MODUL PRAKTIKUM
PEMROGRAMAN BERBASIS OBYEK

dengan JAVA

PROGRAM STUDI D-IV KOMPUTASI STATISTIK

POLITEKNIK STATISTIKA STIS

2021

MODUL PRAKTIKUM PEMROGRAMAN BERBASIS OBYEK
dengan JAVA

Disusun oleh :
Ibnu, SST, MT
Lutfi Rahmatuti Maghfiroh, SST, MT

Hak Cipta 2021 pada penulis.

Edisi Pertama,
Cetakan Pertama: 2021

Penerbit:
Politeknik Statistika STIS
Jl. Otto Iskandardinata No. 64C Jakarta Timur 13330
Telpon. (021) 8508812, 8191437,
Facs (021) 8197577

Hak cipta dilindungi Undang-Undang. Dilarang memperbanyak sebagian atau seluruh isi bahan ajar ini dalam bentuk apa pun, baik secara elektronik maupun mekanik, termasuk mempotokopi, merekap, atau menggunakan sistem penyimpanan lainnya, tanpa izin tertulis dari Penerbit.

UNDANG-UNDANG NOMOR 19 TAHUN 2002 TENTANG HAK CIPTA
1. *Barang siapa dengan sengaja dan tanpa hak mengumukan atau memperbanyak suatu ciptaan atau member izin untuk itu, dipidana dengan penjara paling lama 7 (tujuh) tahun dan atau denda paling banyak Rp. 5.000.000.0000,00 (lima miliar rupiah)* 2. *Barang siapa dengan sengaja menyiarkan, memamerkan, mengedarkan atau menjual kepada umum suatu ciptaan atau barang hasil pelanggaran Hak Cipta atau Hak Terkait sebagaimana dimaksud pada ayat (1), dipidana dengan pidana penjara paling lama 5 (lima) tahun dan atau denda paling banyak Rp. 5.000.000.000,00 (lima miliar rupiah)*

KATA PENGANTAR

Puji syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa karena dengan rahmat, karunia, serta taufik dan hidayah-Nya kami dapat menyelesaikan modul Praktikum Dasar-Dasar Pemrograman ini dengan baik meskipun banyak kekurangan di dalamnya. Kami juga mengucapkan terima kasih yang sebesar-besarnya kepada semua pihak yang telah membantu penyusunan modul ini.

Kami sangat berharap modul ini dapat berguna dalam mengantarkan mahasiswa Program D-IV Komputasi Statistik Politeknik Statistika STIS dalam memahami dan menerapkan mata kuliah Komputasi Statistik. Kami juga menyadari sepenuhnya bahwa modul ini terdapat kekurangan dan jauh dari kata sempurna. Oleh sebab itu, kami berharap adanya kritik, saran dan usulan demi perbaikan modul yang telah kami buat di masa yang akan datang, mengingat tidak ada sesuatu yang sempurna tanpa saran yang membangun.

Akhir kata, semoga modul ini dapat bermanfaat dan dipahami bagi siapapun yang membacanya. Tak lupa kami mohon maaf apabila terdapat kesalahan kata-kata yang kurang berkenan dan kami memohon kritik dan saran yang membangun dari pembaca demi perbaikan modul ini di waktu yang akan datang. Terima kasih.

Jakarta, November 2021

Tim Penulis

DAFTAR ISI

KATA PENGANTAR iii
DAFTAR ISI v
MODUL 1 PENGENALAN PEMROGRAMAN JAVA 1
1.1. Deskripsi Singkat 1
1.2. Tujuan Praktikum 1
1.3. Material Praktikum 1
1.4. Kegiatan Praktikum 1
A. Instalasi Java 1
a. Unduh dan Install Java 1
b. Update PATH Environment Variable 3
B. Program Java Pertama 6
C. Compile 8
D. Running 9
1.5. Penugasan 10
MODUL 2 PENGENALAN *INTEGRATED DEVELOPMENT ENVIRONMENT* (IDE) PEMROGRAMAN JAVA 11
2.1. Deskripsi Singkat 11
2.2. Tujuan Praktikum 11
2.3. Material Praktikum 11
2.4. Kegiatan Praktikum 11
A. Unduh dan Install 11
a. Unduh IDE Netbeans 11
b. Install 13
B. Pemrograman Java di Netbeans 17
C. Objek vs Class 24
2.5. Penugasan 29
MODUL 3 INHERITANCE, POLYMORPHISM, AGGREGATION DAN COMPOSITION 30
3.1. Deskripsi Singkat 30
3.2. Tujuan Praktikum 30
3.3. Material Praktikum 30
3.4. Kegiatan Praktikum 30
A. INHERITANCE 30
B. POLYMORPHISM 34

C. AGGREGATION 35
D. COMPOSITION 36
3.5. Penugasan 37
MODUL 4 ABSTRACT CLASS DAN INTERFACE 38
4.1. Deskripsi Singkat.....38
4.2. Tujuan Praktikum 38
4.3. Material Praktikum.....38
4.4. Kegiatan Praktikum 38
A. ABSTRACT CLASS 38
B. INTERFACE.....39
4.5. Penugasan 41
MODUL 5 KONVERSI DIAGRAM KELAS KE KODE 42
5.1. Deskripsi Singkat.....42
5.2. Tujuan Praktikum 42
5.3. Material Praktikum.....42
5.4. Kegiatan Praktikum 42
5.5. Penugasan 43
MODUL 6 PENANGANAN PENGECUALIAN 44
6.1. Deskripsi Singkat.....44
6.2. Tujuan Praktikum 44
6.3. Material Praktikum.....44
6.4. Kegiatan Praktikum 44
6.5. Penugasan 47
MODUL 7 DESIGN PATTERN 48
7.1. Deskripsi Singkat.....48
7.2. Tujuan Praktikum 48
7.3. Material Praktikum.....48
7.4. Kegiatan Praktikum 48
A. Observer Pattern 48
B. Decorator Pattern.....51
C. Factory Pattern.....54
D. Singleton Pattern.....56
7.5. Penugasan 56
MODUL 8 GRAPHICAL USER INTERFACE BAGIAN 1 57
8.1. Deskripsi Singkat.....57

8.2. Tujuan Praktikum 57
8.3. Material Praktikum.....57
8.4. Kegiatan Praktikum 57
8.5. Penugasan 67
MODUL 9 GRAPHICAL USER INTERFACE BAGIAN 2 68
9.1. Deskripsi Singkat 68
9.2. Tujuan Praktikum 68
9.3. Material Praktikum.....68
9.4. Kegiatan Praktikum 68
9.5. Penugasan 91
MODUL 10 INPUT-OUTPUT 92
10.1. Deskripsi Singkat.....92
10.2. Tujuan Praktikum 92
10.3. Material Praktikum.....92
10.4. Kegiatan Praktikum 92
A. MEMBACA FILE.....92
B. MENULIS FILE 96
C. GUI 97
D. MEMBACA OBJECT 100
10.5. Penugasan 104
MODUL 11 DATABASE ACCESS (JDBC).....105
11.1. Deskripsi Singkat.....105
11.2. Tujuan Praktikum 105
11.3. Material Praktikum.....105
11.4. Kegiatan Praktikum 105
11.5. Penugasan 112
MODUL 12 CONCURRENCY (THREADING).....113
12.1. Deskripsi Singkat.....113
12.2. Tujuan Praktikum 113
12.3. Material Praktikum.....113
12.4. Kegiatan Praktikum 113
12.5. Penugasan 118
MODUL 13 NETWORKING BAGIAN 1.....119
13.1. Deskripsi Singkat.....119
13.2. Tujuan Praktikum 119

13.3. Material Praktikum 119
13.4. Kegiatan Praktikum 119
13.5. Penugasan 125
MODUL 14 NETWORKING BAGIAN 2 126
14.1. Deskripsi Singkat 126
14.2. Tujuan Praktikum 126
14.3. Material Praktikum 126
14.4. Kegiatan Praktikum 126
14.5. Penugasan 135

MODUL 1 PENGENALAN PEMROGRAMAN JAVA

1.1. Deskripsi Singkat

Untuk mempelajari pemrograman, kita akan mempelajari terlebih dahulu dengan bahasa dan software yang akan kita pakai dalam pemrograman ini. Bahasa pemrograman yang akan digunakan adalah JAVA. Oleh karena itu pada praktikum pertama ini mahasiswa akan berkenalan software yang dipakai dan dengan JAVA itu sendiri.

1.2. Tujuan Praktikum

Setelah praktikum pada modul 1 ini diharapkan mahasiswa mempunyai kompetensi memahami software pemrograman Java dan bahasa pemrograman Java. Selain itu mahasiswa juga diharapkan memahami cara melakukan compile dan running pada program yang sudah dibuat.

1.3. Material Praktikum

Kegiatan pada modul 1 ini memerlukan material berupa software pemrograman JAVA dan Editor, baik itu Notepad, Notepad++, ataupun yang lainnya

1.4. Kegiatan Praktikum

Dalam materi ini akan dijelaskan mengenai data, tabel, entri data ke file lembar kerja dalam bentuk tabel, membuat agregasi. Buatlah file dengan nama file *<<nim>>_modul1.*

A. Instalasi Java

a. Unduh dan Install Java[1]

Disarankan, sebelum Anda melanjutkan instalasi online, Anda mungkin ingin menonaktifkan firewall Internet Anda. Dalam beberapa kasus, pengaturan firewall default diatur untuk menolak semua penginstalan otomatis atau online seperti penginstalan online Java. Jika firewall tidak dikonfigurasi dengan benar, itu dapat menghentikan operasi unduh/instal Java dalam kondisi tertentu. Lihat manual firewall Internet spesifik Anda untuk petunjuk tentang cara menonaktifkan Firewall Internet Anda.

1) Buka halaman unduhan Manual
2) Klik pada Windows Online

[1] https://www.java.com/en/download/help/windows_manual_download.html

3) Kotak dialog Unduh File muncul meminta Anda untuk menjalankan atau menyimpan file unduhan
    a. Untuk menjalankan penginstal, klik Jalankan.
    b. Untuk menyimpan file untuk instalasi nanti, klik Simpan.
    c. Pilih lokasi folder dan simpan file ke sistem lokal Anda.
    d. Tips: Simpan file ke lokasi yang diketahui di komputer Anda, misalnya, ke desktop Anda.
    e. Klik dua kali pada file yang disimpan untuk memulai proses instalasi.
4) Proses instalasi dimulai. Klik tombol Instal untuk menerima persyaratan lisensi dan melanjutkan penginstalan.

5) Oracle telah bermitra dengan perusahaan yang menawarkan berbagai produk. Pemasang mungkin memberi Anda opsi untuk menginstal program ini saat Anda menginstal Java. Setelah memastikan bahwa program yang diinginkan telah dipilih, klik tombol Next untuk melanjutkan instalasi.
6) Beberapa dialog singkat mengkonfirmasi langkah terakhir dari proses instalasi; klik Tutup pada dialog terakhir. Ini akan menyelesaikan proses instalasi Java.

a. Deteksi versi yang lebih lama (8u20 dan versi yang lebih baru). Dimulai dengan Java 8 Update 20 (8u20), pada sistem Windows, Java Uninstall Tool terintegrasi dengan penginstal untuk memberikan opsi untuk menghapus versi Java yang lebih lama dari sistem. Perubahan ini berlaku untuk platform Windows 32 bit dan 64 bit.

7) Pemberitahuan tentang Java yang dinonaktifkan dan petunjuk pemulihan
8) Penginstal memberi tahu Anda jika konten Java dinonaktifkan di browser web, dan memberikan instruksi untuk mengaktifkannya. Jika sebelumnya Anda memilih untuk menyembunyikan beberapa perintah keamanan untuk applet dan aplikasi Java Web Start, penginstal menyediakan opsi untuk memulihkan perintah tersebut. Pemasang mungkin meminta Anda untuk me-reboot komputer Anda jika Anda memilih untuk tidak me-restart browser internet ketika diminta untuk melakukannya.
9) Instalasi Uji
   a. Untuk menguji apakah Java telah terinstal dan berfungsi dengan baik di komputer Anda, jalankan applet pengujian ini.
   b. CATATAN: Anda mungkin perlu memulai ulang (menutup dan membuka kembali) browser Anda untuk mengaktifkan instalasi Java di browser Anda.

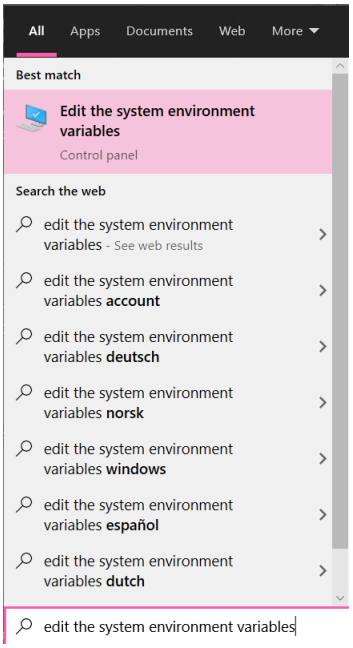
b. Update PATH Environment Variable

Untuk selanjutnya kita atur Path secara manual agar program Java kita bisa di-compile dan di-run menggunakan Command Prompt atau cmd.

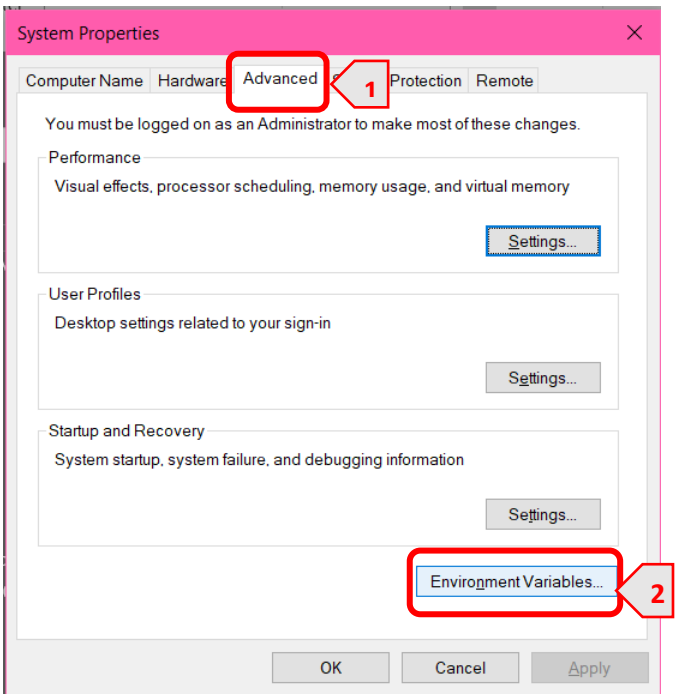
1) Buka folder instalasi Java dan carilah folder Bin, salin alamat folder tersebut untuk digunakan pada tahap selanjutnya.

```
C:\Program Files\Java\jdk1.8.0_112\bin.
```

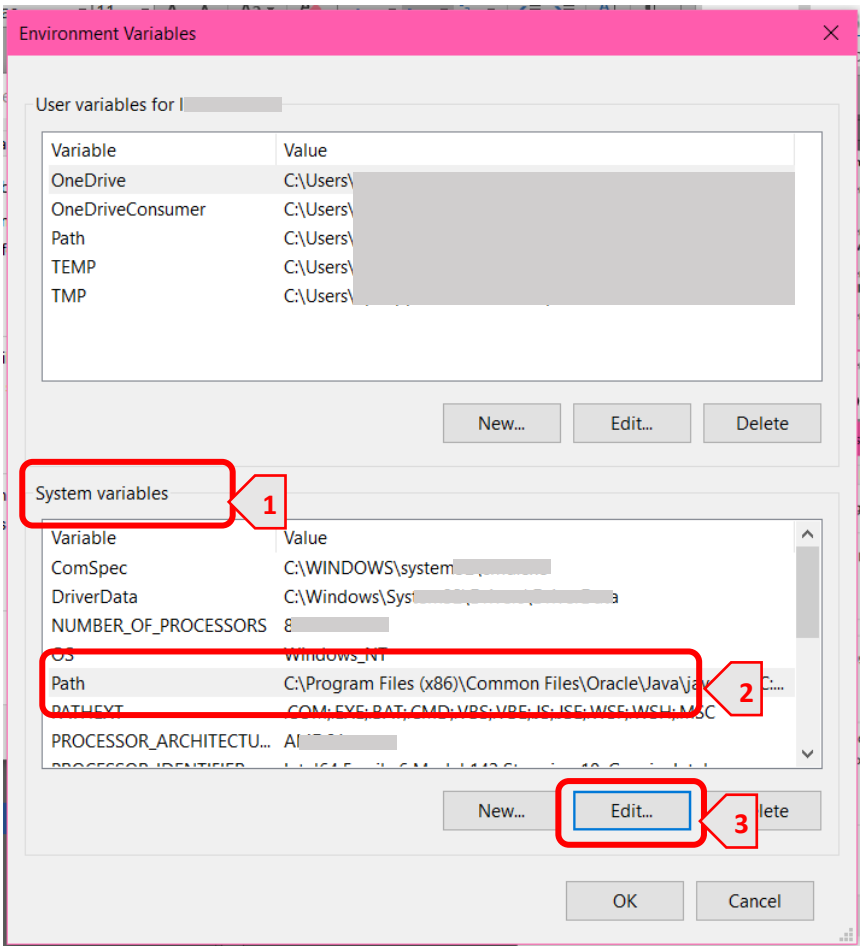
2) Klik Start dan ketikkan system environment, kemudian pilih Edit the system environment variables.



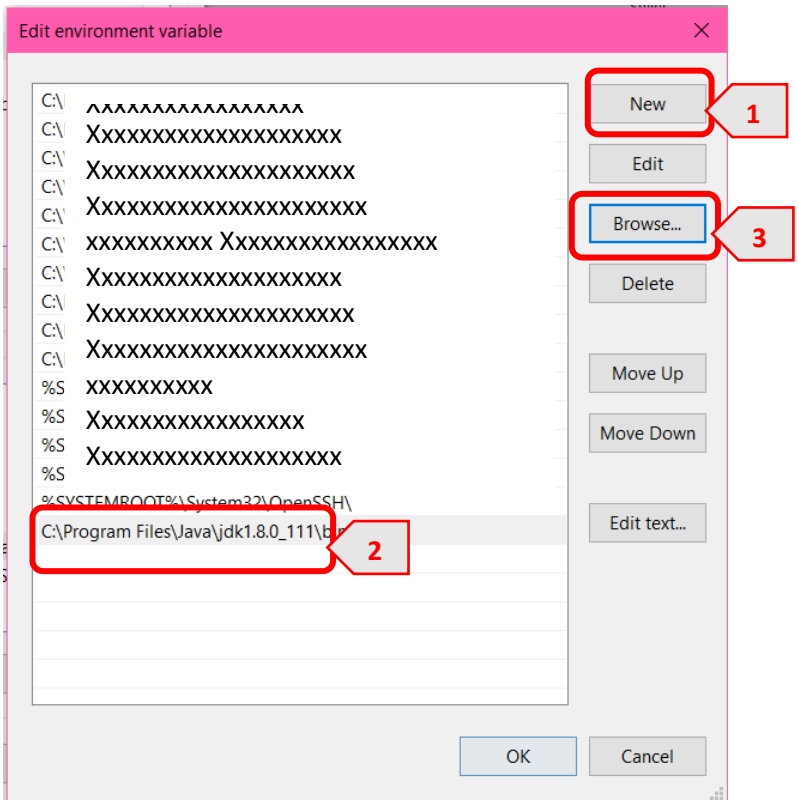
3) Kemudian pada tab Advance, klik Environment Variables.



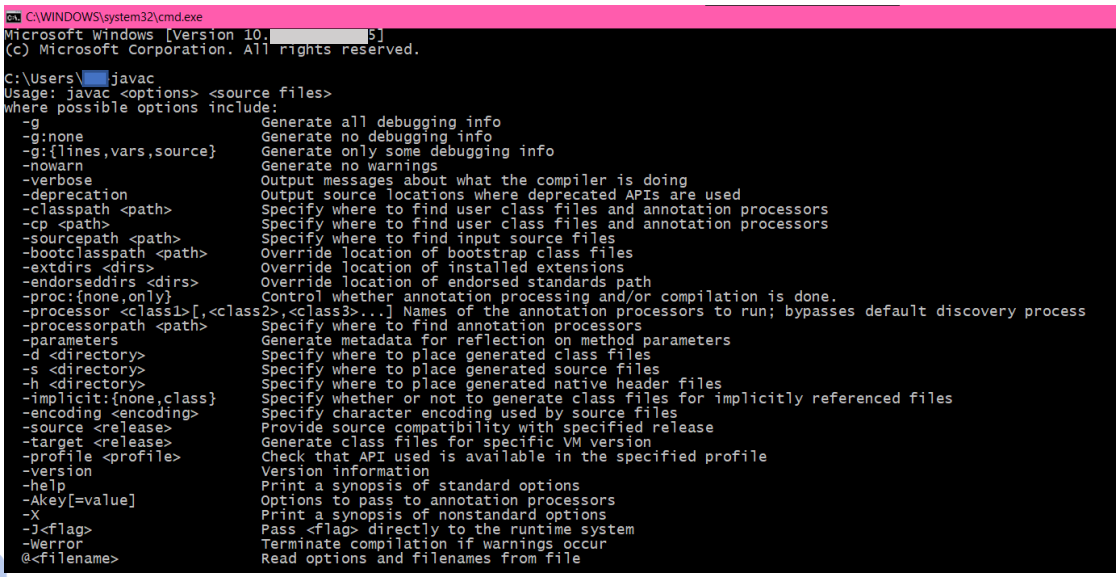
4) Pada System Variables, cari variable Path, kemudian klik Edit.



5) Klik New, kemudian tempel alamat folder yang tadi telah disalin atau klik browse ke folder lokasi install.(C:\Program Files\Java\jdk1.8.0_111\bin).



6) Setelah selesai, klik `OK` untuk semua tab yang masih terbuka, jangan langsung di `close`.
7) Untuk mengecek keberhasilan instalasi dan setting `Path` kita, buka `cmd` dengan cara `Win + R > cmd`. Setelah Command Prompt terbuka, ketikkan `javac` atau `java -version`, lalu `enter`. Berikut adalah gambar jika proses instalasi berhasil:



B. Program Java Pertama

Pertama kita belajar program sederhana yang menampilkan `Hello, World!` di layar. Karena ini adalah program yang sangat sederhana, ini sering digunakan untuk memperkenalkan bahasa pemrograman baru kepada seorang pemula. Buka Text Editor, baik itu Notepad, Notepad++, ataupun yang lainnya. Kemudian, ketikkan program di bawah ini, lalu save dengan judul HaloDunia.java.

```
1 // Program JAVA pertamaku
2
3 class HaloDunia {
4     public static void main(String[] args) {
5         System.out.println("Halo, Dunia!");
6     }
7 }
```

Mari kita jelajahi bagaimana Java "Halo, Dunia!" program bekerja.

Di Java, setiap baris yang dimulai dengan // adalah komentar. Komentar ditujukan bagi pengguna yang membaca kode untuk memahami maksud dan fungsi program. Ini benar-benar diabaikan oleh kompiler Java (aplikasi yang menerjemahkan program Java ke bytecode Java yang dapat dijalankan oleh komputer). Untuk mempelajari lebih lanjut, kunjungi komentar Java.

```
// Program Pertama Anda
```

Di Java, setiap aplikasi dimulai dengan definisi kelas. Dalam program, HelloWorld adalah nama kelas, dan definisi kelas adalah:

```
class HaloDunia { ... }
```

Untuk saat ini, ingat saja bahwa setiap aplikasi Java memiliki definisi kelas, dan nama kelas harus sesuai dengan nama file di Java.

Ini adalah metode utama. Setiap aplikasi di Java harus berisi metode utama. Kompiler Java mulai mengeksekusi kode dari metode utama.

Untuk saat ini, ingatlah bahwa fungsi utama adalah titik masuk aplikasi Java Anda, dan itu wajib dalam program Java. Tanda tangan dari metode utama di Jawa adalah:

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

Kode di berikutnya adalah pernyataan cetak.

```
System.out.println("Halo, Dunia!");
```

Itu mencetak teks Halo, Dunia! ke output standar (layar Anda). Teks di dalam tanda kutip disebut String di Java. Perhatikan pernyataan print berada di dalam fungsi utama, yang ada di dalam definisi kelas.

Setiap Aplikasi Java yang valid harus memiliki definisi kelas (yang cocok dengan nama file).

1) Metode utama harus berada di dalam definisi kelas.
2) Kompiler mengeksekusi kode mulai dari fungsi utama.
3) Ini adalah program Java yang valid yang tidak melakukan apa-apa.

```
1 // Your First Program
2
3 class HelloWorld {
4     public static void main(String[] args) {
5         // Tulis kode di sini
6     }
7 }
```

C. Compile

Setelah membuat program Java sederhana, sekarang waktunya meng-compile program tersebut dengan menggunakan cmd. Buka cmd, kemudian arahkan ke direktori tempat menyimpan file Java. Untuk kali ini, saya menyimpannya di D:\PBOcode. Ketika cmd dibuka, secara default akan menampilkan direktori C:\Users\. Jika file java kita disimpan di drive C:\, kita cukup ketikkan perintah di bawah ini, kemudian Enter:

```
cd C:\namafolder
```

Misalkan:

```
cd C:\Belajar Java
```

Tetapi jika menggunakan drive lain, seperti yang saya lakukan, kita harus mengetikkan perintah d: (karena saya menyimpannya di drive D:) terlebih dahulu untuk memindahkan direktori aktif yang ingin digunakan, kemudian klik **Enter**. Berikut contohnya:

```
C:\Users>d:
D:\>cd PBOcode
D:\PBOcode >
```

Sekarang saya asumsikan kita telah berhasil mengarahkan folder kita ke tempat file Java kita disimpan, baik itu di drive D:\ ataupun C:\, untuk memastikannya, ketikkan perintah dir, lalu **Enter**. Berikut contohnya:

```
D:\PBOcode>dir
 Volume in drive D is
 Volume Serial Number is

 Directory of D:\PBOcode

09/06/2021  10:05 AM    <DIR>          .
09/06/2021  10:05 AM    <DIR>          ..
09/06/2021  09:21 AM               146 HaloDunia.java
               1 File(s)            146 bytes
               2 Dir(s)  89,674,502,144 bytes free

D:\PBOcode>
```

Jika sudah melihat file Java kita, sekarang kita sudah siap untuk meng-*compile*. Ketikkan perintah di bawah ini, kemudian klik **Enter**:

```
javac HaloDunia.java
```

Jika tidak terdapat notifikasi apapun, itu artinya program kita tidak mengalami error. Kecuali, jika program kita error, kita akan mendapatkan notifikasi error yang harus diperbaiki pada program Java kita.

Jika tidak terdapat error, sekarang compiler akan men-*generate* sebuah file `HaloDunia.class`. Ketikkan perintah `dir` untuk melihat apakah file `HaloDunia.class` sudah berhasil terbuat. Berikut contohnya:

```
 Directory of D:\PBOcode

09/06/2021  10:07 AM    <DIR>          .
09/06/2021  10:07 AM    <DIR>          ..
09/06/2021  10:07 AM               424 HaloDunia.class
09/06/2021  10:07 AM               148 HaloDunia.java
               2 File(s)            572 bytes
               2 Dir(s)  89,674,502,144 bytes free

D:\PBOcode>
```

Sekarang kita telah mempunyai file `.class` dari program `HaloDunia.java`. Itu artinya, sekarang kita sudah bisa me-*running* program `HaloDunia.java`.

D. Running

Untuk melakukan running, pada direktori yang sama ketikkan perintah di bawah ini, kemudian Enter:

```
java HaloDunia
```

Atau;

```
java -cp . HaloDunia
```

Dan program pun berhasil dijalankan.

```
D:\PBOcode>java HaloDunia
Halo, Dunia!
```

```
D:\PBOcode>java -cp . HaloDunia
Halo, Dunia!
```

1.5. Penugasan

1) Buat program baru dengan nama HaloDunia2.java
2) Ubah baris kode dengan:

```java
// Program JAVA pertamaku

class HaloDunia2 {
    public static void main(String[] args) {
        System.out.println("Halo, Dunia! dengan argumen:");
        for(int i=0;i<args.length;i++)
            System.out.println(args[i]);
        }
    }
```

3) Compile program tersebut
4) Jalankan dengan tambahan argumen <<nim kalian>> dan <<nama kalian>>

```
D:\PBOcode>java HaloDunia2 <<nim>> <<nama>>
```

5) Bagaimana kah hasilnya?

Laporkan hasil praktikum berikut dengan hasil penugasan dan penjelasannya ke Dosen dalam bentuk file pdf dengan format nama *<<nim>>_modul1.*

1) Tangkapan layar hasil akhir instalasi dengan `javac` atau `java -version`
2) Tangkapan layar hasil compile
3) Tangkapan layar hasil running

MODUL 2 PENGENALAN *INTEGRATED DEVELOPMENT ENVIRONMENT* (IDE) PEMROGRAMAN JAVA

2.1. Deskripsi Singkat

Dalam praktek pemrograman, untuk mempermudah pembangunan program, kita dapat menggunakan suatu software terintegrasi yang disebut dengan Integrated Development Environment (IDE). Dalam pemrograman berorientasi objek dengan Java ini, IDE yang akan digunakan adalah Netbeans. Oleh karena itu pada modul ini kita akan berkenalan dengan Netbeans mulai dari unduh kemudian install sampai dengan membuat kode pertama dengan IDE atau mengimport dari kode yang lain.

2.2. Tujuan Praktikum

Setelah praktikum pada modul 2 ini diharapkan mahasiswa mempunyai kompetensi memahami software IDE dan bagaimana membuat kode program serta pengelolaan di IDE.

2.3. Material Praktikum

Kegiatan pada modul 2 ini memerlukan material berupa Software IDE Netbeans[2].

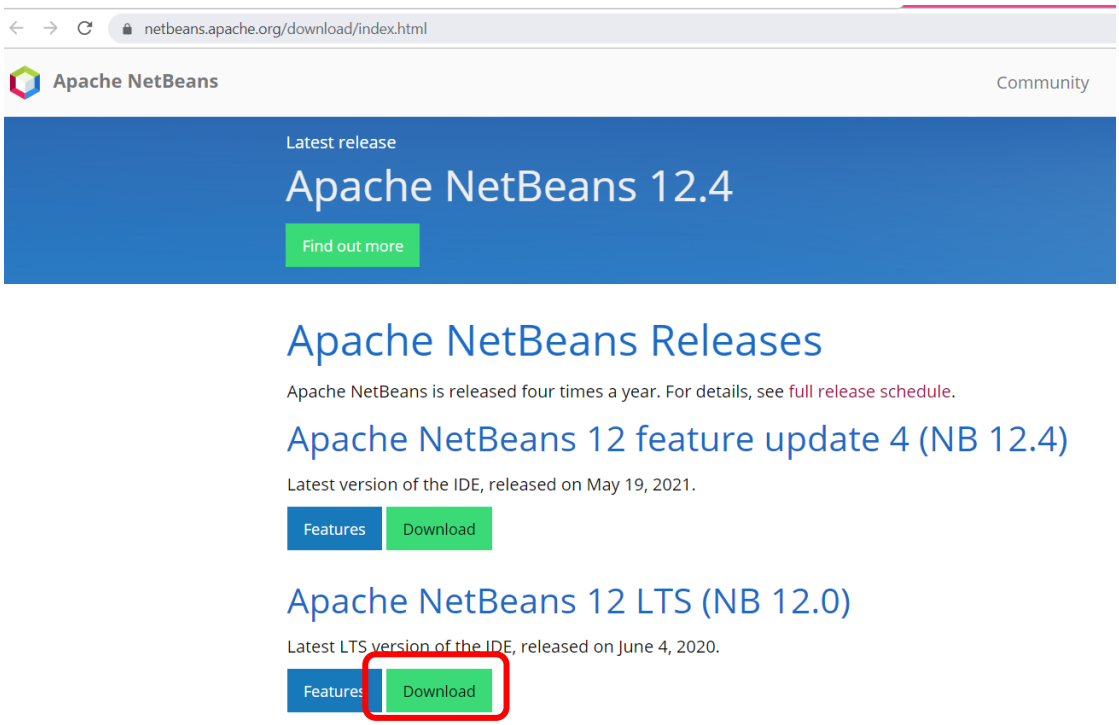
2.4. Kegiatan Praktikum

A. Unduh dan Install

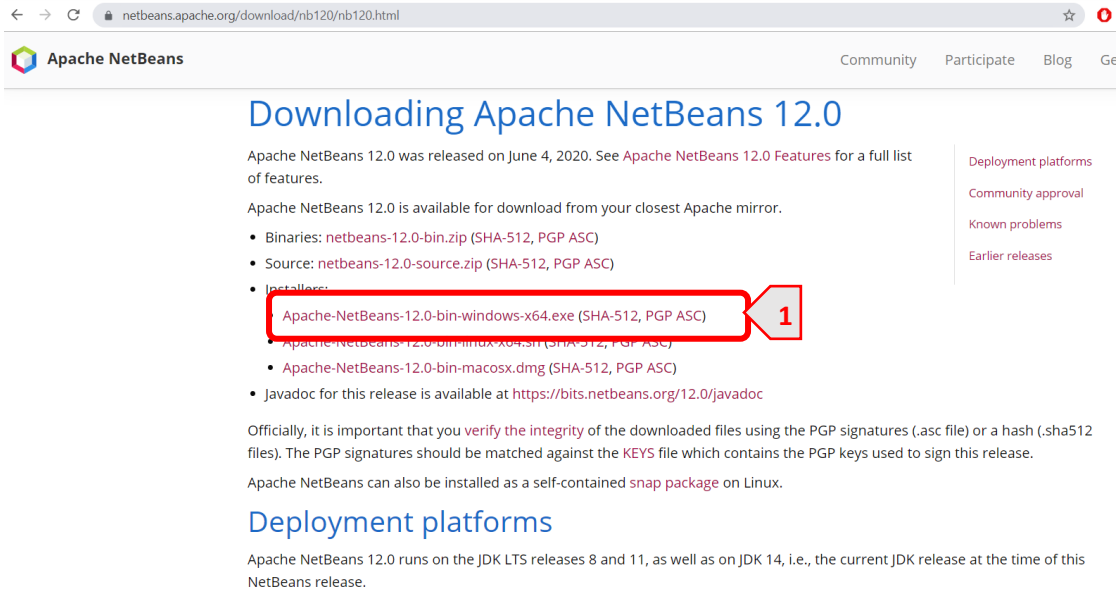
a. Unduh IDE Netbeans

1) Masuk ke halaman resmi Apache Netbeans di https://netbeans.apache.org/download/index.html.
2) Pilih versi yang diinginkan dan klik *download.*

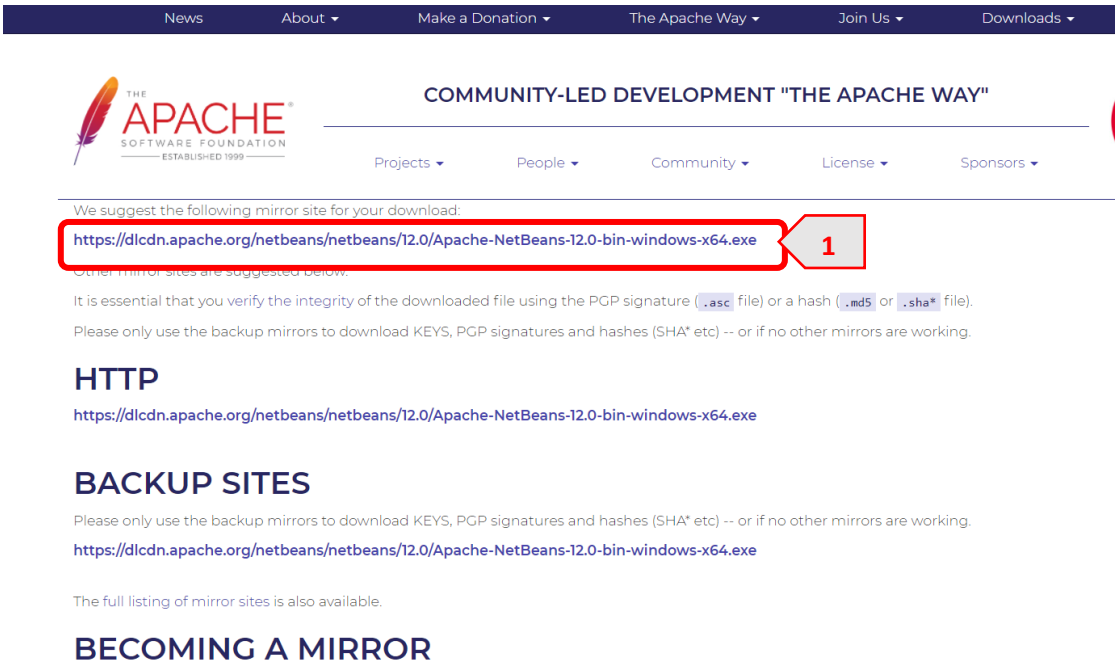
[2] https://netbeans.apache.org/download/index.html



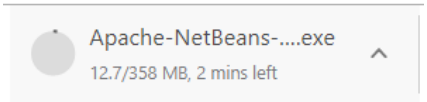
3) Selanjutnya kita akan diarahkan ke halaman download. Pilih file installer sesuai dengan versi OS yang Anda miliki. Untuk windows 64 bit, klik ”Apache-NetBeans-12.4-bin-windows-x64.exe (SHA-512, PGP ASC)”



4) Kemudian kita akan diarahkan ke halaman server. Klik pada salah situs yang direkomendasikan.

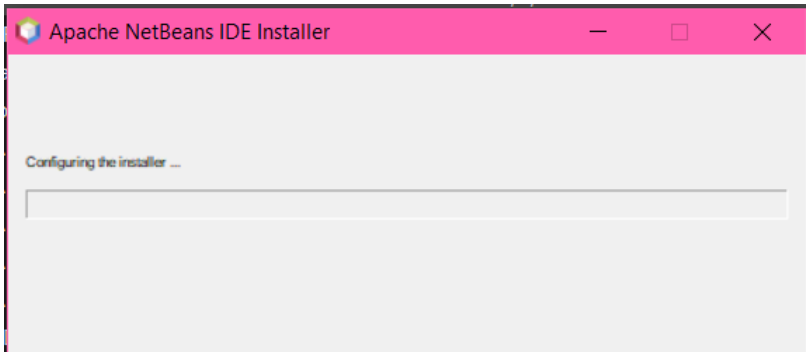


5) Tunggu proses download selesai.

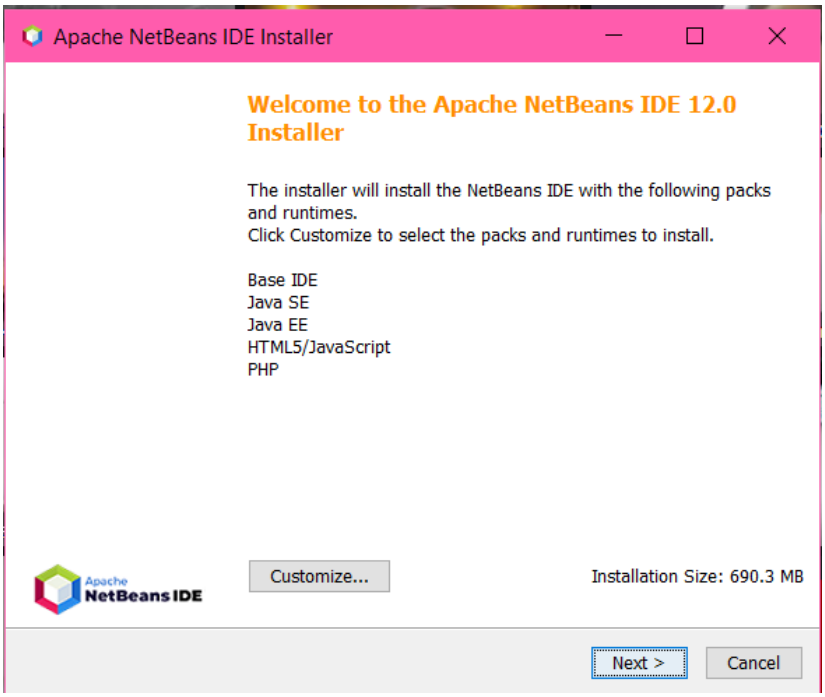


b. Install

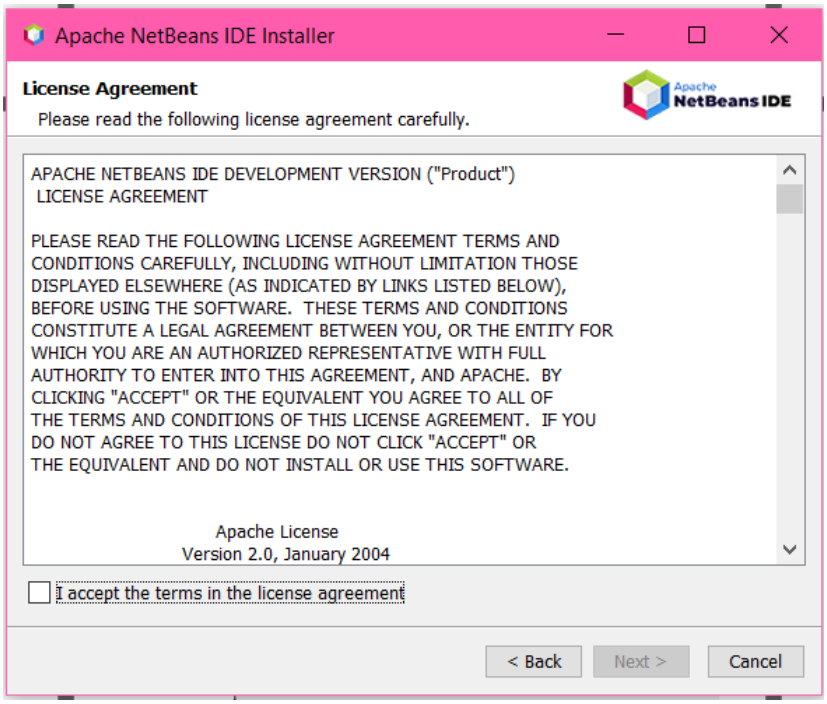
1) Klik ganda pada file installer. Tunggu penginstall bekerja.



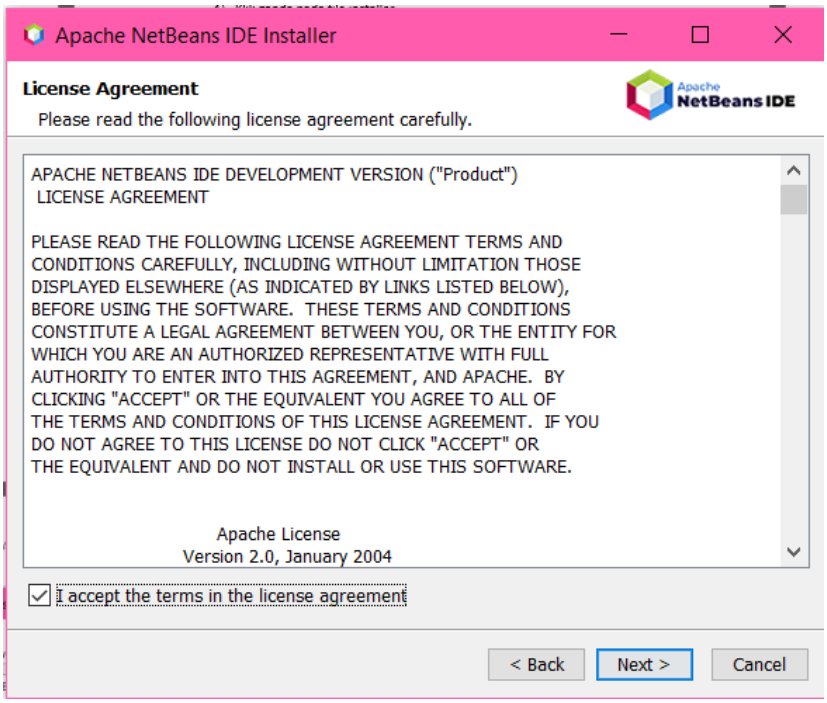
2) Selanjutnya kita diminta untuk memilih apa saja yang akan diinstall. Klik customize untuk mengubah pilihan. Klik Next untuk mengikuti default.



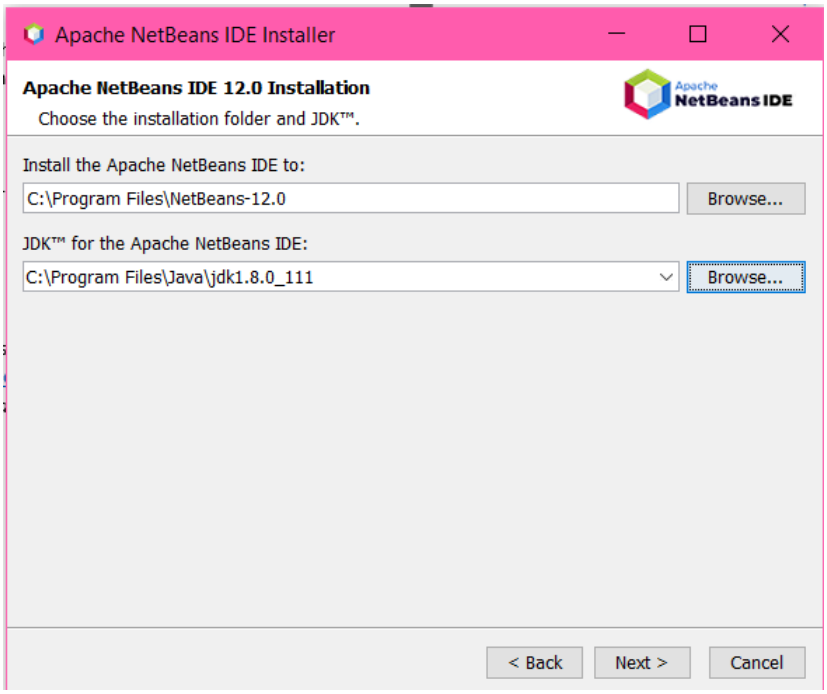
3) Selanjutnya Persetujuan Lisensi akan ditampilkan.



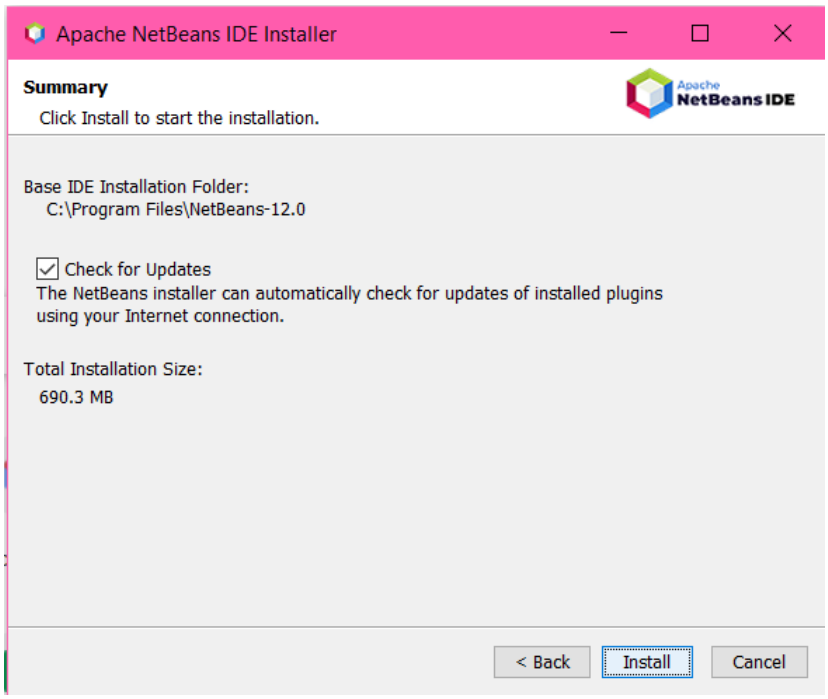
4) Centang pada ”I accept the terms....”, lalu klik Next.



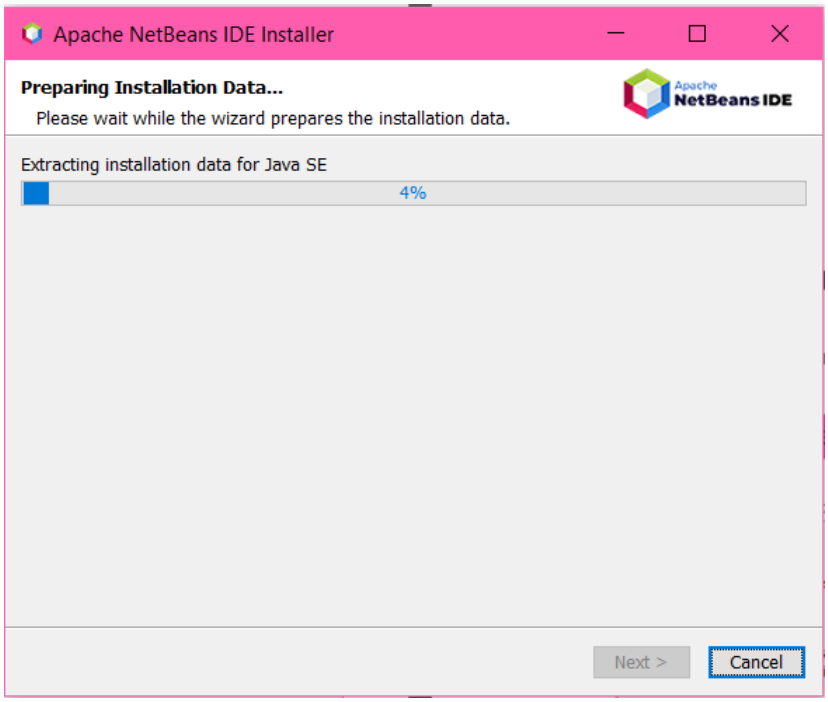
5) Pilih folder lokasi instalasi netbeans dan lokasi jdk.



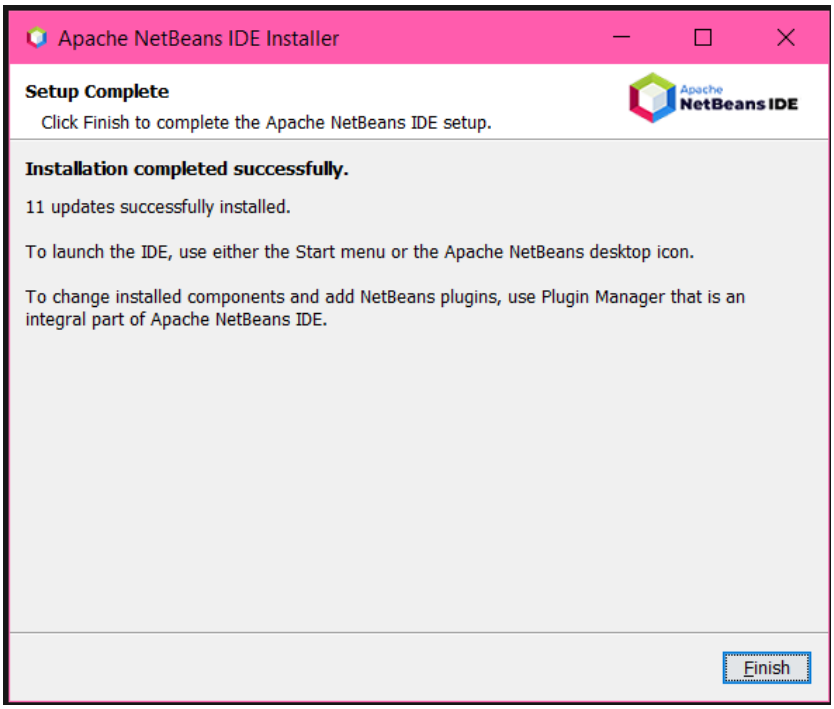
6) Periksa ringkasan persiapan instalasi. Jika sudah sesuai, klik Install. Jika ada yang belum sesuai, klik <Back dan sesuaikan.



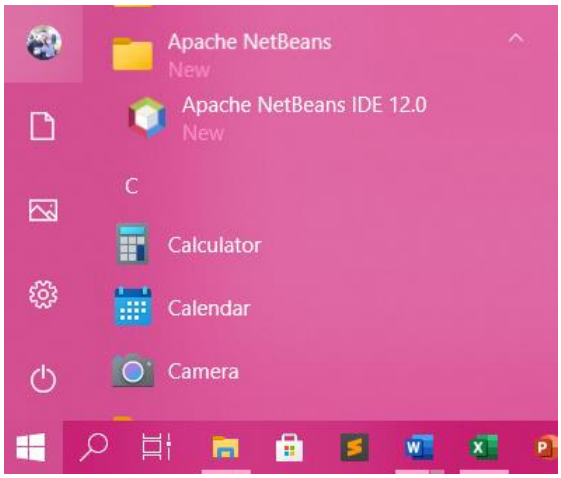
7) Tunggu proses instalasi berjalan sampai selesai.



8) Klik finish untuk menutup jendela.



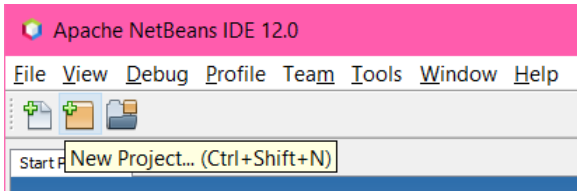
9) Periksa aplikasi, pastikan sudah ada di menu.



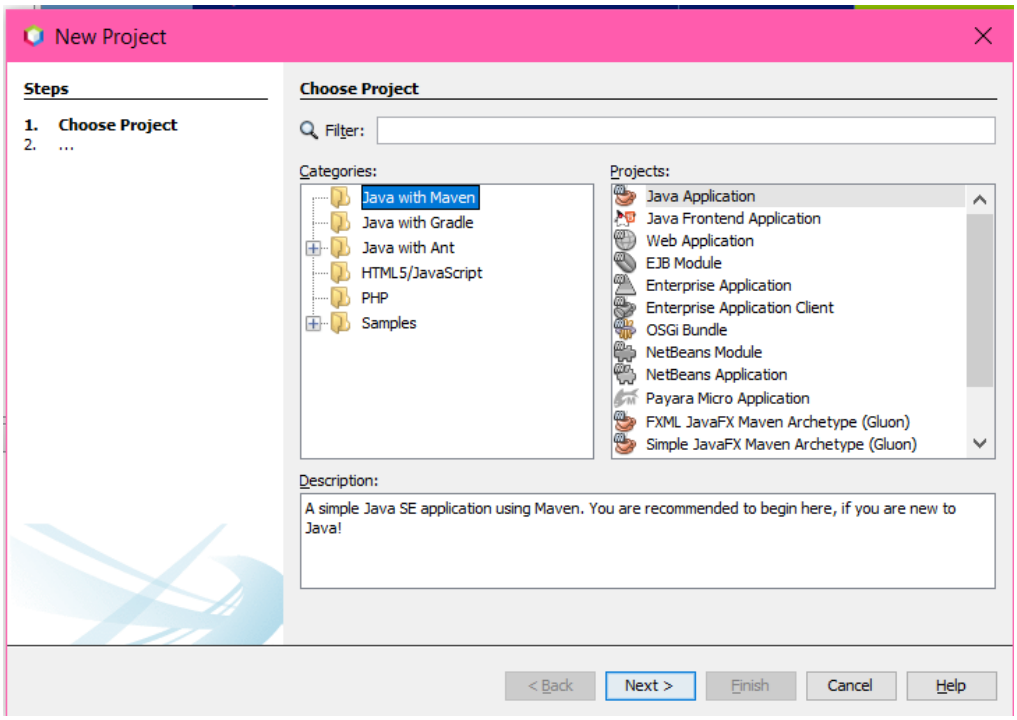
10) Klik ”Apache Netbeans IDE 12.0” untuk membuka aplikasi.

B. Pemrograman Java di Netbeans

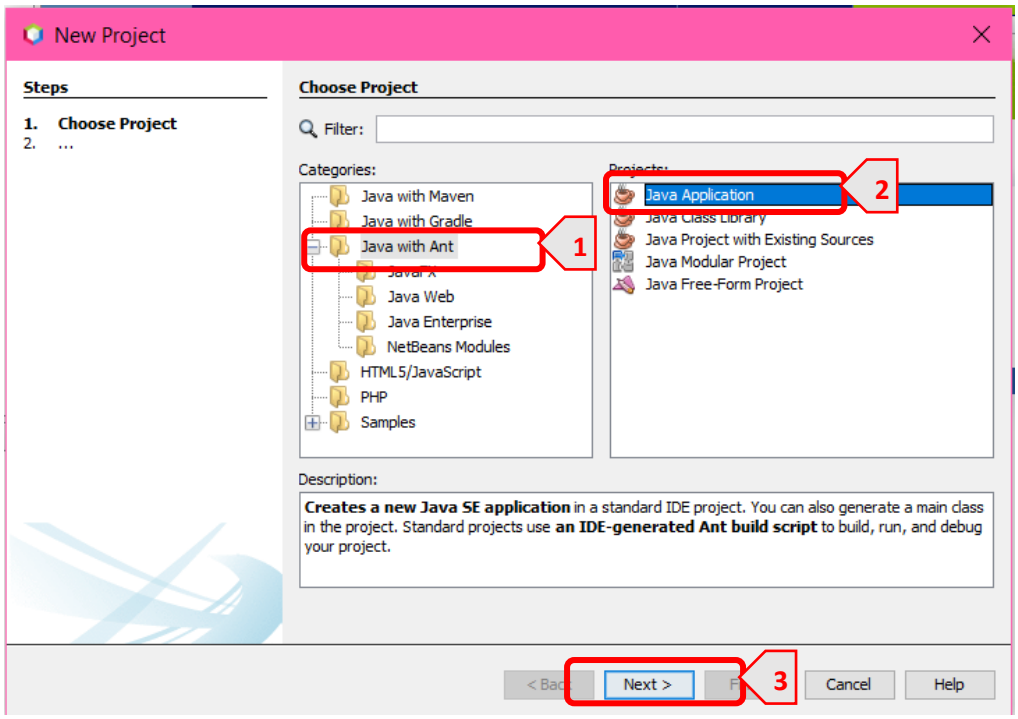
1) Buat new project



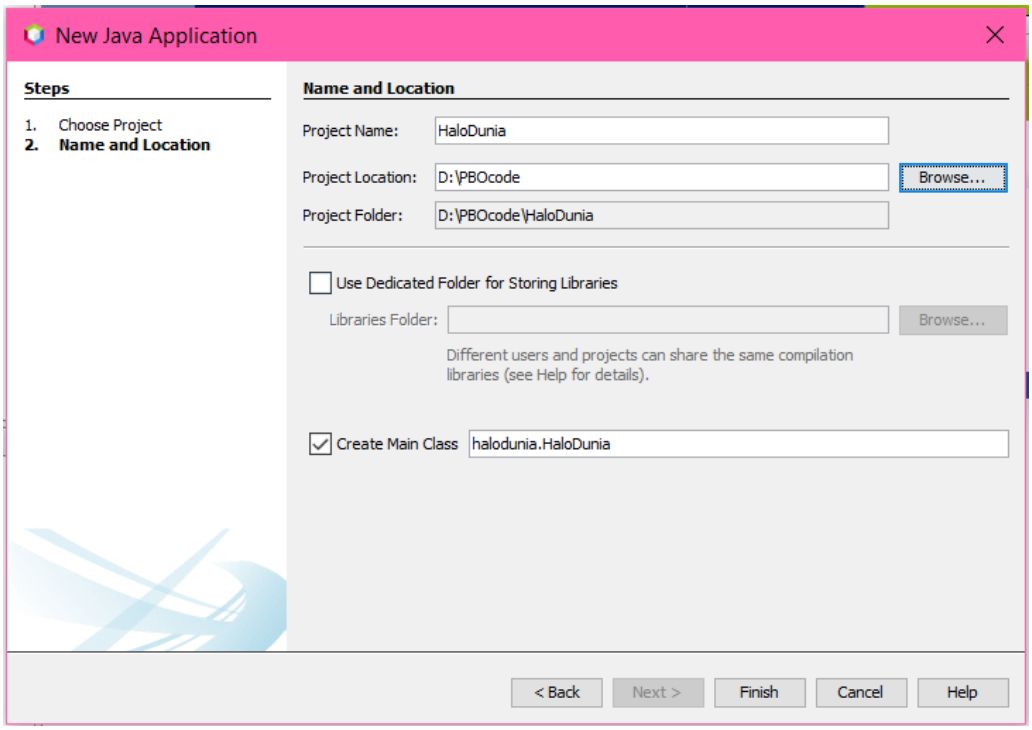
2) Pilih jenis project yang akan dibuat.



3) Pilih Java with Ant >> Java Application. Kemudian klik next

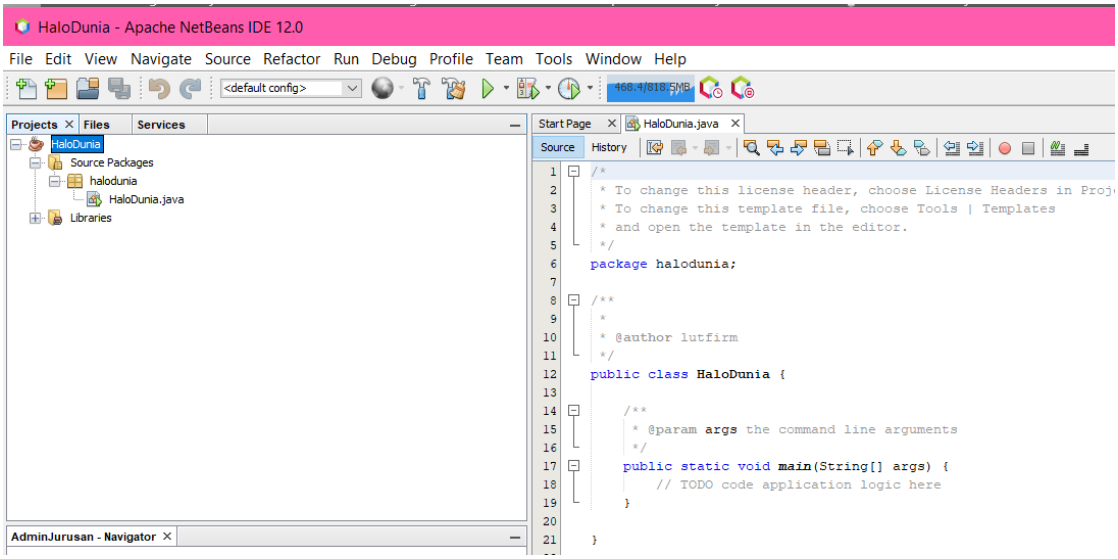


4) Masukkan Nama Project, Lokasi Project, dan nama kelas utama.



5) Klik Finish.

6) Maka project akan di-*generate* dan kelas utama otomatis akan terbuka.

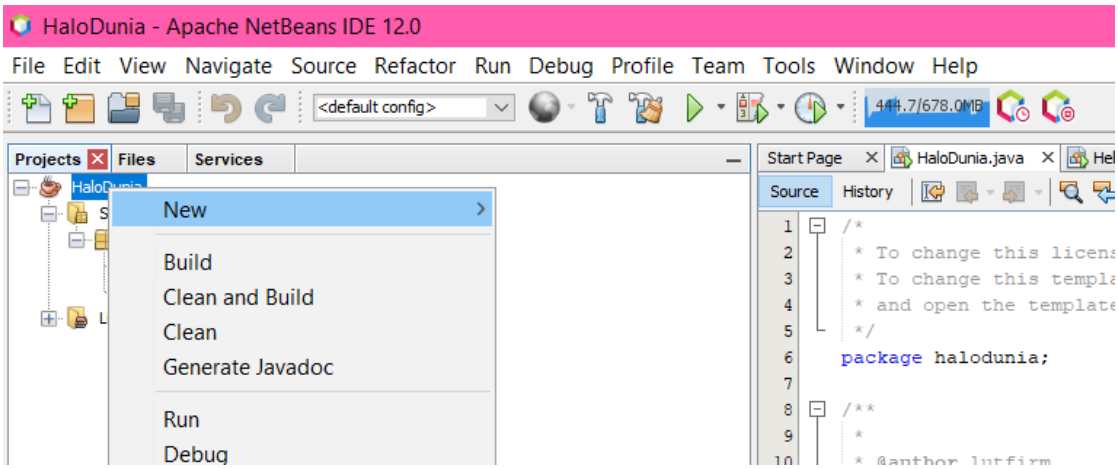


7) Tambahkan baris baru untuk menampilkan ”Halo, Dunia”

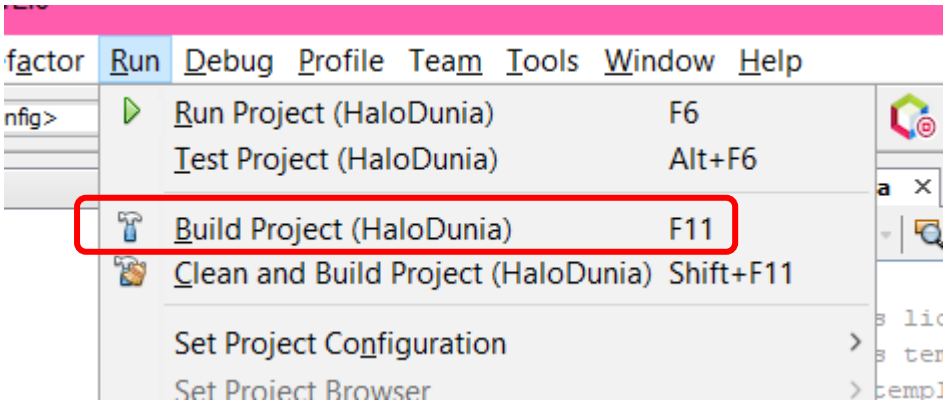
```
12  public class HaloDunia {
13
14      /**
15       * @param args the command line arguments
16       */
17      public static void main(String[] args) {
18          // TODO code application logic here
19          System.out.println("Halo, Dunia!");
20      }
21
22  }
```

8) Compile Program dengan salah satu cara berikut:

   a. Klik kanan di project yang akan compile, kemudian pilih Build.



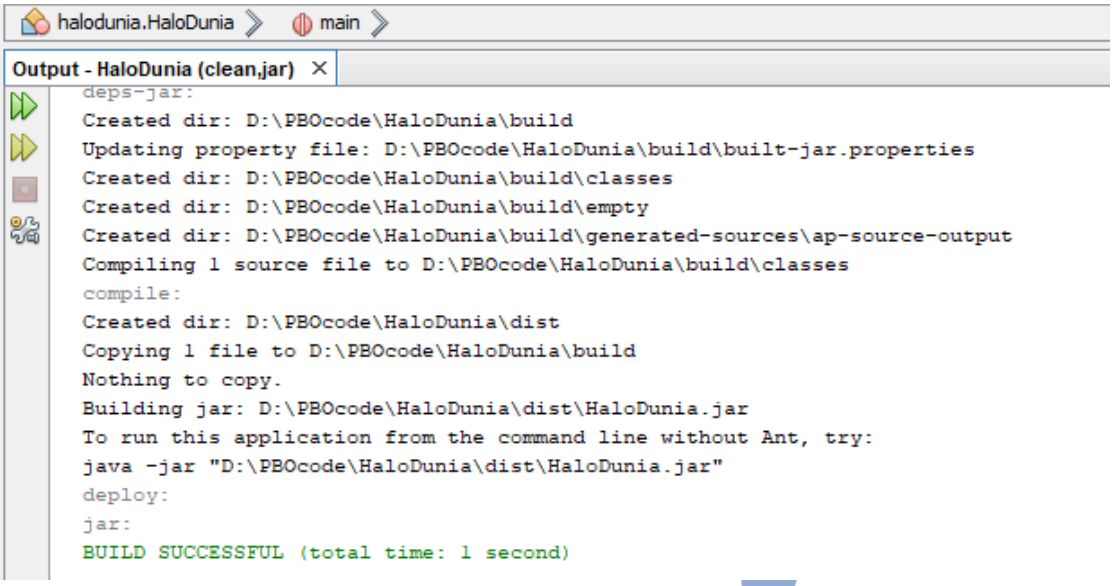
b. Klik menu Run >> Build Project



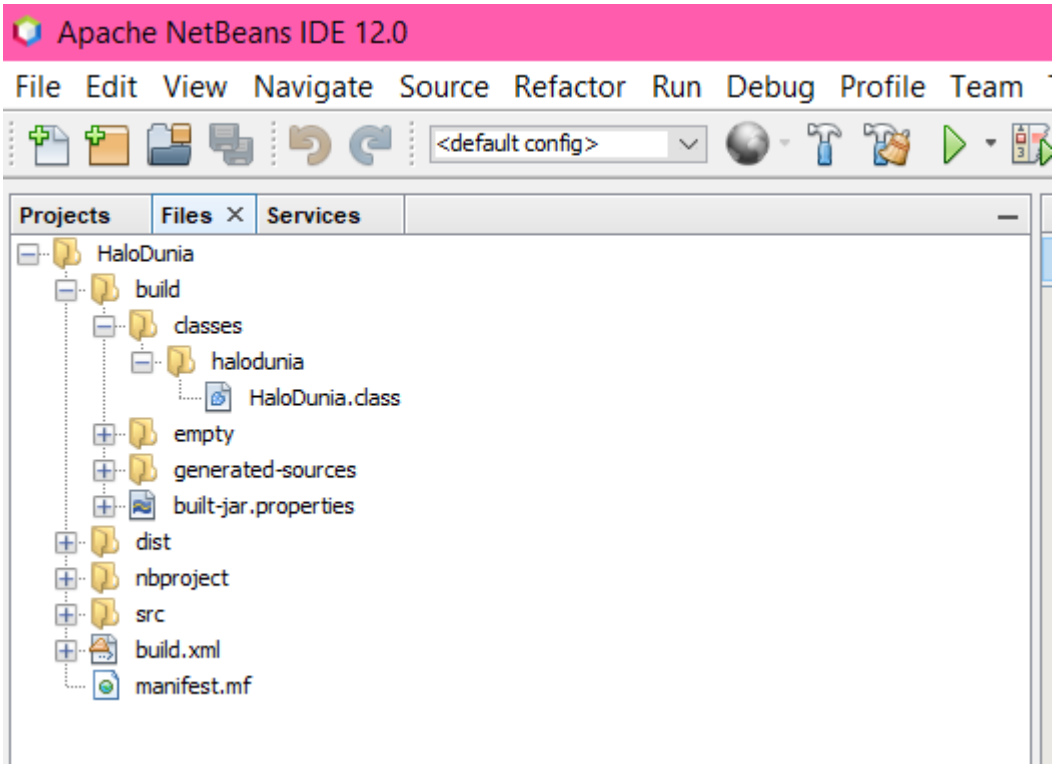
c. Tekan tombol F11

d. Klik ikon Build

9) Jika berhasil maka akan menghasilkan output seperti ini:



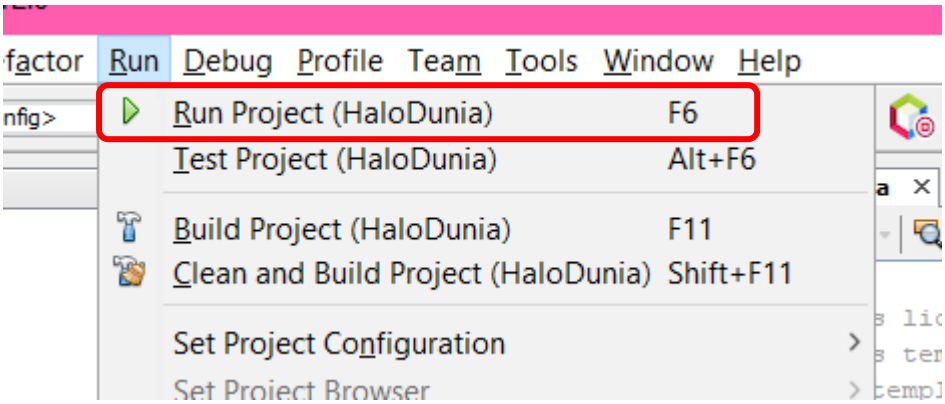
10) Untuk mengecek hasilnya, pindah ke tab File dan lihat isi folder. Akan ada file class yang tergenerate.



11) Jalankan program dengan salah satu cara berikut:

a. Jalankan keseluhan project. Ini akan otomatis menjalakan file program utama.

i. Klik menu Run >> Run Project

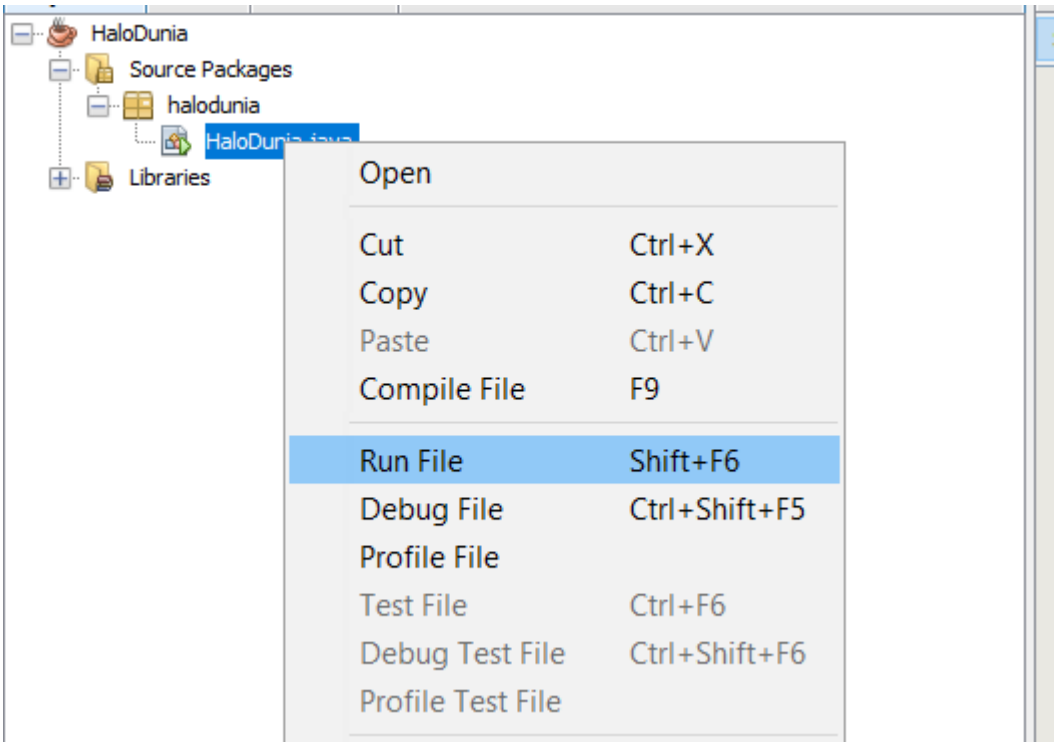


ii. Tekan tombol F6

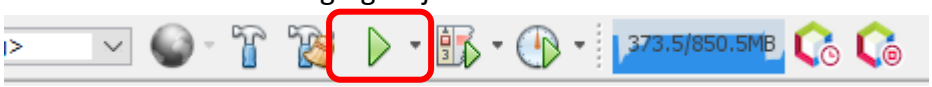
b. Jalankan satu file program

i. Klik kanan file program yang akan dijalankan dan pilih Run File

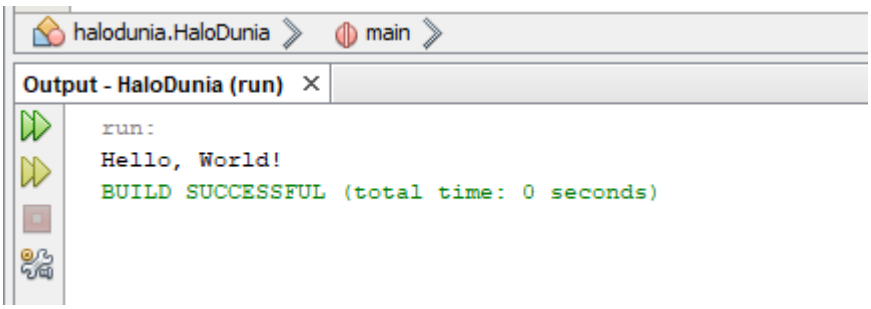
ii. Tekan tombol Shift+F6

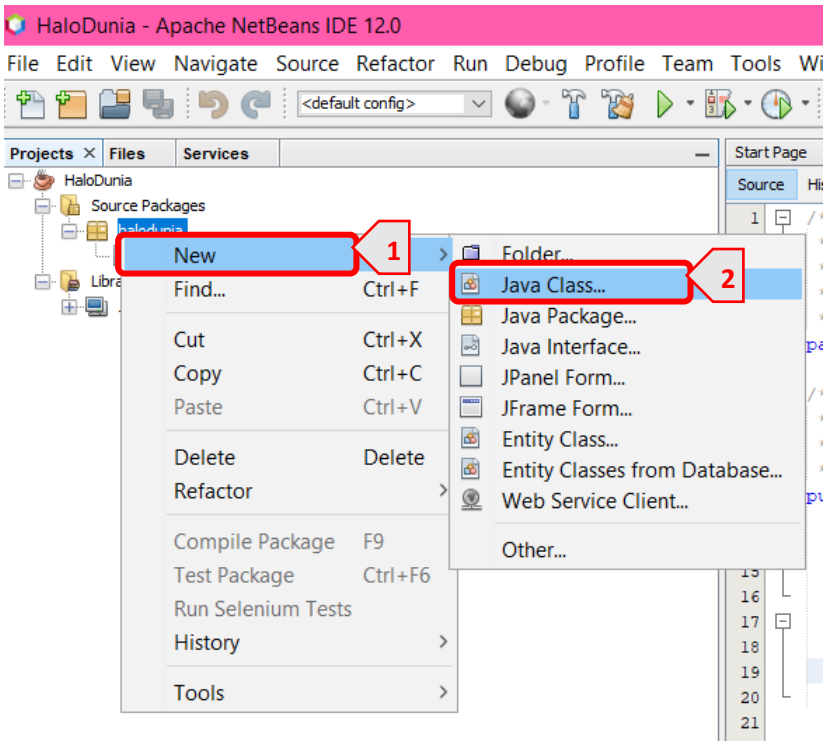


iii. Klik ikon segitiga hijau.

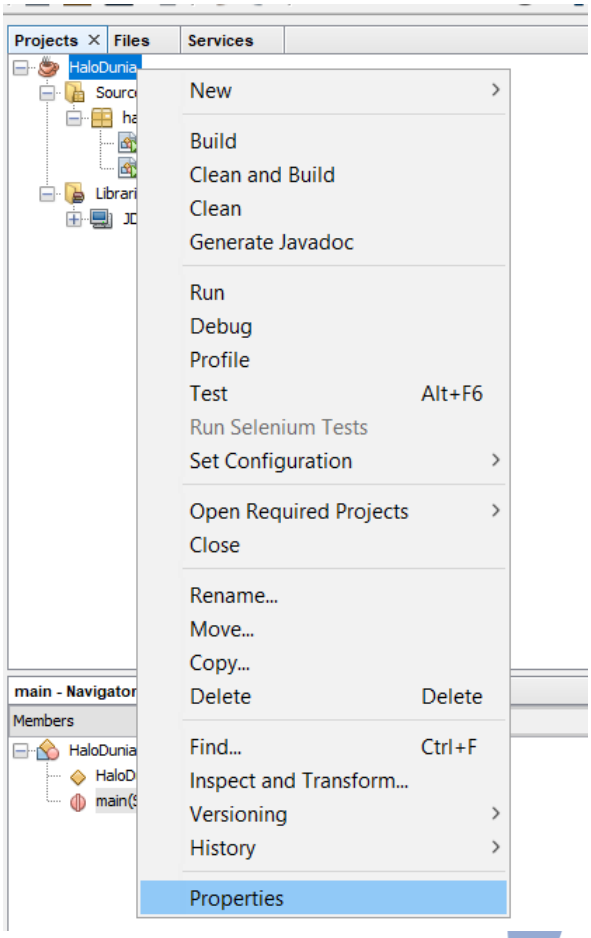


12) Maka hasil akan ditampilkan.

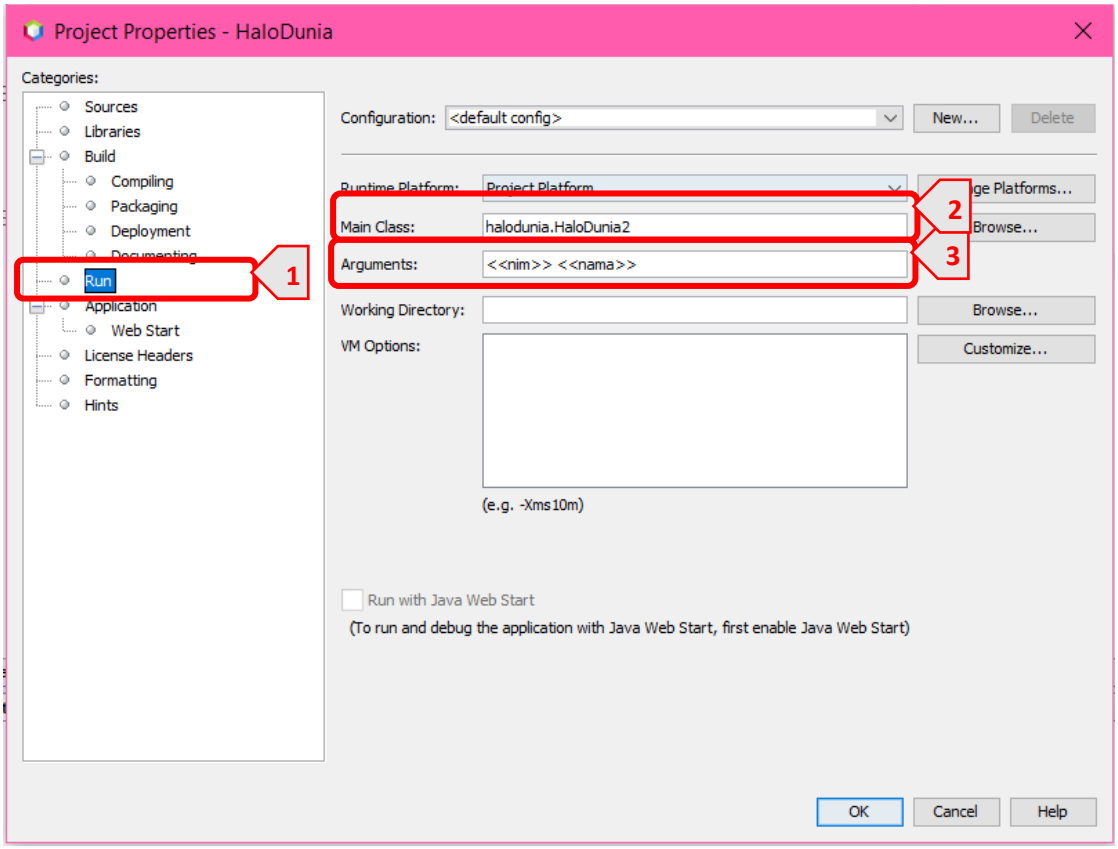




1) Berikan nama ”HaloDunia2”
2) Buat kode yang sama dengan HaloDunia2 di modul1.
3) Klik kanan di project dan pilih properties



4) Pindah ke halaman Run. Ubah main class menjadi HaloDunia2 dan ketikkan argumen berupa <<nim kalian>> dan <<nama kalian>>



5) Run project tersebut, bagaimana hasilnya?
6) Selanjutnya perbaiki semua kode sesuai dengan *coding convention.*

C. Objek vs Class

Objek di Java adalah entitas fisik dan logis, sedangkan kelas di Java adalah entitas logis saja. Kelas adalah sekelompok objek yang memiliki properti umum. Ini adalah template atau cetak biru dari mana objek dibuat. Ini adalah entitas logis. Tidak bisa fisik.

Sebuah kelas di Java dapat berisi:

1) Fields
2) Methods
3) Constructors
4) Blocks
5) Nested class and interface

```
class <class_name>{
    field;
    method;
}
```

Dalam contoh ini, Kita telah membuat kelas Mahasiswa yang memiliki dua id dan nama anggota data. Kita membuat objek kelas Mahasiswa dengan kata kunci baru dan mencetak nilai objek.

Di sini, kita membuat metode main() di dalam kelas.

Mahasiswa.java

```
12  //pertama kita definisikan KELAS Mahasiswa
13  class Mahasiswa{
14   //kemudian definisikan atribut atau variabel yang dimiliki
15
16   //yang pertama misalkan nim yang bertipe int
17   int nim;
18   //yang kedua misalkan nama yang bertipe String
19   String nama;
20   //kemudian kita buat metode utama di dalam kelas Mahasiswa
21   public static void main(String args[]){
22    //Membuat OBJEK atau instance
23
24    //pertama kita buat objek Mahasiswa
25    Mahasiswa s1=new Mahasiswa();
26    //kemudian kita cetak nilai dari objek
27    System.out.println(s1.nim); //kita bisa mengaksesnya melalui variabel referensi
28    System.out.println(s1.nama);
29   }
30  }
31
```

Dalam pengembangan waktu nyata, Kita membuat kelas dan menggunakannya dari kelas lain. Ini adalah pendekatan yang lebih baik dari yang sebelumnya. Mari kita lihat contoh sederhana, di mana kita memiliki metode main() di kelas lain.

Kita dapat memiliki beberapa kelas dalam file Java yang berbeda atau file Java tunggal. Jika Anda mendefinisikan beberapa kelas dalam satu file sumber Java, sebaiknya simpan nama file dengan nama kelas yang memiliki metode main().

Modul2
- Mahasiswa2.java
- MahasiswaMain.java

```
12  //Program Java untuk mendemonstrasikan memiliki metode utama di kelas lain
13  //Membuat kelas Mahasiswa2
14  public class Mahasiswa2 {
15      //kemudian definisikan atribut atau variabel yang dimiliki
16
17      //yang pertama misalkan nim yang bertipe int
18      int nim;
19      //yang kedua misalkan nama yang bertipe String
20      String nama;
21  }
```

```
13  //Membuat kelas lain MahasiswaMain yang berisi metode utama
14  public class MahasiswaMain {
15
16      public static void main(String args[]) {
17          Mahasiswa2 s1 = new Mahasiswa2();
18          System.out.println(s1.nim);
19          System.out.println(s1.nama);
20      }
21
22  }
```

Ada 3 cara untuk menginisialisasi objek di Java.

1. Dengan variabel referensi
2. Dengan metode
3. Oleh konstruktor

1) Objek dan Kelas Contoh: Inisialisasi melalui referensi

Menginisialisasi objek berarti menyimpan data ke dalam objek. Mari kita lihat contoh sederhana di mana kita akan menginisialisasi objek melalui variabel referensi. Ayo coba dengan kelas MahasiswaMain2 yang mempunyai metode utama seperti kode berikut.

```
13  public class MahasiswaMain2 {
14      public static void main(String args[]) {
15          Mahasiswa2 s1 = new Mahasiswa2();
16          //inisiasi variabel
17          s1.nim=123456;
18          s1.nama="Lutfi";
19          System.out.println(s1.nim);
20          System.out.println(s1.nama);
21      }
22  }
```

Kita juga dapat membuat beberapa objek dan menyimpan informasi di dalamnya melalui variabel referensi. Ayo coba dengan kelas MahasiswaMain3 yang mempunyai metode utama seperti kode berikut.

```
13  public class MahasiswaMain3 {
14      public static void main(String args[]) {
15          Mahasiswa2 s1 = new Mahasiswa2();
16          Mahasiswa2 s2 = new Mahasiswa2();
17          //inisiasi variabel
18          s1.nim=123456;
19          s1.nama="Lutfi";
20          s2.nim=123457;
21          s2.nama="Rahma";
22          System.out.println(s1.nim+" "+s1.nama);
23          System.out.println(s2.nim+" "+s2.nama);
24      }
25  }
```

2) Objek dan Kelas Contoh: Inisialisasi melalui metode

Dalam contoh ini, Kita membuat dua objek kelas Mahasiswa3 dan menginisialisasi nilai ke objek ini dengan memanggil metode tambahData. Di sini, kita menampilkan status (data) objek dengan memanggil metode tampilkanInfo().

```
13  public class Mahasiswa3 {
14      int nim;
15      String nama;
16
17      void tambahData(int vnim, String vnama) {
18          nim = vnim;
19          nama = vnama;
20      }
21
22      void tampilkanInfo() {
23          System.out.println(nim + " " + nama);
24      }
25  }
```

Ayo coba dengan kelas MahasiswaMain3 yang mempunyai metode utama seperti kode berikut.

```java
13  public class MahasiswaMain4 {
14      public static void main(String args[]) {
15          Mahasiswa3 s1 = new Mahasiswa3();
16          Mahasiswa3 s2 = new Mahasiswa3();
17
18          s1.tambahData(123456, "Lutfi");
19          s2.tambahData(123457, "Rahma");
20          s1.tampilkanInfo();
21          s2.tampilkanInfo();
22      }
23  }
```

3) Objek dan Kelas Contoh: Inisialisasi melalui konstruktor

Kita akan belajar tentang konstruktor di Java nanti.

Di Java, konstruktor sama seperti metode tetapi tanpa tipe pengembalian. Itu juga bisa kelebihan beban seperti metode Java. Ayo coba dengan kelas Mahasiswa4 yang mempunyai konstruktor seperti metode tapi dengan nama yang sama dengan nama kelasnya seperti kode berikut.

```java
13  public class Mahasiswa4 {
14      int nim;
15      String nama;
16
17      Mahasiswa4(int vnim, String vnama) {
18          nim = vnim;
19          nama = vnama;
20      }
21
22      void tampilkanInfo() {
23          System.out.println(nim + " " + nama);
24      }
25  }
```

Selanjutnya ayo coba dengan kelas MahasiswaMain5 yang mempunyai metode utama seperti kode berikut. Bedakan dengan kelas MahasiswaMain4.

```java
13  public class MahasiswaMain5 {
14      public static void main(String args[]) {
15          Mahasiswa4 s1 = new Mahasiswa4(123456, "Lutfi");
16          Mahasiswa4 s2 = new Mahasiswa4(123457, "Rahma");
17
18          s1.tampilkanInfo();
19          s2.tampilkanInfo();
20      }
21  }
```

2.5. Penugasan

Laporkan hasil praktikum berikut dengan hasil penugasan dan penjelasannya ke Dosen dalam bentuk file pdf dengan format nama *<<nim>>_modul2.*

1) Tangkapan layar hasil akhir instalasi
2) Tangkapan layar hasil running

MODUL 3 INHERITANCE, POLYMORPHISM, AGGREGATION DAN COMPOSITION

3.1. Deskripsi Singkat

Dalam pemrograman berorientasi objek ada beberapa sifat yang dipelajari, yaitu inheritance, polymorphism, aggregation dan composition. Oleh karena itu, pada modul 3 ini kita akan mempraktekkan bagaimana implementasi sifat dalam objek dalam kode.

3.2. Tujuan Praktikum

Setelah praktikum pada modul 3 ini diharapkan mahasiswa mempunyai kompetensi memahami sifat-sifat dalam pemrograman berorientasi yaitu inheritance, polymorphism, aggregation dan composition

3.3. Material Praktikum

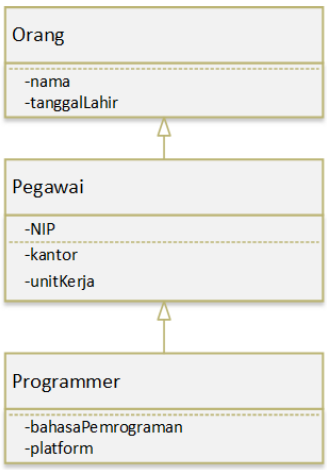
Kegiatan pada modul 3 ini memerlukan material berupa Software IDE Netbeans[3].

3.4. Kegiatan Praktikum

A. INHERITANCE

Misalkan kita punya objek Orang. Orang mempunyai nama dan tanggal lahir. Kemudian ada orang yang merupakan pegawai. Seorang pegawai bekerja di suatu kantor di bagian tertentu. Pegawai diidentifikasi dengan nomor induk pegawai (NIP). Selain itu, pegawai yang berkerja di bagian IT ada yang merupakan seorang programmer. Seorang programmer harus menguasai satu bahasa pemrograman di platform tertentu. Jika kita gambarkan dalam diagram maka akan menjadi seperti berikut.

[3] https://netbeans.apache.org/download/index.html



Pegawai adalah Orang, dengan kata lain Pegawai adalah turunan dari Orang, jenis yang lebih khusus dari Orang. Programmer adalah Pegawai, dengan kata lain Programmer adalah turunan dari Pegawai, jenis yang lebih khusus dari Pegawai.

Untuk membuat kode dari pewarisan atau penurunan sifat ini, Pertama kita buat kelas untuk Orang. Masih ingat caranya?

1) Silakan buat kelas untuk Orang.

```
 */
public class Orang {

}
```

2) Kemudian kita tambahkan atribut yang dimiliki oleh Orang, yaitu nama dan tanggalLahir.

```
public class Orang {
    private String nama;
    private Date tanggalLahir;

}
```

3) Untuk dapat memungkinkan membuat objek dari kelas, kita sediakan constructor. Kita bisa sediakan tiga jenis constructor.
   a. Constructor tanpa atribut, untuk memungkinkan kita membuat objek tanpa memberikan atributnya terlebih dahulu.

```
public Orang() {
}
```

   b. Constructor dengan sebagian atribut, misalkan nama, untuk memungkinkan kita membuat objek dengan memberikan nama pada saat membuat objek dan kemudian bisa memberikan nilai untuk atribut lainnya nanti.

```
public Orang(String nama) {
    this.nama = nama;
}
```

c. Constructor dengan semua atribut, untuk memungkinkan kita membuat objek dengan memberikan nilai untuk semua atribut objek tersebut.

```
public Orang(String nama, Date tanggalLahir) {
    this.nama = nama;
    this.tanggalLahir = tanggalLahir;
}
```

4) Selanjutnya karena tidak semua atribut dimasukkan di awal pada saat membuat objek, maka kita sediakan method untuk memasukkan nilai tersebut.

```
public void setNama(String nama) {
    this.nama = nama;
}
```

a. Kita bisa menggunakan nama variabel yang sama antara parameter method dan atribut di kelas.

b. Untuk mengakses atribut milik kelas gunakan *this*.

5) Selanjutnya kita buat method untuk mengakses nilai dari atribut yang dimiliki.

```
public String getNama() {
    return nama;
}
```

Selanjutnya kita buat kelas untuk Pegawai yang merupakan turunan dari Orang.

1) Buat kelas untuk Pegawai seperti kelas biasanya.
2) Tambahkan extends Orang untuk menandakan bahwa Pegawai adalah turunan dari Orang.

```
public class Pegawai extends Orang{

}
```

3) Selanjutnya tambahkan atribut khusus yang dimiliki pegawai.

```
public class Pegawai extends Orang{
    private String NIP;
    private String namaKantor;
    private String unitKerja;
```

4) Selanjutnya buatlah constructor yang memungkinkan untuk kelas Pegawai seperti pada Orang. Yang berbeda di sini adalah setiap kita membuat constructor, maka kita juga memasukkan atribut yang diperlukan di constructor super class dari Pegawai, yaitu Orang.

```java
public Pegawai() {
}

public Pegawai(String NIP, String namaKantor, String unitKerja) {
    this.NIP = NIP;
    this.namaKantor = namaKantor;
    this.unitKerja = unitKerja;
}

public Pegawai(String nama, String NIP, String namaKantor, String unitKerja) {
    super(nama);
    this.NIP = NIP;
    this.namaKantor = namaKantor;
    this.unitKerja = unitKerja;
}
```

a. Kita bisa menggunakan `super(nama)` untuk memberikan nilai pada atribut nama yang dimiliki dari penurunan.

5) Selanjutnya kita bisa membuat method untuk mengambil nilai atribut dan memasukkan nilai atribut seperti pada Orang. Kita tidak perlu membuat method untuk mengambil dan memasukkan nilai atribut turunan.

```java
public String getNIP() {
    return NIP;
}

public void setNIP(String NIP) {
    this.NIP = NIP;
}

public String getNamaKantor() {
    return namaKantor;
}

public void setNamaKantor(String namaKantor) {
    this.namaKantor = namaKantor;
}

public String getUnitKerja() {
    return unitKerja;
}

public void setUnitKerja(String unitKerja) {
    this.unitKerja = unitKerja;
}
```

Selanjutnya kita bisa membuat instance atau obyek dari kelas inheritance.

1) Kita buat kelas baru dengan method main() di dalamnya. Misalkan nama kelasnya adalah Kantor.

```
public class Kantor {

    public static void main(String[] args) {


    }

}
```

2) Kita bisa buat objek dari Orang tanpa menyebutkan nilai atributnya. Setelah objek terbuat, baru kita masukkan nilai atributnya dengan memanggil methodnya.

```
Orang lutfi=new Orang();
lutfi.setNama("Lutfi");
```

3) Kita bisa buat objek dari Orang dengan langsung memasukkan nilai atributnya. Setelah objek terbuat, kita bisa masukkan nilai atributnya atau mengubah nilainya dengan memanggil methodnya

```
Orang rahma=new Orang("Rahma");
rahma.setTanggalLahir(new Date(1997, 8, 31));
```

4) Kita bisa tampilkan objeknya dengan cara berikut:

```
System.out.println("Ada orang:");
System.out.println(lutfi.getNama()+" lahir pada "+lutfi.getTanggalLahir());
System.out.println(rahma.getNama()+" lahir pada "+rahma.getTanggalLahir());
```

5) Cara yang sama kita bisa lakukan untuk kelas yang merupakan turunan dari Orang.

```
Pegawai tuti=new Pegawai("Tuti",new Date(1997, 8, 2),"6836492379","STIS","IT");
System.out.println(tuti.getNama()+
        " lahir pada "+tuti.getTanggalLahir()+
        " NIP: "+tuti.getNIP()+
        " kantor: "+tuti.getNamaKantor()+
        " bagian: "+tuti.getUnitKerja());
```

6) Silakan bereksperimen dengan kelas turunan dari pewarisan tersebut!

B. POLYMORPHISM

POLYMORPHISM (Polimorfisme) berarti "banyak bentuk", dan itu terjadi ketika kita memiliki banyak kelas yang terkait satu sama lain melalui pewarisan. Warisan memungkinkan kita mewarisi atribut dan metode dari kelas lain. Polimorfisme menggunakan metode tersebut untuk melakukan tugas yang berbeda. Ini memungkinkan kita untuk melakukan satu tindakan dengan cara yang berbeda.

Misalnya Ketika Orang ditanya gaji, kita bisa menjawab tidak ada, sedangkan jika Pegawai ditanya gaji, kita bisa menjawab nilainya misalkan 7 juta, dan untuk

programmer, kita bisa menjawab nilainya adalah 7 juta dan tambahan lembur 500 ribu.

1) Kita tambahkan method untuk mendapatkan gaji di Orang.

```
public void getGaji() {
    System.out.println("tidak ada");
}
```

2) Kita tambahkan method untuk mendapatkan gaji di Pegawai.

```
public void getGaji() {
    System.out.println("7 juta");
}
```

3) Kita bisa cek dengan memanggil method gaji dari objek Orang dan Pegawai yang sudah dibuat sebelumnya.

```
System.out.println("gaji Orang Rahma :"+rahma.getGaji());
System.out.println("gaji Orang Tuti :"+tuti.getGaji());
```

4) Kita akan mendapatkan hasil yang berbeda dari kedua Objek tersebut.

```
gaji Orang Rahma :tidak ada
gaji Orang Tuti :7 juta
```

C. AGGREGATION

Misalkan di suatu Unit Kerja yang mempunyai nama di kantor, ada beberapa pegawai yang bekerja. Kita bisa membuat kelas Unit Kerja dan menambahkan Pegawai sebagai bagian dari Unit Kerja tersebut.

1) Kita buat kelas Unit Kerja dengan atribut berupa nama dan daftar pegawai yang bertipe pegawai. (Unit Kerja mempunyai Pegawai dan Pegawai bagian dari Unit Kerja).

```
public class UnitKerja {
    private String nama;
    private List<Pegawai> daftarPegawai;

}
```

2) Selanjutnya kita buatkan constructor dan method untuk mengakses nilai atributnya.

```java
public UnitKerja(String nama, List<Pegawai> pegawais) {
    this.nama = nama;
    this.daftarPegawai = pegawais;
}

public String getNama() {
    return nama;
}

public List<Pegawai> getDaftarPegawai() {
    return daftarPegawai;
}
```

3) Kita bisa memasukkan Pegawai yang sudah kita buat ke daftar pegawai dan kemudian menambahkan ke Unit Kerja yang dibuat.

```java
List<Pegawai> daftarPegawai=new ArrayList<Pegawai>();
daftarPegawai.add(tuti);
UnitKerja Umum= new UnitKerja("Umum", daftarPegawai);
```

D. COMPOSITION

Komposisi adalah bentuk agregasi terbatas di mana dua entitas sangat bergantung satu sama lain. Komposis mewakili bagian dari hubungan. Dalam komposisi, kedua entitas bergantung satu sama lain. Ketika ada komposisi antara dua entitas, objek yang dikomposisikan tidak dapat eksis tanpa entitas lainnya.

Misalnya Gedung yang terdiri dari ruangan. Setiap Gedung pasti punya ruangan. Jika tidak ada ruangan, maka gedung tidak akan ada.

1) Pertama kita buat Kelas Ruang yang mempunyai nama.

```java
public class Ruang {
    private String namaRuang;

    public Ruang(String namaRuang) {
        this.namaRuang = namaRuang;
    }

    public String getNamaRuang() {
        return namaRuang;
    }

    public void setNamaRuang(String namaRuang) {
        this.namaRuang = namaRuang;
    }

}
```

2) Selanjutnya kita buat Kelas Gedung dengan nama dan ruangannya

```
public class Gedung {
    private List<Ruang> daftarRuang=new ArrayList<Ruang>();

    public Gedung() {
        Ruang ruang=new Ruang("Utama");
        daftarRuang.add(ruang);
    }

    public void addRuang(String namaRuang) {
        Ruang ruang=new Ruang(namaRuang);
        daftarRuang.add(ruang);
    }

    public List<Ruang> getDaftarRuang() {
        return daftarRuang;
    }
}
```

3) Selanjutnya kita buat objek dari gedung STIS yang mempunyai 3 ruang yaitu Lobi, Ruang Bagian Umum dan Ruang Kepala Kantor.

```
Gedung STIS=new Gedung();
STIS.addRuang("Lobi");
STIS.addRuang("Bagian Umum");
STIS.addRuang("Kepala Kantor");

List<Ruang> ruangan = STIS.getDaftarRuang();
for(Ruang ruang : ruangan){
    System.out.println("Ruang : "+ruang.getNamaRuang());
}
```

4) Nah di sini kita bisa lihat bahwa Gedung STIS tidak bisa ada tanpa adanya ruangan dan ketika gedung STIS dihancurkan, maka ruangan tersebut juga akan hancur.

3.5. Penugasan

Laporkan hasil praktikum berikut dengan hasil penugasan dan penjelasannya ke Dosen dalam bentuk file pdf dengan format nama *<<nim>>_modul3.*

1) Tangkapan layar hasil eksperimen
2) Tangkapan layar hasil running

MODUL 4 ABSTRACT CLASS DAN INTERFACE

4.1. Deskripsi Singkat

Dalam pemrograman berorientasi objek ada beberapa sifat yang dipelajari, yaitu abstract class dan interface. Oleh karena itu, pada modul 4 ini kita akan mempraktekkan bagaimana implementasi abstract class dan interface dalam objek dalam kode.

4.2. Tujuan Praktikum

Setelah praktikum pada modul 4 ini diharapkan mahasiswa mempunyai kompetensi memahami sifat-sifat dalam pemrograman berorientasi yaitu abstract class dan interface

4.3. Material Praktikum

Kegiatan pada modul 4 ini memerlukan material berupa Software IDE Netbeans[4].

4.4. Kegiatan Praktikum

A. ABSTRACT CLASS

Misalkan kita punya objek Orang. Orang mempunyai nama dan tanggal lahir. Kemudian ada orang yang merupakan pegawai. Seorang pegawai bekerja di suatu

1) Buat class Orang seperti biasa.
2) Tambahkan abstract di depannya.

```
abstract class Orang {
    String nama;
    Date tanggalLahir;

    public Orang() {
    }

}
```

3) Tambahkan constructor dan method get-set yang diperlukan
4) Tambahkan abstract method yang nanti harus juga dimiliki oleh kelas yang menjadi turunannya. Misalkan untuk get-set alamat.

```
abstract public String getAlamat();
abstract public void setAlamat(String alamat);
```

5) Di kelas pegawai seperti yang sudah kita buat di modul sebelumnya, tambahkan atribut alamat.

[4] https://netbeans.apache.org/download/index.html

```
private String alamat;
```

6) Tambahkan method untuk get set alamat seperti dengan yang dibuat abstract di Orang namun ditambahkan atribut alamat.

```
@Override
public String getAlamat() {
    return alamat;
}
@Override
public void setAlamat(String alamat) {
    this.alamat=alamat;
}
```

7) Sekarang coba buat objek dari kelas abstract Orang. Amati yang terjadi.

```
Orang o=new Orang();
```

8) Sekarang coba buat objek dari kelas Pegawai dan lengkapi atributnya.

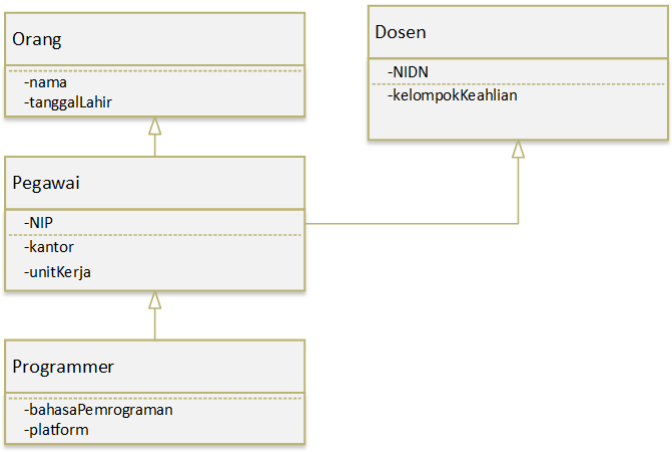
```
Pegawai lutfi=new Pegawai();
```

9) Coba juga untuk memberikan dan mengambil nilai pada alamat melalui method get-set alamat.

```
lutfi.setAlamat("Otista 64 C");

System.out.println(lutfi.getAlamat());
```

B. INTERFACE

Misalkan Pegawai yang merupakan Orang, juga bisa merupakan dosen. Seorang dosen akan mempunyai Nomor Induk Dosen Nasional (NIDN) dan bidang keahlian. Kita bisa gambarkan dengan diagram seperti gambar berikut. Pegawai mempunyai multi inheritance, dari Orang dan dari Dosen.



Tetapi kita tidak bisa membuat kelas yang meng-extends 2 kelas sekaligus. Alternatifnya adalah kita buat superkelas yang 1 lagi sebagai interface. Di Interface Dosen, kita berikan method apa saja (tanpa body) yang akan diimplementasikan oleh kelas yang lain, yaitu method untuk memberikan dan mengambil nilai NIDN dan kelompok Keahlian.

```
interface Dosen {
    public String getNIDN();
    public void setNIDN(String NIDN);
    public String getKelompokKeahlian();
    public void setKelompokKeahlian(String kelompokKeahlian);

}
```

Setelah itu kita bisa implementasikan interface Dosen di Pegawai.

```
public class Pegawai extends Orang implements Dosen{
```

Kemudian kita implementasikan method yang ada di interface Dosen.

```
@Override
public String getNIDN() {

}

@Override
public void setNIDN(String NIDN) {

}

@Override
public String getKelompokKeahlian() {

}

@Override
public void setKelompokKeahlian(String kelompokKeahlian) {

}
```

Selanjutnya kita sesuaikan body dari method tersebut dengan terlebih dahulu menambahkan atribut yang sesuai, yaitu NIDN dan keahlian pada kelas Pegawai.

```
String NIDN;
String keahlian;
```

```java
@Override
public String getNIDN() {
    return NIDN;
}

@Override
public void setNIDN(String NIDN) {
   this.NIDN=NIDN;
}

@Override
public String getKelompokKeahlian() {
    return keahlian;
}

@Override
public void setKelompokKeahlian(String kelompokKeahlian) {
    this.keahlian=kelompokKeahlian;
}
```

Selanjutnya kita bisa mencobanya..

```java
Pegawai lutfi=new Pegawai();

lutfi.setAlamat("Otista 64 C");
System.out.println(lutfi.getAlamat());

lutfi.setNIDN("12345678");
lutfi.setKelompokKeahlian("Computer Science");
System.out.println("ada dosen Lutfi dengan NIDN "+lutfi.getNIDN() +
        " kelompok "+lutfi.getKelompokKeahlian());
```

4.5. Penugasan

Laporkan hasil praktikum berikut dengan hasil penugasan dan penjelasannya ke Dosen dalam bentuk file pdf dengan format nama *<<nim>>_modul4.*

1) Lengkapi kode semua objek
2) Tangkapan layar hasil kode yang dilengkapi
3) Tangkapan layar hasil running

MODUL 5 KONVERSI DIAGRAM KELAS KE KODE

5.1. Deskripsi Singkat

Dalam pemrograman berorientasi objek ada metode yang dipelajari, yaitu Unified Modelling Language (UML). Salah satu diagram yang berhubungan dengan pemrograman berorientasi objek adalah diagram kelas. Oleh karena itu pada modul 5 ini kita akan mempelajari bagaimana mengonversi diagram kelas ke kode.

5.2. Tujuan Praktikum

Setelah praktikum pada modul 5 ini diharapkan mahasiswa mempunyai kompetensi bagaimana mengonversi diagram kelas ke kode.

5.3. Material Praktikum

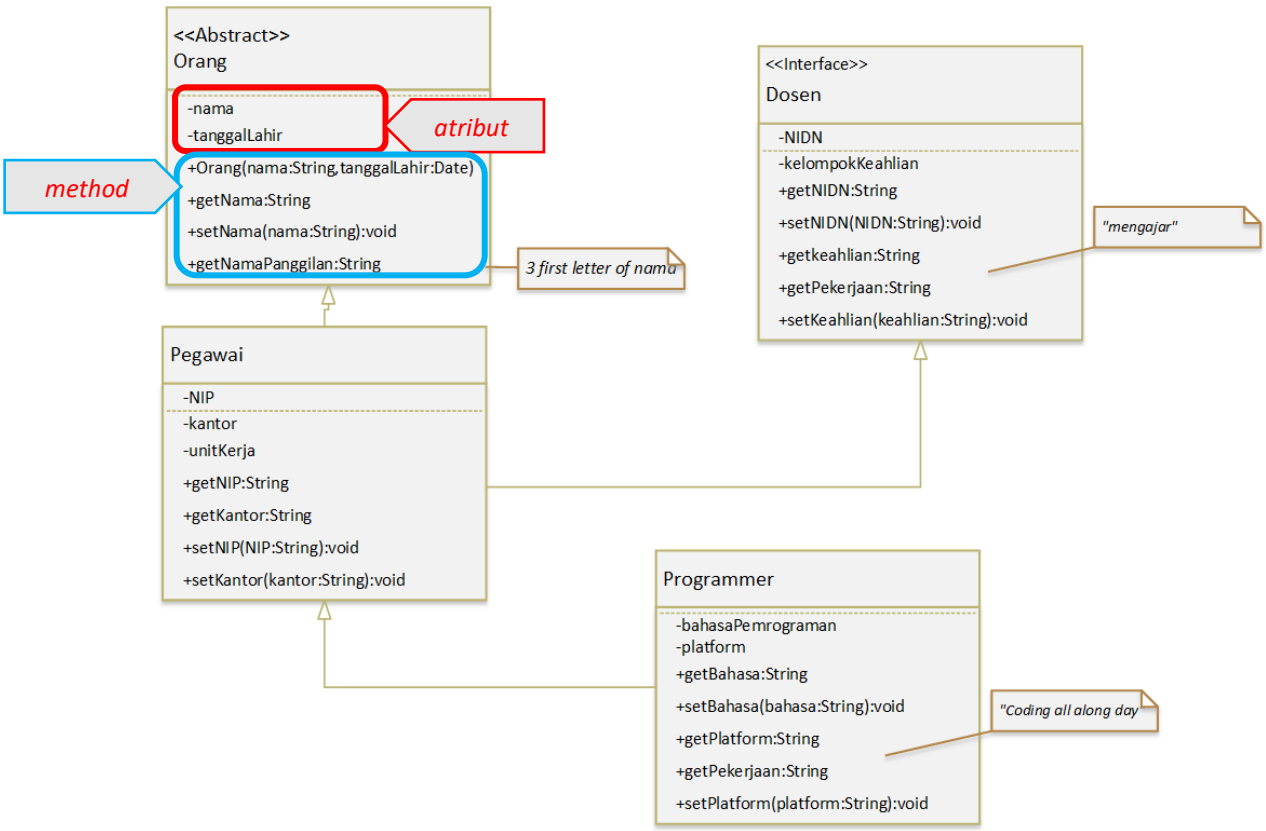
Kegiatan pada modul 5 ini memerlukan material berupa Software IDE Netbeans[5].

5.4. Kegiatan Praktikum

Berikut diberikan diagram kelas. Konversikan ke kode java berdasarkan apa yang sudah dipelajari di modul sebelumnya berdasarkan keterangan yang ada pada diagram berikut.

1) Setiap kelas dalam diagram dikonversi menjadi kelas java.
2) Setiap atribut dalam kelas dikonversi menjadi variabel dalam kelas java tersebut.
3) Behaviour yang ada di setiap kelas dikonversi menjadi method dalam kelas java tersebut dengan parameter, tipe setiap parameter dan tipe keluaran setiap method seperti yang ada di diagram.

[5] https://netbeans.apache.org/download/index.html



5.5. Penugasan

Laporkan hasil praktikum berikut dengan hasil penugasan dan penjelasannya ke Dosen dalam bentuk file pdf dengan format nama *<<nim>>_modul5.*

1) Lengkapi kode semua objek
2) Tangkapan layar hasil kode yang dilengkapi
3) Tangkapan layar hasil running

MODUL 6 PENANGANAN PENGECUALIAN

6.1. Deskripsi Singkat

Dalam modul 6 ini kita akan mempraktekkan cara penanganan pengecualian atau exception handling. Penanganan ini digunakan pada kode yang mungkin menghasilkan error atau kriteria yang tidak sesuai dengan seharusnya. Dengan penanganan ini diharapkan kita bisa menjalankan program tanpa berhenti karena error yang muncul atau menampilkan sesuatu jika kriteria yang tidak sesuai dengan seharusnya.

6.2. Tujuan Praktikum

Setelah praktikum pada modul 6 ini diharapkan mahasiswa mempunyai kompetensi cara penanganan pengecualian atau exception handling pada kode yang mungkin menghasilkan error atau kriteria yang tidak sesuai dengan seharusnya.

6.3. Material Praktikum

Kegiatan pada modul 6 ini memerlukan material berupa Software IDE Netbeans[6].

6.4. Kegiatan Praktikum

Dengan menggunakan kelas Orang, Pegawai, dan Programmer yang sudah dibuat sebelumnya, ubah tipe atribut tanggal lahir menjadi Calendar.

```
private Calendar tanggalLahir;
```

Sesuaikan atribut pada method yang berhubungan dengan atribut tanggal lahir.

```
public Orang(String nama, Calendar tanggalLahir) {
    this.nama = nama;
    this.tanggalLahir = tanggalLahir;
}
```

Kemudian ubah tanggal lahir objek rahma menjadi seperti ini.

```
rahma.setTanggalLahir(new GregorianCalendar(1997,8,31));
```

Kemudian tambahkan kode ini untuk memformat tanggal.

```
SimpleDateFormat sdf = new SimpleDateFormat("yyyy MMM dd");
```

[6] https://netbeans.apache.org/download/index.html

Selanjutnya ubah kode untuk menampilkan orang lutfi dan rahma di Kantor menjadi seperti ini. Perlu diingat bahwa lutfi belum mempunyai tanggal lahir sedangkan rahma sudah mempunyai tanggal lahir.

```
System.out.println("Ada orang:");
System.out.println(lutfi.getNama()+" lahir pada "+sdf.format(lutfi.getTanggalLahir().getTime()));
System.out.println(rahma.getNama()+" lahir pada "+sdf.format(rahma.getTanggalLahir().getTime()));
```

Pada saat program ini dijalankan, maka akan menghasilkan error. Hal ini karena lutfi belum mempunyai nilai alias NULL.

Bagaimana supaya tidak error? Kita tangkap errornya...HAP...

Pertama kita pindahkan format tanggal ke method get di setiap kelas, menjadi seperti ini.

```
public String getTanggalLahir() {
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy MMM dd");
    return sdf.format(tanggalLahir.getTime()).toString();
}
```

Kemudian kita kembalikan kode di Kantor menjadi seperti ini.

```
System.out.println("Ada orang:");
System.out.println(lutfi.getNama()+" lahir pada "+lutfi.getTanggalLahir());
System.out.println(rahma.getNama()+" lahir pada "+rahma.getTanggalLahir());
```

Masih error ya... karena belum ditangkap...selanjutnya kita tambahkan try..catch untuk menangkap error. Jika tanggal lahir belum ada, maka yang ditampilkan adalah ”tidak tersedia”.

```
try {
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy MMM dd");
    return sdf.format(tanggalLahir.getTime()).toString();
} catch (Exception e) {
    return "Belum Tersedia.";
}
```

Maka kalau kita jalankan sudah tidak error.

```
Ada orang:
Lutfi lahir pada Belum Tersedia.
Rahma lahir pada 1997 Oct 01
```

Kita bisa menambahkan blok finally untuk menandakan kode sudah tereksekusi.

```
try {
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy MMM dd");
    return sdf.format(tanggalLahir.getTime()).toString();
} catch (Exception e) {
    return "Belum Tersedia.";
}finally{
    System.out.println("error tertangani");
}
```

selanjutnya kita bisa mencoba memvalidasi isian yang masuk. Misalkan yang tanggal lahirnya kurang tahu tahun 2000 akan disaring. Kita ubah kode menjadi seperti ini.

```
public String getTanggalLahir() {

    try {

       SimpleDateFormat sdf = new SimpleDateFormat("yyyy MMM dd");
       validatetanggallahir();
       return sdf.format(tanggalLahir.getTime());

     } catch (Exception e) {
        return "Belum Tersedia.";
    }finally{
        System.out.println("error tertangani");
    }

}

void validatetanggallahir() throws Exception {
    if(tanggalLahir.get(Calendar.YEAR)<2000) {
        throw new Exception("Dibawah umur");
    }
    else {
        System.out.println("cukup umur");
    }
}
```

Maka ketika Orang dijalankan, akan menghasilkan output seperti ini.

```
Ada orang:
Lutfi lahir pada Belum Tersedia.
Rahma lahir pada Belum Tersedia.
```

Tetapi jika tanggal rahma diubah jadi tahun 2015, maka akan menjadi seperti ini.

```
rahma.setTanggalLahir(new GregorianCalendar(2015,8,31));
```

Dengan output seperti ini.

```
Ada orang:
error tertangani
Lutfi lahir pada Belum Tersedia.
cukup umur
error tertangani
Rahma lahir pada 2015 Oct 01
```

6.5. Penugasan

Laporkan hasil praktikum berikut dengan hasil penugasan dan penjelasannya ke Dosen dalam bentuk file pdf dengan format nama *<<nim>>_modul6.*

1) Lengkapi kode semua objek
2) Tangkapan layar hasil kode yang dilengkapi
3) Tangkapan layar hasil running

MODUL 7 DESIGN PATTERN

7.1. Deskripsi Singkat

Dalam pemrograman berorientasi objek ada *design pattern* yang dipelajari, yaitu Observer Pattern, Decorator Pattern, Factory Pattern, Singleton Pattern. Oleh karena itu pada modul 7 ini kita akan mempelajari bagaimana mengonversi pattern tersebut ke kode.

7.2. Tujuan Praktikum

Setelah praktikum pada modul 7 ini diharapkan mahasiswa mempunyai kompetensi bagaimana mengonversi pattern kelas ke kode.

7.3. Material Praktikum

Kegiatan pada modul 7 ini memerlukan material berupa Software IDE Netbeans[7].

7.4. Kegiatan Praktikum

Berikut diberikan diagram kelas. Konversikan ke kode java berdasarkan apa yang sudah dipelajari di modul sebelumnya berdasarkan keterangan yang ada pada diagram berikut.

A. Observer Pattern

Behaviour Observable dapat kita definisikan dalam sebuah interface. Misalkan kita buat interface dengan nama Observer dan Observable seperti berikut.

```
public interface Observer {
    public void update();
}
```

```
public interface Observable {
    void addObserver(Observer o);
    void removeObserver(Observer o);
    void notifyObserver();

}
```

Mari kita implementasi skenario Rahma mencari buku pink di atas dalam program Java.

[7] https://netbeans.apache.org/download/index.html

```java
15  public class PinkBook implements Observable {
16      private boolean inStock = true;
        private ArrayList<Observer> customers;
18
19      public PinkBook(Boolean inStock) {
20          this.inStock = inStock;
            customers = new ArrayList<Observer>();
22      }
23
24      public boolean isInStock(){
25          return  inStock;
26      }
27
28      public void setInStock(boolean inStock){
29          this.inStock = inStock;
30          if(isInStock()){
31              notifyObserver();
32          }
33      }
34
35      @Override
        public void addObserver(Observer o) {
37          customers.add(o);
38      }
39
40      @Override
        public void removeObserver(Observer o) {
42          customers.remove(o);
43      }
44
45      @Override
        public void notifyObserver() {
47          for(int i=0; i<customers.size(); i++){
48              customers.get(i).update();
49          }
50      }
51  }
```

Kelas PinkBook adalah representasi dari buku yang dicari Rahma. Ia memiliki atribut inStock, artinya stok tersedia. Karena PinkBook merupakan objek yang disubscribe, berarti ia adalah Observable. Oleh karenanya, perlu mengimplementasi interface Observable.

Kemudian, untuk Observer, ia memiliki behaviour 'update'. Artinya melakukan sesuatu ketika di-notify oleh Observable.

Observer dalam hal kasus ini adalah Rahma (Customer). Maka dari itu kita bisa buat representasinya dalam sebuah class yang mengimplementasi interface Observer.

```java
13  public class Customer implements Observer {
14
    private Observable observable;
    private String username;
17
18      public Customer(Observable observable, String username){
19          this.observable = observable;
20          this.username = username;
21      }
22
23      @Override
    public void update() {
25          System.out.println("Buku Pink Tersedia");
26          buyDress();
27      }
28
29
30      private void buyDress(){
31          System.out.println(username +" mendapatkan Buku Pink.");
32      }
33
34      public void unsubscribe(){
35          observable.removeObserver(this);
36      }
37  }
```

Kelas Customer adalah representasi dari customer yang mencari buku pink. Ia memiliki nama, dan dapat melakukan unsubscribe kapan saja. Dalam contoh di atas, ketika PinkBook tersedia, maka method update akan dipanggil. Ketika itu, kita dapat menentukan apa yang dilakukan oleh Observer, misalnya melakukan pembelian buku tersebut. Pada kelas tersebut, constructor dibuat memiliki parameter objek Observable supaya dapat melakukan unsubscribe melalui objek customer/observer.

Untuk selanjutnya kita coba. kita bisa melakukan uji coba dengan membuat beberapa objek customer(Observer) dan mendaftarkannya untuk sebuah PinkDress (Observable).

```java
12  public class ObserverPatternMain {
13      public static void main(String[] args) {
14          PinkBook pinkbook = new PinkBook(true);
15
16          Customer customer1 = new Customer(pinkbook, "Lutfi");
17          pinkbook.addObserver(customer1);
18
19          Customer customer2 = new Customer(pinkbook, "Tuti");
20          pinkbook.addObserver(customer2);
21      }
22  }
```

Implementasi di atas menjelaskan, ada dua customer yang mencari buku pink, Lutfi dan Tuti. Ketika stok buku pink tersedia, ia ingin segera dikabari agar dapat langsung membeli.

Ketika buku pink tersedia.

```
12  public class ObserverPatternMain {
13      public static void main(String[] args) {
14          PinkBook pinkbook = new PinkBook(true);
15
16          Customer customer1 = new Customer(pinkbook, "Lutfi");
17          pinkbook.addObserver(customer1);
18
19          Customer customer2 = new Customer(pinkbook, "Tuti");
20          pinkbook.addObserver(customer2);
21
22          pinkbook.setInStock(true);
23
24      }
25  }
```

Maka, otomatis semua customer akan diberi tahu. Berikut output dari perintah di atas:

```
Notifications | Output - HaloDunia (run)
run:
Buku Pink Tersedia
Lutfi mendapatkan Buku Pink.
Buku Pink Tersedia
Tuti mendapatkan Buku Pink.
BUILD SUCCESSFUL (total time: 0 seconds)
```

B. Decorator Pattern

Misalkan kita mempunyai Pakaian yang bisa dipakai. Pakaian terdiri dari kaos dan celana. Pada dasarnya kaos dan celana tidak berwarna. Namun kita bisa menambahkan fitur warna supaya lebih menarik.

Untuk implementasinya kita buat dulu kelas Pakaian.

```
    public interface Pakaian {
14
        public void pakai();
16
17  }
```

Pakaian ada yang jenisnya kaos,

```
12    public class Kaos implements Pakaian{
13
14        @Override
          public void pakai() {
16            System.out.println("Jenis : Kaos");
17        }
18
19    }
```

Ada juga pakaian yang jenisnya celana.

```
12    public class Celana implements Pakaian{
13
14        @Override
          public void pakai() {
16            System.out.println("Jenis : Celana");
17        }
18    }
```

Kita bisa warnai pakaian tersebut.

```
      public abstract class WarnaiPakaian implements Pakaian {
14        protected Pakaian warnai;
15
16        public WarnaiPakaian(Pakaian warnai){
17            this.warnai = warnai;
18        }
19
20        @Override
          public void pakai() {
22            warnai.pakai();
23        }
24    }
```

Kita bisa warnai dengan warna merah.

```java
12  public class WarnaiMerah extends WarnaiPakaian{
13
14      public WarnaiMerah(Pakaian warnai) {
15          super(warnai);
16      }
17
18      @Override
19      public void pakai(){
20          warnai.pakai();
21          setWarnaPakaian(warnai);
22      }
23
24      private void setWarnaPakaian(Pakaian warnai){
25          System.out.println("Warna Border : Merah");
26      }
27  }
```

Kita bisa cek hasilnya dengan kode berikut.

```java
12  public class DecoratorPatternMain {
13      public static void main(String[] args){
14
15          Pakaian Kaos = new Kaos();
16
17          Pakaian kaosmerah = new WarnaiMerah(new Kaos());
18
19          Pakaian celanamerah = new WarnaiMerah(new Celana());
20
21          System.out.println("Kaos belum diwarnai");
22          Kaos.pakai();
23
24          System.out.println("\nCelana warna merah");
25          celanamerah.pakai();
26
27          System.out.println("\nKaos warna merah");
28          kaosmerah.pakai();
29      }
30  }
```

Yang akan menghasilkan output berikut.

```
Notifications | Output - HaloDunia (run)
run:
Kaos belum diwarnai
Jenis : Kaos

 Celana warna merah
Jenis : Celana
Warna Border : Merah

 Kaos warna merah
Jenis : Kaos
Warna Border : Merah
```

C. Factory Pattern

Misalkan di suatu Institusi membutuhkan banyak pegawai. Pegawai tersebut terdiri pegawai tetap maupun pegawai tidak tetap atau kontrak. Kemudian ada seseorang yang ingin berkerja di institusi tersebut. Orang tersebut kemudian menemui Kepala Bagian Kepegawaian dengan membawa persyaratan yang telah ditentukan. Tetapi, apakah dia akan diteriama atau tidak. Dan jika diterima apakah akan dijadikan sebagai karyawan kontrak atau tetap. Semua tergantung Kepala Bagian Kepegawaian. Kepala Bagian Kepegawaian adalah Factory class dimana ia akan menciptakan beberapa Pegawai tetap dan Pegawai kontrak. Oleh karena itu, pertama kita buat kelas Pegawai.

```
public class Pegawai {
    private String nama;
    private String tipe;
    private String pembayarangaji;

    public void setNama(String nama){
        this.nama = nama;
    }

    public String getNama(){
        return this.nama;
    }

    public void setTipe(String tipe){
        this.tipe = tipe;
    }

    public String getTipe(){
        return this.tipe;
    }

    public void setPembayarangaji(String pembayarangaji) {
        this.pembayarangaji = pembayarangaji;
    }

    public String getPembayarangaji(){
        return this.pembayarangaji;
    }

    @Override
    public String toString(){
        return "nama              :"+this.nama+
                "\nTipe pegawai      :"+this.tipe
                +"\nPembayaran Gaji :"+this.pembayarangaji+"\n";
    }
}
```

Kemudian kita buat kelas untuk pegawai tetap.

```
12  public class PegawaiTetap extends Pegawai {
        public PegawaiTetap (String nama) {
            setNama(nama);
            setTipe("Permanen");
            setPembayarangaji("Perbulan");
17      }
18  }
```

Selanjutnya kita buat kelas untuk pegawai kontrak.

```
12  public class PegawaiKontrak extends Pegawai{
13      public PegawaiKontrak (String nama){
            setNama(nama);
            setTipe("Kontrak");
            setPembayarangaji("Perjam");
17      }
18  }
```

Lalu kita buat kelas Factory untuk Pegawai yang mendaftar.

```
12  public class PegawaiFactory {
13
14      public Pegawai buatPegawai(String nama, String tipe){
15          switch (tipe) {
16              case "tetap":
17                  return new PegawaiTetap(nama);
18              case "kontrak":
19                  return new PegawaiKontrak(nama);
20              default:
21                  return null;
22          }
23      }
24  }
```

Untuk melihat hasilkan, kita uji coba dengan membuat kode berikut.

```
12  public class FactoryPatternMain {
13      public static void main(String[] args) {
14          PegawaiFactory factory = new PegawaiFactory();
15          System.out.println(factory.buatPegawai("Lutfi", "tetap").toString());
16
17          System.out.println(factory.buatPegawai("Dani", "kontrak").toString());
18      }
19  }
```

Output yang dihasilkan adalah sebagai berikut.

```
run:
nama              :Lutfi
Tipe pegawai      :Permanen
Pembayaran Gaji :Perbulan

nama              :Dani
Tipe pegawai      :Kontrak
Pembayaran Gaji :Perjam
```

D. Singleton Pattern

Akan dibahas pada pertemuan selanjutnya.

7.5. Penugasan

Laporkan hasil praktikum berikut dengan hasil penugasan dan penjelasannya ke Dosen dalam bentuk file pdf dengan format nama *<<nim>>_modul7.*

1) Lengkapi kode semua objek
2) Tangkapan layar hasil kode yang dilengkapi
3) Tangkapan layar hasil running

MODUL 8 GRAPHICAL USER INTERFACE BAGIAN 1

8.1. Deskripsi Singkat

Graphical User Interface merupakan sistem komponen visual interaktif untuk software komputer. GUI memungkinkan para pengguna atau user mampu berinteraksi dengan perangkat komputer atau aplikasi melalui berbagai macam alat input seperti mouse, keyboard, touchscreen dan sebagainya. Pada modul 8 ini akan diperkenalkan mengenai pemrograman GUI pada Java.

8.2. Tujuan Praktikum

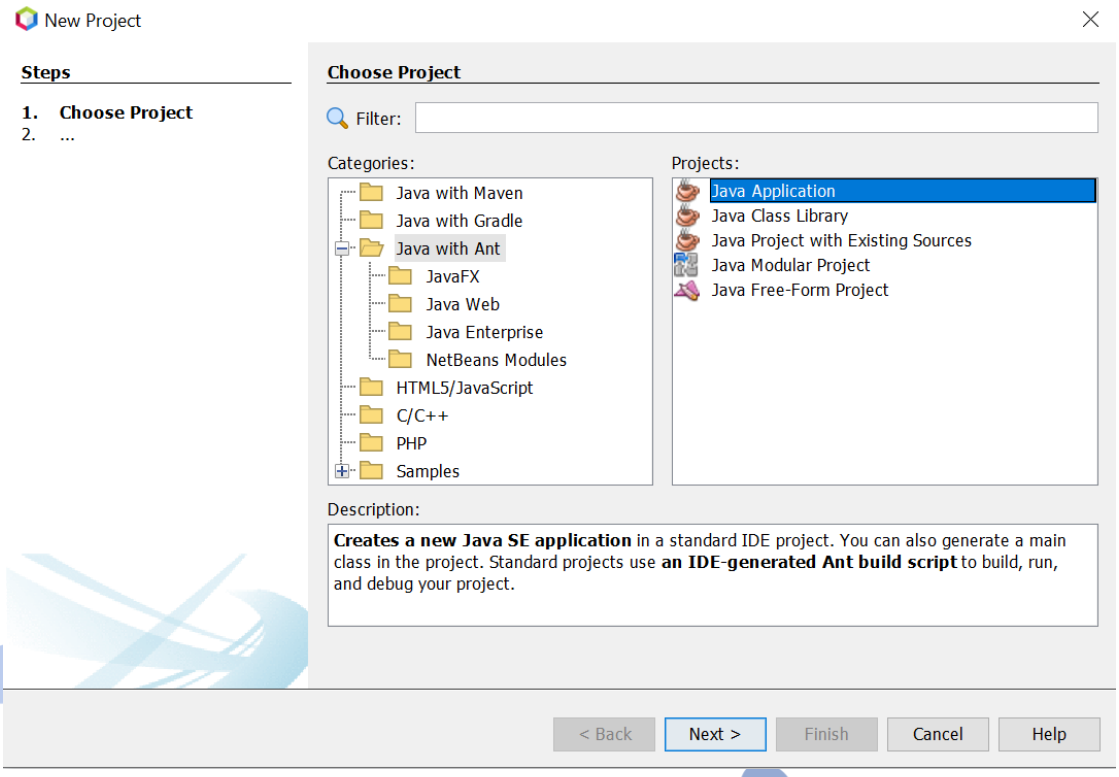
Setelah praktikum pada modul 8 ini diharapkan mahasiswa memahami berbagai macam komponen GUI yang bisa digunakan pada pemrograman Java.

8.3. Material Praktikum

Kegiatan pada modul 8 ini memerlukan material berupa Software IDE Netbeans[8].

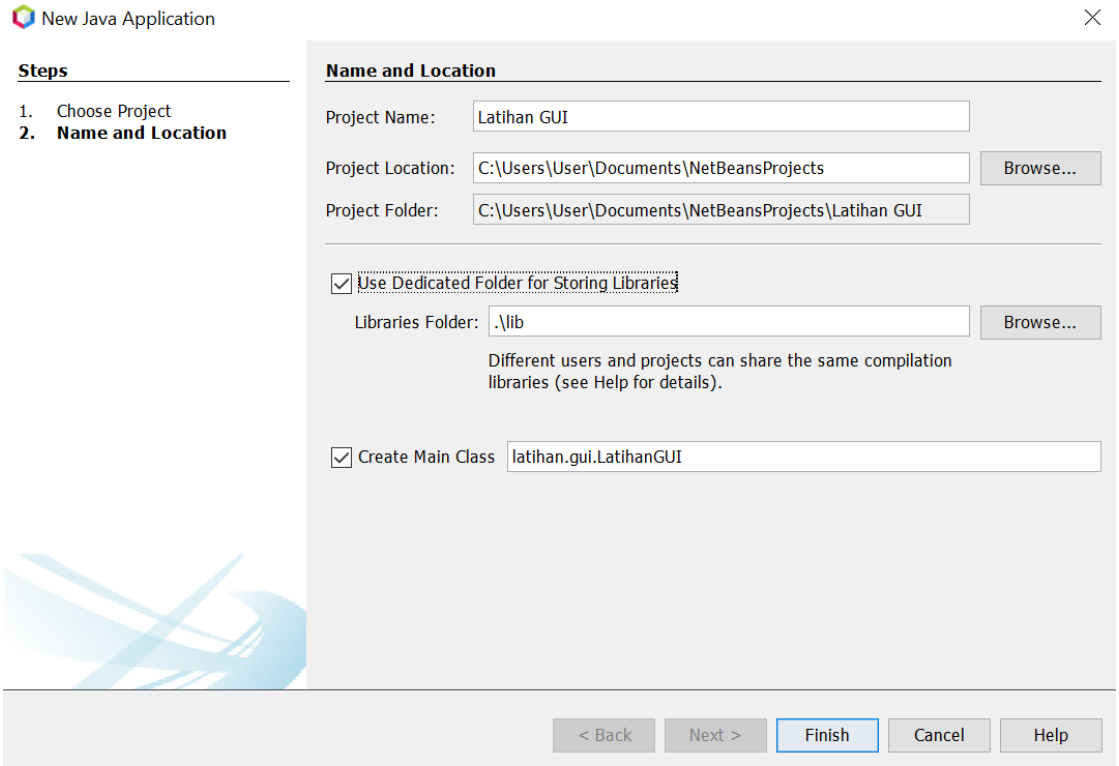
8.4. Kegiatan Praktikum

Buka software Netbeans dan buat project baru dengan Java with Ant

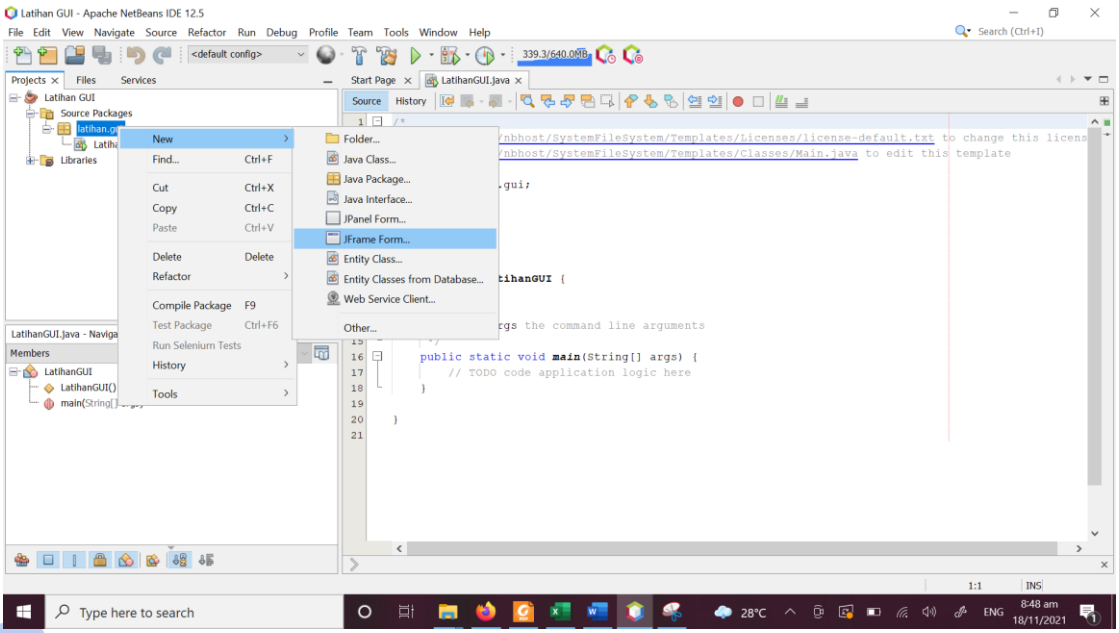


[8] https://netbeans.apache.org/download/index.html

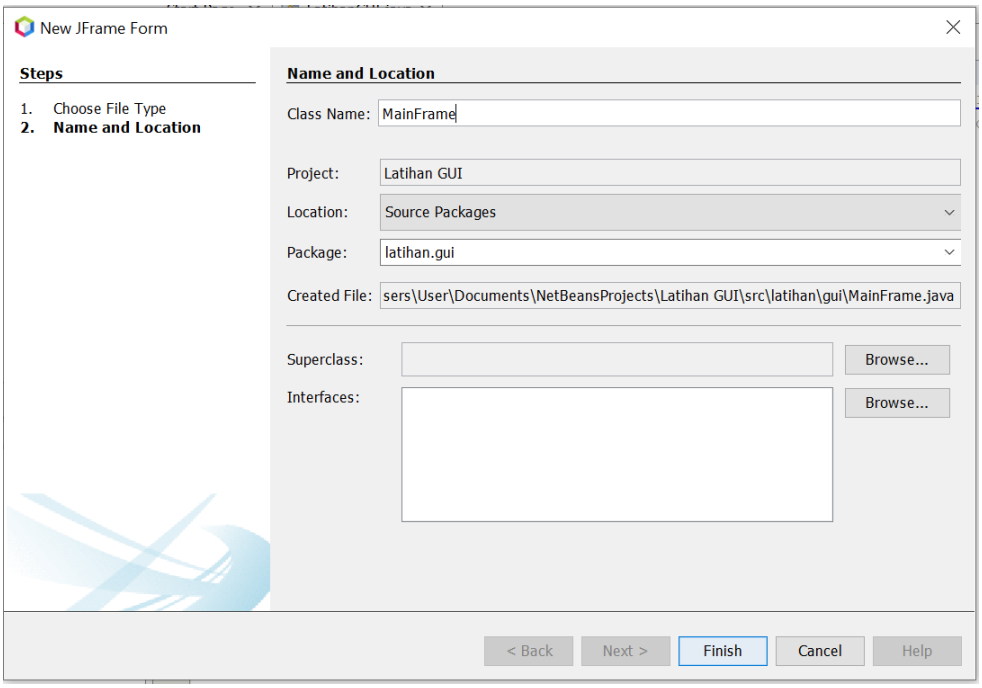
Kemudian klik Next dan isi Project Name dengan Latihan GUI. Kemudian klik Finish



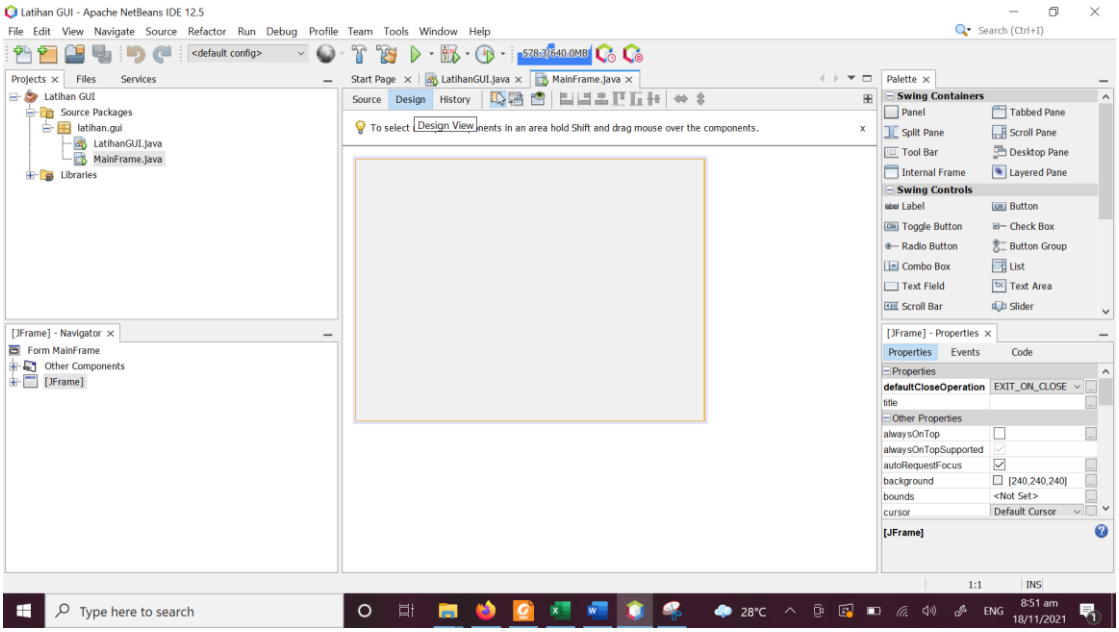
Kemudian pada jendela project pada package latihan.gui klik kanan, New -> JFrame Form.



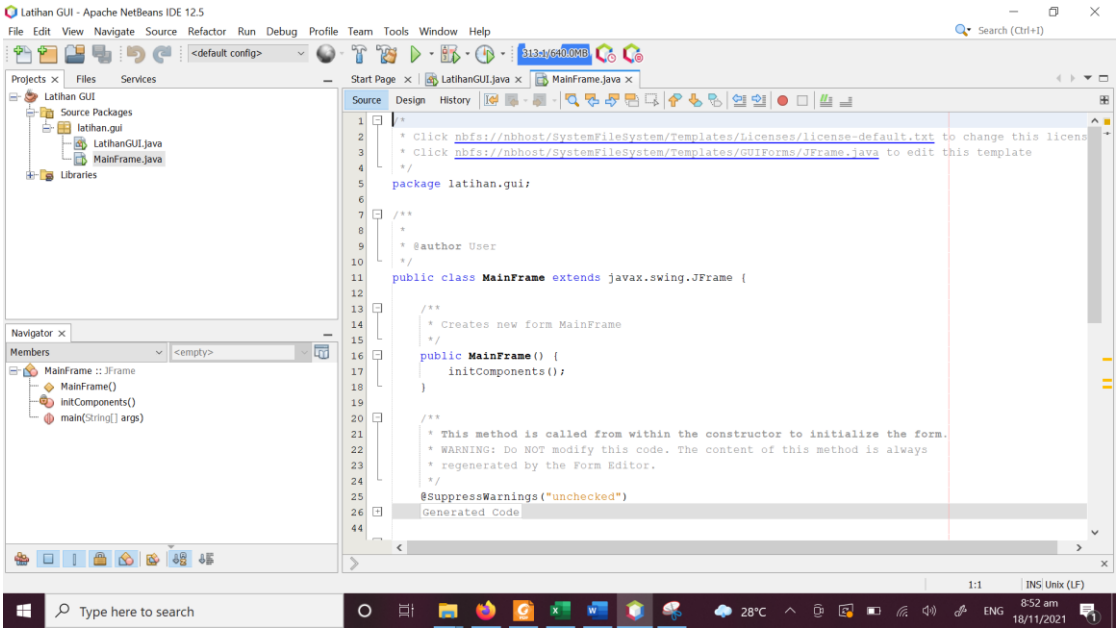
Berikan Class Name dengan nama MainFrame kemudian klik Finish.



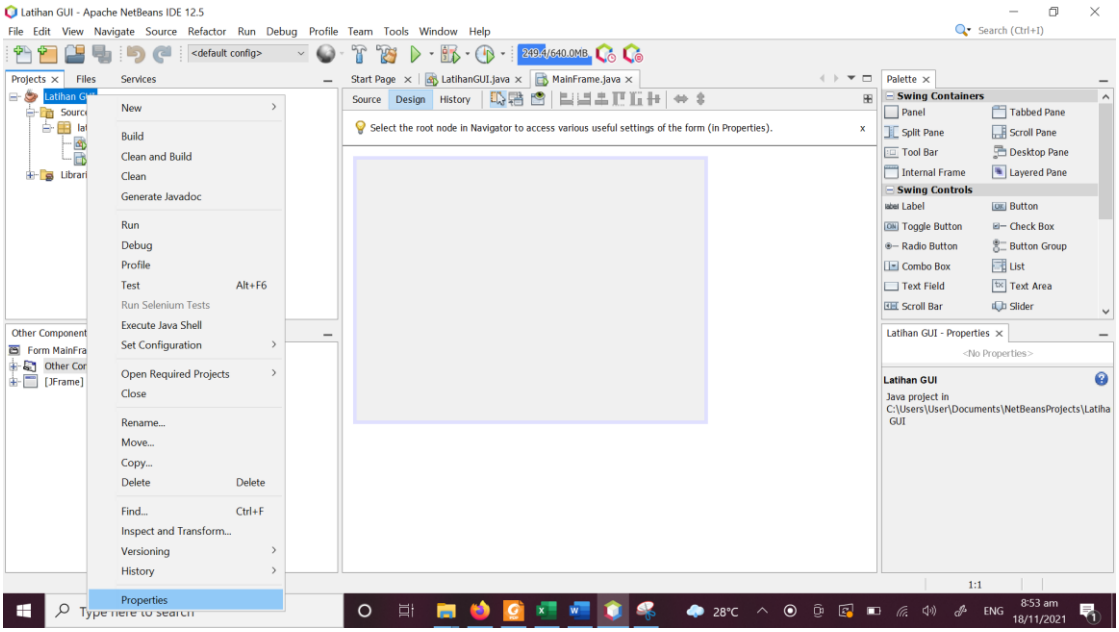
Pada tab Design kita dapat mendesain komponen-komponen GUI dari aplikasi kita. Ada label, button, check box, radio button dan lain sebagainya.

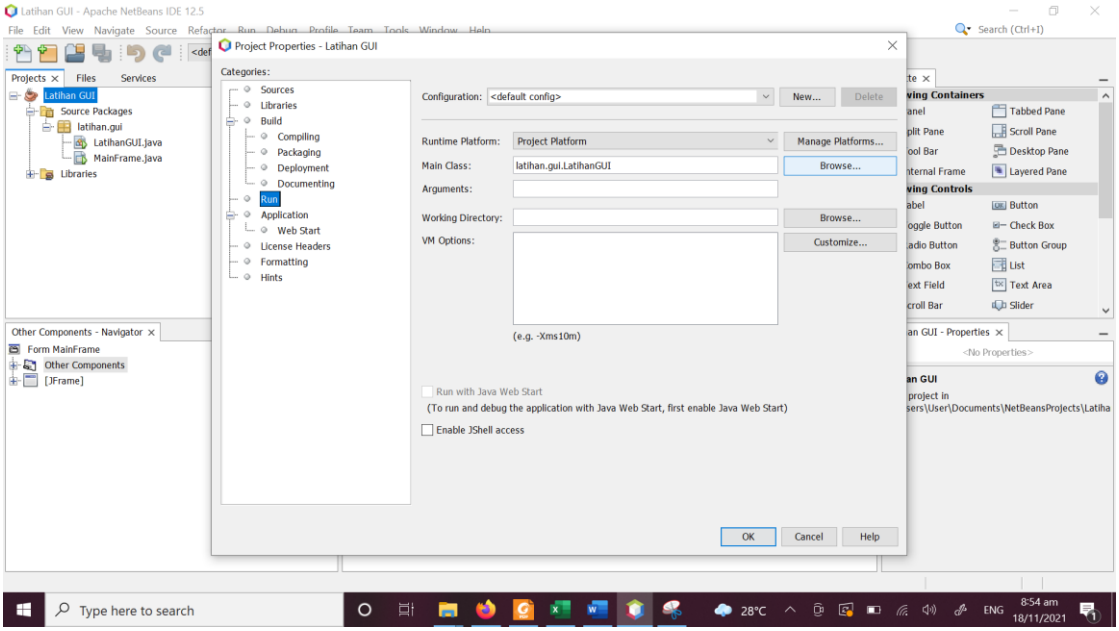


Untuk melihat kode java dari frame tersebut maka bisa kita lihat di tab source

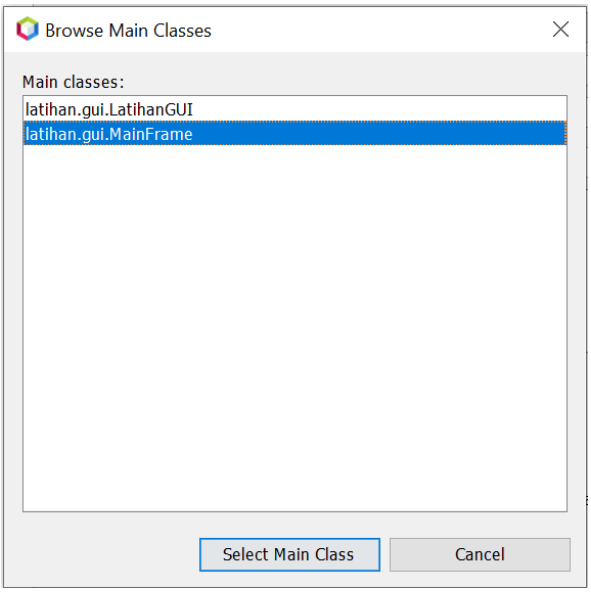


Sampai di sini, jika kita run project, maka tidak akan muncul apa-apa karena yang dijalankan adalah LatihanGUI.java. Untuk mengubah supaya MainFrame tampil setelah dirun, lakukan klik kanan pada project Latihan GUI, buka properties Project, ubah main Class menjadi MainFrame

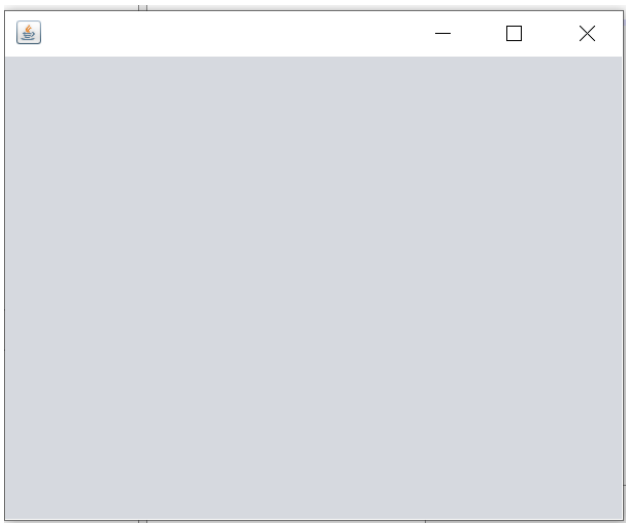




Klik Browse pada Main Class Kemudian pilih latihan.gui.MainFrame sebagai main class dan klik OK



Sekarang ketika aplikasi dijalankan maka akan muncul sebuah jendela kosong

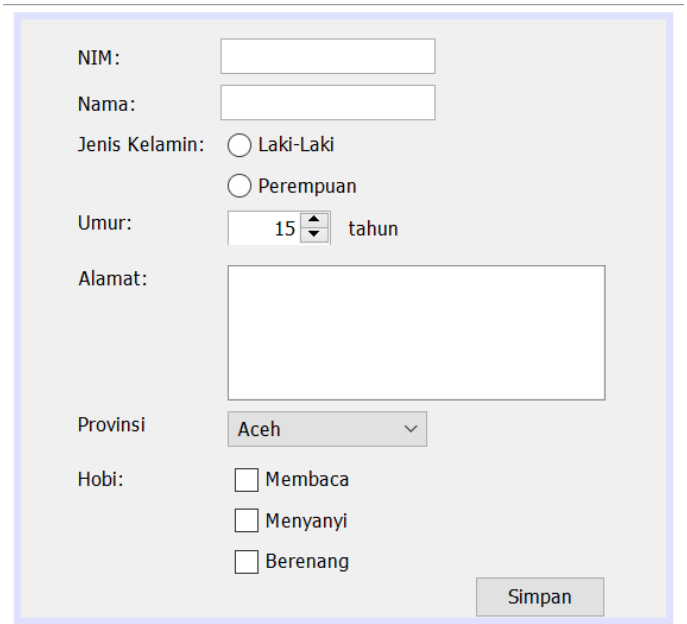


Untuk mengakhiri program, close window tersebut.

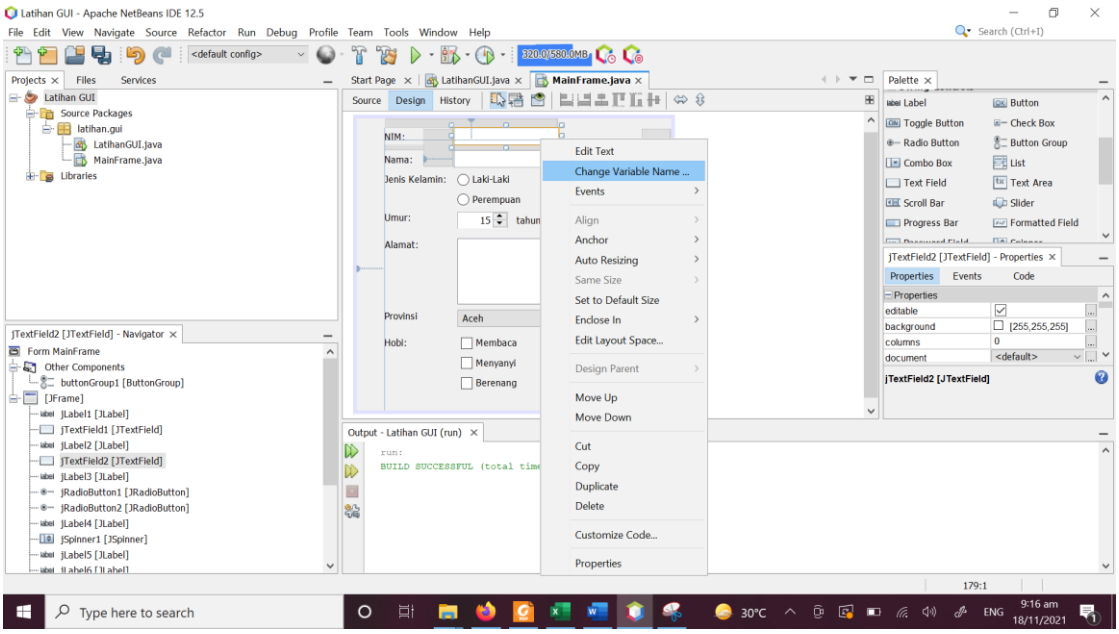
Jika kita melihat pada jendela properties di bagian kanan, maka kita bisa mengubah-ubah properties dari JFrame seperti menambahkan title windows, default close operation, always on top dan lain-lain.

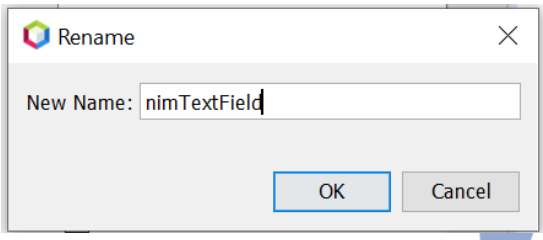
Kemudian anda diminta untuk mendesain frame seperti di bawah ini dengan ketentuan:

1) NIM(text field)
2) Nama (text field)
3) Jenis Kelamin (radio button)
4) Umur (spinner)
5) Alamat (Text Area)
6) Provinsi (combo box)
7) Hobi (check box)

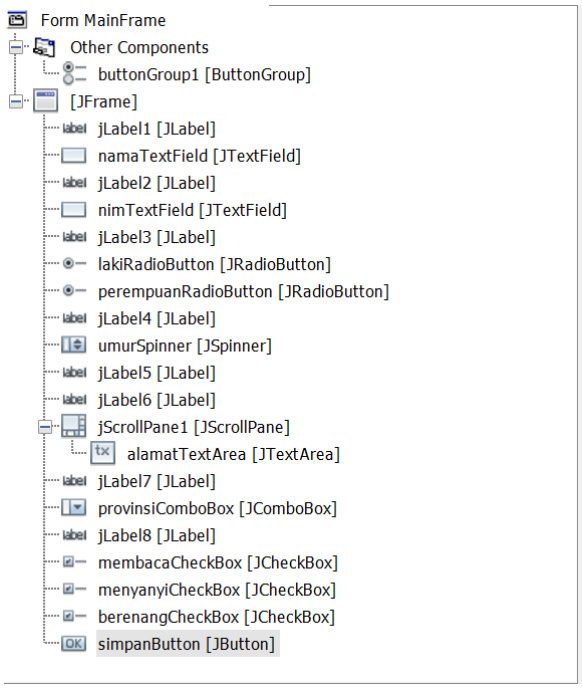


Ubah nama-nama komponen supaya mudah diakses melalui klik kanan, change variable name. Format nama komponen yang disarankan adalah nama_jeniskomponen

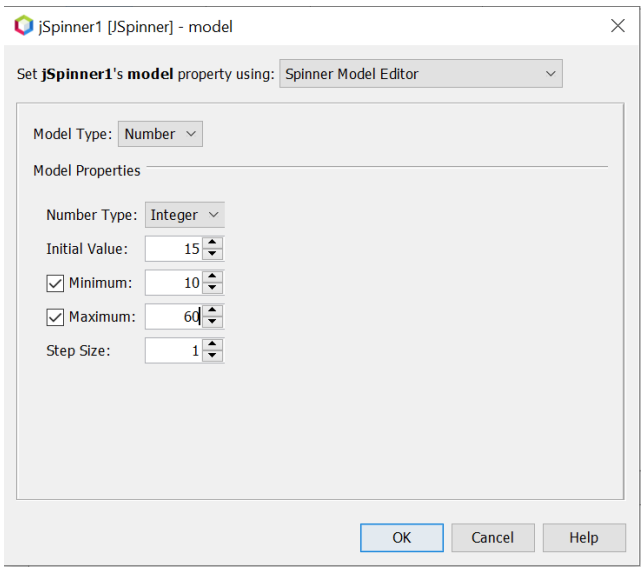




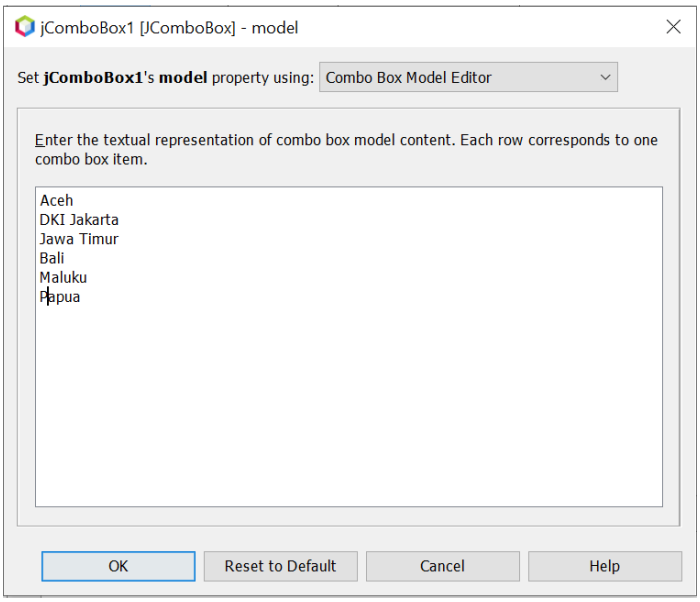
Lakukan semua perubahan nama variabel sehingga pada jendla navigator akan tampil seperti ini:



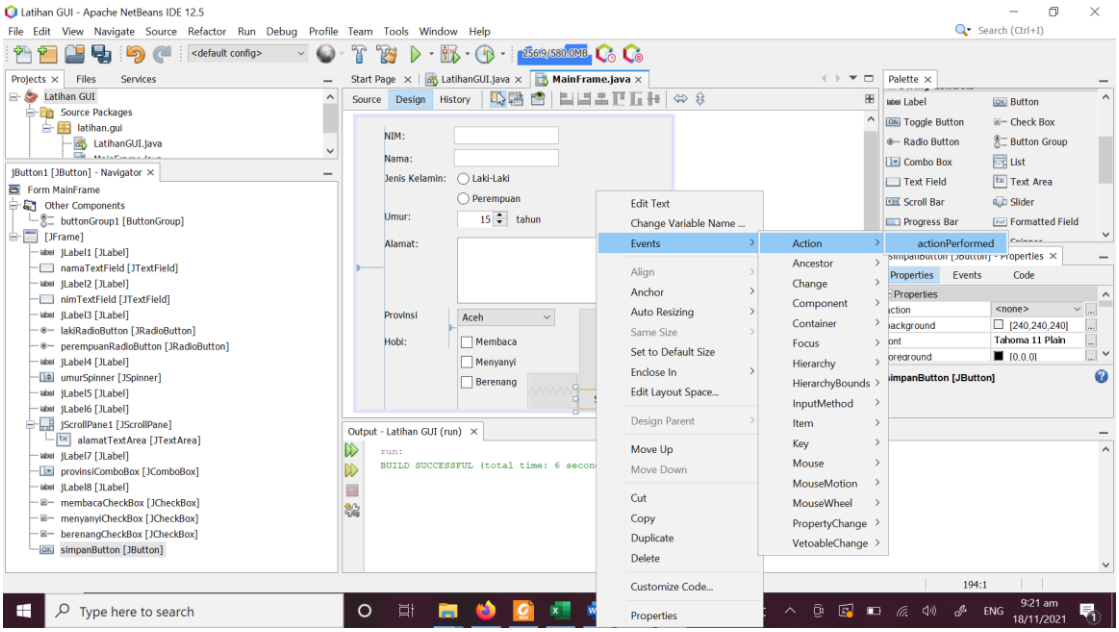
Untuk spinner, anda bisa mengubah model pada jendela properties menjadi seperti berikut ini:



Untuk combo box, anda dapat mengubah model pada jendela properties menjadi seperti berikut:



Untuk menambahkan event pada buttonSimpan, bisa dengan cara klik dua kali tombol tersebut atau dengan cara klik kanan events-> action-> actionPerformed

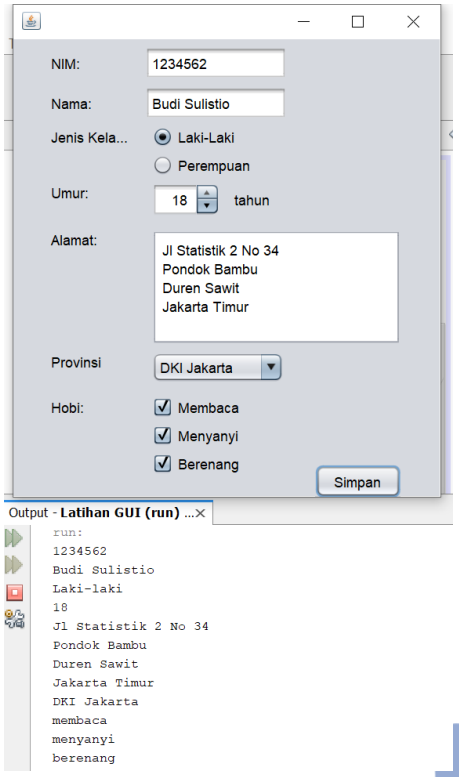


Tambahkan kode berikut pada event simpanButtonActionPerformed

```
private void
simpanButtonActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        System.out.println(nimTextField.getText());
        System.out.println(namaTextField.getText());
        if(lakiRadioButton.isSelected()){
```

```
            System.out.println("Laki-laki");
        }
        if(perempuanRadioButton.isSelected()){
            System.out.println("Perempuan");
        }
        int umur = (Integer) umurSpinner.getValue();
        System.out.println(umur);
        System.out.println(alamatTextArea.getText());
        System.out.println(provinsiComboBox.getSelectedItem());
        if(membacaCheckBox.isSelected()){
            System.out.println("membaca");
        }
        if(menyanyiCheckBox.isSelected()){
            System.out.println("menyanyi");
        }
        if(berenangCheckBox.isSelected()){
            System.out.println("berenang");
        }
}
```

Jalankan Program, kemudian isikan beberapa data dan klik tombol simpan, maka pada jendela output akan tampil apa yang sudah kita tuliskan



8.5. Penugasan

Laporkan hasil praktikum berikut dengan hasil penugasan dan penjelasannya ke Dosen dalam bentuk file pdf dengan format nama *<<nim>>_modul8.*

1) Lengkapi kode semua objek
2) Tangkapan layar hasil kode yang dilengkapi
3) Tangkapan layar hasil running

MODUL 9 GRAPHICAL USER INTERFACE BAGIAN 2

9.1. Deskripsi Singkat

Graphical User Interface merupakan sistem komponen visual interaktif untuk software komputer. GUI memungkinkan para pengguna atau user mampu berinteraksi dengan perangkat komputer atau aplikasi melalui berbagai macam alat input seperti mouse, keyboard, touchscreen dan sebagainya. Pada modul 9 ini akan dilanjutkan perkenalan mengenai pemrograman GUI pada Java.

9.2. Tujuan Praktikum

Setelah praktikum pada modul 9 ini mahasiswa diharapkan lebih menguasai pemrograman GUI pada java

9.3. Material Praktikum

Kegiatan pada modul 9 ini memerlukan material berupa Software IDE Netbeans[9].

9.4. Kegiatan Praktikum

Melanjutkan project pada modul 8, tambahkan kelas Mahasiswa.java dan Database.java sebagai berikut:

Kelas Database.java

```
package latihan.gui;

import java.util.ArrayList;
import java.util.List;

/**
 *
 * @author User
 */
public class Database {
    public static Database instance;
    private ArrayList<Mahasiswa> data = new ArrayList<>();
    private Database(){
    }
```

[9] https://netbeans.apache.org/download/index.html

```
    public static synchronized Database getInstance(){
        if(instance == null){
            instance = new Database();
        }
        return instance;
    }

    public void insertMahasiswa(Mahasiswa mahasiswa){
        data.add(mahasiswa);
    }

    public List<Mahasiswa> getListMahasiswa(){
        return data;
    }
}
```

Kelas **Mahasiswa.java**

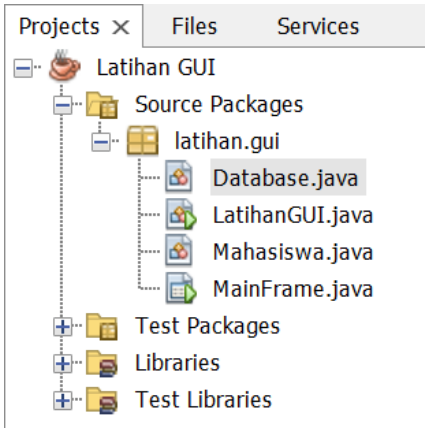
```
package latihan.gui;

import java.util.ArrayList;

/**
 *
 * @author User
 */
public class Mahasiswa {
    private String nim,nama,jenisKelamin;
    private int umur;
    private String alamat, provinsi;
    private ArrayList<String> hobi;

    /**
     * @return the nim
     */
    public String getNim() {
        return nim;
```

```
    }

    /**
     * @param nim the nim to set
     */
    public void setNim(String nim) {
        this.nim = nim;
    }

    /**
     * @return the nama
     */
    public String getNama() {
        return nama;
    }

    /**
     * @param nama the nama to set
     */
    public void setNama(String nama) {
        this.nama = nama;
    }

    /**
     * @return the jenisKelamin
     */
    public String getJenisKelamin() {
        return jenisKelamin;
    }

    /**
     * @param jenisKelamin the jenisKelamin to set
     */
    public void setJenisKelamin(String jenisKelamin) {
        this.jenisKelamin = jenisKelamin;
    }

    /**
```

```
     * @return the umur
     */
    public int getUmur() {
        return umur;
    }

    /**
     * @param umur the umur to set
     */
    public void setUmur(int umur) {
        this.umur = umur;
    }

    /**
     * @return the alamat
     */
    public String getAlamat() {
        return alamat;
    }

    /**
     * @param alamat the alamat to set
     */
    public void setAlamat(String alamat) {
        this.alamat = alamat;
    }

    /**
     * @return the provinsi
     */
    public String getProvinsi() {
        return provinsi;
    }

    /**
     * @param provinsi the provinsi to set
     */
    public void setProvinsi(String provinsi) {
```

```
        this.provinsi = provinsi;
    }

    /**
     * @return the hobi
     */
    public ArrayList<String> getHobi() {
        return hobi;
    }

    /**
     * @param hobi the hobi to set
     */
    public void setHobi(ArrayList<String> hobi) {
        this.hobi = hobi;
    }
}
```

Sehingga pada jendela project akan seperti ini:

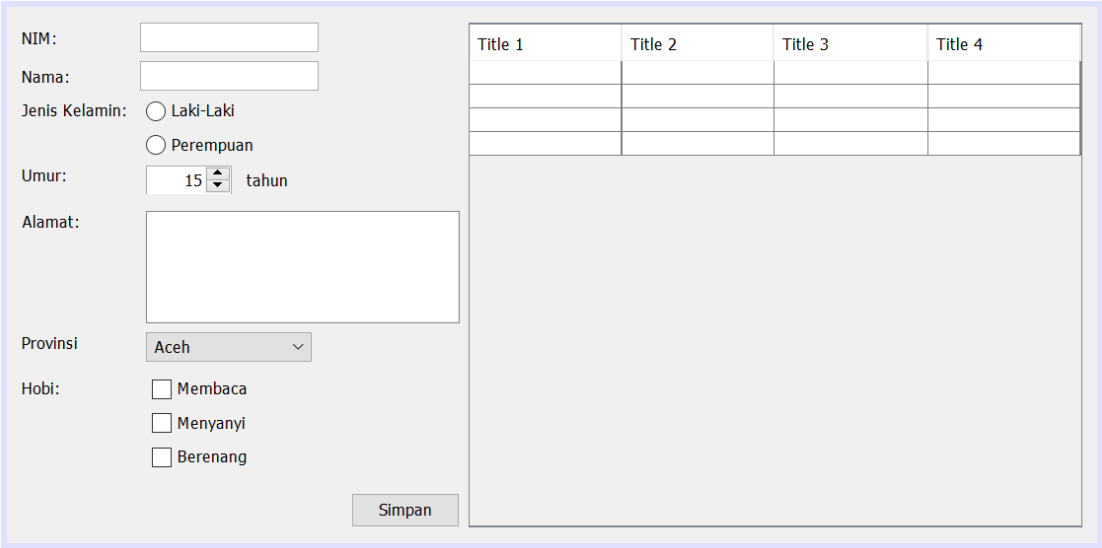


Kemudian ubah kode simpanButtonActionPerformed menjadi seperti berikut ini

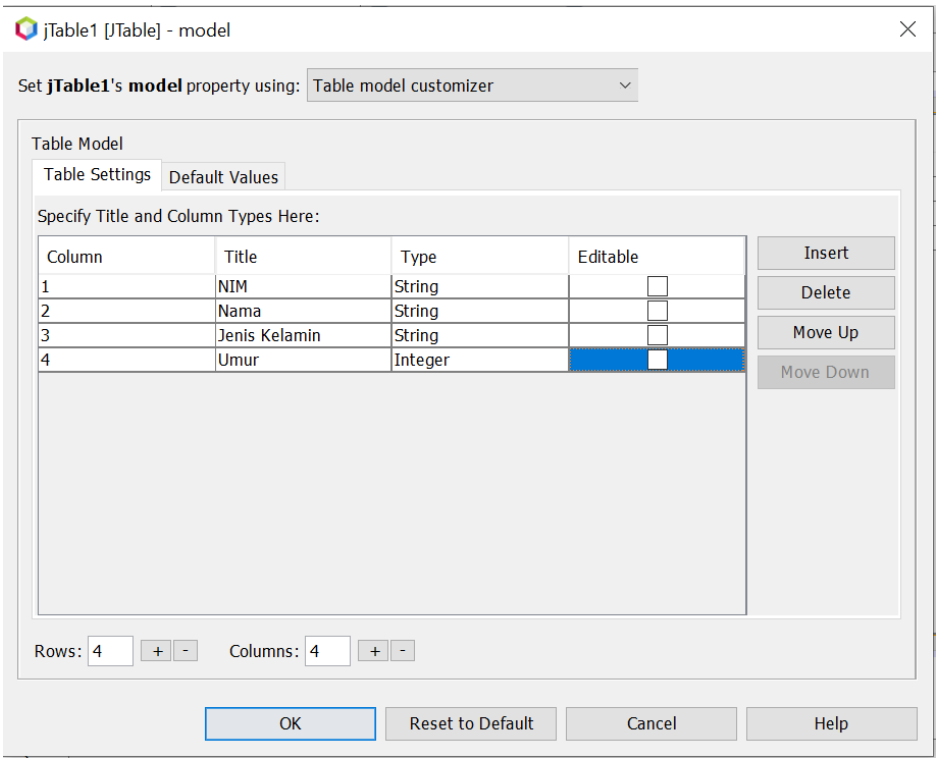
```
private void
simpanButtonActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        Mahasiswa mhs = new Mahasiswa();
        mhs.setNim(nimTextField.getText());
        mhs.setNama(namaTextField.getText());
        if(lakiRadioButton.isSelected()){
```

```
            mhs.setJenisKelamin("Laki-laki");
        }
        if(perempuanRadioButton.isSelected()){
            mhs.setJenisKelamin("Perempuan");
        }
        int umur = (Integer) umurSpinner.getValue();
        mhs.setUmur(umur);
        mhs.setAlamat(alamatTextArea.getText());

mhs.setProvinsi(provinsiComboBox.getSelectedItem().toString());
        ArrayList<String> hobiList = new ArrayList<>();
        if(membacaCheckBox.isSelected()){
            hobiList.add("membaca");
        }
        if(menyanyiCheckBox.isSelected()){
            hobiList.add("menyanyi");
        }
        if(berenangCheckBox.isSelected()){
            hobiList.add("berenang");
        }
        mhs.setHobi(hobiList);
        Database.getInstance().insertMahasiswa(mhs);
    }
```

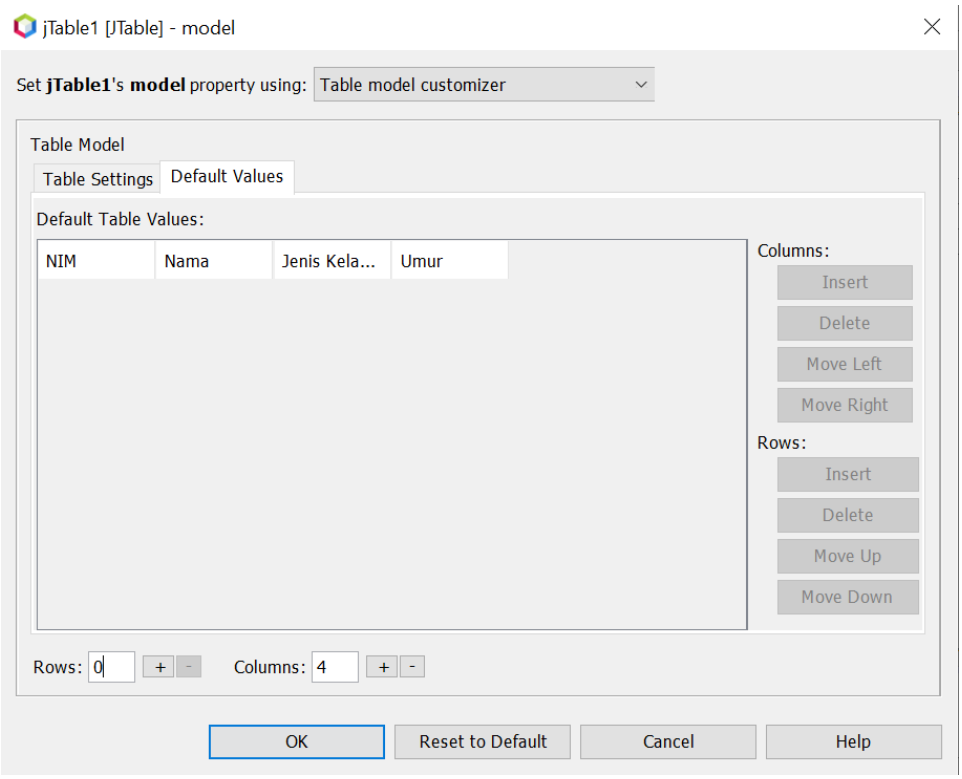
Selanjutnya untuk menampilkan hasil entri data mahasiswa, kita akan menambahkan komponen tabel pada tampilan GUI. Drag and drop komponen tabel sehinggamenjadi seperti berikut ini



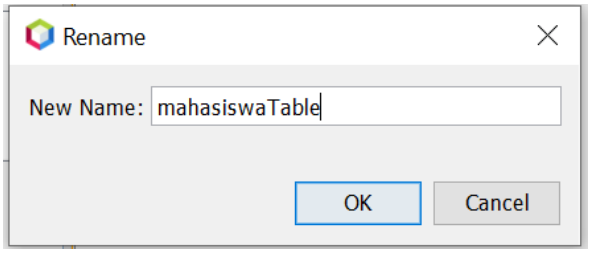
Pada properties dari tabel, klik dan ubah model. Tidak semua atribut mahasiswa penting untuk ditapilkan di tabel jadi akan kita pilih beberapa saja. Ubah moel tabel menjadi seperti di bawah ini:



Pada tab default Values, set rows nya menjadi 0



Ubah variabel name dari tabel menjadi mahasiswaTable

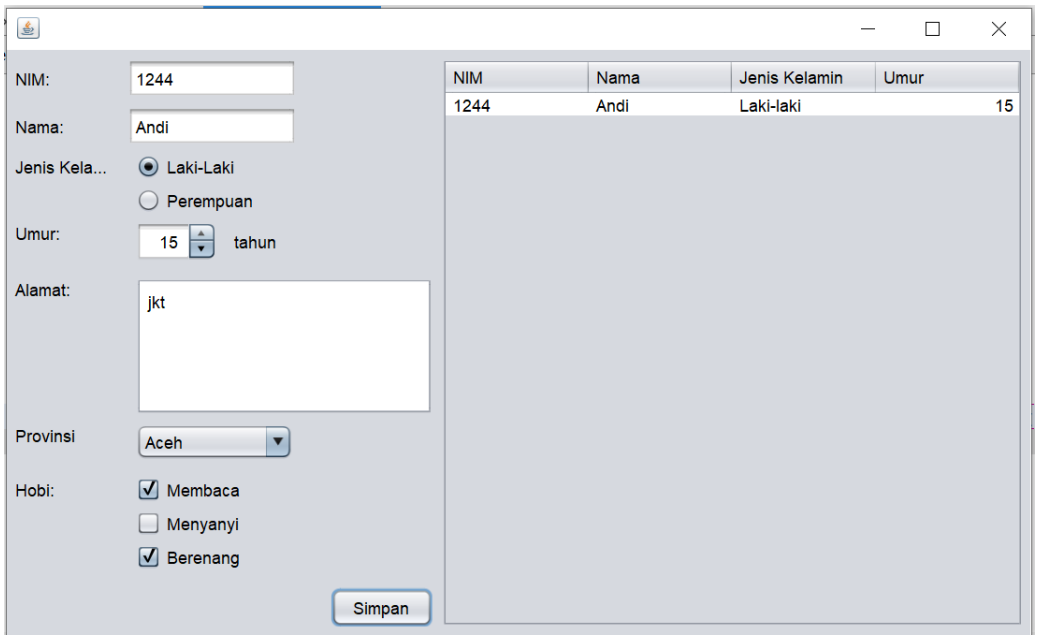


Ubahlah kode pada simpanButtonActionPerformed menjadi seperti berikut ini:

```
        private void
simpanButtonActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        Mahasiswa mhs = new Mahasiswa();
        mhs.setNim(nimTextField.getText());
        mhs.setNama(namaTextField.getText());
        if(lakiRadioButton.isSelected()){
            mhs.setJenisKelamin("Laki-laki");
        }
        if(perempuanRadioButton.isSelected()){
            mhs.setJenisKelamin("Perempuan");
        }
```

```
        int umur = (Integer) umurSpinner.getValue();
        mhs.setUmur(umur);
        mhs.setAlamat(alamatTextArea.getText());

mhs.setProvinsi(provinsiComboBox.getSelectedItem().toString());
        ArrayList<String> hobiList = new ArrayList<>();
        if(membacaCheckBox.isSelected()){
            hobiList.add("membaca");
        }
        if(menyanyiCheckBox.isSelected()){
            hobiList.add("menyanyi");
        }
        if(berenangCheckBox.isSelected()){
            hobiList.add("berenang");
        }
        mhs.setHobi(hobiList);
        Database.getInstance().insertMahasiswa(mhs);


        DefaultTableModel dtm = (DefaultTableModel)
mahasiswaTable.getModel();
        dtm.addRow(new
Object[]{mhs.getNim(),mhs.getNama(),mhs.getJenisKelamin(),mhs.getU
mur()});
    }
```

Ketika program dijalankan dan tombol simpan diklik, maka hasil entri akan disimpan pada tabel.



Sampai di sini, tampilan pada tabel bisa jadi tidak konsisten dengan data sourcenya. Untuk mengkonsistenkan tampilan tabel dan data source, maka setelah melakukan simpan data, kita perlu melakukan refresh tabel tersebut.

Untuk melakukan refresh atau load ulang tambahkan method berikut di source

```
private void loadTableData(){
        DefaultTableModel dtm = (DefaultTableModel)
mahasiswaTable.getModel();
        //refresh tabel
        while(dtm.getRowCount()>0){
            dtm.removeRow(0);
        }
        //isi tabel
        for(Mahasiswa mhs:
Database.getInstance().getListMahasiswa()){
        dtm.addRow(new
Object[]{mhs.getNim(),mhs.getNama(),mhs.getJenisKelamin(),mhs.getU
mur()});
        }     }
```

Untuk membersihkan form entri setelah dilakukan simpan data, tambahkan method berikut ini

```
private void clearForm(){
        nimTextField.setText("");
```

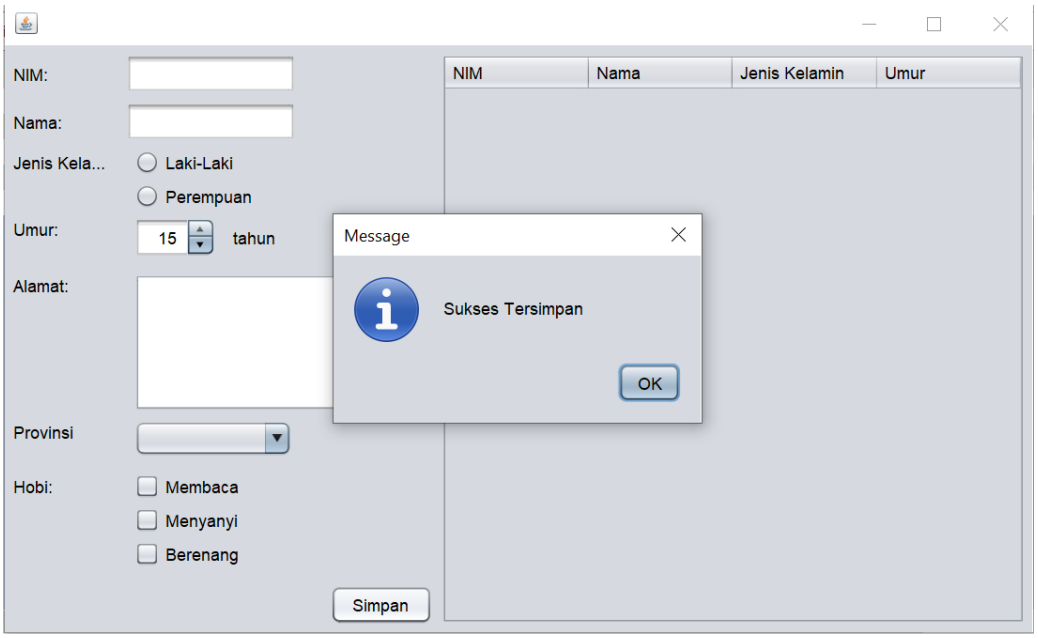
```
        namaTextField.setText("");
        buttonGroup1.clearSelection();
        umurSpinner.setValue(15);
        alamatTextArea.setText("");
        provinsiComboBox.setSelectedIndex(-1);
        membacaCheckBox.setSelected(false);
        menyanyiCheckBox.setSelected(false);
        berenangCheckBox.setSelected(false);
    }
```

Kemudian untuk menambahkan jendela notifikasi, ubah kode simpanButtonActionPerformed menjadi berikut ini:

```
    private void
simpanButtonActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        Mahasiswa mhs = new Mahasiswa();
        mhs.setNim(nimTextField.getText());
        mhs.setNama(namaTextField.getText());
        if(lakiRadioButton.isSelected()){
            mhs.setJenisKelamin("Laki-laki");
        }
        if(perempuanRadioButton.isSelected()){
            mhs.setJenisKelamin("Perempuan");
        }
        int umur = (Integer) umurSpinner.getValue();
        mhs.setUmur(umur);
        mhs.setAlamat(alamatTextArea.getText());

mhs.setProvinsi(provinsiComboBox.getSelectedItem().toString());
        ArrayList<String> hobiList = new ArrayList<>();
        if(membacaCheckBox.isSelected()){
            hobiList.add("membaca");
        }
        if(menyanyiCheckBox.isSelected()){
            hobiList.add("menyanyi");
        }
        if(berenangCheckBox.isSelected()){
            hobiList.add("berenang");
```

```
        }

        Database.getInstance().insertMahasiswa(mhs);
        clearForm();
        JOptionPane.showMessageDialog(this, "Sukses Tersimpan");
        loadTableData();
    }
```

Ketika di-run maka akan muncul jendela sukses tersimpan

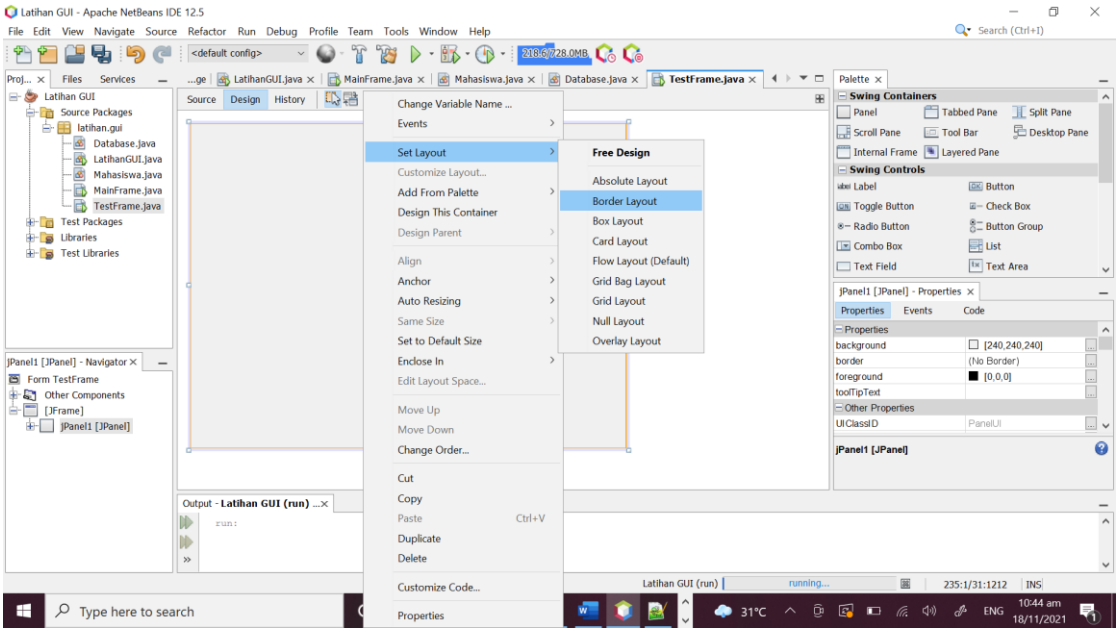


Pada pembahasan berikutnya kita akan membahas tentang layout pada Netbeans. Terdapat berbagai macam layout seperti BorderLayout, BoxLayout, dan lan-lain yang bisa anda pelajari di https://docs.oracle.com/javase/tutorial/uiswing/layout/visual.html

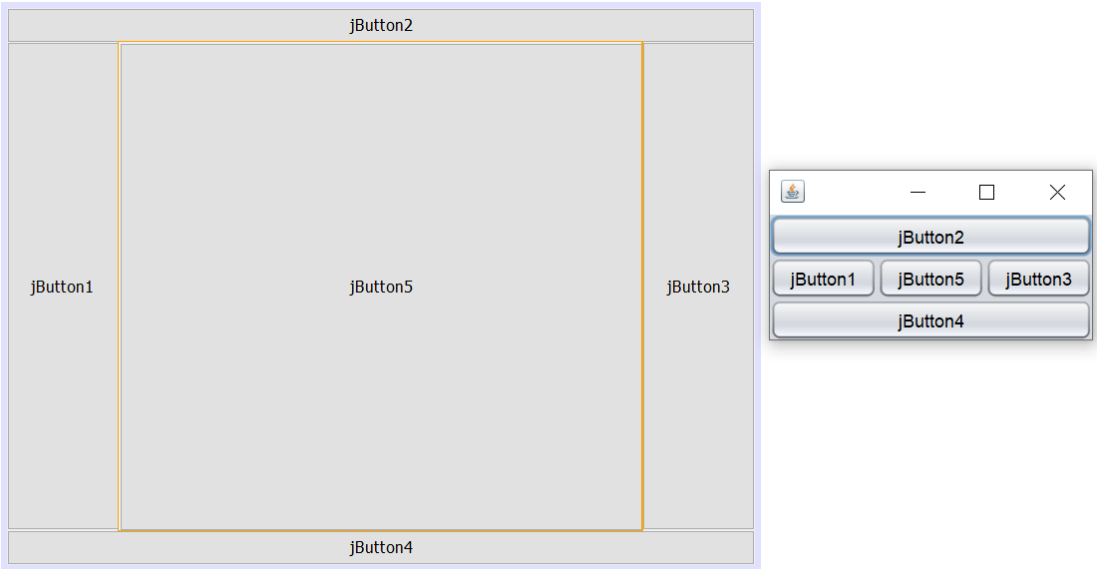
Pengaturan layout ini penting suaya kita bisa mengatur perilaku window ketika dikecilkan, dibesarkan sehingga window tetap berperilaku sesuai dengan yang kita inginkan.

Tambahkan frame baru pada project dengan nama TestFrame. Tambahkan panel pada TestFrame

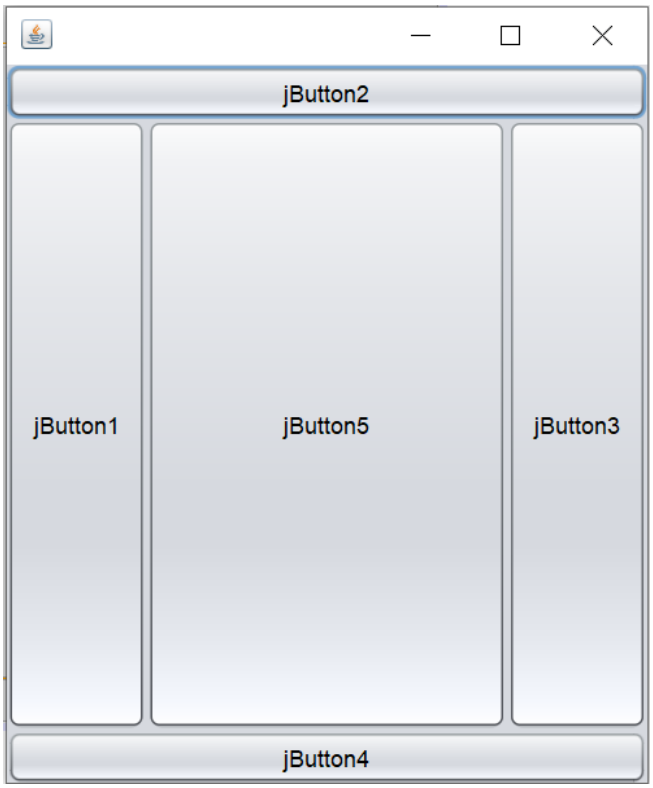
Untuk mengatur layout kita bisa melakuka klik kanan, setlayout dan pilih layout yang diinginkan.



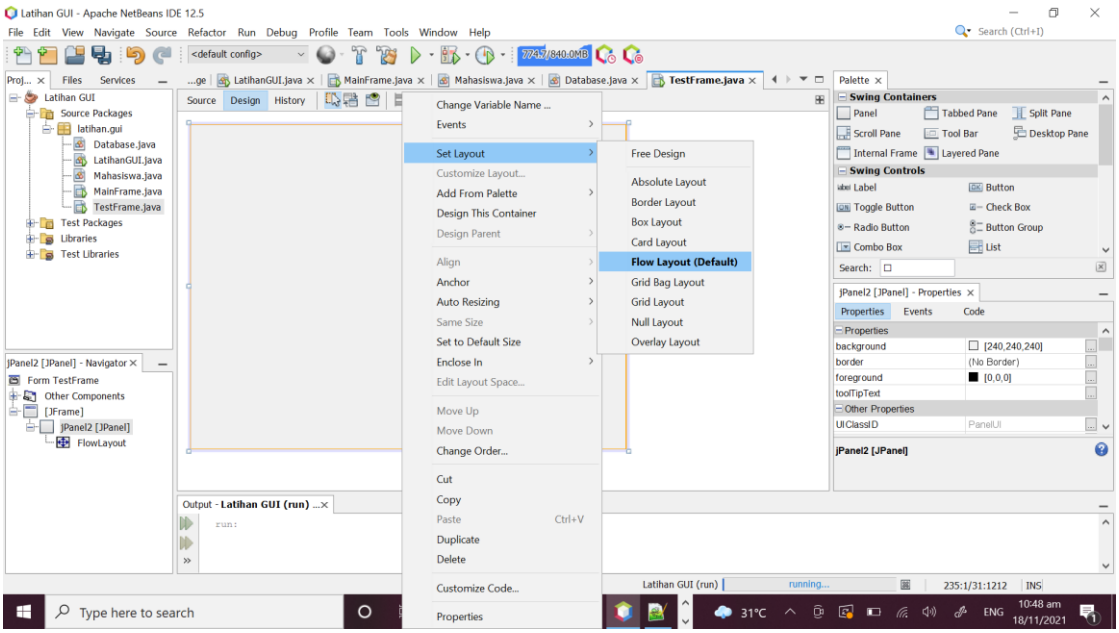
Misal kita pilih borderlayout, kemudian buatlah layout dengan button seperti berikut



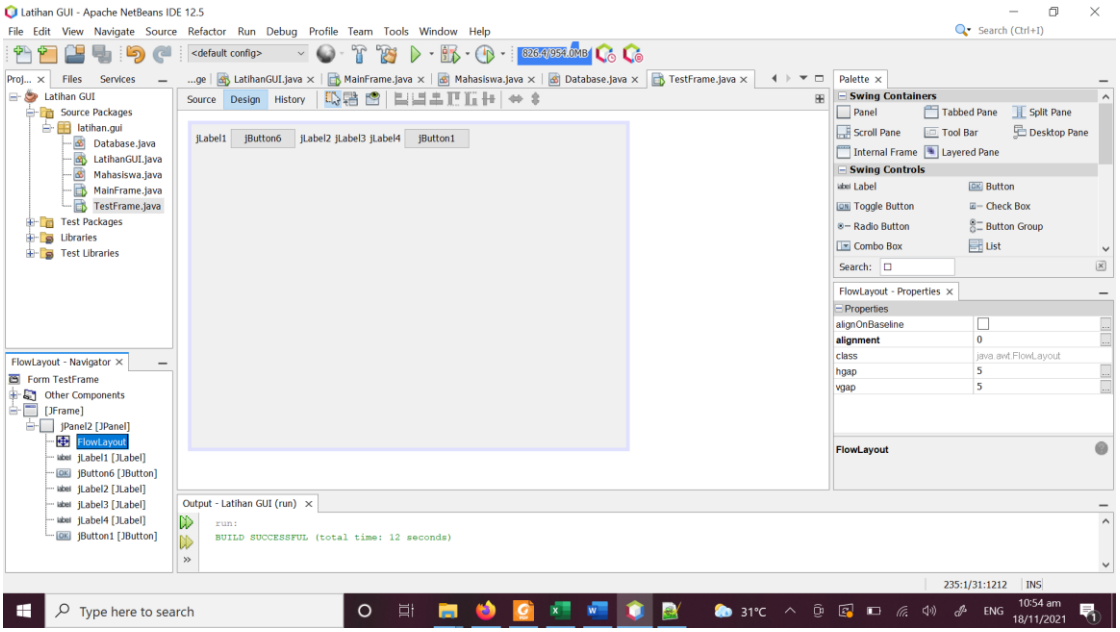
Kita bisa melakukan resize untuk melihat perilaku border layout



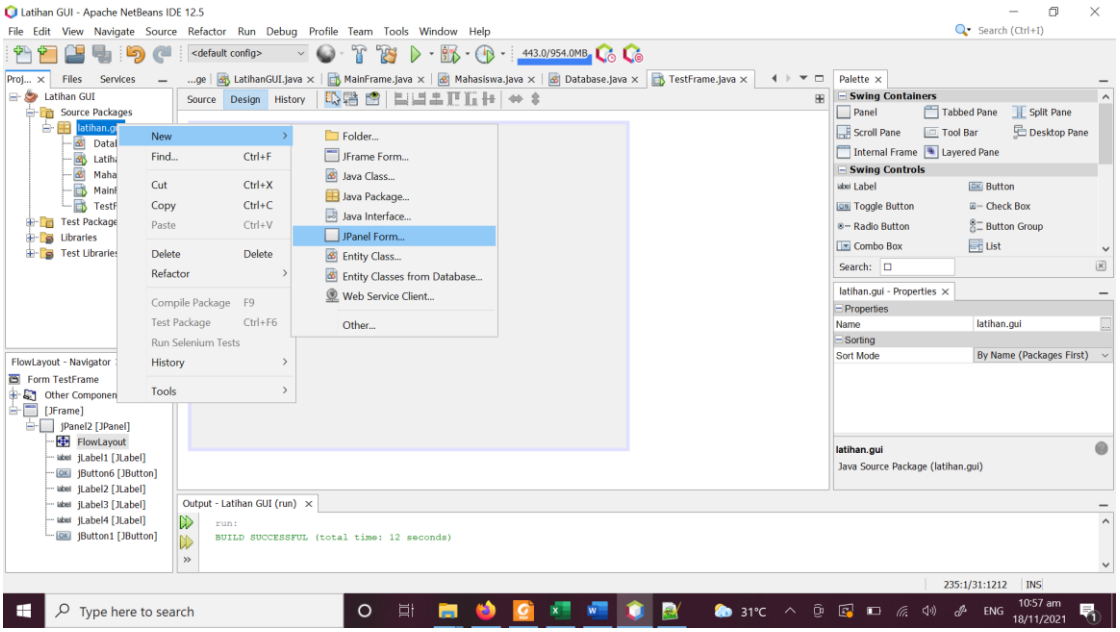
Misal untuk flow layout:

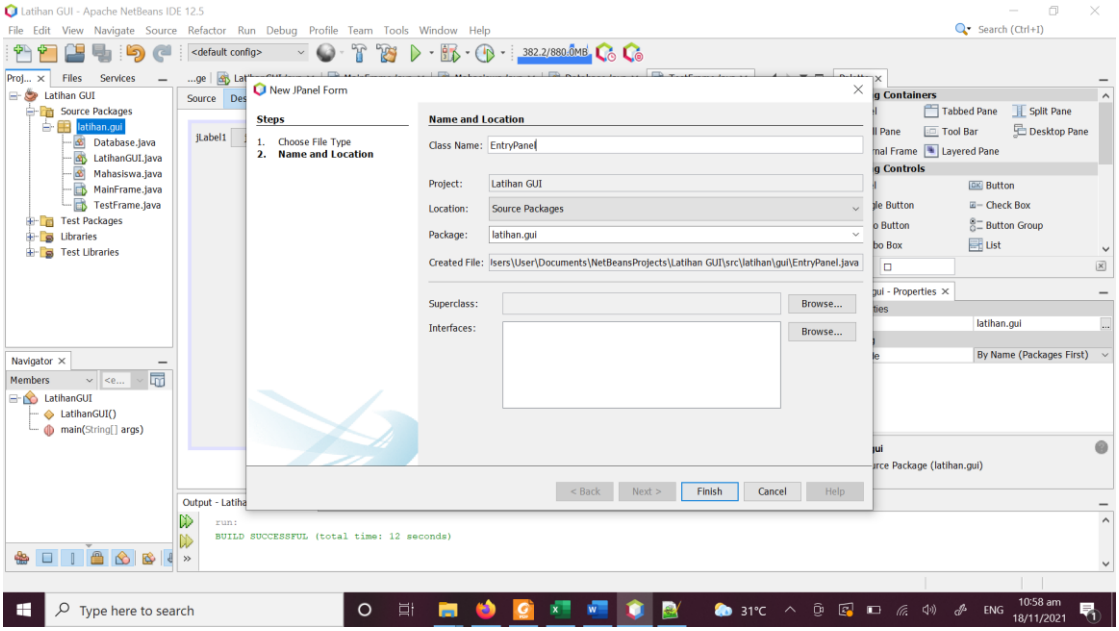


Kemudian alignment pada properties dari flow layout bisa kita atur misalnya rata kiri (0), tengah (1), atau rata kanan (2) dan kita tambahkan komponen. Ketika window diresize maka komponen di dalamnya akan menyesuaikan.

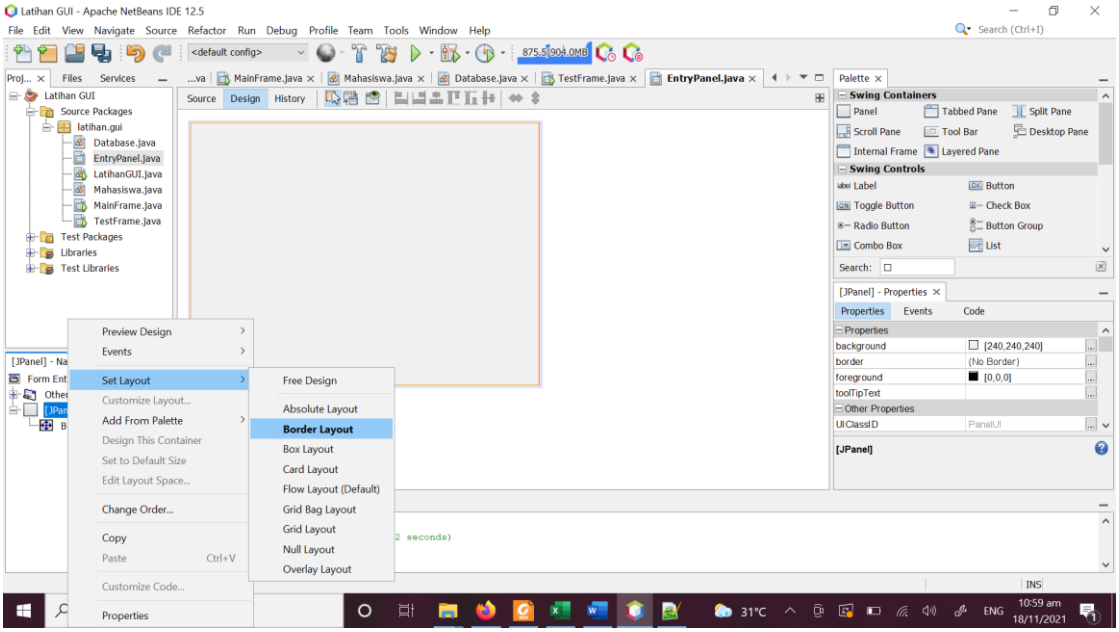


Misal untuk program entri mahasiswa tadi akan kita buat menjadi 2 panel yaitu panel entri dan panel untuk menampilkan data. Pada package latihan.gui tambahkan new Jpanel form

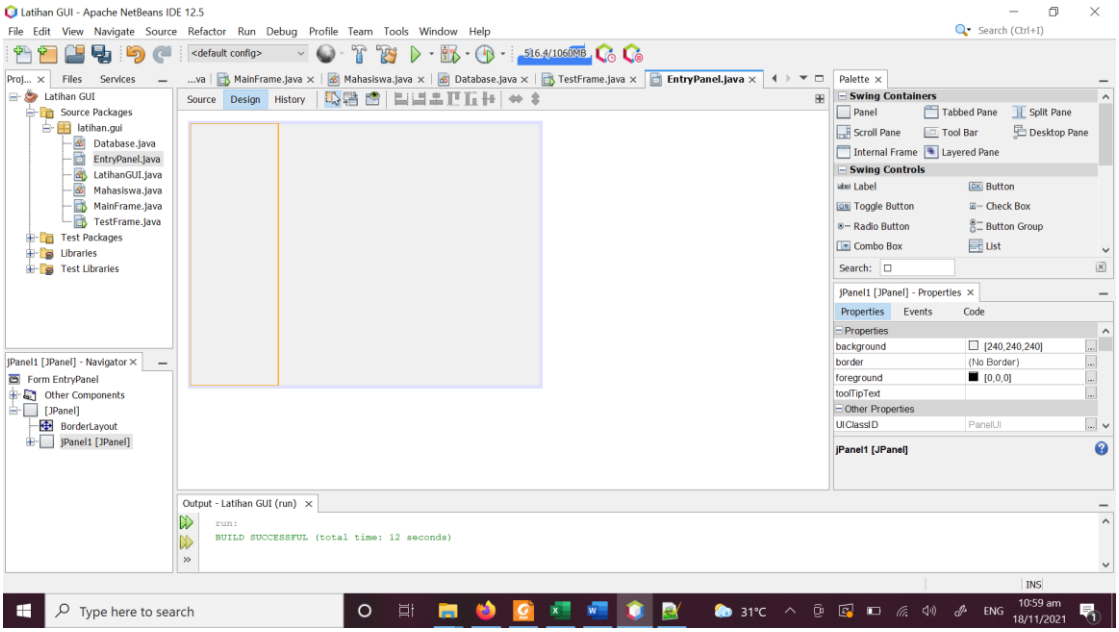




Kita buat layoutnya adalah border layout.

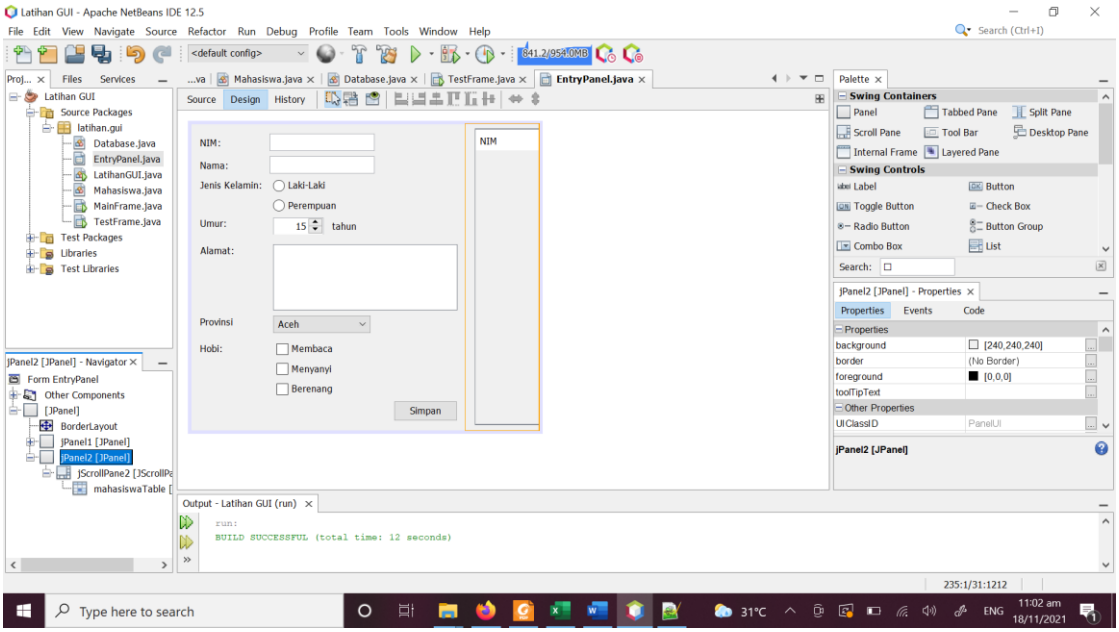


Tambahkan panel di kiri untuk entri data



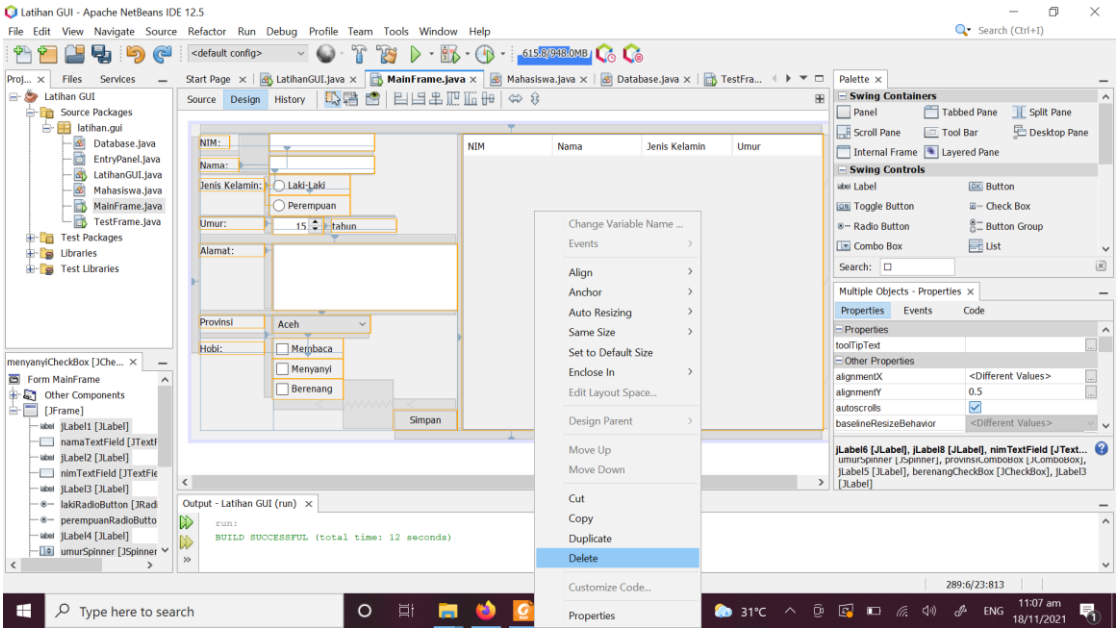
Klik 2 kali pada panel 1 untuk mengedit panel tersebut. Copy paste semua elemen entri pada MainFrame ke panel tersebut.

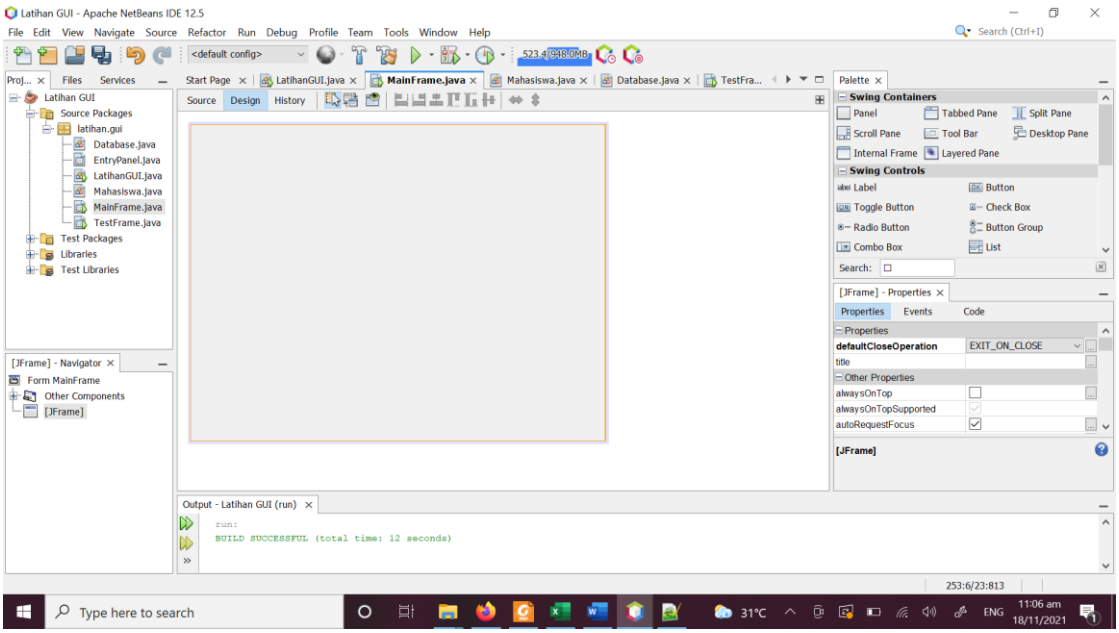
Tambahkan panel kedua untuk memuat tabel. Copy paste tabel di MainFrame ke panel 2 sehingga tampilannya menjadi seperti ini



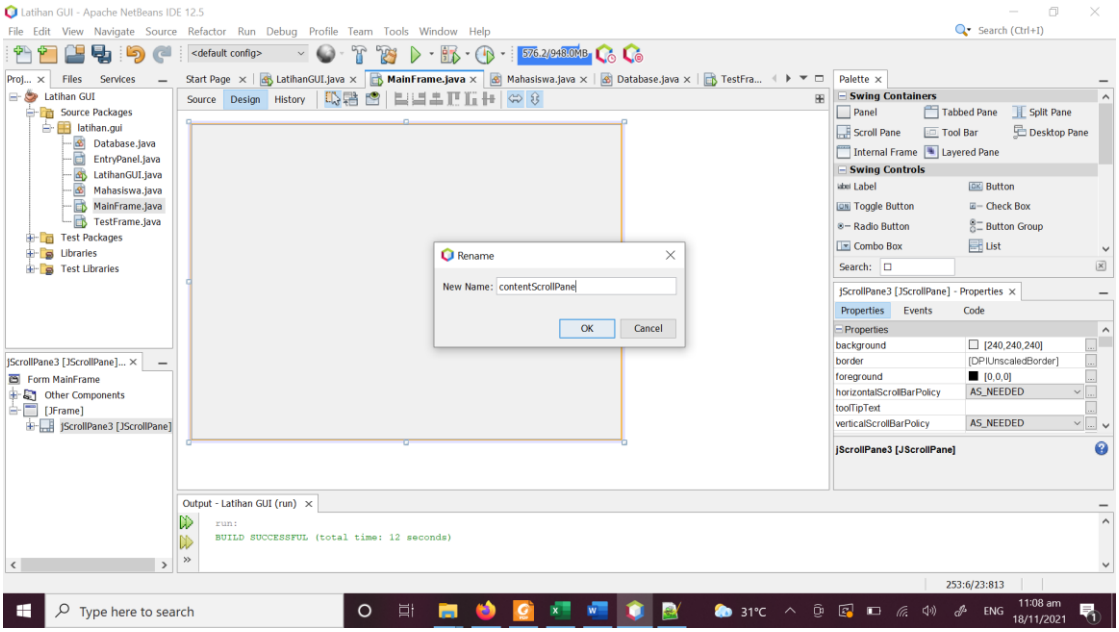
Buka source code entry panel, sesuaikan kode dan perbaiki error.

Kemudian tampilan pada MainFrame dapat kita bersihkan. Drag semua komponen kemudian delete





Kemudian tambahkan JScrollPane pada MainFrame



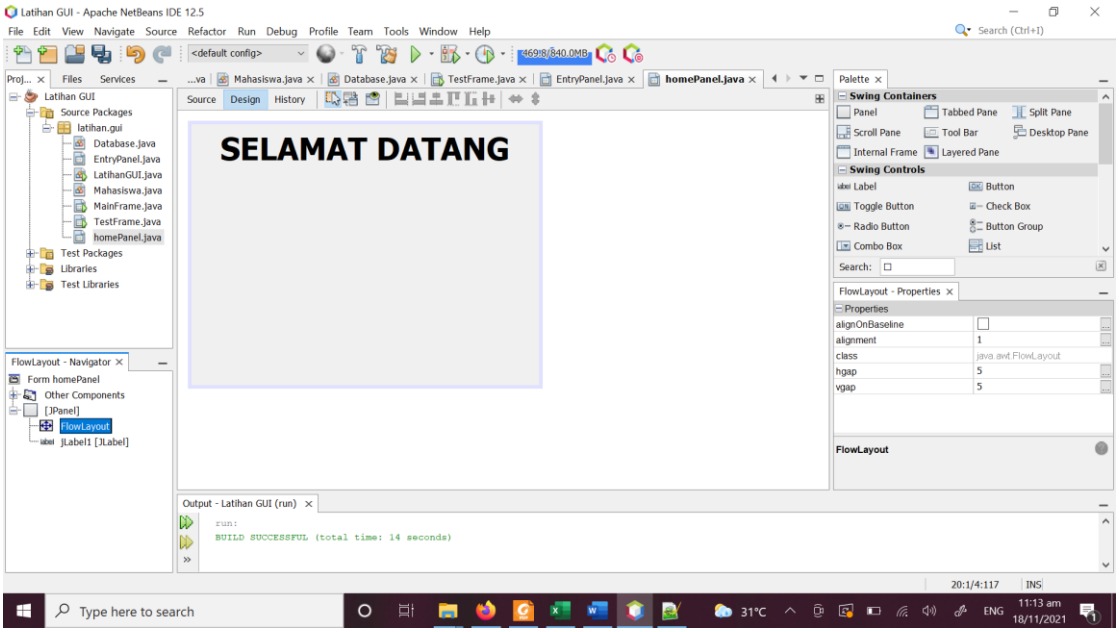
Bersihkan error pada source MainFrame, kemudin edit method constructornya menjadi seprti berikut:

```
    public MainFrame() {
        initComponents();
        contentScrollPane.setViewportView(new EntryPanel());
    }
```

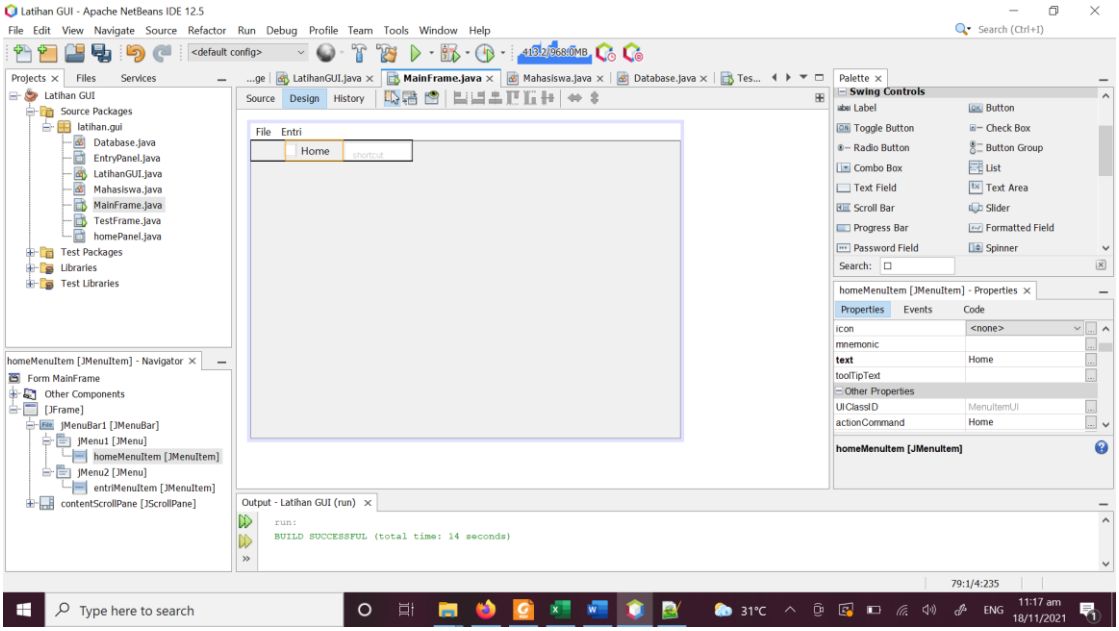
Ketika dijalankan maka kita bisa melakukan resizing pada jendela frame.

Jadi dengan menggunakan satu frame, kita bisa melakukan ganti-ganti panel. Jika kita memiliki beberapa panel seperti misalnya login panel, home panel, entri panel dan lain sebagainya maka bisa kita tampilkan bergantian pada satu Jframe tersebut.

Misal kita tambahkan HomePanel dengan ucapan selamat datang



Kemudian pada MainFrame kita tambahkan Jmenu. Tambahkan JmenuItem untuk File-> Home dan Entri-> Mahasiswa



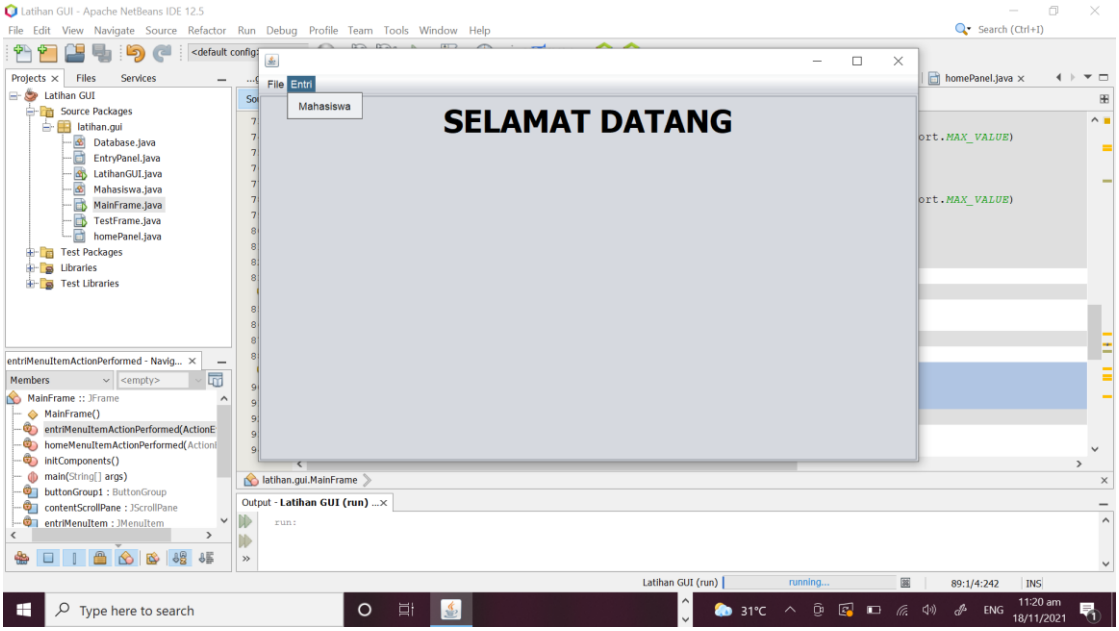
Pada action homeMenuItem tambahkan kode berikut

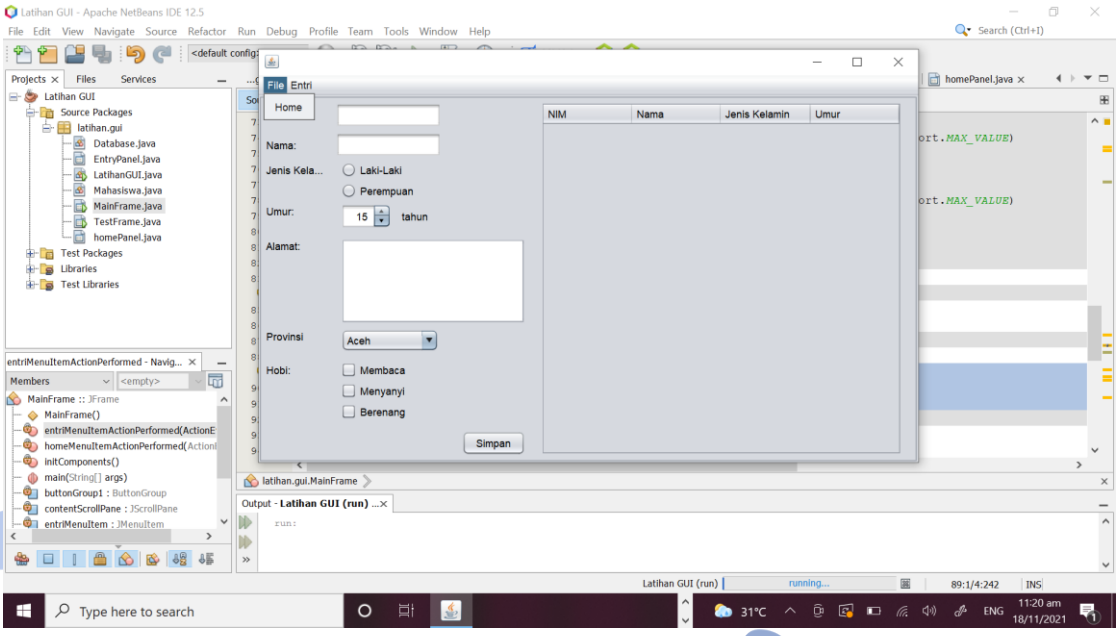
```
   private void
homeMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        contentScrollPane.setViewportView(new homePanel());
    }
```

Kemudian pada action entriMenuItem tambahkan kode berikut

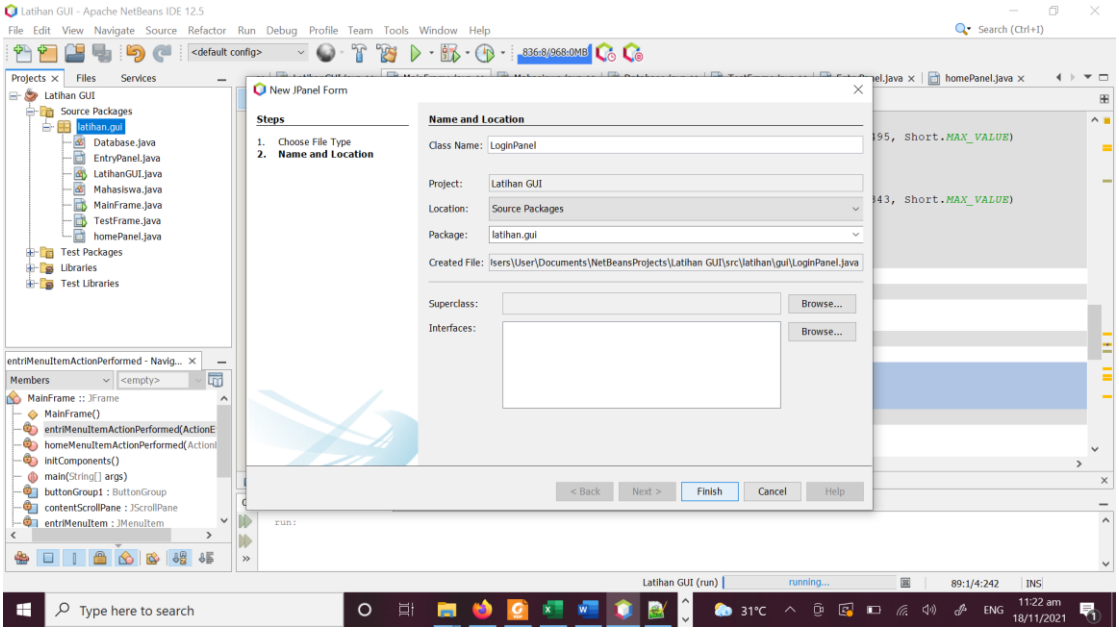
```
    private void
entriMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        contentScrollPane.setViewportView(new EntryPanel());
    }
```

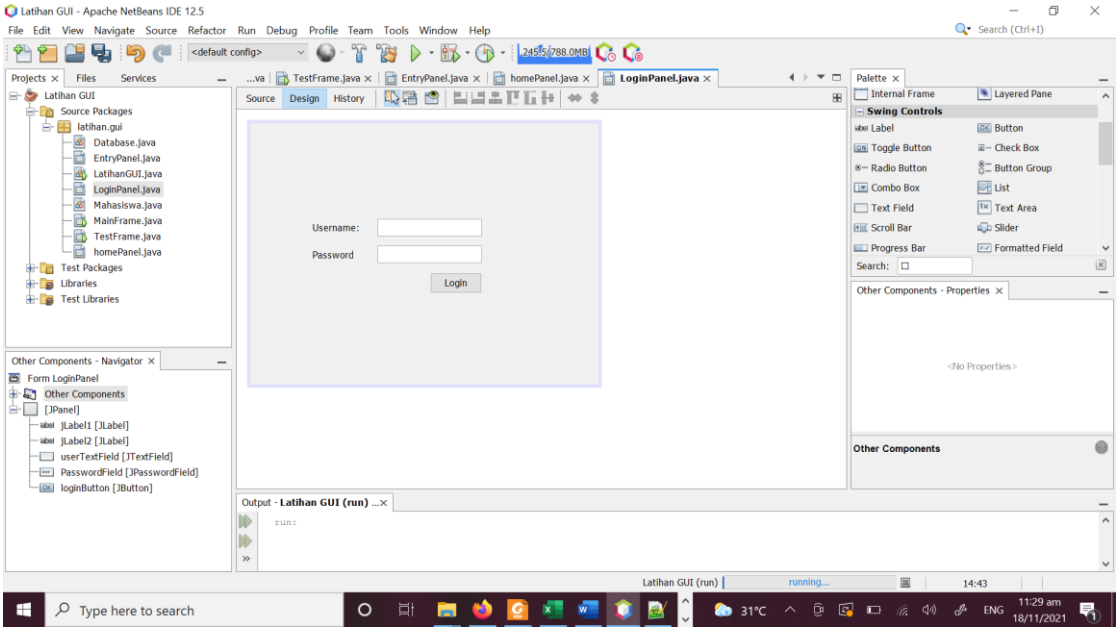
Sehingga tampilan panel pada frame dapat diganti-ganti





Misal satu lagi kita tambahkan login panel





Pada tombol loginButton tambahkan kode berikut ini

```
    private void
loginButtonActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        contentScrollPane.setViewportView(new homePanel());
    }
```

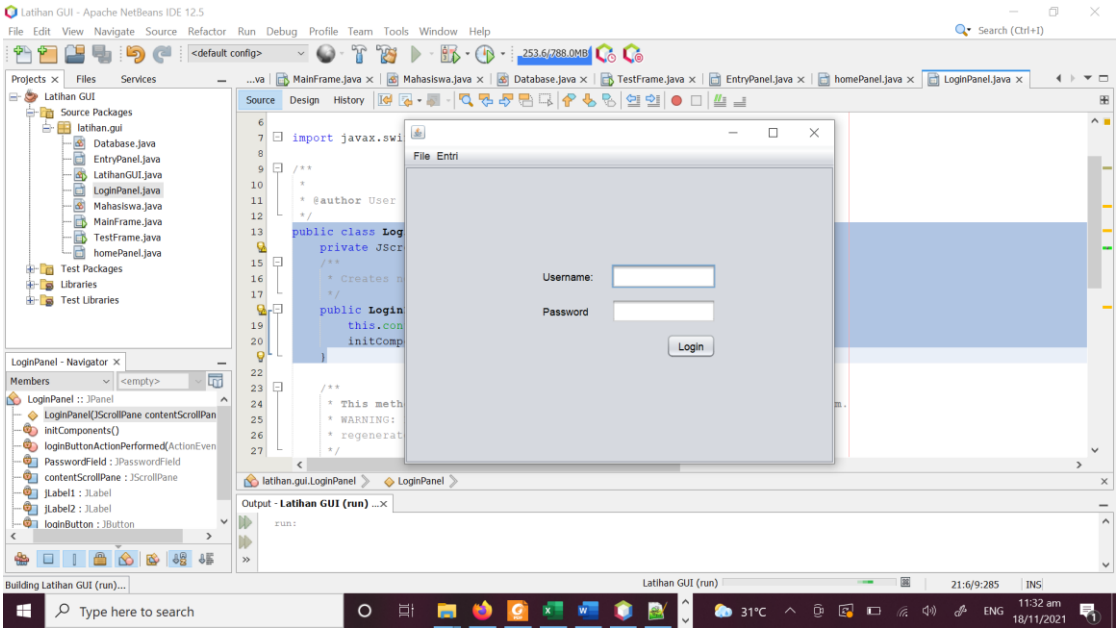
Kemudian tambahkan kode berikut pada awal source loginpanel

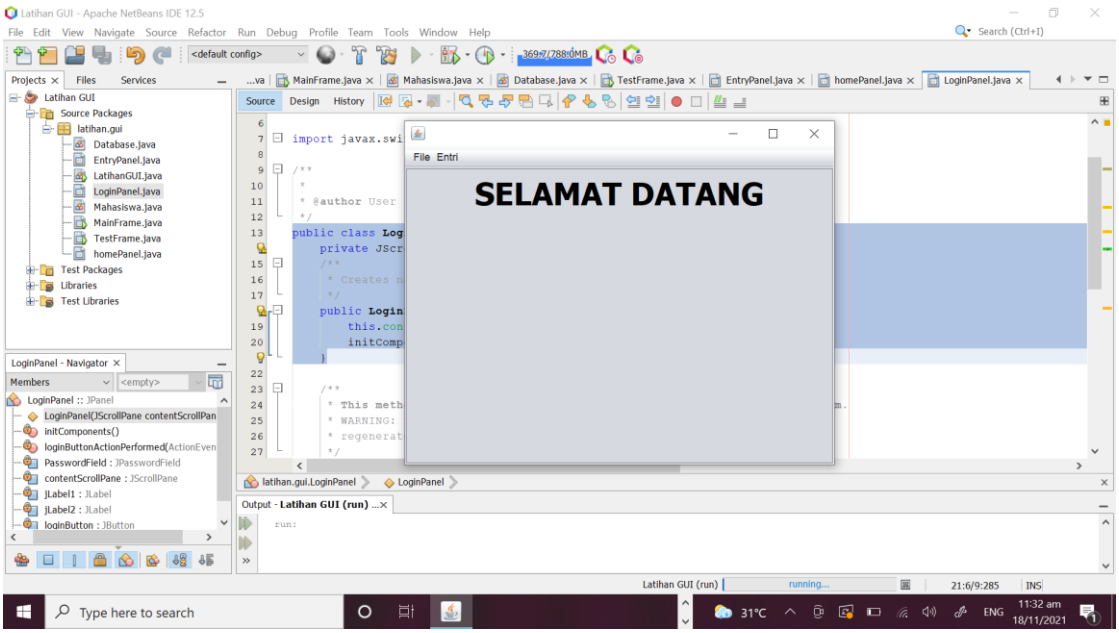
```
public class LoginPanel extends javax.swing.JPanel {
    private JScrollPane contentScrollPane;
    /**
     * Creates new form LoginPanel
     */
    public LoginPanel(JScrollPane contentScrollPane) {
        this.contentScrollPane = contentScrollPane;
        initComponents();
    }
…
…
}
```

Ubah method constructor pada MainFrame.java menjadi berikut ini:

```
public MainFrame() {
        initComponents();
        contentScrollPane.setViewportView(new
LoginPanel(contentScrollPane));
    }
```

Jalankan program, jika kita klik tombol login maka akan muncul tampilan home panel





9.5. Penugasan

Laporkan hasil praktikum berikut dengan hasil penugasan dan penjelasannya ke Dosen dalam bentuk file pdf dengan format nama *<<nim>>_modul9.*

1) Lengkapi kode semua objek
2) Tangkapan layar hasil kode yang dilengkapi
3) Tangkapan layar hasil running

MODUL 10 INPUT-OUTPUT

10.1. Deskripsi Singkat

Kebanyakan program tidak dapat mencapai tujuanya tanpa mengakses data eksternal. Data didapatkan dari *input source*, kemudian hasil dari program dapat dikirimkan ke *output destination.* Contohnya koneksi jaringan, buffer memori, atau file disk dapat dioperasikan dengan menggunakan kelas-kelas pada Java I/O. Perangkat-perangkat tersebut dihandel oleh abstraksi yang sama yang bernama *stream.* Stream adalah sebuah entitas logic yang menghasilkan atau mendapatkan input informasi. Stream terkoneksi dengan perangkat fisik oleh sistem Java I/O. Pada modul kali ini kita akan mempelajari contoh penggunaan java.IO.

10.2. Tujuan Praktikum

Setelah praktikum pada modul 10 ini diharapkan mahasiswa memahami dan menguasai penggunaan kelas-kelas pada Java IO.

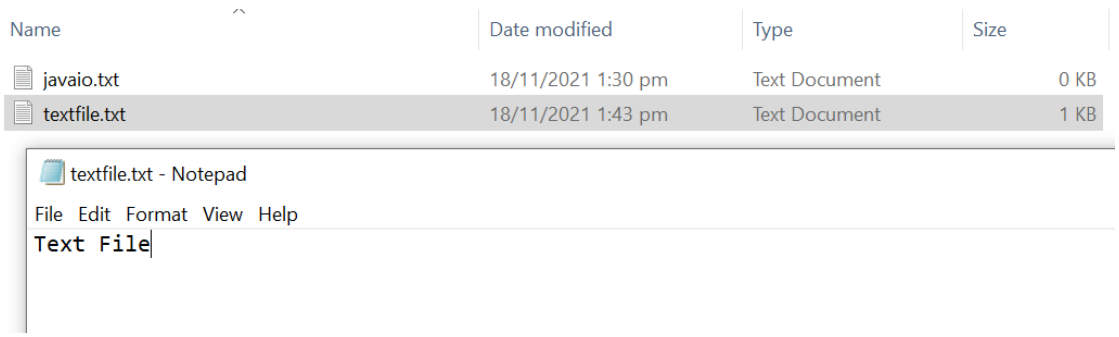
10.3. Material Praktikum

Kegiatan pada modul 10 ini memerlukan material berupa Software IDE Netbeans[10].

10.4. Kegiatan Praktikum

A. MEMBACA FILE

Siapkan sebuah folder dan buatlah 2 buah file .txt dengan nama javaio.txt dan textfile.txt. pada textfile.txt tuliskan Text File kemudian simpan.

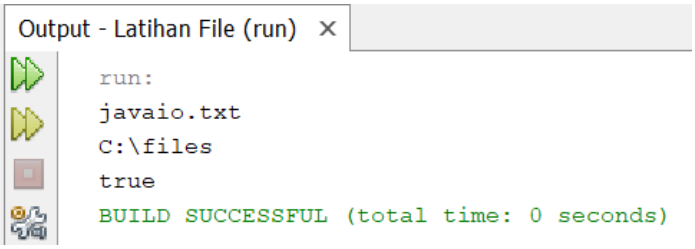


Kemudian buat project baru dengan nama ”Latihan File”. Pada file LatihanFile.java buatlah seperti kode di bawah ini

[10] https://netbeans.apache.org/download/index.html

```
package latihan.file;
import java.io.File;

public class LatihanFile {

    public static void main(String[] args) {
        // TODO code application logic here
        File f = new File("C:\\files\\javaio.txt");
        System.out.println(f.getName());
        System.out.println(f.getParent());
        System.out.println(f.canWrite());
    }
}
```

Outputnya sebagai berikut:



Kita juga bisa melakukan hal-hal yang biasa kita lakukan dengan explorer seperti rename file atau bahkan mendelete file.

```
package latihan.file;
import java.io.File;

public class LatihanFile {

    public static void main(String[] args) {
        // TODO code application logic here
        File f = new File("C:\\files\\javaio.txt");
        f.renameTo(new File("C:\\files\\javaio2.txt"));
        //f.delete();
    }
}
```

Ketika dijalankan maka nama file akan berubah. Untuk menghapus file bisa diuncomment f.delete(). Ketika dijalankan nanti file tersebut akan terhapus.

Kita juga bisa mendapatkan informasi berapa byte yang belum dibaca menggunakan metode available(). Berikut adalah listing programnya

```
package latihan.file;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.logging.Level;
import java.util.logging.Logger;

public class LatihanFile {

    public static void main(String[] args) {
        // TODO code application logic here
        FileInputStream fis = null;
        File f = new File("C:\\files\\textfile.txt");
        try {
            fis = new FileInputStream(f);
            System.out.println(fis.available());
            System.out.println((char)fis.read());
            System.out.println(fis.available());
            System.out.println((char)fis.read());
            System.out.println(fis.available());
        } catch (FileNotFoundException ex) {
            System.err.println("File tidak ditemukan");
        } catch (IOException ex) {
            System.err.println("File gagal dibaca");
        }finally{
            try {
                fis.close();
            } catch (IOException ex) {

Logger.getLogger(LatihanFile.class.getName()).log(Level.SEVERE,
null, ex);
            }
        }
```

```
    }
}
```

Jika ingin membaca hingga akhir teks perkarakter maka kita bisa menggunakan listing program di bawah ini:

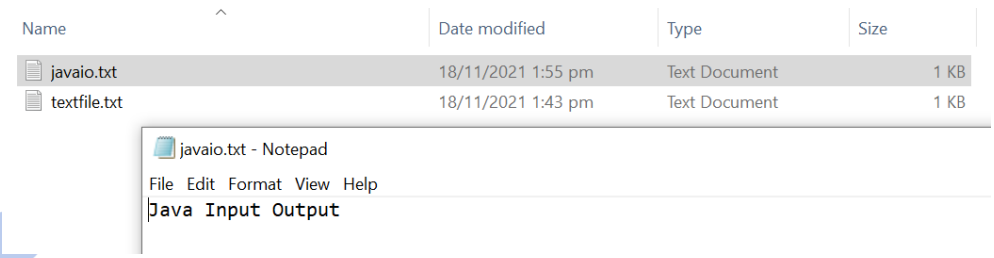
```
package latihan.file;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.logging.Level;
import java.util.logging.Logger;

public class LatihanFile {

    public static void main(String[] args) {
        // TODO code application logic here
        FileInputStream fis = null;
        File f = new File("C:\\files\\textfile.txt");
        try {
            fis = new FileInputStream(f);
            while(fis.available()>0){
                System.out.print((char) fis.read());
            }
        } catch (FileNotFoundException ex) {
            System.err.println("File tidak ditemukan");
        } catch (IOException ex) {
            System.err.println("File gagal dibaca");
        }finally{
            try {
                fis.close();
            } catch (IOException ex) {

Logger.getLogger(LatihanFile.class.getName()).log(Level.SEVERE,
null, ex);
            }
        }
    }
}
```

B. MENULIS FILE

Untuk menulis file, perhatikan listing program di bawah ini:

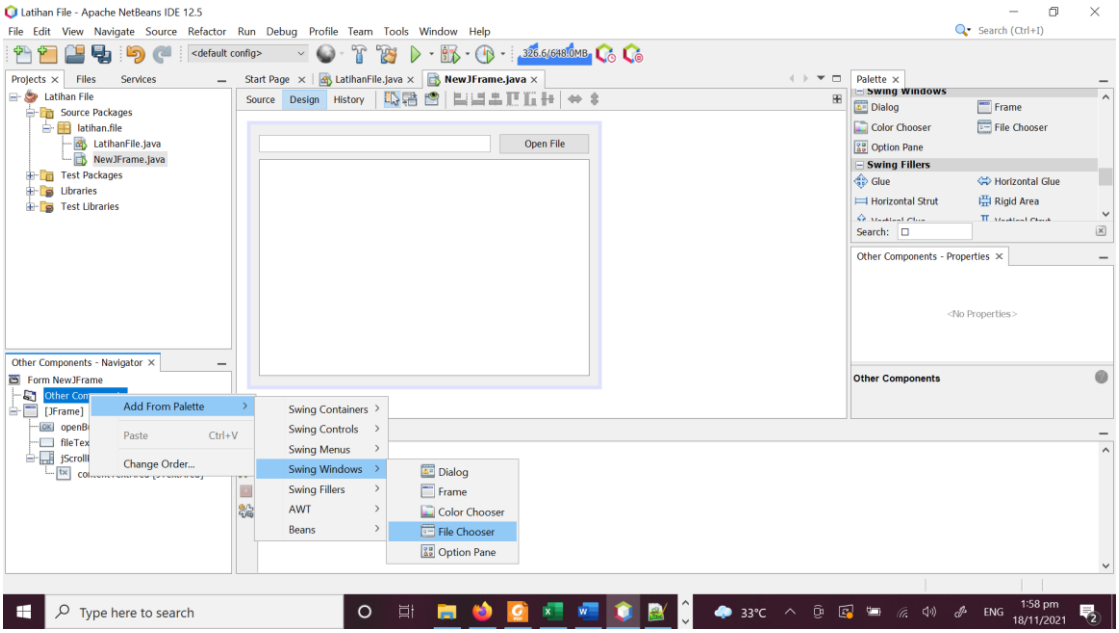
```
package latihan.file;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.logging.Level;
import java.util.logging.Logger;

public class LatihanFile {

    public static void main(String[] args) {
        try {
            FileOutputStream fos = new
FileOutputStream("C:\\files\\javaio.txt");
            String str = "Java Input Output";
            fos.write(str.getBytes());
            fos.close();
        } catch (FileNotFoundException ex) {

Logger.getLogger(LatihanFile.class.getName()).log(Level.SEVERE,
null, ex);
        } catch (IOException ex) {

Logger.getLogger(LatihanFile.class.getName()).log(Level.SEVERE,
null, ex);
        }
    }
}
```

Ketika dijalankan, maka file javaio.txt akan berisi kalimat Java Input Ouput.



C. GUI

Buatlah rancangan GUI sebagai berikut:

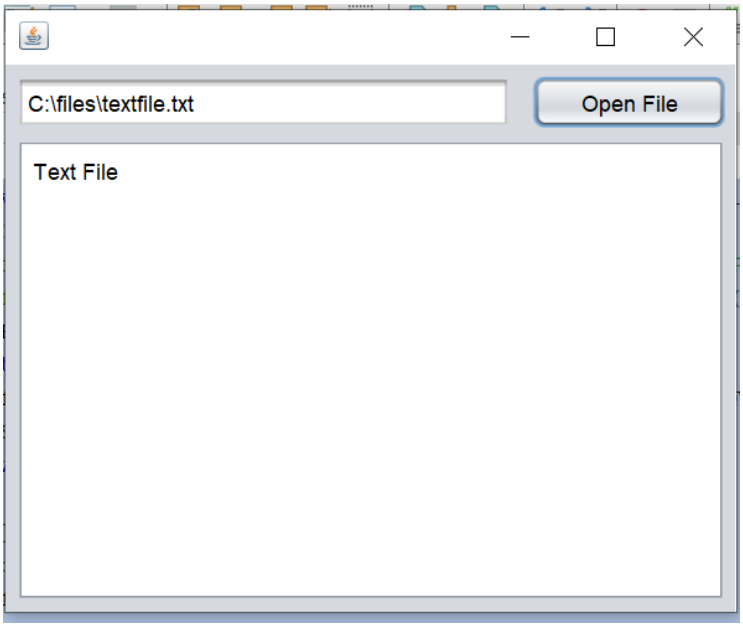


Pada button Open File, tambahkan event berikut

```
    private void
openButtonActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        if(fileChooser.showOpenDialog(this) ==
JFileChooser.APPROVE_OPTION){

fileTextField.setText(fileChooser.getSelectedFile().toString());
            FileInputStream fis = null;
            try {
            fis = new
FileInputStream(fileChooser.getSelectedFile());
            String str = "";
            while(fis.available()>0){
                str += (char) fis.read();
            }
            contentTextArea.setText(str);
            fis.close();
            } catch (FileNotFoundException ex) {
                System.err.println("File tidak ditemukan");
            } catch (IOException ex) {
                System.err.println("File gagal dibaca");
```

```
                }
            }
        }
```

Jalankan file tersebut. Berikut adalah tampilan layarnya



Anda bisa juga menambahkan tombol simpan untuk menyimpan perubahan pada contentTextArea ke file pada fileTextField.

MEMBACA DAN MENULIS OBJECT

Untuk membaca object, kita menggunakan kelas ObjectOutputStream dan untuk membaca object kita menggunakan kelas ObjectInputStream

Buka kembali project sebelumnya, kopikan kelas mahasiswa ke project yang sekarang. Kemudian kelas mahasiswa ini harus mengimplementasikan interface serializable.

```
public class Mahasiswa implements Serializable {
    private String nim,nama,jenisKelamin;
    private int umur;
    private String alamat, provinsi;
    private ArrayList<String> hobi;
```

Kemudian pada kelas Latihan File ubahlah seperti di bawah ini

```
package latihan.file;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectOutputStream;
import java.util.logging.Level;
import java.util.logging.Logger;

public class LatihanFile {

    public static void main(String[] args) {
        try {
            FileOutputStream fos = new
FileOutputStream("C:\\files\\objek.dat");
            ObjectOutputStream oos = new ObjectOutputStream(fos);
            Mahasiswa mhs = new Mahasiswa();
            mhs.setNim("1234");
            mhs.setNama("Andi");
            oos.writeObject(mhs);
            oos.close();
            fos.close();
        } catch (FileNotFoundException ex) {

Logger.getLogger(LatihanFile.class.getName()).log(Level.SEVERE,
null, ex);
        } catch (IOException ex) {

Logger.getLogger(LatihanFile.class.getName()).log(Level.SEVERE,
null, ex);
        }
    }
}
```

Ketika dijalankan maka di folder tersebut ada file baru bernama objek.dat yang menyimpan data 1 mahasiswa. Objek.dat ini bukan dimaksudkan untuk dibaca oleh textviewer. Ketika dibuka dengan text editor maka isinya akan tidak beraturan

Name

javaio.txt
objek.dat
textfile.txt

D. MEMBACA OBJECT

Perhatikan listng program LatihanFile.java berikut

```
package latihan.file;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.ObjectInputStream;

public class LatihanFile {

    public static void main(String[] args) {
        try {
            File f = new File("C:\\files\\objek.dat");
            FileInputStream fis = new FileInputStream(f);
            ObjectInputStream ois = new ObjectInputStream(fis);
            Mahasiswa mhs = (Mahasiswa) ois.readObject();
            System.out.println(mhs.getNim());
            System.out.println(mhs.getNama());
            ois.close();
            fis.close();
        } catch (FileNotFoundException ex) {
            System.err.println("File tidak ditemukan");
        } catch (IOException ex) {
            System.err.println("File gagal dibaca");
        } catch (ClassNotFoundException ex) {
            System.err.println("Format File salah");
        }
    }
}
```

Jika dijalankan maka sebagai berikut:

```
Output - Latihan File (run)
run:
1234
Andi
BUILD SUCCESSFUL (total time: 0 seconds)
```

Berikutnya buka kembali project LatihanGUI sebelumnya, kita akan menyimpan ke dalam file untuk data mahasiswa yang telah dientri.

Buka kelas database.java dan implementasikan interface serializable. Jangan lupa kelas mahasiswa.java pada project tersebut juga harus implements Serializable

Pada Database.java ubahlah menjadi seperti berikut:

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package latihan.gui;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.Serializable;
import java.util.ArrayList;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author User
```

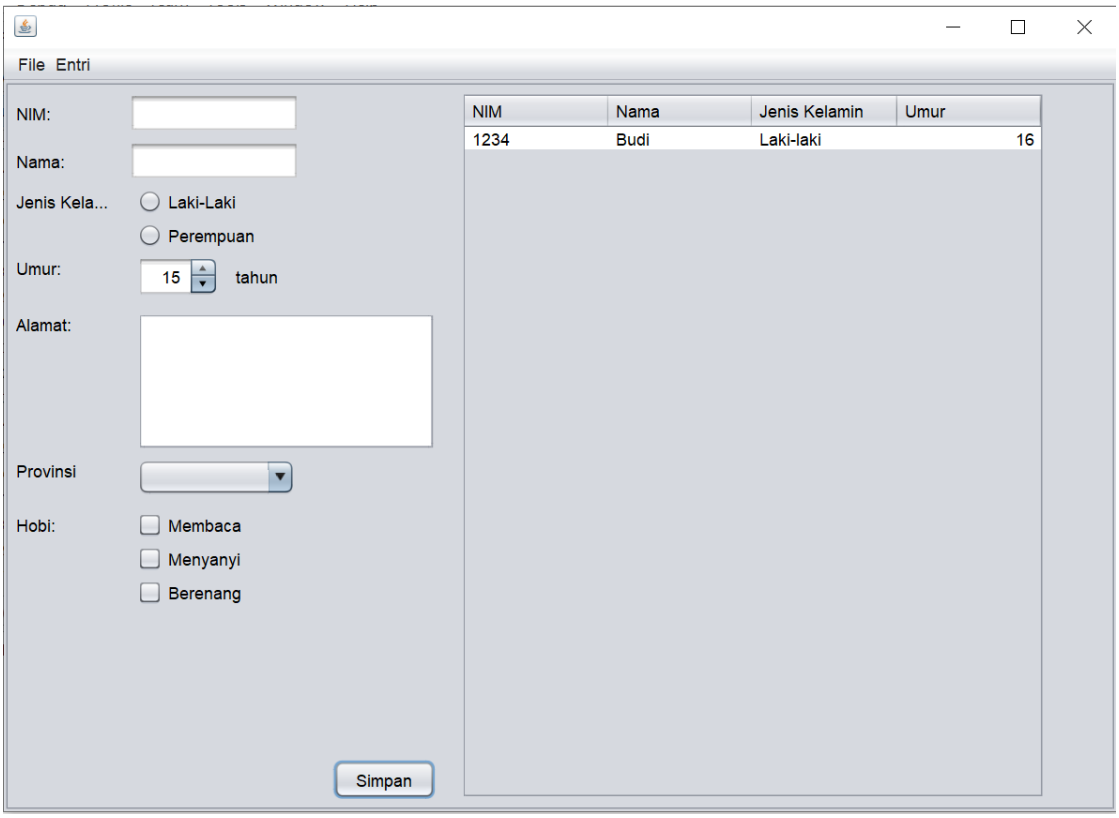
```
 */
public class Database implements Serializable{
    public static Database instance;
    private ArrayList<Mahasiswa> data = new ArrayList<>();
    private Database(){
    }

    public static synchronized Database getInstance(){
        loadFile();
        if(instance == null){
            instance = new Database();
        }
        return instance;
    }

    public void insertMahasiswa(Mahasiswa mahasiswa){
        data.add(mahasiswa);
        updateFile();
    }

    public List<Mahasiswa> getListMahasiswa(){
        return data;
    }

    private static void loadFile(){
        try {
            File f = new File("C:\\files\\database.dat");
            FileInputStream fis = new FileInputStream(f);
            ObjectInputStream ois = new ObjectInputStream(fis);
            instance = (Database) ois.readObject();
            ois.close();
            fis.close();
        } catch (FileNotFoundException ex) {
            System.err.println("File tidak ditemukan");
        } catch (IOException ex) {
            System.err.println("File gagal dibaca");
        } catch (ClassNotFoundException ex) {
            System.err.println("Format File salah");
```

```
        }
    }

    private void updateFile(){
        try {
            FileOutputStream fos = new
FileOutputStream("C:\\files\\database.dat");
            ObjectOutputStream oos = new ObjectOutputStream(fos);
            oos.writeObject(instance);
            oos.close();
            fos.close();

        } catch (FileNotFoundException ex) {
            System.err.println("File tidak ditemukan");
        } catch (IOException ex) {
           System.err.println("Gagal menulis file");
        }
    }
}
```

Pada EntryPanel.java juga perlu ditambahkan loadTableData saat contruksi

```
public EntryPanel() {
        initComponents();
        loadTableData();
    }
```

Kemudian run program tersebut dan entri satu data



Kemudian tutup window tersebut.

Ketika program dijalankan ulang maka data yang sebelumnya kita entri langsung ter-load pada tabel. Tidak hilang seperti sebelumnya.

10.5. Penugasan

Laporkan hasil praktikum berikut dengan hasil penugasan dan penjelasannya ke Dosen dalam bentuk file pdf dengan format nama *<<nim>>_modul10.*

1) Lengkapi kode semua objek
2) Tangkapan layar hasil kode yang dilengkapi
3) Tangkapan layar hasil running

MODUL 11 DATABASE ACCESS (JDBC)

11.1. Deskripsi Singkat

Untuk platform java, kelas-kelas yang menangani operasi database adalah JDBC. Pada praktikum modul 11 ini kita akan mempraktekkan koneksi database dengan menggunakan JDBC dan MySQL.

11.2. Tujuan Praktikum

Setelah praktikum pada modul 11 ini diharapkan mahasiswa memahami dan menguasai cara koneksi database menggunakan JDBC.

11.3. Material Praktikum

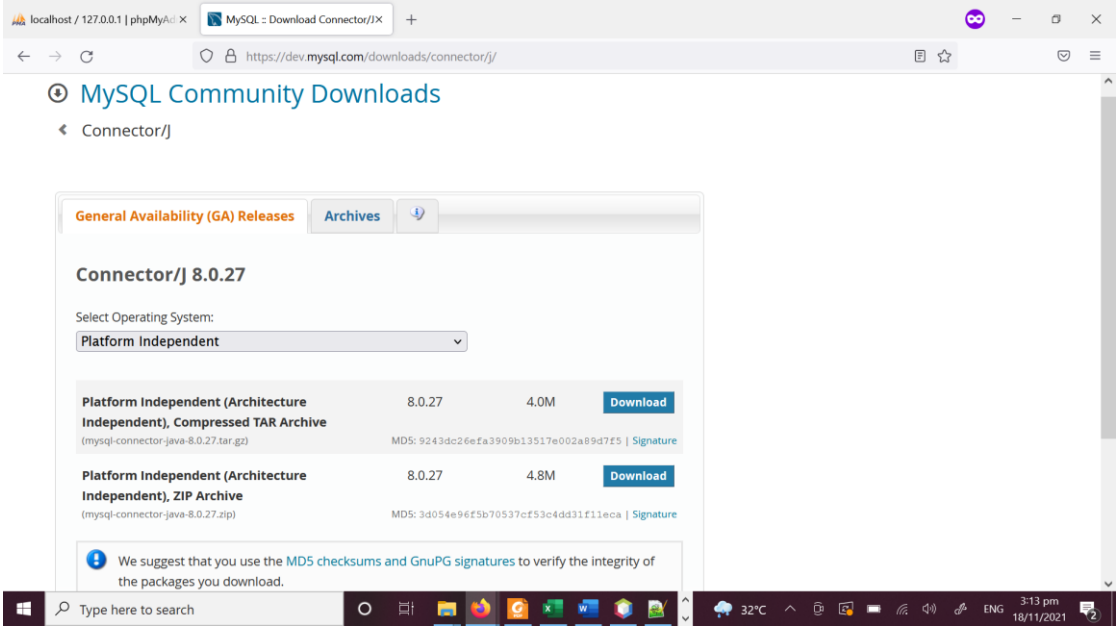
Kegiatan pada modul 11 ini memerlukan material berupa Software IDE Netbeans[11].

11.4. Kegiatan Praktikum

Untuk kegiatan praktikum modul 11 ini kita akan menggunakan project latihan GUI yang sebelumnya kita buat.

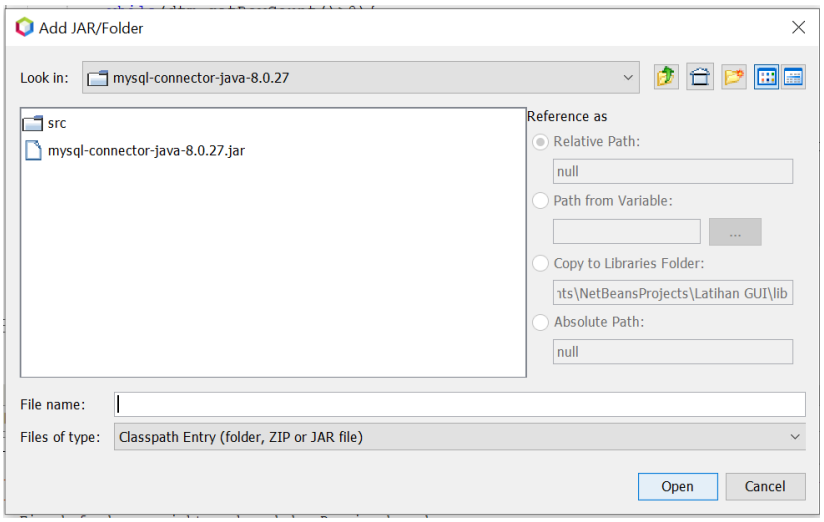
Langkah pertama Unduh MySQL Java connector pada halaman web https://dev.mysql.com/downloads/connecor/j/ kemudian pilihan operating systemnya adalah platform independent, kemudian unduh yang ZIP archive

[11] https://netbeans.apache.org/download/index.html

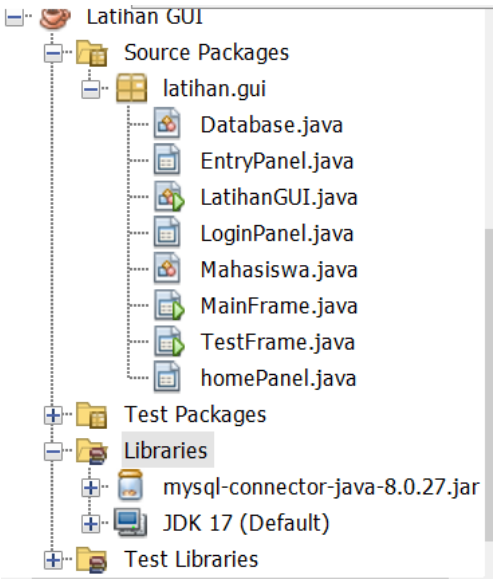


Kemudian ekstrak zip tersebut untuk nanti kita tambahkan pada library project. Pada folder library di panel kiri, klik kanan, Add JAR/Folder

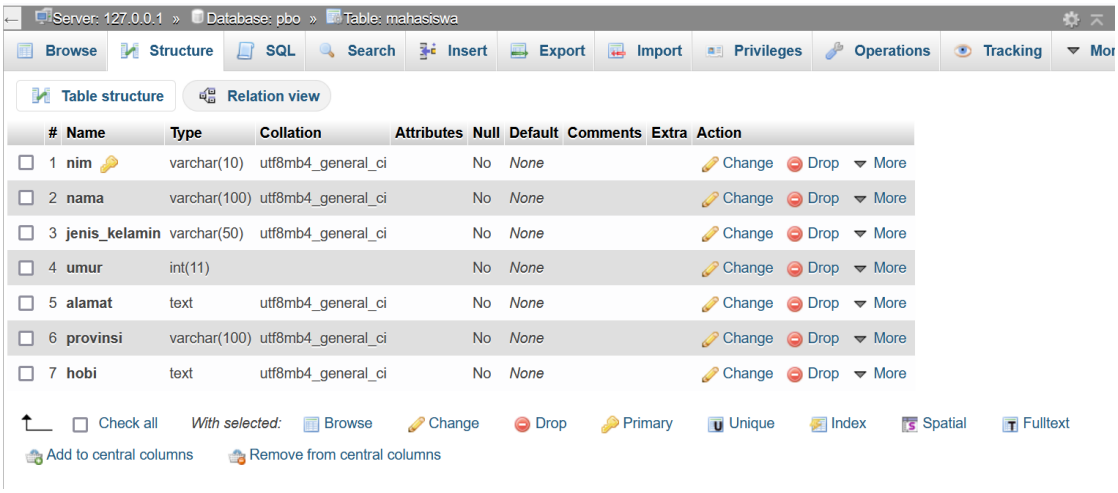
Kemudian tambahkan mysql connector java



Sehiingga pada jendela library akan seperti ini



Bukalah phpmyadmin, kemudian buatlah database bernama ”pbo” dan tabel bernama ”mahasiswa” dengan struktur sebagai berikut:



Kemudian modifikasi Databasae.java pada project latihan GUI menjadi seperti ini

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package latihan.gui;


import java.io.Serializable;
import java.util.ArrayList;
```

```
import java.util.List;
import java.sql.*;
import java.util.Arrays;
/**
 *
 * @author User
 */
public class Database implements Serializable{
    public static Database instance;

    private final String DB_TYPE = "mysql";
    private final String DB_HOST = "localhost";
    private final String DB_PORT = "3306";
    private final String DB_NAME = "pbo";
    private final String DB_USER = "root";
    private final String DB_PASS = "";

    private Database(){
    }

    public static synchronized Database getInstance(){
        if(instance == null){
            instance = new Database();
        }
        return instance;
    }

    public void insertMahasiswa(Mahasiswa mahasiswa) throws
SQLException{
        Connection conn = getConnection();
        try{
            String sql="INSERT INTO mahasiswa
VALUES(?,?,?,?,?,?,?)";
            PreparedStatement pstmt = conn.prepareStatement(sql);
            pstmt.setString(1, mahasiswa.getNim());
            pstmt.setString(2, mahasiswa.getNama());
            pstmt.setString(3, mahasiswa.getJenisKelamin());
            pstmt.setInt(4, mahasiswa.getUmur());
            pstmt.setString(5, mahasiswa.getAlamat());
```

```
            pstmt.setString(6, mahasiswa.getProvinsi());
            pstmt.setString(7, String.join(",",
mahasiswa.getHobi()));
            pstmt.executeUpdate();
        } catch(SQLException ex){
            throw ex;
        } finally{
            if(conn!=null){
                conn.close();
            }
        }
    }

    public List<Mahasiswa> getListMahasiswa() throws SQLException{
        List<Mahasiswa> mhsList = new ArrayList<>();
        Connection conn = getConnection();
        try{
            String sql = "SELECT * FROM mahasiswa";
            Statement stmt = conn.createStatement();
            ResultSet rs = stmt.executeQuery(sql);
            while(rs.next()){
                Mahasiswa mhs = new Mahasiswa();
                mhs.setNim(rs.getString("nim"));
                mhs.setNama(rs.getString("nama"));

mhs.setJenisKelamin(rs.getString("jenis_kelamin"));
                mhs.setUmur(rs.getInt("umur"));
                mhs.setAlamat(rs.getString("alamat"));
                mhs.setProvinsi(rs.getString("provinsi"));
                mhs.setHobi(new
ArrayList<>(Arrays.asList(rs.getString("hobi").split",")))));

                mhsList.add(mhs);
            }
        }catch(SQLException ex){
            throw ex;
        } finally{
            if (conn!=null){
                conn.close();
```

```
            }
        }

        return mhsList;
    }

    private Connection getConnection() throws SQLException {
        return
DriverManager.getConnection("jdbc:"+DB_TYPE+"://"+DB_HOST+":"+DB_P
ORT+"/"+DB_NAME,DB_USER,DB_PASS);
    }

}
```

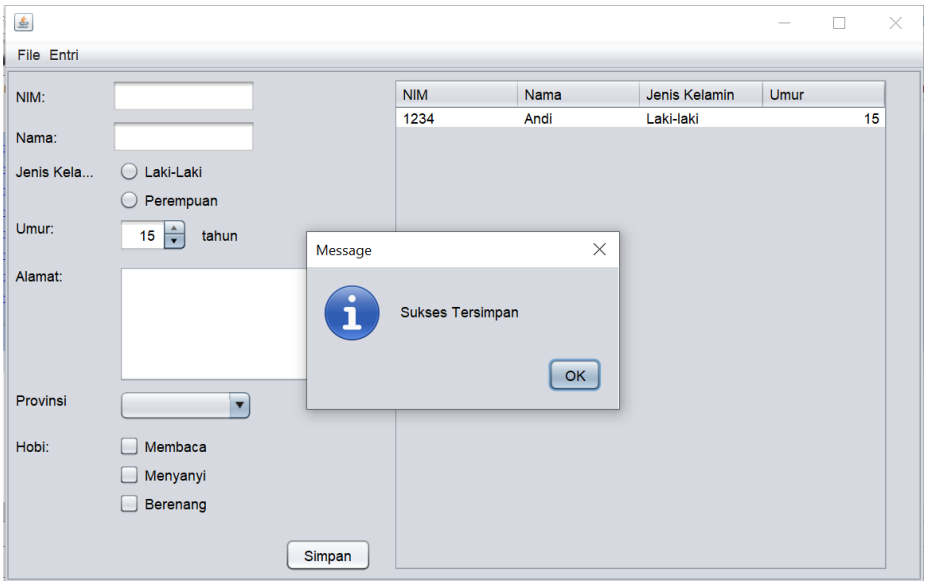
Kemudian ada juga perubahan pada EntryPanel.java pada method insert dan load data table

```
private void
simpanButtonActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        Mahasiswa mhs = new Mahasiswa();
        mhs.setNim(nimTextField.getText());
        mhs.setNama(namaTextField.getText());
        if(lakiRadioButton.isSelected()){
            mhs.setJenisKelamin("Laki-laki");
        }
        if(perempuanRadioButton.isSelected()){
            mhs.setJenisKelamin("Perempuan");
        }
        int umur = (Integer) umurSpinner.getValue();
        mhs.setUmur(umur);
        mhs.setAlamat(alamatTextArea.getText());

mhs.setProvinsi(provinsiComboBox.getSelectedItem().toString());
        ArrayList<String> hobiList = new ArrayList<>();
        if(membacaCheckBox.isSelected()){
            hobiList.add("membaca");
        }
        if(menyanyiCheckBox.isSelected()){
```

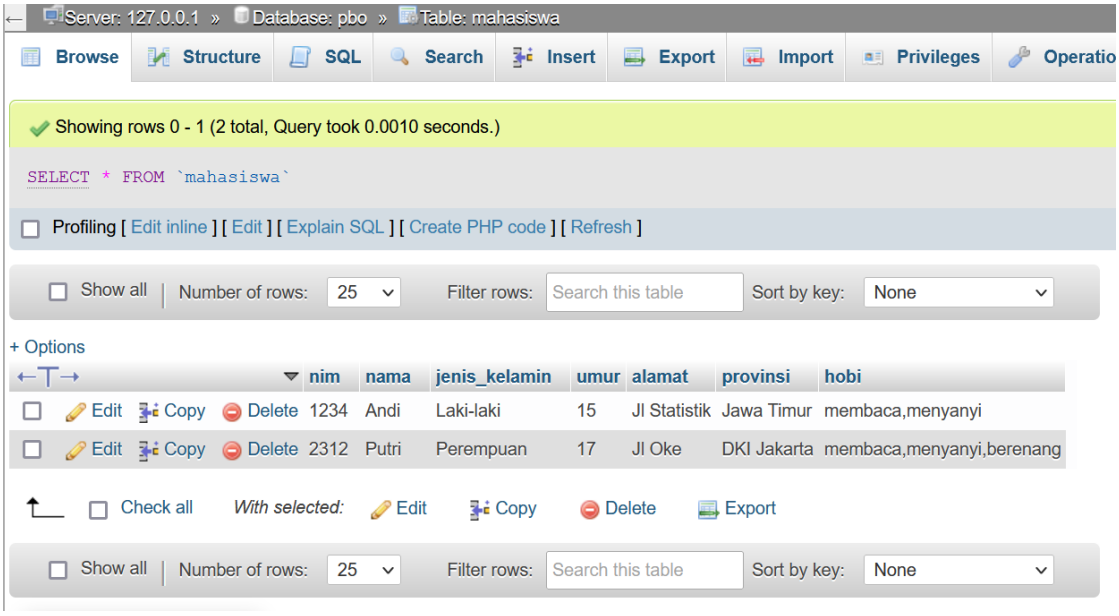
```
            hobiList.add("menyanyi");
        }
        if(berenangCheckBox.isSelected()){
            hobiList.add("berenang");
        }

        mhs.setHobi(hobiList);

        try {
            Database.getInstance().insertMahasiswa(mhs);
            clearForm();
            JOptionPane.showMessageDialog(this, "Sukses
Tersimpan");
            loadTableData();
        } catch (SQLException ex) {
            System.err.println(ex);
            JOptionPane.showMessageDialog(this, "Gagal menyimpan
data", "Gagal", JOptionPane.ERROR_MESSAGE);
        }

    }
    private void loadTableData(){
        DefaultTableModel dtm = (DefaultTableModel)
mahasiswaTable.getModel();
        //refresh tabel
        while(dtm.getRowCount()>0){
            dtm.removeRow(0);
        }
        try {
            //isi tabel
            for(Mahasiswa mhs:
Database.getInstance().getListMahasiswa()){
                dtm.addRow(new
Object[]{mhs.getNim(),mhs.getNama(),mhs.getJenisKelamin(),mhs.getU
mur()});
            }
        } catch (SQLException ex) {
            System.err.println(ex);
            JOptionPane.showMessageDialog(this, "Gagal mengambil
data", "Gagal", JOptionPane.ERROR_MESSAGE);
        }
```

```
        }
```

Kemudian kita jalankan program dan coba cek apakah berhasil masuk ke database



Hasil pengecekan di database sudah ternyata sudah berhasil masuk datanya



11.5. Penugasan

Laporkan hasil praktikum berikut dengan hasil penugasan dan penjelasannya ke Dosen dalam bentuk file pdf dengan format nama *<<nim>>_modul11.*

1) Lengkapi kode semua objek
2) Tangkapan layar hasil kode yang dilengkapi
3) Tangkapan layar hasil running

MODUL 12 CONCURRENCY (THREADING)

12.1. Deskripsi Singkat

Concurrency memungkinkan 2 atau lebih proses untuk berjalan bersamaan. Pada pembahasan sebelumnya, semua proses dilakukan berjalan secara sekuensial. Concurrency adalah kemampuan untuk menjalankan sebagian atau beberapa program secara paralel. Pada modul 12 ini akan dibahas mengenai concurrency pada pemrograman berorientasi objek.

12.2. Tujuan Praktikum

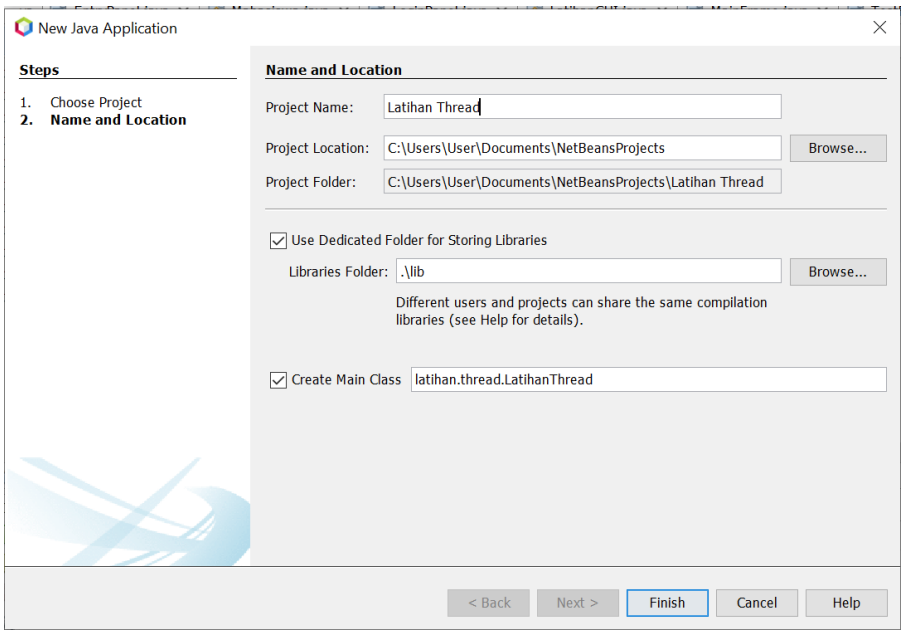
Setelah praktikum pada modul 12 ini diharapkan mahasiswa memahami konsep concurrency dan dapat menerapkannya pada pemrograman berorientasi objek.

12.3. Material Praktikum

Kegiatan pada modul 12 ini memerlukan material berupa Software IDE Netbeans[12].

12.4. Kegiatan Praktikum

Buatla project baru pada netbeans dengan nama Latihan Thread.



Kemudian pada file LatihanThread.java modifikasilah menjadi seperti berikut:

[12] https://netbeans.apache.org/download/index.html

```
package latihan.thread;

/**
 *
 * @author User
 */
public class LatihanThread {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        Thread t = new Thread(new Runnable(){
            @Override
            public void run() {
                for (int i=0; i<10;i++){
                    System.out.println("Thread t: "+ i );
                }
            }
        });
        t.start();
        for (int i=0; i<10;i++){
            System.out.println("Main Thread: "+ i );
        }
    }
}
```

Ketika dijalankan maka outputnya sebagai berikut:

```
run:

Main Thread: 0

Main Thread: 1

Thread t: 0

Main Thread: 2

Thread t: 1

Main Thread: 3
```

```
    Main Thread: 4

    Thread t: 2

    Main Thread: 5

    Main Thread: 6

    Main Thread: 7

    Thread t: 3

    Main Thread: 8

    Main Thread: 9

    Thread t: 4

    Thread t: 5

    Thread t: 6

    Thread t: 7

    Thread t: 8

    Thread t: 9

BUILD SUCCESSFUL (total time: 0 seconds)
```

Dari hasil run program di atas terlihat bahwa proses Thread t dan main thread berjalan secara ”bersamaan”. Main thread tidak menunggu Thread t selesai dieksekusi baru dijalankan.

Terdapat 2 cara untuk mengimplementasikan Thread pada Java. Yang pertama adalah dengan meng-implement interface Runnable, dan yang lainnya adalah dengan meng-extend kelas Thread.

Buatlah kelas java baru dengan nama ThreadA.java, salinlah kode berikut ini

```
package latihan.thread;

/**
 *
 * @author User
 */
public class ThreadA implements Runnable{
```

```
    @Override
    public void run() {
        for (int i=0; i<10;i++){
                    System.out.println("Thread A: "+ i );
                }
    }
}
```

Kemudian buatlah kelas baru dengan nama ThreadB.java dengan kode berikut:

```
package latihan.thread;

/**
 *
 * @author User
 */
public class ThreadB extends Thread{
    @Override
    public void run(){
        for (int i=0; i<10;i++){
                    System.out.println("Thread B: "+ i );
                }
    }
}
```

Kemudian modifikasilah kelas utama LatihanThread menjadi seperti berikut:

```
package latihan.thread;

/**
 *
 * @author User
 */
public class LatihanThread {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
```

```
        Thread t = new Thread(new Runnable(){
            @Override
            public void run() {
                for (int i=0; i<10;i++){
                    System.out.println("Thread t: "+ i );
                }
            }
        });
        t.start();

        Thread t2 = new Thread(new ThreadA());
        t2.start();

        Thread t3 = new ThreadB();
        t3.start();

        for (int i=0; i<10;i++){
            System.out.println("Main Thread: "+ i );
        }
    }
}
```

Ketika dijalankan maka outputnya adalah sebagai berikut:

```
run:
Main Thread: 0
Thread A: 0
Thread A: 1
Main Thread: 1
Main Thread: 2
Main Thread: 3
Main Thread: 4
Thread A: 2
Thread B: 0
Main Thread: 5
Thread B: 1
Thread t: 0
Thread A: 3
Thread t: 1
Thread B: 2
```

```
Main Thread: 6
Main Thread: 7
Thread B: 3
Thread t: 2
Thread A: 4
Thread t: 3
Thread t: 4
Thread B: 4
Main Thread: 8
Thread B: 5
Thread B: 6
Thread B: 7
Thread B: 8
Thread B: 9
Thread t: 5
Thread A: 5
Thread t: 6
Main Thread: 9
Thread t: 7
Thread A: 6
Thread t: 8
Thread t: 9
Thread A: 7
Thread A: 8
Thread A: 9
BUILD SUCCESSFUL (total time: 0 seconds)
```

Terlihat bahwa keempat thead yaitu A,B,C dan main Thread berjalan secara ”bersamaan”.

12.5. Penugasan

Laporkan hasil praktikum berikut dengan hasil penugasan dan penjelasannya ke Dosen dalam bentuk file pdf dengan format nama *<<nim>>_modul12.*

1) Lengkapi kode semua objek
2) Tangkapan layar hasil kode yang dilengkapi
3) Tangkapan layar hasil running

MODUL 13 NETWORKING BAGIAN 1

13.1. Deskripsi Singkat

Package java.net dapat digunakan oleh program java untuk berkomunikasi dalm jaringan menggunakan TCP maupun UDP. Pada modul 13 ini akan dibahas pemrograman Socket dengan memanfaatkan kelas ServerSocket dan Socket.

13.2. Tujuan Praktikum

Setelah praktikum pada modul 13 ini diharapkan mahasiswa mampu menerapkan pemrograman socket pada java

13.3. Material Praktikum

Kegiatan pada modul 13 ini memerlukan material berupa Software IDE Netbeans[13].

13.4. Kegiatan Praktikum

Buatlah project baru pada Netbeans dengan nama Latihan Server. Pada LatihanServer.java ikutilah kode berikut ini

```
/*
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
default.txt to change this license
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit
this template
 */
package latihan.server;


import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.logging.Level;
import java.util.logging.Logger;
```

[13] https://netbeans.apache.org/download/index.html

```
/**
 *
 * @author User
 */
public class LatihanServer {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        int portNumber = 4444;
        try (
                ServerSocket serverSocket = new
ServerSocket(portNumber);
                Socket clientSocket = serverSocket.accept();
                PrintWriter out = new
PrintWriter(clientSocket.getOutputStream(), true);
                BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));) {

            System.out.println("Server Receive: " +
in.readLine());
            out.println("Pesan Diterima");

        } catch (IOException ex) {

Logger.getLogger(LatihanServer.class.getName()).log(Level.SEVERE,
null, ex);
        }
    }

}
```

Kemudian buatlah satu kelas baru bernama LatihanClient.java kemudian sesuaikan dengan kode di bawah ini

```
/*
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
default.txt to change this license
```

```
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to
edit this template
 */
package latihan.server;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.Socket;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author User
 */
public class LatihanClient {
    public static void main(String args[]){
    String hostName = "localhost";
    int portNumber = 4444;
    try (
        Socket echoSocket = new Socket(hostName, portNumber);
        PrintWriter out = new
PrintWriter(echoSocket.getOutputStream(),true);
        BufferedReader in = new BufferedReader(new
InputStreamReader(echoSocket.getInputStream()));
        BufferedReader stdIn = new BufferedReader(new
InputStreamReader(System.in));){

       out.println("Halo");
       System.out.println("Client receive: "+ in.readLine());

    }catch (IOException ex) {

Logger.getLogger(LatihanClient.class.getName()).log(Level.SEVERE,
null, ex);
    }
    }
}
```

Kemudian jalankan LatihanServer.java setelah itu jalankan juga LatihanClient.java

Maka server dan client sudah dapat berkomunikasi. Berikut adalah outputnya

```
Latihan Server (run) ×   Latihan Server (run) #2 ×

run:
Server Receive: Halo
BUILD SUCCESSFUL (total time: 12 seconds)
```

```
Latihan Server (run) ×   Latihan Server (run) #2 ×

run:
Client receive: Pesan Diterima
BUILD SUCCESSFUL (total time: 0 seconds)
```

Kemudian kita modifikasi sedikit pada LatihanServer.java supaya server tetap menerima pesan hingga client mengetikkan teks tertentu

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit this template
 */
package latihan.server;


import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.logging.Level;
import java.util.logging.Logger;


/**
 *
 * @author User
 */
```

```
public class LatihanServer {


    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        int portNumber = 4444;
        try (
                ServerSocket serverSocket = new
ServerSocket(portNumber);
                Socket clientSocket = serverSocket.accept();
                PrintWriter out = new
PrintWriter(clientSocket.getOutputStream(), true);
                BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));) {


            String msg;
            while(!(msg =
in.readLine()).equalsIgnoreCase("exit")){
                System.out.println("Server Receive: " + msg);
                out.println("Pesan Diterima");
            }
        } catch (IOException ex) {

Logger.getLogger(LatihanServer.class.getName()).log(Level.SEVERE,
null, ex);
        }
    }


}
```

Untuk LatihanClient.java dapat kita modifikasi menjadi berikut ini

```
/*
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
default.txt to change this license
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to
edit this template
 */
package latihan.server;
```

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.Socket;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author User
 */
public class LatihanClient {
    public static void main(String args[]){
    String hostName = "localhost";
    int portNumber = 4444;
    try (
        Socket echoSocket = new Socket(hostName, portNumber);
        PrintWriter out = new
PrintWriter(echoSocket.getOutputStream(),true);
        BufferedReader in = new BufferedReader(new
InputStreamReader(echoSocket.getInputStream()));
        BufferedReader stdIn = new BufferedReader(new
InputStreamReader(System.in));){

       String msg;
       while((msg = stdIn.readLine()) != null){
            out.println(msg);
            if(msg.equalsIgnoreCase("exit")) break;
            System.out.println("Client receive: "+ in.readLine());
       }
    }catch (IOException ex) {

Logger.getLogger(LatihanClient.class.getName()).log(Level.SEVERE,
null, ex);
    }
    }
}
```

Jalankan LatihanServer.java kemudian jalankan juga LatihanClient.java. Kemudian ketikkan pesan dari Client ke server. Selama client idak menuliskan ”exit” maka client dapat terus mengirimkan pesan apapun ke server

```
Latihan Server (run) ×   Latihan Server (run) #2 ×

run:
Server Receive: Halo
Server Receive: Apa kabar?
BUILD SUCCESSFUL (total time: 20 seconds)
```

```
Latihan Server (run) ×   Latihan Server (run) #2 ×

run:
Halo
Client receive: Pesan Diterima
Apa kabar?
Client receive: Pesan Diterima
exit
BUILD SUCCESSFUL (total time: 17 seconds)
```

13.5. Penugasan

Laporkan hasil praktikum berikut dengan hasil penugasan dan penjelasannya ke Dosen dalam bentuk file pdf dengan format nama *<<nim>>_modul13.*

1) Lengkapi kode semua objek
2) Tangkapan layar hasil kode yang dilengkapi
3) Tangkapan layar hasil running

MODUL 14 NETWORKING BAGIAN 2

14.1. Deskripsi Singkat

Pada modul 14 ini kita akan mengembangkan kode pada modul 13 supaya Server dapat menerima koneksi dari banyak client dengan menerapkan concurrency. Selain itu, pesan yang dikirimkan antara client ke server adalah object dari kelas Pesan yang menerapkan interface Serializable.

14.2. Tujuan Praktikum

Setelah praktikum pada modul 14 ini diharapkan mahasiswa dapat menerapkan socket programming yang dikombinasikan dengan Object Serialization dan juga Threading.

14.3. Material Praktikum

Kegiatan pada modul 14 ini memerlukan material berupa Software IDE Netbeans[14].

14.4. Kegiatan Praktikum

Pada project Latihan Server, buatlah kelas baru bernama Pesan dengan kode sebagai berikut (**Pesan.java**)

```
/*
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
default.txt to change this license
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to
edit this template
 */
package latihan.server;


import java.io.Serializable;


/**
 *
 * @author User
 */
```

[14] https://netbeans.apache.org/download/index.html

```
public class Pesan implements Serializable{
    private String nama;
    private String pesan;

    public Pesan(String nama, String pesan) {
        this.nama = nama;
        this.pesan = pesan;
    }

    @Override
    public String toString(){
        return "["+nama+"] "+ pesan;
    }

    /**
     * @return the nama
     */
    public String getNama() {
        return nama;
    }

    /**
     * @param nama the nama to set
     */
    public void setNama(String nama) {
        this.nama = nama;
    }

    /**
     * @return the pesan
     */
    public String getPesan() {
        return pesan;
    }

    /**
     * @param pesan the pesan to set
     */
```

```
    public void setPesan(String pesan) {
        this.pesan = pesan;
    }
}
```

Kemudian modifikasilah **LatihanClient.java** menyesuaikan kode berikut ini

```
/*
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
default.txt to change this license
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to
edit this template
 */
package latihan.server;

import java.io.BufferedOutputStream;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.ObjectOutputStream;
import java.io.PrintWriter;
import java.net.Socket;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author User
 */
public class LatihanClient {
    public static void main(String args[]){
    String hostName = "localhost";
    int portNumber = 4444;
    try (
        Socket echoSocket = new Socket(hostName, portNumber);
        BufferedOutputStream bos = new
BufferedOutputStream(echoSocket.getOutputStream());
        ObjectOutputStream oos = new ObjectOutputStream(bos);
        BufferedReader in = new BufferedReader(new
InputStreamReader(echoSocket.getInputStream()));
```

```
        BufferedReader stdIn = new BufferedReader(new
InputStreamReader(System.in));){


        String msg;
        while((msg = stdIn.readLine()) != null){
            oos.writeObject(new Pesan("Andi", msg));
            oos.flush();
            System.out.println("Client receive: "+ in.readLine());
            if(msg.equalsIgnoreCase("exit")) break;


        }
    }catch (IOException ex) {

Logger.getLogger(LatihanClient.class.getName()).log(Level.SEVERE,
null, ex);
    }
    }
}
```

Kemudian modifikasilah **LatihanServer.java** menyesuaikan kode di bawah ini

```
/*
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
default.txt to change this license
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit
this template
 */
package latihan.server;


import java.io.BufferedInputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.logging.Level;
import java.util.logging.Logger;


/**
 *
```

```
 * @author User
 */
public class LatihanServer {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        int portNumber = 4444;
        try (
                ServerSocket serverSocket = new
ServerSocket(portNumber);
                Socket clientSocket = serverSocket.accept();
                PrintWriter out = new
PrintWriter(clientSocket.getOutputStream(), true);
                BufferedInputStream bis = new
BufferedInputStream(clientSocket.getInputStream());
                ObjectInputStream ois = new
ObjectInputStream(bis);
                )
                 {
            Pesan pesan;
            while((pesan = (Pesan) ois.readObject())!=null){
                System.out.println("Server Receive: " +
pesan.toString());
                out.println("Pesan Diterima");
                if(pesan.getPesan().equalsIgnoreCase("exit"))
break;
            }
        } catch (IOException ex) {

Logger.getLogger(LatihanServer.class.getName()).log(Level.SEVERE,
null, ex);
        } catch (ClassNotFoundException ex) {

Logger.getLogger(LatihanServer.class.getName()).log(Level.SEVERE,
null, ex);
        }
    }
}
```

Kemudian jalankan LatihanServer.java kemudian setelah itu jalankan LatihanClient.java. kita sudah menerapkan object serialization pada pertukaran pesan client dan server. Pesan yang ditukarkan berupa object dari kelas Pesan.

```
Output
Latihan Server (run)   Latihan Server (run) #2

run:
Server Receive: [Andi] halo
Server Receive: [Andi] apa kabar?
Server Receive: [Andi] exit
BUILD SUCCESSFUL (total time: 20 seconds)
```

```
Output
Latihan Server (run)   Latihan Server (run) #2

run:
halo
Client receive: Pesan Diterima
apa kabar?
Client receive: Pesan Diterima
exit
Client receive: Pesan Diterima
BUILD SUCCESSFUL (total time: 17 seconds)
```

Supaya server dapat menerima banyak client, maka kita perlu melakukan penyesuaian.

Pertama tambahkan new Class bernama ServerThread.java sebagai berikut

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package latihan.server;


import java.io.BufferedInputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.PrintWriter;
import java.net.Socket;
```

```
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author User
 */
public class ServerThread extends Thread{
    private Socket clientSocket = null;

    public ServerThread(Socket clientSocket) {
        super();
        this.clientSocket = clientSocket;
    }

    @Override
    public void run(){
        try (
            PrintWriter out = new
PrintWriter(clientSocket.getOutputStream(), true);
            BufferedInputStream bis = new
BufferedInputStream(clientSocket.getInputStream());
            ObjectInputStream ois = new ObjectInputStream(bis);){
                Pesan pesan;
                while((pesan = (Pesan) ois.readObject())!=null){
                    System.out.println("Server Receive: " +
pesan.toString());
                    out.println("Pesan Diterima");
                    if(pesan.getPesan().equalsIgnoreCase("exit"))
break;
            }
        } catch (IOException ex) {

Logger.getLogger(ServerThread.class.getName()).log(Level.SEVERE,
null, ex);
        } catch (ClassNotFoundException ex) {

Logger.getLogger(ServerThread.class.getName()).log(Level.SEVERE,
null, ex);
        } finally {
            if(clientSocket!=null){
```

```
                try {
                    clientSocket.close();
                } catch (IOException ex) {
Logger.getLogger(ServerThread.class.getName()).log(Level.SEVERE,
null, ex);
                }
            }
        }
    }
}
```

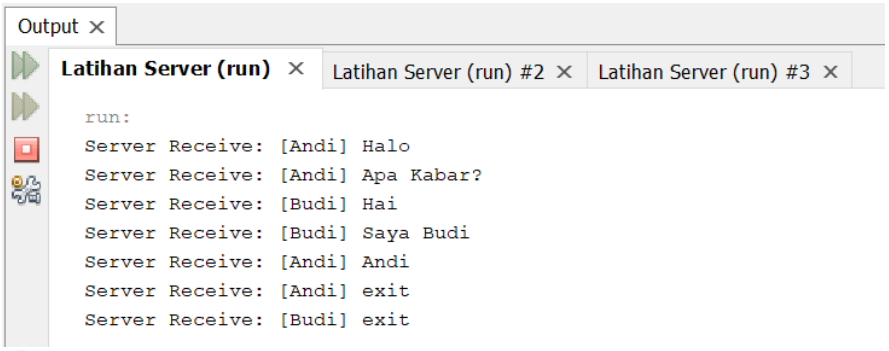
Kemudian modifikasi LatihanServer.java sebagai berikut

```
/*
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
default.txt to change this license
 * Click
nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit
this template
 */
package latihan.server;

import java.io.BufferedInputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author User
 */
public class LatihanServer {

    /**
     * @param args the command line arguments
     */
```

```
    public static void main(String[] args) {
        // TODO code application logic here
        int portNumber = 4444;
        boolean listening = true;
        try (
                ServerSocket serverSocket = new
ServerSocket(portNumber);){
            while(listening){
                Socket clientSocket = serverSocket.accept();
                new ServerThread(clientSocket).start();
            }
        } catch (IOException ex) {

Logger.getLogger(LatihanServer.class.getName()).log(Level.SEVERE,
null, ex);
        }
    }
}
```

Tidak ada perubahan pada kode LatihanClient.java namun ketika mau melakukan run file LatihanClient yang kedua, ubahlah nama Andi menjadi Budi supaya nama pengirimnya berbeda.

Jalankan LatihanServer.java kemudian Jalankan LatihanClient.java dengan nama Andi, kemudian jalankan latihanClient.java dengan nama Budi. Client akan berhenti jika mengetikkan exit sedangkan server masih akanterus mnunggu client baru terkoneksi.



```
Output
Latihan Server (run)   Latihan Server (run) #2   Latihan Server (run) #3
run:
Halo
Client receive: Pesan Diterima
Apa Kabar?
Client receive: Pesan Diterima
Andi
Client receive: Pesan Diterima
exit
Client receive: Pesan Diterima
BUILD SUCCESSFUL (total time: 1 minute 23 seconds)
```

```
Output
Latihan Server (run)   Latihan Server (run) #2   Latihan Server (run) #3
run:
Hai
Client receive: Pesan Diterima
Saya Budi
Client receive: Pesan Diterima
exit
Client receive: Pesan Diterima
BUILD SUCCESSFUL (total time: 50 seconds)
```

14.5. Penugasan

Laporkan hasil praktikum berikut dengan hasil penugasan dan penjelasannya ke Dosen dalam bentuk file pdf dengan format nama *<<nim>>_modul14.*

1) Lengkapi kode semua objek
2) Tangkapan layar hasil kode yang dilengkapi
3) Tangkapan layar hasil running