BoNUS - User Guide

By: `CS2103T T09-B3 Team` Since: `Sep 2017` Licence: `MIT`

1. Introduction

As the name implies, **BoNUS** helps you *better(B) organize(o) your NUS life*.

- As an NUS student, do you feel like your schedules are all over the place and your contacts are too messy making searching through it a pain?
- Sounds like you need a "Bonus"!
- Our **BoNUS** is a desktop personal organizer application dedicated to NUS students to carry out various tasks such as storing contacts, scheduling for upcoming events, timetable planning as well as a calendar to better organise your campus life.
- Apart from storing essential information such as name, email, address and phone number, you have *infinite freedom* to customize your own fields!
- **BoNUS** is convenient, fast, secured, integrated, solely with the motivation of giving you a bonus for your student life in NUS. It is also oriented for typist users (most operations should be done using keyboard rather than mouse), which makes it all the more simple!

Let’s begin!

1.1. Purpose

This user guide will provide you with a comprehensive step-by-step guide on how to better utilise **BoNUS** to make your NUS life a breeze.

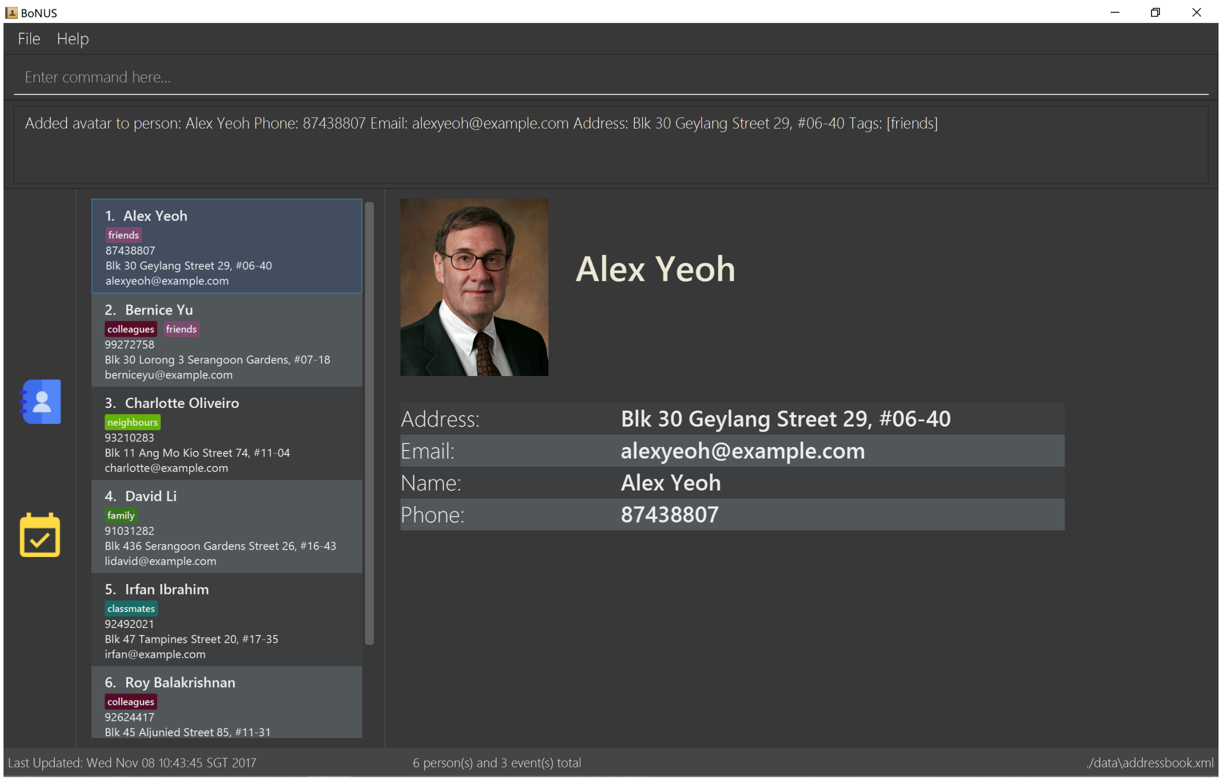
---

2. Quick Start

1. Start by ensuring you have Java version `1.8.0_60` or later installed in your Computer.

    Having any Java 8 version is not enough.
    If you only have earlier versions of Java 8, this app will not work.

2. Download the latest `bonus.jar` release from here.
3. Create a folder you want to use as home folder for**BoNUS** and copy the `jar` you have downloaded there.
4. Double-click the file to start the app. The GUI should appear in a few seconds.



Added avatar to person: Alex Yeoh Phone: 87438807 Email: alexyeoh@example.com Address: Blk 30 Geylang Street 29, #06-40 Tags: [friends]

1. Alex Yeoh
friends
87438807
Blk 30 Geylang Street 29, #06-40
alexyeoh@example.com

2. Bernice Yu
colleagues friends
99272758
Blk 30 Lorong 3 Serangoon Gardens, #07-18
berniceyu@example.com

3. Charlotte Oliveiro
neighbours
93210283
Blk 11 Ang Mo Kio Street 74, #11-04
charlotte@example.com

4. David Li
family
91031282
Blk 436 Serangoon Gardens Street 26, #16-43
lidavid@example.com

5. Irfan Ibrahim
classmates
92492021
Blk 47 Tampines Street 20, #17-35
irfan@example.com

6. Roy Balakrishnan
colleagues
92624417
Blk 45 Aljunied Street 85, #11-31

Alex Yeoh

Address:	Blk 30 Geylang Street 29, #06-40
Email:	alexyeoh@example.com
Name:	Alex Yeoh
Phone:	87438807

Last Updated: Wed Nov 08 10:43:45 SGT 2017 6 person(s) and 3 event(s) total ./data\addressbook.xml

Figure 2.1 : User Interface Demo

5. Type the command in the command box and press `Enter` to execute it.
   e.g. typing `help` and pressing `Enter` will open the help window.

6. Some example commands you can try:

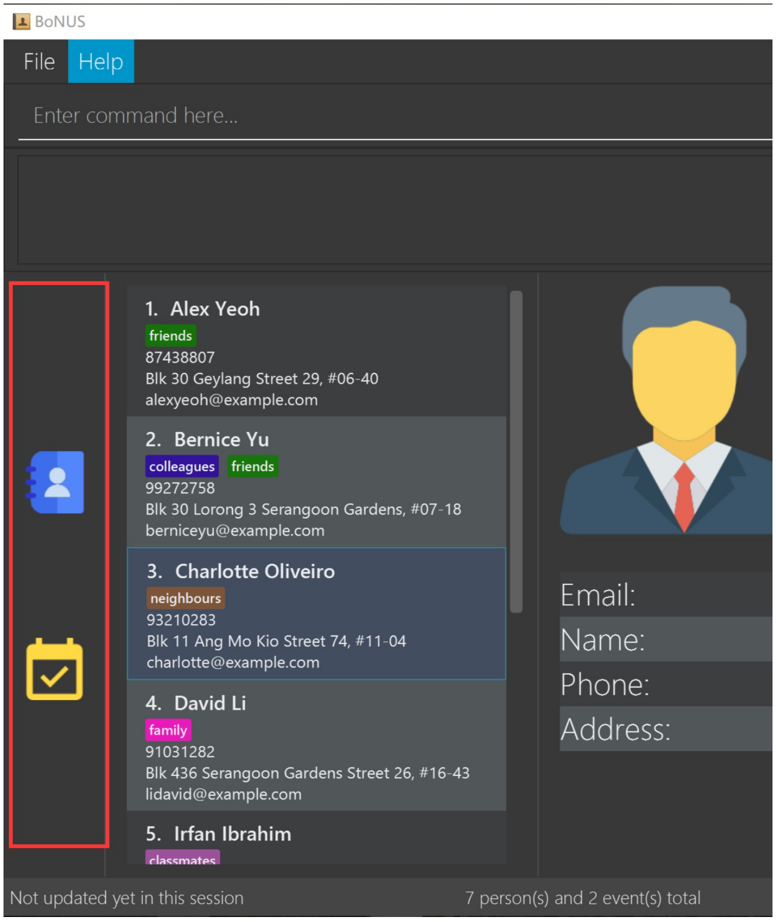
   - `list` : lists all contacts
   - `add n/John Doe p/98765432 e/johnd@example.com a/John street, block 123, #01-01` : adds a contact named `John Doe` to the Address Book.
   - `delete 3` : deletes the 3rd contact shown in the current list
   - `exit` : exits the app

7. Refer to the commands section below for details of each command.

---

3. Components

BoNUS consists of three main components: **contacts**, **events** and **calendar** which you can select using the sidebar on the left-hand side.



1. Alex Yeoh
friends
87438807
Blk 30 Geylang Street 29, #06-40
alexyeoh@example.com

2. Bernice Yu
colleagues friends
99272758
Blk 30 Lorong 3 Serangoon Gardens, #07-18
berniceyu@example.com

3. Charlotte Oliveiro
neighbours
93210283
Blk 11 Ang Mo Kio Street 74, #11-04
charlotte@example.com

4. David Li
family
91031282
Blk 436 Serangoon Gardens Street 26, #16-43
lidavid@example.com

5. Irfan Ibrahim
classmates

Not updated yet in this session
7 person(s) and 2 event(s) total

Figure 3.1 : Sidebar for Switching between Different Components

They are described in details as follows:

3.1. Contacts

Time to deal with the many contacts you have accumulated while studying in NUS.

- By default, you can store and update names, phone numbers, email and mailing addresses. (`add` and `edit` command)
- Additionally, you can customize an additional field to a contact that is not in the default. (`config add-property` command)
- Now, let's organize all your contacts by adding one or more tags to them. Those with the same tag will be classified under the same group. You can even customize the colour of the tags to your liking. (`config --set-tag-color` command)

3.2. Events

Got a meeting with professor or an assignment deadline you need to remember?

- By default, you can store and update the title, interval (starting time and end time), venue and description of the event. (`addE` and `editE` command)
- Similar to contacts, you can add customize properties and add tags to events as well.
- Setting a starting and/or end time for an event, will be reflected accordingly on thecalendar component.
- Reminders will automatically be added upon addition of event and start to notify you from 2 days before the actual event.
- If one or more of your contacts are attending the event with you, you can link them to the event too.

3.3. Calendar

(Current calendar is static, dynamic calendar coming in v2.0)

Calender provides a clearer image for you to view your upcoming events. All events are displayed based on their time interval for you to be able to attend to them one by one. However, events without time intervals will not be displayed here.

- You can choose the style to display the upcoming events (weekly/monthly/yearly view).
- You can add/update/delete events here, but the actual operation will be handled by theevents component.

---

4. Key features

4.1. Convenient

BoNUS is the ultimate application as it will provide you with the utmost convenience and gives you more time to focus on your more important tasks.

- Import contacts from *iCloud*, *Google+*, *Facebook*, etc.
- Import events from *Google Calendar*, *Outlook Calendar*, etc.
- Export data (contacts and events) to `.xml` file (default storage format for **BoNUS**), Excel Worksheet, etc.
- Sync between all your devices.
- (**Exclusive**) automatically generate your school schedule by simply entering your timetable URL from *NUSMods*.

4.2. Fast

As long as the number of records stored is less than 50,000 and the size of the storage file is smaller than 20MB, **BoNUS** will be able to

- Start the application in 5 seconds.
- Return the result of all commands available with *human-invisible* delay.
- Update things displayed on the GUI (graphic user interface) smoothly.

4.3. Secured

The **BoNUS** developers understand that you will be storing personal data in the application. The following are features of our application that ensures you will have privacy and security.

- All data saved to the storage file will be encrypted using the state-of-the-art encryption scheme (AES-256).
- You will be prompted to enter their password whenever they open the application (if you has decided to lock your application the last time before you exited).
- You can set up 2FA (two-factor authentication) to fulfill extra security requirement(s).

5. Commands

The following is a list of commands that you can use in the application.

Command Format

- Words in parentheses represent the command shortcut e.g. in `(a)add n/NAME`, `a` is the shorthand-equivalent notation for the `add` command.
- Words in `UPPER_CASE` is information to be supplied by you e.g. in `add n/NAME`, `NAME` is to be filled in by you. e.g `add n/John Doe`.
- Words in square brackets are optional e.g `n/NAME [t/TAG]`, `t/TAG` is optional. e.g `n/John Doe t/friend` or as `n/John Doe`.
- Items with `…` after them can be used from zero times to as many times as you like e.g. `[t/TAG]…` can be used as ` ` (i.e. 0 times), `t/friend`, `t/friend t/family` etc.
- You can type the information in any order e.g. if the command specifies `n/NAME p/PHONE_NUMBER`, `p/PHONE_NUMBER n/NAME` works too.
- If parameters with the same prefix are typed multiple times, only the last one will be taken. For example, `n/John Doe n/Martin Henz`, then only `Martin Henz` will be taken into actual consideration.

Command in different components

- Most commands can be applied to either a person or an event, whose result will depend on the context e.g. `add` will add a person if the user is currently in the contacts component, otherwise `addE` will add a new event.
- All commands entered in the calendar component will actually be handled by either contacts component or events component.

5.1. Viewing help : `(h)help`

Have a query? Use the `help` command!
Format: `(h)help` (or press `F1` on the keyboard)

5.2. Adding a person : `(a)add`

You just met someone and want to add their contact information into your contact list.
Format: `(a)add n/NAME p/PHONE_NUMBER e/EMAIL a/ADDRESS [t/TAG]…`

This adds the person to your contact list.

Your contact can have any number of tags (including 0).

Examples:

- `add n/John Doe p/98765432 e/johnd@example.com a/John street, block 123, #01-01`
- `add n/Betsy Crowe t/friend e/betsycrowe@example.com a/Newgate Prison p/1234567 t/criminal`

In a similar format, you can also add customize properties when adding a new person, just use `config --add-property` command before to define that property.
Notice all pre-defined properties (name, phone, email, address) are compulsory when adding a new person, while customize properties are optional.

Example:

If you want to add birthday as a field for a contact and use `b` as the short name,

- `config --add-property s/b f/birthday`

Then, you can set a contact's `birthday` when you add a new contact (*see* `b/12091191` *below*)

- `add n/Chris Lee p/98765432 e/johnd@example.com a/23 Chinatown b/12091991 t/friends`

5.3. Adding an event : `(aE)addE`

You want to record your friend's birthday party coming up.
Format: `(aE)addE n/NAME dt/DATE_TIME a/EMAIL a/ADDRESS`

This will add a new event to your list of events.

Examples:

- `addE n/Does Birthday dt/25122015 20:30 a/12 Kent Ridge Drive`
- `addE n/Betsy Birthday dt/25122016 21:30 a/23 Marina Road`

- The standard format for time should be `DDMMYYYY HH:MM` in 24-hour format.
- However, the application may sometimes be *smart* enough to interpret what you typed. For example, if you type `this afternoon` or `tomorrow evening`, it will be automatically converted to the standard format.
- Do NOT try to *challenge* the application, it is not promised to produce an expected result if you use non-standard format or if the expression is not simple or clear enough.
- Avoid putting the year first when using the non-standard format.

- Reminders will automatically be added and start to notify you from 2 days before the actual event.
- Reminders in the form of a bell image will be displayed on the event card itself.
- Red Bell : Day of event
  Orange Bell: One day before event
  Green Bell: Two days before event

5.4. Listing all persons : `(l)list`

You want to view all your contacts in your contact list.
Format: `(l)list`

This will list out all of your contacts in alphabetical order.

Example:

- `list`

> Contacts will be sorted by their names (increment).

5.5. Listing all events : `(lE)listE`

Shows a list of all events in the application.
Format: `(lE)listE`

This will list out all of your events by date/time.

Example:

- `listE`

5.6. Editing a person : `(e)edit`

(Editing of customize properties coming in v2.0)

Edits an existing person in the application.
Format: `(e)edit INDEX [n/NAME] [p/PHONE] [e/EMAIL] [a/ADDRESS] [t/TAG]…`

- Edits the person at the specified `INDEX`. The index refers to the index number shown in the last listing. The index **must be a positive integer** like 1, 2, 3, …
- At least one of the optional fields must be provided.
- Existing values of the selected data item will be updated to the input values.
- When editing tags, the existing tags of that data item will be removed i.e adding of tags is not cumulative.
- You can remove all existing tags by typing `t/` without specifying any tags after it.

Examples:

- `edit 1 p/91234567 e/johndoe@example.com`
  Edits the phone number and email address of the 1st person to be `91234567` and `johndoe@example.com` respectively.
- `edit 2 n/Betsy Crower t/`
  Edits the name of the 2nd person to be `Betsy Crower` and clears all existing tags.

5.7. Editing an event : `(eE)editE`

Edits an existing event in the application.
Format: `(eE)editE INDEX [n/NAME] [dt/DATE_TIME] [a/ADDRESS]`

- Edits the event at the specified `INDEX`. The index refers to the index number shown in the last listing. The index **must be a positive integer** like 1, 2, 3, …
- At least one of the optional fields must be provided.
- Existing values of the selected data item will be updated to the input values.

Examples:

- `editE 1 dt/18052013 03:30`
  Edits the date/time the 1st event to be `18052013 03:30`.
- `editE 2 n/Lunch with Betsy`
  Edits the name of the 2nd event to be `Lunch with Betsy`.

5.8. Locating persons/events by any property : `(f)find`

(`AND` and `OR` search coming in v2.0)

Finds persons/events whose corresponding field(s) contain any of the given keywords.
Format: `(f)find KEYWORD [MORE_KEYWORDS] [p/KEYWORD [MORE_KEYWORDS]]…`

- The search is case insensitive. e.g `hans` will match `Hans`
- The search can be based on one or multiple properties. The short name of the property `n/` can be omitted if the searching criteria is for the `name` property.
- Apply `OR` search for keywords of the same property, i.e. persons/events matching at least one keyword will be returned. e.g. `Hans Bo` will return `Hans Gruber`, `Bo Yang`. Thus, the order of the keywords for the same property does not matter. e.g. `Hans Bo` will match `Bo Hans`.
- Apply `AND` search for keywords of the different property, i.e. only persons/events matching all of the required properties will be returned. e.g. `Hans Bo p/84651943` will only return persons whose name contains either `Hans` or `Bo`, as well as, whose phone number is the same as `84651943`.
- Only full word matching will be returned. e.g. `Han` will not match `Hans`

Examples:

- `find John`
  Returns `john` and `John Doe`
- `find Betsy Tim John`
  Returns any person having names `Betsy`, `Tim`, or `John`

5.9. Locating persons by their tag : `(ft)findtag`

Finds persons whose corresponding tags contain any of the given keywords and lists them out.
Format: `(ft)findtag KEYWORD [MORE_KEYWORDS]…`

- The search is not case insensitive. e.g `friends` will match `Friends`
- Only full word matching will be returned. e.g. `friend` will not match `friends`

Examples:

- `findtag family`
  Returns any person that contains a `family` tag
- `findtag family colleagues`
  Returns any person that contains the tags `family` and `colleagues`

5.10. Deleting a person : `(d)delete`

Deletes the specified person from the application.
Format: `(d)delete INDEX`

- Deletes the person at the specified `INDEX`.
- The index refers to the index number shown in the most recent listing.
- The index **must be a positive integer** like 1, 2, 3, …

Examples:

- `list`
  `delete 2`
  Deletes the 2nd person in the address book.
- `find Betsy`
  `delete 1`
  Deletes the 1st person in the results of the `find` command.

5.11. Deleting an event : `(dE)deleteE`

Deletes the specified event from the application.
Format: `(dE)deleteE INDEX`

- Deletes the event at the specified `INDEX`.
- The index refers to the index number shown in the most recent listing.
- The index **must be a positive integer** like 1, 2, 3, …

Examples:

- `list`
  `deleteE 2`
  Deletes the 2nd event in the address book.

5.12. Selecting a person: `(s)select`

Selects a person (identified by the index number used in the last listing) to view the details of that person.
Format: `(s)select INDEX`

- Selects the person/event and loads the details of this data item.
- The index refers to the index number shown in the most recent listing.
- The index **must be a positive integer** like `1, 2, 3, …`
- All properties of this person/event will be displayed at the right side of the interface (ordered by the name of the property).

Examples:

- `list`

`select 2`
Selects the 2nd person in the address book.

- `find Betsy`
  `select 1`
  Selects the 1st person in the results of the `find` command.

5.13. Emailing a person : `(em)email`

Emails a person (identified by the index number used in the last listing) to email that person.
Format: `(em)email INDEX`

> - Emails the person by opening your default mail app with the "To: " field filled up with the person’s email.
> - The index refers to the index number shown in the most recent listing.
> - The index **must be a positive integer** like `1, 2, 3, …`

Examples:

- `list`
  `email 2`
  Emails the 2nd person in the address book.

5.14. Directions to a person’s address : `(gm)gmap`

Opens Google Maps in default browser with a person’s (identified by the index number used in the last listing) address in the Destination.
Format: `(gm)gmap INDEX`

> - The index refers to the index number shown in the most recent listing.
> - The index **must be a positive integer** like `1, 2, 3, …`

Examples:

- `list`
  `gmap 2`
  The address of the 2nd person in the address book will be in the Destination field in Google Maps.

5.15. Finding a contact on Facebook : `(fb)facebook`

Searches for a person’s (identified by the index number used in the last listing) name on Facebook.
Format: `(fb)facebook INDEX`

> - The index refers to the index number shown in the most recent listing.
> - The index **must be a positive integer** like `1, 2, 3, …`

Examples:

- `list`
  `facebook 2`

The name of the 2nd person in the address book will be searched on Facebook.

5.16. Adding avatar to a person : `(avr)avatar`

(Better checking for file type coming in v2.0)

Adds avatar to a person (identified by the index number used in the last listing).
Format: `(avr)avatar INDEX IMAGE_PATH`

- The given path must be pointed to a valid image in `png`, `jp(e)g` and `svg` format.
- The index refers to the index number shown in the most recent listing.
- The index **must be a positive integer** like `1, 2, 3, …`
- The path should not be empty, nor should it contain any prohibited characters `?!%*+:|"<>`.
- The application will infer the path as from a location relative to the **BoNUS** home folder if the given path is not complete (or an absolute path).

Examples:

- `list`
  `avatar 2 linda.jpg`
  Adds avatar to the 2nd person in the address book.

- Do not move, rename or delete the image you use as an avatar; otherwise, it will not be shown in **BoNUS**.
- You are suggested to move the image into the home folder where you use**BoNUS**.
- Do not be *nasty* about the input file. If you provide an invalid file that looks like an image but is not an actual image, the area reserved for avatar will simply become transparent.

5.17. Switching themes : `(t)theme`

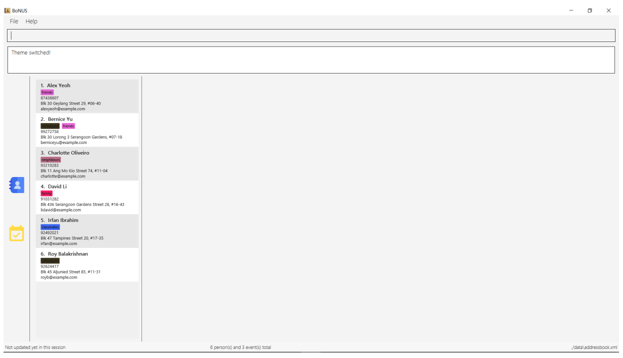
Switches between Dark Theme and Bright Theme.
Format: `(t)theme`

Examples:

- `theme`

Theme will be saved upon exit. When**BoNUS** is opened again, theme will be initialized to the theme that was saved previously.

As shown in Figure 5.15.1 and Figure 5.15.2, Figures depict the Bright Theme and Dark Theme respectively.



Theme switched!

1. Alex Yeoh
friends
87438807
Blk 30 Geylang Street 29, #06-40
alexyeoh@example.com

2. Bernice Yu
[illegible] friends
99272758
Blk 30 Lorong 3 Serangoon Gardens, #07-18
berniceyu@example.com

3. Charlotte Oliveiro
neighbours
93210283
Blk 11 Ang Mo Kio Street 74, #11-04
charlotte@example.com

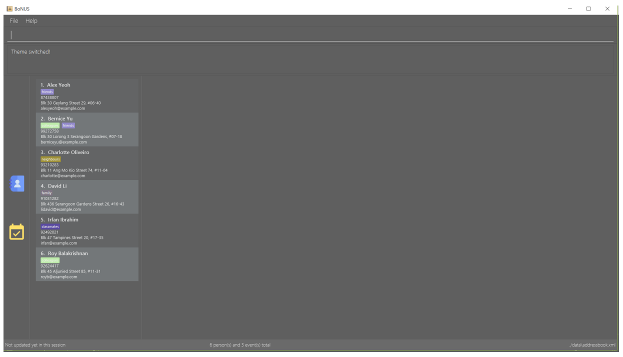
4. David Li
family
91031282
Blk 436 Serangoon Gardens Street 26, #16-43
lidavid@example.com

5. Irfan Ibrahim
classmates
92492021
Blk 47 Tampines Street 20, #17-35
irfan@example.com

6. Roy Balakrishnan
[illegible]
92624417
Blk 45 Aljunied Street 85, #11-31
royb@example.com

Not updated yet in this session
6 person(s) and 3 event(s) total
./data\addressbook.xml

Figure 5.15.1 : Bright Theme



Theme switched!

1. Alex Yeoh
friends
87438807
Blk 30 Geylang Street 29, #06-40
alexyeoh@example.com

2. Bernice Yu
colleagues friends
99272758
Blk 30 Lorong 3 Serangoon Gardens, #07-18
berniceyu@example.com

3. Charlotte Oliveiro
neighbours
93210283
Blk 11 Ang Mo Kio Street 74, #11-04
charlotte@example.com

4. David Li
family
91031282
Blk 436 Serangoon Gardens Street 26, #16-43
lidavid@example.com

5. Irfan Ibrahim
classmates
92492021
Blk 47 Tampines Street 20, #17-35
irfan@example.com

6. Roy Balakrishnan
colleagues
92624417
Blk 45 Aljunied Street 85, #11-31
royb@example.com

Not updated yet in this session
6 person(s) and 3 event(s) total
./data\addressbook.xml

Figure 5.15.2 : Dark Theme

5.18. Listing entered commands : `(his)history`

Lists all the commands that you have entered in reverse chronological order.
Format: `(his)history`

Example:

- `history`

> Pressing the `↑` and `↓` arrows will display the previous and next input respectively in the command box.

5.19. Undoing previous command : `(u)undo`

Restores the application to the state before the previous *undoable* command was executed.
Format: `(u)undo`

> Undoable commands: those commands that modify the application’s content (`add`, `addE`, `edit`, `editE`, `delete`, `deleteE`, and `clear`).

Examples:

- `delete 1`
  `list`
  `undo` (reverses the `delete 1` command)
- `select 1`
  `list`
  `undo`
  The `undo` command fails as there are no undoable commands executed previously.

- `delete 1`
  `clear`
  `undo` (reverses the `clear` command)
  `undo` (reverses the `delete 1` command)

> You can view what command you have undone from the user feedback message.

5.20. Redoing the previously undone command : `(r)redo`

Reverses the most recent `undo` command.
Format: `(r)redo`

Examples:

- `delete 1`
  `undo` (reverses the `delete 1` command)
  `redo` (reapplies the `delete 1` command)
- `delete 1`
  `redo`
  The `redo` command fails as there are no `undo` commands executed previously.
- `delete 1`
  `clear`
  `undo` (reverses the `clear` command)
  `undo` (reverses the `delete 1` command)
  `redo` (reapplies the `delete 1` command)
  `redo` (reapplies the `clear` command)

> You can view what command you have redone from the user feedback message.

5.21. Clearing all entries : `(c)clear`

Clears all entries from all components (both contacts and events).
Format: `(c)clear`

Example:

- `clear`

5.22. Importing data : `(i)import`

5.22.1. From `.xml` format

Imports the data in an external XML file, including data from all three components:**Contacts**, **Events** and **Calendar**, into the current address book of **BoNUS**.
Format: `(i)import FILEPATH`

> The default data format is `.xml` file. Without explicit parameters, the application will treat the given path as a file in `.xml` format. It is unnecessary to explicitly state `import --xml FILEPATH`, although you are allowed to do so.

- Imports data from the location and file name specified by `FILEPATH`.
- `FILEPATH` must end with an extension of `.xml`.
- The file name in `FILEPATH` should not be empty, nor should it contain any prohibited characters `?!%*+:|"<>`.
- If a relative path is provided, the data will be imported from a location relative to the **BoNUS** installation directory.
- Persons and events that exist in **BoNUS** and the specified file will not be imported.
- Data in the specified XML file must be in the format as recognized by**BoNUS**.

Examples:

- For `Windows` users:
  `import C:\Users\John Doe\Documents\bonus.xml`
- For `macOS` and `Linux` users:
  `import /Users/John Doe/Documents/bonus.xml`

For `Windows` users, use `\` as the name-separator in your `FILEPATH`.
For `macOS` and `Linux` users, use `/` instead.

5.22.2. From `.bo` format

(Coming in v2.0)

Imports the data in an external BoNUS script file (which ends with `.bo`), including data from all three components: **Contacts**, **Events** and **Calendar**, into the application.
Format: `(i)import --script FILEPATH`

- You must explicitly provide the `--script` parameter.
- `FILEPATH` must end with an extension of `.bo`.
- The file name in `FILEPATH` should not be empty, nor should it contain any prohibited characters `?!%*+:|"<>`.
- If a relative path is provided, the data will be imported from a location relative to the **BoNUS** installation directory.
- The provided script file should include one or multiple lines of valid **BoNUS** command(s). Each line can only have **at most one** command.
- The **command** here refers to any command mentioned in this guide.

Examples:

- For `Windows` users:
  `import --script C:\Users\John Doe\Documents\bonus.bo`
- For `macOS` and `Linux` users:
  `import --script /Users/John Doe/Documents/bonus.bo`

For `Windows` users, use `\` as the name-separator in your `FILEPATH`.
For `macOS` and `Linux` users, use `/` instead.

5.22.3. From NUSMods URL

(Exclusive feature for NUS students)

The **BoNUS** team understands that NUSMods has become an indispensable school timetable builder for almost all students at NUS. Thus, you are definitely allowed to import your timetable from NUSMods to the **BoNUS** application.
Format: `(i)import --nusmods YOUR_NUSMODS_URL`

- The URL provided must be complete and should begin with `http(s)://nusmods.com/timetable/` .
- Directly copy from the address bar of your browser. Do **NOT** use the short URL generated by the *Sharing Timetable* feature provided by NUSMods.
- Final examinations for all modules in your NUSMods timetable will be automatically added as events to the application.

Example:

- `import --nusmods https://nusmods.com/timetable/2017-2018/sem1?CS2103T[TUT]=C01`

- Make sure you have stable Internet connection when using this command.
- You may need to wait for a while as the application is retrieving information from NUSMods.

5.22.4. From Google Accounts

(Coming in v2.0)

Imports your contacts and events from Google Contacts and Google Calender respectively.
Format: `(i)import --google https://myaccount.google.com/`

- Upon execution of the command, you will be brought to Google’s login page for authentication.
- Upon successful authentication, data from Google contacts and calendar will automatically be imported into **BoNUS**'s contact list and event list.

Example: `import --google https://myaccount.google.com/`

- URL as shown in example is the default URL that will be opened for you
- You would have to enter your login credentials on Google’s sign in page before data from Google Contacts and Google Calendar gets imported.
- In the case of you not entering proper login credentials after three tries,**BoNUS** will cancel the request for importing from Google.

5.23. Exporting data : `(p)export`

5.23.1. To `.xml` format

Exports the current data in the application, including data from all three components:**Contacts**, **Events** and **Calendar**, to an external location.

Format: `(p)export FILEPATH`

- Exports data to the location and file name specified by `FILEPATH`.
- `FILEPATH` must end with an extension of `.xml`.
- The file name in `FILEPATH` should not be empty.
- The file name and any non-existent folder names in `FILEPATH` should not contain any prohibited characters `?!%*+:|"<>`.
- If a relative path is provided, the data will be exported to a location relative to the **BoNUS** installation directory.
- Existing data file at `FILEPATH` will be overwritten.
- Parent directories, if specified in `FILEPATH`, will be created if they do not exist.

Examples:

- For `Windows` users:
  `export C:\Users\John Doe\Documents\bonus.xml`
- For `macOS` and `Linux` users:
  `export /Users/John Doe/Documents/bonus.xml`

> For `Windows` users, use `\` as the name-separator in your `FILEPATH`.
> For `macOS` and `Linux` users, use `/` instead.

5.23.2. To Microsoft Excel^{TM} Worksheet

(Coming in v2.0)

Exports the current data in the application to an external file of Microsoft Excel^{TM} format.
Format: `(p)export --excel FILEPATH`

- The file name should follow similar rules to the section above.
- However, it must end with an extension of `.xls` (`.xlsx` is currently not supported).

Examples:

- For `Windows` users:
  `export --excel C:\Users\John Doe\Documents\bonus.xls`
- For `macOS` and `Linux` users:
  `export --excel /Users/John Doe/Documents/bonus.xls`

> For `Windows` users, use `\` as the name-separator in your `FILEPATH`.
> For `macOS` and `Linux` users, use `/` instead.

5.24. Advance setting : `(cfg)config`

Changes the configuration of the application or applies some advance settings to the data. Make sure you know what you are doing before using any of the following commands. These commands are intended for advance users.

5.24.1. Adding a customize property : `(cfg)config --add-property`

Adds a new customize property field available to all persons or events.
Format: `(cfg)config --add-property s/SHORT_NAME f/FULL_NAME [m/MESSAGE r/REGULAR_EXPRESSION]`

- This command does not add a new property to a specific person. Instead, it defines a property that will be available to all persons/events.
- The short name `s/` and full name `f/` of the new property are compulsory, while the constraint message `m/` and regular expression for validation `r/` are optional. However, `m/` and `r/` must come together, i.e. a regular expression must be accompanied by a constraint message, which will be shown when the validation fails.
- If constraint message and regular expression are not specified, the default would be `[^\s].*`, which mean the value of this property cannot be blank.
- Short name is the identity (primary key) of all properties. Thus, the short name must be unique. The command will fail if there is an existing property with the same short name.
- The given regular expression must use legal syntax. It will be checked by the Pattern.compile method.

Example:

- `config --add-property s/ag f/age`
- `config --add-property s/b f/birthday m/Birthday should be in the format of DD/MM/YYYY r/[^\s].*`

- Short name is used as the prefix for `add` / `addE` and `edit` / `editE` commands.
- Full name is used to display the name of each property on the right panel (to show details of the selected person, use `select` command).
- Constraint message is the string that will be shown in result display box when the input value for this property is invalid.
- Regular expression is used to check whether the input value for this property is valid.

5.24.2. Setting the tag color : `(cfg)config --set-tag-color`

(Tag feature for events coming in v2.0)

Sets the displayed color of a certain tag (for persons/events).
Format: `(cfg)config --set-tag-color TAG_NAME COLOR`

- By default, the application will use a random color to display each tag. The same tags are displayed using the same color, different tags are *usually* displayed using different colors.
- The value of the parameter `COLOR` should be either one of the 140 pre-defined color names or a valid RGB value (a 3-bit or 6-bit hexadecimal number starting with a `#`).
- You can pick the RGB value of your favorite color from here.
- All legal pre-defined color names can be found from here.

Example:

- `config --set-tag-color friends red`

If you enter an invalid color name, the background for that tag will be set to transparent temporarily. You can use this again to set it to a legal color.

5.25. Booking of NUS facilities: `(book)bookNUS`

(Exclusive feature for NUS students)
(Coming in v2.0)

The **BoNUS** team understands that booking of facilities in NUS may be troublesome. Therefore, we have decided to integrate the facility booking website with **BoNUS** such that upon successful booking of a specific facility, your Events List will be updated accordingly with the Date, Time and Venue of booking.

Format: `(book)bookNUS`

- Upon execution the command, you will be brought the the NUS facility booking website on a separate browser.
- Upon successful booking of the facility, Event with date, time and venue will automatically be added as events to the application.

Example:

- `bookNUS`

- Make sure you have stable Internet connection when using this command.
- There may be some waiting time involved due to the retrieval of information from the website to update **BoNUS**.

5.26. Exiting the program : `(x)exit`

Exits the program.
Format: `(x)exit`

Example:

- `exit`

5.27. Saving the data

- Address book data are saved in the hard disk automatically after any command that changes the data.
- These commands are also called undoable commands.
- There is no need to save manually.

6. 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 Address Book folder.

7. Command Summary

- **Add** : `(a)add n/NAME p/PHONE_NUMBER e/EMAIL a/ADDRESS [t/TAG]…`
  e.g. `add n/James Ho p/22224444 e/jamesho@example.com a/123, Clementi Rd, 1234665 t/friend t/colleague`
- **Add event** : `(aE)addE n/NAME dt/DATE_TIME a/ADDRESS`
  e.g. `addE n/James birthday dt/18022017 13:30 a/123, Clementi Rd, 1234665`
- **Clear** : `(c)clear`
- **Delete** : `(d)delete INDEX`
  e.g. `delete 3`
- **Delete event** : `(dE)deleteE INDEX`
  e.g. `deleteE 3`
- **Edit** : `(e)edit INDEX [n/NAME] [p/PHONE_NUMBER] [e/EMAIL] [a/ADDRESS] [t/TAG]…`
  e.g. `edit 2 n/James Lee e/jameslee@example.com`
- **Edit event** : `(eE)editE INDEX [n/NAME] [dt/DATE_TIME] [a/ADDRESS]`
  e.g. `editE 2 n/Lees Day`
- **Find** : `(f)find KEYWORD [MORE_KEYWORDS]`
  e.g. `find James Jake`
- **List** : `(l)list`
- **List event** : `(lE)listE`
- **Switch Themes** : `(t)theme`
- **Help** : `(h)help`
- **Select** : `(s)select INDEX`
  e.g. `select 2`
- **History** : `(i)history`
- **Undo** : `(u)undo`
- **Redo** : `(r)redo`
- **Exit** : `(x)exit`
- **Import** :
  (i) From `.xml` file: `import FILEPATH`
  eg. For `Windows` users: `import C:\Users\John Doe\Documents\bonus.xml`
  eg. For `macOS` and `Linux` users: `import /Users/John Doe/Documents/bonus.xml`
  (ii) From script file: `import --script FILEPATH`
  eg. For `Windows` users: `import C:\Users\John Doe\Documents\bonus.bo`
  eg. For `macOS` and `Linux` users: `import /Users/John Doe/Documents/bonus.bo`
  (iii) From NUSMods timetable: `import --nusmods URL`
  eg. `import --nusmods https://nusmods.com/timetable/2017-2018/sem1?CS2103T[TUT]=C01`
- **Export** : `(p)export [--excel] FILEPATH`
  eg. For `Windows` users: `export C:\Users\John Doe\Documents\bonus.xml`
  eg. For `macOS` and `Linux` users: `export /Users/John Doe/Documents/bonus.xml`
- **Config** :
  (i) Add customize property `(cfg)config --add-property`

eg. `config --add-property s/b f/birthday`
(ii) Change tag color `(cfg)config --set-tag-color`
eg. `config --set-tag-color friends blue`

Last updated 2017-11-13 13:34:34 +00:00