MyCoupons

Proposal

The app is going to be developed with Java programming language.

The app keeps all strings in strings.xlm file and enables RTL layout switching on all layouts.

The app uses Content provider to store data locally and it uses a Loader to move the data to its views.

Libraries used		Version
Android Studio plugin		2.3.0
Build tools version		25.0.2
Gradle		4.4
Butterknife	`com.jakewharton:butterknife`	`8.5.1`
Play services ads	`com.google.android.gms:play-services-ads`	`10.2.1`
Play services drive	`com.google.android.gms:play-services-drive`	`10.2.1`
Google gson	`com.google.code.gson:gson`	`2.8.0`
RxAndroid	`io.reactivex.rxjava2:rxandroid`	`2.0.1`
RxJava	`io.reactivex.rxjava2:rxjava`	`2.0.1`
CardView	`com.android.support:cardview-v7`	`25.2.0`
Eventbus	`org.greenrobot:eventbus`	`3.0.0`
Design tools	`com.android.support:design`	`25.2.0`

Description

MyCoupons helps users keep track of the offers provided by different companies, it can be anything, from tech stores to groceries. Since there are many users like me who use their internet package fast 😊 I made the app work offline so that they won't need internet connection to view their saved coupons. The best with this app is that it makes it possible for you to enter manually coupon info so that it would be easier for you to keep track of them. In the future the app can be used to fetch online coupons/offers from different companies and help users get the products

they want for the best price. If user wants hic coupons to be saved so that he can have access to them even after changing his device, it is possible for him/her to back up saved coupons in Google drive.

Intended User

Intended user is not a special group of people, but I think it would be most useful for students and fathers and mothers, to help them get something they want for a reasonable price. It will help people that forget a lot to remember coupons that they may care about, and will help them get things cheaper, it can be clothes, food, electronics etc.

Features

Main features of the app are:
- Manually create and save different offers.
- User can see all saved coupons.
- User can see all past and upcoming coupons
- User can see how many coupons he used and how many are currently available
- If coupons expire date gets closer user will be able to get notifications
- Back up saved coupons to Drive
- Import same coupons to another device via Google drive
- Widget will make possible to access/browse available coupons
- User can share, edit or delete coupons.
- Set notifications for coupons.

User Interface Mocks
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

These makeups are screenshots of the app while I have been using it.

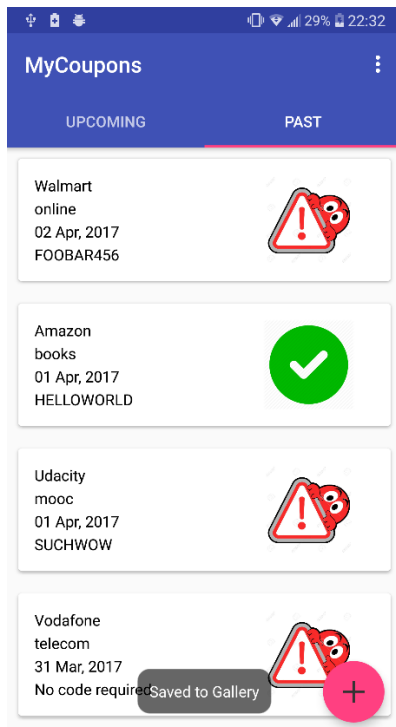**Screen 1 -Main Page where upcoming coupons are showed**
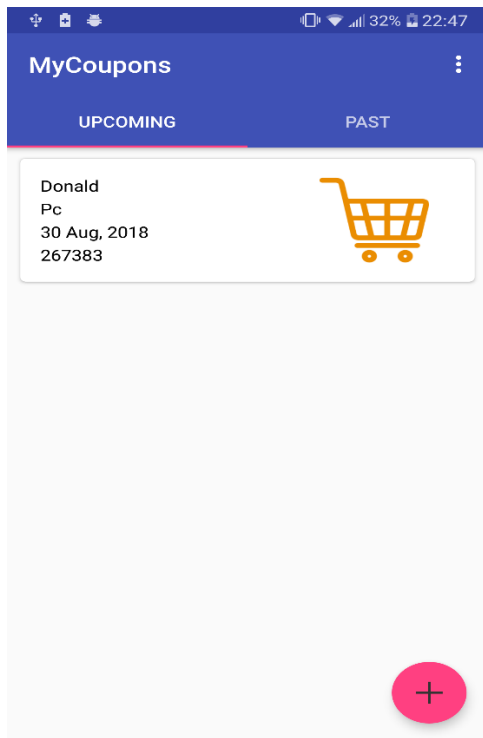
Here user can see upcoming coupons by pressing UPCOMMING button and PAST for the coupons that have expired.

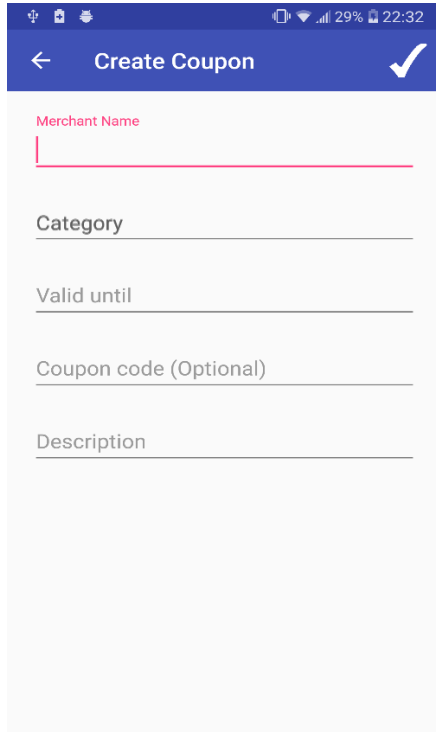Floating button + is used by user if he/she want to add another coupon.

Coupon list is sorted by date, which means that the closest coupons are showed at the top and the others follow down.

Screen 2 - Past Coupons

User can get to Pat Coupons page by pressing on PAST. Here the order of showing the coupons is by decreasing order starting from the one that expired closest to the date we are in to the one that is most away. For coupons that have not been used an Alert icon is shown.

Screen 3 – Create Coupon

Create coupon page lets user crate the coupon he/she found and is intended to use in the future. Here the user is able to save information about the company/merchant, category of the coupon, Valid date, coupon code (If applicable) and coupon description. This information is supposed to help user not forget important info of the coupon. After they have filled in information they will be able to save the info by clicking on the right top corner "tick" or done button and that saves the coupon and places it in the upcoming coupon list.
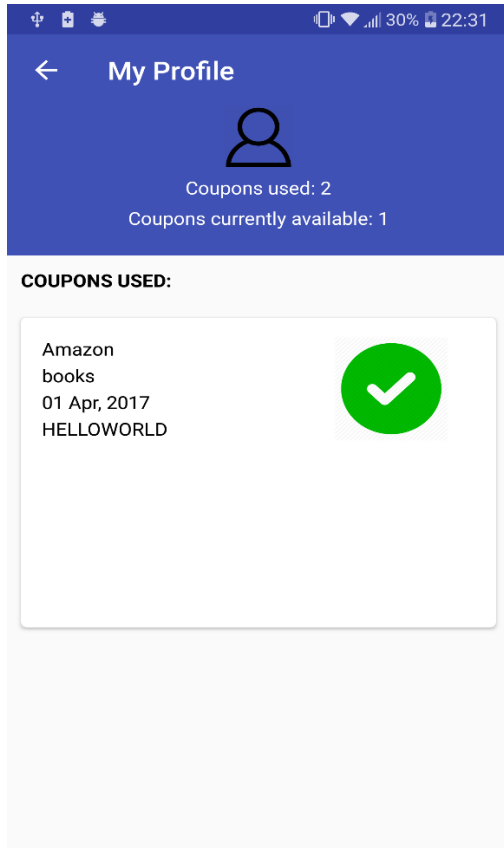
Screen 4 – Right Corner Overflow Menu
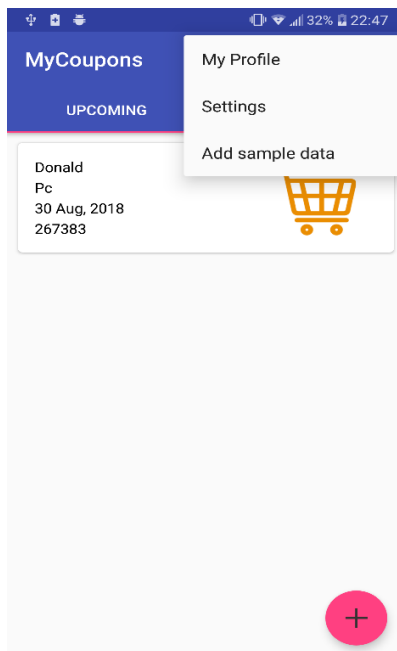
By clicking on the three dots on the right top corner user will be able to show the Overflow menu which will contain 3 main menus but just 2 of them are important:

1-My profile, which will send user to his/her profile

2- Setting which will display setting menu.

3- Add sample data is just for testing.

Screen 5 – My profile or user profile

This screen will show user his/her profile and user will get here by clicking on "Myprofile" from the menu shown on the previous step (step 4). In this screen user will see the number of used coupons and the number of coupons that are still available. From here user can navigate back by using the back arrow button on the left top corner.

Screen 6 - Layout of the coupons

The layout of the coupon is like this:

1-Name of the merchant/company (ex: Donald)

2-Category (Clothing)

3- Valid until date (30 august 2018)
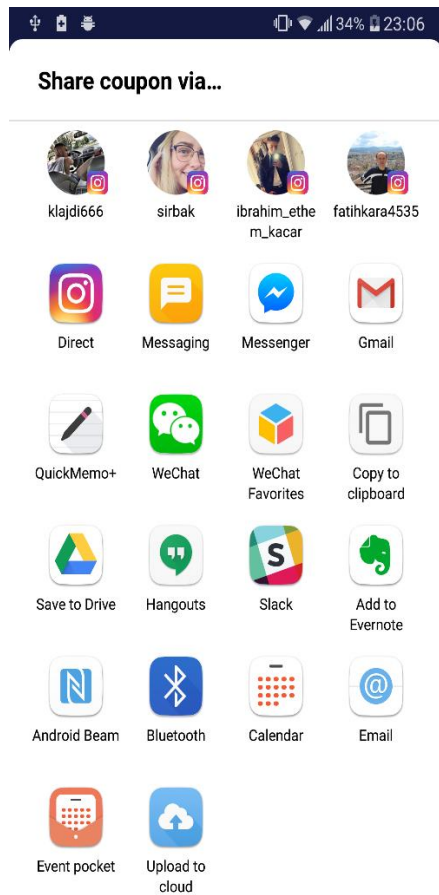
4- Coupon code (if any)

5- description of the coupon (where user can write a short description about the coupon)

Other buttons that user may use are: USE COUPON if user wants to use it or has used it.

EDIT: Where user can edit info about the coupon

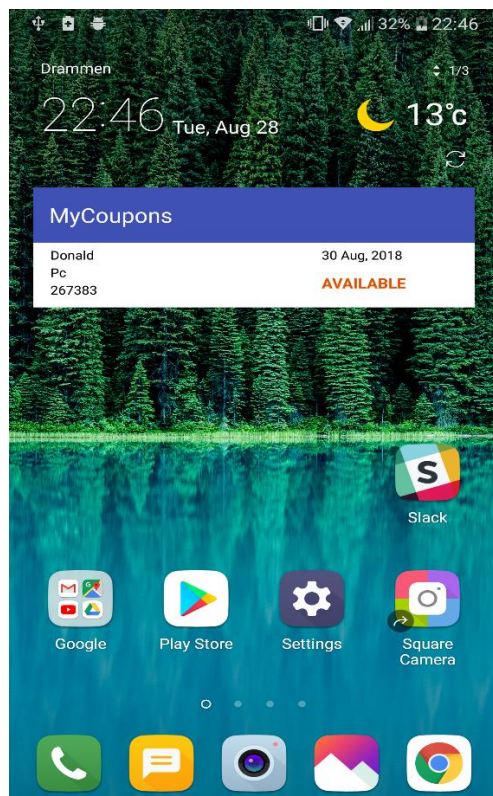DELETE: if user wants to delete coupon.

SHARE: If user want to share the coupon or want to save it to drive

Screen 6 – Share Button

After pressing the share icon/button user will be showed the possibilities to choose where to share the coupon and whether he/she want to upload/save it to drive.

Screen 7 – Widget view on phone

This is how widget looks like on phone. It consists of the list of available coupons and its scrollable. While pressing on a particular coupon the app sends user to the addressed coupon.

I had other screenshots of the app but because of the lack of time I did not include them here. The app is Tablet friendly too but I do not have a table to test it out if it works as it should, that's why I am not including pictures of it here.

Key Considerations

How will your app handle data persistence?

Data persistence is handled by using SQLite table and SharedPreferences. SQLite table is a "coupon" table and it consists of this fields(this handled coupon table objects):
1- _id: Type integer: it's an integer primary key and is the row identifier for each coupon in the table.
2- merchant: Type String: Name of the company/merchant offering the coupon.
3- category: Type String: Type of category the coupon belongs to.
4- valid_until: Type integer: date of form YYYYMMdd converted to a long indicating the expire date of the coupon
5- coupon_code: Type String: Optional code needed to validate coupon
6- description: Type String: Description of the coupon

SharedPreferences are being used for handling notification settings.

Describe any edge or corner cases in the UX.

For example, how does the user return to a Now Playing screen in a media player if they hit the back button?

One thing that can be said here is about the expire date of a coupon. User will be allowed to just choose a date prior to the picker date, I mean they won't be able to choose a date that is older than the day coupon is created. This will be a problem if user changes timezone and enters the time manually. To solve this issue I store the date as a long value, lets say if its 28 august 2018 then it will be stored as 20180828, a long obtained from Long.parseLong("Date string of format YYYYMMdd"). Using other stuff such problems will be obtained:

- Using Unix timestamp will have issues with devices that change timezone.
- Saving date as long is a simple thing and very helpful. It will avoid problems with devices changing the timezone. And it will be very fast since sorting based on an integer in SQLite is quite fast.
- If we save data as text and then invoke datetime() SQLite function to sort coupons by date, will take very long time and will create performance issues.

Another thing can be with daily notifications and alarms.

Describe any libraries you'll be using and share your reasoning for including them.

For example, Picasso or Glide to handle the loading and caching of images.
1- RxAndroid: for RxJava bindings
2- RxJava: for event-based programming and to shorten code
3- CardView: helps displaying coupons as cards.
4- Different Android design libraries: used them to make it possible for me to use elements like tab layout, text input, coordinatorLayout, floating action button etc.
5- Eventbus: to make possible communication between services, fragments, activities etc.
6- Espresso: for testing UI

Describe how you will implement Google Play Services or other external services.

Describe which Services you will use and how.
In my app I will use the following:

1- **Play Services ads api** (com.google.android.gms:play-services-ads:10.2.1) – It will display interstitial ads on the app when user gets back from profile screen. If the ad is loaded, then the users will be able to get/see it.
2- **Play services drive api** (com.google.android.gms:play-services-drive:10.2.1) – Play service will be used to export and import application database. This will made possible for users to save a copy of their database to Google Drive and then access it with another device.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and break them down into tangible technical tasks that you can complete one at a time until you have a finished app.

Task 1: Project Setup
This is an old project of mine and here I will explain steps to create the project:

- Created first some mocks to get the idea how the app interface would look like.
- After thinking about the idea, I thought about which database and dependencies I would use.
- After getting a clear picture of my idea, I created the project in Android studio.
- After that added all dependencies required in gradle and created release file.
- Then I configured build sync file and synced the project.

Task 2: Build database of the app

- Build Coupon that contains Coupon info and CouponContract that contains Db info to begin with.
- Build CouponDbHelper class to handle create and upgrade database.
- Build contentProvider to interact with database and test it

Task 3: Build UI for Coupon and List Activity/Fragment

- Create coupon fragment layout and added modes
- Create tabbed layout for coupon list activity
- Before entering data/details to database, I performed some user input validation for coupon fragment

- Loaded data from database to tabbed layout and create a list with coupons, both past and upcoming.
- Inflate Menu

Task 4: Create My profile

- Created the User profile layout
- Listed all the used coupons
- List no of coupons available and used by user
- After clicking on back button, fixed show interstitial ad. Ads will be loaded with AsyncTask.
- Test

Task 5: Create Setting Fragment

- Create layout of setting fragment.
- Implemented settings preference screen
- Stored and retrieved appropriately user data/preferences from SharedPreferences
- UnitTest

Task 6: Handle data import/export from Google drive

- Implement export database to drive and implement import database from drive.
- Test on device

Task 7: Create widget and set up alarms

- Create layout of widget and handle onClick from widget screen to our app
- Set up alarms for daily notifications
- Set alarm update at 2am daily
- Create BroadcastReceiver to ensure alarms set if device reboots or turns off and then on.

Task 8: Final Testing with Espresso and actual device

- Fix any bug that may occur and test it on device to check if UI is as it should and that everything works as it is supposed to.

Sorry for my English 😊