GitHub Username: Gropoid

PUNTER

Description

My app is a quizz game based on video game industry data.

Intended User

The intended user is a videogame enthusiast who has some time to waste.

Features

My app :

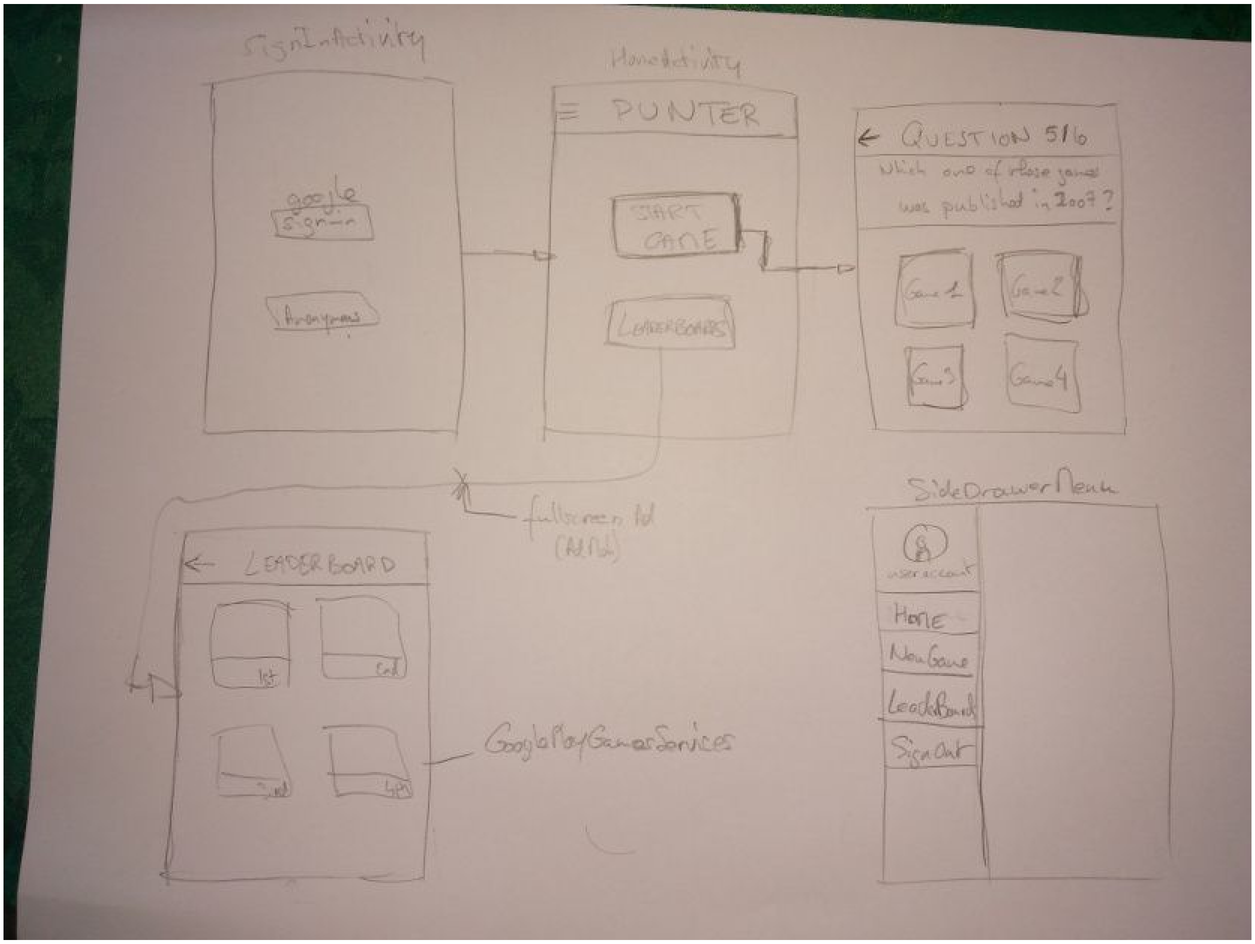
- Generates questions based on videogame data provided by the GiantBomb api
- Maintains a local database of ~100 games
- Presents 10-question quizzes to the player

- All questions are of the “which of these 4 games” type. Thus they can be answered by tapping on a game’s cover picture.
- Allows a user to eithr log into a Google Play Games account, or play anonymously.
- Allows a user to submit his score to a high-score leaderboard
- Allows a user to view leaderboards
- Displays interstitials ads

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 Photoshop or Balsamiq.

Screens



Key Considerations

How will your app handle data persistence?

I intend to maintain a cache of about 100 games (Title, publisher, all sorts of metadata, plus a cover picture).
Questions will be generated at the beginning of a game, and games used for a question will have usage counter incremented. Once a game has been used ~10 times, it gets removed from the local DB. When the number of usable games falls below a certain threshold (50 ?), we query the GiantBomb API again. (via an IntentService)

Describe any corner cases in the UX.

When a user navigates up or back from within the game activity, his current game will be dropped. However a confirmation dialog will let him know the consequence of this navigation action. If he leaves the game by clicking Home, or gets interrupted by another app (incoming call, navigation through Recent Apps…), his game will be lost if the OS garbage collected the application.

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

Retrofit for handling HTTP calls.
Glide for image loading.
Maybe Realm or OrmLite for managing the local DB.

Describe how you will implement Google Play Services.

I will use AdMob for interstitial ads, and Play Games Services for the leaderboard feature.

Next Steps: Required Tasks

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

Task 1: Project Setup

Get an API key with GiantBomb

Explore their API and set up Retrofit for the appropriate requests.
Set up local model: Game, Question, Quizz

Task 2: Content Provider

Access the data model through a Content Provider

Task 3: Implement UI for Each Activity and Fragment

As much as possible, I will follow and MVP pattern. I'll try to follow :
http://antonioleiva.com/mvp-android/
https://github.com/LiveTyping/u2020-mvp-android-studio-template
https://github.com/benoitletondor/Android-Studio-MVP-template

Sign-in activity
Home Activity
Quizz Activity
LeaderBoardActivity
SideDrawer

Task 4: Quizz fragment

Make a custom view for each game (the button on which the player clicks to answer)
Implement the answers as a RecyclerView ?
Implement scoring system : something that depends on the time it took to answer, to make leaderboards interesting.

Task 5: Sign-in

https://firebase.google.com/docs/auth/android/google-signin

Task 6: LeaderBoards

https://developers.google.com/games/services/android/leaderboards

Task 7: Ads

Add an interstitial ad when accessing the leaderboard

Task 8: Widget

Offer a widget that shows the highest (or latest) score achieved by the user.

Task 9: Side Drawer navigation

Offer the user the ability to sign out.