ModPlan v1.1 - User Guide

By: `Team AY1920S1-CS2113T-F10-1` Since: `Aug 2019` License: `NUS School of Computing`

1. Introduction

ModPlan

Your one stop solution to module planning through out your university tenure!
ModPlan is a module planning system that expands on NUSMODS to help NUS Computer Engineering students to plan their modules for all four years of their bachelor's degree.

---

2. Quick Start

1. Ensure you have Java `11` or above installed in your Computer.
2. Download the latest `ModPlan.jar` here.
3. Copy the file to the folder you want to use as the home folder for your ModPlan application.
4. Type the command in the command box and press `Enter` to execute it.

5. Some example commands you can try:
    - `list` : lists all tasks
    - `add todo finish homework` : adds a Todo task with the description `finish homework` to the task list.
    - `delete 3` : deletes the 3rd task shown in the current list
    - `bye` : exits the app
6. Refer to Features for details of each command.

---

3. Features

Command Format

- Words in `UPPER_CASE` are the parameters to be supplied by the user e.g. in `add TASK_TYPE`, `TASK_TYPE` is a parameter which can be used as `add todo` or `add deadline`.
- Items in square brackets are optional e.g `KEYWORD [MORE_KEYWORDS]` can be used as `find book` or as `find book read`.

3.1. Adding a task: `add`

Adds a specific type of task to the task list. The parameters to be filled in by the user differs according to the task type.

3.1.1. Adding a Todo task: `todo`

- Adds a task which can be completed at any timing without restrictions.
- Format: `todo TASK_NAME`

3.1.2. Adding a Deadline task: `deadline`

Adds a task which has a specific date and time to be completed by.
Format: `deadline TASK_NAME /by DATE TIME`

- The date must be in the dd-mm-yyyy format that the program understands it.
- The time must be in the hh:mm format that the program understands it.

3.1.3. Adding a DoWithin task: `doWithin`

Adds a task which has to be done within a specific time period.
Format: `doWithin TASK_NAME /begin DATE TIME /end DATE TIME`

- The first date and time is the starting date.
- The second date and time is the ending date.
- The date must be in the dd-mm-yyyy format that the program understands it.

- The time must be in the hh:mm format that the program understands it.

Examples:

- `doWithin write essay /begin 12-08-2019 12:00 /end 13-08-2019 12:00`
  Adds a task, `write eassay` which has to be completed from 12/08/2019 12pm to 13/08/2019 12pm.

3.1.4. Adding an Event task: `event`

Adds a task which is occuring at a specific date and time.
Format: `event TASK_NAME /at DATE TIME`

- The date must be in the dd-mm-yyyy format that the program understands it.
- The time must be in the hh:mm format that the program understands it.

3.1.5. Adding a FixedDuration task: `fixedDuration`

Adds a task which requires a fixed amount of time to be completed in, but does not have a fixed starting or end time. Format: `fixedDuration TASK_NAME /needs TIME`

- The time must be in the hh:mm format that the program understands it.

3.1.6. Adding a Recurring task: `recurring`

Adds a task which occurs periodically.
Format: `recurring TASK_NAME /every DAYS_NUMBER`

Examples:

- `recurring update diary /every 3` Adds a task, `update diary`, which occurs every 3 days.

3.2. Deleting a task : `delete`

Deletes the specified task from the task list.
Format: `delete INDEX`

- Deletes the task at the specified `INDEX`.
- The index refers to the index number shown in the displayed task list list.
- The index **must be a positive integer** 1, 2, 3, …

Examples:

- `list`
  `delete 2`
  Deletes the 2nd task in the task list.

3.3. Listing all tasks : `list`

Shows a list of all tasks in the task list.
Format: `list`

3.4. Locating tasks in the list by name: `find`

Finds tasks from the task list whose description or date/time contain any of the given keywords.
Format: `find KEYWORD`

- The search is case insensitive. e.g `event` will match `Event`
- Part of the description will also be searched. e.g 'as' will match 'has', 'class' etc.
- Searching for the full date/time must be done in the **dd-mm-yyyy hh:mm** format to match the task list.

Examples:

- `find event`
  Returns `event` and `this event`

3.5. Marking a task as done: `done`

Marks a certain task, which is completed, as done. Format: `done INDEX`

- Changes the status of completion of the task from [✗] to [✓].
- The index of the task must be valid (i.e the task’s index must be in the task list).

3.6. Rescheduling tasks: `reschedule`

Reschedules a certain task to a different date and time.
Format: `reschedule INDEX DATE TIME`

- The index of the task must be valid (i.e the task’s index must be in the task list).
- A valid date and time must be inputted in the format that the program understands.
- The date and time inputted must be an open timing, if there are clashes the program will warn the user about the clashing timing.

3.7. Schedule of the day: `schedule`

Lists the schedule one has for a specific date.
Format: `schedule DATE`

- A valid date must be inputted in the format that the program understands.
- The list will be sorted from earliest to latest.

3.8. Reminders: `reminder`

Reminds the user upon startup of the program of any upcoming tasks.

- Notifies for any task due within the next 6 hours.
- Rechecks every hour for new upcoming tasks.

3.9. Exiting the program : `bye`

- Typing `bye` into the command line shows a goodbye message, saves the task list, and closes the program.

Exits the program.
Format: `bye`

3.10. Saving the data

Task list data are saved in the hard disk automatically after any command that changes the data. There is no need to save manually.

---

4. Errors

Error Handling * When the user inputs commands or parameters in a way that the program does not understand, errors will be thrown, which let the user know what is wrong.

TIP If you follow what the errors tell you to fix in your command, you can get the program to work as intended!

4.1. `DukeInvalidIndexException`

This error appears when the user inputs a number that is out of bounds of the task list.

DukeException

- Example: `delete -1`

4.2. `DukeInvalidTimeException`

This error appears when the user inputs a date or time that is not of an acceptable format.

DukeInvalidTimeException

- Example: `deadline assignment /by 30/9/2019 12 o’clock`
  Here 12 o’clock is not an acceptable format, instead use 1200.

4.3. `DukeInvalidTimePeriodException`

This error appears when the use inputs a time period that is not of an acceptable range.

- This is only valid for the DoWithin task.
- Example: `doWithin write essay /begin 12-08-2019 12:00 /end 1-08-2019 12:00` + Here the end date is earlier than the start date.

4.4. `DukeMissingArgumentException`

This error appears when the user does not input valid parameters into the command line.

- Example: `deadline /by 30-9-2019 12:45`
  Here the description is missing for the Deadline task, and the error message is shown.

4.5. `DukeCommandException`

This error appears when the user does not input a valid command name into the command line.

DukeCommandException

- Example: `activity finish writing book`
  Since there is no such command for activity, the program will be unable to process the unknown command.

4.6. `DukeEmptyCommandException`

This error appears when the user does not input anything after writing a valid command into the command line.

DukeCommandException

- Example: `delete`
  Since there is no index specified after `delete`, the program is unable to process which task to delete.

4.7. `DukeEmptyListException`

This error appears when the user does not input any tasks into the list or has deleted all the current tasks in the list.

DukeEmptyListException

- This is only valid for the `list` command.

4.8. `DukeMultipleValuesForSameArgumentException`

This error appears when the user inputs duplicated commands into the command line.

- Example: `doWithin test /begin 10am /begin 2pm /end 3pm`
  Here the `/begin` is repeated twice so the program is unable to determine which start time to be procesed.

4.9. `DukeScheduleException`

This error appears when the user inputs a task that already exists in the task list.

DukeScheduleException

4.10. `DukeNoTimeException`

This error appears when the user reschedule a non-time-based task.

- This is only valid for rescheduling a todo task.

5. FAQ

Q: How do I transfer my data to another Computer?
A: Install the app in the other computer and overwrite the empty data file it creates with the file that contains the data of your previous Data folder.

6. Command Summary

- **Add** `add TASK_TYPE TASK_DESCRIPTION [TASK_DATETIME]`
  e.g. `add deadline finish project milestone /by 10/10/2019 12:00`
- **Delete** : `delete INDEX`
  e.g. `delete 3`
- **Find** : `find KEYWORD [MORE_KEYWORDS]`
  e.g. `find homework`
- **List** : `list`