

VERSI 0.1
JANUARI 2025



PEMROGRAMAN LANJUT

MODUL 6 - GRAPHICAL USER INTERFACE

DISUSUN OLEH:

Ir. Wildan Suharso, M.Kom.
Muhammad Ega Faiz Fadlillah
M. Ramadhan Titan Dwi .C

TIM LABORATORIUM INFORMATIKA
UNIVERSITAS MUHAMMADIYAH MALANG

PENDAHULUAN

TUJUAN

1. Mahasiswa mampu memahami GUI JavaFx
2. Mahasiswa mampu membuat GUI dengan Java Swing

TARGET MODUL

1. Mahasiswa mampu memahami GUI JavaFx
2. Mahasiswa mampu membuat GUI dengan Java Swing

PERSIAPAN

1. Java Development Kit
2. Text Editor / IDE (**Preferably** IntelliJ IDEA, Visual Studio Code, Netbeans, etc).
3. **Source Code Program** [JavaSwing](#)

KEYWORDS

GUI, JavaFx, Swing

TABLE OF CONTENTS

PENDAHULUAN.....	2
TUJUAN.....	2
TARGET MODUL.....	2
PERSIAPAN.....	2
KEYWORDS.....	2
TABLE OF CONTENTS.....	2
TEORI.....	4
KONSEP GU.....	4
A. Jenis Pustaka GUI.....	4
1. AWT.....	4
2. Swing.....	5
3. JavaFX.....	5
B. JavaFx Tutorial.....	6
1. Istilah-Istilah dalam JavaFX.....	6
2. JavaFX Layouts.....	7
3. JavaFX 2D Shapes.....	9
4. JavaFX Teks.....	9





5. JavaFX UI.....	12
C. Swing Tutorial.....	16
1. Struktur Library Java Swing.....	16
2. Istilah-Istilah dalam Java Swing.....	17
3. Java JFrame.....	18
4. Java JButton.....	19
5. Java JLabel.....	21
7. Java JTextArea.....	25
8. Java JTable.....	27
9. Java Swing Layout Manager.....	29
10. Contoh Java Swing Lainnya.....	34
MATERI PRAKTIKUM.....	36
A. INSTALASI DAN KONFIGURASI JAVA FX.....	36
1. Download JavaFX SDK.....	36
2. Membuat JavaFX Project.....	37
3. Tambahkan SDK Library.....	38
B. INSTALASI DAN KONFIGURASI JAVA SWING.....	39
C. SCENE BUILDER.....	41
D. CARA PENGGUNAAN SCENE BUILDER.....	41
1. Cara 1 : Menggunakan Scene Builder Kit pad IDE.....	41
2. Cara 2 : Menggunakan Aplikasi SceneBuilder.exe.....	42
REFRENSI.....	45
CODELAB.....	46
CODELAB 1.....	46
CODELAB 2.....	49
TUGAS.....	52
KRITERIA & DETAIL PENILAIAN.....	54



TEORI

KONSEP GUI

Graphical User Interface (GUI) adalah antarmuka pengguna berbasis grafis yang memungkinkan interaksi dengan perangkat lunak melalui elemen visual seperti tombol, kotak teks, menu, dan jendela, sehingga pengguna tidak perlu mengetik perintah rumit. GUI menyembunyikan kompleksitas instruksi di balik tampilan visual yang intuitif, memungkinkan interaksi dengan perangkat input seperti mouse atau layar sentuh. Dalam pengembangan perangkat lunak, GUI dibuat menggunakan bahasa pemrograman serta library atau framework khusus, lalu dihubungkan dengan logika program agar setiap aksi pengguna menghasilkan respon yang sesuai. Dengan kelebihan seperti kemudahan penggunaan, interaktivitas, dan peningkatan pengalaman pengguna, GUI menjadi jembatan utama antara manusia dan komputer.

A. Jenis Pustaka GUI

Ada beberapa pustaka GUI yang bisa dimanfaatkan, antara lain AWT, Swing, dan JavaFX. Setiap pustaka memiliki karakteristik serta keunggulan masing-masing, baik dari segi tampilan, fleksibilitas, maupun fitur yang ditawarkan.

1. AWT

AWT (Abstract Window Toolkit) adalah toolkit GUI (Graphical User Interface) bawaan Java yang diperkenalkan sejak Java 1.0. Toolkit ini menyediakan kelas dan API untuk membuat komponen grafis seperti jendela, tombol, label, dan text field. AWT bersifat platform-dependent karena setiap komponennya bergantung pada sistem operasi yang digunakan, sehingga tampilannya bisa berbeda antar OS. Selain itu, AWT disebut heavyweight karena setiap komponennya berhubungan langsung dengan komponen asli dari sistem operasi.

a. Kelebihan

- API sederhana sehingga mudah dipelajari bagi pemula.
- Penampilan natif pada platform
- Kinerja relatif baik karena langsung menggunakan komponen dari sistem operasi.

b. Kekurangan

- Fitur terbatas dan tidak mendukung tampilan modern.
- Penampilan tidak konsisten di seluruh platform.
- Tidak fleksibel dalam mendukung komponen kustom yang lebih kompleks.



2. Swing

Java Swing adalah bagian dari Java Foundation Classes (JFC) yang dikembangkan sebagai penyempurnaan dari AWT. Swing sepenuhnya ditulis dalam Java sehingga bersifat platform-independent dan menggunakan komponen lightweight, membuat tampilannya konsisten di berbagai sistem operasi. Berbeda dari AWT yang bergantung pada komponen native, Swing menawarkan lebih banyak komponen GUI yang fleksibel, dapat dikustomisasi, dan mendukung pembuatan antarmuka pengguna yang lebih kaya fitur.

a. Kelebihan

- Penampilan konsisten di seluruh platform karena tidak bergantung pada GUI native.
- Menyediakan banyak komponen GUI yang fleksibel dan bervariasi.
- Mendukung kustomisasi tampilan dan perilaku komponen sesuai kebutuhan.

b. Kekurangan

- Performa bisa lebih lambat dibanding AWT karena tidak menggunakan komponen native.
- Beberapa komponen terasa kurang responsif pada aplikasi kompleks.
- Membutuhkan lebih banyak sumber daya sistem saat membangun GUI yang besar.

Karena dibangun di atas fondasi AWT, Swing tetap **memanfaatkan** mekanisme dasar seperti **event handling** dan **layout manager milik AWT**. Itulah sebabnya penggunaan layout manager dari AWT masih sangat umum dalam pengembangan aplikasi berbasis Swing, karena AWT menyediakan struktur tata letak yang stabil dan kompatibel.

3. JavaFX

JavaFX adalah pustaka modern dalam Java yang digunakan untuk membangun aplikasi dengan antarmuka pengguna (GUI) yang kaya, interaktif, dan konsisten di berbagai platform. Diperkenalkan oleh Sun Microsystems (kemudian diakuisisi Oracle), JavaFX hadir sebagai penerus Swing dengan fitur lebih canggih seperti dukungan efek visual, animasi, grafik 2D/3D, tata letak yang fleksibel, integrasi dengan CSS, FXML untuk desain antarmuka, serta dukungan multimedia. JavaFX dirancang agar aplikasi dapat berjalan di berbagai perangkat, mulai dari desktop, mobile, tablet, TV, hingga web, sehingga menjadi solusi lebih modern dibanding AWT dan Swing.

a. Kelebihan

- Mendukung tampilan modern dengan efek visual, animasi, grafik 2D/3D, serta integrasi CSS.



- Fleksibel dengan tata letak, FXML, kontrol canggih, dan dukungan multimedia.
- Bersifat lintas platform sehingga dapat dijalankan di desktop, mobile, maupun web.

b. Kekurangan

- Kurang populer dibandingkan Swing atau framework UI lain sehingga komunitasnya lebih kecil.
- Lebih sulit dipelajari, terutama bagi pemula, dan pengembangan bisa lebih lama.

B. JavaFx Tutorial

1. Istilah-Istilah dalam JavaFX

Sebelum mulai mempelajari JavaFX, ada baiknya jika kalian terlebih dahulu memahami beberapa istilah berikut :

- a. **Scene** : Scene dapat diibaratkan sebagai "halaman" dalam aplikasi, tempat kita meletakkan tombol, teks, gambar, maupun elemen lain yang bisa dilihat dan digunakan oleh pengguna.
- b. **Stage** : Stage adalah jendela utama aplikasi JavaFX. Bisa dibayangkan seperti bingkai besar tempat Scene ditampilkan. Setiap aplikasi JavaFX memiliki setidaknya satu Stage.
- c. **Node** : Node merupakan dasar dari semua elemen tampilan di JavaFX. Semua komponen visual seperti tombol, teks, gambar, hingga layout termasuk dalam Node.
- d. **Control** : Control adalah Node khusus yang digunakan untuk interaksi pengguna, seperti tombol, kotak teks, checkbox, dan elemen interaktif lainnya.
- e. **Dialog** : Dialog adalah jendela kecil yang muncul untuk memberi informasi atau meminta input dari pengguna, misalnya kotak peringatan atau kotak konfirmasi.
- f. **EventHandler** : EventHandler adalah mekanisme yang menangani aksi pengguna, seperti saat tombol diklik atau teks dimasukkan.
- g. **CSS Styling** : JavaFX mendukung penggunaan CSS (Cascading Style Sheets) untuk mengatur tampilan. Dengan CSS, kita bisa mengubah warna, ukuran, font, maupun gaya komponen, mirip seperti mendesain tampilan website.

2. JavaFX Layouts

JavaFX Layouts adalah sekumpulan kelas wadah (container) yang digunakan untuk mengatur dan menentukan gaya antarmuka pengguna (UI) pada objek-objek dalam scene graph. Layout berperan sebagai node induk yang mengatur bagaimana elemen-elemen ditampilkan di layar, seperti melalui HBox, VBox, StackPane, FlowPane, AnchorPane, dan lain-lain. Semua kelas tata letak ini berada dalam paket **javafx.scene.layout**, dengan Pane sebagai kelas dasar yang menjadi akar bagi seluruh layout bawaan JavaFX.

Berikut adalah Beberapa Jenis Layout Classes :

Layout	Deskripsi
Borderpane	Mengatur tata letak node pada lima posisi utama atas (top), kiri (left), kanan (right), bawah (bottom), dan tengah (center).
Gridpane	Mengatur tata letak node dalam bentuk grid atau tabel yang fleksibel dengan baris (rows) dan kolom (columns).
FlowPane	Menempatkan node secara berurutan (flow) baik secara horizontal maupun vertikal, dan secara otomatis membungkus (wrap) elemen ke baris atau kolom berikutnya ketika mencapai batas ukuran panel
StackPane	Menempatkan node di atas satu sama lain dalam bentuk tumpukan (stack), dengan elemen yang ditambahkan terakhir berada paling depan.
Pane	Kelas dasar untuk semua kelas tata letak
HBox	Menyusun node dalam satu baris horizontal (sejajar ke samping).
VBox	VBox menyusun node dalam satu kolom vertikal (dari atas ke bawah).

Contoh : Layout Stackpane

Berikut adalah contoh GUI yang menggunakan layout StackPane (tanpa menggunakan file FXML). Dalam contoh ini, StackPane dipakai sebagai tata letak utama untuk menampilkan beberapa elemen yang disusun secara bertumpuk. Seperti terlihat pada program, terdapat tiga buah label dengan warna berbeda (merah, hijau, dan biru) yang saling menumpuk satu sama lain.

```
public class StackPaneExample extends Application {
    @Override
    public void start(Stage primaryStage) {
        Label label1 = new Label("LABEL 1");
        label1.setStyle("-fx-background-color: blue; -fx-text-fill: white; -fx-font-size: 18px;");
        label1.setAlignment(Pos.TOP_CENTER);

        Label label2 = new Label("LABEL 2");
        label2.setStyle("-fx-background-color: green; -fx-text-fill: white; -fx-font-size: 18px;");
        label2.setTranslateX(50);
        label2.setTranslateY(50);
        label2.setAlignment(Pos.TOP_CENTER);

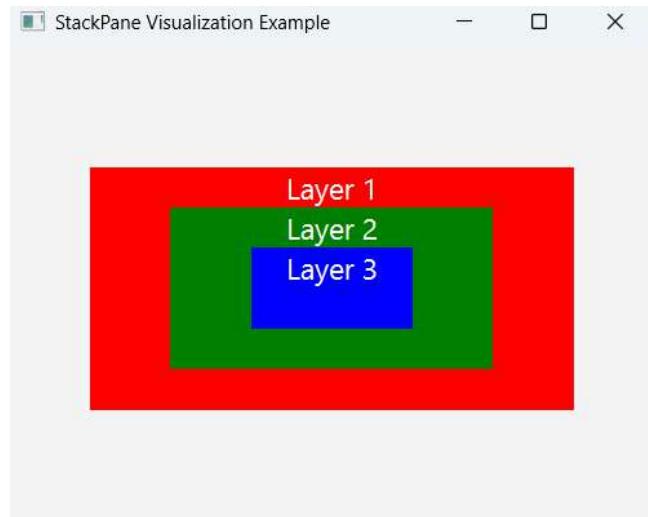
        Label label3 = new Label("LABEL 3");
        label3.setStyle("-fx-background-color: red; -fx-text-fill: white; -fx-font-size: 18px;");
        label3.setTranslateX(-50);
        label3.setTranslateY(-50);
        label3.setAlignment(Pos.TOP_CENTER);

        StackPane root = new StackPane();
        root.getChildren().addAll(label1, label2, label3);

        Scene scene = new Scene(root, 300, 300);

        primaryStage.setTitle("StackPane Visualization Example");
        primaryStage.setScene(scene);
        primaryStage.show();
    }

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



3. JavaFX 2D Shapes

Setiap bentuk 2D di JavaFX direpresentasikan oleh kelas-kelas khusus dalam paket **javafx.scene.shape**, seperti Rectangle, Circle, Ellipse, Line, Polygon, Polyline, dan Arc. Untuk membuatnya ikuti langkah-langkah berikut :

- a. Membuat objek dari kelas shape yang sesuai, misalnya **Circle circle = new Circle();**
- b. Mengatur properti-properti yang diperlukan (seperti posisi, ukuran, warna, dll.) dengan menggunakan method **setter**.

```
Circle circle = new Circle();
circle.setRadius(80);
circle.setFill(Color.CORNFLOWERBLUE);
circle.setStroke(Color.DARKBLUE);
circle.setStrokeWidth(4);
```

- c. Menambahkan objek shape tersebut ke layout (misalnya Group, Pane, atau StackPane) agar dapat ditampilkan di dalam scene.

```
StackPane root = new StackPane(circle);
```

4. JavaFX Teks

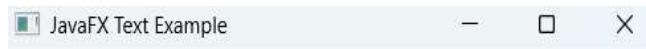
Kadang-kadang, sebuah aplikasi membutuhkan tampilan informasi dalam bentuk teks pada antarmuka. Untuk itu, JavaFX menyediakan kelas **javafx.scene.text**. yang dapat digunakan. Melalui kelas ini, kita bisa mengatur berbagai properti teks sesuai kebutuhan.

- a. Membuat Teks

Untuk menampilkan tulisan di JavaFX, kita bisa membuat objek dari kelas Text terlebih dahulu. Setelah itu, isi teksnya bisa diatur dengan memanggil metode **setText(string)** seperti contoh dibawah :



```
● ● ●  
public class TextExample extends Application {  
    @Override  
    public void start(Stage primaryStage) {  
        Text myText = new Text();  
        myText.setText("Hello, Welcome to JavaFX!");  
  
        StackPane root = new StackPane();  
        root.getChildren().add(myText);  
  
        Scene scene = new Scene(root, 400, 200);  
  
        primaryStage.setTitle("JavaFX Text Example");  
        primaryStage.setScene(scene);  
        primaryStage.show();  
    }  
}
```



Hello, Welcome to JavaFX!

b. Mengatur Font dan Posisi Teks

Untuk mengatur tampilan teks pada kelas Text, kita bisa menggunakan metode **setFont()** yang menerima objek dari kelas Font. Objek font ini dapat dibuat dengan memanggil metode **Font.font()**, di mana kita bisa menentukan parameter seperti family (jenis huruf), weight (ketebalan huruf), posture (gaya huruf), dan size (ukuran huruf). Selain itu, warna teks juga diubah menggunakan metode **setFill()**.





```
public class TextExample extends Application {  
    @Override  
    public void start(Stage primaryStage) {  
        myText.setText("Hello, Welcome to JavaFX!");  
  
        myText.setFont(Font.font("Arial", FontWeight.BOLD, 20));  
        myText.setFill(Color.BLUE);  
  
        StackPane root = new StackPane();  
        root.getChildren().add(myText);  
  
        Scene scene = new Scene(root, 400, 200);  
  
        primaryStage.setTitle("JavaFX Text Example");  
        primaryStage.setScene(scene);  
        primaryStage.show();  
    }  
}
```

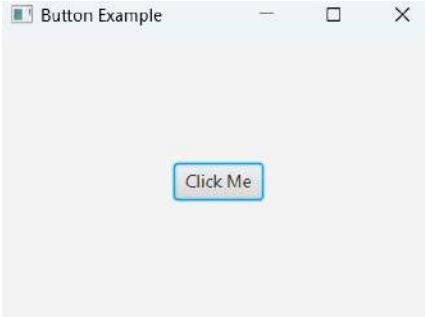


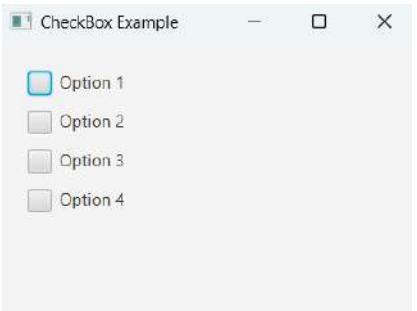
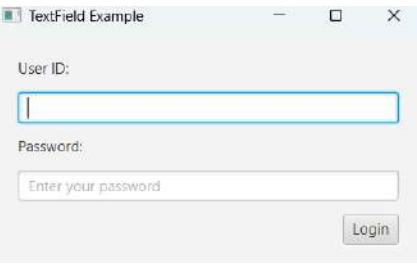
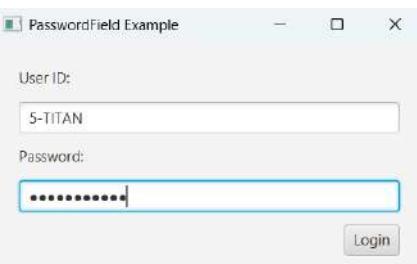
Hello, Welcome to JavaFX!

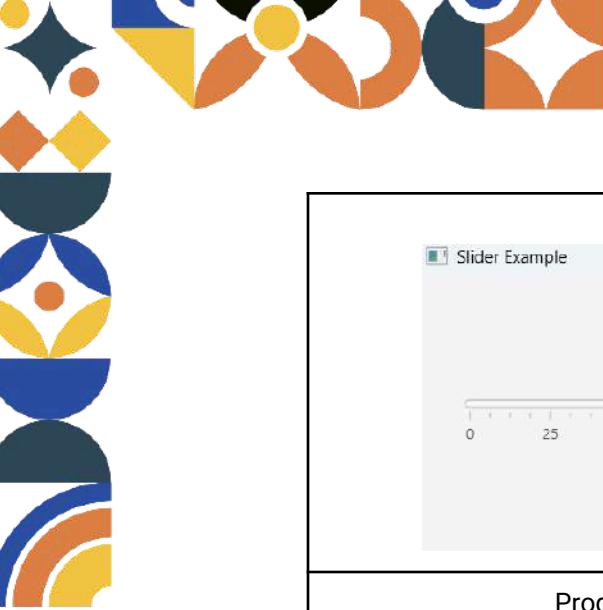


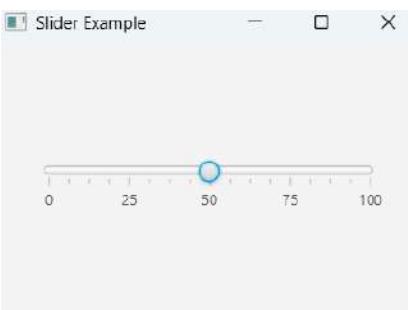
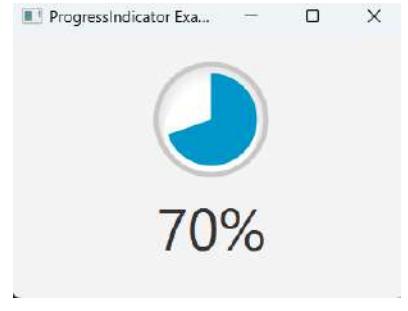
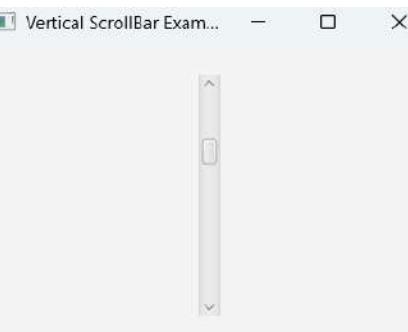
5. JavaFX UI

Elemen UI merupakan komponen yang ditampilkan kepada pengguna untuk melakukan interaksi. Dalam JavaFX, paket **javafx.scene.control** menyediakan berbagai kelas yang dibutuhkan untuk membuat komponen UI seperti Button, Label, Slider, Progress Bar dan lainnya. Berikut ini adalah daftar kontrol UI yang tersedia di JavaFX.

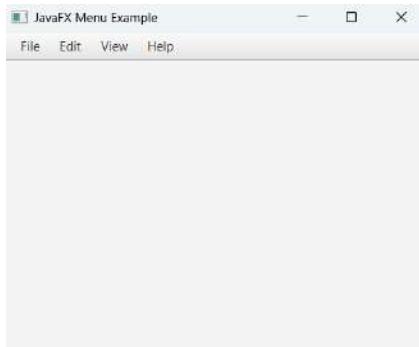
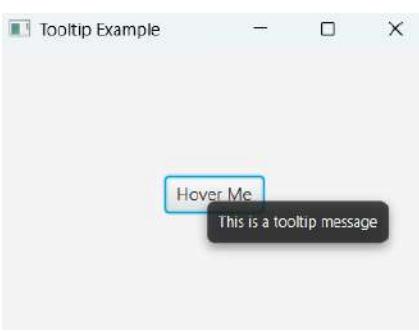
Komponen UI	Deskripsi
Label 	Komponen untuk menampilkan teks sederhana di layar tanpa bisa diedit oleh pengguna.
Button 	Tombol yang bisa diklik untuk menjalankan fungsi tertentu dalam aplikasi.
RadioButton 	Memberikan beberapa pilihan kepada pengguna, tapi hanya satu opsi yang bisa dipilih dalam satu grup.
Checkbox	Kotak centang yang memungkinkan

	<p>pengguna memilih satu atau lebih opsi, menandai centang untuk “benar” dan tidak dicentang untuk “salah”.</p>
<h3>TextField</h3> 	<p>Kotak teks untuk menerima input teks dari pengguna</p>
<h3>PasswordField</h3> 	<p>Kotak teks khusus untuk kata sandi, di mana teks yang diketik disembunyikan dari layar.</p>
<h3>Hyperlink</h3> 	<p>Teks yang bisa diklik untuk membuka halaman web atau alamat URL tertentu.</p>
<h3>Slider</h3>	<p>Komponen grafis untuk memilih nilai dari rentang tertentu dengan</p>



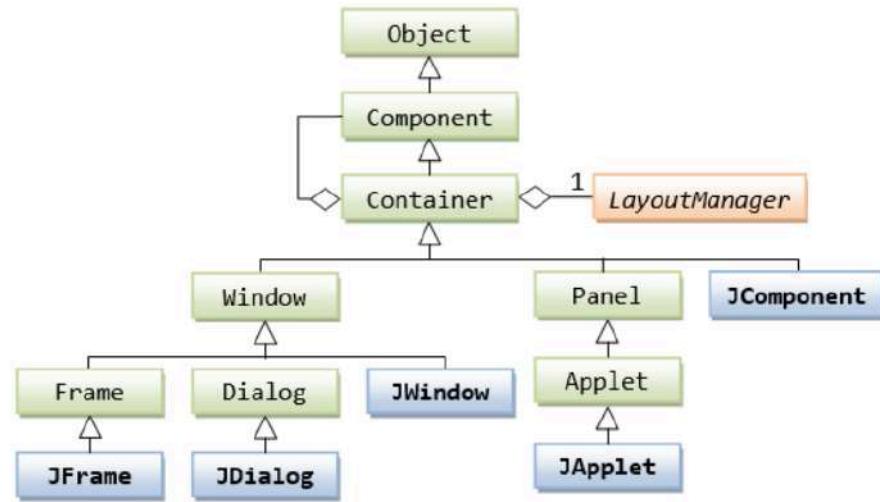
	menggeser pegangan slider.
	Menampilkan progres pekerjaan secara visual, biasanya berupa bar horizontal yang terisi seiring pekerjaan berjalan.
	Menampilkan progres pekerjaan dalam bentuk digital atau bulat
	Menyediakan bilah gulir agar pengguna dapat menggulir konten yang lebih besar dari area tampilan.
Menu	Komponen untuk membuat menu di,



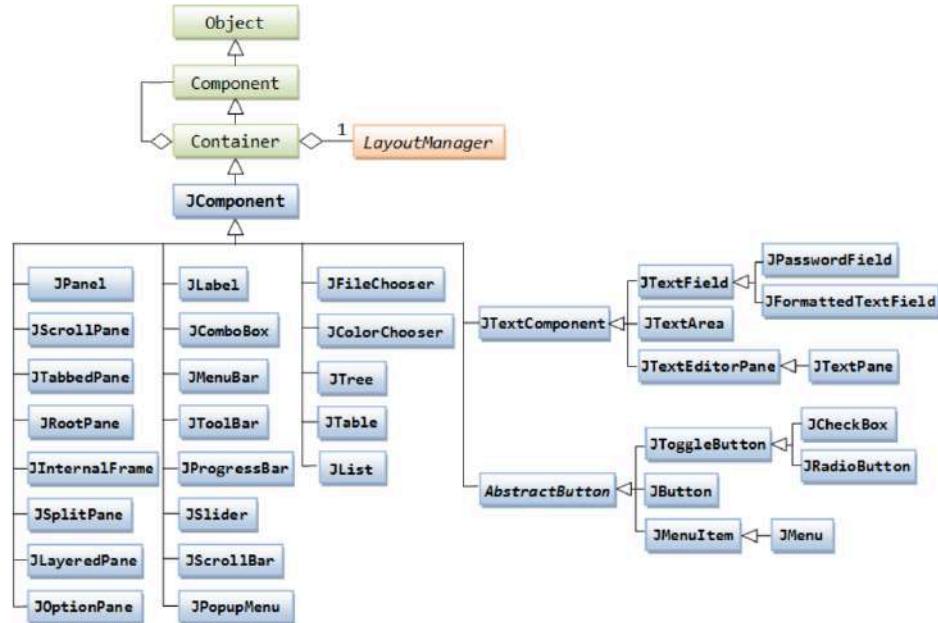
	yang biasanya terdiri dari beberapa item dan submenu.
<h3>ToolTip</h3>  <p>This is a tooltip message</p>	Petunjuk yang muncul saat pengguna mengarahkan kursor ke suatu komponen, misalnya bidang teks atau tombol, untuk memberikan informasi tambahan.

C. Swing Tutorial

1. Struktur Library Java Swing



Java Swing memiliki struktur hierarki yang dimulai dari **Object**, kemudian **Component** sebagai dasar semua elemen GUI, dan **Container** yang dapat menampung komponen lain. Dari sini muncul berbagai jenis jendela seperti **Window**, **Frame**, **JFrame**, **Dialog**, **JDialog**, dan **JWindow** sebagai top-level containers. Ada juga **Panel** untuk mengelompokkan komponen, serta **Applet/JApplet** untuk aplikasi berbasis browser.



Dari **JComponent**, muncul banyak komponen seperti JPanel, JScrollPane dan JTabbedPane sebagai tempat atau wadah untuk menampilkan dan mengatur elemen-elemen lainnya. Ada juga komponen tampilannya, seperti JLabel, JProgressBar, JSlider, JToolBar dan JColorChooser yang berfungsi untuk menampilkan informasi atau menyediakan fitur tambahan yang bisa digunakan oleh pengguna.

Untuk bagian input teks, Swing menyediakan komponen turunan dari **JTextComponent**, misalnya JTextField, JPasswordField dan JTextArea. Komponen-komponen ini digunakan saat aplikasi perlu menerima atau menampilkan teks dari pengguna. Sementara itu, kelompok tombol berasal dari **AbstractButton**, seperti JButton, JToggleButton, JCheckBox dan JRadioButton. Komponen tersebut digunakan untuk membuat tombol, pilihan, dan menu yang memudahkan pengguna menjalankan suatu aksi atau navigasi dalam aplikasi.

2. Istilah-Istilah dalam Java Swing

Istilah-istilah dalam Java Swing yang harus dipahami :

- a. **Container**: Subkelas dari Component yang bisa menampung komponen lain dan mengatur tata letaknya, misalnya JPanel atau JFrame.
- b. **JComponent**: Kelas abstrak dasar untuk semua komponen Swing yang menawarkan fitur tambahan seperti ToolTip dan dukungan untuk tampilan yang dapat diubah (pluggable look-and-feel).
- c. **Window**: Kelas dasar untuk semua jendela GUI yang tidak memiliki dekorasi standar seperti border atau tombol kontrol.
- d. **Panel**: Wadah sederhana untuk mengelompokkan dan menata komponen GUI, biasanya digunakan sebagai area logis dalam antarmuka.
- e. **Applet**: Program Java kecil yang berjalan di browser (jarang digunakan sekarang), memungkinkan aplikasi interaktif di halaman web.
- f. **Frame**: Jendela utama dengan border dan tombol kontrol (minimize, maximize, close), biasanya diimplementasikan dengan JFrame.
- g. **Dialog**: Jendela tambahan untuk interaksi sementara, seperti meminta input atau menampilkan notifikasi, umumnya menggunakan JDialog.

3. Java JFrame

JFrame adalah salah satu komponen penting dalam Java Swing yang berfungsi membuat jendela aplikasi GUI. Bisa dibayangkan sebagai "kotak" atau "bingkai" tempat kita menempatkan berbagai komponen seperti tombol, label, atau bidang teks untuk membangun antarmuka pengguna.

Konstruktor-konstruktor pada kelas JFrame :

Constructor	Deskripsi
JFrame()	Konstruktor default untuk membuat sebuah frame baru tanpa judul.
JFrame(String title)	Konstruktor untuk membuat sebuah frame baru dengan judul sesuai teks yang diberikan.

Contoh Penggunaan JFrame

```

  ● ● ●
public class JFrameExample {
    public static void main(String[] args) {
        // Create a new JFrame Instance
        JFrame frame = new JFrame("My First JFrame");

        // Set the size of the frame (width, height)
        frame.setSize(400,300);

        // Make the frame visible
        frame.setVisible(true);

        // Specify what happens when the user closes the window
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}

```

Output :



Untuk source code nya dapat teman-teman cek ataupun clone melalui link [ini](#)

4. Java JButton

JButton adalah kelas di Java yang digunakan untuk membuat tombol (push button) dalam antarmuka pengguna (GUI). Komponen ini dapat diberi label, menghasilkan event saat ditekan.

Konstruktor-konstruktor pada kelas JButton :

Constructor	Deskripsi
JButton()	Membuat tombol tanpa teks maupun ikon
JButton(String text)	Membuat tombol dengan teks
JButton(Icon i)	Membuat tombol dengan ikon
JButton(String text, Icon i)	Membuat tombol dengan teks dan ikon sekaligus.
JButton(Action a)	Membuat tombol dengan properti yang diambil dari objek Action yang diberikan

Metode-metode umum yang sering digunakan dalam kelas AbstractButton :

Methods	Deskripsi
void setText(String s)	Mengubah teks yang muncul di tombol
String getText()	Mengubah teks yang muncul di tombol
void setEnabled(boolean b)	Mengaktifkan atau menonaktifkan tombol
void setIcon(Icon icon)	Mengatur ikon (gambar) tombol
Icon getIcon()	Mendapatkan ikon yang saat ini ada di tombol
void addActionListener(ActionListener listener)	Menambahkan listener untuk merespons klik tombol
void setToolTipText(String text)	Mengatur teks tooltip saat kursor di atas tombol
String getToolTipText()	Mengambil teks tooltip tombol
Void setBorder(Border border)	Mengatur batas (border) tombol
Border getBorder()	Mendapatkan batas (border) tombol
void setForeground(Color fg)	Mengubah warna teks tombol
Color getForeground()	Mendapatkan warna teks tombol saat ini
void setBackground(Color bg)	Mengubah warna latar belakang tombol

Color getBackground()	Mendapatkan warna latar belakang tombol saat ini
-----------------------	--

Contoh Penggunaan JButton :

```

public class JButtonSimple {
    public static void main(String[] args) {

        // Create frame
        JFrame frame = new JFrame("Styled Button Example");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(300, 200);

        // Create panel for adjust position
        JPanel panel = new JPanel();
        panel.setLayout(new FlowLayout(FlowLayout.CENTER, 0, 50));

        // Create button
        JButton button = new JButton("Click Me");

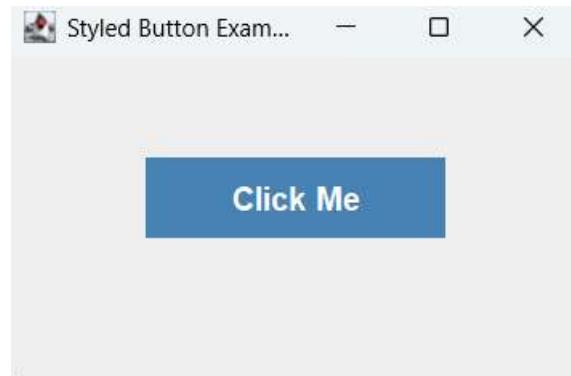
        // Styling button
        button.setPreferredSize(new Dimension(150, 40));
        button.setFont(new Font("Arial", Font.BOLD, 16));
        button.setBackground(new Color(70, 130, 180));
        button.setForeground(Color.WHITE);
        button.setFocusPainted(false);
        button.setBorder(BorderFactory.createEmptyBorder(10, 20, 10, 20));

        // Add action when clicked
        button.addActionListener(e ->
            JOptionPane.showMessageDialog(frame, "Button clicked!")
        );

        // Add button ke panel
        panel.add(button);

        // Add panel ke frame
        frame.add(panel);

        // Show frame
        frame.setVisible(true);
    }
}
  
```



Program diatas membuat sebuah tombol menggunakan **JButton** yang memiliki teks berupa “Click Me” dan diberi ukuran tertentu, warna latar, warna teks, serta font tebal supaya tampil lebih menarik.

```
// Add action when clicked
button.addActionListener(e ->
    JOptionPane.showMessageDialog(frame, "Button clicked!")
);
```

Untuk menangani kondisi apabila tombol di klik, tombol diberi **ActionListener** melalui **button.addActionListener()**. Ketika tombol ditekan, kode di dalam listener langsung dijalankan. Pada contoh ini, program menampilkan popup pesan “Button clicked!” menggunakan **JOptionPane.showMessageDialog()**.

5. Java JLabel

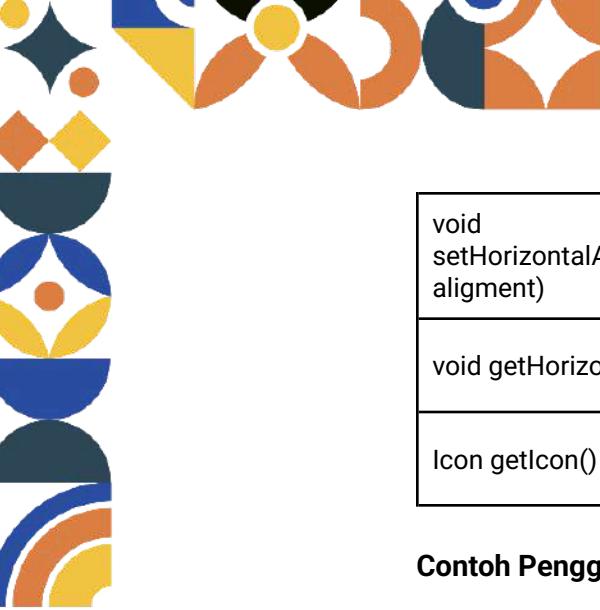
JLabel adalah sebuah kelas dalam Java Swing yang digunakan untuk menampilkan teks, gambar, atau keduanya dalam sebuah wadah (container). **JLabel** menampilkan teks satu baris yang bersifat **read-only**, artinya pengguna tidak dapat mengeditnya secara langsung.

Konstruktor-konstruktor pada Kelas JLabel :

Constructor	Deskripsi
JLabel()	Membuat label kosong tanpa teks maupun gambar.
JLabel(String text)	Membuat label baru dengan teks sesuai string yang diberikan.
JLabel(int columns)	Membuat label baru yang berisi sebuah gambar (icon).
JLabel(String text, int columns)	Membuat label baru dengan teks, gambar,

Metode-metode yang ada dalam kelas JLabel:

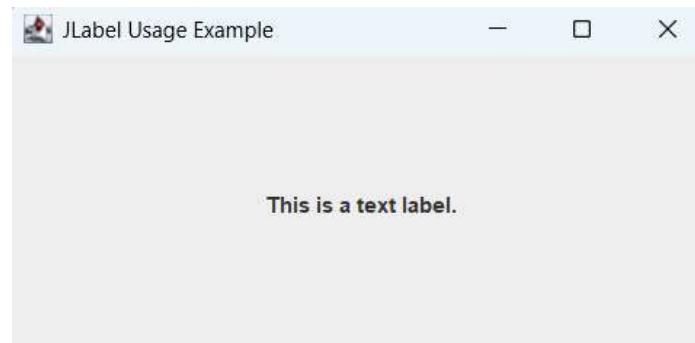
Methods	Deskripsi
String getText()	Mengambil atau mengembalikan teks yang saat ini ditampilkan pada JLabel.
void setText(String text)	Mengatur atau mengganti teks yang ingin ditampilkan oleh JLabel.



void setHorizontalAlignment(int alignment)	Menentukan posisi teks atau ikon JLabel secara horizontal di dalam label.
void getHorizontalAlignment()	Mengembalikan nilai posisi penyelarasan horizontal dari isi JLabel.
Icon getIcon()	Mengambil ikon atau gambar yang sedang ditampilkan pada JLabel.

Contoh Penggunaan JLabel :

```
public class JLabelExample {  
    public static void main(String[] args) {  
        // Create a JFrame  
        JFrame frame = new JFrame("JLabel Usage Example");  
        frame.setSize(400, 200);  
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
        frame.setVisible(true);  
  
        // Create a JLabel with text  
        JLabel label = new JLabel("This is a text label.");  
  
        // Set Label alignment to center  
        label.setHorizontalAlignment(JLabel.CENTER);  
  
        // Add Label to the JFrame  
        frame.add(label);  
    }  
}
```



6. Java JTextField

JTextField adalah salah satu komponen dari paket javax.swing yang memungkinkan pengguna untuk memasukkan atau mengedit teks dalam satu baris.



Konstruktor-konstruktor pada kelas JTextField:

Constructor	Deskripsi
JTextField()	Membuat sebuah TextField baru kosong tanpa teks awal dan ukuran kolom default.
JTextField(String text)	membuat sebuah TextField baru yang sudah berisi teks awal
JTextField(int columns)	Membuat sebuah TextField kosong dengan jumlah kolom tertentu
JTextField(String text, int columns)	Membuat sebuah TextField baru dengan teks awal sesuai text dan jumlah kolom tertentu sesuai columns.

Method-Method yang sering digunakan dalam kelas JTextField:

Method	Deskripsi
void addActionListener(Action Listener l)	Digunakan untuk menambahkan ActionListener tertentu sehingga JTextField bisa mengirim event aksi ke listener tersebut saat terjadi interaksi (misal: menekan Enter).
Action getAction()	Mengembalikan Action yang saat ini terkait dengan JTextField ini. Jika tidak ada aksi yang ditetapkan, maka akan mengembalikan null.
void setFont(Font f)	Digunakan untuk mengatur jenis dan ukuran font yang digunakan
void removeActionListener(ActionListener l)	Digunakan untuk menghapus ActionListener tertentu sehingga listener tersebut tidak lagi menerima event aksi dari JTextField.

Contoh Penggunaan JTextField :

```

● ● ●

class TextFieldExample {
    public static void main(String[] args) {

        JFrame frame = new JFrame("TextField Example");
        frame.setSize(400, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Create a panel using GridLayout to arrange components in 3 rows
        JPanel panel = new JPanel(new GridLayout(3, 1, 5, 10));
        panel.setBorder(BorderFactory.createEmptyBorder(15, 15, 15, 15)); // Add padding around panel.

        // Create a text field with 20 columns width
        JTextField textField = new JTextField(20);

        // Create a button that will trigger the action
        JButton button = new JButton("Show Text");

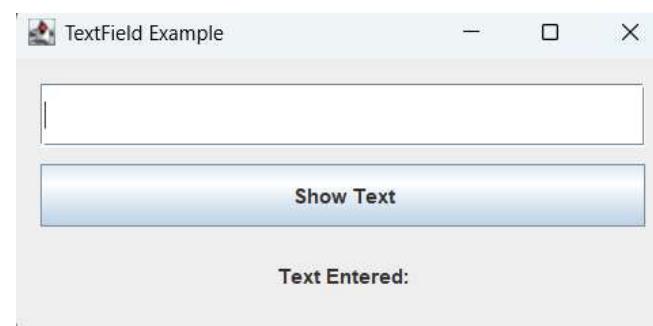
        // Create a Label to display the result text
        JLabel label = new JLabel("Text Entered:", SwingConstants.CENTER);

        // Event handler for button click
        button.addActionListener(e -> {
            String text = textField.getText(); // Get text from the input field
            label.setText("Text Entered: " + text);
        });

        // Add components to the panel in order
        panel.add(textField);
        panel.add(button);
        panel.add(label);

        // Add the panel to the frame
        frame.add(panel);

        // Make the frame visible
        frame.setVisible(true);
    }
}
  
```



7. Java JTextArea

JTextArea adalah bagian dari paket Java Swing yang digunakan untuk membuat area teks yang bisa menampilkan dan mengedit teks dalam beberapa baris. Berbeda dengan JTextField yang hanya untuk satu baris, JTextArea memungkinkan pengguna mengetik, menampilkan, atau memodifikasi teks yang terdiri dari beberapa baris.

Konstruktor-konstruktor pada kelas JTextArea:

Constructor	Deskripsi
JTextArea()	Membuat area teks kosong tanpa teks awal dan ukuran default.
JTextArea(String text)	Membuat area teks baru dengan teks awal.
JTextArea(int row, int column)	Membuat area teks kosong dengan jumlah baris (row) dan kolom (column) tertentu.
JTextArea(String s, int row, int column)	Membuat area teks baru dengan teks awal s dan jumlah baris serta kolom tertentu.

Method-Method yang ada dalam kelas JTextArea:

Method	Deskripsi
void setRows(int rows)	Mengatur jumlah baris pada JTextArea sesuai nilai rows.
void setColumns(int cols)	Mengatur jumlah kolom pada JTextArea sesuai nilai cols.
void setFont(Font f)	Mengatur font teks yang digunakan pada JTextArea.
void insert(String s, int position)	Menyisipkan tekss pada posisi tertentu position dalam JTextArea.
void append(String s)	Menambahkan tekss di akhir konten teks JTextArea.
int getRows()	Mengambil jumlah baris saat ini pada JTextArea.
int getColumns()	Mengambil jumlah kolom saat ini pada JTextArea.
Font getFont()	Mengambil font yang sedang digunakan pada JTextArea.
void setText(String text)	Mengatur konten teks JTextArea menjadi text yang ditentukan.
String getText()	Mengambil konten teks saat ini dari JTextArea.
void setEditable(boolean editable)	Menentukan apakah JTextArea bisa diedit (true) atau tidak (false).

void setLineWrap(boolean wrap)	Menentukan apakah teks harus dibungkus otomatis jika melebihi lebar area.
void setWrapStyleWord(boolean wrapWord)	Menentukan apakah teks dibungkus pada spasi terdekat sehingga kata tidak terpotong.
void replaceRange(String str, int start, int end)	Mengganti teks dalam rentang tertentu start sampai end dengan teks str.
void setForeground(Color fg)	Mengatur warna teks (foreground) dari JTextArea.
void setBackground(Color bg)	Mengatur warna latar belakang (background) dari JTextArea.

Contoh Penggunaan JTextArea :

```

  ● ● ●

public class JTextAreaExample {
    public static void main(String[] args) {
        // Create JFrame
        JFrame frame = new JFrame("Simple JTextArea Example");
        frame.setSize(400,300);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(new FlowLayout());

        // Create JTextArea with 5 rows and 20 columns
        JTextArea textArea = new JTextArea(5, 20);
        textArea.setLineWrap(true); // Enable automatic line wrapping
        textArea.setWrapStyleWord(true); // Wrap at word boundaries
        JScrollPane scrollPane = new JScrollPane(textArea); // Add scroll pane for long text

        // Button to append text
        JButton appendButton = new JButton("Append Text");
        appendButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                textArea.append(" Additional text.\n");
            }
        });

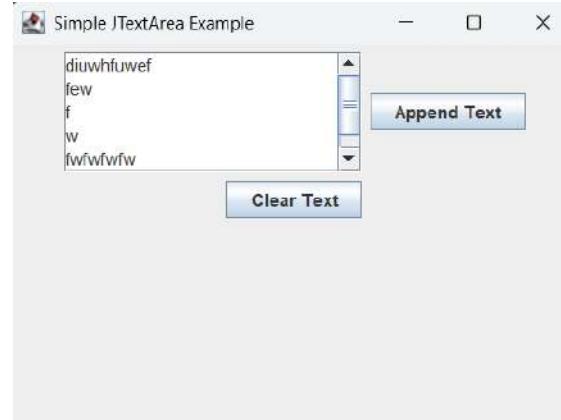
        // Button to clear text
        JButton clearButton = new JButton("Clear Text");
        clearButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                textArea.setText("");
            }
        });

        frame.add(scrollPane);
        frame.add(appendButton);
        frame.add(clearButton);

        frame.setVisible(true);
    }
}

```





8. Java JTable

JTable adalah kelas dalam paket Java Swing yang umumnya digunakan untuk menampilkan atau mengedit data dua dimensi yang memiliki baris dan kolom. JTable mirip seperti spreadsheet, di mana data disusun dalam bentuk tabel.

Konstruktor-konstruktor pada kelas JTable :

Construtor	Deskripsi
JTable()	Membuat tabel dengan sel kosong
JTable(int rows, int cols)	Membuat tabel dengan ukuran rows × cols (baris × kolom) namun seluruh sel akan kosong.
JTable(Object[][] data, Object[] column)	Membuat tabel dengan data yang sudah ada

Contoh Penggunaan JTable :

```
public class JTableExample {
    public static void main(String[] args) {
        // Create JFrame
        JFrame frame = new JFrame("JTable Example");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(500, 350);
        frame.setLayout(new FlowLayout());

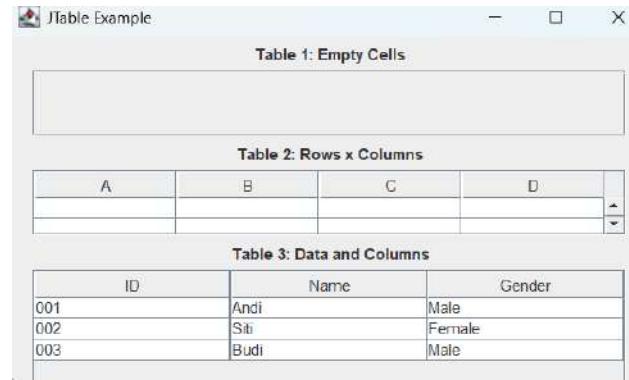
        // ***** 1. JTable with empty cells *****
        JTable table1 = new JTable(); // empty table
        JScrollPane scroll1 = new JScrollPane(table1);
        scroll1.setPreferredSize(new Dimension(450, 50));

        // ***** 2. JTable with specific rows and columns *****
        JTable table2 = new JTable(3, 3); // empty table
        JScrollPane scroll2 = new JScrollPane(table2);
        scroll2.setPreferredSize(new Dimension(450, 50));

        // ***** 3. JTable with data and column names *****
        Object[][] data = {
            {"1", "Andi", "Male"},
            {"2", "Siti", "Female"},
            {"3", "Budi", "Male"}
        };
        String[] columns = {"ID", "Name", "Gender"};
        JTable table3 = new JTable(data, columns);
        JScrollPane scroll3 = new JScrollPane(table3);
        scroll3.setPreferredSize(new Dimension(450, 75));

        // Add tables to JFrame
        frame.add(new JLabel("Table 1: Empty Cells"));
        frame.add(scroll1);
        frame.add(new JLabel("Table 2: Rows & Columns"));
        frame.add(scroll2);
        frame.add(new JLabel("Table 3: Data & Column Names"));
        frame.add(scroll3);

        // Show frame
        frame.setVisible(true);
    }
}
```



9. Java Swing Layout Manager

Di Java Swing, layout manager adalah mekanisme yang digunakan untuk mengatur posisi dan ukuran komponen GUI di dalam container seperti JPanel atau JFrame. Layout manager membantu programmer agar tidak perlu mengatur posisi komponen secara manual, sehingga tampilan antarmuka lebih fleksibel dan responsif terhadap ukuran jendela.

Meskipun Swing adalah library GUI modern, kebanyakan layout manager yang digunakan di Swing sebenarnya berasal dari AWT (java.awt), karena Swing dibangun di atas AWT. Hal ini membuat Swing kompatibel dengan layout lama sambil menambahkan beberapa fitur baru khusus Swing. Berikut adalah contoh layout manager yang biasanya digunakan :

a. **FlowLayout** (java.awt.FlowLayout)

- Menata komponen berurutan secara horizontal, lalu turun ke baris berikutnya jika tidak muat.
- Default layout untuk JPanel.



```

● ● ●

import javax.swing.*;
import java.awt.*;

public class FlowLayoutExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame("FlowLayout Example");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 100);

        JPanel panel = new JPanel();
        panel.setLayout(new FlowLayout()); // bisa FlowLayout.LEFT, RIGHT, CENTER

        panel.add(new JButton("Button 1"));
        panel.add(new JButton("Button 2"));
        panel.add(new JButton("Button 3"));
        panel.add(new JButton("Button 4"));

        frame.add(panel);
        frame.setVisible(true);
    }
}

```



b. **BorderLayout** (java.awt.BorderLayout)

- Membagi container menjadi 5 area: NORTH, SOUTH, EAST, WEST, CENTER.
- Default layout untuk JFrame.

```

● ● ●

import javax.swing.*;
import java.awt.*;

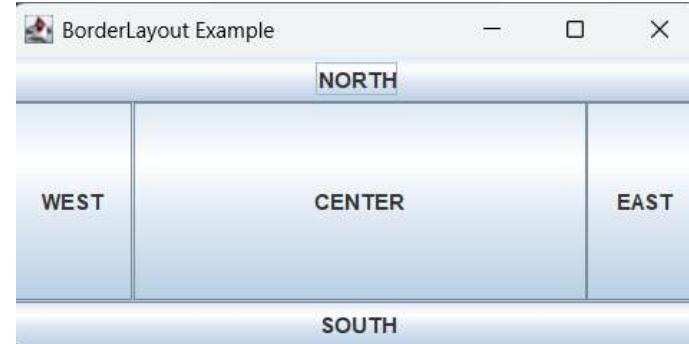
public class BorderLayoutExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame("BorderLayout Example");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 200);

        frame.setLayout(new BorderLayout());

        frame.add(new JButton("NORTH"), BorderLayout.NORTH);
        frame.add(new JButton("SOUTH"), BorderLayout.SOUTH);
        frame.add(new JButton("EAST"), BorderLayout.EAST);
        frame.add(new JButton("WEST"), BorderLayout.WEST);
        frame.add(new JButton("CENTER"), BorderLayout.CENTER);

        frame.setVisible(true);
    }
}

```



c. GridLayout (java.awt.GridLayout)

- Menata komponen dalam grid/baris dan kolom sama besar.

```

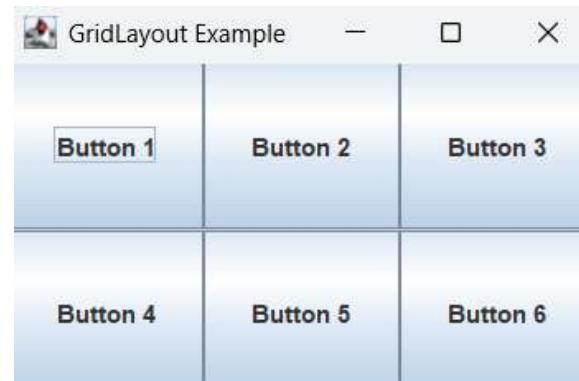
import javax.swing.*;
import java.awt.*;

public class GridLayoutExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame("GridLayout Example");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(300, 200);

        JPanel panel = new JPanel();
        panel.setLayout(new GridLayout(2, 3)); // 2 baris, 3 kolom

        for (int i = 1; i <= 6; i++) {
            panel.add(new JButton("Button " + i));
        }

        frame.add(panel);
        frame.setVisible(true);
    }
}
  
```



d. **GridBagLayout** (java.awt.GridBagLayout)

- Paling fleksibel; bisa mengatur posisi, ukuran, dan berat setiap komponen dalam grid.

```

  ● ● ●

import javax.swing.*;
import java.awt.*;

public class GridBagLayoutExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame("GridBagLayout Example");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 200);

        JPanel panel = new JPanel();
        panel.setLayout(new GridBagLayout());
        GridBagConstraints gbc = new GridBagConstraints();

        gbc.fill = GridBagConstraints.HORIZONTAL;

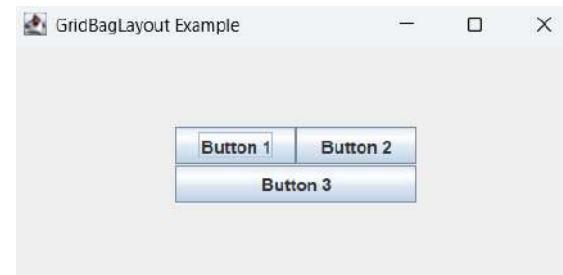
        gbc.gridx = 0;
        gbc.gridy = 0;
        panel.add(new JButton("Button 1"), gbc);

        gbc.gridx = 1;
        gbc.gridy = 0;
        panel.add(new JButton("Button 2"), gbc);

        gbc.gridx = 0;
        gbc.gridy = 1;
        gbc.gridwidth = 2; // span 2 kolom
        panel.add(new JButton("Button 3"), gbc);

        frame.add(panel);
        frame.setVisible(true);
    }
}

```



e. CardLayout (java.awt.CardLayout)

- Mengatur panel seperti kartu yang bisa diganti-ganti. Cocok untuk wizard atau tab custom.

```

● ● ●

package org.example.layoutManager;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;

public class CardLayoutExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame("CardLayout Example");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(300, 200);

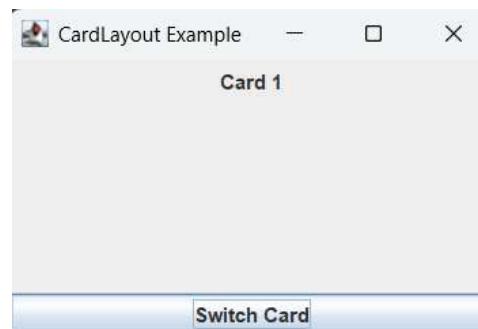
        JPanel panel = new JPanel(new CardLayout());
        JPanel card1 = new JPanel();
        card1.add(new JLabel("Card 1"));
        JPanel card2 = new JPanel();
        card2.add(new JLabel("Card 2"));

        panel.add(card1, "Card1");
        panel.add(card2, "Card2");

        JButton switchButton = new JButton("Switch Card");
        switchButton.addActionListener((ActionEvent e) -> {
            CardLayout cl = (CardLayout)(panel.getLayout());
            cl.next(panel);
        });

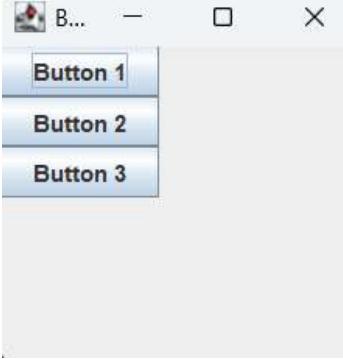
        frame.setLayout(new BorderLayout());
        frame.add(panel, BorderLayout.CENTER);
        frame.add(switchButton, BorderLayout.SOUTH);

        frame.setVisible(true);
    }
}
  
```



f. BoxLayout (javax.swing.BoxLayout)

- Layout manager yang murni Swing, digunakan untuk menata komponen secara horizontal atau vertikal.
- Contohnya, stack button secara vertikal di JPanel.



```

import javax.swing.*;

public class BoxLayoutExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame("BoxLayout Example");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(200, 200);

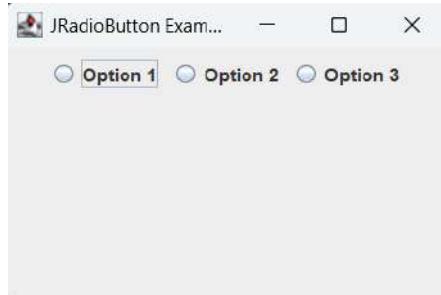
        JPanel panel = new JPanel();
        panel.setLayout(new BoxLayout(panel, BoxLayout.Y_AXIS)); // Y_AXIS = vertikal

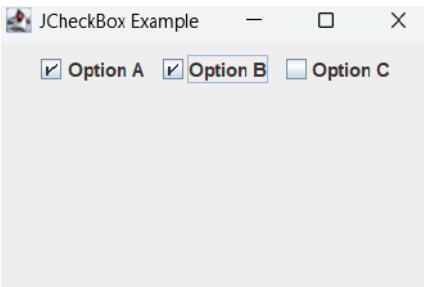
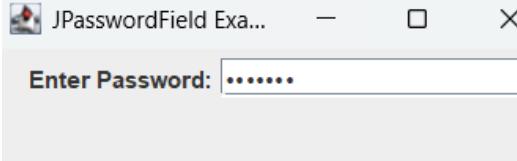
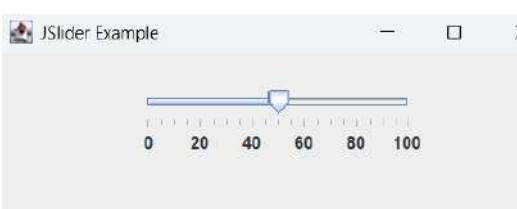
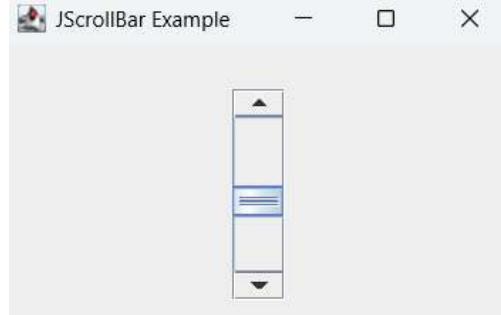
        panel.add(new JButton("Button 1"));
        panel.add(new JButton("Button 2"));
        panel.add(new JButton("Button 3"));

        frame.add(panel);
        frame.setVisible(true);
    }
}

```

10. Contoh Java Swing Lainnya

Constrcutor	Deskripsi
JRadioButton 	Radio button memungkinkan pengguna memilih satu opsi dari beberapa opsi. Biasanya digunakan pada formulir, kuis, atau soal pilihan ganda.

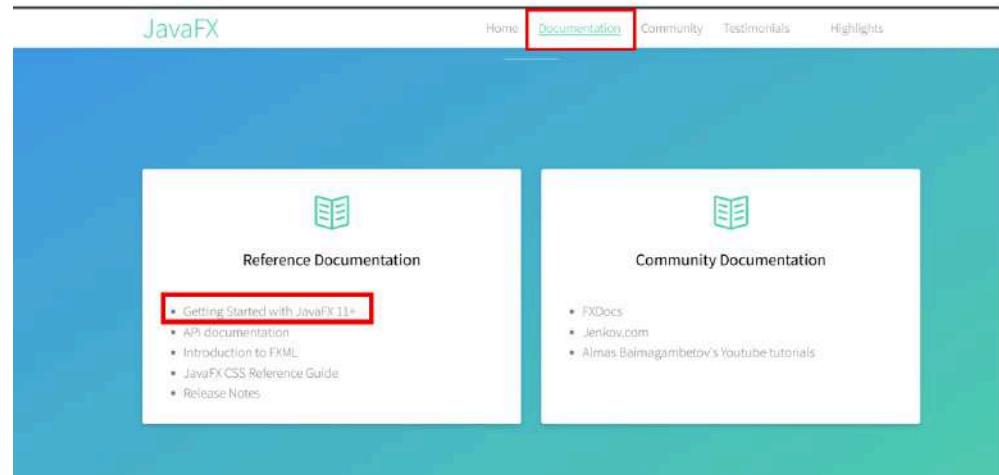
<h3>JCheckBox</h3> 	<p>JCheckBox adalah kelas di Java yang digunakan untuk membuat sebuah checkbox. Checkbox ini memungkinkan pengguna untuk mengaktifkan (true) atau menonaktifkan (false) suatu opsi.</p>
<h3>JPasswordField</h3> 	<p>Komponen ini memungkinkan pengguna mengetik satu baris teks, tetapi karakter yang diketik tidak ditampilkan (diganti dengan simbol seperti *).</p>
<h3>JSlider</h3> 	<p>Digunakan untuk membuat slider yang memungkinkan pengguna memilih nilai dengan menggeser knob</p>
<h3>JProgressBar</h3> 	<p>Digunakan untuk menampilkan progress dari suatu tugas. Progress bar menunjukkan persentase penyelesaian tugas dan bisa menampilkan teks tertentu. Semakin tugas mendekati selesai, progress bar semakin penuh.</p>
<h3>JScrollBar</h3> 	<p>Digunakan untuk membuat scroll bar agar pengguna bisa meng gulir konten dari suatu komponen, misalnya JScrollPane</p>

MATERI PRAKTIKUM

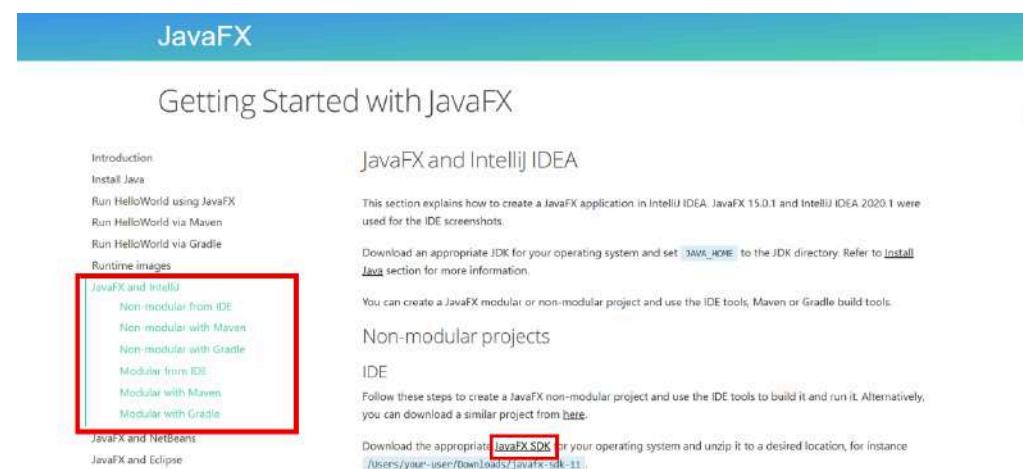
A. INSTALASI DAN KONFIGURASI JAVAFX

1. Download JavaFX SDK

- Buka situs openjfx.io. Lalu, pada bagian *Documentation*, klik menu *Getting Started with JavaFX*.

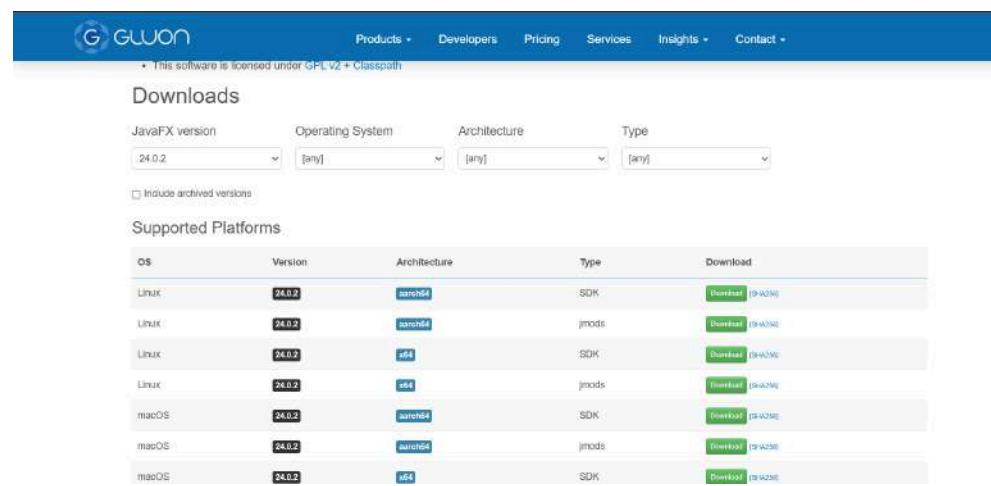


- Jika kamu menggunakan IDE IntelliJ, buka bagian *JavaFX and IntelliJ*. Setelah itu, klik tautan *JavaFX SDK* seperti yang ditunjukkan pada gambar berikut :



The screenshot shows the 'Getting Started with JavaFX' page. In the 'JavaFX and IntelliJ' section, there is a list of links for creating JavaFX applications in IntelliJ IDEA. One of the links, 'JavaFX SDK', is highlighted with a red box. The text next to it says: 'Download the appropriate JavaFX SDK for your operating system and set `JAVA_HOME` to the JDK directory. Refer to [Install Java](#) section for more information.' Below this, there is a 'Non-modular' section with links for JavaFX and NetBeans, JavaFX and Eclipse, and JavaFX and IntelliJ. The 'JavaFX and IntelliJ' section itself has sub-links for Non-modular from IDE, Non-modular with Maven, Non-modular with Gradle, Modular from IDE, Modular with Maven, and Modular with Gradle.

- Setelah itu, kamu akan diarahkan ke halaman [Gluon](#). Scroll Ke bawah, lalu unduh JavaFX sesuai dengan sistem operasi yang kamu gunakan.

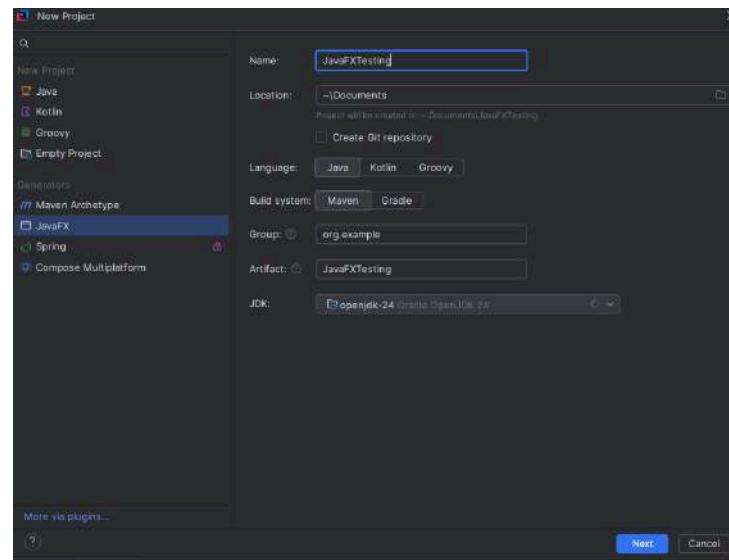


OS	Version	Architecture	Type	Download
Linux	24.0.2	aarch64	SDK	Download
Linux	24.0.2	aarch64	jmods	Download
Linux	24.0.2	x64	SDK	Download
Linux	24.0.2	x64	jmods	Download
macOS	24.0.2	aarch64	SDK	Download
macOS	24.0.2	aarch64	jmods	Download
macOS	24.0.2	x64	SDK	Download

- Ekstrak file hasil unduhan ke lokasi penyimpanan pilihanmu. Pastikan kamu mengingat lokasi ini, karena akan dibutuhkan saat melakukan pengaturan JavaFX di IDE

2. Membuat JavaFX Project

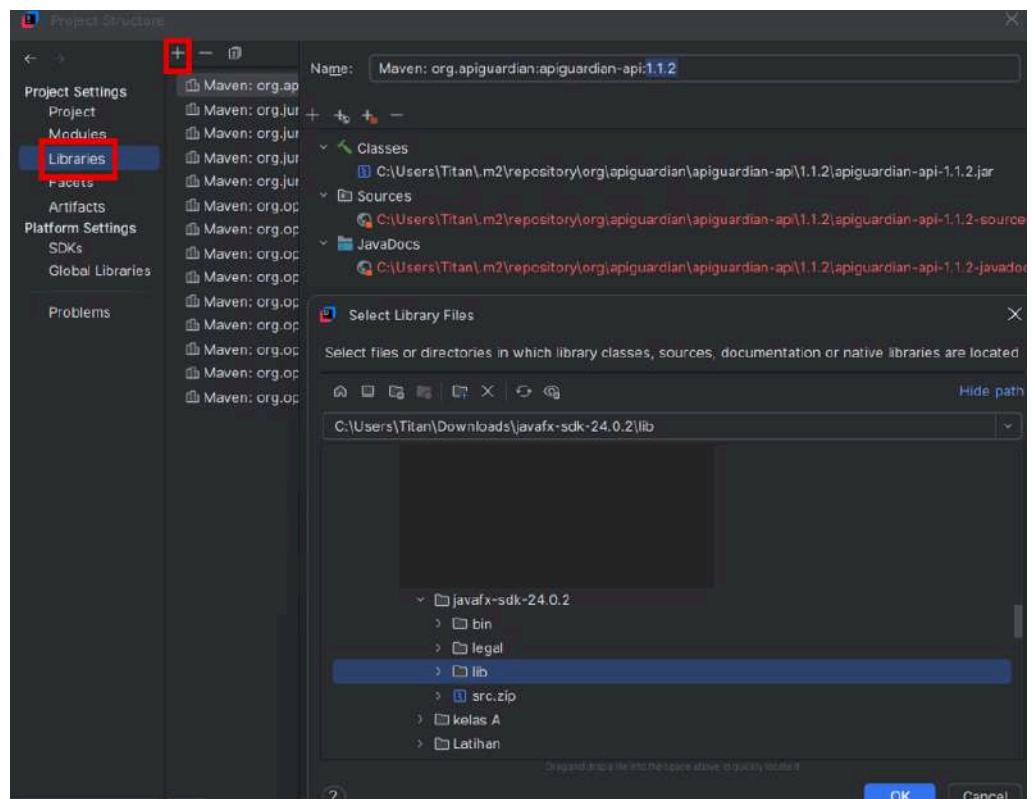
- Buat project JavaFX



Buat project JavaFX baru, klik **Next**, lalu langsung pilih **Create** tanpa mencentang opsi apapun pada halaman berikutnya.

3. Tambahkan SDK Library

- Buka menu **File → Project Structure → Libraries → Icon + → Java**, kemudian tambahkan library project baru seperti yang ditunjukkan pada gambar. Pilih lokasi file library sesuai dengan folder lib dari JavaFX SDK yang telah kamu simpan sebelumnya.



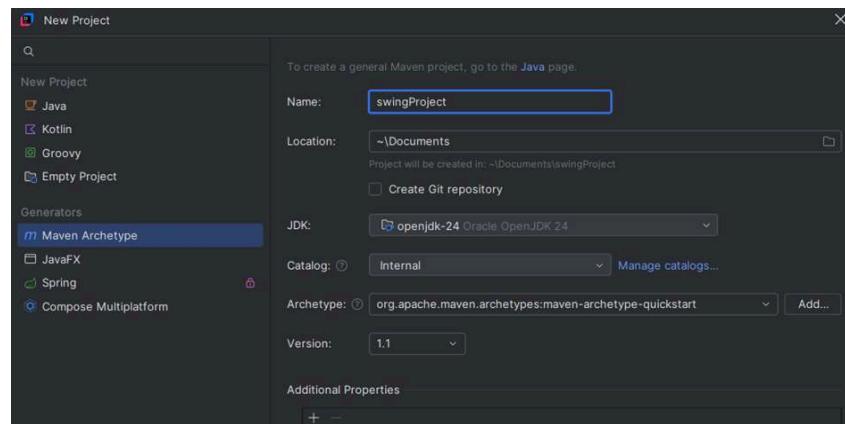
- Jalankan project, dan program akan menampilkan GUI seperti pada gambar di bawah ini



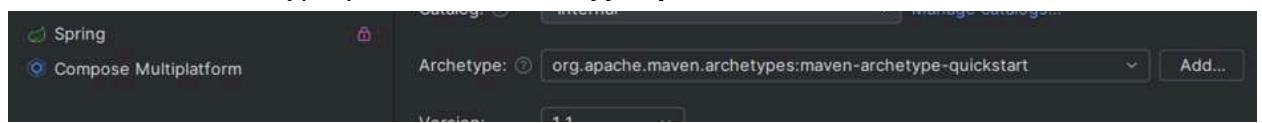
B. INSTALASI DAN KONFIGURASI JAVA SWING

Java Swing sebenarnya sudah termasuk di dalam **JDK**, sehingga kita tidak perlu melakukan instalasi tambahan untuk bisa menggunakannya. Pada praktikum ini, kita menggunakan **IntelliJ IDEA** sebagai IDE IntelliJ mampu melakukan konfigurasi project secara otomatis, termasuk mendeteksi dan mengatur JDK yang terpasang di komputer. Berikut langkah-langkahnya di IntelliJ IDEA:

1. Klik New Project
2. Pilih Maven Archetype pada bagian sebelah kiri (sidebar) lalu isi Name, Location, dan pastikan Project SDK / JDK sudah dipilih



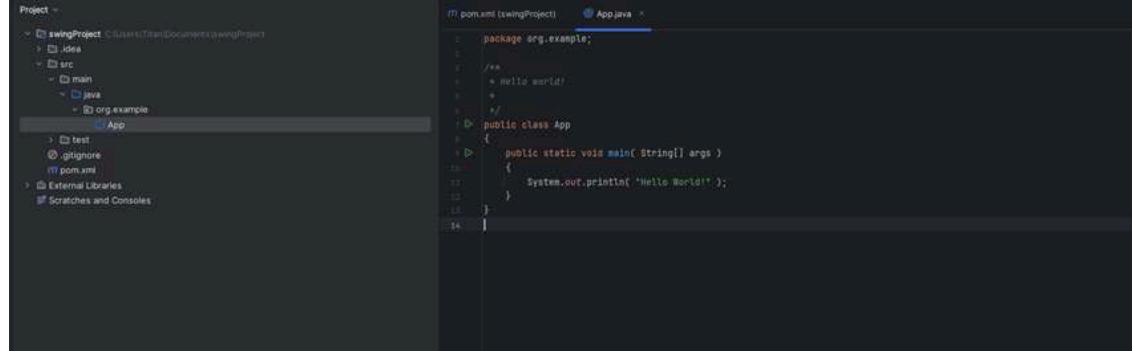
3. Untuk Archetype, pilih **maven-archetype-quickstart**



4. Klik Create and tunggu proses download dependency Maven hingga selesai

```
[INFO] --- archetype:2.4.1:generate (default-cli) > generate-sources @ standalone-pom >>>
[INFO]
[INFO] <<< archetype:2.4.1:generate (default-cli) < generate-sources @ standalone-pom <<<
[INFO]
[INFO]
[INFO] ... archetype:3.4.1:generate (default-cli) @ standalone-pom ...
[INFO] Generating project in Batch mode
[INFO] Archetype repository not defined. Using the one from [org.apache.maven.archetypes:maven-archetype-quickstart:1.1] found in catalog remote
[INFO] ...
[INFO] Using following parameters for creating project from Old (1.x) Archetype: maven-archetype-quickstart:1.1
[INFO] ...
[INFO] Parameter: basedir, Value: C:\Users\Titin\AppData\Local\Temp\archetypetemp\missingProject
[INFO] Parameter: package, Value: org.example
[INFO] Parameter: groupId, Value: org.example
[INFO] Parameter: artifactId, Value: swingProject
[INFO] Parameter: packageName, Value: org.example
[INFO] Parameter: version, Value: 1.0-SNAPSHOT
[INFO] project created from Old (1.x) Archetype in dir: C:\Users\Titin\AppData\Local\Temp\archetypetemp\missingProject
[INFO] ...
[INFO] BUILD SUCCESS
[INFO] ...
[INFO] Total time: 2.354 s
[INFO] Finished at: 2025-11-29T10:11:48+07:00
[INFO] ...
Process finished with exit code 0
```

5. Jika sudah selesai maka tampilannya akan seperti ini



```
pom.xml (swingProject) App.java
```

```
package org.example;
```

```
/*
```

```
 * Hello world!
```

```
 */
```

```
public class App
```

```
{
```

```
    public static void main(String[] args)
```

```
    {
```

```
        System.out.println("Hello World!");
```

```
    }
```

```
}
```

6. Untuk melakukan testing apakah environment sudah berjalan dengan benar, kalian dapat menambahkan kode Java Swing berikut di App.java

```
package org.example;
```

```
import javax.swing.*;
```

```
public class App
```

```
{
```

```
    public static void main(String[] args)
```

```
    {
```

```
        JFrame frame = new JFrame("Demo Swing");
```

```
        frame.add(new JLabel("Hello Swing!", SwingConstants.CENTER));
```

```
        frame.setSize(300, 200);
```

```
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
        frame.setVisible(true);
```

```
    }
```

```
}
```

7. Jika berhasil maka akan mengeluarkan output seperti ini



C. SCENE BUILDER

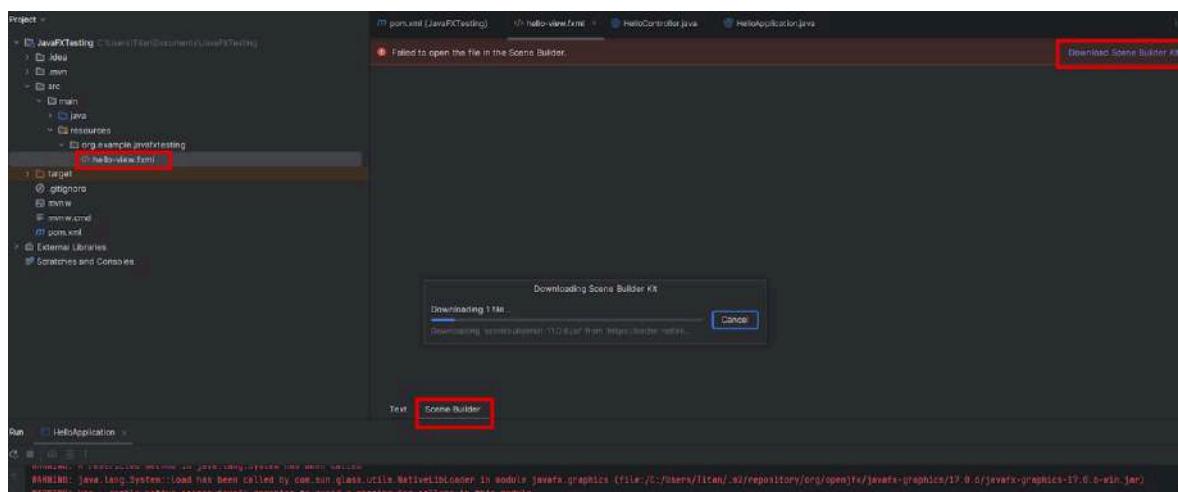
Scene Builder adalah sebuah aplikasi khusus yang digunakan untuk merancang antarmuka pengguna (UI) pada JavaFX secara visual tanpa perlu langsung menulis kode. Dengan pendekatan drag and drop, pengembang dapat menambahkan komponen seperti tombol, label, teks, tabel, gambar, hingga layout container dengan mudah. Hal ini sangat membantu terutama bagi pemula maupun tim pengembang, karena tampilan aplikasi bisa dibuat lebih cepat dan intuitif dibandingkan jika harus menulis kode JavaFX secara manual.

Selain itu, Scene Builder menghasilkan file FXML yang berfungsi sebagai blueprint atau deskripsi dari UI yang dibuat. File ini memisahkan logika program dengan tampilan antarmuka, sehingga pengembang desain dan pengembang logika dapat bekerja secara terpisah namun tetap sinkron. Dengan demikian, Scene Builder tidak hanya mempercepat proses pembuatan UI, tetapi juga menjaga struktur kode lebih rapi, mudah dipelihara, dan mendukung kolaborasi tim yang lebih efisien.

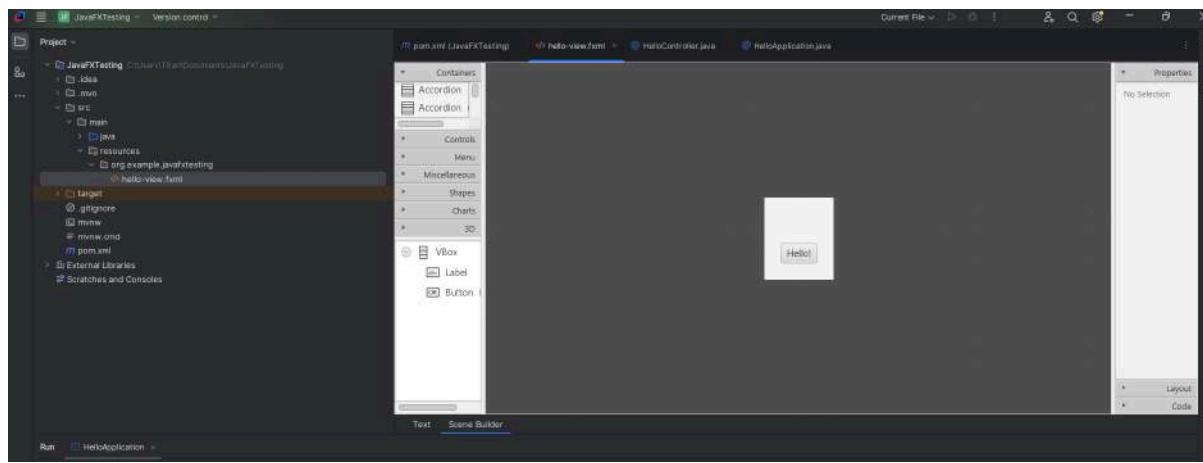
D. CARA PENGGUNAAN SCENE BUILDER

1. Cara 1 : Menggunakan Scene Builder Kit pad IDE

Scene Builder Kit adalah API yang digunakan untuk mengintegrasikan panel Scene Builder beserta fitur-fiturnya langsung ke dalam GUI IDE Java, seperti NetBeans, IntelliJ, maupun Eclipse. Untuk menggunakannya, pertama buka file FXML dari project, kemudian unduh Scene Builder Kit seperti yang ditunjukkan pada gambar, lalu pilih tab Scene Builder di bagian bawah dan lakukan proses download.

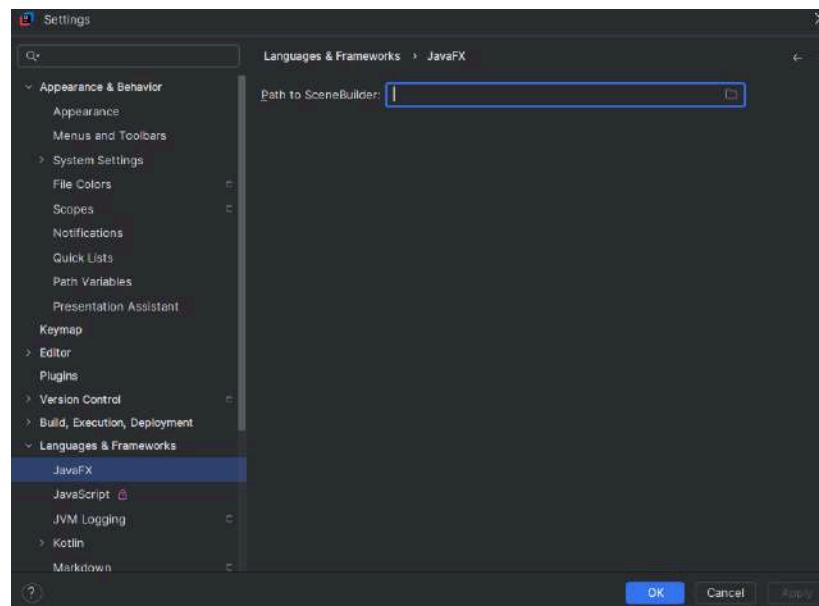


Setelah itu, Anda dapat merancang antarmuka pengguna dengan cara drag and drop untuk mengubah serta mengelola file FXML, sekaligus mempermudah pekerjaan desain GUI yang bersifat berulang.

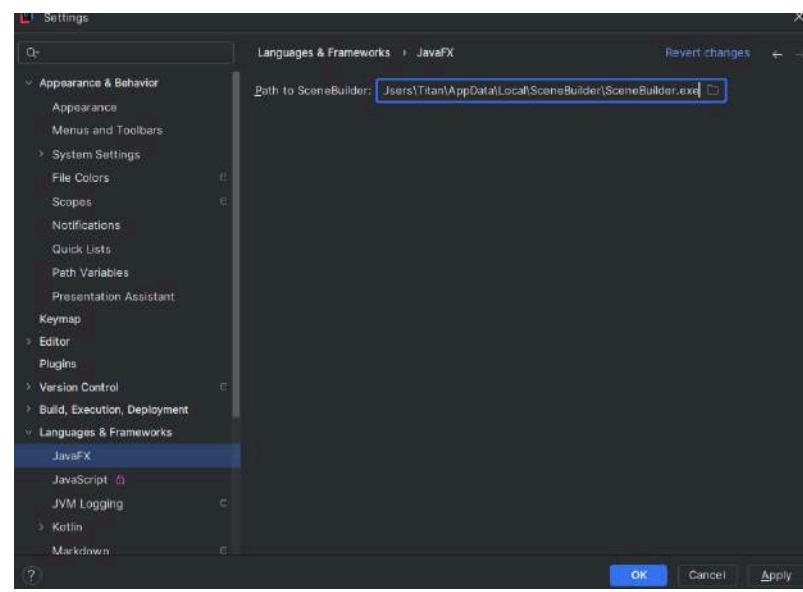


2. Cara 2 : Menggunakan Aplikasi SceneBuilder.exe

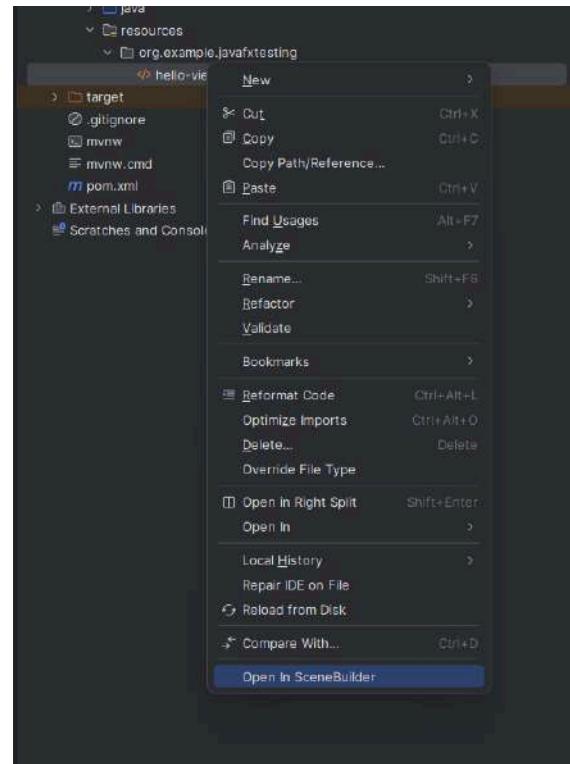
- Unduh Scene Builder melalui tautan [ini](#).
- Lakukan instalasi Scene Builder pada perangkat Anda.
- Setelah terpasang, buka IDE kemudian masuk ke menu **File → Settings → Languages & Frameworks → JavaFX**.



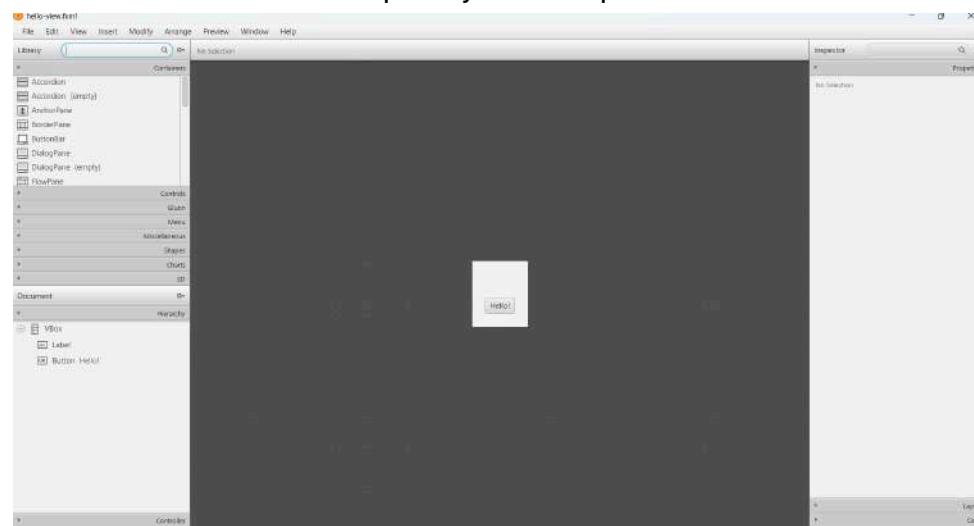
- Masukkan lokasi/path dari file **scenebuilder.exe**, lalu klik **Apply** dan **OK** untuk menyimpan pengaturan.



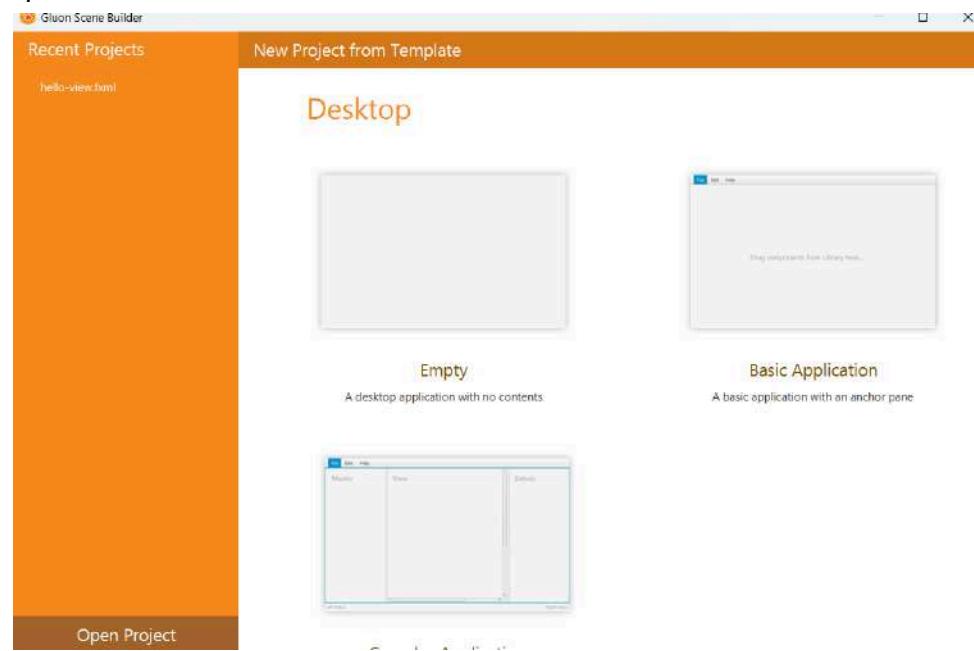
- Jika sudah maka klik kanan pada file FXML di IDE, lalu pilih **Open in Scene Builder**.



- Jika sudah dibuka maka tampilannya akan seperti ini



- Atau, kamu juga bisa membuka Scene Builder secara langsung melalui aplikasi **SceneBuilder**.



REFRENSI

<https://dqlab.id/teknik-pemrograman-gui-dengan-chatgpt>

<https://www.geeksforgeeks.org/java/java.awt-tutorial/>

<https://pressbooks.pub/javaprogramming/chapter/swing-java-programming/>

<https://www.tutorialspoint.com/javafx/index.htm>

<https://docs.oracle.com/javase/8/javafx/api/>

<https://www.tutorialspoint.com/swing/>

https://www3.ntu.edu.sg/home/ehchua/programming/java/J4a_GUI_2.html

https://www3.ntu.edu.sg/home/ehchua/programming/java/JavaFX1_intro.html



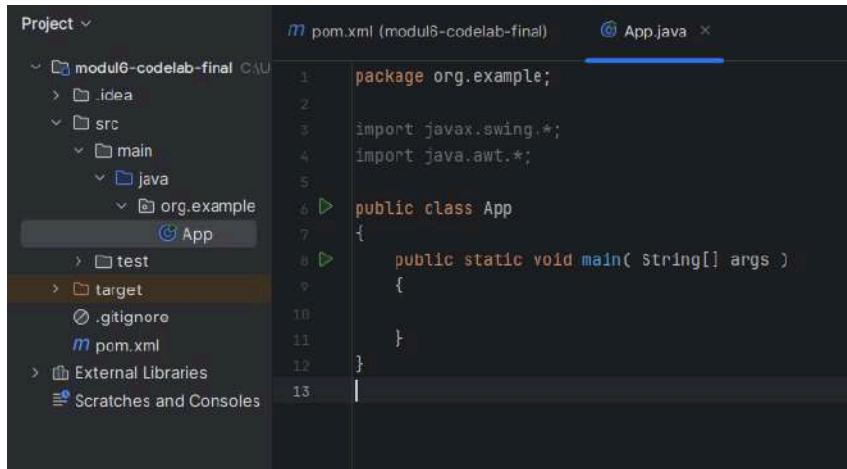
CODELAB

CODELAB 1

Membuat PasswordField merupakan bagian penting dalam pembuatan aplikasi yang aman. Komponen ini biasanya dipakai untuk menyembunyikan data sensitif seperti kata sandi atau PIN agar tidak terlihat langsung oleh orang lain. Di Java Swing, hal tersebut bisa dilakukan menggunakan JPasswordField, yang secara otomatis menampilkan karakter input dalam bentuk simbol bintang (*). Dengan begitu, privasi dan keamanan pengguna lebih terjaga

Berikut langkah-langkah membuat password field dengan java swing :

1. Buat terlebih dahulu sebuah project Java Swing pada IntelliJ IDEA. Setelah project berhasil dibuat, lakukan import library yang diperlukan, yaitu: **javax.swing.*** dan **java.awt.***.



The screenshot shows the IntelliJ IDEA interface. On the left is the Project tool window displaying a file structure for a project named 'modul6-codelab-final'. It includes a .idea folder, a src folder with main and java subfolders, and files like pom.xml, .gitignore, and App.java. The App.java file is open in the main editor area, showing the following code:

```
package org.example;
import javax.swing.*;
import java.awt.*;
public class App
{
    public static void main( String[] args )
    {
    }
}
```

2. Siapkan jendela utama menggunakan JFrame yang nantinya akan menjadi wadah dari komponen UI.

```
public static void main( String[] args )
{
    // Create the main frame
    JFrame frame = new JFrame( "Password Validation" );
    frame.setSize( width: 400, height: 200 );
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setLayout(new FlowLayout());
```



3. Tambahkan label untuk memberikan keterangan pada pengguna.

```
// Label
JLabel label = new JLabel( text: "Enter PIN (6 digits only)");
frame.add(label);
```

4. Buat JPasswordField sebagai tempat mengetikkan PIN.

```
// Password Field
JPasswordField passwordField = new JPasswordField( columns: 10);
frame.add(passwordField);
```

5. Tambahkan sebuah tombol yang akan digunakan untuk melakukan validasi password.

```
// Validation Button
JButton button = new JButton( text: "Validate");
frame.add(button);
```

6. Sertakan kode untuk memproses hasil validasi, yaitu aksi tombol serta logika pengecekan apakah password yang dimasukkan benar atau tidak.

```
// Action for the button
button.addActionListener( ActionEvent e -> {
    String password = new String(passwordField.getPassword());

    if (password.length() != 6) {
        JOptionPane.showMessageDialog(
            frame,
            message: "PIN must be exactly 6 characters long!",
            title: "Error",
            JOptionPane.ERROR_MESSAGE
        );
        return;
    }

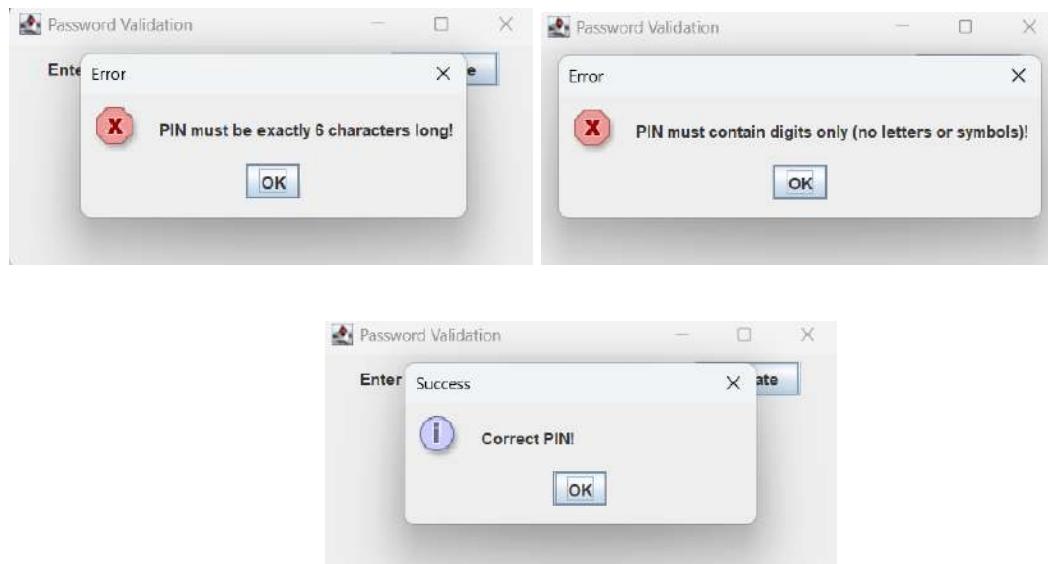
    if (!password.matches( regex: "\\\d+")) {
        JOptionPane.showMessageDialog(
            frame,
            message: "PIN must contain digits only (no letters or symbols)!",
            title: "Error",
            JOptionPane.ERROR_MESSAGE
        );
        return;
    }

    JOptionPane.showMessageDialog(
        frame,
        message: "Correct PIN!",
        title: "Success",
        JOptionPane.INFORMATION_MESSAGE
    );
});
```

7. Terakhir, jangan lupa tambahkan perintah agar frame dapat ditampilkan ke layar.

```
// Show the frame  
frame.setVisible(true);
```

8. Berikut adalah beberapa tampilan jika pin yang dimasukkan valid dan tidak valid



CODELAB 2

Membuat Table Java Swing

- Buat terlebih dahulu sebuah project baru Java Swing pada IntelliJ IDEA. Setelah project berhasil dibuat, lakukan import library yang diperlukan yaitu :

```
import javax.swing.*;
import javax.swing.border.EmptyBorder;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
```

- Buat sebuah JFrame yang akan menjadi jendela utama tempat semua komponen ditampilkan.

```
// Create main frame
JFrame frame = new JFrame("Library Book List");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setSize(600, 400);
frame.setLayout(new BorderLayout());
```

- Setelah itu, siapkan data awal yang akan ditampilkan di dalam tabel.

```
// Initial data for the table
String[][] data = {
    {"1", "Naruto", "Masashi Kishimoto", "1999"},
    {"2", "One Piece", "Eiichiro Oda", "1997"},
    {"3", "Attack on Titan", "Hajime Isayama", "2009"},
    {"4", "Demon Slayer", "Koyoharu Gotouge", "2016"},
    {"5", "Jujutsu Kaisen", "Gege Akutami", "2018"}
};

String[] columns = {"Book ID", "Title", "Author", "Year"};
```

- Buat sebuah tabel dengan memanfaatkan DefaultTableModel dan JTable.

```
// Create table
DefaultTableModel model = new DefaultTableModel(data, columns);
JTable table = new JTable(model);
```

- Tambahkan scroll pane agar tabel bisa digulir jika jumlah datanya cukup banyak.

```
// Scroll pane so table can be scrolled
JScrollPane scrollPane = new JScrollPane(table);
frame.add(scrollPane, BorderLayout.CENTER);
```

- Selanjutnya, buat sebuah panel yang berfungsi sebagai wadah untuk field input data.

```
// Panel for book input
 JPanel panel = new JPanel();
panel.setLayout(new BoxLayout(panel, BoxLayout.Y_AXIS));

// Add padding to panel
panel.setBorder(new EmptyBorder(10, 10, 10, 10));
```

7. Di dalam panel tersebut, tambahkan beberapa komponen input berupa JTextField untuk mengisi ID, nama depan, nama belakang, dan usia.

```
JTextField idField = new JTextField(20);
JTextField titleField = new JTextField(20);
JTextField authorField = new JTextField(20);
JTextField yearField = new JTextField(20);
```

8. Atur komponen input tersebut agar tersusun secara vertikal, lengkap dengan label masing-masing.

```
panel.add(new JLabel("Book ID:"));
panel.add(idField);
panel.add(new JLabel("Title:"));
panel.add(titleField);
panel.add(new JLabel("Author:"));
panel.add(authorField);
panel.add(new JLabel("Year:"));
panel.add(yearField);
```

9. Buat sebuah tombol untuk menambahkan data baru ke tabel

```
// Add book button
JButton addButton = new JButton("Add Book");
addButton.setAlignmentX(Component.LEFT_ALIGNMENT); // align left
panel.add(Box.createRigidArea(new Dimension(0, 10))); // spacing between fields and button
panel.add(addButton);
frame.add(panel, BorderLayout.NORTH);
```

10. Lalu tambahkan action listener yang akan menangani prosesnya penambahan data

```
// Button listener
addButton.addActionListener(e -> {
    String id = idField.getText();
    String title = titleField.getText();
    String author = authorField.getText();
    String year = yearField.getText();

    if (id.isEmpty() || title.isEmpty() || author.isEmpty() || year.isEmpty()) {
        JOptionPane.showMessageDialog(frame,
            "All fields must be filled!",
            "Error",
            JOptionPane.ERROR_MESSAGE);
        return;
    }

    model.addRow(new Object[]{id, title, author, year});

    idField.setText("");
    titleField.setText("");
    authorField.setText("");
    yearField.setText("");
});
```

11. Terakhir, tambahkan kode untuk menampilkan frame

```
// Show frame
frame.setVisible(true);
```

12. Hasil akhirnya, tampilan aplikasi akan memperlihatkan field input di bagian atas dan tabel di bagian bawah.

Library Book List

Book ID	Title	Author	Year
1	Naruto	Masashi Kishimoto	1999
2	One Piece	Eiichiro Oda	1997
3	Attack on Titan	Hajime Isayama	2009
4	Demon Slayer	Koyoharu Gotouge	2016
5	Jujutsu Kaisen	Gege Akutami	2018

Add Book

TUGAS

Berdasarkan tema yang sudah kalian buat di spreadsheet [Tema Progran](#) di modul sebelumnya, buatlah sebuah GUI menggunakan **Java Swing** dengan tampilan yang sekreatif dan semenarik mungkin. Adapun ketentuan dari tugas ini :

1. Wajib menggunakan komponen dasar Swing :

- JLabel
- JTextField
- JButton

Jika ingin menggunakan komponen lain diperbolehkan (**opsional**) seperti JPasswordField, JComboBox, JRadioButton, JCheckBox, dll.

2. Tambahkan syarat bahwa semua input wajib diisi

Contohnya, jika terdapat field bernama usernameField yang bersifat wajib diisi, namun user menekan tombol Submit tanpa mengisinya, maka aplikasi harus menampilkan pesan peringatan menggunakan JOptionPane, misalnya: "Username belum terisi".

3. Aplikasi harus mendukung CRUD (Create, Read, Update, Delete)

- Create → Menambah data baru melalui form input
- Update → Mengubah data yang sudah ada
- Delete → Menghapus data tertentu

Khusus untuk operasi **Read**, hasil data yang tersimpan wajib ditampilkan dalam JTable, agar sesuai dengan **poin nomor 4 mengenai penyajian data**.

4. Menyajikan data dalam bentuk JTable

Sebagai contoh, jika tema aplikasi kamu adalah Manajemen Buku Perpustakaan, maka JTable dapat digunakan untuk menampilkan daftar buku yang tersedia, seperti:

- Judul Buku
- Penulis
- Tahun Terbit
- Kategori
- Ketersediaan

5. Aplikasi minimal memiliki 2 halaman

Contoh :

- Halaman Form Input Data
- Halaman Dashboard / Tabel Data



6. Menerapkan UI yang Menarik

Aplikasi harus memiliki tampilan yang rapi, modern, dan menarik, misalnya dengan :

- Memberi styling warna pada tombol (setBackground, setForeground)
- Menggunakan font yang lebih enak dibaca (setFont)
- Button dengan rounded border
- Header atau banner gambar (opsional)
- Layout yang tertata rapi dan konsisten
- Styling JTable (row height, warna header, border)

7. Aturan Penggunaan Comment pada Code

Komentar hanya boleh digunakan pada bagian yang benar-benar membutuhkan penjelasan tambahan, seperti logika yang kompleks atau fungsi utama yang memerlukan penjelasan. **Praktikan tidak diperbolehkan menuliskan komentar yang menjelaskan kode secara detail, per baris, atau langkah demi langkah.** Komentar yang sifatnya mendeskripsikan hal yang sudah jelas dari nama variabel atau fungsi juga tidak diperbolehkan. Jika ditemukan komentar yang berlebihan atau tidak sesuai ketentuan, maka nilai Pemahaman Materi otomatis E.

Notes:

Jika ada yang ketahuan menggunakan design GUI yang sama, hanya merubah warna, font, dan isi field maka nilainya nilai E . Jika belum bisa mengisi tema maka nilainya otomatis E



KRITERIA & DETAIL PENILAIAN

KRITERIA PENILAIAN	POIN
CODELAB 1	
Program sesuai dengan ketentuan	5%
Program tidak error	5%
CODELAB 2	
Program sesuai dengan ketentuan	5%
Program tidak error	5%
TUGAS	
Program sesuai dengan ketentuan	25%
User interface (UI)	15%
Program tidak error	10%
Pemahaman Materi	30%
TOTAL	100%