inspired on I. Hamid’s previous lab

SE lab 1 - 09 january 2009

I) Introduction

In this lab session we will implement a simple library management software that is going to store the authors and titles of our book collection. The system has the following functional requirements:

- Store an unlimited (theoretically) number of books entries, where each book entry contains an author and a title.
- The application must provide the following operations:
  - Adding new entries, there may not be two entries with the same title.
  - Deleting existing entries based on the title.
  - Listing all the entries in the database.
  - Searching the database for a given title and displaying the matching record (if it exists).

The non-functional requirements (which in this case are only interface requirements) are:

- At startup the user is presented with a list of operations. He can choose among:
  - ’a’ to add a new record
  - ’s’ to search the database
  - ’d’ to delete an existing record
  - ’l’ to list all the entries in the database
  - ’q’ to quit the application
- After completion of the asked operation (except for choice ’q’) the same menu will be displayed.
- If the user presses ’a’:
  - he is prompted a new title
  - he is prompted the author
- If the user presses ’s’:
  - he is prompted the title to search for
  - if found, the book’s author is displayed, otherwise an appropriate message is printed
- if the user presses ’d’:
  - he is prompted the title of the entry to delete
  - if found, the user is asked to confirm the deletion, otherwise an appropriate message is printed
  - if the user confirms the deletion, the record is removed from the database
- if the user presses ’l’:
  - all the entries in the database are printed with title and author

II) Specifications

a) Use case diagram

Exercice 1 Draw a simple use case diagram that recapitulates the different actions that the user can do. You will use a primary actor: the user, and a secondary actor: the database.

b) Model View Controller

We are going to design our application around a well-known architectural pattern called MVC (Model View Controller). The idea behind MVC is to separate in different components: the data of the application (handled by the model), the presentation of the data (handled by the view) and the operation on data (handled by the controller).

In this case the View is the class that will handle the textual interface described in the non-functional specifications. The Controller is the class that will provide the different operations needed: add, search, list, delete. Finally the Model is the class that will store the entries.

c) Class Diagram

Exercice 2 Design a class diagram that satisfies a model view controller pattern for our library management application. You will use four classes:

- UserInterface (the view)
- Controller (the controller)
- Book, a class that models an entry.
- Database (the model), which contains many Books.

Add all the attributes, methods and associations that you deem necessary.

d) Sequence Diagram

Exercice 3 Design a sequence diagram to specify the interactions during:

- the 'add' operation
- the 'delete' operation

III) Implemenation

Exercice 4 If you have time left, implement the above specifications! For implementing the user interface look at the java.io.Console class in the standard API.