Report On

Calendar Generator

Submitted in partial fulfillment of the requirements of the Course project in
Semester III of Second Year Artificial Intelligence and Data Science

by
Pratik Avhad (Roll No. 01)
Priyanka Bhandari (Roll No. 02)
Swarup Kakade (Roll No. 19)

Supervisor
Mrs. Sneha Yadav

University of Mumbai

Vidyavardhini's College of Engineering & Technology

Department of Artificial Intelligence and Data Science

(2023-24)

Vidyavardhini's College of Engineering & Technology
Department of Artificial Intelligence and Data Science

CERTIFICATE

This is to certify that the project entitled "Calendar Generator" is a bonafide work of "Pratik Avhad (Roll No. 01), Priyanka Bhandari (Roll No. 02), Swarup Kakade (Roll No. 19), submitted to the University of Mumbai in partial fulfillment of the requirement for the Course project in semester III of Second Year Artificial Intelligence and Data Science engineering.

Supervisor

Mrs. Sneha Yadav

Dr. Tatwadarshi P. N.
Head of Department

Index

Chapter No	**Calendar Generator**	Page No.
1	Title Page	**1**
2	Certificate	**2**
3	Abstract	**3**
4	Index	**4**
5	Block diagram , its description and working	**5-6**
6	Module Description	**7**
7	Brief description of software & hardware used and its programming	**8**
8	Code and Results	**9-10**
9	Conclusion and References	**11**

Abstract

The Calendar Generator Using AWT Project is a Java-based software application that leverages the Abstract Window Toolkit (AWT) to create a user-friendly platform for generating and customizing calendars. In an era where digital tools dominate daily organization, this project provides an accessible solution for users to craft personalized calendars tailored to their specific needs**.**

The Calendar Generator Using AWT Project is designed to serve a diverse user base, including individuals who wish to create personalized calendars, businesses in need of customized promotional materials, and event planners seeking to streamline their schedules. By combining AWT's capabilities with user-focused design and advanced customization features, this project aims to simplify calendar generation while enhancing its visual appeal.

In summary, the Calendar Generator Using AWT Project represents a significant step forward in digital calendar management. It equips users with the tools needed to efficiently design, personalize, and organize calendars, all while meeting their specific requirements and aesthetic preferences. This project harnesses AWT's capabilities to provide a user-friendly and attractive solution for modern calendar needs, ensuring that the process of calendar generation is as intuitive as it is functional.

The Calendar Generator Java Project is a versatile and user-friendly software application designed to simplify the creation and customization of calendars for a wide range of personal and organizational needs. This Java-based project empowers users to effortlessly design, personalize, and manage calendars according to their unique requirements. It offers features like calendar customization, various calendar views, event synchronization, export and printing capabilities, a reminder and notification system, and a user-friendly graphical interface. This project serves a diverse user base, including individuals, businesses, and event planners, streamlining calendar creation and enhancing its visual appeal.

The Calendar Generator Project is an innovative and user-friendly application designed to simplify the process of creating personalized calendars for individuals and organizations. In a digital age where calendars are an integral part of daily life, this project offers a solution that combines convenience, customization, and aesthetic appeal.

Working:-

The working of a Calendar Generator project involves several key components and steps. Here's an overview of how a typical Calendar Generator project operates:

1. **User Interface (UI)**: The project starts with a user-friendly graphical interface. Users interact with this interface. The UI is often created using technologies like Java AWT, Swing, or web-based frameworks.

2. **Calendar Views:** Calendar generators offer multiple views for users to choose from, such as monthly, weekly, or daily views. This flexibility allows users to select the format that best suits their scheduling requirements.

3. **Calendar Generation:** The heart of the project is the calendar generation component. This part of the system processes the user's selections and customizations to produce the final calendar. It arranges the chosen elements, themes, and events to create the visual and functional calendar.

4. **Backend Logic:** Behind the scenes, the project may involve complex algorithms and data processing to manage the customizations. This is especially true for more advanced calendar generators.

Description:-

In today's fast-paced world, calendars are essential tools for personal organization, scheduling, and planning. This project addresses the need for calendars that are not only functional but also aesthetically pleasing and tailored to individual or organizational preferences.

At its core, the Calendar Generator is built around a Java-based platform, offering a versatile solution for generating calendars for a wide range of purposes. One of its primary features is its user-friendly graphical interface, which makes it accessible to users of varying technical expertise. The interface is often developed using Java's Abstract Window Toolkit (AWT) or Swing, providing an intuitive experience that allows users to interact with the application effortlessly.

Module Description

The Abstract Window Toolkit (AWT) is a core module of Java's standard library that provides a set of classes for creating and managing graphical user interfaces (GUIs) in Java applications. AWT is one of the foundational components of Java's rich GUI capabilities and has been integral to Java application development since the language's inception. Here is a comprehensive description of the AWT module:

1. GUI Component Library:
At the heart of the AWT module is a comprehensive library of GUI components, scrollbars, and more. These components serve as building blocks for creating graphical user interfaces in Java applications. AWT components are platform-independent and can be used to develop GUI applications that run on different operating systems.

2. Platform Independence:
AWT was designed with platform independence in mind. AWT components are drawn by the host operating system's windowing system, which ensures that Java GUIs look and feel native on each supported platform. This means that Java applications built with AWT can run on different operating systems without modification, providing a consistent user experience.

5. Graphics and Drawing:
AWT offers a Graphics class that allows developers to draw shapes, text, and images directly onto GUI components. This is useful for creating custom graphics, charts, and visual elements within Java applications.

8. Integration with Swing:
While AWT is a foundational GUI library, it can be combined with Swing, a more modern and powerful GUI toolkit, for more sophisticated and feature-rich GUI development. Swing components are built on top of AWT components, and both libraries can be used together in the same application.

9. Limitations:
Despite its numerous advantages, AWT does have some limitations. It may not provide the same level of flexibility and customization as more modern GUI frameworks like JavaFX, which is designed for rich, multimedia-rich applications. However, AWT remains a useful choice for simpler applications and for maintaining compatibility with older Java code.

In summary, the AWT module in Java is a foundational component of the Java Standard Library that provides a set of GUI components and features for creating cross-platform graphical user interfaces. Its platform independence, event handling, and accessibility support make it a valuable tool for developing GUI applications in Java. While more modern GUI libraries like Swing and JavaFX offer additional features and flexibility, AWT continues to play a crucial role in Java's GUI development landscape.

Brief description of software & hardware used and its programming:

In a Calendar Generator Project in Java, both software and hardware components play integral roles in the successful development and operation of the project.

Software:

1. **Java Programming Language:** The project is primarily built using Java, which provides the foundation for the application's code and functionality.
2. **Integrated Development Environment (IDE):** A Java IDE such as Eclipse, IntelliJ IDEA, or NetBeans is used for coding, debugging, and managing the project.
3. **Java AWT/Swing:** These libraries provide the essential GUI components and features for creating the user interface and graphical elements of the calendar generator.
4. **Operating System:** The project can run on various operating systems, including Windows, macOS, and Linux, thanks to Java's platform independence.
5. **Version Control System:** Tools like Git may be used for source code version control and collaboration among development teams.

Hardware:

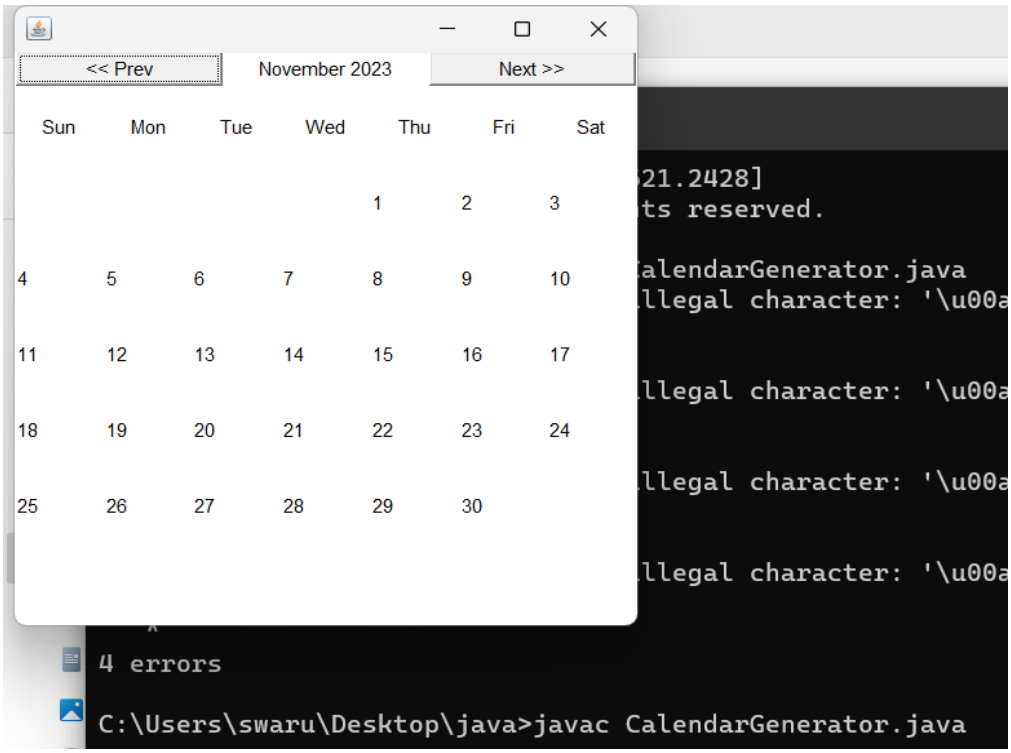
1. **Computer:** A standard desktop or laptop computer serves as the development and execution environment for the project.
2. **Processor:** A modern multi-core processor ensures smooth execution and responsiveness of the application.
3. **Memory (RAM):** Sufficient RAM is necessary for the application to handle graphics and data processing efficiently.
4. **Storage:** Adequate storage, including both hard disk drive (HDD) or solid-state drive (SSD), is essential for storing project files and data.
5. **Monitor/Display:** A computer monitor or display is used to visualize the project's graphical user interface and any generated calendars.
6. **Input Devices:** A keyboard and mouse (or touchpad) are used for user input during development and when interacting with the application.

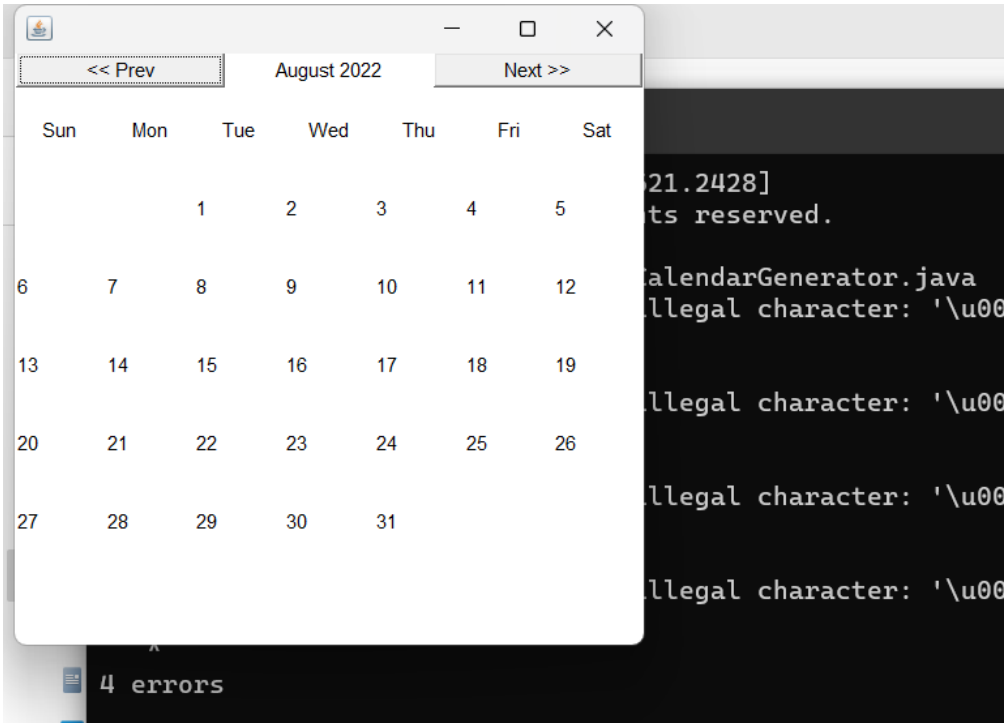
Code

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

public class CalendarGenerator extends Frame {
   private int month, year;
   private Label label;
   private Button prev, next;
   private Panel p1;
   private Panel p2;
   private String[] months = {"January", "February", "March", "April", "May", "June", "July", "August",
"September", "October", "November", "December"};
   private String[] days = {"Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"};
   private int startDay, numberOfDays;

   public CalendarGenerator() {
      month = java.util.Calendar.getInstance().get(java.util.Calendar.MONTH);
      year = java.util.Calendar.getInstance().get(java.util.Calendar.YEAR);

      label = new Label("", Label.CENTER);
      p1 = new Panel();
      p2 = new Panel();
      prev = new Button("<< Prev");
      next = new Button("Next >>");
      prev.addActionListener(new ActionListener() {
         public void actionPerformed(ActionEvent ae) {
            if (month == 0) {
               month = 11;
               year--;
            } else {
               month--;
            }
            displayCalendar();
         }
      });

      next.addActionListener(new ActionListener() {
         public void actionPerformed(ActionEvent ae) {
            if (month == 11) {
               month = 0;
               year++;
            } else {
               month++;
            }
            displayCalendar();
         }
      });
```

```
        p1.setLayout(new GridLayout(1, 3));
        p1.add(prev);
        p1.add(label);
        p1.add(next);
        p2.setLayout(new GridLayout(7, 7));

        for (int x = 0; x < days.length; x++) {
            p2.add(new Label(days[x], Label.CENTER));
        }

        for (int x = 0; x < 42; x++) {
            p2.add(new Label(""));
        }

        add(p1, BorderLayout.NORTH);
        add(p2, BorderLayout.CENTER);
        displayCalendar();
        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent we) {
                System.exit(0);
            }
        });
        setSize(400, 400);
        setVisible(true);
    }

    public void displayCalendar() {
        for (int x = 7; x < p2.getComponentCount(); x++) {
            p2.getComponent(x).setVisible(false);
        }

        java.util.Calendar cal = java.util.Calendar.getInstance();
        cal.set(year, month, 1);
        startDay = cal.get(java.util.Calendar.DAY_OF_WEEK);
        numberOfDays = cal.getActualMaximum(java.util.Calendar.DAY_OF_MONTH);
        label.setText(months[month] + " " + year);

        for (int x = 7 + startDay, day = 1; day <= numberOfDays; x++, day++) {
            Label dayLabel = (Label) p2.getComponent(x);
            dayLabel.setText("" + day);
            dayLabel.setVisible(true);
        }
    }

    public static void main(String[] args) {
        new CalendarGenerator();
    }
}
```

Result


<< Prev
November 2023
Next >>
Sun Mon Tue Wed Thu Fri Sat
1 2 3
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30
21.2428]
ts reserved.
alendarGenerator.java
llegal character: '\u00a
llegal character: '\u00a
llegal character: '\u00a
llegal character: '\u00a
4 errors
C:\Users\swaru\Desktop\java>javac CalendarGenerator.java


<< Prev
August 2022
Next >>
Sun Mon Tue Wed Thu Fri Sat
1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
21.2428]
ts reserved.
alendarGenerator.java
llegal character: '\u00
llegal character: '\u00
llegal character: '\u00
llegal character: '\u00
4 errors

Conclusion

In conclusion, the Calendar Generator Project in Java represents a versatile and user-friendly solution. It empowers users to design calendars that align with their personal or organizational preferences while providing a range of features, such as multiple calendar views. This project serves a diverse user base, including individuals, businesses, and event planners, streamlining the calendar creation process and enhancing its visual appeal. By combining Java's programming capabilities with the Abstract Window Toolkit (AWT) or Swing, the Calendar Generator Project ensures that users can effortlessly create, customize, and manage calendars that meet their unique needs, resulting in efficient and personalized calendar management.

References

Books:

1. "Java: The Complete Reference" by Herbert Schildt - This book covers various aspects of Java programming, including AWT and GUI development.
2. "Java AWT Reference" by John Zukowski - This reference book specifically focuses on the AWT library in Java, making it a valuable resource for your project.
3. "Java How to Program" by Paul Deitel and Harvey Deitel - This book provides a comprehensive guide to Java programming, including GUI development using AWT and Swing.

Websites:

1. Oracle's Java Documentation - The official documentation for Java provides detailed information on the AWT framework and how to use it in your projects.
2. Java Tutorials - Oracle's Java Tutorials offer step-by-step guides on various Java topics, including AWT.

3. Stack Overflow - This is a great place to find answers to specific coding problems related to your Java AWT project. Many developers share their knowledge and solutions here.
4. JavaWorld - JavaWorld is a resource for in-depth Java tutorials and articles, some of which may be relevant to your calendar project.
5. GitHub - You can find open-source Java calendar projects on GitHub, which can serve as a source of inspiration and reference for your own project.