ΤΕΙ ΚΕΝΤΡΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ
ΣΕΡΡΕΣ

Τμήμα Μηχανικών Πληροφορικής ΤΕ

2014

ΣΥΣΤΗΜΑ ΑΝΑΓΝΩΡΙΣΗΣ QR CODES ΓΙΑ ΔΙΑΣΥΝΔΕΣΗ ΜΕ ΔΙΑΔΙΚΤΥΑΚΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ/ΠΥΛΕΣ ΜΕΣΩ SQL ΚΛΗΣΕΩΝ ΣΕ ΚΙΝΗΤΑ ΤΗΛΕΦΩ-ΝΑ ΜΕ ΛΕΙΤΟΥΡΓΙΚΟ ΣΥΣΤΗΜΑ ANDROID

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΒΑΣΙΛΕΙΟΣ ΤΣΟΡΜΠΑΤΖΟΥΔΗΣ

ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ
ΚΕΝΤΡΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΤΕ

ΣΥΣΤΗΜΑ ΑΝΑΓΝΩΡΙΣΗΣ QR CODES ΓΙΑ ΔΙΑΣΥΝΔΕΣΗ ΜΕ ΔΙΑΔΙΚΤΥΑΚΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ/ΠΥΛΕΣ ΜΕΣΩ SQL ΚΛΗΣΕΩΝ ΣΕ ΚΙΝΗΤΑ ΤΗΛΕΦΩ-ΝΑ ΜΕ ΛΕΙΤΟΥΡΓΙΚΟ ΣΥΣΤΗΜΑ ANDROID

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Βασίλειος Τσορμπατζούδης, Α.Μ. 1326

ΣΕΡΡΕΣ, 2014

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

ΣΥΣΤΗΜΑ ΑΝΑΓΝΩΡΙΣΗΣ QR CODES ΓΙΑ ΔΙΑΣΥΝΔΕΣΗ ΜΕ ΔΙΑΔΙΚΤΥΑΚΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ/ΠΥΛΕΣ ΜΕΣΩ SQL ΚΛΗΣΕΩΝ ΣΕ ΚΙΝΗΤΑ ΤΗΛΕΦΩΝΑ ΜΕ ΛΕΙΤΟΥΡΓΙΚΟ ΣΥΣΤΗΜΑ ANDROID

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Τσορμπατζούδης Βασίλειος, Α.Μ. 1326

Επιβλέπων: Δρ. Γεώργιος Παυλίδης
Επιστημονικός Συνεργάτης ΤΕΙ Κεντρικής Μακεδονίας

ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ
ΚΕΝΤΡΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΤΕ

Αφιερώσεις

Θέλω να ευχαριστήσω πρώτον από όλους τον επιβλέποντα καθήγητη της παρούσας πτυχιακής Γεώργιο Παυλίδη που με καθοδήγησε να την ολοκλήρησω , και τους φίλους μου Αλέξανδρο, Θεόδωρο, Αναστάση και την αδερφή μου που με βοήθησαν στις δοκιμές της εφαρμογής.

Περίληψη

Στο πλαίσιο της παρούσας εργασίας αναπτύσσεται σύστημα διασύνδεσης σήμανσης με QR Codes με διαδικτυακές βάσεις δεδομένων με στόχο την παρουσίαση πληροφόρησης σε έξυπνες φορητές συσκευές που χρησιμοποιούν το λειτουργικό σύστημα Android. Για τη δημιουργία χρήσιμης εφαρμογής πάνω σε πραγματικά δεδομένα επιλέχθηκε η χρήση σήμανσης QR Codes που είχε ήδη εγκατασταθεί σε ιστορικά κτήρια της παλιάς πόλης της Ξάνθης από το Ερευνητικό Κέντρο «Αθηνά», η οποία συνοδεύονταν από αντίστοιχη τεκμηρίωση, κατά κύριο λόγο, για τουριστική χρήση. Έτσι, αναπτύχθηκε διαδικτυακή σχεσιακή βάση δεδομένων που συμπεριέλαβε το πολιτιστικό περιεχόμενο της τεκμηρίωσης και η αντίστοιχη εφαρμογή για κινητές συσκευές Android η οποία αναγνωρίζει μέσω κάμερας τους κώδικες και διασυνδέει με τη βάση δεδομένων (με κλήσεις SQL) παρέχοντας σε κάθε επισκέπτη της παλιάς πόλης της Ξάνθης τουριστική ενημέρωση για κάποιο κτήριο ενδιαφέροντος. Η βάση δεδομένων είναι ενεργή στο διαδίκτυο και η εφαρμογή είναι διαθέσιμη online στο Google Play Store στις διευθύνσεις:

(web) http://play.google.com/store/apps/details?id=com.bill.qrxanthiold

(android) market://details?id=com.bill.qrxanthiold

Abstract

In this work an integrated system for the connection of QR Code signage with web databases has being developed in order to provide information in smart mobile devices operating on Android OS. To be practical and useful on real-world data, the existing QR Code signage on historic buildings in the old town of Xanthi has been selected as a case study. This signage was developed by the "Athena" Research Center and was accompanied by documentation provided online mainly for touristic purposes. A relational database has been developed to include the cultural content of the documentation and an Android app has been implemented to provide connection between the QR Code signage with the database (through SQL calls) and to provide tourist information to the visitors of the old town of Xanthi. The database is online and the app is available through the Google Play Store at:

(web) http://play.google.com/store/apps/details?id=com.bill.qrxanthiold

(android) market://details?id=com.bill.qrxanthiold

Πρόλογος

Οι Quick Response Codes ή QR Codes, όπως έχει επικρατήσει να αναφέρονται, αποτελούν την εξέλιξη των κωδίκων τύπου bar-codes και εισήχθησαν για την κάλυψη των αυξημένων σύγχρονων απαιτήσεων σε πιο περιεκτικούς κώδικες αναγνώρισης πάνω σε προϊόντα αλλά και σε οποιαδήποτε αντικείμενα ανάλογα με την εφαρμογή. Οι QR Codes επεκτείνουν τον κώδικα αναγνώρισης των Bar Codes (που είναι μονοδιάστατοι) εκμεταλλευόμενοι τη δεύτερη διάσταση, δημιουργώντας έτσι μια δυαδική εικόνα, η οποία με αποδοτικό τρόπο κωδικοποιεί την πληροφορία που η εκάστοτε εφαρμογή απαιτεί.

Στόχος της χρήσης τους στο πλαίσιο της παρούσας εργασίας είναι η ανάπτυξη εφαρμογής οπτικής αναγνώρισης QR Codes των οποίων το περιεχόμενο οδηγεί στη δημιουργία εντολής SQL για την αποστολή ερωτήματος σε διαδικτυακή βάση δεδομένων ή πύλη και η παρουσίαση πολυμεσικού δελτίου αποτελέσματος του ερωτήματος μέσα σε περιβάλλον λειτουργικού συστήματος Android για έξυπνα κινητά τηλέφωνα.

Η υλοποίηση του συστήματος που αποτελεί αντικείμενο της παρούσας εργασίας απαιτεί την ενασχόληση το ειδικό θέμα του προγραμματισμού έξυπνων κινητών συσκευών με λειτουργικό σύστημα Android που βασίζεται σε γλώσσα προγραμματισμού Java, ενώ παράλληλα απαιτείται και η ενασχόληση με τη σύνδεση και λειτουργία με διαδικτυακές βάσεις δεδομένων/πύλες μέσω SQL ερωτημάτων.

Στα κεφάλαια που ακολουθούν αναπτύσσεται τόσο το υπόβαθρο όσο και η περιγραφή του συστήματος που αναπτύχθηκε και πληροί τις απαιτήσεις που τέθηκαν αρχικά. Αρχικά της παρουσιάζονται πληροφορίες για το QR Code, τα τεχνικά χαρακτηριστικά του και πώς μπορεί να χρησιμοποιηθεί. Στη συνέχεια παρουσιάζονται πληροφορίες για την αρχιτεκτονική, τις χαρακτηριστικές λειτουργίες, τις βασικές αρχές εφαρμογών και τα βασικά συστατικά των εφαρμογών για το λειτουργικό σύστημα Android. Στο κύριο μέρος της εργασίας περιγράφεται το σύστημα που αναπτύχθηκε, ενώ στα Παραρτήματα παρατίθεται ο πηγαίος κώδικας της εφαρμογής καθώς και συνοπτικές βοηθητικές οδηγίες.

Βασίλης Τσορμπατζούδης
Σέρρες, 2014

ΕΙΚΟΝΕΣ

Εικόνα 1. Η δομή του QR Code 18
Εικόνα 2. Πρότυπο εντοπισμού 18
Εικόνα 3. Πρότυπο ευθυγράμμισης 19
Εικόνα 4. Το πρότυπο χρονισμού 19
Εικόνα 5. Εκδόσεις των QR Codes 20
Εικόνα 6. Αναλογίες του πρότυπου εντοπισμού 21
Εικόνα 7. Αναγνώριση μεγέθους και γωνιάς 21
Εικόνα 8. Παραμορφωμένα QR Codes 21
Εικόνα 9. Παράδειγμα κατεστραμμένων QR Codes 22
Εικόνα 10. Διαιρούμενο QR Code 23
Εικόνα 11. Διαδικασία δημιουργίας και ελέγχου μασκών 23
Εικόνα 12. Απευθείας εκτύπωση ή εγχάραξη 24
Εικόνα 13. Λογότυπο Android 29
Εικόνα 14. Αρχιτεκτονική του Android 31
Εικόνα 15. Κύκλος ζωής δραστηριότητας 46
Εικόνα 16. Κύκλος ζωής υπηρεσίας 47
Εικόνα 17. Ιεραρχία ViewGroups και Views 49
Εικόνα 18. Βασική δομή συστήματος 53
Εικόνα 19. Απλοποιημένο διάγραμμα των δραστηριοτήτων ροής 55
Εικόνα 20. Ιεραρχία Δραστηριοτήτων 60
Εικόνα 21 Προσχέδιο της αρχικής οθόνης 61
Εικόνα 22 Η οθόνη της δραστηριότητας MainActivity 61
Εικόνα 23 Λογότυπο του ZXing project 62
Εικόνα 24. Η οθόνη της CaptureActivity 63
Εικόνα 25. Η Οθόνη της δραστηριότητας HistoryActivity 63
Εικόνα 26 Η οθόνη της HelpActicity 64
Εικόνα 27. Διάγραμμα ροής της ResultsActivity από την CaptureActivity 66
Εικόνα 28. Διάγραμμα ροής της ResultsActivity από την HistoryActivity 67
Εικόνα 29. Παράδειγμα σελίδας παρουσίασης κτιρίου 67

ΠΙΝΑΚΕΣ

Πίνακας I. QR Code χωρητικότητα δεδομένων 17
Πίνακας II. Δομή του πίνακα buildings 58
Πίνακας III Βάση δεδομένων συσκευής DBXanthi..... 59

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΕΦΑΛΑΙΟ I QR CODE 17
1.1 Το QR Code 17
1.2. Η δομή του QR Code 18
1.3 Οι εκδόσεις του QR Code 20
1.4 Χαρακτηριστικά του QR Codes 20
1.5 Προτυποποίηση και άδεια χρήσης 24
1.6 QR Code και Android 25
1.7 Χρήσεις του QR Code 25
ΚΕΦΑΛΑΙΟ II ANDROID 29
2.1 Τι είναι το Android 29
2.2 Χαρακτηριστικά και λειτουργίες του Android 30
2.3 Αρχιτεκτονική του Android 30
2.4 Οι εκδόσεις του Android 33
2.5 Βασικές αρχές εφαρμογών Android 40
2.6 Στοιχεία εφαρμογών Android 41
2.7 Κύκλος ζωής των εφαρμογών Android 44
2.7.1 Κύκλος ζωής δραστηριότητας 44
2.7.2 Κύκλος ζωής υπηρεσίας 46
2.7.3 Κύκλος ζωής Broadcaster receiver 48
2.8 Το αρχείο AndroidManifest.xml 48
2.9 Η διεπαφή χρήστη του Android 49
2.10 Υπηρεσίες διαδικτύου (Web Services) 50
2.11 JSON 51
ΚΕΦΑΛΑΙΟ III ΔΙΑΣΥΝΔΕΣΗ QR CODES ΚΑΙ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ ΣΕ ΦΟΡΗΤΕΣ ΣΥΣΚΕΥΕΣ 53
3.1 Εισαγωγή 53
3.2 Προδιαγραφές της εφαρμογής "*Old Town of Xanthi Guide*" 54
3.3 Η υλοποίηση της εφαρμογής "*Old Town of Xanthi Guide*" 57
ΒΙΒΛΙΟΓΡΑΦΙΑ 69
ΠΑΡΑΡΤΗΜΑ Α – ΠΗΓΑΙΟΣ ΚΩΔΙΚΑΣ ΕΦΑΡΜΟΓΗΣ 71
ΠΑΡΑΡΤΗΜΑ Β – ΟΔΗΓΙΕΣ ΕΓΚΑΤΑΣΤΑΣΗΣ ANDOID ADT 83
Εγκατάσταση του Android SDK Bunlde 83

Κεφάλαιο I
QR Code

1.1 Το QR Code

Το QR Code είναι ένας ραβδοκώδικας (barcode) δύο διαστάσεων [1],[2],[4]. Δημιουργήθηκε το 1994 από την εταιρεία Denso, θυγατρική της Toyata και εγκρίθηκε από τον παγκόσμιο οργανισμό τυποποίησης ως πρότυπο τον Ιούνιο του 2004 (ISO/IEC18004). Το QR Code δημιουργήθηκε αρχικά, για τον έλεγχο παραγωγής τμημάτων στην βιομηχανία αυτοκίνητων, αλλά η χρήση του εξαπλώθηκε γρήγορα και σε άλλους τομείς. Το QR προέρχεται από τα αρχικά των λέξεων Quick Response, δηλαδή γρήγορη απόκριση, η οποία σχετίζεται με το γεγονός ότι οι δημιουργοί του είχαν ως κύριο σκοπό τα περιεχόμενα που περιέχονται στον κώδικα να αποκωδικοποιούνται με μεγάλη ταχύτητα.

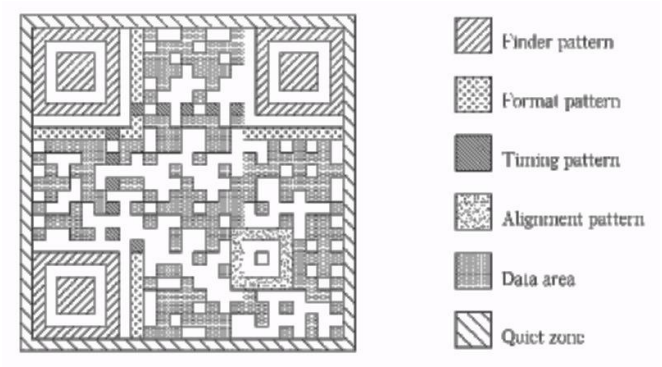
Σε αντίθεση με την τυπική μορφή αποθήκευσης δεδομένων των μονοδιάστατων ραβδοκωδικών (barcodes), τα οποία μπορούν να αποθηκεύσουν έως και 20 ψηφία τα QR Codes μπορούν να επεξεργαστούν από μερικές δεκάδες bytes δεδομένων έως και μερικές χιλιάδες (βλπ. Πίνακας I). Έχουν τη δυνατότητα να συμπεριλαμβάνουν όλα τα είδη δεδομένων, όπως αριθμούς και γράμματα, σύμβολα Kanji, Kana, Hiragana και δυαδικούς αριθμούς. Μπορούν να κωδικοποιούνται μέχρι και 7.089 χαρακτήρες σε ένα QR Code.

Πίνακας I. QR Code χωρητικότητα δεδομένων

Τύπος δεδομένων	Μονάδα μέτρησης σε χαρακτήρες
Αριθμοί	Μέγιστο: 7089 χαρακτήρες
Αλφαριθμητικά	Μέγιστο: 4296 χαρακτήρες
Δυαδικοί (8 bits)	Μέγιστο: 2953 χαρακτήρες
Kanji	Μέγιστο: 1817 χαρακτήρες

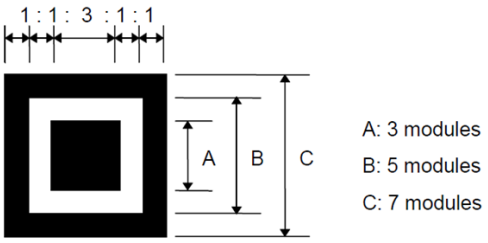
1.2. Η δομή του QR Code

Το QR Code [2],[3] είναι ένας τετραγωνικός πίνακας, που απαρτίζεται από άσπρα και μαύρα τετράγωνα κελιά, τα οποία ονομάζονται modules. Κάθε QR Code αποτελείται από δύο κύριες περιοχές, την περιοχή που κωδικοποιεί τα δεδομένα (data area) και τις περιοχές που τοποθετούνται τα πρότυπα λειτουργίας. Επίσης υπάρχει μια ουδέτερη περιοχή (Quiet zone) που είναι εξωτερικά του QR Code. Τα πρότυπα λειτουργίας που διαθέτει είναι το πρότυπο εντοπισμού θέσης (finder pattern), το πρότυπο ευθυγράμμισης (alignment pattern), και το πρότυπο χρονισμού (timing pattern). Ο ρόλος των προτύπων λειτουργίας είναι να βοηθούν στον εντοπισμό και στην ανάγνωση του QR Code. Στην Εικόνα 1 παρουσιάζεται η δομή ενός τυπικού QR Code.



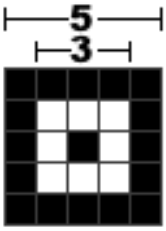
Εικόνα 1. Η δομή του QR Code

Το **πρότυπο εντοπισμού** (finder pattern) χρησιμοποιείται για τον εντοπισμό της θέσης του QR Code. Η θέση, η γωνιά και το μέγεθος του QR Code υπολογίζεται με τη βοήθεια των τριών προτύπων εντοπισμού, που είναι τοποθετημένα στην αριστερά-πάνω γωνία, στην αριστερά-κάτω γωνία και στη δεξιά-πάνω γωνία του κώδικα. Τα πρότυπα εντοπισμού μπορούν να εντοπιστούν εύκολα από όλες τις κατευθύνσεις. Στη Εικόνα 2 παρουσιάζεται η δομή του πρότυπου εντοπισμού.



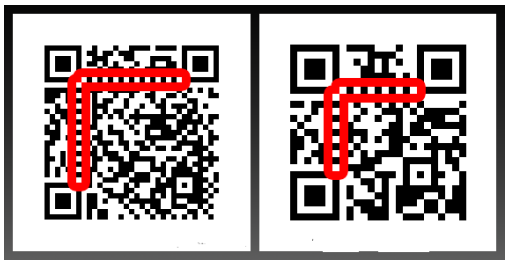
Εικόνα 2. Πρότυπο εντοπισμού

Το **πρότυπο ευθυγράμμισης** (alignment pattern) χρησιμοποιείται για τη διόρθωση των παραμορφωμένων QR Codes. Το πρότυπο αυτό δε συμπεριλαμβάνονταν στην πρώτη έκδοση του QR Code. Το πρότυπο ευθυγράμμισης αποτελείται από 5x5 μαύρα κελιά, 3x3 άσπρα κελιά και ένα μαύρο κελί στο κέντρο (Εικόνα 3). Η κεντρική περιοχή του πρότυπου αναγνωρίζεται για να διορθωθεί η παραμόρφωση στο QR Code.



Εικόνα 3. Πρότυπο ευθυγράμμισης

Το **πρότυπο χρονισμού** αποτελείται από οριζόντιες και κάθετες διακεκομμένες γραμμές που είναι τοποθετημένες ανάμεσα στα τετράγωνα του πρότυπου εντοπισμού θέσης (Εικόνα 4). Αυτές οι γραμμές εξυπηρετούν στον υπολογισμό του κέντρου κάθε κελιού και στον εντοπισμό ενδεχομένως κάποιου είδους παραμόρφωσης.



Εικόνα 4. Το πρότυπο χρονισμού

Το QR Code απαιτεί μία **ουδέτερη περιοχή** (Quiet area). Η ουδέτερη περιοχή βρίσκεται εξωτερικά του κώδικα σε όλες τις πλευρές του και χρησιμοποιείται για την εύκολη και γρήγορη ανάγνωση του QR Code. Η περιοχή καταλαμβάνει το μέγεθος τεσσάρων κελιών προς όλες τις κατευθύνσεις του κώδικα.

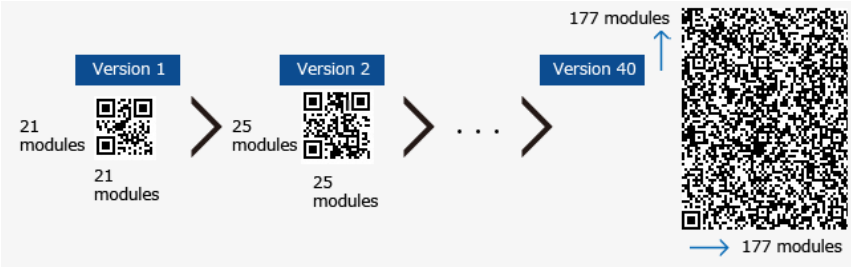
Οι **πληροφορίες τύπου** (Format Information) προσδιορίζουν την έκδοση του κώδικα, το επίπεδο ελέγχου σφαλμάτων και την ακολουθία μάσκας που χρησιμοποιούνται για το κώδικα. Επιπλέον αυτή η περιοχή είναι η πρώτη που αναγνωρίζεται κατά τη διαδικασία αποκωδικοποίησης.

Η **περιοχή δεδομένων** (Data area) είναι η περιοχή που τοποθετείται η πληροφορία που περιέχει το QR Code. Τα δεδομένα κωδικοποιούνται με

τον αλγόριθμο Reed-Solomon. Η κωδικοποίηση Reed-Solomon είναι μια μαθηματική μέθοδος κωδικοποίησης που εξασφαλίζει τη δυνατότητα διόρθωσης σφαλμάτων.

1.3 Οι εκδόσεις του QR Code

Το QR Code έχει σαράντα εκδόσεις. Κάθε έκδοση απαρτίζεται από διαφορετικό αριθμό τμημάτων (Modules). Η πρώτη έκδοση του QR Code αποτελείται από 21x21 τμήματα και η τελευταία έκδοση αποτελείται από 177x177 (Εικόνα 5). Όσο αυξάνεται ο αριθμός έκδοσης του QR Code προστίθενται τέσσερα επιπλέον τμήματα σε κάθε πλευρά [5].



Εικόνα 5. Εκδόσεις των QR Codes

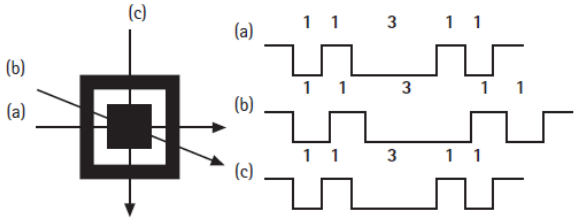
Η μέγιστη χωρητικότητα μιας έκδοσης του QR Code εξαρτάται από το μέγεθος δεδομένων, τον τύπο των δεδομένων, και το επίπεδο διόρθωσης σφαλμάτων.

1.4 Χαρακτηριστικά του QR Codes

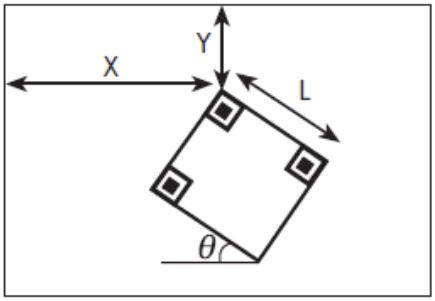
Το QR Code διαθέτει όλα τα χαρακτηριστικά των δυσδιάστατων κωδίκων. Μερικά από τα χαρακτηριστικά που το κάνουν ανώτερο από αυτά είναι [2],[3]:

Γρήγορη ανάγνωση ανεξάρτητα προσανατολισμού (360 μοίρες): Η ανάγνωση ενός δυσδιάστατου κώδικα μπορεί να γίνει με τη χρήση ενός αισθητήρα CCD (τυπικός αισθητήρας ψηφιακής φωτογραφικής μηχανής). Τα δεδομένα από τη λήψη του αισθητήρα αποθηκεύονται στη μνήμη. Με τη χρήση λογισμικού εντοπίζονται τα πρότυπα εντοπισμού θέσης, ώστε να βρεθεί η θέση, η γωνία και το μέγεθος του κώδικα για να ακολουθήσει η διαδικασία αποκωδικοποίησης. Η διαδικασία εντοπισμού χρειάζεται αρκετό χρόνο για να ολοκληρωθεί. Το QR Code διαθέτει πρότυπα εντοπισμού τοποθετημένα στις τρείς γωνιές του και επιτρέπει τη γρήγορη εύρεση και ανάγνωσή του από οποιαδήποτε γωνία και να έχει γίνει η λήψη του από τον αισθητήρα. Η αναλογία του μαύρου και του άσπρου στο πρότυπου εντοπισμού είναι πάντοτε 1:1:3:1:1 από οποιαδήποτε γωνία (Εικόνα 6). Αναζητώντας την αναλογία αυτή, εντοπίζεται το πρότυπο εντοπισμού θέσης στην εικόνα που έχει ληφθεί από τον αισθητήρα, με α-

ποτέλεσμα να γίνει ο εντοπισμός της θέσης του QR Code σε μικρό χρονικό διάστημα. Επίσης με την αναγνώριση των σχετικών θέσεων των τριών προτύπων εντοπισμού επιτυγχάνεται η ταυτόχρονη αναγνώριση του μεγέθους (L), της γωνίας (θ) και του σχήματος του QR Code. Τοποθετώντας τα πρότυπα εντοπισμού στις τρείς γωνιές του κώδικα επιταχύνεται η αποκωδικοποίηση είκοσι φόρες γρηγορότερα (Εικόνα 7). Η διαδικασία αναγνώρισης των προτύπων εντοπισμού μπορεί να υλοποιηθεί σε επίπεδο υλικού (Hardware), για να επιταχυνθεί.



Εικόνα 6. Αναλογίες του πρότυπου εντοπισμού



Εικόνα 7. Αναγνώριση μεγέθους και γωνιάς

Εικόνα 8. Παραμορφωμένα QR Codes

Αντοχή σε παραμορφώσεις: Οι κώδικες που τοποθετούνται σε εύκαμπτες επιφάνειες ή όταν η λήψη τους γίνει υπό γωνία είναι δυνατό να εμφανιστούν με παραμορφώσεις κατά τη φωτογράφιση (Εικόνα 8). Η παραμόρφωση που υφίσταται ο κώδικας QR Code μπορεί να διορθωθεί από τα πρότυπα ευθυγράμμισης, τα οποία υπάρχουν ανά περιοχές στον κώδικα.

Υπολογίζεται η διάφορα μεταξύ της κεντρικής θέση του προτύπου ευθυγράμμισης που υπολογίζεται από το εξωτερικό σχήμα του κώδικα και την πραγματική κεντρική θέση του προτύπου ευθυγράμμισης, για να βρεθεί το κέντρο όλων των κελιών. Με αυτή την μέθοδο οι παραμορφώσεις διορθώνονται και το QR Code γίνεται αναγνώσιμο.

Διόρθωση σφαλμάτων (Error Correction functionality): Οι QR Codes έχουν τη δυνατότητα να επαναφέρουν τα αρχικά δεδομένα σε περίπτωση που ο κώδικας έχει καταστραφεί ή λερωθεί. Στην Εικόνα 9 φαίνονται παραδείγματα κατεστραμμένων QR Codes.

Εικόνα 9. Παράδειγμα κατεστραμμένων QR Codes

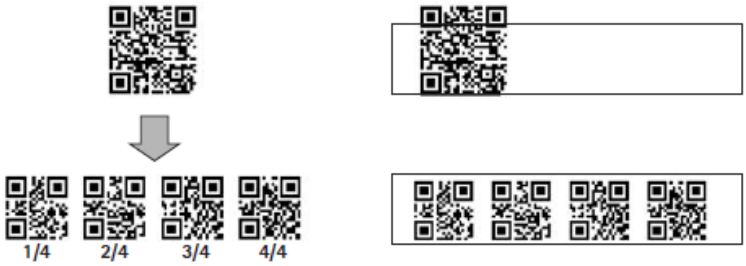
Η διόρθωση σφαλμάτων επιτυγχάνεται με την προσθήκη κώδικα Reed Solomon στα αρχικά δεδομένα. Υπάρχουν τέσσερα διαφορετικά επίπεδα διόρθωσης σφαλμάτων:

- L: περίπου 7%
- M: περίπου 15%
- Q: περίπου 25%
- H: περίπου 30%

Οι χρήστες έχουν τη δυνατότητα επιλογής ανάλογα με τις ανάγκες τους ή το λειτουργικό τους περιβάλλον. Το επίπεδο M είναι αυτό το οποίο επιλέγεται συχνότερα. Το επίπεδο διόρθωσης σφαλμάτων θα πρέπει να επιλέγεται σύμφωνα με τον αριθμό σφαλμάτων που πρέπει να διορθωθούν. Για παράδειγμα εάν πρέπει να διορθωθούν 50 από τις 100 λέξεις κώδικα, χρειάζονται 100 λέξεις του κώδικα Reed Solomon καθώς το ποσό των λέξεων του Reed Solomon κώδικα πρέπει να είναι διπλάσιο από αυτό που επιθυμεί ο χρήστης να διορθώσει.

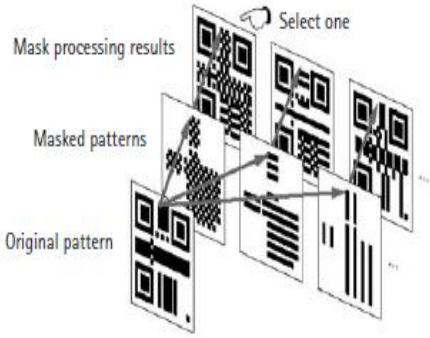
Διαίρεση ενός QR Code σε πολλά QR Codes (Linking functionality of QR Code): Ένα QR Code μπορεί να διασπαστεί σε πολλά μικρότερα σε μέγεθος QR Codes. Ο μέγιστος αριθμός QR Codes που μπορεί να διαιρεθεί ένα QR Code, είναι δεκαέξι. Στα QR Codes που δημιουργούνται από τη διαίρεση του αρχικού, τοποθετείται ένας δείκτης, ο οποίος παρέχει πληροφορίες, όπως το σύνολο των QR Codes που αποτελείται το διαιρεμένο QR Code και σε πια θέση είναι στην σειρά. Έτσι επιτυγχάνεται η

αποκωδικοποίηση του αρχικού QR Code, ανεξαρτήτως της σειράς που αναγνώστηκαν τα παράγωγα του. Εάν δεν υπάρχει αρκετός χώρος σε μία επιφάνεια, για να τυπωθεί ένα QR Code, τότε αυτό μπορεί να διαιρεθεί σε μικρότερου μεγέθους QR Codes, ώστε να χωρέσει στην επιφάνεια. Στην Εικόνα 10 φαίνεται ένα QR Code που διαιρέθηκε σε τέσσερα.



Εικόνα 10. Διαιρούμενο QR Code

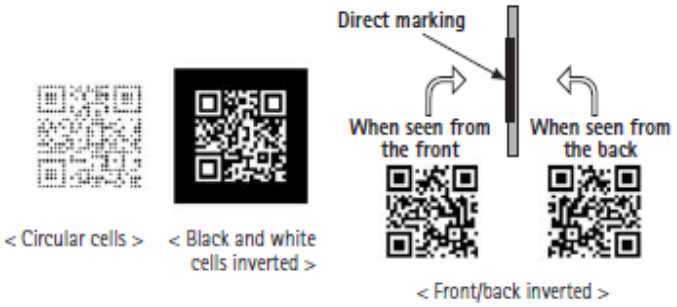
Masking Process: Το masking (Εικόνα 11) είναι μία τεχνική που επιτρέπει την εύκολη και γρήγορη ανάγνωση του QR Code. Η τεχνική αυτή τοποθετεί τα μαύρα και τα άσπρα κελιά με ομοιόμορφο τρόπο και εμποδίζει την παράγωγη διπλότυπων προτύπων. Επίσης εμποδίζει τη δημιουργία της αναλογίας 1:1:3:1:1 στην περιοχή δεδομένων, καθώς η αναλογία αυτή πρέπει να υπάρχει μόνο στο πρότυπο εντοπισμού θέσης, διαφορετικά θα δημιουργούσε ασάφειες στο σύστημα ανάγνωσης.



Εικόνα 11. Διαδικασία δημιουργίας και ελέγχου μασκών

Ευελιξία στη μορφοποίηση: Η λειτουργία αντιστροφής του χρώματος επιτρέπει στα QR Codes να τυπωθούν σε μαύρο φόντο αντιστρέφοντας τα άσπρα με μαύρα κελιά και αντίστροφα (Εικόνα 12). Επιπλέον υπάρχει η δυνατότητα να τυπωθούν τα κελιά σε κυκλικό σχήμα. Η λειτουργία αυτή είναι ιδιαίτερα χρήσιμη όταν πρόκειται να χαραχτεί ένα QR Code μόνιμα

σε μια επιφάνεια που δεν είναι ανοιχτόχρωμη. Στην περίπτωση που χρησιμοποιείται τεχνική laser για τη χάραξη του QR Code, τότε τα κελιά του έχουν κυκλικό σχήμα. Όταν το QR Code τυπώνεται σε διαφανείς επιφάνειες όπως το γυαλί, τότε η αποκωδικοποίηση μπορεί να γίνει και από την πίσω πλευρά.



Εικόνα 12. Απευθείας εκτύπωση ή εγχάραξη

1.5 Προτυποποίηση και άδεια χρήσης

Ο όρος QR Code αποτελεί κατοχυρωμένο σήμα κατατεθέν της εταιρείας Denso Wave Incorporated, η οποία διατηρεί την πατέντα της τεχνολογίας που χρησιμοποιείται [1],[4]. Παρόλα αυτά έχει διαθέσει στο κοινό τις σχετικές πληροφορίες που αφορούν στη δημιουργία και τη σάρωση προκειμένου να μπορέσουν να χρησιμοποιηθούν οι κώδικες από το ευρύ κοινό χωρίς χρέωση ή κάποιου είδους υποχρέωση προς την εταιρεία. Δεν απαιτείται κάποια ειδική άδεια για τη δημιουργία ή χρήση των QR Codes[1]. Τα πρότυπα που σχετίζονται με τους κώδικες QR Code είναι τα εξής:

- Αποδοχώ ως διεθνές πρότυπο των Automatic Identification Manufactures (AIM) με τίτλο ISS-QR Code, Οκτώβριος 1997
- Αποδοχή ως πρότυπο από την Japanese Electronic Industry Development Association, με τίτλο JEIDA-55, Μάρτιος 1998
- Αποδοχή ως Japanese Industrial Standard JIS X 0510, Ιανουάριος 1999
- Αποδοχή ως διεθνές πρότυπο International Organization for Standardization (ISO) με κωδική ονομασία ISO/IEC 18004, Ιούνιος 2000
- Αποδοχή του Micro QR Code ως πρότυπο Japanese Industrial Standard (JIS X 0510), Νοέμβριος 2004

[1] QR Code standardization, Denso Wave, http://www.qrcode.com/en/about/standards.html

- Αποδοχή από το διεθνή οργανισμό τυποποίησης GS1 ως πρότυπο για κινητά τηλέφωνα, Δεκέμβριος 2011

1.6 QR Code και Android

Η ανάγνωση του QR Code γίνεται εύκολα από κάποια κινητή συσκευή με λειτουργικό σύστημα Android που διαθέτει κάμερα. Υπάρχουν πολλές εφαρμογές στο εμπόριο για την ανάγνωση του QR Code. Το ZXing είναι λογισμικό ανοικτού κώδικα, γραμμένο στη γλώσσα προγραμματισμού Java και χρησιμοποιεί την ενσωματωμένη φωτογραφική μηχανή των κινητών συσκευών για τη φωτογράφιση και αποκωδικοποίηση μονοδιάστατων ή δισδιάστατων ραβδοκωδικών. Ανάμεσα στα διάφορα πρότυπα τα οποία υποστηρίζει το λογισμικό βρίσκεται και το QR Code. Οι περισσότερες συσκευές Android, αν όχι όλες, έχουν ήδη εγκατεστημένο το λογισμικό αυτό, αλλά και οι συσκευές που δεν το έχουν μπορούν εύκολα να το εγκαταστήσουν μέσω του Android Market. Επιπρόσθετα οι κατασκευαστές που δουλεύουν στην πλατφόρμα Android, έχουν πρόσβαση στον κώδικα του ZXing με αποτέλεσμα να μπορούν να τον ενσωματώσουν, είτε αυτούσιο, είτε τροποποιημένο, στην οποιαδήποτε εφαρμογή τους.

1.7 Χρήσεις του QR Code

Τα QR Codes μπορούν να χρησιμοποιηθούν όπως και όλοι οι κώδικες. Η διαφορά είναι ότι τα QR Codes επιτρέπουν πολύ περισσότερες εφαρμογές. Έχει περάσει πολύς καιρός από τότε που τα QR Codes αναπτύχθηκαν για να βοηθήσουν στον έλεγχο παραγωγής εξαρτημάτων στην αυτοκινητοβιομηχανία. Τα QR Codes μπορούν να βρεθούν τυπωμένα σε περιοδικά, αφίσες, φλιτζάνια, στην οθόνη της τηλεόρασης, σε βιβλία, παρουσιάσεις, πινακίδες σε στάσεις λεωφορείων, σε συσκευασίες προϊόντων και αλλά πολλά. Χρησιμοποιούνται σε πολλά μέρη και για διαφορετικούς λόγους. Μερικοί από τους λόγους χρήσης μπορεί να είναι:

- **Χρήση του QR Code στη διαφήμιση:** Τα QR Codes τυπώνονται συνήθως σε εφημερίδες, περιοδικά, πινακίδες, αφίσες ή και σε ένα προϊόν και περιέχουν ένα σύνδεσμο που παραπέμπει σε μία ιστοσελίδα που παρουσιάζει πληροφορίες για ένα γεγονός που θα γίνει όπως μία συναυλία, ή πληροφορίες για ένα προϊόν ή μία υπηρεσία.
- **Χρήση των QR Codes σε στάσεις λεωφορείων, σταθμούς τρένων και σταθμούς μετρό:** Με μια απλή λήψη του κώδικα, παρέχονται πληροφορίες για το πότε φθάνει το επόμενο λεωφορείο ή αμαξοστοιχία στη στάση.
- **Τυπωμένα δίπλα σε πίνακες ζωγραφικής ή αγάλματα σε χώρους μουσείων:** Οι επισκέπτες σε ένα μουσείο έχουν τη δυνατότητα μέσω του QR Code να έχουν πρόσβαση σε λεπτομερείς πληροφορίες για το έργο, όπως η χρονολογία κατασκευής του, το

βιογραφικό του καλλιτέχνη ή, ακόμα, να προβούν στην άμεση αγορά ενός φωτοαντιτύπου ή αφίσας από το μαγαζί του μουσείου.

- **Σε ιστορικά μνημεία ή διάφορα αξιοθέατα μέσα σε πόλεις ή στην εξοχή:** παραπέμπει σε κάποια ιστοσελίδα με περισσότερες πληροφορίες.
- **Πάνω σε πινακίδες πωλητηρίων:** Διάθεση όλων των πληροφοριών που αφορούν στο συγκεκριμένο πωλητήριο.
- **Ηλεκτρονική μάθηση (E-Learging):** το QR Code περιέχει ένα μήνυμα ηλεκτρονικού ταχυδρομείου, το οποίο ξεκινά μια αυτόματη απάντηση, στέλνοντας ηλεκτρονικό μήνυμα με πληροφορίες για το καθημερινό μάθημα.
- **Διπλά σε συσκευασίες τροφίμων:** πρόσβαση σε πληροφορίες για τους καταναλωτές σε συνταγές μαγειρικής που περιέχουν τα συστατικά που υπάρχουν στο ράφι.
- **Στα συνέδρια:** Περιέχει πληροφορίες για την ομιλία όπως ημερομηνία έναρξης, τα βιογραφικά των ομιλητών, κοκ.
- **Σε κοσμήματα:** λεπτομέρειες για το κόσμημα, το δημιουργό, την τιμή κλπ.
- **Κάτω από ένα άρθρο σε εφημερίδα ή περιοδικό:** παραπομπή σε online μορφή του άρθρου, όπου υπάρχουν τα σχόλια των αναγνωστών.
- **Σε μέρη που ακούγεται μουσική:** Εφαρμογές που αναγνωρίζουν μουσικά κομμάτια (π.χ. Shazam) χρησιμοποιούνται δύσκολα σε αυτά τα μέρη λόγω θορύβου. Με την τοποθέτηση ενός QR Code σε ένα video-wall, ο χρηστής μπορεί να μάθει πληροφορίες με το κομμάτι που ακούγεται.
- **Σε συνταγές μαγειρικής σε περιοδικά ή στην τηλεόραση:** παραπομπή σε ιστοσελίδα που περιέχει την συνταγή, βίντεο, κριτικές και σχόλια.

Παρακάτω παρουσιάζονται μερικά ενδεικτικά παραδείγματα χρήσεων των QR Codes στον κόσμο.

- **Χρήση του QR Code για παραγγελία προϊόντων**: Ο πελάτης σαρώνει το QR Code που είναι τυπωμένο σε ένα κατάλογο με προϊόντα, και η εφαρμογή στέλνει αυτόματα την παραγγελία στον πωλητή. Με αυτή την εφαρμογή βελτιώνεται η αποδοτικότητα της συναλλαγής μίας παραγγελίας.
- **Χρήση του QR Code σε κάρτα επιβίβασης διαρκείας στις συγκοινωνίες:** Ο χρήστης συμπληρώνει την πρώτη φόρα μία φόρμα για την έκδοση της κάρτας που θα χρησιμοποιηθεί για την μετακίνηση του στα μέσα μεταφοράς. Η κάρτα έχει τυπωμένο ένα QR Code που περιέχει πληροφορίες για τον κάτοχο και την διάρκεια που ισχύει. Για την ανανέωση της κάρτας χρειάζονται να συμπληρωθούν το χρονικό διάστημα που πρόκειται να καλύπτει η

νέα κάρτα και οι υπόλοιπες πληροφορίες λαμβάνονται από το QR Code της παλιάς κάρτας.

- **Χρήση του QR Code για τη διαχείριση δελτίων στοιχήματος:** Το QR Code τυπώνεται πάνω στο δελτίο και περιέχει πληροφορίες του στοιχήματος που παίχτηκε. Στο σημείο πώλησης των δελτίων υπάρχει συσκευή που μπορεί να διαβάσει το τυπωμένο QR Code και να ενημερώσει το χρήστη αν το δελτίο κερδίζει ή χάνει. Παλιότερα χρησιμοποιούνταν μαγνητικές κάρτες για τη λειτουργία αυτή. Με τη χρήση του QR Code χρησιμοποιείται πλέον απλό χαρτί, με αποτέλεσμα τη μείωση του κόστους έκδοσης του δελτίου και της επιβάρυνσης του περιβάλλοντος εξαιτίας της ανακύκλωσης του χαρτιού.
- **Χρήση του QR Code για την διαχείριση επιβατών σε κρουαζιερόπλοιο:** Το QR Code τυπώνεται επάνω στα εισιτήρια, τα οποία περιέχουν τον αριθμό διαβατήριου, ονοματεπώνυμο και διεύθυνση. Το τυπωμένο QR Code χρησιμοποιείται ως πιστοποιητικό, όταν επιστρέφεται το διαβατήριο του επιβάτη κατά την αποβίβαση του. Επίσης χρησιμοποιείται σαν εισιτήριο σε δραστηριότητες κατά την κρουαζιέρα όπως πρόσβαση σε φαγητό και ποτά.
- **Χρήση του QR Code για την αναγνώριση ασθενών σε νοσοκομείο**: Τα νοσοκομεία στην Ιαπωνία, Ταϊβάν και Σιγκαπούρη τυπώνουν ένα QR Code σε βραχιολί που τοποθετείται στο χέρι του ασθενή για την αναγνώριση του. Οι πληροφορίες που περιέχει το QR Code, είναι το όνομα του, ένας αναγνωριστικός αριθμός, η ημερομηνία γεννήσεως, το φύλο του ασθενή, τον θάλαμο και τον αριθμό κρεβατιού. Με την χρήση του QR Code εξασφαλίζεται ότι ο σωστός ασθενής παίρνει την κατάλληλη θεραπεία και τα σωστά φάρμακα την ώρα που πρέπει να τα πάρει.

Κεφάλαιο ΙΙ
Android

2.1 Τι είναι το Android

Το Android είναι ένα λειτουργικό σύστημα για συσκευές κινητής τηλεφωνίας το οποίο τρέχει τον πυρήνα του λειτουργικού συστήματος Linux. Αρχικά αναπτύχθηκε από την Google και αργότερα από την Open Handset Alliance. Επιτρέπει στους κατασκευαστές λογισμικού να συνθέτουν κώδικα με τη χρήση της γλώσσας προγραμματισμού Java, ελέγχοντας τη συσκευή μέσω βιβλιοθηκών λογισμικού ανεπτυγμένων από την Google [6].

Η πρώτη παρουσίαση της πλατφόρμας Android έγινε στις 5 Νοεμβρίου 2007, παράλληλα με την ανακοίνωση της ίδρυσης του οργανισμού Open Handset Alliance, μιας κοινοπραξίας 48 τηλεπικοινωνιακών εταιριών, εταιριών λογισμικού καθώς και κατασκευής hardware, οι οποίες είναι αφιερωμένες στην ανάπτυξη και εξέλιξη ανοιχτών προτύπων στις συσκευές κινητής τηλεφωνίας. Η Google δημοσίευσε το μεγαλύτερο μέρος του κώδικα του Android υπό τους όρους της Apache License, μιας ελεύθερης άδειας λογισμικού. Το λογότυπο για το λειτουργικό σύστημα Android είναι ένα ρομπότ σε χρώμα πράσινου μήλου και σχεδιάστηκε από τη γραφίστρια Irina Blok (Εικόνα 13).

Εικόνα 13. Λογότυπο Android

2.2 Χαρακτηριστικά και λειτουργίες του Android

Τα χαρακτηριστικά του Android [6] είναι τα εξής:

- Πλαίσιο εφαρμογών που επιτρέπει την επαναχρησιμοποίηση και αντικατάσταση συστατικών μερών.
- Εικονική μηχανή Dalvik που μεταγλωττίζει λογισμικό γραμμένο σε γλώσσα Java, σχεδιασμένη για φορητές συσκευές.
- Για την περιήγηση στον ιστό το Android διαθέτει περιηγητή βασισμένο στην ανοιχτή τεχνολογία WebKit.
- Αποστόλη μηνυμάτων SMS και MMS.
- Βελτιστοποιημένα γραφικά που υποστηρίζονται από μία 2D βιβλιοθήκη γραφικών, ενώ για τα 3D γραφικά βασίζονται στο OpenGL ES 1.0.
- Βάση δεδομένων SQLite για την αποθήκευση δεδομένων.
- Υποστήριξη πολυμέσων για την παράγωγη ήχου, παρουσίαση εικόνας και βίντεο: H.264,MPEG-4 SP, AMR, AMR-WB, AAC, HE-ACC, MP3, MIDI, OGG Vorbis, WAN, JPEG, PNG, GIF, BMP.
- Υποστήριξη κάμερας, οθόνης αφής, GPS, αισθητήρες επιτάχυνσης, μαγνητόμετρα.
- Πλούσιο περιβάλλον ανάπτυξης συμπεριλαμβανομένων ενός εξομοιωτή, εργαλεία για εκσφαλμάτωση, παρακολούθηση μνήμης και απόδοσης καθώς και ένα plugin για το περιβάλλον ανάπτυξης του Eclipse.
- Τεχνολογίες συνδεσιμότητας: Bluetooth, WI-FI, GSM/EDGE, CDMA, EV-DO και UMTS.
- Αγορά και εγκατάσταση εφαρμογών: Το Android Market είναι ένας κατάλογος εφαρμογών που μπορούν να μεταφορτωθούν και εγκατασταθούν στην συσκευή άμεσα μέσω ασύρματων καναλιών, χωρίς τη χρήση υπολογιστή. Αρχικά μόνο δωρεάν εφαρμογές ήταν δυνατόν να εγκατασταθούν. Εφαρμογές επί πληρωμή διαθέσιμες στο Android Market στις ΗΠΑ μετά από τις 19 Φεβρουαρίου 2009.

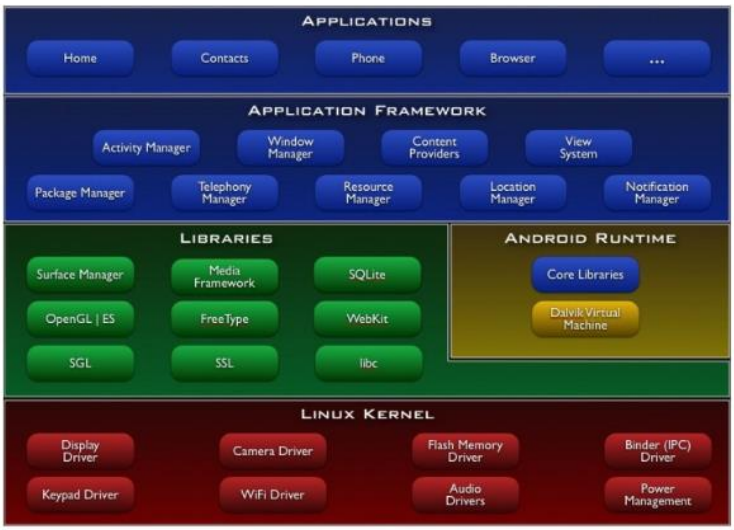
2.3 Αρχιτεκτονική του Android

Το Android αποτελείται από τα 5 επίπεδα που αποτελούν με τη σειρά τους τα κύρια συστατικά του λειτουργικού συστήματος. Τα επίπεδα είναι με σειρά από το υψηλότερο προς χαμηλότερο:

- Εφαρμογές
- Πλαίσιο των εφαρμογών
- Βιβλιοθήκες
- Περιβάλλον χρόνου εκτέλεσης

- Πυρήνας Linux

Στην Εικόνα 14 παρουσιάζεται η αρχιτεκτονική του Android, ενώ στη συνέχεια αναλύεται κάθε επίπεδο ξεχωριστά.



Εικόνα 14. Αρχιτεκτονική του Android

- **Εφαρμογές**: Το Android αποτελείται από ένα σύνολο εφαρμογών που περιλαμβάνει mail client, πρόγραμμα για αποστολή και λήψη SMS, ημερολόγιο, χάρτες, περιηγητή διαδικτύου, κατάλογο επαφών, κλπ. Όλες οι εφαρμογές είναι γραμμένες σε Java.
- **Πυρήνας**: Το Android είναι βασισμένο στα γέρα θεμέλια του Linux. Ο πυρήνας Linux είναι δοκιμασμένος, σταθερός και πετυχημένος και μπορεί να βρεθεί σε μεγάλη ποικιλία ηλεκτρικών συσκευών όπως από ρολόγια συσκευών μέχρι και υπερυπολογιστές. Το Android χρησιμοποιεί την έκδοση 2.6 του πυρήνα Linux για τις βασικές υπηρεσίες του συστήματος όπως η ασφάλεια, η διαχείριση μνήμης, η διαχείριση διεργασιών, η δικτύωση και το μοντέλο οδήγησης των συσκευών. Ο πυρήνας λειτουργεί και σαν αφαιρετικό επίπεδο μεταξύ του υλικού και του υπόλοιπου λογισμικού.
- **Εγγενείς Βιβλιοθήκες-Native Libraries**: Οι βιβλιοθήκες αυτές είναι γραμμένες στη γλώσσα προγραμματισμού C και C++ και μεταγλωττίστηκαν για τη συγκεκριμένη αρχιτεκτονική υλικού που χρησιμοποιείται από το τηλέφωνο. Οι βιβλιοθήκες αυτές δεν είναι εφαρμογές που μπορούν να σταθούν μόνες τους. Από την έκδοση Donut και μετά οι κατασκευαστές μπορούν να γράφουν

τις δικές τους βιβλιοθήκες με τη χρήση της εργαλειοθήκης NDK (Native Development Kit).

- **Περιβάλλον χρόνου εκτέλεσης-Android Runtime**: Στο ίδιο επίπεδο με τις εγγενείς βιβλιοθήκες, βρίσκουμε και το περιβάλλον χρόνου εκτέλεσης του Android. Εδώ υπάρχουν βασικές βιβλιοθήκες της Java και η εικονική μηχανή Dalvik. Η Dalvik είναι μια βελτιστοποιημένη υλοποίηση μιας εικονικής μηχανής Java για φορητές συσκευές από την Google. Η Dalvik τρέχει .dex αρχεία, τα οποία είναι bytecodes που προέρχονται από αρχεία .class και .jar. Τα .dex αρχεία είναι πολύ πιο συμπαγή και αποδοτικά, γεγονός σημαντικό για συσκευές με περιορισμένη μνήμη και μπαταρία. Υπάρχουν βασικές βιβλιοθήκες τις Java που υποστηρίζονται από το Android, ενώ κάποιες άλλες καθόλου, ενώ υπάρχουν και βιβλιοθήκες προσαρμοσμένες για το περιβάλλον Android.
- **Πλαίσιο Εφαρμογής-Application Framework**: Προσφέροντας μία ανοιχτή πλατφόρμα ανάπτυξης, το Android, προσφέρει τη δυνατότητα δημιουργίας εξαιρετικά πλούσιων και καινοτόμων εφαρμογών. Οι προγραμματιστές είναι ελεύθεροι να εκμεταλλευτούν όλα τα πλεονεκτήματα που τους προσφέρει το υλικό της συσκευής, να έχουν πρόσβαση σε πληροφορίες που έχουν σχέση με την τοποθεσία, να τρέξουν υπηρεσίες στο περιθώριο, να ορίσουν ειδοποιήσεις, να προσθέσουν σημειώσεις στη μπάρα κατάστασης κοκ. Οι προγραμματιστές έχουν πλήρη πρόσβαση στα ίδια APIs που χρησιμοποιούν οι βασικές εφαρμογές. Η αρχιτεκτονική των εφαρμογών είναι έτσι σχεδιασμένη ώστε να απλοποιεί την επαναχρησιμοποίηση των διάφορων συστατικών μερών. Οποιαδήποτε εφαρμογή μπορεί να θέσει διαθέσιμες προς χρήση τις δυνατότητές της και οποιαδήποτε άλλη εφαρμογή που μπορεί να κάνει χρήση αυτών των δυνατοτήτων. Όλες οι εφαρμογές αποτελούνται από ένα σύνολο υπηρεσιών και συστημάτων που περιλαμβάνουν:
    - Ένα πλούσιο και επεκτάσιμο σύνολο από Views που μπορεί να χρησιμοποιηθεί για το χτίσιμο εφαρμογών. Σε αυτό το σύνολο περιλαμβάνονται λίστες, πλέγματα, πλαίσια κειμένου, κουμπιά και ένας ενσωματωμένος περιηγητή διαδικτύου.
    - Διανομείς περιεχομένου (Content Providers) που επιτρέπουν στις εφαρμογές να έχουν πρόσβαση στα δεδομένα άλλων εφαρμογών ή να κοινοποιήσουν στις άλλες εφαρμογές τα δικά τους δεδομένα
    - Ένα διαχειριστή πόρων (Resource Manager) που παρέχει πρόσβαση σε πόρους όπως αλφαριθμητικά που έχουν σχέση με την τοποθεσία, τα γραφικά και τα αρχεία των σχεδιαγραμμάτων.

- Ένα διαχειριστή ειδοποιήσεων (Notification Manager) που επιτρέπει στις εφαρμογές να εμφανίζουν ειδοποιήσεις στην μπάρα κατάστασης.
- Ένα διαχειριστή δραστηριοτήτων (Activity Manager) που ελέγχει τον κύκλο ζωής των εφαρμογών και την εναλλαγή μεταξύ αυτών.

2.4 Οι εκδόσεις του Android

Η πλατφόρμα του Android [7] έχει υποστεί ένα σύνολο ενημερώσεων από την αρχική της έκδοση. Με κάθε νέα έκδοση επιλύονται προβλήματα, προστίθενται νέα χαρακτηριστικά και βελτιώνονται προϋπάρχοντα. Κάθε ενημέρωση του λειτουργικού συστήματος Android αναπτύσσεται κάτω από μία κωδική ονομασία που βασίζεται σε κάποιο επιδόρπιο, ενώ το αρχικό γράμμα της ονομασίας κάθε νέας έκδοσης είναι το επόμενο γράμμα της αλφαβήτου από αυτό που χρησιμοποιήθηκε στην ονομασία της προηγούμενης έκδοσης.

Ενημέρωση 1.0 (Apple Pie)

Η πρώτη εμπορική έκδοση της πλατφόρμας, εκδόθηκε στις 23 Σεπτεμβρίου του 2008 και ενσωμάτωνε τα ακόλουθα χαρακτηριστικά:

- Περιηγητής διαδικτύου με δυνατότητα ζουμ για την προβολή HTML και XHTML σελίδων.
- Υποστήριξη κάμερας.
- Υποστήριξη πρωτοκόλλων POP3, IMAP4, και SMTP.
- Συγχρονισμός του Gmail λογαριασμού με την εφαρμογή Gmail.
- Συγχρονισμός των επαφών του λογαριασμού Google με τις επαφές της συσκευής.
- Υπηρεσία άμεσων μηνυμάτων Google Talk.
- Υποστήριξη SMS και MMS.
- Αναπαραγωγή πολυμέσων.
- Ειδοποιήσεις στη μπάρα κατάστασης.
- Φωνητική κλήση. Εφαρμογή για το Youtube.
- Ξυπνητήρι, Αριθμομηχανή, Κλήσεις, Αρχική οθόνη, Συλλογή φωτογραφιών και ρυθμίσεις. Υποστήριξη Wi-Fi και Bluetooth.

Ενημέρωση 1.1 (Banana Bread)

Κυκλοφόρησε στις 9 Φεβρουαρίου του 2009 και επιδιόρθωσε κάποια προβλήματα, άλλαξε το API και προστέθηκαν ένα σύνολο νέων χαρακτηριστικών:

- Διαθέσιμες κριτικές όταν ο χρήστης αναζητεί επιχειρήσεις πάνω στον χάρτη.

- Μεγαλύτερος χρόνος απενεργοποίησης της οθόνης όταν πραγματοποιείται κλήση από το μεγάφωνο και δυνατότητα απόκρυψης του πληκτρολογίου κλήσης.
- Υποστήριξη επισυναπτόμενων αρχείων στα μηνύματα.
- Προσθήκη του χαρακτηριστικού marquee στα στοιχεία που απαρτίζουν τη διάταξη της διεπαφής χρήστη.

Ενημέρωση 1.5 (Cupcake)

Εκδόθηκε στις 30 Απριλίου του 2009, προστέθηκαν πολλά νέα χαρακτηριστικά και έγιναν αρκετές αναβαθμίσεις της διεπαφής χρήστη.

- Νέο εικονικό πληκτρολόγιο με δυνατότητα πρόβλεψης κειμένου.
- Υποστήριξη Widgets (μικρά πλαίσια εφαρμογών που μπορούν να ενσωματωθούν σε άλλες εφαρμογές όπως η Αρχική οθόνη.
- Αναπαραγωγή βίντεο αρχείων σε MPEG4 και 3GP μορφή.
- Αυτόματη σύζευξη με ακουστικό Bluetooth.
- Δυνατότητα αντιγραφής και επικόλλησης στον περιηγητή.
- Προβολή εικόνων στις αγαπημένες επαφές.
- Άμεση μετάβαση στα στοιχεία μίας επαφής από το ιστορικό κλήσεων.
- Κινούμενες μεταβάσεις της οθόνης.
- Δυνατότητα περιστροφής της οθόνης.
- Δυνατότητα ανεβάσματος βίντεο στο Youtube και εικόνας στο Picassa.

Ενημέρωση 1.6 (Donut)

Εκδόθηκε στις 15 Σεπτεμβρίου του 2009 και περιλαμβανόταν:

- Ολοκληρωμένη διεπαφή κάμερας, βιντεοκάμερας και έκθεση φωτογραφιών και βίντεο.
- Δυνατότητα μαζικής διαγραφή φωτογραφιών.
- Καλύτερη αναζήτηση ιστορικού και δυνατότητα προβολής στιγμιοτύπων των εφαρμογών στο Android Market.
- Αναβαθμισμένη φωνητική αναζήτηση, με μικρότερο χρόνο απόκρισης και καλύτερη συνεργασία με τις ενσωματωμένες εφαρμογές της συσκευής.
- Καλύτερη αναζήτηση ιστορικού, επαφών, αγαπημένων και στο διαδίκτυο μέσω της αρχικής οθόνης.
- Βελτιωμένη τεχνολογία υποστήριξης CDMA/EVDO, 802.1x, VPNs και προσθήκη μίας text-to-speech μηχανής.
- Υποστήριξη για ανάλυση οθόνης WVGA.
- Νέο πλαίσιο εφαρμογών Gesture και νέο εργαλείο ανάπτυξης GestureBuilder.

Ενημέρωση 2.0/2.1 (Éclair)

Εκδόθηκε στις 26 Οκτωβρίου του 2009 και περιλαμβάνει:

- Μεγαλύτερη ταχύτητα του υλικού.
- Υποστήριξη περισσότερων αναλύσεων και μεγεθών οθόνης.
- Ανανεωμένο γραφικό περιβάλλον.
- Νέο γραφικό περιβάλλον του περιηγητή και υποστήριξη HTML5.
- Νέα λίστα επαφών
- Καλύτερη αναλογία λευκού-μαύρου στο περιθώριο.
- Βελτιωμένη εφαρμογή Google-Maps 3.1.2.
- Υποστήριξη του Microsoft Exchange.
- Ενσωματωμένη υποστήριξη flash για την κάμερα.
- Ψηφιακό ζουμ.
- Κλάση MotionEvent για την αναγνώριση multi-touch ενεργειών.
- Βελτιωμένο εικονικό πληκτρολόγιο.
- Υποστήριξη Bluetooth 2.1.
- Ζωντανά Wallpapers.

Ενημέρωση 2.2.x (Froyo)

Εκδόθηκε στις 20 Μαΐου 2010 και περιλάμβανε τις εξής αλλαγές:

- Βελτιώσεις στην ταχύτητα του λειτουργικού, στη μνήμη και γενικά στην απόδοση του συστήματος.
- Επιπρόσθετες βελτιστοποιήσεις στην ταχύτητα των εφαρμογών εξαιτίας της υλοιποίησης του JIT (Just in Time compilation).
- Ενσωμάτωση τηςV8 JavaScript μηχανής του Chrome στην εφαρμογή του περιηγητή.
- Καλύτερη υποστήριξη του Microsoft Exchange.
- Βελτιωμένη εκκίνηση των εφαρμογών με συντομεύσεις για τις εφαρμογές του περιηγητή διαδικτύου και του τηλεφώνου.
- Υποστήριξη USB.
- Προσθήκη επιλογής για την απενεργοποίηση της πρόσβασης σε δεδομένα πάνω σε ένα κινητό δίκτυο.
- Αναβαθμισμένη Market εφαρμογή με ομαδική και αυτόματη ανανέωση εφαρμογών.
- Γρήγορη εναλλαγή μεταξύ πολλών γλωσσών πληκτρολογίου και των λεξικών τους.
- Φωνητική κλήση και διαμοίραση αρχείων μέσω Bluetooth.
- Υποστήριξη αριθμητικών και αλφαριθμητικών κωδικών.
- Δυνατότητα αναπαραγωγής GIFs αρχείων από τον περιηγητή.
- Δυνατότητα εγκατάστασης εφαρμογών στην επεκτάσιμη μνήμη.
- Υποστήριξη Adobe Flash 10.1.

Στις 18 Ιανουαρίου 2011 κυκλοφορεί η έκδοση 2.2.1, στις 22 Ιανουάριου κυκλοφορεί η 2.2.2 και τον Νοέμβρη του 2011 κυκλοφορεί η τελική έκδοση του Froyo 2.2.3.

Ενημέρωση 2.3.x (GingerBread)

Εκδόθηκε στις 6 Δεκεμβρίου του 2010 και περιλαμβάνει τις εξής αλλαγές:

- Ανανεωμένος σχεδιασμός του γραφικού περιβάλλοντος για πιο απλή και πιο γρήγορη πλοήγηση.
- Υποστήριξη για πολύ μεγάλες οθόνες και πού μεγάλες αναλύσεις.
- Υποστήριξη για Voip κλήσεις.
- Βελτιωμένο εικονικό πληκτρολόγιο μεγαλύτερης ακρίβειας και καλύτερης πρόβλεψης κειμένου.
- Ενισχυμένη λειτουργία αντιγραφής/επικόλλησης.
- Υποστήριξη Near Field Communication (NFC).
- Νέα εφέ ήχου όπως αντήχηση, εξίσωση, εξομοίωση ακουστικού και μπάσο.
- Νέο Download Manager για εύκολη πρόσβαση σε αρχεία που κατέβασε ο χρήστης.
- Δυνατότητα υποστήριξης πολλαπλών καμερών.
- Υποστήριξη WebM/VP8 αναπαραγωγή βίντεο και υποστήριξη της AAC κωδικοποίησης για αρχεία ήχου.
- Ταυτόχρονο garbage collection για βελτιωμένη απόδοση.

Ενημέρωση 3.x (Honeycomb)

Εκδόθηκε στις 22 Φεβρουαρίου του 2011, η οποία είναι η πρώτη έκδοση που απευθύνεται αποκλειστικά σε tablet φορητές συσκευές. Οι αλλαγές που πραγματοποιήθηκαν είναι:

- Σχεδιασμός του γραφικού περιβάλλοντος έτσι ώστε να είναι βέλτιστο για tablet συσκευές.
- Προσθήκη Action Bar, για πρόσβαση σε επιλογές περιεχομένου, πλοήγησης, widgets κ.α. στο πάνω μέρος της οθόνης.
- Προσθήκη System Bar, για γρήγορη πρόσβαση σε ειδοποιήσεις και εικονικά κουμπιά πλοήγησης στο κάτω μέρος της οθόνης.
- Απλουστευμένο σύστημα διαχείρισης πολλών διεργασιών.
- Επανασχεδιασμένο πληκτρολόγιο, για γρήγορη, αποτελεσματική και ακριβή πληκτρολόγηση σε μεγάλες οθόνες.
- Επανασχεδιασμένη και πιο απλή διεπαφή για αντιγραφή και επικόλληση.
- Υποστήριξη πολλαπλών καρτελών στον περιηγητή.
- Άμεση πρόσβαση στην κάμερα και στις δυνατότητες που προσφέρει.

- Δυνατότητα προβολής άλμπουμ και φωτογραφιών σε πλήρη οθόνη.
- Υποστήριξη βίντεο συνομιλίας μέσω Google Talk.
- Επιτάχυνση υλικού.
- Υποστήριξη πολυπύρηνων επεξεργαστών.
- Δυνατότητα κρυπτογράφησης των δεδομένων.

Ενημέρωση 4.0.x (Ice Cream Sanwich)

Η πρώτη έκδοση της 4.0.1 κυκλοφόρησε στις 19 Οκτώβρη του 2011 με αρκετά νέα χαρακτηριστικά:

- Εικονικά κουμπιά στο γραφικό περιβάλλον που αντικαθιστούν τα πραγματικά κουμπιά
- Διαχωρισμός των widgets σε νέα καρτέλα.
- Εύκολη δημιουργία φακέλων μέσω drag and drop μηχανισμού.
- Δυνατότητα πιο γρήγορης ή πιο αργής αναπαραγωγής των μηνυμάτων του τηλεφωνητή.
- Ενσωματωμένη λήψη screenshots.
- Βελτιωμένη λειτουργία διόρθωσης σφαλμάτων στην πληκτρολόγηση.
- Πρόσβαση στις εφαρμογές ακόμα και με κλειδωμένη οθόνη.
- Βελτιωμένη λειτουργία αντιγραφής/επικόλλησης.
- Υποστήριξη φωνητικής υπαγόρευσης σε πραγματικό χρόνο.
- Δυνατότητα ξεκλειδώματος της συσκευής μέσω λογισμικού αναγνώρισης προσώπου.
- Νέος περιηγητής με δυνατότητα υποστήριξης μέχρι και 16 καρτελών.
- Δυνατότητα επεξεργασίας εικόνων.
- Υποστήριξη Webp, αρχείων εικόνας.
- Δυνατότητα εγγραφής βίντεο σε 1080p.

Ακολούθησε η 4.0.3 στις 16 Δεκεμβρίου 2011 με αρκετές διορθώσεις bugs και βελτιώσεις, και στις 29 Μαρτίου το 2012 η έκδοση 4.0.4 με διορθώσεις και βελτιώσεις στην σταθερότητα του συστήματος, καλύτερη απόδοση κάμερας κ.α.

Ενημέρωση 4.1 (Jelly Bean)

Εκδόθηκε στις 9 Ιουνίου το 2012, βασισμένο πλέον στον πυρήνα Linux 3.0.31, κύριος στόχος της ενημέρωσης αυτής ήταν η βελτίωση των λειτουργιών και απόδοσης της διεπαφής. Μερικά από τα χαρακτηριστικά που προστέθηκαν:

- Πιο ελαφρά διεπαφή.
- Βελτιωμένη προσβασιμότητα.

- Υποστήριξη γραφής από δεξιά προς αριστερά για την υποστήριξη περισσοτέρων γλωσσών.
- Δυνατότητα εγκατάσταση πληκτρολογίων τρίτων.
- Επεκτάσιμες ειδοποιήσεις.
- Δυνατότητα ενεργοποίησης/απενεργοποίησης των ειδοποιήσεων εφαρμογής.
- Αυτόματη τακτοποίηση και αλλαγή μεγέθους των συντομεύσεων και των widget ώστε να επιτραπεί η πρόσθεση νέων.
- Μεταφορά Bluetooth δεδομένων για το Android Beam.
- Βελτιωμένη φωνητική αναζήτηση.
- Βελτιωμένη εφαρμογή κάμερας.
- Υψηλής ευκρίνειας φωτογραφίες από επαφές του Google+.
- Πολυκάναλος ήχος και έξοδο ήχου από το usb.
- Εφαρμογή Google Now φωνητικός βοηθός και αναζήτηση εφαρμογών.
- Αντικατάσταση του περιηγητή με τον περιηγητή Google Chrome.

Ενημέρωση 4.2 Jelly Bean (API level 17)

Η Google ανακοίνωσε τη νέα αυτή έκδοση μέσω ανακοίνωσης στον τύπο, χρησιμοποιώντας το slogan "A new flavor of Jelly Bean". Η έκδοση βασίζεται σε Linux kernel 3.4.0 και πρωτοεμφανίστηκε στις συσκευές Google Nexus 4 και Nexus 10, στις 13 Νοεμβρίου 2012. Κύρια νέα χαρακτηριστικά:

- Βελτιώσεις στη Lock screen με δυνατότητες για widget και απευθείας πρόσβαση στην κάμερα
- Γρήγορη πρόσβαση στις ενημερώσεις
- Πρφύλαξη οθόνης "Daydream" με παρουσίαση πληροφοριών
- Πολλαπλούς λογαριασμούς χρηστών (σε ταμπλέτες)
- Υποστήριξη ασύρματης παρουσίασης
- Βελτιώσεις προσβασιμότητας
- Νέα εφαρμογή clock με ενσωματωμένο world clock, χρονόμετρο και μετρητή χρόνου
- Όλες οι συσκευές χρησιμοποιούν την ίδια δομή οθόνης ανεξάρτητα μεγέθους
- Αναβάθμιση του συστήματος ενημέρωσης για πρόσβαση χωρίς άνοιγμα εφαρμογών
- SELinux
- Πάντα συνδεδεμένη δικτύωση VPN
- Βελτιομένη λειτουργικότητα επιβεβαίωσης SMS
- Group Messaging
- Διόρθωση σφαλμάτων και γενική αναβάθμιση απόδοσης

Ενημέρωση 4.3 Jelly Bean (API level 18)

Το slogan για τη νέα αυτή έκδοση ήταν "An even sweeter Jelly Bean" στις 24 Ιουλίου 2013 κατά τη διάρκεια ειδικού event στο San Francisco με την ονομασία "Breakfast with Sundar Pichai". Οι περισσότερες συσκευές Nexus έλαβαν την ενημέρωση εντός εβδομάδας, αν και οι ταμπλέτες Nexus 7 2ης γενιάς ήταν αυτές που επίσημα κυκλοφόρησαν με αυτή την έκδοση. Τα κυριότερα χαρακτηριστικά της έκδοσης ήταν:

- Υποστήριξη Bluetooth low energy (BLE)
- Υποστήριξη Bluetooth Remote Control Profile 1.3
- Υποστήριξη OpenGL ES 3.0 για προηγμένα γραφικά στα παιχνίδια
- Προστασία πρόσβασης για νέους χρήστες
- Βελτίωση της απόδοσης του συστήματος αρχείων
- Αυτόματη συμπλήρωση στην εφαρμογή πληκτρολόγησης τηλεφώνων
- Βελτιώσεις στο Photo Sphere
- Επανασχεδιασμός του περιβάλλοντος της κάμερας
- Προσθήκη του "App Ops", ενός συστήματος λεπτομερούς καθορισμού των δικαιωμάτων πρόσβασης
- Υποστήριξη ανάλυσης 4K
- Πολλαπλές βελτιώσεις ασφαλείας, απόδοσης και διόρθωση σφαλμάτων

Ενημέρωση 4.4 KitKat (API level 19)

Η Google ανακοίνωσε την έκδοση Android 4.4 KitKat στις 3 Σεπτεμβρίου 2013, μια έκδοση που αναμένονταν με πολύ ενδιαφέρον κυρίως από ους bloggers, οι οποίοι της απέδιδαν την αρίθμηση 5.0 και την επωνυμία "Key Lime Pie". Η έκδοση πρωτοχρησιμοποιήθηκε στο Google Nexus 5 στις 31 Οκτωβρίου 2013 και έχει βελτιστοποιηθεί ώστε να λειτουργεί σε μγαλύτερη γκάμα συσκευών. Τα κυριότερα χαρακτηριστικά της ενημέρωσης είναι:

- Ανανέωση του περιβάλλοντος αλληλεπίδρασης
- Ανανέωση της εφαρμογής Clock
- Δυνατότητα στις εφαρμογές να ενεργοποιούνται στη navigation και στη status bar
- Δυνατότητα στις εφαρμογές να χρησιμοποιούν τον "immersive mode" ώστε να καλύπτουν navigation και status bar
- Βελτιστοποιήσεις στην απόδοση συσκευών με χαμηλά χαρακτηριστικά
- Δυνατότητα ασύρματης εκτύπωσης
- Προσομοιωτής NFC για χρήση της συσκευής ως έξυπνη κάρτα

- Αφαίρεση υποστήριξης μη εξουσιοδοτημένων λογισμικών αναπαραγωγής Flash
- Εκτεταμμένη λειτουργικότητα για υπηρεσίες ενημέρωσης
- Υποστήριξη Bluetooth Message Access Profile (MAP)
- Βελτιώσεις στην κάμερα (auto focus, white balance, HDR+)
- Αφαίρεση της εφαρμογής ελέχου "App Ops"

2.5 Βασικές αρχές εφαρμογών Android

Οι εφαρμογές του Android [5] είναι γραμμένες σε Java. Ο μεταφρασμένος κώδικας με οποιαδήποτε άλλα δεδομένα και αρχεία που απαιτούνται από την εφαρμογή, μετασχηματίζονται σε ένα πακέτο Android, συμπιεσμένο σε αρχείο με την κατάληξη .apk. Αυτό το συμπιεσμένο αρχείο πρέπει να κατεβάσουν οι χρήστες για να εγκαταστήσουν την εφαρμογή στη συσκευή τους. Μετά την εγκατάσταση της εφαρμογής, αυτή είναι ανεξάρτητη και έχει τους δικούς της κανόνες ασφαλείας.

- Το λειτουργικό σύστημα Android είναι ένα σύστημα για πολλούς χρήστες και η κάθε εφαρμογή έχει το δικό της χρήστη.
- Εξ' ορισμού, το σύστημα αναθέτει σε κάθε εφαρμογή, ένα μοναδικό αναγνωριστικό χρήστη. Το σύστημα καθορίζει τα δικαιώματα των αρχείων της εφαρμογής, ώστε αυτά να είναι ορατά από την εφαρμογή που έχει το ίδιο αναγνωριστικό χρήστη.
- Κάθε διεργασία έχει τη δική της εικονική μηχανή και έτσι ο κώδικας της εφαρμογής είναι απομονωμένος από τον κώδικα οποιασδήποτε άλλης εφαρμογής.
- Εξ' ορισμού, κάθε εφαρμογή τρέχει σε δική της διεργασία. Το Android ξεκινάει τη διεργασία, όταν κάποιο μέρος του κώδικα της εφαρμογής πρέπει να εκτελεστεί και την τερματίζει όταν δεν είναι πλέον απαραίτητη και οι πόροι του συστήματος απαιτούνται από άλλες εφαρμογές.

Με αυτό το τρόπο το λειτουργικό σύστημα υλοποιεί την αρχή των ελάχιστων προνομίων (least privilege) για κάθε εφαρμογή. Αυτό σημαίνει, ότι κάθε εφαρμογή έχει πρόσβαση μόνο στα στοιχεία που χρειάζεται και πουθενά αλλού. Με αποτέλεσμα να δημιουργείται ένα ασφαλές περιβάλλον, όπου οι εφαρμογές δεν έχουν πρόσβαση σε στοιχεία του λειτουργικού συστήματος, τα οποία δεν έχουν εξουσιοδότηση να χρησιμοποιούν. Παρόλα αυτά υπάρχουν τρόποι οι εφαρμογές να ανταλλάσουν και να μοιράζουν πληροφορίες μεταξύ τους και να έχουν πρόσβαση σε υπηρεσίες του λειτουργικού συστήματος, όπως:

- Υπάρχει η δυνατότητα δύο εφαρμογές να μοιράζονται το ίδιο αναγνωριστικό χρήστη. Στην περίπτωση αυτή οι εφαρμογές έχουν κοινή πρόσβαση στα αρχεία. Για να διατηρηθούν πόροι του συ-

στήματος οι εφαρμογές μπορούν να τρέχουν στην ίδια διεργασία και να μοιραστούν την ίδια εικονική μηχανή (Virtual Machine).

- Μία εφαρμογή μπορεί να χρειάζεται πρόσβαση σε δεδομένα της συσκευής, όπως τις επαφές, τα μηνύματα (SMS) και άλλα. Η έγκριση δίνεται κατά την εγκατάσταση της εφαρμογής, όπου ζητείται η συναίνεση του χρήστη για την παροχή της κατάλληλης πρόσβασης στις άλλες εφαρμογές και δεδομένα.

2.6 Στοιχεία εφαρμογών Android

Ένα κύριο χαρακτηριστικό της πλατφόρμας του Android είναι ότι μία εφαρμογή μπορεί να χρησιμοποιήσει στοιχεία άλλων εφαρμογών, όταν αυτό επιτρέπεται. Η εφαρμογή που θέλει να χρησιμοποιήσει κάποιο στοιχεία άλλης εφαρμογής δεν ενσωματώνει τον κώδικα της, αλλά εκκινεί το μέρος της άλλης εφαρμογής που χρειάζεται. Για να γίνει το παραπάνω, το σύστημα πρέπει να είναι ικανό να ξεκινάει μια διεργασία για την εφαρμογή, και όταν προκύψει η ανάγκη χρήσης οποιουδήποτε μέρους της, να αρχικοποιεί τα αντικείμενα που είναι απαραίτητα για το μέρος αυτό. Σε αντίθεση με τις εφαρμογές άλλων συστημάτων, οι εφαρμογές στην πλατφόρμα Android δεν έχουν ένα μοναδικό σημείο εκκίνησης. Αντίθετα υπάρχουν τα απαραίτητα συστατικά στοιχεία, τα οποία το σύστημα μπορεί να εκκινήσει και να εκτελέσει όταν απαιτηθεί. Υπάρχουν τέσσερα είδη συστατικών στοιχείων και αυτά είναι: Δραστηριότητες (Activities), Υπηρεσίες (Services), αποδέκτες συμβάντων (Broadcaster receivers), αποδέκτες περιεχομένου (Content receivers).

- **Δραστηριότητες (Activities):** Η δραστηριότητα είναι ένα από τα βασικά συστατικά στοιχεία της εφαρμογής. Μία δραστηριότητα είναι εφοδιασμένη με ένα γραφικό περιβάλλον, στο οποίο ο χρήστης μπορεί να αλληλεπιδράσει για να εκτελέσει μία ενέργεια, όπως να καλέσει μία επαφή, να τραβήξει μία φωτογραφία, να στείλει ένα email ή να δει ένα χάρτη. Σε κάθε δραστηριότητα δίνετε ένα παράθυρο στο οποίο σχεδιάζεται η διεπαφή χρήστη (User Interface). Το παράθυρο καταλαμβάνει συνήθως ολόκληρη την οθόνη της συσκευής, αλλά μπορεί και να είναι μικρότερο από την οθόνη τοποθετημένο πάνω από άλλα παράθυρα. Μία εφαρμογή αποτελείται τις περισσότερες φορές από πολλές δραστηριότητες, οι οποίες συνδέονται μεταξύ τους. Μία από τις δραστηριότητες έχει το ρόλο της «κύριας» δραστηριότητας, που προβάλλεται στο χρήστη με την έναρξη της εφαρμογής για πρώτη φορά. Κάθε δραστηριότητα μπορεί να εκκινεί μία άλλη δραστηριότητα ώστε να εκτελεστούν διαφορετικές ενέργειες. Όταν ξεκινάει μια νέα δραστηριότητα, η προηγούμενη σταματάει και το σύστημα την διατηρεί σε μία στοίβα. Η νέα δραστηριότητα τοποθετείται στην αρχή της στοίβας και εστιάζεται. Η στοίβα λειτουργεί σύμφωνα

με το LIFO (Last In First Out), έτσι όταν ό χρήστης τελειώσει με τη χρήση μιας δραστηριότητας και πιέσει το Back Button τότε αυτή βγαίνει από τη στοίβα και η προηγουμένη δραστηριότητα συνεχίζει από εκεί που είχε μείνει.

- **Υπηρεσία (Service):** Μια υπηρεσία δεν παρέχει κάποιο γραφικό περιβάλλον αλλά εκτελείται στο περιθώριο για ένα αόριστο χρονικό διάστημα. Ένα πρόγραμμα αναπαραγωγής τραγουδιών αποτελεί ένα καλό παράδειγμα, καθώς ένα service μπορεί να αναπαράγει κάποιο μουσικό κομμάτι ενώ ο χρήστης πραγματοποιεί ταυτόχρονα άλλες ενέργειες. Η εφαρμογή πιθανόν να αποτελείται από παραπάνω από ένα activities που θα επιτρέπουν στο χρήστη να επιλέγει τραγούδια για αναπαραγωγή από μία λίστα. Παρ' όλα αυτά, ο χειρισμός της αναπαραγωγής της μουσικής δε γίνεται από κάποιο activity καθώς οι χρήστες αναμένουν η μουσική να συνεχίσει να παίζει όταν θα εγκαταλείψουν την εφαρμογή και θα αρχίσουν τα ασχολούνται με κάποια άλλη δραστηριότητα. Προκειμένου να καταστεί δυνατό αυτό, το activity της εφαρμογής ορίζει κάποιο service να τρέχει στο περιθώριο. Κάθε υπηρεσία επεκτείνει την κλάση Service. Είναι δυνατή η σύνδεση σε κάποιο ενεργό service και όσο υπάρχει η σύνδεση μπορεί να γίνει επικοινωνία με αυτό διαμέσου μίας διεπαφής. Στο παραπάνω παράδειγμα, η διεπαφή μπορεί να επιτρέπει στο χρήστη να κάνει παύση στη μουσική, να γυρίσει προς τα πίσω κάποιο τραγούδι, να το σταματήσει τελείως ή να ξεκινήσει την αναπαραγωγή του από την αρχή. Όπως τα activities και τα άλλα συστατικά μέρη, τα services εκτελούνται από το κύριο νήμα της εφαρμογής. Έτσι, για να μη διακόψουν την εκτέλεση άλλων συστατικών μερών της διεπαφής συχνά γεννούν ένα άλλο νήμα προκειμένου να εκτελέσουν μία χρονοβόρα διαδικασία.
- **Broadcaster receivers:** Ένα broadcast receiver είναι ένα συστατικό στοιχείο το οποίο αναλαμβάνει τη λήψη διάφορων ειδοποιήσεων και αντιδρά στην εκπομπή τους. Πολλές από τις εκπομπές προέρχονται από τον κώδικα του συστήματος, όπως για παράδειγμα ειδοποιήσεις σχετικές με την αλλαγή της ώρας, τη χαμηλή ενέργεια της μπαταρίας, τη λήψη μίας φωτογραφίας, την αλλαγή της γλώσσας. Οι εφαρμογές μπορούν, επίσης, να στείλουν μία ειδοποίηση, όπως για παράδειγμα να κάνουν γνωστό σε άλλες εφαρμογές ότι κάποια δεδομένα έχουν κατέβει στη συσκευή και είναι έτοιμα προς χρήση. Τα broadcast receivers επεκτείνουν την κλάση BroadcastReceiver. Μία εφαρμογή μπορεί να έχει οποιονδήποτε αριθμό από broadcast receivers για να απαντά στις ειδοποιήσεις που θεωρεί σημαντικές. Τα broadcast receivers δεν έχουν κάποιο γραφικό περιβάλλον. Παρ' όλα αυτά μπορούν να εκκινήσουν ένα activity σαν απάντηση σε κάποια πληροφορία που

έλαβαν ή μπορούν να κάνουν χρήση του Notification Manager για να στείλουν κάποια ειδοποίηση στο χρήστη. Μια ειδοποίηση μπορεί μεταξύ άλλων να είναι η δόνηση της συσκευής, η αναπαραγωγή κάποιου ήχου ή το συνεχές άναμμα και σβήσιμο ενός led.

- **Content Providers:** Ένα content provider κάνει ένα συγκεκριμένο σύνολο δεδομένων μίας εφαρμογής διαθέσιμο σε άλλες εφαρμογές. Τα δεδομένα συνήθως είναι αποθηκευμένα είτε στο σύστημα αρχείων είτε σε μία SQLite βάση δεδομένων. Το content provider επεκτείνει την κλάση ContentProvider και υλοποιεί ένα σύνολο μεθόδων που επιτρέπουν σε άλλες εφαρμογές να ανακτήσουν και να αποθηκεύσουν τύπους δεδομένων που αυτό ελέγχει. Παρ' όλα αυτά, οι εφαρμογές δεν καλούν αυτές τις μεθόδους απευθείας. Χρησιμοποιούν ένα αντικείμενο της κλάσης ContentResolver και καλούν τις δικές του μεθόδους. Το αντικείμενο αυτό επικοινωνεί με το content provider και συνεργάζεται μαζί του για τη διαχείριση της επικοινωνίας που λαμβάνει χώρα.

Το **Intent** είναι ένα αντικείμενο που ο ρόλος του είναι να δημιουργεί μία γέφυρα επικοινωνίας, μεταφέροντας ένα μήνυμα για μία ενέργεια που απαιτήθηκε από ένα άλλο στοιχείο της εφαρμογής. Η πληροφορίες που μεταφέρει ένα αντικείμενο Intent, χρησιμοποιούνται από το λειτουργικό σύστημα Android για να καθορίσει ποιο στοιχείο της εφαρμογής θα ξεκινήσει και πληροφορίες που περιέχουν λεπτομέρειες για την ενέργεια που θα κάνει το στοιχείο. Οι κύριες πληροφορίες που περιέχει ένα αντικείμενο Intent είναι:

- Το όνομα του στοιχείου (component name) που θα εκκινήσει
- Η Ενέργεια (Action) που θα εκτελέσει
- Δεδομένα
- Την κατηγορία (Category) του στοιχείου που διαχειρίζεται το Intent
- Επιπλέον πληροφορίες (Extras) που απαιτούνται για την ολοκλήρωση της απαιτούμενης ενέργειας
- Σημαίες (Flags) που ορίζουν τον τρόπο που θα ξεκινήσει μία δραστηριότητα από το λειτουργικό σύστημα Android

Όπως αναφέρθηκε παραπάνω το αντικείμενο Intent μεταφέρει ένα μήνυμα σε ένα στοιχείο της εφαρμογής με πληροφορίες για την εκκίνηση του. Αν και υπάρχουν πολλοί τρόποι που το Intent διευκολύνει την επικοινωνία μεταξύ στοιχείων της εφαρμογής, υπάρχουν τρεις θεμελιώδεις περιπτώσεις χρήσης που χρησιμοποιούνται:

- **Για την εκκίνηση μίας δραστηριότητας**: Κάθε δραστηριότητα εκπροσωπεί μία οθόνη της εφαρμογής. Μπορεί να ξεκινήσει μία δραστηριότητα, παίρνοντας ένα αντικείμενο Intent στη συνάρτηση startActivity(). Το αντικείμενο Intent περιγράφει τον τρόπο

που θα ξεκινήσει η δραστηριότητα μεταφέροντας τις απαραίτητες πληροφορίες.

- **Για την εκκίνηση μίας υπηρεσίας (Service)**: Μία υπηρεσία είναι ένα στοιχείο που εκτελεί ενέργειες στο παρασκήνιο. Η υπηρεσία μπορεί να ξεκινήσει παίρνοντας ένα Intent στην συνάρτηση startService().
- **Για την παράδοση ενός Broadcast**: Το Broadcast είναι ένα μήνυμα που μπορούν να λάβουν όλες οι εφαρμογές. Το λειτουργικό σύστημα παραδίδει διάφορα είδη από Broadcast για γεγονότα που εκτελεί, όπως η σύνδεση της συσκευής με το φορτιστή ή εκκινήσει του λειτουργικού συστήματος. Η παράδοση ενός Broadcast στις εφαρμογές μπορεί να γίνει παίρνοντας ένα Intent στη συνάρτηση sendBroadCast().

Υπάρχουν δύο τύπο Intent:

- Τα Explicit (άμεσα) Intents, τα οποία καθορίζουν την εκκίνηση ενός στοιχειού από το όνομα του. Ο τύπος αυτός χρησιμοποιείται σε εφαρμογές που έχει δημιουργήσει κάποιος και γνωρίζει τα ονόματα των στοιχείων της εφαρμογής του.
- Τα Implicit (έμμεσα) Intents, τα οποία δεν κάνουν χρήση του ονόματος του στοιχείου που θέλουν να ξεκινήσουν, αντί αυτού δηλώνουν μία ενέργεια (Action) που εκτελείται από στοιχείο μίας άλλης εφαρμογής.

2.7 Κύκλος ζωής των εφαρμογών Android

Τα βασικά στοιχεία των εφαρμογών έχουν έναν κύκλο ζωής. Από την αρχή, όταν το σύστημα Android τα αρχικοποιεί προκειμένου να ανταποκριθούν στα Intent μηνύματα, μέχρι και το τέλος που καταστρέφονται. Στα ενδιάμεσα στάδια μπορεί να είναι ενεργά ή αδρανή ή, στην περίπτωση των δραστηριοτήτων να είναι ορατά στο χρήστη ή όχι. Παρακάτω αναλύεται ο κύκλος ζωής των δραστηριοτήτων (Activities), των υπηρεσιών (Service) και των Broadcast receivers.

2.7.1 Κύκλος ζωής δραστηριότητας

Μία δραστηριότητα έχει τρεις βασικές καταστάσεις:

- Είναι **ενεργή** ή εκτελείται όταν είναι στο προσκήνιο της οθόνης. Αυτή είναι η δραστηριότητα που χειρίζεται εκείνη τη στιγμή τις ενέργειες του χρήστη.
- Είναι **σε παύση** ή έχει χάσει την εστίαση της αλλά είναι ακόμα ορατή στο χρήστη. Αυτό σημαίνει ότι μία άλλη δραστηριότητα εκτελείται πάνω από αυτή που είτε είναι διαφανής είτε δεν καλύπτει όλη την οθόνη και η δραστηριότητα που είναι σε παύση είναι ορατή εν μέρει. Η δραστηριότητα που είναι σε παύση είναι ενερ-

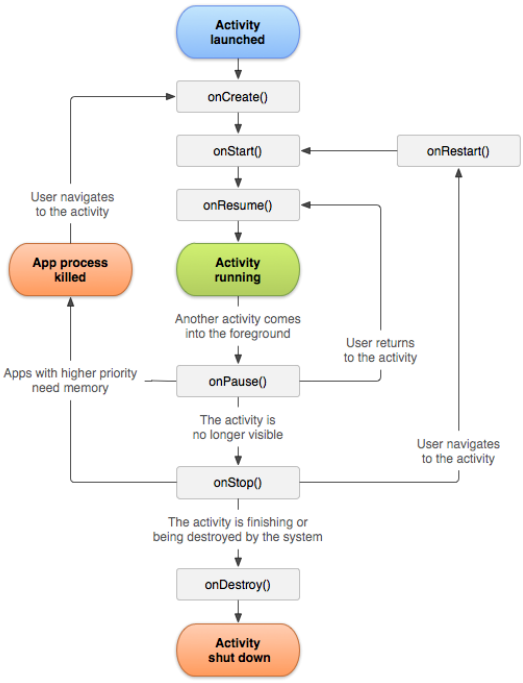
γή, αλλά το σύστημα μπορεί να την τερματίσει αν προκύψει ανάγκη για μνήμη από αλλού.

- Είναι **σταματημένη** αν καλύπτεται εντελώς από κάποια άλλη δραστηριότητα. Συνεχίζει να διατηρεί την κατάσταση της και τις πληροφορίες των μελών της. Επιπροσθέτως, από τη στιγμή που δεν είναι πλέον ορατή στο χρήστη υπάρχει μεγάλη πιθανότητα να τερματιστεί από το σύστημα αν κάποια άλλη εφαρμογή χρειαστεί μνήμη.

Καθώς μία δραστηριότητα μεταβαίνει από κατάσταση σε κατάσταση, ειδοποιείται για τις αλλαγές αυτές μέσω των κλήσεων των συναρτήσεων onCreate(), onStart(), onRestart(), onResume(), onPause(), onStop(), onDestroy(). Οι μέθοδοι αυτές ορίζουν ολόκληρο τον κύκλο ζωής μίας δραστηριότητας. Υπάρχουν τρεις εμφωλευμένοι βρόγχοι που μπορεί να παρατηρήσει κάνεις υλοποιώντας τες:

- Ολόκληρος ο κύκλος ζωής μίας δραστηριότητας βρίσκεται μεταξύ της πρώτης κλήσης onCreate() και της κλήσης της μεθόδου onDestroy(). Μία δραστηριότητα πραγματοποιεί την αρχικοποίηση της κατάσταση της στη μέθοδο onCreate() και απελευθερώνει τους πόρους που κατείχε με την κλήση onDestroy().
- Ο ορατός κύκλος ζωής μιας δραστηριότητας βρίσκεται μεταξύ της κλήσης της μεθόδου onStart() και της κλήσης onStop(). Κατά τη διάρκεια αυτού του χρονικού διαστήματος, ο χρήστης βλέπει τη δραστηριότητα στην οθόνη και αλληλεπιδρά μαζί της. Οι μέθοδοι onStart() και onStop() μπορούν να κληθούν πολλές φορές καθώς η δραστηριότητα γίνεται συχνά ορατή ή το αντίθετο.
- Ο κύκλος ζωής μίας δραστηριότητας στο προσκήνιο βρίσκεται μεταξύ της κλήσης της μεθόδου onResume() και της κλήσης της μεθόδου onPause(). Στη διάρκεια αυτού του χρονικού διαστήματος, η δραστηριότητα βρίσκεται πάνω από άλλες δραστηριότητες στην οθόνη.

Η παρακάτω Εικόνα 15 απεικονίζει αυτούς τους βρόγχους και τα μονοπάτια που μία δραστηριότητα ακολουθεί μεταξύ των καταστάσεων.

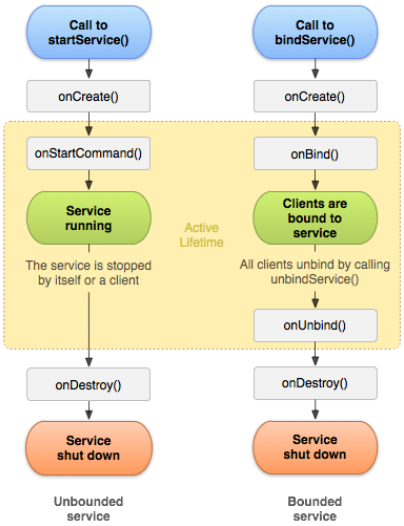


Εικόνα 15. Κύκλος ζωής δραστηριότητας

2.7.2 Κύκλος ζωής υπηρεσίας

Μία υπηρεσία έχει δύο τρόπους λειτουργίας:

- Μπορεί να εκκινήσει και να εκτελείται μέχρι κάποιος να τη σταματήσει ή να σταματήσει μόνη της. Σε αυτήν την περίπτωση ξεκίνα καλώντας τη μέθοδο Context.startService() και σταματά καλώντας την Context.stopService() ή μπορεί να σταματήσει τον εαυτό της καλώντας τις Service.stopSelf() ή stopSetResult(). Μόνο μία κλήση σε κάποια stopService() μέθοδο αρκεί για τον τερματισμό της υπηρεσίας όσες φορές και αν έχει κληθεί η startService().
- Μπορεί να γίνει χρήση της προγραμματιστικά χρησιμοποιώντας μία διαπεφή που την ορίζει. Χρειάζεται η δημιουργία μίας σύνδεσης με το αντικείμενο της υπηρεσίας και μέσω της σύνδεσης αυτής να κληθεί η υπηρεσία. Η δημιουργία της σύνδεσης γίνεται καλώντας την Context.bindService() και κλείνει καλώντας την Context.unbindService(). Στην ίδια υπηρεσία μπορούν να γίνουν πολλαπλές συνδέσεις.



Εικόνα 16. Κύκλος ζωής υπηρεσίας

Όπως και μία δραστηριότητα, έτσι και μία υπηρεσία έχει μεθόδους που ορίζουν τον κύκλο ζωής της και μπορεί να υλοποιήσει ο προγραμματιστής για να παρακολουθεί τις αλλαγές των καταστάσεων. Οι μέθοδοι είναι onCreate(), onStart(Intent intent), onDestroy(). Υλοποιώντας αυτές τις μεθόδους, ο προγραμματιστής έχει τη δυνατότητα να παρακολουθεί δύο εμφωλευμένους βρόγχους του κύκλους ζωής της υπηρεσίας:

- Ολόκληρος ο κύκλος ζωής μίας υπηρεσίας βρίσκεται μεταξύ της πρώτης κλήσης της onCreate() και της κλήσης της μεθόδου onDestroy(). Όπως κα μία δραστηριότητα, μία υπηρεσία αρχικοποιείται με την onCreate() και απελευθερώνει όλους τους πόρους που κατέχει καλώντας την onDestroy().
- Ο ενεργός κύκλος ζωής μίας υπηρεσίας ξεκινά καλώντας την onStart(). Η μέθοδος αυτή χειρίζεται το αντικείμενο Intent που περνάει ως παράμετρο στη startService().

Αν μία υπηρεσία επιτρέπει σε τρίτους να συνδεθούν με αυτή τότε υπάρχουν κάποιες επιπλέον μέθοδοι που πρέπει να υλοποιηθούν:

- IBinder onBind(Intent intent)
- Boolean onUnbind(Intent intent)
- Void onRebind(Intent intent)

Στην Εικόνα 16 παρουσιάζονται διαγραμματικά οι παραπάνω μέθοδοι.

2.7.3 Κύκλος ζωής Broadcaster receiver

Ένα broadcaster receiver έχει μόνο μία μέθοδο, την onReceive(Context curContext, Intent broadcastMsg). Όταν ένα μήνυμα καταφτάσει στο broadcast receiver, το Android καλεί τη μέθοδο onReceive() με όρισμα το Intent αντικείμενο που περιέχει το μήνυμα. Το broadcast receiver θεωρείται ότι είναι ενεργό μόνο όταν εκτελείται αυτή η μέθοδος. Όταν η μέθοδος onReceive() επιστρέψει τότε είναι ανενεργό.

Μία διεργασία με ένα ενεργό broadcast receiver δε μπορεί να τερματιστεί. Αλλά μία διεργασία με μόνο ανενεργά συστατικά μπορεί να τερματιστεί οποιαδήποτε στιγμή από το σύστημα όταν η μνήμη που δεσμεύει ζητηθεί από άλλες διεργασίες. Αυτό δημιουργεί πρόβλημα όταν η απάντηση σε ένα μήνυμα είναι χρονοβόρα και, συνεπώς, πρέπει να γίνει σε ξεχωριστό νήμα, διαφορετικό από το κύριο που χρησιμοποιούν τα άλλα συστατικά της διεπαφής για να εκτελεστούν. Αν όμως η onReceive() γεννήσει ένα νέο νήμα και μετά τερματίσει, τότε όλη η διεργασία, μαζί με το νήμα, κινδυνεύει να θεωρηθεί ανενεργή και άρα να τερματίσει. Η λύση σε αυτό το πρόβλημα είναι η μέθοδος onReceive() να εκκινήσει ένα service για να πραγματοποιήσει την εργασία που χρειάζεται, έτσι ώστε το σύστημα να ξέρει ότι η διεργασία είναι ακόμα ενεργή.

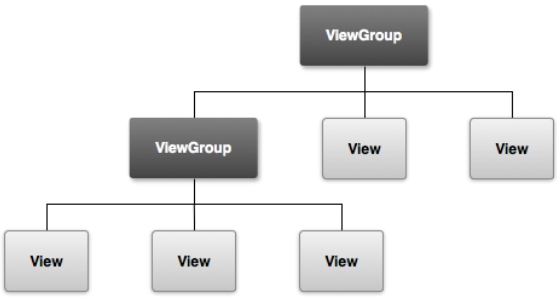
2.8 Το αρχείο AndroidManifest.xml

Κάθε εφαρμογή πρέπει να έχει ένα αρχείο AndroidManifest.xml στο ριζικό κατάλογο της εφαρμογής. Το αρχείο παρουσιάζει αναγκαίες πληροφορίες της εφαρμογής, για το σύστημα Android, πληροφορίες οι όποιες πρέπει να έχει το σύστημα για να εκτελεστεί με επιτυχία ο κώδικας της εφαρμογής. Ανάμεσα στα πολλά πράγματα που κάνει το αρχείο, κάποια είναι τα παρακάτω:

- Ονομάζει το πακέτο java για την εφαρμογή. Το όνομα του πακέτου εξυπηρετεί ως ένα μοναδικό αναγνωριστικό για την εφαρμογή.
- Περιγράφει τα στοιχεία (components) της εφαρμογής. Ονομάζει τις κλάσεις που εφαρμόζουν τα στοιχεία της εφαρμογής και δημοσιεύει τις δυνατότητες τους. Οι δηλώσεις αυτές επιτρέπουν στο σύστημα να γνωρίζει τι είναι τα στοιχεία αυτά, τι κάνουν και ποιες πρέπει να είναι οι συνθήκες για να εκκινήσουν.
- Αποφασίζει ποιες διαδικασίες θα ενσωματώσουν τα στοιχεία της εφαρμογής.
- Αποφασίζει ποια δικαιώματα απαιτούνται για να υπάρξει πρόσβαση σε προστατευμένα μέρη του API.

2.9 Η διεπαφή χρήστη του Android

Η διεπαφή χρήστη είναι η οθόνη που βλέπει ο χρήστης και αλληλεπιδρά με αυτήν σε μία εφαρμογή. Το Android παρέχει ένα σύνολο από έτοιμα εξαρτήματα για τον έλεγχο και την εμφάνιση τους, επιτρέποντας τη δημιουργία μίας ολοκληρωμένης διεπαφής για μία εφαρμογή. Όλα τα στοιχεία της διεπαφή χρήστη δημιουργούνται μέσω των αντικειμένων View και ViewGroup. Το View είναι ένα αντικείμενο που ζωγραφίζει κάτι στην οθόνη και ο χρήστης μπορεί να αλληλεπιδράσει μαζί του. Ενώ τα αντικείμενα ViewGroup έχουν το ρόλο να διατηρούν αντικείμενα View και να ορίζουν τη θέση των αντικειμένων Views στην οθόνη. Η διάταξη της διεπαφής χρήστη ορίζεται από μία ιεραρχία των αντικειμένων αυτών, όπως παρουσιάζεται στην Εικόνα 17. Κάθε αντικείμενο view group είναι ένα αόρατο δοχείο, που εμπεριέχει αντικείμενα view και οργανώνει τη θέση τους στην διάταξη.



Εικόνα 17. Ιεραρχία ViewGroups και Views

Η δημιουργία της διάταξης μπορεί να γίνει με την χρήση κώδικα κατά την εκτέλεση της εφαρμογής ή με τον ορισμό της σε ένα αρχείο XML. Ο δεύτερος τρόπος είναι πιο εύκολος, αφού προσφέρει μία πιο κατανοητή δομή της διάταξης για έναν άνθρωπο.

Οι **διατάξεις (Layouts)** είναι αντικείμενα ViewGroup που ορίζουν τον τρόπο που θα τοποθετηθούν τα αντικείμενα View που περιέχουν. Οι βασικές διατάξεις είναι:

- Η διάταξη **LinearLayout** οργανώνει τα αντικείμενα View που περιέχει σε μία οριζόντια ή κάθετη σειρά στην οθόνη.
- Η διάταξη **RelativeLayout** επιτρέπει τον προσδιορισμό της θέσης των αντικείμενων σε σχέση με άλλα αντικείμενα που περιέχει (π.χ. το αντικείμενο A να τοποθετηθεί δεξιά του αντικειμένου B).
- Η διάταξη **WebView** παρουσιάζει το περιεχόμενο μίας ιστοσελίδας.

Οι διατάξεις έχουν έναν σύνολο από ιδιότητες, που καθορίζουν τις ιδιότητες των αντικειμένων που περιέχουν όπως το μέγεθος τους, τη θέση τους και την απόσταση μεταξύ τους.

Τα **στοιχεία ελέγχου (controls)** είναι αντικείμενα View που παρουσιάζουν μία γραφική μορφή. Μπορεί να παριστάνουν ένα κουμπί, ένα κείμενο, μία εικόνα. Μερικά από τα βασικά εξαρτήματα είναι:

Σχόλιο [s1]:
ΕΙΝΑΙ ΣΩΣΤΟ ΑΥΤΟ ΤΟ ΑΓΓΛΙΚΟ;

Ετσίτ το είχε

- Κουμπιά (**Buttons**): Μπορούν να πατηθούν από το χρήστη για να γίνει μία ενέργεια.
- Πλαίσιο Κείμενου (**TextView**): Παρουσιάζει ένα κείμενο.
- Επεξεργασία Κείμενου (**EditText**): Χρησιμοποιείται για τυπική επεξεργασία κειμένου από το χρήστη.
- Εικόνα (**ImageView**): Παρουσιάζει μία εικόνα.

Τα στοιχεία ελέγχου διαθέτουν ιδιότητες, που ορίζουν τα χαρακτηριστικά από τα εξαρτήματα. Τέτοιες Ιδιότητες είναι το μέγεθος, γραμματοσειρά χρώμα των χαρακτήρων.

2.10 Υπηρεσίες διαδικτύου (Web Services)

Οι υπηρεσίες διαδικτύου [9] είναι προγραμματιστικές διεπαφές εφαρμογών (Application Programming Interfaces ή APIs) ή δικτυακά APIs στα όποια η πρόσβαση γίνεται μέσω HTTP και εκτελούνται σε ένα απομακρυσμένο υπολογιστικό σύστημα που φιλοξενεί τις υπηρεσίες που προσφέρουν.

Διαφορετικά προγραμμάτα έχουν σύνηθως την ανάγκη να ανταλάσουν πληροφοριές μεταξύ τους. Η υπηρεσία διαδικτύου είναι η μέθοδος που χρησιμοποιείται, για να επιτευχθεί η επικοινώνια μεταξύ δύο διαφορετικών προγραμμάτων. Το προγραμμά που ζητάει τις πληροφορίες, ονομάζεται service requester και το πρόγραμμα που επεξαργάζεται το αίτημα και παρέχει τα δεδομένα ονομάζεται service provider.

Επειδή τα προγραμμάτα υλοποιούνται με διαφορετίκες γλώσσες προγραμματισμού, υπαρχεί η αναγκή να γίνει χρήση μίας μέθοδού, που η ανταλλαγή δεδομένων δεν εξαρταταί από μία συγκεκριμένη γλώσσα προγραμματισμού. Οι υπηρεσίες διαδικτύου χρησιμοποιούν αρχεία XML ή το πρότυπο JSON για την ανταλλαγή δεδομένων.

Κανονές για την επικοινώνια μέταξύ δύο διαφορετικών λογισμικών πρέπεί να όριστουν όπως:

- Ο τρόπος που ζητάει δεδομένα ένα πρόγραμμα από ενά άλλο προγραμμα.
- Οι παραμέτροι που χρειάζονται για την αίτηση δεδομένων.
- Τη δομή που έχουν τα δεδομένα που θα παραχθούν.

- Και τέλος τα μηνυμάτα λάθους που για καποιά αιτία αποτύχει η επικοινώνια.

2.11 JSON

Το JSON ή JavaScript Object Notation [8] είναι ένα ανοιχτό πρότυπο ανταλλαγής δεδομένων. Είναι εύκολο στην ανάγνωση και γραφή από ανθρώπους. Επίσης είναι εύκολο για τις ηλεκτρονικές συσκευές να το παράγουν και να το αναλύουν. Είναι βασισμένο σε ένα υποσύνολο της γλώσσας προγραμματισμού JavaScript (Standard ECMA-262 Έκδοση 3η- Δεκέμβριος 1999). Το JSON είναι ένα πρότυπο κειμένου το όποιο είναι τελείως ανεξάρτητο από γλώσσες προγραμματισμού αλλά χρησιμοποιεί πρακτικές οι όποιες είναι γνωστές στους προγραμματιστές της οικογένειας προγραμματισμού C, C++, C#, Java, JavaScript, Perl, Python και πολλών άλλων. Το JSON είναι υλοποιημένο σε δύο δομές:

- Μια συλλογή από ζευγάρια ονομάτων-τιμών. Σε διάφορες γλώσσες προγραμματισμού, αυτό γίνεται αντιληπτό ως ένα object (αντικείμενο), καταχώρηση, δομή πίνακα hash, λίστα κλειδιών ή associative πίνακα.
- Μια ταξινομημένη λίστα τιμών. Στις περισσότερες γλώσσες προγραμματισμού, αυτό γίνεται αντιληπτό ως ένα πίνακας (array), διάνυσμα, λίστα ή ακολουθία.

Αυτά είναι τα ονομαζόμενα universal data structures (καθολικές δομές δεδομένων). Ουσιαστικά όλες οι μοντέρνες γλώσσες προγραμματισμού τα υποστηρίζουν με τον έναν ή τον άλλον τρόπο. Λογικό είναι πως ένα πρότυπο δεδομένων, το οποίο είναι εύκολα μεταβαλλόμενο με γλώσσες προγραμματισμού, οι οποίες επίσης είναι βασισμένες σε αυτές τις δομές που αναφερθήκαν παραπάνω.

Τα αντικείμενα JSON παίρνουν την εξής μορφή: Ένα αντικείμενο JSON (object) είναι ένα άτακτο σύνολο από ζευγάρια ονόματων-τιμών. Ένα αντικείμενο (object) JSON ξεκινάει με «{« (αριστερό άγκιστρο) και τελειώνει με «}» (δεξιό άγκιστρο). Κάθε όνομα ακολουθείται από «:» (άνω-κάτω τελεία) και τα ζευγάρια ονόματος/τιμής χωρίζονται από «,» (κόμμα).

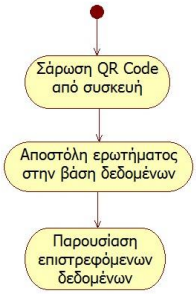
Σχόλιο [s2]: ΔΕ ΒΓΑΖΕΙ Ν ΤΕΛΕΙΩΝΕΙ Ε

Σχόλιο [s3]: Ποια παίρνουν νοητό...

Κεφάλαιο III
Διασύνδεση QR Codes και βάσης δεδομένων σε φορητές συσκευές

3.1 Εισαγωγή

Σκοπός αυτής της εργασίας ήταν η δημιουργία εφαρμογής για έξυπνες κινητές συσκευές με λειτουργικό σύστημα Android, η οποία με τη βοήθεια της κάμερας της κινητής συσκευής θα σαρώνει και θα αναγνωρίζει ένα QR Code και έπειτα θα αποστέλλει το αποτέλεσμα της αποκωδικοποίησης ως ερώτημα σε μια απομακρυσμένη βάση δεδομένων και θα παρουσιάζει το αποτέλεσμα του ερωτήματος. Για το σκοπό αυτό σχεδιάστηκε και υλοποιήθηκε εφαρμογή που η βασική της λειτουργία παρουσιάζεται συνοπτικά και γραφικά στην Εικόνα 18. Για την επίδειξη της αποτελεσματικότητας και χρησιμότητας της ανάπτυξης του εν λόγω συστήματος, το σύστημα συνδέθηκε με πραγματική βάση δεδομένων που υλοποιήθηκε βάσει υπαρκτών στοιχείων τεκμηρίωσης πολιτιστικής κληρονομιάς που περιλαμβάνουν σημαντικά ιστορικά κτήρια της παλιάς πόλης της Ξάνθης, για τα οποία έχει ήδη γίνει σήμανση με QR Codes.



Εικόνα 18. Βασική δομή συστήματος

3.2 Προδιαγραφές της εφαρμογής "*Old Town of Xanthi Guide*"

Στην παλιά πόλη της Ξάνθης υπάρχουν συγκεκριμένα κτίρια ιστορικού και αρχιτεκτονικού ενδιαφέροντος. Σε επιλεγμένα από τα κτίρια αυτά, υπάρχει τυπωμένο QR Code που περιέχει σύνδεσμο, ο οποίος παραπέμπει σε συγκεκριμένη ιστοσελίδα στο διαδίκτυο. Η ιστοσελίδα περιέχει βασικές πληροφορίες πολιτιστικής τεκμηρίωσης και περιλαμβάνει πληροφορίες για την αρχιτεκτονική, την ιστορία του κτιρίου και μία ενδεικτική φωτογραφία του. Έτσι οι επισκέπτες της παλιάς πόλης της Ξάνθης με έξυπνη κινητή συσκευή μπορούν να φωτογραφήσουν το αντίστοιχο QR Code πάνω στο κτίριο που τους ενδιαφέρει και να λάβουν άμεσα αυτή την πληροφόρηση.

Ως στόχος της παρούσας εργασίας τέθηκε η ανάπτυξη εφαρμογής για κινητές συσκευές με λειτουργικό σύστημα Android, που θα μπορεί να προσφέρει στο χρήστη λειτουργικότητα πολιτιστικής ενημέρωσης για τα επιλεγμένα κτίρια της παλιάς πόλης της Ξάνθης, μέσα από ένα ενοποιημένο περιβάλλον προβολής με διασύνδεση σε διαδικτυακή βάση δεδομένων. Στις βασικές προδιαγραφές της εφαρμογής τέθηκαν τα εξής:

- **Σάρωση QR Code** κάνοντας χρήση της πίσω κάμερας της κινητής συσκευής.
- **Σύνδεση σε βάση δεδομένων** που περιέχει τις πληροφορίες πολιτιστικής τεκμηρίωσης για τα κτίρια της παλιάς πόλης της Ξάνθης, αντιστοιχώντας τα QR Code που είναι τυπωμένα έξω από κάθε κτίριο με τις εγγραφές στη βάση δεδομένων που περιέχουν τις πληροφορίες κάθε κτιρίου.
- **Αποθήκευση του QR Code** που σαρώθηκε για την εκ των υστέρων σύνδεση με τη βάση δεδομένων σε περίπτωση που η κινητή συσκευή δεν είναι συνδεδεμένη στο διαδίκτυο.

Παράλληλα, η εφαρμογή πρέπει να έχει εκτός από την οθόνη σάρωσης του QR Code και την οθόνη αποτελεσμάτων και άλλα χαρακτηριστικά γνωρίσματα όπως:

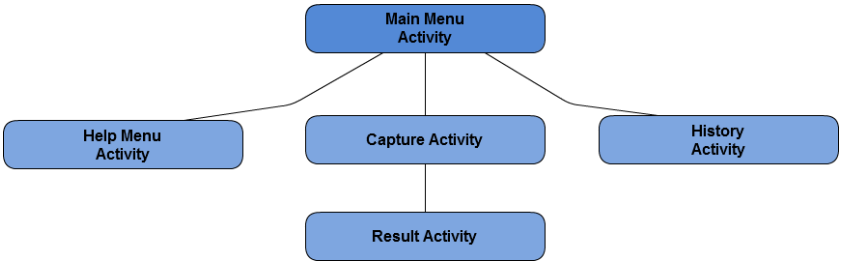
- Μια αρχική οθόνη
- Μια οθόνη που παρουσιάζει το ιστορικό των σαρώσεων
- Μια οθόνη βοήθειας, με οδηγίες χρήσεις

Μέσω τις αρχικής οθόνης ο χρήστης μπορεί να μεταβεί στις υπόλοιπες δραστηριότητες της εφαρμογές. Αναλύοντας τις απαιτήσεις της εφαρμογής, διαπιστώνουμε ότι απαιτούνται πέντε αρχικές οθόνες. Είναι οι ακόλουθες:

- Μια κύρια οθόνη (Main menu Screen)
- Μια οθόνη σάρωσης του QR Code (Capture Screen)

- Μια οθόνη βοηθείας (Help Screen)
- Μια οθόνη αποτελεσμάτων (Result Screen)
- Μια οθόνη που δείχνει το ιστορικό των σαρώσεων

Αυτές οι οθόνες αποτελούν τη διεπαφή χρήστη για την εφαρμογή. Κάθε οθόνη της εφαρμογής έχει τη δική της κλάση Δραστηριότητας (Activity class). Το σχήμα στην Εικόνα 19 παρουσιάζει τις πέντε δραστηριότητες που απαιτούνται, μία για κάθε οθόνη.



Εικόνα 19. Απλοποιημένο διάγραμμα των δραστηριοτήτων ροής

Η **αρχική οθόνη** (Main menu), χρησιμεύει ως το αρχικό σημείο εισόδου της εφαρμογής. Η λειτουργία της πρέπει να ενσωματωθεί μέσα στην κλάση MainMenuActivity. Οι λειτουργίες αυτής της οθόνης είναι οι εξής:

- Να εμφανίζει το όνομα της εφαρμογής
- Να έχει ένα ενδιαφέρον γραφικό (Background) σχετικό με την εφαρμογή
- Να παρέχει λειτουργίες για μετάβαση στις υπόλοιπες οθόνες της εφαρμογής (Βοήθεια, Σάρωση, About)

Στην **οθόνη σάρωσης** πρέπει να υπάρχει μια περιοχή σάρωσης, όπου ο χρήστης θα τοποθετεί το QR Code με εύκολο τρόπο ώστε να επιτευχθεί ορθή σάρωση του QR Code. Η Οθόνη αυτή ενσωματώνεται στην κλάση CaptureActivity της εφαρμογής.

Η **οθόνη βοήθειας** εμφανίζει στο χρήστη κάποιες χρήσιμες πληροφορίες για την εφαρμογή. Η λειτουργία της πρέπει να ενσωματωθεί μέσα στην κλάση HelpActivity. Αυτή η δραστηριότητα πρέπει να εκτελεί τα εξής:

- Να εμφανίζει το κείμενο βοήθειας στο χρήστη
- Να επιτρέπει την κύλιση του κειμένου στο χρήστη αν αυτό ξεφεύγει από τα φυσικά όρια της οθόνης

Η **οθόνη αποτελεσμάτων** επιτρέπει στο χρήστη να δει τα αποτελέσματα από τη σάρωση του QR code. Η λειτουργία της πρέπει να ενσωματωθεί στην κλάση ResultActivity της εφαρμογής και τα χαρακτηριστικά της είναι τα εξής:

- Να εμφανίζει το όνομα του κτιρίου
- Να εμφανίζει μία εικόνα του κτιρίου
- Να εμφανίζει ένα κείμενο με πληροφορίες για το κτίριο
- Να επιτρέπει την κύλιση της οθόνης
- Να εμφανίζει μήνυμα σε περίπτωση αστοχίας

Στην **οθόνη ιστορικού** εμφανίζεται το ιστορικό επιτυχών σαρώσεων κτιρίων. Για κάθε κτίριο και μετά από τη σάρωση του QR Code του αποθηκεύονται μερικά από τα στοιχεία του στη μνήμη της συσκευής. Η λειτουργικότητα ενσωματώνεται στην κλάση HistoryAcitvity της εφαρμογής και τα χαρακτηριστικά της είναι τα εξής:

- Να εμφανίζει το όνομα του κτιρίου και το αντίστοιχο QR Code
- Να εμφανίζει τα δεδομένα σε μορφή λίστας
- Να επιτρέπει την κύλιση της οθόνης

Η εφαρμογή χρειάζεται να διαθέτει ένα σαρωτή QR Code, που με την βοήθεια της πίσω κάμερας του κινητού τηλεφώνου με λειτουργικό σύστημα Android, να πραγματοποιεί σάρωση του επιθυμητού QR Code. Επίσης η εφαρμογή πρέπει να παρέχει τρόπο δικτυακής σύνδεσης σε βάση δεδομένων και επιστροφή των δεδομένων στην συσκευή Android.

Αναλυτικότερα, η εφαρμογή πρέπει να έχει την δυνατότητα να ανοίγει την πίσω κάμερα της συσκευής και όταν τοποθετείται μπροστά από ένα QR Code, να μπορεί να το αναγνωρίζει και να το αποκωδικοποιήσει.

Τα δεδομένα για τα κτίρια μέχρι πριν από την υλοποίηση της εφαρμογής ήταν προσβάσιμα μέσω συγκεκριμένων στατικών ιστοσελίδων στο διαδίκτυο. Το πλήθος των κτιρίων είναι 13 και υπάρχει μία αντίστοιχη ιστοσελίδα για κάθε κτίριο. Οι διευθύνσεις των ιστοσελίδων έχουν την μορφή: http://67100.gr/qr/page.php?which=1&lang=en, όπου το πεδίο *which* παίρνει τιμές από το 1 έως και το 13 αντιστοιχώντας σε κάθε κτίριο ξεχωριστά, ενώ η μεταβλητή *lang* αντιστοιχεί στην επιλογή γλώσσας παρουσίασης. Το περιεχόμενο κάθε ιστοσελίδας αποτελείται από το τίτλο του κτιρίου, μία εικόνα του και ένα κείμενο με πληροφορίες.

Η εφαρμογή πρέπει να διαθέτει τρόπο διασύνδεσης με εξωτερική βάση δεδομένων. Μέτα τη σάρωση ενός QR Code το αποτέλεσμα πρέπει να σταλεί ως ερώτημα SQL και να επιστρέψει τα ανάλογα αποτελέσματα. Για να επιτευχθεί η σύνδεση χρειάζεται η συσκευή να είναι συνδεδεμένη στο διαδίκτυο, είτε μέσω ενός ασύρματου δικτύου (Wi-Fi) είτε μέσω δικτύου τηλεφωνίας που επιτρέπει τη μεταφορά δεδομένων. Υπάρχουν δύο τρόποι για την επίτευξη σύνδεσης σε εξωτερική βάση δεδομένων. Ο πρώτος τρόπος είναι η απευθείας σύνδεση, η οποία γίνεται συνήθως μέσω JDBC (Java Database Connectivity) και εφόσον έχει εγκατασταθεί ο αντίστοιχος JDBC Driver στο database server. Η συγκεκριμένη λύση όχι μόνο δεν είναι δυνατή για όλες τις βάσεις δεδομένων, αλλά θεωρείται και

κατά κάποιο τρόπο λανθασμένη. Οι βασικοί λόγοι που η συγκεκριμένη τεχνική θεωρείται λανθασμένη είναι η χαμηλή ασφάλεια και η χαμηλή απόδοσή της. Επίσης αποτελεί λανθασμένη προγραμματιστική πρακτική, καθώς δεν καταφέρνει να διαχωρίσει το πρόβλημα επιτυχώς σε client-server. Ο δεύτερος τρόπος είναι μέσω υπηρεσίας διαδικτύου (web service), ο όποιος ενδείκνυται καθώς είναι προσβάσιμος ανεξάρτητα από εφαρμογή και συσκευή. Για χρόνια το πρωτόκολλο SOAP (Simple Object Access Protocol) ήταν και εξακολουθεί να είναι το πιο δημοφιλές για την ανάπτυξη web services. Στο Android όμως δεν υπάρχει υποστήριξη σε αυτό, αφού οι μεγάλες εταιρείες όπως η Google έχουν στραφεί προς το ταχύτατα αναδυόμενο REST (Representational State Transfer). Το REST βασίζεται στα βασικά συστατικά του internet (HTTP, URL) και είναι ιδιαίτερα ευέλικτο, καθώς υποστηρίζει διάφορους τύπους δεδομένων (MIME) όπως XML, JSON αλλά και απλό κείμενο (plain text).

Ένας πετυχημένος τρόπος υλοποίησης της εφαρμογής περιλαμβάνει την αποστολή μέσω του πρωτοκόλλου HTTP με μέθοδο Get δεδομένα στον απομακρυσμένο Server, αυτός δημιουργεί ερώτημα στη βάση δεδομένων, και επιστρέφει το αποτέλεσμα ως JSON Object στην εφαρμογή. Σημειώνεται ότι βασική προϋπόθεση για την online (πραγματικού χρόνου) λειτουργία της εφαρμογής απαιτείται σύνδεση στο διαδίκτυο. Για την αντιμετώπιση του προβλήματος της αδυναμίας σύνδεσης με το διαδίκτυο, η εφαρμογή χρειάζεται κάποιο τρόπο αποθήκευσης δεδομένων για τη λειτουργία του ιστορικού. Οι εφαρμογές για Android, μπορούν να αποθηκεύσουν δεδομένα με τρεις τρόπους: μέσω τοπικής βάσης δεδομένων (SQLite), σε τοπικά αρχεία ή στις ρυθμίσεις, όλα στην ίδια συσκευή. Στην περίπτωση της παρούσας εφαρμογής θα χρησιμοποιηθεί η μέθοδος της βάσης δεδομένων.

3.3 Η υλοποίηση της εφαρμογής "*Old Town of Xanthi Guide*"

Όπως προαναφέρθηκε σε προηγούμενες παραγράφους τα πραγματικά δεδομένα για τα κτίρια υπάρχουν σε ιστοσελίδες στο διαδίκτυο και κάθε ιστοσελίδα περιέχει τον τίτλο του κτιρίου, μία φωτογραφία του και την περιγραφή του. Σύμφωνα με τα παραπάνω στοιχεία, δημιουργήθηκε η βάση δεδομένων με την ονομασία **xanthibuildings**. Η βάση δεδομένων αποτελείται από ένα πίνακα με ονομασία buildings που διαθέτει τα εξής πεδία:

- Ένα πεδίο τύπου χαρακτήρων που είναι μοναδικό αναγνωριστικό για κάθε γραμμή, με ονομασία qrcode
- Ένα πεδίο με την ονομασία placename, τύπου χαρακτήρων που περιέχει την ονομασία του κτιρίου
- Ένα πεδίο με την ονομασία description, τύπου κειμένου που περιέχει τις λεπτομέρειες του κτιρίου

- Και ένα πεδίο με την ονομασία photolink, τύπου χαρακτήρων που περιέχει url μιας φωτογραφίας

Επειδή σε κάθε κτίριο υπάρχει τοποθετημένο ένα QR Code, το οποίο αντιστοιχεί στην ιστοσελίδα του κτιρίου, η οποία είναι μοναδική για κάθε κτίριο, στο πεδίο qrcode αποθηκεύεται η διεύθυνσης της ιστοσελίδας που ανήκει στο κτίριο. Έτσι επιτρέπεται στην εφαρμογή να αντιστοιχεί το τυπωμένο QR Code που είναι στο κτίριο με τις πληροφορίες του κτιρίου που είναι στη βάση δεδομένων. Από κάθε ιστοσελίδα που διαθέτει πληροφορίες για τα κτίρια ελήφθησαν τα δεδομένα που περιέχει η βάση δεδομένων. Στον παρακάτω πίνακα (Πίνακας ΙΙ) παρουσιάζεται η δομή του πίνακα buildings.

Πίνακας ΙΙ. Δομή του πίνακα buildings

Όνομα πεδίου	Τύπος	μέγεθος
qrname (Primary key)	varchar	100
placename	varchar	100
Description	long Text	-
photolink	varchar	100

Η σύνδεση της εφαρμογής με τη βάση δεδομένων γίνεται μέσω υπηρεσίας διαδικτύου (web service) επιστρέφοντας τα δεδομένα σε μορφή JSON. Η υπηρεσία διαδικτύου υλοποιείται σύμφωνα με το πρωτόκολλο REST. Η εφαρμογή στέλνει ένα αίτημα HTTP με μέθοδο GET, το οποίο περιέχει το αποτέλεσμα της σάρωσης στην υπηρεσία διαδικτύου και αυτή αναλαμβάνει να κάνει το ερώτημα στη βάση δεδομένων και να επιστρέψει το αποτέλεσμα σε μορφή JSON. Η υπηρεσία διαδικτύου αποτελείται από αρχεία σε γλώσσα δυναμικού διαδικτυακού προγραμματισμού PHP και συγκεκριμένα από τα αρχεία: db_config.php, db_connect.php και getData.php.

Το αρχείο **db_config.php** περιέχει τα στοιχεία για τη σύνδεση στη βάση δεδομένων, που αυτά είναι:

- Server: το όνομα του server που βρίσκεται η βάση δεδομένων
- Database: το όνομα της βάσης δεδομένων
- User: το όνομα του χρήστη που έχει δικαίωμα πρόσβασης στη βάση δεδομένων
- Password: ο κωδικός χρήστη για πρόσβαση στη βάση δεδομένων

Το αρχείο **db_connect.php** είναι υπεύθυνο για τη δημιουργία σύνδεσης με τη βάση δεδομένων, ενώ στο αρχείο **getData.php** γίνεται το HTTP αίτημα με μέθοδο GET από την εφαρμογή στην υπηρεσία διαδικτύου. Το HTTP αίτημα γίνεται στο αρχείο getData.php και έχει τη μορφή *http://serverurl/getData.php?ονομά_παραμέτρου=τιμή_παραμέτρου*. Όπου

serverurl η διεύθυνση του server που βρίσκεται το αρχείο. Το αρχείο getData.php δημιουργεί τη σύνδεση με τη βάση δεδομένων και δημιουργεί ερώτημα SQL προς αυτήν. Το ερώτημα που δημιουργεί είναι της μορφής: *"SELECT * FROM places where qrcode='$qrcoderesult'"*, όπου το πεδίο $qrcoderesult παίρνει την τιμή της μεταβλητής που περιέχει το HTTP αίτημα. Το αρχείο getData.php παράγει ένα αντικείμενο JSON που επιστρέφει στην εφαρμογή. Η παραγωγή του εξαρτάται από το αποτέλεσμα του ερωτήματος στη βάση δεδομένων. Υπάρχουν τρεις περιπτώσεις στη δημιουργία του JSON αντικείμενου:

- Περίπτωση κατά την οποία το HTTP αίτημα δεν έχει παραμέτρους: στην περίπτωση αυτή δε γίνεται ούτε ερώτημα στη βάση δεδομένων αφού δεν υπάρχει παράμετρος. Το αντικείμενο JSON έχει τη μορφή:
  {"Success:"0,"message":"Required fields is missing"}
- Περίπτωση το ερώτημα να επιστρέψει στήλες: στην περίπτωση αυτή έχει γίνει το ερώτημα στη βάση δεδομένων και έχουν επιστραφεί τα δεδομένα που αντιστοιχούν στο QR Code. Το αντικείμενο JSON έχει την μορφή:
  {"Success":1,"places":["qrcode":" πεδίο qrcode","name":"όνομα κτιρίου","description":"Περιγραφή κτιρίου","photourl":"το url της εικονας"]}
- Περίπτωση κατά την οποία το ερώτημα δεν επιστρέφει δεδομένα: σε αυτή την περίπτωση έχει γίνει ερώτημα στη βάση δεδομένων αλλά δεν επιστρέφει δεδομένα που σημαίνει ότι δεν υπάρχει αντιστοιχία του QR Code με κτίριο στη βάση δεδομένων. Το αντικείμενο JSON που δημιουργείται έχει τη μορφή:
  {"Success:"0,"message":"Place not found"}

Η εφαρμογή διαχειρίζεται το αντικείμενο JSON που της επιστρέφεται από το HTTP αίτημα και αναλόγως την περίπτωση παρουσιάζει το αποτέλεσμα στο χρήστη. Οι περιπτώσεις αυτές θα αναλυθούν σε επόμενες παραγράφους.

Η εφαρμογή διαθέτει επίσης μία **βάση δεδομένων στην συσκευή**, που πρωταρχικός της σκοπός είναι η αποθήκευση πληροφοριών των κτιρίων σε αύτη, ενώ εξασφαλίζει και την αποθήκευση κάποιου QR Code όταν δεν υπάρχει σύνδεση στο διαδίκτυο την στιγμή που γίνεται σάρωση, ώστε ο χρήστης να μπορέσει να έχει πρόσβαση σε αυτό μέσω του ιστορικού της εφαρμογής όταν διατίθεται πρόσβαση στο διαδίκτυο. Για την υλοποιήση της δημιουργήθηκε μία βάση δεδομένων με όνομα **DBXanthi** βασισμένη στην SQLite που χρησιμοποιεί η πλατφόρμα Android για την αποθηκεύση δεδομένων σε εφαρμογές. Αποτελείται από ένα πίνακα με ονομασία buildings και διαθέτει τα εξής πεδία (βλπ. Πίνακας III) :

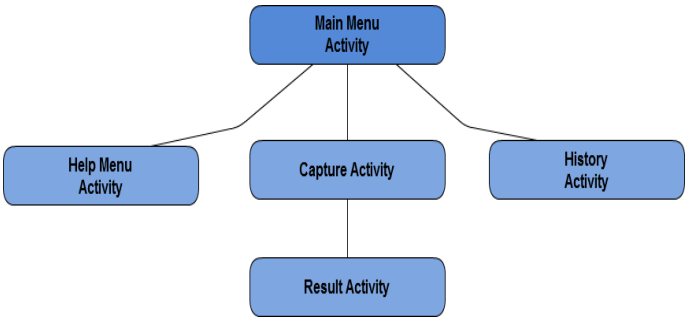
Πίνακας III Βάση δεδομένων συσκευής DBXanthi

Όνομα πεδίου	**Τύπος**	**μέγεθος**
qrname (Primary key)	TEXT	100
placename	TEXT	100
Description	TEXT	-
photolink	TEXT	100

Για τη λειτουργία της εφαρμογής υλοποιήθηκαν οι εξής δραστηριότητες (Activities):

- MainActivity
- CaptureActivity
- HistoryActivity
- HelpActivity
- ResultActivity

Κάθε δραστηριότητα διαθέτει τη δική της διεπαφή χρήστη (User Interface). Στη συνέχεια αναλύεται η σχεδίαση της κάθε δραστηριότητας και της διεπαφής, οι σχέσεις μεταξύ τους και οι λειτουργίες που προσφέρουν στο χρήστη. Στην Εικόνα 20 παρουσιάζεται η ιεραρχία των δραστηριοτήτων.



Εικόνα 20. Ιεραρχία Δραστηριοτήτων

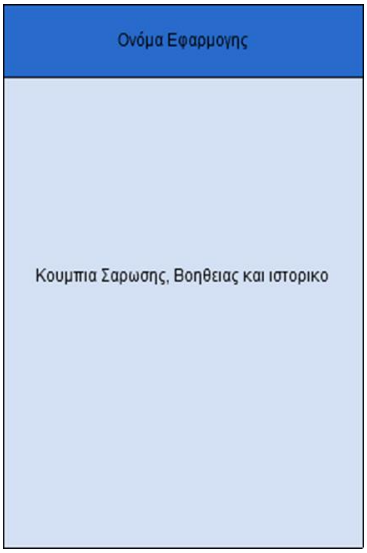
Η δραστηριότητα **MainActivity** υλοποιεί την αρχική οθόνη της εφαρμογής. Η αρχική οθόνη, η εναρκτήρια οθόνη της εφαρμογής, αποτελείται από μια εικόνα παρασκηνίου και τρία πλήκτρα που το καθένα εκτελεί καθεμιά από τις βασικές λειτουργίες της εφαρμογής. Στην **Σφάλμα! Το αρχείο προέλευσης της αναφοράς δεν βρέθηκε.** φαίνεται το βασικό προσχέδιο/σκαρίφημα της αρχικής οθόνης.

Μέσω των πλήκτρων που περιέχει η αρχική οθόνη, ο χρήστης μπορεί να μεταβαίνει στις ορισμένες από την λειτουργία κάθε κουμπιού δραστηριότητες της εφαρμογής. Τα πλήκτρα αυτά είναι:

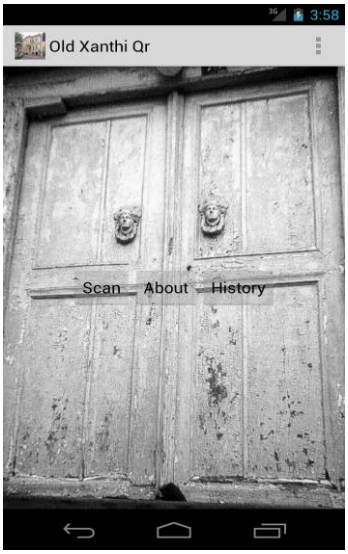
- **Scan Button (**πλήκτρο σάρωσης): Στο πλήκτρο σάρωσης υλοποιείται η μετάβαση στη δραστηριότητα CaptureAcitivity.

- **History Button** (πλήκτρο ιστορικού): Με την πίεση στο πλήκτρο ιστορικού ο χρήστης μεταφέρεται στη δραστηριότητα HistoryActivity. Πριν γίνει η μετάβαση, γίνεται ένας έλεγχος αν υπάρχουν διαθέσιμα δεδομένα στο ιστορικό της εφαρμογής, τα οποία είναι σε εσωτερική βάση δεδομένων στη συσκευή. Στην περίπτωση που δεν είναι διαθέσιμα, εμφανίζεται ένα σύντομο μήνυμα πληροφορώντας ότι δεν υπάρχει ιστορικό (History Records not exist) και η μετάβαση στη δραστηριότητα HistoryActivity δεν πραγματοποιείται.
- **Help Button** (πλήκτρο βοήθειας): Στο πλήκτρο βοηθείας υλοποιείται η μετάβαση στη δραστηριότητα HelpActivity.

Η υλοποίηση της οθόνης της MainActivity δραστηριότητας γίνεται στο αρχείο activity_main.xml. Ορίζεται η διάταξη LinearLayout με οριζόντιο προσανατολισμό και στοίχιση στο κέντρο της οθόνης για τα Views που περιέχει. Επίσης ορίζεται η εικόνα για φόντο στην οθόνη (αρχείο mybackground.png). Μέσα στη διάταξη τοποθετούνται τα πλήκτρα. Σε κάθε πλήκτρο ορίζεται το όνομα της ιδιότητας onClick. Το όνομα της ιδιότητας onClick αντιστοιχεί σε μία μέθοδο της δραστηριότητας MainActivity που σε αυτήν υλοποιείται η λειτουργία κάθε πλήκτρου. Στην Εικόνα 22 παρουσιάζεται η τελική μορφή της αρχικής οθόνης.



Εικόνα 21 Προσχέδιο της αρχικής οθόνης



Εικόνα 22 Η οθόνη της δραστηριότητας MainActivity

Στη δραστηριότητα **CaptureActivity** υλοποιείται μία από τις βασικές λειτουργίες της εφαρμογής, η οποία είναι η λήψη QR Code με την κάμερα της κινητής συσκευής και η αποκωδικοποίηση του. Για αυτό το σκοπό

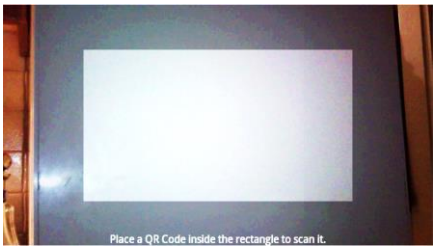
χρησιμοποιήθηκε κώδικας από το έργο ZXing (Εικόνα 23) ο οποίος ενσωματώθηκε στην εφαρμογή. Ο πήγαιος κώδικας βρίσκεται στον ιστότοπο https://github.com/zxing/zxing. Ο κώδικας που προστέθηκε αφορά τη βιβλιοθήκη core του ZXing, η οποία υλοποιεί την αποκωδικοποίηση διαφορών ειδών ραβδοκωδίκων, ανάμεσα στους οποίους είναι και το QR Code και κομμάτια από την εφαρμόγη για το Android που διαχείριζεται την κάμερα της συσκευής ωστέ να σάρωνει ένα QR Code και να το αποκωδικοποιεί.

Εικόνα 23 Λογότυπο του ZXing project

Η μετάβαση στη δραστηριότητα CaptureActivity γίνεται από την αρχική δραστηριότητα MainActivity με την πίεση του πλήκτρου σάρωσης. Με την εκκίνηση της CaptureActivity ενεργοποιείται η πίσω κάμερα της κινητής συσκευής και στην οθόνη παρουσιάζεται το περιεχόμενο που βλέπει η κάμερα. Ο χρήστης πρέπει να στοχεύσει ένα QR Code με τέτοιο τρόπο, ώστε στην οθόνη να παρουσιάζεται ολόκληρο. Για την αναγνώριση και αποκωδικοποίηση του QR Code υπάρχει στην οθόνη μία ορισμένη περιοχή που πρέπει να τοποθετηθεί ολόκληρο το QR Code. Η περιοχή αυτή περιβάλλεται από μία πιο σκουρόχρωμη περιοχή, ώστε να διαχωρίζεται η περιοχή που πρέπει να τοποθετήσει ο χρήστης το QR Code (Εικόνα 24). Η δραστηριότητα αναμένει από το χρήστη να τοποθετήσει ένα QR Code σε αυτή την περιοχή λαμβάνοντας συνέχεια στιγμιότυπα εικόνων από το περιεχόμενο που δείχνει η κάμερα μέχρι να εντοπίσει ένα QR Code και να συνεχίσει με την αποκωδικοποίηση του που γίνεται σε πολύ μικρό χρονικό διάστημα και αυτόματα. Ο κώδικας της δραστηριότητας CaptureActivity έχει τροποποιηθεί κατά τέτοιο τρόπο, ώστε με το που γίνει η αποκωδικοποίηση, γίνεται η μετάβαση στη δραστηριότητα ResultActivity μαζί με το αποτέλεσμα της αποκωδικοποίησης.

Η οθόνη που παρουσιάζεται στην CaptureActivity, υλοποιείται στο αρχείο capture.xml και αποτελείται από ένα FrameLayout, το οποίο ορίζεται από ένα surface view και καταλαμβάνει όλη την οθόνη. Το surface view χρησιμεύει στο να ανανεώνει συνέχεια την οθόνη από τη λήψη της κάμερας. Υπάρχει και ένα custom View το οποίο βρίσκεται πάνω από την επιφάνεια που δείχνει η κάμερα και δημιουργεί το εξωτερικό πλαίσιο (μά-

σκα). Τέλος υπάρχει ένα textView, το οποίο εμφανίζει ένα κείμενο για το που πρέπει να τοποθετηθεί το QR Code.



Εικόνα 24. Η οθόνη της CaptureActivity

Η δραστηριότητα **HistoryActivity** υλοποιεί ένα μέρος της λειτουργίας Ιστορικού της εφαρμογής. Η μετάβαση σε αυτή τη δραστηριότητα γίνεται από τη MainActivity και μόνο αν υπάρχουν δεδομένα στην εσωτερική βάση δεδομένων της εφαρμογής. Με την εκκίνηση της δραστηριότητας παρουσιάζονται τα δεδομένα που υπάρχουν στη βάση δεδομένων σε μορφή λίστας. Κάθε γραμμή της λίστας περιέχει δύο στοιχεία, που είτε είναι το όνομα του κτιρίου και το αποκωδικοποιημένο QR Code του, είτε το όνομα του κτιρίου με τιμή "Unknown" και το αποκωδικοποιημένο QR Code. Τα στοιχεία τοποθετούνται το ένα κάτω από το άλλο και κάθε γραμμή της λίστας διαχωρίζεται από μία γραμμή. Παρέχεται η δυνατότητα στο χρήστη με την επιλογή μίας γραμμής από τη λίστα να μεταβεί στην δραστηριότητα ResultActivity. Η διεπαφή της δραστηριότητας υλοποιείται στο αρχείο historyactivity.xml που ορίζεται η διάταξη LinearLayout με την ιδιότητα προσανατολισμού κάθετα και περιέχει δύο textview. Στην Εικόνα 25 παρουσιάζεται η διεπαφή της HistoryActivity.

Εικόνα 25. Η Οθόνη της δραστηριότητας HistoryActivity

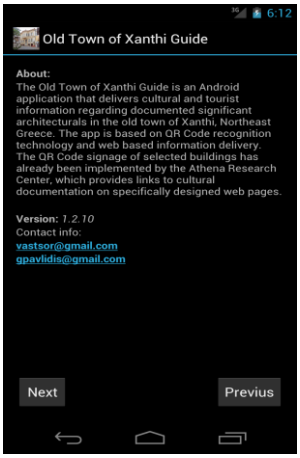
Σκοπός της δραστηριότητας **HelpActivity** είναι η εμφάνιση ενός κειμένου, που παρέχει πληροφορίες για την εφαρμογή και τη χρήση της. Η μετάβαση σε αυτή γίνεται από την αρχική οθόνη. Η οθόνη της δραστηριότητας HelpActivity αποτελείται από έξι TextViews και δύο πλήκτρα (Buttons). Τα TextViews είναι τοποθετημένα κάθετα, το ένα κάτω από το άλ-

λο. Το πρώτο TextView εμφανίζει τον τίτλο του θέματος που περιγραφεί το δεύτερο. Ακολουθεί το τρίτο TextView που εμφανίζει την τρέχον έκδοση της εφαρμογής. Το τέταρτο εμφανίζει τον τίτλο «Contacts info» και στα υπόλοιπα εμφανίζονται οι διευθύνσεις ηλεκτρονικού ταχυδρομείου που προσφέρονται στον χρήστη για επικοινωνία σχετικά με την εφαρμογή. Τέλος τα δύο πλήκτρα είναι τοποθετήμενα στην κάτω αριστερά και δεξία γωνία της οθόνης και έχουν τους τίτλους «Next»και «Previous» αντιστοίχα. Στην δραστηρίοτητα HelpActivity ύπαρχουν τρία διαφορετίκα θέματα που μπορεί να εμφανίστουν στην οθόνη. Αυτά είναι:

- **About**: Μία παραγράφος που παρέχει πληροφορίες για την εφαρμόγη.
- **Scan**: Μία παράγραφος που περιέχει οδήγιες χρήσεις για την λειτουργία σάρωσης.
- **History**: Μία παράγραφος που περίεχει οδήγιες χρήσεις για την λειτουργία του ιστορικού.

Με την πίεση του πλήκτρου «Next» αλλάζει το κείμενο που εμφανίζει το δεύτερο και το πρώτο TextView πηγαίνοντας με αύτο τον τρόπο στο επομένο θέμα που περιέχει η οθόνη βοήθειας. Το αντίστροφο γίνεται με την πίεση του πλήκτρου «Previous».

Στην Εικόνα 26 παρούσιαζεται η οθόνη της HelpActivity



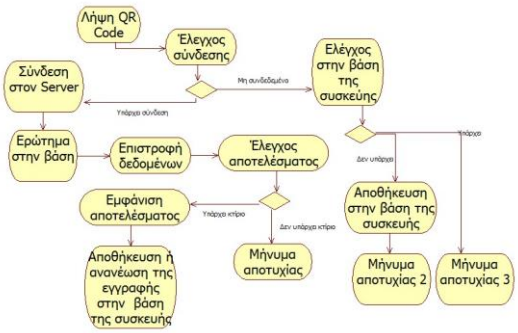
Εικόνα 26 Η οθόνη της HelpActicity

Στην δραστηριότητα **ResultActivity** υλοποιείται η σύνδεση στην υπηρεσία διαδικτύου που βρίσκεται η βάση δεδομένων και η παρουσίαση του αναλόγου αποτελέσματος. Η μετάβαση στη δραστηριότητα μπορεί να γίνει με δύο τρόπους και το πρώτο που ελέγχεται είναι από ποια δραστηριότητα κλήθηκε.

Στην περίπτωση που η δραστηριότητα ξεκίνησε από την CaptureActivity το πρώτο που ελέγχεται είναι αν υπάρχει διαθεσιμότητα πρόσβασης στο διαδίκτυο. Το αποτέλεσμα του ελέγχου αυτό καθορίζει την έκβαση του επόμενου βήματος:

- **Υπάρχει σύνδεση στο διαδίκτυο:** μεταφέρεται το αποκωδικοποιημένο QR Code που έλαβε η CaptureActivity σε μία μεταβλητή. Μετά γίνεται HTTP αίτημα με παράμετρο το QR Code στη διεύθυνση που είναι το αρχείο getData.php. Αν επιστρέψει ένα αντικείμενο JSON που έχει τη μορφή
  {"Success":1,"places":["qrcode":"πεδίο qrcode","name":"όνομα κτιρίου","description":"Περιγραφή κτιρίου","photourl":"url εικόνας"]}
  τότε εμφανίζεται στην οθόνη ο τίτλος του κτιρίου, η εικόνα του και η περιγραφή. Ταυτόχρονα η δραστηριότητα ελέγχει αν δεν υπάρχει στην εσωτερική βάση της συσκευής το εν λόγω κτίριο και αποθηκεύει το QR Code του κτιρίου, το όνομα του, την περιγραφή του κτιρίου και την εικόνα στην sd card της συσκευής: */mnt/sdcard/Pictures/xanthipictures/*. Στην περίπτωση που το getData.php επιστρέψει ένα αντικείμενο JSON που έχει τη μορφή
  {"Success":0, "message":"Place not found"}
  τότε στην οθόνη εμφανίζεται το μήνυμα
  "No information available for this QR Code".
- **Δεν υπάρχει σύνδεση στο διαδίκτυο:** ελέγχεται με ερώτημα στην εσωτερική βάση δεδομένων αν υπάρχει ήδη αποθηκευμένο το QR Code. Στην περίπτωση που δεν υπάρχει γίνεται αποθήκευση στην εσωτερική βάση δεδομένων της συσκεύης με όνομα κτιρίου «Unknown» και το QR Code που λήφθηκε από την σάρωση και στην οθόνη εμφανίζεται το μήνυμα «No internet connection available. The QR Code saved for later». Στην περίπτωση που είναι ήδη αποθηκευμένο, γίνεται έλεγχος για το εάν το όνομα του κτιριού έχει τιμή «Unknown» ή ένα πραγματικό όνομα κτιρίου. Στην περίπτωση που έχει την τιμή «Unknown» εμφανίζεται στην οθόνη το μήνυμα «No internet connection available.». Ενώ αν είναι ένα πραγματικό όνομα εμφανίζεται το μήνυμα «No internet connection available. You may check "όνομα κτιρίου" from history ».

Στην Εικόνα 27 παρουσιάζεται ένα διάγραμμα ροής της παραπάνω περίπτωσης.



Εικόνα 27. Διάγραμμα ροής της ResultsActivity από την CaptureActivity

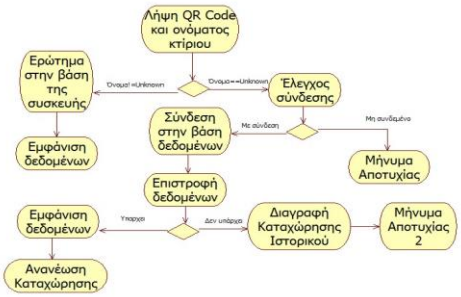
Στην περίπτωση που η δραστηριότητα ξεκίνησε από την HistoryActivity, στην αρχή μεταφέρονται το QR Code και το όνομα του κτιρίου από τη γραμμή ιστορικού που επιλέχτηκε στην οθόνη της δραστηριότητας HistoryActivity. Στην αρχή ελέγχεται η τιμή που έχει το όνομα του κτιρίου.

Στην περιπτώση που το όνομα του κτιρίου δεν έχει την τίμη «Unknown», αύτο σήμαινει ότι όλες οι πληροφορίες για το συγκεκριμένο κτίριο υπάρχουν στην βάση της συσκευής. Γίνεται ερώτημα στην εσωτερική βάση δεδομένων και επιστρέφονται η περιγραφή του κτιρίου και η θέση που βρίσκεται η εικόνα του στην συσκεύη. Τέλος εμφανίζονται στην οθόνη το όνομα του κτιρίου, η εικόνα του και η περιγραφή του.

Στην περιπτώση που το όνομα του κτιρίου έχει την τιμή «Unknown» τότε γίνεται έλεγχος σύνδεσης στο διαδίκτυο και η δραστηριότητα ανάλογα το αποτέλεσμα του έλεγχου συνεχίζει σε δύο διαφορετικά βήματα:

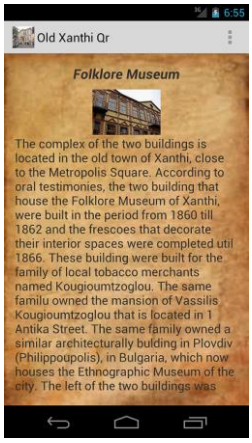
- **Υπάρχει σύνδεση στο διαδίκτυο:** Η δραστηριότητα συνεχίζει με παρόμοιο τρόπο όπως και γίνεται στην ιδία περίπτωση όταν έχει ξεκινήσει από την CaptureActivity μέχρι και το σημείο που επιστρέφεται το αντικείμενο JSON. Αν το αντικείμενο JSON επιστρέφει δεδομένα από τη βάση δεδομένων, τότε γίνεται ενημέρωση του πεδίου ονόματος αν το όνομα του κτιρίου που μεταφέρθηκε από την HistoryAcitivity έχει την τιμή "Unkown" και εμφανίζονται στην οθόνη τα δεδομένα από το αντικείμενο JSON. Αν το αντικείμενο JSON δεν επιστρέψει δεδομένα από τη βάση δεδομένων τότε διαγράφεται η καταχώρηση από την εσωτερική βάση και εμφανίζεται το μήνυμα "No information available for this QR Code".
- **Χωρίς σύνδεση στο διαδίκτυο**: Τότε εμφανίζεται στην οθόνη το μήνυμα "No internet connection available".

Στην Εικόνα 28 παρουσιάζεται το διάγραμμα ροής όταν η δραστηριότητα ξεκινά από την HistoryActivity.



Εικόνα 28. Διάγραμμα ροής της ResultsActivity από την HistoryActivity

Αν υπάρχει σύνδεση στο διαδίκτυο στην οθόνη εμφανίζεται μία μπάρα προόδου που εξαφανίζεται όταν η διαδικασία επιστρέψει αποτέλεσμα. Αν το αποτέλεσμα περιέχει δεδομένα για ένα κτίριο τότε εμφανίζεται στην οθόνη το όνομα του κτιρίου, μία εικόνα του και η περιγραφή του. Στην Εικόνα 29 παρουσιάζεται ένα παράδειγμα της οθόνης όπου εμφανίζεται ένα πλήρες αποτέλεσμα. Αν το αποτέλεσμα που επιστρέψει η διαδικασία δεν περιέχει δεδομένα για ένα κτίριο τότε εμφανίζεται στην οθόνη ένα μήνυμα που ενημερώνει το χρήστη ότι το κτίριο δεν υπάρχει. Στην περίπτωση που δεν υπάρχει σύνδεση στο διαδίκτυο στην οθόνη εμφανίζεται ένα μήνυμα που ενημερώνει το χρήστη ότι δεν υπάρχει σύνδεση στο διαδίκτυο.



Εικόνα 29. Παράδειγμα σελίδας παρουσίασης κτιρίου

Βιβλιογραφία

[1] QR Code http://en.wikipedia.org/wiki/QR_code, τελευταία επίσκεψη: 10 Σεπ. 2013.

[2] Cambridge University Press, Hiroko Kato, Douglas Chai, Keng T. Tan, “Barcodes for mobile device”, 8 Απρ 2008, 51-59.

[3] Tan jin Soon, “Three way QR Code”, Synthesis Journal 2008, 2008, 59-78.

[4] QR Code Denso Wave. http://www.qrcode.com/en/. 31 Sep 2013

[5] Android development, http://developer.android.com, 2 Jan, 2014.

[6] Android, http://el.wikipedia.org/wiki/Android, 2 Jan 2014.

[7] Android version, http://en.wikipedia.org/wiki/Android_version_history, 2 Jan 2014

[8] JSON, http://www.json.org/, 28 Nov 2013

[9] Web Services, http://en.wikipedia.org/wiki/Web_service, 29 Nov 2013

Παράρτημα Α – Πηγαίος Κώδικας Εφαρμογής

Πηγαίος κώδικας της κλάσης JSONParser

```
public class JSONParser {
     static InputStream is = null;
     static JSONObject jObj = null;
     static String json = "";

// constructor
        public JSONParser() {
}
// function get json from url
// by making HTTP POST or GET mehtod
public JSONObject makeHttpRequest(String url, String method,
    List<NameValuePair> params) {
    Making HTTP request
   try {
// check for request method
if(method == "POST"){
// request method is POST
// defaultHttpClient
DefaultHttpClient httpClient = new DefaultHttpClient();
HttpPost httpPost = new HttpPost(url);
httpPost.setEntity(new UrlEncodedFormEntity(params));

HttpResponse httpResponse = httpClient.execute(httpPost);
HttpEntity httpEntity = httpResponse.getEntity();
is = httpEntity.getContent();
httpClient.clearRequestInterceptors();

}else if(method == "GET"){
// request method is GET
DefaultHttpClient httpClient = new DefaultHttpClient();
String paramString = URLEncodedUtils.format(params, "utf-8");
url += "?" + paramString;
HttpGet httpGet = new HttpGet(url);
Log.d("MyUrl",httpGet.toString());

HttpResponse httpResponse = httpClient.execute(httpGet);
HttpEntity httpEntity = httpResponse.getEntity();
is = httpEntity.getContent();
httpClient.clearRequestInterceptors();
}

} catch (UnsupportedEncodingException e) {
```

```
e.printStackTrace();
        } catch (ClientProtocolException e) {
        e.printStackTrace();
        } catch (IOException e) {
        e.printStackTrace();
}
        try {
BufferedReader reader = new BufferedReader(new InputStreamReader(
is, "iso-8859-1"), 8);
        StringBuilder sb = new StringBuilder();
        String line = null;
        while ((line = reader.readLine()) != null) {
                sb.append(line + "\n");
        }
        is.close();
        json = sb.toString();
        } catch (Exception e) {
        Log.e("Buffer   Error",   "Error   converting   result   "   +
e.toString());
                }

                // try parse the string to a JSON object
                try {
        jObj = new JSONObject(json);
        } catch (JSONException e) {
        Log.e("JSON Parser", "Error parsing data " + e.toString());
        }

                // return JSON String

                return jObj;

        }
}
```

Πηγαίος κώδικας της MainActivity

```
public class MainActivity extends Activity  {
        Button bscan;
        Button babout;
        Button bhistory;

        @Override
        protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        bscan=(Button)findViewById(R.id.buttonScan);
        babout=(Button)findViewById(R.id.buttonAbout);
        bhistory=(Button)findViewById(R.id.buttonHistory);
        }
        @Override
        public boolean onCreateOptionsMenu(Menu menu) {
// Inflate the menu; this adds items to the action bar if it is pre-
sent.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
        }
        public void openScanner(View v) {
        Intent scanIntent=new Intent(this,CaptureActivity.class);
        startActivity(scanIntent);
        }
        public void openHistory(View view){
        int counter=0;
```

```
        ServerHelper dbHelper=new ServerHelper(this);
        counter=dbHelper.getBuildingsCount();
if(counter!=0){
Intent historyIntent=new Intent(this,HistoryAcitivity.class);
        startActivity(historyIntent);
        }
else {
        String message="No history Records available";
        Toast.makeText(getApplicationContext(),              message,
Toast.LENGTH_SHORT).show();
        }
}
public void openAbout(View v) {
        Intent i=new Intent(this,HelpActivity.class);
        startActivity(i);
        }
}
```

Πηγαίος κώδικας της ResultActivity

```
public class GetXanthiData extends Activity {
    public String activityStarter="capture";
    public String buildingname;
    public String imageName;
    TextView txvTitle;
    TextView txvDescription;
    ImageView img;
    LinearLayout mainLayout;
    private ProgressDialog pDialog;
    public String resultData;
    public String name;
    public String pName;
    public String pDescription;
    public String urlName;
    public boolean saveState;
    ServerHelper history=new ServerHelper(this);
    JSONParser jsonParser = new JSONParser();
    Private                        final                        String
url_get_place="http://oldtownofxanthi.ipet.gr/getData.php";
    private static final String TAG_CAPTURE="capture";
    private static final String TAG_HISTORY="history";
    private static final String TAG_UHISTORY="unhistory";
    private static final String TAG_SUCCESS = "success";
    private static final String TAG_PLACES ="places";
    private         static         final         String         im-
ageUrl="http://oldtownofxanthi.ipet.gr/images/";

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

        mainLayout=                                        (LinearLay-
out)findViewById(R.id.placeMainLayout);
        mainLayout.setVisibility(View.INVISIBLE);
        txvTitle = (TextView)findViewById(R.id.txtPlaceName);
        txvDescription
=(TextView)findViewById(R.id.txtPlaceDescription);

        img = (ImageView) findViewById(R.id.placeimageView);


         Intent intent=getIntent();
```

```
        resultData=intent.getExtras().getString("result");
        activityStart-
er=intent.getExtras().getString("activityName");
        name=resultData;
        boolean networkState=haveNetworkConnection();
        Log.d("activity start", activityStarter);
        if(activityStarter.equals(TAG_CAPTURE)  &&  haveNetworkCon-
nection()) {
                Log.d("connection","true");
                new GetDataFromDb().execute();
            }

        if(activityStarter.equals(TAG_CAPTURE)  &&  !haveNetworkCon-
nection()) {
            if(!history.checkIfExist(name)){
               history.addServer("Unknown", name,"empty" ,"empty");
               String message="No Internet connection available.  A
building with name Unknown has been saved in History";
               Toast.makeText(getApplicationContext(),     message,
Toast.LENGTH_LONG).show();
                }
                else {
                    String message="No Internet connection availa-
ble.  A building with name Unknown has been saved in History";
                      Toast.makeText(getApplicationContext(), mes-
sage, Toast.LENGTH_LONG).show();

                }
            }

        if(activityStarter.equals(TAG_UHISTORY)) {
            if(haveNetworkConnection()) {
                new GetDataFromDb().execute();
                }
            else {
                String message="No Internet connection available.";
                Toast.makeText(getApplicationContext(),    message,
Toast.LENGTH_LONG).show();
            }

        }

        if(activityStarter.equals(TAG_HISTORY)) {
            String                 sdcard=                 Environ-
ment.getExternalStoragePublicDirectory(Environment.DIRECTORY_PICTURE
S).toString() +"/xanthipicture/";
            HistoryBuilding building=history.getBuilding(name);
            mainLayout.setVisibility(View.VISIBLE);
            txvTitle.setText(building.getBname());
            txvDescription.setText(building.getBinfo());
            Bitmap
bm=BitmapFactory.decodeFile(sdcard+building.getBimg());
            img.setImageBitmap(bm);

        }
    }

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

```
        return true;
    }
    class GetDataFromDb extends AsyncTask<String, String, String>{
        boolean successState=false;
        String pictureLink;
        /**
         * Before starting background thread Show Progress Dialog
         */
        @Override
        protected void onPreExecute(){
            super.onPreExecute();
            pDialog = new ProgressDialog(GetXanthiData.this);
            pDialog.setMessage("Loading ... Please Wait");// String
            pDialog.setIndeterminate(false);
            pDialog.setCancelable(true);
            pDialog.show();
        }
        @Override
        protected String doInBackground(String... args) {
            int success;
            try{
                List<NameValuePair>     params     =     new     Ar-
rayList<NameValuePair>();
                params.add(new        BasicNameValuePair("objectId",
name));

                // getting product details by making HTTP request
                // Note that product details url will use GET re-
quest

                JSONObject           json           =           json-
Parser.makeHttpRequest(url_get_place,"GET",params);

                // check your log for json response
                Log.d("Single Product Details", json.toString());
                //JSON success tag
                success=json.getInt(TAG_SUCCESS);
                if(success==1) {
                    successState=true;
                     // successfully received product details
                    JSONArray               placeObj               =
json.getJSONArray(TAG_PLACES);
                    // JSON Array

                    // get first product object from JSON Array
                    JSONObject place = placeObj.getJSONObject(0);

                    pName=place.getString("placeName");
                    pDescription=place.getString("pdescription");



                    imageName=place.getString("photoLink");

                    pic-
tureLink=imageUrl+place.getString("photoLink");
                    if(activityStarter.equals(TAG_CAPTURE) || activ-
ityStarter.equals(TAG_UHISTORY)){
                         if(history.checkIfExist(name)) {
                             history.updateBuilding(name,      pName,
pDescription, imageName);
```

```
                        //   history.addServer(pName,          name,
pDescription, imageName);
                            }
                        else {
                            history.addServer(pName,          name,
pDescription, imageName);
                        }
                    }

                    successState=true;
                   return pictureLink;


                }
                else {
                    successState=false;
                    if(activityStarter.equals(TAG_HISTORY) || activ-
ityStarter.equals(TAG_UHISTORY)) {

                            history.deleteBuilding(name);
                    }
                }
            }catch(JSONException e){
                e.printStackTrace();

            }

            return null;

        }
        @Override
        protected void onPostExecute(String res)

        {
            //super.onPostExecute(res);
            pDialog.dismiss();
            if(successState){
            txvTitle.setText(pName);
            txvDescription.setText(pDescription);
            GetPictureImage getPictures= new GetPictureImage();
            getPictures.execute(pictureLink);
            }
            else
            {
                String message="No information available for this QR
Code";

                Toast.makeText(getApplicationContext(),     message,
Toast.LENGTH_SHORT).show();
            }
        }
    }
    /*
     * Class that show image
     */

    class GetPictureImage extends AsyncTask<String, String, Bitmap>
    {
        Bitmap bitmap1=null;
        @Override
        protected void onPreExecute() {
            pDialog=new ProgressDialog(GetXanthiData.this);
```

```
            pDialog.setMessage("Dl photo");

            pDialog.setCancelable(true);
            pDialog.setIndeterminate(false);
            pDialog.show();
        }

        @Override
        protected Bitmap doInBackground(String... params) {
            String newUrl=params.toString();
            int count=params.length;
            try {
                Bitmap         bitmap         =         BitmapFacto-
ry.decodeStream((InputStream)new URL(params[0]).getContent());
                bitmap1=bitmap;
                savePicture(bitmap1, imageName);
                return bitmap;
            } catch (MalformedURLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
            return null;
        }
        @Override
        protected void onPostExecute(Bitmap result){
             super.onPostExecute(result);
             pDialog.dismiss();
             img.setImageBitmap(result);
             mainLayout.setVisibility(View.VISIBLE);

        }

    }
    private boolean haveNetworkConnection() {
        boolean haveConnectedWifi = false;
        boolean haveConnectedMobile = false;

        ConnectivityManager  cm  =  (ConnectivityManager)  getSystem-
Service(Context.CONNECTIVITY_SERVICE);
        NetworkInfo[] netInfo = cm.getAllNetworkInfo();
        for (NetworkInfo ni : netInfo) {
            if (ni.getTypeName().equalsIgnoreCase("WIFI"))
                if (ni.isConnected())
                    haveConnectedWifi = true;
            if (ni.getTypeName().equalsIgnoreCase("MOBILE"))
                if (ni.isConnected())
                    haveConnectedMobile = true;
        }
        return haveConnectedWifi || haveConnectedMobile;
    }

    private void savePicture(Bitmap bm, String imgName)
    {
     OutputStream fOut = null;
     String            strDirectory1            =            Environ-
ment.getExternalStoragePublicDirectory(Environment.DIRECTORY_PICTURE
S).toString();
```

```
     String            strDirectory            =            Environ-
ment.getExternalStorageDirectory().toString();
     Log.d("Directory path", strDirectory);
     File dirs =new File(strDirectory1+"/xanthipicture");
     dirs.mkdirs();
     File f = new File(dirs.getAbsolutePath(), imgName);

     try {
      fOut = new FileOutputStream(f);

      /**Compress image**/
      bm.compress(Bitmap.CompressFormat.JPEG, 85, fOut);
      fOut.flush();
      fOut.close();
     } catch (Exception e) {
      e.printStackTrace();
     }
    }


}
public class HistoryAcitivity extends ListActivity {
private final static String CHECKHISTORY="Unknown";
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);

TextView msg=(TextView)findViewById(R.id.text2);
final ArrayList<Map<String, String>> list= buildData();
String[] from= {"Building", "url"};
int[] to= { R.id.text1,R.id.text2};
SimpleAdapter adapter = new SimpleAdapter(this, list,
R.layout.simple_list_item_2, from, to);
setListAdapter(adapter);
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
return true;
}
private ArrayList<Map<String, String>> buildData() {
ArrayList<Map<String,  String>>  list  =  new  ArrayList<Map<String,
String>>();
ServerHelper dbHelper=new ServerHelper(this);
List<HistoryBuilding> listHistory;
listHistory=dbHelper.getAllBuilding();
int j=0;
while(j< listHistory.size()){
HistoryBuilding building = new HistoryBuilding();
building=listHistory.get(j);
list.add(putData(building.getBurl(),building.getBname()));
j++;
}
return list;

}
private HashMap<String, String> putData(String name, String purpose)
{
HashMap<String, String> item = new HashMap<String, String>();
item.put("Building", name);
item.put("url", purpose);
return item;
```

```
}
@Override
protected void onListItemClick(ListView list, View view, int posi-
tion, long id) {
super.onListItemClick(list, view, position, id);
ArrayList<Map<String, String>> lista= buildData();
String msg=lista.get(position).get("url");
String check=lista.get(position).get("Building");
if(check.equals(CHECKHISTORY)) {
Intent getData = new Intent(this,GetXanthiData.class);
getData.putExtra("activityName", "unhistory");
getData.putExtra("result", msg);
startActivity(getData);
this.finish();
}
else {
Intent getData= new Intent(this,GetXanthiData.class);
getData.putExtra("activityName", "history");
getData.putExtra("result", msg);
startActivity(getData);
this.finish();
}
}

}
```

Πηγαίος κώδικας της εσωτερική βάσης

```
public class ServerHelper extends SQLiteOpenHelper {

    @Override
    public synchronized void close() {
        // TODO Auto-generated method stub
        super.close();
    }

    @Override
    public void onOpen(SQLiteDatabase db) {
        // TODO Auto-generated method stub
        super.onOpen(db);
    }

    private static final int DATABASE_VERSION = 2;
    private static final String DATABASE_NAME = "DbXanthi";
    private static final String SERVER_TABLE_NAME = "buildings";
    private static final String KEY_ID="id";
    private static final String KEY_URL="burl";
    private static final String KEY_NAME="bname";
    private static final String KEY_INFO="binfo";
    private static final String KEY_IMG="bimg";
    private      static      final      String[]      COLUMNS      =
{KEY_ID,KEY_NAME,KEY_URL,KEY_INFO, KEY_IMG};

    public ServerHelper(Context context
            ) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
        // TODO Auto-generated constructor stub
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        /* String SERVER_TABLE_CREATE = "CREATE TABLE IF NOT EXISTS
"
```

```
                + SERVER_TABLE_NAME+ "(  id INTEGER PRIMARY KEY AU-
TOINCREMENT, " +
         "bname TEXT , "+"burl TEXT UNIQUE )"; */
        String SERVER_TABLE_CREATE = "CREATE TABLE IF NOT EXISTS "
                + SERVER_TABLE_NAME+ "(  burl TEXT PRIMARY KEY, " +
         "bname TEXT, "+ "binfo TEXT,"+ "bimg TEXT )";
         db.execSQL(SERVER_TABLE_CREATE);
    }

    @Override
    public  void  onUpgrade(SQLiteDatabase  db,  int  oldVersion,  int
newVersion) {
            // Drop older books table if existed
        db.execSQL("DROP TABLE IF EXISTS buildings");

        // create fresh books table
        this.onCreate(db);

    }
     public void addServer(String bname, String burl, String binfo,
String bimg){
            //for logging
            Log.d("addBook", bname+""+burl);

            // 1. get reference to writable DB
            SQLiteDatabase db = this.getWritableDatabase();

            // 2. create ContentValues to add key "column"/value
            ContentValues values = new ContentValues();
            values.put(KEY_URL, burl);
            values.put(KEY_NAME, bname);
            values.put(KEY_INFO, binfo);
            values.put(KEY_IMG, bimg);
            // 3. insert
            db.insert(SERVER_TABLE_NAME, // table
                    null, //nullColumnHack
                    values);  //  key/value  ->  keys  =  column  names/
values = column values

            // 4. close
            db.close();
        }
       // Getting All Buildings
     public List<HistoryBuilding> getAllBuilding() {
        List<HistoryBuilding>     buildingList     =     new     Ar-
rayList<HistoryBuilding>();
        // Select All Query
        String selectQuery = "SELECT  * FROM " + SERVER_TABLE_NAME;

        SQLiteDatabase db = this.getWritableDatabase();
        Cursor cursor = db.rawQuery(selectQuery, null);

        // looping through all rows and adding to list
        if (cursor.moveToFirst()) {
            do {
                HistoryBuilding building = new HistoryBuilding();
              building.setBname(cursor.getString(0));
              building.setBurl(cursor.getString(1));

                buildingList.add(building);
            } while (cursor.moveToNext());
        }
```

```
        cursor.close();
      //  db.close();
        // return building list
        return buildingList;
    }

    public boolean checkIfExist(String burl) {
        SQLiteDatabase db = this.getReadableDatabase();
        Cursor cursor = db.query(SERVER_TABLE_NAME, new String[] {
                KEY_NAME, KEY_URL }, KEY_URL + "=?",
                new String[] { String.valueOf(burl) }, null, null,
null, null);

        boolean ret=cursor.moveToFirst();
        cursor.close();
        return ret;
        }

    public int getBuildingsCount() {
        String countQuery = "SELECT  * FROM " + SERVER_TABLE_NAME;
        SQLiteDatabase db = this.getReadableDatabase();
        Cursor cursor = db.rawQuery(countQuery, null);
        int i=cursor.getCount();
        cursor.close();

        // return count
        return i;
    }
     public HistoryBuilding getBuilding(String burl) {

         SQLiteDatabase db = this.getReadableDatabase();

            Cursor cursor = db.query(SERVER_TABLE_NAME, new String[]
{ KEY_URL,
                    KEY_NAME, KEY_INFO, KEY_IMG}, KEY_URL + "=?",
                    new String[] { String.valueOf(burl) }, null,
null, null,null);
            if (cursor != null)
                cursor.moveToFirst();

            HistoryBuilding building = new HistoryBuild-
ing(cursor.getString(1),cursor.getString(0),cursor.getString(2),curs
or.getString(3));
            cursor.close();
           // db.close();
            return building;
     }
     public int updateBuilding(String burl, String bname, String
bdescription, String bimg) {
            SQLiteDatabase db = this.getWritableDatabase();

            ContentValues values = new ContentValues();
            values.put(KEY_NAME, bname );
            values.put(KEY_INFO, bdescription);
            values.put(KEY_IMG, bimg);


            // updating row
             int i= db.update(SERVER_TABLE_NAME, values, KEY_URL + "
= ?",
                    new String[] { String.valueOf(burl) });
             db.close();
```

```
            return i;
        }

        // Deleting single building
    public void deleteBuilding(String burl) {
        SQLiteDatabase db = this.getWritableDatabase();
        db.delete(SERVER_TABLE_NAME, KEY_URL + " = ?",
                new String[] { String.valueOf(burl) });
        db.close();
    }

}
```

Παράρτημα Β – Οδηγίες εγκατάστασης Andoid ADT

Εγκατάσταση του Android SDK Bunlde

Το Android SDK προσφέρει στο προγραμματιστή όλες τις βιβλιοθήκες της πλατφόρμας Android και όλα τα απαραίτητα εργαλεία για την ανάπτυξη εφαρμογών στο Android.

Στην ιστοσελίδα http://developer.android.com/sdk/index.html προσφέρεται για λήψη ένα πακέτο του ADT (Android Development Tools) που περιέχει τα βασικά κομμάτια από το SDK και το περιβάλλον ανάπτυξης Eclipse με ενσωματωμένο το ADT για την ανάπτυξη εφαρμογών σε περιβάλλον Android.

Το πακέτο περιέχει:

- Το Eclipse και το πρόσθετο του ADT.
- Τα εργαλεία από το Android SDK.
- Τα εργαλεία πλατφόρμας του Android.
- Την τελευταία έκδοση του Android.
- Καθώς και την τελευταία εικόνα συστήματος του Android για τον εξομοιωτή.

Αμέσως μετά την λήψη του πάκετου, αποσυμπίεζουμαι το αρχείο με όνομα adt-bundle-windowsx86.zip σε ένα φακέλο της επιλογής μας, οπώς για παράδειγμα C:\Development. Εφόσον ολοκλήρωθει η αποσυμπίεση θα έχουν δημιουργηθεί δύο φάκελοι με όνομα eclipse και SDK στο κατάλογο που επιλέχτηκε να γίνει η αποσυμπίεση. Εκτελούμε το αρχείο eclipse που βρίσκεται στο φάκελο C:\Development\adt-bundle-windows86x\eclipse\ και φορτώνεται το περιβάλλον ανάπτυξης Eclipse με το πρόσθετο ADT και το SDK του Android.

[Αυτή η σελίδα είναι εσκεμμένα λευκή]

ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ
ΚΕΝΤΡΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΤΕ

ΣΥΣΤΗΜΑ ΑΝΑΓΝΩΡΙΣΗΣ QR CODES ΓΙΑ ΔΙΑΣΥΝΔΕΣΗ ΜΕ ΔΙΑΔΙΚΤΥΑΚΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ/ΠΥΛΕΣ ΜΕΣΩ SQL ΚΛΗΣΕΩΝ ΣΕ ΚΙΝΗΤΑ ΤΗΛΕΦΩΝΑ ΜΕ ΛΕΙΤΟΥΡΓΙΚΟ ΣΥΣΤΗΜΑ ANDROID

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Βασίλης Τσορμπατζούδης

Στο πλαίσιο της παρούσας εργασίας αναπτύσσεται σύστημα διασύνδεσης σήμανσης με QR Codes με διαδικτυακές βάσεις δεδομένων με στόχο την παρουσίαση πληροφόρησης σε έξυπνες φορητές συσκευές που χρησιμοποιούν το λειτουργικό σύστημα Android. Για τη δημιουργία χρήσιμης εφαρμογής πάνω σε πραγματικά δεδομένα επιλέχθηκε η χρήση σήμανσης QR Codes που είχε ήδη εγκατασταθεί σε ιστορικά κτήρια της παλιάς πόλης της Ξάνθης από το Ερευνητικό Κέντρο «Αθηνά», η οποία συνοδεύονταν από αντίστοιχη τεκμηρίωση, κατά κύριο λόγο, για τουριστική χρήση. Έτσι, αναπτύχθηκε διαδικτυακή σχεσιακή βάση δεδομένων που συμπεριέλαβε το πολιτιστικό περιεχόμενο της τεκμηρίωσης και η αντίστοιχη εφαρμογή για κινητές συσκευές Android η οποία αναγνωρίζει μέσω κάμερας τους κώδικες και διασυνδέει με τη βάση δεδομένων (με κλήσεις SQL) παρέχοντας σε κάθε επισκέπτη της παλιάς πόλης της Ξάνθης τουριστική ενημέρωση για κάποιο κτήριο ενδιαφέροντος. Η βάση δεδομένων είναι ενεργή στο διαδίκτυο και η εφαρμογή είναι διαθέσιμη online στο Google Play Store στις διευθύνσεις:
(web) http://play.google.com/store/apps/details?id=com.bill.qrxanthiold
(android) market://details?id=com.bill.qrxanthiold

INTEGRATED SYSTEM FOR THE CONNECTION OF QR CODE SIGNAGE WITH WEB DATABASES/PORTALS THROUGH SQL CALLS ON ANDROID SMART MOBILE DEVICES

THESIS

Vasilis Tsorbatzoudis

In this work an integrated system for the connection of QR Code signage with web databases has being developed in order to provide information in smart mobile devices operating on Android OS. To be practical and useful on real-world data, the existing QR Code signage on historic buildings in the old town of Xanthi has been selected as a case study. This signage was developed by the "Athena" Research Center and was accompanied by documentation provided online mainly for touristic purposes. A relational database has been developed to include the cultural content of the documentation and an Android app has been implemented to provide connection between the QR Code signage with the database (through SQL calls) and to provide tourist information to the visitors of the old town of Xanthi. The database is online and the app is available through the Google Play Store at:
(web) http://play.google.com/store/apps/details?id=com.bill.qrxanthiold
(android) market://details?id=com.bill.qrxanthiold