**Unit Testing with JUnit Practical**

Contents

[Part I: Quick recap the code implemented in Java Core Practical 1](#_Toc74658887)

[Section 0: Create java pojo classes 1](#_Toc74658888)

[Section 1: Implement exercises that not using java 8 Stream API 2](#_Toc74658889)

[Section 2: Implement exercises that using Java 8 Stream API, File IO and Java Time 2](#_Toc74658890)

[Part II: Implement Unit Testing by jUnit 5 3](#_Toc74658891)

[1. Exercise 1: Write Unit Test for all methods in the DatetimeUtils class 3](#_Toc74658892)

[2. Exercise 2: Write Unit Test for one function in the PostService class 3](#_Toc74658893)

# Part I: Quick recap the code implemented in Java Core Practical

## Section 0: Create java pojo classes

Please create 3 java pojo classes as described below:

|  |  |
| --- | --- |
| Class | Properties |
| Post | id |
| author |
| title |
| description |
| content |
| date |
| Author | username |
| password |
| firstName |
| lastName |
| email |
| birthdate |
| added |
| Authority | username |
| authority |

## Section 1: Implement exercises that not using java 8 Stream API

Note: *These exercises are prepared for Lambda expressions, Functional interfaces and Method references. So please do not use Stream API and java.util.function package*.

|  |  |  |
| --- | --- | --- |
| Exercise | Content | Suggest class name |
| 1 | Given a list of authors, find author by name | AuthorService |
| 2 | Given a list of authors, find author by email | AuthorService |
| 3 | Given a list of posts, find all posts of the specific author | PostService |
| 4 | Given a list of posts, find all posts that their title or content contains the specific string | PostService |
| **5** | **Given a list of posts, find the top 10 newest posts of one specific author** | **PostService** |
| 6 | Given a list of posts, find the top 10 newest posts of one specific author and their title or content contains the specific string | PostService |
| 7 | Given a list of posts, find all posts of all users (group by user) | PostService |
| 8 | Given a list of posts, find all author's names | AuthorService |
| 9 | Given a list of authorities, find all authorities of the specific user | AuthorityService |

## Section 2: Implement exercises that using Java 8 Stream API, File IO and Java Time

Note: *These exercises are prepared for Stream API, Optional Type, File IO and Java Time*

|  |  |
| --- | --- |
| Exercise | Content |
| 1 | Re-write all functions/methods in Session1 using Stream API and java.util.function package |
| 2 | Given a list of posts, find all posts created today |
| 3 | Given a list of authors (A) and a list of posts, find all posts which are created by authors in the list (A) |
| 4 | Given a list of authors (A) and a file named posts.csv which is formatted as below: author|title|description|content|date  Write a function that gets all posts from the specific date on. Things need to be done: - Read csv file, handle FileNotFoundException - Filter all posts later than the specific date - Filter all posts which are created by authors in the list (A) - Close resources |
| **5** | **Write a DatetimeUtils that contains some important functions:  - formatDate: Inputs *date* in java date, *format pattern* in string. Expected output: the date in string with the predefined pattern.**  **Ex: public static String formatDate (LocalDateTime date, String pattern) {};**  **- createDate: Inputs *date* in string, *format pattern* in string. Expected output: the *date* in java date.**  **Ex: public static LocalDateTime createDate(String date, String pattern) {};**  **- convertDate: Input date in string, format pattern in string to be converted, format patter in string to convert to. Expected output: the date in string with converted pattern.**  **Ex: public static String convertDate(String date, String fromPattern, String toPattern) {};** |

# Part II: Implement Unit Testing by jUnit 5

Note: These exercises will be used to implement unit testing by jUnit 5 (jUnit Jupiter version 5.7.2)

1. **Exercise 1**: Create StringUtils class and provides:

* Method to check string is a positive integer value.
* Method to check given string is a valid IPV4 address.

1. **Exercise 2**: Write Unit Test for all methods in the DatetimeUtils class. DatetimeUtils class was implemented in section 2, exercise 5.
2. **Exercise 3**: The code of function that implemented in section 1, exercise 5 of PostService class.