ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

Ανάπτυξη εφαρμογής υπολογισμού κόστους διαδρομής με ταξί σε περιβάλλον Google Android

Application development in Google Android environment for calculating taxi fare

Διπλωματική Εργασία

Μπέκος Δημήτριος

Υπεύθυνος καθηγητής : Αλκιβιάδης Γ. Ακρίτας
Καθηγητής ΠΘ

Επιβλέπων Καθηγήτρια: Γεώργιος Σταμούλης
Καθηγητής ΠΘ

Βόλος, 2015

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

Ανάπτυξη εφαρμογής υπολογισμού κόστους διαδρομής με ταξί σε περιβάλλον Google Android

Διπλωματική Εργασία

Μπέκος Δημήτριος

Υπεύθυνος καθηγητής : Αλκιβιάδης Γ. Ακρίτας
Καθηγητής ΠΘ

Επιβλέπων Καθηγήτρια: Γεώργιος Σταμούλης
Καθηγητής ΠΘ

Εγκρίθηκε από την Διμελή Εξεταστική επιτροπή τον Φεβρουάριο του 2015.

Υπογραφή Υπογραφή

(……) (……)

Ευχαριστίες

Αρχικά θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου κύριο Αλκιβιάδη Ακρίτα για την αμέριστη συμπαράσταση, υποστήριξη και εμπιστοσύνη που μου έδειξε κατά την διάρκεια εκπόνησης της διπλωματικής μου εργασίας.

Ευχαριστώ πολύ την καθηγήτρια Ασπασία Δασκαλοπούλου που δέχτηκε να αναλάβει την επίβλεψη της διπλωματικής εργασίας μου.

Τέλος, ένα μεγάλο ευχαριστώ στους φίλους μου οι οποίοι είναι δίπλα μου όλο αυτό τον καιρό καθώς και την οικογένεια μου της οποία η συμπαράσταση, ηθικά και οικονομικά, με βοήθησε σε όλη τη διάρκεια των σπουδών μου.

Δημήτρης Μπέκος

Βόλος, 2015

Περίληψη

Η παρούσα πτυχιακή εργασία με τίτλο “Ανάπτυξη εφαρμογής υπολογισμού κόστους διαδρομής με ταξί σε περιβάλλον Google Android" έχει ως στόχο την υλοποίηση μιας εφαρμογής σε περιβάλλον Google Android συμβατό με τις περισσότερες κινητές συσκευές.

Η δημιουργία της εφαρμογής αυτής έγινε με την χρήση του λογισμικού Eclipse, το οποίο είναι ένα ολοκληρωμένο περιβάλλον μέσα από το οποίο μπορούμε να γράψουμε και να εκτελέσουμε κώδικα. Είναι ελεύθερης διανομής (freeware) και ανοικτού κώδικα (open source).

Η εφαρμογή θα μπορεί να χρησιμοποιηθεί από όλους τους χρήστες κινητών με λειτουργικό Android .Ο σκοπός της εφαρμογής θα είναι ο κάθε χρήστης που θα θέλει να χρησιμοποιήσει ταξί για την μεταφορά του, να έχει την δυνατότητα να υπολογίσει με μια μικρή απόκλιση το κόστος της διαδρομής του.

Ο χρήστης θα εισάγει τον τόπο αφετηρίας της διαδρομής καθώς και τον προορισμό του και λαμβάνοντας υπόψιν και άλλους παράγοντες όπως για παράδειγμα την ύπαρξη αποσκευών ή την επιπλέον χρέωση κατά την νυχτερινή βάρδια, να υπολογίζει το κόστος της διαδρομής.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΥΡΕΤΗΡΙΟ ΕΙΚΟΝΩΝ

1 ΕΙΣΑΓΩΓΗ

1.1 Έξυπνα τηλέφωνα

Τα έξυπνα τηλέφωνα (smartphones) είναι κινητά τηλέφωνα τα οποία στηρίζουν την λειτουργία τους στο λειτουργικό σύστημα το οποίο έχουν, διαθέτοντας περισσότερη υπολογιστική ικανότητα και συνδεσιμότητα σε σχέση με ένα απλό κινητό τηλέφωνο. Τα πρώτα έξυπνα κινητά (PDA) συνδύαζαν τις λειτουργίες ενός προσωπικού ψηφιακού βοηθού και ενός κινητού τηλεφώνου. Αργότερα στις επόμενες έξυπνες συσκευές προστέθηκαν λειτουργίες, όπως αναπαραγωγή μουσικών κομματιών, ψηφιακή φωτογραφική μηχανή και με δυνατότητα λήψης βίντεο καθώς και δυνατότητα πλοήγησης GPS, πράγμα το οποίο τα κατέστησε πολυχρηστικές συσκευές.

Πλέον τα έξυπνα τηλέφωνα διαθέτουν όλα οθόνες αφής υψηλής ανάλυσης και διαδικτυακούς περιηγητές (web browsers) που εμφανίζουν τυποποιημένες ιστοσελίδες καθώς και βελτιστοποιημένες ιστοσελίδες για κινητά. Η πρόσβαση στο διαδίκτυο με την χρήση των τηλεφώνων αυτών είναι πιο εύκολη από ποτέ λόγω και της ανάπτυξης της Wi-Fi τεχνολογίας, δηλαδή τις ασύρματης πρόσβασης στο διαδίκτυο.

Εικόνα 1: Έξυπνα τηλέφωνα (Smartphones)

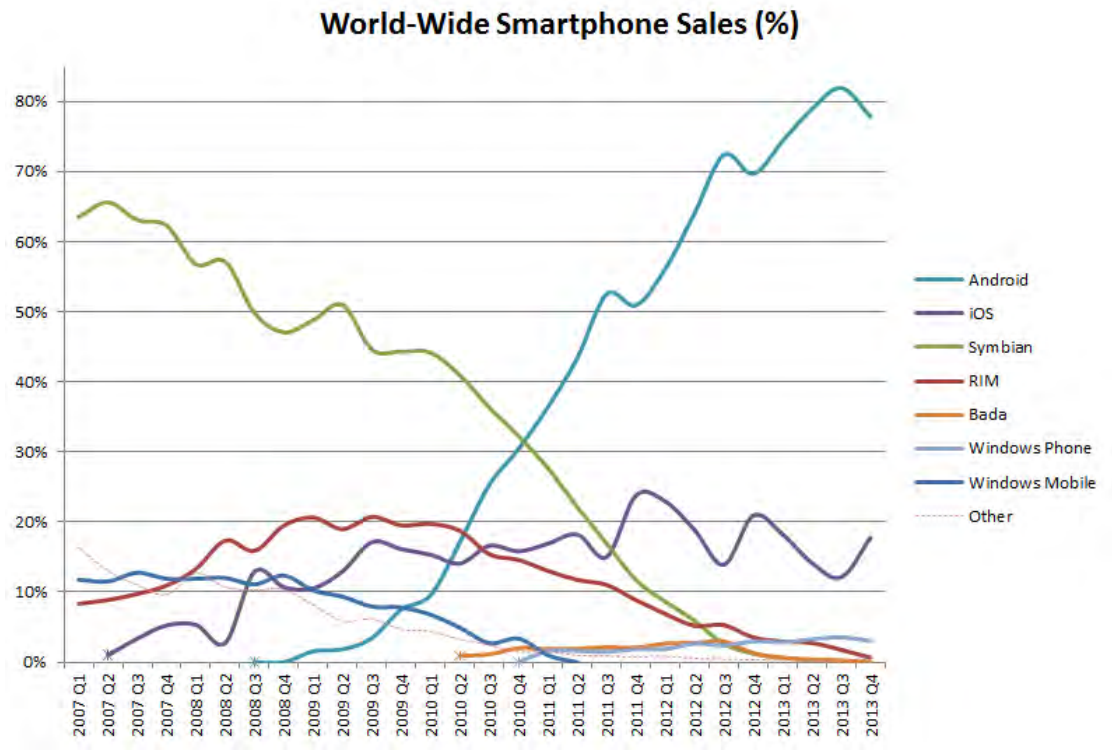
1.2 Λειτουργικό σύστημα

Λειτουργικό σύστημα ονομάζεται στην επιστήμη της πληροφορικής το λογισμικό του υπολογιστή που είναι υπεύθυνο για τη διαχείριση και τον συντονισμό των εργασιών, καθώς και την κατανομή των διαθέσιμων πόρων. Το λειτουργικό σύστημα παρέχει

ένα θεμέλιο, ένα μεσολαβητικό επίπεδο λογικής διασύνδεσης μεταξύ λογισμικού και υλικού, διαμέσου του οποίου οι εφαρμογές αντιλαμβάνονται εμμέσως τον υπολογιστή. Μια από τις κεντρικές αρμοδιότητες του λειτουργικού συστήματος είναι η διαχείριση του υλικού, απαλλάσσοντας έτσι το λογισμικό του χρήστη από τον άμεσο και επίπονο χειρισμό του υπολογιστή και καθιστώντας ευκολότερο τον προγραμματισμό τους.

1.2.1 Λειτουργικά συστήματα για έξυπνα τηλέφωνα

Υπάρχουν διάφορα λειτουργικά συστήματα διαθέσιμα για τους χρήστες ανάλογα με τις ανάγκες τους. Συγκεκριμένα, το Android της Google, το iOS της Apple, το Symbian της Nokia, το BlackBerry OS της RIM, η Bada της Samsung, το Windows Phone της Microsoft, το webOS της Hewlett-Packard, το Firefox της Mozilla, τα Maemo και MeeGo της Linux. Τα πιο διαδεδομένα βέβαια είναι αυτά της Google και της Apple, πράγμα που φαίνεται και στην Εικόνα 2, καθώς κατέχουν την μερίδα του λέοντος της παγκόσμιας αγοράς.



Εικόνα 2: Μερίδιο αγοράς λειτουργικών συστημάτων

1.3 Τι είναι το Android;

Το Android είναι ένα λειτουργικό σύστημα ανοιχτού κώδικα, βασισμένο στο Linux, για φορητές συσκευές όπως smartphones και tablets. Αναπτύχθηκε από την Google και αργότερα από την Open Handset Alliance η οποία είναι μια κοινοπραξία εταιριών λογισμικού, κατασκευής hardware και τηλεπικοινωνιών, οι οποίες είναι αφιερωμένες στην 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. Η Google δημοσίευσε το μεγαλύτερο μέρος του κώδικα του Android υπό τους όρους της Apache License, μιας ελεύθερης άδειας λογισμικού.

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 λειτουργίες της Android Inc, εκτός του ότι ανέπτυσσαν λογισμικό για κινητά τηλέφωνα. Αυτή ήταν η αρχή της φημολογίας περί σχεδίων της Google για να διεισδύσει στην αγορά κινητής τηλεφωνίας.

Στην Google, η ομάδα με επικεφαλής τον Andy Rubin ανέπτυξε μια κινητή πλατφόρμα που στηρίζεται στον πυρήνα του Linux, την οποία προώθησαν με την παροχή ενός ευέλικτου, αναβαθμίσιμου συστήματος. Έχει αναφερθεί ότι η Google είχε ήδη συγκεντρώσει μια σειρά από εταίρους hardware και software και επισήμανε στους παρόχους ότι ήταν ανοικτή σε διάφορους βαθμούς συνεργασίας εκ μέρους της. Έντυπα και ηλεκτρονικά μέσα ενημέρωσης σύντομα ανέφεραν φήμες ότι η Google ανέπτυσσε μια Google-branded συσκευή. Περισσότερες φήμες ακολούθησαν, αναφέροντας ότι η Google καθόριζε τις τεχνικές προδιαγραφές και έδειχνε πρωτότυπα στους κατασκευαστές κινητών τηλεφώνων και τους φορείς δικτύων. Τελικά η Google παρουσίασε το smartphone της, Nexus One που χρησιμοποιεί το open source λειτουργικό σύστημα Android. Η συσκευή κατασκευάστηκε από την HTC , και έγινε διαθέσιμη στις 5 Ιανουαρίου 2010.

Εικόνα 3: Google Nexus One

1.4 Εφαρμογές Android

Το Android έχει μια μεγάλη κοινότητα προγραμματιστών που γράφουν εφαρμογές, οι οποίες επεκτείνουν τη λειτουργικότητα των συσκευών. Οι εφαρμογές γράφονται σε μια προσαρμοσμένη έκδοση της JAVA και μπορεί κάνεις να κατεβάσει από το online κατάστημα Google Play (πρώην Android Market) της Google όπως και από άλλα sites. Μέχρι τον Ιούνιο του 2014 περισσότερες από 1,2 εκατομμύρια εφαρμογές ήταν διαθέσιμες για Android ενώ ο αριθμός των downloads από το Google Play μέχρι το Ιούλιο του 2013 είχε υπερβεί τα 50 δισεκατομμύρια.

Εικόνα 4: Google Play

1.5 Εξέλιξη του Android

Όπως αναφέραμε παραπάνω, το Android είναι ένα λειτουργικό σύστημα ανοιχτού κώδικα. Η εξέλιξη του, λόγω της open source φύσης του, είναι ραγδαία και αυτό αντικατοπτρίζεται στο γεγονός ότι οι 9 κύριες εκδόσεις του έχουν κυκλοφορήσει σε διάστημα 4.5 ετών, από τον Απρίλη του 2009 μέχρι τον Σεπτέμβριο του 2013.

Στην πληροφορική συνηθίζεται τα προϊόντα hardware και software να κυκλοφορούν εκτός από τον αριθμό έκδοσης τους, και με μία κωδική ονομασία. Η κωδική ονομασία μπορεί να είναι πχ ονόματα πόλεων (Windows Viena, Chicago), ονόματα ζώων (OSX Leopard, Lion), στην περίπτωση όμως του Android τα κώδικα ονόματα έρχονται στη μορφή επιδόρπιου!

1.5.1 Android 1.5 Cupcake (API Level: 3)

Παρουσιάστηκε στις 30/4/2009. Κάποια χαρακτηριστικά της έκδοσης αυτής είναι:

- Υποστήριξη με εικονικά πληκτρολόγια με πρόβλεψη κειμένου και λεξικού χρήστη για τις λέξεις
- Εγγραφή και αναπαραγωγή βίντεο
- Αυτόματη αντιστοίχιση και στερεοφωνική υποστήριξη για Bluetooth (A2DP και AVRCP προφίλ)
- Αντιγραφή και επικόλληση χαρακτηριστικών στο πρόγραμμα περιήγησης στο Web
- Κινούμενη μετάβαση στην οθόνη - Αυτόματη εναλλαγή επιλογής - Νέο animation boot απόθεμα - Δυνατότητα να ανεβάσετε βίντεο στο YouTube - Δυνατότητα να ανεβάσετε φωτογραφίες στο Picasa

1.5.2 Android 1.6 Donut (API Level: 4)

Παρουσιάστηκε στις 15/9/2009. Κάποια χαρακτηριστικά της έκδοσης αυτής είναι:

- η φωνή και το κείμενο αναζήτησης εισόδου ενισχύθηκαν και συμπεριέλαβαν σελιδοδείκτες, τις επαφές και το διαδίκτυο
- φωτογραφική μηχανή και βιντεοκάμερα έχουν ενσωματωθεί πλήρως με καλύτερες και πιο γρήγορες προσβάσεις

- Δυνατότητα στους χρήστες να επιλέξουν πολλαπλές φωτογραφίες για διαγραφή

1.5.3 Android 2.0/2.1 Eclair (API Level: 7)

Παρουσιάστηκε στις 26/10/2009. Κάποια χαρακτηριστικά της έκδοσης αυτής είναι:

- επιλογή στους χρήστες να προσθέτουν πολλαπλούς λογαριασμούς σε μία συσκευή για το συγχρονισμό των email και των επαφών τους
- Bluetooth 2.1 υποστήριξη
- Πολλά νέα χαρακτηριστικά κάμερας, συμπεριλαμβανομένης της υποστήριξης φλας, ψηφιακό zoom, λειτουργία σκηνής, ισορροπία λευκού, εφέ χρώματος και macro εστίαση
- Βελτιωμένη ταχύτητα πληκτρολόγησης σε εικονικό πληκτρολόγιο, με εξυπνότερο λεξικό που μαθαίνει από τη χρήση των λέξεων και περιλαμβάνει τα ονόματα των επαφών σαν προτάσεις
- Βελτιωμένο Google Maps 3.1.2 -Κατηγορία MotionEvent για να παρακολουθείτε multi-touch γεγονότα

Προσθήκη live wallpapers, επιτρέποντας την κίνηση στην εικόνα που έχουμε για φόντο στο κινητό

1.5.4 Android 2.2 Froyo (API Level: 8)

Παρουσιάστηκε στις 26/5/2010. Κάποια χαρακτηριστικά της έκδοσης αυτής είναι:

- καλύτερη ταχύτητα και μεγαλύτερη μνήμη
- Ένταξη του Chrome 's V8 κινητήρα JavaScript στην εφαρμογή περιήγησης
- USB tethering και Wi-Fi hotspot λειτουργικότητα
- Επιλογή για να απενεργοποιήσετε την πρόσβαση στα δεδομένα μέσω του δικτύου κινητής
- Adobe Flash υποστήριξη

1.5.5 Android 2.3 Gingerbread (API Level: 9)

Παρουσιάστηκε στις 6/12/2010, ενώ τον Φεβρουάριο του 2011 επανεκδόθηκε σε Android 2.3.3. Κάποια χαρακτηριστικά της έκδοσης αυτής είναι:

- υποστήριξη για πολύ μεγάλα μεγέθη οθόνης -υποστήριξη για SIP VoIP τηλεφωνία μέσω Internet
- Πιο γρήγορη εισαγωγή κειμένου στο εικονικό πληκτρολόγιο, με βελτιωμένη ακρίβεια, καλύτερο προτεινόμενο κείμενο και φωνητική λειτουργία εισόδου
- Ενισχυμένη copy / paste λειτουργικότητα
- υποστήριξη για Near Field Communication (NFC), που επιτρέπει στο χρήστη να διαβάσει μια ετικέτα NFC που είναι ενσωματωμένη σε μια αφίσα, αυτοκόλλητο, ή σε διαφήμιση
- δυνατότητα στους χρήστες για εύκολη πρόσβαση σε οποιοδήποτε αρχείο που έχει ληφθεί από τον browser, e-mail, ή σε άλλη εφαρμογή
- Υποστήριξη για πολλαπλές κάμερες στη συσκευή, συμπεριλαμβανομένων κάμερα στο μπροστινό μέρος
- υποστήριξη για περισσότερους αισθητήρες (όπως γυροσκόπια και βαρόμετρα)
- Βελτιωμένη απόδοση μπαταρίας - Υποστήριξη για συνομιλία με φωνή ή βίντεο χρησιμοποιώντας το Google

1.5.6 Android 3.0 Honeycomb

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. Υπάρχει ένα νέο, εντελώς διαφορετικό, User Interface και υποστηρίζονται διπύρηνοι και τετραπύρηνοι επεξεργαστές. Ακόμα, έχει απλοποιηθεί το multitasking έτσι ώστε ο χρήστης να μπορεί με τη χρήση ενός πλήκτρου (recent apps) να περνάει από μια εφαρμογή σε άλλη. Υπάρχει η δυνατότητα για Video Chat μέσω της εφαρμογής Google Talk καθώς η ανάγνωση βιβλίων μέσω του Google eBooks. Επιπλέον, μπορούν να κρυπτογραφηθούν όλα τα δεδομένα του χρήστη.

1.5.7 Android 4.0 Ice cream sandwich (API Level: 15)

Παρουσιάστηκε στις 19/10/2011. Κάποια χαρακτηριστικά της έκδοσης αυτής είναι:

- Ολοκληρωμένη σύλληψη screenshot (επιτυγχάνεται κρατώντας πατημένο το Power και Volume-Down κουμπιά)
- Βελτιωμένη διόρθωση σφαλμάτων στο πληκτρολόγιο
- Δυνατότητα πρόσβασης σε εφαρμογές απευθείας από την οθόνη κλειδώματος
- Η καλύτερη ενοποίηση φωνής και συνεχή, σε πραγματικό χρόνο ομιλίας για υπαγόρευση κειμένου
- Face Unlock, ένα χαρακτηριστικό που επιτρέπει στους χρήστες να ξεκλειδώσουν συσκευές που χρησιμοποιούν λογισμικό αναγνώρισης προσώπου
- Νέα καρτέλες πρόγραμμα περιήγησης στο Web με το Google Chrome , επιτρέποντας έως και 16 καρτέλες
- Δυνατότητα να κλείσει εφαρμογές που χρησιμοποιούν δεδομένα στο παρασκήνιο
- Βελτιωμένη app κάμερα με μηδενική υστέρηση κλείστρου, ρυθμίσεις πάροδο του χρόνου, panorama mode
- ενσωματωμένο πρόγραμμα επεξεργασίας φωτογραφιών -1080p εγγραφή βίντεο

1.5.8 Android 4.1 Jelly Bean (API Level: 16)

Παρουσιάστηκε στις 9/7/2012. Κάποια χαρακτηριστικά της έκδοσης αυτής είναι:

- Ανανεωμένο σύστημα ειδοποιήσεων
- δυνατότητα χρήσης εξωτερικής συσκευής USB ήχου
- Βελτιωμένη φωνητική αναζήτηση
- Δυνατότητα χρήσης της υπηρεσίας Google Wallet
- OpenGL ES 3.0 υποστήριξη, που βελτιώνει τα γραφικά ενός παιχνιδιού

1.5.9 Android 4.4 Kit Kat (API Level: 19)

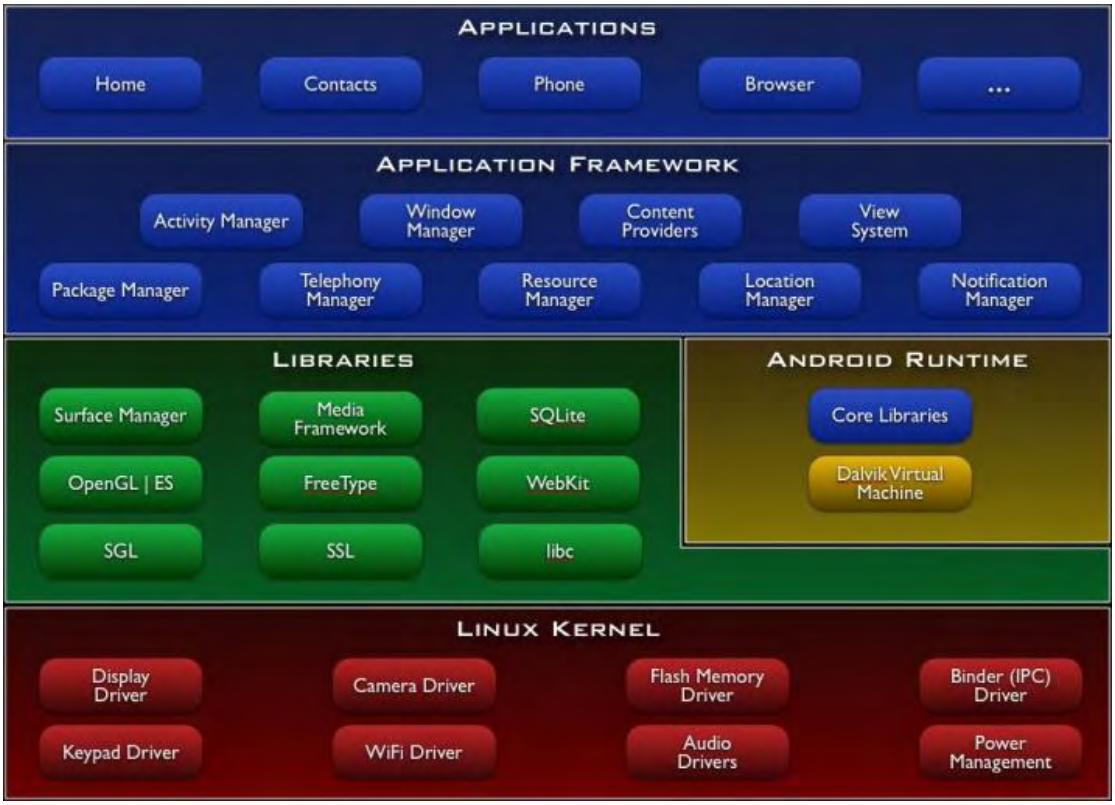
Παρουσιάστηκε στις 31/10/2013. Κάποια χαρακτηριστικά της έκδοσης αυτής είναι:

- Αρκεί να πεις "Ok Google" για να ξεκινήσεις τη φωνητική λειτουργία
- Κατά την αναπαραγωγή μουσικής ή videos, στο κλείδωμα οθόνης εμφανίζει κάποιο είδος "τέχνης"
- Γρηγορότερο multitasking
- Με το Hangouts τοποθετούνται όλα τα μηνύματα σε ένα μέρος
- Υποστηρίζεται η ασύρματη εκτύπωση για τους εκτυπωτές που είναι συνδεδεμένοι στο Google Cloud Print

Εικόνα 5: Χρονοδιάγραμμα εκδόσεων του Android OS

2 Αρχιτεκτονική Android

Το Android δεν είναι μόνο ένα λειτουργικό σύστημα. Είναι μια στοίβα λογισμικού η οποία αποτελείτε από το λειτουργικό σύστημα, τις υπηρεσίες διασύνδεσης με τις εφαρμογές (middleware) και τέλος από τις κύριες (core) εφαρμογές, μεταξύ αυτών, ενός email cient, μιας εφαρμογής διαχείρισης SMS, ενός ημερολογίου, ενός browser, εφαρμογή διαχείρισης επαφών, και άλλες οι οποίες έρχονται δεμένες με την υπόλοιπη στοιβάδα λογισμικού του Android.



Εικόνα 6: Η αρχιτεκτονική του Android

Από ότι βλέπουμε λοιπόν η αρχιτεκτονική του λειτουργικού συστήματος αποτελείται από 5 βασικά επίπεδα.

- Τον πυρήνα Linux (Linux Kernel)
- Τις εγγενείς και τις προηγμένες βιβλιοθήκες (Libraries)
- Την εικονική μηχανή Dalvik (Dalvik VM)
- Τον χρόνο εκτέλεσης (Android Runtime)
- Το πλαίσιο εφαρμογής (Application Framework)

2.1 Πυρήνας Linux (Linux Kernel)

Η βάση της στοίβας λογισμικού του Android είναι ο πυρήνας Linux. Ο τροποποιημένος πυρήνας του συστήματος βασίζεται στην έκδοση 2.6 (και στην έκδοση 3.0.1 για το Android 4.0) του Linux Kernel, η οποία υποστηρίζει όλες τις κύριες λειτουργίες του λειτουργικού συστήματος. Οι λειτουργίες αυτές αφορούν διαχείριση μνήμης, διαχείριση διεργασιών, λειτουργίες δικτύου, ασφάλεια του λειτουργικού, και ένα σύνολο οδηγών υλικού (hardware drivers). Οι οδηγοί αυτοί είναι υπεύθυνοι για την επικοινωνία του software με το hardware της συσκευής. Ενδεικτικά ο πυρήνας του Android περιέχει:

- Οδηγό προβολής οθόνης
- Οδηγό Wifi και Bluetooth
- Οδηγό κάμερας κλπ

Ο πυρήνας του Android μπορεί να βασίζεται στον πυρήνα του Linux, αλλά διαφέρει αρκετά από αυτόν. Ο λόγος είναι οι αλλαγές στην αρχιτεκτονική που έχει κάνει η Google για να είναι ελαφρύτερος και βελτιστοποιημένος για χρήση σε κινητές συσκευές. Αυτό σημαίνει ότι παρότι το Android είναι κατά βάση Linux, επί της ουσίας είναι αρκετά δύσκολο να τρέξουν εφαρμογές ή να χρησιμοποιηθούν βιβλιοθήκες από τη μία πλατφόρμα στην άλλη. Ο Linus Torvalds έχει αναφέρει ότι τελικά στο μέλλον το Android και το Linux θα μοιράζονται έναν κοινό πυρήνα.

2.2 Βιβλιοθήκες

Στο δεύτερο επίπεδο της στοίβας έχουμε τις βιβλιοθήκες του Android. Αυτές ουσιαστικά αποτελούν τα APIs που είναι διαθέσιμα στους προγραμματιστές για την ανάπτυξη των εφαρμογών. Οι βιβλιοθήκες από μόνες τους δεν αποτελούν εφαρμογές αλλά ενσωματώνονται και χρησιμοποιούνται από τις εφαρμογές για τις διάφορες λειτουργίες που παρέχει η καθεμία από αυτές. Ουσιαστικά αποτελούν ένα από τα δομικά υλικά των εφαρμογών, και άρα είναι αναπόσπαστο κομμάτι τους. Οι δυνατότητες των βιβλιοθηκών του Android γίνονται εμφανείς στους προγραμματιστές στην στοίβα του πλαισίου εφαρμογής.

Το σύνολο σχεδόν των βιβλιοθηκών είναι γραμμένο σε C και C++, οι οποίες έχουν μεταγλωττιστεί για τη χρήση τους από το λειτουργικό. Μερικές από τις κύριες βιβλιοθήκες του Android είναι:

- **System C library** – μια ενσωμάτωση της standard βιβλιοθήκης συστήματος της C (libc) τροποποιημένη για κινητές συσκευές βασισμένες στο Linux.
- **Βιβλιοθήκες Πολυμέσων** – Υποστηρίζει αναπαραγωγή και εγγραφή πολλών δημοφιλών μέσων ήχου και εικόνας, όπως: MPEG4, H.264, MP3, AAC, AMR, JPG και PNG
- **Surface Manager** – διαχειρίζεται την πρόσβαση στο υποσύστημα προβολής, και συνθέτει απρόσκοπτα δισδιάστατα και τρισδιάστατα επίπεδα γραφικών τα οποία προέρχονται από πολλαπλές εφαρμογές.
- **LibWebCore** – μια μοντέρνα μηχανή υποστήριξης πλοήγηση στο διαδίκτυο (browser engine) η οποία χρησιμοποιείτε και από τον ενσωματωμένο browser του Android αλλά και από τις WebViews που ενσωματώνονται στις εφαρμογές.
- **SGL** – η γνωστή μηχανή δισδιάστατων γραφικών
- **Βιβλιοθήκες 3D** – μια υλοποίηση βασισμένη στα APIs του OpenGL ES 1. Οι βιβλιοθήκες χρησιμοποιούν είτε τρισδιάστατη επιτάχυνση υλικού, όπου αυτή είναι διαθέσιμη, είτε μια υψηλά βελτιωμένη τρισδιάστατη επιτάχυνση λογισμικού σε περίπτωση που η πρώτη δεν είναι διαθέσιμη.
- **FreeType** – παρέχει ευκρίνεια γραφικών στα bitmaps και τις γραμματοσειρές των εφαρμογών του συστήματος.
- **SQLite** – μια πανίσχυρη και συνάμα πολύ ελαφριά σχεσιακή βάση δεδομένων

2.3 Η εικονική μηχανή Dalvik

Σχεδόν το σύνολο των APIs του Android βασίζονται στη γλώσσα προγραμματισμού Java. Στην Java ως γνωστόν υπάρχει η λεγόμενη Java Virtual Machine στην οποία εκτελείτε ο κώδικας bytecode των εφαρμογών. Στο Android υπάρχει κάτι παρόμοιο και δεν είναι άλλο από την εικονική μηχανή Dalvik.

Η Dalvik λοιπόν είναι η εικονική μηχανή μέσω της οποίας τρέχουν οι εφαρμογές του Android. Η κάθε εφαρμογή τρέχει μέσω τις δικής της εικονικής μηχανής στη δικιά

της διεργασία και για αυτό το λόγο καμία εφαρμογή δεν έχει επαφή με την άλλη, ενώ εκτελούνται ταυτόχρονα. Η Dalvik δεν υποστηρίζει τον κώδικα bytecode, αντί αυτού οι κλάσεις της Java γίνονται compile σε αρχεία .dex ώστε να τρέξουν στην VM. Τα αρχεία dex ουσιαστικά αποτελούν συμπιεσμένα δεδομένα για εξοικονόμηση χώρου κατά την εκτέλεση.

Το Android είναι από τη φύση του multitasking λειτουργικό σύστημα και για αυτό επιτρέπει στις εφαρμογές του να τρέχουν σε πολλά νήματα ταυτόχρονα και να απασχολούν πολλές διαδικασίες εάν αυτό είναι αναγκαίο. Για να γίνει αυτό εφικτό η μηχανή Dalvik είναι σχεδιασμένη για να έχει ελάχιστο αντίκτυπο στη χρήση της μνήμης. Χάρη στον λιτό της σχεδιασμό, το σύστημα είναι σε θέση να τρέχει πολλές εικονικές μηχανές ταυτόχρονα.

2.4 Χρόνος Εκτέλεσης Εφαρμογής (Android Runtime)

Ο χρόνος εκτέλεσης των εφαρμογών του Android, βρίσκεται στο ίδιο επίπεδο με τις κύριες βιβλιοθήκες και την μηχανή Dalvik. Εδώ βρίσκουμε το κοινό σημείο επαφής μεταξύ των δυνατοτήτων που παρέχουν οι βιβλιοθήκες και του χρόνου εκτέλεσης της εικονικής μηχανής Dalvik τις λειτουργίες τις οποίας, περιγράψαμε παραπάνω.

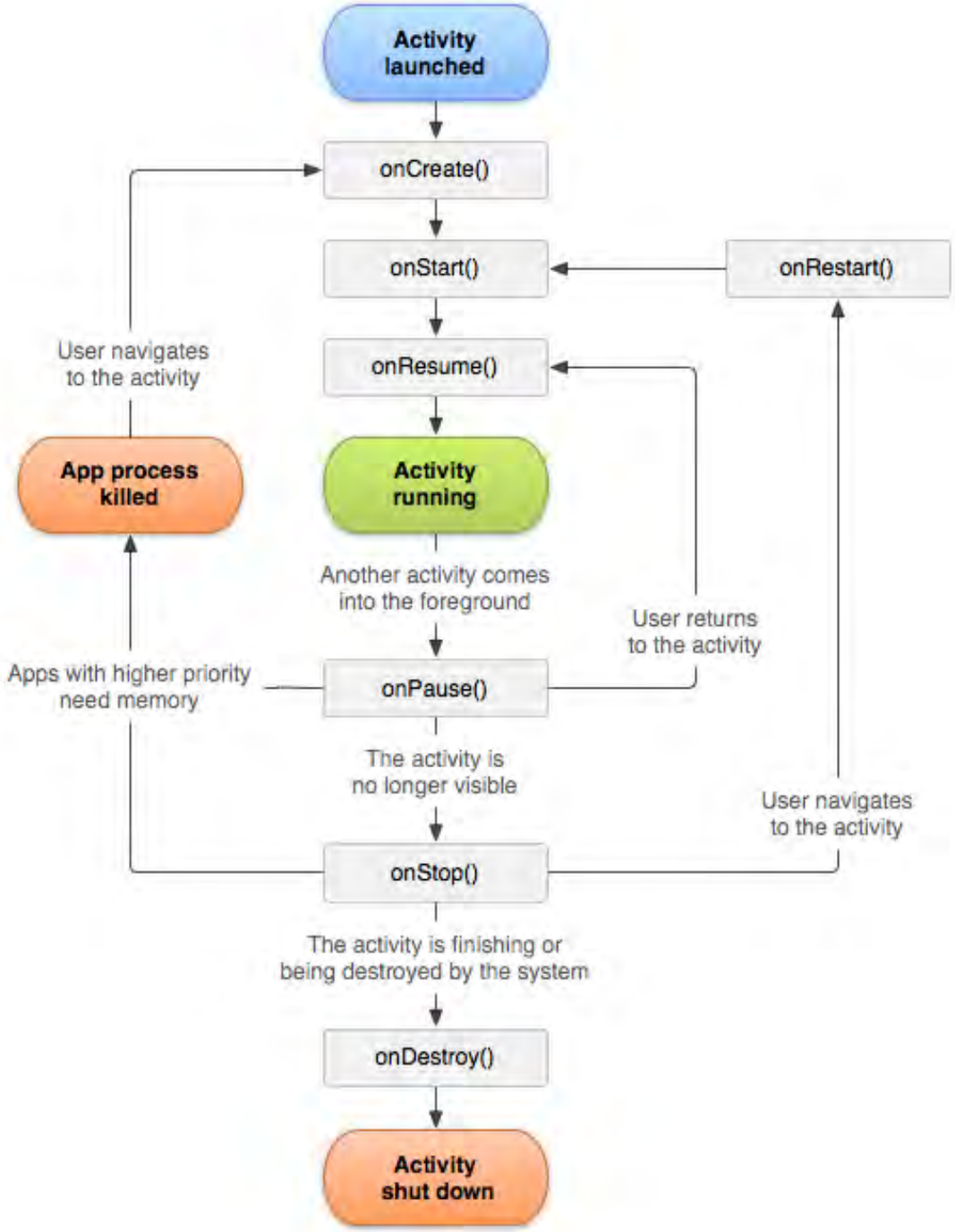
2.5 Πλαίσιο Εφαρμογής (Application Framework)

Το Android παρέχει στους developers μια ανοιχτού κώδικα πλατφόρμα ανάπτυξης και τη δυνατότητα να αναπτύξουν με αυτή ιδιαίτερα καινοτόμες και πλούσιες σε υλικό, εφαρμογές. Οι developers έχουν στην 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 που είναι διαθέσιμα.

Στο επόμενο επίπεδο της αρχιτεκτονικής του Android λοιπόν, συναντάμε το πλαίσιο των εφαρμογών. 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 τις μίας εφαρμογής από άλλες εφαρμογές, βέβαια κάτω από τις προδιαγραφές ασφάλειας του Android. Τα σημαντικότερα δομικά στοιχεία του πλαισίου εφαρμογών είναι:

- **Σύστημα προβολών (View System)** – αποτελεί ένα εκτενές σύνολο από αντικείμενα GUI τα οποία μπορούν να χρησιμοποιηθούν κατά το σχεδιασμό μιας εφαρμογής. Παραδείγματα προβολών είναι οι λίστες (listView), το πλέγμα (GridView), πεδία εισαγωγής κειμένου, κουμπιά, κλπ
- **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.
- **Διαχειριστής Πόρων (Resource Manager)** – παρέχει πρόσβαση σε υλικό το οποίο δεν είναι σε μορφή κώδικα όπως πχ, εικόνες, αρχεία xml, πίνακες χαρακτήρων, κλπ
- **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
- **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. Στο σχεδιάγραμμα (Εικόνα 7) φαίνεται λεπτομερώς ο κύκλος ζωής κάθε δραστηριότητας.



Εικόνα 7: Κύκλος ζωής μιας δραστηριότητας (Activity lifecycle)

3 Εργαλεία ανάπτυξης μιας εφαρμογής Android

Η γλώσσα προγραμματισμού που χρησιμοποιείται για την κατασκευή εφαρμογών Android είναι η Java, ενώ γίνονται προσπάθειες να συμπεριληφθούν και άλλες γλώσσες όπως η c και c++.

Οπότε βασική προϋπόθεση της κατασκευής είναι να διαθέτουμε τα αντίστοιχα εργαλεία της γλώσσας προγραμματισμού που θα χρησιμοποιήσουμε και συγκεκριμένα το Java Development Kit (JDK). Ακόμη χρειαζόμαστε ένα ολοκληρωμένο περιβάλλον ανάπτυξης (integrated development environment) για να μεταγλωττίζουμε και να τρέχουμε τα προγράμματα μας όπως το Eclipse.

Το βασικότερο εργαλείο αποτελεί το Android software development kit (SDK) το οποίο μας παρέχει τα επιπλέον εργαλεία ώστε να μπορούμε να γράψουμε κώδικα για την κατασκευή εφαρμογής Android. Η σύνδεση του android SDK με το γραφικό μας περιβάλλον γίνεται μέσω μιας επέκτασης (Android Development Tools ή ADT Plugin) που εγκαθιστάμε στο Eclipse, ώστε να μπορέσουμε να μεταγλωττίσουμε την εφαρμογή μας και έπειτα να την τρέξουμε.

Καθώς το λειτουργικό σύστημα Android κυκλοφορεί σε διάφορες εκδόσεις, καθίσταται σαφές ότι η κάθε έκδοση θα χρησιμοποιεί και ορισμένα διαφορετικά προγραμματιστικά εργαλεία. Έτσι μέσα από την ADT Plugin μπορούμε να εγκαταστήσουμε τα εργαλεία για την υλοποίηση της εφαρμογής σε οποιαδήποτε έκδοση. Για παράδειγμα αν θέλουμε η εφαρμογή μας να είναι συμβατή και σε παλαιότερες εκδόσεις του Android, τότε θα πρέπει να εγκαταστήσουμε τα αντίστοιχα εργαλεία και να δοκιμάσουμε την εφαρμογή μας στις εκδόσεις αυτές.

Τέλος, για να δοκιμάσουμε την εφαρμογή μας και να δούμε τα αποτελέσματα της θα χρειαστούμε κάποιον προσομοιωτή κινητού τηλεφώνου στον υπολογιστή μας. Την λύση μας δίνει μια εικονική συσκευή Android (Android Virtual Device ή AVD) η οποία ουσιαστικά αποτελεί προσομοιωτή τόσο software όσο και hardware ενός κινητού τηλεφώνου με λειτουργικό σύστημα Android. Την συσκευή αυτή, την εγκαθιστάμε μέσα από την ADT Plugin, από όπου μπορούμε και να ρυθμίσουμε πολλές παραμέτρους της, όπως την έκδοση του Android που θα χρησιμοποιεί, το μέγεθος της οθόνης, το μέγεθος της κάρτας SD και της cache, καθώς και άλλα χαρακτηριστικά όπως δυνατότητα λήψης φωτογραφιών.

Απαιτούνται βέβαια αρκετοί υπολογιστικοί πόροι για την εκτέλεση της AVD, πράγμα που καθιστά τις περισσότερες φορές αργή την εκτέλεση της εφαρμογής μας στη συσκευή αυτή. Φυσικά σε κάθε περίπτωση μπορούμε να εγκαθιστάμε και να εκτελούμε τις εφαρμογές μας κατ' ευθείαν σε φυσική συσκευή όπως κινητό τηλέφωνο που χρησιμοποιεί το λειτουργικό σύστημα Android.

3.1 Android SDK

Android SDK σημαίνει "Android Software Development Kit" και είναι το επίσημο εργαλείο της Google για αυτούς που θέλουν να δημιουργήσουν στο Android.

Το πρώτο βήμα στην πορεία ανάπτυξης της εφαρμογής είναι η εγκατάσταση και ρύθμιση του Android SDK. Το Android SDK περιλαμβάνει μια μεγάλη λίστα με εργαλεία ανάπτυξης. Σε αυτά περιλαμβάνονται:

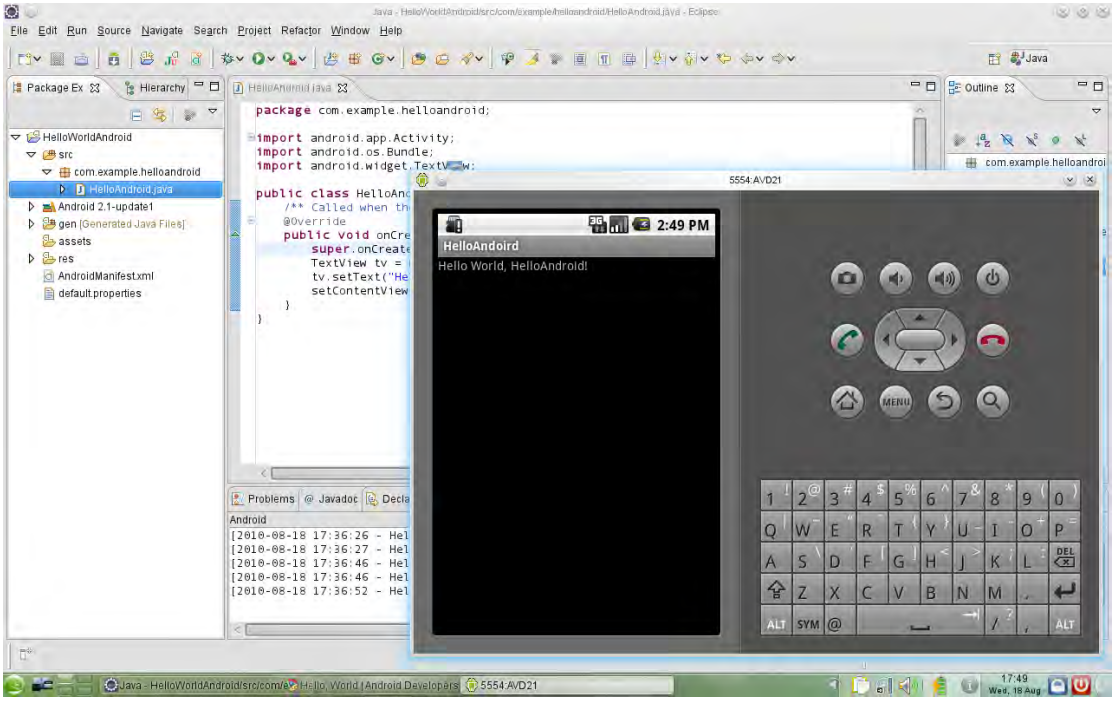
- Εργαλεία Debugging των εφαρμογών
- Βιβλιοθήκες
- Εξομοιωτής συσκευών (Android Virtual Machines)
- Documentation
- Δείγματα Κώδικα
- Tutorial

Μέσω του SDK μπορούμε να χρησιμοποιήσουμε εργαλεία όπως το ADB, για να μεταφέρουμε αρχεία σε χώρους που κανονικά δεν επιτρέπεται και το fastboot για να εγκαθιστούμε custom recovery εικόνες και να ξεκλειδώνουμε τον bootloader της συσκευής μας, κυρίως όμως μπορούμε να κατασκευάσουμε προγράμματα για το Android αρκεί να έχουμε γνώσεις Java προγραμματισμού για να το καταφέρουμε με επιτυχία.

Περιλαμβάνει παραδείγματα εφαρμογών με τον πηγαίο τους κώδικα, εργαλεία ανάπτυξης, ένα εξομοιωτή και τις απαιτούμενες βιβλιοθήκες για την ανάπτυξη των εφαρμογών στη γλώσσα προγραμματισμού Java. Ακόμη αναλαμβάνει τη μεταγλώττιση του πηγαίου κώδικα ώστε να τρέχει στην εικονική μηχανή Dalvik.

Το Android SDK περιλαμβάνει μια κινητή συσκευή emulator - μια εικονική φορητή συσκευή που τρέχει στον υπολογιστή σας. Ο εξομοιωτής επιτρέπει να αναπτύξουν και να δοκιμάσουν Android εφαρμογές χωρίς τη χρήση μιας φυσικής συσκευής. Ο εξομοιωτής του Android χρησιμοποιεί τα Android Virtual Device (AVD). Τα AVD επιτρέπουν τον ορισμό διαφόρων πτυχών του υλικού του μιμούμενου τηλεφώνου και επιτρέπουν τη δημιουργία πολλών συνθέσεων ώστε να υπάρχει ποικιλία παραλλαγών υλικού διασφαλίζοντας την ομαλή λειτουργία της εφαρμογής σε περισσότερες συσκευές Android.

Όταν η εφαρμογή λειτουργεί με τον εξομοιωτή, μπορεί να χρησιμοποιήσει τις υπηρεσίες της πλατφόρμας Android για να επικαλεστεί άλλες εφαρμογές, να έχει πρόσβαση στο δίκτυο, να αναπαράγει ήχο και βίντεο, να αποθηκεύει και να ανακτά δεδομένα, να ενημερώνει το χρήστη και άλλα. Ο εξομοιωτής περιλαμβάνει επίσης μια ποικιλία από δυνατότητες εντοπισμού σφαλμάτων, όπως μια κονσόλα στην οποία καταγράφονται τα μηνύματα εξόδου του πυρήνα, προσομοίωση τυχόν διακοπών της εφαρμογής (όπως όταν φθάνουν μηνύματα SMS ή τηλεφωνικές κλήσεις), καθώς και απορρίψεις στο δίκτυο δεδομένων.



Εικόνα 8: Eclipse and Android SDK

3.2 Android NDK

Το NDK είναι ένα εργαλείο που επιτρέπει τη μεταγλώττιση κάποιων μερών μίας android εφαρμογής χρησιμοποιώντας native γλώσσες προγραμματισμού όπως η C και η C++.

Για ορισμένους τύπους εφαρμογών, αυτό μπορεί να είναι χρήσιμο, καθώς με αυτόν τον τρόπο μπορούν να επαναχρησιμοποιηθούν ορισμένες υπάρχουσες βιβλιοθήκες που ενδέχεται να έχουν αυξημένη απόδοση.

Η χρήση του NDK, ως επί των πλείστον, δεν θα ωφελήσει τις περισσότερες εφαρμογές. Για σωστό αποτέλεσμα, ένας προγραμματιστής θα πρέπει να εξισορροπήσει τα οφέλη και τα μειονεκτήματά αυτής της χρήσης. Αξίζει να σημειωθεί ότι, η χρήση native κώδικα στις εφαρμογές Android, τις περισσότερες φορές δεν οδηγεί σε αισθητή βελτίωση των επιδόσεων, αλλά αυξάνει την πολυπλοκότητα της εφαρμογής. Σε γενικές γραμμές, θα πρέπει η χρήση του NDK να γίνεται μόνο επειδή ταιριάζει στη φύση της εφαρμογής που αναπτύσσεται και όχι απλά επειδή ο προγραμματιστής της εν λόγω εφαρμογής, προτιμά γλώσσες σαν τη C ή τη C++.

Τυπικά, εφαρμογές που αναπτύσσονται καλύτερα με χρήση NDK, είναι εφαρμογές που είναι αυτόνομες και που δε χρησιμοποιούν πολλή μνήμη. Κάποια τέτοια παραδείγματα είναι η επεξεργασία σήματος και η προσομοίωση της φυσικής. Κατά την εξέταση του κατά πόσο θα πρέπει ή όχι να αναπτυχθεί η εφαρμογή σε native κώδικα, πρέπει να ληφθούν σοβαρά υπόψη οι απαιτήσεις της καθώς επίσης και το αν τα Android APIs, που θα χρησιμοποιηθούν, παρέχουν τη λειτουργικότητα που χρειάζεται.

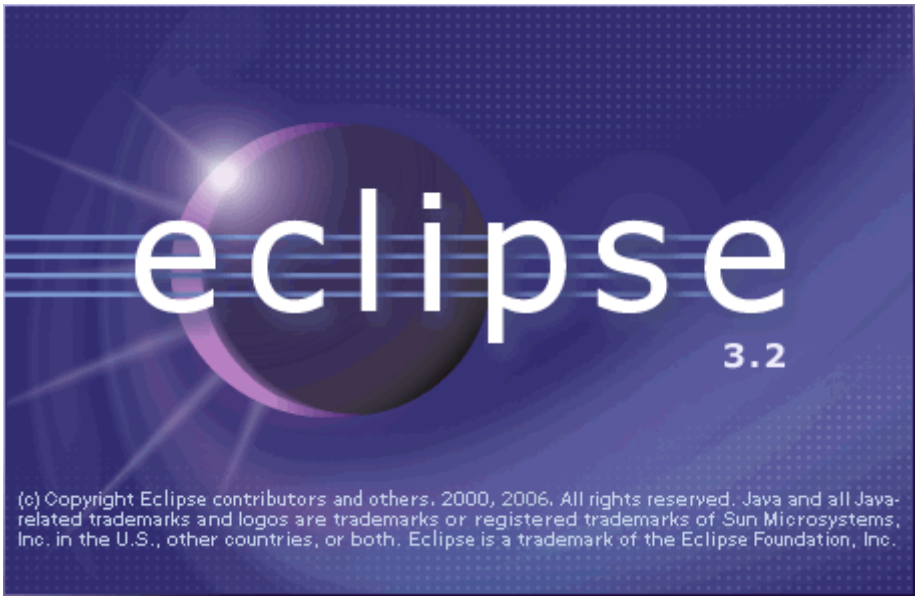
3.3 Eclipse

Οι περισσότερες εφαρμογές στο Android βασίζονται στην γλώσσα προγραμματισμού Java. Έτσι, ο κάθε προγραμματιστής μπορεί να χρησιμοποιήσει έναν οποιονδήποτε text editor για να γράψει τον κώδικα και μετέπειτα να μεταγλωττίσει τα αρχεία μέσω γραμμής εντολών.

Ο συγκεκριμένος τρόπος ανάπτυξης δεν είναι ιδιαίτερα φιλικός προς το χρήστη γι' αυτό συνίσταται η χρήση ενός IDE (Integrated Development Environment) που να υποστηρίζει Java, όπως το Eclipse ή το Netbeans.

Το Eclipse είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης λογισμικού με δυνατότητα εύκολης επεκτασιμότητας χάρη στο σύστημα αρθρωμάτων που διαθέτει. Είναι γραμμένο κυρίως στη γλώσσα προγραμματισμού Java και μπορεί να χρησιμοποιηθεί για την ανάπτυξη εφαρμογών σε Java και μέσω των διαφόρων αρθρωμάτων μπορεί να υποστηρίξει και επιπλέον γλώσσες προγραμματισμού όπως C, C++, Perl, PHP, Python, Ruby και άλλες.

Το Eclipse είναι μια εφαρμογή Ελεύθερου Λογισμικού καθώς διατίθεται κάτω από την άδεια Eclipse Public License. Αναπτύσσεται ραγδαία από την παγκόσμια κοινότητα του Ελεύθερου Λογισμικού ενώ χρηματοδοτείται από το ίδρυμα Eclipse και από διάφορες δωρεές.



Εικόνα 9: Eclipse IDE

3.4 Android Development Tools (ADT)

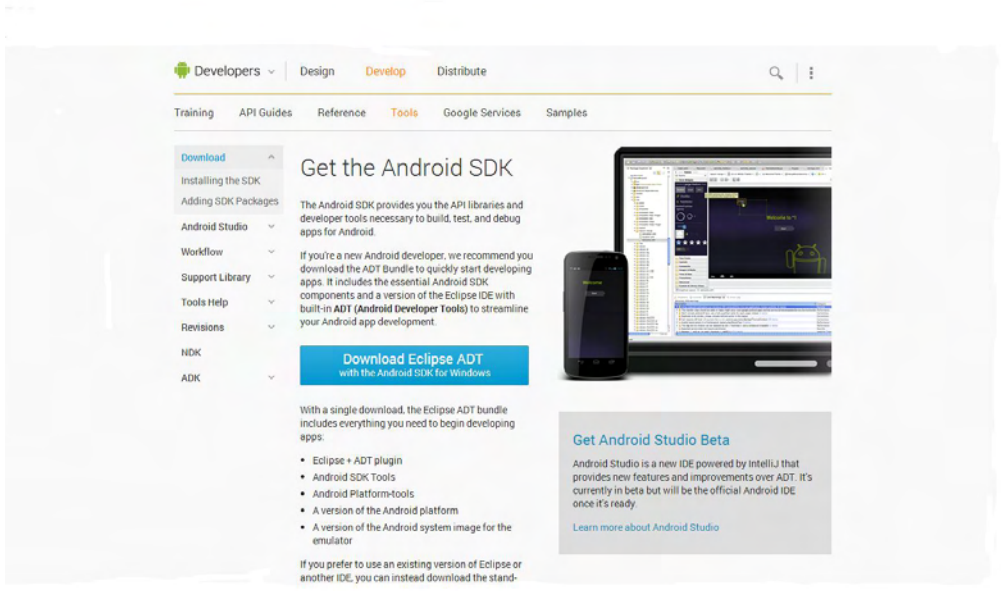
Η Google υποστηρίζει επίσημα το Eclipse και έχει αναπτύξει ειδικά για αυτό ένα επιπρόσθετο άρθρωμα, το Android Development Tools, το οποίο συνδέει το Eclipse με το Android SDK και όλες τις δυνατότητες του. Επίσης το άρθρωμα παρέχει

σύνδεση με τον διαχειριστή εικονικών συσκευών του Android SDK, για τη διαχείριση και εκκίνηση των εικονικών συσκευών μέσω γραφικής διεπαφής. Έτσι διευκολύνει τις δοκιμές σε διάφορους τύπους συσκευών καθώς και στην αποσφαλμάτωση τους.

Το ADT είναι ένα plug-in για το Eclipse που παρέχει μια σειρά από εργαλεία ενσωματωμένα με το Eclipse IDE. Το ADT προσφέρει πρόσβαση σε πολλές λειτουργίες που βοηθούν στην ανάπτυξη Android εφαρμογών γρήγορα. Επίσης παρέχει GUI πρόσβαση σε πολλά από τα εργαλεία της γραμμής εντολών SDK, καθώς και ένα εργαλείο σχεδιασμού UI για την ταχεία τροποποίηση, το σχεδιασμό και την οικοδόμηση της διεπαφής χρήστη μιας εφαρμογής.

3.5 Εγκατάσταση του Eclipse IDE

Για την εγκατάσταση του Eclipse IDE μεταβαίνουμε στην επίσημη ιστοσελίδα της Google στο σύνδεσμο http://developer.android.com/sdk/index.html και ακολουθούμε τις οδηγίες. Επίσης θα πρέπει να είναι εγκαταστημένο και το jdk και το JRE , τα οποία τα κατεβάζουμε από την ιστοσελίδα της Oracle http://www.oracle.com/technetwork/java/javase/downloads/index.html και προχωραμε στην εγκατάσταση τους.

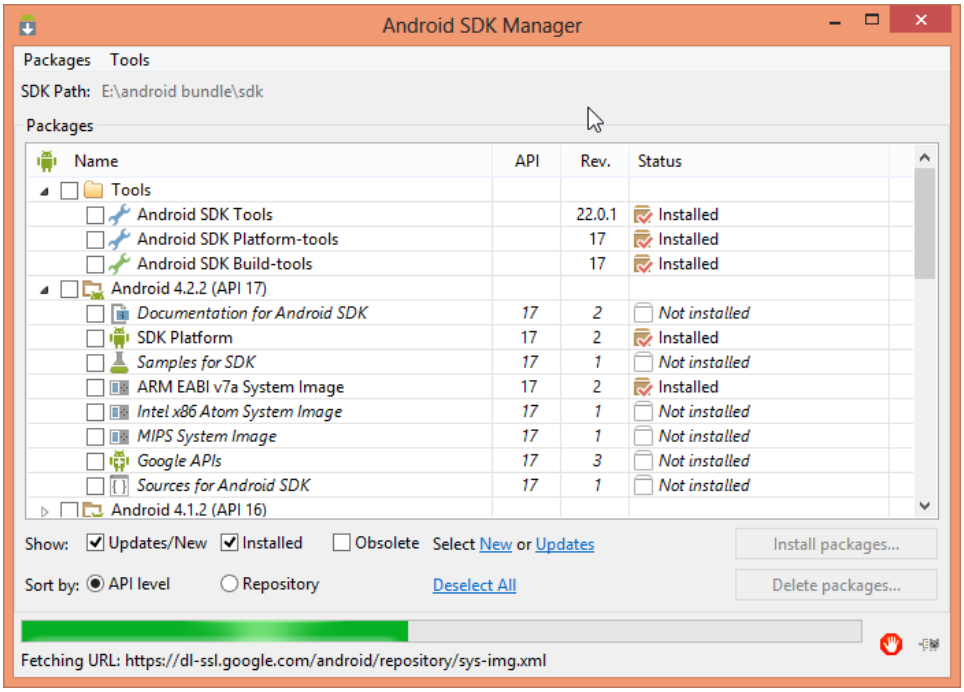


Εικόνα 10: Eclipse download

Εικόνα 11: JDK download

3.6 Εγκατάσταση του Android SDK Manager

Το Android SDK, παρέχει τα εργαλεία και το API που χρειάζεται κάποιος για να ξεκινήσει να αναπτύσσει εφαρμογές στην πλατφόρμα Android χρησιμοποιώντας την γλώσσα προγραμματισμού Java. Μέσα από το Eclipse, πηγαίνουμε στο Window και πατάμε στο Android SKD Manager. Όταν αυτό ανοίξει επιλέγουμε και εγκαθιστούμε τα πακέτα (packages) που θα μας είναι απαραίτητα για την δημιουργία της εφαρμογής.

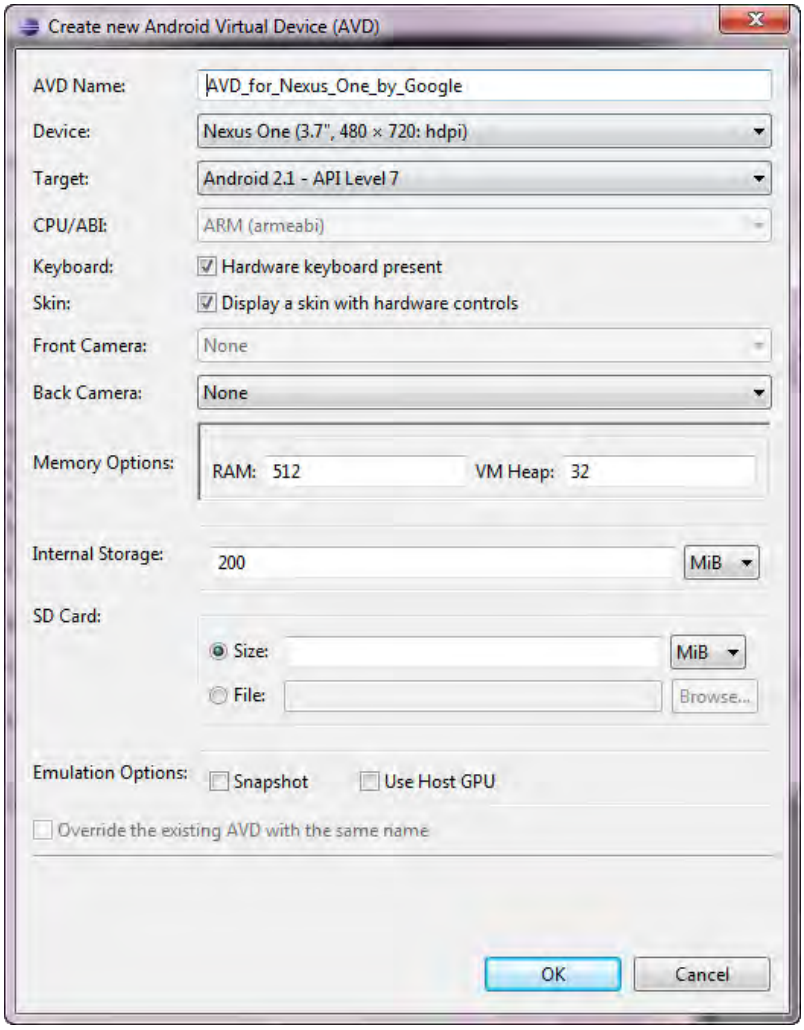


Εικόνα 12: Android SDK Manager

3.7 Δημιουργία εικονικής Android συσκευής

Το Android Virtual Device (AVD) είναι ένα εικονικό smartphone που τρέχει Android το οποίο μπορούμε εύκολα να εκτελέσουμε οποιαδήποτε εφαρμογή δημιουργήσουμε. Το πλεονέκτημα της AVD είναι ότι μπορούμε να δούμε πώς θα προβάλλεται το λογισμικό που δημιουργήσαμε σε διαφορετικά smartphones με Android.

Για την δημιουργία μιας τέτοιας εικονικής συσκευής πηγαίνουμε στο πρόγραμμα Eclipse στην επιλογή Window =>AVD Manager και επιλέγουμε View. Στο παράθυρο εμφανίζονται οι επιλογές μας και διαλέγουμε τα χαρακτηριστικά του smartphone μας και πατάμε την επιλογή Create AVD.

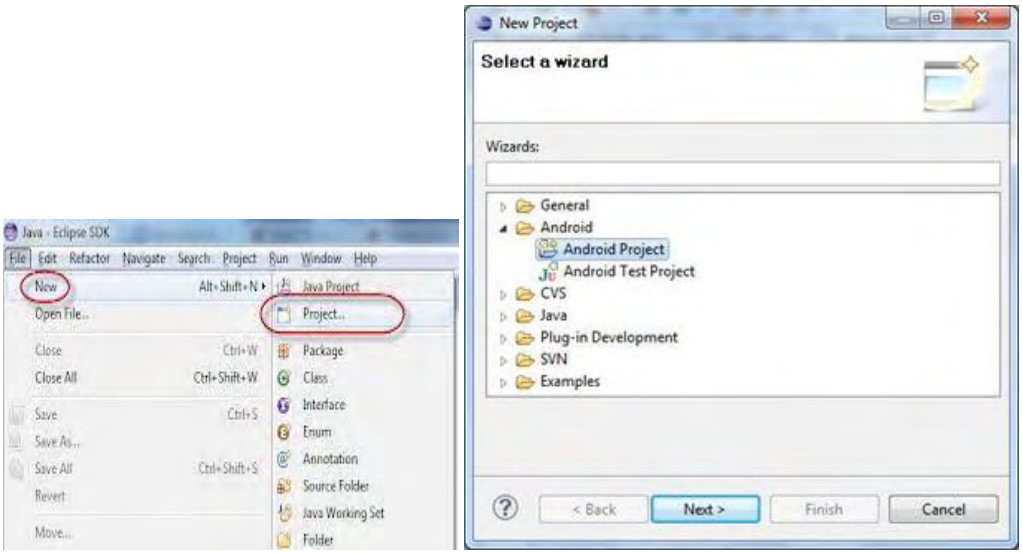


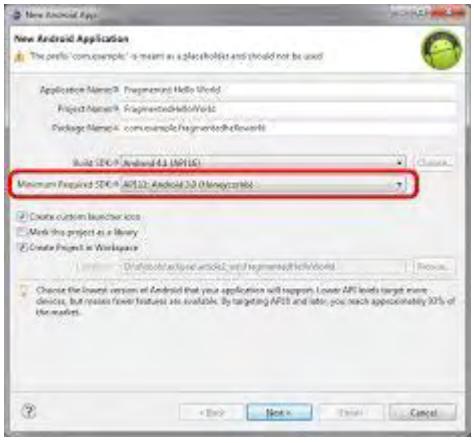
Εικόνα 13: Δημιουργία εικονικής συσκευής

3.8 Δημιουργία νέου Project

Για να δημιουργήσουμε ένα καινούργιο Android Project επιλέγουμε την επιλογή File =>New όπου επιλέγουμε την επιλογή Android Application Project.

Στο παράθυρο που ανοίγει βάζουμε το όνομα της εφαρμογής του Project και του πακέτου. Στην συνέχεια, επιλέγουμε την μικρότερη και μεγαλύτερη έκδοση στην οποία θέλουμε να τρέχει η εφαρμογή μας και πατάμε Next. Επιλέγουμε την εικόνα της εφαρμογής και πατάμε Next. Διαλέγουμε Blank activity και πατάμε Next. Επιλέγουμε τα ονόματα της Activity και του Layout και πατάμε Finish.





Εικόνα 14: New Android Project

4 Ανατομία της εφαρμογής Android Taxicost

Υπάρχουν τέσσερα δομικά blocks από τα οποία αποτελείται μια Android εφαρμογή: Activity, Intent Receiver, Service, Content Provider. Δεν χρειάζεται κάθε εφαρμογή να έχει και τα τέσσερα αυτά συστατικά , αλλά σίγουρα χρειάζεται κάποιον συνδυασμό από αυτά.

4.1 Android Manifest

Όταν ο προγραμματιστής αποφασίσει ποια στοιχεία του χρειάζονται για να αναπτύξει την εφαρμογή, αυτά πρέπει να καθοριστούν και να καταγραφούν σε ένα αρχείο που ονομάζεται AndroidManifest.xml. Στο αρχείο αυτό, δηλώνονται τα δομικά blocks που θα χρησιμοποιηθούν και ποιες δυνατότητες και προδιαγραφές θα εξυπηρετούν.

Σε κάθε εφαρμογή πρέπει να υπάρχει το αρχείο AndroidManifest.xml, το οποίο δημιουργείται αυτόματα όταν ξεκινάμε καινούργιο project ενός android application. Το αρχείο αυτό περιέχει βασικές πληροφορίες σχετικά με την εφαρμογή, τις οποίες το λειτουργικό σύστημα πρέπει να γνωρίζει προτού τρέξει οποιοδήποτε άλλο κώδικα. Οι σημαντικότερες από αυτές τις πληροφορίες περιγράφονται παρακάτω:

- Η ονομασία του πακέτου της Java της εφαρμογής (Java package).
- Η έκδοση της εφαρμογής (π.χ. 2.2, 3.0, 4.2.2 κλπ).
- Η ελάχιστη έκδοση του λειτουργικού συστήματος Android που απαιτεί η εφαρμογή (min sdk version). Για παράδειγμα αν έχει δηλωθεί min sdk version ο αριθμός 7, που ισοδυναμεί με την έκδοση Android 2.1, τότε η εφαρμογή θα μπορεί εκτελεστεί σε συσκευές με έκδοση Android μεγαλύτερη ή ίση της 2.1.
- Το όνομα της εφαρμογής, καθώς και το εικονίδιό της.
- Οι άδειες που απαιτούνται για να εκτελεστούν ορισμένες λειτουργίες της εφαρμογής.

Ιδιαίτερο στοιχείο το οποίο δηλώνεται στο αρχείο Manifest είναι οι άδειες πρόσβασης της εφαρμογής. Στο android οι διάφοροι πόροι της συσκευής (δίκτυο, εξωτερική κάρτα μνήμης, GPS) προστατεύονται και για να μπορεί κάποια εφαρμογή να τους χρησιμοποιήσει πρέπει να έχει δηλωθεί η αντίστοιχη άδεια στο αρχείο αυτό. Για

παράδειγμα, αν η εφαρμογή μας χρησιμοποιεί την κάμερα της συσκευής, θα πρέπει να δηλώνεται και αντίστοιχη άδεια. Αν θέλουμε να έχουμε πρόσβαση στο διαδίκτυο από την εφαρμογή μας, θα πρέπει να δηλώνεται η αντίστοιχη άδεια. Όμοια ισχύουν για το αν θέλουμε να αποθηκεύσουμε αρχεία στην κάρτα SD, αν θέλουμε να στείλουμε μηνύματα SMS, αν θέλουμε να πραγματοποιήσουμε τηλεφωνικές κλήσεις κλπ. Αν δε δηλώσουμε τις άδειες που απαιτούνται για τις διάφορες λειτουργίες της εφαρμογής, τότε θα εμφανίζεται σφάλμα και η εφαρμογή δε θα λειτουργεί. Οι άδειες αυτές περιγράφονται στο χρήστη τη στιγμή που εγκαθιστά την εφαρμογή, και πρέπει να συμφωνήσει με αυτές για να ολοκληρωθεί η εγκατάσταση. Με τον τρόπο αυτό, ο χρήστης αισθάνεται ασφαλής απέναντι σε κακόβουλες εφαρμογές. Για παράδειγμα αν ο χρήστης επιχειρήσει να εγκαταστήσει μία εφαρμογή άσχετη με τηλεφωνικές κλήσεις ή μηνύματα, και δει πριν την εγκατάσταση ότι αυτή ζητάει άδεια για τηλεφωνικές κλήσεις ή αποστολή μηνυμάτων, τότε θα καταλάβει ότι η εφαρμογή αυτή είναι πιθανότατα κακόβουλη και δεν πρέπει να εγκατασταθεί. Αν, βέβαια, οι συγκεκριμένες αυτές άδειες δεν αναγράφονται, τότε ο χρήστης είναι σίγουρος ότι η εφαρμογή δε θα μπορέσει με κανένα τρόπο να στείλει κάποιο γραπτό μήνυμα ή να πραγματοποιήσει κάποια τηλεφωνική κλήση. Ακόμα και αν επιχειρούσε να το κάνει, η εφαρμογή θα εμφάνιζε σφάλμα τη στιγμή που επιχειρούσαμε να την τρέξουμε και δε θα λειτουργούσε.

Το αρχείο AndroidManifest.xml της εφαρμογής μας:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.dbekos.taxicost"
   android:versionCode="1"
   android:versionName="1.0" >

 <uses-feature
      android:glEsVersion="0x00020000"
      android:required="true"/>

<uses-sdk
      android:minSdkVersion="10"
      android:targetSdkVersion="19" />
```

```
<permission
      android:name="com.dbekos.taxicost.permission.MAPS_RECEIVE"
      android:protectionLevel="signature" />
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="com.dbekos.taxicost.permission.MAPS_RECEIVE" />
<uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVICES"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>

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

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

<activity
         android:name=".MainActivity"
         android:label="@string/app_name"
         android:screenOrientation="portrait"
         android:windowSoftInputMode="stateHidden" >
         <intent-filter>
            <action android:name="com.dbekos.taxicost.MAINACTIVITY" />
            <category android:name="android.intent.category.DEFAULT" />
         </intent-filter>
      </activity>
```

```
        <activity
            android:name=".Preferences"
            android:label="@string/app_name"
            android:screenOrientation="portrait"
            android:windowSoftInputMode="stateHidden" >
        </activity>

        <activity
            android:name=".Cost"
            android:label="@string/app_name"
            android:screenOrientation="portrait"
            android:windowSoftInputMode="stateHidden" >
        </activity>

        <activity
            android:name=".About"
            android:label="@string/app_name"
            android:theme="@android:style/Theme.Dialog">
            <intent-filter>
                <action android:name="com.dbekos.taxicost.ABOUT" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>

        <activity
            android:name=".Help"
            android:label="@string/app_name"
            android:theme="@android:style/Theme.Dialog">
            <intent-filter>
                <action android:name="com.dbekos.taxicost.HELP" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>

      <meta-data
            android:name="com.google.android.maps.v2.API_KEY"
            android:value="AIzaSyBjzOep_-wIU2Uriccrlfmjr-YWXsbnS1o"/>

    </application>

</manifest>
```

4.2 Activity

Activity είναι ένα συστατικό της εφαρμογής το οποίο παρέχει τη διεπαφή με την οποία αλληλεπιδρά ο χρήστης ώστε να κάνει κάποιες ενέργειες, όπως για παράδειγμα κλήση μιας επαφής, αποστολή ενός mail, τράβηγμα φωτογραφίας κλπ.

Κάθε οθόνη που βλέπει και αλληλεπιδρά ο χρήστης είναι μια κλάση που κληρονομεί την κλάση activity. Μια εφαρμογή συνήθως αποτελείται από πολλές activities που συνδέονται μεταξύ τους. Το πιο κοινό σενάριο είναι να υπάρχει μια κεντρική activity, η οποία παρουσιάζεται στο χρήστη κατά την εκτέλεση της εφαρμογής. Κάθε activity μπορεί να ξεκινήσει μια άλλη ώστε να εκτελέσει διαφορετικές ενέργειες. Κάθε φορά που μια νέα activity ξεκινά, η προηγούμενη σταματάει, αλλά το σύστημα την βάζει σε μια στοίβα. Έτσι όταν ο χρήστης ολοκληρώσει τις ενέργειες του σε αυτή την activity και πατήσει το πλήκτρο επιστροφής, η προηγούμενη activity θα επαναφερθεί από τη στοίβα και θα συνεχίσει την εκτέλεσή της.

Κάθε Activity έχει ένα συγκεκριμένο κύκλο ζωής και κάποιες καταστάσεις, όπως για παράδειγμα Resumed, Paused και Stopped. Όταν μια Activity περνάει σε μια κατάσταση αυτό αναγνωρίζεται από κάποιες callback methods.

4.3 Intent Receiver

Ένας intent receiver χρειάζεται όταν ο προγραμματιστής της εφαρμογής θέλει να χρησιμοποιήσει κώδικα μέσα στην εφαρμογή του, που θα εκτελείται όταν συμβαίνει ένα εξωτερικό γεγονός, για παράδειγμα όταν χτυπά το τηλέφωνο ή όταν ένα ασύρματο δίκτυο γίνεται διαθέσιμο. Οι intent receivers δεν προβάλλουν κάποιο interface χρήστη, αλλά προβάλλουν Notifications για να ειδοποιήσουν τον χρήστη, εάν κάτι σημαντικό λαμβάνει χώρα. Οι Intent Receivers δηλώνονται και αυτοί στο AndroidManifest.xml.

4.4 Services

Ένα service είναι τμήμα κώδικα που εκτελείται χωρίς κάποιο interface χρήστη. Ένα καλό παράδειγμα service είναι ο media player που παίζει τραγούδια από μια λίστα. Σε μια εφαρμογή media player, είναι λογικό να υπάρχουν διάφορες οθόνες, άρα και πολλές activities, όπου ο χρήστης θα μπορεί να επιλέξει τραγούδια και να τα ακούσει. Παρόλα αυτά το playback δε θα πρέπει να χειρίζεται από μια activity, γιατί ο χρήστης περιμένει να μπορεί να περιηγείται στη λίστα τραγουδιών του, χωρίς το τραγούδι που ακούει εκείνη τη στιγμή να σταματήσει . Σε αυτή την περίπτωση, η κύρια activity του media player θα ξεκινήσει να εκτελεί ένα service στο background, οπότε ο χρήστης θα μπορεί να κάνει pause, rewind κλπ.

4.5 Content Providers

Οι εφαρμογές μπορούν να αποθηκεύσουν τα δεδομένα τους σε αρχεία, στη βάση δεδομένων SQLite, σε preferences ή χρησιμοποιώντας οποιονδήποτε άλλο μηχανισμό που τους παρέχει αυτή τη δυνατότητα. Ένας content provider επίσης είναι χρήσιμος εάν θέλουμε τα δεδομένα μιας εφαρμογής να γίνουν διαθέσιμα και σε άλλες εφαρμογές. Ένας content provider είναι μια κλάση που υλοποιεί ένα standard set από μεθόδους, οι οποίες επιτρέπουν σε άλλες εφαρμογές να αποθηκεύουν και να ανακτούν τον τύπο δεδομένων που χειρίζεται ο content provider.

4.6 Resources

Στα resources μιας εφαρμογής ορίζεται το layout των activities, οι διάφορες εικόνες και λεκτικά που χρησιμοποιούνται στα activities. Σε κάθε activity αντιστοιχεί ένα Layout αρχείο, το οποίο περιγράφει τη θέση των διάφορων αντικειμένων στην οθόνη. Το layout αρχείο είναι ένα αρχείο XML. Στην πράξη το αρχείο αυτό διαμορφώνεται από κατάλληλους γραφικούς editors που προσφέρονται από ολοκληρωμένα περιβάλλοντα ανάπτυξης όπως το Eclipse. Στην ενότητα για την διεπαφή χρήστη αναφέρθηκε ότι η διάταξη των γραφικών στοιχείων δηλώνεται σε αρχεία xml. Συγκεκριμένα στο project της εφαρμογής μας υπάρχει ο φάκελος res/layout/ στον οποίο τοποθετούμε όλα τα αρχεία xml που αφορούν στο user interface της εφαρμογής

μας (το res προέρχεται από το resources). Η εντολή που χρησιμοποιείται για να δηλωθεί το αρχείο xml που θα χρησιμοποιηθεί σε μία activity είναι η setContentView().

4.7 Διεπαφή χρήστη

Η διεπαφή χρήστη έχει τεράστια σημασία για κάθε εφαρμογή. Αποτελεί την τελική εικόνα που βλέπει ο χρήστης, το γραφικό περιβάλλον στο οποίο θα περιηγείται, και ενεργοποιεί όλες τις λειτουργίες της εφαρμογής.

Το user interface και η λειτουργικότητα της εφαρμογής αποτελούν αλληλένδετα στοιχεία και χωρίς το ένα δε μπορεί να υπάρξει το άλλο. Πολλές φορές μάλιστα είναι δυσκολότερος ο σχεδιασμός ενός όμορφου και εύχρηστου περιβάλλοντος εργασίας, παρά η ίδια η λειτουργικότητα της εφαρμογής. Καθίσταται σαφές, λοιπόν, ότι απαιτεί μεγάλη προσπάθεια και προσοχή η δημιουργία ενός γραφικού περιβάλλοντος που θα προσελκύει τους χρήστες και θα τους ωθεί να χρησιμοποιούν μία συγκεκριμένη εφαρμογή έναντι μίας άλλης, με τις ίδιες λειτουργίες.

4.7.1 Layout

Σε κάθε οθόνη της εφαρμογής, πρωταρχικό στοιχείο του γραφικού περιβάλλοντος αποτελεί η διάταξη των γραφικών στοιχείων ή layout. Το layout περιλαμβάνει όλα τα γραφικά στοιχεία της οθόνης, τα οποία μπορεί να είναι διατεταγμένα σε επιμέρους layouts.

Υπάρχουν 4 είδη layout, τα LinearLayout (γραμμική διάταξη), RelativeLayout (σχετική διάταξη), FrameLayout (διάταξη πλαισίου) και TableLayout (διάταξη πίνακα). (Υπάρχει και το AbsoluteLayout, το οποίο όμως έχει προταθεί να μην χρησιμοποιείται πλέον, διότι ορίζει τις απόλυτες θέσεις κάθε στοιχείου, οι οποίες όμως διαφέρουν ανάλογα με το κινητό τηλέφωνο και την οθόνη που χρησιμοποιείται η εφαρμογή).

Το LinearLayout αποτελεί διάταξη στοιχείων σε οριζόντια ή κατακόρυφη σειρά. Αν δηλαδή δηλώσουμε τρία στοιχεία A, B, C μέσα σε ένα οριζόντιο LinearLayout τότε

τα στοιχεία αυτά θα εμφανίζονται στην οθόνη σε μία οριζόντια διάταξη με τη σειρά που τα δηλώσαμε το ένα δίπλα στο άλλο.

Το RelativeLayout μας δίνει περισσότερη ελευθερία στη δήλωση των γραφικών στοιχείων, με την έννοια ότι κάθε στοιχείο μπορούμε να επιλέξουμε να το εμφανίσουμε σε συγκεκριμένο σημείο της οθόνης, όπως π.χ. στην αρχή, στο κέντρο, στο τέλος ή να επιλέξουμε τη θέση του σε σχέση με κάποιο άλλο στοιχείο.

Το FrameLayout αποτελεί την απλούστερη διάταξη στοιχείων. Είναι απλά ένας κενός χώρος τον οποίο μπορούμε να γεμίσουμε με κάποιο αντικείμενο, π.χ. μία εικόνα. Για το λόγω αυτό συνήθως χρησιμοποιείται σαν ρίζα (root) στο δέντρο των γραφικών στοιχείων της οθόνης.

Τέλος, το TableLayout όπως φανερώνει και η ονομασία του αποτελεί διάταξη πίνακα, δηλαδή μπορεί να διατάσσει τα παιδιά του (children) σε σειρές και στήλες. Σε όλα τα παραπάνω στοιχεία μπορούμε να ρυθμίσουμε αρκετές παραμέτρους όπως μέγεθος (πλάτος και ύψος), οριζόντια ή κατακόρυφη διάταξη, βαρύτητα (layout gravity) και άλλες.

Υπάρχουν δύο τρόποι για να δηλώσει κανείς τα layouts (όπως και κάθε άλλο γραφικό στοιχείο) της εφαρμογής: α) μέσω αρχείων xml ή β) μέσα στις activities της εφαρμογής. Τα αρχεία xml αποτελούν στατικό τρόπο δημιουργίας των γραφικών στοιχείων. Αποθηκεύονται σε συγκεκριμένο φάκελο του project και καλούνται για τη δημιουργία του γραφικού περιβάλλοντος μέσα από τις activities.

Πολλές φορές ωστόσο δε γνωρίζουμε εξαρχής τη διάταξη που θα χρησιμοποιήσουμε, διότι ίσως να εξαρτάται από ορισμένες επιλογές του χρήστη. Στις περιπτώσεις αυτές δημιουργούμε δυναμικά τα layouts μέσα στις activities και ορίζουμε εκεί τις παραμέτρους αυτών. Ωστόσο ο πιο εύκολος και συνηθέστερος τρόπος είναι (αν έχουμε τη δυνατότητα) να δημιουργήσουμε για κάθε οθόνη ένα διαφορετικό xml αρχείο που θα περιλαμβάνει τη διάταξη όλων των γραφικών της στοιχείων, και να το καλέσουμε μέσα από την αντίστοιχη activity.

4.7.2 Menu

Τα μενού αποτελούν ένα σημαντικό κομμάτι της διεπαφής χρήστη για κάθε οθόνη της εφαρμογής, διότι παρέχουν στο χρήστη ένα γνωστό και φιλικό τρόπο για να εισάγει τις επιλογές του. Στο λειτουργικό σύστημα Android, υπάρχουν τρία διαφορετικά είδη μενού, το μενού επιλογών (options menu), το μενού πλαισίου (context menu) και το υπομενού (submenu), τα οποία δηλώνονται και αυτά σε αρχεία xml.

Το options menu αποτελεί το βασικότερο μενού μίας εφαρμογής. Εμφανίζεται τη στιγμή που πατάμε το κουμπί menu του κινητού μας τηλεφώνου και περιέχει όλες τις βασικές επιλογές της εφαρμογής μας. Αποτελεί κυρίως τον τρόπο με τον οποίο περιηγούμαστε μεταξύ των διαφορετικών οθονών και activities της εφαρμογής μας. Στον κώδικα της εφαρμογής μας ορίζουμε κάθε επιλογή του μενού σε ποια activity θα οδηγήσει το χρήστη.

Το context menu (το οποίο περιλαμβάνει επιλογές αντίστοιχες με το δεξί κλικ σε κλασικά λειτουργικά συστήματα) οποιοδήποτε γραφικό στοιχείο το ορίσει ο προγραμματιστής (εικόνα, κείμενο κλπ) ενεργοποιείται από τον χρήστη με παρατεταμένο πάτημα (press and hold ή long press) του στοιχείου αυτού. Για παράδειγμα στο context menu ενός κειμένου θα όριζε κανείς επιλογές “αντιγραφή”, “αποκοπή” κλπ, στο context menu μίας διεύθυνσης URL σε εφαρμογή web browser θα ορίζαμε επιλογές “άνοιγμα”, “άνοιγμα σε νέα καρτέλα” κ.ο.κ.

Το submenu μπορεί να προστεθεί σαν επιλογή στα δύο παραπάνω menu και όπως δείχνει και το όνομά του ανοίγει ένα επιπλέον μενού για περισσότερες επιλογές. Χρησιμοποιείται σε περιπτώσεις που η εφαρμογή μας εκτελεί αρκετές λειτουργίες και θέλουμε να τις οργανώσουμε σε μενού. (Αντίστοιχα με τις επιλογές “Αρχείο”, “Επεξεργασία”, “Προβολή” κλπ που διαθέτουν τα περισσότερα προγράμματα).

4.7.3 Dialogs

Ο διάλογος (dialog) είναι συνήθως ένα μικρό παράθυρο που εμφανίζεται στην οθόνη μπροστά από την activity που τον κάλεσε. Η activity αυτή χάνει την εστίαση που είχε (focus) και το παράθυρο του διαλόγου είναι το μοναδικό με το οποίο μπορεί να αλληλεπιδράσει ο χρήστης.

Χρησιμοποιείται είτε για ενημέρωση του χρήστη για κάποιο γεγονός είτε για να ορίσει ο χρήστης κάποια επιλογή του. Τα κυριότερα είδη διαλόγων είναι ο AlertDialog (διάλογος ειδοποίησης) και ο ProgressDialog (διάλογος προόδου).

Ο AlertDialog είναι ο πιο συνηθισμένος διάλογος. Αποτελείται από ένα τίτλο, ένα μήνυμα, ορισμένα κουμπιά ή μία λίστα από επιλογές. Για κάθε κουμπί του διαλόγου, ορίζουμε μέσα στην activity τις ενέργειες που θα ακολουθήσουν όταν το πατήσει ο χρήστης.

Ο ProgressDialog αποτελεί ουσιαστικά επέκταση του AlertDialog και χρησιμοποιείται όταν θέλουμε να εμφανίσουμε στο χρήστη την πρόοδο για κάποια ενέργεια. Για παράδειγμα όταν θέλουμε να κατεβάσουμε κάποιες εικόνες από το διαδίκτυο και να τις εμφανίσουμε στο χρήστη, θα χρειαστούμε κάποιο χρονικό διάστημα για να ολοκληρωθεί αυτή η ενέργεια. Οπότε για να μη βλέπει ο χρήστης μία κενή μαύρη οθόνη, εμφανίζουμε έναν ProgressDialog και στο background εκτελούμε τις χρονοβόρες διαδικασίες.

4.7.4 Ειδοποιήσεις (Notifications)

Σε πολλές περιπτώσεις θέλουμε να ενημερώσουμε το χρήστη για κάποιο γεγονός ή αποτέλεσμα σχετικό με την εφαρμογή μας. Ορισμένα από αυτά τα γεγονότα απαιτούν κάποια απάντηση από το χρήστη και άλλα όχι.

Για παράδειγμα όταν ο χρήστης αποθηκεύει ένα αρχείο, θα θέλαμε να δει κάποιο μήνυμα ότι το αρχείο αποθηκεύτηκε επιτυχώς. Ή όταν η εφαρμογή μας τρέχει στο background και θέλει να ενημερώσει το χρήστη για κάποιο γεγονός, θα πρέπει να στείλει κάποια ειδοποίηση την οποία ο χρήστης να μπορεί να "ανοίξει" όταν αυτός επιθυμεί. Στην πρώτη περίπτωση χρησιμοποιούμε toast notification, ενώ στη δεύτερη status bar notification.

Toast Notification

Το toast notification είναι ένα μήνυμα που εμφανίζεται για λίγα δευτερόλεπτα στο παράθυρο που βρίσκεται ο χρήστης, οποιασδήποτε εφαρμογής και αν είναι αυτό. Ο χώρος που καταλαμβάνει είναι ο ελάχιστος απαιτούμενος ώστε το μήνυμα να είναι εμφανές, ενώ ο χρήστης μπορεί όσο εμφανίζεται το μήνυμα, να αλληλεπιδρά με την activity στην οποία βρίσκεται. Δεν υπάρχει κάποια επιλογή σε αυτή την ειδοποίηση, παρά μόνο ενημέρωση, δηλαδή ο χρήστης δε μπορεί να αλληλεπιδράσει με την ειδοποίηση. Χρησιμοποιείται συνήθως για μικρά μηνύματα που δεν απαιτούν κάποια ενέργεια από το χρήστη, όπως για παράδειγμα "Το αρχείο αποθηκεύτηκε επιτυχώς", "Το ξυπνητήρι ορίστηκε στις ..." κλπ.

Status Bar Notification

Το status bar notification, όπως φανερώνει και το όνομά της, είναι μία ειδοποίηση η οποία εμφανίζεται στη status bar του κινητού τηλεφώνου μας, και την οποία μπορούμε να ανοίξουμε είτε βρισκόμαστε στο κεντρικό μενού του τηλεφώνου μας, είτε σε κάποια εφαρμογή. Αντίθετα με την toast, η status bar notification μπορεί να επιλεχθεί και να ξεκινήσει κάποια λειτουργία ανάλογα με τις ενέργειες που έχουμε ορίσει στον κώδικα της εφαρμογής. Για παράδειγμα, όταν κατεβάζουμε ένα αρχείο από το διαδίκτυο, όταν η λήψη ολοκληρωθεί, θα θέλαμε να επιλέξουμε την ειδοποίηση αυτή και με τον τρόπο αυτό είτε να ανοίξουμε το φάκελο που βρίσκεται το αρχείο, είτε να το τρέξουμε. Τις περισσότερες φορές οι toast notifications ενεργοποιούνται από activities, ενώ οι status bar notifications από services.

4.8 Google Maps

4.8.1 Περιγραφή Google Maps API

Το Google Maps API είναι μια web εφαρμογή υπηρεσιών, η οποία παρέχεται από την Google και προσφέρει οδικούς χάρτες και υπηρεσίες πλοήγησης σε δικτυακούς τόπους ή σε εφαρμογές Android. Οι υπηρεσίες που προσφέρονται είναι οι εξής:

- οδικοί χάρτες
- πλοήγηση διαδρομής (με αυτοκίνητο, με δημόσια μέσα μεταφοράς ή με τα πόδια)
- εντοπισμός επιχειρήσεων σε χώρες σε όλο τον κόσμο.

Το Google Maps API είναι δωρεάν για εμπορική χρήση, υπό τον όρο ότι το site/εφαρμογή στην οποία χρησιμοποιείται θα είναι προσβάσιμο στο κοινό χωρίς να χρεώνει τον χρήστη του για κάθε πρόσβαση, και δεν παράγουν περισσότερα από 25.000 προσβάσεις χάρτη ανά ημέρα. Η ερώτηση στο Google Maps API γίνεται με την αποστολή ενός αιτήματος HTTP GET στην Web εφαρμογή, και επιστρέφει την απάντηση σε μορφή XML ή JSON. Η εφαρμογή η οποία περιγράφεται σε αυτή την πτυχιακή χρησιμοποιεί μηνύματα JSON όπως περιγράφονται στη συνέχεια.

Οι εφαρμογές Android χρησιμοποιούν μια ειδικά σχεδιασμένη για αυτές έκδοση του API, το Google Maps Android API v2. Για να χρησιμοποιηθεί το Google Maps API από μία εφαρμογή Android απαιτείται η απόκτηση ενός "κλειδιού" (Google Maps API Key) από τον δημιουργό της εφαρμογής που το χρησιμοποιεί και η εισαγωγή του στον κώδικα της εφαρμογής. Για την απόκτηση είναι απαραίτητη η δημιουργία λογαριασμού Google και η εγγραφή στο Google APIs Console (https://code.google.com/apis/console/).

Στη συνέχεια για τη χρήση του εργαλείου πρέπει να οριστεί στο xml αρχείο Manifest της εφαρμογής (AndroidManifest.xml) το κλειδί το οποίο έχει δοθεί από την Google με την εισαγωγή του παρακάτω αντικειμένου σαν "παιδί" στοιχείου <application>:

```
<meta-data
 android:name="com.google.android.maps.v2.API_KEY"
 android:value="your_api_key"/>
```

Στο παραπάνω αντικαθίσταται το "your_api_key" με το κλειδί το οποίο έχει αποκτηθεί από το Google APIs Console.

4.8.2 Καθορισμός Δικαιωμάτων

Για τη χρήση του Google Maps Android API v2 χρειάζεται ο καθορισμός στην εφαρμογή δικαιωμάτων πρόσβασης σε λειτουργείες του συστήματος. Ο καθορισμός

ενός δικαιώματος γίνεται με την προσθήκη ενός στοιχείου <uses-permission> σαν παιδί του αντικειμένου <manifest> στο αρχείο AndroidManifest.xml της εφαρμογής με το συντακτικό:

```
<uses-permission android:name="permission_name"/>
```

Τα απαραίτητα δικαιώματα που χρειάζεται να καθοριστούν για τη χρήση του Google Maps API v2 είναι:

- android.permission.INTERNET: Χρησιμοποιεί το API για τη λήψη του χάρτη απο τους Google Maps servers στη συσκευή.
- android.permission.ACCESS_NETWORK_STATE: Επιτρέπει στο API να ελέγξει τη κατάσταση των συνδέσεων τις συσκευής, έτσι ώστε να καθορίσει αν μπορεί να γίνει λήψη δεδομένων.
- android.permission.WRITE_EXTERNAL_STORAGE: Επιτρέπει στο API να καταχωρίσει τα δεδομένα του χάρτη στον χώρο αποθήκευσης της συσκευής.
- com.google.android.providers.gsf.permission.READ_GSERVICES: Επιτρέπει στο API την πρόσβαση σε web υπηρεσίες της Google .

4.9 JavaScript Object Notation

Το JavaScript Object Notation, το οποίο είναι γνωστό και ως JSON, είναι ένα ελαφρύ πρότυπο ανταλλαγής δεδομένων. Το JSON είναι ένα πρότυπο κειμένου, το οποίο είναι τελείως ανεξάρτητο από γλώσσες προγραμματισμού αλλά χρησιμοποιεί πρακτικές οι οποίες είναι γνωστές στους προγραμματιστές της οικογένειας προγραμματισμού C, συμπεριλαμβανομένων των C, C++, C#, Java, JavaScript και πολλών άλλων.

Το JSON προσφέρεται ως εναλλακτική λύση στην XML, όμως δεν μπορεί να την αντικαταστήσει καθότι δεν υποστηρίζει Schema validation. Η χρήση της είναι πολύ εύκολη υπόθεση καθότι το JSON έχει και αυτό self-documented format, το οποίο περιγράφει τη δομή των δεδομένων. Ο κώδικας που χρησιμοποιείται έχει πολύ μικρό μέγεθος πράγμα που το καθιστά ιδανικό για κινητές συσκευές λόγω του ότι στην πλειοψηφία τους οι πάροχοι κινητής τηλεφωνίας ακολουθούν την πολιτική της ογκοχρέωσης.

Στην εφαρμογή μας ουσιαστικά τα δεδομένα που παίρνουμε από το Google Maps API για τα km και την διάρκεια της διαδρομής είναι JSON

4.10 AsyncTask

Η AsyncTask επιτρέπει την σωστή και εύκολη χρήση του UI Thread. Η κλάση αυτή μας επιτρέπει να εκτελέσουμε λειτουργίες στο παρασκήνιο και να δημοσιεύονται τα αποτελέσματα στο UI Thread χωρίς να χρειάζεται να τα χειριστούν τα threads.

Η AsyncTask έχει σχεδιαστεί για να είναι μια κλάση αρωγός γύρω από το Thread και τον Handler και δεν αποτελεί ένα γενικό πλαίσιο threading. Τα AsyncTasks θα πρέπει ιδανικά να χρησιμοποιηθούν για μικρής διάρκειας χειρισμούς (λίγα δευτερόλεπτα το πολύ). Εάν πρέπει να κρατήσει το thread για μεγάλο χρονικό διάστημα, συνιστάται να χρησιμοποιήσουμε τα διάφορα APIs που παρέχονται από την java.util.concurrentpackage όπως Thread Pool Executor και Future Task. Στην δική μας εφαρμογή η AsyncTask ήταν αρκετή για τις διεργασίες που εκτελούμε.

Η AsyncTask έχει τέσσερα στάδια:

- onPreExecute: Αυτή η μέθοδος εκτελείται πριν την doInBackground.
- doInBackground: Η doInBackground χρησιμοποιείται για υπολογισμούς που παίρνουν ώρα να εκτελεστούν, δηλαδή είναι το βασικό βήμα για να εκτελέσουμε τον κώδικα που θέλουμε. Οι παράμετροι της AsyncTask παίρνονται από αυτό το στάδιο και το αποτέλεσμα επιστρέφεται στο επόμενο βήμα που αναλύουμε παρακάτω.
- onPostExecute: Αυτή η μέθοδος εκτελείται αφότου η doInBackground ολοκληρώσει την κλήση της. Το αποτέλεσμα από την doInBackround περνιέται σε αυτή τη μέθοδο.
- onProgressUpdate: Αυτή η μέθοδος καλείται καλώντας την publishProgress ανά πάσα στιγμή ενώ εκτελείται η doInBackground. Συνήθως η onProgressUpdate χρησιμοποιείται για να ενημερώσουμε το UI Thread με κάποια μπάρα προόδου ενώ εκτελείται η doInBackground.

4.10.1 Το βασικό πλεονέκτημα της AsyncTask

Το λειτουργικό Android εφαρμόζει ένα μοντέλου ενιαίου Thread και οποτεδήποτε μια Android εφαρμογή ξεκινάει, δημιουργείται και ένα αντίστοιχο Thread. Τώρα, υποθέτοντας ότι εκτελούμε μια λειτουργία δικτύου με το πάτημα ενός κουμπιού στην εφαρμογή μας. Όταν πατήσουμε το κουμπί θα γίνει μια αίτηση στον server και έτσι πρέπει να περιμένουμε την απάντηση του server. Εξαιτίας του ενιαίου μοντέλου στο Thread, όση ώρα περιμένουμε την απάντηση του server, η εφαρμογή μας δεν θα ανταποκρίνεται. Έτσι χρησιμοποιούμε την AsyncTask για να αποφύγουμε λειτουργίες που παίρνουν αρκετό σχετικά χρόνο να εκτελεστούν και κάνουν το user interface της εφαρμογής μας να μην ανταποκρίνεται.

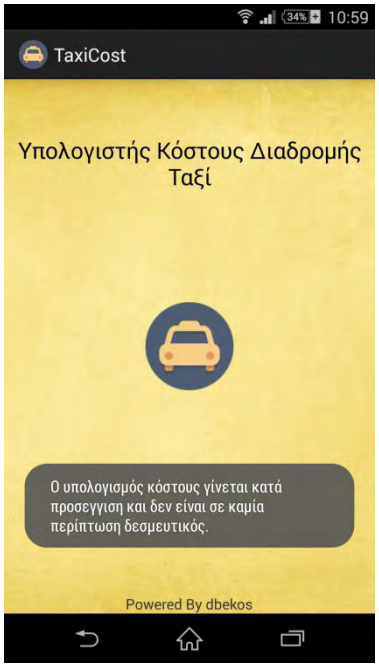
Έτσι δημιουργούμε ένα νέο Thread και τρέχουμε εκεί τη λειτουργία μας κάνοντας το βασικό Thread της εφαρμογής να παραμένει ανταποκρίσιμο στο χρήστη. Η κλάση AsyncTask περιέχει δηλαδή μεθόδους για να τρέξουμε μια λειτουργία που απαιτεί κάποιο χρόνο, στο παρασκήνιο, αλλά ταυτόχρονα μπορεί να επικοινωνεί και με το Thread της εφαρμογής και να συγχρονίζεται με αυτό ή και να το ενημερώνει όταν χρειάζεται.

5 Υλοποίηση της εφαρμογής Taxicost

Σε αυτό το κεφάλαιο θα παρουσιάσουμε τη λειτουργία της εφαρμογής μέσα από κάποια screenshots και ενδεικτικά μέσα από κάποια κομμάτια κώδικα.

5.1 Οθόνη Υποδοχής

Στην πρώτη οθόνη ελέγχουμε αν το κινητό διαθέτει τα google play services και σε αντίθετη περίπτωση εμφανίζεται ένα alert dialog. Επίσης υπάρχει και ένα toast notification για τον χρήστη.



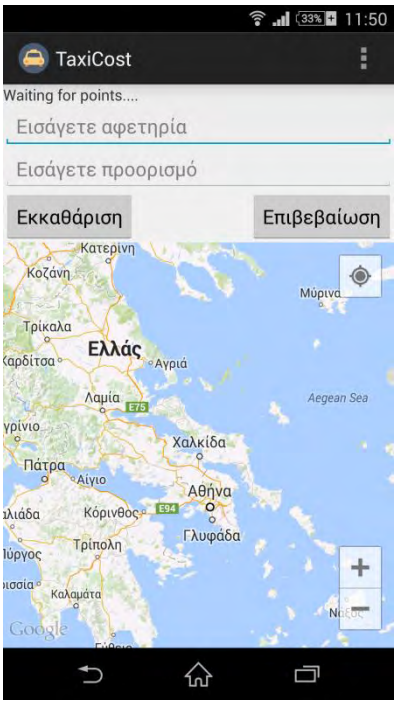
Ο κώδικας που το υλοποιεί είναι:

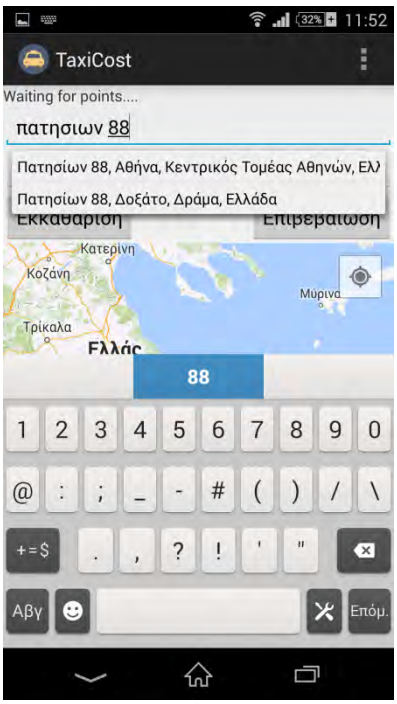
```
package com.dbekos.taxicost;

import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.common.GooglePlayServicesUtil;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.widget.Toast;
```

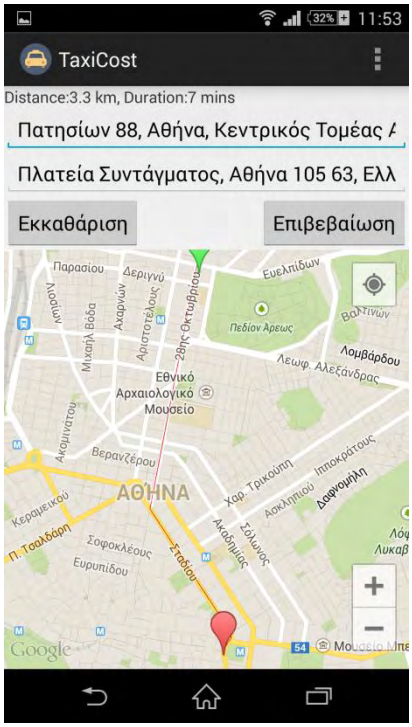
```
public class Splash extends Activity{

	@Override
	protected void onCreate(Bundle savedInstanceState) {
		// TODO Auto-generated method stub
		super.onCreate(savedInstanceState);
		setContentView(R.layout.splash);

		// Getting status of google play services availability
	int status = GooglePlayServicesUtil.isGooglePlayServicesAvailable(getBaseContext());

	// If not available, an alert dialog is shown.
	if(status!=ConnectionResult.SUCCESS){

	   AlertDialog alertDialog = new AlertDialog.Builder(this).create();
			alertDialog.setTitle("No Google Play Services Found!");
			alertDialog.setMessage("Exit application");
			alertDialog.setButton("OK", new DialogInterface.OnClickListener() {
			public void onClick(DialogInterface dialog, int which) {
			}
			});
			alertDialog.setIcon(R.drawable.alert);
			alertDialog.show();
	}

Toast.makeText(getBaseContext(), "Ο υπολογισμός κόστους γίνεται κατά προσεγγιση και δεν είναι σε καμία περίπτωση δεσμευτικός.", Toast.LENGTH_LONG).show();

	//using threads to show the splash screen for a certain time period
		Thread timer = new Thread(){
			public void run(){
				try{
					//sleeping for 2 seconds
					sleep(2000);
				} catch(InterruptedException e){
					e.printStackTrace();
				}finally{
					//starting the MainActivity
					Intent openMainActivity = new Intent(Splash.this, MainActivity.class);
					startActivity(openMainActivity);
				}
			}
		};
		timer.start();
	}

@Override
protected void onPause() {
	// TODO Auto-generated method stub
	super.onPause();
	finish();
}
}
```

5.2 Αφετηρία – Προορισμός

Ο χρήστης εδώ εισάγει την αφετηρία και τον προορισμό, είτε γράφοντας στο πλαίσιο, είτε πατώντας πάνω στο χάρτη, είτε κάνοντας συνδυασμό αυτών. Ακόμα υπάρχει και η δυνατότητα εντοπισμού της αφετηρίας μέσω GPS. Υπάρχει το button "εκκαθάριση" για να σβήσει ό,τι έχουμε κάνει μέχρι εκείνη τη στιγμή και μόλις είμαστε έτοιμοι πατάμε επιβεβαίωση για να περάσουμε στην επόμενη οθόνη. Φυσικά υπάρχουν όλοι οι έλεγχοι για Internet connection, για GPS, για το αν το σημείο είναι εντός Ελλάδας αλλιώς πετάει toast notification καθώς και για το κουμπί "επιβεβαίωση" όπου πρέπει να πατιέται αφού συμπληρωθούν τα δύο πεδία.







```
public class MainActivity extends FragmentActivity implements OnClickListener,
OnMyLocationButtonClickListener{

   GoogleMap map;
   ArrayList<LatLng> markerPoints;
   TextView tvDistanceDuration;
   MarkerOptions markerOptions;
   LatLng latLng;
   Marker myorigin, mydest;
   Context context;
   AutoCompleteTextView etfrom, etto;

   private boolean prediction; //access variable for setting destination instead of
                                  origin on first map click
   private String afethria, proorismos;     //variables to store the addresses

   private boolean doubleBackToExitPressedOnce = false;
   private static String distance = "";
   private static String duration = "";

   private boolean org, dst, noroute;          //helping access variables
   private int idn, actv_id;                   //helping access variables
   private final int acfrom = 1;               //final variable to assign to actv_id
   private final int acto = 2;                 //final variable to assign to actv_id

        @Override
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);
                org = dst = noroute  = prediction =  false;
                afethria = proorismos = "";

//initializing origin autocompletetextview (αντιστοιχα και για το destination)
                etfrom = (AutoCompleteTextView) findViewById(R.id.etfrom);
                etfrom.setOnItemClickListener(new OnItemClickListener() {

        @Override
         public void onItemClick(AdapterView<?> adapterView, View v, int
position,
                    long id) {
                // TODO Auto-generated method stub
                //Getting user input location
                InputMethodManager in = (InputMethodManager)
                getSystemService(Context.INPUT_METHOD_SERVICE);
in.hideSoftInputFromWindow(MainActivity.this.getCurrentFocus().getWindowToke
n(), 0);

                //get the id of the text field
                idn = MainActivity.this.getResources().getIdentifier("etfrom", "id",
                MainActivity.this.getPackageName());
                String origin = etfrom.getText().toString();
                afethria = origin;
                 if(origin!=null && !origin.equals("")){
                   new GeocoderTask().execute(origin);
                } }    });
```

```
//setting the adapter for the autocomplete (αντιστοιχα για το destination)

etfrom.setAdapter(new PlaceHolder(MainActivity.this,R.layout.list_item));
etfrom.addTextChangedListener(new TextWatcher() {

            @Override
            public void afterTextChanged(Editable s) {
               String value = s.toString();
               //check if text in field is empty
              if (value.equals("")) {
                    if(myorigin!=null){          //remove marker
                        myorigin.remove();
                    }
                    if(line!=null){              //remove line
                        line.remove();
                    }
                    if(markerPoints.size() == 1 && proorismos.equals("")){
                        markerPoints.clear();
                    }
                    if(markerPoints.size()==2 && !proorismos.equals("")){
                        markerPoints.remove(0);
                    }

                    afethria = "";
                    tvDistanceDuration.setText("Αναμονή για σημεία...");
                    //hide keyboard
                    InputMethodManager in = (InputMethodManager)
                    getSystemService(Context.INPUT_METHOD_SERVICE);

                    in.hideSoftInputFromWindow(MainActivity.this.getCurrentF
                    ocus().getWindowToken(), 0);

                        if(markerPoints.size()>0){
map.animateCamera(CameraUpdateFactory.newLatLng(markerPoints.get(0)) );
                        }
                    }
            }

            @Override
            public void beforeTextChanged(CharSequence s, int start, int count, int
after) {

            }

            @Override
            public void onTextChanged(final CharSequence s, int start, int before,
int count) {

            }
    });
```

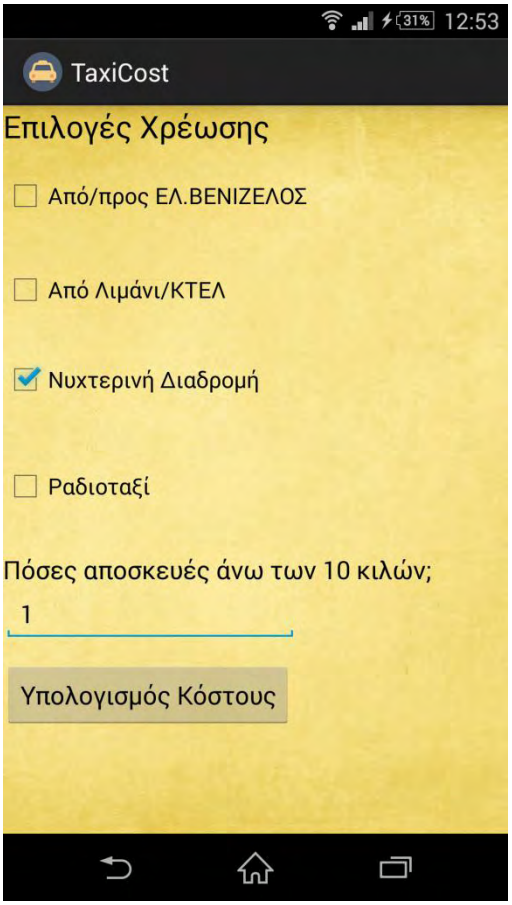
```
//initializing the distance and duration textview
tvDistanceDuration = (TextView) findViewById(R.id.tv_distance_time)

//Initializing the arraylist which will hold the points
markerPoints = new ArrayList<LatLng>();

//Getting reference to SupportMapFragment of the activity_main
SupportMapFragment fm =
(SupportMapFragment)getSupportFragmentManager().findFragmentById(R.id.map);

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

//setting a custom point as the center and zoom level
map.moveCamera( CameraUpdateFactory.newLatLngZoom(new LatLng(38.568478,
23.588942) , 6.75f) );

        //Enable MyLocation Button in the Map
        map.setMyLocationEnabled(true);
        map.setOnMyLocationButtonClickListener(this);

        //initializing the buttons
        Button bconfirm = (Button) findViewById(R.id.bcon);
        Button bclear = (Button) findViewById(R.id.bclear);

        //Setting button click event listener for the find button
        bconfirm.setOnClickListener(this);
        bclear.setOnClickListener(this);

        //Setting onclick event listener for the map
        map.setOnMapClickListener(new OnMapClickListener() {

        @Override
        public void onMapClick(LatLng point) {
        new OnMap().execute(point);
                        }
                });
        }

//A method for making the camera to show both markers on screen
private static void zoomToCoverAllMarkers(ArrayList<LatLng> latLngList,
GoogleMap googleMap) {
    LatLngBounds.Builder builder = new LatLngBounds.Builder();

    for (LatLng marker : latLngList)
    {
       builder.include(marker);                 //include all markers
    }

    LatLngBounds bounds = builder.build();
    int padding = 134;                  //offset from edges of the map in pixels
    CameraUpdate cu = CameraUpdateFactory.newLatLngBounds(bounds, padding);
    googleMap.animateCamera(cu);
  }
```

```
//A method for accessing the distance from another classes
        public double getDistance(){

            String str = distance;
            double dist = 0.00;
            String lastWord = str.substring(str.lastIndexOf(" ")+1);
            str = str.substring(0, distance.length()-2);      //discard the "km" characters
            str = str.replaceAll(",","");
            if(lastWord.equals("m")){
                  dist = Double.valueOf(str);
                  dist = dist/100;
                  return dist;
            }else if(lastWord.equals("km")){
                     dist = Double.valueOf(str);
                     return dist;
            }
                  return dist;
        }

//A method for setting the address in the autocompletetextview after the user touched the map to set a point
        private void gsAddress(LatLng point, int autoid){

        try{
                  Geocoder geo;
                  List<Address> address;
                  geo = new Geocoder(getBaseContext());
                  //getting the address, the city and the country
                  if (point.latitude != 0 || point.longitude != 0) {
                           address = geo.getFromLocation(point.latitude ,point.longitude, 1);
                           String adr = address.get(0).getAddressLine(0);
                           String city = address.get(0).getAddressLine(1);
                           String country = address.get(0).getAddressLine(2);

//checking the number of points on map in order to put the address in the correct field
                            if(autoid==acfrom){
                           etfrom.setText(adr + ", " + city + ", " + country);
                           afethria = (adr + ", " + city + ", " + country);
                           }
                            if(autoid==acto){
                           etto.setText(adr + ", " + city + ", " + country);
                           proorismos = (adr + ", " + city + ", " + country);
                           }
                  } else {
            Toast.makeText(getBaseContext(), "το γεωγραφικό πλάτος και μήκος είναι κενά", Toast.LENGTH_LONG).show();
            return;
                           }
        }   catch (Exception e) {
        e.printStackTrace();
        return;
      }
}
```

5.3 Εισαγωγή έξτρα επιλογών – χρεώσεων

Σε αυτό το σημείο ο χρήστης, για τον ακριβέστερο υπολογισμό του κόστους της διαδρομής του, καλείτε να επιλέξει, εάν το επιθυμεί, κάποιες υπηρεσίες οι οποίες περιέχουν επιπλέον χρεώσεις από την κανονική. Αυτές είναι:

- Από/προς ΕΛ. ΒΕΝΙΖΕΛΟΣ
- Από Λιμάνι/ΚΤΕΛ
- Νυχτερινή Διαδρομή
- Ραδιοταξί
- Αριθμός αποσκευών



Ο κώδικας της υλοποίησης δίνεται παρακάτω:

```
        //creating an object of MainActivity for accessing the distance variable
        MainActivity data = new MainActivity();

        CheckBox air, ktel, night, radio;
        Button calc;
        EditText numofbags;

private static double total = 1.19;
private final int cdist = 40;      //the average distance from el/venizelos airport to the center of athens
private final double tfdist = 24.00;     //the max distance where the price is 0.68 per kilometer
private final double lagcost = 0.40;    //the cost of a luggage over 10 kilograms
double extra, min, tarif, traffic;
int lag;                          //assisting access variable
boolean flag,airp;                //assisting access variables

/*The flag variable is set to true when the night checkbox is pressed. when the night checkbox is
pressed the flag variable helps us to check if night was previously checked and assign the
correct value. The airp variable is set to true when the airport checkbox is pressed and allows
us to enter the code for calculating the cost which refers to the airport. This happens because
the airport cost calculation differs from the normal.*/
/*We could not retrieve traffic info so we calculate traffic as 10% of the cost per klm*/

        @Override
        protected void onCreate(Bundle savedInstanceState) {
                // TODO Auto-generated method stub
                super.onCreate(savedInstanceState);
                setContentView(R.layout.preferences);

                //initializing the layout components
                air = (CheckBox) findViewById(R.id.air);
                ktel = (CheckBox) findViewById(R.id.ktel);
                night = (CheckBox) findViewById(R.id.night);
                radio = (CheckBox) findViewById(R.id.radio);
                numofbags = (EditText) findViewById(R.id.etbags);
                calc = (Button) findViewById(R.id.calc);

                //setting listeners
                air.setOnCheckedChangeListener(this);
                ktel.setOnCheckedChangeListener(this);
                night.setOnCheckedChangeListener(this);
                radio.setOnCheckedChangeListener(this);
                calc.setOnClickListener(this);

                initialize();
        }
```

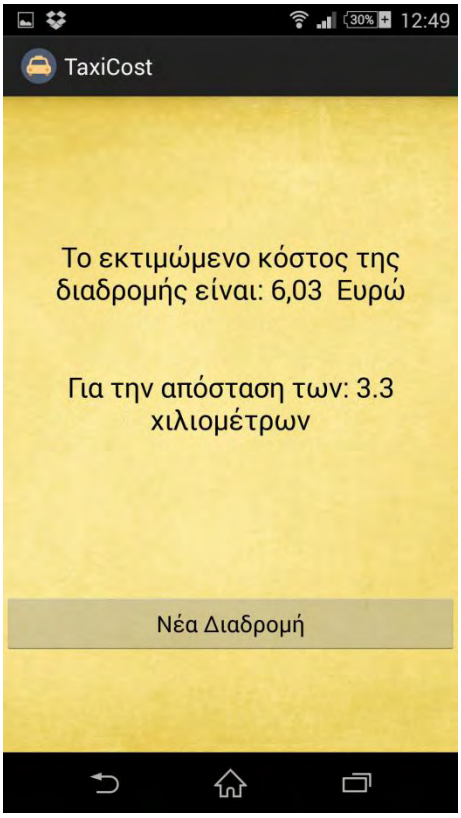
```
//method for initializing the variables.
        private void initialize(){
                total = 1.19;
                extra = 0.00;
                min = 3.16;
                tarif = 0.68;
                lag = 0;
                airp = false;
                flag = false;
                distance = data.getDistance();
                Calendar c = Calendar.getInstance();
                hour=c.get(Calendar.HOUR_OF_DAY);

//if the distance is over 40 klm then deactivate the night checkbox because the price is set to 1.19
        if((distance > tfdist) && (hour>=00 && hour<=05)){
                        night.setEnabled(false);
                        night.setChecked(true);
                }else if(distance > tfdist){
                        night.setEnabled(false);
                        night.setChecked(true);
                }else if(hour>=00 && hour<=05){ //check night if hour is between 00.00 and 05.00
                        night.setChecked(true);
                }
        }

@Override
        public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
                // TODO Auto-generated method stub
/*The if statements refer to the case of the checked checkbox and the else statements to the case we uncheck the checkbox. This helps us to undo the the changes to our cost variables.*/

        if(buttonView == night){
                if(isChecked){                  //if night is checked then change the price per klm
                                                and set to true  the access variable flag
                        tarif = 1.19;
                        flag = true;
                }else{                          //if night is not checked then undo the changes
                        tarif = 0.68;
                        flag=false;
                      }
        }
```

```
//Ομοίως και για τις άλλες περιπτώσεις (Ενδεικτικά για την περίπτωση Λιμάνι/ΚΤΕΛ)

if(buttonView == ktel){
        if(isChecked){                  //if checked add the extra cost
                extra += 1.07;
        }
        else{                                   //undo changes if unchecked
                extra -= 1.07;
                }
        }
}
//method for calculating the cost of the ride
        protected void calculateCost(){

                //getting the number of luggage

                if(numofbags.getText().toString().equals("")){
                        lag = 0;
                }else{
                        lag = Integer.parseInt(numofbags.getText().toString());
                }

                //block of code where the airport cost calculation is executed
                if(airp==true){
                        if(data.getDistance()>cdist){
                                double tmp = (data.getDistance() - cdist)*tarif;
                                traffic = tmp*0.1;
                                total += tmp + traffic;
                        }
                        airp=false;
                }else{

                        if(distance  >  tfdist){
                                tarif = 1.19;
                        }
                        total += distance* tarif;
                        traffic = total*0.1;                //calculating traffic
                        total += traffic + extra + lag*lagcost;         //calculating total
                        if(total<min){          //if the total is less than the minimum cost then
        set it to minimum
                                total = min;
                }

                }
```

```
@Override
	public void onClick(View v) {
		// TODO Auto-generated method stub
		//checking if the entry in luggage is a number
if(!numofbags.getText().toString().equals("") &&
isNumeric(numofbags.getText().toString())==false){
Toast.makeText(getBaseContext(), "Παρακαλώ εισάγετε αριθμό στο πλαίσιο των
αποσκευών.", Toast.LENGTH_SHORT).show();
			numofbags.setText("");
			return;
		}
		//calculate cost and start the cost activity
		calculateCost();
		Intent end = new Intent(this, Cost.class);
		startActivity(end);
	}

@Override
	protected void onPause() {
		// TODO Auto-generated method stub
		super.onPause();
		finish();
	}

	//get method for accessing the total variable from another class
	public double getCost(){
		return total;
	}
}
```

5.4 Υπολογισμός Κόστους

Με τα δεδομένα της προηγούμενης class, στην οθόνη αυτή εμφανίζουμε το ποσό που θα πληρώσει ο χρήστης και τα km που θα έχει κάνει. Τέλος, υπάρχει και το κουμπί "Νέα Διαδρομή" όπου μας πηγαίνει στην MainActivity, κλείνουν όλες οι υπόλοιπες activities και μπορούμε να αναζητήσουμε από την αρχή μια νέα διαδρομή.



Ο κώδικας για την activity αυτή είναι:

```
public class Cost extends Activity implements OnClickListener{
        TextView cost, dist;

        @Override
        protected void onCreate(Bundle savedInstanceState) {
                // TODO Auto-generated method stub
                super.onCreate(savedInstanceState);
                setContentView(R.layout.cost);
                //initializing the two textviews and the button
                cost = (TextView) findViewById(R.id.tvcost);
                dist = (TextView) findViewById(R.id.tvdist);
                Button newride = (Button) findViewById(R.id.bnew);
                newride.setOnClickListener(this);

                //creating an object of the preferences class so we get access to the
                total cost variable.
                Preferences amount = new Preferences();
                MainActivity data = new MainActivity();

                //creating a number format so only two decimal digits are shown.
                double number = amount.getCost();
                DecimalFormat dec = new DecimalFormat("0.00");
                dec.setMinimumFractionDigits(2);
                String money = dec.format(number);
```

```
                //showing the cost and the distance
cost.setText("Το εκτιμώμενο κόστος της διαδρομής είναι: " + money + "  Ευρώ");
dist.setText("Για την απόσταση των: " + Double.toString(data.getDistance()) + "
χιλιομέτρων");
}

        @Override
        protected void onPause() {
                // TODO Auto-generated method stub
                super.onPause();
                finish();
        }

        @Override
        public void onClick(View v) {
                // TODO Auto-generated method stub
                //starting and placing on top the MainActivity and closing all other
activities
                Intent mapadr = new Intent(this, MainActivity.class);
                mapadr.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                startActivity(mapadr);
        }

        //starting the previous activity when back button is pressed
        @Override
        public void onBackPressed() {
           Intent i = new Intent(this, Preferences.class);
           startActivity(i);
           finish();
        }
}
```

6 Συμπεράσματα και μελλοντική εξέλιξη

Πλέον είναι κατανοητό από όλους πως ο κόσμος των smartphones είναι ένας κόσμος ταχέως αναπτυσσόμενος. Όλες οι κατασκευάστριες εταιρίες κινητών τηλεφώνων δαπανούν τεράστια ποσά στην έρευνα και στην ανάπτυξη νέων τεχνολογιών με σκοπό να μείνουν στην κορυφή σε ένα όλο και πιο ανταγωνιστικό περιβάλλον.

Τα έξυπνα κινητά έχουν γίνει πλέον μέρος τις καθημερινότητας μας πράγμα που δημιουργούν καθημερινά νέες ανάγκες και απαιτήσεις πράγμα που κάνει την ανάπτυξη των κινητών αυτών αναγκαία.

Το android με το οποίο ασχοληθήκαμε, δείχνει ακριβώς αυτή την εικόνα. Είναι ένα λειτουργικό που συνεχώς βελτιώνεται και εξελίσσεται λαμβάνοντας υπόψιν τις ανάγκες των χρηστών αλλά και τον ανταγωνισμό.

Φροντίζει ώστε ο χρήστης να έχει ένα αρκετά φιλικό περιβάλλον στα χέρια του που θα εξυπηρετεί τις ανάγκες του όπως η γνώση της τοποθεσίας του, την μεταφορά δεδομένων μεταξύ άλλων συσκευών, την αποθήκευση όλο και μεγαλύτερου όγκου δεδομένων καθώς και την εμφάνιση των διεπαφών με διαδραστικό και εύχρηστο τρόπο.

Το Android ανά τακτά χρονικά διαστήματα εξελίσσεται και βελτιώνεται. Όσο θετικό είναι κάτι τέτοιο, ταυτόχρονα είναι και αρνητικό. Η συνεχής εξέλιξη και την ίδια στιγμή η απαρχαίωση μεθόδων των αντικειμένων καθιστούν αρνητικό παράγοντα στην ανάπτυξη του λογισμικού, με την έννοια ότι πρέπει να υποστηριχθούν αρκετές προηγούμενες εκδόσεις Android. Ελπίζουμε ότι το πρόβλημα αυτό θα ξεπεραστεί με συνεχείς αναβαθμίσεις των κινητών ώστε να υποστηρίζουν μεταγενέστερες εκδόσεις Android.

Για την ανάπτυξη της εφαρμογής μας χρησιμοποιήσαμε τη Java, η οποία είναι μια αντικειμενοστραφής γλώσσα προγραμματισμού, που χρησιμοποιείται κατά κόρον στην ανάπτυξη εφαρμογών. Δημιουργήσαμε αρχεία Java και αντίστοιχα .xml για να πάρουμε το συγκεκριμένο αποτέλεσμα.

Όπως είδαμε, το Android έχει εξελικτική πορεία. Έτσι και η δική μας εφαρμογή έχει περιθώρια ανάπτυξης. Μερικές προτάσεις για μελλοντική εξέλιξη και επέκταση της εφαρμογής είναι:

- Η δυνατότητα διατήρησης ιστορικού με τις τελευταίες αναζητήσεις κόστους διαδρομής του χρήστη
- Να προστεθεί και η αγγλική γλώσσα σαν επιλογή για τον χρήστη.

ΒΙΒΛΙΟΓΡΑΦΕΙΑ

[1] http://el.wikipedia.org/wiki/%CE%88%CE%BE%CF%85%CF%80%CE%BD%CE%BF_%CF%84%CE%B7%CE%BB%CE%AD%CF%86%CF%89%CE%BD%CE%BF

[2] http://el.wikipedia.org/wiki/%CE%9B%CE%B5%CE%B9%CF%84%CE%BF%CF%85%CF%81%CE%B3%CE%B9%CE%BA%CE%BF%CC%81_%CF%83%CF%8D%CF%83%CF%84%CE%B7%CE%BC%CE%B1

[3] http://en.wikipedia.org/wiki/Android_(operating_system)

[4] http://en.wikipedia.org/wiki/Google_Play

[5] http://faqoid.com/advisor/android-versions.php#android-timeline

[6] http://www.techotopia.com/index.php/An_Overview_of_the_Android_Architecture

[7] http://developer.android.com/sdk/index.html

[8] http://developer.android.com/tools/sdk/ndk/index.html

[9] http://en.wikipedia.org/wiki/Eclipse_(software)

[10] http://developer.android.com/tools/help/adt.html

[11] http://www.vogella.com/tutorials/AndroidBackgroundProcessing/article.html#concurrency_asynchtask

[12] http://www.javacodegeeks.com/page/2/?s=Android+Tutorial

[13] http://developer.android.com/guide/topics/ui/declaring-layout.html

[14] http://developer.android.com/guide/components/activities.html

[15] http://www.compiletimeerror.com/2013/01/why-and-how-to-use-asynctask.html#.U_wu7MV_uvR