

LAPORAN RESMI
MODUL II
EVENT LISTENER
PEMROGRAMAN VISUAL



NAMA	: ANISYAFAAH
N.R.P	: 22041100105
DOSEN	: Ir. ACH. DAFID, S.T., M.T.
ASISTEN	: NURI HIDAYATULOH
TGL PRAKTIKUM	: 04 OKTOBER 2023

Disetujui : 09 Oktober 2023
Asisten

NURI HIDAYATULOH
21.04.411.00100



LABORATORIUM BISNIS INTELIJEN SISTEM
PRODI SISTEM INFORMASI
JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS TRUNOJOYO MADURA

BAB I

PENDAHULUAN

1.1 Latar Belakang

Event listener dalam bahasa pemrograman Java merupakan salah satu konsep yang penting dalam pengembangan aplikasi berbasis peristiwa (event-driven). Konsep ini memungkinkan program Java untuk merespons peristiwa-peristiwa tertentu, seperti tindakan pengguna pada antarmuka pengguna, perubahan status suatu objek, atau bahkan peristiwa sistem. Dengan menggunakan event listener, pengembang dapat membuat program yang lebih interaktif dan responsif, karena program dapat merespons perubahan-perubahan yang terjadi dalam waktu nyata.

Salah satu kegunaan utama dari event listener adalah dalam pengembangan antarmuka pengguna (GUI). Ketika pengguna berinteraksi dengan elemen-elemen seperti tombol atau kotak teks dalam aplikasi Java, event listener dapat digunakan untuk mendeteksi tindakan pengguna, seperti mengklik tombol atau memasukkan teks. Ini memungkinkan program untuk merespons dengan benar terhadap tindakan tersebut, seperti mengeksekusi kode tertentu ketika tombol ditekan.

Selain itu, event listener juga dapat digunakan dalam konteks lain selain GUI. Misalnya, dalam pemrograman server, event listener dapat digunakan untuk mendeteksi peristiwa seperti permintaan masuk dari klien atau perubahan status sistem. Ini memungkinkan pengembang untuk mengatur tindakan yang harus diambil berdasarkan peristiwa tersebut. Dengan demikian, event listener adalah komponen penting dalam pengembangan aplikasi Java yang memungkinkan program beroperasi secara adaptif dan merespons dengan tepat terhadap berbagai peristiwa yang terjadi dalam program.

1.2 Tujuan

- Mampu memahami konsep pemrograman Swing
- Mampu membuat halaman sederhana menggunakan komponen Swing dibantu tool GUI Builder
- Mampu memahami konsep penggunaan komponen Swing

BAB II

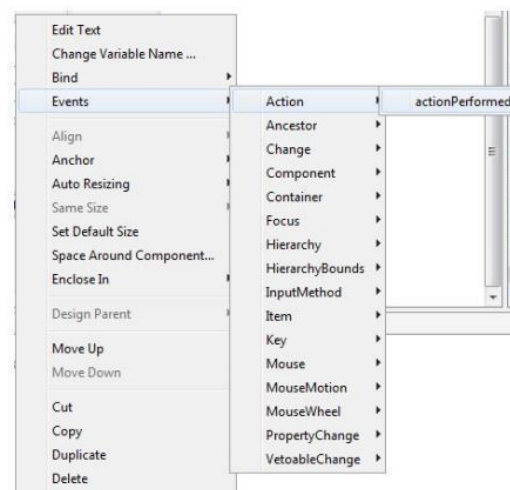
DASAR TEORI

2.1 Event Handling

Event Handling merupakan sebuah method yang digenerate oleh pengguna pada saat sesuatu terjadi terhadap suatu komponen. Sebagai contoh, event digenerate pada saat pengguna mengklik sebuah tombol, drag posisi mouse, atau memilih item dari combo box. Event handling dalam menangani event terbagi menjadi 2 macam yaitu Event Source, Event Listener atau Event Handler. Berikut langkah-langkah untuk membangun Java Desktop Application.

2.2 Event Listener

Event listener adalah objek yang diberitahu pada saat suatu event terjadi pada event source. Event listener diimplementasikan ke dalam bentuk interface. Dengan demikian, untuk mengimplementasikan salah satu listener tertentu, perlu mendefinisikan method yang terdapat pada interface bersangkutan. Sebagai contoh, interface ActionListener mendeklarasikan sebuah method abstrak dengan nama `actionPerformed()`. Jadi, apabila ingin membuat kelas yang mengimplementasikan interface ActionListener, maka harus mendefinisikan method `actionPerformed()` didalam kelas yang dibuat.



Atau bisa juga dengan meng-klik 2x komponen yang ingin ditambahkan event. Untuk tombol (JButton), event yang terbentuk adalah `action` → `actionPerformed()`.

Akan muncul bagian source code pada gui seperti berikut. Logika pemrograman dapat dituliskan di method tersebut.

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    //tuliskan logika pemrograman di blok ini
}
```

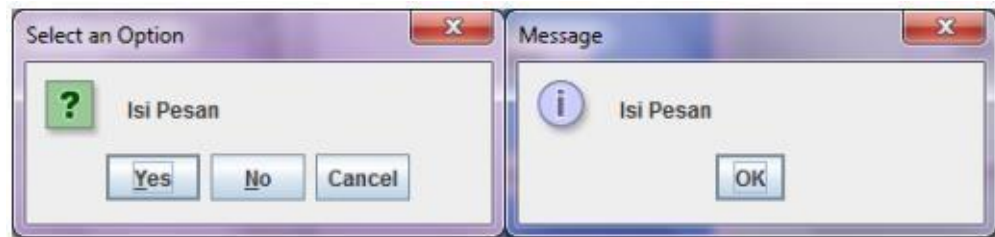
Beberapa komponen swing memiliki method yang bisa digunakan untuk dipakai pada logika pemrograman yang terdapat pada event handler. Beberapa yang paling sering digunakan adalah sebagai berikut:

Method	Komponen	Fungsi
setText("X")	JButton, JTextfield, JLabel	Mengatur tulisan yang muncul pada masing-masing komponen
getText()	JButton, JTextfield, JLabel	Mengambil nilai String yang terdapat pada masing masing komponen
setToolTipText("X")	JButton, JTextfield, JLabel	Memberikan tooltip pada komponen
setEnabled(false)	JButton, JTextfield, JLabel, JComboBox, JRadioButton, JCheckBox	Meng-enable suatu komponen atau tidak (dapat di-klik atau tidak)
setSelected(true)	JRadioButton, JCheckBox	Membuat masing-masing komponen terpilih atau tidak
isSelected()	JRadioButton, JCheckBox	Menge-cek apakah suatu komponen sedang terpilih atau tidak
setSelectedIndex(4)	JComboBox	Membuat indeks pada angka tertentu sebagai komponen terpilih
getSelectedIndex()	JComboBox	Mengambil indeks terpilih dari komponen. Indeks dimulai dari 0
getSelectedItem()	JComboBox	Mengambil objek terpilih dari komponen. Dapat langsung ditampilkan jika yang terpilih adalah objek String.
insertItemAt(objek, indek)	JComboBox	Memasukkan item pilihan berupa sebuah objek bertipe Object pada index ke index
setValueAt(Objek, baris, kolom)	JTable	Menge-set table dengan nilai Objek yang bertipe data Object pada baris dan kolom tertentu
getSelectedRow()	JTable	Mengambil indeks baris tabel terpilih. Indeks dimulai dari 0
getSelectedColumn()	JTable	Mengambil indeks kolom tabel terpilih. Indeks dimulai dari 0

2.3 Penggunaan Dialog (JOptionPane)

Dialog di modul ini merujuk pada sebuah komponen yang muncul sebagai informasi atau peringatan bahkan meminta input user setelah melakukan sesuatu.

Bentuk:



Pada komponen swing, komponen ini bernama JOptionPane. Terdapat 4 buah option pane:

- a) `showConfirmDialog` : Untuk memberi konfirmasi yes/no/cancel
- b) `showInputDialog`: Untuk menampilkan pop-up sebagai input data
- c) `showMessageDialog`: Untuk menampilkan status/pesan dengan 1 tombol "OK".
- d) `showOptionDialog`: Gabungan dari ketiga komponen di atas. Ada kasi konfirmasi, meminta input dan menampilkan status/pesan

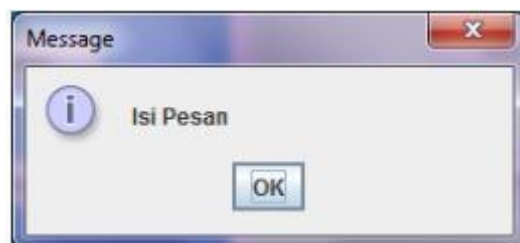
Pada modul praktikum ini hanya dibahas 2 tipe option pane, `showConfirmDialog` dan `showMessageDialog`. Setiap option pane dapat dibuat objeknya dengan menggunakan jumlah parameter yang berbeda-beda. Hal ini akan berpengaruh pada tampilan option pane yang dihasilkan.

2.3.1 `showMessageDialog`

Contoh Kode `showMessageDialog` dengan 2 parameter:

```
JOptionPane.showMessageDialog(null, "Isi Pesan");
```

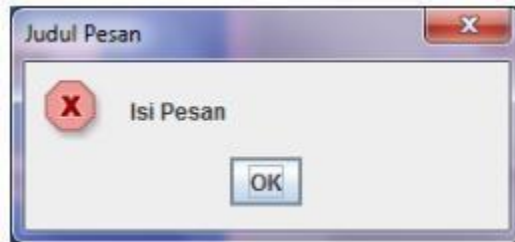
Hasil:







Contoh Kode showMessageDialog dengan 4 parameter:

```
JOptionPane.showMessageDialog(nu  
ll, "Isi Pesan", "Judul Pesan",  
JOptionPane.ERROR_MESSAGE);
```

Hasil:



Lambang/symbol/icon yang dihasilkan (silang merah atau huruf “i” dalam lingkaran), tergantung pada parameter terakhir dari contoh kode di atas (untuk kode 4 parameter). Parameter ini disebut “option type”. Jenis-jenis dari option type adalah sebagai berikut:

Icon	Code	IDE Value
No icon	JOptionPane.PLAIN_MESSAGE	-1
	JOptionPane.ERROR_MESSAGE	0
	JOptionPane.INFORMATION_MESSAGE	1
	JOptionPane.WARNING_MESSAGE	2
	JOptionPane.QUESTION_MESSAGE	3

2.3.2 showConfirmDialog

Contoh Kode showMessageDialog dengan 2 parameter:

```
JOptionPane.showConfirmDialog(null, "Isi Pesan");
```

Hasil:



Untuk mengambil nilai dari setiap tombol yang ditekan, gunakan kode sebagaiberikut:

```
//meminta input yang ditekan sekaligus deklarasi option pane int  
hasil = JOptionPane.showConfirmDialog(null, "Isi Pesan");  
//membandingkan hasil dengan konstanta yang dimiliki option  
pane if(hasil == JOptionPane.YES_OPTION){  
System.out.println("Yes!");  
}else if(hasil == JOptionPane.NO_OPTION){  
System.out.println("No!");  
}else if(hasil == JOptionPane.CANCEL_OPTION){  
System.out.println("Cancel!"); }
```

Perbandingan pesan terdiri dari:

- YES_OPTION: Jika memilih tombol “YES”
- OK_OPTION: Jika memilih tombol “OK”
- NO_OPTION: Jika memilih tombol “NO”
- CLOSED_OPTION: Jika memilih tombol “CLOSED”
- CANCEL_OPTION: Jika memilih tombol “CANCEL”

[Signature]
NURAH

BAB III TUGAS PENDAHULUAN

3.1 Soal

1. Jelaskan mengenai konsep pemrograman Swing dan sebutkan apa saja komponen swing dalam pemrograman GUI!
2. Apa yang dimaksud tool gui builder?
3. Jelaskan pengertian dan fungsi implementasi event listener dalam pemrograman GUI!

3.2 Jawab

1. Pemrograman Swing adalah konsep dalam pengembangan aplikasi berbasis Java untuk menciptakan antarmuka grafis pengguna (GUI). Swing sangat populer dalam pengembangan aplikasi desktop Java karena memiliki komponen-komponen GUI yang kaya dan fleksibel serta mendukung pengembangan aplikasi lintas platform. Beberapa komponen swing dalam pemrograman GUI antara lain JFrame, JPanel, JButton, JLabel, JTextField, JTextArea, dan masih banyak komponen swing lainnya.
2. GUI Builder adalah alat atau perangkat lunak yang dirancang untuk memudahkan pembuatan antarmuka pengguna grafis (GUI) dalam pengembangan perangkat lunak. Tujuannya adalah untuk mempermudah proses desain dan pengaturan elemen-elemen GUI tanpa perlu menulis kode secara manual. GUI Builder biasanya digunakan dalam pengembangan aplikasi desktop atau perangkat lunak berbasis GUI.
3. Dalam pemrograman GUI (Graphical User Interface), event listener adalah mekanisme yang digunakan untuk mendeteksi dan merespons peristiwa (event) yang terjadi pada komponen-komponen GUI. Event listener memungkinkan aplikasi untuk merespons interaksi pengguna dengan GUI. Ketika event terjadi, event listener akan menangkap event tersebut dan menjalankan kode yang sesuai. Fungsinya yaitu mendeteksi peristiwa, merespons peristiwa, mengatur alur program, meningkatkan pengalaman pengguna (User Experience), mengustomisasi antarmuka, melakukan validasi dan kontrol aplikasi.

BAB IV

IMPLEMENTASI

4.1 Source Code

1. Program Perhitungan Gaji Karyawan

```
package modul2;

public class Nomor1 extends javax.swing.JFrame {
    public int GajiBruto;

    public Nomor1() {
        initComponents();
    }
    @SuppressWarnings("unchecked")
    //Generated Code

    private void HitungBersihActionPerformed(java.awt.event.ActionEvent evt) {
        String Jam = TextJam.getText();
        String Harga = TextTarif.getText();

        int JamKerja = Integer.parseInt(Jam);
        int Tarif = Integer.parseInt(Harga);

        if(JamKerja <= 40){
            GajiBruto = JamKerja * Tarif;
        }else {
            int Lembur = JamKerja - 40;
            GajiBruto = (int) ((40 * Tarif) + (Lembur * Tarif * 1.5));
        }

        double pajakPenghasilan = 0.10 * GajiBruto;
        double GajiBersih = GajiBruto - pajakPenghasilan;

        String formattedGajiBersih = "Rp " + String.format("%,d", (int) GajiBersih);
        TextBersih.setText(formattedGajiBersih);

    }

    private void HitungBrutoActionPerformed(java.awt.event.ActionEvent evt) {
        String jamkerja = TextJam.getText();
        String tarif = TextTarif.getText();

        int JamKerja = Integer.parseInt(jamkerja);
        int Tarif = Integer.parseInt(tarif);

        if(JamKerja<=40){
            GajiBruto = JamKerja * Tarif;
        }else {
            int JamLembur = JamKerja - 40;
            GajiBruto = (int) ((40 * Tarif) + (JamLembur * Tarif * 1.5));
        }
        String formattedGajiBruto = "Rp " + String.format("%,d", GajiBruto);
        TextBruto.setText(formattedGajiBruto);
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
```

```

        new Nomor1().setVisible(true);
    }
});
}

```

2. Program Penjualan Produk

```

package modul2;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JOptionPane;

public class Nomor2 extends javax.swing.JFrame {
    String menu;
    int harga, Pesanan, Total;

    public Nomor2() {
        initComponents();
        TextJumlah.setText("" + Pesanan);
        Tambah.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                Pesanan++;
                TextJumlah.setText(Integer.toString(Pesanan));
            }
        });

        Kurang.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                if (Pesanan > 0) {
                    Pesanan--;
                    TextJumlah.setText(Integer.toString(Pesanan));
                }
            }
        });
    }

    @SuppressWarnings("unchecked")
    //Generated Code
    private void BeliActionPerformed(java.awt.event.ActionEvent evt) {
        if (RadioKopi.isSelected()) {
            menu = "KOPI";
            harga = 10000;
        } else if (RadioMilkshake.isSelected()) {
            menu = "MILKSHAKE";
            harga = 13000;
        } else if (RadioJus.isSelected()) {
            menu = "JUS BUAH";
            harga = 12000;
        } else if (RadioTeh.isSelected()) {
            menu = "TEH";
            harga = 5000;
        } else if (RadioEs.isSelected()) {
            menu = "ES KRIM";
            harga = 7000;
        } else if (RadioMilkTea.isSelected()) {
            menu = "MILKTEA";
            harga = 12000;
        } else if (RadioKentang.isSelected()) {
            menu = "KENTANG GORENG";
            harga = 10000;
        } else if (RadioPasta.isSelected()) {

```

```

        menu = "PASTA";
        harga = 15000;
    }else if (RadioWaffle.isSelected()){
        menu = "WAFFLE";
        harga = 12000;
    }else if (RadioAir.isSelected()){
        menu = "AIR MINERAL";
        harga = 5000;

        }else if (!RadioKopi.isSelected() && !RadioMilkshake.isSelected() &&
!RadioJus.isSelected() && !RadioTeh.isSelected() &&
!RadioEs.isSelected() && !RadioMilkTea.isSelected()&&
!RadioKentang.isSelected() && !RadioPasta.isSelected() &&
!RadioWaffle.isSelected() && !RadioAir.isSelected()) {
        JOptionPane.showMessageDialog(this, "Silahkan Pilih Menu!");
        return;
    }

    Pesanan = Integer.parseInt(TextJumlah.getText());
    Total = Pesanan * harga ;
    TextPesanan.setText(menu);
    String PesananStr = Integer.toString(Pesanan);
    TextJml.setText(PesananStr);
    String TotalStr = Integer.toString(Total);
    TextTotal.setText(TotalStr);

}

private void HapusActionPerformed(java.awt.event.ActionEvent evt) {

    buttonGroup1.clearSelection();
    TextJumlah.setText("");
    TextPesanan.setText("");
    TextJml.setText("");
    TextTotal.setText("");

}

public static void main(String args[]) {

    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new Nomor2().setVisible(true);
        }
    });
}

```

4.2 Hasil

1. Program Perhitungan Gaji Karyawan

The application window is titled "PERHITUNGAN GAJI KERJA KARYAWAN" and features a blue header with a user icon. It is divided into three main columns:

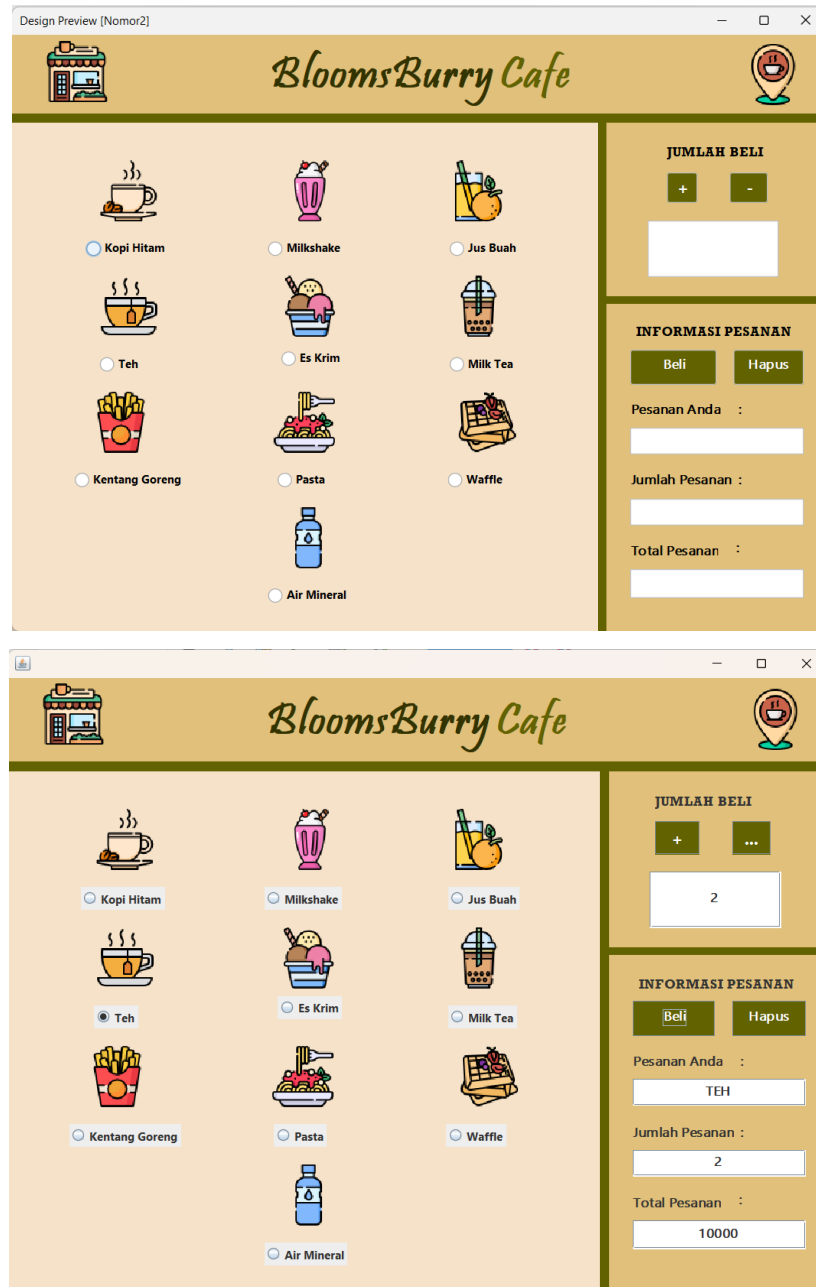
- Left Column:** Contains a text field for "Jumlah Jam Kerja 1 Minggu" (initially empty), an icon of a bar chart and money, and a text field for "Tarif Per Jam" (initially "10000").
- Middle Column:** Contains a text field for "Gaji Bruto" (initially empty), a "Hitung" button, and an icon of a person with a clock.
- Right Column:** Contains a text field for "Gaji Bersih" (initially empty), a "Hitung" button, and an icon of a document with a dollar sign.

The bottom screenshot shows the results after clicking the "Hitung" buttons:

- Left Column:** "Jumlah Jam Kerja 1 Minggu" is now "39".
- Middle Column:** "Gaji Bruto" is now "Rp 390,000".
- Right Column:** "Gaji Bersih" is now "Rp 351,000".

Program ini menampilkan program untuk menghitung gaji karyawan. Gaji ini dihitung sesuai dengan ketentuan yang terdapat pada soal. Pada design/ramennya terdapat beberapa komponen swing, seperti label, button, dan textfield. Komponen ini tentu saja memiliki perannya masing-masing. Pada source code juga sudah terdapat rumus-rumus untuk menghitung gaji bruto dan gaji bersihnya. Sehingga setelah program dirun, user harus menginputkan jam kerja. Setelah itu klik kedua button maka gaji akan otomatis muncul pada masing-masing textfield sesuai dengan aturan perhitungan gaji yang tertera pada soal

2. Program Penjualan Produk



Program ini menampilkan penjualan makanan dan minuman pada café. Pada design/ramennya terdapat beberapa komponen swing, seperti label, button, dan textfield. Komponen ini tentu saja memiliki perannya masing-masing. Menu hanya bisa dipilih salah satu saja dan user harus menambah jumlah pesanan sehingga setelah user mengklik button beli, makan program akan menampilkan nama pesanan, jumlah pesanan, dan total pesanannya. Program ini tentu saja menggunakan perhitungan rumus seperti perkalian untuk menghitung total pesanan pilihan user.

BAB V

PENUTUP

5.1 Analisa

Dari hasil praktikum, praktikan menganalisa bahwa salah satu keuntungan utama dari penggunaan event listener adalah pemisahan antara logika bisnis dan antarmuka pengguna. Dengan cara ini, pengembang dapat dengan mudah mengelola tindakan yang terjadi di antarmuka pengguna tanpa harus mengubah logika bisnis inti program. Ini membuat kode lebih mudah dipelihara dan diperluas. Selain itu, event listener memungkinkan program Java untuk mendukung banyak peristiwa sekaligus, memungkinkan aplikasi yang lebih kompleks dengan berbagai jenis interaksi pengguna.

Dalam pembuatan program Java dengan event listener, pengembang perlu mendefinisikan event listener yang sesuai dan mengaitkannya dengan komponen antarmuka pengguna yang relevan. Selanjutnya, mereka harus mengimplementasikan metode yang akan dipanggil ketika event tertentu terjadi. Ini memerlukan pemahaman yang kuat tentang konsep event-driven programming dan kemampuan untuk merancang respons yang sesuai terhadap event yang berbeda. Keseluruhan, penggunaan event listener adalah bagian integral dari pengembangan aplikasi Java yang interaktif dan responsif.

5.2 Kesimpulan

1. Event listener adalah objek yang diberitahu pada saat suatu event terjadi pada event source. Event listener diimplementasikan ke dalam bentuk interface.
2. Pada komponen swing, komponen ini bernama JOptionPane. Terdapat 4 buah option pane antara lain showConfirmDialog, showInputDialog, showMessageDialog, showOptionDialog.
3. Pada program yang telah diimplementasikan di atas menggunakan komponen Event Listener dan beberapa komponen JOptionPane.