Modul Praktikum

Object Oriented Programing

Program Studi Teknik Informatika

STMIK STIKOM Indonesia

DAFTAR ISI

MODUL 1
Pengenalan Java disertai First Simple Program
(Pertemuan 1)

Tujuan :

Dengan praktikum ini mahasiswa diharapkan dapat:

1. Mengenal dan mempersiapkan lingkungan kerja Java
2. Membuat program sederhana dengan Java dengan Netbean
3. Mengkompilasi dan menjalankan program Java Netbean
4. Menganalisa beberapa problem yang terjadi saat pemrograman dan memberikan solusi

Tugas Pendahuluan :

1. Apa yang anda ketahui tentang Netbeans?
2. Sebutkan keunggulan bahasa pemrograman JAVA!

DASAR TEORI

Bahasa pemrograman Java pada awalnya dikembangkan oleh Sun Microsystems, yang digagas oleh James Gosling dan dirilis pada tahun 1995 sebagai komponen inti dari Sun Microsystems.s platform Java (Java 1.0 [J2SE]).

Sun Microsystems telah berganti nama menjadi J2 versi baru sebagai Java SE, Java EE dan Java ME masing-masing. Dalam java semuanya adalah Obyek. Java dapat dengan mudah diperpanjang karena didasarkan pada model Obyek. Platform independen: Tidak seperti banyak bahasa pemrograman lain termasuk C dan C + + ketika Jawa dikompilasi, tidak dikompilasi ke dalam mesin platform tertentu, bukan menjadi platform independen kode byte. Ini kode byte didistribusikan melalui web dan ditafsirkan oleh Mesin virtual (JVM) di mana platform itu yang dijalankan.

Java dirancang agar mudah untuk belajar. Jika Anda memahami konsep dasar OOP java akan mudah untuk menguasai. Dengan fitur aman Java memungkinkan untuk mengembangkan bebas virus, sistem tamper-bebas. Teknik otentikasi didasarkan pada enkripsi kunci public. Java compiler menghasilkan format file objek arsitektur netral yang membuat kode dikompilasi menjadi executable pada banyak prosesor, dengan hadirnya sistem runtime Java. Menjadi arsitektur netral dan tidak memiliki implementasi aspek tergantung dari spesifikasi Java membuat portabel. Compiler dan Java ditulis dalam ANSI C dengan batas portabilitas bersih yang merupakan subset POSIX. Java membuat upaya untuk menghilangkan situasi rawan kesalahan dengan menekankan terutama pada waktu kompilasi kesalahan pengecekan dan pemeriksaan runtime.

KEGIATAN PRAKTIKUM

1. Persiapan

1.1 Peralatan yang Diperlukan

Pada pelatihan Java Dasar ini, peralatan yang diperlukan adalah :

1. Java Development Kit versi 1.6 keatas.
2. Java Runtime Environtment versi 1.6 keatas.
3. NetBeans IDE versi 6.9 keatas.

1.2 Java Development Kit

Java Development Kit merupakan perangkat lunak yang digunakan untukmelakukan proses kompilasi dari kode Java menjadi *bytecode* yang dapat dimengerti dan dapat dijalankan oleh Java Runtime Environtment.

1.3 Java Runtime Environtment

Java Runtime Environtment merupakan perangkat lunak yang digunakan untukmenjalankan aplikasi yang dibangun menggunakan java. Versi JRE harus sama atau lebih tinggi dari JDK yang digunakan untuk membangun aplikasi agar aplikasi dapat berjalan sesuai dengan yang diharapkan.

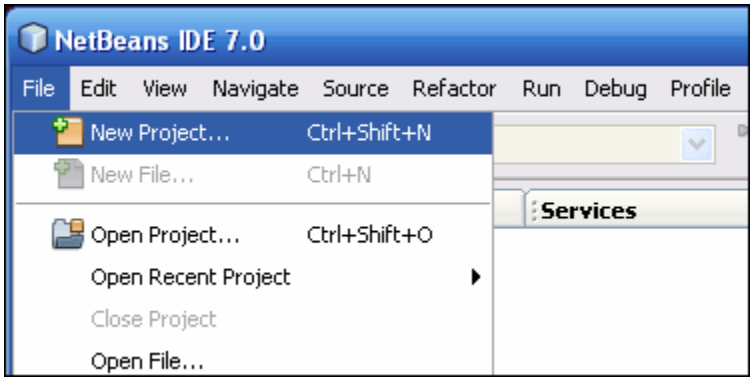
1.4 NetBeans IDE

NetBeans IDE merupakan perangkat lunak yang digunakan untuk membangunperangkat lunak yang lain. NetBeans IDE dapat digunakan untuk membangun perangkat lunak berbasis Java Standard Edition, Java Enterprise Edition, Java Micro Edition, JavaFX, PHP, C/C++, Ruby, Groovy dan Python.

2. Sintaks Cetak String Sederhana

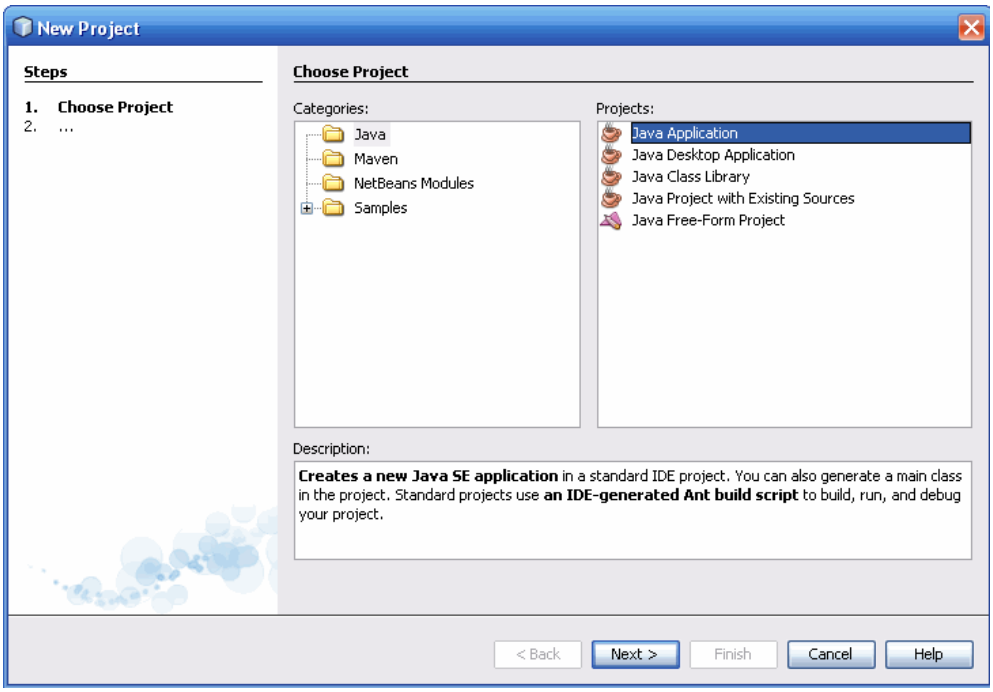
2.1 Membuat Project Sederhana

Buka Netbeans pada PC anda, kemudian klik **File | New Project** Seperti contoh pada **Gambar 1.1** berikut :



Gambar 1.1 Create New Project

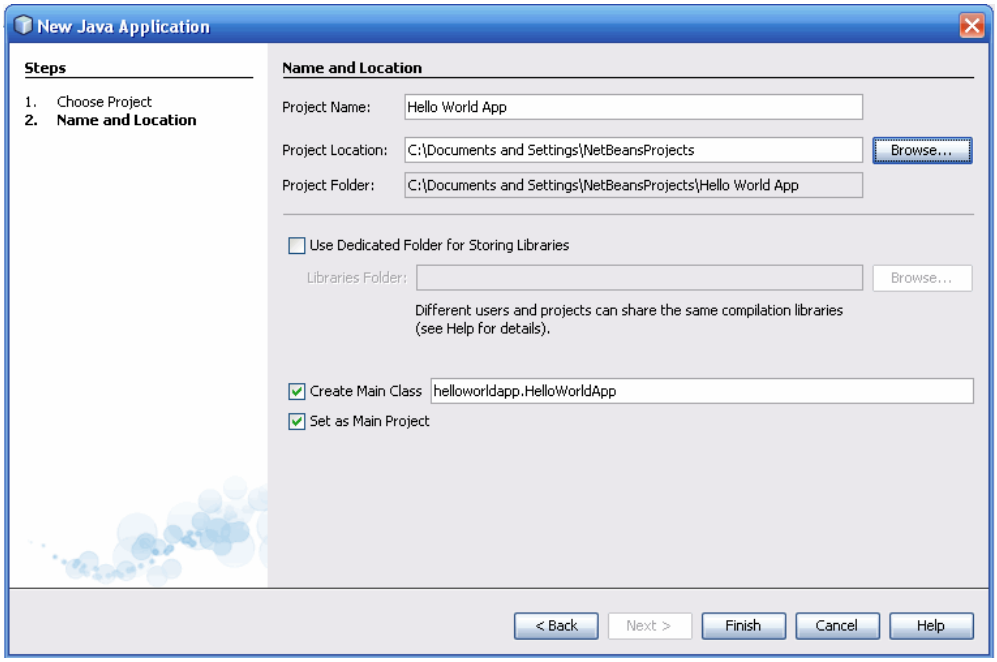
Dalam New Project wizard, klik **Java** pada Categories. Selanjutnya klik **Java Application** pada Projects seperti yang ditunjukkan oleh **Gambar 1.2** berikut:



Gambar 1.2 Create New Project 2

Dalam **Name and Location**, lakukan hal berikut (seperti yang ditunjukkan pada **Gambar 1.3** di bawah):

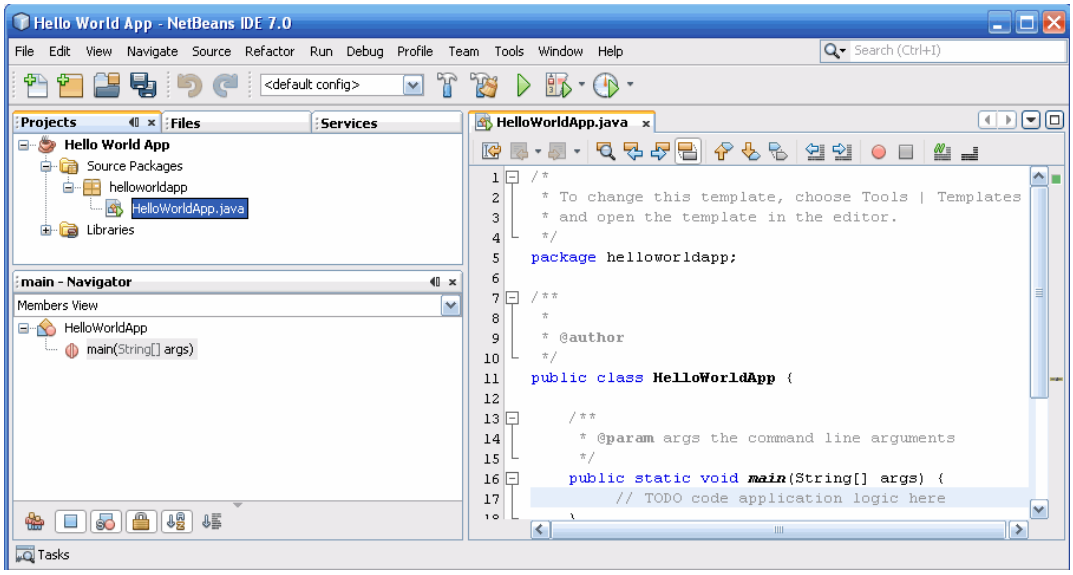
- Pada Project Name field, ketik Hello World App.
- Pada Create Main Class field, Ketik helloworldapp.HelloWorldApp.
- Biarkan pilihan Set as Main Project tercentang.



Gambar 1.3 Create New Project 3

Proyek dibuat dan dibuka dalam IDE. Anda akan melihat komponen-komponen seperti yang ditampilkan pada **Gambar 1.4** :

- Jendela **Projects**, yang berisi tampilan pohon dari berbagai komponen proyek, berisi file sumber, pustaka yang mendasari kode Anda tergantung pada, dan sebagainya.
- Jendela **Source Editor** dengan sebuah file yang bernama HelloWorldApp terbuka.
- Jendela **Navigator**, yang dapat Anda gunakan untuk navigasi cepat di antara elemen-elemen dalam kelas yang dipilih.



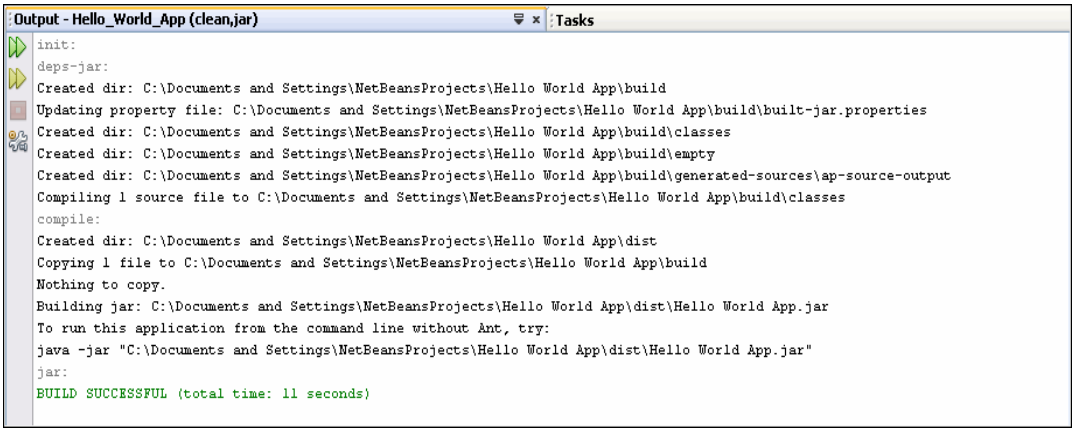
Gambar 1.4 Create New Project 4

Selanjutnya lakukan pengkodean pada listing Project, dengan memasukan syntax `System.out.println("Hello World!");` di bagian `public static void main(String[] args)` atau tepat di bawah komentar `//TODO code application logic here`. Perhatikan contoh berikut:

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package helloworldapp;

/**
 * The HelloWorldApp class implements an application that
 * simply prints "Hello World!" to standard output.
 */
public class HelloWorldApp {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {

      // TODO code application logic
        here System.out.println("Hello
        World!");
    }
}
```

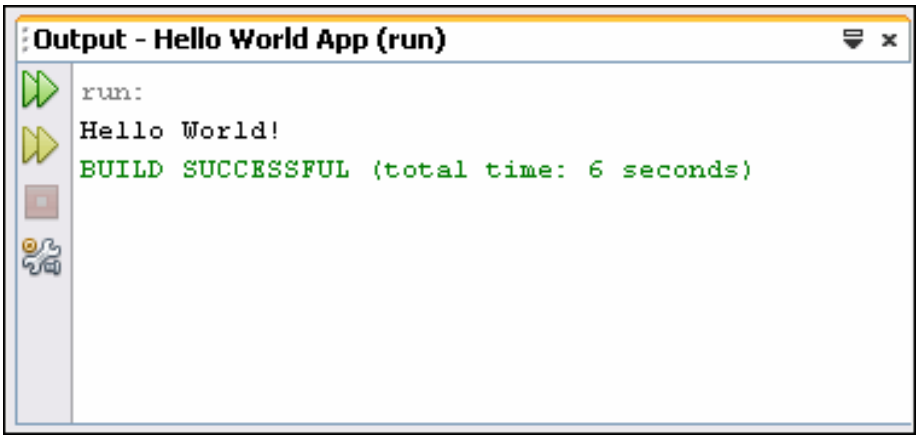
Pemrograman Java merupakan salah satu bahasa **pemrograman yang Case Sensitive** maka dari itu perhatikan huruf besar dan huruf kecil seperti contoh a (a kecil) dan A (A Besar). Untuk memberikan komentar pada listing code dengan menggunakan tanda // atau /* untuk mengawali komentar dan */ untuk mengakhiri komentar. Untuk menuliskan String, Anda dapat memberikan tanda petik dua pada awal kalimat dan akhir kalimat seperti “Hello World!”.

Untuk mengkompilasi file sumber Anda, pilih **Run | Build Main Project** dari menu utama IDE atau menggunakan shortcut Compile File **(F9).** Jendela Output terbuka dan menampilkan hasil yang serupa dengan apa yang Anda lihat pada **Gambar 1.5** berikut:



Gambar 1.5 Running Project

Jika hasil pembangunan diakhiri dengan pernyataan **BUILD SUCCESSFUL**, selamat! Anda telah berhasil mengkompilasi program! Jika hasil Build diakhiri dengan pernyataan **BUILD FAILED**, Anda mungkin memiliki kesalahan sintaks dalam kode Anda. Kesalahan dilaporkan dalam jendela Output sebagai teks hyperlink. Klik dua kali pada hyperlink untuk menavigasi ke sumber kesalahan. Anda kemudian dapat memperbaiki kesalahan dan sekali lagi pilih **Run | Build Main Project.** Jika proses compile sukses maka langkah selanjutnya adalah RUN project anda dengan cara pilih **Run | Run Main Project** atau menggunakan shortcut Run File **(Shift-F6).** Setelah itu Anda akan melihat tampilan Hello World seperti pada **Gambar 1.6** berikut :



Gambar 1.6 Build Project Sukses

Percobaan 1 : Menampilkan suatu tulisan ke layar

```
Hallo.java

public class Hallo {
  public static void main(String args[]) {
   System.out.println("Hallo...");
  }
}
```

Percobaan 2 : Melibatkan suatu class dalam program

```
TestGreeting.java

public class TestGreeting {
  public static void main (String[] args) {
   Greeting hello = new Greeting();
   hello.greet();
  }
}

Greeting.java

public class Greeting {
  public void greet() {
   System.out.println("hi");
  }
}
```

Latihan

Latihan 1 : Menganalisa dan membenahi kesalahan pada program

Tulislah program berikut ini dan simpanlah dengan nama **Test.java**

```
Test.java

public class Testing {

  public static void main(String[] args) {
   System.out.println("What's wrong with this program?");

  }

}
```

Lakukan kompilasi pada file tersebut dan amati hasilnya. Kenapa terjadi kegagalan pada saat kompilasi? Benahilah kesalahan tersebut sehingga program dapat berjalan dengan baik.

Latihan 2 : Menganalisa dan membenahi kesalahan pada program

Tulislah program di bawah ini dan simpanlah dengan nama tertentu. Lakukan kompilasi pada file tersebut dan amati hasilnya. Kenapa terjadi kegagalan pada saat kompilasi? Benahilah kesalahan tersebut sehingga program dapat berjalan dengan baik.

```
public class Test {
   public static void main(String[] args) {
      System.out.println("What's wrong with this program?");
   }
}
```

Latihan 3 : Menganalisa dan membenahi kesalahan pada program

Tulislah program di bawah ini dan simpanlah dengan nama tertentu. Lakukan kompilasi pada file tersebut dan amati hasilnya. Kenapa terjadi kegagalan pada saat kompilasi? Benahilah kesalahan tersebut sehingga program dapat berjalan dengan baik.

```
public class Test {
   public static void main(String args) {
      System.out.println("What's wrong with this program?");
 }
```

Latihan 4 : Menganalisa dan membenahi kesalahan pada program

Tulislah program berikut ini dan simpan.

```
public class Test {
   public void main(String args[]) {
      System.out.println("What's wrong with this program?");
 }
```

Lakukan kompilasi pada program tersebut dan jalankan. Kenapa terjadi kesalahan pada saat menjalankan program tersebut? Benahilah kesalahan di atas sehingga program tersebut dapat berjalan dengan baik.

TUGAS

1. Ganti sintaks “println” dengan “print”, perhatikan output yang dihasilkan, analisa hasilnya.
2. Buatlah sebuah program untuk menampilkan string yang berisikan biodata diri masing Nama, nim, alamat, no telp

MODUL 2
Pengenalan Variable, Tipe Data, Operator dan Metode Input di JAVA
(Pertemuan 2)

Tujuan :

Dengan praktikum ini mahasiswa diharapkan dapat:

1. Mengetahui tipe data di Java dan cara menggunakannya
2. Mengetahui cara mendeklarasikan variabel di Java dan cara menggunakannya
3. Mengetahui dan memahami conditional statement di Java serta mampu untuk menggunakannya
4. Mengetahui cara kerja reader input di Java

Tugas Pendahuluan :

1. Apa yang Anda ketahui tentang variable di java dan apa kegunaannya?
2. Apa yang anda ketahui tentang metode input di Java dan kegunaanya?

DASAR TEORI

1. Variabel Java

Variabel merupakan tempat atau wadah untuk menyimpan nilai/value pada bahasa pemrograman. Pada pemrograman Java, semua variabel harus dideklarasikan sebelum mereka dapat digunakan. Bentuk dasar dari sebuah deklarasi variabel yang ditampilkan di sini:

```
type identifier [ = value]
```

Jenis ini merupakan salah satu tipe data Java. Identifier adalah nama variabel. Menyatakan lebih dari satu variabel dari jenis tertentu, menggunakan daftar dipisahkan koma. Berikut adalah beberapa contoh deklarasi variabel dari berbagai jenis. Perhatikan bahwa beberapa mencakup inisialisasi.

```
int a, b, c;          // deklarasi 3 variabel a,b,c bertipe integer
long d = 3, e, f = 5; // deklarasi 3 variabel d,e,f bertipe long
byte z = 22;          // deklarasi dan inisialisasi variable z
char x = 'x';         // variable x yang diberikan value/nilai x
String nama = "adi";  // variable nama yang diberikan value/nilai
```

2. Metode Input Java

Pada pemrograman Java tentunya memiliki metode input yang digunakan sebagai tempat menangkap value atau nilai pada variabel sehingga mampu menjadikan project Anda bersifat dinamis. Pemrograman Java memiliki banyak metode input antara lain sebagai berikut:

- BufferedReader
- JoptionPane
- Scanner

3. Operator Java

Java menyediakan banyak set operator untuk memanipulasi variabel. Kita dapat membagi semua operator Java ke dalam kelompok berikut:

- **Arithmetic Operators**

Operator	Hasil
+	Penjumlahan
-	Pengurangan
*	Perkalian
/	Pembagian
%	Modulus
++	Increment
--	Decrement
+=	Persamaan penjumlahan
-=	Persamaan pengurangan

- **Relational Operators**

Operator	Penggunaan	Keterangan
>	op1 > op2	op1 lebih besar dari op2
>=	op1 >= op2	op1 lebih besar dari atau sama dengan op2
<	op1 < op2	op1 kurang dari op2
<=	op1 <= op2	op1 kurang dari atau sama dengan op2
==	op1 == op2	op1 sama dengan op2
!=	op1 != op2	op1 tidak sama dengan op2

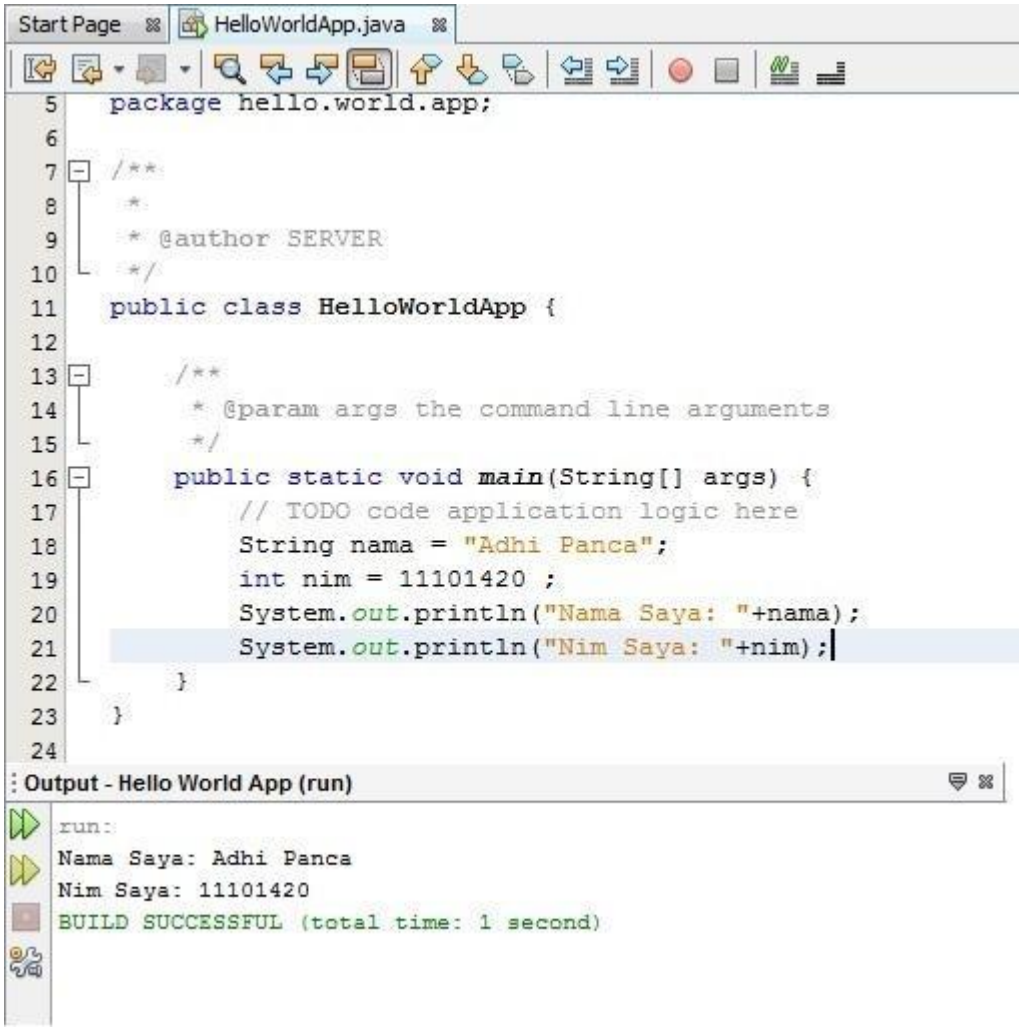
- **Logical Operators**

Operator	Keterangan
&&	Operasi AND
||	Operasi OR
^	Operasi XOR (exclusive OR)
!	Operasi NOT (negasi)

KEGIATAN PRAKTIKUM

1. Variabel

Gambar 2.1 merupakan contoh project dari variabel yang sudah dijelaskan.



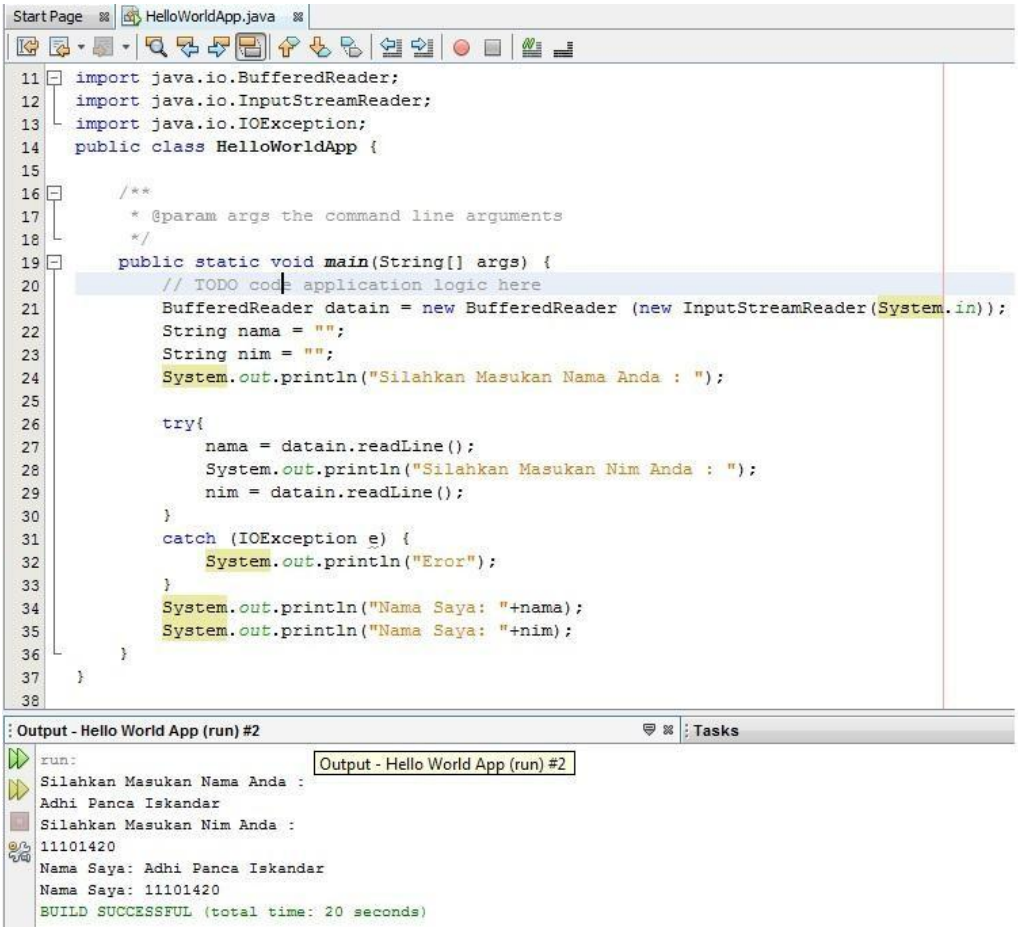
Gambar 2.1 Contoh Penggunaan Variabel

2. Metode Input BufferedReader

Pada bufferedReader memerlukan file bawaan Java dimana kita harus mengimport file–file:

- java.io.BufferedReader;
- java.io.InputStreamReader;
- java.io.IOException;

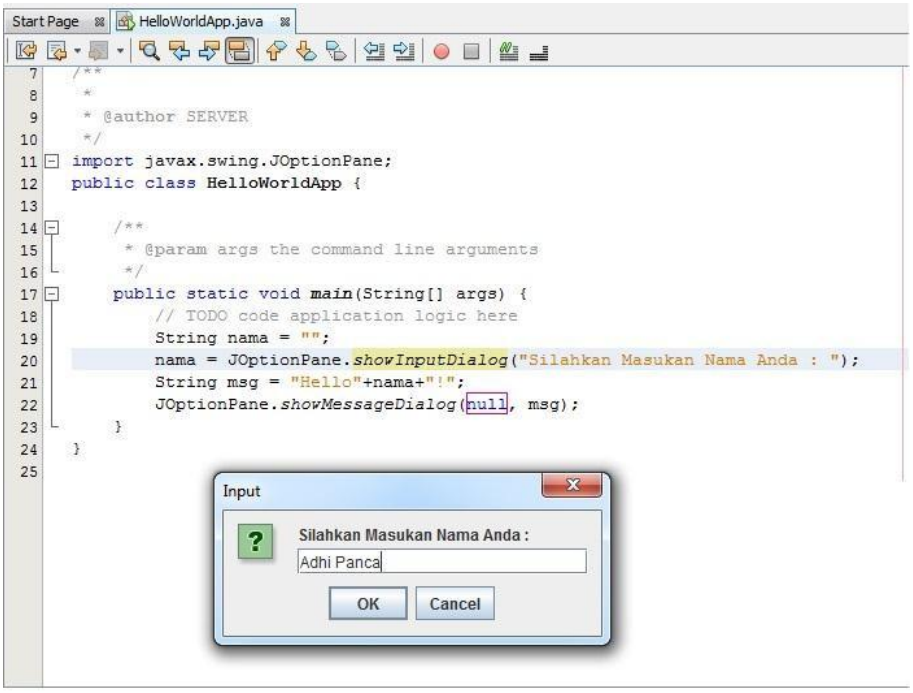
di luar main class atau pada baris 1, 2, dan 3. BufferedReader memiliki variabel dengan tipe Buffer yaitu Try dan Catch yang berfungsi untuk menangkap value/nilai dari variabel utama. **Gambar 2.2** menunjukkan contoh penggunaan dari metode input BufferedReader.



Gambar 2.2 Contoh Penggunaan Metode Input BufferedReader

3. Metode Input JOptionPane

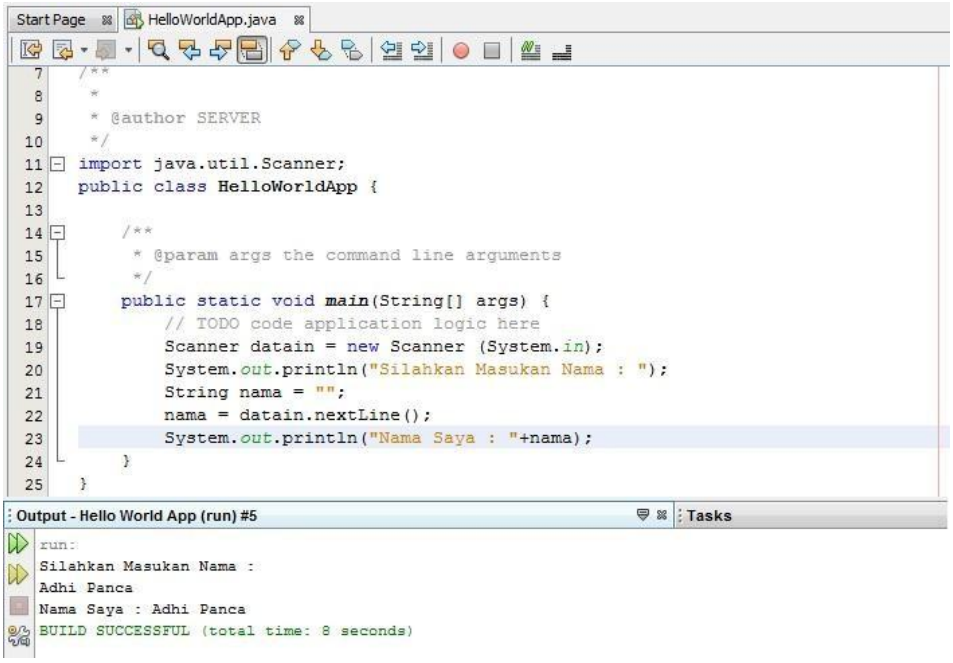
Pada JOptionPane memerlukan file bawaan Java dimana kita harus mengimport file javax.JOptionPane tersebut di luar main class atau pada baris pertama seperti yang ditampilkan pada **Gambar 2.3**.



Gambar 2.3 Contoh Penggunaan Metode Input JOptionPane

4. Metode Input Scanner

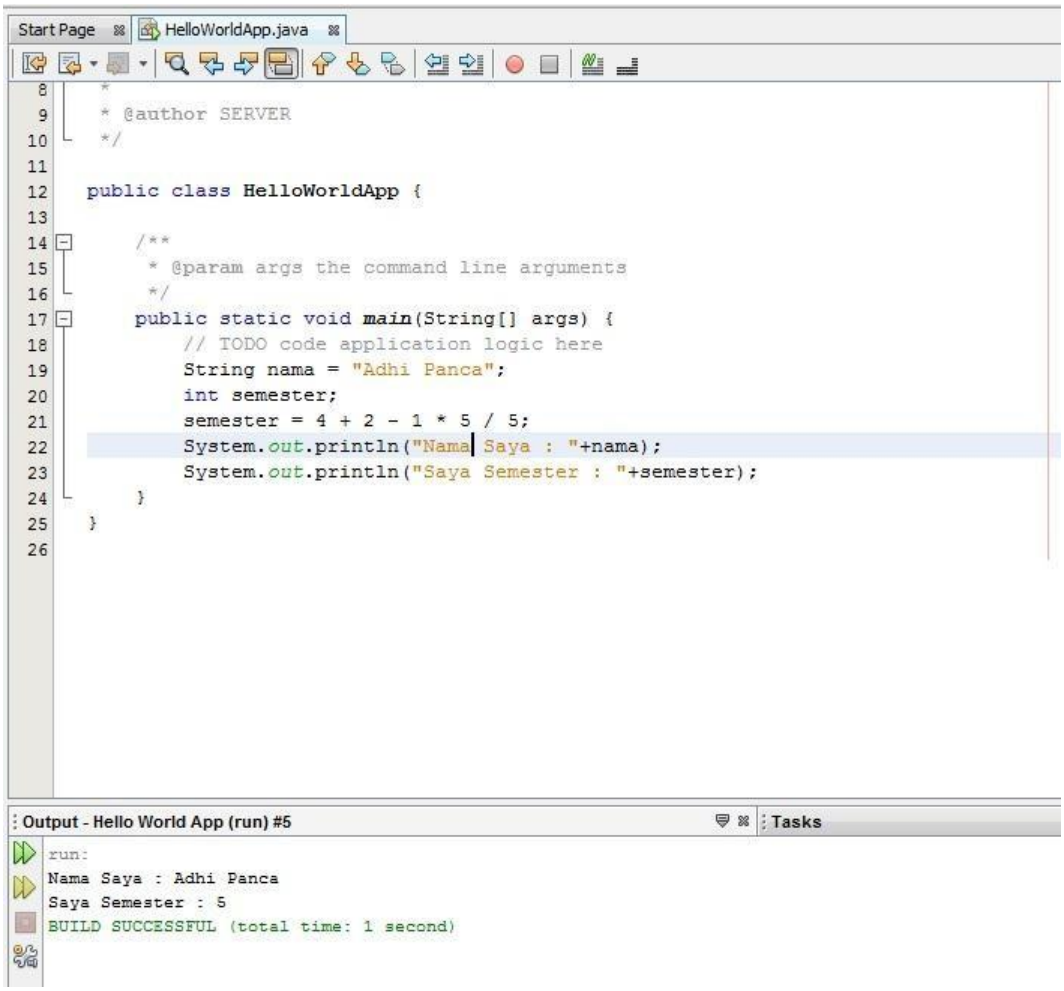
Pada Scanner memerlukan file bawaan Java dimana kita harus mengimport file java.util.Scanner tersebut di luar main class atau pada baris pertama seperti yang ditampilkan pada **Gambar 2.4**.



Gambar 2.4 Contoh Penggunaan Metode Input Scanner

5. Operator Pada Java

Pada contoh operator Java kita gunakan operator aritmatika, karena operator relational dan operator logika biasa digunakan pada sebuah kondisi atau perulangan, maka untuk contoh operator tersebut akan dicantumkan pada penjelasan percabangan dan perulangan.



Gambar 2.5 Contoh Penggunaan Metode Input Scanner

Percobaan 1 : Melakukan increment dan decrement nilai

```
class IncDec {
     public static void main (String args[]) {
          int x = 8, y = 13;
          System.out.println("x = " + x);
          System.out.println("y = " + y);
          System.out.println("x = " + ++x);
          System.out.println("y = " + y++);
          System.out.println("x = " + x--);
```

```
            System.out.println("y = " + --y);
      }
}
```

Percobaan 2 : Melakukan operasi bit

```
class Bitwise {
      public static void main (String args[]) {
            int x = 5, y = 6;
            System.out.println("x = " + x);
            System.out.println("y = " + y);
            System.out.println("x & y = " + (x & y));
            System.out.println("x | y = " + (x | y));
            System.out.println("x ^ y = " + (x ^ y));
      }
}
```

Percobaan 3 : Melakukan operasi komplemen

```
class BitwiseComplement {
      public static void main (String args[]) {
            int x = 8;
            System.out.println("x = " + x);
            int y = ~x;
            System.out.println("y = " + y);
      }
}
```

Percobaan 4 : Melakukan operasi shift

```
class Shift {
      public static void main (String args[]) {
            int x = 7;
            System.out.println("x = " + x);
            System.out.println("x >> 2 = " + (x >> 2));
            System.out.println("x << 1 = " + (x << 1));
            System.out.println("x >>> 1 = " + (x >>> 1));
      }
}
```

Percobaan 5 : Menggunakan logical operator

```
class LogicalOperator {
      public static void main (String args[]) {
            int x = 7, y = 11, z = 11;
            System.out.println("x = " + x);
            System.out.println("y = " + y);
            System.out.println("z = " + z);
            System.out.println("x < y = " + (x < y));
            System.out.println("x > z = " + (x > z));
            System.out.println("y <= z = " + (y <= z));
            System.out.println("x >= y = " + (x >= y));
```

```
            System.out.println("y == z = " + (y == z));
            System.out.println("x != y = " + (x != z));
      }
}
```

Percobaan 6 : Menggunakan operator boolean AND

```
public class BooleanAnd {
    public static void main(String args[]) {
        int a=5, b=7;
        if ((a<2) & (b++<10)) b+=2;
        System.out.println(b);
    }
}
```

Percobaan 7 : Menggunakan operator boolean and short-circuit

```
public class ShortCircuitBooleanAnd {
    public static void main(String args[]) {
        int a=5, b=7;
        if ((a<2) && (b++<10)) b+=2;
        System.out.println(b);
    }
}
```

Percobaan 8 : Menggunakan boolean or

```
public class BooleanOr {
    public static void main(String args[]) {
        int a=5, b=7;
        if ((a>2) | (b++<10)) b+=2;
        System.out.println(b);
    }
}
```

Percobaan 9 : Menggunakan boolean or short-circuit

```
public class ShortCircuitBooleanOr {
    public static void main(String args[]) {
        int a=5, b=7;
        if ((a>2) || (b++<10)) b+=2;
        System.out.println(b);
    }
}
```

Percobaan 10 : Menggunakan operator kondisi

```
class Conditional {
     public static void main (String args[]) {
          int x = 0;
          boolean isEven = false;
          System.out.println("x = " + x);
```

```
            x = isEven ? 4 : 7;
            System.out.println("x = " + x);
      }
}
```

TUGAS

1. Buatlah sebuah program untuk menghitung luas segitiga dengan metode BufferedReader!
2. Buatlah sebuah program untuk menampilkan bilangan genap dan ganjil dengan menggunakan metode Scanner!

MODUL 3
Perulangan, Percabangan dan Array pada Java
(Pertemuan 3)

Tujuan :

Dengan praktikum ini mahasiswa diharapkan dapat:

1. Mahasiswa mengetahui dan memahami perulangan dan percabangan di Java
2. Mahasiswa mengetahui dan memahami array satu dimensi dan multi dimensi
3. Mahasiswa memahami cara mengakses array

Tugas Pendahuluan :

1. Apa yang anda ketahui tentang perulangan dan percabangan di Java?
2. Sebutkan kegunaan dari perulangan dan percabangan di Java!

DASAR TEORI

1. Percabangan

1.1 Percabangan If

Pernyataan if merupakan salah satu bentuk pernyataan yang berguna untuk mengambil keputusan terhadap sebuah kemungkinan. Bentuk pernyataan if berupa:

```
if(kondisi){

      // yang akan dijalankan
}
```

1.2 Percabangan If-Else

Percabangan if−else merupakan percabangan yang sama dengan percabangan if namun memiliki kondisi false, artinya jika kondisi pada if tidak terpenuhi maka perintah pada else akan dijalankan. Bentuk pernyataan if−else berupa:

```
if(kondisi){
      // jalankan jika kondisi true
}else{
      // jalankan jika kondisi false
}
```

1.3 Percabangan If Bersarang

Percabangan if bersarang merupakan gabungan beberapa if dan dapat pula digabung dengan if−else. Bentuk pernyataan if bersarang adalah sebagai berikut:

```
if(kondisi1){
      // perintah kondisi1
}else if(kondisi2){
      // perintah kondisi2
}else if(kondisi3){
      // perintah kondisi3
}else{
      // perintah jika xemua kondixi tidak ada yang
benar}
```

1.3.1 Percabangan Switch-Case

Percabangan switch−case merupakan percabangan yang kondisinya hanya dapat menggunakan perbandingan == (sama dengan). Bentuk pernyataan percabangan switch−case adalah sebagai berikut:

```
switch(variabel){ case nilai1:
      // jalankan instruksi  break; // hentikan
case nilai2:
      // jalankan instruksi  break; // hentikan
case nilai2:
      // jalankan instruksi  break; // hentikan
case nilai4:
      // jalankan instruksi  break; // hentikan
default:
      // jalankan instruksi  break; // hentikan}
```

Pada percabangan switch pertama, diperlukan sebuah variabel, setelah itu pada bagian case dibandingkan, jika sama, maka instruksi akan dijalankan sampai menemui tanda break.

1.4 Perulangan

1.4.1 Perulangan While

Pernyataan while berguna untuk melakukan proses perulangan untuk sebuah kondisi, selama kondisi tersebut bernilai benar (true), maka perulangan akan terus berjalan, dan terhenti ketika kondisi bernilai salah (false). Bentuk pernyataan while seperti berikut ini :

```
while(kondisi){
  // isi instruksi
}
```

1.4.2 Perulangan Do-While

Perulangan do−while merupakan perulangan yang hampir mirip dengan perulangan while namun perbedaannya, pada perulangan do−while, maka minimal instruksi akan dijalankan sekali. Bentuk pernyataan do−while sebagai berikut :

```
do{
// instruksi
} while(kondisi);
```

1.4.3 Perulangan For

Perulangan for merupakan perulangan yang memiliki variabel untuk melakukan pengkondisian, berbeda dengan while dan do−while yang kita harus membuat sebuah variabel diluar untuk melakukan penkondisian, pada perulangan for, ditempatkan sebuah blok untuk membuat variabel dan melakukan proses pengkondisian. Bentuk pernyataan for seperti berikut :

```
for(inisialisasi;kondisi;penaikan/penurunan){
  instruksi
}
```

1.4.4 Perintah Break

Perintah break merupakan perintah yang dapat digunakan untuk menghentikan proses perulangan.

1.4.5 Perintah Continue

Perintah continue dapat digunakan untuk meloncati sebuah perulangan, maksudnya adalah instruksi yang seharusnya dapat dilewat, hal ini berarti instrukti tidak akan dijalankan.

1.5 Array

Suatu array adalah sebuah struktur data yang terdiri atas banyak variabel dengan tipe data sama,

dimana masing-masing elemen variabel mempunyai nilai indeks. Setiap elemen array mampu untuk menyimpan satu jenis data (variabel).

Array merupakan sekumpulan obyek yang memiliki tipe data yang sama dan dapat di akses secara random dengan menggunakan index. Array mempunyai panjang yang tetap, artinya ketika kita mendeklarasikan suatu array dengan panjang 10, maka array tersebut panjangnya akan tetap 10 walaupun kita hanya memakai 5 elemen.

1.5.1 Deklarasi Array Satu Dimensi

Untuk mendeklarasikan array dapat dilakukan dengan cara berikut ini :

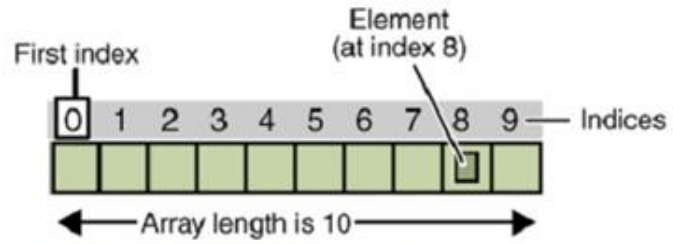
```
int value [];
value = new int [100]; // menginisialisaikan
//[100] merupakan batas ruang array
```

atau bisa deklarasi sekaligus menginisialisasikan

```
Int value []=new int [100];
```

Cara pendeklarasian tanpa batas

```
int value []={1,2,3}; //{1,2,3} adalah isi dari array
```



Gambar 3.1 Ilustrasi Array Satu Dimensi

Untuk mengetahui panjang array yaitu gunakan syntax **nama array.length** berdasarkan contoh diatas nama array = value maka untuk mengetahui panjang value kita gunakan syntax **value.length.** cara pemanggilan yaitu value[0] akan menampilkan angka 1.

1.5.2 Array Multidimensi

Array tidak hanya terdiri dari 1 dimensi, array juga bisa terdiri dari 2 dimensi, 3 dimensi dan n-dimensi. Array di atas 3 dimensi sangat jarang digunakan karena sangat sulit untuk digambarkan. Array multidimensi diimplementasikan sebagai array dalam array. Untuk lebih jelas mengenai cara pendeklarasian array multidimensi dan penggunaanya, silakan ketikkan program di bawah ini.

```
Array 2 dimensi:
char multiChar[ ][ ]=new char[10][5];
char multiChar2[ ][ ]={{'a','b','c'},{'d','e','f'}}; inisialisasi array 2 dimensi
Array 3 dimensi:
int tigaDimensi[ ][ ][ ]=new int[10][10][10];
int tigaDimensi2[ ][ ][ ]={{{1,2,3},{4,5,6}},{{1,2,3},{4,5,6}}}; inisialisasi array 3 dimensi
```

Tambahan:

Array lebih dari 2-dimensi sangat jarang dipakai.

KEGIATAN PRAKTIKUM

1. Simple Project If

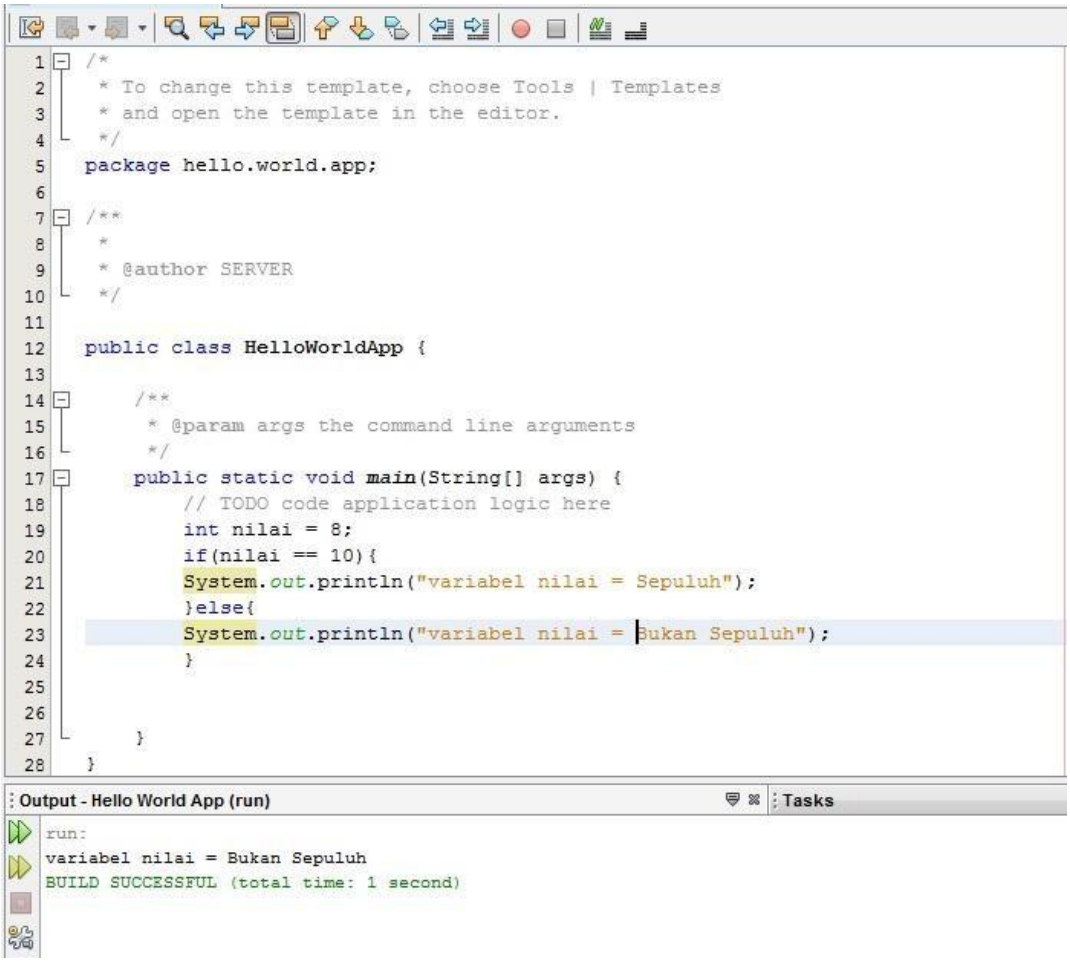
```
1  /*
2   * To change this template, choose Tools | Templates
3   * and open the template in the editor.
4   */
5  package hello.world.app;
6
7  /**
8   *
9   * @author SERVER
10  */
11
12 public class HelloWorldApp {
13
14     /**
15      * @param args the command line arguments
16      */
17     public static void main(String[] args) {
18         // TODO code application logic here
19         int nilai = 10;
20         if(nilai == 10){
21             System.out.println("isi variabel nilai = Sepuluh");
22         }
23
24     }
25 }
26
```

Gambar 3.2 Contoh Penggunaan If

Gambar 3.3 Output Penggunaan If

Jika program diatas dijalankan, maka hasilnya adalah tulisan "isi variabel nilai = Sepuluh" seperti pada **Gambar 3.3** karena kondisi pada if bernilai true, jika kondisi bernilai salah, misal nilai == 100, maka program tidak akan menghasilkan tulisan apa-apa.

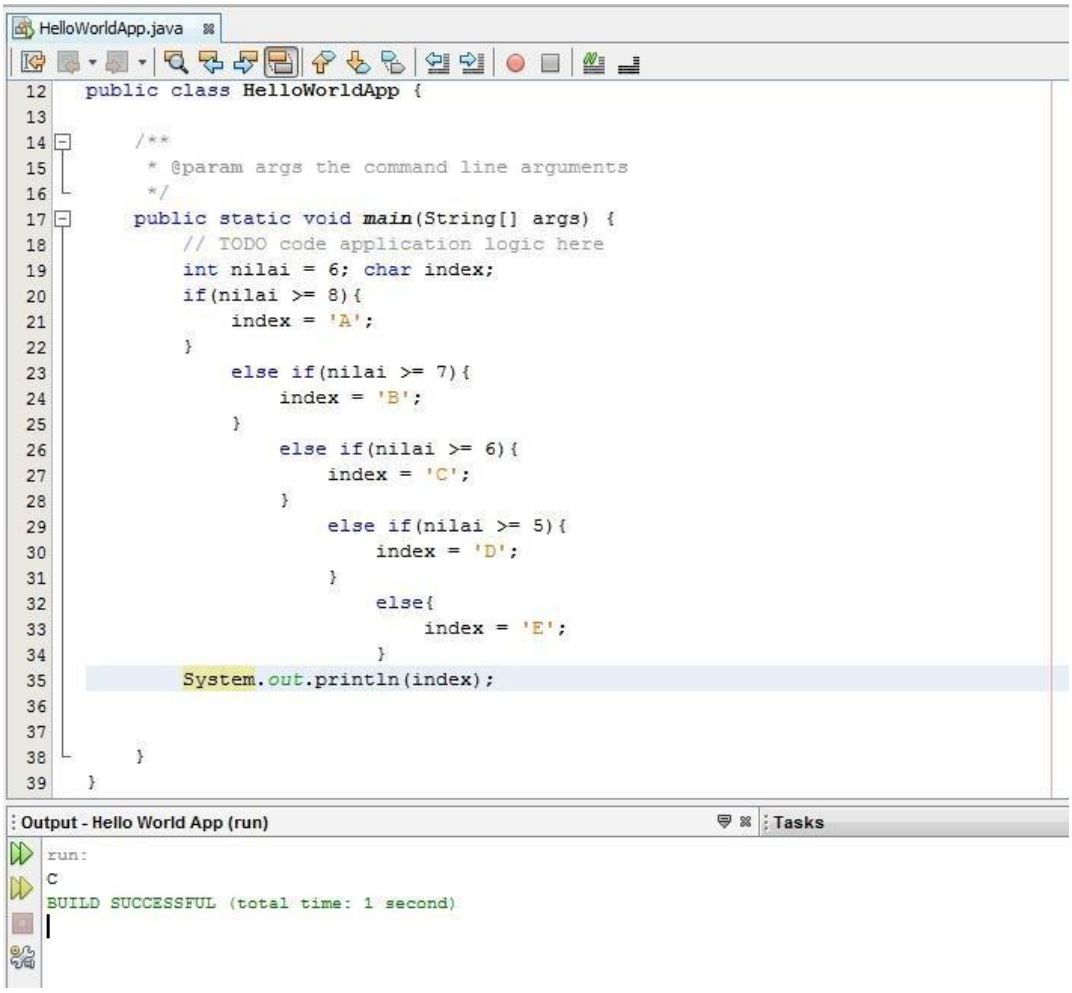
2. Simple Project If-Else



Gambar 3.4 Contoh Penggunaan If-Else

Jika program diatas dijalankan, maka hasilnya adalah tulisan "variabel nilai = Bukan Sepuluh" seperti yang tertera pada **Gambar 3.4**, hal ini dikarenakan variabel nilai bernilai 8, bukan 10.

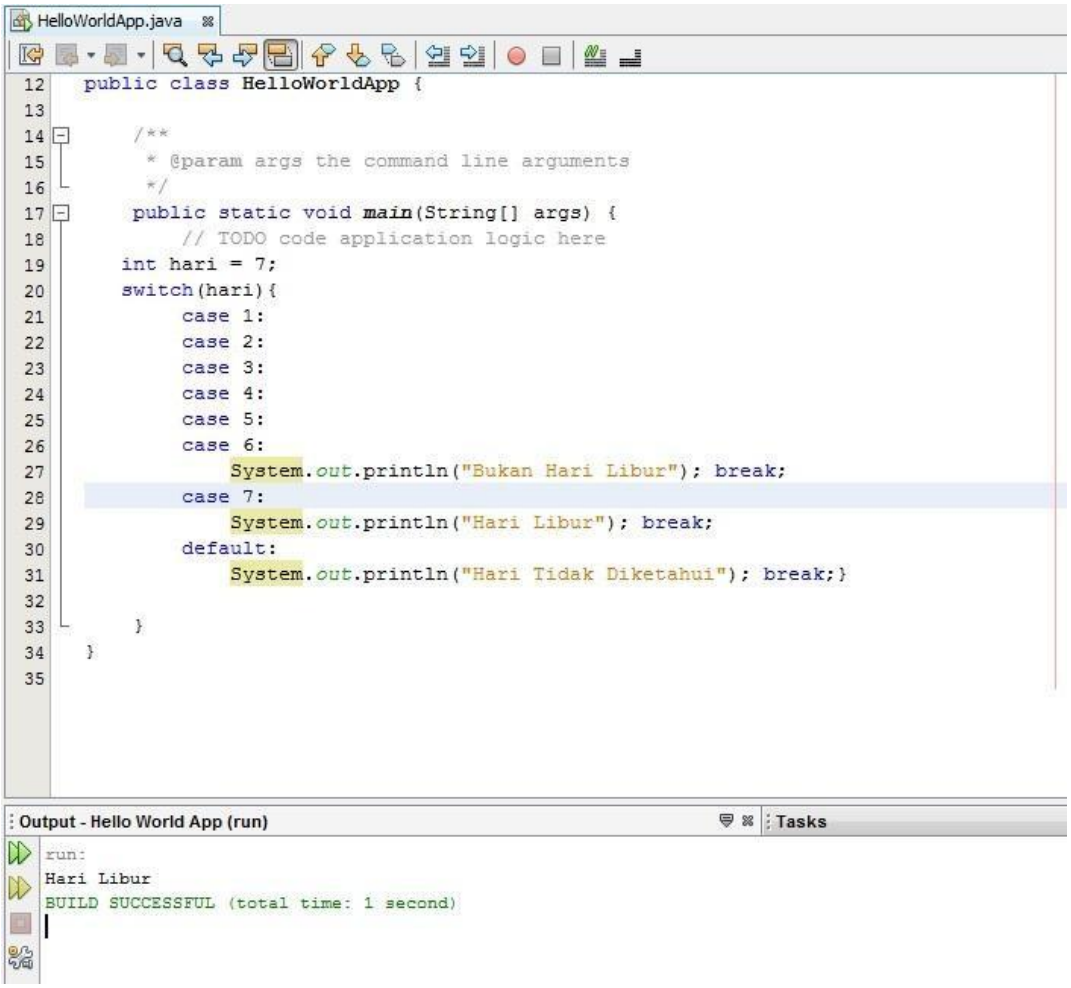
3. Simple Project If Else If



Gambar 3.5 Contoh Penggunaan If Else If

Jika program diatas dijalankan, maka hasilnya adalah “C” seperti yang tertera pada **Gambar 3.5**, hal ini dikarenakan isi variabel nilai berada pada kondisi lebih dari atau sama dengan 6 (nilai >= 6).

4. Simple Project Switch-Case



Gambar 3.6 Contoh Penggunaan Switch-Case

Jika program diatas dijalankan, maka program akan menghasilkan tulisan “Hari Libur” seperti yang tertera pada **Gambar 3.6**, hal ini dikarenakan isi variabel hari berada pada kondisi case 7.

5. Simple Project While

```
 2    * To change this template, choose Tools | Templates
 3    * and open the template in the editor.
 4    */
 5   package hello.world.app;
 6
 7   /**
 8    *
 9    * @author SERVER
10    */
11
12   public class HelloWorldApp {
13
14       /**
15        * @param args the command line arguments
16        */
17       public static void main(String[] args) {
18           // TODO code application logic here
19       int jumlah = 1;
20       while(jumlah <= 10){
21           System.out.println(jumlah); jumlah++;//jumlah ++ berfungsi menaikkan jumlah
22       }
23
24       }
25   }
26
```

```
Output - Hello World App (run)
run:
1
2
3
4
5
6
7
8
9
10
BUILD SUCCESSFUL (total time: 1 second)
```

Gambar 3.7 Contoh Penggunaan While

Jika program tersebut dijalankan, maka hasilnya adalah tulisan dari 1 sampai dengan 10 seperti yang tertera pada **Gambar 3.7**. Saat jumlah bernilai 11, maka perulangan akan terhenti dikarenakan kondisi bernilai false (11 <= 10).

6. Simple Project Do-While

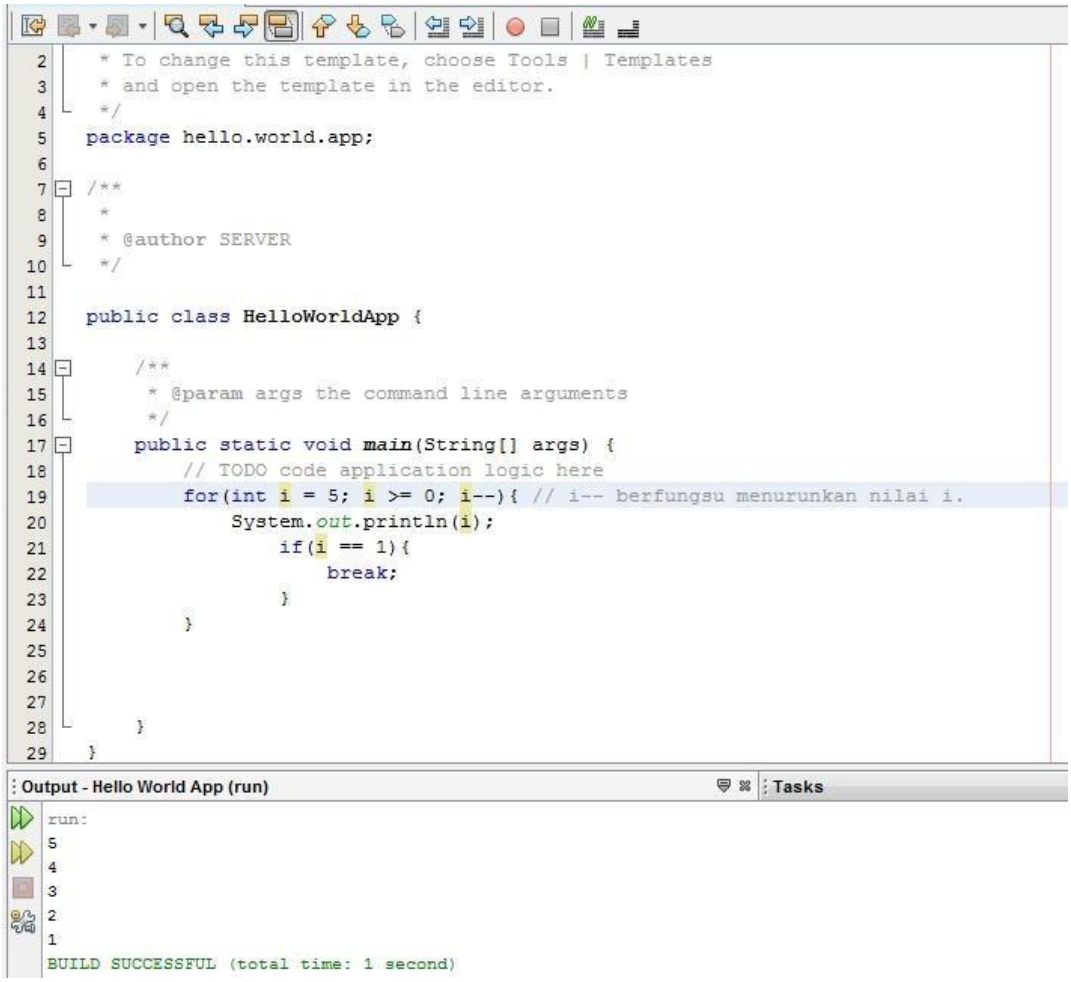
```
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package hello.world.app;

/**
 *
 * @author SERVER
 */

public class HelloWorldApp {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
    int jumlah = 100;
    do{
    System.out.println(jumlah); jumlah++; // naikkan jumlah

    }while(jumlah <= 10);


    }
}
```

```
Output - Hello World App (run)
run:
100
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 3.8 Contoh Penggunaan Do-While

Jika program tersebut dijalankan, maka akan menghasilkan keluaran 100, artinya walaupun kondisi salah, namun minimal isi instruksi akan dijalankan sekali, hal ini dikarenakan proses do−while berbeda dengan while, dimana do−while pertama melakukan instruksi baru mengecek kondisi, sedangkan while pertama mengecek kondisi baru melakukan instruksi.

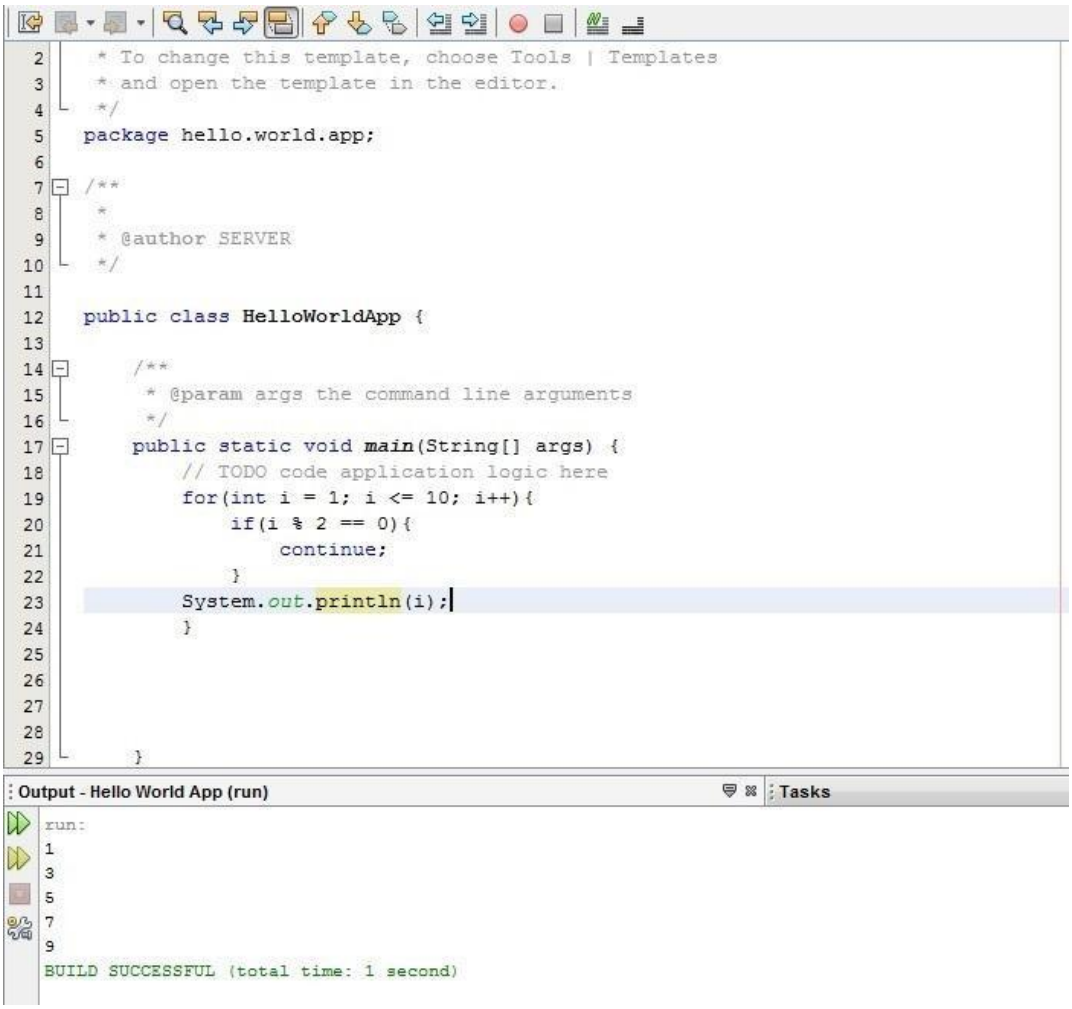
7. Simple Project Break



Gambar 3.9 Contoh Pengunaan Break

Jika program tersebut dijalankan, maka program hanya akan menampilkan angka dari 5 sampai 1 seperti yang ditampilkan pada **Gambar 3.9**, karena pada saat i mencapai 1, program dihentikan oleh perintah break.

8. Simple Project Continue



```
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package hello.world.app;

/**
 *
 * @author SERVER
 */

public class HelloWorldApp {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        for(int i = 1; i <= 10; i++){
            if(i % 2 == 0){
                continue;
            }
        System.out.println(i);
        }


    }
```

```
Output - Hello World App (run)
run:
1
3
5
7
9
BUILD SUCCESSFUL (total time: 1 second)
```

Gambar 3.10 Contoh Penggunaan Project

Jika program diatas dijalankan, maka hasilnya akan menampilkan angka-angka ganjil saja seperti yang ditampilkan pada **Gambar 3.10**, hal ini dikarenakan saat nilai i merupakan angka genap, maka perintah continue membuat program tidak menampilkan angka genap.

.

9. Simple Project Array 1 Dimensi

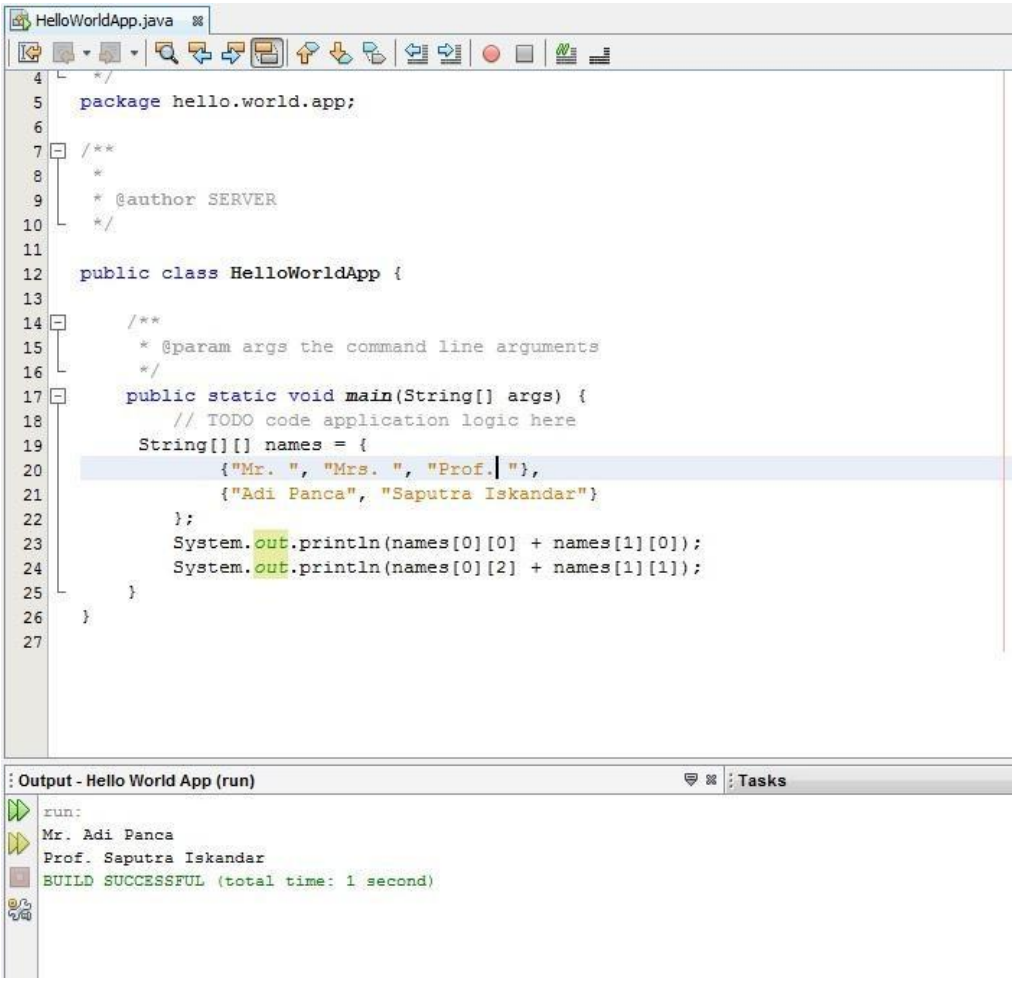
```
5  package hello.world.app;
6
7  /**
8   *
9   * @author SERVER
10  */
11
12 public class HelloWorldApp {
13
14     /**
15      * @param args the command line arguments
16      */
17     public static void main(String[] args) {
18         // TODO code application logic here
19        int[]x=new int[3];
20
21         x[0]= 1;
22         x[1]= 2;
23         x[2]= 3;
24
25         System.out.println(x[0]);
26         System.out.println(x[1]);
27         System.out.println(x[2]);
28
29
30
31
32     }
```

```
Output - Hello World App (run)        Tasks
run:
1
2
3
BUILD SUCCESSFUL (total time: 1 second)
```

Gambar 3.10 Contoh Penggunaan Array 1 Dimensi

Jika program diatas dijalankan, maka hasilnya akan menampilkan angka 1, 2, 3 seperti yang ditampilkan pada **Gambar 3.11**. Hal ini dikarenakan array x telah dideklarasikan untuk menampung nilai integer. Masing-masing indeks pada array x diinisialisasi dengan nilai 1, 2, dan 3.

10. Simple Project Array 2 Dimensi / Multidimensi



Gambar 3.12 Contoh Penggunaan Array 2 Dimensi/Multidimensi

Jika program diatas dijalankan, maka hasilnya akan menampilkan tulisan Mr. Adi Panca | Prof. Saputra Iskandar seperti yang ditampilkan pada **Gambar 3.12**. Hal ini dikarenakan elemen array names yang dipanggil pada pemanggilan pertama adalah pada indeks [0] [0] = **Mr** dan [1][0] = **Adi Panca**. Pada pemanggilan kedua, elemen array names yang dipanggil adalah pada indeks [0][2] = **Prof.** dan [1][1] = **Saputra Iskandar**.

Percobaan 1 : Percabangan menggunakan if, if-else dan else-if

```
  class IfElseName {
     public static void main (String args[]) {
          char firstInitial = 'a';

          if (firstInitial == 'a')
               System.out.println("Nama anda pasti Asep!");
          else if (firstInitial == 'b')
               System.out.println("Nama anda pasti Brodin!");
          else if (firstInitial == 'c')
               System.out.println("Nama anda pasti Cecep!");
          else
               System.out.println("Nama anda tidak terkenal!");
     }
  }
```

Percobaan 2 : Percabangan menggunakan switch

```
class SwitchName {
     public static void main (String args[]) {
          char firstInitial = 'a';

          switch (firstInitial) {
              case 'a':
                   System.out.println("Nama anda pasti Asep!");
              case 'b':
                   System.out.println("Nama anda pasti Brodin!");
              case 'c':
                   System.out.println("Nama anda pasti Cecep!");
              default:
                   System.out.println("Nama anda tidak terkenal!");
          }
     }
}
```

Percobaan 3 : Percabangan menggunakan switch dengan break

```
class ForCount {
     public static void main (String args[]) {
          int count=1;
          for (int i=0; i<9; i++) {
            for (int j=0; j<i+1; j++) {
                System.out.print(count);
            }
            count++;
            System.out.println();
        }
```

Percobaan 4 : Perulangan menggunakan for

```
class SwitchNameBreak {
     public static void main (String args[]) {
          char firstInitial = 'a';


          switch (firstInitial) {
              case 'a':
                    System.out.println("Nama anda pasti Asep!");
                       break;
              case 'b':
                    System.out.println("Nama anda pasti Brodin!");
                        break;
              case 'c':
                    System.out.println("Nama anda pasti Cecep!");
                        break;
              default:
                    System.out.println("Nama anda tidak terkenal!");
          }
     }
}
```

Percobaan 5 : Perulangan menggunakan while

```
class WhileCount {
     public static void main (String args[]) {
        int count=1;
        int i=0;
        while (i<9) {
            int j=0;
            while (j<i+1) {
                System.out.print(count);
                j++;
            }
            count++;
            System.out.println();
            i++;
        }
     }
}
```

Percobaan 6 : Perulangan dengan break

```
class BreakLoop {
     public static void main (String args[]) {
        int i = 0;
        do {
            System.out.println("Iterasi ke " + i);
            i++;
            if (i > 10) break;
        }
        while (true);
     }
}
```

Percobaan 7 : Perulangan dengan continue

```
public class ContinueLoop {
    public static void main(String args[]) {
        int a, b;
        for(a=0;a<2;a++)
            for(b=0;b<3;b++) {
               if (b==1) continue;
               System.out.println("a=" + a + " ; b=" + b);
            }
    }
}
```

Percobaan 8 : Pemakaian label pada kondisi break

```
public class BreakLabel {
    public static void main(String args[]) {
        int a, b;
        Mulai: for(a=0;a<2;a++)
            for(b=0;b<3;b++) {
               if (b==1) break Mulai;
               System.out.println("a=" + a + " ; b=" + b);
            }
     }
}
```

Percobaan 9 : Pemakaian label pada kondisi continue

```
public class ContinueLabel {
    public static void main(String args[]) {
        int a, b;
        Mulai:
        for(a=0;a<2;a++)
            for(b=0;b<3;b++) {
               if (b==1) continue Mulai;
               System.out.println("a=" + a + " ; b=" + b);
            }
    }
}
```

TUGAS

1. Buatlah program mencari nilai rata rata,nilai terbesar dimana nilai yg diinput oleh user, minimal inputan 10x.

2. Buatlah program mencari bilangan berpangkat, dengan ketentuan user menginput bilangan bulat, kemudian user menginput bilangan pangkat, maka akan muncul hasil dari kedua bilangan tersebut.

3. Buatlah program untuk menentukan suatu tahun kabisat atau bukan dimana tahun dibatasi mulai dari tahun 1900 sampai dengan tahun 2020

   Contoh tampilan:

   Masukkan tahun (1900-2005) :
   1923 1923 bukan tahun kabisat

   Masukkan tahun (1900-2005) :
   1898 Maaf, tahun input dibawah
   1900

   Masukkan tahun (1900-2005) :
   1996 1996 adalah tahun kabisat

   Masukkan tahun (1900-2005) :
   2008 Maaf, tahun input diatas
   2005

MODUL 4
Class, Object, Elemen dalam Class pada Java
(Pertemuan 4)

Tujuan :

Dengan praktikum ini mahasiswa diharapkan dapat:

1. Mahasiswa mampu memahami dan mempraktikkan konsep Class dan Objek di Java
2. Mahasiswa mengetahui elemen Class di Java

Tugas Pendahuluan :

1. Apa yang anda ketahui tentang Class dan objek di Java?
2. Apa keunggulan penggunaan konsep Class dan objek dibandingkan konsep pemrograman terstruktural

DASAR TEORI

1. Deklarasi Class, Atribut, dan Metode

- Deklarasi class dapat dilakukan dengan sintaks sebagai berikut:

```
<modifier> class <nama_class> {
   [deklarasi_atribut]
   [deklarasi_konstruktor]
   [deklarasi_metode]
}
```

Contoh:

```
public class Siswa {
   …
}
```

- Deklarasi atribut dapat dilakukan dengan sintaks sebagai berikut:

```
<modifier> <tipe> <nama_atribut> ;
```

Contoh:

```
public class Siswa {
   public int nrp;
   public String nama;
}
```

- Deklarasi metode dapat dilakukan dengan sintaks sebagai berikut:

```
<modifier> <return_type> <nama_metode> ([daftar_argumen])
{
   [<statement>]
}
```

Contoh:

```
public class Siswa{
   public int nrp;
   public String nama;
   public void info() {
      System.out.println("Ini siswa STIKI");
   }
}
```

- Untuk dapat mengakses anggota-anggota dari suatu obyek, maka harus dibuat instance dari class tersebut terlebih dahulu. Berikut ini adalah contoh pengaksesan anggota-anggota dari class Siswa:

```
public class Siswa {
   public static void main(String args[]) {
       Siswa it=new Siswa();
       it.nrp=5;
       it.nama="Andi";
       it.info();
   }
}
```

2. Metode (Method)

Dalam Java terdapat dua buah metode:

i. Fungsi (Non Void), merupakan metode yang memiliki nilai balik jika metode tersebut dipanggil, cara pembuatan sebuah fungsi adalah dengan cara menentukan nilai baliknya, lalu membuat nama metodenya.

Contoh:

```
class Manusia {String nama;
   //fungsi
   Public String ambilNama() {
   //untuk mengembalikan nilai gunakan kata kunci return
     return nama;
   }
}
```

ii. Prosedur (Void), merupakan metode yang tidak memiliki nilai balik, cara pembuatan prosedur sama dengan fungsi namun bedanya, nilai baliknya menggunakan kata kunci void.

Contoh :

```
// prosedur
Public void hapusNama() {
   nama = "";
}
```

Pada kode diatas, kelas manusia memiliki 2 buah metode yaitu ambilNama() dan hapusNama(). Dimana ambilNama() merupakan sebuah fungsi karena mengembalikan nilai String, sedangkan hapusNama() merupakan prosedur karena tidak mengembalikan nilai.

Saat membuat sebuah fungsi maka untuk mengembalikan nilainya, harus menggunakan kata kunci return, diikuti nilai yang akan dikembalikannya. Untuk mengambil nilai balik dari fungsi dapat dilihat pada contoh berikut:

```
Manusia manusia =  new Manusia();
manusia.nama = "Eko Kurniawan Khannedy";
//mengambil nilai dari fungsi
String nama = manusia.ambilNama();
```

KEGIATAN PRAKTIKUM

1. Simple Project Class, Object dan Method

```
12  public class HelloWorldApp { // ini adalah class
13      int panjang,lebar,tinggi,hasil; //deklarasi variabel dalam class
14
15      public int volume(){ //ini adalah method return / fungsi
16          hasil= panjang * lebar * tinggi;
17          return hasil;
18      } // ini akhir dari method fungsi / return
19
20      public void vol (){ // method Void / prosedur
21          System.out.println ("Volume Method Void  = " + panjang * lebar * tinggi);
22          // tanda plus diatas (+) untuk menghubungkan string dan hasil perhitungan
23      } // ini akhir dari method void
24
25      /**
26       * @param args the command line arguments
27       */
28      public static void main(String[] args) {
29          // TODO code application logic here
30       HelloWorldApp object = new HelloWorldApp () ; //ini merupakan object
31       object.panjang = 10; // untuk mengakses variabel / method dalam class gunakan object
32       object.lebar = 2 ; //disambung dengan titik (.) dan ini merupakan bagian
33       object.tinggi = 4; // penginisialisasi variabel dari class helloworldapp
34       System.out.println("Volume Method Return =  " + object.volume());
35       //diatas merupakan cara mengakses method return
36       object.vol(); // cara mengakses method void
37      }
38  }
```

```
Output - Hello World App (run)
run:
Volume Method Return =  80
Volume Method Void  = 80
BUILD SUCCESSFUL (total time: 1 second)
```

Gambar 4.1 Contoh Penggunaan Class, Object, dan Method

Jika program di atas dijalankan, maka hasilnya akan menampilkan "Volume Method Return = 80" dan "Volume Method Void = 80" seperti yang ditampilkan pada **Gambar 4.1**. Hal ini dikarenakan pada baris ke-34 terjadi pemanggilan method volume() dan pada baris ke-36 terjadi pemanggilan method vol().

Percobaan 1 : Mengakses anggota suatu class

Amati program dibawah ini:

```
public class Siswa {
    int nrp;
    public void setNrp(int i){
       nrp=i;
    }
}
public class Test {
    public static void main(String args[]) {
       Siswa anak=new Siswa();
       anak.setNrp(5);
       System.out.println(anak.nrp);
    }
}
```

Percobaan 2 : Mengakses anggota suatu class

Amati program dibawah ini:

```
public class Siswa {
    int nrp;
    String nama;
    public void setNrp(int i) {
        nrp=i;
    }
    public void setNama(String i) {
        nama=i;
    }
}
```

Percobaan 3 : Mengimplementasikan UML class diagram dalam program

Berikut adalah sebuah UML class diagram dari suatu kasus:

Siswa
+ nrp : int
+ Siswa(i : int) + setNrp(i : int) + getNrp() : int

Dari class diagram tersebut, dapat diimplementasikan ke dalam program sebagai berikut:

```
public class Siswa {
    public int nrp;

    public Siswa(int i) {
        nrp=i;
    }

    public void setNrp(int i) {
         nrp=i;
    }
    public int getNrp() {
         return nrp;
     }
}
```

Latihan

Latihan 1 : Mengimplementasikan UML class diagram dalam program untuk class Tabungan

Tabungan
+ saldo : int
+ Tabungan(initsaldo : int) + am bilUang(jum lah : int)

Transformasikan class diagram diatas ke dalam bentuk program!. Tulislah listing program berikut ini sebagai pengetesan.

```
public class TesLatihan1 {
    public static void main(String args[]) {
        Tabungan tabungan=new Tabungan(5000);
        System.out.println("Saldo awal : "+tabungan.saldo);
        tabungan.ambilUang(2300);
        System.out.println("Jumlah uang yang diambil : 2300");
        System.out.println("Saldo sekarang : " + tabungan.saldo);
    }
}
```

Lakukan kompilasi pada program diatas dan jalankan. Jika tampilan di layar tampak seperti dibawah ini, maka program anda sudah benar. Jika tidak sama, benahi kembali program anda dan lakukan

hal yang sama seperti diatas.

```
Saldo awal : 5000
Jumlah uang yang diambil : 2300
Saldo sekarang : 2700
```

Latihan 2 : Mengimplementasikan UML class diagram dalam program untuk class Mahasiswa

Mahasiswa
- nrp : int - nama : String
+ Mahasiswa(i : int, String n) + getNrp() : int + getNama(): String

Transformasikan class diagram di atas ke dalam bentuk program! Tulislah listing program berikut ini sebagai pengetesan.

```
public class TesLatihan2 {
    public static void main(String args[]) {
        Mahasiswa mhs=new Mahasiswa(12345, "Jono");
        System.out.println("NRP : " + mhs.getNrp());
        System.out.println("Nama : " + mhs.getNama());
    }
}
```

Lakukan kompilasi pada program diatas dan jalankan. Jika tampilan di layar tampak seperti dibawah ini, maka program anda sudah benar. Jika tidak sama, benahi kembali program anda dan lakukan hal yang sama seperti di atas.

```
NRP : 12345
Nama : Jono
```

Latihan 3 : Mengimplementasikan UML class diagram dalam program untuk class Truk

Truk
- muatan : double - muatanmaks : double
+ Truk(beratmaks : double) + getMuatan() : double + getMuatanMaks() : double + tambahMuatan(berat : double)

Transformasikan class diagram diatas ke dalam bentuk program! Tulislah listing program berikut ini sebagai pengujian.

```
public class TesLatihan3 {
    public static void main(String args[]) {
        Truk truk=new Truk(1000);
        System.out.println("Muatan maksimal : " +
        truk.getMuatanMaks());
        truk.tambahMuatan(500.0);
        System.out.println("Tambah muatan : 500");
        truk.tambahMuatan(350.0);
        System.out.println("Tambah muatan : 350");
        truk.tambahMuatan(100.0);
        System.out.println("Tambah muatan : 100");
        truk.tambahMuatan(150.0);
        System.out.println("Tambah muatan : 150");
        System.out.println("Muatan sekarang = " + truk.getMuatan());
    }
}
```

Lakukan kompilasi pada program diatas dan jalankan. Jika tampilan di layar tampak seperti dibawah ini, maka program anda sudah benar. Jika tidak sama, benahi kembali program anda dan lakukan hal yang sama seperti di atas.

```
Muatan maksimal : 1000.0
Tambah muatan : 500
Tambah muatan : 350
Tambah muatan : 100
Tambah muatan : 150
Muatan sekarang = 950.0
```

Tugas

Tugas 1 : Mengimplementasikan UML class diagram dalam program untuk class Tabungan

Tabungan
- saldo : int
+ Tabungan(initsaldo : int) + getSaldo() : int + simpanUang(jumlah : int) + ambilUang(jumlah : int) : boolean

Transformasikan class diagram diatas ke dalam bentuk program?. Tulislah listing program berikut ini untuk melakukan pengujian.

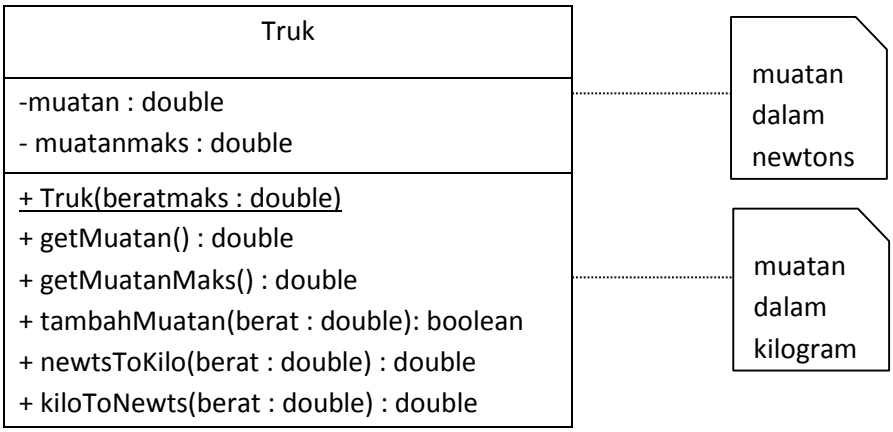
```
public class TesTugas1 {
    public static void main(String args[]) {
        boolean status;
        Tabungan tabungan=new Tabungan(5000);
        System.out.println("Saldo awal : "+tabungan.getSaldo());
        tabungan.simpanUang(3000);
        System.out.println("Jumlah uang yang disimpan : 3000");
        status=tabungan.ambilUang(6000);
        System.out.print("Jumlah uang yang diambil : 6000");
        if (status)
           System.out.println(" ok");
        else
           System.out.println(" gagal");
        tabungan.simpanUang(3500);
        System.out.println("Jumlah uang yang disimpan : 3500");
        status=tabungan.ambilUang(4000);
        System.out.print("Jumlah uang yang diambil : 4000");
```

```
        if (status)
            System.out.println("  ok");
        else
            System.out.println("  gagal");
        status=tabungan.ambilUang(1600);
        System.out.print("Jumlah uang yang diambil : 1600");
        if (status)
            System.out.println("  ok");
        else
            System.out.println("  gagal");
        tabungan.simpanUang(2000);
        System.out.println("Jumlah uang yang disimpan : 2000");
        System.out.println("Saldo sekarang = " + tabungan.getSaldo());
    }
}
```

Lakukan kompilasi pada program diatas dan jalankan. Jika tampilan di layar tampak seperti dibawah ini, maka program Anda sudah benar. Jika tidak sama, benahi kembali program Anda dan lakukan hal yang sama seperti diatas.

```
Saldo awal : 5000
Jumlah uang yang disimpan : 3000
Jumlah uang yang diambil : 6000   ok
Jumlah uang yang disimpan : 3500
Jumlah uang yang diambil : 4000   ok
Jumlah uang yang diambil : 1600   gagal
Jumlah uang yang disimpan : 2000
Saldo sekarang = 3500
```

Tugas 2 : Mengimplementasikan UML class diagram dalam program untuk class Truk




```
Keterangan : 1 kilogram = 9,8 newtons
```

Transformasikan class diagram di atas ke dalam bentuk program! Tulislah listing program berikut ini untuk melakukan pengujian.

```
public class TesTugas2 {
    public static void main(String args[]) {
        Truk truk=new Truk(900);
        boolean status;
        System.out.println("Muatan maksimal : " +
        truk.getMuatanMaks());
        status=truk.tambahMuatan(500.0);
        System.out.print("Tambah muatan : 500");
        if (status)
            System.out.println("  ok");
        else
            System.out.println("  gagal");
        status=truk.tambahMuatan(300.0);
        System.out.print("Tambah muatan : 300");
        if (status)
            System.out.println(" ok");
        else
            System.out.println(" gagal");
        status=truk.tambahMuatan(150.0);
        System.out.print("Tambah muatan : 150");
        if (status)
            System.out.println("  ok");
        else
            System.out.println("  gagal");
        status=truk.tambahMuatan(50.0);
        System.out.print("Tambah muatan : 50");
        if (status)
            System.out.println("  ok");
        else
            System.out.println("  gagal");
        System.out.println("Muatan sekarang = " + truk.getMuatan());
    }
}
```

Lakukan kompilasi pada program diatas dan jalankan. Jika tampilan di layar tampak seperti dibawah ini, maka program Anda sudah benar. Jika tidak sama, benahi kembali program Anda dan lakukan hal yang sama seperti di atas.

```
Muatan maksimal : 900.0
Tambah muatan : 500  ok
Tambah muatan : 300  ok
Tambah muatan : 150  gagal
Tambah muatan : 50  ok
Muatan sekarang = 849.9999999999999
```

MODUL 5
Enkapsulasi, Constructor, Overloading Constructor
(Pertemuan 5)

Tujuan :

Dengan praktikum ini mahasiswa diharapkan dapat:

1. Menerapkan konsep enkapsulasi pada class
2. Mendeklarasikan suatu constructor

Tugas Pendahuluan :

1. Apa yang anda ketahui tentang konsep enkapasulasi di Java?
2. Apa yang adan ketahui tentang konstruktor di Java

DASAR TEORI

- Kita dapat menyembunyikan informasi dari suatu class sehingga anggota-anggota class tersebut tidak dapat diakses dari luar. Adapun caranya adalah cukup dengan memberikan akses kontrol private ketika mendeklarasikan suatu atribut atau method.

  Contoh:

  ```
  private int nrp;
  ```

- Encapsulation (Enkapsulasi) adalah suatu cara untuk menyembunyikan implementasi detail dari suatu class. Enkapsulasi mempunyai dua hal mendasar, yaitu:
  - information hiding
  - menyediakan suatu perantara (method) untuk pengaksesan data

  Contoh:

  ```
  public class Siswa {
     private int nrp;
     public void setNrp(int n) {
        nrp=n;
     }
  }
  ```

- Contructor (konstruktor) adalah suatu method yang pertama kali dijalankan pada saat pembuatan suatu obyek. Konstruktor mempunyai ciri yaitu:
  - mempunyai nama yang sama dengan nama class
  - tidak mempunyai return type (seperti void, int, double, dan lain-lain)

Contoh:

```
public class Siswa {
   private int nrp;
   private String nama;

   public Siswa(int n, String m) {
      nrp=n;
      nama=m;
   }
}
```

- Suatu class dapat mempunyai lebih dari 1 konstruktor dengan syarat daftar parameternya tidak boleh ada yang sama.

  Contoh:

```
public class Siswa {
   private  int  nrp;
   private String nama;
   public Siswa(String m) {
      nrp=0;
      nama="";
   }
   public Siswa(int n, String m) {
      nrp=n;
      nama=m;
   }
}
```

Percobaan 1 : Melakukan enkapsulasi pada suatu class

Mahasiswa
+ nrp : int + nama : String
+ Mahasiswa(nrp : int, nama : String)

Jika enkapsulasi dilakukan pada class diagram di atas, maka akan berubah menjadi:

Mahasiswa
- nrp : int - nama : String
+ Mahasiswa(nrp : int, nama : String) + getNrp() : int + getNama(): String + setNrp(nrp : int) + setNama(nama : String)

Percobaan 2 : Melakukan overloading constructor

Mahasiswa
- nrp : int - nama : String
+ Mahasiswa() + Mahasiswa(nama : String) + Mahasiswa(nrp : int, nama : String)

Dari class diagram tersebut, dapat diimplementasikan ke dalam program sebagai berikut:

```
public class Mahasiswa {
    private int nrp;
    private String nama;

    public Mahasiswa() {
        nrp=0;
        nama="";
    }

    public Mahasiswa(String nama) {
        nrp=0;
        this.nama=nama;
    }

    public Mahasiswa(int nrp, String nama) {
        this.nrp=nrp;
        this.nama=nama;
    }
}
```

Latihan

Mengimplementasikan UML class diagram dalam program untuk class Kalender

Kalender
+ tanggal : int + bulan : int + tahun : int
+ Kalender(tanggal : int) + Kalender(bulan : int, tahun : int) + Kalender(tanggal : int, bulan : int, tahun : int)

Dari class diagram di atas, desainlah suatu class yang memenuhi konsep enkapsulasi. Untuk nilai inisialisasi, dipakai 1-1-2000. Pakailah kata kunci *this* untuk mempersingkat pengkodean. Tulislah listing program berikut ini sebagai pengujian.

```
public class TesLatihan {
    public static String getTime(Kalender kal) {
        String tmp;
         tmp=kal.getTanggal() + "-" + kal.getBulan() + "-" +
             kal.getTahun();
             return tmp;
    }
    public static void main(String args[]) {
        Kalender kal=new Kalender(8);
        System.out.println("Waktu awal : " + getTime(kal));
        kal.setTanggal(9);
        System.out.println("1 hari setelah waktu awal : " + getTime(kal));
        kal=new Kalender(6,2003);
        System.out.println("Waktu berubah : " + getTime(kal));
        kal.setBulan(7);
        System.out.println("1 bulan setelah itu : " + getTime(kal));
        kal=new Kalender(20,10,2004);
        System.out.println("Waktu berubah : " + getTime(kal));
        kal.setTahun(2005);
        System.out.println("1 tahun setelah itu : " + getTime(kal));
    }
}
```

Lakukan kompilasi pada program diatas dan jalankan. Jika tampilan di layar tampak seperti dibawah ini, maka program Anda sudah benar. Jika tidak sama, benahi kembali program Anda.

```
Waktu awal : 8-1-2000
1 hari setelah waktu awal : 9-1-2000
Waktu berubah : 1-6-2003
1 bulan setelah itu : 1-7-2003
Waktu berubah : 20-10-2004
1 tahun setelah itu : 20-10-2005
```

TUGAS

1. Buatlah sebuah program bebas yang menunjukkan penggunaan konsep enkapsulasi!
2. Buatlah sebuat program bebas yang menggunakan konsep overloading konstruktor!

Modul 6
Inheritance, Overloading, Overriding
(Pertemuan 6)

Tujuan

Dengan praktikum ini mahasiswa diharapkan dapat:

1. Memahami konsep Inheretance dan cara penggunaanya
2. Memahami konsep Overloading, overriding dan cara penggunaanya

Tugas Pendahuluan

1. Apa yang Anda ketahui tentang konsep inheritance di Java ?
2. Apa yang Anda ketahui tentang konsep overloading, overriding di Java?

DASAR TEORI

- Konsep inheritance ini mengadopsi dunia riil dimana suatu entitas/obyek dapat mempunyai entitas/obyek turunan. Dengan konsep inheritance, sebuah class dapat mempunyai class turunan. Suatu class yang mempunyai class turunan dinamakan parent class atau base class. Sedangkan class turunan itu sendiri seringkali disebut subclass atau child class. Suatu subclass dapat mewarisi apa-apa yang dipunyai oleh parent class-nya, sehingga member dari suatu subclass adalah terdiri dari apa-apa yang ia punyai dan juga apa-apa yang ia warisi dari class parent-nya. Kesimpulannya, boleh dikatakan bahwa suatu subclass adalah tidak lain hanya memperluas (extend) parent class-nya.
- Di dalam Java untuk mendeklarasikan suatu class sebagai subclass dilakukan dengan cara menambahkan kata kunci extends setelah deklarasi nama class, kemudian diikuti dengan nama parent class-nya. Kata kunci extends tersebut memberitahu kompiler Java bahwa kita ingin melakukan perluasan class. Berikut adalah contoh deklarasi inheritance:

  Contoh:

```
public class B extends A {
...
}
```

  Contoh di atas memberitahukan kompiler Java bahwa kita ingin meng-extend class A ke class B. Dengan kata lain, class B adalah subclass (class turunan) dari class A, sedangkan class A adalah parent class dari class B.
- Java hanya memperkenankan adanya single inheritance. Konsep single inheritance hanya memperbolehkan suatu sublass mempunyai satu parent class. Dengan konsep single inheritance

ini, masalah pewarisan akan dapat diamati dengan mudah.

- Dalam konsep dasar inheritance dikatakan bahwa suatu subclass adalah tidak lain hanya memperluas (extend) parent class-nya.

  Contoh :

```
public class Pegawai {
    public String nama;
    public double gaji;
}
public class Manajer extends Pegawai {
    public String departemen;
}
```

3. Pada saat class Manajer menurunkan atau memperluas (extend) class Pegawai, maka ia mewarisi data member yang dipunyai oleh class Pegawai. Dengan demikian, class Manajer mempunyai data member yang diwarisi oleh Pegawai (nama, gaji), ditambah dengan data member yang ia punyai (departemen).
4. Pengaksesan member yang ada di parent class dari subclass-nya tidak jauh berbeda dengan pengaksesan member subclass itu sendiri.
5. Suatu parent class dapat tidak mewariskan sebagian member-nya kepada subclass-nya. Sejauh mana suatu member dapat diwariskan ke class lain, ataupun suatu member dapat diakses dari class lain, sangat berhubungan dengan access control (kontrol pengaksesan).

   Di dalam Java, kontrol pengaksesan dapat digambarkan dalam tabel berikut ini:

Modifier	**Class yang sama**	**Package yang sama**	**Subclass**	**Class manapun**
private	√			
default	√	√		
protected	√	√	√	
public	√	√	√	√

- Kata kunci *super* dipakai untuk merujuk pada member dari parent class, sebagaimana kata kunci *this* yang dipakai untuk merujuk pada member dari class itu sendiri. Adapun format penulisannya adalah sebagai berikut:

  super.data_member ‰ merujuk pada data member pada parent class

  super.function_member() ‰ merujuk pada function member pada parent class

  super() ‰ merujuk pada konstruktor pada parent class

Contoh:

```
public class Siswa {
      private int nrp;

      public setNrp(int nrp) {
         this.nrp=nrp;
      }
}
```

- Overloading adalah suatu keadaan dimana beberapa method sekaligus dapat mempunyai nama yang sama, akan tetapi mempunyai fungsionalitas yang berbeda. Contoh penggunaan overloading dilihat dibawah ini:

  Gambar(int t1) ‰ 1 parameter titik, untuk menggambar titik

  Gambar(int t1,int t2) ‰ 2 parameter titik, untuk menggambar garis

  Gambar(int t1,int t2,int t3) ‰ 3 parameter titik, untuk menggambar segitiga

  Gambar(int t1,int t2,int t3,int t4) ‰ 4 parameter titik, untuk menggambar persegi empat

  Overloading ini dapat terjadi pada class yang sama atau pada suatu parent class dan subclass-nya. Overloading mempunyai ciri-ciri sebagai berikut:

  1. Nama method harus sama
  2. Daftar parameter harus berbeda
  3. Return type boleh sama, juga boleh berbeda

- Overriding adalah suatu keadaan dimana method pada subclass menolak method pada parent class-nya. Overriding mempunyai ciri-ciri sebagai berikut :

  1. Nama method harus sama
  2. Daftar parameter harus sama
  3. Return type harus sama

  Berikut ini contoh terjadinya overriding dimana method Info() pada class Child meng- override method Info() pada class parent:

```
class Parent {
   public void Info() {
      System.out.println("Ini class Parent");
   }
}
class Child extends Parent {
   public void Info() {
      System.out.println("Ini class Child");
   }
}
```

Method yang terkena override (overridden method) diharuskan tidak boleh mempunyai modifier yang lebih luas aksesnya dari method yang meng-override (overriding method).

Percobaan 1: Melakukan overloading pada method

Tulislah listing program berikut ini dan amati yang terjadi pada saat terjadinya overloading pada method.

```
import java.awt.Point;
public class Segiempat {
     int x1 = 0;
     int y1 = 0;
     int x2 = 0;
     int y2 = 0;

    public void buatSegiempat(int x1, int y1, int x2, int y2) {
          this.x1 = x1;
          this.y1 = y1;
          this.x2 = x2;
          this.y2 = y2;
    }

    public void buatSegiempat(Point topLeft, Point bottomRight) {
          x1 = topLeft.x;
          y1 = topLeft.y;
          x2 = bottomRight.x;
          y2 = bottomRight.y;
     }

     public void buatSegiempat(Point topLeft, int w, int h) {
          x1 = topLeft.x;
          y1 = topLeft.y;
          x2 = (x1 + w);
          y2 = (y1 + h);
    }

    void cetakSegiempat(){
          System.out.print("Segiempat: <" + x1 + ", " + y1);
          System.out.println(", " + x2 + ", " + y2 + ">");
    }

    public static void main(String[] arguments) {
         Segiempat rect = new Segiempat();
         System.out.println("Buat segiempat dengan koordinat (25,25)
             dan (50,50)");
         rect.buatSegiempat(25, 25, 50, 50);
         rect.cetakSegiempat();
```

```
        System.out.println();

        System.out.println("Buat segiempat dengan point (10,10) dan
        point (20,20):");
        rect.buatSegiempat(new Point(10,10), new Point(20,20));
        rect.cetakSegiempat();
        System.out.println();
        System.out.print("Buat segiempat dengan 1 point (10,10),
        koodinat (50,50)");
        rect.buatSegiempat(new Point(10,10), 50, 50);
        rect.cetakSegiempat();
    }
}
```

Ketika program tersebut dijalankan, akan tampak hasil seperti dibawah ini :

```
Buat segiempat dengan koordinat (25,25) dan (50,50)
Segiempat: <25, 25, 50, 50>

Buat segiempat dengan point (10,10) dan point (20,20):
Segiempat: <10, 10, 20, 20>
Buat segiempat dengan 1 point (10,10), koodinat (50,50)Segiempat:
<10, 10, 60, 60>
```

Percobaan 2 : Menggunakan kata kunci super

Berikut ini listing penggunaan kata kunci super.

```
class Parent {
    public int x = 5;
}

class Child extends Parent {
   public int x = 10;
   public void Info(int x) {
       System.out.println("Nilai x sebagai parameter = " + x);
       System.out.println("Data member x di class Child = " + this.x);
       System.out.println("Data member x di class Parent = " + super.x);
   }
}

public class NilaiX {
    public static void main(String args[]) {
        Child tes = new Child(); tes.Info(20);
    }
}
```

Ketika program tersebut dijalankan, akan tampak hasil seperti dibawah ini :

```
Nilai x sebagai parameter = 20
Data member x di class Child = 10
Data member x di class Parent = 5
```

Percobaan 3 : Kontrol pengaksesan

Buatlah class Pegawai seperti dibawah ini:

```
public class Pegawai {
     private String nama;
     public double gaji;
}
```

Kemudian buatlah class Manajer seperti di bawah ini

```
public class Manajer extends Pegawai {
     public String departemen;

     public void IsiData(String n, String d) {
          nama=n;
          departemen=d;
       }
}
```

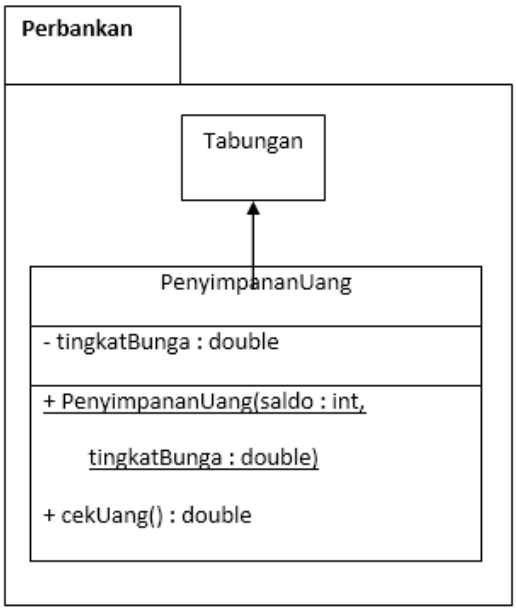
Sekarang cobalah untuk mengkompilasi class Manajer diatas. Apa yang terjadi?. Pesan kesalahan akan muncul seperti ini:

```
Manajer.java:5: nama has private access in Pegawai
                         nama=n;
```

Ini membuktikan bahwa class Manajer tidak mewarisi data member nama dari parent class-nya (Pegawai).

Percobaan 4 : Konstruktor tidak diwariskan

Buatlah class kosong bernama Tabungan seperti di **Gambar 6.1** berikut:



Gambar 6.1 Ilustrasi Class Tabungan

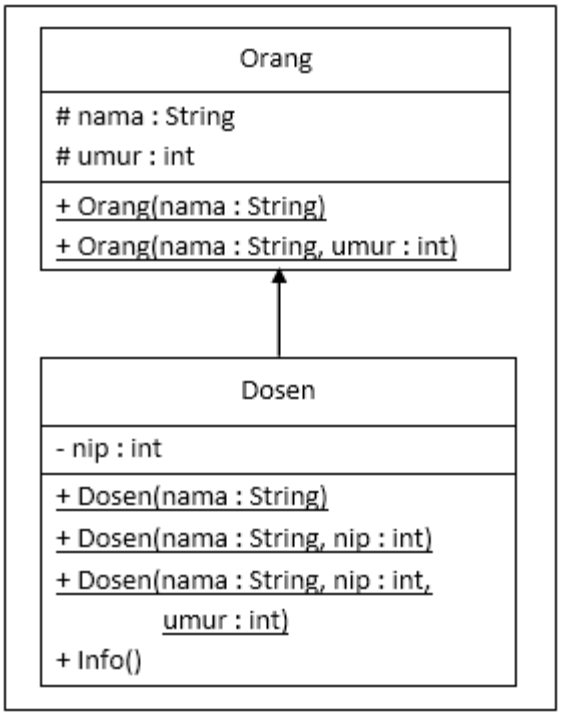
Ubahlah mode akses atribut saldo pada Tabungan menjadi protected. Lalu transformasikan class diagram di atas ke dalam bentuk program! Tulislah listing program berikut ini sebagai pengetesan.

```
import perbankan.*;

public class TesLatihan {
    public static void main(String args[]) {
        PenyimpananUang tabungan=new PenyimpananUang(5000,8.5/100);
        System.out.println("Uang yang ditabung : 5000");
        System.out.println("Tingkat bunga sekarang : 8.5%");
        System.out.println("Total uang anda sekarang : " +
      tabungan.cekUang());

    }
}
```

Lakukan kompilasi pada program diatas dan jalankan. Jika tampilan di layar tampak seperti dibawah ini, maka program anda sudah benar. Jika tidak sama, benahi kembali program anda dan lakukan hal yang sama seperti di atas.

```
Uang yang ditabung : 5000
Tingkat bunga sekarang : 8.5%
Total uang anda sekarang : 5425.0
```

Percobaan 5 : Mengimplementasikan UML class diagram dalam program



Gambar 6.2 Class Diagram

Transformasikan class diagram diatas ke dalam bentuk program!. Tulislah listing program berikut ini sebagai pengetesan.

```
public class TesLatihan {
    public static void main(String args[]) {
       System.out.println("Masukkan identitas dosen 1 : Agus");
       Dosen dosen1=new Dosen("Agus");
       System.out.println("Masukkan identitas dosen 2 : Budi, NIP. 1458");
       Dosen dosen2=new Dosen("Budi", 1458);
       System.out.println("Masukkan identitas dosen 3 : Iwan, NIP. 1215,
                           umur 47");
       Dosen dosen3=new Dosen("Iwan", 1215, 47);
       System.out.println();
       dosen1.Info();
       System.out.println();
       dosen2.Info();
       System.out.println();
       dosen3.Info();
     }
}
```

Lakukan kompilasi pada program diatas dan jalankan. Jika tampilan di layar tampak seperti dibawah ini, maka program anda sudah benar. Jika tidak sama, benahi kembali program anda dan lakukan hal yang sama seperti di atas.

```
Masukkan identitas dosen 1 : Agus
Masukkan identitas dosen 2 : Budi, NIP. 1458
Masukkan identitas dosen 3 : Iwan, NIP. 1215, umur 47

Nama : Agus
NIP : -
Umur : -

Nama : Budi
NIP : 1458
Umur : -

Nama : Iwan
NIP : 1215
Umur : 47
```

Tugas

Mengimplementasikan UML class diagram dalam program untuk package perbankan

Transformasikan class diagram diatas ke dalam bentuk program! Tulislah listing program berikut ini untuk pengujian.

```
import perbankan.*;
public class TesTugas {
    public static void main(String args[]) {
        PengambilanUang tabungan=new PengambilanUang(5000,1000);
        System.out.println("Uang yang ditabung : 5000");
        System.out.println("Uang yang diproteksi : 1000");
        System.out.println("-----------------");
        System.out.println("Uang yang akan diambil : 4500 " +
        tabungan.ambilUang(4500));
        System.out.println("Saldo sekarang : " + tabungan.getSaldo());
        System.out.println("-----------------");
        System.out.println("Uang yang akan diambil : 2500 " +
        tabungan.ambilUang(2500));
        System.out.println("Saldo sekarang : " + tabungan.getSaldo());
    }
}
```

Lakukan kompilasi pada program diatas dan jalankan. Jika tampilan di layar tampak seperti dibawah ini, maka program anda sudah benar. Jika tidak sama, benahi kembali program anda dan lakukan hal yang sama seperti di atas.

```
Uang yang ditabung : 5000
Uang yang diproteksi : 1000
-----------------
Uang yang akan diambil : 4500 false
Saldo sekarang : 5000
-----------------
Uang yang akan diambil : 2500 true
Saldo sekarang : 2500
```

MODUL 7
JAVA GUI Swing
(Pertemuan 7 dan 8)

Tujuan :

Dengan praktikum ini mahasiswa diharapkan dapat:

1. Mahasiswa mengetahui dan mampu mempraktekkannya komponen Swing di Java
2. Mahasiswa mampu membuat form sederhana dengan memanfaatkan komponen Swing di Java

Tugas Pendahuluan :

1. Apa yang Anda ketahui tentang Java GUI Swing dan AWT?
2. Kenapa dalam GUI di Java Swing lebih sering digunakan ?
3. Sebutkan komponen Swing yang ada di Java?

DASAR TEORI

Java swing merupakan toolkit GUI pada Java yang serng dipaka untuk membuat aplikasi dengan Interface berbasis grafis. Beberapa komponen dari Java Swing yaitu :

- Button : Tombol
- Label : teks untuk memberikan suatu keterangan
- Text Field : media input text sepanjang 1 baris
- Text Area : media input text dengan ukuran bisa lebih dari 1 baris
- Menu Bar : Bar yang biasanya menu utama suatu aplikasi
- Menu : Menu-menu pada aplikasi
- Table : untuk menampilkan data dalam bentuk table
- Combo box : media input untuk memilih 1 opsi dari bebrapa opsi
- Radio box : seperti combo box namun semua opsi langsiung ditampilkan
- Chack Box : media input untuk memilih beberapa opsi dari opsi yang tersedia
- Tool bar : bar-bar untuk memilih tool-tool yang disediakan aplikasi dan bisanya ditampilkan dalam bentuk ikon

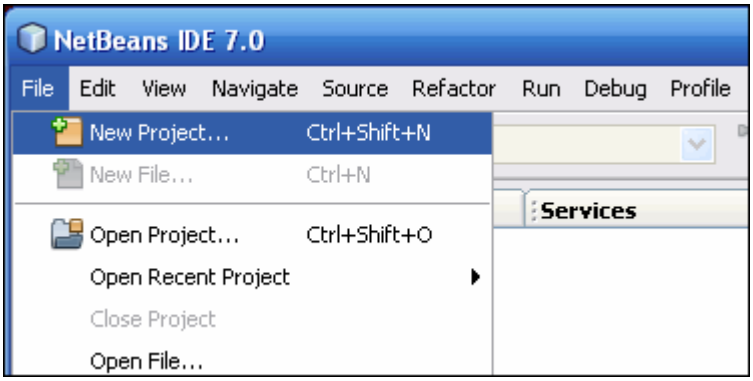
Setiap komponen memiliki metode setter maupun getter untuk mengakses atributnya. Semisal untuk JLabel terdapat metode setText() untuk mengubah tulisan pada label dan pada JTextField terdapat method getText() untuk mengambil data yang diinputkan ke dalam teks.

Setiap komponen juga memiliki yang disebut event listener (atau kadang disebut event handler) yaitu suatu aksi yang dilakukan ketika terjadi suatu event tertentu. Misal, ketika tombol ditekan, ketika teks ditulis dalam text field, dsb.

KEGIATAN PRAKTIKUM

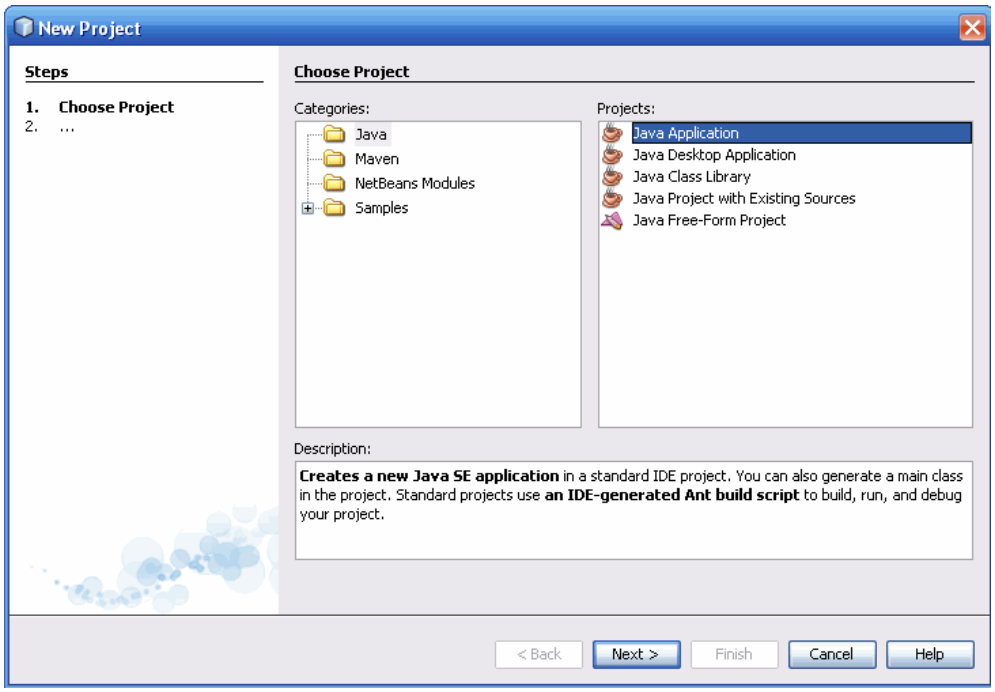
1. Latihan Swing

Buka Netbeans pada PC anda, kemudian Klik **File | New Project** seperti contoh pada **Gambar 7.1** berikut :



Gambar 7.1 Pembuatan New Project

Dalam New Project wizard, pilih kategori Java dan pilih Java Application seperti yang ditunjukkan pada **Gambar 7.2** berikut:



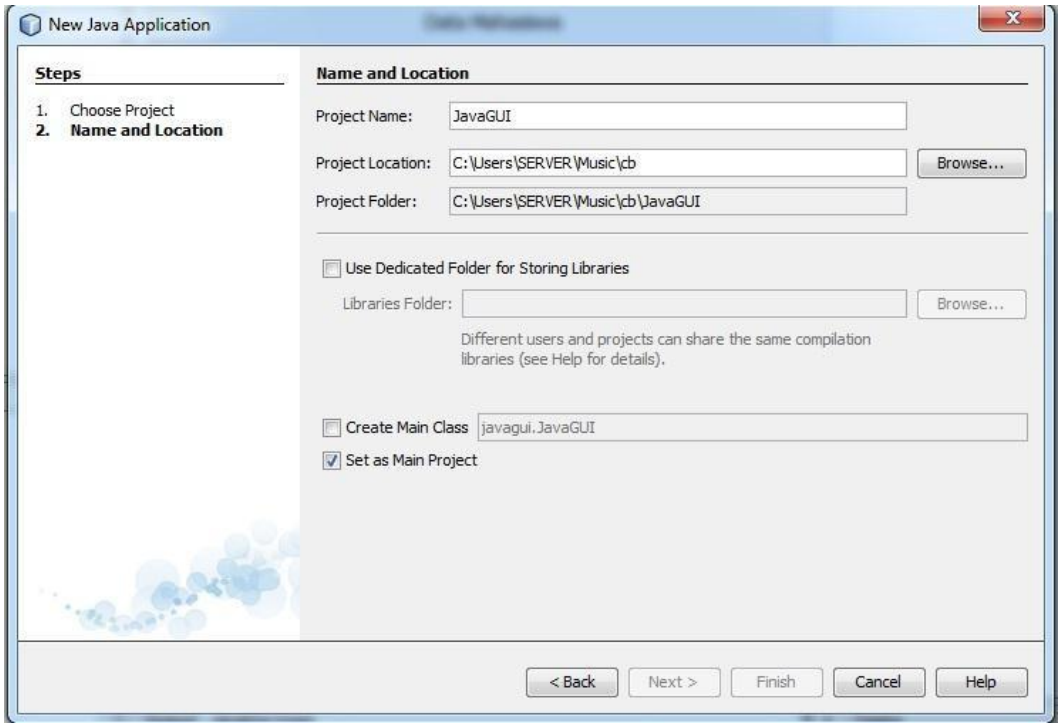
Gambar 7.2 Tampilan Jendela New Project

Dalam halaman wizard Name and Location, lakukan hal berikut (seperti yang ditunjukkan pada gambar di bawah):

- Pada Project Name field, Ketik JavaGUI.

- Pada Create Main Class field, Lepaskan Centang.
- Biarkan kotak centang Set sebagai Main Project dipilih.

Seperti pada contoh **Gambar 7.3** berikut :

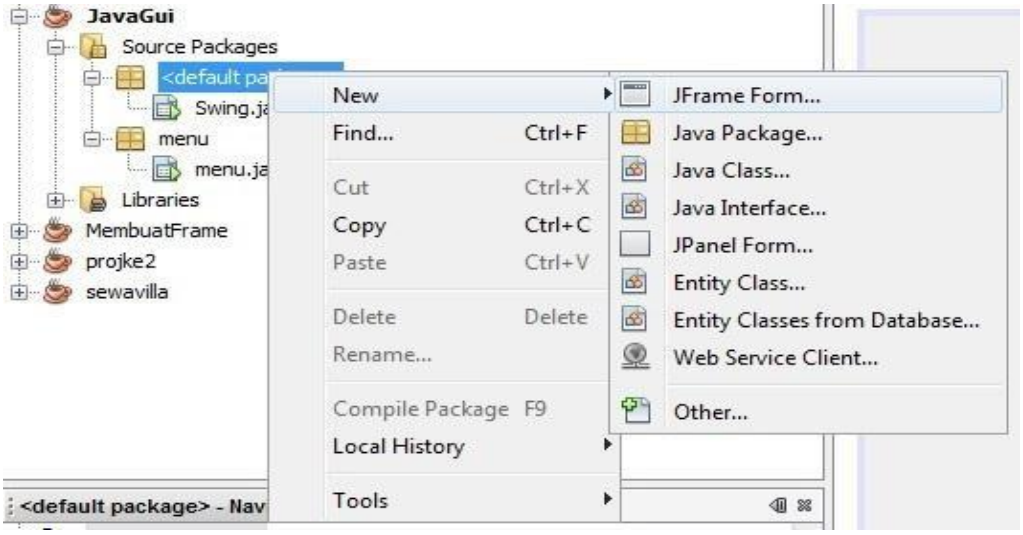


Gambar 7.3 Tampilan Jendela New Java Application

Proyek dibuat dan dibuka dalam IDE. Anda akan melihat komponen-komponen berikut :

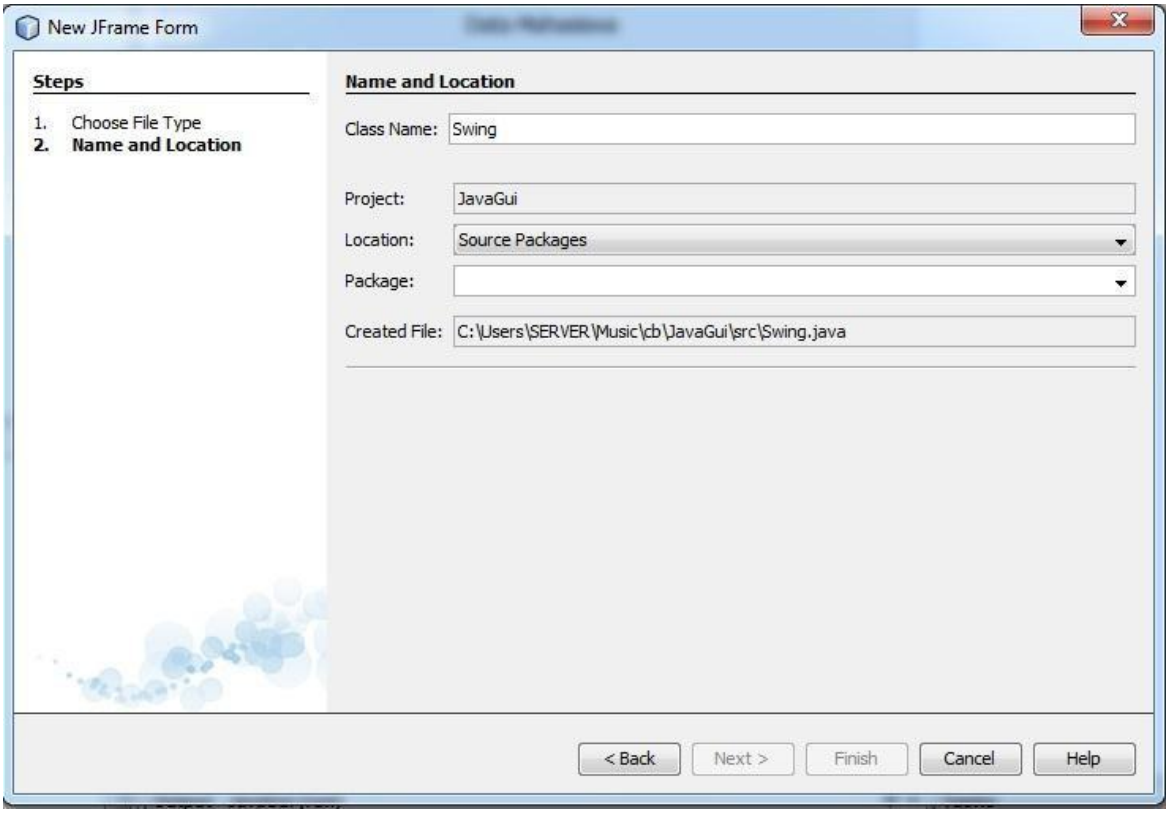
- Jendela Projects, yang berisi tampilan pohon dari berbagai komponen proyek, berisi file sumber, pustaka yang mendasari kode Anda tergantung pada, dan sebagainya.
- Jendela Source Editor dengan sebuah file yang bernama JavaGUI terbuka.
- Jendela Navigator, yang dapat Anda gunakan untuk navigasi cepat di antara elemen-elemen dalam kelas yang dipilih.

Klik kanan pada <default Package> untuk membuat frame New | Jframe Form. Seperti pada gambar berikut.



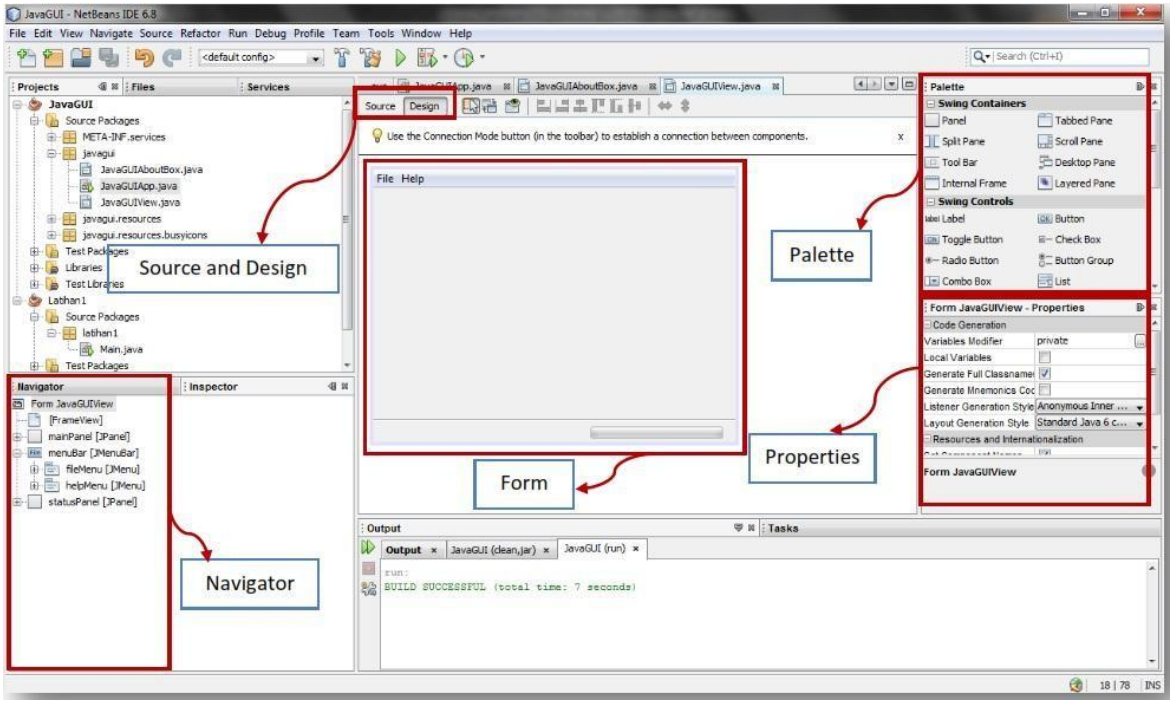
Gambar 7.4 Tahapan Pemilihan New | Jframe Form

Maka akan muncul jendela new Jframe form. Pada field Class Name berikan nama Swing. Lalu klik Finish. Seperti pada gambar berikut.



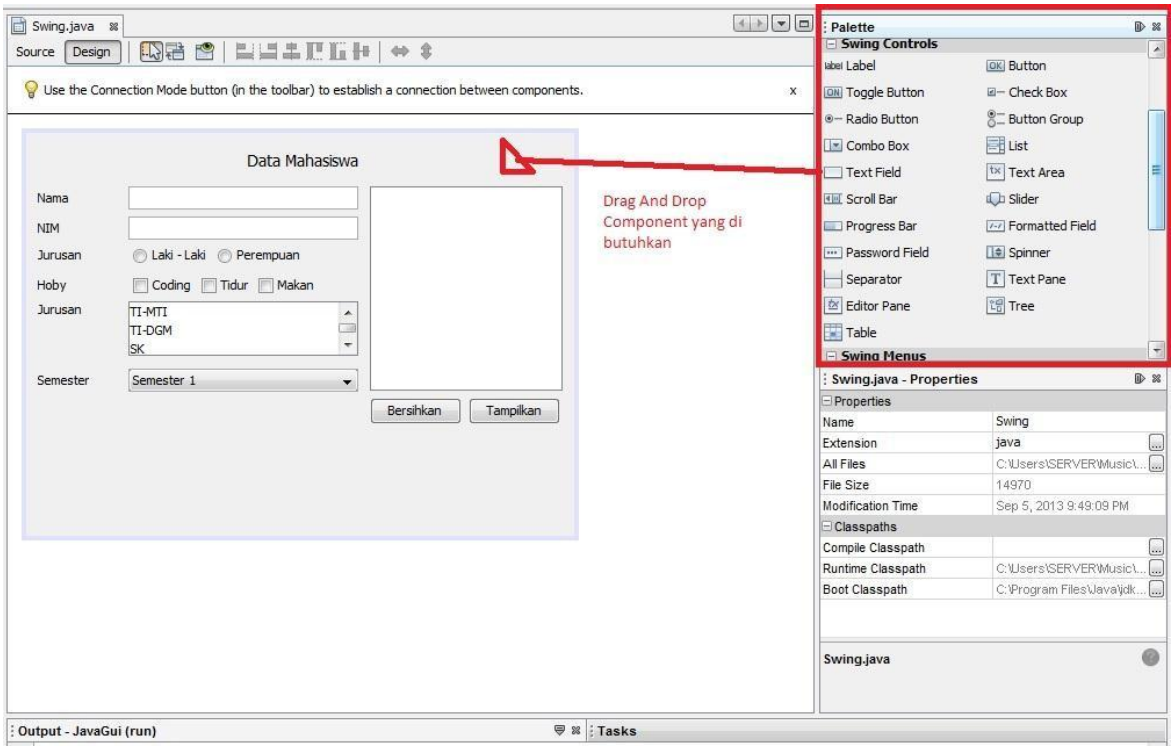
Gambar 7.5 Tampilan Jendela New Jframe Form

Selanjutnya adalah perkenalan bagian NetBeans.



Gambar 7.6 Tampilan Antarmuka NetBeans

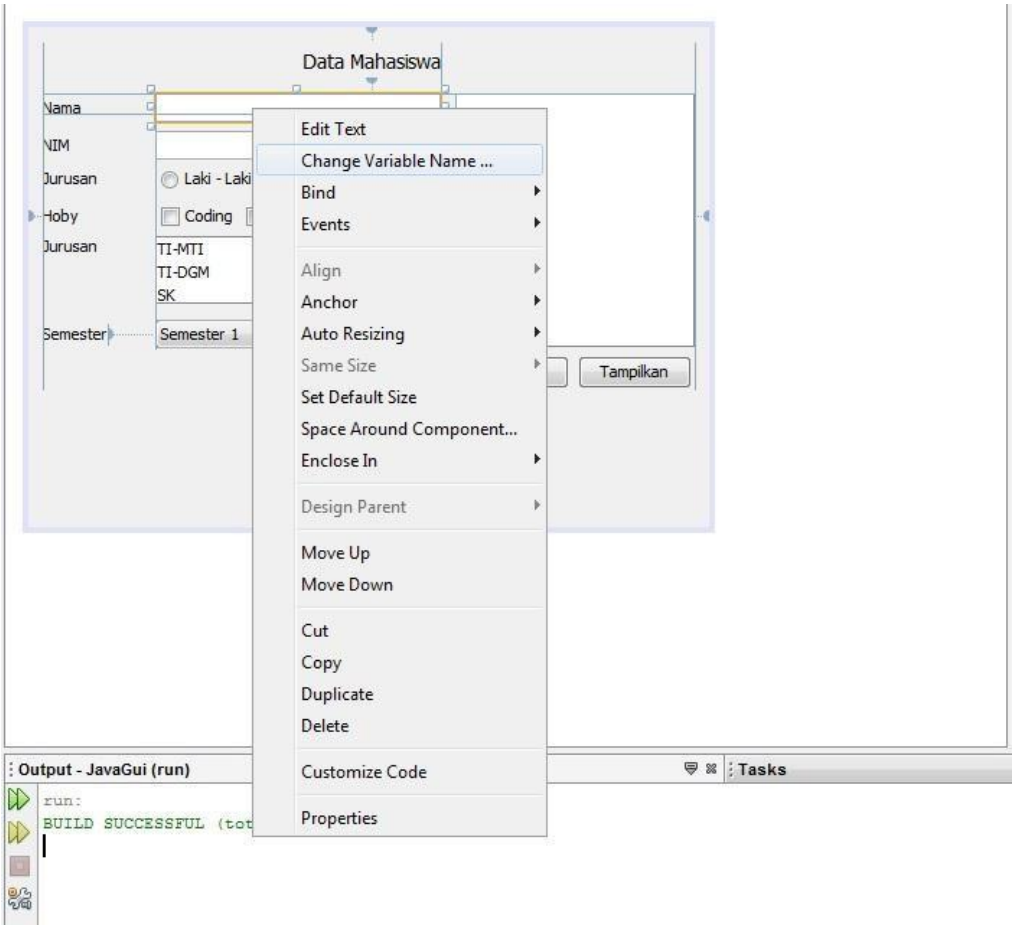
Langkah selanjutnya yaitu lakukan drag dan drop swing yang dibutuhkan, lakukan seperti gambar berikut.



Gambar 7.7 Tahapan Pembuatan Form

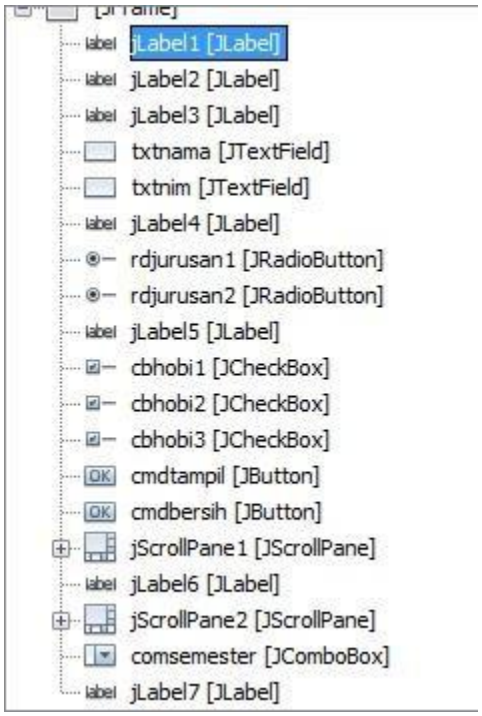
Buatlah design seperti data mahasiswa diatas dengan menggunakan :

1. 2 buah JTextfield
2. 2 buah JRadioButton
3. 3 buah JCheckbox
4. 1 buah JList
5. 1 buah JComboBox
6. 1 buah JTextArea
7. 2 buah JButton
8. 7 buah Jlabel



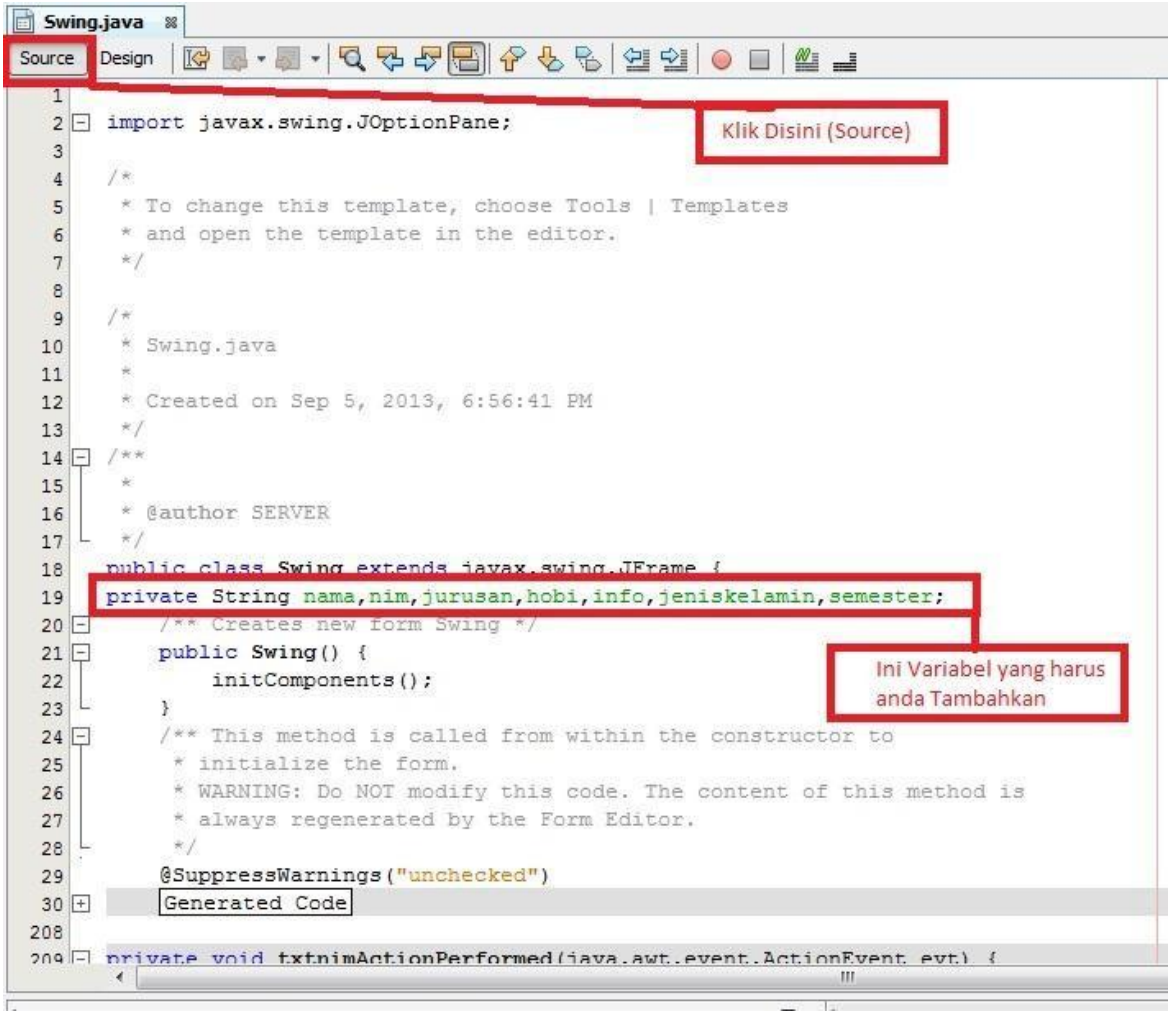
Gambar 7.5 Pembuatan Form Data Mahasiswa

Klik Kanan Pada setiap Swing pilih **Change Variable Name** setelah itu ubah nama **klik ok,** dan klik kanan lagi lalu pilih **Edit Text,**sesuaikan nama variabel dan text seperti pada gambar berikut:



Gambar 7.6 Komponen-komponen pada Form

Kemudian anda dapat memberikan variabel pada source code dengan cara klik source dan ketikan syntax berikut tepat di bawah class yang anda buat.




```
1
2  import javax.swing.JOptionPane;
3
4  /*
5   * To change this template, choose Tools | Templates
6   * and open the template in the editor.
7   */
8
9  /*
10  * Swing.java
11  *
12  * Created on Sep 5, 2013, 6:56:41 PM
13  */
14 /**
15  *
16  * @author SERVER
17  */
18 public class Swing extends javax.swing.JFrame {
19     private String nama,nim,jurusan,hobi,info,jeniskelamin,semester;
20     /** Creates new form Swing */
21     public Swing() {
22         initComponents();
23     }
24     /** This method is called from within the constructor to
25      * initialize the form.
26      * WARNING: Do NOT modify this code. The content of this method is
27      * always regenerated by the Form Editor.
28      */
29     @SuppressWarnings("unchecked")
30     Generated Code
208
209 private void txtnimActionPerformed(java.awt.event.ActionEvent evt) {
```

Gambar 7.7 Source Code (1)

Kemudian kembali ke **Design** dan double klik pada cmdtampil atau klik kanan, kemudian **pilih Event | Action | actionPerformed.** Maka anda akan masuk pada bagian coding / source code. Dan ketikan syntax berikut:

```
private void cmdtampilActionPerformed(java.awt.event.ActionEvent evt) {
  // TODO add your handling code here:
     nama=txtnama.getText();
     nim=txtnim.getText();
     if (cbhobi1.isSelected())
         hobi = "Coding";
     if (cbhobi2.isSelected())
         hobi += " ,Tidur";
     if (cbhobi3.isSelected())
         hobi += " ,Makan";

     if (rdjurusan1.isSelected())
         jeniskelamin = "Laki - Laki";
     else
         jeniskelamin = "Perempuan";

     jurusan = listjurusan.getSelectedValue().toString();
     semester = comsemester.getSelectedItem().toString();

     info="Nama      : "+nama+ "\n";
     info+="NIM      : "+nim+ "\n";
     info+="Jenis Kelamin : "+jeniskelamin+"\n";
     info+="Jurusan : "+jurusan+"\n";
     info+="Semester : "+semester+"\n";
     info+="Hobi     : "+hobi+"";
     hasil.setText(info);
     JOptionPane.showMessageDialog(null, info);

 }
```

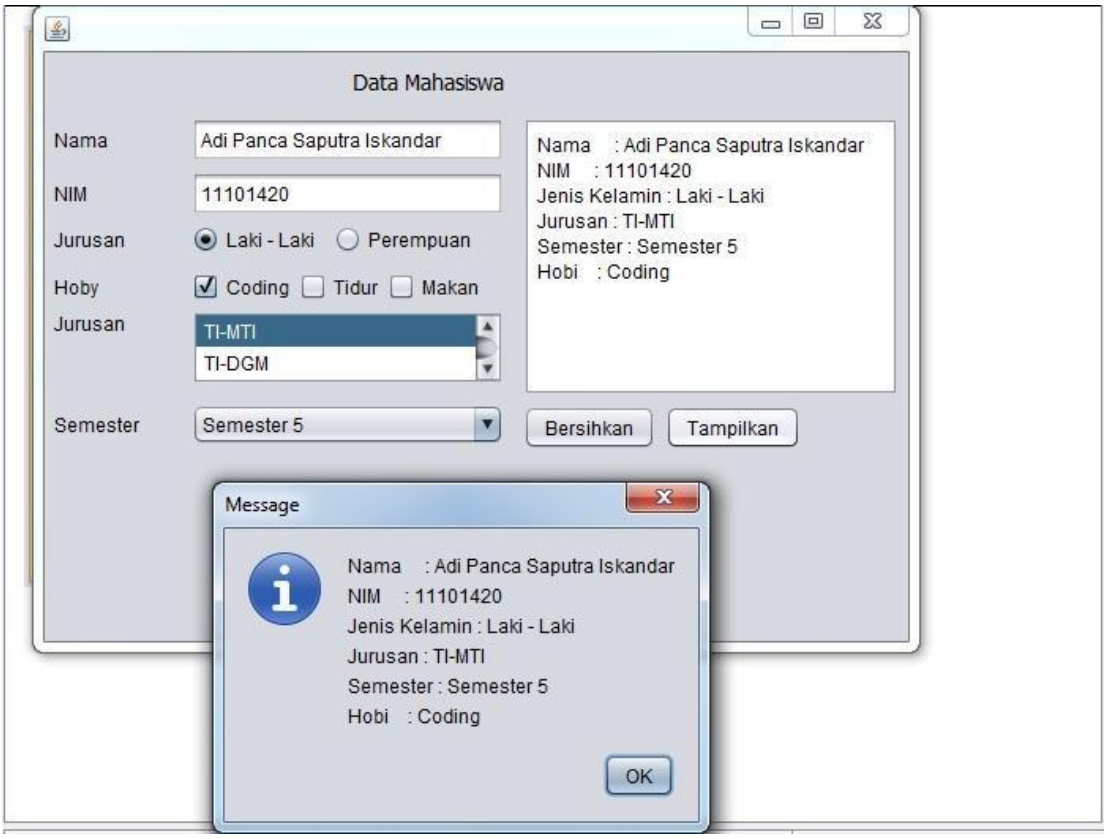
Gambar 7.8 Source Code (2)

Dan pada cmdbersih lakukan hal yang sama dan ketikan syntax berikut.

```
private void cmdbersihActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
     txtnama.setText("");
     txtnim.setText("");
     cbhobi1.setSelected(false);
     cbhobi2.setSelected(false);
     cbhobi3.setSelected(false);
     hasil.setText("");

}
```

Gambar 7.9 Source Code (3)

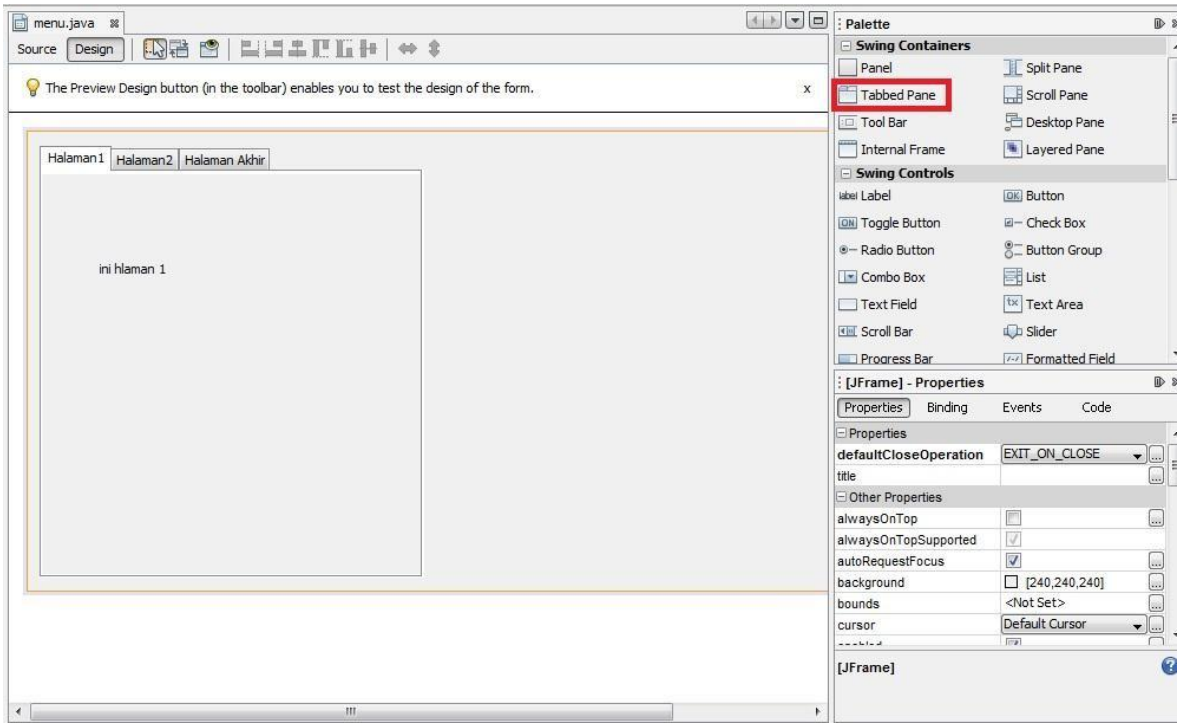
Jika sudah maka project anda siap di **compile** dan di **run**.maka program anda akan tampil seperti gambar berikut dan silahkan diisi data anda dan klik **Tampilkan**.



Gambar 7.9 Form Data Mahasiswa

2. Latihan JTabPane, JTree, JTable

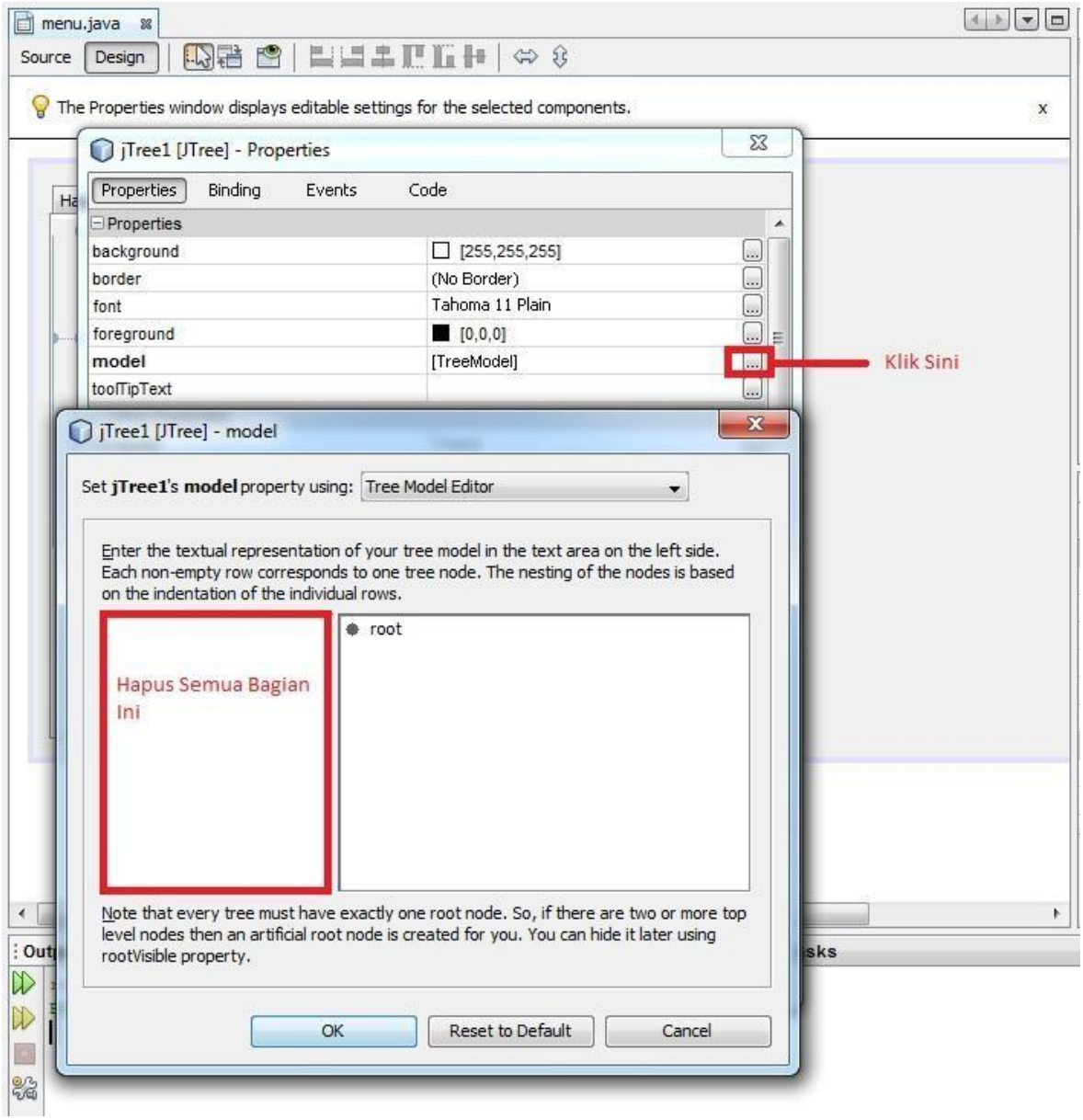
Dalam Pembuatan JTabPane lakukan hal yang sama **new project** hingga **new frame** kemudian pilih **Design** lalu **drag & drop JtabPane**, **s**eperti pada gambar berikut.



Gambar 7.10 Pembuatan JtabPane

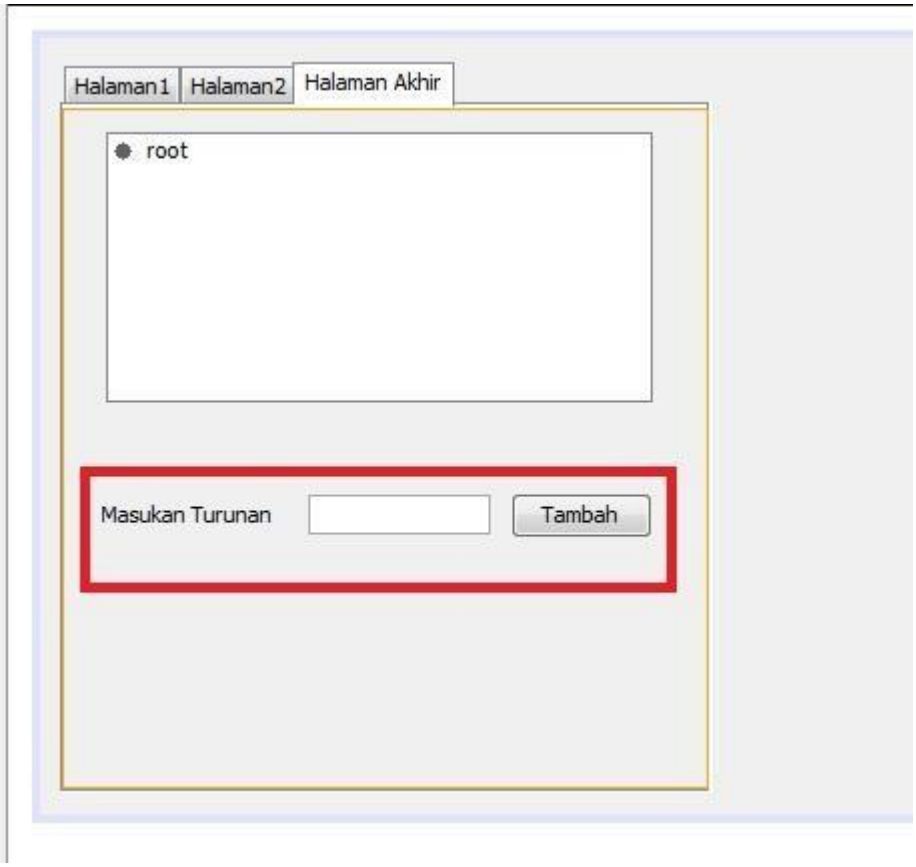
Setelah itu Drag & Drop **Panel** kedalam **JTabPane** maka Tab berhasil di buat.lakukan drag & drop **panel** hingga **JTabPane** memiliki 3 Tab. Setelah itu anda dapat mengedit Text pada Tab tersebut dengan cara yang sama yaitu klik kanan pada tab tersebut dan **Edit Text.** Ubahlah menjadi **Halaman1 , Halaman 2 dan Halaman Akhir.**

Pada pada halaman 1 anda dapat memberikan **JLabel** untuk menandai ini halaman1. Sedangkan pada halaman2 anda dapat memberikan **JTabel** dengan cara darg & drop Jtabel ke Halaman2, dan pada halaman 3 anda dapat memberikan **JTree**. Kemudian anda dapat klik kanan pada **JTree** dan pilih **Property** selanjutnya klik pada Model kemudian klik titik2 pada bagian kanan model "…" Maka akan muncul jendela baru Dan hapus semua item di kolom sebelah kiri seperti gambar berikut.



Gambar 7.11 Pengaturan Properties

Kemudian tambahkan 1 buah Jlabel, 1 buah JTextField, dan 1 buah JButton, seperti yang ditampilkan pada gambar berikut:



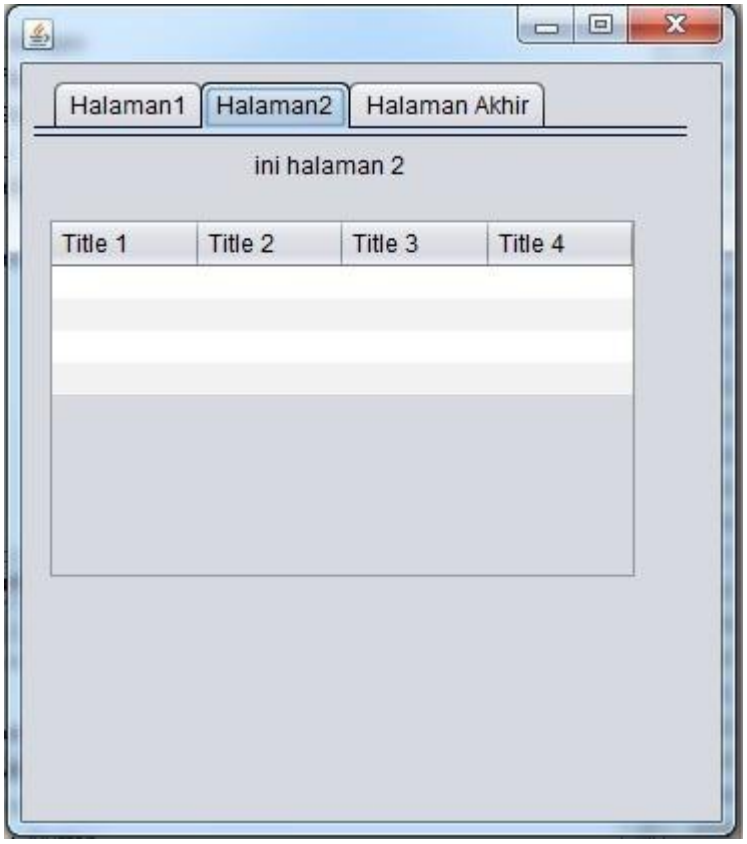
Gambar 7.12 Pengaturan Properties

Kemudian double klik pada Jbutton atau klik kanan, kemudian **pilih Event | Action | actionPerformed.** Maka anda akan masuk pada bagian coding / source code. Dan ketikan syntax berikut.

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
    DefaultMutableTreeNode admin = new DefaultMutableTreeNode(jTextField1.getText());
    DefaultMutableTreeNode set = (DefaultMutableTreeNode) jTree1.getLastSelectedPathComponent();
    DefaultTreeModel dt= (DefaultTreeModel) jTree1.getModel();
    dt.insertNodeInto(admin, set, set.getChildCount());

}
```

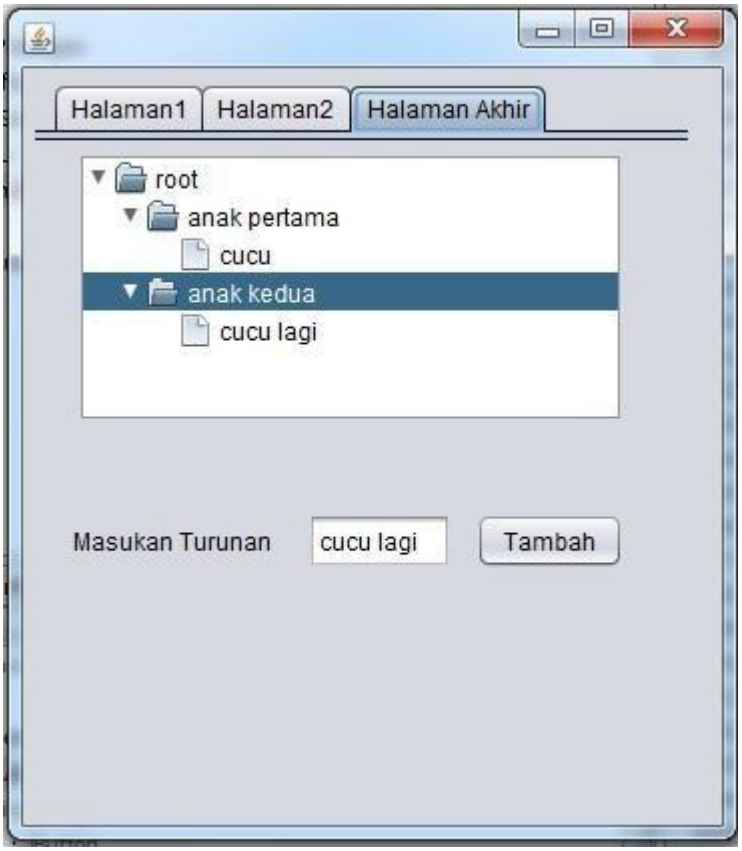
Gambar 7.13 Source Code (4)

Setelah itu Anda dapat Compile dan Run Project Anda, maka akan tampil sebagai berikut.



Gambar 7.14 Hasil Eksekusi Program (1)

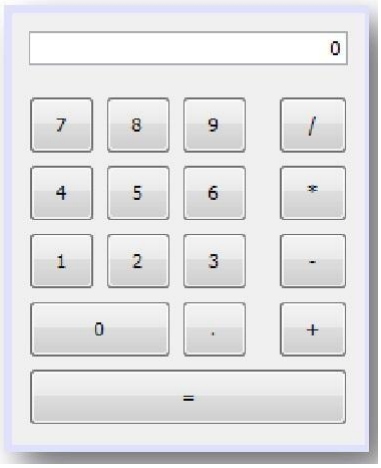
Pilih Halaman Akhir dan Klik pada **Jtree** ROOT. Dan ketikan anak pertama pada JTextField. Dan klik Tambah. Maka **Jtree** anda akan bertambah seperti pada gambar berikut.



Gambar 7.15 Hasil Eksekusi Program (2)

TUGAS

Buatlah sebuah program yang mempunyai fungsi seperti kalkulator (mampu menjumlahkan,mengurangkan,mengalikan dan membagikan seperti berikut:



MODUL 8
JavaGUI , JDBC Connector & MySQL
(Pertemuan 9 dan 10)

Tujuan :

Diharapkan setelah praktikum ini mahasiswa diharapkan dapat:

1. Mahasiswa dituntun untuk membuat form sederhana dengan memanfaatkan komponen Swing
2. Koneksi ke database Mysql dengan memanfaatkan JDBC Connector
3. Form yang telah terkoneksi dapat melakukan insert, update, delete

Tugas Pendahuluan :

1. Apa yang ketahui tentang JDBC Connector ?
2. Apa yang ada ketahui tentang Mysql?

DASAR TEORI

JDBC merupakan singkatan dari Java Database Connectivity yaitu API java yang membantu aplikasi java untuk mengeksekusi SQL statement. JDBC merupakan interface pemrograman aplikasi yang mendefinisikan bagaimana pemrogram java dapat mengakses database dalam format tabular dari kode2 java menggunakan sekumpulan interface standard an class-class yang tertulis dalam bahasa java.

Interface pemrograman aplikasi java menyediakan mekanisme untuk memuat package java beserta driver-driver dan memasang pada JDBC Driver Manager yang digunakan sebagai pembuat koneksi JDBC yang mendukung eksekusi syntax SQL seperti INSERT, UPDATE dan DELETE. Driver Manager adalah bagian itama dari JDBC.

Secara umum semua RDBMS (Relational Database Management System) dan Java merupakan platform yang independen, jadi JDBC memungkinkan untuk membuat sebuah aplikasi database yang dapat dijalankan pada platform-platform berbeda dan berinteraksi dengan DBMS.

Singkatnya, JDBC membantu programmer untuk membuat aplikasi java yang mengatur 3 aktivitas pemrograman ini :

1. Membantu menghubungkan dengan sumber data, seperti database
2. Membantu mengirim query dan perintah update ke database
3. Menerima dan memroses hasil yang diterima dari database sebagai respon dari query yang dikirim

KEGIATAN PRAKTIKUM

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.

1. **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;
```

2. **Membuat Tabel**

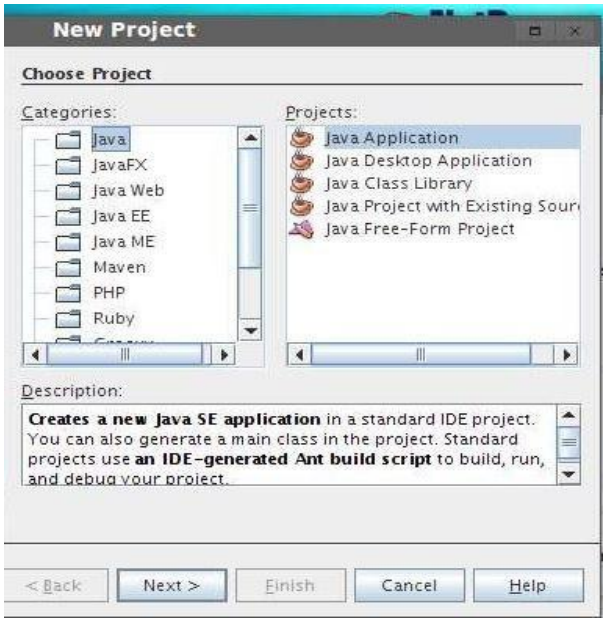
   Setelah membuat database, kita terlebih dahulu perlu membuat tabel MAHASISWA. Kita dapat menggunakan perintah sebagai berikut untuk membuat tabel MAHASISWA :

```
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.

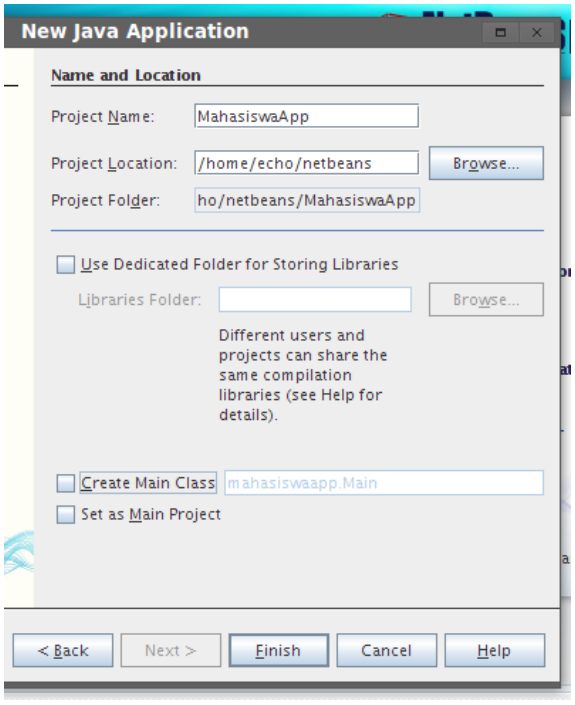
3. Membuat Project

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.



Gambar 8.1 Pembuatan Project Baru (1)

Klik Next untuk melanjutkan pembuatan project.

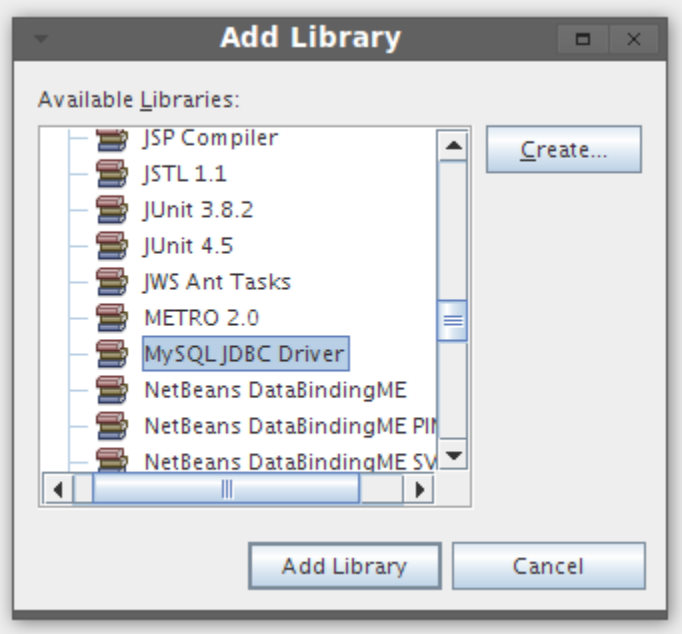


Gambar 8.2 Pembuatan Project Baru (2)

Beri nama project dan jangan diceklis ceckbox 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.

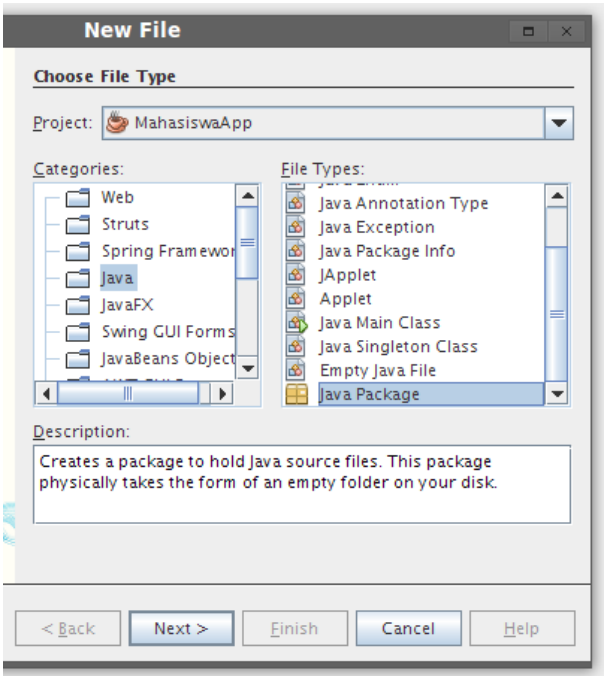
4. Membuat Koneksi MySQL

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.



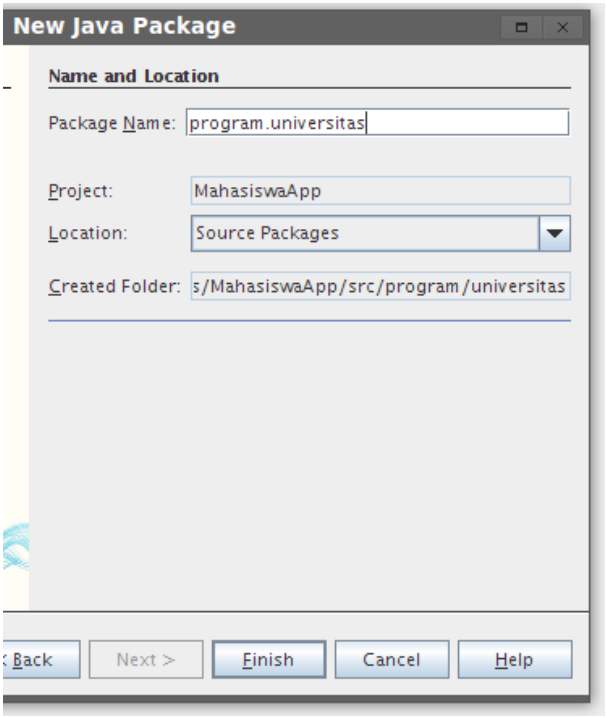
Gambar 8.2 Pembuatan Project Baru (2)

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 Next –> Other.



Gambar 8.3 Tahapan Pembuatan Package

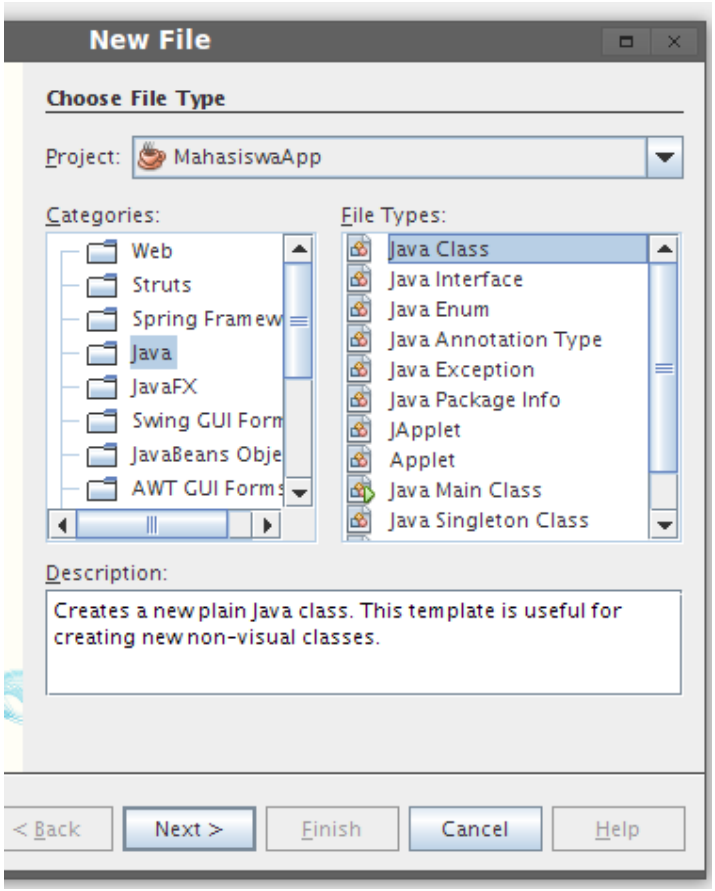
Setelah keluar dialog Next File, pilih kategori Java dan jenis file Java Package. Klik Next untuk melanjutkan membuat package.



Gambar 8.4 Tampilan Jendela New Java Package

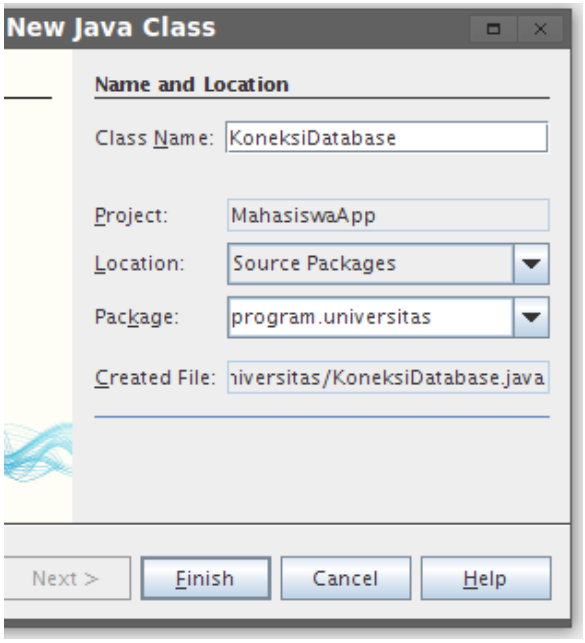
Setelah itu beri nama packagenya, misal **program.universitas**, setelah itu klik Finish untuk membuat package−nya.

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



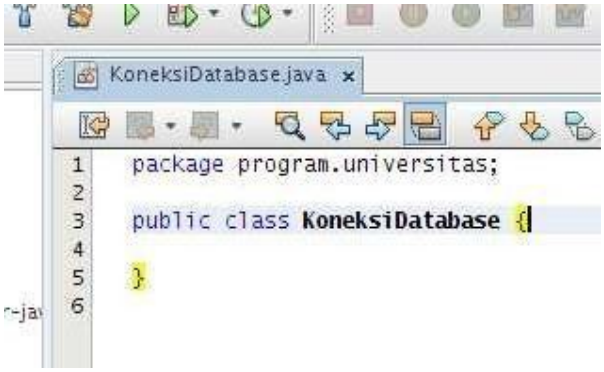
Gambar 8.5 Pembuatan Class Baru

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



Gambar 8.6 Tampilan Jendela New Java Class

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



Gambar 8.7 Tahapan Pembuatan Class

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?

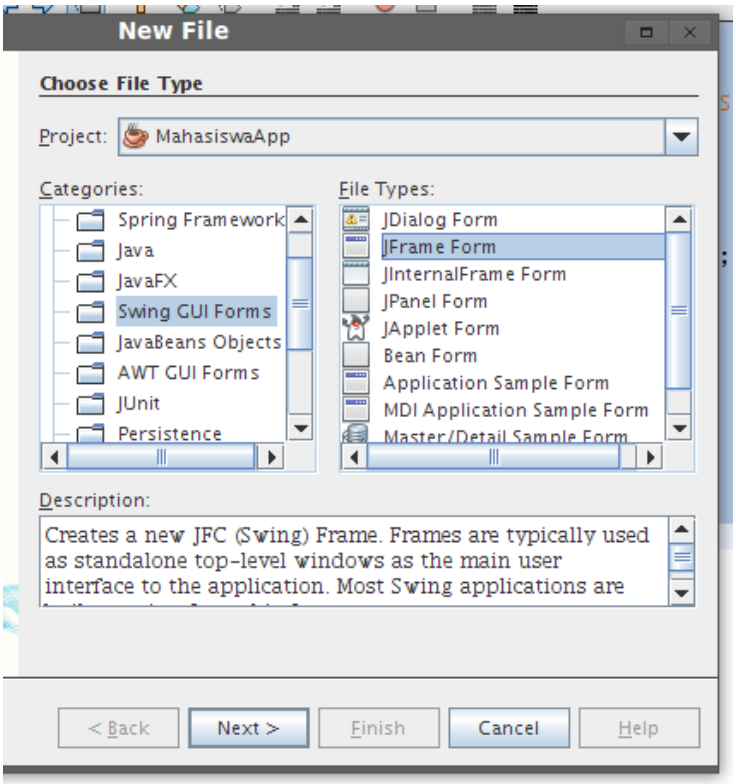
Tentu karena pada kelas KoneksiDatabase tersebut kita belum membuat koneksinya, jadi sebelum return koneksi, pada metode getKoneksi() seharusnya kita cek dulu apakah koneksi–nya null, jika null, maka kita deklarasikan sebuah koneksi yang baru.

```
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 = "sssss";
                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.

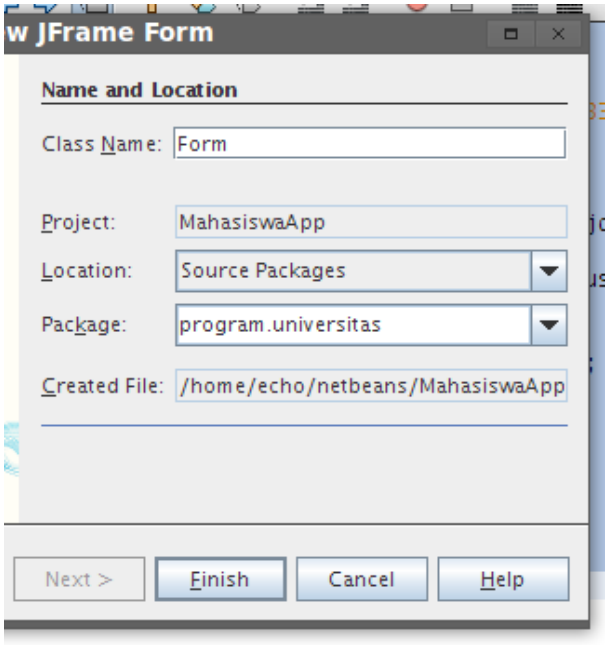
5. Membuat Form Aplikasi

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.



Gambar 8.8 Pemilihan JFrameForm

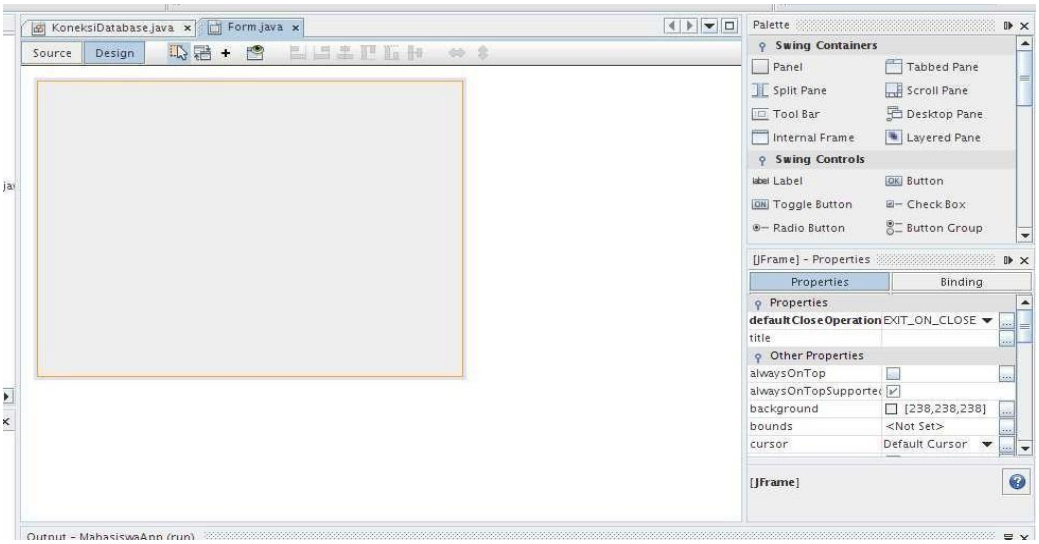
Pilih kategori String GUI Forms dan pilih tipe file JFrame Form. Lalu klik Nest untuk melanjutkan membuat Form.



Gambar 8.9 Tahapan Pembuatan Jframe Form

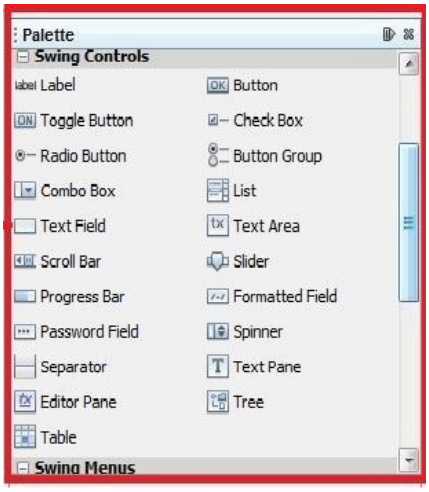
Beri nama Form tersebut, misal dengan nama Form, dengan begitu maka NetBeans akan membuatkan sebuah kelas dengan nama Form yang merupakan turunan dari kelas JFrame, dimana kelas JFrame ini merupakan kelas Java Swing.

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

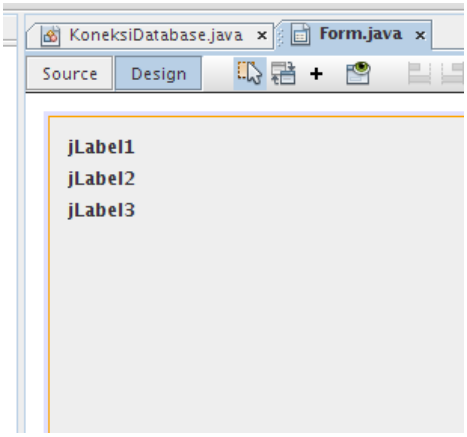


Gambar 8.10 Tahapan Pembuatan Form

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.

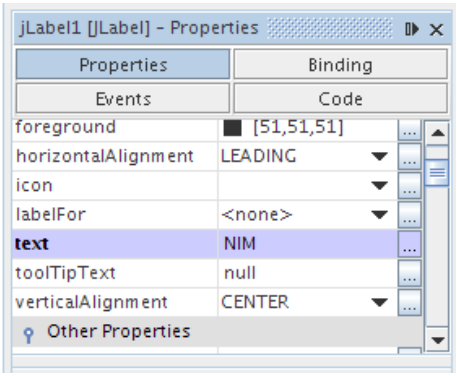


Gambar 8.11 Penggunaan Pallete



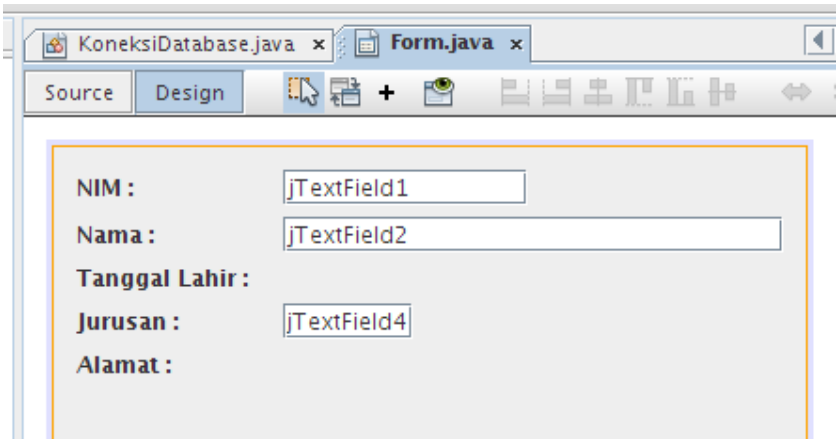
Gambar 8.12 Menyusun Pelabelan

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



Gambar 8.13 Mengatur Properties

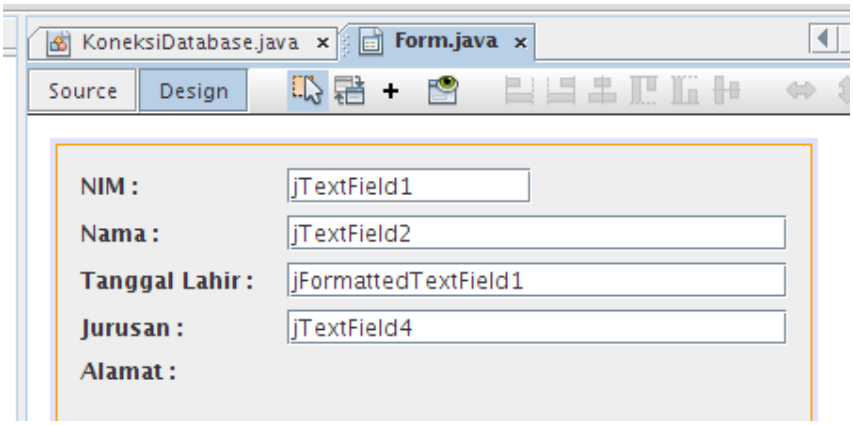
Setelah itu klik dan drag tiga buat Text Field yang ada dipallete ke Form, gunakan Text Field untuk Nim, Nama dan Jurusan.



Gambar 8.14 Pengaturan Form (1)

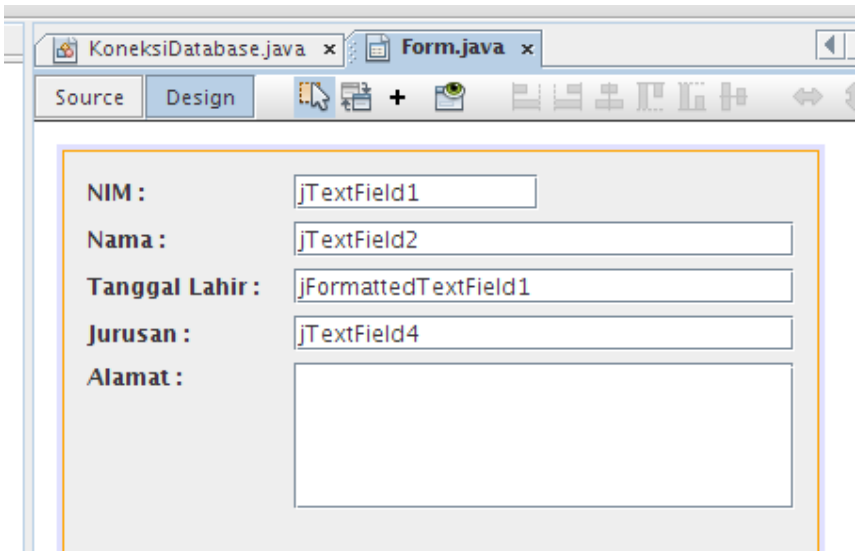
Untuk Tanggal Lahir dan Alamat kita tidak menggunakan Text Field, hal ini dikarenakan Tanggal Lahir memerlukan inputan berupa tanggal sedangkan Text Field hanya mendukung teks (string), sedangkan untuk Alamat, biasanya isi alamat itu panjang, sehingga lebih tidak cocok menggunakan Text Field, karena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



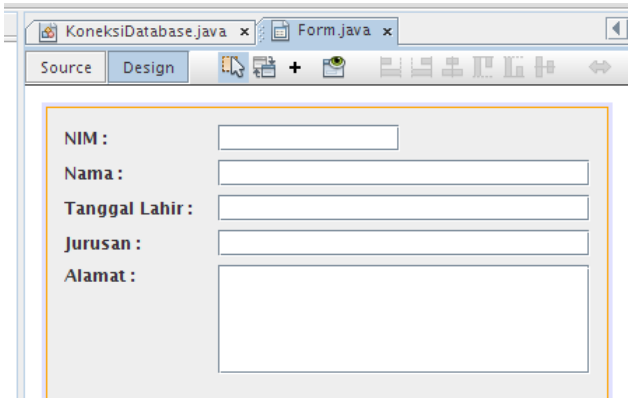
Gambar 8.15 Pengaturan Form (2)

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



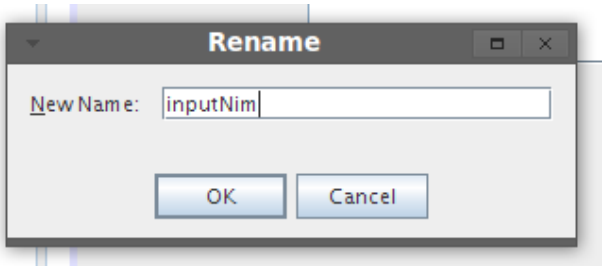
Gambar 8.16 Pengaturan Form (3)

Untuk mengosongkan isi tulisan pada NIM, Nama, Tanggal Lahir dan Jurusan, kosongkan atribut Text pada setiap komponen pada Properties−nya.



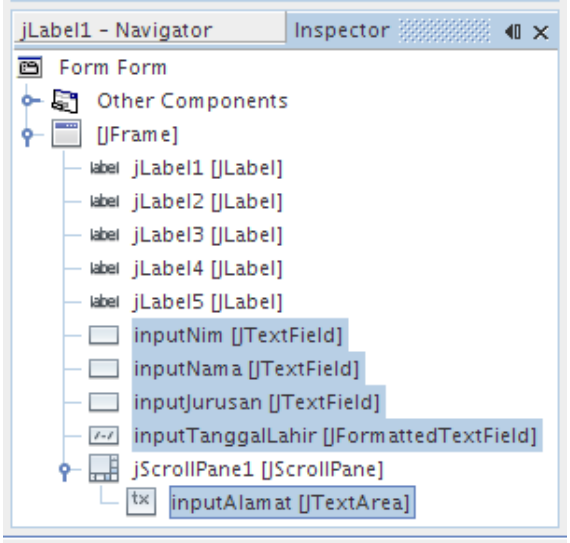
Gambar 8.16 Pengaturan Form (4)

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**.



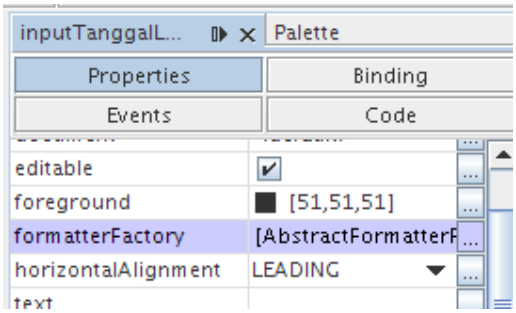
Gambar 8.17 Pengaturan Form (5)

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



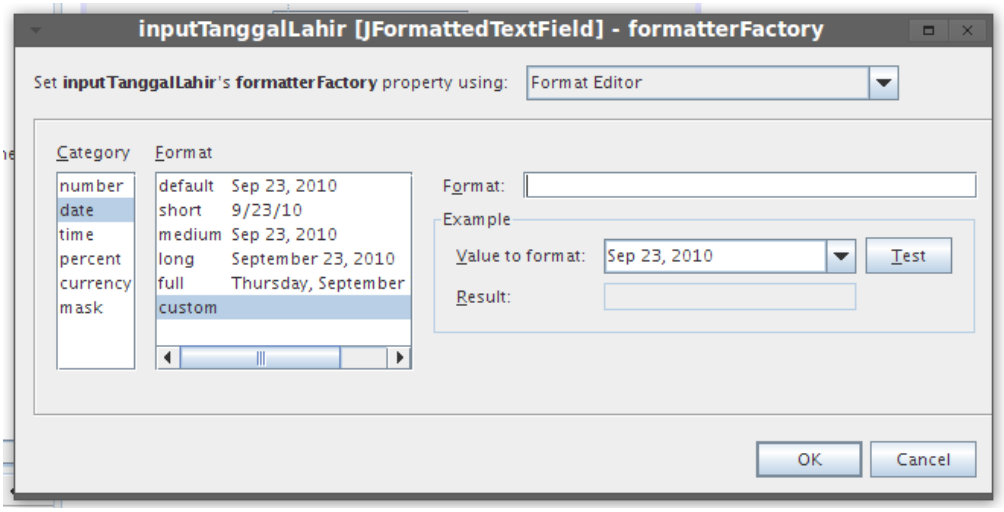
Gambar 8.18 Tampilan Setiap Komponen pada Form

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.



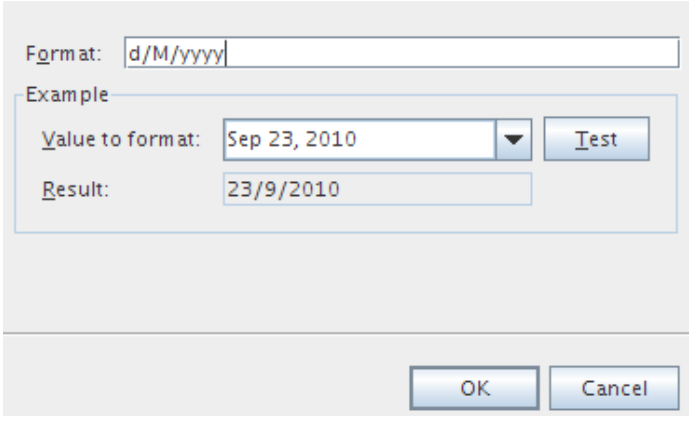
Gambar 8.19 Pengaturan Properties

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



Gambar 8.20 Pengaturan Tanggal (1)

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

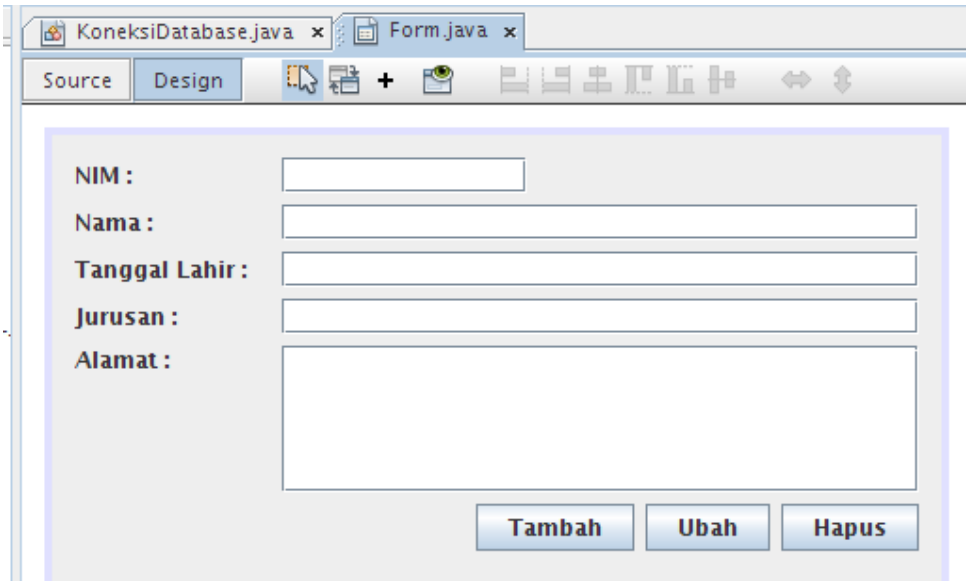


Gambar 8.21 Pengaturan Tanggal (2)

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.

6. Menambah Tombol ke Form

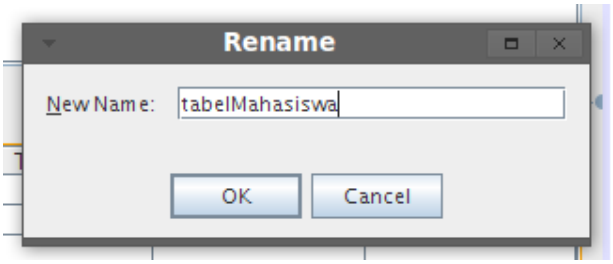
Setelah menambahkan input Form, sekarang saatnya kita menambah tombol ke dalam Form. Caranya dengan mengklik dan drag komponen Button pada Palette ke dalam Form.



Gambar 8.22 Penambahan Tombol ke Form

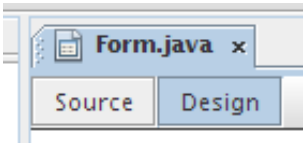
Tambahkan 3 buah tombol, Tambah, Ubah dan Hapus. Untuk mengubah teks tombolnya caranya sama seperti Label, yaitu dengan mengubah atribut Text pada Properties.

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.



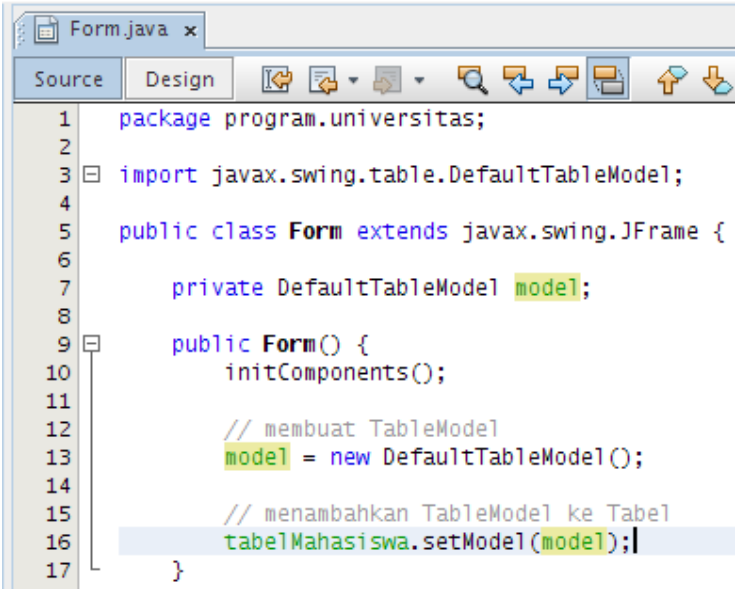
Gambar 8.23 Penggantian Nama Tabel

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.



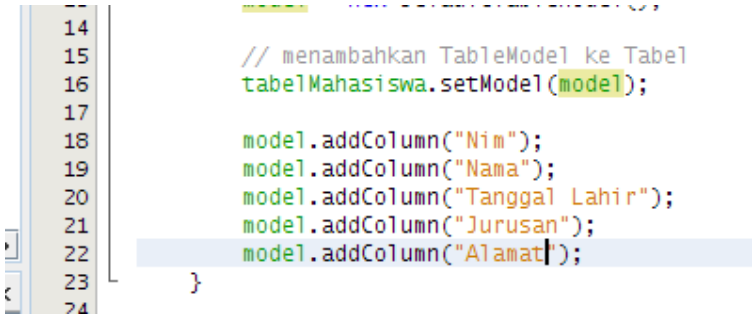
Gambar 8.24 Bagian Source

Setelah itu tambahkan sebuah variabel DefaultTableModel pada kelas Form tersebut.



Gambar 8.25 Source Code (1)

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.



Gambar 8.26 Source Code (2)

7. 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.

8. Menambah Aksi Load Data

Saat pertama kali aplikasi muncul, maka otomatis kita h arus 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.

Namun sebelum anda membuat method tersebut tambahkan beberapa import dibawah ini.
Tambahkan tepat di bawah coding package proram.universitas;

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

Import diatas merupakan bawaan dari java sehingga syntax yang akan anda buat tidak mengalami eror / user define.

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 load 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.creaTextatement();



        String sql = "SELECT * FROM MAHASISWA";

        ResultSet r = s.executeQuery(sql);

        while(r.nest()){

             // 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.nest()){

    // 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.nest()){

            // lakukan penelusuran baris

            Object[] o = new Object[5];
            o[o] = 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(SQLEsception 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
```

Gambar 8.27 Source Code (3)

9. 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 -> actionPerfor ed**, maka otomatis NetBeans IDE akan membuatkan sebuah metode baru untuk aksi tombol Tambah.

```
197
198     private void jButton3ActionPerformed(java.awt.event
199         // TODO add your handling code here:
200     }
201
```

Gambar 8.27 Source Code (4)

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 k e l a s KoneksiDatabase setelah itu membuat PrepareStatement untuk menyimpan datanya.

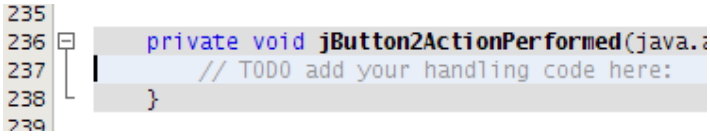
```
String nim = inputNim.getText(); String nama =  inputNama.getText();
java.util.Date tanggalLahir = (java.util.Date) inputTanggalLahir.getValue();
String jurusan =inputJurusan.getText();
String alamat =inputAlamat.getText();

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(SQLEsception 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.

10. 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**.



Gambar 8.28 Source Code (5)

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();

  }
```

11. 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**.

Gambar 8.29 Source Code (6)

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 =(String) 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{
```

12. 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 -> ouse ->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
```

Gambar 8.30 Source Code (7)

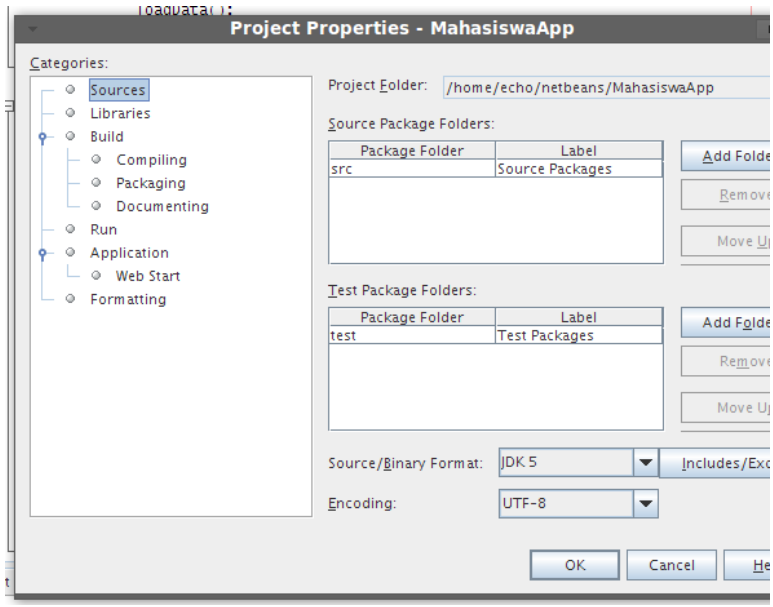
Pertama yang harus dilakukan adalah melakukan pengecekah apakah ada baris yang terseleksi, jika ada maka ambil data yang terseleksi dari DefaultTableModel setelah itu tampilkan pada Form, namun jika tidak ada baris yang terseleksi, maka batalkan proses.

```
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);
```

13. Run Program

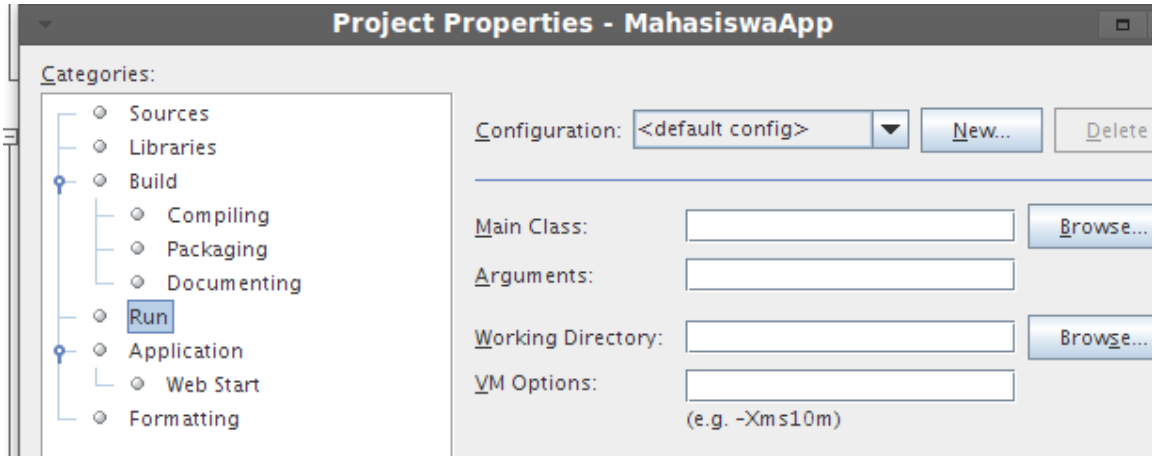
Setelah pembuatan program telah selesai, saatnya menjalankan aplikasi. Untuk menjalankan aplikasi, pertama kita harus menentukan dahulu kelas yang akan digunakan sebagai program, dimana pada project yang telah kita buat, kelas program adalah kelas Form.

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.



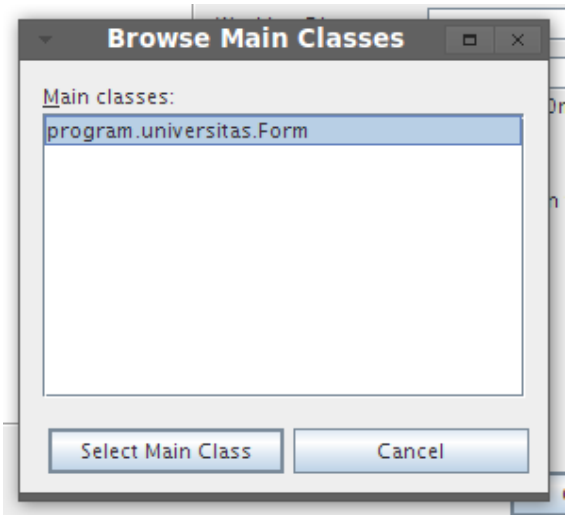
Gambar 8.31 Project Properties (1)

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



Gambar 8.32 Project Properties (2)

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

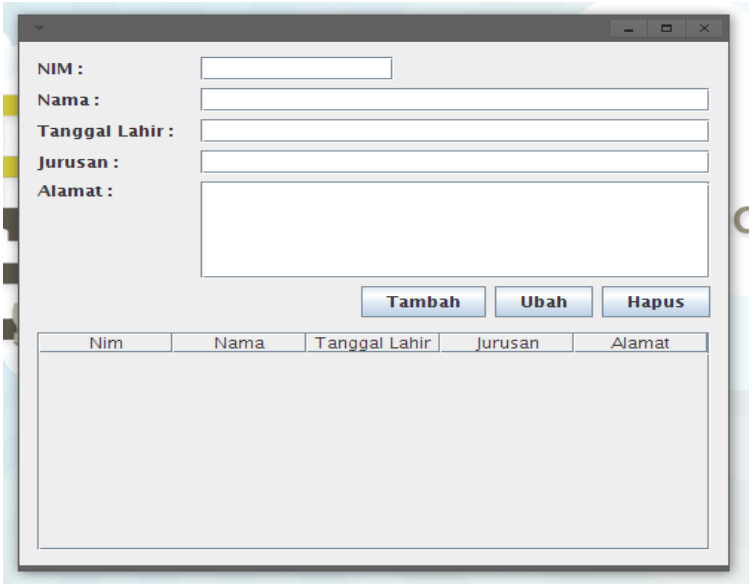


Gambar 8.33 Browse Main Classes

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.

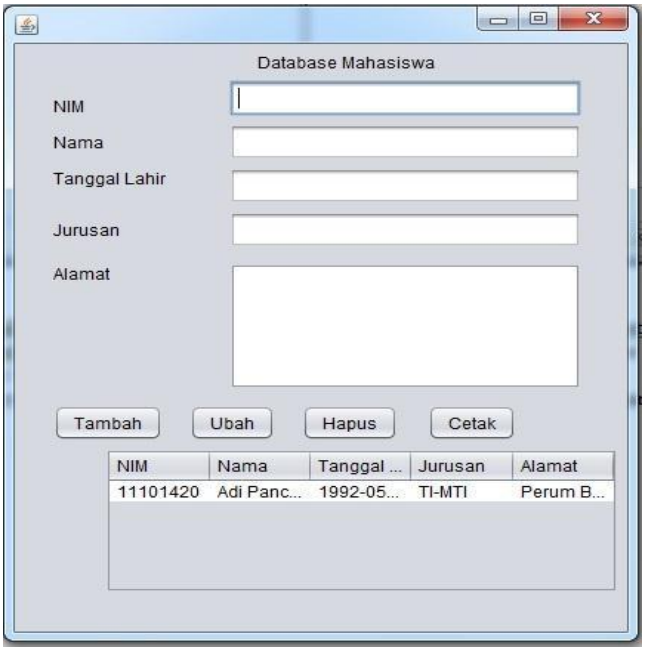
Untuk menjalankan aplikasi, klik kanan project yang telah kita buat, setelah itu pilih menu

Run, maka otomatis program akan berjalan. Pada komputer saya, program terlihat seperti pada gambar dibawah ini.



Gambar 8.34 Hasil Eksekusi Program (1)

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



Gambar 8.35 Hasil Eksekusi Program (2)

TUGAS

Implementasikan materi yang telah didapat dalam project sistem informasi yang telah dirancang!

MODUL 9
JavaGUI Database Dengan iReport
(Pertemuan 11, 12 dan 13)

Tujuan :

Diharapkan setelah praktikum ini mahasiswa diharapkan dapat:

1. Mahasiswa bisa dituntun untuk membuat single form yang dilengkapi dengan report
2. Form yang telah terkoneksi dapat melakukan insert update delete

Tugas Pendahuluan :

1. Apa yang anda ketahui tentang iReport?

DASAR TEORI

Terdapat banyak tools untuk reporting dalam java. Diantaranya yang dapat digunakan adalah :

1. JasperReports

   Merupakan software open source untuk reporting
2. iReport

   Merupakan Visual Designer untuk membuat laporan yang komplek menggunakan JasperReports library tanpa harus memiliki pengetahuan tentang XML .

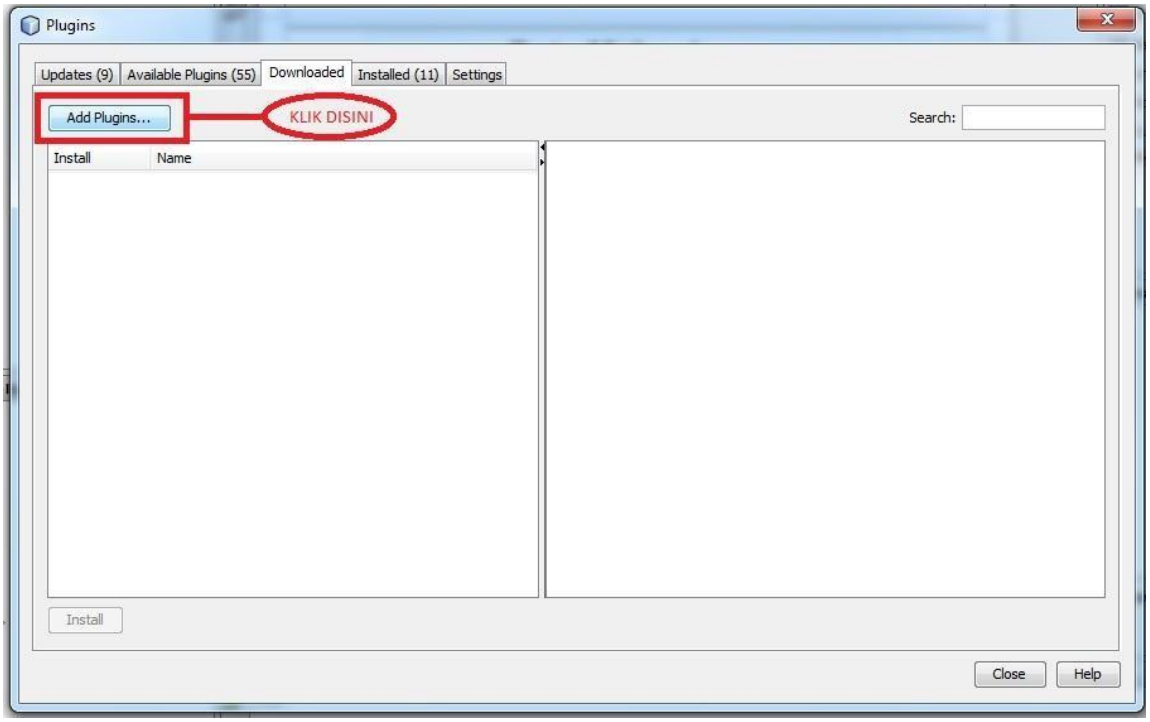
Beberapa fitur iReport :

- 98% mendukung JasperReports tags
- Visual designer wysiwyg untuk menggambar rectangles, lines, ellipses, text fields fields,charts, sub reports...
- Built-in editor dengan syntax highlighting
- Mendukung Unicode dan bahasa non Latin (Russian, Chinese, Korean,...)
- Document structure browser
- Menggabungkan compiler dan exporter
- Mendukung semua JDBC compliant databases
- Memiliki Wizard untuk membuat report secara otomatis
- Mendukung sub reports
- Save backup
- Support for templates

KEGIATAN PRAKTIKUM

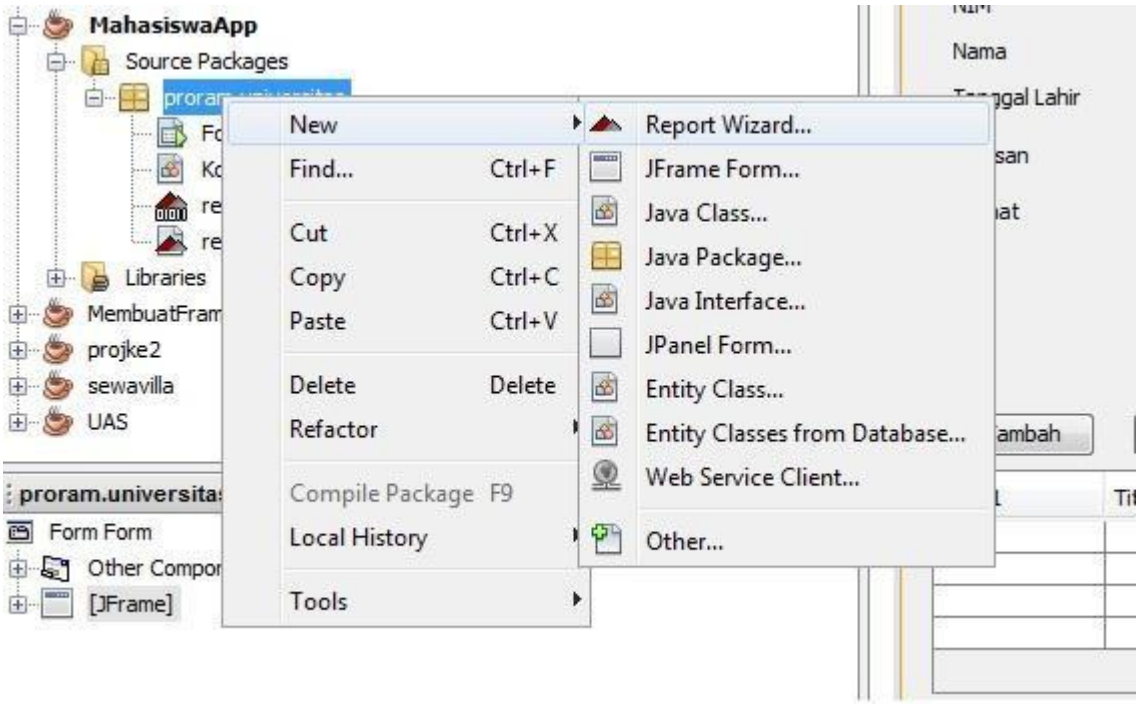
Melanjutkan project pada module 9. Kita akan membuat report dengan iReport. Sebelum anda mempraktekan iReport. Anda perlu mendownload file yang sudah saya siapkan di link ini : http://www.4shared.com/rar/wbWFR7cb/Java-Report-file.html atau https://docs.google.com/file/d/0BxYcW8LUKhiRMUpDVnVjSEIxM0k/edit?usp=sharing

1. Buka Netbeans, selanjutnya pilih **Tools | Plugin**. Pada tab download klik **add plugin** dan masukkan plugin iReport yang sudah anda download tepatnya didalam folder **iReport- nb-3.5.2-plugin**.



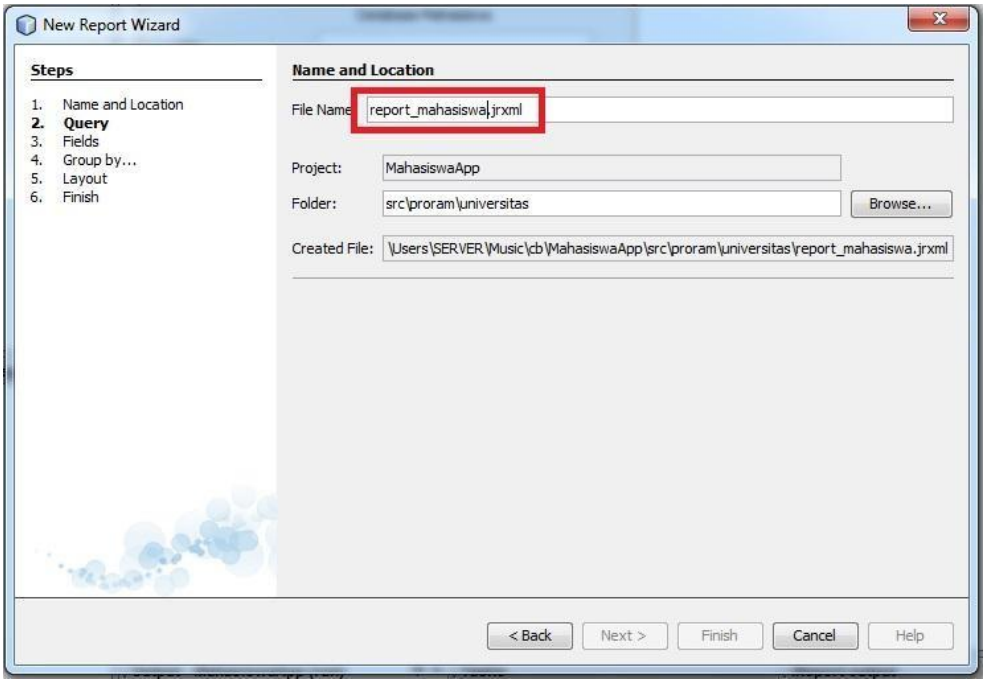
Gambar 9.1 Tampilan Jendela Plugins

2. Klik Install dan ikuti langkah selanjutnya. Hingga anda diminta untuk merestart IDE Netbeans anda. Biarkan IDE di restart.
3. Setelah itu buka kembali IDE Netbeans anda dan buka pada project MahasiswaApp atau project module 9 yang sudah dibuat dan buat file Report dengan cara klik kanan pada **program.universitas** | pilih **New** | pilih **Report Wizard** atau pilih **other** | pilih **iReport** | pilih **Report Wizard** seperti gambar.



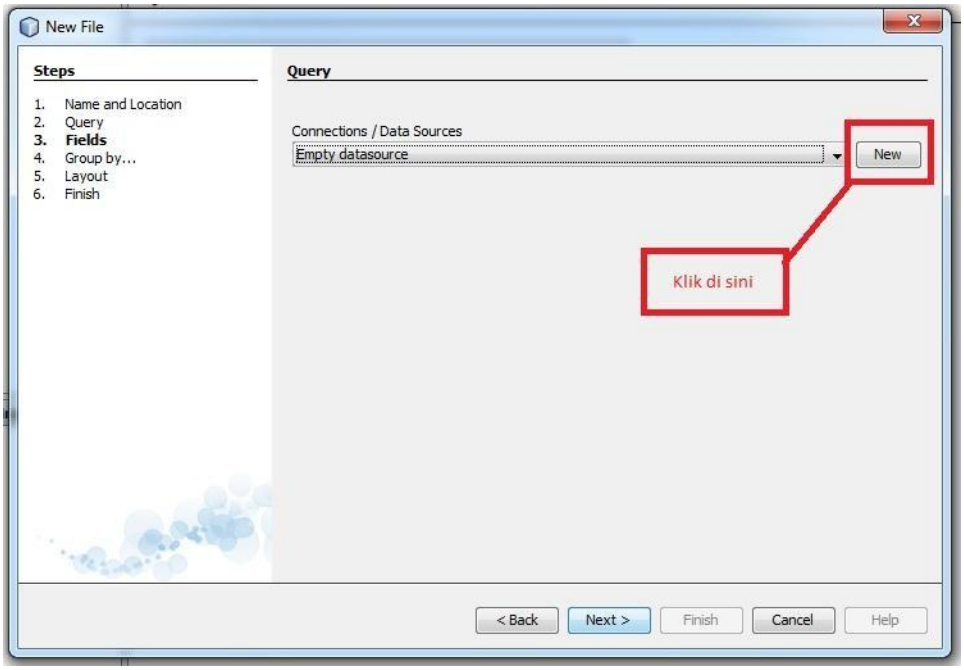
Gambar 9.2 Tahapan Report Wizard

4. Beri nama File **report_mahasiswa.jrxml**



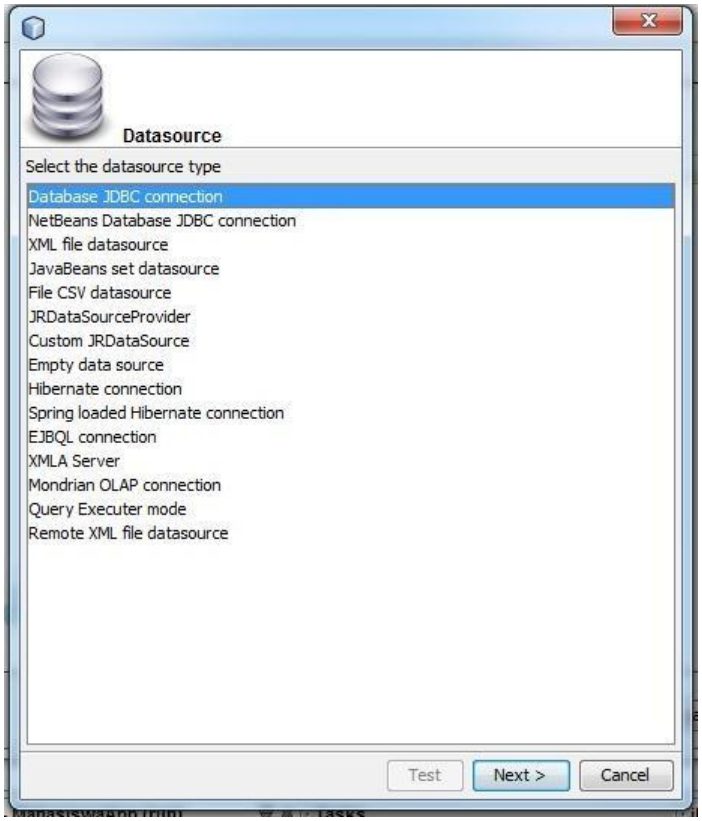
Gambar 9.3 Tampilan Jendela New Report Wizard

5. Selanjutnya koneksi ke database, pilih **New.**



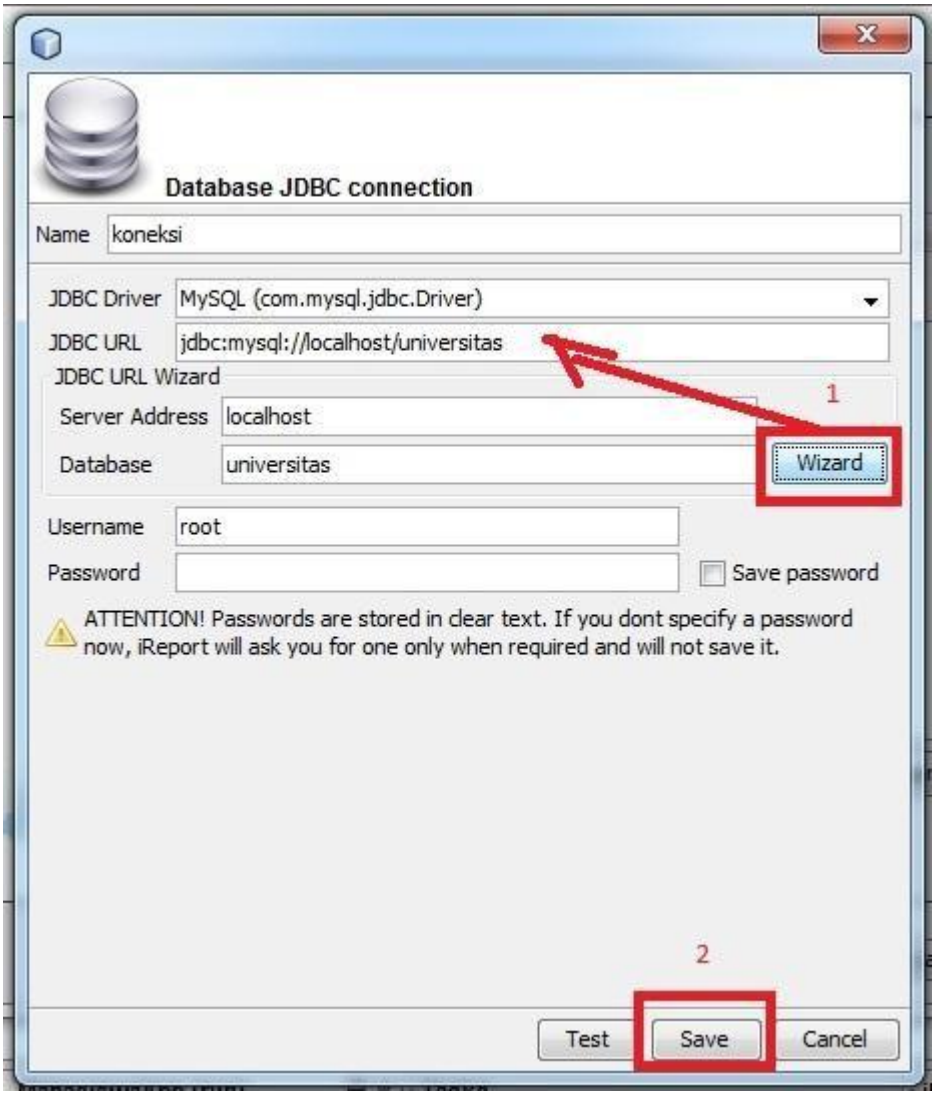
Gambar 9.4 Tampilan Jendela New File

6. Setelah itu bakalan ada pilihan datasource, krn project MahasiswaApp kita pakai MySQL jadi pilih yang Database **JDBC Connection**. Klik **Next**.



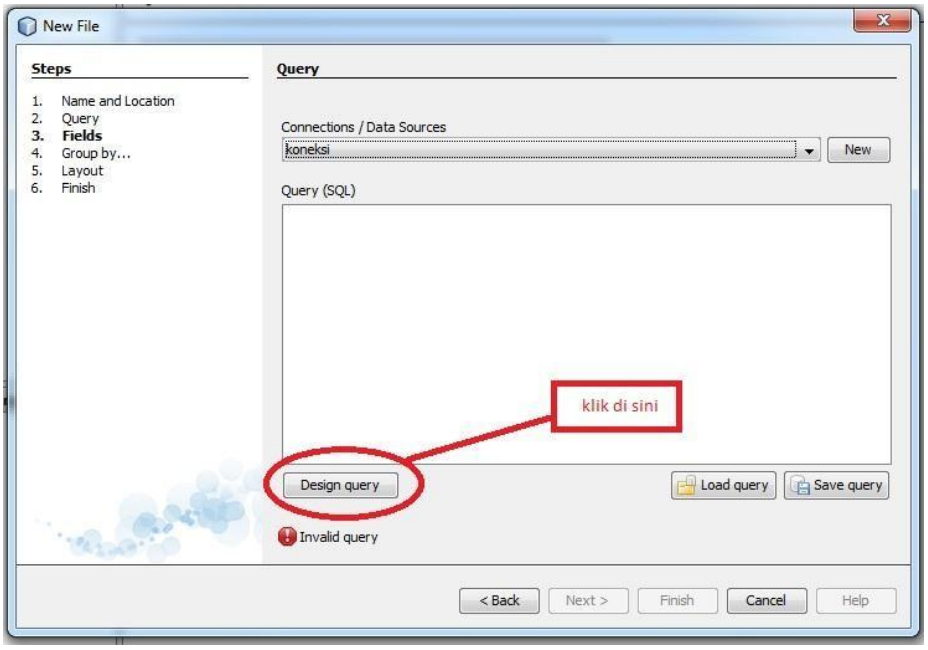
Gambar 9.5 Tampilan Jendela Datasource

7. Step selanjutnya Setting koneksi, sesuaikan dengan database universitas. Berikan **nama: koneksi , server address : localhost , database universitas , username root, password :** kosongkan, Jika sudah klik **Wizard** agar url berubah dan klik **Save**.



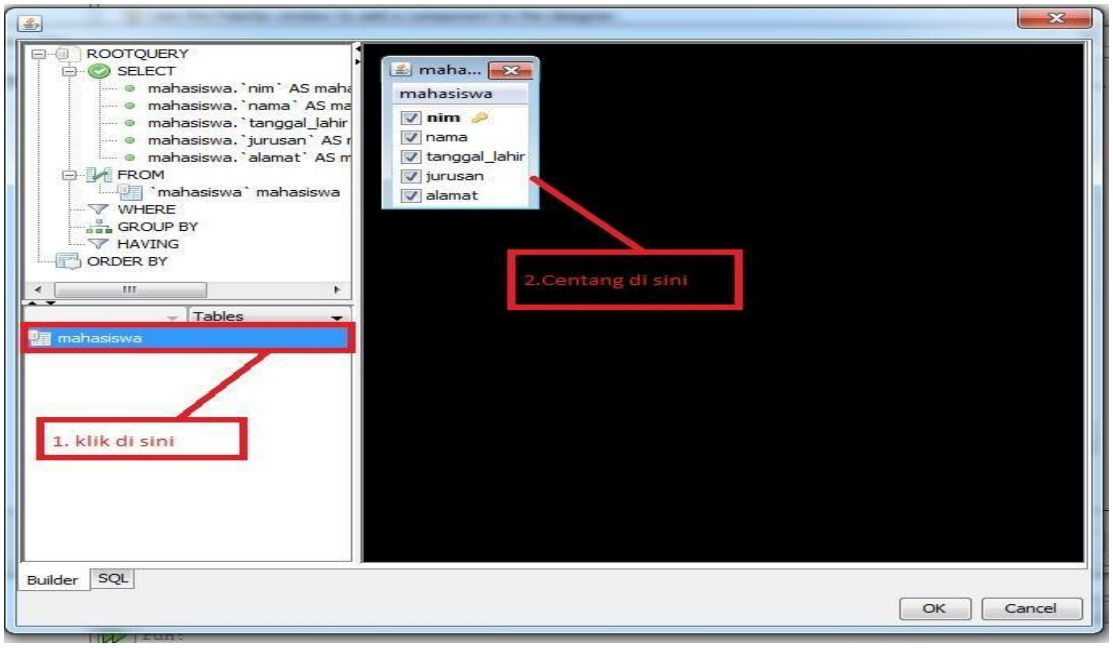
Gambar 9.6 Tampilan Jendela Database JDBC Connection

8. Setelah di Save, pilih Design Query.



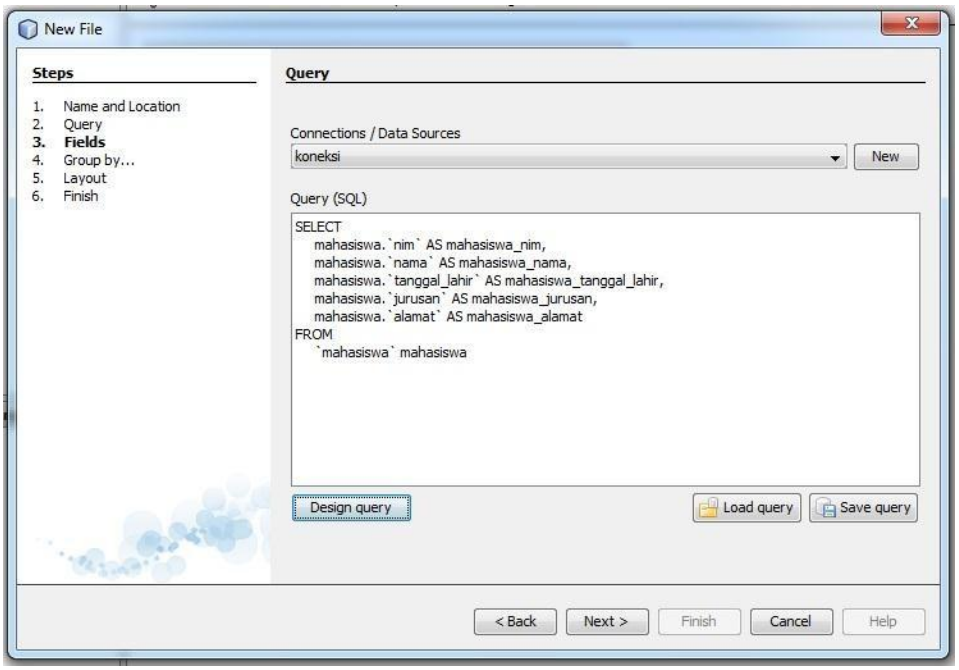
Gambar 9.7 Tampilan Jendela Query

9. Selanjutnya klik oke jika muncul kotak dialog masukan password. Karena database kita tanpa password maka kosongkan dan klik OK. Selanjutnya klik table mahasiswa dan Centang field atau atribut dari tabelyang ingin di tampilkan.



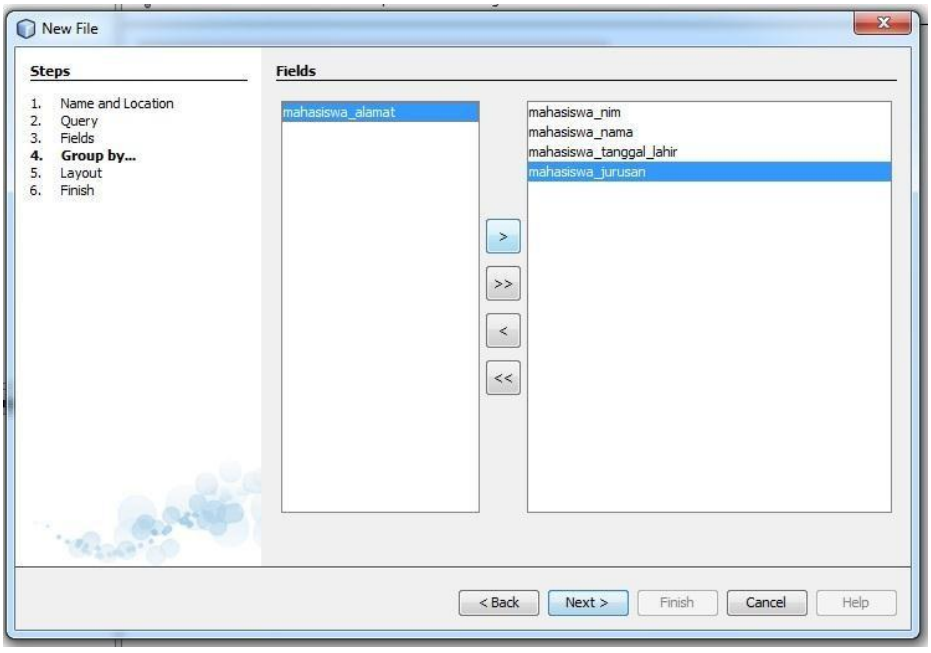
Gambar 9.8 Pemilihan Tabel

10. Setelah di klik OK. akan ada tampilan seperti dibawah. Lalu klik Next.



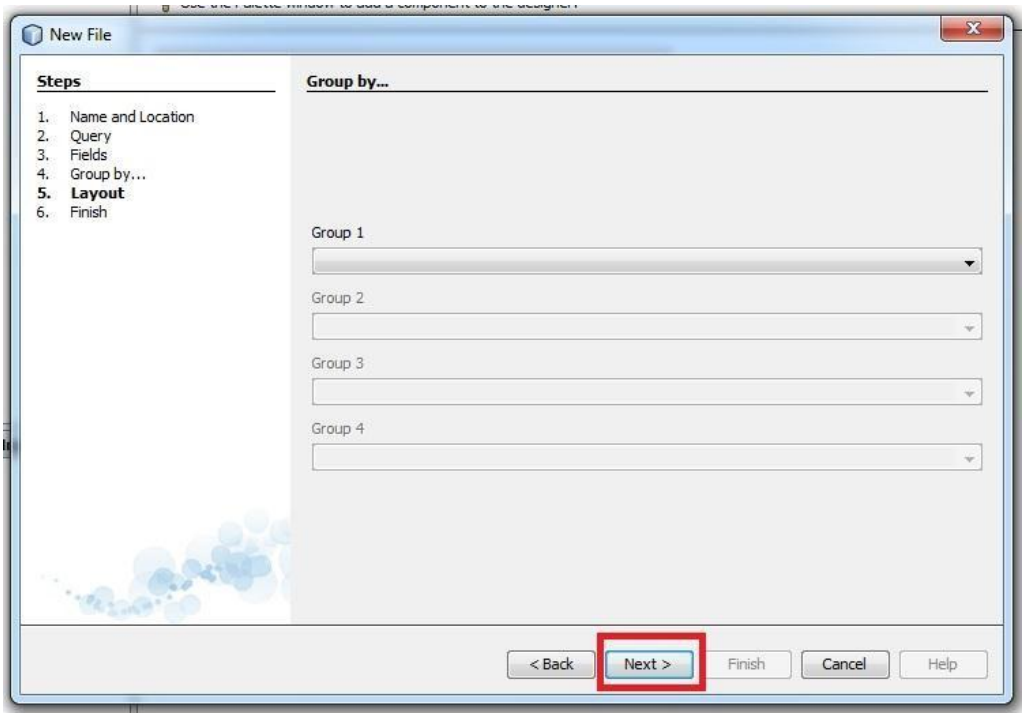
Gambar 9.9 Tampilan Jendela Query

11. Pada tab ini pilih field yang ingin ditampilkan dilaporan dan pindahkan ke sisi kanan. Lalu klikNext.



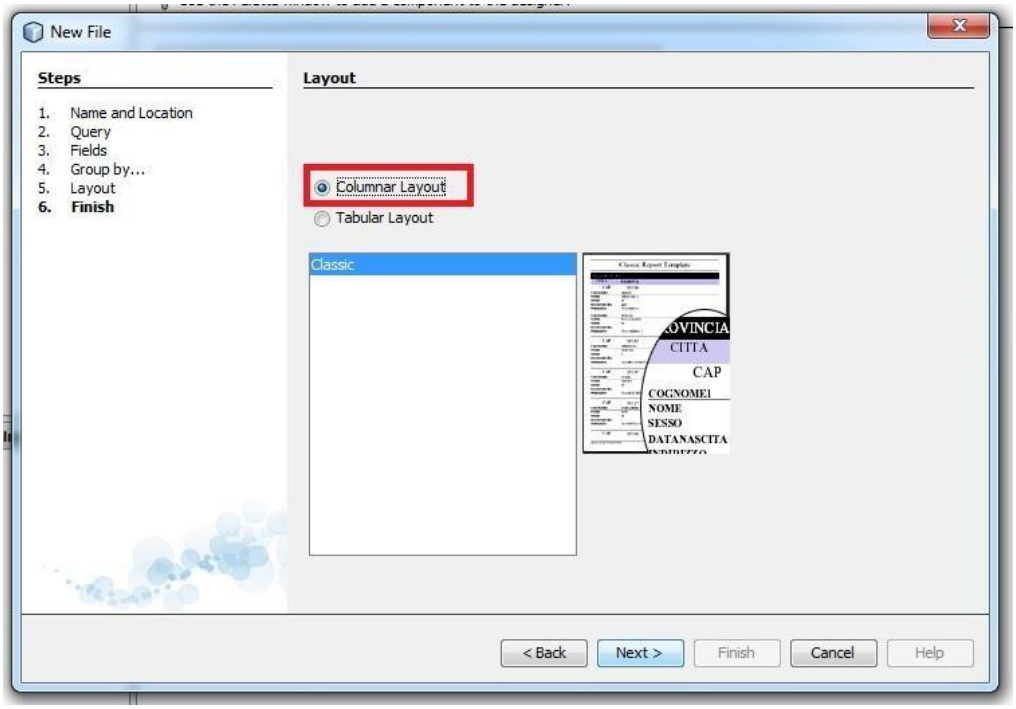
Gambar 9.10 Tampilan Jendela Fields

12. Step selanjutnya pilih berdasarkan group by.jika ingin membuat group, jika tidak klik Next, untuk saat ini kita langsung saja klik next.



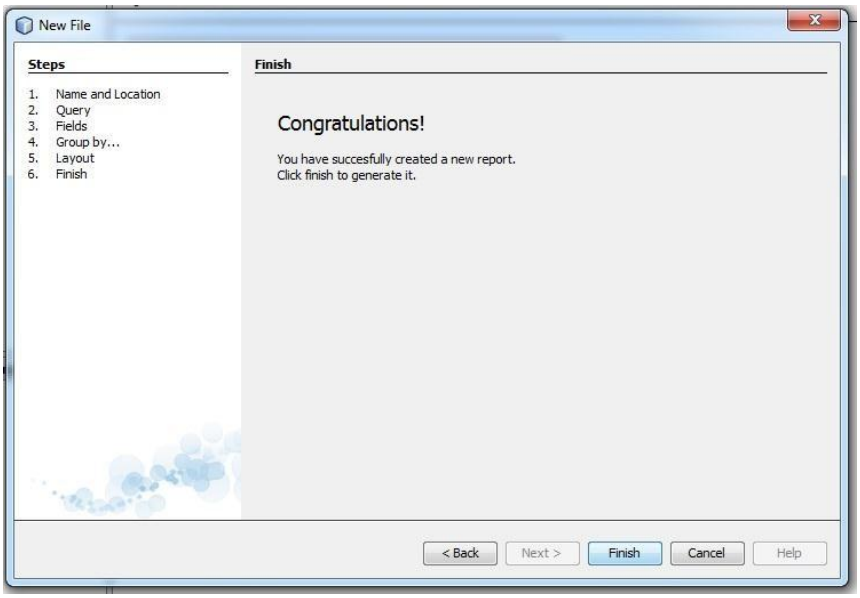
Gambar 9.11 Tampilan Jendela Group by..

13. Pilih sesuai yang anda inginkan. Untuk contoh kali ini kita gunakan columnar layout. Setelah itu klik next.



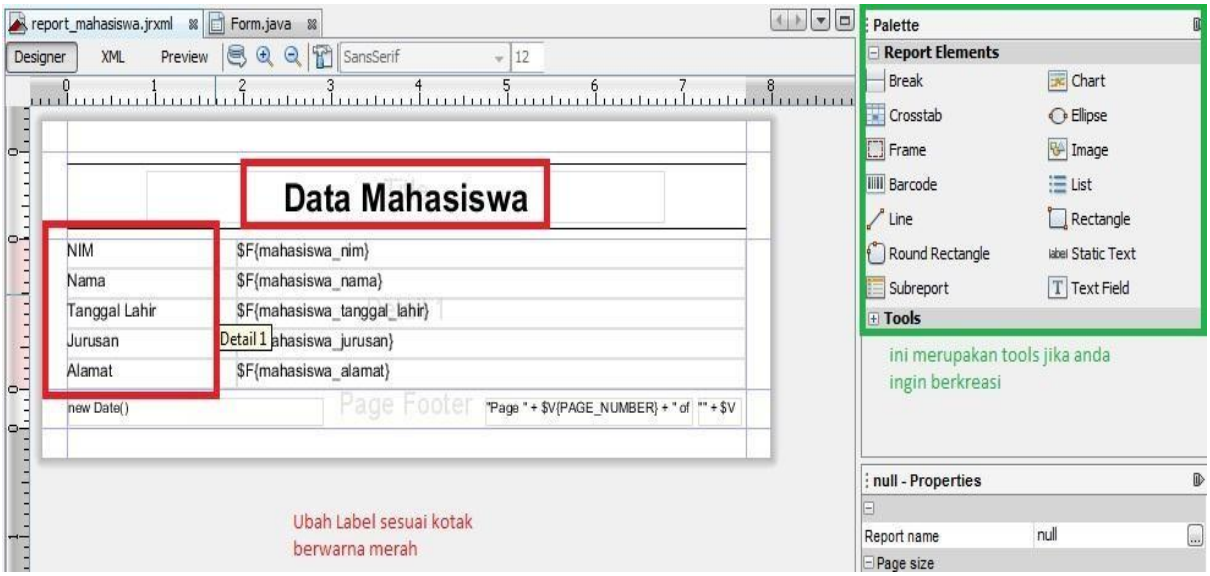
Gambar 9.12 Tampilan Jendela Layout

14. Klik finish untuk menyelesaikan settingan wizard.



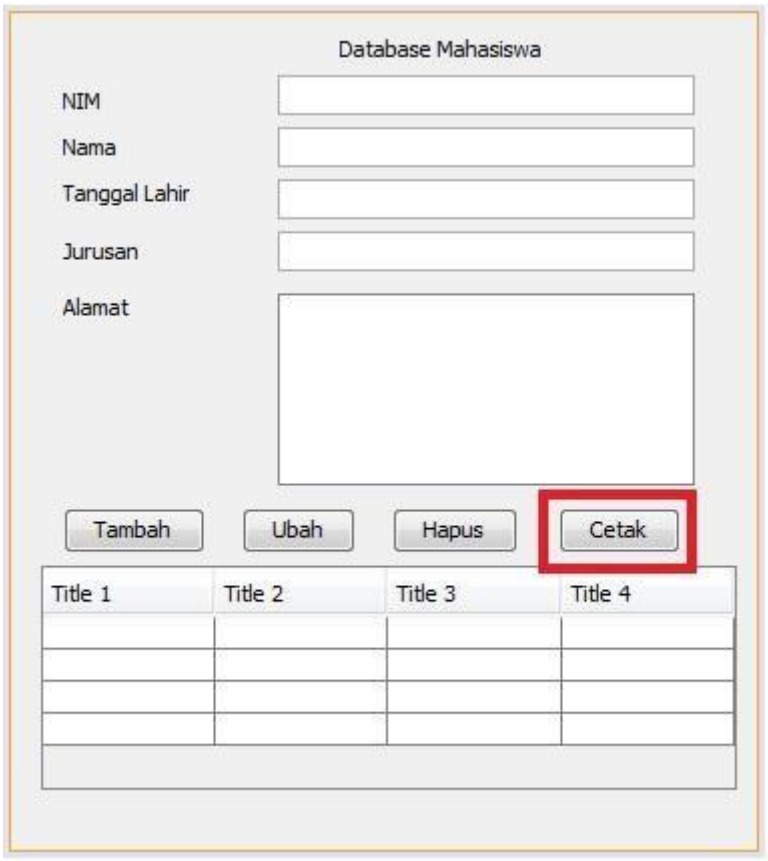
Gambar 9.12 Tampilan Jendela Finish

15. Selanjutnya atur format laporan sesuai gambar berikut. Klik 2x untuk mengubah label. Dan klik preview untuk compile report.



Gambar 9.13 Tampilan Rancangan Form

16. Selanjutnya kembali ke form GUI. Dan tambahkan Jbutton unuk mencetak data kita.

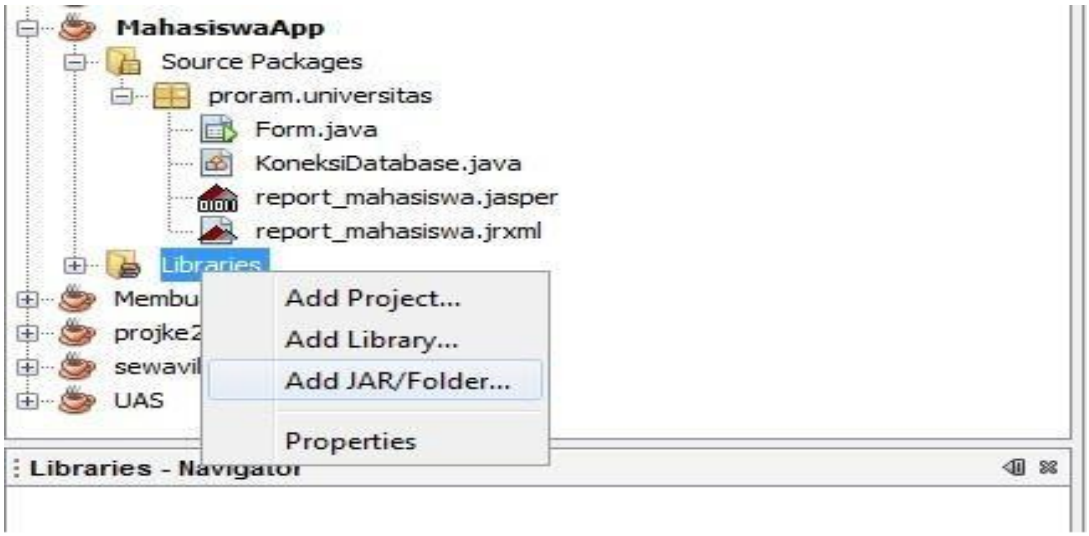


Gambar 9.14 Menambah Tombol Cetak

17. Klik 2x pada jbutton **Cetak,** kemudian tambahkan coding berikut.

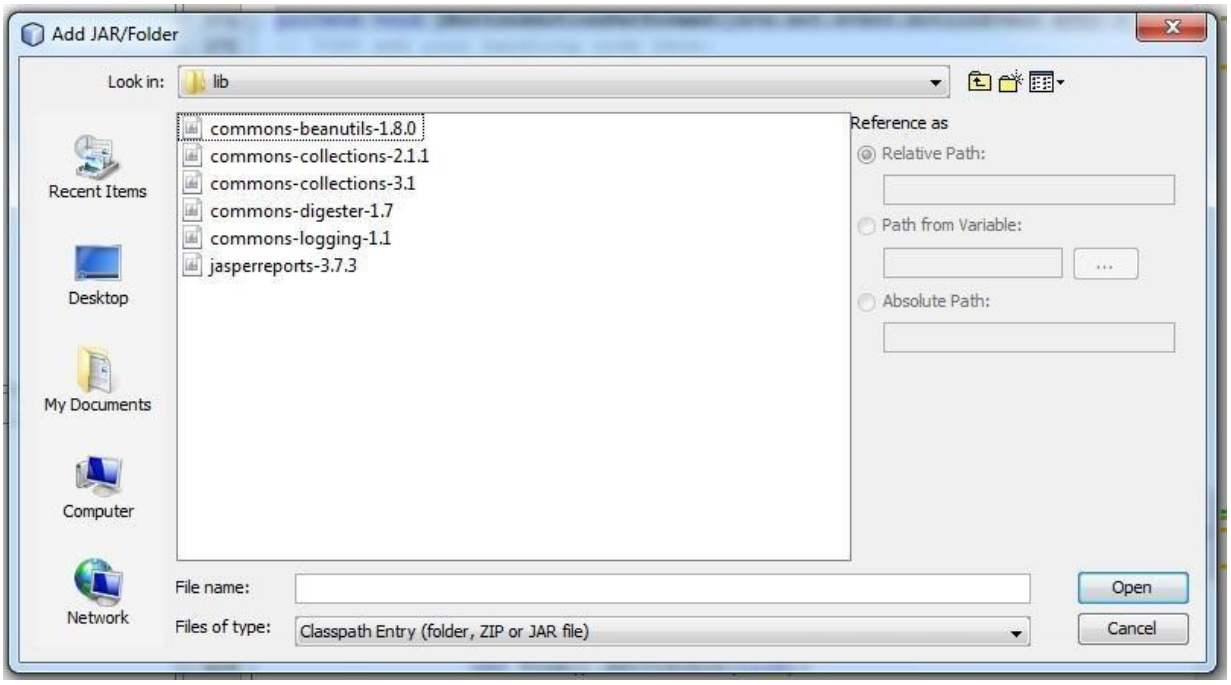
```
try {
  File file = new
  File("src/program/universitas/report_mahasiswa.jrxml ");
  jasperDesign = JRXmlLoader.load(file);
  param.clear();
  jasperReport =
  JasperCompileManager.compileReport(jasperDesign);
  jasperPrint = JasperFillManager.fillReport(jasperReport,
  param, koneksi.getConnection());
  JasperViewer.viewReport(jasperPrint, false);
} catch (Exception e) { e.printStackTrace();}
```

18. Jika sudah pasti anda akan menemukan beberapa lampu eror dari netbeans. Tapi jangan khawatir anda hanya perlu menambahkan **libraries**, **variabel** serta **import** untuk mengatasi masalah tersebut.pertama kita akan menambahkan **libraries**. Dengan cara klik kanan pada **libraries |** Pilih **Add JAR/Folder** seperti pada gambar berikut.



Gambar 9.15 Tampilan Add JAR/Folder (1)

19. Kemudian pilih semua file yang telah anda download tepatnya di dalam folder **add-jar-library** tambahkan semua file seperti pada gambar berikut.

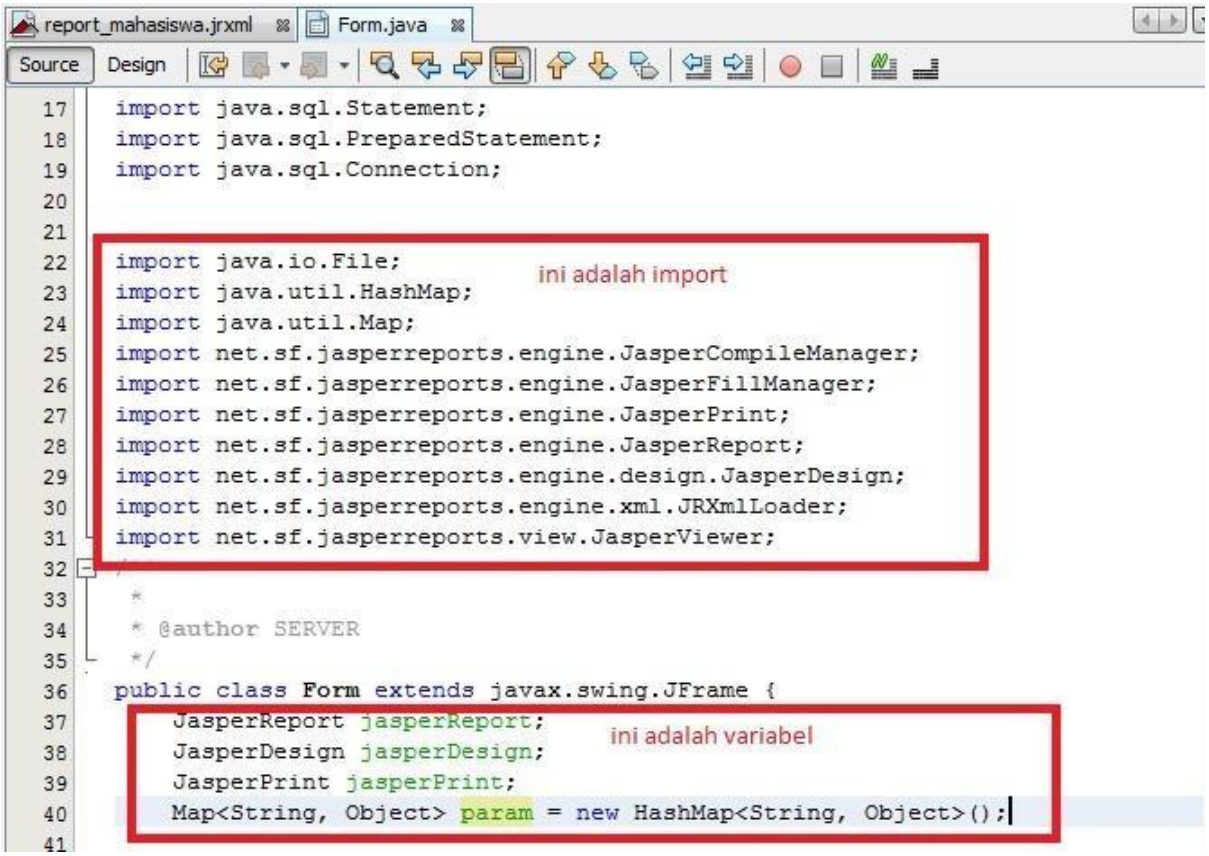


Gambar 9.16 Tampilan Add JAR/Folder (2)

20. Selanjutnya tambahkan **variabel** berikut. Ketikan syntax berikut pada program anda.tepat dibawah class form.

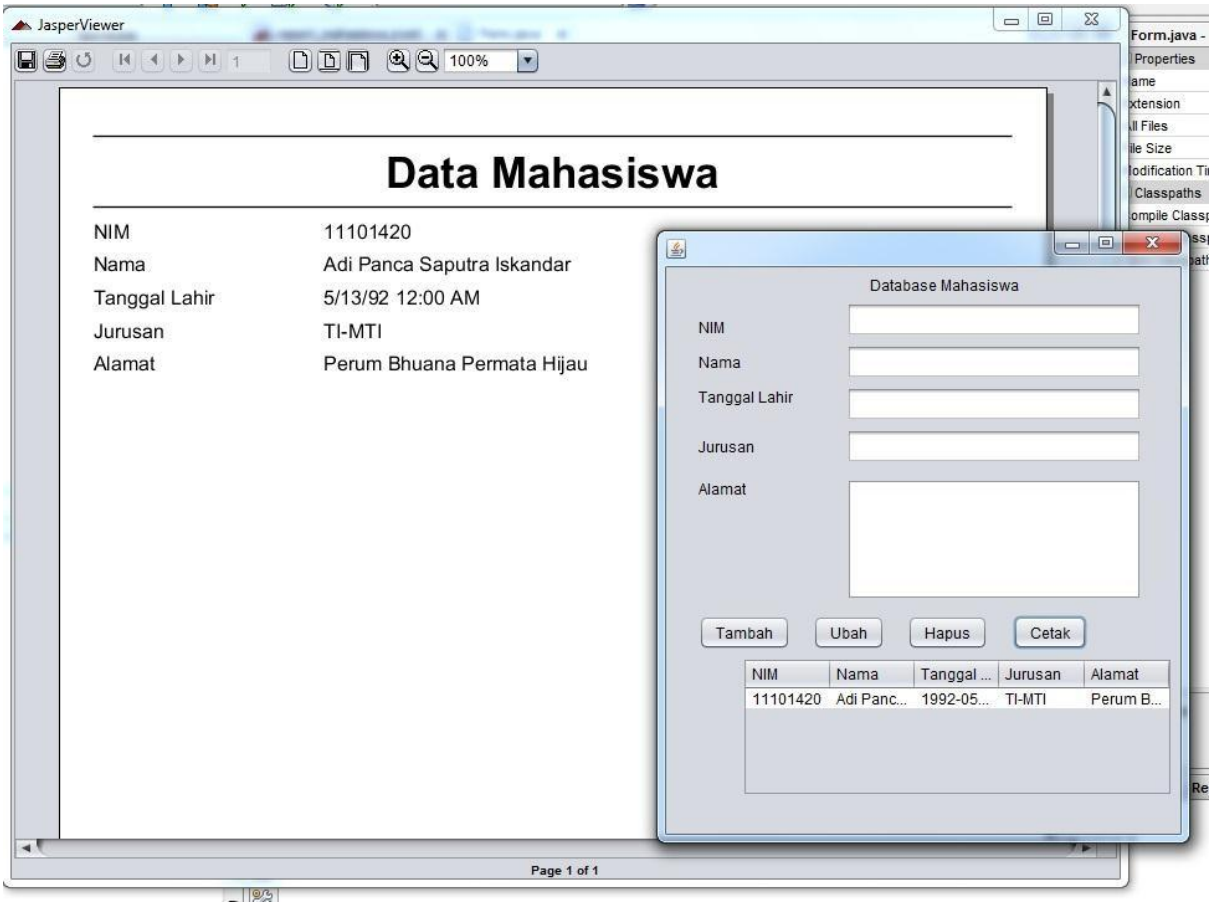
```
public class Form extends javax.swing.JFrame {
    JasperReport jasperReport;
    JasperDesign jasperDesign;
    JasperPrint jasperPrint;
    Map<String, Object> param = new HashMap<String, Object>();
```

21. Dan yang terakhir tambahkan beberapa import pada syntax anda. Ketikan syntax berikut. Tepat di bawah import sebelumnya atau di atas class form.



Gambar 9.17 Source Code

Selanjutnya anda dapat compile dan Run project MahasiswaApp anda. Dan klik tombol cetak untuk melihat hasil dari report yang telah anda buat. Tampilan akan seperti gambar berikut.



Gambar 9.18 Hasil Cetak Report

TUGAS

Implementasikan materi yang telah di dapat dalam project sistem informasi yang telah dibuat