2011

Membuat Aplikasi Sederhana Menggunakan Java

Eko Kurniawan Khannedy

StripBandunk

9/1/2011

Membuat Aplikasi Sederhana Menggunakan Java

Pada buku ini, kita akan mencoba membuat sebuah aplikasi sederhana menggunakan Java yang menggunakan sistem basis data **MySQL**. Programnya sederhana, hanya membuat sebuah Form dengan fasilitas CRUD (Create, Read, Update dan Delete).

Tabel yang akan kita buat sekarang adalah tabel MAHASISWA, dimana tabel tersebut memiliki beberapa kolom, yaitu NIM, NAMA, TANGGAL_LAHIR, JURUSAN dan ALAMAT.

Membuat Database

Sebelum membuat program sederhana ini, hal yang pertama perlu kita buat adalah database yang akan kita gunakan. Misal kita akan membuat database UNIVERSITAS, maka kita bisa menggunakan perintah :

```
CREATE DATABASE UNIVERSITAS;
```

Membuat Tabel

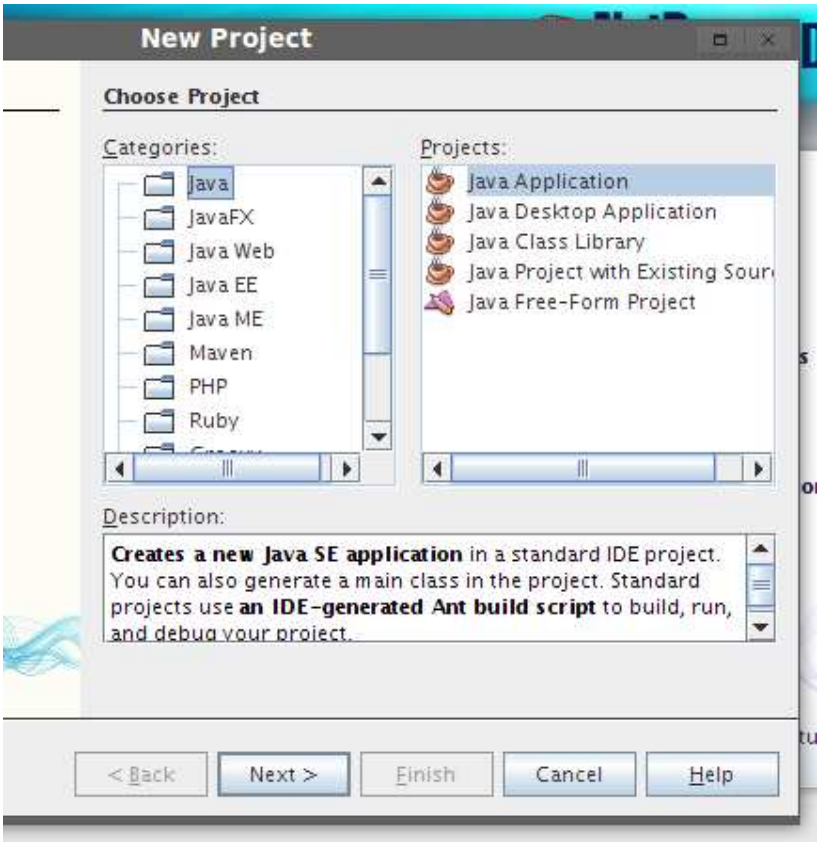
Setelah membuat database, kita terlebih dahulu perlu membuat tabel MAHASISWA. Kita dapat menggunakan perintah sebagai berikut untuk membuat tabel MAHASISWA :

```
CREATE TABLE MAHASISWA(
   NIM VARCHAR(8) PRIMARY KEY,
   NAMA VARCHAR(50) NOT NULL,
   TANGGAL_LAHIR DATE NOT NULL,
   JURUSAN VARCHAR(50) NOT NULL,
   ALAMAT VARCHAR(500) NOT NULL
);
```

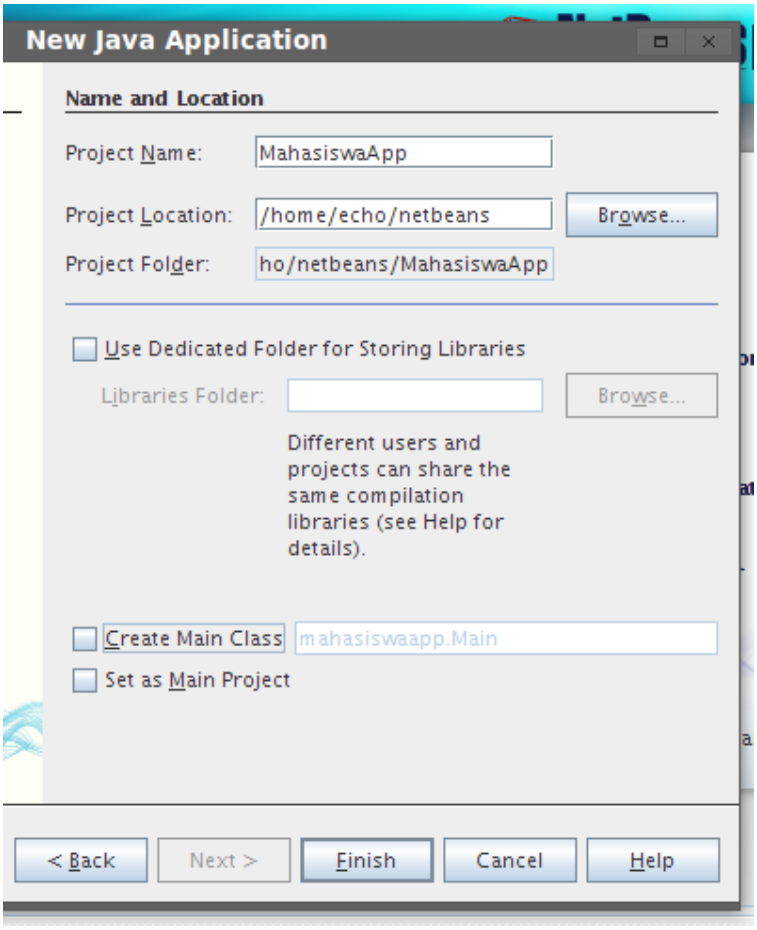
Sekarang kita sudah punya sebuah tabel dengan nama MAHASISWA. Saatnya kita lanjutkan membuat project Java-nya menggunakan NetBeans IDE.

Membuat Project

Sama seperti sebelumnya, untuk membuat sebuah project dalam NetBeans IDE kita dapat membuatnya menggunakan menu File -> New Project. Setelah itu pilih kategori Java dan pilih tipe project-nya Java Application.



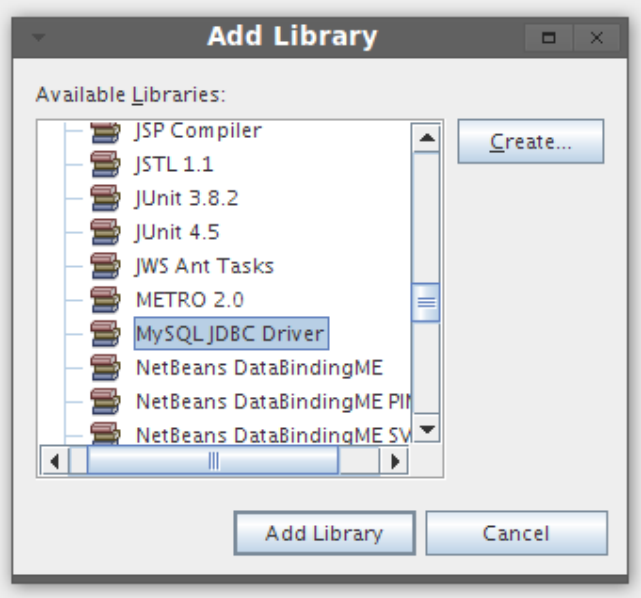
Klik Next untuk melanjutkan pembuatan project.



Beri nama project dan jangan diceklis checkbox Create Main Class. Hal ini dikarenakan kita tidak memerlukan dahulu membuat sebuah Main Class. Setelah itu klik tombol Finish, sekarang kita telah membuat project Java menggunakan NetBeans IDE.

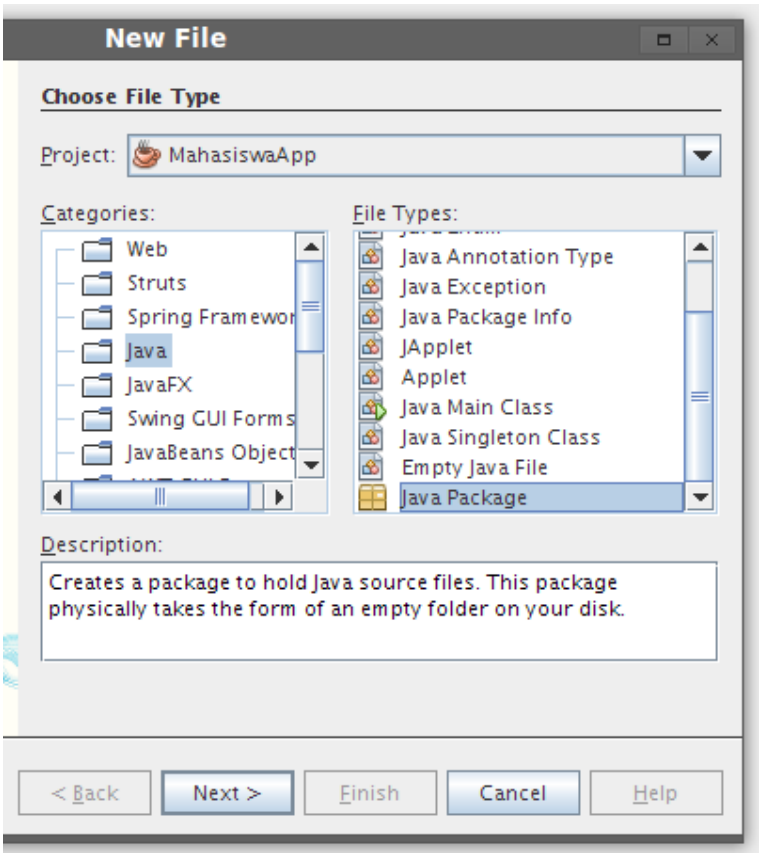
Membuat Koneksi MySQL

Setelah membuat project, saatnya membuat koneksi ke database UNIVERSITAS yang telah kita buat sebelumnya. Jadi hal yang pertama kitalakukan adalah menambah driver MySQL ke dalam project yang telah kita buat. Caranya klik kanan bagian Libraries project yang telah kita buat lalu pilih Add Library.

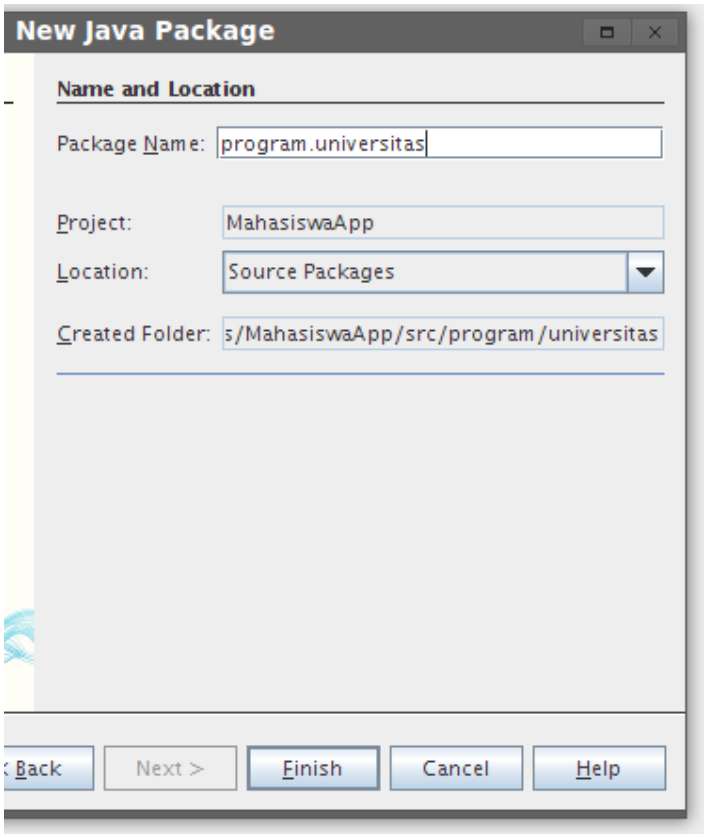


Setelah keluar dialog Add Library, pilih MySQL JDBC Driver lalu klik tombol Add Library untuk menambahkan MySQL Driver kedalam project.

Setelah menambahkan driver MySQL, sekarang saatnya membuat sebuah kelas yang akan kita gunakan untuk melakukan koneksi ke database MySQL. Tapi sebelum membuat sebuah kelas, pastikan kita membuat package dulu,caranya klik kanan bagian Source project yang telah kita buat lalu pilih New -> Other.

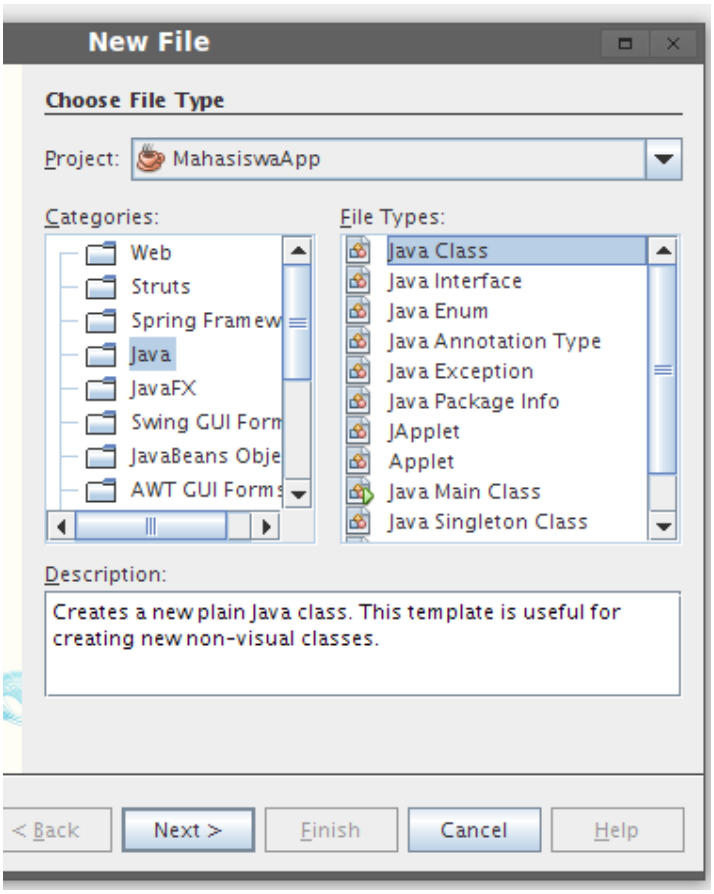


Setelah keluar dialog New File, pilih kategori Java dan jenis file Java Package. Klik Next untuk melanjutkan membuat package.

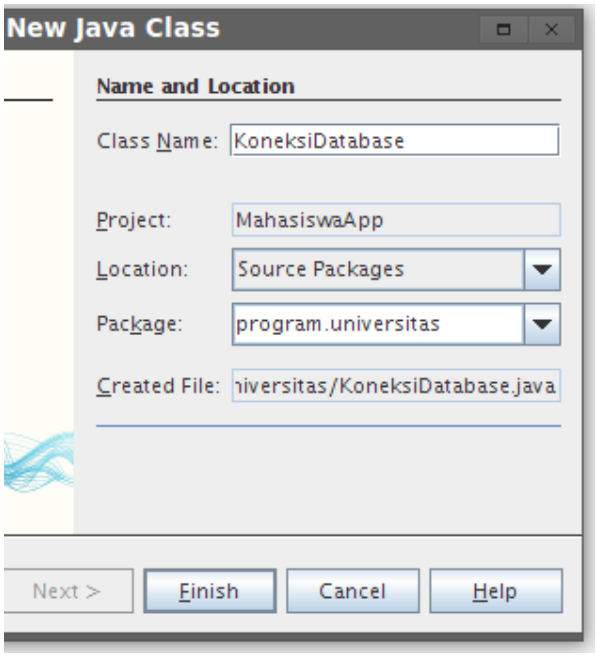


Setelah itu beri nama packagenya, misal **program.universitas**, setelah itu klik Finish untuk membuat package-nya.

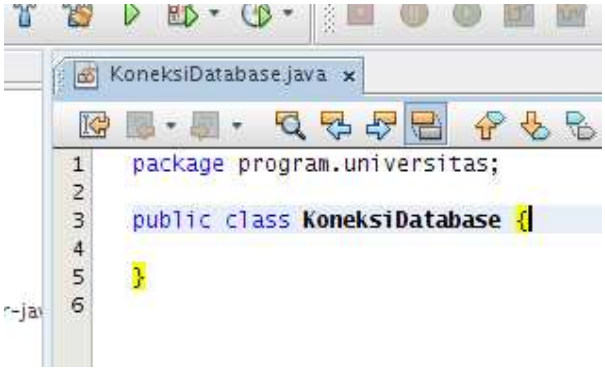
Setelah membuat package **program.universitas**, sekarang kita buat sebuah kelas untuk melakukan koneksi ke MySQL. Caranya klik kanan package**program.universitas** lalu pilih New -> Other.



Pilih kategori Java dan tipe filenya Java Class, setelah itu klik tombol Next untuk melanjutkan membuat sebuah kelas.



Beri nama kelas tersebut, misal KoneksiDatabase, setelah itu klik Finish agar kelas KoneksiDatabase terbuat.



Sekarang, saatnya melakukan proses pengkodean. Pertama buat sebuah variabel static yang bertipe java.sql.Connection, kita menggunakan static agar nanti aplikasi dapat mengakses koneksi secara langsung tanpa harus membuat object KoneksiDatabase.

```
package program.universitas;

import java.sql.Connection;

public class KoneksiDatabase {

    private static Connection koneksi;

}
```

Setelah itu buat sebuah metode static getKoneksi(), metode ini digunakan untuk mendapatkan koneksi itu sendiri.

```
package program.universitas;

import java.sql.Connection;

public class KoneksiDatabase {

    private static Connection koneksi;

    public static Connection getKoneksi() {
        return koneksi;
    }
}
```

Sekarang untuk mengambil koneksi kita dapat langsung menggunakan petintah KoneksiDatabase.getKoneksi(), namun pastinya jika kita melakukan hal itu akan terjadi error, kok bisa?

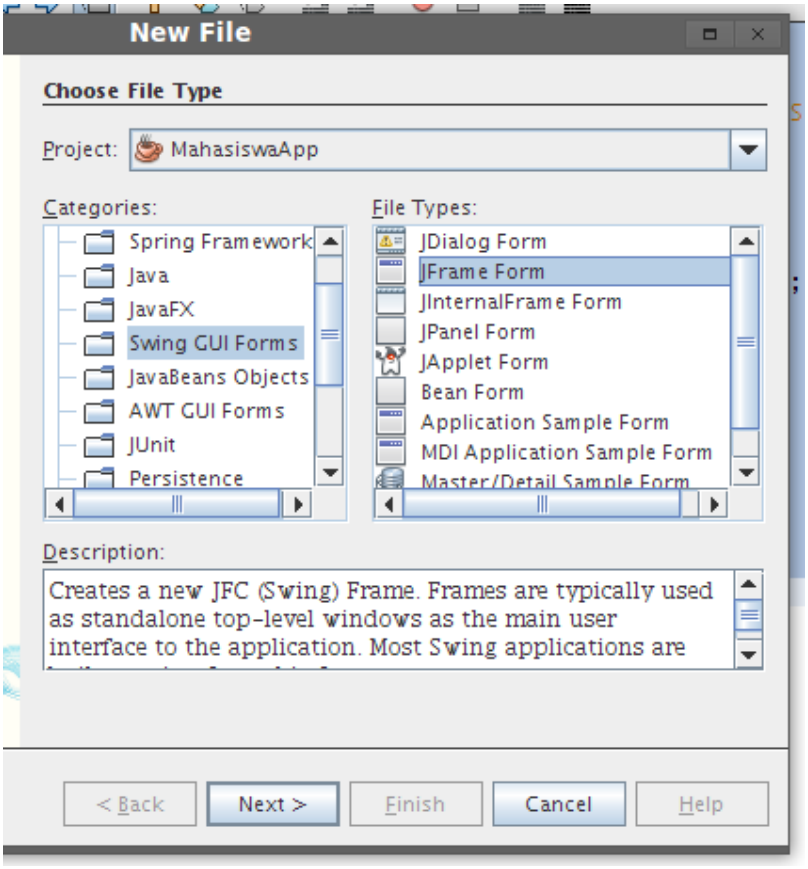
Tentu karena pada kelas KoneksiDatabase tersebut kita belum membuat koneksinya, jadi sebelum return koneksi, pada metode getKoneksi() seharusnya kita cek dulu apakah koneksi-nya null, jika null, maka kita deklarasikan sebuah koneksi yang baru.

```
package program.universitas;

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

public class KoneksiDatabase {

    private static Connection koneksi;

    public static Connection getKoneksi() {

        // cek apakah koneksi null
        if (koneksi == null) {

            try {
                String url = "jdbc:mysql://khannedy.server:3306/UNIVERSITAS";
                String user = "echo";
                String password = "xxxxx";

                DriverManager.registerDriver(new com.mysql.jdbc.Driver());

                koneksi = DriverManager.getConnection(url, user, password);

            } catch (SQLException t) {
                System.out.println("Error Membuat Koneksi");
            }

        }

        return koneksi;
    }
}
```

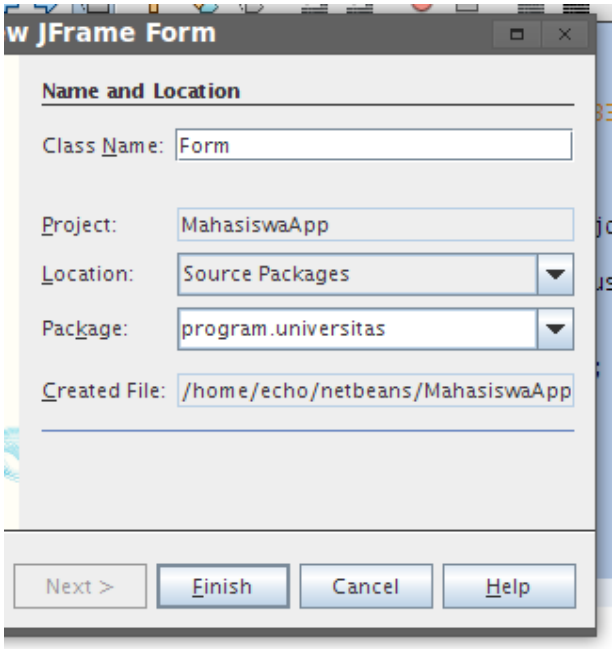
Sekarang, kita telah selesai membuat sebuah kelas untuk melakukan proses koneksi ke MySQL. Saatnya kita membuat Form aplikasinya.

Membuat Form Aplikasi

Pada program sederhana ini, kita tidak akan membuat program berbasis terminal (command line) lagi, tapi kita akan membuat aplikasi berbasis GUI. Dalam java teknologi untuk membuat program berbasis GUI disebut Java Swing. Sekarang untuk membuat sebuah Form, kita harus membuat JFrame, caranya dengan klik kanan package **program.universitas**, lalu pilih New -> Other.

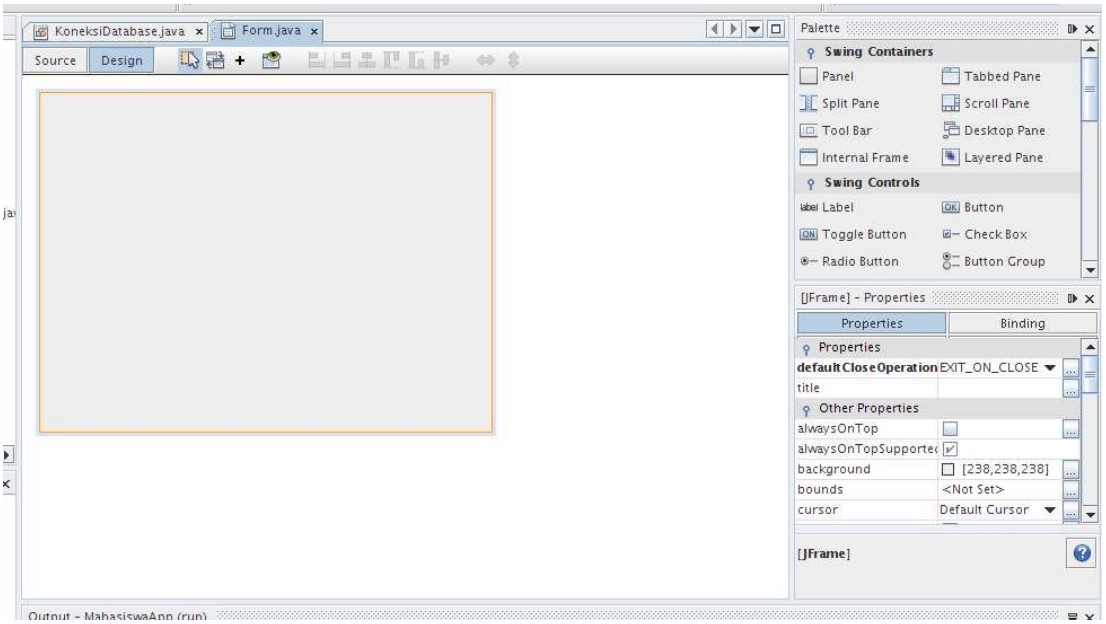


Pilih kategori Swing GUI Forms dan pilih tipe file JFrame Form. Lalu klik Next untuk melanjutkan membuat Form.

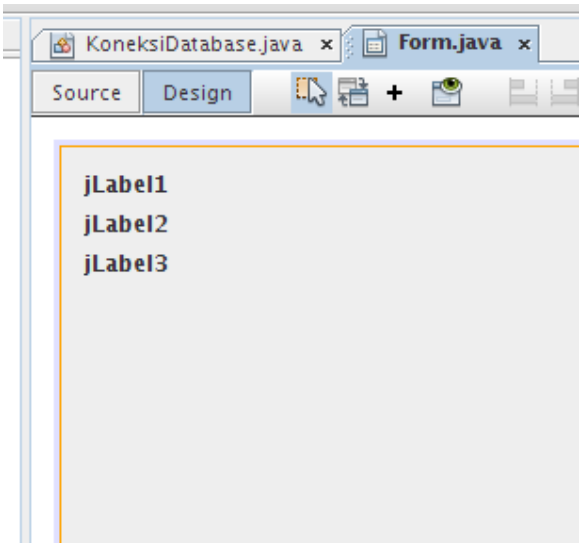


Beri nama Form tersebut, misal dengan nama Form, dengan begitu maka NetBeans akan membuatkan sebuah kelas dengan nama Form yang merupakan turunan dari kelas JFrame, dimana kelas JFrame ini merupakan kelas Java Swing.

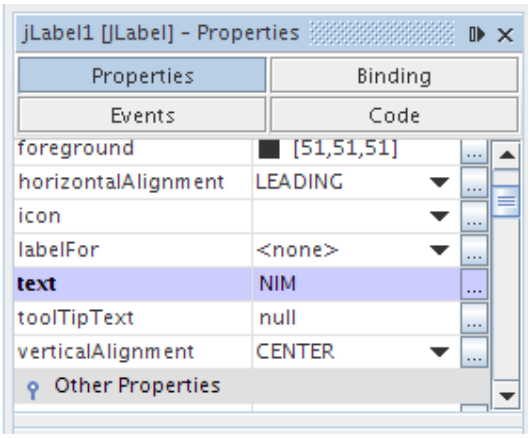
Sekarang kita dapat melihat GUI builder pada editor NetBeans dan disebelah kanannya terdapat Pallete yang merupakan komponen-komponen GUI yang ada di Java dan Properties yang merupakan editor atribut-atribut komponen yang kita klik pada GUI Builder.



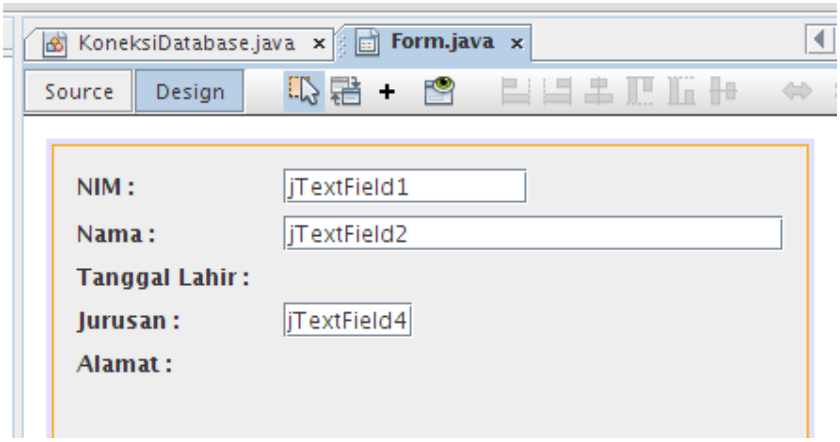
Untuk menambahkan komponen-komponen GUI lainnya, kita cukup mengklik dan mendrag salah satu komponen yang ada dalam Pallete ke dalam Form. Misal kita klik dan drag sebuah Label dari Pallete.



Untuk mengubah tulisan pada Label, kita dapat mengklik label tersebut, lalu lihat pada bagian Properties. Ubah atribut text, misa menjadi NIM, Nama, Tanggal Lahir, Jurusan dan Alamat.

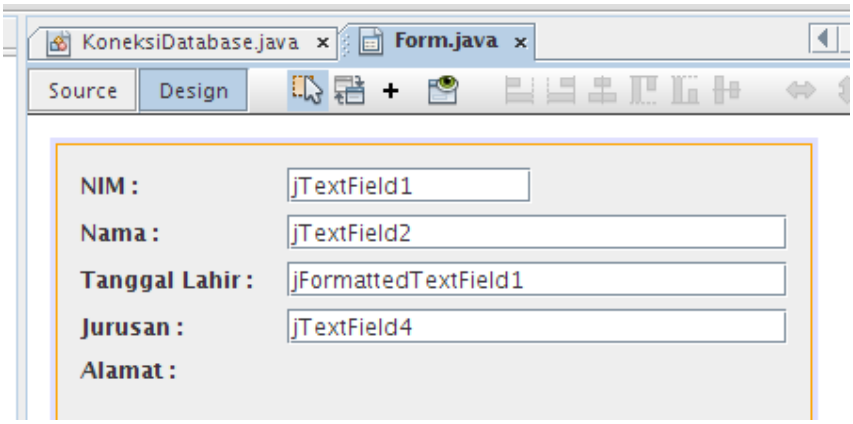


Setelah itu klik dan drag tiga buat Text Field yang ada dipallete ke Form, gunakan Text Field untuk Nim, Nama dan Jurusan.

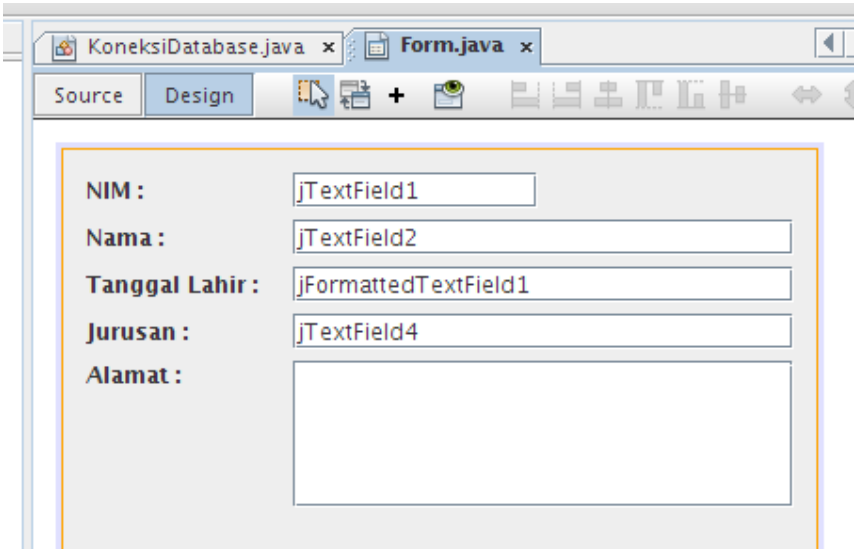


Untuk Tanggal Lahir dan Alamat kita tidak menggunakan Text Field, hal ini dikarenakan Tanggal Lahir memerlukan inputan berupa tanggal sedangkan Text Field hanya mendukung teks (string), sedangkan untuk Alamat, biasanya isi alamat itu panjang, sehingga lebih tidak cocok menggunakan Text Field, karena Text Field hanya mendukung satu baris.

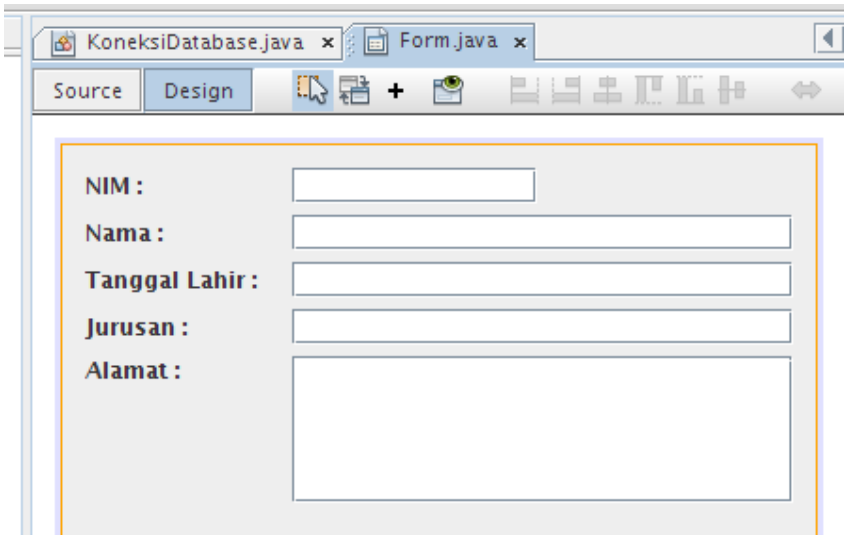
Dengan demikian, untuk Tanggal Lahir kita akan menggunakan Formatted Field, tinggal kita klik dan drag Formatted Field dari Palette ke dalam Form.



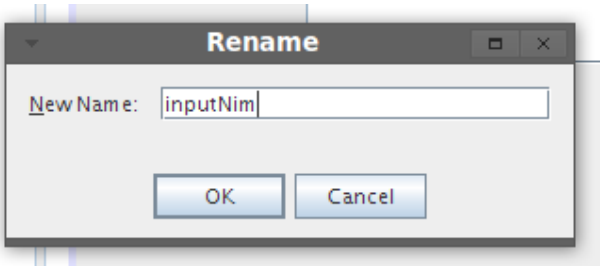
Dan untuk Alamat, gunakan komponen Text Area. Text Area hampir mirip dengan Text Field, namun mendukung lebih dari satu baris.



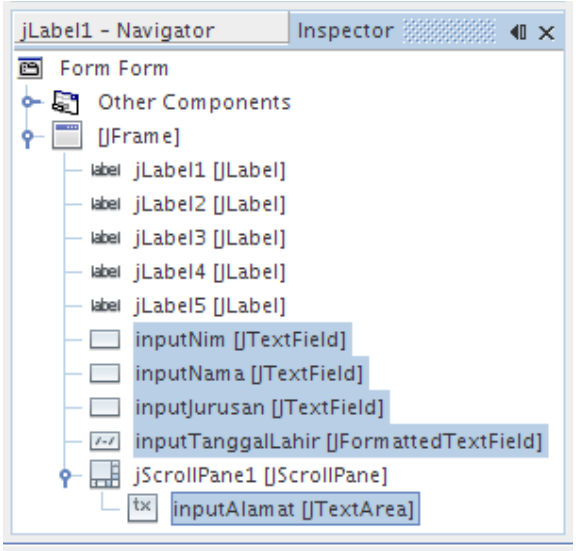
Untuk mengosongkan isi tulisan pada NIM, Nama, Tanggal Lahir dan Jurusan, kosongkan atribut text pada setiap komponen pada Propertiesnya.



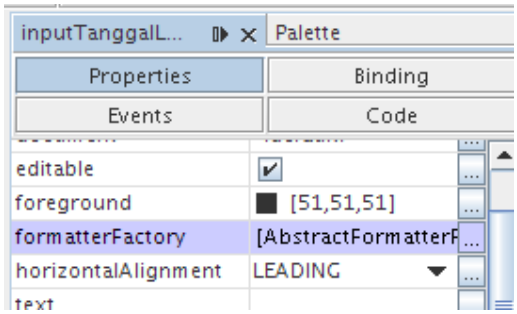
Setelah itu, sekarang saatnya kita mengubah setiap nama variabel komponenya, misal untuk Text Field NIM kita beri nama variabelnya dengan nama inputNim, untuk Text Field Nama dengan nama inputNama dan seterusnya, caranya dengan mengklik kanan komponenya lalu pilih menu Change Variable Name.



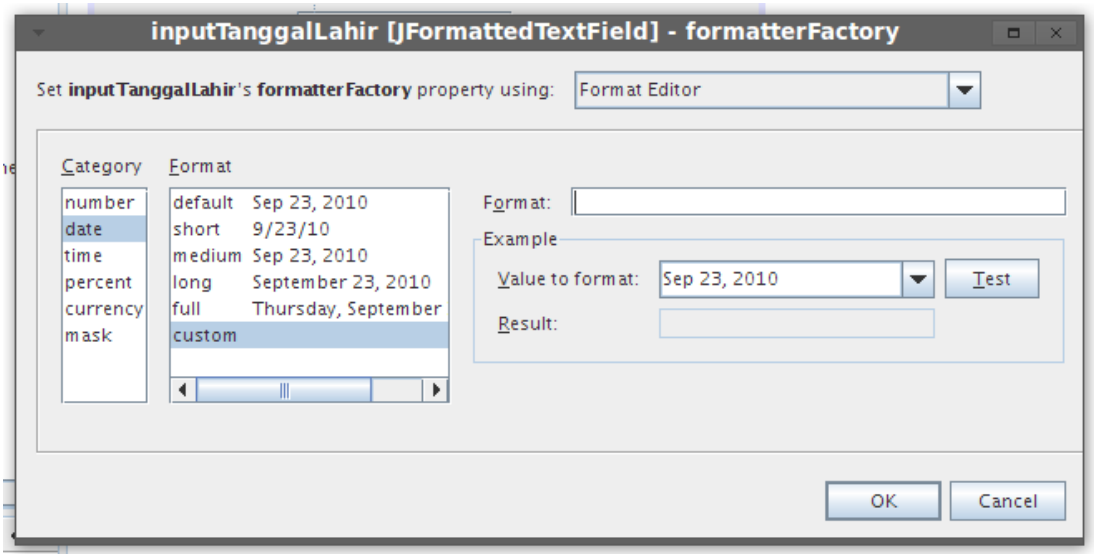
Untuk melihat seluruh nama variabelnya, kita dapat melihatnya pada bagian Inspector di sebelah kiri bawah Form NetBeans.



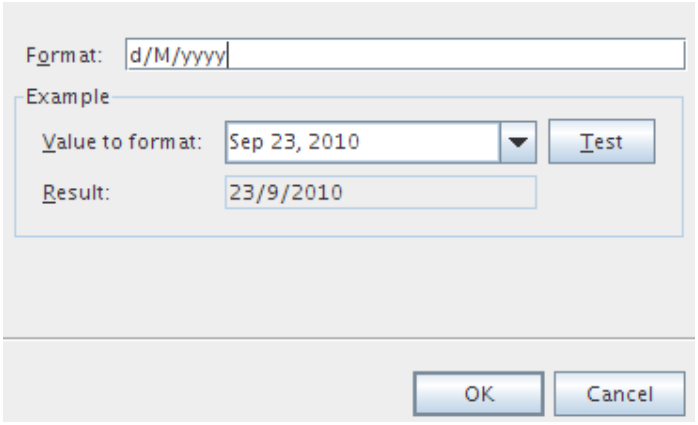
Secara default Formatted Field seperti Text Field, dia hanya menerima teks (String), agar Formatted Field hanya menerima input berupa tanggal, maka kita perlu memberitahukannya ke Formatted Field nya, caranya klik inputTanggalLahir, lalu pada bagian Properties, cari atribut formatterFactory, ubah atribut tersebut.



Pada saat mengklik tombol [...] pada atribut formatterFactory, maka akan muncul dialog formatterFactory.



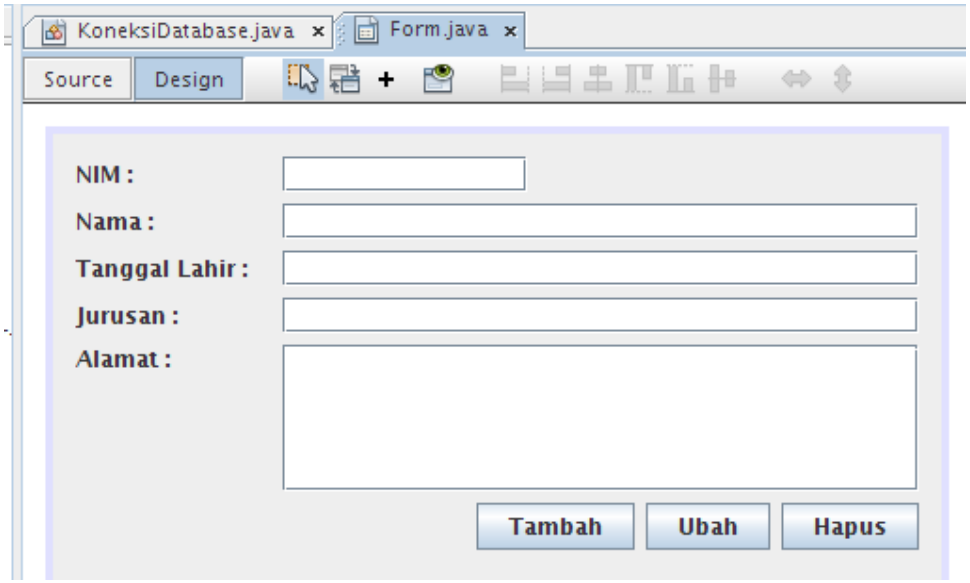
Agar Formatted Field hanya meneripa input tanggal, maka ubah kategorinya menjadi date, formatnya menjadi custom, lalu pada input Format beri teks “d/M/yyyy”.



Maksud dari “d/M/yyyy” merupakan representasi tanggal/bulan/tahun dalam angka, misal jika tanggal 1 Januari 2010, maka input harus 1/1/2010 dan seterusnya. Klik tombol OK untuk mengkonfirmasi perubahan.

Menambah Tombol ke Form

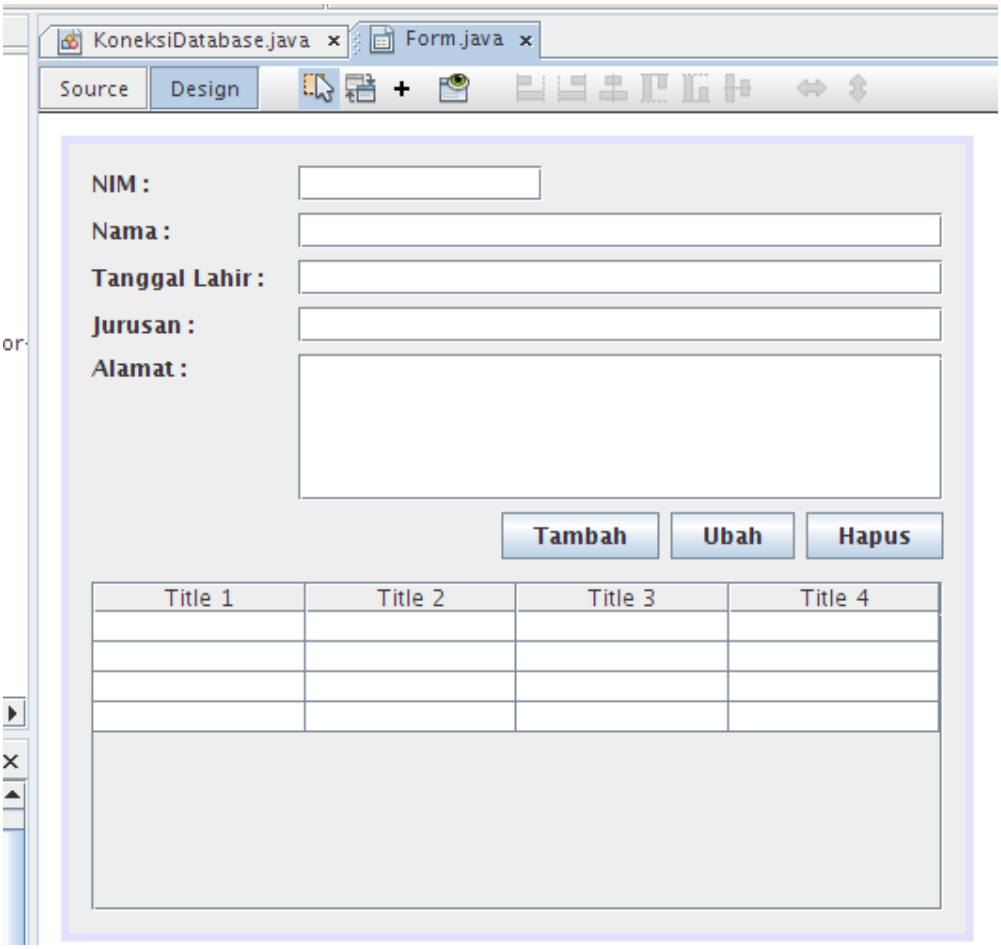
Setelah menambahkan input Form, sekarang saatnya kita menambahtombol ke dalam Form. Caranya dengan mengklik dan drag komponen Button pada Palette ke dalam Form.



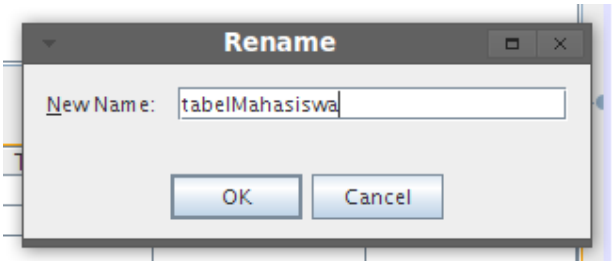
Tambahkan 3 buah tombol, Tambah, Ubah dan Hapus. Untuk mengubah teks tombolnya caranya sama seperti Label, yaitu dengan mengubah atribut text pada Properties.

Menambah Tabel ke Form

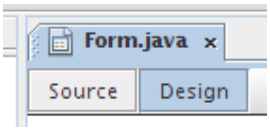
Setelah menambahkan input Form beserta tombolnya, sekarang saatnya menambahkan Tabel ke Form, caranya tinggal kita klik dan drag komponen Table dari Palette ke Form, hasilnya seperti terlihat pada gambar dibawah ini.



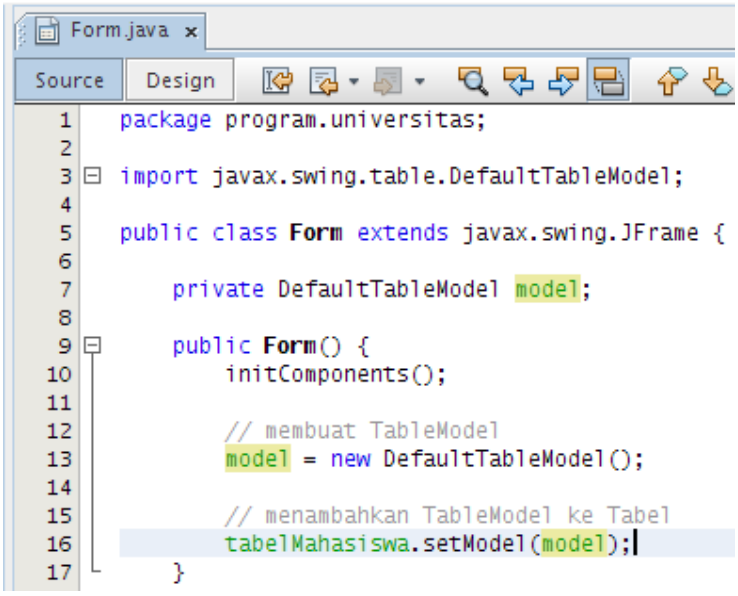
Jangan lupa untuk mengubah nama variabel Tabel yang tadi kita masukkan ke Form, caranya klik kanan Tabel nya lalu pilih Change Variabel Name, misal dengan nama tabelMahasiswa.



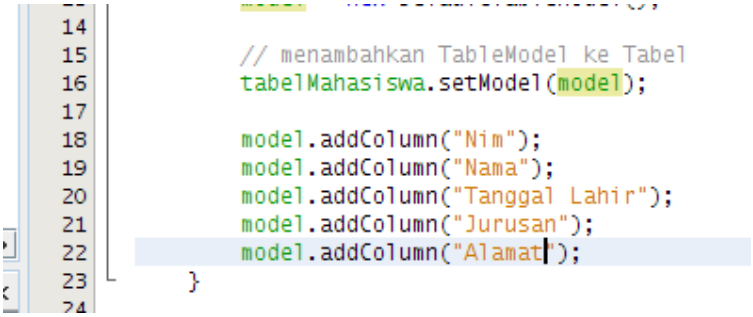
Sekarang saatnya mengubah kolom pada Tabel. Berbeda dengan komponen lain, untuk mengubah kolom pada komponen Tabel, kita memerlukan kelas lain, namanya kelas DefaultTableModel, sehingga kita perlu melakukan pengkodean, caranya masuk ke bagian Source.



Setelah itu tambahkan sebuah variabel DefaultTableModel pada kelas Form tersebut.



Untuk menambahkan kolom ke Tabel, maka kita dapat menggunakan metode addColumn(nama) milik kelas DefaultTableModel. Dan saat ini kita perlu menambahkan kolom Nim, Nama, Tanggal Lahir, Jurusan dan Alamat.



Menambahkan Aksi

Sekarang kita telah selesai membuat Form, saatnya kita menambahkan aksi-aksi database, seperti load data dari database, menambah data ke database, mengubah data dari database dan menghapus data dari database.

Menambah Aksi Load Data

Saat pertama kali aplikasi muncul, maka otomatis kita harus mengambil seluruh data mahasiswa yang ada dalam tabel MAHASISWA dan ditampilkan ke dalam Table yang ada di Form. Dengan demikian, maka pertama kita perlu membuat sebuah aksi melakukan load data dari database.

Sekarang kita buat sebuah metode dengan nama loadData() dimana metode tersebut dibuat dalam kelas Form dan dalam metode tersebut berisikan proses load data dari database.

```
public void loadData(){
```

```
}
```

Sebelum melakukan proses load data dari database, maka pertama kali, kita perlu menghapus seluruh isi baris yang ada pada Table yang ada di Form. Hal ini perlu dilakukan agar saat kita akan melakukan load ulang data, maka Tabel dikosongkan dulu. Untuk mengosongkan isi Table, kita harus menggunakan DefaultTableMode.

```
public void loadData(){
    // menghapus seluruh data
    model.getDataVector().removeAllElements();
    // memberi tahu bahwa data telah kosong
    model.fireTableDataChanged();
}
```

Setelah itu baru kita melakukan laod datanya, untuk mengambil data dari database, kita memerlukan Connection yang ada dalam kelas KoneksiDatabase. Setelah itu buat Statement dan ResultSet seperti biasanya.

```
public void loadData(){
    // menghapus seluruh data
    model.getDataVector().removeAllElements();
    // memberi tahu bahwa data telah kosong
    model.fireTableDataChanged();

    try{
        Connection c = KoneksiDatabase.getKoneksi();
        Statement s = c.createStatement();

        String sql = "SELECT * FROM MAHASISWA";
        ResultSet r = s.executeQuery(sql);

        while(r.next()){
            // lakukan penelusuran baris
        }

        r.close();
        s.close();
    }catch(SQLException e){
        System.out.println("Terjadi Error");
    }
}
```

Pada saat melakukan proses penelurusan data menggunakan ResultSet, maka kita dapat menambahkan data tersebut ke dalam Table yang ada dalam Form. Untuk menambah sebuah baris ke Table kita menambahkannya ke DefaultTableModel dengan menggunakan metode addRow(Object[]).

```
while(r.next()){
    // lakukan penelusuran baris
    Object[] o = new Object[5];
    o[0] = r.getString("NIM");
    o[0] = r.getString("NAMA");
    o[0] = r.getDate("TANGGAL LAHIR");
    o[0] = r.getString("JURUSAN");
    o[0] = r.getString("ALAMAT");
```

```
    model.addRow(o);
}
```

Lengkapnya metode loadData() akan berisi sepertipada kode dibawah ini.

```
public void loadData(){
    // menghapus seluruh data
    model.getDataVector().removeAllElements();
    // memberi tahu bahwa data telah kosong
    model.fireTableDataChanged();

    try{
        Connection c = KoneksiDatabase.getKoneksi();
        Statement s = c.createStatement();

        String sql = "SELECT * FROM MAHASISWA";
        ResultSet r = s.executeQuery(sql);

        while(r.next()){
            // lakukan penelusuran baris
            Object[] o = new Object[5];
            o[0] = r.getString("NIM");
            o[1] = r.getString("NAMA");
            o[2] = r.getDate("TANGGAL_LAHIR");
            o[3] = r.getString("JURUSAN");
            o[4] = r.getString("ALAMAT");

            model.addRow(o);
        }

        r.close();
        s.close();
    }catch(SQLException e){
        System.out.println("Terjadi Error");
    }
}
```

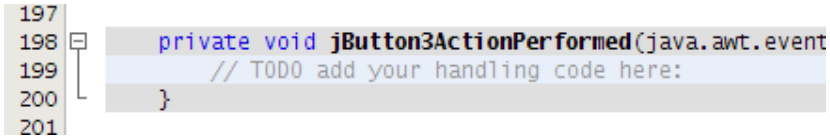
Agar metode loadData() dipanggil ketika program berjalan, maka kita perlu memanggil metode loadData() dalam konstruktor Form.

```
25          model.addColumn("Jurusan
26          model.addColumn("Alamat"
27
28          // panggil loadData()
29          loadData();
30      }
31
32      public void loadData() {
33          // menghapus seluruh dat
34          model.getDataVector().re
35          // memberi tahu bahwa da
36          model.fireTableDataChang
```

Menambah Aksi Tombol Tambah

Sekarang saatnya kita menambahkan aksi tombol, dimana aksi tombol itu akan berjalan ketika tombol Tambah diklik. Untuk menambah sebuah aksi ke tombol Tambah, pertama kita masuk lagi ke bagian Design, setelah itu tinggal klik kanan tombol Tambah-nya setelah

itu pilih menu **Events -> Action -> actionPerformed**, maka otomatis NetBeans IDE akan membuatkan sebuah metode baru untuk aksi tombol Tambah.



Dalam metode tersebutlah kita melakukan proses penambahan data ke dalam database. Untuk menambahkan data ke dalam tabel MAHASIWA, otomatis kita memerlukan data input dari pengguna. Untuk mendapatkan data tulisan dari Text Field dan Text Area, maka kita dapat menggunakan metode getText(), sedangkan untuk mendapatkan tanggal dari Formatted Field, kita dapat menggunakan metode getValue(), namun dikarenakan getValue() menghasilkan Object, maka kita perlu mengkonversinya ke tanggal.

```
String nim = inputNim.getText();
String nama = inputNama.getText();
java.util.Date tanggalLahir = (java.util.Date)inputTanggalLahir.getValue();
String jurusan = inputJurusan.getText();
String alamat = inputAlamat.getText();
```

Setelah mengambil seluruh data dari input, maka baru kita menyimpannya ke dalam database MySQL. Caranya adalah dengan membuat Connection dari kelas KoneksiDatabase setelah itu membuat PrepareStatement untuk menyimpan datanya.

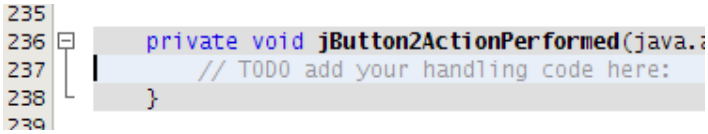
```
String nim = inputNim.getText();
String nama = inputNama.getText();
java.util.Date tanggalLahir = (java.util.Date) inputTanggalLahir.getValue();
String jurusan = inputJurusan.getText();
String alamat = inputAlamat.getText();

try{
    Connection c = KoneksiDatabase.getKoneksi();

    String sql = "INSERT INTO MAHASISWA VALUES (?, ?, ?, ?, ?)";

    PreparedStatement p = c.prepareStatement(sql);

    p.setString(1, nim);
    p.setString(2, nama);
    p.setDate(3, new java.sql.Date(tanggalLahir.getTime()));
    p.setString(4, jurusan);
    p.setString(5, alamat);

    p.executeUpdate();
    p.close();
}catch(SQLException e){
    System.out.println("Terjadi Error");
}finally{
    loadData();
}
```

Pada blok finally, kita perlu memanggil metode loadData(), hal ini dilakukan agar setelah proses penyimpanan data ke database, maka data akan dimuat ulang ke Table yang ada di Form.

Menambah Aksi Tombol Ubah

Untuk aksi tombol Ubah, agak sedikit berbeda dengan aksi tombol Tambah, perbedaannya adalah pertama kita harus mendeteksi baris yang sedang diklik, setelah itu baru melakukan proses pengubahan data yang diklik dengan data baru yang ada dalam input Form.

Untuk menambah aksi tombol Ubah caranya sama dengan tombol Tambah, tinggal klik kanan tombol Ubah lalu pilih **Events -> Action -> actionPerformed**.



Seperti yang telah ditulis sebelumnya, pertama kita harus mendapatkan baris yang terseleksi pada Table, jika tidak ada baris yang terseleksi, maka proses Ubah dibatalkan. Untuk mendapatkan baris yang terseleksi kita dapat menggunakan metode getSelectedRow() milik Table, jika return-nya -1 artinya tidak ada baris yang terseleksi.

```
int i = tabelMahasiswa.getSelectedRow();
if(i == -1){
    // tidak ada baris terseleksi
    return;
}

// ambil nim yang terseleksi
String nim = (String) model.getValueAt(i, 0);

String nama = inputNama.getText();
java.util.Date tanggalLahir = (java.util.Date) inputTanggalLahir.getValue();
String jurusan = inputJurusan.getText();
String alamat = inputAlamat.getText();
```

Setelah mengambil data nim yang terseleksi dan data lainnya dari input, baru kita lakukan proses ubah data yang ada di database berdasarkan nim yang baris yang terseleksi.

```
int i = tabelMahasiswa.getSelectedRow();
if(i == -1){
    // tidak ada baris terseleksi
    return;
}

// ambil nim yang terseleksi
String nim = (String) model.getValueAt(i, 0);

String nama = inputNama.getText();
java.util.Date tanggalLahir = (java.util.Date) inputTanggalLahir.getValue();
String jurusan = inputJurusan.getText();
String alamat = inputAlamat.getText();

try{
    Connection c = KoneksiDatabase.getKoneksi();
```

```
    String sql = "UPDATE MAHASISWA SET NAMA = ?, TANGGAL_LAHIR = ?, JURUSAN = ?,
ALAMAT = ? WHERE NIM = ?";

    PreparedStatement p = c.prepareStatement(sql);

    p.setString(1, nama);
    p.setDate(2, new java.sql.Date(tanggalLahir.getTime()));
    p.setString(3, jurusan);
    p.setString(4, alamat);
    p.setString(5, nim);

    p.executeUpdate();
p.close();
}catch(SQLException e){
    System.out.println("Terjadi Error");
}finally{
    loadData();
}
```

Menambah Aksi Tombol Hapus

Untuk aksi hapus, kita tidak perlu menggunakan input Form, yang kita perlukan hanyalah baris yang terseleksi. Jika baris tidak ada yang terseleksi, maka proses penghapusan dibatalkan. Untuk menambah aksi pada tombol Hapus caranya sama seperti tombol Tambah dan Ubah, klik kanan tombol Hapus, lalu pilih menu**Events -> Action -> actionPerformed**.

Setelah itu sama seperti pada proses Ubah, kita cek dulu apakah ada baris yang terseleksi atau tidak, jika ada ambil nim yang terseleksi, jika tidak ada, maka batalkan proses Hapus.

```
int i = tabelMahasiswa.getSelectedRow();
if(i == -1){
    // tidak ada baris terseleksi
    return;
}

String nim = model.getValueAt(i, 0);
```

Setelah itu, baru kita lakukan proses penghapusan data dari database berdasarkan data baris yang terseleksi.

```
int i = tabelMahasiswa.getSelectedRow();
if(i == -1){
    // tidak ada baris terseleksi
    return;
}

String nim = (String) model.getValueAt(i, 0);

try{
    Connection c = KoneksiDatabase.getKoneksi();

    String sql = "DELETE FROM MAHASISWA WHERE NIM = ?";
```

```
    PreparedStatement p = c.prepareStatement(sql);

    p.setString(1, nim);

    p.executeUpdate();
    p.close();
}catch(SQLException e){
    System.err.println("Terjadi Error");
}finally{
    loadData();
}
```

Menambahkan Aksi Baris Terseleksi

Aksi terakhir yang perlu kita tambahkan adalah aksi ketika baris Table terseleksi, misal jika baris pertama terseleksi, maka program akan menampilkan data yang terseleksi tersebut pada Form. Hal ini agar pengubahan lebih mudah, karena kita tidak perlu memasukkan seluruh datanya lagi.

Untuk menambahkan aksi ketika baris terseleksi, kita dapat menggunakan aksi Mouse Click, yaitu aksi yang dijalankan ketika mouse mengklik. Caranya, klik kanan componen Table pada Form, setelah itu pilih **Events -> Mouse -> mouseClicked**. Sekarang akan terbuat sebuah metode baru yang akan dipanggil ketika Table diklik.

```
312
313     private void tabelMahasiswaMouseClicked(java
314         // TODO add your handling code here:
315     }
316
```

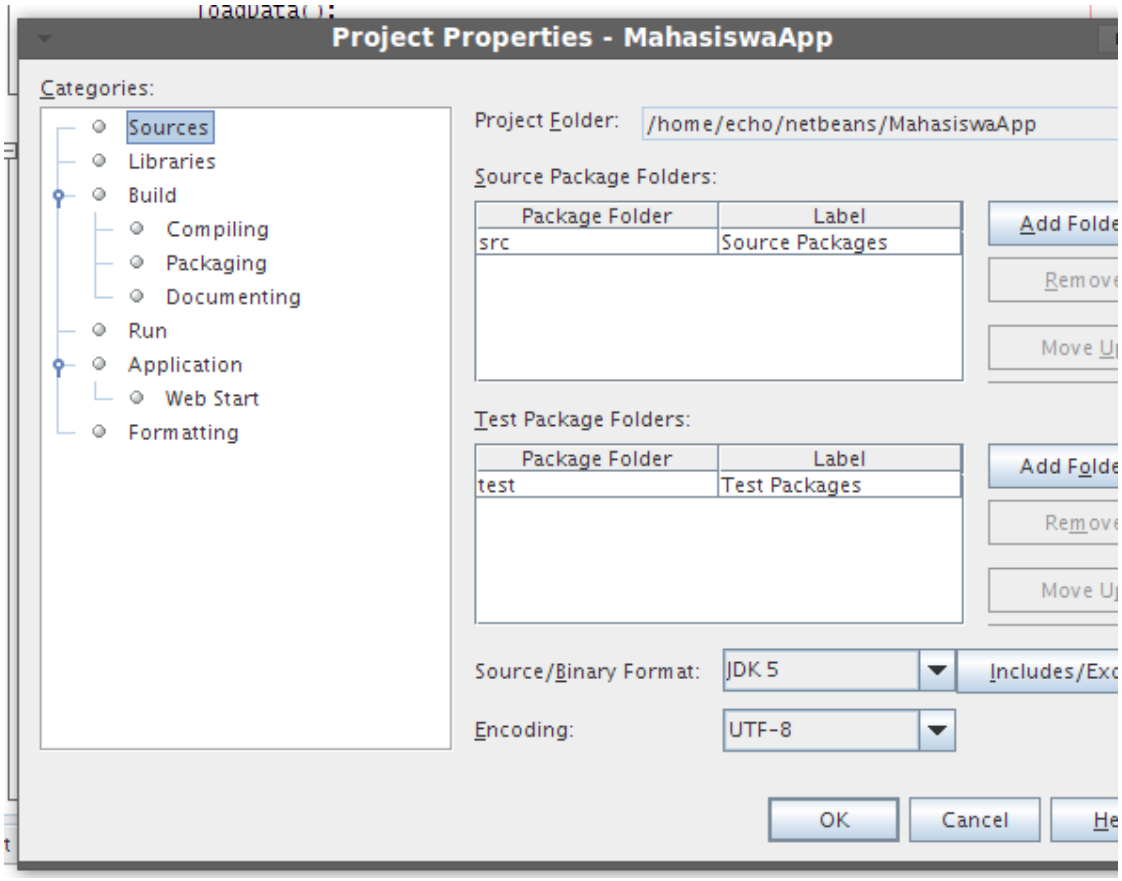
Pertama yang harus dilakukan adalah melakukan pengecekah apakah ada baris yang terseleksi, jika ada maka ambil data yang terseleksi dari DefaultTableModel setelah itu tampilkan pada Form, namun jika tidak ada baris yang terseleksi, maka batalkan proses.

```
int i = tabelMahasiswa.getSelectedRow();
if(i == -1){
    // tak ada baris terseleksi
    return;
}

String nim = (String) model.getValueAt(i, 0);
inputNim.setText(nim);

String nama = (String) model.getValueAt(i,1);
inputNama.setText(nama);

java.util.Date tanggalLahir = (java.util.Date) model.getValueAt(i, 2);
inputTanggalLahir.setValue(tanggalLahir);

String jurusan = (String) model.getValueAt(i, 3);
inputJurusan.setText(jurusan);

String alamat = (String) model.getValueAt(i, 4);
inputAlamat.setText(alamat);
```

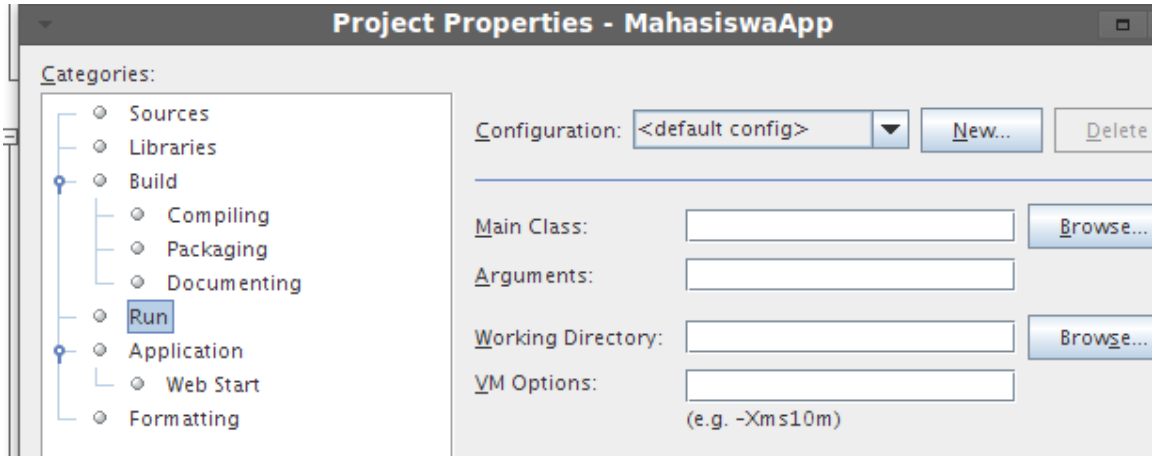
Menjalankan Program

Setelah pembuatan program telah selesai, saatnya menjalankan aplikasi. Untuk menjalankan aplikasi, pertama kita harus menentukan dahulu kelas yang akan digunakan sebagai program, dimana pada project yang telah kita buat, kelas program adalah kelas Form.

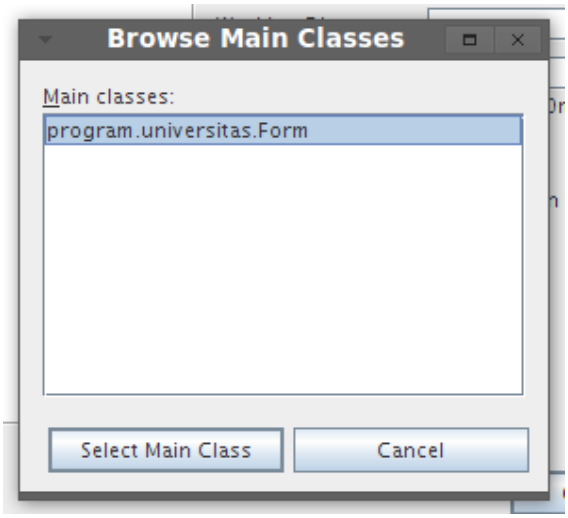
Untuk mengubah kelas program menjadi kelas Form, maka kita dapat mengubahnya dengan cara klik kanan Project yang telah kita buat, lalu pilih menu **Properties**, setelah itu akan keluar dialog Project Properties.



Pada bagian Categories, pilihlah menu Run, untuk mengubah kelas yang akan dijalankan sebagai kelas Program.

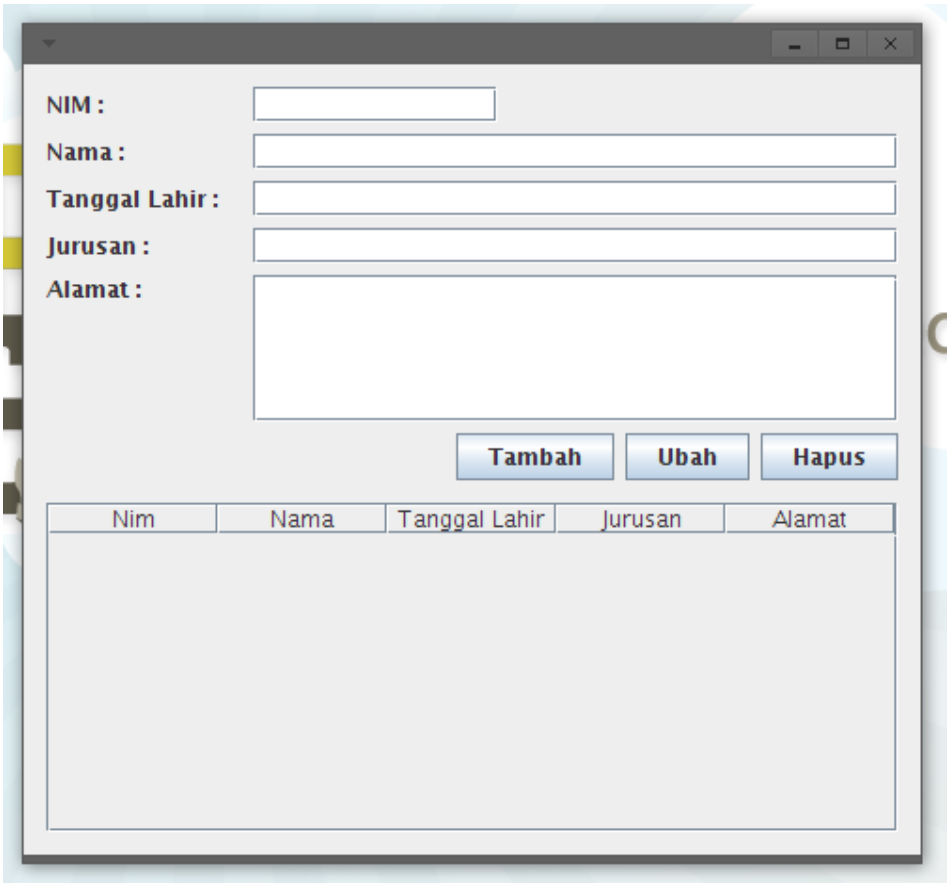


Pada input Main Class, klik tombol Browse, untuk menampilkan daftar kelas yang dapat dijadikan sebagai kelas program. Maka akan keluar dialog pemilihan kelas.

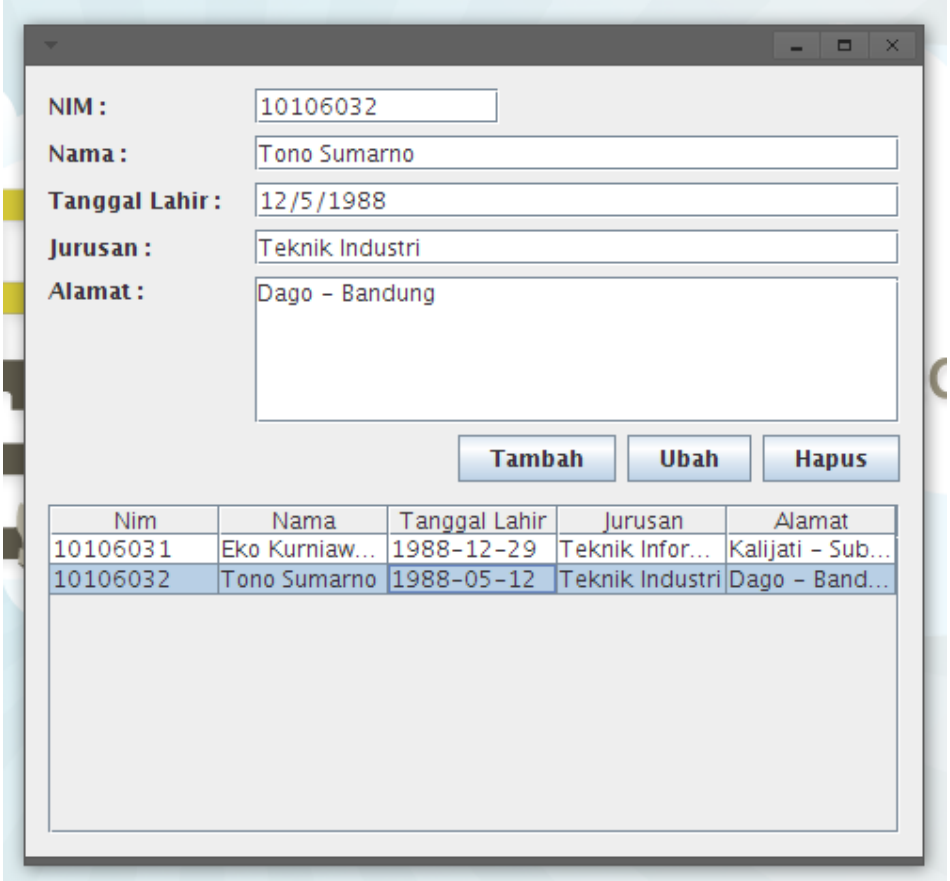


Pilih kelas Form yang telah dibuat tadi, setelah itu klik tombol Select Main Class. Untuk mengkonfirmasi perubahan, klik tombol OK. Sekarang kelas Form akan menjadi kelas yang dijalankan jika Project dijalankan.

Untuk menjalankan aplikasi, klik kanan project yang telah kita buat, setelah itu pilih menu **Run**, maka otomatis program akan berjalan. Pada komputer saya, program terlihat seperti pada gambar dibawah ini.



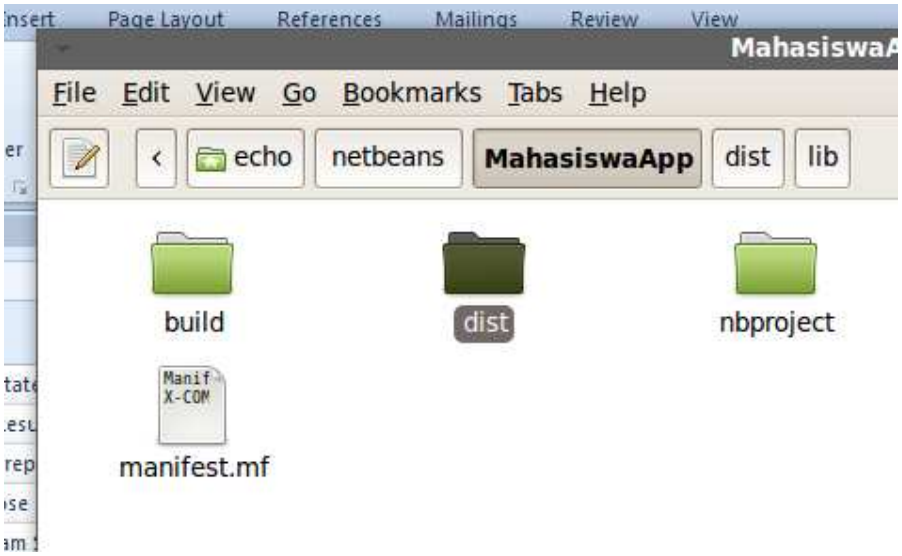
Sekarang, kita dapat menambah data, mengubah data dan menghapus data yang telah kita masukkan.



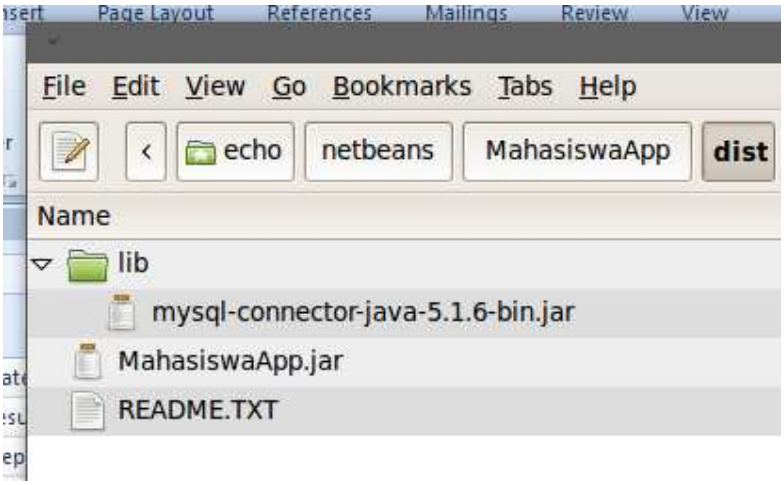
Mendistribusikan Program

Saat ini kita telah menyelesaikan program yang kita buat, saatnya mendistribusikanprogram yang kita buat ke komputer lain, hal ini perlu dilakukan agar untuk menjalankan program kita tidak perlu menggunakan NetBeans IDE lagi.

Langkah pertama adalah build project yang telah kita bangun tadi agar menjadi file yang siap didistribusikan. Caranya klik kanan project-nya lalu pilih **Clean and Build**. Setelah selesai maka akan tercipta sebuah folder**dist** pada folder project yang telah kita buat.



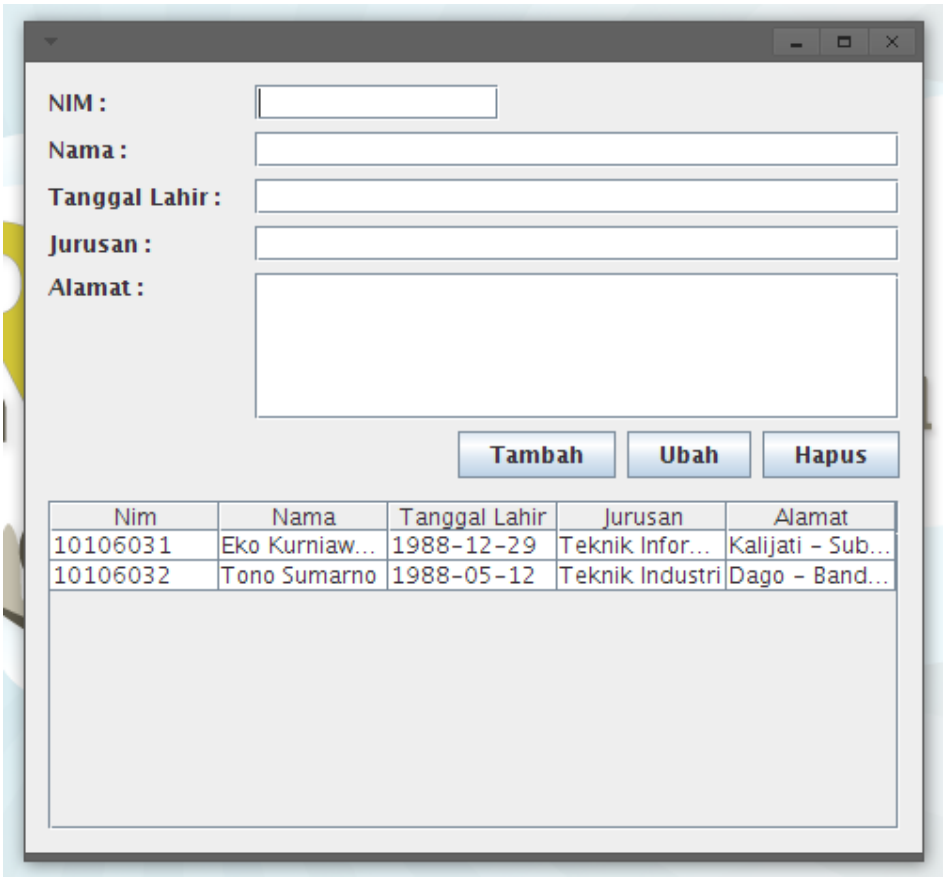
Dalam folder dist tersebut terdapat sebuah file JAR yang bernama MahasiswaApp, sesuai dengan nama project yang kita buat. Selain itu akan ada folder lib yang berisikan library-library yang dibutuhkan oleh program yang kita buat, misal driver untuk MySQL.



Untuk menjalankan dari luar NetBeans, kita hanya perlu menjalankan menggunakan terminal atau command menggunakan perintah :

```
java -jar MahasiswaApp.jar
```

Maka program akan berjalan seperti berjalan pada NetBeans IDE.


NIM :
Nama :
Tanggal Lahir :
Jurusan :
Alamat :
Tambah
Ubah
Hapus
Nim
Nama
Tanggal Lahir
Jurusan
Alamat
10106031
Eko Kurniaw...
1988-12-29
Teknik Infor...
Kalijati - Sub...
10106032
Tono Sumarno
1988-05-12
Teknik Industri
Dago - Band...

Tentang Penulis

Penulis bernama **Eko Kurniawan Khannedy S.Kom**. Lahir di kota Subang tanggal 29 Desember 1988, dan besar di kota Subang. Penulis merupakan lulusan Universitas Komputer Indonesia.

Saat ini penulis menjabat sebagai **Chief Executive Officer** di **StripBandunk**, yaitu perusahaan yang bergerak di pengembangan teknologi informasi dan komunikasi.

Penulis aktif di berbagai komunitas teknologi dan saat ini penulis adalah **Leader** di komunitas **Java User Group Bandung** dan juga **Moderator** di komunitas **NetBeans Indonesia**.

Penulis dapat dihubungi di :

- echo.khannedy@gmail.com
- http://twitter.com/khannedy
- http://facebook.com/khannedy

:D