Bab 7
Abstract Windowing Toolkit dan Swing

7.1 Tujuan

Tanpa mempelajari tentang *grapichal 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.

Setelah menyelesaikan bab ini, Anda diharapkan untuk :

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 komplek 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 dengan 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 kelas penting pada kontainer yang telah disediakan oleh AWT.

AWT Class	*Description*
Komponen	Abstract Class untuk objek yang dapat ditampilkan pada console dan berinteraksi dengang user. Bagian utama dari semua kelas 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 objek yang lainnya.Tidak memiliki border dan menubar.

Frame	Turunan dari window class. Window dengan judul, menubar, border dan pengatur ukuran di pojok. Memiliki empat konstruktor , dua diantaranya memiliki penulisan seperti dibawah ini :
	`Frame()`
	`Frame(String title)`

Tabel 1.2.1: Kelas kontainer AWT

Untuk mengatur ukuran window, menggunakan metode 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 metode *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 metode grafik ditemukan dalam *Graphic* class. Dibawah ini adalah daftar dari beberapa metode.

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

Tabel 1.2.2a: Beberapa metode dari kelas Graphics

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

Constructor Format	***Description***
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 metode di dalam *Graphic* class.

```
import java.awt.*;

public class GraphicPanel extends Panel {
   GraphicPanel() {
      setBackground(Color.black);   //constant in Color class
   }
   public void paint(Graphics g) {
      g.setColor(new Color(0,255,0));     //green
      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 subkelas dari *Components* class.

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());  //more on this later
        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 metode *setLayout* dari *Container* class*.* Metode ini dapat ditulis sebagai berikut.

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

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

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

Metode 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 objek kontainer. Anda akan memanggil metode ini untuk setiap komponen.

7.4.1 FlowLayout Manager

FlowLayout Manager adalah default manager untuk *Panel* class dan subkelasnya, termasuk applet class. 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 konstruktor seperti daftar di bawah ini.

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

FlowLayout Constructors
Membuat objek baru FlowLayout dengan posisi spesifik dan lima unit horizontal dan vertikal gap dimasukkan pada komponen sebagai default.
`FlowLayout(int align, int hgap, int vgap)`
Membuat objek baru FlowLayout dengan argumen pertama sebagai posisi pada komponen dan *hgap* untuk horizontal dan *vgap* untuk vertikal pada komponen

Tabel 1.3.1: Konstruktor 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 objek *Window*, termasuk objek dari subkelas *Window* yaitu tipe Frame dan Dialog.

Konstruktor BorderLayout
`BorderLayout()`
Membuat objek BorderLayout baru tanpa spasi yang diaplikasikan diantara komponen yang berbeda.
`BorderLayout(int hgap, int vgap)`
Membuat objek BorderLayout baru dengan spasi unit *hgap* horizontal dan unit *vgap* vertikal yang diaplikasikan diantara komponen yang berbeda.

Tabel 1.3.2: Konstruktor 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 metode menambahkan dan melewatkan 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

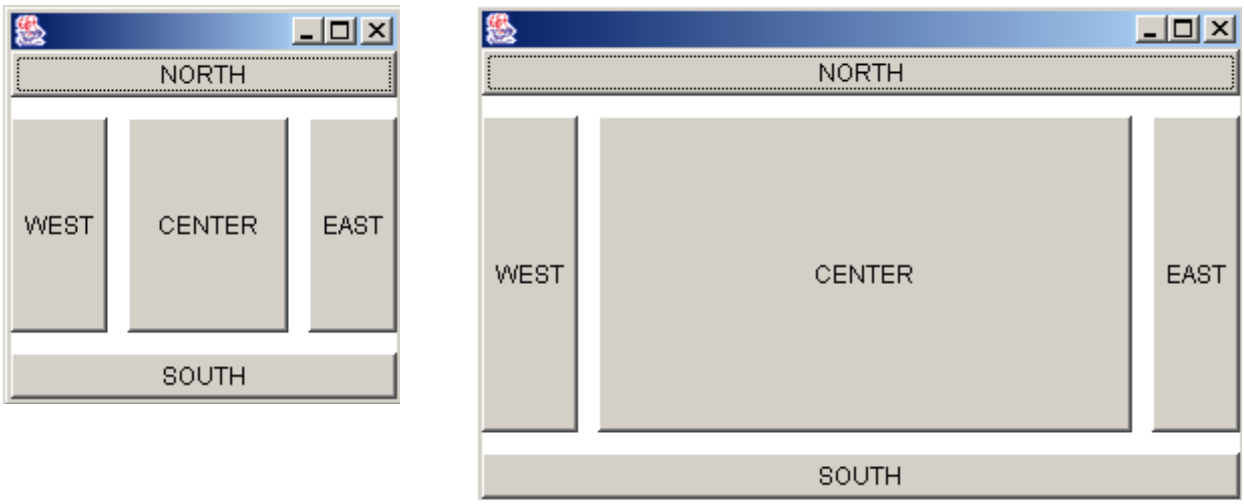
Berikut ini adalah contoh program yang menunjukkan bagaimana *BorderLayout* bekerja.

```
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 mempedulikan ukuran sebenarnya dari komponen.

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

Konstruktor GridLayout
`GridLayout()`
Membuat objek GridLayout baru dengan satu baris dan satu kolom sebagai default
`GridLayout(int rows, int cols)`
Membuat objek GridLayout baru dengan jumlah baris dan kolom sesuai dengan keinginan
`GridLayout(int rows, int cols, int hgap, int vgap)`
Membuat objek GridLayout baru dengan jumlah baris dan kolom yang ditentukan. Unit spasi hgap horizontal dan vgap vertikal diaplikasikan ke dalam komponen.

Tabel 1.3.3: Konstruktor GridLayout

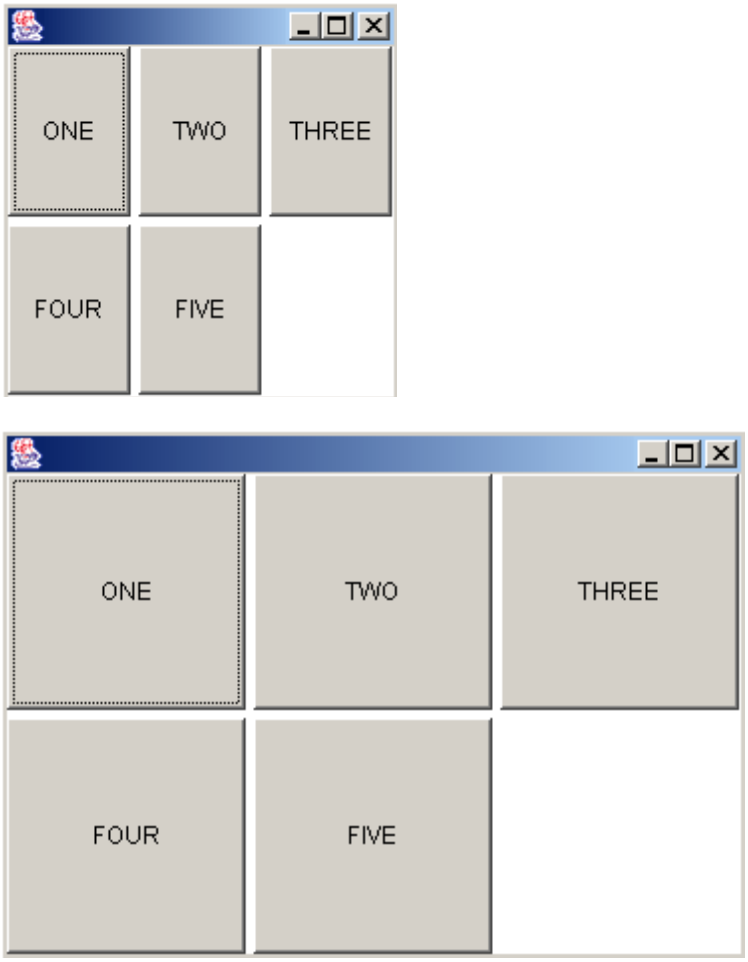
Cobalah program ini.

```
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 hasil dari program.



Gambar 1.3.3: hasil contoh program

7.4.4 Panel dan Tampilan kompleks

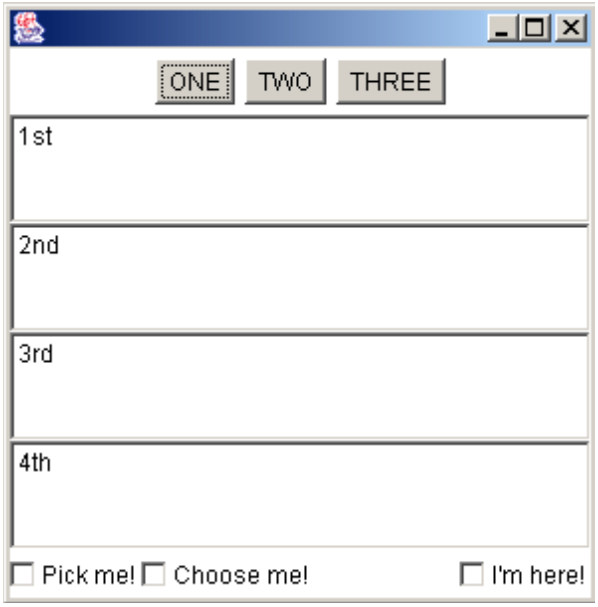
Untuk membuat tampilan yang lebih komplek, Anda dapat menggabungkan layout manager yang berbeda dengan menggunakan panel. Ingatlah bahwa panel adalah kontainer dan komponen pada saat yang sama. Anda dapat memasukkan komponen ke dalam panel dan kemudian menambahkan panel ke dalam region yang Anda inginkan 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();
      /* North Panel */
      //Panels use FlowLayout by default
      panelNorth.add(new Button("ONE"));
      panelNorth.add(new Button("TWO"));
      panelNorth.add(new Button("THREE"));
      /* Center Panel */
      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"));
      /* South Panel */
      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);
      /* Adding the Panels to the Frame container */
      //Frames use BorderLayout by 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 hasil dari program.



Gambar 1.3.4: Hasil dari contoh program

7.5 Komponen Swing

Seperti pada package AWT, package dari Swing menyediakan banyak kelas untuk membuat aplikasi GUI. Package tersebut dapat ditemukan di *javax.swing*. Perbedaan utama antara keduanya adalah komponen Swing ditulis menyeluruh menggunakan Java mengingat yang belakangan tidak. Kesimpulannya, program GUI ditulis menggunakan banyak kelas dari package Swing yang mempunyai tampilan look and feel yang sama meski dijalankan pada beda paltform. 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.

Komponen Swing	***Penjelasan***
JComponent	Kelas induk untuk semua komponen Swing, tidak termasuk top-level kontainer
JButton	Tombol “push”. Korespondesi pada button class dalam package AWT
JCheckBox	Item yang dapat dipilih atau tidak oleh pengguna. Korespondensi pada checkbox class dalam package AWT

Komponen Swing	*Penjelasan*
JFileChooser	Mengijinkan pengguna untuk memilih sebuah file. Korespondensi pada filechooser class dalam package AWT
JTextField	Mengijinkan untuk mengedit text satu baris. Korespondensi pada textfield class dalam package AWT.
JFrame	Turunan dan korepondensi pada frame class 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. Kontainer class sederhana tetapi bukan top-level. Korespondensi pada panel class dalam package AWT.
JApplet	Turunan dan korepondensi ke Applet class dalam package AWT. Juga sedikit tidak cocok dengan applet class dalam kaitannya dengan menambahkan komponen pada kontainer
JOptionPane	Turunan Jcomponent. Disediakan untuk mempermudah menampilkan pop-up kotak dialog.
JDialog	Turunan dan korespondensi pada dialog class dalam package AWT. Biasanya digunakan untuk menginformasikan sesuatu kepada pengguna atau prompt pengguna untuk input.
JColorChooser	Turunan Jcomponent. Mengijinkan pengguna untuk memilih warna

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 metode *getContentPane* dari kontainer.

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() {
```

```
        /* initialization */
        frame = new JFrame("My First Swing Application");
        panel = new JPanel();
        textField = new JTextField("Default text");
        button = new JButton("Click me!");
        contentPane = frame.getContentPane();
        /* add components to panel- uses FlowLayout by default */
        panel.add(textField);
        panel.add(button);
        /* add components to contentPane- uses BorderLayout */
        contentPane.add(panel, BorderLayout.CENTER);
        frame.pack();
        //causes size of frame to be based on the components
        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, metode launchFrame ditentukan, dinisialisasikan dan penambahan semua komponen dilaksanakan di dalam metode launchFrame. Kita tidak lagi meng-extend Frame class. Keuntungan penggunaan gaya 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