

# Membuat Aplikasi Java Database dengan WindowBuilder Eclipse



Author : Resa Cr  
Chief on [www.marisharingilmu.wordpress.com](http://www.marisharingilmu.wordpress.com)

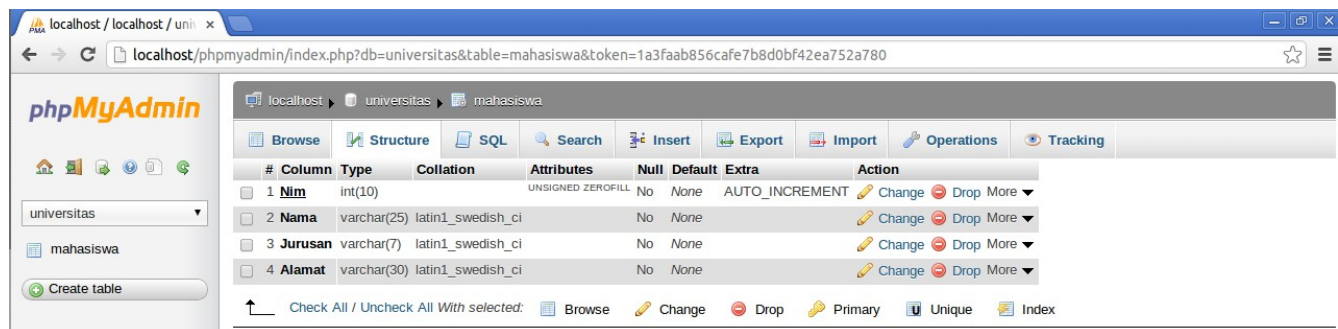
### **Lisensi Pocket Book**

Pocket Book ini dapat anda copy dan distribusikan dengan gratis untuk kegiatan pembelajaran.

Selamat datang di Pocket Book edisi selanjutnya yaitu **Pocket Book : Membuat Aplikasi Java dengan WindowBuilder Eclipse**. Sebelumnya saya pernah membuat Pocket Book dengan judul **Pocket Book MySQL**. Tools yang dibutuhkan dalam tutorial ini adalah :

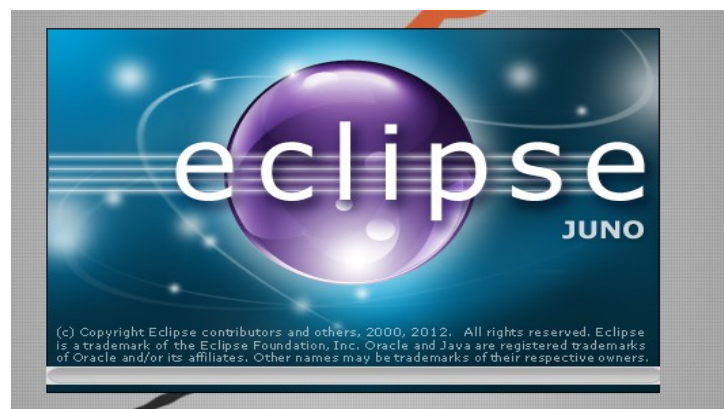
1. ) Eclipse Juno : <http://eclipse.org/downloads/packages/eclipse-ide-java-developers/junosr1>
2. ) MySQL Connector : <http://dev.mysql.com/downloads/connector/j/>
3. ) JTattoo : <http://www.jtattoo.net/>

WindowBuilder adalah GUI Builder yang berfungsi untuk memudahkan programmer dalam membuat aplikasi Java dengan Eclipse. Jadi dengan tool ini, anda hanya drag and drop komponen Swing Java. WindowBuilder include di dalam Eclipse Juno dan Eclipse edisi sebelumnya yaitu Helios. WindowBuilder cara menggunakannya hampir sama dengan GUI Builder milik Netbeans. Akan tetapi menurut pengalaman penulis, code generator dari komponen yang di drag and drop WindowBuilder lebih simpel dari GUI Builder milik Netbeans. Jadi anda dapat mengubahnya dengan mudah. Pada tutorial kali ini saya akan membuat aplikasi database sederhana dengan menggunakan Java dan MySQL. Untuk membuat aplikasi database, sebelumnya kita buat dulu database dan tabelnya dengan menggunakan **PhpMyAdmin**. Buat Database **universitas** dengan tabel **mahasiswa**. Untuk field/atributnya dapat dilihat pada gambar di bawah ini

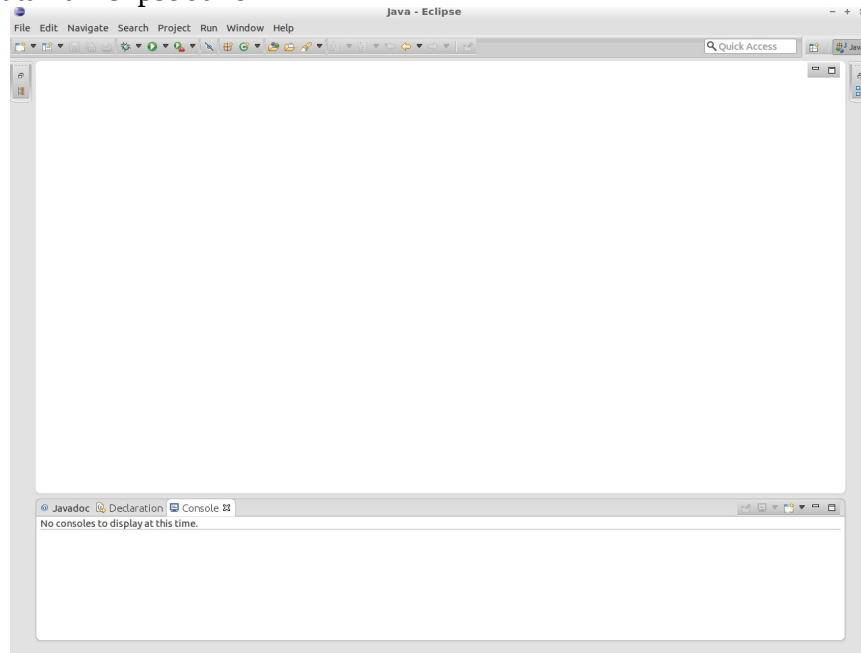


#	Column	Type	Collation	Attributes	Null	Default	Extra	Action
1	Nim	int(10)		UNSIGNED ZEROFILL	No	None	AUTO_INCREMENT	Change Drop More
2	Nama	varchar(25)	latin1_swedish_ci		No	None		Change Drop More
3	Jurusan	varchar(7)	latin1_swedish_ci		No	None		Change Drop More
4	Alamat	varchar(30)	latin1_swedish_ci		No	None		Change Drop More

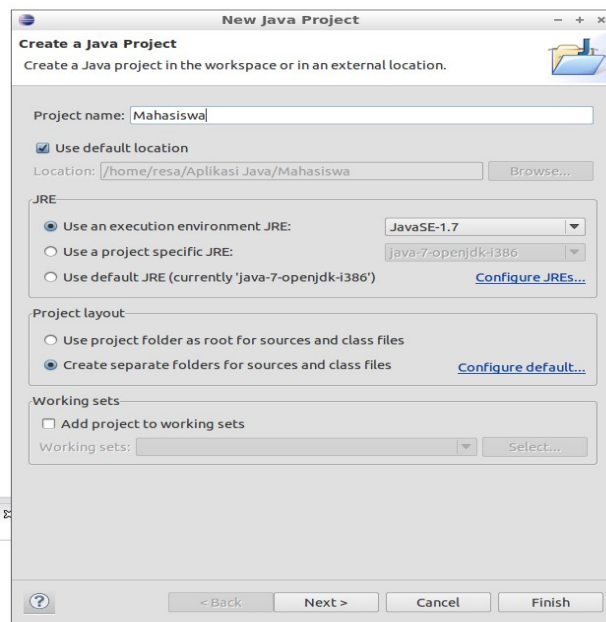
Baru saja kita membuat database dan tabel – tabelnya. Selanjutnya untuk membuat aplikasi Java buka Eclipsenya. Pada tutorial kali ini saya menggunakan Eclipse Juno.



Berikut ini menu utama Eclipse Juno

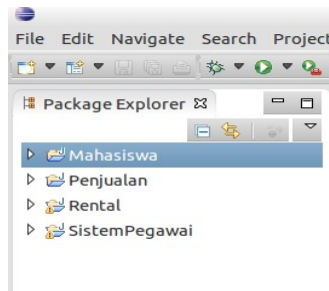


Untuk membuat aplikasi, klik **File-->New-->Java Project** .Beri nama Projectnya, **Mahasiswa** .Kalau sudah klik **Next** dan **Finish**.

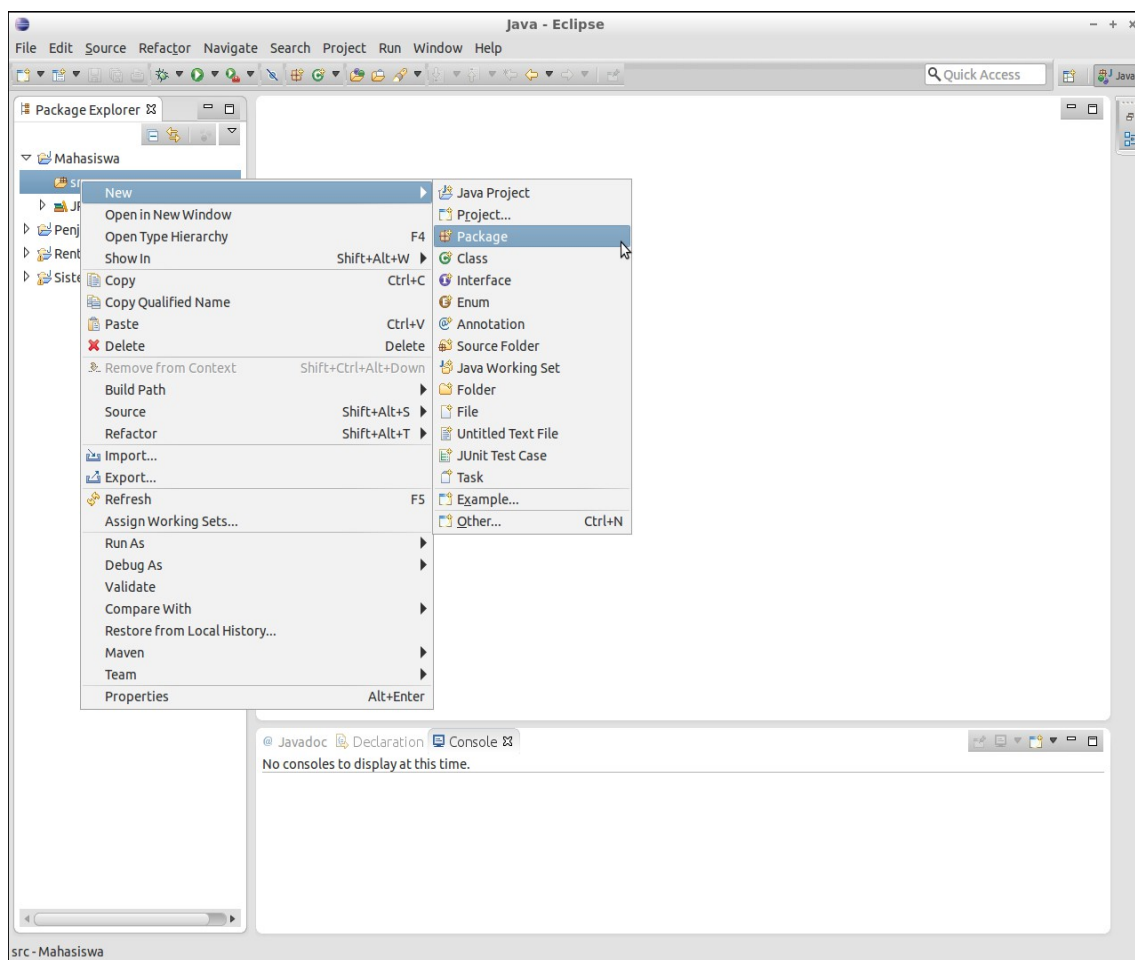


Kalau berhasil, maka anda akan mendapatkan Folder Mahasiswa di **Package Explorer**.

visit : [www.marisharingilmu.wordpress.com](http://www.marisharingilmu.wordpress.com)

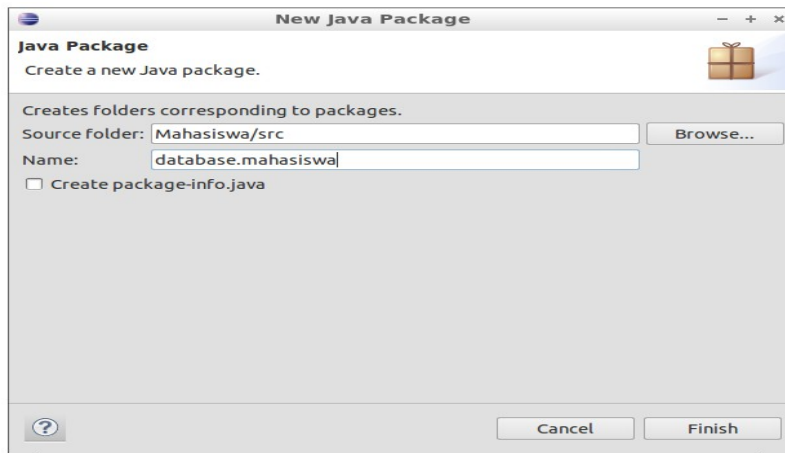


Setelah membuat Java Project, langkah selanjutnya adalah membuat **Package**. Untuk membuat Package klik kanan pada **src**-->**New**-->**Package**.

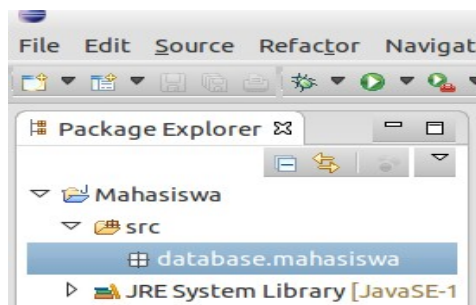


Kemudian beri nama Packagenya **database.mahasiswa** , Kemudian klik **Finish**.

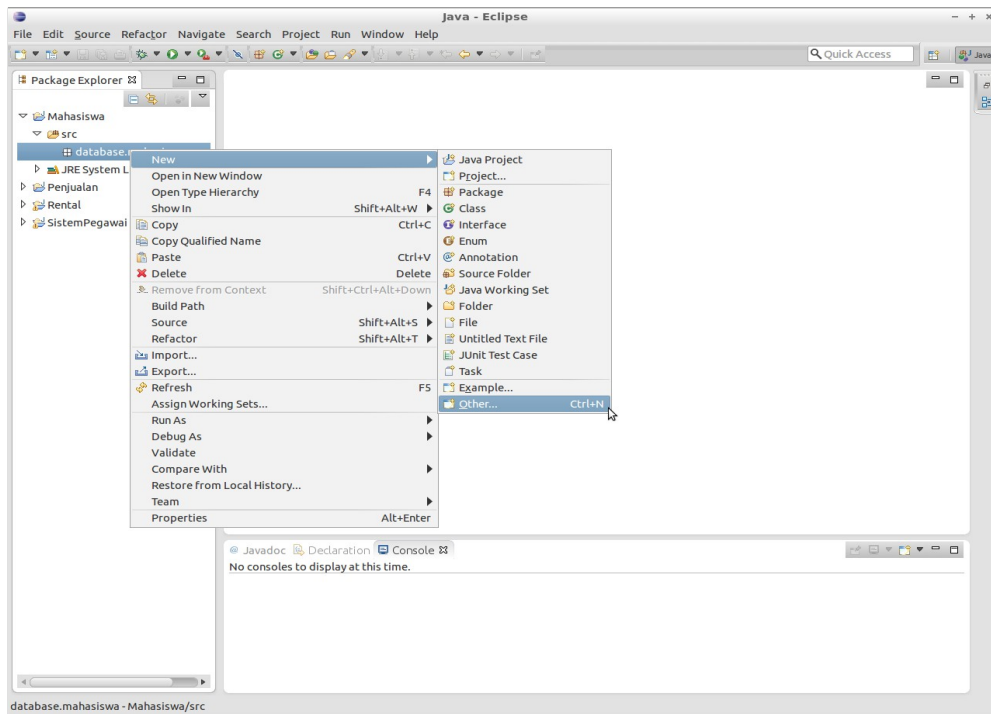
visit : [www.marisharingilmu.wordpress.com](http://www.marisharingilmu.wordpress.com)



Tampilan Package di Project Explorer

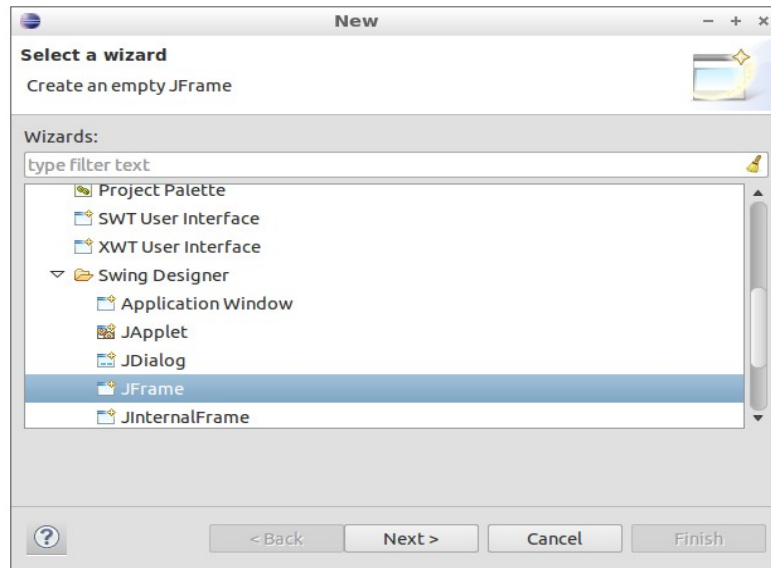


Untuk membuat tampilan Form. Klik kanan Package **database.mahasiswa**-->**New**-->**Other**.

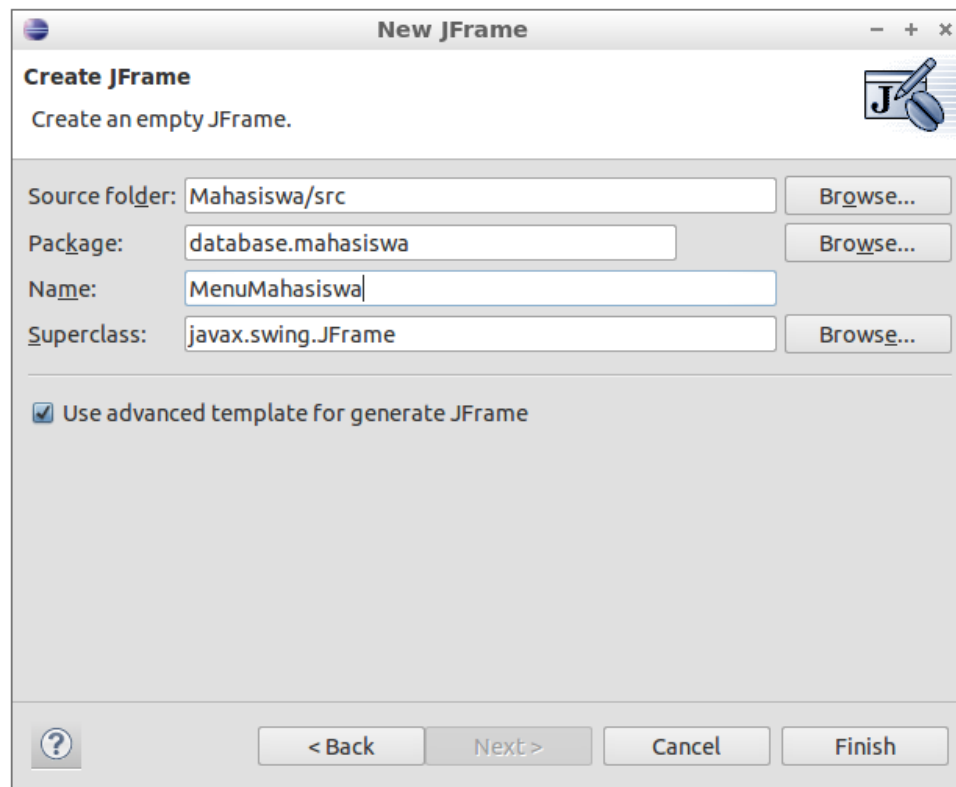


visit : [www.marisharingilmu.wordpress.com](http://www.marisharingilmu.wordpress.com)

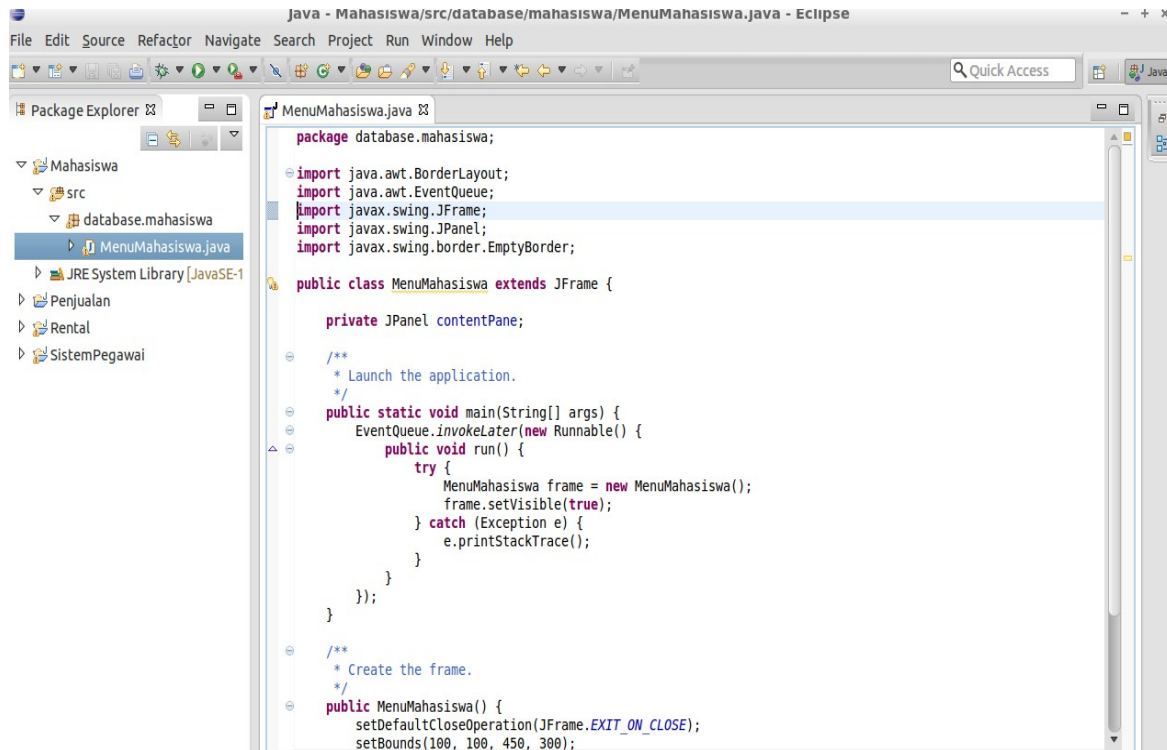
Dilanjutkan dengan klik **WindowBuilder-->Swing Designer-->JFrame** . Selanjutnya klik **Next**.



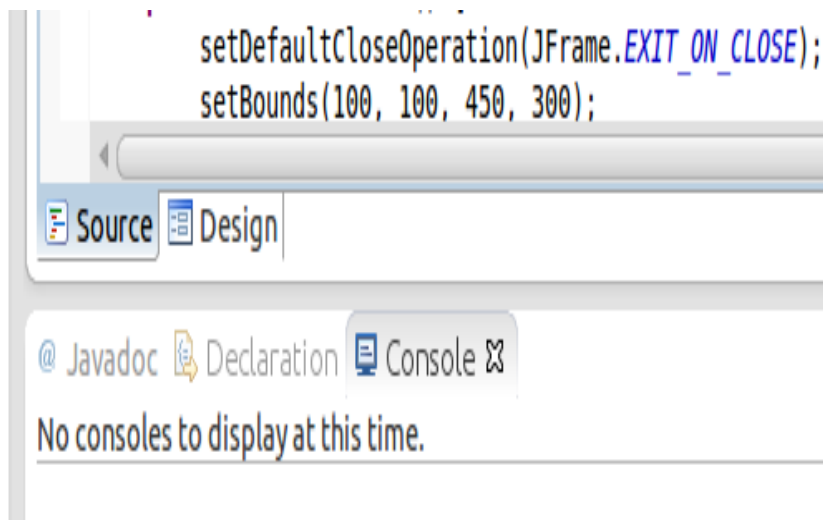
Beri nama JFramenya **MenuMahasiswa**.Lalu klik **Finish**.



Kalau berhasil anda akan mendapatkan tampilan code seperti berikut ini

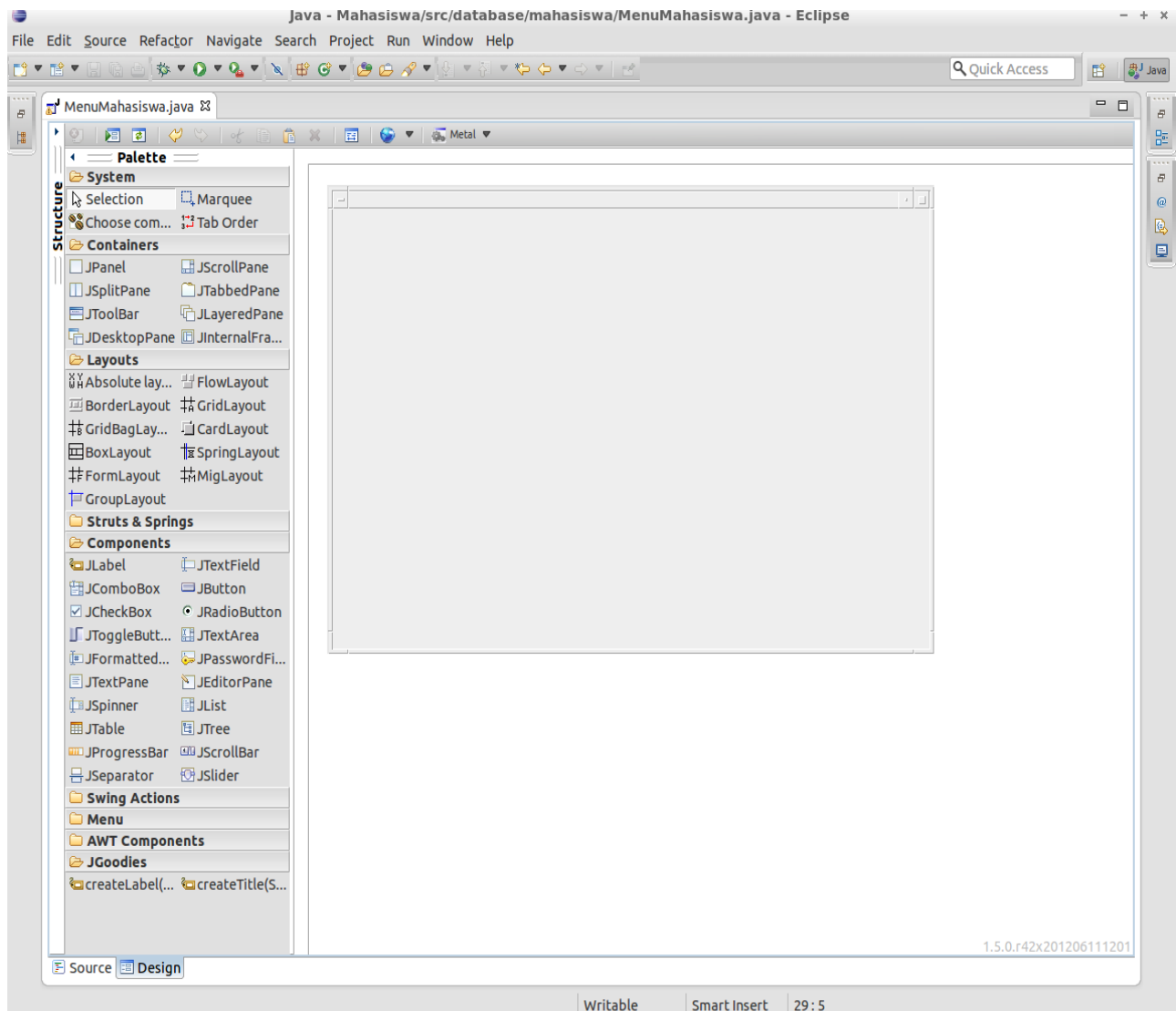


Untuk membuka GUI Buildernya klik pada tab **Design**.

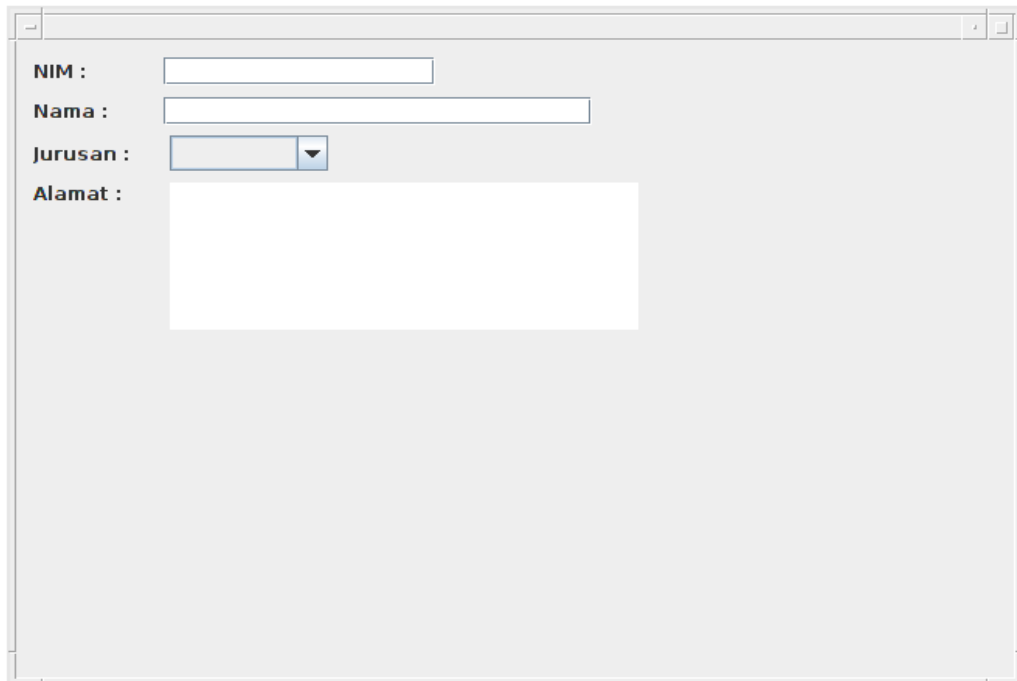




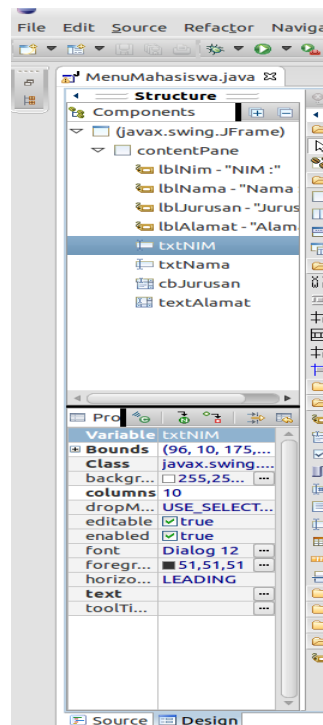
Berikut ini tampilan GUI Buildernya di Eclipse Juno



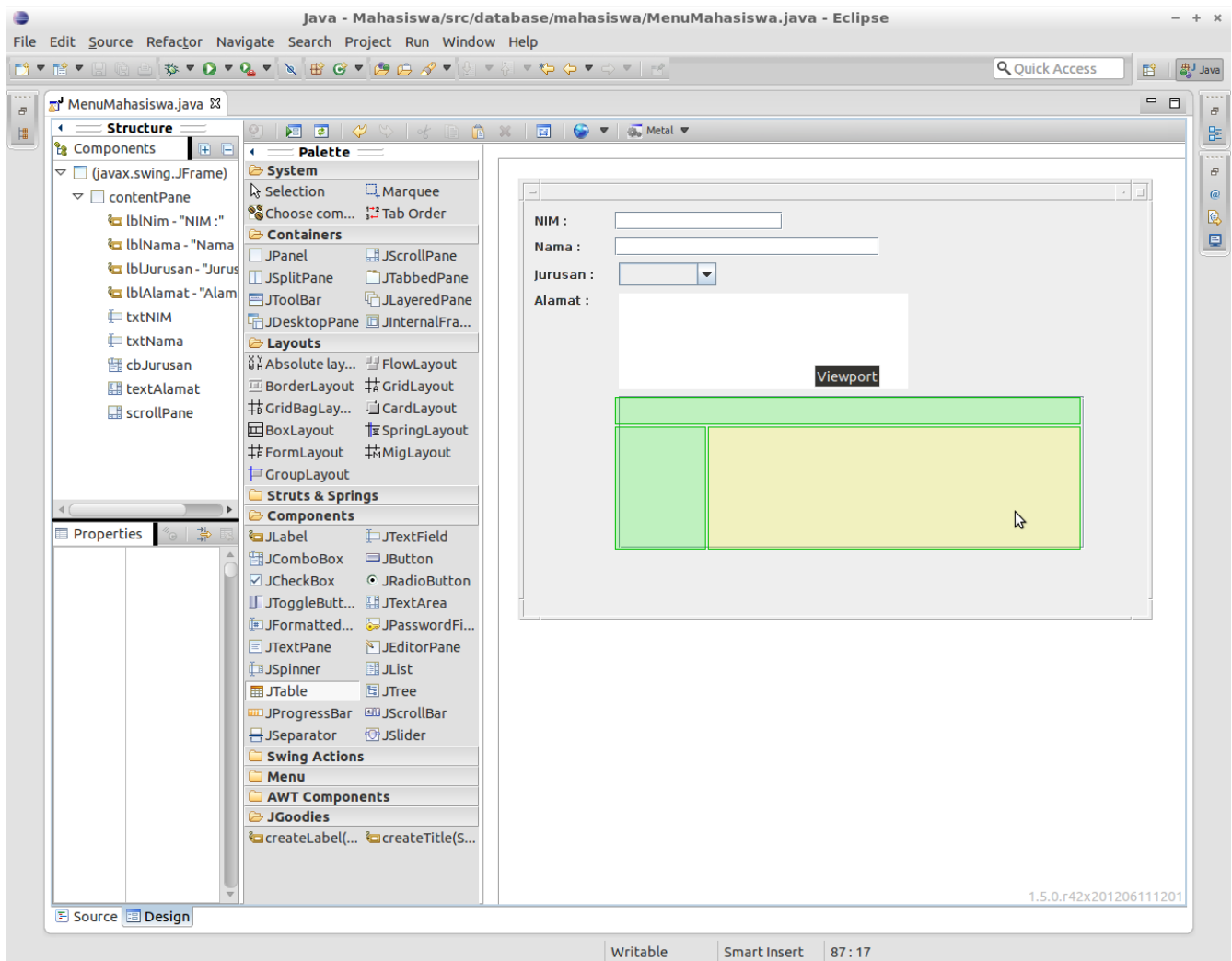
Di WindowBuilder terdapat komponen Java Swing dan AWT yang lengkap, seperti milik Netbeans. Untuk menggunakannya anda hanya drag & drop komponen yang dibutuhkan di Frame yang telah dibuat. Agar JFrame dapat menampung komponen Swing lainnya, pertama klik **Absolute Layout** pada menu **Layout**. Drag and drop **Absolute Layout** ke dalam JFrame. Nah, sekarang buatlah tampilan seperti berikut ini



Untuk Jurusan menggunakan komponen **JComboBox** dan untuk Alamat menggunakan komponen **JTextArea**. Anda juga dapat mengubah nama variabel sebuah komponen di menu **Structure**.



Untuk membuat tabel masukkan dulu komponen **JScrollPane** lalu tempatkan **JTable** di dalam **JScrollPane** pada **ViewPortnya**. **JScrollPane** berfungsi untuk membuat scroll ketika data yang ada di tabel berjumlah banyak.



Untuk membuat header field pada tabel. Tambahkan kode berikut ini di source codenya

```
import javax.swing.table.DefaultTableModel;
```

```
import java.awt.EventQueue;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.JLabel;
import javax.swing.JTextField;
import javax.swing.JComboBox;
import javax.swing.JTextArea;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;
```

visit : [www.marisharingilmu.wordpress.com](http://www.marisharingilmu.wordpress.com)

Dilanjutkan dengan membuat array 1 dimensi untuk membuat header field tablenya dan deklarasi DefaultTableModel.

```
String header[] = {"NIM", "Nama", "Jurusan", "Alamat"};  
DefaultTableModel tabelModel;
```

```
String header[] = {"NIM", "Nama", "Jurusan", "Alamat"};  
DefaultTableModel tabelModel;  
  
/**  
 * Launch the application.  
 */  
public static void main(String[] args) {  
    EventQueue.invokeLater(new Runnable() {  
        public void run() {
```

Langkah selanjutnya adalah menempatkan nama headernya pada tabel. Berikut ini listing programnya

```
tabelModel = new DefaultTableModel(null, header);  
tabel = new JTable();  
tabel.setModel(tabelModel);
```

```
scrollPane.setBounds(100, 227, 446, 116);  
contentPane.add(scrollPane);  
  
tabelModel = new DefaultTableModel(null, header);  
tabel = new JTable();  
tabel.setModel(tabelModel);  
scrollPane.setViewportView(tabel);
```

Pada code tersebut terdapat value **null**, sebenarnya null nanti kita isi dengan data yang kita ambil dari database. Berikut ini tampilan ketika program dijalankan. Sekarang tabel sudah memiliki nama pada headernya.

NIM :

Nama :

Jurusan :

Alamat :

NIM	Nama	Jurusan	Alamat

Sekarang kita tambahkan komponen tombol Simpan, Update, Hapus .Untuk proses manipulasi data. Drag and drop komponen JButton ke dalam frame utama.Berikut ini tampilan ketika JButton sudah ditambahkan di frame utama.

NIM :

Nama :

Jurusan :

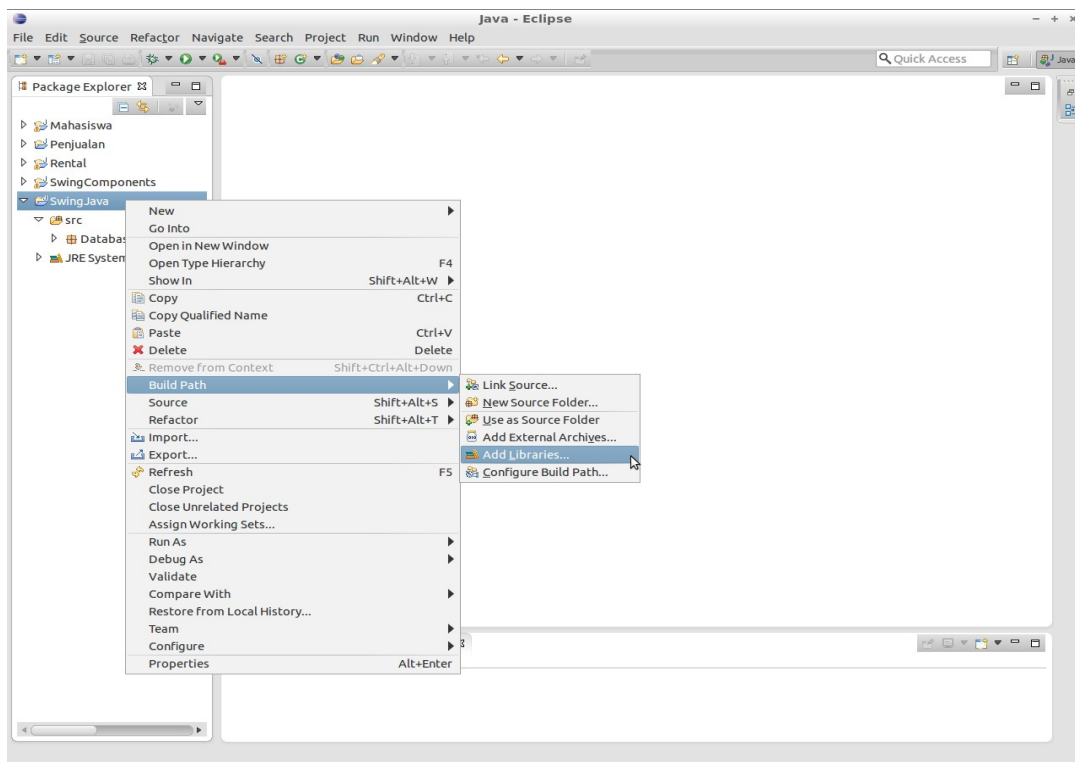
Alamat :

NIM	Nama	Jurusan	Alamat

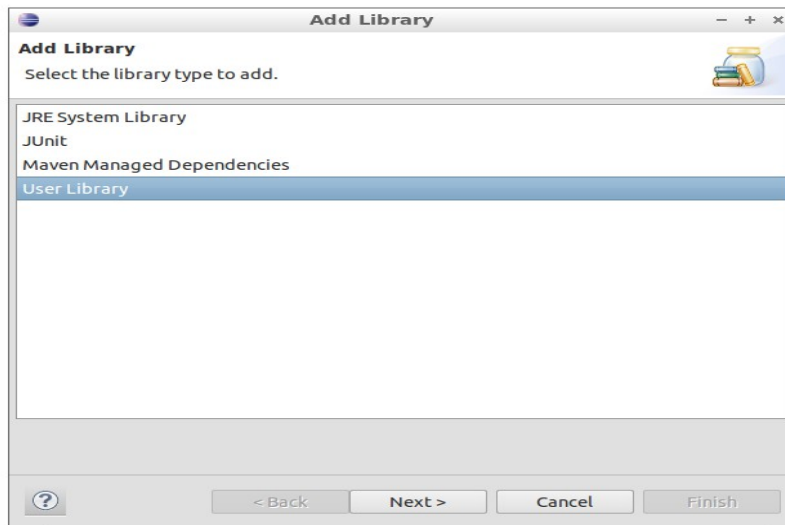
Simpan      Update      Hapus

Agar dapat koneksi dengan DBMS MySQL, dibutuhkan library MySQL Connector. Berikut ini cara langkah – langkah menambahkan library MySQL Connector di Eclipse.

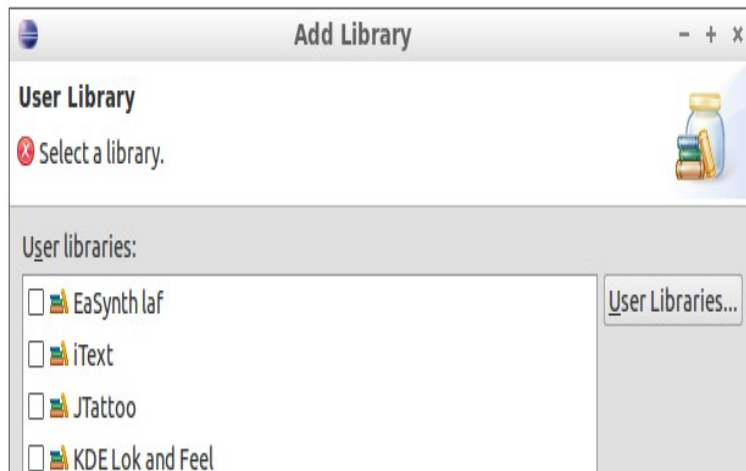
Untuk menambahkan librarynya, pertama klik kanan nama project Javanya.Lalu klik pada **Build Path**–> **Add Libraries**



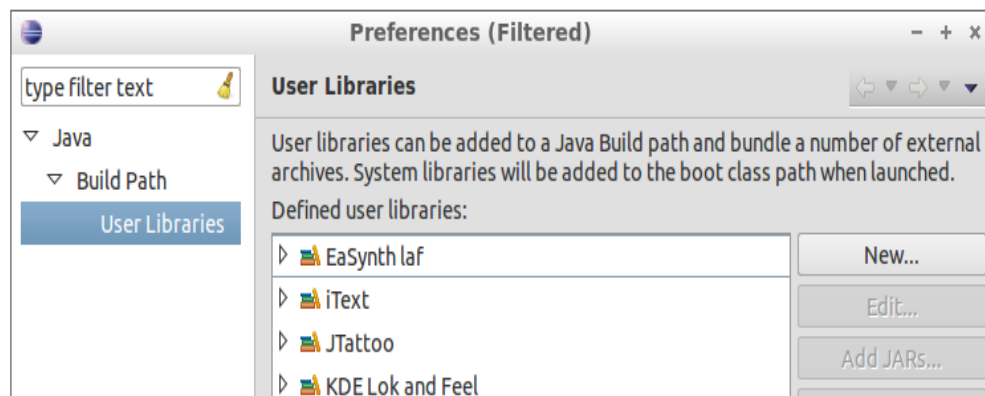
Pada menu **Add Library**, pilih **User Library**.Kalau sudah klik **Next**.



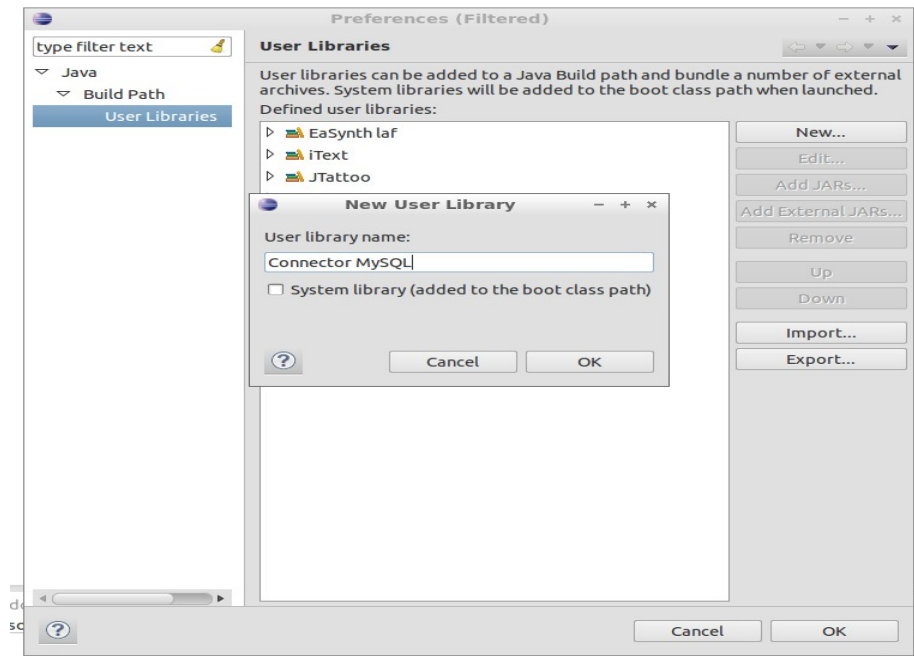
Selanjutnya pada menu **Add Library** pilih **User Libraries**.



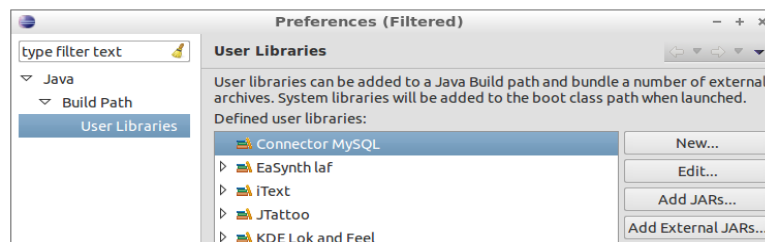
Selanjutnya anda akan dibawa ke menu **Preferences (Filtered)** ,lalu pilih **New**.



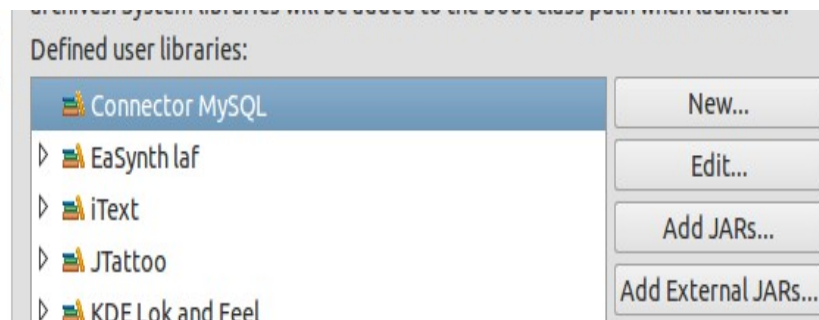
Di menu **New User Library**, berikan nama librarynya.Contoh **Connector MySQL**.



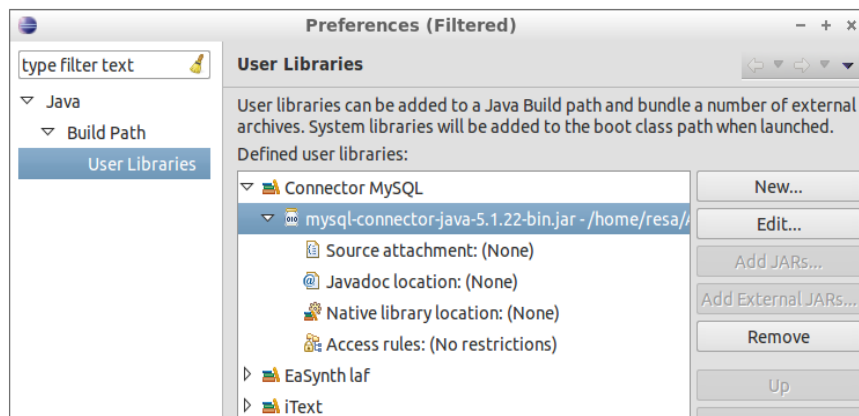
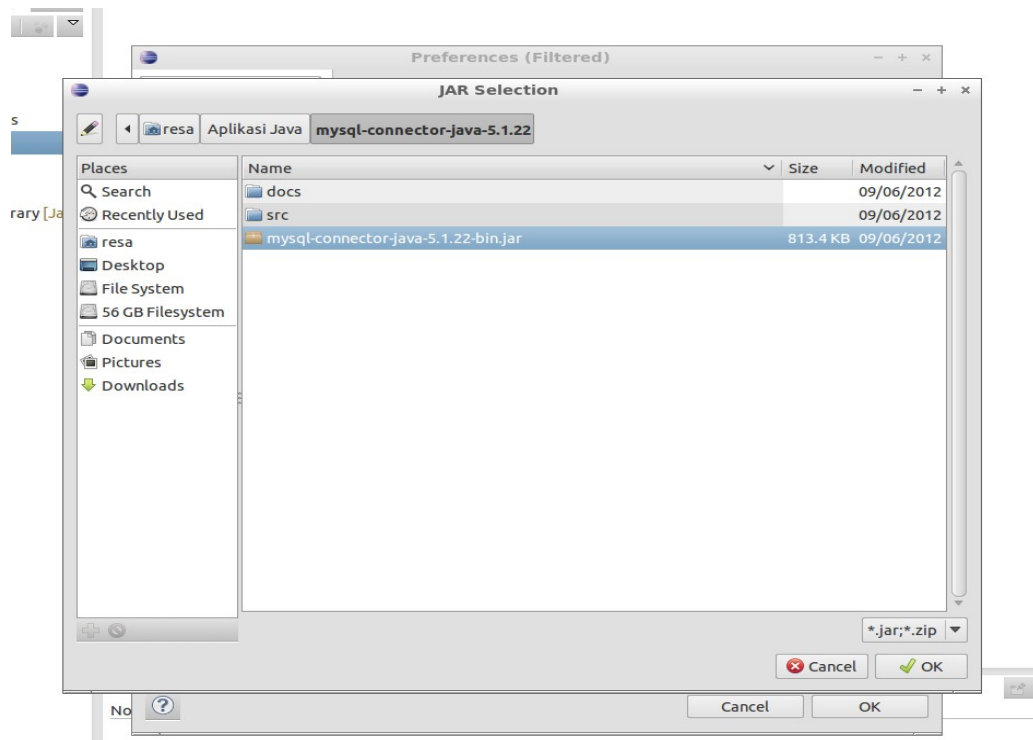
Kalau berhasil, anda akan mendapatkan nama library **Connector MySQL**.



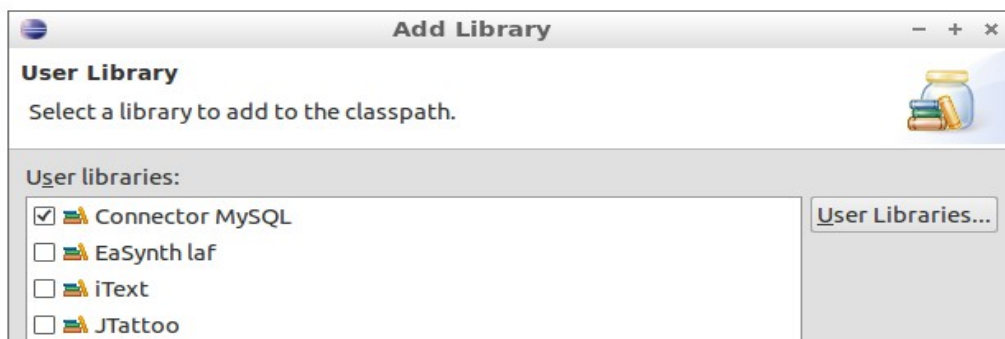
Langkah selanjutnya, klik **Add External JARs**, dan cari lokasi **MySQL Connector.jar** nya. Kalau sudah klik **Ok**, kemudian klik **Ok** lagi.



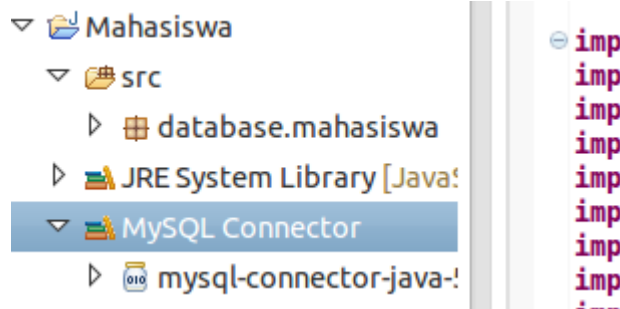




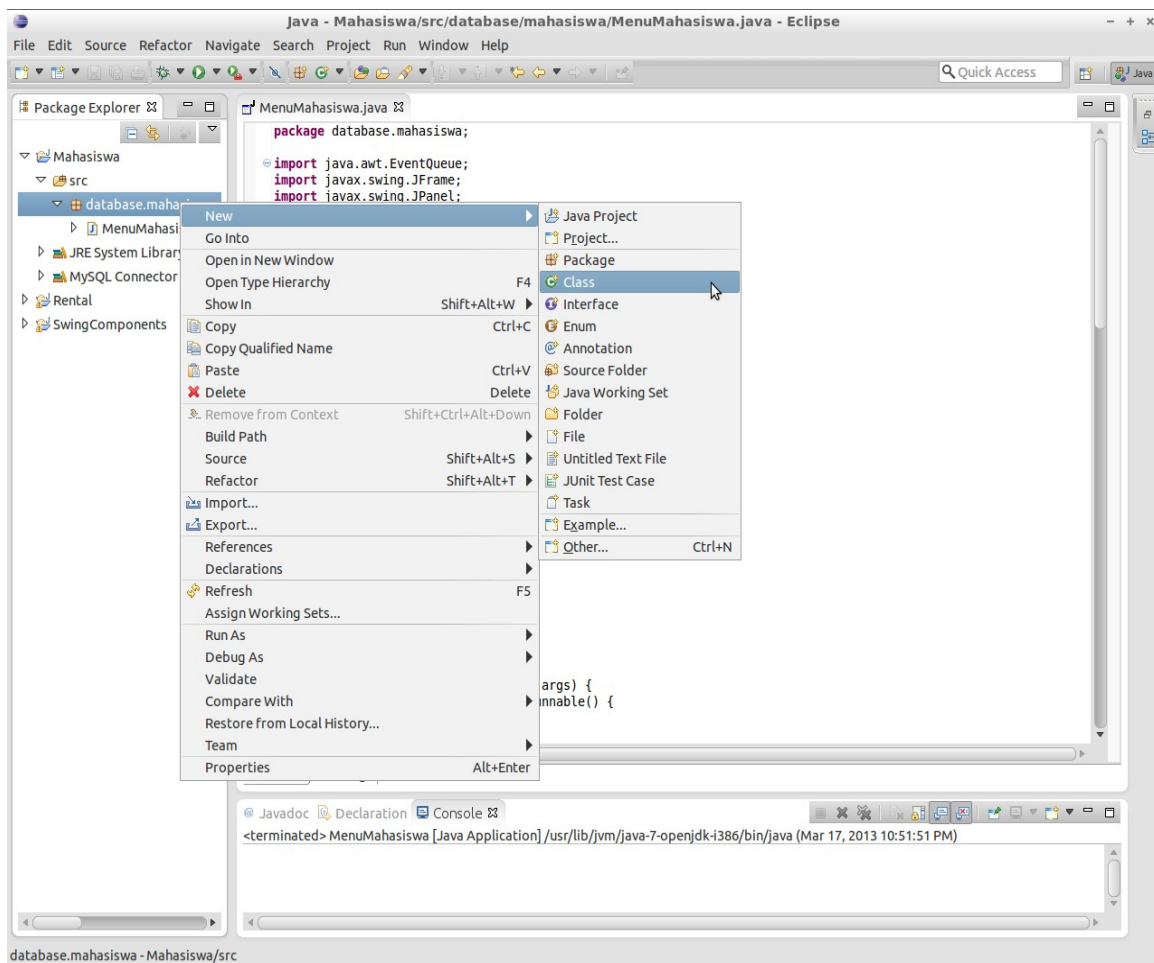
Untuk menggunakannya, berikan tanda checklist pada library yang akan digunakan pada projectnya. Kalau sudah klik **Finish**.

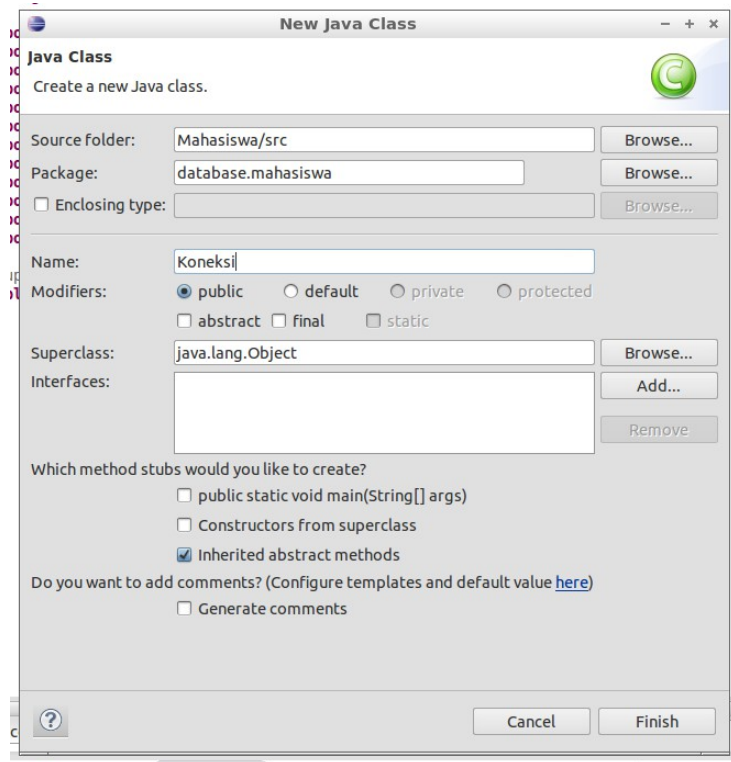


Sekarang cek pada bagian bawah Java Projectnya. Kalau Connector MySQL sudah muncul, berarti anda telah sukses menambahkan Library di Eclipse.



Setelah menambahkan library MySQL Connector, saatnya membuat class koneksi. Class koneksi berfungsi untuk menghubungkan antara program Java dan MySQL lewat MySQL Connector. Karena di dalam class koneksi ada nama database, user MySQL, password MySQL, dll. Untuk membuat class koneksi klik packagenya dilanjutkan dengan **New-->Class** dan berikan nama classnya **Koneksi**.





Berikut ini listing program class Koneksi

```
import java.sql.DriverManager;
import java.sql.Connection;

public class Koneksi
{
    private static Connection koneksi;

    public static Connection getKoneksi()
    {
        if(koneksi == null)
        {
            try
            {
                String url = "jdbc:mysql://localhost/universitas";
                String username = "root";
                String password = "root";

                DriverManager.registerDriver(new com.mysql.jdbc.Driver());
                koneksi =
DriverManager.getConnection(url,username,password);
            }
            catch(Exception ex)
            {
                System.out.println(ex);
            }
        }
        return koneksi;
    }
}
```

visit : [www.marisharingilmu.wordpress.com](http://www.marisharingilmu.wordpress.com)

```
}
```

```
import java.sql.DriverManager;  
import java.sql.Connection;  
  
public class Koneksi  
{  
    private static Connection koneksi;  
  
    public static Connection getKoneksi()  
    {  
        if(koneksi == null)  
        {  
            try  
            {  
                String url = "jdbc:mysql://localhost/universitas";  
                String username = "root";  
                String password = "root";  
  
                DriverManager.registerDriver(new com.mysql.jdbc.Driver());  
                koneksi = DriverManager.getConnection(url,username,password);  
            }  
            catch(Exception ex)  
            {  
                System.out.println(ex);  
            }  
        }  
        return koneksi;  
    }  
}
```

Pada variabel **url**, sesuaikan dengan nama databasenya. Untuk variabel **username** dan **password**, diisi dengan user dan password MySQL sesuai dengan komputer masing – masing. Pada contoh tersebut saya menggunakan user root dan password root.

Semuanya sudah lengkap, sekarang saatnya membuat proses insert / simpan data yang kita input masuk ke database. Klik kanan button **Simpan** , pilih **Add event handler-->action-->actionPerformed**.



```

String jurusan = "";
        if(cbJurusan.getSelectedIndex() == 0)
        {
            jurusan = "TI";
        }
        else if(cbJurusan.getSelectedIndex() == 1)
        {
            jurusan = "SI";
        }
        else if(cbJurusan.getSelectedIndex() == 2)
        {
            jurusan = "Ekonomi";
        }
        try
        {
            Connection konek = Koneksi.getKoneksi();
            String query = "INSERT INTO mahasiswa
VALUES(?,?,?,?)";
            PreparedStatement prepare =
            konek.prepareStatement(query);

            prepare.setInt(1,Integer.parseInt(txtNIM.getText()));
            prepare.setString(2, txtNama.getText());
            prepare.setString(3, jurusan);
            prepare.setString(4, textAlamat.getText());

            prepare.executeUpdate();
            JOptionPane.showMessageDialog(null,"Data berhasil
ditambahkan ke database");
        }
        catch(Exception ex)
        {
            JOptionPane.showMessageDialog(null,"Data gagal
ditambahkan ke database");
            System.out.println(ex);
        }
    }
}

```

```

String jurusan = "";
if(cbJurusan.getSelectedIndex() == 0)
{
    jurusan = "TI";
}
else if(cbJurusan.getSelectedIndex() == 1)
{
    jurusan = "SI";
}
else if(cbJurusan.getSelectedIndex() == 2)
{
    jurusan = "Ekonomi";
}
try
{
    Connection konek = Koneksi.getKoneksi();
    String query = "INSERT INTO mahasiswa VALUES(?,?,?,?)";
    PreparedStatement prepare = konek.prepareStatement(query);

    prepare.setInt(1,Integer.parseInt(txtNIM.getText()));
    prepare.setString(2, txtNama.getText());
    prepare.setString(3, jurusan);
    prepare.setString(4, textAlamat.getText());

    prepare.executeUpdate();
    JOptionPane.showMessageDialog(null,"Data berhasil ditambahkan ke database");
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null,"Data gagal ditambahkan ke database");
    System.out.println(ex);
}

```

Program tersebut masih proses insert ke dalam database. Berikutnya setelah data disimpan ke dalam database maka otomatis data tersebut tampil di tabel. Tambahkan syntax berikut untuk tampil di tabel. Pertama buat method **getDataTable()**. Dan tambahkan kode berikut.

```

public void getDataTable()
{
    try
    {
        Connection konek = Koneksi.getKoneksi();
        Statement state = konek.createStatement();
        String query = "SELECT * FROM mahasiswa";

        ResultSet rs = state.executeQuery(query);
        while(rs.next())
        {
            Object obj[] = new Object[4];
            obj[0] = rs.getInt(1);
            obj[1] = rs.getString(2);
            obj[2] = rs.getString(3);
            obj[3] = rs.getString(4);

            tabelModel.addRow(obj);
        }
        rs.close();
        state.close();
    }
    catch(Exception ex)
    {
    }
}

```

```

        btnHapus.setBounds(399, 366, 117, 25);
        contentPane.add(btnHapus);
    }

    public void getDataTable()
    {
        try
        {
            Connection konek = Koneksi.getKoneksi();
            Statement state = konek.createStatement();
            String query = "SELECT * FROM mahasiswa";

            ResultSet rs = state.executeQuery(query);
            while(rs.next())
            {
                Object obj[] = new Object[4];
                obj[0] = rs.getInt(1);
                obj[1] = rs.getString(2);
                obj[2] = rs.getString(3);
                obj[3] = rs.getString(4);

                tabelModel.addRow(obj);
            }
            rs.close();
            state.close();
        }
        catch(Exception ex)
        {
        }
    }
}

```

Pada code tersebut ada **addRow(obj)**, maksudnya adalah bahwa data yang kita ambil dari database akan ditambahkan pada setiap baris tabel. Untuk menampilkan data kita gunakan **ResultSet** dan **executeQuery()**. Berbeda dengan proses insert, update, ataupun hapus, karena proses tersebut menggunakan **executeUpdate()**.

Setelah itu tempatkan method **getDataTable()** di bawah try dan catch action. Simpan dengan tambahan syntax **finally**. Jadi seperti berikut ini

```

finally
{
    getDataTable();
}

try
{
    Connection konek = Koneksi.getKoneksi();
    String query = "INSERT INTO mahasiswa VALUES(?,?,?,?)";
    PreparedStatement prepare = konek.prepareStatement(query);

    prepare.setInt(1, Integer.parseInt(txtNIM.getText()));
    prepare.setString(2, txtNama.getText());
    prepare.setString(3, jurusan);
    prepare.setString(4, textAlamat.getText());

    prepare.executeUpdate();
    JOptionPane.showMessageDialog(null, "Data berhasil ditambahkan ke database");

    prepare.close();
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null, "Data gagal ditambahkan ke database");
    System.out.println(ex);
}
finally
{
    getDataTable();
}

```

Agar data yang ada di dalam database ditampilkan di tabel setiap frame dijalankan tambahkan method **getTable()** di dalam konstruktor.



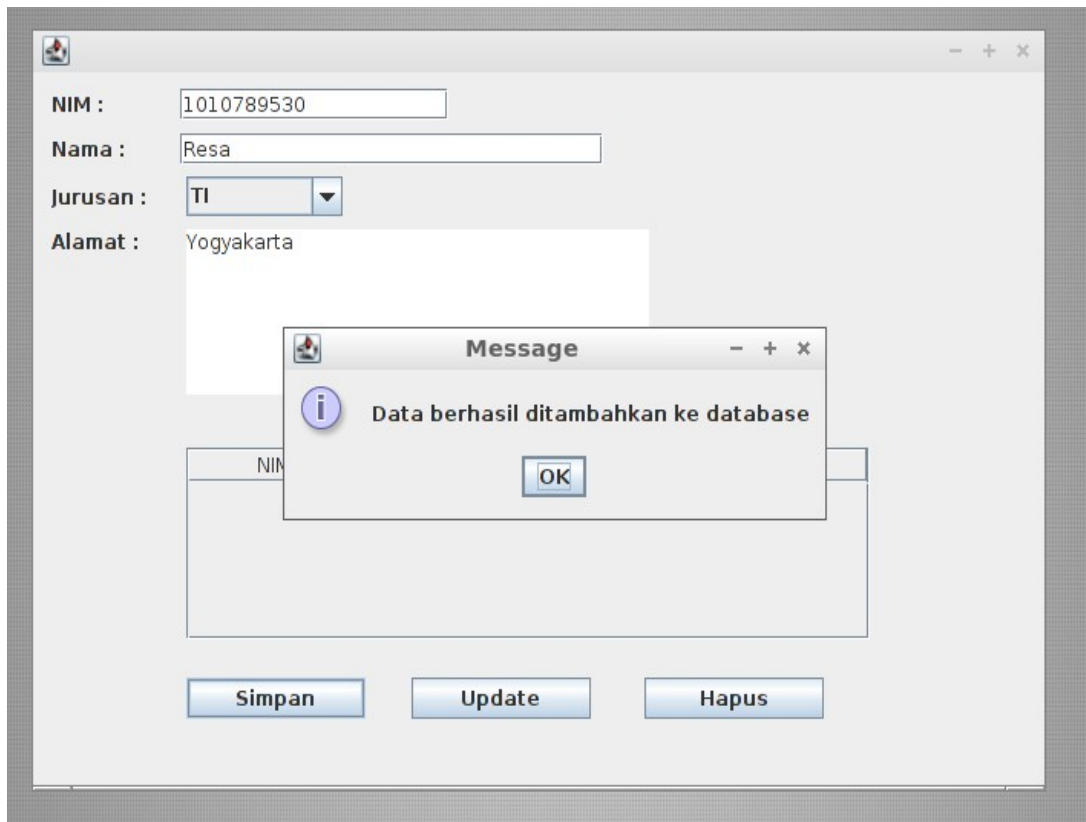
```

        btnUpdate = new JButton("Update");
        btnUpdate.setBounds(247, 366, 117, 25);
        contentPane.add(btnUpdate);

        btnHapus = new JButton("Hapus");
        btnHapus.setBounds(399, 366, 117, 25);
        contentPane.add(btnHapus);
        getDataTable();
    } //Akhir Konstruktor

```

Tampilan program insert ketika program dijalankan



NIM : 1010789530

Nama : Resa

Jurusan : TI

Alamat : Yogyakarta

NIM	Nama	Jurusan	Alamat
1010789530	Resa	TI	Yogyakarta

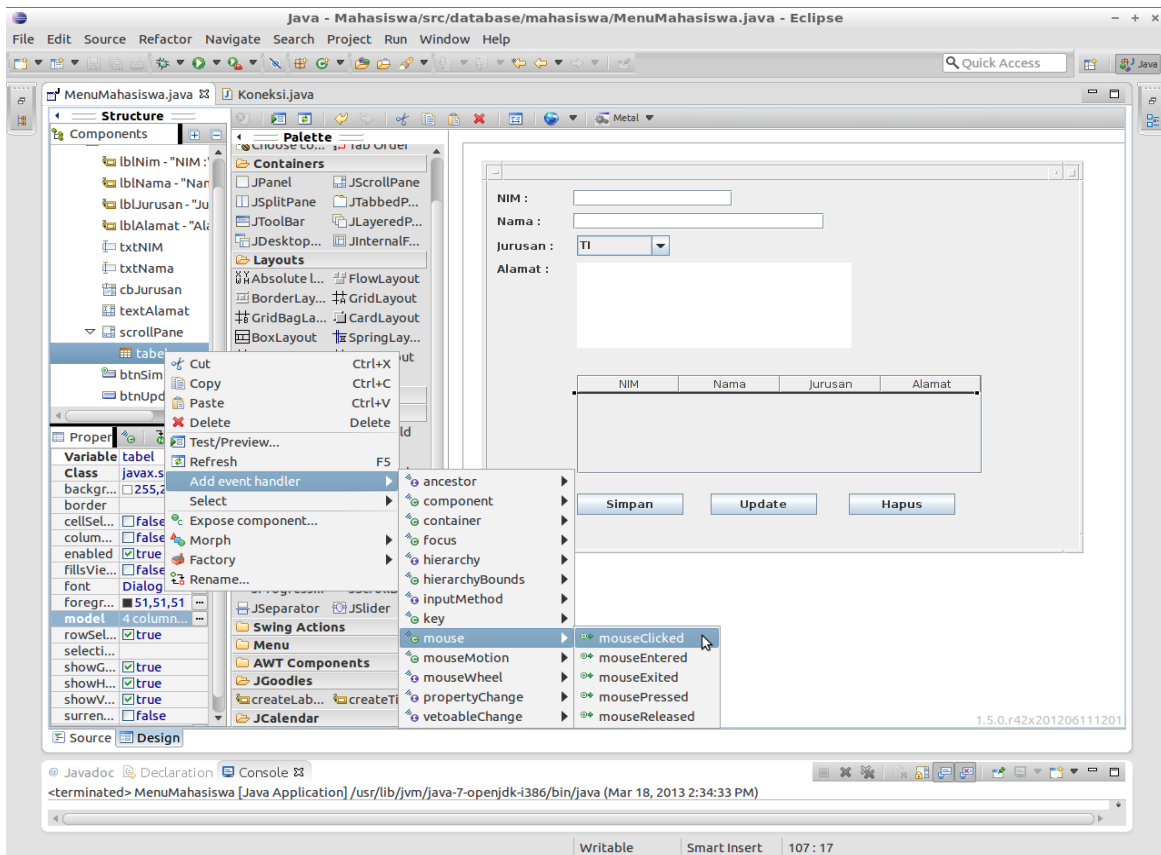
Simpan Update Hapus

Untuk dapat menampilkan data ke dalam TextField, ComboBox, dan TextArea yang diambil dari dalam tabel kita buat method baru dengan nama **getData()**. Dan tambahkan code berikut

```
public void getData()
{
    int pilih = tabel.getSelectedRow();
    if(pilih == -1)
    {
        return;
    }

    int nim = (int) tabelModel.getValueAt(pilih, 0);
    txtNIM.setText("" + nim);
    String nama = (String) tabelModel.getValueAt(pilih, 1);
    txtNama.setText(nama);
    String jurusan = (String) tabelModel.getValueAt(pilih, 2);
    cbJurusan.setSelectedItem(jurusan);
    String alamat = (String) tabelModel.getValueAt(pilih, 3);
    textAlamat.setText(alamat);
}
```

Kemudian tambahkan aksi mouse click pada tabel. Caranya klik kanan pada tabel pilih **Add event handler-->mouse-->mouseClicked**.



Kemudian tempatkan method **getData()** tadi di dalam action mouseClicked tabel.

```

tabel.addMouseListener(new MouseAdapter()
{
    @Override
    public void mouseClicked(MouseEvent e)
    {
        getData();
    }
});

```

Karena proses insert sudah, maka sekarang kita berikan aksi update data pada Button **Update**. Untuk proses pemberian action pada tombol update, caranya sama dengan proses pemberian action pada tombol simpan. Kalau sudah pada action tombol update tambahkan code seperti berikut

```

String jurusan = "";
        if(cbJurusan.getSelectedIndex() == 0)
        {
            jurusan = "TI";
        }
        else if(cbJurusan.getSelectedIndex() == 1)
        {
            jurusan = "SI";
        }
        else if(cbJurusan.getSelectedIndex() == 2)
        {
            jurusan = "Ekonomi";
        }
        try
        {
            Connection konek = Koneksi.getKoneksi();
            String query = "UPDATE mahasiswa SET Nama = ?,
Jurusan = ?, Alamat = ? WHERE NIM = ?";
            PreparedStatement prepare =
            konek.prepareStatement(query);

            prepare.setString(1, txtNama.getText());
            prepare.setString(2, jurusan);
            prepare.setString(3, textAlamat.getText());
            prepare.setInt(4,Integer.parseInt(txtNIM.getText()));

            prepare.executeUpdate();
            JOptionPane.showMessageDialog(null,"Data berhasil
diupdate");

            prepare.close();
        }
        catch(Exception ex)
        {
            JOptionPane.showMessageDialog(null,"Data gagal
diupdate");

            System.out.println(ex);
        }
    }
}

```

```

public void actionPerformed(ActionEvent act)
{
    String jurusan = "";
    if(cbJurusan.getSelectedIndex() == 0)
    {
        jurusan = "TI";
    }
    else if(cbJurusan.getSelectedIndex() == 1)
    {
        jurusan = "SI";
    }
    else if(cbJurusan.getSelectedIndex() == 2)
    {
        jurusan = "Ekonomi";
    }
    try
    {
        Connection konek = Koneksi.getKoneksi();
        String query = "UPDATE mahasiswa SET Nama = ?, Jurusan = ?, Alamat = ? WHERE NIM = ?";
        PreparedStatement prepare = konek.prepareStatement(query);

        prepare.setString(1, txtNama.getText());
        prepare.setString(2, jurusan);
        prepare.setString(3, textAlamat.getText());
        prepare.setInt(4, Integer.parseInt(txtNIM.getText()));

        prepare.executeUpdate();
        JOptionPane.showMessageDialog(null, "Data berhasil diupdate");

        prepare.close();
    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(null, "Data gagal diupdate");
        System.out.println(ex);
    }
}

```

Agar data yang diupdate dapat langsung ditampilkan di tabel tambahkan method **getDataTable()** setelah try dan catch dengan tambahan syntax finally

```

}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null, "Data gagal diupdate");
    System.out.println(ex);
}
finally
{
    getDataTable();
}

```

Pada method **getDataTable()** , tambahkan syntax berikut agar tidak terjadi data ganda yang ditampilkan pada tabel setelah proses update.

```

tabelModel.getDataVector().removeAllElements();
tabelModel.fireTableDataChanged();

```

```

public void getDataTable()
{
    tabelModel.getDataVector().removeAllElements();
    tabelModel.fireTableDataChanged();
    try
    {
        Connection konek = Koneksi.getKoneksi();

```

Proses terakhir adalah hapus data yang ada di dalam database. Berikan action pada button **Hapus**, caranya sama seperti pada button **Simpan dan Update**, setelah itu tambahkan syntax berikut

```

        try
        {
            Connection konek = Koneksi.getKoneksi();
            String query = "DELETE FROM mahasiswa WHERE NIM = ?";
            PreparedStatement prepare = konek.prepareStatement(query);

            prepare.setInt(1,Integer.parseInt(txtNIM.getText()));
            prepare.executeUpdate();

            JOptionPane.showMessageDialog(null,"Data berhasil
dihapus");

            prepare.close();
        }
        catch(Exception ex)
        {
            JOptionPane.showMessageDialog(null,"Data gagal
dihapus");

            System.out.println(ex);
        }
        finally
        {
            getDataTable();
        }
    }

    btnHapus.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent act)
        {
            try
            {
                Connection konek = Koneksi.getKoneksi();
                String query = "DELETE FROM mahasiswa WHERE NIM = ?";
                PreparedStatement prepare = konek.prepareStatement(query);

                prepare.setInt(1,Integer.parseInt(txtNIM.getText()));
                prepare.executeUpdate();

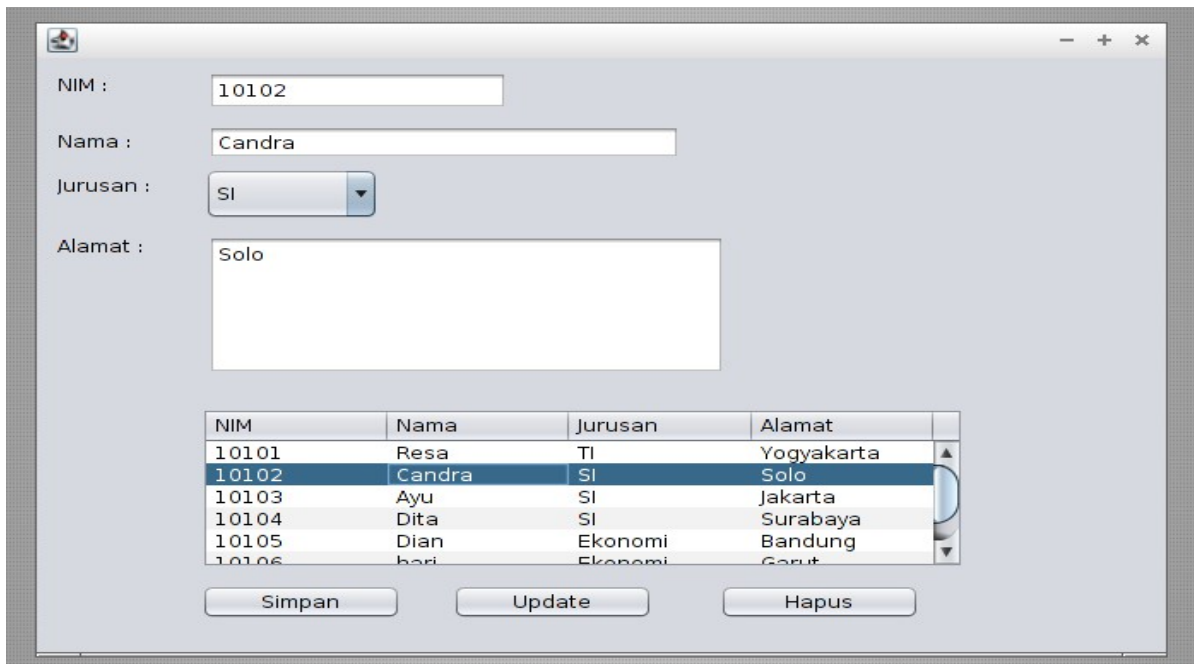
                JOptionPane.showMessageDialog(null,"Data berhasil dihapus");
                prepare.close();
            }
            catch(Exception ex)
            {
                JOptionPane.showMessageDialog(null,"Data gagal dihapus");
                System.out.println(ex);
            }
            finally
            {
                getDataTable();
            }
        }
    });

```

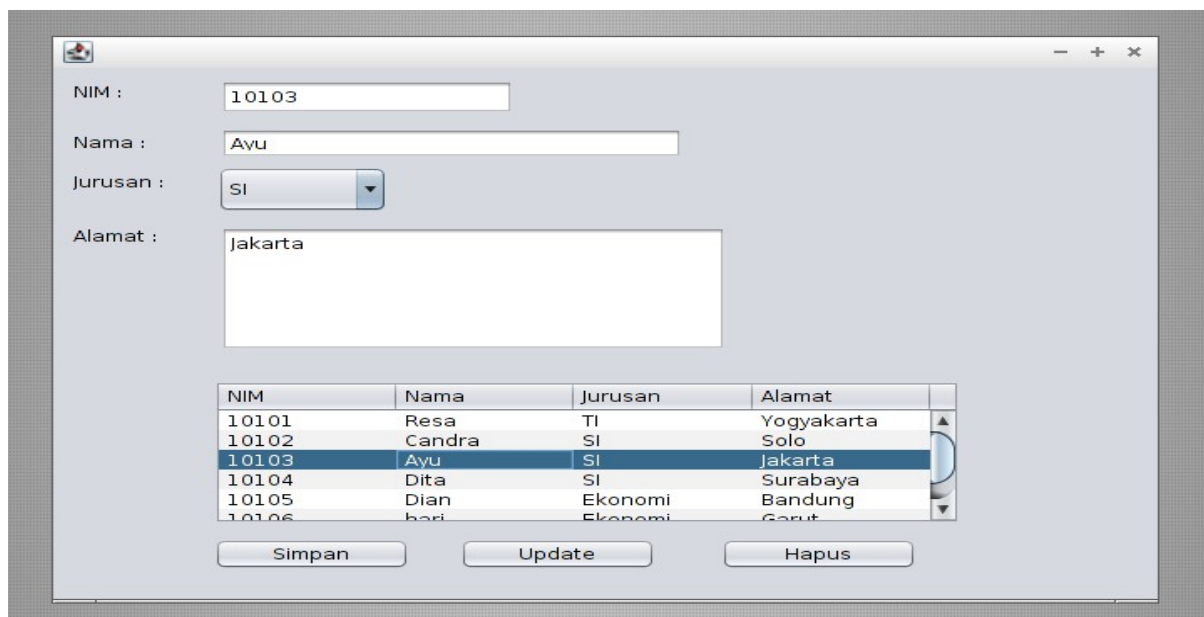
Sampai disini anda sudah dapat membuat aplikasi database dengan Java. Jika tampilan yang dibuat dirasa kurang menarik, anda dapat menambahkan look and feel Nimbus agar tampilan yang dibuat lebih cantik dan menarik. Tambahkan syntax berikut pada method main

```
UIManager.setLookAndFeel("com.sun.java.swing.plaf.nimbus.NimbusLookAndFeel");
```

```
public void run() {  
    try {  
        UIManager.setLookAndFeel("com.sun.java.swing.plaf.nimbus.NimbusLookAndFeel");  
        MenuMahasiswa frame = new MenuMahasiswa();  
        frame.setVisible(true);  
    } catch (Exception e) {  
        e.printStackTrace();  
    }  
}
```



NIM	Nama	Jurusan	Alamat
10101	Resa	TI	Yogyakarta
10102	Candra	SI	Solo
10103	Ayu	SI	Jakarta
10104	Dita	SI	Surabaya
10105	Dian	Ekonomi	Bandung
10106	hari	Ekonomi	Garut



NIM	Nama	Jurusan	Alamat
10101	Resa	TI	Yogyakarta
10102	Candra	SI	Solo
10103	Ayu	SI	Jakarta
10104	Dita	SI	Surabaya
10105	Dian	Ekonomi	Bandung
10106	hari	Ekonomi	Garut

NIM : 10101

Nama : Resa

Jurusan : TI

Alamat : Yogyakarta

NIM	Nama	Jurusan	Alamat	
10101	Resa	TI	Yogyakarta	
10102	Candra	SI	Solo	
10103	Ayu	SI	Jakarta	
10104	Dita	SI	Surabaya	
10105	Dian	Ekonomi	Bandung	
10106	hari	Ekonomi	Garut	

Simpan Update Hapus

Semoga bermanfaat :)

Author : Resa Cr  
Chief on [www.marisharingilmu.wordpress.com](http://www.marisharingilmu.wordpress.com)

visit : [www.marisharingilmu.wordpress.com](http://www.marisharingilmu.wordpress.com)