



DAFTAR ISI

DAFTAR ISI.....	1
A. IDENTITAS.....	2
B. HASIL PRAKTIKUM/ TUGAS	2
1. RINGKASAN MODUL	3
2. PRAKTIKUM.....	3
a. Demo 1	3
b. Demo 2.....	8
c. Demo 3.....	13
d. Demo 4.....	18
e. Hitung Mundur Worker	22
f. Demo 5.....	24
3. LATIHAN	30
a. Task Worker	30
b. Demo 6.....	32
C. PENGALAMAN PEMBELAJARAN	38



A. IDENTITAS

NIM : 2403001
Nama Lengkap : Siti Sa'adah
Kelas : D3 TI 2A
Program Studi : D3 TEKNIK INFORMATIKA
Jurusan : TEKNIK INFORMATIKA

B. HASIL PRAKTIKUM/ TUGAS

Sajikanlah Laporan **Praktikum 12 - SwingWorker** dengan format (*.pdf) dan ketentuan sebagai berikut:

1. Terdapat **Ringkasan** materi pada Modul 12 - SwingWorker
2. Terdapat penyelesaian **Praktikum** Demo 1 UI Freeze yang disertai bukti hasil running dan pembuktian UI Freeze
3. Terdapat penyelesaian **Praktikum** Demo 2 SwingWroker (Anonymous Inner Class) yang disertai bukti hasil running dan pembuktian UI Interaktif (tidak freeze)
4. Terdapat penyelesaian **Praktikum** Demo 3 SwingWroker (Static Nested Class) yang disertai bukti hasil running dan pembuktian UI Interaktif (tidak freeze)
5. Terdapat penyelesaian **Praktikum** Demo 4 SwingWroker (Separate Top-Level Class) yang terdiri dari 2 file (Demo4.java dan HitungMundurWorker.java) disertai bukti hasil running dan pembuktian UI Interaktif (tidak freeze)
6. Terdapat penyelesaian **Praktikum** Demo 5 SwingWroker + Cancelation (Anonymous Inner Class) yang disertai bukti hasil running dan pembuktian UI Interaktif (tidak freeze)
7. Terdapat penyelesaian **Latihan** Demo 6 SwingWroker + Cancelation (Separate Top-Level Class) yang terdiri dari 2 file (Demo6.java dan TaskWorker.java) disertai bukti hasil running dan pembuktian UI Interaktif (tidak freeze)
8. Terdapat penyajian redaksi **pengalaman pembelajaran yang didapat** (secara **pribadi**) selama mempelajari materi SwingWorker secara keseluruhan (baik teori, praktikum, maupun latihan)



1. RINGKASAN MODUL

Pada materi kali ini membahas mengenai SWING WORKER. Dimana di dalam nya menjelaskan mengenai *Corcurency* yang di implemntasikan dengan berbagai cara;

diantaranya :

- Multithreading
- Multiprocessing
- Single-threaded concurrency

2. PRAKTIKUM

a. Demo 1

- Kode Program

```
package Praktikum;
import java.awt.*;
import java.awt.event.ActionEvent;

import javax.swing.*;

import net.miginfocom.swing.MigLayout;

public class Demo1 extends JFrame {

    private JPanel panelUtama;
    private JButton buttonMulaiTask;
    private JLabel labelStatus;
    private JTextArea textAreaHasil;

    public Demo1() {
        initializeUI();
        setupEventHandlers();
    }

    private void initializeUI() {
        setTitle("Demo 1: UI Freeze");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setPreferredSize(new Dimension(560, 420));

        panelUtama = new JPanel(
            new MigLayout(
                "fill, wrap 1, insets 20",
                "[grow]"
            )
        );
    }
};
```



```
buttonMulaiTask = new JButton("Mulai Task (8 detik)");
buttonMulaiTask.setFont(new Font("Inter", Font.BOLD, 18));
panelUtama.add(buttonMulaiTask, "growx, h 20%");

labelStatus = new JLabel("Status: Siap", JLabel.CENTER);
labelStatus.setFont(new Font("Inter", Font.PLAIN, 22));
labelStatus.setForeground(new Color(0, 102, 204));
panelUtama.add(labelStatus, "growx, h 20%");

textAreaHasil = new JTextArea(10, 50);
textAreaHasil.setEditable(false);
textAreaHasil.setFont(new Font("Consolas", Font.PLAIN, 15));
JScrollPane scroll = new JScrollPane(textAreaHasil);
panelUtama.add(scroll, "growx, h 60%");

add(panelUtama);
pack();
setLocationRelativeTo(null);
}

private void setupEventHandlers() {
    buttonMulaiTask.addActionListener((ActionEvent e) -> {
        buttonMulaiTask.setEnabled(false);
        labelStatus.setText("Status: Sedang bekerja");
        textAreaHasil.setText("Mengerjakan Background Task (8 detik)\n");

        for (int i = 8; i >= 0; i--) {
            final int seconds = i;
            SwingUtilities.invokeLater(() -> {
                textAreaHasil.append("Hitung mundur: " + seconds + "\n");
                textAreaHasil.setCaretPosition(textAreaHasil.getDocument()
.getLength());

                labelStatus.setText("Sisa " + seconds + " detik");
            });

            try {
                Thread.sleep(1000);
            } catch (InterruptedException ex) {
                Thread.currentThread().interrupt();
            }
        }

        textAreaHasil.append("Background Task selesai\n");
        labelStatus.setText("Status: Selesai");
        JOptionPane.showMessageDialog(
            this,
            "Background Task Selesai",
            "Informasi",
```



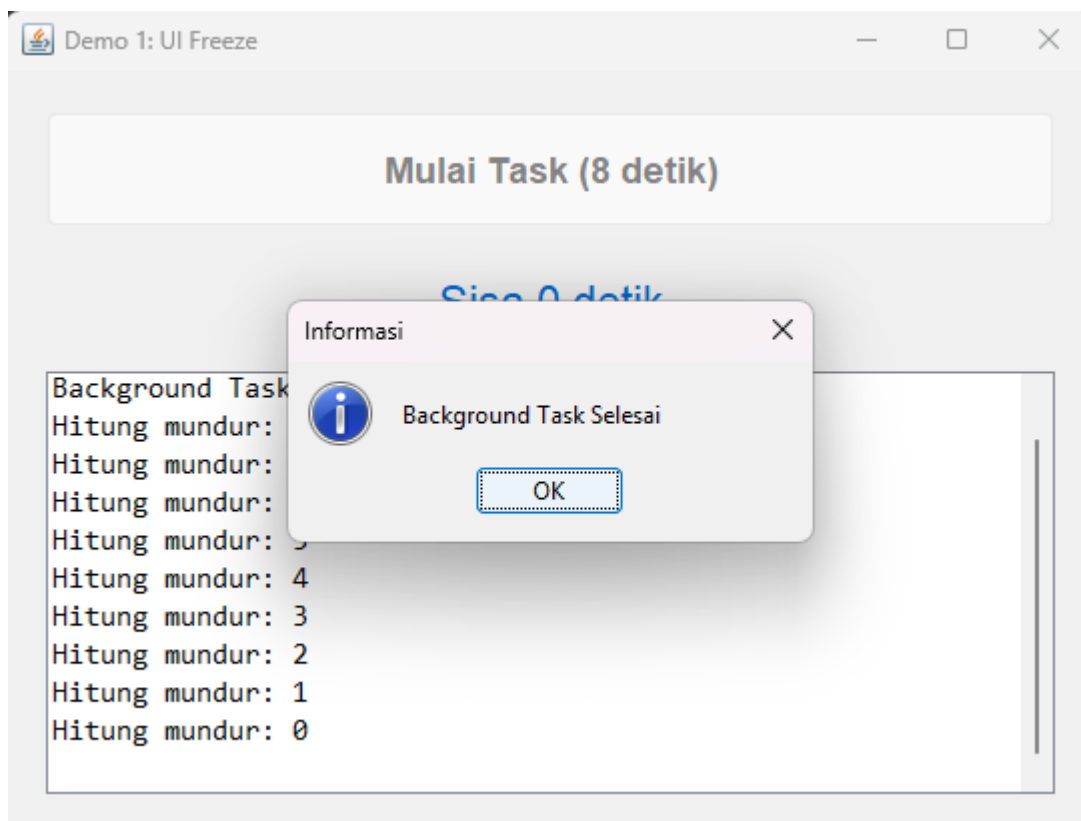
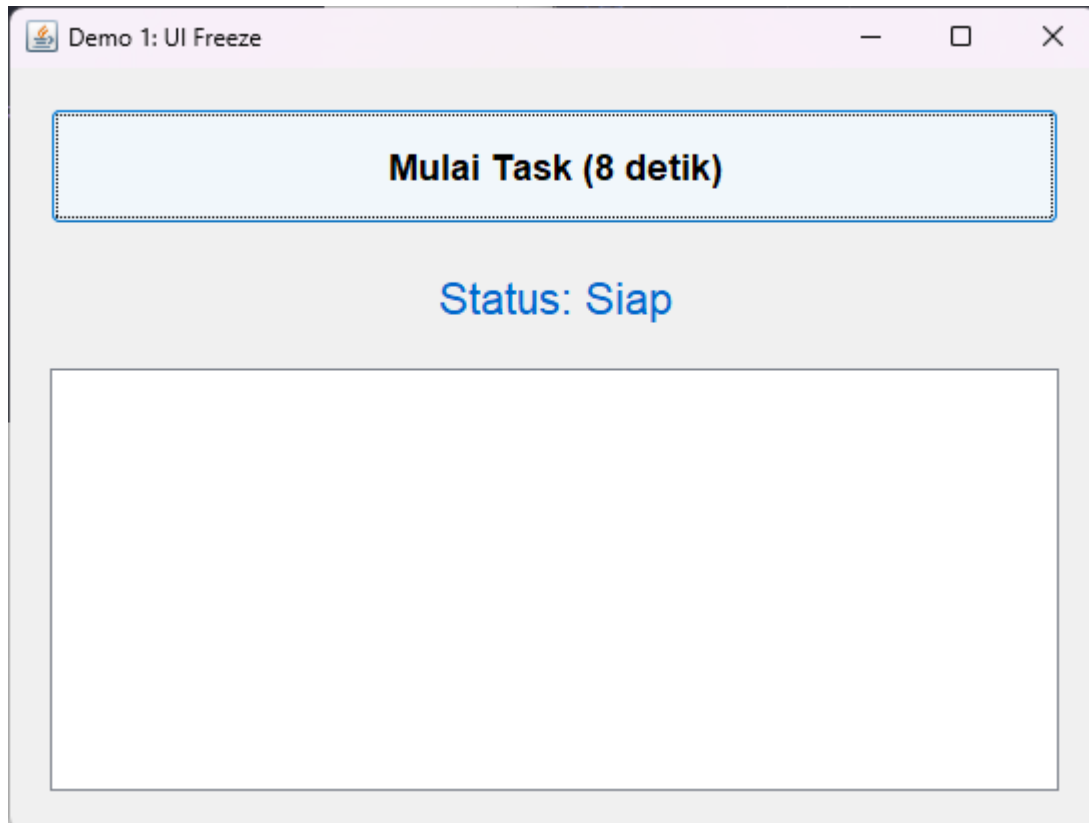
```
        JOptionPane.INFORMATION_MESSAGE
    );
    buttonMulaiTask.setEnabled(true);
});
}

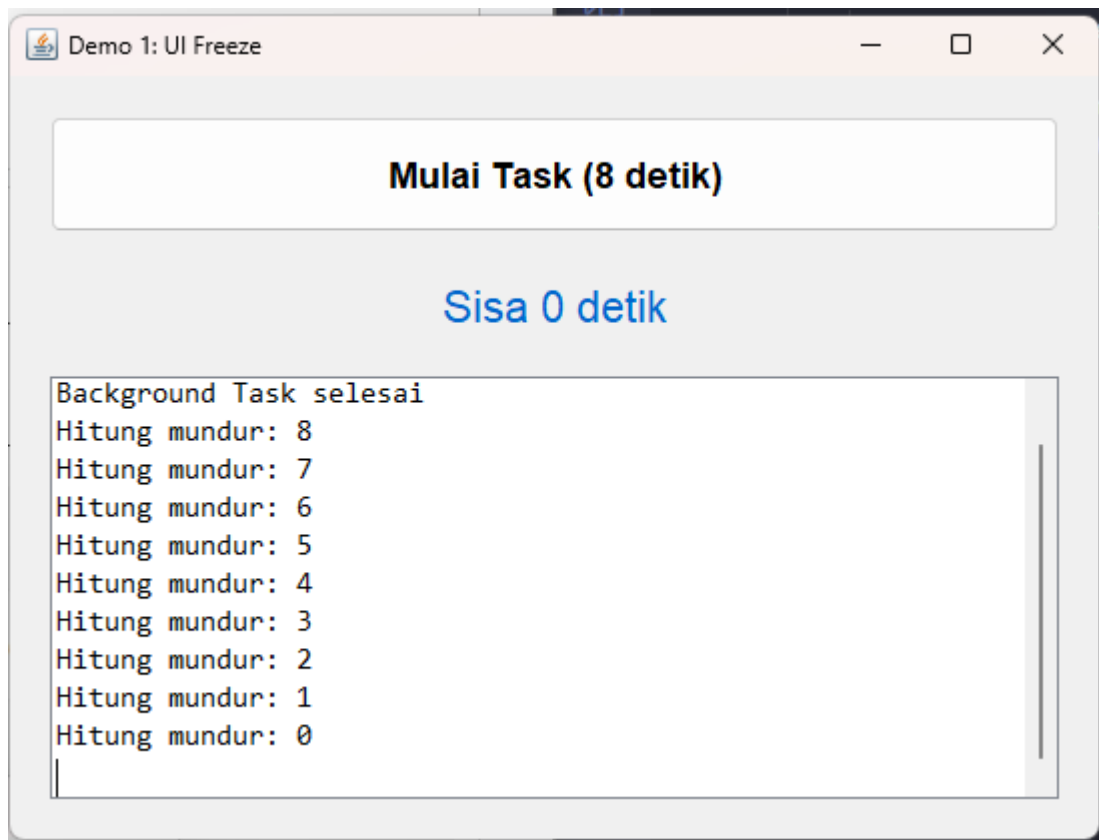
public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> {
        try {
            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
        } catch (Exception e) { }

        new Demo1().setVisible(true);
    });
}
```



- Hasil Run







b. Demo 2

- Kode Program

```
package Praktikum;
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import java.util.List;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.SwingUtilities;
import javax.swing.SwingWorker;
import javax.swing.UIManager;

import net.miginfocom.swing.MigLayout;

public class Demo2 extends JFrame {

    private JPanel panelUtama;
    private JButton buttonMulaiTask;
    private JLabel labelStatus;
    private JTextArea textAreaHasil;
    private SwingWorker<String, Integer> worker;

    public Demo2() {
        initializeUI();
        setupEventHandlers();
    }

    private void initializeUI() {
        setTitle("Demo 2: SwingWorker (Anonymous Inner Class)");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setPreferredSize(new Dimension(600, 450));

        panelUtama = new JPanel(
            new MigLayout("fill, wrap 1, insets 20")
        );

        buttonMulaiTask = new JButton("Mulai Task (8 detik)");
        buttonMulaiTask.setFont(new Font("Inter", Font.BOLD, 18));
        panelUtama.add(buttonMulaiTask, "growx, h 20%");
```



```
labelStatus = new JLabel("Status: Siap", JLabel.CENTER);
labelStatus.setFont(new Font("Inter", Font.PLAIN, 22));
labelStatus.setForeground(new Color(0, 128, 0));
panelUtama.add(labelStatus, "growx, h 20%");

textAreaHasil = new JTextArea();
textAreaHasil.setEditable(false);
textAreaHasil.setFont(new Font("Consolas", Font.PLAIN, 15));
JScrollPane scroll = new JScrollPane(textAreaHasil);
panelUtama.add(scroll, "growx, h 60%");

add(panelUtama);
pack();
setLocationRelativeTo(null);
}

private void setupEventHandlers() {
    buttonMulaiTask.addActionListener(e -> {
        buttonMulaiTask.setEnabled(false);
        labelStatus.setText("Status: Mulai bekerja");
        textAreaHasil.setText("Mengerjakan Background Task (8 detik)\n");

        worker = new SwingWorker<String, Integer>() {
            @Override
            protected String doInBackground() throws Exception {
                for (int i = 8; i >= 0; i--) {
                    Thread.sleep(1000);
                    publish(i);
                }
                return "Background Task Selesai";
            }

            @Override
            protected void process(List<Integer> chunks) {
                int detik = chunks.get(chunks.size() - 1);
                textAreaHasil.append("Hitung mundur: " + detik + "\n");
                textAreaHasil.setCaretPosition(textAreaHasil.getDocument()
                    .getLength());

                labelStatus.setText("Sisa " + detik + " detik");
            }

            @Override
            protected void done() {
                try {
                    String hasil = get();
                    textAreaHasil.append(hasil);
                } catch (InterruptedException e) {}
            }
        };
    });
}
```



```
        labelStatus.setText("Status: Selesai");
        JOptionPane.showMessageDialog(
            Demo2.this,
            hasil,
            "Informasi",
            JOptionPane.INFORMATION_MESSAGE
        );
    } catch (Exception ex) {
        labelStatus.setText("Status: Error");
    } finally {
        buttonMulaiTask.setEnabled(true);
    }
}

};

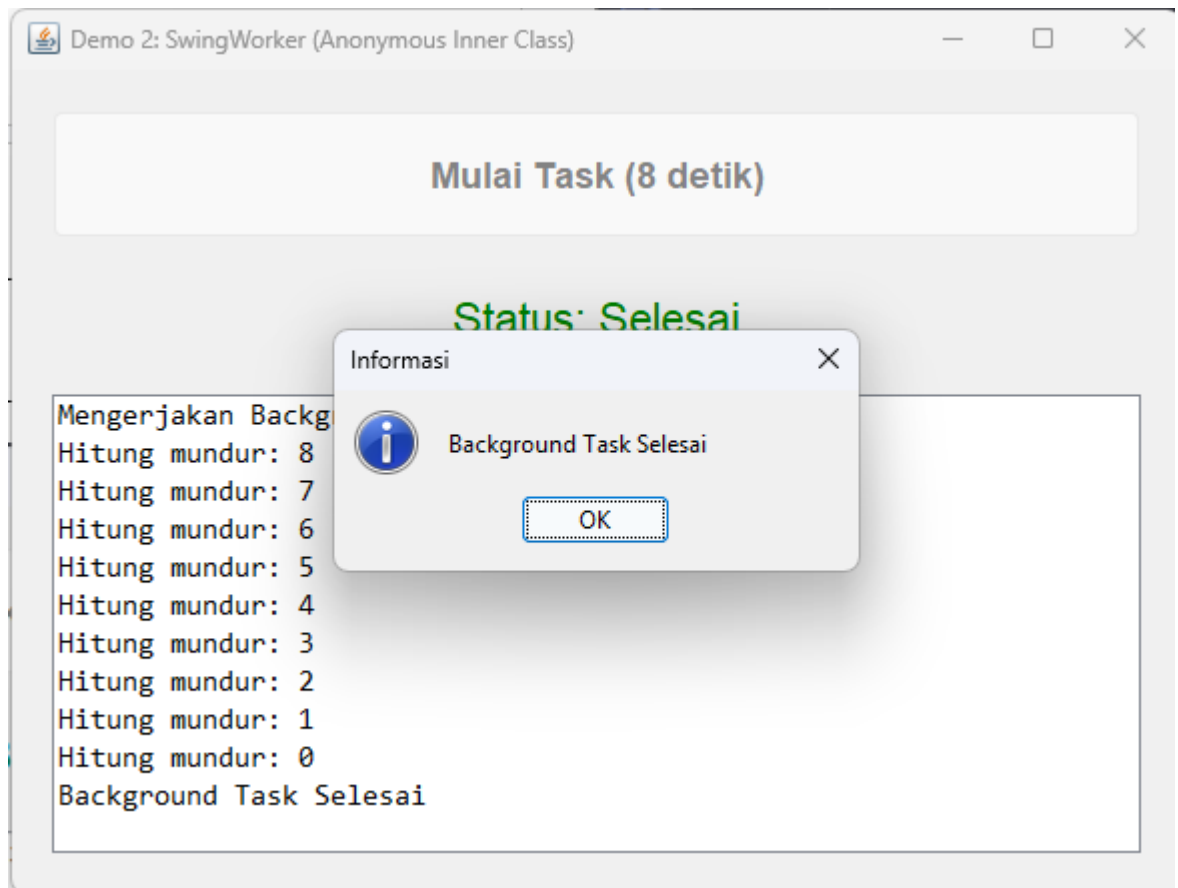
worker.execute();
});
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> {
        try {
            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
        } catch (Exception e) {
            System.err.println(e.getMessage());
        }
        new Demo2().setVisible(true);
    });
}
}
```



- Hasil Run







c. Demo 3

- Kode Program

```
package Praktikum;
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import java.util.List;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.SwingUtilities;
import javax.swing.SwingWorker;
import javax.swing.UIManager;

import net.miginfocom.swing.MigLayout;

public class Demo3 extends JFrame {

    private JPanel panelUtama;
    private JButton buttonMulaiTask;
    private JLabel labelStatus;
    private JTextArea textAreaHasil;

    public Demo3() {
        initializeUI();
        setupEventHandlers();
    }

    private void initializeUI() {
        setTitle("Demo 3: SwingWorker (Static Nested Class)");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setPreferredSize(new Dimension(600, 450));

        panelUtama = new JPanel(
            new MigLayout("fill, wrap 1, insets 20")
        );

        buttonMulaiTask = new JButton("Mulai Task (8 detik)");
        buttonMulaiTask.setFont(new Font("Inter", Font.BOLD, 18));
        panelUtama.add(buttonMulaiTask, "growx, h 20%");
    }
}
```



```
labelStatus = new JLabel("Status: Siap", JLabel.CENTER);
labelStatus.setFont(new Font("Inter", Font.PLAIN, 22));
labelStatus.setForeground(new Color(0, 128, 0));
panelUtama.add(labelStatus, "growx, h 20%");

textAreaHasil = new JTextArea();
textAreaHasil.setEditable(false);
textAreaHasil.setFont(new Font("Consolas", Font.PLAIN, 15));
JScrollPane scroll = new JScrollPane(textAreaHasil);
panelUtama.add(scroll, "growx, h 60%");

add(panelUtama);
pack();
setLocationRelativeTo(null);
}

private void setupEventHandlers() {
    buttonMulaiTask.addActionListener(
        e -> new HitungMundurWorker(this).execute()
    );
}

private static class HitungMundurWorker extends SwingWorker<String,
Integer> {
    private final Demo3 frame;

    private HitungMundurWorker(Demo3 frame) {
        this.frame = frame;
    }

    @Override
    protected String doInBackground() throws Exception {
        frame.buttonMulaiTask.setEnabled(false);
        frame.labelStatus.setText("Status: Mulai bekerja");
        frame.textAreaHasil.setText("Mengerjakan Background Task (8
detik)\n");

        for (int i = 8; i >= 0; i--) {
            Thread.sleep(1000);
            publish(i);
        }
        return "Background Task Selesai";
    }

    @Override
    protected void process(List<Integer> chunks) {
        int detik = chunks.get(chunks.size() - 1);
```



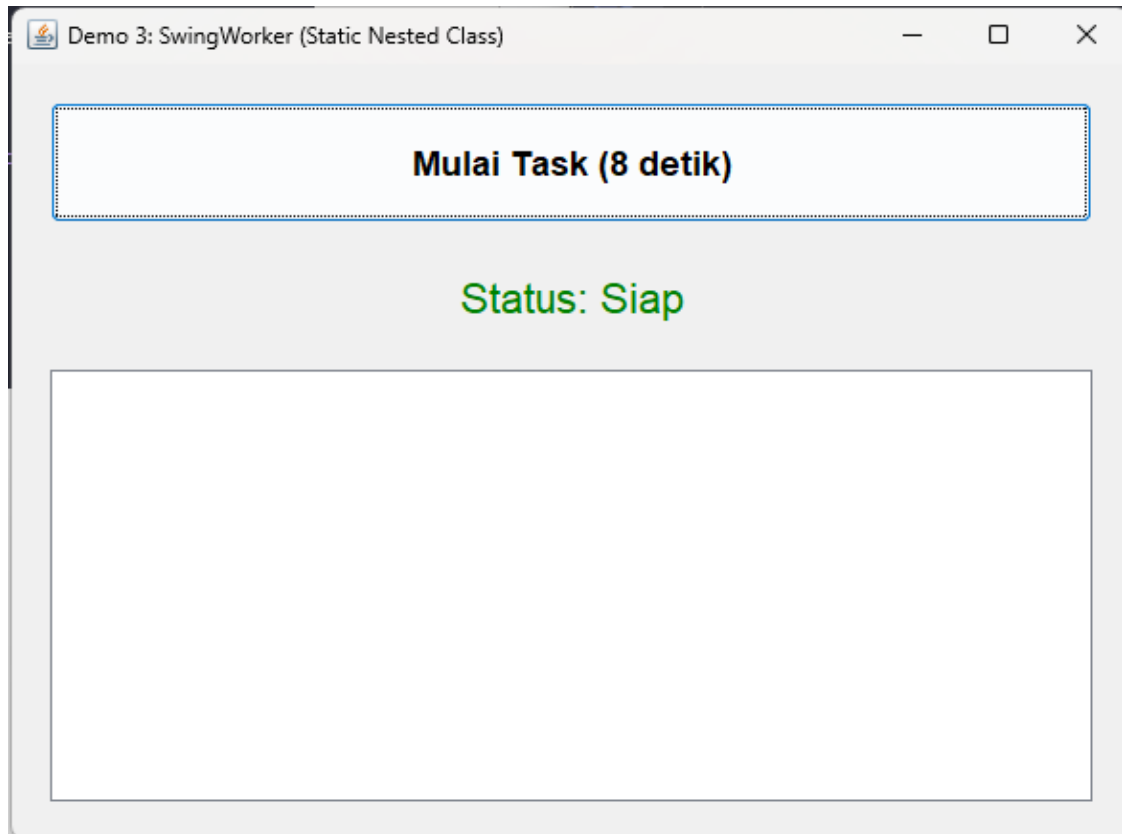
```
        frame.textAreaHasil.append("Hitung mundur: " + detik + "\n");
        frame.textAreaHasil.setCaretPosition(frame.textAreaHasil.getDocument().getLength());
        frame.labelStatus.setText("Sisa " + detik + " detik");
    }

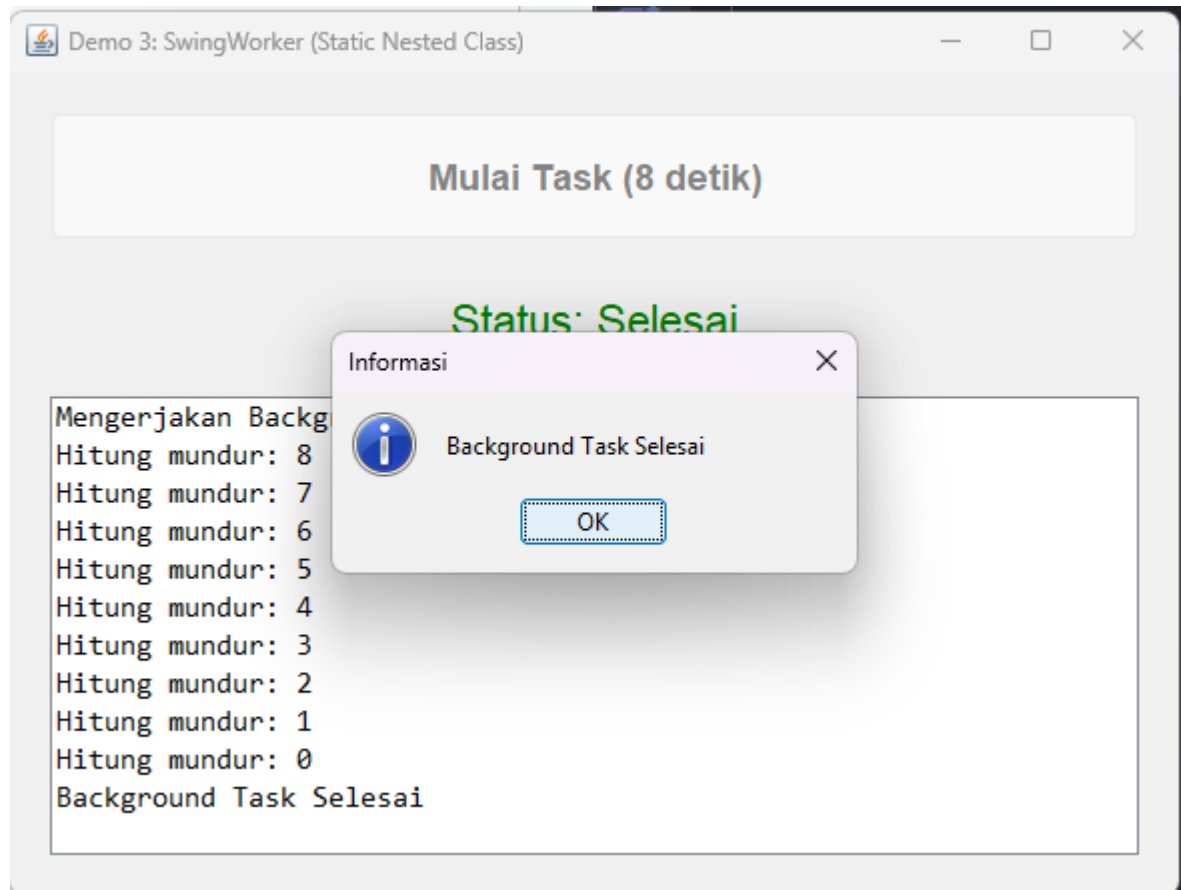
    @Override
    protected void done() {
        try {
            String hasil = get();
            frame.textAreaHasil.append(hasil);
            frame.labelStatus.setText("Status: Selesai");
            JOptionPane.showMessageDialog(frame, hasil, "Informasi",
JOptionPane.INFORMATION_MESSAGE);
        } catch (Exception ex) {
            frame.labelStatus.setText("Status: Error");
        } finally {
            frame.buttonMulaiTask.setEnabled(true);
        }
    }
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> {
        try {
            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
        } catch (Exception e) {
            System.err.println(e.getMessage());
        }
        new Demo3().setVisible(true);
    });
}
```



- Hasil Run







d. Demo 4

- Kode Program

```
package Praktikum;
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.SwingUtilities;
import javax.swing.UIManager;

import net.miginfocom.swing.MigLayout;

public class Demo4 extends JFrame {

    private JPanel panelUtama;
    private JButton buttonMulaiTask;
    private JLabel labelStatus;
    private JTextArea textAreaHasil;

    public Demo4() {
        initializeUI();
        setupEventHandlers();
    }

    private void initializeUI() {
        setTitle("Demo 4: SwingWorker (Separate Top-Level Class)");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setPreferredSize(new Dimension(600, 450));

        panelUtama = new JPanel(new MigLayout("fill, wrap 1, insets 20"));

        buttonMulaiTask = new JButton("Mulai Task (8 detik)");
        buttonMulaiTask.setFont(new Font("Inter", Font.BOLD, 18));
        panelUtama.add(buttonMulaiTask, "growx, h 20%");

        labelStatus = new JLabel("Status: Siap", JLabel.CENTER);
        labelStatus.setFont(new Font("Inter", Font.PLAIN, 22));
        labelStatus.setForeground(new Color(0, 128, 0));
        panelUtama.add(labelStatus, "growx, h 20%");
    }
}
```



```
        textAreaHasil = new JTextArea();
        textAreaHasil.setEditable(false);
        textAreaHasil.setFont(new Font("Consolas", Font.PLAIN, 15));
        JScrollPane scroll = new JScrollPane(textAreaHasil);
        panelUtama.add(scroll, "growx, h 60%");

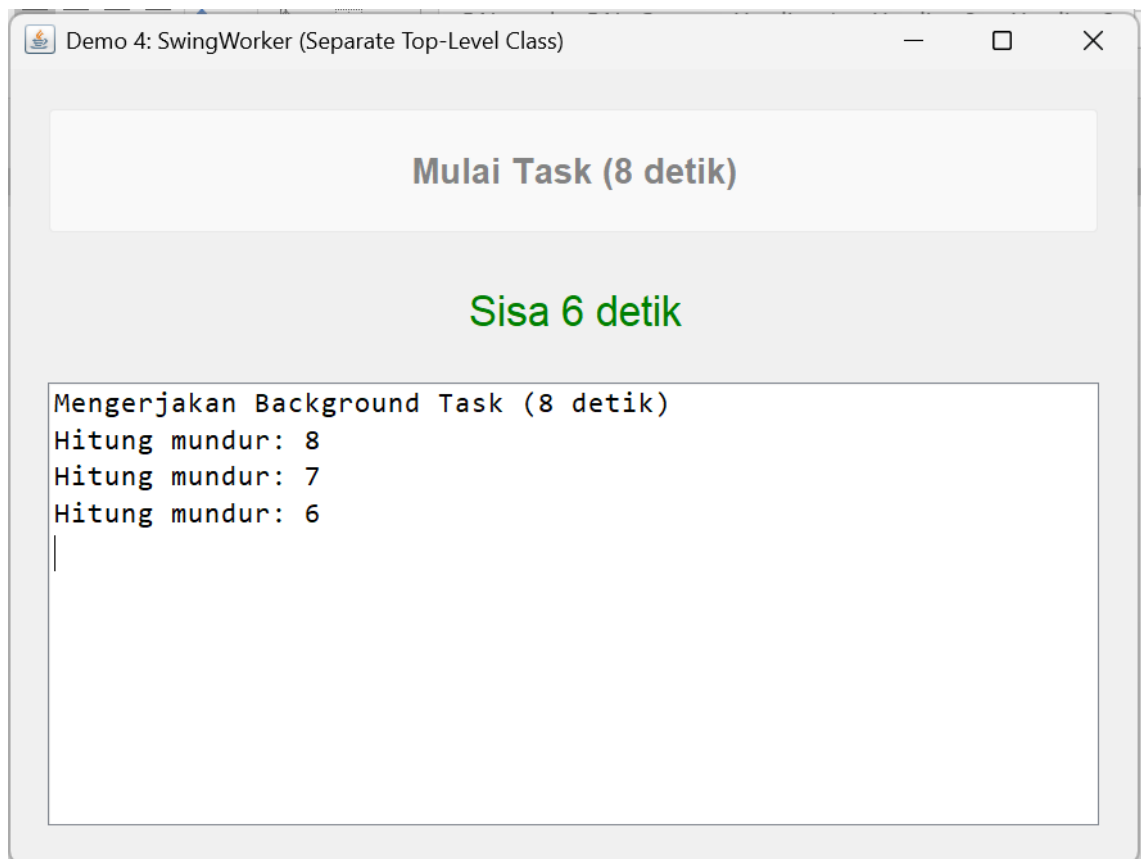
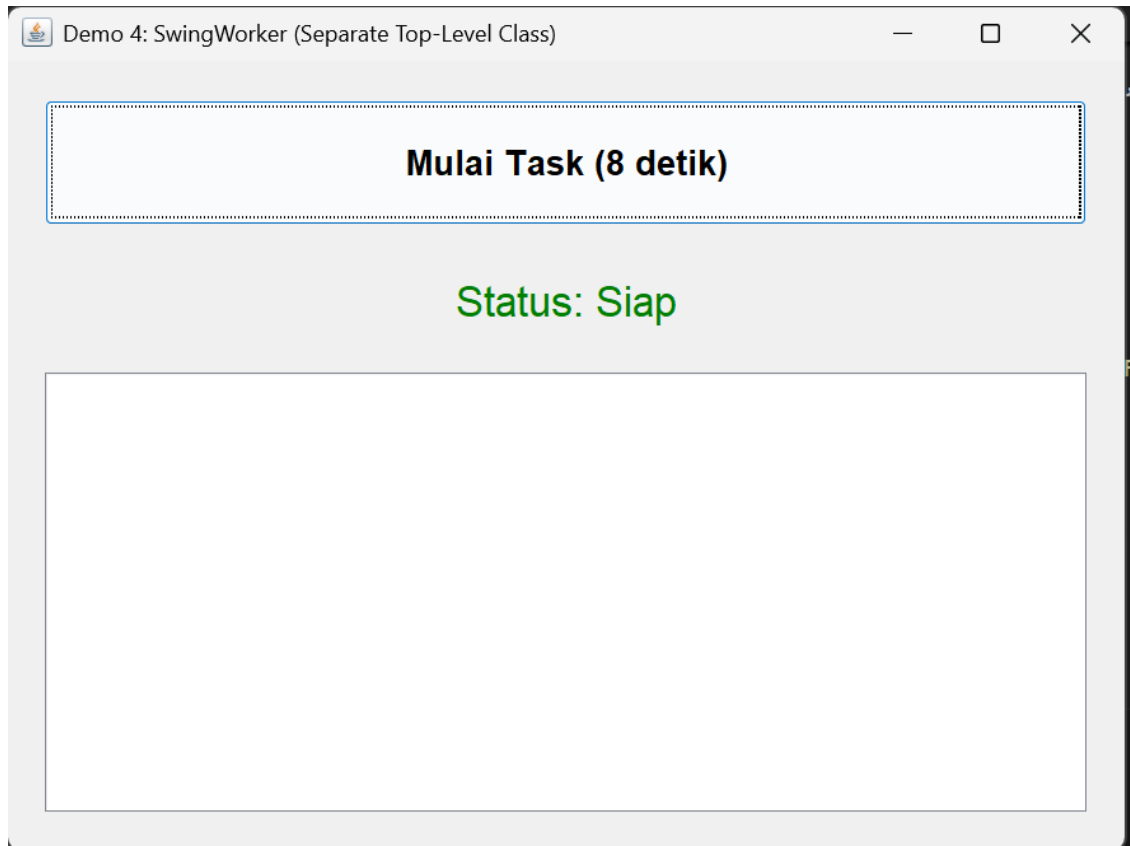
        add(panelUtama);
        pack();
        setLocationRelativeTo(null);
    }

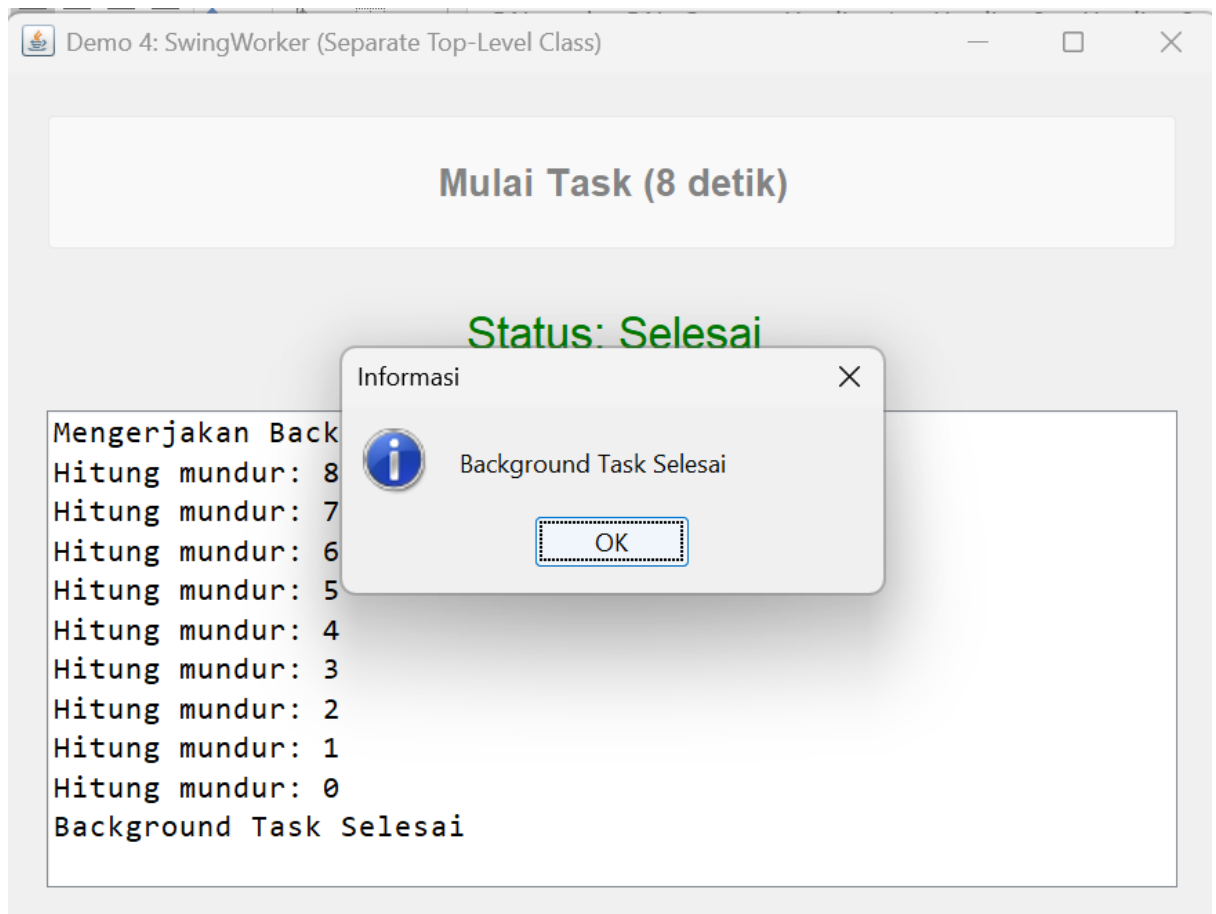
    private void setupEventHandlers() {
        buttonMulaiTask.addActionListener(e -> {
            buttonMulaiTask.setEnabled(false);
            new HitungMundurWorker(this, labelStatus, textAreaHasil,
buttonMulaiTask).execute();
        });
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            try {
                UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
            } catch (Exception e) {
                System.err.println(e.getMessage());
            }
            new Demo4().setVisible(true);
        });
    }
}
```



- Hasil Run







e. Hitung Mundur Worker

- Kode Program

```
package Praktikum;
import java.util.List;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JTextArea;
import javax.swing.SwingWorker;

public class HitungMundurWorker extends SwingWorker<String, Integer> {
    private final JFrame frame;
    private final JLabel labelStatus;
    private final JTextArea textAreaHasil;
    private final JButton buttonMulaiTask;

    public HitungMundurWorker(JFrame frame, JLabel labelStatus, JTextArea
textAreaHasil, JButton buttonMulaiTask) {
        this.frame = frame;
        this.labelStatus = labelStatus;
        this.textAreaHasil = textAreaHasil;
        this.buttonMulaiTask = buttonMulaiTask;
    }

    @Override
    protected String doInBackground() throws Exception {
        buttonMulaiTask.setEnabled(false);
        labelStatus.setText("Status: Mulai bekerja");
        textAreaHasil.setText("Mengerjakan Background Task (8 detik)\n");

        for (int i = 8; i >= 0; i--) {
            Thread.sleep(1000);
            publish(i);
        }
        return "Background Task Selesai";
    }

    @Override
    protected void process(List<Integer> chunks) {
        int detik = chunks.get(chunks.size() - 1);
        textAreaHasil.append("Hitung mundur: " + detik + "\n");
        textAreaHasil.setCaretPosition(textAreaHasil.getDocument().getLength()
);
        labelStatus.setText("Sisa " + detik + " detik");
    }
}
```



```
}

@Override
protected void done() {
    try {
        String hasil = get();
        textAreaHasil.append(hasil);
        labelStatus.setText("Status: Selesai");
        JOptionPane.showMessageDialog(frame, hasil, "Informasi",
JOptionPane.INFORMATION_MESSAGE);
    } catch (Exception ex) {
        labelStatus.setText("Status: Error");
    } finally {
        buttonMulaiTask.setEnabled(true);
    }
}
}
```



f. Demo 5

- Kode Program

```
package Praktikum;
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import java.util.List;
import java.util.concurrent.CancellationException;
import java.util.concurrent.ExecutionException;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JProgressBar;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.SwingUtilities;
import javax.swing.SwingWorker;
import javax.swing.UIManager;

import net.miginfocom.swing.MigLayout;

public class Demo5 extends JFrame {

    private JPanel panelUtama;
    private JButton buttonMulaiTask;
    private JButton buttonCancel;
    private JLabel labelStatus;
    private JProgressBar progressBar;
    private JTextArea textAreaHasil;
    private SwingWorker<Integer, Integer> worker;

    public Demo5() {
        initializeUI();
        setupEventHandlers();
    }

    private void initializeUI() {
        setTitle("Demo 5: SwingWorker + Cancellation");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setPreferredSize(new Dimension(720, 520));

        panelUtama = new JPanel(
            new MigLayout("fill, wrap 1, insets 20")
```



```
);
addPanelTombol();
addLabelStatus();
addProgressBar();
addScrollTextArea();

add(panelUtama);
pack();
setLocationRelativeTo(null);
}

private void addPanelTombol() {
    JPanel panelTombol = new JPanel(
        new MigLayout("fill", "[grow][grow]")
    );
    buttonMulaiTask = new JButton("Mulai Task (8 detik)");
    buttonCancel = new JButton("Cancel");
    buttonMulaiTask.setFont(new Font("Inter", Font.BOLD, 18));
    buttonCancel.setFont(new Font("Inter", Font.BOLD, 18));
    buttonCancel.setEnabled(false);
    panelTombol.add(buttonMulaiTask, "w 90%, growy");
    panelTombol.add(buttonCancel, "w 10%, growy");
    panelUtama.add(panelTombol, "h 20%, hmin pref, grow");
}

private void addLabelStatus() {
    labelStatus = new JLabel("Status: Siap", JLabel.CENTER);
    labelStatus.setFont(new Font("Inter", Font.BOLD, 24));
    labelStatus.setForeground(new Color(0, 100, 0));
    panelUtama.add(labelStatus, "h 10%, hmin pref, growx");
}

private void addProgressBar() {
    progressBar = new JProgressBar(0, 100);
    progressBar.setStringPainted(true);
    progressBar.setFont(new Font("Inter", Font.BOLD, 18));
    progressBar.setValue(0);
    panelUtama.add(progressBar, "h 10%, hmin pref, growx");
}

private void addScrollTextArea() {
    textAreaHasil = new JTextArea();
    textAreaHasil.setEditable(false);
    textAreaHasil.setFont(new Font("Consolas", Font.PLAIN, 14));
    JScrollPane scroll = new JScrollPane(textAreaHasil);
    panelUtama.add(scroll, "h 60%, growx");
}
```



```
private void setupEventHandlers() {
    buttonMulaiTask.addActionListener(e -> mulaiProses());
    buttonCancel.addActionListener(e -> {
        if (worker != null && !worker.isDone()) {
            worker.cancel(true);
            addLog("Proses dibatalkan oleh user...");
            buttonCancel.setEnabled(false);
            buttonMulaiTask.setEnabled(true);
        }
    });
}

private void mulaiProses() {
    buttonMulaiTask.setEnabled(false);
    buttonCancel.setEnabled(true);
    progressBar.setValue(0);
    textAreaHasil.setText("");
    labelStatus.setText("Status: Mulai bekerja");
    addLog("Mengerjakan Background Task (8 detik)");

    worker = new SwingWorker<Integer, Integer>() {
        @Override
        protected Integer doInBackground() throws Exception {
            int total = 100;
            for (int i = 1; i <= total; i++) {
                if (isCancelled()) return null;
                Thread.sleep(80);
                publish(i);
            }
            return total;
        }

        @Override
        protected void process(List<Integer> chunks) {
            if (isCancelled()) return;
            int progress = chunks.get(chunks.size() - 1);
            progressBar.setValue(progress);
            int remainingSeconds = 8 - (8 * progress / 100);
            labelStatus.setText("Sisa " + remainingSeconds + " detik");
            addLog("Persentase pengerjaan task: " + progress + "%");
        }

        @Override
        protected void done() {
            try {
                Integer result = get();
            }
        }
    };
}
```



```
        labelStatus.setText("Status: Selesai");
        addLog("Background Task Selesai " + result + "%");
        JOptionPane.showMessageDialog(Demo5.this,
            "Background Task Selesai",
            "Informasi", JOptionPane.INFORMATION_MESSAGE);
    } catch (ExecutionException ex) {
        labelStatus.setText("Status: Error");
        addLog("Error: " + ex.getCause().getMessage());
        JOptionPane.showMessageDialog(Demo5.this,
            "Terjadi kesalahan: " + ex.getMessage(),
            "Error", JOptionPane.ERROR_MESSAGE);
    } catch (CancellationException ee){
        labelStatus.setText("Status: Dibatalkan");
    } catch (InterruptedException e) {
        Thread.currentThread().interrupt();
    } finally {
        buttonCancel.setEnabled(false);
        buttonMulaiTask.setEnabled(true);
        worker = null;
    }
}

};

worker.execute();
}

private void addLog(String text) {
    textAreaHasil.append(text + "\n");
    textAreaHasil.setCaretPosition(textAreaHasil.getDocument().getLength());
};

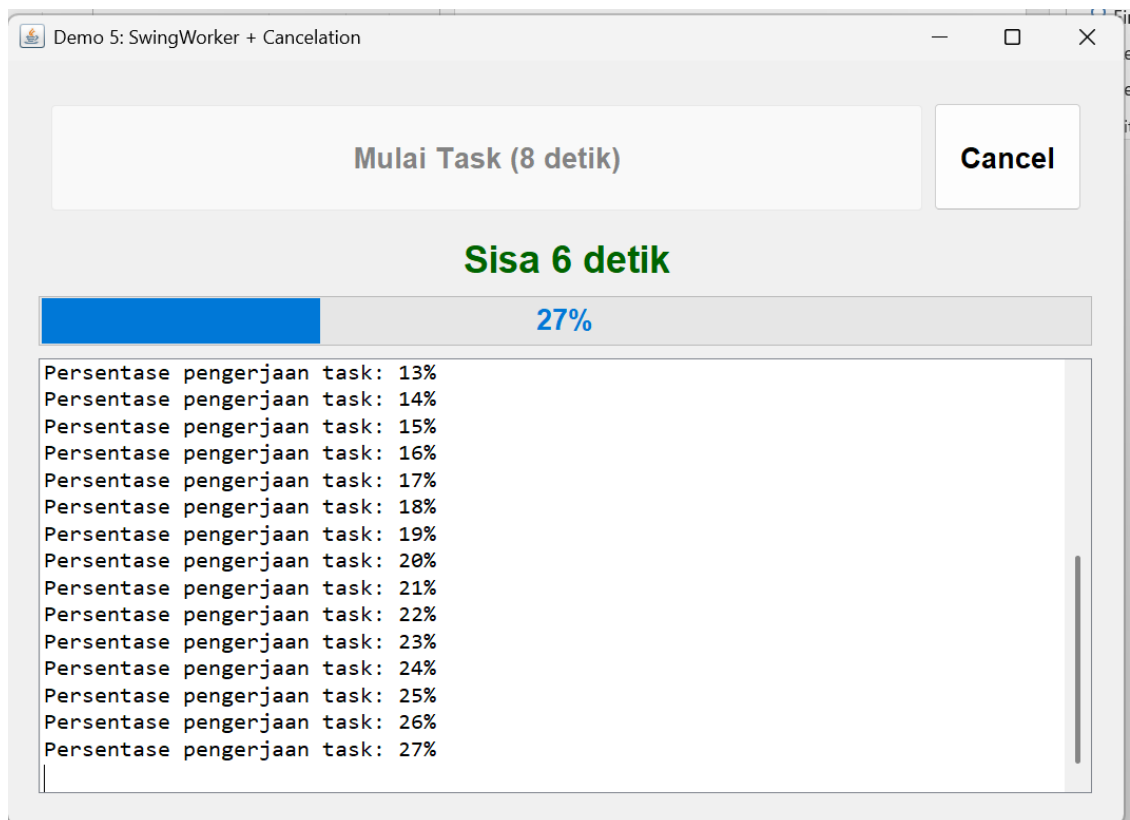
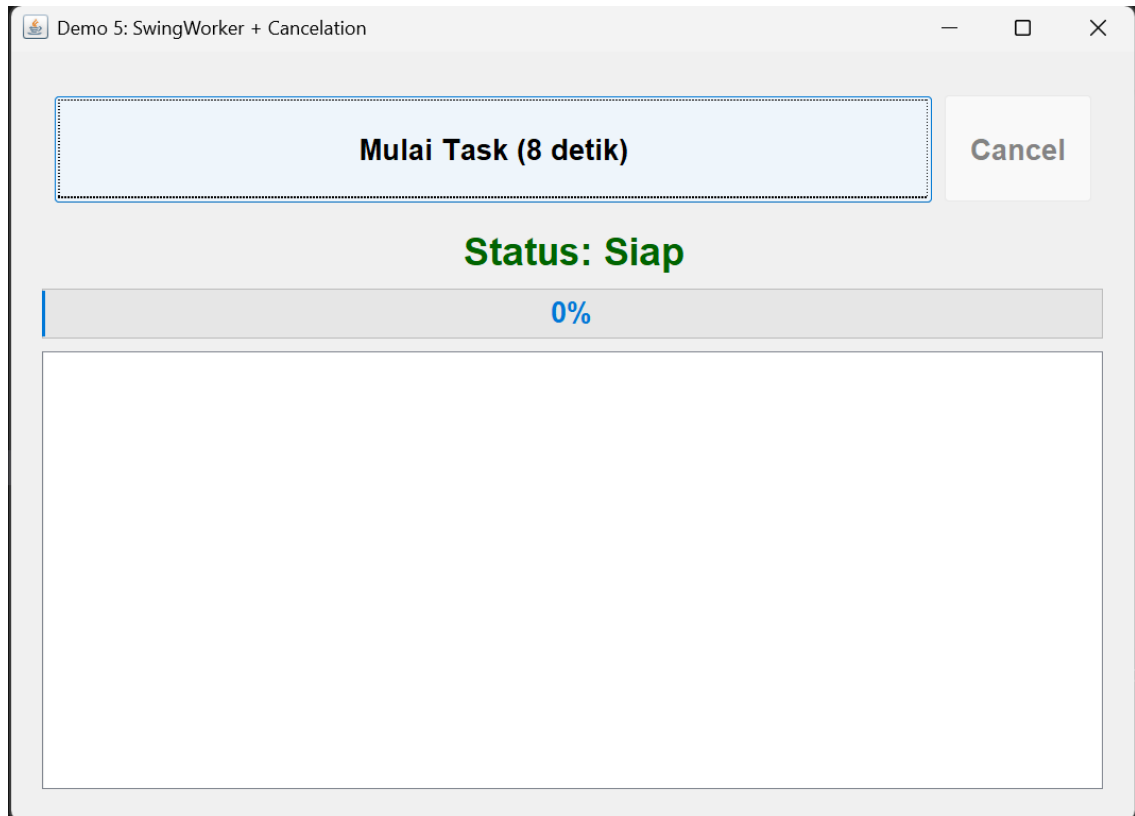
}

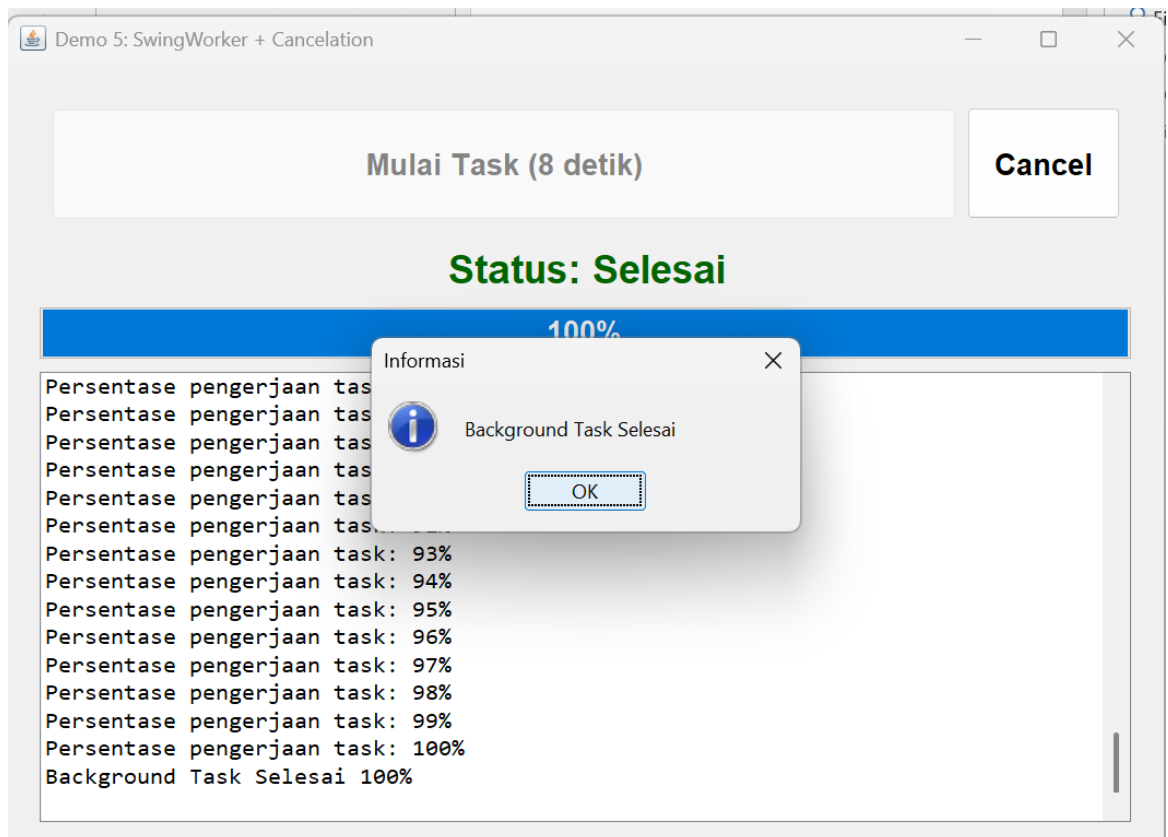
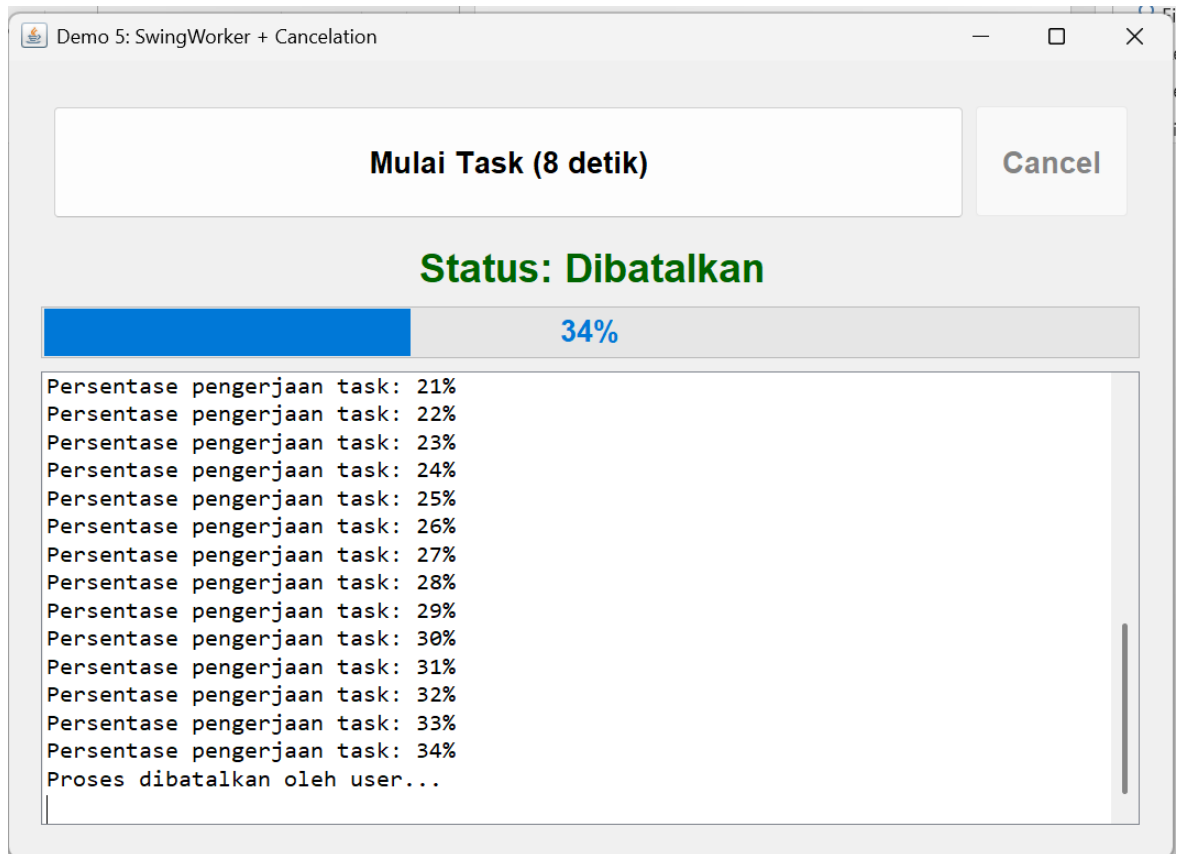
public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> {
        try {
            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
        } catch (Exception ignored) {}
        new Demo5().setVisible(true);
    });
}

}
```



- Hasil Run







3. LATIHAN

a. Task Worker

- Kode Program

```
package Latihan;

import java.util.List;
import java.util.concurrent.CancellationException;
import java.util.concurrent.ExecutionException;

import javax.swing.SwingWorker;

public class TaskWorker extends SwingWorker<Integer, Integer> {

    public interface Listener {
        void onProgressUpdate(int progress, int remainingSeconds);
        void onLog(String msg);
        void onCompleted(int result);
        void onCancelled();
        void onError(Exception ex);
    }

    private final Listener listener;

    public TaskWorker(Listener listener) {
        this.listener = listener;
    }

    @Override
    protected Integer doInBackground() throws Exception {
        int total = 100;
        for (int i = 1; i <= total; i++) {
            if (isCancelled()) return null;

            Thread.sleep(80);
            publish(i);
        }
        return total;
    }

    @Override
    protected void process(List<Integer> chunks) {
        if (isCancelled()) return;

        int progress = chunks.get(chunks.size() - 1);
        int remainingSeconds = 8 - (8 * progress / 100);
```



```
        listener.onProgressUpdate(progress, remainingSeconds);
        listener.onLog("Persentase pengerjaan task: " + progress + "%");
    }

    @Override
    protected void done() {
        try {
            Integer result = get();
            listener.onCompleted(result);
        } catch (CancellationException e) {
            listener.onCancelled();
        } catch (ExecutionException e) {
            listener.onError(e);
        } catch (InterruptedException e) {
            Thread.currentThread().interrupt();
        }
    }
}
```



b. Demo 6

- Kode Program

```
package Latihan;

import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JProgressBar;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.SwingUtilities;
import javax.swing.UIManager;

import net.miginfocom.swing.MigLayout;

public class Demo6 extends JFrame implements TaskWorker.Listener {

    private JPanel panelUtama;
    private JButton buttonMulaiTask;
    private JButton buttonCancel;
    private JLabel labelStatus;
    private JProgressBar progressBar;
    private JTextArea textAreaHasil;

    private TaskWorker worker;

    public Demo6() {
        initializeUI();
        setupEventHandlers();
    }

    private void initializeUI() {
        setTitle("Demo 6: SwingWorker Toplevel + Cancelation");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setPreferredSize(new Dimension(720, 520));

        panelUtama = new JPanel(new MigLayout("fill, wrap 1, insets 20"));

        addPanelTombol();
        addLabelStatus();
    }
}
```



```
        addProgressBar();
        addScrollTextArea();

        add(panelUtama);
        pack();
        setLocationRelativeTo(null);
    }

    private void addPanelTombol() {
        JPanel panelTombol = new JPanel(new MigLayout("fill",
"[grow][grow]"));
        buttonMulaiTask = new JButton("Mulai Task (8 detik)");
        buttonCancel = new JButton("Cancel");

        buttonMulaiTask.setFont(new Font("Inter", Font.BOLD, 18));
        buttonCancel.setFont(new Font("Inter", Font.BOLD, 18));
        buttonCancel.setEnabled(false);

        panelTombol.add(buttonMulaiTask, "w 90%, growy");
        panelTombol.add(buttonCancel, "w 10%, growy");

        panelUtama.add(panelTombol, "h 20%, grow");
    }

    private void addLabelStatus() {
        JLabel labelStatus = new JLabel("Status: Siap", JLabel.CENTER);
        labelStatus.setFont(new Font("Inter", Font.BOLD, 24));
        labelStatus.setForeground(new Color(0, 100, 0));
        panelUtama.add(labelStatus, "h 10%, growx");
    }

    private void addProgressBar() {
        progressBar = new JProgressBar(0, 100);
        progressBar.setStringPainted(true);
        progressBar.setFont(new Font("Inter", Font.BOLD, 18));
        panelUtama.add(progressBar, "h 10%, growx");
    }

    private void addScrollTextArea() {
        JTextArea textAreaHasil = new JTextArea();
        textAreaHasil.setEditable(false);
        textAreaHasil.setFont(new Font("Consolas", Font.PLAIN, 14));
        JScrollPane scroll = new JScrollPane(textAreaHasil);

        panelUtama.add(scroll, "h 60%, grow");
    }
}
```



```
private void setupEventHandlers() {

    buttonMulaiTask.addActionListener(e -> mulaiProses());

    buttonCancel.addActionListener(e -> {
        if (worker != null && !worker.isDone()) {
            worker.cancel(true);
            addLog("Proses dibatalkan oleh user...");
        }
    });
}

private void mulaiProses() {
    buttonMulaiTask.setEnabled(false);
    buttonCancel.setEnabled(true);
    progressBar.setValue(0);
    textAreaHasil.setText("");

    labelStatus.setText("Status: Mulai bekerja");
    addLog("Mengerjakan Background Task (8 detik)");

    worker = new TaskWorker(this);
    worker.execute();
}

private void addLog(String msg) {
    textAreaHasil.append(msg + "\n");
    textAreaHasil.setCaretPosition(textAreaHasil.getDocument().getLength());
};

}

@Override
public void onProgressUpdate(int progress, int remainingSeconds) {
    progressBar.setValue(progress);
    labelStatus.setText("Sisa " + remainingSeconds + " detik");
}

@Override
public void onLog(String msg) {
    addLog(msg);
}

@Override
public void onCompleted(int result) {
    labelStatus.setText("Status: Selesai");
    addLog("Background Task Selesai " + result + "%");
}
```



```
JOptionPane.showMessageDialog(this,
    "Background Task Selesai",
    "Informasi",
    JOptionPane.INFORMATION_MESSAGE);

resetButtons();
}

@Override
public void onCancelled() {
    labelStatus.setText("Status: Dibatalkan");
    addLog("Task dibatalkan.");
    resetButtons();
}

@Override
public void onError(Exception ex) {
    labelStatus.setText("Status: Error");
    addLog("Error: " + ex.getMessage());

    JOptionPane.showMessageDialog(this,
        "Terjadi kesalahan: " + ex.getMessage(),
        "Error",
        JOptionPane.ERROR_MESSAGE);

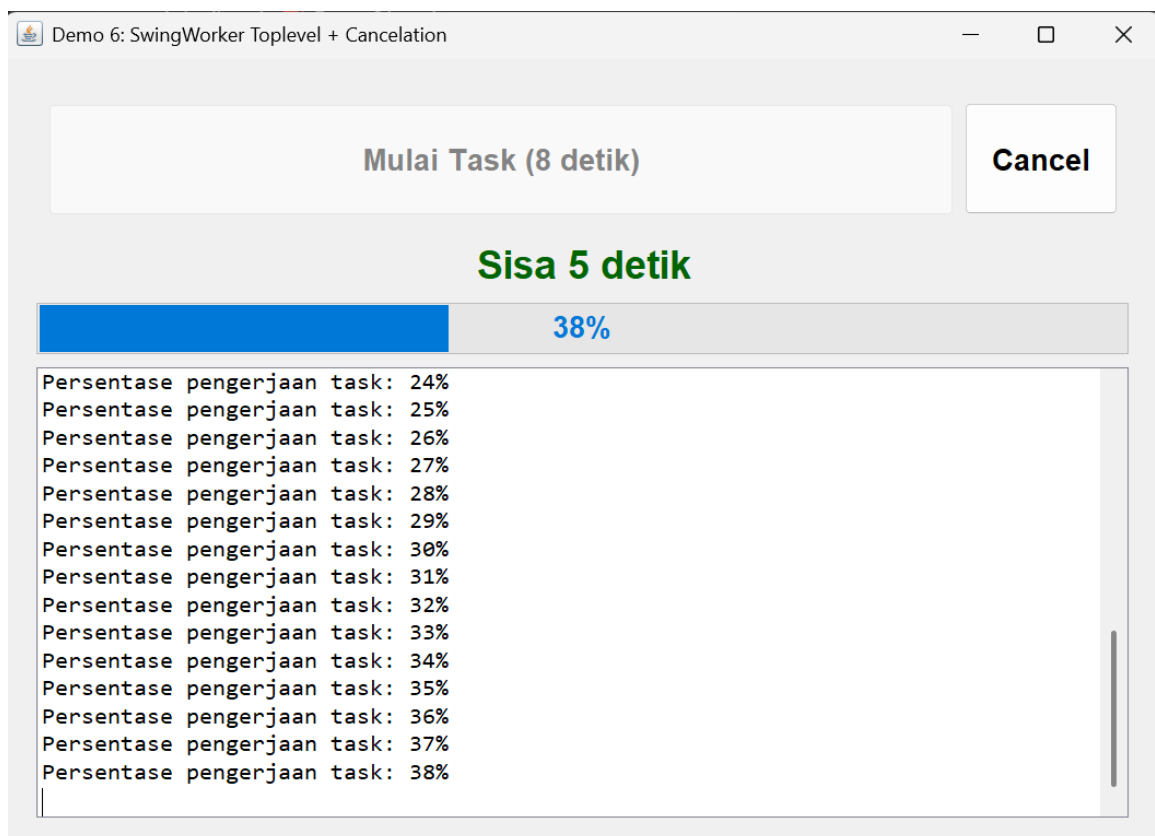
    resetButtons();
}

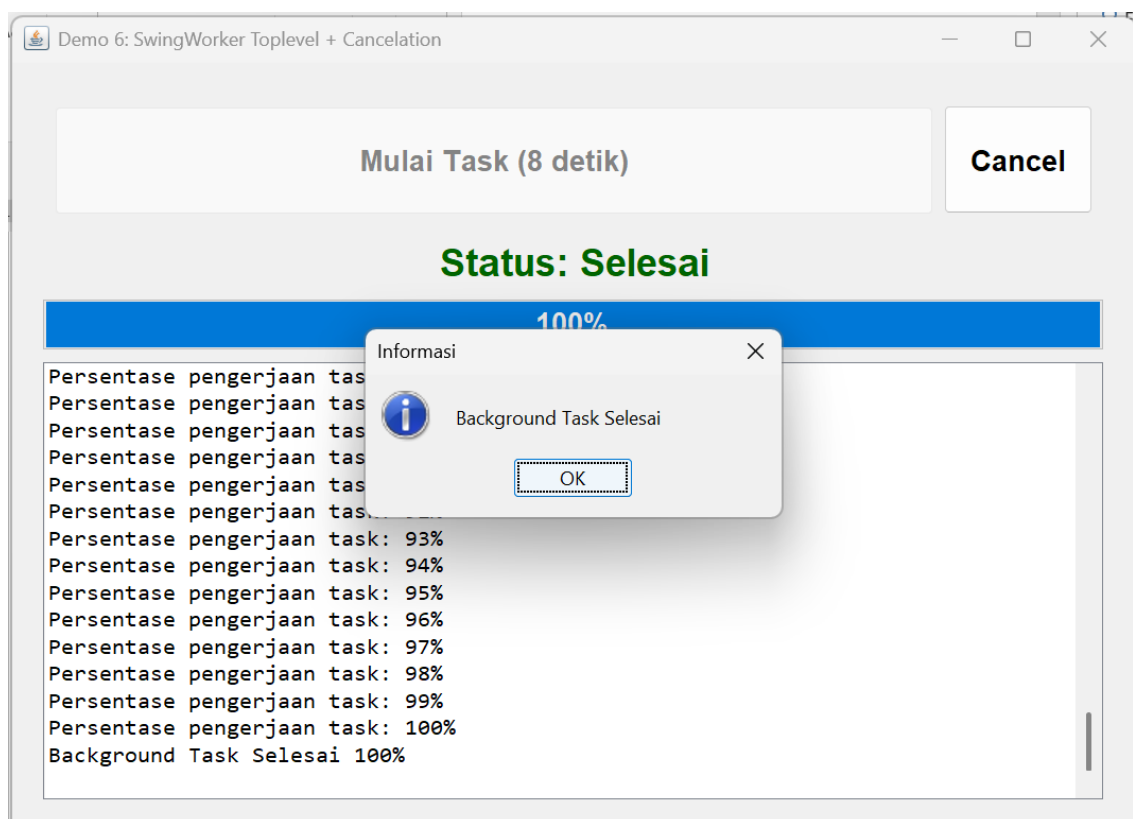
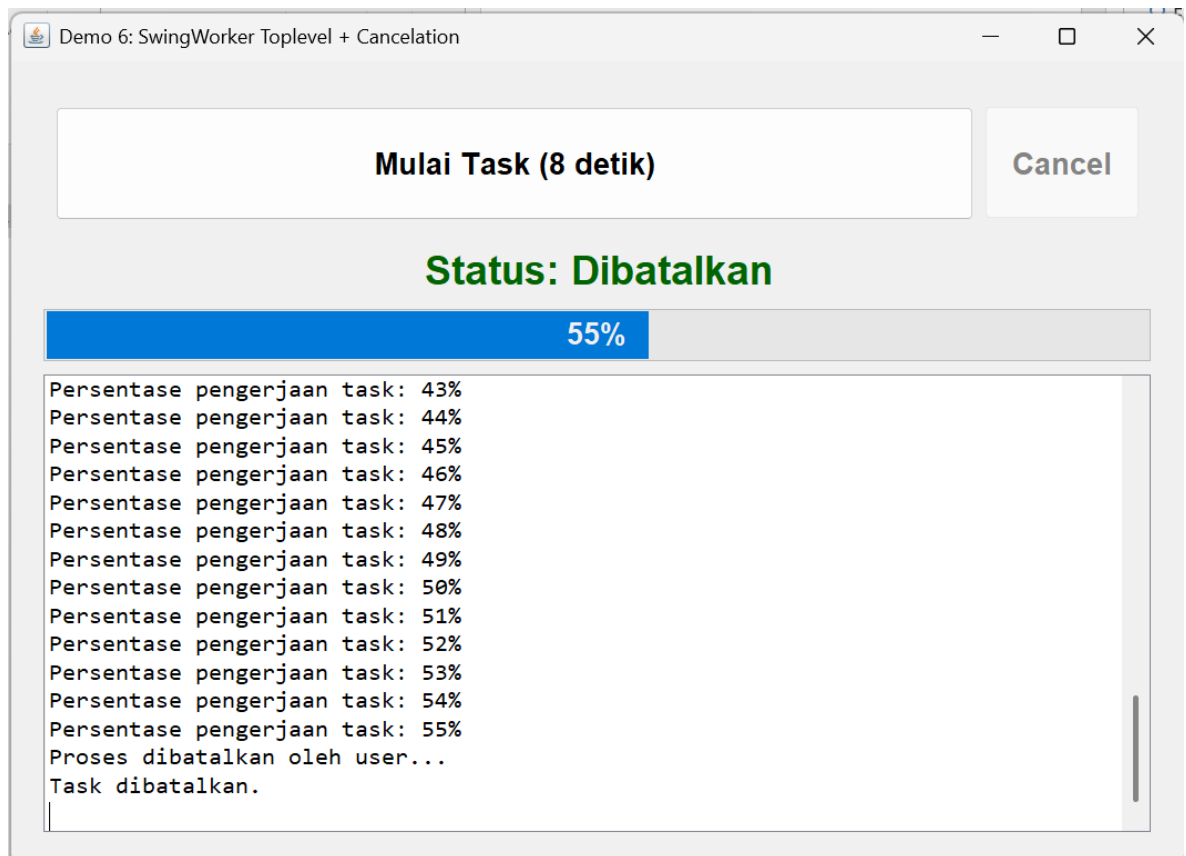
private void resetButtons() {
    buttonMulaiTask.setEnabled(true);
    buttonCancel.setEnabled(false);
    worker = null;
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> {
        try {
            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
        } catch (Exception ignored) {}
        new Demo6().setVisible(true);
    });
}
```



- Hasil Run







C. PENGALAMAN PEMBELAJARAN

Pada pengalaman pembelajaran kali ini, kita belajar bagaimana suatu perangkat melakukan operasi yang berat dapat membuat freeze, namun segala sesuatunya pasti ada jalan keluarnya. Kemudian mempelajari Concurrency yang mana dapat melakukan eksekusi secara bersamaan.

Dalam pengerjaan praktikum agak sedikit bingung, pada bagian Demo2 – Demo4, dikarenakan hasil Outputnya hampir tidak ada bedanya. Sehingga pada pengerjaan tugas pun, Output dari Demo5 dan Demo6 (Latihan) hampir sama.