MI2294
PEMROGRAMAN VISUAL

MODUL PRAKTIKUM

Departemen Teknologi Informasi
Telkom Applied Science School
2013

DAFTAR PENYUSUN

1. Versi 1 : 2013 10 : Reza Budiawan, M.T.

Daftar Isi

1 BAB I MATISSE BUILDER 1 (KOMPONEN SWING)

1.1 IDENTITAS

Kajian

Pengenalan Swing Java (Menggunakan Tools--Visual Editor/Matisse Builder)

Topik

1. Komponen Dasar Swing
2. Penggunaan Matisse Builder

Referensi

1. https://netbeans.org

Kompetensi Utama

1. Mahasiswa memahami konsep pemrograman swing
2. Mahasiswa mampu membuat halaman sederhana menggunakan komponen swing dibantu tool visual editor
3. Mahasiswa mampu memahami konsep penggunaan komponen swing: JLabel, JTextField, JComboBox, JTable, JRadioButton, JCheckBox, JMenuBar, JMenu, JMenuItem, JRadioButton, ButtonGroupdan JButton

Lama Kegiatan Praktikum

1. Pertemuan Terbimbing : 1 x 50 menit
2. Kegiatan Mandiri : 1 x 50 menit

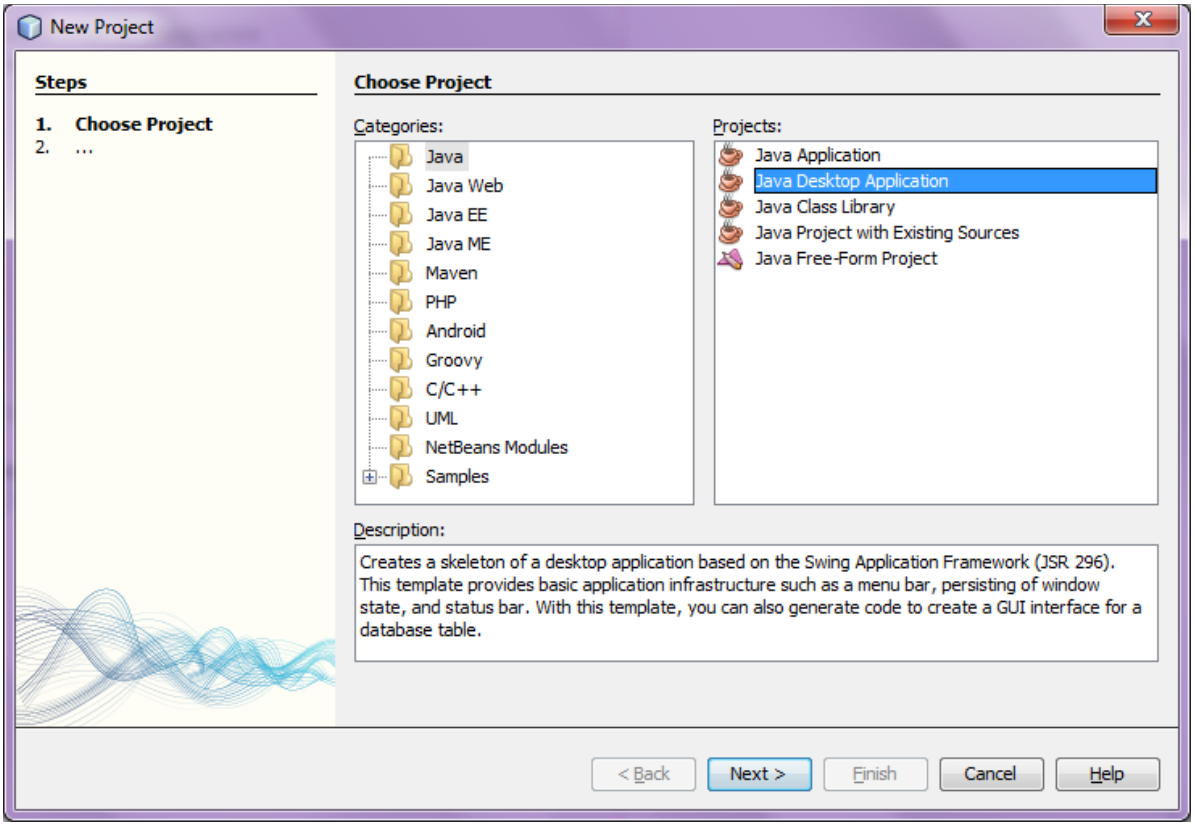
Parameter Penilaian

1. Jurnal Mandiri

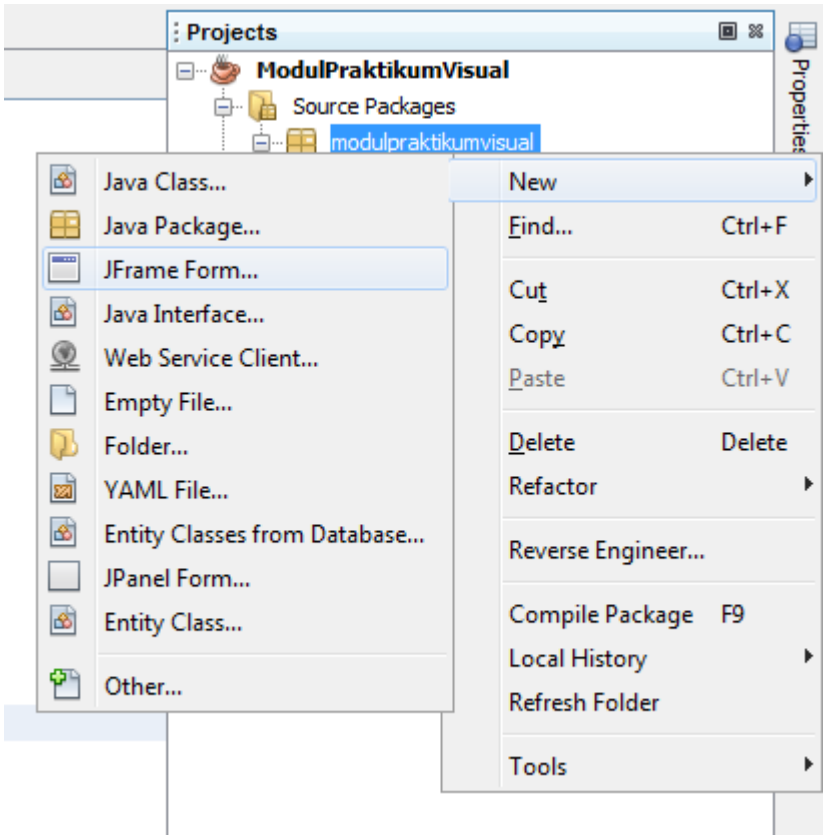
1.2 PENGENALAN MATISSE BUILDER

Netbeans memiliki sebuah project bernama Project Matisse untuk membuat sebuah builder GUI dengan basis swing menggunakan bahasa pemrograman java. Swing GUI builder ini membantu para programmer untuk membangun sebuah aplikasi desktop karena dapat membangun GUI secara visual dan bukan hanya sekedar text-based code. Dengan melakukan drag-and-drop komponen swing ke top level container-nya, sebuah aplikasi gui menggunakan bahasa java sudah dapat dibangun.

Untuk menggunakan matisse builder dapat memilih aplikasi "Java Desktop Application" pada saat pembuatan project baru.

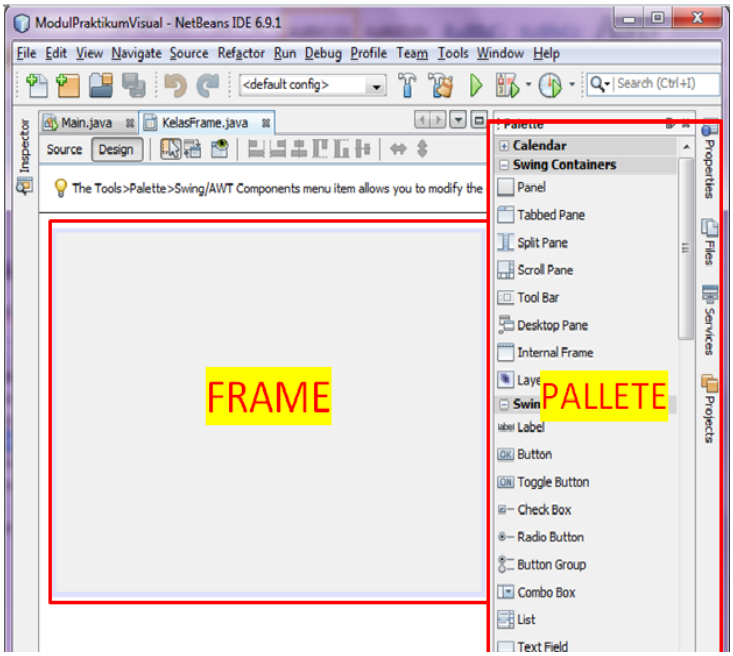


Atau dapat memilih “Java Application” dan menambahkan “JFrame Form” pada project tersebut



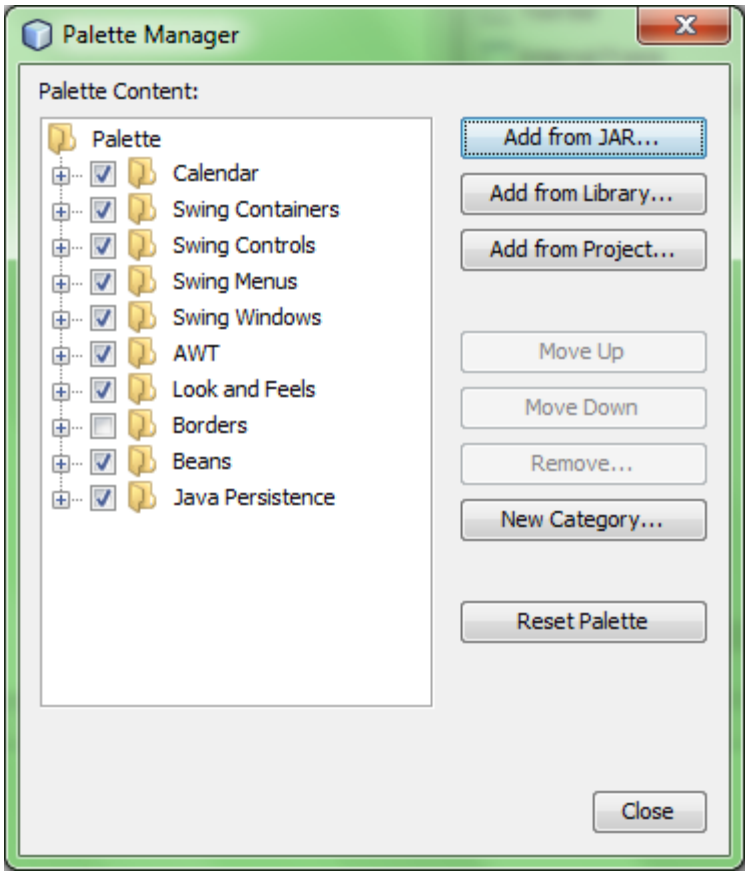
Hal ini akan menciptakan satu kelas java yang meng-extends JFrame yang merupakan container tertinggi dari sebuah aplikasi java.

Matisse Builder/GUI editor ini terdiri dari beberapa bagian, 2 diantaranya yaitu bagian frame dan palette.



Bagian frame merupakan sebuah class java yang meng-extends class dari komponen swing, yaitu JFrame. JFrame merupakan top-level-container pada paket swing. Bagian frame ini layaknya sebuah kanvas yang dapat diisi komponen lain dari paket swing, container ataupun komponen umum gui seperti button, textfield dan lainnya. Palette merupakan tempat peletakan komponen swing yang bisa ditambahkan ke sebuah frame. Penambahannya dilakukan dengan cara drag-and-drop.

Sedangkan, palette merupakan bagian dari matisse builder yang berisikan komponen swing/awt (komponen pembentuk gui menggunakan bahasa java). Komponen ini bisa ditambahkan pada bagian frame yang telah dibentuk sebelumnya. Palette dapat ditambahkan komponennya dari luar yang dibangun oleh pihak ketiga (third-party). Caranya, klik kanan di bagian palette, pilih palette manager. Klik “Add From Jar” untuk menambahkan library yang dibangun oleh pihak ketiga. Contoh penggunaannya pada saat menambahkan date picker ke komponen palette. Atau menambahkan komponen swing yang belum tercantum ke default palette (contohnya, Border).

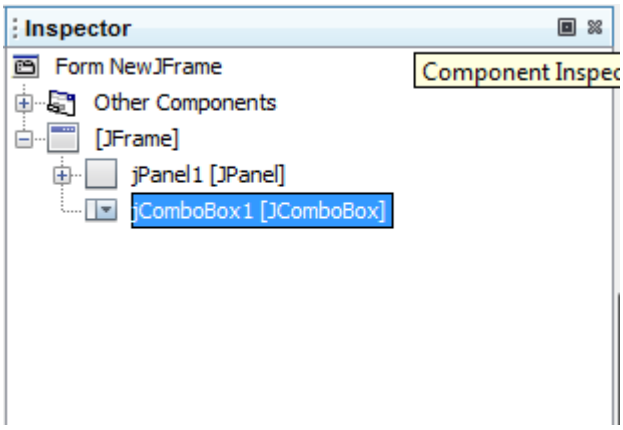


Palette Default terbagi ke dalam beberapa kategori, beberapa diantaranya adalah sebagai berikut:

a) Swing container: merupakan container pada pemrograman gui menggunakan bahasa java. Biasa digunakan untuk windowing (cara/bentuk menampilan aplikasi ke user)
b) Swing control: kategori yang menyimpan komponen swing yang penting seperti label untuk membuat tulisan, button untuk membuat tombol, combo box untuk menambahkan menu pull/drop down, dan lainnya
c) Swing menu: kategori untuk menambahkan menu yang terdapat pada bagian atas suatu window/frame. Menu memiliki hirarki tersendiri. Hirarki menu biasanya dituliskan sebagai "*Menu Bar → Menu → Menu Item | Menu Check Box | Menu Radio Button*". Pop-up menu digunakan untuk menambahkan menu "klik-kanan".

Menambahkan komponen pada matisse builder dilakukan dengan melakukan penarikan komponen yang ada di palette ke bagian frame. Pengaturan peletakan komponen di frame juga dilakukan secara visual. Saat melakukan drag-and-drop, sebuah objek dari kelas tertentu dari package swing akan dibentuk dan ditambahkan ke frame. Untuk beberapa komponen, terdapat modifikasi agar dapat digunakan sesuai keinginan. Hal-hal yang bisa diubah terdapat di window properties.

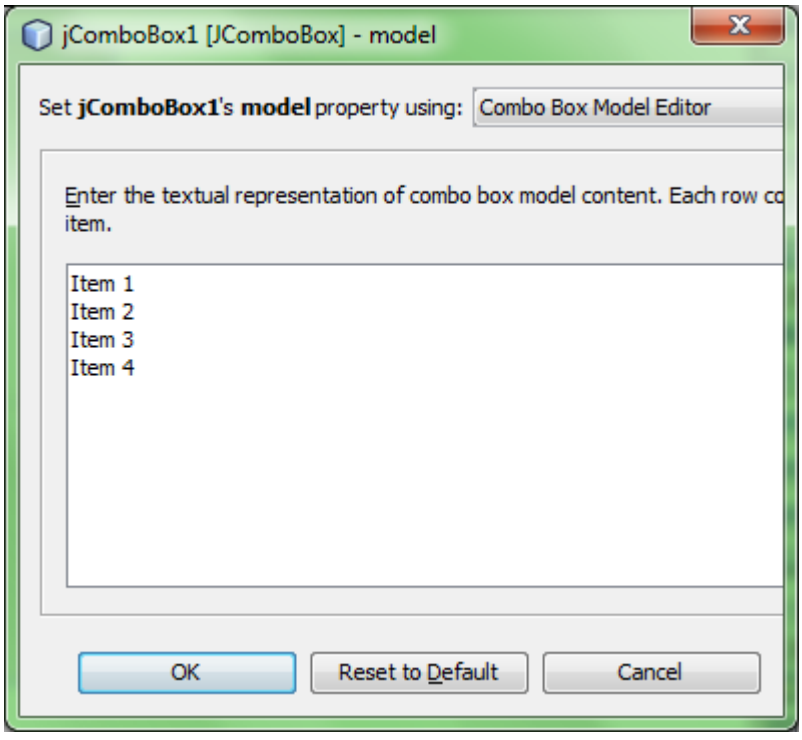
Untuk melihat susunan hirarki gui, bisa dilihat di window inspector (Netbeans 6.9.x ke bawah) atau di window navigator (Netbeans 7.0 ke atas).



Beberapa contoh penggunaan komponen yang dituliskan pada modul ini adalah penggunaan combo box, table, radio button—button group dan menu.

1.3 COMBO BOX

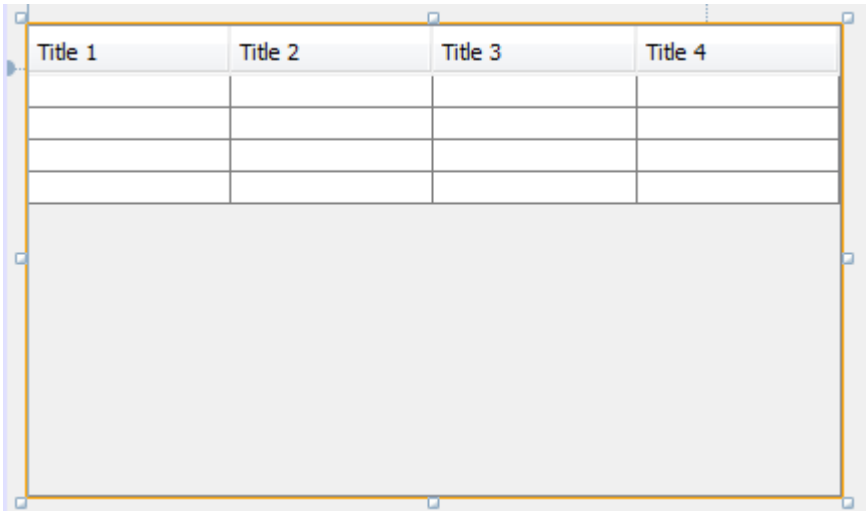
Combo box pada komponen swing diambil dari kelas JComboBox. Saat melakukan drag-and-drop, pada kelas frame, objek dari kelas JComboBox dibentuk. Setelah melakukan drag-and-drop dari palette ke frame, untuk mengubah pilihan yang ada di combo box, lihat properties, cari model. Pada model, tekan eclipse button (…) dan akan tampil tampilan sebagai berikut:



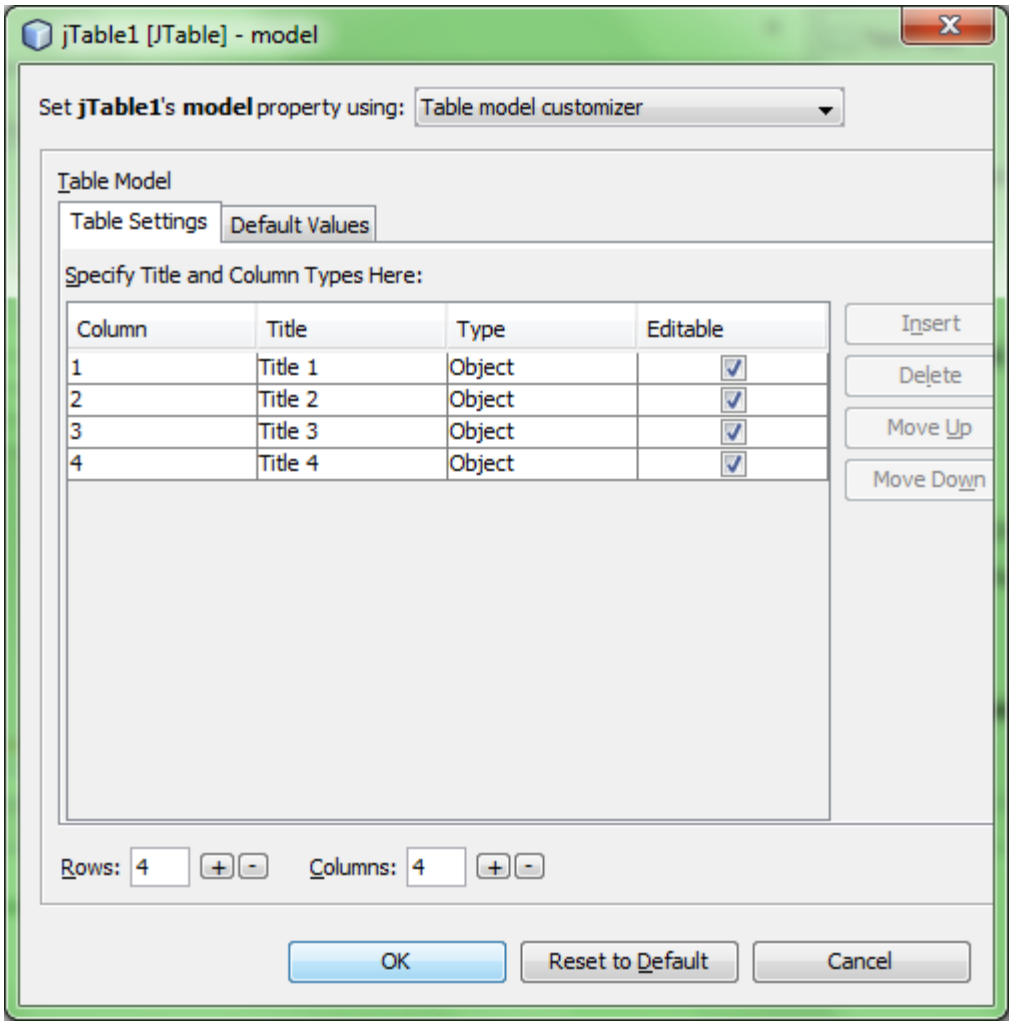
Item 1 sampai Item 4 merupakan isian dari combo box, gantilah “Item 1” sampai “Item 4” seperti yang diinginkan.

1.4 TABLE

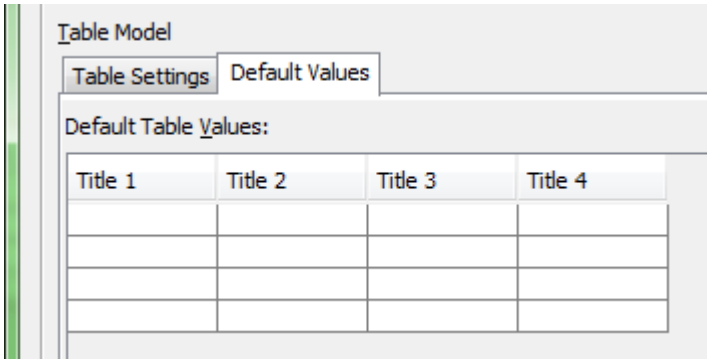
Sama seperti komponen lain, table yang objeknya dibentuk dari kelas JTable memiliki default tampilan sebagai berikut:



Jumlah kolom yang ditampilkan, serta title dapat diubah di bagian model pada properties table.



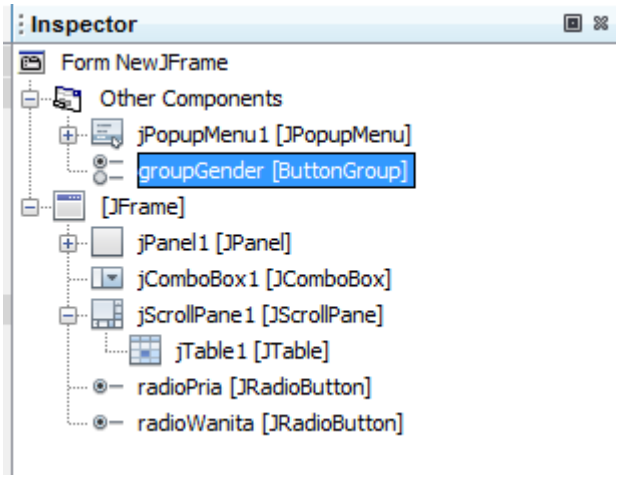
Rows merupakan jumlah baris yang ditampilkan (biasanya berisi data). Pengubahan data pada rows dapat dilakukan pada tab “default values”.



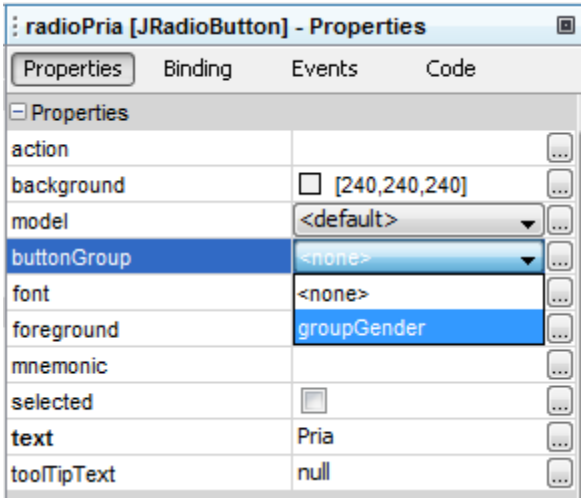
1.5 RADIO BUTTON & BUTTON GROUP

Radio Button biasa digunakan untuk memilih tepat 1 dari banyak pilihan yang ada. Untuk dapat meng-grupkan n radio button menjadi 1 kategori yang dapat dipilih, digunakan button group (contoh ada radio button "wanita" dan radio button "pria". Agar radio button ini dapat dipilih salah satu saja, digunakan 1 button group "gender")

Sebelum mengelompokkan radio button dalam 1 button group, sebaiknya drag-and-drop semua radio button terlebih dahulu ke frame. Lalu tambahkan button group. Untuk memastikan, pastikan button group dan radio button tercantum di window Inspector/Navigator.

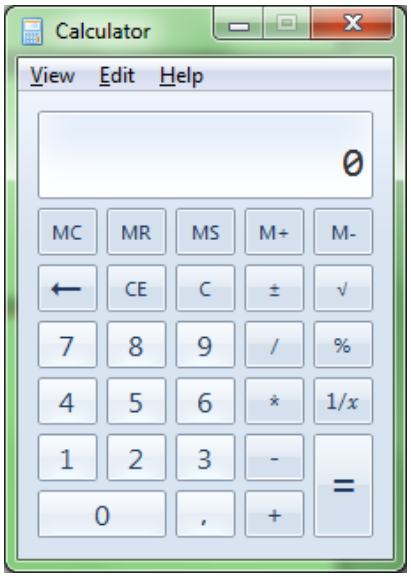


Untuk menjadikan 1 grup, pada properties radio button, pilih button group sesuai yang diinginkan

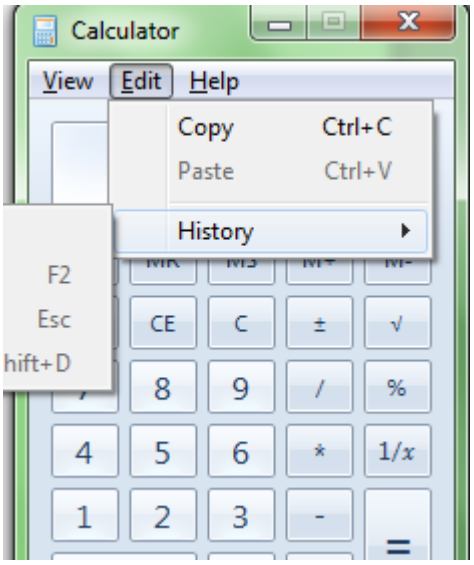


1.6 MENU BAR, MENU DAN MENU ITEM

Menu merupakan struktur menu yang paling klasik dari sebuah aplikasi. Biasa terdapat di sudut kiri atas sebuah aplikasi. Contoh: (menu yang terdapat pada kotak garis merah)



Menu Bar merupakan tempat meletakkan Menu. Dalam menu, terdapat menu item yang bisa dipilih (di-klik) yang menimbulkan suatu aksi. Dalam menu, masih boleh ditambahkan menu lain yang mengandung menu item.



Menu Item terbagi 3 jenis, yaitu menu item itu sendiri, menu item berupa radio button dan menu item berupa check box. Setiap komponen ini dapat ditambahkan ke canvas dengan menarik masing-masing komponen ke canvas kosong. Yang perlu diperhatikan, hirarkis dari menu harus ada di bawah JFrame (top-level container) dan bukan content container.

1.7 PRAKTIK

1.7.1 Contoh Kasus

Mine Nakahara merupakan pemilik penginapan terkenal di desa konoha. Hinata ingin memperluas cakupan bisnisnya ke bidang IT. Salah satu langkah nyata yang ia ambil adalah membuat aplikasi untuk menghitung biaya penginapan bagi para penyewa penginapannya.

Hinata ingin membuat aplikasi tersebut dengan GUI sebagai berikut:

Buatlah GUI seperti yang diinginkan Hinata.

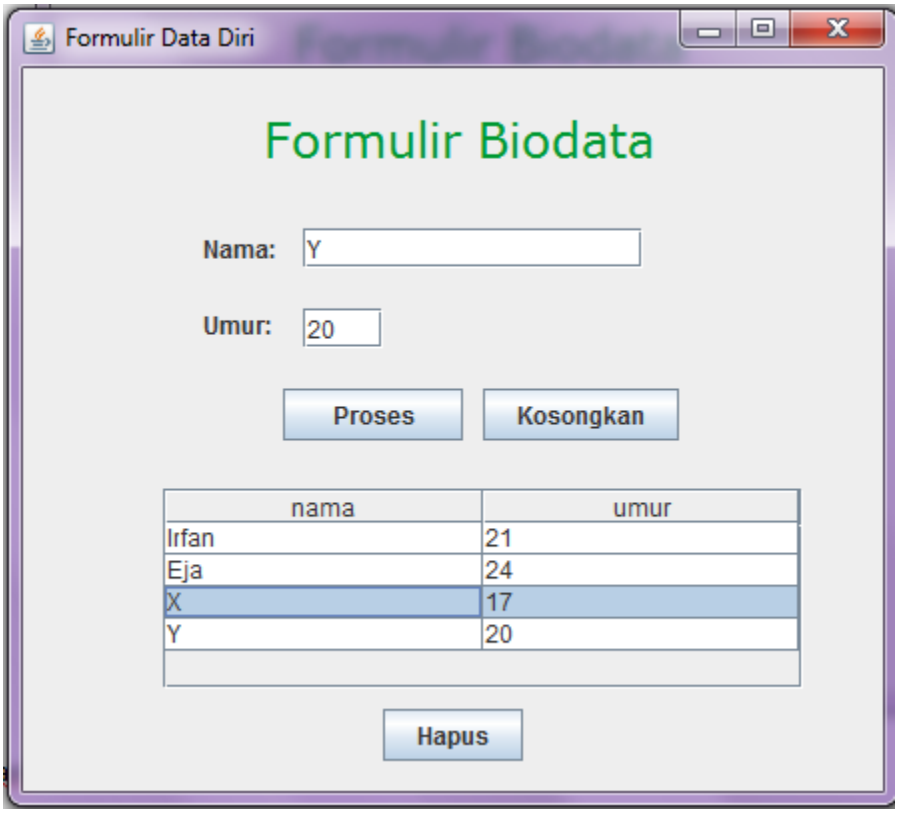
1.7.2 Penyelesaian

Drag-and-drop komponen swing dari palette ke frame, letakkan ke posisi yang diinginkan. Untuk mengubah tulisan pada label, textfield, radio button, check box dan button, bisa dilihat pada bagian "properties" di bagian "text". Untuk mengubah pilihan dari combo box bisa dilihat pada bagian "properties" di bagian "model". Tekan eclipse button (tombol (…)), dan ubah value "item 1" dst seperti yang diinginkan.

Untuk menjadikan 1 radio button ke dalam 1 group, drag-and-drop sebuah komponen "Button Group" ke frame. Lalu pilih salah satu radio button, di "properties" cari "buttonGroup", pilih nama buttonGroup yang telah ditambahkan. Lakukan hal yang sama dengan radio button yang lain.

1.8 LATIHAN

Buatlah sebuah GUI java sebagai berikut:



Keterangan:

- Masukkan komponen dalam panel; Jangan memasukkan langsung komponen dalam frame
- Tidak ada aksi saat menekan tombol

Latihan Tambahan:

- ✓ Berikan border pada panel
- ✓ Munculkan window di tengah-tengah layar saat aplikasi dijalankan
- ✓ Tambahkan menu bar pada frame, pada menu bar terdapat menu “Pilihan” dan dalam menu terdapat menu item “Kosongkan”

2 BAB II MATISSE BUILDER 2 (ACTION)

2.1 IDENTITAS

Kajian

Pengenalan Swing Java (Menggunakan Tools--Visual Editor/Matisse Builder)

Topik

1. Penggunaan Matisse Builder
2. Action Listener Komponen Swing

Referensi

1. http://netbeans.org

Kompetensi Utama

1. Mahasiswa memahami konsep pemrograman swing
2. Mahasiswa mampu membuat halaman sederhana menggunakan komponen swing dibantu tool gui builder
3. Mahasiswa mampu menambahkan action pada komponen swing

Lama Kegiatan Praktikum

1. Pertemuan Terbimbing : 1 x 50 menit
2. Kegiatan Mandiri : 1 x 50 menit

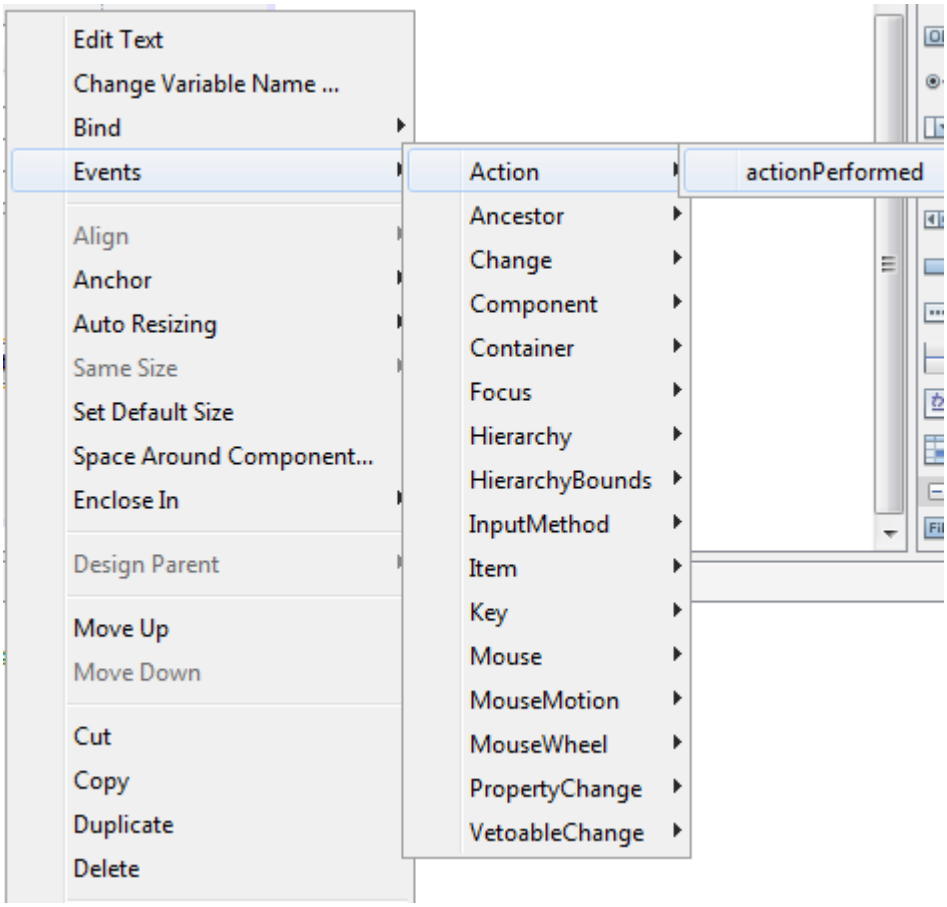
Parameter Penilaian

1. Tugas Pendahuluan
2. Jurnal Mandiri

2.2 EVENT LISTENER

Untuk mendapatkan kesan "responsif" terhadap input user, aplikasi gui yang dibangun menggunakan bahasa java dilengkapi oleh event handler. Event handler didapat dari **interface listener**. Interface listener mempunyai method yang harus dituliskan kembali sebagai method yang harus dijalankan saat mendapat input user.

Dengan menggunakan matisse builder, penambahan event handler menjadi lebih mudah. Tambahkan event dengan meng-klik kanan komponen yang akan ditambahkan event, lalu pilih event apa yang diinginkan (biasanya tipe action dengan method actionPerformed()).



Atau bisa juga dengan meng-klik 2x komponen yang ingin ditambahkan event. Untuk tombol (JButton), event yang terbentuk adalah action → actionPerformed(). Akan muncul bagian source code pada gui seperti berikut. Logika pemrograman dapat dituliskan di method tersebut.

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

2.3 METHOD KOMPONEN SWING

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

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

2.4 PENGGUNAAN DIALOG (JOPTIONPANE)

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

Bentuk:

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

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

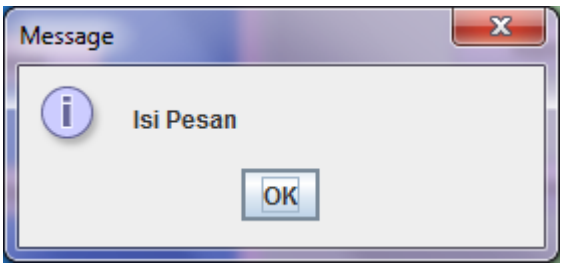
Pada modul praktikum ini hanya dibahas 2 tipe option pane, showConfirmDialog dan showMessageDialog. Setiap option pane dapat dibuat objeknya dengan menggunakan jumlah parameter yang berbeda-beda. Hal ini akan berpengaruh pada tampilan option pane yang dihasilkan.

2.4.1 showMessageDialog

Contoh Kode showMessageDialog dengan 2 parameter:

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

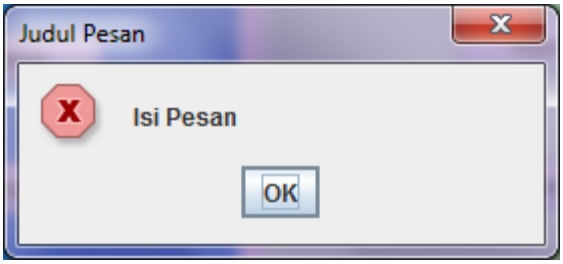
Hasil:



Contoh Kode showMessageDialog dengan 4 parameter:

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

Hasil:



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

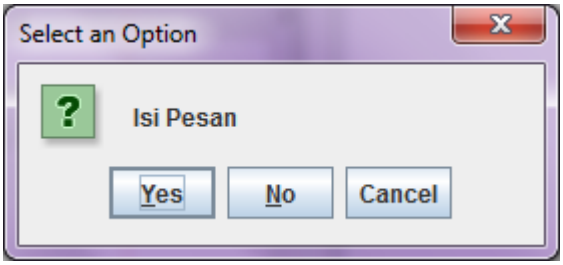
Icon	Code	IDE Value
No icon	JOptionPane.PLAIN_MESSAGE	-1
	JOptionPane.ERROR_MESSAGE	0
	JOptionPane.INFORMATION_MESSAGE	1
	JOptionPane.WARNING_MESSAGE	2
	JOptionPane.QUESTION_MESSAGE	3

2.4.2 showConfirmDialog

Contoh Kode showMessageDialog dengan 2 parameter:

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

Hasil:



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

```
//meminta input yang ditekan sekaligus deklarasi option pane
int hasil = JOptionPane.showConfirmDialog(null, "Isi Pesan");

//membandingkan hasil yang diterima dengan konstanta yang dimiliki option pane
if(hasil == JOptionPane.YES_OPTION){
   System.out.println("Yes!");
}else if(hasil == JOptionPane.NO_OPTION){
   System.out.println("No!");
}else if(hasil == JOptionPane.CANCEL_OPTION){
   System.out.println("Cancel!");
}
```

Pembanding pesan terdiri dari:

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

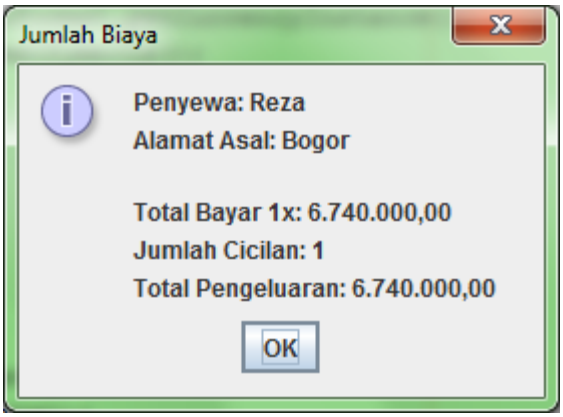
2.5 PRAKTIK

2.5.1 Penginapan Mine Nakahara

Mine Nakahara merupakan pemilik penginapan terkenal di desa konoha. Nakahara ingin memperluas cakupan bisnisnya ke bidang IT. Salah satu langkah nyata yang ia ambil adalah membuat aplikasi untuk menghitung biaya penginapan bagi para penyewa penginapannya.

Nakahara ingin membuat aplikasi tersebut dengan GUI sebagai berikut:

Buatlah aplikasi berdasarkan GUI di atas dan ketika tombol “Hitung Biaya” ditekan, maka muncul pesan informasi sebagai berikut:



Dengan informasi:

Aturan aksi:

- Jika memilih periode “Tahunan” pilihan cicilan berubah menjadi {“-pilihan-”, “3 bulan”,”6 bulan”,”1 tahun”}
- Jika memilih perode “Bulanan” pilihan cicilan berubah menjadi {“-pilihan-”,”mingguan”,”bulanan”}

Aturan Perhitungan:

Aturan	Biaya	Aturan	Biaya
Periode Tahunan	Total: Rp. 6.500.000	Cicilan 3 Bulan	Total: total/4 Jumlah cicilan: 4
		Cicilan 6 Bulan	Total: total/2 Jumlah cicilan: 2
		Cicilan 1 tahun	Jumlah cicilan: 1
Periode Bulanan	Total: Rp. 800.000	Cicilan Mingguan	Total: total/3 Jumlah cicilan: 3
		Cicilan Bulanan	Jumlah cicilan: 1
Dispenser		Fasilitas Dispenser Bulanan	Total: Total + 30.000
		Fasilitas Dispenser Tahunan	Total: Total + 240.000
Televisi		Fasilitas Televisi Bulanan/Tahunan	Total: Total + 272.000
Kipas Angin		Fasilitas Kipas Angin Bulanan	Total: Total + 5000
		Fasilitas Kipas Angin Tahunan	Total: Total + 15000

2.5.2 Penyelesaian Contoh Kasus

Langkah pertama, munculkan pilihan pada combo box melalui pilihan radio button. Caranya, klik 2x di radio button pertama (pilihan tahunan), dan window akan memunculkan source dari class tersebut dan membuat method sebagai berikut. Tapi sebelumnya, ubah nama radio button menjadi "radioTahunan" (klik kanan, pilih "change variable name"):

```
private void radioTahunanActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}
```

Method tersebut berarti menambahkan suatu "aksi" jika radio button dipilih. Dalam blok yang sama dengan baris "// TODO add your handling code here:", tambahkan logika untuk menambahkan komponen combo box (sebelumnya, ubah nama combo box menjadi "comboCicilan").

```
private void radioTahunanActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    comboCicilan.removeAllItems();
    comboCicilan.addItem("-Pilihan-");
    comboCicilan.addItem("3 Bulan");
    comboCicilan.addItem("6 Bulan");
    comboCicilan.addItem("1 Tahun");
}
```

Cara lain yang bisa dilakukan untuk menambahkan method tersebut adalah dengan klik kanan di komponen radio button, pilih "Events → Action → actionPerformed".

Lakukan hal yang sama dengan radio button yang lain (radioBulanan):

```
private void jRadioButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    comboCicilan.removeAllItems();
    comboCicilan.addItem("-Pilihan-");
    comboCicilan.addItem("Mingguan");
    comboCicilan.addItem("Bulanan");
}
```

Setelah itu, lakukan perhitungan saat tombol ditekan. Klik 2x pada tombol (ganti variabel name menjadi "tombolHitung") untuk menambahkan method actionPerformed:

```
private void tombolHitungActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}
```

Tambahkan logika perhitungan biaya dan cara menampilkannya sebagai berikut:

```
    //deklarasi variabel untuk menyimpan total biaya
    //dan fasilitas yang harus dibayar
    double total = 0, fasilitas = 0;

    //deklarasi variabel untuk menyimpan total cicilan
    //dan status jika user tidak memilih pilihan
    int jumCicilan = 0;
    boolean salah = false;
```

```
//ambil harga dasar bulanan atau tahunan
if (radioTahunan.isSelected()) {
   total = 6500000;
   //pilihan cicilan
   if (comboCicilan.getSelectedIndex() == 1) {
      jumCicilan = 4;
   } else if (comboCicilan.getSelectedIndex() == 2) {
      jumCicilan = 2;
   } else if (comboCicilan.getSelectedIndex() == 3) {
      jumCicilan = 1;
   } else {
      salah = true;
   }

   //pilihan fasilitas
   if (cekDispenser.isSelected()) {
      fasilitas = fasilitas + 240000;
   }
   if (cekTv.isSelected()) {
      fasilitas = fasilitas + 272000;
   }
   if (cekKipas.isSelected()) {
      fasilitas = fasilitas + 15000;
   }

} else if (radioBulanan.isSelected()) {
   total = 800000;
   //pilihan cicilan
   if (comboCicilan.getSelectedIndex() == 1) {
      jumCicilan = 3;
   } else if (comboCicilan.getSelectedIndex() == 2) {
      jumCicilan = 1;
   } else {
      salah = true;
   }

   //pilihan fasilitas
   if (cekDispenser.isSelected()) {
      fasilitas = fasilitas + 30000;
   }
   if (cekTv.isSelected()) {
      fasilitas = fasilitas + 272000;
   }
   if (cekKipas.isSelected()) {
      fasilitas = fasilitas + 5000;
```

```
        }

    } else {
        salah = true;
    }
    //hitung total biaya per-1x bayar ditambah fasilitas
    total = total + fasilitas;
    total = total / jumCicilan;

    if (teksNama.getText().isEmpty() || teksAlamat.getText().isEmpty()) {
        salah = true;
    }
```

Cek input sudah benar atau belum, jika sudah tampilkan total harga, jika belum tampilkan pesan peringatan untuk mengecek masukan yang ada

```
    //cek apakah radio button sudah dipilih
    //cek apakah combo box sudah dipilih
    if (salah) {
        JOptionPane.showMessageDialog(this, "Periksa Nama & Alamat,\n"+
                "Pilihan Bulanan/Tahunan,\n"
                + "dan Jumlah Cicilan", "Status", JOptionPane.WARNING_MESSAGE);
    } else {
        //format tampilan uang
        DecimalFormat df = (DecimalFormat) DecimalFormat.getCurrencyInstance();
        DecimalFormatSymbols dfs = new DecimalFormatSymbols();
        dfs.setCurrencySymbol("");
        dfs.setMonetaryDecimalSeparator(',');
        dfs.setGroupingSeparator('.');
        df.setDecimalFormatSymbols(dfs);

        //format tampilan dialog
        String pesan = "";
        pesan = "Penyewa: " + teksNama.getText();
        pesan += "\nAlamat Asal: " + teksAlamat.getText() + "\n\n";
        pesan += "Total Bayar 1x: " + df.format(total);
        pesan += "\nJumlah Cicilan: " + jumCicilan;
        pesan += "\nTotal Pengeluaran: " + df.format(total * jumCicilan);
        JOptionPane.showMessageDialog(this, pesan, "Jumlah Biaya",
                JOptionPane.INFORMATION_MESSAGE);
```

2.5.3 Solusi Lengkap

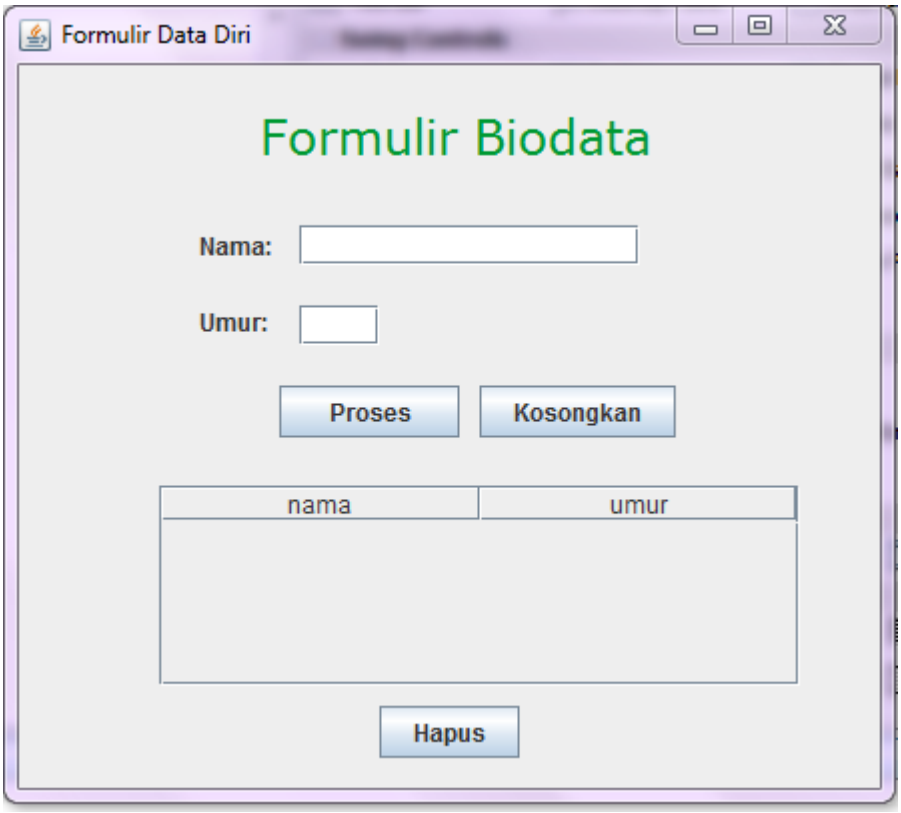
```
private void tombolHitungActionPerformed(java.awt.event.ActionEvent evt) {
    //deklarasi variabel untuk menyimpan total biaya
    //dan fasilitas yang harus dibayar
    double total = 0, fasilitas = 0;

    //deklarasi variabel untuk menyimpan total cicilan
    //dan status jika user tidak memilih pilihan
    int jumCicilan = 0;
    boolean salah = false;

    //ambil harga dasar bulanan atau tahunan
    if (radioTahunan.isSelected()) {
        total = 6500000;
        //pilihan cicilan
        if (comboCicilan.getSelectedIndex() == 1) {
            jumCicilan = 4;
        } else if (comboCicilan.getSelectedIndex() == 2) {
            jumCicilan = 2;
        } else if (comboCicilan.getSelectedIndex() == 3) {
            jumCicilan = 1;
        } else {
            salah = true;
        }
        //pilihan fasilitas
        if (cekDispenser.isSelected()) {
            fasilitas = fasilitas + 240000;
        }
        if (cekTv.isSelected()) {
            fasilitas = fasilitas + 272000;
        }
        if (cekKipas.isSelected()) {
            fasilitas = fasilitas + 15000;
        }
    } else if (radioBulanan.isSelected()) {
        total = 800000;
        //pilihan cicilan
        if (comboCicilan.getSelectedIndex() == 1) {
            jumCicilan = 3;
        } else if (comboCicilan.getSelectedIndex() == 2) {
            jumCicilan = 1;
        } else {
            salah = true;
        }
```

```
    //pilihan fasilitas
    if (cekDispenser.isSelected()) {
        fasilitas = fasilitas + 30000;
    }
    if (cekTv.isSelected()) {
        fasilitas = fasilitas + 272000;
    }
    if (cekKipas.isSelected()) {
        fasilitas = fasilitas + 5000;
    }

} else {
    salah = true;
}
//hitung total biaya per-1x bayar ditambah fasilitas
total = total + fasilitas;
total = total / jumCicilan;

if (teksNama.getText().isEmpty() || teksAlamat.getText().isEmpty()) {
    salah = true;
}

//cek apakah radio button sudah dipilih
//cek apakah combo box sudah dipilih
if (salah) {
    JOptionPane.showMessageDialog(this, "Periksa Nama & Alamat,\n"+
            "Pilihan Bulanan/Tahunan,\n"
            + "dan Jumlah Cicilan", "Status", JOptionPane.WARNING_MESSAGE);
} else {
    //format tampilan uang
    DecimalFormat df = (DecimalFormat) DecimalFormat.getCurrencyInstance();
    DecimalFormatSymbols dfs = new DecimalFormatSymbols();
    dfs.setCurrencySymbol("");
    dfs.setMonetaryDecimalSeparator(',');
    dfs.setGroupingSeparator('.');
    df.setDecimalFormatSymbols(dfs);
```

```
        //format tampilan dialog
        String pesan = "";
        pesan = "Penyewa: " + teksNama.getText();
        pesan += "\nAlamat Asal: " + teksAlamat.getText() + "\n\n";
        pesan += "Total Bayar 1x: " + df.format(total);
        pesan += "\nJumlah Cicilan: " + jumCicilan;
        pesan += "\nTotal Pengeluaran: " + df.format(total * jumCicilan);
        JOptionPane.showMessageDialog(this, pesan, "Jumlah Biaya",
                JOptionPane.INFORMATION_MESSAGE);
    }
}
```

2.6 LATIHAN

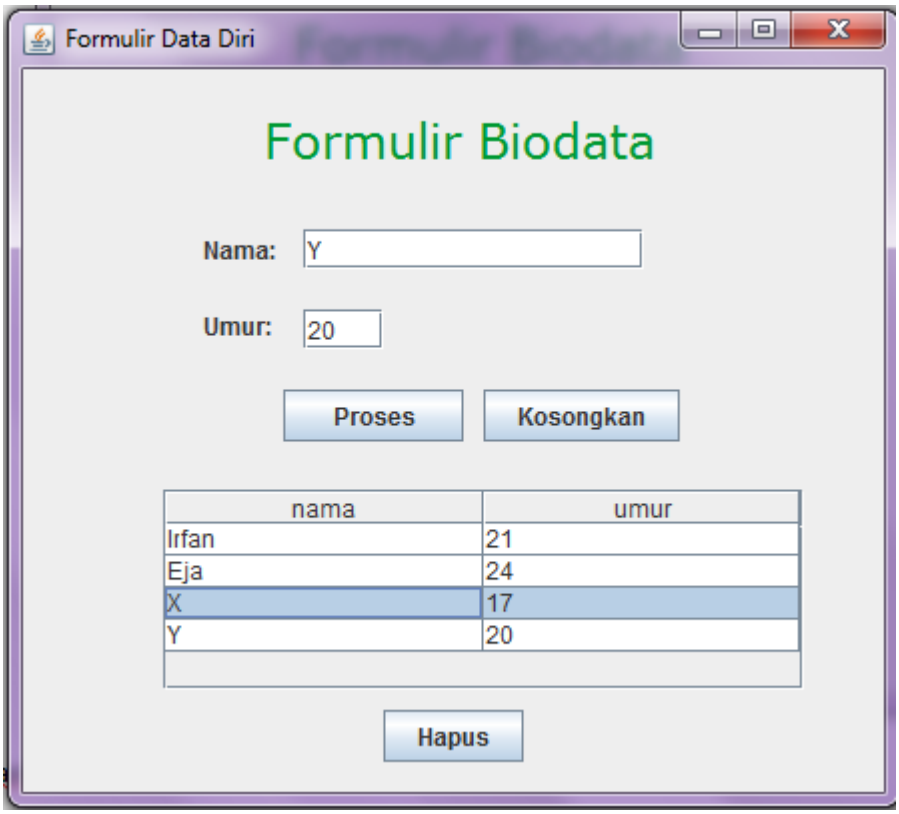
Buatlah GUI sebagai berikut menggunakan komponen swing java:



Keterangan:

- Masukkan komponen dalam panel; Jangan memasukkan langsung komponen dalam frame
- Terdapat 2 input
- Frame muncul di tengah screen
- Setelah menekan “OK” data input muncul di tabel
- Tabel dapat ditambah dan dihapus tapi tidak dapat diubah
- Jika ada yang salah satu textfield yang kosong, maka program menampilkan pesan kesalahan

Hasil Akhir:



Latihan Tambahan:

- ✓ Gantilah “Umur”, menjadi combo box yang berisi tahun lahir. Tapi nantinya tetap tampilkan umur di table. Beri Option pane untuk masing-masing tombol. Contohnya, setelah menekan tombol “proses” muncul pesan yang menyatakan bahwa data sudah dimasukkan.

3 BAB III MATISSE BUILDER 3 (LAYOUT)

3.1 IDENTITAS

Kajian

Pengenalan Swing Java (Menggunakan Tools--Visual Editor/Matisse Builder)**.**

Topik

1. Penggunaan Matisse Builder
2. Layout: Border, Grid Layout, GridBag Layout

Referensi

1. http://netbeans.org

Kompetensi Utama

1. Mahasiswa memahami konsep pemrograman swing
2. Mahasiswa mampu membuat halaman sederhana menggunakan komponen swing dibantu tool gui builder
3. Mahasiswa mampu membuat halaman sederhana menggunakan salah satu layout dari komponen swing (Border, Grid, GridBag) atau kombinasinya

Lama Kegiatan Praktikum

1. Pertemuan Terbimbing : 1 x 50 menit
2. Kegiatan Mandiri : 1 x 50 menit

Parameter Penilaian

1. Tugas Pendahuluan
2. Jurnal Mandiri

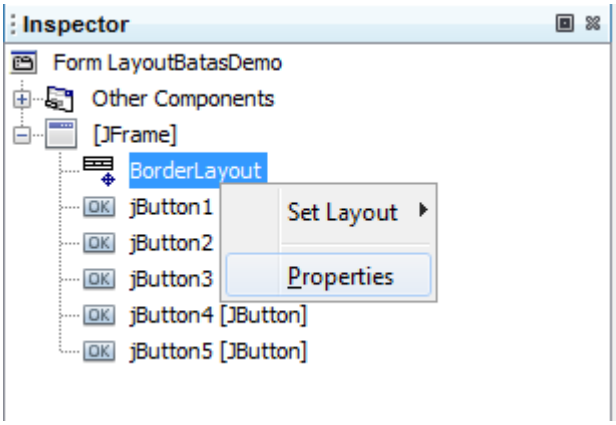
3.2 PENGENALAN LAYOUT

Layout merupakan pengaturan peletakan komponen swing pada container-nya (container bisa panel atau frame atau apapun). Cara mengatur layout pada gui builder yang disediakan netbeans adalah dengan klik kanan di bagian kosong pada container, lalu pilih "set Layout". Ada 8 layout yang dapat dipilih ditambah 1 layout default dari gui builder. Tapi pada modul ini hanya dibahas 3 modul saja, Border Layout, Grid Layout dan GridBag Layout.

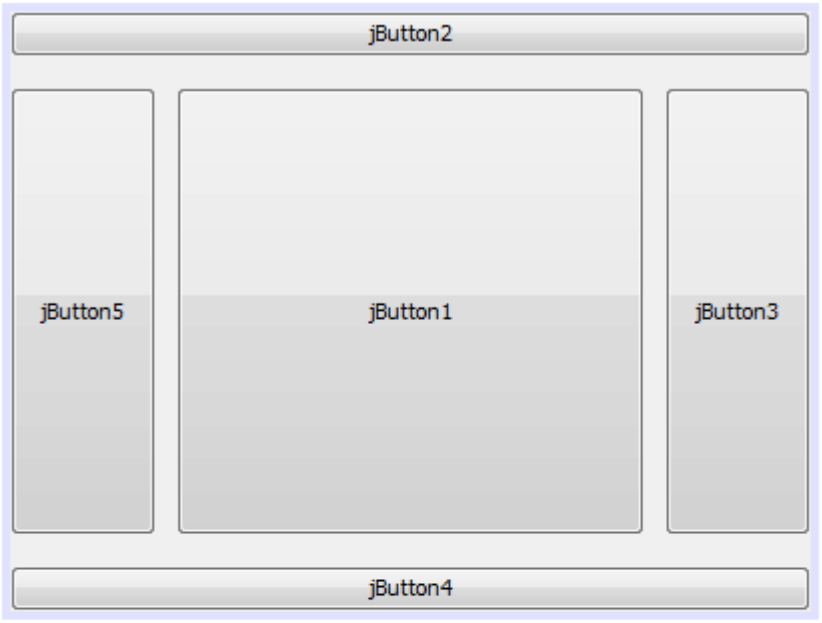
3.3 BORDER LAYOUT

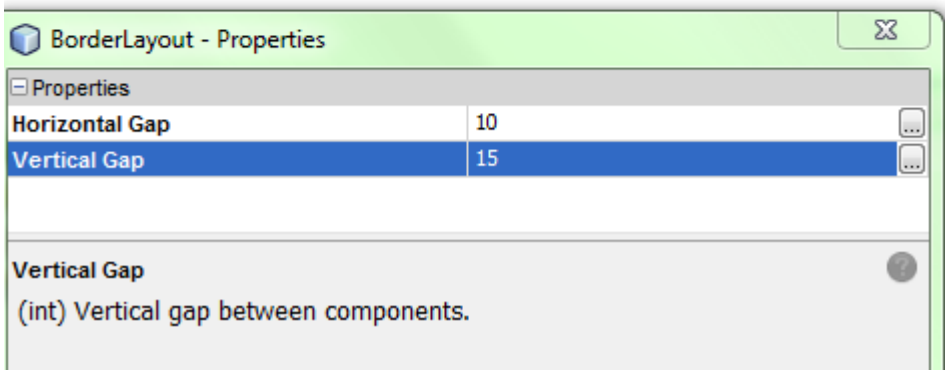
Merupakan layout yang memungkinkan komponen untuk memenuhi semua frame. Ukuran frame akan mempengaruhi ukuran komponen pada frame tersebut. Jika dilakukan resize, maka komponen di tengah yang akan mengalami pembesaran maksimal (secara horizontal dan vertikal), sedangkan komponen di bagian tepi frame hanya mengalami pembesaran secara horizontal atau vertical saja.

Peletakan komponen pada border layout hanya dibatasi di 5 area: PAGE_START, PAGE_END, LINE_START, LINE_END, CENTER (lihat modul 9). Untuk mengubah properties layout, bisa dilihat di window "inspector" pada bagian layout, klik kanan → properties. Hal yang bisa diatur dari Border Layout adalah jarak antar komponen secara vertical dan horizontal.



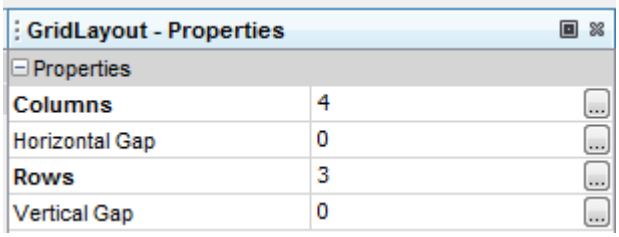
Pengubahan peletakan komponen jika terdapat kesalahan setelah laying out, bisa dilakukan di window properties komponen, kategori layout.





3.4 GRID LAYOUT

Grid layout akan menempatkan komponen ke dalam cell yang terkoordinat pada n kolom, dan m baris. 1 komponen menempati 1 cell dengan ukurang yang seragam. Pada properties dapat diatur jumlah baris dan kolom maksimal yang bisa ditempati oleh komponen. Selain itu dapat diatur jarak vertical dan horizontal antar komponen.

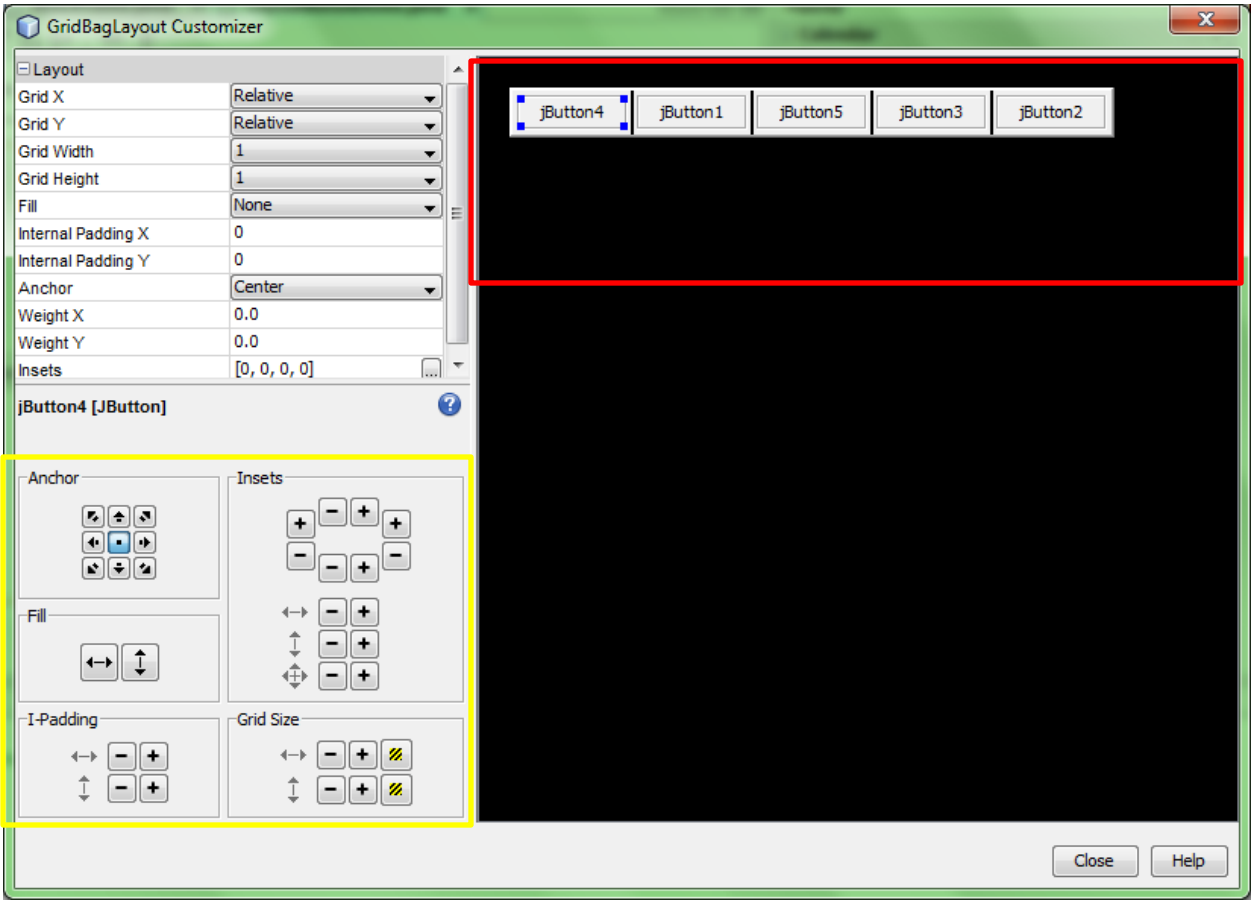


Pengaturan jumlah rows akan lebih dominan dibandingkan jumlah kolom. Dengan pengaturan seperti di atas, menggunakan 5 komponen button, maka letak komponen yang dihasilkan adalah sebagai berikut:

Pengubahan letak komponen setelah set layout bisa dilakukan dengan melakukan drag-and-drop ke posisi yang diinginkan.

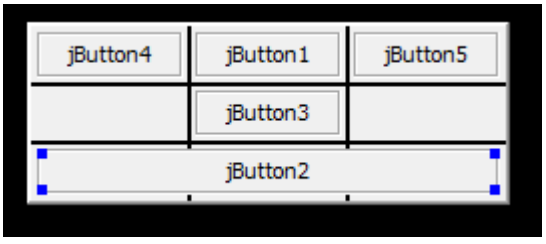
3.5 GRIDBAG LAYOUT

GridBag, sama seperti halnya Grid Layout, tapi dengan menggunakan GridBag Layout, 1 komponen bisa mengambil space penempatan lebih dari 1 cell. Tidak seragam seperti Grid Layout. Pengaturan peletakan komponen dapat dilakukan dengan klik kanan pada container yang dikenai layout, dan pilih “Customize Layout”. Dengan window yang muncul, letak komponen dapat diatur sesuai keinginan.



Kotak merah memperlihatkan peletakan komponen di frame, dan kotak kuning memperlihatkan control komponen terhadap layout grid bag (padding, inset dll). Peletakan komponen dapat diubah menjadi n baris dan m kolom dengan menarik komponen yang ada.

Contoh pengaturan:



3.6 PRAKTIK

3.6.1 Penginapan Mine Nakahara

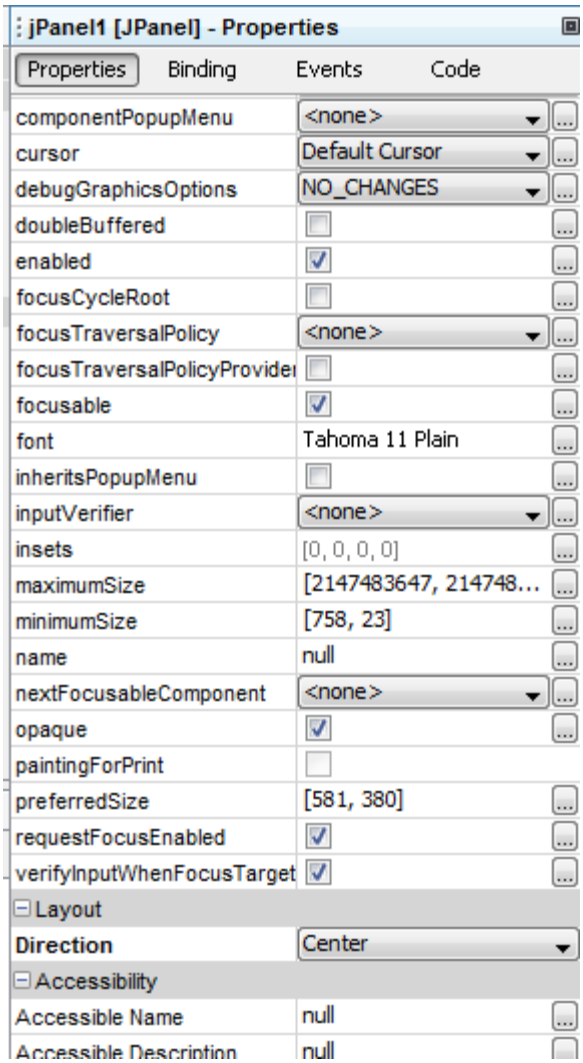
Mine Nakahara merupakan pemilik penginapan terkenal di desa konoha. Nakahara ingin memperluas cakupan bisnisnya ke bidang IT. Salah satu langkah nyata yang ia ambil adalah membuat aplikasi untuk menghitung biaya penginapan bagi para penyewa penginapannya.

Nakahara ingin membuat aplikasi tersebut dengan GUI sebagai berikut:

Buatlah aplikasi sesuai yang Nakahara inginkkan, dan aturlah komponen GUI tersebut dengan menggunakan Border Layout (untuk frame) + GridBag Layout (untuk container pane).

3.6.2 Penyelesaian Contoh Kasus

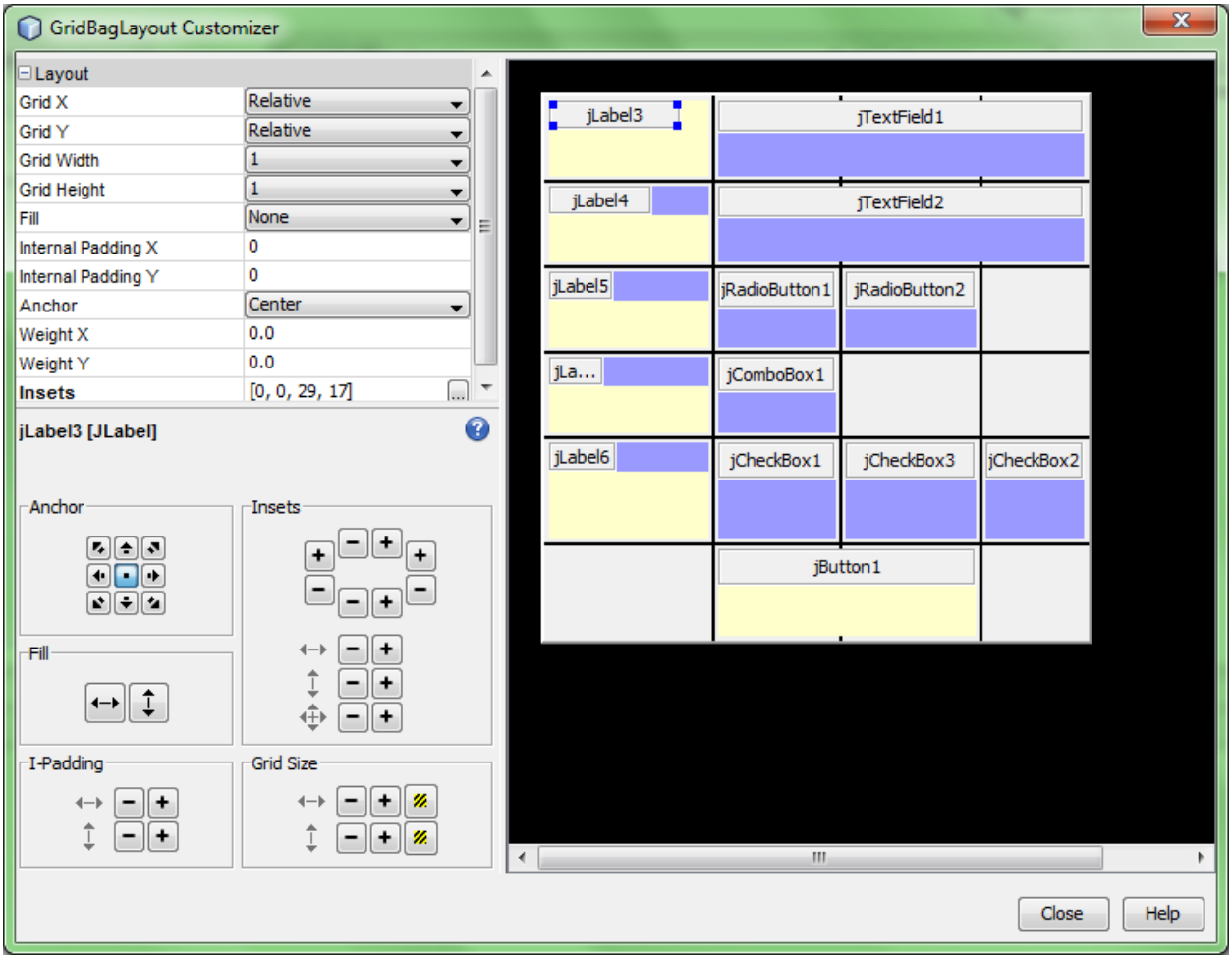
Atur layout frame menjadi border. Tambahkan panel. Di bagian properties, pastikan “Direction” diisi “Center”.



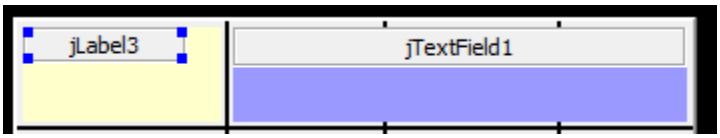
Panel ini akan diisi dengan komponen selain label “Aplikasi Perhitungan Biaya Kamar”. Untuk label tersebut, tambahkan label, ganti “text” di bagian “properties”, dan pastikan memilih “First” atau “North” untuk”Direction”-nya.

Set panel tempat peletakan komponen menjadi “GridBag”, masukkan semua komponen (peletakan komponen berada dalam 1 garis horizontal). Ubah peletakan komponen dengan klik kanan panel tersebut, pilih “Customize Layout…”

Set dengan layout sebagai berikut (kotak kuning) dengan melakukan penarikan terhadap masing-masing komponen.

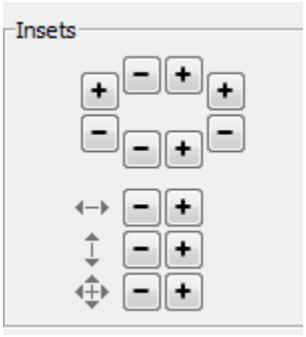


Untuk mengatur jarak bisa diatur dengan control yang ada di panel kiri bawah. Untuk mengatur label agar berada di pojok kiri atas dan memberi jarak vertical dan horizontal pada komponen tetangganya, dapat dilakukan hal berikut:

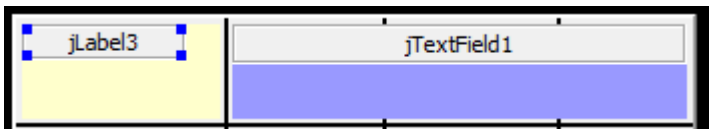


Pastikan anchor di sudut kiri atas:

Jarak horizontal terhadap komponen tetangga di bagian kanan, didapat dengan mengatur inset pada komponen di sebelah kanan. Tekan "+" untuk memperlebar jarak, tekan "-" untuk mengecilkan jaraknya.



Agar satu komponen dapat mengisi lebih dari 1 grid (seperti textfield dan button), atur gridsize sesuai yang diinginkan.



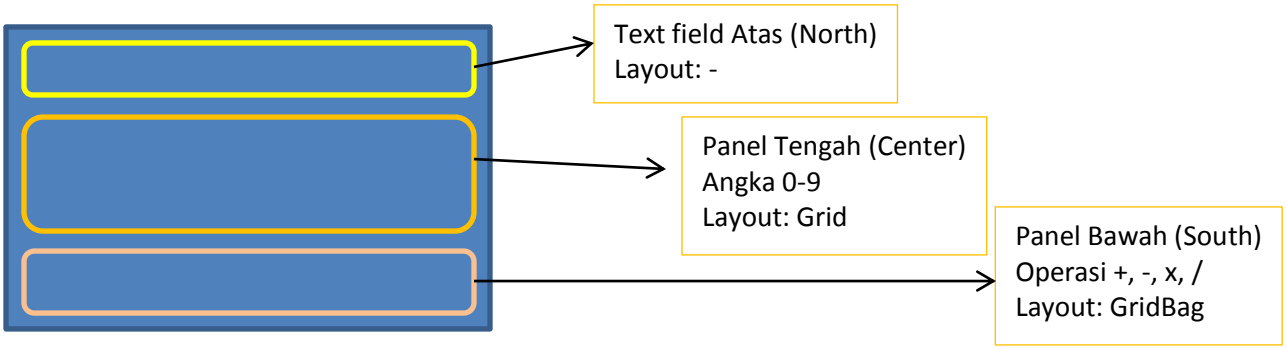
Dengan menekan 2x Gridsize horizontal, textfield mengisi 3 kolom/cell.

Modifikasi control dan peletakan komponen pada “customize layout” untuk mendapatkan hasil sebagai berikut saat dijalankan:

3.7 LATIHAN

Buatlah sebuah kalkulator sebagai berikut menggunakan Border Layout, Grid Layout dan GridBag Layout. Lengkapi kalkulator tersebut dengan action-nya.

Ilustrasi:



4 BAB IV MATISSE BUILDER 4 (LAYOUT 2—WINDOWING)

4.1 IDENTITAS

Kajian

Pengenalan Swing Java (Menggunakan Tools--Visual Editor/Matisse Builder).

Topik

1. Penggunaan Matisse Builder
2. Penggunaan JInternalFrame, Layout Tabbed Pane

Referensi

1. https://netbeans.org

Kompetensi Utama

1. Mahasiswa memahami konsep pemrograman swing
2. Mahasiswa mampu membuat halaman sederhana menggunakan komponen swing dibantu tool gui builder
3. Mampu menggunakan internal frame
4. Mampu menggunakan tabbed pane

Lama Kegiatan Praktikum

1. Kegiatan Mandiri : 1 x 100 menit

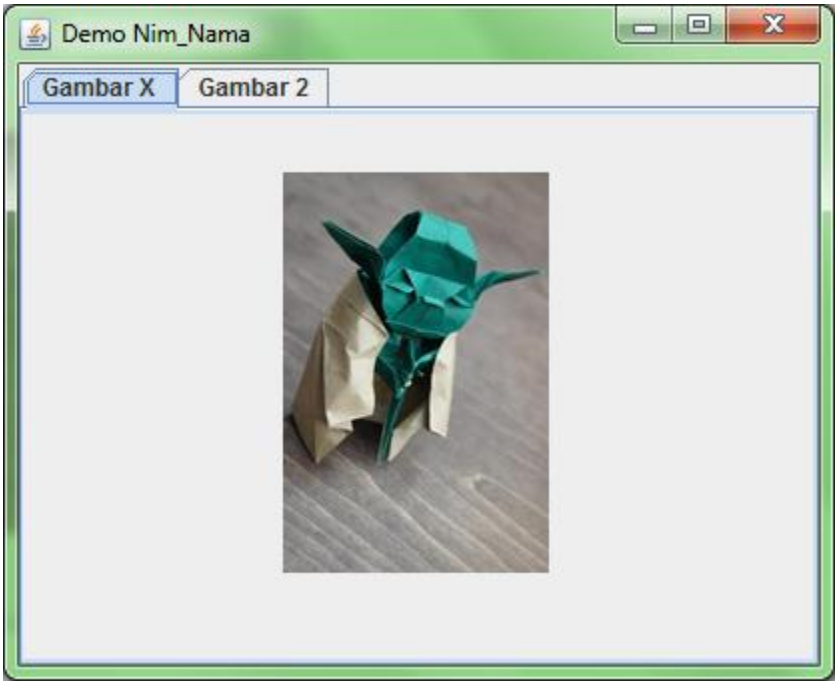
Parameter Penilaian

1. Jurnal Mandiri

4.2 PRAKTIK

4.2.1 Penggunaan JTabbedPane

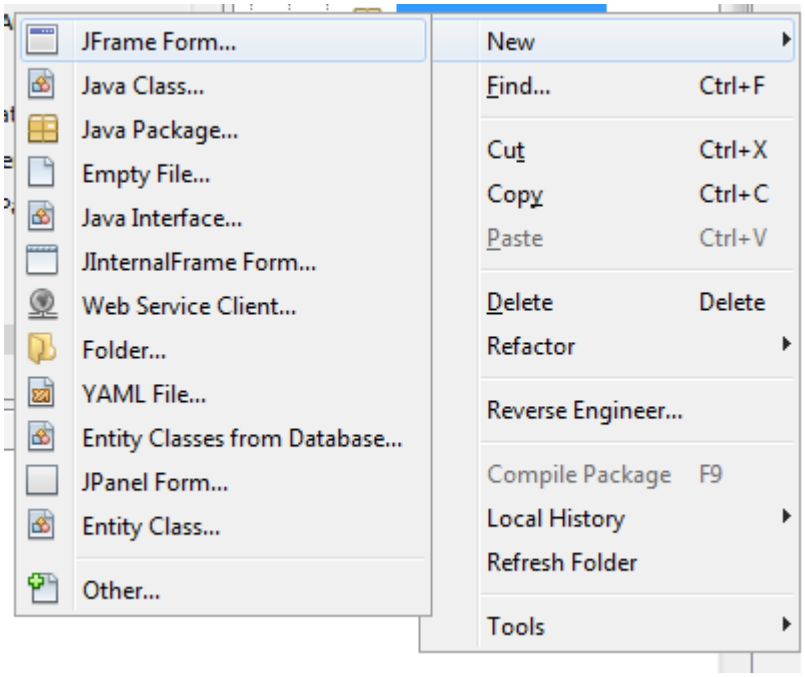
Hasil akhir yang diharapkan:



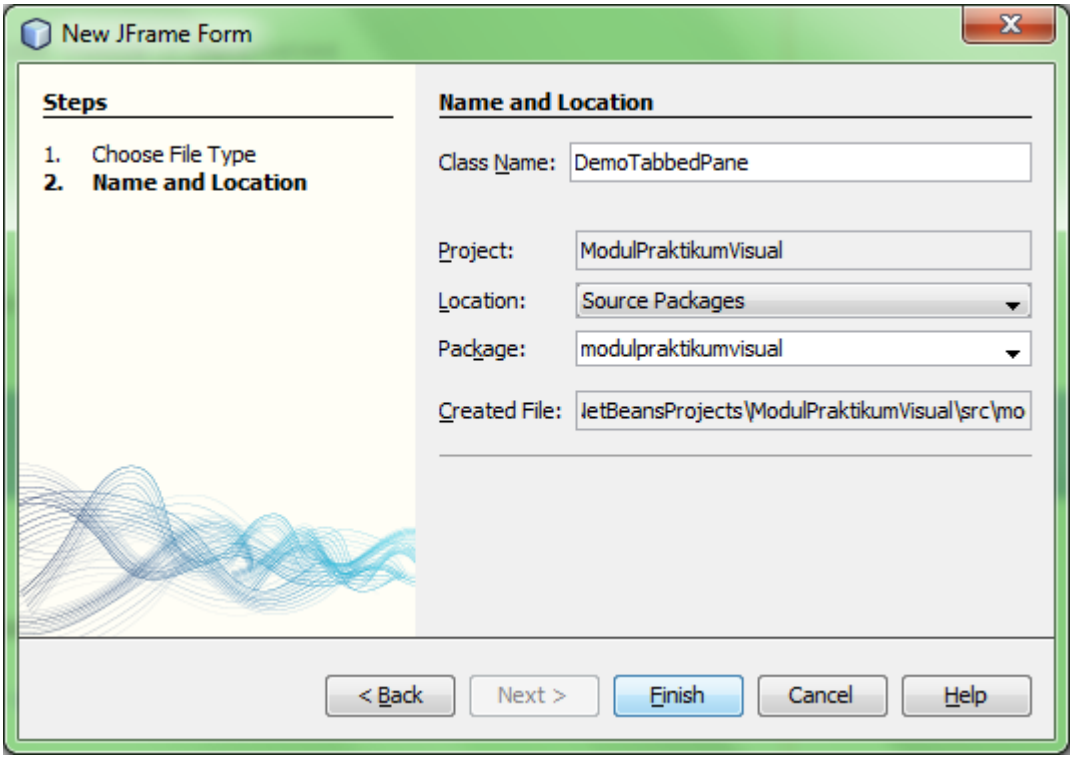
Terdapat 2 tabbed pane dengan masing-masing tab menampilkan sebuah gambar.

4.2.2 Solusi

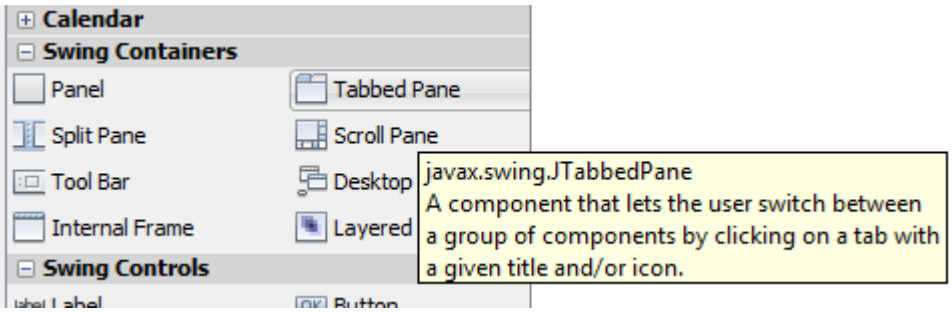
Buatlah sebuah kelas turunan JFrame pada suatu package, dengan memilih New→ JFrame Form…



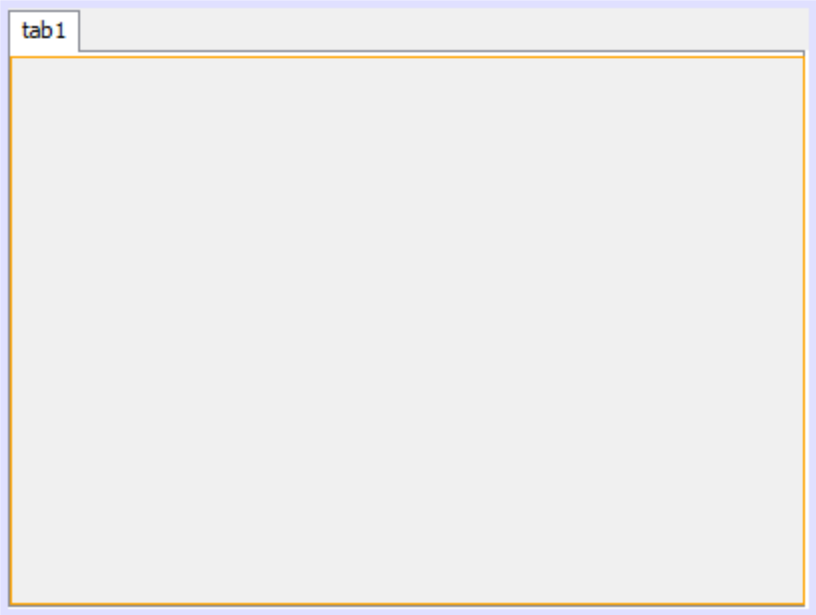
Berikan nama dari Frame tersebut



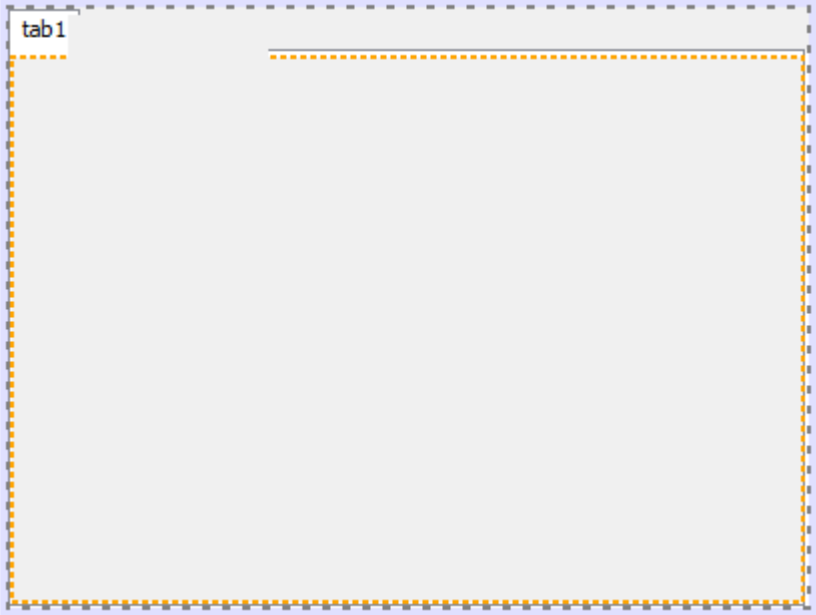
Pada frame kosong, tarik Tabbed Pane untuk memenuhi frame tersebut:

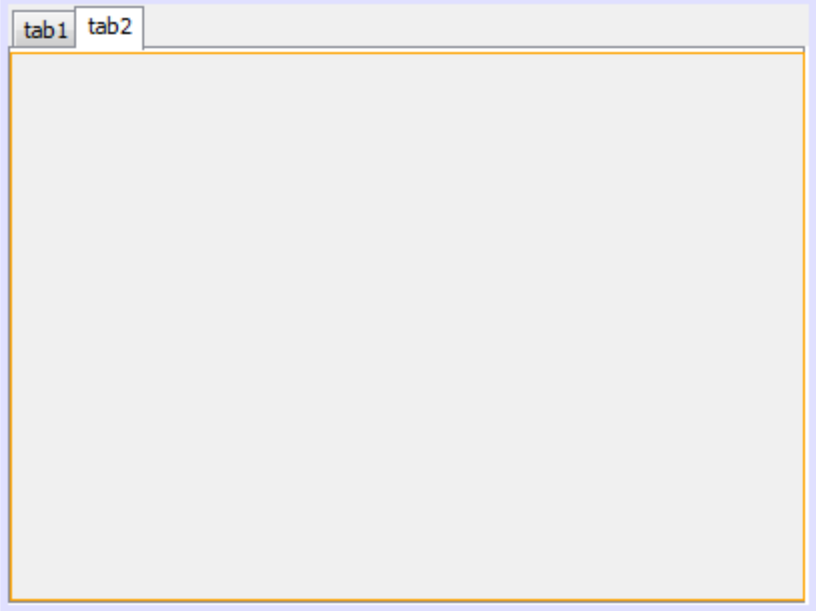


Tarik sebuah panel ke frame, pastikan panel tersebut masuk ke dalam area tabbed pane.

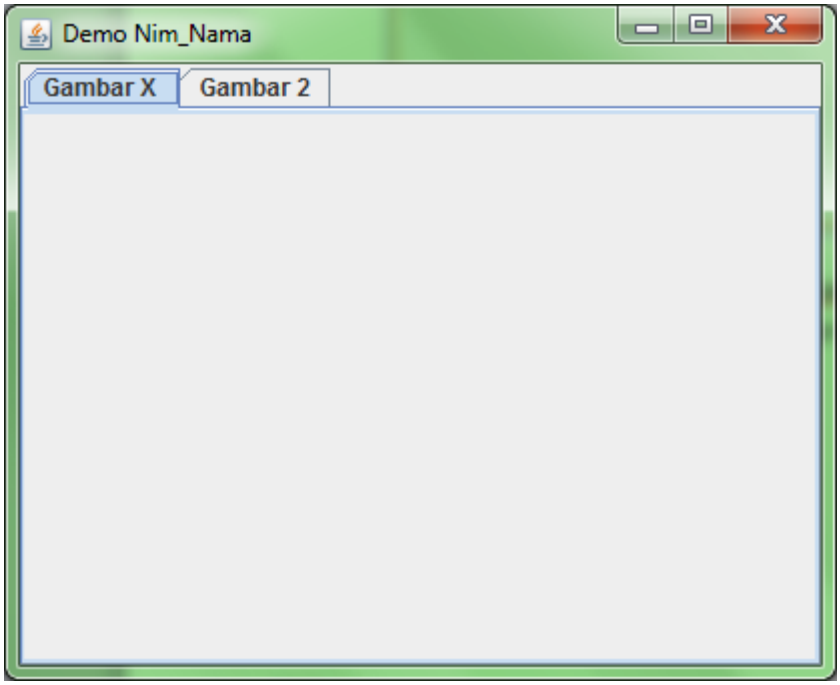


Tarik sebuah panel lagi, dan pastikan panel masuk ke dalam area tabbed pane

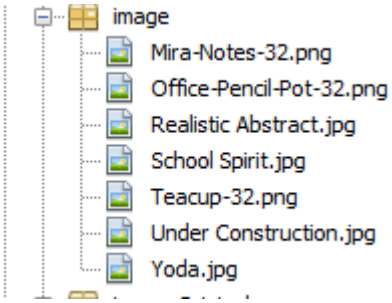




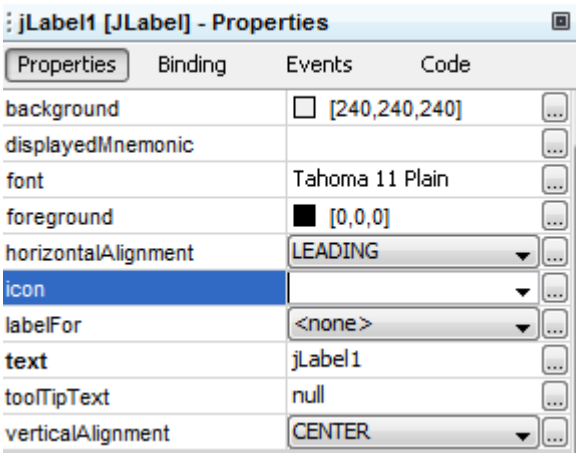
Atur semua property sehingga jika dijalankan menjadi tampil seperti gambar berikut (atur title frame pada properties frame, dan tab title pada properties panel):



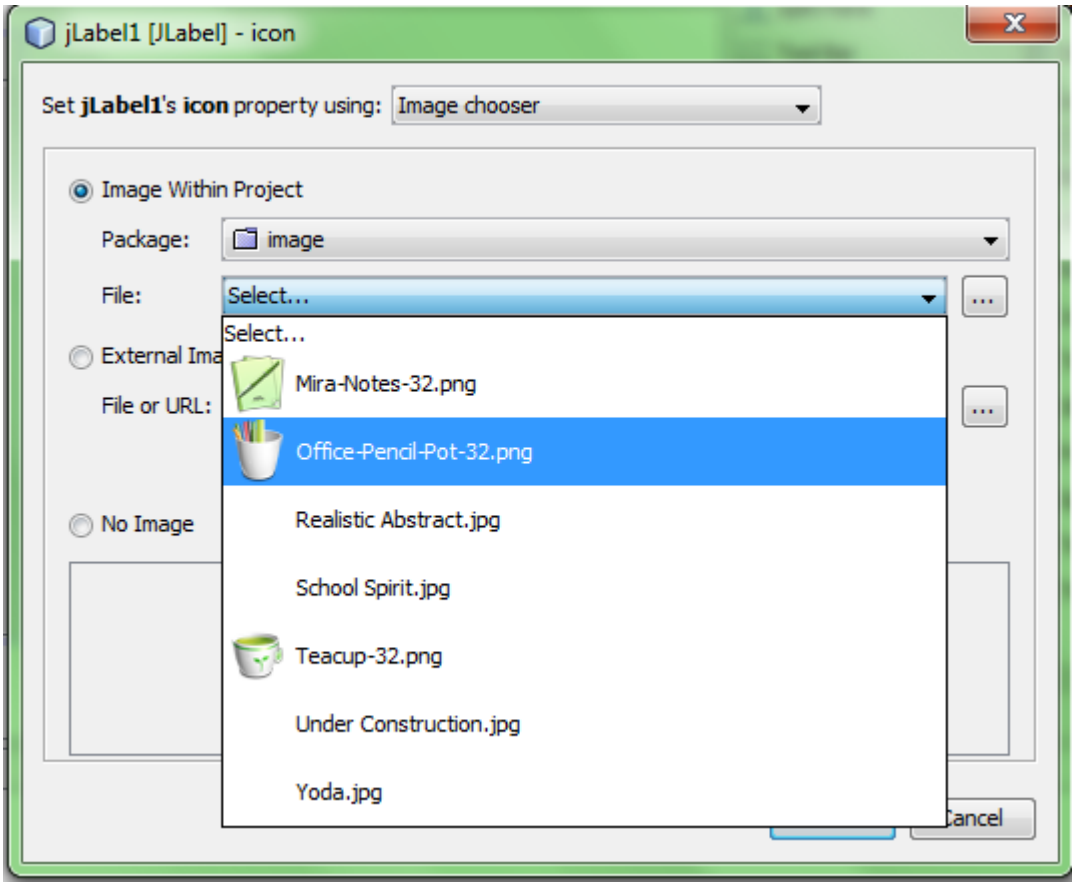
Untuk memasukkan gambar pada frame, bisa menggunakan bantuan dari jlabel. Pastikan package dari gambar yang ingin dimasukkan telah tersedia. Pilih new → package, dan paste image ke package tersebut.



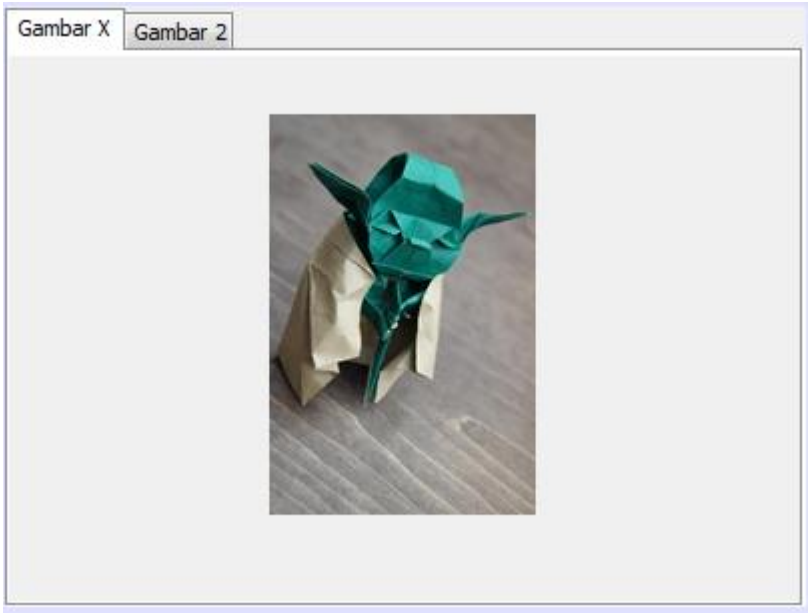
Masukkan sebuah label ke dalam frame, pada bagian properties, cari “icon” dan tekan eclipse button (…) pada icon tersebut.



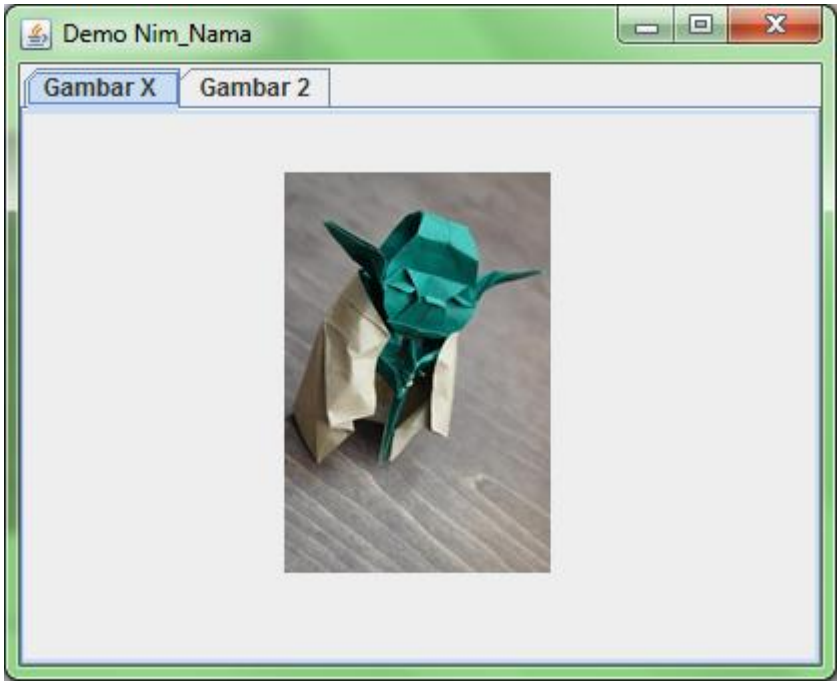
Maka akan tampil menu sebagai berikut, pilih pilihan “Image Within Project” dan pilih package yang mengandung image, serta file yang ingin dimasukkan ke label.



Hapus text dari label, dan Hasil akhir:



Jika dijalankan:



Agar frame, tampil ditengah, tambahkan baris berikut di konstruktor:

```
this.setLocationRelativeTo(null);
```

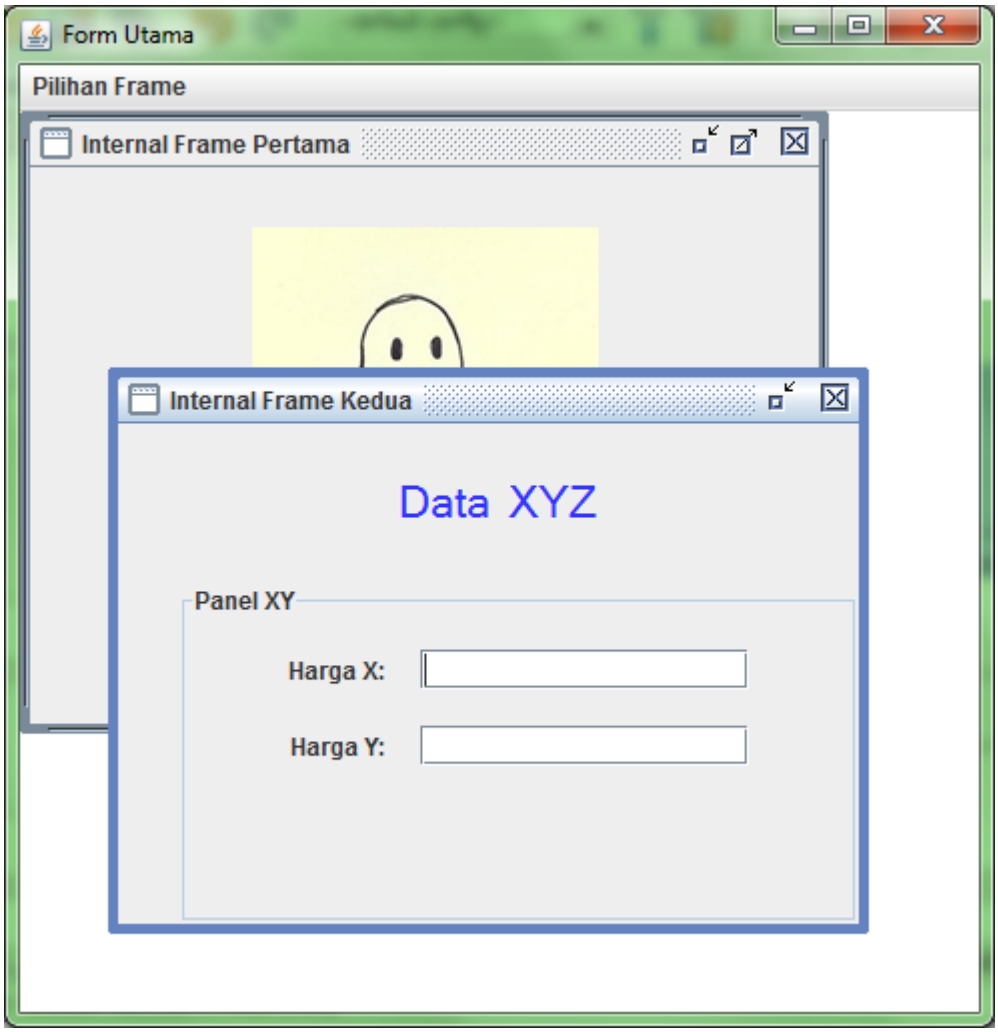
Lengkapnya pada konstruktor tersebut:

```
public DemoTabbedPane() {
    initComponents();
    this.setLocationRelativeTo(null);
}
```

4.2.3 Penggunaan JInternalFrame

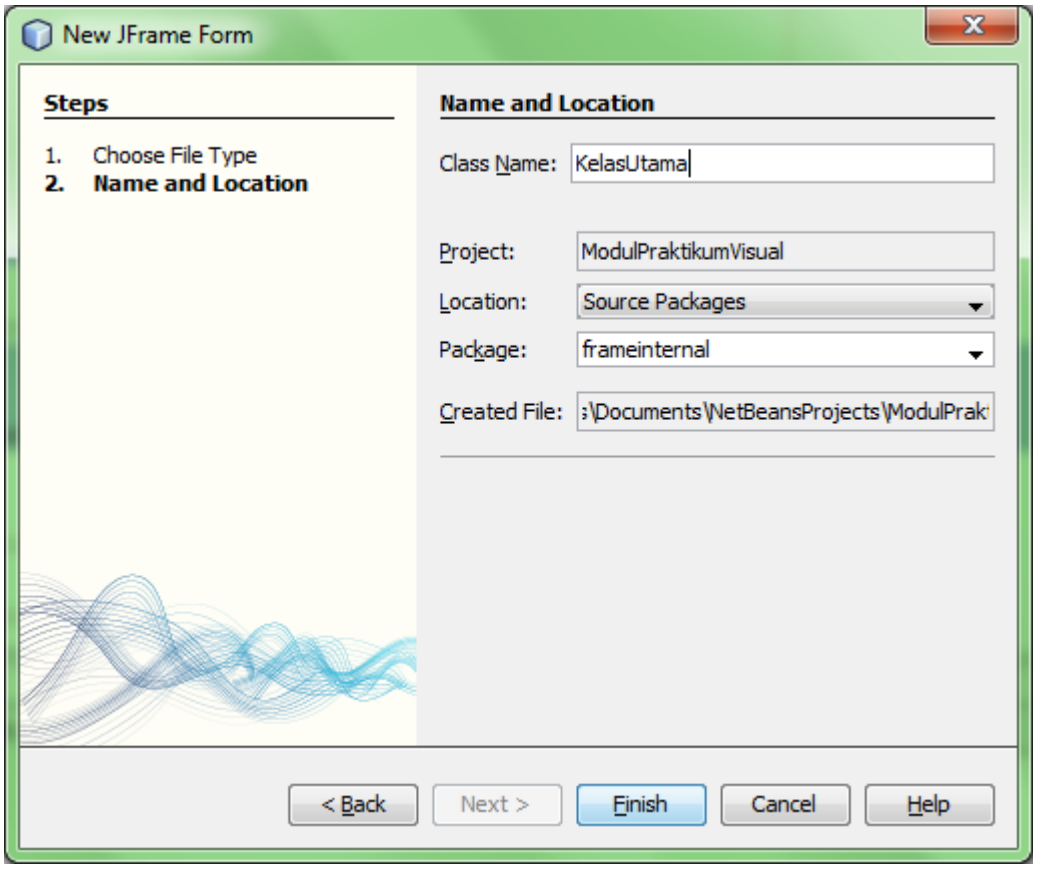
Internal frame digunakan ketika membutuhkan beberapa objek window yang terbuka dalam satu aplikasi.

Hasil akhir yang diharapkan:

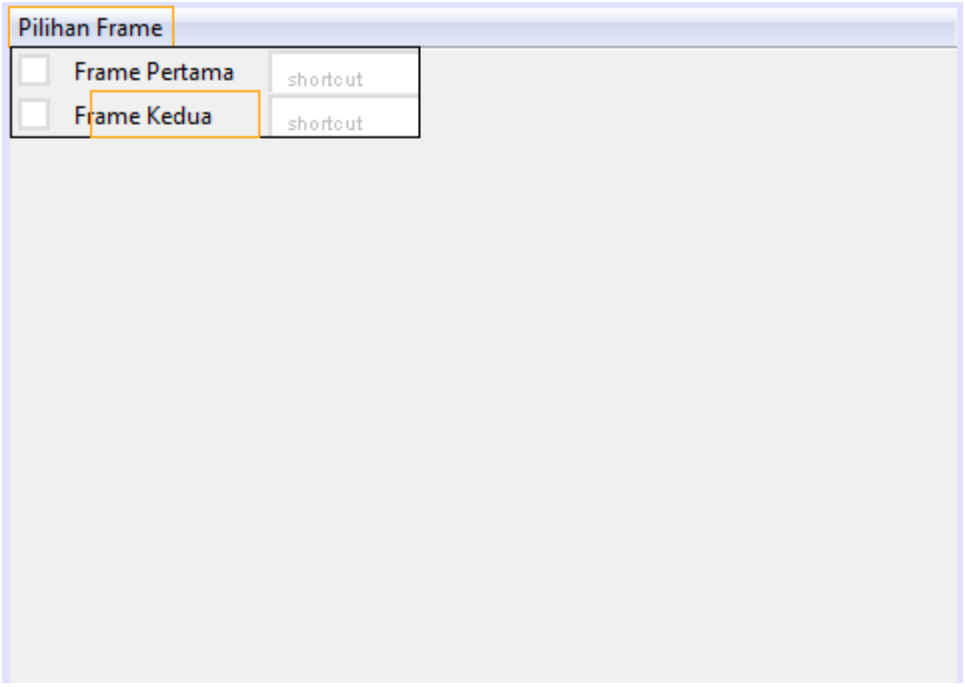


4.2.4 Solusi

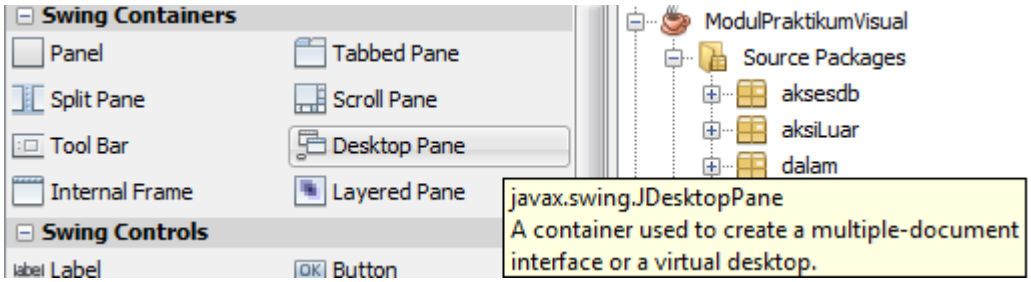
Buat sebuah kelas yang bertindak sebagai frame utama. Kelas ini meng-extends JFrame. Buat kelas seperti membuat Frame Form seperti biasa.



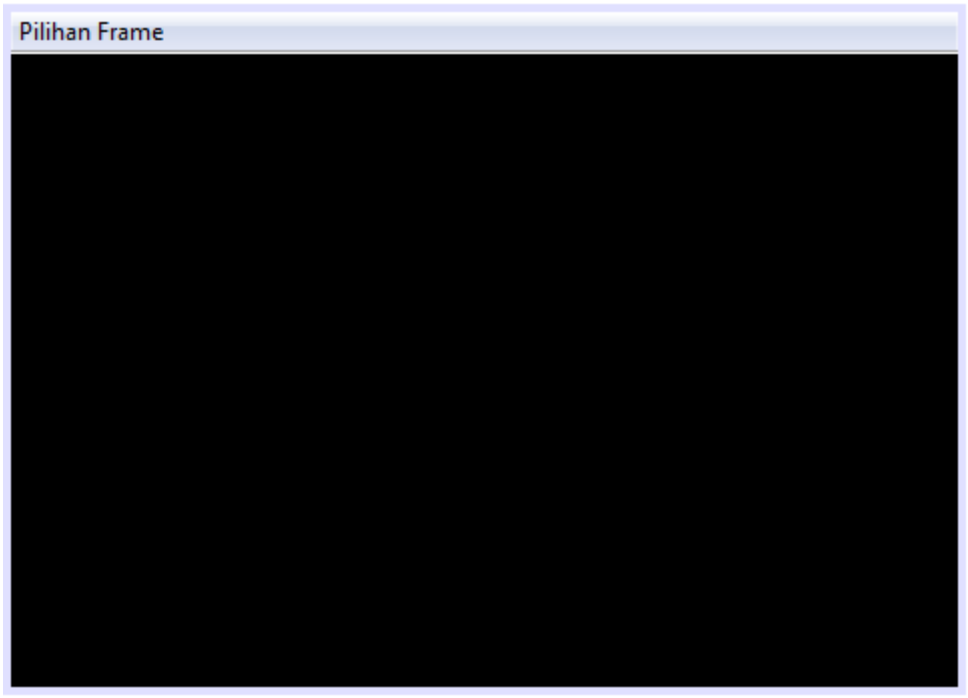
Berikan menu Bar dan 2 menu item, ubah menu bar hingga punya tampilan sebagai berikut:



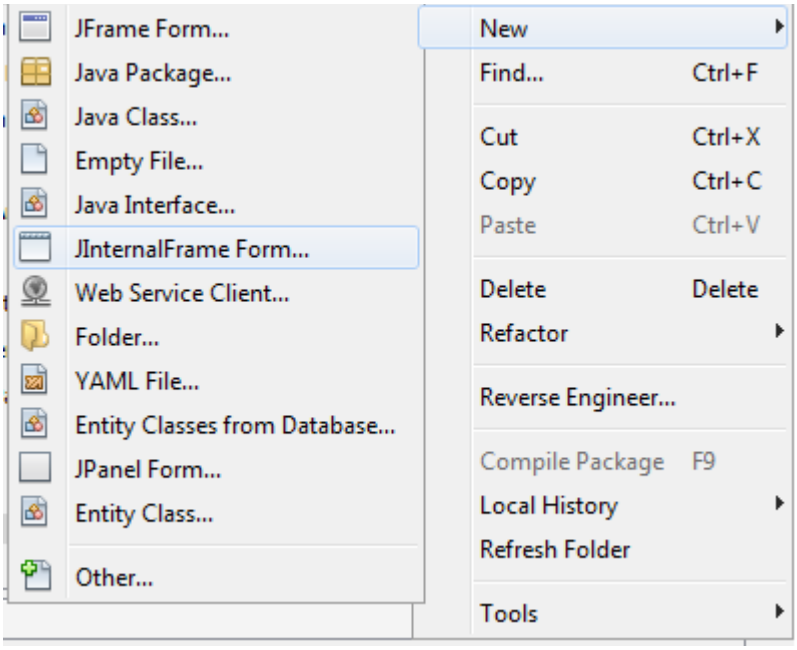
Tambahkan Desktop Pane pada Frame:



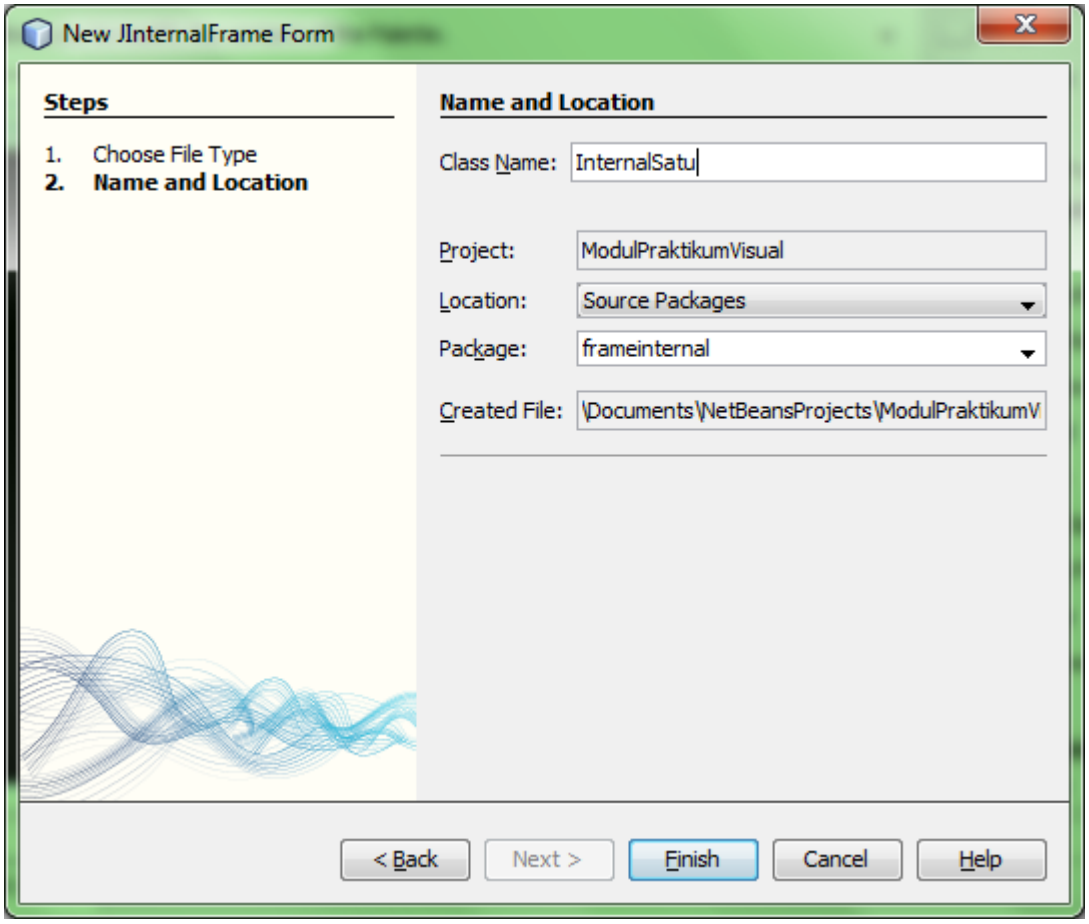
JDesktopPane ini merupakan tempat bagi internal frame nantinya. Agar desktop pane melebar dan memenuhi frame utama, pastikan frame utama memiliki layout “Border”.

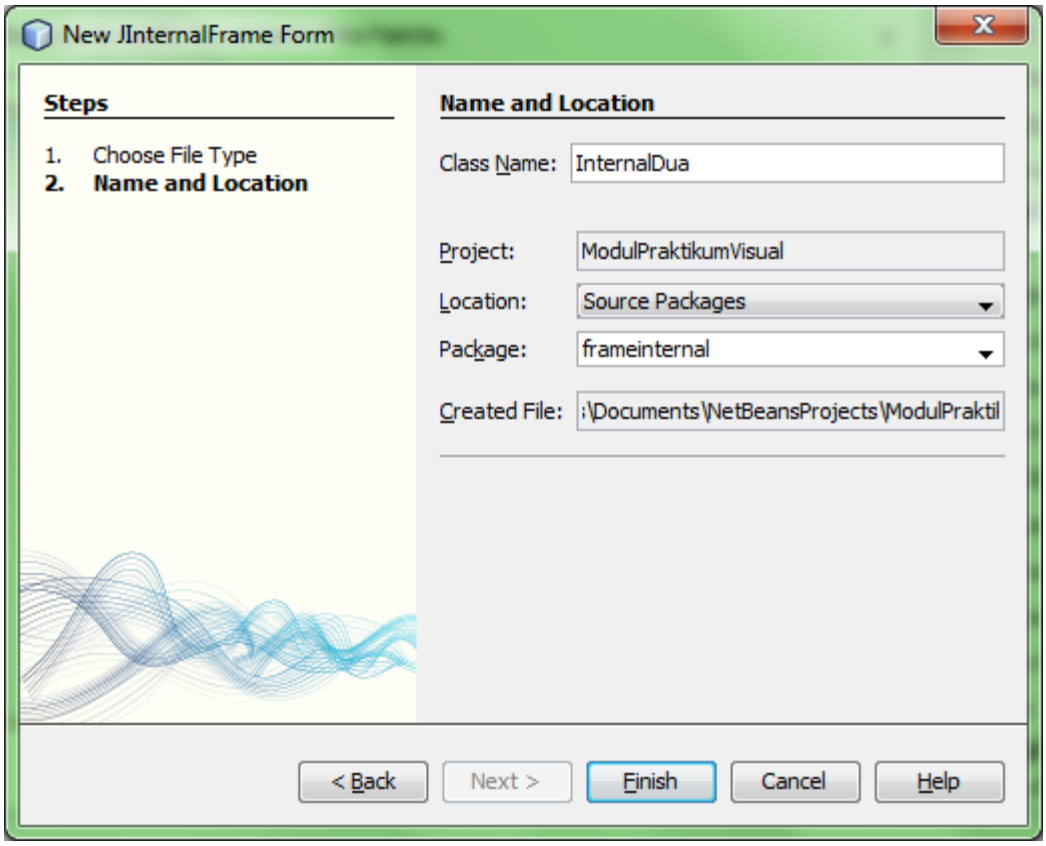


Buatlah 2 kelas internal frame dengan memilih di package terkait “New → JInternalFrame Form”

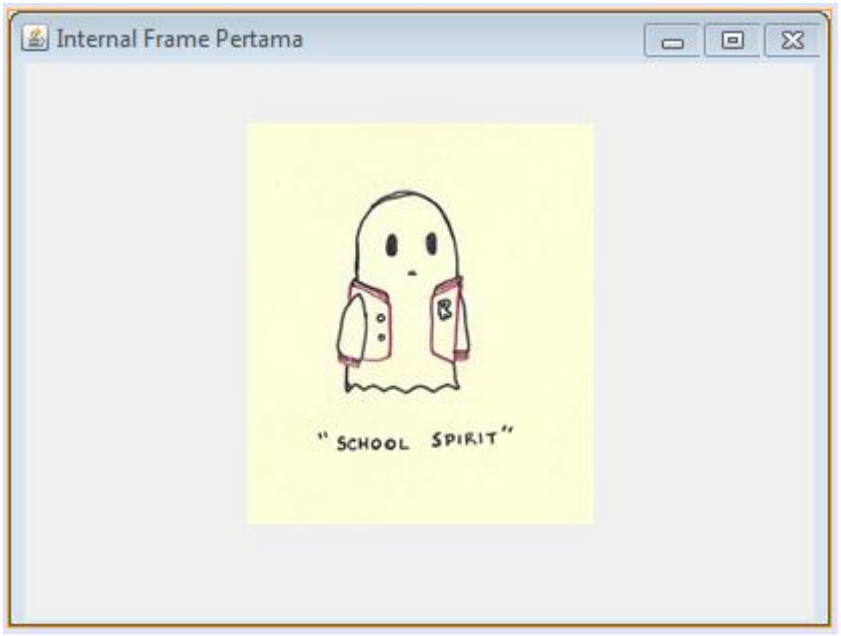


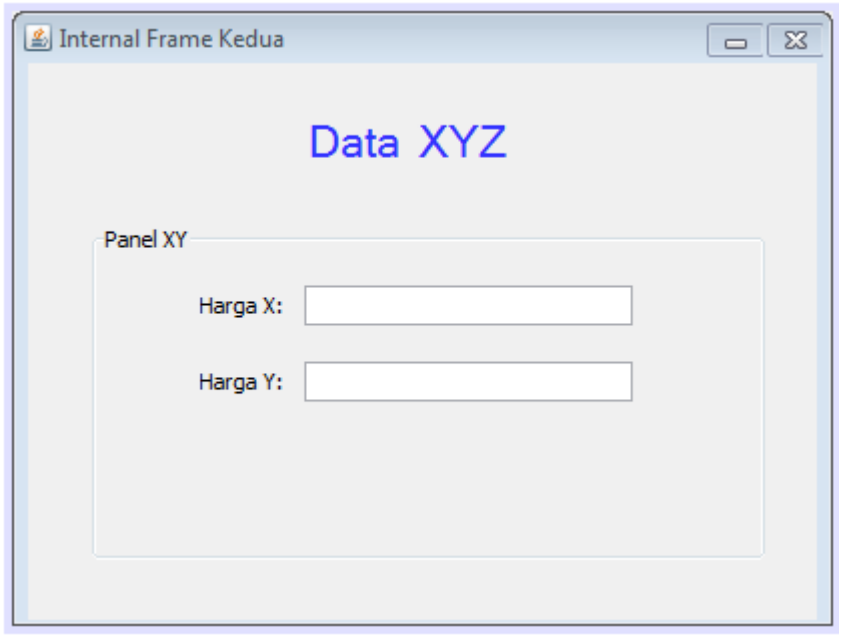
Jika pilihan ini tidak ada, pilih “Other”, pada list sebelah kiri pilih “Swing GUI Forms” dan di sebelah kanan, cari “JInternalFrame Form”.




New JInternalFrame Form
Steps
1. Choose File Type
2. Name and Location
Name and Location
Class Name: InternalDua
Project: ModulPraktikumVisual
Location: Source Packages
Package: frameinternal
Created File: :\Documents\NetBeansProjects\ModulPraktil
< Back
Next >
Finish
Cancel
Help

Baik internal frame 1 dan internal frame 2, tambahkan komponen layaknya JFrame dan JPanel. Ubah sebagai berikut (agar bisa di-close, resize, maximize dan dijadikan dalam bentuk icon— *diletakkan layaknya minimize window*, pada bagian properties centang “closable”, “maximizable”, “resizable” dan”iconifiable”):





Frame sudah terbentuk, yang perlu dilakukan, menghubungkan antara JMenuItem dengan JInternalFrame. Ketika menu di-klik, Internal Frame akan muncul. Berikan aksi (actionPerformed()) berikut pada menu item pertama:

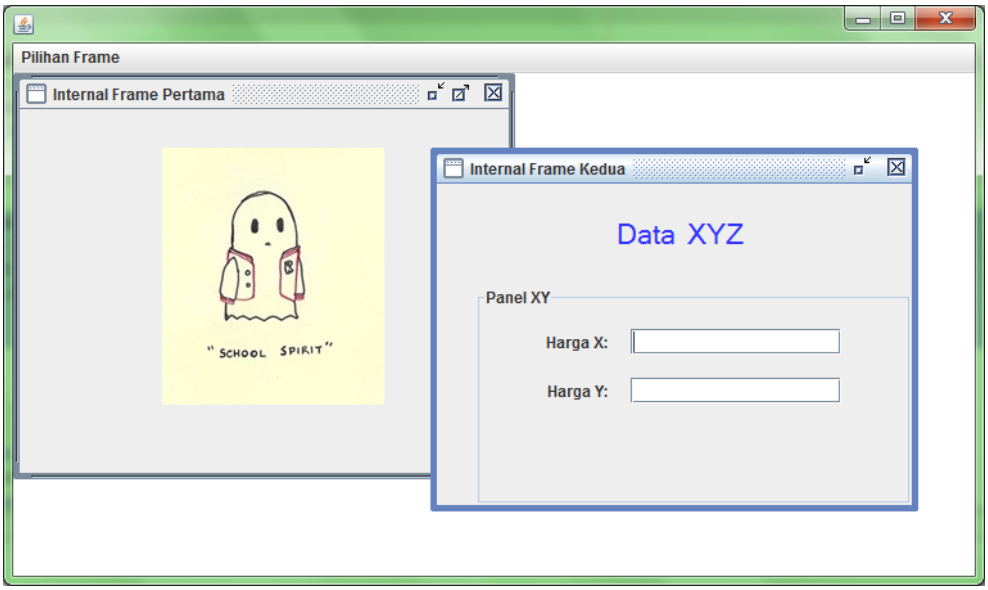
```
private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    InternalSatu is = new InternalSatu();
    jDesktopPane1.add(is);
    is.show();
}
```

Kode di atas, maksudnya, jika menu item pertama ditekan, maka objek dari JInternalFrame kelas "InternalSatu" akan dibentuk. Tambahkan objek tersebut di desktop pane, dan tampilkan internal frame yang sudah ditambahkan ke desktop pane.

Lakukan pula hal yang sama dengan menu item kedua:

```
private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    InternalDua id = new InternalDua();
    jDesktopPane1.add(id);
    id.show();
}
```

Sehingga ketika di-run:



Agar frame utama muncul di tengah-tengah screen, tambahkan kode berikut di konstruktor:

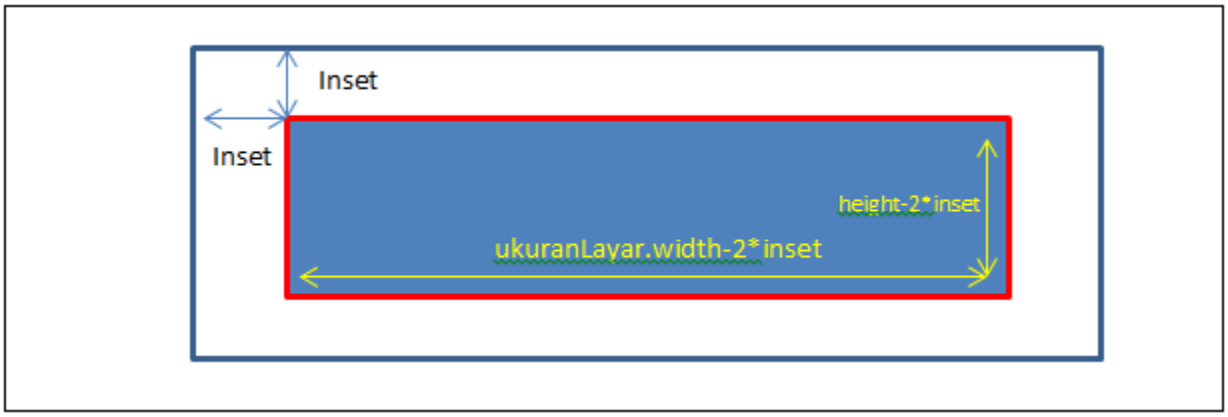
```
public KelasUtama() {
    initComponents();
    setLocationRelativeTo(null);
}
```

Agar Frame utama tidak tampil sekecil ukuran berikut,

tambahkan kode berikut:

```
public KelasUtama() {
    initComponents();
    setLocationRelativeTo(null);
    int inset = 50;
    Dimension ukuranLayar = Toolkit.getDefaultToolkit().getScreenSize();
    this.setBounds(inset, inset, (ukuranLayar.width-2*inset), (ukuranLayar.height-2*inset));
}
```

Maksud kode di atas, akan membuat ukuran frame yang muncul sebagai berikut:



5 BAB V KONEKSI DATABASE 1

5.1 IDENTITAS

Kajian

Swing Java, Database & Class Diagram**.**

Topik

1. Koneksi Database (JDBC) Oracle & MySQL
2. Read/Lihat Data

Referensi

1. http://docs.oracle.com/javase/tutorial/jdbc/index.html

Kompetensi Utama

1. Mahasiswa memahami konsep pengaksesan basis data MySQL menggunakan bahasa java
2. Mahasiswa mampu menerapkan query SELECT menggunakan bahasa pemrograman java.

Lama Kegiatan Praktikum

1. Jurnal Mandiri : 1 x 100 menit

Parameter Penilaian

1. Tugas Pendahuluan
2. Jurnal Mandiri

5.2 PENDAHULUAN & PENGENALAN JDBC

JDBC dibutuhkan untuk menghubungkan bahasa pemrograman java dengan database. JDBC merupakan singkatan dari Java DataBase Connectivity. JDBC merupakan driver untuk mengakses database. Analoginya seperti driver printer untuk menggunakan sebuah printer melalui computer. Driver JDBC sendiri merupakan koleksi class-class Java yang dikumpulkan dalam satu atau beberapa file .jar. JDBC yang digunakan berbeda-beda utuk setiap database yang digunakan.

Ada beberapa langkah yang harus dilakukan untuk mengakses database menggunakan bahasa java:

1) Import packages terkait. Untuk menggunakan JDBC package terkait harus diimport. Package yang biasanya diimport yaitu **import java.sql.***
2) Mendaftarkan driver JDBC tersebut. Biasanya merupakan 1 kelas tersendiri sebagai inisialisasi driver yang digunakan sehingga bisa membuka kanal pengaksesan database.
3) Membuka koneksi. Caranya dengan menggunakan method DriverManager.getConnection(), hal ini merepresentasikan koneksi fisik dengan database.
4) Eksekusi query. Pengaksesan database pasti tidak lepas dari query yang akan dikirimkan agar dieksekusi. Hal ini membutuhkan sebuah objek dengan tipe **Statement** untuk membangun dan men-submit SQL statement ke database.
5) Ekstrak data hasil eksekusi query. Umumnya, method yang dijalankan untuk mengakses nilai database (DML Query), tipe data hasil eksekusi query adalah **ResultSet** yang memiliki nilai record data dari database atau **integer** yang menyatakan jumlah row yang berpengaruh terhadap hasil query. Untuk meng-ekstrak ResultSet ke dalam bentuk Object (String dll), dibutuhkan method ResultSet.getXXX(). Untuk tipe integer, biasanya ditranslasikan sebagai bentuk boolean yang menyatakan apakah eksekusi berhasil dilakukan atau tidak.
6) Clean up. Setelah mengakses data, sebaiknya dilakukan closing database. Pengubahan nilai objek koneksi sebagai null (closing), kadang tidak dilakukan karena programmer lebih sering menyerahkan proses closing pada garbage collection JVM. Clean up dilakukan dengan memanggil method close() dari objek Statement yang dibentuk.

Terkait langkah ke-4 (point d), objek bertipe Statement dibentuk dari 3 interface yang mengandung method untuk mengeksekusi query yang diberikan:

a) Statement: merupakan general-purpose pengaksesan database. Digunakan untuk mengeksekusi query yang static seperti “select * from nama_tabel”. Interface ini tidak menerima parameter.
b) PreparedStatement: Digunakan untuk mengeksekusi query dinamis dan memiliki input parameter. Jika query static harus dieksekusi berulang kali, penggunaan preparedStatement akan lebih efektif dibandingkan Statement.
c) CallableStatement: berfungsi untuk mengakses stored procedure dari database (procedure, function dll)

Untuk mengeksekusi query, terdapat 3 method yang digunakan (terdapat pada objek bertipe Statement dan **PreparedStatement**):

a) boolean execute(String SQL) : mengembalikan nilai true jika objek ResultSet dapat diambil dan mengembalikan nilai false jika kondisi tersebut tidak terpenuhi; Method ini digunakan untuk mengeksekusi DDL statements atau ketika mengeksekusi truly dynamic SQL.
b) int executeUpdate(String SQL) : Mengembalikan jumlah baris yang terpengaruh oleh query yang dieksekusi. Penggunaannya ketika mengeksekusi query INSERT, UPDATE, atau DELETE statement.
c) ResultSet executeQuery(String SQL) : Mengembalikan objek ResultSet. Penggunaan method ini ketika diharapkan ada data yang diambil dari database menggunakan query select. Perbedaan dengan yang pertama, method tersebut hanya mengembalikan nilai true/false dan tidak mengembalikan datanya. Sedangkan method ini memiliki return value berupa data dari database.

5.3 PRAKTIK

5.3.1 Koneksi Database—MySQL

Terdapat sebuah table pada suatu database dengan informasi sebagai berikut:

Nama database: “pis12noltiga”
Url database: “jdbc:mysql://localhost/pis12noltiga”
Username: “root”
Password: “” <tidak ada password>
Driver JDBC: "com.mysql.jdbc.Driver"
Nama Table: desa_ninja
Atribut: id_desa, nama, pemimpin

Untuk mengakses database, dibutuhkan 1 kelas untuk mewakilkan langkah a-c. Hal ini tidak harus dilakukan, tetapi agar memudahkan maintainance data, jika terdapat perubahan database yang digunakan atau perubahan informasi lain seperti username dan password, dibuat 1 kelas untuk merepresentasikan hal ini. Jadi kelas lain yang mengakses database cukup membuat objek dari kelas “Pool Connection” ini.

Informasi yang dibutuhkan untuk membuat kelas ini adalah nama database, username, password dan driver JDBC yang digunakan.

Contoh dari Kelas “Connection Pool” berikut bernama “KoneksiDB”.

```
package aksesdb;

//Langkah pertama, import package terkait
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

/**
 *
 * @author Eja
 */
public class KoneksiDB {
   // driver JDBC driver dan database URL

   private final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
   private final String DB_URL = "jdbc:mysql://localhost/pis12noltiga";
   //  Database credentials
   private final String USER = "root";
   private final String PASS = "";
   private Connection conn = null;

   public void bukaKoneksi() {
      boolean flag = false;
      try {
         //Langkah ke-2: Registrasi JDBC
         Class.forName(JDBC_DRIVER);
      } catch (Exception e) {
         System.out.println(e.getMessage());
         flag = true;
      }
      if (!flag) {
         try {
            //Langkah ke-3: buka koneksi
            conn = DriverManager.getConnection(DB_URL, USER, PASS);
         } catch (SQLException e) {
            System.out.println(e.getMessage());
         }
      }
   }

   public Connection getConn() {
      return conn;
   }
}
```

5.3.2 Koneksi Database—Oracle

Terdapat sebuah table pada suatu database dengan informasi sebagai berikut:

URL database: “jdbc:oracle:thin:@//localhost:1521/XE”
Username: “uname”
Password: “password”
Driver JDBC: " oracle.jdbc.driver.OracleDriver" <tergantung versi Oracle yang digunakan>
Nama Table: desa_ninja
Atribut: id_desa, nama, pemimpin

URL database dipengaruhi jenis database yang digunakan. Sama seperti MySQL, untuk mengakses database, dibutuhkan 1 kelas untuk mewakilkan langkah a-c.

JDBC untuk oracle dapat di-download di:

http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html

Contoh dari Kelas “Connection Pool” untuk koneksi Oracle berikut bernama “KoneksiDB”.

```
package aksesdb;

//Langkah pertama, import package terkait
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

/**
 *
 * @author Eja
 */
public class KoneksiDB {
   // driver JDBC driver dan database URL

   private final String JDBC_DRIVER = " oracle.jdbc.driver.OracleDriver ";
   private final String DB_URL = " jdbc:oracle:thin:@//localhost:1521/XE ";
   //  Database credentials
   private final String USER = "uname";
   private final String PASS = "password";
   private Connection conn = null;

   public void bukaKoneksi() {
      boolean flag = false;
      try {
         //Langkah ke-2: Registrasi JDBC
         Class.forName(JDBC_DRIVER);
      } catch (Exception e) {
         System.out.println(e.getMessage());
         flag = true;
      }
      if (!flag) {
         try {
            //Langkah ke-3: buka koneksi
            conn = DriverManager.getConnection(DB_URL, USER, PASS);
         } catch (SQLException e) {
            System.out.println(e.getMessage());
         }
      }
   }

   public Connection getConn() {
      return conn;
   }
}
```

JDBC merupakan package class yang bukan bawaan dari bahasa Java. Sehingga library tersebut harus di-import terlebih dahulu ke project yang terkait. Caranya, klik kanan di bagian "Libraries" pada project yang mengakses database, pilih "Add Library". Di sini, bisa ditambahkan Library MySQL ataupun Oracle. Khusus Oracle, library ini bisa jadi sudah tersedia ataupun belum. Jika belum bisa memilih "Add Jar/Folder" atau "Create" setelah memilih "Add Library".

5.3.3 Menggunakan Query SELECT

Terdapat sebuah tabel "desa_ninja" di database (pis12noltiga) yang telah didefenisikan sebelumnya. Tabel tersebut memiliki struktur sebagai berikut:

Nama Atribut	Tipe Data	Keterangan
id_desa	Varchar(6)	Primary Key
nama	Varchar(20)	
pemimpin	Varchar(20)	

Isi dari tabel di atas adalah sebagai berikut:

			id_desa	nama	pemimpin
☐	✎	✕	D-001	Konohagakure	Hokage
☐	✎	✕	D-002	Sunagakure	Kazekage
☐	✎	✕	D-003	Iwagakure	Tsuchikage
☐	✎	✕	D-004	Otogakure	Orochimaru
☐	✎	✕	D-005	Kumogakure	Raikage
☐	✎	✕	D-006	Hoshigakure	Hoshikage

Untuk mengakses tabel di tas, buat 1 kelas "DesaNinja" yang merepresentasikan tabel desa_ninja. Atribut kelas tersebut disesuaikan dengan atribut dari database. Gunakan phpmyadmin, Netbeans atau editor gui apapun untuk mempercepat proses membentuk tabel. Hal ini dilakukan untuk mengimplementasikan langkah ke-4 sampai ke-6.

Kelas DesaNinja.java memiliki outline sebagai berikut:

```
/**
 *
 * @author Eja
 */

public class DesaNinja {
   private String id_desa;
   private String desa;
   private String pemimpin;
   private KoneksiDB kdb = new KoneksiDB();

   //konstruktor kosong
   public DesaNinja() {
   }

   //overloading konstruktor dengan parameter input adalah semua atribut class
   public DesaNinja(String id_desa, String desa, String pemimpin) {
      this.id_desa = id_desa;
      this.desa = desa;
      this.pemimpin = pemimpin;
   }

   public boolean masukkanData(){
       return true;
   }

   public void lihatData(String id_desa){

   }

   public void lihatNData(){

   }

   public boolean hapusData(String id_desa){

   }


   public boolean ubahData(String id_desa, String nama_baru, String pemimpin_baru){

   }
}
```

Terdapat 2 skenario untuk melakukan read data. Yaitu skenario untuk read tepat 1 data atau skenario untuk read N data. 2 skenario diwakilkan oleh method lihatData() dan lihatNData().

```
public void lihatData(String id_desa) throws SQLException {
    Connection dbConnection = null;
    PreparedStatement ps = null;
    ResultSet rs = null;

    this.id_desa = id_desa;

    String viewTableSQL = "SELECT * FROM desa_ninja WHERE id_desa = ?";

    try {
        //buka koneksi saat objek dari desa ninja dibentuk
        kdb.bukaKoneksi();
        //inisialisasi dbConnection dari objek Connection
        dbConnection = kdb.getConn();

        //Langkah ke 4 bagian 1
        ps = dbConnection.prepareStatement(viewTableSQL);
        ps.setString(1, this.id_desa);

        //langkah 4: eksekusi query
        rs = ps.executeQuery();

        //langkah ke 5
        //ekstrak data dari ResultSet
        if (rs.next()) {
            System.out.println("ID Desa: " + rs.getString(1));
            System.out.println("Nama Desa: " + rs.getString(2));
            System.out.println("Pemimpin: " + rs.getString(3));
        }

    } catch (Exception e) {
        System.out.println(e.getMessage());
    } finally {
        //langkah ke-6
        ps.close();
    }
}
```

Akan ada tambahan keyword throws pada method tersebut dikarenakan ada baris berikut pada method tersebut. Adanya throws ini membuat baris program yang memanggil method ini harus diselubungi blok try-catch.

```
finally{
        ps.close();
    }
```

Method tersebut dapat berlaku untuk Oracle maupun MySQL. Perbedaan hanya pada kelas "KoneksiDB" yang digunakan. Kelas tersebut mempunyai perbedaan di nama driver, nama database, username dan password menyesuaikan dengan informasi database yang digunakan. LihatNData dan lihatData sebenarnya memiliki logika pemrograman yang sama. Perbedaan hanya pada query yang digunakan dan terdapat masukan primary key rujukan sebagai pengambilan data untuk digunakan pada query SELECT.

Untuk menjalankan method di atas, dituliskan sebuah class Main.java sebagai berikut:

```
package aksesdb;

import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;

public class Main {
   public static void main(String[] args) {
      DesaNinja dn = new DesaNinja();
      try {
         dn.lihatData("D-001");
      } catch (SQLException ex) {
         Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null, ex);
      }
   }
}
```

Isi method lihatNData adalah sebagai berikut:

```
public void lihatNData() throws SQLException {
      Connection dbConnection = null;
      PreparedStatement ps = null;
      ResultSet rs = null;

      String viewTableSQL = "SELECT * FROM desa_ninja";


      try {
         //buka koneksi saat objek dari desa ninja dibentuk
         kdb.bukaKoneksi();
         //inisialisasi dbConnection dari objek Connection
         dbConnection = kdb.getConn();

         //Langkah ke 4 bagian 1
         ps = dbConnection.prepareStatement(viewTableSQL);

         //langkah 4: eksekusi query
         rs = ps.executeQuery();

         //langkah ke 5
         //ekstrak data dari ResultSet

         System.out.println("ID Desa\t\tNama\t\t\tPemimpin");

System.out.println("================================================================")
;

         while (rs.next()) {
            System.out.print(" " + rs.getString(1));
            System.out.print("\t\t" + rs.getString(2));
            System.out.println("\t\t " + rs.getString(3));
         }

      } catch (Exception e) {
         System.out.println(e.getMessage());
      } finally {
         //langkah ke-6
         ps.close();
      }
   }
```

Yang perlu diperhatikan, ekstrak data dari result set harus dilakukan sebelum langkah ke-6 dilakukan. Jika memang harus menampilkan data setelah menutup koneksi, disarankan untuk mengubah/mengambil result set ke objek lain bertipe Object[][] atau tipe lainnya. Untuk pengambilan result set bisa menggunakan while ataupun if. Syntax if dapat digunakan karena memang hanya mengembalikan 1 row data saja.

Untuk menjalankan method di atas, buat sebuah kelas Main.java sebagai berikut:

```
public class Main {
   public static void main(String[] args) {
      DesaNinja dn = new DesaNinja();
      try {
         dn.lihatNData();
      } catch (SQLException ex) {
         Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null, ex);
      }
   }
}
```

5.3.4 Solusi Lengkap

```
package aksesdatabase;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

/**
 *
 * @author EJA
 */
public class KoneksiDB {
   private final String NAMA_DRIVER = "com.mysql.jdbc.Driver";
   private final String NAMA_DB = "jdbc:mysql://localhost/pis12noltiga";
   private final String USERNAME = "root";
   private final String PASSWORD = "";
   private Connection conn;

   public void bukaKoneksi(){
      boolean flag = false;
      try {
         //langkah ke-2
         Class.forName(NAMA_DRIVER);
         flag = true;
      } catch (ClassNotFoundException ex) {
         System.out.println(ex.getMessage());
      }

      if(flag){
         try {
            //langkah ke-3
            conn = DriverManager.getConnection(NAMA_DB,USERNAME,PASSWORD);
         } catch (SQLException ex) {
            System.out.println(ex.getMessage());
         }
      }
   }

   public Connection getConn() {
      return conn;
   }
}
```

```
package aksesdatabase;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

/**
 *
 * @author EJA
 */
public class DesaNinja {

   private Connection koneksi;
   private PreparedStatement ps;
   private KoneksiDB k = new KoneksiDB();
   private String id, nama, pemimpin;

   public DesaNinja(){}

   public void lihatNData(){}

   public void lihatNData() throws SQLException{
      try {
         k.bukaKoneksi();
         koneksi = k.getConn();
         String kueri = "SELECT * FROM desa_ninja";
         ps = koneksi.prepareStatement(kueri);
         ResultSet rs = ps.executeQuery();
         while(rs.next()){
            System.out.println("Data Desa Ninja");
            System.out.println("Id Desa: "+rs.getString(1));
            System.out.println("Nama: "+rs.getString(2));
            System.out.println("Pemimpin: "+rs.getString(3));
            System.out.println("");
         }
      } catch (SQLException ex) {
         System.out.println(ex.getMessage());
      }finally{
         ps.close();
      }
   }

}
```

```
package aksesdatabase;

import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;

/**
 *
 * @author EJA
 */
public class Main {
   public static void main(String[] args) {
      DesaNinja dn = new DesaNinja();
      try {
         dn.lihatNData();
      } catch (SQLException ex) {
         System.out.println(ex.getMessage());
      }
   }
}
```

5.4 PERTANYAAN EVALUASI

Apakah pengaksesan basis data di atas sudah sesuai dengan konsep OO (berdasarkan aturan yang digunakan framework OO) ??

6 BAB VI KONEKSI DATABASE 2

6.1 IDENTITAS

Kajian

Swing Java, Database & Class Diagram.

Topik

1. Koneksi Database (JDBC) Oracle & MySQL
2. Query INSERT, UPDATE, DELETE

Referensi

1. http://docs.oracle.com/javase/tutorial/jdbc/index.html

Kompetensi Utama

1. Mahasiswa memahami konsep pengaksesan basis data MySQL menggunakan bahasa java
2. Mahasiswa mampu menerapkan query INSERT menggunakan bahasa pemrograman java.
3. Mahasiswa mampu menerapkan query DELETE menggunakan bahasa pemrograman java.
4. Mahasiswa mampu menerapkan query UPDATE menggunakan bahasa pemrograman java.

Lama Kegiatan Praktikum

1. Pertemuan Mandiri : 1 x 100 menit

Parameter Penilaian

1. Jurnal Mandiri

6.2 PRAKTIK

6.2.1 Menggunakan Query INSERT

Isi dari method untuk memasukkan data input user ke dalam database adalah sebagai berikut:

```
public boolean masukkanData() throws SQLException {
    //deklarasi connection dan preparedStatement
    Connection dbConnection = null;
    PreparedStatement ps = null;
    int rowAffect = 0;

    String insertTableSQL = "INSERT INTO desa_ninja"
            + "(id_desa, nama, pemimpin) VALUES"
            + "(?,?,?)";
    try {
        //buka koneksi saat objek dari desa ninja dibentuk
        kdb.bukaKoneksi();
        //inisialisasi dbConnection dari objek Connection
        dbConnection = kdb.getConn();

        //Langkah ke 4 bagian 1
        ps = dbConnection.prepareStatement(insertTableSQL);
        ps.setString(1, this.id_desa);
        ps.setString(2, this.nama);
        ps.setString(3, this.pemimpin);
        //langkah 4: eksekusi query
        rowAffect = ps.executeUpdate();

    } catch (Exception e) {
        System.out.println(e.getMessage());
    } finally{
         //langkah ke 6
        ps.close();
    }

     //langkah ke-5: menterjemahkan hasil yang dikembalikan
     //dari bentuk integer ke dalam bentuk boolean sebagai representasi keberhasilan eksekusi
    if(rowAffect > 0){
        return true;
    }else{
        return false;
    }
}
```

Untuk menjalankan method di atas, buat sebuah kelas Main.java sebagai berikut:

```
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;

/**
 *
 * @author Eja
 */
public class Main {
   public static void main(String[] args) {
      try {
         DesaNinja dn1 = new DesaNinja("D-004", "Kirigakure", "Mizukage");
         if (dn1.masukkanData()) {
            JOptionPane.showMessageDialog(null, "Berhasil", "Status",
JOptionPane.INFORMATION_MESSAGE, null);
            System.out.println("Berhasil");
         } else {
            JOptionPane.showMessageDialog(null, "Gagal", "Status", JOptionPane.ERROR_MESSAGE,
null);
         }
      } catch (SQLException ex) {
         Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null, ex);
      }
   }
}
```

6.2.2 Menggunakan Query DELETE

Delete sama hal-nya seperti lihat 1 data, butuh masukan berupa id mana yang harus di-delete. Untuk logika, query dan lain-lain mirip sama insert data. Methodnya dituliskan sebagai berikut:

```
public boolean hapusData(String id_desa) throws SQLException {
    //deklarasi connection dan preparedStatement
    Connection dbConnection = null;
    PreparedStatement ps = null;
    int rowAffect = 0;

    this.id_desa = id_desa;

    String deleteTableSQL = "DELETE from desa_ninja WHERE id_desa = ? ";
    try {
        //buka koneksi saat objek dari desa ninja dibentuk
        kdb.bukaKoneksi();
        //inisialisasi dbConnection dari objek Connection
        dbConnection = kdb.getConn();

        //Langkah ke 4 bagian 1
        ps = dbConnection.prepareStatement(deleteTableSQL);
        ps.setString(1, this.id_desa);
        //langkah 4: eksekusi query
        rowAffect = ps.executeUpdate();

    } catch (Exception e) {
        System.out.println(e.getMessage());
    } finally {
        //langkah ke 6
        ps.close();
    }
    //langkah ke 5
    if (rowAffect > 0) {
        return true;
    } else {
        return false;
    }
}
```

Main.java dituliskan sebagai berikut:

```
/**
 *
 * @author Eja
 */

import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;

public class Main {
   public static void main(String[] args) {
      try {
         DesaNinja dn = new DesaNinja();
         if(dn.hapusData("D-003")){
            JOptionPane.showMessageDialog(null, "Berhasil", "Status",
JOptionPane.INFORMATION_MESSAGE, null);
         }else{
            JOptionPane.showMessageDialog(null, "Gagal", "Status", JOptionPane.ERROR_MESSAGE,
null);
         }
      } catch (SQLException ex) {
         Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null, ex);
      }
   }
}
```

6.2.3 Menggunakan Query UPDATE

Prose update diwakili oleh method ubahData(). Isi method adalah sebagai berikut:

```
public boolean ubahData(String id_desa, String nama_baru, String pemimpin_baru)
                    throws SQLException {
  //deklarasi connection dan preparedStatement

  Connection dbConnection = null;
  PreparedStatement ps = null;
  int rowAffect = 0;

  String updateTableSQL = "UPDATE desa_ninja SET nama = ?, pemimpin = ?"
      +" WHERE id_desa = ?";
  try {
    //buka koneksi saat objek dari desa ninja dibentuk
    kdb.bukaKoneksi();
    //inisialisasi dbConnection dari objek Connection
    dbConnection = kdb.getConn();

    //Langkah ke 4 bagian 1
    ps = dbConnection.prepareStatement(updateTableSQL);
    ps.setString(3, id_desa);
    ps.setString(1, nama_baru);
    ps.setString(2, pemimpin_baru);
    //langkah 4: eksekusi query
    rowAffect = ps.executeUpdate();

  } catch (Exception e) {
    System.out.println(e.getMessage());
  } finally {
    //langkah ke 6
    ps.close();
  }
  //langkah ke 5
  if (rowAffect > 0) {
    return true;
  } else {
    return false;
  }
}
```

Main.java dituliskan sebagai berikut:

```
/**
 *
 * @author Eja
 */

import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;

public class Main {
   public static void main(String[] args) {
      DesaNinja dn = new DesaNinja();
      boolean x = false;
      try {
         x = dn.ubahData("D-002", "Iwagakure 2", "Tsuchikage Gen 2");
      } catch (SQLException ex) {
         Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null, ex);
      }
      if(x){
         JOptionPane.showMessageDialog(null, "Berhasil", "Status",
JOptionPane.INFORMATION_MESSAGE, null);
      }else{
         JOptionPane.showMessageDialog(null, "Gagal", "Status", JOptionPane.ERROR_MESSAGE,
null);
      }
   }
}
```

Terdapat sebuah variabel penampung boolean x untuk menampung return value dari pemanggilan fungsi ubah. Variabel ini bersifat optional, dan bisa dilakukan seperti method masukkanData()—tanpa variabel boolean.

6.3 PERTANYAAN EVALUASI

Apakah pengaksesan basis data di atas sudah sesuai dengan konsep OO (berdasarkan aturan yang digunakan framework OO) ??

7 BAB VII KOMPONEN SWING-NON VISUAL EDITOR

7.1 IDENTITAS

Kajian

Komponen Swing Java non-Visual Editor; Database & Swing

Topik

1. Komponen Swing: JFrame, JPanel, JLabel, JButton, JRadioButton, JButtonGroup, JCheckBox, JComboBox, JMenuBar, JMenu, JMenuItem, JTable, JTextField

Referensi

1. http://docs.oracle.com/javase/tutorial/uiswing/index.html

Kompetensi Utama

1. Mahasiswa memahami penggunaan komponen swing untuk membuat aplikasi desktop.
2. Mahasiswa mampu membuat aplikasi desktop menggunakan komponen swing—sesuai yang dituliskan di topik—tanpa bantuan visual editor.

Lama Kegiatan Praktikum

1. Pertemuan Terbimbing : 1 x 50 menit
2. Kegiatan Mandiri : 1 x 50 menit

Parameter Penilaian

1. Tugas Pendahuluan
2. Jurnal Mandiri

7.2 JFRAME

JFrame merupakan top-level-container dari komponen swing. Terdapat 2 cara untuk mengimplementasikan JFrame pada pemrograman swing java. Cara pertama dengan membuat objek dari class JFrame pada kelas yang mengimplementasikan GUI. Setelah objek JFrame dibentuk, komponen lain dapat ditambahkan. Cara kedua adalah dengan membuat sebuah class yang meng-extends JFrame tersebut.

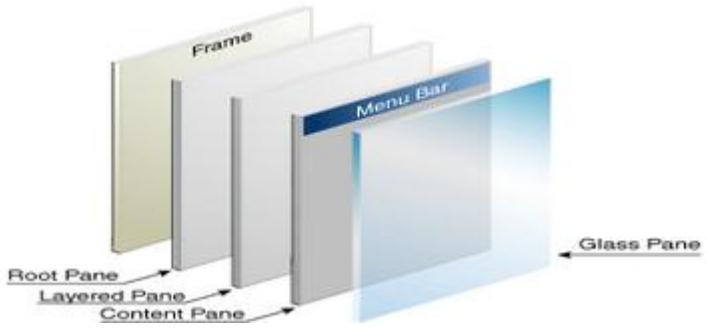
Cara Pertama:

```
import javax.swing.*;

public class FrameDemo {
   private static void createAndShowGUI() {
      JFrame frame = new JFrame("FrameDemo");
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      //Display the window.
      frame.pack();
      frame.setVisible(true);
   }

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

Cara Kedua:

```
public class KelasFrame extends javax.swing.JFrame {

  public KelasFrame() {
      initComponents();
   }
   private void initComponents() {
          //hasil generate tools
   }
   public static void main(String args[]) {
      java.awt.EventQueue.invokeLater(new Runnable() {
         public void run() {
            new KelasFrame().setVisible(true);
         }
      });
   }
}
```

Yang perlu diperhatikan, pada pemrograman swing GUI, terdapat hirarki sebagai berikut:



Setiap container mempunyai fungsinya masing-masing. Jadi, sebaiknya tidak (secara paradigm pemrograman) untuk meletakkan komponen swing (seperti JButton, JTextField, JLabel dll) di dalam objek JFrame secara langsung. Lebih baik didahului oleh JPanel. Khusus objek JMenuBar, komponen ini harus diletakkan di objek JFrame dengan memanggil method "setJMenuBar".

7.3 JPANEL

JPanel merupakan container yang termasuk ke dalam content pane. Content pane merupakan tempat peletakan komponen swing seperti button, textfield dan komponen "swing control" lainnya. Komponen tersebut dapat diletakkan langsung ke dalam JFrame, tapi JFrame bertindak sebagai top-level-container, dan bukan pane tempat meletakkan content, sehingga lebih baik meletakkan komponen "swing control" pada JPanel dan bukan pada JFrame.

JPanel dapat dideklarasikan sebagai berikut:

```
JPanel p = new JPanel(new BorderLayout());
```

Atau

```
JPanel p = new JPanel();
```

Perbedaan pembuatan objek cara pertama dengan kedua terletak pada inisialisasi layout yang digunakan (Lihat modul praktikum 9 & 3). Layout pada pendeklarasian pertama menggunakan Border layout, sedangkan pada cara kedua layout tidak ditentukan. Jika tidak ditentukan, maka panel akan memiliki layout bertipe Flow (default JFrame adalah Flow Layout).

7.4 JLABEL

Label merupakan komponen untuk menghasilkan “unselectable” gambar dan teks. Pada label dapat diletakkan gambar (objek dari kelas ImageIcon) dan Teks (objek String). Parameter pada konstruktor dapat berisi text dan image saja atau dapat berisi keduanya ditambah posisi tampilan label tersebut. Contoh pembuatan objek label:

```
ImageIcon icon = createImageIcon("images/middle.gif", "a pretty but meaningless splat");
JLabel label1 = ("Image and Text", icon,JLabel.CENTER);
JLabel label2 = new JLabel("Text-Only Label");
JLabel label3 = new JLabel(icon);
```

Beberapa method yang digunakan beserta fungsinya yaitu:

Method	Fungsi
setText(“X”)	Untuk menge-set tulisan teks pada label
getText()	Mengambil tulisan teks pada label
setToolTipText()	Memberikan tooltip pada label

7.5 JBUTTON

Merupakan komponen untuk membuat tombol. Kelas yang digunakan adalah JButton. Pada komponen ini, selain keterangan teks, dapat juga ditambahkan image/icon. Contoh pembuatan objek:

```
JButton b2 = new JButton("Tombol B2");
```

Atau

```
ImageIcon leftButtonIcon = createImageIcon("images/right.gif");
JButton b1 = new JButton("Tombol B1", leftButtonIcon);
```

Pada button dan menu dapat ditambahkan mnemonic. Mnemonic merupakan penggunaan tombol dengan menggunakan keyboard. Biasanya terdapat 1 huruf yang digunakan sebagai penanda yang berfungsi sama dengan menekan tombol. Mnemonic bekerja dengan menekan “Alt+huruf penanda”. Penanda yang biasa digunakan adalah huruf pertama dari keterangan tombol. Contoh cara menge-set mnemonic (contohnya menunjukkan bahwa tombol akan aktif jika ALT+D ditekan):

```
b1.setMnemonic(KeyEvent.VK_D);
```

Beberapa method yang digunakan untuk button dan fungsinya adalah sebagai berikut:

Method	Fungsi
setText("X")	Untuk menge-set tulisan teks pada button
setMnemonic(KeyEvent.VK_M);	Menge-set mnemonic pada tombol
setToolTipText()	Memberikan tooltip pada button
setEnabled(false);	Menge-set button dapat di-klik atau tidak. Parameter "false" menyatakan button disable, dan sebaliknya
setActionCommand	Menge-set nama action performed dari button tersebut

Contoh:

```
/**
 *
 * @author Eja
 */
import javax.swing.AbstractButton;
import javax.swing.JButton;
import javax.swing.JPanel;
import javax.swing.JFrame;
import java.awt.event.KeyEvent;


public class SimpleButtonDemo extends JPanel{
   protected JButton b1;

   public SimpleButtonDemo() {

      b1 = new JButton("Tombol 1");
      b1.setVerticalTextPosition(AbstractButton.CENTER);
      b1.setHorizontalTextPosition(AbstractButton.LEADING); //aka LEFT, for left-to-right locales
      b1.setMnemonic(KeyEvent.VK_D);
      b1.setActionCommand("lakukan");

      //Tambahkan action listener untuk button
      b1.setToolTipText("Bentuk tombol 1");

      //menambahkan button ke kontainer
      //hal ini bisa dilakukan karena class meng-extends JPanel
      add(b1);
```

```
    }

    private static void createAndShowGUI() {
        //membuat frame
        JFrame frame = new JFrame("ButtonDemo");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        //membuat content pane
        SimpleButtonDemo newContentPane = new SimpleButtonDemo();
        newContentPane.setOpaque(true); //content panes must be opaque
        frame.setContentPane(newContentPane);

        //Memunculkan window
        frame.pack();
        frame.setVisible(true);
    }


    public static void main(String[] args) {

        javax.swing.SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                createAndShowGUI();
            }
        });
    }
}
```

7.6 JRadioButton & ButtonGroup

Contoh Konstruktor dan method:

```
//konstruktor radio button
JRadioButton birdButton = new JRadioButton(birdString);
birdButton.setMnemonic(KeyEvent.VK_B);
birdButton.setActionCommand(birdString);
birdButton.setSelected(true);

JRadioButton catButton = new JRadioButton(catString);
catButton.setMnemonic(KeyEvent.VK_C);
catButton.setActionCommand(catString);

//konstruktor button group
ButtonGroup group = new ButtonGroup();
group.add(birdButton);
group.add(catButton);
```

Beberapa method yang digunakan untuk komponen button group:

Method	Fungsi
add(radio1)	Menambahkan radio button untuk menjadi anggota pada grup tersebut
getButtonCount()	Mengembalikan nilai berupa jumlah radio button pada grup tersebut
clearSelection()	Menghapus state terpilih dari semua radio button pada grup tersebut
remove()	Menghapus radio button untuk menjadi anggota pada grup tersebut

Beberapa method yang digunakan untuk komponen radio button dan fungsinya adalah sebagai berikut:

Method	Fungsi
setText("X")	Untuk menge-set tulisan teks pada radio button
getText()	Untuk meng-get tulisan teks pada radio button
setMnemonic(KeyEvent.VK_M);	Menge-set mnemonic pada radio button
setToolTipText()	Memberikan tooltip pada radio button
setEnabled(false);	Menge-set button dapat di-klik atau tidak. Parameter "false" menyatakan radio button disable, dan sebaliknya
setSelected(true)	Menge-set radio button apakah mempunyai state dipilih saat pertama kali dijalankan atau tidak.
setActionCommand	Menge-set nama action performed dari radio button tersebut
isSelected()	Mengecek apakah radio button sedang dipilih atau tidak

Contoh:

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

/**
 *
 * @author Eja
 */
public class SimpleRadioButtonDemo extends JPanel{

   static String birdString = "Bird";
   static String catString = "Cat";
   static String rabbitString = "Rabbit";
   JRadioButton birdButton, catButton, rabbitButton;

   public SimpleRadioButtonDemo() {
      super();

      //Membuat Radio Button
      birdButton = new JRadioButton(birdString);
      birdButton.setMnemonic(KeyEvent.VK_B);
      birdButton.setActionCommand(birdString);
      birdButton.setSelected(true);

      catButton = new JRadioButton(catString);
      catButton.setMnemonic(KeyEvent.VK_C);
      catButton.setActionCommand(catString);

      rabbitButton = new JRadioButton(rabbitString);
      rabbitButton.setMnemonic(KeyEvent.VK_R);
      rabbitButton.setActionCommand(rabbitString);

      //Membuat Grup Button
      ButtonGroup group = new ButtonGroup();
      //Memasukkan radio button ke grup
      group.add(birdButton);
      group.add(catButton);
      group.add(rabbitButton);

      //menambahkan radio button pada panel
      this.add(birdButton);
      this.add(catButton);
      this.add(rabbitButton);
```

```
  }
  private static void createAndShowGUI() {
    //membuat frame
    JFrame frame = new JFrame("RadioButtonDemo");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    //membuat content pane
    SimpleRadioButtonDemo newContentPane = new SimpleRadioButtonDemo();
    newContentPane.setOpaque(true); //content panes must be opaque
    frame.setContentPane(newContentPane);

    //Memunculkan window
    frame.pack();
    frame.setVisible(true);
  }

  public static void main(String[] args) {
    javax.swing.SwingUtilities.invokeLater(new Runnable() {
      public void run() {
        createAndShowGUI();
      }
    });
  }
}
```

7.7 JCheckBox

Contoh konstruktor dan method:

```
    chinButton = new JCheckBox("Chin");
    chinButton.setMnemonic(KeyEvent.VK_C);
    chinButton.setSelected(true);
```

Method yang sering digunakan:

Method	Fungsi
setText("X")	Untuk menge-set tulisan teks
getText()	Untuk meng-get tulisan teks
setMnemonic(KeyEvent.VK_M);	Menge-set mnemonic
setToolTipText()	Memberikan tooltip
setEnabled(false);	Menge-set apakah komponen dapat di-klik atau tidak. Parameter "false" menyatakan radio button disable, dan sebaliknya
setSelected(true)	Menge-set check box apakah mempunyai state dipilih saat pertama kali dijalankan atau tidak.
setActionCommand	Menge-set nama action performed dari komponen tersebut
isSelected()	Mengecek apakah check box sedang dipilih atau tidak

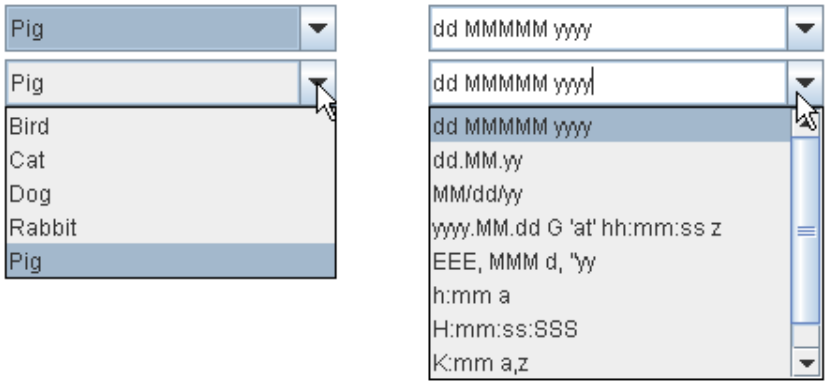
Contoh:

```
/**
 *
 * @author Eja
 */
import java.awt.event.*;
import javax.swing.*;


public class SimpleCheckBoxDemo extends JPanel {
   JCheckBox miButton;
   JCheckBox kaButton;
   JCheckBox tkButton;

   StringBuffer choices;

   public SimpleCheckBoxDemo() {
      super();

      //Create the check boxes.
      miButton = new JCheckBox("MI");
      miButton.setMnemonic(KeyEvent.VK_M);
      miButton.setSelected(true);
```

```
        kaButton = new JCheckBox("KA");
        kaButton.setMnemonic(KeyEvent.VK_K);
        kaButton.setSelected(true);

        tkButton = new JCheckBox("TK");
        tkButton.setMnemonic(KeyEvent.VK_T);
        tkButton.setSelected(true);

        //Pilihan pertama secara default
        //semua radio button terpilih
        choices = new StringBuffer("mkt");

        //Tambahkan cek box di panel
        this.add(miButton);
        this.add(kaButton);
        this.add(tkButton);
    }

    private static void createAndShowGUI() {
        //membuat frame
        JFrame frame = new JFrame("CheckBoxDemo");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        //membuat content pane
        JComponent newContentPane = new SimpleCheckBoxDemo();
        newContentPane.setOpaque(true); //content panes must be opaque
        frame.setContentPane(newContentPane);

        //Memunculkan window
        frame.pack();
        frame.setVisible(true);
    }

    public static void main(String[] args) {
        javax.swing.SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                createAndShowGUI();
            }
        });
    }
}
```

7.8 JComboBox

Combo box merupakan pemilihan menu melalui menu drop down. Sifatnya harus memilih salah satu. Terdapat 2 tipe combo box, editable combo box (kanan) dan uneditable combo box (kiri). Perbedaannya, untuk editable combo box, combo box dapat dituliskan layaknya text field.



Konstruktor dan method:

```
String[] petStrings = { "Bird", "Cat", "Dog", "Rabbit", "Pig" };
JComboBox petList = new JComboBox(petStrings);
```

Konstruktor memiliki parameter masukan berupa model dari comboBox (bisa dideklarasikan terpisah). Selain itu, Model dapat berupa Array of Object (Object[]) atau Vector. Method yang paling sering digunakan dan fungsinya adalah sebagai berikut:

Method	Fungsi
setEditable(true)	Menge-set tipe combo box, apakah editable combo box atau uneditable combo box
setSelectedIndex(4)	Menge-set indeks item pada combo box. Item dihitung dari nilai 0. (jika seperti contoh di atas, maka “Pig” yang dipilih)
getSelectedItem()	Mengembalikan objek terpilih dari combo box. Jika ingin dimasukkan ke dalam String, harus dilakukan casting terlebih dahulu
getSelectedIndex()	Mengembalikan index item terpilih
getItemAt(3)	Mengembalikan item pada index yang ditentukan
insertItemAt(objek, indek)	Memasukkan item pilihan berupa sebuah objek bertipe Object pada index ke indek
getItemCount()	Mengembalikan jumlah item pada combo box

Contoh:

```
/**
 *
 * @author Eja
 */
import javax.swing.*;

public class SimpleComboBoxDemo extends JPanel{

   public SimpleComboBoxDemo() {
      super();

      String[] daftarPilihan = {"-Pilih-", "Komputer", "Ekonomi", "Bahasa", "Eksakta"};

      JComboBox pilMK = new JComboBox(daftarPilihan);
      pilMK.setSelectedIndex(0);
      add(pilMK);
   }

   private static void createAndShowGUI() {
      //membuat frame
      JFrame frame = new JFrame("ComboBoxDemo");
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      //membuat content pane
      JComponent newContentPane = new SimpleComboBoxDemo();
      newContentPane.setOpaque(true); //content panes must be opaque
      frame.setContentPane(newContentPane);

      //Memunculkan window
      frame.pack();
      frame.setVisible(true);
   }

   public static void main(String[] args) {
      javax.swing.SwingUtilities.invokeLater(new Runnable() {
         public void run() {
            createAndShowGUI();
         }
      });
   }
}
```

7.9 JMENUBAR, JMENU & JMENUITEM

Konstruktor dan method:

```
JMenuBar menuBar = new JMenuBar();
JMenu menu = new JMenu("A Menu");
JMenuItem menuItem = new JMenuItem("A text-only menu item",
                         KeyEvent.VK_T);

menuBar.add(menu);
menu.add(menuItem);
menu.addSeparator();
```

Method dari menu, menu bar dan menu item beserta fungsinya:

Method	Komponen	Fungsi
add()	MenuBar, Menu, MenuItem	Menambahkan komponen kepada komponen lain
setMnemonic()	MenuBar, Menu, MenuItem	
addSeparator()	MenuBar, Menu, MenuItem	Menambahkan separator

Contoh:

```
/**
 *
 * @author EJA
 */
public class SimpleMenu extends JFrame {

   private JMenuBar menuBar;
   private JMenu menu1, menu2;
   private JMenuItem kosong, isi;

   private JMenuBar buatMenuBar(){
      menuBar = new JMenuBar();

      menu1 = new JMenu("Pilihan Satu");
      menu2 = new JMenu("Pilihan Dua");
      menuBar.add(menu1); menuBar.add(menu2);

      kosong = new JMenuItem("Kosongkan");
      isi = new JMenuItem("Isi");
      menu1.add(kosong);menu1.add(isi);
      menu1.add(kosong);menu1.add(isi);

      return menuBar;
   }

   private static void createAndShowGUI() {
      //membuat frame
      SimpleMenu sm = new SimpleMenu();
      sm.setJMenuBar(sm.buatMenuBar());

      //menampilkan window
      sm.setSize(450, 260);
      sm.setVisible(true);
      sm.setDefaultCloseOperation(sm.EXIT_ON_CLOSE);
   }
   public static void main(String[] args) {
      createAndShowGUI();
   }
}
```

7.10 JTABLE & JSCROLLPANE

Konstruktor dan method:

```
JTable table = new JTable(new MyTableModel());
table.setFillsViewportHeight(true);
JScrollPane scrollPane = new JScrollPane(table);
add(scrollPane);
```

Konstruktor dari Jtable mempunyai parameter masukan dengan tipe data AbstractTableModel. Selain itu juga dapat mempunyai parameter masukan berupa array of Object sebagai row yang akan ditampilkan dan kolom.

```
JTable table = new JTable(rowData, columnNames)
```

Selain konstruktor dengan contoh di atas, terdapat juga bentuk konstruktor dengan parameter masukan lainnya, seperti vector atau jumlah baris dan kolom yang ditampilkan.

Method yang paling sering digunakan dan fungsinya:

Method	Fungsi
setModel(x)	Menge-set model dari JTable
setValueAt(Objek, baris, kolom)	Menge-set table dengan nilai Objek yang bertipe data Object pada baris dan kolom tertentu
getRowCount()	Mengembalikan jumlah baris yang ditampilkan di tabel
getSelectedColumn()	Mengambil indeks kolom tabel terpilih. Indeks dimulai dari 0
getSelectedRow()	Mengambil indeks baris tabel terpilih. Indeks dimulai dari 0
setPreferredScrollableViewportSize()	Menge-set ukuran viewport dari objek jtable
setFillsViewportHeight(true)	Menge-set apakah table “mengisi penuh” container tempat objek jtable berada.

Selain menggunakan objek dari table, manipulasi isi dan bentuk table juga bisa dilakukan via model jtable (lihat modul praktikum 10).

Contoh:

```
/**
 *
 * @edit by Eja
 */

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.table.AbstractTableModel;
import java.awt.Dimension;
import java.awt.GridLayout;

public class TableDemo extends JPanel {
   private boolean DEBUG = false;

   public TableDemo() {
      super(new GridLayout(1,0));

      JTable table = new JTable(new MyTableModel());
      table.setPreferredScrollableViewportSize(new Dimension(500, 70));
      table.setFillsViewportHeight(true);

      //Membuat scroll pane pada table
      JScrollPane scrollPane = new JScrollPane(table);

      //menambah scroll pane dan table di panel
      add(scrollPane);
   }

   class MyTableModel extends AbstractTableModel {
      private String[] columnNames = {"First Name",
                                      "Last Name",
                                      "Sport",
                                      "# of Years",
                                      "Vegetarian"};
      private Object[][] data = {
              {"Kathy", "Smith",
               "Snowboarding", new Integer(5), false},
              {"John", "Doe",
               "Rowing", new Integer(3), true},
              {"Sue", "Black",
               "Knitting", new Integer(2), false},
              {"Jane", "White",
```

```
            "Speed reading", new Integer(20), true},
           {"Joe", "Brown",
            "Pool", new Integer(10), false}
    };

    public int getColumnCount() {
       return columnNames.length;
    }

    public int getRowCount() {
       return data.length;
    }

    public String getColumnName(int col) {
       return columnNames[col];
    }

    public Object getValueAt(int row, int col) {
       return data[row][col];
    }

    /*
     * Dipergunakn untuk me-render bentuk table.
     * Jika tidak meng-implement method berikut,
     * kolom terakhir akan bernilai true/false saja
     * bukan berbentuk check box
     */
    @Override
    public Class getColumnClass(int c) {
       return getValueAt(0, c).getClass();
    }

    /*
     * Digunakan untuk mengubah table menjadi dapat diubah nilainya
     *
     */
    @Override
    public boolean isCellEditable(int row, int col) {
       //Hanya dapat mengubah isi table kolom 1 dan kolom 0
       if (col < 2) {
          return false;
       } else {
          return true;
       }
    }
```

```
    @Override
    public void setValueAt(Object value, int row, int col) {
      data[row][col] = value;
      fireTableCellUpdated(row, col);
    }
  }
  private static void createAndShowGUI() {
    //Membuat frame
    JFrame frame = new JFrame("TableDemo");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    //Cmembuat content pane
    TableDemo newContentPane = new TableDemo();
    newContentPane.setOpaque(true);
    frame.setContentPane(newContentPane);

    //menampilkan window
    frame.pack();
    frame.setVisible(true);
  }

  public static void main(String[] args) {
    javax.swing.SwingUtilities.invokeLater(new Runnable() {
      public void run() {
        createAndShowGUI();
      }
    });
  }
}
```

7.11 JTEXTFIELD

Konstruktor:

```
JTextField entry = new JTextField();
JTextField entry2 = new JTextField(25);
```

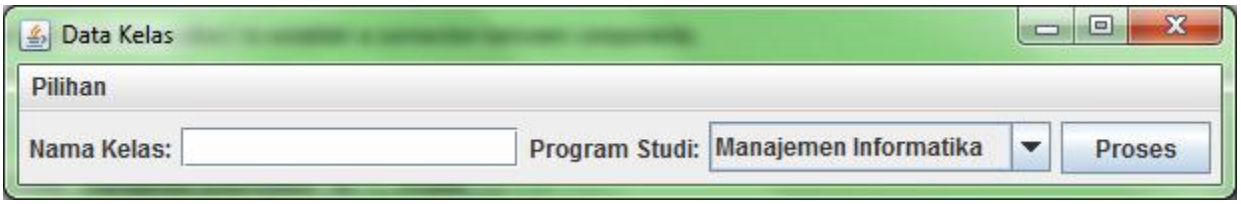
Method yang paling sering digunakan dan fungsinya:

Method	Fungsi
setText("x")	Memasukkan text pada komponen
getText()	Mengambil text dari komponen

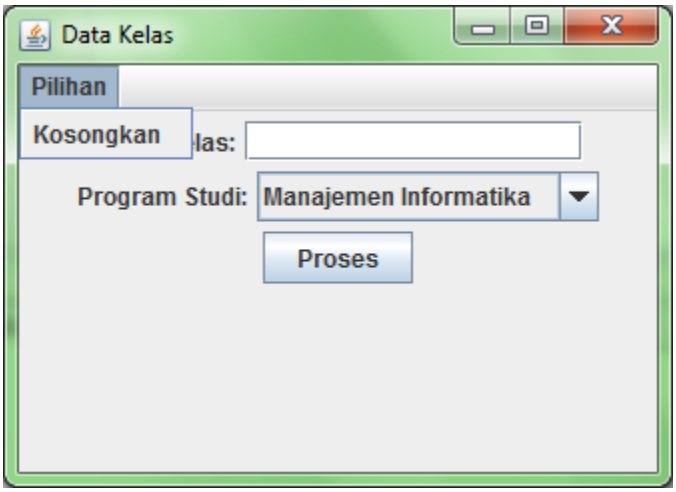
7.12 PRAKTIK

7.12.1 Soal I

Buatlah GUI sebagai berikut tanpa menggunakan gui builder:



Jika di-resize:



Pada menu "Pilihan" terdapat menu item "Kosongkan". Gunakan layout bawaan masing-masing container.

7.12.2 Solusi

Pertama, tentukan konsep GUI yang mau dibuat. Apakah kelas akan meng-extends JFrame atau meng-extends JPanel. Hal ini berpengaruh untuk melihat kelas tersebut bertugas sebagai container pane atau top-level-container.

Di penyelesaian ini, kelas-nya bertugas sebagai top-level-container, sedangkan JPanel sebagia container pane akan dideklarasikan dengan membuat objek barunya.

```
import javax.swing.JFrame;
/**
```

```
 *
 * @author EJA
 */
public class DataKelasGUI extends JFrame{

}
```

Deklarasikan semua komponen GUI yang ada. Perhatikan bahwa aplikasi ini memiliki komponen:

- 1 menu bar dengan 1 menu dan 1 menu item
- 2 label (labelKelas & labelProdi)
- 1 text field (teksKelas)
- 1 combo box (comboProdi)
- 1 button (tombolProses)

Sehingga, pada kelas dapat ditambahkan sebagai berikut:

```
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;
import javax.swing.JTextField;

/**
 *
 * @author EJA
 */
public class DataKelasGUI extends JFrame{

   private JLabel labelKelas, labelProdi;
   private JTextField teksKelas;
   private JComboBox comboProdi;
   private JButton tombolProses;
   private JMenuBar menuBar;
   private JMenu menu;
   private JMenuItem menuItem;

}
```

Karena turunan dari JFrame, jangan lupa untuk mengambil konstruktor JFrame untuk menambahkan title saat membuat objeknya:

```
public class DataKelasGUI extends JFrame{

   private JLabel labelKelas, labelProdi;
   private JTextField teksKelas;
   private JComboBox comboProdi;
   private JButton tombolProses;
   private JMenuBar menuBar;
   private JMenu menu;
   private JMenuItem menuItem;

   public DataKelasGUI(String judul){
      super(judul);
   }

}
```

Lalu, tambahkan 1 method “createAndShowGUI” yang akan membuat 1 objek dari kelas yang dimaksud yang sekaligus membuat objek dari JFrame sebagai top level container. Method ini bersifat private (agar tidak bisa diakses dari luar), dan static (agar method bisa diakses tanpa dibentuk object-nya).

```
public class DataKelasGUI extends JFrame{

   private JLabel labelKelas, labelProdi;
   private JTextField teksKelas;
   private JComboBox comboProdi;
   private JButton tombolProses;
   private JMenuBar menuBar;
   private JMenu menu;
   private JMenuItem menuItem;

   public DataKelasGUI(String judul){
      super(judul);
   }
   private static void createShowGUI(){
      DataKelasGUI dkg = new DataKelasGUI("Data Kelas");
      dkg.setDefaultCloseOperation(EXIT_ON_CLOSE);
      dkg.pack();
      dkg.setVisible(true);
   }
}
```

Perhatikan bahwa method tersebut membuat objek dari JFrame (dengan membuat objek dari kelas yang meng-extends JFrame), lalu menge-set apa yang harus dilakukan jika tombol close (x) ditekan (EXIT_ON_CLOSE maksudnya membuat aplikasi keluar dan di-terminate. Pilihan lain ada DISPOSE_ON_CLOSE, DO_NOTHING_ON_CLOSE dan HIDE_ON_CLOSE).

Setelah itu, frame di-pack. Maksud dari method pack() adalah frame tersebut akan dibuat ukurannya sesuai dengan method setPreferredSize() jika method ini dipanggil atau akan dibuat sesuai ukuran dari layout dan sub-komponen pada frame tersebut. Method setVIsible(), berfungsi untuk menampilkan frame tersebut ke layar.

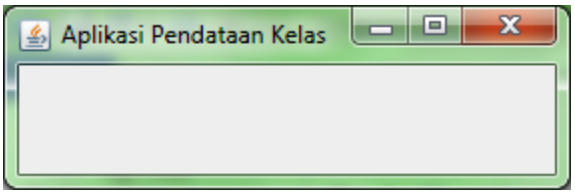
Setelah membentuk objek JFrame, pastikan aplikasi bisa dijalankan dengan membuat method main-nya.

```
public class DataKelasGUI extends JFrame{

   private JLabel labelKelas, labelProdi;
   private JTextField teksKelas;
   private JComboBox comboProdi;
   private JButton tombolProses;
   private JMenuBar menuBar;
   private JMenu menu;
   private JMenuItem menuItem;

   public DataKelasGUI(String judul){
      super(judul);
   }
   private static void createShowGUI(){
      DataKelasGUI dkg = new DataKelasGUI("Aplikasi Pendataan Kelas");
      dkg.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      dkg.pack();
      dkg.setVisible(true);
   }

   public static void main(String[] args) {
      javax.swing.SwingUtilities.invokeLater(new Runnable() {
         public void run() {
            createShowGUI();
         }
      });
   }
}
```

Perhatikan bahwa gui tersebut dijalankan dengan menggunakan thread. Hal ini sifatnya optional. Method main tersebut dapat dibuat sebagai berikut:

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

Jika di-run, maka akan keluar window sebagai berikut:



Frame masih kosong, sehingga perlu ditambahkan panel dan pada panel dapat ditambahkan komponen swing yang sudah dideklarasikan (label dll). Penambahan panel ini dapat dilakukan pada sebuah method. Method ini dinamai "addComponentsToPane".

```
public void addComponentsToPane(final Container pane){

}
```

Method di atas dipanggil pada main dengan cara:

```
private static void createShowGUI(){
   DataKelasGUI dkg = new DataKelasGUI("Data Kelas");
   dkg.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   dkg.addComponentsToPane(dkg.getContentPane());
   dkg.pack();
   dkg.setVisible(true);
}
```

Pada method "addComponentsToPane", flow layout dan panel dideklarasikan.

```
public void addComponentsToPane(final Container pane){
   FlowLayout tataLetak = new FlowLayout();
   final JPanel panelKomponen = new JPanel();
   panelKomponen.setLayout(tataLetak);
```

```
    tataLetak.setAlignment(FlowLayout.CENTER);
}
```

Tambahkan kode untuk mengatur peletakan komponen pada objek panel. Dimulai dari menu bar dan teman-temannya. Hanya saja, menu bar ini harus dimasukkan langsung di frame bukan melalui panel, jadi dibutuhkan 1 method yang akan mengembalikan menu bar tersebut. Method tersebut bernama "createMenuBar".

```
public JMenuBar createMenuBar() {
    menuBar = new JMenuBar();
    menu = new JMenu("Pilihan");
    menuBar.add(menu);

    menuItem = new JMenuItem("Kosongkan");
    menu.add(menuItem);
    return menuBar;
}
```

Hasil dari method ini akan digunakan di "createShowGUI" dengan cara memanggil method "setJMenuBar" dan parameter masukannya adalah return value dari method "createMenuBar":

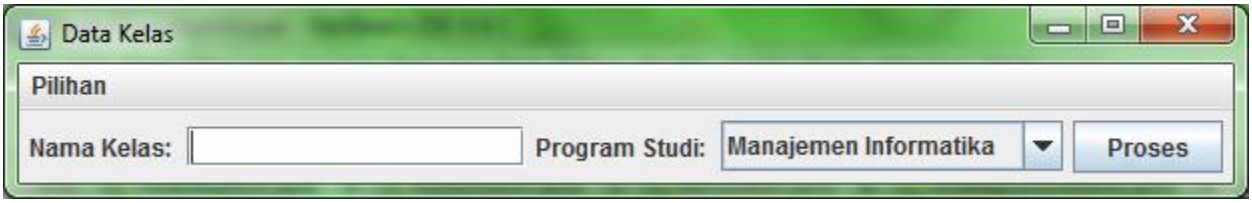
```
private static void createShowGUI() {
    DataKelasGUI dkg = new DataKelasGUI("Aplikasi Pendataan Kelas");
    dkg.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    dkg.setJMenuBar(dkg.createMenuBar());
    dkg.addComponentsToPane(dkg.getContentPane());
    dkg.pack();
    dkg.setVisible(true);
}
```

Jika dijalankan maka akan tampil sebagai berikut:

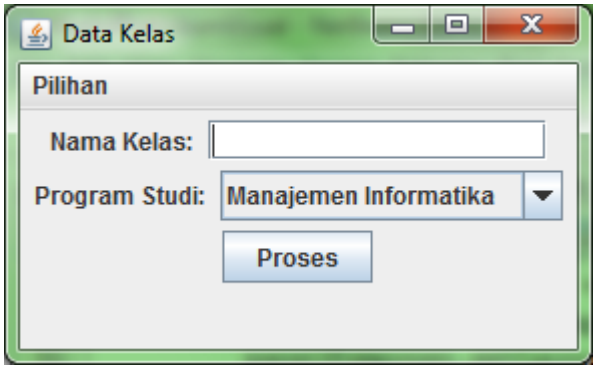
Tugas berikutnya adalah menambahkan komponen lain ke panel. Meneruskan method "addComponentsToPane":

```
public void addComponentsToPane(final Container pane) {
    FlowLayout tataLetak = new FlowLayout();
    final JPanel panelKomponen = new JPanel();
    panelKomponen.setLayout(tataLetak);
    tataLetak.setAlignment(FlowLayout.CENTER);

    labelKelas = new JLabel("Nama Kelas: ");
    panelKomponen.add(labelKelas);

    teksKelas = new JTextField(15);
    panelKomponen.add(teksKelas);

    labelProdi = new JLabel("Program Studi: ");
    panelKomponen.add(labelProdi);

    String daftarPilihan[] = {"Manajemen Informatika",
                "Komputerisasi Akuntansi",
                "Teknik Komputer"};
    comboProdi = new JComboBox(daftarPilihan);
    panelKomponen.add(comboProdi);

    tombolProses = new JButton("Proses");
    panelKomponen.add(tombolProses);

    pane.add(panelKomponen);
  }
```

Jika dijalankan, hasilnya:



Dan karena sifatnya yang “flow”, maka ketika di-resize ke ukuran yang tepat, maka tampilannya adalah sebagai berikut:



7.13 LATIHAN

Buatlah sebuah aplikasi gui sebagai berikut tanpa menggunakan bantuan matisse builder atau visual gui-editor lainnya:

8 BAB VIII LISTENER/EVENT HANDLER

8.1 IDENTITAS

Kajian

Komponen Swing Java non-Visual Editor; Database & Swing

Topik

1. Event Listener: ActionListener, MouseListener
2. Inner Class

Referensi

1. http://docs.oracle.com/javase/tutorial/uiswing/index.html

Kompetensi Utama

1. Mahasiswa memahami penggunaan komponen swing untuk membuat aplikasi desktop.
2. Mahasiswa mampu membuat aplikasi desktop menggunakan komponen swing—sesuai yang dituliskan di topik—tanpa bantuan visual editor.
3. Mahasiswa mampu menggunakan interface listener (ActionListener dan MouseListener) untuk menambahkan event handler pada komponen swing yang telah dibuat.

Lama Kegiatan Praktikum

1. Kegiatan Mandiri : 1 x 100 menit

Parameter Penilaian

1. Tugas Pendahuluan
2. Jurnal Mandiri

8.2 PENGENALAN EVENT HANDLER

Aplikasi GUI merupakan aplikasi berbasis "event". Komponen seperti tombol dan text field akan menunggu aksi dari keyboard ataupun mouse. Aksi tersebut merupakan event terhadap komponen GUI. Langkah pertama yang dilakukan untuk menambahkan event tersebut adalah dengan meng-import paket "awt.event.*" Walaupun komponen yang digunakan merupakan komponen swing, untuk event, paket awt.event tetap yang harus di-import.

Langkah berikutnya adalah meng-implements "xxxListener". Daftarkan event listener untuk widget/komponen yang dikenai event dengan menggunakan method "addxxxListener(this)". Keyword this merupakan representasi objek dari class yang memiliki handler method seperti "xxxPerformed()".

Yang perlu diperhatikan, interface listener harus di-implements oleh kelas atau innerclass yang digunakan untuk menuliskan event dari komponen. Dan penulisannya biasanya bersifat private untuk secure implementation. Daftar dari listener (yang umum digunakan) adalah sebagai berikut:

Interface	Method	Penggunaan
ActionListener	actionPerformed()	Klik Tombol dan lain-lain
AdjustmentListener	adjustmentValueChanged()	Posisi pergerakan scroll bar
ChangeListener	stateChanged()	Pergerakan slider
ComponentListener	componentResized()	Perubahan ukuran komponen
ContainerListener	componentAdded() componentRemoved()	Penambahan dan pengurangan komponen
FocusListener	focusGained() focusLost()	Focus pada text field
ItemListener	itemStateChanged()	Perubahan state pada checkbox
KeyListener	keyPressed() keyReleased() keyTyped()	Penggunaan keyboard
MouseListener	mouseClicked()	Klik mouse
MouseMotionListener	mouseDragged() mouseMoved()	Pergerakan mouse
MouseWheelListener	mouseWheelMoved()	Pergerakan mouse wheel
TextListener	textValueChanged()	Perubahan konten teks
WindowListener	windowActivated() windowClosed() windowClosing() windowDeactivated()	Perubahan state window

Setiap listener merupakan interface yang harus dituliskan kembali setiap methodnya walaupun method tersebut tidak akan digunakan. Untuk menghindari hal ini, terdapat suatu adapter class untuk setiap listener. Adapter class akan meng-implements interface dan menuliskan kembali method pada interface tanpa body methodnya. Class bentukan hanya perlu meng-extends kelas adapter dan meng-override method yang diperlukan.

Contoh perbandingan penggunaan (atas—listener; bawah—adapter):

```
//An example that implements a listener interface directly.
public class MyClass implements MouseListener {
    ...
        someObject.addMouseListener(this);
    ...
    /* Empty method definition. */
    public void mousePressed(MouseEvent e) {
    }

    /* Empty method definition. */
    public void mouseReleased(MouseEvent e) {
    }

    /* Empty method definition. */
    public void mouseEntered(MouseEvent e) {
    }

    /* Empty method definition. */
    public void mouseExited(MouseEvent e) {
    }

    public void mouseClicked(MouseEvent e) {
        ...//Event listener implementation goes here...
    }
}
```

```
/*
 * An example of extending an adapter class instead of
 * directly implementing a listener interface.
 */
public class MyClass extends MouseAdapter {
    ...
        someObject.addMouseListener(this);
    ...
    public void mouseClicked(MouseEvent e) {
        ...//Event listener implementation goes here...
    }
}
```

Jika ingin menggunakan class adapter tapi class tersebut tidak ingin meng-extends class adapter secara langsung, konsep innerclass bisa digunakan. Berikut contohnya:

```
//An example of using an inner class.
public class MyClass extends Applet {
    ...
        someObject.addMouseListener(new MyAdapter());
    ...
    class MyAdapter extends MouseAdapter {
        public void mouseClicked(MouseEvent e) {
            ...//Event listener implementation goes here...
        }
    }
}
```

Contoh di atas, terdapat sebuah class yang harus meng-extends Applet sekaligus MouseAdapter. Hal ini pastinya tidak bisa dilakukan. Dikarenakan MouseAdapter hanya bertindak sebagai event dan tidak memiliki peranan lain pada kelas utama, event tersebut dapat dipindahkan kepada satu class lain yang terdapat pada class utama. Dengan kondisi class lain tersebut lah yang meng-extends MouseAdapter.

Variasi dari inner class adalah menggunakan anonymous inner class. Hal ini seolah2 menjadikan suatu kelas yang dibentuk objeknya menjadi parameter masukan untuk suatu method. Sama dengan contoh di atas, hanya saja contoh di atas, class memiliki nama “MyAdapter” dan terdapat pada blok program yang berbeda. Sedangkan anonymous inner class sebaliknya. Contoh:

```
//An example of using an anonymous inner class.
public class MyClass extends Applet {
    ...
        someObject.addMouseListener(new MouseAdapter() {
            public void mouseClicked(MouseEvent e) {
                ...//Event listener implementation goes here...
            }
        });
    ...
    }
}
```

Tidak semua Listener memiliki Adapter. Daftarnya yang paling sering digunakan adalah sebagai berikut:

Listener Interface	Adapter Class	Listener Methods
`ActionListener`	-	`actionPerformed(ActionEvent)`
`ChangeListener`	-	`stateChanged(ChangeEvent)`
`FocusListener`	`FocusAdapter`	`focusGained(FocusEvent)` `focusLost(FocusEvent)`
`ItemListener`	-	`itemStateChanged(ItemEvent)`

`KeyListener`	`KeyAdapter`	`keyPressed(KeyEvent)`
		`keyReleased(KeyEvent)`
		`keyTyped(KeyEvent)`

`MenuListener`	-	`menuCanceled(MenuEvent)`
		`menuDeselected(MenuEvent)`
		`menuSelected(MenuEvent)`

`MouseListener`	`MouseAdapter`,
	`MouseInputAdapter`
		`mouseClicked(MouseEvent)`
		`mouseEntered(MouseEvent)`
		`mouseExited(MouseEvent)`
		`mousePressed(MouseEvent)`
		`mouseReleased(MouseEvent)`

`TableModelListener`	-	`tableChanged(TableModelEvent)`

Setiap komponen swing memerlukan interface listener yang berbeda-beda. Untuk ComponentListener, FocusListener, KeyListener, MouseListener, MouseMotionListener, MouseWheelListener, HierarchyBoundsListener dan HierarchyListener dapat diaplikasikan ke semua komponen swing

Tapi ada beberapa listener yang hanya support beberapa komponen saja karena event tersebut tidak berasal dari awt container. Berikut daftar komponen umum yang sering digunakan dengan interface yang dapat diaplikasikan pada komponen tersebut:

Komponen	Listener
Button	ActionListener, ChangeListener, ItemListener
Check Box	ActionListener, ChangeListener, ItemListener
Combo Box	ActionListener, ItemListener
Menu Item	ActionListener, ChangeListener, ItemListener
Radio Button	ActionListener, ChangeListener, ItemListener
Table	ListSelectionListener
Text field	ActionListener, CaretListener, DocumentListener
Text area	CaretListener, DocumentListener
Tabbed Pane	ChangeListener

Umumnya, untuk button dan turunannya, digunakan interface ActionListener dengan method "actionPerformed(ActionEvent e)". Bahkan untuk menu, ActionListener juga umum digunakan.

Terdapat 2 cara umum yang digunakan untuk menambahkan listener pada komponen. Cara pertama seperti yang digunakan oleh netbean, yaitu memberikan listener dengan mendeklarasikan method performed yang akan digunakan:

```
    jButton1.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton1ActionPerformed(evt);
        }
    });
```

```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }
```

Dengan menggunakan cara di atas, class yang digunakan tidak perlu mengimplements ActionListener. Dalam kasus Netbeans gui builder terlihat bahwa kelas yang digunakan untuk merancang dan menampilkan gui hanya meng-extends JFrame tanpa meng-implements listener apapun.

Cara kedua yang umum digunakan dengan meng-implements listener terkait (tergantung komponen yang akan diberikan event handler) selain meng-extends class yang digunakan sebagai container (bisa secondary ataupun primary container). Penentuan komponen yang dikenai event handler tergantung dengan command yang diberikan. Contoh:

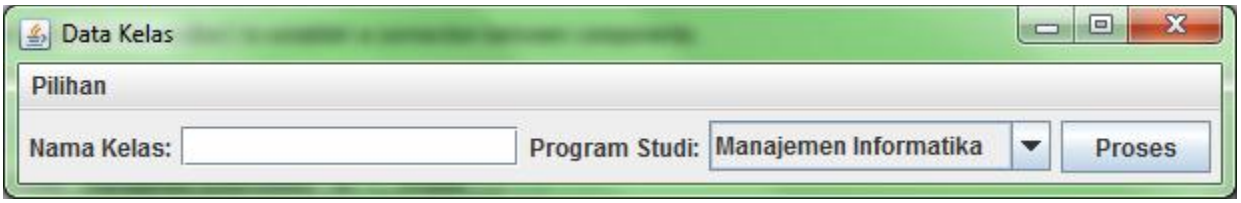
```
public class SimpleButtonDemo extends JPanel implements ActionListener {
    protected JButton b1;

    public SimpleButtonDemo() {

        b1 = new JButton("Tombol 1");
        b1.setActionCommand("lakukan");

        //Tambahkan action listener untuk button
        b1.addActionListener(this);
    }

    public void actionPerformed(ActionEvent e) {
        if ("lakukan".equals(e.getActionCommand())) {
```

```
        System.out.println("Tombol Ditekan");
      }
    }
}
```

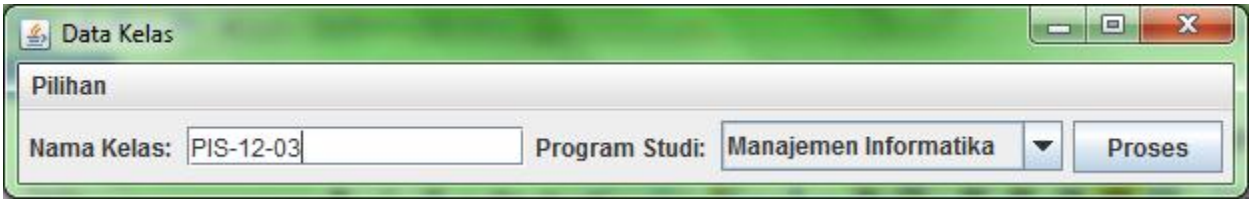
8.3 PRAKTIK

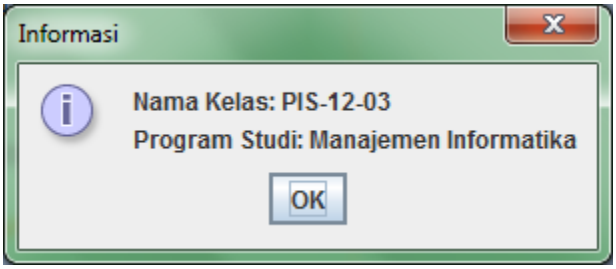
8.3.1 Soal

Buatlah GUI sebagai berikut tanpa menggunakan gui builder:



Pada menu "Pilihan" terdapat menu item "Kosongkan". Gunakan layout bawaan masing-masing container. Tambahkan listener, sehingga saat menekan menu "Kosongkan", semua field menjadi kosong dan saat menekan tombol "Proses", keluar option pane yang menampilkan teks yang dimasukkan ke text field serta informasi prodi yang dipilih. Contoh:





8.3.2 Solusi

Terdapat 2 cara untuk menambahkan event listener ke sebuah kelas GUI, dengan menggabungkan di kelas yang sama dengan kelas GUI, atau membuat kelas tersendiri khusus untuk melakukan handler.

Cara termudah adalah dengan menggabungkan antara GUI dengan action-nya.

Untuk menjalankan solusi ini, pada kelas GUI yang telah dibuat sebelumnya di modul 7, tambahkan interface listener yang diinginkan di kelas tersebut. Dalam hal ini, kita tambahkan listener ActionListener. Setelah menambahkan interface ini, maka method actionPerformed harus di-override, sehingga perubahan dari kelas sebelumnya terlihat sebagai berikut:

```
public class DataKelasGUIHandler extends JFrame implements ActionListener{
…
   public void actionPerformed(ActionEvent e) {

   }
}
```

Tuliskan logika pemrograman di dalam method actionPerformed tersebut. Method ini dijalankan ketika ada suatu aksi di aplikasi GUI, dengan catatan komponen yang sudah dikenai aksi telah dikenali oleh listener (dilakukan dengan memanggil method “addActionListener()” pada objek komponen terkait). Hal ini akan menyulitkan jika ada beberapa tombol yang harus ditekan, beberapa menu item yang bisa di-klik, beberapa radio button yang dapat dipilih dan semua aksi yang dilakukan.

Untuk membedakan setiap aksi yang terjadi, pada komponen terkait, bisa ditambahkan “setActionCommand()” yang nantinya di actionPerformed diambil dengan method “getActionCommand()”.

Untuk membuktikan bahwa actionPerformed bersifat universal, tambahkan listener untuk tombol dan menu item:

```
  public JMenuBar createMenuBar() {
     menuBar = new JMenuBar();
     menu = new JMenu("Pilihan");
     menuBar.add(menu);

     menuItem = new JMenuItem("Kosongkan");
     menu.add(menuItem);
     menuItem.addActionListener(this); //perhatikan baris ini
     return menuBar;
  }

  public void addComponentsToPane(final Container pane) {
     FlowLayout tataLetak = new FlowLayout();
     final JPanel panelKomponen = new JPanel();
     panelKomponen.setLayout(tataLetak);
     tataLetak.setAlignment(FlowLayout.CENTER);

     labelKelas = new JLabel("Nama Kelas: ");
     panelKomponen.add(labelKelas);

     teksKelas = new JTextField(15);
     panelKomponen.add(teksKelas);

     labelProdi = new JLabel("Program Studi: ");
     panelKomponen.add(labelProdi);

     String daftarPilihan[] = {"Manajemen Informatika",
                       "Komputerisasi Akuntansi",
                       "Teknik Komputer"};
     comboProdi = new JComboBox(daftarPilihan);
     panelKomponen.add(comboProdi);

     tombolProses = new JButton("Proses");
     panelKomponen.add(tombolProses);
     tombolProses.addActionListener(this); //perhatikan baris ini

     pane.add(panelKomponen);
  }
```

Lalu ubah actionPerformed menjadi sebagai berikut:

```
public void actionPerformed(ActionEvent e) {
   System.out.println("Hello World");
}
```

Perhatikan ketika dijalankan, saat tombol ditekan dan menu item dipilih, aksi yang dilakukan sama. Untuk membedakan, tambahkan method “setActionCommand()” untuk setiap komponen terkait.

```
public JMenuBar createMenuBar() {
   menuBar = new JMenuBar();
   menu = new JMenu("Pilihan");
   menuBar.add(menu);

   menuItem = new JMenuItem("Kosongkan");
   menu.add(menuItem);
   menuItem.setActionCommand("kosong"); //perhatikan baris ini
   menuItem.addActionListener(this); //perhatikan baris ini
   return menuBar;
}
```

```
public void addComponentsToPane(final Container pane) {
   …
   tombolProses = new JButton("Proses");
   panelKomponen.add(tombolProses);
   tombolProses.setActionCommand("tombol"); //perhatikan baris ini
   tombolProses.addActionListener(this); //perhatikan baris ini
   …
}
```

Parameter untuk method setActionCommand bersifat bebas, selama masih bertipe String. String inilah nantinya yang akan di-filter oleh action performed dengan method “getActionCommand()” sehingga dapat dibedakan aksi dari button atau aksi dari menu item.

```
public void actionPerformed(ActionEvent e) {
   if(e.getActionCommand().equalsIgnoreCase("kosong")){
      //lakukan aksi jika menu item ditekan
   }else if(e.getActionCommand().equals("tombol")){
      //lakukan aksi jika tombol ditekan
   }
}
```

Untuk mem-filternya, dapat digunakan perbandingan string, yaitu equalsIgnoreCase dan equals. Perbedaannya, untuk equalsIgnoreCase, huruf besar dan huruf kecil tidak dianggap berbeda, sedangkan equals sebaliknya. “kosong” dan “tombol” disesuaikan dengan action command yang diberikan ke masing-masing komponen.

Untuk mengeluarkan option pane saat menekan tombol, ubah method actionPerformed sebagai berikut:

```
public void actionPerformed(ActionEvent e) {
    if (e.getActionCommand().equalsIgnoreCase("kosong")) {

    } else if (e.getActionCommand().equals("tombol")) {
        //ambil teks nama kelas dari field teks
        String namaKelas = teksKelas.getText();
        //ambil item dari combo box
        String prodi = (String) comboProdi.getSelectedItem();
        //atur string yang akan ditampilkan
        String pesan = "Nama Kelas: " + namaKelas + "\n"
                + "Program Studi: " + prodi;
        //tampilkan dengan option pane
        JOptionPane.showMessageDialog(null, pesan,
                "Informasi", JOptionPane.INFORMATION_MESSAGE);
    }
}
```

Untuk mengosongkan sesuai tampilan default, tambahkan satu method bernama kosong yang akan dipanggil di actionPerformed:

```
public void kosong(){
    teksKelas.setText("");
    comboProdi.setSelectedIndex(0);
}
```

Panggil method tersebut di actionPerformed:

```
public void actionPerformed(ActionEvent e) {
    if (e.getActionCommand().equalsIgnoreCase("kosong")) {
        kosong();
    } else if (e.getActionCommand().equals("tombol")) {
        ……
    }
}
```

Pastikan setiap aksi berjalan semestinya.

9 BAB IX LAYOUTING

9.1 IDENTITAS

Kajian

Komponen Swing Java non-Visual Editor; Database & Swing

Topik

1. Layout—Grid, GridBag dan Border

Referensi

1. http://docs.oracle.com/javase/tutorial/uiswing/layout/index

Kompetensi Utama

1. Mahasiswa memahami konsep layout pada pemrograman swing
2. Mahasiswa mampu membuat halaman sederhana menggunakan komponen swing dan mengaturnya menggunakan layout tertentu (grid, gridbag atau border) tanpa dibantu tool gui builder

Lama Kegiatan Praktikum

1. Kegiatan Mandiri : 1 x 100 menit

Parameter Penilaian

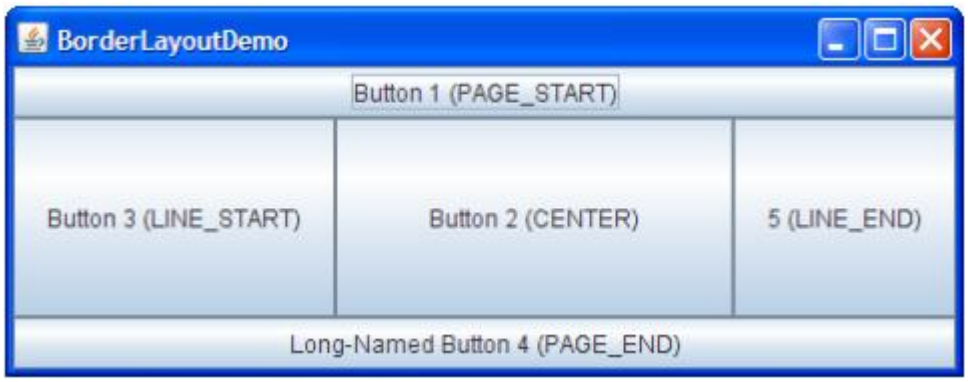
1. Tugas Pendahuluan
2. Jurnal Mandiri

9.2 BORDER LAYOUT

Layout yang memungkinkan komponen hanya dapat diletakkan di 5 area saja:

- PAGE_START
- PAGE_END
- LINE_START
- LINE_END
- CENTER

Ilustrasi:



Contoh Penggunaan:

```
/**
 *
 * edit By Eja
 */
import javax.swing.*;
import java.awt.BorderLayout;
import java.awt.Container;
import java.awt.Dimension;

public class BorderLayoutDemo {

   public static void addComponentsToPane(Container pane) {

      JButton button = new JButton("Button 1 (PAGE_START)");
      pane.add(button, BorderLayout.PAGE_START);

      //Make the center component big, since that's the
      //typical usage of BorderLayout.
      button = new JButton("Button 2 (CENTER)");
      button.setPreferredSize(new Dimension(200, 100));
      pane.add(button, BorderLayout.CENTER);
```

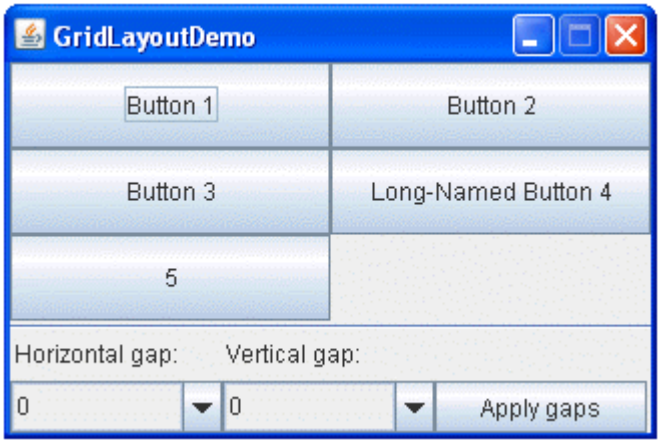
```
        button = new JButton("Button 3 (LINE_START)");
        pane.add(button, BorderLayout.LINE_START);

        button = new JButton("Long-Named Button 4 (PAGE_END)");
        pane.add(button, BorderLayout.PAGE_END);

        button = new JButton("5 (LINE_END)");
        pane.add(button, BorderLayout.LINE_END);

    }
    private static void createAndShowGUI() {

        //membuat frame
        JFrame frame = new JFrame("BorderLayoutDemo");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        //tambahkan pane
        addComponentsToPane(frame.getContentPane());
        //tampilkan frame
        frame.pack();
        frame.setVisible(true);
    }

    public static void main(String[] args) {

        javax.swing.SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                createAndShowGUI();
            }
        });
    }
}
```

Perhatikan pada contoh di atas bahwa border layout tidak dibentuk objeknya sama sekali. Hal ini karena pane yang digunakan (JFrame) menggunakan default layout BorderLayout, jadi komponen bisa langsung ditambahkan dengan pengaturan manajemen layout "BorderLayout".

9.3 GRID LAYOUT

Grid layout menempatkan objek komponen berdasarkan grid cell. Dengan setiap cell memiliki ukuran yang sama.

Ilustrasi:



Terdapat beberapa method yang digunakan pada layout ini:

Method	Fungsi
setRows(3)	Menge-set jumlah maksimal baris dari peletakan komponen.
setColumns(4)	Menge-set jumlah maksimal kolom dari peletakan komponen. Orientasi setRows lebih besar dibandingkan setColumns.
setHgap(30)	Menge-set jarak antar komponen secara horizontal
setVgap(30)	Menge-set jarak antar komponen secara vertikal

Jumlah baris dan kolom maksimum pada layout juga dapat di-set via parameter konstruktor. Parameter pertama adalah nilai baris maksimal, parameter kedua merupakan pengaturan nilai kolom maksimal.

Contoh Penggunaan:

```
/**
 *
 * Edit By Eja
 */
import java.awt.*;
import javax.swing.*;

public class GridLayoutDemo extends JFrame {
```

```
final static int maxGap = 20;
//deklarasi layout dengan baris tidak ditentukan dan kolom maksimal 2
GridLayout experimentLayout = new GridLayout(0, 2);

public GridLayoutDemo(String name) {
   super(name);
}

public void addComponentsToPane(final Container pane) {
   final JPanel compsToExperiment = new JPanel();
   compsToExperiment.setLayout(experimentLayout);

   //tambahkan tombol ke panel
   compsToExperiment.add(new JButton("Button 1"));
   compsToExperiment.add(new JButton("Button 2"));
   compsToExperiment.add(new JButton("Button 3"));
   compsToExperiment.add(new JButton("Long-Named Button 4"));
   compsToExperiment.add(new JButton("5"));
   //atur jarak vertical dan horizontal-nya
   experimentLayout.setHgap(20);
   experimentLayout.setVgap(5);

   //mengatur letak panel pada objek JFrame
   pane.add(compsToExperiment, BorderLayout.NORTH);
}

private static void createAndShowGUI() {
   //membuat objek frame
   GridLayoutDemo frame = new GridLayoutDemo("GridLayoutDemo");
   frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   //memanggil method buatan addComponentsToPane()
   frame.addComponentsToPane(frame.getContentPane());
   //menampilkan window
   frame.pack();
   frame.setVisible(true);
}

public static void main(String[] args) {
   javax.swing.SwingUtilities.invokeLater(new Runnable() {

      public void run() {
         createAndShowGUI();
      }
   });
}
```

```
}
```

9.4 GRIDBAG LAYOUT

Merupakan layout yang paling sering digunakan programmer java untuk pengaturan peletakan komponen objek swing karena fleksibilitas yang ditawarkan. Sama seperti grid layout, membagi peletakan dalam grid cells. Hanya saja ukuran dari setiap komponen dapat berbeda. Contohnya, saat melakukan peletakan komponen, komponen tersebut dapat memakai 2 grid secara horizontal dan 3 grid vertical.

Ilustrasi:

Contoh Penggunaan:

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

public class GridBagLayoutDemo {
   final static boolean shouldFill = true;
   final static boolean shouldWeightX = true;
   final static boolean RIGHT_TO_LEFT = false;

   public static void addComponentsToPane(Container pane) {
      if (RIGHT_TO_LEFT) {
         pane.setComponentOrientation(ComponentOrientation.RIGHT_TO_LEFT);
      }

      JButton button;
           pane.setLayout(new GridBagLayout());
           GridBagConstraints c = new GridBagConstraints();
           if (shouldFill) {
           //natural height, maximum width
           c.fill = GridBagConstraints.HORIZONTAL;
           }
```

```
        button = new JButton("Button 1");
        if (shouldWeightX) {
        c.weightx = 0.5;
        }
        c.fill = GridBagConstraints.HORIZONTAL;
        c.gridx = 0;
        c.gridy = 0;
        pane.add(button, c);

        button = new JButton("Button 2");
        c.fill = GridBagConstraints.HORIZONTAL;
        c.weightx = 0.5;
        c.gridx = 1;
        c.gridy = 0;
        pane.add(button, c);

        button = new JButton("Button 3");
        c.fill = GridBagConstraints.HORIZONTAL;
        c.weightx = 0.5;
        c.gridx = 2;
        c.gridy = 0;
        pane.add(button, c);

        button = new JButton("Long-Named Button 4");
        c.fill = GridBagConstraints.HORIZONTAL;
        c.ipady = 40;      //make this component tall
        c.weightx = 0.0;
        c.gridwidth = 3;
        c.gridx = 0;
        c.gridy = 1;
        pane.add(button, c);

        button = new JButton("5");
        c.fill = GridBagConstraints.HORIZONTAL;
        c.ipady = 0;       //reset to default
        c.weighty = 1.0;   //request any extra vertical space
        c.anchor = GridBagConstraints.PAGE_END; //bottom of space
        c.insets = new Insets(10,0,0,0);  //top padding
        c.gridx = 1;       //aligned with button 2
        c.gridwidth = 2;   //2 columns wide
        c.gridy = 2;       //third row
        pane.add(button, c);
    }

    private static void createAndShowGUI() {
```

```
        //Membuat Frame
        JFrame frame = new JFrame("GridBagLayoutDemo");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        //memanggil method addComponentsToPane
        //untuk membuat panel dan menambahkan komponen
        addComponentsToPane(frame.getContentPane());

        //Menampilkan frame
        frame.pack();
        frame.setVisible(true);
    }

    public static void main(String[] args) {
        javax.swing.SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                createAndShowGUI();
            }
        });
    }
}
```

Keterangan Method:

Method/Atribut	Fungsi
gridx, gridy	Menentukan baris dan kolom pada sudut kiri atas komponen. Koordinat paling kiri memiliki nilai gridx = 0; dan paling atas punya nilai gridy = 0; Gunakan GridBagConstraints.RELATIVE untuk penempatan sesuai pendeklarasian komponen terhadap layout
gridwidth, gridheight	Menentukan lebar atau panjang komponen terhadap layout. Nilai defaultnya adalah satu
ipadx, ipady	Menentukan besarnya komponen. Hampir sama dengan gridwidth/gridheight, hanya saja atribut ini tetap terhadap 1 nilai cell (padding).
insets	Menentukan space komponen dengan komponen atas, bawah, samping kiri dan samping kanan (spacing)
anchor	Memungkingkan peletakan komponen di koordinat tertentu layaknya border layout. FIRST_LINE_START PAGE_START FIRST_LINE_END LINE_START CENTER LINE_END LAST_LINE_START PAGE_END LAST_LINE_END Digunakan jika space content pane lebih besar dari keseluruhan komponen

9.5 LATIHAN

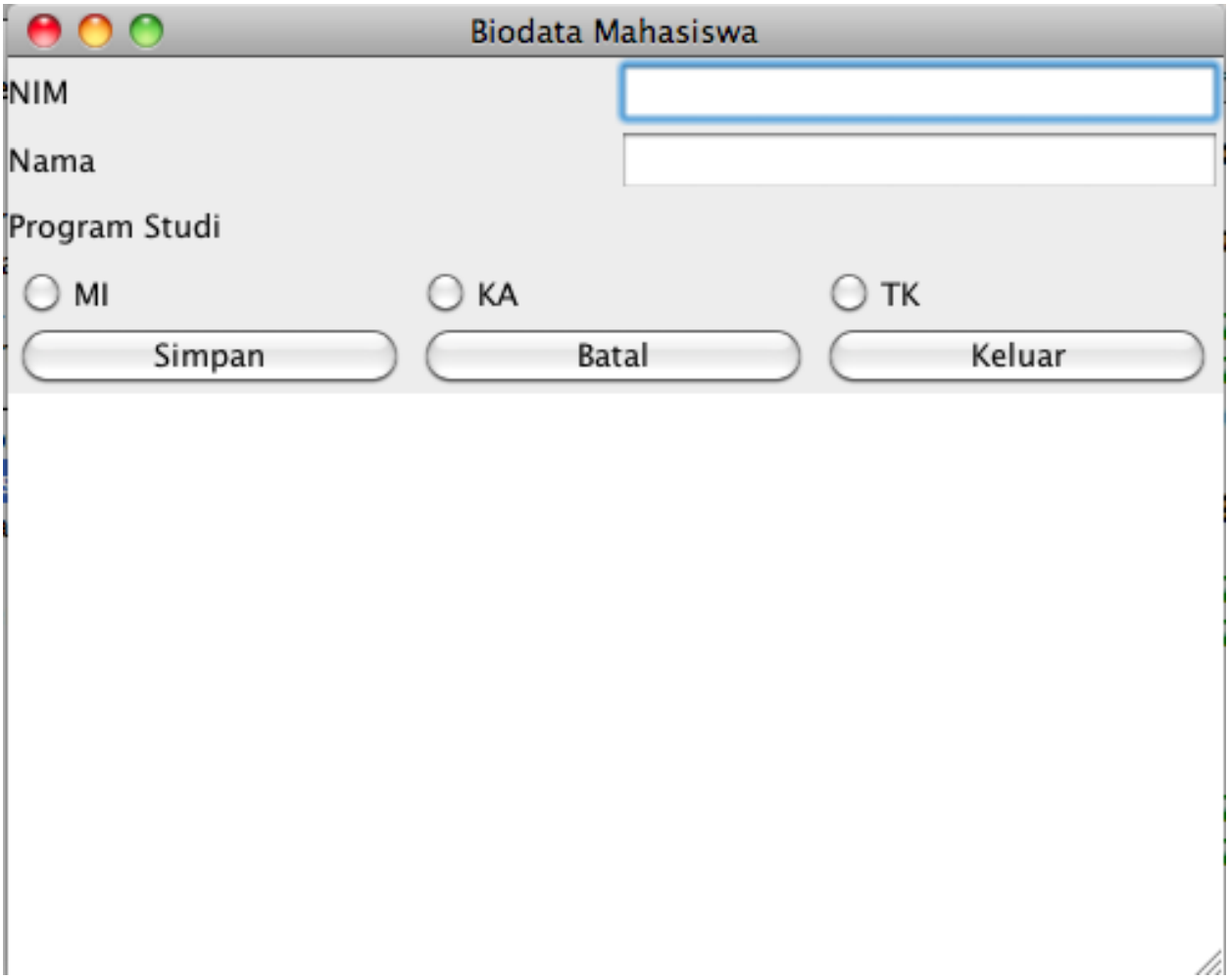
9.5.1 Soal

Buatlah sebuah tampilan program sebagai berikut menggunakan kombinasi layout flow (default layout dari JPanel), Grid, GridBag atau Border. Jumlah dan jenis layout (dari yang telah disebutkan) untuk digunakan tidak ditentukan.

Berikan Action untuk setiap pemilihan combo box. Setiap combo box terpilih, maka text field akan berisi nilai dari combo box.

9.5.2 Soal

Buatlah sebuah tampilan GUI sebagai berikut:

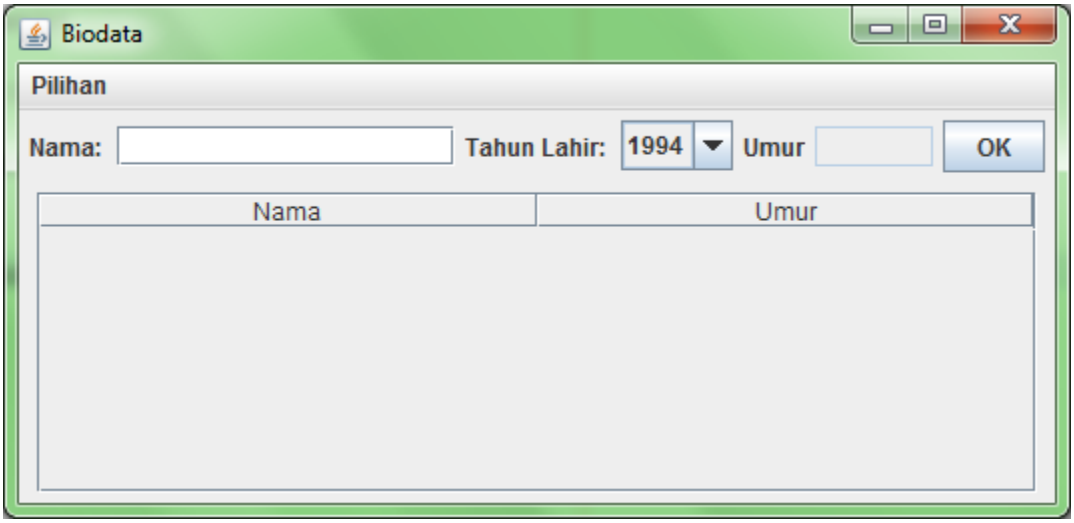


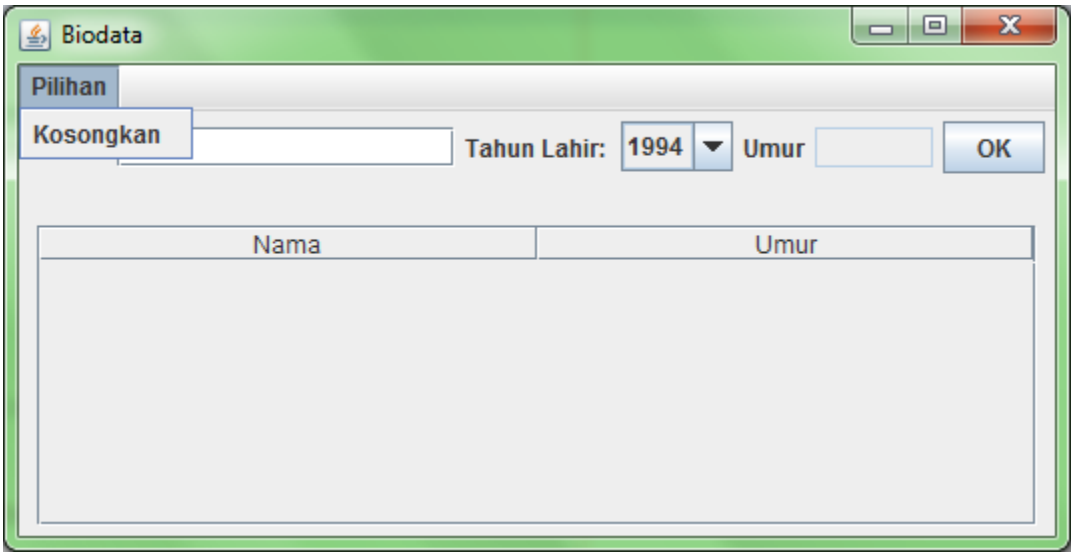
Buatlah GUI dengan event handler dan petunjuk sebagai berikut:

a) Pada saat pengguna menekan tombol Simpan, maka nilai dari NIM, Nama, Pilihan Program Studi akan tersimpan/tampil di TextArea
b) Pada saat pengguna menekal tombol Batal, maka nilai dari NIM, Nama, Pilihan Program Studi akan kosong
c) Pada saat pengguna menekan tombol Keluar, maka form Biodata Mahasiswa akan tertutup

9.5.3 Soal

Buatlah sebuah form gui sebagai berikut tanpa menggunakan bantuan gui builder.





Keterangan:

- Gui tidak terhubung dengan database
- Merupakan Class java yang meng-extends “JFrame” dan meng-implements “ActionListener”
- Frame memiliki 2 panel, panel atas (center) dan panel bawah (south)
    - Isi panel atas
        - 3 label
        - 2 textfield
        - 1 combobox
        - 1 button
        - Layout yang digunakan adalah layout default (flow layout)
    - Isi panel bawah
        - 1 Tabel

10 BAB X GUI-DATABASE

10.1 IDENTITAS

Kajian

Memahami pembuatan komponen swing (native-way) dan pengaksesan database (MySQL) dengan bahasa pemrograman java

Topik

1. Menghubungkan antara GUI dan database
2. Menampilkan data database pada JTable
3. Menampilkan data database pada JComboBox

Referensi

1. http://docs.oracle.com/javase/tutorial/uiswing/index.html
2. http://docs.oracle.com/javase/tutorial/jdbc/index.html

Kompetensi Utama

1. Mampu membuat aplikasi berbasis GUI untuk memanipulasi basis data.
2. Mampu mem-populate isi untuk komponen swing GUI dengan isi dari database.
3. Mampu melakukan CRUD database via aplikasi swing java

Lama Kegiatan Praktikum

1. Kegiatan Mandiri : 2 x 100 menit

Parameter Penilaian

1. Tugas Pendahuluan
2. Jurnal Mandiri

10.2 PRAKTIK

10.2.1 Insert Database—Form GUI

Dengan memanfaatkan modul pengaksesan database yang telah ada sebelumnya, Input ke GUI dengan menerima input user melalui text field, proses menjadi lebih gampang untuk dilakukan.

Kasus: Input ke table DesaNinja yang terdiri dari atribut id_desa, nama, pemimpin.

Urutannya:

a) Buat 3 textfield di aplikasi gui dan 1 tombol “simpan”
b) Ambil input user dengan method getText. Simpan input tersebut ke beberapa variabel.
c) Buat 1 objek dari clas DesaNinja dengan parameter konstruktor adalah variabel yang telah dimiliki sebelumnya.
d) Panggil method yang merepresentasikan method save() atau method untuk menyimpan data tersebut.
e) Tampilkan pesan “Berhasil” jika input user berhasil dimasukkan ke database; tampilkan gagal jika gagal

Secara detil: Langkah pertama, buat kelas “KoneksiDB” seperti yang tercantum di modul 5.

```
//Langkah pertama, import package terkait
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;


/**
 *
 * @author Eja
 */
public class KoneksiDB {
   // driver JDBC driver dan database URL

   private final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
   private final String DB_URL = "jdbc:mysql://localhost/pis12noltiga";
   //  Database credentials
   private final String USER = "root";
   private final String PASS = "";
   private Connection conn = null;
```

```
    public void bukaKoneksi() {

        boolean flag = false;
        try {
            //Langkah ke-2: Registrasi JDBC
            Class.forName(JDBC_DRIVER);
        } catch (Exception e) {
            System.out.println(e.getMessage());
            flag = true;
        }
        if (!flag) {
            try {
                //Langkah ke-3: buka koneksi
                conn = DriverManager.getConnection(DB_URL, USER, PASS);
            } catch (SQLException e) {
                System.out.println(e.getMessage());
            }
        }
    }

    public Connection getConn() {
        return conn;
    }
}
```

Buatlah Kelas DesaNinja (seperti yang tercantum di modul 5/6—Fokus hanya ke method untuk menyimpan data saja).

```
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

/**
 *
 * @author Eja
 */
public class DesaNinja {

    private String id_desa;
    private String nama;
    private String pemimpin;
    //ambil objek KoneksiDB karena membutuhkan connection dll
    private KoneksiDB kdb = new KoneksiDB();
```

```
    public DesaNinja(String id_desa, String desa, String pemimpin) {
        this.id_desa = id_desa;
        this.nama = desa;
        this.pemimpin = pemimpin;
    }
    public boolean masukkanData() throws SQLException {
        //deklarasi connection dan preparedStatement
        Connection dbConnection = null;
        PreparedStatement ps = null;
        int rowAffect = 0;

        String insertTableSQL = "INSERT INTO desa_ninja"
                + "(id_desa, nama, pemimpin) VALUES"
                + "(?,?,?)";
        try {
            //buka koneksi saat objek dari desa ninja dibentuk
            kdb.bukaKoneksi();
            //inisialisasi dbConnection dari objek Connection
            dbConnection = kdb.getConn();

            //Langkah ke 4 bagian 1
            ps = dbConnection.prepareStatement(insertTableSQL);
            ps.setString(1, this.id_desa);
            ps.setString(2, this.nama);
            ps.setString(3, this.pemimpin);
            //langkah 4: eksekusi query
            rowAffect = ps.executeUpdate();

        } catch (Exception e) {
            System.out.println(e.getMessage());
        } finally {
            //langkah ke 6
            ps.close();
        }
        //langkah ke 5
        if (rowAffect > 0) {
            return true;
        } else {
            return false;
        }
    }
}
```

Kelas untuk swing java:

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextField;

/**
 *
 * @author Eja
 */
public class SimpleInsertGui extends JPanel implements ActionListener {

   //3 textfield di deklarasikan
   private JTextField id, nama, pemimpin;
   private JButton eksekusi;

   public SimpleInsertGui() {
      //langkah a, buat 3 textfield (beserta label) dan 1 tombol
      id = new JTextField(20);
      nama = new JTextField(20);
      pemimpin = new JTextField(20);

      //buat label untuk tiap textfield:
      JLabel l1 = new JLabel("No Registrasi Desa: ");
      JLabel l2 = new JLabel("Nama: ");
      JLabel l3 = new JLabel("Pemimpin: ");

      eksekusi = new JButton("Eksekusi");
      eksekusi.setActionCommand("oke");
      eksekusi.addActionListener(this);

      //tambahkan label, text dan button ke panel
      add(l1);add(id);
      add(l2);add(nama);
      add(l3);add(pemimpin);
      add(eksekusi);
   }
```

```
public void actionPerformed(ActionEvent e) {
   if(e.getActionCommand().equals("oke")){
      //langkah b, ambil semua isian textfield
      String n = nama.getText();
      String i = id.getText();
      String p = pemimpin.getText();

      //cek kalau isian kosong
      if (n.isEmpty() || i.isEmpty() || p.isEmpty()) {
         JOptionPane.showMessageDialog(this, "Ada Field yang Kosong",
               "Peringatan",JOptionPane.WARNING_MESSAGE);
      }else{
         //langkah c, buat objek desa Ninja
         DesaNinja dn = new DesaNinja(i, n, p);
         boolean status = false;
         try {
            //langkah d, panggil method yang merepresentasikan 'save()'
            status = dn.masukkanData();
         } catch (SQLException ex) {
            Logger.getLogger(SimpleInsertGui.class.getName()).log(Level.SEVERE, null, ex);
         }
         if(status){
           //langkah e, tampilkan pesan berhasil jika eksekusi berhasil
            JOptionPane.showMessageDialog(this, "Eksekusi Berhasil",
                  "Status",JOptionPane.INFORMATION_MESSAGE);
         }else{
           //langkah e, tampilkan pesan gagal jika eksekusi gagal
            JOptionPane.showMessageDialog(this, "Eksekusi Gagal",
                  "Status",JOptionPane.ERROR_MESSAGE);
         }
      }
   }
}

private static void createAndShowGUI() {

   //membuat frame
   JFrame frame = new JFrame("ButtonDemo");
   frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

   //membuat content pane
   SimpleInsertGui newContentPane = new SimpleInsertGui();
   newContentPane.setOpaque(true);
   frame.setContentPane(newContentPane);
```

```
        //Memunculkan window
        frame.pack();
        frame.setVisible(true);
        frame.setLocationRelativeTo(null);
    }
    public static void main(String[] args) {

        javax.swing.SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                createAndShowGUI();
            }
        });
    }
}
```

10.2.2 Select Database—JTable

Mengisi tabel dengan data dari database mempunyai banyak cara. Salah satunya yang dituliskan pada modul ini, dengan tetap memanfaatkan kelas dari modul praktikum sebelumnya (modul 5/6).

Langkah-langkah:

a) Gunakan Kelas "KoneksiDB" untuk membuka koneksi dan lain-lain
b) Deklarasikan JTable dengan modelnya pada kelas yang digunakan untuk penempatan swing.
c) Ikuti langkah pengaksesan data, kecuali langkah ke-5 harus dimodifikasi. Tuliskan method pengaksesan data ini di kelas yang sama dengan kelas untuk menuliskan swing (boleh diletakkan berbeda, tapi yang dituliskan di modul ini adalah cara yang cukup sederhana).
d) Langkah ke-5 dimodifikasi dengan tidak menampilkan hasil ResultSet tapi memasukkannya ke dalam model JTable.

Lengkapnya:

```
/**
 *
 * @author Eja
 */

import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import java.awt.Dimension;
```

```
import java.awt.GridLayout;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.table.DefaultTableModel;

public class SimpleTableDemoDB extends JPanel {
   //langkah b
   private DefaultTableModel model;
   private JTable table;

   //langkah a
   private KoneksiDB kdb = new KoneksiDB();

   public SimpleTableDemoDB() {
      super(new GridLayout(1,0));

      model = new DefaultTableModel();
      table = new JTable(model);
      table.setPreferredScrollableViewportSize(new Dimension(500, 70));
      table.setFillsViewportHeight(true);

      //Membuat scroll pane pada table
      JScrollPane scrollPane = new JScrollPane(table);
      //menambah scroll pane dan table di panel
      add(scrollPane);
      try {
         isiDataTabel();
      } catch (SQLException ex) {
         Logger.getLogger(SimpleTableDemoDB.class.getName()).log(Level.SEVERE, null, ex);
      }
   }

   //langkah c
   private void isiDataTabel() throws SQLException {
      Connection dbConnection = null;
      PreparedStatement ps = null;
      ResultSet rs = null;

      String viewTableSQL = "SELECT * FROM desa_ninja";

      try {
         //buka koneksi saat objek dari desa ninja dibentuk
         kdb.bukaKoneksi();
         //inisialisasi dbConnection dari objek Connection
```

```
        dbConnection = kdb.getConn();

        //Langkah ke 4 bagian 1
        ps = dbConnection.prepareStatement(viewTableSQL);

        //langkah 4: eksekusi query
        rs = ps.executeQuery();

        //tentukan header tabel
        model.addColumn("Nomor Registrasi Desa");
        model.addColumn("Nama Desa");
        model.addColumn("Pemimpin");

        //langkah ke 5
        //ekstrak data dari ResultSet dan masukkan ke table
        while (rs.next()) {
           Object[] o = new Object[3];
           //langkah d
           o[0] = rs.getString(1);
           o[1] = rs.getString(2);
           o[2] = rs.getString(3);
           model.addRow(o);
        }

     } catch (Exception e) {
        System.out.println(e.getMessage());
     } finally {
        //langkah ke-6
        ps.close();
     }
  }

  private static void createAndShowGUI() {
     //Membuat frame
     JFrame frame = new JFrame("TableDemo");
     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

     //Membuat content pane
     SimpleTableDemoDB newContentPane = new SimpleTableDemoDB();
     newContentPane.setOpaque(true);
     frame.setContentPane(newContentPane);

     //menampilkan window
     frame.pack();
     frame.setVisible(true);
  }
```

```
  public static void main(String[] args) {
    javax.swing.SwingUtilities.invokeLater(new Runnable() {
      public void run() {
        createAndShowGUI();
      }
    });
  }
}
```

10.2.3 Select Database—JComboBox

Mengisi combo box dengan data yang berasal dari database, sama seperti mengisi data dari database untuk diisikan ke table. Ada banyak cara yang bisa dilakukan. Urutan langkah pengerjaan yang dilakukan oleh modul praktikum ini adalah sebagai berikut:

a) Gunakan Kelas "KoneksiDB" untuk membuka koneksi dan lain-lain
b) Deklarasikan JComboBox pada kelas yang digunakan untuk penempatan swing.
c) Ikuti langkah pengaksesan data, kecuali langkah ke-5 harus dimodifikasi. Tuliskan method pengaksesan data ini di kelas yang sama dengan kelas untuk menuliskan swing (alasan yang sama dengan sebelumnya).
d) Panggil method pengaksesan data tersebut pada konstruktor, sehingga saat ditampilkan pertama kali, combo box sudah menampilkan isi dari database.
e) Langkah ke-5 dimodifikasi dengan tidak menampilkan hasil ResultSet tapi memasukkannya ke dalam komponen JComboBox.

Lengkapnya sebagai berikut:

```
/**
 *
 * @author Eja
 */
import java.util.logging.Level;
import java.util.logging.Logger;
import java.awt.event.*;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.*;

public class SimpleComboBoxDemo extends JPanel implements ActionListener {
```

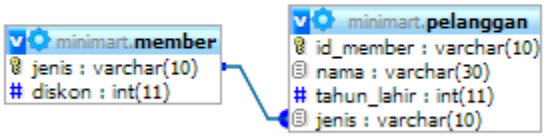
```
//langkah a
private KoneksiDB kdb = new KoneksiDB();
//langkah b
private JComboBox kombo;

public SimpleComboBoxDemo() {
   super();
   kombo = new JComboBox();

   try {
      //langkah d
      this.isiDataKombo();
   } catch (SQLException ex) {
      Logger.getLogger(SimpleComboBoxDemo.class.getName()).log(Level.SEVERE, null, ex);
   }
   kombo.addActionListener(this);

   add(kombo);
}

/** Listener combo box */
public void actionPerformed(ActionEvent e) {
   JComboBox cb = (JComboBox) e.getSource();
   int pil = cb.getSelectedIndex();
   if (pil == 0) {
      System.out.println("Tidak ada yang terpilih");
   } else {
      System.out.println(cb.getSelectedItem());
   }
}
//langkah c
private void isiDataKombo() throws SQLException {
   Connection dbConnection = null;
   PreparedStatement ps = null;
   ResultSet rs = null;

   String viewTableSQL = "SELECT * FROM desa_ninja";
   try {
      //buka koneksi saat objek dari desa ninja dibentuk
      kdb.bukaKoneksi();
      //inisialisasi dbConnection dari objek Connection
      dbConnection = kdb.getConn();

      //Langkah ke 4 bagian 1
      ps = dbConnection.prepareStatement(viewTableSQL);
```

```
        //langkah 4: eksekusi query
        rs = ps.executeQuery();

        //langkah ke 5
        //ekstrak data dari ResultSet

        while (rs.next()) {
          //langkah e
          kombo.addItem(rs.getString(2));
        }
        kombo.setSelectedIndex(0);

      } catch (Exception e) {
        System.out.println(e.getMessage());
      } finally {
        //langkah ke-6
        ps.close();
      }
    }
    private static void createAndShowGUI() {
      //membuat frame
      JFrame frame = new JFrame("ComboBoxDemo");
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      //membuat content pane
      JComponent newContentPane = new SimpleComboBoxDemo();
      newContentPane.setOpaque(true); //content panes must be opaque
      frame.setContentPane(newContentPane);

      //Memunculkan window
      frame.pack();
      frame.setVisible(true);
    }

    public static void main(String[] args) {
      javax.swing.SwingUtilities.invokeLater(new Runnable() {
        public void run() {
          createAndShowGUI();
        }
      });
    }
}
```

10.3 LATIHAN

Diketahui sebuah tabel dengan struktur dan isi sebagai berikut:

Struktur:



Isi:

EDIT	JENIS	DISKON
	gold	2
	platinum	10
		row(s) 1 - 2 of 2

EDIT	ID_MEMBER	NAMA	TAHUN_LAHIR	JENIS
	MEM-001	Melody Nurramdhani Laksani	1992	gold
	MEM-002	Sonya Pandarmawan	1996	gold
	MEM-003	Nabilah Ratna Ayu Azalia	1999	platinum
				row(s) 1 - 3 of 3

Query DDL dan DML untuk tabel di atas adalah sebagai berikut:

Oracle:

```
CREATE table "MEMBER" (
   "JENIS"     VARCHAR2(10) NOT NULL,
   "DISKON"    NUMBER,
   constraint  "MEMBER_PK" primary key ("JENIS")
)
/

CREATE table "PELANGGAN" (
   "ID_MEMBER"   VARCHAR2(10),
   "NAMA"       VARCHAR2(30),
   "TAHUN_LAHIR" NUMBER,
   "JENIS"      VARCHAR2(10),
   constraint  "PELANGGAN_PK" primary key ("ID_MEMBER")
)
/

ALTER TABLE "PELANGGAN" ADD CONSTRAINT "PELANGGAN_FK"
FOREIGN KEY ("JENIS")
REFERENCES "MEMBER" ("JENIS")
ON DELETE CASCADE

/
```

MySQL:

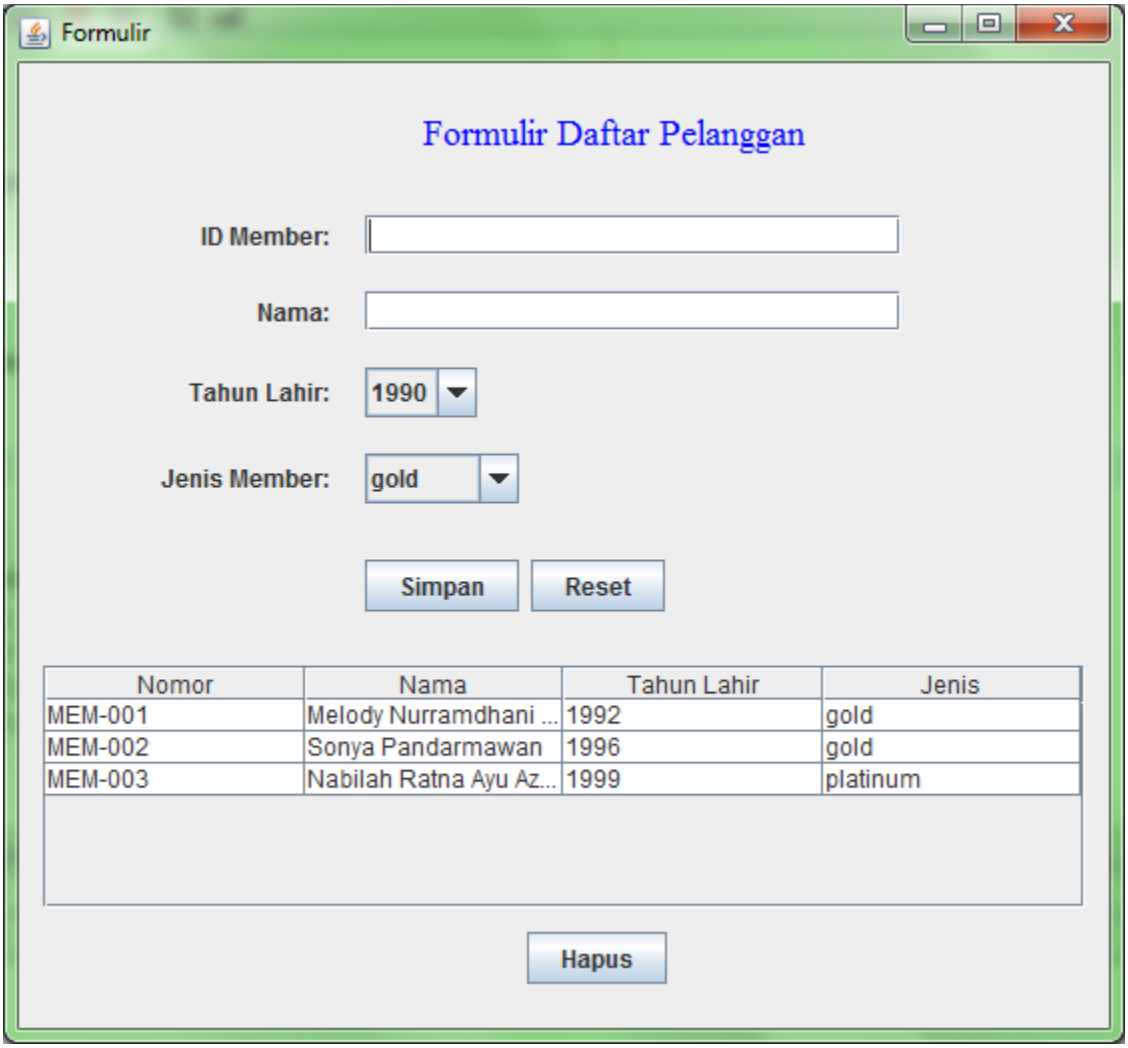
```
CREATE TABLE IF NOT EXISTS `member` (
  `jenis` varchar(10) NOT NULL,
  `diskon` int(11) NOT NULL,
  PRIMARY KEY (`jenis`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `pelanggan` (
  `id_member` varchar(10) NOT NULL,
  `nama` varchar(30) NOT NULL,
  `tahun_lahir` int(11) NOT NULL,
  `jenis` varchar(10) NOT NULL,
  PRIMARY KEY (`id_member`),
  KEY `jenis` (`jenis`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

ALTER TABLE `pelanggan`
  ADD CONSTRAINT `pelanggan_ibfk_1` FOREIGN KEY (`jenis`) REFERENCES `member` (`jenis`) ON DELETE CASCADE ON UPDATE CASCADE;
```

Isi:

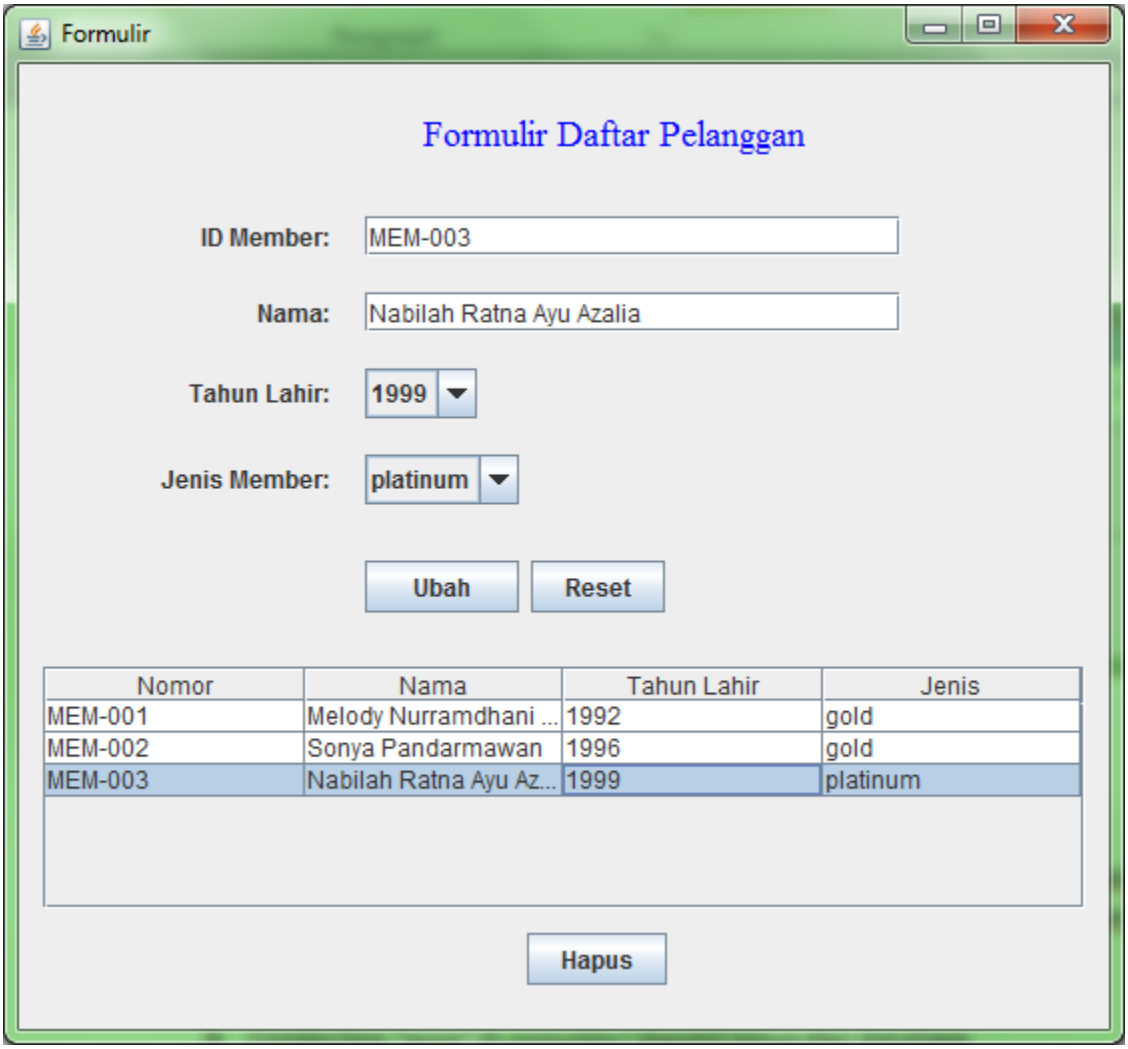
```
INSERT INTO `minimart`.`member` (`jenis`, `diskon`) VALUES ('gold', '2');
INSERT INTO `minimart`.`member` (`jenis`, `diskon`) VALUES ('platinum', '10');

INSERT INTO pelanggan (id_member, nama, tahun_lahir, jenis) VALUES ('MEM-001', Melody Nurramdhani Laksani', 1992, 'gold');

INSERT INTO pelanggan (id_member, nama, tahun_lahir, jenis) VALUES ('MEM-002', 'Sonya Pandarmawan', 1996, 'gold');

INSERT INTO pelanggan (id_member, nama, tahun_lahir, jenis) VALUES ('MEM-003', 'Nabilah Ratna Ayu Azalia', 1999, 'platinum');
```

Tugas:

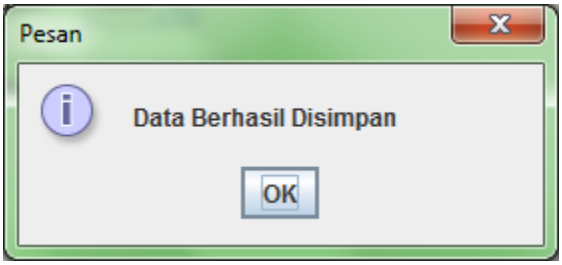
Buatlah sebuah form untuk memasukkan data pelanggan ke database (pilih salah satu Antara oracle/mysql) sebagai berikut:

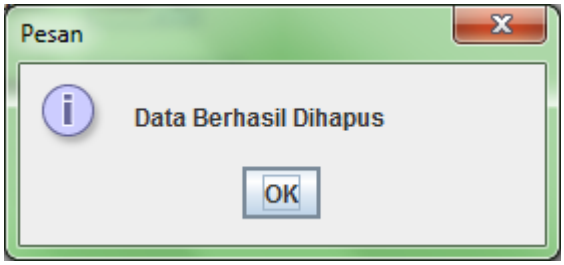


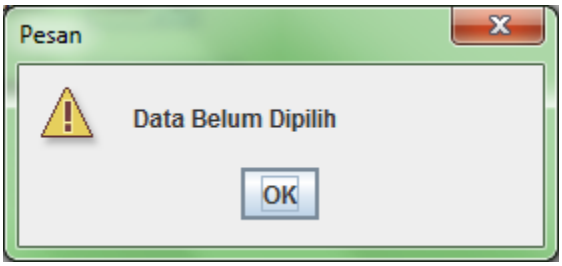
Ketika tabel dipilih:



Pesan yang dimunculkan:







Keterangan:

- Ketika tabel dipilih, setiap textfield berisikan data sesuai baris terpilih
- Tombol “Simpan” berubah menjadi “Ubah” ketika data di tabel dipilih.
- Tabel tidak bisa diubah nilainya dengan meng-klik baris tabel 2x, tapi masih bisa dipilih.
- Tombol “Reset” akan mengosongkan setiap textfield dan mengembalikan tombol ke state “Simpan”
- Setelah menekan tombol “Simpan” atau “Ubah”, kembali ke state seperti habis di-reset
- Combo box “tahun lahir” berisi tahun 1990-2000
- Combo box “jenis” di-populate/ diambil isinya dari database
- Boleh menggunakan GUI-Builder (Matisse Builder)

11 DAFTAR PUSTAKA

[1] Netbeans, [Online]. Available: https://netbeans.org.

[2] Oracle, [Online]. Available: http://docs.oracle.com/javase/tutorial/jdbc/index.html .

[3] Oracle, [Online]. Available: http://docs.oracle.com/javase/tutorial/uiswing/index.html .

[4] Oracle. [Online]. Available: http://docs.oracle.com/javase/tutorial/uiswing/layout/index .