



ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

**Οδηγός της πόλης του Βόλου για κινητές
συσκευές**

A guide for the town of Volos for mobile devices

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Συμεών Ιορδανίδης

Επιβλέποντες Καθηγητές:

Μιχαήλ Βασιλακόπουλος

Αναπληρωτής Καθηγητής

Εμμανουήλ Βάβαλης

Καθηγητής

Βόλος, Σεπτέμβριος 2015



ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

**Οδηγός της πόλης του Βόλου για κινητές
συσκευές**

A guide for the town of Volos for mobile devices

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Συμεών Ιορδανίδης

Επιβλέποντες Καθηγητές:

Μιχαήλ Βασιλακόπουλος

Αναπληρωτής Καθηγητής

Εμμανουήλ Βάβαλης

Καθηγητής

Εγκρίθηκε από τη διμελή εξεταστική επιτροπή την Παρασκευή 11/9/2015

.....
Μιχαήλ Βασιλακόπουλος

Αναπληρωτής Καθηγητής

.....
Εμμανουήλ Βάβαλης

Καθηγητής

.....
Συμεών Ιορδανίδης

Διπλωματούχος Μηχανικός Ηλεκτρονικών Υπολογιστών, Τηλεπικοινωνιών και
Δικτύων του Τμήματος Ηλεκτρολόγων Μηχανικών και Μηχανικών
Υπολογιστών, Πανεπιστημίου Θεσσαλίας

Copyright © Simeon Iordanidis, 2015
Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Στην οικογένεια και στους φίλους μου

Ευχαριστίες

Θέλω να ευχαριστήσω τον επιβλέποντα καθηγητή κ.Βασιλακόπουλο για την άριστη συνεργασία μας κατά την εκπόνηση της διπλωματικής εργασίας. Οι κατευθύνσεις που μου έδωσε αποδείχτηκαν πολύτιμες . Ιδιαίτερα τον ευχαριστώ για το άμεσο ενδιαφέρον που έδειξε όποτε του ζητήθηκε.

Ένα μεγάλο ευχαριστώ στους φίλους μου για ότι περάσαμε μαζί αυτά τα χρόνια. Οι φιλίες που κέρδισα ήταν πραγματικές και ελπίζω να παραμείνουν φιλίες ζωής.Τους ευχαριστώ για την στήριξη προς το πρόσωπο μου,για τις λύπες και τις χαρές που μοιραστήκαμε.

Κυρίως θέλω να ευχαριστήσω την οικογένεια μου αλλά και όλους αυτούς τους ανθρώπους γύρω μου, που μου έμαθαν να αγωνίζομαι στην ζωή μου για το οτιδήποτε, απο το πιο μικρό ως το πιο μεγάλο. Να μην υποτάσσομαι, να αναζητώ την πραγματική ρίζα των προβλημάτων και κυρίως να αγωνίζομαι στην πράξη για έναν άλλο κόσμο, απαλλαγμένο απο το άδικο. Είναι δύσκολο να έχεις υπομονή, να μην απογοητεύεσαι, να μην λυγίζεις στην πρώτη δυσκολία. Ταυτόχρονα όμως είναι και τόσο ωραίο... Μια στιγμή χαράς είναι αυτή που σου δίνει το κουράγιο να συνεχίσεις.

Χωρίς την οικογένεια μου δεν θα έγραφα αυτές τις λέξεις σήμερα ... Είμαι σίγουρος πως δεν περιμένατε αυτά τα λίγα λόγια για να καταλάβετε πόσο σας ευχαριστώ. Θεωρώ όμως πως πρέπει να σας ευχαριστήσω για την υλική και ηθική στήριξη που μου έχετε δείξει όλα αυτά τα χρόνια που θυμάμαι τον εαυτό μου. Για το γεγονός ότι μου μάθατε να παίρνω αποφάσεις στη ζωή μου όσο δύσκολες κι αν είναι.

Περίληψη

Εκατομμύρια χρήστες καθημερινά χρησιμοποιούν smartphones. Αυτό έχει ως αποτέλεσμα την ανάγκη ανάπτυξης εφαρμογών, οι οποίες θα προσφέρουν στον χρήστη χρήσιμες για την καθημερινότητα του υπηρεσίες. Η πλειονότητα των χρηστών έχει συσκευές με Android λογισμικό. Γι' αυτό το λόγο και η παρούσα εφαρμογή αναπτύχθηκε για συσκευές Android.

Η εφαρμογή αποτελεί έναν οδηγό κίνησης για την πόλη του Βόλου. Απευθύνεται κυρίως στους πολίτες που χρησιμοποιούν ως μέσο μετακίνησης το Αστικό Λεοφορείο. Παρέχει όλες τις απαραίτητες πληροφορίες για τα δρομολόγια, τις στάσεις, τα εκδοτήρια εισιτηρίων. Δίνει στον χρήστη τη δυνατότητα αναζήτησης της κατάλληλης Γραμμής Εξυπηρέτησης.

Μελλοντικά προτείνεται η περαιτέρω επεκτασή της με επιλογές που θα προσφέρουν περαιτέρω δυνατότητες στον χρήστη, λαμβάνοντας υπόψιν για παράδειγμα κι άλλες επιθυμίες, όπως την αναζήτηση κατάλληλης διαδρομής με κριτήριο κάποιο εστιατόριο, cafe κτλ.

Η δημιουργία της έγινε με τη χρήση του ολοκληρωμένου προγραμματιστικού περιβάλλοντος της Google, Android Studio. Λειτουργεί σε εκδόσεις 4.2 και άνω, έχοντας δοκιμαστεί σε πραγματικές συσκευές.

Abstract

Millions of users daily use smartphones. This results in the need to develop applications that will provide the user with useful everyday services. The majority of users have Android-powered devices. For this reason, this application developed for Android devices.

Implementation is a motion guide for the city of Volos. It is mainly addressed to people who use it as a means of getting around the Urban bus for. It provides all the necessary information on timetables, stops, ticket eisitirion.Dinei the user to find the appropriate Service Line.

Future proposed further expansion with options that offer a potential user, taking into account for instance and other desires, as seeking appropriate route criterion for a restaurant, cafe etc.

The creation was done using the integrated programming environment of Google, Android Studio. It works in versions 4.2 and above, having been tested on real devices.

Περιεχόμενα

Κεφάλαιο 1	15
Εισαγωγή	15
1.1 Η αναγκαιότητα ανάπτυξης της εφαρμογής.....	15
1.2 Παρόμοιες εφαρμογές.....	16
1.3 Διάρθρωση της Διπλωματικής Εργασίας.....	16
Κεφάλαιο 2	17
Λειτουργικό Σύστημα Android.....	17
2.1 Τι είναι το Android	17
2.2 Η εξέλιξη του Android.....	18
2.2.1 Android 1.0 (API level 1)	18
2.2.2 Android 1.1 (API level 2)	18
2.2.3 Android 1.5 Cupcake(API level 3).....	18
2.2.4 Android 1.6 Donut(API level 4)	18
2.2.5 Android 2.0 Éclair (API level 5).....	18
2.2.6 Android 2.2 Froyo (API level 8)	19
2.2.7 Android 2.3 Gingerbread (API level 9).....	19
2.2.8 Android 3.0 Honeycomb (API level 11).....	19
2.2.9 Android 4.0 Ice Cream Sandwich(API level 14).....	19
2.2.10 Android 4.1 Jelly Bean (API level 16)	19
2.2.11 Android 4.4 KitKat(API level 19).....	19
2.2.12 Android 5.0 Lollipop(API level 21).....	20
2.3 Εφαρμογές Android	20
2.4 Η γλώσσα προγραμματισμού Java	21
2.5 Η γλώσσα σήμανσης XML	22
2.6 Πού αναπτύσσουμε μια εφαρμογή Android - Τι υπάρχει στο εσωτερικό της.....	22
2.6.1 Προγραμματιστικά Περιβάλλοντα.....	22
2.6.2 Τα βασικότερα συστατικά μιας εφαρμογής	24
Κεφάλαιο 3	27
Επίδειξη της εφαρμογής	27
3.1 Γενικά	27
3.2 Σενάρια Εκτέλεσης.....	28

Κεφάλαιο 4	43
Υλοποίηση της Εφαρμογής.....	43
4.1 Γενική Δομή.....	43
4.2 Αναλυτική Παρουσίαση.....	43
Κεφάλαιο 5	69
Συμπεράσματα.....	69
5.1 Αποτελέσματα.....	69
5.2 Δυσκολίες.....	69
5.3 Μελλοντικές επεκτάσεις.....	70
Αναφορές.....	73
Βιβλιογραφία.....	75
Βιβλία	75
Tutorials.....	75
ΠΑΡΑΡΤΗΜΑ.....	77

Κεφάλαιο 1

Εισαγωγή

1.1 Η αναγκαιότητα ανάπτυξης της εφαρμογής

Η εφαρμογή απευθύνεται στους πολίτες που κινούνται στην πόλη του Βόλου. Σε μια τέτοια πόλη η εξυπηρέτηση με τα Αστικά Λεοφορεία είναι μέρος της καθημερινότητας. Άνθρωποι κάθε ηλικίας χρησιμοποιούν αυτό το μέσο για την μετακίνηση τους κατά τη διάρκεια της ημέρας, για να πάνε στον χώρο εργασίας τους, στο Πανεπιστήμιο, στο σπίτι τους. Είναι χαρακτηριστικό ότι υπάρχουν 12 λεωφορειακές γραμμές, που καλύπτουν όλο το Πολεοδομικό συγκρότημα του Βόλου και της Νέας Ιωνίας καθώς και γειτονικές περιοχές [1]. Για τον παραπάνω λόγο επιλέχθηκε στην εφαρμογή VolosGuide να δοθεί βαρύτητα σε αυτό το μέσο μεταφοράς.

Ένα συχνό φαινόμενο που παρατηρείται είναι η άγνοια για την γραμμή που πρέπει να επιβιβαστεί κάποιος με βάση την διαδρομή που θέλει να ακολουθήσει όπως επίσης και η έλλειψη γνώσης ανα πάσα στιγμή των δρομολογίων. Αποτέλεσμα να καταφεύγει σε λύσεις που μπορεί να αποδειχτούν λαθεμένες ή χρονοβόρες. Οι ερωτήσεις στους γύρω μας ενέχουν συχνά τον κίνδυνο μιας λαθεμένης απάντησης. Η αναζήτηση μέσω κάποιου browser ειδικά σε μια κινητή συσκευή, δεν αποτελεί πάντα μια γρήγορη και το ίδιο απλή για όλους λύση.

Η έλλειψη μιας απλής στην χρήση εφαρμογής που θα την εγκαθιστά κάποιος στην συσκευή του και θα μπορεί κάθε στιγμή μέσω απλών βημάτων να παίρνει όλες τις απαραίτητες παραπάνω πληροφορίες, οδήγησε στην ανάπτυξη αυτής της εφαρμογής.

1.2 Παρόμοιες εφαρμογές

Υπάρχει ήδη μια πληθώρα εφαρμογών σχετικές με τα Αστικά Λεοφορεία για διάφορες πόλεις της Ελλάδας. Άλλες προσφέρουν μόνο βασικές δυνατότητες όπως η προβολή δρομολογίων-στάσεων ενώ άλλες περισσότερες, όπως ενημέρωση σε πραγματικό χρόνο για τις αφίξεις δρομολογίων όπου αυτό είναι εφικτό.

Αναζητώντας κάποιος στο Google Play Store αυτές τις εφαρμογές μπορεί να αποκτήσει μια σφαιρική εικόνα. Χαρακτηριστικό παράδειγμα αποτελεί η εφαρμογή του ΟΑΣΘ [2].

1.3 Διάρθρωση της Διπλωματικής Εργασίας

Το Κεφάλαιο 1 αποτέλεσε μια μικρή εισαγωγή για το αντικείμενο της Διπλωματικής Εργασίας.

Στο Κεφάλαιο 2 γίνεται εκτενής αναφορά στις τεχνολογίες που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής. Συγκεκριμένα περιγράφεται το Λειτουργικό Σύστημα Android καθ' όλη τη διάρκεια εξέλιξης του, όπως επίσης και ο εξομοιωτής που βοήθη στην δοκιμή της εφαρμογής. Παράλληλα γίνεται εκτενής αναφορά στο περιβάλλον προγραμματισμού Android Studio και στις γλώσσες προγραμματισμού Java , xml, php .

Στο Κεφάλαιο 3 γίνεται στον αναγνώστη πλήρης επίδειξη της εφαρμογής μέσα από διάφορα screenshots κατά την εκτέλεση στον Emulator.

Το Κεφάλαιο 4 παρουσιάζει την υλοποίηση της εφαρμογής. Δεν παρουσιάζεται αυτούσιος όλος ο κώδικας αλλά ενδεικτικά κομμάτια από τις βασικές κλάσεις που αναπτύχθηκαν. Ταυτόχρονα περιγράφεται η βάση δεδομένων που δημιουργήθηκε καθώς και οι πίνακες που την αποτελούν.

Τέλος το Κεφάλαιο 5 αξιολογεί την εφαρμογή και παρουσιάζει ιδέες για την βελτίωση και επέκταση της εφαρμογής στο μέλλον.

Λειτουργικό Σύστημα Android

2.1 Τι είναι το Android

Το Android [3] είναι λειτουργικό σύστημα για συσκευές κινητής τηλεφωνίας το οποίο βασίζεται στον πυρήνα του λειτουργικού Linux. Αρχικά αναπτύχθηκε από την Google και αργότερα από την Handset Alliance/Open Handset Alliance. Είναι λογισμικό ανοιχτού κώδικα, πράγμα το οποίο σημαίνει ότι οποιοσδήποτε μπορεί να πάρει τον πηγαίο κώδικα και να τον χρησιμοποιήσει κατά το δοκούν. Επιτρέπει στους κατασκευαστές λογισμικού να συνθέτουν κώδικα με τη χρήση της γλώσσας προγραμματισμού Java, ελέγχοντας τη συσκευή μέσω βιβλιοθηκών λογισμικού ανεπτυγμένων από την Google. Το Android είναι κατά κύριο λόγο σχεδιασμένο για συσκευές με οθόνη αφής, όπως τα έξυπνα τηλέφωνα (smartphones) και τα τάμπλετ (tablets) με διαφορετικό περιβάλλον χρήσης για τηλεοράσεις (Android TV), αυτοκίνητα (Android Auto) και ρολόγια χειρός (Android Wear). Παρόλο που έχει αναπτυχθεί για συσκευές με οθόνη αφής, έχει χρησιμοποιηθεί σε κονσόλες παιχνιδιών, ψηφιακές φωτογραφικές μηχανές, συνηθισμένους Η/Υ (π.χ. το HP Slate 21) και σε άλλες ηλεκτρονικές συσκευές.

Το Android στις μέρες μας είναι το πιο ευρέως διαδεδομένο λογισμικό στον κόσμο. Οι συσκευές με Android έχουν περισσότερες πωλήσεις από όλες τις συσκευές Windows, iOS και Mac OS X μαζί. Η πρώτη παρουσίαση της πλατφόρμας Android έγινε στις 5 Νοεμβρίου 2007, παράλληλα με την ανακοίνωση της ίδρυσης του οργανισμού Open Handset Alliance, μιας κοινοπραξίας 48 τηλεπικοινωνιακών εταιριών, εταιριών λογισμικού καθώς και κατασκευής hardware, οι οποίες είναι αφιερωμένες στην ανάπτυξη και εξέλιξη ανοιχτών προτύπων στις συσκευές κινητής τηλεφωνίας.



Εικόνα 2.1 Λογότυπο Android

2.2 Η εξέλιξη του Android

Η εξέλιξη του Android είναι ραγδαία, κάτι που οφείλεται σε μεγάλο βαθμό στο ότι είναι λογισμικό ανοιχτού κώδικα. Είναι χαρακτηριστικό ότι από το 2009 έως και το 2015 έχουν κυκλοφορήσει δέκα κύριες εκδόσεις.

Όπως όλα τα λογισμικά, έτσι και το Android, όλα αυτά τα χρόνια αναπτύσσει νέες εκδόσεις, προσθέτοντας νέα χαρακτηριστικά και βελτιώσεις. Η Google, χρησιμοποιεί αλφαβητική σειρά στις ονομασίες των εκδόσεων του Android οι οποίες έχουν πάντα σαν θέμα κάποιο γλύκισμα, με εξαίρεση τις δύο πρώτες.

Συγκεντρωτικά λοιπόν υπάρχουν οι εξής εκδόσεις:

2.2.1 Android 1.0 (API level 1)

Είναι η πρώτη εμπορική έκδοση του λογισμικού. Κυκλοφόρησε στις 23 Σεπτεμβρίου 2008. Η πρώτη εμπορικά διαθέσιμη Android συσκευή ήταν το HTC Dream (Android 1.0).

2.2.2 Android 1.1 (API level 2)

Στις 9 Φεβρουαρίου 2009, κυκλοφόρησε η ενημερωμένη έκδοση του Android 1.0. Η Android 1.1, γνωστή ως "Petit Four", αν και αυτό το όνομα δεν χρησιμοποιείται επίσημα, ήταν διαθέσιμη αρχικά μόνο για το HTC Dream.

2.2.3 Android 1.5 Cupcake(API level 3)

Το χαρακτηριστικό που ανέδειξε η έκδοση Android 1.5 Cupcake είναι η εισαγωγή ψηφιακού πληκτρολογίου. Φαίνεται περίεργο, αλλά το 2008/2009, τα περισσότερα smartphones είχαν φυσικό πληκτρολόγιο QWERTY.

2.2.4 Android 1.6 Donut(API level 4)

Το Android 1.6 Donut είχε αλλαγές στο εσωτερικό του λειτουργικού και αυτό είχε ως αποτέλεσμα να υποτιμηθεί αρκετά από τους καταναλωτές. Παρόλα αυτά, ήταν πολύ σημαντική αναβάθμιση, καθώς έφερε υποστήριξη διαφορετικών αναλύσεων οθόνης, ανεξάρτητα από την πυκνότητα pixel, κάτι που καθόρισε πραγματικά το μέλλον των Android συσκευών.

2.2.5 Android 2.0 Éclair (API level 5)

Αν πρέπει να επιλέξουμε ένα από τα χαρακτηριστικά του Eclair, αυτό είναι οι βελτιώσεις στην κάμερα. Μέχρι τότε το Android δεν είχε υποστήριξη LED flash στην κάμερα, ούτε δυνατότητες που σήμερα θεωρούμε δεδομένες, όπως χρωματικά εφέ, λειτουργία σκηνών, λειτουργίες εστίασης, ή ακόμη και ρύθμιση της ισορροπίας λευκού.

2.2.6 Android 2.2 Froyo (API level 8)

Το Froyo ανέδειξε δύο χαρακτηριστικά και αυτά είναι η ταχύτητα και η υποστήριξη για Wi-Fi hotspots. Ως τότε το Wi-Fi Hotspot δεν ήταν διαθέσιμο παγκοσμίως. Στο Froyo ορίστηκε σαν runtime ο Dalvik, που έφερε σοβαρές βελτιώσεις στην απόδοση της συσκευής, που σύμφωνα με την Google γίνεται από 2 έως 5 φορές ταχύτερη.

2.2.7 Android 2.3 Gingerbread (API level 9)

Το Android 2.3 Gingerbread, σίγουρα ήταν μία από τις πιο διάσημες εκδόσεις του λειτουργικού και με την προσθήκη υποστήριξης για περισσότερους αισθητήρες, όπως το βαρόμετρο και γυροσκόπιο, βοήθησε πραγματικά στην εξέλιξη του Android. Πριν το Gingerbread, με την έλλειψη πολλών αισθητήρων, υπήρχε σοβαρός περιορισμός στο τι μπορούμε να κάνουμε με το κινητό μας και ειδικότερα στα παιχνίδια.

2.2.8 Android 3.0 Honeycomb (API level 11)

Το Honeycomb, ήταν μια από τις λιγότερο δημοφιλείς εκδόσεις του Android, αλλά αυτό σίγουρα δεν έχει να κάνει με τη συνεισφορά του στην πλατφόρμα. Το Android, είχε κακή εμπειρία χρήσεως στα tablets, κάτι που άλλαξε με την κυκλοφορία του Honeycomb, που ήταν σχεδιασμένο για συσκευές με μεγαλύτερη οθόνη.

2.2.9 Android 4.0 Ice Cream Sandwich (API level 14)

Το Ice Cream Sandwich, έφερε πραγματική ανανέωση στο σχεδιασμό του Android με το Holo UI. Το ICS, ήταν η πρώτη έκδοση του Android που έδωσε μοντέρνα χαρακτηριστικά στο λειτουργικό. Ακόμη και σήμερα συναντάμε πολλά στοιχεία του στις Android συσκευές.

2.2.10 Android 4.1 Jelly Bean (API level 16)

Το Jelly Bean είναι μια σημαντική έκδοση για πολλούς λόγους, ιδιαίτερα αν σκεφτούμε ότι κατέχει περίπου το 50% των Android συσκευών. Οι συνεισφορές του στο Android είναι πολλές, αλλά αυτή που ξεχώρισε και αναπτύσσεται μέχρι και σήμερα, είναι το Google Now. Ο ψηφιακός βοηθός της Google, όχι μόνο παρέχει γρήγορες απαντήσεις για τα πάντα, αλλά και εμφανίζει νέα και ειδήσεις σύμφωνα με τα ενδιαφέροντα του χρήστη. Το Google Now έχει οργανωμένες κάρτες, που μας δίνουν πληροφορίες σχετικές με την ημέρα, τον καιρό αλλά και την περιοχή που βρισκόμαστε.

2.2.11 Android 4.4 KitKat (API level 19)

Με το KitKat, η Google όχι απλά ενίσχυσε το UI, αλλά και ανέδειξε τη δύναμη του λειτουργικού ακόμη και σε φθηνές συσκευές. Χάρη στις πολλές βελτιώσεις επιδόσεων που έγιναν στο KitKat, η Google υποστήριξε πως ακόμη και συσκευές με μόλις 512MB RAM θα είναι σε θέση να τρέξουν το KitKat χωρίς

προβλήματα. Αυτό ήταν ιδιαίτερα σημαντικό, δεδομένου ότι οι εταιρείες έδειχναν αδιαφορία στις οικονομικές συσκευές. Έτσι, δόθηκε η δυνατότητα και σε αναπτυσσόμενες χώρες να αποκτήσουν Android συσκευές και να αυξηθούν οι πωλήσεις φθηνότερων συσκευών.

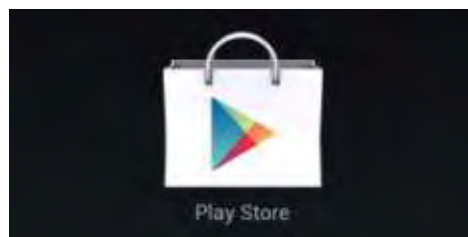
2.2.12 Android 5.0 Lollipop(API level 21)

Σήμερα, το Lollipop είναι η πιο ελκυστική έκδοση του Android. Με το Material Design της Google, η εμφάνιση του λειτουργικού έγινε επίπεδη, με περισσότερα χρώματα και με πολλά εφέ κίνησης.

Ενδιάμεσα υπάρχουν βελτιωμένες εκδόσεις, επεκτάσεις των υπάρχοντων.

2.3 Εφαρμογές Android

Μεγάλος αριθμός προγραμματιστών στις μέρες μας αναπτύσσουν εφαρμογές για Android, οι οποίες επεκτείνουν την λειτουργικότητα των συσκευών. Υπάρχουν εκατοντάδες εφαρμογές που προσφέρουν πολλές δυνατότητες στους χρήστες στην καθημερινότητά τους. Σημαντικό κομμάτι επίσης αποτελεί η ανάπτυξη παιχνιδιών. Όπως προαναφέρθηκε για να αναπτύξει κάποιος μια εφαρμογή χρειάζεται να έχει γνώσεις Java και xml . Ο χρήστης μπορεί να κατεβάσει μια εφαρμογή στη συσκευή του από το online κατάστημα της Google, το Google Play Store.



Εικόνα 2.2 Λογότυπο Google Play Store

2.4 Η γλώσσα προγραμματισμού Java

Η Java [4] έχει τα εξής βασικά χαρακτηριστικά. Είναι απλή, αντικειμενοστραφής, συμβατή με δικτυακά πρωτόκολλα, φορητή και ασφαλής, έχει υψηλή απόδοση.

Αρχικός στόχος της ομάδας της Sun που ανέπτυξε την Java, ήταν να δημιουργηθεί μια γλώσσα εύκολη στην χρήση, ώστε να μην απαιτεί πολλή εξάσκηση και εκπαίδευση. Όπως αποδείχτηκε τα κατάφερε καθώς η Java είναι μια από τις πιο διαδεδομένες γλώσσες στις μέρες μας.

Ως αντικειμενοστραφής, ο σχεδιασμός ενός προγράμματος Java βασίζεται σε αντικείμενα. Κάθε αντικείμενο αντιμετωπίζεται σαν ένα "μαύρο κουτί". Τα αντικείμενα δεν είναι ανεξάρτητα μεταξύ τους. Βρίσκονται σε σχέση αλληλεξάρτησης με τα υπόλοιπα. Μεταξύ των αντικειμένων υπάρχει η έννοια της κληρονομικότητας, δηλαδή ένα αντικείμενο μπορεί να κληρονομήσει δεδομένα από άλλα. Η Java χρησιμοποιείται σε ανοικτά, δικτυωμένα περιβάλλοντα. Ιδιαίτερη προσοχή έχει δοθεί στην ασφάλεια που παρέχει η γλώσσα. Είναι δυνατή η κατασκευή προγραμμάτων ελεύθερων από ιούς. Το γεγονός ότι είναι ανεξάρτητη της υποκείμενης πλατφόρμας και του λειτουργικού συστήματος αποτελεί μεγάλο πλεονέκτημα. Προγράμματα γραμμένα σε Java μπορούν να τρέχουν ακριβώς το ίδιο σε Windows, Linux, Unix, Macintosh χωρίς να χρειαστεί να ξαναγίνει μεταγλώττιση ή να αλλάξει ο πηγαίος κώδικας για κάθε διαφορετικό λειτουργικό σύστημα.

Στην Java υπάρχουν δυο είδη προγραμμάτων : τα *applets* και οι *applications*.

Τα *Applets* είναι μικρά κομμάτια εκτελέσιμου κώδικα που απαιτούν Web browser για να τρέξουν. Υποδηλώνεται στην HTML σελίδα με ένα `<applet>` tag. Όταν ο Web browser ενός χρήστη επεξεργαστεί μία ηλεκτρονική σελίδα που περιέχει ένα τέτοιο tag, αυτόματα κατεβάζει το applet και το τρέχει στην μηχανή του client.

Τα *Applications* είναι προγράμματα που δεν απαιτούν Web browser για να τρέξουν και δεν διαθέτουν ενσωματωμένο μηχανισμό για μεταφορά στο δίκτυο. Βρίσκονται τοπικά αποθηκευμένες. Όπως και τα applets, οι applications απαιτούν μία Java Virtual Machine για να τρέξουν.

Για να γράψει κάποιος κώδικα Java δε χρειάζεται τίποτα άλλο παρά έναν επεξεργαστή κειμένου, όπως το Σημειωματάριο (Notepad) των Windows ή ο vi (γνωστός στο χώρο του Unix). Παρ'όλ'αυτά, ένα ολοκληρωμένο περιβάλλον ανάπτυξης (*IDE*) βοηθάει πολύ, ιδιαίτερα στον εντοπισμό σφαλμάτων (*debugging*). Υπάρχουν αρκετά διαθέσιμα, ενώ πολλά από αυτά έρχονται δωρεάν.

2.5 Η γλώσσα σήμανσης XML

Η XML [5] είναι μια επεκτάσιμη γλώσσα σήμανσης που αναπτύχθηκε για να διατηρήσει την ευελιξία και την δύναμη της HTML. Ταυτόχρονα βασικό χαρακτηριστικό της ήταν η μείωση του μεγαλύτερου μέρους της πολυπλοκότητας.

Η XML σχεδιάστηκε να ικανοποιήσει πολλές ανάγκες δίνοντας στα έγγραφα ένα μεγαλύτερο επίπεδο προσαρμοστικότητας στο στυλ και τη δομή από αυτό που υπήρχε παλαιότερα στην HTML. Χρησιμοποιείται για την απλή και εύχρηστη αποθήκευση και διανομή δεδομένων. Οι εφαρμογές της αποτελούν περισσότερο βάσεις δεδομένων παρά σελίδες περιεχομένου. Τέλος η XML προσφέρει στους σχεδιαστές της HTML τη δυνατότητα να προσθέτουν περισσότερα στοιχεία στη γλώσσα. Δεν αναφέρεται μονάχα στους σχεδιαστές του web αλλά σε οποιονδήποτε ασχολείται με εκδόσεις.

2.6 Πού αναπτύσσουμε μια εφαρμογή Android - Τι υπάρχει στο εσωτερικό της.

2.6.1 Προγραμματιστικά Περιβάλλοντα

Τα δύο βασικότερα προγραμματιστικά περιβάλλοντα όπου μπορεί κάποιος να υλοποιήσει μια εφαρμογή είναι (α) το Eclipse όπου υπάρχει η δυνατότητα ενσωμάτωσης του Android SDK και (β) το επίσημο εργαλείο της Google, Android Studio.

Αυτά τα εργαλεία μπορεί εύκολα να τα κατεβάσει από τα επίσημα sites και ακολουθώντας τις κατάλληλες οδηγίες να τα εγκαταστήσει στον υπολογιστή του. Στην παρούσα εργασία θα δοθεί βαρύτητα στο Android Studio καθώς είναι το βασικό εργαλείο που χρησιμοποιήθηκε.

Eclipse

Το Eclipse [6] είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE). Το Eclipse είναι γραμμένο κυρίως σε Java και η κύρια χρήση του είναι για την ανάπτυξη εφαρμογών Java, αλλά μπορεί επίσης να χρησιμοποιηθεί για την ανάπτυξη εφαρμογών σε άλλες γλώσσες προγραμματισμού με τη χρήση των προσθέτων, που περιλαμβάνει: Ada, ABAP, C, C ++, COBOL, Fortran, Haskell, JavaScript, λάσο, Lua, Φυσικό, Perl, PHP, Prolog, Python, R, Ruby.

Απο την Google διατίθεται ένα πρόσθετο λογισμικού για το Eclipse IDE. Προσαρμόζοντας το εκεί ο προγραμματιστής έχει την δυνατότητα να αναπτύσσει εφαρμογές Android.



Εικόνα 2.3 Λογότυπο Eclipse

Android Studio

Το **Android Studio** [7] είναι ένα ολοκληρωμένο προγραμματιστικό περιβάλλον (IDE) για ανάπτυξη εφαρμογών στην Android πλατφόρμα. Ανακοινώθηκε στις 16 Μαΐου 2013 στο συνέδριο Google I/O από την Google Product Manager, Katherine Chou. Το Android Studio είναι διαθέσιμο ελεύθερα με την άδεια Apache License 2.0.

Το Android Studio ήταν διαθέσιμο σε πρώιμο στάδιο για προεπισκόπηση ξεκινώντας από την έκδοση 0.1 τον Μάιο του 2013, έπειτα ξεκίνησε το δοκιμαστικό στάδιο από την έκδοση 0.8 που βγήκε τον Ιούνιο του 2014. Η πρώτη σταθερή έκδοση βγήκε το Δεκέμβριο του 2014, με την έκδοση 1.0.

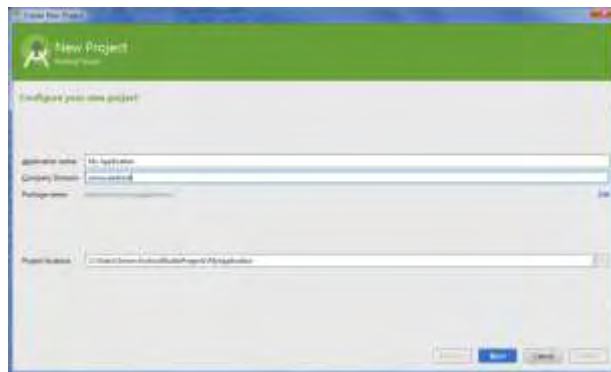


Εικόνα 2.4 Λογότυπο Android Studio

2.6.2 Τα βασικότερα συστατικά μιας εφαρμογής

Κάθε project αποτελείται από πολλά αρχεία και φακέλους, τα οποία συνδυάζονται και αφού γίνουν compile, δίνουν το τελικό αρχείο .apk το οποίο εγκαθίσταται στις συσκευές.

Ξεκινώντας τη δημιουργία ενός καινούριου project, του δίνουμε το όνομα το οποίο επιθυμούμε και ένα μοναδικό όνομα πακέτου το οποίο χρησιμοποιείται από το λειτουργικό σύστημα.



Εικόνα 2.5 New project στο Android Studio

Τα βασικότερα συστατικά τα οποία σίγουρα θα χρειαστεί κάποιος προγραμματιστής να διαχειριστεί είναι τα εξής:

A) Activities

B) Φάκελος res

Γ) AndroidManifest.xml

Δ) Η εικονική μηχανή(emulator)

Activities

Είναι το βασικότερο στοιχείο. Πρόκειται για Java classes στις οποίες υλοποιούμε τη λειτουργικότητα της εφαρμογής. Προϋπάρχουν έτοιμα activities ανάλογα με το τι θέλουμε να κάνουμε (Blank Activity, Blank Activity with Fragment, Settings Activity, Google Maps Activity κλπ). Στην εφαρμογή υπάρχουν τόσες activities όσες και οι διαφορετικές «τελικές οθόνες» που θέλουμε να εμφανίζονται στον χρήστη. Μπορούν μεταξύ τους να επικοινωνούν μέσω των Intents, όπως θα δούμε παρακάτω.

Φάκελος res

Εδώ συναντάμε τρεις σημαντικούς υποφακέλους (drawable, layout, values).

Στο φάκελο *drawable* υπάρχει η δυνατότητα αποθήκευσης σε διάφορες ποιότητες των εικόνων που θα χρησιμοποιήσουμε στην εφαρμογή.

Στον φάκελο *layouts* συναντάμε το «άλλο μισό» των activities, τα αρχεία .xml. Τα αρχεία αυτά είναι υπεύθυνα για αυτό που βλέπει ο χρήστης. Για κάθε διαφορετική «οθόνη» υπάρχει ένα διαφορετικό .xml. Το Android Studio δίνει τη δυνατότητα στον προγραμματιστή να τοποθετεί στα layouts μια πληθώρα έτοιμων στοιχείων (Buttons, Textview, ListViews, ScrollViews κλπ) και να τα επεξεργάζεται όπως αυτός θέλει.

Στο φάκελο *values* το κυριότερο αρχείο είναι το strings.xml στο οποίο μπορούμε να δίνουμε ονομασίες στα strings που έχουμε στο project και να τα χρησιμοποιούμε με βάση αυτές.

AndroidManifest.xml

Σε αυτό το αρχείο δηλώνονται όλες οι σημαντικές πληροφορίες της εφαρμογής. Όποιο activity δημιουργούμε πρέπει να καταχωρηθεί εδώ. Συνήθως, περιβάλλοντα όπως το Android Studio εξασφαλίζουν την καταχώρηση όλων των απαραίτητων πληροφοριών στο manifest, με τη δημιουργία της Activity και τη διαμόρφωση του layout της, χωρίς ο

προγραμματιστής να πρέπει εξ αρχής να γράψει όλο τον κώδικα εδώ. Στο AndroidManifest δηλώνονται ακόμη το όνομα του πακέτου της εφαρμογής και όλες οι απαραίτητες «άδειες» που πρέπει να έχει η εφαρμογή για να πραγματοποιήσει κάποιες λειτουργίες της όπως για παράδειγμα η ανάγκη σύνδεσης στο Internet. Χωρίς αυτό το αρχείο δεν μπορεί κάποιος να προχωρήσει επιτυχώς.

Emulator

Βασικό πλεονέκτημα του Android Studio είναι ότι προσφέρει τον δικό του εξομοιωτή. Μέσω αυτού, ο προγραμματιστής μπορεί να σχεδιάζει και να ελέγχει την εφαρμογή που αναπτύσσει στον προσωπικό του υπολογιστή. Ο εξομοιωτής λειτουργεί ακριβώς με τον ίδιο τρόπο όπως μια κινητή συσκευή. Η μόνη διαφορά είναι ότι δεν υπάρχει η δυνατότητα τηλεφωνικής κλήσης. Κατα τ'άλλα μπορεί να χρησιμοποιήσει όλες τις υπόλοιπες υπηρεσίες. Μπορεί να εγκαταστήσει την εφαρμογή του και να την δοκιμάσει.

Μέσω της εικονικής AVD Manager ο χρήστης μπορεί να ορίζει τα δικά του επιθυμητά χαρακτηριστικά στον Emulator. Μπορεί να επιλέξει το μοντέλο της συσκευής που θέλει, την έκδοση Android, το μέγεθος της μνήμης RAM, την χωρητικότητα της SD card και πολλά άλλα.



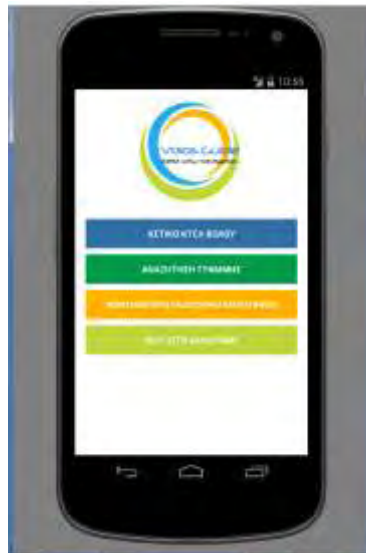
Εικόνα 2.6 Emulator

Κεφάλαιο 3

Επίδειξη της εφαρμογής

3.1 Γενικά

Όπως έχει προαναφερθεί, σκοπός της εφαρμογής είναι να διευκολύνει τον χρήστη δίνοντας του οδηγίες για τη μετακίνηση του μέσα στην πόλη του Βόλου, κυρίως με το Αστικό Λεοφορείο. Οι υπηρεσίες που του προσφέρονται είναι **(1)** απαραίτητες πληροφορίες για το Αστικό Κτελ (Δρομολόγια, Στάσεις κλπ), **(2)** η δυνατότητα να προτείνεται η κατάλληλη Γραμμή Εξυπηρέτησης με βάση τα επιθυμητά σημεία αφετηρίας και τερματισμού, **(3)** η αναζήτηση του κοντινότερου εκδοτηρίου εισητηρίων μετά από την επιλογή ενός σημείου στον χάρτη από τον χρήστη, **(4)** η δυνατότητα εμφάνισης της βέλτιστης διαδρομής με τα πόδια μεταξύ συγκεκριμένων «σημείων ενδιαφέροντος. Όταν ο χρήστης ανοίγει την εφαρμογή του παρουσιάζεται το βασικό μενού



Εικόνα 3.1 App Main Menu

3.2 Σενάρια Εκτέλεσης

Έστω ότι επιλέγουμε την πρώτη κατηγορία **«ΑΣΤΙΚΟ ΚΤΕΛ ΒΟΛΟΥ»**. Εδώ θα βρει κάποιος τα δρομολόγια για όλες τις γραμμές, τις στάσεις της κάθε γραμμής, όπως επίσης και την προβολή της διαδρομής που εκτελεί μια γραμμή στο χάρτη.

Έτσι, μπαίνοντας σε αυτήν την κατηγορία, εμφανίζεται το παρακάτω menu.



Εικόνα 3.2 Menu κατηγορίας «ΑΣΤΙΚΟ ΚΤΕΛ ΒΟΛΟΥ»

Πατώντας **«Δρομολόγια Γραμμών»**, εμφανίζονται αρχικά όλες οι πιθανές Γραμμές σε μια λίστα, ώστε ο χρήστης να επιλέξει αυτή που θέλει.



Εικόνα 3.3 Menu υποκατηγορίας «Δρομολόγια Γραμμών»

Έστω ότι επιλέγεται η πρώτη (Γραμμή No1 Άναυρος-Ν.Ιωνία). Στη συνέχεια ο χρήστης πρέπει να επιλέξει αν ενδιαφέρεται για το δρομολόγιο μετάβασης ή επιστροφής.



Εικόνα 3.4 Επιλογή Μετάβασης-Επιστροφής Γραμμής Νο1

Υπάρχουν ακόμα τρία κουμπιά. Κατά σειρά ο χρήστης μπορεί:

- 1) να επιστρέψει στο menu της κατηγορίας «ΑΣΤΙΚΟ ΚΤΕΛ ΒΟΛΟΥ»
- 2) να προβάλλει το δρομολόγιο στο χάρτη



Εικόνα 3.5 Χάρτης Γραμμής Νο1

- 3) να επιστρέψει στο αρχικό menu της εφαρμογής

Αφού λοιπόν επιλέξει κατεύθυνση, στη συνέχεια καλείται να επιλέξει ημέρα, καθώς τα δρομολόγια διαφέρουν.



Εικόνα 3.6 Επιλογή ημέρας δρομολογίων

Αν για παράδειγμα η επιλογή είναι «**Σάββατο**», τότε εμφανίζονται οι ώρες των δρομολογίων.



Εικόνα 3.7 Προβολή δρομολογίων

Αν ο χρήστης επιλέξει **«Γραμμές και Στάσεις»** από το menu της κατηγορίας, μπορεί να πληροφορηθεί για τις στάσεις που κάνει κάθε γραμμή τόσο για τη μετάβαση από την αφετηρία στον τερματισμό όσο και για το αντίστροφο. Αφού λοιπόν διαλέξει γραμμή (στην περίπτωση μας Γραμμή Νο1)



Εικόνα 3.8 Επιλογή Γραμμής απο υποκατηγορία «Γραμμές και Στάσεις»

στη συνέχεια επιλέγει ανάμεσα σε **«Στάσεις Μετάβασης»** και **«Στάσεις Επιστροφής»** και παίρνει το αντίστοιχο αποτέλεσμα.



Εικόνα 3.9 Επιλογή «Στάσεων Μετάβασης»-«Στάσεων Επιστροφής»



Εικόνα 3.10 Προβολή στάσεων

Με την επιλογή **«Χάρτης Γραμμών»** από το αρχικό menu της κατηγορίας, υπάρχει η δυνατότητα επιλογής της επιθυμητής Γραμμής και προβολής της διαδρομής της στο χάρτη (όπως στην εικόνα που παρουσιάστηκε παραπάνω) για κάποιον που ενδιαφέρεται να την δει απευθείας εκεί.

Τέλος με την επιλογή **«Χρήσιμες Πληροφορίες»** προσφέρονται στο χρήστη πληροφορίες σχετικά με την επικοινωνία με την εταιρία.

Επιλέγοντας ο χρήστης την κατηγορία **«ΑΝΑΖΗΤΗΣΗ ΓΡΑΜΜΗΣ»** από το αρχικό μενού έχει την δυνατότητα μέσω της επιλογής ενός σημείου αφετηρίας και ενός σημείου τερματισμού στον χάρτη να μάθει με ποια γραμμή του Αστικού Λεοφορείου μπορεί να εξυπηρετηθεί, σε ποια στάση πρέπει να επιβιβαστεί και που να αποβιβαστεί, με γνώμονα ότι η αποβίβαση πρέπει να γίνει όσο το δυνατόν κοντινότερα στο σημείο τερματισμού.



Εικόνα 3.11 Επιλογή σημείων στον Χάρτη

Ο χρήστης επιλέγει αφετηρία:



Εικόνα 3.12 Επιλογή αφετηρίας στον Χάρτη

Στη συνέχεια επιλέγει τερματισμό:



Εικόνα 3.13 Επιλογή τερματισμού στον Χάρτη

Αφού επιλέξει σημεία του δίνονται οι εξής επιλογές:



Εικόνα 3.14 Μενού μετά την επιλογή σημείων

- **ΠΛΗΡΟΦΟΡΙΕΣ:** Οδηγεί σε ένα νέο layout όπου δίνονται οι πληροφορίες όπως περιγράφηκε νωρίτερα. Με βάση τα σημεία που επιλέχθηκαν δίνεται το παρακάτω αποτέλεσμα εξυπηρέτησης



Εικόνα 3.15 Προβολή αποτελέσματος «Αναζήτησης Γραμμής»

- **MENU icon:** Οδηγεί στο αρχικό menu.
- **ΕΠΙΛΟΓΗ ΣΗΜΕΙΩΝ:** Οδηγεί ξανά στο χάρτη ώστε να επιλεγούν σημεία από την αρχή.

Υποθέτουμε τώρα ότι ο χρήστης επιλέγει από το αρχικό menu την κατηγορία «**ΚΟΝΤΙΝΟΤΕΡΟ ΕΚΔΟΤΗΡΙΟ ΕΙΣΗΤΗΡΙΩΝ**». Όπως ειπώθηκε σκοπός της κατηγορίας αυτής είναι να μπορεί να πληροφορηθεί ο χρήστης για το που βρίσκεται το κοντινότερο εκδοτήριο εισητηρίων σε σχέση με ένα επιθυμητό σημείο στο χάρτη. Με το πάτημα του κουμπιού, οδηγείται σε μια «νέα» οθόνη που εμφανίζεται ο χάρτης της πόλης.



Εικόνα 3.16 Χάρτης κατηγορίας «ΕΚΔΟΤΗΡΙΟ ΕΙΣΗΤΗΡΙΩΝ»

Το επόμενο βήμα είναι η επιλογή ενός σημείου στον χάρτη.



Εικόνα 3.17 Επιλογή σημείου

Στη συνέχεια, υπολογίζεται το κοντινότερο εκδοτήριο εισητηρίων και εμφανίζεται στο χρήστη σχετικό μήνυμα.



Εικόνα 3.18 Πληροφορίες Κοντινότερου Εκδοτηρίου

Εδώ υπάρχουν δύο επιλογές.

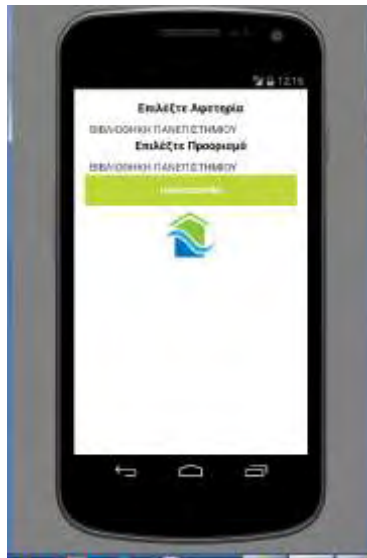
- Πατώντας το κουμπί «ΧΑΡΤΗΣ» προβάλλεται στο χάρτη με marker η θέση του σχετικού εκδοτηρίου.



Εικόνα 3.19 Προβολή Κοντινότερου Εκδοτηρίου

- Πατώντας το icon οδηγείται στο αρχικό menu.

Επόμενο σενάριο είναι η επιλογή «**ΒΕΛΤΙΣΤΗ ΔΙΑΔΡΟΜΗ**» από το αρχικό menu.



Εικόνα 3.20 Μενού «Βέλτιστης Διαδρομής»

Μέσω μιας λίστας που εμφανίζει όλα τα προϋπάρχοντα σημεία μπορεί να επιλέξει τόσο σημείο αφετηρίας όσο και σημείο τερματισμού και στη συνέχεια πατώντας το κουμπί «**ΠΛΗΡΟΦΟΡΙΕΣ**» να πάρει τις σχετικές οδηγίες.



Εικόνα 3.22 Επιλογή Σημείων Ενδιαφέροντος 2

Ας υποθεθεί ότι ως αφετηρία επιλέχθηκε «ΝΟΣΟΚΟΜΕΙΟ» και ως τερματισμός «ΠΑΝΕΠΙΣΤΗΜΙΟ (ΘΟΛΟΣ)».



Εικόνα 3.23 Επιλογή Σημείων Ενδιαφέροντος 3

Το αποτέλεσμα:



Εικόνα 3.24 Αποτέλεσμα Βέλτιστης Διαδρομής

Υλοποίηση της Εφαρμογής

4.1 Γενική Δομή

Η υλοποίηση της εργασίας έχει γίνει σε 2 τμήματα. Το πρώτο είναι η mobile εφαρμογή η οποία εγκαθίσταται στο τηλέφωνο του χρήστη. Το δεύτερο είναι ένας server, όπου υπάρχουν δεδομένα αποθηκευμένα σε μια βάση. Η εφαρμογή έχει απομακρυσμένη πρόσβαση σε αυτές τις πληροφορίες. Προτιμήθηκε η χρήση server ώστε να μειωθεί το μέγεθος της εφαρμογής και των δεδομένων που είναι εγκατεστημένα στην κινητή συσκευή.

Η mobile εφαρμογή είναι γραμμένη σε Java και xml. Αποτελείται όπως θα δούμε παρακάτω από πολλές κλάσεις που επικοινωνούν μεταξύ τους και από ένα xml αρχείο για κάθε κλάση, υπεύθυνο για τη διεπαφή με το χρήστη. Για την επικοινωνία με τον εξυπηρετητή χρησιμοποιείται το πρωτόκολλο https.

Για τον εξυπηρετητή χρησιμοποιήθηκε αρχικά το ελεύθερο λογισμικό XAMPP [8] το οποίο περιέχει τον Apache Server [9] και τη MySQL [10] που μας είναι απαραίτητα. Στη συνέχεια η εργασία πραγματοποιήθηκε με τη χρήση πραγματικού server. Εκεί υλοποιήθηκε σε MySQL μια βάση δεδομένων. Τοποθετήθηκαν επίσης κάποια scripts υλοποιημένα σε γλώσσα προγραμματισμού php [11] για την «επικοινωνία» με τη βάση δεδομένων.

4.2 Αναλυτική Παρουσίαση

Στη συνέχεια δεν θα παρουσιαστεί το σύνολο του κώδικα, αλλά ενδεικτικά τμήματα από κάποιες κλάσεις τα οποία θα δώσουν μια πρώτη εικόνα για το πως δημιουργήθηκε η εφαρμογή. Αρκετές κλάσεις είναι μεταξύ τους παρόμοιες ή υλοποιούν πράγματα για τα οποία μπορεί εύκολα κάποιος να βρει πολλές απαντήσεις στο Internet. [12]

MainActivity.java – activity_main.xml

Η *MainActivity* είναι η κλάση που είναι υπεύθυνη για την υλοποίηση του αρχικού μενού που εμφανίζεται στον χρήστη (σελ.22). Εδώ έχουμε 4 κουμπιά που όταν γίνεται click, το καθένα εξ αυτών μας οδηγεί σε μια νέα activity.

Παρακάτω θα παρουσιαστεί πως δημιουργήθηκε το κουμπί «**ΑΣΤΙΚΟ ΚΤΕΛ ΒΟΛΟΥ**» της αρχικής οθόνης που οδηγεί σε ένα νέο menu αυτής της κατηγορίας, όπως περιγράφηκε στο προηγούμενο κεφάλαιο. Με παρόμοιο τρόπο κατασκευάζονται και τα υπόλοιπα τρία κουμπιά.

```
public class MainActivity extends FragmentActivity {
```

Με τη συνάρτηση onCreate() αρχικοποιείται το activity μας. Μέσα σε αυτή τη συνάρτηση μεταξύ άλλων δηλώνουμε ότι θέλουμε να συνδέσουμε την activity με το αντίστοιχο layout που θα βλέπει ο χρήστης (στην περίπτωση μας το activity_main.xml). Αυτό γίνεται με την έτοιμη συνάρτηση setContentView(R.layout.activity_main);

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
    addListenerOnButton();  
}
```

Στη συνάρτηση addListenerOnButton() ξεκινάμε με τη δημιουργία ενός αντικειμένου τύπου Button, το astiko_Button. Για να το συνδέσουμε με το view του, με το πώς θα φαίνεται δηλαδή στο χρήστη (χρώμα, μέγεθος, τίτλος κλπ) χρησιμοποιούμε τη συνάρτηση findViewById(R.id.astikoButton).

Στο activity_main.xml έχουμε ορίσει λοιπόν παράλληλα την εικόνα του Button με όλες τις απαραίτητες παραμέτρους.

```
<Button  
    android:layout_width="fill_parent"  
    android:layout_height="45dp"  
    android:id="@+id/astikoButton"  
    android:contentDescription="@string/astikoContent"  
    android:clickable="true"  
    android:text="ΑΣΤΙΚΟ ΚΤΕΛ ΒΟΛΟΥ"  
    android:background="#ff306fa3"  
    android:textStyle="bold"  
    android:textColor="#ffffff"  
    android:layout_below="@+id/imageView2"  
    android:layout_alignParentLeft="false"  
    android:layout_alignParentStart="true"  
    android:layout_marginTop="25dp" />
```

```
private void addListenerOnButton() {
```

```
Button astiko_Button = (Button)findViewById(R.id.astikoButton);
```

*Συνεχίζοντας πρέπει να ορίσουμε τι θα γίνει όταν πατήσουμε το κουμπί. Έτσι καλώντας την συν/ση **setOnClickListener** όταν το κουμπί γίνει **onClick()** πηγαίνουμε σε μια νέα συν/ση την **goToAstikoActivity()**;*

```
astiko_Button.setOnClickListener(new View.OnClickListener() {
```

```
public void onClick(View v) {  
    goToAstikoActivity();  
}  
});
```

```
.....
```

```
}
```

*Στην παρακάτω συνάρτηση έχουμε την τελική διαδικασία όπου το πάτημα του κουμπιού που περιγράφηκε πάνω θα μας οδηγήσει σε μια νέα activity. Αυτό γίνεται μέσω του **Intent** [13]. Δημιουργούμε ένα αντικείμενο αυτού του τύπου δηλώνοντας ότι θα συνδεθούμε με την κλάση **AstikoMainActivity**. Με την συν/ση **startActivity** οδηγούμαστε εκεί.*

```
private void goToAstikoActivity(){
```

```
Intent intent = new Intent(this,AstikoMainActivity.class);  
startActivity(intent);
```

```
}
```

```
.....
```

```
}
```

AstikoMainActivity.java – activity_astiko_main.xml

Με την κλάση αυτή δημιουργούμε το μενού της κατηγορίας «**ΑΣΤΙΚΟ ΚΤΕΛ ΒΟΛΟΥ**». Είναι διαμορφωμένο σε μια λίστα με τέσσερις επιλογές όπως παρουσιάστηκε παραπάνω(σελ.22).

```
public class AstikoMainActivity extends Activity {
```

Αρχικά λοιπόν δημιουργούμε το αντικείμενο AstikoList τύπου ListView [14]. Στη συνέχεια στην onCreate() το συνδέουμε με το layout του, Base_list , μέσω της findViewById.

```
ListView AstikoList ;
```

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_astiko_main);  
    addListenerOnButton();  
    AstikoList = (ListView) findViewById(R.id.Base_list) ;
```

Στο activity_astiko_main.xml έχουμε ορίσει το ListView:

```
<ListView  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:id="@+id/Base_list"  
  
    android:footerDividersEnabled="true"  
    android:headerDividersEnabled="true"  
    android:drawSelectorOnTop="false"  
    android:dividerHeight="1dp"  
    android:divider="#ff306fa3"  
    android:layout_alignParentLeft="true"  
    android:layout_alignParentStart="true" />
```

Συνεχίζοντας στην AstikoMainActivity, η λίστα αποτελείται από τις παρακάτω επιλογές, τα ονόματα των οποίων τα δηλώνουμε σε έναν πίνακα από strings που δημιουργούμε.

```
String[] values = new String[] {  
    "Δρομολόγια Γραμμών",  
    "Γραμμές και Στάσεις",  
    "Χάρτης Γραμμών",  
    "Χρήσιμες Πληροφορίες"  
};
```

Στη συνέχεια χρησιμοποιούμε την κλάση ArrayAdapter [15]. Αυτή «στέκεται» ανάμεσα σε μια λίστα στοιχείων ,στην περίπτωση μας τον πίνακα από strings και μιας ListView. Γεμίζει τη λίστα με τα στοιχεία αυτού του πίνακα. Δημιουργούμε λοιπόν τον adapter

```
ArrayAdapter<String> adapter = new ArrayAdapter<>(this,  
    android.R.layout.simple_list_item_activated_2,android.R.id.text2,values);
```

και μέσω της **setAdapter** τον προσαρμόζουμε στη λίστα την οποία είχαμε ορίσει προηγουμένως.

```
AstikoList.setAdapter(adapter);
```

```
AstikoList.setOnItemClickListener(new  
AdapterView.OnItemClickListener() {
```

Μορφοποιήθηκε: Αγγλικά (Ηνωμένων Πολιτειών)

Όπως και νωρίτερα χρησιμοποιούμε την **onItemClick** ώστε να δηλώσουμε τι θα γίνει με το «πάτημα» κάθε στοιχείου της λίστας. Έτσι, με το σχήμα **switch-case**, σε κάθε περίπτωση οδηγούμαστε σε ένα νέο activity.

```
public void onItemClick(AdapterView<?> parent, View view, int position, long  
id) {
```

```
switch( position )  
{  
case 0: Intent newActivity = new Intent(AstikoMainActivity.this,  
Dromologia.class);  
startActivity(newActivity);  
break;  
  
case 1: Intent newActivity1 = new Intent(AstikoMainActivity.this,  
Staseis.class);  
startActivity(newActivity1);  
break;  
  
case 2: Intent newActivity2 = new Intent(AstikoMainActivity.this,  
XartisAstiko.class);  
startActivity(newActivity2);  
break;  
  
case 3: Intent newActivity3 = new Intent(AstikoMainActivity.this,  
BusInformation.class);  
startActivity(newActivity3);  
break;  
}  
}  
});
```

```
}
```

```
.....
```

```
}
```

Με τον ίδιο τρόπο έχουν δημιουργηθεί και τα υπόλοιπα αντίστοιχα **ListViews** της εφαρμογής.

StasiL.java

Με την κλάση αυτή δημιουργούμε τη λειτουργία εμφάνισης των στάσεων για την Γραμμή 1 όπως παρουσιάστηκε (σελ.26). Έχουμε μια «επεκτάσιμη» λίστα αποτελούμενη από δυο στοιχεία, τις «Στάσεις Μετάβασης» και «Στάσεις Επιστροφής». Με το πάτημα της καθεμιάς εμφανίζεται το σύνολο των στάσεων σε μορφή λίστας. Παρόμοιος τρόπος ~~που θα με αυτόν που περιγράφεται~~ παρακάτω ακολουθήθηκε σε κάθε μια από τις υπόλοιπες γραμμές ~~όσο~~ και για την εμφάνιση των δρομολογίων ανάλογα με την ημέρα.

Προτού όμως παρουσιαστεί αυτή η κλάση, Αρχικά—αρχικά λοιπόν κατασκευάζουμε το δικό μας adapter, ο οποίος θα περιέχει όλες τις μεθόδους διαχείρισης της **ExpandableList** [16] που θέλουμε να φτιάχνουμε. Οι 2 βασικές «οντότητες» που παίζουν τον κυριότερο ρόλο είναι τα «παιδιά» και οι «γονείς» τη λίστας. Όπως θα δούμε αργότερα στο παράδειγμα μας ~~θα~~ παιδιά θα είναι οι στάσεις και γονείς οι επιλογές **(Στάσεις Μετάβασης και Στάσεις Επιστροφής)**.

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα, Πλάγια

Ακολουθεί ο κώδικας του adapter **MyExpandableAdapter.java** :

Μορφοποιήθηκε: Ελληνικά (Ελλάδας)

```
import android.app.Activity;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseExpandableListAdapter;
import android.widget.CheckedTextView;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;

public class MyExpandableAdapter extends BaseExpandableListAdapter
{
    private Activity activity;
    private ArrayList<Object> childItems;
    private LayoutInflater inflater;
    private ArrayList<String> parentItems, child;

    // constructor
    public MyExpandableAdapter(ArrayList<String> parents, ArrayList<Object> children)
    {
        this.parentItems = parents;
        this.childItems = children;
    }

    public void setInflater(LayoutInflater inflater, Activity activity)
    {
        this.inflater = inflater;
        this.activity = activity;
    }
}
```



```
// method getChildView is called automatically for each child view.  
// Implement this method as per your requirement
```

*Η παρακάτω μέθοδος **getChildView** είναι υπεύθυνη για το View, της λίστας των παιδιών. καθώς επικοινωνεί με το **child_view.xml** που παρουσιάζεται παρακάτω.*

└

```
public View getChildView(int groupPosition, final int childPosition, boolean  
isLastChild, View convertView, ViewGroup parent)  
{  
  
    child = (ArrayList<String>) childItems.get(groupPosition);  
  
    TextView textView = null;  
  
    if (convertView == null) {  
        convertView = inflater.inflate(R.layout.child_view, null);  
    }  
    // get the textView reference and set the value  
    textView = (TextView) convertView.findViewById(R.id.textViewChild);  
    textView.setText(child.get(childPosition));  
  
    // set the ClickListener to handle the click event on child item  
    convertView.setOnClickListener(new View.OnClickListener() {  
  
        @Override  
        public void onClick(View view) {  
            Toast.makeText(activity, child.get(childPosition),  
                Toast.LENGTH_SHORT).show();  
        }  
    });  
    return convertView;  
}
```

*Η μέθοδος **getParentView** είναι υπεύθυνη για το View, της λίστας των γονέων. καθώς επικοινωνεί με το **parent_view.xml** που παρουσιάζεται παρακάτω.*

└

```
// method getGroupView is called automatically for each parent item  
// Implement this method as per your requirement  
  
public View getGroupView(int groupPosition, boolean isExpanded, View  
convertView, ViewGroup parent)  
{  
  
    if (convertView == null) {  
        convertView = inflater.inflate(R.layout.parent_view, null);  
    }  
    ((CheckedTextView) convertView).setText(parentItems.get(groupPosition));  
    ((CheckedTextView) convertView).setChecked(isExpanded);  
  
    return convertView;  
}  
  
public Object getChild(int groupPosition, int childPosition)  
{  
    return null;  
}  
  
public long getChildId(int groupPosition, int childPosition)  
{  
    return 0;  
}
```

Μορφοποιήθηκε: Γραμματοσειρά: (Προεπιλεγμένη) Times New Roman, 12 pt, Πλάγια

Μορφοποιήθηκε: Γραμματοσειρά: (Προεπιλεγμένη) Times New Roman, 12 pt, Πλάγια, Ελληνικά (Ελλάδας)

Μορφοποιήθηκε: Γραμματοσειρά: (Προεπιλεγμένη) Times New Roman, 12 pt, Πλάγια

Μορφοποιήθηκε: Γραμματοσειρά: (Προεπιλεγμένη) Times New Roman, 12 pt, Πλάγια, Ελληνικά (Ελλάδας)

Μορφοποιήθηκε: Γραμματοσειρά: (Προεπιλεγμένη) Times New Roman, 12 pt, Πλάγια

Μορφοποιήθηκε: Γραμματοσειρά: Πλάγια, Ελληνικά (Ελλάδας)

Μορφοποιήθηκε: Γραμματοσειρά: 13 pt, Όχι Έντονα

Μορφοποιήθηκε: Γραμματοσειρά: 13 pt, Όχι Έντονα

Μορφοποιήθηκε: Γραμματοσειρά: Πλάγια, Ελληνικά (Ελλάδας)

Μορφοποιήθηκε: Ελληνικά (Ελλάδας)

```

}

public int getChildrenCount(int groupPosition)
{
return ((ArrayList<String>) childItems.get(groupPosition)).size();
}

public Object getGroup(int groupPosition)
{
return null;
}

public int getGroupCount()
{
return parentItems.size();
}

public void onGroupCollapsed(int groupPosition)
{
super.onGroupCollapsed(groupPosition);
}

public void onGroupExpanded(int groupPosition)
{
super.onGroupExpanded(groupPosition);
}

public long getGroupId(int groupPosition)
{
return 0;
}

public boolean hasStableIds()
{
return false;
}

public boolean isChildSelectable(int groupPosition, int childPosition)
{
return false;
}
}

```

child view.xml

```

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="40dp"
    android:background="#ffffff"
    android:clickable="true"
    android:orientation="vertical"

    tools:context=".MainActivity" >

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="39dp"
        android:gravity="center_vertical" >

```

Μορφοποιήθηκε: Γραμματοσειρά: 10 pt

Μορφοποιήθηκε: Γραμματοσειρά: 9 pt, Αγγλικά (Ηνωμένων Πολιτειών)

```

<TextView
    android:id="@+id/textViewChild"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"

    android:text="@string/hello_world"
    android:textSize="16sp"
    android:textColor="#1919A3"
    android:textStyle="bold" />
</LinearLayout>

<View
    android:layout_width="match_parent"
    android:layout_height="1dp"
    android:background="#ff306fa3" />
</LinearLayout>

```

parent view.xml

```

<CheckedTextView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/textViewGroupName"
    android:layout_width="wrap_content"
    android:layout_height="60sp"

    android:gravity="center_vertical"
    android:text="@string/hello_world"
    android:textSize="18sp"
    android:textColor="#FFFFFF"
    android:padding="10dp"
    android:textSelectHandleLeft="@string/hello_world"
    android:background="#ff306fa3"
    android:textStyle="bold" />

```

Ακολουθεί η τελική κλάση **StasiI**, όπου θα χρησιμοποιήσουμε τον **Adapter**, που φτιάξαμε προηγουμένως. Θα δημιουργήσουμε τη λίστα και θα βάλουμε σε αυτήν τα δεδομένα που θέλουμε.

```
public class StasiI extends ExpandableListActivity {
```

Δημιουργούμε δυο **ArrayLists**, ένα για τους γονείς και ένα για τα παιδιά.

```
private ArrayList<String> parentItems = new ArrayList<String>();
private ArrayList<Object> childItems = new ArrayList<Object>();
```

```
public void onCreate(Bundle savedInstanceState) {
```

```
super.onCreate(savedInstanceState);
```

Δημιουργούμε την **Expandable List** με βάση την κλάση **ExpandableListView** και θέτουμε τις ιδιότητες της.

```
ExpandableListView expandableList = getExpandableListView();
expandableList.setDividerHeight(2);
expandableList.setGroupIndicator(null);
expandableList.setClickable(true);
```

Μορφοποιήθηκε: Γραμματοσειρά: (Προεπιλεγμένη) Courier New, 9 pt

Μορφοποιήθηκε: Γραμματοσειρά: 9 pt, Αγγλικά (Ηνωμένων Πολιτειών)

Μορφοποιήθηκε: Γραμματοσειρά: 13 pt, Όχι Έντονα, Αγγλικά (Ηνωμένων Πολιτειών)

Μορφοποιήθηκε: Γραμματοσειρά: 13 pt, Όχι Έντονα

Μορφοποιήθηκε: Γραμματοσειρά: 13 pt

Μορφοποιήθηκε: Γραμματοσειρά: 13 pt, Ελληνικά (Ελλάδας)

Μορφοποιήθηκε: Γραμματοσειρά: 13 pt, Όχι Έντονα, Ελληνικά (Ελλάδας)

Μορφοποιήθηκε: Γραμματοσειρά: 13 pt, Όχι Έντονα

Μορφοποιήθηκε: Γραμματοσειρά: 13 pt, Όχι Έντονα, Ελληνικά (Ελλάδας)

Μορφοποιήθηκε: Γραμματοσειρά: 13 pt, Όχι Έντονα

Μορφοποιήθηκε: Αγγλικά (Ηνωμένων Πολιτειών)

Μορφοποιήθηκε: Γραμματοσειρά: Πλάγια, Χρώμα γραμματοσειράς: Αυτόματο

Μορφοποιήθηκε: Πλήρης

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα, Πλάγια, Χρώμα γραμματοσειράς: Αυτόματο

Μορφοποιήθηκε: Γραμματοσειρά: Πλάγια, Χρώμα γραμματοσειράς: Αυτόματο

Μορφοποιήθηκε: Γραμματοσειρά: (Προεπιλεγμένη) Times New Roman, 13 pt, Πλάγια, Χρώμα γραμματοσειράς: Αυτόματο

Μορφοποιήθηκε: Αγγλικά (Ηνωμένων Πολιτειών)

Μορφοποιήθηκε: Γραμματοσειρά: (Προεπιλεγμένη) Times New Roman, 13 pt, Πλάγια, Χρώμα γραμματοσειράς: Αυτόματο

Μορφοποιήθηκε: Γραμματοσειρά: (Προεπιλεγμένη) Times New Roman, 13 pt, Πλάγια, Χρώμα γραμματοσειράς: Αυτόματο, Αγγλικά (Ηνωμένων Πολιτειών)

Μορφοποιήθηκε: Πλήρης, Εσοχή: Αριστερά, 1,27 εκ.

Μορφοποιήθηκε: Αγγλικά (Ηνωμένων Πολιτειών)

Μορφοποιήθηκε: Πλήρης

Καλούμε την συν/ση που προσθέτουμε τους γονείς («Στάσεις Μετάβασης» & «Στάσεις Επιστροφής»)

```
adapter.setGroupParents();
```

Καλούμε την συν/ση που προσθέτουμε τα παιδιά(Στάσεις)

```
adapter.setChildData();
```

Δημιουργούμε τον Adapter με βάση την κλάση που φτιάξαμε πριν.

```
MyExpandableAdapter adapter = new MyExpandableAdapter(parentItems, childItems);
```

```
adapter.setInflater((LayoutInflater) getSystemService(Context.LAYOUT_INFLATER_SERVICE), this);
```

Τον προσαρμόζουμε στην expandableList

```
expandableList.setAdapter(adapter);  
expandableList.setOnChildClickListener(this);  
}
```

```
public void setGroupParents() {  
    parentItems.add("Στάσεις Μετάβασης");  
    parentItems.add("Στάσεις Επιστροφής");  
}
```

```
public void setChildData() {  
    // Add Child Items  
    ArrayList<String> child = new ArrayList<String>();  
    child.add("Ανδρους");  
    child.add("Νοσοκομείο");  
    child.add("Στρατηγού Σαράφη");  
    child.add("Φιλίππου Ιωάννου");  
    .....  
    childItems.add(child);  
    // Add Child Items  
    child = new ArrayList<String>();  
    child.add("Πέτρου και Παύλου");  
    child.add("Κυρίλλου");  
    child.add("Χαλκίδωνος");  
    child.add("Βαμβακουργιά");  
    .....  
    childItems.add(child); }
```

Μορφοποιήθηκε: Γραμματοσειρά: (Προεπιλεγμένη) Times New Roman, 13 pt, Πλάγια, Χρώμα γραμματοσειράς: Αυτόματο

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα

Μορφοποιήθηκε: Χρώμα γραμματοσειράς: Μαύρο

Μορφοποιήθηκε: Αγγλικά (Ηνωμένων Πολιτειών)

Μορφοποιήθηκε: Γραμματοσειρά: (Προεπιλεγμένη) Times New Roman, 13 pt, Πλάγια, Χρώμα γραμματοσειράς: Αυτόματο

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα

Μορφοποιήθηκε: Γραμματοσειρά: (Προεπιλεγμένη) Times New Roman, 13 pt, Πλάγια, Χρώμα γραμματοσειράς: Αυτόματο

Μορφοποιήθηκε: Αγγλικά (Ηνωμένων Πολιτειών)

Μορφοποιήθηκε: Γραμματοσειρά: (Προεπιλεγμένη) Times New Roman, 13 pt, Πλάγια, Χρώμα γραμματοσειράς: Αυτόματο

Μορφοποιήθηκε: Γραμματοσειρά: (Προεπιλεγμένη) Times New Roman, 13 pt, Πλάγια

Μορφοποιήθηκε: Αγγλικά (Ηνωμένων Πολιτειών)

Μορφοποιήθηκε: Αγγλικά (Ηνωμένων Πολιτειών)

Μορφοποιήθηκε ...

Μορφοποιήθηκε ...

Μορφοποιήθηκε ...

Μορφοποιήθηκε ...

Μορφοποιήθηκε ...

Μορφοποιήθηκε ...

Μορφοποιήθηκε ...

Μορφοποιήθηκε ...

Μορφοποιήθηκε ...

Μορφοποιήθηκε ...

Μορφοποιήθηκε ...

Μορφοποιήθηκε ...

Μορφοποιήθηκε ...

Μορφοποιήθηκε ...

Μορφοποιήθηκε ...

Μορφοποιήθηκε ...

Μορφοποιήθηκε ...

XartisGrammi2.java

Με την κλάση αυτή δημιουργούμε την προβολή του δρομολογίου της Γραμμής Νο2 στον χάρτη. Για να το πετύχουμε αυτό πρέπει να χρησιμοποιήσουμε το **Google Maps Android API**. Ένας πλήρης οδηγός μπορεί να βρεθεί στον ιστότοπο της GoogleDevelopers [17].

*Για να μπορεί να χρησιμοποιήσει κάποιος το παραπάνω API πρέπει να λάβει από την Google ένα συγκεκριμένο κλειδί. Με το που φτιάξει μια Google Maps activity δημιουργείται αυτόματα και ένα xml αρχείο το **google_maps_api.xml**. Εκεί δίνεται ένα link. Ακολουθώντας τις οδηγίες μπορεί να αποκτήσει key για το project του.*

Αρχικά, πέραν της υλοποίησης της κλάσης είναι απαραίτητη προϋπόθεση να «ενεργοποιήσουμε» κάποια permissions στο

AndroidManifest.xml .

```
.....  
<uses-permission android:name="android.permission.INTERNET" />  
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />  
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />  
<uses-permission  
  android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />  
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />  
  
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />  
  
.....  
<meta-data  
  android:name="com.google.android.gms.version"  
  android:value="@integer/google_play_services_version" />  
  
<meta-data  
  android:name="com.google.android.maps.v2.API_KEY"  
  android:value="@string/google_maps_key" />  
  
.....
```

Στην κλάση **XartisGrammi2.java** λοιπόν έχουμε :

```
public class XartisGrammi2 extends FragmentActivity {
```

Δημιουργούμε ένα αντικείμενο τύπου **GoogleMap**.

```
private GoogleMap mMap;
```

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_xartis_grammi2);  
    setUpMapIfNeeded();  
}
```

```
@Override  
protected void onResume() {  
    super.onResume();  
    setUpMapIfNeeded();  
}
```

```
private void setUpMapIfNeeded() {
```

Ελέγχουμε αν το αντικείμενο μας είχε κάποια τιμή. Αν όχι «παίρνουμε» την αναφορά στον χάρτη από το **xml** αρχείο. Ελέγχουμε αν έχουμε επιτυχώς έναν χάρτη. Αν ναι ,τότε καλούμε τις συν/σεις **setUpMap** και **addLines** διαδοχικά.

```
if (mMap == null) {
```

```
    mMap = ((SupportMapFragment)  
            getSupportFragmentManager().findFragmentById(R.id.map))  
            .getMap();
```

```
if (mMap != null) {  
    setUpMap();  
}  
}
```

Προσθέτουμε έναν **marker** στον χάρτη σε μια συγκεκριμένη θέση και με τον τίτλο που επιθυμούμε.

```
private void setUpMap() {  
    mMap.addMarker(new MarkerOptions().position(new LatLng(39.36137,  
22.93258)).title("Αφειτηρία Γραμμής Νο2"));  
    addLines();  
}
```

Στην παρακάτω συν/ση προσθέτουμε στον χάρτη γραμμές που δείχνουν το δρομολόγιο μετάβασης και επιστροφής. Αυτό το επιτυγχάνουμε χρησιμοποιώντας το **Interactive Polyline Encoder Utility** [18] που μας προσφέρει το **Google Maps API**. Επιλέγοντας τα σημεία που θέλουμε πάνω στον χάρτη που υπάρχει εκεί, κωδικοποιείται η διαδρομή που θέλουμε. Στην συνέχεια στον κώδικα μας πολύ απλά χρησιμοποιώντας την **PolyUtil.decode** αποκωδικοποιούμε και έχουμε το επιθυμητό αποτέλεσμα.

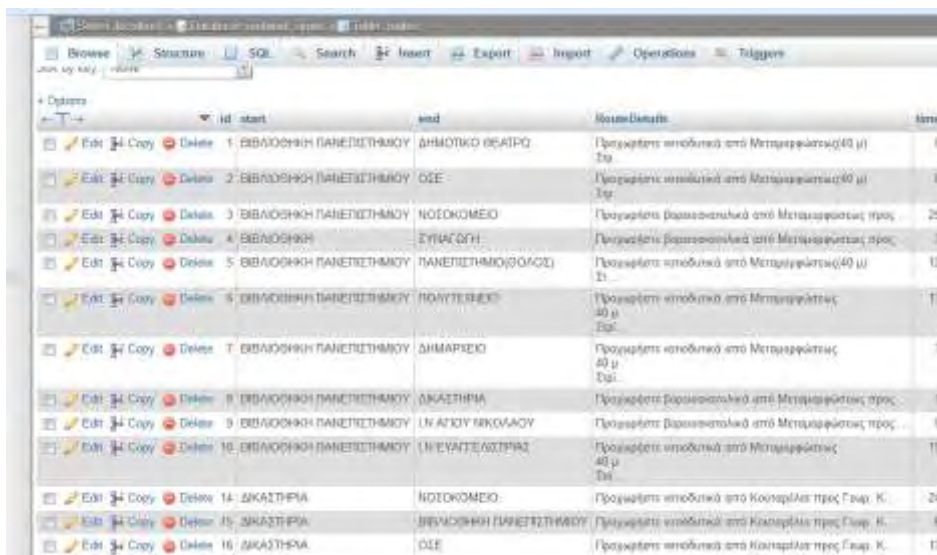
Τέλος με την `mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(start,15));` καθορίζουμε το σημείο που θα «κεντράρει» η οθόνη του χάρτη όταν θα την ανοίξει ο χρήστης.

```
private void addLines() {  
    LatLng start= new LatLng(39.36137, 22.93258);  
  
    mMap.addPolyline((new PolylineOptions())  
        .addAll(PolyUtil.decode("iwfoFc~jCIqAs@kCeAoEuAuFU{ALuAV_Cb@iC@mE?_LJgBrAqB  
Aa@n@mCb@eBnAeDdA{C~AeEkF}EsG{FwIwH{HeHSOaCbF}BzEwCtGkA`CkCzFeCxF{BbFAr@nBjH  
zB|Hd@bBmCXaCVqC^eDZsDb@mD`@mD`@gBNeC\\s@JiB_DkB{CyB}Dwa@u@d@eBbA}CxAkBr@sBr  
@u@T}@H{@KsBc@iBSiA@{@P_Bf@UL{Ad@BTVf@t@|B`A~Cr@vBvAxCzAzCj@rAtApC_B`A{CfB_B  
|@a@q@eB}DeBgDuAqCqBeE_@_Ao@mDWwBSI_AfKbD@mA\\OJMCCiAT]l@Ot@Y^iAh@sCz@eEvAkG  
uB}IfA")) .width(5) .color(Color.RED)  
        .geodesic(true));  
  
    mMap.addPolyline((new PolylineOptions())  
        .addAll(PolyUtil.decode("eukoFw|_kCStDx@b@Kf@y@~Dw@jDY^kATWRG^Dn@DTb@AtCu@xA  
YPEP~Ab@tCl@nBhBvD`CxEzBxEvAnC|A}@bCwApAu@p@]i@iAsAmCgBuDgAwBkAgDmAeEw@sBBKh  
Aa@|Ag@rA[v@Gz@FjBXbBZjATVWl@QlC{@|DcBfCuAjAu@jBbDhB`D`CdEVLfBWfM`CSbCWdC[p  
C[`CW~BYvC_@`D]rBSZOk@qBw@gCm@wBuBiHSmAdA}BbByDtBuEtBsEnAoChBeEbCkFj@{@|CrC?  
?lCdCzBvBjBdB~CtCtAfAhClBtDjCa@rA_AdCiAvC{@bDUtA@nE@lEC|FEpEt@^j@bA^`Bb@nBhA  
hElAzEoAv@t@nCz@dDuBz@YG")) .width(5) .color(Color.BLUE)  
        .geodesic(true));  
  
    mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(start,15));  
}  
  
}
```

Μορφοποιήθηκε: Πλήρης

VeltistiDiadromi.java

Η κλάση αυτή είναι υπεύθυνη για την εμφάνιση των οδηγιών βέλτιστης διαδρομής μεταξύ συγκεκριμένων σημείων ενδιαφέροντος. Η εφαρμογή μας αντλεί τις πληροφορίες από τον πίνακα **routes** που έχουμε σε μια βάση δεδομένων που δημιουργήθηκε στο <http://www.multihosting.gr/>



	id	start	end	RouteDetails	time
	1	ΕΒΔΛΟΘΗΚΗ ΠΑΝΕΠΙΣΤΗΜΙΟΥ	ΔΗΜΟΤΙΚΟ ΘΕΑΤΡΟ	Προσκαίτεται υποδοτικό από Μεταμαρφώσεως(40 μ) Ση	6
	2	ΕΒΔΛΟΘΗΚΗ ΠΑΝΕΠΙΣΤΗΜΙΟΥ	ΟΔΕ	Προσκαίτεται υποδοτικό από Μεταμαρφώσεως(40 μ) Ση	8
	3	ΕΒΔΛΟΘΗΚΗ ΠΑΝΕΠΙΣΤΗΜΙΟΥ	ΝΟΣΟΚΟΜΕΙΟ	Προσκαίτεται βαρυσκοπικό από Μεταμαρφώσεως προς	25
	4	ΕΒΔΛΟΘΗΚΗ	ΣΥΝΑΓΩΓΗ	Προσκαίτεται βαρυσκοπικό από Μεταμαρφώσεως προς	3
	5	ΕΒΔΛΟΘΗΚΗ ΠΑΝΕΠΙΣΤΗΜΙΟΥ	ΠΑΝΕΠΙΣΤΗΜΙΟ(ΟΔΟΣ)	Προσκαίτεται υποδοτικό από Μεταμαρφώσεως(40 μ) Ση	12
	6	ΕΒΔΛΟΘΗΚΗ ΠΑΝΕΠΙΣΤΗΜΙΟΥ	ΠΟΛΥΤΕΧΝΕΙΟ	Προσκαίτεται υποδοτικό από Μεταμαρφώσεως (40 μ) Ση	13
	7	ΕΒΔΛΟΘΗΚΗ ΠΑΝΕΠΙΣΤΗΜΙΟΥ	ΔΗΜΑΡΧΕΙΟ	Προσκαίτεται υποδοτικό από Μεταμαρφώσεως (40 μ) Ση	3
	8	ΕΒΔΛΟΘΗΚΗ ΠΑΝΕΠΙΣΤΗΜΙΟΥ	ΔΙΚΑΣΤΗΡΙΑ	Προσκαίτεται βαρυσκοπικό από Μεταμαρφώσεως προς	7
	9	ΕΒΔΛΟΘΗΚΗ ΠΑΝΕΠΙΣΤΗΜΙΟΥ	ΛΗ ΑΓΙΟΥ ΝΙΚΟΛΑΟΥ	Προσκαίτεται βαρυσκοπικό από Μεταμαρφώσεως προς	9
	10	ΕΒΔΛΟΘΗΚΗ ΠΑΝΕΠΙΣΤΗΜΙΟΥ	ΛΗ ΕΥΑΓΓΕΛΙΣΤΡΙΑΣ	Προσκαίτεται υποδοτικό από Μεταμαρφώσεως (40 μ) Ση	19
	14	ΔΙΚΑΣΤΗΡΙΑ	ΝΟΣΟΚΟΜΕΙΟ	Προσκαίτεται υποδοτικό από Κουσαρίλας προς Γουρ. Κ.	24
	15	ΔΙΚΑΣΤΗΡΙΑ	ΕΒΔΛΟΘΗΚΗ ΠΑΝΕΠΙΣΤΗΜΙΟΥ	Προσκαίτεται υποδοτικό από Κουσαρίλας προς Γουρ. Κ.	6
	16	ΔΙΚΑΣΤΗΡΙΑ	ΟΔΕ	Προσκαίτεται υποδοτικό από Κουσαρίλας προς Γουρ. Κ.	13

Εικόνα 4.1 Πίνακας Οδηγιών Βέλτιστης Διαδρομής

Ο παραπάνω πίνακας περιέχει τις πληροφορίες για όλες τις πιθανές διαδρομές μεταξύ αφετηρίας και τερματισμού. Οι πληροφορίες έχουν αντληθεί από τις υπηρεσίες του GoogleMaps. Προτιμήθηκε η βάση δεδομένων κυρίως λόγω του μεγάλου όγκου δεδομένων με σκοπό να μην επιφορτιστεί η εφαρμογή μας με αυτόν.

Ο πίνακας αποτελείται από τις εξής στήλες.

- *Id*: Αναγνωριστικό
- *start*: Σημείο Αφετηρίας
- *end*: Σημείο Τερματισμού
- *RouteDetails*: Οι αντίστοιχες οδηγίες
- *Time*: χρόνος διαδρομής

Μέσω ενός SQL ερωτήματος που υπάρχει σε php αρχείο, όπως φαίνεται παρακάτω, παίρνουμε τα στοιχεία από τον πίνακα μας, ώστε στην συνέχεια να συγκριθούν με τις επιλογές που θα κάνει ο χρήστης από την εφαρμογή και να επιλεγεί από τον πίνακα το κατάλληλο πεδίο πληροφοριών.

demo.php

```
<?php

$conn=mysql_connect("address","username","password") or
die(mysql_error());

mysql_query("SET NAMES 'utf8'", $conn);

mysql_select_db("database_name");

$sql=mysql_query("select * from table_name");
while($row=mysql_fetch_assoc($sql))
$output[]=$row;
print(json_encode($output));
mysql_close();
?>
```



```
public class VeltistiDiadromi extends Activity implements OnClickListener{
    Button fetch,start;
    TextView text,text2;
    Spinner et,et2;

    .....

    Συνεχίζοντας στην κλάση έχουμε αρχικά δυο spinners που τα γεμίζουμε με τα
    στοιχεία που θέλουμε.
```

```
public void addItemOnSpinner1() {

    et = (Spinner) findViewById(R.id.spinner1);
    List<String> list = new ArrayList<String>();
    list.add("ΒΙΒΛΙΟΘΗΚΗ ΠΑΝΕΠΙΣΤΗΜΙΟΥ");
    list.add("ΟΤΕ");
    list.add("ΟΣΕ");
    .....

    ArrayAdapter<String> dataAdapter = new ArrayAdapter<String>(this,
    android.R.layout.simple_spinner_item, list);
    dataAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown
    _item);
    et.setAdapter(dataAdapter);
}

public void addItemOnSpinner2() {

    et2 = (Spinner) findViewById(R.id.spinner2);
    List<String> list = new ArrayList<String>();
```

```

list.add("ΒΙΒΛΙΟΘΗΚΗ ΠΑΝΕΠΙΣΤΗΜΙΟΥ");
list.add("ΟΤΕ");
list.add("ΟΣΕ");
list.add("ΔΗΜΑΡΧΕΙΟ");

.....

ArrayAdapter<String> dataAdapter = new ArrayAdapter<String>(this,
android.R.layout.simple_spinner_item, list);
dataAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_
_item);
et2.setAdapter(dataAdapter);
}

.....

```

Με την παρακάτω κλάση επικοινωνούμε με τον server.

```

class task extends AsyncTask<String, String, Void>
{
private ProgressDialog progressDialog = new
ProgressDialog(VeltistiDiadromi.this);
InputStream is = null ;
String result = "";
protected void onPreExecute() {
progressDialog.setMessage("Fetching data...");
progressDialog.show();
progressDialog.setOnCancelListener(new OnCancelListener() {
@Override
public void onCancel(DialogInterface arg0) {
task.this.cancel(true);
}
});
}
@Override
protected Void doInBackground(String... params) {

```

Ανλώνουμε την url με την τοποθεσία που βρίσκεται το php αρχείο μας στον server.

```

String url_select = "http://url/demo.php";

HttpClient httpClient = new DefaultHttpClient();
HttpPost httpPost = new HttpPost(url_select);

ArrayList<NameValuePair> param = new ArrayList<NameValuePair>();

try {
httpPost.setEntity(new UrlEncodedFormEntity(param));
HttpResponse httpResponse = httpClient.execute(httpPost);
HttpEntity httpEntity = httpResponse.getEntity();

//read content
is = httpEntity.getContent();

} catch (Exception e) {

Log.e("log_tag", "Error in http connection "+e.toString());

}
try {
BufferedReader br = new BufferedReader(new InputStreamReader(is));
StringBuilder sb = new StringBuilder();
String line = "";

```

```

while((line=br.readLine())!=null)
{
    sb.append(line+"\n");
}
is.close();
result=sb.toString();

} catch (Exception e) {
    // TODO: handle exception
    Log.e("log_tag", "Error converting result "+e.toString());
}

return null;

}

protected void onPostExecute(Void v) {

```

```

try {
    JSONArray Jarray = new JSONArray(result);
    for(int i=0;i<Jarray.length();i++)
    {
        JSONObject Jasonobject = null;

        Jasonobject = Jarray.getJSONObject(i);

        String start = Jasonobject.getString("start");
        String end = Jasonobject.getString("end");
        String db_detail="";

```

Ελέγχουμε αν τα στοιχεία που επιλέξαμε από τα spinners συμπίπτουν με τα στοιχεία στον πίνακα μας. Αν ναι τότε παίρνουμε τις πληροφορίες από το αντίστοιχο πεδίο του πίνακα και το εμφανίζουμε στον χρήστη.

```

if(et.getSelectedItemAt().toString().equalsIgnoreCase(start)           &&
et2.getSelectedItemAt().toString().equalsIgnoreCase(end)) {
    db_detail = Jasonobject.getString("RouteDetails");
    text.setText(db_detail);

break;
}

}
this.progressBar.dismiss();

} catch (Exception e) {
    // TODO: handle exception
    Log.e("log_tag", "Error parsing data "+e.toString());
}
}

.....
}

```

BusOnDemand.java

Πρόκειται για την αρχική μας κλάση στην διαδικασία «ΑΝΑΖΗΤΗΣΗΣ ΓΡΑΜΜΗΣ». Η λειτουργία που υλοποιείται εδώ είναι η εξής:

Υπολογίζονται και ταξινομούνται σε αύξουσα σειρά οι αποστάσεις των σημείων αφετηρίας και τερματισμού που επιλέγει ο χρήστης στον χάρτη, από τα σημεία όλων των στάσεων που υπάρχουν. Με αυτό τον τρόπο τόσο για την αφετηρία όσο και για τον τερματισμό υπάρχουν ταξινομημένες σε πίνακες οι κοντινότερες στάσεις.

```
public class BusOnDemandActivity extends FragmentActivity {  
    private GoogleMap mMap; // Might be null if Google Play services APK is not  
    available.
```

```
    float x1,x2;  
    float y1,y2;  
    int metritis=0;
```

*Δηλώνουμε και αρχικοποιούμε τους πίνακες **distances[]** και **distances2[]**, που θα περιέχουν τις αποστάσεις από αφετηρία και τερματισμό αντίστοιχα.*

```
    double  
    distances[]={ (float)0.0, (float)0.0, (float)0.0, (float)0.0, (float)0.0, (float)0  
    .0, (float)0.0, (float)0.0, (float)0.0, ..... (float)0.0 };
```

```
    double distances2[]=distances.clone();
```

Δηλώνουμε τις συντεταγμένες x & y όλων των στάσεων που υπάρχουν και τα ονόματά τους με την αντίστοιχη σειρά.

```
    double  
    xNums[]={ (float)39.35061, (float)39.353075, (float)39.354653, (float)39.355657,  
    (float)39.356642, (float)39.357648, (float)39.358373, (float)39.360179 ,  
    ..... (float)39.364325  
    };  
    double xNums2[]=xNums.clone();
```

```
    double  
    yNums[]={ (float)22.96362, (float)22.962304, (float)22.960117, (float)22.958199,  
    (float)22.956254,  
    ..... (float)22.963189
```

```
    };  
    double yNums2[]=yNums.clone();
```

```
    String  
    names[]={ "Αναυρος", "Νοσοκομείο (Πολυμέρη)", "Στρατηγού  
    Σαράφη", "Φιλίππου Ιωάννου", "Βλαχάβα (Πολυμέρη)", "Κασσαβέτη",  
    ..... "Περραιβού (Οδός Γ. Δήμου)" };
```

```
    String names2[]=names.clone();
```

```
    protected void onCreate(Bundle savedInstanceState) {
```

```

super.onCreate(savedInstanceState);
setContentView(R.layout.activity_bus_on_demand);
setUpMapIfNeeded();

final Toast infomsg=Toast.makeText(this,"Επιλέξτε σημείο αφετηρίας και
σημείο τερματισμού!",Toast.LENGTH_LONG);
infomsg.show();

```

Με την παρακάτω συν/ση δίνεται η δυνατότητα επιλογής ενός σημείου στον χάρτη. Με την μεταβλητή **metritis** εξασφαλίζουμε ότι αυτό μπορεί να γίνει το πολύ δυο φορές. Επίσης για κάθε σημείο καλούνται δυο συναρτήσεις οι οποίες παρουσιάζονται παρακάτω. Η πρώτη είναι η **distFrom** και υπολογίζει τις αποστάσεις των σημείων επιλογής στον χάρτη από όλες τις στάσεις. Η δεύτερη είναι η γνωστή συνάρτηση ταξινόμησης **bubblesort** και χρησιμοποιείται για να ταξινομήσει όπως είπαμε το αποτέλεσμα της προηγούμενης συνάρτησης.

```

mMap.setOnMapClickListener(new GoogleMap.OnMapClickListener() {

@Override
public void onMapClick(LatLng latLng) {

if(metritis==0) {
metritis++;
// First point

MarkerOptions markerOptions = new MarkerOptions();
markerOptions.position(latLng);
markerOptions.title("Βρίσκεισαι εδώ!");
x1 = (float) latLng.latitude;
y1 = (float) latLng.longitude;

// Animating to the touched position
mMap.animateCamera(CameraUpdateFactory.newLatLng(latLng));
mMap.addMarker(markerOptions);
int i;
for (i = 0; i < 216; i++) {
distances[i] = distFrom(x1, y1, (float) xNums[i], (float) yNums[i]);
}
bubbleSort(distances, xNums, yNums, names);

}
else{
// Second point
MarkerOptions markerOptions = new MarkerOptions();
markerOptions.position(latLng);
markerOptions.title("Προορισμός!");
x2 = (float) latLng.latitude;
y2 = (float) latLng.longitude;

mMap.animateCamera(CameraUpdateFactory.newLatLng(latLng));
mMap.addMarker(markerOptions);
int i;
for (i = 0; i < 216; i++) {
distances2[i] = distFrom(x2, y2, (float) xNums2[i], (float) yNums2[i]);
}
bubbleSort(distances2, xNums2, yNums2, names2);

goToBus2onDemandActivity();

}
}

```

```

}
});

```

```

}

```

H bubbleSort:

```

private static void bubbleSort(double[] distArray, double[] xNums, double[]
yNums, String [] names) {

    int n = distArray.length;
    double temp = 0.0;
    double temp2=0.0;
    double temp3=0.0;
    String tempName="";

    for(int i=0; i < n; i++){
        for(int j=1; j < (n-i); j++){

            if(distArray[j-1] > distArray[j]){
                //swap the elements!
                temp = distArray[j-1];
                distArray[j-1] = distArray[j];
                distArray[j] = temp;

                temp2 = xNums[j-1];
                xNums[j-1] = xNums[j];
                xNums[j] = temp2;

                temp3 = yNums[j-1];
                yNums[j-1] = yNums[j];
                yNums[j] = temp3;

                tempName = names[j-1];
                names[j-1] = names[j];
                names[j] = tempName;
            }

        }
    }

}
)

```

H distFrom:

Παίρνει ως παραμέτρους τις συντεταγμένες δυο σημείων. Το 1^ο έχει (xstart, ystart) και το 2^ο (lat2, lng2) και υπολογίζει την μεταξύ τους απόσταση.

```

    public static float distFrom (float xstart, float ystart, float lat2,
float lng2 )
    {
        double earthRadius = 3958.75;
        double dLat = Math.toRadians(lat2-xstart);
        double dLng = Math.toRadians(lng2-ystart);
        double a = Math.sin(dLat/2) * Math.sin(dLat/2) +
            Math.cos(Math.toRadians(xstart)) *
Math.cos(Math.toRadians(lat2)) *
            Math.sin(dLng/2) * Math.sin(dLng/2);
        double c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
        double dist = earthRadius * c;

        int meterConversion = 1609;

```

```

        return new Float(dist * meterConversion).floatValue();
    }
}

```

.....

Στη συνέχεια μέσω της επόμενης συν/σης θα οδηγηθούμε σε μια νέα activity στην οποία θα επικοινωνήσουμε με έναν πίνακα στη βάση δεδομένων, τον **busondemand**. Μέσω ενός αντικειμένου τύπου **Bundle** [19] μπορούμε να περάσουμε παραμέτρους στην νέα activity **FinalNew**. Στην προκειμένη περίπτωση, τους ταξινομημένους πίνακες ονομάτων και συντεταγμένων.

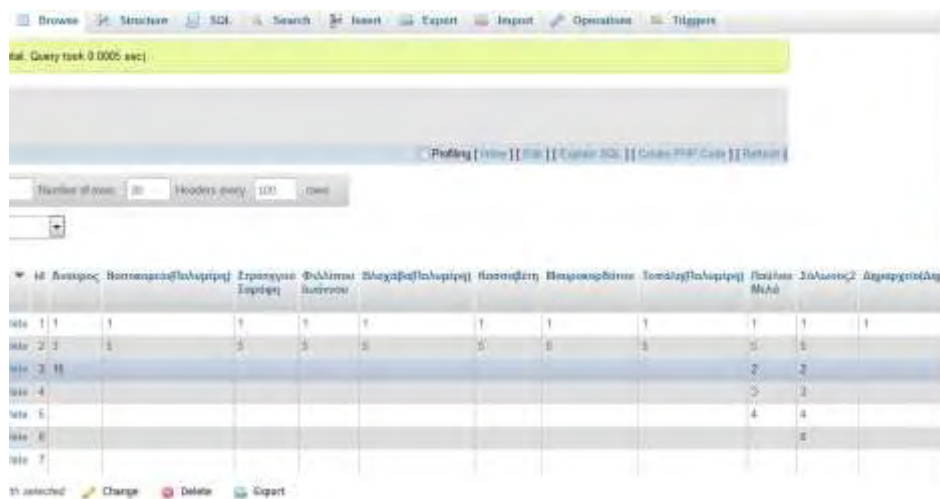
```

private void goToBus2onDemandActivity() {

    Intent intent = new Intent(this, FinalNew.class);
    Bundle b = new Bundle();
    b.putStringArray("names_start", names);
    b.putDoubleArray("x_start", xNums);
    b.putDoubleArray("y_start", yNums);
    b.putStringArray("names_start2", names2);
    b.putDoubleArray("x_start2", xNums2);
    b.putDoubleArray("y_start2", yNums2);
    b.putFloat("x1", x1);
    b.putFloat("y1", y1);
    b.putFloat("x2", x2);
    b.putFloat("y2", y2);
    intent.putExtras(b);
    startActivityForResult(intent, 0);
}
}

```

Ο πίνακας **busondemand** στη βάση δεδομένων στον server:



ID	Αριθμός	Ημερομηνία/Ώρα	Ώρα	Φόρτος	Ώρα/Ώρα	Ώρα/Ώρα	Ώρα/Ώρα	Ώρα/Ώρα	Ώρα/Ώρα	Ώρα/Ώρα	Ώρα/Ώρα
1	1	1	1	1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7	7	7	7	7

Εικόνα 4.2 Πίνακας Γραμμών Εξυπηρέτησης

Σε αυτόν τον πίνακα υπάρχουν συγκεντρωμένες όλες οι στάσεις. Κάθε στήλη είναι και μια στάση. Καθεμία έχει τις γραμμές λεωφορείων που τη

χρησιμοποιούν. Μέσω ενός SQL ερωτήματος όπως και στην προηγούμενη περίπτωση παίρνουμε τα πεδία του πίνακα.

FinalNew.java

```
public class FinalNew extends Activity implements View.OnClickListener {
    Button fetch, menu_b, drom_b;
    TextView text;
    String[] elem;
    String[] namesStart;
    ArrayList<String> elements= new ArrayList<>();
    String[] namesEnd;
    String nStart, nEnd;
    double[] x_start;
    double[] y_start;
    double[] x_end;
    double[] y_end;
    float x1, y1, x2, y2;
    String end;
    String start;

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

        addListenerOnButton();
    }
}
```

Παίρνουμε τις παραμέτρους που «στέίλαμε» πριν μέσω του Bundle.

```
Bundle el = this.getIntent().getExtras();
namesStart = el.getStringArray("names_start");
namesEnd = el.getStringArray("names_start2");
x_start=el.getDoubleArray("x_start");
y_start=el.getDoubleArray("y_start");
x_end=el.getDoubleArray("x_start2");
y_end=el.getDoubleArray("y_start2");
x1=el.getFloat("x1");
y1=el.getFloat("y1");
x2=el.getFloat("x2");
y2=el.getFloat("y2");
fetch= (Button) findViewById(R.id.fetch);
text = (TextView) findViewById(R.id.text);
```

```
fetch.setOnClickListener(this);
}
```

.....

Επικοινωνούμε με τον server με τρόπο παρόμοιο με αυτόν που παρουσιάστηκε νωρίτερα.

```
class task extends AsyncTask<String, String, Void>
{
    .....
    protected Void doInBackground(String... params) {
```

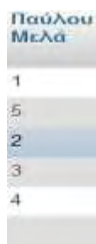

Δηλώνουμε την url με την τοποθεσία που βρίσκεται το php αρχείο μας στον server.

```
String url_select = "http://url/ondemand.php";  
.....  
}  
protected void onPostExecute(Void v) {  
  
try {  
  
int i;  
int k=0;  
boolean temp=true;  
JSONArray Jarray = new JSONArray(result);
```

Στον πίνακα **namesEnd** έχουμε ταξινομημένες τις στάσεις με βάση το σημείο τερματισμού. Στον **namesStart** με βάση το σημείο αφετηρίας αντίστοιχα. Βρίσκουμε στον πίνακα **busondemand** τα αντίστοιχα πεδία με τα ονόματα των 1^{ων} στοιχείων των 2 πινάκων. Συγκρίνουμε μεταξύ τους τις γραμμές αυτών των πεδίων, ώστε να βρούμε όλα τα πιθανά ταιριάσματα, δηλαδή όλες τις κοινές γραμμές εξυπηρέτησης. Αν δεν έχουν κάτι κοινό, τότε δίνουμε βαρύτητα στην κοντινότερη στον τερματισμό στάση, αφήνοντας την σταθερή και προχωράμε στα επόμενα στοιχεία του πίνακα **namesStart**, επαναλαμβάνοντας την ίδια διαδικασία. Αν δεν έχει βρεθεί «κοινό σημείο» τότε προχωράμε και τον πίνακα **namesEnd**.

Παράδειγμα:

namesStart={Άναυρος,.....}
namesEnd={Παύλου Μελά,.....}



Εικόνα 4.3 Παύλου Μελά



Εικόνα 4.3 Άναυρος

Για το 1^ο στοιχείο της «Παύλου Μελά», ψάχνουμε κάθε στοιχείο της «Αναύρου». Βρίσκουμε το «1» και το κρατάμε στην *elements*. Ψάχνουμε για το στοιχείο «5» κάθε γραμμή της 2^{ης}. Μετά για το στοιχείο «2» κ.τ.λ. Τελικώς συμπεραίνουμε ότι ο χρήστης μπορεί να επιβιβαστεί στην στάση «Άναυρος» και να κατεβεί στην στάση «Παύλου Μελά», εξυπηρετούμενος από τις Γραμμές 1 και 3.

Αν δεν βρίσκαμε κανένα κοινό στοιχείο, τότε θα συγκρίναμε το *namesEnd[0]* με το *namesStart[1]*. Αν πάλι δε βρίσκαμε, τότε με το *namesStart[2]* κ.ο.κ.

```
for (int x2 = 0; x2 < namesEnd.length; x2++) {
    for(int x1=0;x1<namesStart.length;x1++) {

        for (i = 0; i < Jarray.length(); i++) {
            JSONObject Jasonobject = null;
            Jasonobject = Jarray.getJSONObject(i);
            end = Jasonobject.getString(namesEnd[x2]);

            nEnd=namesEnd[x2];

            for (int j = 0; j < Jarray.length(); j++) {

                JSONObject Jasonobject2 = null;
                Jasonobject2 = Jarray.getJSONObject(j);
                start = Jasonobject2.getString(namesStart[x1]);
                nStart=namesStart[x1];

                if (start.equalsIgnoreCase(end) &&
                    !start.equalsIgnoreCase("") && !end.equalsIgnoreCase("")) {

                    elements.add(end);
                    temp=false;

                }
            }
        }
        if(temp==false){
            goToFinal1();
            break;
        }
    }
}
//if (start.equalsIgnoreCase(end)) {break}
}
if(temp==false){break;}
}
this.progressDialog.dismiss();
```

```

    } catch (Exception e) {
        // TODO: handle exception
        Log.e("log_tag", "Error parsing data "+e.toString());
    }
}

.....

private void goToFinal1(){

    elem=new String[elements.size()];

    for(int m=0;m<elements.size();m++){

        elem[m]=elements.get(m);
    }

    .....

    Intent intent = new Intent(this,Final1.class);
    Bundle t = new Bundle();
    t.putFloat("x_s",x1);
    t.putFloat("x_e",x2);
    t.putFloat("y_s",y1);
    t.putFloat("y_e",y2);
    t.putString("str_val",end);
    t.putString("str_start",nStart);
    t.putString("str_end",nEnd);
    t.putStringArray("namesStart",namesStart);
    t.putStringArray("namesEnd",namesEnd);
    t.putStringArray("elem", elem);
    intent.putExtras(t);
    startActivityForResult(intent, 0);}}

```


Κεφάλαιο 5

Συμπεράσματα

5.1 Αποτελέσματα

Με το τέλος της εργασίας, ο κύριος στόχος της δημιουργίας μιας εφαρμογής η οποία θα λειτουργεί ως οδηγός κίνησης στην πόλη του Βόλου, με ιδιαίτερη έμφαση στην Αστική Συγκοινωνία, έχει επιτευχθεί. Παρ' όλα αυτά αφέθηκαν για την ώρα στο περιθώριο άλλες ιδέες που είχαν να κάνουν

1. Με τον εμπλουτισμό με άλλα μέσα μεταφοράς, συγκεκριμένα ταξί.
2. Με οδηγό αναζήτησης για εστιατόρια, καφετέριες, ξενοδοχεία, αξιοθέατα και τη δυνατότητα των χρηστών να αξιολογούν τα παραπάνω.
3. Με έναν πληρέστερο οδηγό που θα συνδυάζει όλες τις κατηγορίες μεταξύ τους.

Το σημαντικότερο αποτέλεσμα που αποκόμισα από τη συγκεκριμένη εργασία, ήταν η επαφή με έναν άγνωστο ως τότε και άκρως ενδιαφέρον τελικά για μένα τομέα, αυτόν της ανάπτυξης εφαρμογών για κινητές συσκευές. Κατά τη διάρκεια μελετήθηκε σε βάθος το λειτουργικό σύστημα Android. Η ανάγκη επίσης χρησιμοποίησης server, βοήθησε στην απόκτηση γνώσεων του συστήματος διαχείρισης βάσεων δεδομένων MySQL και της γλώσσας php.

5.2 Δυσκολίες

Κατά τη διάρκεια της εργασίας οι κύριες δυσκολίες που προέκυψαν ήταν δύο. Πρώτον, η εξοικώωση με το συγκεκριμένο περιβάλλον προγραμματισμού, δυσκολία η οποία γρήγορα ξεπεράστηκε. Δεύτερον, όσον αφορά στο κομμάτι της υλοποίησης αυτό που αφιερώθηκε ο περισσότερος χρόνος μέχρι να λειτουργήσει σωστά, ήταν η κατηγορία «ΑΝΑΖΗΤΗΣΗ ΓΡΑΜΜΗΣ». Λόγω της έλλειψης πληροφοριών στο Google Maps όπως υπάρχει σε άλλες πόλεις, έπρεπε να βρεθούν οι κατάλληλοι αλγόριθμοι που θα λαμβάνουν υπόψιν τις

θέσεις που επιλέγει ο χρήστης, το σύνολο των στάσεων και των γραμμών που περνάνε από αυτές και τέλος την κατευθυντικότητα των γραμμών, καθώς έχουμε άλλες στάσεις στη μετάβαση και άλλες στην επιστροφή. Τελικώς όπως παρουσιάστηκε στο Κεφ.3 βρέθηκε ικανοποιητική λύση.

5.3 Μελλοντικές επεκτάσεις

Είναι σπάνιο μια εφαρμογή να μείνει στην αρχική της έκδοση. Συνήθως λόγω των νέων αναγκών που προκύπτουν μια εφαρμογή βελτιώνεται και επεκτείνεται. Αυτό αποτελεί στόχο και για την παρούσα.

Άμεσος στόχος λοιπόν της επόμενης έκδοσης είναι η βελτίωση κάποιων υπάρχοντων λειτουργιών.

- 1) Η προβολή των στάσεων στο χάρτη όταν ο χρήστης κάνει την επιλογή «Γραμμές και Στάσεις» από την κατηγορία «ΑΣΤΙΚΟ ΚΤΕΛ ΒΟΛΟΥ».
- 2) Στην κατηγορία «ΑΝΑΖΗΤΗΣΗ ΓΡΑΜΜΗΣ» πέρα από την υπάρχουσα δυνατότητα επιλογής δύο σημείων στο χάρτη, να δίνεται η δυνατότητα στον χρήστη να καθορίζει την αρχική του θέση με την χρήση GPS.
- 3) Στην κατηγορία «ΑΝΑΖΗΤΗΣΗ ΓΡΑΜΜΗΣ» μετά την παροχή των πληροφοριών για τις στάσεις επιβίβασης και αποβίβασης και για τη Γραμμή Εξυπηρέτησης, να δίνεται η δυνατότητα προβολής της διαδρομής και των συγκεκριμένων στάσεων στο χάρτη.
- 4) Στην κατηγορία «ΒΕΛΤΙΣΤΗ ΔΙΑΔΡΟΜΗ» η επιλογή «βάσει σημείων ενδιαφέροντος» να εμπλουτιστεί με πολλά περισσότερα σημεία.
- 5) Το πέραςμα των δεδομένων της κατηγορίας «ΑΣΤΙΚΟ ΚΤΕΛ ΒΟΛΟΥ» σε server.

Επόμενος στόχος είναι η υλοποίηση των ιδεών που αναφέρθηκαν αρχικά σε αυτό το κεφάλαιο. Δηλαδή,

- 1) Μια ξεχωριστή κατηγορία για τα ταξί. Ο χρήστης θα μπορεί να έχει πλήρη εικόνα για αυτά. Θα μπορεί σε αντιστοιχία με τα αστικά λεοφορεία να αναζητά τις κοντινότερες στη θέση του πιάτσες ταξί. Ενδεχομένως, σε συνεργασία με τα ταξί Βόλου θα μπορεί να καλεί μέσω της εφαρμογής ταξί.
- 2) Ένας πλήρης οδηγός αναψυχής και διαμονής στον Βόλο. Ο χρήστης θα μπορεί να αναζητά κοντινά ξενοδοχεία, εστιατόρια κτλ μέσω της θέσης

του. Θα μπορεί επίσης να εγγράφεται στην εφαρμογή και να αλληλεπιδρά με άλλους χρήστες μέσω της αξιολόγησης των παραπάνω.

- 3) Ο χρήστης να μπορεί να συνδυάσει όλα τα παραπάνω. Θα πρέπει να υπάρχει ένας πληρέστερος οδηγός που θα έχει ως βάση πάλι το που βρίσκεται και το που θέλει να πάει. Βάσει αυτών οι πληροφορίες που θα δέχεται θα είναι σαφέστερες. Για παράδειγμα, θα είναι του τύπου (*«Προχωρήστε στο Χ σημείο. Στρίψτε δεξιά κτλ. - Επιβιβαστείτε στη Γραμμή Νο Υ στην στάση Ζ. - Αποβιβαστείτε στο σημείο Κ και προχωρήστε στον τελικό σας προορισμό»*). Σε περίπτωση επίσης που δεν βολεύει η χρήση Λεοφωρείο να μπορεί να προτείνει εναλλακτικές (ταξί). Μέσω φίλτρων που θα υπάρχουν όπως για παράδειγμα ότι ένας χρήστης κατά τη διάρκεια της διαδρομής του ενδιαφέρεται για κάποιο εστιατόριο, θα δίνονται οι κατάλληλες πληροφορίες διαδρομής.

Αναφορές

[1] Αστικό Κτελ Βόλου

<http://astikovolou.gr/>

[2] ΟΑΣΘ Android App

<https://play.google.com/store/apps/details?id=oasth.mobile.transport>

[3] Android

<https://el.wikipedia.org/wiki/Android>

<http://www.allaboutandroid.gr/?p=6362>

[4] Java

http://www.islab.demokritos.gr/gr/html/ptixiakes/kostas-aris_ptyxiakh/Phtml/java.htm

[5] Xml

<https://el.wikipedia.org/wiki/XML>

<http://www.internetnow.gr/node/90>

[6] Eclipse Download

<https://eclipse.org/downloads/>

[7] Android Studio Download

<https://developer.android.com/sdk/index.html>

[8] XAMPP

<https://www.apachefriends.org/index.html>

[9] Apache Server

https://el.wikipedia.org/wiki/Apache_HTTP_%CE%B5%CE%BE%CF%85%CF%80%CE%B7%CF%81%CE%B5%CF%84%CE%B7%CF%84%CE%AE%CF%82

[10] MySql

<https://el.wikipedia.org/wiki/MySQL>

[11]Php

<https://el.wikipedia.org/wiki/PHP>

[12]StackOverflow

<http://stackoverflow.com>

[13]Intent

<http://developer.android.com/reference/android/content/Intent.html>

[14]ListView

<http://developer.android.com/guide/topics/ui/layout/listview.html>

[15]Array Adapter

<http://developer.android.com/reference/android/widget/ArrayAdapter.html>

[16]ExpandableListView

<http://developer.android.com/reference/android/widget/ExpandableListView.html>

[17]Google Maps Android API

<https://developers.google.com/maps/documentation/android/start>

[18] Interactive Polyline Encoder Utility

<https://developers.google.com/maps/documentation/utilities/polylineutility>

[19]Bundle

<http://developer.android.com/reference/android/os/Bundle.html>

Βιβλιογραφία

Βιβλία

[1] Walter Savitch, *Μια Εισαγωγή στην Επιλύση Προβλημάτων & στον Προγραμματισμό*, Εκδόσεις Τζιόλα, 4^η Έκδοση

[2] Jerome DiMarzio , *Android : A PROGRAMMER'S GUIDE*, McGrawHill, 2008

Tutorials

[1] Java Tutorial

<http://www.tutorialspoint.com/java/>

[2] Xml Tutorial

http://www.tutorialspoint.com/xml/xml_overview.htm

[3] MySQL Tutorial

<http://www.tutorialspoint.com/mysql/>

[4] PHP Tutorial

<http://www.tutorialspoint.com/php/>

[5] Vogella - Introduction to Android development with Android Studio

<http://www.vogella.com/tutorials/Android/article.html>

[6] RayWenderlich-Android Tutorial for Beginners

<http://www.raywenderlich.com/78574/android-tutorial-for-beginners-part-1>

ΠΑΡΑΡΤΗΜΑ

Η εργασία συνοδεύεται απο CD, το οποίο περιέχει το αρχείο apk προς εγκατάσταση όπως επίσης και όλες τις κλάσεις και τα αρχεία που αναπτύχθηκαν κατά την δημιουργία της εφαρμογής.

Τα αρχεία Java είναι:

1. **AstikoMainActivity.java:** η κλάση για το menu της κατηγορίας «ΑΣΤΙΚΟ ΚΤΕΛ ΒΟΛΟΥ».
2. **BusOnDemand.java:** η κλάση επιλογής σημείων στον χάρτη στην κατηγορία «ΑΝΑΖΗΤΗΣΗ ΓΡΑΜΜΗΣ».
3. **Dromologia.java:** η κλάση με την λίστα γραμμών εξυπηρέτησης.
4. **Ekdotirio2.java:** η κλάση προβολής του κοντινότερου εκδοτηρίου στον χάρτη.
5. **EkdotirioActivity.java:** η κλάση εύρεσης του κοντινότερου εκδοτηρίου εισητηρίων
6. **EkdotirioResultActivity.java:** η κλάση εμφάνισης του αποτελέσματος κοντινότερου εκδοτηρίου.
7. **ExpandableListMainActivity.java:** βοηθητική κλάση για την υλοποίηση ExpandableListView
8. **Final1.java :** η κλάση εμφάνισης των πληροφοριών στην κατηγορία «ΑΝΑΖΗΤΗΣΗ ΓΡΑΜΜΗΣ»
9. **FinalNew.java:** η κλάση εύρεσης της κατάλληλης γραμμής εξυπηρέτησης στην κατηγορία «ΑΝΑΖΗΤΗΣΗ ΓΡΑΜΜΗΣ»
10. **Grammi1.java:** μενού Γραμμής 1
11. **Grammi1a.java:** δρομολόγια Γραμμής 1-Μετάβαση
12. **Grammi1b.java:** δρομολόγια Γραμμής 1-Επιστροφή
13. **Grammi2.java:** μενού Γραμμής 2
14. **Grammi2a.java:** δρομολόγια Γραμμής 2-Μετάβαση
15. **Grammi2b.java:** δρομολόγια Γραμμής 2-Επιστροφή
16. **Grammi3.java:** μενού Γραμμής 3
17. **Grammi3a.java:** δρομολόγια Γραμμής 3-Μετάβαση
18. **Grammi3b.java:** δρομολόγια Γραμμής 3-Επιστροφή
19. **Grammi4a.java:** μενού Γραμμής 4 (1^η διαδρομή)
20. **Grammi4aa.java:** δρομολόγια Γραμμής 4-Μετάβαση(1^η διαδρομή)
21. **Grammi4ab.java:** δρομολόγια Γραμμής 4-Επιστροφή(1^η διαδρομή)
22. **Grammi4b.java:** μενού Γραμμής 4 (2^η διαδρομή)
23. **Grammi4ba.java:** δρομολόγια Γραμμής 4-Μετάβαση(2^η διαδρομή)

- 24. Grammi4bb.java:** δρομολόγια Γραμμής 4-Επιστροφή(2^η διαδρομή)
- 25. Grammi5.java:** μενού Γραμμής 5
- 26. Grammi5a.java:** δρομολόγια Γραμμής 5-Μετάβαση
- 27. Grammi5b.java:** δρομολόγια Γραμμής 5-Επιστροφή
- 28. Grammi6.java:** μενού Γραμμής 6
- 29. Grammi6a.java:** δρομολόγια Γραμμής 6-Μετάβαση
- 30. Grammi6b.java:** δρομολόγια Γραμμής 6-Επιστροφή
- 31. Grammi7.java:** μενού Γραμμής 7
- 32. Grammi7a.java:** δρομολόγια Γραμμής 7-Μετάβαση
- 33. Grammi7b.java:** δρομολόγια Γραμμής 7-Επιστροφή
- 34. Grammi8.java:** μενού Γραμμής 8
- 35. Grammi8a.java:** δρομολόγια Γραμμής 8-Μετάβαση
- 36. Grammi8b.java:** δρομολόγια Γραμμής 8-Επιστροφή
- 37. Grammi9.java:** μενού Γραμμής 9
- 38. Grammi9a.java:** δρομολόγια Γραμμής 9-Μετάβαση
- 39. Grammi9b.java:** δρομολόγια Γραμμής 9-Επιστροφή
- 40. Grammi10.java:** μενού Γραμμής 10
- 41. Grammi10a.java:** δρομολόγια Γραμμής 10-Μετάβαση
- 42. Grammi10b.java:** δρομολόγια Γραμμής 10-Επιστροφή
- 43. Grammi11.java:** μενού Γραμμής 11
- 44. Grammi11a.java:** δρομολόγια Γραμμής 11-Μετάβαση
- 45. Grammi11b.java:** δρομολόγια Γραμμής 11-Επιστροφή
- 46. Grammi15.java:** μενού Γραμμής 15
- 47. Grammi15a.java:** δρομολόγια Γραμμής 15-Μετάβαση
- 48. Grammi15b.java:** δρομολόγια Γραμμής 15-Επιστροφή
- 49. MainActivity.java:** η κλάση του αρχικού μενού
- 50. MyExpandableAdapter.java**
- 51. Staseis.java:** η κλάση με την λίστα των Γραμμών Εξυπηρέτησης για την εμφάνιση των στάσεων
- 52. Stasi1.java:** η κλάση με τις στάσεις της Γραμμής 1
- 53. Stasi2.java:** η κλάση με τις στάσεις της Γραμμής 2
- 54. Stasi3.java:** η κλάση με τις στάσεις της Γραμμής 3
- 55. Stasi4.java:** η κλάση με τις στάσεις της Γραμμής 4 (1^η διαδρομή)
- 56. Stasi4a.java:** η κλάση με τις στάσεις της Γραμμής 4 (2^η διαδρομή)
- 57. Stasi5.java:** η κλάση με τις στάσεις της Γραμμής 5
- 58. Stasi6.java:** η κλάση με τις στάσεις της Γραμμής 6
- 59. Stasi7.java:** η κλάση με τις στάσεις της Γραμμής 7
- 60. Stasi8.java:** η κλάση με τις στάσεις της Γραμμής 8
- 61. Stasi9.java:** η κλάση με τις στάσεις της Γραμμής 9
- 62. Stasi10.java:** η κλάση με τις στάσεις της Γραμμής 10

- 63. Stasi11.java:** η κλάση με τις στάσεις της Γραμμής 11
- 64. Stasi15.java:** η κλάση με τις στάσεις της Γραμμής 15
- 65. VeltistiDiadromi.java:** η κλάση για την εμφάνιση βέλτιστης διαδρομής μεταξύ σημείων ενδιαφέροντος
- 66. XartisAstiko.java:** η λίστα με τις Γραμμές για την μετέπειτα επιλογή χάρτη
- 67. XartisGrammes4a.java:** η κλάση προβολής του Χάρτη της Γραμμής Νο4 (1^η διαδρομή)
- 68. XartisGrammes6.java:** η κλάση προβολής του Χάρτη της Γραμμής Νο6
- 69. XartisGrammi1.java :** η κλάση προβολής του Χάρτη της Γραμμής Νο1
- 70. XartisGrammi2.java:** η κλάση προβολής του Χάρτη της Γραμμής Νο2
- 71. XartisGrammi3.java:** η κλάση προβολής του Χάρτη της Γραμμής Νο3
- 72. XartisGrammi4b.java:** η κλάση προβολής του Χάρτη της Γραμμής Νο4 (2^η διαδρομή)
- 73. XartisGrammi5.java:** η κλάση προβολής του Χάρτη της Γραμμής Νο5
- 74. XartisGrammi9.java:** η κλάση προβολής του Χάρτη της Γραμμής Νο9
- 75. XartisGrammi15.java:** η κλάση προβολής του Χάρτη της Γραμμής Νο15

Τα αρχεία php είναι:

- 1. BusOndemand.php:** Για την επικοινωνία με το table για την αναζήτηση γραμμής της βάσης δεδομένων.
- 2. demo.php:** Για την επικοινωνία με το table για τις πληροφορίες βέλτιστης διαδρομής της βάσης δεδομένων.

Για να εγκαταστήσετε το .apk στην συσκευή σας:

- 1.** Περάστε το αρχείο στο τηλέφωνο σας μέσω καλωδίου usb, κατά προτίμηση στην SD card.
- 2.** Αλλάξτε τις ρυθμίσεις της συσκευής. Στο «Settings» επιλέξτε «Application Settings». Ενεργοποιήστε το «Unknown Settings».
- 3.** Βρείτε το αρχείο σας μέσω κάποιου FileManager και εγκαταστήστε το.

