Calender App

Jeremy Gordon
Kyle Byassee

Abstract

Our group project is a calender application. Some of the functions are notifying the user of events and customizing the user experience with pictures. The target audience are users that want more freedom from their calendar application.

1. Introduction

Our group project is a calender application, it is a C# WPF application. Some of the functions are notifying the user of events and customizing the user experience with pictures. We are targeting individuals who want the ability to customize, export, import and set specific day/month/year/time notifications. We want to give the user a calendar application that better than what is offered currently from calendly or canvas.

1.1. Background

We decided on this project because we do not have much experience in C#. We also felt we could improve on the traditional calender application providing additional freedom and features. Providing features like viewing historic information and customizing backgrounds with user pictures. A calender application sounded like a project that could be a fun, challenging and worthwhile semester long project.

1.2. Impacts

The project is geared towards a general audience. Our application is appealing to that audience because of the ability to customize the look of the applications background and set custom notifications. It may have a small, but positive effect on the end user by helping to manage and organize life events.

1.3. Challenges

The layout of the GUI seems daunting since some months have more days than others. We plan to take inspiration from the numerous calender applications that are already in the wild. While also improving on the GUI layout making it more user friendly and intuitive.

One challenge we encountered early on was changing the background of the calendar. We eventually overcame that challenge by searching online forums and other sources of information. Another challenge was setting the dates for notifications. The GUI is a still a challenge to make it appealing and modern looking.

2. Scope

The app will display the current date and time when first opened. Let the user change the date and set a notification on that date at a specific time. The user is be able to set the background of the app to a designated picture format.

The stretch goals are to export a user profile to be imported later into the same or different calender application and adding weather to be displayed in the GUI of the application.

2.1. Requirements

The functional and non-functional requirements were gathered based upon what the program would need to be considered complete. We decided early on that the functional requirements should put our calendar on functional parity with other existing calendars. So we minimally came up with the following list.

2.1.1. Functional.

- User needs to be able to specify a specific day/month/year to add a notification.
- User can change background of the calendar with a picture.
- User can change and navigate through the month, year and day.
- User can cancel or modify an existing notification.

Use Case ID	Use Case Name	Primary Actor	Complexity	Priority
1	Set a notification	User	Med	1
2	remove or change a set notification	User	Med	1
3	Change calender background	User	Low	1
4	View historic information	User	Low	1

TABLE 1. SAMPLE USE CASE TABLE

2.1.2. Non-Functional.

- The calendar can run in the background.

2.2. Use Cases

Use Case Number: 1
Use Case Name: Set a notification
Description: The user has a specific date and time they with for a notification to appear on. The user navigates to the desired date. Selects the date and then specifies a time where the program saves that data and creates a notification.

1) User navigates to desired date.
2) User selects the date.
3) User specifies a time on that date.

Termination Outcome: The user has a notification set on that time and day.
Alternative: Notification has already been created
Use Case Number: 2
Use Case Name: Cancel or Modify notification
Description: The user has a date and time that they either wish to reschedule or cancel. The user navigates to the date on the calendar. Then the user selects the notification.

1) User navigates to the same date
2) User selects the date
3) User selects the same time
4) User chooses to cancel or modify existing notification

Termination Outcome: The notification has been modified or canceled
Use Case Number: 3
Use Case Name: Change calender background
Description: The user desires to change the background of the calender

1) User clicks on a button to upload an image.
2) User selects the image.
3) User confirms the selected image.

Termination Outcome: The user has change the background of the calender.
Use Case Number: 4
Use Case Name: View historic information
Description: A user desires to view specific historic information on a date.

1) User navigates to a specific date.
2) User selects the specific date.

Termination Outcome: The historic information is displayed to the user.

2.3. Interface

Once the application is opened the user is greeted to a pleasant blue calendar displaying the current date. On the left is are the controls to change the background of the calendar and on the right are the controls to set an event.

3. Project Timeline

The above figure is the approximate timeline that our project followed. We followed the waterfall software development life cycle. Starting with requirements, we decided on what would define done for our project and what we would like to accomplish. Then we progressed to design, where we considered design patterns and the general structure of the program. Then came implementation, we encountered the most difficulties of the project in this phase. We eventually over came them and prevailed. Then we progressed to verification, this phase was shorter than planned as is common in waterfall. Now we are in the maintenance phase.

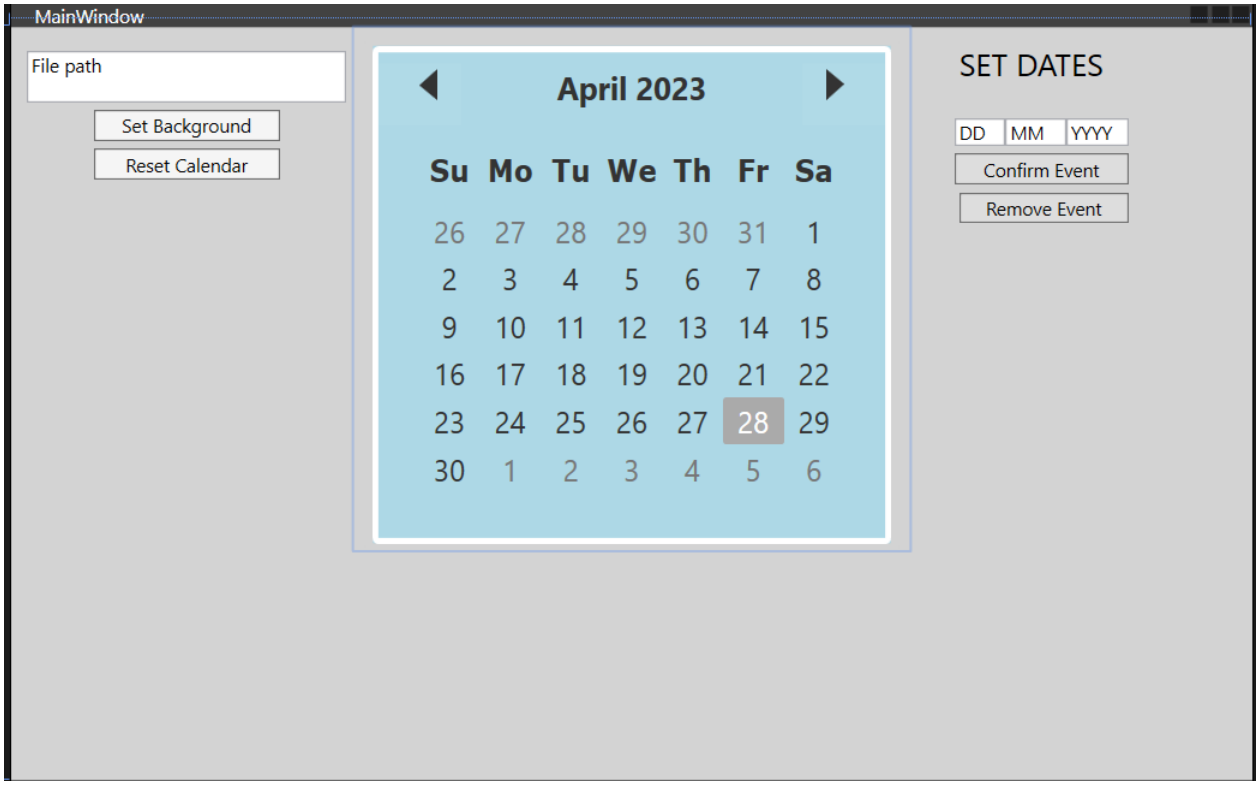

Figure 1. GUI of application

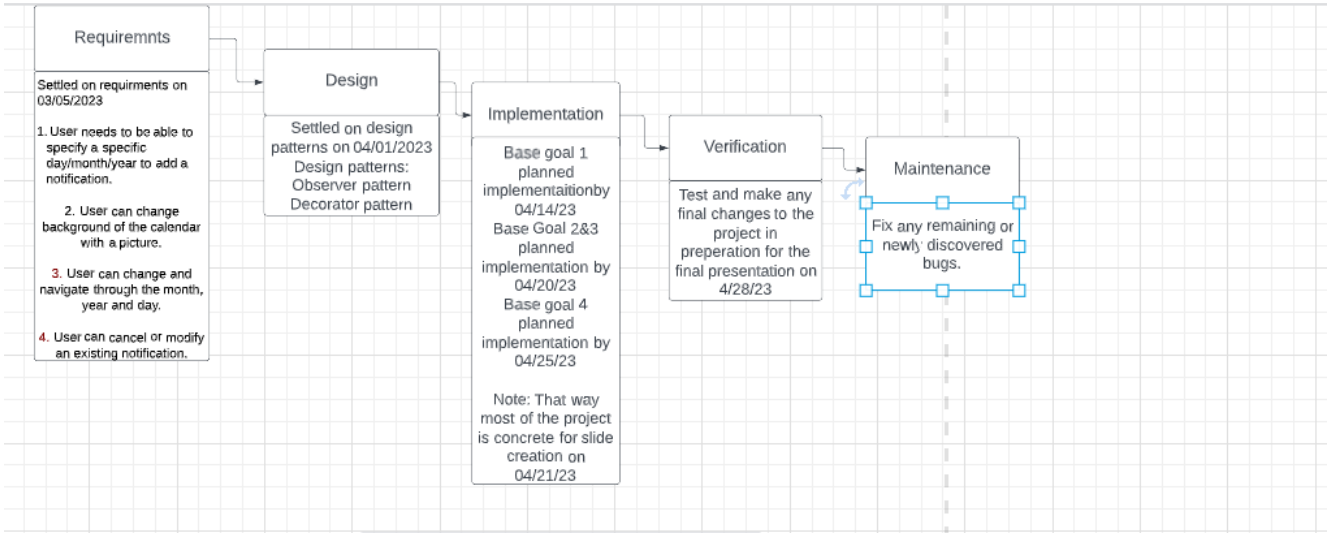

Figure 2. Project timeline

4. Project Structure

The way we designed the project so that where the user sees all their current events and notifications is clean. We tried implementing new windows for settings and events but could not get it working. We laid out all the functionality on the main window in a clean and strait forward way for the user.

4.1. UML Outline

We chose the decorator pattern as we already found a calender object we could just decorate to help meet one of our base goals. We chose the observer pattern so that while the calender is not actively running the notifications can be checked to see if the set time has passed. Our concrete observers are the calendar blackout dates objects and the notification objects. For example, see Figure 3.

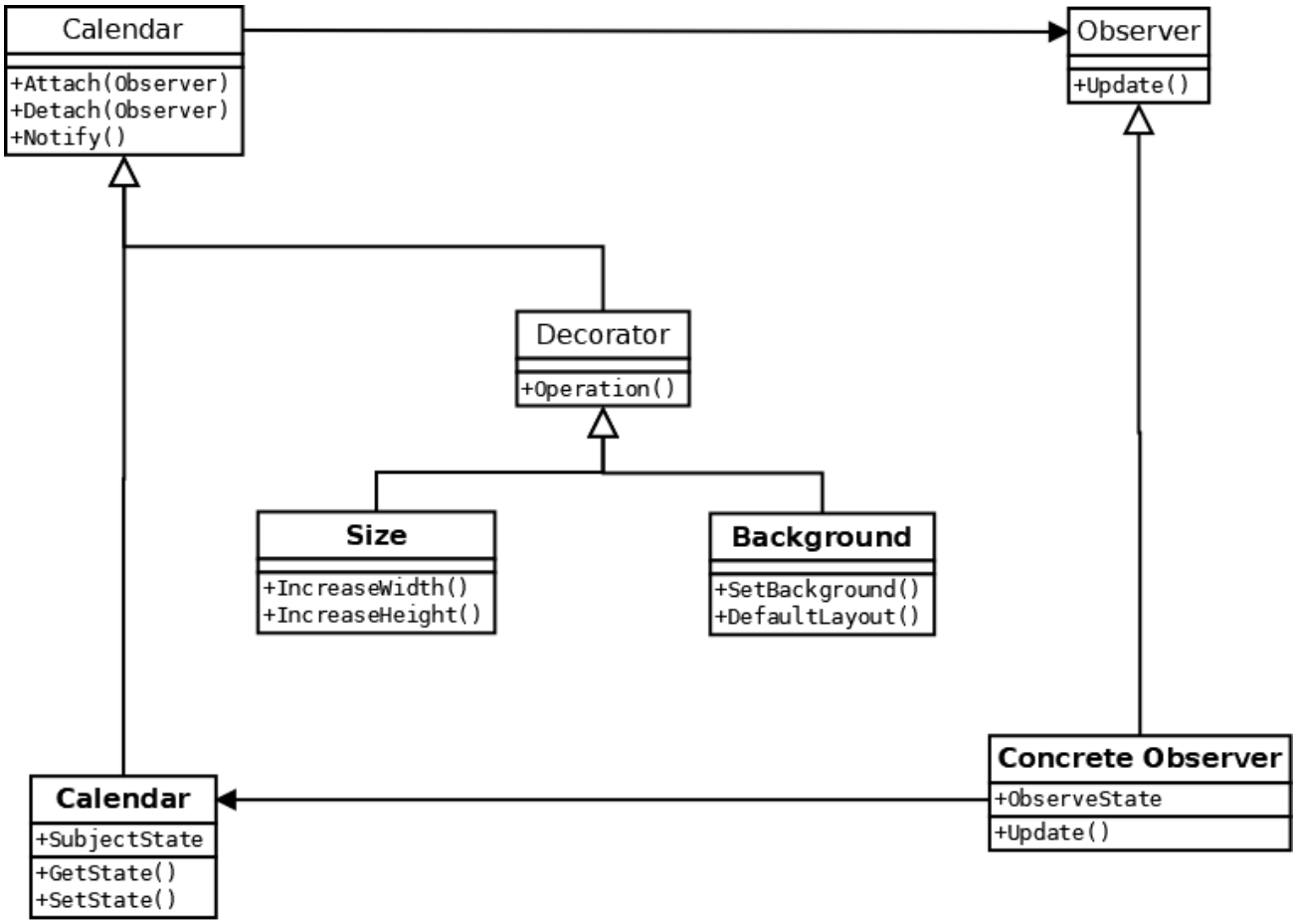

Figure 3. Uml outline of our Calender project

4.2. Design Patterns Used

We are implementing the observer and decorator patterns. The decorator pattern to add new functionality to the calendar object we are using. The observer pattern to notify the user when a set event is happening. In regards to 3 the decorator pattern can be seen in the center of the figure and the observer can be seen around the outer edge. The two patterns both utilize the calendar object but they do not interfere with each other.

5. Results

We implemented use cases 1, 2 and 3. The user can set the background of the calendar with an image using the images file path. The user can set a notification and remove a set notification. We did not finish implementing use case 4. Historic information and holidays are not in the current version of the application.

5.1. Future Work

We would like to build new features into our project in the future. We did not meet one of our base goals and would like to finish implementing that as we feel it is important to our project. We also want to expand our knowledge with C# and

push the boundary of what we thought was possible. One feature we wanted to implement was the exporting and importing from other calendar applications. However, if that does not come to fruition our project will be linked on our resume.

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.