Billboard - User guide

By: `CS2103T-F12-4` Since: `Sep 2019` Licence: `MIT`

1. Introduction

Billboard is for those who prefer to use desktop applications to manage and keep track of their expenses. *Billboard* is optimized for those who prefer to work with a Command Line Interface (CLI) while still enjoying all the benefits of having a fully fledged Graphical User Interface (GUI). If you can type fast, *Billboard* can help you manage your expenses faster than traditional GUI apps!

2. Quick Start

1. Ensure you have Java 11 or above installed on your machine.
2. Download the latest version of *Billboard* here.
3. Copy the file to the folder you want to use as the Home folder for *Billboard*
4. Double-click the file to open the application
5. Refer to Section 3, “Features” for the details of each command

3. Features

Command Format

- Words in `[SQUARE BRACKETS]` are the mandatory parameters to be supplied by the user e.g. in `expense add [DESCRIPTION] [AMOUNT]`, `DESCRIPTION` and `AMOUNT` are parameters which can be used as `expense add "buy a book" 10`.
- The mandatory parameters must be supplied in the order specified.
- Default values can be specified in the square brackets, e.g `[NUM_ENTRIES, DEFAULT = 10]`. If the user does not provide a value for that parameter, it will default to the default value stated.
- Some commands may have optional parameters denoted under options. These can be included by specifying a flag, usually of the format `--flag` or `-f`. E.g `--description` can be used as `expense edit 2 --description "changed description"`.
- Flags can be in any order.
- Items with ⋯ after them can be used multiple times including zero times e.g. `[TAG]⋯` can be used as (i.e. 0 times), `bills`, `bills fees` etc.

3.1. Managing expenses: `expense`

1. *Add a new expense: expense add*
   Adds an expense to records.
   Usage:

   ```
   expense add [DESCRIPTION] [AMOUNT]
   ```

   Example:
   `expense add "buy a book" 10`
   Adds an expense to record where the description is “buy a book” and the amount is 10.

2. *Edit an existing expense: expense edit*
   Edits an existing expense that is being displayed on the GUI. Either description or amount must be supplied.

   Usage:

   ```
   expense edit [INDEX] [OPTIONS]
   ```

   Options:

   - `--description`, `-d`: Specify new description to update expense with.
   - `--amount`, `-a`: Specify new amount to update expense with.

Example:
`expense edit 2 -d "buy a math book"`
Edit the expense at index 2 with new description of: “buy a math book”

NOTE	This command overwrites existing tags if you input tags.

1. *Delete an existing expense: expense rm*
   Deletes an existing expense that is being displayed on the GUI.

   Usage:

   ```
   expense rm [INDEX]
   ```

   Example:
   `expense rm 3`
   Deletes the expense with index 3.

2. *List all expenses: expense list*
   Lists all the expenses in the record list.

   Usage:

   ```
   expense list
   ```

   Example:
   `expense list`
   Displays below expenses (for example) in the GUI:

   - buy a book / amount: $10
   - pay for dinner / amount: $5

3.2. Tagging expenses: `tag`

This feature allows you to assign tags to your expenses. Expenses with the same tag will be grouped together, allowing you to search for related expenses easily.

NOTE	Tag names should be **alphanumeric**. This means that they should not contain spaces or special characters. Tag names are **case sensitive.** You are **not** able to use this feature on **archived expenses**.

To view a list of supported tag commands, you could type `tag` in the command box and press `enter`.

Similarly, to view parameters for supported tag commands, you could type `tag` followed by the supported tag command and press `enter`.
Example: `tag rm`

3.2.1. Adding a tag: `tag add`

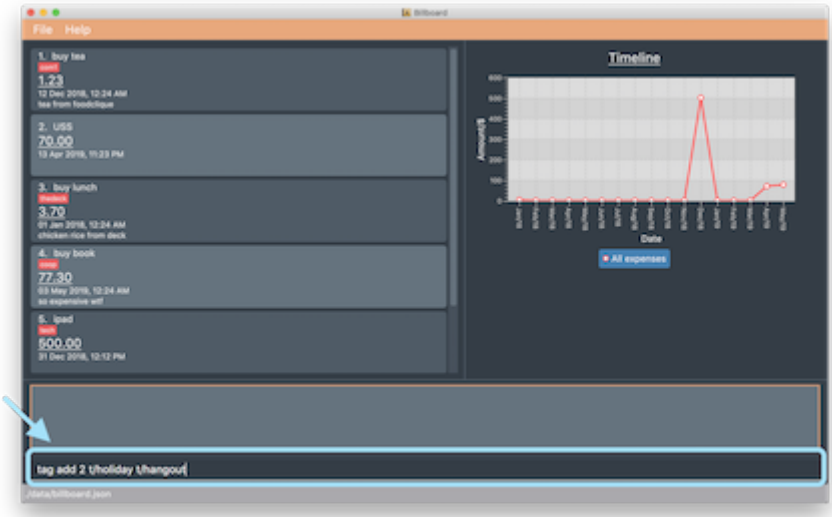
This command adds your input tag(s) to the expense at the index you have specified.

Usage:

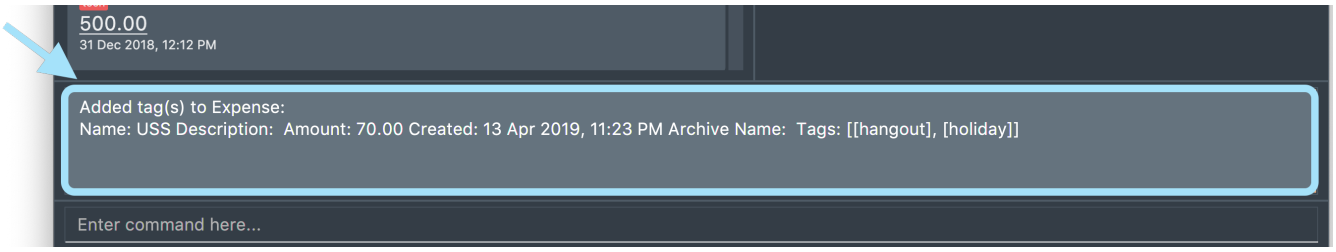
```
tag add [INDEX] t/[TAG] t/[TAG…]
```

Example:
Lets say that you want to add the tags "*holiday*" and "*hangout*" to the expense at index 2 which is "*USS*".

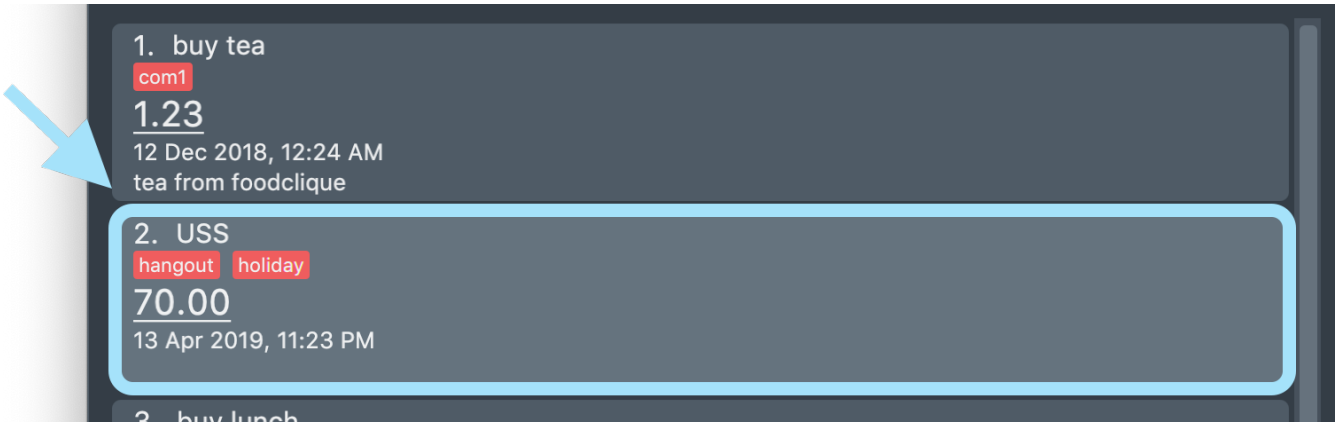
To add the tags:
1. Type `tag add 2 t/holiday t/hangout` in the command box and press `enter` to execute it.



2. The result box will display the message "Added tag(s) to Expense:
Name: USS Description: Amount: 70.00 Created: 13 Apr 2019, 11:23 PM Archive Name: Tags: [[hangout], [holiday]]"



3. You could see the tags "*holiday*" and "*hangout*" in the expense at index 2.



NOTE	You are not able to add **existing** tags. If you input **existing** and **non-existing** tags, Billboard adds the **non-existing** tags only. If you input **duplicate** tags, Billboard adds them **once**. i.e Billboard does **not** allow Duplicate tags in an expense. Example: `tag add t/holiday t/holiday`

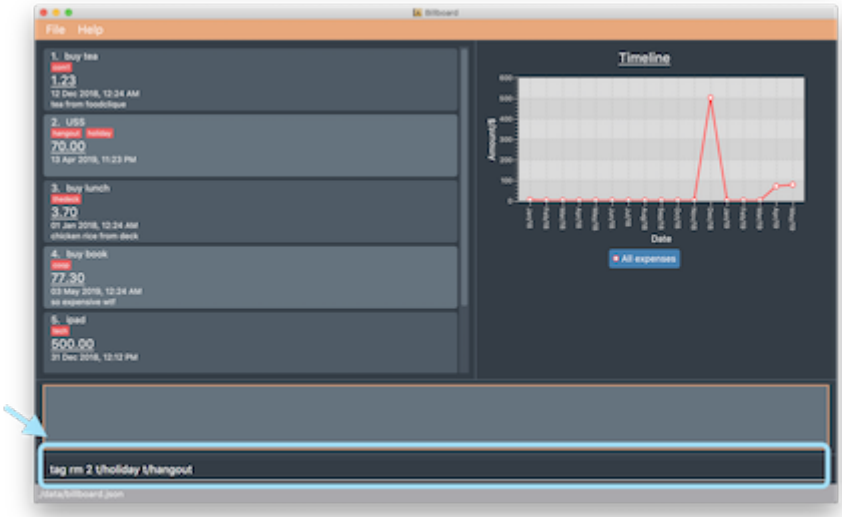
3.2.2. Removing a tag: `tag rm`

This command removes your input tag(s) from the expense at the index you have specified.

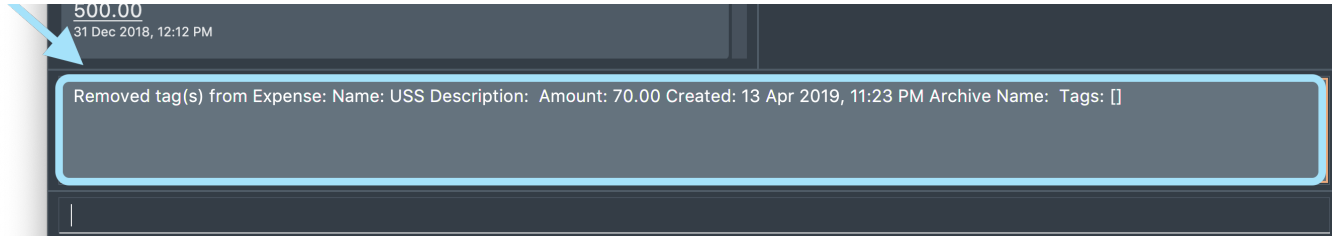
Usage:

```
tag rm [INDEX] t/[TAG] t/[TAG…]
```

Example:
Lets say that you want to remove the tags "*holiday*" and "*hangout*" from the expense at index `2` which is "*USS*".
To remove the tags:
1. Type `tag rm 2 t/holiday t/hangout` in the command box and press `enter` to execute it.



2. The result box will display the message "Removed tag(s) from Expense:
Name: USS Description: Amount: 70.00 Created: 13 Apr 2019, 11:23 PM Archive Name: Tags: []"



3. You could see that the expense at index 2 no longer has the tags "*hangout*" and "*holiday*".

NOTE	You are not able to remove **non-existing** tags. If you input **duplicate** tags, Billboard removes them **once**. Example: `tag rm t/holiday t/holiday`

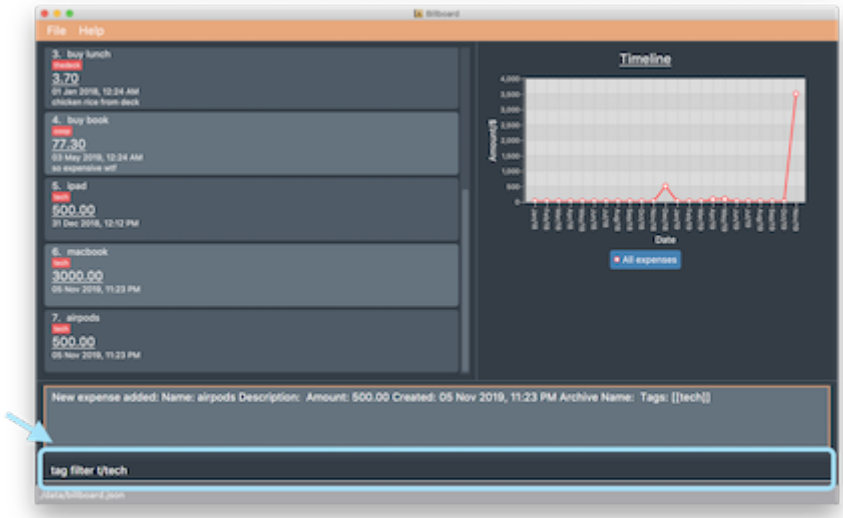
3.2.3. Filtering by tag: `tag filter`

This command filters expenses by your input tag(s).

Usage:

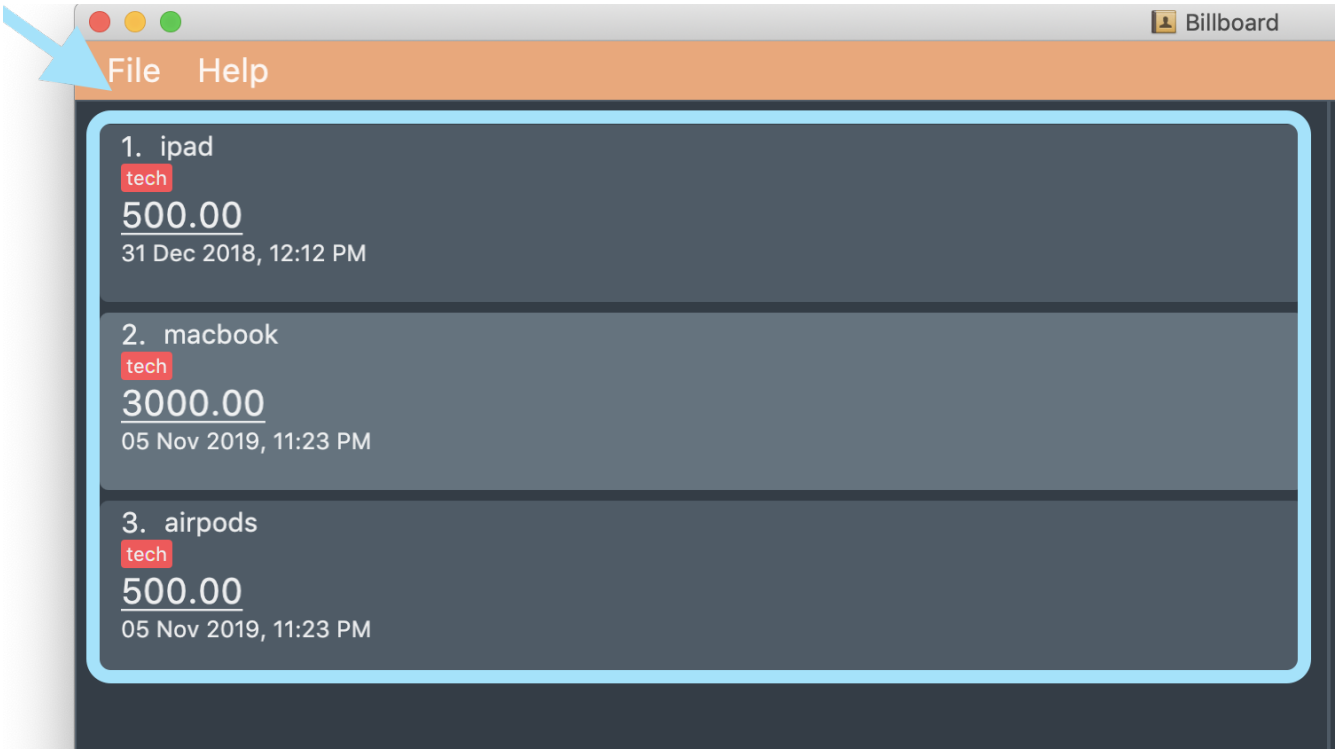
```
tag filter t/[TAG] t/[TAG…]
```

Example:
Lets say that you want to filter your expenses by the tag "*tech*".
To filter your expenses:
1. Type `tag filter t/tech` in the command box and press `enter` to execute it.



2. The result box will display the message "3 expense(s) listed!"

3. Billboard lists all expenses under the tag "*tech*".



NOTE

If you input **more than one** tags, Billboard lists out all expenses tagged with **one or more** input tags.
This command allows **duplicate** tags as input.
This command allows **non-existing** tags as input.
After executing this command, you could edit (eg. `edit`, `tag add` etc) the filtered expenses using the displayed indexes.

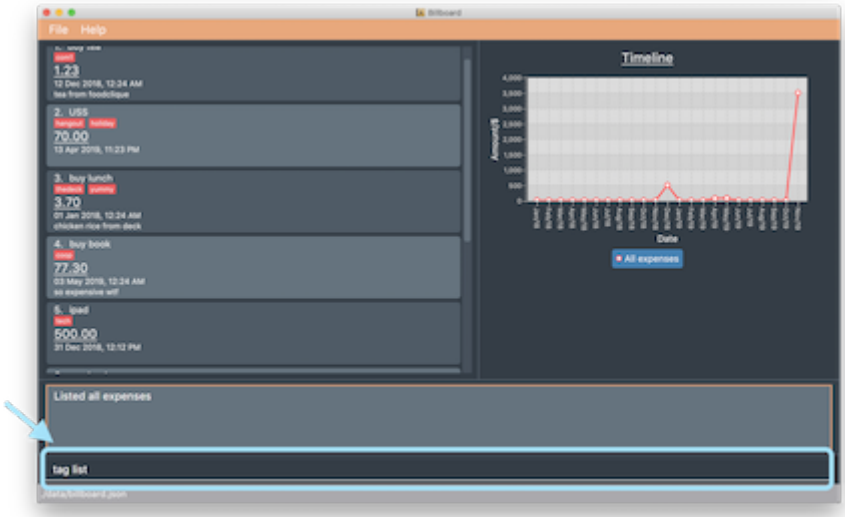
3.2.4. Listing out all the tags: `tag list`

This command lists out all existing tags.

Usage:

```
tag list
```

Example:
Lets say that you want to list out all existing tags.
To list them:
1. Type `tag list` in the command box and press `enter` to execute it.



2. The result box displays a list of all existing tags.

3.3. Finding expenses by attributes: `find`

1. *Find : find*
   Find expenses that satisfies conditions specified in the arguments.

   Usage:

   ```
   find [d/KEYWORD[ KEYWORD…]] [a/LOWER AMOUNT LIMIT] [al/UPPER AMOUNT LIMIT]
   [sd/START DATE] [ed/END DATE]
   ```

   Example:
   `find d/lunch book a/2 al/20 sd/1/1/2019 ed/1/10/2019`
   Finds all expenses that contain the keyword ‘lunch’ or 'book' and amount between 2 and 20 dollars and date between 1 Jan 2019 to 1 Oct 2019 in any of its attributes.

3.4. Sorting expenses by attributes: `[Coming in v1.3]`

1. *Sort by name:*
   Sort expenses by name in alphabetical order.

   Usage:

```
sort name
```

2. *Sort by date:*
   Sort expenses by date from newest to oldest

   Usage:

   ```
   sort date
   ```

3. *Sort by amount*
   Sort expenses by amount of expense in from largest to smallest.

   Usage:

   ```
   sort amt
   ```

3.5. Archiving past expenses: `archive`

1. *Adding a record to an archive: archive add*
   Transfers the expense at the specified index to your specified archive.
   If archive you entered does not exist, then the new archive is created before the expense is added.

   Usage:

   ```
   archive add [INDEX] [OPTIONS]
   ```

   Options:

   - `/arc`: Specifies the name of the archive to add the expense to. (REQUIRED)

   Example:
   `archive add 3 arc/MBS casino winnings`
   Archives the record at the 3rd index into an archive named “MBS casino winnings”

2. *Listing out all archives: archive listall*
   Displays the list of all archive names.

   Usage:

```
archive listall
```

3. *Listing records in a particular archive: archive list*
   Displays the list of records in your specified archive

   Usage:

   ```
   archive list [NAME]
   ```

   Example:
   `archive list 2018 expenses`
   Lists out all the records in the “2018 expenses” archive

4. *Deleting an archived record: archive delete*
   Deletes the record at the specified index from your specified archive.
   If the archive record deleted was the last record in the archive, the empty archive will be deleted.

   Usage:

   ```
   archive delete [INDEX] [OPTIONS]
   ```

   Options:

   - `arc/`: Specifies the name of the archive to delete the expense from. (REQUIRED)

   Example:
   `archive delete 5 arc/2018 expenses`
   Deletes the record at the 5th index in the “2018 expenses” archive

5. *Revert/unarchive an archived record: archive revert*
   Unarchives the record at the specified index from your specified archive, transferring it back to your current list of expenses.
   If the archive record reverted was the last record in the archive, the empty archive will be deleted.

   Usage:

   ```
   archive revert [INDEX] [OPTIONS]
   ```

   Options:

- `arc/`: Specifies the name of the archive to revert the expense from. (REQUIRED)

Example:
`archive revert 5 arc/2018 expenses`
Unarchives the record at the 5th index in the “2018 expenses” archive

3.6. Displaying of statistics: `display-stats`

1. *Display timeline overview of expenses: display-stats timeline*
   You can view a timeline overview of your currently displayed expenses. The expenses over a specified time period are aggregated and added as data points on the timeline.

   Usage:

   ```
   display-stats timeline [OPTIONS]
   ```

   Options:

   - `interval/`: Specifies the date interval to be used to split the expenses into groups by. Supported date intervals include 'day', 'week', 'month' and 'year'.

   Example:
   `display-stats timeline interval/week`
   Displays a timeline overview of the displayed expenses with the expenses being grouped into weeks.

2. *Display breakdown of expenses: display-stats breakdown*
   You can view a breakdown of expenses by tag for the currently displayed expenses. The total expenses per tag are totalled and displayed in a pie chart.

   Usage:

   ```
   display-stats breakdown
   ```

   Example:
   `display-stats breakdown`
   Shows a pie chart breakdown by tag of all currently displayed expenses.

3. *Display heatmap of expenses: display-stats heatmap*
   You can view a heatmap of expenses per day for the currently displayed expenses, limited to the past year. The larger the bubble for the particular day, the higher the expense.

Usage:

```
display-stats heatmap
```

Example:
`display-stats heatmap`
Shows a bubble chart heatmap of all currently displayed expenses, limited to a year.

3.7. Undo and Redo: `undo/redo`

1. *Undo the previous action: undo*
   Undo will restore the previous billboard state from state history. Undo will ignore all arguments.

   Usage:

   ```
   undo
   ```

2. *Redo the previous undo action: redo*
   Redo will restore a previously undone billboard state from state history. Redo will ignore all arguments.

   Usage:

   ```
   redo
   ```

3.8. Viewing History: `history`

1. *View the past command history: history*
   History will show all previous command histories. History will ignore all arguments.

   Usage:

   ```
   history
   ```

3.9. Up and down key: <code>↑/↓</code>

1. Get the previous entered command: ↑
 Up arrow key(↑) will get the previous command entered in the command history on the text field.

2. Get the sequential entered command: ↓
 Down arrow key(↓) will get the sequential command entered in the command history on the text field.

3.10. Viewing help: `help`

1. *Help list of complete set of commands: help*
   Shows the complete list of commands and instructions/description on how to use them.

   Usage:

   ```
   help
   ```

2. *Help list of a specific command: help*
   Shows the description and instructions on how to use the specified command.

   Usage:

   ```
   help [COMMAND]
   ```

   Example:
   `help archive`
   Shows the help message for the archive commands.

3.11. Exporting data: `[Coming in v2.0]`

{explain how the user can enable/disable data encryption}

4. FAQ

Coming soon!