MODUL PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK

GUI DAN EVENT HANDLING

Deskripsi Singkat

Praktikum pemrograman berorientasi objek adalah praktikum yang menggunakan bahasa Java sebagai bantuan dalam memahami konsep pemrograman berorientasi objek. Materi praktikum berisi teori, latihan dan soal pemrograman.

Tujuan

1. Memahami konsep GUI dan Event handling.
2. Membuat aplikasi sederhana dengan menggunakan konsep GUI dan event handling.

Prasyarat

Siswa telah melakukan praktikum 1-12.

Materi 1 : GUI

GUI adalah singkatan dari Graphical User Interface. GUI merupakan desain aplikasi dengan tampilan visual sehingga pengguna dapat dengan mudah menggunakan aplikasi. 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.

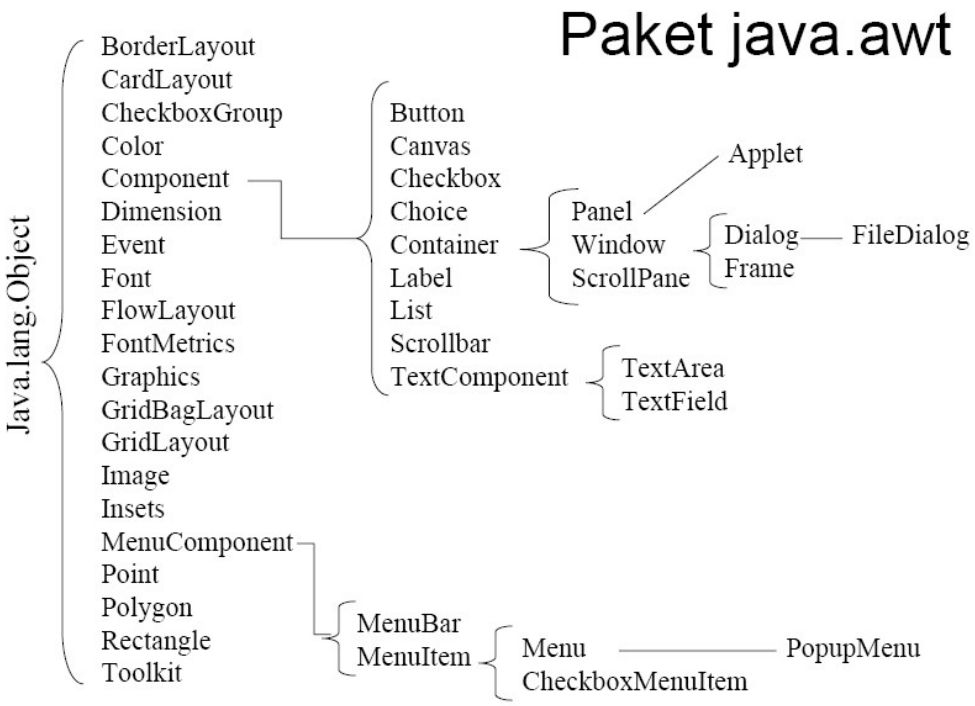
AWT dan Swing menyediakan komponen GUI yang dapat digunakan dalam membuat aplikasi Java dan applet. Anda akan mempelajari applet pada bab berikutnya. Tidak seperti beberapa komponen AWT yang menggunakan native code, keseluruhan Swing ditulis menggunakan bahasa pemrograman Java. Swing menyediakan implementasi 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-sama dengan komponen Swing.

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.

Komponen GUI pada AWT

Pembagian komponen GUI pada AWT (di dalam paket java.awt) dapat dilihat pada gambar di bawah.



Berikut ini adalah daftar dari beberapa kelas penting pada kontainer yang telah disediakan oleh AWT.

- **Component** : Abstract Class untuk objek yang dapat ditampilkan pada console dan berinteraksi dengang user. Bagian utama dari semua kelas AWT.
- **Container** : 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)
  ```

  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)
```

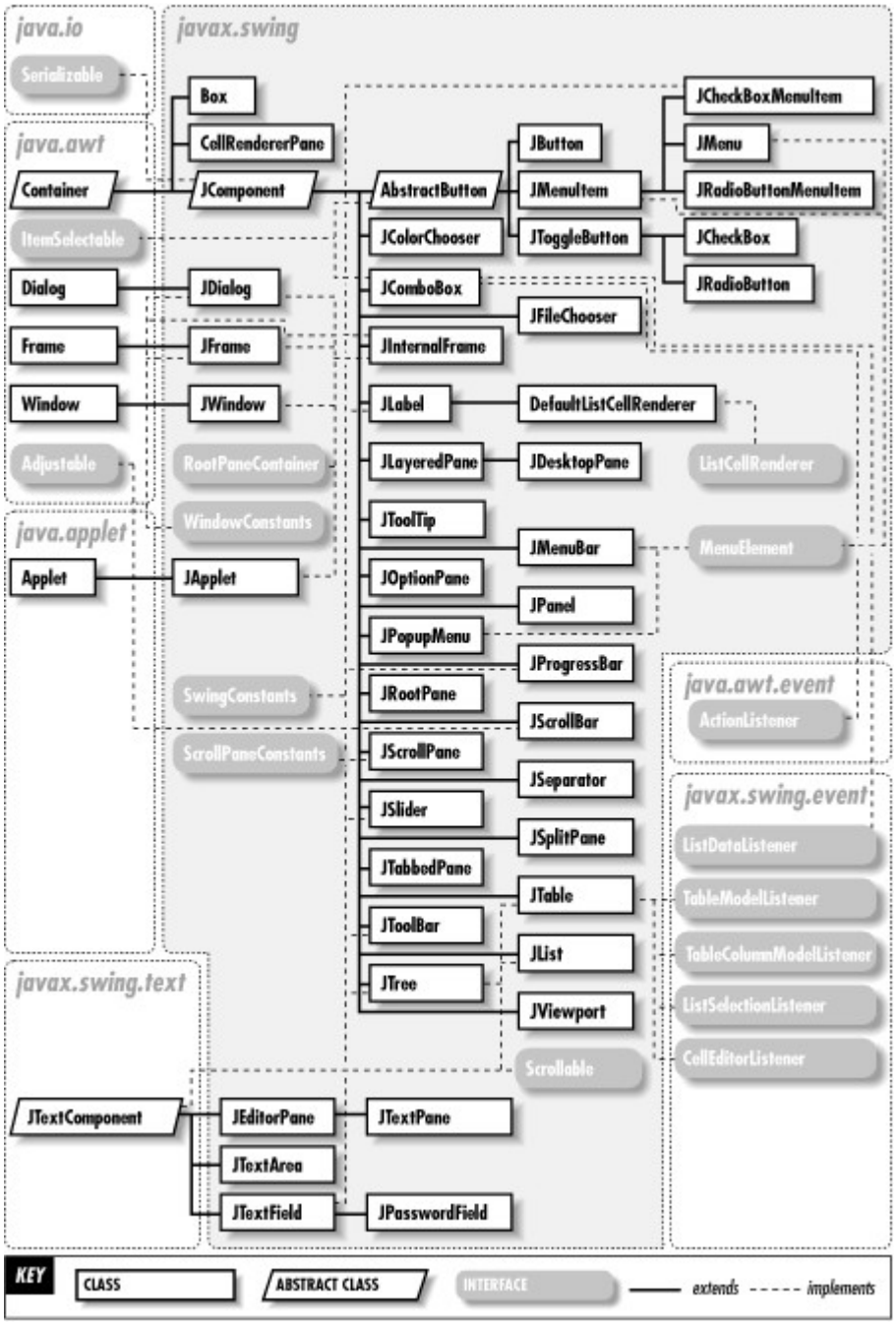
Komponen GUI pada Swing

Komponen GUI pada Swing terdapat dalam paket javax.swing. Berikut adalah daftar dari beberapa komponen Swing:

- **JComponent** : Kelas induk untuk semua komponen Swing, tidak termasuk top-level Container
- **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.
- **JApplet** : Turunan dan korepondensi ke Applet class dalam package AWT. Juga sedikit tidak cocok dengan applet class dalam kaitannya dengan menambahkan komponen pada container
- **JButton** : Tombol “push”. JButton adalah komponen berbentuk tombol. Komponen ini banyak digunakan sebagai eksekusi terhadap tindakan yang diinginkan. Pada aplikasi komputer, biasanya dibutuhkan tombol untuk mengeksekusi sebuah perintah.
- **JLabel** : komponen yang digunakan untuk membuat tulisan atau gambar pada frame sebagai suatu informasi untuk pengguna program.
- **JTextField** : komponen yang digunakan untuk memasukkan sebaris string yang selanjutnya dapat digunakan sebagai input bagi proses selanjutnya.
- **JTextArea** : komponen yang mirip dengan JTextField tetapi dapat menampung lebih dari 1 baris.
- **JCheckBox** : komponen yang digunakan ketika pengguna memerlukan komponen untuk melakukan satu atau banyak pilihan sekaligus .
- **JRadioButton** : komponen yang digunakan ketika pengguna perlu memilih satu diantara beberapa piliha.
- **JComboBox** : komponen yang digunakan untuk memilih satu diantara sekian banyak pilihan yang berbentuk semacam TextField dan ada panah ke bawah.
- **JFileChooser** : Mengijinkan pengguna untuk memilih sebuah file.
- **JColorChooser** : Turunan Jcomponent. Mengijinkan pengguna untuk memilih warna.
- **JTable** : digunakan untuk menampilkan data dalam bentuk tabel (biasanya dalam pemrograman database).
- **JScrollPane** : komponen yang digunakan untuk menggerakkan obyek ke atas, ke bawah, atau ke samping agar semua obyek terlihat di layar.
- **JMenu** : komponen yang digunakan untuk membuat menu.
- **JInternalFrame** : frame yang hanya dapat berada dalam frame lain.

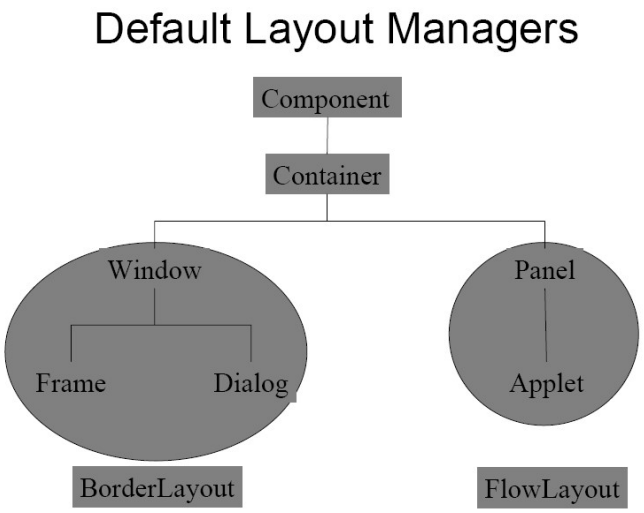
- **JOptionPane** : Turunan Jcomponent. Disediakan untuk mempermudah menampilkan popup kotak dialog.
- **JDialog** : Turunan dan korespondensi pada dialog class dalam package AWT. Biasanya digunakan untuk menginformasikan sesuatu kepada pengguna atau prompt pengguna untuk input.

Pembagian komponen GUI pada Swing dapat dilihat pada gambar di bawah.



Layout

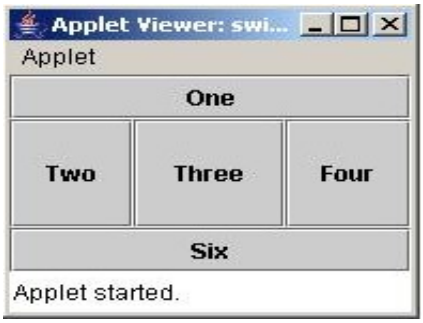
Pengaturan layout digunakan untuk mengatur posisi dari komponen visual penyusun program sesuai dengan desain user interface. Beberapa pilihan layout telah disediakan java, dimana keputusan untuk menggunakan jenis layout tertentu bergantung pada jenis aplikasi yang ingin dibuat serta tingkat kerapian yang diinginkan.



Berikut beberapa layout manager yang terdapat di dalam Java:

- **FlowLayout** : jenis pengaturan layout yang paling sederhana, dimana semua komponen akan tersusun dari kiri ke kanan sepanjang frame, dan akan pindah ke bawah bila telah sampai batas kanan frame. Default pada java.awt.

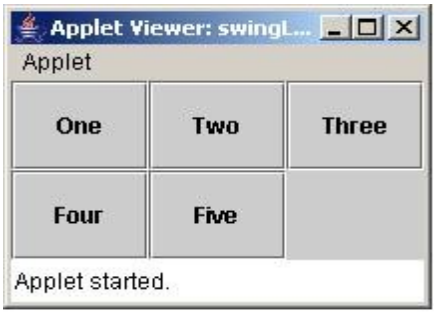
- **BorderLayout** : jenis layout yang bekerja dengan membagi frame menjadi lima bagian yaitu NORTH, EAST, SOUTH, WEST dan CENTER. Komponen visual dapat diletakkan pada bagian-bagian tersebut.



- **NoneLayout** : jenis layout yang dapat menghasilkan tampilan yang rapi karena kita dapat mengatur posisi komponen secara detil berdasar koordinatnya. Konsekuensinya dengan layout ini waktu yang

diperlukan relatif lebih banyak dibanding layout yang lain karena kita perlu menentukan posisi koordinat tiap komponen.

- **GridLayout** : jenis layout yang bekerja berdasar baris dan kolom. Dengan layout ini kita dapat memberikan argumen banyaknya baris dan kolom sesuai dengan kebutuhan.



- **GridBagLayout** : ukuran grid bisa berubah, lebih dari satu komponen bisa masuk pada satu grid.
- **CardLayout** : komponen ditimpa seperti kartu, hanya satu komponen yg nampak pada satu waktu.
- **BoxLayout** : Komponen disusun kiri-kanan atau atas-bawah.

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.

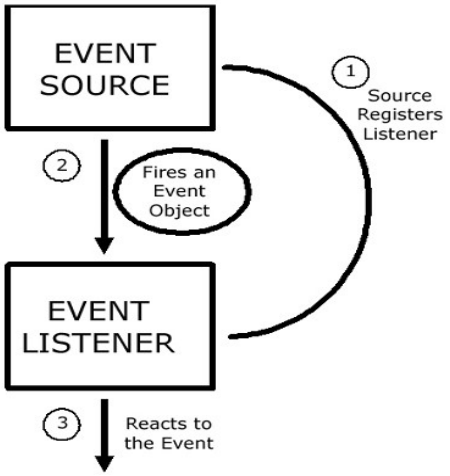
```
public void setBounds(int x, int y, int width, int height)
```

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.

Materi 2 : Event Handling

Java menggunakan *delegation event model* untuk mengendalikan peristiwa (event). Pada model ini terdapat:

- ***Event source*** (sumber peristiwa) : mengacu pada komponen GUI yang menghasilkan event.
  Contoh: jika pengguna menekan button, maka event sourcenya adalah button.
- ***Event listener/event handler*** (pendengar peristiwa) : objek yang mendengar peristiwa dan melakukan tindakan terhadap peristiwa tersebut.
  Contoh: ketika button ditekan, listener akan mengendalikan dengan handler yang sesuai.
- ***Event object*** : ketika sebuah event terjadi, maka objek event akan diciptakan. Objek berisi semua informasi yang perlu mengenai event yang telah terjadi.



Langkah-langkah membuat aplikasi yang dapat menghandel event:

- Menyediakan program sebagai event listener. Contoh:
```
import java.awt.event.*;
public class NamaClass implements ActionListener {
```
- Mendaftarkan komponen pada event listener. Contoh:
```
someComponent.addActionListener (instanceOfNamaClass)
```
- Melakukan tindakan terhadap event dari pengguna. Contoh:
```
public void actionPerformed (ActionEvent e){
… //kode yang mengakomodasi aksi dari pengguna}
```
- Implementasi program dengan inner class, contoh:
```
addMouseMotionListener(new MyMouseMotionListener());

class MyMouseMotionListener extends MouseAdapter {
   public void mouseDragged(MouseEvent e) {
   }
}
```
- Implementasi program dengan anonymous inner class, contoh:
```
addMouseMotionListener(new MouseMotionAdapter(){
   public void mouseDragged(MouseEvent e) {
      ...
   }
}); // tutup dengan titik koma
```

Kategori	Name Interface	Method
Action	ActionListener	actionPerformed(ActionEvent)
Item	ItemListener	itemStateChanged(ItemEvent)
Mouse	MouseListener	mousePressed(MouseEvent) mouseReleased(MouseEvenr) mouseEntered(MouseEvent)

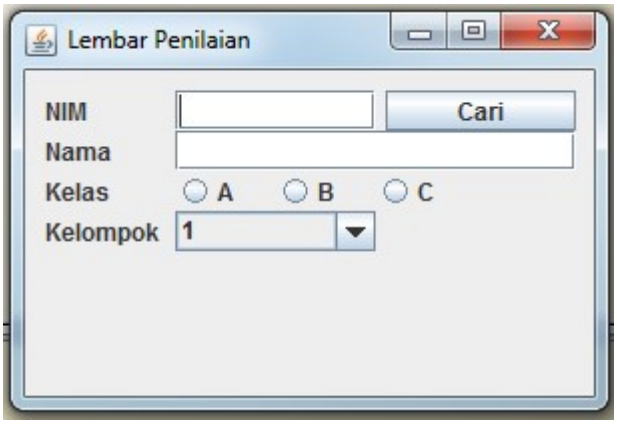
		mouseExited(MouseEvent) mouseClicked(MouseEvent)
Mouse Motion	MouseMotionListener	mouseDragged(MouseEvent) mouseMoved(MouseEvent)
Key	KeyListener	keyPressed(KeyEvent) keyReleased(KeyEvent) keyTyped(KeyEvent)
Focus	FocusListener	focusGained(FocusEvent) focusLost(FocusEvenr)
Adjustment	AdjustmentListener	adjustmentValueChanged(AdjustmentEvent)
Component	ComponentListener	componentMoved(ComponentEvent) componentHidden(ComponentEvent) componentResized(ComponentEvent) componentShown(ComponentEvent)
Window	WindowListener	windowClosing(WindowEvent) windowOpened(WindowEvent) windowIconified(WindowEvent) windowDeiconified (WindowEvent) windowClosed(WindowEvent) windowActivated(WindowEvent) windowDeactivated (WindowEvent)
Container	ContainerListener	componentAdded(ContainerEvent) componentRemoved(ContainerEvent)
Text	TextListener	textValueChanged(TextEvent)

LATIHAN 1

Buka BlueJ. Download projek gui-praktikum yang terdapat pada website:
http://informatika.unsyiah.ac.id/viska/pbo/gui-praktikum.zip

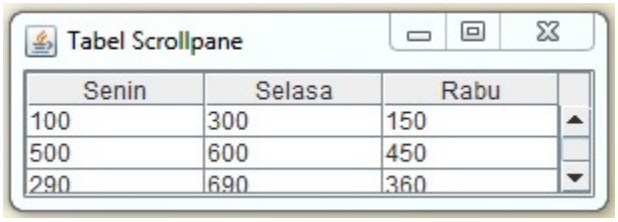
Di dalam projek tersebut terdapat beberapa class yang berkaitan dengan aplikasi gui sederhana dan event sederhana. Open editor dan coba cipta objek untuk class agar anda dapat memahami cara membuat aplikasi gui yang melibatkan pembuatan container, pembuatan component control dan cara menampilkannya.

1. Class AplikasiPenilaian akan menghasilkan tampilan seperti gambar di bawah.



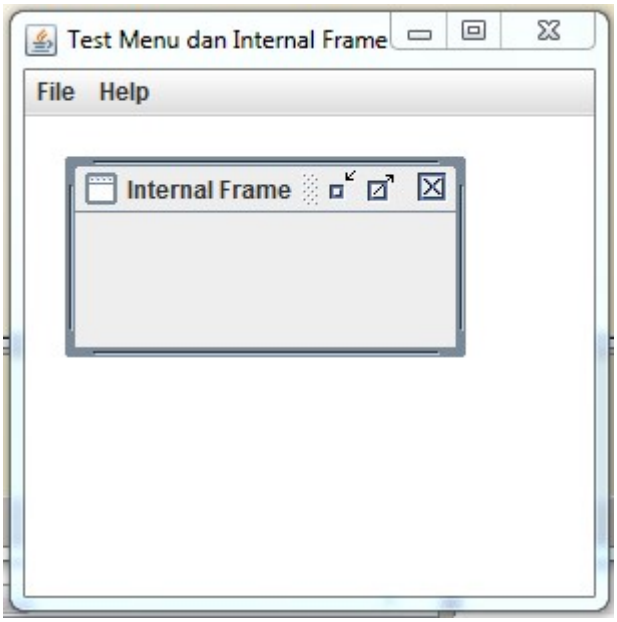
Container yang digunakan pada class tersebut adalah JFrame dan JPanel.
Component control yang digunakan pada class tersebut adalah JLabel, JTextField, JButton, JRadioButton, ButtonGroup, JComboBox.
Layout yang dipakai adalah NoneLayout (null) sehingga peletakan semua component perlu menggunakan method setBounds.
Coba buat tampilan yang sama dengan menggabungkan layout-layout yang lain (yang sesuai) seperti FlowLayout, BorderLayout, GridLayout.

2. Class TabelScrollPane akan menghasilkan tampilan seperti gambar di bawah.

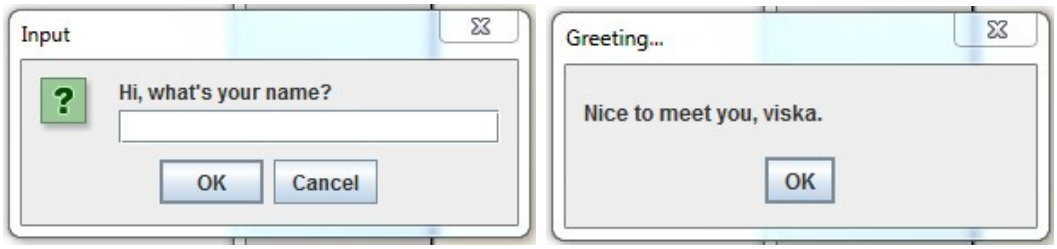


Container yang digunakan pada class tersebut adalah JFrame, JPanel, JScrollPane.
Component control yang digunakan pada class tersebut adalah JTable.
Layout yang dipakai adalah BorderLayout.
JScrollPane digunakan untuk memberikan scroll pada bagian samping kanan untuk component yang memerlukan scrolling.

3. Class MenuInternalFrame akan menghasilkan tampilan seperti gambar di bawah.
Container yang digunakan pada class ini adalah JFrame dan JInternalFrame.
Pada class ini ditunjukkan cara membuat dan menampilkan internal frame yang akan muncul di dalam frame induknya.
Class ini juga menampilkan cara membuat Menubar serta menu item di dalamnya.

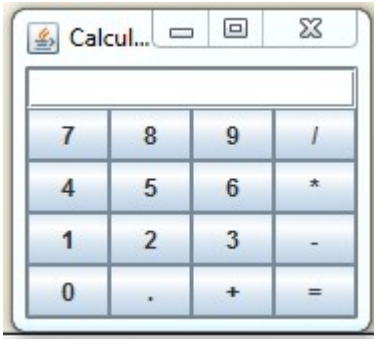


4. Class OptionPaneDemo akan menghasilkan tampilan seperti di bawah.



Class ini menunjukkan cara menggunakan JOptionPane yang dapat digunakan untuk menampilkan input dialog, message dialog dan confirm dialog.

5. Class CalcInterface menghasilkan tampilan seperti di bawah. Class ini menggunakan layout BorderLayout dan GridLayout.



Setelah kita belajar mengenai event, coba anda buat even-event di balik semua JButton sehingga calculator ini dapat berjalan normal.

Aplikasi event handling dapat dilihat pada class TambahKurang, TambahKurang1 dan EventSederhana. Class TambahKurang dan TambahKurang1 menghasilkan aplikasi dengan tampilan dan perilaku yang sama namun keduanya memiliki perbedaan dalam menangani event dari event source.

Class TambahKurang langsung mengimplementasi interface ActionListener, dapat dilihat disini:

```
public class TambahKurang implements ActionListener
```

Oleh karena itu setiap event source harus diregister dengan method yang terkait dengan ActionListener yaitu addActionListener seperti contoh di bawah:

```
//Register actionlistener pada button --
krgButton.addActionListener(this);
```

Karena pada class TambahKurang mengimplement ActionListener, maka kelas ini **harus** mengoverride method actionPerformed, seperti di bawah:

```
public void actionPerformed(ActionEvent event)
{
    //cara ini digunakan dgn menguji teks dari button
    String command = event.getActionCommand();

    if(command.equals("--"))
        this.kurangX();
    else if(command.equals("++"))
        this.tambahX();

    nilai.setText(Integer.toString(this.getX()));
}
```

atau

```
public void actionPerformed(ActionEvent event)
{
    //Jika objek yg diuji berbeda-beda, sebaiknya menggunakan method getSource
    Object obj = event.getSource();

    if(obj == krgButton)
        this.kurangX();
    else if(obj == tmbButton)
        this.tambahX();

    nilai.setText(Integer.toString(this.getX()));
     //*/
}
```

Kedua bentuk actionPerformed di atas berbeda dari segi penanganan event sourcenya. Pada cara yang pertama, event source dikenali melalui text yang tertera pada Button. Sedangkan pada cara yang kedua, event source langsung dikenali melalui objeknya.

Pada class TambahKurang1, class ini tidak mengimplement Listener apapun pada classnya, namun memanfaatkan anonymous inner class untuk meregister listener ke event source. Berikut merupakan method yang digunakan untuk melakukan tindakan jika ada event dari pengguna terhadap event source:

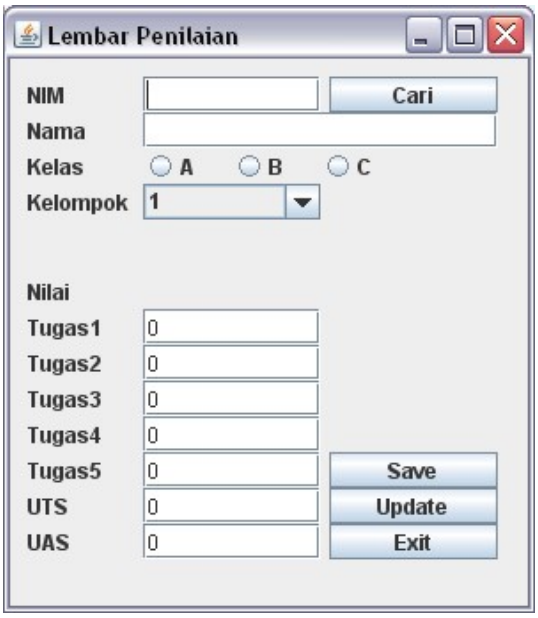
```
public void AksiReaksi()
{
    tmbButton.addActionListener(
        new ActionListener()
        {
            public void actionPerformed(ActionEvent event)
            {
                tambahX();
                nilai.setText(Integer.toString(getX()));
            }
        }
    );

    krgButton.addActionListener(
        new ActionListener()
        {
            public void actionPerformed(ActionEvent event)
            {
                kurangX();
                nilai.setText(Integer.toString(getX()));
            }
        }
    );
}
```

Pahami kedua cara event handling pada class TambahKurang dan TambahKurang1. Diskusikan jika perlu.

Diantara kedua cara pada class TambahKurang dan TambahKurang1, anda dapat memilih cara apapun yang anda sukai dan anda rasa paling mudah untuk menangani event.

SOAL-SOAL

1. Buatlah tampilan GUI seperti di bawah ini.



2. Buatlah class LayoutTester yang memiliki 5 komponen JButton didalamnya. Class LayoutTester dapat mengubah layout dari kelima JButton. Jika method setFlowLayout() yang dipanggil, maka tampilan kelima-lima JButton menggunakan layout FlowLayout. Jika method setBorderLayout() yang dipanggil, maka tampilan kelima-lima JButton menggunakan layout BorderLayout. Jika method setGridLayout() yang dipanggil, maka tampilan kelima-lima JButton menggunakan layout GridLayout. Contoh:

setBorderLayout ⇒

setFlowLayout ⇒

setGridLayout ⇒