



Pemrograman Berorientasi Objek

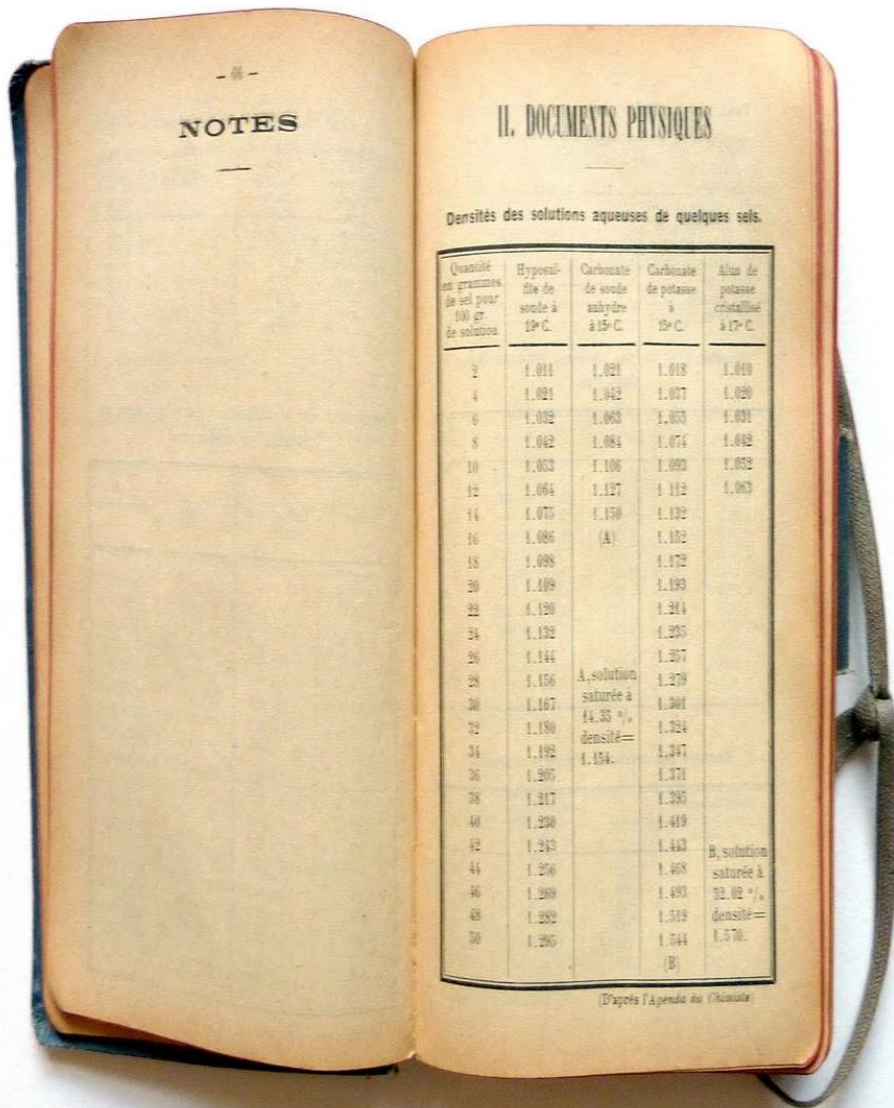


Pertemuan 13

Event Handling pada JAVA GUI

Pemateri : Chrismikha Hardyanto S.Kom., M.Kom.

KONTEN PERKULIAHAN



- Konsep *Event* Handling didalam **JAVA**
- Jenis - Jenis Event pada **JAVA Swing**
- Membuat & Menambahkan Event
- Pratikum: Mengambil nilai pada component **GUI**
- Praktikum: Membuat proses aritmatika sederhana

Setelah sebelumnya kita belajar **membuat tampilan berbasis GUI**, selanjutnya kita akan mencoba memahami bagaimana **membuat aksi didalam program**

Apa itu **Event** didalam program?

Dalam pembuatan **aksi** didalam program **berbasis gui**, Maka ada sebuah konsep yang perlu kita ketahui yaitu **Event**

- ❑ **Aplikasi** berbasis visual/GUI itu bersifat **event driven**.
- ❑ Ketika User **berinteraksi** dengan component GUI, bentuk interaksi tersebut dikenal sebagai **event**. Event akan **memicu (trigger)** program untuk **menjalankan sebuah aksi/proses**.
- ❑ Contoh interaksi yang memicu event :
 - ❑ Klik tombol
 - ❑ Menulis di textfield
 - ❑ Memilih dari check box
 - ❑ Menggerakkan mouse di frame
 - ❑ Memilih menu, dsb



Apa itu **Event** didalam program?

Event handling adalah suatu penanganan terhadap event didalam sebuah aplikasi yang **menentukan bentuk dari aksi** yang akan dikerjakan oleh aplikasi dan juga menentukan dimana dan bagaimana event tersebut dapat dijalankan (oleh interaksi dari pengguna)

Dari definisi diatas, maka dapat disimpulkan ada **3 komponen utama** ketika kita akan melakukan event handling didalam aplikasi (Event Model)



Delegation Event Model

Delegasi event model menguraikan bagaimana program yang kita buat **dapat merespon interaksi** dari user. Untuk memahami model, kita pelajari pertama-tama dengan **tiga komponen** utamanya :

Event Source

Event source mengacu pada **komponen-komponen GUI** didalam program yang akan **men-generate/meghasilkan event**. Sebagai contoh, jika user menekan tombol, maka event source dalam hal ini adalah tombol.

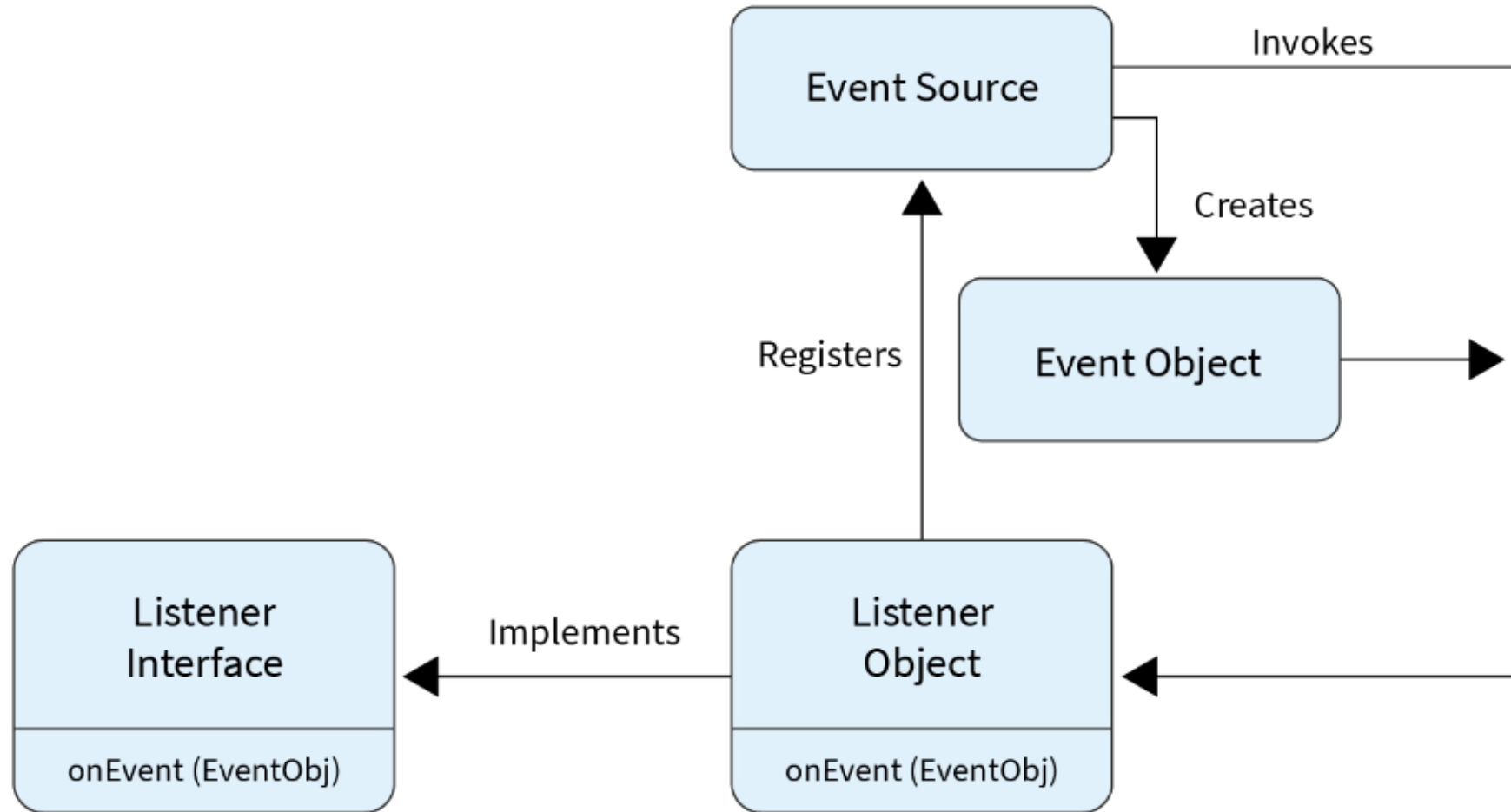
Event Object

Event Object merupakan Objek yang terbentuk **saat terjadi event**. Objek ini akan berisi semua **informasi** yang diperlukan tentang event yang telah terjadi **pada event source**. (**event didalam JAVA dikelompokkan kedalam Class Event**)

Event Listener

Event listener bertugas untuk **menerima berita** dari event-event yang dihasilkan oleh event source dan **memprosesnya** (menjalankan kode program yang telah didefinisikan) untuk menjalankan aksi yang sesuai dengan event tersebut.

Ilustrasi Event Model



[Sumber : google.com]

Class – Class Event di JAVA

Di JAVA, event dibagi kedalam beberapa class event yang merepresentasikan **dimana event** dapat terjadi didalam aplikasi. Memahami jenis event akan dapat membantu programmer dalam menentukan **bentuk interaksi yg dapat dilakukan aplikasi dengan pengguna**

<i>Class Event</i>	<i>Deskripsi</i>
ComponentEvent	Extends <i>AWTEvent</i> . Dijalankan ketika sebuah komponen dipindahkan, di-resize, dibuat <i>visible</i> atau <i>hidden</i> .
InputEvent	Extends <i>ComponentEvent</i> . Abstrak root class event untuk semua komponen-level input class-class event.
ActionEvent	Extends <i>AWTEvent</i> . Dijalankan ketika sebuah tombol ditekan, melakukan double-klik daftar item, atau memilih sebuah menu.
ItemEvent	Extends <i>AWTEvent</i> . Dijalankan ketika sebuah item dipilih atau di-deselect oleh user, seperti sebuah list atau checkbox.

<i>Class Event</i>	<i>Deskripsi</i>
KeyEvent	Extends <i>InputEvent</i> . Dijalankan ketika sebuah <i>key</i> ditekan, dilepas atau diketikkan.
MouseEvent	Extends <i>InputEvent</i> . Dijalankan ketika sebuah tombol mouse ditekan, dilepas, atau di-klik (tekan dan lepas), atau ketika sebuah kursor mouse masuk atau keluar dari bagian visible dari komponen.
TextEvent	Extends <i>AWTEvent</i> . Dijalankan ketika nilai dari text field atau text area dirubah.
WindowEvent	Extends <i>ComponentEvent</i> . Dijalankan sebuah objek <i>Window</i> dibuka, ditutup, diaktifkan, nonaktifkan, <i>iconified</i> , <i>deiconified</i> , atau ketika <i>focus</i> ditransfer kedalam atau keluar window.

Event Listener Di JAVA

Untuk menangkap object yang ditembak oleh event object (ketika terjadi Interaksi oleh pengguna), setiap jenis event umumnya akan memiliki sebuah **event listener** yang didalamnya terdapat **beberapa method yang mendefinisikan bentuk interaksi** yg dapat **memicu event**

❑ Tabel di bawah menunjukkan beberapa **listener di JAVA** yang umum digunakan :

<i>Class Event</i>	<i>Deskripsi</i>
ActionListener	Bereaksi atas perubahan mouse atau keyboard.
MouseListener	Bereaksi atas pergerakan mouse.
MouseMotionListener	Interface MouseMotionListener mendukung MouseListener. Menyediakan method-method yang akan memantau pergerakan mouse, seperti drag dan pemindahan mouse.
WindowListener	Bereaksi atas perubahan window.

❑ Contoh dari entuk interaksi dari mouse event dapat dilihat di slide berikutnya. Untuk event listener lainnya yang dikenal JAVA. Silakan explore di **JAVADOC**

Event Listener Di JAVA

- ❑ Interface `MouseListener` terdiri dari lima method abstract yang merepresentasikan bentuk aksi yang akan memicu(trigger) event dari dalam component

MouseListener Method

`public void mouseClicked(MouseEvent e)`

Dipanggil pada saat tombol mouse di click (seperti tekan dan lepas).

`public void mouseEntered(MouseEvent e)`

Dipanggil pada saat kursor mouse memasuki area komponen.

`public void mouseExited(MouseEvent e)`

Dipanggil pada saat kursor mouse meninggalkan area komponen.

`public void mousePressed(MouseEvent e)`

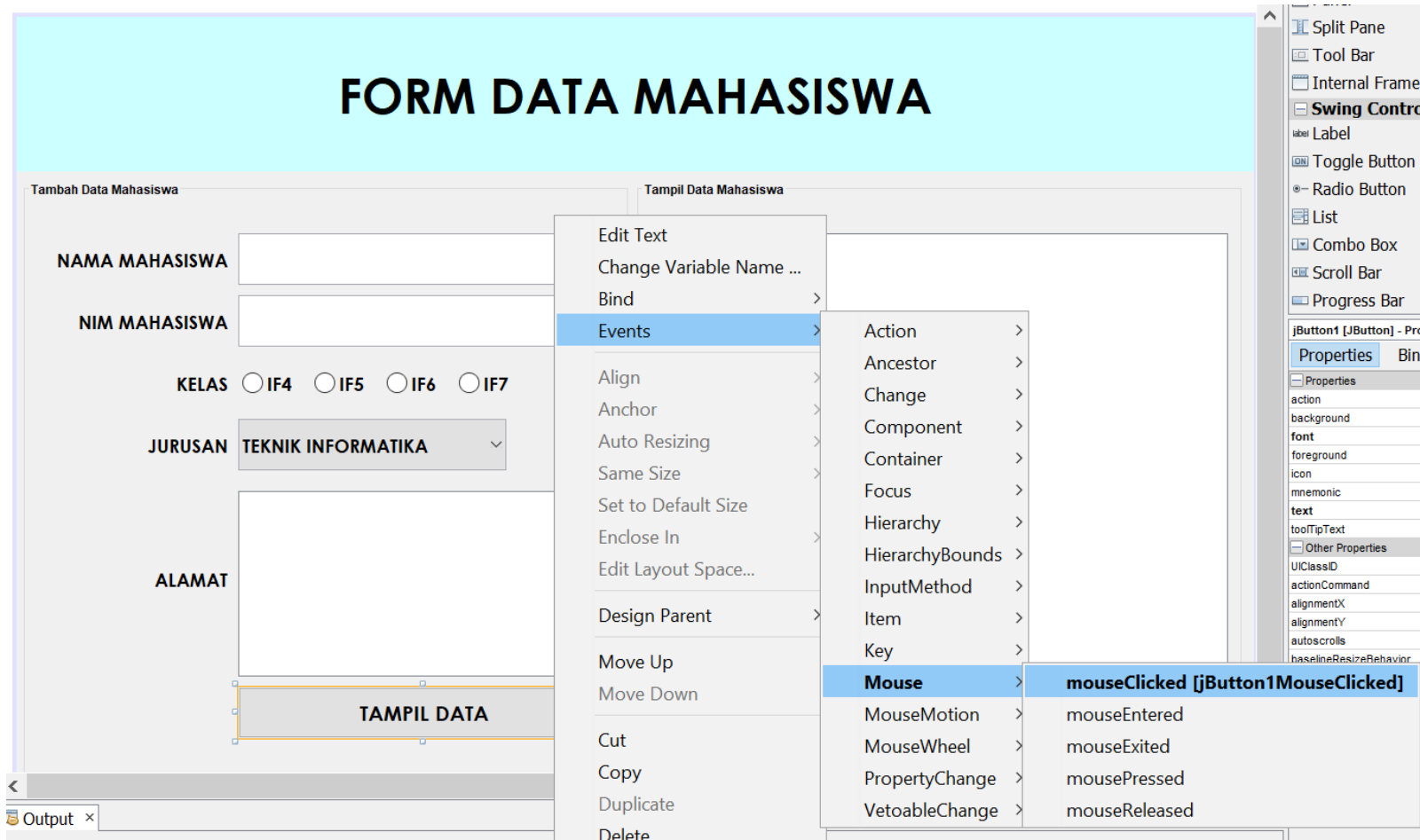
Dipanggil pada saat tombol mouse ditekan di atas komponen

`public void mouseReleased(MouseEvent e)`

Dipanggil pada saat tombol mouse dilepas di atas komponen

Menambahkan Event pada Component

Untuk **Menambahkan suatu event (register event listener)** didalam component GUI, caranya cukup mudah jika kita **menggunakan GUI Builder** didalam pembuata aplikasi.



1. Pilih Component GUI yang akan melempar event untuk ditambahkan event listener.
2. Klik Kanan pada area component tersebut (cth : JButton) -> pilih Event.
3. Didalam event Cari jenis event yang akan terjadi.
4. Tentukan bentuk trigger yang akan memicu event didalam component
5. Event Listener Berhasil ditambahkan. Selanjutnya masuk kedalam mode source dan tuliskan sintak & logika program untuk mendefinisikan aksi/pekerjaan yang akan dilakukan aplikasi

Mari kita coba praktek...

Bagaimana cara kita membuat logika program didalam aplikasi JAVA berbasis GUI ??

Mari kita coba melakukan beberapa praktikum berikut untuk lebih memahami pembuatan program JAVA berbasis GUI



Ada 3 Praktikum Dasar yang akan kita coba pada pertemuan hari ini

- 1) Mengambil & Menampilkan nilai (Value) dari component**
- 2) Menerapkan Konsep PBO sederhana didalam rancangan program**
- 3) Membuat Proses Aritmatika Sederhana**

Praktikum 1A:

Mengambil nilai dari text field

Praktikum 1A

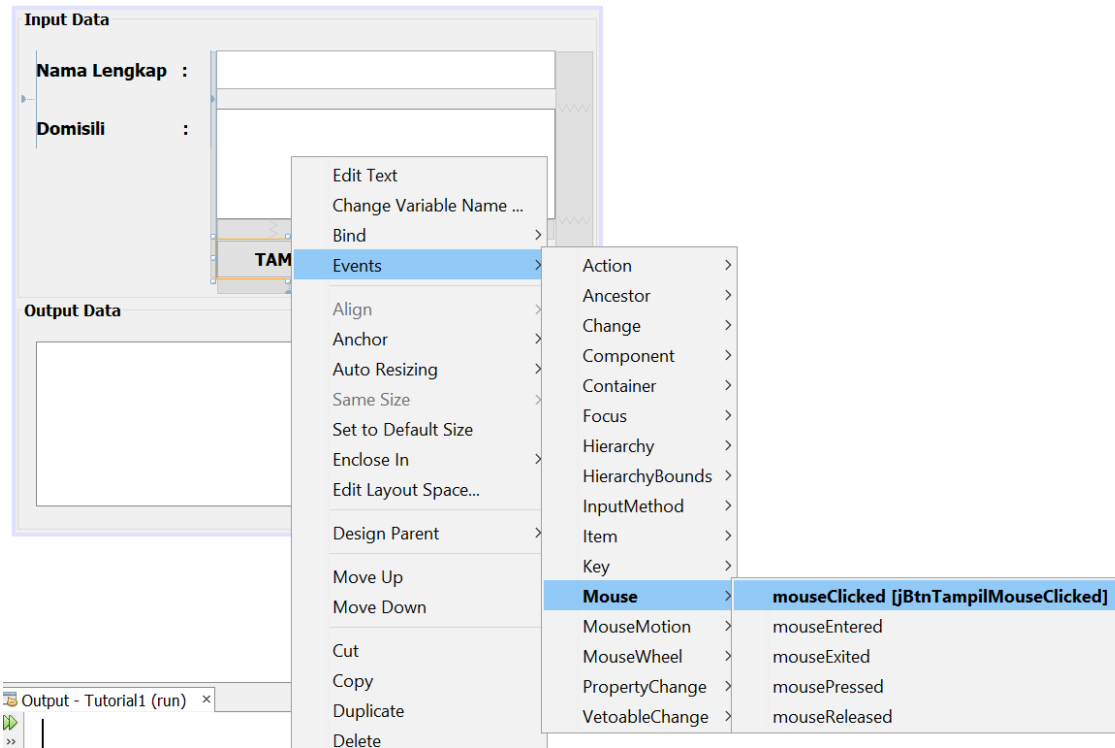
Langkah 1: Buatlah project baru pada netbeans. Selanjutnya tambahkan kelas JFrame dengan cara : **New -> JFrame Form -> Berikan Nama Frame -> finish**

The screenshot shows a Java Swing window with two main sections: 'Input Data' and 'Output Data'. The 'Input Data' section contains a label 'Nama Lengkap :', a text field 'jTextField1', a label 'Domisili :', a text area 'jTextArea1', and a button 'TAMPIL' with 'jButton1'. The 'Output Data' section contains a text area 'jTextArea2'. All input components are highlighted with red rectangles.

- ❑ Ubahlah ukuran frame dengan dimensi **x = 600, y = 600**
- ❑ Tambahkan 2 buah Label dengan jenis border yaitu title border. Label 1 untuk input & label 2 untuk output.
- ❑ Tambahkan **beberapa komponen pada masing – masing label** seperti contoh berikut.
- ❑ Atur lah dimensi dan tata letak komponen didalam label dengan rapi (Silakan diedit properties untuk tiap komponen sesuai kebutuhan)
- ❑ Untuk **JTextField1** ubah nama variabelnya dengan **jTextNama**
- ❑ Untuk **JTextArea1 & 2** ubah nama variabelnya dengan **jTextAlamat & jTextOutput**
- ❑ Untuk **JButton1** ubah nama variabelnya dengan **jBtnTampil**

Praktikum 1A

Langkah 2: Untuk kasus ini kita akan menambahkan event ketika **button tampil** diclick user. Untuk itu kita perlu menambahkan **Listener** untuk menerima object **MouseEvent** dari **button TEKAN (Object Source)**



- ❑ Pada komponen GUI yang akan menerima event (Object Source) : **Klik Kanan -> event untuk menambahkan listener**
- ❑ Pilih listener yang **sesuai dengan jenis event** yang akan **diterima** oleh komponen gui tersebut. (cth pilih Mouse untuk menambahkan MouseListener)
- ❑ Pilih **bentuk event** yang akan dipanggil/dihasilkan oleh Mouse.
- ❑ Misalkan jika event terjadi ketika **mouse di click**, malah **pilih mouseClicked** untuk menambahkan method listener pada komponen GUI

Praktikum 1A

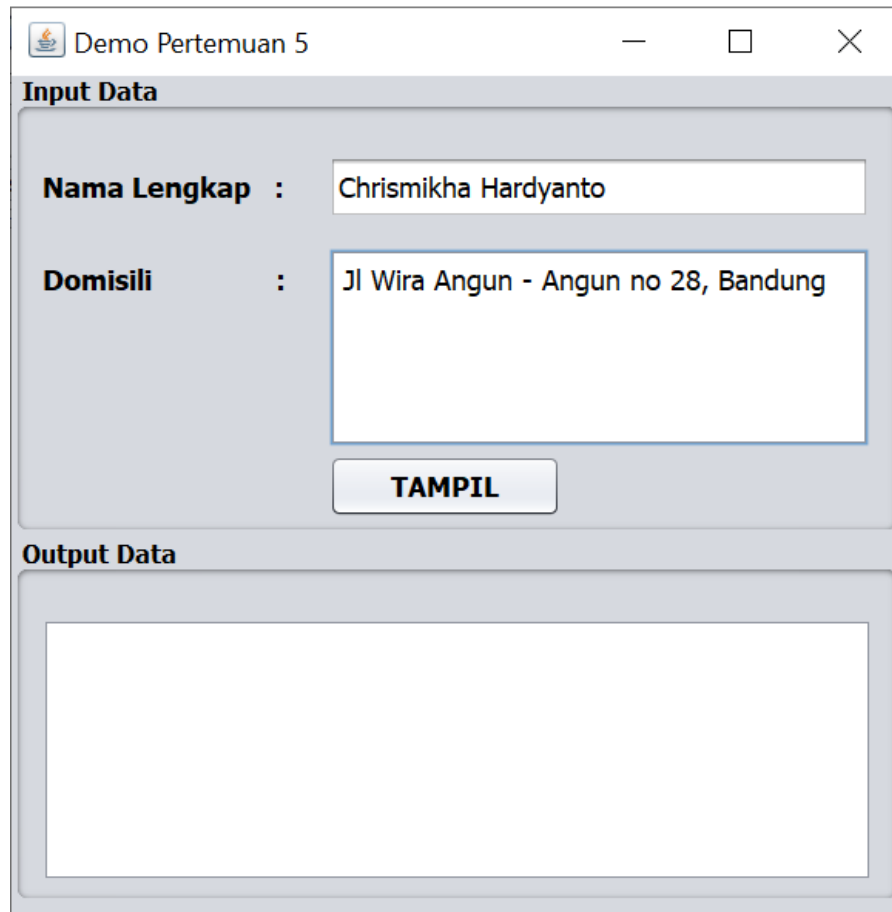
Langkah 3: Tampilan GUI akan berpindah pada Mode Kode(Source). Carilah method event listener yang terbentuk didalam source. Isilah method tersebut dengan kode program untuk melakukan aksi yang ingin dilakukan (**Membuat Event Handler**)

```
151 private void jBtnTampilMouseClicked(java.awt.event.MouseEvent evt) {  
152     // TODO add your handling code here:  
153     String nama;  
154     String alamat;  
155     String hasil;  
156  
157     nama = jTextNama.getText();//Ambil Nilai String dari Field Nama  
158     alamat = jTextAlamat.getText();//Ambil Nilai String dari Field Area Alamat  
159     hasil = "Halo, Nama Saya Adalah "+nama+"\nSaya tinggal di "+alamat; //hasil yang akan ditampilkan pada output  
160  
161     jTextOutput.setText(hasil);  
162 }  
163
```

- ❑ Kode pada **baris 157 & 158** adalah kode untuk mengambil nilai dari komponen JTextField dan JTextArea
- ❑ Kode pada **baris 161** adalah kode untuk menset (menampilkan text) pada komponen JTextArea

Praktikum 1A

Langkah 4: Jalankan Program lalu **Coba click Button TEKAN**. Lihat apa aksi yang terjadi pada program ketika event pada Button aktif.



Demo Pertemuan 5

Input Data

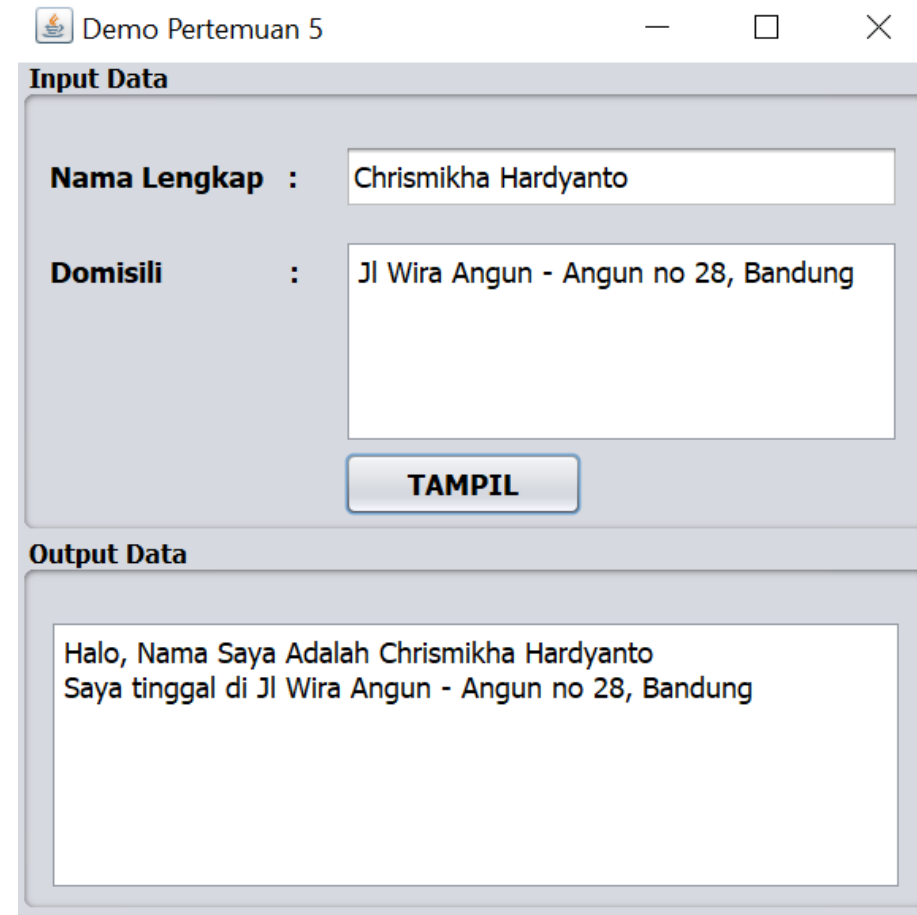
Nama Lengkap : Chrismikha Hardyanto

Domisili : Jl Wira Angun - Angun no 28, Bandung

TAMPIL

Output Data

[Sebelum Button di Click]



Demo Pertemuan 5

Input Data

Nama Lengkap : Chrismikha Hardyanto

Domisili : Jl Wira Angun - Angun no 28, Bandung

TAMPIL

Output Data

Halo, Nama Saya Adalah Chrismikha Hardyanto
Saya tinggal di Jl Wira Angun - Angun no 28, Bandung

[Setelah Button di Click]

Praktikum 1B:

**Mengambil nilai (value) dari
radioButton & ComboBox**

Praktikum 1B

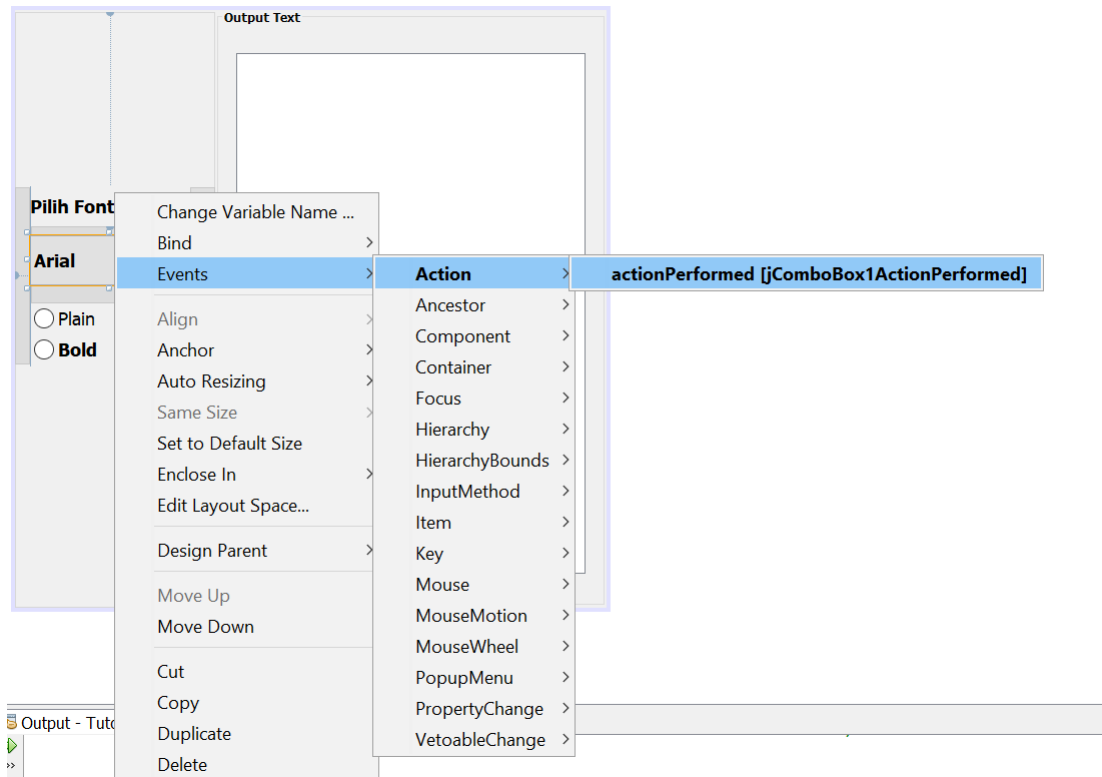
Langkah 1 : Buatlah project dan package baru pada netbeans. Selanjutnya tambahkan kelas JFrame dengan cara : **New -> JFrame Form -> Berikan Nama Frame -> finish**



- ☐ Ubahlah ukuran frame dengan dimensi **x = 600, y = 600**
- ☐ Tambahkan 2 buah Label dengan jenis border yaitu default untuk label 1 dan title border untuk label 2. Berikan keterangan output text pada border label 2
- ☐ Tambahkan **beberapa komponen pada masing – masing label** seperti contoh berikut. (1 label, 1 combo Box, 2 Radio Button, & 1 text area)
- ☐ Atur lah dimensi dan tata letak komponen didalam label dengan rapi (Silakan diedit properties untuk tiap komponen sesuai kebutuhan)
- ☐ Pada combo box berikan opsi pilihan yaitu : **"Arial", "Verdana", "Tahoma."**
- ☐ Sesuaikan nama variable untuk masing – masing komponen. (pada contoh ini namanya default)

Praktikum 1B

Langkah 2: Untuk kasus ini kita akan menambahkan event ketika memilih suatu opsi pada combo box. Untuk itu kita perlu menambahkan **Listener** untuk menerima object **ActionEvent** dari **pemilihan item pada combo box(Object Source)**



- ❑ Pada komponen GUI yang akan menerima event (Object Source) : **Klik Kanan -> event untuk menambahkan listener**
- ❑ Pilih listener yang **sesuai dengan jenis event** yang akan **diterima** oleh komponen gui tersebut. (cth pilih action untuk menambahkan ActionListener)
- ❑ Pilih **bentuk event** yang akan dipanggil/dihasilkan oleh aksi pada komponen. (hanya ada 1 method pada action listener)

Praktikum 1B

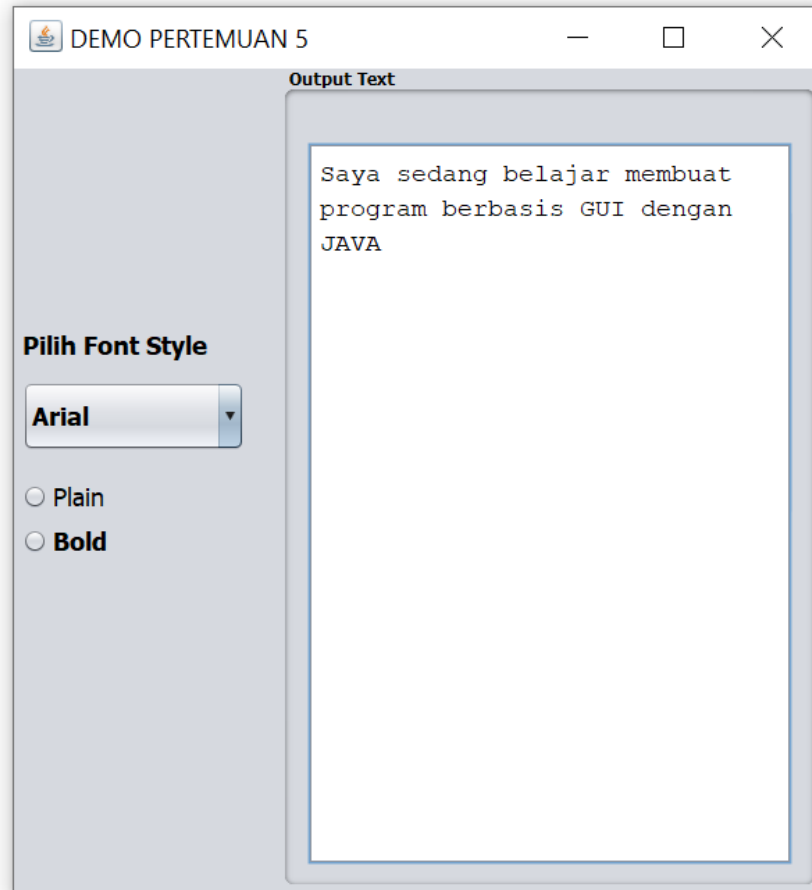
Langkah 3 : Tampilan GUI akan berpindah pada Mode Kode(Source). Carilah method event listener yang terbentuk didalam source. Isilah method tersebut dengan kode program untuk melakukan aksi yang ingin dilakukan (**Membuat Event Handler**)

```
146 |
147 | private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {
148 |     // TODO add your handling code here:
149 |     if(jComboBox1.getSelectedItem() == "Arial"){
150 |         jTextArea1.setFont(new Font("Arial",Font.PLAIN,18));
151 |     }else if(jComboBox1.getSelectedItem() == "Verdana"){
152 |         jTextArea1.setFont(new Font("Verdana",Font.PLAIN, 18));
153 |     }else if(jComboBox1.getSelectedItem() == "Tahoma"){
154 |         jTextArea1.setFont(new Font("Tahoma",Font.PLAIN, 18));
155 |     }
156 |
```

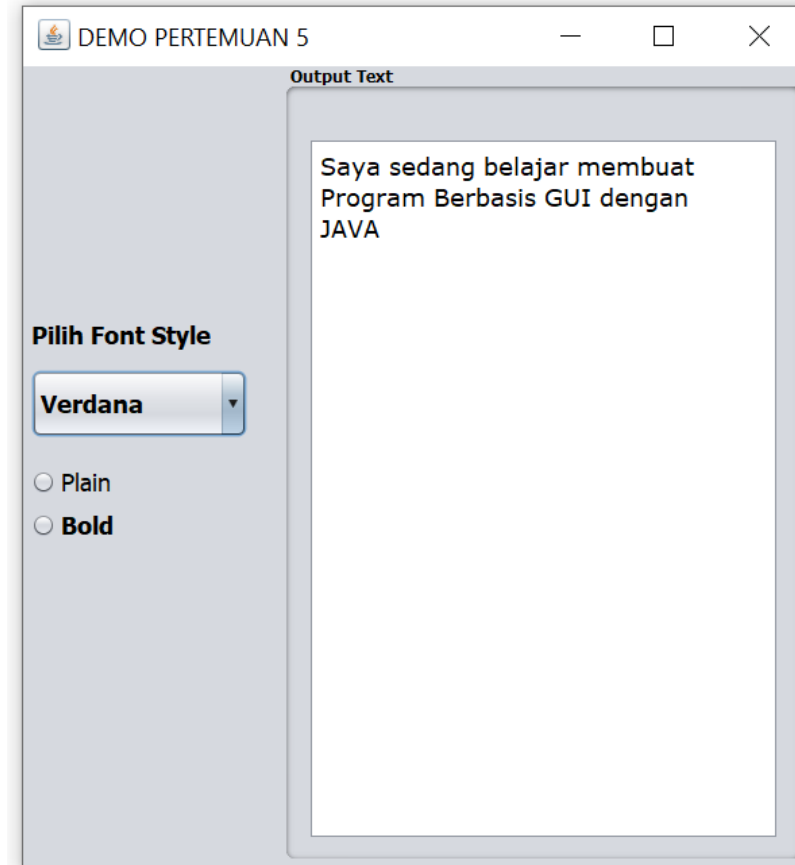
- ❑ Kode pada **baris 149 s/d 150 artinya** jika Opsi item pada combo box yang dipilih adalah "Arial" . Maka Ubah font pada jTextArea1 dengan "Arial".
- ❑ Kode pada **baris 151 s/d 152 artinya** jika Opsi item pada combo box yang dipilih adalah "Verdana" . Maka Ubah font pada jTextArea1 dengan "Verdana".
- ❑ Kode pada **baris 153 s/d 154 artinya** jika Opsi item pada combo box yang dipilih adalah "Tahoma" . Maka Ubah font pada jTextArea1 dengan "Tahoma".

Praktikum 1B

Langkah 4 : Jalankan Program lalu **Coba click Button TEKAN.** Lihat apa aksi yang terjadi pada program ketika event pada Button aktif.



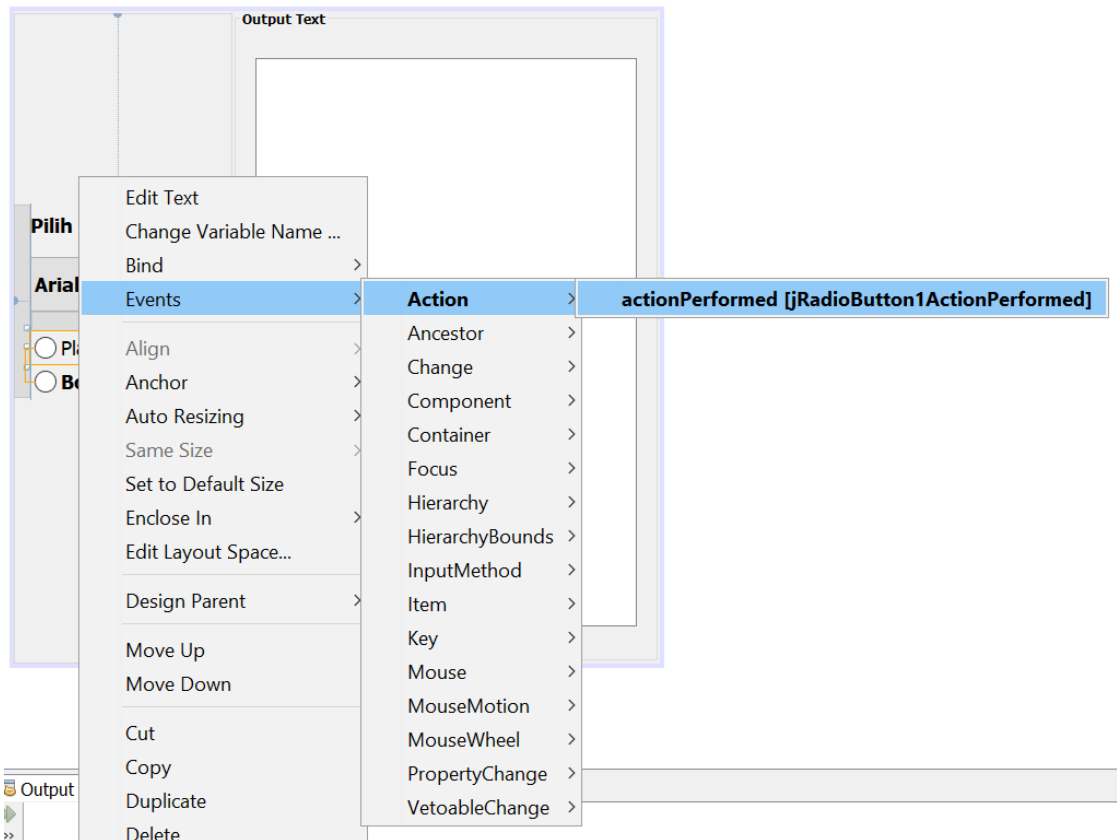
[Sebelum Memilih opsi pada ComboBox]



[Setelah Memilih Opsi Verdana]

Praktikum 1B

Langkah 5 : Selanjutnya kita tambahkan event ketika memilih pilihan di radio button. kita perlu menambahkan **Listener** untuk menerima object **ActionEvent** dari **pemilihan opsi pada radio button(Object Source)**



- ❑ Pada komponen GUI yang akan menerima event (Object Source) : **Klik Kanan -> event untuk menambahkan listener**
- ❑ Pilih listener yang **sesuai dengan jenis event** yang akan **diterima** oleh komponen gui tersebut. (cth pilih action untuk menambahkan ActionListener)
- ❑ Pilih **bentuk event** yang akan dipanggil/dihasilkan oleh aksi pada komponen. (hanya ada 1 method pada action listener)

Praktikum 1B

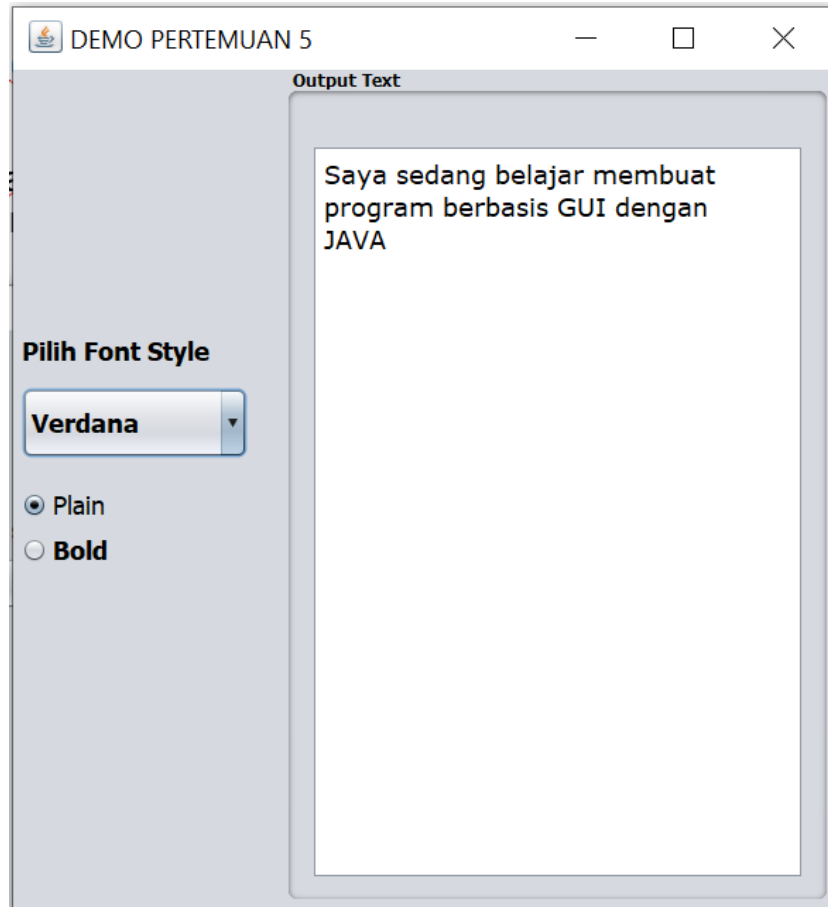
Langkah 6: Tampilan GUI akan berpindah pada Mode Kode(Source). Carilah method event listener untuk radioButton yang terbentuk didalam source. Isilah method tersebut dengan kode program untuk melakukan aksi yang ingin dilakukan (**Membuat Event Handler**)

```
159 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
160     // TODO add your handling code here:  
161     if(jRadioButton1.isSelected()){  
162         jTextArea1.setFont(new Font(jTextArea1.getFont().getName(), Font.PLAIN, 18));  
163     }  
164 }  
165  
166 private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {  
167     // TODO add your handling code here:  
168     if(jRadioButton2.isSelected()){  
169         jTextArea1.setFont(new Font(jTextArea1.getFont().getName(), Font.BOLD, 18));  
170     }  
171 }  
172
```

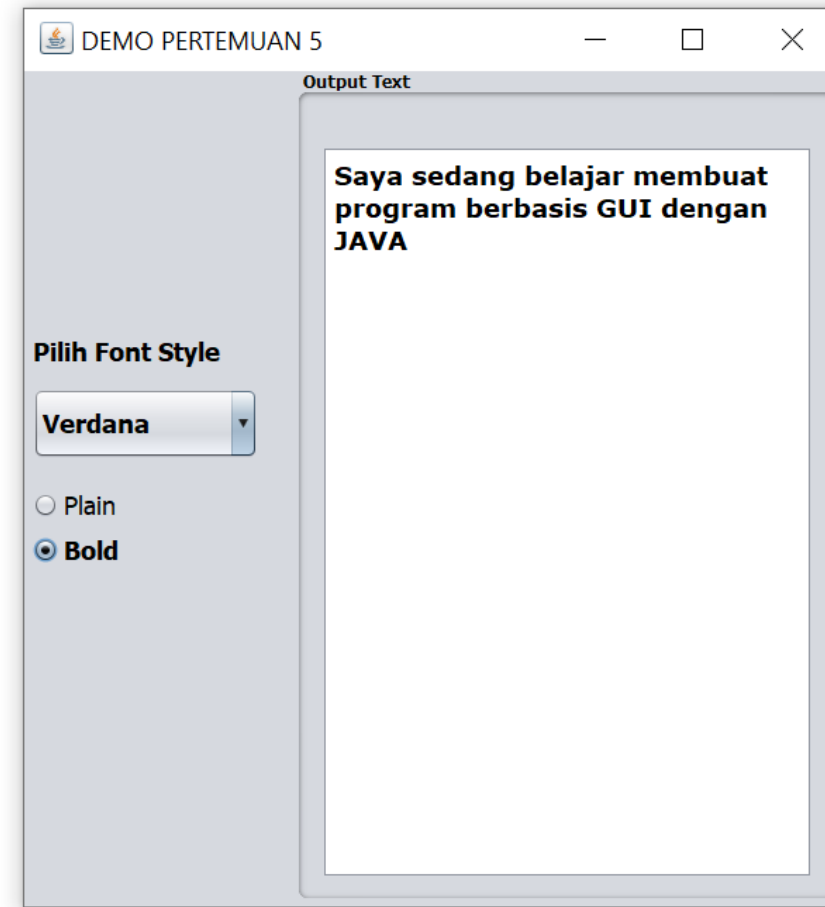
- ❑ Kode pada **baris 159 s/d 164 artinya** jika radio button yang terpilih adalah button dengan nama JRadioButton1. Maka Ubah font pada jTextArea1 menjadi plain . (Style dan size tetap)
- ❑ Kode pada **baris 166 s/d 170 artinya** jika radio button yang terpilih adalah button dengan nama JRadioButton2. Maka Ubah font pada jTextArea1 menjadi bold . (Style dan size tetap)

Praktikum 1B

Langkah 7: Jalankan Program lalu **Coba click Button TEKAN**. Lihat apa aksi yang terjadi pada program ketika event pada Button aktif.



[Memilih opsi pada Plain]



[Setelah Memilih Opsi Bold]

Praktikum 2:

**Menerapkan PBO didalam Program
GUI**

Praktikum 2

Langkah 1 : Buatlah project dan package baru pada netbeans. Selanjutnya tambahkan kelas JFrame dengan cara : **New -> JFrame Form -> Berikan Nama Frame -> finish**

FORM DATA MAHASISWA

Tambah Data Mahasiswa

NAMA MAHASISWA

NIM MAHASISWA

KELAS ☐ IF4 ☐ IF5 ☐ IF6 ☐ IF7

JURUSAN TEKNIK INFORMATIKA

ALAMAT

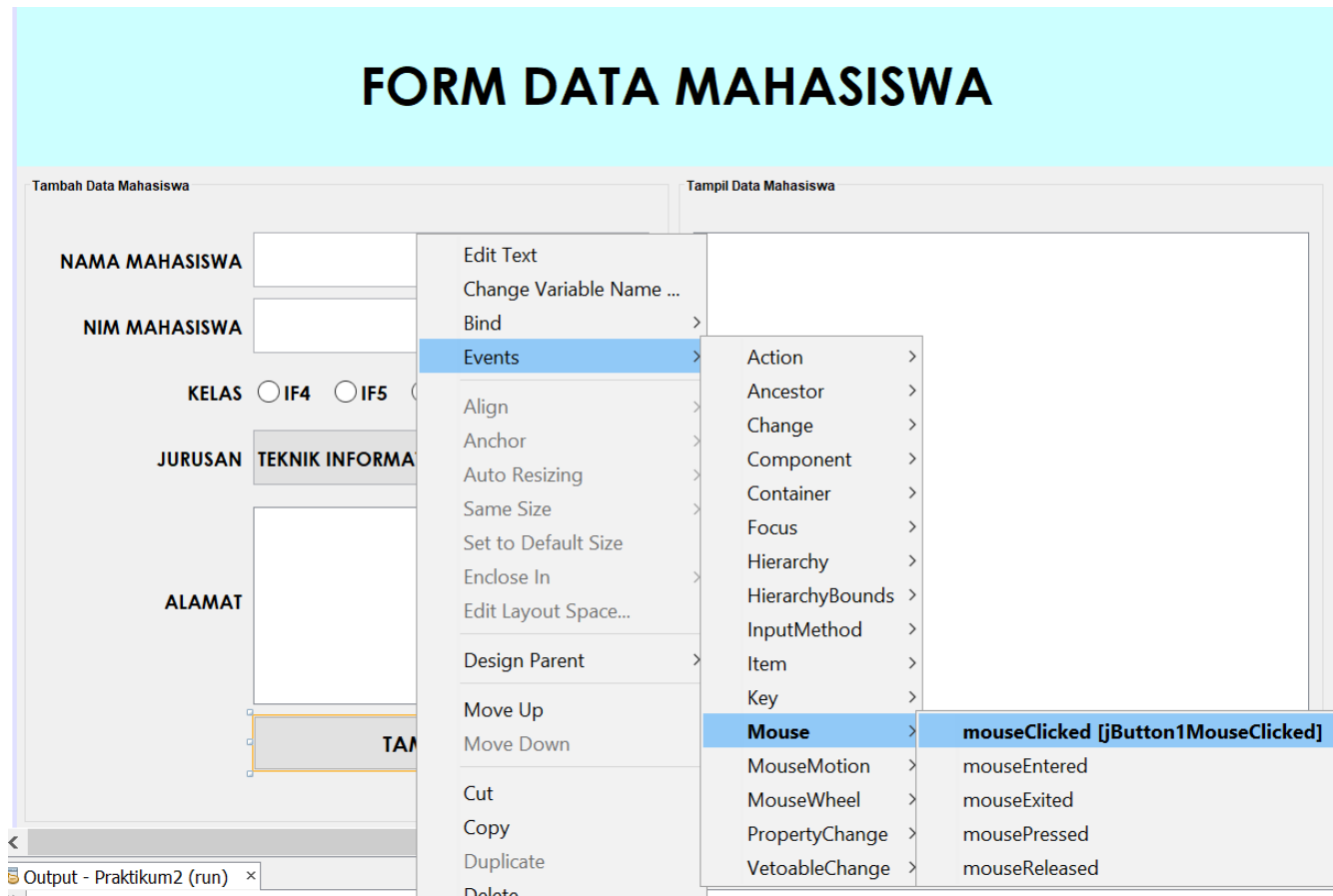
TAMPIL DATA

Tampil Data Mahasiswa

1. Ubah Dimensi / Size dari Frame dengan ukuran **x=1200 & y=800**
2. Gunakan Absolut Layout
3. Frame Selalu berada ditengah
4. Berikan title pada Frame dengan isi = **'Form Mahasiswa'**.
5. Tambahkan sebuah panel sebagai mainPanel
6. Tambahkan **3 buah panel** dan diletakan diatas mainPanel sebagai panelHeader, panelInput, panelOutput
7. Tambahkan component GUI sesuai contoh berikut. Sesuaikan nama variable dari setiap component sesuai kebutuhan

Praktikum 2

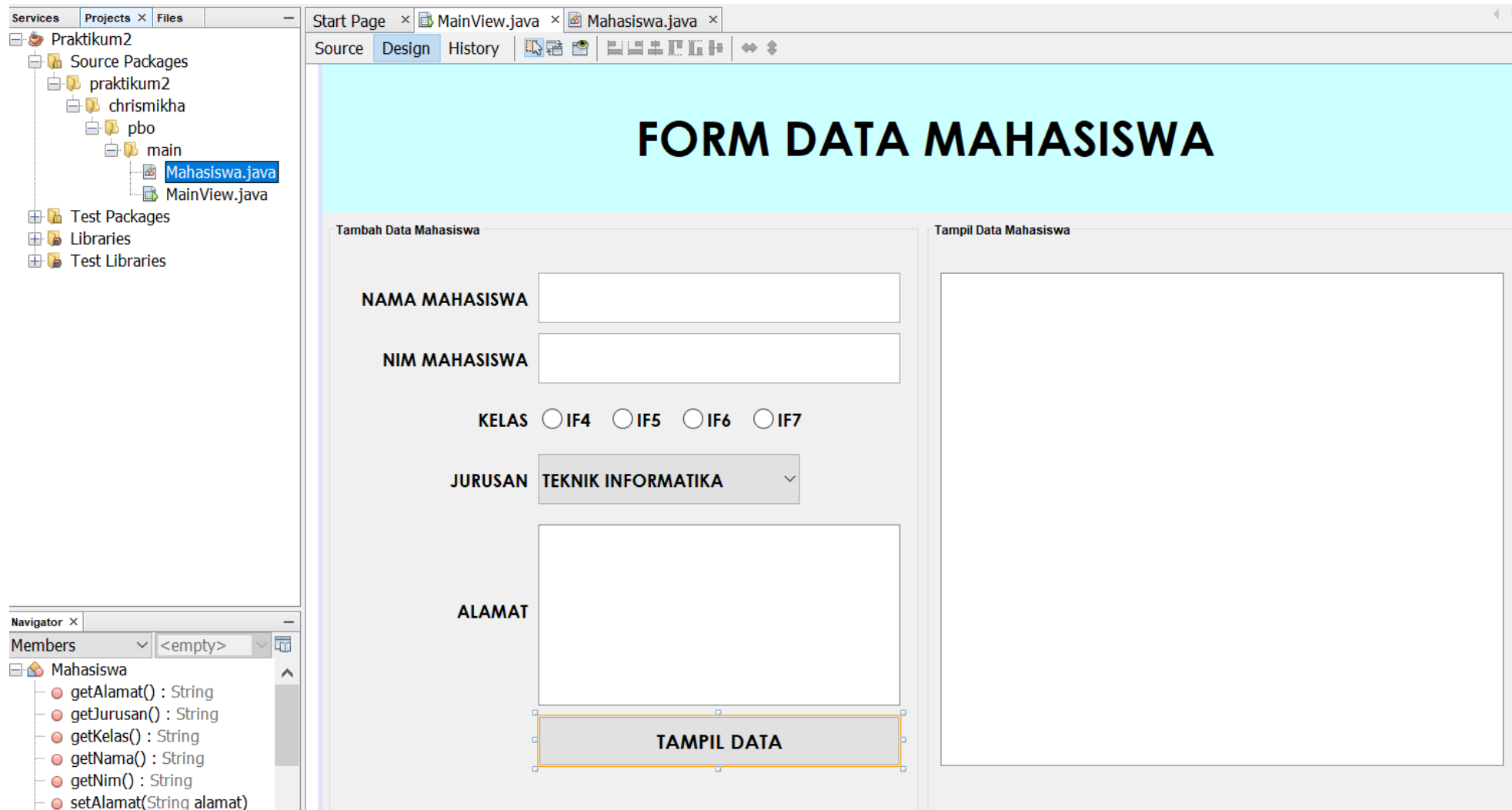
Langkah 2: Untuk kasus ini kita akan menambahkan event ketika **button tampil data** diclick user. Untuk itu kita perlu menambahkan **Listener** untuk menerima object **MouseEvent** dari **button (Object Source)**



- ❑ Pada komponen GUI yang akan menerima event (Object Source) : **Klik Kanan -> event untuk menambahkan listener**
- ❑ Pilih listener yang **sesuai dengan jenis event** yang akan **diterima** oleh komponen gui tersebut. (cth pilih Mouse untuk menambahkan MouseListener)
- ❑ Pilih **bentuk event** yang akan dipanggil/dihasilkan oleh Mouse.
- ❑ Misalkan jika event terjadi ketika **mouse di click**, malah **pilih mouseClicked** untuk menambahkan method listener pada komponen GUI

Praktikum 2

Langkah 3: Tambahkan sebuah class baru (Java Class) dengan nama **Mahasiswa**. Kita akan coba menerapkan konsep PBO dasar didalam aplikasi ini.



Praktikum 2

Langkah 4: Masuk kedalam mode source dari JFrame. Disini , karena kita akan menerapkan konsep PBO maka kita perlu membuat **method getter** untuk **setiap component yang nilainya akan diambil dan digunakan di class lain** .

```
37 //Tambahkan Method Getter untuk Component yang nilainya akan diam
38 public JComboBox<String> getBoxJurusan() {
39     return boxJurusan;
40 }
41 public JRadioButton getBtnKelas1() {
42     return btnKelas1;
43 }
44 public JRadioButton getBtnKelas2() {
45     return btnKelas2;
46 }
47 public JRadioButton getBtnKelas3() {
48     return btnKelas3;
49 }
50 public JRadioButton getBtnKelas4() {
51     return btnKelas4;
52 }
53 public JTextArea getTxtAlamat() {
54     return txtAlamat;
55 }
56 public JTextField getTxtNama() {
57     return txtNama;
58 }
59 public JTextField getTxtNim() {
60     return txtNim;
61 }
62 public JTextArea getTxtTampilData() {
63     return txtTampilData;
64 }
65 }
```

[Gunakan alt + insert -> getter]

Praktikum 2

Langkah 5: Masuk kedalam Class Mahasiswa, **tambahkan 5 atribut** untuk menampung nilai yang diambil dari JFrame. Jangan lupa **buatkan method getter dan setternya**

```
1  package praktikum2.chrismikha.pbo.main;
2
3  public class Mahasiswa {
4      private String nama;
5      private String nim;
6      private String kelas;
7      private String jurusan;
8      private String alamat;
9
10     public String getNama() { ...3 lines }
13     public void setNama(String nama) { ...3 lines }
16
17     public String getNim() { ...3 lines }
20     public void setNim(String nim) { ...3 lines }
23
24     public String getKelas() { ...3 lines }
27     public void setKelas(String kelas) { ...3 lines }
30
31     public String getJurusan() { ...3 lines }
34     public void setJurusan(String jurusan) { ...3 lines }
37
38     public String getAlamat() { ...3 lines }
41     public void setAlamat(String alamat) { ...3 lines }
44
```

Praktikum 2

Langkah 6: Tambahkan method didalam class Mahasiswa sesuai dengan aksi yang akan dikerjakan oleh aplikasi. Dalam kasus ini adalah aksi untuk menampilkan seluruh inputan dari form mahasiswa. Berikut adalah contoh kode programnya :

```
45 //Method untuk aksi pada object Mahasiswa
46 public void tampilDataMahasiswa(MainView view){
47     //Ambil nilai dari setiap component GUI di MainView
48     this.setNama(view.getTxtNama().getText());
49     this.setNim(view.getTxtNim().getText());
50     this.setAlamat(view.getTxtAlamat().getText());
51
52     //Sintak Untuk mengambil nilai diRadio Button
53     if(view.getBtnKelas1().isSelected()){
54         this.setKelas(view.getBtnKelas1().getText());
55     }
56     if(view.getBtnKelas2().isSelected()){
57         this.setKelas(view.getBtnKelas2().getText());
58     }
59     if(view.getBtnKelas3().isSelected()){
60         this.setKelas(view.getBtnKelas4().getText());
61     }
62     if(view.getBtnKelas4().isSelected()){
63         this.setKelas(view.getBtnKelas4().getText());
64     }
65 }
```

Praktikum 2

Langkah 6: Lanjutan ...

```
66 //Sintak Untuk mengambil nilai diComboBox
67 if(view.getBoxJurusan().getSelectedIndex() == 0){
68     this.setJurusan("TEKNIK INFORMATIKA");
69 } else if(view.getBoxJurusan().getSelectedIndex() == 1){
70     this.setJurusan("SISTEM INFORMASI");
71 } else if(view.getBoxJurusan().getSelectedIndex() == 2){
72     this.setJurusan("TEKNIK KOMPUTER");
73 }
74
75 //Menampilkan Data mahasiswa kedalam Panel Output
76 view.getTxtTampilData().setText("NAMA MAHASISWA      : " +this.nama
77                                  +"\nNIM MAHASISWA      : " +this.nim
78                                  +"\nKELAS MAHASISWA     : " +this.kelas
79                                  +"\nJURUSAN            : " +this.jurusan
80                                  +"\nALAMAT MAHASISWA    : " +this.alamat);
81
82 }
```

Praktikum 2

Langkah 7: Masuk kembali kedalam Class MainView (Jframe). Untuk dapat berkomunikasi dengan Class mahasiswa, kita perlu membuat objectnya .


```
24 //Deklarasi Class Mahasiswa
25 Mahasiswa mahasiswa;
26
27 public MainView() {
28     //Buat object dari kelas Mahasiswa
29     mahasiswa = new Mahasiswa();
30
31     initComponents();
32
33     setExtendedState(JFrame.MAXIMIZED_HORIZ);
34     setVisible(true);
35     setResizable(false);
36 }
37
```

Praktikum 2

Langkah 8: Langkah terakhir, kita komunikasikan method tampilDataMahasiswa didalam listener mouseClicked pada JFrame. Panggilan method tersebut didalam listener

```
231
232 private void btnTampilMouseClicked(java.awt.event.MouseEvent evt) {
233     // TODO add your handling code here:
234     mahasiswa.tampilDataMahasiswa(this);
235 }
236
```


Praktikum 2 (Hasil Akhir)

 FORM DATA MAHASISWA

FORM DATA MAHASISWA

Tambah Data Mahasiswa

NAMA MAHASISWA

Chrismikha Hardyanto

NIM MAHASISWA

10110259

KELAS

☐ IF4 ☐ IF5 ☒ IF6 ☐ IF7

JURUSAN

TEKNIK INFORMATIKA

ALAMAT

Margahayu

TAMPIL DATA

Tampil Data Mahasiswa

NAMA MAHASISWA : Chrismikha Hardyanto

NIM MAHASISWA : 10110259

KELAS MAHASISWA : IF6

JURUSAN : TEKNIK INFORMATIKA

ALAMAT MAHASISWA : Margahayu

Praktikum 3:

Membuat Proses Aritmatika

Menangani Operasi Aritmatika

- ❑ Ketika mengambil nilai (value) dari component GUI (khususnya pada JTextField) , nilai yang didapatkan selalu **bertipe data String**
- ❑ Masalahnya tipe data String **tidak dapat digunakan** untuk melakukan operasi aritmatika. Operasi matematika hanya bisa menggunakan tipe data **bilangan(number)** atau **pecahan(double)**
- ❑ Solusinya setiap nilai yang akan digunakan untuk operasi matematika harus **diubah tipenya** terlebih dahulu



Konversi Tipe Data

- ❑ Ada **beberapa cara** yang dapat digunakan untuk melakukan mengubah tipe data **menjadi tipe data tertentu** di JAVA
- ❑ Salah satu caranya adalah memanfaatkan method **valueOf()**.
- ❑ Bentuk umum dari pendeklarasian method valueOf pada JAVA adalah sebagai berikut :

```
ClassTipeData.valueOf(Nilai);
```

Contoh :

```
String.valueOf(Nilai); {konversi menjadi String}
```



Mari kita coba praktek...

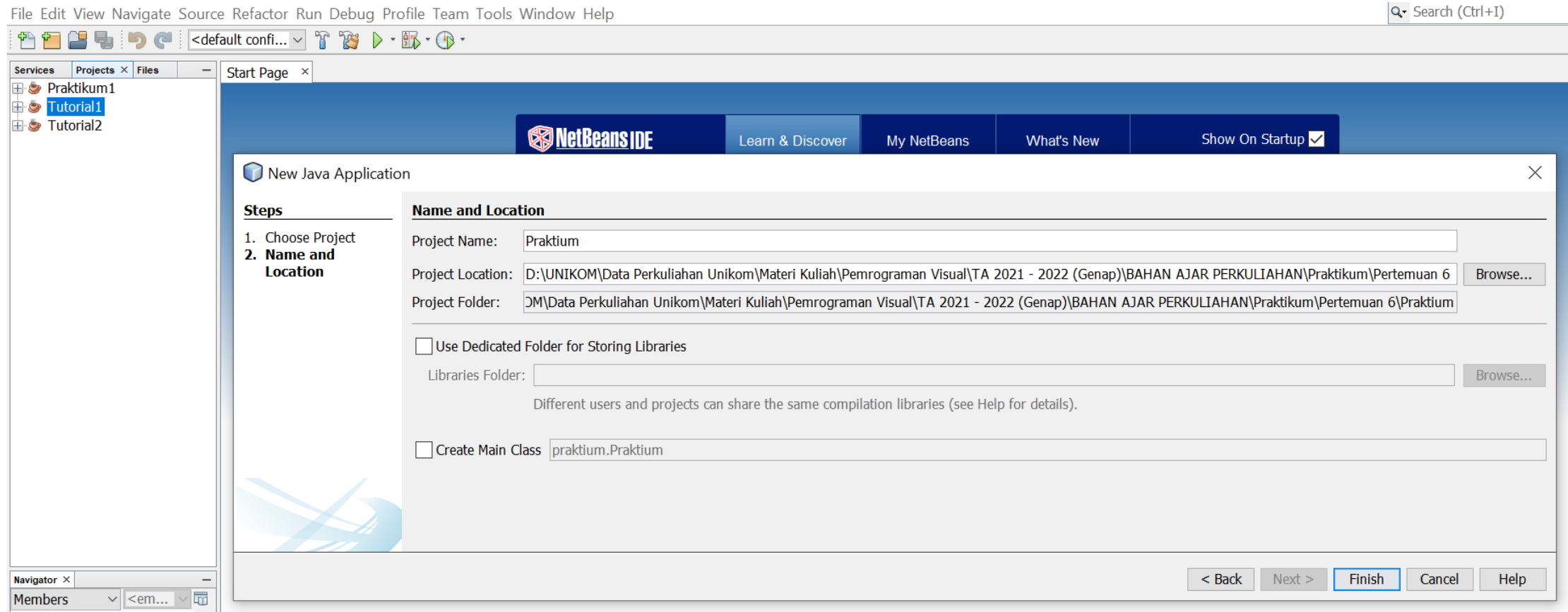
Setelah memahami konsep dasar yang kita butuhkan untuk menangani masalah operasi aritmatika dalam pembuatan program berbasis GUI. Selanjutnya kita coba untuk membuat program sederhana yang didalamnya terdapat operasi aritmatika

Kita akan membuat program kalkulator sederhana



Praktikum : Program Kalkulator

Langkah 1 : Buatlah project dan package baru pada netbeans. dengan cara : **File -> New Project-> Java Application -> Tulis nama Project -> unchecklist main class - > finish**



Praktikum : Program Kalkulator

Langkah 2 : Buatlah tampilan program seperti contoh dibawah ini (Silakan Anda berkreasi untuk menata letak komponen sesuai kebutuhan kasus) dan atur posisi program selalu ditengah.

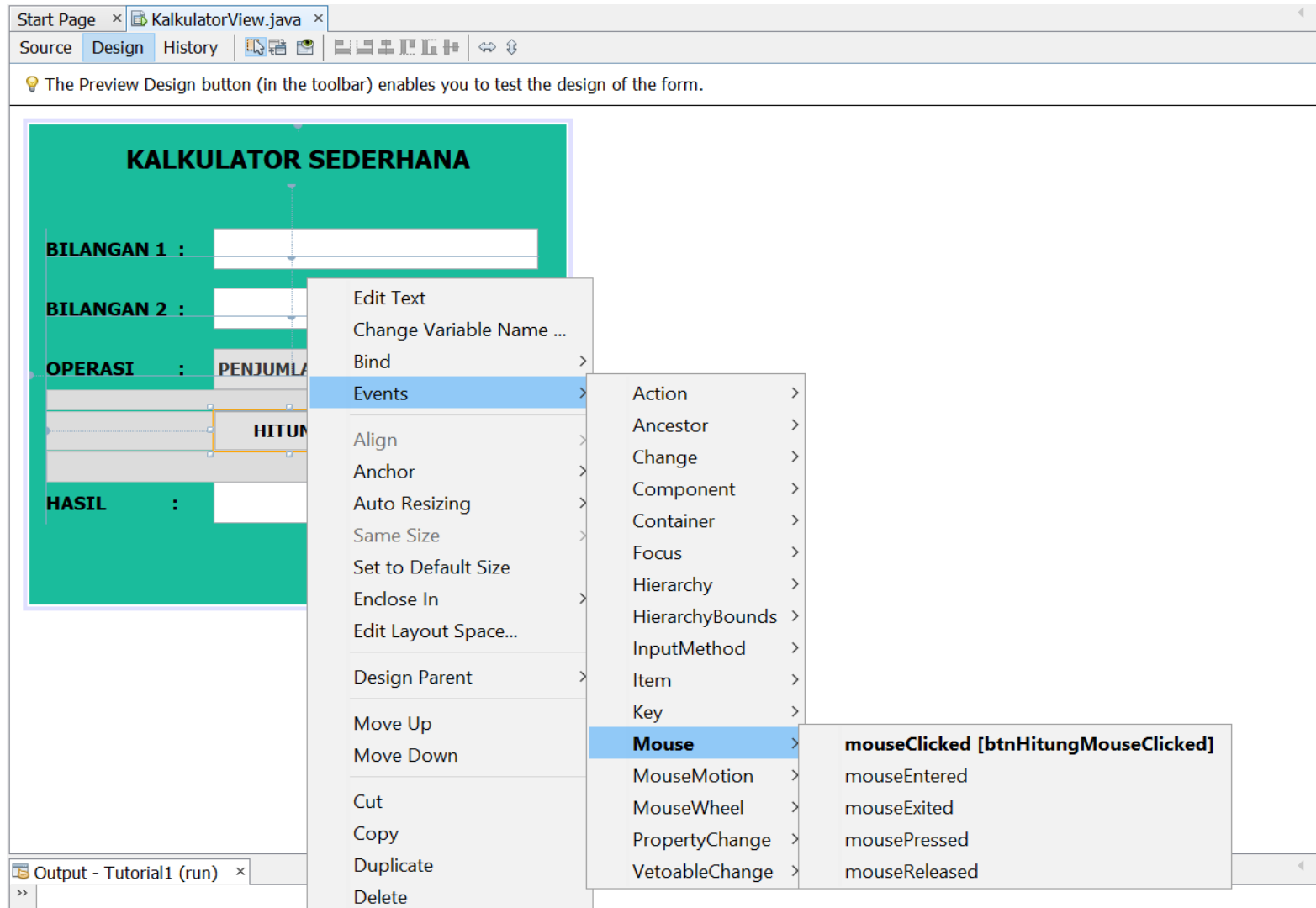
The image shows a Java Swing window titled "KALKULATOR SEDERHANA" with a teal background. It contains the following components:

- BILANGAN 1 :** A text label followed by a text input field labeled `{txtBilangan1}`.
- BILANGAN 2 :** A text label followed by a text input field labeled `{txtBilangan2}`.
- OPERASI :** A text label followed by a JComboBox labeled `{cmbOperasi}` with "PENJUMLAHAN" selected.
- Buttons:** Two buttons labeled `{btnHitung}` (HITUNG) and `{btnClear}` (CLEAR).
- HASIL :** A text label followed by a text output field labeled `{txtHasil}`.

- ❑ Tambahkan **2 buah component JTextField** untuk mengisi nilai bilangan 1 dan bilangan 2. Ubah nama variabelnya dengan `txtBilangan1` dan `txtBilangan2`
- ❑ Tambahkan **sebuah component JComboBox** untuk memilih jenis operator aritmatika. Tambahkan item pada combo box dengan **"penjumlahan", "Pengurangan", "perkalian", "pembagian"**. Ubah nama variabelnya dengan `cmbOperasi`
- ❑ Tambahkan **2 buah component JButton**. Ubah nama variabelnya dengan `btnHitung` dan `btnClear`
- ❑ Tambahkan **sebuah component Jtext** untuk menampilkan nilai hasil operasi aritmatika. Ubah nama variabelnya dengan `txtHasil`.

Praktikum : Program Kalkulator

Langkah 3 : Tambahkan event listener pada Button Hitung untuk menjalankan aksi operasi matematika dengan cara **Event -> Mouse -> mouseClicked**



Praktikum : Program Kalkulator

Langkah 4 : Masuk ke **mode Source**. Tambahkan kode program berikut yang digunakan untuk mengambil nilai yang diinputkan pada JTextField `bilangan1` dan `bilangan2`.

```
private void btnHitungMouseClicked(java.awt.event.MouseEvent evt) {  
  
    //Mengambil nilai Bilangan 1 & Bilangan 2 dari Text Field  
    double bilangan1 = Double.valueOf(txtBilangan1.getText());  
    double bilangan2 = Double.valueOf(txtBilangan2.getText());  
  
}
```

Praktikum : Program Kalkulator

Langkah 5 : Tambahkan kode program berikut yang digunakan untuk menjalankan operasi aritmatika yang dipilih oleh user pada JComboBox

```
//Kode Untuk Memilih Operasi Perhitungan dari ComboBox
double hasil;
int pilih = cmbOperasi.getSelectedIndex(); //Ambil Nilai Indeks ComboBox
switch(pilih) {
    case 0:
        hasil = bilangan1 + bilangan2 ;
        txtHasil.setText(String.valueOf(hasil));
        break;
    case 1:
        hasil = bilangan1 - bilangan2 ;
        txtHasil.setText(String.valueOf(hasil)); ;
        break;
    case 2:
        hasil = bilangan1 * bilangan2 ;
        txtHasil.setText(String.valueOf(hasil)); ;
        break;
    case 3:
        hasil = bilangan1 / bilangan2 ;
        txtHasil.setText(String.valueOf(hasil)); ;
        break;
}
```

Praktikum : Program Kalkulator

Langkah 6 : Tambahkan event listener pada Button Clear. Buatlah operasi untuk menset nilai pada setiap JTextField menjadi kosong (default). Untuk contoh kodenya sebagai berikut :

```
private void btnClearMouseClicked(java.awt.event.MouseEvent evt) {  
    //Set text field kedalam status Default  
    txtBilangan1.setText("");  
    txtBilangan2.setText("");  
    txtHasil.setText("");  
}
```

Praktikum : Program Kalkulator

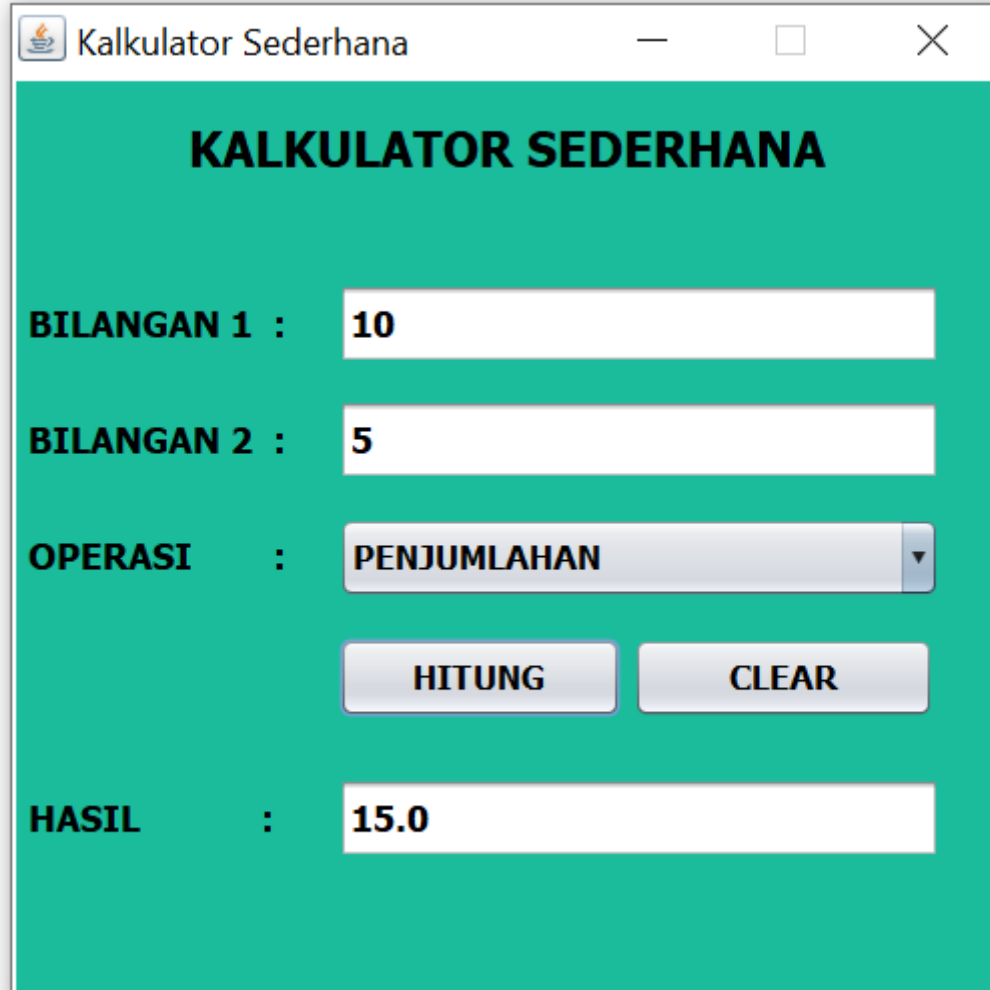
Langkah 7 : Tambahan, mungkin saja didalam pembuatan program berbasis GUI Anda ingin membuat program anda ukurannya selalu tetap sama (tidak bisa di maximize dst). Kita bisa mensettingnya didalam kode program dengan cara seperti ini

```
public KalkulatorView() {  
    initComponents();  
  
    setExtendedState(JFrame.MAXIMIZED_HORIZ);  
    setVisible(true);  
    setResizable(false);  
}
```

- ❑ Pada consturktor JFrame Form, tambahkan 3 buah kode berikut setelah initComponents()
- ❑ Fungsi dari ketiga kode tersebut untuk **mematikan fungsi maximize** pada tampilan program

Praktikum : Program Kalkulator

Langkah 8 : Jalankan program Anda , dan lihat hasilnya



Kalkulator Sederhana

KALKULATOR SEDERHANA

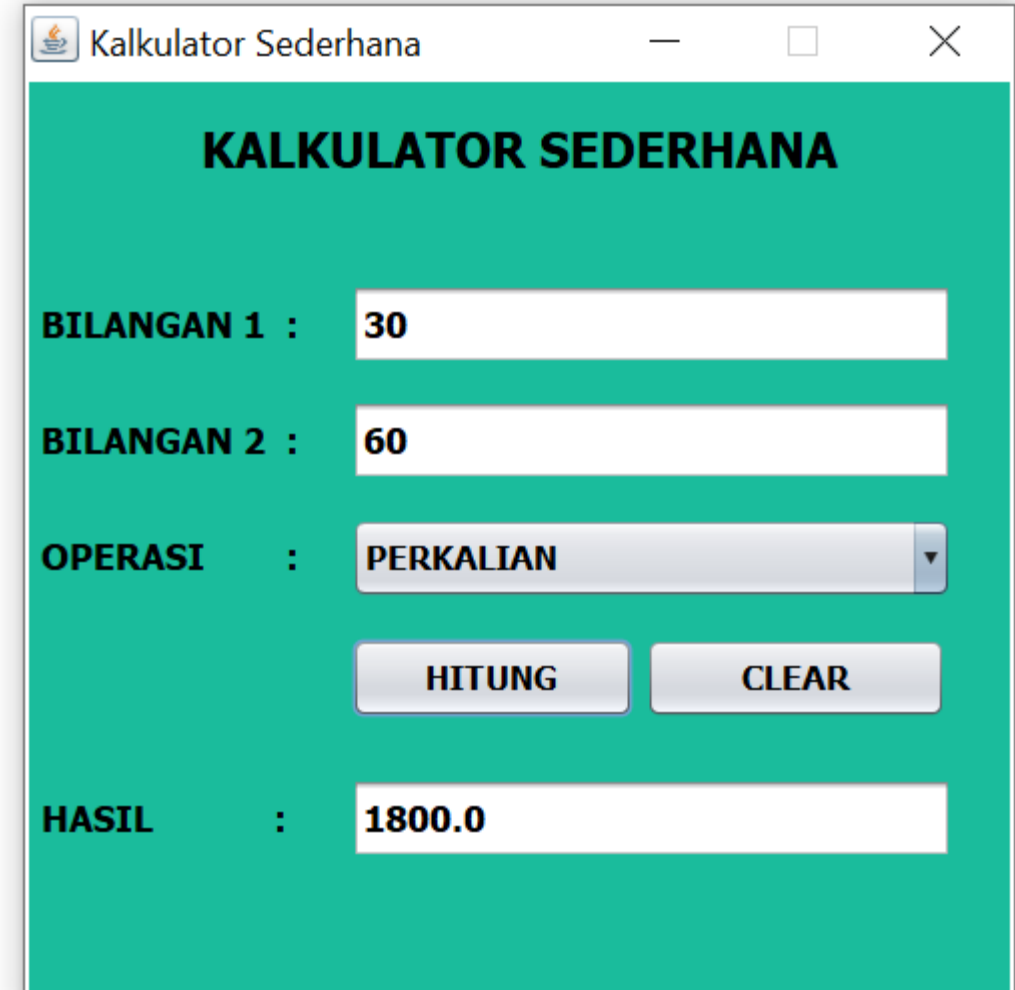
BILANGAN 1 : 10

BILANGAN 2 : 5

OPERASI : PENJUMLAHAN

HITUNG CLEAR

HASIL : 15.0



Kalkulator Sederhana

KALKULATOR SEDERHANA

BILANGAN 1 : 30

BILANGAN 2 : 60

OPERASI : PERKALIAN

HITUNG CLEAR

HASIL : 1800.0

Untuk Praktikum 3(B)

Cobalah kerjakan kembali praktikum tersebut namun cobalah rancang kode programnya dengan menggunakan konsep PBO (pisahkan class untuk logika program dengan tampilannya).

**Jadikan Program tersebut
sebagai Latihan Praktikum
Anda dipertemuan ini**



Terima Kasih