# Location Based Event Triggering Android Apps

Amitoz Azad @ Master 3DMT
3-12-2017

## Abstract

*Time based event triggering applications are very common in the market. That is an event is triggered at a particular time specified by the user. Applications in which an event is triggered based on the target location specified by user or the locations of two or more users are very desirable. In this paper, I provide brief and concise description of project for course IMT4889-Specializing in Mobile and Wearable Technology taught at NTNU Gjovik in fall of 2017 to create such applications. Points like how to use my apps, important technical details, criticism of my apps and their advantages over current apps in Market are also conferred.*

## 1 Introduction

Before reading this report it is indispensable for the reader to go through my video [8], this will make reader understand the report better as I have shown demo of the two apps. Broadly this project is divided in two parts. In the Part1 work is done to create an application in which an event is triggered whenever user enters in a radius, with respect to a fixed location selected by the user. In the Part2 the aim was to create an app in which an event is triggered with respect to mobile locations of two users. Please note that objective was not to develop fully functional working apps which should be publishable but rather to be able to implement core functionality of such apps. *Applications created by me are only to serve basic purpose to enrich my understanding while learning how to accomplish the objectives of project. They are still very far from being publishable.*

To understand the Part1 consider the this example. Every time I go to campus, I keep forgetting that I need to borrow a book from library. In this scenario it will be very useful, to have an application which triggers an event (in this case a reminder) that reminds me whenever I am in 500m of the campus to go the library and borrow the book.

The Part2 of the project was dedicated to create an application in which some event is triggered based on the locations of two users, to understand it consider this example. Binu is my good friend. He has borrowed my notes. To have an application which alarms me whenever Binu is in range of say 1000m could be very useful, so that I can go, meet him and pick my notes. This part was much challenging. As I had to set up a server to track users.

Brief outline of this report is as follows. Section 2 is a brief guide to use my apps. Section 3 is dedicated to important technical details of the project, for both the Part1 and Part2. In section 4 we discuss some advantages of these apps over the apps in the market. In section 5 some directions on future work to make these apps publishable, basically this section can also be considered as self criticism of my apps. Finally in section 6 some key observations while testing the apps.

Due technicality of the project(specifically for Part2) it was overwhelming for me to explain everything from scratch, therefore it is assumed that reader is somewhat familiar with android studio(like basic terminologies minSDK, targetSDK, creating simple UIs, what is MainActivity, what is AndroidManifest.xml), parse server [3](parse query, parse objects) and and basics of amazon ec2 instance [11] (like setting up and connecting to your instance remotely etc).

## 2 How to Use My Apps?

My apps can be downloaded and installed from the repository [10]. Even though I am explaining here on using my apps, it is again suggested to go

through my video [8] for better visualization and understanding.

## 2.1 Part1 Application

1. Download the Part1.apk and install it, Figure 1 is the home screen of this app.

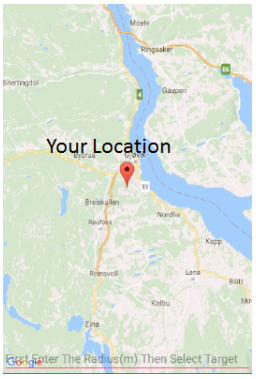


Figure 1: Home screen Part1.apk.

2. Enter the radius in meters in EditText at bottom of map see Figure 2.


Figure 2: Radius as 1000m being inserted.

3. Now long press to select a target. Reminder is set, Figure 3.

4. Now whenever you are in the 1000m radius of the target(NTNU Gjovik) and event(in my app it's an air horn) is triggered.

## 2.2 Part2 Application

1. Download the Part2.apk, run it, Figure 4 shows the home screen of this app.

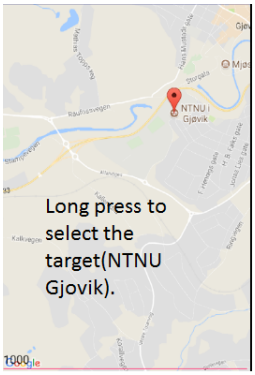


Figure 3: Reminder is set a NTNU Gjovik campus.

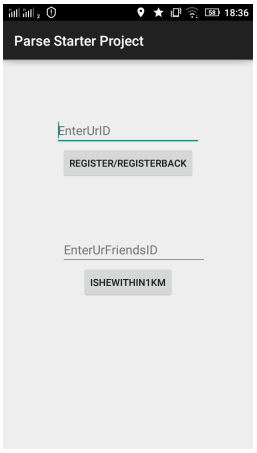


Figure 4: Home screen of Part2.apk

2. Choose an ID and enter in top EditText, say 'Amitoj' and press the button Register/RegisterBack. See Figure 5

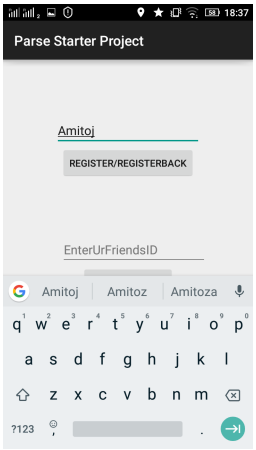


Figure 5: Registering with a new ID 'Amitoj'.

3. Now enter your friends ID(assuming he has registered) in bottom EditText , say 'mdbinu' and press button ISHEWIHTIN1KM(to be read as *is he with in 1km*). See Figure 6

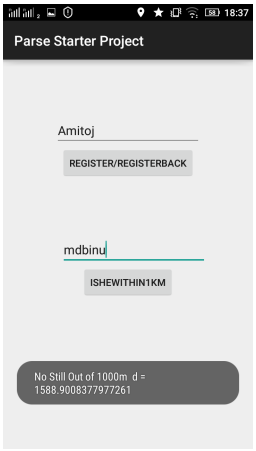


Figure 6: Amitoj is checking is mdbinu is with in 1km.

4. If he is with in 1km range an event(air horn) is triggered. If not a toast appears saying not in 1km, Figure 6.

One can monitor the entries in the Parse server through Parse dashboard, which is an extremely useful tool. See point 6 of next section on how to connect to dashboard or jump to 9:30 in my video for more explanation [8].

# 3 Important Technical Details

Although not complete details, but If someone is interested in reproducing this project. He/She should find these details as the most useful.

1. **Development and Testing Environment**: For Part1 Android Studio Version 2.3, minimum SDK version 21, target SDK version 26, Gradle version 3.3. For the Part2 everything is same except minimum SDK version 9 and target is 23. For testing of the Part1 and Part2 API 21(Lollipop) and 23(Marshmellow) were used.

2. **Permissions**: Whenever a user sensitive information is utilized by an app. Android requires us to specify the permission in AndroidManifest.xml file. Permission pertaining to user sensitive information are Dangerous Permissions, which are always asked to the user. Until android SDK 22, Android 5.1 listing a dangerous permission in manifest was enough and user grants permission when they install the app. If they do not grant permission, the system does not install the app. Beginning Android 6.0( i.e target SDK 23) or higher the app has to list the permissions in the manifest, and it must request each dangerous permission it needs while the app is running. This makes the code more complicated to understand for novice. Since the target sdk was kept as 26 and 23 in my apps. I had to write my code in such a way that app request permission in run time to work it on Marshmellow and higher [2].

3. **External APIs used**: For the Part1 of the project google map API key is being used. This key is in under 'values' folder in 'res' directory. For the Part2 of the project I am using code by Rob Percival [12], 'Parse Starter Android Studio Project'. This code assisted me in setting up a parse server. Apart from that his code also has basic example of how to store some basic data(like strings) on parse, however it doesn't contain any info on how to retrieve those values through parse query and also the idea of storing geolocation in parse was my own. Since I was using external code which was written in legacy version of android studio . I had to build everything with studio version specified in one. Even then there was an issue which was resolved by adding jcentre() to my repositories in build.gradle file [5]. There is a better option which I realized later, in-

stead of using Rob Percivals code, use the official code available 'Parse-SDK-android' [1] and follow their READMe.md to connect server and client(also briefly explained in following points).

4. **Server**: I am running an ec2 instance (t2.micro) in amazon aws [4]. Under ec2 instance a Parse server powered by bitnami is available in AWS Market Place (this means it comes with the default installation of Parse) and it is eligible for free tier. The server is running in Frankfruit in zone eu-central-1b. The ip address of the server is 18.195.26.128.

5. **Connecting to Parse Server from Your App**: Brief explanation is given on github [1]. We essentially need client key and appID to connect our app to server. This data can be obtained from connecting to parse server remotely from your system (in my case ubunut16.04LTS). Kindly look at Figure 7 to see how it was done from my ubuntu system. Once you are logged into the Parse server, you

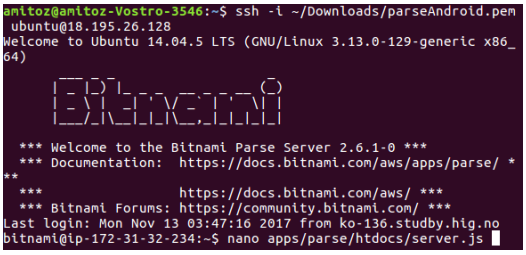


Figure 7: Remote access to Parse server and trying to open server.js file.

need to open server.js file. The command can be seen in the above Figure and below.

***nano apps/parse/htdocs/server.js***

This file has all the data we need to connect our app to the server, see Figure 8. Copy the appID and clentKey(masterKey) from this file and then use it in your app as specified in the official documentation [1]. Apart from clientKey and appID we also need our server's IP4 address of ec2 instance, which is very easy to get(if you are familiar with amazon aws).

6. **Connecting to Parse Dashboard**: While saving data on the Parse server, dashboard is a handy tool to check if data has been saved on and when it was saved etc. Kindly check my video [8] at 9:30 for better understanding. To

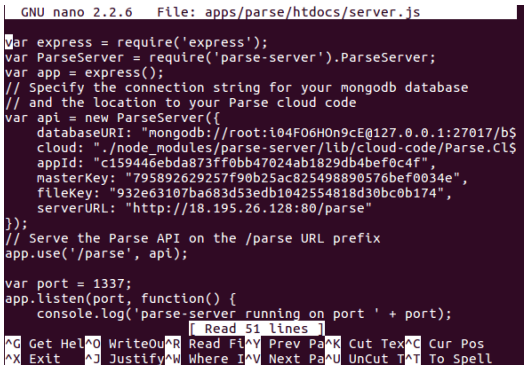


Figure 8: The server.js file opened in nano editor.

log into the Parse dashboard simply go to your server ip address (in my case 18.195.26.128) in your web browser. Access the dashboard by entering the Username as 'user' and Password can be obtained in 'Get System Log' in 'Instance Setting' in 'Actions' Tab in your running instance on amazon aws(again familiarity with Amazon aws is assumed), see Figure 9. Inside the system log one can then find the password see Figure 10.

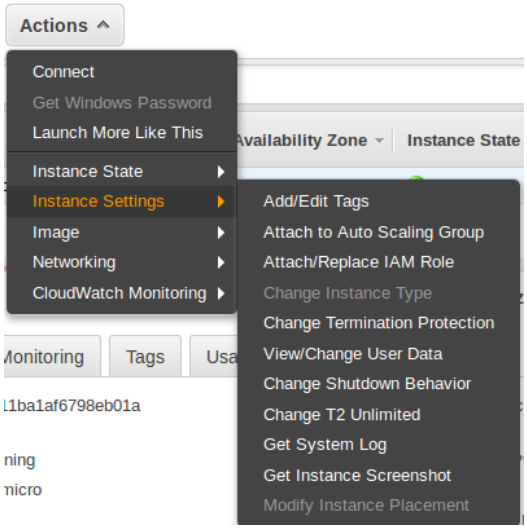


Figure 9: How to open system log in aws.

# 4 Edge Over Market Apps

Even though there are some location based event triggering applications in the market. But according to my knowledge they lack this user convenient feature of selecting the target form the map which

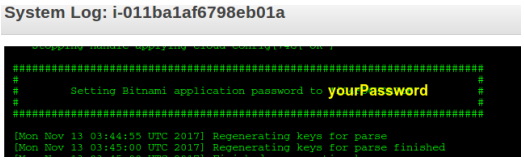


Figure 10: Copy your password and paste it in Parse dashboard login page.

I had introduced in Part1 app. Most of these application one has to specify the longitude and latitude of the target destination.

For the Part2 of the project there is **no such** application in the market which is working on idea of triggering an event when two users are in the range of some particular distance. There are family locator applications in which people from a family willing share their location to other family members. From the reviews I came to know that people also use these apps to spy on their partners by secretly installing this application. The core point is this idea of such app in which there is triggering an event when two or more user come within a specific distance range is new.

# 5 Future Work

This section can also be considered as self-criticism. Following I have written some points which highlight main drawbacks of my apps. And would also be useful for someone working in future on this project to publish these apps.

1. **User Friendliness**: User friendliness is very less in my apps, I am doing everything in one Activity. In my Part2, I have hardcored the distance as 1km as range of distance between two users. It is very easy to include a third EditText in which user can specify the distance range. Error handling has to be done properly. For example the case when user enters string value instead of numeric value, such kind of small-small checks have to be made.

2. **Objective of the Part2 was not met**: Right now the second app doesn't serve its purpose as it is not automatic. Because user has to press a button('ISHEWITHIN1KM') repetitively to see if the other user is in the range of 1km. In order to make this app automatic the call-back behind the button 'ISHEWITHIN1KM' has to be called repetitively say once in one hour.

3. **Use offline map in Part1**: Prior to including this user convenient feature of selecting the target from the google map, my app didn't need wifi connection to work. Since now google map api is used in Part1. User needs wifi connection to select the target. Once the target is selected the whole game is wifi independent. One can think on using offline maps so that the app becomes again wifi-independent.

4. **Not using the user class in Parse**: Parse server has this user class option. Which I am not utilizing. The advantage of using this user class is that this will ensure that each user will have different username. It will create a proper login setup for them, where they enter username and password before logging in the app(Part2). This has not been taken care by me.

5. **Running the app when user kills the activity**: As an example if a user sets reminder at particular location and then kills the app then app should still work in that case (as normal alarm applications do work). Right now the app has to be running in UI thread for it to be working (this means one should be able to see the app in task manager). This running in background can be done by using IntentService class [6] Please note that if app is made to run in background then from API 26 onwards then there is a limit on how frequently app can retrieve the user's current location. Apps can receive location updates only a few times each hour [7].

6. **Integrations of both Part1 and Part2**: There is no specific reason why there cannot be one app fulfilling both the purpose. The only reason I did everything in two separate apps was to Keep the things neat and easy.

7. **Battery Consumption Should be Taken Care of**: In the call-back 'locationManager.requestLocationUpdate' developer can specify minimum time and the minimum distance that will lead to location updates. Right now I have kept them as 0,0 which means user get every update. To be careful of user's battery one can easily change those values.

# 6 Some Key Observations While Testing

For Part1 of the project, the app behaves slightly different on Lollipop and Marshmellow. In home screen you should see a marker at your location with zoom level of 10 on the map, somehow this works properly on Marshmellow but not on Lollipop. However the core functionality of the app that is to specify the radius and select the target to set reminder works properly on both.

For Part2 of the app while testing it with my friend Binu, we were at the entrance of the G building. We both registered simultaneously at same location. But in Parse dashboard, it was observed that location coordinates of us were differ by distance of 330 meters. Reason for this much difference, even though we were standing at same location is not clear. It might has to do that GPS receivers of our mobile were different, check SO [9].

# References


[1] Parse SDK Android. https://github.com/parse-community/Parse-SDK-Android.

[2] Requesting Permissions at Run Time. https://developer.android.com/training/permissions/requesting.html.

[3] Parse Server Documentation. http://docs.parseplatform.org/parse-server/guide/.

[4] Amazon EC2. https://aws.amazon.com/ec2/.

[5] Gradle Sync Failed. https://stackoverflow.com/questions/32859819/androidstudio-gradle-sync-failed-to-resolve?noredirect=1&lq=1.

[6] Running in a Background Service. https://developer.android.com/training/run-background-service/index.html.

[7] Background Location Limits. https://developer.android.com/about/versions/oreo/background-location-limits.html.

[8] Video Description of Project. https:www.youtube.com/watch?v=U0ppt u8V1rs&t=825s.

[9] Selecting Specific GPS Provider. https://stackoverflow.com/questions/47608281/is-there-any-way-to-choose-a-specific-gps-provider.

[10] My Repository. https://github.com/amitNTNU/Apks-Part1and2.

[11] Setting up Amazon aws EC2 instance. http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html.

[12] Setting up your Parse Server. http://www.robpercival.co.uk/parse-server-on-heroku/.