

BAB 7

Abstract Windowing Toolkit dan Swing

7.1 Tujuan

Tanpa mempelajari tentang *graphical user interface* (GUI) API, Anda masih tetap bisa membuat suatu program. Tetapi, program Anda akan kelihatan tidak menarik dan tidak nyaman digunakan bagi para user. Memiliki GUI yang baik dapat memberi efek pada penggunaan aplikasi. Java menyediakan banyak *tool* seperti *Abstract Windowing Toolkit* dan *Swing* untuk mengembangkan aplikasi GUI yang interaktif.

Pada akhir pembahasan, diharapkan pembaca dapat :

1. Memahami persamaan dan perbedaan antara AWT dan Swing
2. Perbedaan antara komponen dan kontainer.
3. Mendesain aplikasi GUI menggunakan AWT.
4. Mendesain aplikasi GUI menggunakan Swing.
5. Menjelaskan tentang *flow layout*, *border layout*, dan *grid layout* dalam komponen GUI
6. Membuat tampilan yang kompleks dalam mendesain aplikasi GUI.

7.2 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 dan 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. Anda akan mempelajari applet pada bab berikutnya. 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.

7.3 Komponen GUI pada AWT

7.3.1 Window Classes Fundamental

Dalam mengembangkan aplikasi GUI, komponen GUI seperti tombol atau textfield diletakkan di dalam kontainer. Berikut ini adalah daftar dari beberapa class penting pada kontainer yang telah disediakan oleh AWT.

<i>Class AWT</i>	<i>Deskripsi</i>
Komponen	Abstract Class untuk object yang dapat ditampilkan pada console dan berinteraksi dengan user. Bagian utama dari semua class AWT.
Kontainer	Abstract Subclass dari Component Class. Sebuah komponen yang dapat menampung komponen yang lainnya.
Panel	Turunan dari Container Class. Sebuah frame atau window tanpa titlebar, menubar tidak termasuk border. Superclass dari applet class.
Window	Turunan dari Container class. Top level window, dimana berarti tidak bisa dimasukkan dalam object yang lainnya. Tidak memiliki border dan menubar.
Frame	Turunan dari window class. Window dengan judul, menubar, border dan pengatur ukuran di pojok. Memiliki empat constructor , dua diantaranya memiliki penulisan seperti dibawah ini : Frame() Frame(String title)

Tabel 1.2.1: Class kontainer AWT

Untuk mengatur ukuran window, menggunakan method setSize.

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

mengubah ukuran komponen ini dengan *width* dan *height* sebagai parameter.

```
void setSize(Dimension d)
```

mengubah ukuran dengan *d.width* dan *d.height* berdasar pada spesifikasi *Dimension d*.

Default dari window adalah *not visible* atau tak tampak hingga Anda mengatur *visibility* menjadi *true*. Inilah *syntax* untuk method *setVisible*.

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

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

```
import java.awt.*;

public class SampleFrame extends Frame {
    public static void main(String args[]) {
        SampleFrame sf = new SampleFrame();
        sf.setSize(100, 100); //Coba hilangkan baris ini
        sf.setVisible(true); //Coba hilangkan baris ini
    }
}
```

```

    }
}

```

perhatikan bahwa tombol tutup pada frame tidak akan bekerja karena tidak ada mekanisme *event handling* yang ditambahkan di dalam aplikasi. Anda akan belajar tentang *event handling* pada modul selanjutnya.

7.3.2 Grafik

Beberapa method grafik ditemukan dalam class *Graphic*. Dibawah ini adalah daftar dari beberapa method.

drawLine()	drawPolyline()	setColor()
fillRect()	drawPolygon()	getFont()
drawRect()	fillPolygon()	setFont()
clearRect()	getColor()	drawString()

Tabel 1.2.2a: Beberapa metode dari kelas *Graphics*

Hubungan dari class ini adalah class *Color*, dimana memiliki tiga constructor.

Format Constructor	Deskripsi
Color(int r, int g, int b)	Nilai integer 0 - 255.
Color(float r, float g, float b)	Nilai float 0.0 - 1.0.
Color(int rgbValue)	Panjang nilai : 0 ke $2^{24}-1$ (hitam ke putih). Red: bits 16-23 Green: bits 8-15 Blue: bits 0-7

Dibawah ini adalah contoh program yang menggunakan beberapa method di dalam class *Graphic*.

```

import java.awt.*;

public class GraphicPanel extends Panel {
    GraphicPanel() {
        setBackground(Color.black); //Konstanta dalam class Color
    }
    public void paint(Graphics g) {
        g.setColor(new Color(0,255,0)); //hijau
        g.setFont(new Font("Helvetica",Font.PLAIN,16));
        g.drawString("Hello GUI World!", 30, 100);
        g.setColor(new Color(1.0f,0,0)); //red
        g.fillRect(30, 100, 150, 10);
    }
    public static void main(String args[]) {
        Frame f = new Frame("Testing Graphics Panel");
    }
}

```

```

        GraphicPanel gp = new GraphicPanel();
        f.add(gp);
        f.setSize(600, 300);
        f.setVisible(true);
    }
}

```

Agar panel dapat terlihat atau *visible*, dia harus diletakkan didalam window yang dapat terlihat seperti sebuah frame.

7.3.3 Beberapa komponen AWT

Berikut ini adalah daftar dari kontrol AWT. Kontrol adalah komponen seperti tombol atau textfield yang mengijinkan user untuk berinteraksi dengan aplikasi GUI. Berikut ini semua subclass dari class *Components*.

Label	Button	Choice
TextField	Checkbox	List
TextArea	CheckboxGroup	Scrollbar

Tabel 1.2.3: Komponen AWT

Berikut adalah aplikasi membuat sebuah frame dengan kontrol yang telah dimasukkan di dalamnya.

```

import java.awt.*;

class FrameWControls extends Frame {
    public static void main(String args[]) {
        FrameWControls fwc = new FrameWControls();
        fwc.setLayout(new FlowLayout()); //akan dibahas lebih detail
pada pembahasan berikutnya
        fwc.setSize(600, 600);
        fwc.add(new Button("Test Me!"));
        fwc.add(new Label("Labe"));
        fwc.add(new TextField());
        CheckboxGroup cbg = new CheckboxGroup();
        fwc.add(new Checkbox("chk1", cbg, true));
        fwc.add(new Checkbox("chk2", cbg, false));
        fwc.add(new Checkbox("chk3", cbg, false));
        List list = new List(3, false);
        list.add("MTV");
        list.add("V");
        fwc.add(list);
        Choice chooser = new Choice();
        chooser.add("Avril");
        chooser.add("Monica");
        chooser.add("Britney");
        fwc.add(chooser);
        fwc.add(new Scrollbar());
    }
}

```

```

        fwc.setVisible(true);
    }
}

```

7.4 Layout Manager

Posisi dan ukuran suatu komponen ditentukan oleh layout manager. Layout manager mengatur tampilan dari komponen di dalam kontainer. Berikut ini beberapa layout manager yang terdapat di dalam Java.

- 1.FlowLayout
- 2.BorderLayout
- 3.GridLayout
- 4.GridBagLayout
- 5.CardLayout

Layout manager dapat diatur menggunakan method *setLayout* dari class *Container*. Method ini dapat ditulis sebagai berikut.

```
void setLayout(LayoutManager mgr)
```

Jika Anda memilih untuk tidak menggunakan layout manager, Anda dapat mengisi null sebagai argumen untuk method ini. Tetapi selanjutnya, Anda akan mengatur posisi elemen secara manual dengan menggunakan method *setBounds* dari class *Components*.

```
public void setBounds(int x, int y, int width, int height)
```

Method ini mengatur posisi berdasarkan pada argumen x dan y, dan ukuran berdasarkan argumen *width* dan *height*. Hal ini akan cukup menyulitkan dan membosankan untuk aplikasi jika Anda memiliki beberapa objek komponen didalam object container. Anda akan memanggil method ini untuk setiap komponen.

7.4.1 FlowLayout Manager

FlowLayout Manager adalah default manager untuk class *Panel* dan subclassnya, termasuk class applet. Cara meletakkan komponen dari FlowLayout Manager dimulai dari kiri ke kanan dan dari atas ke bawah, dimulai dari pojok kiri atas. Seperti pada saat Anda mengetik menggunakan editor kata pada umumnya. Berikut adalah bagaimana FlowLayout Manager bekerja, dimana memiliki tiga constructor seperti daftar di bawah ini.

FlowLayout Constructors
FlowLayout()
Membuat object baru FlowLayout dengan posisi di tengah dan lima unit horizontal dan vertikal gap dimasukkan pada komponen sebagai default.
FlowLayout(int align)
Membuat object baru FlowLayout dengan posisi spesifik dan lima unit horizontal dan vertikal gap dimasukkan pada komponen sebagai default.

FlowLayout Constructors

```
FlowLayout(int align, int hgap, int vgap)
```

Membuat object baru FlowLayout dengan argumen pertama sebagai posisi pada komponen dan *hgap* untuk horizontal dan *vgap* untuk vertikal pada komponen

Tabel 1.3.1: Constructor FlowLayout

Gap dapat dikatakan sebagai jarak antara komponen dan biasanya diukur dengan satuan pixel. Posisi argumen mengikuti penulisan sebagai berikut :

- 1.FlowLayout.LEFT
- 2.FlowLayout.CENTER
- 3.FlowLayout.RIGHT

Bagaimanakah output dari program berikut?

```
import java.awt.*;

class FlowLayoutDemo extends Frame {
    public static void main(String args[]) {
        FlowLayoutDemo fld = new FlowLayoutDemo();
        fld.setLayout(new FlowLayout(FlowLayout.RIGHT, 10, 10));
        fld.add(new Button("ONE"));
        fld.add(new Button("TWO"));
        fld.add(new Button("THREE"));
        fld.setSize(100, 100);
        fld.setVisible(true);
    }
}
```

Gambar berikut adalah contoh dari hasil yang berjalan pada platform Window.



Tabel 13.1: Contoh hasil dalam window

7.4.2.BorderLayout Manager

BorderLayout membagi kontainer menjadi lima bagian diantaranya utara, selatan, timur,

barat, dan tengah. Setiap komponen dimasukkan ke dalam region yang spesifik. Region utara dan selatan membentuk jalur horizontal sedangkan region timur dan barat membentuk jalur vertikal. Dan region tengah berada pada perpotongan jalur horizontal dan vertikal. Tampilan ini adalah bersifat default untuk object *Window*, termasuk object dari subclass *Window* yaitu tipe *Frame* dan *Dialog*.

Constructor BorderLayout
<code>BorderLayout()</code>
Membuat object <i>BorderLayout</i> baru tanpa spasi yang diaplikasikan diantara komponen yang berbeda.
<code>BorderLayout(int hgap, int vgap)</code>
Membuat object <i>BorderLayout</i> baru dengan spasi unit <i>hgap</i> horizontal dan unit <i>vgap</i> vertikal yang diaplikasikan diantara komponen yang berbeda.

Tabel 1.3.2: Constructor *BorderLayout*

Seperti pada *FlowLayout* Manager, parameter *hgap* dan *vgap* disini juga menjelaskan jarak antara komponen dengan kontainer.

Untuk menambahkan komponen kedalam region yang spesifik, gunakan method menambahkan dan melewati dua argumen yaitu : komponen yang ingin dimasukkan ke dalam region dan region mana yang ingin dipakai untuk meletakkan komponen. Perlu diperhatikan bahwa hanya satu komponen yang dapat dimasukkan dalam satu region. Menambahkan lebih dari satu komponen pada kontainer yang bersangkutan, maka komponen yang terakhir ditambahkan yang akan ditampilkan. Berikut ini adalah daftar dari kelima region.

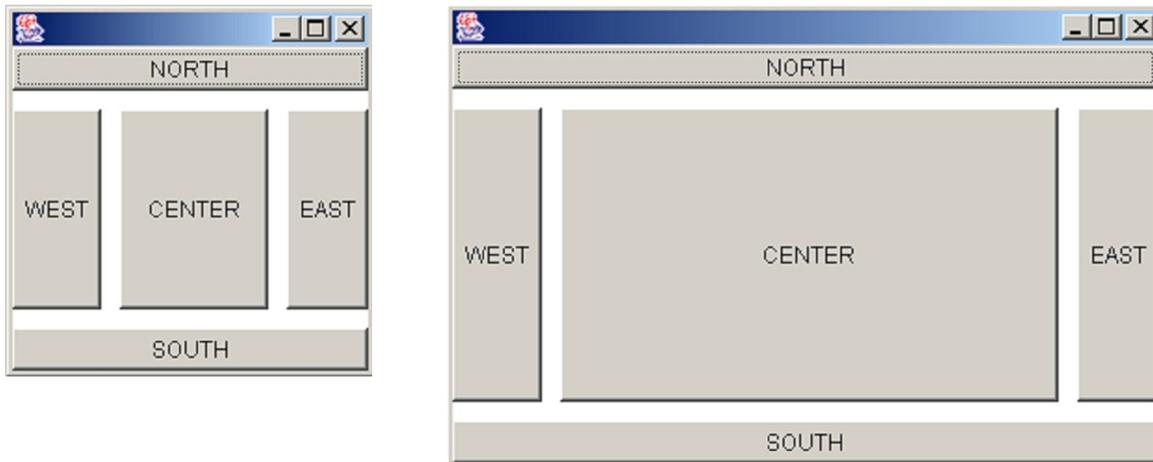
1. *BorderLayout.NORTH*
2. *BorderLayout.SOUTH*
3. *BorderLayout.EAST*
4. *BorderLayout.WEST*
5. *BorderLayout.CENTER*

Berikut ini adalah contoh program yang menunjukkan bagaimana *BorderLayout* bekerja.

```
import java.awt.*;

class BorderLayoutDemo extends Frame {
    public static void main(String args[]) {
        BorderLayoutDemo bld = new BorderLayoutDemo();
        bld.setLayout(new BorderLayout(10, 10)); //may remove
        bld.add(new Button("NORTH"), BorderLayout.NORTH);
        bld.add(new Button("SOUTH"), BorderLayout.SOUTH);
        bld.add(new Button("EAST"), BorderLayout.EAST);
        bld.add(new Button("WEST"), BorderLayout.WEST);
        bld.add(new Button("CENTER"), BorderLayout.CENTER);
        bld.setSize(200, 200);
        bld.setVisible(true);
    }
}
```

Berikut ini adalah hasil dari contoh program tersebut. Gambar kedua menunjukkan efek dari mengubah bentuk dari frame.



Gambar 1.3.2: hasil contoh program

7.4.3 GridLayout Manager

Dengan *GridLayout manager*, komponen juga diposisikan dari kiri ke kanan dan dari atas ke bawah seperti pada *FlowLayout manager*. *GridLayout manager* membagi kontainer menjadi baris dan kolom. Semua region memiliki ukuran yang sama. Hal tersebut tidak mepedulikan ukuran sebenarnya dari komponen.

Berikut ini adalah daftar dari constructor untuk class *GridLayout*.

Constructor GridLayout
<code>GridLayout()</code>
Membuat object GridLayout baru dengan satu baris dan satu kolom sebagai default
<code>GridLayout(int rows, int cols)</code>
Membuat object GridLayout baru dengan jumlah baris dan kolom sesuai dengan keinginan
<code>GridLayout(int rows, int cols, int hgap, int vgap)</code>
Membuat object GridLayout baru dengan jumlah baris dan kolom yang ditentukan. Unit spasi hgap horizontal dan vgap vertikal diaplikasikan ke dalam komponen.

Tabel 1.3.3: Constructor GridLayout

Cobalah program ini.

```
import java.awt.*;

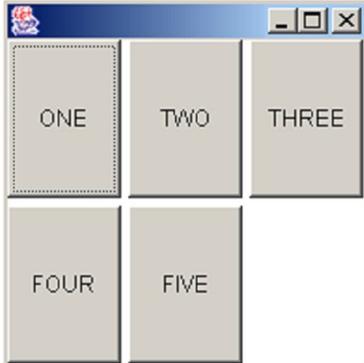
class GridLayoutDemo extends Frame {
    public static void main(String args[]) {
        GridLayoutDemo gld = new GridLayoutDemo();
        gld.setLayout(new GridLayout(2, 3, 4, 4));
        gld.add(new Button("ONE"));
        gld.add(new Button("TWO"));
    }
}
```

```

        gld.add(new Button("THREE"));
        gld.add(new Button("FOUR"));
        gld.add(new Button("FIVE"));
        gld.setSize(200, 200);
        gld.setVisible(true);
    }
}

```

Berikut ini adalah output dari program.



Gambar 1.3.3: hasil contoh program

7.4.4 Panel dan Tampilan kompleks

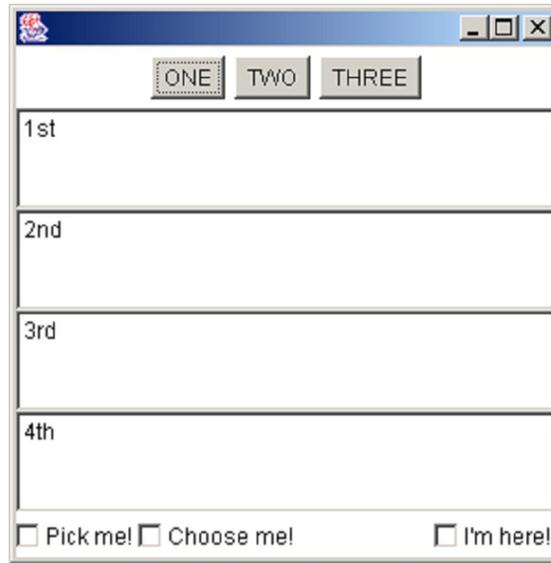
Untuk membuat tampilan yang lebih lengkap, Anda dapat menggabungkan layout manager yang berbeda dengan menggunakan panel. Ingatlah bahwa panel adalah kontainer dan komponen pada saat yang sama. Anda dapat memasukkan komponen ke dalam panel dan kemudian menambahkan panel ke dalam region yang Anda inginkan di dalam kontainer.

Perhatikan teknik yang digunakan pada contoh berikut.

```
import java.awt.*;
```

```
class ComplexLayout extends Frame {
    public static void main(String args[]) {
        ComplexLayout cl = new ComplexLayout();
        Panel panelNorth = new Panel();
        Panel panelCenter = new Panel();
        Panel panelSouth = new Panel();
        /* Panel utara */
        //Panel menggunakan FlowLayout sebagai default
        panelNorth.add(new Button("ONE"));
        panelNorth.add(new Button("TWO"));
        panelNorth.add(new Button("THREE"));
        /* Panel tengah */
        panelCenter.setLayout(new GridLayout(4,4));
        panelCenter.add(new TextField("1st"));
        panelCenter.add(new TextField("2nd"));
        panelCenter.add(new TextField("3rd"));
        panelCenter.add(new TextField("4th"));
        /* Panel selatan */
        panelSouth.setLayout(new BorderLayout());
        panelSouth.add(new Checkbox("Choose me!"),
            BorderLayout.CENTER);
        panelSouth.add(new Checkbox("I'm here!"),
            BorderLayout.EAST);
        panelSouth.add(new Checkbox("Pick me!"),
            BorderLayout.WEST);
        /* Menambahkan panel pada container Frame*/
        //Frame menggunakan BorderLayout sebagai default
        cl.add(panelNorth, BorderLayout.NORTH);
        cl.add(panelCenter, BorderLayout.CENTER);
        cl.add(panelSouth, BorderLayout.SOUTH);
        cl.setSize(300,300);
        cl.setVisible(true);
    }
}
```

Berikut ini adalah output dari program.



Gambar 1.3.4: Hasil dari contoh program

7.5 Komponen Swing

Seperti pada package AWT, package dari Swing menyediakan banyak class untuk membuat aplikasi GUI. Package tersebut dapat ditemukan di *javax.swing*. Perbedaan utama antara keduanya adalah komponen Swing ditulis menyeluruh menggunakan Java. Kesimpulannya, program GUI ditulis menggunakan banyak class dari package Swing yang mempunyai tampilan look and feel yang sama meski dijalankan pada platform yang berbeda. Lebih dari itu, Swing menyediakan komponen yang lebih menarik seperti *color chooser* dan *option pane*.

Nama dari komponen GUI milik Swing hampir sama persis dengan komponen GUI milik AWT. Perbedaan jelas terdapat pada penamaan komponen. Pada dasarnya, nama komponen Swing sama dengan nama komponen AWT tetapi dengan tambahan huruf J pada prefixnya. Sebagai contoh, satu komponen dalam AWT adalah *button class*. Sedangkan pada Swing, nama komponen tersebut menjadi *Jbutton class*. Berikut adalah daftar dari komponen Swing.

<i>Komponen Swing</i>	<i>Penjelasan</i>
JComponent	class induk untuk semua komponen Swing, tidak termasuk top-level kontainer
JButton	Tombol "push". Berhubungan dengan class <i>button</i> dalam package AWT
JCheckBox	Item yang dapat dipilih atau tidak oleh pengguna. Berhubungan dengan class <i>checkbox</i> dalam package AWT
JFileChooser	Mengijinkan pengguna untuk memilih sebuah file. Berhubungan dengan class <i>filechooser</i> dalam package AWT

Komponen Swing	Penjelasan
JTextField	Mengijinkan untuk mengedit text satu baris. Berhubungan dengan class textfield dalam package AWT.
JFrame	Turunan dan Berhubungan dengan class frame dalam package AWT tetapi keduanya sedikit tidak cocok dalam kaitannya dengan menambahkan komponen pada kontainer. Perlu mendapatkan content pane yang terbaru sebelum menambah sebuah komponen.
JPanel	Turunan Jcomponent. Class Container sederhana tetapi bukan top-level. Berhubungan dengan class panel dalam package AWT.
JApplet	Turunan dan Berhubungan dengan class Applet dalam package AWT. Juga sedikit tidak cocok dengan class applet dalam kaitannya dengan menambahkan komponen pada container
JOptionPane	Turunan Jcomponent. Disediakan untuk mempermudah menampilkan pop-up kotak dialog.
JDialog	Turunan dan Berhubungan dengan class dialog dalam package AWT. Biasanya digunakan untuk menginformasikan sesuatu kepada pengguna atau prompt pengguna untuk input.
JColorChooser	Turunan Jcomponent. Memungkinkan pengguna untuk memilih warna yang diinginkan.

Tabel 1.4: Beberapa komponen Swing

Untuk daftar yang lengkap dari komponen Swing, Anda dapat melihatnya di dokumentasi API.

7.5.1 Setting Up Top-Level Containers

Seperti disebutkan diatas, top-level containers seperti *Jframe* dan *Japplet* dalam Swing sangat tidak cocok dengan AWT. Ini adalah syarat menambahkan komponen ke dalam kontainer. Jika Anda ingin menambahkan langsung sebuah komponen kedalam kontainer sebagai kontainer AWT, pertama-tama Anda telah mendapatkan content pane dari kontainer. Untuk melakukan hal tersebut, Anda akan menggunakan method *getContentPane* dari container.

7.5.2 Contoh JFrame

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

class SwingDemo {
    JFrame frame;
    JPanel panel;
    JTextField textField;
    JButton button;
    Container contentPane;
    void launchFrame() {
        /* inisialisasi */
        frame = new JFrame("My First Swing Application");
```

```

        panel = new JPanel();
        textField = new JTextField("Default text");
        button = new JButton("Click me!");
        contentPane = frame.getContentPane();
        /* menambahkan komponen-komponen ke panel- menggunakan
        FlowLayout sebagai default */
        panel.add(textField);
        panel.add(button);
        /* menambahkan komponen-komponen contentPane- menggunakan
        BorderLayout */
        contentPane.add(panel, BorderLayout.CENTER);
        frame.pack();
        //menyebabkan ukuran frame menjadi dasar pengaturan komponen
        frame.setVisible(true);
    }
    public static void main(String args[]) {
        SwingDemo sd = new SwingDemo();
        sd.launchFrame();
    }
}

```

Perlu diperhatikan pada package *java.awt* masih saja diimpor karena layout manager yang digunakan terdapat pada package tersebut. Juga, memberi judul pada frame dan mengepack komponen di dalam frame dapat juga dilakukan untuk frame AWT.

Petunjuk penulisan program:

Perhatikan penulisan kode yang digunakan pada contoh ini tampak berlawanan dengan contoh untuk AWT. Komponen dideklarasikan sebagai fields, method launchFrame ditentukan, dinisialisasikan dan penambahan semua komponen dilaksanakan di dalam method launchFrame. Kita tidak lagi meng-extend Frame class. Keuntungan penggunaan model ini akan lebih berguna ketika sampai pada event handling.

Berikut adalah keluaran dari program diatas.



Gambar 1.4.2: Hasil contoh program

7.5.3 Contoh JOptionPane

```

import javax.swing.*;

class JOptionPaneDemo {
    JOptionPane optionPane;
    void launchFrame() {
        optionPane = new JOptionPane();
        String name = optionPane.showInputDialog("Hi, what's your
                                                name?");
        optionPane.showMessageDialog(null,
            "Nice to meet you, " + name + ".", "Greeting...",
            optionPane.PLAIN_MESSAGE);
    }
}

```

```
        System.exit(0);  
    }  
    public static void main(String args[]) {  
        new JOptionPaneDemo().launchFrame();  
    }  
}
```

Lihat, begitu mudahnya memasukkan input dari user.
Berikut ini adalah hasil dari contoh program diatas



Gambar 1.4.3: Hasil contoh program

7.6 Latihan

7.6.1 Tic-Tac-Toe

Buatlah tampilan GUI untuk program tic-tac-toe. Papannya terdiri dari enam kotak. Ingatlah bahwa Anda akan menambahkan kode ini pada tahap akhir untuk mengatasi interaksi antar pengguna. Jadi, desainlah papan Anda dengan benar. Pastikanlah Anda memilih komponen yang pantas untuk papan tersebut. keluarkan semua sisi artistik Anda. Anda dapat menggunakan AWT atau Swing untuk latihan ini.



Gambar 1.5.1: papan Tic-Tac-Toe