Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτη

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Μηχανικών Πληροφορικής

Πτυχιακή Εργασία

Τίτλος: **Έξυπνος χάρτης πληροφοριών για άτομα με κινητικά προβλήματα. Πιλοτική εφαρμογή στην πόλη των Χανίων.**

Τούζου Κυριάκος-Αλέξανδρος (ΑΜ:2940)
Μπουζάκης Ιωάννης-Γεώργιος (ΑΜ:3411)

Επιβλέπων καθηγητής: ***Μαλάμος Αθανάσιος***

Επιτροπή Αξιολόγησης: ***Μαλάμος Αθανάσιος, Ιωάννης Παχουλάκης***

Ημερομηνία Παρουσίασης: ***18/01/2016***

Ευχαριστίες

Με την εκπνοή της πτυχιακής μας εργασίας θα θέλαμε να ευχαριστήσουμε τον επιβλέπων καθηγητή κ. Μαλάμο Αθανάσιο για την πολύτιμη βοήθειά του καθώς και τα οικογενειακά και φιλικά μας πρόσωπα για την στήριξή τους όλο αυτό το διάστημα στην ολοκλήρωση της πτυχιακής εργασίας μας.

Abstract

The purpose of the thesis is the development of application supported by smart phones and tablets, in the environment of the Android operating system.

This application is addressed and concerns on safety guidance and movement of people with physical disabilities in the city center of Chania. The user is able to select between two maps, the beginning and the end of his journey and with the help of smart map the most accessible path will display automatically. Except of the safe user guidance with disability trolley, the application allows viewing of available car parking spots for people with disabilities, available ramps for easy user access to his destination and other smart features that serve him.

For the implementation of the application, using the application development program Android Studio which formation use the programming language Java, XML, and many of the libraries of the SDK (Software Development Kit) or tools such as an emulator, AVD (Android Virtual Device). Finally we used the GPS technology (Global Positioning System) and the Dijkstra's algorythm finding shorter routes node basis.

ΠΕΡΙΕΧΟΜΕΝΑ

Table of Contents

1 Εισαγωγή

1.1 Περίληψη

Σκοπός της πτυχιακής εργασίας είναι η ανάπτυξη μιας εφαρμογής που υποστηρίζεται από κινητά τηλέφωνα νέας γενιάς και tablets, μέσα στο περιβάλλον του λειτουργικού συστήματος Android.

Η συγκεκριμένη εφαρμογή απευθύνεται και αφορά στην ασφαλή καθοδήγηση και μετακίνηση των ανθρώπων με κινητικές αναπηρίες στο κέντρο της πόλης των Χανίων. Ο χρήστης έχει τη δυνατότητα να επιλέξει ανάμεσα σε δυο χάρτες, την αρχή και το τέλος της διαδρομής του και με τη βοήθεια του έξυπνου χάρτη θα προβληθεί η πιο κατάλληλη διαδρομή σε αυτόν. Επιπλέον, εκτός από την ασφαλή καθοδήγηση του χρήστη με το αναπηρικό του αμαξίδιο, η εφαρμογή επιτρέπει την προβολή των διαθέσιμων σημείων στάθμευσης αυτοκινήτων για άτομα με αναπηρία, τις διαθέσιμες ράμπες για εύκολη πρόσβαση του χρήστη στον προορισμό του και άλλες έξυπνες λειτουργίες που τον εξυπηρετούν.

Για την υλοποίηση της εφαρμογής χρησιμοποιείται το πρόγραμμα ανάπτυξης εφαρμογών Android Studio το οποίο χρησιμοποιεί τη γλώσσα προγραμματισμού Java, XML, καθώς και πολλές από τις βιβλιοθήκες του SDK (Software Development Kit) ή εργαλεία όπως ένα εξομοιωτή (Emulator), AVD (Android Virtual Device). Τέλος χρησιμοποιήθηκε η τεχνολογία GPS (Global Positioning System) όπως και ο αλγόριθμος εύρεσης συντομότερων διαδρομών βάση κόμβων Dijkstra.

1.2 Κίνητρο για τη Διεξαγωγή της Εργασίας

Κάποιοι άνθρωποι είναι περισσότερο ευαισθητοποιημένοι, κάποιοι λιγότερο και κάποιοι αδιαφορούν, πάντως όλοι γνωρίζουμε πως υπάρχει μια κατηγορία ανθρώπων γύρω μας τους οποίους αποκαλούμε « Άτομα με κινητικές αναπηρίες » και με τους οποίους συνυπάρχουμε ακόμα και στην ίδια πόλη. Ευτυχώς στις μέρες μας, γίνεται ότι το δυνατόν καλύτερο ώστε αυτοί οι άνθρωποι να ζούνε και να μετακινούνται με περισσότερη άνεση και ασφάλεια. Την ευθύνη για την αναβάθμιση της ποιότητας ζωής αυτών των ανθρώπων έχει ξεκάθαρα ο τομέας της τεχνολογίας. Έτσι λοιπόν η παρούσα πτυχιακή εργασία μέσω της εφαρμογής του έξυπνου χάρτη, στοχεύει στη εξυπηρέτηση και καθοδήγηση στη μετακίνηση των ανθρώπων αυτών με βάση τη διευκόλυνση και την ασφάλειά τους.

Εικόνα 1: Απεικόνιση ανθρώπων με κινητικές αναπηρίες

1.3 Σκοπός και Στόχοι Εργασίας

Ο σκοπός της συγκεκριμένης πτυχιακής εργασίας όπως προαναφέραμε είναι η ανάπτυξη και υλοποίηση μιας εφαρμογής βασισμένη στην ελεύθερη και ανοιχτή πλατφόρμα Android με τη βοήθεια των χρήσιμων εργαλείων της. Αρχικά όμως ας δούμε κάποια βασικά και εισαγωγικά πράγματα τα οποία αφορούν αυτή την πολλά υποσχόμενη λειτουργική πλατφόρμα.

1.3.1 Τι είναι το Android

Η λέξη "Android" αναφέρεται συγκεκριμένα σε ένα λειτουργικό σύστημα για κινητά τηλέφωνα με οθόνη αφής και όχι μόνο (βασισμένο στον πυρήνα του λειτουργικού Linux) το οποίο αναπτύχθηκε για πρώτη φορά από την Google. Είναι λογισμικό ανοιχτού κώδικα, που πάει να πει ότι οποιοσδήποτε μπορεί να πάρει τον πηγαίο κώδικα και να τον χρησιμοποιήσει, να τον τροποποιήσει ή ακόμα και να τον εξελίξει. Αφού λοιπόν είναι πάντα διαθέσιμο, κατασκευαστές λογισμικού όπως η Motorola, η HTC και η LG παίρνουν τον κώδικά του και το χρησιμοποιούν σαν μια βάση για να κατασκευάσουν πάνω του τις δικές τους ιδιόκτητες υλοποιήσεις που τρέχουν στα κινητά τους τηλέφωνα.

Εικόνα 2: Λογότυπο Android

1.3.2 Ιστορική αναδρομή

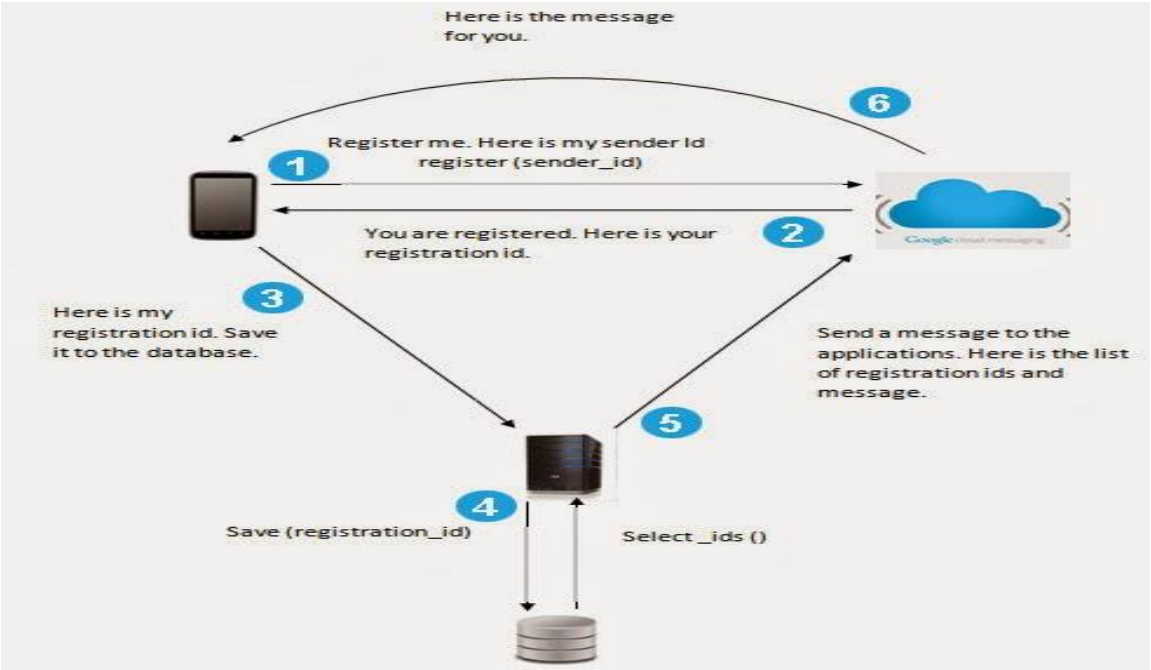
Όλα ξεκίνησαν την άνοιξη του 2005 από μια ιδέα του Andy Rubin για τη δημιουργία ενός δυναμικού και ανοικτού παγκόσμιου λειτουργικού συστήματος το οποίο θα άλλαζε ριζικά τα δεδομένα στη χρησιμότητα των κινητών τηλεφώνων. Αυτή την ιδέα την οποία ονόμασε "Android Inc" θέλησε να τη μοιραστεί με ένα από τους ιδρυτές της εταιρίας Google, τον Larry Page, ο οποίος σύντομα έγινε και ο εκπρόσωπος του λειτουργικού συστήματος Android. Έτσι, μετά από λίγο καιρό αρκετές εταιρίες ένωσαν τις δυνάμεις τους με την Google με σκοπό τη δημιουργία μιας πλατφόρμας ανοικτού κώδικα, θέτοντας τον πήχη του ανταγωνισμού στα ύψη. Αυτός ο οργανισμός που δημιουργήθηκε ονομάστηκε "Open Handset Alliance" και συμπεριλαμβάνονται δεκάδες εταιρίες του τεχνολογικού κόσμου μέχρι σήμερα όπως για παράδειγμα η HTC και η LG.

Εικόνα 3: Εταιρίες του οργανισμού Open Handset Alliance

1.3.3 Τα Χαρακτηριστικά του Android

- Η πλατφόρμα παρέχει μεγάλο εύρος δυνατοτήτων στον τομέα ανάλυσης της οθόνης, υποστηρίζοντας παράλληλα πολυμέσα ήχου και στατικής – κινούμενης εικόνας όπως H.263, H.264, MPEG-4, MP3, MIDI, JPEG, PNG, GIF κ.α.
- Αποθήκευση των δεδομένων με τη χρήση της σχεσιακής βάσης δεδομένων SQLite.
- Το Android υποστηρίζει λογισμικά γραμμένα στη προγραμματιστική γλώσσα JAVA, τα οποία τρέχουν πάνω στην εικονική μηχανή Dalvik, μια βελτιστοποιημένη μηχανή για φορητές συσκευές περιορισμένης ισχύος όπως τα κινητά τηλέφωνα.
- Περιέχει ολοκληρωμένο πρόγραμμα περιήγησης βασισμένο στην ανοιχτή τεχνολογία Web Kit και υποστηρίζει αρκετές μορφές συνδεσιμότητας όπως GSM, EDGE, 3G, 4G, Wi-Fi και Bluetooth.
- Αλληλεπιδρά με διάφορες έξυπνες τεχνολογίες όπως κάμερες, οθόνες αφής, GPS κ.α.

- Επιτρέπει το κομμάτι της επικοινωνίας με την ανταλλαγή SMS & MMS μηνυμάτων, καθώς και το Android Cloud To Device Messaging, το οποίο είναι μια υπηρεσία που βοηθάει τους προγραμματιστές να στέλνουν μηνύματα στις εφαρμογές τους και κατ επέκταση στους χρήστες με τη μορφή ειδοποιήσεων.



Εικόνα 4: Android Cloud to Device Messaging

1.3.4 Οι εκδόσεις του Android

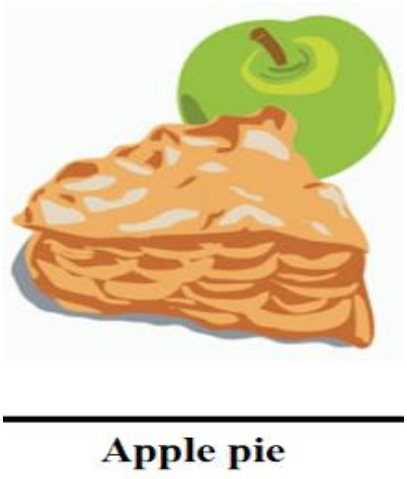
Το Android έχει κυκλοφορήσει αρκετές καινοτόμες εκδόσεις μέχρι σήμερα, τα ονόματα των οποίων σε παραπέμπουν περισσότερο στον τομέα της μαγειρικής παρά σε αυτόν της πληροφορικής. Από την πρώτη «πειραματική» έκδοση του Android Apple pie 1.0 που κυκλοφόρησε τον Σεπτέμβριο του 2008, χρειάστηκε περίπου μόλις ένας χρόνος για την κυκλοφορία των νέων πρωτοποριακών εκδόσεων που επέφεραν σημαντικές αλλαγές στον χρήστη πρώτο smartphone που λειτούργησε βασισμένο στην τεχνολογία του Android ήταν το T-Mobile G1 που κατασκευάστηκε από την HTC.

Εικόνα 5: Smartphone HTC Dream

1.3.4.1 Apple pie 1.0

Τα βασικά χαρακτηριστικά λοιπόν της πρώτης έκδοσης του Android είναι τα εξής:

- Δυνατότητα λήψης εφαρμογών και ενημερώσεων με τη βοήθεια του Market app.
- Υποστήριξη κάμερας, αν και η συγκεκριμένη έκδοση δεν έδινε τη δυνατότητα στο χρήστη να «πειράξει» την ανάλυση, την ισορροπία του λευκού και γενικότερα την ποιότητα της κάμερας.
- Επιτρέπεται η ομαδοποίηση των εικονιδίων των εφαρμογών μαζί, σε ένα ενιαίο εικονίδιο φακέλου στην αρχική οθόνη.
- Πρόσβαση σε διακομιστές ηλεκτρονικού ταχυδρομείου, που υποστηρίζει POP3, IMAP4 και SMTP και συγχρονισμός του Gmail με την εφαρμογή Gmail.
- Google search, επιτρέποντας στους χρήστες να κάνουν αναζήτηση στο Internet, στις εφαρμογές του τηλεφώνου, στις επαφές, το ημερολόγιο, κ.λπ.
- Media Player, καθιστώντας δυνατή τη διαχείριση, την εισαγωγή και αναπαραγωγή αρχείων πολυμέσων.
- Τηλεφωνητής επιτρέπει κλήση και διάθεση των τηλεφωνημάτων χωρίς να πληκτρολογήσετε ένα όνομα ή έναν αριθμό
- Υποστήριξη των υπηρεσιών Wi- fi και Bluetooth.
- Διαθέτει Google Maps, οπότε υπάρχει πρόσβαση σε χάρτες, δορυφορικές εικόνες και πραγματοποιείται αναγνώριση τρέχουσας τοποθεσία με τη βοήθεια της υπηρεσίας GPS.

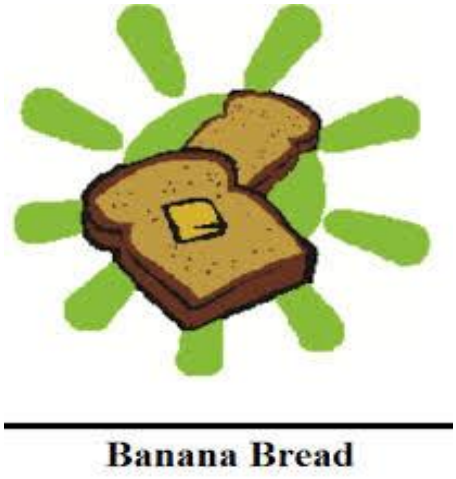


Εικόνα 6: Λογότυπο Apple pie

1.3.4.2 Banana Bread 1.1

Το Android Banana Bread 1.1 ήταν κυρίως ένα μικρό update που δημοσιεύτηκε το Φεβρουάριο του 2009, με σκοπό να επιλύσει τα σφάλματα που εντοπίστηκαν στο HTC Dream βάση της προηγούμενης έκδοσης. Έτσι, αυτή η ενημερωμένη έκδοση ήταν προσανατολισμένη αποκλειστικά στη βελτίωση - αλλαγή του API και την προσθήκη μιας σειράς νέων χαρακτηριστικών όπως:

- Προστέθηκαν λεπτομέρειες και σχόλια σχετικά με τα μέρη και τις τοποθεσίες στα Google Maps.
- Δόθηκε η δυνατότητα αποθήκευσης συνημμένων μηνυμάτων στο e-mail.
- Αναβαθμίστηκε η πλατφόρμα στον τομέα του Design.



Εικόνα 7: Λογότυπο Banana Bread

1.3.4.3 Cupcake 1.5

Η έκδοση αυτή έκανε την εμφάνισή της τον Μάιο του 2009 και ήταν η πρώτη έκδοση που είχε πραγματικά μέλλον. Μία καινοτομία που την ανέδειξε ήταν η εισαγωγή ψηφιακού πληκτρολογίου (όταν τα περισσότερα smartphones τότε είχαν φυσικό πληκτρολόγιο Qwerty) και κάποια ανανεωμένα - επιπρόσθετα χαρακτηριστικά όπως:

- Ταχύτερη εκκίνηση κάμερας – λήψη εικόνας και αρκετά ταχύτερη απόκτηση θέσης του GPS με την υποστήριξη του SUPL AGPS.
- Εισαγωγή μικροεφαρμογών widgets που «τρέχουν» στην αρχική οθόνη της συσκευής όπως αναλογικό ρολόι, ημερολόγιο, music player και αναζήτηση.
- Δυνατότητα βιντεοσκόπησης και αναπαραγωγής βίντεο υποστηρίζοντας μορφές όπως MPEG4 – 3GP.
- Υποστήριξη του Stereo Bluetooth (A2DP και AVCRP) και βελτιωμένο handsfree.
- Ενημερωμένο με την τελευταία έκδοση του Webkit Browser που περιλαμβάνει δυνατότητες όπως copy-paste στον browser, ενιαίο πλαίσιο αναζήτησης, σελιδοδείκτες, ιστορικό και προβολή αγαπημένων σελίδων.
- Νέος πυρήνας Linux 2.6.27 με κάρτα SD αυτόματου ελέγχου.
- Δυνατότητα upload για βίντεο απευθείας στο YouTube και εικόνων στο Picasa αντίστοιχα.
- Δυνατότητα πρόβλεψης λέξεων κατά την πληκτρολόγηση (Auto complete) και δημιουργία λεξικού με λέξεις που ορίζει ο χρήστης.

Εικόνα 8: Λογότυπο Cupcake

1.3.4.4 Donut 1.6

Τον Σεπτέμβριο του 2009 έκανε την εμφάνισή της μία νέα έκδοση ονόματι Donut βασισμένη στο Linux Kernel 2.6.29. Αυτή η πλατφόρμα περιλαμβάνει νέα χαρακτηριστικά για τους χρήστες και τους προγραμματιστές, καθώς και ορισμένες αλλαγές στο πλαίσιο της διασύνδεσης προγραμματισμού εφαρμογών (Android API).

- Ενίσχυση του Android Market με στόχο την ευκολότερη αναζήτηση και την προβολή εικόνων των εφαρμογών.
- Πιο ολοκληρωμένο πακέτο πολυμέσων, με ταχύτερη πρόσβαση στη βιντεοκάμερα.
- Υποστήριξη για οθόνες ανάλυσης WVGA (768x480).
- Δυνατότητα μαζικής διαγραφής των φωτογραφιών.
- Ενημερωμένη τεχνολογία που υποστηρίζει CDMA/EVDO, 802.1x, VPNs και του μηχανισμού μετατροπής κειμένου σε ομιλία.

Εικόνα 9: Λογότυπο Donut

1.3.4.5 Éclair 2.0 – 2.1

Αυτή η έκδοση δημοσιεύτηκε τον Οκτώβριο του 2009 προσθέτοντας νέα χαρακτηριστικά και διορθώνοντας διάφορα σφάλματα που προέκυψαν στις προηγούμενες,

εκδόσεις της πλατφόρμας. Έτσι, η αναβάθμιση του Android Éclair 2.0-2.1 είναι πλέον έτοιμη να ανταγωνιστεί το iPhone της Apple. Οι κυριότερες αλλαγές είναι οι εξής:

- Υποστήριξη περισσοτέρων επιλογών στο κομμάτι της κάμερας όπως ψηφιακό zoom, flash και προσθήκη εφέ χρωμάτων.
- Νέες λίστες επαφών με άμεση πρόσβαση του χρήστη στις πληροφορίες κάθε επαφής, αλλά και δυνατότητα στην επιλογή άμεσου τρόπου επικοινωνίας όπως SMS ή Email.
- Νέες λειτουργίες ημερολογίου.
- Βελτιωμένο και ταχύτερο εικονικό πληκτρολόγιο με έξυπνο λεξικό που «μαθαίνει» από τη χρήση των λέξεων.
- Υποστήριξη του Bluetooth 2.1.
- Υποστήριξη σε περισσότερα μεγέθη οθόνης με καλύτερη αναλογία αντίθεσης.
- Αναβαθμισμένοι χάρτες της Google(google Maps) έκδοσης 3.1.2.
- Οι δύο επόμενες εκδόσεις ήταν οι Éclair 2.0.1(API level 6) και Eclair 2.1(API level 7) με ελάχιστες διαφορές στο API που διέθεταν.

Εικόνα 10: Λογότυπο Éclair

1.3.4.6 Froyo 2.2

Με την συνέχεια της αλφαβητικής σειράς των εκδόσεων στην έκτη θέση συναντάμε την έκδοση ονόματι Froyo 2.2 (συντομογραφία της λέξης Frozen yogurt) η οποία δημοσιεύτηκε τον Μάιο του 2010 και περιέχει τα εξής χαρακτηριστικά:

- Βελτίωση στην ταχύτητα των εφαρμογών και ενσωμάτωση του κινητήρα V8 JavaScript του Chrome στην εφαρμογή του προγράμματος περιήγησης.
- Υποστήριξη της υπηρεσίας Android Cloud to Device Messaging (C2DM) που επιτρέπει τα Push Notifications.
- Υποστήριξη της υπηρεσίας Wi-Fi hotspot όπου ο χρήστης μπορεί να μετατρέψει το smartphone του σε κινούμενο rooter.
- Επιλογή για την απενεργοποίηση της πρόσβασης των δεδομένων στο δίκτυο του κινητού τηλεφώνου.
- Υποστήριξη του Adobe Flash.
- Υποστήριξη για την εγκατάσταση εφαρμογών στην επεκτάσιμη μνήμη.
- Υποστηρίζει οθόνες υψηλής ανάλυσης τεσσάρων ιντσών 720p.

- Παρέχεται η προστασία της συσκευής μέσω αλφαβητικού ή αριθμητικού κωδικού πρόσβασης.

Εικόνα 11: Λογότυπο Froyo

1.3.4.7 Gingerbread 2.3

Η πλατφόρμα του Android 2.3 που πρωτοεμφανίστηκε τον Δεκέμβριο του 2010 εισάγει πολλές νέες και συναρπαστικές δυνατότητες για τους χρήστες και τους προγραμματιστές. Τον Φεβρουάριο του 2011 έγινε η αναβάθμιση του Gingerbread με την έκδοση 2.3.3 οπότε τα νέα χαρακτηριστικά που προστέθηκαν είναι τα εξής:

- Αναβάθμιση του σχεδιασμού διεπαφής χρήστη με ευκολία και μεγάλη ταχύτητα.
- Υποστήριξη σε μεγάλα μεγέθη και αναλύσεις οθόνης όπως WXGA(1366x768).
- Ταχύτερη και πιο διαισθητική εισαγωγή κειμένου στο εικονικό πληκτρολόγιο, με βελτιωμένη ακρίβεια, καλύτερη προτεινόμενο κείμενο και την εισαγωγή της λειτουργίας φωνητικής εισόδου.
- Προσθήκη νέων ηχητικών εφέ όπως Reverb, Equalization και ενίσχυση στην ποιότητα του μπάσου.
- Υποστηρίζει τη χρήση πολλαπλών καμερών στη συσκευή, όπως την κάμερα που υπάρχει στο μπροστινό μέρων των smartphone.

Βελτιωμένη διαχείριση ενέργειας με έναν πιο ενεργό ρόλο στη διαχείριση των εφαρμογών που κρατούν τη συσκευή ενεργή για πάρα πολύ καιρό.

Εικόνα 12: Λογότυπο Gingerbread

1.3.4.8 Honeycomb 3.0

Το Android Honeycomb που κυκλοφόρησε τον Φεβρουάριο του 2011, είναι μια έκδοση ειδικά βελτιστοποιημένη για συσκευές με μεγάλο μέγεθος οθόνης όπως τα tablets. Προσφέρει ένα ολοκαίνουριο εικονικό και ολογραφικό design που εστιάζει στο περιεχόμενο του μοντέλου αλληλεπίδρασης. Τα κυριότερα χαρακτηριστικά που αναπτύχθηκαν στη συγκεκριμένη πλατφόρμα είναι τα εξής:

- Προστέθηκε το System Bar το οποίο διαθέτει γρήγορη πρόσβαση στις κοινοποιήσεις, το status και προσθήκη κουμπιών πλοήγησης στο κάτω μέρος της οθόνης.
- Προστέθηκε το Action Bar στο πάνω μέρος της οθόνης δίνοντας πρόσβαση σε επιλογές πλοήγησης, widgets και περαιτέρω ρυθμίσεις.
- Δυνατότητα προβολής των τρέχων εφαρμογών της συσκευής στη μπάρα του συστήματος(Multitasking), με εύκολη και γρήγορη μετάβαση από τη μία εφαρμογή στην άλλη.
- Επανασχεδιασμένο πληκτρολόγιο, καθιστώντας την πληκτρολόγηση γρήγορη, αποτελεσματική και ακριβής για μεγαλύτερα μεγέθη οθόνης.
- Γρήγορη πρόσβαση τόσο στην κάμερα και στις λειτουργίες της, όσο και στα άλμπουμ της, με τη δυνατότητα προβολής των εικόνων σε πλήρη οθόνη.
- Υποστήριξη συσκευών με περισσότερους από έναν επεξεργαστές (multi-core processors).
- Απαγορεύεται η πρόσβαση σε δευτερεύουσες μνήμες αποθήκευσης.

Εικόνα 13: Λογότυπο Honeycomb

1.3.4.9 Ice Cream Sandwich 4.0 – 4.0.4

Το SDK για το Android 4.0.1 (Ice Cream Sandwich), που βασίζεται στον πυρήνα Linux 3.0.1, δημοσιεύτηκε τον Οκτωβρίου 2011 και ήταν θεωρητικά συμβατό με οποιαδήποτε συσκευή Android τη συγκεκριμένη χρονική περίοδο. Ο πηγαίος κώδικας της πλατφόρμας έγινε διαθέσιμος το Νοέμβριο του 2011 και ήταν η τελευταία έκδοση που υποστήριζε επίσημα το Adobe Flash Player System και εισήγαγε πολλά νέα και έξυπνα χαρακτηριστικά όπως:

- Διαχωρισμός των widgets σε νέα καρτέλα και ευκολότερη δημιουργία φακέλων με τη μέθοδο του drag and drop.

- Επιλογή τραβήγματος screenshots, κρατώντας πατημένο το κάτω κουμπί της τροφοδοσίας του ήχου.
- Δυνατότητα γρήγορης πρόσβασης στις εφαρμογές απευθείας από την οθόνη κλειδώματος και υποστήριξη της υπηρεσίας Face-Unlock, όπου ο χρήστης μπορεί να ξεκλειδώσει τη συσκευή του χρησιμοποιώντας το λογισμικό αναγνώρισης προσώπου.
- Υπηρεσία που προειδοποιεί τους χρήστες όταν πλησιάζουν το επιτρεπόμενο όριο χρήσης των δεδομένων τους και απενεργοποιεί τα δεδομένα, όταν γίνεται υπέρβαση του ορίου.
- Προσθήκη προγράμματος επεξεργασίας φωτογραφιών και εγγραφής βίντεο στα 1080p.
- Υποστήριξη της υπηρεσίας Wi-Fi Direct, όπου ο χρήστης μπορεί να συνδεθεί με άλλες συσκευές χωρίς την παρουσία ασύρματου δικτύου πρόσβασης.

Μέσα στους επόμενους μήνες ακολούθησαν οι εκδόσεις Ice Cream Sandwich 4.0.1, 4.0.2 και 4.0.3, 4.0.4 με API Level 14 και 15 αντίστοιχα, όπου παρείχαν ορισμένες διορθώσεις και βελτιώσεις στα γραφικά, τη βάση δεδομένων και τη λειτουργία Bluetooth. Ακόμα, αναπτύχθηκαν σημαντικά οι λειτουργίες της κάμερας με ενίσχυση σταθερότητας της εικόνας, ανάλυση QVGA και ομαλότερη περιστροφή οθόνης.

Εικόνα 14: Λογότυπο Ice Cream Sandwich

1.3.4.10 Android Jelly Bean 4.1 – 4.3.1

Η τελευταία μεγάλη αναβάθμιση του Android με όνομα Jelly Bean έκανε το ντεμπούτο της τον Ιούνιο του 2012. Με ακόμη πιο όμορφο Android UI, με εξεζητημένο λογισμικό που ανανεώνει την ταχύτητα του συστήματος αλλά και με τις πιο ανανεωμένες εκδώσεις της σειράς, το Android πλέον επιβεβαιώνεται ως το λειτουργικό σύστημα που προτιμούν οι χρήστες. Η πρώτη συσκευή που «έτρεξε» τη συγκεκριμένη έκδοση είναι το tablet Nexus 7 με χαρακτηριστικά όπως:

- Δυνατότητα απενεργοποίησης των ειδοποιήσεων στις εφαρμογές.
- Τα Widgets και οι συντομεύσεις εφαρμογών μπορούν αυτόματα να προσαρμόσουν το μέγεθος εμφάνισής τους στην αρχική οθόνη της συσκευής (home screen).
- Πολλαπλούς λογαριασμούς χρηστών σε tablet, συναντάμε στην επόμενη έκδοση της Jelly Bean 4.2, βασισμένη στον πυρήνα Linux 3.4.0, καθώς και βελτιώσεις προσβασιμότητας όπως zoom με δύο δάχτυλα και triple-tap για μεγέθυνση ολόκληρης της οθόνης.
- Ο κωδικοποιητής Fraunhofer FDK ACC γίνεται πρότυπο για το Android, προσθέτοντας στη συνέχεια το κανάλι κωδικοποίησης-αποκωδικοποίησης ACC 5.1.

- Δυνατότητα μεταφοράς δεδομένων μέσω Bluetooth για το Android Beam.
- Υποστήριξη ήχου USB Audio(για εξωτερικό ήχο DACs) αλλά και του Audio Chaining(γνωστό και ως αναπαραγωγή Gapless).

Εικόνα 15: Λογότυπο Android Jelly Bean

1.3.4.11 Android KitKat 4.4

Η Google ανακοίνωσε την έκδοση του Android 4.4 το Σεπτέμβριο του 2013 τελικά με το όνομα «KitKat» , παρότι η αρχική ονομασία της συγκεκριμένης έκδοσης άκουγε στο όνομα «Key Lime Pie». Έτσι, η έκδοση αυτή έκανε το ντεμπούτο της με τη συσκευή Nexus 5 και είναι βελτιστοποιημένη στο να λειτουργεί με μεγαλύτερη γκάμα συσκευών απ' ότι οι παλιότερες εκδόσεις, έχοντας 512MB μνήμη RAM και επιπρόσθετες δυνατότητες όπως:

- Το ψηφιακό κουμπί του μενού είναι πάντα ορατό ακόμα και στις συσκευές που η μετάβαση τους στο μενού γίνεται με το πάτημα ενός πλήκτρου (αν και η συγκεκριμένη μέθοδος καταργήθηκε επίσημα από το Android 4.0).
- Βελτιστοποίηση για την απόδοση σε συσκευές με χαμηλότερες προδιαγραφές, συμπεριλαμβανομένης της υποστήριξης zRAM και low-RAM.
- Υποστήριξη του Bluetooth Message Access Profile.
- Δημόσιο API για την ανάπτυξη και τη διαχείριση των μηνυμάτων κειμένου.
- Εισαγωγή του Android Runtime ως μια νέα πειραματική εφαρμογή χρόνου εκτέλεσης, την οποία μπορεί να ενεργοποιήσει ο χρήστης και δημιουργήθηκε με σκοπό να αντικαταστήσει την εικονική μηχανή Dalvik.
- Απενεργοποίηση κειμένου στο πρόγραμμα περιήγησης Web View.

Εικόνα 16: Λογότυπο Android KitKat

1.3.4.12 Android Lollipop 5.0

Αυτή η έκδοση που πρωτοπαρουσιάστηκε με κωδικό όνομα «Android L» τον Ιούνιο του 2014, είναι γεμάτη με νέα χαρακτηριστικά για τους χρήστες και χιλιάδες νέα API για τους προγραμματιστές. Διαθέτει ένα επανασχεδιασμένο περιβάλλον εργασίας, χτισμένο γύρω από μια ευέλικτη σχεδιαστική γλώσσα που ονομάζεται «Material Design». Έτσι, η πλατφόρμα Android επεκτείνεται ακόμα περισσότερο από smartphones, tablets μέχρι και σε τηλεοράσεις, αυτοκίνητα με χαρακτηριστικά όπως:

- Υποστήριξη επεξεργαστών 64-Bit.
- Ανανεωμένο κλείδωμα οθόνης που παρέχει συντομεύσεις για τις εφαρμογές και τις κοινοποιήσεις, χωρίς να υποστηρίζει πλέον τα widgets.
- Οι αναζητήσεις μπορούν να πραγματοποιηθούν εντός των ρυθμίσεων του συστήματος για την ταχύτερη πρόσβαση σε συγκεκριμένους χώρους.
- Επιτρέπεται η εισαγωγή πολλαπλών λογαριασμών των χρηστών για τις περισσότερες συσκευές που «τρέχουν» το Android 5.0.
- Δυνατότητα της εισόδου και εξόδου του ήχου μέσω της θύρας USB.
- Προσθήκη 15 νέων γλωσσών.
- Πρόσφατες χρησιμοποιούμενες εφαρμογές, μένουν στη μνήμη ακόμα και μετά την επανεκκίνηση της συσκευής.
- Προσθήκη του «Project Volta» το οποίο αναφέρεται στην βελτίωση και τη μεγαλύτερη διάρκεια ζωής της μπαταρίας της συσκευής.

Εικόνα 17: Λογότυπο Android Lollipop

1.4 Δομή Εργασίας

Ξεκινώντας την αναφορά, στο 1ο κεφάλαιο "Εισαγωγή" αναφερόμαστε στους στόχους και στους σκοπούς που εξυπηρετεί η πτυχιακή εργασία, καθώς και σε γενικές πληροφορίες που αφορούν την πλατφόρμα του λειτουργικού Android. Το 2ο κεφάλαιο

"Τρόπος Υλοποίησης" περιλαμβάνει εισαγωγικές πληροφορίες, τα χρήσιμα εργαλεία καθώς και το προγραμματιστικό περιβάλλον του Android Studio, βάση του οποίου θα υλοποιήσουμε την εφαρμογή μας. Στο 3ο κεφάλαιο "Ανάλυση και Σχεδιασμός" αναλύουμε τις επιμέρους λειτουργίες και τις τεχνολογίες - μεθόδους που χρησιμοποιήθηκαν. Τέλος, στο 4ο κεφάλαιο "Προγραμματισμός και Κωδικοποίηση" παρουσιάζουμε τον επιμέρους κώδικα που συνθέτει την εφαρμογή μας και στο 5ο κεφάλαιο "Βιβλιογραφία" παρέχουμε τις πηγές άντλησης πληροφοριών για τη διεξαγωγή της εργασίας.

2 Τρόπος Υλοποίησης

Όπως αναφέραμε στο προηγούμενο κεφάλαιο ο σκοπός της πτυχιακής εργασίας είναι η ανάπτυξη μιας εφαρμογής στο γραφικό περιβάλλον του λειτουργικού συστήματος Android. Για να δημιουργήσουμε μία εφαρμογή Android απαιτείται η χρήση της αντικειμενοστραφής γλώσσας προγραμματισμού Java. Το λειτουργικό σύστημα Android είναι μία πλατφόρμα ανοικτού κώδικα, που δίνει τη δυνατότητα στους προγραμματιστές να συνθέσουν και να υλοποιήσουν δικές τους εφαρμογές δωρεάν. Έτσι, η συγκεκριμένη εργασία θα υλοποιηθεί στο προγραμματιστικό περιβάλλον του Android Studio IDE (Integrated Development Environment), το οποίο είναι ένα σχετικά νέο πρόγραμμα ανάπτυξης εφαρμογών για πλατφόρμες τύπου Android που σκοπό έχει να αντικαταστήσει τα Eclipse Android Development Tools (ADT) ως το κύριο IDE της Google.

2.1 Εισαγωγή στο Android Studio

Το Android Studio που βασίζεται στο λογισμικό της JetBrains' IntelliJ IDEA, παρέχει οδηγούς και περιγράμματα που επαληθεύουν τις απαιτήσεις του συστήματος. Ορισμένοι οδηγοί και υπηρεσίες αυτής της ολοκληρωμένης πλατφόρμας είναι το Java Development Kit (JDK), η διαθέσιμη μνήμη RAM και η διαμόρφωση των προεπιλεγμένων ρυθμίσεων, όπως η βελτιστοποιημένη προεπιλογή Android Virtual Device (AVD). Επιπλέον περιλαμβάνεται μια σειρά από χαρακτηριστικά που συμβάλλουν στην πιο παραγωγική και αποτελεσματική κωδικοποίηση του χρήστη όπως:

Εικόνα 18: Λογότυπο Android Studio IDE

1. Έξυπνη ερμηνεία: Με την έξυπνη ερμηνεία το Android Studio εμφανίζει επιλογές κατά την πληκτρολόγηση για γρήγορες λύσεις σε σφάλματα που δημιουργήθηκαν. Για παράδειγμα αν προσθέσουμε στην εφαρμογή μας ένα κουμπί χωρίς να έχουμε συμπληρώσει τα πεδία των χαρακτηριστικών του (πλάτος, ύψος κ.α.), τότε AS εμφανίζει μήνυμα αυτόματης συμπλήρωσης των επιμέρους χαρακτηριστικών. Έτσι κλικάροντας το μήνυμα προσθέτονται τα ελλείποντα χαρακτηριστικά στη φόρμα.

2. Δημιουργία νέου αρχείου: Δυνατότητα γρήγορης προσθήκης νέου κωδικού και δημιουργία αρχείων, κάνοντας κλικ στο κατάλληλο κατάλογο στο παράθυρο του Project πατώντας ALT + INSERT για Windows και Linux ή Command + N για Mac. Με βάση το είδος του επιλεγμένου καταλόγου, το AS δημιουργεί τον κατάλληλο τύπο αρχείου. Για παράδειγμα αν επιλέξετε έναν κατάλογο διάταξης πατώντας ALT + INSERT ανοίγει ένα πλαίσιο κειμένου ώστε να μπορείτε να ονομάσετε το αρχείο.

3. Ιεραρχική ρύθμιση parent: Η δραστηριότητα parent μπορεί να οριστεί πλέον στον Οδηγό δραστηριότητας κατά τη δημιουργία μιας νέας δραστηριότητας.

4. Δημιουργία διαγράμματος: Το AS προσφέρει ένα προηγμένο διάγραμμα διάταξης που επιτρέπει την δραστηριότητα drag-and-drop μικροεφαρμογών (widgets) και τη δυνατότητα προεπισκόπησης του διαγράμματος κατά την επεξεργασία του XML αρχείου. Κατά την επεξεργασία του κειμένου, μπορείτε να κάνετε προεπισκόπηση του διαγράμματος για τις συσκευές με το άνοιγμα του παραθύρου προεπισκόπησης το οποίο είναι διαθέσιμο στη δεξιά πλευρά του παραθύρου. Μέσα στο παράθυρο αυτό, μπορείτε να τροποποιήσετε την προεπισκόπηση αλλάζοντας διάφορες επιλογές στο επάνω μέρος του παραθύρου, συμπεριλαμβανομένης της συσκευής, το θέμα του διαγράμματος κ.α. Τέλος υπάρχει επιλογή " Preview All Screen Sizes" για την προεπισκόπηση του διαγράμματος σε πολλαπλές συσκευές ταυτόχρονα και επιλογή "Design" στο κάτω μέρος του παραθύρου για τη μετάβαση στο γραφικό περιβάλλον με πολλές επιπρόσθετες δυνατότητες.

5. Java class decompiling: Το AS επιτρέπει να δούμε τι υπάρχει μέσα στις βιβλιοθήκες της Java όταν δεν υπάρχει πρόσβαση στον πηγαίο κώδικα. Ο Decompiler βρίσκεται μέσα στο AS για εύκολη πρόσβαση. Για να χρησιμοποιήσουμε αυτή τη λειτουργία, κάνουμε δεξί κλικ σε μια κλάση, μέθοδο ή πεδίο από μια βιβλιοθήκη για την οποία δεν έχουμε πρόσβαση στο αρχείο προέλευσης και επιλέγουμε αποσυμπίεσε. Για να προσαρμόσουμε τις ρυθμίσεις του Java Decompiler, επιλέγουμε Αρχείο> Ρυθμίσεις> Άλλες ρυθμίσεις> Java Decompiler.

6. Εντοπισμός σφαλμάτων και βελτίωση απόδοσης: Προσφέρεται εντοπισμός σφαλμάτων και βελτιώσεις απόδοσης όπως προσαρμοσμένα keymaps. Για να τροποποιήσουμε το τρέχων keymap επιλέγουμε Αρχείο> Ρυθμίσεις> Keymap.

2.1.1 Android SDK

Android SDK σημαίνει "Android Software Development Kit " και είναι το επίσημο εργαλείο της Google που επιτρέπει στους προγραμματιστές να δημιουργήσουν εφαρμογές για την πλατφόρμα Android. Το Android SDK περιλαμβάνει δείγματα με τον πηγαίο κώδικα, πρόγραμμα εντοπισμού σφαλμάτων, εργαλεία ανάπτυξης, ένα εξομοιωτή, tutorials και τις απαιτούμενες βιβλιοθήκες για τη δημιουργία εφαρμογών Android. Οι εφαρμογές είναι γραμμένες στη γλώσσα προγραμματισμού Java και τρέχουν στην εικονική μηχανή Dalvik, που έχει σχεδιαστεί για να τρέχει πάνω από ένα Linuxkernel.

Οι βελτιώσεις του SDK πάνε παράλληλα με τη συνολική ανάπτυξη της πλατφόρμας. Το SDK υποστηρίζει επίσης παλαιότερες εκδόσεις της πλατφόρμας Android, σε περίπτωση που οι προγραμματιστές θέλουν να χρησιμοποιήσουν τις εφαρμογές τους σε παλαιότερες συσκευές. Τα Development tools είναι πάντα διαθέσιμα για download όπως επίσης και οι

παλαιότερες εκδόσεις τους, ώστε ο χρήστης να μπορεί να ελέγξει μετά από καιρό τη συμβατότητά τους.

Το Android SDK περιλαμβάνει μια ποικιλία εργαλείων που βοηθούν στην ανάπτυξη κινητών εφαρμογών για την πλατφόρμα Android. Τα εργαλεία αυτά κατατάσσονται σε δύο κατηγορίες: εργαλεία SDK και εργαλεία της πλατφόρμας. Τα εργαλεία του SDK απαιτούνται ανεξάρτητα από ποιά πλατφόρμα Android χρησιμοποιούμε, ενώ τα εργαλεία της πλατφόρμας προσαρμόζονται με σκοπό να υποστηρίζουν τα εκάστοτε χαρακτηριστικά της. Έτσι δε μένει παρά να κατεβάσουμε τα SDK tools (τα οποία ενημερώνονται περιοδικά) στον υπολογιστή μας, πράγμα που γίνεται αυτόματα με την εγκατάσταση του Android Studio.

Μέσω του SDK μπορούμε να χρησιμοποιήσουμε εργαλεία όπως το ADB, για να μεταφέρουμε αρχεία σε χώρους που κανονικά δεν επιτρέπεται και το fastboot, για να εγκαθιστούμε custom recovery εικόνες και να ξεκλειδώνουμε τον bootloader της συσκευής μας. Το fastboot είναι ένα πρόγραμμα με το οποίο μπορούμε να ξεκλειδώσουμε τον bootloader σε πολλές συσκευές αλλά και να εγκαταστήσουμε αρχεία εικόνων (.img) στη συσκευή μας μέσω υπολογιστή. Μπορούμε να εγκαταστήσουμε οποιοδήποτε .img αρχείο καθώς δε διαθέτει κάποια δικλείδα ασφαλείας οπότε πρέπει να δίνουμε μεγάλη προσοχή στο αν το αρχείο που έχουμε στα χέρια μας, είναι κατασκευασμένο για τη συσκευή μας ή όχι. Τέτοια αρχεία .img είναι οι recovery που εγκαθιστούμε στη συσκευή μας (π.χ. ClockworkMod) και χρειαζόμαστε το fastboot για να αντικαταστήσουμε την επίσημη με μία άλλη που θα διαθέτει περισσότερες λειτουργίες.

Εικόνα 19: Testflight Android – SDK

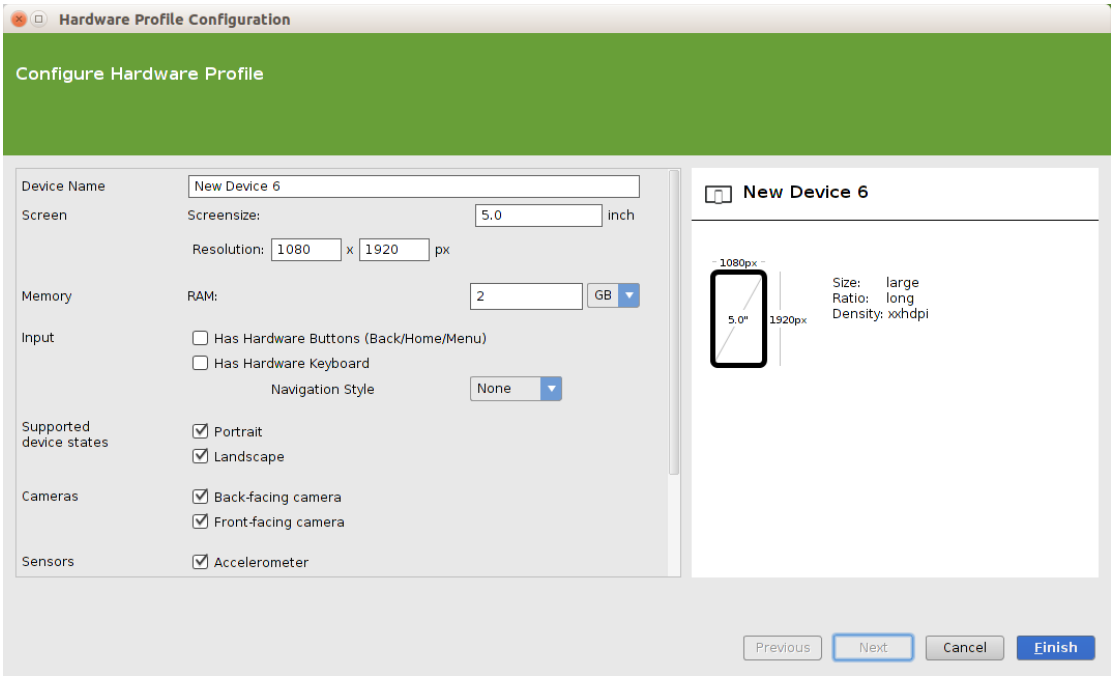
2.1.2 Android Virtual Device

Το Android Virtual Device (AVD) είναι ένα εικονικό smartphone που τρέχει Android, στο οποίο μπορούμε εύκολα να εκτελέσουμε ότι πρόγραμμα δημιουργήσαμε. Ο ευκολότερος τρόπος να δημιουργήσουμε ένα AVD είναι να χρησιμοποιήσουμε το γραφικό

AVD Manager, κλικάροντας Tools > Android > AVD Manager. Μπορούμε επίσης να το ξεκινήσουμε πατώντας το εικονίδιο AVD Manager από το toolbar.

- Ένα hardware profile: Καθορίζει τα χαρακτηριστικά του υλικού της εικονικής συσκευής. Για παράδειγμα, μπορούμε να ορίσουμε αν η συσκευή διαθέτει κάμερα, αν χρησιμοποιεί ψηφιακό ή φυσικό πληκτρολόγιο (QWERTY), να καθορίσουμε τη χωρητικότητα της μνήμη κ.α. (βλ.εικόνα 19)
- Άλλες επιλογές : Μπορούμε να επιλέξουμε την έκδοση του Android που θα "τρέχει" στην εικονική μας μηχανή, αλλά και ότι έχει να κάνει με τις διαστάσεις και την εμφάνιση της οθόνης.
- Ένα ειδικό χώρο αποθήκευσης στον υπολογιστή που αποθηκεύονται τα δεδομένα της συσκευής του χρήστη (όπως εγκατεστημένες εφαρμογές, ρυθμίσεις και ούτω καθεξής).

Μπορούμε να δημιουργήσουμε όσα AVD χρειάζεστε, με βάση τους τύπους της συσκευής που θέλουμε να μοντελοποιήσουμε. Για να ελέγξουμε προσεκτικά την εφαρμογή, θα πρέπει να φτιάξουμε ένα AVD για κάθε γενική διαμόρφωση της συσκευής (για παράδειγμα, σε διαφορετικά μεγέθη οθόνης και εκδόσεις της πλατφόρμας) όπου η εφαρμογή θα είναι συμβατή και θα "τεστάρετε" σε κάθε μία από αυτές.



Εικόνα 20: AVD's Hardware Profile

Σε περίπτωση δημιουργίας ενός νέου AVD μπορούμε να καθορίσουμε τις ακόλουθες επιλογές υλικού:

Χαρακτηριστικά	Περιγραφή	Ιδιότητα
Μέγεθος μνήμης RAM της συσκευής	Το ποσό της φυσικής μνήμης RAM της συσκευής σε megabyte.Η προεπιλεγμένη	hw.ramSize

	τιμή είναι "96".	
Υποστήριξη οθόνης αφής	Είτε υπάρχει οθόνη αφής είτε όχι στη συσκευή, η προεπιλεγμένη τιμή είναι "ναι".	hw.touchScreen
Υποστήριξη Trackball	Αν υπάρχει ένα trackball στη συσκευή, η προεπιλεγμένη τιμή είναι "ναι".	hw.trackBall
Υποστήριξη πληκτρολογίου	Αν η συσκευή έχει πληκτρολόγιο QWERTY, η προεπιλεγμένη τιμή είναι "ναι".	hw.keyboard
Υποστήριξη DPad	Εάν η συσκευή διαθέτει πλήκτρα Dpad τότε η προεπιλεγμένη τιμή είναι "ναι".	hw.dPad
Υποστήριξη GSM Modem	Εάν υπάρχει ένα GSM μόντεμ στη συσκευή τότε η προεπιλεγμένη τιμή είναι "ναι".	hw.gsmModem
Υποστήριξη κάμερας	Αν η συσκευή διαθέτει κάμερα, η προεπιλεγμένη τιμή είναι "όχι".	hw.camera
Maximum horizontal camera pixels	Η προεπιλεγμένη τιμή είναι "640".	hw.camera.maxHorizontalPixels
Maximum vertical camera pixels	Η προεπιλεγμένη τιμή είναι "480".	hw.camera.maxVerticalPixels
Υποστήριξη GPS	Αν η συσκευή διαθέτει GPS, η προεπιλεγμένη τιμή είναι "ναι".	hw.gps
Υποστήριξη μπαταρίας	Αν η συσκευή μπορεί να λειτουργεί με μπαταρία τότε η προεπιλεγμένη τιμή είναι "ναι".	hw.battery

Επιταχυνσιόμετρο	Αν υπάρχει επιταχυνσιόμετρο στη συσκευή τότε η προεπιλεγμένη τιμή είναι "ναι".	hw.accelerometer
Υποστήριξη εγγραφής ήχου	Αν η συσκευή μπορεί να καταγράφει ήχο τότε η προεπιλεγμένη τιμή είναι "ναι".	hw.audioInput
Υποστήριξη αναπαραγωγής ήχου	Αν η συσκευή μπορεί να αναπαράγει ήχο τότε η προεπιλεγμένη τιμή είναι "ναι".	hw.audioOutput
Υποστήριξη κάρτας SD	Αν η συσκευή υποστηρίζει την εισαγωγή / αφαίρεση της εικονικής κάρτας SD τότε η προεπιλεγμένη τιμή είναι "ναι".	hw.sdCard
Υποστήριξη Cache partition	Αν η συσκευή χρησιμοποιεί κάποιο cache partition, η προεπιλεγμένη τιμή είναι "ναι".	disk.cachePartition
Μέγεθος Cache partition	Η προεπιλεγμένη τιμή είναι "66MB".	disk.cachePartition.size
Abstracted LCD density	Ρυθμίζει τα γενικότερα χαρακτηριστικά που χρησιμοποιούνται από την οθόνη του AVD. Η προεπιλεγμένη τιμή είναι "160".	hw.lcd.density

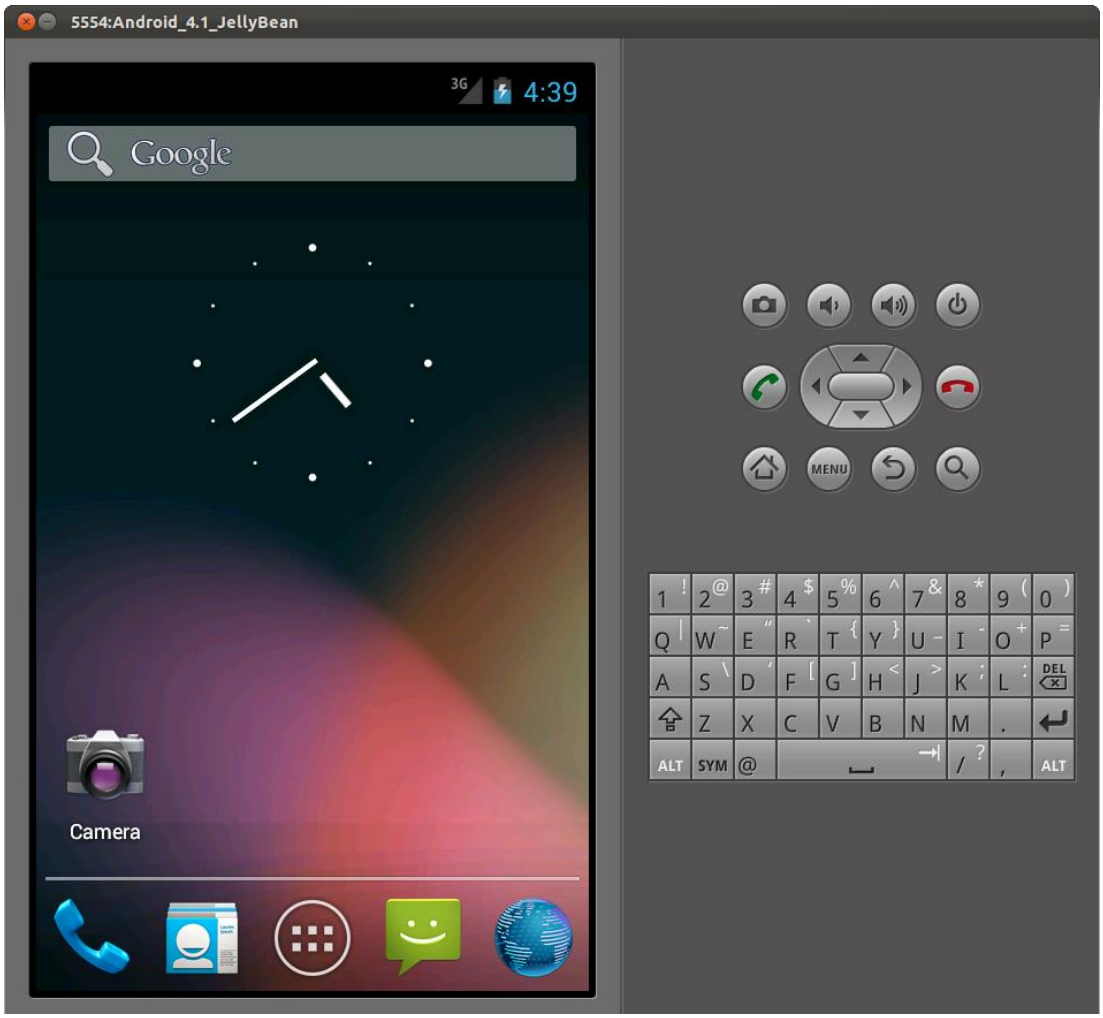
2.1.3 Android Emulator

Το Android SDK περιλαμβάνει ένα εξομοιωτή εικονικής κινητής συσκευής που τρέχει στον υπολογιστή μας. Ο εξομοιωτής συμβάλλει στην πρωτοτυπία του AS με τη δυνατότητα ανάπτυξης και δοκιμής εφαρμογών Android, χωρίς τη χρήση κάποιας φυσικής συσκευής.

Το Android Emulator μιμείται όλα τα χαρακτηριστικά λογισμικού και υλικού μιας κινητής συσκευής, εκτός από τη δυνατότητα πραγματοποίησης μιας τηλεφωνικής κλήσης.

Παρέχει επιπλέον πλήκτρα ελέγχου, τα οποία μπορούμε να πατήσουμε χρησιμοποιώντας το πληκτρολόγιο ή το ποντίκι ώστε να δημιουργήσουμε τα "γεγονότα" στην εφαρμογή μας. Τέλος, υπάρχει μία οθόνη στην οποία προβάλλεται η εφαρμογή μας μαζί με άλλες πιθανές ενεργές εφαρμογές Android.

Προκειμένου να δοκιμάσουμε και να τεστάρουμε την εφαρμογή μας πιο εύκολα, ο εξομοιωτής χρησιμοποιεί το AVD. Το AVD όπως προαναφέραμε μας επιτρέπει να ορίσουμε κάποια χαρακτηριστικά στη συσκευή του εξομοιωτή όπως και να δοκιμάσουμε την εφαρμογή σε πολλαπλές πλατφόρμες Android. Έτσι όταν η εφαρμογή μας τρέξει στον εξομοιωτή, μπορεί να χρησιμοποιήσει τις υπηρεσίες της πλατφόρμας Android και να επικαλεστεί άλλες εφαρμογές όπως η πρόσβαση στο δίκτυο, η αναπαραγωγή ήχου και βίντεο, η αποθήκευση και ανάκτηση δεδομένων, οι ειδοποιήσεις κ.α. Ο εξομοιωτής περιλαμβάνει επίσης μια ποικιλία από δυνατότητες εντοπισμού σφαλμάτων.

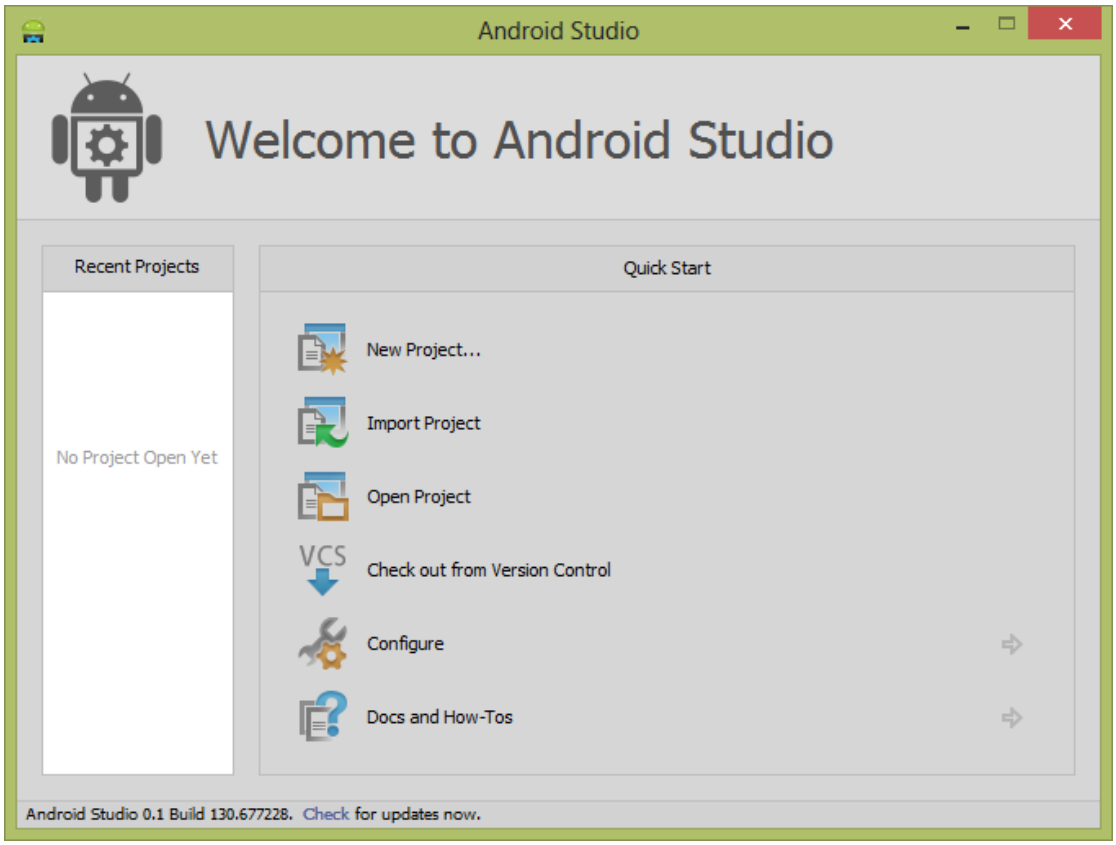


Εικόνα 21: Εξομοιωτής Κινητής Συσκευής Android

2.2 Ξεκινώντας το Android Studio

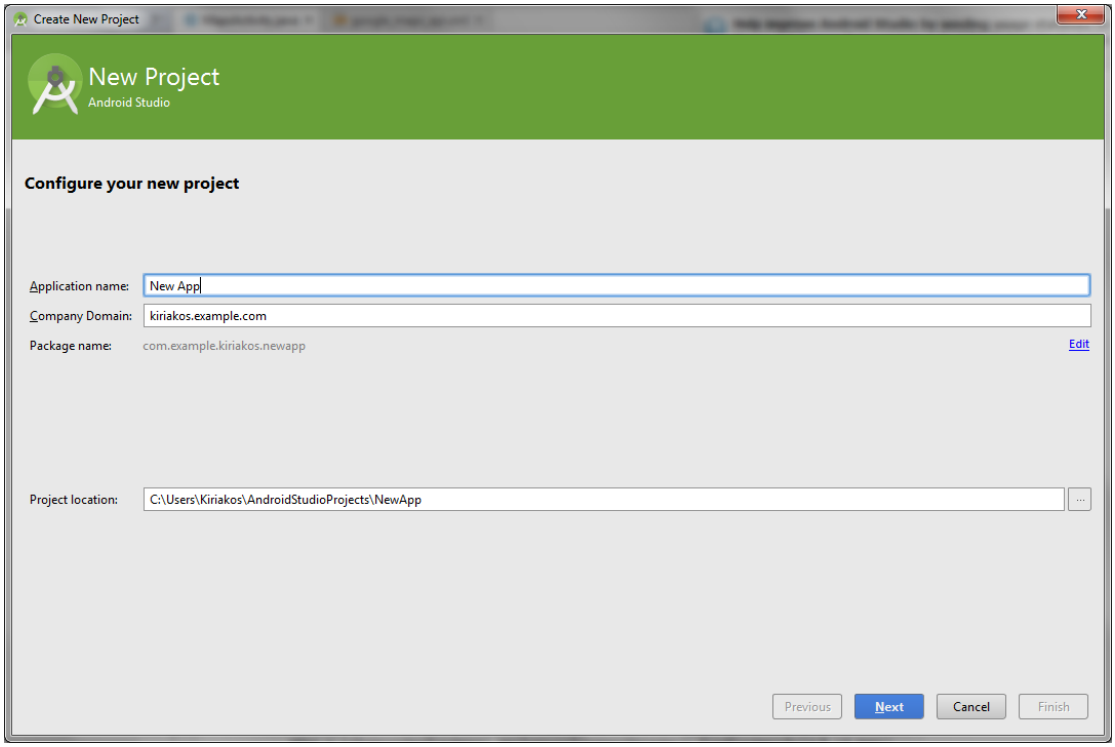
Σε αυτό το κομμάτι θα τονίσουμε τα πρώτα βήματα που πρέπει να ακολουθήσουμε για τη δημιουργία του νέου μας Project. Ξεκινώντας, το AS μας καλωσορίζει με την εμφάνιση ενός παραθύρου στο οποίο έχουμε τη δυνατότητα να κλικάρουμε μία από τις

διαθέσιμες επιλογές ανάλογα με το τι θέλουμε να κάνουμε (βλ.εικόνα 22). Στην περίπτωσή μας, πατάμε την επιλογή New Project. Στην συνέχεια, εμφανίζει το παράθυρο όπου ορίζουμε το όνομα του Project, το όνομα πακέτου αλλά και την τοποθεσία στην οποία θέλουμε να



αποθηκευτεί στον υπολογιστή μας (βλ.εικόνα 23).

Εικόνα 22: Αρχικό παράθυρο Android Studio

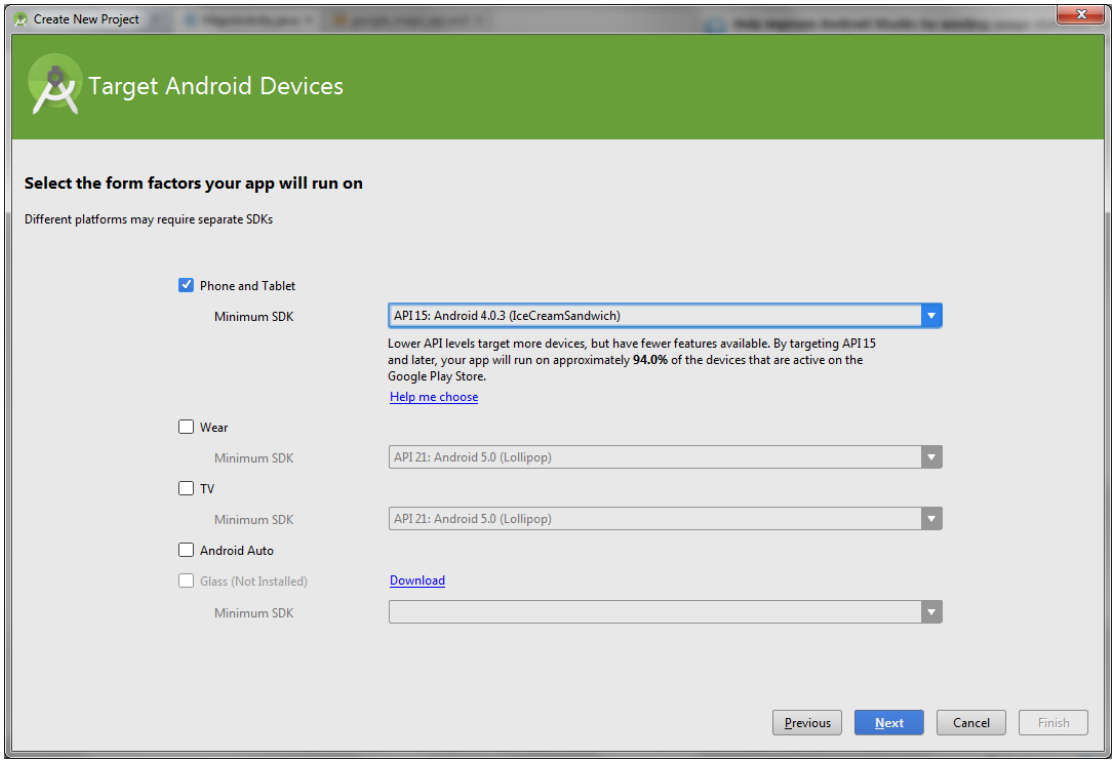


Εικόνα 23: Παράθυρο ονομασίας Project

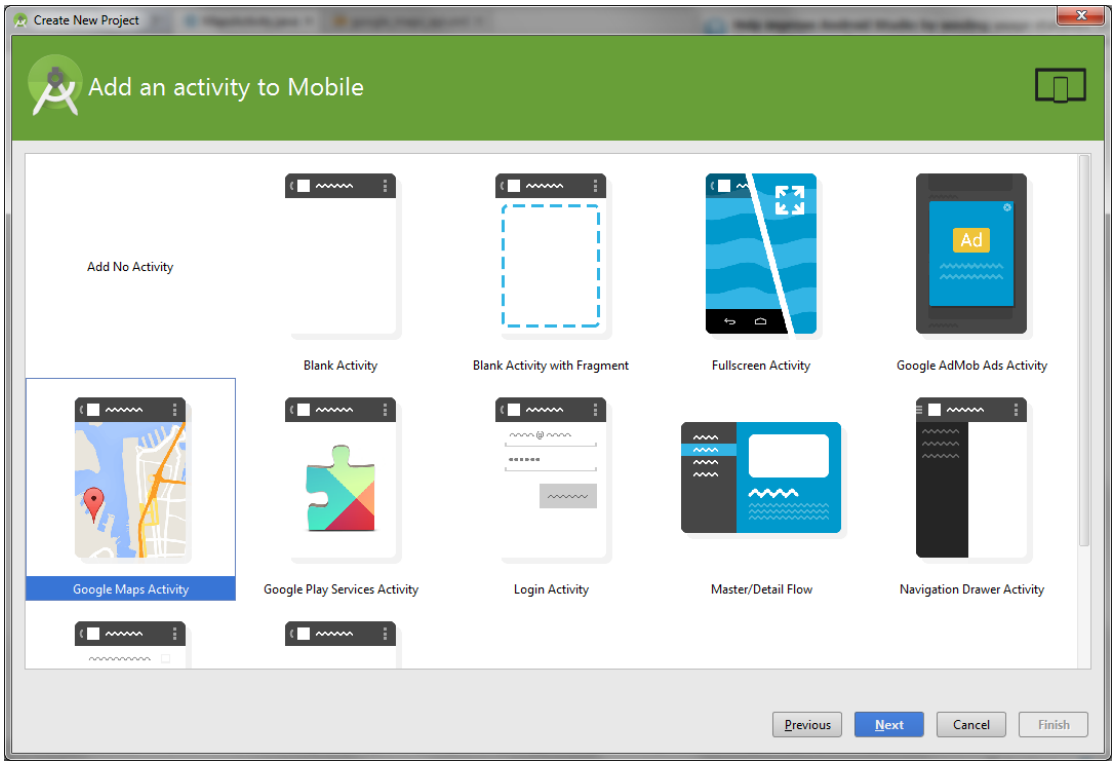
Ορίζοντας λοιπόν το όνομα της εφαρμογής μας, περνάμε στο παράθυρο όπου επιλέγουμε σε τι συσκευή επιθυμούμε να "τρέξει" η εφαρμογή μας και κλικάρουμε Phone and Tablet. Από κάτω ακριβώς επιλέγουμε την χαμηλότερη έκδοση όπου θα τρέξει η εφαρμογή (Παρ' ότι τα χαμηλότερα επίπεδα API διαθέτουν λιγότερες επιλογές, μπορούν να εξυπηρετήσουν μεγαλύτερο εύρος συσκευών) API 15: Android 4.0.3 Ice Cream Sandwich.

Έπειτα μας δίνεται η δυνατότητα να επιλέξουμε τον τύπο του πρωταρχικού Activity από το οποίο θα εκκινείτε η εφαρμογή μας και με αφορμή τον τύπο της εφαρμογής που θα αναπτύξουμε, θα επιλέξουμε το Google Maps Activity (βλ. Εικόνα 25). Πατώντας το κουμπί next ορίζουμε την ονομασία του Activity, το όνομα του layout που αντιπροσωπεύει το Activity αυτό αλλά και τον τίτλο που θα αναγράφεται στο Action Bar της εφαρμογής μας (βλ. Εικόνα 26).

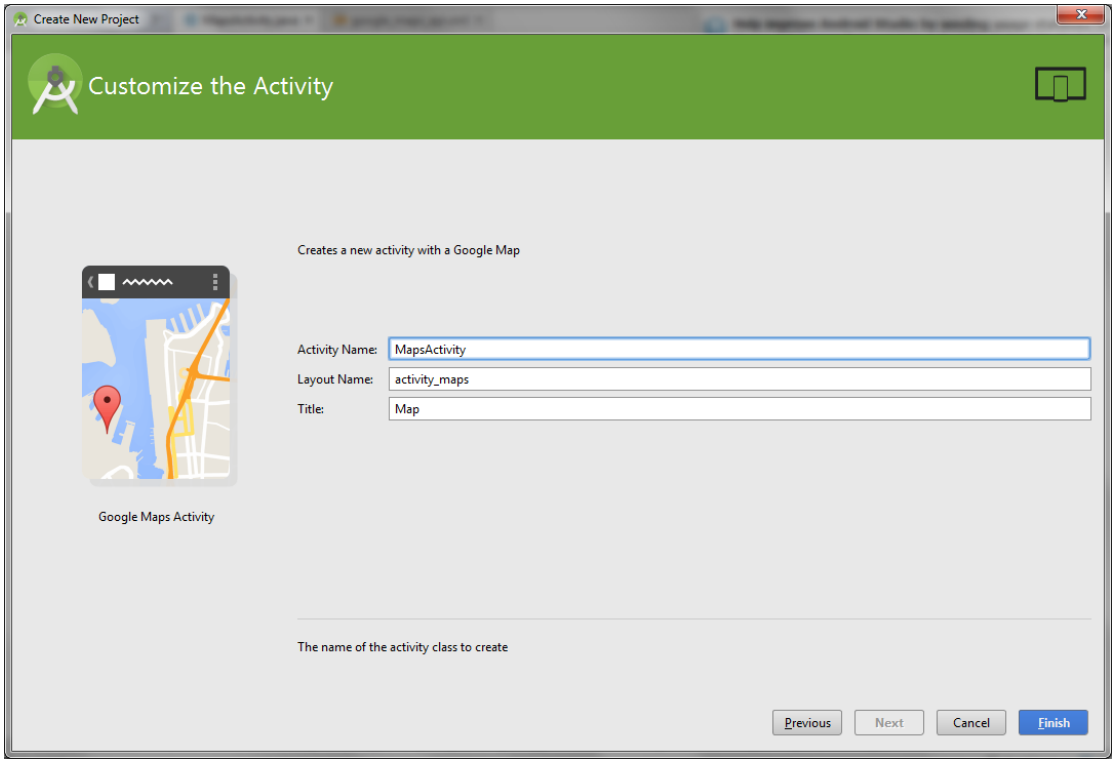
Τέλος, πατώντας το κουμπί Finish μεταφερόμαστε στο βασικό περιβάλλον εργασίας του AS όπου αριστερά βρίσκονται οι δομημένοι φάκελοι του και στα δεξιά το πάνελ που θα προγραμματίσουμε την εφαρμογή μας (βλ. Εικόνα 27).



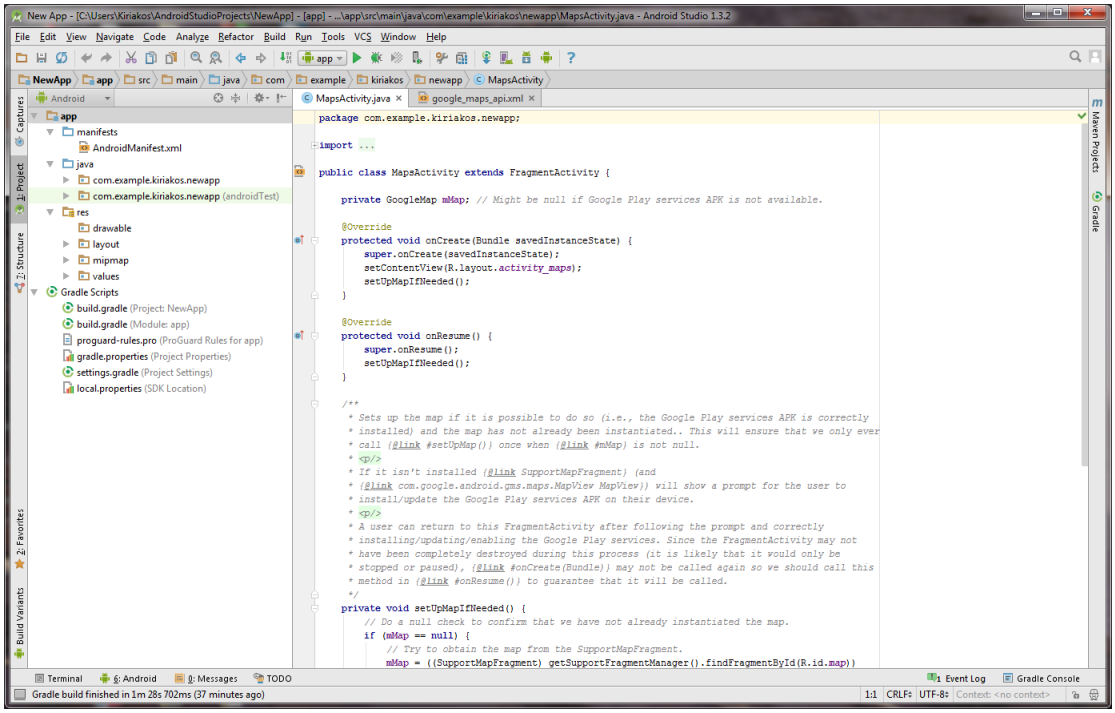
Εικόνα 24: Παράθυρο Επιλογής SDK API Level



Εικόνα 25: Επιλογή του Activity



Εικόνα 26: Ονομασία του Activity



Εικόνα 27: Προγραμματιστικό Περιβάλλον του Android Studio

3 Ανάλυση και σχεδιασμός

3.1 Γενική Εικόνα

Η εργασία αυτή έχει ως στόχο την υλοποίηση μιας εφαρμογής, στο λειτουργικό σύστημα Android, η όποια δίνει την δυνατότητα σε άτομα με κινητικά προβλήματα να κινηθούν γρήγορα και με ευκολία στην πόλη των Χανιών. Η κίνησή τους μπορεί να γίνεται είτε με την χρήση αυτοκίνητου είτε με αναπηρικό αμαξίδιο.

Στην πρώτη περίπτωση, ο χρήστης έχει τη δυνατότητα να βρίσκει και να επιλέγει χώρο στάθμευσης του αναπηρικού του οχήματος (μέσα από ειδικές θέσεις parking αναπήρων που έχουν εντοπιστεί στην πόλη των Χανίων), προκειμένου να διευκολύνει την μετακίνησή του στην πόλη. Αντίθετα, αν η κίνηση γίνεται με τη χρήση μόνο του αναπηρικού αμαξιδίου, η εφαρμογή παρέχει στο χρήστη τις απαραίτητες οδηγίες για να μεταβεί εύκολα και άνετα στον προορισμό του, με τη βοήθεια ειδικών ράμπων για αναπήρους.

3.2 Λειτουργίες Εφαρμογής

Αρχικά, η εφαρμογή παρουσιάζει μια εικόνα και δυο κουμπιά με τις ενδείξεις: «ΚΙΝΗΣΗ ΜΕ ΑΥΤΟΚΙΝΗΤΟ/PARKING» και «ΚΙΝΗΣΗ ΜΕ ΑΝΑΠΗΡΙΚΟ ΑΜΑΞΙΔΙΟ». Με την επιλογή των συγκεκριμένων κουμπιών, ο χρήστης ξεκινάει την πλοήγησή του στην εφαρμογή. Εκεί παρουσιάζεται το βασικό στάδιο της εφαρμογής.

Συγκεκριμένα, στο βασικό στάδιο της εφαρμογής θα εμφανίζεται ο χάρτης της πόλης από τις δύο καρτέλες επιλογής που θα καλούν τον επισκέπτη να επιλέξει αν η πλοήγησή του στην πόλη θα γίνει με αυτοκίνητο ή με αναπηρικό αμαξίδιο. Στην περίπτωση που ο χρήστης κινηθεί με το αυτοκίνητό του, θα πρέπει να επιλέξει το κουμπί καρτέλας «ΚΙΝΗΣΗ ΜΕ ΑΥΤΟΚΙΝΗΤΟ/PARKING». Σε επόμενο στάδιο, το άτομο θα μεταβεί σε μία καρτέλα στην οποία θα εμφανίζεται ο χάρτης της πόλης και δύο μπάρες εκείνη της αφετηρίας και εκείνη του προορισμού. Πάνω στον χάρτη θα είναι καταγεγραμμένοι όλοι οι χώροι στάθμευσης που προορίζονται για άτομα με κινητικά προβλήματα. Αφού ο χρήστης εισάγει στην μπάρα αφετηρίας το σημείο στο οποίο βρίσκεται και στη μπάρα προορισμού το σημείο στο οποίο θέλει να φτάσει, έχει τη δυνατότητα να επιλέξει την πλησιέστερη, στο σημείο προορισμού του, θέση parking. Μετά από αυτό, η εφαρμογή παρέχει στον επισκέπτη τον ευκολότερο και συντομότερο τρόπο για να μεταβεί στο χώρο στάθμευσης που έχει επιλέξει. Στην αντίθετη περίπτωση, αν δηλαδή η κίνηση στην πόλη γίνεται αποκλειστικά με αναπηρικό αμαξίδιο, ο χρήστης θα πρέπει να επιλέξει το κουμπί καρτέλας με τίτλο «ΚΙΝΗΣΗ ΜΕ ΑΝΑΠΗΡΙΚΟ ΑΜΑΞΙΔΙΟ». Τότε, ο επισκέπτης της εφαρμογής θα μεταβεί στην επόμενη καρτέλα στην οποία θα εμφανίζεται ξανά ο χάρτης των Χανίων και οι μπάρες αφετηρίας και προορισμού. Και σε αυτή την περίπτωση ο χρήστης πρέπει να ακολουθήσει την ίδια διαδικασία δηλώνοντας το σημείο στο οποίο βρίσκεται και το σημείο στο οποίο θέλει να καταλήξει. Στη συνέχεια, ακολουθώντας τις οδηγίες κίνησης με βάση των ειδικών ράμπων για αναπηρικά αμαξίδια, που θα του προσφέρει η εφαρμογή, μπορεί να μεταβεί γρήγορα και εύκολα στον επιθυμητό προορισμό. Όπως φαίνεται και από την παραπάνω περιγραφή, ο χάρτης κάθε καρτέλας είναι άρρηκτα συνδεδεμένος με την αντίστοιχη επιλογή, καθώς εστιάζει σε διαφορετικά ζητήματα.

Στη συνέχεια παρουσιάζονται και αναλύονται λεπτομερώς όλες οι παράμετροι, οι οποίες είναι αναγκαίες, προκειμένου να τεθεί σε λειτουργία η εφαρμογή:

Α) Χάρτης της πόλης των Χανιών: Οι χάρτες Google, είναι μια διαδικτυακή εφαρμογή υπηρεσιών χαρτογράφησης. Η τεχνολογία αυτή παρέχεται δωρεάν, για προσωπική

χρήση, από την Google. Ανάμεσα στις υπηρεσίες που προσφέρονται είναι η ιστοσελίδα των χαρτών Google, η υπηρεσία Google transit και χάρτες ενσωματωμένοι σε ιστοσελίδες τρίτων μέσω της υπηρεσίας Google Maps API. Η Google παρέχει οδικούς χάρτες, εφαρμογής δρομολόγησης για ταξίδι με ποδιά και αυτοκίνητο. Παρακάτω θα αναφερθούν λεπτομερώς οι λειτουργιές των χαρτών Google-android.

Ο χάρτης που θα χρησιμοποιηθεί για την υλοποίηση της εφαρμογής είναι ο χάρτης της πόλης των Χανιών και έχει οριοθετηθεί στο κέντρο της πόλης. Για την διευκόλυνση του χρηστή, στο χάρτη αναγράφονται κεντρικά σημεία της πόλης και σημεία στα όποια υπάρχουν θέσεις Parking για αναπήρους. Η παρακάτω εικόνα του χάρτη είναι αυτή που χρησιμοποιήθηκε για την υλοποίηση της εφαρμογής, έχει δοθεί από το δημαρχείο Χανιών και έχει αυτούσιες τις θέσεις Parking αναπήρων που χρησιμοποιήθηκαν στην καρτέλα «ΚΙΝΗΣΗ ΜΕ ΑΥΤΟΚΙΝΗΤΟ/PARKING».

Εικόνα 28: Χάρτης της πόλης των Χανιών με κόμβους τις θέσεις Parking.

Β) Μπάρα προορισμού (αναζήτησης): Μια μπάρα αναζήτησης ή πεδίο αναζήτησης είναι ένα γραφικό στοιχείο ελέγχου που χρησιμοποιείται σε προγράμματα ηλεκτρονικών υπολογιστών. Ένα κουτί αναζήτησης είναι συνήθως μια ενιαία γραμμή πλαίσιο κειμένου με την ειδική λειτουργία της αποδοχής εισόδου του χρήστη πρέπει να αναζητηθεί σε μια βάση δεδομένων.

Εδώ ο χρήστης προσθέτει το σημείο στο όποιο βρίσκεται στην επιλογή αφετηρίας και στην επιλογή προορισμού, τον προορισμό προς τον όποιο θέλει να κατευθυνθεί. Ανάλογα με την επιλογή καρτέλας ο χρήστης έχει διαφορετικές επιλογές. Όταν ο χρήστης δώσει τις απαιτούμενες πληροφορίες θα χαραχτεί στον χάρτη της πόλης η συγκεκριμένη διαδρομή. Αν, προηγουμένως, έχει επιλέξει την καρτέλα «ΚΙΝΗΣΗ ΜΕ ΑΥΤΟΚΙΝΗΤΟ/PARKING» τότε θα εμφανίσει το σημείο Parking που έχει επιλέξει για την στάθμευση του. Αν έχει επιλεγεί η καρτέλα «ΚΙΝΗΣΗ ΜΕ ΑΝΑΠΗΡΙΚΟ ΑΜΑΞΙΔΙΟ» (σε αυτή την περίπτωση αναγράφονται στο χάρτη της πόλης τα ονόματα των οδών) θα χαραχτεί η πορεία η όποια πρέπει να

ακολουθήσει ο χρήστης για να φτάσει στον προορισμό του με το αναπηρικό του αμαξίδιο. Οι πληροφορίες οδών, δρόμων κτλ. παίρνονται ως πληροφορίες από την υπηρεσία Google maps.

Γ) Επιλογή καρτέλας «ΚΙΝΗΣΗ ΜΕ ΑΥΤΟΚΙΝΗΤΟ/PARKING»: Αυτή η καρτέλα αφορά τους χρηστές που χρησιμοποιούν αναπηρικά αυτοκίνητα. Κατά την επιλογή αυτής της καρτέλας, εμφανίζεται ο παραπάνω χάρτης, στον οποίο αναγράφονται οι θέσεις Parking που προορίζονται για αναπήρους. Επιλέγοντας αφετηρία και προορισμό , και με την χρήση του αλγορίθμου dijkstra, επιτυγχάνεται η εύρεση και η χάραξη του πιο σύντομου δρόμου στο χάρτη της πόλης, λαμβάνοντας παράλληλα υπόψη την ύπαρξη σημάτων στάθμευσης αναπηρικών οχημάτων. Στο σημείο αυτό πρέπει να αναφερθεί ότι ο σκοπός είναι να βρεθεί η πιο σύντομη διαδρομή που να έχει περάσει τουλάχιστον από έναν κόμβο του αλγορίθμου dijkstra.

Εικόνα 29: Σήμανση χώρου στάθμευσης ανάπηρων

Δ) Επιλογή καρτέλας «ΚΙΝΗΣΗ ΜΕ ΑΝΑΠΗΡΙΚΟ ΑΜΑΞΙΔΙΟ»: Αυτή η καρτέλα αφορά τους χρηστές που χρησιμοποιούν αποκλειστικά τα αναπηρικά τους αμαξίδια. Με την επιλογή αυτής της καρτέλας εμφανίζεται ξανά ο χάρτης πλοήγησης αυτήν την φόρα οι κόμβοι του αλγορίθμου dijkstra είναι οι ράμπες ανάπηρων οι οποίες αναγράφονται στον χάρτη μας. Ο χρήστης πρέπει να συμπληρώσει πάλι την αφετηρία και τον προορισμό του και να μετακινηθεί στην πόλη με το αναπηρικό του αμαξίδιο. Με την βοήθεια του αλγορίθμου dijkstra επιτυγχάνεται, ξανά, η εύρεση της πιο σύντομης διαδρομής για τον χρηστή.

Ε) Πινέζες (κομβόι αλγορίθμου djikstra) : Συγκεκριμένα στην επιλογή καρτέλας «ΚΙΝΗΣΗ ΜΕ ΑΥΤΟΚΙΝΗΤΟ/PARKING» συναντάμε ένα χάρτη σημειωμένο με διάφορα σημεία, τα οποία έχουν ως σήμα 

, που δηλώνουν τις θέσεις parking στον χάρτη μας. Ενώ στην επιλογή καρτέλας «ΚΙΝΗΣΗ ΜΕ ΑΝΑΠΗΡΙΚΟ ΑΜΑΞΙΔΙΟ», αυτή τη φόρα ο χάρτης έχει σημεία αναφοράς  , τα συγκεκριμένα εικονίδια είναι και οι ράμπες (κομβόι) οπού ο χρήστης κατά την επιλογή διαδρομής, θα μετακινηθεί. Γενικά, τα σήματα αυτά αναπαριστούν τις έννοιες (κόμβους) και κάθε κόμβος έχει μια τιμή που τον χαρακτηρίζει. Οι κόμβοι προσδιορίζουν τις σχέσεις μεταξύ εννοιών, ενώ στον συγκεκριμένο χάρτη περιγράφουν πώς μια έννοια συνδέεται με μια άλλη. Η χρήση ενός εννοιολογικού χάρτη καλείται εννοιολογική χαρτογράφηση, και παρέχει τη δυνατότητα έκφρασης εννοιών με την χρήση εικόνων, σχημάτων, σημάτων κτλ. και την δυνατότητα σύνδεσης ανάμεσα των εννοιών αυτών. Ειδικότερα, τα συγκεκριμένα σήματα στον χάρτη μας, παρουσιάζουν τα σημεία στα οποία ο χρήστης θα μπορέσει να εντοπίσει τις θέσεις Parking για αναπήρους. Επίσης, χρησιμοποιούνται σαν κόμβοι για την λειτουργία του αλγόριθμου djikstra και την εύρεση του πιο σύντομου μονοπατιού κατά την επιλογή αφετηρίας και προορισμού από το χρήστη.

Ζ) Κουμπί «Toggle Transport» : Η επιλογή του συγκεκριμένου κουμπιού επιτρέπει στον χρηστή μας να κάνει εναλλαγή μεταξύ των δυο τρόπων μετακίνησης δυναμικά. Με τον συγκεκριμένο τρόπο λοιπόν, μπορούμε ενώ έχουμε επιλέξει να μετακινηθούμε με αυτοκίνητο, να δούμε και την αντίστοιχη διαδρομή για το άλλο μέσο μετακίνησης, με αναπηρικό αμαξίδιο. Έτσι δίνεται η δυνατότητα στον χρηστή να συνεχίσει την πορεία του οποιαδήποτε στιγμή και με τους δυο τρόπους.

Η) Κουμπί «Αποθήκευση σε SD» : Η επιλογή του συγκεκριμένου κουμπιού βοηθάει τον χρηστή στο να αποθηκεύει οποιαδήποτε στιγμή την διαδρομή την όποια έχει επιλέξει στην εξωτερική κάρτα αποθηκεύσεις SD την όποια διαθέτει η συσκευή του, με τον τρόπο αυτό επιτυγχάνεται να κρατάει ο χρήστης αρχείο με τις διαδρομές τις οποίες χρησιμοποιεί συχνά.

Θ) Κουμπί «Καθαρισμός διαδρομής» : Η επιλογή του συγκεκριμένου κουμπιού δίνει την δυνατότητα στον χρηστή να σβήσει τα σημεία που έχει επιλέξει για να βρει μια συγκεκριμένη διαδρομή, σε περίπτωση λανθασμένης επιλογής διαδρομής, ώστε να του δίνετε η δυνατότητα να επιλέξει εκ νέου νέα διαδρομή με διαφορετικά σημεία αφετηρίας-προορισμού.

3.3 Αλγόριθμος Dijkstra

Οι αλγόριθμοι είναι μια επιστήμη της εξυπνάδας. Μια φυσική εκδήλωση της λογικής reasoning- μαθηματική επαγωγή, ειδικότερα ένας καλός αλγόριθμος χαρακτηρίζεται ως «φευγαλέος». Είναι το στιγμιότυπο στο κέντρο της λύσης του προβλήματος. Ένα χάος των ιδιοτήτων και των σχέσεων γίνεται μια απλή αναδρομική σχέση, μια ενιαία γραμμή αναδρομικού βήματος που παράγει ένα επάρατο χάος και πολυπλοκότητας. Και για να δούμε μέσα από την βαθειά πολυπλοκότητα, χρειάζεται εξυπνάδα. Ήταν ο πρωτοπόρος προγραμματισμού Έντσγκερ Ντάικστρα, ο όποιος το κατάφερε. Ο αλγόριθμος dijkstra, τον οποίο επινόησε, παραμένει ένας από τους πιο έξυπνους αλγορίθμους στην επιστήμη των υπολογιστών. Η απλότητα και η κομψότητα στα μαθηματικά, όπως χαρακτήρισε τον συγκεκριμένο αλγόριθμο, κάθε πολύπλοκο πρόβλημα έχει έναν προσιτό τρόπο επίλυσης και τα μαθηματικά είναι το εργαλείο για να τον βρει και να τον εκμεταλλευτεί αυτόν τον τρόπο.

Το 1956, ο Ντάικστρα έθεσε σε εφαρμογή τον αλγόριθμο dijkstra στο ARMAC, μια παράλληλη υπολογιστική μηχανή που βασίζεται στην ολλανδική μέθοδο «Μαθηματική κέντρου». Το βασικό ερώτημα που απάντησε ο συγκεκριμένος αλγόριθμος ήταν: “ Πως θα βρεθεί η συντομότερη διαδρομή μεταξύ δυο πόλεων στην Ολλανδία, χρησιμοποιώντας μειωμένο τον χάρτη πορείας της Ολλανδίας, στην οποία έχουν επιλεχθεί 64 πόλεις.”

Πλέον χάρτες, όπως του Google, χρησιμοποιούν αυτόν τον αλγόριθμο. Έτσι εμείς, οι χρήστες, χωρίς να σκεφτούμε την πολυπλοκότητα του προβλήματος, αποκτούμε την πληροφορία για το πρόβλημα της συντομότερης διαδρομής.

3.3.1 Περιγραφή Αλγορίθμου Dijkstra

Σκοπός του αλγορίθμου αυτού είναι να βρει, σε ένα δίκτυο, τον ταχύτερο δρόμο δηλαδή την διαδρομή με το μικρότερο μήκος που θα ακολουθήσει μια κλήση για να φτάσει στον προορισμό της.

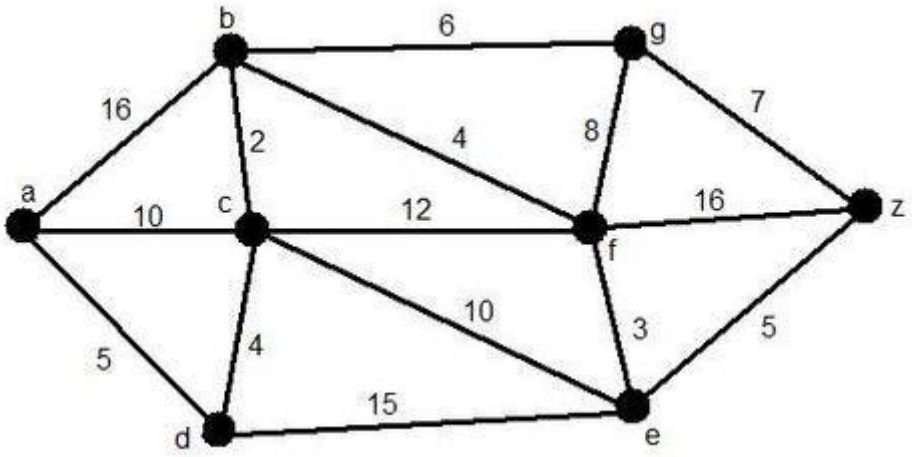
Χρησιμοποιούμε δυο βοηθητικούς πινάκες με την ονομασία προσωρινές τιμές (d) και μόνιμες τιμές (r). Για κάθε δίκτυο υπάρχει ένας πινάκας, ο πινάκας κόστους, τα στοιχειά του οποίου εκφράζουν κάποια όρια, όπως είναι η απόσταση, το κόστος μεταφοράς, η χωρητικότητα κτλ. Ορίζουμε αρχικά ποιος θα είναι ο αρχικός μας κόμβος και έπειτα

ορίζουμε ποιος θα είναι ο τελικός μας κόμβος. Θεωρούμε ότι εφόσον δεν υπάρχει ζεύξη δηλαδή μονοπάτι μεταξύ δυο διαφορετικών κόμβων τότε η αντίστοιχη τιμή τους στον πινάκα κόστους θα είναι άπειρη και θα παριστάνεται από μια μεγάλη τιμή.

Στο σημείο αυτό θέτουμε το μόνιμο πινάκα με πολλές τιμές και ακλουθούμε συγκεκριμένη διαδικασία μέχρι το στοιχειό που αντιστοιχεί στον τέλος της διαδρομής να έχει τιμή διάφορη του απείρου. Ο αλγόριθμος θέτει σε όλους τους κόμβους τιμές μόνιμες ή προσωρινές. Εδώ όλοι οι κομβόι πλην του αρχικού θα πάρουν μια προσωρινή τιμή που μεταγλωττίζεται ως την απόσταση το κόμβου έναρξης και του αντιστοίχου συγκεκριμένου κόμβου. Πάντα στον κόμβο έναρξης δίνουμε την τιμή μηδέν. Όποιος κόμβος δεν συνδέεται άμεσα με τον κόμβο έναρξης δίνεται σ αυτόν μια προσωρινή τιμή με άπειρες τιμές, ενώ οι υπόλοιποι κομβόι παίρνουν την προσωρινή τιμή. Έπειτα όταν ένας κόμβος ανήκει στην μικρότερη διαδρομή τότε η τιμή του γίνεται μόνιμη. Τέλος για να βρούμε τον κόμβο με τον όποιο βρίσκεται πιο κοντά στον κόμβο έναρξης βρίσκουμε την μικρότερη από τις προσωρινές τιμές και την μετατρέπουμε σε μόνιμη. Από αυτό το σημείο και έπειτα ο αλγόριθμος επαναλαμβάνεται για τα παρακάτω δυο σταδία έτσι ώστε ο τελικός κόμβος να έχει μόνιμη τιμή. Σταδία:

1. Αναζητώ τους κόμβους που έχουν απομένει με προσωρινές τιμές, έπειτα συγκρίνω κάθε προσωρινή τιμή με το άθροισμα της τελευταίας μόνιμης τιμής και της απάθειας απόστασης του κόμβου, με την προσωρινή τιμή, από τον υπό εξέταση κόμβο. Με αυτό τον τρόπο η ελάχιστη από τις αποστάσεις γίνεται μόνιμη τιμή για τον συγκεκριμένο κόμβο.
2. Διαλέγω τις μικρότερες τιμές που έχουν απομείνει και τις μετατρέπω σε μόνιμες. Αν η τιμή του κόμβου τερματισμού γίνει μόνιμη ο αλγόριθμος λαμβάνει τέλος, αλλιώς επιστρέφει στο βήμα 1 και ξανά υπολογίζει το μονοπάτι με τις μόνιμες τιμές.

3.3.2 Παράδειγμα Αλγορίθμου Dijkstra και Πίνακας με Τιμές Κόμβων



Εικόνα 30: Παράδειγμα Σχήματος Αλγορίθμου Dijkstra

Επανάληψη	S (επεξεργασμέ	(d[b], prev[b	(d[c], prev[c	(d[d], prev[d	(d[e], prev[e	(d[f], prev[f	(d[g], prev[g	(d[z], prev[z	Κόμβος με το ελάχισ

	νοι)	**])**	**])**	**])**	**])**	**])**	**])**	**])**	**το d[*] στην Q**
Αρχικοποίηση	{-}	(∞, -)	(∞, -)	(∞, -)	(∞, -)	(∞, -)	(∞, -)	(∞, -)	a
1	{a}	(16, a)	(10, a)	**(5, a)**	(∞, -)	(∞, -)	(∞, -)	(∞, -)	d
2	{a, d}	(16, a)	**(9, d)**	(5, a)	(20, d)	(∞, -)	(∞, -)	(∞, -)	c
3	{a, d, c}	**(11, c)**	(9, d)	(5, a)	(19, c)	(21, c)	(∞, -)	(∞, -)	b
4	{a, d, c, b}	(11, c)	(9, d)	(5, a)	(19, c)	**(15, b)**	(17, b)	(∞, -)	f
5	{a, d, c, b, f}	(11, c)	(9, d)	(5, a)	(18, f)	(15, b)	**(17, b**)	(31, f)	g
6	{a, d, c, b, f, g}	(11, c)	(9, d)	(5, a)	**(18, f**)	(15, b)	(17, b)	(24, g)	e
7	{a, d, c, b, f, g, e}	(11, c)	(9, d)	(5, a)	(18, f)	(15, b)	(17, b)	**(23, e)**	z

Συγκεκριμένα, στο παραπάνω παράδειγμα του σχήματος αλγορίθμου dijkstra και έπειτα στο πινάκα ακλουθεί τα έξης βήματα:

Στο σχήμα έχουμε ως αρχικό κόμβο, τον κόμβο α και ως τελικό κόμβο τον κόμβο z. Αρχίζοντας από τον αρχικό κόμβο εξετάζουμε όλες τις πιθανές τιμές των γειτονικών κόμβων, δίνοντας σε όλους τους γειτονικούς κόμβους προσωρινές τιμές. Στο σημείο αυτό βλέπουμε ότι από τον αρχικό κόμβο, ο επόμενος κόμβος με την μικρότερη τιμή από τους b (16),c (10),d (5) κόμβους είναι ο d κόμβος, οπού η τιμή του γίνεται μόνιμη τιμή και είναι ο επόμενος κόμβος στην διαδρομή μας. Έπειτα από το κόμβο d(5) εξετάζουμε τους επομένους υποψηφίους κόμβους προσθέτοντας τον κόμβο που εξετάσαμε με τους νέους και έχουμε, d(5+4) για τον c κόμβο και d(5+15) για τον e κόμβο, στο σημείο αυτό κρατάμε το μικρότερο άθροισμα το οποίο είναι d(5+4) όποτε ο επόμενος κόμβος οπού η τιμή του θα γίνει μόνιμη είναι ο c κόμβος. Επομένη κίνηση είναι να συνεχίσουμε να εξετάζουμε τους γειτονικούς κόμβους από την τελευταία μόνιμη τιμή κόμβου c, οπού εδώ έχουμε c(9+2) για τον d κόμβο, c(9+12) για τον f κόμβο και c(9+10) για τον e κόμβο. Πάλι θα εξετάσουμε την διαδρομή με το μικρότερο άθροισμα, έτσι θα επιλέξουμε τον κόμβο d(11) σαν μόνιμη τιμή πλέον. Συνεχίζουμε εξετάζοντας τον νέο μας κόμβο με τους γειτονικούς από αυτόν κόμβο και

έχουμε τις έξης νέες τιμές d(11+6) για τον κόμβο g, και d(11+4) για τον κόμβο f και πάλι επιλεγούμε το μικρότερο άθροισμα κόμβων που στο συγκεκριμένο σημείο είναι ο f. Από την νέα μόνιμη τιμή f κόμβο θα συνεχίσουμε εξετάζοντας τους νέους γειτονικούς κόμβους και έχουμε f(15+8) για τον κόμβο g, f(15+16) για τον κόμβο z και f(15+3) για τον κόμβο e, οπού και έχοντας το μικρότερο άθροισμα γίνετε η νέα μόνιμη τιμή των κόμβων μας. Στο σημείο, αυτό έχουμε πετύχει τον στόχο μας να βρούμε το μονοπάτι με το μικρότερο άθροισμα κόμβων, οπού μας αποκαλύπτει και την συντομότερη διαδρομή, καθώς μετά τον κόμβο e, φτάνουμε στον τελικό κόμβο που θέλαμε (z κόμβος). Το μονοπάτι που καλύψαμε και φαίνεται και στον πινάκα του παραδείγματος είναι: α(0) → d(5) → c(9) → b(11) → f(15) → e(18) → z(23).

3.4 Χάρτες Google Για Smartphones Και Η Χρήση Του GPS

Οι χάρτες Google (Google maps) είναι μια υπηρεσία χαρτών, τούς οποίους μπορεί ο καθένας να προσβάλει στο πρόγραμμα περιήγησης ιστού που διαθέτει. Ανάλογα με την τοποθεσία οπού βρίσκεσαι, μπορεί να δει βασικούς ή προσαρμοσμένους χάρτες, καθώς και πληροφορίες τοπικών επιχειρήσεων, όπως τοποθεσίες επιχειρήσεων, πληροφορίες επικοινωνίας και οδηγίες οδήγησης από και προς τα εκεί. Κάντε κλικ και μετακινήστε τους χάρτες για να δείτε τις επιθυμητές τοποθεσίες που μπορείτε να μεγεθύνεται ή να σμικρύνετε και να τους αλλάξετε την οπτική γωνιά.

Η Google δημιούργησε το Google MAPS API τον Ιούλιο του 2005 για να επιτρέπει στου προγραμματιστές να ενσωματώσουν το Google maps στις εφαρμογές τους. Η υπηρεσία διατίθεται δωρεάν με την απόκτηση ενός κλειδιού (API-KEY) για την εκάστοτε εφαρμογή από την Google. Πρόκειται για ένα JavaScript API (πράγμα που επιτρέπει την ενσωμάτωση του σε όλες τις εφαρμογές δίχως ιδιαίτερες απαιτήσεις από την μεριά του server) που αποτελείται από ένα σύνολο κλάσεων (classes), ιδιοτήτων (properties), συμβάντων (events), μεθόδων (methods) δομημένα με τέτοιο τρόπο ώστε να κάνουν εφικτή την πρόσβαση στα δεδομένα και να επιτρέπουν την ανάκτηση τους καθώς και τον εμπλουτισμό τούς πριν την απεικόνιση τους.

Οι χάρτες Google για τα κινητά είναι μια εφαρμογή σε java η οποία εκδόθηκε από την Google το 2006, με σκοπό να τρέχει σε οποιαδήποτε βασισμένο σε java κινητό ή φορητή συσκευή. Πολλά από τα χαρακτηριστικά των χαρτών Google υπάρχουν σε αυτήν την εφαρμογή. Το 2008 εκδόθηκε και η δεύτερη έκδοση της εφαρμογής αυτής, οι χάρτες Google για κινητά 2.0. Σε αυτήν την καινούρια έκδοση η Google υλοποίησε μια υπηρεσία παρόμοια με το παγκόσμιο σύστημα εντοπισμού θέσης (GPS-like) η οποία όμως δεν χρησιμοποίησε δεκτή GPS. Η λειτούργει «η τοποθεσία μου» (my location), λειτούργει με την χρήση της τοποθεσίας GPS του κινητού αν αυτή είναι διαθέσιμη. Αυτή η εφαρμογή λειτούργει με τον έξης τρόπο:

Το λογισμικό ψάχνει για το πλησιέστερο ασύρματο δίκτυο, έπειτα ψάχνει για την τοποθεσία του δικτύου, χρησιμοποιώντας μια βάση δεδομένων με γνωστά σύρματα δίκτυα. Ο προσδιορισμός θέσης του δικτύου γίνεται με μια τριγωνική μέθοδο οπού μελετούνται οι διαφορετικές εντάσεις του σήματος από διαφορετικούς πομπούς, που βρίσκονται στο σταθμό δικτύου. Έπειτα, χρησιμοποιώντας την τοποθεσία η οποία ανακτάται από μια διαδικτυακή βάση δεδομένων αλλάζει « η τοποθεσία μου», ορίζοντας έτσι την τρέχουσα θέση του χρηστή. Η μέθοδος εντοπισμού ασύρματου δικτύου συμβαίνει βρίσκοντας τις πιο κοντινές δυναμικές ζώνες wifi. Έπειτα, η τοποθεσία αυτή των ζωνών που ανακτάται από αντίστοιχη βάση δεδομένων που χρησιμοποιείται για την εύκολη εύρεση της θέσης του χρηστή. Η εφαρμογή

κάνει χρήση υπηρεσιών βασισμένες στο GPS, υπηρεσίες βασισμένες στους πομπούς των δικτύων. Στον χάρτη φαίνονται οι δρόμοι και η εκτιμώμενη θέση του κινητού τηλέφωνου, μπορεί βεβαία να μην είναι απόλυτα σωστή καθώς περιβάλλεται από έναν κύκλο που απεικονίζει το εύρος χώρου στο οποίο μπορεί να βρίσκεται το κινητό. Το εκτιμώμενο εύρος χώρου, υπολογίζεται με βάση την ισχύ του σήματος του τηλέφωνου που υποδεικνύει ποσό κοντά είναι η συσκευή στον σταθμό δικτύου αρά και στον πομπό.

Το Android προσφέρει στους προγραμματιστές και κατά συνέπεια στους χρηστές την δυνατότητα ενσωματώσεις κάποιου χάρτη στην εφαρμογή τους. Αυτό μπορεί να επιτευχτεί με δυο τρόπους. Ο πρώτος είναι και ποιο απλός είναι να ενσωματώσεις στην οθόνη ένα στοιχειό το οποίο καλείται από μια κλάση view. Στο view, ο προγραμματιστής μπορεί να απεικονίσει έναν χάρτη από το Google maps.

Η Google, παράλληλα, θέλοντας να δώσει περισσότερη προγραμματιστική ελευθερία δημιούργησε ένα άλλο στοιχειό διεπιφάνειας χρηστή, το «Map view» οπού και είναι ο δεύτερος τρόπος. Το Map view σε συνδυασμό με την κλάση Map Activity και τις διεπαφές mapping APIs, είναι ένα ισχυρό εργαλείο σε όποιον επιθυμεί να δημιουργήσει καινοτομίες και πρωτοποριακές εφαρμογές με την χρήση των χαρτών. Με αυτό τον τρόπο, ο κατασκευαστής μπορεί να δουλέψει πάνω σε ήδη υπάρχων χάρτη και να τους τροποποιήσει όπως επιθυμεί εκείνος με την δυνατότητα να χειριστεί τις αλληλεπιδράσεις του χρηστή με τον χάρτη, να τοποθετήσει δικά του δεδομένα κτλ.

Εικόνα 31: Γραφική Αναπαράσταση Google Map

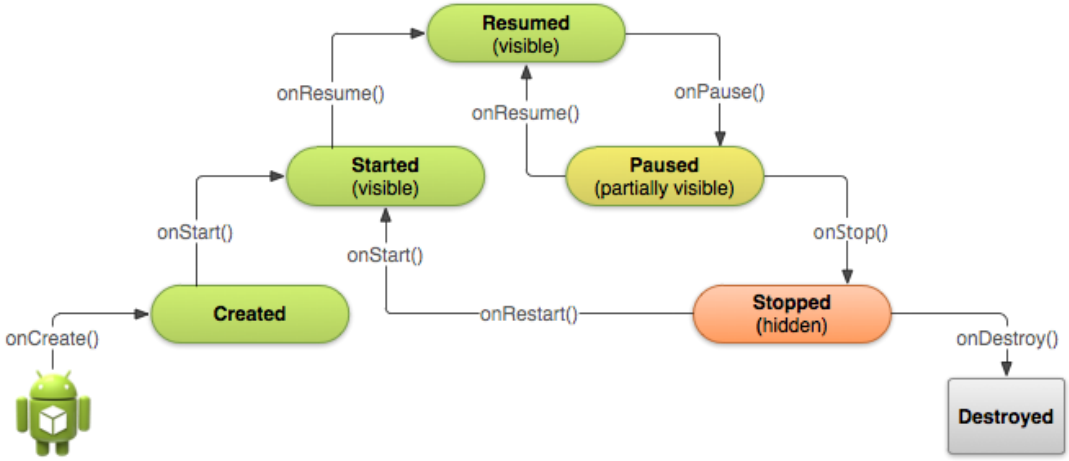
4 Προγραμματισμός και Κωδικοποίηση

Σε αυτό το κεφάλαιο θα επικεντρωθούμε στα επιμέρους κομμάτια κώδικα που συνθέτουν την εφαρμογή μας, στις λειτουργίες τους καθώς και στο τρόπο που αλληλεπιδρούν. Τα πιο σημαντικά αρχεία που περιλαμβάνονται και υλοποιούν την εφαρμογή στην πλατφόρμα του Android Studio, είναι οι κλάσεις Java των Activity και τα xml αρχεία. Ας ξεκινήσουμε όμως πρώτα να αναλύσουμε το κάθε κομμάτι ξεχωριστά.

4.1.1 Τι είναι η Activity

Μια εφαρμογή Android αποτελείται συνήθως από πολλαπλές Activitys που είναι συνδεδεμένες η μία με την άλλη. Κάθε Activity που βρίσκεται μέσα σε μία εφαρμογή είναι ένα στιγμιότυπο της οθόνης και έχει ένα μοναδικό σκοπό. Τυπικά, σε μια εφαρμογή έχει οριστεί μία Activity ως η «κύρια» Activity, η οποία παρουσιάζεται στο χρήστη όταν ξεκινάμε την εφαρμογή για πρώτη φορά και κάθε Activity μπορεί στη συνέχεια να αρχίσει μια άλλη Activity, προκειμένου να εκτελέσει διάφορες ενέργειες. Κάθε φορά που μια νέα Activity ξεκινά, η προηγούμενη δραστηριότητα έχει σταματήσει, αλλά το σύστημα διατηρεί την δραστηριότητα σε μια στοίβα. Έτσι, κάθε χρονική στιγμή μόνο μία εργασία είναι στο προσκήνιο.

Όταν μια Activity έχει διακοπεί λόγω μίας νέας, τότε καλούνται κάποιοι μέθοδοι και όλη αυτή η διαδικασία ονομάζεται κύκλος ζωής του Activity (Lifecycle Activity). Υπάρχουν διάφορες μέθοδοι επανάκλησης που θα μπορούσε να λάβει μια Activity όπως η onCreate() (καλείτε όταν δημιουργούμε κάποιο Activity για πρώτη φορά), onStart() (καλείτε όταν ξεκινήσει η Activity), onPause() (τρέχει όταν γίνεται παύση της συγκεκριμένης Activity καθώς δεν βρίσκεται πλέον στο προσκήνιο και μία καινούρια ξεκινάει να εκτελείται), onResume() (καλείται όταν ξανατρέξουμε τη συγκεκριμένη Activity επαναφέροντάς την πάλι πρώτη στην ιεραρχία της στοίβας), onStop() (καλείται όταν γίνεται διακοπή της Activity) και σε περίπτωση που θέλουμε να την επαναφέρουμε από την παρούσα κατάσταση εκτελείται η μέθοδος onRestart() και στη συνέχεια η onStart(). Τέλος, η onDestroy() (καλείτε για να τερματίσει ένα Activity επειδή ολοκλήρωσε τον κύκλο ζωής του).

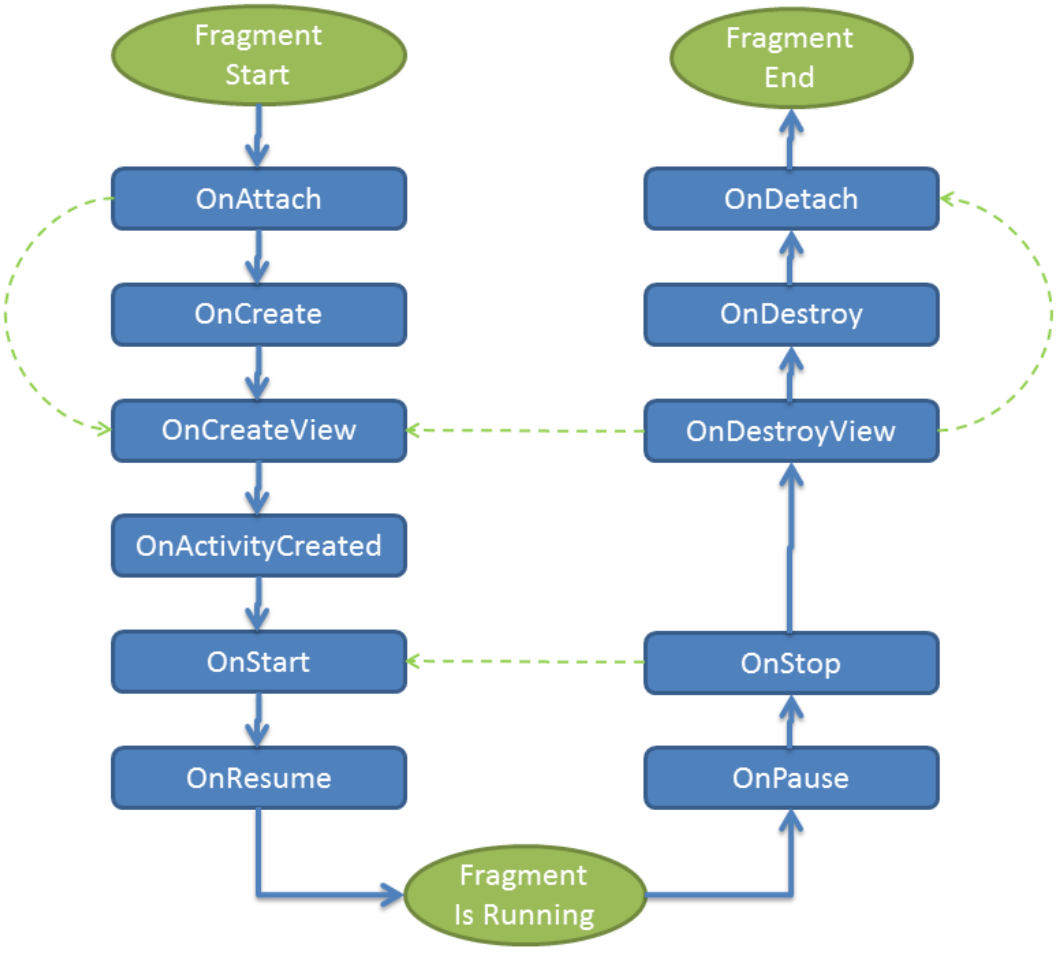


Εικόνα 32: Ο Κύκλος Ζωής Μιας Activity

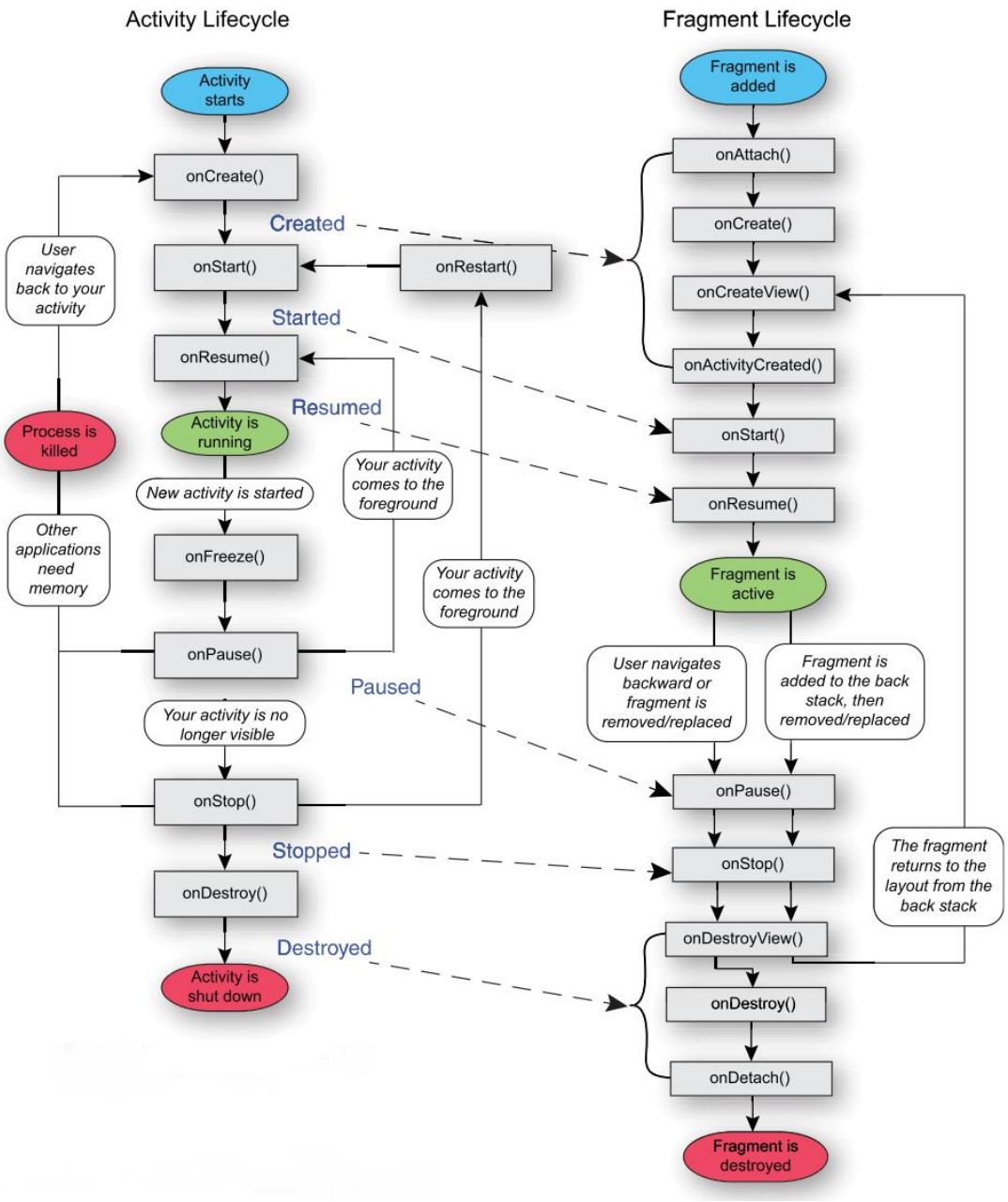
4.1.2 Τι είναι το Fragment

Ένα Fragment αντιπροσωπεύει μια συμπεριφορά ή ένα τμήμα της διεπαφής χρήστη σε μια Activity. Μπορούμε να συνδυάσουμε πολλά Fragments σε μια ενιαία Activity και να ξαναχρησιμοποιήσουμε ένα Fragment σε πολλαπλές Activitys. Μπορούμε να θεωρήσουμε το Fragment ως ένα κομμάτι μιας Activity, το οποίο έχει το δικό του κύκλο ζωής και αποκτά τα δικά του δεδομένα εισόδου, τα οποία μπορούμε να προσθέσουμε ή να αφαιρέσουμε όταν η Activity βρίσκεται σε λειτουργία.

Ένα Fragment πρέπει πάντα να βασίζεται σε μια Activity και ο κύκλος ζωής του επηρεάζεται άμεσα από τον κύκλο ζωής της Activity. Για παράδειγμα, όταν μία Activity διακοπεί ή καταστραφεί, σταματούν ή καταστρέφονται όλα τα Fragments που υπάρχουν μέσα σε αυτή. Παρ' όλα αυτά, όταν μια Activity βρίσκεται σε λειτουργία (είναι στην επαναληπτική κατάσταση του κύκλου ζωής της) μπορούμε να διαχειριστούμε κάθε Fragment ξεχωριστά, να του δώσουμε δεδομένα ή ακόμα και να το αφαιρέσουμε ολοκληρωτικά από την Activity.



Εικόνα 33: Ο Κύκλος Ζωής Ενός Fragment



Εικόνα 34: Σχήμα Λειτουργίας και Αλληλεπίδρασης Activity - Fragment

4.2 Επιμέρους Κώδικας

4.2.1 Maps activity

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_maps);
```

```
    // Initializing
    markerPoints = new ArrayList<LatLng>();
    initMarkers = new ArrayList<LatLng>();
    parkingMarkers = new ArrayList<LatLng>();
    // Getting reference to SupportMapFragment of the
activity_main
    SupportMapFragment fm =
(SupportMapFragment)getSupportFragmentManager().findFragm
entById(R.id.map);

    // Getting Map for the SupportMapFragment
    mMap = fm.getMap();
```

Η μέθοδος onCreate καλείται για την αρχικοποίηση του χάρτη της εφαρμογής μας. Στη συνέχεια χρησιμοποιούμε 3 arraylist για την αποθήκευση των Markers στον χάρτη τα οποία είναι:

- MarkerPoints: Για την αποθήκευση των σημείων επιλογής του χρηστή.

- InitMarkers: Για να κρατάμε τις θέσεις όπου υπάρχουν ράμπες.
- ParkingMarkers: Για να κρατάμε τις θέσεις παρκινγκ.

```
Switch switch1 = (Switch)findViewById(R.id.switch1);
```

Αλλαγή στην επιλογή μεταξύ μετακίνησης με «αυτοκίνητο» ή με «αμαξίδιο» σε οποιοδήποτε χάρτη και αν βρισκόμαστε αρχικά.

```
if (markerPoints.size() >= 2) {
    LatLng origin = markerPoints.get(0);
    LatLng dest = markerPoints.get(1);

    // Getting URL to the Google Directions API
    String url = getDirectionsUrl(origin, dest);

    DownloadTask downloadTask = new DownloadTask();

    // Start downloading json data from Google Directions
API
    downloadTask.execute(url);
}
```

Όταν έχουμε 2 markers, τότε βρίσκουμε την κοντινότερη απόσταση μεταξύ των 2 σημείων (με βάση κάποιους κανόνες που θα εξηγήσουμε αργότερα).

```
MarkerOptions options1 = new MarkerOptions();
mMap.setMyLocationEnabled(true);
LatLng point1 = new
LatLng(35.515346325911665,24.03094958513975);

initMarkers.add(point1);
```

```
LatLng point2 = new
LatLng(35.5104091159649,24.030606262385845);
    initMarkers.add(point2);
LatLng point3 = new
LatLng(35.507802705596426,24.025781974196434);
    initMarkers.add(point3);

LatLng point4 = new
LatLng(35.510734160790605,24.021793864667416);
    initMarkers.add(point4);
LatLng point5 = new
LatLng(35.51357353923431,24.016733206808567);
    initMarkers.add(point5);
LatLng point6 = new
LatLng(35.50367048479257,24.020578488707542);
    initMarkers.add(point6);
```

Εδώ γίνεται η αρχικοποίηση των σημείων (points) στο χάρτη.

```
for (int i = 0; i <initMarkers.size() ; i++) {
    if (initMarkers.get(i) != null) {
        options1.position(initMarkers.get(i));

options1.icon(BitmapDescriptorFactory.fromResource(R.draw
able.parking));
        mMap.addMarker(options1);
    }
}

for (int i = 0; i <parkingMarkers.size() ; i++) {
    if (parkingMarkers.get(i) != null) {
        options1.position(parkingMarkers.get(i));

options1.icon(BitmapDescriptorFactory.fromResource(R.draw
able.bmpp));
        mMap.addMarker(options1);
    }
}
```

Για κάθε σημείο στο χάρτη, βάζουμε συγκεκριμένο εικονίδιο που δηλώνει την χρησιμότητά του.

```
String location = "Chania, Crete";
Geocoder geocoder = new Geocoder(this);
try {
    addressListZoom =
geocoder.getFromLocationName(location,1);

} catch (IOException e) {
    e.printStackTrace();
```

```
}

Address address = addressListZoom.get(0);

LatLng latLng = new LatLng(address.getLatitude() ,
address.getLongitude());
mMap.animateCamera(CameraUpdateFactory.newLatLngZoom(latL
ng,11));
```

Η εφαρμογή περιλαμβάνει τον χάρτη και τον decoder. Αυτά τα δυο χαρακτηριστικά μαζί είναι υπεύθυνα για το Geocoding, την ένδειξη των γεωγραφικών συντεταγμένων στον χάρτη. Η ένδειξη των σημείων γίνεται με την βοήθεια των δεικτών (markers). Ο χρήστης έχει την δυνατότητα αλληλεπίδρασης των δεικτών, καθώς και την δυνατότητα αλλαγής ή και διαγράφης κάποιου επιλεγμένου σημείου στον χάρτη. Η απεικόνιση των σημείων (δεικτών) αποτελεί χαρακτηριστικό του Google maps. Ιδιαίτερη σημασία στην εφαρμογή μας, και σε συνδυασμό με όσα προαναφέρθηκαν, είναι το αντικείμενο χρήσης του geocoding και reverse geocoding. Το geocoding εφαρμόζεται προκειμένου να καταστεί εύκολη η εύρεση ενός κωδικοποιημένου σημείου σε μια αναγνώσιμη περιοχή / οδό / διεύθυνσης, η όποια αποσκοπεί στην μετατροπή των διευθύνσεων σε γεωγραφικές συντεταγμένες (γεωγραφικό πλάτος και μήκος). Οι συντεταγμένες αυτές μπορούν να χρησιμοποιηθούν για να τοποθετήσει κανείς δείκτες πάνω στον χάρτη. Επιπλέον, υπάρχει η υπηρεσία που επιτρέπει την εκτέλεση της αντίστροφης λειτουργιάς (μετατροπή συντεταγμένων σε διευθύνσεις). Η διαδικασία αυτή είναι γνωστή ως reserve geocoding. Η λειτουργία του reverse geocoding είναι απαραίτητη για την διόρθωση ενός σημείου (δείκτη) στον χάρτη. Η αλλαγή ενός σημείου (δείκτη) από τον χρήστη βοηθάει στην απόκτηση των νέων συντεταγμένων που θα έχει ο νέος δείκτης πάνω στον χάρτη.

Μόλις ξεκινήσει ο χάρτης έχουμε ως σημείο αναφοράς την πόλη των Χανίων και κάνουμε το απαραίτητο zoom in. Κάθε φορά που θα πατήσουμε ένα σημείο στον χάρτη καλείται η εξής μέθοδος:

```
@Override
public void onMapClick(LatLng point) {
```

Η οποία βάζει το επιθυμητό χρώμα στο σημείο που πατήσαμε και το εμφανίζει στον χάρτη.

```
/**
 * For the start location, the color of marker is GREEN
and
 * for the end location, the color of marker is RED.
 */
if(markerPoints.size()==1){

options.icon(BitmapDescriptorFactory.defaultMarker(Bitmap
DescriptorFactory.HUE_GREEN));
}else if(markerPoints.size()==2){

options.icon(BitmapDescriptorFactory.defaultMarker(Bitmap
DescriptorFactory.HUE_YELLOW));
```

```
}

// Add new marker to the Google Map Android API V2
mMap.addMarker(options);
```

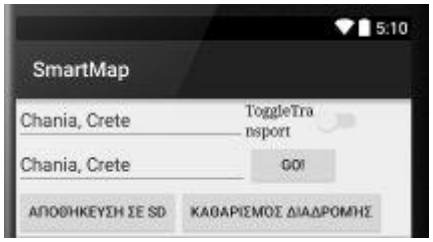
Επιπλέον, γίνεται έλεγχος για το πόσα σημεία έχουμε πατήσει στο χάρτη. Αν είναι 2, τότε φτιάχνουμε ένα URL το οποίο το στέλνουμε στο API της Google, ούτος ώστε να μας δώσει όλες τις σωστές (πιθανές) διαδρομές:

```
if(markerPoints.size() >= 2){
    LatLng origin = markerPoints.get(0);
    LatLng dest = markerPoints.get(1);

    // Getting URL to the Google Directions API
    String url = getDirectionsUrl(origin, dest);

    DownloadTask downloadTask = new DownloadTask();

    // Start downloading json data from Google Directions
API
    downloadTask.execute(url);
}
```

Στην κύρια οθόνη της εφαρμογής μας εκτός από τον διαθέσιμο χάρτη, πάνω αριστερά υπάρχει ένα κουμπί αναζήτησης χωρισμένο σε δύο πεδία :

όπου μπορούμε να πληκτρολογήσουμε την αρχική μας θέση και τον προορισμό μας, χωρίς να χρειαστεί να επιλέξουμε τα σημεία αυτά πατώντας πάνω στο χάρτη. (Μας εξυπηρετεί στην περίπτωση που δεν γνωρίζουμε την οδό στην όποια θέλουμε να μεταβούμε).

Ακριβώς από κάτω υπάρχουν 2 κουμπιά «Αποθήκευση σε SD» και «Καθαρισμός διαδρομής». Το πρώτο κουμπί μας βοηθάει στο να αποθηκεύσουμε ένα στιγμιότυπο της οθόνης και της διαδρομής στην εξωτερική κάρτα αποθήκευσης (αν υπάρχει), προκειμένου να μπορεί ο χρήστης να αποθηκεύσει και να ξαναδεί κάποια από τις διαδρομές που έχει επιλέξει κατά καιρούς.



Ενώ το δεύτερο κουμπί μας επιτρέπει να σβήσουμε τα σημεία και τη διαδρομή που έχουμε επιλέξει σε περίπτωση λάθους (καθαρισμός του χάρτη), ωστε να μπορούμε να

επιλέξουμε εκ νέου τα σημεία και τη διαδρομή που χρειαζόμαστε. Την αποθήκευση του στιγμιότυπου την αναλαμβάνει η παρακάτω μέθοδος η οποία παίρνει το «μονοπάτι» που βρίσκεται η εξωτερική κάρτα και με την σειρά της αποθηκεύει εκεί το στιγμιότυπο σαν εικόνα Bitmap υψηλής ανάλυσης.

```
public void save Bitmap(Bitmap bitmap) {
    File imagePath = new
File(Environment.getExternalStorageDirectory() +
"/screenshot.png");
    FileOutputStream fos;
    try {
        fos = new FileOutputStream(imagePath);
        bitmap.compress(CompressFormat.JPEG, 100, fos);
        fos.flush();
        fos.close();
    } catch (FileNotFoundException e) {
        Log.e("GREC", e.getMessage(), e);
    } catch (IOException e) {
        Log.e("GREC", e.getMessage(), e);
    }
}
```

Στη συνέχεια πατώντας το κουμπί αναζήτησης GO! καλείται η μέθοδος:

```
public void onSearch(View view){
```

η οποία αυτόματα βάζει εικόνες στις δύο τοποθεσίες και επιπρόσθετα μας καθοδηγεί για το ποιά διαδρομή μας «συμφέρει» να ακολουθήσουμε (φτιάχνοντας με παρόμοιο τρόπο ένα URL και στέλνοντάς το στο API της Google). Υπεύθυνη για την κατασκευή του URL είναι η μέθοδος:

```
private String getDirectionsUrl(LatLng origin,LatLng
dest){
```

η οποία δέχεται ως παραμέτρους την αρχική μας θέση και τον προορισμό.

```
// Origin of route
String str_origin =
"origin="+origin.latitude+","+origin.longitude;

// Destination of route
String str_dest =
"destination="+dest.latitude+","+dest.longitude;

// Sensor enabled
String sensor = "sensor=false";

// Building the parameters to the web service
parameters = str_origin+"&"+str_dest+"&"+sensor;
```

Έπειτα, ανάλογα με το ποιά απο τις δύο επιλογές έχουμε επιλέξει (με αμαξίδιο ή με αυτοκίνητο), βρίσκει το κοντινότερο parking / ράμπα αναπήρων στην αρχική μας θέση (αν έχουμε επιλέξει μετακίνηση με αμαξίδιο), αλλιώς βρίσκει το κοντινότερο σημείο σε σχέση με το σημείο προορισμού μας.

```
if (walking) {


    String mode = "mode=walking";

double
less=Math.sqrt(Math.abs(Math.pow((initMarkers.get(0).lati
tude - origin.latitude),2) +
Math.pow((initMarkers.get(0).longitude -
origin.longitude),2)));
    int index=0;

    // Waypoints
    String waypoints = "";
    for(int i=1;i<initMarkers.size();i++){
        LatLng point  = (LatLng) initMarkers.get(i);
        if(less>(double) Math.sqrt(
Math.abs(Math.pow((point.latitude -origin.latitude ),2) +
Math.pow((point.longitude - origin.longitude), 2)))) {
            less = (double) Math.sqrt(
Math.abs(Math.pow((point.latitude -origin.latitude ),2) +
Math.pow((point.longitude - origin.longitude),2)));
            index = i;
        }
    }

    waypoints =
"waypoints="+initMarkers.get(index).latitude + "," +
initMarkers.get(index).longitude;
    parameters = str_origin + "&" + str_dest + "&" +
sensor + "&" + waypoints+"&"+mode;

}else{

    double
less=Math.sqrt(Math.abs(Math.pow((initMarkers.get(0).lati
tude - dest.latitude),2) +
Math.pow((initMarkers.get(0).longitude -
dest.longitude),2)));
    int index=0;

    // Waypoints
    String waypoints = "";
    for(int i=1;i<parkingMarkers.size();i++){
```

```
        LatLng point  = (LatLng) parkingMarkers.get(i);
        if(less>(double) Math.sqrt(
Math.abs(Math.pow((point.latitude -dest.latitude ),2) +
Math.pow((point.longitude - dest.longitude), 2)))) {
            less = (double) Math.sqrt(
Math.abs(Math.pow((point.latitude -dest.latitude ),2) +
Math.pow((point.longitude - dest.longitude),2)));
            index = i;
        }
    }
    waypoints =
"waypoints="+parkingMarkers.get(index).latitude + "," +
parkingMarkers.get(index).longitude;
    parameters = str_origin + "&" + str_dest + "&" +
sensor + "&" + waypoints;
```

Έπειτα βάζει αυτό το σημείο στο URL που θέλουμε να στείλουμε και φτιάχνει το τελικό URL

```
// Output format
String output = "json";

// Building the url to the web service
String url =
"https://maps.googleapis.com/maps/api/directions/"+output
+"?"+parameters;

return url;
```

Μόλις τελειώσει η κλήση της παραπάνω μεθόδου, καλείται η μέθοδος:

```
private String downloadUrl(String strUrl) throws
IOException{
```

η οποία συνδέεται με το API Google μέσω HttpURLConnection.

```
urlConnection = (HttpURLConnection) url.openConnection();

// Connecting to url
urlConnection.connect();

// Reading data from url
iStream = urlConnection.getInputStream();

BufferedReader br = new BufferedReader(new
InputStreamReader(iStream));

StringBuffer sb = new StringBuffer();
```

```
String line = "";
while( ( line = br.readLine()) != null){
    sb.append(line);
}

data = sb.toString();

br.close();
```

Τώρα είμαστε έτοιμοι να δεχτούμε τα δεδομένα της Google μετά την επιτυχή σύνδεση μας με το API της. Την περισυλλογή των δεδομένων την αναλαμβάνει η μέθοδος DownloadTask που παρουσιάζεται παρακάτω.

```
// Fetches data from url passed
private class DownloadTask extends AsyncTask<String,
Void, String> {

    // Downloading data in non-ui thread
    @Override
    protected String doInBackground(String... url) {

        // For storing data from web service
        String data = "";

        try{
            // Fetching the data from web service
            data = downloadUrl(url[0]);
        }catch(Exception e){
            Log.d("Background Task",e.toString());
        }
        return data;
    }

    // Executes in UI thread, after the execution of
    // doInBackground()
    @Override
    protected void onPostExecute(String result) {
        super.onPostExecute(result);

        ParserTask parserTask = new ParserTask();

        // Invokes the thread for parsing the JSON data
        parserTask.execute(result);
    }
}
```

```
// Fetches data from url passed
private class DownloadTask extends AsyncTask<String,
```

```
Void, String> {

    // Downloading data in non-ui thread
    @Override
    protected String doInBackground(String... url) {

        // For storing data from web service
        String data = "";

        try{
            // Fetching the data from web service
            data = downloadUrl(url[0]);
        }catch(Exception e){
            Log.d("Background Task",e.toString());
        }
        return data;
    }

    // Executes in UI thread, after the execution of
    // doInBackground()
    @Override
    protected void onPostExecute(String result) {
        super.onPostExecute(result);

        ParserTask parserTask = new ParserTask();

        // Invokes the thread for parsing the JSON data
        parserTask.execute(result);
    }
}
```

Τα δεδομένα που μας ήρθαν είναι σε JSON FORMAT και μπορούμε πολύ εύκολα να τα αποθηκεύσουμε σε μία κλάση (εφόσων τα αποκωδικοποιήσουμε πρώτα). Γι'αυτή την διαδικασία μας βοηθάει η κλάση DirectionsJSONParser η οποία αφού τα «διαβάσει» μπορεί μετά να τα αποθηκεύσει στην κλάση MapsACtivity, που είναι και η κύρια κλάση της εφαρμογής μας.

4.2.2 Directions JSON Parser

Το **JSON** (JavaScript Object Notation) είναι ένα απλό πρότυπο ανταλλαγής δεδομένων. Είναι εύκολο για τους ανθρώπους να το αναγνώσουν και εύκολο για τις μηχανές να το αναλύσουν και να το παράγουν αφού βασίζεται πάνω σε ένα υποσύνολο της γλώσσας προγραμματισμού JavaScript (Standard ECMA-262 Έκδοση 3η - Δεκέμβριος 1999). Το JSON είναι ένα πρότυπο κειμένου το οποίο είναι πολύ ανεξάρτητο όσον αφορά τις υπόλοιπες γλώσσες προγραμματισμού, αλλά χρησιμοποιεί "πρακτικές" τις οποίες γνωρίζουν οι προγραμματιστές της οικογένειας προγραμματισμού C, συμπεριλαμβανομένων των "C, C++, Python, Java, JavaScript" και πολλών άλλων. Αυτές οι ιδιότητες κάνουν το JSON μια εξερετική γλώσσα προγραμματισμού που βασίζεται στην ανταλλαγή δεδομένων.

Η παρακάτω κλάση παίρνει ως δεδομένα ένα αντικείμενο τύπου JSON και αφού το διαβάσει κατάλληλα, μπορούμε να αποσπάσουμε από αυτό μια λίστα με συντεταγμένες στο χάρτη, που υποδηλώνουν τις διαδρομές τις οποίες θα ακολουθήσουμε.

```
public class DirectionsJSONParser {

    /** Receives a JSONObject and returns a list of lists
containing latitude and longitude */
    public List<List<HashMap<String,String>>>
parse(JSONObject jObject){

        List<List<HashMap<String, String>>> routes = new
ArrayList<List<HashMap<String,String>>>() ;
        JSONArray jRoutes = null;
        JSONArray jLegs = null;
        JSONArray jSteps = null;

        try {

            jRoutes = jObject.getJSONArray("routes");

            /** Traversing all routes */
            for(int i=0;i<jRoutes.length();i++){
                jLegs = (
(JSONObject)jRoutes.get(i)).getJSONArray("legs");
                List path = new ArrayList<HashMap<String,
String>>();

                /** Traversing all legs */
                for(int j=0;j<jLegs.length();j++){
                    jSteps = (
(JSONObject)jLegs.get(j)).getJSONArray("steps");

                    /** Traversing all steps */
                    for(int k=0;k<jSteps.length();k++){
                        String polyline = "";
                        polyline =
(String)((JSONObject)((JSONObject)jSteps.get(k)).get("pol
yline")).get("points");
                        List<LatLng> list =
decodePoly(polyline);

                        /** Traversing all points */
                        for(int l=0;l<list.size();l++){
                            HashMap<String, String> hm =
new HashMap<String, String>();
                            hm.put("lat",
Double.toString(((LatLng)list.get(l)).latitude) );
                            hm.put("lng",
Double.toString(((LatLng)list.get(l)).longitude) );
                            path.add(hm);
```

```
                        }
                    }
                    routes.add(path);
                }
            }

        } catch (JSONException e) {
            e.printStackTrace();
        }catch (Exception e){
        }

        return routes;
    }
    /**
     * Method to decode polyline points
     * Courtesy :
http://jeffreysambells.com/2010/05/27/decoding-polylines-
from-google-maps-direction-api-with-java
     * */
    private List<LatLng> decodePoly(String encoded) {

        List<LatLng> poly = new ArrayList<LatLng>();
        int index = 0, len = encoded.length();
        int lat = 0, lng = 0;

        while (index < len) {
            int b, shift = 0, result = 0;
            do {
                b = encoded.charAt(index++) - 63;
                result |= (b & 0x1f) << shift;
                shift += 5;
            } while (b >= 0x20);
            int dlat = ((result & 1) != 0 ? ~(result >>
1) : (result >> 1));
            lat += dlat;

            shift = 0;
            result = 0;
            do {
                b = encoded.charAt(index++) - 63;
                result |= (b & 0x1f) << shift;
                shift += 5;
            } while (b >= 0x20);
            int dlng = ((result & 1) != 0 ? ~(result >>
1) : (result >> 1));
            lng += dlng;

            LatLng p = new LatLng(((double) lat / 1E5)),
                    (((double) lng / 1E5)));
            poly.add(p);
        }
```

```
        return poly;
    }
```

Πηγή : http://jeffreysambells.com/2010/05/27/decoding-polylines-from-google-maps-direction-api-with-java

Είναι η πιό διαδεδομένη μορφή αποκωδικοποίησης JSON όσων αφορά τα JSON που μας έχει στήλει το API της Google. Η παραπάνω κλάση δέχεται ως όρισμα ενα αντικείμενο τύπου JSON και μας επιστρέφει μια λίστα λιστών απο συντεταγμένες.
Έπειτα, επιλέγουμε την συντομότερη διαδρομή αφού αποθηκεύσουμε τα παραπάνω σημείων στην κλάση MapsActivity με την βοήθεια της :

```
/** A class to parse the Google Places in JSON format */
private class ParserTask extends AsyncTask<String,
Integer, List<List<HashMap<String,String>>> >{

    // Parsing the data in non-ui thread
    @Override
    protected List<List<HashMap<String, String>>>
doInBackground(String... jsonData) {

        JSONObject jObject;
        List<List<HashMap<String, String>>> routes =
null;

        try{
            jObject = new JSONObject(jsonData[0]);
            DirectionsJSONParser parser = new
DirectionsJSONParser();

            // Starts parsing data
            routes = parser.parse(jObject);
        }catch(Exception e){
            e.printStackTrace();
        }
        return routes;
    }

    // Executes in UI thread, after the parsing process
    @Override
    protected void
onPostExecute(List<List<HashMap<String, String>>> result)
{
        ArrayList<LatLng> points = null;
        PolylineOptions lineOptions = null;
        MarkerOptions markerOptions = new
MarkerOptions();
```

```
        // Traversing through all the routes
        for(int i=0;i<result.size();i++){
            points = new ArrayList<LatLng>();
            lineOptions = new PolylineOptions();

            // Fetching i-th route
            List<HashMap<String, String>> path =
result.get(i);

            // Fetching all the points in i-th route
            for(int j=0;j<path.size();j++){
                HashMap<String,String> point =
path.get(j);

                double lat =
Double.parseDouble(point.get("lat"));
                double lng =
Double.parseDouble(point.get("lng"));
                LatLng position = new LatLng(lat, lng);

                points.add(position);
            }

            // Adding all the points in the route to
LineOptions
            lineOptions.addAll(points);
            lineOptions.width(2);


            if (!walking) {
                lineOptions.color(Color.RED);
            }
            else{
                lineOptions.color(Color.GREEN);

            }
        }

        // Drawing polyline in the Google Map for the i-
th route
        mMap.addPolyline(lineOptions);
    }
}
```

Η οποία βρίσκει την συντομότερη διαδρομή και την σχεδιάζει στον χάρτη μας με την βοήθεια της μεταβλητής LineOptions στην οποία αποθηκεύεται η παραπάνω διαδρομή.

4.2.3 Splash Screen

```
public class SplashScreen extends MapsActivity {
```

Η συγκεκριμένη κλάση δημιουργεί μια εισαγωγική εικόνα, πριν την έναρξη και εμφάνιση της κύριας οθόνης.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.startscreen);

    final ImageView iv = (ImageView) findViewById(R.id.imageView);
```

Σε αυτό το σημείο φορτώνουμε μια εικόνα ως εισαγωγική με ενα συγκεκριμένο LOGO της αρεσκείας μας. (startscreen).

```
Button button = (Button)findViewById(R.id.button);

button.setOnClickListener(
        new Button.OnClickListener() {
            public void onClick(View v) {
                MapsActivity.walking = false;
                Intent i = new Intent(getBaseContext(),
MapsActivity.class);
                startActivity(i);

            }
        }
);

Button button2 = (Button)findViewById(R.id.button2);

button2.setOnClickListener(
        new Button.OnClickListener() {
            public void onClick(View v) {
                MapsActivity.walking = true;
                Intent i = new Intent(getBaseContext(),
MapsActivity.class);
                startActivity(i);

            }
        }
);
```

Χρησιμοποιούμε "button listeners" οι οποίοι μας βοηθούν στο να κατανοήσουμε ποιά επιλογή έκανε ο χρήστης (M
η επιλογή, η αμέσως επόμενη ενέργεια που θα εκτελεστεί είναι η εμφάνιση της κύριας οθόνης
ουτως ώστε ο χρηστης να επιλέξει την διαδρομή που επιθυμεί με το μέσο μετακίνησης που
επέλεξε προηγουμένως.

4.3 Αρχείο Android Manifest

Το AndroidManifest.xml είναι το ένα και μοναδικό αρχείο που χρειάζεται κάθε Android εφαρμογή για να ξεκινήσει. Από την κατάληξη καταλαβαίνουμε ότι κάθε εντολή που περιέχει είναι γραμμένη σε XML μορφή και είναι το αρχείο που αρχικά διαβάζει το λειτουργικό Android για να ξέρει πώς να εκτελέσει την εφαρμογή μας με όλα τα στοιχεία που την αποτελούν. Επιπλέον χρησιμοποιείται για να δηλώσουμε στο πρόγραμμα, ποιες άδειες να ζητήσει από τον χρήστη κατά την εγκατάσταση της εφαρμογής π.χ. «αδεια για χρήση του εξωτερικού χώρου αποθηκευσης».

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.xartes.smartmap" >

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission
android:name="com.google.android.providers.gsf.permission.READ_GSERVI
CES" />
    <!--
 The ACCESS_COARSE/FINE_LOCATION permissions are not required to use
         Google Maps Android API v2, but are recommended.
    -->
    <uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <meta-data
            android:name="com.google.android.gms.version"
            android:value="@integer/google_play_services_version" />
        <meta-data
            android:name="com.google.android.maps.v2.API_KEY"
            android:value="@string/google_maps_key" />

        <activity
            android:name=".MapsActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category
android:name="android.intent.category.default" />
            </intent-filter>
        </activity>

        <activity android:name=".SplashScreen">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category
android:name="android.intent.category.LAUNCHER"/>

            </intent-filter>
```

```
        </activity>
    </application>
</manifest>
```

Το συγκεκριμένο αρχείο μας βοηθάει να «πούμε» στην εφαρμογή ποιο Activity file (ποια κλαση) θέλουμε να ξεκινάει πρώτη, και ποιά κλάση θέλουμε να παίρνει την σειρά της πρώτης, εφόσων η πρωτη ολοκλήρωσε σωστά την διεργασία της.

4.4 Εικονική Παρουσίαση Λειτουργιών Εφαρμογής

Έχοντας αναφερθεί ήδη στο παρόν κεφάλαιο στα κομμάτια του κώδικα που

Εικόνα 36: User Interface Αρχικής Εικόνας

Στην αρχική εικόνα (StartScreen) εμφανίζεται η διεπαφή όπου ο χρήστης επιλέγει τον τρόπο με τον οποίο επιθυμεί να μετακινηθεί μέσα στην πόλη των Χανίων. Εμφανίζονται λοιπόν δύο διαθέσιμες επιλογές :

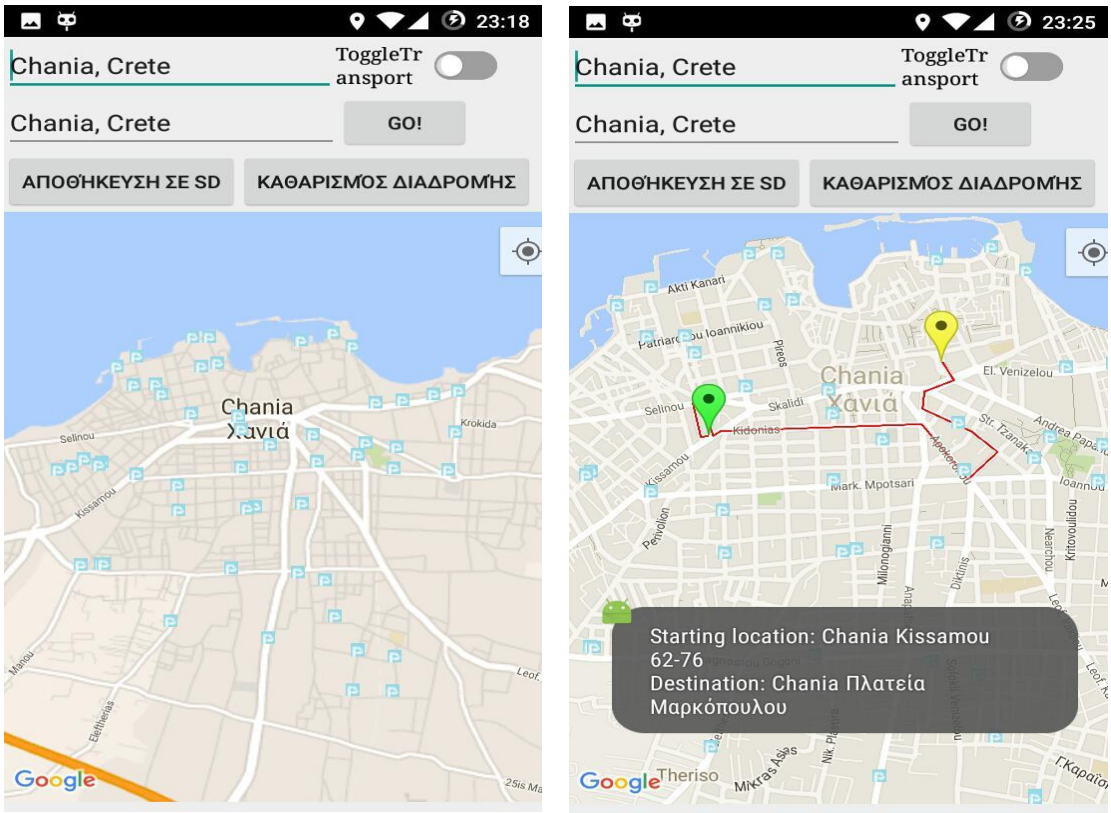
1. Μετακίνηση με Αυτοκίνητο.
2. Μετακίνηση με Αναπηρικό Αμαξίδιο.

Στην περίπτωση που ο χρήστης επιλέξει να μετακινηθεί με αυτοκίνητο, εμφανίζεται η επόμενη διεπαφή της εφαρμογής που περιέχει τον χάρτη με τις διαθέσιμες θέσεις parking που υπάρχουν στην πόλη και επιπρόσθετες λειτουργίες που εξυπηρετούν τον χρήστη, όπως βλέπουμε στην παρακάτω εικόνα :

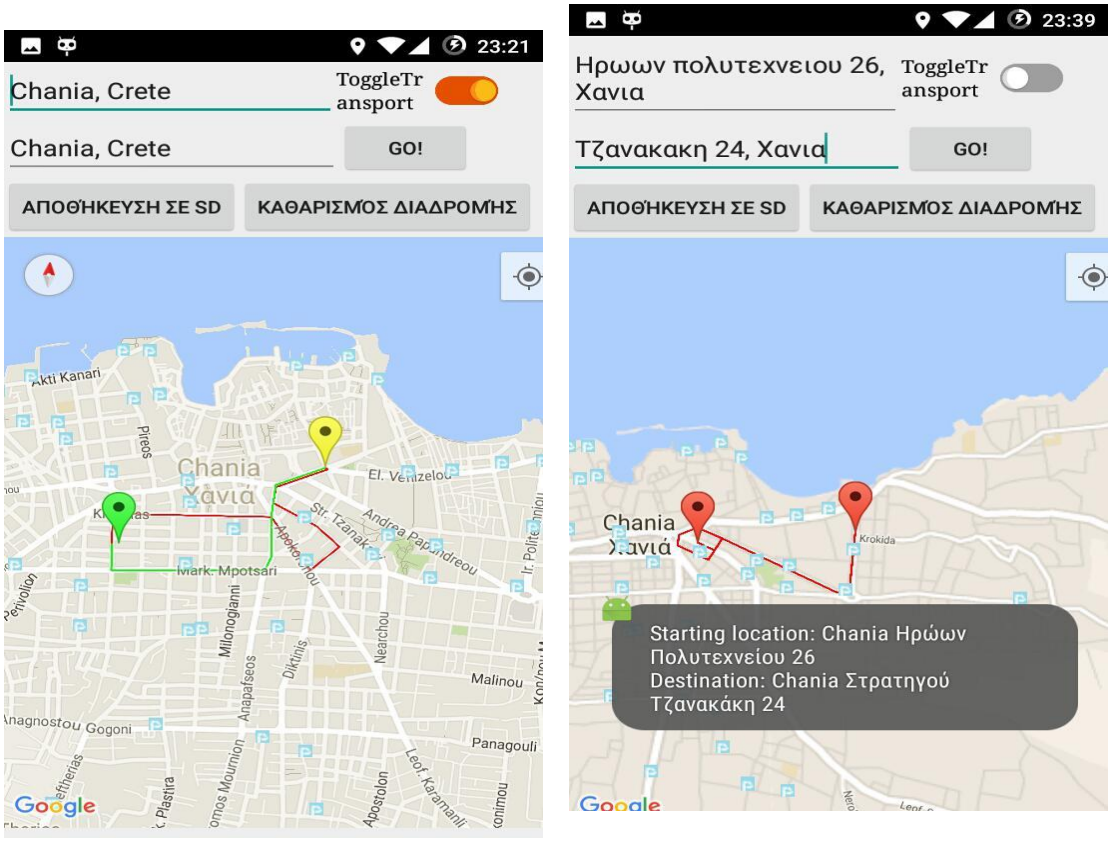
Εδώ ο χρήστης έχει τη δυνατότητα να επιλέξει την αρχή και το τέλος της διαδρομής του είτε με το να πληκτρολογήσει τις διευθύνσεις στα πεδία αναζήτησης (και στη συνέχεια πατώντας το κουμπί "GO!"), είτε απλά με το να "πατήσει" τα δύο σημεία

που επιθυμεί πάνω στο χάρτη. Επιπλέον μπορεί να κάνει zoom in/zoom out, να διαγράψει ή να αποθηκεύσει σε μορφή εικόνας(.jpg) τη διαδρομή που επέλεξε και τέλος, επιλέγοντας το κουμπί Toggle Transport εμφανίζεται ταυτόχρονα η διαδρομή που εξυπηρετεί το δεύτερο χάρτη της εφαρμογής μας. Οι παραπάνω ενέργειες απεικονίζονται στις εικόνες που ακολουθούν :

Εικόνα 37: User Interface Για Χάρτη Με Αυτοκίνητο

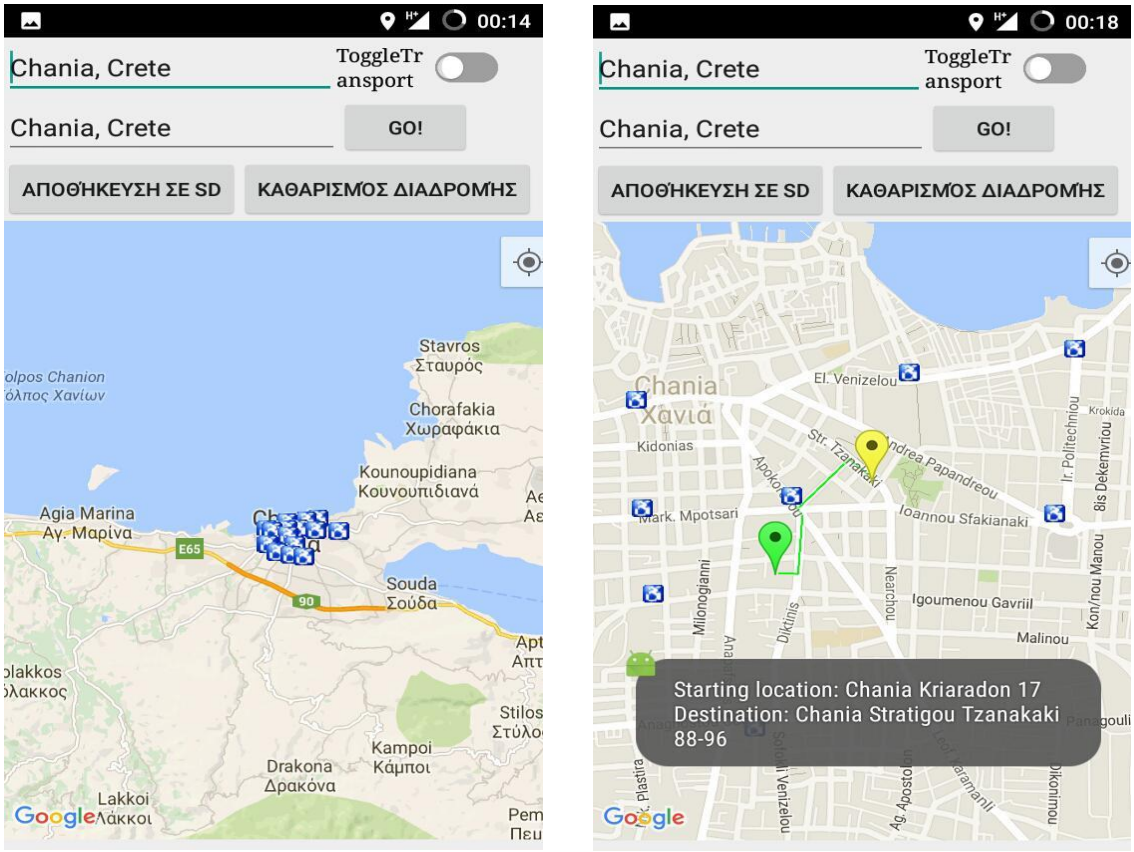


Εικόνα 38–39: Zoon In Στο Χάρτη Με Αυτοκίνητο – Προβολή Διαδρομής Μέσω TouchScreen Στο Χάρτη Με Αυτοκίνητο

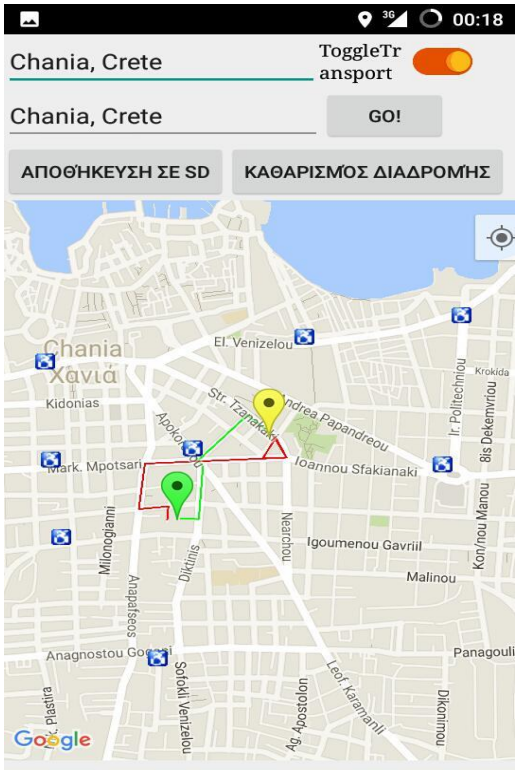


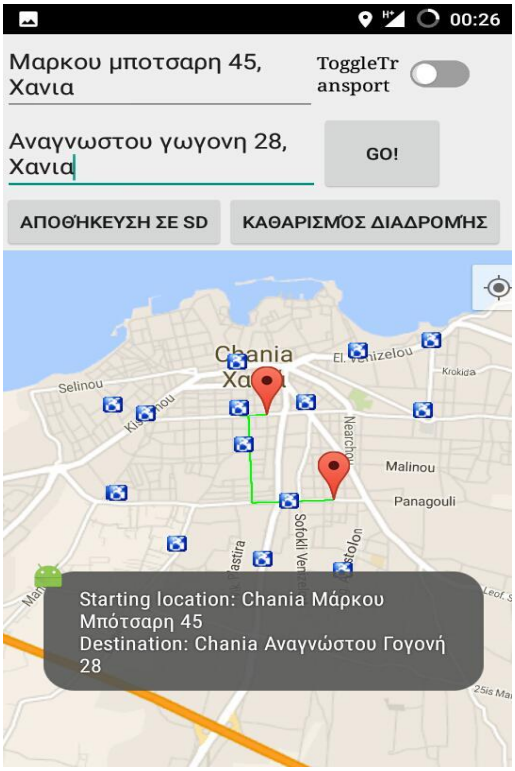
Εικόνα 40–41: Toggle Transport Στο Χάρτη Με Αυτοκίνητο – Προβολή Διαδρομής Μέσω Κουμπιού Αναζήτησης Στο Χάρτη Με Αυτοκίνητο

Στην περίπτωση που ο χρήστης επιλέξει να μετακινηθεί με το αναπηρικό αμαξίδιο, εμφανίζεται ένα νέο interface της εφαρμογής που περιέχει τον χάρτη με τις διαθέσιμες ράμπες και κόμβους (σημεία) εύκολης πρόσβασης για ΑΜΕΑ μέσα στο κέντρο της πόλης. Επίσης, ο χρήστης έχει τη δυνατότητα να χρησιμοποιήσει τις ίδιες λειτουργίες που προαναφέραμε και στο συγκεκριμένο κομμάτι της εφαρμογής όπως βλέπουμε στις παρακάτω εικόνες :

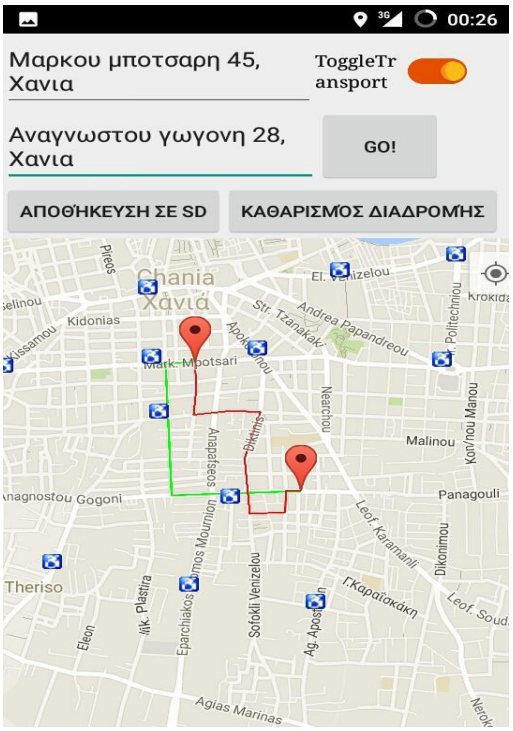


Εικόνα 42–43: User Interface Για Χάρτη Με Αναπηρικό Αμαξίδιο – Προβολή Διαδρομής Μέσω TouchScreen Στο Χάρτη Με Αναπηρικό Αμαξίδιο





Εικόνα 44–45: Toggle Transport Στο Χάρτη Με Αναπηρικό Αμαξίδιο – Προβολή Διαδρομής Μέσω Κουμπιού Αναζήτησης Στο Χάρτη Με Αναπηρικό Αμαξίδιο



Εικόνα 46: Toggle Transport + Προβολή Διαδρομής Μέσω Κουμπιού Αναζήτησης

5 Βιβλιογραφία

- Το επίσημο site του Android : *http://developer.android.com/index.html*
- Πηγές εύρεσης πληροφοριών για την πλατφόρμα Android : *http://www.allaboutandroid.gr/* , *https://el.wikipedia.org* , *http://www.adds.gr/* , *http://www.webopedia.com/* , *http://androidruller.blogspot.gr/*
- Πληροφορίες σχετικά με τον αλγόριθμο Dijkstra : *http://vivliothmmy.ee.auth.gr/* , *https://el.wikipedia.org*
- Οδηγίες για το πρότυπο JSON : *http://jsonviewer.stack.hu/*
- Ιστοσελίδα αντιμετώπισης προβλημάτων : *http://stackoverflow.com/*
- Download Android Studio : *http://developer.android.com/sdk/index.html*
- Ιστοσελίδες με οδηγίες για συγγραφή κώδικα και ανάπτηξη εφαρμογών Android : *http://www.androidhive.info/* , *http://www.vogella.com/tutorials/android.html* , *http://wptrafficanalyzer.in/blog/category/android/*
- Πηγή εύρεσης εικόνων : *http://theinspirationroom.com/*