BookBooking code review.

Overview

The link contains two service files. All in all it has less than 150 lines of core code. Each file contains only two methods with logic.

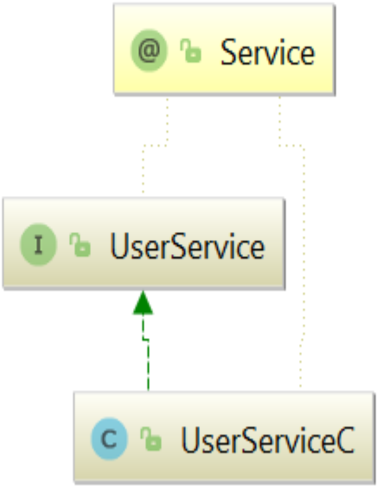
Goals

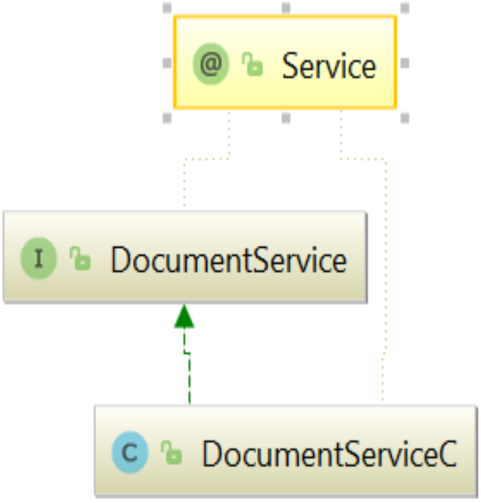
Usage Scenarios

Functions in UserService is responsible for checkouting, renewing the documents.

Functions in DocumentService is responsible for the creating queue and outstanding request for queue.

Class Diagrams





Code

Categories for review

1. Design decisions

[1.1] Reviewer1: I think it would be better if you used the "case", instead of "if"

[1.1.1] Owner: Invocation of case statement entails may duplicates of code.[CLOSED]

[1.2] Reviewer2: some functions return String, which is only “true” or “false”. Probably can switch to boolean.

[1.2.1] Owner: it is done to avoid duplications of code in controller class, where directly returned the value of such kind of functions(boolean string should be returned)[CLOSED]

[1.3] Reviewer2: checkoutDocument function return String. It could be “true” or “false” or contain some message. Strange design implementation. May be return pair, where first parameter is status code (int or bool) and second parameter is message (String)?
[1.3.1] Owner: Some message just third state(other then true or false). I fully agree that it is kind of strange. But it is specified in documentation and it directly returned to the client. [TO BE DISCUSSED]->[CLOSED]
[1.4] Reviewer2: keepingTime is always calculated using big numbers inside main code and incomprehensible for third-party programmers. Better to make some constants and make calculations like 3 * MINUTE.
[1.4.1] Owner: Fully agree, will be changed [TO BE FIXED]->[CLOSED]

2. API design

3. Architecture (including inheritance hierarchy, if any)

4. Implementation techniques, in particular choice of data structures and algorithms

[4.1] Reviewer3: Why don't you use more lightweight data types for such data as status of document or type of user? In context of big application it will help you to save memory and in developing context it will allow you to avoid typos and save time.
[4.1.1] Owner: In some cases it will be difficult to remember what state means the Integer status, programmer should always refer to the documentation. And in case of strings it is easy to remember states, and typos are underlined by IDE. [TO-BE-DISCUSSED]->[CLOSED]
[4.2] Reviewer3: In getQueueForDocument method you implemented two loops: in first loop you’re copying data from one list to another with time of users quantity, and then, in second loop, you're copying this data to another several lists with same time. I don't see the need for first loop and first list considering, that you could just copy data from it's initial storage to several lists, where you will use them after.
[4.2.1] Owner: Fully agree, it will be changed.[TO BE FIXED]
[4.3] Reviewer3: Is it possible to don't use getQueueForDocument at all? Seems that every time when you need to do something with queue(I think, very often) you need to build it again and again. Can you store this queue as array in database? If it is not suitable for your application, please explain why.
[4.3.1] Owner: No, it is not possible. This method simply extract list of users for the particular document from the Database. Since User can not have the order to the book as well as the position in queue, queue table is combined with the order table in Database. [TO BE DISCUSSED]->[CLOSED]
[4.4] Reviewer2: queueRequest return String, which is always “true”. Probably can switch to boolean. If “true” is always returned, why don’t you make it void?
[4.4.1] Owner: It is not void to directly return the value of the function to the client, which must be string. [CLOSED]

5. Exceptions handling - Contracts

6. Programming style, names

[6.1] Reviewer2: some variable names are just one letter like “o” or “u”. Even if they are used one time, I would recommend to make them understandable.

[6.1.1] Owner: I think it is better to have the same short name for frequently user objects in the program, e.g. “u” is always user object, and “o” is always order object.[TO BE DISCUSSED]->[CLOSED]

7. Comments and documentation

[7.1] Reviewer4: Comments should always be on the same language, and very preferably for it to be English as the most common world language

[7.1.1] Owner: Fully agree, it will be changed.[TO BE FIXED]->[CLOSED]

[7.2] Reviewer2: Very few comments. Most code is understandable due to good variable names, but it would be faster to understand code in case some fors and ifs will be commented.

[7.2.1] Owner: Fully agree, it will be updated.[TO BE FIXED]->[CLOSED]