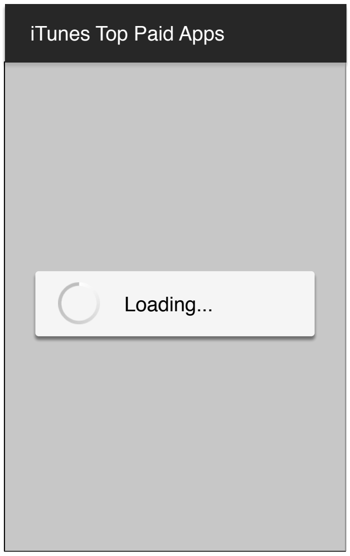
ITIS/ITCS 4180/5180 Mobile Application Development
Midterm

Basic Instructions:

1. This is the Midterm Exam, which will count for 20% of the total course grade.
2. In every file submitted you MUST place the following comments:
    a. Your Full Name.
3. This Midterm is an individual effort. Each student is responsible for her/his own Midterm and its submission.
4. Once you have picked up the exam, you may not discuss it in any way with anyone until the exam period is over.
5. During the exam, you are allowed to use the course videos, slides, and your code from previous homeworks and in class assignments. You are NOT allowed to use code provided by other students or from other sources.
6. Answer all the exam parts, all the parts are required.
7. Please download the support files provided with the Midterm and use them when implementing your project.
8. Your assignment will be graded for functional requirements and efficiency of your submitted solution. You will loose points if your code is not efficient, does unnecessary processing or blocks the UI thread.
9. Export your Android project and create a zip file which includes all the project folder and any required libraries. The file name is very important and should follow the following format: **Lastname_Midterm.zip.** Submit the exported file using the provided canvas submission link.
10. **Failure to do the above instructions will result in loss of points.**
11. **Any violation of the rules regarding consultation with others will not be tolerated and will result disciplinary action and failing the course.**

Midterm (100 points)

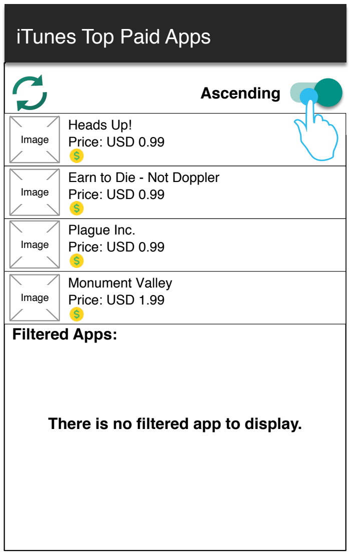
In this assignment you will develop “iTunes Top Paid Apps” app, which displays the list of top paid apps from “iTunes Top Paid Apps” rss feed. In addition to displaying the top paid apps, the app allows the user to filter specific apps and store them in a SQLite database.



iTunes Top Paid Apps

Loading...

(a) Loading Dialog.



iTunes Top Paid Apps

Ascending

Heads Up!
Price: USD 0.99
$

Earn to Die - Not Doppler
Price: USD 0.99
$

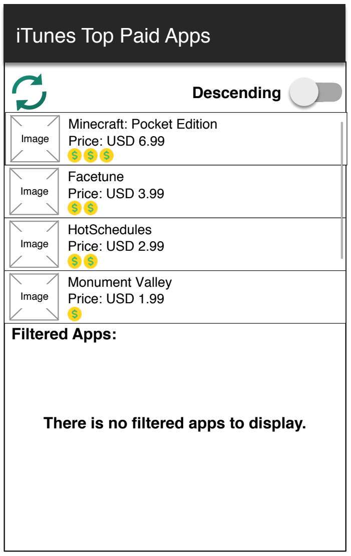
Plague Inc.
Price: USD 0.99
$

Monument Valley
Price: USD 1.99
$

Filtered Apps:

There is no filtered app to display.

(b) Sorting, default is Ascending.



iTunes Top Paid Apps

Descending

Minecraft: Pocket Edition
Price: USD 6.99
$$$

Facetune
Price: USD 3.99
$$

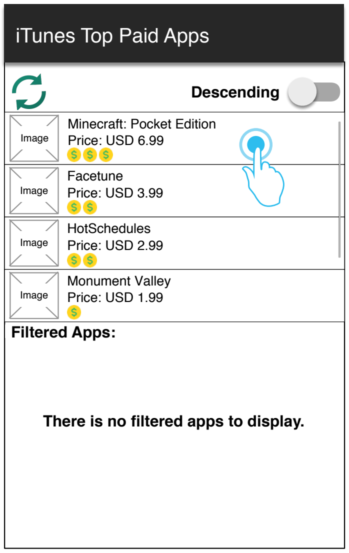
HotSchedules
Price: USD 2.99
$$

Monument Valley
Price: USD 1.99
$

Filtered Apps:

There is no filtered apps to display.

(c) ListView sorted descending



iTunes Top Paid Apps

Descending

Minecraft: Pocket Edition
Price: USD 6.99
$$$

Facetune
Price: USD 3.99
$$

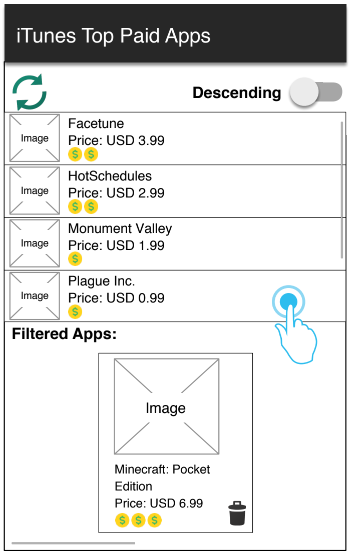
HotSchedules
Price: USD 2.99
$$

Monument Valley
Price: USD 1.99
$

Filtered Apps:

There is no filtered apps to display.

(d) Long Click on item



iTunes Top Paid Apps

Descending

Facetune
Price: USD 3.99
$$

HotSchedules
Price: USD 2.99
$$

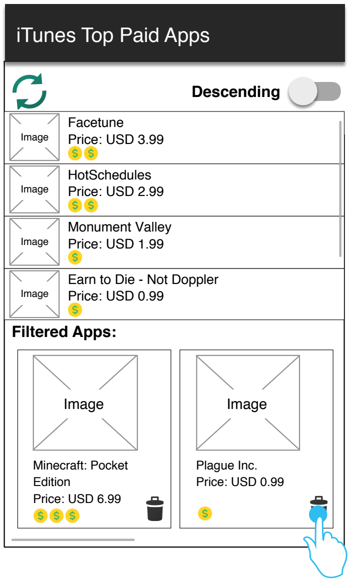
Monument Valley
Price: USD 1.99
$

Plague Inc.
Price: USD 0.99
$

Filtered Apps:

Minecraft: Pocket Edition
Price: USD 6.99
$$$

(e) Added to Filtered Apps list.



iTunes Top Paid Apps

Descending

Facetune
Price: USD 3.99
$$

HotSchedules
Price: USD 2.99
$$

Monument Valley
Price: USD 1.99
$

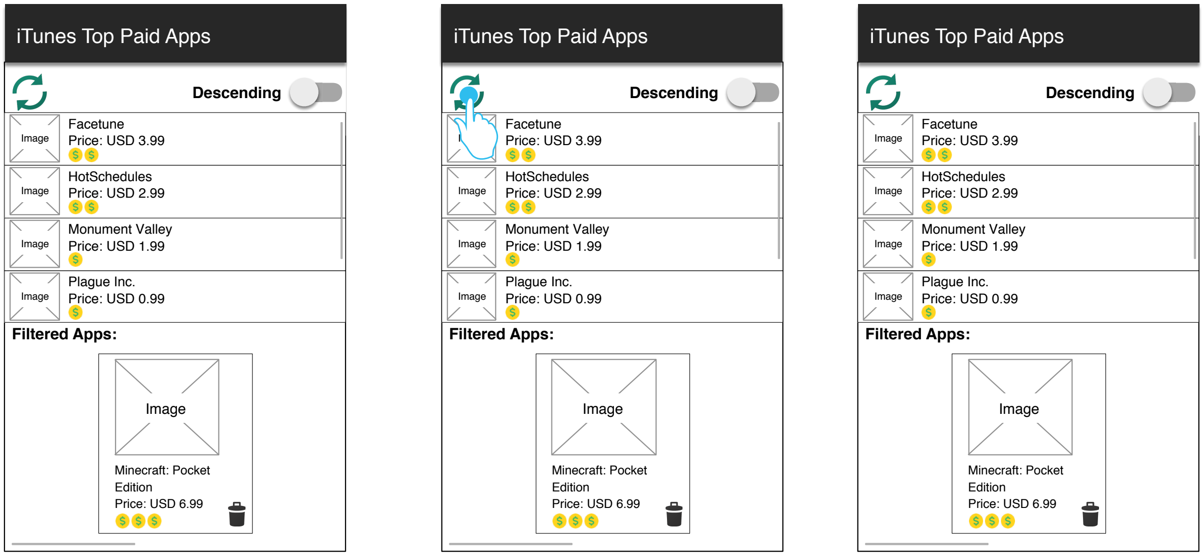
Earn to Die - Not Doppler
Price: USD 0.99
$

Filtered Apps:

Minecraft: Pocket Edition
Price: USD 6.99
$$$

Plague Inc.
Price: USD 0.99
$

(f) Click on Trash icon.



(g) Item deleted from filtered list. (h) Click on Refresh button. (i) Refreshed and Filtered list.

Figure 1, App Wireframe

Main Activity (100 points):

This assignment has only one activity, MainActivity (Figure 1). This activity contains a top ListView of apps and a bottom RecyclerView of filtered apps. The API to be used for this assignment is: **https://itunes.apple.com/us/rss/toppaidapplications/limit=25/json**.
The top ListView is a list of apps retrieved from the API and should not include any of the apps that have been filtered by the user. The bottom RecyclerViewer displays the apps that have been filtered by the user. The requirements are as follows:

Part 1:

12. When the app starts, it should call the API to retrieve the list of apps, parse the retrieved items and display the top ListView and bottom RecyclerView. While the API is being accessed display a progress dialog as shown in Figure 1(a). Use a thread pool or AsyncTask to access the API and parse the result using JSON parser.
13. On top of the activity there is a refresh button, and a sorting Switch. By default the Switch should be set to **ascending** state, which is sorting the list in ascending order by the app price. Note that the Switch label indicates “ascending” to reflect the state of the Switch. The other state of the sorting switch is descending, which should sort the list in descending order. The label of the Switch should be changed to reflect the sorting state.
14. When the sorting Switch is toggled the list should be refreshed to reflect the updated sorting state.
15. The retrieved items should be displayed in top ListView as shown in Figure 1(b). The top list should be created using a customized ListView. Each item should contain the app thumbnail, app name, app price and the dollar image according to the app price. The app price images should be displayed according to the following:
    i) One dollar icon, for apps with prices from $0 to $1.99.
    ii) Two dollar icon, for apps with prices from $2.00 to $5.99.
    iii) Three dollar icon, for apps with prices greater than or equal to $6.00.

16. In the top list, the apps should be displayed in sorted order based on the sorting order set by the sorting Switch at the top of the activity.
17. This app has a feature which allows the user to filter apps and store them in a SQLite database locally. The filtered apps **should not** be displayed in the top ListView. Create the SQLite database with table called **Filter** having columns**:** name, price, thumb_url. This table will be used to save the filtered apps.
18. When displaying the top ListView, the apps included in the Filter table **should not** be displayed in the top ListView.

Part 2:

1. The bottom RecyclerView should display the apps present in the Filter table, which are the apps that have been filtered by the user. When the Filter table is empty, display a message, "There is no filtered apps to display."
2. A long click on any item in the top ListView (Figure 1(d)), should:
    i) Add the selected item to the SQLite Filter table.
    ii) Remove the selected item from the top ListView, because it is now filtered. The top ListView should be refreshed to display only the items that are not filtered.
    iii) The bottom RecyclerView should be refreshed to display the updated list of items in the Filter table. See Figure 1(e).
3. The bottom RecyclerView should be displayed as a horizontal RecyclerView as shown in Figure 1(e). The bottom RecyclerView should show all the apps present in the Filter table.
4. Each item in the bottom RecyclerView should include the app thumbnail, name, price and the dollar sign icon. A trash icon should be displayed on the bottom right corner of each item as shown in Figure 1(e).
5. Clicking on the trash icon should perform the following operations:
    i) Delete the selected item from Filter table.
    ii) Refresh the upper ListView to display all the apps that retrieved from the API that are not present in the Filter table.
    iii) Refresh the bottom RecyclerView to display the apps in the Filter table.
6. The apps should be sorted based on the sorting order indicated by the Switch.
7. Clicking on the Refresh icon should perform the following tasks:
    i) Call the API to retrieve an updated list of apps.
    ii) Refresh the upper ListView to display all the apps that retrieved from the API that are not present in the Filter table.
    iii) Refresh the bottom RecyclerView to display the apps in the Filter table.
8. Use can use the Picasso library to load images.

Grading Key Overview

- 10 Points - API access and Parsing
- 10 Points - Sorting and switch event management
- 30 Points - Upper ListView (mainly for Display, Filtering, Events)
- 20 Points - Lower RecyclerView (mainly for Display, Events)
- 20 Points - SQLite (Setup, Insert, Delete, Queries)
- 10 Points - Event Management for the ListView and RecyclerView and the reloading.