

Android Project Description: Georienteering

Description

This is an individual project work made by Pertti Palokangas (pp222es) in 2014 within the university course *Android for Java Programmers - 2DV106 - 7,5hp* – Linnaeus University, Växjö.

General project description

An orienteering app for walking or running and searching for control points using GPS and map.

A minimum of two and a maximum of ten control points forms a course, with a starting and ending point at a start point location where user starts the exercise activity, by selecting Start from the menu. The goal is to find each control point as quickly as possible, and the user has the freedom to choose the most suitable way to them.

Goals

To motivate user to get some exercise and hopefully having fun! A specific goal was to use the “*gamification*” concept by combining training app functions with a simple game idea. Another goal was to achieve robustness so that an ongoing game would survive when app is put to background or closed. An important goal was to keep the game and app as clean and simple as possible so it would be easy to understand how to use it.

Use Cases

1. User changes preferences. Done by choosing menu item **Settings**. Four preferences can be updated:
 - Difficulty Level
 - Default Start Point location
 - Display Ghost
 - Display Route
2. User starts app to use it for looking at map, zooming in/out, switching map type etc. just like any other map app.
3. User creates a new game map. Done by choosing menu item **Create new game**. A new activity *CreateNewGameActivity* is shown. Here user can enter:
 - Desired way of placing the start flag. 3 options:
 - Use a flag that is dragged to the position
 - Use current location for the position (requires GPS)
 - Use coordinates from edit fields.
 - The number of control points to be placed on the map.
4. User has created a new game map (by selecting OK in use case 3). User now places the control points at any location on the map. It is the responsibility of the user to place control points at locations that are accessible, and that are not inside

dangerous or forbidden areas. Depending on selection in use case 3 user may also move the start flag.

5. When user is ready with use case 4, user can push the **Save** button, which will save the game map and the positions of the control and start points into internal storage.
6. If a game has been saved, user can also choose to load a previously saved game via menu item **Load last saved game**. Currently only the last saved game is possible to load.
7. When a game has either been created or loaded, the game can be started from menu item **Start game**. The user is advised to move to the start point, however it is possible to start the game whenever user likes to, provided that use cases 5 or 6 have been performed.
8. During the game, user's position is shown on the map. The update frequency is set depending on the chosen difficulty level in use case 1.
9. User reaches a control point, and the broadcast receiver *GeofenceDetectorReceiver* notices that a geofence enter transition has occurred. A broadcast message is then sent to main activity. A notification icon is displayed in notification bar if main activity is not visible.
10. If user in preference settings has enabled the ghost in use case 1, then a ghost will appear after a certain time. The time is determined out of the chosen difficulty level.
11. During game, the time and distance that user has moved is displayed. If a ghost was enabled then also the distance to the ghost is displayed.
12. If user finds all controls, then when reaching back to the finish point the game has been won.
13. If user does not find all controls when reaching the finish, the game was lost.
14. If user stops the game by choosing menu item **Stop game**, the game is lost.
15. If the ghost catches up the user before the finish line, the game has been lost.
16. When game is won a winner toast is displayed and a fanfare sound is played.
17. When game is lost a toast is shown and a spooky ghost sound is played.
18. User chooses to save the game data to database.
19. User discards the game data.
20. User displays game data by clicking on menu item **View game data**. The game data consist of:
 - Date
 - Time
 - Distance
 - Number of control points found
 - Total number of control points
 - An icon telling if the game was won or lost
21. User displays license information via menu item **License Info**.
22. User displays app info via menu item **About this app**.
23. User puts app to background via Home or Task History buttons.
24. User closes app via Back button.
25. App is unexpectedly killed by system.
26. Screen orientation is changed, when not an ongoing game.
27. Game is started in portrait mode and screen orientation will be locked in portrait mode during the ongoing game. This is unlocked when game stops.

28. Game is started in landscape mode and screen orientation is locked in landscape mode during the ongoing game. This is unlocked when game stops.

Error use cases:

- 29. App is started but Google Play Services is not available. In this case a toast is shown with the message from resource string `R.string.google_play_services_not_available`.
- 30. Use case 29 occurs, but the error is recoverable. In this case an error dialog fragment is shown where the user is prompted to install Google Play Services.
- 31. App is started and geofences are still set up, but not in Prepared or Play Game mode. An attempt to remove the geofences will be made.
- 32. A geofence request (add or remove) fails with `onConnectionFailed`. An intent is sent to main activity.
- 33. Use case 32 occurs, but error is recoverable. A Google Play services activity that can resolve the error is started.
- 34. If a geofence error is detected in the broadcast receiver, then an intent `GeofenceUtils.ACTION_GEOFENCE_ERROR` is broadcasted.
- 35. User enters invalid values for latitude and/or longitude in preference activity. Since validation fails a toast with resource string `R.string.location_input_error_converted_latitude_invalid` and/or `R.string.location_input_error_converted_longitude_invalid` is shown. Also invalid edit fields will have a red color.
- 36. User enters invalid values for latitude and/or longitude in Create New Game activity. Since validation fails a toast with resource string `R.string.location_input_error_latitude_invalid` and/or `R.string.location_input_error_longitude_invalid` is shown. Also invalid edit fields will have a red color.
- 37. App is started or resumed, but the Google map is not set up. A toast with string `R.string.map_error_msg` is shown.
- 38. A database error occurs. A toast with string `R.string.database_open_error` is shown.
- 39. App is started but no location provider is available. A toast with string `R.string.location_provider_not_enabled` is shown.
- 40. An unsupported action is performed while game is ongoing. A toast with message `play_mode_info_msg` is shown.
- 41. An unsupported action is performed while a game has been prepared (after Save Game). A toast with message `prepared_mode_info_msg` is shown.
- 42. An unsupported action is performed while not playing a game. A toast with message `not_in_play_mode_info_msg` is shown.
- 43. An unsupported action is performed while not in a prepared state. A toast `not_in_prepared_mode_info_msg` is shown.
- 44. Wrong state for recreate game action. A toast `cannot_recreate_game_in_prepared_mode_info_msg` or `cannot_recreate_game_in_play_mode_info_msg` is shown.
- 45. User starts a game but location tracking fails to start. Toast `location_tracking_failed_to_start` is shown.
- 46. The display route function failed. A toast `show_route_failure` is shown.
- 47. No internet connection. A toast `R.string.network_error_msg` is shown. This check is done in `onResume`.

Explanations

Screen orientation lock during ongoing game

Because I and others found it impractical and rather annoying that the screen configuration changes were often triggered even when we did not want them, while moving (jogging, running) and when having the phone in a pocket, I decided to lock the screen orientation in the current mode when the game is started. This means that if the user prefers to play the game in landscape mode, then user must start the game while in landscape mode, and vice versa. The lock is unlocked when game is stopped. I found information that for a full screen game this is a technique that is quite wide-spread and acceptable. Like in ref [1].

Searching for control points in order is not a requirement

Control points are numbered starting from 1, but I found it more interesting and challenging to let the user decide in which order to find the controls. Also this actually made the game concept a bit easier to grasp for those not familiar with traditional orienteering rules. In other words there is no requirement to search for the controls in a particular order.

Features

1. Current map position (start point) and zoom level is saved and restored when app is restarted.
2. Game map data is saved (in internal storage file) and is recreated if app is killed or destroyed during the prepared state, i.e. the state between the Use case 5 and a started game.
3. Ongoing game data is saved (in preferences file) and is recreated if app is killed or destroyed during the play game state, i.e. during use cases 8-11.
4. The state of main activity (visible or alive) is updated to the service LocationTrackerService.
5. LocationTrackerService stores each location in a static list that is persistent as long as the service runs.
6. LocationTrackerService starts a Timer thread when game is started and stops it when game is stopped.
7. Callback with location update from Location services is received in LocationTrackerService.
8. LocationTrackerService increments the total distance.
9. LocationTrackerService broadcasts an intent with location data and distance, but only if client (main activity) is alive.
10. GeofenceDetectorReceiver is a broadcast receiver that works independently of any other component. It will detect a geofence enter transition within a radius of 20 meters.
11. GeofenceDetectorReceiver broadcasts the ids of the geofences that were triggered in an action intent `GeofenceUtils.ACTION_GEOFENCE_TRANSITION`.
12. GeofenceDetectorReceiver saves the state for the found controls in preferences and shows a notification if the main activity is not alive or visible.
13. Main activity restores the map view depending on the state and updates the control points found by reading their state from preferences.
14. Game data is saved in a database using classes `DbOpenHelper`, `GameDataSource`.

Components

Activities

- MainActivity
- CreateNewGameActivity

- `GameDataActivity`
- `MyPreferenceActivity`

Fragments

- `MapFragment` with tag `MAP_FRAGMENT_TAG`
- `ErrorDialogFragment`
- `MyPreferenceFragment`
- `SaveDialogFragment`

Broadcast Receivers

- `GeofenceDetectorReceiver`

Services

- `LocationTrackerService`

Background tasks

- `GhostMover`
- `AudioClipPlayerTask`
- `TimerTask`

Helper and Utility classes

- `DbOpenHelper`
- `FileUtils`
- `GameData`
- `GameDataSource`
- `GeofenceRemover`
- `GeofenceRequester`
- `LocationServiceErrorMessages`
- `LocationUtils`
- `SimpleGeofence`

Layouts

- `activity_create_new_game.xml`
- `layout-land/activity_create_new_game.xml`
- `activity_game_data.xml`
- `activity_main.xml`
- `my_toast.xml`
- `row.xml`
- `xml/preferences.xml`
- `menu/main.xml`

Dependencies

- Google Play Services Library
- OpenGL ES 2.0

Permissions

- `android.permission.INTERNET`
- `android.permission.ACCESS_NETWORK_STATE`
- `android.permission.WRITE_EXTERNAL_STORAGE`
- `android.permission.ACCESS_FINE_LOCATION`

- `android.permission.ACCESS_COARSE_LOCATION`

Package name

- `dv106.pp222es.georienteering2`

Technical features and parameters

Geofence radius: 20.0 meters

Location update frequencies:

- EASY level: 20 seconds
- MEDIUM level: 10 seconds
- HARD level: 4 seconds

Default Game Difficulty Level: EASY (walking)

Default start point location:

```
public static final double DEFAULT_LOCATION_LATITUDE =
    LINNAEUS_UNIVERSITY_LOCATION_LATITUDE;
public static final double DEFAULT_LOCATION_LONGITUDE =
    LINNAEUS_UNIVERSITY_LOCATION_LONGITUDE;

// LINNAEUS UNIVERSITY, TRACK AND FIELD ARENA, WGS84 decimal (lat, lon):
56.882341, 14.822005
public static final double LINNAEUS_UNIVERSITY_LOCATION_LATITUDE =
    56.88234d;
public static final double LINNAEUS_UNIVERSITY_LOCATION_LONGITUDE =
    14.8220d;
```

Note however that the preferences.xml has 0.0 as default values for both latitude (*latitude_pref_key*) and longitude (*longitude_pref_key*), so at very first start these will be the default values:

```
android:defaultValue="0.0"
```

Unless not updated (Was updated 20150117).

Number of minutes from start of game until ghost starts to move (If ghost is enabled):

EASY level: 3 minutes

MEDIUM level: 2 minutes

HARD level: 1 minute

MapMode states:

```
INIT_MODE           //Initial mode

CREATE_MODE         //Mode for creating a new game (placing control points on
the map)

PREPARED_MODE       //Prepared mode, when everything has been set up and is
ready for a new game

PLAY_GAME_MODE      //Play mode, when user has started to play the game
```

```
STOPPED_MODE    //Stopped mode, when game was stopped either by user or
game was won/lost
```

Problems

Problems faced during project implementation, how they were solved.

1. Failed to get GoogleMap handle in onCreate. Map fragment was not yet initialized, which led to that the method call setupMapIfNeeded FAILED in onCreate. Solved by moving the map fragment setup code to onStart instead, so that the map fragment has had time to initialize fully.
2. Problems with not being able to resume a saved game (with map markers placed out), when app was killed and then restarted. Solved with adding recreate game and map functionality, saving data to preferences and the course data to internal file storage. Using a flag set in onCreate so that knowing later on in setupGameMap method (where map handle is valid) whether a recreate scenario shall commence.
3. Problems with not being able to resume an ongoing game, when app was killed and then restarted. Solved by adding recreate game and map functionality, saving data in preferences. Using a flag set in onCreate so that knowing later on in setupGameMap method whether a recreate scenario shall commence.
4. *W/System.err(4042): java.lang.IllegalArgumentException: Receiver not registered: dv106.pp222es.georienteering.MainActivity\$4@b1023860.*
Solved by changing the unregister code for the LocalBroadcast receiver to this:
`LocalBroadcastManager.getInstance(this).unregisterReceiver`
5. Sometimes geofences were not activated, even though the GeofenceRequester sent a broadcast message indicating that adding the geofences were successful. Not sure if this was an emulator problem or a problem with the mock locations or a real problem, because I have not had the chance to test the app that much “irl” as I would like to. Workaround: Stop the game and either create a new game or load a saved game and then start the game.
Possible root cause location: seems to be a bug with `removeGeofencesByIntent` and/or `getRequestPendingIntent` when calling the method `removeMyGeofences()`.
Possible solution: Using `removeGeofencesById` instead seems to solve the problem.
6. Problem with Location Tracking and Timer not starting, even if service seems to be running (on JellyBean phone).
Root cause: After a screen orientation change an automatic binding to service is being made in onCreate. And then when starting the game the `onServiceConnected` callback will not be called since the service was already bound. And thus no call of `myService.startTracking` was made.
Solution: Only call `doBindToServiceIfIsRunning()` when in Play Game mode.

Problems left unsolved, ideas about possible solutions.

7. Google maps crashes with fatal signal 11 on emulator. Reason unknown. Crash signature:
01-06 06:43:34.710: A/libc(2878): Fatal signal 11 (SIGSEGV) at 0x00000000 (code=1), thread 2927 (Thread-138).
8. Due to problem #1 above, it has been very difficult to use the emulator to test how restoring and recreation of an ongoing game works, because when app is destroyed with this particular crash, then the service (running in the same process) that otherwise should have continued executing in background also is killed.
9. *Cast exception in ServiceConnection.*
E/AndroidRuntime(2420): java.lang.ClassCastException:
android.os.BinderProxy cannot be cast to
dv106.pp222es.georienteering2.LocationTrackerService\$LocationTrackerBinder.
Seen only once. Have no idea about why this happened.
10. Seen sometimes on emulator. Root cause unknown.
01-12 11:36:44.077: E/HardwareRenderer(5347): An error has occurred while drawing:
01-12 11:36:44.077: E/HardwareRenderer(5347):
java.lang.IllegalStateException: The display list is not valid.
01-12 11:36:44.077: E/HardwareRenderer(5347): at
android.view.GLES20DisplayList.getNativeDisplayList(GLES20DisplayList.java:49)
01-12 11:36:44.077: E/HardwareRenderer(5347): at
android.view.GLES20Canvas.drawDisplayList(GLES20Canvas.java:420)
11. SpannableStringBuilder errors seen on JellyBean phone (not emulator) in conjunction with screen orientation configuration change:
01-12 18:11:08.427: I/MainActivity(845): Attempting to bind to already running service...
01-12 18:11:08.527: I/System.out(845): StateMonitoring: Start
01-12 18:11:08.547: I/System.out(845): StateMonitoring: RestoreInstanceState
01-12 18:11:08.547: I/System.out(845): StateMonitoring: Resume
01-12 18:11:08.617: I/LocationTrackerService(845): Service Destroyed...
01-12 18:11:08.627: I/LocationTrackerService(845): Service Running...
01-12 18:11:08.647: I/libblt_hw(845): Library opened (handle = 1, fd = 80)
01-12 18:11:08.757: I/libblt_hw(845): Library closed (handle = 0, fd = 83)
01-12 18:11:08.777: I/MainActivity(845): onServiceConnected
01-12 18:11:08.777: W/MainActivity(845): DEBUG:onServiceConnected
01-12 18:11:08.807: E/SpannableStringBuilder(845):
SPAN_EXCLUSIVE_EXCLUSIVE spans cannot have a zero length
01-12 18:11:08.807: E/SpannableStringBuilder(845):
SPAN_EXCLUSIVE_EXCLUSIVE spans cannot have a zero length
Problem is recreated by changing the screen orientation before starting a new game.

Slightly improved by avoiding empty strings like "" when calling `positionDisplay.setText`. So now it occurs ONCE after an orientation change, but an attempt to start game should still work fine.

12. `SpannableStringBuilder` errors were also seen (sometimes?) on JellyBean phone (not emulator) when adding geofences:

01-12 18:18:55.763: I/GeofenceRequester(845): Add request was sent to Location Client

01-12 18:18:55.783: E/SpannableStringBuilder(845): SPAN_EXCLUSIVE_EXCLUSIVE spans cannot have a zero length

01-12 18:18:55.783: E/SpannableStringBuilder(845): SPAN_EXCLUSIVE_EXCLUSIVE spans cannot have a zero length

01-12 18:18:55.783: I/GeofenceRequester(845): Add Geofences: Success
GeofenceRequestIds=[0, 1, 2, 3, 4, 5]

These errors have not been possible to reproduce.

Future work

Possible future work, additional features.

1. Database table for saving several courses, and linking that table with the existing `GameData` database table using *courseID* as secondary key.
2. Saving map and game data to a server and/or in the cloud, making it possible for several users to run the same game simultaneously, and thus compete against each other.
3. Possibility to add several Start Point locations (saved in a database or internal storage) and let user select the default location in user preference settings.

Instructions

Instructions on how to prepare the device.

1. First you need to enable Location on your device:

Settings -> Locations and access -> Check the **Enable wifi and gps** option

2. You are highly recommended to set High accuracy for GPS, Wi-Fi and mobile networks to determine an accurate location:

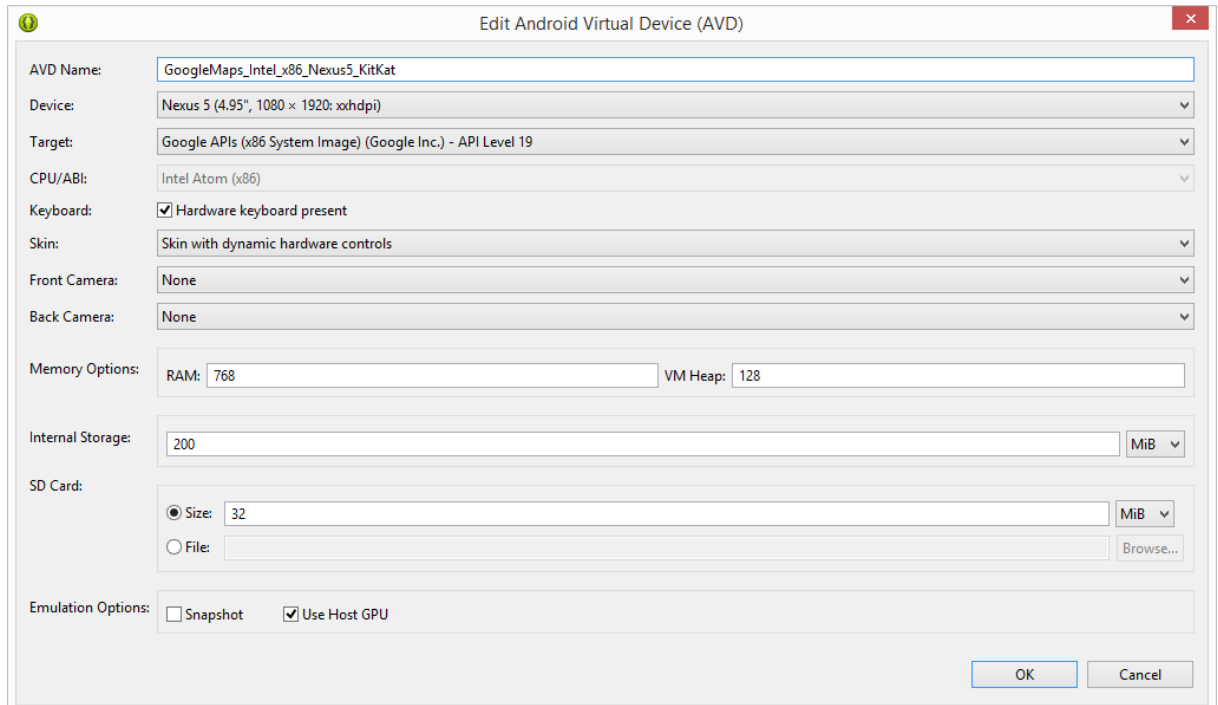
Settings-> Location Mode -> Check **High accuracy** option

You need to agree to the Google terms.

3. For testing on emulator using mock locations:
 1. Dev Tools - Developer options:
Check **Allow mock locations** option.
 2. Start mock location test program, like `LocationProvider` sample from Google, or use DDMS.

Instructions for debugging the app using Eclipse and Emulator

1. Use Eclipse Virtual Device Manager to start an AVD. Use following configuration (for KitKat, v4.4.2, API19) on a Windows 7 PC:



AVD Name: GoogleMaps_Intel_x86_Nexus5_KitKat

Device: Nexus 5 (4.95", 1080 x 1920: xxhdpi)

Target: Google APIs (x86 System Image) (Google Inc.) - API Level 19

CPU/ABI: Intel Atom (x86)

Keyboard: ☒ Hardware keyboard present

Skin: Skin with dynamic hardware controls

Front Camera: None

Back Camera: None

Memory Options: RAM: 768 VM Heap: 128

Internal Storage: 200 MiB

SD Card: ☒ Size: 32 MiB ☐ File: Browse...

Emulation Options: ☐ Snapshot ☒ Use Host GPU

OK Cancel

2. Start and Launch the AVD.
3. Run app from Eclipse using Play button.

Instructions for debugging the app using Eclipse and a real phone

1. Connect the phone via USB to PC.
2. Run and install app on your device using Play button in Eclipse.
3. Select your device when prompted.

References

References to external resources (documents, books, projects, source codes) used in the project.

1. Android™ Programming Pushing the Limits, Erik Hellman, John Wiley & Sons Ltd, ISBN 978-1-118-71737-0 (paperback); ISBN 978-1-118-71730-1 (ePDF); 978-1-118-71735-6 (ePub)
2. Professional Android 4 Application Development, Reto Meier, Wiley, ISBN-13: 978-1118102275 ISBN-10: 1118102274
3. <http://www.intertech.com/Blog/saving-and-retrieving-android-instance-state-part-1/>
4. <http://stackoverflow.com/questions/13670374/android-span-exclusive-exclusive-spans-cannot-have-a-zero-length>
5. <http://stackoverflow.com/questions/3611457/android-temporarily-disable-orientation-changes-in-an-activity>

6. <http://stackoverflow.com/questions/6493517/detect-if-android-device-has-internet-connection>
7. <https://mymoodle.lnu.se/mod/resource/view.php?id=420382>
8. <https://mymoodle.lnu.se/course/view.php?id=10531>

References in code has been marked with the tag REF.