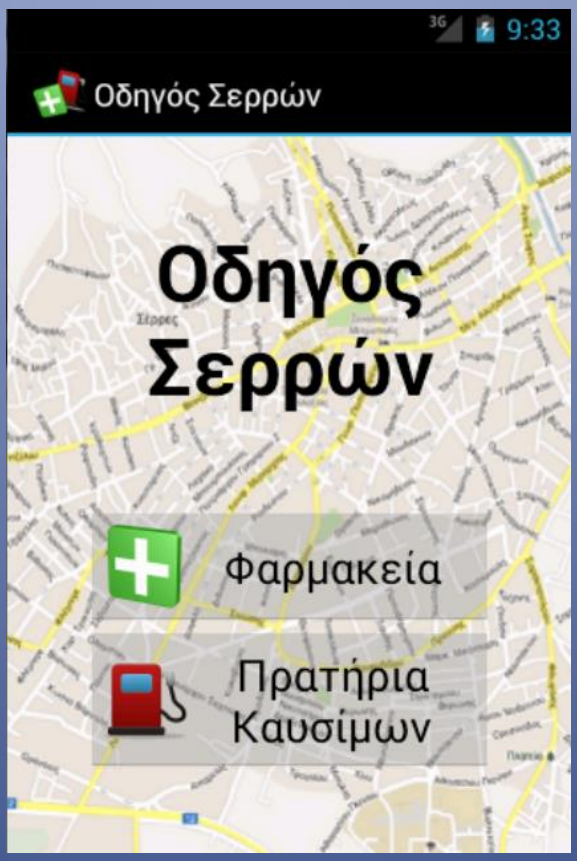
ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΣΕΡΡΩΝ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΕΠΙΚΟΙΝΩΝΙΩΝ

ΣΕΡΡΕΣ
2013



ΣΧΕΔΙΑΣΗ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΕΜΠΟΡΙΚΟΥ PORTAL ΓΙΑ ΤΗΝ ΠΟΛΗ ΤΩΝ ΣΕΡΡΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΑΝΑΣΤΑΣΙΑ ΚΟΥΣΤΟΥΔΑ

ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΣΕΡΡΩΝ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΕΠΙΚΟΙΝΩΝΙΩΝ

ΣΧΕΔΙΑΣΗ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΕΜΠΟΡΙΚΟΥ PORTAL ΓΙΑ ΤΗΝ ΠΟΛΗ ΤΩΝ ΣΕΡΡΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Αναστασία Κουστούδα, Α.Μ. 2339

ΣΕΡΡΕΣ, 2013

<u>Υπεύθυνη Δήλωση</u>: *Βεβαιώνω ότι είμαι συγγραφέας αυτής της πτυχιακής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται στην πτυχιακή εργασία. Επίσης έχω αναφέρει τις πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επίσης βεβαιώνω ότι αυτή η πτυχιακή εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τις απαιτήσεις του προγράμματος σπουδών του τμήματος Πληροφορικής & Επικοινωνιών του Τ.Ε.Ι. Σερρών.*

ΣΧΕΔΙΑΣΗ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΕΜΠΟΡΙΚΟΥ PORTAL ΓΙΑ ΤΗΝ ΠΟΛΗ ΤΩΝ ΣΕΡΡΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Αναστασία Κουστούδα, Α.Μ. 2339

Επιβλέπων: Δρ. Γεώργιος Παυλίδης
Επιστημονικός Συνεργάτης ΤΕΙ Σερρών

ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΣΕΡΡΩΝ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΕΠΙΚΟΙΝΩΝΙΩΝ

Αφιερώνεται στους γονείς μου, Αριστείδη και Άννα…

Περίληψη

Στην παρούσα πτυχιακή εργασία αναλύθηκαν και περιγράφονται τεχνικές ανάπτυξης εφαρμογών για έξυπνες κινητές συσκευές με λειτουργικό σύστημα Android. Οι τεχνικές αυτές μελετήθηκαν στην πράξη με την ανάπτυξη εφαρμογής στην πλατφόρμα Android SDK. Η εφαρμογή που παρουσιάζεται έχει ως σκοπό τη συγκέντρωση και παροχή πληροφοριών για τα φαρμακεία και τα πρατήρια καυσίμων στο Νομό Σερρών, μια δυνατότητα που δεν υπήρχε διαθέσιμη για έξυπνες κινητές συσκευές, τουλάχιστον μέχρι το διάστημα συγγραφής της παρούσας εργασίας. Η ανάκτηση των σχετικών πληροφοριών γίνεται από επίσημες ιστοσελίδες στο διαδίκτυο και η παρουσίασή τους γίνεται πάνω σε διαδραστικό χάρτη, όπου παρέχονται δυνατότητες γεωκωδικοποίησης, αντίστροφης γεωκωδικοποίησης και εύρεσης διαδρομής.

Abstract

In this thesis techniques for developing applications for smart mobile devices running on Android OS were analyzed and described. These techniques have been practically studied with the development of an application using the platform Android SDK. The application aims to collect information on the pharmacies and gas stations within the Municipality of Serres, Greece, from relevant pages on authoritative sites on the internet and present them on an interactive map with geocoding, inverse geocoding and navigation capabilities.

Πρόλογος

Στόχος της παρούσας εργασίας ήταν η ανάπτυξη εφαρμογής για έξυπνες κινητές συσκευές με λειτουργικό σύστημα Android, η οποία θα μπορούσε να παρέχει έγκυρη και έγκαιρη πληροφόρηση σχετικά με τα φαρμακεία και τα πρατήρια καυσίμων στο Νομό Σερρών. Για την υλοποίηση της εφαρμογής μελετήθηκε το λειτουργικό σύστημα Android OS, οι βιβλιοθήκες ανάπτυξης εφαρμογών Android SDK, το περιβάλλον ανάπτυξης Eclipse IDE και επιπρόσθετες βιβλιοθήκες ανάπτυξης εφαρμογών με βάση χάρτες τύπου Google Maps. Κατά την εκπόνηση της εργασίας μελετήθηκαν και εφαρμόστηκαν στην πράξη θέματα όπως η ανάπτυξη εφαρμογών για σύγχρονες έξυπνες κινητές συσκευές διαφόρων μεγεθών και δυνατοτήτων, η ανάπτυξη βάσεων δεδομένων σε κινητές συσκευές, η ανάκτηση πληροφοριών από έγκυρες ιστοσελίδες και η ανάσυρση χρήσιμης πληροφορίας μέσα από αυτές, η χρήση διαδραστικών χαρτών, γεωκωδικοποίησης, αντίστροφης γεωκωδικοποίησης και εύρεσης διαδρομών και τέλος η ενσωμάτωση δυνατοτήτων αλληλεπίδρασης συσκευής χρήστη και προσθήκης ρυθμίσεων για την εναρμόνιση με τις ανάγκες του χρήστη. Ουσιαστικά, η εργασία περιελάμβανε ένα πλήρη κύκλο ανάπτυξης εφαρμογής για έξυπνες κινητές συσκευές με Android OS επαγγελματικού επιπέδου καλύπτοντας μάλιστα μια τρέχουσα ανάγκη της αγοράς, καθώς καμία γνωστή εφαρμογή σε οποιοδήποτε λειτουργικό σύστημα έξυπνης κινητής συσκευής δεν περιλαμβάνει τη λειτουργικότητα και την πληροφόρηση που παρέχει η εφαρμογή που αναπτύχθηκε.

Θα ήθελα να ευχαριστήσω ιδιαίτερα τους καθηγητές κ. Γεώργιο Παυλίδη και κ. Αλέξανδρο Βακαλούδη για την επίβλεψη, τη βοήθεια, αλλά και για την εμπιστοσύνη που μου έδειξαν σε ότι έκανα.

Τέλος, θέλω να ευχαριστήσω φυσικά τους γονείς μου για την υποστήριξη που μου έδειξαν και τη βοήθεια που μου πρόσφεραν όλα αυτά τα χρόνια, τα αδέρφια μου που μου στάθηκαν, τους φίλους μου που έδειξαν κατανόηση και όλους τους ανθρώπους που γνώρισα κατά τη διάρκεια τον σπουδών μου και μου πρόσφεραν εμπειρίες και γνώσεις.

Αναστασία Κουστούδα
Σέρρες, 2013

ΕΙΚΟΝΕΣ

ΠΙΝΑΚΕΣ

ΠΕΡΙΕΧΟΜΕΝΑ

Εισαγωγή

Η χρήση έξυπνων κινητών τηλεφώνων (smartphones) είναι πλέον ευρύτατα διαδεδομένη καθώς καλύπτει καθημερινές ανάγκες των χρηστών παγκόσμια, ανάγκες όπως η σύνδεση στο διαδίκτυο ανά πάσα στιγμή και στις πληροφορίες που αυτό προσφέρει. Τα smartphones δίνουν στους χρήστες τη δυνατότητα συνεχούς ενημέρωσης για θέματα του ενδιαφέροντός τους. Πλήθος εφαρμογών, χιλιάδες πλέον, για διάφορες σύγχρονες πλατφόρμες (Android, iOS, Windows Phone), διατίθενται στους χρήστες για τη διευκόλυνσή τους. Κάποιες από αυτές, χρησιμοποιούν το διαδίκτυο και συνεισφέρουν στην ενημέρωση του χρήστη. Συγκεκριμένα, οι «εφαρμογές πύλης» αποσκοπούν στη συγκέντρωση χρήσιμων πληροφοριών από διαδικτυακές πηγές και την παρουσίασή τους με έναν εύκολο και ξεκούραστο τρόπο στο χρήστη. Πληροφορίες όπως οι τιμές καυσίμων, ή τα εφημερεύοντα φαρμακεία είναι πάντα χρήσιμες και επίκαιρες. Η παρουσίαση τέτοιων πληροφοριών μπορεί γίνει με χρήση χαρτών, ώστε ο χρήστης ταυτόχρονα να βλέπει τη θέση τους και να επωφελείται από τις πρόσθετες λειτουργίες των διαδραστικών χαρτών όπως η λήψη οδηγιών προς μια συγκεκριμένη τοποθεσία. Οι σύγχρονες αυτές προσεγγίσεις και τάσεις μαζί με μια μεγάλη ζήτηση που φαίνεται να έχουν ανάλογες εφαρμογές, υπήρξαν το κίνητρο για την εκπόνηση της παρούσας εργασίας και την υλοποίηση εφαρμογής «ζωντανής» πληροφόρησης για την πόλη των Σερρών, στοχεύοντας μάλιστα σε μια περιοχή που δεν είχε γνωρίσει αντίστοιχη εφαρμογή κατά το διάστημα υλοποίησης.

1. Σκοπός της Εργασίας

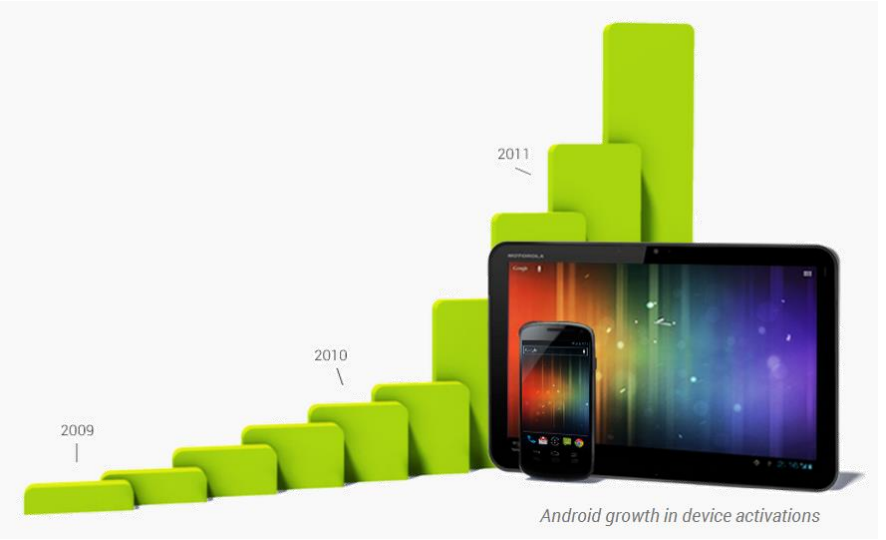
Η παρούσα πτυχιακή εργασία έχει ως θέμα τη σχεδίαση και υλοποίηση εμπορικού portal για την πόλη των Σερρών για έξυπνες κινητές συσκευές με λειτουργικό σύστημα Android. Ειδικότερα, η εφαρμογή πρέπει να συγκεντρώνει στοιχεία για τα φαρμακεία (εφημερίες και στοιχεία διεύθυνσης) και τα πρατήρια καυσίμων (τιμές και στοιχεία διεύθυνσης) της πόλης των Σερρών δια μέσου επίσημων ιστοσελίδων με σχετικές πληροφορίες. Τα στοιχεία αυτά, καθώς και η τοποθεσία του χρήστη πρέπει να επιδεικνύονται σε διαδραστικό χάρτη της πόλης των Σερρών. Η εφαρμογή θα πρέπει να ανιχνεύει αλλαγές σχετικές με την τοποθεσία του χρήστη, την

οποία πρέπει να απεικονίζει σε πραγματικό χρόνο. Πέρα από τις βασικές αυτές λειτουργίες που αποτέλεσαν τον αρχικό στόχο της εφαρμογής, προστέθηκαν στη συνέχεια οι εξής λειτουργίες, για περαιτέρω βελτίωση της λειτουργικότητας:

- Δυνατότητα λήψης οδηγιών, ανάλογα με το μέσο μεταφοράς (πεζός, με αυτοκίνητο, με ποδήλατο).
- Δυνατότητα προβολής των επιλεγμένων καταστημάτων (φαρμακεία ή πρατήρια καυσίμων) και σε λίστα, με ταξινόμηση αλφαβητικά ή ανάλογα με την απόσταση του καταστήματος ή ανάλογα με την τιμή καυσίμου, αν έχουν επιλεχθεί πρατήρια καυσίμων.
- Άμεση τηλεφωνική κλήση του φαρμακείου που επιθυμεί ο χρήστης, από τα στοιχεία του καταστήματος που εμφανίζει η εφαρμογή.

2. Αναγκαιότητα της Εργασίας

Το λειτουργικό σύστημα Android χρησιμοποιείται σε εκατοντάδες εκατομμύρια κινητές συσκευές σε περισσότερες από 190 χώρες σε όλο τον κόσμο. Είναι η πλατφόρμα με το μεγαλύτερο αριθμό εγκαταστάσεων από οποιαδήποτε άλλη, και ο αριθμός αυτός αυξάνεται συνεχώς, κυρίως λόγω του ότι είναι λογισμικό ανοικτού κώδικα. Εκτιμάται σήμερα ότι κάθε μέρα 1 εκατομμύριο χρήστες ενεργοποιούν τις Android συσκευές τους για πρώτη φορά και αρχίζουν να αναζητούν εφαρμογές, παιχνίδια και άλλο ψηφιακό περιεχόμενο (Εικόνα 1) [1].



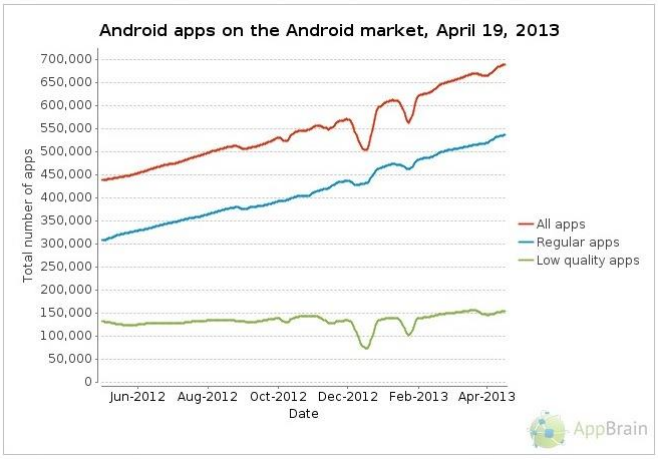
Εικόνα 1. Η αύξηση του αριθμού ενεργοποιήσεων android συσκευών

Το Android δίνει μια ενοποιημένη πλατφόρμα για τη δημιουργία εφαρμογών για τους χρήστες ανά τον κόσμο, καθώς και μια ανοιχτή αγορά για

την άμεση διανομή τους. Με βάση τις συνεισφορές της κοινότητας ανοικτού λογισμικού του Linux και περισσότερους από 300 συνεργάτες, το Android έχει γίνει γρήγορα το ταχύτερα αναπτυσσόμενο λειτουργικό σύστημα κινητής τηλεφωνίας. Η διαφάνεια του Android το έχει καταστήσει ως αγαπημένο για τους καταναλωτές αλλά και τους προγραμματιστές, οδηγώντας σε μεγάλη αύξηση της κατανάλωσης εφαρμογών. Οι χρήστες του Android κατεβάζουν περισσότερες από 1,5 δισεκατομμύρια εφαρμογές και παιχνίδια από το Google Play κάθε μήνα. Με τους συνεργάτες του, το Android πιέζει συνεχώς τα όρια του υλικού και του λογισμικού ώστε να φέρει νέες δυνατότητες για τους χρήστες και τους προγραμματιστές. Για τους προγραμματιστές, η καινοτομία συνίσταται στη δημιουργία ισχυρών, διαφοροποιημένων εφαρμογών που χρησιμοποιούν τις τελευταίες τεχνολογίες έξυπνων κινητών τηλεφώνων. Το Android παρέχει όλα τα απαραίτητα εργαλεία για την ανάπτυξη κορυφαίων εφαρμογών. Δίνει ένα ενιαίο μοντέλο εφαρμογών που επιτρέπει την εξάπλωση των εφαρμογών σε εκατοντάδες εκατομμύρια χρήστες, για ένα ευρύ φάσμα των συσκευών, όπως κινητά τηλέφωνα και tablets. Το Android δίνει επίσης εργαλεία για τη δημιουργία εφαρμογών, με ωραία εμφάνιση, που επωφελούνται από τις δυνατότητες του υλικού που διατίθενται σε κάθε συσκευή. Προσαρμόζει αυτόματα το User Interface (UI), ώστε η εφαρμογή να έχει την καλύτερη δυνατή εμφάνιση σε κάθε συσκευή, ενώ δίνει στον προγραμματιστή όλο τον έλεγχο που χρειάζεται πάνω στο UI της εφαρμογής σε διαφορετικούς τύπους συσκευών. Το Google Play είναι το κύριο ηλεκτρονικό κατάστημα για την πώληση και διανομή των εφαρμογών Android. Καθένας έχει το δικαίωμα να προωθήσει τις εφαρμογές του στο Google Play, το οποίο δίνει τον έλεγχο για το πώς μπορούν να πωληθούν (π.χ. με κόστος ή χωρίς) οι εφαρμογές. Οι εφαρμογές μπορούν να διανεμηθούν ευρέως σε όλα τα ηλεκτρονικά καταστήματα εφαρμογών Android, για όλες συσκευές ή σε συγκεκριμένα μέρη, για συγκεκριμένες συσκευές, ανάλογα με τις δυνατότητες του υλικού τους [1].

Το γράφημα στην Εικόνα 2 δείχνει τον αριθμό των διαθέσιμων εφαρμογών στο Android Market-Google Play (τελευταία ενημέρωση: 25 Απριλίου, 2013) [2].

Από τα προαναφερθέντα στοιχεία προκύπτει ότι το ενδιαφέρον των χρηστών, αλλά και των προγραμματιστών, για νέες εφαρμογές ολοένα και αυξάνεται. Το περιβάλλον μέσα στο οποίο πραγματοποιείται το έργο είναι πολύ πρόσφορο, ειδικά για μία εφαρμογή σαν αυτή που πραγματεύεται η παρούσα εργασία.



Εικόνα 2. Η αύξηση του αριθμού των διαθέσιμων εφαρμογών στο Google Play

3. Όφελος που προκύπτει από αυτήν την εργασία

Η εφαρμογή που υλοποιήθηκε στο πλαίσιο της παρούσας εργασίας παρέχει στο χρήστη εύκολη και άμεση πρόσβαση σε καίριες πληροφορίες σχετικά με τα πρατήρια καυσίμων και τα φαρμακεία των Σερρών. Συνεχώς ενημερώνει τις πληροφορίες αυτές και τις παρουσιάζει σε ένα όμορφο και εύκολο στη χρήση περιβάλλον.

Επίσης, μετά από έρευνα που πραγματοποιήθηκε σε αντίστοιχες εφαρμογές, παρατηρήθηκε ότι για τα φαρμακεία της πόλης των Σερρών δεν υπάρχουν πληροφορίες σε καμία εφαρμογή για έξυπνες κινητές συσκευές ή τουλάχιστον δεν υπάρχουν πληροφορίες σε χάρτη, ενώ δεν υπάρχει καμία εφαρμογή που να συνδυάζει τις δύο πληροφορίες (φαρμακεία, καύσιμα) σε πλήρη μορφή με έγκυρη ενημέρωση. Συνεπώς *η εφαρμογή που υλοποιήθηκε σε αυτήν την εργασία, είναι η πρώτη που έρχεται να καλύψει αυτό το κενό*.

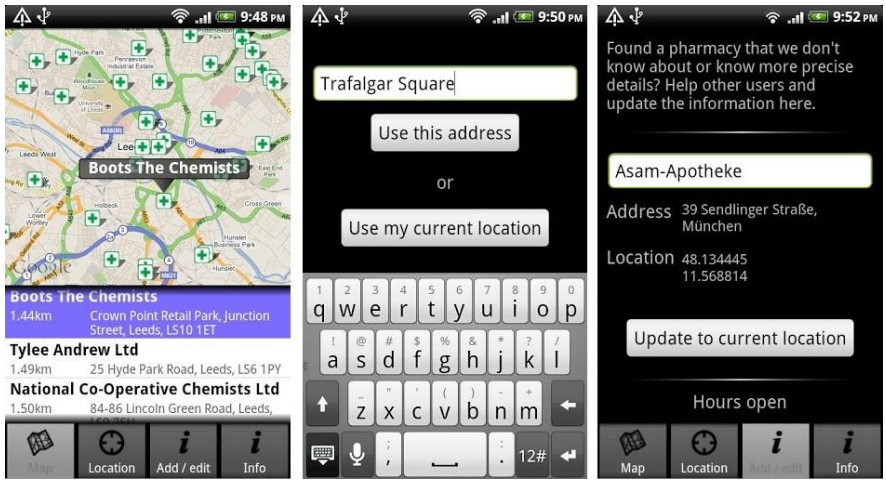
Το όφελος για το δημιουργό της εφαρμογής είναι η ενασχόληση και εξοικείωση με τεχνολογίες αιχμής σχετικές με τις έξυπνες κινητές συσκευές και το λειτουργικό σύστημα Android, και μάλιστα στην περίοδο άνθησης τους σε επίπεδο ανάπτυξης επαγγελματικού τύπου εφαρμογής. Η εμπειρία και οι γνώσεις που συλλέχτηκαν κατά την υλοποίηση αυτής της εργασίας είναι ένα πολύτιμο εφόδιο για κάθε προγραμματιστή και μπορούν να αξιοποιηθούν και σε άλλες εφαρμογές, ενώ διερευνώνται τρόποι εκμετάλλευσης της παρούσας εφαρμογής.

4. Παραδείγματα Σχετικών Εφαρμογών

Δεν είναι λίγες οι ήδη υπάρχουσες εφαρμογές που αντιμετωπίζουν παρόμοια θέματα. Στις παραγράφους που ακολουθούν παρουσιάζονται κάποιες από αυτές.

4.1. "Find Pharmacies" για Android

Η εφαρμογή Find Pharmacies[1] της Elbatrop Ltd. Διατίθεται χωρίς κόστος από το Google Play και έχει στόχο να βοηθάει στον εντοπισμό κοντινών φαρμακείων σε οποιοδήποτε μέρος του κόσμου. *Ωστόσο, δεν περιλαμβάνει τα φαρμακεία της πόλης των Σερρών.* Δίνει στο χρήστη τη δυνατότητα αναζήτησης φαρμακείου γύρω από την τρέχουσα θέση του, ή σε μια άλλη περιοχή μέσω ενός τοπωνύμιου ή ενός ταχυδρομικού κώδικα. Υποστηρίζει λήψη οδηγιών, ανάλογα με το μέσο μεταφοράς (πεζός, με αυτοκίνητο, με ποδήλατο). Δίνει τη δυνατότητα στο χρήστη, να δηλώσει μέσα από την εφαρμογή, ένα φαρμακείο που δεν υπάρχει καταχωρημένο, στο OpenStreetMap project[2]. Έτσι, ο χρήστης μπορεί να συνεισφέρει στην καταγραφή χρήσιμων πληροφοριών που μπορεί να τις χρησιμοποιήσει όποιος επιθυμεί [3].



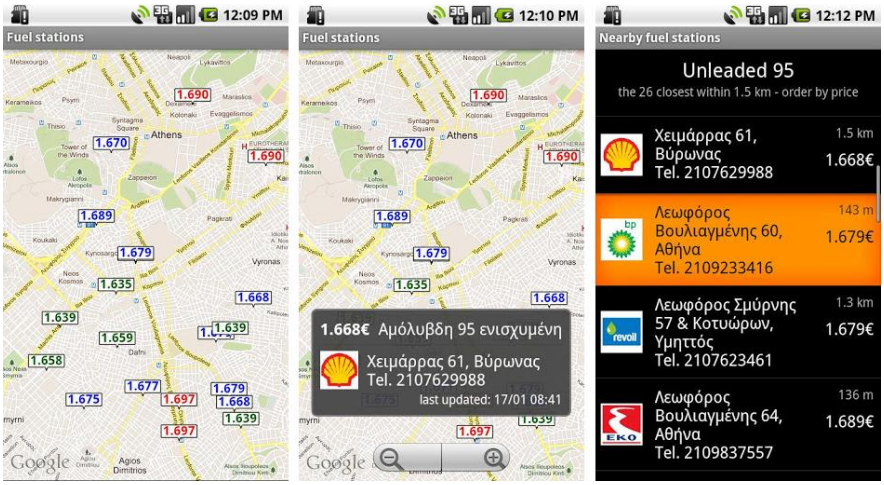
Εικόνα 3. Στιγμιότυπα της εφαρμογής Find Pharmacies

4.2. "Fuel Prices in Greece" για Android

Η εφαρμογή Fuel Prices in Greece[3] (Εικόνα 4) της εταιρείας 4real.gr διατίθεται ελεύθερα μέσω του Google Play. Η εφαρμογή εμφανίζει στο χάρτη τα βενζινάδικα με τις χαμηλότερες τιμές καυσίμων (βενζίνης, πετρελαίου και υγραερίου κίνησης), είτε κοντά στη θέση του χρήστη, είτε οπουδήποτε στην Ελλάδα. Οι τιμές προέρχονται από το Παρατηρητήριο

[1] *Εφαρμογή διαθέσιμη από το Google Play στη διεύθυνση https://play.google.com/store/apps/details?id=com.elbatrop.pharmacies&hl=el*
[2] *OpenStreetMap Project στη διεύθυνση: http://www.openstreetmap.org*
[3] *Εφαρμογή διαθέσιμη από το Google Play στη διεύθυνση https://play.google.com/store/apps/details?id=gr.x4real.fuelprices.android&feature=search_result#?t=W251bGwsMSwxLDEsImdyLng0cmVhbC5mdWVscHJpY2VzLmFuZHJvaWQiXQ..*

Τιμών Υγρών Καυσίμων του Υπουργείου Ανάπτυξης και από το www.gasprice.gr [4].



Εικόνα 4. Στιγμιότυπα της εφαρμογής Fuel Prices

4.3. "LiveMap" για iOS

Η εφαρμογή LiveMap[4] από την Terra Mapping The Globe ltd. είναι διαθέσιμη χωρίς κόστος από το App Store της Apple (Εικόνα 5). Περιλαμβάνει ζωντανά σημεία ενδιαφέροντος για:

- εφημερεύοντα φαρμακεία σε περισσότερες από 60 πόλεις σε όλη την Ελλάδα
- εφημερεύοντα νοσοκομεία για Αθήνα και Θεσσαλονίκη ανά ειδικότητα
- τιμές υγρών καυσίμων για 4.000 περίπου πρατήρια σε όλη τη χώρα με ωριαία ανανέωση

Προσφέρει δυνατότητα αναζήτησης σημείων ενδιαφέροντος:

- με ελεύθερη αναζήτηση
- με κατηγορίες/υποκατηγορίες σημείων ενδιαφέροντος
- με auto-complete κατηγοριών
- 90.000 σημεία ενδιαφέροντος σε 280 κατηγορίες για όλη τη χώρα

Υποστηρίζει εύρεση διαδρομής (πεζή ή με αυτοκίνητο) [5]. Και αυτή η εφαρμογή παρουσιάζει την αδυναμία να *μην περιλαμβάνει ζωντανή ενημέρωση για τα φαρμακεία μόνο στο Νομό Σερρών*, όπως φαίνεται από την

[4] *Η εφαρμογή είναι διαθέσιμη από το app store της Apple στη διεύθυνση: https://itunes.apple.com/us/app/livemap/id415607850?mt=8*

Εικόνα 6, όπου φαίνεται το «κενό» που υπάρχει στο Νομό ενδιαφέροντος κατά τη χρήση της εφαρμογής.

Εικόνα 5. Στιγμιότυπα της εφαρμογής LiveMap

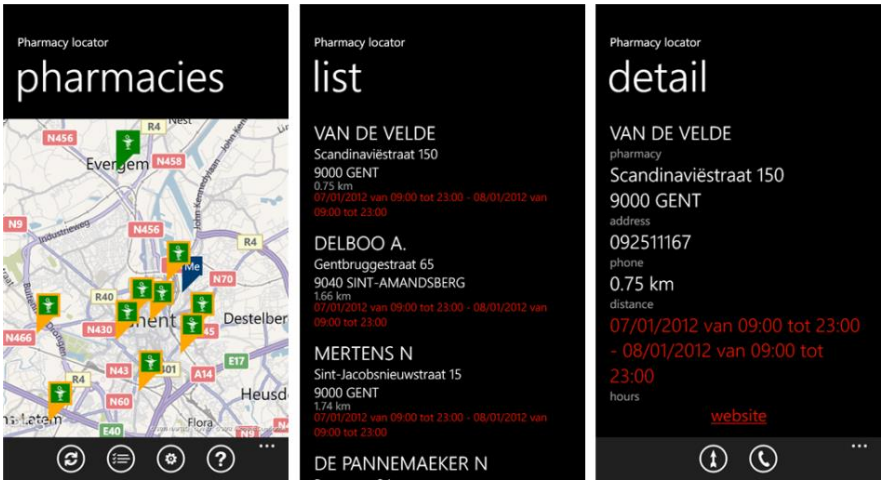
Εικόνα 6. Στιγμιότυπο από την εφαρμογή LiveMap-Φαρμακεία όπου φαίνεται η έλλειψη στοιχείων για το Νομό Σερρών

4.4. "Pharmacy Locator" για Windows Phone

Η εφαρμογή Pharmacy Locator[5] της εταιρείας DepSoft (Εικόνα 7) διατίθεται χωρίς κόστος από το Windows Phone Apps+Games Store. Κατά την εκκίνηση της εφαρμογής Pharmacy Locator, η εφαρμογή αναζητεί όλα τα εφημερεύοντα φαρμακεία κοντά στην τοποθεσία του χρήστη. Αν και η

[5] *Η εφαρμογή είναι διαθέσιμη μέσω του Windows Phone Apps+Games Store στη διεύθυνση http://www.windowsphone.com/en-us/store/app/pharmacy-locator/96b9853c-2470-41c7-914f-2e897dce8816*

εφαρμογή παρέχει πληροφόρηση μόνο για το Βέλγιο, μελετήθηκε ως προς τη λειτουργικότητά της για λόγους σύγκρισης. Υπάρχει δυνατότητα προβολής των φαρμακείων σε χάρτη ή σε λίστα [6].

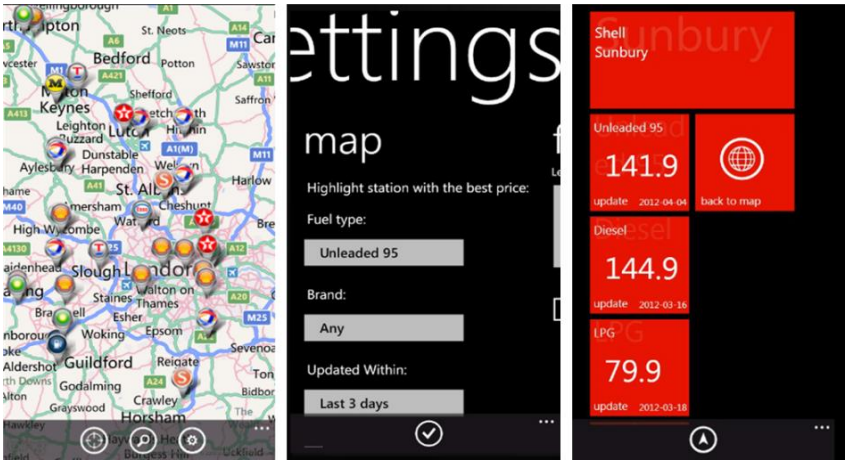


Εικόνα 7. Στιγμιότυπα της εφαρμογής Phamacy Locator

4.5. "Petrol Price Finder UK" για Windows Phone

Η εφαρμογή Petrol Price Finder UK[6] της εταιρείας trandaDesign (Εικόνα 8) είναι διαθέσιμη χωρίς κόστος μέσω του Windows Phone Apps+Games Store και προβάλλει σε χάρτη τα πρατήρια καυσίμων της Αγγλίας και δίνει στους χρήστες την δυνατότητα να ελέγξουν την τιμή των καυσίμων. Τα δεδομένα προέρχονται από http://whatgas.com/. Και σε αυτή την περίπτωση αν και η εφαρμογή αναφέρεται σε άλλη χώρα μελετήθηκε από πλευράς λειτουργικότητας για λόγους σύγκρισης [7].

[6] *Η εφαρμογή είναι διαθέσιμη μέσω του Windows Phone Apps+Games Store στη διεύθυνση http://www.windowsphone.com/en-us/store/app/petrol-price-finder-uk/039959ce-1b5b-e011-854c-00237de2db9e*



Εικόνα 8. Στιγμιότυπα της εφαρμογής Petrol Price Finder UK

5. Η βασική δομή της εργασίας

- **Κεφάλαιο 1: Εισαγωγή** – Στο κεφάλαιο αυτό περιγράφεται το πρόβλημα που αντιμετωπίζει η εργασία, οι λόγοι που παρουσιάζει ενδιαφέρον (σκοπός, αναγκαιότητα, οφέλη) και παρουσιάζεται περιληπτικά η προσέγγιση που ακολουθήθηκε. Αναφέρεται τέλος οποιαδήποτε συμβολή στην επιστήμη και γνώση έχει να επίδειξη η εκπόνηση της εργασίας. Επίσης αναφέρονται άλλες εργασίες που σχετίζονται με το θέμα και ο τρόπος με τον οποίο έχει ήδη αντιμετωπιστεί από άλλους.
- **Κεφάλαιο 2: Android και ανάπτυξη εφαρμογών** – Στο κεφάλαιο αυτό παρουσιάζεται αναλυτικά όλο το θεωρητικό και πρακτικό υπόβαθρο που απαιτείται για την υλοποίηση της εφαρμογών Android και ειδικότερα του λογισμικού της παρούσας εργασίας.
- **Κεφάλαιο 3: Περιγραφή προτεινόμενης Λύσης** – Στο κεφάλαιο αυτό περιγράφεται αναλυτικά η λειτουργία λογισμικού της εργασίας, παρουσιάζονται η αρχιτεκτονική και ο σχεδιασμός του και γίνεται εκτενής σχολιασμός τους με σχήματα, εικόνες, πίνακες, διαγράμματα. Ακόμη περιλαμβάνεται παρουσίαση της εφαρμογής και μελλοντικές επεκτάσεις και βελτιώσεις της λύσης που έχει δοθεί.
- **Βιβλιογραφία** – Στο κεφάλαιο αυτό παρουσιάζεται όλη η σχετική βιβλιογραφία από όλα τα κεφάλαια.
- **Παραρτήματα Α: Πηγαίος Κώδικας Εφαρμογής** – Στο παράρτημα αυτό περιέχεται ο πηγαίος κώδικας της εφαρμογής.
- **Παραρτήματα Β: Ανάπτυξη android εφαρμογής με προβολή χάρτη** – Στο παράρτημα αυτό περιλαμβάνονται οδηγίες για την

υλοποίηση μιας απλής εφαρμογής Android με μόνη λειτουργία την προβολή ενός διαδραστικού χάρτη.

Android και ανάπτυξη εφαρμογών

1. Το λειτουργικό σύστημα Android

Το Android είναι ένα λειτουργικό σύστημα ανοιχτού κώδικα, βασισμένο στο Linux, για φορητές συσκευές όπως smartphones και tablets. Τον Ιούλιο του 2005, η Google εξαγόρασε την Android Inc, μια μικρή εταιρεία με έδρα το Palo Alto στην California των ΗΠΑ. Εκείνη την εποχή ελάχιστα ήταν γνωστά για τις δραστηριότητες της Android Inc, εκτός του ότι ανέπτυσσαν λογισμικό για κινητά τηλέφωνα. Στην Google, η ομάδα με επικεφαλής τον Andy Rubin ανέπτυξε μια πλατφόρμα που στηρίζεται στον πυρήνα του Linux, την οποία προώθησαν με την παροχή ενός ευέλικτου, αναβαθμίσιμου συστήματος. Αργότερα αναπτύχθηκε από την Open Handset Alliance, η οποία είναι μια κοινοπραξία εταιριών λογισμικού, κατασκευής hardware και τηλεπικοινωνιών, οι οποίες σχετίζονται με την ανάπτυξη και εξέλιξη ανοιχτών προτύπων στις φορητές συσκευές. Η πρώτη παρουσίαση της πλατφόρμας Android έγινε στις 5 Νοεμβρίου 2007, παράλληλα με την ανακοίνωση της ίδρυσης του οργανισμού Open Handset Alliance. Το πρώτο τηλέφωνο με λειτουργικό σύστημα Android είναι το HTC T-Mobile G1 (Εικόνα 9) και βγήκε στην αγορά τον Οκτώβριο του 2008 [8],[9].

Εικόνα 9. Το πρώτο Android-powered τηλέφωνο, HTC T-Mobile G1

Τον Ιανουάριο του 2010, η Google ξεκίνησε την σειρά συσκευών Nexus, μια σειρά από smartphones και tablets με λειτουργικό σύστημα Android, που κατασκευάζονται από εταίρο κατασκευαστή για την Google. Η HTC συνεργάστηκε με την Google για να κυκλοφορήσει το πρώτο smartphone Nexus, το Nexus One (Εικόνα 10) [8],[10]. Η σειρά έχει έκτοτε ενημερωθεί με νεότερες συσκευές, όπως το Nexus 4 τηλέφωνο και το Nexus 10 tablet, τα οποία κατασκευάστηκαν από την LG και Samsung αντίστοιχα. Τα Nexus τηλέφωνα και ταμπλέτες της Google, επιδεικνύουν τις νεότερες εκδόσεις Android, καθώς και τα νεότερα χαρακτηριστικά υλικού [8].

Εικόνα 10. Το πρώτο Nexus τηλέφωνο, HTC Nexus One

Η Google δημοσίευσε το μεγαλύτερο μέρος του κώδικα του Android υπό τους όρους της άδειας Apache, μιας ελεύθερης άδειας λογισμικού, που επιτρέπει στο λογισμικό να τροποποιηθεί ελεύθερα και να διανεμηθεί από κατασκευαστές συσκευών, πάροχους ασύρματης επικοινωνίας και προγραμματιστές. Επιπλέον, το Android έχει μια μεγάλη κοινότητα προγραμματιστών που αναπτύσσουν εφαρμογές (apps), που επεκτείνουν τη λειτουργικότητα των συσκευών. Τον Οκτώβριο του 2012, υπήρχαν περίπου 700.000 διαθέσιμες εφαρμογές για Android, και ο εκτιμώμενος αριθμός των εφαρμογών που μεταφορτώθηκαν από το Google Play ήταν 25 δισεκατομμύρια [8].

Το πλάνο διάχυσης που υιοθετήθηκε επέτρεψε στο Android για να γίνει η πιο ευρέως χρησιμοποιούμενη πλατφόρμα για έξυπνες κινητές συσκευές στον κόσμο, ξεπερνώντας στα τέλη του 2010 το Symbian ως το λογισμικό της επιλογής των εταιρειών τεχνολογίας, που χρειάζονται ένα χαμηλού κόστους, προσαρμόσιμο, ελαφρύ λειτουργικό σύστημα για συσκευές υψηλής τεχνολογίας, αποφεύγοντας την εκ του μηδενός ανάπτυξη. Ως αποτέλεσμα, παρά το γεγονός ότι σχεδιάστηκε αρχικά για κινητά τηλέφωνα και ταμπλέτες, γνώρισε πρόσθετες εφαρμογές για τηλεοράσεις, κονσόλες παιχνιδιών, ψηφιακές φωτογραφικές μηχανές και άλλα ηλεκτρονικά. Η ανοιχτή φύση του Android έχει ενθαρρύνει περαιτέρω μια μεγάλη κοινό-

τητα προγραμματιστών, να χρησιμοποιούν το λογισμικό ανοιχτού κώδικα ως θεμέλιο για τα project τους, τα οποία προσθέτουν νέα χαρακτηριστικά για προχωρημένους χρήστες, ή εισάγουν το Android σε συσκευές που επίσημα είχαν κυκλοφορήσει με άλλο λειτουργικό σύστημα [8].

Το Android είχε στα μέσα του 2012 ένα παγκόσμιο μερίδιο αγοράς smartphone 75% , με 750 εκατομμύρια συσκευές να έχουν ενεργοποιηθεί στο σύνολο και 1,5 εκατομμύρια ενεργοποιήσεις ανά ημέρα. Η επιτυχία του λειτουργικού συστήματος το έχει καταστήσει στόχο για διπλώματα ευρεσιτεχνίας ως μέρος των λεγόμενων "smartphone wars» μεταξύ των εταιρειών τεχνολογίας [8].

1.1. Η διεπαφή του Android

Η διεπαφή χρήστη του Android βασίζεται στον άμεσο χειρισμό χρησιμοποιώντας τις εισόδους αφής που αντιστοιχούν σε πραγματικές ενέργειες όπως το σύρσιμο, τον ελαφρύ χτύπο, το τσίμπημα και το αντίστροφο τσίμπημα για να χειρίζεται τα αντικείμενα στην οθόνη. Η ανταπόκριση στις εισόδους του χρήστη είναι σχεδιασμένη να είναι άμεση και να παρέχει ένα εύχρηστο περιβάλλον αφής, συχνά χρησιμοποιώντας τις δυνατότητες δόνησης της συσκευής να παρέχει απτική ανάδραση στο χρήστη. Εσωτερικό υλικό (όπως γυροσκόπια, επιταχυνσιόμετρα και αισθητήρες εγγύτητας) χρησιμοποιείται από ορισμένες εφαρμογές, ώστε να ανταποκριθούν στις πρόσθετες ενέργειες του χρήστη, όπως η προσαρμογή της οθόνης από κατακόρυφη σε οριζόντια ανάλογα με τον προσανατολισμό της συσκευής, ή δυνατότητα του χρήστη να κατευθύνει ένα όχημα σε ένα αγωνιστικό παιχνίδι περιστρέφοντας τη συσκευή, προσομοιώνοντας τον έλεγχο ενός τιμονιού [8].

Οι Android συσκευές πραγματοποιούν εκκίνηση στην αρχική οθόνη (home screen), το κύριο σημείο πλοήγησης και ενημέρωσης της συσκευής, το οποίο είναι παρόμοιο με την επιφάνεια εργασίας που υπάρχει στους υπολογιστές. Οι αρχικές οθόνες Android συνήθως αποτελούνται από εικονίδια των εφαρμογών και μικροεφαρμογές που καλούνται widgets. Τα εικονίδια των εφαρμογών ξεκινούν τη σχετική εφαρμογή, ενώ τα widgets εμφανίζουν ζωντανά, πληροφορία που απαιτεί αυτόματη ενημέρωση του περιεχομένου, όπως η πρόβλεψη του καιρού, τα εισερχόμενα email του χρήστη και τίτλοι ειδήσεων απευθείας στην αρχική οθόνη. Η αρχική οθόνη μπορεί να αποτελείται από μερικές σελίδες, όπου ο χρήστης μπορεί να περιηγηθεί, σέρνοντας δεξιά ή αριστερά μεταξύ των σελίδων, αλλά και να προσαρμόσει την εμφάνιση της ανάλογα με τις προτιμήσεις του. Εφαρμογές τρίτων που διατίθενται στο Google Play ή σε άλλα καταστήματα εφαρμογών, μπορούν να αλλάξουν εκτενώς το θέμα της αρχικής οθόνης, ακόμα και να μιμηθούν την εμφάνιση άλλων λειτουργικών συστημάτων, όπως το Windows Phone. Οι περισσότεροι κατασκευαστές, και κάποιοι πάροχοι ασύρματης επικοινωνίας, προσαρμόζουν την

εμφάνιση και την αίσθηση των Android συσκευών τους ώστε να διαφοροποιούνται από τους ανταγωνιστές [8].

Στο πάνω μέρος της οθόνης βρίσκεται μια μπάρα κατάστασης (status bar), που δείχνει πληροφορίες για τη συσκευή και τη συνδεσιμότητα της. Αυτή η γραμμή κατάστασης μπορεί να "συρθεί" κάτω, για να αποκαλύψει μια οθόνη κοινοποιήσεων (notification screen), όπου οι εφαρμογές εμφανίζουν σημαντικές πληροφορίες ή ενημερώσεις, όπως η ειδοποίηση νέου e-mail ή SMS κειμένου, με έναν τρόπο που δεν διακόπτει ή ενοχλεί άμεσα το χρήστη. Σε παλαιότερες εκδόσεις του Android οι εν λόγω κοινοποιήσεις μπορούσαν να επιλεχθούν από το χρήστη, ώστε να ανοίξει η σχετική εφαρμογή, αλλά με πρόσφατες ενημερώσεις προστέθηκαν βελτιωμένες λειτουργίες, όπως τη δυνατότητα κλήσης ενός αριθμού απευθείας από την ειδοποίηση της αναπάντητης κλήσης χωρίς να χρειάζεται το άνοιγμα του παραθύρου κλήσης εφαρμογών πρώτα. Οι κοινοποιήσεις παραμένουν στην οθόνη κοινοποιήσεων, μέχρι να διαβαστούν ή να απορριφθούν από το χρήστη [8].

1.2 Εξέλιξη του Android

Το ιστορικό εκδόσεων του Android λειτουργικού συστήματος ξεκίνησε με την κυκλοφορία του Android beta το Νοέμβριο του 2007. Η πρώτη εμπορική έκδοση, Android 1.0, κυκλοφόρησε το Σεπτέμβριο του 2008. Το Android είναι υπό συνεχή ανάπτυξη από την Google και την Open Handset Alliance (OHA), και έχει γνωρίσει μια σειρά ενημερώσεων στη βάση του λειτουργικού συστήματος από την αρχική κυκλοφορία του. Αυτές οι ενημερωμένες εκδόσεις συνήθως διορθώνουν σφάλματα και προσθέτουν νέες δυνατότητες [11].

Από τον Απρίλιο του 2009, οι Android εκδόσεις έχουν αναπτυχθεί υπό κωδική ονομασία και κυκλοφόρησαν σε αλφαβητική σειρά: Cupcake, Donut, Eclair, Froyo, Gingerbread, Honeycomb, Ice Cream Sandwich και Jelly Bean. Από το 2013, πάνω από 500 εκατομμύρια ενεργές συσκευές χρησιμοποιούν το Android OS σε όλο τον κόσμο. Η πιο πρόσφατη σημαντική Android ενημέρωση ήταν η Jelly Bean 4.2, η οποία κυκλοφόρησε στις εμπορικές συσκευές το Νοέμβριο του 2012 [11]. Στις παραγράφους που ακολουθούν αναφέρονται βασικά χαρακτηριστικά των κυριότερων εκδόσεων του λειτουργικού συστήματος Android.

1.2.1. Android 1.5 Cupcake (API level 3)

Η έκδοση “Cupcake” (Εικόνα 11), βασισμένη στο Linux Kernel 2.6.27, παρουσιάστηκε στις 30 Απριλίου του 2009. Η ενημέρωση περιλαμβάνει αρκετά νέα χαρακτηριστικά και τις τροποποιήσεις στο UI:

- Υποστήριξη Widget (μινιατούρες από εφαρμογές οι οποίες μπορούν να ενσωματωθούν σε άλλες εφαρμογές όπως π.χ. αρχική

οθόνη και να δέχονται περιοδικές ενημερώσεις) και δυνατότητα προσθήκης φακέλων στην αρχική οθόνη.
- Αναπαραγωγή και εγγραφή βίντεο σε MPEG-4 και 3GP μορφές.
- Χρήση λειτουργίας αντιγραφή-επικόλληση και στις εφαρμογές περιήγησης διαδικτύου.
- Αρχείο καταγραφής κλήσεων με συγκεκριμένη ημερομηνία και ώρα και πρόσβαση μέσα από το αρχείο αυτό στη καρτέλα κάποιας επαφής.
- Εικονικό πληκτρολόγιο με πρόβλεψη λέξεων.
- Υποστήριξη Bluetooth A2DP και AVRCP.
- Επιλογή αυτόματης περιστροφής οθόνης.
- Δυνατότητα ανεβάσματος φωτογραφιών στο Picasa και βίντεο στο YouTube απευθείας από το κινητό [11]

Εικόνα 11. Το λογότυπο του Android 1.5 "Cupcake"

1.2.2. Android 1.6 Donut (API level 4)

Η έκδοση "Donut" (Εικόνα 12), βασισμένη στο Linux Kernel 2.6.29, παρουσιάστηκε στις 15 Σεπτεμβρίου του 2009. Στην ενημερωμένη έκδοση περιλαμβάνονται πολλά νέα χαρακτηριστικά:

- Φωνητική αναζήτηση και αναζήτηση κειμένου από την αρχική οθόνη σε σελιδοδείκτες, επαφές και διαδίκτυο.
- Ευκολότερη αναζήτηση στο Android Market (Google Play) και δυνατότητα εμφάνισης στιγμιότυπων οθόνης της κάθε εφαρμογής.
- Εφαρμογές Συλλογής, Φωτογραφικής Μηχανής και Βίντεο πλήρως ολοκληρωμένες με γρηγορότερη πρόσβαση στη κάμερα.
- Δυνατότητα πολλαπλής επιλογής φωτογραφιών από την εφαρμογή Συλλογή για διαγραφή ή μετακίνηση.
- Υποστήριξη τεχνολογίας CDMA/EVDO, 802.1x, VPNs και text-to-speech.
- Υποστήριξη οθονών αναλύσεως WVGA.
- Βελτίωση ταχύτητας στις εφαρμογές αναζήτησης και κάμερας.
- Gesture framework και εργαλείο ανάπτυξης Gesture Builder [11]

Εικόνα 12. Το λογότυπο του Android 1.6 "Donut"

1.2.3. Android 2.0 Eclair (API level 5)

Η έκδοση “Eclair” (Εικόνα 13), βασισμένη και αυτή στον Linux Kernel 2.6.29, παρουσιάστηκε στις 26 Οκτωβρίου του 2009, ενώ τον Ιανουάριο του 2010 επανεκδόθηκε σε Android 2.1 (API level 7) Eclair (MR1). Περιελάμβανε:

- Δυνατότητα προσθήκης πολλαπλών λογαριασμών email σε μία συσκευή και συγχρονισμό με τις επαφές.
- Υποστήριξη Bluetooth 2.1.
- Δυνατότητα αναζήτησης στα αποθηκευμένα μηνύματα κειμένου (sms) και μηνύματα πολυμέσων (mms).
- Δυνατότητα επιλογής της αυτόματης διαγραφής παλαιότερων μηνυμάτων όταν ξεπεραστεί ένα καθορισμένο όριο.
- Νέα χαρακτηριστικά στη κάμερα όπως: υποστήριξη φωτογραφικού φλας, ψηφιακό ζουμ, λειτουργία σκηνής, ισορροπία λευκού, εφέ χρώματος και macro εστίασης.
- Βελτιωμένη ταχύτητα δαχτυλογράφησης στο εικονικό πληκτρολόγιο, με λεξικό που συμπεριλαμβάνει προτάσεις και ονόματα επαφών.
- Ανανεωμένη διεπαφή χρήστη (UI) στο πρόγραμμα περιήγησης διαδικτύου με μικρογραφίες σελιδοδεικτών, μεγέθυνση διπλού αγγίγματος και υποστήριξη HTML5.
- Βελτιωμένη ταχύτητα του υλικού (hardware) και ανανεωμένη διεπαφή χρήστη.
- Υποστήριξη περισσότερων μεγεθών οθόνης και ανάλυσης, με καλύτερη αναλογία αντίθεσης.
- Βελτιωμένο Google Map 3.1.2.
- Δυνατότητα αντίληψης Multi-Touch [11]

Εικόνα 13. Το λογότυπο του Android 2.0 "Eclair"

1.2.4. Android 2.2 Froyo (API level 8)

Η έκδοση "Froyo" (Εικόνα 14), βασισμένη στο Linux Kernel 2.6.32, παρουσιάστηκε στις 20 Μαΐου του 2010. Περιελάμβανε:

- Βελτιστοποίηση της ταχύτητας του λειτουργικού συστήματος, της διαχείρισης μνήμη και της γενικής απόδοσης.
- Ενσωμάτωση του Chrome V8 JavaScript στις εφαρμογές περιήγησης διαδικτύου.
- Βελτιστοποίηση ταχύτητας εφαρμογών μέσω του JIT (Just-In-Time compilation).
- Υποστήριξη του C2DM (Android Cloud to Device Messaging).
- Αναβαθμισμένη υποστήριξη Microsoft Exchange συμπεριλαμβανομένων των πολιτικών ασφαλείας.
- USB Tethering και λειτουργία Wi-Fi Hotspot.
- Βελτιωμένη εφαρμογή Launcher με συντομεύσεις για τις εφαρμογές τηλεφώνου και τις περιήγησης διαδικτύου.
- Προσθήκη επιλογής για την απενεργοποίηση της πρόσβασης δεδομένων μέσω κινητής τηλεφωνίας.
- Αναβάθμιση του Android Market με δυνατότητα αυτόματων ενημερώσεων.
- Γρήγορή μετάβαση ανάμεσα στις γλώσσες πληκτρολογίου και στα λεξικά.
- Φωνητική κλήση και κοινή χρήση επαφών μέσω Bluetooth.
- Υποστήριξη αριθμητικών και αλφαριθμητικών κωδικών πρόσβασης.
- Δυνατότητα ανεβάσματος αρχείων μέσω των εφαρμογών περιήγησης διαδικτύου.
- Δυνατότητα εγκατάστασης εφαρμογών στην κάρτα μνήμης
- Υποστήριξη Adobe Flash.
- Υποστήριξη έξτρα μεγάλων PPI (Pixel Per Inch) οθονών όπως 4 ιντσών 720p [11]

Εικόνα 14. Το λογότυπο του Android 2.2 "Froyo"

1.2.5. Android 2.3 Gingerbread (API level 9)

Η έκδοση "Gingerbread" (Εικόνα 15), βασισμένη στο Linux Kernel 2.6.35.7, παρουσιάστηκε στις 6 Δεκεμβρίου του 2010, ενώ το Φεβρουάριο του 2011 επανεκδόθηκε σε Android 2.3.3. Περιελάμβανε:

- Πιο απλό και ταχύ User Interface.
- Υποστήριξη πολύ μεγάλων μεγεθών οθονών και ανάλυσης (WXGA και πάνω).
- Προεγκατεστημένη υποστήριξη για SIP (Session Initiation Protocol) VoIP (Voice over Internet Protocol) τηλεφωνία.
- Γρηγορότερη και πιο έξυπνη εισαγωγή κειμένου μέσα από το εικονικό πληκτρολόγιο, με μεγαλύτερη ακρίβεια, με καλύτερες προτεινόμενες λέξεις και με την επιλογή εισαγωγής κειμένου μέσω της φωνητικής λειτουργίας.
- Ανανεωμένη λειτουργία αντιγραφής-επικόλλησης σε όλο το λειτουργικό σύστημα και στις εφαρμογές του.
- Υποστήριξη NFC (Near Field Communication).
- Νέα ηχητικά εφέ.
- Εφαρμογή Download Manager για το κατέβασμα αρχείων από email, περιηγητή διαδικτύου ή οποιαδήποτε άλλη εφαρμογή.
- Προεγκατεστημένη υποστήριξη για πολλαπλές κάμερες στη συσκευή.
- Υποστήριξη WebM/VP8 για αναπαραγωγή βίντεο και AAC για κωδικοποίηση ήχου.
- Βελτιωμένη διαχείριση ενέργειας με καλύτερη διαχείριση των εφαρμογών που κατανάλωναν ενέργεια από τη συσκευή για μεγάλο χρονικό διάστημα ακόμα και όταν βρισκόταν σε κατάσταση χαμηλής λειτουργίας.
- Αυξημένη υποστήριξη για την ανάπτυξη κώδικα του λειτουργικού.
- Αλλαγή από YAFFS (Yet Another Flash File System) σε σύστημα αρχείου ext4.

- Βελτίωση ήχου και γραφικών για τους προγραμματιστές παιχνιδιών.
- Προεγκατεστημένη υποστήριξη για περισσότερους αισθητήρες (όπως γυροσκόπιο και βαρόμετρο).
- Αυξημένη απόδοση με την ταυτόχρονη συλλογή και διαγραφεί των αντικειμένων που δεν είναι πλέον χρήσιμα από το πρόγραμμα [11]

Εικόνα 15. Το λογότυπο του Android 2.3 "Gingerbread"

1.2.6. Android 3.0 Honeycomb (API level 11)

Η έκδοση "Honeycomb" (Εικόνα 16), βασισμένη στο Linux Kernel 2.6.36, παρουσιάστηκε στις 9 Μαΐου του 2011,με την ιδιαιτερότητα ότι προοριζόταν αποκλειστικά για tablets. Οι αλλαγές που έγιναν στην έκδοση αυτή έχουν να κάνουν κυρίως με τη βελτίωση της υποστήριξης των tablets:

- Επανασχεδιασμένο εικονικό πληκτρολόγιο για αποτελεσματικότερη, γρηγορότερη και ακριβέστερη πληκτρολόγηση σε μεγαλύτερες οθόνες.
- Πιο απλοποιημένη και πιο έξυπνη λειτουργία αντιγραφής-επικόλλησης.
- Αντικατάσταση παραθύρων με καρτέλες στην εφαρμογή περιήγησης διαδικτύου και δυνατότητα ανώνυμης περιήγησης.
- Γρήγορη πρόσβαση στη κάμερα και στα χαρακτηριστικά της (εστίαση, φλας, ζουμ, μπροστινή κάμερα κ.α.).
- Δυνατότητα εμφάνισης φωτογραφιών και άλμπουμ σε λειτουργία πλήρης οθόνη.
- Υποστήριξη βίντεο-συνομιλίας (video chat) μέσω του Google Talk.
- Γρηγορότερη απόδοση του υλικού συστήματος (hardware).
- Υποστήριξη πολυπύρηνων επεξεργαστών.
- Δυνατότητα κρυπτογράφησης όλων των δεδομένων του χρήστη.
- Νέα παράθυρα διεπαφής χρήστη (UI) για καλύτερη οργάνωση και εύρεση επαφών και με αποτελεσματικότερη προβολή και οργά-

νωση των μηνυμάτων email, που επιτρέπει στους χρήστες να επιλέξουν ταυτόχρονα περισσότερα από ένα μήνυμα.
- Βελτίωση στοίβας HTTPS με την ένδειξη ονόματος διακομιστή SNI (Server Name Indication) [11]

Εικόνα 16. Το λογότυπο του Android 3.0 "Honeycomb"

1.2.7. Android 4.0 Ice Cream Sandwich (API level 14)

Η έκδοση “Ice Cream Sandwich”, βασισμένη στο Linux Kernel 3.0.1, παρουσιάστηκε στις 19 Οκτωβρίου του 2011. Περιελάμβανε:

- Ευκολότερη δημιουργία φακέλων με τη λειτουργία drag-and-drop.
- Πιο προσαρμόσιμο Launcher.
- Βελτιωμένος τηλεφωνητής με ικανότητα να επιταχύνει ή να επιβραδύνει τα μηνύματα του.
- Ικανότητα λήψης στιγμιότυπων οθόνης (Screenshots).
- Διόρθωση των σφαλμάτων του πληκτρολογίου.
- Δυνατότητα πρόσβασης σε εφαρμογές απευθείας από την οθόνη κλειδώματος.
- Βελτιωμένη λειτουργία αντιγραφής-επικόλλησης.
- Face Unlock, μια λειτουργία που επιτρέπει στους χρήστες να ξεκλειδώνουν τα κινητά τους με λογισμικό αναγνώρισης προσώπου και με την προϋπόθεση βέβαια να υπάρχει στη συσκευή μπροστινή κάμερα.
- Νέα εφαρμογή περιήγησης διαδικτύου με την υποστήριξη του Google Chrome η οποία αντί για νέα παράθυρα εμφάνιζε καρτέλες, με μέγιστο αριθμό καρτελών 16 καρτέλες.
- Επιλογή στις ρυθμίσεις που προειδοποιεί το χρήστη ότι πλησιάζει ένα συγκεκριμένο όριο χρήσης δεδομένων (που έχει ορίσει ο ίδιος) και την απενεργοποιεί όταν υπερβεί το όριο.
- Δυνατότητα να κλείνει τις εφαρμογές που τρέχουν στο παρασκήνιο και καταναλώνουν δεδομένα.
- Βελτιωμένη εφαρμογή της κάμερας με: μηδενική υστέρηση κλείστρου, λειτουργία πανοραμικής φωτογραφίας και δυνατότητα χρήσης ζουμ κατά την εγγραφή βίντεο.

- Προεγκατεστημένο πρόγραμμα επεξεργασίας φωτογραφιών.
- Ανανέωση της εφαρμογής People, με συγχρονισμό των επαφών με τα δεδομένα τους από τα κοινωνικά δίκτυα.
- Μεταφορά δεδομένων μεταξύ συσκευών με τη χρήση του συστήματος NFC (Android Beam).
- Υποστήριξη της μορφής εικόνας WebP που έχει να κάνει με τη συμπίεση της εικόνας χωρίς απώλειες.
- Λειτουργία που επιτρέπει δύο συσκευές να συνδεθούν μεταξύ τους μέσω ασυρμάτου δικτύου (Wi-Fi Direct).
- Εγγραφή βίντεο υψηλής ευκρίνειας (1080p) [11]

Εικόνα 17. Το λογότυπο του Android 4.0 "Ice Cream Sandwich"

1.2.8. Android 4.1/4.2 Jelly Bean (API level 16/17)

Η έκδοση "Jelly Bean" (Εικόνα 18), βασισμένη στο Linux Kernel 3.0.31, Κυκλοφόρησε στις 27 Ιουνίου του 2012 στην έκδοση 4.1 και στη αργότερα επανεκδόθηκε στην έκδοση 4.2. Περιελάμβανε:

- Επανασχεδιασμένο UI και widgets για χρήση σε tablets και σε smartphones.
- Νέες δυνατότητες μέσω της κάμερας.
- Δυνατότητα απενεργοποίησης των ειδοποιήσεων μιας εφαρμογής για μια συγκεκριμένη περίοδο.
- Συντομεύσεις και widgets μπορούν αυτόματα να αναδιαταχτούν ή μεγέθους για να ώστε τα νέα στοιχεία να χωρούν στην Αρχική Οθόνη.
- Δυνατότητα μεταφοράς δεδομένων μέσω Bluetooth για το Android Beam.
- Βελτιωμένη φωνητική αναζήτηση.
- Νέα εφαρμογή αναζήτησης Google Now.
- Το διαθέσιμο πρόγραμμα περιήγησης Android αντικαθίσταται με την Android mobile έκδοση του Google Chrome, στις συσκευές με προεγκατεστημένο το Android 4.1.
- Υποστήριξη πολλαπλών λογαριασμών χρηστών (για tablet μόνο).

- Υποστήριξη προβολής σε εξωτερική οθόνη μέσω wi-fi.
- Προσθήκη νέων λειτουργιών κλειδώματος οθόνης.
- Group Messaging.
- Βελτιωμένη φωνητική αναζήτηση.
- Επανασχεδιασμένες ειδοποιήσεις των εφαρμογών [11]

Εικόνα 18. Το λογότυπο του Android 4.1 "Jelly Bean"

1.3. Η Αρχιτεκτονική του Android

Στον πυρήνα της πλατφόρμας Android, όπως παρουσιάζεται στην Εικόνα 19, βρίσκεται ένα Linux kernel το οποίο είναι υπεύθυνο για τη διαχείριση των οδηγών συσκευών (device drivers), τον έλεγχο πρόσβασης στους πόρους του συστήματος, τη διαχείριση μνήμης και τις λοιπές υπηρεσίες που παρέχει ένα λειτουργικό σύστημα. Στους device drivers συγκαταλέγονται αυτοί της οθόνης, του WiFi, της κάμερας, του ήχου κ.α [12],[13].

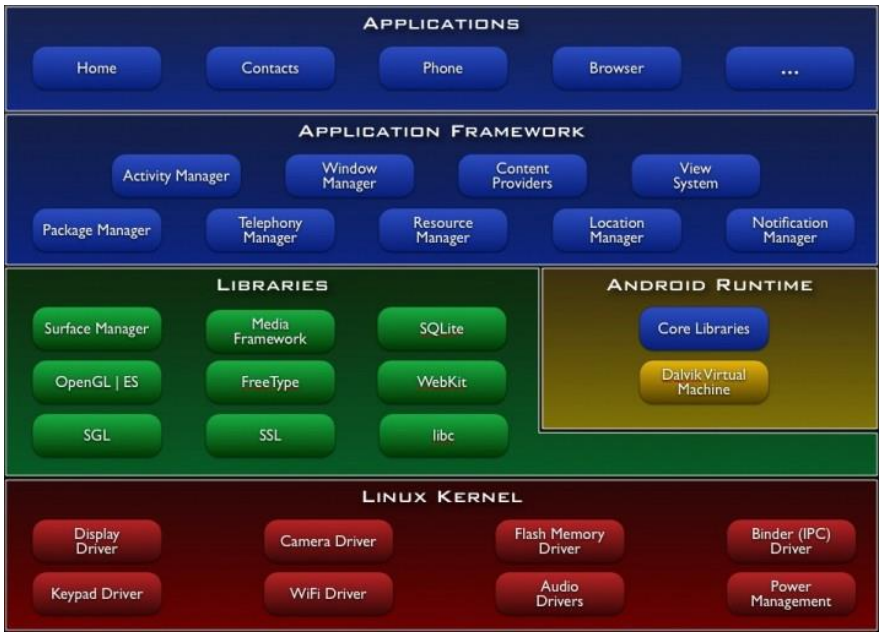
Ένα επίπεδο ψηλότερα βρίσκονται οι βιβλιοθήκες του συστήματος που είναι γραμμένες σε C++ και περιλαμβάνουν την OpenGL, την SQLite, τη Media library κ.α. Οι εφαρμογές που τρέχουν σε μια έξυπνη κινητή συσκευή μπορούν να έχουν πρόσβαση στις βιβλιοθήκες αυτές μέσω της Dalvik JVM (Java Virtual Machine). Οι εφαρμογές Android είναι γραμμένες σε Java και άρα για να τρέξουν χρειάζονται το αντίστοιχο περιβάλλον. Όπως λοιπόν για να εκτελέσουμε μία εφαρμογή σε ένα PC είναι απαραίτητο να είναι εγκατεστημένο το κατάλληλο JRE (Java Runtime Environment), για τις εφαρμογές Android το ρόλο του JRE παίζει η Dalvik VM [12].

Δεδομένης της σαφώς πιο περιορισμένης υπολογιστικής ισχύος καθώς και της ποσότητας διαθέσιμης μνήμης που έχουν οι κινητές συσκευές σε σχέση με τους υπολογιστές, η συγκεκριμένη VM είναι βελτιστοποιημένη στο να χρησιμοποιεί μικρότερα σε μέγεθος αρχεία ενδιάμεσου κώδικα, τα οποία σε αντίθεση με αυτά της Java SE έχουν κατάληξη .dex αντί για .class. Επίσης, σύμφωνα με την Google, διαθέτει και πιο καλά γραμμένο σύστημα απόρριψης μη χρησιμοποιούμενων πληροφοριών (garbage collector). Αν και υπάρχει έκδοση της Java για κινητά τηλέφωνα, η Java

ΜΕ, η Google έκρινε σωστό να χρησιμοποιήσει τη δική της υλοποίηση και έτσι γεννήθηκε η Dalvik. Κάθε εφαρμογή λοιπόν που γράφεται και εκτελείται στο Android, χρησιμοποιεί τη Dalvik και τρέχει σε ξεχωριστό instance του VM [12].

Στο αμέσως υψηλότερο επίπεδο βρίσκεται το Android SDK (Software Development Kit) που περιέχει όλες τις απαραίτητες βιβλιοθήκες για τη συγγραφή εφαρμογών. Ο κώδικας που γράφεται βάσει του SDK για τη δημιουργία εφαρμογών στοχεύει στην εκτέλεση κάποιας συγκεκριμένης λειτουργίας, π.χ. για την πραγματοποίηση μιας κλήσης, την αποστολή ενός SMS, τον εντοπισμό της τρέχουσας θέσης κλπ. Στην ουσία καλεί κάποια από τις μεθόδους που παρέχονται από το συγκεκριμένο SDK. Παρακάτω γίνεται περαιτέρω ανάλυση αυτού του επιπέδου [12].

Οι εφαρμογές βρίσκονται στο υψηλότερο επίπεδο, στην κορυφή του ονομαζόμενου application stack που ονομάζεται Applications Layer [12].



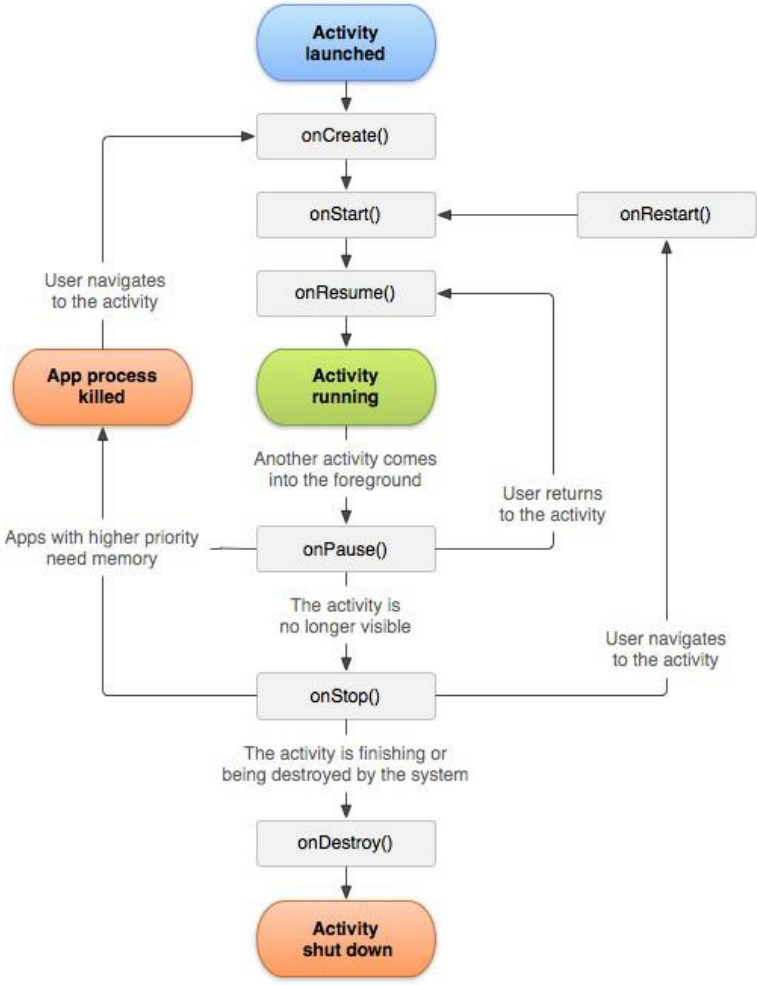
Εικόνα 19. Η αρχιτεκτονική του Android

1.3.1. Application Framework

Το Android παρέχει στους developers μια ανοιχτού κώδικα πλατφόρμα ανάπτυξης και τη δυνατότητα να αναπτύξουν με αυτή καινοτόμες και πλούσιες σε υλικό, εφαρμογές. Οι developers έχουν στη διάθεση τους τη δυνατότητα ελέγχου του υλικού της συσκευής και μέσω αυτής μπορούν να αποκτήσουν πρόσβαση σε υπηρεσίες εντοπισμού, εκτέλεση διεργασιών παρασκηνίου, και πάρα πολλές ακόμη δυνατότητες, οι οποίες βασίζονται στα APIs (Application Programming Interfaces) που είναι διαθέσι-

μα. Οι developers έχουν πρόσβαση σε όλα τα APIs μεταξύ αυτών και στα κύρια APIs που χρησιμοποιούν οι ενσωματωμένες εφαρμογές. Η δομή των εφαρμογών είναι τέτοια που ευνοείται η επαναχρησιμοποίηση δομικών συστατικών, και επίσης επιτρέπεται η χρήση των δυνατοτήτων της μίας εφαρμογής από άλλες εφαρμογές, βέβαια κάτω από τις προδιαγραφές ασφάλειας του Android [14]. Τα σημαντικότερα δομικά στοιχεία του πλαισίου εφαρμογών είναι:

- **Σύστημα προβολών** (View System) – αποτελεί ένα εκτενές σύνολο από αντικείμενα GUI (Graphical User Interface), τα οποία μπορούν να χρησιμοποιηθούν κατά το σχεδιασμό μιας εφαρμογής. Παραδείγματα προβολών είναι οι λίστες (listView), το πλέγμα (GridView), πεδία εισαγωγής κειμένου, κουμπιά, κλπ [14].
- **Πάροχος Περιεχομένου** (Content Provider) – δίνει τη δυνατότητα στις εφαρμογές να μοιράζονται ή να ανταλλάσσουν δεδομένα μιας συγκεκριμένης μορφής, η οποία ορίζεται από τον πάροχο. Παραδείγματα δεδομένων, είναι οι επαφές χρήστη και οι βάσεις δεδομένων των εφαρμογών [14].
- **Διαχειριστής Πόρων** (Resource Manager) – παρέχει πρόσβαση σε υλικό το οποίο δεν είναι σε μορφή κώδικα όπως πχ, εικόνες, αρχεία xml, πίνακες χαρακτήρων, κλπ [14].
- **Διαχειριστής Ειδοποιήσεων** (Notification Manager) – δίνει στις εφαρμογές πρόσβαση στις υπηρεσίες ειδοποιήσεων χρήστη. Τέτοιες είναι οι ειδοποιήσεις στη notification bar, τα toast μηνύματα στο κάτω μέρος της οθόνης, η δόνηση του κινητού και η ενεργοποίηση της οθόνης, κλπ [14].
- **Διαχειριστής τοποθεσίας** (Location Manager) – παρέχει πρόσβαση στις υπηρεσίες τοποθεσίας του συστήματος. Οι υπηρεσίες αυτές επιτρέπουν στις εφαρμογές να λαμβάνουν περιοδικές ενημερώσεις της γεωγραφικής θέσης της συσκευής ή να πυροδοτούν μια καθορισμένη ενέργεια όταν η συσκευή εισέρχεται μια δεδομένη γεωγραφική θέση [14],[15].
- **Διαχειριστής Δραστηριοτήτων** (Activity Manager) – διαχειρίζεται τον κύκλο ζωής των δραστηριοτήτων και παρέχει δυνατότητα πλοήγησης από δραστηριότητα σε δραστηριότητα κρατώντας αποθηκευμένη στη μνήμη τη σειρά εκτέλεσης αυτών. Στην Εικόνα 20 παρουσιάζεται ο κύκλος ζωής κάθε δραστηριότητας [14],[16].



Εικόνα 20. Κύκλος ζωής μίας δραστηριότητας (Activities Lifecycle)

1.4. Η δομή μιας εφαρμογής Android

Κάθε project λειτουργεί ως «πακέτο», όπου φυλάσσονται κώδικας, αρχεία πόροι κ.α. Το SDK προϋποθέτει από το κάθε project να έχει μία συγκεκριμένη δομή, ώστε να μπορεί να μεταγλωττίζει και να δημιουργεί το «πακέτο» - package της εφαρμογής σωστά. Σε ένα android project, όλα τα περιεχόμενα του πακέτου στο τέλος ενσωματώνονται σε ένα αρχείο με επέκταση .apk, με το οποίο γίνεται η εγκατάσταση σε συσκευή. Μερικά από τα περιεχόμενα δημιουργούνται αυτόματα από προεπιλογή, ενώ άλλα θα πρέπει να δημιουργηθούν, εφόσον απαιτείται [17]. Οι παρακάτω κατάλογοι και αρχεία συμπεριλαμβάνονται σε ένα Android project:

1.4.1 Το αρχείο Manifest.xml

Κάθε project εφαρμογής περιέχει ένα αρχείο στο οποίο καταχωρούνται οι σημαντικότερες πληροφορίες της εφαρμογής, και το αρχείο αυτό ονομάζεται AndroidManifest.xml. Πρόκειται για ένα πολύ σημαντικό αρχείο και αποτελεί κύριο συστατικό κάθε εφαρμογής. Στο αρχείο αυτό της εφαρμογής γίνονται διάφορες δηλώσεις, με τις οποίες περιγράφονται τα θεμελιώδη χαρακτηριστικά της εφαρμογής και προσδιορίζονται τα συστατικά της. Κάποιες από αυτές τις πληροφορίες είναι:

- Το όνομα του πακέτου της εφαρμογής
- Το όνομα της εφαρμογής όπως εμφανίζεται στους χρήστες
- Η έκδοση των APIs που χρησιμοποιούνται
- Ο αριθμός έκδοσης της εφαρμογής
- Οι άδειες χρήσης που ζητάει η εφαρμογή
- Όλες οι δραστηριότητες, πάροχοι περιεχομένου, υπηρεσίες, κλπ, που περιέχει και χρησιμοποιεί η εφαρμογή.

Τα δομικά μέρη του project αναφέρονται αναλυτικότερα παρακάτω. Πάντως ως ενδεικτικό παράδειγμα αναφέρεται το εξής: ένα από τα στοιχεία που περιλαμβάνονται στο Manifest είναι το <uses-sdk>, στο οποίο προσδιορίζονται οι εκδόσεις Android που είναι συμβατές με την εφαρμογή. Το στοιχείο <application> αναφέρεται στο σύνολο της εφαρμογής. Ένα στοιχείο <activity> εισάγεται ως «απόγονος» του στοιχείου <application> για κάθε κλάση Activity του project. Επίσης, ως «απόγονος» του στοιχείου <application> εισάγεται το στοιχείο <uses-library> για την χρήση εξωτερικής βιβλιοθήκης [18].

1.4.2. Ο φάκελος res

Ο φάκελος res (resources) περιέχει διάφορους υποκαταλόγους με τους πόρους (αρχεία εικόνας, κειμένου, xml layout) της εφαρμογής. Ένας από αυτούς είναι ο κατάλογος drawable που περιέχει τα αρχεία εικόνας (.jpg, .png, .gif) του Project. Στον υποκατάλογο values αποθηκεύονται όλοι οι πόροι κειμένου (μεταβλητές τύπου String) που χρησιμοποιούνται στην εφαρμογή. Ο υποκατάλογος layout περιέχει τα .xml αρχεία που ορίζουν το UI της εφαρμογής [17].

1.4.3. Ο φάκελος src

Στο φάκελο src (source) περιέχονται τα αρχεία κλάσης της Java όλων των Activities, Services, Content Providers, βοηθητικά αρχεία, κλπ. Ο φάκελος περιέχει το πακέτο ή τα πακέτα της εφαρμογής, τα οποία περιέχουν τα αρχεία Java, και αποτελεί το μοναδικό φάκελο στο project στον οποίο αποθηκεύονται τα αρχεία του κώδικα της εφαρμογής [17].

1.4.4. Δομικά μέρη ενός Project

- **Δραστηριότητες (Activities)** – είναι μια οθόνη διεπαφής χρήστη (GUI) και προβολής πληροφοριών. Πρόκειται ίσως για το κύριο δομικό στοιχείο μιας εφαρμογής. Κάθε εφαρμογή έχει τόσα Activities όσες και οι διαφορετικές οθόνες, οι οποίες εμφανίζονται στο χρήστη. Όλες οι δραστηριότητες συνεργάζονται μεταξύ τους για να δώσουν στο χρήστη μια συνολική εμπειρία χρήσης της εφαρμογής [19].
- **Προθέσεις (Intents) –** Οι δραστηριότητες επικοινωνούν και εναλλάσσουν την λειτουργία τους μέσω των Intents. Ουσιαστικά τα Intents εξασφαλίζουν την μετάβαση από τη μία δραστηριότητα σε μια άλλη και επίσης χρησιμοποιούνται για ανταλλαγή δεδομένων. Η ανταλλαγή δεδομένων, μπορεί να γίνει είτε μεταξύ των Activities μιας εφαρμογής, είτε από τη μία εφαρμογή στην άλλη. Για παράδειγμα μπορούμε μέσω ενός Intent να εκκινήσουμε έναν browser ώστε να μας ανοίξει απευθείας ένα url το οποίο έχουμε παρέχει εμείς μέσω ενός Intent [19].
- **Υπηρεσίες (Services) –** Πρόκειται για λειτουργίες της εφαρμογής οι οποίες είναι σχεδιασμένες να τρέχουν στο παρασκήνιο και να επιστρέφουν αποτελέσματά ακόμη και όταν η εφαρμογή δεν είναι στο προσκήνιο. Για παράδειγμα μια εφαρμογή media player μπορεί μέσω μιας υπηρεσίας να συνεχίσει να παίζει μουσική ακόμη και όταν το κύριο παράθυρο της εφαρμογής δεν βρίσκεται στο προσκήνιο [19].
- **Πάροχος Περιεχόμενου (Content Providers) –** Η ανταλλαγή δεδομένων από μια εφαρμογή στην άλλη μπορεί να γίνει μέσω ενός Intent, ένας πάροχος περιεχομένου, όμως, έχει πιο σύνθετη λειτουργία. Οι content providers μιας εφαρμογής διαχειρίζονται συγκεκριμένα δεδομένα της εφαρμογής, τα οποία έχει ορίσει ο προγραμματιστής κατά την κατασκευή της εφαρμογής. Συνηθισμένα δεδομένα τα οποία μοιράζονται μέσω Content Providers, είναι οι βάσεις δεδομένων SQLite μιας εφαρμογής, και οι επαφές του χρήστη [19].
- **Δέκτες Μετάδοσης (Broadcast Receivers) –** Πρόκειται για ένα είδος υπηρεσίας, η οποία αντιλαμβάνεται κάποια γεγονότα του συστήματος και αναλαμβάνει να ενημερώσει το σύστημα ή τις υπόλοιπες εφαρμογές. Ο σκοπός τους είναι διπλός καθότι μπορούν και να ενημερωθούν για κάποιο συμβάν από άλλες εφαρμογές, αλλά και να ειδοποιήσουν τις υπόλοιπες εφαρμογές και το σύστημα για κάποιο συμβάν που τις ενεργοποίησε. Δεν έχουν γραφικό περιβάλλον αλλά μπορούν να προβάλουν ειδοποίηση στο χρήστη μέσω της μπάρας ειδοποιήσεων. Συνήθως χρησιμοποιούνται ως διαμεσολαβητές μεταξύ των Activities και των Services μιας εφαρμογής [19].

1.5. Υποστήριξη ποικίλων συσκευών Android

Οι συσκευές Android κυκλοφορούν σε πολλές μορφές και μεγέθη σε όλο τον κόσμο. Το ευρύ φάσμα των τύπων των συσκευών Android, παρέχει τη δυνατότητα στον προγραμματιστή να συστήσει την εφαρμογή του σε ένα τεράστιο ακροατήριο. Για να έχει τη μεγαλύτερη δυνατή επιτυχία μια εφαρμογή Android, θα πρέπει να μπορεί να προσαρμόζεται στα διάφορα χαρακτηριστικά της κάθε συσκευής. Μερικές από τις σημαντικές διαφοροποιήσεις που πρέπει να ληφθούν υπόψη, είναι οι διαφορετικές γλώσσες, τα μεγέθη οθόνης, και οι εκδόσεις της πλατφόρμας Android [20],[21]. Η εφαρμογή που υλοποιήθηκε στο πλαίσιο της παρούσας εργασίας είναι συμβατή με όλα τα μεγέθη και αναλύσεις οθονών, με όλες τις πλατφόρμες Android από 2.1 μέχρι 4.2.2 και υποστηρίζει αγγλικά και ελληνικά. Στις παραγράφους που ακολουθούν αναλύονται αυτά τα χαρακτηριστικά της εφαρμογής.

1.5.1. Υποστήριξη διαφόρων γλωσσών

Οι πληροφορίες στις οποίες προσφέρει πρόσβαση η εφαρμογή της παρούσας εργασίας μπορούν να αξιοποιηθούν όχι μόνο από Έλληνες, αλλά και από περαστικούς στην περιοχή των Σερρών άλλης εθνικότητας. Συνεπώς κρίθηκε χρήσιμο, η εφαρμογή να υποστηρίζει και Αγγλικά εκτός από Ελληνικά. Για το λόγο αυτό μελετήθηκαν οι δυνατότητες που παρέχει το Android για την παροχή πολύγλωσσης ενημέρωσης μέσα από το ίδιο περιβάλλον.

Οι μεταβλητές τύπου String που εμφανίζονται στο UI (η κειμενική πληροφορία), συνιστάται να αποθηκεύονται σε ένα εξωτερικό αρχείο strings.xml στο φάκελο res/values του project. Η υποστήριξη άλλων γλωσσών, γίνεται με δημιουργία πρόσθετων φακέλων values μέσα στο φάκελο res, με ονομασία values-«ISO country code» [22].

Στην εφαρμογή της παρούσας εργασίας υπάρχει ένας φάκελος values με τις προεπιλεγμένες τιμές των String στην Αγγλική γλώσσα, και ένας values-el με τις τιμές στην Ελληνική. Έτσι, αν η γλώσσα της συσκευής είναι η Ελληνική, η εφαρμογή αυτόματα θα φορτώσει τις τιμές από τον κατάλογο values-el. Αν η γλώσσα της συσκευής είναι οποιαδήποτε άλλη, θα φορτώσει τις τιμές από τον κατάλογο με τις εξ΄ ορισμού τιμές (default values). Η δομή των καταλόγων values, για την παρούσα εργασία:

```
SerresGuide (project-name)/
    res/
       values/
           strings.xml
       values-el/
           strings.xml
```

Η ίδια μέθοδος μπορεί να εφαρμοστεί για μετάφραση κάθε είδους πόρου (resources), όπως π.χ. στον φάκελο drawable [21].

Υπόδειγμα του αρχείου values/strings.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">Serres Guide</string>
    <string name="Pharmacies">Pharmacies</string>
    <string name="GasStations">Gas Stations</string>
</resources>
```

Υπόδειγμα του αρχείου values-el/strings.xml:

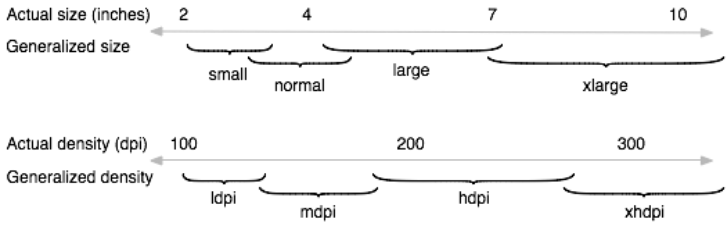
```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">Οδηγός Σερρών</string>
    <string name="Pharmacies">Φαρμακεία</string>
    <string name="GasStations">Πρατήρια Καυσίμων</string>
</resources>
```

1.5.2. Υποστήριξη διαφόρων οθονών

Οι συσκευές Android ποικίλουν από μικρά κινητά τηλέφωνα έως μεγάλα TV set. Γι' αυτό είναι σημαντικό οι εφαρμογές να σχεδιάζονται έτσι ώστε να είναι συμβατές με όλα τα μεγέθη και αναλύσεις οθονών, ώστε να είναι διαθέσιμες σε περισσότερους χρήστες [23].

Το User Interface της εφαρμογής που δημιουργήθηκε στο πλαίσιο της παρούσας εργασίας, σχεδιάστηκε ώστε να προσαρμόζεται στο μέγεθος και την ανάλυση κάθε οθόνης, για προσανατολισμό οριζόντιο και κατακόρυφο.

Το Android κατηγοριοποιεί τις οθόνες συσκευών χρησιμοποιώντας δύο γενικές ιδιότητες: το μέγεθος και την ανάλυση ή πυκνότητα (density). Το Android χαρτογραφεί τα πραγματικά μεγέθη και πυκνότητες σε γενικευμένες. Υπάρχουν τέσσερα γενικευμένα μεγέθη (Εικόνα 21): μικρό (small), κανονικό (normal), μεγάλο (large), πολύ μεγάλο (xlarge) και τέσσερις γενικευμένες πυκνότητες: χαμηλή (ldpi), μεσαίου (mdpi), υψηλή (hdpi), πολύ υψηλής (xhdpi) [24],[25].



Εικόνα 21. Χαρτογράφηση πραγματικών μεγεθών και πυκνοτήτων

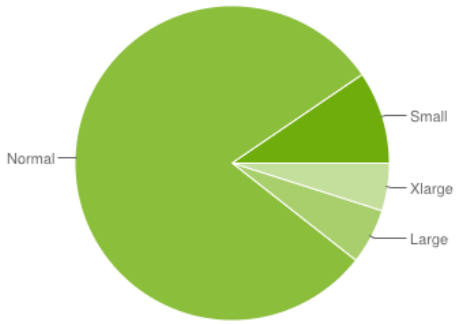
Ο προγραμματιστής κατά το σχεδιασμό του User Interface καλείται να δώσει περισσότερη βαρύτητα στη συμβατότητα με τις συσκευές που έχουν μεγαλύτερα ποσοστά διανομής [26]. Ο Πίνακας Ι συγκεντρώνει τα ποσοστά υποστήριξης των αναλύσεων οθόνης που παρέχουν οι συσκευές Android σύμφωνα με επίσημα στοιχεία μέχρι τον Απρίλιο του 2013. Α-

ντίστοιχα στις εικόνες που ακολουθούν (Εικόνα 22, Εικόνα 23) εμφανίζονται με γραφικό τρόπο τα ποσοστά υποστήριξης μεγεθών οθόνης και αναλύσεων.

Πίνακας I. Ποσοστό υποστήριξης των χαρακτηριστικών οθονών από τις συσκευές

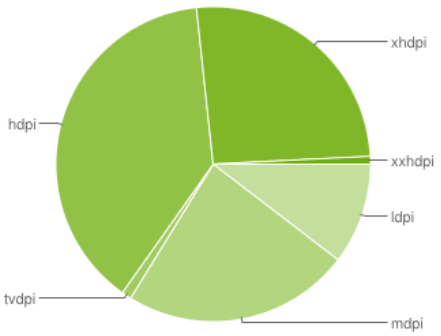
	ldpi	mdpi	tvdpi	hdpi	xhdpi	xxhdpi	**Total**
Small	9.5%						9.5%
Normal	0.1%	16.1%		37.9%	25.0%	0.8%	79.9%
Large	0.7%	2.7%	1.0%	0.5%	0.8%		5.7%
Xlarge	0.1%	4.6%		0.1%	0.1%		4.9%
Total	10.4%	23.4%	1.0%	38.5%	25.9%	0.8%	

Τελευταία ενημέρωση: 2 Απρ 13. Εκδόσεις με λιγότερο από 0,1% της διανομής δεν απεικονίζονται.



Τελευταία ενημέρωση: 2 Απρ 13. Εκδόσεις με λιγότερο από 0,1% της διανομής δεν απεικονίζονται.

Εικόνα 22. Υποστήριξη των μεγεθών οθονών από τις Android συσκευές



Τελευταία ενημέρωση: 2 Απρ 13. Εκδόσεις με λιγότερο από 0,1% της διανομής δεν απεικονίζονται.

Εικόνα 23. Υποστήριξη των πυκνοτήτων οθονών από τις Android συσκευές.

Στις παραγράφους που ακολουθούν περιγράφονται οι τεχνικές σχεδίασης που προτείνονται και έχουν υιοθετηθεί στην υλοποίηση της εφαρμογής της παρούσας εργασίας.

Υποστήριξη διαφόρων πυκνοτήτων οθόνης:

- Χρήση εικονοστοιχείων ανεξάρτητα της πυκνότητας (dp, sp): Κατά το σχεδιασμό του UI, πρέπει να αποφεύγεται η χρήση πραγματικών pixel για τον ορισμό αποστάσεων ή μεγεθών. Λόγω των διαφόρων πυκνοτήτων που υποστηρίζονται από τις οθόνες, ο ίδιος αριθμός pixels μπορεί να αντιστοιχεί σε διαφορετική φυσική απόσταση σε κάθε οθόνη. Αντί αυτού συνιστάται η χρήση των μονάδων dp και sp. Dp είναι pixel ανεξάρτητα πυκνότητας, που αντιστοιχεί στο φυσικό μέγεθος ενός εικονοστοιχείου σε 160 dpi. Sp είναι κατά βάση ή ίδια μονάδα με το dp, αλλά κλιμακώνεται κατά προτιμώμενο μέγεθος του κειμένου του χρήστη και χρησιμοποιείται μόνο για καθορισμό μεγέθους γραμματοσειράς κειμένου [27]. Π.χ.:

```
<Button android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="@string/clickme"
                android:textSize="20sp"
                android:layout_marginTop="20dp" />
```

- Παροχή εναλλακτικών εικόνων: Η παροχή προσαρμοσμένων εικόνων για κάθε γενικευμένη πυκνότητα οδηγεί σε καλύτερη ποιότητα και επιδόσεις γραφικών. Η δημιουργία των εικόνων για κάθε πυκνότητα, γίνεται σύμφωνα με την ακόλουθη κλίμακα μεγέθους:

```
xhdpi: 2.0
hdpi: 1.5
mdpi: 1.0 (baseline)
ldpi: 0.75
```

Αυτό σημαίνει ότι αν έχει δημιουργηθεί μια εικόνα 200x200 για xhdpi συσκευές, θα πρέπει να παραχθεί η ίδια εικόνα σε 150x150 για hdpi, σε 100x100 για mdpi και τέλος σε 75x75 για ldpi συσκευές. Στη συνέχεια, αυτές οι εικόνες θα πρέπει να τοποθετηθούν στον κατάλληλο υποφάκελο του res, όπως παρουσιάζεται παρακάτω, και το σύστημα θα επιλέξει αυτόματα την κατάλληλη ανάλογα με την πυκνότητα της οθόνης, στην οποία εκτελείται η εφαρμογή [27],[24]:

```
SerresGuide (project-name)/
  res/
    drawable-xhdpi/
        image.png
    drawable-hdpi/
        image.png
    drawable-mdpi/
        image.png
    drawable-ldpi/
        image.png
```

Υποστήριξη διαφόρων μεγεθών οθόνης:

- Χρήση των «wrap-content» και «match-parent»: Για να διασφαλιστεί ότι το UI (layouts) είναι ευέλικτο και προσαρμόζεται σε διαφορετικά μεγέθη οθόνης, θα πρέπει να γίνει χρήση των "wrap_content" και "match_parent" για το πλάτος και το ύψος ορισμένων στοιχείων (views). Με τη χρήση του "wrap_content", το πλάτος ή το ύψος της προβολής ορίζεται στο ελάχιστο μέγεθος που απαιτείται για να χωρέσει το περιεχόμενο σε αυτό το στοιχείο (view). Το "match_parent" (επίσης γνωστό ως "fill_parent" πριν το επίπεδο API 8) αναγκάζει το στοιχείο (view) να επεκταθεί ώστε να ταιριάζει με το μέγεθος του στοιχείου γονέα (parent view) [28]. Π.χ.:

```
<TextView
               android:id="@+id/label"
               android:layout_width="match_parent"
               android:layout_height="wrap_content" />
```

- Χρήση του RelativeLayout: Αρκετά πολύπλοκα layouts μπορούν να παραχθούν με την χρήση του LinearLayout και των μεγεθών "wrap-content" και "match-parent". Ωστόσο, το LinearLayout δεν επιτρέπει τον ακριβή έλεγχο των χωρικών σχέσεων των views που περιέχει. Τα views σε ένα LinearLayout απλά στοιχίζονται το ένα δίπλα στο άλλο. Ο προσανατολισμός των views σε κάποια άλλη παραλλαγή εκτός από ευθεία γραμμή, γίνεται με χρήση RelativeLayouts. Στο RelativeLayout η θέση ενός child view καθορίζεται σε σχέση με τη θέση των άλλων child views. Για παράδειγμα μπορεί να ένα view α μπορεί να εφάπτεται στην αριστερή πλευρά του view γονέα και ένα view β να εφάπτεται στη δεξιά πλευρά του view γονέα [28]. Π.χ.:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:id="@+id/label"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Type here:"/>
    <EditText
        android:id="@+id/entry"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/label"/>
    <Button
        android:id="@+id/ok"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/entry"
        android:layout_alignParentRight="true"
        android:layout_marginLeft="10dp"
        android:text="OK" />
    <Button
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_toLeftOf="@id/ok"
        android:layout_alignTop="@id/ok"
        android:text="Cancel" />
</RelativeLayout>
```

Εικόνα 24. Εμφάνιση του layout σε μικρή και σε μεγάλη οθόνη αντίστοιχα

Σημειώνεται ότι αν και το μέγεθος των views έχει αλλάξει, οι χωρικές σχέσεις τους έχουν διατηρηθεί, όπως ορίζεται από τις παραμέτρους του RelativeLayout [28].

Άλλες τεχνικές σχεδίασης που προτείνονται για υποστήριξη περισσότερων μεγεθών οθονών:

- Χρήση διαφορετικών layout: Ο προγραμματιστής για κάθε μέγεθος οθόνης που επιθυμεί να υποστηρίζει η εφαρμογή του μπορεί να δημιουργήσει και ένα ξεχωριστό layout xml αρχείο. Κάθε layout πρέπει να αποθηκεύεται σε κατάλληλο κατάλογο πόρων, με ονομασία layout-<χαρακτηριστικό οθόνης>. Για παράδειγμα, ένα layout για μεγάλες οθόνες θα πρέπει να αποθηκευτεί στον κατάλογο res/layout-large [24]. Αντίστοιχα μπορούν παραχθούν layout αρχεία για οθόνες με κατακόρυφο/οριζόντιο προσανατολισμό, με συγκεκριμένο πλάτος (σε dp) ή με συνδυασμούς των προηγουμένων [24],[28]. Π.χ.:

```
MyProject/
    res/
        layout/                     : default (portrait)
            main.xml
        layout-sw600dp/             : screen size 600 dp
            main.xml
        layout-land/                : landscape
            main.xml
        layout-large/               : large (portrait)
            main.xml
        layout-large-land/          : large and landscape
            main.xml
```

1.5.3. Υποστήριξη διαφόρων εκδόσεων Android

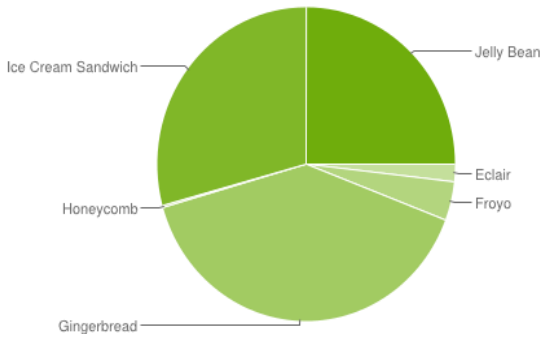
Ενώ οι τελευταίες εκδόσεις του Android συχνά παρέχουν εξαιρετικά APIs, θα πρέπει να συνεχίσουν να υποστηρίζονται παλαιότερες εκδόσεις του Android μέχρι να λάβουν την ενημέρωση περισσότερες συσκευές [29]. Έτσι, η *εφαρμογή που υλοποιήθηκε στο πλαίσιο αυτής της εργασίας είναι συμβατή με όλες της εκδόσεις Android από 2.1 μέχρι τη νεότερη 4.2.2.*

Ο προγραμματιστής κατά το σχεδιασμό εφαρμογής καλείται να δώσει περισσότερη βαρύτητα στη συμβατότητα με τις εκδόσεις Android που έχουν μεγαλύτερα ποσοστά διανομής (Πίνακας ΙΙ) [26].

Πίνακας ΙΙ. Εκδόσεις Android και υποστήριξη από Android συσκευές

Έκδοση	Κωδική ονομασία	API level	Διανομή/διάχυση
1.6	Donut	4	0.1%
2.1	Eclair	7	1.7%
2.2	Froyo	8	4.0%
2.3-2.3.2	Gingerbread	9	0.1%
2.3.3-2.3.7	Gingerbread	10	39.7%
3.2	Honeycomb	13	0.2%
4.0.3-4.0.4	Ice Cream Sandwich	15	29.3%
4.1.x	Jelly Bean	16	23.0%
4.2.x	Jelly Bean	17	2.0%

Τελευταία ενημέρωση: 2 Απρ 13. Εκδόσεις με λιγότερο από 0,1% της διανομής δεν απεικονίζονται.



Τελευταία ενημέρωση: 2 Απρ 13. Εκδόσεις με λιγότερο από 0,1% της διανομής δεν απεικονίζονται.

Εικόνα 25. Εκδόσεις Android και υποστήριξη από Android συσκευές

Οι παρακάτω τεχνικές βοηθούν την εφαρμογή να επωφεληθεί από τα τελευταία APIs, αλλά και να υποστηρίζει παλαιότερες εκδόσεις Android:

- Καθορισμός Minimum και Target επιπέδου API: Το αρχείο AndroidManifest.xml περιγράφει λεπτομέρειες για την εφαρμογή, όπως τις εκδόσεις του Android που υποστηρίζει. Συγκεκριμένα, τα χαρακτηριστικά minSdkVersion και targetSdkVersion του στοιχείου <uses-SDK> προσδιορίζουν το χαμηλότερο επίπεδο

API με το οποίο η εφαρμογή είναι συμβατή και το υψηλότερο επίπεδο API κατά το οποίο έχει σχεδιαστεί και δοκιμαστεί [29]. Π.χ.:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
... >
<uses-sdk android:minSdkVersion="4" android:targetSdkVersion="15" />
...
</manifest>
```

Καθώς νέες εκδόσεις γίνονται διαθέσιμες, η μορφή κάποιων χαρακτηριστικών αλλάζει. Για να επωφεληθεί η εφαρμογή από αυτές τις αλλαγές και να εξασφαλιστεί ότι η εφαρμογή ταιριάζει με τη μορφή που υποστηρίζει κάθε συσκευή, συνιστάται ο ορισμός της τιμής targetSdkVersion στην τελευταία διαθέσιμη έκδοση του Android [29]. Στην παρούσα εργασία χρησιμοποιήθηκε minSdkVersion "7" και targetSdkVersion "17".

- Έλεγχος της Έκδοσης Συστήματος κατά το χρόνο εκτέλεσης: Το Android παρέχει ένα μοναδικό κωδικό για κάθε έκδοση Android στην κλάση Build constants. Αυτοί οι κωδικοί μπορούν να χρησιμοποιηθούν σε συνθήκες για να διασφαλιστεί ότι ο κώδικας που εξαρτάται από υψηλά επίπεδα API θα εκτελείται μόνο όταν τα API είναι διαθέσιμα στο σύστημα [29]. Π.χ.:

```
private void setUpActionBar() {
  // Make sure we're running on Honeycomb or higher
  // to use ActionBar APIs
  if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.HONEYCOMB) {
    ActionBar actionBar = getActionBar();
    actionBar.setDisplayHomeAsUpEnabled(true);
  }
}
```

- Χρήση των στυλ και θεμάτων της πλατφόρμας: Το Android παρέχει θέματα που δίνουν την εμφάνιση και την αίσθηση του υποκείμενου λειτουργικού συστήματος. Αυτά τα θέματα μπορούν να εφαρμοστούν στο αρχείο Manifest μιας εφαρμογής, ώστε η εφαρμογή να ακολουθεί φυσικά την τελευταία εμφάνιση και αίσθηση του Android με κάθε νέα έκδοση. Θέμα μπορεί να οριστεί σε ολόκληρη την εφαρμογή, αλλά και σε μεμονωμένα activities [29]. Π.χ.:

```
<activity
android:theme="@android:style/Theme.Translucent">
```

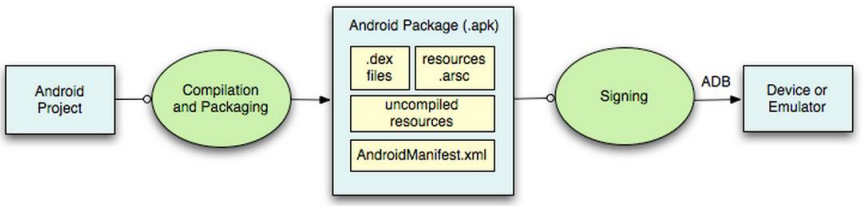
1.6. Το πιστοποιητικό των εφαρμογών Android

Το σύστημα Android απαιτεί όλες οι εγκατεστημένες εφαρμογές να είναι ψηφιακά υπογεγραμμένες με ένα πιστοποιητικό του οποίου το ιδιωτικό κλειδί κατέχει ο προγραμματιστής της εφαρμογής. Το σύστημα Android χρησιμοποιεί το πιστοποιητικό ως μέσο προσδιορισμού του συντάκτη της εφαρμογής και τη δημιουργία σχέσεων εμπιστοσύνης μεταξύ των εφαρ-

μογών. Το πιστοποιητικό δεν χρησιμοποιείται για να ελέγχει ποιες εφαρμογές ο χρήστης μπορεί να εγκαταστήσει [30].

- Για τον έλεγχο και την διόρθωση της εφαρμογής, τα εργαλεία Build υπογράφουν την εφαρμογή με ένα ειδικό debug κλειδί που δημιουργείται από το Android SDK build tools [30]
- Όταν η εφαρμογή είναι έτοιμη να κυκλοφορήσει στους χρήστες, θα πρέπει να υπογραφθεί με ένα κατάλληλο ιδιωτικό κλειδί. Δεν μπορεί να δημοσιευθεί μια εφαρμογή που έχει υπογραφεί με το debug κλειδί που δημιουργούνται από τα εργαλεία SDK [30]
- Όλες οι εφαρμογές πρέπει να υπογράφονται. Το σύστημα δεν θα εγκαταστήσει μια εφαρμογή σε έναν εξομοιωτή ή μια συσκευή, εάν δεν έχει υπογραφεί [30]
- Το πιστοποιητικό δεν πρέπει να υπογραφεί από μια αρχή πιστοποίησης. Είναι απολύτως επιτρεπτή και τυπική, η χρήση αυτό-υπογραμμένων πιστοποιητικών από τον προγραμματιστή, για τις Android εφαρμογές [30]
- Το σύστημα ελέγχει την ημερομηνία λήξης ενός πιστοποιητικού μόνο κατά τη διάρκεια της εγκατάστασης. Αν το πιστοποιητικό μιας εφαρμογής λήξει μετά την εγκατάσταση της εφαρμογής, η εφαρμογή θα συνεχίσει να λειτουργεί κανονικά [30]

Το παρακάτω διάγραμμα απεικονίζει τα στοιχεία που εμπλέκονται στην κατασκευή και τη λειτουργία μιας εφαρμογής [31]:



Εικόνα 26. Building and running an android application

1.7. Επιλογές αποθήκευσης των δεδομένων

Το Android παρέχει αρκετές επιλογές για την αποθήκευση των μόνιμων δεδομένων των εφαρμογών. Η λύση που επιλέγεται εξαρτάται από τις συγκεκριμένες ανάγκες της κάθε εφαρμογής [32]:

- Shared Preferences - Αποθήκευση ιδιωτικών πρωτογενών (primitive) δεδομένων σε ζεύγη «κλειδί-τιμή».
- Εσωτερική αποθήκευση - Αποθήκευση ιδιωτικών δεδομένων στη μνήμη της συσκευής.
- Εξωτερικά μέσα αποθήκευσης - Αποθήκευση δημόσιων δεδομένων στην κοινόχρηστη εξωτερική μνήμη.

- SQLite βάσεις δεδομένων - Αποθήκευση δομημένων δεδομένων σε ιδιωτική βάση δεδομένων.
- Σύνδεση δικτύου - Αποθήκευση δεδομένα στο διαδίκτυο με χρήση διακομιστή δικτύου (server).

Στην εφαρμογή της παρούσας εργασίας χρησιμοποιήθηκε βάση δεδομένων SQLite και Shared Preferences.

1.7.1. Βάση δεδομένων SQLite

Το Android παρέχει πλήρη υποστήριξη για SQLite βάσεις δεδομένων. Κάθε βάση δεδομένων που δημιουργείται σε μία εφαρμογή είναι προσβάσιμη με χρήση του ονόματος της από οποιαδήποτε κλάση της εφαρμογής. Δεν είναι όμως προσβάσιμη από στοιχεία άλλων εφαρμογών [32].

Η συνιστώμενη μέθοδος δημιουργίας μια νέας βάσης δεδομένων SQLite είναι με τη δημιουργία μιας υποκλάσης της με επέκταση (extends) SQLiteOpenHelper. Η κλάση SQLiteOpenHelper είναι μια βοηθητική κλάση για τη διαχείριση της δημιουργίας βάσης δεδομένων και τη διαχείριση των εκδόσεων της. Στη συνάρτηση onCreate() της υποκλάσης που δημιουργήθηκε, γίνεται η δημιουργία των πινάκων με SQLite εντολές [32],[33]. Η βάση δεδομένων της παρούσας εργασίας έχει δημιουργηθεί με αυτή τη μέθοδο.

Η συνάρτηση getWritableDatabase() επιστρέφει ένα αντικείμενο που αντιπροσωπεύει την SQLiteDatabase βάση δεδομένων και παρέχει μεθόδους για SQLite λειτουργίες. Αντίστοιχη είναι η λειτουργία της συνάρτησης getReadableDatabase(), αλλά παρέχει μόνο λειτουργίες ανάγνωσης [32].

Κάθε ερώτημα SQLite επιστρέφει ένα δείκτη που δείχνει σε όλες τις σειρές που επεστράφησαν από το ερώτημα. Ο δείκτης αυτός είναι ο μόνος μηχανισμός για περιήγηση και ανάγνωση των γραμμών – στηλών των αποτελεσμάτων σε ένα ερώτημα βάσης δεδομένων (query) [32].

1.7.2. Shared Preferences

Η κλάση SharedPreferences παρέχει ένα γενικό πλαίσιο που επιτρέπει την αποθήκευση και να ανάκτηση πρωτογενών (primitive) δεδομένων σε ζεύγη «κλειδί-τιμή». Πρωτογενή (primitive) στοιχεία είναι οι μεταβλητές τύπου boolean, float, int, long, και string. Τα στοιχεία αυτά θα συνεχίσουν να υπάρχουν ακόμα και αν η εφαρμογή τερματισθεί [32]. Τα δεδομένα αυτά βρίσκονται καταχωρημένα σε ένα xml αρχείο. Το αρχείο αυτό δημιουργείται αυτόματα από το σύστημα κατά την εισαγωγή στοιχείων σε αυτό. Ο φάκελο της εφαρμογής (που δημιουργείται στη συσκευή κατά την εγκατάστασή της) περιέχει έναν φάκελο «shared_prefs», ο οποίος περιέχει όλα τα xml αρχεία που έχουν δημιουργηθεί από Shared Preferences.

Η ανάκτηση/λήψη ενός αντικείμενου SharedPreferences γίνεται με τη συνάρτηση getSharedPreferences(String name, int mode). Η εγγραφή τιμών γίνεται με τα παρακάτω βήματα:

1. Κλήση της συνάρτησης edit() για λήψη ενός αντικειμένου SharedPreferences.Editor.
2. Προσθήκη των τιμών με μεθόδους όπως putBoolean() και putString().
3. Καταχώρηση των αλλαγών με την εντολή commit().

Η εγγραφή θα γίνει στο xml αρχείο του οποίου το όνομα δόθηκε κατά την δημιουργία του SharedPreferences.Editor αντικειμένου. Η ανάγνωση τιμών γίνεται με εντολές όπως getBoolean() και getString() [32].

2. Απαραίτητα εργαλεία για ανάπτυξη εφαρμογών

Η ανάπτυξη εφαρμογών Android απαιτεί τη χρήση μίας πλατφόρμας IDE (Integrated Development Environment) με ενσωματωμένο το ADT (Android Developer Tools) Plugin. Υπάρχει δυνατότητα επιλογής ανάμεσα α) σε εγκατάσταση του ADT Bundle[7], το οποίο περιλαμβάνει την τελευταία έκδοση του περιβάλλοντος ανάπτυξης Eclipse[8] με ενσωματωμένο το ADT Plugin (προτεινόμενο), β) σε ενσωμάτωση του ADT Plugin σε μία ήδη εγκατεστημένη IDE πλατφόρμα [34]. Αρχικά απαιτείται η εγκατάσταση της πλατφόρμας Java JDK[9]. Προσοχή πρέπει να δίνεται σε ιδιαιτερότητες της εγκατάστασης σε πλατφόρμες x64 (64-bit) όπου απαιτείται η εγκατάσταση και της 32-bit έκδοσης της JRE. Στη συνέχεια μπορεί να γίνει εγκατάσταση του ADT Bundle ή ενσωμάτωση του ADT Plugin σε ήδη εγκατεστημένη πλατφόρμα IDE [34]. Ένα πρακτικό παράδειγμα με οδηγίες για ανάπτυξη εφαρμογής Android, με δυνατότητα εμφάνισης διαδραστικού χάρτη δίνεται στο Παράρτημα Β.

2.1. Εγκατάσταση του ADT Bundle

Η εγκατάσταση της πλατφόρμας ανάπτυξης εφαρμογών Android, συνιστάται να γίνει με τη χρήση του πακέτου ADT Bundle. Με το ADT Bundle γίνεται πολύ πιο εύκολη η εγκατάσταση, καθώς περιέχει όλα τα παρακάτω απαραίτητα για την ανάπτυξη εφαρμογών:

- Eclipse IDE + ADT plugin
- Android SDK Tools

---

[7] *Το πλήρες πακέτο ανάπτυξης εφαρμογών Android είναι διαθέσιμο στη διεύθυνση http://developer.android.com/sdk/index.html*

[8] *Το Eclipse IDE μπορεί να μεταφορτωθεί από τη διεύθυνση http://www.eclipse.org/downloads/, ή απευθείας από τη διεύθυνση μεταφόρτωσης του περιβάλλοντος ανάπτυξης Android ADK*

[9] *Λήψη μπορεί να γίνει από το επίσημο site της Oracle: http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html. Οδηγίες εγκατάστασης δίδονται στο: http://docs.oracle.com/javase/7/docs/webnotes/install/index.html*

- Android Platform-tools
- Την τελευταία έκδοση του Android
- Την τελευταία έκδοση του συστήματος Android για τον προσομοιωτή (emulator) κινητής συσκευής [34]

Οδηγίες εγκατάστασης:

1. Το κατάλληλο πακέτο ADT Bundle μπορεί να ληφθεί από το site: http://developer.android.com/sdk/index.html [35]
2. Απαιτείται η αποσυμπίεση του αρχείου, με ονομασία adt-bundle-<os_platform>.zip και η αποθήκευση του περιεχομένου του σε κατάλληλη τοποθεσία, π.χ. σε έναν φάκελο "Development" στον κεντρικό κατάλογο [35]
3. Στη συνέχεια απαιτείται το άνοιγμα του φάκελου adt-bundle-<os_platform>/eclipse/ και η εκκίνηση του Eclipse [35]

Μετά από τα τρία αυτά βήματα, η εγκατάσταση των απαραίτητων εργαλείων για την ανάπτυξη εφαρμογών πραγματοποιήθηκε. Η λήψη άλλων εκδόσεων android ή πακέτων όπως η βιβλιοθήκη Google Play In-app Billing, μπορεί να πραγματοποιηθεί από το μενού SDK Manager του Eclipse IDE [35]. Απαιτείται προσοχή ώστε τα περιεχόμενα του φακέλου adt-bundle-<os_platform> να μη μετακινηθούν, γιατί δεν θα μπορούν να εντοπιστούν από το ADT [35].

2.2. Ενσωμάτωση ADT Plugin σε ήδη εγκατεστημένο IDE

Η ενσωμάτωση του ADT Plugin σε μια ήδη εγκατεστημένη έκδοση του Eclipse ή άλλης πλατφόρμας IDE, γίνεται με εγκατάσταση του πακέτου SDK Tools Only και όχι του ADT Bundle [36]. Ακολουθούν συνοπτικές οδηγίες για την επιτυχή εγκατάσταση για τις βασικές υπολογιστικές πλατφόρμες,

2.2.1. Οδηγίες εγκατάστασης για Windows

Για την επιτυχή εγκατάσταση σε περιβάλλον Microsoft Windows απαιτούνται τα παρακάτω βήματα [36]:

1. Το κατάλληλο πακέτου SDK Tools Only λαμβάνεται από το site: http://developer.android.com/sdk/index.html
2. Πραγματοποιείται εγκατάσταση του ληφθέντος αρχείου
3. Σημειώνεται η διαδρομή όπου αποθηκεύτηκε το SDK, γιατί ζητείται παρακάτω
4. Όταν η εγκατάσταση τελειώσει γίνεται πρόταση για εκκίνηση του SDK Manager. Αν γίνεται χρήση του Eclipse, δεν πρέπει να γίνει εκκίνηση του SDK Manager, αλλά εκκίνηση του Eclipse και ακολουθούνται οι οδηγίες για εγκατάσταση του ADT Plugin στο

Eclipse. Αν γίνεται χρήση άλλης πλατφόρμας IDE, μπορεί να γίνει απ' ευθείας εκκίνηση του SDK Manager

2.2.2. Οδηγίες εγκατάστασης για Mac

Για την επιτυχή εγκατάσταση σε περιβάλλον Mac OS απαιτούνται τα παρακάτω βήματα [36]:

1. Το κατάλληλο πακέτο SDK Tools Only λαμβάνεται από το site: http://developer.android.com/sdk/index.html
2. Απαιτείται η αποσυμπίεση του zip αρχείου που έχει ληφθεί. Από προεπιλογή το περιεχόμενο αποσυμπιέζεται σε φάκελο με όνομα android-sdk-mac_x86 ή αντίστοιχο ανάλογα με την έκδοση. Πρέπει να γίνει μετακίνηση του φάκελο αυτού σε κατάλληλη τοποθεσία, π.χ. σε έναν φάκελο "Development" στον κεντρικό κατάλογο
3. Σημειώνεται η διαδρομή όπου αποθηκεύτηκε το SDK, γιατί ζητείται παρακάτω
4. Στη συνέχεια πρέπει να γίνει εκκίνηση του Eclipse και ακολουθούνται οι οδηγίες για εγκατάσταση του ADT Plugin στο Eclipse. Αν γίνεται χρήση άλλης πλατφόρμας IDE, μπορεί να πραγματοποιηθεί απ' ευθείας εκκίνηση του SDK Manager

2.2.3. Οδηγίες εγκατάστασης για Linux

Για την επιτυχή εγκατάσταση σε περιβάλλον Linux απαιτούνται τα παρακάτω βήματα [36]:

1. Το κατάλληλο πακέτου SDK Tools Only λαμβάνεται από το site: http://developer.android.com/sdk/index.html
2. Απαιτείται η αποσυμπίεση του tgz αρχείου που έχει ληφθεί. Από προεπιλογή το περιεχόμενο αποσυμπιέζεται σε φάκελο με όνομα android-sdk-linux_x86, ή αντίστοιχο ανάλογα με την έκδοση. Πρέπει να γίνει μετακίνηση του φάκελου αυτού σε κατάλληλη τοποθεσία, π.χ. σε έναν φάκελο "Development" στον κεντρικό κατάλογο
3. Σημειώνεται η διαδρομή όπου αποθηκεύτηκε το SDK, γιατί ζητείται παρακάτω
4. Στη συνέχεια πρέπει να γίνει εκκίνηση του Eclipse και ακολουθούνται οι οδηγίες για εγκατάσταση του ADT Plugin στο Eclipse. Αν γίνεται χρήση άλλης πλατφόρμας IDE, μπορεί να πραγματοποιηθεί απ' ευθείας εκκίνηση του SDK Manager.

2.2.4. Οδηγίες για εγκατάσταση του ADT Plugin στο Eclipse

Για την εγκατάσταση του ADT Plugin στο περιβάλλον ανάπτυξης Eclipse ακολουθούνται τα παρακάτω βήματα [37]:

1. Γίνεται επιλογή του μενού του Eclipse Help > Install New Software
2. Πραγματοποιείται κλικ στο κουμπί Add... στην γωνία πάνω δεξιά
3. Στο παράθυρο που εμφανίζετε, εισάγεται στο πεδίο Name: "ADT Plugin" και στο πεδίο Location: "https://dl-ssl.google.com/android/eclipse/"
4. Γίνεται κλικ στο Ok. Αν εμφανιστεί μήνυμα σφάλματος, προτείνεται η χρήση "http" πρωτοκόλλου στο Location URL, αντί για "https"
5. Πραγματοποιείται ανάγνωση και αποδοχή των Όρων άδειας χρήσης και επιλογή Finish
6. Αν εμφανιστεί προειδοποίηση ασφαλείας, με μήνυμα ότι η αυθεντικότητα ή η εγκυρότητα του λογισμικού δεν μπορεί να προσδιοριστεί, επιλέγεται το Ok
7. Όταν ολοκληρωθεί η εγκατάσταση, πρέπει να γίνει επανεκκίνηση του Eclipse
8. Γίνεται επιλογή "Use existing SDKs", στο παράθυρο "Welcome to Android Development" του Eclipse
9. Πραγματοποιείται κλικ στο κουμπί Browse, επιλογή της θέσης του Android SDK (από το βήμα 3) και κλικ στο Next
10. Συνιστάται η λήψη της τελευταίας έκδοσης Android για τον Emulator από μενού SDK Manager του Eclipse

Η εγκατάσταση των απαραίτητων εργαλείων για την ανάπτυξη εφαρμογών έχει πραγματοποιηθεί. Η λήψη άλλων εκδόσεων Android ή πακέτων όπως η βιβλιοθήκη Google Play In-app Billing, μπορεί να τα πραγματοποιηθεί από το μενού SDK Manager του Eclipse [37].

3. Η βιβλιοθήκη Google Maps API

Η εξωτερική βιβλιοθήκη Google Maps API επιτρέπει την εύκολη ενσωμάτωση διαδραστικών χαρτών της Google στις εφαρμογές Android, μέσω της κλάσης MapView. Η κλάση αυτή παρέχει ενσωματωμένη λήψη, απόδοση, και προσωρινή αποθήκευση χαρτών, καθώς και μια ποικιλία από επιλογές εμφάνισης και ελέγχου. Αποτελεί ένα περίβλημα γύρω από το Google Maps API που επιτρέπει στις εφαρμογές Android να ζητούν και να χειρίζονται δεδομένα Google Maps μέσω μεθόδων της κλάσης, και επιτρέπει τη χρήση δεδομένων των χαρτών μέσω των χαρακτηριστικών ενός αντικειμένου MapView, τύπου View [38].

Η πρόσβαση στα Google Maps δεδομένα παρέχεται σε μια εφαρμογή με χρήση του Google Maps API Key. Η απόκτηση του Google Maps API Key γίνεται με εγγραφή του πιστοποιητικού της εφαρμογής στην υπηρεσία Google Maps. Αυτό ισχύει είτε πρόκειται για εφαρμογή που

ακόμα αναπτύσσεται σε προσομοιωτή, είτε για εφαρμογή ήδη έτοιμη για εγκατάσταση σε συσκευές Android [38].

3.1 Απόκτηση Google Maps API Key

Η εγγραφή για το Google Maps API v1 Android Key έχει δύο μέρη [38]:

- Καταχώριση του MD5 fingerprint του πιστοποιητικού (keystore) που χρησιμοποιεί η εφαρμογή, για την απόκτηση του Google Maps API v1 Key
- Αναφορά του Google Maps API Android v1 Key σε κάθε αντικείμενο MapView, είτε αυτό δηλώνεται σε XML είτε αρχικοποιείται απευθείας από τον κώδικα

Οδηγίες απόκτησης του Google Maps API Key του SDK Debug πιστοποιητικού [38]:

1. Απαιτείται εκκίνηση της κονσόλας και άνοιγμα του φάκελου ".android". Η τοποθεσία του φακέλου εμφανίζεται και από το Eclipse από το μενού Windows > Prefs > Android > Build
2. Πραγματοποιείται εκτέλεση της παρακάτω Keytool εντολής, για λήψη του MD5 fingerprint debug πιστοποιητικού:

```
keytool -list -alias androiddebugkey -keystore debug.keystore -
storepass android -keypass android
```

3. Ακολουθούνται οι οδηγίες απόκτησης του Google Maps API Key

Οδηγίες απόκτησης του Google Maps API Key αυτο-υπογραμμένου πιστοποιητικού [38]:

1. Απαιτείται εκκίνηση της κονσόλας και εκτέλεση της παρακάτω Keytool εντολής, για λήψη του MD5 fingerprint του επιλεγμένου "my-release-key" πιστοποιητικού:

```
keytool -list -alias "alias_name" -keystore "my-release-
key".keystore
```

2. Ακολουθούνται οι οδηγίες απόκτησης του Google Maps API Key

Οδηγίες απόκτησης του Google Maps API Key [39]:

1. Γίνεται άνοιγμα της σελίδας απόκτησης κλειδιών Goolge Maps https://developers.google.com/maps/documentation/android/v1/maps-api-signup και σύνδεση με Google λογαριασμό
2. Απαιτείται ανάγνωση και αποδοχή των όρων άδειας χρήσης. Στη συνέχεια γίνεται εισαγωγή του MD5 fingerprint και επιλογή "Generate API Key"
3. Χρήση του Google Maps API Key ως χαρακτηριστικό, απαιτείται να γίνει σε κάθε αντικείμενο MapView, κάθε εφαρμογής, που έχει

υπογραφεί με αυτό το πιστοποιητικό, για το οποίο αποκτήθηκε το Google Maps API v1 Key

π.χ. Δήλωση αντικειμένου MapView σε XML.

```
<com.google.android.maps.MapView
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:apiKey="05U_W5NSIk4SIsc7kavIA8r1pvLO7dLRFSCp4dA" />
```

π.χ. Αρχικοποίηση του αντικειμένου MapView απ' ευθείας από τον κώδικα.

```
map = new MapView (this,
       "05U_W5NSIk4SIsc7kavIA8r1pvLO7dLRFSCp4dA" );
```

4. Επίσης πρέπει να προστεθεί ένα στοιχείο <uses-library> με αναφορά στην εξωτερική βιβλιοθήκη com.google.android.maps στο Manifest της εφαρμογής. Το στοιχείο αυτό πρέπει να είναι απόγονος του στοιχείου <application>, π.χ.

```
<manifest
       xmlns:android="http://schemas.android.com/apk/res/android"
       package="com.example.package.name">
...
<application android:name="MyApplication" >
<uses-library android:name="com.google.android.maps" />
...
</application>
</manifest>
```

5. Τέλος, η εφαρμογή «υπογράφεται» με το πιστοποιητικό που αντιστοιχεί στο Maps API Key, στο οποίο αναφέρονται τα στοιχεία MapView

Σημειώνεται ότι η έκδοση 1 του Google Maps Android API έχει επίσημα καταργηθεί από 18 Μαρτίου του 2013. Αυτό σημαίνει ότι δε θα προστεθούν νέες λειτουργίες στο Google Maps Android API v1. Επίσης, δε θα δίνονται νέα κλειδιά API για αυτήν την έκδοση, αλλά τα ήδη υπάρχοντα κλειδιά θα λειτουργούν κανονικά [38].

4. Η βιβλιοθήκη Android MapView Balloons

Η βιβλιοθήκη αυτή παρέχει, με εύκολο τρόπο, εμφάνιση πληροφοριών στα στοιχεία overlays (αντικείμενα της βασικής κλάσης overlay, που α-ντιπροσωπεύουν στοιχεία τα οποία μπορεί να εμφανίζεται πάνω σε ένα χάρτη [40]), κατά τη χρήση της εξωτερικής βιβλιοθήκης Android Maps API (com.google.android.maps). Οι πληροφορίες εμφανίζονται σε ένα απλό διαλογικό «μπαλόνι», όπως παρουσιάζεται στην Εικόνα 27 [41].

Εικόνα 27. Μπαλόνι με πληροφορίες επιλεγμένου στοιχείου

Αποτελείται από το BalloonOverlayView, ένα αντικείμενο που αντιπροσωπεύει το μπαλόνι που εμφανίζεται στο χάρτη και το BalloonItemizedOverlay, μια αφηρημένη επέκταση της κλάσης ItemizedOverlay [41]. Η κλάση ItemizedOverlay είναι μια βασική κλάση για ένα στοιχείο overlay και αποτελείται από μια λίστα OverlayItems. Αυτή η κλάση είναι υπεύθυνη για λειτουργίες όπως η απεικόνιση ενός δείκτη (marker) για κάθε στοιχείο overlay και η διατήρηση της εστίασης ενός επιλεγμένου στοιχείου [42].

4.1 Ενσωμάτωση βιβλιοθήκης MapView Ballons σε Project

Η ενσωμάτωση της βιβλιοθήκης σε ένα project γίνεται ακολουθώντας τα παρακάτω βήματα [41]:

1. Μεταφόρτωση της βιβλιοθήκης-project από τη διεύθυνση https://github.com/jgilfelt/android-mapviewballoons.
2. Άνοιγμα του Eclipse, δεξί-κλικ στον Package Explorer και επιλογή Import.
3. Στο παράθυρο που εμφανίζεται, πραγματοποιούνται επιλογή του General και Existing Projects into Workspace, και κλικ στο κουμπί Next.
4. Πραγματοποιούνται επιλογή του Select archive file: και κλικ στο κουμπί Browse.
5. Πρέπει να γίνει αναζήτηση του αρχείου που μεταφορτώθηκε και κλικ στο κουμπί Άνοιγμα.
6. Πραγματοποιείται κλικ στο κουμπί Finish.

7. Αν εμφανίζεται σφάλμα στο project android-mapviewballoons, πρέπει να δημιουργηθεί ένας emulator με επίπεδο API 8 Google-Maps. Και δεξί κλικ στο project, επιλογή Android Tools < Fix Project Properties.
8. Στη συνέχεια, πρέπει να γίνει, στον Package Manager, δεξί κλικ στο project που χρειάζεται τη βιβλιοθήκη και επιλέγεται Properties.
9. Στο παράθυρο που εμφανίζεται, επιλέγεται η καρτέλα Android.
10. Στο πλαίσιο Library, γίνεται κλικ στο κουμπί Add.
11. Στο παράθυρο που εμφανίζεται, πραγματοποιούνται επιλογή του project-βιβλιοθήκη android-mapviewballoons, και κλικ στο κουμπί Ok.
12. Στη συνέχεια, πρέπει να δημιουργηθεί μια υποκλάση που εκτείνεται της κλάσης BalloonItemizedOverlay<OverlayItem>, η οποία περιέχεται στη βιβλιοθήκη Android MapView Balloons. (Όταν μια κλάση Γ εκτείνεται μιας κλάσης Β, συνεπάγεται πως η κλάση Γ διαθέτει αυτόματα όλες τις μεταβλητές και τις μεθόδους που καθορίζονται στην κλάση Β [43])
13. Αντί να εισαχθεί κώδικας στη συνάρτηση ONTAP() (η οποία ήδη χρησιμοποιείται για να εμφανίσει το μπαλόνι για κάθε στοιχείο overlay), θα εισαχθεί στην onBalloonTap()
14. Τα δεδομένα που εμφανίζονται σε κάθε μπαλόνι είναι ο τίτλος (title) και το απόσπασμα (snippet) και παρέχουν στον constructor κάθε OverlayItem

Σημειώνεται ότι στο αρχείο που μεταφορτώθηκε περιέχεται και μια εφαρμογή που παρουσιάζει τη λειτουργία της βιβλιοθήκης ως δείγμα των δυνατοτήτων και της λειτουργικότητας [41].

Η εφαρμογή για το Νομό Σερρών

1. Απαιτήσεις του συστήματος

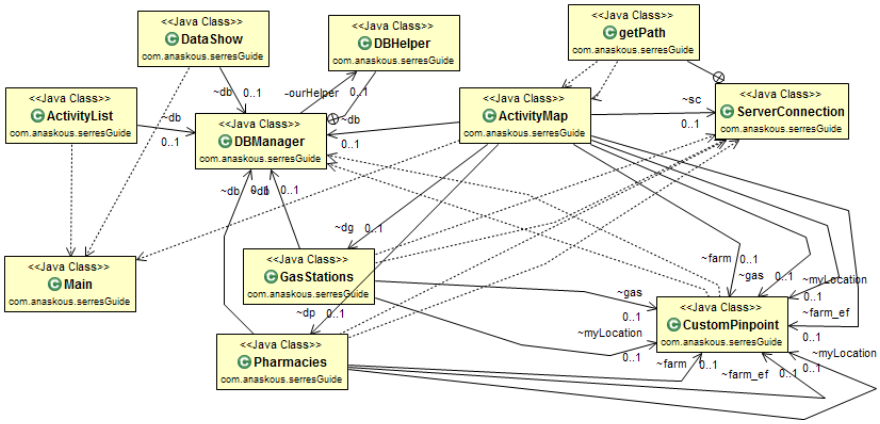
Σύμφωνα με το σκοπό της εφαρμογής στο πλαίσιο της παρούσας εργασίας, η εφαρμογή θα πρέπει να δίνει πρόσβαση σε πληροφορίες για τα φαρμακεία και τα πρατήρια καυσίμων των Σερρών μέσω διαδραστικού χάρτη, στον οποίο θα εμφανίζεται και η τοποθεσία του χρήστη. Ο σχεδιασμός της εφαρμογής θα πρέπει να είναι τέτοιος, ώστε ο χρήστης να μπορεί αμέσως να εντοπίζει το αντικείμενο που τον ενδιαφέρει, π.χ. κοντινότερο φαρμακείο ή φθηνότερο πρατήριο. Συνεπώς προκύπτουν οι παρακάτω λειτουργικές απαιτήσεις:

- Συγκέντρωση των απαραίτητων πληροφοριών μέσω ιστοσελίδων με σχετικά στοιχεία. Για τα φαρμακεία οι πληροφορίες αυτές είναι το όνομα, το τηλέφωνο, η διεύθυνση και οι ώρες εφημερίας (αν πρόκειται για εφημερεύον). Για τα πρατήρια καυσίμων οι πληροφορίες αυτές είναι η επωνυμία, το όνομα ιδιοκτήτη, η διεύθυνση, οι τιμές των καυσίμων και η ημερομηνία καταχώρησης των τιμών.
- Αυτόματη ενημέρωση των στοιχείων των φαρμακείων/πρατηρίων καυσίμων ανά ένα ορισμένο χρονικό διάστημα.
- Δυνατότητα χειροκίνητης ενημέρωσης των στοιχείων των φαρμακείων/πρατηρίων καυσίμων από το χρήστη. Ο χρήστης μπορεί να επιλέξει ανάμεσα σε κάποιες ομάδες πληροφοριών μία προς ενημέρωση. Π.χ. ενημέρωση μόνο των στοιχείων που αφορούν ένα συγκεκριμένο είδος καυσίμου, για γρηγορότερη απόδοση.
- Διατήρηση των στοιχείων των φαρμακείων/πρατηρίων καυσίμων σε βάση δεδομένων, για γρηγορότερη απόδοση του συστήματος. Τα στοιχεία δε θα συγκεντρώνονται κάθε φορά από την αρχή, αλλά μόνο όταν χρειάζεται ή όταν ο χρήστης το επιθυμεί.
- Εμφάνιση της θέσης του χρήστη στο χάρτη και ανανέωση αυτής (παρακολούθηση) κατά τη μετακίνηση του χρήστη.
- Εμφάνιση των φαρμακείων με μπαλόνι πληροφοριών (όνομα, ώρες εφημερίας) στο χάρτη.

- Εμφάνιση των πρατηρίων καυσίμων, που διαθέτουν το επιλεγμένο είδος καυσίμου από το χρήστη, με μπαλόνι πληροφοριών (επωνυμία, όνομα ιδιοκτήτη, είδος καυσίμου, τιμή, ημερομηνία καταχώρησης τιμής) στο χάρτη.
- Δυνατότητα εμφάνισης των φαρμακείων/πρατηρίων καυσίμων σε λίστα. Στη λίστα θα συμπεριλαμβάνονται και καταστήματα τα οποία πιθανών να μην έχουν καταχωρημένη έγκυρη διεύθυνση.
- Δυνατότητα ταξινόμησης ανά απόσταση, αλφαβητικά ή ανά τιμή στη λίστα, για διευκόλυνση του χρήστη στην εύρεση των πληροφοριών που τον ενδιαφέρουν.
- Εμφάνιση όλων των στοιχείων (όνομα, ώρες εφημερίας, διεύθυνση, απόσταση, τηλέφωνο) ενός φαρμακείου. Δυνατότητα άμεσης κλήσης των τηλεφωνικών αριθμών.
- Εμφάνιση όλων των στοιχείων (όνομα ιδιοκτήτη, επωνυμία, διεύθυνση, απόσταση, είδος καυσίμου, τιμή, ημερομηνία καταχώρησης τιμής) ενός πρατηρίου καυσίμων.
- Δυνατότητα λήψης οδηγιών προς το επιλεγμένο κατάστημα από το χρήστη. Αν η απόσταση του χρήστη από το επιλεγμένο κατάστημα είναι μεγαλύτερη των χιλιομέτρων, η δυνατότητα λήψης οδηγιών απενεργοποιείται, ώστε να μην επιβαρύνεται το σύστημα.
- Εμφάνιση μενού ρυθμίσεων για τη λήψη οδηγιών. Δυνατότητα επιλογής On/Off της λήψης οδηγιών και του μέσου μεταφοράς (πεζός, με αυτοκίνητο, με ποδήλατο).
- Υποστήριξη ελληνικών ή αγγλικών ανάλογα με τη γλώσσα της συσκευής.

2. Αρχιτεκτονική του συστήματος

Η αρχιτεκτονική της εφαρμογή σχεδιάστηκε έτσι ώστε να μην απαιτείται διαφορετικό Activity ή layout σε κανένα σημείο της εφαρμογής για τη λειτουργία των φαρμακείων ή των πρατηρίων καυσίμων. Η προσέγγιση αυτή εξασφαλίζει και μια ομοιομορφία στην παρουσίαση των πληροφοριών που βοηθά το χρήστη στην εύκολη εκμάθηση και τη χρήση της εφαρμογής. Οι λειτουργίες που υποχρεωτικά διαφέρουν για φαρμακεία και πρατήρια καυσίμων διαχωρίστηκαν σε κλάσεις που για ευκολία ονομάστηκαν Pharmacies και GasStations. Όλα τα Activities και οι κλάσεις της περιγράφονται παρακάτω. Στην Εικόνα 28 παρατίθεται το διάγραμμα των κλάσεων της εφαρμογής. Οι κλάσεις που περιέχονται στις βιβλιοθήκες της πλατφόρμας Android, του GoogleMapsAPI και του android-mapviewballoons δεν εμφανίζονται στο διάγραμμα για λόγους ευκολίας και κατανόησης.



Εικόνα 28. Διάγραμμα κλάσεων

2.1. Τα Activities της εφαρμογής

2.1.1. Main

Το Activity Main χρησιμοποιείται ως σημείο εκκίνησης της εφαρμογής. Δίνει στο χρήστη την δυνατότητα επιλογής “Φαρμακεία” ή “Πρατήρια Καυσίμων” και στέλνει τον κωδικό της επιλογής στο επόμενο Activity (ActivityMap).

2.1.2. ActivityMap

Το Activity αυτό είναι το βασικότερο της εφαρμογής καθώς περιλαμβάνει τις περισσότερες λειτουργίες της εφαρμογής. Αποτελεί επέκταση (extends) του MapActivity, ώστε να διαχειρίζεται τις λειτουργίες του χάρτη και χρησιμοποιείται για την εμφάνιση των φαρμακείων και των πρατηρίων καυσίμων στο χάρτη. Πριν την εμφάνιση των στοιχείων ελέγχεται αν τα στοιχεία απαιτείται ενημέρωση και, εάν απαιτείται, το ActivityMap εκκινεί τη διαδικασία λήψης των απαραίτητων στοιχείων. Επίσης είναι υπεύθυνο για τον εντοπισμό, την ενημέρωση και την προβολή στο χάρτη της τοποθεσίας του χρήστη. Ταυτόχρονα υποστηρίζει τις εξής πρόσθετες λειτουργίες:

- Δυνατότητα ανανέωσης των στοιχείων και ενημέρωση του χάρτη
- Δυνατότητα προβολής δρομολογίου προς ένα επιλεγμένο κατάστημα, με χρήση της υπηρεσίας "Λήψη οδηγιών" του Google-Maps
- Προβολή παραθύρου ρυθμίσεων δρομολογίου

2.1.3. ActivityList

Το Activity αυτό αποτελεί επέκταση (extends) του ExpandableListActivity, ώστε να διαχειρίζεται την επεκτάσιμη λίστα και χρησιμοποιείται για

την εμφάνιση στοιχείων των φαρμακείων αλλά και των πρατηρίων καυσίμων σε λίστα. Ο λόγος που χρησιμοποιήθηκε επεκτάσιμη και όχι απλή λίστα, είναι για το διαχωρισμό των εφημερευόντων φαρμακείων από τα υπόλοιπα για διευκόλυνση του χρήστη. Παράλληλα, υποστηρίζονται οι εξής πρόσθετες λειτουργίες:

- Υπολογισμός απόστασης του κάθε καταστήματος από το χρήστη
- Προβολή παράθυρου με επιλογές ταξινόμησης (Επωνυμία, Απόσταση, Τιμή)
- Ταξινόμηση της λίστας ανάλογα με την επιλογή του χρήστη

2.1.4. DataShow

Το Activity DataShow χρησιμοποιείται για την προβολή των στοιχείων των φαρμακείων και των πρατηρίων καυσίμων. Ο σχεδιασμός του πίνακα στον οποίο εμφανίζονται τα στοιχεία TableLayout, γίνεται δυναμικά ώστε να προσαρμόζεται στα στοιχεία των φαρμακείων ή των πρατηρίων καυσίμων. Παράλληλα, υποστηρίζονται οι εξής πρόσθετες λειτουργίες:

- Υπολογισμός απόστασης του επιλεγμένου καταστήματος από το χρήστη
- Εντοπισμός τηλεφωνικού αριθμού και δυνατότητα άμεσης κλήσης του

2.2. Οι συμπληρωματικές κλάσεις της εφαρμογής

Στις παραγράφους που ακολουθούν περιγράφονται συνοπτικά οι συμπληρωματικές κλάσεις της εφαρμογής. Σημειώνεται ότι οι κλάσεις που περιέχονται στις βιβλιοθήκες της πλατφόρμας Android, του GoogleMapsAPI και του android-mapviewballoons δεν αναφέρονται.

2.2.1. CustomPinpoint

Η κλάση αυτή αποτελεί επέκταση (extends) της BalloonItemizedOverlay<OverlayItem> και είναι υπεύθυνη για τη διαχείριση των σημείων που εμφανίζονται στο χάρτη. Επίσης, μέσω αυτής σχεδιάζονται τα μπαλόνια πληροφοριών.

2.2.2. DBManager

Η κλάση DBManager περιέχει συναρτήσεις με όλα τα sqlite ερωτήματα που χρησιμοποιούνται στην εφαρμογή. Επίσης περιέχει την υποκλάση DBHelper. Η κλάση DBHelper αυτή αποτελεί επέκταση (extends) της SQLiteOpenHelper και περιέχει συναρτήσεις για τη δημιουργία των πινάκων της βάσης δεδομένων.

2.2.3. ServerConnection

Η κλάση αυτή περιέχει λειτουργίες που απαιτούν σύνδεση με το διαδίκτυο. Παρέχει τις εξής δυνατότητες:

- Έλεγχος σύνδεσης με μία ιστοσελίδα
- Λήψη πηγαίου κώδικα μιας ιστοσελίδας
- Λήψη στοιχείων δρομολογίου από ένα σημείο σε ένα άλλο και αποκωδικοποίηση αυτών
- Λήψη συντεταγμένων μίας διεύθυνσης

2.2.4. Pharmacies

Η κλάση Pharmacies περιέχει συναρτήσεις αποκλειστικά για το χειρισμό των στοιχείων των φαρμακείων. Συγκεκριμένα υποστηρίζει:

- Συντονισμό της λήψης στοιχείων των φαρμακείων
- Διαχωρισμό στοιχείων των φαρμακείων από τον πηγαίο κώδικα της σελίδας αναφοράς
- Εμφάνιση φαρμακείων στο χάρτη
- Διαγραφή στοιχείων των φαρμακείων και αφαίρεση των φαρμακείων από το χάρτη

2.2.5. GasStations

Η κλάση GasStations περιέχει συναρτήσεις αποκλειστικά για το χειρισμό των στοιχείων των πρατηρίων καυσίμων. Συγκεκριμένα υποστηρίζει:

- Συντονισμό της λήψης στοιχείων των πρατηρίων καυσίμων
- Διαχωρισμό στοιχείων των πρατηρίων καυσίμων από τον πηγαίο κώδικα της σελίδας αναφοράς
- Εμφάνιση πρατηρίων καυσίμων στο χάρτη
- Διαγραφή στοιχείων των πρατηρίων καυσίμων και αφαίρεση των πρατηρίων καυσίμων από το χάρτη

2.2.6. RoutePathOverlay

Η κλάση RoutePathOverLay αποτελεί επέκταση (extends) της Overlay και περιέχει συναρτήσεις για τη δημιουργία ενός αντικειμένου - δρομολογίου.

2.3. Η βάση δεδομένων της εφαρμογής

Στην παρούσα εργασία αν και δεν ήταν απαραίτητη η χρήση βάσης δεδομένων αποφασίστηκε να χρησιμοποιηθεί. Ο λόγος είναι ότι κάθε φορά που πραγματοποιείται λήψη δεδομένων παρουσιάζεται σημαντική επιβάρυνση του συστήματος. Επίσης όσο γίνεται η μεταφορά δεδομένων ο χρήστης μένει στην αναμονή. Έτσι αν η εφαρμογή κατέβαζε κάθε φορά όλες τις απαιτούμενες πληροφορίες θα παρουσιαζόταν μεγάλη χρονική

καθυστέρηση και η εφαρμογή δε θα ήταν λειτουργική. Η λύση σε αυτό το πρόβλημα δόθηκε με τη χρήση βάσης δεδομένων. Τα δεδομένα φυλάσσονται στη βάση δεδομένων και με έναν έλεγχο ενημερώσεων αποφασίζεται πότε είναι απαραίτητη η εκ νέου λήψη δεδομένων.

Ωστόσο δόθηκε η δυνατότητα στο χρήστη να ανανεώνει την πληροφορία που επιθυμεί, ώστε ο χρήστης να μπορεί να έχει χειροκίνητο έλεγχο και τη δυνατότητα να γνωρίζει ότι η πληροφορία που εμφανίζεται είναι η τρέχουσα.

Η βάση δεδομένων αποτελείται από δύο ανεξάρτητους πίνακες με τις προφανείς ονομασίες PHARMACIES και GASSTATIONS. Οι πίνακες αυτοί περιέχουν τα στοιχεία των φαρμακείων και των πρατηρίων αντίστοιχα. Τα πεδία του πίνακα PHARMACIES είναι id, name, phone, address, latitude, longitude, distance, hours και mapId. Τα πεδία του πίνακα GASSTATIONS είναι id, name, brand, address, latitude, longitude, distance, hours, mapId, price, product και prodId. Στα πεδία latitude, longitude καταχωρούνται οι γεωγραφικές συντεταγμένες που υπολογίζονται από τη διεύθυνση με τη διαδικασία της αντίστροφης γεωκωδικοποίησης (reverse geocoding) που παρέχει το Google Maps API. Στο πεδίο mapId εισάγεται το index της θέσης του σημείου στη λίστα OverlayItems. Το mapId χρησιμεύει στην αναγνώριση του σημείου από τη θέση του στην λίστα. Στο πεδίο distance αποθηκεύεται προσωρινά η απόσταση του καταστήματος από την τοποθεσία του χρήστη. Χρησιμοποιείται για την εύκολη ταξινόμηση των καταστημάτων ανά απόσταση. Το πεδίο prodId περιέχει το id του προϊόντος καυσίμου. Σε όλα τα υπόλοιπα πεδία των δύο πινάκων εισάγονται οι πληροφορίες που λαμβάνονται από την λήψη δεδομένων.

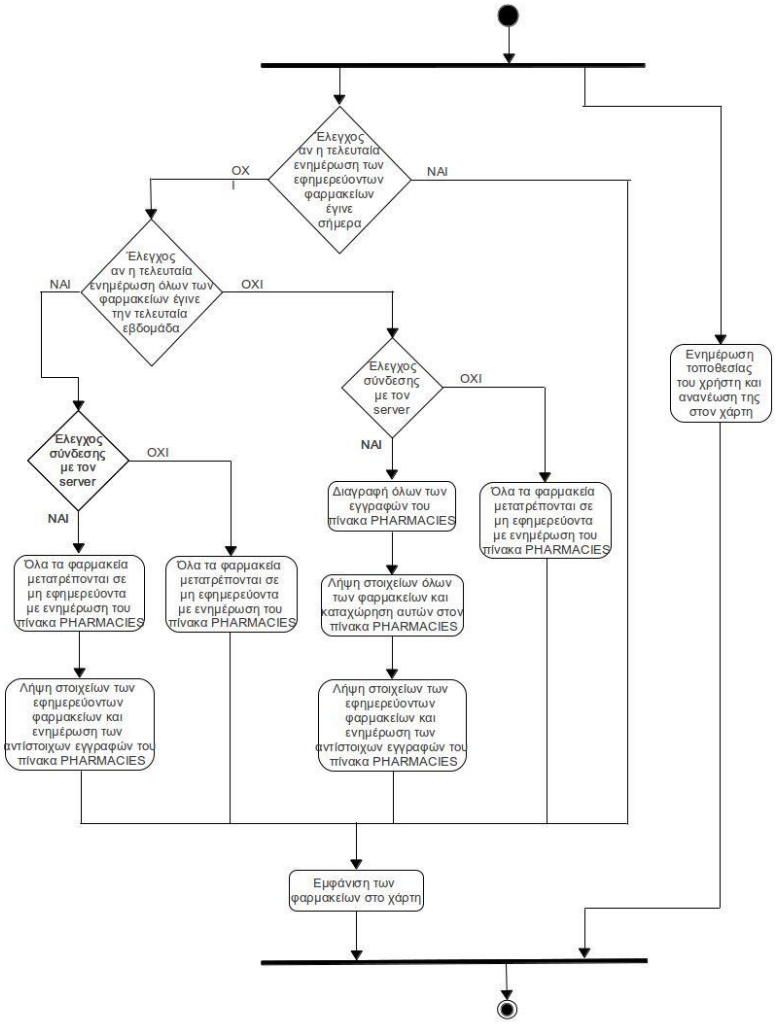
3. Η εφαρμογή

Το βασικό Activity της εφαρμογής είναι το ActivityMap. Το ActivityMap έχει τις περισσότερες λειτουργίες, συνεπώς και τη μεγαλύτερη δυσκολία στο σχεδιασμό. Στη συνάρτηση onCreate() του ActivityMap αρχικά γίνονται οι απαραίτητες αρχικοποιήσεις και στη συνέχεια καλείται η συνάρτηση getPharmacies() ή η getGasStations() του ActivityMap, ανάλογα με την επιλογή του χρήστη στο κεντρικό μενού. Οι συναρτήσεις αυτές συντονίζουν τις παρακάτω βασικές λειτουργίες του ActivityMap:

- ενημέρωση – λήψη πληροφοριών
- εμφάνιση των σημείων στο χάρτη

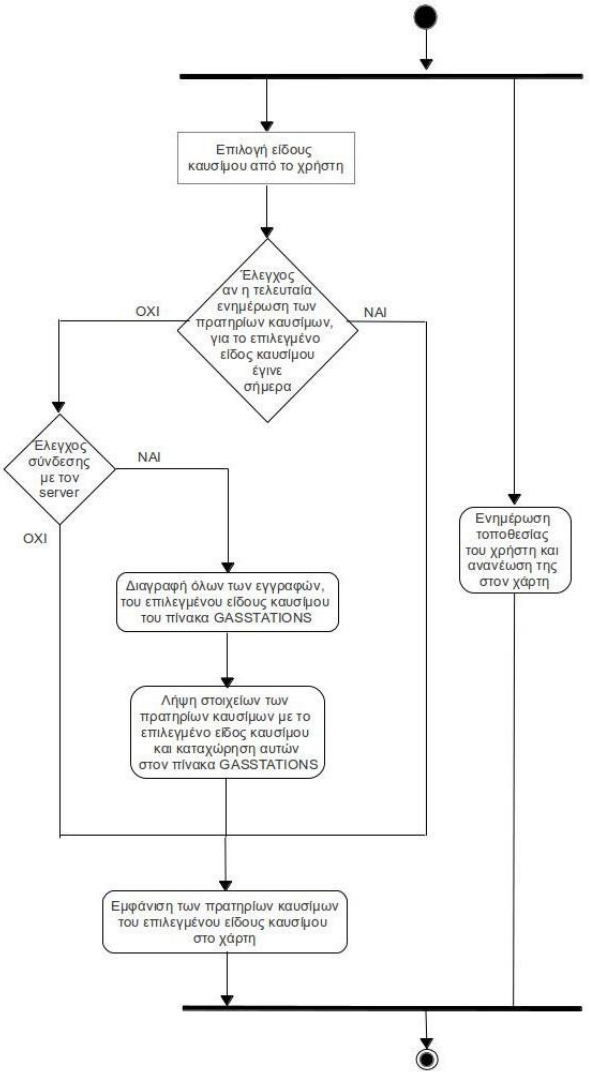
Οι παραπάνω λειτουργίες και οι συναρτήσεις που χρησιμοποιούνται σε αυτές, καθώς και άλλες βασικές λειτουργίες της εφαρμογής περιγράφονται αναλυτικότερα σε επόμενες παραγράφους. Το ActivityMap είναι επίσης υπεύθυνο για τον εντοπισμό και την ενημέρωση της θέσης του χρήστη. Για την καλύτερη λειτουργία της εφαρμογής αποφασίστηκε η θέση

του χρήστη να εντοπίζεται από wifi αλλά και από GPS. Έτσι αν κάποιο από τα δύο δε λειτουργεί ικανοποιητικά, η θέση θα συνεχίζει να εντοπίζεται. Αν η θέση του χρήστη εντοπίζεται και από το wifi αλλά και από το GPS, προτιμάται η πληροφορία που προέρχεται από το GPS καθώς είναι πιο ακριβής [44]. Ο εντοπισμός και η ενημέρωση της θέσης του χρήστη γίνονται αυτόματα από το ActivityMap και η ενημέρωση του χάρτη μετά από αλλαγή γίνεται στη συνάρτηση onLocationChanged(Location l) του ActivityMap. Στην Εικόνα 29 παρουσιάζεται η λειτουργία της συνάρτησης getPharmacies(). Παράλληλα παρουσιάζεται η αυτόματη ενημέρωση της θέσης του χρήστη.



Εικόνα 29. Διάγραμμα ροής του ActivityMap σχετικά με τα φαρμακεία

Στην Εικόνα 30 παρουσιάζεται η λειτουργία της συνάρτησης getGasStations(). Παράλληλα παρουσιάζεται η αυτόματη ενημέρωση της θέσης του χρήστη.



Εικόνα 30. Διάγραμμα ροής του ActivityMap σχετικά με τα πρατήρια καυσίμων

3.1. Συλλογή των πληροφοριών

Οι πληροφορίες των φαρμακείων λαμβάνονται από το επίσημο site του Φαρμακευτικού Συλλόγου Σερρών http://www.fsserron.gr/ και των πρατηρίων καυσίμων από το site Παρατηρητήριο τιμών υγρών καυσίμων του Υπουργείου Ανάπτυξης Ανταγωνιστικότητας, Υποδομών, Μεταφορών και Δικτύων http://www.fuelprices.gr/. Αυτή η προσέγγιση υιοθετήθηκε

για τη συλλογή των εγκυρότερων κατά το δυνατόν πληροφοριών για τους χρήστες.

Τρεις συγκεκριμένες συνάρτησης, οι συναρτήσεις downloadOnDutyPharmacies(), downloadAllPharmacies() της κλάσης Pharmacies και downloadGasStations(String prod) της κλάσης GasStations συντονίζουν τη λήψη των αντίστοιχων δεδομένων. Κάθε μία από αυτές περιλαμβάνει κλήση της συνάρτησης readHtml(String url) της κλάσης ServerConnection για λήψη του περιεχομένου της σελίδας με τις επιθυμητές πληροφορίες. Στη συνέχεια δύο άλλες συναρτήσεις, οι συναρτήσεις downloadOnDutyPharmacies() και downloadAllPharmacies() καλούν τη συνάρτηση getPharmacies(BufferedReader reader, String flag) της κλάσης Pharmacies για διαχωρισμό και αποθήκευση των πληροφοριών των φαρμακείων. Ενώ η downloadGasStations(String prod) καλεί τη συνάρτηση getGasStations(BufferedReader reader) της κλάσης GasStations, για διαχωρισμό και αποθήκευση των πληροφοριών των πρατηρίων καυσίμων.

3.1.1. Λήψη περιεχομένου σελίδας

Οι πληροφορίες των εφημερευόντων φαρμακείων πηγάζουν από την ιστοσελίδα του Φαρμακευτικού Συλλόγου των Σερρών (http://www.fsserron.gr/). Ωστόσο στη σελίδα αυτή δεν περιλαμβάνονται βασικές πληροφορίες όπως η διεύθυνση των εφημερευόντων φαρμακείων. Γι' αυτό το λόγο αποφασίστηκε ότι είναι απαραίτητη και η λήψη των στοιχείων όλων των φαρμακείων από αντίστοιχες σελίδες http://www.fsserron.gr/index.php?option=com_contact&view=category&catid=12, http://www.fsserron.gr/index.php?option=com_contact&view=category&catid=12&limitstart=20, http://www.fsserron.gr/index.php?option=com_contact&view=category&catid=12&limitstart=40, http://www.fsserron.gr/index.php?option=com_contact&view=category&catid=12&limitstart=60. Οι πληροφορίες των πρατηρίων καυσίμων αντίστοιχα πηγάζουν από την ιστοσελίδα του Παρατηρητηρίου Υγρών Καυσίμων (http://www.fuelprices.gr/CheckPrices?DD=62010200&DD=62010300&DD=62010100&DD=62010400&date_filter=7&prodclass=1), όπου στη μεταβλητή prodclass δίνεται τιμή από 1 έως 7, ανάλογα με το είδος καυσίμου που επιλέγει ο χρήστης. Οι επιπρόσθετοι κωδικοί που χρησιμοποιούνται στο URL περιορίζουν την αναζήτηση στο Νομό Σερρών και για την τρέχουσα εβδομάδα.

Η συνάρτηση readHtml(String url, String encoding) της κλάσης ServerConnection λαμβάνει τον πηγαίο κώδικα της σελίδας που δίνεται ως όρισμα και τον επιστρέφει σε μια μεταβλητή τύπου BufferedReader. Η παράμετρος encoding περιέχει την κωδικοποίηση της σελίδας, για τη σω-

στή λήψη των δεδομένων. Η κωδικοποίηση του site των φαρμακείων είναι “utf-8”, ενώ των πρατηρίων καυσίμων “iso-8859-7”.

3.1.2. Διαχωρισμός των χρήσιμων πληροφοριών

Ο πηγαίος κώδικας μιας σελίδας μπορεί εύκολα να προβληθεί μέσω κάθε browser. Κάθε σελίδα περιλαμβάνει πολλές πληροφορίες που δεν είναι χρήσιμες στην παρούσα εργασία. Έτσι, η εφαρμογή καλείται να συλλέξει τις χρήσιμες πληροφορίες από τον πηγαίο κώδικα της κάθε σελίδας. Σε κάθε σελίδα όμως η δομή του πηγαίου κώδικα είναι διαφορετική. Συνεπώς η μέθοδος διαχωρισμού (parsing) των χρήσιμων δεδομένων είναι διαφορετική για τη σελίδα τον φαρμακείων από ότι για τη σελίδα των πρατηρίων καυσίμων.

Με κάποια έκπληξη παρατηρήθηκε διαφορά στο διαχωρισμό πηγαίου κώδικα σελίδας που λήφθηκε από wifi με πηγαίου κώδικα σελίδας που λήφθηκε από 3G. Συγκεκριμένα παρατηρήθηκε ότι το περιεχόμενο μιας σελίδας, όταν λαμβάνεται από δίκτυο wifi, είναι ακριβώς όπως εμφανίζεται στον browser ενός υπολογιστή, ενώ όταν η ίδια σελίδα λαμβάνεται από δίκτυο 3G, λόγω εσωτερικής επεξεργασίας από τον πάροχο του 3G, μπορεί να απουσιάζουν οι χαρακτήρες αλλαγής γραμμής “\n” ή “\r”. Αυτό προκάλεσε επιπρόσθετο πρόβλημα, αφού σύμφωνα με την αρχική λύση που υιοθετήθηκε, το περιεχόμενο της σελίδας διαβαζόταν γραμμή - γραμμή με την εντολή readLine() και κάθε γραμμή δεχόταν επεξεργασία για το διαχωρισμό των χρήσιμων πληροφοριών. Έτσι, όταν γινόταν χρήση του 3G δικτύου, όλο το περιεχόμενο της σελίδας θεωρούνταν μία γραμμή και το περιεχόμενο της σελίδας δεν διαχωριζόταν σωστά. Η λύση που δόθηκε είναι η αποθήκευση κάθε γραμμής σε ένα αντικείμενο StringBuilder. Έτσι το αντικείμενο αυτό θα έχει το ίδιο περιεχόμενο από όποιο δίκτυο κι αν προέρχονται τα δεδομένα. Στη συνέχεια γίνεται διαχωρισμός των χρήσιμων πληροφοριών από αυτό το αντικείμενο, όπως παρουσιάζεται παρακάτω.

3.1.2.1. Διαχωρισμός χρήσιμων πληροφοριών φαρμακείων

Η συνάρτηση getPharmacies(BufferedReader reader, String flag), της κλάσης Pharmacies, είναι υπεύθυνη για το διαχωρισμό των χρήσιμων πληροφοριών των φαρμακείων. Η διαδικασία αυτή περιλαμβάνει:

- Αρχικά καταχωρείται σε μία μεταβλητή τύπου StringBuilder το τμήμα του περιεχόμενου της μεταβλητής reader που περιλαμβάνει τις χρήσιμες πληροφορίες. Ο λόγος που χρησιμοποιήθηκε StringBuilder και όχι String είναι ότι ο αριθμός χαρακτήρων που μπορεί να κρατήσει ένα String δεν επαρκεί.
- Ο πρώτος διαχωρισμός των χρήσιμων πληροφοριών γίνεται κατά την αρχικοποίηση του StringBuilder, με τη χρήση δύο μεταβλητών τύπου String, dataStart και dataEnd. Οι μεταβλητές dataStart

και dataEnd έχουν ως τιμή δύο λέξεις κλειδιά που εντοπίζονται στο περιεχόμενο της σελίδας και ορίζουν την αρχή και το τέλος της χρήσιμης πληροφορίας. Οι μεταβλητές dataStart και dataEnd έχουν διαφορετική τιμή όμως για τη σελίδα των εφημερευόντων φαρμακείων από ότι για τις σελίδες όλων των φαρμακείων. Η μεταβλητή flag έχει τιμή "Pharmacy" όταν πρόκειται για τις σελίδες όλων των φαρμακείων ή "OnDuty" όταν πρόκειται για τη σελίδα των εφημερευόντων και χρησιμοποιείται σαν σημαία για τη σωστή αρχικοποίηση των μεταβλητών dataStart και dataEnd. Στην περίπτωση που η μεταβλητή flag έχει τιμή "OnDuty" η dataStart ορίζεται "Τηλέφωνο" και η dataEnd "</tbody>". Στην περίπτωση που η μεταβλητή flag έχει τιμή "Pharmacy" η dataStart ορίζεται "Σήμερα" και η dataEnd "</td>". Οι ενέργειες που ακολουθούν γίνονται με τον ίδιο τρόπο για ότι τιμή κι αν έχει το flag.

- Στη συνέχεια, οι πληροφορίες διαχωρίζονται ανά φαρμακείο. Ο διαχωρισμός αυτός γίνεται με χρήση της εντολής split(String regularExpression) και της regular expression "<a href[^>]+\\>". Η έκφραση αυτή ορίζει μία συμβολοσειρά που ξεκινάει από "<a href", ακολουθούν οποιοσδήποτε ή οποιοιδήποτε χαρακτήρες εκτός από ">" και τελειώνει με το χαρακτήρα ">". Όπου συναντάται η έκφραση αυτή γίνεται διαχωρισμός και τα αποτελέσματα αποθηκεύονται σε πίνακα String.
- Η επόμενη ενέργεια που πραγματοποιείται είναι ο διαχωρισμός κάθε εγγραφής του προηγούμενου πίνακα σε χαρακτηριστικά των φαρμακείων. Για την απλοποίηση του προβλήματος αρχικά κάθε εγγραφή του πίνακα δέχεται κάποιες τροποποιήσεις με χρήση της εντολής replaceAll(String regularExpression, String replacement). Η πρώτη τροποποίηση γίνεται με την εντολή replaceAll(">([0-9]|[0-9][0-9]|[ ])<", "><"). Η έκφραση που χρησιμοποιείται, ορίζει μία συμβολοσειρά που ξεκινάει με τον χαρακτήρα ">", ακολουθεί ένας μονοψήφιος ή διψήφιος αριθμός και τελειώνει με τον χαρακτήρα "<". Έτσι όπου συναντάται αυτή η έκφραση αντικαθίσταται από τους χαρακτήρες "><". Η επόμενη τροποποίηση γίνεται με την εντολή replaceAll("<[^>]+\\>", "|"). Η έκφραση που χρησιμοποιείται, ορίζει μία συμβολοσειρά που ξεκινάει με τον χαρακτήρα "<", ακολουθούν οποιοσδήποτε ή οποιοιδήποτε χαρακτήρες εκτός από ">" και τελειώνει με τον χαρακτήρα ">". Έτσι όπου συναντάται αυτή η έκφραση αντικαθίσταται από τον χαρακτήρα "|". Αφού έχουν γίνει αυτές οι τροποποιήσεις ο διαχωρισμός της κάθε εγγραφής του πίνακα σε χαρακτηριστικά γίνεται εύκολα με χρήση της εντολής split("[\\|]+"). Η έκφραση που χρησιμοποιείται, ορίζει μία συμβολοσειρά που περιέχει τον χαρακτήρα "|" μία ή περισσότερες φορές. Τα δεδομένα αποθηκεύονται σε νέο δισδιάστατο πίνακα String.

- Τελευταία ενέργεια είναι η καταχώρηση των περιεχομένων του δισδιάστατου αυτού πίνακα στην βάση δεδομένων για άμεση πρόσβαση σε αυτά όποτε είναι απαραίτητο. Σε αυτό το σημείο η τιμή της μεταβλητής flag παίζει ρόλο καθώς λαμβάνονται διαφορετικές πληροφορίες από τη σελίδα των εφημερευόντων φαρμακείων και διαφορετικές από τις σελίδες όλων των φαρμακείων. Από τις σελίδες όλων των φαρμακείων πηγάζουν τα στοιχεία Όνομα, Τηλέφωνο και Διεύθυνση όλων των φαρμακείων, ενώ από την σελίδα των εφημερευόντων πηγάζουν τα στοιχεία Όνομα και Ώρες Εφημερίας των εφημερευόντων. Επίσης για κάθε φαρμακείο, με χρήση της συνάρτησης getCoordinartes(String address) της κλάσης ServerConnection, υπολογίζεται το γεωγραφικό μήκος και πλάτος του. Όταν η μεταβλητή flag έχει τιμή "Pharmacy" εκτελείται ένα sql ερώτημα τύπου insert into που καταχωρεί τα στοιχεία όλων των φαρμακείων στη βάση καθώς και το γεωγραφικό μήκος και πλάτος. Όταν η μεταβλητή flag έχει τιμή "OnDuty" εκτελείται ένα ερώτημα τύπου update που ενημερώνει τις εγγραφές των εφημερευόντων φαρμακείων με τις ώρες εφημερίας.

3.1.2.2. Διαχωρισμός χρήσιμων πληροφοριών καυσίμων

Η συνάρτηση getGasStations(BufferedReader reader), της κλάσης GasStations, είναι υπεύθυνη για το διαχωρισμό των χρήσιμων πληροφοριών των πρατηρίων καυσίμων. Η διαδικασία αυτή είναι παρόμοια με αυτή που ακολουθείται στην περίπτωση των φαρμακείων και περιλαμβάνει:

- Αρχικά καταχωρείται σε μία μεταβλητή τύπου StringBuilder το τμήμα του περιεχόμενου της μεταβλητής reader που περιλαμβάνει τις χρήσιμες πληροφορίες. Κατά την αρχικοποίηση του StringBuilder γίνεται ο πρώτος διαχωρισμός των χρήσιμων πληροφοριών, με τη χρήση δύο μεταβλητών τύπου String, dataStart και dataEnd. Οι μεταβλητές dataStart και dataEnd έχουν ως τιμή δύο λέξεις κλειδιά "</select>", "</p>" αντίστοιχα, που εντοπίζονται στο περιεχόμενο της σελίδας και ορίζουν την αρχή και το τέλος της χρήσιμης πληροφορίας.
- Στη συνέχεια, οι πληροφορίες διαχωρίζονται ανά πρατήριο καυσίμων. Ο διαχωρισμός αυτός γίνεται με χρήση της εντολής split("top"). Όπου συναντάται η λέξη "top" γίνεται διαχωρισμός και τα αποτελέσματα αποθηκεύονται σε πίνακα String.
- Η επόμενη ενέργεια που πραγματοποιείται είναι ο διαχωρισμός κάθε εγγραφής του προηγούμενου πίνακα σε χαρακτηριστικά των πρατηρίων καυσίμων. Για την απλοποίηση του προβλήματος αρχικά κάθε εγγραφή του πίνακα δέχεται κάποιες τροποποιήσεις με χρήση της εντολής replaceAll(String regularExpression, String

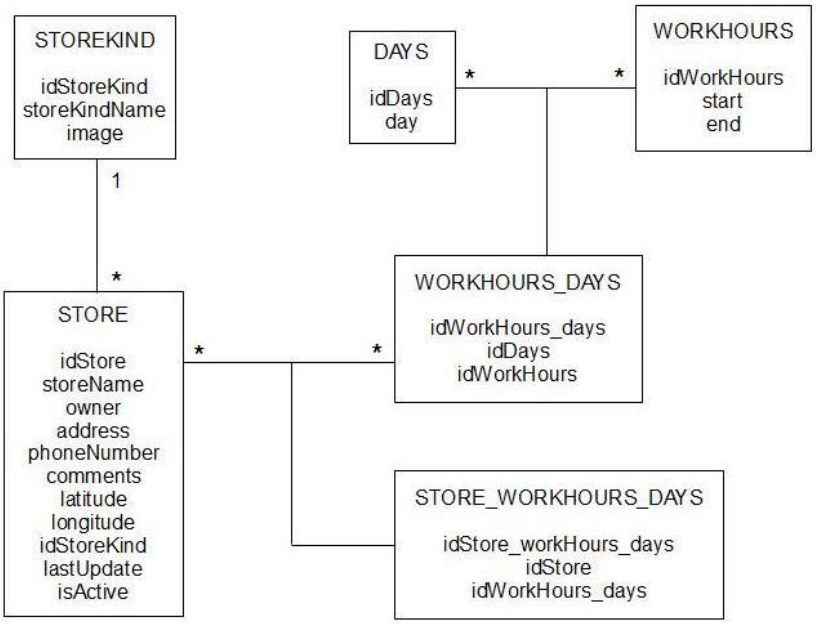
replacement). Η πρώτη τροποποίηση γίνεται με την εντολή replaceAll("<[^>]+\\>", "|"). Η έκφραση που χρησιμοποιείται, ορίζει μία συμβολοσειρά που ξεκινάει με τον χαρακτήρα "<", ακολουθούν οποιοσδήποτε ή οποιοιδήποτε χαρακτήρες εκτός από ">" και τελειώνει με τον χαρακτήρα ">". Έτσι όπου συναντάται αυτή η έκφραση αντικαθίσταται από τον χαρακτήρα "|". Η επόμενες τροποποιήσεις γίνονται με τις εντολές replace("Τιμή: ", ""), replace("Προϊόν: ", ""), replace("Σήμα: ", ""), replace("Διεύθυνση: ", ""), replace("Ημερ. / Ώρα δήλωσης", "") και replace(": ", ""), αντικαθιστούν την πρώτη παράμετρο με "", διαγράφοντας έτσι την περιττή πληροφορία. Αφού έχουν γίνει αυτές οι τροποποιήσεις ο διαχωρισμός της κάθε εγγραφής του πίνακα σε χαρακτηριστικά γίνεται εύκολα με χρήση της εντολής split("[\\|]+"). Η έκφραση που χρησιμοποιείται, ορίζει μία συμβολοσειρά που περιέχει τον χαρακτήρα "|" μία ή περισσότερες φορές. Τα δεδομένα αποθηκεύονται σε νέο δισδιάστατο πίνακα String.

- Τελευταία ενέργεια είναι η καταχώρηση των περιεχομένων του δισδιάστατου αυτού πίνακα στη βάση δεδομένων. Τα στοιχεία που περιέχονται στο δισδιάστατο αυτό πίνακα είναι Επωνυμία, Όνομα Ιδιοκτήτη, Διεύθυνση, Προϊόν Καυσίμου, Τιμή Προϊόντος και Ημερομηνία Τελευταίας Ενημέρωσης Τιμής. Επίσης, για κάθε πρατήριο καυσίμων, με χρήση της συνάρτησης getCoordinartes(String address) της κλάσης ServerConnection, υπολογίζεται το γεωγραφικό μήκος και πλάτος του. Με την εκτέλεση ενός sql ερωτήματος τύπου insert into που καταχωρούνται όλα τα στοιχεία, μαζί με το γεωγραφικό μήκος και πλάτος στη βάση δεδομένων.

3.1.3. Λύσεις που απορρίφτηκαν

Στον αρχικό σχεδιασμό της παρούσας πτυχιακής εργασίας θεωρήθηκε αναγκαίο να δημιουργηθεί μια ενδιάμεση πύλη συλλογής πληροφοριών. Στην περίπτωση αυτή, η εφαρμογή Android θα έπαιρνε τις πληροφορίες επικοινωνώντας με την πύλη και θα τις καταχωρούσε στη βάση δεδομένων της συσκευής. Από την αρχική αυτή προσέγγιση είχε προκύψει και ο τίτλος της πτυχιακής εργασίας που περιλαμβάνει την έννοια της διαδικτυακής πύλης. Κατά τη μελέτη αυτής της λύσης δεν είχε αποφασιστεί ακόμα η ενσωμάτωση της δυνατότητα πληροφόρησης για τα πρατήρια καυσίμων, οπότε οι πληροφορίες για τα πρατήρια καυσίμων θα ήταν σχεδόν στατικές. Ο πίνακας με τις πληροφορίες των πρατηρίων καυσίμων θα ήταν ανεξάρτητος από των φαρμακείων. Επίσης, θεωρήθηκε ότι οι εφημερίες των φαρμακείων αλλάζουν μία φορά την εβδομάδα, οπότε η εφαρμογή θα τις ζητούσε από την πύλη και θα τις καταχωρούσε στην εσωτερική βάση δεδομένων της συσκευής. Η sqlite όμως δεν υποστηρίζει τύπους ημερομηνίας και ώρας. Η βάση δεδομένων σχεδιάστηκε ώστε να δίνει λύση σε αυτό το πρόβλημα, χρησιμοποιώντας τύπο integer με τιμή από 1

έως 7 για τις ημέρες και τύπο real με τιμές από 0.00 έως 24.00 για τις ώρες, όπως φαίνεται στην Εικόνα 31.



Εικόνα 31. Διάγραμμα E-R, μέρος λύσης που απορρίφτηκε

Κατά την εμφάνιση των εφημερευόντων φαρμακείων η εφαρμογή θα έπρεπε να ανατρέχει στη βάση δεδομένων με ένα sql ερώτημα και συγκρίνοντας τις ώρες εφημερίας να αντιλαμβάνεται ποια φαρμακεία εφημερεύουν. Παρακάτω απεικονίζεται το πολύπλοκο sql ερώτημα που φέρνει τα στοιχεία των εφημερευόντων φαρμακείων.

```
select distinct STORE.* from STORE, STOREKIND, WORKHOURS, DAYS,
WORKHOURS_DAYS, STORE_WORKHOURS_DAYS where
STORE.idStore=STOREKIND.idStore and
DAYS.idDays=WORKHOURS_DAYS.idDays and WORK-
HOURS.idWorkHours=WORKHOURS_DAYS.idWorkHours and WORK-
HOURS_DAYS.idWorkHours_days=STORE_WORKHOURS_DAYS.idWorkHours_days
and STORE_WORKHOURS_DAYS.idStore=STORE.idStore and idStoreKind='0'
and ( (start<=end and (select strftime("%H%M", 'now','localtime'))
between start and end and DAYS.idDAYS=(select strftime("%w", 'now',
'locatime')+1)) or (start>end and (((select strftime("%H%M",
'now','localtime')) between start and '24' and DAYS.idDays=(select
strftime("%w", 'now', 'locatime')+1)) or ((select strftime("%H%M",
'now', 'localtime')) between '0' and end and (((select
strftime("%w", 'now','localtime')+1)!='1' and DAYS.idDays=(select
strftime("%w", 'now', 'localtime'))) or ((select strftime("%w",
'now','localtime')+1)!='1' and DAYS.idDays='7')))));
```

Τα προβλήματα που προέκυψαν από την υιοθέτηση αυτής της λύσης ήταν πολλά. Κατ' αρχάς η χρήση ενός server προϋποθέτει όχι μόνο ότι θα είναι πάντα διαθέσιμος, αλλά και ενημερωμένος με τις εφημερίες, οι οποίες

ανανεώνονται κάθε βδομάδα. Όμως οι εφημερίες των φαρμακείων όπως αναρτούνται στην πόρτα κάθε φαρμακείου κάθε εβδομάδα δεν υπάρχουν σε ηλεκτρονική μορφή. Οπότε η ενημέρωση του server θα έπρεπε να γίνεται χειροκίνητα. Η λύση αυτή απορρίφτηκε πριν να προχωρήσει περαιτέρω η υλοποίηση καθώς είναι πολύ πολύπλοκη, χωρίς να προσφέρει καλύτερα αποτελέσματα. Παράλληλα η λύση κρίθηκε ως μη αναγκαία και ακριβή καθώς εισάγει την ανάγκη ενδιάμεσης πύλης με ειδικές διαδικτυακές υπηρεσίες αυτόματης ενημέρωσης και υπηρεσίες παροχής πρόσβασης σε κινητές συσκευές.

3.2. Ενημέρωση των πληροφοριών

Η συνάρτηση checkForUpdates(String lastUpdate, int days) του Activity ActivityMap ελέγχει αν πρέπει να γίνει ενημέρωση των πληροφοριών και επιστρέφει αποτέλεσμα boolean. Αν επιστρέψει true πρέπει να γίνει ενημέρωση των δεδομένων. Η παράμετρος lastUpdate παίρνει ως τιμή την ημερομηνία τελευταίας ενημέρωσης της πληροφορίας. Οι ημερομηνίες τελευταίας ενημέρωσης όλων των φαρμακείων, των εφημερευόντων φαρμακείων, και των πρατηρίων καυσίμων για καθένα από τα 7 είδη καυσίμων βρίσκονται σε μεταβλητές τύπου String με μορφή “DD/MM/YYYY” στο SharedPreferences αρχείο Settings.xml. Αν κάποια από αυτές τις μεταβλητές δεν είναι καταχωρημένη στο αρχείο, συνεπάγεται ότι δεν έχουν ληφθεί ποτέ στο παρελθόν στοιχεία για την πληροφορία αυτής της μεταβλητής. Σε αυτήν την περίπτωση η παράμετρος lastUpdate παίρνει τιμή “” και η συνάρτηση checkForUpdates επιστρέφει “true”. Η παράμετρος days, όταν πρόκειται για ενημέρωση στοιχείων των εφημερευόντων φαρμακείων παίρνει τιμή “1”, καθώς πρόκειται για πληροφορία που αλλάζει κάθε μέρα. Όταν πρόκειται για ενημέρωση στοιχείων των πρατηρίων καυσίμων (για επιλεγμένο είδος) επίσης παίρνει τιμή “1”. Η πληροφορία υπάρχει μικρή πιθανότητα να αλλάξει παραπάνω από μια φορά την ημέρα, αλλά αποφασίστηκε να μην επιβαρυνθεί το σύστημα καθώς δίνεται στο χρήστη η δυνατότητα ενημέρωσης όποτε επιθυμεί. Όταν πρόκειται για ενημέρωση στοιχείων όλων των φαρμακείων παίρνει τιμή “7”, καθώς πρόκειται για σχεδόν στατική πληροφορία που σπάνια αλλάζει.

3.2.1. Ενημέρωση φαρμακείων

Η περίπτωση ενημέρωσης των φαρμακείων παρουσιάζεται στην Εικόνα 29. Αν ο χρήστης έχει επιλέξει “Φαρμακεία” στο κεντρικό μενού της εφαρμογής, η συνάρτηση checkForUpdates(String lastUpdate, int days) καλείται 2 φορές. Πρώτα για τον έλεγχο ενημέρωσης όλων των φαρμακείων. Η παράμετρος lastUpadate παίρνει ως τιμή την ημερομηνία τελευταίας ενημέρωσης όλων των φαρμακείων και η παράμετρος days την τιμή “7”.

- Αν αυτός ο έλεγχος επιστρέψει true, γίνεται έλεγχος της σύνδεσης με τη συνάρτηση checkConnection("http://www.fsserron.gr/") της κλάσης ServerConnection. Αν και ο έλεγχος σύνδεσης επιστρέψει true, ο πίνακας PHARMACIES στη βάση δεδομένων αδειάζει και καλείται η συνάρτηση downloadAllPharmacies() της κλάσης Pharmacies, όπου γίνεται λήψη των στοιχείων όλων των φαρμακείων και καταχώρηση τους στον πίνακα PHARMACIES. Μετά καλείται η συνάρτηση downloadOnDutyPharmacies() της ίδιας κλάσης, όπου γίνεται λήψη των στοιχείων των εφημερευόντων φαρμακείων και γίνεται ενημέρωση των εγγραφών των φαρμακείων που εφημερεύουν. Τέλος, οι μεταβλητές που κρατούν τις ημερομηνίες τελευταίας ενημέρωσης για όλα και για τα εφημερεύοντα φαρμακεία ενημερώνεται με την τρέχουσα ημερομηνία στο αρχείο Settings.xml.
- Αν ο έλεγχος ενημέρωσης όλων των φαρμακείων επιστρέψει false, καλείται η συνάρτηση checkForUpdates(String lastUpdate, int days) για έλεγχο ενημέρωσης των εφημερευόντων φαρμακείων. Η παράμετρος lastUpadate παίρνει ως τιμή την ημερομηνία τελευταίας ενημέρωσης των εφημερευόντων φαρμακείων και η παράμετρος days την τιμή "1".
    - Αν επιστρέψει true, γίνεται έλεγχος της σύνδεσης με τη συνάρτηση checkConnection("http://www.fsserron.gr/") της κλάσης ServerConnection. Αν και ο έλεγχος σύνδεσης επιστρέψει true, καλείται η συνάρτηση downloadOnDutyPharmacies() και γίνεται ενημέρωση μόνο των εφημερευόντων φαρμακείων. Τέλος, η μεταβλητή που κρατάει την ημερομηνία τελευταίας ενημέρωσης των εφημερευόντων φαρμακείων ενημερώνεται με την τρέχουσα ημερομηνία στο αρχείο Settings.xml.
    - Αν επιστρέψουν false και οι δύο έλεγχοι ενημέρωσης, τα δεδομένα των φαρμακείων είναι στη βάση, ενημερωμένα και έτοιμα για χρήση.

Αν οποιοσδήποτε έλεγχος σύνδεσης επιστρέψει false, ο χρήστης ενημερώνεται για την κατάσταση του δικτύου του, όλα τα φαρμακεία στη βάση δεδομένων τίθενται ως μη εφημερεύοντα και χρησιμοποιούνται τα δεδομένα που ήδη υπάρχουν στη βάση δεδομένων.

3.2.2. Ενημέρωση πρατηρίων καυσίμων

Η περίπτωση ενημέρωσης των πρατηρίων καυσίμων παρουσιάζεται στην Εικόνα 30. Όταν ο χρήστης έχει επιλέξει "Πρατήρια Καυσίμων" στο κεντρικό μενού της εφαρμογής, παραπέμπεται να διαλέξει είδος καυσίμου. Τότε καλείται η checkForUpdates(String lastUpdate, int days). Η παράμετρος lastUpdate έχει την ημερομηνία τελευταίας ενημέρωσης του επιλεγμένου είδους καυσίμου και η παράμετρος days έχει τιμή 1.

- Αν αυτός ο έλεγχος επιστρέψει true, γίνεται έλεγχος της σύνδεσης με την συνάρτηση checkConnection("http://www.fuelprices.gr/") της κλάσης ServerConnection. Αν και ο έλεγχος σύνδεσης επιστρέψει true, οι εγγραφές του πίνακα GASSTATIONS για το επιλεγμένο προϊόν σβήνονται και καλείται η συνάρτηση downloadGasStations(String prod) της κλάσης GasStations, όπου γίνεται λήψη των στοιχείων των πρατηρίων καυσίμων για το επιλεγμένο καύσιμο (prod) και καταχώρηση τους στον πίνακα GASSTATIONS. Τέλος, η μεταβλητή που κρατάει την ημερομηνία τελευταίας ενημέρωσης των πρατηρίων καυσίμων για το επιλεγμένο είδος ενημερώνεται με την τρέχουσα ημερομηνία στο αρχείο Settings.xml.
- Αν επιστρέψει false, τα δεδομένα των πρατηρίων για το επιλεγμένο είδος καυσίμου είναι στη βάση δεδομένων, ενημερωμένα και έτοιμα για χρήση.

Αν ο έλεγχος σύνδεσης επιστρέψει false, ο χρήστης ενημερώνεται για την κατάσταση του δικτύου του και χρησιμοποιούνται τα δεδομένα που ήδη υπάρχουν στη βάση δεδομένων.

3.3. Προβολή σημείων στο χάρτη

Η διαχείριση των σημείων στο χάρτη πραγματοποιείται από την κλάση CustomPinpoint. Για κάθε λίστα σημείων (OverlayItems) δημιουργείται ένα αντικείμενο της κλάσης αυτής. Όλα τα σημεία μιας λίστα μοιράζονται το ίδιο εικονίδιο (marker). Στην παρούσα εργασία χρησιμοποιούνται τέσσερις λίστες σημείων. Μια για την τοποθεσία του χρήστη (CustomPinpoint::myLocation), μια για τα εφημερεύοντα φαρμακεία (CustomPinpoint::farm_ef), μια για τα υπόλοιπα φαρμακεία (CustomPinpoint::farm) και μια για τα πρατήρια καυσίμων (CustomPinpoint::gas). Σε κάθε λίστα προστίθενται όσα σημεία χρειάζονται, με χρήση της συνάρτησης insertPinpoint(OverlayItem item) της κλάσης CustomPinpoint. Στη συνάρτηση αυτή με την εντολή add(item) το αντικείμενο item προστίθεται στη λίστα και με την εντολή populate() γίνεται αυτόματα όλη η επεξεργασία που απαιτείται για την διαχείριση του νέου αντικειμένου ItemizedOverlay. Αφού το αντικείμενο OverlayItem προστεθεί στην λίστα, μπορεί να γίνει αναφορά σε αυτό μέσω της θέσης (index) του στη λίστα.

Το στοιχείο του χάρτη (MapView) διαθέτει μία λίστα τύπου List<Overlay>, η οποία περιέχει όλα τα αντικείμενα που εμφανίζονται στο χάρτη και δημιουργείται αυτόματα από το MapActivity κατά τον ορισμό του στοιχείου του χάρτη. Η λίστα (List<Overlay>::overlayList) μπορεί να ληφθεί από το αντικείμενο του χάρτη, με χρήση της εντολής getOverlays(). Στη συνέχεια προστίθενται σε αυτήν νέα αντικείμενα προς εμφάνιση, όπως οι τέσσερις λίστες CustomPinpoint, που αναφέρθηκαν

προηγουμένως, με χρήση της εντολής add(Overlay object). Τα αντικείμενα που προστίθενται στη λίστα αυτόματα προβάλλεται στο χάρτη. Φυσικά, μετά την προσθήκη των αντικειμένων πρέπει να γίνει ανανέωση του χάρτη, με χρήση της εντολής invalidate(), ώστε να εμφανιστούν τα νέα στοιχεία.

Συγκεκριμένα, στις συναρτήσεις showPharmacies() της κλάσης Pharmacies και showGasStations() της κλάσης GasStations, πραγματοποιείται η προβολή των φαρμακείων και των πρατηρίων καυσίμων αντίστοιχα. Τα σημεία (OverlayItems) εισάγονται στα αντίστοιχα αντικείμενα CustomPinpoint, τα αντικείμενα CustomPinpoint προστίθενται στη λίστα List<Overlay> του χάρτη και ο χάρτης ενημερώνεται.

3.4. Ανανέωση του χάρτη από το χρήστη

Η εφαρμογή δίνει στο χρήστη τη δυνατότητα ανανέωσης του χάρτη από το αντίστοιχο κουμπί του UI (layout) του ActivityMap. Η ανανέωση περιλαμβάνει λήψη των πληροφοριών εκ νέου και προβολή τους στο χάρτη.

3.4.1. Ανανέωση των φαρμακείων

Στην περίπτωση ανανέωσης των φαρμακείων, προβάλλεται παράθυρο με επιλογή ανανέωσης των εφημερευόντων ή όλων των φαρμακείων.

- Αν ο χρήστης επιλέξει ανανέωση των εφημερευόντων τότε διαγράφεται η ημερομηνία τελευταίας ενημέρωσης των εφημερευόντων φαρμακείων από το SharedPreferences αρχείο Settings.xml.
- Αν ο χρήστης επιλέξει ανανέωση όλων των φαρμακείων τότε διαγράφεται η ημερομηνία τελευταίας ενημέρωσης όλων των φαρμακείων, καθώς και η ημερομηνία τελευταίας ενημέρωσης των εφημερευόντων από το SharedPreferences αρχείο Settings.xml.

Στη συνέχεια καλείται η συνάρτηση getPharmacies(), όπου επαναλαμβάνεται η διαδικασία ελέγχου ενημέρωσης και λήψης των στοιχείων που απαιτούν ενημέρωση και η προβολή των φαρμακείων εκ νέου στο χάρτη.

3.4.2. Ανανέωση των πρατηρίων καυσίμων

Στην περίπτωση ανανέωσης των πρατηρίων καυσίμων, διαγράφεται η ημερομηνία τελευταίας ενημέρωσης των πρατηρίων καυσίμων για το επιλεγμένο είδος καυσίμου, από το SharedPreferences αρχείο Settings.xml. Στη συνέχεια καλείται η συνάρτηση getGasStations(), όπου επαναλαμβάνεται η διαδικασία ελέγχου ενημέρωσης και λήψης των στοιχείων που απαιτούν ενημέρωση και η προβολή των πρατηρίων καυσίμων εκ νέου στο χάρτη.

3.5. Προβολή οδηγιών διαδρομής

Οι οδηγίες διαδρομής από ένα σημείο σε ένα άλλο παρέχονται μέσω της υπηρεσίας Google Maps. Φυσικά, προϋπόθεση για τη χρήση οδηγιών είναι η ύπαρξη της θέσης του χρήστη. Αν κατά τη εκτέλεση της εφαρμογής δεν είναι δυνατός ο εντοπισμός της θέσης του χρήστη, δεν είναι δυνατή η εμφάνιση του δρομολογίου.

Η διαδικασία λήψης οδηγιών ενός δρομολογίου επιφέρει επιβάρυνση στο σύστημα, ανάλογη της ποιότητας της σύνδεσης της συσκευής στο διαδίκτυο, αλλά και της απόστασης του δρομολογίου λόγω του όγκου των δεδομένων. Για την καλύτερη λειτουργία της εφαρμογής δόθηκαν οι παρακάτω λύσεις:

- Δημιουργία ενός μενού "Ρυθμίσεις Δρομολογίου", όπου ο χρήστης θα μπορεί να ρυθμίζει αν επιθυμεί την εμφάνιση δρομολογίου, αλλά και το μέσο μεταφοράς που επιθυμεί. Τα δεδομένα αυτά αποθηκεύονται στο SharedPreferences αρχείο Settings.xml.
- Απενεργοποίηση της λήψης και προβολής οδηγιών αν η θέση του χρήστη απέχει παραπάνω από 50 km. Ο έλεγχος αυτός πραγματοποιείται με χρήση της συνάρτησης calcDistance(Location dest), που περιγράφεται στην παράγραφο "Υπολογισμός απόστασης".

Η συνάρτηση getPath(GeoPoint src, GeoPoint dest, int color, MapView map) της κλάσης ServerConnection, συντονίζει την λήψη των οδηγιών και την εμφάνιση του δρομολογίου, όπως περιγράφονται παρακάτω.

3.5.1. Λήψη και αποκωδικοποίηση οδηγιών

Η λήψη των οδηγιών πραγματοποιείται στην υποκλάση getPath με επέκταση (extends) AsyncTask, της κλάσης ServerConnection. Αρχικά γίνεται η σύνθεση ενός url της μορφής "http://maps.googleapis.com/maps/api/directions/json?origin=«x1»,«x2» &destination=«y1»,«y2»&sensor=false&mode=«z»", όπου «x1» και «x2» είναι το γεωγραφικό μήκος και πλάτος του σημείου αφετηρίας, «y1» και «y2» το γεωγραφικό μήκος και πλάτος του σημείου προορισμού και «z» ο τρόπος μεταφοράς. Τα γεωγραφικά μήκη και πλάτη εκφράζονται με χρήση δεκαδικών αριθμών, ενώ ο τρόπος μεταφοράς περιγράφεται με τις λέξεις walking, driving ή biking. Πχ http://maps.googleapis.com/maps/api/directions/json?origin=41.0871911,23.5384083&destination=41.0772415,23.5388108&sensor=false&mode=driving. Η σελίδα αυτή περιέχει ένα KML αρχείο με κωδικοποιημένες πληροφορίες. Στην συνέχεια λαμβάνεται ο πηγαίος κώδικας της σελίδας αυτής και οι πληροφορίες αποκωδικοποιούνται με τη συνάρτηση decodePoly(String encoded). Η συνάρτηση αυτή επιστρέφει μία λίστα σημείων (List<GeoPoint>) τα οποία αν ενωθούν με μια γραμμή σχηματίζουν το δρομολόγιο από τη θέση του χρήστη στο επιλεγμένο κατάστημα.

3.5.2. Δημιουργία και προβολή του δρομολογίου στο χάρτη

Ένα νέο αντικείμενο της κλάσης RoutePathOverlay, δημιουργείται με χρήση του constructor RoutePathOverlay(List<GeoPoint> points). Η κλάση αυτή με χρήση των συναρτήσεων drawOval(Canvas canvas, Paint paint, Point point), draw(Canvas canvas, MapView mapView, boolean shadow, long when), draw(Canvas canvas, MapView mapView, boolean shadow, long when) και setDrawStartEnd(boolean markStartEnd) δημιουργεί αυτόματα ένα νέο δρομολόγιο βάσει των σημείων (List<GeoPoint>::points) που δόθηκαν ως παράμετροι στον constructor.

Το αντικείμενο RoutePathOverlay, προστίθεται στη λίστα αντικειμένων (List<Overlay>::overlayList) που απεικονίζονται στο χάρτη, με χρήση της συνάρτησης add(Overlay object). Στη συνέχεια γίνεται ανανέωση του χάρτη με την εντολή invalidate().

3.6. Υπολογισμός απόστασης

Η ακριβής απόσταση περιλαμβάνεται στις οδηγίες διαδρομής και εξαρτάται από το μέσο μεταφοράς. Κατά την προβολή ενός δρομολογίου ο χρήστης ενημερώνεται με μήνυμα (Toast) για την απόσταση του καταστήματος που έχει επιλέξει. Ωστόσο, υπολογισμός της απόστασης του χρήστη από κάποιο κατάστημα πρέπει να γίνεται και κατά την εμφάνιση των φαρμακείων ή των πρατηρίων καυσίμων σε λίστα και κατά την προβολή των στοιχείων ενός καταστήματος. Η λήψη οδηγιών όμως προκαλεί επιβάρυνση και καθυστέρηση στο σύστημα. Ειδικά κατά την εμφάνιση των φαρμακείων ή των πρατηρίων καυσίμων σε λίστα, όπου η προεπιλεγμένη ταξινόμηση είναι ανά απόσταση, θα προκαλούσε πολύ μεγάλη επιβάρυνση και καθυστέρηση στη λειτουργία του συστήματος. Για την επίλυση αυτού του προβλήματος, αποφασίστηκε ο υπολογισμός της απόστασης με λήψη οδηγιών να γίνεται μόνο κατά την προβολή δρομολογίου.

Κατά την εμφάνιση των φαρμακείων ή των πρατηρίων καυσίμων σε λίστα και κατά την προβολή των στοιχείων ενός καταστήματος, υπολογισμός της απόστασης να γίνεται με χρήση της συνάρτησης distanceTo(Location dest). Η συνάρτηση αυτή επιστρέφει την κατά προσέγγιση απόσταση σε μέτρα μεταξύ αυτής της θέσης (Location) για την οποία κλήθηκε η συνάρτηση και της τοποθεσίας (Location dest) που δόθηκε ως παράμετρος.

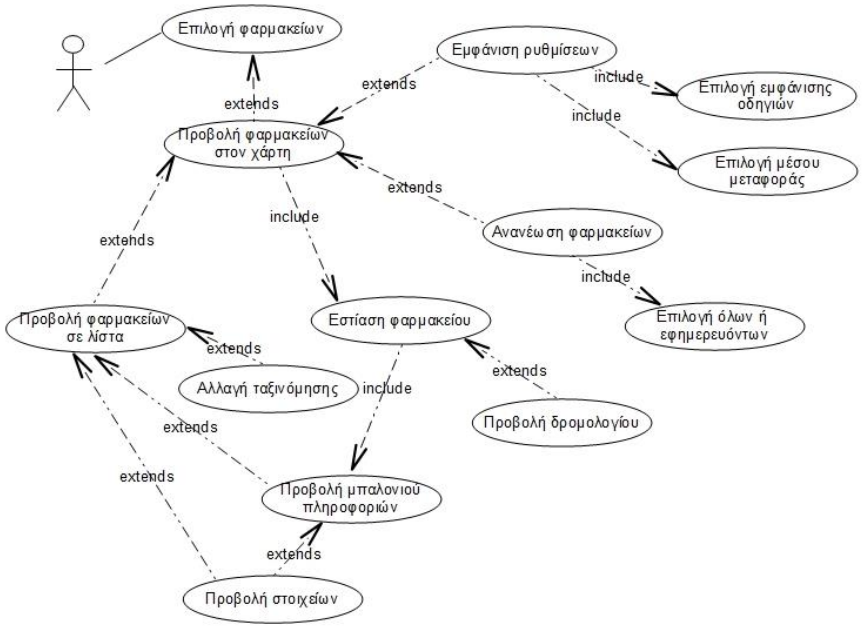
4. Σενάρια χρήσης της εφαρμογής

Στην παρούσα εφαρμογή διακρίνονται δύο βασικές περιπτώσεις χρήσης, όπου η κάθε μία οδηγεί σε αρκετές εναλλακτικές ροές:

4.1. Προβολή φαρμακείων

1. Επιλογή “Φαρμακεία” στο κεντρικό μενού (Εικόνα 32)
   1.1 Εστίαση σε ένα κατάστημα

1.1.1. Προβολή δρομολογίου, αν είναι ενεργοποιημένη η επιλογή οδηγιών δρομολογίου
1.1.2. Προβολή μπαλονιού πληροφοριών καταστήματος
1.1.2.1. Επιλογή του μπαλονιού πληροφοριών και προβολή στοιχείων του φαρμακείου
1.2. Προβολή ρυθμίσεων
1.2.1. Επιλογή ενεργοποίησης οδηγιών δρομολογίου
1.2.2. Επιλογή μέσου μεταφοράς
1.3. Ανανέωση φαρμακείων
1.3.1. Επιλογή ανανέωσης όλων ή εφημερευόντων φαρμακείων
1.4. Εμφάνιση φαρμακείων σε λίστα
1.4.1. Αλλαγή ταξινόμησης.
1.4.2. Επιλογή ενός καταστήματος και επιστροφή στο χάρτη με εστίαση στο επιλεγμένο κατάστημα, ή προβολή των στοιχείων του καταστήματος αν αυτό δεν εμφανίζεται στο χάρτη

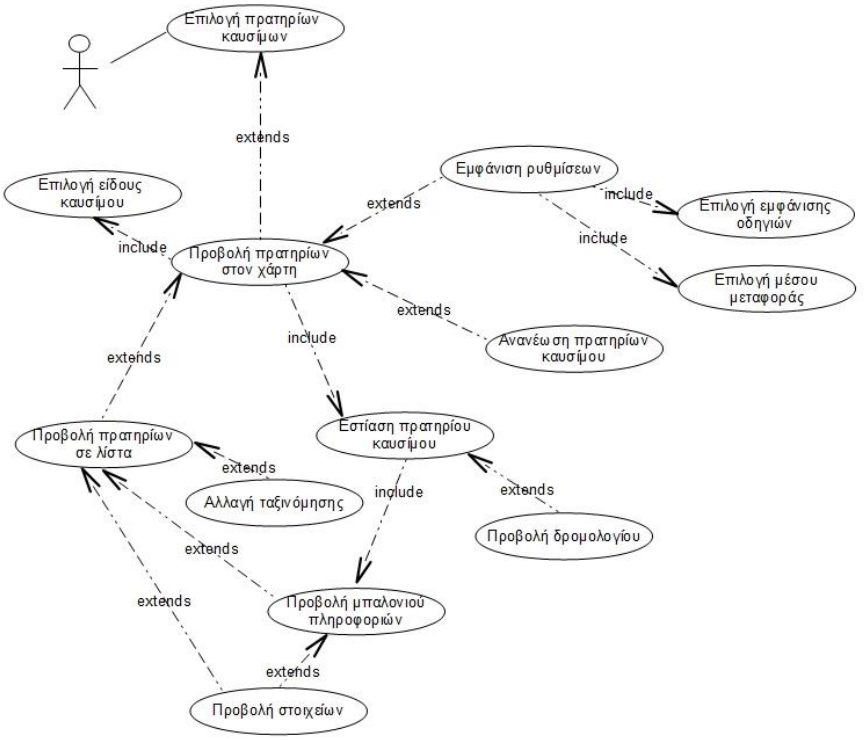


Εικόνα 32. Διάγραμμα σεναρίου χρήσης: Προβολή φαρμακείων

4.2. Προβολή πρατηρίων καυσίμων

1. Επιλογή "Πρατήρια Καυσίμων" στο κεντρικό μενού και επιλογή είδους καυσίμου (Εικόνα 33)
1.1 Εστίαση σε ένα κατάστημα
1.1.1. Προβολή δρομολογίου, αν είναι ενεργοποιημένη η επιλογή οδηγιών δρομολογίου

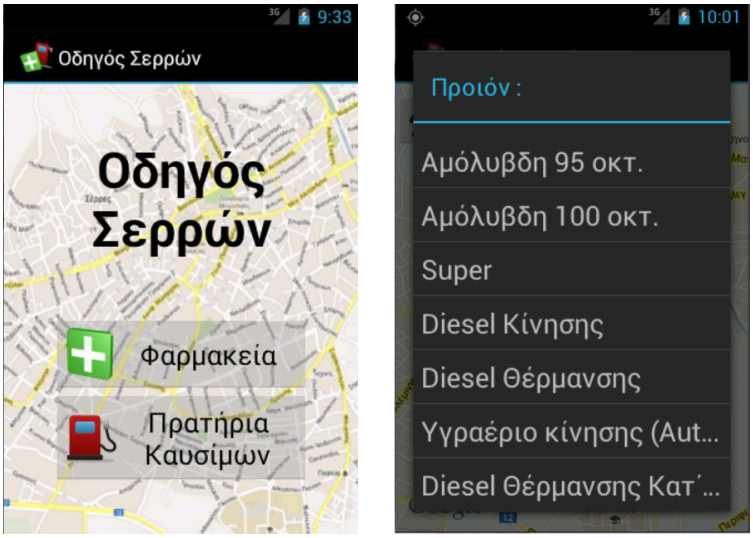
1.1.2. Προβολή μπαλονιού πληροφοριών καταστήματος
1.1.2.1. Επιλογή του μπαλονιού πληροφοριών και προβολή στοιχείων του πρατηρίου
1.2. Προβολή ρυθμίσεων
1.2.1. Επιλογή ενεργοποίησης οδηγιών δρομολογίου
1.2.2. Επιλογή μέσου μεταφοράς
1.3. Ανανέωση πρατηρίων
1.4. Εμφάνιση πρατηρίων καυσίμων σε λίστα
1.4.1. Αλλαγή ταξινόμησης
1.4.2. Επιλογή ενός καταστήματος και επιστροφή στο χάρτη με εστίαση στο επιλεγμένο κατάστημα, ή προβολή των στοιχείων του καταστήματος αν αυτό δεν εμφανίζεται στο χάρτη



Εικόνα 33. Διάγραμμα σεναρίου χρήσης: Προβολή πρατηρίων καυσίμων

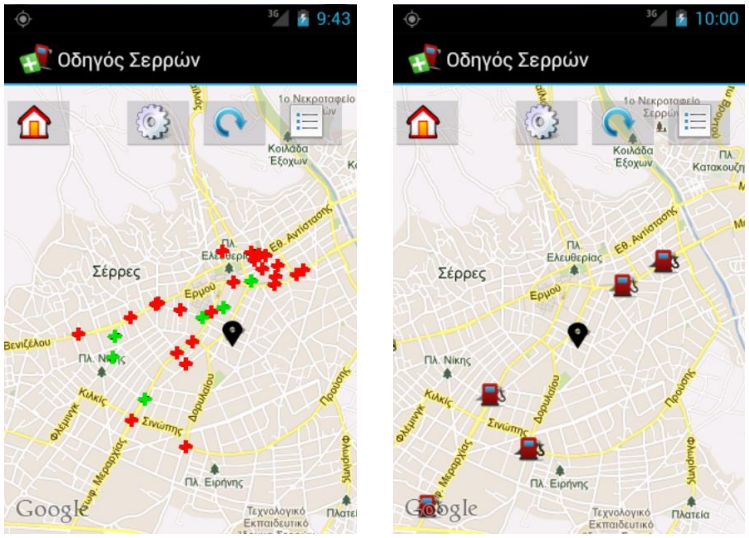
4.3. Στιγμιότυπα της εφαρμογής

Στο κεντρικό μενού της εφαρμογής ο χρήστης μπορεί να επιλέξει αν αναζητεί πληροφορίες για φαρμακεία ή πρατήρια καυσίμων (Εικόνα 34). Στην ίδια εικόνα (δεξιά) εμφανίζεται το μενού επιλογής είδους καυσίμου αν επιλεγεί η παρουσίαση πρατηρίων καυσίμων.



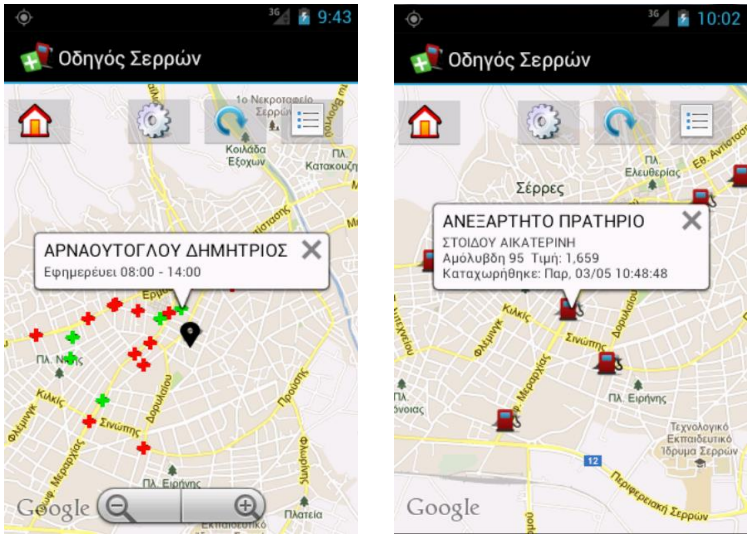
Εικόνα 34. Αριστερά: Το κεντρικό μενού της εφαρμογής. Δεξιά: Επιλογή είδους καυσίμου μετά την επιλογή "Πρατήρια Καυσίμων" από το κεντρικό μενού

Στην Εικόνα 35 παρουσιάζονται οι οθόνες που προκύπτουν από την επιλογή "Φαρμακεία" στο κεντρικό μενού και από την επιλογή είδους καυσίμου, αντίστοιχα.



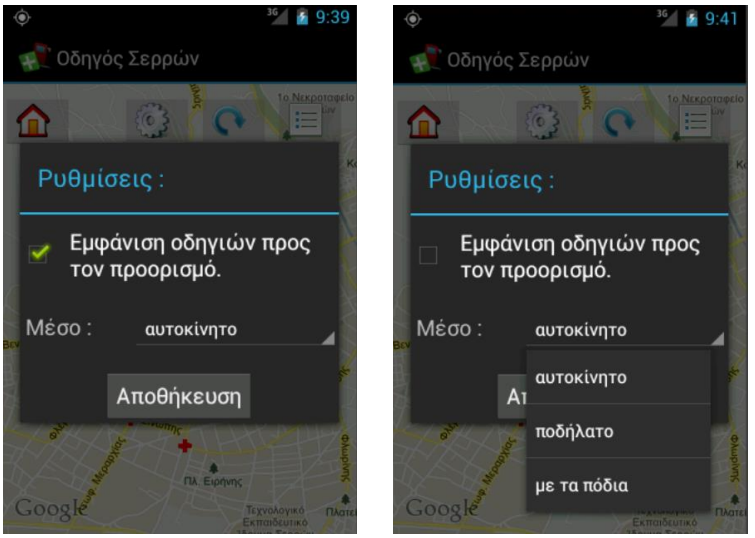
Εικόνα 35. Προβολή του χάρτη με την τοποθεσία του χρήστη και αριστερά τα φαρμακεία, δεξιά τα πρατήρια καυσίμων

Κατά την εστίαση σε κάποιο αντικείμενο του χάρτη προβάλλεται το μπαλόνι πληροφοριών του, όπως φαίνεται στην Εικόνα 36.



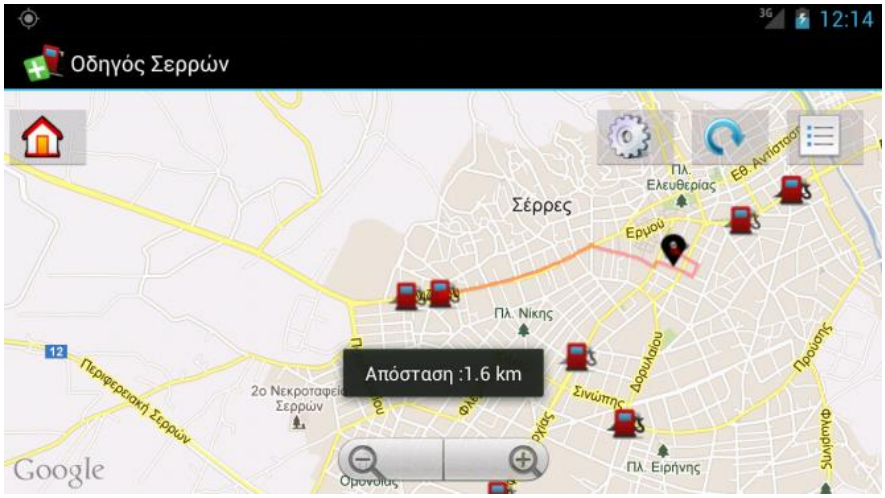
Εικόνα 36. Εστίαση σε ένα αντικείμενο του χάρτη

Στην Εικόνα 37 παρουσιάζονται οι ρυθμίσεις σχετικά με την παροχή πληροφοριών διαδρομών.



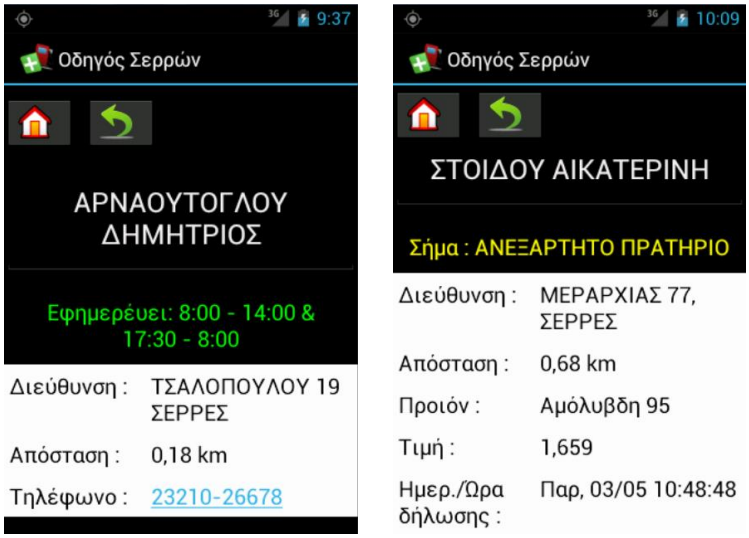
Εικόνα 37. Ρυθμίσεις πληροφόρησης διαδρομής

Αν η εμφάνιση οδηγιών προς τον προορισμό είναι ενεργοποιημένη, κατά την εστίαση σε ένα αντικείμενο εμφανίζεται η διαδρομή προς αυτό, όπως φαίνεται στην Εικόνα 38.



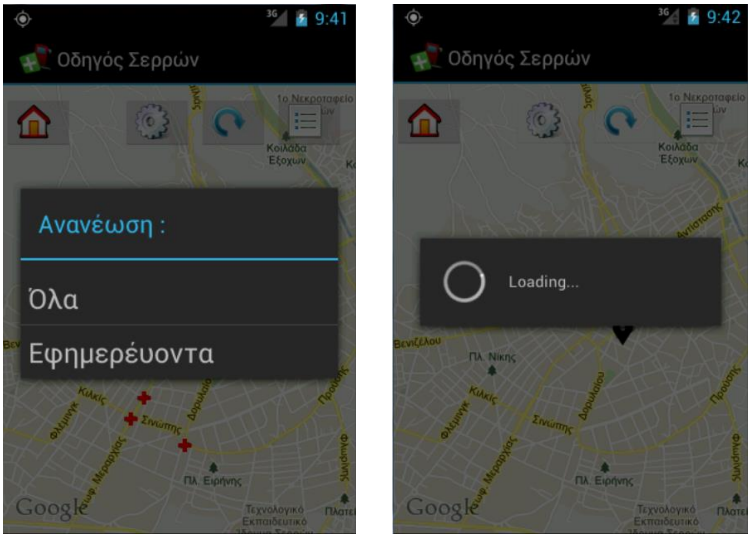
Εικόνα 38. Εμφάνιση οδηγιών προς ένα επιλεγμένο κατάστημα

Μετά από επιλογή του μπαλονιού πληροφοριών, εμφανίζονται τα πλήρη στοιχεία του επιλεγμένου καταστήματος, όπως φαίνεται στην Εικόνα 39.



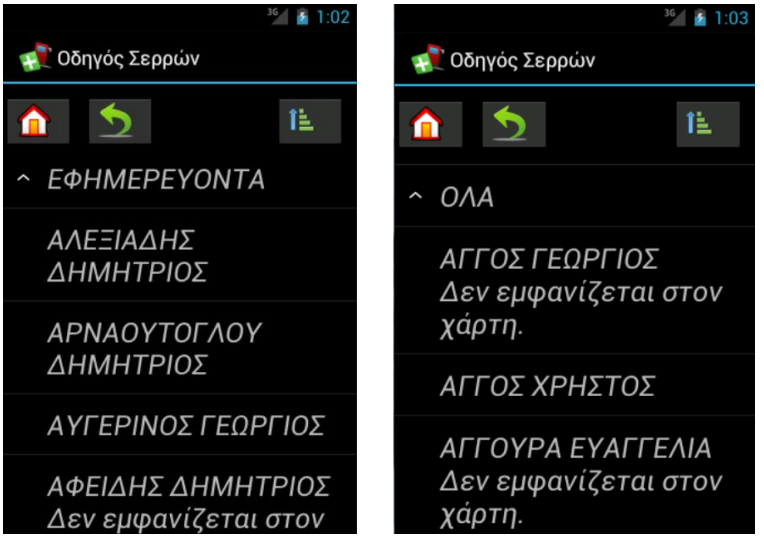
Εικόνα 39. Προβολή στοιχείων ενός επιλεγμένου σημείου

Κατά την προβολή του χάρτη, ο χρήστης μπορεί να επιλέξει ανανέωση. Στην περίπτωση που έχει επιλέξει φαρμακεία, δίνεται η δυνατότητα ανανέωσης όλων ή μόνο των εφημερευόντων (Εικόνα 40).



Εικόνα 40. Αριστερά :Μενού ανανέωσης Φαρμακείων. Δεξιά: Αναμονή κατά την ανανέωση

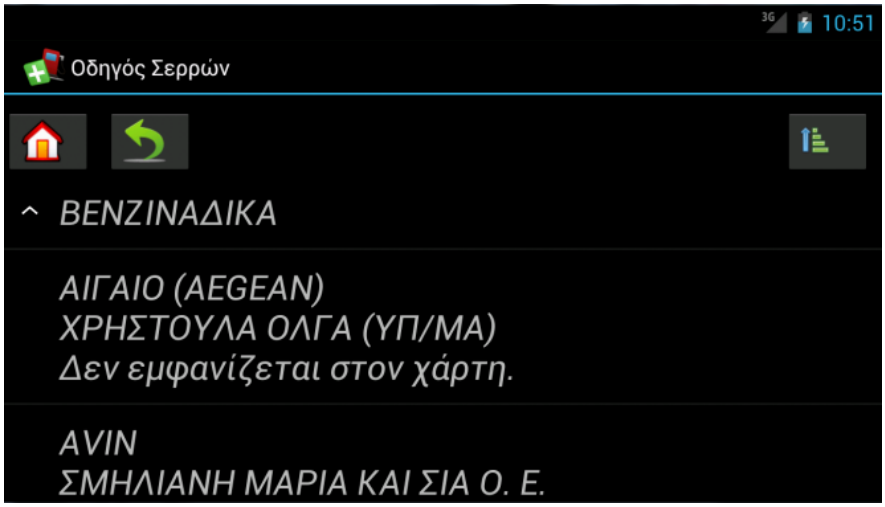
Κατά την επιλογή ενός φαρμακείου που υπάρχει στο χάρτη, προβάλλεται ο χάρτης με εστίαση στο συγκεκριμένο κατάστημα. Κατά την επιλογή ενός φαρμακείου που δεν υπάρχει στο χάρτη, προβάλλονται τα πλήρη στοιχεία του καταστήματος. Η προβολή φαρμακείων σε λίστα παρουσιάζεται στην Εικόνα 41.



Εικόνα 41. Προβολή λίστας των φαρμακείων

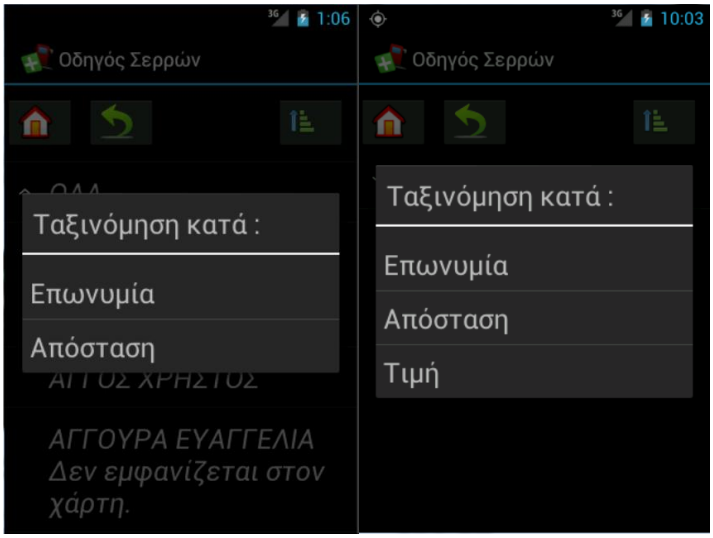
Κατά την επιλογή ενός πρατηρίου καυσίμων που υπάρχει στο χάρτη, προβάλλεται ο χάρτης με εστίαση στο συγκεκριμένο κατάστημα. Κατά

την επιλογή ενός πρατηρίου καυσίμων που δεν υπάρχει στο χάρτη, προβάλλονται τα πλήρη στοιχεία του καταστήματος. Η προβολή πρατηρίων καυσίμων σε λίστα φαίνεται στην Εικόνα 42.



Εικόνα 42. Προβολή λίστας των πρατηρίων καυσίμων

Στην Εικόνα 43 παρουσιάζονται οι επιλογές ταξινόμησης στη λίστα των φαρμακείων και πρατηρίων καυσίμων αντίστοιχα.



Εικόνα 43. Επιλογές ταξινόμησης στη λίστα των φαρμακείων αριστερά και των πρατηρίων καυσίμων δεξιά

5. Μελλοντικές επεκτάσεις της εφαρμογής

Η παρούσα εργασία είναι δυνατό να επεκταθεί μελλοντικά ως προς τους παρακάτω τομείς:

- Παροχή πληροφοριών για νοσοκομεία, κέντρα υγείας και ιατρούς των Σερρών, μέσω του site http://www.fsserron.gr/
- Εμφάνιση σήματος (logo) των πρατηρίων καυσίμων ως εικονίδιο (marker) στο χάρτη
- Παροχή πληροφοριών των πρατηρίων καυσίμων για όλη την Ελλάδα μέσω του site http://www.fuelprices.gr/
- Πλήρη ενοποίηση σε πανελλαδικό σύστημα παροχής πληροφόρησης με έρευνα και ενσωμάτωση όλων των έγκυρων πηγών πληροφόρησης για κάθε Νομό της χώρας για τις πληροφορίες υγείας και καυσίμων
- Προσθήκη και άλλων τύπων πληροφορίας πέραν των ιατροφαρμακευτικών και καυσίμων

Βιβλιογραφία

[1] Android, the world's most popular mobile platform, online στη διεύθυνση http://developer.android.com/about/index.html, τελευταία επίσκεψη: 26 Απρ. 2013.

[2] Number of available Android applications, online στη διεύθυνση http://www.appbrain.com/stats/number-of-android-apps, τελευταία επίσκεψη: 26 Απρ. 2013.

[3] Google Play – Find Pharmacies, online στη διεύθυνση https://play.google.com/store/apps/details?id=com.elbatrop.pharmacies&feature=search_result, τελευταία επίσκεψη: 26 Απρ. 2013.

[4] Google Play – Fuel Prices in Greece, online στη διεύθυνση https://play.google.com/store/apps/details?id=gr.x4real.fuelprices.android&feature=search_result#?t=W251bGwsMSwxLDEsImdyLng0cmVhbC5mdWVscHJpY2VzLmFuZHJvaWQiXQ.., τελευταία επίσκεψη: 26 Απρ. 2013.

[5] iTunes – LiveMap, online στη διεύθυνση https://itunes.apple.com/us/app/livemap/id415607850?mt=8, τελευταία επίσκεψη: 26 Απρ. 2013.

[6] Windows Phone Εφαρμογές – Pharmacy Locator, online στη διεύθυνση http://www.windowsphone.com/el-gr/store/app/pharmacy-locator/96b9853c-2470-41c7-914f-2e897dce8816, τελευταία επίσκεψη: 26 Απρ. 2013.

[7] Window Phone Εφαρμογές – Petrol Price Finder UK, online στη διεύθυνση http://www.windowsphone.com/el-gr/store/app/petrol-price-finder-uk/88957598-9f2c-e011-854c-00237de2db9e, τελευταία επίσκεψη: 26 Απρ. 2013.

[8] Android (operating system), online στη διεύθυνση http://en.wikipedia.org/wiki/Android_(operating_system), τελευταία επίσκεψη: 24 Απρ. 2013.

[9] HTC T-Mobile G1 Specifications, online στη διεύθυνση http://www.gsmarena.com/t_mobile_g1-2533.php, τελευταία επίσκεψη: 24 Απρ. 2013.

[10] HTC Google Nexus One Specifications, online στη διεύθυνση http://www.gsmarena.com/htc_google_nexus_one-3069.php, τελευταία επίσκεψη: 24 Απρ. 2013.

[11] Android version history, online στη διεύθυνση http://en.wikipedia.org/wiki/Android_version_history, τελευταία επίσκεψη: 24 Απρ. 2013.

[12] Εισαγωγή στην Ανάπτυξη Android Εφαρμογών, online στη διεύθυνση http://www.dga.gr/web/publications/files/android.pdf, τελευταία επίσκεψη: 24 Απρ. 2013.

[13] Mobile Devices - An Introduction to the Android Operating Environment, online στη διεύθυνση http://www.dhtusa.com/media/AndroidInternals.pdf, τελευταία επίσκεψη: 24 Απρ. 2013.

[14] App Framework, online στη διεύθυνση http://developer.android.com/about/versions/index.html, τελευταία επίσκεψη: 24 Απρ. 2013.

[15] Location Manager, online στη διεύθυνση http://developer.android.com/reference/android/location/LocationManager.html, τελευταία επίσκεψη: 24 Απρ. 2013.

[16] Activity, online στη διεύθυνση http://developer.android.com/reference/android/app/Activity.html, τελευταία επίσκεψη: 24 Απρ. 2013.

[17] Managing Projects, online στη διεύθυνση http://developer.android.com/tools/projects/index.html, τελευταία επίσκεψη: 24 Απρ. 2013.

[18] The Android Manifest.xml File, online στη διεύθυνση http://developer.android.com/guide/topics/manifest/manifest-intro.html, τελευταία επίσκεψη: 24 Απρ. 2013.

[19] Application Fundamentals, online στη διεύθυνση http://developer.android.com/guide/components/fundamentals.html, τελευταία επίσκεψη: 24 Απρ. 2013.

[20] Supporting Different Devices, online στη διεύθυνση http://developer.android.com/training/basics/supporting-devices/languages.html, τελευταία επίσκεψη: 29 Απρ. 2013.

[21] Localization, online στη διεύθυνση http://developer.android.com/guide/topics/resources/localization.html, τελευταία επίσκεψη: 29 Απρ. 2013.

[22] Supporting Different Languages, online στη διεύθυνση http://developer.android.com/training/basics/supporting-devices/languages.html, τελευταία επίσκεψη: 29 Απρ. 2013.

[23] Designing for Multiple Screens, online στη διεύθυνση http://developer.android.com/training/multiscreen/index.html τελευταία επίσκεψη: 30 Απρ. 2013.

[24] Supporting Different Screens, online στη διεύθυνση http://developer.android.com/training/basics/supporting-devices/screens.html, τελευταία επίσκεψη: 29 Απρ. 2013.

[25] Supporting Multiple Screens, online στη διεύθυνση http://developer.android.com/guide/practices/screens_support.html, τελευταία επίσκεψη: 30 Απρ. 2013.

[26] Dashboards, online στη διεύθυνση http://developer.android.com/about/dashboards/index.html, τελευταία επίσκεψη: 30 Απρ. 2013.

[27] Supporting Different Densities, online στη διεύθυνση http://developer.android.com/training/multiscreen/screendensities.html, τελευταία επίσκεψη: 30 Απρ. 2013.

[28] Supporting Different Screen Sizes, online στη διεύθυνση http://developer.android.com/training/multiscreen/screensizes.html, τελευταία επίσκεψη: 30 Απρ. 2013.

[29] Supporting Different Platform Versions, online στη διεύθυνση http://developer.android.com/training/basics/supporting-devices/platforms.html, τελευταία επίσκεψη: 30 Απρ. 2013.

[30] Singing Your Applications, online στη διεύθυνση http://developer.android.com/tools/publishing/app-signing.html, τελευταία επίσκεψη: 24 Απρ. 2013.

[31] Building and Running, online στη διεύθυνση http://developer.android.com/tools/building/index.html, τελευταία επίσκεψη: 24 Απρ. 2013.

[32] Storage Options, online στη διεύθυνση http://developer.android.com/guide/topics/data/data-storage.html#db, τελευταία επίσκεψη: 2 Μαίου 2013.

[33] SQLiteOpenHelper, online στη διεύθυνση http://developer.android.com/reference/android/database/sqlite/SQLiteOpenHelper.html, τελευταία επίσκεψη: 2 Μαίου 2013.

[34] Get the Android SDK, online στη διεύθυνση http://developer.android.com/sdk/index.html, τελευταία επίσκεψη: 26 Απρ. 2013.

[35] Setting up the ADT Bundle, online στη διεύθυνση http://developer.android.com/sdk/installing/bundle.html, τελευταία επίσκεψη: 26 Απρ. 2013.

[36] Setting up an Existing IDE, online στη διεύθυνση http://developer.android.com/sdk/installing/index.html, τελευταία επίσκεψη: 26 Απρ. 2013.

[37] Installing the Eclipse Plugin, online στη διεύθυνση http://developer.android.com/sdk/installing/installing-adt.html, τελευταία επίσκεψη: 26 Απρ. 2013.

[38] Obtaining a Google Maps API v1 Key, online στη διεύθυνση https://developers.google.com/maps/documentation/android/v1/mapkey, τελευταία επίσκεψη: 29 Μαρ. 2013.

[39] Obtaining a Google Maps API v1 Key, online στη διεύθυνση https://developers.google.com/maps/documentation/android/v1/maps-api-signup, τελευταία επίσκεψη: 29 Μαρ. 2013.

[40] Class Overlay, online στη διεύθυνση https://developers.google.com/maps/documentation/android/v1/reference/com/google/android/maps/Overlay, τελευταία επίσκεψη: 25 Απρ. 2013.

[41] Android MapView Balloons, online στη διεύθυνση http://jgilfelt.github.io/android-mapviewballoons/, τελευταία επίσκεψη: 25 Απρ. 2013.

[42] Class ItemizedOverlay<Item extends OverlayItem>, online στη διεύθυνση https://developers.google.com/maps/documentation/android/v1/reference/com/google/android/maps/ItemizedOverlay, τελευταία επίσκεψη: 25 Απρ. 2013.

[43] Java Tutorial: The “extend” Keyword, online στη διεύθυνση http://xahlee.info/java-a-day/extend.html, τελευταία επίσκεψη: 25 Απρ. 2013.

[44] How much better is GPS over Wi-Fi positioning? http://gigaom.com/2012/08/17/how-much-better-is-gps-over-wi-fi-positioning-yelp-knows/, τελευταίαεπίσκεψη: 12 Μαιου 2013.

Παράρτημα Α – Πηγαίος κώδικας

Α.1. Κλάσεις Activities

Α.1.1. Main

```
/*Class Main handles the main menu*/
public class Main extends Activity implements OnClickListener {

	/*Function onCreate intializes the components*/
	@Override
	protected void onCreate(Bundle savedInstanceState) {
		// TODO Auto-generated method stub
		super.onCreate(savedInstanceState);
		setContentView(R.layout.main);
		Button pharmacy = (Button) findViewBy-
Id(R.id.pharmacyBt);
		pharmacy.setOnClickListener(this);
		Button gasstation = (Button) findViewBy-
Id(R.id.gasstationBt);
		gasstation.setOnClickListener(this);
	}

	/*Function onClick opens the ActivityMap and
	 * passes the user's choice (pharmacies or gas stations)*/
	public void onClick(View v) {
		// TODO Auto-generated method stub
		Intent map = new In-
tent("com.anaskous.serresGuide.MAP");
		Bundle b = new Bundle();
		switch(v.getId()){
		case R.id.pharmacyBt:
			b.putInt("choice", 0);
			b.putString("hasFocushedOverlay", "");
			map.putExtras(b);
			startActivity(map);
			break;
		case R.id.gasstationBt:
			b.putInt("choice", 1);
			b.putString("hasFocushedOverlay", "");
			map.putExtras(b);
			startActivity(map);
			break;
		}
	}
}
```

A.1.2. ActivityMap

```
/*Class ActivityMap handles the map*/
public class ActivityMap extends MapActivity implements Location-
Listener,
		OnClickListener, OnItemClickListener, OnFocusChange-
Listener {
	MapView map;
	MapController controller;
	int x, y;
	List<Overlay> overlayList;
	LocationManager lm;
	String towers, prod = "", hasFocushedOverlay, focushedIn-
dex="", focushedIsOnDuty="";
	int lat, longi;
	OverlayItem overlayItem;
	CustomPinpoint myLocation;
	GeoPoint ourLocation;
	List<List<String>> farmakeia, efimereuonta, gasStations;
	Reader r;
	CustomPinpoint farm, farm_ef, gas;
	DBManager db = new DBManager(this);
	private ProgressDialog dialog;
	Drawable cross_green, cross_red, myloc, gas_station;
	Button btList, btRefresh, btSetts, btSettsOk, btHome;
	boolean isOnline;
	Pharmacies dp;
	GasStations dg;
	ServerConnection sc;
	SharedPreferences prefs;
	boolean update, update1;
	int choice;
	AlertDialog prodAlert, pharmAlert, routeAlert;
	RoutePathOverlay rpo;
	Spinner routeSp;
	CheckBox routeCb;

	/*Function onCreate contains code for variables/components
	 * intialization and shows the users location and the stores
on map*/
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.map);
		BroadcastReceiver myReceiver = new BroadcastReceiv-
er() {
			@Override
			public void onReceive(Context context, Intent
intent) {
				Bundle getBundle = intent.getExtras();
				String id = getBundle.getString("id");
				dataShow(id);
			}
		};
		IntentFilter filter = new IntentFil-
ter("BalloonItemizedOverlay");
		registerReceiver(myReceiver, filter);
		if (savedInstanceState != null){
			prod = savedInstanceState.getString("prod");
		}
		prefs = this.getSharedPreferences("Settings", 0);
		dp = new Pharmacies(this);
```

```
                dg = new GasStations(this);
                sc = new ServerConnection(this);
                dialog = new ProgressDialog(this);
                setMap();
                findMyLocation();
                Bundle getBundle = getIntent().getExtras();
                choice = getBundle.getInt("choice", 0);
                hasFocushedOverlay = getBun-
dle.getString("hasFocushedOverlay");
                if (!hasFocushedOverlay.equals("")){
                        db.open();
                        String[] focushedData = new String[12];
                        if(choice==1){
                                focushedData =
db.fetchGasStationsData(hasFocushedOverlay);
                                prod = focushedData[11];
                        } else if (choice==0){
                                focushedData =
db.fetchPharmacysData(hasFocushedOverlay);
                                focushedIsOnDuty = focushedData[7];
                        }
                        focushedIndex = focushedData[8];
                        db.close();
                }
                switch (choice) {
                case 0:
                        getPharmacies();
                        break;
                case 1:
                        if(prod.equals("")){
                                showProductDialog();
                        }else{
                                getGasStations();
                        }
                        break;
                }
        }

        /*Function dataShow opens the Activity DataShow for showing
the data about the selected store*/
        public void dataShow(String id){
                Intent i = new In-
tent("com.anaskous.serresGuide.DATASHOW");
                Bundle b = new Bundle();
                b.putInt("choice", choice);
                if(myLocation!=null){
                        b.putDouble("srcLat", myLoca-
tion.getItem(0).getPoint().getLatitudeE6() / 1E6);
                        b.putDouble("srcLon", myLoca-
tion.getItem(0).getPoint().getLongitudeE6() / 1E6);
                } else {
                        b.putDouble("srcLat", 0.00);
                        b.putDouble("srcLon", 0.00);
                }
                b.putString("id", id);
                i.putExtras(b);
                startActivity(i);
        }

        /*Function onSaveInstanceState keep the value of variable
prod
```

```
         * during screentransmition from portrait to landscape and
vice versa*/
        @Override
        protected void onSaveInstanceState(Bundle outState) {
                // TODO Auto-generated method stub
                super.onSaveInstanceState(outState);
                outState.putString("prod", prod);
        }

        /*Function setMap initializes map's components*/
        public void setMap() {
                btList = (Button) findViewById(R.id.btList);
                btList.setOnClickListener(this);
                btRefresh = (Button) findViewById(R.id.btRefresh);
                btRefresh.setOnClickListener(this);
                btSetts = (Button) findViewById(R.id.btSetts);
                btSetts.setOnClickListener(this);
                btHome = (Button) findViewById(R.id.btHome);
                btHome.setOnClickListener(this);
                map = (MapView) findViewById(R.id.mvMap);
                map.setBuiltInZoomControls(true);
                map.setEnabled(false);
                btRefresh.setEnabled(false);
                btList.setEnabled(false);
                btSetts.setEnabled(false);
                overlayList = map.getOverlays();
                controller = map.getController();
                cross_green =
this.getResources().getDrawable(R.drawable.cross_green);
                cross_red =
this.getResources().getDrawable(R.drawable.cross_red);
                gas_station =
this.getResources().getDrawable(R.drawable.gas_station1);
                myloc =
this.getResources().getDrawable(R.drawable.pin1);
                farm_ef = new CustomPinpoint(cross_green, map);
                farm_ef.setOnFocusChangeListener(this);
                farm = new CustomPinpoint(cross_red, map);
                farm.setOnFocusChangeListener(this);
                gas = new CustomPinpoint(gas_station, map);
                gas.setOnFocusChangeListener(this);
        }
/*Function findMyLocation initializes the locationManagerand re-
trieves user's location*/
        public void findMyLocation() {
                ConnectivityManager connMgr = (ConnectivityManager)
this

        .getSystemService(Context.CONNECTIVITY_SERVICE);
                NetworkInfo netInfo = connMgr.getActiveNetworkInfo();
                lm = (LocationManager) getSystem-
Service(Context.LOCATION_SERVICE);
                Location l = null;
                if (netInfo != null && netInfo.isConnected()) {
                        Criteria crit = new Criteria();
                        towers = lm.getBestProvider(crit, false);
                        l = lm.getLastKnownLocation(towers);
                        isOnline = true;
                } else {
                        l =
lm.getLastKnownLocation(LocationManager.GPS_PROVIDER);
                        isOnline = false;
```

```
                }
                if (l != null) {
                        myLocation = new CustomPinpoint(myloc, map);
                        showMyLocation(l);
                        controller.animateTo(ourLocation);
                        controller.setZoom(15);
                }
        }

        /*Function onResume activates the location updates*/
        @Override
        protected void onResume() {
                // TODO Auto-generated method stub
                super.onResume();
                ConnectivityManager connMgr = (ConnectivityManager)
this

        .getSystemService(Context.CONNECTIVITY_SERVICE);
                NetworkInfo netInfo = connMgr.getActiveNetworkInfo();
                if (netInfo != null && netInfo.isConnected()) {
                        Criteria crit = new Criteria();
                        towers = lm.getBestProvider(crit, false);

                        lm.requestLocationUpdates(towers, 500, 1,
this);
                        isOnline = true;
                } else {

        lm.requestLocationUpdates(LocationManager.GPS_PROVIDER, 500,
1,
                                        this);
                        isOnline = false;
                }
        }

        /*Function saveLastUpdate saves on Preferences the value of
today by the name lastUpdate*/
        public void saveLastUpdate(String lastUpdate) {
                Date now = new Date();
                SimpleDateFormat dateFormat = new SimpleDateFor-
mat("dd/MM/yyyy");
                SharedPreferences.Editor editor =
this.getSharedPreferences("Settings",
                                0).edit();
                editor.putString(lastUpdate, dateFormat.format(now));
                editor.commit();
        }

        /*Function showMyLocation shows user's location on map*/
        public void showMyLocation(Location l) {
                lat = (int) (l.getLatitude() * 1E6);
                longi = (int) (l.getLongitude() * 1E6);
                ourLocation = new GeoPoint(lat, longi);
                overlayItem = new OverlayItem(ourLocation,
getString(R.string.myLocation),
                                "");
                myLocation.insertPinpoint(overlayItem);
                overlayList.add(myLocation);
        }

        /*Function onLocationChanged updates with the new user's lo-
cation*/
```

```
        public void onLocationChanged(Location l) {
                // TODO Auto-generated method stub
                overlayList.remove(myLocation);
                myLocation = new CustomPinpoint(myloc, map);
                map.invalidate();
                showMyLocation(l);
        }

        /*Function onPause stops location updates*/
        @Override
        protected void onPause() {
                // TODO Auto-generated method stub
                // compass.disableCompass();
                ConnectivityManager connMgr = (ConnectivityManager)
this

        .getSystemService(Context.CONNECTIVITY_SERVICE);
                NetworkInfo netInfo = connMgr.getActiveNetworkInfo();
                lm.removeUpdates(this);
                super.onPause();
        }

        /*Function checkForUpdates calculates if lastUpdate was at
least that days before today*/
        @SuppressLint("SimpleDateFormat")
        public boolean checkForUpdates(String lastUpdate, int days)
{
                boolean update = false;
                SimpleDateFormat dateFormat = new SimpleDateFor-
mat("dd/MM/yyyy");
                Date oldDate = null, newDate = new Date();
                if (!lastUpdate.equals("")) {
                        try {
                                oldDate = dateFor-
mat.parse(lastUpdate);
                        } catch (ParseException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        }
                        if (newDate.getTime() - oldDate.getTime() >
days * 24 * 60 * 60* 1000) {
                                update = true;
                        }
                } else {
                        update = true;
                }
                return update;
        }

        @Override
        protected boolean isRouteDisplayed() {
                // TODO Auto-generated method stub
                return false;
        }

        public void onProviderDisabled(String arg0) {
                // TODO Auto-generated method stub
        }

        public void onProviderEnabled(String arg0) {
                // TODO Auto-generated method stub
        }
```

```
        public void onStatusChanged(String arg0, int arg1, Bundle
arg2) {
                // TODO Auto-generated method stub
        }

        public void onClick(View arg0) {
                // TODO Auto-generated method stub
                switch (arg0.getId()) {
                case R.id.btList:
                        Bundle b = new Bundle();
                        Intent list = new In-
tent("com.anaskous.serresGuide.LISTACTIVITY");
                        b.putInt("choice", choice);
                        if(myLocation!=null){
                                b.putDouble("srcLat", myLoca-
tion.getItem(0).getPoint().getLatitudeE6() / 1E6);
                                b.putDouble("srcLon", myLoca-
tion.getItem(0).getPoint().getLongitudeE6() / 1E6);
                        } else {
                                b.putDouble("srcLat", 0.00);
                                b.putDouble("srcLon", 0.00);
                        }
                        if (choice == 1) {
                                b.putString("prod", prod);
                                b.putInt("order", 2);
                        } else {
                                b.putInt("order", 1);
                        }
                        list.putExtras(b);
                        startActivity(list);
                        break;
                case R.id.btRefresh:
                        runOnUiThread(new Runnable(){
                                public void run() {
                                        // TODO Auto-generated method
stub
                                        map.getOverlays().remove(rpo);
                                        map.postInvalidate();
                                }});
                        if(choice==0){
                                showPharmaciesRefreshDialog();
                        } else {
                                SharedPreferences.Editor edit = get-
SharedPreferences("Settings", 0).edit();
                                edit.putString("lastGasStationsUpdate"
+ prod, "");
                                edit.commit();
                                map.setEnabled(false);
                                btRefresh.setEnabled(false);
                                btList.setEnabled(false);
                                btSetts.setEnabled(false);
                                getGasStations();
                        }
                        break;
                case R.id.btSetts:
                        showRouteSettings();
                        break;
                case R.id.routeBt:
                        SharedPreferences.Editor edit = get-
SharedPreferences("Settings", 0).edit();
                        if(routeCb.isChecked()){
```

```
                                edit.putBoolean("showRoute", true);
                        }else{
                                edit.putBoolean("showRoute", false);
                                map.getOverlays().remove(rpo);
                                map.postInvalidate();
                        }
                        edit.putInt("routeMode",
routeSp.getSelectedItemPosition());
                        edit.commit();
                        routeAlert.dismiss();
                        break;
                case R.id.btHome:
                        Intent i = new Intent(this, Main.class);
                        i.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                        startActivity(i);
                        break;
                }
        }

        /*Function getPharmacies handles pharmacies download and il-
lustration*/
        public void getPharmacies() {
                new Thread() {
                        @Override
                        public void run() {
                                boolean internet = false;
                                Pharmacies p = new Pharma-
cies(ActivityMap.this, map,
                                                overlayList, farm,
farm_ef, dialog);
                                update = checkForUpdates(

        prefs.getString("lastPharmaciesUpdate", ""), 7);
                                update1 = checkForUpdates(

        prefs.getString("lastOnDutyUpdate", ""), 1);
                                if(update || update1){
                                        internet =
sc.checkConnection("http://www.fsserron.gr");
                                }
                                if(internet){
                                        if (update) {
                                                p.deletePharmacies("");

        dp.downloadAllPharmacies();

        saveLastUpdate("lastPharmaciesUpdate");
                                        }
                                        if (update1) {

        p.deletePharmacies("onDuty");

        dp.downloadOnDutyPharmacies();

        saveLastUpdate("lastOnDutyUpdate");
                                        }
                                } else {
                                        if (update1) {
                                                db.open();

        db.setPharmacyOffDuty();
                                                db.close();
```

```
                                        }
                                        if(update || update1){
                                                showNoInternetMes-
sage();
                                        }
                                }
                                farmakeia = new Ar-
rayList<List<String>>();
                                db.open();
                                farmakeia =
db.fetchPharmacies("name");
                                db.close();
                                Pharmacies p2 = new Pharma-
cies(ActivityMap.this, map, controller, overlayList, myLocation,
farmakeia,
                                                farm, farm_ef, dialog,
isOnline, update1, btRefresh, btList, btSetts, focushedIndex, fo-
cushedIsOnDuty);
                                p2.showPharmacies();
                        }
                }.start();
        }

        /*Function getGasStations handles gas station's download and
illustration*/
        public void getGasStations() {
                new Thread() {
                        @Override
                        public void run() {

                                GasStations g = new GasSta-
tions(ActivityMap.this, map,
                                                overlayList, gas, dia-
log);
                                update = checkForUpdates(

        prefs.getString("lastGasStationsUpdate" + prod, ""), 1);
                                if (update) {
                                        boolean internet =
sc.checkConnection("http://www.fuelprices.gr/GetGeography");
                                                if(internet){

        g.deleteGasStations(prod);

        dg.downloadGasStations(prod);

        saveLastUpdate("lastGasStationsUpdate" + prod);
                                                }else if (!internet){
                                                showNoInternetMes-
sage();
                                        }
                                }
                                gasStations = new Ar-
rayList<List<String>>();
                                db.open();
                                gasStations =
db.fetchGasStations(prod, "name");
                                db.close();
                                GasStations g2 = new GasSta-
tions(ActivityMap.this, map,
                                                controller, over-
layList, myLocation, gasStations, gas,
```

```
                                                dialog, isOnline, btRe-
fresh, btList, btSetts, focushedIndex);
                                g2.showGasStations();
                        }
                }.start();
        }

        /*Function showProductDialog shows dialog with all gas prod-
ucts for the user to choose*/
        private void showProductDialog() {
                // TODO Auto-generated method stub
                prodAlert = new AlertDia-
log.Builder(ActivityMap.this).create();

        //prodAlert.requestWindowFeature(Window.FEATURE_NO_TITLE);
                prodAlert.setTitle(getString(R.string.product));
                LayoutInflater inflater = (LayoutInflater) Activi-
tyMap.this

        .getSystemService(Context.LAYOUT_INFLATER_SERVICE);
                View layout = inflat-
er.inflate(R.layout.dialog_listview, null);
                prodAlert.setView(layout);
                //prodAlert.setCancelable(false);
                prodAlert.show();
                prodAlert.setOnCancelListener(new OnCancelListener(){
                        public void onCancel(DialogInterface arg0) {
                                // TODO Auto-generated method stub
                                onBackPressed();
                        }
                });

                ListView productsLv = (ListView) prodAlert
                                .findViewById(R.id.dialogLv);
                ArrayList<String> products = new Ar-
rayList(Arrays.asList(
                                getString(R.string.unleaded95oct),
getString(R.string.unleaded100oct),
                                getString(R.string.super_gas),
getString(R.string.dieselD), getString(R.string.dieselH),
                                getString(R.string.autogas),
getString(R.string.dieselHD)));
                ArrayAdapter<String> adapter = new ArrayAdapt-
er<String>(
                                ActivityMap.this,
R.layout.simple_textview, products);
                productsLv.setAdapter(adapter);
                productsLv.setOnItemClickListener(ActivityMap.this);
        }

        /*Function showPharmaciesRefreshDialog shows dialog with
pharmacies refresh options for the user to choose*/
        public void showPharmaciesRefreshDialog(){
                pharmAlert = new AlertDia-
log.Builder(ActivityMap.this).create();

        //pharmAlert.requestWindowFeature(Window.FEATURE_NO_TITLE);
                pharmAlert.setTitle(getString(R.string.refresh));
                LayoutInflater inflater = (LayoutInflater) Activi-
tyMap.this

        .getSystemService(Context.LAYOUT_INFLATER_SERVICE);
```

```
                View layout = inflat-
er.inflate(R.layout.dialog_listview, null);
                pharmAlert.setView(layout);
                //pharmAlert.setCancelable(false);
                pharmAlert.show();

                ListView productsLv = (ListView) pharmAlert
                                .findViewById(R.id.dialogLv);
                ArrayList<String> products = new Ar-
rayList(Arrays.asList(getString(R.string.all),
getString(R.string.onDuty)));
                ArrayAdapter<String> adapter = new ArrayAdapt-
er<String>(
                                ActivityMap.this,
R.layout.simple_textview, products);
                productsLv.setAdapter(adapter);
                productsLv.setOnItemClickListener(ActivityMap.this);
        }

        public void onItemClick(AdapterView<?> arg0, View arg1, fi-
nal int arg2,
                        long arg3) {
                // TODO Auto-generated method stub
                map.setEnabled(false);
                btRefresh.setEnabled(false);
                btList.setEnabled(false);
                btSetts.setEnabled(false);
                if(arg0.getCount()==2){
                        final SharedPreferences.Editor edit = get-
SharedPreferences("Settings", 0).edit();
                        if(arg2==0){
                                edit.putString("lastPharmaciesUpdate",
"");
                        }
                        edit.putString("lastOnDutyUpdate", "");
                        edit.commit();
                        pharmAlert.cancel();
                        getPharmacies();
                } else {
                        prodAlert.dismiss();
                        prod = String.valueOf(arg2 + 1);
                        getGasStations();
                }
        }

        /*Function onFocusChanged shows the route to the new fo-
cushed item*/
        public void onFocusChanged(ItemizedOverlay overlay, final
OverlayItem newFocus) {
                // TODO Auto-generated method stub
                SharedPreferences prefs = getSharedPrefer-
ences("Settings", 0);
                boolean showRoute = prefs.getBoolean("showRoute",
false);
                if(showRoute && myLocation!=null && newFocus!=null){
                        double dist = calcDis-
tance(newFocus.getPoint());
                        if(dist<50000){
                                map.getOverlays().remove(rpo);
                                map.postInvalidate();
```

```
                                //rpo =
sc.getPath(myLocation.getItem(0).getPoint(), newFocus.getPoint(),
Color.RED, map);
                                sc.getPath(this, myLoca-
tion.getItem(0).getPoint(), newFocus.getPoint(), Color.RED, map);
                        }else{
                                Toast.makeText(this,
getString(R.string.noRouteWarning), Toast.LENGTH_SHORT).show();
                        }
                }
        }

        /*Function calcDistance calculates the distance between the
user and dest*/
        public float calcDistance(GeoPoint dest) {
                Location source = new Location("");

        source.setLatitude(myLocation.getItem(0).getPoint().getLatit
udeE6() / 1E6);

        source.setLongitude(myLocation.getItem(0).getPoint().getLong
itudeE6() / 1E6);

                Location destination = new Location("");
                destination.setLatitude(dest.getLatitudeE6() / 1E6);
                destination.setLongitude(dest.getLongitudeE6() /
1E6);

                return source.distanceTo(destination);
        }

        /*Function showNoInternetMessage does the necessary action
if something is wrong with the connection*/
        public void showNoInternetMessage(){
                runOnUiThread(new Runnable() {
                        public void run() {
                                Toast.makeText(ActivityMap.this,
getString(R.string.noInternetWarning),

        Toast.LENGTH_SHORT).show();
                                map.setEnabled(true);
                                btRefresh.setEnabled(true);
                                btList.setEnabled(true);
                                btSetts.setEnabled(true);
                        }
                });
        }

        /*Function showRouteSettings shows the settings dialog*/
        public void showRouteSettings(){
                routeAlert = new AlertDia-
log.Builder(ActivityMap.this).create();

        //routeAlert.requestWindowFeature(Window.FEATURE_NO_TITLE);
                routeAlert.setTitle(getString(R.string.settings));

                Rect displayRectangle = new Rect();
                Window window = this.getWindow();
                win-
dow.getDecorView().getWindowVisibleDisplayFrame(displayRectangle);
                // inflate and adjust layout
```

```
                LayoutInflater inflater = (LayoutInflater) Activi-
tyMap.this

        .getSystemService(Context.LAYOUT_INFLATER_SERVICE);
                View layout = inflat-
er.inflate(R.layout.route_settings_dialog, null);
                layout.setMinimumWidth((int)(displayRectangle.width()
* 0.8f));
                lay-
out.setMinimumHeight((int)(displayRectangle.height() * 0.3f));
                routeAlert.setView(layout);
                //routeAlert.setCancelable(false);
                routeAlert.show();

                SharedPreferences prefs = getSharedPrefer-
ences("Settings", 0);
                int routeMode = prefs.getInt("routeMode", 0);
                boolean showRoute = prefs.getBoolean("showRoute",
false);
                routeSp = (Spin-
ner)routeAlert.findViewById(R.id.routeSp);
                ArrayList<String> mode = new Ar-
rayList(Arrays.asList(getString(R.string.car),
getString(R.string.bike), getString(R.string.onFoot)));
                ArrayAdapter adapter = new ArrayAdapter(
                                this, an-
droid.R.layout.simple_spinner_item, mode);
                adapter.setDropDownViewResource(
                   android.R.layout.simple_spinner_dropdown_item);
                routeSp.setAdapter(adapter);
                routeSp.setSelection(routeMode);
                btSettsOk = (But-
ton)routeAlert.findViewById(R.id.routeBt);
                btSettsOk.setOnClickListener(this);
                routeCb = (Check-
Box)routeAlert.findViewById(R.id.routeCb);
                if(showRoute){
                        routeCb.setChecked(true);
                }else{
                        routeCb.setChecked(false);
                }
                map.getOverlays().remove(rpo);
                map.postInvalidate();
        }
}
```

A.1.3. ActivityList

```
/* Class ActivityList is responsible for handling the expandable
list*/
public class ActivityList extends ExpandableListActivity implements
                OnChildClickListener, OnClickListener, OnItemClick-
Listener {
        ListView list;
        ArrayList<String> arraylist = new ArrayList<String>();
        List<List<String>> farmakeia, benzinadika;
        DBManager db = new DBManager(this);
        String[] farm, efim, gas, titlesP, titlesG;
        AlertDialog orderDialog;
        String order, prod;
        int choice;
```

```
        double srcLat, srcLon;

        /*Function onCreate contains code for variables/components
         * intialization and expandable list completion*/
        @Override
        protected void onCreate(Bundle savedInstanceState) {
                // TODO Auto-generated method stub
                super.onCreate(savedInstanceState);
                setContentView(R.layout.list);
                titlesP = new String[]{getString(R.string.ONDUTY),
getString(R.string.ALL)};
                titlesG = new
String[]{getString(R.string.GASSTATIONS)};
                Bundle b = getIntent().getExtras();
                choice = b.getInt("choice");
                srcLat = b.getDouble("srcLat");
                srcLon = b.getDouble("srcLon");
                if (b.getInt("order") == 0) {
                        if(choice==0){
                                order = "name";
                        } else {
                                order = "brand";
                        }
                } else if (b.getInt("order") == 1) {
                        order = "distance";
                } else if (b.getInt("order") == 2) {
                        order = "price";
                }
                if(choice==1){
                        prod = b.getString("prod");
                }
                ExpandableListView expList =
this.getExpandableListView();
                expList.setOnChildClickListener(this);
                Button btOrder = (Button) findViewById(R.id.btOrder);
                btOrder.setOnClickListener(this);
                Button btHome = (Button) findViewById(R.id.btHome);
                btHome.setOnClickListener(this);
                Button btBack = (Button) findViewById(R.id.btBack);
                btBack.setOnClickListener(this);
                if(choice==0){
                        fillPharmaciesListView();
                } else {
                        fillGasStationsListView();
                }
                expList.expandGroup(0);
        }

        /*Function calcDistance calculates the distance between the
user
         * and each store of the list stores and updates database*/
        public void calcDistance(List<List<String>> stores) {
                db.open();
                if (srcLat!=0.00 && srcLon!=0.00) {
                        Location source = new Location("");
                        source.setLatitude(srcLat);
                        source.setLongitude(srcLon);
                        for (int i = 0; i < stores.size(); i++) {
                                Location destination = new Loca-
tion("");
                                String dist;
```

```
                                if
(!stores.get(i).get(4).equals("0.0") ||
!stores.get(i).get(5).equals("0.0")) {
                                        destina-
tion.setLatitude(Double.valueOf(stores.get(i).get(4)));
                                        destina-
tion.setLongitude(Double.valueOf(stores.get(i).get(5)));
                                        dist =
String.valueOf(source.distanceTo(destination));
                                } else {
                                        dist = "";
                                }
                                if (choice == 0) {

        db.setPharmacyDistance(stores.get(i).get(0), dist);
                                } else if (choice == 1) {

        db.setGasStationDistance(stores.get(i).get(0), dist);
                                }
                        }
                } else {
                        db.setAllDistanceNull();
                }
                db.close();
        }

        /*Function fillGasStationsListView inserts the informations
         * of the gas stations to the expandable list*/
        public void fillGasStationsListView() {
                db.open();
                benzinadika = db.fetchGasStations(prod, order);
                db.close();
                if(order.equals("distance")){
                        calcDistance(benzinadika);
                        db.open();
                        benzinadika = db.fetchGasStations(prod, or-
der);
                        db.close();
                }
                gas = new String[benzinadika.size()];
                int k = 0, l = 0;
                for (int i = 0; i < benzinadika.size(); i++) {
                        gas[i] = benzinadika.get(i).get(2) + "\n" +
benzinadika.get(i).get(1);
                        db.open();
                        boolean isShowing =
db.isShowing(benzinadika.get(i).get(1), choice);
                        db.close();
                        if(!isShowing){
                                gas[i]+="\n" +
getString(R.string.notShowingOnMap);
                        }
                }
                SimpleExpandableListAdapter expListAdapter = new Sim-
pleExpandableListAdapter(
                                this, createGroupList(),
                                R.layout.child_row,
                                new String[] { "title" },
                                new int[] { R.id.childname },
                                createChildList(),
                                R.layout.child_row,
                                new String[] { "body" },
```

```
                                new int[] { R.id.childname }
                );
                setListAdapter(expListAdapter);
        }


        /*Function fillPharmaciesListView inserts the informations
         * of the pharmacies to the expandable list*/
        public void fillPharmaciesListView() {
                db.open();
                farmakeia = db.fetchPharmacies(order);
                int countOnDuty = db.countOnDuty();
                db.close();
                if(order.equals("distance")){
                        calcDistance(farmakeia);
                        db.open();
                        farmakeia = db.fetchPharmacies(order);
                        db.close();
                }
                farm = new String[farmakeia.size()];
                efim = new String[countOnDuty];
                int k = 0, l = 0;
                for (int i = 0; i < farmakeia.size(); i++) {
                        if (!farmakeia.get(i).get(7).equals("")) {
                                efim[l] = farmakeia.get(i).get(1);
                                l++;
                        }
                        farm[i] = farmakeia.get(i).get(1);
                        db.open();
                        boolean isShowing =
db.isShowing(farmakeia.get(i).get(1), choice);
                        db.close();
                        if(!isShowing){
                                if
(!farmakeia.get(i).get(7).equals("")) {
                                        efim[l-1]+="\n" +
getString(R.string.notShowingOnMap);
                                }
                                farm[i]+="\n" +
getString(R.string.notShowingOnMap);
                        }
                }
                SimpleExpandableListAdapter expListAdapter = new Sim-
pleExpandableListAdapter(
                                this, createGroupList(),
                                R.layout.child_row,
                                new String[] { "title" },
                                new int[] { R.id.childname },
                                createChildList(),
                                R.layout.child_row,
                                new String[] { "body" },
                                new int[] { R.id.childname }
                );
                setListAdapter(expListAdapter);
        }

        /**
         * Function createGroupList reates the group list out of the
colors[] array according to the
         * structure required by SimpleExpandableListAdapter. The
resulting List
```

```
         * contains Maps. Each Map contains one entry with key "col-
orName" and value
         * of an entry in the colors[] array.
         */
        private List createGroupList() {
                ArrayList result = new ArrayList();
                if(choice==0){
                        for (int i = 0; i < titlesP.length; ++i) {
                                HashMap m = new HashMap();
                                m.put("title", titlesP[i]);
                                result.add(m);
                        }
                } else {
                        for (int i = 0; i < titlesG.length; ++i) {
                                HashMap m = new HashMap();
                                m.put("title", titlesG[i]);
                                result.add(m);
                        }
                }
                return (List) result;
        }

        /**
         * Function createChildList reates the child list out of the
shades[] array according to the
         * structure required by SimpleExpandableListAdapter. The
resulting List
         * contains one list for each group. Each such second-level
group contains
         * Maps. Each such Map contains two keys: "shadeName" is the
name of the
         * shade and "rgb" is the RGB value for the shade.
         */
        private List createChildList() {
                ArrayList result = new ArrayList();
                ArrayList secList = new ArrayList();
                if(choice==0){
                        for (int i = 0; i < efim.length; i++) {
                                HashMap child = new HashMap();
                                child.put("body", efim[i]);
                                secList.add(child);
                        }
                        result.add(secList);

                        secList = new ArrayList();
                        for (int i = 0; i < farm.length; i++) {
                                HashMap child = new HashMap();
                                child.put("body", farm[i]);
                                secList.add(child);
                        }
                        result.add(secList);
                } else {
                        for (int i = 0; i < gas.length; i++) {
                                HashMap child = new HashMap();
                                child.put("body", gas[i]);
                                secList.add(child);
                        }
                        result.add(secList);
                }
                return result;
        }
```

```
        /*Function onChildClick opens the ActivityMap with focus on
the selected store,
         * or the DataShow if the selected store isn't showing on
map*/
        public boolean onChildClick(ExpandableListView parent, View
v,
                        int groupPosition, int childPosition, long
id0) {
                // TODO Auto-generated method stub
                String id = "";
                boolean isShowing = false;
                db.open();
                if(choice==0){
                        if (groupPosition == 0) {
                                String[] temp =
efim[childPosition].split("\n");
                                id = db.getIdFromName(temp[0],
choice);
                                isShowing = db.isShowing(temp[0],
choice);
                        } else if (groupPosition == 1) {
                                String[] temp =
farm[childPosition].split("\n");
                                id = db.getIdFromName(temp[0],
choice);
                                isShowing = db.isShowing(temp[0],
choice);
                        }
                } else {
                        String[] temp =
gas[childPosition].split("\n");
                        id = db.getIdFromName(temp[1], choice);
                        isShowing = db.isShowing(temp[1], choice);
                }
                db.close();
                Intent i;
                Bundle b = new Bundle();
                b.putInt("choice", choice);
                b.putDouble("srcLat", srcLat);
                b.putDouble("srcLon", srcLon);
                if(isShowing){
                        i = new In-
tent("com.anaskous.serresGuide.MAP");
                        i.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                        b.putString("hasFocushedOverlay", id);

                }else{
                        i = new In-
tent("com.anaskous.serresGuide.DATASHOW");
                        b.putString("id", id);
                }
                i.putExtras(b);
                startActivity(i);
                return super.onChildClick(parent, v, groupPosition,
childPosition, id0);
        }

        public void onClick(View v) {
                // TODO Auto-generated method stub
                switch (v.getId()) {
                case R.id.btOrder:
                        showOrderDialog();
```

```
                        break;
                case R.id.btHome:
                        Intent i = new Intent(this, Main.class);
                        i.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                        startActivity(i);
                        break;
                case R.id.btBack:
                        super.onBackPressed();
                        break;
                }
        }

        /*Function showOrderDialog shows the dialog with order se-
lections*/
        public void showOrderDialog() {
                orderDialog = new AlertDialog.Builder(this).create();
                order-
Dialog.requestWindowFeature(Window.FEATURE_NO_TITLE);
                orderDialog.setCancelable(true);
                Rect displayRectangle = new Rect();
                Window window = this.getWindow();
                win-
dow.getDecorView().getWindowVisibleDisplayFrame(displayRectangle);
                // inflate and adjust layout
                LayoutInflater inflater = (LayoutInflater) this

        .getSystemService(Context.LAYOUT_INFLATER_SERVICE);
                View layout = inflater.inflate(R.layout.dialog_order,
null);
                layout.setMinimumWidth((int) (displayRectan-
gle.width() * 0.6f));
                // lay-
out.setMinimumHeight((int)(displayRectangle.height() * 0.9f));
                orderDialog.setView(layout);
                orderDialog.show();
                ListView lvOrder = (ListView) order-
Dialog.findViewById(R.id.lvOrder);
                List<String> orderList = new ArrayList<String>();
                orderList.add(getString(R.string.name));
                orderList.add(getString(R.string.distance));
                if(choice==1){
                        orderList.add(getString(R.string.price));
                }
                ArrayAdapter<String> lvOrdersAdapter = new ArrayA-
dapter<String>(this,
                                R.layout.simple_textview, orderList);
                lvOrder.setAdapter(lvOrdersAdapter);
                lvOrder.setOnItemClickListener(this);
        }

        /*Function onItemClick restarts the ActivityList and passes
the selected order*/
        public void onItemClick(AdapterView<?> arg0, View arg1, int
pos, long arg3) {
                // TODO Auto-generated method stub
                Bundle b = new Bundle();
                Intent list = new In-
tent("com.anaskous.serresGuide.LISTACTIVITY");
                list.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                b.putInt("order", pos);
                b.putInt("choice", choice);
                b.putDouble("srcLat", srcLat);
```

```
                b.putDouble("srcLon", srcLon);
                if(choice==1){
                        b.putString("prod", prod);
                }
                list.putExtras(b);
                startActivity(list);
        }
}
```

A.1.4. DataShow

```
/*Class DataShow shows the store's data on a dynamic table*/
public class DataShow extends Activity implements OnClickListener{

        int choice;
        String id;
        DBManager db = new DBManager(this);
        String[] data =new String[12];
        double srcLat, srcLon;

        /*Function onCreate contains code for variables/components
         * intialization and expandable list completion*/
        @Override
        protected void onCreate(Bundle savedInstanceState) {
                // TODO Auto-generated method stub
                super.onCreate(savedInstanceState);
                setContentView(R.layout.data_show);
                Button btHome = (Button)findViewById(R.id.btHome);
                btHome.setOnClickListener(this);
                Button btBack = (Button) findViewById(R.id.btBack);
                btBack.setOnClickListener(this);

                Bundle getBundle = getIntent().getExtras();
                choice = getBundle.getInt("choice", 0);
                id = getBundle.getString("id");
                srcLat = getBundle.getDouble("srcLat");
                srcLon = getBundle.getDouble("srcLon");

                if(choice==0){
                        fillPharmacyData();
                }else if(choice==1){
                        fillGasStationData();
                }

        }

        /*Function fillGasStationData creates dynamicly the table
and fills in the gas station's data*/
        public void fillGasStationData(){
                db.open();
                data = db.fetchGasStationsData(id);
                db.close();
                TextView nameTv =
(TextView)findViewById(R.id.titleTv);
                nameTv.setText(data[1]);
                TextView brandTv =
(TextView)findViewById(R.id.detailsTv);
                brandTv.setText(getString(R.string.trademark) + da-
ta[2]);
                brandTv.setTextColor(Color.YELLOW);
```

```
                TableLayout table = (TableLay-
out)findViewById(R.id.tableTl);
                TableRow row =
(TableRow)LayoutInflater.from(this).inflate(R.layout.data_show_grid,
null);

((TextView)row.findViewById(R.id.etWhat)).setText(getString(R.string
.address));

((TextView)row.findViewById(R.id.etValue)).setText(data[3]);
            table.addView(row);
            String dist = calcDistance();
            if(!dist.equals("")){
                float distance = Float.valueOf(dist)/1000;
                row =
(TableRow)LayoutInflater.from(this).inflate(R.layout.data_show_grid,
null);

        ((TextView)row.findViewById(R.id.etWhat)).setText(getString(
R.string.distance1));

        ((TextView)row.findViewById(R.id.etValue)).setText(String.fo
rmat("%.02f", distance) + " km");
                table.addView(row);
            }
                row =
(TableRow)LayoutInflater.from(this).inflate(R.layout.data_show_grid,
null);

((TextView)row.findViewById(R.id.etWhat)).setText(getString(R.string
.product));

((TextView)row.findViewById(R.id.etValue)).setText(data[10]);
            table.addView(row);
                row =
(TableRow)LayoutInflater.from(this).inflate(R.layout.data_show_grid,
null);

((TextView)row.findViewById(R.id.etWhat)).setText(getString(R.string
.price1));

((TextView)row.findViewById(R.id.etValue)).setText(data[9]);
            table.addView(row);
                row =
(TableRow)LayoutInflater.from(this).inflate(R.layout.data_show_grid,
null);

((TextView)row.findViewById(R.id.etWhat)).setText(getString(R.string
.regDate));

((TextView)row.findViewById(R.id.etValue)).setText(data[7]);
            table.addView(row);
        }


        /*Function fillPharmacyData creates dynamicly the table and
fills in the pharmacy's data*/
        public void fillPharmacyData(){
                db.open();
                data = db.fetchPharmacysData(id);
                db.close();
```

```
                TextView nameTv =
(TextView)findViewById(R.id.titleTv);
                nameTv.setText(data[1]);
                TextView onDutyTv =
(TextView)findViewById(R.id.detailsTv);
                if(data[7].equals("")){
                        onDu-
tyTv.setText(getString(R.string.isNotOnDuty));
                        onDutyTv.setTextColor(Color.RED);
                } else {
                        onDutyTv.setText(getString(R.string.isOnDuty)
+ ": " + data[7]);
                        onDutyTv.setTextColor(Color.GREEN);
                }
                TableLayout table = (TableLay-
out)findViewById(R.id.tableTl);
                TableRow row =
(TableRow)LayoutInflater.from(this).inflate(R.layout.data_show_grid,
null);

((TextView)row.findViewById(R.id.etWhat)).setText(getString(R.string
.address));

((TextView)row.findViewById(R.id.etValue)).setText(data[3]);
            table.addView(row);
            String dist = calcDistance();
            if(!dist.equals("")){
                float distance = Float.valueOf(dist)/1000;
                row =
(TableRow)LayoutInflater.from(this).inflate(R.layout.data_show_grid,
null);

        ((TextView)row.findViewById(R.id.etWhat)).setText(getString(
R.string.distance1));

        ((TextView)row.findViewById(R.id.etValue)).setText(String.fo
rmat("%.02f", distance) + " km");
                table.addView(row);
            }
                row =
(TableRow)LayoutInflater.from(this).inflate(R.layout.data_show_grid,
null);

((TextView)row.findViewById(R.id.etWhat)).setText(getString(R.string
.phoneNumber));

((TextView)row.findViewById(R.id.etValue)).setText(data[2]);
            table.addView(row);
        }

        /*Function calcDistance calculates the distance between the
selected store and the user's location*/
        public String calcDistance() {
                if ((srcLat!=0.00 || srcLon!=0.00) && (!da-
ta[4].equals("0.0") || !data[5].equals("0.0"))) {
                        Location source = new Location("");
                        source.setLatitude(srcLat);
                        source.setLongitude(srcLon);

                        Location destination = new Location("");
                        destina-
tion.setLatitude(Double.valueOf(data[4]));
```

```
                        destina-
tion.setLongitude(Double.valueOf(data[5]));

                        return
String.valueOf(source.distanceTo(destination));
                } else {
                        db.open();
                        db.setAllDistanceNull();
                        db.close();
                        return "";
                }
        }

        public void onClick(View arg0) {
                // TODO Auto-generated method stub
                switch(arg0.getId()){
                        case R.id.btHome:
                                Intent i = new Intent(this,
Main.class);

        i.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                                startActivity(i);
                                break;
                        case R.id.btBack:
                                super.onBackPressed();
                                break;
                }
        }
}
```

A.2. Κλάσεις

A.2.1. ServerConnection

```
/*Class ServerConnection handles the functions that require internet
connection*/
public class ServerConnection {

        Context c;
        boolean error;

        /*Constructor ServerConnection initializes variables*/
        ServerConnection(Context c_) {
                c = c_;
        }

        /*Function checkConnection checks connectivity by opening a
trial connection*/
        public boolean checkConnection(String url) {
                boolean isOk = false;
                // First, check we have any sort of connectivity
                final ConnectivityManager connMgr = (ConnectivityMan-
ager) c

        .getSystemService(Context.CONNECTIVITY_SERVICE);
                final NetworkInfo netInfo =
connMgr.getActiveNetworkInfo();
                HttpURLConnection urlc = null;
                if (netInfo != null && netInfo.isConnected()) {
                        // Some sort of connection is open, check if
server is reachable
```

```
                        try {
                                URL url1 = new URL(url);
                                urlc = (HttpURLConnection)
url1.openConnection();
                                urlc.setRequestProperty("User-Agent",
"Android Application");
                                urlc.setRequestProperty("Connection",
"close");
                                urlc.setConnectTimeout(5 * 1000);// 10
seconds timeout in milliseconds
                                urlc.setReadTimeout(4000);
                                urlc.connect();
                                if (urlc.getResponseCode() == 200) {
// Good response
                                        isOk = true;
                                } else { // Anything else is unwanted
                                        isOk = false;
                                }
                        } catch (IOException e) {
                                e.printStackTrace();
                                isOk = false;
                        } catch (Exception e) {
                                e.printStackTrace();
                                isOk = false;
                        } finally {
                                urlc.disconnect();
                        }
                } else {// Den uparxei sundesi diktuou
                        isOk = false;
                }
                return isOk;
        }

        /*Function readHtml receives the url's source code*/
        public BufferedReader readHtml(String url, String encoding)
{
                HttpClient httpClient = new DefaultHttpClient();
                HttpContext localContext = new BasicHttpContext();
                HttpGet httpGet = new HttpGet(url);
                HttpResponse response = null;

                try {
                        response = httpClient.execute(httpGet, local-
Context);
                } catch (ClientProtocolException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }

                BufferedReader reader = null;
                try {
                        reader = new BufferedReader(new In-
putStreamReader(response
                                        .getEntity().getContent(), en-
coding));
                } catch (IllegalStateException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                } catch (IOException e) {
```

```
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
                return reader;
        }

        /*Subclass getPath receives directions and handles route's
illustration on map*/
        class getPath extends AsyncTask<String, Void, RoutePathOver-
lay> {
         private Exception exception;
         GeoPoint src; GeoPoint dest; int color; MapView map;
                RoutePathOverlay rpo=null; Context c; String sDis-
tance="";

                /*Constructor getPath initializes variables*/
         getPath(Context c_, GeoPoint src_, GeoPoint dest_, int col-
or_, MapView map_){
                src=src_;
                dest=dest_;
                map=map_;
                c=c_;
         }

         /*Function doInBackground receives directions and handles
route's illustration on map*/
         protected RoutePathOverlay doInBackground(String... urls) {
                HttpParams params = new BasicHttpParams();

        params.setParameter(CoreProtocolPNames.PROTOCOL_VERSION,
HttpVersion.HTTP_1_1);
                HttpClient httpclient = new DefaultHttpCli-
ent(params);

                SharedPreferences prefs =
c.getSharedPreferences("Settings", 0);
                int routeMode = prefs.getInt("routeMode", 0);
                String mode = "";

                if(routeMode==0){
                        mode = "driving";
                }else if(routeMode==1){
                        mode = "biking";
                }else if(routeMode==2){
                        mode = "walking";
                }

                String url =
"http://maps.googleapis.com/maps/api/directions/json?origin=" +
                                src.getLatitudeE6()/1E6 +"," +
src.getLongitudeE6()/1E6 + "&destination=" +
                                dest.getLatitudeE6()/1E6 + "," +
dest.getLongitudeE6()/1E6 + "&sensor=false"
                                + "&mode=" + mode;

                HttpPost httppost = new HttpPost(url);
                HttpResponse response = null;

                try {
                        response = httpclient.execute(httppost);
                HttpEntity entity = response.getEntity();
                InputStream is = null;
```

```
                is = entity.getContent();
                BufferedReader reader = new BufferedReader(new In-
putStreamReader(is, "iso-8859-1"), 8);
                StringBuilder sb = new StringBuilder();
                sb.append(reader.readLine() + "\n");
                String line = "0";
                while ((line = reader.readLine()) != null) {
                    sb.append(line + "\n");
                }
                is.close();
                reader.close();
                String result = sb.toString();

                JSONObject jsonObject = new JSONObject(result);
                JSONArray routeArray = jsonOb-
ject.getJSONArray("routes");
                JSONObject routes = routeArray.getJSONObject(0);

                JSONObject overviewPolylines =
routes.getJSONObject("overview_polyline");
                String encodedString = overviewPol-
ylines.getString("points");
                final List<GeoPoint> pointToDraw = decode-
Poly(encodedString);

                JSONArray legs = routes.getJSONArray("legs");
            JSONObject steps = legs.getJSONObject(0);
            JSONObject distance = steps.getJSONObject("distance");
            sDistance = distance.getString("text");

            rpo = new RoutePathOverlay(pointToDraw);
               map.getOverlays().add(rpo);
               map.postInvalidate();

               } catch (ClientProtocolException e) {
                       // TODO Auto-generated catch block
                       e.printStackTrace();
                } catch (IOException e) {
                       // TODO Auto-generated catch block
                       e.printStackTrace();
                } catch (JSONException e) {
                       // TODO Auto-generated catch block
                       e.printStackTrace();
                }
                return rpo;
        }

        /*Function onPostExecute returns the RoutePathOverlay to
ActivityMap*/
                @Override
                protected void onPostExecute(RoutePathOverlay result)
{
                       // TODO Auto-generated method stub
                    ActivityMap am = (ActivityMap) c;
                    am.rpo = result;
            Toast.makeText(c, c.getString(R.string.distance1) +
sDistance, Toast.LENGTH_SHORT).show();
                       super.onPostExecute(result);
                }
     }

       /*Fuction getPath triggers the getPath's class function*/
```

```
        public void getPath(Context c, GeoPoint src, GeoPoint
dest,int color, MapView map) {
                new getPath(c, src, dest, color, map).execute();
        }

        /*Function decodePoly decodes the directions*/
        private List<GeoPoint> decodePoly(String encoded) {
            List<GeoPoint> poly = new ArrayList<GeoPoint>();
            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;

                GeoPoint p = new GeoPoint((int) (((double) lat /
1E5) * 1E6), (int) (((double) lng / 1E5) * 1E6));
                poly.add(p);
            }
            return poly;
        }

        /*Function getCoordinartes gets the latitude and latitude of
address*/
        public double[] getCoordinartes(String address) {
                Geocoder geocoder = new Geocoder(c, Lo-
cale.getDefault());
                error = false;
                double latitude = 0, longitude = 0;
                List<Address> addresses = null;
                try {
                        addresses = geocod-
er.getFromLocationName(address, 1);
                } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                        error = true;
                }
                // Address address = addresses.get(0);
                if (!error && addresses.size() > 0) {
                        if (addresses.get(0).hasLatitude()
                                        && address-
es.get(0).hasLongitude()) {
```

```
                                latitude = address-
es.get(0).getLatitude();
                                longitude = address-
es.get(0).getLongitude();
                        }
                }
                return new double[]{latitude, longitude};
        }

}
```

2.2. DBManager

```
/*Class DBManager contains functions that manages the database*/
public class DBManager {

        public static final String DBName = "serresguide.db";
        public static final int DBVersion = 1;
        private DBHelper ourHelper;
        private final Context ourContext;
        private SQLiteDatabase ourDatabase;
        Cursor c;

        /*Subclass DBHelper contains functions for database for-
mation*/
        public static class DBHelper extends SQLiteOpenHelper {

                /* Constructor DBHelper*/
                public DBHelper(Context c) {
                        super(c, DBName, null, DBVersion);
                }

                /*Function onCreate creates the tables*/
                @Override
                public void onCreate(SQLiteDatabase db) {
                        // TODO Auto-generated method stub

                        db.execSQL("create table PHARMACIES (id inte-
ger primary key autoincrement, name text, phone text, address text,
latitude text, longitude text, " +
                                        "distance text, hours text,
mapid text);");

                        db.execSQL("create table GASSTATIONS (id in-
teger primary key autoincrement, name text, brand text, address
text, latitude text, longitude text, " +
                                        "distance text, hours text,
mapId text, price text, product text, prodId text);");
                }

                /*Function onUpgrade handles the tables transfor-
mation when there is a new version of the database*/
                @Override
                public void onUpgrade(SQLiteDatabase db, int oldVer-
sion, int newVersion) {
                        // TODO Auto-generated method stub
                        db.execSQL("DROP TABLE IF EXISTS PHARMA-
CIES");
                        db.execSQL("DROP TABLE IF EXISTS GASSTA-
TIONS");
                        onCreate(db);
```

```
                }
        }

        /*Constructor DBManager initializes variables*/
        public DBManager(Context context) {
                ourContext = context;
        }

        /*Function open returns a SQLiteDatabase object for edit-
ing*/
        public DBManager open() throws SQLException {
                ourHelper = new DBHelper(ourContext);
                ourDatabase = ourHelper.getWritableDatabase();
                return this;
        }

        /* Function close terminates the SQLiteDatabase object's ed-
iting*/
        public void close() {
                ourHelper.close();
        }

        /*Function savePharmacies inserts a pharmacy's information
into table PHARMACIES*/
        public void savePharmacies(String name, String address,
String phone,
                        String latitude, String longitude) {
                String sql = "insert into PHARMACIES (name, phone,
address, latitude, longitude, distance, hours, mapId) " +
                                "values ('" + name + "','" + phone +
"','" + address + "','" + latitude + "','"
                                + longitude + "', '', '', '');";
                ourDatabase.execSQL(sql);
        }

        /*Function saveGasStations inserts a gas station's infor-
mation into table GASSTATIONS*/
        public void saveGasStations(String name, String address,
String brand,
                        String latitude, String longitude, String
hours, String price, String product, String prodId) {
                String sql = "insert into GASSTATIONS (name, brand,
address, latitude, longitude, distance, hours, mapId, " +
                                "price, product, prodId) " +
                                "values ('" + name + "','" + brand +
"','" + address + "','" + latitude + "','"
                                + longitude + "', '', '" + hours + "',
'', '" + price +"', + '" + product + "', '" + prodId + "');";
                ourDatabase.execSQL(sql);
        }

        /*Function deletePharmacies delete all PHARMACIES contents*/
        public void deletePharmacies(){
                ourDatabase.delete("PHARMACIES", null, null);
        }

        /*Function deleteGasStations delete GASSTATIONS contents for
product prod*/
        public void deleteGasStations(String prod){
                String sql = "delete from GASSTATIONS where prodId="
+ prod;
                ourDatabase.execSQL(sql);
```

```
        }

        /*Function deleteOnDutyPharmacies sets all pharmacies off
duty*/
        public void deleteOnDutyPharmacies(){
                String sql = "update PHARMACIES set hours='' where
hours<>''";
                ourDatabase.execSQL(sql);
        }
        /*Function setPharmacyOnDuty sets the pharmacy with that
name on duty*/
        public void setPharmacyOnDuty(String name, String hours) {
                String sql = "update PHARMACIES set hours='" + hours
+ "' where name='" + name + "'";
                ourDatabase.execSQL(sql);
        }

        /*Function setPharmacyOffDuty sets all pharmacies off duty*/
        public void setPharmacyOffDuty() {
                String sql = "update PHARMACIES set hours=''";
                ourDatabase.execSQL(sql);
        }

        /*Function setGassStationsNotShowing sets all gas stations
not showing map*/
        public void setGassStationsNotShowing() {
                String sql = "update GASSTATIONS set mapId=''";
                ourDatabase.execSQL(sql);
        }

        /*Function setPharmacyIndex sets the pharmacy with that id
showing on map with index mapId*/
        public void setPharmacyIndex(String id, String index) {
                String sql = "update PHARMACIES set mapId='" + index
+ "' where id='" + id + "'";
                ourDatabase.execSQL(sql);
        }

        /*Function setGasStationsIndex sets the gas station with
that id showing on map with index mapId*/
        public void setGasStationsIndex(String id, String index) {
                String sql = "update GASSTATIONS set mapId='" + index
+ "' where id='" + id + "'";
                ourDatabase.execSQL(sql);
        }

        /* Function setPharmacyDistance sets pharmacy's with that id
distance*/
        public void setPharmacyDistance(String id, String distance)
{
                String sql = "update PHARMACIES set distance='" +
distance + "' where id='" + id + "'";
                ourDatabase.execSQL(sql);
        }

        /* Function setGasStationDistance sets gas station's with
that id distance*/
        public void setGasStationDistance(String id, String dis-
tance) {
                String sql = "update GASSTATIONS set distance='" +
distance + "' where id='" + id + "'";
                ourDatabase.execSQL(sql);
```

```
        }

        /*Function setAllDistanceNull deletes all distances from ta-
bles PHARMACIES and GASSTATIONS*/
        public void setAllDistanceNull() {
                String sql = "update PHARMACIES set distance=''";
                ourDatabase.execSQL(sql);
                sql = "update GASSTATIONS set distance=''";
                ourDatabase.execSQL(sql);
        }

        /*Function fetchPharmacies fetches all pharmacies by that
order*/
        public List<List<String>> fetchPharmacies(String order) {
                Cursor c;
                List<List<String>> pharmacies = new Ar-
rayList<List<String>>();
                if(order.equals("distance")){
                        order = "cast(distance as real), name";
                }
                String sql = "select * from PHARMACIES order by " +
order;
                c = ourDatabase.rawQuery(sql, null);

                if (c != null) {
                        for (c.moveToFirst(); !c.isAfterLast();
c.moveToNext()) {
                                List<String> row = new Ar-
rayList<String>();
                                for (int i = 0; i < 8; i++) {
                                        row.add(c.getString(i));
                                }
                                pharmacies.add(row);
                        }
                        c.close();
                }
                return pharmacies;
        }

        /*Function fetchGasStations fetches alla gas stations with
product prod by that order*/
        public List<List<String>> fetchGasStations(String prod,
String order) {
                Cursor c;
                List<List<String>> gasStations = new Ar-
rayList<List<String>>();
                if(order.equals("distance")){
                        order = "cast(distance as real), brand";
                } else if (order.equals("price")) {
                        order += ", brand";
                }

                String sql = "select * from GASSTATIONS where
prodId=? order by " + order;
                c = ourDatabase.rawQuery(sql, new String[]{ prod });

                if (c != null) {
                        for (c.moveToFirst(); !c.isAfterLast();
c.moveToNext()) {
                                List<String> row = new Ar-
rayList<String>();
                                for (int i = 0; i < 11; i++) {
```

```
                                        row.add(c.getString(i));
                                }
                                gasStations.add(row);
                        }
                        c.close();
                }
                return gasStations;
        }

        /*Functon fetchPharmacysId identifies the farmacy's id by
it's index*/
        public String fetchPharmacysId(boolean isOnDuty, String in-
dex) {
                Cursor c;
                String sql = "select id from PHARMACIES where mapId=?
and ";
                if (isOnDuty){
                        sql += "hours<>''";
                } else {
                        sql += "hours=''";
                }
                c = ourDatabase.rawQuery(sql, new String[]{ index });
                c.moveToFirst();
                return c.getString(0);
        }

        /*Functon fetchGasStationsId identifies the gas station's id
by it's index*/
        public String fetchGasStationsId(String index){
                Cursor c;
                String sql = "select id from GASSTATIONS where
mapId=?";
                c = ourDatabase.rawQuery(sql, new String[]{ index });
                c.moveToFirst();
                return c.getString(0);
        }

        /*Function fetchPharmacysData fetches all data for pharmcy
with that id*/
        public String[] fetchPharmacysData(String id){
                Cursor c;
                String[] data = new String[9];
                String sql = "select * from PHARMACIES where id=?";
                c = ourDatabase.rawQuery(sql, new String[]{ id });
                if (c != null) {
                        c.moveToFirst();
                        for (int i = 0; i < 9; i++) {
                                data[i]=c.getString(i);
                        }
                }
                return data;
        }

        /*Function fetchGasStationsData fetches all data for gas
station with that id*/
        public String[] fetchGasStationsData(String id){
                Cursor c;
                String[] data = new String[12];
                String sql = "select * from GASSTATIONS where id=?";
                c = ourDatabase.rawQuery(sql, new String[]{ id });
                if (c != null) {
                        c.moveToFirst();
```

```
                        for (int i = 0; i < 12; i++) {
                                data[i]=c.getString(i);
                        }
                }
                return data;
        }

        /*Function countOnDuty returns the number of on duty pharma-
cies*/
        public int countOnDuty(){
                String sql = "select count(hours) from PHARMACIES
where hours<>''";
                c = ourDatabase.rawQuery(sql, null);
                c.moveToFirst();
                return c.getInt(0);
        }

        /* Function getIdFromName identifies a pharmacy's or gas
station's id by it's name*/
        public String getIdFromName(String name, int choice){
                String sql = "select id from ";
                if(choice==0){
                        sql+="PHARMACIES";
                }else if(choice==1){
                        sql+="GASSTATIONS";
                }
                sql+=" where name like '" + name + "'";
                c = ourDatabase.rawQuery(sql, null);
                c.moveToFirst();
                return c.getString(0);
        }

        /* Function isShowing checks if the pharmacy or gas station
with that name is showing on map*/
        public boolean isShowing(String name, int choice){
                String sql = "select mapId from ";
                if(choice==0){
                        sql+="PHARMACIES";
                }else if(choice==1){
                        sql+="GASSTATIONS";
                }
                sql+=" where name like '" + name + "'";
                c = ourDatabase.rawQuery(sql, null);
                c.moveToFirst();
                if(c.getString(0).equals("")){
                        return false;
                }else{
                        return true;
                }
        }
```

A.2.3. Pharmacies

```
/*Class Pharmacies contains functions related to pharmacies*/
public class Pharmacies {
        Context c;
        DBManager db;
        double latitude, longitude;
        boolean internet, isOnline, needUpdate;
        String dataStart = "", dataEnd = "", focushedIndex, fo-
cushedIsOnDuty;
```

```
        GeoPoint serresCenter = new GeoPoint( (int) (41.0887627 *
1E6), (int) (23.5496917 * 1E6) );
        MapView map;
        MapController controller;
        List<Overlay> overlayList;
        List<List<String>> farmakeia;
        CustomPinpoint farm, farm_ef, myLocation;
        private ProgressDialog dialog;
        Button btRefresh, btList, btSetts;

        /*Constructor Pharmacies initializes variables*/
        Pharmacies(Context c_) {
                c = c_;
                db = new DBManager(c);
        }

        /*Constructor Pharmacies initializes variables*/
        Pharmacies(Context c_, MapView map_, List<Overlay> over-
layList_,
                        CustomPinpoint farm_, CustomPinpoint
farm_ef_,
                        ProgressDialog dialog_) {
                c = c_;
                db = new DBManager(c);
                map = map_;
                overlayList = overlayList_;
                farm = farm_;
                farm_ef = farm_ef_;
                dialog = dialog_;
        }

        /*Constructor Pharmacies initializes variables*/
        Pharmacies(Context c_, MapView map_, MapController control-
ler_,   List<Overlay> overlayList_,
                        CustomPinpoint myLocation_,
List<List<String>> farmakeia_, CustomPinpoint farm_,
                        CustomPinpoint farm_ef_, ProgressDialog dia-
log_, boolean isOnline_, boolean needUpdate_,
                        Button btRefresh_, Button btList_, Button
btSetts_, String focushedIndex_,
                        String focushedIsOnDuty_) {
                c = c_;
                db = new DBManager(c);
                map = map_;
                controller = controller_;
                overlayList = overlayList_;
                myLocation = myLocation_;
                farmakeia = farmakeia_;
                farm = farm_;
                farm_ef = farm_ef_;
                dialog = dialog_;
                isOnline = isOnline_;
                needUpdate = needUpdate_;
                btRefresh = btRefresh_;
                btList = btList_;
                btSetts = btSetts_;
                focushedIndex = focushedIndex_;
                focushedIsOnDuty = focushedIsOnDuty_;
        }

        /*Function downloadAllPharmacies handles the download of the
catalog with all pharmacies*/
```

```
        public void downloadAllPharmacies() {
                String[] html = new String[] {

        "http://www.fsserron.gr/index.php?option=com_contact&view=ca
tegory&catid=12",

        "http://www.fsserron.gr/index.php?option=com_contact&view=ca
tegory&catid=12&limitstart=20",

        "http://www.fsserron.gr/index.php?option=com_contact&view=ca
tegory&catid=12&limitstart=40",

        "http://www.fsserron.gr/index.php?option=com_contact&view=ca
tegory&catid=12&limitstart=60" };
                ServerConnection sc = new ServerConnection(c);
                for (int i = 0; i < 4; i++) {
                        BufferedReader reader = sc.readHtml(html[i],
"utf-8");
                        getPharmacies(reader, "Pharmacy");
                }
        }

        /*Function downloadOnDutyPharmacies handles the download of
the on duty pharmacies info*/
        public void downloadOnDutyPharmacies() {
                ServerConnection sc = new ServerConnection(c);
                BufferedReader reader =
sc.readHtml("http://www.fsserron.gr/", "utf-8");
                getPharmacies(reader, "OnDuty");
        }

        /*Function getPharmacies seperates and stores the usefull
data of the pharmacies page source code*/
        public void getPharmacies(BufferedReader reader, String
flag) {
                if (flag.equals("Pharmacy")) {
                        dataStart = c.getString(R.string.RDP1);
                        dataEnd = "</tbody>";
                } else if (flag.equals("OnDuty")) {
                        dataStart = c.getString(R.string.RDP2);
                        dataEnd = "</td>";
                }
                String data = "";
                try {
                        boolean end = false;
                        StringBuilder builder = new StringBuild-
er(32768);

                        while ((build-
er.append(reader.readLine().trim())) != null
                                        && end == false) {
                                if (build-
er.toString().contains(dataStart)) {
                                        int a = build-
er.indexOf(dataStart);
                                        builder.delete(0, a + dataS-
tart.length());
                                }
                                if (build-
er.toString().contains(dataEnd)) {
                                        int a = build-
er.indexOf(dataEnd);
```

```
                                        builder.delete(a, build-
er.length());
                                        int b = builder.indexOf("<a
href=");
                                        builder.delete(0, b);
                                        end = true;
                                        data = builder.toString();
                                        data = data.replaceAll("\t",
"").replaceAll("&amp;", "&")

        .replaceAll("\\.", ". ");
                                }
                        }
                } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                } catch (Exception e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
                String[] test = data.split("<a href[^>]+\\>");
                String[][] results = new String[test.length][];
                for (int j = 1; j < test.length; j++) {
                        test[j] = test[j].replaceAll(">([0-9]|[0-
9][0-9]|[ ])<", "><");
                        test[j] = test[j].replaceAll("<[^>]+\\>",
"|");
                        results[j] = test[j].split("[\\|]+");
                        if(results[j].length!=0){
                                db.open();
                                if (flag.equals("Pharmacy")) {
                                        ServerConnection sc = new
ServerConnection(c);
                                        double[] loc =
sc.getCoordinartes(results[j][1]);
                                        latitude = loc[0];
                                        longitude = loc[1];

        db.savePharmacies(results[j][0].trim(), re-
sults[j][1].trim(),
                                                        re-
sults[j][2].trim(), String.valueOf(latitude),

        String.valueOf(longitude));
                                } else if (flag.equals("OnDuty")) {

        db.setPharmacyOnDuty(results[j][0].trim(), re-
sults[j][1].trim());
                                }
                                db.close();
                        }
                }
        }

        /* Function showPharmacies handles the pharmacies illustra-
tion on map*/
        public void showPharmacies() {
                try {
                        ((Activity) c).runOnUiThread(new Runnable() {
                                public void run() {
                                        if (farmakeia.size() != 0) {
```

```
                                        for (int i = 0; i <
farmakeia.size(); i++) {
                                                if
(!farmakeia.get(i).get(4).equals("0.0")
        && !farmakeia.get(i).get(5).equals("0.0")) {
                                                    GeoPoint
geo = new GeoPoint(
        (int) (Double.valueOf(farmakeia.get(i)
                        .get(4)) * 1E6), (int) (Double
                        .valueOf(farmakeia.get(i)
                                        .get(5)) * 1E6));
        db.open();
                                                    if
(!farmakeia.get(i).get(7).equals("")
        && !(!isOnline && needUpdate)) {
        OverlayItem store = new OverlayItem(geo,
                farmakeia.get(i).get(1),
                c.getString(R.string.isOnDuty) + " "
                                + farmakeia.get(i).get(7));
        farm_ef.insertPinpoint(store);
        String temp = String.valueOf(farm_ef.size() - 1);
        db.setPharmacyIndex(
                farmakeia.get(i).get(0),
                String.valueOf(farm_ef.size() - 1));
                                                    } else {
        OverlayItem store = new OverlayItem(geo,
                farmakeia.get(i).get(1),
                c.getString(R.string.isNotOnDuty));
        farm.insertPinpoint(store);
        String temp = String.valueOf(farm.size() - 1);
        db.setPharmacyIndex(
                farmakeia.get(i).get(0),
                String.valueOf(farm.size() - 1));
                                                    }
        db.close();
                                                }
                                        }
```

```
                                                overlayList.add(farm);
                                                over-
layList.add(farm_ef);
                                                if (myLocation == null
&& farm.size() > 0) {
                                                        control-
ler.animateTo(serresCenter);
                                                        control-
ler.setZoom(15);
                                                }

                                                if (!focushedIn-
dex.equals("")){

        if(focushedIsOnDuty.equals("")){

        farm.setFocus(farm.getItem(Integer.valueOf(focushedIndex)));
                                                                control-
ler.animateTo(farm.getItem(Integer.valueOf(focushedIndex)).getPoint(
));
                                                        } else {

        farm_ef.setFocus(farm_ef.getItem(Integer.valueOf(focushedInd
ex)));
                                                                control-
ler.animateTo(farm_ef.getItem(Integer.valueOf(focushedIndex)).getPoi
nt());
                                                        }
                                                }
                                                dialog.dismiss();
                                                map.invalidate();

        map.setKeepScreenOn(false);
                                                map.setEnabled(true);
                                                btRe-
fresh.setEnabled(true);

        btList.setEnabled(true);

        btSetts.setEnabled(true);
                                        }
                                }
                        });
                } catch (final Exception ex) {
                        Log.i("---", "Exception in thread");
                }
        }

        /* Function deletePharmacies deletes all pharmacies data and
removes pharmacies from map*/
        public void deletePharmacies(final String flag) {
                try {
                        ((Activity) c).runOnUiThread(new Runnable() {
                                public void run() {
                                        dialog.setCancelable(false);
                                        dia-
log.setMessage("Loading...");
                                        dialog.show();

                                        db.open();
                                        if (flag.equals("onDuty")) {
```

```
        db.deleteOnDutyPharmacies();
                                        } else {
                                                db.deletePharmacies();
                                        }
                                        db.close();

                                        map.setKeepScreenOn(true);
                                        farm.hideAllBalloons();
                                        farm_ef.hideAllBalloons();
                                        overlayList.remove(farm);
                                        overlayList.remove(farm_ef);
                                        map.invalidate();
                                }
                        });
                } catch (final Exception ex) {
                        Log.i("---", "Exception in thread");
                }
        }
}
```

A.2.4. GasStations

```
/*Class GasStations contains functions related to gas stations*/
public class GasStations {
        Context c;
        DBManager db;
        double latitude, longitude;
        boolean internet, isOnline;
        String dataStart = "</select>", dataEnd = "</p>", fo-
cushedIndex;
        GeoPoint serresCenter = new GeoPoint( (int) (41.0887627 *
1E6), (int) (23.5496917 * 1E6) );
        MapView map;
        MapController controller;
        List<Overlay> overlayList;
        CustomPinpoint myLocation;
        List<List<String>> gasStations;
        CustomPinpoint gas;
        private ProgressDialog dialog;
        Button btRefresh, btList, btSetts;

        /*Constructor GasStations initializes variables*/
        GasStations(Context c_) {
                c = c_;
                db = new DBManager(c);
        }

        /*Constructor GasStations initializes variables*/
        GasStations(Context c_, MapView map_, List<Overlay> over-
layList_,
                        CustomPinpoint gas_, ProgressDialog dialog_)
{
                c = c_;
                db = new DBManager(c);
                map = map_;
                overlayList = overlayList_;
                gas = gas_;
                dialog = dialog_;
        }
```

```
	/*Constructor GasStations initializes variables*/
	GasStations(Context c_, MapView map_, MapController control-
ler_,
			List<Overlay> overlayList_, CustomPinpoint
myLocation_,
			List<List<String>> gasStations_, Cus-
tomPinpoint gas_,
			ProgressDialog dialog_, boolean isOnline_,
Button btRefresh_,
			Button btList_, Button btSetts_, String fo-
cushedIndex_) {
		c = c_;
		db = new DBManager(c);
		map = map_;
		controller = controller_;
		overlayList = overlayList_;
		myLocation = myLocation_;
		gasStations = gasStations_;
		gas = gas_;
		dialog = dialog_;
		isOnline = isOnline_;
		btRefresh = btRefresh_;
		btList = btList_;
		btSetts = btSetts_;
		focushedIndex = focushedIndex_;
	}

	/*Function downloadGasStations handles the download of gas
station's with product prod information*/
	public void downloadGasStations(String prod) {
		String html =
"http://www.fuelprices.gr/CheckPrices?DD=62010200&DD=62010300&DD=620
10100&DD=62010400&date_filter=7&prodclass="
				+ prod;
		ServerConnection sc = new ServerConnection(c);
		BufferedReader reader = sc.readHtml(html, "iso-8859-
7");
		getGasStations(reader, prod);
	}

	/*Function getGasStations seperates and stores the usefull
data of the gas stations page source code*/
	public void getGasStations(BufferedReader reader, String
prod) {
		String data = "";
		try {
			boolean end = false;
			StringBuilder builder = new StringBuild-
er(32768);

			while ((build-
er.append(reader.readLine().trim())) != null
					&& end == false) {
				if (build-
er.toString().contains(dataStart)) {
					int a = build-
er.indexOf(dataStart);
					builder.delete(0, a + dataS-
tart.length());
				}
				if (build-
er.toString().contains(dataEnd)) {
```

```
                                        int a = build-
er.indexOf(dataEnd);
                                        builder.delete(a, build-
er.length());
                                        int b = build-
er.indexOf("<tr>");
                                        builder.delete(0, b);
                                        end = true;
                                        data = builder.toString();
                                        data = data.replaceAll("\t",
"").replaceAll("&amp;", "&")

        .replaceAll("\\.", ". ").replaceAll(" ", "");
                                }
                        }
                } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                } catch (Exception e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
                String[] test = data.split("top");
                String[][] results = new String[test.length][];

                for (int j = 1; j < test.length; j++) {
                        test[j] = test[j].replaceAll("<[^>]+\\>",
"|");
                        test[j] =
test[j].replace(c.getString(R.string.RDG1), "")

        .replace(c.getString(R.string.RDG2), "")

        .replace(c.getString(R.string.RDG3), "")

        .replace(c.getString(R.string.RDG4), "")

        .replace(c.getString(R.string.RDG5), "").replace(": ", "");
                        results[j] = test[j].split("[\\|]+");
                        ServerConnection sc = new ServerConnec-
tion(c);
                        double[] loc =
sc.getCoordinartes(results[j][4]);
                        latitude = loc[0];
                        longitude = loc[1];
                        db.open();
                        db.saveGasStations(results[j][5].trim(), re-
sults[j][4].trim(),
                                        results[j][3].trim(),
String.valueOf(latitude),
                                        String.valueOf(longitude), re-
sults[j][6].trim(),
                                        results[j][1].trim(), re-
sults[j][2].trim(), prod);
                        db.close();
                }
        }

        /* Function deleteGasStations deletes all gas stations data
and removes gas stations from map*/
        public void deleteGasStations(final String prod) {
                try {
```

```
                        ((Activity) c).runOnUiThread(new Runnable() {
                                public void run() {
                                        dialog.setCancelable(false);
                                        dia-
log.setMessage("Loading...");
                                        dialog.show();

                                        db.open();
                                        db.deleteGasStations(prod);
                                        db.close();
                                        map.setKeepScreenOn(true);
                                        gas.hideAllBalloons();
                                        overlayList.remove(gas);
                                        map.invalidate();
                                }
                        });
                } catch (final Exception ex) {
                        Log.i("---", "Exception in thread");
                }
        }

        /* Function showGasStations handles the gas stations illus-
tration on map*/
        public void showGasStations() {
                try {
                        ((Activity) c).runOnUiThread(new Runnable() {
                                public void run() {
                                        if (gasStations.size() != 0) {
                                                db.open();

        db.setGassStationsNotShowing();
                                                db.close();
                                                for (int i = 0; i <
gasStations.size(); i++) {
                                                        if (!gasSta-
tions.get(i).get(4).equals("0.0")

        && !gasStations.get(i).get(5).equals("0.0")) {
                                                                GeoPoint
geo = new GeoPoint(

        (int) (Double.valueOf(gasStations

                        .get(i).get(4)) * 1E6),

        (int) (Double.valueOf(gasStations

                        .get(i).get(5)) * 1E6));

        db.open();
                                                                Over-
layItem store = new OverlayItem(geo,

        gasStations.get(i).get(2), gasStations.get(i).get(1) + "\n"

                        + gasStations.get(i).get(10) + "\t" +
c.getString(R.string.price2)

                        + gasStations.get(i).get(9) + "\n" +
c.getString(R.string.registered)

                        + gasStations.get(i).get(7));
```

```
        gas.insertPinpoint(store);
                                                                String
temp = String.valueOf(gas.size() - 1);

        db.setGasStationsIndex(gasStations.get(i)

        .get(0), String.valueOf(gas.size() - 1));


        db.close();
                                                        }
                                                }
                                                overlayList.add(gas);
                                                if (myLocation == null
&& gas.size() > 0) {
                                                        control-
ler.animateTo(serresCenter);
                                                        control-
ler.setZoom(15);
                                                }
                                                if (!focushedIn-
dex.equals("")){

        gas.setFocus(gas.getItem(Integer.valueOf(focushedIndex)));
                                                        control-
ler.animateTo(gas.getItem(Integer.valueOf(focushedIndex)).getPoint()
);
                                                }
                                                dialog.dismiss();
                                                map.invalidate();

        map.setKeepScreenOn(false);
                                                map.setEnabled(true);
                                                btRe-
fresh.setEnabled(true);

        btList.setEnabled(true);

        btSetts.setEnabled(true);
                                        }
                                }
                        });
                } catch (final Exception ex) {
                        Log.i("---", "Exception in thread");
                }
        }
}
```

A.2.5. CustomPinpoint

```
/*Class CustomPinpoint handles the OverlayItems*/
public class CustomPinpoint extends BalloonItemizedO-
verlay<OverlayItem> {

        private Context c;
        private ArrayList<OverlayItem> pinpoints = new Ar-
rayList<OverlayItem>();

        /*Constructor CustomPinpoint initializes variables*/
```

```
        public CustomPinpoint(Drawable defaultMarker, MapView
mapView) {
                super(boundCenter(defaultMarker), mapView);
                c = mapView.getContext();
                // TODO Auto-generated constructor stub
                populate();
        }

        @Override
        protected OverlayItem createItem(int i) {
                // TODO Auto-generated method stub
                return pinpoints.get(i);
        }

        /*Function size returnsthe size of the CustomPinpoint list*/
        @Override
        public int size() {
                // TODO Auto-generated method stub
                return pinpoints.size();
        }

        /*Function insertPinpoint insert an OverlayItem to the Cus-
tomPinpoint list*/
        public void insertPinpoint (OverlayItem item) {
                pinpoints.add(item);
                this.populate();
        }

        /*Function onBalloonTap sends the id of taped item to MapAc-
tivity*/
        @Override
        protected boolean onBalloonTap(int index, OverlayItem item)
{
                DBManager db = new DBManager(c);
                String id = "";
                if(!item.getSnippet().equals("")){
                        db.open();

        if(item.getSnippet().equals(c.getString(R.string.isNotOnDuty
))) {
                                id = db.fetchPharmacysId(false,
String.valueOf(index));
                        }else
if(item.getSnippet().startsWith(c.getString(R.string.isOnDuty))){
                                id = db.fetchPharmacysId(true,
String.valueOf(index));
                        }else{
                                id =
db.fetchGasStationsId(String.valueOf(index));
                        }
                        db.close();
                        Intent intent = new Intent();
                        Bundle b = new Bundle();
                        b.putString("id", id);
                        intent.setAction("BalloonItemizedOverlay");
                        intent.putExtras(b);
                        c.sendBroadcast(intent);
                }
                return true;
        }
}
```

A.2.6. RoutePathOverlay

```
/*Class RoutePathOverlay handles the creation of a route path*/
public class RoutePathOverlay extends Overlay {

    private int _pathColor;
    private final List<GeoPoint> _points;
    private boolean _drawStartEnd;

    /*Constructor RoutePathOverlay*/
    public RoutePathOverlay(List<GeoPoint> points) {
            this(points, Color.RED, true);
    }

    /*Constructor RoutePathOverlay initializes variables*/
    public RoutePathOverlay(List<GeoPoint> points, int pathColor,
boolean drawStartEnd) {
            _points = points;
            _pathColor = pathColor;
            _drawStartEnd = drawStartEnd;
    }

    /*Function drawOval creates route's curved lines*/
    private void drawOval(Canvas canvas, Paint paint, Point point) {
            Paint ovalPaint = new Paint(paint);
            ovalPaint.setStyle(Paint.Style.FILL_AND_STROKE);
            ovalPaint.setStrokeWidth(2);
            int _radius = 6;
            RectF oval = new RectF(point.x - _radius, point.y -
_radius, point.x + _radius, point.y + _radius);
            canvas.drawOval(oval, ovalPaint);
    }

    /*Function draw creates the route*/
    public boolean draw(Canvas canvas, MapView mapView, boolean
shadow, long when) {
            Projection projection = mapView.getProjection();
            if (shadow == false && _points != null) {
                    Point startPoint = null, endPoint = null;
                    Path path = new Path();
                    //We are creating the path
                    for (int i = 0; i < _points.size(); i++) {
                            GeoPoint gPointA = _points.get(i);
                            Point pointA = new Point();
                            projection.toPixels(gPointA, pointA);
                            if (i == 0) { //This is the start point
                                    startPoint = pointA;
                                    path.moveTo(pointA.x, pointA.y);
                            } else {
                                    if (i == _points.size() -
1)//This is the end point
                                            endPoint = pointA;
                                    path.lineTo(pointA.x, pointA.y);
                            }
                    }

                    Paint paint = new Paint();
                    paint.setAntiAlias(true);
                    paint.setColor(_pathColor);
                    paint.setStyle(Paint.Style.STROKE);
                    paint.setStrokeWidth(5);
                    paint.setAlpha(90);
```

```
                    if (getDrawStartEnd()) {
                            if (startPoint != null) {
                                    drawOval(canvas, paint,
startPoint);
                            }
                            if (endPoint != null) {
                                    drawOval(canvas, paint, end-
Point);
                            }
                    }
                    if (!path.isEmpty())
                            canvas.drawPath(path, paint);
            }
            return super.draw(canvas, mapView, shadow, when);
    }

    /*Function getDrawStartEnd checks if destination s initialized*/
    public boolean getDrawStartEnd() {
            return _drawStartEnd;
    }

    public void setDrawStartEnd(boolean markStartEnd) {
            _drawStartEnd = markStartEnd;
    }
}
```

A.3. layouts

A.3.1 main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This layout represents the Main Menu and is used by the Main
Activity -->
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent" >

    <ImageView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:src="@drawable/serres"
        android:adjustViewBounds="true"
        android:scaleType="centerCrop" />

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:gravity="center"
        android:orientation="vertical" >

        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginBottom="50dp"
            android:gravity="center"
            android:text="@string/app_name"
            android:textColor="#000000"
            android:textColorHighlight="#FFFFFF"
            android:textSize="45sp"
            android:textStyle="bold" />
```

```
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:orientation="vertical" >

            <Button
                android:id="@+id/pharmacyBt"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:drawableLeft="@drawable/pharmacies"
                android:ems="8"
                android:padding="10dp"
                android:scaleType="centerInside"
                android:text="@string/Pharmacies"
                android:textSize="25sp"
                android:textColor="#000000" />

            <Button
                android:id="@+id/gasstationBt"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:drawableLeft="@drawable/gasstations"
                android:ems="8"
                android:padding="10dp"
                android:scaleType="centerInside"
                android:text="@string/GasStations"
                android:textSize="25sp"
                android:textColor="#000000" />
        </LinearLayout>
    </LinearLayout>

</RelativeLayout>
```

A.3.2 map.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This layout represent the map form and it used by the Activi-
tyMap Activity -->
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <RelativeLayout
        android:id="@+id/headerlayout"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" >

        <com.google.android.maps.MapView
            android:id="@+id/mvMap"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:apiKey="05U_W5NSkI4QeIoS7Q14Ikl7TG20s89Xytp92_g"
            android:clickable="true"
            android:state_enabled="true" />

        <Button
        android:id="@+id/btList"
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:layout_alignTop="@+id/mvMap"
        android:layout_alignRight="@+id/mvMap"
        android:layout_marginTop="10dp"
        android:layout_marginRight="10dp"
               android:drawableLeft="@drawable/list1" />

        <Button
        android:id="@+id/btRefresh"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignTop="@+id/mvMap"
        android:layout_toLeftOf="@+id/btList"
        android:layout_marginTop="10dp"
        android:layout_marginRight="5dp"
               android:drawableLeft="@drawable/refresh1" />

        <Button
        android:id="@+id/btSetts"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignTop="@+id/mvMap"
        android:layout_toLeftOf="@+id/btRefresh"
        android:layout_marginTop="10dp"
        android:layout_marginRight="5dp"
               android:drawableLeft="@drawable/setts" />

        <Button
        android:id="@+id/btHome"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignTop="@+id/mvMap"
        android:layout_alignLeft="@+id/mvMap"
        android:layout_marginTop="10dp"
        android:layout_marginRight="10dp"
               android:drawableLeft="@drawable/home" />

    </RelativeLayout>
</LinearLayout>
```

A.3.3 list.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- This layout represent the list form and it is used by the Ac-
tivityList Activity -->
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <Button
        android:id="@+id/btOrder"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_alignParentTop="true"
        android:layout_marginRight="10dp"
        android:layout_marginTop="10dp"
        android:drawableLeft="@drawable/sort" />

    <Button
```

```
        android:id="@+id/btHome"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginRight="10dp"
        android:layout_marginTop="10dp"
        android:drawableLeft="@drawable/home" />

    <Button
        android:id="@+id/btBack"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_marginRight="10dp"
        android:layout_marginTop="10dp"
        android:layout_toRightOf="@+id/btHome"
        android:drawableLeft="@drawable/back" />

    <LinearLayout
        android:id="@+id/window"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_below="@+id/btHome"
        android:orientation="vertical" >

        <ExpandableListView
            android:id="@+id/android:list"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:layout_marginTop="0dp" />

        <TextView
            android:id="@+id/android:empty"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:padding="20dp"
            android:text="@string/main_no_items"
            android:textSize="30sp" />
    </LinearLayout>
</RelativeLayout>
```

A.3.4 data_show.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This layout represents the information form and it is used by
the DataShow Activity -->
<ScrollView
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <RelativeLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent" >

        <Button
            android:id="@+id/btHome"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
```

```
            android:layout_alignParentLeft="true"
            android:layout_alignParentTop="true"
            android:layout_marginRight="10dp"
            android:layout_marginTop="10dp"
            android:drawableLeft="@drawable/home" />

        <Button
            android:id="@+id/btBack"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentTop="true"
            android:layout_marginRight="10dp"
            android:layout_marginTop="10dp"
            android:layout_toRightOf="@+id/btHome"
            android:drawableLeft="@drawable/back" />
<LinearLayout
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:layout_below="@+id/btHome"
            android:orientation="vertical" >

            <EditText
                android:id="@+id/titleTv"
                android:layout_width="fill_parent"
                android:layout_height="fill_parent"
                android:layout_marginBottom="10dp"
                android:editable="false"
                android:focusable="false"
                android:gravity="center"
                android:paddingBottom="20dp"
                android:textSize="25sp" />

            <TextView
                android:id="@+id/detailsTv"
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:gravity="center"
                android:padding="10dp"
                android:textSize="20sp" />

            <TableLayout
                android:id="@+id/tableTl"
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:background="#000000"
                android:divider="#000000"
                android:dividerHeight="0dp"
                android:scrollingCache="true" >
            </TableLayout>
        </LinearLayout>
    </RelativeLayout>
</ScrollView>
```

A.3.5 data_show_grid.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This layout represents a table row and it is udes by DataShow
Activity -->
<TableRow xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal" >
    <TextView
```

```
        android:id="@+id/etWhat"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_weight="70"
        android:padding="5dp"
        android:background="#FFFFFF"
        android:editable="false"
        android:ems="10"
        android:focusable="false"
        android:text="TextView"
        android:textColor="#000000"
        android:textSize="20sp" />

    <TextView
        android:id="@+id/etValue"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_weight="30"
        android:padding="5dp"
        android:background="#FFFFFF"
        android:editable="false"
        android:ems="10"
        android:autoLink="all"
        android:focusable="false"
        android:text="TextView"
        android:textColor="#000000"
        android:textSize="20sp" />
</TableRow>
```

A.3.6. child_row.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This layout represents the expandable list chid row and it is
used by ActivityList Activity-->
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content">

    <TextView android:id="@+id/childname"
        android:layout_marginLeft="30dp"
         android:textStyle="italic"
         android:layout_width="fill_parent"
         android:layout_height="wrap_content"
               android:textSize="25sp"
               android:padding="10dp"/>
</LinearLayout>
```

A.3.7. dialog_listview.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This dialog represents a dialog list and it's used by the Ac-
tivityMap Activity
for pharmacies refresh dialog and choose gas product dialog -->
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
```

```
    <ListView
        android:id="@+id/dialogLv"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_marginTop="10dp"
        android:textSize="20sp"
        android:cacheColorHint="#00000000" />
</LinearLayout>
```

A.3.8. dialog_order.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This dialog represents the order dialog that is ashowing on
ActivityList Activity -->
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:padding="10dp"
        android:text="@string/sortBy"
        android:textSize="25sp" />

    <View
        android:layout_width="fill_parent"
        android:layout_height="2dp"
        android:layout_marginBottom="10dp"
        android:background="#FFFFFF" />

    <ListView
        android:id="@+id/lvOrder"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content" />
</LinearLayout>
```

A.3.9. route_settings_dialog.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This layout represents the settings dialog that is showing on
ActivityMap Activity -->
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <CheckBox
        android:id="@+id/routeCb"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:paddingLeft="45dp"
        android:paddingRight="10dp"
        android:paddingBottom="10dp"
        android:paddingTop="10dp"
        android:text="@string/routeOnOff"
        android:textSize="20sp" />
```

```
    <LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_marginTop="10dp"
    android:orientation="horizontal" >

        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_weight="65"
            android:padding="5dp"
            android:text="@string/routeMode"
            android:textSize="20sp" />

        <Spinner
            android:id="@+id/routeSp"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_weight="35" />
    </LinearLayout>

    <Button
        android:id="@+id/routeBt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:layout_gravity="center"
        android:text="@string/saveBt"
        android:textSize="20sp" />
</LinearLayout>
```

A.3.10.simple_textview.xml

```
<!-- This layout is used as a row on other layout's listviews.
That other layouts are dialog_listview.xml and dialog_order.xml.
It is used by ActivityMap and ActivityList Activities. -->
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
 android:id="@+id/etList"
 android:layout_width="fill_parent"
 android:layout_height="wrap_content"
 android:padding="7dp"
 android:singleLine="true"
 android:textSize="25sp" >
</TextView>
```

A.4. values

A.4.1. strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">Serres Guide</string>
       <string name="main_no_items">No items</string>
    <string name="Pharmacies">Pharmacies</string>
    <string name="GasStations">Gas Stations</string>
    <string name="routeOnOff">Show instructions to destina-
tion.</string>
    <string name="routeMode">Mode :\u0020</string>
    <string name="saveBt">Save</string>
    <string name="ONDUTY">ON DUTY</string>
```

```
    <string name="ALL">ALL</string>
    <string name="GASSTATIONS">GAS STATIONS</string>
    <string name="notShowingOnMap">Not showing on map.</string>
    <string name="sortBy">Sort by :</string>
    <string name="name">Name</string>
    <string name="distance">Distance</string>
    <string name="price">Price</string>
    <string name="myLocation">My location</string>
    <string name="product">Product :</string>
    <string name="unleaded95oct">Unleaded 95 oct.</string>
    <string name="unleaded100oct">Unleaded 100 oct.</string>
    <string name="super_gas">Super</string>
    <string name="dieselD">Diesel for driving</string>
    <string name="dieselH">Diesel for heating</string>
    <string name="autogas">Autogas</string>
    <string name="dieselHD">Diesel for heating domiciliary</string>
    <string name="refresh">Refresh :</string>
    <string name="onDuty">On duty</string>
    <string name="all">All</string>
    <string name="noRouteWarning">Quite a distance! The route will
not be displayed.</string>
    <string name="noInternetWarning">There\'s a problem with the
connection...</string>
    <string name="settings">Settings :</string>
    <string name="car">car</string>
    <string name="bike">bike</string>
    <string name="onFoot">on foot</string>
    <string name="isNotOnDuty">Not on duty today.</string>
    <string name="isOnDuty">On duty</string>
    <string name="trademark">Trademark :\u0020</string>
    <string name="address">Address :</string>
    <string name="distance1">Distance :</string>
    <string name="price1">Price :</string>
    <string name="price2">Price:\u0020</string>
    <string name="regDate">Registration Date/Time :</string>
    <string name="phoneNumber">Phone Number :</string>
    <string name="registered">Regisred: </string>
    <string name="RDG1">Τιμή:\u0020</string>
    <string name="RDG2">Προϊόν:\u0020</string>
    <string name="RDG3">Σήμα:\u0020</string>
    <string name="RDG4">Διεύθυνση:\u0020</string>
    <string name="RDG5">Ημερ.\u0020\u0020/\u0020Ωρα δήλωσης</string>
    <string name="RDP1">Τηλέφωνο</string>
    <string name="RDP2">Σήμερα</string>
</resources>
```

A.5. values-el

A.5.1. string.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">Οδηγός Σερρών</string>
       <string name="main_no_items">No items</string>
    <string name="Pharmacies">Φαρμακεία</string>
    <string name="GasStations">Πρατήρια Καυσίμων</string>
    <string name="routeOnOff">Εμφάνιση οδηγιών προς τον
προορισμό.</string>
    <string name="routeMode">Μέσο :\u0020</string>
    <string name="saveBt">Αποθήκευση</string>
    <string name="ONDUTY">ΕΦΗΜΕΡΕΥΟΝΤΑ</string>
```

```
    <string name="ALL">ΟΛΑ</string>
    <string name="GASSTATIONS">ΒΕΝΖΙΝΑΔΙΚΑ</string>
    <string name="notShowingOnMap">Δεν εμφανίζεται στο
χάρτη.</string>
    <string name="sortBy">Ταξινόμηση κατά :</string>
    <string name="name">Επωνυμία</string>
    <string name="distance">Απόσταση</string>
    <string name="price">Τιμή</string>
    <string name="myLocation">Η τοποθεσία μου</string>
    <string name="product">Προιόν :</string>
    <string name="unleaded95oct">Αμόλυβδη 95 οκτ.</string>
    <string name="unleaded100oct">Αμόλυβδη 100 οκτ.</string>
    <string name="super_gas">Super</string>
    <string name="dieselD">Diesel Κίνησης</string>
    <string name="dieselH">Diesel Θέρμανσης</string>
    <string name="autogas">Υγραέριο κίνησης (Autogas)</string>
    <string name="dieselHD">Diesel Θέρμανσης Κατ΄οίκον</string>
    <string name="refresh">Ανανέωση :</string>
    <string name="onDuty">Εφημερέυοντα</string>
    <string name="all">Όλα</string>
    <string name="noRouteWarning">Πολύ μεγάλη απόσταση! Η διαδρομή
δεν θα εμφανιστεί.</string>
    <string name="noInternetWarning">Υπάρχει κάποιο πρόβλημα στη
σύδεση...</string>
    <string name="settings">Ρυθμίσεις :</string>
    <string name="car">αυτοκίνητο</string>
    <string name="bike">ποδήλατο</string>
    <string name="onFoot">με τα πόδια</string>
    <string name="isNotOnDuty">Δεν εφημερέυει σήμερα.</string>
    <string name="isOnDuty">Εφημερέυει</string>
    <string name="trademark">Σήμα :\u0020</string>
    <string name="address">Διεύθυνση :</string>
    <string name="distance1">Απόσταση :</string>
    <string name="price1">Τιμή :</string>
    <string name="price2">Τιμή:\u0020</string>
    <string name="regDate">Ημερ./Ώρα δήλωσης :</string>
    <string name="phoneNumber">Τηλέφωνο :</string>
    <string name="registered">Καταχωρήθηκε:\u0020</string>
</resources>
```

Παράρτημα Β – Ανάπτυξη εφαρμογής διαδραστικού χάρτη

Β.1. Δημιουργία ενός Νέου Project Android

1. Πραγματοποιείται εκκίνηση του Eclipse και επιλογή του New από την γραμμή εργαλείων.
2. Γίνεται άνοιγμα του φάκελου Android, από το παράθυρο που εμφανίζεται, επίσης γίνονται επιλογή του Android Application Project και κλικ στο Next.
3. Συμπληρώνεται η φόρμα που θα εμφανιστεί:
    - Application Name: Το όνομα της εφαρμογής όπως εμφανίζεται στο χρήστη. Σε αυτό το παράδειγμα συμπληρώνεται με "My First Map App".
    - Project Name: Το όνομα του φακέλου του project και το όνομα που θα εμφανίζεται στο Eclipse. Σε αυτό το παράδειγμα συμπληρώνεται με "MyFirstMapApp".
    - Package Name: Το namespace του πακέτου της εφαρμογής(ισχύουν οι ίδιοι κανόνες όπως και στα πακέτα των Java εφαρμογών). Το όνομα του πακέτου, λειτουργεί ως αναγνωριστικό, συνεπώς πρέπει να είναι μοναδικό, ανάμεσα σε όλα τα πακέτα που είναι εγκατεστημένα στο Android σύστημα. Γι' αυτό το λόγο προτιμάται η χρήση ενός ονόματος που να ξεκινάει με το αντεστραμμένο domain name του οργανισμού ή τον φορέα του εκδότη. Σε αυτό το παράδειγμα συμπληρώνεται με "com.example.myfirstmapapp". Ωστόσο δεν μπορεί να γίνει μεταφόρτωση εφαρμογής στο Google Play με χρήση του namespace "com.example".
    - Minimum Required SDK: Η παλαιότερη έκδοση Android που υποστηρίζει η εφαρμογή. Υποδεικνύεται χρησιμοποιώντας το επίπεδο API. Για να υποστηρίζεται η εφαρμογή από όσες περισσότερες συσκευές γίνεται, επιλέγεται η χαμηλότερη έκδοση

που επιτρέπει όμως στην εφαρμογή να παρέχει το σύνολο των βασικών χαρακτηριστικών της. Αν κάποιο χαρακτηριστικό είναι διαθέσιμο μόνο σε νεότερες εκδόσεις Android και δεν είναι κρίσιμο για την ορθή λειτουργία της εφαρμογής μπορεί να ενεργοποιείται μόνο στις συσκευές που το υποστηρίζουν. Σε αυτό το παράδειγμα, παραμένει το προεπιλεγμένο επίπεδο API.

- Target SDK: Η νεότερη έκδοση Android με την οποία έχει δοκιμαστεί η εφαρμογή. Προτιμάται η εφαρμογή να δοκιμάζεται στη νεότερη έκδοση android που κυκλοφορεί, ώστε να αξιοποιούνται τα νέα χαρακτηριστικά. Σε αυτό το παράδειγμα, παραμένει το προεπιλεγμένο επίπεδο API.
- Compile with: Η έκδοση Android με την οποία γίνεται compile η εφαρμογή. Προεπιλεγμένη είναι η νεότερη έκδοση διαθέσιμη στο SDK. Έτσι, η εφαρμογή μπορεί να υποστηρίζει παλαιότερες εκδόσεις Android, αλλά μπορεί και να χρησιμοποιεί τα νέα χαρακτηριστικά όταν είναι δυνατό. Σε αυτό το παράδειγμα, παραμένει το προεπιλεγμένο επίπεδο API.
- Theme: Το στυλ του Android UI που εφαρμόζεται στην εφαρμογή. Σε αυτό το παράδειγμα, παραμένει το προεπιλεγμένο θέμα.

4. Επιλέγεται το Next και στην επόμενη οθόνη "Configure Project" αφήνονται οι προεπιλεγμένες τιμές και πάλι επιλέγεται το Next.
5. Στην επόμενη οθόνη ορίζεται το launcher εικονίδιο της εφαρμογής. Σε αυτήν την περίπτωση απλά επιλέγεται το Next.
6. Στην οθόνη "Create Activity" επιλέγεται ένα activity template από το οποίο ξεκινά το χτίσιμο της εφαρμογής. Επιλέγεται το Blank Activity και στη συνέχεια το Next.
7. Σε αυτήν την περίπτωση αφήνονται οι προεπιλεγμένες τιμές και επιλέγεται το Finish.

Β.2. Υλοποίηση Περίπτωσης Χρήσης : Προβολή Χάρτη

Αρχικά εντοπίζεται το project MyFirstMapApp στον Package Explorer του Eclipse και γίνεται δεξί-κλικ πάνω του. Επιλέγεται Properties > Android. Εκεί επιλέγονται Google APIs για πλατφόρμα 4.2.2, αντί για Android 4.2.2 και Ok. *Έτσι προστίθεται η εξωτερική βιβλιοθήκη των Google APIs στο project.*

Επίσης διασφαλίζεται ότι έχει κατεβεί η έκδοση Android 4.2.2 (API 17), καθώς και όλα της τα πρόσθετα όπως το Google APIs, από τον SDK Manager.

Αρχείο Manifest :

Σε αυτό το παράδειγμα, προστίθεται στο αρχείο Manifest.xml ένα στοιχείο <uses-permision> για άδεια χρήσης Internet, ως παιδί του στοιχείου <manifest>. Η άδεια χρήσης Internet απαιτείται ώστε να κατέβουν τα δεδομένα του χάρτη. Επίσης, προστίθεται ένα στοιχείο <uses-library> με αναφορά στην εξωτερική βιβλιοθήκη com.google.android.maps, ώς παιδί του στοιχείου <application>.

Manifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
        package="com.example.myfirstmapapp"
        android:versionCode="1"
        android:versionName="1.0" >
        <uses-sdk android:minSdkVersion="8"
                        android:targetSdkVersion="17" />
        <uses-permission
                android:name="android.permission.INTERNET"/>
        <application android:allowBackup="true"
                        android:icon="@drawable/ic_launcher"
                        android:label="@string/app_name"
                        android:theme="@style/AppTheme" >
                <uses-library android:name="com.google.android.maps"
/>
                <activity an-
droid:name="com.example.myfirstmapapp.MainActivity"
                        android:label="@string/app_name" >
            <intent-filter>
                        <action an-
droid:name="android.intent.action.MAIN" />
                        <category
                        an-
droid:name="android.intent.category.LAUNCHER" />
            </intent-filter>
         </activity>
    </application>
</manifest>
```

Αρχείο UI:

Ο φάκελος res περιέχει διάφορους υποκαταλόγους με τους πόρους της εφαρμογής. Ένας από αυτούς είναι ο κατάλογος layout που περιέχει τα .xml αρχεία που ορίζουν το UI της εφαρμογής.

Πραγματοποιείται το άνοιγμα του αρχείου activity_main.xml. Στην περίπτωση που είναι επιλεγμένη η καρτέλα Graphical Layout, επιλέγεται η καρτέλα activity_main.xml. Διαγράφεται το στοιχείο TextView και στη θέση του προστίθεται ένα στοιχείο MapView όπως φαίνεται παρακάτω. Στο χαρακτηριστικό apiKey προστίθεται το Google Maps API v1 Key του Debug SDK πιστοποιητικού (βλ. Κεφ 2).

activity_main.xml:

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"
```

```
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingBottom="@dimen/activity_vertical_margin"
        android:paddingLeft="@dimen/activity_horizontal_margin"
        android:paddingRight="@dimen/activity_horizontal_margin"
        android:paddingTop="@dimen/activity_vertical_margin"
        tools:context=".MainActivity" >
                <com.google.android.maps.MapView
                        android:id="@+id/mvMap"
                        android:layout_width="fill_parent"
                        android:layout_height="fill_parent"
       android:apiKey="Εισάγεται τοGoogle Maps API v1 KEY " />
</RelativeLayout>
```

Αρχείο Κλάσης:

Ο φάκελος res περιέχει όλα τα αρχεία των κλάσεων της εφαρμογής. Πραγματοποιείται άνοιγμα του αρχείου MyFirstMapApp.java και ακολουθούνται οι παρακάτω οδηγίες:

1. Αλλάζεται η επέκταση της κλάσης MainActivity από Activity σε MapActivity. Το MapActivity υπογραμμίζεται με κόκκινη γραμμή. Σύρεται πάνω της το ποντίκι και από το παράθυρο που εμφανίζεται επιλέγεται το <<Import 'MapActivity' (com.google.android.maps)>>. Η MainActivity υπογραμμίζεται με κόκκινη γραμμή. Σύρεται πάνω της το ποντίκι και από το παράθυρο που εμφανίζεται επιλέγεται το <<Add unimplemented methods>>.
2. Δηλώνεται το αντικείμενο MapView. Το MapView υπογραμμίζεται με κόκκινη γραμμή. Σύρεται πάνω της το ποντίκι και από το παράθυρο που εμφανίζεται επιλέγεται το <<Import 'MapView' (com.google.android.maps)>>.

MainActivity.java:

```
package com.example.myfirstmapapp;
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapView;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
public class MainActivity extends MapActivity {

        @Override
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);
                MapView map = (MapView)findViewById(R.id.mvMap);
                map.setBuiltInZoomControls(true);
        }

        @Override
        public boolean onCreateOptionsMenu(Menu menu) {
                // Inflate the menu; this adds items to the action
bar if it is present.
                getMenuInflater().inflate(R.menu.main, menu);
                return true;
        }
```

```
        @Override
        protected boolean isRouteDisplayed() {
                // TODO Auto-generated method stub
                return false;
        }
}
```

Β.3. Εκτέλεση της Εφαρμογής

Σε πραγματική συσκευή Android:

1. Συνδέεται η συσκευή στον υπολογιστή.
2. Από τις Ρυθμίσεις της συσκευής ενεργοποιείται το USB debugging.
    - Για εκδόσεις android 3.2 και παλαιότερες Settings > Applications > Development.
    - Για εκδόσεις android 4.0 και νεότερες Settings > Developer options.
3. Εντοπίζεται το project, στον Package Explorer του Eclipse. Γίνεται δεξί-κλικ πάνω του και επιλέγεται Run As > Android Application.

Αν η έκδοση android της συσκευής είναι μικρότερη από αυτήν που έχει δηλωθεί ως minSdkVersion στο Manifest της εφαρμογής, η εφαρμογή δεν μπορεί να εκτελεστεί σε αυτήν την συσκευή.

Σε Προσομοιωτή (Emulator):

1. Πραγματοποιείται άνοιγμα του Android Virtual Device Manager από το Eclipse και επιλέγεται New.
2. Συμπληρώνεται η φόρμα με τα επιθυμητά χαρακτηριστικά. Διασφαλίζεται ότι στο Target επιλέγεται Google APIs με API level μεγαλύτερο από αυτόν που έχει δηλωθεί ως minSdkVersion στο Manifest. Επιλέγεται κλικ Create AVD.
3. Επιλέγονται ο προσομοιωτής που δημιουργήθηκε και το Start.
4. Εντοπίζεται το project, στον Package Explorer του Eclipse και πραγματοποιείται δεξί-κλικ πάνω του. Επιλέγεται το Run As > Android Application.

ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΣΕΡΡΩΝ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΕΠΙΚΟΙΝΩΝΙΩΝ

ΣΧΕΔΙΑΣΗ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΕΜΠΟΡΙΚΟΥ PORTAL ΓΙΑ ΤΗΝ ΠΟΛΗ ΤΩΝ ΣΕΡΡΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Κουστούδα Αναστασία

Στην παρούσα πτυχιακή εργασία αναλύθηκαν και περιγράφονται τεχνικές ανάπτυξης εφαρμογών για έξυπνες κινητές συσκευές με λειτουργικό σύστημα Android. Οι τεχνικές αυτές μελετήθηκαν στην πράξη με την ανάπτυξη εφαρμογής στην πλατφόρμα Android SDK. Η εφαρμογή που παρουσιάζεται έχει ως σκοπό τη συγκέντρωση και παροχή πληροφοριών για τα φαρμακεία και τα πρατήρια καυσίμων στο Νομό Σερρών, μια δυνατότητα που δεν υπήρχε διαθέσιμη για έξυπνες κινητές συσκευές, τουλάχιστον μέχρι το διάστημα συγγραφής της παρούσας εργασίας. Η ανάκτηση των σχετικών πληροφοριών γίνεται από επίσημες ιστοσελίδες στο διαδίκτυο και η παρουσίασή τους γίνεται πάνω σε διαδραστικό χάρτη, όπου παρέχονται δυνατότητες γεωκωδικοποίησης, αντίστροφης γεωκωδικοποίησης και εύρεσης διαδρομής.

DESIGN AND IMPLEMENTATION OF A TRADING PORTAL FOR THE CITY OF SERRES, GREECE

THESIS

Koustouda Anastasia

In this thesis techniques for developing applications for smart mobile devices running on Android OS were analyzed and described. These techniques have been practically studied with the development of an application using the platform Android SDK. The application aims to collect information on the pharmacies and gas stations within the Municipality of Serres, Greece from relevant pages on authoritative sites on the internet and present them on an interactive map with geocoding, inverse geocoding and navigation capabilities.