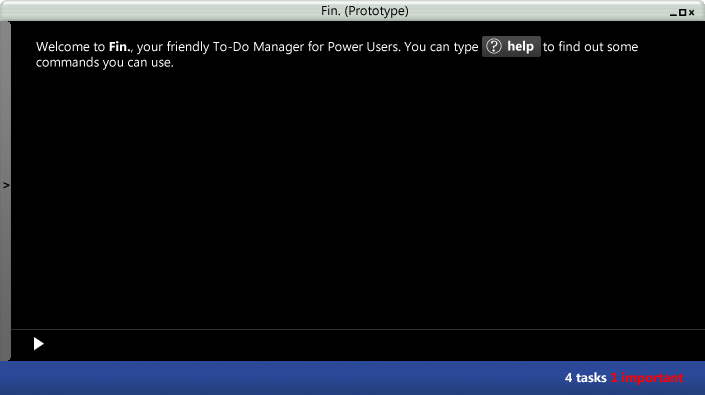
Fin.



|  |  |  |  |
| --- | --- | --- | --- |
| **http://a2.sphotos.ak.fbcdn.net/hphotos-ak-snc3/17169_1279600164243_1657221838_1386776_1920415_n.jpgAlex Liew**  Team Lead Coder & UI Design | **http://a6.sphotos.ak.fbcdn.net/hphotos-ak-snc6/281303_10150335148441337_717631336_9216185_1999747_n.jpgKoh Zi Chun**  Tester Coder | **http://a6.sphotos.ak.fbcdn.net/hphotos-ak-ash2/31763_10150196755330367_885040366_13055568_5579080_n.jpgJoe Chee**  Code Maintenance Coder | **http://a7.sphotos.ak.fbcdn.net/hphotos-ak-snc6/34152_134662803229821_100000583054500_274648_880008_n.jpgWei Jing**  Administration Coder |

# Product: Fin.

## Credits

#### Fin. uses the SnakeYAML libraries to store our data in human-readable format and JUnit to set up test cases for our product.

# Fin User Guide

Fin. is a ToDo manager that will allow you to keep track of your daily tasks the way you want it to be tracked.

Currently, Fin. is a command-line text parser that would store your data in human-readable format. Essentially, one can just open up the calendar file to see the tasks that you need to complete.

### Launching Fin

Command usage: Fin [-file <taskfile>] where taskfile is the files you want to use.

You should then be taken to this welcome screen and a list of tasks that you have done:

<Insert screenshot of command prompt here>

### Commands

You would then be presented with a command prompt in which to type commands. The commands are as follows:

|  |  |
| --- | --- |
| add <task> | Adds a task to your task list |
| show <filters> | Shows the task list with <filters>. Examples of <filters> can be “tasks due on Friday” |
| delete <task> | Deletes a task from your task list |
| edit <task> | Edit a <task> |
| tag <task> <tag> | Tag a <task> with <tag> |
| fin <task> | Mark a task as complete |
| joke | Generates a joke for you |

**Tasks**

You can specify the following attributes for a task:

1. Task Name: the name of a task. Task Name is set as the first Tag, or the front part of the user input which is not prefixed by other keywords. For example, the Task Name for “by 12:00 next Wed #CS2103 Project #Fin Manual ^important” is “#CS2103 Project”. For “Project CS2103 #Fin Manual ^important by 12:00 next Wednesday”, the Task Name is also “#Project CS2103”.
2. Tags: a tag is prefixed by a ‘#’, for example, “#difficult homework”. A task can have multiple tags.
3. Due Time and Date: the due time and date of a task, which is prefixed by “due”, “before” or “by” (in either case), where the Due Date comes right after the Due Time. Due Time is in format of “HH:MM” or “HH:MM:SS” or non-specified (by default is the last second 23:59:59). Due Date can be a single word “today” or “tomorrow”, or in the form of “in <amount> days/weeks/months”, “next/this week/weekend”, “next/this <day of week>”, “<MM> <DD> <YYYY>”, “<MM> <DD>”, or non-specified (by default is today). <MM> is a word from {January, February, …, December} or their prefixes.

For example, you can type “before next weekend #CS2103 Project”, and the due Time and Date are set as the last second (23:59:59) of next Sunday. You can also type “DUE 12:00 Oct 29 2011 #CS2220 Assignment” and the Due Time and Date will be specified as desired.

An explicit example is “due in 4 weeks #cs2220 assignment 3 #hard #10 marks ^important”. However, if neither Time nor Date is specified, the default value is the current time.

1. Importance: importance is leaded by ‘^’, and it is a word (in either case) either in {“high”, “low”, “normal”, “important”} or their prefixes. For example, “^H” means the Importance is high. By default, the importance is normal.
2. Priority: priority is leaded by “\*” and it is a non-negative integer. For example “\*98”. By default, the priority is 0.

# Fin Developer Guide

Thanks dudes for checking out how you are able to help out in the Fin. project to create the best ToDo manager for geeks!

TTD:

* What do we know about the system at this point?
* Which part of that knowledge will be useful for the next team who will be taking over?
* What is the best way to pass that knowledge to them? Note that some information can be passed on in the form of code without any additional documentation.
* Architecture
* important APIs
* Design descriptions (class diagrams, sequence diagrams, notable algorithms ...)
* Code examples
* Instructions for testing

#### Development Plan

**v0.1 – Week 9**

**Core**

- Adding tasks – simple adding of tasks

- Tags for tasks – Organize tasks with tags

- Current tasks – Normal view will display only currently undone tasks

- Edit tasks – Ability to modify tasks

- Scheduling of tasks – tasks scheduled by due date instead of entry order

- Check off tasks – Check off tasks to remove them from current tasks

- All tasks – Listing of all tasks present

- Completion percentage for tasks

- Task summary – Short summary of all current tasks

- Relative dates – Due dates displayed with respect to current date

- Delete tasks – Removes tasks from history altogether

- Arrange tasks – Default sorting is by due date; allow arranging tasks in arbitrary manner

- Find tasks – search for tasks containing given search parameter

- Sub tasks – break tasks into sub tasks

- Alarm option

- Help command – help page for each command

**v0.2 – Week 12**

**GUI**

- UI Design

- GUI analog to all CLI commands

- GUI interaction via Mouse

- Keyword highlighting

- Autocomplete