6. Java Stream API

Monday, November 30, 2020. 8:24 PM

1. **What is Stream API in Java +**
2. **Problems that Stream API solves.+**
3. **Characteristics of a Stream+**
4. **Ways of building Streams+**
5. **Role of Functional Interfaces in Stream API+**
6. **Types of methods over Streams (Intermediate, Terminal)+**
7. **Java Stream Short Circuiting Operations+**
8. **Converting Java Stream to Collection or Array+**
9. **Parallel Execution, ForkJoinPool+**

-Djava.util.concurrent.ForkJoinPool.common.parallelism=4

10. **Method reference**
11. **Optional class**
12. **Spliterator**

Resources:
https://www.journaldev.com/2774/java-8-stream#stream-intermediate-terminal-operations
https://vertex-academy.com/tutorials/ru/java-8-stream-map/
https://www.baeldung.com/java-8-parallel-streams-custom-threadpool -
custom thread pool in java 8
https://annimon.com/article/2778 -
візуальне представлення роботи прикладів
https://docs.oracle.com/javase/8/docs/api/java/util/stream/package-summary.html -
документація
https://www.journaldev.com/2774/java-8-stream
https://vertex-academy.com/tutorials/ru/java-8-uchebnik/

https://github.com/Vedenin/java_in_examples/tree/master/stream_api/src/com/github/vedenin/rus/stream_api -
тут багато прикладів по багатьох методах

Homework:

Task 1:

Let the key of Map is project name and value contains list of participants.
Create a Stream<String> nameList(Map<String, Stream<String>> map) method
of the MyUtils class to build sorted stream of all participants without duplication.
Please ignore null or empty strings, extra spaces and case sensitivity.
Throw NullPointerException if map is null.
For example, for a given map:
{
"Desktop"=[" iVan", "PeTro ", " Ira "],
"Web"=["STepan", "ira ", " Andriy ", "an na"],
"Spring"=["Ivan", "Anna"]
}
You have to get:
["Andriy", "Anna", "Ira", "Ivan", "Petro ", "Stepan"]

Task 2:

Create a Map<String, Stream<String>> phoneNumbers(List<Stream<String>> list)
method of the MyUtils class to build a Map of all phone numbers.

The key of Map is code of network and value contains sorted list of phones.
Remove all spaces, brakets and dashes from phone numbers.
For example, for a given:
[["093 987 65 43", "(050)1234567", "12-345"], ["067-21-436-57", "050-2345678",
"0939182736", "224-19-28"], ["(093)-11-22-334", "044 435-62-18", "721-73-45"]]
You have to get:
{"050"=["1234567", "2345678"], "067"=["2143657"], "093"=["1122334", "9182736",
"9876543"], "044"=["4356218"], "loc"=["2241928", "7217345"], "err"=["12345"]}

Task 3:

Create a int sumEven(Stream<IntStream> stream) method of the MyUtils class
to sum minimal positive even numbers from all Streams.
Return zero if minimum positive even element was not found in stream.
For example, for a given:
[[-2, -4, 1, 8, 3, 10], [2, -4, 4, 0, 3, 1], [1, -4, 3, 5, 3, 1]]
You have to get: 10 (8+2+0)

Task 4:
Create a int countNumbers(IntStream intNum, Stream<String> strNum) method
of the MyUtils class to count of numbers that is divisible by 3 or contains the digit 3.
The parameters of the method are two Streams with integers and Strings with one number.
For example, for a given:
[[3, 2, 1, 13, 21, 15], ["9", "4", "23", "0", "32", "5"]]
You have to get: 7

Task 5:
Create a Stream<Integer> duplicateElements(Stream<Integer> stream) method
of the MyUtils class to return a sorted stream of duplicated elements of the input stream.
For example, for a given elements
[3, 2, 1, 1, 12, 3, 8, 2, 4, 2]
You have to get:
[1, 2, 3]