Algorithm design

In the following paragraph are presented the most relevant and significant algorithm used in Travlendar+ application.

More specifically are described the following algorithm:

* Creation of new appointment;
* View of daily schedule;
* Check appointments overlapping;
* Check appointments unreachability;

The application is written in Java.

Before to explain the algorithms details, it is needed to introduce the class Appointment and Travel that are the most important entities for the whole application.

The class Appointment has as attributes:

* The appointment name;
* The date;
* The time when it starts;
* Address of departure;
* Address of destination;
* The desired time to leave;
* The appointment duration;
* The expected time of arrival;
* The associated travel to reach the appointment;



The travel class has the following attributes:

* The starting point;
* The destination point;
* The travel mode;
* The estimated speed;
* The distance between starting point and destination;
* The estimated time;
* The weather condition;
* The movements list in which the travel is split;

Before to show the related java code it is necessary to make some observations:

* One travel is split in some movements, each composed by starting point and destination point, the distance between these two points, speed related to travel mode and therefor the computed travel time.
* The “travel mode” is the travel way, possible ways are drive, walk, bike, transit that means public transport means.
* The “estimated speed” is computed starting from the travel mode, each mode has a specific related speed.
* Form “travel mode” and “estimated speed”, it is possible to compute the estimated time. Before to perform the computation, it is necessary to convert the distance from kilometers in meters.
* Weather is another java class that contains all information that regard the position, the time and the weather condition. The last information is fetched from external APIs.



# Create new appointment

This algorithm allows the user to create a new appointment.

In this algorithm is used one method called readText, this is a method created to read from EditText and has as parameters only the id of EditText.



# View of daily schedule

When the user clicks on view daily schedule button, the system computes the daily schedule through this method and show to the user all appointments expected for the selected day and all travel to reach them.

This method has two parameters, the first is the instance of the calendar and the second in the desired day.

The first thing performed is the link to user calendar thus to extract all appointment for the inserted day.

For each appointment it is necessary to re-compute the related travel, therefore is possible to have updated information.

But it is necessary to control, for every appointment, the unreachability and the overlapping with the next one.

The algorithm starts from the first appointment and control its reachability, if it is then fetch the next one appointment and control its reachability, if it is also reachable then it is possible to control the overlapping between them.

If the second appointment is not reachable (and it is deleted) then the algorithm fetches the next one and control its reachability. This cycle goes on until “the next appointment” is reachable and it is possible to control the overlapping.

If overlap control is positive (return true) means that the first appointment overlaps with the second and user must choose which is the appointment to keep.

At the end of all controls, the system computes the travel for the first appointment.

Then repeat all these steps for each appointment expected for the selected day.



# Check overlap

When the system has to check the overlapping between two appointments invokes this method that has as parameter the two appointments.

The method controls which appointment starts before and then controls if the begin time of the second appointment overlaps with end time of the first appointment.

The method returns false if there is not overlap and true otherwise.



# Check unreachability

This method checks the unreachability of an appointment passed as parameter.

To check the unreachability it is necessary to control if the arrival time of one appointment is compatible with departure time added to estimated travel time.

The method returns false if the appointment is not reachable, true otherwise.

