**Submission for Assignment 08**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Task 1**

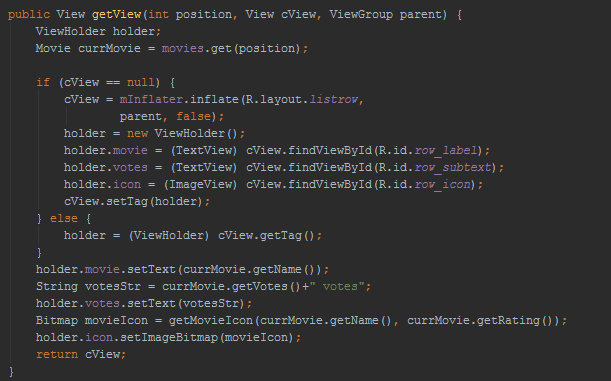
**Introduction**

This report will contain the two major problems of the MovieRatings application. There was one optimization problem and one usability problem. Optimization problems are when we overload the memory with multiple tasks, and create many unnecessary objects. The other problem was an improvement upon the usability (Though it is also arguably an optimization problem). When an application is required to do many tasks, it needs to wait for each task to complete before it can move on to the next. The solution to this is to use asynchronous events that allow multiple events to occur at once. For the user, this means that there will be visual cues that indicate if something is trying to load and prevents them from continuing to use the application until it is loaded.

**Performance Optimisation**

The first problem was the Optimisation performance. When the user would scroll down the Movie list, it was laggy and jittery, feeling like the application may freeze at any moment. This is caused when the device is required to constantly load new objects. If the platform cannot see the item on the screen, it will destroy the object, and recreate it later if it needs to be seen again. When a list is small it does not have such an impact on the performance. However, long lists create unnecessary commands for the device to do, and hogs the memory of the application thus making the rendering more sluggish. In order to make this smoother, we can store the views that have already been populated in a temporary "holder" variable to be called later. This is referred to as Recycling views.

Solution -

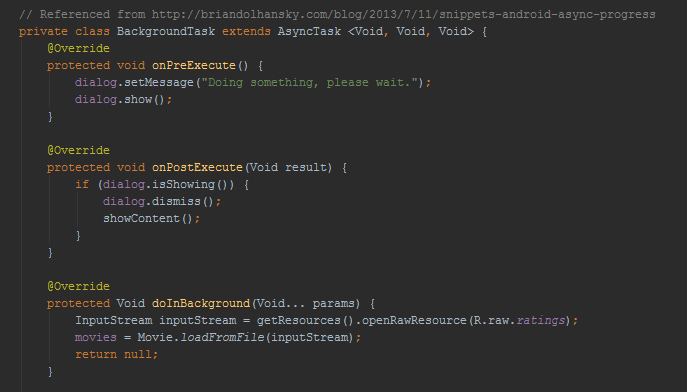


**Performance Optimisation**

In opening the MovieRatings application, we are presented with a black blank screen for an extended period of time. This is happens when an application is initially opened, all the commands are put into an event queue. If one of the commands requests a file that is very large to be downloaded, no other command will be executed until this download is complete. In the case of the MovieRatings application, the CSV file that is being downloaded is very large and takes awhile to download. To rectify this problem, we can use an asynchronous event showing a ProgressDialog that will tell the user that the content is loading and that they need to wait before proceeding with giving more commands to the application.

To achieve this, we create an asynchronous task that runs in the background. (doInBackground()). While the download is happening, a dialog message is sent to the screen telling the user to wait (dialog.setMessage("Doing something, please wait.")). Once the download is complete, the dialog message will close (dialog.dismiss())

Solution-

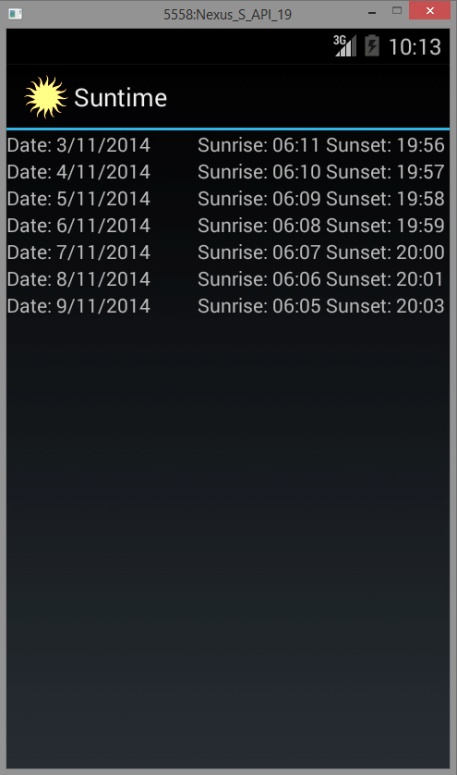
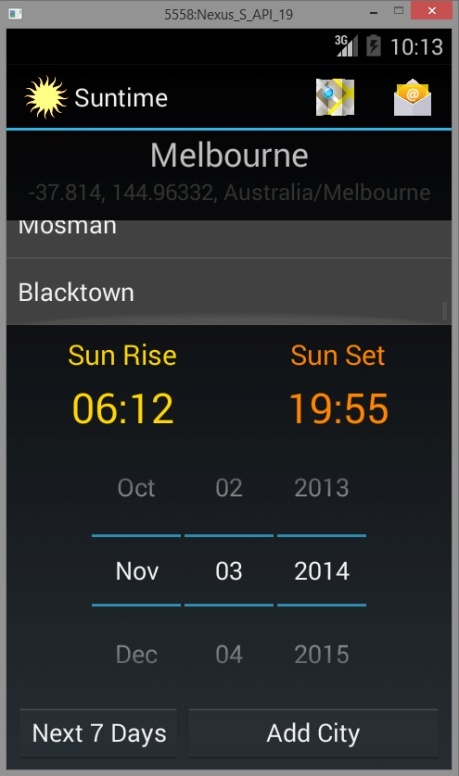


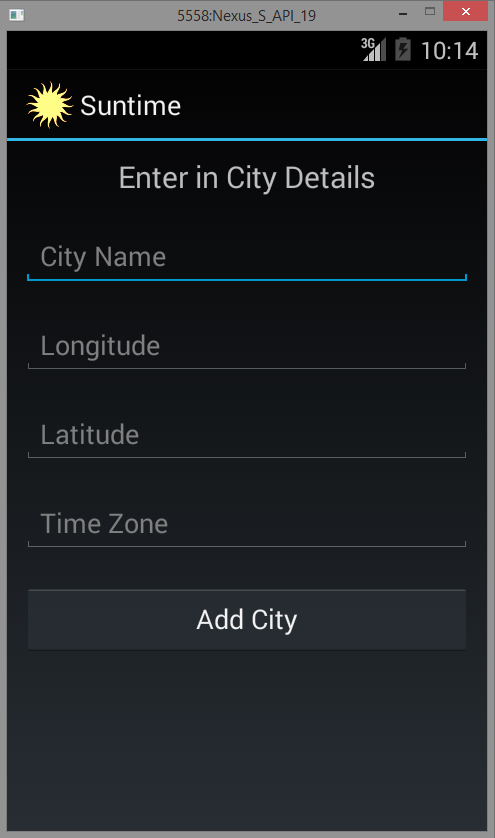
**Task 2**

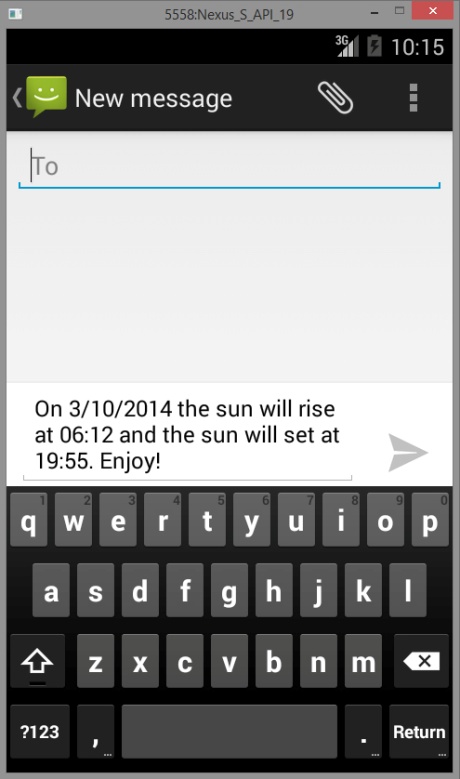
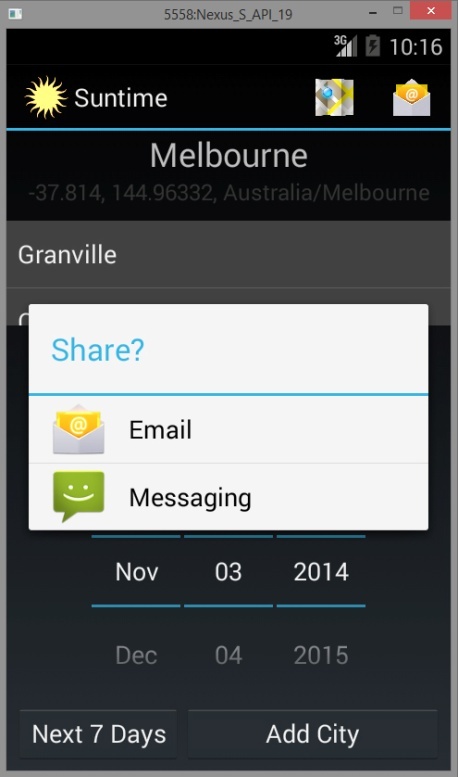
Fragments used:

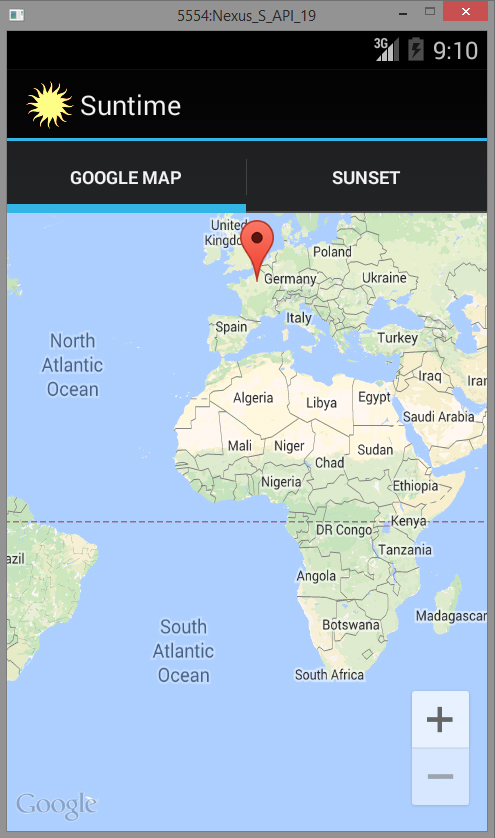
* sunsetFragment.java
* mapFragment.java

Main Activity Table of Sunrise/set Add City Activity

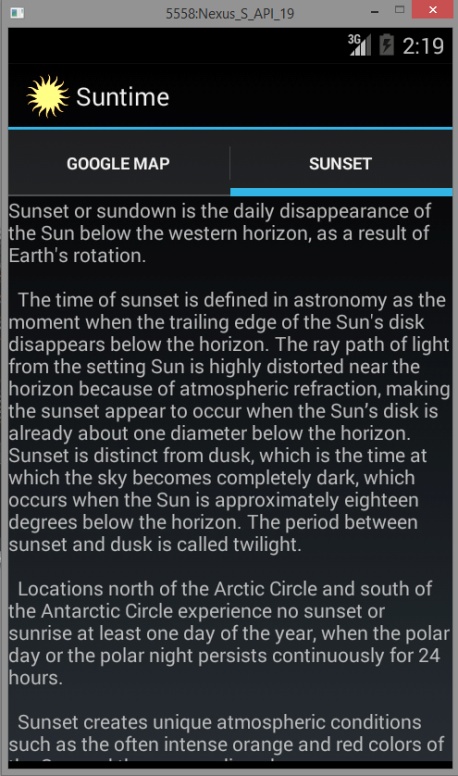
  
  
Maps Fragment Share Menu Share Intent



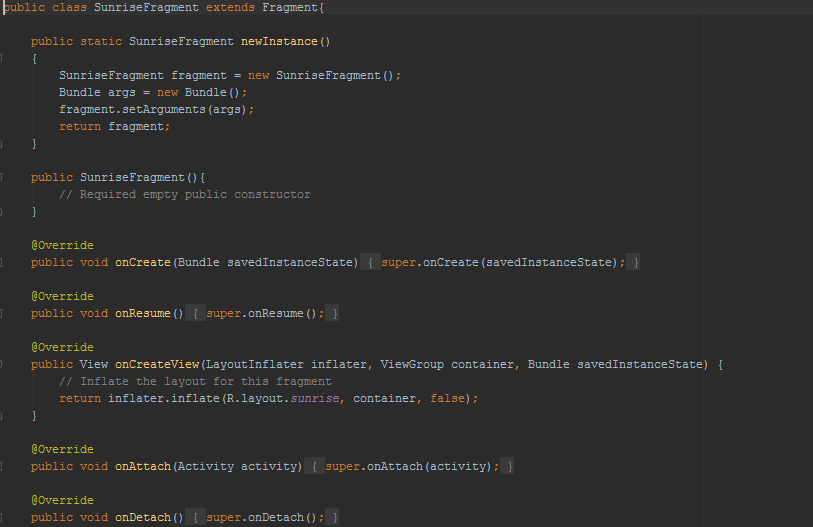




Sunset Fragment



Fragment Code-



Main Activity calling Fragments -

