**CODING**

**DIGITAL CLOCK**

import java.awt.\*;

import java.awt.event.\*;

import java.text.SimpleDateFormat;

import java.util.Date;

public class DigitalClockAWT extends Frame {

private Label timeLabel, dateLabel, locationLabel, temperatureLabel;

private Button format12Button, format24Button;

private boolean is24HourFormat = false;

public DigitalClockAWT() {

// Set the title and layout for the frame

setTitle("Digital Clock with Extras");

setLayout(new GridLayout(5, 1)); // Organize components in 5 rows

// Initialize labels

timeLabel = new Label();

dateLabel = new Label();

locationLabel = new Label("Location: Trichy"); // Example location

temperatureLabel = new Label("Temperature: 20°C"); // Example temperature

// Set fonts for better appearance

timeLabel.setFont(new Font("Arial", Font.PLAIN, 48));

dateLabel.setFont(new Font("Arial", Font.PLAIN, 24));

locationLabel.setFont(new Font("Arial", Font.PLAIN, 18));

temperatureLabel.setFont(new Font("Arial", Font.PLAIN, 18));

// Initialize buttons

format12Button = new Button("12-Hour Format");

format24Button = new Button("24-Hour Format");

// Add components to the frame

add(dateLabel); // Date

add(timeLabel); // Time

add(locationLabel); // Location

add(temperatureLabel); // Temperature

add(format12Button); // Button for 12-hour format

add(format24Button); // Button for 24-hour format

// Button actions

format12Button.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

is24HourFormat = false; // Switch to 12-hour format

}

});

format24Button.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

is24HourFormat = true; // Switch to 24-hour format

}

});

// Set the size and close operation

setSize(500, 400);

setVisible(true);

// Ensure the frame closes properly

addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent we) {

System.exit(0);

}

});

// Start the clock thread

startClock();

}

public void startClock() {

// Clock update every second

Thread clockThread = new Thread() {

public void run() {

while (true) {

updateClock();

try {

Thread.sleep(1000); // Wait for 1 second before updating the clock

} catch (InterruptedException e) {

System.out.println("Thread interrupted");

}

}

}

};

clockThread.start();

}

public void updateClock() {

// Get current system time and date

Date currentTime = new Date();

SimpleDateFormat timeFormat;

SimpleDateFormat dateFormat = new SimpleDateFormat("EEEE, MMMM dd, yyyy"); // e.g., Friday, November 22, 2024

// Use 12-hour or 24-hour format based on user selection

if (is24HourFormat) {

timeFormat = new SimpleDateFormat("HH:mm:ss");

} else {

timeFormat = new SimpleDateFormat("hh:mm:ss a");

}

// Update labels

timeLabel.setText("Time: " + timeFormat.format(currentTime));

dateLabel.setText("Date: " + dateFormat.format(currentTime));

// Update location and temperature (static for this example)

// To dynamically fetch temperature, use an API like OpenWeatherMap or WeatherAPI.

locationLabel.setText("Location: New York");

temperatureLabel.setText("Temperature: 20°C"); // Hardcoded temperature

}

public static void main(String[] args) {

// Create an instance of the clock GUI

new DigitalClockAWT();

}

}