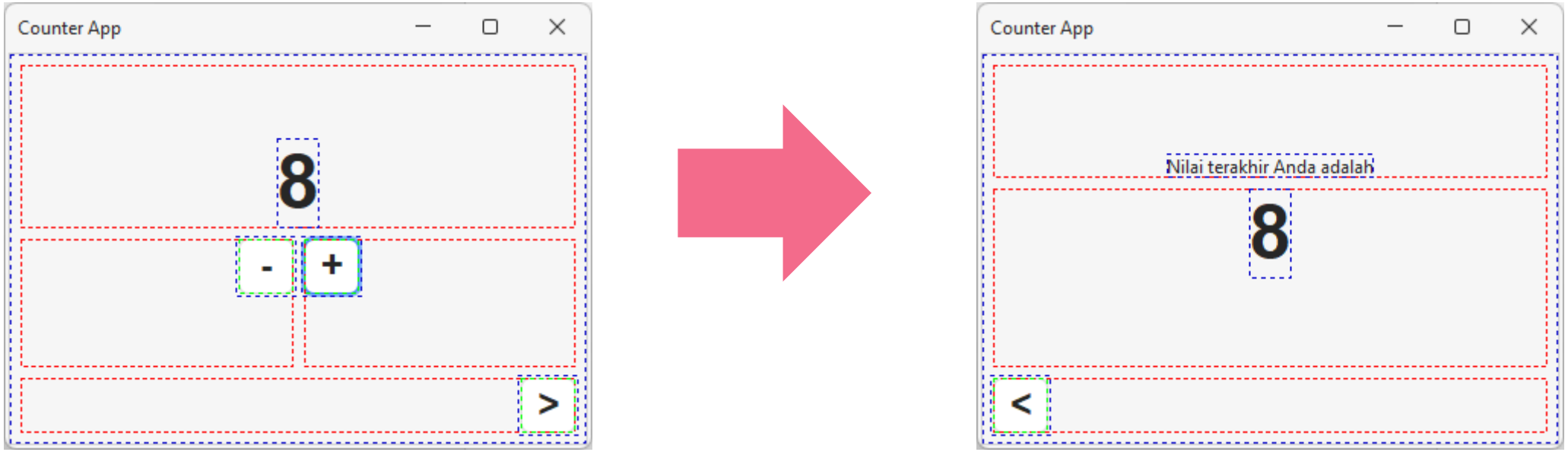
```
            JButton buttonAdd = new JButton("+");
            buttonAdd.setFont(new Font("Arial", Font.BOLD, 25));
            panelCounter.add(buttonAdd, "top");

            buttonAdd.addActionListener(e -> {
                nilaiCounter++;
                labelCounter.setText(nilaiCounter+"");
            });

            JScrollPane scrollPane = new JScrollPane(panelCounter);
            scrollPane.getVerticalScrollBar().setUnitIncrement(10);
            scrollPane.getVerticalScrollBar().putClientProperty("JScrollBar.fastWheelScrolling", true);
            scrollPane.setOpaque(true);

            frame.add(scrollPane);
            frame.pack();
            frame.setLocationRelativeTo(null);
            frame.setVisible(true);
        });
    }

}
```

Tugas



** Perubahan tampilan JPanel secara dinamis dilakukan dengan memanfaatan objek **CardLayout***

Tugas



** Layout yang disajikan dapat mengacu pada gambar tersebut*

Selesai

Terima kasih 🙏