D3 Teknik Informatika (TI)
Jurusan Teknik Informatika
Politeknik Negeri Indramayu

Graphical User Interface (GUI)

Fachrul Pralienka Bani Muhamad, S.ST., M.Kom.
fachrul.pbm@polindra.ac.id

2025

Tujuan

Setelah materi ini disampaikan, mahasiswa diharapkan mampu:

- **Membuat** tampilan aplikasi menggunakan berbagai komponen GUI sesuai kebutuhan
- **Memanfaatkan** library GUI untuk menyajikan tampilan aplikasi yang modern

Outlines

1. Pengantar GUI
2. Korelasi GUI dengan PBO
3. Platform Desktop
4. Teknologi Java Desktop
5. Java Desktop GUI
6. Java Swing
7. Kategori Komponen Java Swing
8. Java Documentation
9. Thread (Main Thread & EDT)
10. Pembuatan Project
11. MigLayout
    a. Layout Constraint
    b. Row/Column Constraint
    c. Component Constraint

Pengantar GUI

- Antarmuka pengguna grafis, suatu sistem interaksi antara manusia (pengguna) dan komputer yang menggunakan **elemen visual**, misalnya ikon, jendela, tombol, menu, label, tabel, dan lainnya
- Sebelum GUI, aplikasi yang dikembangkan berjalan pada command line interface (cli) atau **console** (terminal)
- Aplikasi **modern** yang berjalan di Windows, MacOS, Android, iOS, Web Browser, menggunakan GUI

Korelasi GUI dengan PBO

- Melalui GUI, **objek** dan **logika bisnis** yang dirancang menggunakan paradigma berorientasi objek, dapat “ditunjukkan” secara konkret (nyata) serta dimanfaatkan atau digunakan oleh manusia (pengguna)
- GUI menjadi standar universal dalam teknologi antarmuka, meskipun teknologi di baliknya berbeda-beda bergantung pada **PLATFORM** dan **BAHASA PEMROGRAMAN** yang digunakan

Platform **Desktop**

Tren industri

- Secara umum industri software beralih ke teknologi web/mobile untuk jangkauan pengguna yang lebih luas, namun tidak berarti kebutuhan atas platform desktop tidak lagi ada

Kebutuhan industri

- Kebutuhan aplikasi desktop tidak hilang, melainkan berpindah ke segmen pasar yang lebih spesifik
- Aplikasi internal khusus (Enterprise) misalnya perbankan, manufaktur, dan logistik menggunakan aplikasi desktop, karena mendukung performa tinggi dan keamanan jaringan, serta dapat diintegrasikan dengan hardware spesifik
- Kinerja (performa) dan kendali penuh, menyediakan akses langsung ke sumber daya sistem operasi dengan performa yang maksimal (native)
- Bekerja penuh tanpa koneksi internet yang stabil atau offline

Platform **Desktop**

Pembelajaran fundamental

- Fondasi arsitektur 3-tier pada platform web/mobile cenderung lebih kompleks
- Konsep fundamental Threading cenderung hanya ada di balik abstraksi framework web
- Setiap objek pada konsep fundamental PBO dapat ditunjukkan secara nyata di layar platform desktop
- Pembelajaran menggunakan studi kasus platform desktop tidak tertinggal (usang), melainkan mendukung prinsip-prinsip fundamental PBO, arsitektur perangkat lunak, dan Threading yang terfokus dan efektif
- Fundamental pada studi kasus platform desktop menjadi fondasi (bekal) untuk beradaptasi ke framework di platform lainnya

Teknologi **Java** (Desktop)

Teknologi modern berubah setiap waktu, tetapi fundamental tidak

- Teknologi berkembang cepat, framework web/mobile populer misalnya Java Springboot, VueJS, ReactJS, Flutter, Swift, Jetpack Compose, mungkin selanjutnya ada teknologi yang lain lagi
- Java desktop memenuhi kebutuhan pembelajaran prinsip fundamental PBO secara eksplisit dan jelas

Konsep PBO cenderung "bias" pada kompleksitas teknologi modern

- Web modern, perlu mempelajari HTML, CSS, JavaScript, NodeJS, React untuk membuat satu tombol yang berfungsi dan terhubung ke backend (fokus terpecah ke tools web, bukan konsep PBO)

Teknologi **Java** (Desktop)

Mendukung kemampuan adaptasi dalam menyelesaikan masalah

- Mendukung kemampuan beradaptasi serta menyelesaikan masalah fundamental
- Jika fokusnya hanya pada framework tertentu, maka cenderung akan mengalami kesulitan ketika framework tersebut tidak lagi populer, atau dihadapkan pada masalah threading yang kompleks

Mendukung demonstrasi konsep fundamental PBO secara efektif

- Mengharuskan memahami PBO murni
- Mempelajari prinsip arsitektur yang kuat
- Membangun fondasi multithreading yang krusial
- Mendukung kesiapan adaptasi tren teknologi

Java Desktop GUI

Teknologi Java menyediakan beberapa pilihan pustaka **(library)** yang berisi kelas-kelas siap pakai untuk membangun GUI (desktop)

Abstract Window Toolkit

- Toolkit GUI pertama di Java
- Bersifat heavyweight (komponen asli OS)
- Tampilan bergantung pada platform

Swing

- Pengembangan dari AWT
- Bersifat lightweight (di-render penuh oleh Java)
- Tampilan konsisten (platform-independent)

JavaFX (Oracle)

- Evolusi Swing
- Performa tinggi (akselerasi hardware)
- Tampilan konsisten (platform-independent)
- Menggunakan CSS untuk styling dan FXML untuk desain layout

Standard Widget Toolkit (IBM)

- Pengembangan dari AWT
- Pendekatan native
- Performa tinggi (akselerasi hardware)
- Platform-dependent

Java Desktop GUI

Teknologi Java menyediakan beberapa pilihan pustaka **(library)** yang berisi kelas-kelas siap pakai untuk membangun GUI (desktop)

Abstract Window Toolkit

- Toolkit GUI pertama di Java
- Bersifat heavyweight (komponen asli OS)
- Tampilan bergantung pada platform

Swing

- Pengembangan dari AWT
- Bersifat lightweight (di-render penuh oleh Java)
- Tampilan konsisten (platform-independent)

JavaFX (Oracle)

- Evolusi Swing
- Performa tinggi (akselerasi hardware)
- Tampilan konsisten (platform-independent)
- Menggunakan CSS untuk styling dan FXML untuk desain layout

Standard Widget Toolkit (IBM)

- Pengembangan dari AWT
- Pendekatan native
- Performa tinggi (akselerasi hardware)
- Platform-dependent

Java Swing

Menyediakan lingkungan ekosistem stabil dan matang

- Standar industri software sejak 1996
- Banyak proyek aplikasi enterprise yang masih eksis (relevansi industri)
- Sumber belajar yang komprehensif dan banyak

Mendukung Penerapan PBO murni

- Struktur Swing mendorong untuk berpikir secara objektif (kesederhanaan konsep yang berfokus pada PBO inti)
- Netral terhadap teknologi tambahan, misalnya CSS dan FXML (mirip XML/HTML) pada JavaFX, yang cenderung mengalihkan fokus konsep PBO ke web design
- Fokus pada logika dan struktur aplikasi (styling hanya sebagai tambahan)

Java Swing

Mendukung konsep fundamental GUI

- Memberikan visibilitas yang lebih jelas terhadap tampilan UI (konsep Threading dasar) yang mejadi keterampilan penting pengembang aplikasi

Mendukung kemandirian platform (independent)

- AWT dan SWT tidak direkomendasikan mengingat kerumitan pada deployment lintas platform, Swing lebih praktis untuk dijalankan di berbagai platform

Sebagai dasar pemahaman JavaFX atau teknologi lain

- Kesamaan konsep pengembangan perangkat lunak Java Swing dengan JavaFX maupun framework lain dapat digunakan untuk mendukung pembelajaran, sehingga kurva pembelajaran menjadi lebih rendah

Kategori Komponen Java Swing

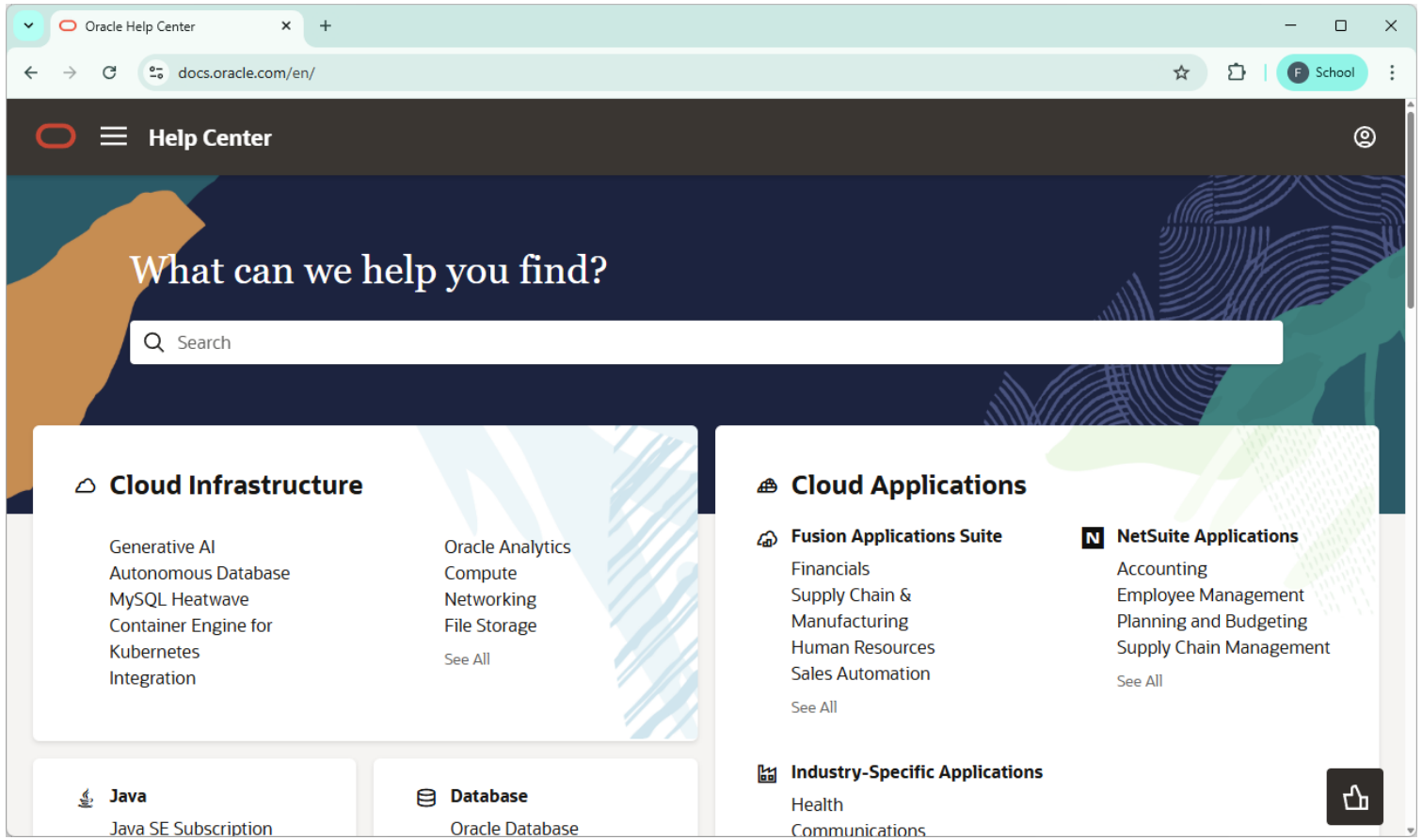
NO	KATEGORI	DESKRIPSI	KOMPONEN	PACKAGE
1	Structure & Layout	Fondasi UI; mendefinisikan jendela, area, dan aturan penempatan komponen	**Top-Level Containers:** JFrame, JDialog, JWindow	javax.swing
			Intermediate Containers: JPanel, JScrollPane, JSplitPane, JTabbedPane, JDesktopPane, JInternalFrame	javax.swing
			Layout Managers: BorderLayout, GridLayout, FlowLayout, GridBagLayout, BoxLayout, CardLayout	java.awt
			Layout Managers (3rd party): MigLayout, TableLayout	(eksternal)
2	Interactive Controls	Elemen yang memungkinkan masukan pengguna, seleksi, dan pemicu aksi, atau antarmuka utama untuk kendali pengguna.	JButton, JToggleButton JTextField, JPasswordField, JTextArea JCheckBox, JRadioButton JComboBox, JList JSlider, JSpinner JFileChooser, JColorChooser	javax.swing
			Utility: ButtonGroup	javax.swing

Kategori Komponen Java Swing

NO	KATEGORI	DESKRIPSI	KOMPONEN	PACKAGE
3	Output & Feedback	Komponen yang digunakan terutama untuk menampilkan informasi, memberikan pembaruan status, atau meningkatkan keterbacaan.	JLabel, JProgressBar, JSeparator JToolTip	javax.swing
			Rich Text Display: JEditorPane, JTextPane	javax.swing
			Dialog Boxes: JOptionPane	javax.swing
4	Navigation & Menu	Elemen terstruktur yang memfasilitasi pergerakan di dalam aplikasi dan akses ke fungsi inti.	JMenuBar, JMenu, JMenuItem JPopupMenu JToolBar	javax.swing
5	Complex Data Views	Komponen yang dirancang untuk memvisualisasikan struktur data yang besar atau hierarkis secara efektif.	**Tabular Data:** JTable	javax.swing
			Hierachical Data: JTree	javax.swing
6	Event Handling	Mekanisme untuk mendengarkan dan menanggapi interaksi pengguna (klik, tekan tombol, gerakan mouse).	ActionListener, MouseListener, KeyListener ActionEvent, MouseEvent, KeyEvent	java.awt.event

Java Documentation

https://docs.oracle.com/en/



Thread

- Thread artinya “benang” atau jalur eksekusi tunggal dalam sebuah program
- Suatu program (proses) dapat memiliki banyak thread (jalur eksekusi), sehingga dapat mengeksekusi beberapa hal sekaligus
- Java Swing memiliki 2 jalur eksekusi (thread) ideal, yaitu **Main Thread** dan **Event Dispatch Thread (EDT)**

Main Thread

- Konsep fundamental dalam eksekusi kode program
- Sebagai titik masuk atau titik awal (entry point) suatu program aplikasi
- Tanggung jawab utama:
  - Mengendalikan alur hidup program
  - Memulai thread lain
  - Melakukan inisialisasi awal
- Main thread **bukan** thread yang aman untuk melakukan pembaruan UI secara interaktif

```
1  public static void main(String[] args) {
2
3      // bagian yang dieksekusi oleh Main Thread
4
5  }
6
```

Event Dispatch Thread (EDT)

- Jalur eksekusi (thread) khusus yang menangani semua interaksi dan pembaruan UI
- Terdapat 2 fungsi utama EDT:
  - Memproses semua events ➔ menerima dan mengeksekusi semua kejadian yang berasal dari interaksi pengguna (klik mouse, ketik keyboard, resize ukuran jendela, dan lainnya)
  - Menggambar ulang UI ➔ bertanggung jawab untuk me-render (menggambar) semua komponen Swing di layar
- Semua kode yang berinteraksi langsung dengan komponen UI **perlu** dieksekusi di dalam EDT

```
1  SwingUtilities.invokeLater(new Runnable() {
2      @Override
3      public void run() {
4
5          // bagian yang dijadwalkan untuk dieksekusi oleh EDT
6
7      }
8  });
```

atau
(menggunakan lambda expression)

```
1  SwingUtilities.invokeLater(() -> {
2
3      // bagian yang dijadwalkan untuk dieksekusi oleh EDT
4
5  });
```

Komunikasi Main Thread & EDT

my-app - MyApp.java

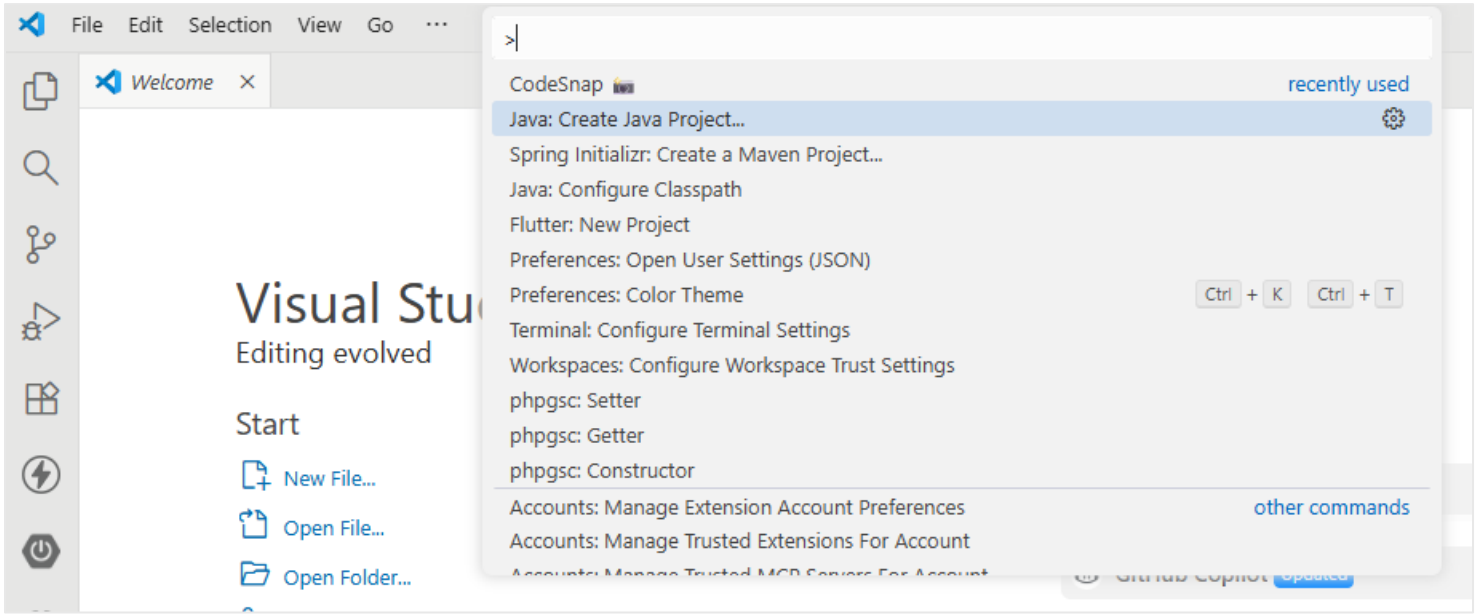
```
1  import javax.swing.SwingUtilities;
2
3  public class MyApp {
4
5      public static void main(String[] args) {
6
7          SwingUtilities.invokeLater(() -> {
8
9
10         });
11
12     }
13
14 }
15
```

dieksekusi oleh Main Thread

dijadwalkan untuk dieksekusi oleh EDT

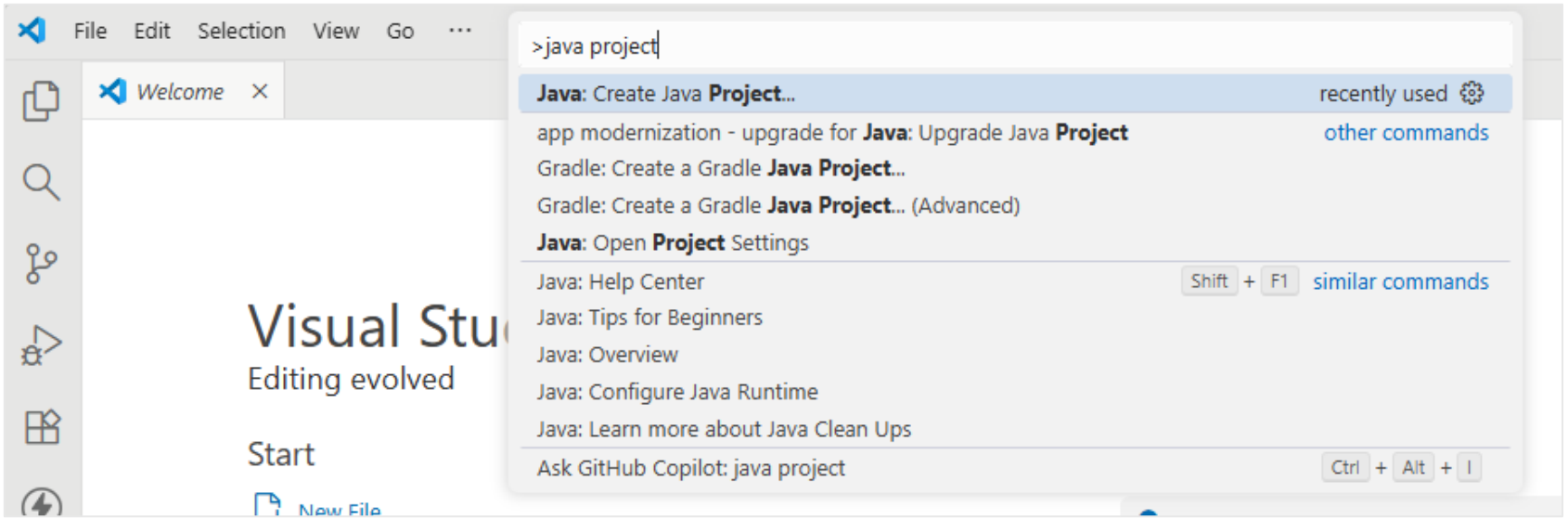
Langkah Pembuatan Project

- Buka VSCode
- Tekan tombol CTRL + Shift + P



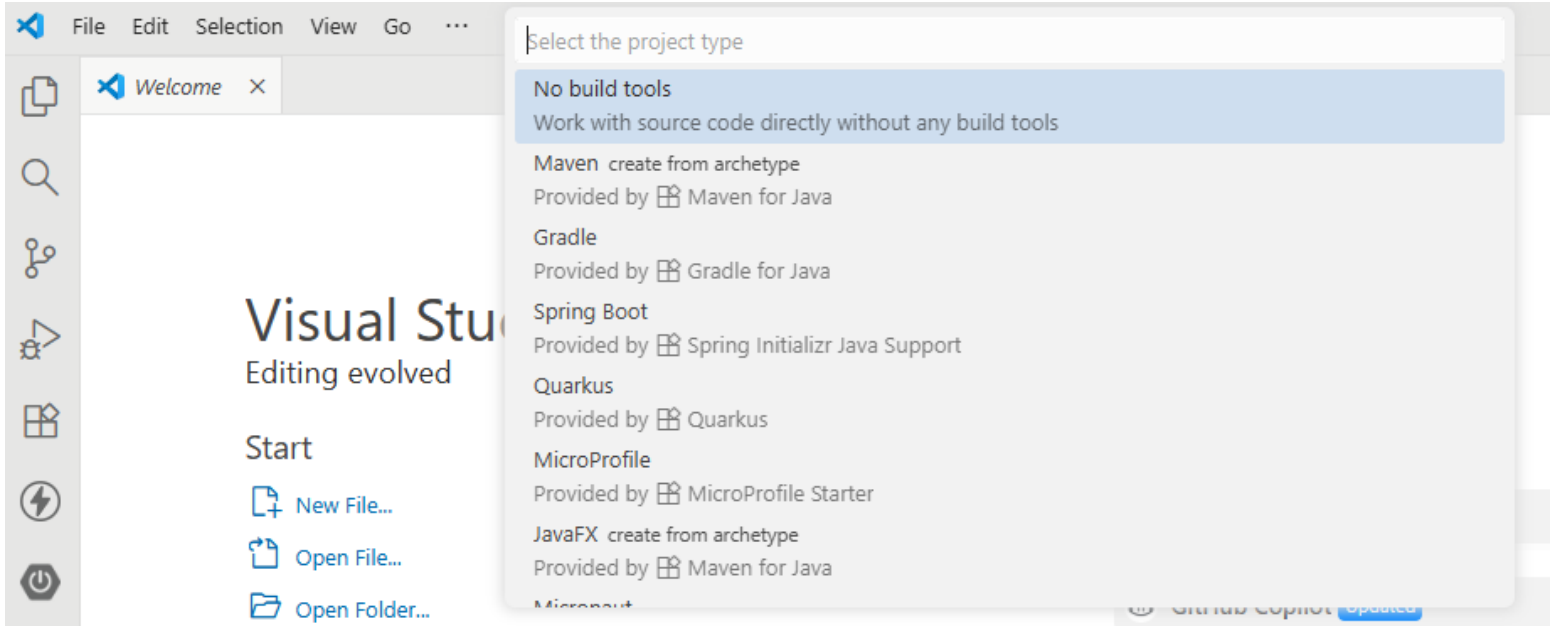
Langkah Pembuatan Project

- Java: Create Java Project...



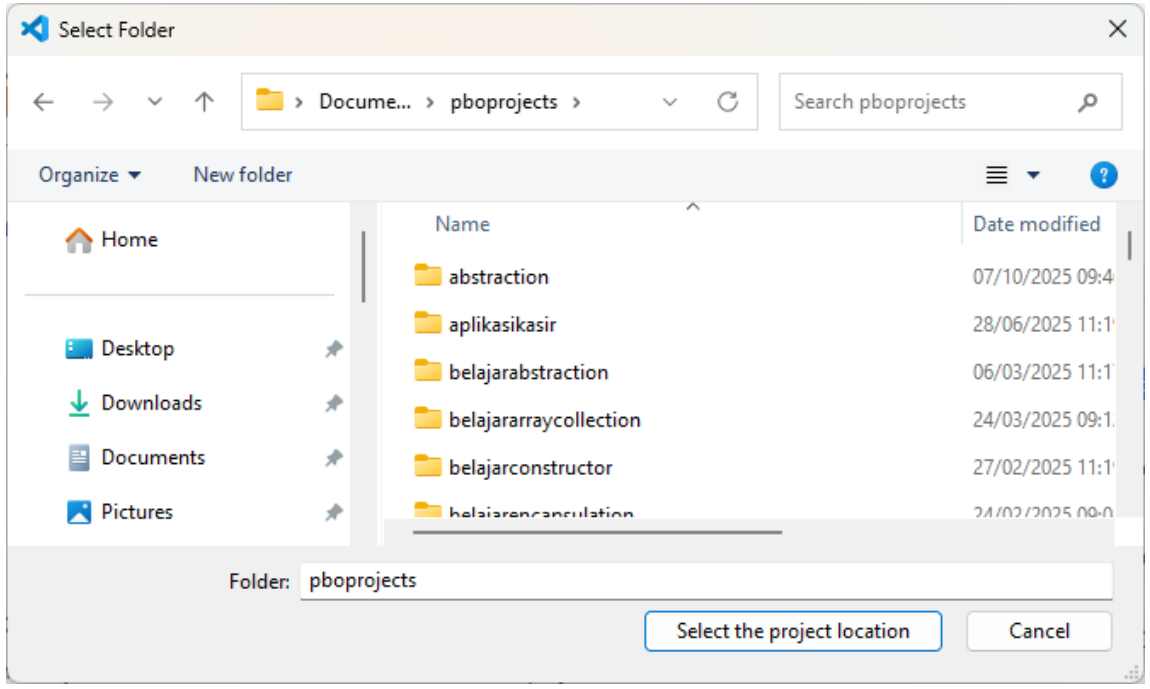
Langkah Pembuatan Project

- No build tools



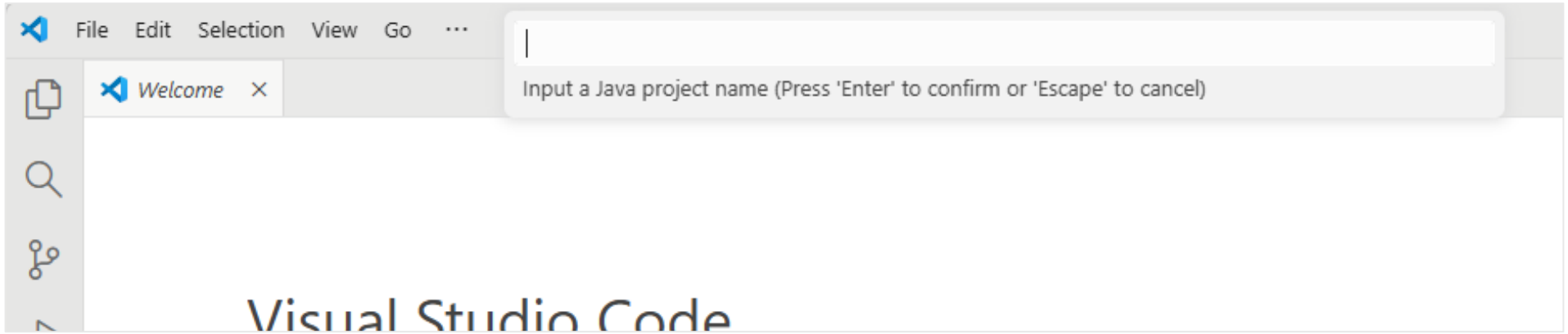
Langkah Pembuatan Project

- pboprojects



Langkah Pembuatan Project

- Beri nama proyek aplikasi java

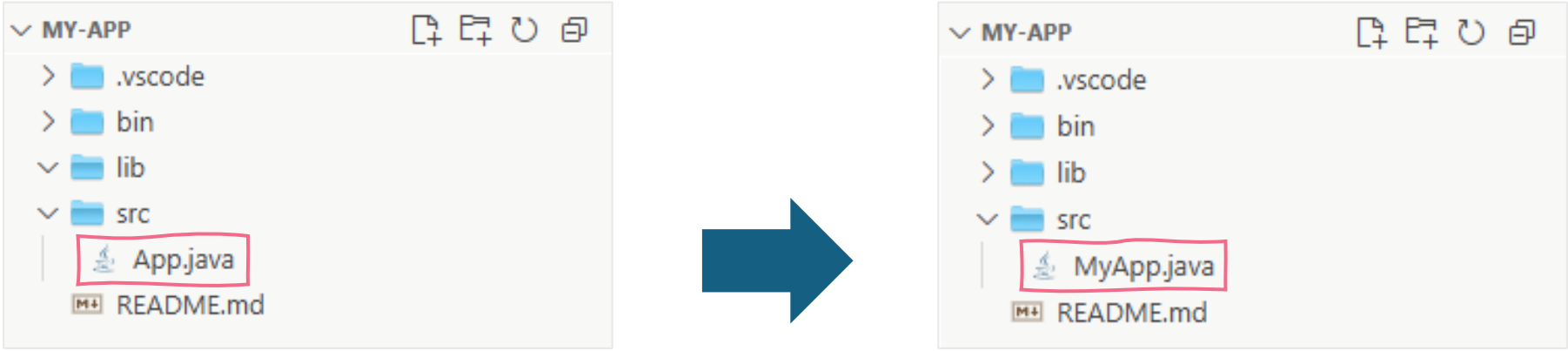


Langkah Pembuatan Project

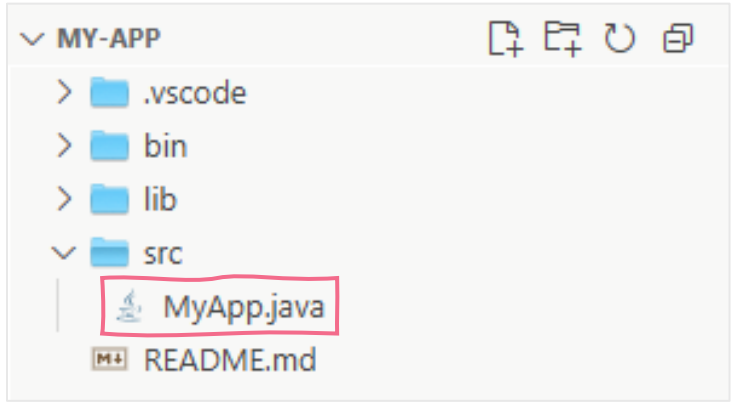
- Dengan konvensi penamaan:
  - gaya kebab-case
  - huruf kecil (non-kapital) semua
  - menggunakan tanda hubung -
- **Misalnya:** my-app

Langkah Pembuatan Project

- Sesuaikan nama class entry-point dengan nama project
- nama project: **my-app**
- nama file: **MyApp**



Langkah Pembuatan Project



my-app - MyApp.java

```java
import java.awt.Dimension;

import javax.swing.JFrame;
import javax.swing.SwingUtilities;

public class MyApp {

    public static void main(String[] args) {

        SwingUtilities.invokeLater(() -> {

            JFrame frame = new JFrame("Main Frame");
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            frame.setPreferredSize(new Dimension(1024, 768));
            frame.setMinimumSize(new Dimension(800, 600));
            frame.pack();
            frame.setLocationRelativeTo(null);
            frame.setVisible(true);

        });

    }

}
```

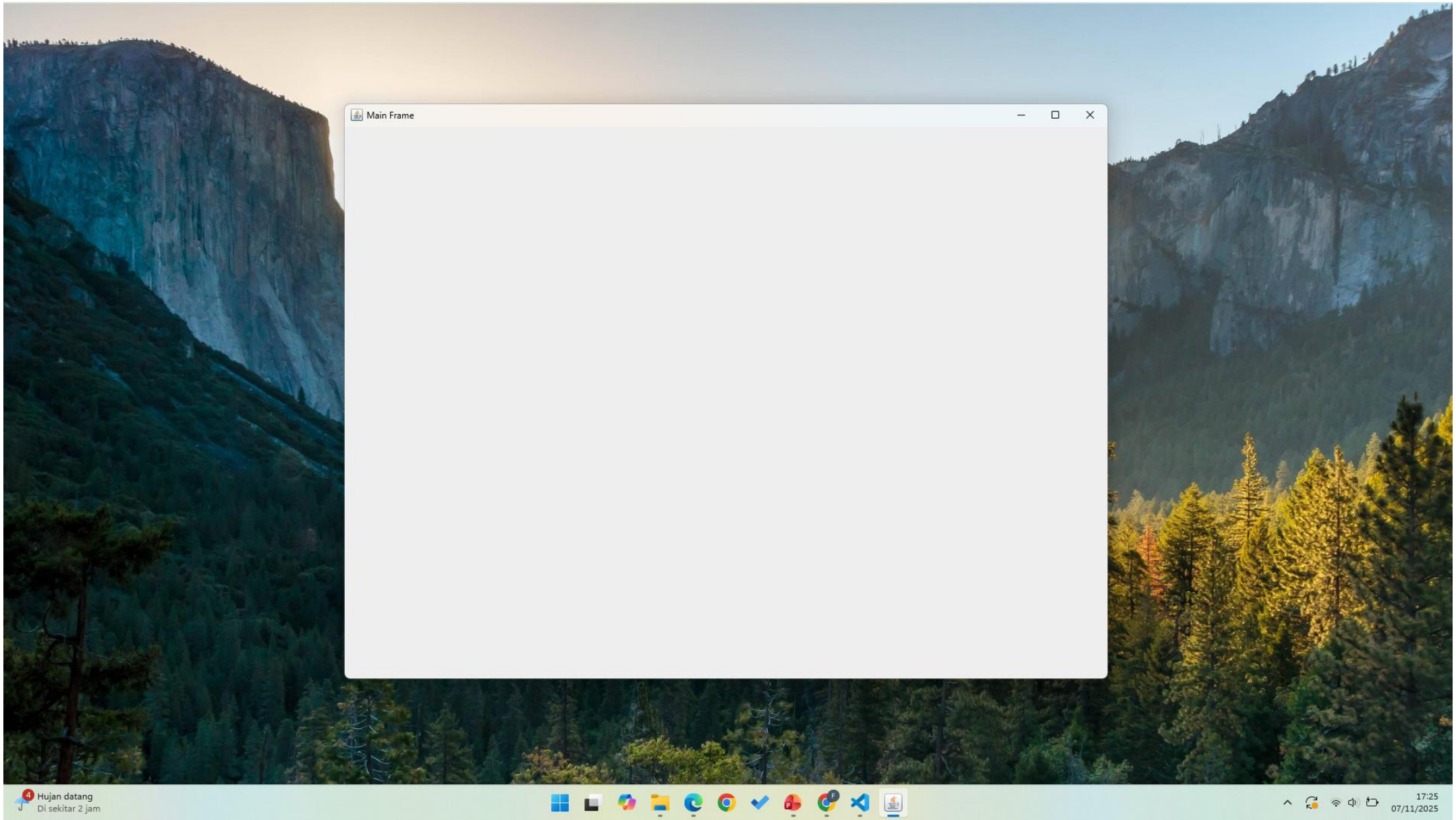
Langkah Running Project

- Compile & Run

```
fachr@fachrulpbm MINGW64 ~/Documents/pboprojects/my-app
$ javac -d bin src/*.java

fachr@fachrulpbm MINGW64 ~/Documents/pboprojects/my-app
$ java -cp bin MyApp
```

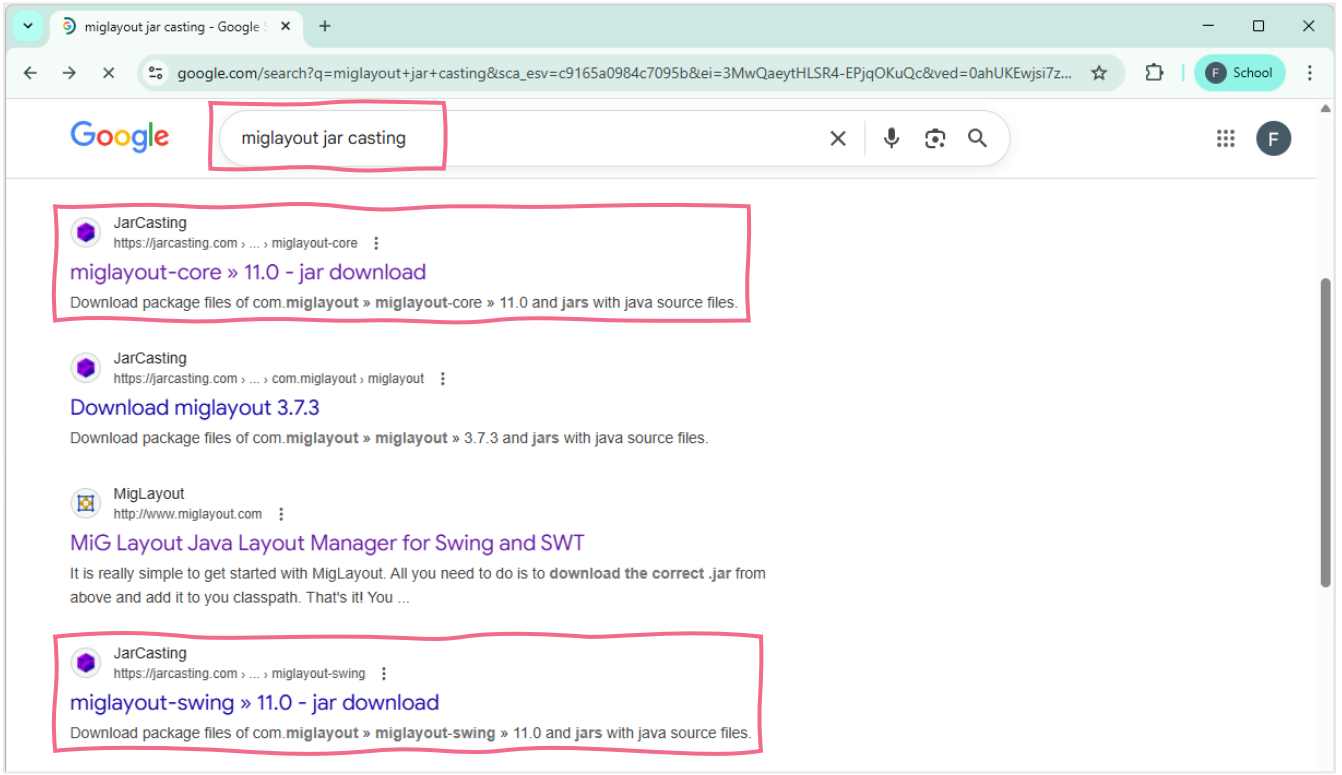
- Build & Run

```
fachr@fachrulpbm MINGW64 ~/Documents/pboprojects/my-app
$ jar cfe my-app.jar MyApp -C bin/ .

fachr@fachrulpbm MINGW64 ~/Documents/pboprojects/my-app
$ java -jar my-app.jar
```

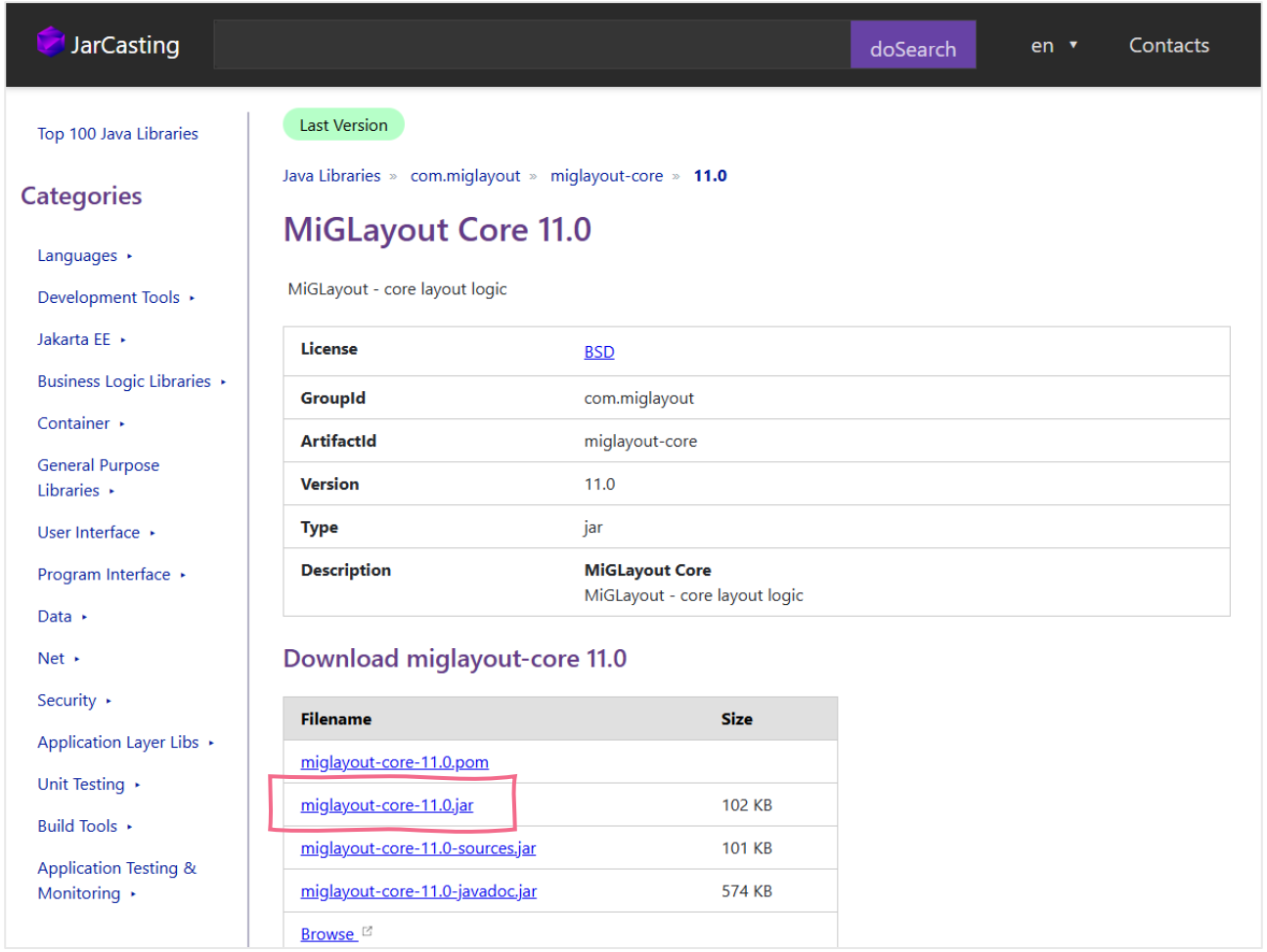

Main Frame
Hujan datang
Di sekitar 2 jam
17:25
07/11/2025

MigLayout

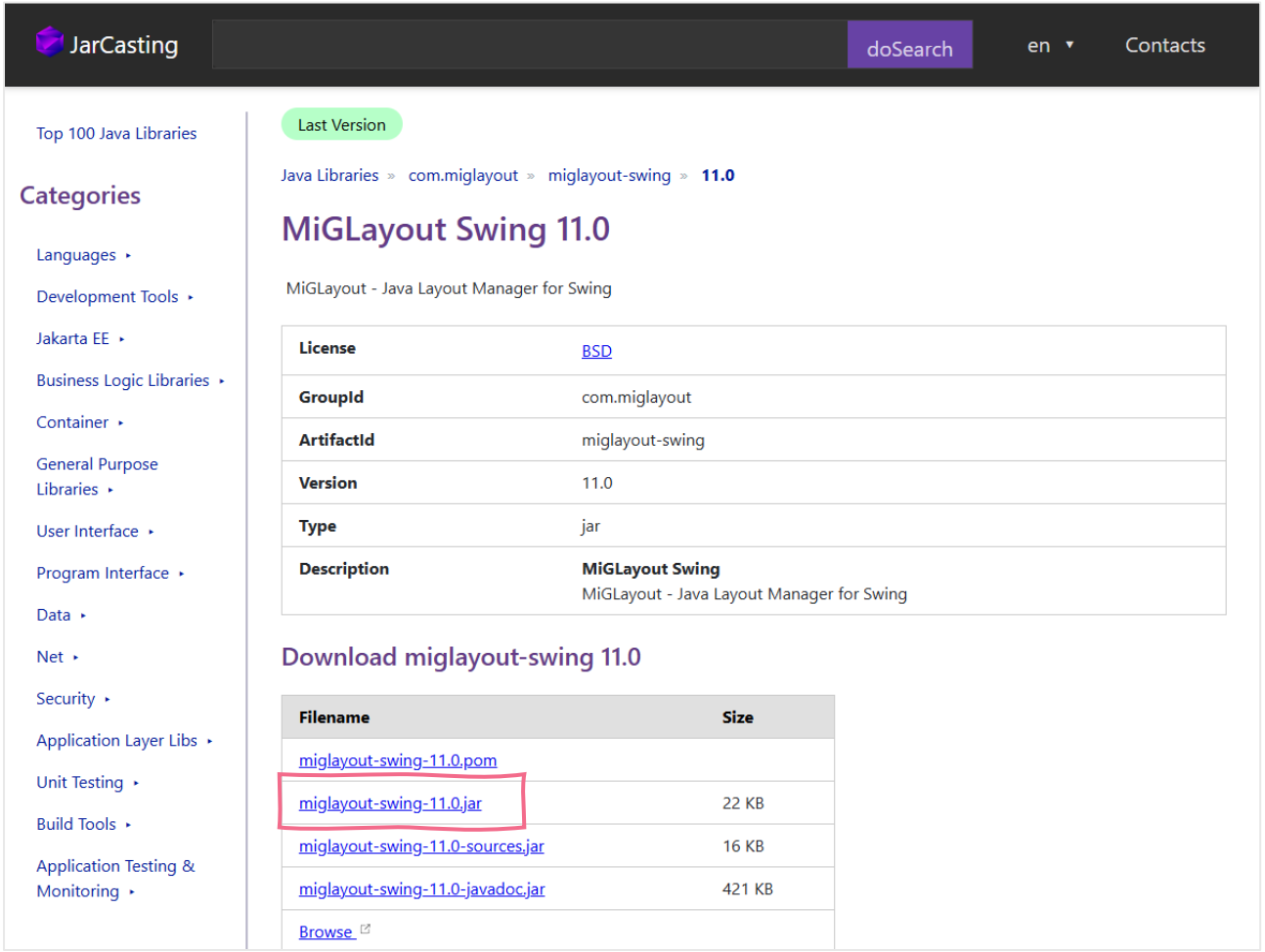
- Layout manager pihak ketiga yang populer untuk menyederhanakan penempatan komponen UI yang seringkali rumit dan kaku apabila menggunakan layout manager bawaan standar Java



MigLayout **Core** Download

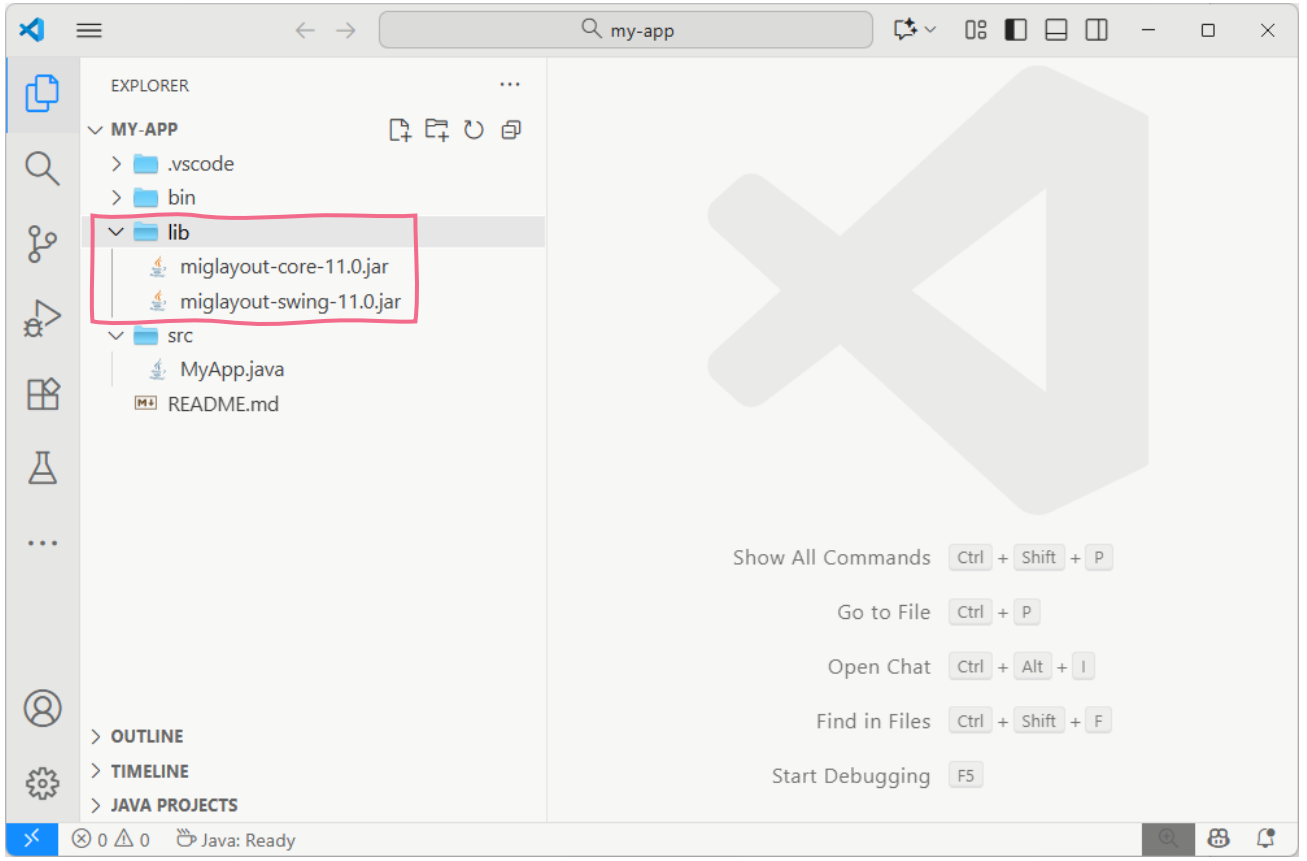


MigLayout **Swing** Download



Import MigLayout

- Pindahkan file *.jar yang telah di-download ke folder **lib** pada project
  - miglayout-**core**-11.0.jar
  - miglayout-**swing**-11.0.jar



Penggunaan MigLayout

- MigLayout bekerja berdasarkan grid dan aliran komponen melalui “string constraint” yang ringkas, yaitu
  - **Layout** constraints ➔ diterapkan pada intermediate container, misalnya JPanel
  - **Column/Row** constraints ➔ diterapkan pada container untuk mendefinisikan property kolom dan baris (lebar, tinggi, perataan)
  - **Component** constraints ➔ diterapkan pada setiap komponen individual saat ditambahkan ke container, untuk menentukan posisi dan ukurannya dalam grid

```
JPanel panel = new JPanel(
    new MigLayout(
        "", // Layout constraints
        "", // Column constraints
        "" // Row constraints
    )
);
panel.add(
    new JButton("Button 1"),
    "" // Component constraints
);
```

Layout Constraints

Properti	Deskripsi	Contoh Penggunaan
fill / fillx / filly	Mengisi ruang yang tersedia di kontainer (horizontal/vertikal/keduanya)	"fill"
debug	Menampilkan garis bantu grid berwarna untuk debugging	"debug, fillx"
insets	Menentukan padding/margin di dalam kontainer (top, left, bottom, right)	"insets 10 20 10 20“ atau "insets 10" (untuk semua sisi)
nogrid	Menonaktifkan mode grid, menggunakan mode aliran bebas (flow)	"nogrid"
align / center	Perataan global untuk seluruh grid dalam kontainer	"center"
hidemode	Menentukan bagaimana komponen yang tidak terlihat memengaruhi layout • **hidemode 0** (membiarkan ruang kosong). Komponen disembunyikan, tetapi ruang yang ditempatinya di grid tetap ada dan kosong. • **hidemode 1** (menyembunyikan komponen, tapi tidak menghapus ukuran). Komponen menjadi tidak terlihat, tetapi ukurannya masih dihitung oleh layout manager. • **hidemode 2** (menyembunyikan dan hapus ukuran, tapi tidak melibatkan margin/gap). Ruang yang ditempati komponen dilipat, tetapi gap (jarak antar sel) yang terhubung ke komponen tersebut masih terlihat. • **hidemode 3** (menyembunyikan dan lipat ruang sepenuhnya). Komponen sepenuhnya dihapus dari perhitungan tata letak, seolah-olah tidak pernah ditambahkan, termasuk gap dan margin-nya.	"hidemode 3"

Layout Constraints (Contoh)

my-app - ContohLayoutConstraint.java

```java
import java.awt.Dimension;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;

import net.miginfocom.swing.MigLayout;

public class ContohLayoutConstraint {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Layout Constraints (Contoh)");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setPreferredSize(new Dimension(500, 400));

        String layoutConstraints = "fill, debug, insets 20, hidemode 3";
        // Opsi pengganti:
        // String layoutConstraints = "fillx, debug";
        // String layoutConstraints = "center, wmax 300";
        // String layoutConstraints = "nogrid, debug, insets 10";

        JPanel panel = new JPanel(new MigLayout(layoutConstraints));

        JButton button1 = new JButton("Button 1");
        JButton button2 = new JButton("Button 2");
        JButton button3 = new JButton("Button 3");

        panel.add(button1, "w 100, h 50");
        panel.add(button2, "w 100, h 50, wrap");
        panel.add(button3, "w 100, h 50, span 2, growx, wrap");
        JLabel statusLabel = new JLabel("Button 2 terlihat");
        panel.add(statusLabel, "span 2, align center, gaptop 20");

        button1.addActionListener(e -> {
            boolean isVisible = button2.isVisible();
            button2.setVisible(!isVisible);
            if (isVisible) {
                statusLabel.setText("Button 2 Disembunyikan (Ruang Kosong Dilipat)");
            } else {
                statusLabel.setText("Button 2 Terlihat Kembali");
            }
        });

        frame.add(panel);
        frame.pack();
        frame.setLocationRelativeTo(null);
        frame.setVisible(true);
    }
}
```

Langkah Running Project (Update)

- Compile & Run

```
fachr@fachrulpbm MINGW64 ~/Documents/pboprojects/my-app
$ javac -cp "lib/*" -d bin src/*.java

fachr@fachrulpbm MINGW64 ~/Documents/pboprojects/my-app
$ java -cp "lib/*;bin" ContohLayoutConstraint
```

Langkah Running Project (Update)

- Build & Run

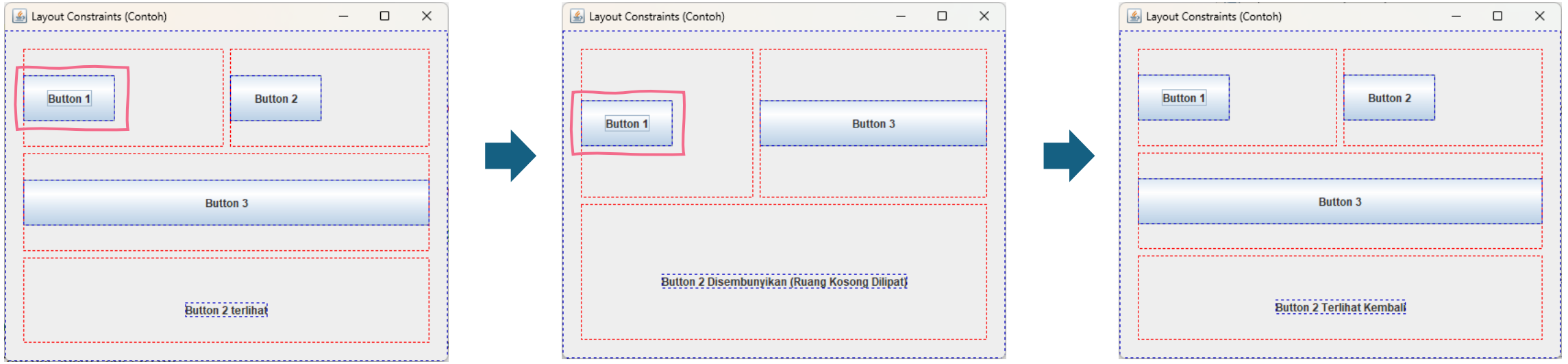
my-app - MANIFEST.MF

```
1  Manifest-Version: 1.0
2  Main-Class: ContohLayoutConstraint
3  Class-Path: lib/miglayout-core-11.0.jar lib/miglayout-swing-11.0.jar
4
```

← Baris terakhir MANIFEST.MF **perlu** dikosongkan

```
fachr@fachrulpbm MINGW64 ~/Documents/pboprojects/my-app
$ jar cvfm contoh-layout-constraint.jar MANIFEST.MF -C bin/ .
added manifest
adding: ContohLayoutConstraint.class(in = 2435) (out= 1321)(deflated 45%)
adding: MyApp.class(in = 1383) (out= 730)(deflated 47%)

fachr@fachrulpbm MINGW64 ~/Documents/pboprojects/my-app
$ java -jar contoh-layout-constraint.jar
```

Layout Constraints (Contoh)



A **B** **C**

Tampilan berubah **mengisi** ruang yang kosong (**hidemode 3**)

Column & Row Constraints

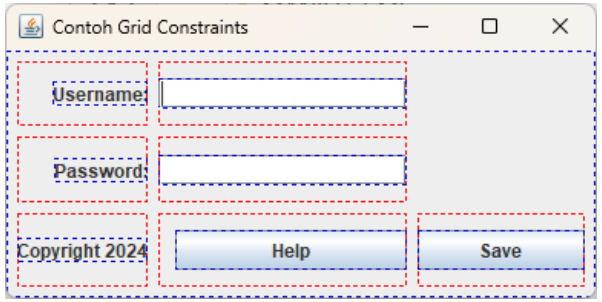
Properti	Deskripsi
[width] / [height]	Menentukan ukuran tetap atau rentang ukuran kolom/baris.
grow	Memungkinkan kolom/baris mengambil ruang ekstra saat jendela diperbesar.
shrink	Prioritas penyusutan kolom/baris saat jendela diperkecil.
align/pos	Perataan default untuk konten di dalam kolom/baris (right, center, top, bottom).
gap	Menentukan jarak setelah kolom/baris tersebut (20px, related, unrelated).
push	Memberikan prioritas tertinggi dalam mengambil sisa ruang yang tersedia.

Column & Row Constraints (Contoh)

my-app - ContohGridConstraint.java

```java
import java.awt.Dimension;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;

import net.miginfocom.swing.MigLayout;

public class ContohGridConstraint {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Contoh Grid Constraints");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setPreferredSize(new Dimension(400, 200));

        // Contoh 1: width, grow, align (studi kasus login form)
        String colConstraints = "[right][grow, fill]";
        String rowConstraints = ""; // Default

        // Contoh 2: shrink, gap, push (layout footer)
        // String colConstraints = "[shrink][][grow, push, align right]";
        // String rowConstraints = "[height 50]"; // Contoh height/width di row/col constraint

        JPanel panel = new JPanel(new MigLayout("debug, fill", colConstraints, rowConstraints));

        // Komponen untuk Contoh 1
        panel.add(new JLabel("Username:")); // Kolom 0
        panel.add(new JTextField(10), "wrap"); // Kolom 1 (grow/fill)

        panel.add(new JLabel("Password:")); // Kolom 0
        panel.add(new JPasswordField(10), "wrap"); // Kolom 1 (grow/fill)

        // resize window untuk melihat efek grow/fill

        // Komponen untuk Contoh 2 (Footer style)
        panel.add(new JLabel("Copyright 2024"), ""); // shrink by default
        panel.add(new JButton("Help"), "gap unrelated"); // gap
        panel.add(new JButton("Save"), ""); // grow, push, align right

        frame.add(panel);
        frame.pack();
        frame.setLocationRelativeTo(null);
        frame.setVisible(true);
    }
}
```

Column & Row Constraints (Contoh)



Component Constraints

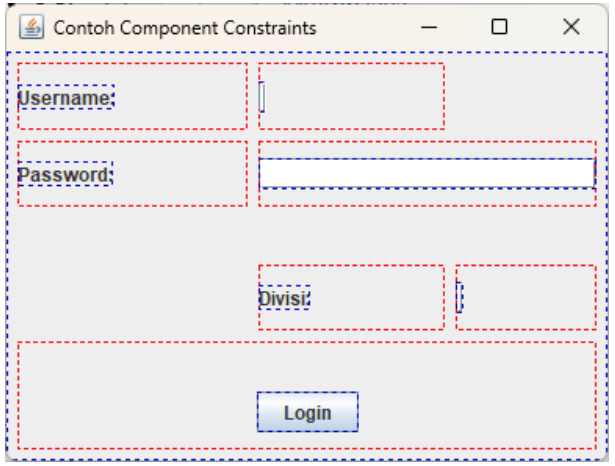
Properti	Deskripsi
wrap	Komponen berikutnya akan dimulai di baris baru
span N	Komponen merentang N kolom (merge)
span N M	Komponen merentang N kolom dan M baris (merge)
align / pos	Perataan komponen di dalam selnya (right, center, top, bottom)
skip N	Melewati N jumlah sel kosong sebelum menempatkan komponen
skip N M	Melewati N jumlah sel kosong dan M jumlah sel kosong sebelum menempatkan komponen
grow / push	Mengizinkan komponen tumbuh di dalam selnya atau mendorong layout
width / height	Mengatur ukuran spesifik komponen (min:pref:max)
gap	Menambahkan jarak sebelum/sesudah komponen (gaptop, gapleft, dll)
newline	Secara eksplisit memulai baris baru
wmax / hmax	Batas lebar/tinggi maksimum untuk komponen tersebut

Component Constraints (Contoh)

my-app - ContohComponentConstraint.java

```java
import java.awt.Dimension;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;

import net.miginfocom.swing.MigLayout;

public class ContohComponentConstraint {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Contoh Component Constraints");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setPreferredSize(new Dimension(400, 300));

        JPanel panel = new JPanel(new MigLayout("debug, fill"));

        panel.add(new JLabel("Username:"));

        // wrap: pindah baris setelah ini
        panel.add(new JTextField(), "wrap");

        panel.add(new JLabel("Password:"));
        // span 2: rentang 2 kolom penuh
        // growx: hanya tumbuh horizontal
        // wmax 250: maksimum lebar 250px
        panel.add(new JPasswordField(), "span 2, growx, wmax 250, wrap");

        // skip: lewati 1 sel (kolom pertama kosong)
        // newline: mulai baris baru (sama dengan wrap di akhir baris sebelumnya)
        panel.add(new JLabel("Divisi:"), "skip 1, newline");
        panel.add(new JTextField(), "wrap");

        // align center/pos 50% 50%
        panel.add(new JButton("Login"), "span 3, align center, gaptop 20");

        frame.add(panel);
        frame.pack();
        frame.setLocationRelativeTo(null);
        frame.setVisible(true);
    }
}
```

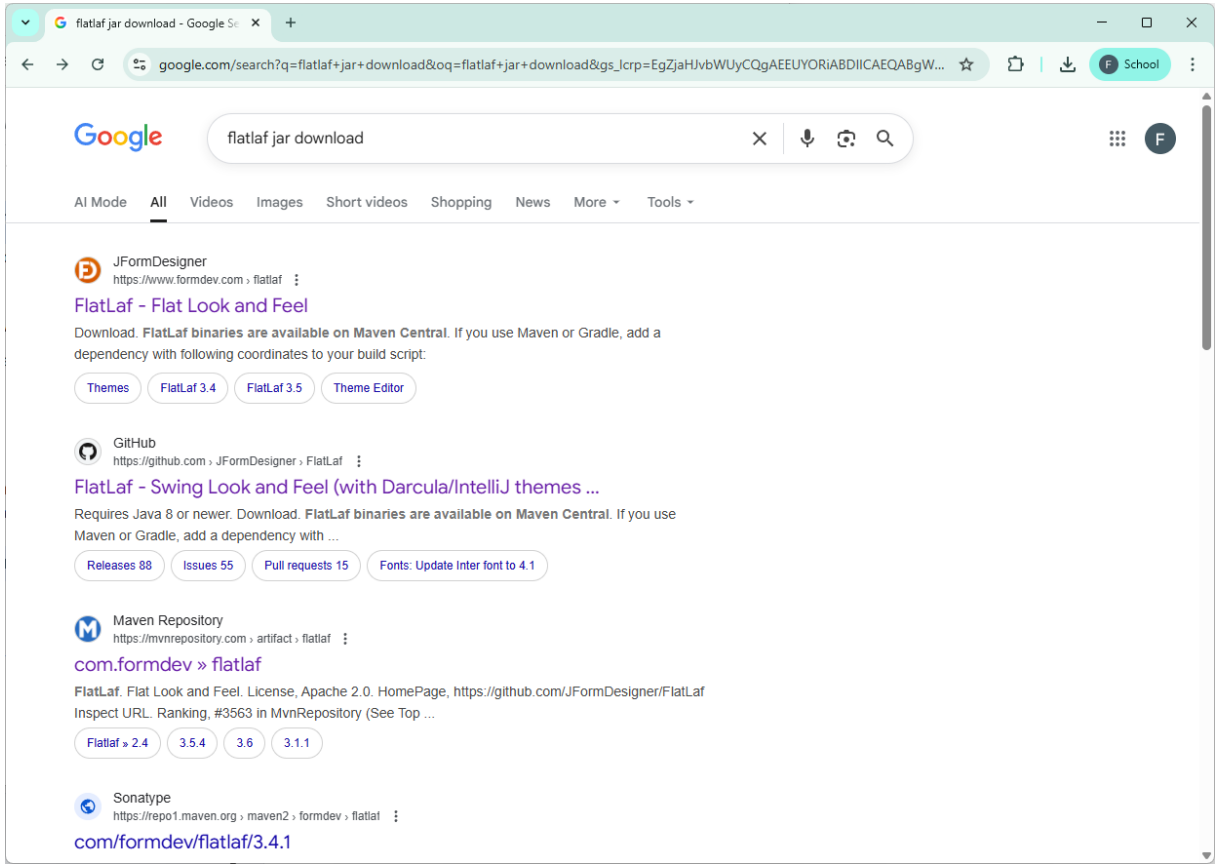
Component Constraints (Contoh)



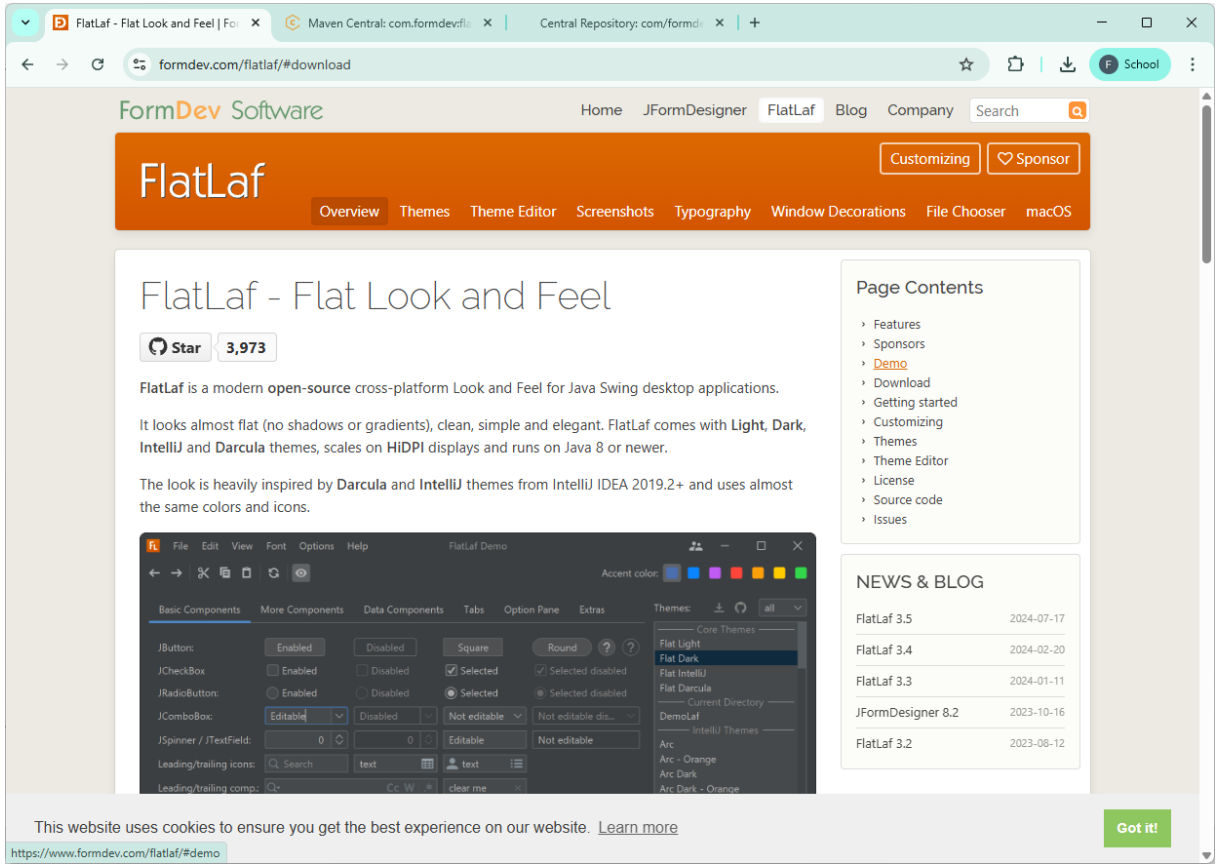
FlatLaF

- FlatLaf adalah Look and Feel (L&F) open source yang modern dan populer untuk aplikasi desktop Java Swing
- FlatLaf dirancang untuk memberikan tampilan yang bersih, minimalis (gaya "datar" atau flat design) serta elegan
- Secara default, UI Swing cenderung dinilai kurang menarik dengan tampilan Metal atau tampilan khas OS Windows/macOS versi lama
- FlatLaf berfungsi sebagai pengganti yang membuat aplikasi Swing terlihat modern, mendukung HiDPI (layar 4K/Retina), serta menyediakan tema terang dan gelap

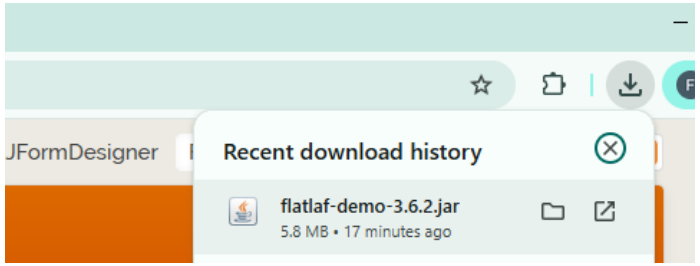
FlatLaF Demo



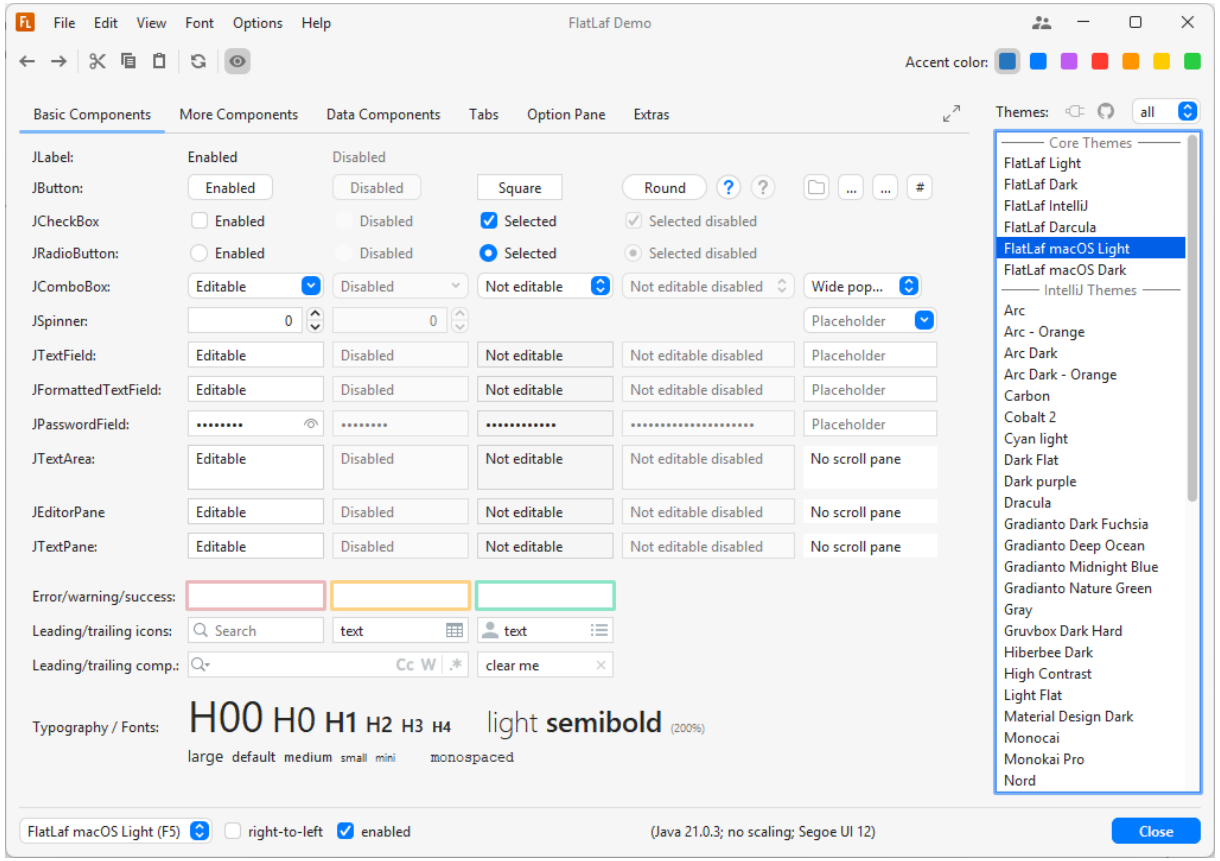
FlatLaF Demo



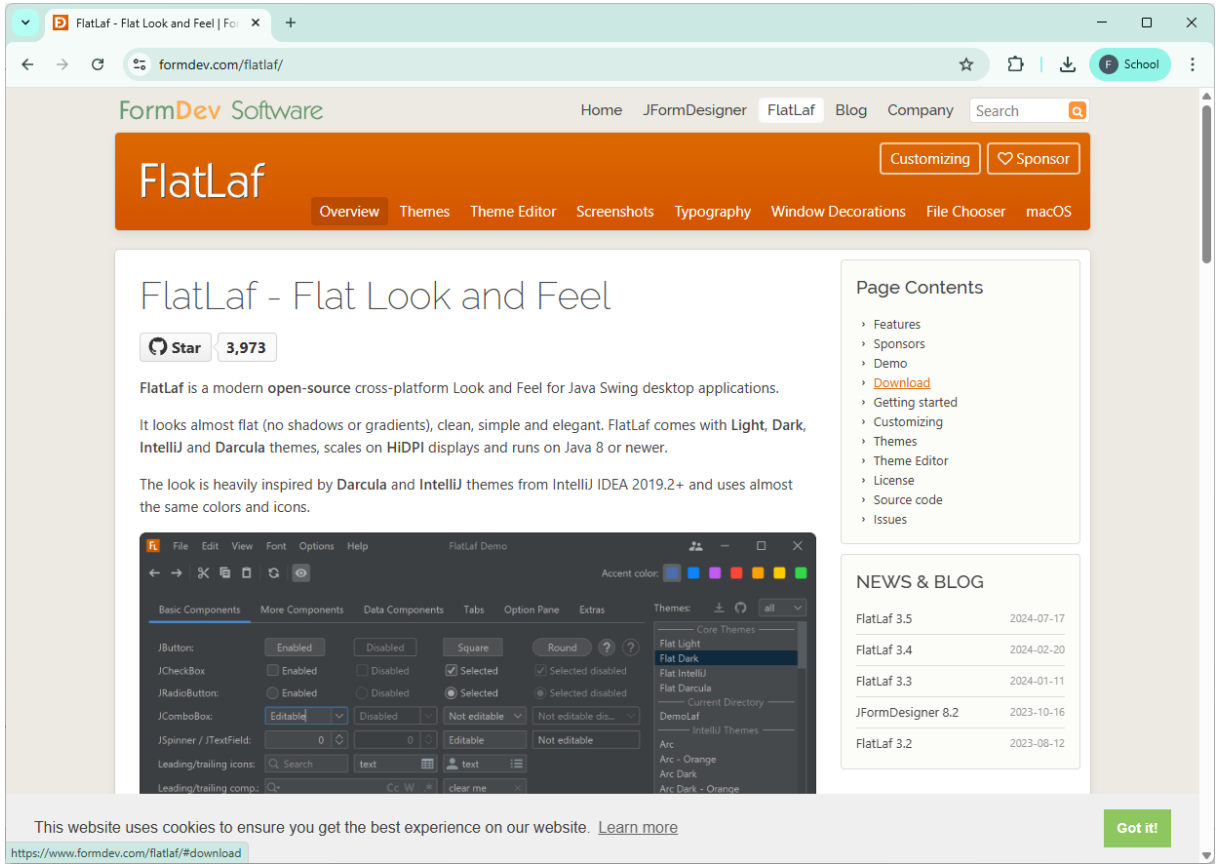
FlatLaF Demo



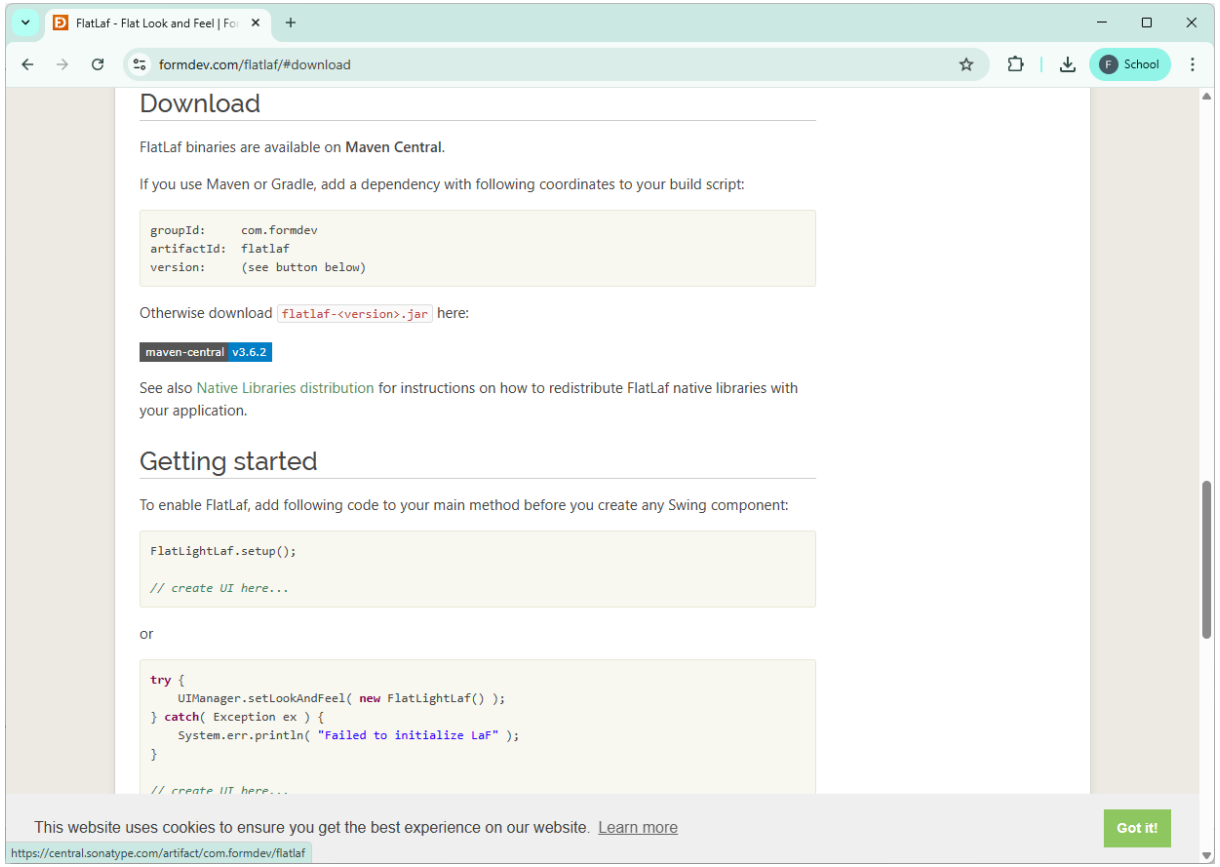
FlatLaF Demo



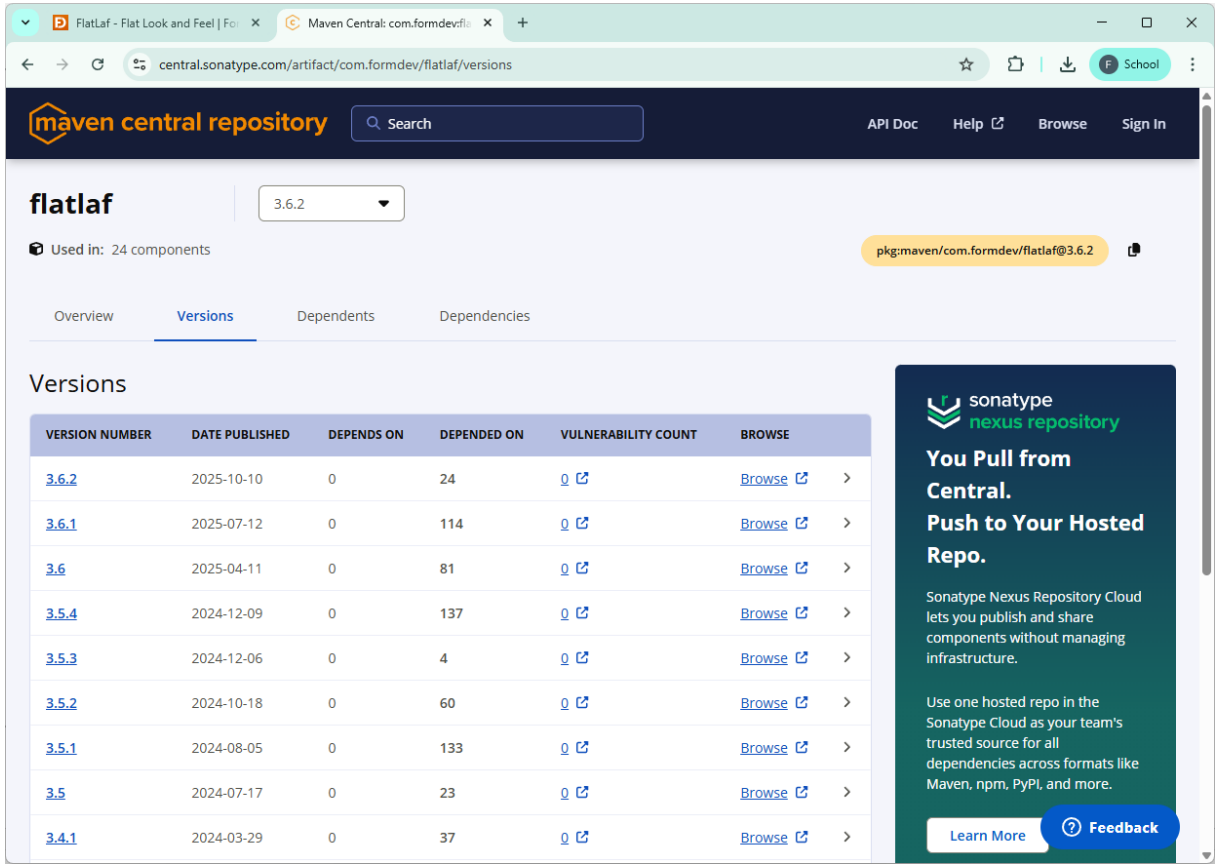
FlatLaF Download



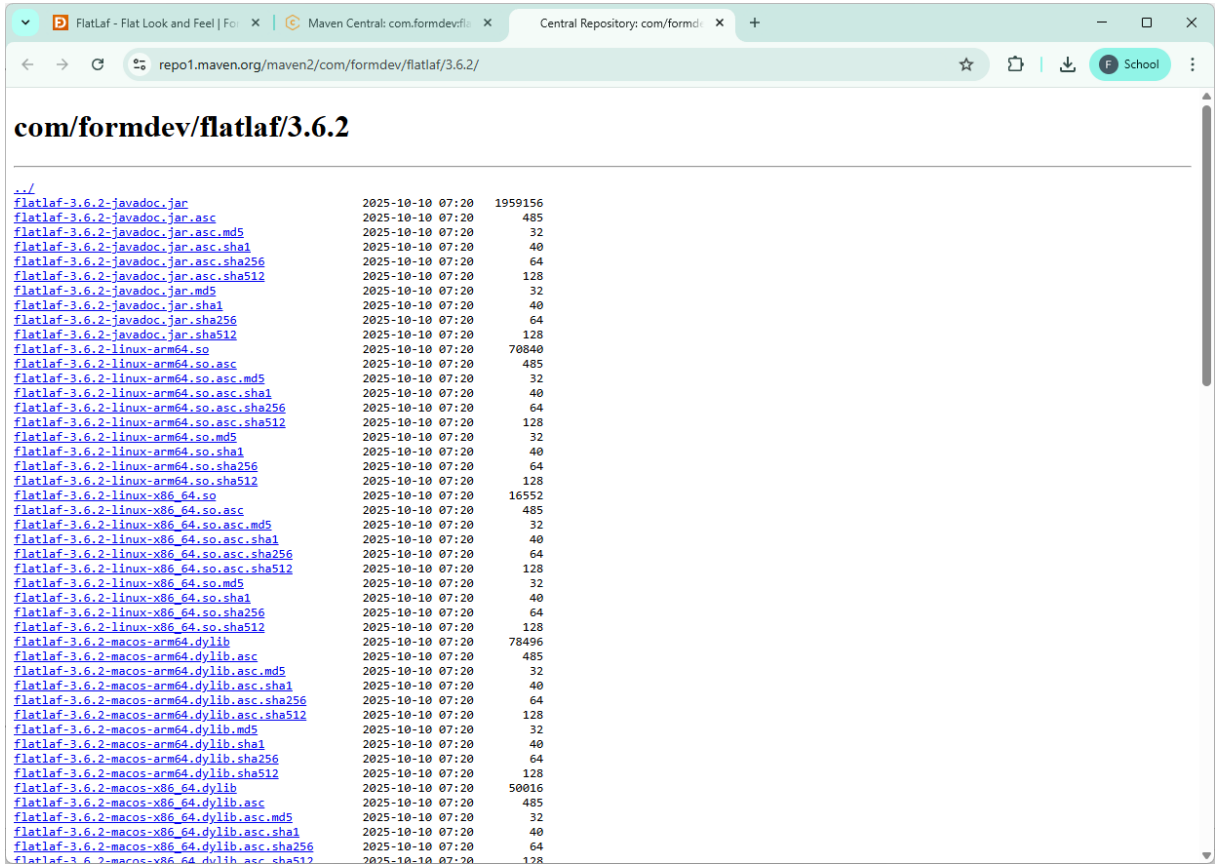
FlatLaF Download



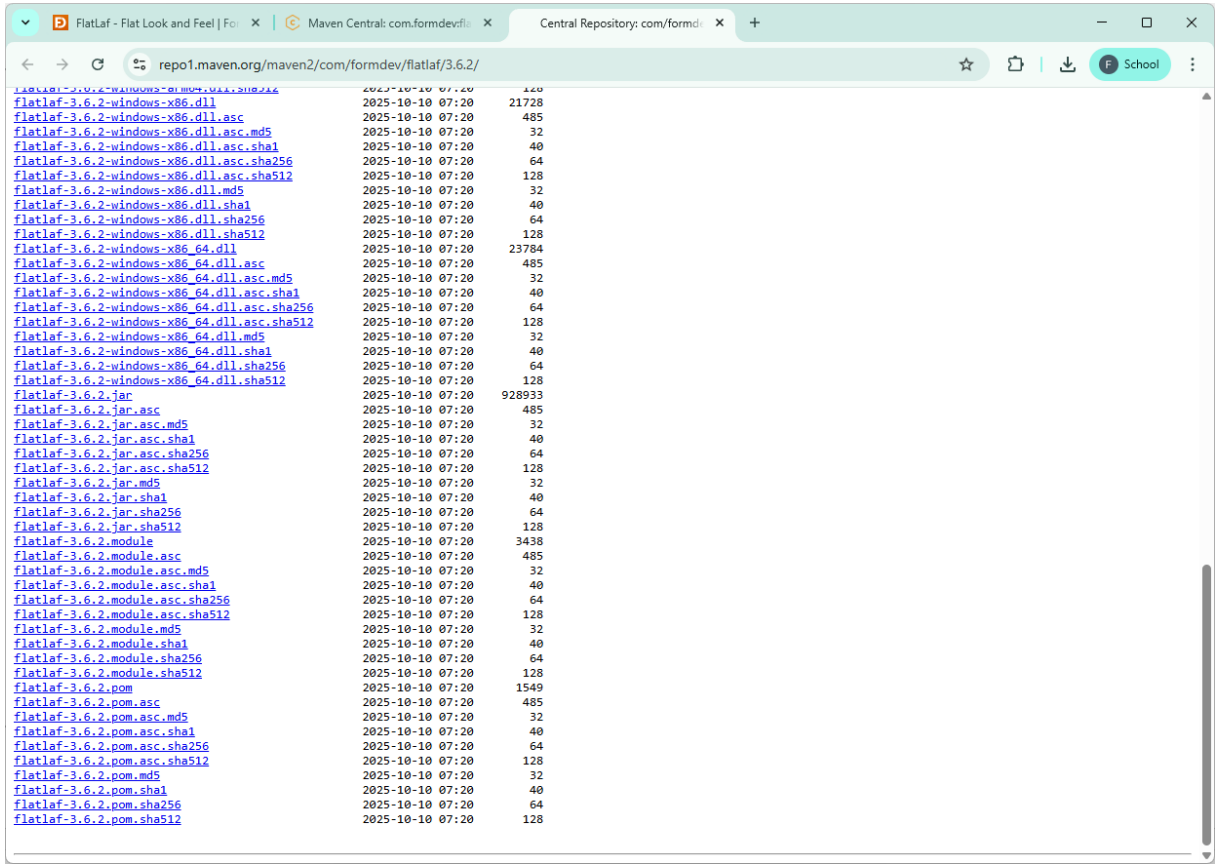
FlatLaF Download



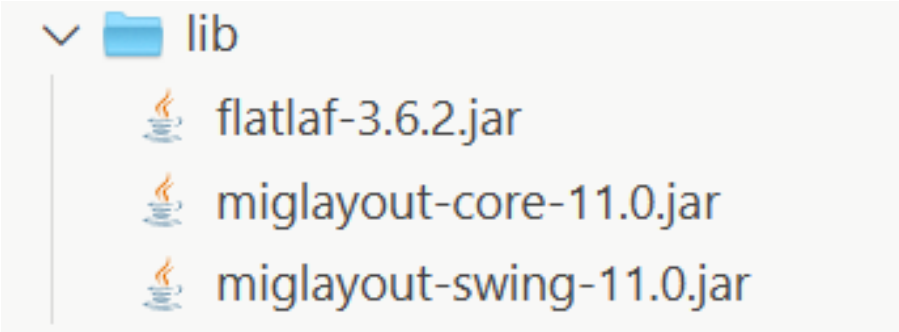
FlatLaF Download



FlatLaF Download



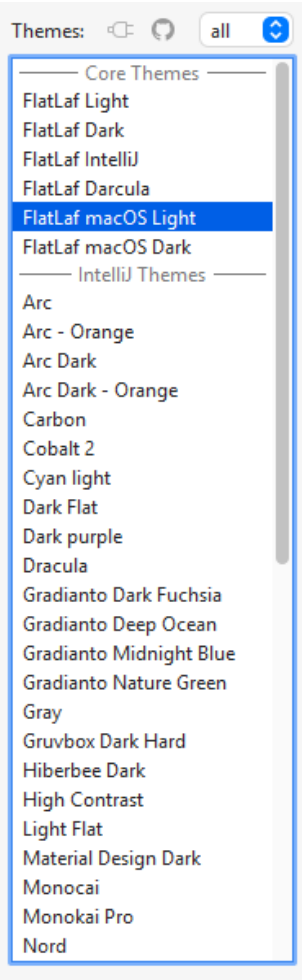
Import FlatLaF



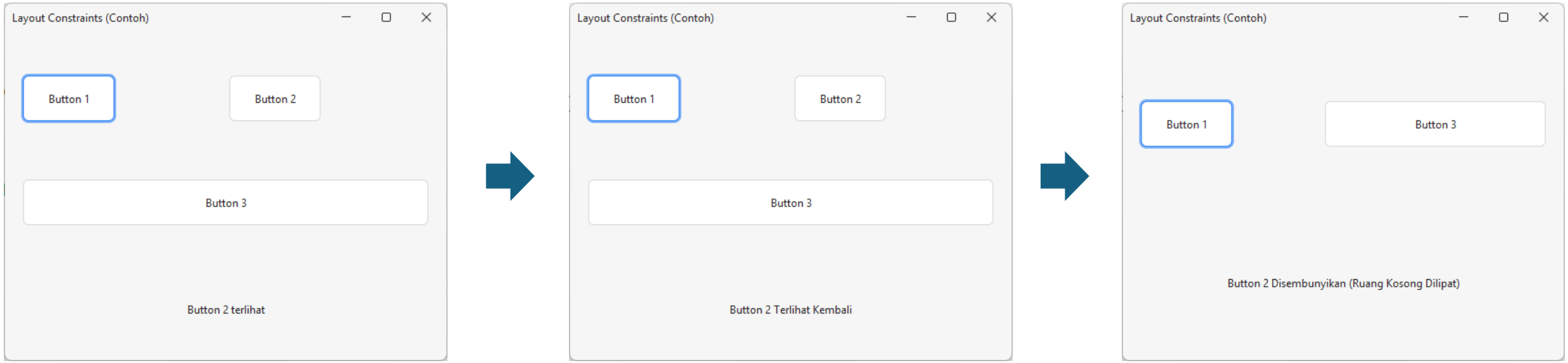
Penggunaan FlatLaF

my-app - MyApp.java

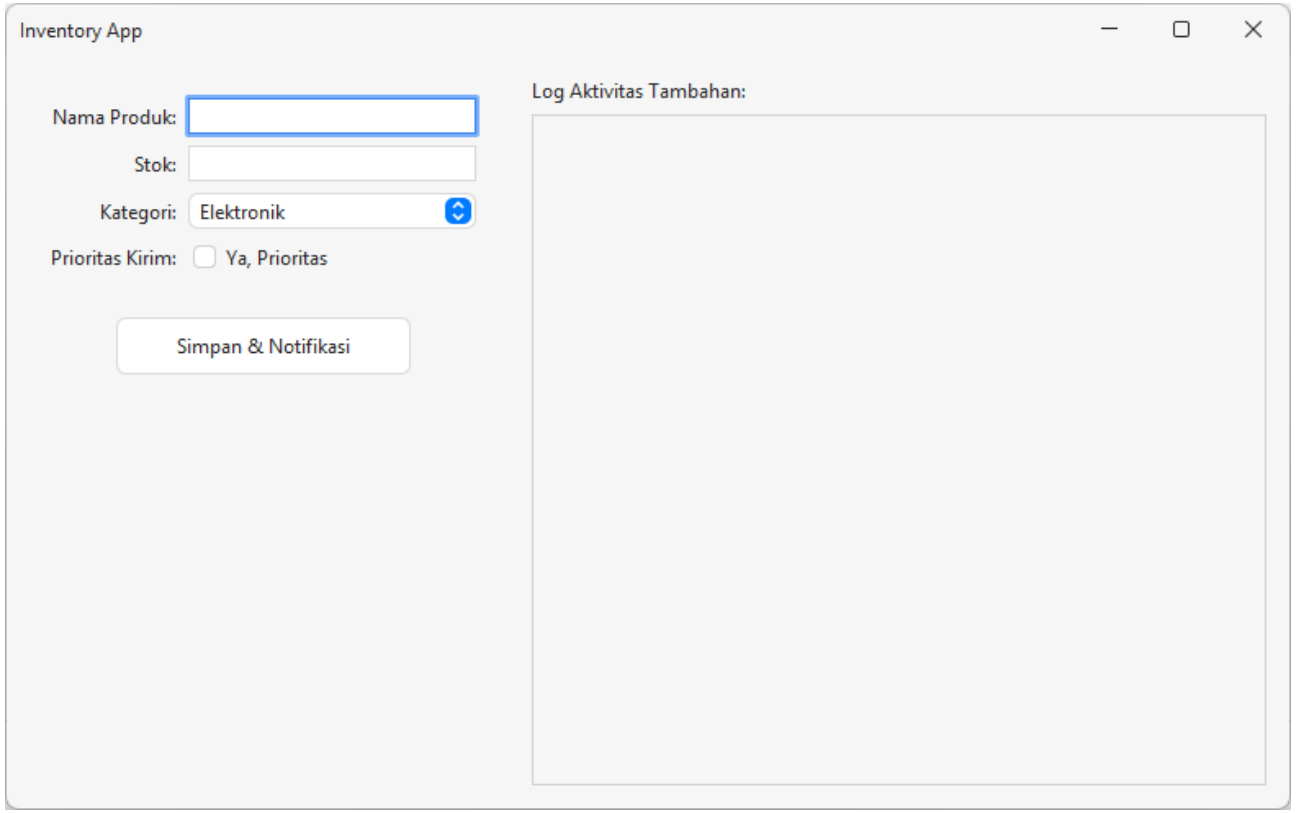
```java
import javax.swing.SwingUtilities;
import javax.swing.UIManager;
import javax.swing.UnsupportedLookAndFeelException;

import com.formdev.flatlaf.themes.FlatMacLightLaf;

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

        try {
            UIManager.setLookAndFeel(new FlatMacLightLaf());
        } catch (UnsupportedLookAndFeelException e) {
            System.err.println("Gagal mengatur Look and Feel: " + e.getMessage());
        }

        SwingUtilities.invokeLater(() -> {



        });

    }
}
```



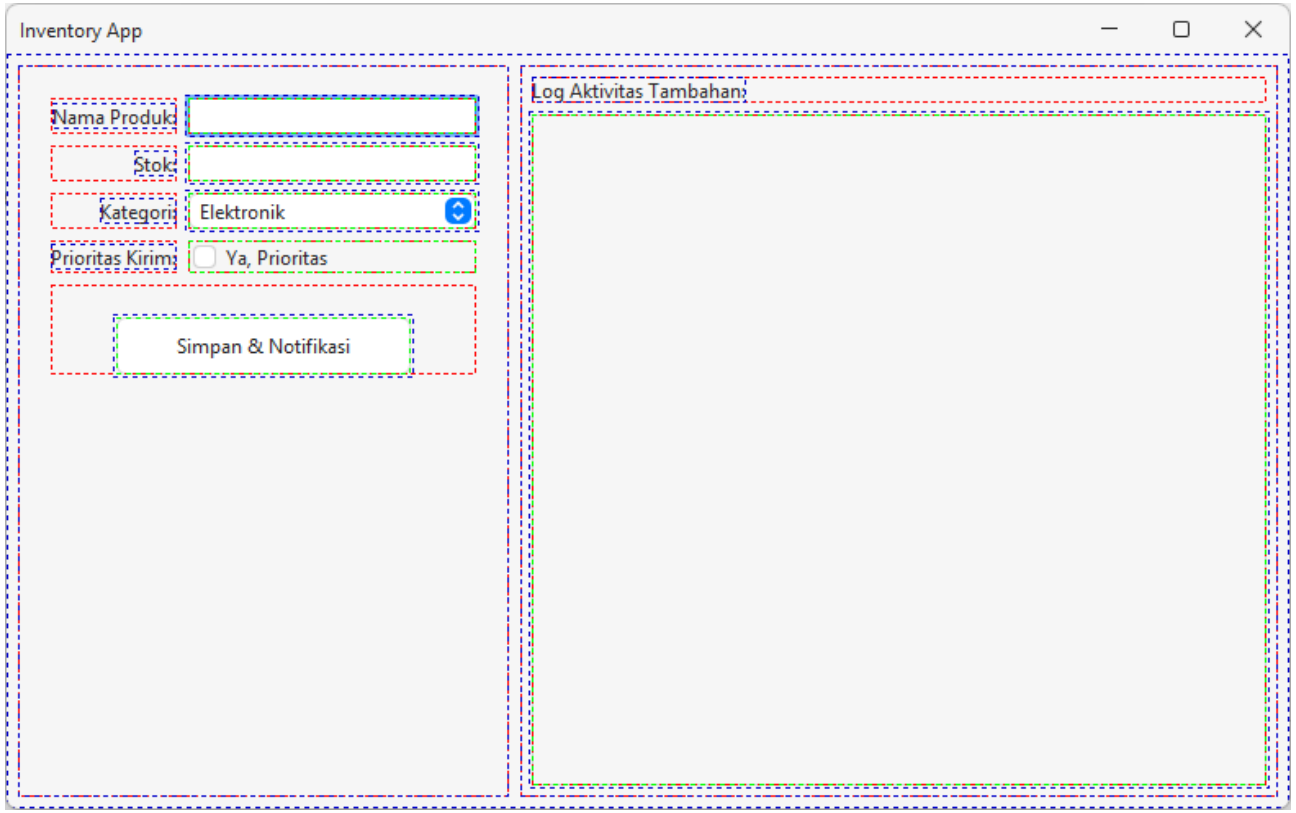
Penggunaan FlatLaF



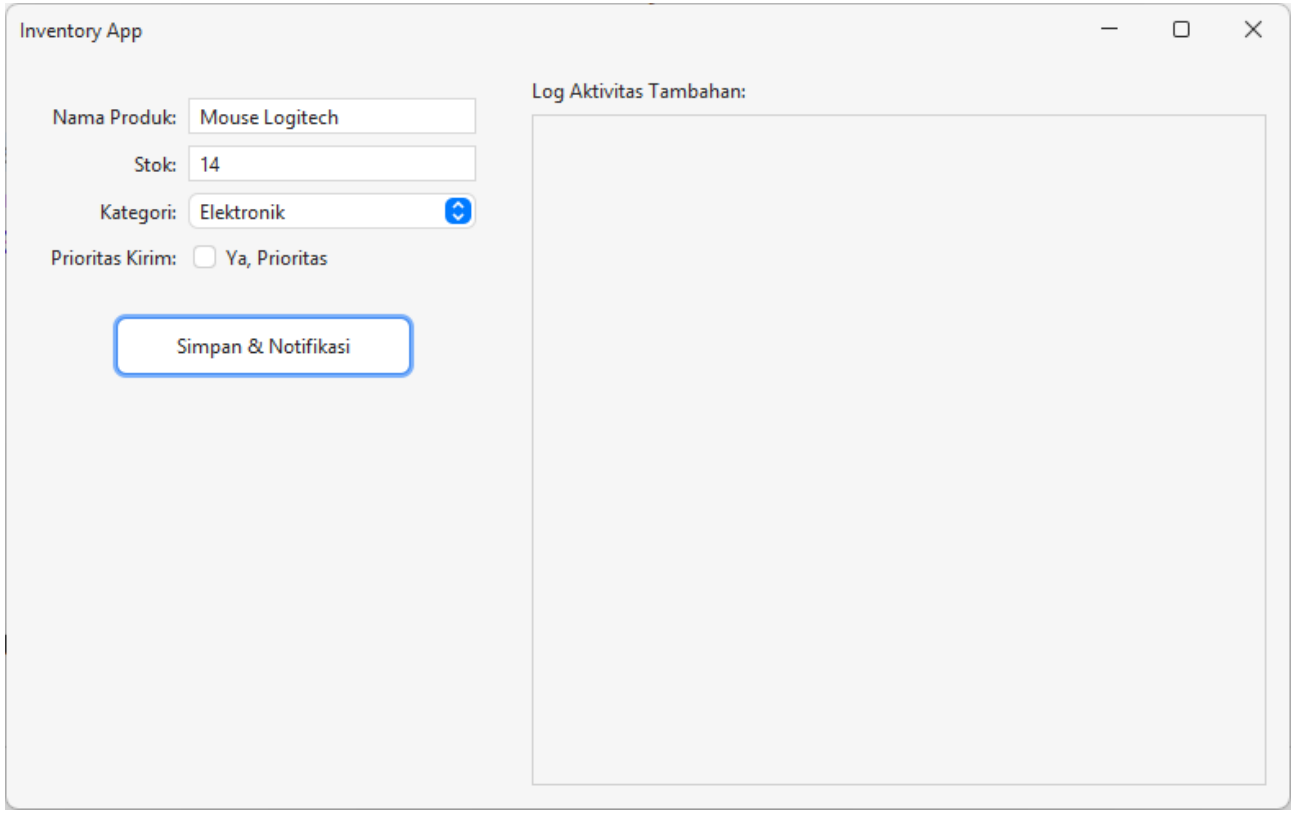
Analisis Komponen Swing + MigLayout



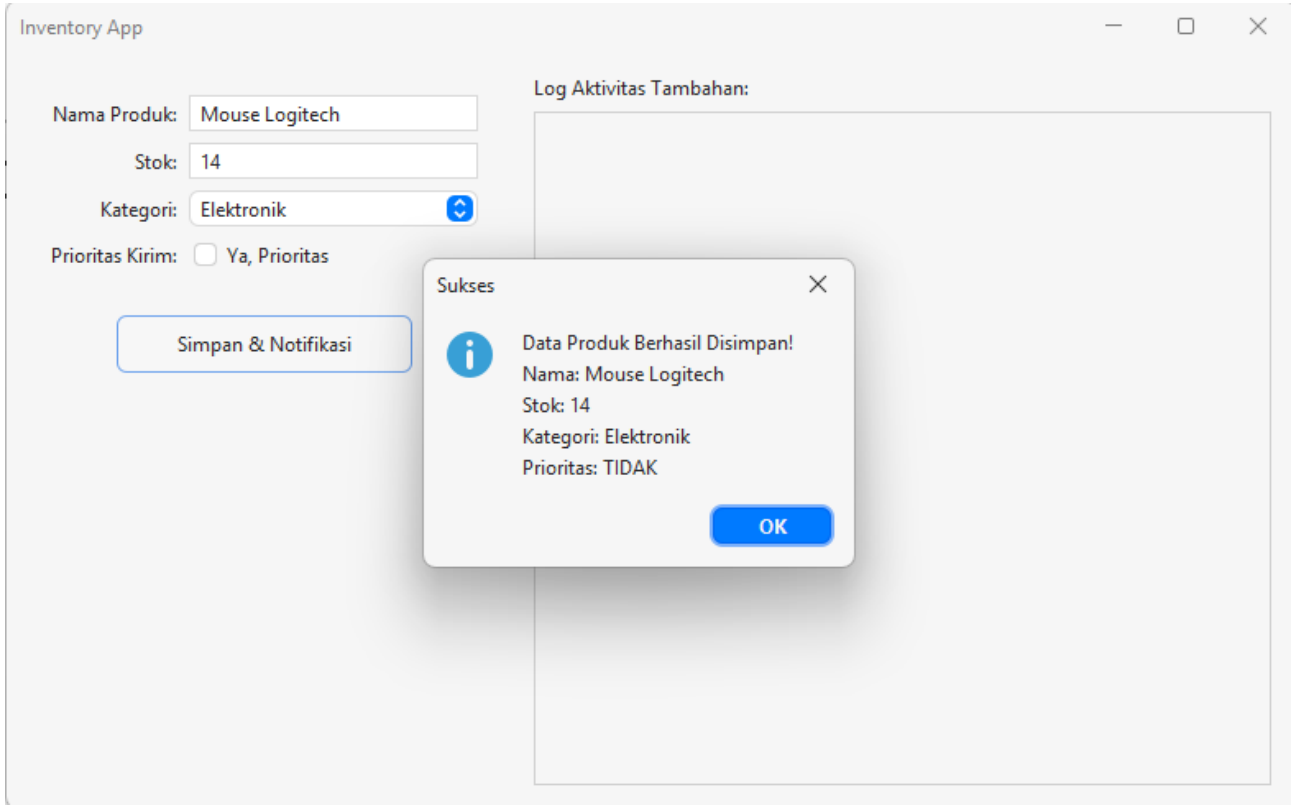
Analisis Komponen Swing + MigLayout



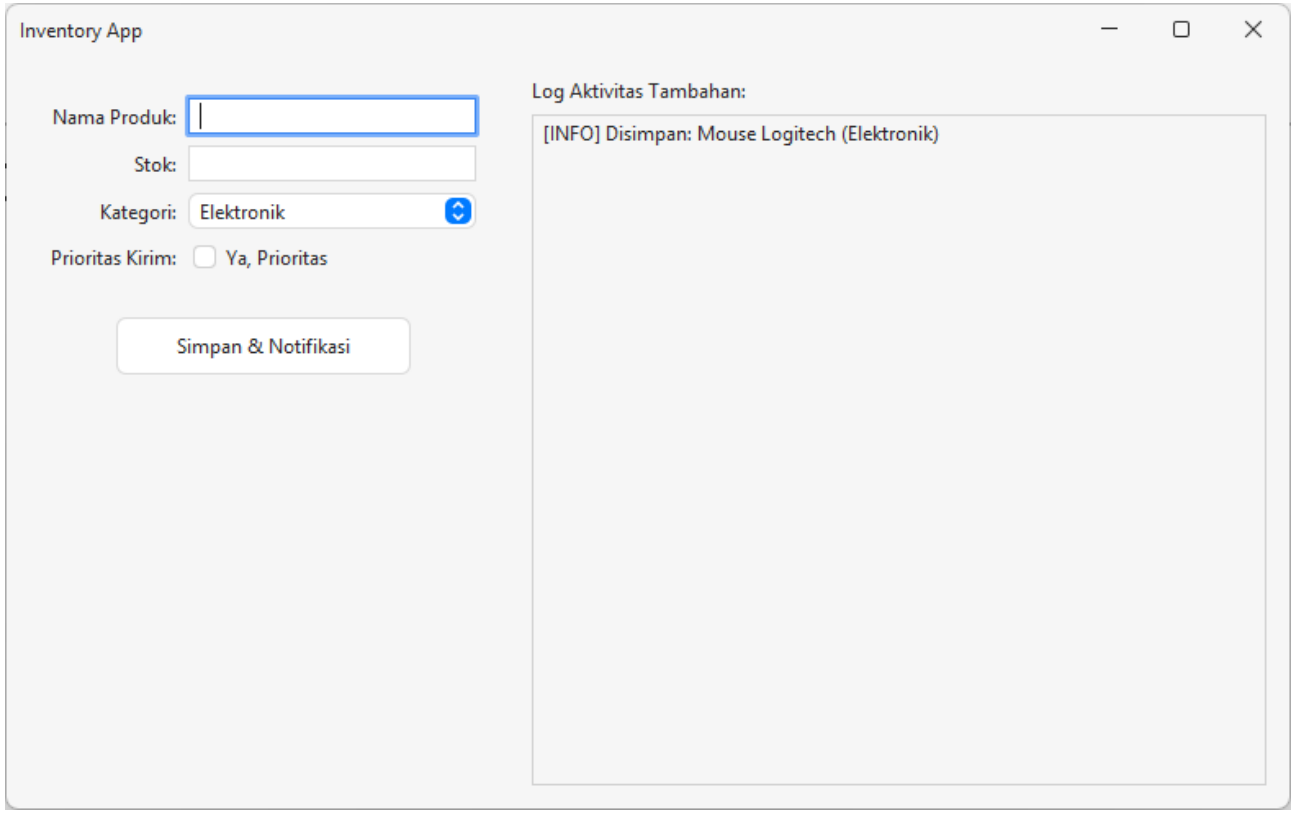
Analisis Komponen Swing + MigLayout



Analisis Komponen Swing + MigLayout

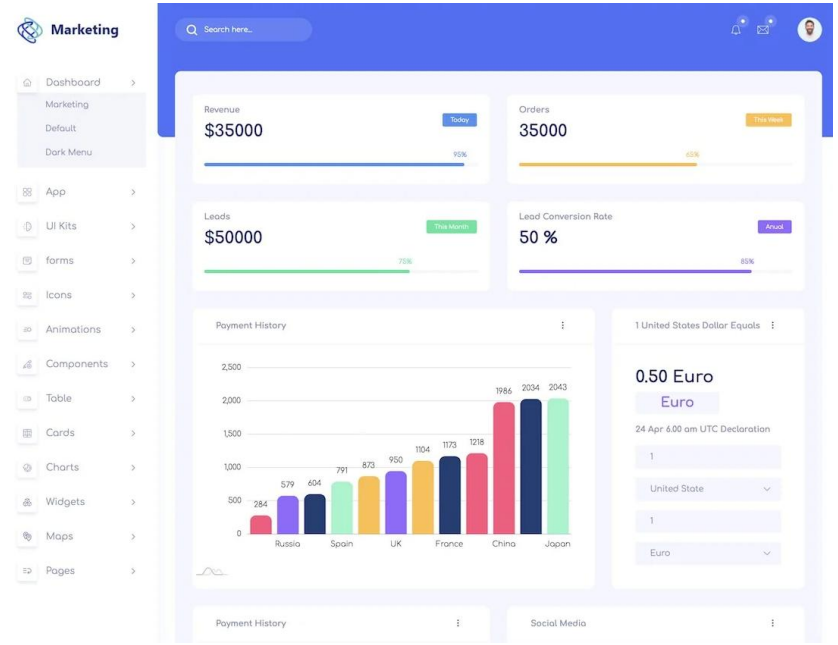


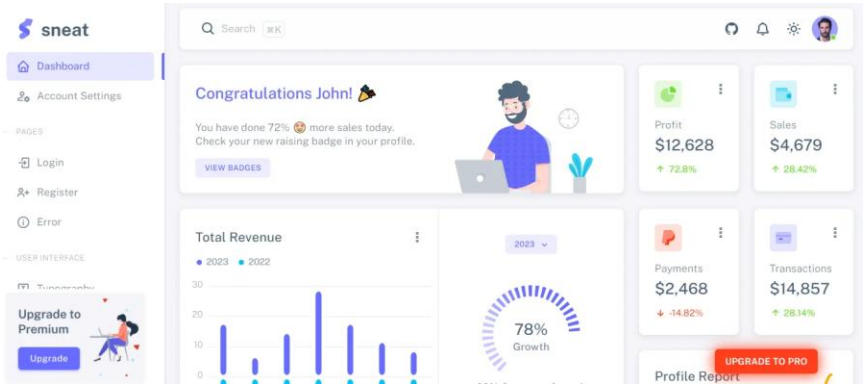
Analisis Komponen Swing + MigLayout

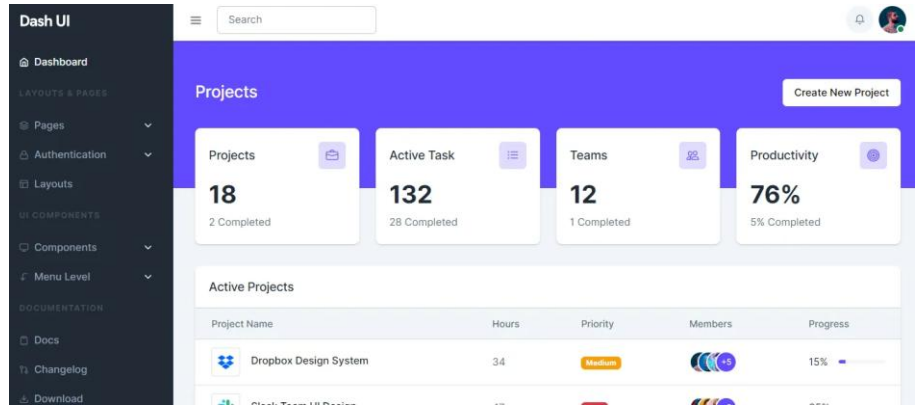


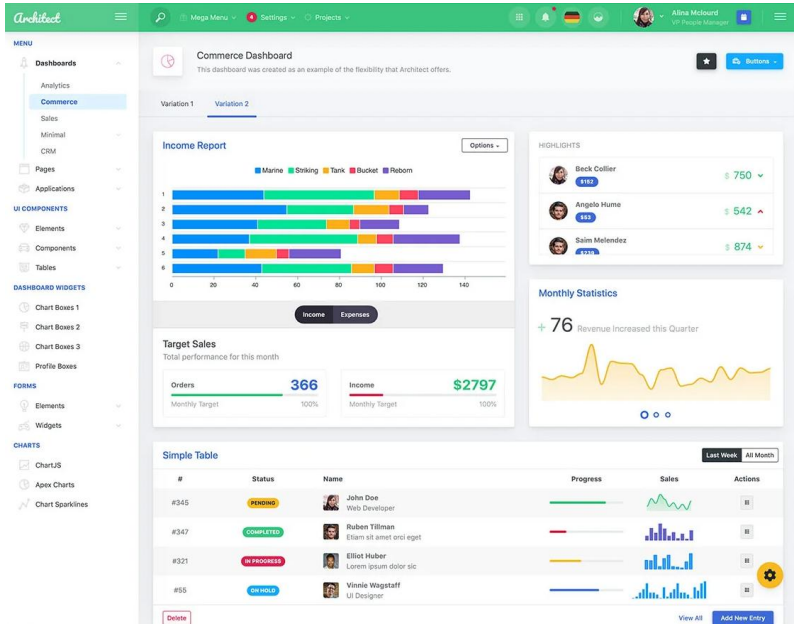
Menu UI

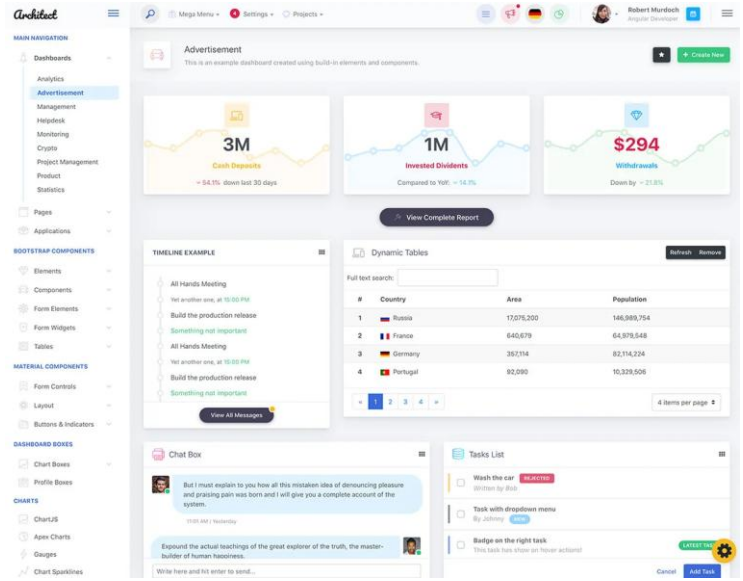

LOREM IPSUM


Marketing
Search here...
Dashboard
Marketing
Default
Dark Menu
App
UI Kits
forms
Icons
Animations
Components
Table
Cards
Charts
Widgets
Maps
Pages
Revenue
$35000
Orders
35000
Leads
$50000
Lead Conversion Rate
50 %
Payment History
1 United States Dollar Equals
0.50 Euro
Euro
Russia
Spain
UK
France
China
Japan
Social Media


sneat
Dashboard
Account Settings
Login
Register
Error
Congratulations John!
You have done 72% more sales today.
Check your new raising badge in your profile.
VIEW BADGES
Profit
$12,628
Sales
$4,679
Total Revenue
78%
Growth
Payments
$2,468
Transactions
$14,857
Upgrade to Premium
Upgrade
Profile Report
UPGRADE TO PRO


Dash UI
Dashboard
Pages
Authentication
Layouts
Components
Menu Level
Docs
Changelog
Download
Search
Projects
Create New Project
Projects
18
2 Completed
Active Task
132
28 Completed
Teams
12
1 Completed
Productivity
76%
5% Completed
Active Projects
Project Name
Hours
Priority
Members
Progress
Dropbox Design System
34
Medium


Architect
Commerce Dashboard
Income Report
Target Sales
Orders
366
Income
$2797
Monthly Statistics
Simple Table


Architect
Advertisement
3M
Cash Deposits
1M
Invested Dividents
$294
Withdrawals
View Complete Report
Dynamic Tables
Chat Box
Tasks List


Modernize
HOME
Dashboard
UI COMPONENTS
Buttons
Alerts
Card
Forms
Typography
AUTH
Login
Register
Download Free
Sales Overview
March 2023
Yearly Breakup
$36,358
+9% last year
Monthly Earnings
$6,820
+9% last year

Ringkasan Alur Kerja Penggunaan Swing

- Mulai di EDT, menggunakan SwingUtilities.invokeLater()
- Membuat container
- Mengorganisasikan layout (menentukan penempatan komponen)
- Menambahkan komponen
- Mengatur visibilitas (set visible untuk ditampilkan ke pengguna)

Selesai

Alhamdulillah. Terima kasih 🙏