Jess Teo - Project Portfolio

PROJECT: Gazeeebo

Overview

My team of 5 software engineering students and I were tasked with developing a basic command line interface application for our Software Engineering project. We chose to morph it into a Computer Engineering (CEG) school life planner called Gazeeebo. This enhanced application enables CEG students to plan their CEG life in university with more knowledge about the CEG course. Gazeeebo not only has the capabilities of a normal planner, but also the ability to give you an overview of the core modules you will need to take, thus enabling us to better plan our future schedule. This portfolio serves to document my contribution to the project.

This the current design of Gazeeebo:

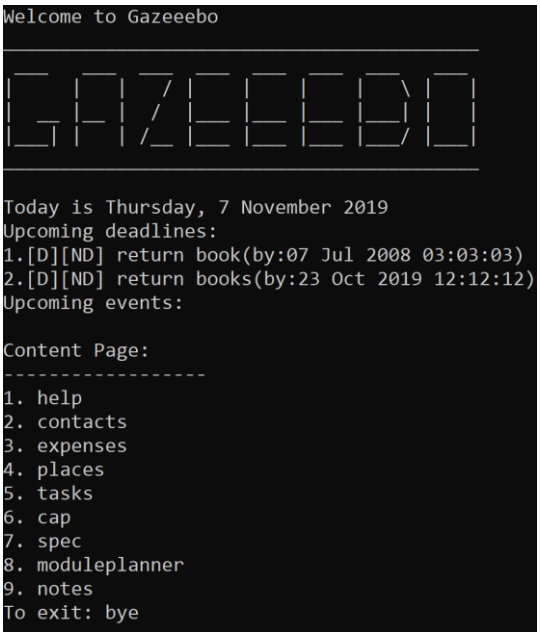

Figure 1.1 Gazeeebo Main Menu

My role was to come up with a function for students to search for places in National University of Singapore (NUS) School of Computing (SOC), design a calendar view and write the codes for the undo features. The following sections illustrate these enhancements in more detail, as well as the relevant sections I have added to the user and developer guides in relation to these enhancements.

Summary of contributions

> *This section shows the features that I have contributed to the project.*

- **Major Enhancement added: Places Page**
    - What it does: It is a function for users to search for places in SOC, they can add new places, delete existing ones and list out all places. This function allows users to locate places in

SOC and easily navigate their way to the location. It allows the user to undo all previous commands one at a time.

    - Justification: This feature improves the product significantly because a user might be unsure of the location in their timetable, this function enables them to easily search and locate the venue.

    - Highlights: This enhancement needs to be able to store and read information from a .txt file.

- **Major Enhancement added: Ability to undo commands**
    - What it does: It allows the user to undo all previous commands one at a time.

    - Justification: This feature improves the product significantly because a user can make mistakes in commands and the app should provide a convenient way to rectify them.

    - Highlights: This enhancement affects existing commands and commands to be added in future.

- **Code contributed**: Click on this link to see a sample of my code [RepoSense]
- **Other contributions**:
    - Project management:
        - Managed releases v1.3.2 on GitHub
    - Minor enhancements to existing features:
        - Calendar View. A monthly calendar can be print out which shows the current month and the current date, days that have tasks are demarcated by an "*". This function allows users to visualise and better plan their schedule. (Pull requests #146, #148, #149)

    - Documentation (User Guide and Developer Guide):
        - Reorganise the user guide and group it according to the feature so that it will be more user-friendly.

        - Standardised font and font size used for different sections.
        - Organised the contents of the Developer Guide (Pull Requests #272)
    - Community:
        - Reviewed Pull Requests (Pull Requests #166, #147, #100, #97, #78, #68, #66, #61, #51, #49

        - Reported bugs and suggestions for other teams in the class (examples: 1, 2, 3)

Contributions to the User Guide

We created a user guide for Gazeeebo. The following is an excerpt from our Gazeeebo User Guide, showing additions that I have made for the places feature and Calendar view feature.

Given below are examples of some sections that I contributed to the User Guide which showcase my ability to write documentation targeting end-users. For the full section, please visit the User Guide.

Places page: places

A function to locate places and rooms in NUS School of Computing.

In the places page, you can search for a place that you want to find or you can add and delete places and locations to the current list.

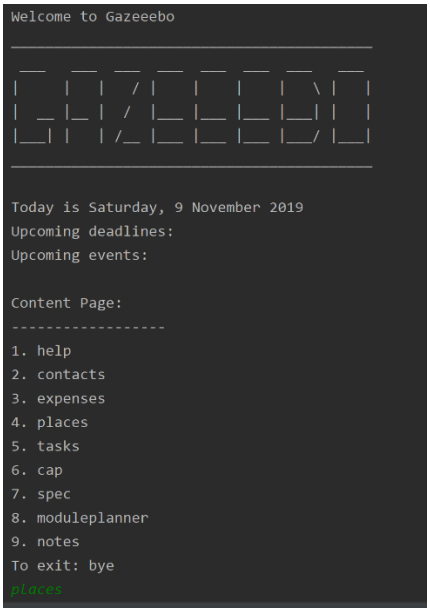

Figure 3.1 How to get to places page

How to get to places page:

- Type in the command `places` in the main menu page and press ENTER as shown in Figure 3.1 above. Alternatively, you can type in the index of places, `4` and press ENTER.
- You can only go to the places page from the main menu.

Adding a new place and location: add

Adds and stores a new place and location.

Format: `add-room,location`

Steps for adding a place:

1. Type in the command `add-room,location` in the above format. Eg. `LT19,COM2 Level 1`. Press ENTER. A success message will appear as seen in Figure 3.2.
2. Alternatively, you can input `add` or the index of the add command, `1`.
3. The system will then prompt you to enter the room and location in this format, `room,location`.

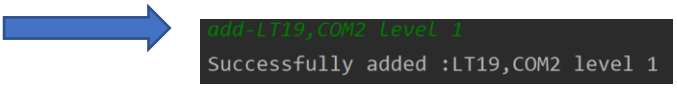

Figure 3.2 An example of adding a place

Finding a place in NUS School of Computing (SOC): find-place

Gives you the location of a specific place in SOC.

Format: `find-place`

Format details:

- `place` is the name of the place you want to find.

Steps for finding a place:

1. Type in the command in the format specified above. Eg. `find-LT19`. Press ENTER.
2. Alternatively, type `find` or the index for find command, `2`.
3. Press ENTER. The system will then prompt you to enter the place you want to find.

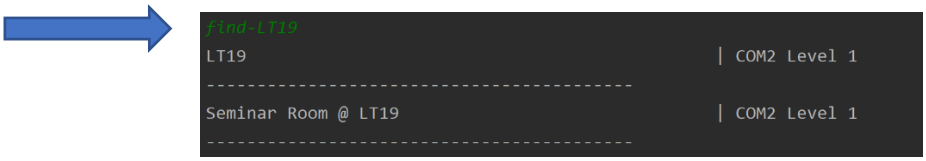

Figure 3.3 An example of finding a place

Listing all places in NUS School of Computing (SOC): list

Lists out all places in SOC.

Format: `list`

Steps for list command:

1. Type in `list`. Press ENTER.
2. Alternatively, type the index of list command, `4` and press ENTER.

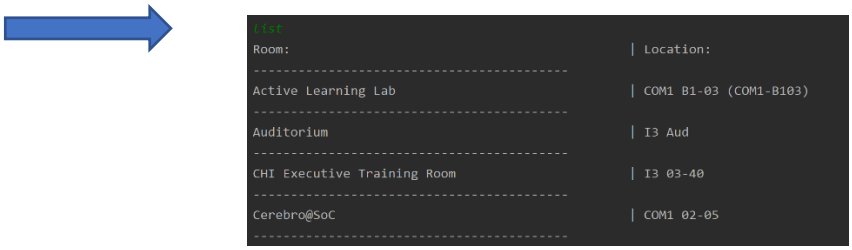

Figure 3.4 An example of list command

Deleting a place in the list: delete-place

Deletes an existing place list of places.

Format : `delete-place`

Format details:

- `place` is the name of the place you want to delete

Steps for deleting a place:

1. Type in the command in the format specified above. Eg. `delete-LT19`
2. Press ENTER.
3. Alternatively, you can input `delete` or the index of the delete command, `3`. Press ENTER
4. The system will then prompt you to enter the name of the room you wish to delete.

Figure 3.5 An example of deleting a place

Undo previous places command: undo

Undo the previous places command.

Commands that can be undone:

- add
- delete

Format: `undo`

Steps for undo command:

1. Type in `undo` and press ENTER.
2. Alternatively, type the index of undo command, `5` and press ENTER.

View current month in a calendar view: calendar monthly view

Shows the dates and current month in a calendar view.

Dates with tasks will be demarcated with a ‘ * ’.

Current date will be demarcated between ‘ | ’.

Format: `calendar monthly view`

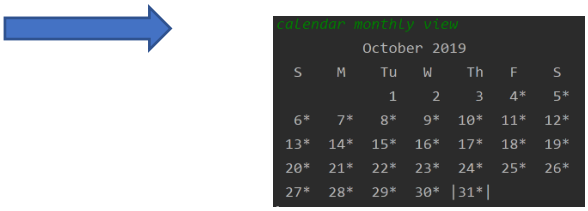

Figure 3.6 Calendar monthly view

View current year in a calendar view: calendar annual view

Shows the dates and months in a calendar view.

Dates with tasks will be demarcated with a ‘ * ’.

Current date will be demarcated between ‘ | ’.

Format: `calendar annual view`

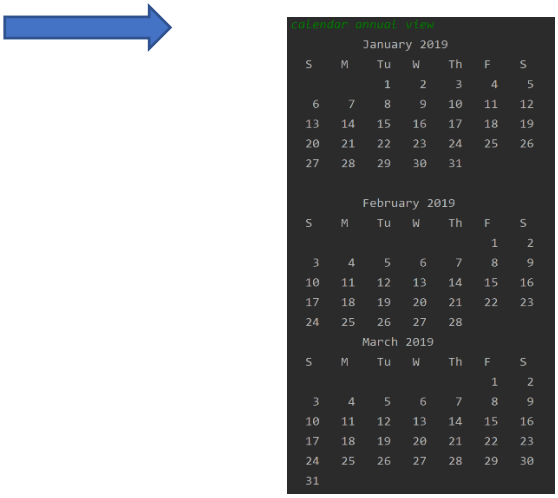

Figure 3.7 First 3 months of the calendar annual view

Contributions to the Developer Guide

The following section shows examples of my additions to the Gazeeebo Developer Guide. For the full section, please visit the Developer Guide.

Calendar View Feature

The Calendar view feature is facilitated by `CalendarView`. It contains the following methods:

- `CalendarView#isLeapYear(year)` - Check if the year passed in as a parameter is a leap year.
- `CalendarView#StartDay(month,day,year)` - Check the day of the 1st of each month.

- CalendarView#MonthlyView(list) - Print out the monthly calendar onto the command line
- CalendarView#AnnualView(list) - Print out the annual calendar onto the command line

Here is a class diagram to show the structure of the classes that implement the Calendar View feature:

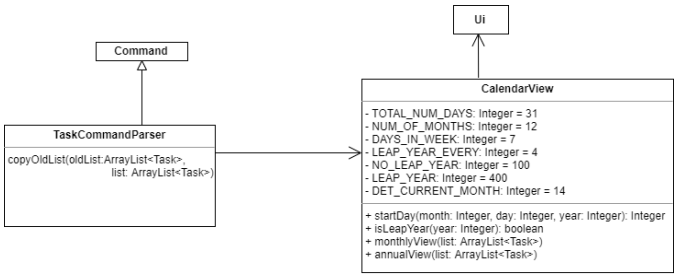

Figure 4.1 Class diagram for Calendar View feature

The following steps show how the Monthly View Calendar is built :

1. The constructor of CalendarView is called, thereby creating a new instance of CalendarView
2. MonthlyView method is called which calls get(calendar.MONTH)， get(calendar.YEAR) and get(calendar.DATE), from an instance of Calendar.
3. It then calls a method isLeapYear(year) to determine if the current year is a leap year or not.
4. Next, it calls a method StartDay(month,day,year) to determine the day of the first day of the month (e.g monday)
5. Finally, the calendar is printed onto the command line interface

The sequence diagram below outlines MonthlyView process:

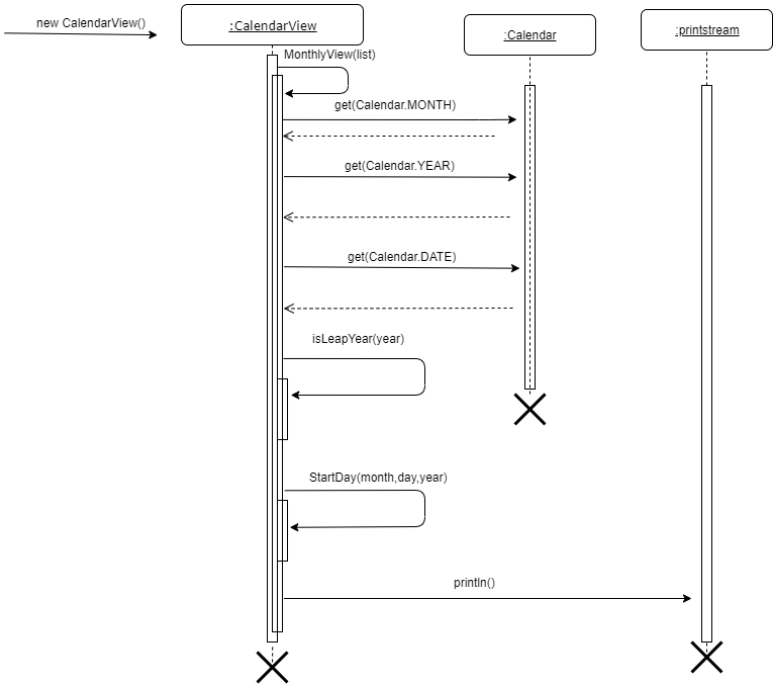

Figure 4.2 UML Sequence Diagram of CalendarView

Design Considerations

Aspect: Showing the current date

- Alternative 1(current choice): Print the current date between “|”.
    - Pros: Works for all types of command line interfaces.
    - Cons: Might disrupt the columns of the calendar causing the dates to be printed slightly out of position.
- Alternative 2: Change the colour of the date being printed.

- Pros: Aesthetically pleasing using colours to mark the current date.
- Cons: The use of ANSI escape codes to print coloured numbers on the command line does not work on Windows Command Prompt.

Places Feature

The Places feature is facilitated by PlacesCommand. It extends Command and stores the location information internally as an `Map<String, String> places` and also externally as `Places.txt`. Additionally, it implements the following operations:

- PlacesCommand#add() - Adds a location into the list of places.
- PlacesCommand#delete() - Delete a location from list of places.
- PlacesCommand#find() - Find a location from places based on the name.
- PlacesCommand#list() - List out the locations from places.
- PlacesCommand#undo() - Undo the previous command.

Here is a class diagram to show the structure of the classes that implement the Places page feature:

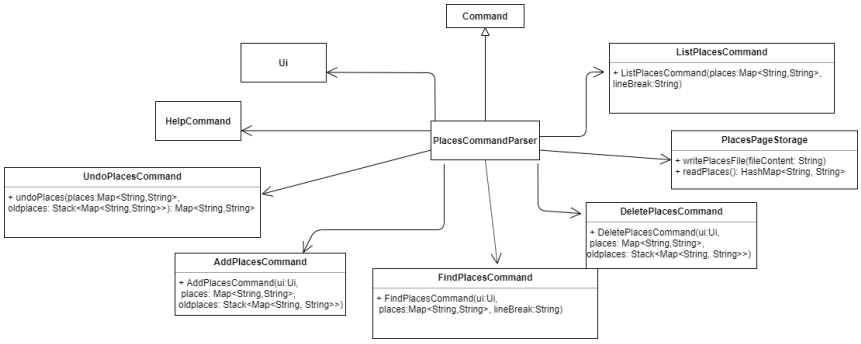

Figure 4.3 Class Diagram of PlacesCommand

The following steps show a usage scenario:

Step1. The user enters the application for the first time. PlacesCommand is initialised

Step 2. The user inputs a command, `add, delete, list or find.`

Step 3. The respective command will be initialised.

Step 4. The user exits by inputting `esc.`

The following sequence diagram illustrates the initialization steps:

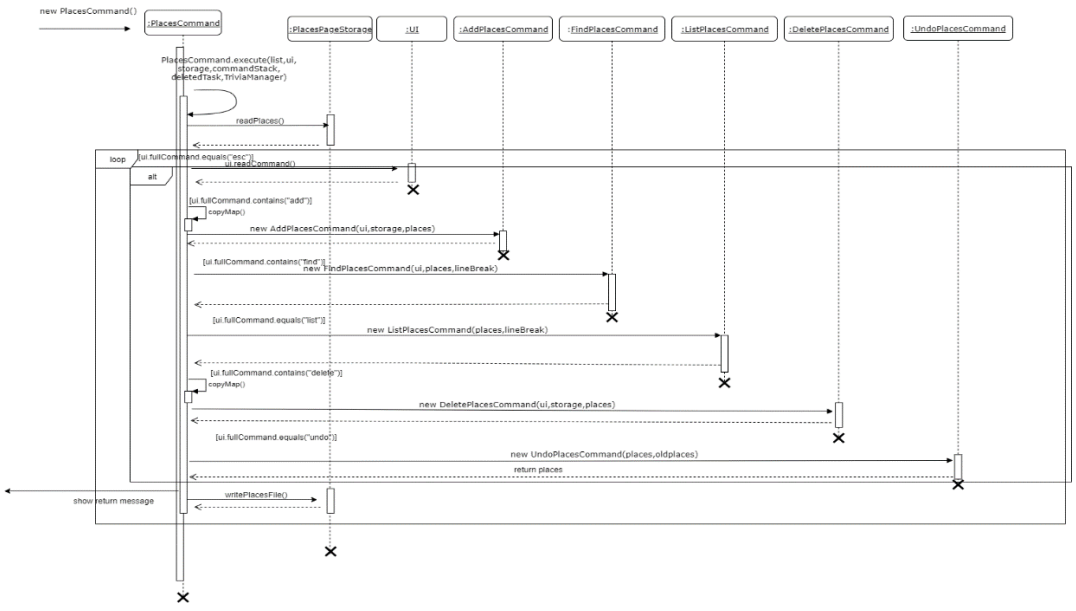

Figure 4.4 UML Sequence Diagram of PlacesCommand

When an AddPlacesCommand/ DeletePlacesCommand is called:

Step 1. User will have to input the place and location they want to add/delete.

Step 2. The location will be added/deleted from `Map<String, String> places`.

When a ListPlacesCommand is called:

Step 1. User inputs `list` into the command line.

Step 2. The `Map<String, String> places` will be printed out.

When a FindPlacesCommand is called:

Step 1. User inputs the name of the place they want to find.

Step 2. The places that have similar names will be printed out.

Step 3. If there is no such place in the `Map<String,String> places`, a message will be printed which indicates the place searched is not in `Map<String,String> places`.

The following activity diagram summarizes what happens when a user executes a new places command:

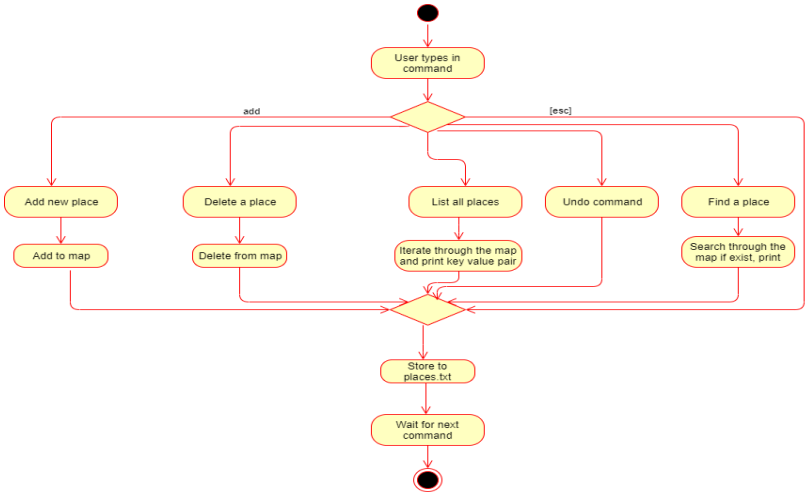

Figure 4.5 UML activity Diagram of PlacesCommand

Design Considerations

Aspect: Data structure to support Places

- Alternative 1(current choice): Using TreeMap to store locations and places.
  - Pros: TreeMap is has a natural ordering and the keys are sorted which reduces the need to sort the map when it is listed out.
  - Cons: TreeMap's time complexity of insertion and delete is O(log n) which is slower than HashMap.
- Alternative 2: Using a HashMap to store locations and places
  - Pros: HashMap has a time complexity of O(1) for insertion and deletion, which is faster than TreeMap.
  - Cons: Null values/keys are allowed in a HashMap. Requires an additional validation to ensure users do not input null values/keys.