Technical University of Cluj-Napoca

Computer Science and Automation Faculty

Computer Science Department

2nd semester 2015-2016

Programming Techniques

Homework 5

Pascaru Cosmina –Roxana

Group: 30425

CONTENT

Assignment Objectives . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

Problem Analysis. Modeling . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .4

Usage Scenarios . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . ..4

Design . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .6

* UML Diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .6
* Package Diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
* Sequence Diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .8
* UseCase Diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9
* Classes . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9

Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10

Conclusions. Further implementations . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11

Bibliography . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .11

1. Assignment Objectives

1. Study the Java Collection Framework Map <https://docs.oracle.com/javase/tutorial/collections/interfaces/map.html>

2. Consider the implementation of one of the following:

a) A dictionary of Romanian language or a dictionary of English language or

b) A dictionary of synonyms (thesaurus) for Romanian or English language. It is required to use Java Collection Framework Map for the implementation.

Define and implement a domain specific interface (populate / add / remove / copy / save / search, etc.). Consider the implementation of specific utility programs for dictionary processing. For example: - Implement a method for checking dictionary consistency. A dictionary is consistent, if all words that are used for defining a certain word are also defined by the dictionary. - Implement dictionary searching using \* (any string, including null) and ? (one character). For example, you can search for a?t\*. Use the above examples to warm up your imagination.

Note. The good things acquired as a result Homework 4 (i.e. contracts, invariants, assert, separating the interface from implementation, javadoc, etc.) will be also used for this homework.

I chose to implement a dictionary of English synonyms that will maintain its consistency when adding or removing new words/synonyms.

1. Problem Analysis. Modelling

The application simulates a dictionary of synonyms. It has multiple functionalities, such as adding a new word, adding synonyms for a word, viewing all the words in the dictionary and also removing and searching for a word.

The application uses a JSON file to store the added words. The file is updated at each addition/deletion of words.

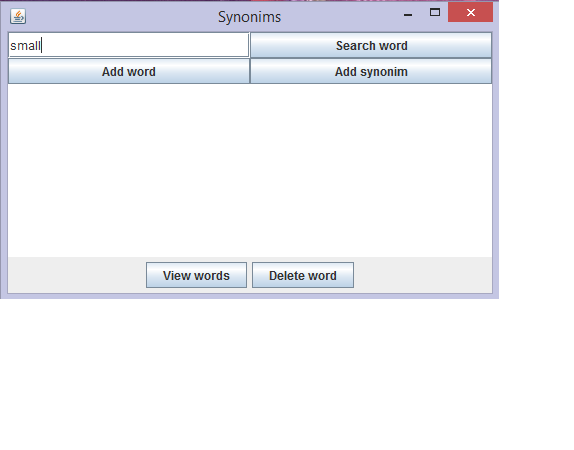
A word can be added on its own, or it can be added with a synonym. If the second case occurs, then the synonym will also be added to the dictionary, having as synonym the initial word. This makes the dictionary keep its consistency.

When deleting a word, the program checks the number of synonyms it has; if it has only one, then the word can be easily deleted and then its synonym will be checked. If it has more than one synonym, then all its synonyms will be checked.

The search option also has some capabilities. A “?” can be replaced by a letter and a “\*” can be replaced by multiple letters.

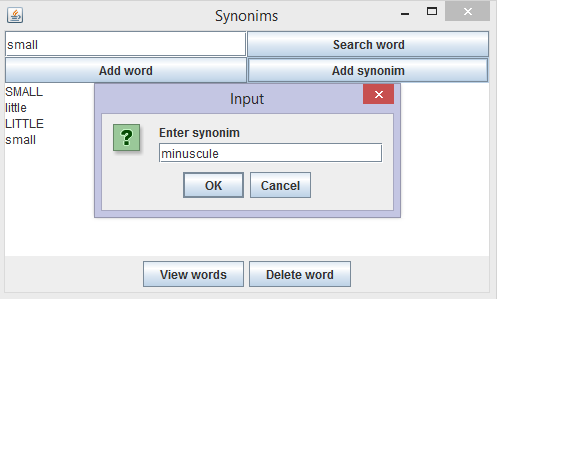
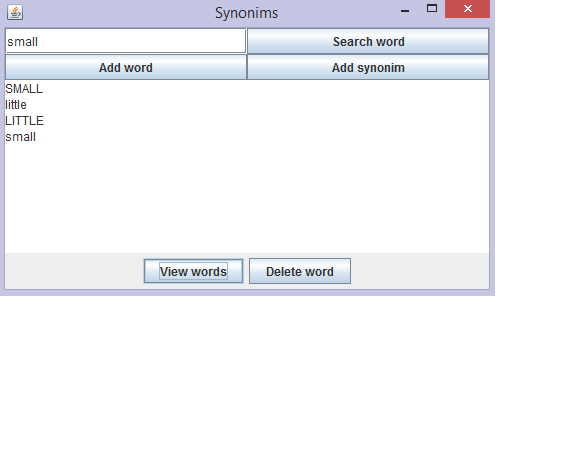
1. Usage scenarios

The application has a search bar and button and then multiple buttons for the operations. The result is displayed in a text area.



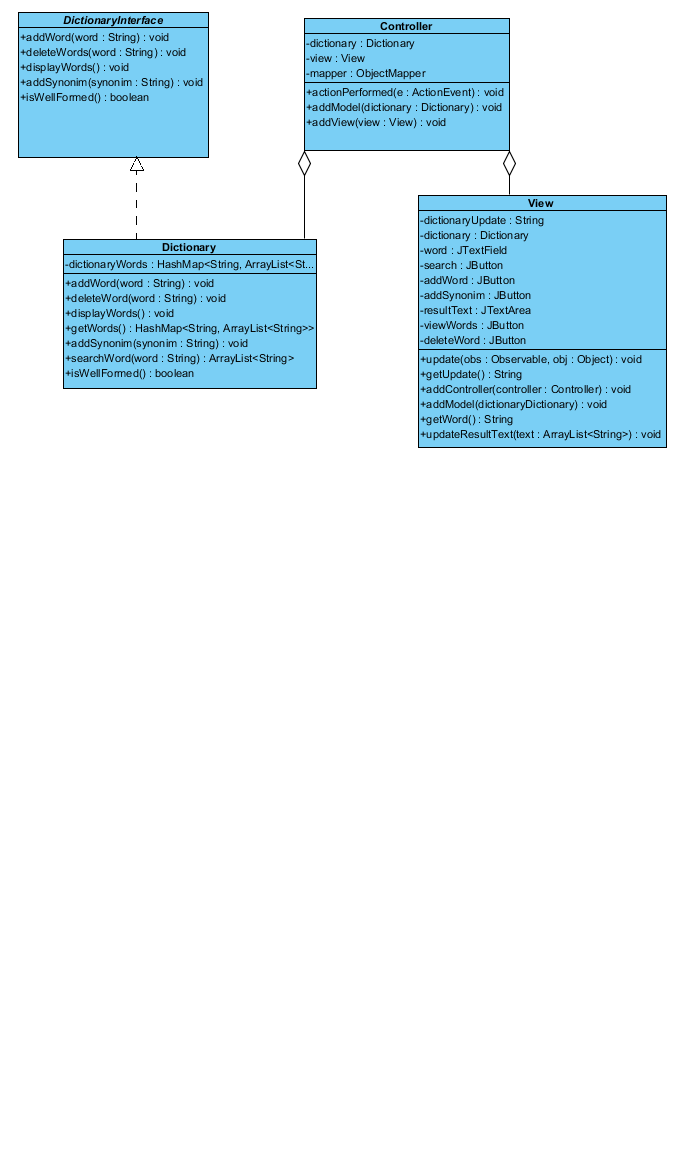
The user can add a new word, or synonyms to a word. If the word does not exist, then it will be created, if it does, then the synonym will just be added to its list of synonyms.

The “view words” button will display all the words in the dictionary.



1. Design

4.1 UML Diagram



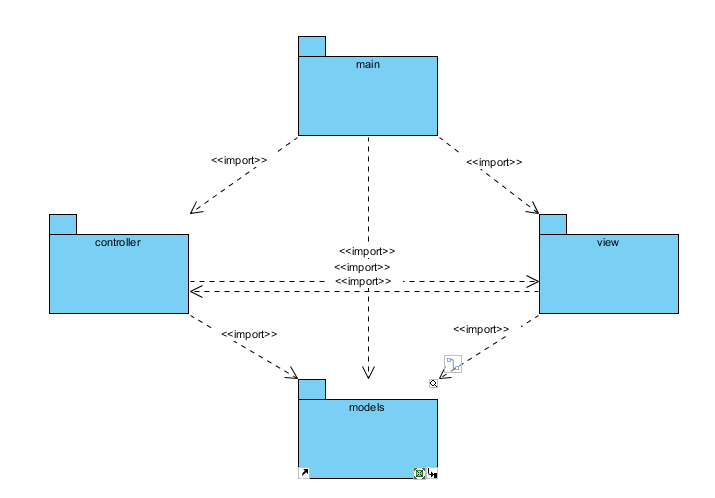
The class “Dictionary” implements the interface “DictionaryInterface”. The interface has pre and post conditions for each method. The method isWellFormed from the class “Dictionary” checks the consistency of the dictionary.

The project uses the Observable and the Model-View-Controller design pattern. The model is represented by the class “Dictionary”. The class “Dictionary” extends the “Observable” class and the class “View” implements the “Observer” interface.

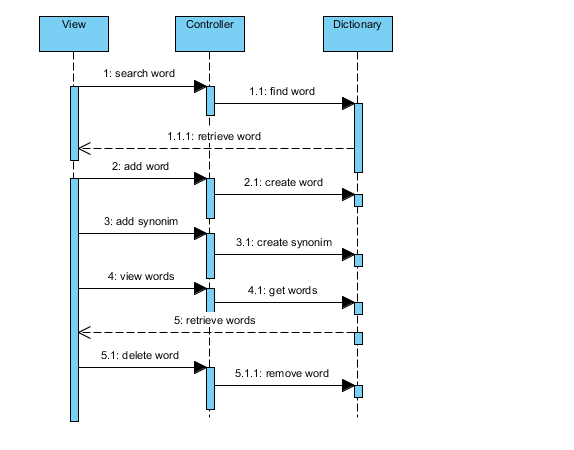
4.2 Package Diagram

The project is divided into 4 packages: main, models, view and controller.

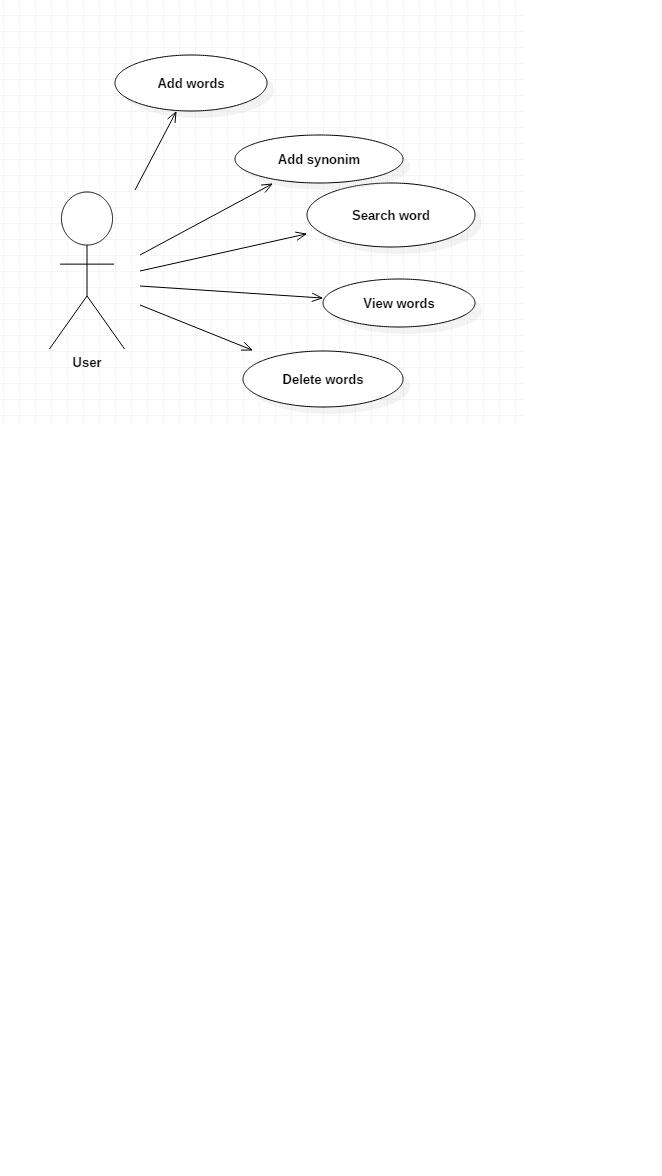
In the package models there are the interface “DictionaryInterface” and the class “Dictionary”.



4.3 Sequence Diagram



4.4 UseCase Diagram



The user can perfom multiple operations.

Adding a word will create a new word in the dictionary.

Adding a synonym will add a synonym to the list of synonyms of a word, if that word exists, else it will create a new word and then add the synonym. In order to maintain the consistency of the dictionary, each newly added synonym will also be added as a word having as synonym the initial word.

Deleting a word will also remove it as a synonym from other words.

4.5Classes

DictionaryInterface

* Is an interface
* Has pre and post conditions
* Contains the methods: addWord(), deleteWord(), displayWords(), addSynonim(), searchWord() and isWellFormed()

Dictionary

* Implements the DictionaryInterface
* Has the methos isWellFormed() which checks the consistency of the dictionary
* Has a HashMap which hold the info about the words and their synonyms
* The HashMap is of the form HashMap<String, ArrayList<String>>

View

* Represents the graphical user interface class which creates the frame which displays the application
* Is composed of a text field, multiple buttons and a text area

Controller

* Implements ActionListener
* Contains all the functionalities for the class View

1. Results

The application is user friendly, it is easy to use and any further developments will be easy to implement.

It has many functionalities and more could be added. Because the Observer is used, each operation that takes place is registered and the user gets a notification each time they update the dictionary.

1. Conclusions. Further implementations

The implementation of the dictionary was relatively simple, as it did not require a large number of classes. Of course it could have been done in a more sophisticated way, but for the basic implementation it was a really easy project to develop.

Some further implementations would include using a database to store the words and the synonyms, using more design patterns, improving the search option to be able to use the “\*” and “?” at the same time, maybe using a JTable to display the results.It would also be useful to also include a short definition of each of the words and maybe even pictures.

1. Bibliography

<https://docs.oracle.com/javase/7/docs/api/java/util/Observable.html>

<http://www.javaworld.com/article/2077258/learn-java/observer-and-observable.html>

<http://www.javaworld.com/article/2077258/learn-java/observer-and-observable.html>

<http://www.javaworld.com/article/2077258/learn-java/observer-and-observable.html>

<http://crunchify.com/how-to-write-json-object-to-file-in-java/>

<http://stackoverflow.com/questions/10926353/how-to-read-json-file-into-java-with-simple-json-library>

<http://winterbe.com/posts/2014/07/31/java8-stream-tutorial-examples/>