Fundamental Programming Techniques

Processing Sensor Data of Daily Living Activites

Assignment 5

Hunor Debreczeni

Group: 30423

Objective

The objective of the application is to implement a program that analyses the behavior of a person whose movement and actions are recorded by sensors installed in his/ her house. The log of the activities done by this person is saved with its starting time and ending time and by the name of the given activity.The activities may be the following: Leaving, Toileting, Showering, Sleeping, Breakfast, Lunch, Dinner, Snack, Spare_Time/TV, Grooming.

Problem Analysis, Modeling, Scenarios, Use cases

The application may be used to analyze people's behavior and their bad habits. Many people are unaware of their daily habits and spend a lot of time by doing unnecessary things. With the help of this software, it's easy to check any needless task that a person may do frequently.

In order to use the application, the end-user should simply run the *.jar* file the following way:

Java -jar <File-Name.jar>

It's also important to include an Activites.txt file in the same folder, which contain the recorded activites on each line with the following format:

<start_time> <end_time> <activty_name>

Design

For implementing the system and to make it easier to manage, the model view controller was used, so that the methods would be split up into different parts that are easy to identify later.

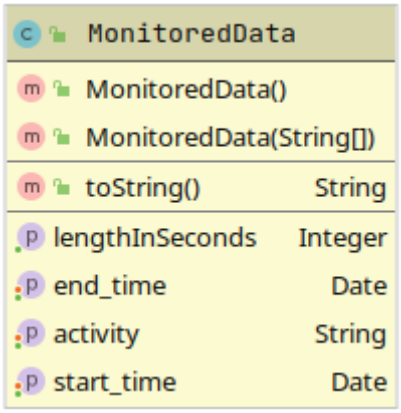
- The M**odel** represents the objects, which in our case is the MonitoredData class. It contains a given record, that specifies the start_time, end_time and the name of the activity.
- The **View** contains the handling of the output file, more exactly the results that will be visible by the end-user after running successfully the application.
- The **Controller** includes the TaskController class, that does the analysis of the data and calculates statistics like total occurrence of an activity, the duration of it, etc.

To make the development process faster and shorter, lambda expressions and stream processing were used to make the desired lists / maps.

Packages

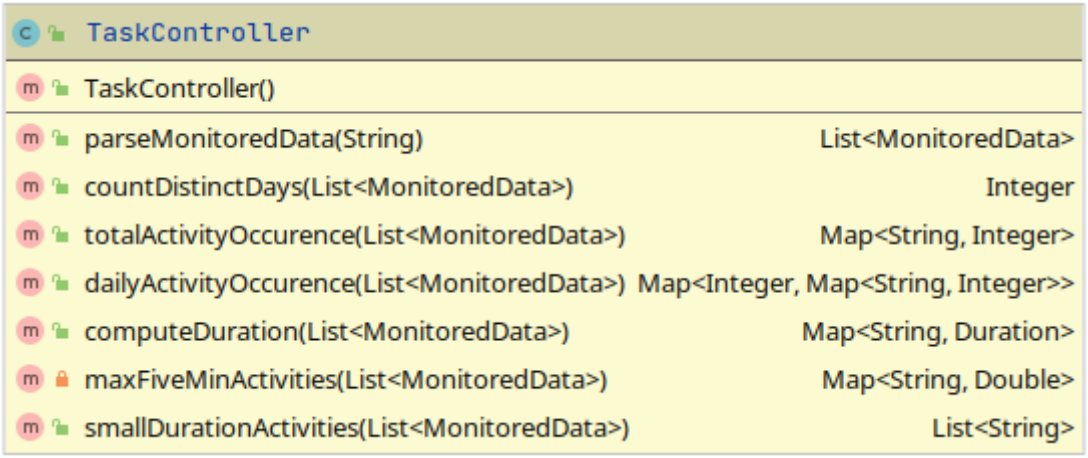
Model

Contains the objects, that only contain data. These are used to get information and analyse it, so further results can be made. The MonitoredData is the only class that appears in this package.



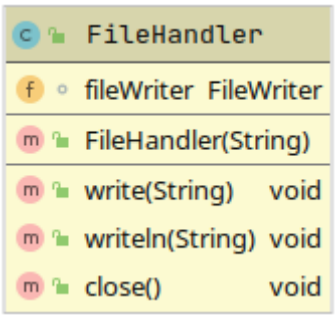
Controller

This is the core of the application. The data is processed in this package, more exactly in the TaskController class, that makes the necessary groupings and calculations to calculate different results.



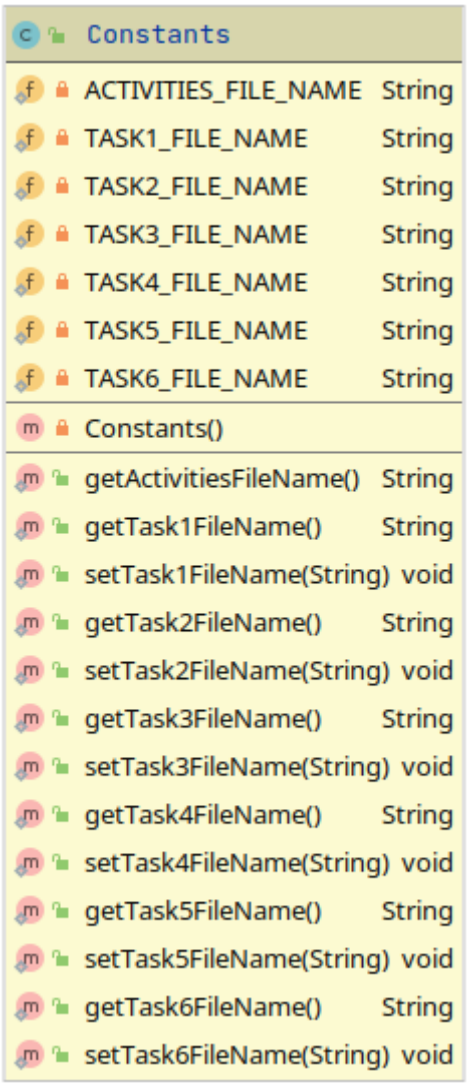
View

This package contains the FileHandler class, which simply outputs a specific text in a specific file next to the .jar application. In other words, it makes the results visible to the end-user also.



Util

Contains helper classes that are mostly use static methods/ variables. The Constants class is included, which contain the name of the input and output files that is used by the application. The use of this class helps further developments/ changes easier for other developers.



Implementation

The implementation is mostly done in the TaskController class, which implements the following analysis:

- **Count distinct days**
  Counts the number of days that contain recorded activities. It looks for the current year, current month and current day. Groups each monitoredData by the date and counts the final size of the list.

- Returns: Integer = Count of distinct days

- **Total activity occurrence**
  Groups the activities by their names and counts the number of recorded occurrences of them. Groups each monitoredData by it's activity name and counts the appearance of such cases.
  - Returns: Map<String,Integer> = Map<ActivityName,TotalOccurrence>

- **Daily activity occurrence**
  Returns a list that contains every day that tells us for each activity, that how many times it was done on that specific day. By other words, it counts that how many times an activity has appeared for each day over the monitoring period. Groups by the given date and after by the given name.
  - Returns: Map<Integer, Map<String,Integer>> = Map<Day, Map<ActivityName,Occurrence>>

- **Compute duration**
  Calculates the duration of each activity over the entire monitoring period. Groups by the name of the activity and sums up the seconds.
  - Returns: Map<String,Duration> = Map<ActivityName, Duration>
    Duration is used to make it easier to get the number of days / hours / minutes / seconds, to give more information for the end-user.

- **Small duration activities :** uses **maxFiveMinActivites**
  Filters those type of activities, that have at least 90% of their monitoring period with less than 5 minutes. This function first collects and counts those activities that have a duration of at most five minutes. Then gets every activity with their count by using the **Total activity occurrence** function and uses the following formula: *maxFiveMinActivityCount/totalActivityCount > 0.9* If it's true, than we found one of our activites that match the filter.
  - Returns: List<String> = List<ActivityName>

Results

As results, the user of this application gets a software that helps him sort out and visualize his/ her daily activites. It may also help to manage children/ workers from a company, etc.

Conclusions

I have learnt a lot about using lambda functions and their functionalities. They make programming a lot faster and easier. With the help of these expressions the code also gets easier to read for anyone.

Bibliography

- https://winterbe.com/posts/2014/07/31/java8-stream-tutorial-examples/
- https://www.baeldung.com/java-groupingby-collector