



# USER GUIDE

TASKELL

MARCUS NG WEN JIAN

JAZLYN ANG CHUE CHING

MADASAMY RAVI NADAR MAMTHA

ZUO ZHUO LIN

# Table of Contents

1.0	Acknowledgement .....	2
2.0	Introduction .....	3
3.0	Quick Start.....	4
4.0	Features .....	5
5.0	Frequently Asked Questions (FAQ).....	15
6.0	Glossary .....	15
7.0	Command Summary.....	20
8.0	Date Format Summary .....	21
9.0	Time Format Summary.....	21

# Acknowledgement

We would like to thank our professors, teaching staff and fellow classmates for helping us, Team Autumn (W15-C3), in this project.

Firstly, we would like to thank Professor Damith C. Rajapakse for teaching us Software Engineering concepts and providing us with a basic address book program to work with. This helped us better reinforce the concept learnt and put into practice.

Secondly, we would like to thank Mr. Joshua Lee, Mr. Leow Yijin, Mr. Martin Choo, Mr. Thien Nguyen, Mr. You Liang and Mr. Sam Yong for enhancing the code, and showing us greater insight to coding.

Thirdly, we would like to thank *Marco Jakob* for providing a clear explanation of javafx in his tutorial.

Lastly, we would like to make a special mention to our tutor, Mr Akshay Narayan, for providing useful feedback each week and showing us how the product can be improved.

# Introduction

Are you having a hard time remembering all the work you have to do? Do you have trouble finding a task manager that suits your preference for keyboard input? Well, worry no more, Taskell is here for you!

Taskell will be your personal secretary. It will keep track of your daily tasks and remind you of any important dates and deadlines. What distinguishes Taskell from other task managers is that Taskell only requires a single line of command for every task input. This means that you can record each one of your tasks with just a single statement. You will no longer have to use a mouse if you do not wish to.

Ready to begin life anew with a more efficient task manager? Read on to find out more!

# Quick Start

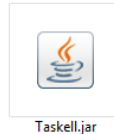
This section guides you through the installation of Taskell.

Step 1: Ensure you have Java version 1.8.0\_60 or later installed in your computer.

Having any Java 8 version is not enough.

This application will not work with earlier versions of Java 8

Step 2: Download the latest Taskell.jar from  
<https://github.com/CS2103AUG2016-W15-C3/main/releases>.



Step 3: Copy the file to the folder you want to use as the home folder for Taskell.

Step 4: Double-click the file to start the application. The GUI should appear in a few seconds.

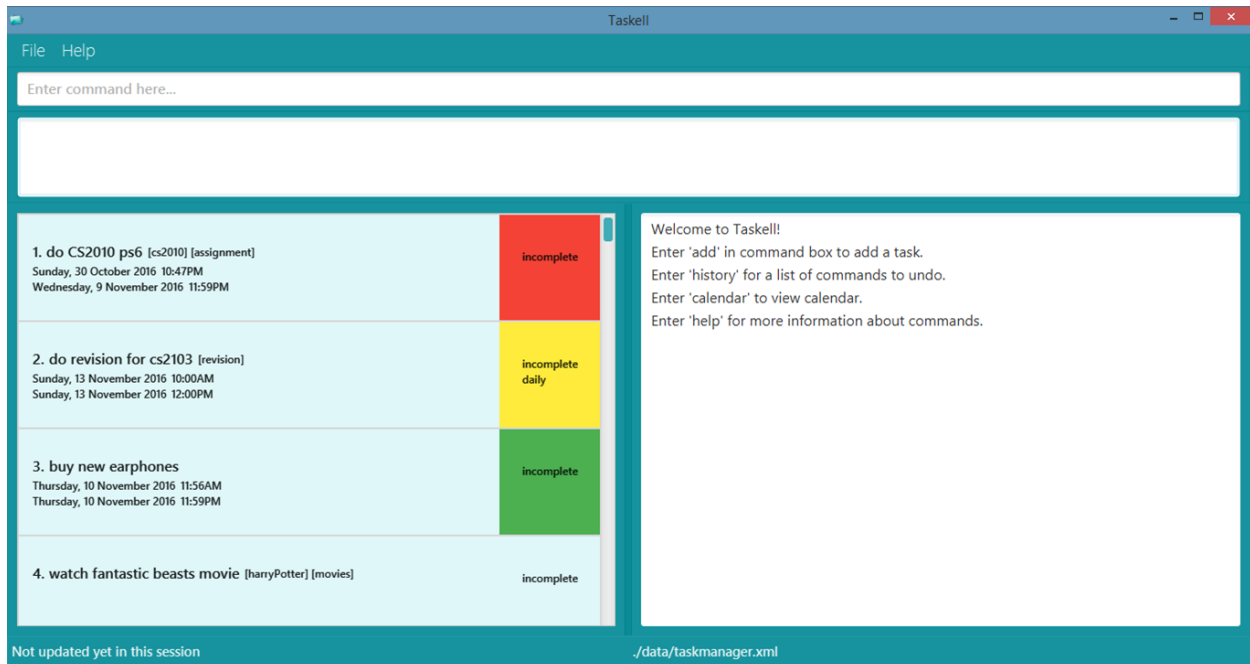


Figure 1: A screenshot of the Graphical User Interface (GUI)

Step 6: Type the relevant command in the command box and press  to execute it.

Step 7: Some example commands you can try:

- `list`: Displays all contacts
- `add` buy MA1101R textbook *by* today: Adds a task called buy MA1101R textbook to be done by today.
- `delete` 3: Deletes the third task shown in the current list
- `exit`: Exits Taskell

Refer to the Features section on the next page for details of each command.

# Features

This section shows the different commands that you can use in Taskell. Words that are in UPPER\_CASE are parameters. The parameters are listed below.

- TASK: Indicates the content of a work
- DATE: Indicates a date
  - Please refer to Date Format Summary for the date formats that Taskell suggests.
  - Default start date has been set to today's date.
  - Default end date has been set to be the same as the start date.
- TIME: Indicates a time
  - Please refer to Time Format Summary for the time formats that Taskell supports.
  - Default start time has been set to 12:00AM.
  - Default end time has been set to 11:59PM.
  - If you added a task today without a start time, the default start time will be set to the current time.
- PRIORITY: Indicates the level of importance of a task ranging from level 0 to 3. Level 0, 1, 2, 3 indicates default, low, medium and high priority respectively
  - In the GUI, level 1, 2 and 3 tasks are marked as green, yellow and red respectively.
  - Tasks with default priority level are not marked with any colors.
- RECURRING: Indicates the repetitive nature of a task. A task can be repeated daily, weekly or monthly
- TAG: Indicates the category in which a task belongs to

Words that are in *italics* are used to identify the parameters while words enclosed in SQUARE\_BRACKETS are optional. INDEX refers to the index number shown in the most recent listing.

## Viewing list of commands: `help`

You can use the `help` command to view a summary of all the commands.

To open the help window

Format: `help`

## Adding a task: `add`

You can use the `add` command to add different tasks.

To add a floating task

Format: `add TASK`

Example: `add Read Harry Potter Book`

To add a task with priority

Format: `add TASK p/PRIORITY`

Example: `add Complete math assignment p/3`

To add a task with tag(s)

Formats:

- `add TASK #TAG`  
Example: `add Meet Alice in Bugis #friends`
- `add TASK #TAG [#MORE_TAGS]`  
Example: `add Swimming with Jane #friends #leisure`

To add a recurring task

Format: `add TASK r/RECURRING`

Example: `add Read newspaper on mon r/daily`

Take Note!

Floating tasks are not allowed to have recurring status since they do not have any element of date or time.

To add a task with date and time

Formats:

- `add TASK from START_DATE to END_DATE`  
Example: `add Go camping at Kota Tinggi from 3-jun-2016 to 7-jun-2016`
- `add TASK from START_TIME to END_TIME`  
Example: `add Watch Dr Strange from 7.30pm to 9.25pm`

To allow greater flexibility in the command format, Taskell supports a few natural variation such as *by*, *on* and *at*.

The *by* prefix indicates that the tasks is a deadline. Any date or time preceded by this keyword will be stored as an end date and end time respectively.

Formats:

- `add TASK by[DATE]`  
Example: `add Buy textbook by today`
- `add TASK by[TIME]`  
Example: `add Visit Sandy at her house by the seaside by 3.35pm`
- `add TASK by[DATE] by[TIME]`  
Example: `add Do lab homework by Friday by 7pm`

The *on* prefix indicates that the task has to be done on the given date. Any date preceded by this keyword will be stored as a start date.

Format: `add TASK on DATE`

Example: `add Go for meeting on mon`

The *at* prefix indicates that the task has to be done at the given time. Any time preceded by this keyword will be stored as a start time.

Format: `add TASK at TIME`

Example: `add Go for meeting at 3pm`

Having understood the aforementioned behaviors of the *by*, *on*, *at*, *from* and *to* prefixes, you can fuse them together to form more complex tasks.

Format:

- `add TASK on DATE at TIME`  
Example: `add Go for meeting on Sunday at 3pm`
- `add TASK by TIME on DATE`  
Example: `add Go for meeting by 3pm on 1-jan`
- `add TASK from DATE`  
Example: `add Go out with friends from 9pm`
- `add TASK on DATE from TIME to TIME`  
Example: `add Watch webcast on sat from 4.45pm to 7pm`
- `add TASK from DATE to DATE from TIME to TIME [#TAG][p/PRIORITY][r/RECURRING]`  
Example: `add Holiday in San Francisco at Ocean Beach by the sea from 7-may to 2-jun from 9am to 11pm #holiday #leisure p/3 r/monthly`

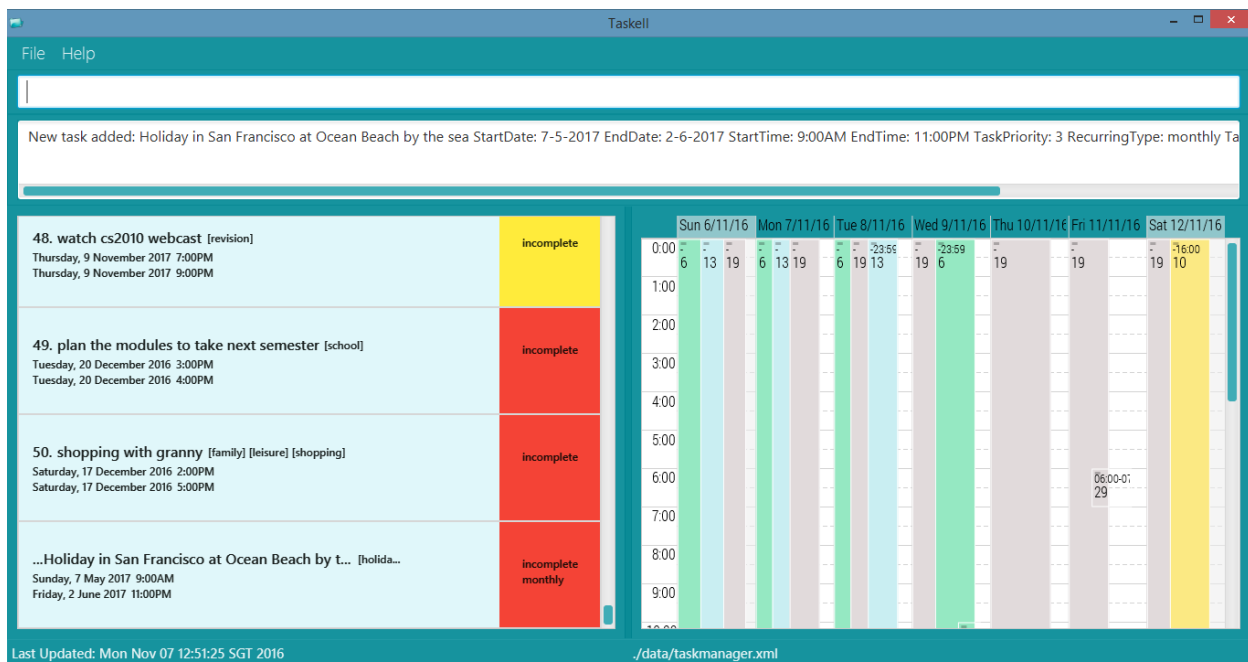


Figure 2: Adds a new task to Taskell



Take note!

- A task can only contain up to a maximum of 1 start time, 1 end time, 1 start date and 1 end date. Any additional date time parameters will be treated as part of the description.
- Any date or time not preceded by *by*, *on*, *at*, *from* and *to* will also be treated as part of the description.

Moreover, Taskell is able to adjust the date and time automatically so that the task entered remains relevant.

Examples:

- `add` Create powerpoint slides for project *from* 11pm *to* 3am  
This task will be added as a valid task that starts from today at 11pm and ends tomorrow at 3am
- `add` Staycation with friends *from* Sunday *to* tues  
If today is a Saturday, this task will be added as a valid task that starts from tomorrow and end on this coming Tuesday.  
If today is a Sunday, this task will be added as a valid task that starts from next Saturday and end on the following Tuesday.

## Editing a task: `edit`

You can use the `edit` command to edit any parts of a task.

Format: `edit` INDEX [*st*: NEW\_START\_TIME] [*et*: NEW\_END\_TIME]  
[*desc*: NEW\_DESCRIPTION] [*sd*: NEW\_START\_DATE] [*ed*: NEW\_END\_DATE]  
[*p*: NEW\_PRIORITY]

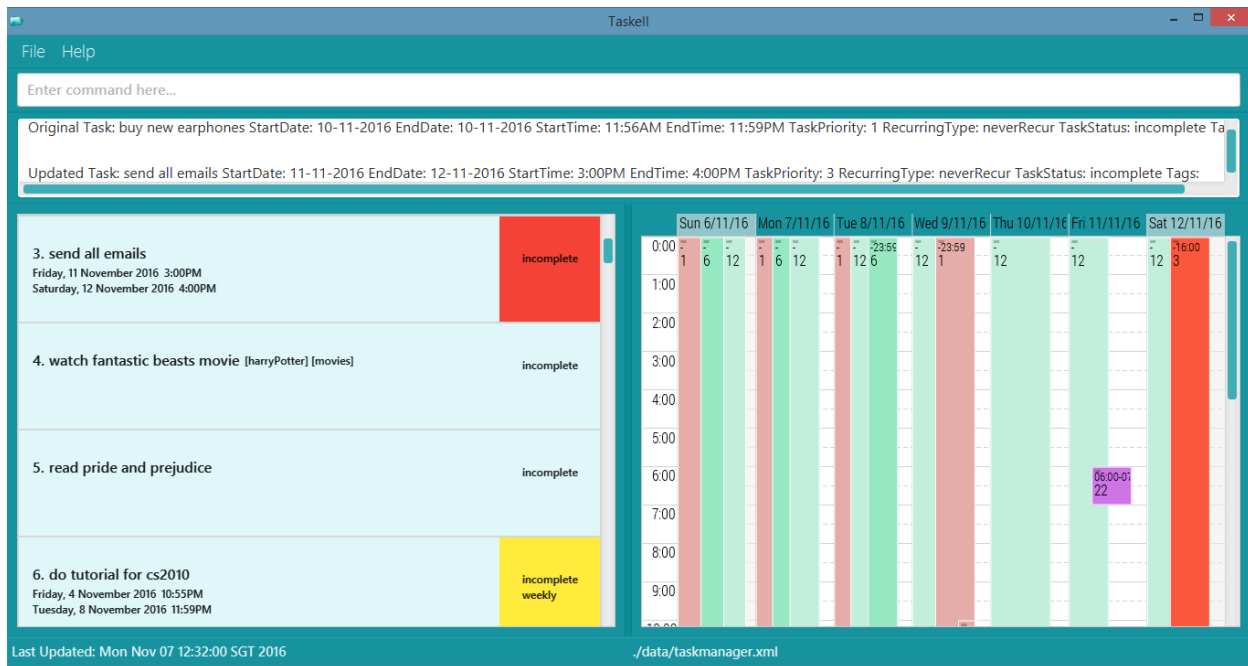


Figure 3: Edits the third task on the list

Entering '`edit` 3 *desc*: sends all emails *sd*: 11-11-2016 *ed*: 12-11-2016 *st*: 3pm *et*: 4pm *p*: 3', will update description to 'send all emails', start date to '11-11-2016', end date to '12-11-2016', start time to '3pm' end time to '4pm' and priority to '3'.

### Take note!

- You only need to key in the necessary parameters of the task you would to change. Not all parameters are required.
- Order of the parameters is not important.
- Editing of overdue tasks will result in both the date and time to be set to the default values, unless otherwise stated.

## Finding tasks: `find`

You can use the `find` command to view tasks with specific keywords.

Formats:

- `find` TAG [MORE\_TAGS]  
Displays a list of tasks with description or tags matching all the keywords.  
Example: `find` banana milk essay  
Displays a list of all the tasks with description or tags matching 'banana', 'milk', 'essay'.  

Take note!  
Tasks with words that match the keyword include those that contain the keyword.  
For example, searching for 'book' will match with 'book', 'textbook', 'storybook' etc.
- `find-tag` TAG [MORE\_TAGS]  
Displays list of tasks with the same tags.  
Example: `find-tag` homework essay cs2103  
Displays a list of tasks having tagged as either 'homework', 'essay' or 'cs2103'.

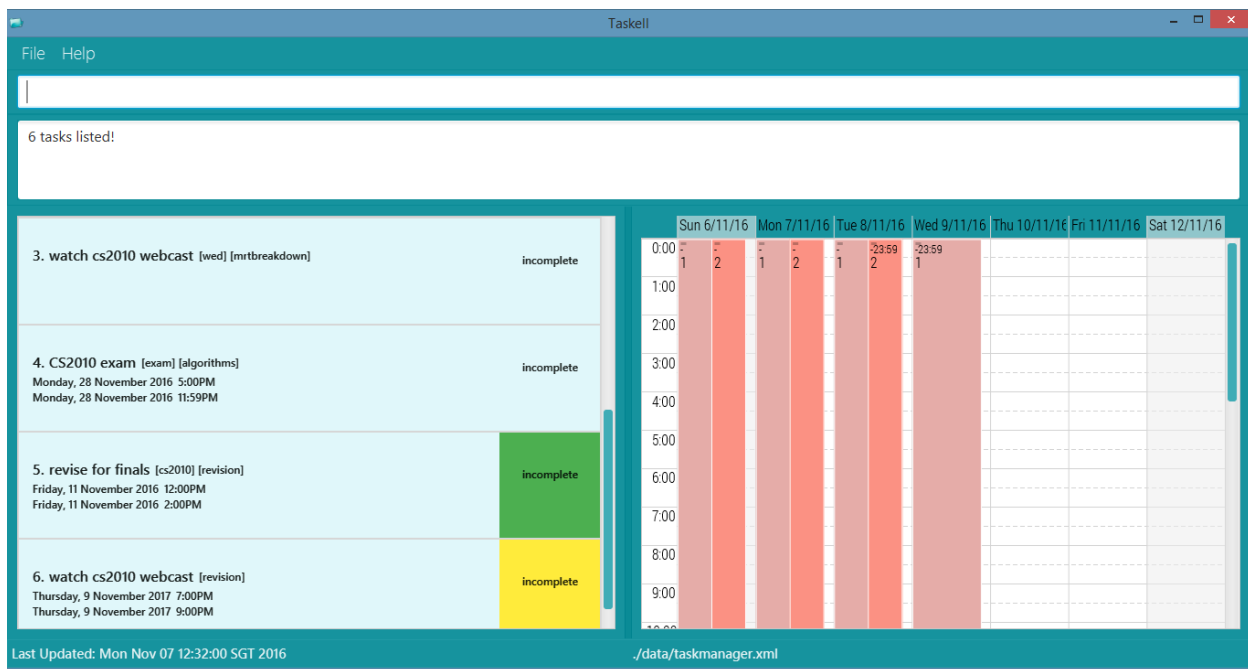


Figure 4: `find` CS2010 displays a list of tasks with 'CS2010' as one of the keywords in task description

Take note!

- The order of the keywords does not matter e.g. 'chicken egg' will match 'egg chicken'.
- Tasks matching at least one keyword will be displayed e.g. 'chicken' will match 'chicken duck'

## Deleting a task: `delete`

You can use the `delete` command to delete a task at a specified INDEX.

Format: `delete` INDEX

Example: `delete 3`

Deletes the third task shown in the list.

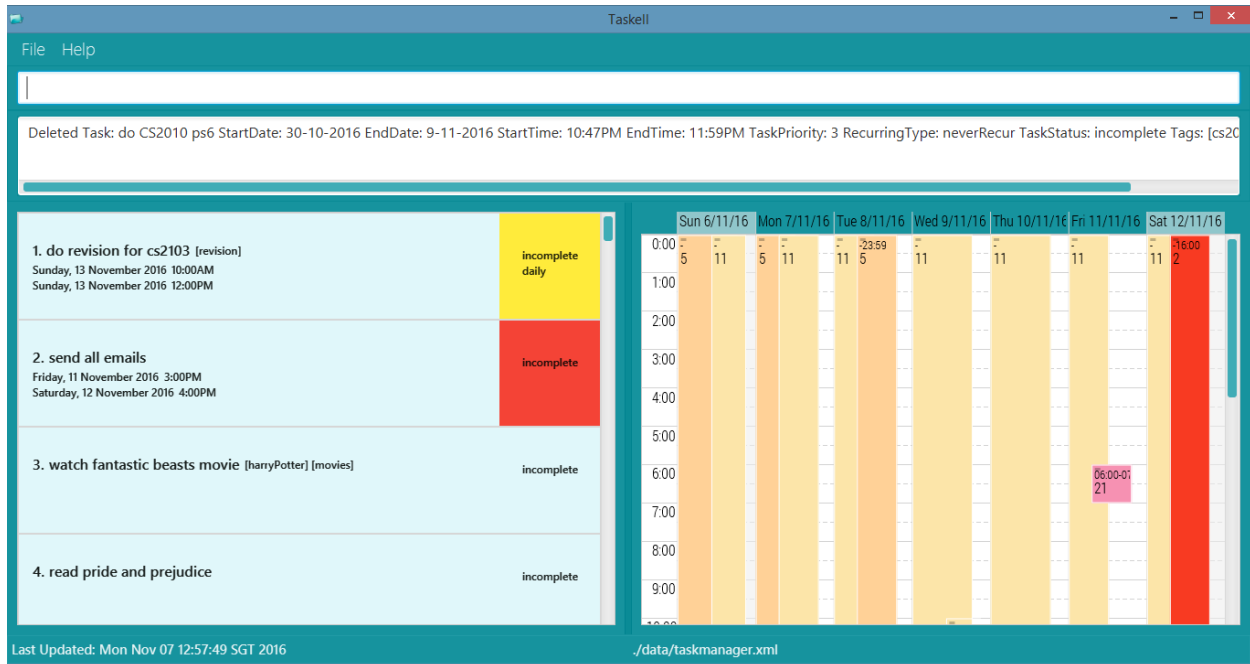


Figure 5: `delete 3` deletes the third task on the list

## Marking a task as completed: `done`

You can use the `done` command to mark an uncompleted task as completed.

Format: `done` INDEX

Example: `done 1`

Marks the first task as finished and moves it to the list of completed tasks.

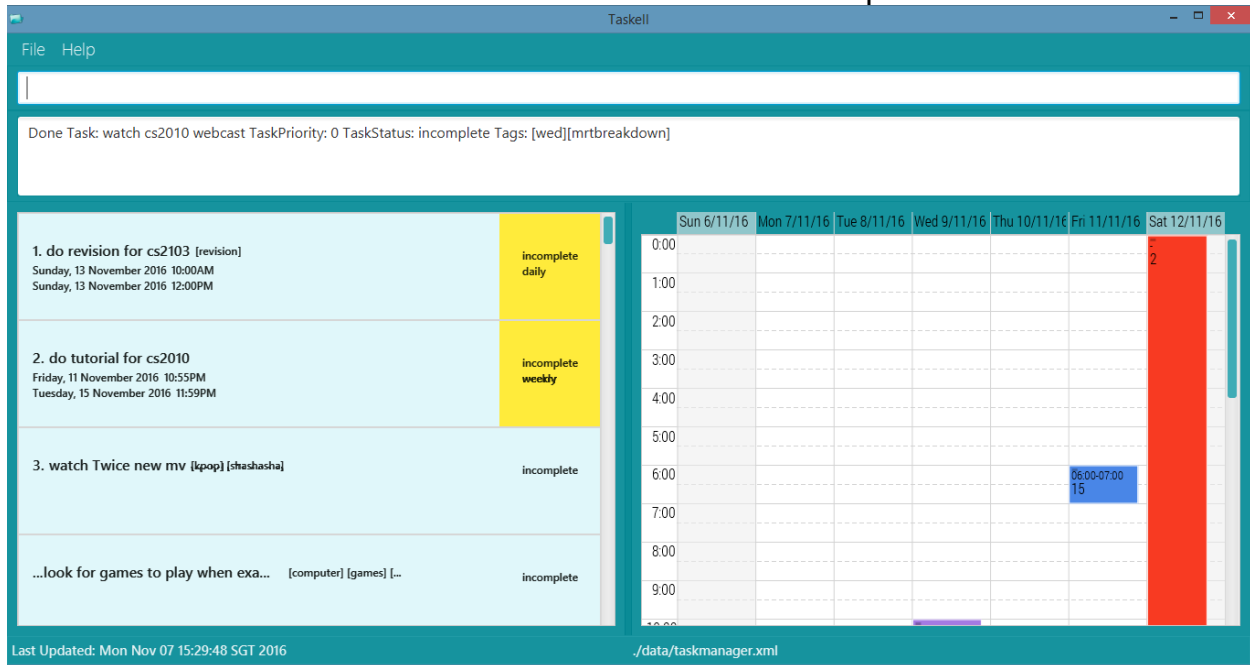


Figure 6: Marks the task as finished and moves it to the list of completed tasks

## Marking a task as completed: `undone`

You can use the `undone` command to mark a completed task as incomplete.

Format: `undone` INDEX

Example: `undone 1`

Marks the first task as incomplete and moves it to the list of uncompleted tasks.

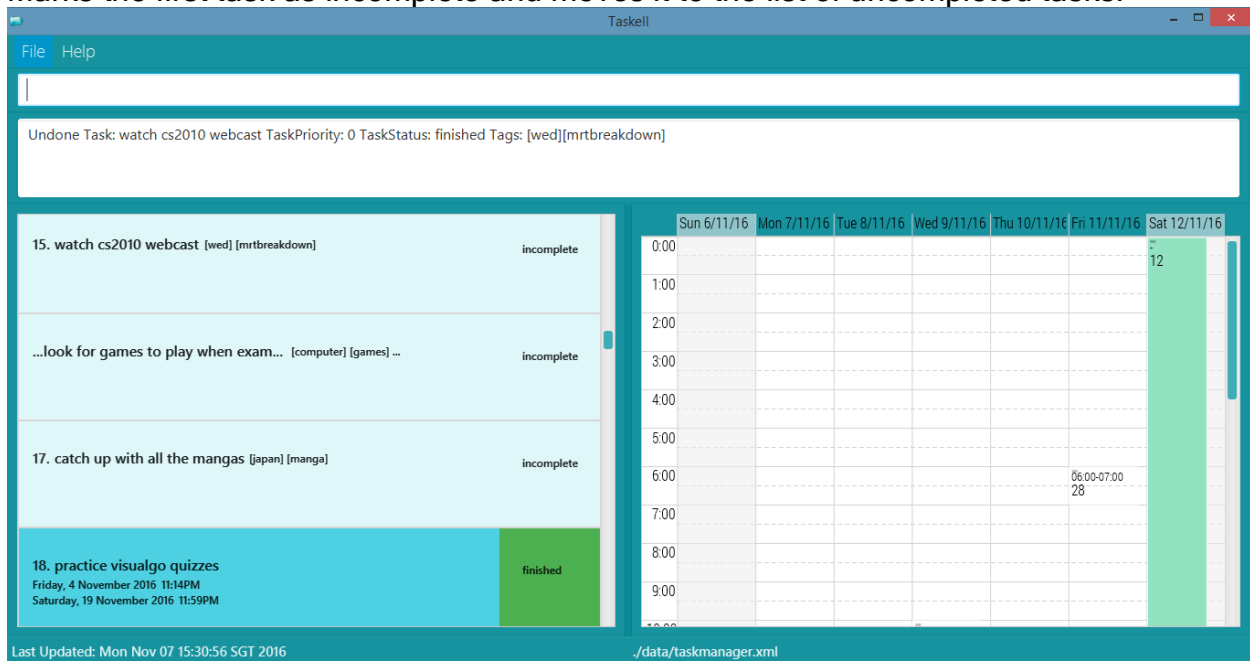


Figure 7: Marks the task as incomplete and moves it to the list of uncompleted tasks

## Listing tasks: `list`

You can use the `list` command to display a certain type of tasks  
Formats:

- `list`  
Displays a list of uncompleted tasks
- `list-all`  
Displays a list of all tasks, both completed and uncompleted
- `list-date` DATE  
Displays a list of all the tasks due on the specific date
- `list-done`  
Displays a list of completed tasks
- `list-priority` PRIORITY  
Displays a list of tasks with the specified priority

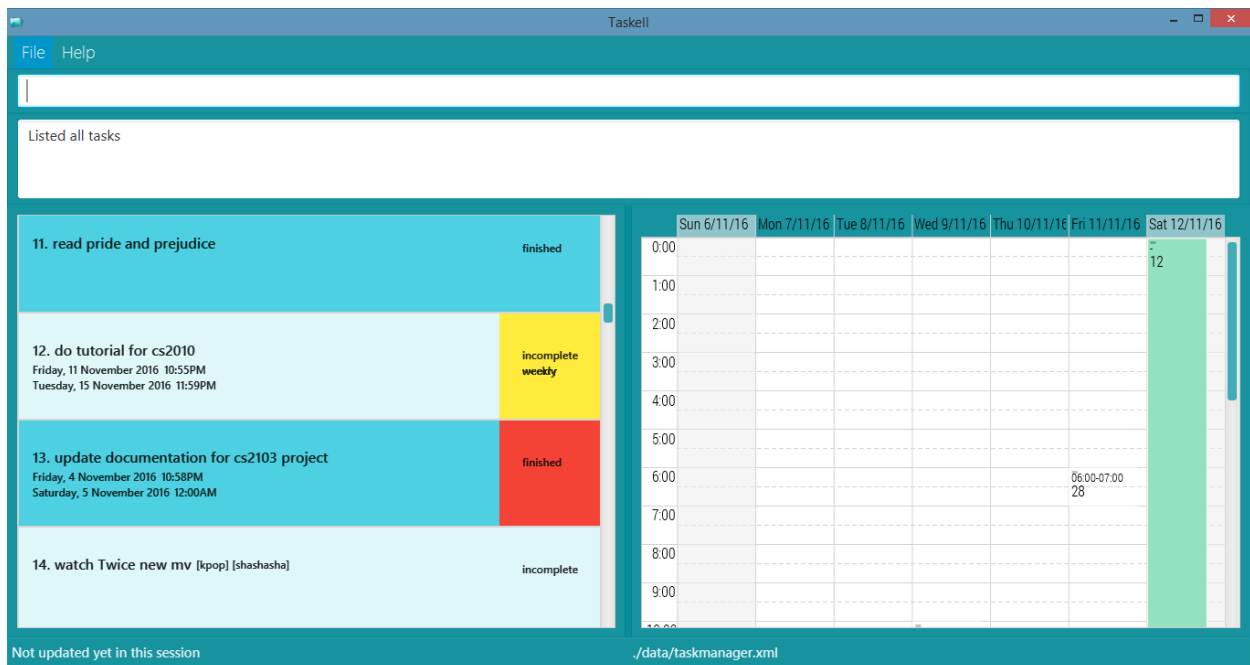


Figure 8: `list-all` displays both completed and uncompleted tasks

Clearing all entries: `clear`

You can use the `clear` command to clear all task data permanently.

Format: `clear`

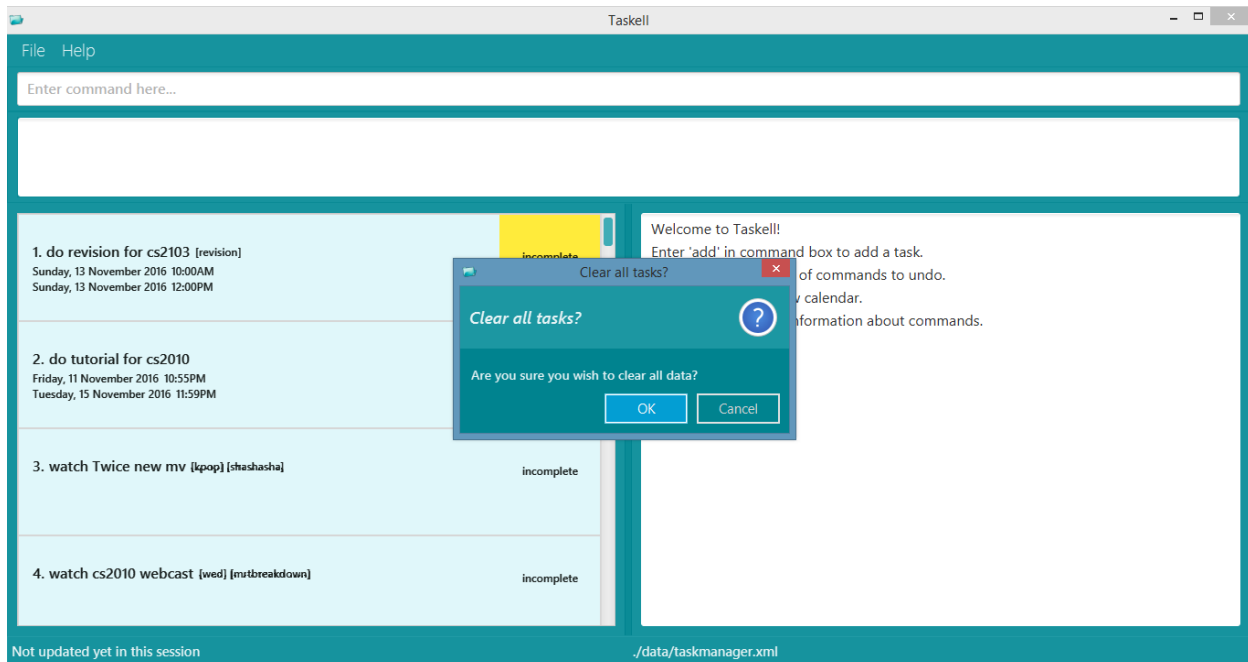


Figure 9: `clear` prompts a confirmation pop-up window

Take note!

`clear` command is irreversible!

## Showing history: `history` or `hist`

You can use the `history` command to view a list of actions that can be undone. The command history will be shown in the right panel.

Format: `history` or `hist`

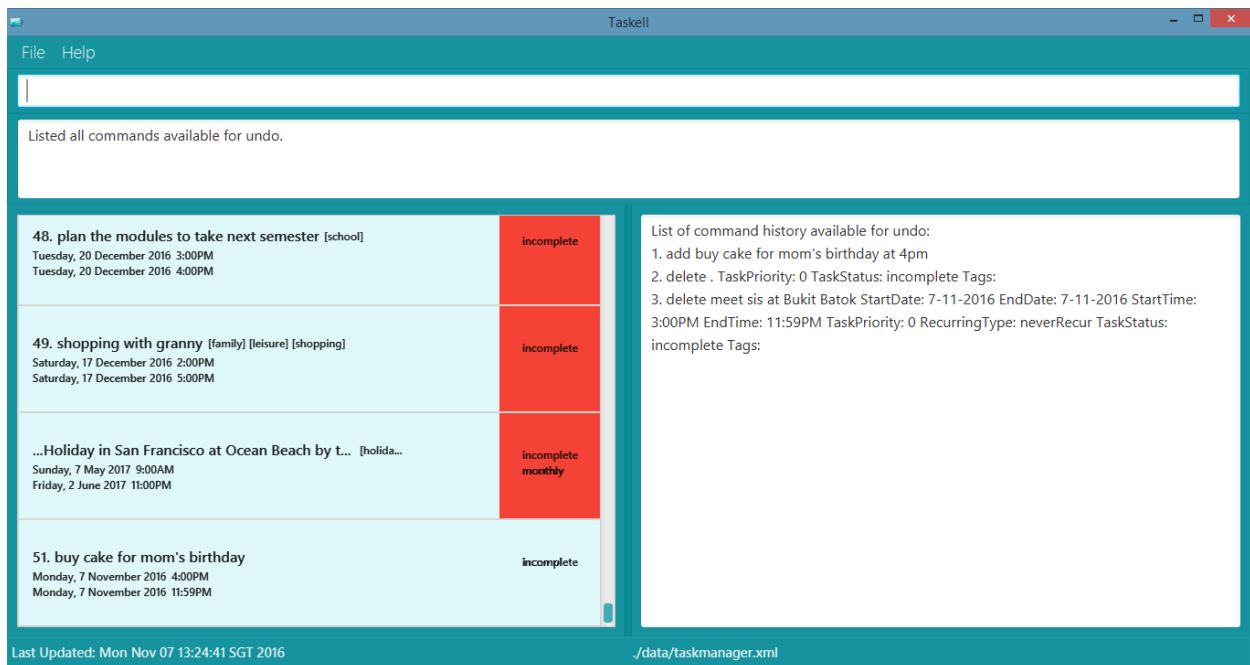


Figure 10: `hist` displays the command history on the right panel

### Take note!

- Only commands that are available for undo will be shown here.
- Refer to the `undo` section below to find out which commands can be undone.



## Reverting previous actions: `undo`

You can use the `undo` command to undo your previous actions.

Formats:

- `undo`  
Undo the most recent command executed
- `undo INDEX`  
Undo the command at the specified index in the command history.  
Example: `hist` , then `undo 3`, will undo the third command in the command history

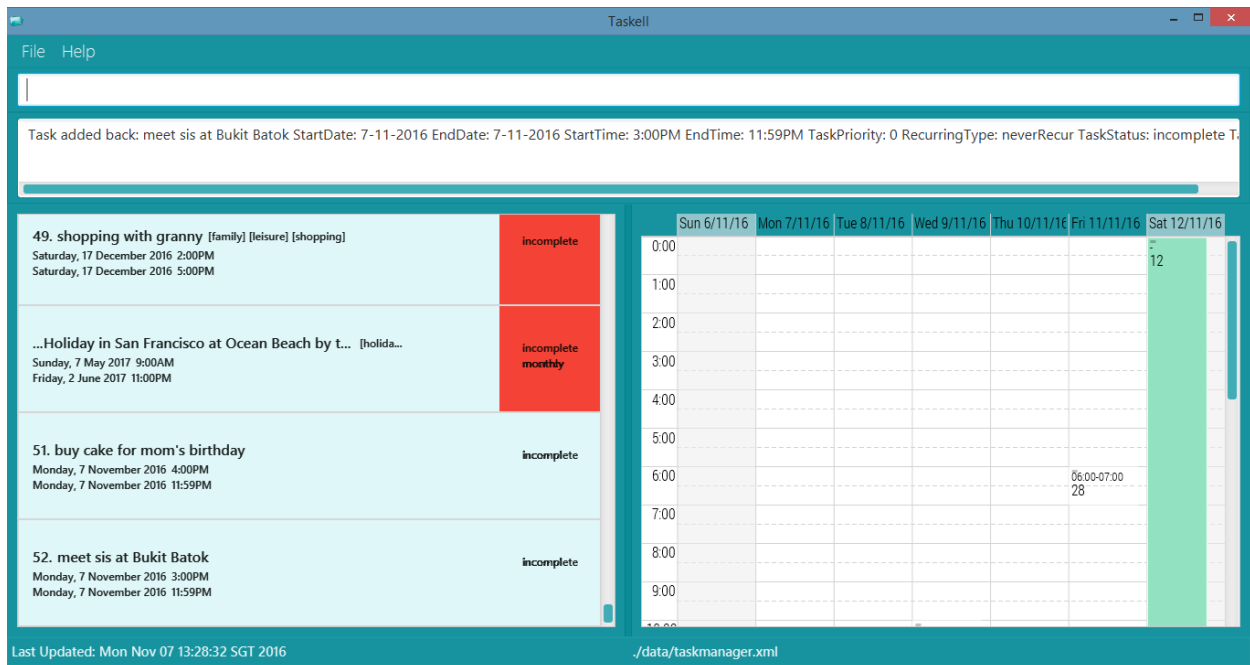


Figure 11: `undo` reverts the last item in the command history

Take note!

`undo` command only supports add, edit, delete, done, undone and undo commands.

Showing calendar view: `calendar` or `cal`

You can use the `calendar` command to view a calendar.

Format: `calendar` or `cal`

You can refer to the calendar on the right panel when adding tasks and scheduling events. A fine red line is used to indicate the current time.

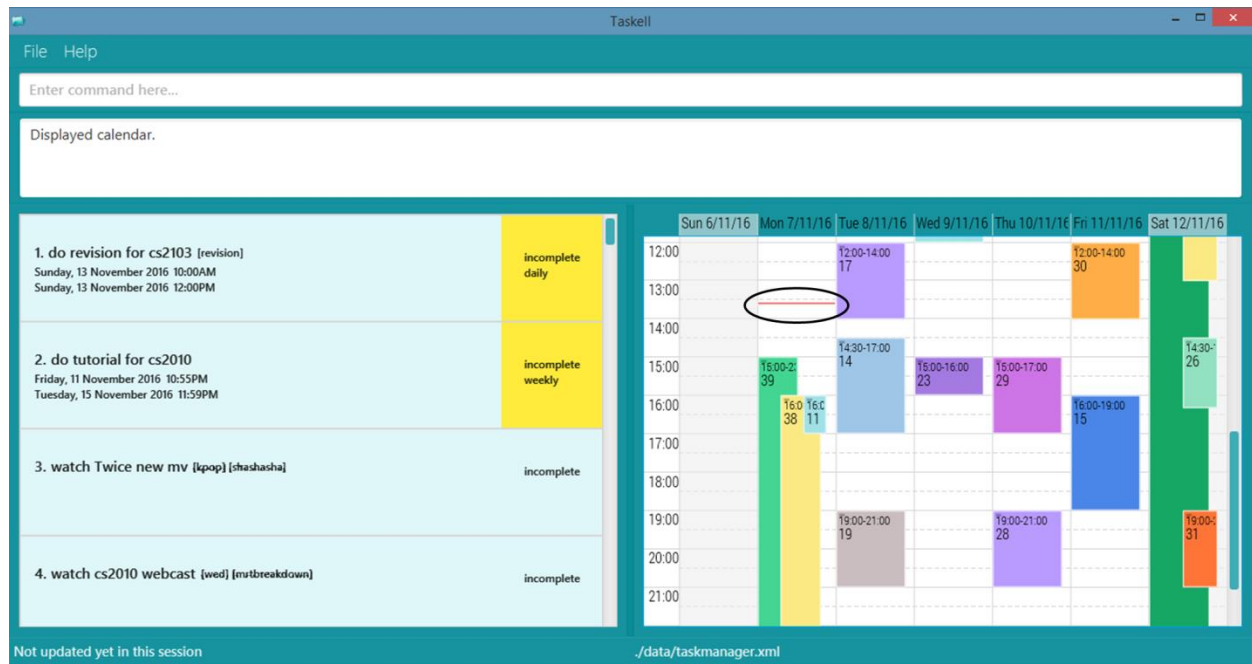


Figure 12: Example of how the current time marker looks like

The calendar view reflects the tasks shown in the left panel. Each block is marked with the index corresponding to the task, meaning a block marked '4' would correspond to the fourth task.

By default, the calendar view will be shown on the right panel. As the `history` command also utilizes the right panel to display the command history, you can use the `calendar` command to toggle between both views.

Take note!

Entering other commands (i.e. `find`, `add`) will revert the right panel back to the calendar view as it is the default view.

## Saving the information in Taskell: `save`

You can use the `save` command to specify the path of a folder to store Taskell's data file. Please note that you should have permissions to access the folder.

Format: `save` FILE\_PATH

Example: `save` C:\Users\Jim\Documents

To obtain the file-path, navigate to the required file in your File Explorer. Copy the path at the top of the screen and paste into Taskell. Refer to diagram below for an example of a file-path

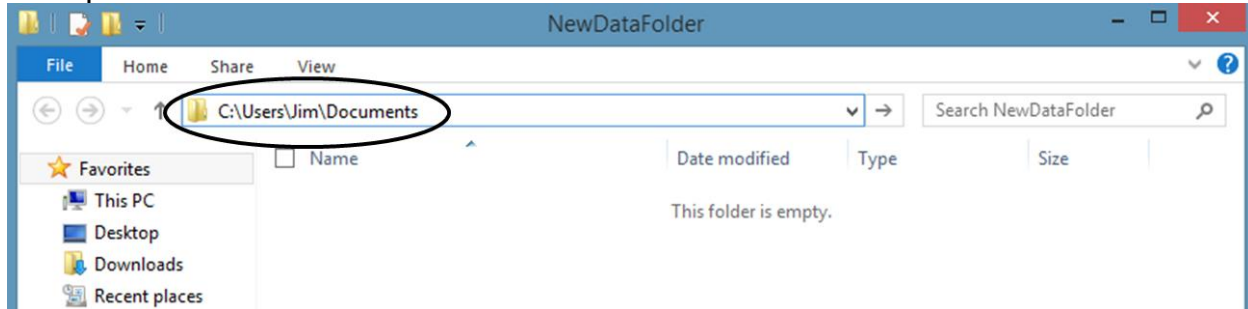


Figure 13: Screenshot of File Explorer in Windows

If the specified directory is valid but the folder is missing, for example if command is `save` C:\Users\Jim\Documents\Project, and C:\Users\Jim\Documents\Project is valid but 'Project' folder is not created, Taskell will create the folder for you.

### Take note!

- If you only specify a folder name without directory, i.e. `save` project, Taskell will create a folder named 'project' within Taskell's own directory. Whereas `save` C:/Users/Jim/Documents/project will open a folder named 'project' within your desktop's Document's folder.
- Both Windows and Linux OS have restricted symbols that are not allowed for folder names. Please be aware of the symbols shown in the table below.

Windows	Linux
>	>
<	<
:	:
"	&
/	
\	
?	
*	

Table 1: Restricted symbols in Windows and Linux

## Exiting Taskell: `exit`

You can use the `exit` command to exit Taskell.

Format: `exit`

Alternatively, you can hold down `Alt` + `F4`

## Frequently Asked Questions

**Q:** How do I transfer my data to another computer?

**A:** Install the application in the other computer and overwrite the empty data file it creates with the file that contains the data in your previous Taskell folder.



**Q:** Do I have to save the data every time I enter new tasks?

**A:** No, Taskell automatically saves your data every time you enter new tasks. Use `save` only when you want to transfer your data to a new location on your computer.

## Glossary

1. GUI: Graphical User Interface
2. Floating task: A task without date and time

# Command Summary

Command	Format
Add	add TASK [ <i>p</i> / PRIORITY] [#TAG] add TASK <i>by</i> DATE add TASK <i>by</i> TIME add TASK <i>by</i> DATE <i>by</i> TIME add TASK <i>at</i> TIME add TASK <i>on</i> DATE add TASK <i>on</i> DATE <i>by</i> TIME add TASK <i>on</i> DATE <i>at</i> TIME add TASK <i>from</i> DATE <i>to</i> DATE add TASK <i>from</i> TIME <i>to</i> TIME add TASK <i>on</i> DATE <i>from</i> TIME <i>to</i> TIME [ <i>p</i> /PRIORITY] [#TAG] [ <i>r</i> /RECURRING]
Calendar view	calendar or cal
Clear	clear
Delete	delete INDEX
Edit	edit INDEX <i>desc</i> : NEW_DESCRIPTION <i>sd</i> : NEW_START_DATE <i>st</i> : NEW_START_TIME <i>ed</i> : NEW_END_DATE <i>et</i> : NEW_END_TIME <i>p</i> : NEW_PRIORITY
Exit	exit  + 
Find	find KEYWORD [MORE_KEYWORD]
Find by tag	find-tag TAG [MORE_TAG]
Help	help
History	history or hist
List all tasks	list-all
List by priority	list-priority PRIORITY
List by specified date	list-date DATE
List completed tasks	list-done
List uncompleted tasks	list
Mark task as finished	done INDEX
Mark task as incomplete	undone INDEX
Save	save FILE_PATH
Undo	undo undo INDEX

## Date Format Summary

Supported Date Format	Example
DD-MM-YYYY	1-1-2016 1/2/2016 1-mar-2016 1-April-2016 1.May.2016 1.Jun.2016
MM-YYYY	Jul-2016 july-2016
MM	jan sept December
day	today tdy tomorrow tmr thursday thurs thu

## Time Format Summary

Supported Date Format	Example
In 12-hour format	12am 5:30am 1pm 10-35pm 11.45pm
In words	midnight afternoon noon