



# Modul Praktikum Pemrograman Berorientasi Objek

Jurusan Ilmu Komputer  
Fakultas Matematika dan Ilmu Pengetahuan Alam  
**Universitas Lampung**

**2022**  
Semester Ganjil

## Modul 15. JavaFX (Interactions)

JavaFX adalah library Java yang digunakan untuk membangun Rich Internet Applications. Aplikasi yang dikembangkan menggunakan JavaFX dapat berjalan di berbagai perangkat seperti Komputer Desktop, Mobile Phones, TV, Tablet, dll.

### Capaian Pembelajaran

1. Mahasiswa mampu membuat interaksi dari desain FXML
2. Mahasiswa mampu menghubungkan Program JavaFX dengan Database

### Materi

#### Event Source

Event source mengacu pada komponen JavaFX yang meng-generate event. Sebagai contoh, jika user menekan tombol, event source dalam hal ini adalah tombol.

#### Event Listener/Handler

The event listener menerima berita dari event-event dan proses-proses interaksi user. Ketika tombol ditekan, listener akan mengendalikan dengan menampilkan sebuah informasi yang berguna untuk user.

#### Event Object

Ketika sebuah event terjadi (misal, ketika user berinteraksi dengan komponen Control JavaFX), sebuah object event diciptakan. Object berisi semua informasi yang perlu tentang event yang telah terjadi. Informasi meliputi tipe dari event yang telah terjadi, seperti ketika mouse telah di-klik. Ada beberapa class event untuk kategori yang berbeda dari user action. Sebuah event object mempunyai tipe data mengenai salah satu class ini.

## Langkah Praktikum

### Membuat Project JavaFX

1. Buka NetBeans
2. Buat project JavaFX baru dengan nama **prak15\_nama**
3. Hapus file **FXMLDocument.fxml** dan juga **FXMLDocumentController.java**
4. Buat beberapa file fxml baru dengan nama **Output.fxml**, dan **FormInput.fxml**
5. Pada saat membuat file, centang pada kolom create Controller

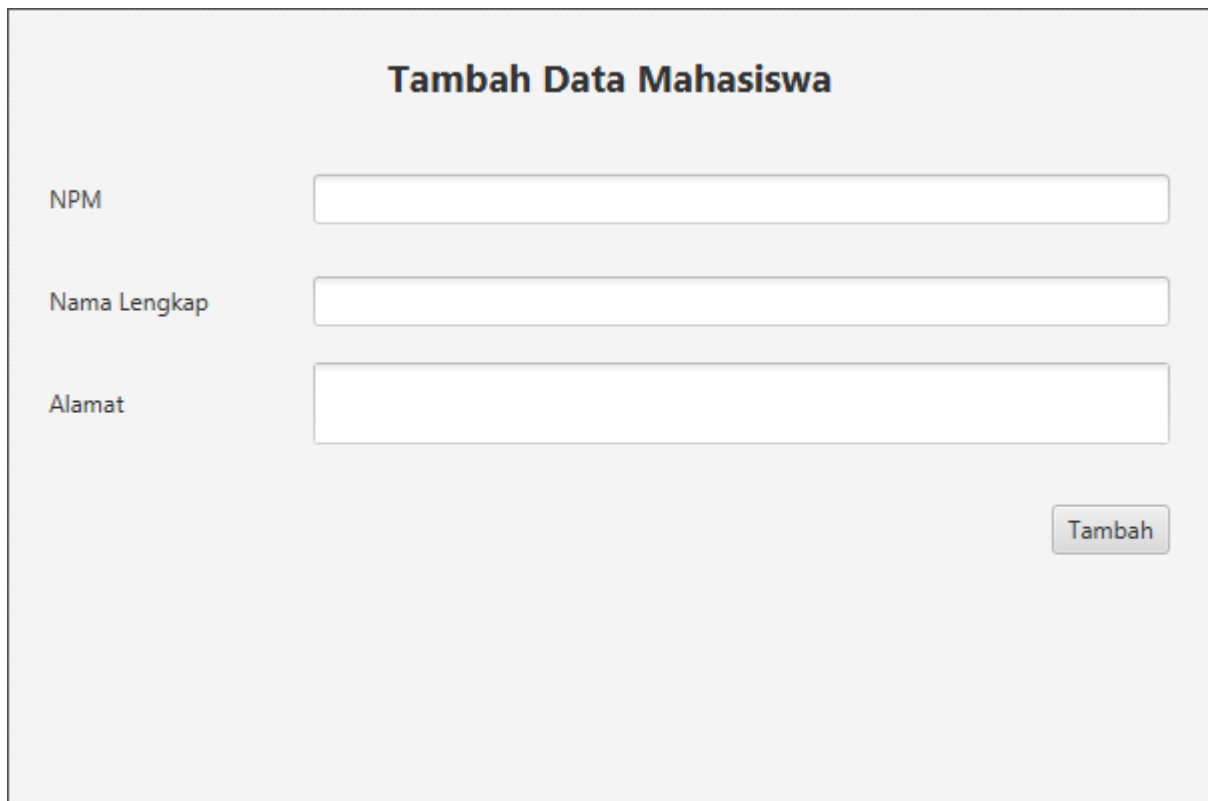
### Membuat Desain Aplikasi

1. Buka file fxml yang telah dibuat menggunakan SceneBuilder
2. Kemudian buatlah tampilan Output menjadi seperti berikut :

Data Mahasiswa	
NPM	NPM
Nama Lengkap	Nama
Alamat	Alamat

Data Baru

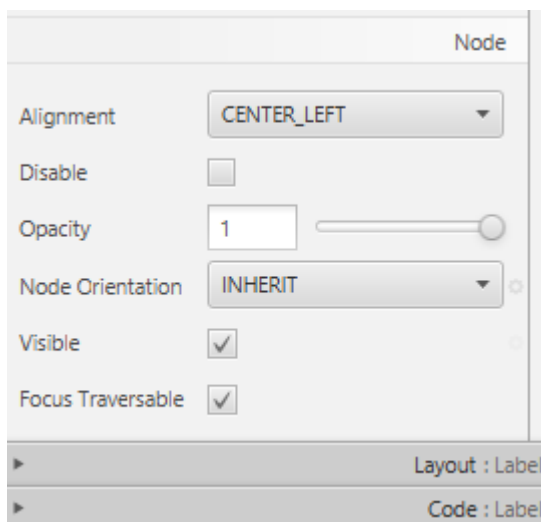
3. Kemudian buat juga tampilan **FormInput** menjadi seperti berikut :



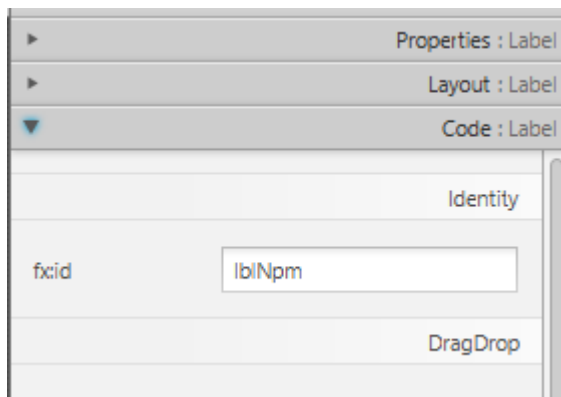
The screenshot shows a web form titled "Tambah Data Mahasiswa". It contains three input fields stacked vertically, each with a label to its left: "NPM", "Nama Lengkap", and "Alamat". To the right of the "Alamat" field is a button labeled "Tambah". The entire form is enclosed in a light gray border.

### Menambahkan ID Element

1. Klik pada salah satu element yang ingin diberi ID
2. Pada sisi bawah kanan SceneBuilder bukalah panel **Code**



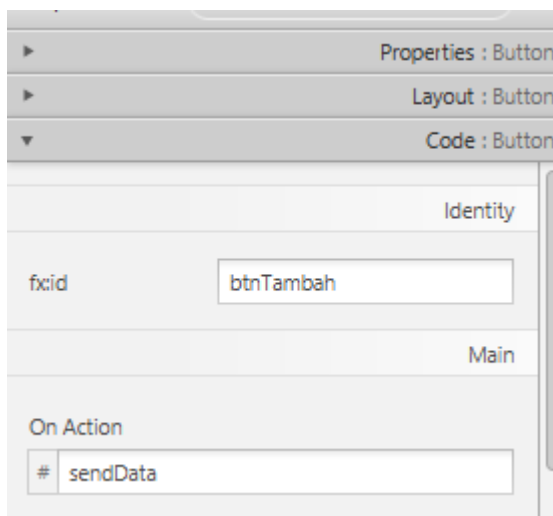
3. Kemudian tambahkan ID element tersebut pada field **fx:id**



4. Lakukan hal tersebut kepada semua elemen yang nantinya akan diberi interaksi, dan sesuaikan namanya

### Menambahkan method onAction

1. Pada button di masing-masing fxml tambahkan sebuah nama Method pada field **On Action**.
2. Pada FormInput isi field **On Action** dengan **sendData**



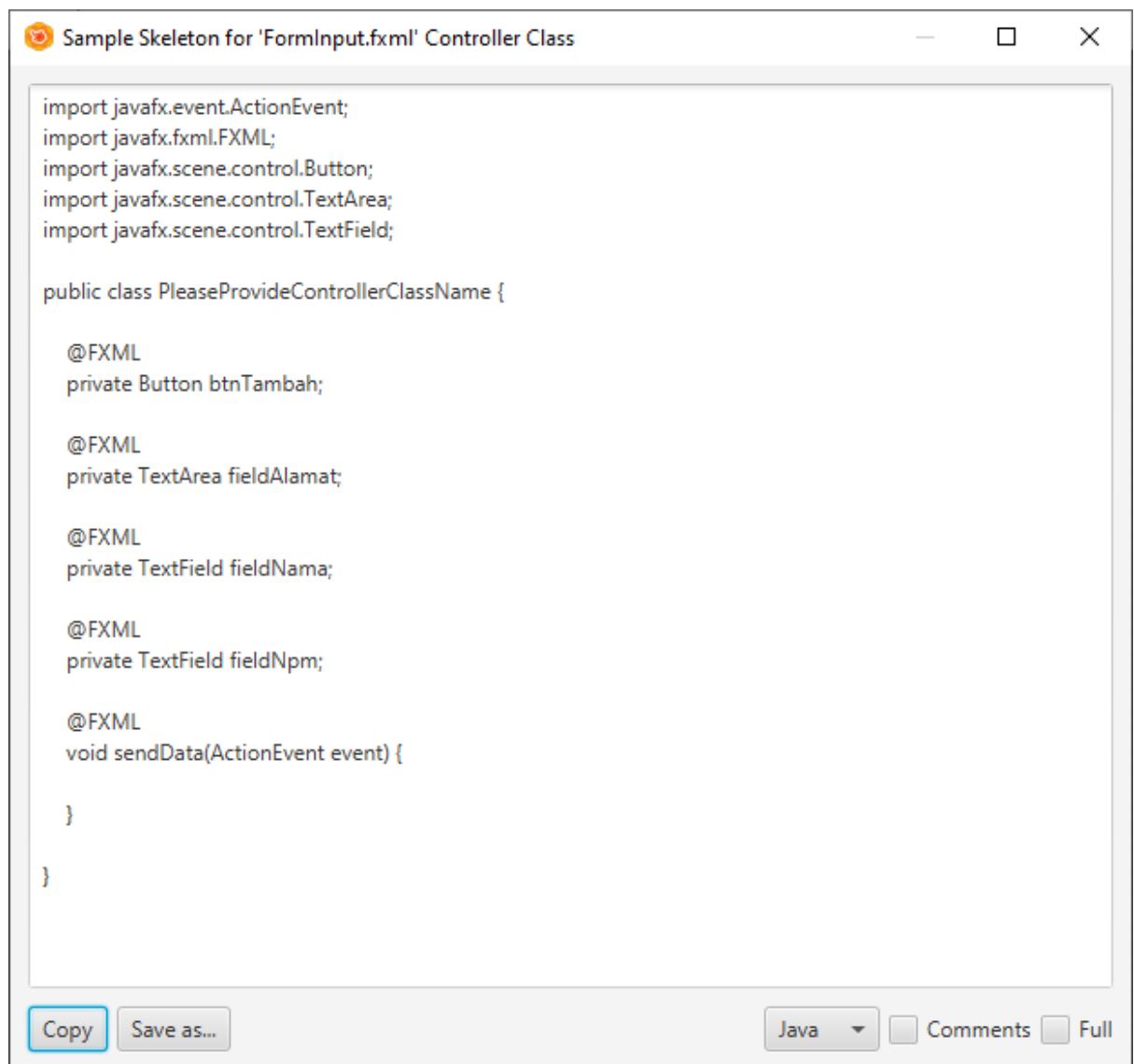
3. Pada Output isi field **On Action** dengan **openForm**

### Membuat Controller

1. Pada setiap file fxml yang dibuka di SceneBuilder bukalah tab **view > show Sample Controller Skeleton**

View	Insert	Modify	Arrange	Prev
Content			Ctrl+0	
Properties			Ctrl+1	
Layout			Ctrl+2	
Code			Ctrl+3	
Hide Library			Ctrl+4	
Hide Document			Ctrl+5	
Show CSS Analyzer			Ctrl+6	
Hide Left Panel			Ctrl+7	
Hide Right Panel			Ctrl+8	
Show Outlines			Ctrl+E	
Show Sample Data				
Disable Alignment Guides				
Zoom				
Show Sample Controller Skeleton				

Maka akan muncul tampilan seperti berikut :



2. Salin isi dari classnya kemudian masukkan kedalam Controller yang telah dibuat sesuai dengan file fxmInya.

3. Isi dari file controller FormInputController akan menjadi seperti berikut :

```
public class FormInputController {  
  
    @FXML  
    private Button btnCancel;  
  
    @FXML  
    private Button btnTambah;  
  
    @FXML  
    private TextArea fieldAlamat;  
  
    @FXML  
    private TextField fieldNama;  
  
    @FXML  
    private TextField fieldNpm;  
  
    @FXML  
    void sendData(ActionEvent event) throws IOException {  
  
    }  
  
}
```

4. Isi dari file controller OutputController akan menjadi seperti berikut :

```
public class OutputController {  
  
    @FXML  
    private Button btnNew;  
  
    @FXML  
    private Label lblAlamat;  
  
    @FXML  
    private Label lblNama;  
  
    @FXML  
    private Label lblNpm;  
  
    @FXML  
    void openForm(ActionEvent event) throws IOException {  
  
    }  
  
}
```



## Membuat Interaksi Button

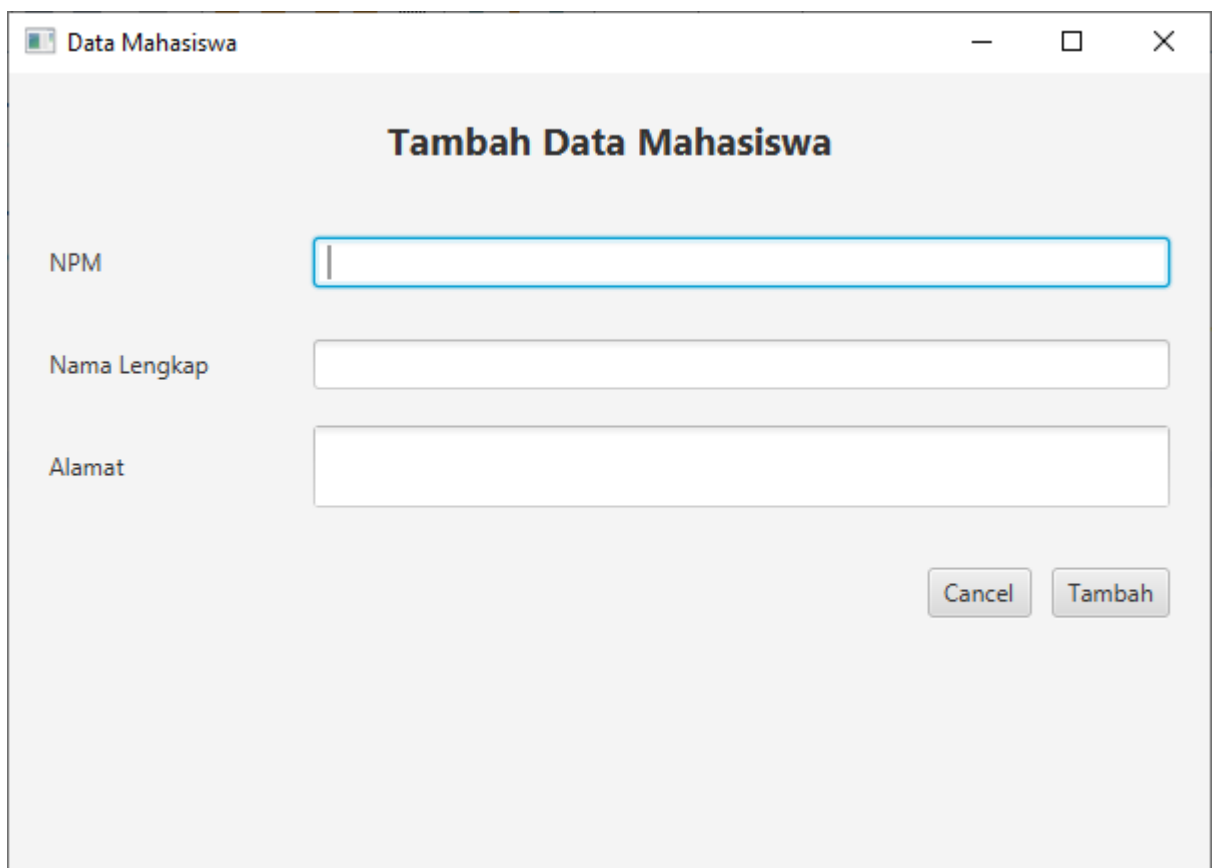
1. Buka file utama, kemudian ubah isi dari method start agar dapat menampilkan FormInput ketika pertama kali dijalankan.

```
@Override
public void start(Stage stage) throws Exception {
    Parent root = FXMLLoader.load(getClass().getResource("FormInput.fxml"));

    Scene scene = new Scene(root);

    stage.setTitle("Data Mahasiswa");
    stage.setScene(scene);
    stage.show();
}
```

2. Jika anda menjalankan program, maka tampilannya akan terlihat seperti berikut :



The screenshot shows a JavaFX application window titled "Data Mahasiswa". Inside the window, there is a form titled "Tambah Data Mahasiswa". The form contains three input fields: "NPM", "Nama Lengkap", and "Alamat". Below the input fields, there are two buttons: "Cancel" and "Tambah".

3. Buka file **FormInputController** isi method **sendData** sehingga ketika tombol tambah di klik maka akan berpindah scene menjadi **Output**

```
@FXML
void sendData(ActionEvent event) throws IOException {

    FXMLLoader loader = new FXMLLoader(getClass().getResource("Output.fxml"));
    Parent root = loader.load();

    Stage stage = (Stage) btnTambah.getScene().getWindow();
    stage.setScene(new Scene(root));
}
```

Jika anda menjalankan program dan menekan tombol “Tambah” scene akan berganti menjadi tampilan Output.

4. Buka file OutputController, kemudian ubah method **openForm** agar ketika tombol ditekan maka scene akan berganti menjadi FormInput

```
void openForm(ActionEvent event) throws IOException {
    Parent root = FXMLLoader.load(getClass().getResource("FormInput.fxml"));

    Stage stage = (Stage) btnNew.getScene().getWindow();
    stage.setScene(new Scene(root));
}
```

5. Jika anda menjalankan program sekarang anda dapat berpindah-pindah dari tampilan FormInput ke Output dan sebaliknya

## Mengirim Data Antar Scene

1. Sekarang buatlah agar ketika tombol tambah pada FormInput ditekan, data yang ada pada form dapat ditampilkan pada Output.
2. Pertama-tama buatlah sebuah class **Mahasiswa** dengan atribut NPM, Nama, dan Alamat (Semuanya String) beserta Constructor, dan Setter Getternya.
3. Selanjutnya buka controller **OutputController** kemudian buat sebuah method void dengan nama **showValue** dengan satu parameter Mahasiswa.

4. Isi method **showValue** dengan melakukan setText terhadap semua label tempat menampilkan data

```
public void showValue(Mahasiswa mhs) {
    lblNpm.setText(mhs.getNpm());
    lblNama.setText(mhs.getNama());
    lblAlamat.setText(mhs.getAlamat());
}
```

5. Kemudian buka controller **FormInputController**, kemudian pada method **sendData** buat objek Mahasiswa yang diisi dengan nilai dari textField yang ada pada FXMLnya
6. Buat objek dari controller OutputController menggunakan objek FXMLLoader yang telah dibuat.
7. Panggil method **showValue** menggunakan objek OutputController dan kirimkan Objek Mahasiswa yang telah dibuat melalui argumennya.

```
void sendData(ActionEvent event) throws IOException {
    Mahasiswa mhs = new Mahasiswa(fieldNpm.getText(), fieldNama.getText(), fieldAlamat.getText());
    FXMLLoader loader = new FXMLLoader(getClass().getResource("Output.fxml"));
    Parent root = loader.load();
    OutputController outputController = loader.getController();
    outputController.showValue(mhs);
    Stage stage = (Stage) btnTambah.getScene().getWindow();
    stage.setScene(new Scene(root));
}
```

8. Sekarang jika anda menjalankan program, maka data yang diinput pada Form akan terkirim dan ditampilkan pada Scene Output

## Tugas

1. Screenshot hasil running program
2. Kirimkan hasilnya ke Vclass