GitHub Username: seeffff

GPSAlarm

Description

Ever fall asleep on the bus or train and miss your stop? How about driving right past where you need to be after a long day? This app will save you. GPSAlarm will notify you before it's too late in ways that are up to you. You can customize your alarm by proximity to the location desired, set vibrate or customize your ringtone volume.

Intended User

The intended user would be people who use public transit systems.

Features

- Create and save alarms for later use
- Ability to customize alarms
- Find your location on a map

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.

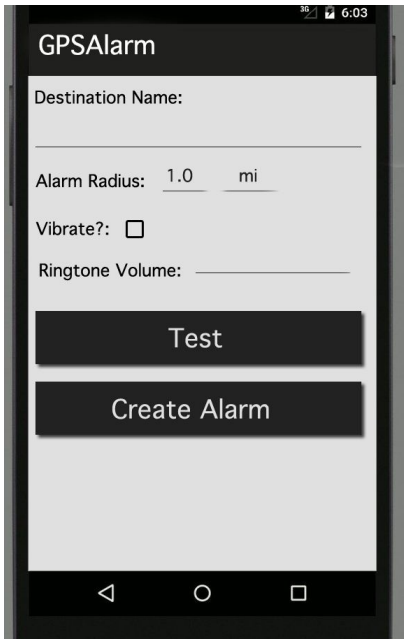
Main Screen

Main screen with the alarms listed. The destination will be a name given by the user. Alarm radius will be also given by the user and be in miles or kilometers. The ringtone volume slider will be able to be changed and the vibrate checkbox will be able to be checked and unchecked as long as the alarm is off. The delete forever icon will show a dialog asking the user if they're sure they want to delete the alarm. The off button will change to red and the background will also be given a light red tint to make sure the user knows an alarm is on.

Set new alarm location

This screen will just be a map with a fab. Once this activity is shown, the user's location will be retrieved and shown on the map. The user will then place a pin somewhere on the map as the alarm location. If the fab is pressed the location of the pin will be saved. If there is no pin, a toast will appear instructing the user to place the pin somewhere. If there was a valid location, an asynctask will query the google places api to see if a name will show up. If the user verifies that the location is correct, the name of the location will be used as the destination.

Set new alarm details



The destination is an edit text allowing the user to create a custom name for the alarm they are creating. To set the alarm radius the user will use two spinners to set miles or kilometers and the amount. The checkbox next to vibrate will either allow for vibration or not. The line next to ringtone volume will be a slider allowing the user to set a custom volume for the alarm. The test button will hide the create alarm button and sound the ringtone and vibrate the phone to the user's specs. The test buttons text will also be changed to stop. Once pressed everything will go back to normal.

Widget

The widget will essentially be another way to trigger or monitor which alarm is active. It will contain a list of all alarms the user currently has saved.

Key Considerations

How will your app handle data persistence?

I will use an SQLite database to store alarms and use a custom content provider to retrieve them.

Describe any corner cases in the UX.

When the service is started there will be a persistent notification being displayed. When pressed it will take the user to the alarm list to turn off the alarm if needed.

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

ButterKnife to bind variables.
Retrolambda for lambdas

Describe how you will implement Google Play Services.

Location to get the user's location and listen for updates.
Maps to allow the user to pick a location for the alarm.
AdMob to display ads.

Task 1: Project Setup

Add the necessary dependencies into the build.gradle file and configure the manifest.

Task 2: Implement UI for Each Activity and Fragment

- Build UI for the alarm list activity
- Build UI for the alarm list details
- Build UI for the create alarm map activity
- Build UI for the create alarm detail activity
- Build UI for the alarm sound activity

Task 3: Handle creation of new alarms

- Make the FAB click send you to the create alarm activity.
- Get the user specified location and send to a specify alarm details activity.

Task 4: Store the alarms

- Create an SQLite database to hold the alarms

Task 5: Get and display the alarms

- Create a custom list adapter and content provider
- Use the content provider to populate the adapter and display the alarms.

Task 6: Handle alarm

- Allow user to delete a create alarm, create dialog to double check
- Create location listener service to get location changes and compare user's location to the alarm location.
- When the user is within the area specified the alarm activity will start

Task 7: Clean up

- Handle any exceptions, device orientation change.
- Polish the UI in any places necessary.
