Pertemuan XVI

GUI (*GRAPHICAL USER INTERFACE*)

16.1. *Abstract Windowing Toolkit* (AWT) vs. *Swing*

The Java Foundation Class (JFC), merupakan bagian penting dari Java SDK, yang termasuk dalam koleksi dari API dimana dapat mempermudah pengembangan aplikasi JAVA GUI. JFC termasuk diantara 5 bagian utama dari API yaitu AWT dan *Swing*. Tiga bagian yang lainnya dari API adalah *Java2D*, *Accessibility*, dan *Drag and Drop*. Semua itu membantu pengembang dalam mendesain dan mengimplementasikan aplikasi visual yang lebih baik.

AWT dan *Swing* menyediakan komponen GUI yang dapat digunakan dalam membuat aplikasi Java dan *Applet*. Tidak seperti beberapa komponen AWT yang menggunakan *native code*, keseluruhan *Swing* ditulis menggunakan bahasa pemrograman Java. *Swing* menyediakan implementasi *platform-independent* dimana aplikasi yang dikembangkan dengan platform yang berbeda dapat memiliki tampilan yang sama. Begitu juga dengan AWT menjamin tampilan *look and feel* pada aplikasi yang dijalankan pada dua mesin yang berbeda menjadi terlihat sama. *Swing* API dibangun dari beberapa API yang mengimplementasikan beberapa jenis bagian dari AWT. Kesimpulannya, komponen AWT dapat digunakan bersama komponen *Swing*.

16.2. Komponen GUI

16.2.1. JFrame

Frame merupakan komponen dasar dalam membuat aplikasi GUI, dimana frame berfungsi sebagai container atau wadah untuk menampung komponen GUI lainnya.

Untuk mengatur ukuran JFrame, dapat menggunakan method setSize.

```
void setSize(int width, int height)
```

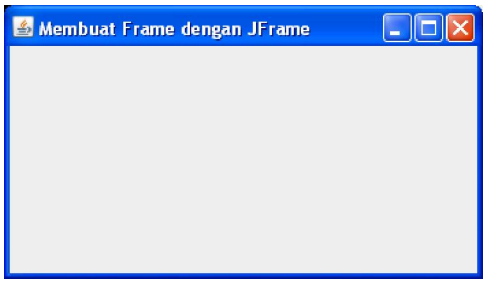
Default dari JFrame adalah *not visible* atau tak tampak hingga kita harus mengatur *visibility* menjadi *true*. Perintahnya adalah :

```
void setVisible(boolean b)
```

Dalam mendesain aplikasi GUI, Object Frame selalu digunakan. Dibawah ini adalah contoh bagaimana membuat sebuah aplikasi.

```
import javax.swing.*;
public class TampilFrame extends JFrame{
  public TampilFrame(){
    super("Membuat Frame dengan JFrame");
     setSize(350,200);
     setDefaultCloseOperation(EXIT_ON_CLOSE);
     setVisible(true);
  }

  public static void main(String args[]){
    TampilFrame frameku = new TampilFrame();
  }
}
```

Tampilan program diatas adalah :



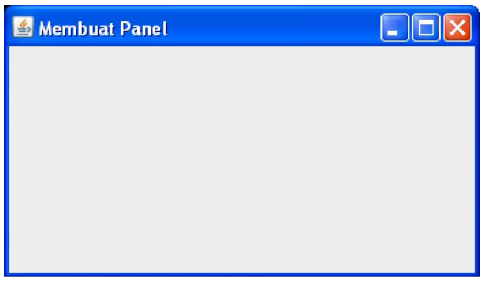
Gambar 16.1. Tampilan JFame

16.2.2. JPanel

JPanel digunakan untuk membuat sebuah panel yang berfungsi sebagai container untuk menampung berbagai macam komponen, seperti label, button, textfield, table dan lain-lain. Kode program dibawah ini adalah contoh untuk membuat panel, tetapi obyek panelnya tidak terlihat, karena hanya sebagai penampung dan mengatur tata letak komponen-komponen GUI.

```
import javax.swing.*;

public class TampilPanel extends JFrame{
  private JPanel panelku = new JPanel();

  public TampilPanel(){
    super("Membuat Panel");
     panelku.setLayout(null);
     panelku.setSize(150,200);
     setSize(350,200);
     setDefaultCloseOperation(EXIT_ON_CLOSE);
     getContentPane().add(panelku);
     setVisible(true);
  }

  public static void main(String args[]){
    TampilPanel frameku = new TampilPanel();
  }
}
```

Tampilan program diatas adalah :



Gambar 16.2. Tampilan JPanel

16.2.3. JLabel

JLabel digunakan untuk menampilkan teks yang berfungsi untuk memberikan keterangan atau menjelaskan komponen GUI lainnya agar mudah dimengerti oleh user.

Contoh program untuk menampilkan JLabel adalah sebagai berikut :

```
import javax.swing.*;

public class TampilLabel extends JFrame{
  private JPanel panelku = new JPanel();
  private static JLabel labelku = new JLabel("Label dengan Java");

  TampilLabel(){
    super("Membuat Label");
     panelku.setLayout(null);
     setSize(350,200);
     setDefaultCloseOperation(EXIT_ON_CLOSE);
     labelku.setBounds(60,20,160,25);
     panelku.add(labelku);
     getContentPane().add(panelku);
     setVisible(true);
  }

  public static void main(String args[]){
    TampilLabel frameku = new TampilLabel();
  }
}
```

Tampilan program diatas adalah :

Gambar 16.3. Tampilan JLabel

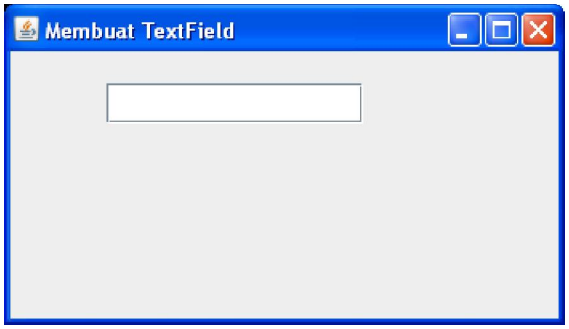
16.2.4. JTextField

JTextField adalah komponen GUI yang biasa digunakan untuk memasukkan data dengan mengetik dari keyboard. Kode program dibawah ini adalah contoh untuk membuat JTextField :

```
import javax.swing.*;
public class TampilTextField extends JFrame{
  private JPanel panelku = new JPanel();
  private static JTextField TxtNama = new JTextField();

  TampilTextField(){
    super("Membuat TextField");
```

```
      panelku.setLayout(null);
      setSize(350,200);
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      TxtNama.setBounds(60,20,160,25);
      panelku.add(TxtNama);
      getContentPane().add(panelku);
      setVisible(true);
   }

   public static void main(String args[]){
     TampilTextField frameku = new TampilTextField();
   }
}
```

Tampilan program diatas adalah :



Gambar 16.4. Tampilan JTextField

16.2.5. JButton

JButton digunakan untuk membuat sebuah tombol yang berfungsi untuk menerima input dari user berupa klik menggunakan mouse atau tombol enter dari keyboard.

Contoh program untuk menampilkan JButton adalah sebagai berikut :

```
import javax.swing.*;
public class TampilTombol extends JFrame{
  private JPanel panelku = new JPanel();
  private static JButton Tombol = new JButton("Tombol");

  TampilTombol(){
    super("Membuat Tombol");
     panelku.setLayout(null);
     setSize(350,200);
     setDefaultCloseOperation(EXIT_ON_CLOSE);
     Tombol.setBounds(130,90,80,25);
     panelku.add(Tombol);
     getContentPane().add(panelku);
     setVisible(true);
  }

  public static void main(String args[]){
    TampilTombol frameku = new TampilTombol();
     //Tombol.setText("sffasf");
  }
}
```

Tampilan program diatas adalah :

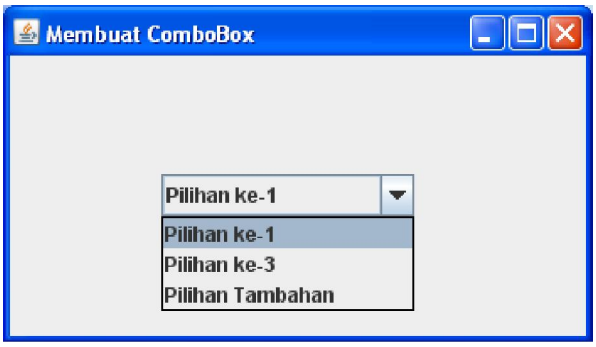
Gambar 16.5. Tampilan JButton

16.2.6. JComboBox

JComboBox merupakan sebuah kelas pada swing yang berguna untuk membuat sebuah ComboBox. ComboBox biasanya digunakan untuk menampilkan daftar item sebagai pilihan untuk user. Berikut ini adalah kode program untuk menampilkan JComboBox :

```
 import javax.swing.*;
 public class TampilComboBox extends JFrame{
   final static String Pilihan[] = {"Pilihan ke-1","Pilihan ke-2",
"Pilihan ke-3"};
   private JPanel panelku = new JPanel();
   private static JComboBox ComboBox = new JComboBox(Pilihan);

   TampilComboBox(){
     super("Membuat ComboBox");
      panelku.setLayout(null);
      setSize(350,200);
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      ComboBox.setBounds(90,70,150,25);
      panelku.add(ComboBox);
      getContentPane().add(panelku);
      setVisible(true);
   }

   public static void main(String args[]){
     TampilComboBox frameku = new TampilComboBox();
      ComboBox.addItem("Pilihan Tambahan");
      ComboBox.removeItem("Pilihan ke-2");
   }
 }
```

Tampilan program diatas adalah :



Gambar 16.6. Tampilan JComboBox

16.2.7. JMenu

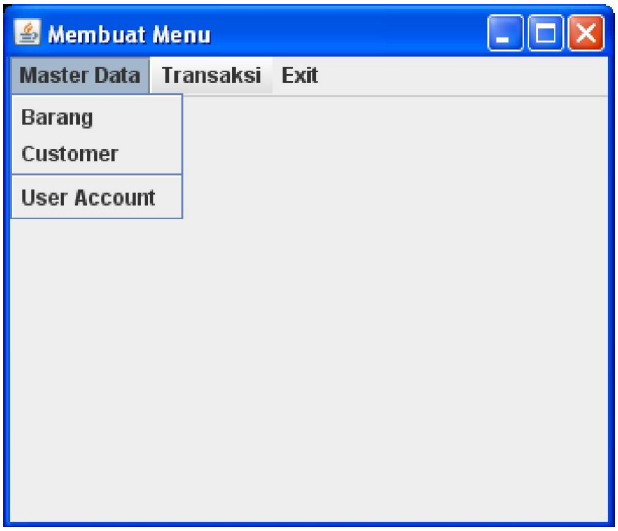
JMenu digunakan untuk membuat menu pull-down yang dapat digunakan untuk memanggil suatu form. Suatu menu biasa terdiri dari beberapa bagian, seperti menu induk, submenu dan menu item. Untuk membuat menu secara lengkap, diperlukan beberapa komponen pendukung lainnya, seperti JMenuBar, JMenu, dan JMenuItem.

Berikut ini adalah contoh untuk menampilkan JMenu :

```
 import javax.swing.*;
 import java.awt.*;

 public class TampilMenuBar extends JFrame{
   private JMenuBar MenuBar = new JMenuBar();
   private JMenu MenuMaster = new JMenu("Master Data");

   private JMenuItem MenuBarang = new JMenuItem("Barang"),
           MenuCustomer = new JMenuItem("Customer"),
           MenuUserAccount = new JMenuItem("User Account");

   private JMenu MenuTransaksi = new JMenu("Transaksi");
   private JMenuItem MenuPenjualan = new JMenuItem("Penjualan"),
           MenuPembelian = new JMenuItem("Pembelian");

   private JMenuItem MenuExit = new JMenuItem("Exit");

   Dimension dimensi = Toolkit.getDefaultToolkit().getScreenSize();

   TampilMenuBar(){
     super("Membuat Menu");
      setSize(350,300);
      setLocation(dimensi.width/2-getWidth()/2,dimensi.height/2-
getHeight()/2);
      setDefaultCloseOperation(EXIT_ON_CLOSE);

      MenuMaster.add(MenuBarang);
      MenuMaster.add(MenuCustomer);
      MenuMaster.addSeparator();
      MenuMaster.add(MenuUserAccount);
      MenuBar.add(MenuMaster);

      MenuTransaksi.add(MenuPenjualan);
      MenuTransaksi.add(MenuPembelian);
```

```
      MenuBar.add(MenuTransaksi);
      MenuBar.add(MenuExit);

     setJMenuBar(MenuBar);
      setVisible(true);
   }

   public static void main(String args[]){
     TampilMenuBar frameku = new TampilMenuBar();
   }
 }
```

Tampilan program diatas adalah :



Gambar 16.7. Tampilan JMenu

16.2.8. JTable

JTable merupakan kelas yang digunakan untuk membuat tabel. Selain menggunakan kelas JTable, dalam membuat sebuah table juga diperlukan kelas lain yang terdapat dalam paket Swing, seperti kelas DefaultTableModel dan kelas JScrollPane.

Berikut ini adalah contoh program untuk menampilkan table :

```
 import javax.swing.*;
 import javax.swing.table.DefaultTableModel;
 import javax.swing.table.TableColumn;
 import javax.swing.table.*;

 public class TampilTabel extends JFrame{
   static String JudulKolom[] = {"No.","NIM","Nama","Angkatan","Kelas"};
   static DefaultTableModel ModelTabel = new
DefaultTableModel(null,JudulKolom);
   static JTable Tabel = new JTable();
   JScrollPane ScrollBar = new JScrollPane();
   private JPanel panelku = new JPanel();

   TampilTabel(){
     super("Menampilkan Tabel");
     setSize(400,240);
```

```
     setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
     panelku.setLayout(null);
     Tabel.setModel(ModelTabel);
     ScrollBar.getViewport().add(Tabel);
     Tabel.setEnabled(true);

     // Disable auto resizing
     Tabel.setAutoResizeMode(JTable.AUTO_RESIZE_OFF);

     // set Column width
     TableColumn col = Tabel.getColumnModel().getColumn(0);
     col.setPreferredWidth(30);

     // Column Alignment
     DefaultTableCellRenderer rightRenderer = new
DefaultTableCellRenderer();
     rightRenderer.setHorizontalAlignment( JLabel.RIGHT );
     Tabel.getColumnModel().getColumn(0).setCellRenderer( rightRenderer
);

     ScrollBar.setBounds(20,20,350,160);
     panelku.add(ScrollBar);
     getContentPane().add(panelku);
     //getContentPane().add(ScrollBar);
     setVisible(true);
   }

   public static void main(String args[]){
     TampilTabel frameku = new TampilTabel();
     int i;
     for (i=0;i<=15;i++){
       ModelTabel.insertRow(i,new Object[]{i+1,"NIM ke-"+i,"Nama ke-
"+i,"Angkatan ke-"+i,"Kelas ke-"+i});
     }
     Tabel.setValueAt("Nama 1",0,1);
   }
 }
```

Tampilan program diatas adalah :

Gambar 16.8. Tampilan JTable

16.3. GUI Event Handling

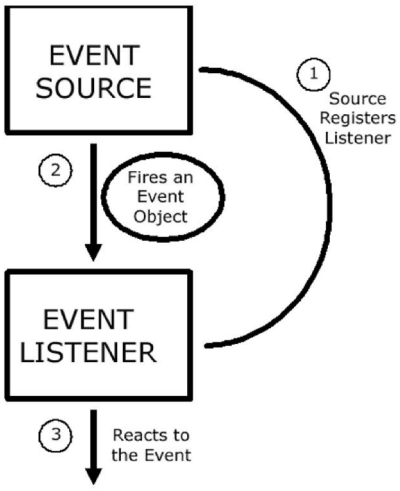
Agar komponen GUI yang kita buat dapat menghasilkan suatu *event*, ketika *user* melakukan interaksi terhadap komponen GUI, seperti melakukan klik terhadap menu atau *button*, menekan tombol *keyboard*, menggerakkan mouse dan lain-lain, diperlukan *listener* untuk mendeteksinya. Dengan adanya *listener*, *event* yang dihasilkan dari komponen GUI dapat mengarahkan ke suatu instruksi program. Untuk dapat menggunakan *listener*, diperlukan paket yang terdapat dalam *java.awt.event.**.

16.3.1. Delegation Event Model

Delegasi event model menguraikan bagaimana program yang kita buat dapat merespon interaksi dari user. Untuk memahami model, pertama-tama mari kita pelajari melalui tiga komponen utamanya.

a. Event Source
   Event source mengacu pada komponen GUI yang meng-generate event. Sebagai contoh, jika user menekan tombol, event source dalam hal ini adalah tombol.
b. Event Listener/Handler
   Event listener menerima berita dari event-event dan proses-proses interaksi user. Ketika tombol ditekan, listener akan mengendalikan dengan menampilkan sebuah informasi yang berguna untuk user.
c. Event Object
   Ketika sebuah event terjadi (misal, ketika user berinteraksi dengan komponen GUI), sebuah obyek event diciptakan. Obyek berisi semua informasi yang perlu tentang event yang telah terjadi. Informasi meliputi tipe dari event yang telah terjadi, seperti ketika mouse telah di-klik. Ada beberapa class event untuk kategori yang berbeda dari user action. Sebuah event object mempunyai tipe data mengenai salah satu dari class ini.

Di bawah ini adalah delegation event model.



Gambar 16.9. Delegation Event Model

Pada awalnya, sebuah listener seharusnya diregistrasikan dengan sebuah source sehingga dapat menerima informasi tentang event-event yang terjadi pada source tersebut. Hanya listener yang sudah teregistrasi yang dapat menerima pemberitahuan event-event. Ketika telah teregistrasi, sebuah listener hanya tinggal menunggu sampai event terjadi.

Ketika sesuatu terjadi dengan event source, sebuah event object akan menguraikan event yang diciptakan. Event kemudian ditembak oleh source pada listener yang teregistrasi.

Saat listener menerima sebuah event object (pemberitahuan) dari source, dia akan bekerja. Menerjemahkan pemberitahuan dan memproses event yang terjadi.

16.3.2. Registrasi Listeners

Event source mendaftarkan sebuah listener melalui method add<Type>Listener.

```
void add<Type>Listener(<Type>Listener listenerObj)
```

<Type> tergantung pada tipe dari event source. Dapat berupa Key, Mouse, Focus, Component, Action dan lainnya.

Beberapa listeners dapat diregistrasi dengan satu event source untuk menerima pemberitahuan event. Listener yang telah teregistrasi dapat juga tidak diregistrasikan lagi menggunakan method remove<Type>Listener.

```
void remove<Type>Listener(<Type>Listener listenerObj)
```

16.3.3. Class-Class Event

Sebuah event object mempunyai sebuah class event sebagai tipe data acuannya. Akar dari hirarki class event adalah class EventObject, yang dapat ditemukan pada paket java.util. Immediate subclass dari class EventObject adalah class AWTEvent. Class AWTEvent didefinisikan pada paket java.awt. Itu merupakan akar dari semua AWTbased events. Berikut ini beberapa dari class-class AWT event.

Tabel 16.1. Class-Class Event

Class Event	Deskripsi
ComponentEvent	Extends AWTEvent. Dijalankan ketika sebuah komponen dipindahkan, di-resize, dibuat visible atau hidden.
InputEvent	Extends *ComponentEvent*. Abstrak root class event untuk semua komponen-level input class-class event.
ActionEvent	Extends *AWTEvent*. Dijalankan ketika sebuah tombol ditekan, melakukan double-klik daftar item, atau memilih sebuah menu.
ItemEvent	Extends *AWTEvent*. Dijalankan ketika sebuah item dipilih atau di*deselect* oleh user, seperti sebuah list atau checkbox.
KeyEvent	Extends *InputEvent*. Dijalankan ketika sebuah *key* ditekan, dilepas atau diketikkan.
MouseEvent	Extends *InputEvent*. Dijalankan ketika sebuah tombol mouse ditekan, dilepas, atau di-klik (tekan dan lepas), atau ketika sebuah kursor mouse masuk atau keluar dari bagian visible dari komponen.

TextEvent	Extends *AWTEvent*. Dijalankan ketika nilai dari text field atau text area dirubah.
WindowEvent	Extends *ComponentEvent*. Dijalankan sebuah object *Window* dibuka, ditutup, diaktifkan, nonaktifkan, *iconified*, *deiconified*, atau ketika *focus* ditransfer kedalam atau keluar window.

Catatan, bahwa semua subclass-subclass AWTEvent mengikuti konvensi nama berikut ini:

```
<Type>Event
```

16.3.4. Event Listeners

Event listeners adalah class yang mengimplementasikan interfaces <Type>Listener. Tabel di bawah menunjukkan beberapa listener interfaces yang biasanya digunakan.

Tabel 16.2. Event Listeners

Event Listeners	Deskripsi
ActionListener	Bereaksi atas perubahan mouse atau atau keyboard.
MouseListener	Bereaksi atas pergerakan mouse.
MouseMotionListener	Interface *MouseMotionListener* mendukung *MouseListener*. Menyediakan method-method yang akan memantau pergerakan mouse,seperti drag dan pemindahan mouse.
WindowListener	Bereaksi atas perubahan window.

16.3.4.1. Method ActionListener

Interface ActionListener hanya terdiri dari satu method, yaitu :

```
public void actionPerformed(ActionEvent e)
```

Mengendalikan ActionEvent e yang terjadi.

16.3.4.2. Method MouseListener

Di bawah ini adalah method-method MouseListener yang seharusnya digunakan dalam penerapan class.

a. `public void mouseClicked(MouseEvent e)`
   Dipanggil pada saat tombol mouse di click (seperti tekan dan lepas).
b. `public void mouseEntered(MouseEvent e)`
   Dipanggil pada saat kursor mouse memasuki area komponen.
c. `public void mouseExited(MouseEvent e)`
   Dipanggil pada saat kursor mouse meninggalkan area komponen.
d. `public void mousePressed(MouseEvent e)`
   Dipanggil pada saat tombol mouse ditekan di atas komponen
e. `public void mouseReleased(MouseEvent e)`
   Dipanggil pada saat tombol mouse dilepas di atas komponen

16.3.4.3. Method-Method MouseMotionListener

MouseMotionListener mempunyai dua method untuk diimplementasikan.

a. `public void mouseDragged(MouseEvent e)`
   Digunakan untuk memantau pergerakan mouse yang melintasi object pada saat tombol mouse ditekan. Tindakan ini persis sama dengan tindakan pada saat memindahkan sebuah window.
b. `public void mouseMoved(MouseEvent e)`
   Digunakan untuk memantau pergerakan mouse pada saat mouse melintasi area suatu object. Pada saat ini tidak ada mouse yang ditekan, hanya memindahkan pointer mouse melalui objek.

16.3.4.4. Method-Method WindowListener

Di bawah ini method-method dari interface WindowListener.

a. `public void windowOpened(WindowEvent e)`
   Dipanggil pada saat object window dibuka (pertama kali window dibuat tampil).
b. `public void windowClosing(WindowEvent e)`
   Dipanggil pada saat user mencoba untuk menutup object Window dari menu system object.
c. `public void windowClosed(WindowEvent e)`
   Dipanggil pada saat object Window ditutup setelah memanggil penempatan (misal, release dari resource-resource yang digunakan oleh source) pada object.
d. `public void windowActivated(WindowEvent e)`
   Dilibatkan ketika object Window adalah window yang aktif (window masih dipakai).
e. `public void windowDeactivated(WindowEvent e)`
   Dilibatkan ketika object Window tidak lagi merupakan window yang aktif.
f. `public void windowIconified(WindowEvent e)`
   Dipanggil ketika object Window di-minimize.
g. `public void windowDeiconified(WindowEvent e)`
   Dipanggil ketika object Window kembali setelah di-minimize ke keadaan normal.

16.3.5. Petunjuk untuk Menciptakan Aplikasi Handling GUI Events

Berikut ini langkah-langkah yang dibutuhkan untuk mengingat ketika ingin membuat aplikasi GUI dengan event handling.

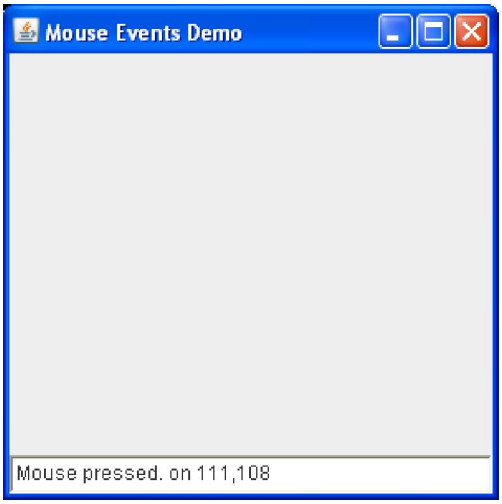
a. Buatlah sebuah class yang menguraikan dan membuat suatu tampilan dari aplikasi GUI.
b. Buatlah sebuah class yang menerapkan interface listener yang sesuai. Class ini boleh mengacu pada class yang sama seperti pada langkah awal.
c. Dalam menerapkan class, gunakan semua method-method dengan interface listener yang sesuai. Uraikan masing-masing method bagaimana kita ingin mengendalikan event-event. Kita dapat memberikan implementasi kosong untuk method yang tidak ingin kita gunakan.
d. Daftarkan object listener, instansiatiate dari class listener pada langkah b, dengan source component menggunakan method add<Type>Listener.

16.3.6. Contoh Mouse Events

```
 import java.awt.*;
 import java.awt.event.*;
 import javax.swing.*;

 public class MouseEventsDemo extends JFrame implements MouseListener,
MouseMotionListener {
   TextField tf;

   public MouseEventsDemo(String title){
     super(title);
      setDefaultCloseOperation(EXIT_ON_CLOSE);

     tf = new TextField(60);
     addMouseListener(this);
   }

   public void launchFrame() {
     /* Menambah komponen pada frame */
     add(tf, BorderLayout.SOUTH);
      setSize(300,300);
     setVisible(true);
   }

   public void mouseClicked(MouseEvent me) {
     String msg = "Mouse clicked.";
     tf.setText(msg);
   }

   public void mouseEntered(MouseEvent me) {
     String msg = "Mouse entered component.";
     tf.setText(msg);
   }

   public void mouseExited(MouseEvent me) {
     String msg = "Mouse exited component.";
     tf.setText(msg);
   }

   public void mousePressed(MouseEvent me) {
     String msg = "Mouse pressed. on " + me.getX() + "," + me.getY();
     tf.setText(msg);
   }

   public void mouseReleased(MouseEvent me) {
     String msg = "Mouse released.";
     tf.setText(msg);
   }

   public void mouseDragged(MouseEvent me) {
     String msg = "Mouse dragged at " + me.getX() + "," + me.getY();
     tf.setText(msg);
   }

   public void mouseMoved(MouseEvent me) {
     String msg = "Mouse moved at " + me.getX() + "," + me.getY();
     tf.setText(msg);
   }

   public static void main(String args[]) {
```

```
    MouseEventsDemo med = new MouseEventsDemo("Mouse Events Demo");
    med.launchFrame();
  }
}
```

Tampilan program diatas adalah :



Gambar 16.10. Tampilan Mouse Event

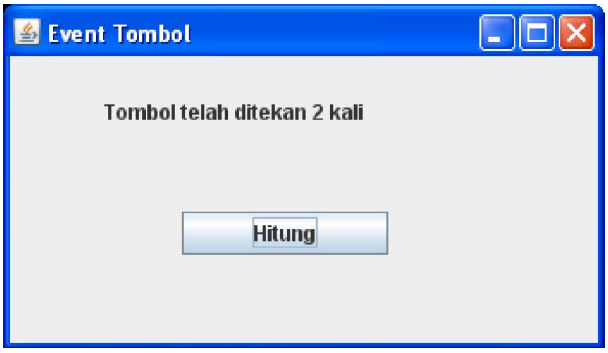
16.3.7. Contoh Event Listener

Berikut ini adalah contoh penggunaan event listener :

```
import javax.swing.*;
import java.awt.event.*;
import java.awt.*;

public class evenTombol extends JFrame{
  private JButton tombolku = new JButton("Hitung");
  private JLabel labelku = new JLabel();
  private JPanel panelku = new JPanel();
  Dimension dimensi = Toolkit.getDefaultToolkit().getScreenSize();
  private int Jml=1;

  evenTombol(){
    super("Event Tombol");
    panelku.setLayout(null);
    setSize(350,200);
    setLocation(dimensi.width/2-getWidth()/2,dimensi.height/2-
getHeight()/2);
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    labelku.setBounds(55,20,200,25);
    labelku.setVisible(false);
    tombolku.setBounds(100,90,120,25);
    tombolku.addActionListener(new ActionListener(){
          public void actionPerformed(ActionEvent env){
              labelku.setVisible(true);
              labelku.setText("Tombol telah ditekan "+Jml+" kali");
              Jml++;
          }
    });
```

```
    panelku.add(labelku);
    panelku.add(tombolku);
    getContentPane().add(panelku);
    setVisible(true);
  }

  public static void main(String args[]){
    evenTombol evenku = new evenTombol();
  }
}
```

Tampilan program diatas adalah :



Gambar 16.11. Tampilan Event Listener

16.4. Contoh Aplikasi Kalkulator Sederhana

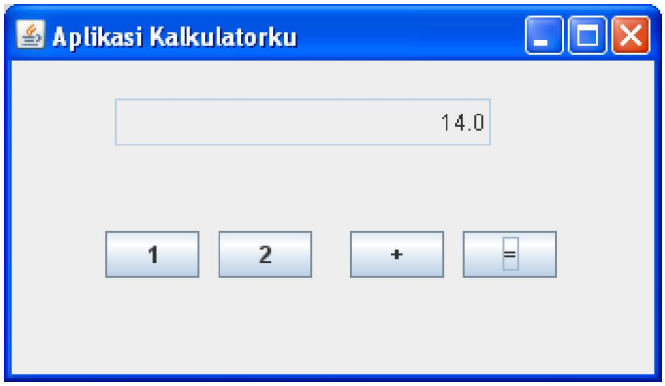
Berikut ini adalah aplikasi kalkulator sederhana yang terdiri dari tombol angka 1, angka 2, tambah dan sama dengan :

```
import javax.swing.*;
import java.awt.event.*;
import java.awt.*;

public class Kalkulatorku extends JFrame{
  private static JTextField tampilanTextField = new JTextField();

  private static String operant1 = "";
  private static String operant2 = "";
  private static String operatorDipilih = "";
  private static boolean setelahOperator = false;

  private JButton tombol1 = new JButton("1");
  private JButton tombol2 = new JButton("2");
  private JButton tombolTambah = new JButton("+");
  private JButton tombolSamaDengan = new JButton("=");
  private JPanel panelku = new JPanel();

  Dimension dimensi = Toolkit.getDefaultToolkit().getScreenSize();
  private int Jml=1;

  Kalkulatorku(){
    super("Aplikasi Kalkulatorku");
    panelku.setLayout(null);
    setSize(350,200);
```

```
     setLocation(dimensi.width/2-getWidth()/2,dimensi.height/2-
getHeight()/2);
     setDefaultCloseOperation(EXIT_ON_CLOSE);

     /* Menambah TextField pada koordinat (X,Y,Lebar,Tinggi) */

tampilanTextField.setHorizontalAlignment(javax.swing.JTextField.RIGHT);
     tampilanTextField.setEditable(false);
     tampilanTextField.setBounds(55,20,200,25);

     /* Menambah Button pada koordinat (X,Y,Lebar,Tinggi) */
     tombol1.setBounds(50,90,50,25);
     tombol2.setBounds(110,90,50,25);
     tombolTambah.setBounds(180,90,50,25);
     tombolSamaDengan.setBounds(240,90,50,25);

     /* Menambahkan Action Listener pada button */
     tombol1.addActionListener(new TombolHandler());
     tombol2.addActionListener(new TombolHandler());
     tombolTambah.addActionListener(new TombolHandler());
     tombolSamaDengan.addActionListener(new TombolHandler());

     /* Menambahakan TextField pada panel */
     panelku.add(tampilanTextField);

     /* Menambahakan button pada panel */
     panelku.add(tombol1);
     panelku.add(tombol2);
     panelku.add(tombolTambah);
     panelku.add(tombolSamaDengan);

     /* Menambahakan panel pada frame */
     getContentPane().add(panelku);
     setVisible(true);
   }

   private class TombolHandler implements ActionListener {
     public void actionPerformed(ActionEvent e){
       JButton tombol = (JButton)e.getSource();

       if (tombol.getText().equals("1")) {
         if (setelahOperator) {
           tampilanTextField.setText("1");
         } else {
           tampilanTextField.setText(tampilanTextField.getText()+"1");
         }
         setelahOperator = false;
       } else if (tombol.getText().equals("2")) {
         if (setelahOperator) {
           tampilanTextField.setText("2");
         } else {
           tampilanTextField.setText(tampilanTextField.getText()+"2");
         }
         setelahOperator = false;
       } else if (tombol.getText().equals("+")) {
         operant1 = tampilanTextField.getText();
         operatorDipilih = "+";
         setelahOperator = true;
       } else if (tombol.getText().equals("=")) {
         operant2 = tampilanTextField.getText();
```

```
          double operantPertama = 0;
          double operantKedua = 0;

          try {
            operantPertama = Double.parseDouble(operant1);
          } catch (Exception ex) {
          }

          try {
            operantKedua = Double.parseDouble(operant2);
          } catch (Exception ex) {
          }

          if (operatorDipilih == "+") {

tampilanTextField.setText(Double.toString(operantPertama+operantKedua));
          }
        }
      }
    }

    public static void main(String args[]){
      Kalkulatorku kalkulator = new Kalkulatorku();
    }
  }
```

Tampilan aplikasi kalkulator :



Gambar 16.12. Tampilan aplikasi kalkulator

16.5. Desain Aplikasi MDI

Sekarang kita membuat desain aplikasi took menggunakan MDI (*Multiple Document Interface*).

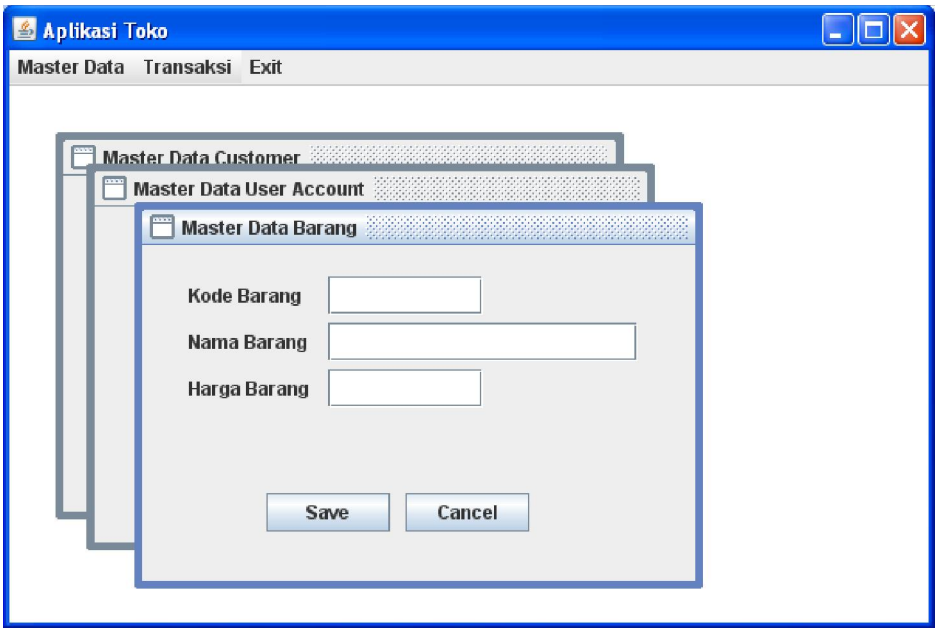
```
 import javax.swing.*;
 import java.awt.*;
 import java.awt.event.*;

 public class AplikasiToko extends JFrame{
   private JDesktopPane frmMDI;
   private JInternalFrame frmBarang;
   private JInternalFrame frmCustomer;
   private JInternalFrame frmUserAccount;
   private JPanel pnlBarang;
```

```
   private JMenuBar MenuBar = new JMenuBar();
   private JMenu MenuMaster = new JMenu("Master Data");

   private JMenuItem MenuBarang = new JMenuItem("Barang"),
           MenuCustomer = new JMenuItem("Customer"),
           MenuUserAccount = new JMenuItem("User Account");

   private JMenu MenuTransaksi = new JMenu("Transaksi");
   private JMenuItem MenuPenjualan = new JMenuItem("Penjualan"),
           MenuPembelian = new JMenuItem("Pembelian");

   private JMenuItem MenuExit = new JMenuItem("Exit");

   private static JLabel LblKodeBarang = new JLabel("Kode Barang");
   private static JTextField TxtKodeBarang = new JTextField();
   private static JLabel LblNamaBarang = new JLabel("Nama Barang");
   private static JTextField TxtNamaBarang = new JTextField();
   private static JLabel LblHargaBarang = new JLabel("Harga Barang");
   private static JTextField TxtHargaBarang = new JTextField();

   private static JButton TblBarangSave = new JButton("Save");
   private static JButton TblBarangCancel = new JButton("Cancel");

   Dimension dimensi = Toolkit.getDefaultToolkit().getScreenSize();

   AplikasiToko(){
     super("Aplikasi Toko");
     setSize(650,500);
     setLocation(dimensi.width/2-getWidth()/2,dimensi.height/2-
getHeight()/2);
     setDefaultCloseOperation(EXIT_ON_CLOSE);
     frmMDI = new JDesktopPane ();
     frmMDI.setLayout(null);
     this.add(frmMDI);

     MenuMaster.add(MenuBarang);
     MenuMaster.add(MenuCustomer);
     MenuMaster.addSeparator();
     MenuMaster.add(MenuUserAccount);
     MenuBar.add(MenuMaster);

     MenuTransaksi.add(MenuPenjualan);
     MenuTransaksi.add(MenuPembelian);
     MenuBar.add(MenuTransaksi);
     MenuBar.add(MenuExit);

     /* Menambahkan action listener */
     MenuBarang.addActionListener(new MenuHandler());
     MenuCustomer.addActionListener(new MenuHandler());
     MenuUserAccount.addActionListener(new MenuHandler());
     MenuExit.addActionListener(new MenuHandler());

     TblBarangSave.addActionListener(new TombolHandler());
     TblBarangCancel.addActionListener(new TombolHandler());

     setContentPane(frmMDI);

     frmBarang = new JInternalFrame();
     frmBarang.setTitle("Master Data Barang");
     frmCustomer = new JInternalFrame("Master Data Customer");
```

```
    frmUserAccount = new JInternalFrame("Master Data User Account");

    pnlBarang = new JPanel ();
    LblKodeBarang.setBounds(30,20,160,25);
    pnlBarang.add(LblKodeBarang);
    TxtKodeBarang.setBounds(120,20,100,25);
    pnlBarang.add(TxtKodeBarang);
    LblNamaBarang.setBounds(30,50,160,25);
    pnlBarang.add(LblNamaBarang);
    TxtNamaBarang.setBounds(120,50,200,25);
    pnlBarang.add(TxtNamaBarang);
    LblHargaBarang.setBounds(30,80,160,25);
    pnlBarang.add(LblHargaBarang);
    TxtHargaBarang.setBounds(120,80,100,25);
    pnlBarang.add(TxtHargaBarang);
    TblBarangSave.setBounds(80,160,80,25);
    pnlBarang.add(TblBarangSave);
    TblBarangCancel.setBounds(170,160,80,25);
    pnlBarang.add(TblBarangCancel);
    pnlBarang.setLayout(null);
    frmBarang.add(pnlBarang);

    frmMDI.add(frmBarang);
    frmMDI.add(frmCustomer);
    frmMDI.add(frmUserAccount);

    frmBarang.setBounds(10,10,367,250);
    frmCustomer.setBounds(30,30,367,250);
    frmUserAccount.setBounds(50,50,367,250);

    setJMenuBar(MenuBar);
    setVisible(true);
  }

  private class MenuHandler implements ActionListener {
    public void actionPerformed(ActionEvent e){
      JMenuItem M = (JMenuItem)e.getSource();

      if (M.getText().equals("Barang")) {
        frmBarang.setVisible(true);
      } else if (M.getText().equals("Customer")) {
        frmCustomer.setVisible(true);
      } else if (M.getText().equals("User Account")) {
        frmUserAccount.setVisible(true);
      } else if (M.getText().equals("Exit")) {
        dispose();
      }
    }
  }

  private class TombolHandler implements ActionListener {
    public void actionPerformed(ActionEvent e){
      JButton TblPilih = (JButton)e.getSource();

      if (TblPilih.getText().equals("Save")) {
        JOptionPane.showMessageDialog(null,"Dipilih Save");
        frmBarang.setVisible(false);
      } else if (TblPilih.getText().equals("Cancel")) {
        JOptionPane.showMessageDialog(null,"Dipilih Cancel");
        frmBarang.setVisible(false);
      }
```

```
    }
  }

  public static void main(String args[]){
    AplikasiToko frameku = new AplikasiToko();
  }
}
```

Tampilan program diatas adalah :



Gambar 16.13. Tampilan Mouse Event

Referensi:

1. Hariyanto, Bambang, (2007), *Esensi-esensi Bahasa Pemrograman Java*, Edisi 2, Informatika Bandung, November 2007.
2. Utomo, Eko Priyo, (2009), *Panduan Mudah Mengenal Bahasa Java*, Yrama Widya, Juni 2009.
3. Tim Pengembang JENI, JENI 1-6, Depdiknas, 2007
4. Supriyatno, *Pemrograman Database Menggunakan Java & MySQL Untuk Pemula*, Cetakan Pertama, Mediakita, Jakarta, 2010
5. http://download.oracle.com/javase/tutorial/uiswing/components/index.html
6. http://www.compucranks.com/subpage11.html