Πανεπιστήμιο Θεσσαλίας
Πολυτεχνική Σχολή

Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Καταγραφή και διαχείριση ηλεκτρονικού εξοπλισμού με τεχνολογίες RFID και NFC

Electronic equipment administration with RFID and NFC technologies

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Μαρέλας Γεώργιος

Επιβλέποντες:

Μποζάνης Παναγιώτης
Αναπληρωτής Καθηγητής Π.Θ

Τσομπανοπούλου Παναγιώτα
Επίκουρος Καθηγήτρια Π.Θ

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

Καταγραφή και διαχείριση ηλεκτρονικού εξοπλισμού με τεχνολογίες RFID και NFC

Διπλωματική Εργασία

Μαρέλας Γεώργιος

Επιβλέποντες Καθηγητές: Μποζάνης Παναγιώτης
Αναπληρωτής Καθηγητής Π.Θ

Τσομπανοπούλου Παναγιώτα
Επίκουρος Καθηγήτρια Π.Θ

Εγκρίθηκε από την διμελή εξεταστική επιτροπή την 6η Μαρτίου 2015

.....	.....
Μποζάνης Παναγιώτης	Τσομπανοπούλου Παναγιώτα
Αναπληρωτής Καθηγητής Π.Θ	*Επίκουρος Καθηγήτρια Π.Θ*

Ευχαριστίες

Στα πλαίσια της διπλωματικής μου εργασίας θα ήθελα να ευχαριστήσω θερμά τον κύριο Μποζάνη Παναγιώτη -αναπληρωτή καθηγητή και επιβλεποντα της πτυχιακής- για την καθοδήγηση του, και την πολύτιμη στήριξη του, καθώς επίσης και την κυρία Τσομπανοπούλου Παναγιώτα –επίκουρο καθηγήτρια και συνεπιβλέπουσα της πτυχιακής. Επίσης θα ήθελα να ευχαριστήσω τον κύριο Φεύγα Αθανάσιο, συνεργάτη των καθηγητών μου και μέλος ΕΤΕΠ, καθώς σε όλη την διάρκεια της εργασίας είχαμε άψογη συνεργασία. Με καθοδήγησε με τρόπο καταλυτικό, αντιμετώπίζοντας τις όποιες δυσκολίες προέκυπταν, για την διεκπεραίωση της εφαρμογής και την συγγραφή της εργασίας.

Ένα μεγάλο ευχαριστώ στην οικογένεια μου, για την πολύτιμη υπομονή και στήριξη τους προς το πρόσωπο μου όλα αυτά τα χρόνια.

Μαρέλας Γιώργος
Βόλος, 2015

Περίληψη

Η παρούσα διπλωματική παρέχει μία επισκόπηση της νέας τεχνολογίας ανέπαφων συνδιαλλαγών (NFC: Near Field Communication) και των δυνατοτήτων της. Εκτός από μία σύντομη εισαγωγή στο θεωρητικό υπόβαθρο των βάσεων δεδομένων, της δομής και λειτουργίας του λογισμικού Android γίνεται εκτενής αναφορά στη δομή των εφαρμογών που πλαισιώνουν το συγκεκριμένο λογισμικό καθώς και των μεθόδων χρήσης της νέας τεχνολογίας. Παρουσιάζονται τρέχοντα αλλά και προβλεπόμενα πεδία εφαρμογής καθώς και πλεονεκτήματα και μειονεκτήματα της χρήσης. Η παρουσα εργασία καταλήγει παρουσιάζοντας την εφαρμογή Uth Labs Equipment που υποστηρίζει βάση δεδομένων και δίνει στον χρήστη την δυνατότητα διαχείρησης της, μέσω NFC. Αναλύεται πλήρως το τεχνικό υπόβαθρο της υλοποίησης της εφαρμογής και των λειτουργιών της.

Περιεχόμενα

Εισαγωγή

Η εργασία αυτή αποτελείται απο τέσσερα κεφάλαια. Στο πρώτο κεφάλαιο αναφέρονται αρχικά εφαρμογές που σχετίζονται με ηλεκτρονικές βάσεις δεδομένων και γίνεται αφορά στην δημιουργία και έμπνευση του λειτουργικού συστήματος Android. Επίσης γίνεται αναφορά στην εξέλιξη του και τις εκδόσεις του μέχρι και σήμερα.

Στο δεύτερο κεφάλαιο αναλύεται και περιγράφεται πλήρως η δομή του λογισμικού Android. Γίνεται περιγραφή της δομής εφαρμογών που εκτελούνται στο λογισμικό και παρουσιάζεται η ιεραρχία των καταλόγων και των δομικών αρχείων που υλοποιούνται. Αναφέρονται δομικά στοιχεία της εφαρμογής καθώς και μοντέλα δημιουργίας της ενώ τέλος, δίνεται μεγάλη έμφαση στην επεξήγηση του κύκλου ζωής των εφαρμογών.

Το τρίτο κεφάλαιο περιλαμβάνει εισαγωγή στην τεχνολογία του NFC, αναλύωντας τα χαρακτηριστικά της. Παρουσιάζονται πλεονεκτήματα και εφαρμογές που υπάρχουν ήδη και υποστηρίζουν την συγκεκριμένη τεχνολογία καθώς και ιδέες για μελλοντικές δημιουργίες βασιζόμενες στο NFC. Τέλος, γίνεται κατανοητό στον αναγνώστη το πώς μπορεί να προγραμματίσει μια εφαρμογή Android που υποστηρίζει την χρήση ανέπαφων συνδιαλλαγών.

Το τέταρτο κεφάλαιο περιγράφει την λειτουργία της εφαρμογής που αναπτύχθηκε καθώς και τα στάδια της υλοποίησης της. Αναφέρονται εργαλεία και λογισμικό που χρησιμοποίηθηκε και ξεδιπλώνονται οι επιμέρους λειτουργίες της παραθέτοντας λεπτομέρειες ως προς τον τρόπο υλοποίησης τους.

Τέλος στο Παράρτημα παρουσιάζονται με ιεραρχία κομμάτια κώδικα που υλοποιήθηκαν για την εφαρμογή. Παρετίθενται η βιβλιογραφία και οι ιστοσελίδες που χρησιμοποιήθηκαν για άντληση πληροφοριών.

ΚΕΦΑΛΑΙΟ 1^{Ο}

1.1 Βάσεις Δεδομένων και διασύνδεση τους με εφαρμογές

Με τον όρο βάση δεδομένων εννοούμε μια σύλλογή από ειδικά διαμορφωμένα δεδομένα στα οποία είναι δυνατή η αναζήτηση και προσθήκη χαρακτηριστικών κατόπιν απαίτησης του χρήστη. Υπάρχει μια σχηματική δομή ώς προς την οργάνωση τους βασιζόμενη σε πρωτεύοντα και γενικά χαρακτηριστικά. Είναι οργανωμένες, διακριτές συλλογές σχετιζόμενων δεδομένων. Η μορφή αποθήκευσης τους είναι ψηφιακή και διαχειρίζονται από ειδικό λογισμικό σύστημα το οποίο παρέχει και την δυνατότητα άντλησης πληροφοριών και ανανέωσης της βάσης.

Οι βάσεις δεδομένων παρέχουν πολλές δυνατότητες οργάνωσης και καταγραφής υλικού και πληροφοριών. Η ηλεκτρονική υλοποίηση τους και η σύνδεση με εφαρμογές που εκτελούνται σε λογισμικό έξυπνων τηλεφώνων είναι πλεόν πραγματικότητα. Δίνεται έτσι η δυνατότητα διαχείρισης και ενημέρωσης τους είτε μέσα από έναν συγκεκριμένο χώρο (για παράδειγμα, πανεπιστημιακά εργαστήρια) είτε εξ'αποστάσεως μέσω της πρόσβασης σε αυτές απο περιηγητές διαδικτύου ή εφαρμογές διαχείρησης τους. Παραθέτονται ορισμένα παραδείγματα :

- *ΚΕΠΛΗΝΕΤ* (Κέντο Πληροφορικής και Νέων Τεχνολογιών) : Είναι μία ηλεκτρονική βάση δεδομένων καταγραφής ηλεκτρονικού εξοπλισμού των σχολείων Α'Βάθμιας και Β΄Βάθμιας εκπαίδεσης.
- *e – Inventory* : Μητρώο Ψηφιακής Υποδομής των δημοσίων φορέων από το Υπουργείο Διοικητικής Μεταρύθμισης και Ηλεκτρονικής Διακυβέρνησης.
- *Ηλεκτρονική Βάση Κτηματολογίου*.
- Βάσεις Δεδομένων που χρησιμοποιούνται σε εταιρείες για καταγραφή βλαβών και συντήρησης του εξοπλισμού
- Βάσεις καταγραφής ηλεκτρονικού εξοπλισμού με χρήση τεχνολογίας RFID. Η συγκεκριμένη τεχνολογία είναι ένα μέσο για την συλλογή δεδομένων σχετικά με ένα στοιχείο, με ασύρματη διασύνδεση που οφείλονται στα ραδιοκύματα που μεταδίδονται μέσω του αέρα.
- *Badge NFC* : Είναι μια εφαρμογή για Android λογισμικό που υποστηρίζει NFC τεχνολογία. Καταγράφει σε μία βάση όλες τις συσχετίσεις του τηλεφώνου του με NFC ετικέτες, που πραγματοποιούνται από τον χρήστη.
- *Android Gym Application* : Εφαρμογή Android που χρησιμοποιεί NFC τεχνολογία και μέσω των αντίστοιχων ετικέτων(tags) καταγράφει σε βάση δεδομένων τον χρόνο και τις επαναλήψεις για κάθε άσκηση που εκτελεί ο χρήστης στο γυμναστήριο.

1.2 Android…Από την έμπνευση στην πράξη

[1] Στον κόσμο της τεχνολογίας, όπου τα αμύθητα ποσά και οι τεχνολογικές καινοτομιες διαδέχονται η μία την άλλη με ταχύτατους ρυθμούς, οι κολοσοί διαρκώς

μάχονται μεταξύ τους : Apple vs Microsoft, Google vs Apple vs Samsung κ.ο.κ. Άλλος κερδίζει, άλλος χάνει. Και το έπαθλο; ....Ο χρήστης!

Την Άνοιξη του 2005 ο Andy Rubin εμπνεύστηκετην χρήση της Google ως κατ' εξοχήν μηχανή αναζήτησης για το T-Mobile Sidekick. Ο Rubin παρουσίασε στον Larry Page (συνιδρυτής της Google) το Android ως ένα εν δυνάμει παγκόσμιο ανοικτό λειτουργικό σύστημα τονίζοντας τη σταθερή υπεροχή που παρατηρείται στις συνήθειες του αγοραστικού κοινού των κινητών τηλεφώνων, σε αντιδιαστολή με τις πωλήσεις ηλεκτρονικών υπολογιστών. Ο Page εκτός απο υποστηρικτής του Android, αποφάσισε να γίνει και ιδιοκτήτης του. Την ίδια χρόνική περίοδο στον αντίποδα εμφανίστηκε στο προσκήνιο το αντίπαλο δέος του Android, το iOS της Apple. Μέσω του ανταγωνισμού, στα τέλη του 2005 παρουσιάστηκε στην αγορά το IPHONE.

Το Φθινόπωρο του ίδιου έτους 34 εταιρίες, όπως η Texas Instruments, η Intel, η T-Mobile και η Sprint Nextel, συμφώνησαν με την Google για τη δημιουργία μιας πλατφόρμας ανοιχτού κώδικα που θα είχε ενσωματωμένο το λογισμικό Linux και θα εκπροσωπούνταν από μια νέα συστάδα εταιριών, την Open Handset Alliance. Η αρχή είχε γίνει και στην εταιρεία προστέθηκαν και άλλες εταιρείες, όπως η HTC, η Motorola και η LG, ανακοινώνοντας την πρόθεσή τους να δώσουν προς πώληση στην αγορά smartphones με λειτουργικό σύστημα Android σε διάφορα σχήματα και μεγέθη, με τα οποία θα μπορεί να έχει ο χρήστης να ενσωματώνει στο κινητό του πλήθος εφαρμογών.

1.3 Έξυπνα τηλέφωνα

Η βασική διαφορά των έξυπνων τηλεφώνων από τις απλές συσκευές κινητής τεχνολογίας είναι πως πλέον η λειτουργία των τηλεφώνων στηρίζεται στο λειτουργικό σύστημα που έχουν. Προσφέρουν περισσότερες δυνατότητες καθώς υποστηρίζουν υπολογιστική ικανότητα αντάξια με τους προσωπικούς υπολογιστές. Η δυνατότητα αναπαραγωγής μουσικής και βίντεο καθώς και η υποστήριξη φωτογραφικής μηχανής και καταγραφής βίντεο ήταν από τις πρωτοεμφανιζόμενες δυνατότητες των έξυπνων τηλεφώνων. Στη συνέχεια προστέθηκαν επιπλέον λειτουργίες, όπως η οθόνες αφής και η ασύρματη πρόσβαση στο διαδίκτυο κάτι το οποίο κατέστησε τα έξυπνα κινητά ως πολύχρηστες συσκευές.

1.4 Τι είναι το Android;

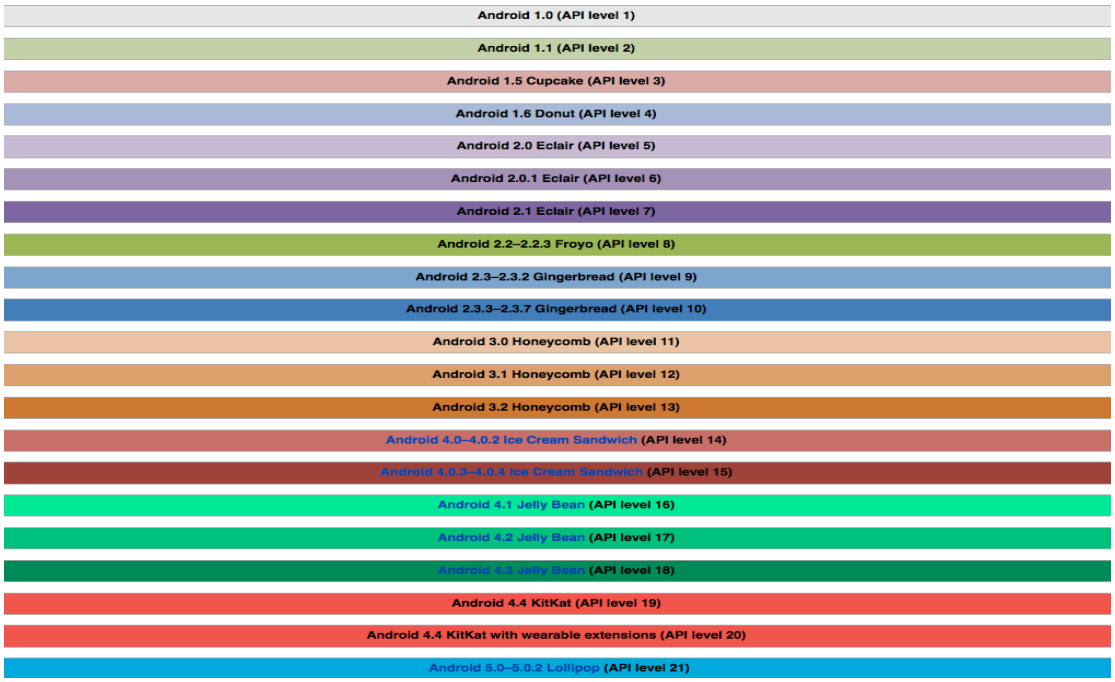
[1][2] Το Android είναι ένα λειτουργικό σύστημα που ενσωματώνεται σε συσκευές κινητής τηλεφωνίας που διαθέτουν οθόνη αφής και τρέχουν τον πυρήνα (kernel) του λειτουργικού Linux (κατεξοχήν γνωρίσματα). Επιπλέον επιτρέπει στους κατασκευαστές λογισμικού να συνθέτουν κώδικα με τη χρήση της γλώσσας προγραμματισμού Java, ελέγχοντας τη συσκευή μέσω βιβλιοθηκών λογισμικού της Google. Συσκευές με Android υπάρχουν πλέον πάρα πολλές, η καθεμία με διαφορετικά χαρακτηριστικά και από διάφορες κατασκευάστριες εταιρίες: η LG,

Samsung, HTC, Sony Ericsson, Motorola, είναι μερικές από τις εταιρίες που χρησιμοποιούν το λειτουργικό Android για τα έξυπνα τηλέφωνα τους.

Οι συσκευές Android υποστηρίζουν εφαρμογές πολυμέσων,οι οποίες έχουν την δυνατότητα να εκτελούνται σε παράλληλα νήματα. Αυτό έχει ώς αποτέλεσμα την υποστήριξη ταυτόχρονης εκτέλεσης πολλών εφαρμογών. Υποστηρίζεται γρήγορη πρόσβαση και περιήγηση στο διαδίκτυο. Επιπλέον υποστηρίζεται από λογισμικό γραφικών και υπάρχουν πολλοί Περιηγητές Διαδικτύου για να καλύψουν και τους πλέον απαιτητικούς.

Ανεξάρτητα από το κόστος, όλες οι συσκευές Android διαθέτουν GPS και Wi-fi, δικαιώνοντας έτσι το βασικό λόγο δημιουργίας του εν λόγω λειτουργικού συστήματος που δεν είναι άλλος παρά η ανεμπόδιστη και εύκολη πρόσβαση στο διαδίκτυο, σε συνδυασμό με ένα πλήθος εφαρμογών (apps), όπως χάρτες, αναζήτηση, chat και e-mail, που πραγματικά επιτρέπουν στο χρήστη να μένει διαρκώς δικτυωμένος και ενημερωμένος.

Βασικό χαρακτηριστικό του Android, επίσης, είναι η πληθώρα εφαρμογών που διατηρούν τη συνεχή σύνδεση με Facebook, Twitter ,Instagram και δεκάδες άλλες υπηρεσίες social networking. Ακόμη, το Android σας δίνει τη δυνατότητα να προσθέσετε widgets, δηλαδή εικονίδια για την ταχύτερη πρόσβαση στα προγράμματα, τα οποία τοποθετούνται στη home screen του κινητού (launcher).Επιπλέον η notification bar είναι εξαιρετικά χρήσιμη, καθώς με ένα απλό drag βλέπετε όλες τις ειδοποιήσεις για τη συσκευή σας, αλλά και τα προγράμματα (applications) που έχετε εγκαταστήσει.Όσον αφορά το hardware, οι τετραπύρηνοι επεξεργαστές και οι διακεκριμένες GPU είναι πλέον γεγονός. Παραθέτονται οι εκδόσεις του Android λογισμικού με χρονολογηκή σειρά:

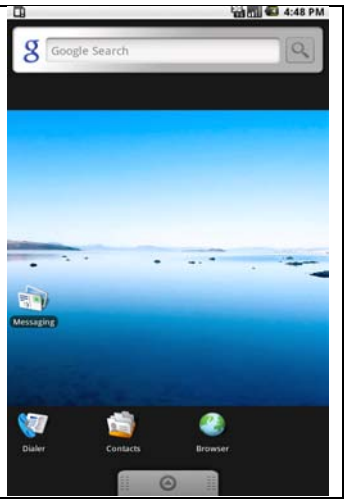


1.1 Android Version History (πηγή : Wikipedia – Android version history [2])

1.5 Εκδόσεις λογισμικού Android

[2] Όπως ήδη αναφέρθηκε το λειτουργικό σύστημα είναι ανοιχτού κώδικα.Το Android αποτελείται από μια μεγάλη κοινότητα προγραμματιστών που αναπτύσουν εφαρμογές οι οποίες επεκτείνουν την λειτουργικότητα των συσκευών. Η ανοιχτού κώδικα φύση του συντέλεσαι στην ραγδαία εξέλιξη του και αυτό αντικατοπτρίζεται στο γεγονός ότι από τον Απρίλιο του 2009 εώς τον Νοέμβριο του 2014 έχουν κυκλοφορήσει 10 κύριες εκδόσεις. Στην τεχνολόγια συνηθίζεται τα προϊόντα Υλικού και Λογισμικού να κυκλοφορούν με μία κωδική ονομασία εκτός απο τον αριθμό έκδοσης τους. Όμως οι εκδόσεις του Android βασίζουν το όνομα τους σε επιδόρπια και ακολουθούν αλφαβητική σειρα.

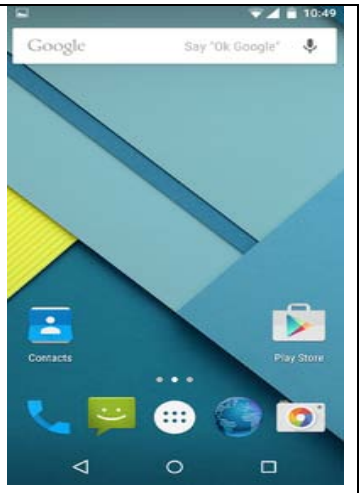
Η κυρίως πλατφόρμα υλικού για Android,είναι αρχιτεκτονικής ARM με υποστήριξη αρχιτεκτονικών x86 αλλά και MIPS επεξεργαστών. Με την τελευταία έκδοση 5.0 υποστηρίζονται πλεόν και τα 64-bit αλλά και 32-bit συστήματα διευθύνσεων.Παρακάτω παραθέτονται ορισμένες εκδόσεις Android λογισμικού:

Version 1.5 Cupcake (2009 features) - Android Market - Web browser (HTML, XHTML pages) - Gmail - Google Contacts, Calendar, Maps, Sync, Search,Talk - IM - Media Player - Youtube - Wi-fi - Support for Widgets - MPEG-4 - Ability to uploads videos to Youtube - Ability to upload photos to Picasa	

Εικόνα 1.2 (πηγή : Wikipedia – Android version history [2])

Version 3.0 Honeycomb (2011 features, for Motorola Xoom tablet) • "Holographic" user interface • Added System Bar • Simplified Multitasking • Ability to view albums in full-screen • Hardware acceleration • Support for multi-core processors • Ability to encrypt all user data • HTTP stack improved with Server Name Indication • Filesystem in Userspace(FUSE) • Application's write access to secondary storage	

Εικόνα 1.3 1η Έκδοση Android για Tablet(πηγή : Wikipedia – Android version history [2])

Version 5.0 Lollipop (November 2014) • Android Runtime with ahead-of-time compilation • Support 64-bit CPU • Vector drawables • Project Volta for battery life improvements • Lock Screen provides shortcuts to applications • Guest logins and multiple user accounts • Audio input/output through USB devices • Ability to read and modify data located anywhere on external storage • Updates through Google Play • Tap and Go allow users to quickly migrate to a new Android device, using NFC and Bluetooth to transfer Google Account Details and more.	

Εικόνα 1.2 (πηγή : Wikipedia – Android version history [2])

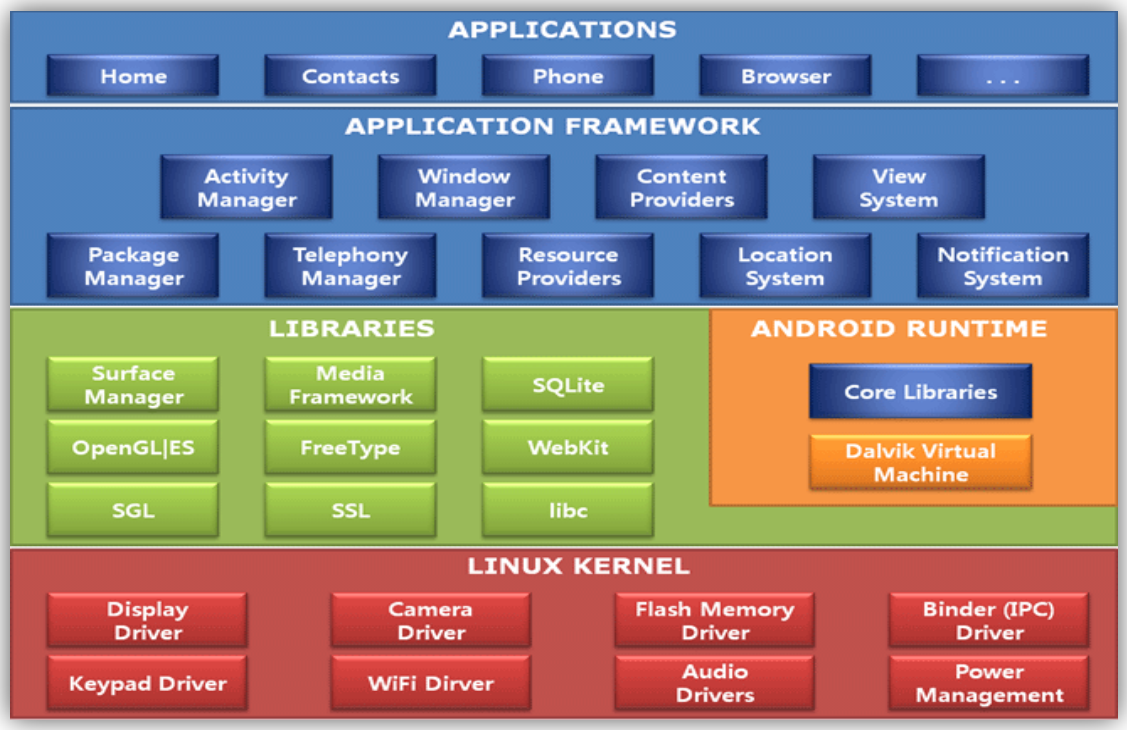
ΚΕΦΑΛΑΙΟ 2^{ο}

Οργάνωση λογισμικού Android και εφαρμογών

2.1 Η δομή του Android λογισμικού

[4] Για την δημιουργία εφαρμογής Android είναι απαραίτητη η εγκατάσταση του ADK (Android Development Kit) και η χρήση γλώσσας προγραμματισμού java. Η υλοποιήση του προγράμματος σε ποιό χαμηλό επίπεδο χρησιμοποιεί το NDK (Native Development Kit) για χρήση γλώσσας C.

Η εκτέλεση του λογισμικού πραγματοποιείται από την DVM (Dalvic Virtual Machine) η οποία λειτυργεί πάνω στον πυρήνα Linux. Παρακάτω απεικονίζονται όλα τα συστατικά μέρη από τα οποία αποτελείται το λειτουργικό σύστημα:



Εικόνα 2.1 Structure of Android (πηγή : Android,at a glance,[4])

2.1.1 Εφαρμογές (Applications)

[4] Στο υψηλότερο επίπεδο υπάρχουν οι εφαρμογές που έχουν αναπτυχθεί με Java. Σε γενικές γραμμές, e-mail client, ημερολόγιο, οι επαφές του προγράμματος περιήγησης και του χρήστη, οι περιηγητές διαδικτύου και όλες οι υπόλοιπες εφαρμογές που φορτώνονται.

2.1.2 Πλαίσιο Εφαρμογής (Application Framework)

[4] Στο δεύτερο σε ιεραρχία επίπεδο, υποστηρίζονται όλα τα είδη των APIs που είναι απαραίτητα για την ανάπτυξη μίας πλήρως εξοπλισμένης εφαρμογής. Χρησιμοποιώντας APIs σε αυτό τον τομέα, υλοποιείται η σύνδεση και έκφραση των κουμπιών και του κείμενου στην οθόνη καθώς και η χρήση άλλων εφαρμογών, όπως εικόνες και ακολουθίες χαρακτήρων. Οι προγραμματιστές έχουν πρόσβαση σε όλα τα APIs. Η δομή των εφαρμογών είναι τέτοια που ευνοείται η επαναχρησιμοποίηση δομικών συστατικών και επίσης επιτρέπεται η χρήση των δυνατοτήτων μίας εφαρμογής από άλλες. Σε αυτό το επίπεδο γίνεται διαχείριση και του κύκλου ζωής της εφαρμογής. Λεπτομέρειες αναφέρονται στη συνέχεια του κεφαλαίου. Τα σημαντικότερα δομικά μέρη του πλαισίου εφαρμογών είναι:

- **Σύστημα Προβολών (View System)** : Αποτελείται από ένα σύνολο αντικειμένων που αφορούν το γραφικό περιβάλλον χρήστη, με την βοήθεια των οποίων σχεδιάζεται μια εφαρμογή.
- **Πάροχος Περιεχομένου (Content Provider)** : Οι εφαρμόγες αποκτούν την δυνατότητα να μοιράζονται δεδομένα μιας συγκεκριμένης μορφής (Επαφές χρήστη,ΒΔ των εφαρμογών).
- **Διαχειριστής Πόρων (Resource Manager)** : Δίνεται πρόσβαση σε υλικό το οποίο δεν είναι σε μορφή κώδικα όπως εικόνες, αρχεία xml, πίνακες χαρακτήρων.
- **Διαχειριστής Ειδοποιήσεων (Notification Manager)** : Οι εφαρμογές αποκτούν πρόσβαση στις υπηρεσίες ειδοποιήσεων του χρήστη.
- **Διαχειριστής Δραστηριοτήτων (Activity Manager)** : Μέσω αυτού επιτυγχάνεται η διαχείρηση του κύκλου ζωής των δραστηριοτήτων και παρέχεται δυνατότητα πλοήγησης ανάμεσα στις δραστηριότητες. Λεπτομέρειες αναφέρονται στην συνέχεια του κεφαλαίου.

2.1.3 Βιβλιοθήκες

[4] Στο αμέσως επόμενο επίπεδο υποστηρίζονται βιβλιοθήκες C / C++. Όλες οι βιβλιοθήκες του τομέα μπορούν να χρησιμοποιηθούν από τους προγραμματιστές μέσω του πλαισίου εφαρμογής. Ενσωματώνονται και χρησιμοποιούνται από τις εφαρμογές για τις διάφορες προσφερόμενες λειτουργίες. Ουσιαστικά αποτελούν δομικό υλικό των εφαρμογών και αναπόσπαστο κομμάτι τους. Οι τυπικές C βιβλιοθήκες συστήματος που βασίζονται στο BSD (Berkeley Software Distribution) αναθεωρήθηκαν για να είναι κατάλληλες για συσκευές που βασίζονται σε Linux. Βιβλιοθήκη πολυμέσων, βασισμένη στο OpenCore του PacketVideo υποστηρίζει MPEG4, H.264, MP3, AAC, AMR, JPG και PNG αρχεία. Υποστηρίζονται 2D και 3D γραφικά για την διεπαφή χρήστη ενώ το WebKit υποστηρίζει τις λειτουργίες του προγράμματος περιήγησης. Σημαντικό στοιχείο αποτελεί η διαθεσιμότητα SQLite ως μιας μηχανή βάσης δεδομένων που μπορεί να χρησιμοποιηθεί από τις εφαρμογές.

2.1.4 Εικονική μηχανή Dalvik - Android Runtime

[4] Το λειτουργικό σύστημα Android έχει αναπτυχθεί σε γλώσσα Java, όμως χρησιμοποιεί την DVM (Dalvik Virtual Machine) αντί της JVM (Java Virtual Machine). Η Davlik λοιπόν είναι η εικονική μηχανή μέσω της οποίας εκτελούνται οι

εφαρμογές. Έτσι, ένα αρχείο με πηγαίο κώδικα (Java) μεταγλωτίζεται σε ένα αρχείο κλάσης (.class) με το μεταγλωττιστή της Java,ο οποίος αργότερα μετατρέπεται σε Dalvik εκτελέσιμο αρχείο (.dex) από εργαλεία DX. Η Dalvik έχει βελτιστοποιηθεί για λειτουργία μικρών συσκευών με περιορισμένη μνήμη. Επιπλέον υποστηρίζει εφαρμογές που εκτελούνται σε πολλά νήματα ταυτόχρονα και απασχολούν πολλές διεργασίες. Η κάθε εφαρμογή εκτελείται μέσω της δικής της εικονικής μηχανής, στη δική της διεργασία και έτσι δεν έχουν επαφή μεταξύ τους ενώ εκτελούνται ταυτόχρονα.

2.1.5 Πυρήνας Linux (Linux Lernel)

[4] Η ανάπτυξη του έγινε βασίζόμενη στον Linux kernel 2.6 ο οποίος παρέχει υψηλή ασφάλεια, διαχείριση της μνήμης, στοίβα δικτύου και πληθώρα οδηγών (drivers). Οι οδηγοί αυτοί είναι υπεύθυνοι για την επικοινωνία του υλικού με το λογισμικό της συσκευής. Ενδεικτικά περιεχει:

- Οδηγό προβολής οθόνης
- Οδηγό υποστήριξης Ασύρματης Πρόσβασης στο Διαδίκτυο
- Οδηγό για υποστήριξη Bluetooth επικοινωνίας
- Υλικό που υποστηρίζει NFC τεχνολογία.
- Οδηγό κάμερας που πλέον υποστηρίζει λειτουργίες που ξεπερνούν και την λειτουργία των φωτογραφικών μηχανών

Ο πυρήνας του Android μπορεί να βασίζεται στον πυρήνα του Linux, αλλά διαφέρει αρκετα. Έχουν υπάρξει σημαντικές αλλαγές στην αρχιτεκτονκή του με αποτέλεσμα να είναι ελαφρύτερος και βελτιστοποιημένος για χρήση σε συσκευές κινητής τηλεφωνίας.

2.2 Η δομή μιας εφαρμογής Android.

[5] Η δομή μιας εφαρμογής Android είναι αυστηρά καθορισμένη. Για να την σωστή λειτουργία της υπάρχει συγκεκριμένη ιεραρχία στον τρόπο οργάνωσης των φακέλων και των αρχείων που αποθηκεύονται σε αυτούς.

2.2.1 Πακέτα της Java και χρήση τους από το Android

[5] Ο όρος πακέτα στην Java περιγράφει ουσιαστικά μόνο φακέλους στους οποίους μπορούν να αποθηκευτούν οι κλάσεις Αυτό δίνει την δυνατότητα να παράγεται κώδικας που έχει μία καλώς οργανωμένη δομή, όπου οι κλάσεις που συνδέονται είναι στο ίδιο πακέτο και έχουν ονόματα που υποδηλώνουν τον σκοπό της ύπαρξης τους. Είναι σχεδόν αδύνατο να υλοποιηθεί μια εφαρμογή που δεν χρησιμοποιεί κλάσεις σε διαφορετικά πακέτα. Για παράδειγμα η Java ορίζει ένα πακέτο ανώτατου επιπέδου που ονομάζεται java και μέσα σε αυτό υπάρχουν πακέτα όπως το Lang, το οποίο περιέχει κλάσεις όπως η String και η util. Η util με την σειρά της περιέχει συλλογές κλάσεων, όπως η ArrayList. Παρομοίως το Android API περιέχει ένα πακέτο πρωταρχικό στο οποίο συμπεριλαμβάνονται πακέτα όπως graphics,view και widget.

Ο προγραμματιστής θέλοντας για παράδειγμα να χρησιμοποιήσει την ArrayList στον κώδικα του, θα πρέπει να δημιουργήσει αναφορά σε αυτήν χρησιμοποιώντας το πλήρες αναγνωρίσιμο όνομα της που περιλαμβάνει όλη την διαδρομή του πακέτου: *java.uti.ArrayList.* Η java παρέχει διευκολύνσεις με εντολές όπως η import:

Import java.util.ArrayList

Έτσι γράφοντας το όνομα *ArrayList* θα γίνεται αναφορά στην κλάση που βρίσκεται στο πακέτο *java.util* . Πλεονέκτημα αποτελεί η δυνατότητα ύπαρξης πολλαπλών κλάσεων με το ίδιο όνομα χωρίς να δημιουργείται σύγγρουση, με την προϋπόθεση να είναι σε διαφορετικά πακέτα.

Χρησιμοποιούνται πακέτα, όχι μόνο για την οργάνωση του κώδικα σε μια εφαρμογή, αλλά για την διαχείριση των ίδιων των κλάσεων των εφαρμογών. Απαιτείται από κάθε εφαρμογή, που είναι εγκατεστημένη σε μια συσκευή, να έχει ένα αναγνωριστικό πακέτου που θα έχει βάθος τουλάχιστο δύο επιπέδων. Για παράδειγμα, ένα πακέτο που ονομάζεται *mycoolapp* δεν θα ήταν κατάλληλο, αλλά ένα που ονομάζεται *mycompany.mycoolapp* θα ήταν αποδεκτό. Το πακέτο που περιέχει τον κώδικα μιας εφαρμογής χρησιμοποιείται για να προσδιορίσει ακριβώς την εφαρμογή στη συσκευή, καθώς και το να επιτρέπεται στις εφαρμογές να επικοινωνούν και να μοιράζονται πληροφορίες μεταξύ τους.

Μία ευρέως διαδεδομένη σύμβαση που χρησιμοποιείται είναι πως οι περισσότεροι προγραμματιστές Java δημιουργούν μια δομή πακέτων που βασίζεται σε ένα σύστημα αντιστροφής του *domain* όνοματος της εταιρείας τους. Για παράδειγμα σε μια εταιρεία που ονομάζεται CoolSoftware και έχει μία ιστοσελίδα coolsoftware.com, όλος ο κώδικας για τις εφαρμογές της εταιρείας θα μπορούσε να αποθηκευτεί σε υποπακέτα του com.coolsoftware. Αναπτύσοντας λοιπόν μία εφαρμογή που ονομάζεται AwesomeApp, είναι σύνηθες να αποθηκεύεται ο κώδικάς της στο com.coolsoftware.awesomeapp.

2.2.2 Κατάλογος αρχείων

[4][5] Όπως έχει ήδη αναφερθεί οι εφαρμογές αναπτύσονται σε Java. Ωστόσο, σε αντίθεση με την desktop έκδοση των εφαρμογών Java, τα αρχεία κλάσεων σε εφαρμογές Android δεν φορτώνονται απευθείας. Υπάρχουν συγκεκριμένα εργαλεία ανάπτυξης που συνδυάζουν τα αρχεία πόρων (resources) και τα αρχεία DEX(που έχουν προκύψει απο τα αρχεία κλάσης) για να δημιουργηθεί το αρχείο APK (.apk). Τα αρχεία APK είναι ο ο τύπος(format) αρχείων που χρησιμοποιούνται για την εγκατάσταση λογισμικού πάνω στο λειτουργικό σύστημα.

Παρακάτω αναφέρεται ο βασικός κατάλογος των φακέλων που αποτελούν την δομή για την ανάπτυξη μιας εφαρμογής Android και τα αρχεία που περιλαμβάνουν

/assets	Αρχεία δεδομένων που εγκαθίστανται στην συσκευή μαζί με την εφαρμογή. Περιέχει πολυμέσα όπως βίντεο, ήχους, μεγάλες εικόνες που δεν χρησιμοποιούνται άμμεσα από τα layouts.
/bin	Το εκτελέσιμο - μεταγλωτισμένο αρχείο της εφαρμογής
/gen	Αρχεία αναφοράς που δημιουργούνται αυτόματα απο το σύστημα Android. Δεν μπορούμε να μορφοποιήσουμε κάτι σε αυτά.
/res	Περιέχει άλλους φακέλους με πόρους για την εφαρμογή: φωτογραφίες,μενού και άλλα που χρησιμοποιούνται στην περιβάλλον χρήστη
/src	Πηγαίος Κώδικας της Εφαρμογής. Ο κώδικας που γράφεται από τον προγραμματιστή.
AndroidManifest.xml	Βασικό αρχείο της εφαρμογής. Λειτουργία παραπλήσιας με τα «Περιεχόμενα» ενός βιβλίου.

Πίνακας 2.1 (πηγή : Android at a glance [4])

Είναι απαραίτητη η αποθήκευση των αρχείων στο σωστό κατάλογο, ιδιαίτερα για τους πόρους. Από την στιγμή που οι κατάλογοι έχουν ήδη δεσμευθεί, αλλάζοντας τους κατά βούληση θα είναι δύσκολο να αναγνωριστούν καιαυτό οδηγεί σε μη-προβλέψιμα αποτελέσματα.

2.2.3 Resourse Directory

[4][5]Ο κατάλογος των πόρων(Resource Directory) αποτελείται από καταλόγους που επιτρέπεται η τοποθέτηση εικόνων (drawable) και από κατάλογους διάταξης (layouts) για την τοποθέτηση των αρχείων XML. Οι κατάλογοι raw και XML δεν δημιουργούνται αυτόματα από το project και έτσι θα πρέπει να δημιουργηθούν από τον χρήστη.

/res/drawable-*dpi	Περιέχει εικόνες που χρησιμοποιούνται σε διάφορα μέρη της εφαρμογής, όπως στην έναρξη της ή αυτές που προσάπτονται σε κουμπιά και μενού. Κάθε σχεδίαση μπορεί να έχει πολλαπλούς *drawable* φακέλους που παίρνουν το όνομα τους σύμφωνα με την ανάλυση της συσκευής. Για παράδειγμα ο drawable-ldpi θα περιέχει εικόνες που χρησιμοποιουνται σε χαμηλής ανάλυσης συσκευές και ο drawable-hdpi εικόνες για συσκευές υψηλής ανάλυσης. Αν δεν βρεθεί το ακριβές ταίριασμα ανάλυσης το λογισμικό θα αναζητήσει την πλησιέστερη.
/res/layout	Περιέχει XML αρχεία που καθορίζει τη διάταξη της διεπαφής του χρήστη όπως τα widgets (κουμπιά,πεδία κειμένου κτλ) στην

	οθόνη της εφαρμογής. Δεν χρειάζεται παραγωγή κώδικα για αυτά καθώς το εργαλείο ανάπτυξης παρέχει *drag-and-drop editor*.
/res/menu/	Περιέχει XML αρχεία που περιγράφουν τα μενού που σχετίζονται με τις οθόνες που εμφανίζονται στην εφαρμογή πατώντας ο χρήστης το κουμπί *Menu* .
/res/values/	Περιέχει "τιμές" που χρησιμοποιούνται στην εφαρμογή όπως συμβολοσειρές και τις μορφοποιήσεις τους. Η κύρια χρήση του είναι το *string.xml* το οποίο χρησιμοποιείται για την αποθήκευση των συμβολοσειρών κειμένου που χρησιμοποιούνται στην διεπαφή.
/res/xml/	Περιέχει πληθώρα αρχείων αποθηκευμένα σε μορφή XML.
/res/raw/	Περιέχει διάφορα αρχεία (mp3,mp4 κτλ.)

Πίνακας 2.2 (πηγή : Android at a glance [4])

Στην οθόνη της συσκευής οι λειτουργίες για κάθετη και οριζόντια απεικόνιση διαφέρουν σε μέγεθος.Για να εμφανίζονται σωστά και οι δύο λειτουργίες, είναι απαραίτητη η δημιουργία του αρχείου διατάξεως της οθόνης στον κατάλογο /res/layout/ . Για να φαίνεται η διαφορά δημιουργείται το αρχείο διατάξεως (layout) στους καταλόγους layout-port και στο layout-land. Συχνά ίσως χρειάζεται μόνο ένα αρχείο διάταξηςκαι το layout-land. Στη συνέχεια, τα αρχεία διάταξης εφαρμόζονται στην λειτουργία κάθετης διάταξης(vertical mode layout).

2.2.4 Result Directory

[5] Η δομή του καταλόγου bin και των περιεχόμενων τους είναι η εξής :

bin/classes/	Τα μεταγλωτισμένα αρχεία κλάσεων της Java
bin/classes.dex	Αρχείο κλάσης Dalvik το οποίο δημιουργείται απο την μεταγλώτιση των κλάσεων της Java
bin/resources.ap_	Όλα τα αρχεία πόρων της εφαρμογής (συμπιεσμένη μορφή)
bin/app_name.apk	Η συμπιεσμένη μορφή του τελικού αρχείου της εφαρμογής

Πίνακας 2.3 (πηγή: Structure of Android app [5])

Τα αρχεία με κατάληξη .apk είναι τα τελικά, δηλαδή, αυτά τα οποία χρησιμοποιούνται για την εγκατάσταση της εφαρμογής στο PlayStore. Η συμπιεσμένη μορφή αυτού του αρχείου είναι zip, το οποίο μας επιτρέπει την εύκολη

πρόσβαση σε αυτά μέσω προγραμμάτων συμπίεσης. Ωστόσο το άνοιγμα αυτών των αρχείων δεν βοηθάει στην κατανόηση λεπτομερειών καθώς τα πηγαία αρχεία έχουν ήδη μεταγλωτιστεί.

2.2.5 AndroidManifest.xml

[5] Το συγκεκριμένο αρχείο είναι πολύ σημαντικό, καθώς περιέχει όλες τις πληροφορίες σχετικά με όλη την δομή της εφαρμογής. Μέσα από αυτό το αρχείο, δηλώνονται είδη δομικών συστατικών (components) και φίλτρων intent ενώ στη συνέχεια ελέγχονται οι αρμοδιότητες.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.androidside"
android:versionCode="1"
android:versionName="1.0">
<application android:icon="@drawable/icon"
android:label="@string/app_name">
<activity android:name=".HelloWorld"
android:label="@string/app_name">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>
<uses-sdk android:minSdkVersion="9" /> <!-- android 2.3 -->
</manifest>
```

Εικόνα 2.2 Παράδειγμα αρχείου AndroidManifest.xml (πηγή: Structure of Android app [5])

Το συγκεκριμένο αρχείο μπορεί να ορίσει τα παρακάτω στοιχεία :

<user-permission />	Η εξουσιοδότηση που απαιτείται απο την Εφαρμογή
<permission />	Η εξουσιοδότηση που απαιτείται για εξωτερική πρόσβαση
<instrumentation />	Προσδιορίζει τι πρέπει να καλέσει η εφαρμογή κατά την διάρκεια σημαντικών γεγονότων όπως η λειτουργία δραστηριότητας(*Activity*).
<uses-library />	Υλοποιεί επιπρόσθετες απαιτούμενες βιβλιοθήκες (Google maps)
<uses-sdk />	Καθορίζει την έκδοση Android που χρειάζεται για την εφαρμογή
<application />	Περιέχει πληροφορίες σχετικά με την εφαργμογή

Πίνακας 2.4 (πηγή :Structure of Android app [5])

2.3 Στοιχεία(elements) και δομή μιας εφαρμογής

[5] Υπάρχουν 4 δομικά συστατικά στις εφαρμογές Android:

- Activity
- Service
- Broadcast Receiver
- Content Provider

Η συνδεση μεταξύ αυτών επιτυγχάνεται με την δημιουργία Intent(Υποδηλώνει την έναρξη μιάς δραστηριότητας). Ώς δραστηριότητα ερμηνεύεται η λειτουργία ***Activity.*** Παρακάτω βλέπουμε βασικές έννοιες και τα τέσσερα δομικά συστατικά

Component	Δομικό συστατικό.
Activity	Δομικό συστατικό για την σύνθεση της διεπαφής χρήστη.
Service	Η εκτελεσή του γίνεται στο παρασκήνιο. Δεν υπάρχει οπτικοποίηση στην διεπαφή χρήστη
Intent	Λειτουργέι ώς τύπος μηνύματος που στέλνει οδηγίες λειτουργίας και τα δεδομένα στα συστατικά συστατικά
Intent Filter	Καθορίζει τους κανόνες για ένα δομικό συστατικό που έχει εκδηλώσει πρόθεση λήψης οδηγιών λειτουργίας και δεδομένων
Broadcast Receiver	Είναι υπεύθυνο για την λήψη / ανταπόκριση σε κάποια συγκεκριμένη εκπομπή όπως η έλλειψη μπαταρίας, ρύθμιση γλώσσας. Δεν υπάρχει οπτικοποίηση του στην διεπαφή χρήστη
Content Provider	Πραγματοποιεί παροχή τυποποιημένης διεπαφή για την ανταλλαγή δεδομένων μεταξύ των εφαρμογών
Notification	Λειτουργία κοινοποίησης ενός συγκεκριμένου συμβάντος στον χρήστη.

Πίνακας 2.5 (πηγή :Structure of Android app [5]))

2.4 Λογισμικό MVC (Model – View - Controller)

[6] Υπάρχει μια πληθώρα δομικών συστατικών για την ανάπτυξη λογισμικού σε περιβάλλον Android, αλλά για την ανάπτυξη εφαρμογών γίνεται χρήση μόνο αυτών που μπορούν να περιγραφούν στην δομή **MVC**. Είναι ένα λογισμικό αρχιτεκτονικού προτύπου για την υλοποίηση των διεπαφών χρήστη. Χωρίζει μια εφαρμογή σε τρία διασυνδεδεμένα μέρη, έτσι ώστε να διαχωρίζονται οι εσωτερικές αναπαραστάσεις πληροφοριών από τον τρόπο με τον οποίο οι πληροφορίες παρουσιάζονται στον χρήστη ή εισάγονται από αυτόν.

i) View

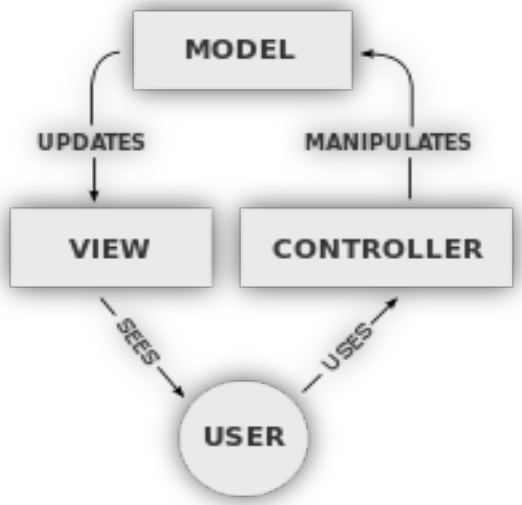
Χρησιμοποιείται για την παρουσίαση πληροφοριών στην οθόνη και το χρήστη. Οι πληροφορίες αυτές λαμβάνονται από τον τομέα Model. Μπορεί να δημιουργηθεί χρησιμοποιώντας την κλάση που κληρονομεί την View κλάση που παρέχεται από το Android.

ii) Control

Η χρήση αυτού του τομέα είναι η διασύνδεση και ο έλεγχος του View και του Model. Περιλαμβάνει δομικά συστατικά όπως τα Activity, Service και BroadcastReceiver.

iii) Model

Σε αυτή την περιοχή γίνεται η αποθήκευση των δεδομένων της εφαρμογής. Περιλαμβάνονται η SQLite η οποία μπορεί να χρησιμοποιηθεί για την ανταλλαγή δεδομένων μεταξύ των εφαρμογών



Εικόνα 2.3 (πηγη : Wikipedia Model-view-controller [6])

2.5 Κύκλος Ζωής εφαρμογών

Το σύστημα χειρίζεται τις δραστηριότητες (Activity) ως μια στοίβα δραστηριοτήτων. Όταν μία δραστηριότητα έχει ξεκινήσει, τοποθετείται στην κορυφή της στοίβας και πλέον είναι η δραστηριότητα που εκτελείται. Η προηγούμενη δραστηριότητα παραμένει πάντα κάτω στην στοίβα και δεν επανέρχεται στο προσκήνιο όσο υπάρχει η νέα.

[8] Καθώς ο χρήστης εισερχεται/εξέρχεται στην εφαρμογή, η κλάση που ορίζει την δραστηριότητα στην εφαρμογή, μεταβαίνει ανάμεσα σε ορισμένες

καταστάσεις που χαρακτηρίζουν τον κύκλο ζωή της. Για παράδειγμα, όταν μία δραστηριότητά εκτελείται για πρώτη φορά, έρχεται στο προσκήνιο του συστήματος και είναι αυτή που δημιουργεί ένα παράθυρο για την αλληλεπίδραση με τον χρήστη. Κατά τη διάρκεια αυτής της διαδικασίας, το σύστημα Android καλεί μια σειρά από μεθόδους του κύκλου ζωής, με τις οποίες καθορίζειται το γραφικό περιβάλλον του χρήστη. Εάν ο χρήστης εκτελέσει μια ενέργεια που εκκινεί μια άλλη δραστηριότητα ή μεταβεί σε άλλη εφαρμογή, το σύστημα καλεί ένα άλλο σύνολο μεθόδων κύκλου ζωής, για τη δραστηριότητά που ήταν σε εκτέλεση, καθώς αυτή πλέον λειτουργεί στο παρασκήνιο (κατάσταση που η δραστηριότητα δεν είναι πλέον ορατή, αλλά η παρουσία και η κατάσταση του παραμένει άθικτη).

Εντός των μεθόδων του κύκλου ζωής επανάκλησης, καθορίζεται το πώς η δραστηριότητα θα συμπεριφέρεται, όταν ο χρήστης εγκαταλείπει την λειτουργία της και επανέρχεται αργότερα. Για παράδειγμα, εάν εκτελείται μια εφαρμογή αναπαραγωγής βίντεο, οδηγείται το βίντεο σε παύση και τερματίζεται η σύνδεση με το δίκτυο, όταν ο χρήστης μεταβαίνει σε άλλη εφαρμογή. Όταν ο χρήστης επιστρέψει, θα επανασυνδεθεί στο δίκτυο η εφαρμογή και ο χρήστης θα έχει την δυνατότητα να συνεχίσει την αναπαραγωγή του βίντεο από το ίδιο σημείο.
Μία δραστηριότητα-εφαρμογή έχει ουσιαστικά τέσσερις καταστάσεις:

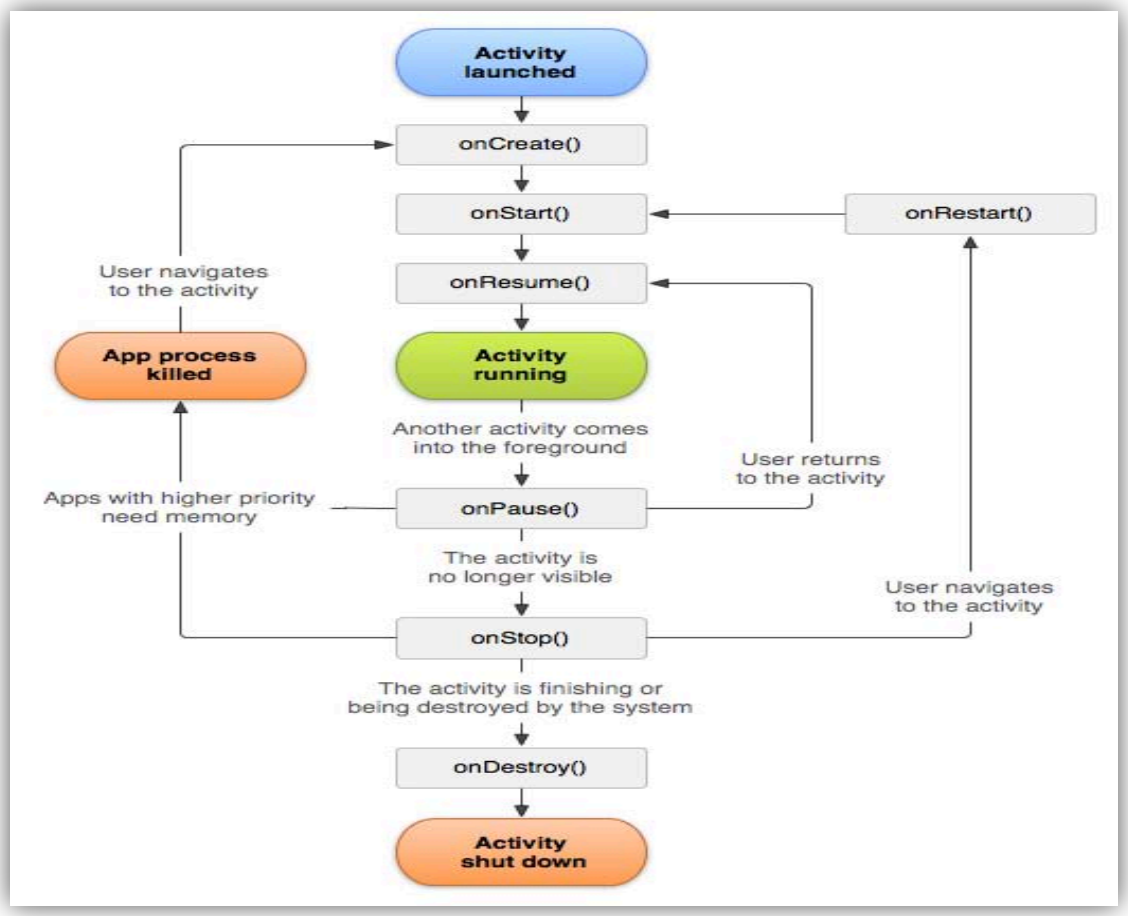
- Αν βρίσκεται στο προσκήνιο της οθόνης (κορυφή της στοίβας) σημαίνει πώς είναι ενεργή και εκτελείται.

- Αν έχει χαθεί η κεντρική εστίαση της, η παρουσία της χαρακτηρίζεται από ημιδιαφάνη κατάσταση και σημαίνει πώς βρίσκεται σε παύση. Διατηρεί όλα τα δεδομένα της πρόσφατης χρήσης αλλά μπορεί να τερματιστεί απο το σύστημα σε περίπτωση έλλειψης μνήμης.

- Αν έχει επισκιαστεί από άλλη δραστηριότητα και δεν είναι καθόλου ορατή στον χρήστη σημαίνει πώς παρόλου που διατηρεί τις τρεχουσες καταστάσεις και τις πληροφορίες της, έχει τερματιστεί.

- Αν γίνεται προσωρινή ή οριστική διακοπή, το σύστημα μπορεί να αποδεσμεύσει την εφαρμογή από την μνήμη ζητώντας τον τερματισμό της ακαριαία. Για να εμφανιστεί και πάλι στον χρήστη θα πρέπει γίνει επανέναρξη της για να επανέλθει στην προηγούμενη κατάσταση.

Το παρακάτω διάγραμμα δείχνει τα σημαντικά μονοπάτια καταστάσεων μιάς δραστηριότητας. Τα ορθόγωνια σχήματα αναπαριστούν μεθόδους επανάκλησης που μπορούν να εφαρμοστούν για να εκτελεσθούν λειτουργίες όταν η δραστηριότητα μεταβαίνει μεταξύ των καταστάσεων. Τα οβάλ σχήματα αποτελούν τις τέσσερις αποδεκτές καταστάσεις δραστηριότητας.



Εικόνα 2.4 (πηγή : javatpoint.com [7])

Ο κύκλος ζωής ελέγχεται από 7 μεθόδους της κλάσης android.app.Activity. Η δραστηριότητα είναι μία υποκλάση της ContextThemeWrapper. Είναι μια ενιαία οθόνη στην εφαρμογή. Εμφανίζεται σαν παράθυρο ή πλαίσιο της Java. Με τη βοήθεια της δραστηριότητας, τοποθετούνται όλα τα στοιχεία ή τα widgets του περιβάλλοντος του χρήστη σε μία μόνο οθόνη[7].

Οι 7 μέθοδοι του κύκλου ζωής περιγράφουν πώς η δραστηριότητα θα συμπεριφερθεί σε διάφορες καταστάσεις [7]:

- **onCreate** : Καλείται στην δημιουργία της δραστηριότητας.
- **onStart :** Καλείται όταν η δραστηριότητα γίνεται ορατή στον χρήστη.
- **onResume** : Γίνεται κλήση της όταν ο χρήστης αλληλεπιδρά με την εφαρμογή
- **onPause** : Καλείται όταν η εφαρμογή δεν είναι ορατή στο χρήστη.
- **onStop** : Κλήση γίνεται όταν δεν είναι ορατή η εφαρμογή απο τον χρήστη και έχει σταματήσει οποιαδήποτε αλληλεπίδραση.

- **onRestart** : Καλείται όταν ανοίξει ξανά η εφαρμογή και τα τελευταία δεδομένα και πληροφορίες είναι διαθέσιμα πάλι

- **onDestroy** : Καλείται πρίν την οριστική διακοπή είτε μετά από αίτημα του χρήστη ή λόγω έλλειψης μνήμης.

Method	Description	Killable?	Next
`onCreate()`	Called when the activity is first created. This is where you should do all of your normal static set up: create views, bind data to lists, etc. This method also provides you with a Bundle containing the activity's previously frozen state, if there was one. Always followed by `onStart()`.	No	`onStart()`
`onRestart()`	Called after your activity has been stopped, prior to it being started again. Always followed by `onStart()`	No	`onStart()`
`onStart()`	Called when the activity is becoming visible to the user. Followed by `onResume()` if the activity comes to the foreground, or `onStop()` if it becomes hidden.	No	`onResume()` or `onStop()`
`onResume()`	Called when the activity will start interacting with the user. At this point your activity is at the top of the activity stack, with user input going to it. Always followed by `onPause()`.	No	`onPause()`
`onPause()`	Called when the system is about to start resuming a previous activity. This is typically used to commit unsaved changes to persistent data, stop animations and other things that may be consuming CPU, etc. Implementations of this method must be very quick because the next activity will not be resumed until this method returns. Followed by either `onResume()` if the activity returns back to the front, or `onStop()` if it becomes invisible to the user.	**Pre-**`HONEYCOMB`	`onResume()` or `onStop()`
`onStop()`	Called when the activity is no longer visible to the user, because another activity has been resumed and is covering this one. This may happen either because a new activity is being started, an existing one is being brought in front of this one, or this one is being destroyed. Followed by either `onRestart()` if this activity is coming back to interact with the user, or `onDestroy()` if this activity is going away.	**Yes**	`onRestart()` or `onDestroy()`
`onDestroy()`	The final call you receive before your activity is destroyed. This can happen either because the activity is finishing (someone called `finish()` on it, or because the system is temporarily destroying this instance of the activity to save space. You can distinguish between these two scenarios with the `isFinishing()` method.	**Yes**	*nothing*

Εικόνα 2.5 (πηγή : developer.android.com [8])

ΚΕΦΑΛΑΙΟ 3^{Ο}

Τεχνολογία NFC

3.1 Τι είναι το NFC;

Η επικοινωνία κοντινού πεδίου (NFC: Near Field Communication) αποτελεί μία πρότυπη τεχνολογία συνδεσημότητας, η οποία διαδίδεται και εξελίσεται ραγδαία με κύριο σκοπό τη λύση αρκετων προβλημάτων, σύγχρονων αλλά και μελλοντικών. Η συγκεκριμένη τεχνολογία επιτρέπει σε συσκευές να ανταλλάσσουν πληροφορίες μεταξύ τους, με μόνη προϋπόθεση να βρίσκονται σε επαφη ή να είναι σε πολύ κοντινή απόσταση. Επίσης συνδιάζει και την χρήση συγκεκριμένων ετικετών (NFC tags) με συσκευές που υποστηρίζουν την συγκεκριμένη τεχνολογία.Υπάρχουν πολλές πρακτικές εφαρμογές που είναι ήδη διαθέσιμες σε μερικές χώρες ενώ το NFC είναι ένα «ανοικτό πρότυπο» κάτι που σημαίνει πως αποτελεί πρόκληση για την δημιουργία καινοτόμων εφαρμογών με την χρήση της συγκεκριμένης τεχνολογίας.

Η αρχική έμπνευση του αφορούσε μεταφορά δεδομένων που θα γινόταν από συσκευή σε συσκευή κάτι το οποίο θα άλλαζε τον τρόπο που χρησιμοποιούνται οι συσκευές για επικοινωνία. Η βάση για αυτό το εγχείρημα ήταν η μεγιστοποίηση της συμβατότητας μεταξύ των συσκευών. Ορισμένες εταιρείες που χρησιμοποιούν Android λογισμικό στις συσκευές τους παρέχουν πιο αναβαθμισμένες υπηρεσίες από αυτές του πρωτότυπου Android. Ωστόσο, υπάρχει ένα πολύ βασικό μειονέκτημα. Λειτουργούν μόνο με άλλες συμβατές συσκευές της ίδιας εταιρείας περιορίζοντας την χρησιμότητα τους.

Σε σύγκριση με τις υπόλοιπες ασύρματες τεχνολογίες, παρόλο που υστερεί σημαντικά σε εμβέλεια, έχει ένα σημαντικό πλεονέκτημα. Δεν χρειάζεται καμία άλλη ενέργεια για να εκκινηθεί η ανταλλαγή πληροφοριών. Η σύζευξη μιας συσκευής που υποστηρίζει NFC με μία άλλη σε μικρή απόσταση είναι αρκετή για να εμφανίστεί αυτόματα και χωρίς την παραμικρή κίνηση του χρήστη η σχετική ενημέρωση στην οθόνη με τις πληροφορίες που υπάρχουν και το πώς μπορεί να αλληλεπιδράσει η μία με την άλλη.

Το NFC δεν μεταφέρει δεδομένα. Αποστέλλει πολύ μικρές, βασικές πληροφορίες σε πολύ μικρές αποστάσεις. Για παράδειγμα δεν χρησιμοποιείται για την αποστολή ενός βίντεο αλλά μπορεί να ορίσει την έναρξη λειτουργίας μεταφοράς δεδομένων ώστε να πραγματοποιηθεί η μεταφορά του αρχείου, αλλά η αυτούσια μεταφορά πραγματοποιείται μέσω Wi-Fi Direct ή Bluetooth.

Ευρέως διαδεμένα είναι πλέον τα NFC tags τα οποία προγραμματίζονται για ενεργοποιούν συγκεκριμένες λειτουργίες της συσκευής ή και εφαρμογών. Μέσω του NFC μπορεί να μεταδοθεί και ενέργεια. Αυτή την ιδιότητα εκμεταλεύεται η λειτουργία των ετικέτων. Είναι μικρές πλακέτες που υπάρχουν σε μορφές όπως χάρτινες κάρτες, αυτοκόλλητα και μπρελόκ. Στο εσωτερικό τους βρίσκεται ένα NFC Chip στο οποίο μπορεί να αποθηκευτεί πληροφορία για κάτι συγκεκριμένο. Υπάρχουν 4 ειδών ετικέτες. Η διαφοροποίηση τους έγγυται στην χωριτικότητα τους και την ταχύτητα μετάδοσης δεδομένων που υποστηρίζουν. Η μικρότερη

χωριτικότητα έιναι 96 bytes με ταχύτητα μετάδοσης 106kbps ενώ η μεγαλύτερη έιναι 32kb με ταχύτητα μετάδοσης 424 kbps.

3.2 Τεχνικά Χαρακτηριστικά

Η τεχνολογία NFC (Near Field Communications) αφορά την ασύρματη μετάδοση δεδομένων. Επιτρέπει την γρήγορη ανάγνωση/εγγραφή δεδομένων (σχετικά λίγων- ενδεικτικά 48B – 9kB) και εκλαμβάνεται ως απόδειξη φυσική παρουσίας. Επίσης, μπορεί να ενεργοποιήσει εύκολα υπηρεσίες.

- Το πρωτόκολλο επικοινωνίας που επιλέχτηκε ήταν η συχνότητα 13.56MHz (στην οποία λειτουργεί και το RFID). Το NFC πληροί τις προδιαγραφές των στάνταρτ ISO/IEC 14443 A & B, και Felica (ISO 18092).
- Η ταχύτητα διαμεταγωγής δεδομένων μπορεί να είναι 106 kbps, 212 kbps ή 424 kbps. Αυτή η ταχύτητα αναμένεται να αυξηθεί μελλοντικά.
- Απόσταση λειτουργίας: Η απόσταση που μπορούν να αλληλεπιδράσουν δύο NFC συσκευές είναι περίπου τα 10 εκατοστά και χρειάζονται περίπου 0.2 δευτερόλεπτα για να γίνει η σύνδεση (η ταχύτητα που ανοιγοκλείνουμε τα μάτια μας είναι περίπου η ίδια).
- Κατανάλωση ενέργειας: Η κατανάλωση είναι μόλις στα 15mA (περίπου όση χρειάζεται ένα LED φωτάκι για να ανάψει) αν και ενδέχεται να είναι περισσότερη όταν υπάρχει εγγραφή δεδομένων.
- Εκτός από τα κινητά όμως υπάρχουν και NFC ετικέτες και κάρτες (εσωτερικού ή εξωτερικού χώρου, σε μπρελόκ, αυτοκόλλητα, μινιατούρες, κ.λπ.) με διαφορετική χωρητικότητα ή υπό μορφή έξυπνων καρτών σε σχήμα πιστωτικής ή σε άλλες υποδομές contactless (Contactless POS, Ticketing stands, Kiosks, κ.ά.)

Η τεχνολογία αναπτύσσεται και προωθείται κυρίως από το NFC Forum στο οποίο συμμετέχουν 140 εταιρίες και από άλλους οργανισμούς, όπως GSMA, GlobalPlatform και EMVCo.

Επιπλέον, ένα NFC μπορεί να βρίσκεται σε 3+1 διαφορετικές καταστάσεις λειτουργίας: Μία λειτουργία αφορά επικοινωνία Εγγραφής/Ανάγνωσης (το ένα Ενεργό και το άλλο Παθητικό), ενώ μια δεύτερη αποτελεί η περίπτωση που οι δύο συσκευές έχουν ομότιμη σχέση (Peer-to-peer, αμφότερες ενεργές, όπως για παράδειγμα μεταξύ δύο κινητών τηλεφώνων με NFC τσιπ).Τέλος μια ακόμη λειτουργία ειναι ως Card emulation.

Τα μηνύματα τα οποία μεταδίδουν ονομάζονται μηνύματα NDEF και είναι τα εξής:

- Smart Poster (για την ανάγνωση επιπλέον πληροφορίας από διαφημιστικά πόστερ)
- Handover (για παράδειγμα, την άμεση σύνδεση δύο συσκευών bluetooth με το άγγιγμά τους)
- vCard (μεταφορά στοιχείων επαφών υπό μορφή vCard)
- URL (παραπομπή σε ιστοσελίδα)
- SMS

3.3 Εξυπηρέτηση αναγκών και πλεονεκτήματα χρήσης

[12]Η τεχνολογία NFC παρουσιάζει συγκεκριμένα πλεονεκτήματα ως προς τις ήδη υπάρχουσες μορφές απομακρυνσμένης σύνδεσης. Αυτά είναι η ασφάλεια και η χρηστικότητα. Το Bluetooth και το Wi-Fi έχουν μεγαλύτερο εύρος σύνδεσης από το NFC. Αυτό σημαίνει πως υπάρχει μία θεωρητική ακτίνα που μπορούν να συνδεθούν δύο συσκευές η οποία ποικίλει,από ένα μέχρι εκατό μέτρα. Αυτό δημιουργεί δύο προβλήματα:

- Εάν οι συσκευές είναι ρυθμισμένες έτσι ώστε να δέχονται συνδέσεις χωρίς κάποιο κωδικό, ο οποιοσδήποτε μπορεί να συνδεθεί και να αντλήσει πληροφορίες δημιουργώντας έτσι σημαντικά κενά ασφαλείας.
- Χρησιμοποιώντας κωδικούς προκειμένου να παρακαμφθεί το πρώτο πρόβλημα τότε αυτομάτως οι χρήστες θα πρέπει να πλοηγηθούν στη συσκευή τους, και να εισάγουν τον σωστό κωδικό κάτι που σημαίνει ότι χάνεται ο παράγοντας «ευκολία».

Το NFC εξαλείφει και τα δύο αυτά προβλήματα απαιτώντας από τις δύο συσκευές να βρίσκονται αρκετά κοντά, ενώ με το που πραγματοποιηθεί η σύνδεση εμφανίζονται αυτόματα οι επιλογές στην οθόνη της συσκευής προκειμένου ο χρήστης να επιλέξει την επιθυμή λειτουργία εύκολα και γρήγορα.

Η χρήση της συγκεκριμένης τεχνολογίας παρουσιάζει πλεονεκτημάτα, μερικά από τα οποία περιγράφονται παρακάτω:

1) Είναι εύχρηστη, σχεδόν αυτονόητη (με ένα «άγγιγμα» γίνονται όλα)

2) Η ανάγκαιότητα της επαφής των συσχετιζόμενων συσκευών είναι ο καλύτερος τρόπος επιβεβαίωσης φυσικής παρουσίας του ατόμου που το χρησιμοποιεί

- Δηλώνεται η παρουσία του χρήστη.
- Δηλώνεται η θέληση για εκτέλεση συγκεκριμένης ενέργειας.
- Είναι ανέφικτη η υποκλοπή δεδομένων ασύρματα (σε αντίθεση με π.χ. Wi-Fi, RFID, κ.λπ)

3) Δίνεται λύση σε σενάρια χρήσης όπως:

- Πληρωμές
- Εισιτήρια
- Διαφήμιση
- Ανταλλαγή δεδομένων
- Καταγραφή παρουσίας / Έλεγχος πρόσβασης

4) Τα κινητά τηλέφωνα αξιοποιούνται ως μέσο αλληλεπίδρασης:

Δεδομένου ότι είναι ευρέως διαδεδομένα και αποτελούν αναπόσπαστο κομμάτι της καθημερινότητας, ο συνδιασμός τους με την συγκεκριμένη τεχνολογία μπορεί να υπερκαλύψει λειτουργίες και αντικείμενα όπως τα εισητήρια, το πορτοφόλι και τις κάρτες επισκεπτηρίων. Επιπλέον οι νέες συσκευές έχουν χαρακτηριστικά κατασκευής ισοδύναμα με αυτά των προσωπικών υπολογιστών. Έτσι μπορεί να

επιτευχθεί υποστηρίξη πολύπλοκων υπολογισμών, κρυπτογράφησης, κ.λπ. Με την δεδομένη σύνδεση τους στο internet επιταχύνεται η άντληση περισσότερων πληροφοριών άμεσα, χωρίς αναμονή. Επιπλέον το περιβάλλον χρήστη που προσφέρουν διευκολύνει την έγγριση / απόριψη μιας αλληλεπίδρασης.

Εικόνα 3.1 (πηγή: andro4all.com)

3.4 Εφαρμογές στην καθημερινότητα

Η χρήση του NFC συνεχώς διευρύνεται και πλέον εφαρμόζεται σε όλο και περισσότερς περιπτώσεις. Αποτελεί μάλιστα σημαντική λύση στην ψηφιοποίηση υπηρεσιών[14].

Ως RFID Scanner

Μία συσκευή με NFC μπορεί να λειτουργήσει ως ένα RFID Tag Scanner διαβάζοντας πληροφορίες που βρίσκονται με τη μορφή RFID tags (μικρά τσιπάκια λίγων χιλιοστών που περιέχουν πληροφορίες και που βρίσκονται οπουδήποτε υπό μορφή αυτοκόλλητου) σε αφίσες, φυλλάδια, διαφημίσεις και άλλα μέσα προβολής. Σε ένα εστιατόριο για παράδειγμα, είναι εφικτό για τους πελάτες να εμφανίζονται αυτομάτως στην οθόνη της συσκευής τους,με διαδραστικό τρόπο, τα πιάτα του εστιατορίου, τα σπέσιαλ πίατο της ημέρας και βίντεο με την παρασκευή του κάθε πιάτου. Επιτυγχάνεται με την προσέγγιση της συσκευής τους στο μενού του εστιατορίου.

Ως κάρτα πληρωμών

Οι πληρωμές πλέον διευκολύνονται μέσω του NFC, και σταδιακά αντικαθιστούν τις χρεωστικές και πιστωτικές κάρτες. Ο τρόπος χρήσης είναι απλός. Μόλις οι πληροφορίες της κάρτας μεταφερθούν στη συσκευή (υπεύθυνη για αυτή τη διαδικασία είναι η τράπεζα), τότε με ένα απλό άγγιγμα του smartphone σε κατάλληλα διαμορφωμένη συσκευή, πραγματοποιείται πληρωμή μέσω τραπεζικού λογαριασμού. Για παράδειγμα στην Ελλάδα τους τελευταίους μήνες χρησιμοποιείται η συγκεκριμένη λειτουργία από εταιρείες κινητής τηλεφωνίας παρέχοντας υπηρεσίες Tap’nPay σε συνεργασία με συγκεκριμένα τραπεζικά καταστήματα.

Ως επαγγελματική κάρτα

Η τεχνολογία NFC έχει ήδη αρχίσει να αντικαθιστά τις επαγγελματικές κάρτες. Με την σύζευξη δύο συσκευών, μπορεί αμέσως να πραγματοποιηθεί μεταφορά δεδομένων της μίας συσκευής στην άλλη. Με αυτό τον τρόπο, τα δεδομένα που αφορούν επαγγελματικά χαρακτηριστικά ενός ατόμου,αποθηκεύονται στους ενδιαφερόμενους χωρίς να χρειάζεται τίποτα περισσότερο, άμεσα, εύκολα και γρήγορα.

Σύνδεση με Bluetooth και Wi-Fi

Το NFC χρησιμοποιείται και σε συνδυασμό με τις συνδέσεις Bluetooth και Wi-Fi. Φέρνοντας κοντά τις δύο συσκευές , αυτομάτως εμφανίζεται η επιλογή στην οθόνη και με ένα μόνο άγγιγμα οι συσκευές αυτές ενώνονται. Έπειτα μπορεί να ακολουθήσει η μεταφορά δεδομένων μέσω Wi-fi ή Bluetooth.

Εικόνα 3.2 (πηγή: andro4all.com)

3.5 Μελλοντικές εφαρμογές

Εκτός από τις παραπάνω εφαρμογές του NFC οι οποίες είναι ήδη διαθέσιμες, η τεχνολογία έχει μόλις αρχίσει να δείχνει τις πραγματικές της δυνατότητές. Παρουσιάζονται τρόποι με τους οποίους θα μπορούν να αξιοποιηθούν οι δυνατότητες των ανέπαφων συναλλαγων :

1. Μεταφορά χρηματικών ποσών

Μία χρήσιμη εφαρμογή θα ήταν η ηλεκτρονική μεταφορά χρημάτων μέσω σύζευξης δύο συσκευών. Για παράδειγμα, τα χρήματα που δίνουν οι γονείς στα παδιά, θα μπορουσαν να έχουν ηλεκτρονική μορφη παρακάπτωντας τον παραδοσιακό τρόπο και μεταφέροντας συγκεκριμένο ποσό στη συσκευή τους. Με αυτόν τον τρόπο θα καλύπτονται οι καθημερινές τους ανάγκες.

2. Κλειδί χωρίς… κλειδί

Οποιαδήποτε συσκευή θα μπορεί να λειτουργήσει ως κλειδί για το αυτοκίνητό, το σπίτι ή το χρηματοκιβώτιό. Βέβαια, προκειμένου να μην υπάρξει κενό ασφαλείας σε περίπτωση κλοπής της συσκευής, το NFC θα ενεργοποιείται μόνο με

διαδικασία πιστοποίησης,η οποία θα ολοκληρώνεται με τον έλεγχο του προσώπου, ή της ίριδας του ματιού.

3. Κατάργηση της γραφειοκρατίας

Μία επιπλέον χρήση θα μπορούσε να είναι η αποθήκευση πληροφοριών όπως η ταυτότητά, η άδεια οδήγησης, ο αριθμός μητρώου της ασφαλιστικής, το διαβατήριό. Ο χρήστης θα έχει πλέον την δυνατότητα με την χρήση της συσκευής του, να ενημερώνει την εκάστοτε υπηρεσία που απαιτεί ορισμένες προσωπικές του πληροφορίες.

4. Κοινωνική δικτύωση

Το NFC θα μπορέσει να χρησιμοποιηθεί προκειμένου να βοηθήσει την αλληλεπίδραση στην κοινωνική ζωή. Για παράδειγμα, καταγράφοντας πληροφορίες σχετικές με τις υπηρεσίες ενός καταστήματος και την ποιότητα τους σε ένα NFC tag, θα μπορούν οι επόμενοι πελάτες να τις αξιοποιήσουν και να γνωρίζουν το τι πρέπει να προσέξουν. Μία επιχείρηση θα μπορεί να προσφέρει προνόμια σε όσους κάνουν check-in μέσω NFC.

5. Υγεία και Ασφάλιση

Στην ιατρική η χρήση του NFC μπορεί να βοηθήσει στην άμεση ενημέρωση του φακέλου των ασθενών. Διατηρώντας το ιατρικό ιστορικό στη συσκευή, θα είναι εύκολο ο γιατρός να έχει μία πλήρη εικόνα της κατάστασης των ασθενών και επιπλέον πληροφορίες που θα συλλέγονται, θα συγχρονίζονται με το κινητό τους. Παρομοίως, εισερχόμενος ένας ασθενής σε νοσοκομείο, θα μπορεί να κάνει check-in κατά την είσοδό του, και αυτομάτως το σύστημα θα ενημερώνεται για το ιατρικό ιστορικό του.

3.6 Χρήση τεχνολογίας NFC στο Android

[10]Υπάρχει μεγάλη ποικιλία από ετικέτες που μπορεί να διαβαστούν από τα έξυπνα κινητά. Το φάσμα κυμαίνεται από απλά αυτοκόλλητα και μπρελόκ σε πολύπλοκες κάρτες με ενσωματομένο κρυπτογραφικό υλικό. Οι ετικέτες (tags) διαφέρουν επίσης στην τεχνολογία των chip. Η πιό σημαντική είναι η *NDEF*, η οποία υποστηρίζεται από τις περισσότερες ετικέτες. Επιπλέον πρέπει να αναφέρθεί η Mifare, δεδομένου ότι είναι η πλέον χρησιμοποιούμενη τεχνολογία chip ανέπαφων συναλλαγών σε όλο τον κόσμο.

Για να αποκτήθεί πρόσβαση στο υλικό του NFC θα πρέπει να γίνει η κατάλληλη αναφορά στο αρχείο AndroidManifest.xml. Αν η εφαρμογή δεν λειτουργεί χωρίς την υποστήριξη NFC,μπορεί να καθορίστεί η κατάσταση της και ο τρόπος λειτουργίας της μέσα στο αρχείο της MainActivity. Σε περίπτωση που το NFC απαιτείται η εφαρμογή δεν μπορεί να εγκατασταθεί σε συσκευές χώρις την υποστήριξη της τεχνολογίας αυτής και το Google Play θα εμφανίζει την εφαρμογή μόνο σε NFC συμβεβλημένες συσκευές.

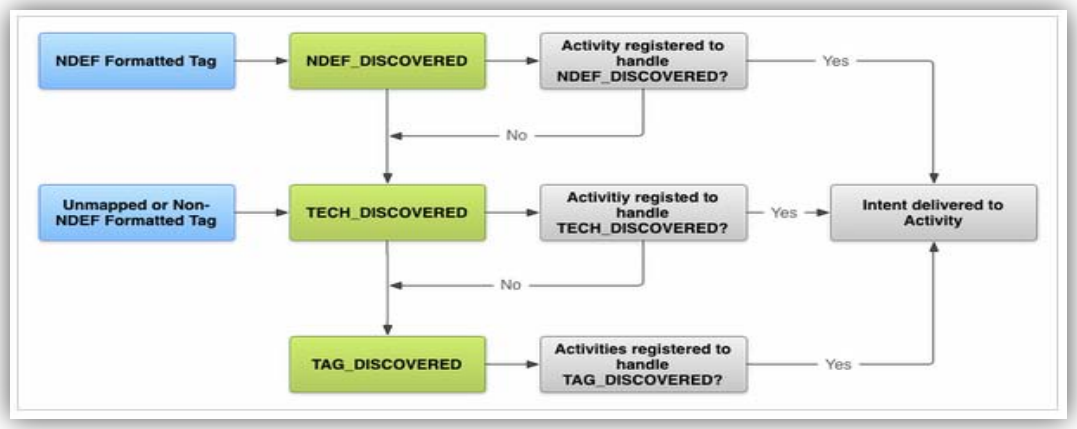
3.6.1 Διαμόρφωση φίλτρων και χειρισμός προτεραιοτήτων NFC

[10]Είναι απαραίτητη η εμφάνιση ειδοποίησης όταν η συσκευή έρχεται σε επαφή με μία ετικέτα. Το Android χρησιμοποιεί Intent συστήματος για να παραδώσει σις εφαρμογές τις πληροφορίες των ετικέτων. Αν υπάρχουν πολλές εφαρμογές που χειρίζονται NFC, εμφανίζεται ο επιλογέας δραστηριοτήτων στον χρήστη ώστε να αποφασίσει την επιθημητή εκτέλεση. Με τον ίδιο τρόπο αντιμετωπίζεται η ενεργοποίηση συσκεκριμένων διευθύνσεων στο Διαδίκτυο καθώς και η ανταλλαγή πληροφοριών.

Υπάρχουν 3 διαφορετικά είδη φίλτρων για τις ετικέτες(Η λίστα είναι ταξινομημένη με φθίνουσα σειρά προτεραιότητας) :

- ACTION_NDEF_DISCOVERED
- ACTION_TECH_DISCOVERED
- ACTION_TAG_DISCOVERED

Εάν το σύστημα ανιχνεύσει μια ετικέτα με υποστήριξη NDEF, δημιουργεί και ενεργοποιεί ένα νέο Intent. Η ετικέτα που υποστηρίζει NDEF σημαίνει πως έχει την μέγιστη προτεραίοτητα και μόλις ανιχθευθεί θα γίνει εκκίνηση της εφαρμογής στην οποία αντιστοχεί χωρίς ερώτηση του χρήστη. Ένα φίλτρο ACTION_TECH_DISCOVERED Intent είναι αυτό που δημιουργείται έαν καμία εφαρμογή δεν υποστηρίζει NDEF Intent ή η ίδια η ετικέτα δεν υποστηρίζει NDEF. Η προτεραιότητα του συγκεκριμένου φίλτρου μεταφράζεται ώς εμφάνιση στον χρήστη ενός παράθυρου διαλόγου με το οποίο του δίνεται η δυνατότητα να επιλέξει την εφαρμογή που θέλει να εκτελέσει.Εάν πάλι δεν βρεθεί καμία εφαρμογή για το Intent, τότε ενεργοποιείται ένα νέο Intent το ACTION_TAG_DISCOVERED. Το παρακάτω γράφημα δείχνει την διαδικασία:



Εικόνα 3.3 (πηγή : Reading NFC tags with Android [10])

Συνοψίζοντας, κάθε εφαρμογή θα πρέπει να φιλτράρει το Intent με την μεγαλύτερη προτεραιότητα.

Στην παρακάτω εικόνα φαίνεται φαίνεται η υλοποίηση του Tech Discovered Intent. Δημιουργείται ένας υποφάκελος xml στον φάκελο res. Σε αυτόν τον φάκελο δημιουργείται το αρχείο nfc_tech_filter.xml, στο όποιο προσδιορίζονται οι τεχνολογίες που προαναφέρθηκαν. Το επόμενο βήμα είναι η δημιουργία ενός IntentFilter στο αρχείο AndroidManifest.xml. Με αυτόν τον τρόπο θα γίνεται εκκίνηση της εφαρμογής μόλις προσεγγίζεται μια ετικέτα. Λεπτομερής αναφορά γίνεται στο 4ο Κεφάλαιο και παρουσιάζεται αναλυτικά η χρήση της τεχνολογίας αυτής στις εφαρμογές.[10]

```
01  <?xml version="1.0" encoding="utf-8"?>
02  <resources xmlns:xliff="urn:oasis:names:tc:xliff:document:1.2">
03      <tech-list>
04          <tech>android.nfc.tech.Ndef</tech>
05          <!-- class name -->
06      </tech-list>
07  </resources>
```

Εικόνα 3.4 (πηγή : Reading NFC tags with Android [10])

4ο Κεφάλαιο

Υλοποίηση εφαρμογής Uth Labs Equipment

Σε αυτό το κεφάλαιο παρουσιάζονται οι τεχνικές υλοποίησης της εφαρμογής. Συνοπτικά η κύρια λειτουργία της εφαρμογής είναι η εξής. Μέσω της συσκευής τηλεφώνου και έχοντας εντοπιστεί μία ετικέτα NFC εκτελείται αναζήτηση στη βάση δεδομένων.Υποστηρίζεται επίσης η λειτουργία προσθήκης εγγραφών στη βάση δεδομένων και η προβολή των περιεχομένων αυτής. Η προσθήκη και η προβολή απαιτούν την εισαγωγή ονόματος χρήστη και κωδικού. Επιλέγοντας ο χρήστης την προβολή των εγγραφών της βάσης, του δίνεται η δυνατότητα να προβάλει και όλα τα επιμέρους χαρακτηριστικά τους.

4.1 Εργαλεία λογισμικού και συσκευές

4.1.1 Περιβάλλον ανάπτυξης και προσομοίωσης

Για την ανάπτυξη της εφαρμογής χρησιμοποιήθηκε το Android Studio και ο Geny Motion Emulator ο οποίος είναι βασισμένος στο VirtualBox.

4.1.2 Συσκευές

Για την δοκιμή της εφαρμογής χρησιμοποιήθηκε κινητό Samsung Galaxy A3 με έκδοση 4.4. Για τον προσομοιωτή ένα LG Nexus 5 με έκδοση 4.4.2 . Η εφαρμογή είναι συμβατή με android 4.1.x (API 16) και πάνω.Χρησιμοποιήθηκαν ετικέτες Samsung NFC tags. Ο προγραμματισμός τους και η εγγραφή των αντίστοιχων id, υλοποιήθηκε μέσω της εφαρμογής TagWriter.

Εικόνα 4.1 (πηγή: www.samsung.com)

Εικόνα 4.2 (πηγή: TagWritter application)

4.1.3. Υλοποίηση της βάσης δεδομένων

Η υλοποίηση της βάσης δεδομένων έγινε με την χρήση του XAMPP. Είναι ένα πακέτο προγραμμάτων ελεύθερου λογισμικού, λογισμικού ανοιχτού κώδικα και ανεξάρτητου πλατφόρμας το οποίο παριέχει τον εξυπηρετητή σελίδων (http Apache), την βάση δεδομένων MySQL και ένα διερμηνέα για κώδικα γραμμένο σε γλώσσεσ προγραμματισμού PHP και Perl.

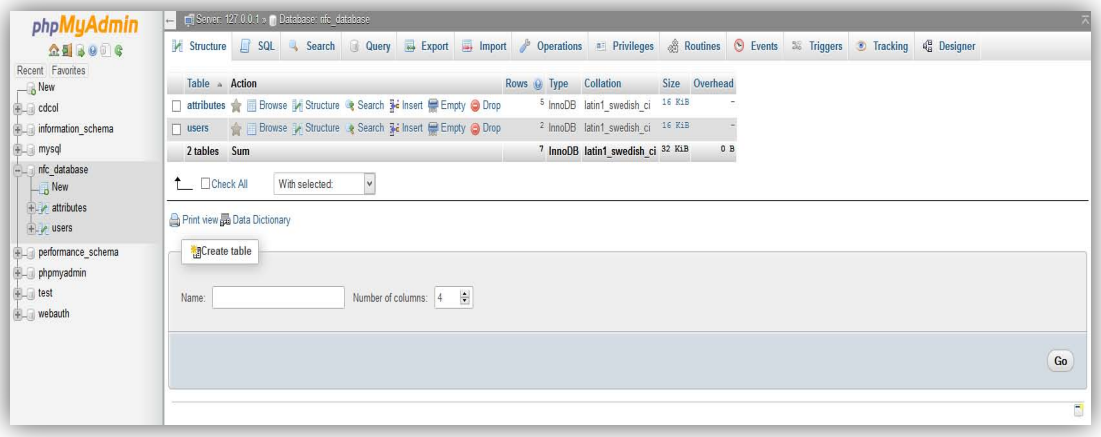
Δημιουργήθηκε τοπικά (localhost) μία βάση δεδομένων με όνομα nfc_database η οποία περιέχει δύο πίνακες. Τον πίνακα που τα πεδία του προσδιορίζουν τα χαρακτηριστικα κάθε καταχώρησης και τον πίνακα των χρηστών της βάσης. Στη βάση μπορούν να προστεθούν χρήστες και να προσδιοριστούν τα ανάλογα δικαιώματα τους. Επιπλέον υπάρχει και η δυνατότητα προσπέλασης της από οποιαδήποτε IP και όχι μόνο από τον localhost. Για προσπέλαση της βάσης από τον emulator η IP που χρησιμοποιείται είναι η ίδια με αυτή του virtualbox ενώ όταν γινέται προσπέλαση από φυσική συσκευή η IP είναι αυτή του προσωπικού υπολογιστή. Απαραίτητη προυπόθεση αποτελεί η σύνδεση και των δύο στο ίδιο δίκτυο.

Πεδία / Χαρακτηριστικά Εγγραφών:

(A/AId	ΤΥΠΟΣ	ΜΟΝΤΕΛΟ	ΚΑΤΑΣΚΕΥΑΣΤΗΣ	ΕΤΙΚΕΤΑ ΕΠΙΣΚΕΥΗΣ
ΤΙΜΗ	ΕΓΓΥΗΣΗ	ΤΙΜΟΛΟΓΙΟ	ΠΡΟΜΗΘΕΥΤΗΣ	ΗΜΕΡΟΜΗΝΙΑ ΑΓΟΡΑΣ
ΧΩΡΟΣ ΤΟΠΟΘΕΤΗΣΗΣ	ΧΡΗΣΤΗΣ			

Χρήστες:

Id (A/A)	ΟΝΟΜΑ ΧΡΗΣΤΗ	ΚΩΔΙΚΟΣ ΠΡΟΣΒΑΣΗΣ



Εικόνα 4.3 Σχηματικό της βάσης δεδομένων (πηγή: XAMMP, localhost/)

4.2 Κύρια δομικά στοιχεία

4.2.1 Η κλάση AsyncTask

Το λειτουργικό εφαρμόζει ένα μοντέλο ενιαίου νήματος (thread) και όταν εκτελείται μια εφαρμογή δημιουργείται και ένα αντίστοιχο νήμα. Η κλαση *AsyncTask* παρέχει την ένκολη διαχείριση νήματος του γραφικού περιβάλλοντος χρήστη. Αυτή η κλάση επιτρέπει να εκτελούνται διεργασίες στο παρασκήνιο και να προβάλουν τα αποτελεσματα τους στο γραφικο περιβάλλον χωρίς να χρειάζεται να διαχειριστούμε νήματα. Χρησιμοποιείται για χειρισμούς μικρής δίαρκειας (χρόνος που εκτελείται το νήμα). Περιλαμβάνει εκτέλεση μεθόδων σε τέσσερα στάδια:

I. *onPreExecute*: Η μέθοδος που εκτελείται πριν την μέθοδο *doInBackground*
II. *doInBackground:* Γίνεται χρήση της μεθόδου ώς βασικού βήματος εκτέλεσης του κώδικα καθώς χρησιμοποιείται για χρονοβόρους υπολογισμούς.
III. *onPostExecute*: Σε αυτή τη μέθοδο διοχετεύεται το αποτελεσμα μετά το πέρας της εκτέλεσης της doInBackground
IV. *onProgressUpdate*: Η μέθοδος χρησιμοποιείται για την ενημέρωση του νήματος που εκτελεί την διεπαφή του χρήστη, μέσω ενός παράθυρου διαλόγου που εμφανίζει την πρόοδο μιας διεργασίας.

4.2.2 Σύνδεση με την βάση δεδομένων

Για τη σύνδεση με τη βάση δεδομένων χρησιμοποιήθηκε το API Java DataBase Connectivity (JDBC) κάνοντας εισαγωγή (import) το αρχείο mysql-connector-java-5.1.20.jar . Χρησιμοποιώντας τον οδηγό (driver) του jdbc, τον σύνδεσμο (link) της βάσης, το όνομα και τον κωδικό πρόσβασης για τη βάση, γίνεται σύνδεση με τη βάση.

```
private final static String DRIVER="com.mysql.jdbc.Driver";
   private final static String CON_STRING="jdbc:mysql://192.168.2.14/nfc_database";
   private final static String USERNAME="test";
   private final static String PASSWORD="0000";
.
.
Class.forName(DRIVER);
connection = DriverManager.getConnection(CON_STRING, USERNAME, PASSWORD);
.
.
```

4.2.3 Επερωτήματα στη βάση δεδομένων

Για τα επερωτήματα χρησιμοποιείται το PreparedStatement το οποίο παρέχει ευκολία στην εισαγωγή παραμέτρων και την επαναχρησιμοποίηση των επερωτημάτων με νέες παραμέτρους. Επίσης παρέχει αύξηση της απόδοσης των

επερωτημάτων και διευκολύνει τη μαζική ενημέρωση χαρακτηριστικών στη βάση. Τέλος παρέχει προστασία από το SQL injection. Μετά από κάθε επερώτημα ή οποιδήποτε ενέργεια που εκτελείται στη βάση δεδομένων η σύνδεση κλείνει για λόγους ασφαλείας.

```
PreparedStatement ps = conn.prepareStatement("Select * from attributes where serialNum = ?");
```

4.2.4 XML manifest

Στο αρχείο manifest δηλώνονται οι άδειες (permissions) που απαιτεί η εφαρμογή για να λειτουργήσει και οι ιδιότητες της κάθε δραστηριότητας (activity). Κύριο χαρακτηριστικό της MainActivity είναι το intent-filter ώστε να ξεκινά η εφαρμογή όταν έρχεται σε επαφή με ένα nfc tag. Σε κάθε δραστηριότητα (activity) δηλώνεται ποια είναι η γονική της (activity), ετσι ώστε με την επιλογή στη μπάρα ενεργειών, να μπορεί ο χρήστης να επιστρέψει στην προηγουμενη οθόνη. Επίσης σε κάθε δραστηριότητα, με τη χρήση του **screenOrientation = “portrait”** εξαναγκάζεται η εφαρμογή να προβάλλεται μόνο σε κατακόρυφη προβολή ακόμα και όταν περιστρέφεται η συσκευή.

layouts

Για το γραφικό περιβάλλον χρησιμοποιήθηκαν απλά στοιχεία όπως TextViews όταν επρόκειτο για εμφάνιση πληροφοριών και EditTexts για εισαγωγή στοιχείων από το χρήστη. Δημιουργήθηκαν δύο ακόμα αρχεία διάταξης (layout):

1) Για την εμφάνιση του διαλόγου του login και
2) Για τη δομή των στοιχείων όταν εμφανίζεται όλη η βάση.

strings

Στο αρχείο strings.xml στο φάκελο values βρίσκονται δηλωμένα όλα τα κείμενα και οι λέξεις που χρησιμοποιούνται στο γραφικό περιβάλλον της εφαρμογής. Αυτό επιτρέπει την εύκολη αποσφαλμάτωση και επαναχρησιμοποίηση κομματιών κώδικα.

menu

Στο φάκελο menu βρίσκονται τα αρχεία menu_main, info_actions και add_actions στα οποία υπάρχουν οι ενέργειες που εμφανίζονται όταν χρησιμοποιείται το κουμπί του μενού. Ανατίθενται σε ενέργειες εικονίδια και είναι εφικτό να δηλωθούν ώστε να εμφανίζονται πάντα.

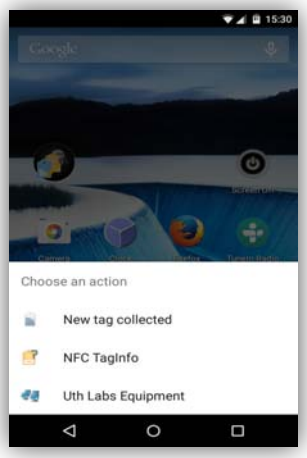
drawable

Οι εικόνες και τα εικονίδια που χρησιμοποιήθηκαν βρίσκονται στο φάκελο drawable. Στους υποφακέλους του υπάρχουν αρχεία που αναφέρονται σε διαφορετικές αναλύσεις.

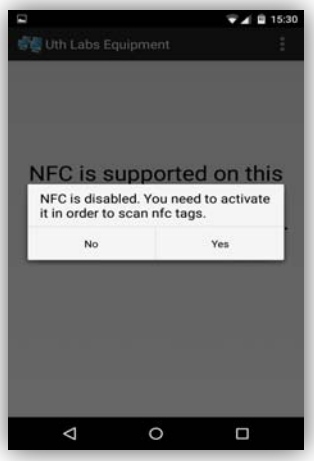
4.3 Περιγραφή επιμέρους λειτουργιών

4.3.1 Έλεγχος Προγράμματος

Η εκκίνηση της εφαρμογής είναι εφικτή με δύο τρόπους. Ο ένας είναι ο κλασικός τρόπος μέσω της λίστας εφαρμογών. Ο δεύτερος προσφέρει την δυνατότητα “επαφής” με κάποιο NFC tag δίνοντας την δυνατότητα εκκίνησης εφαρμογών που χρησιμοποιούν την συγκεκριμένη τεχνολογία. Με την έναρξη λειτουργίας της εφαρμογής καλείται η onCreate() της mainActivity.



Εικόνα 4.4(Στιγμιότυπο από την εφαρμογή)



Εικόνα 4.5 (Στιγμιότυπο από την εφαρμογή)

Το πρώτο στάδιο λειτουργίας περιλαμβάνει έλεγχο της συσκευής για την ύπαρξη NFC τεχνολογίας όπως επίσης και έλεγχο για το αν η σύνδεση δεδομένων στην συσκευή είναι ενεργοποιημένη. Σε περίπτωση μή υποστήριξης NFC εμφανίζεται σχετικό μήνυμα στην κυρίως οθόνη της εφαρμογής, ενώ αν είναι απενεργοποιημένα το NFC και η χρήση δεδομένων του διαδικτύου στη συσκευή, εμφανίζεται κατάλληλο μήνυμα ώστε ο χρήστης να τα ενεργοποιήσει. Χωρίς την ύπαρξη σύνδεσης στο διαδίκτυο δεν μπορεί να εκτελέστει κάποια από τις λειτουργίες της εφαρμογής. Η εμφάνιση του μηνύματος σε επίπεδο κώδικα, γίνεται με την κλήση της μεθόδου settingsCheck(String title,final int settigs). Τα ορίσματα της μεθόδου είναι αυτά που καθορίζουν τον τίτλο του διαλόγου που εμφανίζεται και την ενέργεια που θα εκτελεστεί σε τυχόν θετική επιλογή του χρήστη. Καίριας σημασίας είναι η χρήση της κλάσης AlertDialog για την δημιουργία του παραθύρου διαλόγου.

Το δεύτερο στάδιο λειτουργίας περιλαμβάνει την κλήση της μεθόδου handleIntent(), η οποία είναι υπεύθυνη για τον έλεγχο του φίλτρου που χρησιμοποείται και την ανάγνωση της ετικέτας (NFC tag). Το φίλτρο ορίζεται στο αρχείο manifestAndroid.xml στο πεδίο της mainActivity. Στην εφαρμογή χρησιμοποιείται το φίλτρο ACTION_TECH_DISCOVERED (βλ. 3ο Κεφάλαιο) όπως φαίνεται παρακάτω:

```
<intent-filter>
        <action android:name=”android.nfc.action.TECH_DISCOVERED” />
        </intent-filter>
<meta-data
        android:name=”android.nfc.action.TECH_DISCOVERED”
        android:resource=”@xml/nfc_tech_filter”
/>
```

Το τρίτο στάδιο περιλαμβάνει την εκτέλεση της ασύγχρονης διεργασίας (Async Task) η οποία είναι υπεύθυνη για την ανάγνωση του αλφαριθμητικού που είναι γραμμένο στην ετικέτα. Πραγματοποιείται έλεγχος αν η ετικέτα υποστηρίζει τον τύπο μηνύματος NDEF. Γίνεται κλήση της μεθόδου readText() η οποία είναι υπεύθυνη, για την σωστή ανάγνωση του αλφαριθμητικού, διαβάζει την κωδικοποίηση του μηνύματος, τον κωδικό της γλώσσας και επιστρέφει το αναγνωσμένο αλφαριθμητικό.

```
Private String readText(NdefRecord record) throws UnsupportedEncodingException {
        /* bit_7 defines encoding
        bit_6 reserved for future use, must be 0
        bit_5..0 length of IANA language code
        See NFC forum specification for “Text Record Type Definition” at 3.2.1
        http://www.nfc-forum.org/specs/
          */
        byte[] payload = record.getPayload();
        // Get the Text Encoding
        String textEncoding = ((payload[0] & 128) == 0) ? “UTF-8” : “UTF-16”;
                // Get the Language Code
                int languageCodeLength = payload[0] & 63;
                // Get the Text
                return new String(payload, languageCodeLength + 1, payload.length -
                languageCodeLength – 1, textEncoding);
    }
```

Στο επόμενο στάδιο χρησιμοποιείται το επιστρεφόμενο αλφαριθμητικό ως όρισμα για την εκτέλεση της ασύγχρονης αναζήτησης. Αρχικά εμφανίζεται ένα παράθυρο διαλόγου το οποίο εκτελείται μέχρι να ολοκληρωθεί η αναζήτηση. Παράλληλα δίνεται η δυνατότητα στο χρήστη ακύρωσης της αναζήτησης.

```
//showing the progress dialog window
@Override
protected void onPreExecute(){
    super.onPreExecute();
    progressDialog = new ProgressDialog(MainActivity.this);
    progressDialog.setTitle("Please wait");
    progressDialog.setMessage("Searching....");
    progressDialog.setCancelable(false);
    //set the dialog cancelable and closing on cancel button press
    progressDialog.setButton(DialogInterface.BUTTON_NEGATIVE, "Cancel",(OnClickListener) (dialog, which) -> {
        QueryTask.this.cancel(true);    //cancel the async task
        progressDialog.dismiss();       //close progress dialog
        Toast.makeText(getApplicationContext(),
                getResources().getString(R.string.canceled_operation), Toast.LENGTH_SHORT).show();
    });
    progressDialog.show();

}
```

Εικόνα 4.6 Κώδικας για τη εκτέλεση ασύγχρονης αναζήτησης (Αρχείο: MainActivity.java)

Στη συνέχεια γίνεται η σύνδεση με τη βάση δεδομένων.Δημιουργείται ένα αντικείμενο της κλάσης DBCon και καλείται η μέθοδος getCon ( ). Συμπληρώνεται το επερώτημα για αναζήτηση στη βάση, μέσω σειριακού αριθμού, και εκτελείται. Μετά το πέρας της αναζήτησης κλείνει το παράθυρο που παρουσιάζεται η πρόοδος της εκτέλεσης. Αν η αναζήτηση έχει επιστρέψει έγκυρο αποτέλεσμα τότε εμφανίζονται στο layout της εφαρμογής τα στοιχεία που υπάρχουν στη βάση. Αν η αναζήτηση δεν ήταν επιτυχής τότε όλα τα πεδία παρουσιάνται κενά και ένα μήνυμα ειδοποιεί το χρήστη για την αποτυχία. Με το τέλος της αναζήτησης τερματίζεται και η σύνδεση με την βάση για λόγους ασφάλειας.(βλ. αρχείο MainActivity.java)

Μία επιπλέον λειτουργία που ρυθμίζεται στο MainActivity.java είναι η μοναδικότητα της εκτέλεσης. Δηλαδή αν η εφαρμογή εκτελείται ήδη και γίνει ανάγνωση μιας ετικέτας, δεν επιχειρείται η επανέναρξη της εφαρμογής. Με αυτό τον τρόπο επιτυγχάνεται η συνεχής ανάγνωση ετικετών χωρίς να εμφανίζεται το παράθυρο «Επιλογή Εφαρμογής Εκτέλεσης». (Εικόνα 4.3)

```
/*parameter activity refers to the corresponding MainActivity requesting the foreground dispatch.
  and parameter adapter refers to the NfcAdapter used for the foreground dispatch.*/
public static void setupForegroundDispatch(final Activity activity, NfcAdapter adapter) {
    final Intent intent = new Intent(activity.getApplicationContext(), activity.getClass());
    intent.setFlags(Intent.FLAG_ACTIVITY_SINGLE_TOP);

    final PendingIntent pendingIntent = PendingIntent.getActivity(activity.getApplicationContext(), 0, intent, 0);

    IntentFilter[] filters = new IntentFilter[1];
    String[][] techList = new String[][]{};

    // Notice that this is the same filter as in our manifest.
    filters[0] = new IntentFilter();
    filters[0].addAction(NfcAdapter.ACTION_TECH_DISCOVERED);
    filters[0].addCategory(Intent.CATEGORY_DEFAULT);
    try {
        filters[0].addDataType(MIME_TEXT_PLAIN);
    } catch (IntentFilter.MalformedMimeTypeException e) {
        throw new RuntimeException("Check your mime type.");
    }

    adapter.enableForegroundDispatch(activity, pendingIntent, filters, techList);
}

/*parameter activity refers to he corresponding MainActivity requesting to stop the foreground dispatch.
  and parameter adapter refers to the NfcAdapter used for the foreground dispatch.*/
public static void stopForegroundDispatch(final Activity activity, NfcAdapter adapter) {
    adapter.disableForegroundDispatch(activity);
}
```

Εικόνα 4.7 (Αρχείο: MainActivity.java)

```
/* This method gets called, when a new Intent gets associated with the current activity instance.
  Instead of creating a new activity, onNewIntent will be called. For more information have a look
  at the documentation. In our case this method gets called, when the user attaches a Tag to the device.*/
@Override
protected void onNewIntent(Intent intent) {

    handleIntent(intent);
}
```

Εικόνα 4.8 (Αρχείο: MainActivity.java)

Από την αρχική οθόνη της εφαρμογής ο χρήστης μπορεί να μεταβεί σε λειτουργίες όπως η προσθήκη στοιχείου, η προβολή της βάσης δεδομένων, η προβολή πληροφοριών της εφαρμογής και η προβολή βοήθειας. Για την εμφάνιση του μενού πραγματοποιείται κλήση της onCreateOptionsMenu ( ). Όταν ο χρήστης

κάνει μια επιλογή τότε καλείται η onOptionsItemSelected ( ) με την λειτουργία της οποίας υλοποιούνται οι απαραίτητες ενέργειες για την εκτέλεση της επιλογής χρήστη.

- Επιλέγοντας About εμφανίζεται ένα παράθυρο διαλόγου όπου αναφέρονται πληροφορίες της εφαρμογής.
- Επιλέγοντας Help εμφανίζονται πληροφορίες για τον τρόπο χρήσης της εφαρμογής.
- Επιλέγοντας την προσθήκη στοιχείο, ζητείται από το χρήστη να εισάγει όνομα και κωδικό (εάν δεν τα έχει εισάγει προηγουμένως). Εάν τα διαπιστευτήρια είναι σωστά τότε εμφανίζεται το παράθυρο με όλα τα πεδία που πρέπει να συμπληρώσει ο χρήστης. Για την εμφάνιση του παραθύρου καλείται η μέθοδος showLogin. Ο χρήστης επιλέγοντας να συνδεθεί, εκτελείται η ασυγχρονη διεργασία για την αναζήτηση στον πίνακα χρηστών της βάσης δεδομένων. Εάν υπάρχει ο χρήστης και γίνει ταυτοποίηση του, καλείται η addItem δημιουργώντας ένα καινούργιο Intent το οποίο θα περιέχει και το όνομα του χρήστη που συνδέθηκε.

```
/*method for preparing the intent for Adding an item and adding to the extras the added_by parameter*/
private void addItem() {
    Intent i = new Intent(this, AddItem.class);
    i.putExtra("added_by", added_by);
    startActivityForResult(i, ACTIVITY_CREATE);
}
```

Εικόνα 4.9 (Αρχείο: MainActivity.java)

- Επιλέγοντας την προβολή της βάσης δεδομένων εμφανίζεται η λίστα με όλες τις καταχωρήσεις. Για κάθε αντικείμενο εμφανίζεται το όνομα και ο σειριακός αριθμός. Αν δεν έχει υπάρξει σύνδεση του χρήστη εμφανίζεται το προηγούμενο παράθυρο διαλόγου όπως αναλύθηκε παραπάνω. Γίνεται κλήση της μεθόδου showdb δημιουργώντας νέο intent.

```
/*method for preparing the intent for showing the database*/
private void showdb(){
    Intent i = new Intent(this, ShowDB.class);
    startActivityForResult(i, ACTIVITY_CREATE);
}
```

(Π,

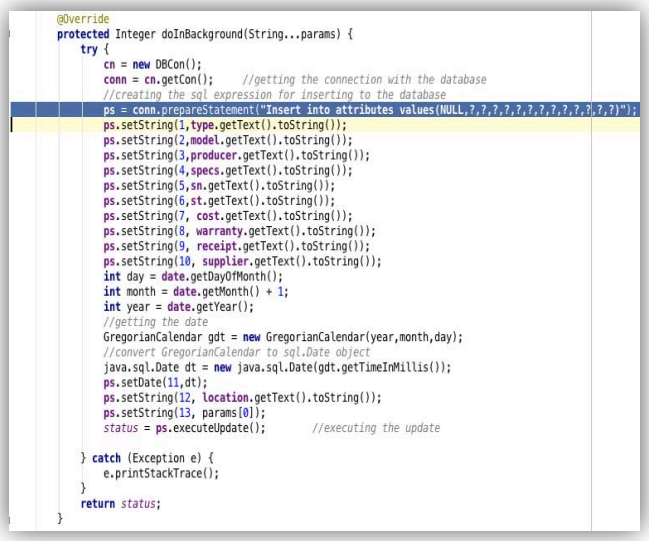
Εικόνα 4.10 (Αρχειο: MainActivity.java)

4.3.2 Προσθήκη νέων εγγραφών

Σε ξεχωριστό αρχείο υλοποιείται ο κώδικας που χρησιμοποιείται απο την εφαρμογή για προσθήκη νέας καταχώρησης στη βάση δεδομένων. Αρχικά καλείται η onCreate με την οποία γίνεται αρχικοποίηση των στοιχείων της διεπαφής του χρήστη. Επίσης γίνεται έλεγχος για τα επιπλέον στοιχεία του Intent. Αν δεν είναι μηδενικά (null) λαμβάνεται το όνομα του χρήστη που έχει συνδεθεί.

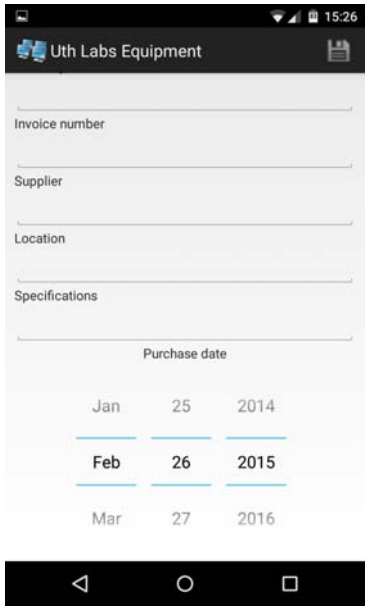
Έχοντας συμπληρώσει τα στοιχεία ο χρήστης και δίνοντας την εντολή καταχώρησης τους, εκτελείται η μέθοδος save η οποία ελέγχει την εγκυρότητα των πεδίων.Στην συνέχεια, εάν όλα είναι σωστά εκτελείται ασύγχρονη διεργασία που υλοποιεί την σύνδεση με τη βάση δεδομένων και την καταχώρηση της νέας

εγγραφής. Χρησιμοποιείται η μέθοδος doInBackground με την οποία γίνεται η προετοιμασία των πεδίων και εκτελείται η αποθήκευση των δεδομένων.



```
@Override
protected Integer doInBackground(String...params) {
    try {
        cn = new DBCon();
        conn = cn.getCon();     //getting the connection with the database
        //creating the sql expression for inserting to the database
        ps = conn.prepareStatement("Insert into attributes values(NULL,?,?,?,?,?,?,?,?,?,?,?,?)");
        ps.setString(1,type.getText().toString());
        ps.setString(2,model.getText().toString());
        ps.setString(3,producer.getText().toString());
        ps.setString(4,specs.getText().toString());
        ps.setString(5,sn.getText().toString());
        ps.setString(6,st.getText().toString());
        ps.setString(7, cost.getText().toString());
        ps.setString(8, warranty.getText().toString());
        ps.setString(9, receipt.getText().toString());
        ps.setString(10, supplier.getText().toString());
        int day = date.getDayOfMonth();
        int month = date.getMonth() + 1;
        int year = date.getYear();
        //getting the date
        GregorianCalendar gdt = new GregorianCalendar(year,month,day);
        //convert GregorianCalendar to sql.Date object
        java.sql.Date dt = new java.sql.Date(gdt.getTimeInMillis());
        ps.setDate(11,dt);
        ps.setString(12, location.getText().toString());
        ps.setString(13, params[0]);
        status = ps.executeUpdate();        //executing the update

    } catch (Exception e) {
        e.printStackTrace();
    }
    return status;
}
```



Εικόνες 4.10,4.11 Κώδικας που υλοποιεί το συγκεκριμένο στιγμιότυπο της εφαρμογής (Παράρτημα, AddItem.java)

4.3.3 Εμφάνιση λίστας καταχωρήσεων

Μια από τις δυνατότητες που έχει ο χρήστης, είναι η επιλογή εμφάνισης όλων των καταχωρήσεων της βάσης δεδομένων μέσω του μενού της εφαρμογής. Καλείται η onCreate για αρχικοποίηση της διεπαφής και εκτελείται η ασύγχρονη διεργασία για εμφάνιση της βάσης. Για την εμφάνιση των καταχωρήσεων ως λίστα χρησιμοποιείται ένας adapter της κλάσης SimpleAdapter. Βασικά ορίσματα για τον constructor της SimpleAdapter αποτελούν δύο πίνακες. Ο ένας περιέχει δύο αλφαριθμητικά (item , serialn) και ο άλλος περιέχει τα ids των στοιχείων του αρχείου list_item.xml . Αυτοί οι πίνακες χρησιμεύουν στο να αντιστοιχηθούν τα σωστά πεδία στην διεπαφή.

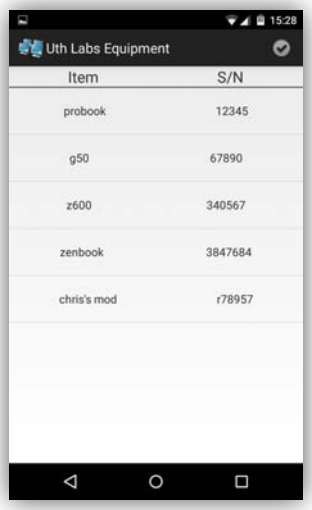
```
//creating an adapter for the items
SimpleAdapter adapter = new SimpleAdapter(ShowDB.this, list_items, R.layout.list_item, from, to);
setListAdapter(adapter);        //showing the list
```

Εικόνα 4.12 (Παράρτημα, ShowDB.java)

Επίσης η δραστηριότητα (Activity) αυτή υποστηρίζει την επιλογή ενός στοιχείου, από την προβληθείσα λίστα, και την εμφάνιση των χαρακτηριστικών του. Η υλοποίηση επιτυγχάνεται ώς εξής : Με τον σειριακό αριθμό του στοιχείου που επιλέχθηκε δημιουργείται ένα Intent με αυτό το χαρακτηριστικό.

```
/*method for getting the item selected*/
@Override
protected void onListItemClick(ListView l, View v, int pos, long id) {
    Intent i = new Intent(this, ShowItem.class);
    item = list_items.get(pos);
    i.putExtra("sn", item.get("serialn"));
    startActivityForResult(i, ACTIVITY_CREATE);
}
```

Κώδικας που υλοποιεί το παρακάτω στιγμιότυπο της εφαρμογής



Εικόνες 4.13 , 4.14 (Παράρτημα, ShowDB.java)

4.3.4 Εμφάνιση χαρακτηριστικών

Για την εμφάνιση των χαρακτηριστικών μιας καταχώρησης, αρχικά καλείται η onCreate στην οποία γίνεται αρχικοποίηση των στοιχείων της διεπαφής του χρήστη. Επίσης γίνεται έλεγχος για τα επιπλέον στοιχεία του Intent. Αν δεν είναι μηδενικά (null) λαμβάνεται το όνομα του χρήστη που έχει συνδεθεί. Τέλος, εκτελείται η ασύγχρονη διεργασία που είναι υπεύθυνη για την εμφάνιση των πληροφοριών της καταχώρησης.

Μέσω της μεθόδου doInBackGround γίνεται σύνδεση με την βάση και εκτελείται το επερώτημα. Παρακάτω εμφανίζεται το τμήμα κώδικο που την υλοποιεί.

```
@Override
protected Integer doInBackground(String...params) {

    try {
        cn = new DBCon();
        conn = cn.getCon();      //getting the connection
        ps = conn.prepareStatement("Select * from attributes where serialNum = ?");
        ps.setString(1, params[0]);
        rs = ps.executeQuery();      //searching for a specific item in the database


    } catch (Exception e) {
        e.printStackTrace();
    }
    return 0;
}
```

Εικόνα 4.15 (Παράρτημα, ShowItem.java)

4.4 Μελλοντικές χρήσεις της εφαρμογής

Η συγκεκριμένη εφαρμογή δύναται να χρησιμοποιηθεί για την καταγραφη του ηλεκτρονικού εξοπλισμού σε οποιοδήποτε οργανισμό, υπηρεσία, εταιρεία ή πανεπιστημιακό ίδρυμα. Προσφέρει εύκολη οργάνωση και καταγραφή καθώς και επεξεργασία των καταχωρήσεων. Συγκεκρίμενα, για την χρήση σε κάποιο Πανεπιστημιακό τμήμα, ο server μπορεί να φιλοξενεί την βάση δεδομένων. Οι χρήστες δικτύου του τμήματος έχουν πρόσβαση σε αυτήν, χρησιμοποιώντας ονόματα χρηστών και κωδικούς. Ορίζοντας στην εφαρμογή την IP του server και χρησιμοποιώντας προκαθορισμένα αναγνωριστικά, θα μπορούν να εκτελεστούν όλες οι λειτουργίες της εφαρμογής. Για χρήση χωρίς σύνδεση, η εφαρμογή εμφανίζει μόνο τα χαρακτηριστικά που προκύπτουν από την ανάγνωση των NFC ετικετών.

ΠΑΡΑΡΤΗΜΑ:

AndroidManifest.xml

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

   <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="android.permission.READ_EXTERNAL_STORAGE"/>
   <uses-permission android:name="android.permission.NFC" />

   <uses-feature
      android:name="android.hardware.nfc"
      android:required="true" />

   <application
      android:allowBackup="true"
      android:icon="@drawable/ic_launcher"
      android:label="@string/app_name"
      android:theme="@style/AppTheme" >
      <activity
         android:name=".MainActivity"
         android:label="@string/app_name"
         android:screenOrientation="portrait"
         android:launchMode = "singleInstance">
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
         <!--<intent-filter>
            <action android:name="android.nfc.action.NDEF_DISCOVERED" />

            <category android:name="android.intent.category.DEFAULT" />

            <data android:mimeType="text/plain" />
         </intent-filter>
         -->
         <intent-filter>
            <action android:name="android.nfc.action.TECH_DISCOVERED" />
         </intent-filter>

         <meta-data
            android:name="android.nfc.action.TECH_DISCOVERED"
            android:resource="@xml/nfc_tech_filter" />

      </activity>
      <activity android:name=".AddItem"
         android:windowSoftInputMode="adjustResize"
         android:screenOrientation="portrait">
         <meta-data
            android:name="android.support.PARENT_ACTIVITY"
            android:value="MainActivity" />
      </activity>
      <activity android:name=".ShowDB"
         android:label="@string/app_name"
         android:screenOrientation="portrait">
         <meta-data
            android:name="android.support.PARENT_ACTIVITY"
            android:value="MainActivity"/>
      </activity>
      <activity android:name=".ShowItem"
         android:label="@string/app_name"
         android:screenOrientation="portrait">
         <meta-data
            android:name="android.support.PARENT_ACTIVITY"
            android:value="ShowDB"/>
      </activity>
      <activity
         android:name=".About"
         android:label="@string/app_name"
```

```
            android:theme="@android:style/Theme.Holo.Light.Dialog"
            android:screenOrientation="portrait">
            <intent-filter>
                <action android:name="com.gmarelas.uthlabsequipment.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.Holo.Light.Dialog"
            android:screenOrientation="portrait">
            <intent-filter>
                <action android:name="com.gmarelas.uthlabsequipment.HELP" />

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

</manifest>
```

DBCon.java

```
package com.gmarelas.uthlabsequipment;

import java.sql.*;
import java.util.logging.Level;
import java.util.logging.Logger;

public class DBCon {

    private final static String DRIVER="com.mysql.jdbc.Driver";
    private final static String CON_STRING="jdbc:mysql://192.168.1.251/nfc_database";       //link to the database
    private final static String USERNAME="marelas";
    private final static String PASSWORD="marelas";

    private Connection connection;

    private void connect(){
        try {
            Class.forName(DRIVER);      //initializes the class and gets registered in the jdbc driver manager
            connection = DriverManager.getConnection(CON_STRING, USERNAME, PASSWORD);
        }
        catch(SQLException e){      //catching exceptions when trying to connect
        System.out.println("Exception in DBCon");
        }catch(ClassNotFoundException ex){
            Logger.getLogger(DBCon.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
    /*public getter method for getting the connection*/
    public Connection getCon(){
        connect();
        return connection;
    }
    /*public method for closing the connection*/
    public void closeCon(Connection conn){
        try {
            conn.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

}
```

AddItem.java

```
package com.gmarelas.uthlabsequipment;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.DialogInterface;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.Toast;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import java.util.GregorianCalendar;

public class AddItem extends Activity{

    private static final int LENGTH1 = 35;      //length of some attributes in the database
    private static final int LENGTH2 = 15;      //length of some attributes in the database
    private static final int LENGTH3 = 200;     //length of some attributes in the database
    private EditText type, model, sn, st, supplier, specs, producer, cost, warranty, receipt, location;
    private DatePicker date;
    private ProgressDialog progressDialog;
    private String added_by;                    //helper variable to store the user who added an item to the database
    static int status = 0;

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

        progressDialog = new ProgressDialog(AddItem.this);

        type = (EditText) findViewById(R.id.type);
        model  = (EditText) findViewById(R.id.model);
        sn  = (EditText) findViewById(R.id.sn);
        st  = (EditText) findViewById(R.id.service);
        supplier  = (EditText) findViewById(R.id.supplier);
        specs  = (EditText) findViewById(R.id.specs);
        producer  = (EditText) findViewById(R.id.producer);
        cost  = (EditText) findViewById(R.id.cost);
        warranty  = (EditText) findViewById(R.id.warranty);
        receipt  = (EditText) findViewById(R.id.receipt);
        location  = (EditText) findViewById(R.id.location);
        date = (DatePicker) findViewById(R.id.date_widget);

        Bundle extras = getIntent().getExtras();
        if(extras!=null){       //getting the extras
            added_by = extras.getString("added_by");
        }

    }

    /*inflate the options menu*/
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.add_actions, menu);
        return super.onCreateOptionsMenu(menu);
    }

    /*actions triggered on option selection*/
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.action_save:
                if(!save()){
                    return false;
                }
```

```
            setResult(RESULT_OK);
            finish();
            return true;
        }
        return super.onOptionsItemSelected(item);
    }

    /*Asynchronous task for saving item to the database*/
    private class SaveTask extends AsyncTask<String,Void,Integer>{

        private Connection conn = null;
        private PreparedStatement ps = null;
        private DBCon cn;

        //showing the progress dialog
        @Override
        protected void onPreExecute(){
            super.onPreExecute();
            progressDialog = new ProgressDialog(AddItem.this);
            progressDialog.setTitle(getResources().getString(R.string.wait));
            progressDialog.setMessage(getResources().getString(R.string.saving));
            progressDialog.setCancelable(false);
            //set the dialog cancelable and closing on cancel button press
            progressDialog.setButton(DialogInterface.BUTTON_NEGATIVE, getResources().getString(R.string.cancel),new
DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    SaveTask.this.cancel(true);    //cancel the async task
                    progressDialog.dismiss();       //close progress dialog
                    Toast.makeText(getApplicationContext(),
                            getResources().getString(R.string.canceled_operation), Toast.LENGTH_SHORT).show();
                }
            });
            progressDialog.show();

        }

        @Override
        protected Integer doInBackground(String...params) {
            try {
                cn = new DBCon();
                conn = cn.getCon();     //getting the connection with the database
                //creating the sql expression for inserting to the database
                ps = conn.prepareStatement("Insert into attributes values(NULL,?,?,?,?,?,?,?,?,?,?,?,?,?)");
                ps.setString(1,type.getText().toString());
                ps.setString(2,model.getText().toString());
                ps.setString(3,producer.getText().toString());
                ps.setString(4,specs.getText().toString());
                ps.setString(5,sn.getText().toString());
                ps.setString(6,st.getText().toString());
                ps.setString(7, cost.getText().toString());
                ps.setString(8, warranty.getText().toString());
                ps.setString(9, receipt.getText().toString());
                ps.setString(10, supplier.getText().toString());
                int day = date.getDayOfMonth();
                int month = date.getMonth() + 1;
                int year = date.getYear();
                //getting the date
                GregorianCalendar gdt = new GregorianCalendar(year,month,day);
                //convert GregorianCalendar to sql.Date object
                java.sql.Date dt = new java.sql.Date(gdt.getTimeInMillis());
                ps.setDate(11,dt);
                ps.setString(12, location.getText().toString());
                ps.setString(13, params[0]);
                status = ps.executeUpdate();        //executing the update

            } catch (Exception e) {
                e.printStackTrace();
            }
            return status;
        }

        protected void onPostExecute(Integer result){
            progressDialog.dismiss();       //closing the progress dialog window
            try {
```

```
            ps.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        cn.closeCon(conn);      //closing the connection
    }

}

/*method for checking validity of fields and saving the item*/
private boolean save(){

    //helper variables for checking validity of fields
    boolean type_check, model_check, prod_check, supp_check, loc_check, specs_check, sn_check, st_check, cost_check,
warr_check, rec_check;

    type_check = model_check = prod_check = supp_check = loc_check = specs_check = sn_check = st_check = cost_check =
warr_check = rec_check = true;

    if(type.getText().toString().length()>LENGTH1 || type.getText().toString().equals("")){
        type_check = false;
        type.setError(getResources().getString(R.string.data_length) + " " + LENGTH1 + " " +
getResources().getString(R.string.chars));
    }
    if(model.getText().toString().length()>LENGTH1 || model.getText().toString().equals("")){
        model_check = false;
        model.setError(getResources().getString(R.string.data_length) + " " + LENGTH1 + " " +
getResources().getString(R.string.chars));
    }
    if(producer.getText().toString().length()>LENGTH1 || producer.getText().toString().equals("")){
        prod_check = false;
        producer.setError(getResources().getString(R.string.data_length) + " " + LENGTH1 + " " +
getResources().getString(R.string.chars));
    }
    if(supplier.getText().toString().length()>LENGTH1 || supplier.getText().toString().equals("")){
        supp_check = false;
        supplier.setError(getResources().getString(R.string.data_length) + " " + LENGTH1 + " " +
getResources().getString(R.string.chars));
    }
    if(location.getText().toString().length()>LENGTH1 || location.getText().toString().equals("")){
        loc_check = false;
        location.setError(getResources().getString(R.string.data_length) + " " + LENGTH1 + " " +
getResources().getString(R.string.chars));
    }
    if(specs.getText().toString().length()>LENGTH3 || specs.getText().toString().equals("")){
        specs_check = false;
        specs.setError(getResources().getString(R.string.data_length) + " " + LENGTH3 + " " +
getResources().getString(R.string.chars));
    }
    if(sn.getText().toString().length()>LENGTH1 || sn.getText().toString().equals("")){
        sn_check = false;
        sn.setError(getResources().getString(R.string.data_length) + " " + LENGTH1 + " " +
getResources().getString(R.string.chars));
    }
    if(st.getText().toString().length()>LENGTH1 || st.getText().toString().equals("")){
        st_check = false;
        st.setError(getResources().getString(R.string.data_length) + " " + LENGTH1 + " " +
getResources().getString(R.string.chars));
    }
    if(cost.getText().toString().length()>LENGTH2 || cost.getText().toString().equals("")){
        cost_check = false;
        cost.setError(getResources().getString(R.string.data_length) + " " + LENGTH2 + " " +
getResources().getString(R.string.chars));
    }
    if(warranty.getText().toString().length()>LENGTH2 || warranty.getText().toString().equals("")){
        warr_check = false;
        warranty.setError(getResources().getString(R.string.data_length) + " " + LENGTH2 + " " +
getResources().getString(R.string.chars));
    }
    if(receipt.getText().toString().length()>LENGTH2 || receipt.getText().toString().equals("")){
        rec_check = false;
        receipt.setError(getResources().getString(R.string.data_length) + " " + LENGTH2 + " " +
getResources().getString(R.string.chars));
    }
```

```
        //if all fields contain valid values then save the item
        if(type_check && model_check && prod_check && supp_check && loc_check && specs_check && sn_check &&
st_check && cost_check && warr_check && rec_check) {
            new SaveTask().execute(added_by);
        }
        else{
            Toast.makeText(getApplicationContext(),
                    getResources().getString(R.string.wrong_values), Toast.LENGTH_SHORT).show();
            return false;
        }
        return true;
    }

    /*overriding onStop and closing progress dialog window to avoid window leaks*/
    @Override
    protected void onStop() {

        progressDialog.dismiss();

        super.onStop();
    }

}
```

ShowDB.java

```
package com.gmarelas.uthlabsequipment;

import android.app.ListActivity;
import android.app.ProgressDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.ListView;
import android.widget.SimpleAdapter;
import android.widget.Toast;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;

/*class for showing the whole database. Only the model and the serial number are shown.*/
public class ShowDB extends ListActivity {

    public static final int ACTIVITY_CREATE = 1;
    private List<HashMap<String,String>> list_items;
    private HashMap<String, String> item;
    private ProgressDialog progressDialog;

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

        progressDialog = new ProgressDialog(ShowDB.this);

        registerForContextMenu(getListView());
        new ShowTask().execute();
    }


    private class ShowTask extends AsyncTask<Void,Void,Integer> {
```

```
    private PreparedStatement ps = null;
    private Connection conn = null;
    private ResultSet rs = null;
    private DBCon cn;
    private String[] from = new String[] { "item", "serialn" };     //setting the id keys
    private int[] to = new int[] { R.id.item, R.id.serial_num };    //setting the destination in the list_item

    //showing the progress dialog window
    @Override
    protected void onPreExecute(){
       super.onPreExecute();
       progressDialog = new ProgressDialog(ShowDB.this);
       progressDialog.setTitle(getResources().getString(R.string.wait));
       progressDialog.setMessage(getResources().getString(R.string.fetch));
       progressDialog.setCancelable(false);
       //set the dialog cancelable and closing on cancel button press
       progressDialog.setButton(DialogInterface.BUTTON_NEGATIVE, getResources().getString(R.string.cancel),new
DialogInterface.OnClickListener() {
          @Override
          public void onClick(DialogInterface dialog, int which) {
             ShowTask.this.cancel(true);    //cancel the async task
             progressDialog.dismiss();      //close progress dialog
             Toast.makeText(getApplicationContext(),
                   getResources().getString(R.string.canceled_operation), Toast.LENGTH_SHORT).show();
          }
       });
       progressDialog.show();

    }

    @Override
    protected Integer doInBackground(Void...params) {
       try {
          list_items = new ArrayList<>();
          cn = new DBCon();
          conn = cn.getCon();     //getting the connection
          //statement for selecting all entries in the database
          ps = conn.prepareStatement("Select * from attributes");
          rs = ps.executeQuery();     //executing the query
          while(rs.next()){       //while there is an item in the database
             item = new HashMap<>();     //create new HashMap for every item
             item.put("item", rs.getString(3));      //insert model paired with key item
             item.put("serialn", rs.getString(6));   //insert serial number paired with key serialn
             list_items.add(item);       //adding each item to the list
          }
       } catch (Exception e) {
          e.printStackTrace();
       }
       return 0;
    }

    protected void onPostExecute(Integer result){
       progressDialog.dismiss();       //closing the progress dialog window
       //creating an adapter for the items
       SimpleAdapter adapter = new SimpleAdapter(ShowDB.this, list_items, R.layout.list_item, from, to);
       setListAdapter(adapter);        //showing the list
       try {
          //close everything
          ps.close();
          rs.close();
          cn.closeCon(conn);
       } catch (SQLException e) {
          e.printStackTrace();
       }

    }

 }

 /*method for returning the item selected*/
 @Override
 public boolean onContextItemSelected(MenuItem item) {

    return super.onContextItemSelected(item);
 }
```

```
    /*method for getting the item selected*/
    @Override
    protected void onListItemClick(ListView l, View v, int pos, long id) {
        Intent i = new Intent(this, ShowItem.class);
        item = list_items.get(pos);
        i.putExtra("sn", item.get("serialn"));
        startActivityForResult(i, ACTIVITY_CREATE);
    }

    /*inflate the settings menu*/
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.info_actions, menu);
        return super.onCreateOptionsMenu(menu);
    }

    /*triggered actions for option selected*/
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.action_done:
                setResult(RESULT_OK);
                finish();
                return true;
        }
        return super.onOptionsItemSelected(item);
    }

    /*overriding onStop and closing progress dialog window to avoid window leaks*/
    @Override
    protected void onStop() {

        progressDialog.dismiss();

        super.onStop();
    }

}
```

Αναφορές – Βιβλιογραφία

[1] Η ιστορία του Android http://www.allaboutandroid.gr/?p=6362

[2] Android version history (http://en.wikipedia.org/wiki/Android_version_history)

[3] Java tutorial.Simply Easy Lerarnig by tutorialspoint.com (pdf)

[4] Android, at a glance (http://www.cubrid.org/blog/dev-platform/android-at-a-glance/)

[5] Structure of an Android App (http://sofia.cs.vt.edu/sofia-2114/book/chapter2.html)

[6] Model-View- Controller (http://en.wikipedia.org/wiki/Model-view-controller)

[7] Android Activity Lifecycle (http://www.javatpoint.com/android-life-cycle-of-activity)

[8] Managing the Activity Lifecycle (http://developer.android.com/training/basics/activity-lifecycle/index.html)

[9] Public class Activity (http://developer.android.com/reference/android/app/Activity.html)

[10]Reading NFC Tags with Android (http://code.tutsplus.com/tutorials/reading-nfc-tags-with-android--mobile-17278)

[12] Τι είναι το NFC (http://tech.in.gr/short-news/?aid=1231116834)

[13] Async Task (http://developer.android.com/reference/android/os/AsyncTask.html)

[14] Angroid (http://www.angroid.gr/featured/)

[15] Simple Android NFC program (http://stackoverflow.com/questions/11743222/simple-android-nfc-program)

[16] NFC Programming on Android (http://www.developer.com/ws/android/nfc-programming-in-android.html)

[17] NFC Workshop Series: How To Write Content To An NFC Tag (http://tapintonfc.blogspot.gr/2012/07/the-above-footage-from-our-nfc-workshop.html)

[18] The new boston Android Tutorials (https://www.youtube.com/playlist?list=PL023BC9408BAFEC0C)