

DIGITAL CLOCK

A PROJECT REPORT

Submitted by

SHEIKH MOHAMMED M (2303811724321103)

in partial fulfillment of requirements for the award of the course

CGB1201 – JAVA PROGRAMMING

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by
AICTE, New Delhi)

SAMAYAPURAM – 621 112

DECEMBER, 2024

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on “**DIGITAL CLOCK**” is the bonafide work of **SHEIKH MOHAMMED M (2303811724321103)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

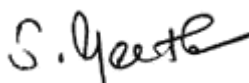


Signature

Dr. T. AVUDAIAPPAN M.E., Ph.D.,

HEAD OF THE DEPARTMENT,

Department of Artificial Intelligence,
K. Ramakrishnan College of Technology,
Samayapuram, Trichy -621 112.



Signature

Mrs. S. GEETHA M.E.,

SUPERVISOR,

Department of Artificial Intelligence,
K. Ramakrishnan College of Technology,
Samayapuram, Trichy -621 112.

Submitted for the viva-voce examination held on 3.12.24



INTERNAL EXAMINER



EXTERNAL EXAMINER

DECLARATION

I declare that the project report on “**DIGITAL CLOCK** ” is the result of original work done by me and best of my knowledge, similar work has not been submitted to “**ANNA UNIVERSITY CHENNAI**” for the requirement of Degree of **BACHELOR OF TECHNOLOGY**. This project report is submitted on the partial fulfillment of the requirement of the award of the **CGB1201 – JAVA PROGRAMMING**.



Signature

SHEIKH MOHAMMED M

Place: Samayapuram

Date: 3/12/2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and indebtedness to our institution, “**K. Ramakrishnan College of Technology (Autonomous)**”, for providing us with the opportunity to do this project.

I extend our sincere acknowledgement and appreciation to the esteemed and honourable Chairman, **Dr. K. RAMAKRISHNAN, B.E.**, for having provided the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director, **Dr. S. KUPPUSAMY, MBA, Ph.D.**, for forwarding our project and offering an adequate duration to complete it.

I would like to thank **Dr. N. VASUDEVAN, M.TECH., Ph.D.**, Principal, who gave the opportunity to frame the project to full satisfaction.

I thank **Dr.T.AVUDAIAPPAN, M.E.,Ph.D.**, Head of the Department of **ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**, for providing her encouragement in pursuing this project.

I wish to convey our profound and heartfelt gratitude to our esteemed project guide **Mrs.S.GEETHA M.E.**, Department of **ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**, for her incalculable suggestions, creativity, assistance and patience, which motivated us to carry out this project.

I render our sincere thanks to the Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards.

MISSION OF THE INSTITUTION

- Be a centre of excellence for technical education in emerging technologies by exceeding the needs of industry and society.
- Be an institute with world class research facilities.
- Be an institute nurturing talent and enhancing competency of students to transform them as all- round personalities respecting moral and ethical values.

VISION AND MISSION OF THE DEPARTMENT

To excel in education, innovation and research in Artificial Intelligence and Data Science to fulfill industrial demands and societal expectations.

Mission 1: To educate future engineers with solid fundamentals, continually improving teaching methods using modern tools.

Mission 2: To collaborate with industry and offer top-notch facilities in a conducive learning environment.

Mission 3: To foster skilled engineers and ethical innovation in AI and Data Science for global recognition and impactful research.

Mission 4: To tackle the societal challenge of producing capable professionals by instilling employability skills and human values.

PROGRAM EDUCATIONAL OBJECTIVES (PEOS)

PEO 1: Compete on a global scale for a professional career in Artificial Intelligence and Data Science.

PEO 2: Provide industry-specific solutions for the society with effective communication and ethics.

PEO 3: Hone their professional skills through research and lifelong learning initiatives.

PROGRAM OUTCOMES

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and

write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

- **PSO 1:** Capable of working on data-related methodologies and providing industry-focussed solutions.
- **PSO2:** Capable of analysing and providing a solution to a given real-world problem by designing an effective program.

ABSTRACT

A digital clock is a user-friendly device designed to display and manage time efficiently while hiding underlying technical complexities. It provides core features such as a real-time display of the current time in 12-hour or 24-hour formats with AM/PM indicators, alongside date and day information in customizable formats. Advanced functionalities include multiple alarms with snooze options, recurring schedules, and countdown timers. For large-scale applications, digital clocks can provide centralized synchronization across multiple devices in spaces like offices, schools, and airports, ensuring consistency. They are equipped with scalable features such as network connectivity via Wi-Fi or LAN, IoT compatibility, and APIs for seamless integration with smart home systems or organizational tools. The user interface is highly customizable, allowing adjustments in brightness, themes, fonts, and even branded designs. Hidden layers of hardware, like microcontrollers and quartz crystal oscillators, handle precise timekeeping, while software algorithms manage synchronization and error correction. Advanced integrations like voice assistant compatibility, weather updates, and notifications enhance functionality, making digital clocks versatile solutions for both individual and large-scale application

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No.
	ABSTRACT	viii
1	INTRODUCTION	1
	1.1 INTRODUCTION	1
	1.2 OBJECTIVE	1
	PROJECT METHODOLOGY	1
2	2.1 PROPOSED WORK	2
	2.2 BLOCK DIAGRAM	3
3	JAVA PROGRAMMING CONCEPTS	4
	3.1 CLASSES AND OBJECTS	4
	3.2 DATE AND TIME HANDLING	4
	3.3 LOOPS	4
	3.4 THREADING	4
4	MODULE DESCRIPTION	5
	4.1 JAVAX.SWING (SWING)	5
	4.2 JAVA.AWT (ABSTRACT WINDOW TOOLKIT)	5
	4.3 JAVA.TEXT (FORMATTING)	5
	4.4 JAVA.AWT.EVENT (EVENT HANDLING)	6
5	CONCLUSION	7
	REFERENCES	8

	APPENDICES	10
	APPENDIX-A SOURCE CODE	10
	APPENDIX-B SCREEN SHOT	16

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

A digital clock in Java displays the current time in real-time using classes like Date and Simple Date Format for formatting. It updates every second using a loop with Thread. sleep(). The clock can be extended with features like alarms, timers, or time zone support. Using Java Swing or JavaFX, the clock can have a graphical interface with customizable themes. Additionally, it can synchronize with time servers for accuracy, making it a versatile and practical application of Java's real-time programming capabilities.

1.2 OBJECTIVE

The objective of a digital clock is to provide accurate and user-friendly timekeeping by displaying time, date, and day in an easily readable format. It includes features like alarms, timers, and synchronization with standard time sources, ensuring precision and convenience. For large-scale applications, it offers centralized synchronization across networks for consistent time management in offices, schools, and public spaces. Advanced functionalities like IoT compatibility, weather updates, and voice assistant integration make it versatile for modern needs, balancing simplicity, reliability, and scalability.

CHAPTER 2

PROJECT METHODOLOGY

2.1 PROPOSED WORK

Start with the LCD Display:

The system begins with the LCD screen displaying time.

Increment Seconds:

The seconds counter is incremented by one.

Check if 60 Seconds Passed:

If the seconds reach 60, the minutes display is incremented, and the seconds reset to Zero

If not, it continues to increment seconds.

Increment Minutes:

If 60 minutes are reached, the hours display is incremented, and the minutes reset to zero.

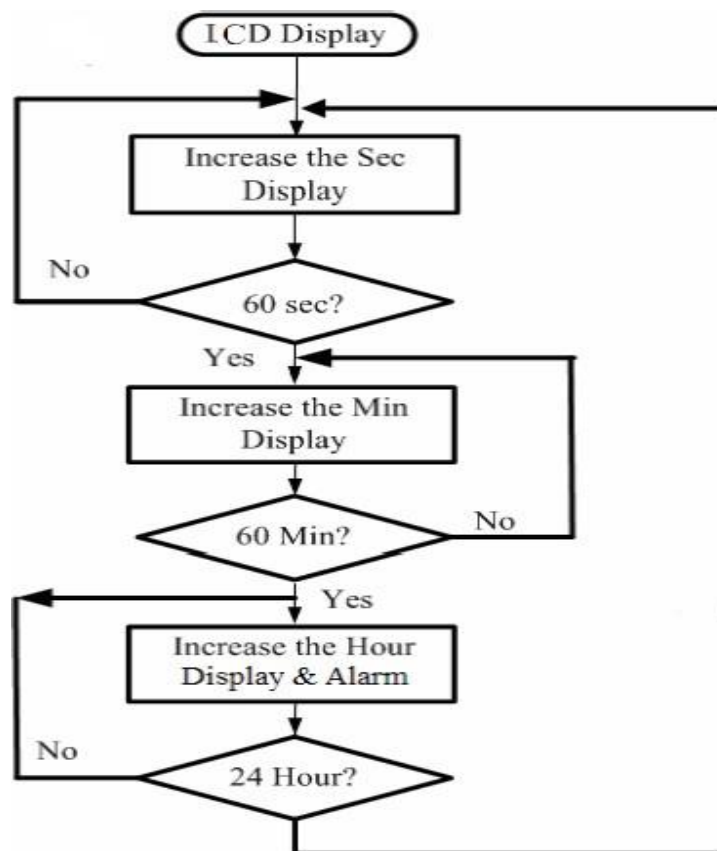
Check if 24 Hours Passed:

If 24 hours are reached, the display resets the hour counter to zero (possibly to account for a new day).

Loop Back:

The process continuously loops to update the time on the display in real-time.

2.2 BLOCK DIAGRAM



CHAPTER 3

JAVA PROGRAMMING CONCEPTS

3.1 CLASSES AND OBJECTS

Create a Digital Clock class to handle time-related functionality.

3.2 DATE AND TIME HANDLING

Use Date and Simple Date Format to retrieve and format the current time.

3.3 LOOPS

A loop updates the time every second using Thread.sleep(1000).

3.4 THREADING

Use multithreading to run the clock continuously without blocking the main program

CHAPTER 4

MODULE DESCRIPTION

4.1 JAVAX.SWING (SWING)

Purpose: Used for creating the graphical user interface (GUI).

Key Classes:

JFrame: Used for creating the main window for the clock.

JLabel: Displays the time and date in the window.

JButton: For adding buttons like "Toggle 12/24-hour format" and "Set Alarm".

4.2 JAVA.AWT (ABSTRACT WINDOW TOOLKIT)

Purpose: Provides graphical user interface components and event handling.

Key Classes:

BorderLayout: A layout manager for arranging components in five regions.

Font: Used for setting custom fonts to the time and date displays.

4.3 JAVA.TEXT (FORMATTING)

Purpose: Used for formatting and parsing dates and times.

Key Classes:

SimpleDateFormat: Used for formatting the time and date in a user-friendly format (e.g., HH:mm:ss for time, dd MMM yyyy for date).

4.4 JAVA.AWT.EVENT (EVENT HANDLING)

Purpose: Provides classes to handle user input and events.

Key Classes:

ActionListener: Listens for user actions like clicking a button to toggle the time format or set the alarm.

CHAPTER 5

CONCLUSION

In conclusion, building a digital clock application in Java demonstrates the seamless integration of multiple packages to deliver a functional and interactive user experience. The **javax.swing** package provides the core components for the graphical user interface, ensuring the application is visually appealing and user-friendly. Complementary packages like **java.awt** enhance the layout and aesthetics, while **java.text** ensures the accurate and user-friendly formatting of time and date. The **java.util** package simplifies time management, and **java.awt.event** enables efficient event handling for interactive features like toggling time formats or setting alarms. Together, these modules empower developers to create a versatile and robust digital clock that not only displays real-time information but also provides additional functionalities to user needs.

REFERENCES:

1. JAVA: THE COMPLETE REFERENCE" BY HERBERT SCHILDT

- This book covers Java fundamentals and advanced topics, including GUI programming with Swing.

2. HEAD FIRST JAVA" BY KATHY SIERRA AND BERT BATES

- A beginner-friendly book that introduces Java concepts in an engaging way, including GUI applications.

3. "JAVA SWING" BY MARC LOY, ROBERT ECKSTEIN, AND DAVE WOOD

- A comprehensive guide to Swing, which is the GUI toolkit used in the digital clock example.

ONLINE TUTORIALS AND DOCUMENTATION

1. Oracle's Java Tutorials

- Java Tutorials by Oracle
- This is the official documentation and tutorials provided by Oracle, covering various aspects of Java programming, including Swing.

2. Java Swing Tutorial

- Java Swing Tutorial
- A detailed tutorial on Swing components and how to use them to build GUI applications.

3. GeeksforGeeks Java Swing

- GeeksforGeeks Java Swing
- Offers explanations, examples, and projects related to Java Swing.

4. W3Schools Java GUI

- W3Schools Java GUI
- A beginner-friendly resource that covers basic Java GUI concepts.

VIDEO TUTORIALS

1. YouTube

- Search for "Java Swing Tutorial" or "Java GUI Programming" on YouTube for numerous video tutorials that walk you through creating GUI applications in Java.

2. Udemy Courses

- Look for courses on Udemy that focus on Java GUI programming or Swing. These often include hands-on projects.

CODE REPOSITORIES

1. GitHub

- Search GitHub for repositories related to "Java Digital Clock" or "Java Swing Projects" to find sample projects and code snippets.

2. Stack Overflow

- Use Stack Overflow to ask specific questions or find answers related to Java programming issues you may encounter.

FORUMS AND COMMUNITY

1. Java Forums

- Participate in forums like JavaRanch or Oracle Community to ask questions and share knowledge with other Java developers.

2. Reddit

- Subreddits like r/java can be useful for discussions and advice on Java projects.

APPENDIX-A SOURCE CODE

```
import java.awt.*;
import java.awt.event.*;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import javax.swing.*;
import java.io.File;
import javax.imageio.ImageIO;
import java.io.IOException;

public class Main extends JFrame {

    private Calendar currentTime;
    private JPanel clockPanel;
    private Color frameColor = Color.BLACK;
    private Calendar alarmTime = null;
    private boolean alarmRinging = false;
    private Image backgroundImage; // To store the background image

    public Main() {
        setTitle("Enhanced Digital Clock");
        setSize(600, 600);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setResizable(false);
        currentTime = Calendar.getInstance();
        clockPanel = new ClockPanel();
        clockPanel.setBackground(frameColor);
    }
}
```

```

JPanel featurePanel = new JPanel();
featurePanel.setLayout(new GridLayout(1, 3));

JButton setTimeButton = new JButton("Set Time");
setTimeButton.addActionListener(e -> setTime());

JButton setAlarmButton = new JButton("Set Alarm");
setAlarmButton.addActionListener(e -> setAlarm());

JButton changeColorButton = new JButton("Change Frame Color");
changeColorButton.addActionListener(e -> changeFrameColor());

featurePanel.add(setTimeButton);
featurePanel.add(setAlarmButton);
featurePanel.add(changeColorButton);
setLayout(new BorderLayout());
add(clockPanel, BorderLayout.CENTER);
add(featurePanel, BorderLayout.SOUTH);
loadBackgroundImage();
Timer timer = new Timer(1000, e ->
    {updateClock();
     checkAlarm();
    });
timer.start();
}

private void loadBackgroundImage()
{try {
    backgroundImage = ImageIO.read(new File("background.jpg"));
} catch (IOException e) {
    System.out.println("Error loading background image.");
}
}

```

```

    }
}

private class ClockPanel extends JPanel
{
    @Override
    protected void paintComponent(Graphics g)
    {
        super.paintComponent(g);
        if (backgroundImage != null) {
            g.drawImage(backgroundImage, 0, 0, getWidth(), getHeight(), this);
        }
        int hours = currentTime.get(Calendar.HOUR);
        int minutes = currentTime.get(Calendar.MINUTE);
        int seconds = currentTime.get(Calendar.SECOND);
        String amPm = currentTime.get(Calendar.AM_PM) == Calendar.AM ? "AM" :
"PM";
        int centerX = getWidth() / 2;
        int centerY = getHeight() / 2;
        int radius = Math.min(center X, center Y) - 40;
        g.setColor(Color.WHITE);
        g.drawOval(centerX - radius, centerY - radius, 2 * radius, 2 * radius);

        // Draw numbers
        g.setFont(new Font("Arial", Font.BOLD, 16));
        for (int i = 1; i <= 12; i++) {
            double angle = Math.toRadians(i * 30 - 90);
            int x = (int) (centerX + radius * 0.85 * Math.cos(angle));
            int y = (int) (centerY + radius * 0.85 * Math.sin(angle));
            String number = Integer.toString(i);
            int stringWidth = g.getFontMetrics().stringWidth(number);
            int stringHeight = g.getFontMetrics().getHeight();
            g.drawString(number, x - stringWidth / 2, y + stringHeight / 4);
        }
    }
}

```

```

        drawHand(g, centerX, centerY, radius * 0.6, hours * 30 + minutes / 2,
Color.WHITE);
        drawHand(g, centerX, centerY, radius * 0.8, minutes * 6, Color.WHITE) ;
        drawHand(g, centerX, centerY, radius * 0.9, seconds * 6, Color.RED);
        g.setFont(new Font("Arial", Font.BOLD, 24));
        String digitalTime = new SimpleDateFormat("hh:mm:ss
a").format(currentTime.getTime());
        int stringWidth = g.getFontMetrics().stringWidth(digitalTime);
        g.setColor(Color.YELLOW);
        g.drawString(digitalTime, centerX - stringWidth / 2, centerY);
    }

    private void drawHand(Graphics g, int x, int y, double length, double angleDegrees,
Color color) {
        g.setColor(color);
        double angle = Math.toRadians(angleDegrees - 90);
        int xEnd = x + (int) (length * Math.cos(angle));
        int yEnd = y + (int) (length * Math.sin(angle));
        g.drawLine(x, y, xEnd, yEnd);
    }
}

private void updateClock()
{
    currentTime.add(Calendar.SECOND,
1);clockPanel.repaint();
}

private void setTime() {
    String time = JOptionPane.showInputDialog(this, "Enter current time (HH:mm:ss):");
    try {
        SimpleDateFormat sdf = new SimpleDateFormat("HH:mm:ss");
        java.util.Date date = sdf.parse(time); currentTime.setTime(date);
    }
}

```

```

        JOptionPane.showMessageDialog(this, "Time set to: " + time);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, "Invalid time format!");
    }
}

```

```

private void setAlarm() {
    String time = JOptionPane.showInputDialog(this, "Enter alarm time (HH:mm):");
    try {
        SimpleDateFormat sdf = new SimpleDateFormat("HH:mm");
        Calendar calendar = Calendar.getInstance();
        calendar.setTime(sdf.parse(time));
        alarmTime = calendar;
        JOptionPane.showMessageDialog(this, "Alarm set for: " + time);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, "Invalid time format!");
    }
}

```

```

private void changeFrameColor() {
    Color newColor = JColorChooser.showDialog(this, "Choose Frame Color",
frameColor);
    if (newColor != null)
    { frameColor =
      newColor;
      clockPanel.setBackground(frameColor);
    }
}

```

```

private void checkAlarm()
{if (alarmTime != null) {
    Calendar current = Calendar.getInstance();

```



```

        if(current.get(Calendar.HOUR_OF_DAY) ==
alarmTime.get(Calendar.HOUR_OF_DAY)
            &&current.get(Calendar.MINUTE) == alarmTime.get(Calendar.MINUTE)
            && !alarmRinging) {
                alarmRinging = true;
                showAlarmMessage();
            }
        }
    }

    private void showAlarmMessage() {
        int option = JOptionPane.showOptionDialog(this, "ALARM IS RINGING! Time to
wake up!", "Alarm",
            JOptionPane.DEFAULT_OPTION, JOptionPane.INFORMATION_MESSAGE,
            null,
            new String[]{"Stop", "Snooze (5 mins)"}, "Stop");

        if (option == 0) {
            alarmRinging = false;
        } else if (option == 1)
        { alarmTime.add(Calendar.MINUTE,
            5);alarmRinging = false;
        }
    }

    public static void main(String[] args)
    {SwingUtilities.invokeLater(() -> {
        Main clock = new Main();
        clock.setVisible(true);
    });
    }}

```

APPENDIX-B SCREEN SHOT

