Mobile Application Development
COSC2309/2347 Semester 1, 2019
Movie Night Planner

Assignment 1 (20 marks)

You are to implement a simple Movie Night Planner app to create and schedule movie viewing events and invite attendees. In this first assignment, you will implement the basic user interface for event creation (including movie details), editing, and calendar viewing. This will be extended in Assignment 2 to provide a full application complete with persistent storage and networking/cloud service behaviour (including a location-based alarm that will trigger based on your distance from your current location to that of the movie event).

At the core of the functional requirements of this application are the following two entities:

Event: An *event* is a social movie viewing activity that occurs at a specific date, time and location and has a number of attendees. It must (at a minimum) maintain the following information:

- **Id**: A randomly generated combination of numbers and letters, which uniquely identifies an event (not visible to the user).
- **Title**: The title of the event (e.g. “Scary Saturday!”)
- **Start Date:** The start date and time of an Event (e.g. April 6th, 2019, 7.30PM)
- **End Date:** The end date and time of an Event (must be later than start date)
- **Venue:** The location of the event (e.g. “MegaDupaMultiPlex, 13 Node.js”)
- **Location:** A String representation of geographical latitude and longitude of the venue (e.g. -37.805631, 144.963053), you can get this from Google Maps web page directly (i.e. outside of your app by just copying and pasting the data).
- **Attendees**: A list of individuals who are invited to this event (as picked from the user’s contacts list .. see supplied `contacts_data/`).

Movie: Each event has associated details of the movie to be viewed/screened as follows:

- **Id**: A randomly generated combination of numbers and letters, which uniquely identifies a movie (not visible to the user)..
- **Title**: The title of the movie (e.g. “Blade Runner”)

- **Year**: the year the movie was released (e.g. 1982)
- **Poster**: an image of the poster used to promote the movie.
  NOTE: use of copyrighted material for education purposes in your assignment is covered under fair use i.e. https://www.alrc.gov.au/publications/4-case-fair-use-australia/what-fair-use but should not be redistributed outside the educational setting.

Functional Requirements
Your application must provide the following functionalities and meet the non-functional requirements stated under the "Other Requirements" section below.

- **Schedule and Unschedule an Event:** The application should allow the creation of an unbounded set of events. For simplicity, you can ignore duplicate entries or events with overlapping times (we will make sure test data does not overlap when assessing your assignment).
- **Edit Event Details:** Users should be able to edit the event details as well as add or remove attendees.
- **Add/Edit Movie details for an event:** This should be implemented in a separate screen or Activity from the Event scheduling/editing described in the previous two points.
- **View Events:** Users should be able to display a **list** of events where each element in the list will be a synoptic view (summary) of the Event e.g. event title, date, venue, movie title and number of attendees. The list should be sorted according to date (user can toggle ascending/descending order).
- **View Calendar:** Users should be able to view the entries in a calendar style layout, based on either* a week or month view (you can look at the calendar app on an Android device or emulator for ideas but are free to be creative with your layout). **IMPORTANT NOTE:** You must create your own UI using standard layouts i.e. you cannot just use a standard or third party Calendar widget.
- **Selection/Editing:** In the view modes described above, i.e. when events are shown in a list or in a Calendar, users should be able to add or edit items via direct manipulation (e.g. long press or gesture).

* You only have to choose one or the other but if you are having fun with layouts you are welcome to create both so that you have a total of three different views!

Other Requirements:

- In assignment part 1 you are not expected to persist data however your data must not be limited to the lifetime of any specific activity (i.e. must have application scope to facilitate testing). You should read data from the supplied *events.txt* and *movies.txt* when your app is first loaded.

- Your Graphical User Interface (GUI) must support all of the functionalities presented under “functional requirements” above.
- The preferred Target Android Version is API Level 25, however, for students with old devices who want to target lower levels you are allowed to do so.
- You can write your application in the single Activity per screen phone style however you may choose to write a tablet version using Fragments.
- You should use the **Model View Controller (MVC)** approach to assist in writing modular and cohesive code. In particular we will be looking for a **domain neutral model** implementation which is not dependent upon Android specific features (i.e. `java.*` class/package dependencies only not `android.*`). Some examples of class/interface usage in your model would be: `Event(interface)<-AbstractEvent<-EventImpl` and `Movie(interface)<-AbstractMovie<-MovieImpl`.
- For your UI you should aim for simplicity. You will not be marked on the aesthetics of your design (beyond common sense!) but we are looking for clear workflows. The only constraints on the design and classes used are those explicitly stated above, otherwise you are free to be creative (menus, buttons, toolbars, navigation drawers etc.)
- Your implementation must make efficient use of common values (such as Strings, Dimensions or Colors) by (re)using values from the appropriate XML resource file. i.e. do not hardcode such values into your layout files or your application.
- You may want to consider branding, business models, distribution frameworks, 3rd party integration (maps, navigation, social media sites) etc. and how these would impact on your app design although you will not be assessed on these features (some of these will be covered in assignment part 2).

Code Quality

Since this is an advanced programming elective you will be marked on code quality as well as functional correctness (approximately 60/40 split of functionality versus quality respectively). A marking rubric is available on Canvas for further details.

In particular:

1. You should aim to provide high cohesion and low coupling.
2. You should aim for maximum encapsulation and information hiding.
3. You should rigorously avoid code duplication.
4. You should comment important sections of your code remembering that clear and readily comprehensible code is preferable to a comment.
5. If you use lambdas you should carefully consider the impact on coupling, cohesion and comprehensibility. In many/most cases inheritance, polymorphism and delegation are better approaches when used well.

6. Where appropriate, MVC controller functionality should be placed in separate classes in a separate controller package. A good rule of thumb is if it is more than a single method call, or there are non-local variable dependencies, then put it in a separate class!

Submission Instructions
Your project should be implemented using Android Studio and your project exported as a single compressed .zip archive before submission. **Do not** use any other compression formats - use of other formats (e.g. tar.gz, RAR, etc.) may lead to delays in marking and/or a deduction of assignment marks.

Important Regulations
- You are free to refer to textbooks and notes, and discuss design issues (and associated general solutions) with your fellow students and on Canvas; however, **the assignment should be your own individual work (or optionally a pair of students, see details below).**
- Note that you will only be assessed on your own work so the use of third party designs and APIs (beyond the standard Android libraries) is not allowed.

Optionally forming pairs:
This assignment is of a size and scope that can be done as an individual assignment. However, since we understand that some students benefit from working in pairs we will allow optional pairs to be formed. We will however strictly adhere to the following procedure, keeping in mind that group work is NOT a core learning outcome of this course and is only offered as an extra/bonus option.

- This assignment can be done in pairs or individually. If done individually there is no marking advantage. If done as a pair, students are NOT marked individually, only as a pair (No ifs, no buts .. so when choosing a partner it is *Caveat Emptor* i.e. Buyer Beware!).
- If students choose to be a pair they must register **together** in their tutelab class in week 2 with their student ids (NOTE: Student pairs are encouraged to be in the same scheduled tutelab but this is not strictly required other than initial signup).
- If students are unable to attend together **due to timetabling issues** students must promptly confirm their pairing via email to their tutor.
- If you fail to inform your tutor that you are working as a pair you must submit individually and not share any source code.
- Students cannot change pairs but pairs can separate and both students become individuals if problems arise with the pairing (both students being able to use any shared code that has been created to this point). In this case your tutor must be notified at the latest by Friday 5th April 2019 (end of week 5).
- No changes will be allowed after this time unless special consideration applies.

NOTE: The required implementation language for this assignment is **Java** since it is stable, widely used and most importantly students can leverage their experience from the pre-requisite courses. Moreover, Android is challenging enough (by design) without having to learn a new language at the same time. However, if students have a compelling reason to use Kotlin and can demonstrate a track record of high competency and independent learning they can make their case to their tutor in the tutelab (email applications are not acceptable since this is a special arrangement) and your tutor may use their discretion and make a decision to allow this.

This assignment is due at 6:00PM Thur. 18th April 2019 and is worth 20% of your final assessment