PRATIKUM PEMROGRAMAN

PERTEMUAN KE - 4

Disusun Oleh :

Muhammad Fauzan Dwi Putera (223040033)

Kelas B

Teknik Informatika

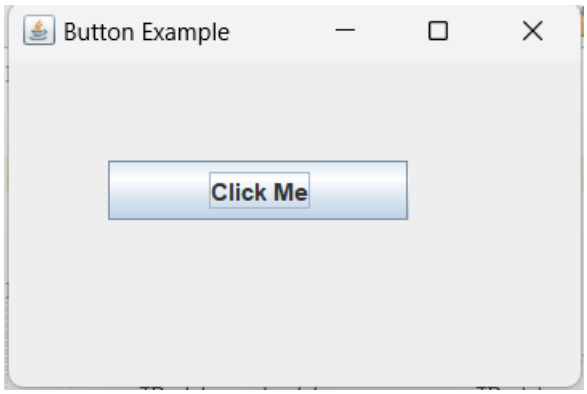
Fakultas Teknik

Universitas Pasundan

A. File ButtonExample

```java
package Latihan_1;

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


public class ButtonExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Button Example");
        JButton button = new JButton("Click Me");

        // Menambahlan Action listener pada  button
        button.addActionListener(new ActionListener() {
            public void actionPerformed (ActionEvent e){
                System.out.println("Button clicked");
            }
        });

        button.setBounds(50,50,150,30);
        frame.add(button);
        frame.setSize(300,200);
        frame.setLayout(null);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);


    }
}
```

Output



Penjelasan :

- **Import Library**

```
java
Copy code
import java.awt.event.*;
import javax.swing.*;
```

  - `import java.awt.event.*;` digunakan untuk mengimpor semua kelas yang berhubungan dengan *event handling* dari paket `java.awt.event`, seperti `ActionListener` dan `ActionEvent`.
  - `import javax.swing.*;` mengimpor semua kelas dari paket `javax.swing`, yang digunakan untuk membuat elemen GUI seperti `JFrame` (jendela) dan `JButton` (tombol).

- **Kelas Utama: `ButtonExample`**

```
java
Copy code
public class ButtonExample {
```

  - `ButtonExample` adalah nama kelas utama di mana semua logika program berada.

- **Metode `main`:**

```
java
Copy code
public static void main(String[] args) {
```

  - Metode `main` adalah titik awal eksekusi program.

- **Membuat JFrame:**

```
java
Copy code
JFrame frame = new JFrame("Button Example");
```

  - `JFrame` digunakan untuk membuat sebuah jendela GUI. Pada kasus ini, sebuah jendela bernama "Button Example" dibuat.

- **Membuat JButton:**

```
java
Copy code
JButton button = new JButton("Click Me");
```

  - `JButton` digunakan untuk membuat tombol yang menampilkan teks "Click Me".

- **Menambahkan `ActionListener` ke Tombol:**

```
java
Copy code
button.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        System.out.println("Button clicked");
    }
});
```

- Baris ini menambahkan *event listener* (pendengar kejadian) ke tombol. Saat tombol diklik, metode `actionPerformed` dipanggil, dan pesan "Button clicked" dicetak ke konsol.
- `ActionListener` adalah antarmuka yang menangani aksi seperti klik tombol, dan `ActionEvent` menangkap informasi mengenai kejadian tersebut.

- **Mengatur Lokasi dan Ukuran Tombol:**

```
java
Copy code
button.setBounds(50, 50, 150, 30);
```

- `setBounds` digunakan untuk mengatur posisi (x=50, y=50) dan ukuran (lebar=150, tinggi=30) tombol di dalam jendela.

- **Menambahkan Tombol ke JFrame:**

```
java
Copy code
frame.add(button);
```

- Tombol ditambahkan ke dalam jendela (frame) yang dibuat sebelumnya.

- **Mengatur Ukuran dan Properti JFrame:**

```
java
Copy code
frame.setSize(300, 200);
frame.setLayout(null);
frame.setVisible(true);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

- `frame.setSize(300, 200);`: Mengatur ukuran jendela menjadi lebar 300 piksel dan tinggi 200 piksel.
- `frame.setLayout(null);`: Mengatur tata letak frame menjadi *null* untuk memberi kontrol penuh pada penempatan elemen GUI menggunakan `setBounds`.
- `frame.setVisible(true);`: Membuat jendela terlihat.
- `frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);`: Menutup program saat jendela ditutup.

B. MouseListenerExample

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

public class MouseListenerExample {
    public static void main(String[] args) {
        // Membuat frame
        JFrame frame = new JFrame("MouseListener Example");

        // Membuat label untuk menampilkan pesan
        JLabel label = new JLabel("Arahkan dan klik mouse pada area ini.");
        label.setBounds(50, 50, 300, 30);

        // Menambahkan MouseListener ke label
        label.addMouseListener(new MouseListener() {
            // Dijalankan ketika mouse diklik (klik kiri, kanan atau tengah)
            public void mouseClicked(MouseEvent e) {
                label.setText("Mouse Clicked at: (" + e.getX() + ", " +
e.getY() + ")");
            }

            // Dijalankan ketika mouse ditekan (tombol belum dilepaskan)
            public void mousePressed(MouseEvent e) {
                label.setText("Mouse Pressed at: (" + e.getX() + ", " +
e.getY() + ")");
            }

            // Dijalankan ketika mouse dilepaskan setelah ditekan
            public void mouseReleased(MouseEvent e) {
                label.setText("Mouse Released at: (" + e.getX() + ", " +
e.getY() + ")");
            }

            // Dijalankan ketika mouse masuk ke area komponen
            public void mouseEntered(MouseEvent e) {
                label.setText("Mouse Entered the area.");
            }

            // Dijalankan ketika mouse keluar dari area komponen
            public void mouseExited(MouseEvent e) {
                label.setText("Mouse Exited the area.");
            }
        });
```

```
        // Menambahkan label ke frame
        frame.add(label);

        // Setting frame
        frame.setSize(400, 200);
        frame.setLayout(null);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

1. Import Library

```
java
Copy code
import java.awt.event.*;
import javax.swing.*;
```

- `import java.awt.event.*;` digunakan untuk mengimpor semua kelas terkait event seperti `MouseListener` dan `MouseEvent`.
- `import javax.swing.*;` mengimpor semua kelas dari paket `javax.swing`, yang digunakan untuk membuat komponen GUI seperti `JFrame` dan `JLabel`.

2. Kelas Utama: `MouseListenerExample`

```
java
Copy code
public class MouseListenerExample {
```

- `MouseListenerExample` adalah kelas utama yang akan berisi semua logika untuk menangani event dari mouse.

3. Metode `main`:

```
java
Copy code
public static void main(String[] args) {
```

- Metode `main` adalah titik awal program, di mana semua komponen dan event listener diatur.

4. Membuat JFrame:

```
java
Copy code
JFrame frame = new JFrame("MouseListener Example");
```

- Membuat sebuah jendela (frame) dengan judul "MouseListener Example" yang akan menampung komponen GUI seperti label.

5. Membuat JLabel:

```
java
Copy code
JLabel label = new JLabel("Arahkan dan klik mouse pada area ini.");
label.setBounds(50, 50, 300, 30);
```

- `JLabel` digunakan untuk menampilkan teks pada GUI. Di sini, label menampilkan pesan "Arahkan dan klik mouse pada area ini.".
- `setBounds(50, 50, 300, 30)` mengatur posisi label pada jendela (x=50, y=50) serta lebar 300 piksel dan tinggi 30 piksel.

6. Menambahkan MouseListener ke JLabel:

```
java
Copy code
label.addMouseListener(new MouseListener() {
```

- Menambahkan *MouseListener* ke label untuk memantau pergerakan atau aksi yang dilakukan oleh mouse. *MouseListener* adalah antarmuka yang menyediakan lima metode untuk menangani kejadian yang berbeda.

7. Metode-metode dalam MouseListener:

- **`mouseClicked(MouseEvent e)`**:
  - Dijalankan ketika mouse diklik. Program akan mengubah teks pada label untuk menampilkan koordinat X dan Y tempat mouse diklik.

```
java
Copy code
public void mouseClicked(MouseEvent e) {
    label.setText("Mouse Clicked at: (" + e.getX() + ", " + e.getY()
+ ")");
}
```

- **`mousePressed(MouseEvent e)`**:
  - Dijalankan saat mouse ditekan, sebelum dilepaskan. Teks pada label diubah untuk menunjukkan lokasi di mana mouse ditekan.

```
java
Copy code
public void mousePressed(MouseEvent e) {
    label.setText("Mouse Pressed at: (" + e.getX() + ", " + e.getY()
+ ")");
}
```

- **`mouseReleased(MouseEvent e)`**:
  - Dijalankan saat tombol mouse dilepaskan. Teks label akan diubah untuk menunjukkan posisi tempat mouse dilepaskan.

```
java
Copy code
public void mouseReleased(MouseEvent e) {
```

```
    label.setText("Mouse Released at: (" + e.getX() + ", " + e.getY()
+ ")");
}
```

- **`mouseEntered(MouseEvent e)`**:
  - Dijalankan ketika mouse masuk ke area label. Teks label akan diubah menjadi "Mouse Entered the area."

```
java
Copy code
public void mouseEntered(MouseEvent e) {
    label.setText("Mouse Entered the area.");
}
```

- **`mouseExited(MouseEvent e)`**:
  - Dijalankan ketika mouse keluar dari area label. Teks label akan diubah menjadi "Mouse Exited the area."

```
java
Copy code
public void mouseExited(MouseEvent e) {
    label.setText("Mouse Exited the area.");
}
```

8. Menambahkan JLabel ke JFrame:

```
java
Copy code
frame.add(label);
```

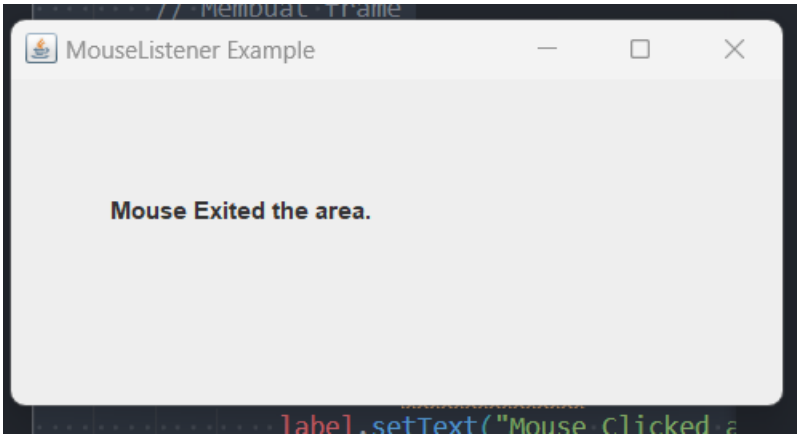
- Menambahkan label yang telah dibuat ke dalam frame (jendela).

9. Pengaturan JFrame:

```
java
Copy code
frame.setSize(400, 200);
frame.setLayout(null);
frame.setVisible(true);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

- `frame.setSize(400, 200);`: Mengatur ukuran frame menjadi 400 piksel lebar dan 200 piksel tinggi.
- `frame.setLayout(null);`: Menggunakan *layout manager* null untuk memberi kontrol penuh pada penempatan komponen.
- `frame.setVisible(true);`: Membuat frame terlihat di layar.
- `frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);`: Mengakhiri program ketika jendela ditutup.

Output :



C. KeyListenerExample

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



public class KeyListenerExample {

    public static void main(String[] args){
        // Membuat frame
        JFrame frame = new JFrame("KeyListener Example");

        // Membuat label untuk menampilkan pesan
        JLabel label = new JLabel("Tekan tombol pada keyboard");

        // Membuat text field untuk fokus keyboard
        JTextField textField = new JTextField();
        textField.setBounds(50, 100, 200, 30);

        // Menambahkan KeyListener ke text field
        textField.addKeyListener(new KeyListener() {
            // Dijalankan ketika tombol ditekan
            @Override
            public void keyPressed(KeyEvent e) {
                label.setText("Key Pressed: " +
KeyEvent.getKeyText(e.getKeyCode()));
            }

            // Dijalankan ketika tombol dilepaskan
            @Override
            public void keyReleased(KeyEvent e) {
                label.setText("Key Released: " +
KeyEvent.getKeyText(e.getKeyCode()));
```

```
            }

            // Dijalankan ketika tombol ditekan dan dilepaskan (sama dengan
mengetik karakter)
            @Override
            public void keyTyped(KeyEvent e) {
                label.setText("Key Typed: " + e.getKeyChar());
            }
        });

        // Menambahkan Komponen ke frame
        frame.add(label);
        frame.add(textField);

        // Setting frame
        frame.setSize(400, 200);
        frame.setLayout(null);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    }

}
```

1. Import Library

```
java
Copy code
import java.awt.event.*;
import javax.swing.*;
```

- `import java.awt.event.*;` mengimpor semua kelas yang berhubungan dengan event dari keyboard, seperti `KeyListener` dan `KeyEvent`.
- `import javax.swing.*;` mengimpor semua kelas dari paket `javax.swing`, yang digunakan untuk membuat elemen GUI seperti `JFrame`, `JLabel`, dan `JTextField`.

2. Kelas Utama: `KeyListenerExample`

```
java
Copy code
public class KeyListenerExample {
```

- `KeyListenerExample` adalah kelas utama yang berisi logika untuk menangani event dari keyboard.

3. Metode `main`:

```
java
Copy code
```

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

- Metode `main` adalah titik awal eksekusi program. Di dalam metode ini, semua komponen GUI dan *event listener* diatur.

4. Membuat JFrame:

```
java
Copy code
JFrame frame = new JFrame("KeyListener Example");
```

- Membuat sebuah jendela (frame) dengan judul "KeyListener Example".

5. Membuat JLabel:

```
java
Copy code
JLabel label = new JLabel("Tekan tombol pada keyboard");
```

- Membuat sebuah label untuk menampilkan pesan awal "Tekan tombol pada keyboard". Teks pada label ini akan diubah ketika ada event dari keyboard.

6. Membuat JTextField:

```
java
Copy code
JTextField textField = new JTextField();
textField.setBounds(50, 100, 200, 30);
```

- Membuat *text field* untuk tempat input teks dari keyboard. Komponen ini akan menjadi fokus untuk menangkap event keyboard.
- `setBounds(50, 100, 200, 30)` mengatur posisi *text field* pada jendela (x=50, y=100) serta lebar 200 piksel dan tinggi 30 piksel.

7. Menambahkan KeyListener ke JTextField:

```
java
Copy code
textField.addKeyListener(new KeyListener() {
```

- *KeyListener* ditambahkan ke *text field* untuk memonitor event dari keyboard, seperti tombol ditekan, dilepaskan, atau diketik.

8. Metode-metode dalam KeyListener:

- **`keyPressed(KeyEvent e)`**:
  - Dijalankan ketika tombol pada keyboard ditekan. Pesan yang ditampilkan pada label akan menunjukkan tombol yang ditekan.

```
java
Copy code
public void keyPressed(KeyEvent e) {
```

```
    label.setText("Key Pressed: " +
KeyEvent.getKeyText(e.getKeyCode()));
}
```

- `KeyEvent.getKeyText(e.getKeyCode())` digunakan untuk mendapatkan nama tombol yang ditekan, seperti "A", "Shift", atau "Enter".

- **`keyReleased(KeyEvent e)`**:
  - Dijalankan saat tombol keyboard dilepaskan setelah ditekan. Teks label akan diubah menjadi "Key Released: [nama tombol]".

```
java
Copy code
public void keyReleased(KeyEvent e) {
    label.setText("Key Released: " +
KeyEvent.getKeyText(e.getKeyCode()));
}
```

- **`keyTyped(KeyEvent e)`**:
  - Dijalankan ketika sebuah karakter diketik (ketika tombol ditekan dan dilepaskan). Berbeda dengan `keyPressed` atau `keyReleased`, metode ini fokus pada karakter yang dihasilkan dari penekanan tombol.

```
java
Copy code
public void keyTyped(KeyEvent e) {
    label.setText("Key Typed: " + e.getKeyChar());
}
```

  - `e.getKeyChar()` mengembalikan karakter yang diketik, seperti 'a', '1', atau simbol lainnya.

9. Menambahkan Komponen ke JFrame:

```
java
Copy code
frame.add(label);
frame.add(textField);
```

- Label dan *text field* ditambahkan ke dalam frame (jendela).

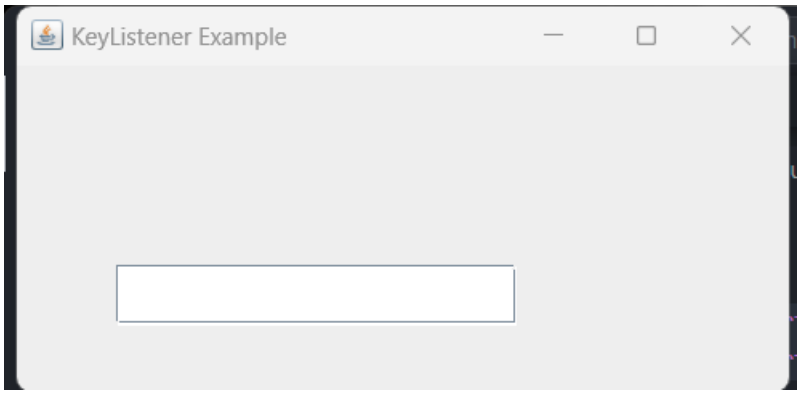
10. Pengaturan JFrame:

```
java
Copy code
frame.setSize(400, 200);
frame.setLayout(null);
frame.setVisible(true);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

- `frame.setSize(400, 200);`: Mengatur ukuran jendela menjadi 400 piksel lebar dan 200 piksel tinggi.
- `frame.setLayout(null);`: Mengatur *layout manager* menjadi null untuk memberi kontrol penuh terhadap penempatan komponen.
- `frame.setVisible(true);`: Membuat jendela terlihat di layar.

- `frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);`: Menutup program ketika jendela ditutup.

Outputnya:



D.WindowsListenerExample

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

public class WindowListenerExample {
    public static void main(String[] args) {
        JFrame f = new JFrame("Window Listener Example");

        JLabel label = new JLabel("lakukan operasi pada jendela");
        label.setBounds(50,50,300, 30);

        f.addWindowListener(new WindowListener() {

            public void windowOpened(WindowEvent e) {
                label.setText("Window Opened");
            }
            public void windowClosing(WindowEvent e) {
                label.setText("Window Closing");

            }
            public void windowClosed(WindowEvent e) {
                System.out.println("Window Closed");
            }
            public void windowIconified(WindowEvent e) {
                label.setText("Window Minimized");
            }
            public void windowDeiconified(WindowEvent e) {
```

```
                label.setText("Window Restored");
            }
            public void windowActivated(WindowEvent e) {
                label.setText("Window Activated");
            }
            public void windowDeactivated(WindowEvent e) {
                label.setText("Window Deactivated");
            }
        });

        f.add(label);
        f.setSize(400, 400);
        f.setLayout(null);
        f.setVisible(true);
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

1. Import Library

```
java
Copy code
import java.awt.event.*;
import javax.swing.*;
```

- `import java.awt.event.*;` mengimpor semua kelas yang berhubungan dengan event dari jendela, seperti `WindowListener` dan `WindowEvent`.
- `import javax.swing.*;` mengimpor semua kelas yang diperlukan untuk membuat antarmuka grafis (GUI) seperti `JFrame` dan `JLabel`.

2. Membuat JFrame:

```
java
Copy code
JFrame f = new JFrame("Window Listener Example");
```

- Membuat objek `JFrame` (jendela) dengan judul "Window Listener Example". Frame ini adalah jendela utama aplikasi.

3. Membuat JLabel:

```
java
Copy code
JLabel label = new JLabel("lakukan operasi pada jendela");
label.setBounds(50, 50, 300, 30);
```

- Membuat label yang menampilkan teks "lakukan operasi pada jendela". Label ini digunakan untuk menampilkan pesan sesuai dengan event yang terjadi pada jendela.

- `setBounds(50, 50, 300, 30)` mengatur posisi label pada jendela di koordinat (x=50, y=50) serta lebar 300 piksel dan tinggi 30 piksel.

4. Menambahkan WindowListener ke JFrame:

```
java
Copy code
f.addWindowListener(new WindowListener() {
```

- Baris ini menambahkan *WindowListener* ke jendela untuk menangani event-event yang terkait dengan perubahan status jendela, seperti dibuka, ditutup, diminimalkan, dsb.

5. Implementasi Metode-metode dalam WindowListener:

Di dalam *WindowListener*, kita mengimplementasikan beberapa metode untuk menangani event yang terjadi pada jendela. Berikut adalah event-event yang ditangani:

- **`windowOpened(WindowEvent e)`**:
  - Dijalankan ketika jendela pertama kali dibuka.

  ```
  java
  Copy code
  label.setText("Window Opened");
  ```

- **`windowClosing(WindowEvent e)`**:
  - Dijalankan saat jendela sedang dalam proses ditutup (misalnya saat pengguna mengklik tombol 'X' di pojok jendela).

  ```
  java
  Copy code
  label.setText("Window Closing");
  ```

- **`windowClosed(WindowEvent e)`**:
  - Dijalankan setelah jendela benar-benar ditutup. Di sini, program hanya mencetak pesan "Window Closed" ke konsol.

  ```
  java
  Copy code
  System.out.println("Window Closed");
  ```

- **`windowIconified(WindowEvent e)`**:
  - Dijalankan saat jendela diminimalkan (di-icon-kan).

  ```
  java
  Copy code
  label.setText("Window Minimized");
  ```

- **`windowDeiconified(WindowEvent e)`**:
  - Dijalankan saat jendela dipulihkan dari status diminimalkan.

  ```
  java
  ```

```
Copy code
label.setText("Window Restored");
```

- **`windowActivated(WindowEvent e)`**:
  - Dijalankan saat jendela diaktifkan (misalnya, ketika pengguna beralih ke jendela ini dari jendela lain).

```
java
Copy code
label.setText("Window Activated");
```

- **`windowDeactivated(WindowEvent e)`**:
  - Dijalankan ketika jendela dinonaktifkan (misalnya, ketika pengguna beralih ke aplikasi lain atau jendela lain).

```
java
Copy code
label.setText("Window Deactivated");
```

6. Menambahkan Komponen ke JFrame:

```
java
Copy code
f.add(label);
```

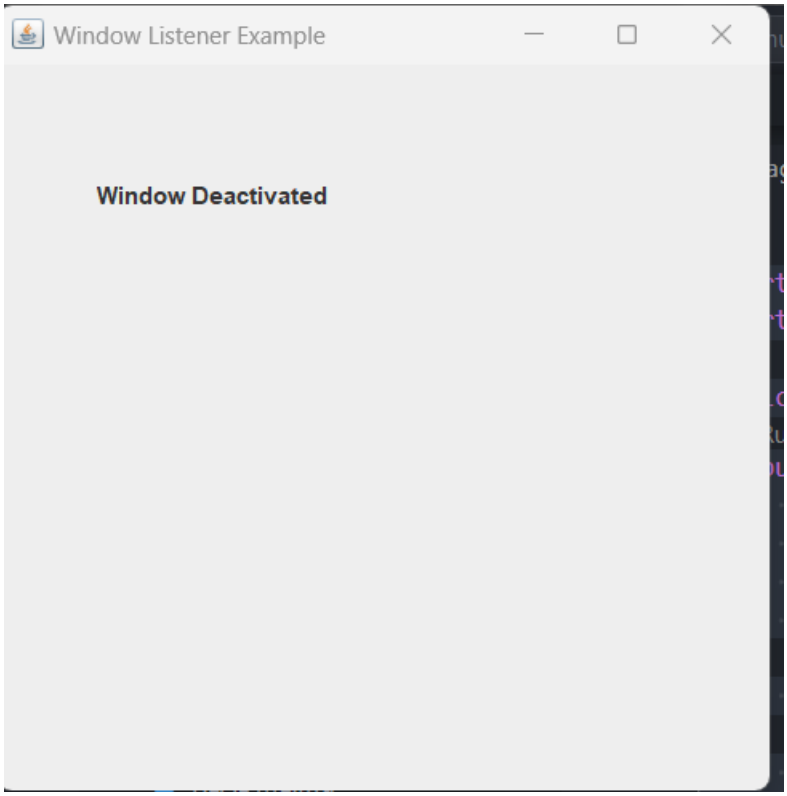
- Menambahkan label ke dalam jendela yang sudah dibuat.

7. Pengaturan JFrame:

```
java
Copy code
f.setSize(400, 400);
f.setLayout(null);
f.setVisible(true);
f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

- `f.setSize(400, 400);`: Mengatur ukuran jendela menjadi 400x400 piksel.
- `f.setLayout(null);`: Menggunakan *null layout* untuk penempatan manual komponen-komponen GUI.
- `f.setVisible(true);`: Membuat jendela terlihat oleh pengguna.
- `f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);`: Mengatur agar program dihentikan ketika jendela ditutup.

Outputnya :


Window Listener Example
Window Deactivated