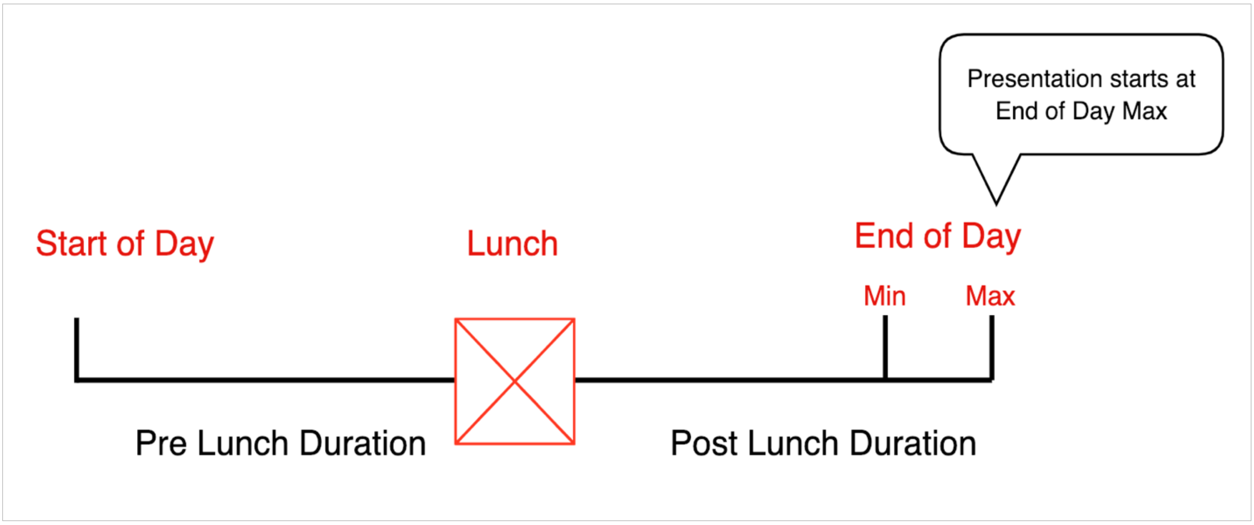
Digital Away Day

Inputs

1. Activities File – contains
   - Activity Name
   - Activity Duration (in minutes)

Constants

1. Start of Day = 9am
2. End of Day Min = 4pm
3. End of Day Max = 5pm
4. Lunch Start = 12pm
5. Lunch End = 1pm
6. Sprint Time = 15min



Assumptions

1. The input times are correct and fits the schedule based on the diagram above
2. An activity cannot be performed twice (I.e), if an activity is allocated to one schedule, it cannot be present in another schedule
3. **Pre-Lunch** Activities have to start at SOD and finish exactly at Lunch Start
4. **Post-Lunch Activities** have to start exactly at Lunch End and can finish anywhere between EOD min & EOD max

5. If the activity file has an invalid entry, only that activity is discarded/ignored. Rest of the file is processed
6. Time is assumed to be in minutes for all activities
7. Each activity has a minimum duration of 1 minute

Derived Variables

1. Pre-Lunch Duration = Lunch Start – Start of Day
2. Post Lunch Duration Min = End of Day Min – Lunch End
3. Post Lunch Duration Max = End of Day Max – Lunch End
4. Lunch Duration = Lunch End – Lunch Start
5. Maximum Possible Teams = Total Time for all activities / (Pre Lunch Duration + Post Lunch Duration Min)

Validation

1. Check if a file is empty
2. Validate lines for the correct format using regex
3. Check and eliminate duplicates
4. If Total Time of all Activities is less than (Pre-Lunch + Post-Lunch duration) then not possible to create a schedule

How to run

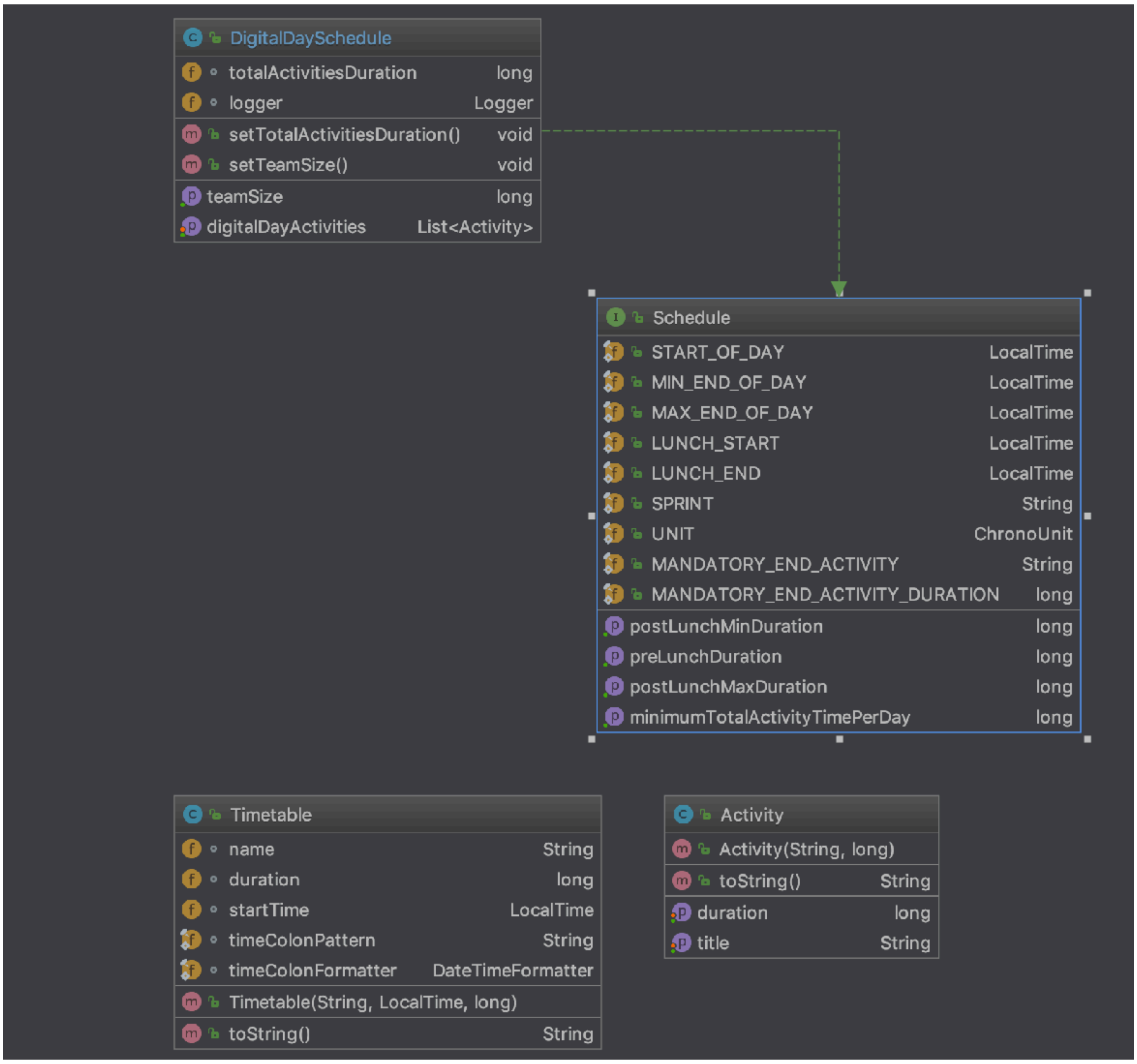
1) Environment used: Java 8, Gradle 4.3, Junit5
2) Git url :: https://github.com/ClozerUK/DigitalAwayDay
3) Download the repository ‘master’ branch
4) Import as a gradle project and run DigitalAwayDay (or)

5) Go to project root folder
6) Run ./gradlew build
7) And then

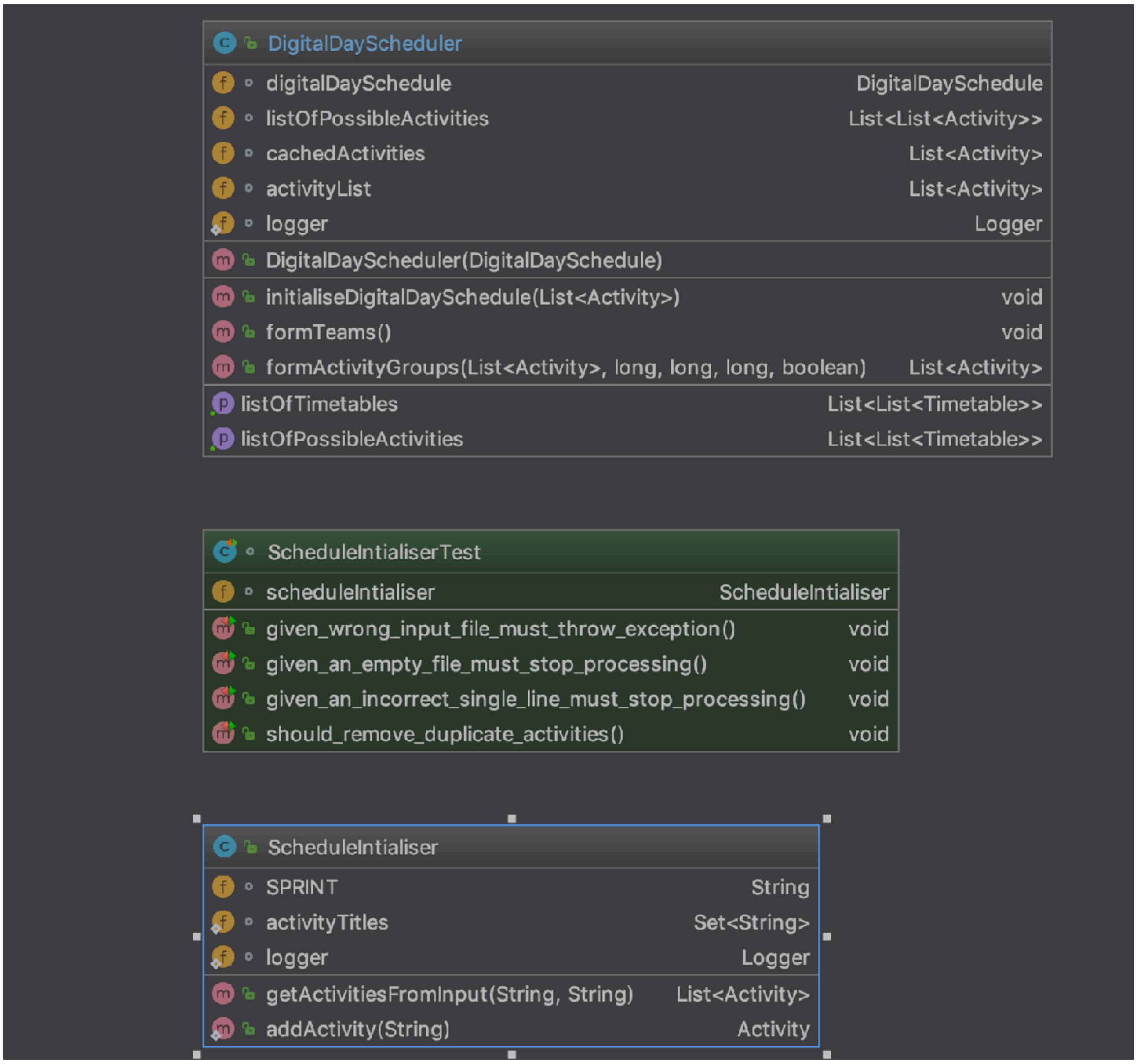
```
java -cp build/resources/main:build/classes/java/main
com.deloitte.digital.DigitalAwayDay
```

Solution / Implementation Details

Model



Service

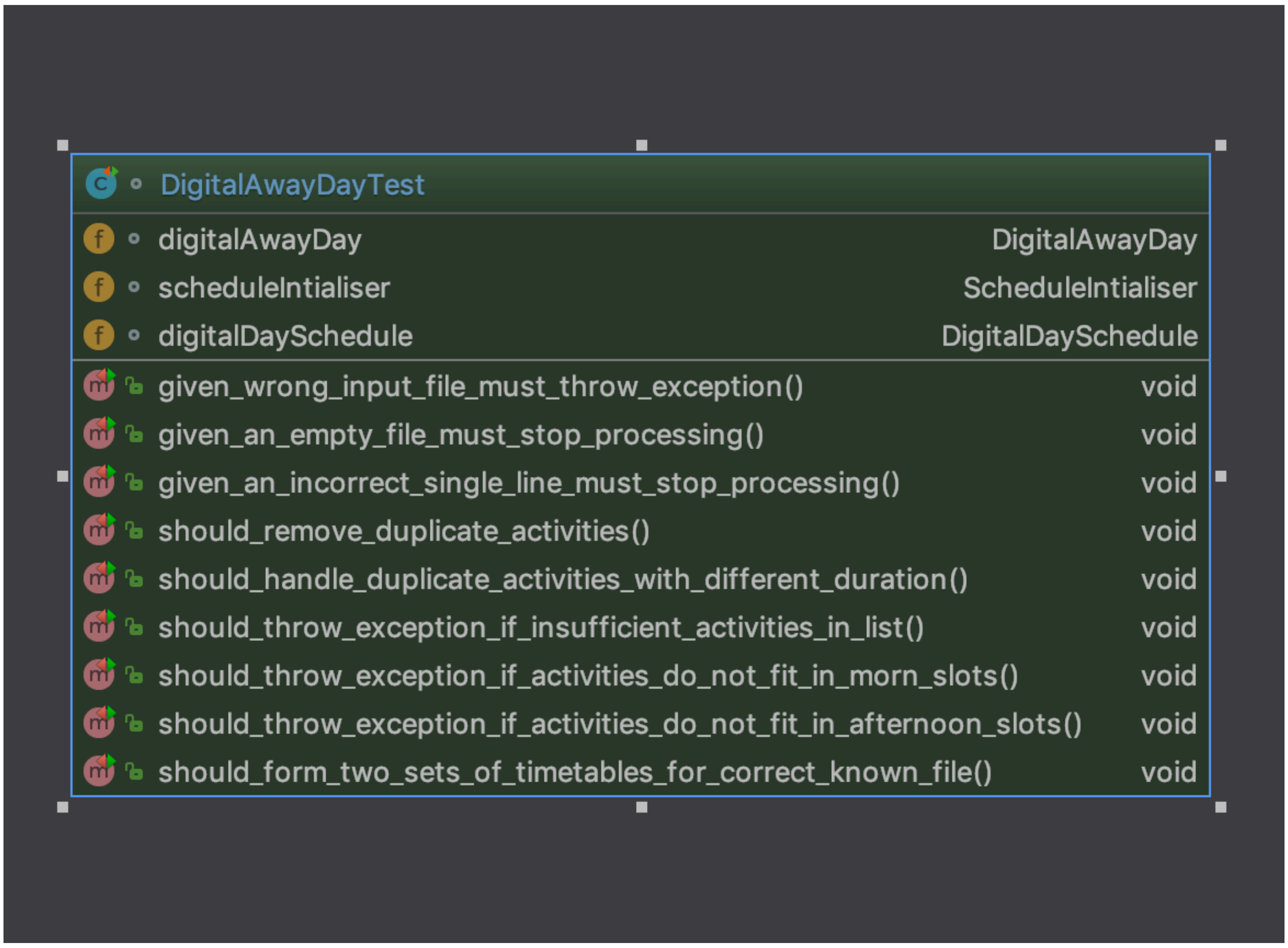


DigitalDayScheduler	
digitalDaySchedule	DigitalDaySchedule
listOfPossibleActivities	List<List<Activity>>
cachedActivities	List<Activity>
activityList	List<Activity>
logger	Logger
DigitalDayScheduler(DigitalDaySchedule)	
initialiseDigitalDaySchedule(List<Activity>)	void
formTeams()	void
formActivityGroups(List<Activity>, long, long, long, boolean)	List<Activity>
listOfTimetables	List<List<Timetable>>
listOfPossibleActivities	List<List<Timetable>>

ScheduleIntialiserTest	
scheduleIntialiser	ScheduleIntialiser
given_wrong_input_file_must_throw_exception()	void
given_an_empty_file_must_stop_processing()	void
given_an_incorrect_single_line_must_stop_processing()	void
should_remove_duplicate_activities()	void

ScheduleIntialiser	
SPRINT	String
activityTitles	Set<String>
logger	Logger
getActivitiesFromInput(String, String)	List<Activity>
addActivity(String)	Activity

Tests



```
DigitalAwayDayTest
digitalAwayDay                                                   DigitalAwayDay
scheduleIntialiser                                               ScheduleIntialiser
digitalDaySchedule                                               DigitalDaySchedule
given_wrong_input_file_must_throw_exception()                    void
given_an_empty_file_must_stop_processing()                       void
given_an_incorrect_single_line_must_stop_processing()            void
should_remove_duplicate_activities()                             void
should_handle_duplicate_activities_with_different_duration()     void
should_throw_exception_if_insufficient_activities_in_list()      void
should_throw_exception_if_activities_do_not_fit_in_morn_slots()  void
should_throw_exception_if_activities_do_not_fit_in_afternoon_slots()  void
should_form_two_sets_of_timetables_for_correct_known_file()      void
```

How it all works

Read and Validate File - *ScheduleInitialiser Class*

1. Read the given input file
2. Validate for non-empty file length, valid line format
3. Check that duration is more than 1 minute
4. Remove duplicates if exist
5. If an activity is repeated twice, with different durations, only the first entry is recorded
6. Create a list of valid activities

Get the Digital Day Away Scheduler – *DigitalDayScheduler Class*

1. Initialise Digital Day Schedule
   (1) Set the digital day list of activities read from the file
   (2) Find the total duration of all the activities in list and check if it can fit into the daily day's schedule
   (3) Find out how many teams are possible given the list of activities
2. Form teams with list of activity per team
   (1) Build Cache of shortlisted activities
   (2) Activities are sorted from the longest duration to the shortest
   (3) Iterate through the activity list to see what activities would fit into the pre-lunch and post-lunch block

Duck Herding	60
Laser Clay Shooting	60
Viking Axe Throwing	60
Giant Digital Graffiti	60
Cricket 2020	60
Digital Tresure Hunt	60
Monti Carlo or Bust	60
Archery	45
2-wheeled Segways	45
Enigma Challenge	45
Indiano Drizzle	45
Learning Magic Tricks	40
Human Table Football	30
Buggy Driving	30
Giant Puzzle Dinosaurs	30
Arduino Bonanza	30
New Zealand Haka	30
Salsa & Pickles	15
Wine Tasting	15
Time Tracker	15

   (4) If there are more teams to be iterate through remaining activities that are not in the cache
      (a) If adding duration between activities equals target time, add activity to the cache and to the shortlisted pre/post activity block
      (b) If minimum day length is taken into account (like for post lunch sessions), then check if the total time of activities selected, is greater than or equal

to the minimum target time and less than the target time. Ex: The speech can begin anytime after 4 PM and 5 PM

(c) If however, adding the time exceeds target time, then that activity should be ignored and the next item in the list should be analysed.

Monti Carlo or Bust	60
Archery	45
2-wheeled Segways	45
Enigma Challenge	45
Indiano Drizzle	45
Learning Magic Tricks	40
Human Table Football	30
Buggy Driving	30
Giant Puzzle Dinosaurs	30
Arduino Bonanza	30
New Zealand Haka	30
Salsa & Pickles	15
Wine Tasting	15
Time Tracker	15

Example, 3 activities are skipped after 2-wheeled Segways and ‘Human Table Football’ is seleted for pre-lunch block as it adds up to 180 mins

(5) Repeat the above process till the activities per team is formed

<u>Form the timetables and print them– *DigitalDayScheduler Class*</u>

1. Ensure to add ‘Lunch’ between the pre-lunch and post-lunch sessions
2. Ensure to add ‘End of day mandatory activity’ after post-lunch session
3. Iterate through the list and form start time (LocalTime in hh:mm a format)
4. Print the list for all the available teams