

JavaCake - User Guide

1. Introduction	3
2. Getting Started	4
3. Features	5
3.1. Getting the available commands: help	6
3.2. Showing the table of contents: list and overview	8
3.3. Navigating the table of contents: goto	10
3.4. Navigating back to the previous view: back	11
3.5. Doing a quiz	12
3.6. Reviewing your answers in a quiz: review	14
3.7. Adding a deadline: deadline	15
3.8. Meeting a deadline: done	17
3.9. Deleting a deadline: delete	18
3.10. Postponing a deadline: snooze	19
3.11. Viewing current progress in the app: score	20
3.12. Creating personal notes in the app: createnote	21
3.13. Editing personal notes in the app: editnote	24
3.14. Viewing personal notes in the app: viewnote	26
3.15. Deleting personal notes in the app: deletenote	27
3.16. Listing personal notes in the app: listnote	28
3.17. Exiting the program: exit	29
3.18. Saving data:	30

3.19. Helper for commands:	30
4. FAQ	31
5. Do we really need Common Command Examples	31

1. Introduction

JavaCake is a desktop-based application designed for inexperienced programmers who are either interested in learning the basics of the Java programming language and also for NUS students without Java experience who are going to take CS2113T.

The application breaks down the wordy online documentation into smaller snippets of information to make learning the language less intimidating. You can easily and quickly jump from topic to topic by typing inside the in-built command line interface.

On top of that, quizzes of various difficulties can be found at the end of each topic to test your understanding. Not only that, JavaCake keeps track of your quiz attempts and records your progression.

Note the following symbols to be used:



This symbol indicates important information.



This symbol indicates a tip that the user can follow in order to achieve the best result from the given information.

`list`

A grey highlight (called a mark-up) indicates that this is a command that can be inputted into the command line and executed by the application.

`Logic`

Blue text with grey highlight indicates a component, class or object in the architecture of the application.

currentFilePath

Italicised Consolas font will be used to denote variable names.

2. Getting Started

1. Install JDK 11.
2. Download the latest .jar and put it in a directory of your choice



It is recommended to put the .jar file into a new folder of your choice.

This is preferred since there are folders auto-generated by the application, which stores the user's data and progress.

3. Double click the .jar file to run. The GUI should appear in a few seconds.

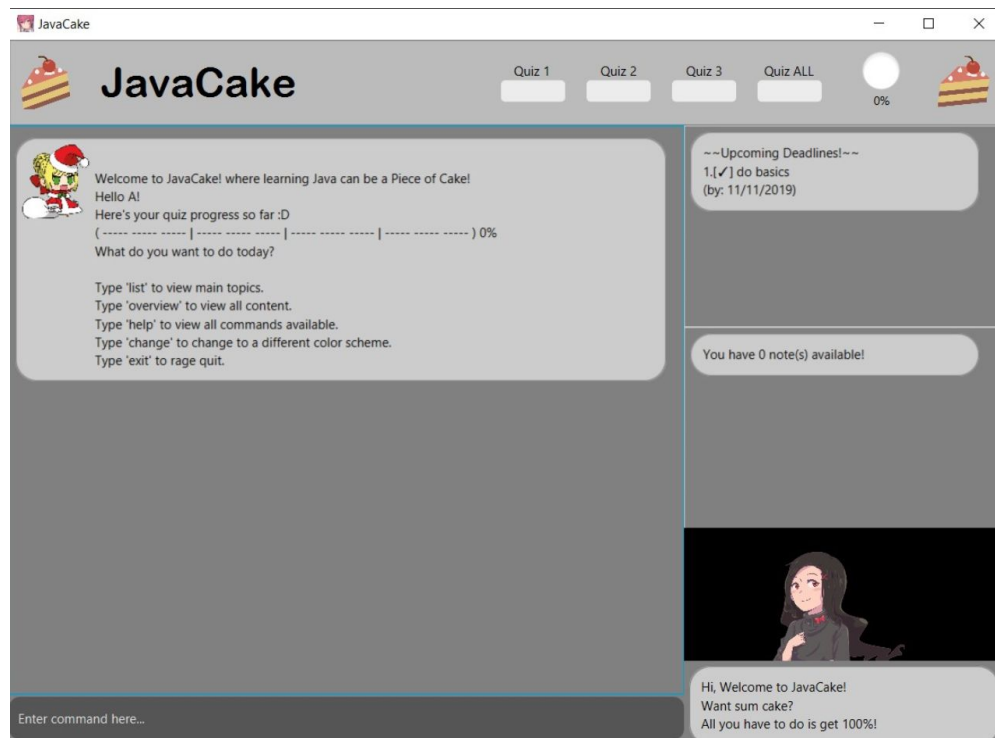


Figure 1: Graphics User Interface (GUI) for JavaCake

4. Type the command in the command box and press Enter on the keyboard to execute it.
5. Refer to Section 3, “Features” for details of each command.

3. Features

This section covers the list of commands and their syntax for *JavaCake*.



Inputs are formatted according to a *command* and an *argument*.

An example would be:

```
goto 1.3
```

In this case, the command is `goto` and the argument is `1.3`.

When the user first launches the application, they will be greeted by the following message and prompted to provide a username as shown in the figure below.

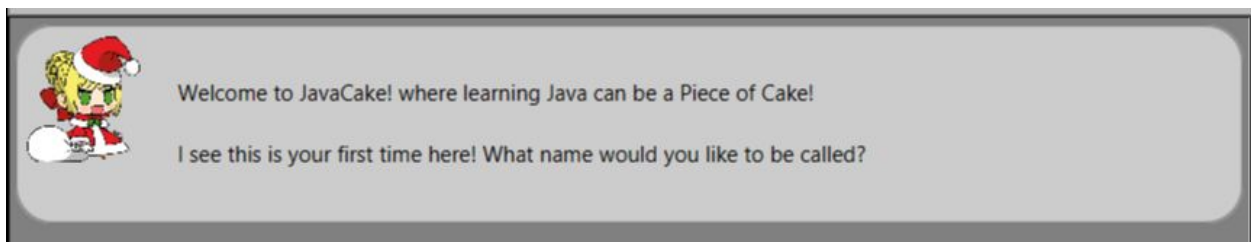


Figure 2: Message shown to a new user

Upon typing in any username, the user will then be presented with a few common commands that they can input as shown below.



This welcome message will not appear for testers.

Testers will have a preloaded save file installed upon running their .jar file, and will be shown the following follow-up message instead.

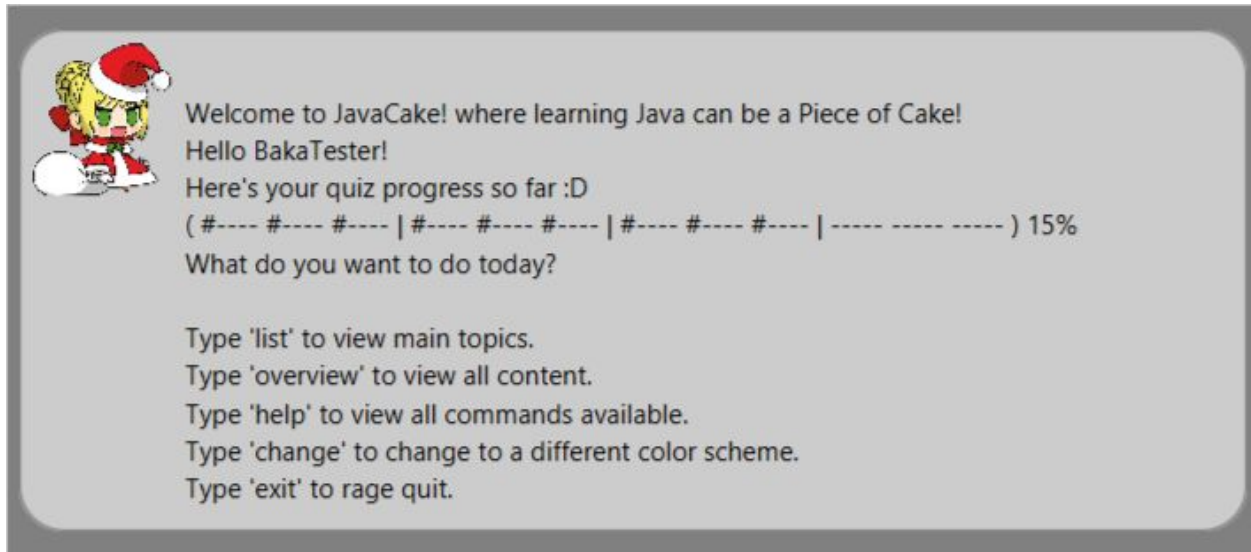


Figure 3: Message shown to tester

3.1. Getting the available commands: `help`

To request for `help`:

1. Type `help` into the command box and press Enter as shown in Figure 4.



Figure 4: Input 'help' in the command box

2. The list of available built-in commands will be displayed for you in the dialogue box in *JavaCake* as shown in *Figure 5*.

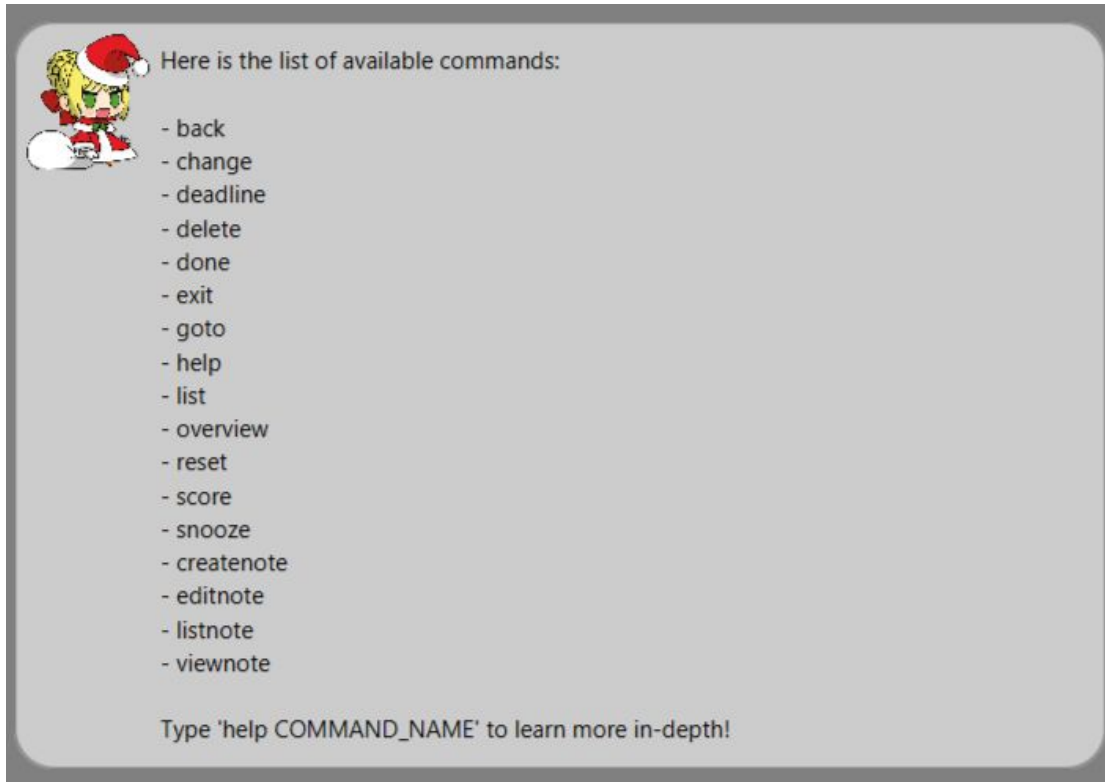


Figure 5: List of built-in commands available for users

3. If you wish to view a detailed explanation for a specific command, you can type `help 'COMMAND'` into the command box and press Enter on the keyboard.
4. E.g. To view a detailed explanation on 'overview' command, you can type `help overview` as shown in the figures below. The dialog box will output an explanation of what the inputted command does.



Figure 6: Input 'help overview' in the command box

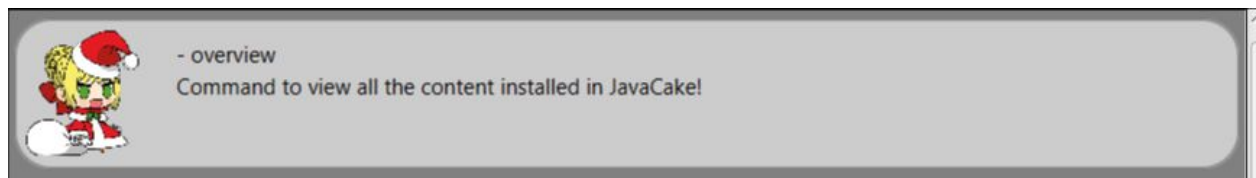


Figure 7: Explanation on overview command

3.2. Showing the table of contents: **list** and **overview**

If you wish to view the main list of content for *JavaCake*, simply use **list**.

If you wish to view the entire list of content for *JavaCake*, simply use **overview**.

To **list**:

1. Type **list** into the command box and press Enter.



Figure 8: Input 'list' in the command box

2. The main list of content will be displayed for you in the dialogue box in *JavaCake* as shown below.

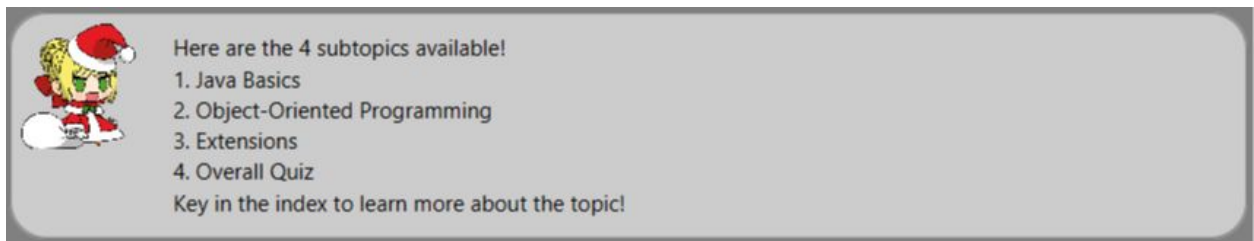


Figure 9: Main list of content when 'list' is used

To overview:

1. Type `overview` into the command box and press Enter.



Figure 10: Input 'overview' in the command box

2. The entire list of content will be displayed for you in the dialogue box in *JavaCake* as shown below.

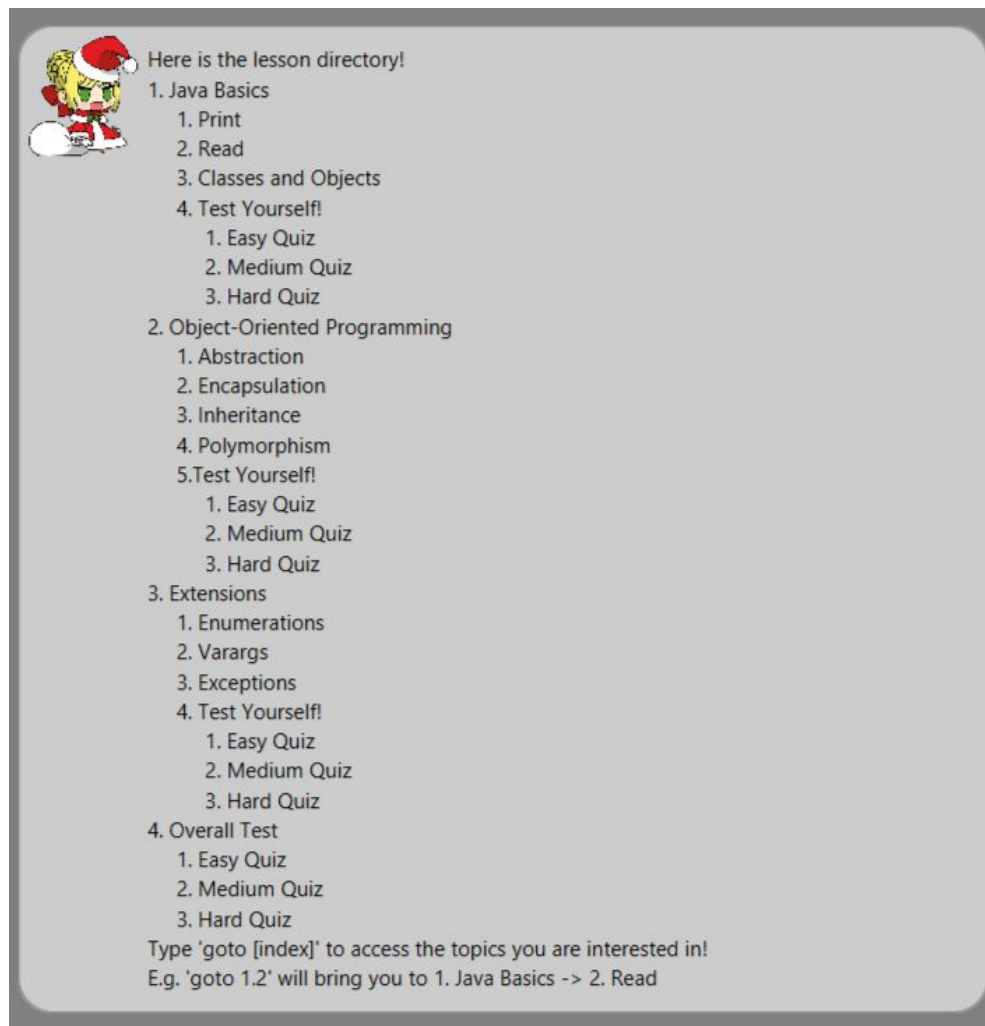


Figure 11: Whole list of content when overview' is used

3.3. Navigating the table of contents: goto

If you wish to visit a particular topic, you can simply access it by using `goto 'index'`, where the `'index'` corresponds to the topic number shown on-screen currently. However, if you know the index of the file in the unopened directory, you can also append it into the index. For example, instead of typing `goto 1` and then `goto 2`, you can simply type `goto 1.2`.

To `goto 'index'`:

1. Type `goto` followed by the related index into the command box and press Enter.



Figure 12: Input goto 'index' where 'index' = 1 in the command box

2. If the topic that you choose contains subtopics, the new list of subtopics will be displayed for you in the dialogue box in *JavaCake* as shown below.



Figure 13: New list of subtopics

3. If the topic that you choose contains content, the content will be displayed for you in the dialogue box in *JavaCake* as shown below.

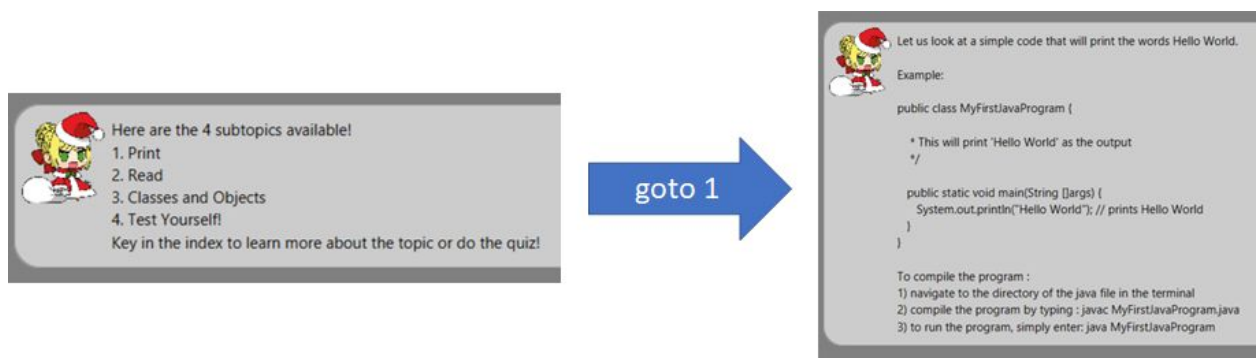


Figure 14: Display of content



Only valid indexes are allowed.

If you see a message containing the words 'out of bounds', simply press **list** to view the current directory and re-type the command with the correct index.

3.4. Navigating back to the previous view: **back**

If you wish to navigate back to the previous view, simply use **back**.

To **back**:

1. Type **back** into the command box and press Enter.



Figure 15: Input back in the command box

2. If the view that you are currently at is a sublist, the main list of topics will be displayed for you in the dialogue box in *JavaCake* as shown below.

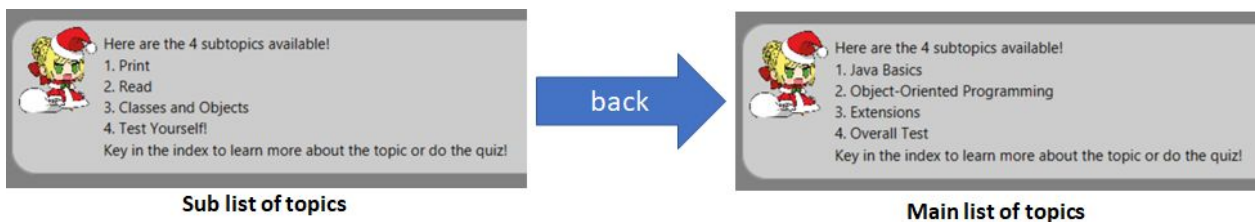


Figure 16: Sub list of topics back to Main list of topics

3. If the view that you are currently at is a content, the sublist of topics will be displayed for you in the dialogue box in *JavaCake* as shown below.

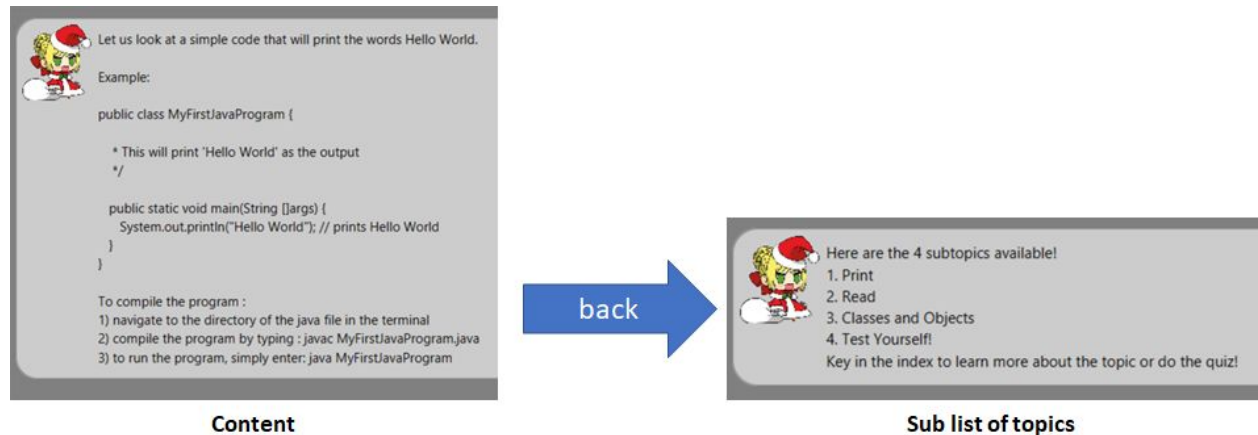


Figure 17: Content back to Sub list of topics

3.5. Doing a quiz

JavaCake offers quizzes of varying difficulties for every topic for you to reinforce your understanding of the topics incrementally.

To start a quiz,

1. Type `goto` to navigate to any subtopic with the name “Test Yourself!”



Figure 18: Quiz level displayed

2. Type `goto` to choose your difficulty.

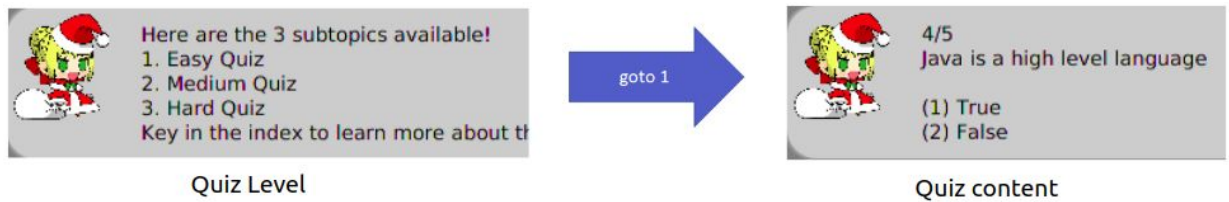


Figure 19: Quiz launched in the dialogue box

Alternatively, you can type appended `goto` numberings to jump directly into a quiz as explained in [section 3.3](#). For example, if you want to access the “Easy Quiz” in the “Java Basics” topic, you may enter `goto 1.4.1` from the main list.

Each quiz has 5 multiple-choice questions. A question can have between 2 to 5 options, of which only one is the correct answer. Enter the number corresponding to your choice to answer the question. When you have entered your answer, the quiz will move on to the next question, until you have answered all the questions in that quiz.

Once you have answered all the questions in the quiz, *JavaCake* will show you a results screen that shows you how many questions you answered correctly, as shown in the figure below.

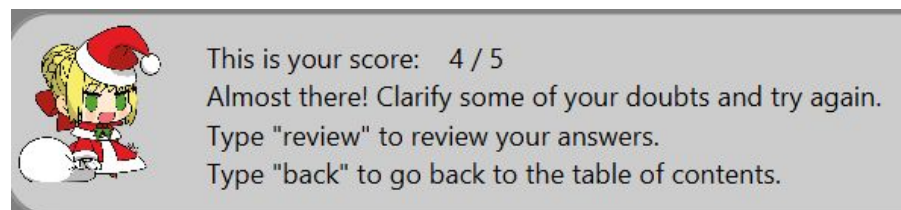


Figure 20: Results screen

Additionally, if your new score is higher than your previous score for that topic and difficulty, your total progress will increase, reflected in the progress bars in the top bar. In order to get complete *JavaCake*, you need to score full marks for all the quizzes in all three difficulties. There are an unlimited number of attempts for any quiz in *JavaCake*, and *JavaCake* will only remember your maximum score for each quiz. If you wish to start from scratch as a new user, you can delete your progress by typing the `reset` command.

From the results screen, you may choose to review your answers by typing `review` (refer to [Section 3.6](#)) or return to the table of contents by typing `back` as shown in Figure 19.



All other commands (EXCEPT `exit`) are disabled during the quiz. JavaCake will only accept valid integer inputs that correspond to the available options for that question.

You cannot exit the quiz until you have completed all the questions in that quiz and exited the results screen of the quiz.



Do not close the application in the middle of the quiz.

If you close the application during the quiz session, progress for that specific quiz session will be lost and you would have to re-do the session.

3.6. Reviewing your answers in a quiz: `review`

After you have completed a quiz, you may check your answers in that quiz by entering `review` on the results page of a quiz.

The review will show questions that you have answered, your answers to those questions and the correct answers to the questions. To navigate the questions, simply enter the question number. For example, enter “4” to go to question 4.

To quit the review, enter `back` at any point to return to the table of contents.



`review` can only be entered at the results page of a quiz.

JavaCake does not keep track of previously attempted quiz sessions after you exit the quiz.



During a review, all commands except `back` are disabled.

For example, you cannot edit notes while in the middle of a review.

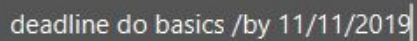
3.7. Adding a deadline: `deadline`

If you wish to add a deadline in *JavaCake*, simply use `deadline 'DEADLINE_NAME' /by 'DEADLINE_DATE'`.

To use `deadline`:

An example:

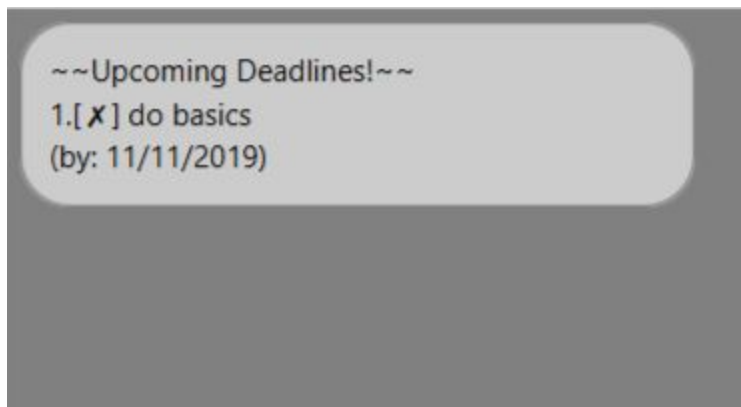
1. Type `deadline do basics /by 11/11/2019` into the command box and press Enter inside the command box as shown.



```
deadline do basics /by 11/11/2019
```

Figure 21: Input deadline in the command box

2. The newly added deadline will be shown in this format in the top-right window of the app along with the list of deadlines previously entered, in sorted order of most recent.



```
~~Upcoming Deadlines!~~  
1.[X] do basics  
(by: 11/11/2019)
```

Figure 22: List of deadlines at top-right side of app



The **deadline** feature can only accept certain date formats.

The following is a list of accepted date formats using **1st of January 2019** as base:

- 01012019
- 010119
- 01-01-2019
- 01-01-19
- 01/01/2019
- 01/01/19
- 01 01 2019
- 01 01 19
- 1 jan 2019
- 1 jan 19
- 1 january 2019
- 1 january 19

If attaching a time to the current date, the following time formats are accepted only (using **1st of January 2019, 1pm** as base)

- 1300
- 13:00

The time is appended to the back of each date input e.g.
01/01/2019 1300



Deadlines without TIME parameters are defaulted to 12 a.m. of the current date specified.

1 january 2019 is the same as **1 january 2019 0000**

3.8. Meeting a deadline: done

If you wish to set a deadline as done in *JavaCake*, simply use `done 'DEADLINE_NUMBER'`.

To use `done`:

An example:

1. Type `done 1` into the command box and press Enter inside the command box.



Figure 23: Input done in the command box

2. The selected deadline will have its 'x' symbol replaced with a '✓' instead, indicating that the deadline has been met.

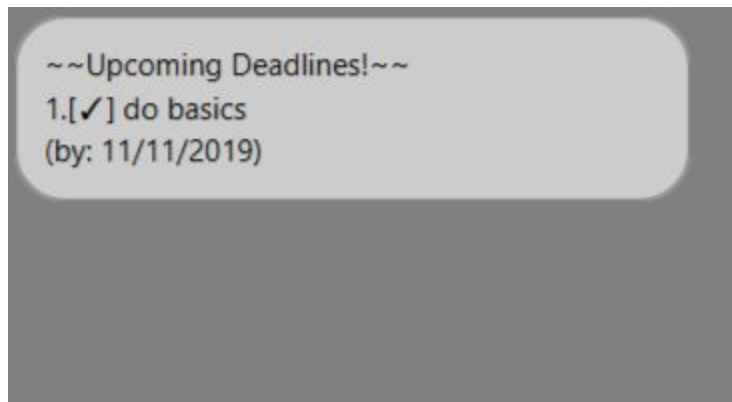


Figure 24: Selected deadline is labelled as 'done'

3.9. Deleting a deadline: delete

If you wish to set a deadline as done in *JavaCake*, simply use `delete 'DEADLINE_NUMBER'`.

To delete:

An example:

1. Type `delete 1` into the command box and press `Enter` inside the command box.



Figure 25: Input delete in the command box

2. The list now displays the remaining deadlines (if there is any) minus the deadline that was previously deleted.

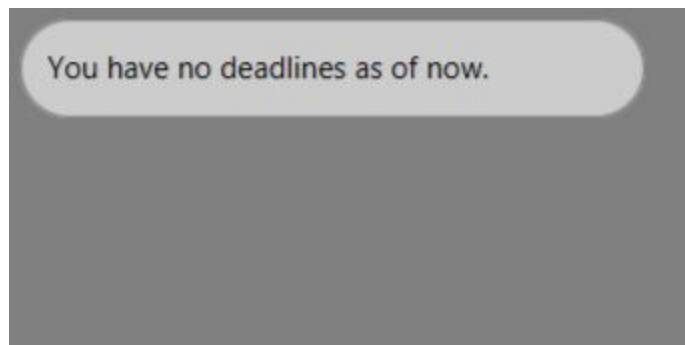


Figure 26: Selected deadline is 'deleted' from the list

3.10. Postponing a deadline: **snooze**

If you wish to change the date of a deadline in *JavaCake*, simply use `snooze 'DEADLINE_NUMBER' /by 'NEW_DEADLINE_DATE'`.

To **snooze**:

An example:

1. Type `snooze 1 /by 12/11/2019 2359` into the command box and press Enter inside the command box.



Figure 27: Input snooze in the command box

2. The selected deadline will now have its date parameter changed, as indicated in the figure below.

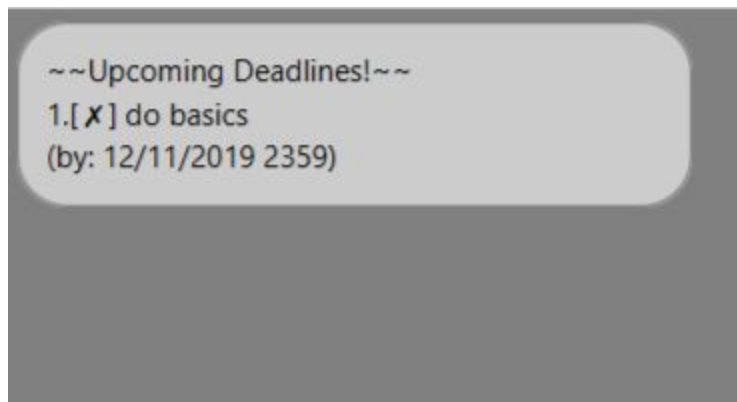


Figure 28: Selected deadline has its 'date' changed

3.11. Viewing current progress in the app: `score`

The app supports two ways of viewing your current progress; through the top bar of the GUI or inside the content's dialog box.

If you wish to see your overall quiz score in the dialog box in *JavaCake*, simply use `score`.

To `score`:

1. Type `score` into the command box and press `Enter` inside the command box.



Figure 29: Quiz score of the user is shown



The `score` command separates progress into discrete values.

Each `|` separates the different quizzes, while a **space** between 5 consecutive characters separates the different difficulties.

Each `#` indicates the exact number of marks the user has obtained throughout their use of *JavaCake*.

Each `-` indicates the remaining marks they have yet to achieve.

3.12. Creating personal notes in the app: `createnote`

If you wish to write your own notes to consolidate your learning, simply use `createnote`. You can also name your notes by `createnote 'name of note'`.

To `createnote`:

1. Type `createnote` into the command box and press Enter.


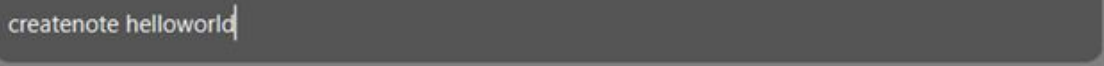


Figure 30: Input `createnote` in the command box

To `createnote 'name of note'`:

1. Type `createnote 'name of note'` into the command box and press Enter.
2. For example, if you wish to name your file 'helloworld', simply type `createnote 'helloworld'` into the command box and press Enter as shown in Figure 31.



File [helloworld] has been created successfully!

Figure 31: Creating a note file 'helloworld'



Creating notes without specifying file name will be given a default file name of 'Notes'. Subsequently, a number will be appended to ensure a unique file name.

`createnote` will produce a file - 'Notes'. Using `createnote` again will produce 'Notes1'.



JavaCake does not allow special characters for file names and limit the length of name to be 20 characters.

Special characters that are not allowed:

/	\n	\r	\t	\0	\f	`	?	*
\\	<	>		\	:	.	,	



When creating notes with file names that already exist, JavaCake will notify you to edit the existing note ([section 3.13](#)) instead.

E.g. trying to `createnote` 'helloworld' note again will prompt user to edit current 'helloworld' note.

3. Upon creation, the list of notes can be seen in the side window as seen below.

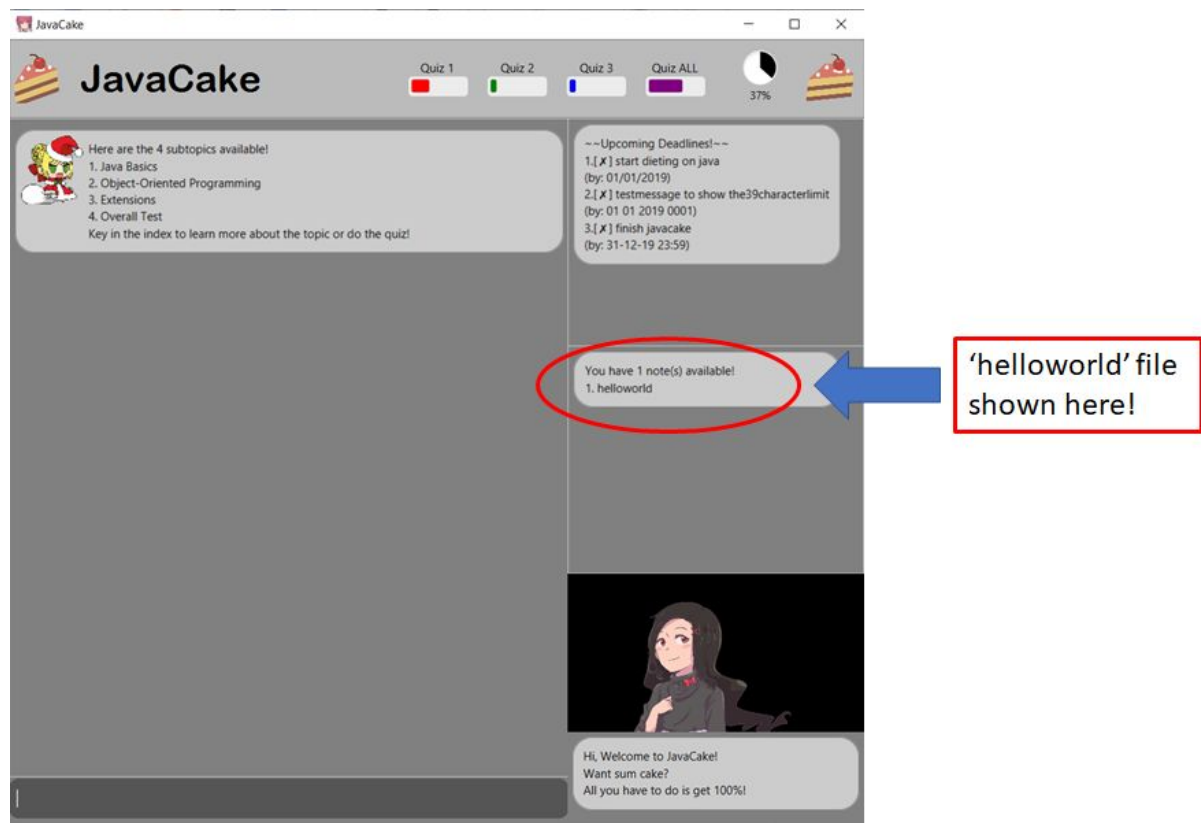


Figure 32: 'helloworld' file being stored and displayed at side window

3.13. Editing personal notes in the app: editnote

If you wish to write new content in the notes or make changes to a note, simply use `editnote 'name of note'`. You have to make sure that the note you wish to edit exist and you can view all the notes that exist in the side window of *JavaCake* as seen in *Figure 34*.

To `editnote`:

1. Type `editnote` followed by the name of the note into the command box and press Enter shown in figure below.



Figure 33: Input editnote notes in the command box

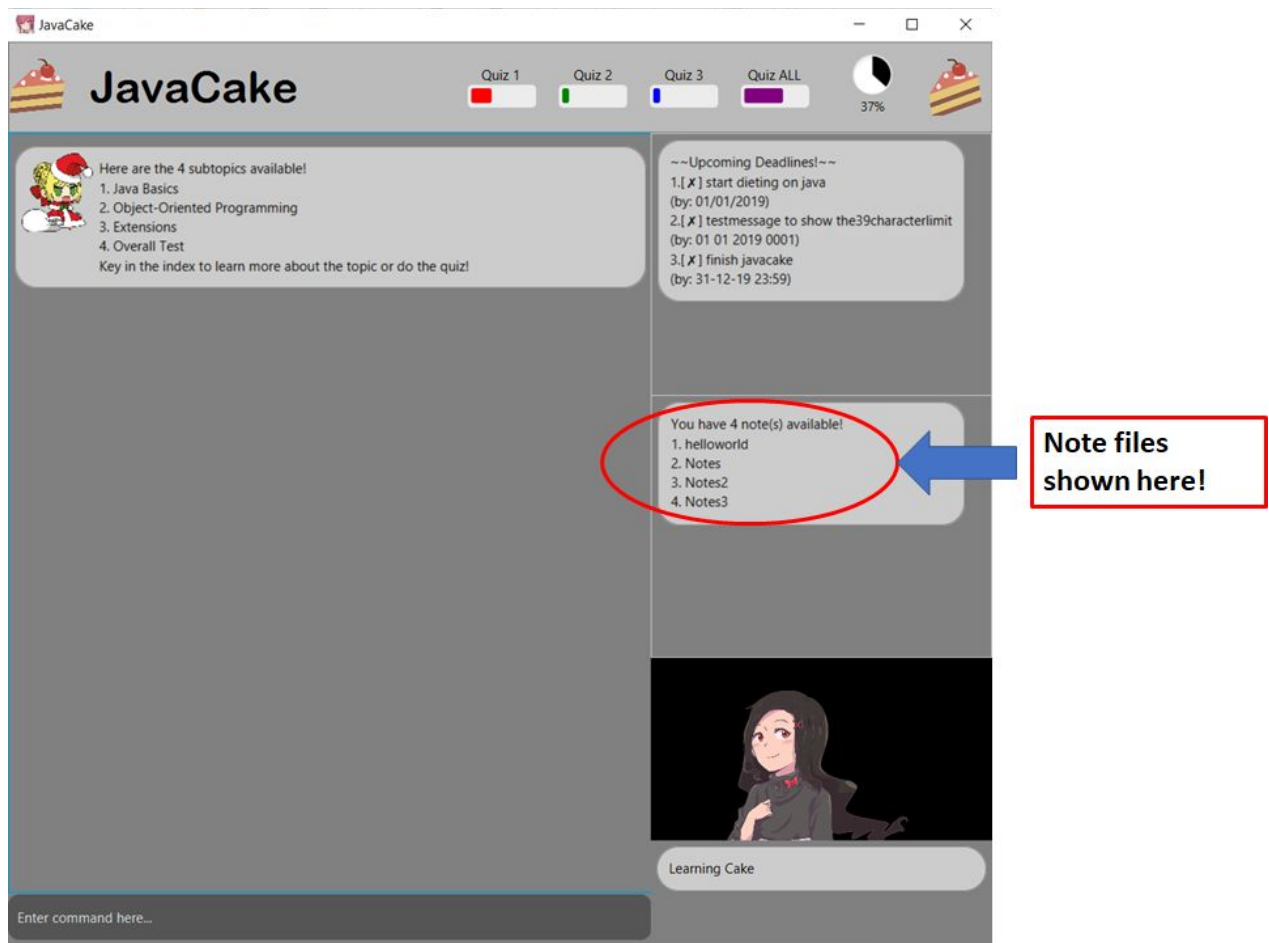


Figure 34: View of the pre-existing notes in JavaCake

2. If you have the particular note contains content, all the content will be displayed for you. You can choose to copy the old content and append the new content before saving as shown below.

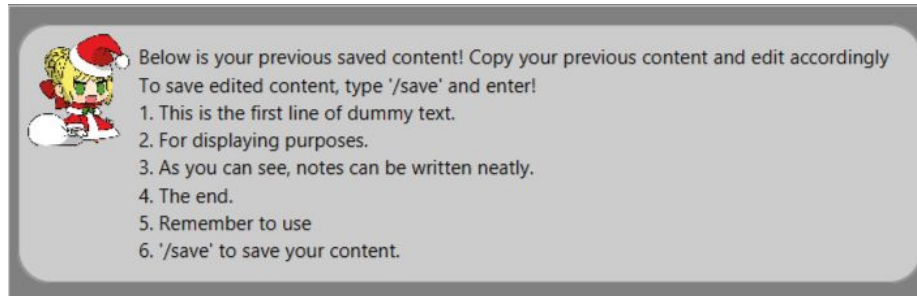


Figure 35: Pre-existing notes stored being displayed

3. After you are done writing new content into the note, save all your content by using `/save`.



Take note that (no pun intended) by using editnote, all previous content written will be WIPED by the new content.

If you only wish to read the content without making any changes, you can use the viewnote command ([section 3.14](#)) instead.

3.14. Viewing personal notes in the app: **viewnote**

If you wish to write view the content of the note without the need to make changes, simply use **viewnote** 'name of note'. You have to make sure that the note you wish to view exist just like in [section 3.13](#).

To **viewnote**:

1. Type **viewnote** followed by the name of the note into the command box and press Enter shown in figure below.



Figure 36: Input viewnote notes in the command box

2. The current save in 'notes' will be displayed as shown below.

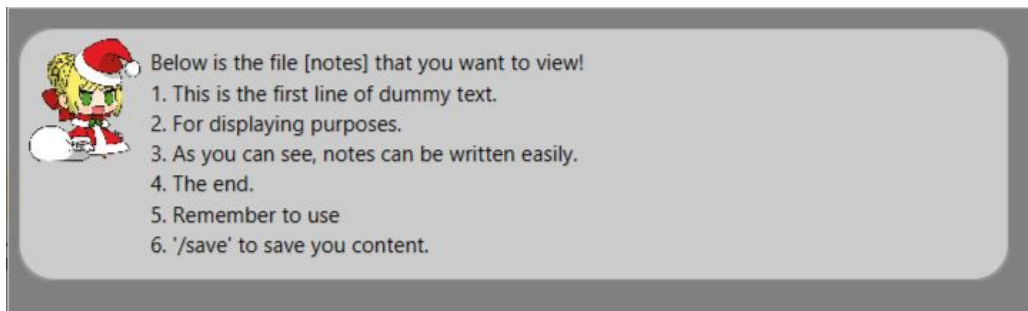


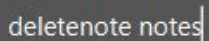
Figure 37: Display of current content written in 'notes'

3.15. Deleting personal notes in the app: deletenote

Too many notes can be hard to manage, as such, you can choose to delete the notes that you feel are old or irrelevant by simply using `deletenote 'name of note'`. You have to make sure that the note you wish to delete exist just like in [section 3.13](#).

To `deletenote`:

1. Type `deletenote` followed by the name of the note into the command box and press Enter shown in figure below.



```
deletenote notes|
```

Figure 38: Input deletenote notes in the command box

2. A notification message will be displayed to notify that 'notes' has been deleted as shown below.



File [Notes] has been deleted successfully!

Figure 39: Success message displayed upon deletion of notes

3.16. Listing personal notes in the app: `listnote`

To refresh the notes that you have created, you can simply use the `listnote`.

To `listnote`:

1. Type `listnote` into the command box and press Enter.



Figure 40: Input listnote in the command box

2. The side window for notes will be refreshed to display all the current notes available.

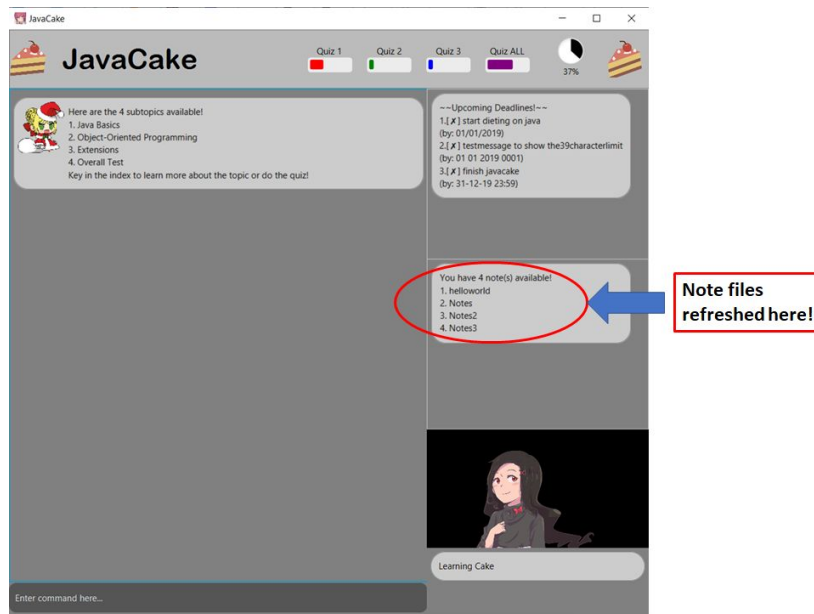


Figure 41: Side window for notes will be refreshed



`listnote` is hardly used since the previous four note commands automatically refreshed the list of notes available.

3.17. Exiting the program: `exit`

When the `exit` command is used, JavaCake will display a short goodbye message (Refer to Fig 42) and shut down shortly after.

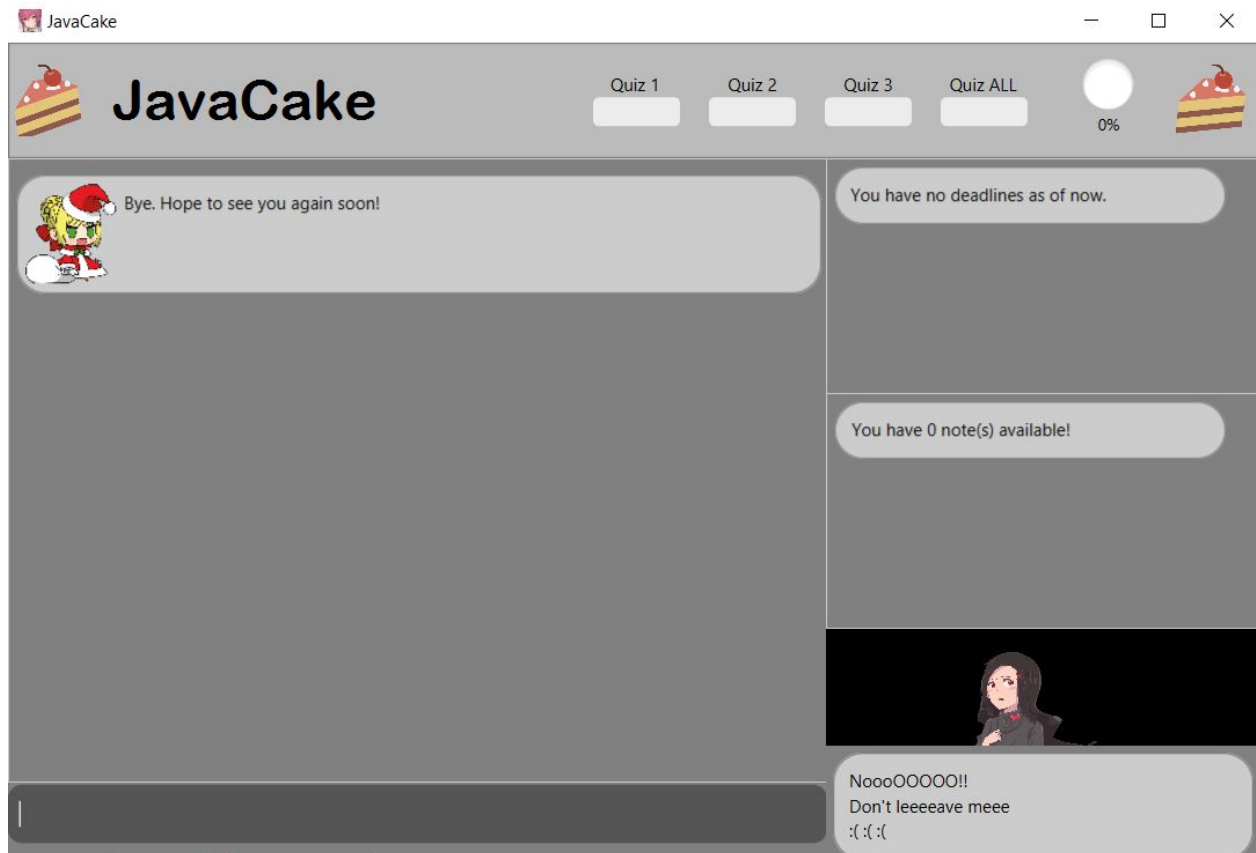


Figure 42: Message in GUI upon exit

3.18. Saving data:

Profile data is automatically saved after each command.

3.19. Helper for commands:

If you have made a small typo in inputting a command, JavaCake will help you out by replying to the correct command.

For example, if `lissr` is entered instead of `list` as shown in *figure 43* below, JavaCake will prompt you for the correct command, as shown in *figure 44*.



Figure 43: User inputting incorrect command

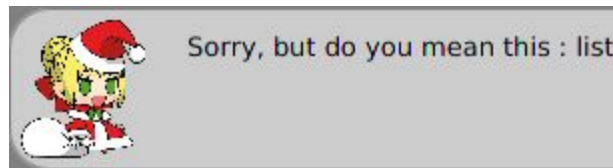


Figure 44: JavaCake helping user to use the correct command

4. FAQ

Below are some frequently asked questions about the usage of JavaCake.

Qn: Will I be able to master the Java programming language from using this app?

No, this app serves as a beginner's tutorial by introducing the basics of Java. To learn more about Java, refer to the [Java Documentation](#).

5. Do we really need Common Command Examples

Refer to [Developer Guide's "Appendix C: Use Cases"](#)