ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ ΚΑΙ ΔΙΚΤΥΩΝ

Ανάπτυξη εφαρμογών σε έξυπνα κινητά τηλέφωνα για καταγραφή και μελέτη κίνησης σε εξωτερικούς χώρους

Application development on smart phones for study of outdoor positioning and locomotion

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΦΩΤΕΙΝΗΣ ΜΠΕΛΗΓΙΑΝΝΗ

Βόλος, Μάρτιος 2012

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ ΚΑΙ ΔΙΚΤΥΩΝ

Ανάπτυξη εφαρμογών σε έξυπνα κινητά τηλέφωνα για καταγραφή και μελέτη κίνησης σε εξωτερικούς χώρους

Application development on smart phones for study of outdoor positioning and locomotion

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΦΩΤΕΙΝΗΣ ΜΠΕΛΗΓΙΑΝΝΗ

Επιβλέποντες :

Τσομπανοπούλου Παναγιώτα
Επίκουρη Καθηγήτρια Π.Θ.

Μποζάνης Παναγιώτης
Αναπληρωτής Καθηγητής Π.Θ.

Εγκρίθηκε από την διμελή εξεταστική επιτροπή την 2^{α} Μαρτίου 2012

(Υπογραφή)	*(Υπογραφή)*
…..... ΤΣΟΜΠΑΝΟΠΟΥΛΟΥ ΠΑΝΑΓΙΩΤΑ Επίκουρη Καθηγήτρια Π.Θ.	..... ΜΠΟΖΑΝΗΣ ΠΑΝΑΓΙΩΤΗΣ Αναπληρωτής Καθηγητής Π.Θ.

Βόλος, Μάρτιος 2012

(Υπογραφή)

.....

ΜΠΕΛΗΓΙΑΝΝΗ ΦΩΤΕΙΝΗ

Διπλωματούχος Μηχανικός Ηλεκτρονικών Υπολογιστών, Τηλεπικοινωνιών και Δικτύων Πανεπιστημίου Θεσσαλίας

Ευχαριστίες

Σε αυτό το σημείο θα ήθελα να πω ένα μεγάλο ευχαριστώ στους γονείς μου, Γιώργο και Κατερίνα, που όλα αυτά τα χρόνια όχι απλά δεν μου στέρησαν τίποτα, αλλά μου έδωσαν και πολλά περισσότερα από όσα μπορούσα να ζητήσω.
Ακόμη, θέλω να ευχαριστήσω την κα Τσομπανοπούλου για τη βοήθεια και την υποστήριξη της όλους αυτούς τους μήνες.
Πώς να μην ευχαριστήσω τους φίλους μου για όλες αυτές τις στιγμές, που αν και τώρα στο τέλος αυτής της πορείας με γεμίζουν μελαγχολία, μου αποδεικνύουν πόσο σημαντικοί είναι στη ζωή μου.
Τέλος, ευχαριστώ το Νικόλα Φραγγογιάννη για τις εποικοδομητικές μας συζητήσεις που με βοήθησαν κατά την υλοποίηση αυτής της διπλωματικής.

Περίληψη

Η παρούσα διπλωματική έχει ως στόχο την ανάπτυξη μίας εφαρμογής για έξυπνα κινητά τηλέφωνα και συγκεκριμένα για τηλέφωνα με λογισμικό Android, για την καταγραφή και μελέτη της κίνησης του χρήστη σε εξωτερικούς χώρους. Η εφαρμογή RunTracker που υλοποιήθηκε επιτρέπει στον χρήστη να καταγράφει την πορεία του μέσω GPS σημάτων και να βλέπει στιγμιαίες μετρήσεις, όπως διάρκεια, θερμίδες, στιγμιαία ταχύτητα και απόσταση, καθώς αυτός ποδηλατεί, τρέχει ή συμμετέχει σε οποιαδήποτε εξωτερική δραστηριότητα. Αφού ο χρήστης καταγράψει και αποθηκεύσει την αθλητική του δραστηριότητα του, μπορεί να μοιραστεί τις μετρήσεις αυτής με τους φίλους του στο Facebook ή να την εξάγει στην SD κάρτα του κινητού του. Τέλος, η εφαρμογή RunTracker προσφέρει στον χρήστη τη δυνατότητα της απεικόνισης της διαδρομής και των μετρήσεων των αποθηκευμένων αθλητικών δραστηριοτήτων.

Η εφαρμογή RunTracker μπορεί να χρησιμοποιηθεί από όλους, αθλητές και μη αθλητές, για τη καταγραφή ενός ιστορικού με τις αθλητικές τους δραστηριότητες καθώς επίσης και για να τους βοηθήσει στη διατήρηση της φυσικής τους κατάστασης.

Abstract

The scope of this thesis is the development of an application on smart phones and especially on Android phones, for study of outdoor positioning and locomotion. The RunTracker application that has been implemented enables the user to record GPS tracks and view live statistics – such as time, calories, speed and distance – while biking, running or participating in other outdoor activities. Once recorded, the user can share workout data with his/her friends on Facebook or export his/her tracks to phone's SD card. Finally, RunTracker offers historic mapping and detailed information of previous sessions. RunTracker can be used by everyone, athletes and non-athletes, to keep a record of their exercise during time and help them maintain their shape and fitness.

Contents

1 Introduction

1.1 Smart phones and their use

Recent years have seen an increasing use of smart phones, which is due to the functionalities they provide, as they serve the growing needs of users. The different functionalities of smart phones are provided through the applications that are available depending on their mobile operating system.

The most common mobile operating systems (OS) used by modern smart phones include Google's Android, Apple iOS, Microsoft's Windows Phone etc.

Google's Android platform has grown tremendously in the latest years, as it is straightforward and uncomplicated to use it. The most vital feature of Android is its compatibility with different phones, which make it very helpful among the end users. There is a big variety of applications available for Android phones, which covers different categories such as Entertainment, Personalization, Lifestyle, Sports and Fitness, Education etc.

RunTracker is a fitness application for smart phones using Android OS devices.

In general, fitness applications help their user to keep track of his/her fitness activities while some help him/her to keep a closer look at calories gained or burned.

1.2 Thesis Structure

In the 2nd Chapter related works are presented and compared to RunTracker. Chapter 3 discusses about the required theoretical background. In Chapter 4 the system design

and architecture is described as well as the file formats that are used. Chapter 5 discusses the implementation details and presents the platforms and development software that were used. At Chapter 6 is presented a manual of RunTracker. Finally, Chapter 7 presents the conclusion of this work and directions for future work and Chapter 8 contains the bibliography.

2 Related Works

There are plenty free fitness applications in the Android Market that can track someone's exercise and gather useful statistics, that the user is able to see live or can save them for later use.

Two of them are:

- RunKeeper
- Keep Running

2.1 Tracking of Fitness Activities Functionality

RunKeeper just like RunTracker uses GPS to track the user's fitness activities and display his/her path on a map. RunKeeper has two different Layouts for the display of the map view and the activity's data, whereas RunTracker displays both the map view and the data on a single Layout.

Keep Running does not display the user’s route on a map. The user is only able to see his/her current speed, which can be calculated either by a pedometer or by the GPS and he/she is also able to listen to music, while he/she is running. Moreover, the user can set a certain speed limit that when the user's speed is under this limit, the music on his/her phone stops playing.

2.2 Calculated Fitness Data Functionality

Application Keep Running calculates only the user's pace (m/s), whereas RunTracker and RunKeeper calculate time, distance, speed and average speed and calories.

Moreover, RunKeeper is feasible to manage real time heart rate data straight from a heart rate monitor over Bluetooth Protocol.

2.3 Accessing Stored Activities Functionality

RunKeeper and RunTracker give the user the possibility to review his/her stored routes and corresponding metrics, with the difference that RunKeeper has two different Layouts for the map view and the metrics view, whereas RunTracker includes both views on a single Layout. Moreover, with RunKeeper the users can either delete a specific route or send it to his/her account on RunKeeper.com, whereas with RunTracker the user can either delete the route or extract it in GPX format on SD card.

Keep Running does not offer the user the functionality to store his/her fitness activities.

3 Theoretical Background

Android applications are written in the Java programming language. The Android SDK tools compile the code—along with any data and resource files—into an Android package, an archive file with an . apk suffix. All the code in a single . apk file is considered to be one application and is the file that Android-powered devices use, in order to install the application.

Application Components are the main components of all Android applications. There are four different types of application components:

1. Activities
2. Services
3. Content Providers
4. Broadcast Receivers

Each one of the above application components serves a distinct purpose and has a distinct lifecycle that defines how the component is created and destroyed.

RunTracker consists only of Activities.

The User Interface (UI) in an Android application is provided by a hierarchy of views. Android provides a number of ready-made views that an Android developer can use to design his/her application's UI, such as Widgets, Layouts, Dialogs and Menus. Moreover, an Android developer can create his/her own custom views, to use on his application UI.

One of the most important aspects of an Android application is the Manifest file. The Manifest file gives essential information to the Android System about the application.

Finally, since an Android application with UI is more than just source code, it cannot be completed without the application resources such as images, audio files, and anything relating to the visual presentation of the application.

3.1 Activities

An Activity is an Application Component that provides a user interface. Each activity draws its user interface on a window, with which users can interact in order to do something, such as listen to music, or dial a phone. The activity's window may either fill the screen, or may be smaller than the screen.

3.1.1 Activity's Lifecycle

Activities in the Android System are managed as an activity stack. When a new activity is started, it is placed on the top of the stack and becomes the running activity -the previous activity always remains below it in the stack, and will not come to the foreground again until the new activity exits.

To create an activity, we must create a subclass of Activity class (or one of its subclasses). In our subclass, we need to manage the Activity's lifecycle, by implementing certain callback methods that the system calls when the activity moves between various states of its lifecycle, such as when the activity is being created, stopped, resumed, or destroyed.

The following diagram (Figure 3.1.1-1) shows the important states of an Activity. The square rectangles represent callback methods. The colored ovals are major states that the Activity can be in:

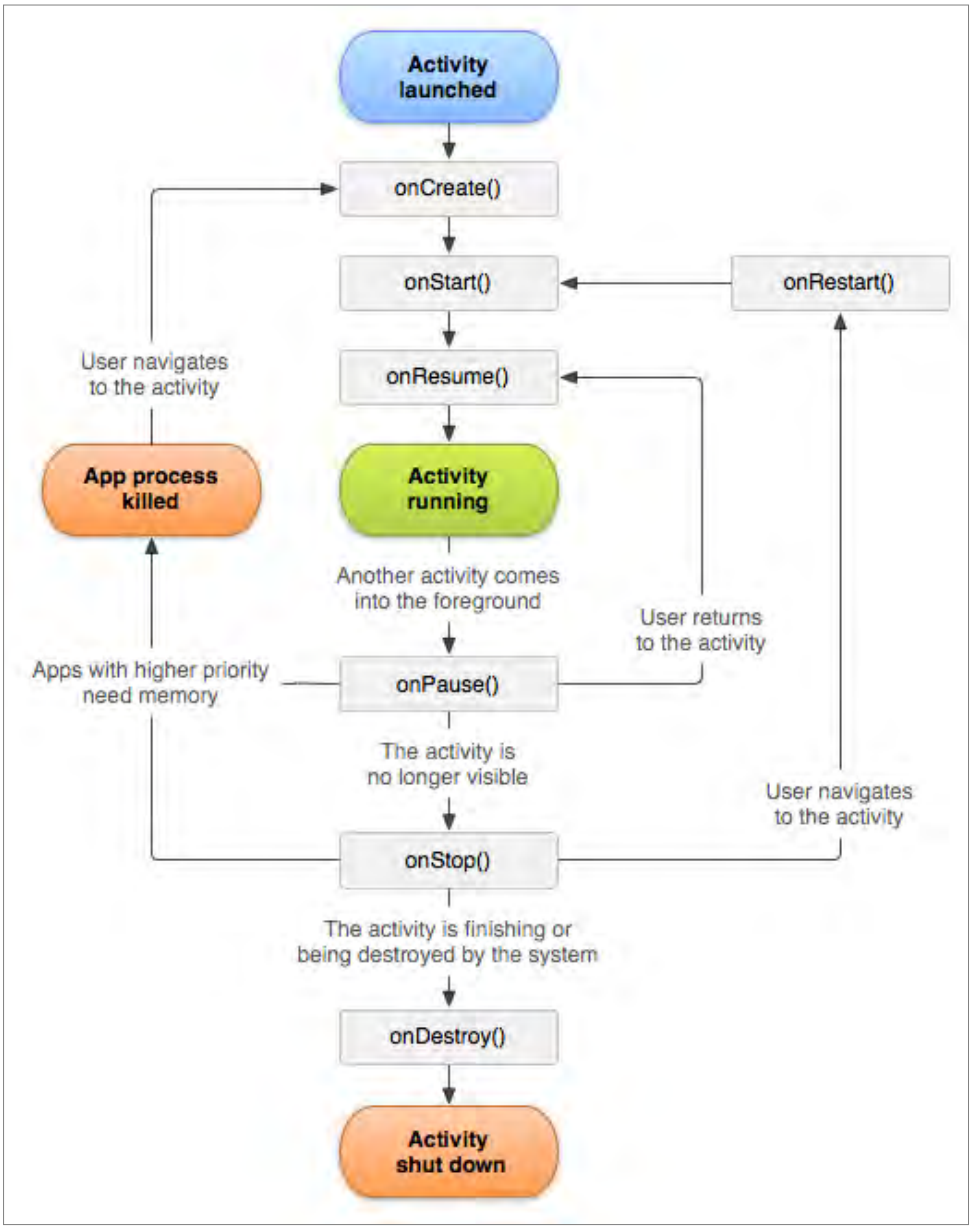

Figure 3.1.1-1: Lifecycle of an Activity [3]

The two most important callback methods are:

onCreate ()

The system calls this method when creating our activity. Here the essential components of the activity should be initialized and the hierarchy of views should be attached to the screen.

onPause ()

The system calls this method as the first indication that the user is leaving the activity (though it does not always mean that the activity is being destroyed). Here are usually stored any changes that should be persisted beyond the current session.

There are several other lifecycle callback methods that should be implemented in order to provide the best user experience between activities, and handle activities' state transitions.

3.1.2 Starting Activities

An Android application usually consists of multiple activities that are bound to each other. Typically, the first activity which is presented to the user when launching the application is specified as the "main" activity. Each activity can then start another activity in order to perform different actions. Every activity should be declared on the application's Manifest file.

3.1.3 Types of Activities

There are different types of Activities, depending on the functionality they implement. RunTracker uses the following Activities:

- Activity: displays the defined by the developer User Interface.
- MapActivity: displays a Map View in the application.
- ListActivity: displays in a List View a list of items and triggers event handlers when the user selects a list's item.
- TabActivity: displays a Tab View.

3.2 User Interface

The user interface is one of the most important aspects of an Android application, since is the only way for the user to interact with an application. The application's user interface is constructed by layouts, widgets, menus, dialogs and custom views, depending on the application’s needs.

3.2.1 Layouts

Layouts define the User Interface architecture of an Android application, with some of them providing their own form of a visible UI (ListView), whilst others being invisible structures that only manage the display of their child views (LinearLayout).

An application developer can specify a Layout, in two ways:

1. Declare the UI elements in XML code, or
2. Instantiate them at runtime.

Android SDK gives at the developer the flexibility to use each or both of the above ways to construct the application’s UI. For example, the developer could instantiate his/her application Layout and View elements in XML code and then change some of their properties at runtime.

3.2.2 Widgets

Widgets are views that provide visual and interactive elements for the UI, such as a button, a text-entry field, a checkbox or the zoom controls. A developer can either use an existing widget or can create his/her own custom widgets.

3.2.3 Menus

Menus are an important part of an activity's UI. Through Menus the user is able to access application's settings and provided functionalities. There are three types of application Menus in Android:

1. Options Menu
2. Context Menu
3. Submenu

In RunTracker we use only Options Menu, which is the menu type that appears when the Menu button on the phone is touched.

3.2.4 Dialogs

Dialogs are small windows that appear in front of the current Activity. The Activity looses focus and the user can interact only with the Dialog. Dialogs are usually used either when the user should be interrupted in order to be notified about something, or when a short task should be performed, such as a prompt for a file name, which we want to store. There are several types of Dialogs, depending on the functionality they implement. RunTracker uses the following types:

- AlertDialogs and
- Custom Dialogs.

3.3 The Manifest File

All Android applications must include in their root directory the Manifest file (AndroidManifest.xml). This file gives important information to the system about the application, information that is required for the application to be executed into our system. Specifically, in the Manifest file we must declare:

- the application's components

- the minimum version of Android required to run the application
- all the application's requirements
- user permissions, such as if the application needs to have internet or GPS access
- libraries that the application will use, like Google Maps library.

If something is not declared into an application's Manifest file, the application cannot use it, which means that the application will crash

4 System Design

This chapter analyzes the system's design and software architecture.

4.1 Architecture

RunTracker consists of 5 Activities:

1. MyTabsActivity: creates a tabbed UI
2. GpsSignalActivity: appears at the launching of RunTracker and informs the user about the GPS signal.
3. ListFilesActivity: displays the list of stored files.
4. PreviousMapActivity: displays a map view with one or more PathOverlay that correspond to a stored route.
5. NewTrackActivity: displays a map view, the user's route and current position and real time information about the user's fitness activity.

Except of the Activities, there are also four classes for the overlays which are used on the maps:

1. MyBalloonOverlay
2. PathOverlay: draws an overlay that represents a stored track.
3. MyLocatOverlay: displays an overlay that shows user's route and current position.
4. RouteLineOverlay: creates an overlay that displays a path on the map.

For writing or reading to/from files, there are two classes:

1. ExternalFileHandler: contains methods for writing to SD card and obtaining user's privileges (read, write, read & write) to it.
2. InternalFileHandler: includes methods for writing or reading files to/from internal storage space.

Lastly, there is one more class:

- BalloonOverlayView: is used by MyBalloonOverlay class to design the balloon overlays.

Both of these classes, MyBalloonOverlay and BalloonOverlayView were obtained from GitHub, a web-based hosting service for software development [2].

4.2 Description of Classes

4.2.1 *My*TabsActivity extends TabActivity

MyTabsActivity is the main activity of RunTracker, as we have defined it on the Manifest file, which means that it is the first activity that is created when RunTracker is launched. Here we define that RunTracker has two tabs, the:

1. "New Route" tab: hosts GpsSignalActivity and is the preselected tab
2. "Stored Routes" tab: hosts ListFilesActivity

4.2.2 GpsSignalActivity extends Activity

GpsSignalActivity is the Activity that is hosted in the preselected tab of MyTabsActivity, which means that the User Interface of this activity appears to the user when RunTracker launches. This Activity has a simple UI that is consisted of:

- an Image View
- a Text View and
- a Button.

Depending on the current state (disabled, acquiring satellites, enabled) of the GPS provider, which is obtained by a LocationListener, all of the above views change color (red, orange, green), text and states (disabled, enabled) correspondingly.

Moreover, GpsSignalActivity implements GpsStatus.Listener, which receives notifications when GPS system's status change (stopped, started, got its first fix, etc) and changes the appearance of the Image View, the text View and the Button correspondingly.

Lastly, we have created a custom Dialog that appears when the application is launched, to notify the user if the GPS is disabled. The user can either be redirected to enable the GPS module or can close the dialog.

4.2.3 NewTrackActivity extends MapActivity

NewTrackActivity displays a Map View with its Zoom Controls, a Start/Stop button and two Table Layouts. The map view holds the 4/5 of the phone's screen, whereas the button and the two Table Layouts share the other 1/5 of the window. We set the screen to be turned on and bright as long as the window of the NewTrackActivity is visible to the user, and we initialize:

- **the Map View:** gets the map controller in order to manage panning and zooming of the map. Then, a new MyLocatOverlay is declared and the Zoom Control listeners are implemented.
- **the Table Layouts:** each one of the first rows of the two Table Layouts contains, two Text Views that are static. In the second row of the first TableLayout there are a Chronometer and a Text View which change dynamically, and in the second row of the second TableLayout there are two Text Views that change dynamically too.

- **the Start/Stop button:** is placed between the two Table Layouts. This button controls the calculation of metrics and is either green and has its text set to "START" or is red and its text is set to "STOP".

Moreover, a handler is declared which receives from MyLocatOverlay messages that contain the user's speed and covered distance and posts them on the corresponding cells of the activity's Table Layouts.

When the user launches the application, a custom dialog appears and asks him/her to give some information, if he/she has not already submitted it. The fields of gender and weight are required to be filled in order to calculate the spent calories.

NewTrackActivity has an Options Menu, which is displayed when the user touches the device's Menu button. Through the Options Menu the follow functionalities are provided to the user:

- **Set Info**: the user can set or change (if has already set it) the information (gender, age, weight) that the application asks about him.
- **Save Route**: the user can save his/her route on the internal storage either with the proposed by the application name or with a name of his preference. If the user has not pressed the Stop button then, before the route's storage, the button's color change to green, its text changes to Start and the GPS and Acceleration events are discarded.
- **Upload Metrics**: the user can upload his/her route data to his/her profile to Facebook, after he/she has saved the route.
- **Extract to SD**: if the user has stored the route on internal storage space, he/she can extract the route on directory RunTracker on SD card.

The last two options (upload metrics, extract to SD) are disabled if the user has not stored the route to phone's internal storage.

Additionally, in this class we implement four dialogs:

1. **User Settings Dialog:** a custom dialog that represents a form which is displayed at the launching of the application or when the user selects the "set user info" item of Options Menu.
2. **Alert Dialog:** appears if the user tries to submit the information on User Settings Dialog without having filled the required fields (gender and weight).

Alert Dialog displays a message which notifies the user that in order to submit the information he has to fill both fields.

3. **Save Dialog:** a custom dialog that appears when the user selects the item "Save Route" of Options Menu. This dialog's layout is consisted of a text field, with a proposed by the application name that has the format: YYYY_MM_DD_HH_mm, which corresponds to the current date and time, and two buttons:
   - **Save:** saves the route's metrics and the track as txt file on file "routes" of internal storage.
   - **Cancel:** cancels the storing.

   The user can choose to save the route with a name of his preference, different from the proposed.
4. **Extract File Dialog:** an Alert Dialog which appears when the user selects the "Extract Route" item of the Options Menu. This Dialog displays a list of the names of all stored routes. The user can select a route to extract it on RunTracker directory on the SD card as a GPX file.
5. **Back Button Dialog:** an Alert Dialog that appears if the user presses the Back button of his phone. Here the user is asked to confirm if he/she really wants to exit the application and discard the current session.

Lastly, here are calculated the user's burned calories by the changes of acceleration, obtained from the phone's Accelerometer. In order to calculate the user's burned calories, we have used the non-linear model developed by Kong Y. Chen and Ming Sun [1].

4.2.4 MyLocatOverlay extends MyLocationOverlay

MyLocatOverlay extends MyLocationOverlay class, that draws the user's current location on the map and implements LocationListener in order to receive notification from the GPS provider, when the user's location changes.

Specifically, if the user has pressed the START button, except from the user's current location, a red line is displayed that connects his/her previous location point with his/her current location point on the map. Moreover, the distance that the user has covered is calculated (in km) and his/her current speed is obtained from the GPS signal (and converted in km/h). Speed and distance are sent on NewTrackActivity in order to be displayed on its Table Layout. If the user has pressed STOP, then every new GPS location fix is discarded.

Lastly, if the user taps on the location overlay, then a MyBalloonOverlay is created to display the address or the geographical coordinates, of the current location.

4.2.5 RouteLineOverlay extends ItemizedOverlay<OverlayItem>

When the user presses the STOP button on NewTrackActivity, then a RouteLineOverlay is created and displayed which represents the route that the user has done until that moment.

4.2.6 ListFilesActivity extends ListActivity

ListFilesActivity shows a list view that contains as items the names of all routes which are stored in the application's internal storage. In case that there are no stored routes, the list view is replaced from a single notification message.

When the user chooses a route, then its filename is passed on PreviousMapActivity that is launched in order to display the stored route.

4.2.7 PathOverlay extends Overlay

PathOverlay represents a custom overlay which may be drawn on a map. PathOverlay displays on the map a user’s stored track as a green line.

4.2.8 PreviousMapActivity extends MapActivity

PreviousMapActivity is launched from ListFilesActivity. PreviousMapActivity displays on a Map View the chosen route as one or more PathOverlays and information (duration, distance, average speed, spent calories) about the corresponding route.

PreviousMapActivity layout consists of a Map View with its zoom controls and two Table Layouts. The Map View holds the 4/5 of the phone's screen, whereas the Table Layouts hold the other 1/5 of the screen.

Firstly, the Zoom Control listeners are implemented to zoom in or out depending on the button (-/+) that the user presses. Then, the information about the specific route is displayed on the two Table Layouts with a simple view that represents a green line separating the two Table Layouts.

Subsequently, the file that the user chose is read and the corresponding PathOverlays are created. Then, it adjusts the zoom levels and sets the screen's center in order all the PathOverlays to be displayed on the Map View's center.

Additionally, PreviousMapActivity has an Options Menu that supplies the user with the following functionalities:

- Extract Route: extracts the specific route to SD card as a GPX file.
- Delete Route: deletes the specific route from application's internal storage.

4.2.9 ExternalFileHandler

ExternalFileHandler is the class that interacts with the phone's SD card. Here we implement two methods that relate to the user's privileges and the storage of files on the SD card. ExternalFileHandler implementation gives the functionality of the extraction of the stored routes from the application's internal storage space to the phone's SD card.

4.2.10 InternalFileHandler

InternalFileHandler is the class that interacts with the application's internal storage. In order to read or write on application's internal storage, we have to use one of the methods of the InternalFileHandler class.

4.3 File Formats

RunTracker uses two different types of files for storing and accessing fitness data, txt and gpx files. Plain text format is used for storing and reading activities' data, whereas GPX format, based on XML schema, is used when the user wants to extract his/her route, in order to use it on any other compatible application (e.g. Google Earth, Figure 4.3-1).

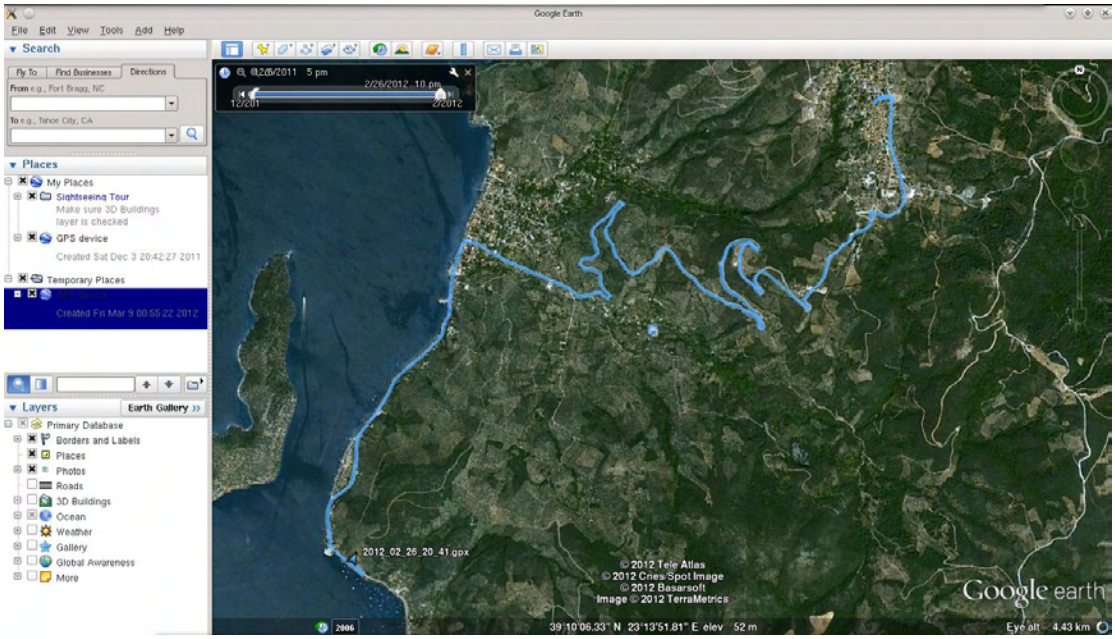

Figure 4.3-1- A RunTracker's extracted gpx file, as displayed on Google Earth.

Specifically, when a user chooses to save a fitness activity, then a txt file is created on "routes" directory on internal storage, where fitness' data are stored. The file contains information about the activity's statistics (duration, distance, average speed and

calories) and about the user's route obtained from GPS (latitude, longitude, altitude, speed and time).

The stored txt files can be exported to GPX format files for later use. GPX is a light-weight XML data format for the interchange of GPS data (waypoints, routes, and tracks) between applications and Web services on the Internet [3].

RunTracker uses internally plain text files, instead of the large gpx, in order to avoid the overhead of parsing XML schema. Specifically, the estimated time for parsing a gpx file was nearly two times the time needed for processing a txt file, which contains the same route information. Moreover the size of a gpx file is almost twice the size of the corresponding txt file, due to the tags that the XML schema contains.

5 Implementation

5.1 Implementation details

5.1.1 Implementation of start/stop/resume functionality

NewTrackActivity has a START/STOP button, from which the user is able to control when the application will take into account, for the statistics, the GPS and Acceleration Events. Specifically,

- When the user presses “START” then the chronometer starts and the dynamically changeable rows of the Table Layouts are refreshed, if new GPS signals are obtained. Additionally, all the Acceleration events that are received are taken into account, for the calculation of the calories, and MyLocatOverlay is notified that it has to record the user's locations. Lastly, the button's color changes to red and its text changes to “STOP”.
- When the user presses “STOP” then the chronometer stops and MyLocatOverlay is notified to discard new Location changes. Additionally, a new RouteLineOverlay, is created and displayed, the corresponding Location objects are stored on temp.txt file on internal storage and cleared from phone's memory. Lastly, the button's color becomes red and its text is set to “START”.

5.1.2 Uploading route's data to Facebook

RunTracker provides the user the functionality to upload the route's data to Facebook, as displayed in Figure 5.1.2-1.

Once the user has saved the route then the menu item “Upload Data to Fb” is enabled and gives the user the possibility to upload the entire route's data on Facebook. The user is asked to give username and password, in order to log in to Facebook, whereas if the Facebook Application is installed, then the login data are obtained from it.

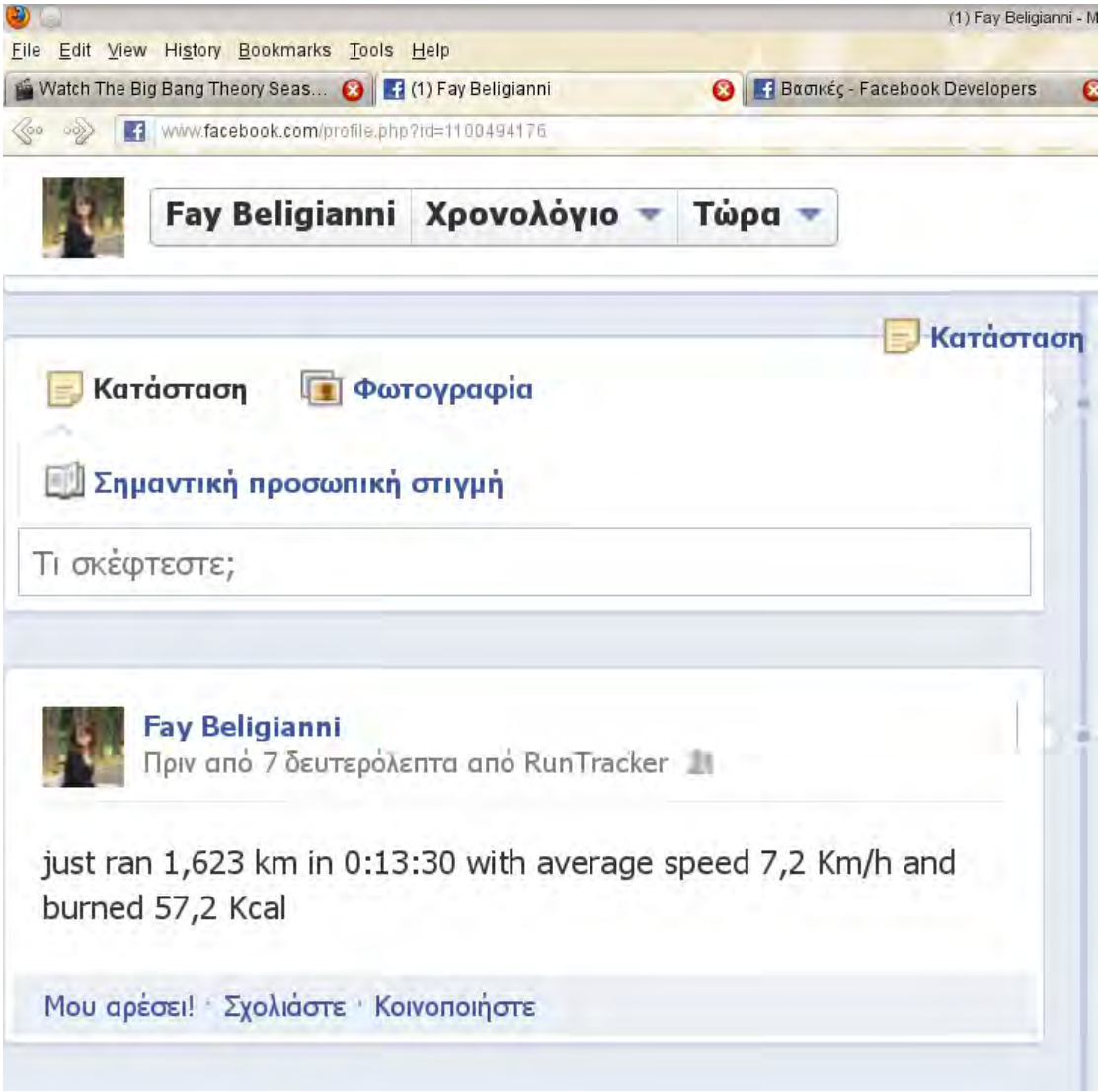

Figure 5.1.2-1 – RunTracker’s uploaded Facebook message.

5.1.3 Calculating calories

After the user has pressed the START button and only if he has provided the application with the necessary information (gender and weight), then by using the acceleration changes and the equations established by Kong Y. Chen and Ming Sun [1], we calculate the calories that the user burns throughout his/her activity.

5.1.4 Definition of zoom controls

In order the zoom controls to be always visible and not to fade in and out, we have defined our own zoom controls. Specifically, when the user presses the zoom in button (+), the user's current location is set as centric point of the map and then zooms in. When the zoom out button (-) is pressed then it just zooms out.

5.1.5 Displaying a stored route

In order a stored route to be displayed properly, except from the PathOverlays that we create, it is also essential to calculate:

- the correct zoom level and
- the center point of our map view.

Specifically, when we read the file with the stored route, in order to display it, we calculate the minimum and maximum values of the latitude and the longitude as well as the mean value for each one of them. Then we use the range between the maximum and the minimum values to set the zoom level and the mean values as the center point of the map.

5.2 Platforms and Development Software

RunTracker was developed on a personal computer running OpenSuse 12.1 64-bit Linux operating system, where the following open source development tools were used:

- Eclipse Indigo (3.7) IDE
- Open JDK 1.6.0
- Android SDK
- Facebook SDK
- Android 2.3.3 platform

Furthermore, the following APIs and plug-ins for the Eclipse IDE were used:

- Google API
- Android Development Tools (ADT)
- an Android Virtual Device (AVD)

6 User Guide

6.1 Application's launching

When the user launches RunTracker he/she is asked to enable the GPS, as depicted in Figure 6.1-1.

Figure 6.1-1 – Launching RunTracker

In case the user presses the "Cancel" button, he can access only the stored routes through the "Stored Routes" application's tab as shown in Figure 6.1-2.

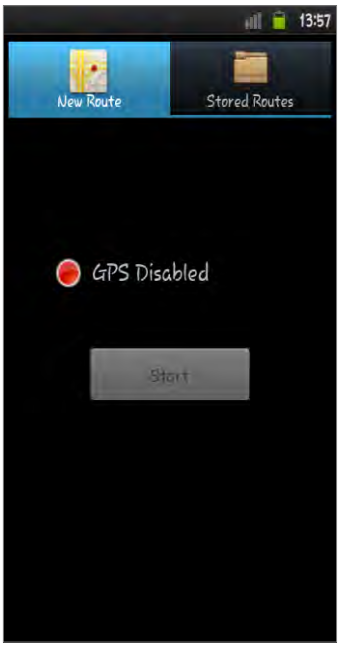

Figure 6.1-2 – GPS disabled

If the user enables the GPS, through GPS Settings (Figure 6.1-3), he/she can start tracking a new route after the GPS has acquired satellites as shown in Figure 6.1-4.

Figure 6.1-3 – GPS Enabled

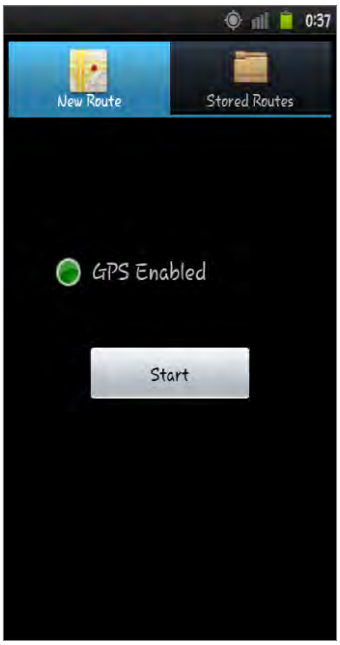

Figure 6.1-4 – GPS signal obtained

6.2 Tracking a new route

Before the user starts tracking a route, RunTracker asks from him/her to fill some personal information if he/she has not already submitted it, as displayed in Figure 6.2-1.

Figure 6.2-1 – User's profile form

In case that the user tries to submit the form without having filled the required fields, gender and weight, then a notification appears (Figure 6.2-2).

Figure 6.2-2 – Submitting profile notification error

When the user presses the START button, then his route and current location is displayed on the map. Moreover, the calculation of the metrics starts and the button's color changes to red and its text to STOP as depicted in Figure 6.2-3. In case that the user presses Stop, his/her location points are not taken into consideration neither for the calculation of metrics nor during the draw of the user's route (Figure 6.2-4).

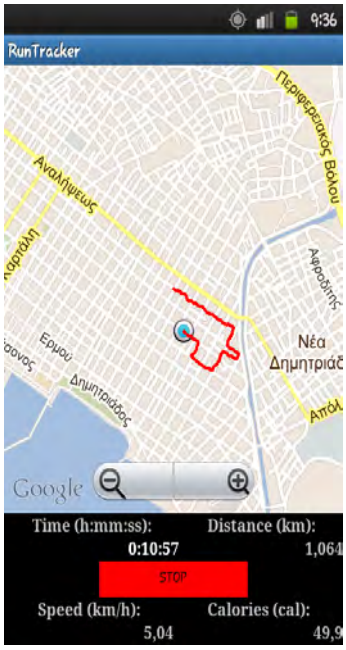

Figure 6.2-3 - Recording a route.

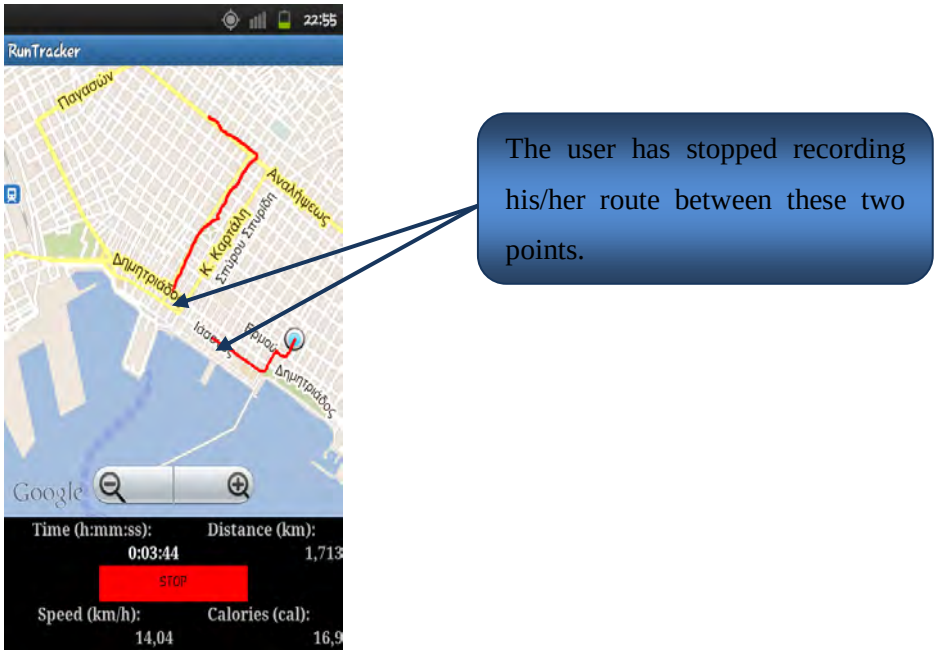

Figure 6.2-4 – Recording a route with a pause.

When the user presses the back button of his phone, he is asked to confirm that he wants to exit the application, as shown in Figure 6.2-5.

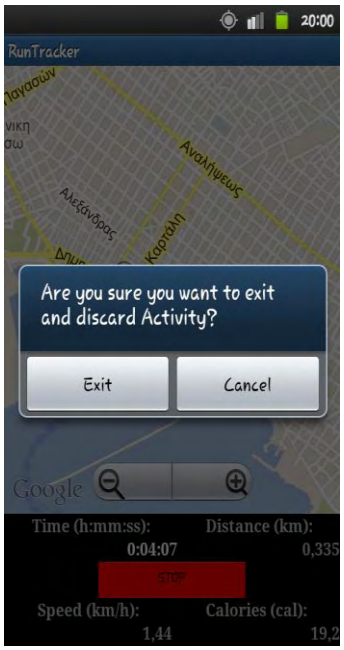

Figure 6.2-5 – Exit activity notification.

If the user has not stored his route, when he/she presses the phone's MENU button, he is able to change the user's profile or to store his route (Figure 6.2-6).

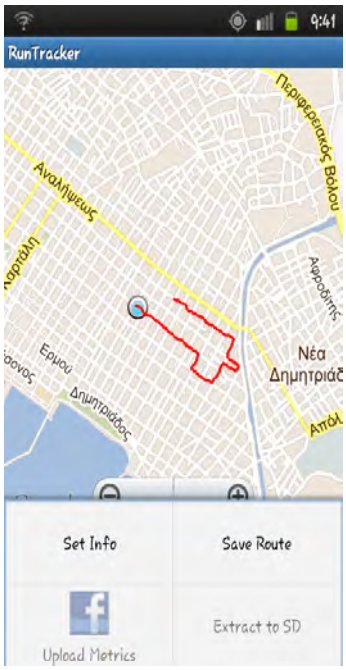

Figure 6.2-6 – Menu options for the recorded route.

If the user chooses to save his/her root, the proposed be the application name for the file to be saved has the format: DATE_HOUR and specifically YYYY_MM_DD_HH_mm (year_month_day_hour_minute), as displayed on Figure 6.2-7.

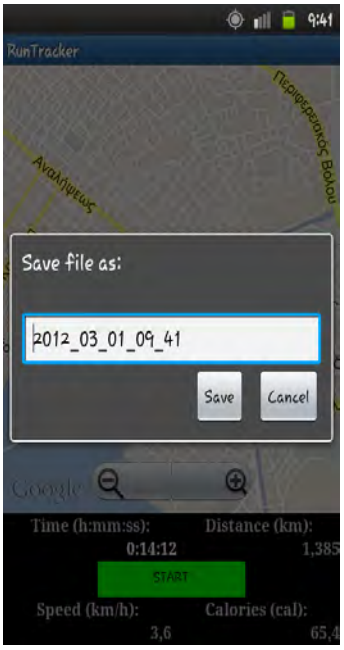

Figure 6.2-7 – Save route dialog.

After the user has saved his/her route, by pressing the phone's MENU button, he/she is able to upload the route's metrics on Facebook and he can extract his route as a GPX file on phone's SD card, as shown in Figure 6.2-8.

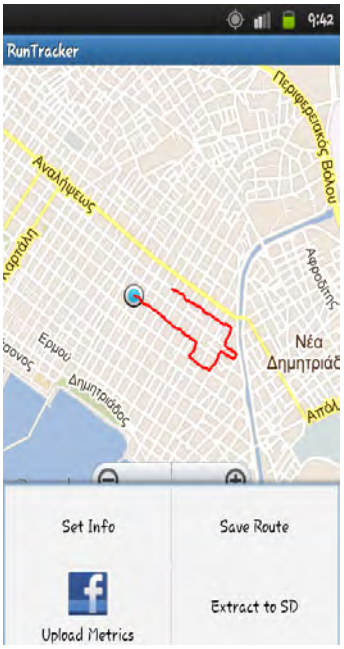

Figure 6.2-8 – Menu options after saving route.

6.3 Display of stored routes

The user can access his/her stored routes from the "Stored Routes" tab of the RunTracker application. In case that there are no stored routes, then appears only a notification at the "Stored Routes" tab (Figure 6.3-1).

Figure 6.3-1 – No existence of stored routes.

If there are stored routes on the application's internal storage space, then the names of all the stored routes are displayed on the "Stored Routes" tab and the user can choose anyone of them in order to display the corresponding route and data.

Figure 6.3-2 – Viewing list of user's stored routes.

When the user chooses one of the routes, then the path of the chosen route is drawn on the map, and its data are displayed, as depicted on Figure 6.3-3.

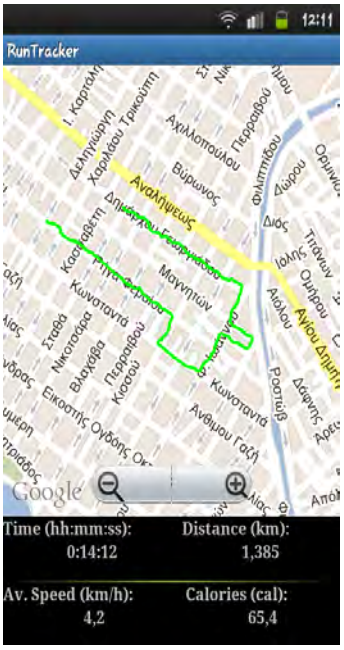

Figure 6.3-3 – Display of the selected route.

In this case, by pressing the MENU button, the user can delete the specific route or extract it on phone's SD card as a GPX file (Figure 6.3-4).

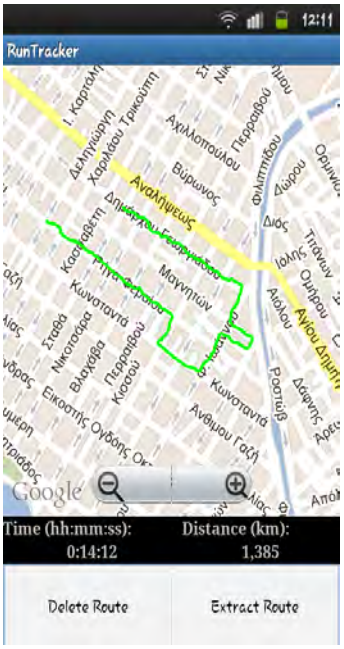

Figure 6.3-4 – Menu options for the stored routes.

7 Conclusion

In this final section the functionalities of RunTracker are restated and potential future work is discussed.

7.1 Conclusion

Within this thesis the Android application RunTracker was implemented. Specifically, was implemented an Android application that:

- enables its user to record and display on a map his/her path through GPS signals
- studies the user's movement and calculates metrics, such as:
  - ❖ duration
  - ❖ distance
  - ❖ average and instant speed
  - ❖ calories
- historic mapping and view of details of his/her routes

7.2 Future Work

An enhancement of this work could be the design of charts of elevation and speed, as well as the use of heart rate, obtained by a heart rate monitor, for a more accurate calculation of calories consumption during exercise.

8 Bibliography

[1] Kong Y. Chen and Ming Sun. Improving energy expenditure estimation by using a triaxial accelerometer. J Appl Physiol 83:2112, 1997.

[2] https://github.com/jgilfelt/android-mapviewballoons

[3] http://developer.android.com/reference/android/app/Activity.html