System design document

for Timeline Android Application

Version: 1.0
Date: 25/05/2017
Author: Melina, Hannes, Carlos and Nikolai

This version overrides all previous versions.

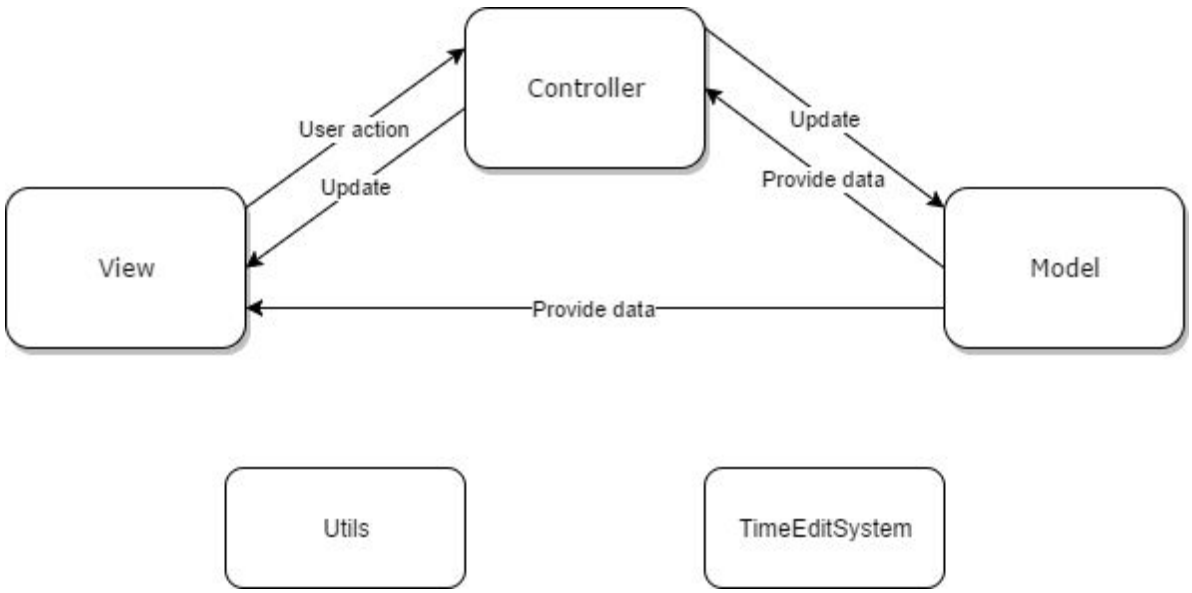
1 Introduction

This is a system design document which is aiming to provide information about our application and its components. The goal is to explain the system and its architecture enough for a developer to be able to extend our project.

1.1 Definitions, acronyms and abbreviation

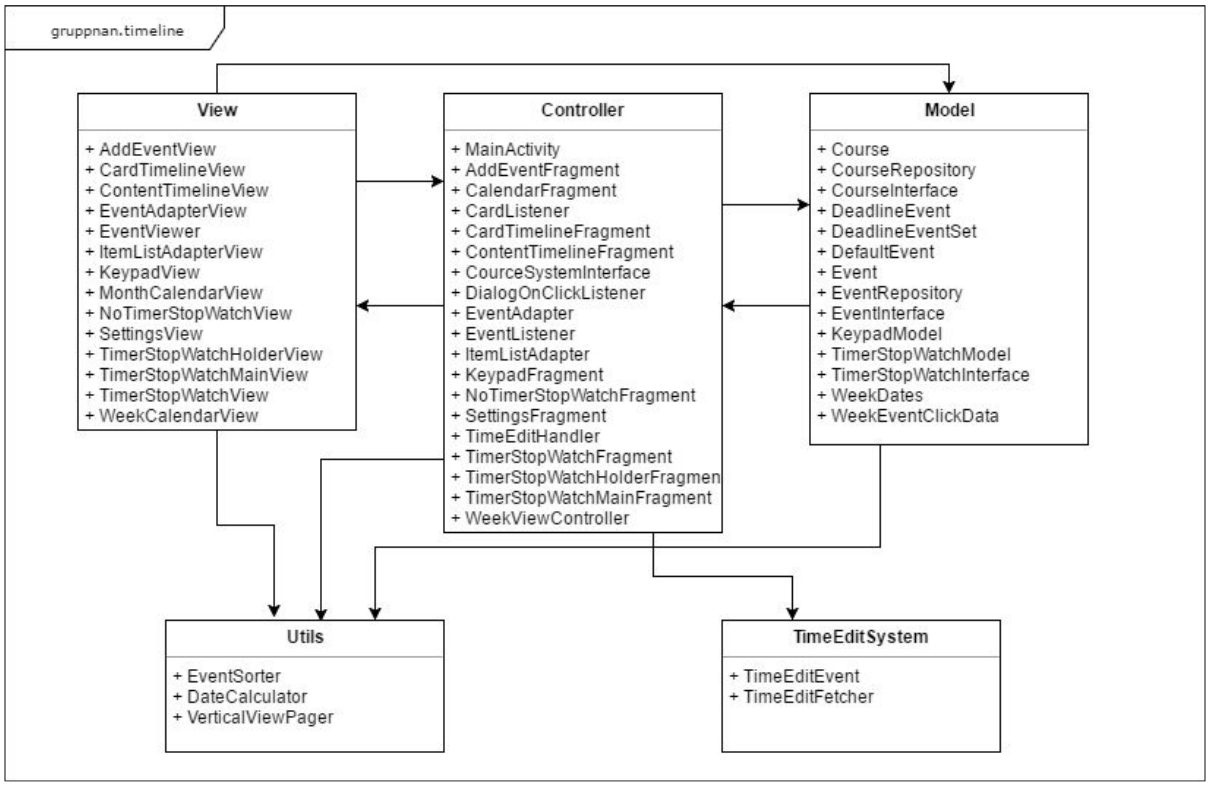
- MVC, Model-View-Controller
- TimeEdit, a schedule handler used by Chalmers.
- STAN, a program for showing dependencies between packages and classes
- FindBugs, a tool for identifying common errors with code design

2 System architecture



We only have one software and its content is designed as above.
You start the system by pressing the launch icon at the android device. Stop the system by closing the app.

3. Subsystem decomposition



3.1 Design

The architecture is constructed according to MVC-pattern. The model holds the data, the view updates the graphical user interface and the controller handles all actions for the view by transferring data between the model and the view.

Since the android platform is built on activities and fragments which serves the function of both view and controller, we have extracted view classes from controllers in order to follow MVC.

The utils package contains utilities for:

- Sorting events to be displayed on timeline correctly
- Calculating weekdays from calendar objects
- Scrolling through a viewpager vertically

TimeEditSystem has classes that fetches and handles data from TimeEdit.

We structured it this way because we wanted the model to only contain classes and methods to store, remove, set and get data, the view to only handle graphical elements, and the controller to handle actions. Hence, the help classes that now are in utils were better placed there than in any of the MVC-packages according to us.

The communication between view and controller is solved by setting the associated controller as listener for the elements in the view. By using the Observer design pattern we lower the coupling between the classes.

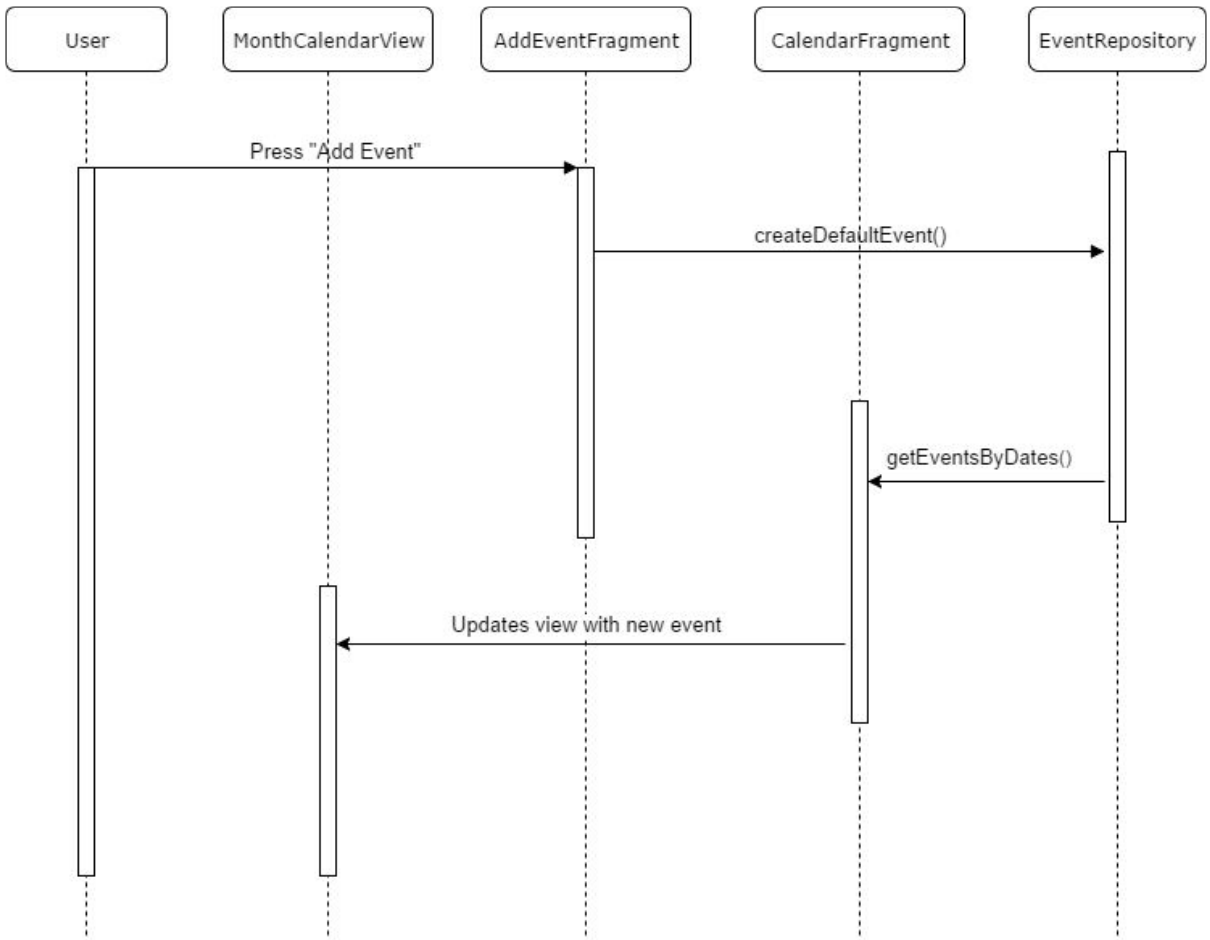
We have also used the Singleton design pattern for our EventRepository and CourseRepository. This is because we want to ensure that only one class that stores events and one class that stores courses could be created.

For those classes named above we also use the facade pattern, it is here we have the methods for creating, removing and getting events and courses. We use facade to simplify the handling of events and courses from the other subsystems.

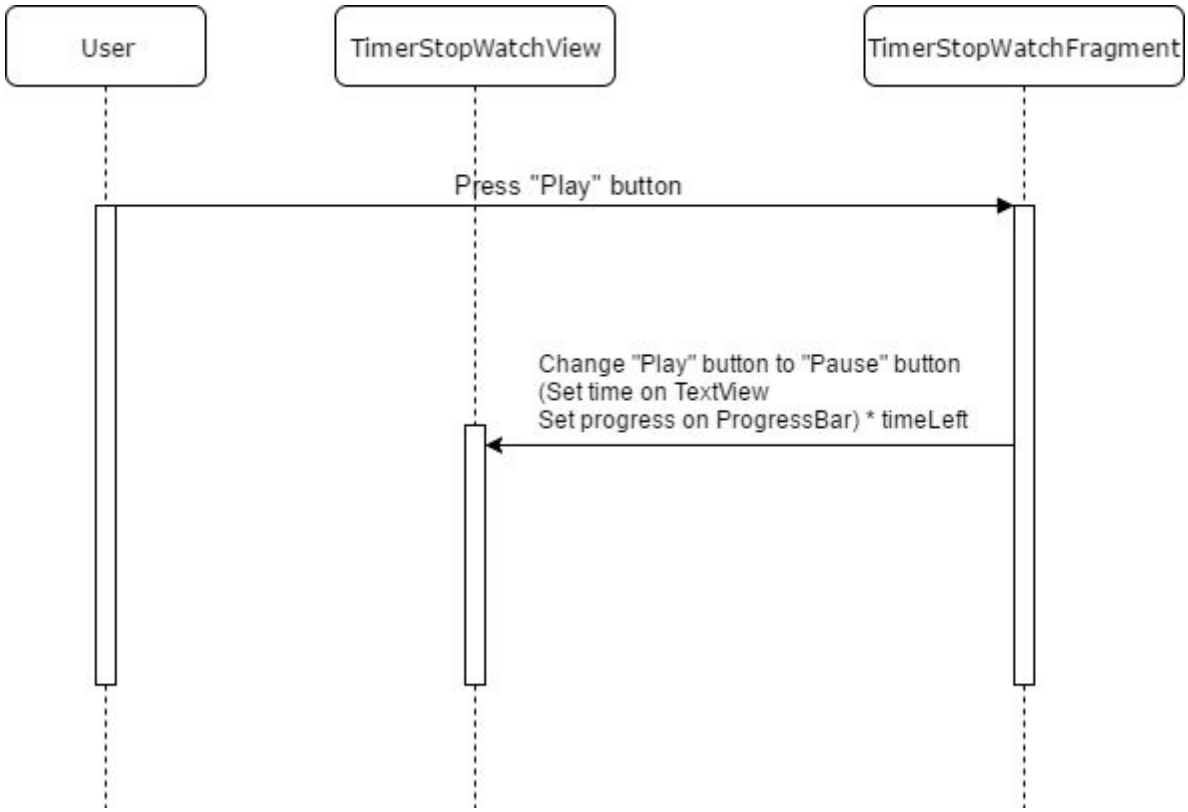
3.1 Diagrams

Sequence diagrams

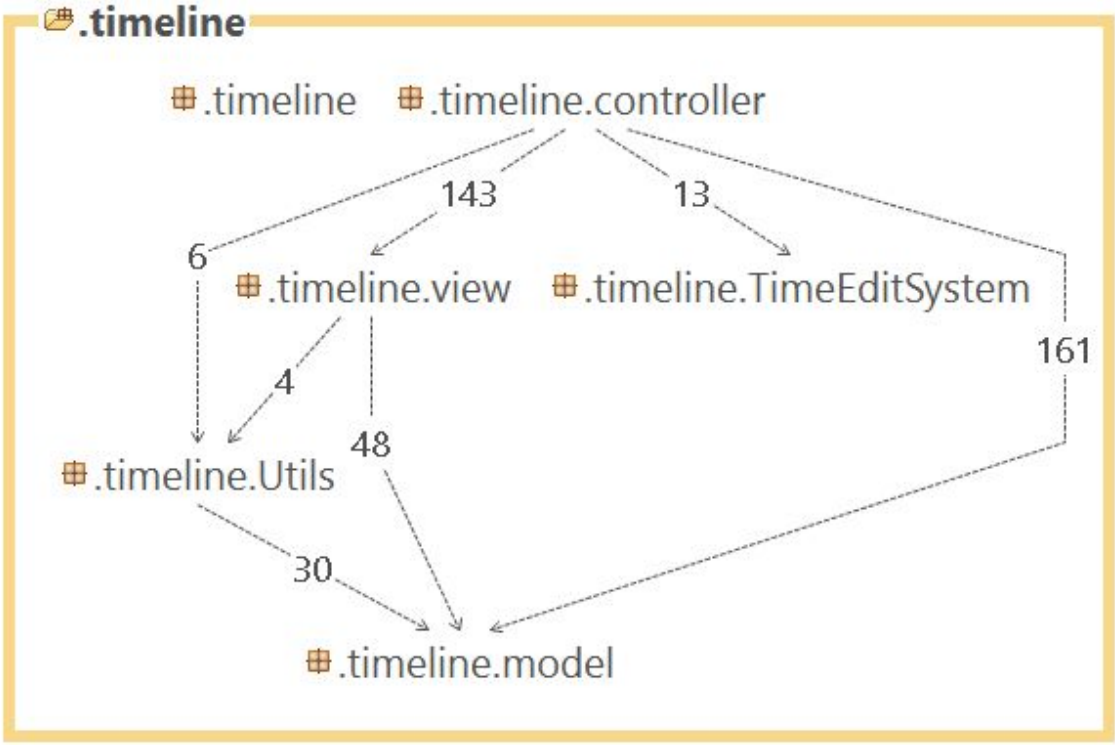
Sequence for adding an event after entering information in event form

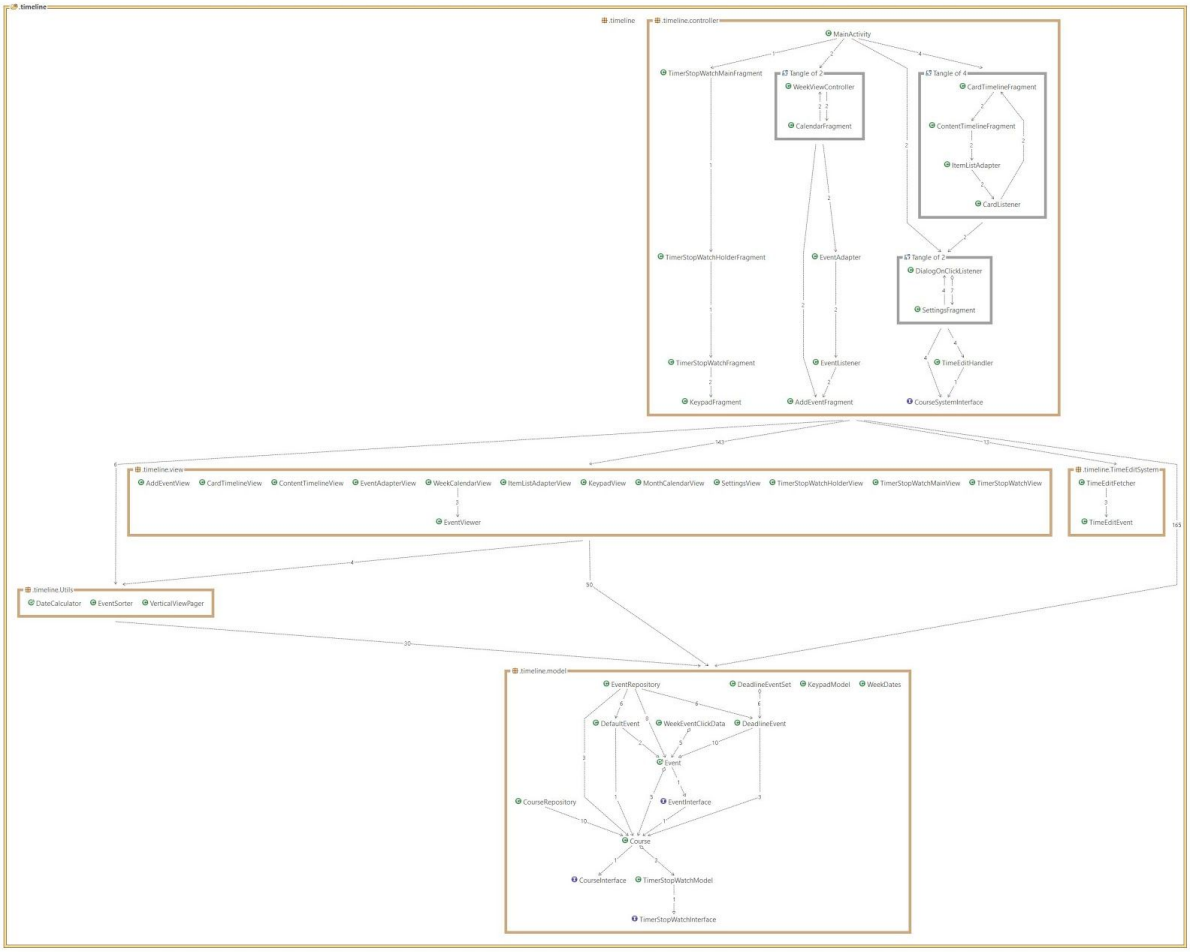


Sequence for starting a timer after initiating it in settings


User
TimerStopWatchView
TimerStopWatchFragment
Press "Play" button
Change "Play" button to "Pause" button
(Set time on TextView
Set progress on ProgressBar) * timeLeft

Dependencies - STAN



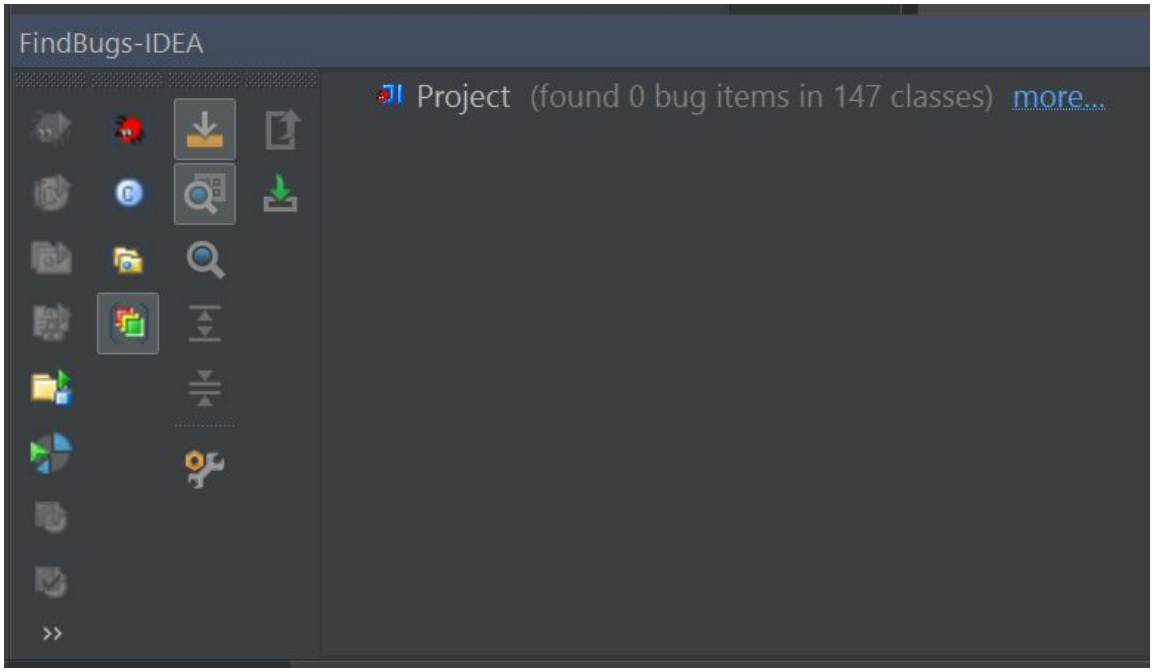


The image can be viewed closer at http://imgur.com/GPQcG1Q

3.3 Quality

Tests are placed in the “src” folder, the full link is:
https://github.com/nik003/tda367/tree/master/Project/app/src/test/java/gruppnan/timeline

We didn’t know how to make a file out of the report from findbugs but when we had no bugs. Here is a print screen:



Known issues:

- The data is not stored after app is closed
- The courses can not be deleted manually
- Sometimes the built in back-button may behave undesirably
- Changes to add/change/delete events shown in the list view is first shown when re-selecting date in month calendar view.
- Using the phone’s back button from week calendar view will sometimes result in a completely white view.
- The system does not warn you if you try to add multiple event on same time
- The keypad view does not display when invoked
- The stopwatch and timers stop running when changing view in the drawer
- Only one course and its corresponding stopwatch and timers are shown at a time in the viewpager
- The app can only import events from two weeks in the past to three months in the future
- The columns of week view are not of equal width

4. Access control and security

The app requires the user to give permission for the app to use internet. The app uses an internet connection to retrieve data from TimeEdit.