# Ευχαριστίες


## Δόξα τω Κυρίω ημών Ιησού Χριστό

που με αξίωσε να συνεργαστώ με αυτούς τους αξιόλογους καθηγητές του ΤΕΙ Ηπείρου και που μου έδωσε τόσους ανθρώπους που με στήριξαν κατά την διάρκεια της εκπόνησης της πτυχιακής εργασίας καθώς και καθ όλη την διάρκεια της φοίτησής μου.

# Περιεχόμενα

# 1. Εισαγωγή

## 1.1 Σκοπός της εργασίας

Η Εφαρμογή έχει σκοπό να διευκολύνει την ενημερώσει του χρήστη παρέχοντας ένα μεγάλο αριθμό ειδήσεων και άρθρων στην κινητή συσκευή του android. Στην σημερινή εποχή ο καταιγισμός των ειδήσεων είναι τεράστιος και σε αυτό έχει συμβάλει πάρα πολύ το διαδίκτυο.
Η εργασία αυτή έχει ως σκοπό την δημιουργία εφαρμογής που θα παρέχει στον χρήστη επιλογές από πληθώρα χριστιανικών ιστοσελίδων δίνοντας του την δυνατότητα να ενημερωθεί εύκολα γρήγορα και διασκεδαστικά.

## 1.2 Προβλήματα που επιλύει

Ο χρήστης ενός smartphone έχει πλέον την απαίτηση να μπορεί να συνδεθεί σε ένα site απευθείας από το κινητό του.

Με την εφαρμογή αυτή:

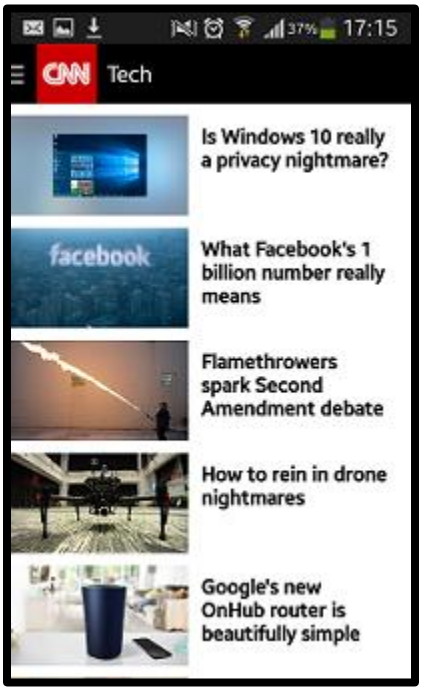
- παρέχεται απευθείας πρόσβαση σε αρκετά ενημερωτικά site.
- δημιουργεί πιο ‘’καλές σχέσεις”’ ανάμεσα στον χρήστη και στις ιστοσελίδες κάνοντας τις ιστοσελίδες δημοφιλέστερες.
- παρέχει στον χρήστη μόνο την πληροφορία που τον αφορά χωρίς περισσότερο περιεχόμενο που ίσως να τον κουράζει.
- ο χρήστης δεν χάνει χρόνο πληκτρολογώντας διευθύνσεις ιστοσελίδων στο web περιηγητή του.
- παρέχεται η δυνατότητα στον χρήστη να διαβάσει κάποιο άρθρο εκτός σύνδεσης και να το αποθηκεύσει στην συσκευή του.

## 1.3 Παρόμοια προγράμματα λογισμικού

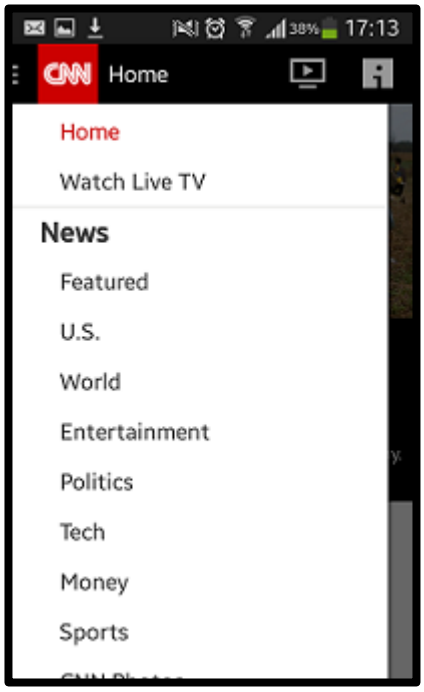
Τα πιο διάσημα κανάλια ενημέρωσης έχουν δημιούργησει εφαρμογές για να δίνουν την δυνατότητα στους αναγνώστες τους να ενημερώνονται από το κινητό τους τηλέφωνο και από το tablet τους. Όπως φαίνεται στις εικόνες που παραθέτουμε, από το κανάλι CNN, παρακάτω:


*Εικόνα 1 : Αρχικό μενού*


*Εικόνα 2 : Περιεχόμενο άρθρου*




*Εικόνα 3 : Λίστα ειδήσεων*

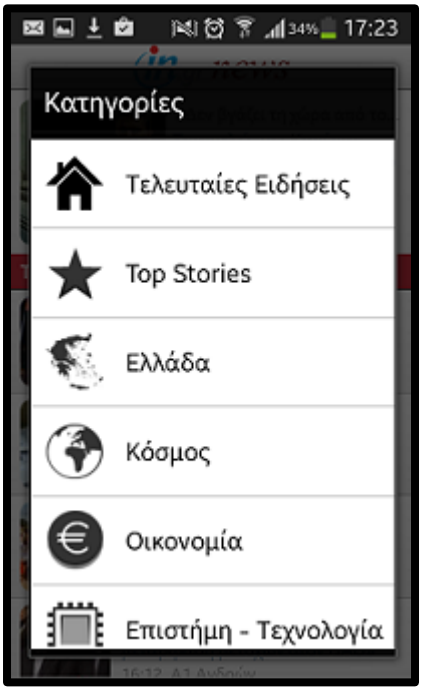



*Εικόνα 4 : "Κουρτίνα" Κατηγοριών*

Ακόμα, πολλές άλλες ενημερωτικές ιστοσελίδες του διαδικτύου, όπως το in.gr, έχουν δημιου-

ργήσει παρόμοιες εφαρμογές android για να κρατούν ενημερωμένους τους επισκέπτες τους.


*Εικόνα 5 : Αρχική σελίδα*




*Εικόνα 6 : Επιλογή κατηγοριών*


*Εικόνα 7 : Περιεχόμενο άρθρου*

## 1.4 Χρησιμοποιούμενο λογισμικό

Για την ανάπτυξη της εφαρμογής χρησιμοποιήθηκε το Android Development Tools (ADT). Το Android Development Tools (ADT) είναι ένα plugin για τον IDE Eclipse που επεκτείνει τις δυνατότητες του Eclipse και μας δίνει την δυνατότητα να δημιουργήσουμε νέα έργα Android. Το ADT μας παρέχει την δυνατότητα να χρησιμοποιήσουμε προσομοιωτές κινητών συσκευών και προσομοιωτές tablets. Ακόμα χρησιμοποιώντας το Eclipse μπορούμε να κάνουμε εύκολα αποσφαλμάτωση, να χρησιμοποιήσουμε τις βιβλιοθήκες που μας ενδιαφέρουν και ακόμη, να εξάγουμε υπογεγραμμένα (ή unsigned) .apk αρχεία για την διανομή της εφαρμογή μας.

Επίσης χρησιμοποιήθηκε και μία μοντέρνα βιβλιοθήκη REST client για το Android που δημιουργήθηκε από την Square Inc (2013-2015)[1]. Παρέχει έναν βολικό τρόπο για την αποστολή αιτημάτων στο διαδίκτυο ενώ επίσης λαμβάνει δεδομένα JSON ή XML από τις υπηρεσία web απλά και όμορφα. Το Rertofit όταν λάβει την απάντηση από τον διακομιστή, ταιριάζει τα δεδομένα της απάντησης σε αντικείμενα Plain Old Java Object(POJO).

Μία ακόμη βιβλιοθήκη που χρησιμοποιήθηκε είναι και η GSON. Αυτή η βιβλιοθήκη, υποστηρίζεται από την βιβλιοθήκη Retrofit και χρησιμοποιείται για να αυτοματοποιεί την ανάλυση των διαδικτυακών δεδομένων JSON που λαμβάνει η εφαρμογή.

Χρησιμοποιήθηκε επίσης, άλλη μία βιβλιοθήκη της Square Inc, η Picasso. Η βιβλιοθήκη αυτή επιτρέπει την προσθήκη εικόνων σε εφαρμογές Android. βελτιώνοντας την οπτική αίσθηση των εφαρμογών. Ένα ακόμα από τα πολλά πλεονεκτήματα της βιβλιοθήκης είναι ότι απλουστεύει την φόρτωση εικόνων σε εφαρμογές - συχνά σε μία γραμμή κώδικα!

Το YouTube Android API Player χρησιμοποιήθηκε γιατί παρέχει τη δυνατότητα να ενσωματώσει λειτουργίες αναπαραγωγής βίντεο στις Android εφαρμογές σας. Το API ορίζει τις μεθόδους για τη φόρτωση και την αναπαραγωγή βίντεο του YouTube και για την προσαρμογή και τον έλεγχο των βίντεο που αναπαράγει η εφαρμογή. Έτσι, για παράδειγμα, παρέχονται στον χρήστη πολλές και χρήσιμες επιλογές όπως να παίξει, να κάνει παύση, και να αναζητήσει ένα συγκεκριμένο σημείο στο ήδη φορτωμένο βίντεο.

# 2. Έξυπνα κινητά τηλέφωνα

## 2.1 Οι πρώτες κινητές συσκευές

Το τηλέφωνο είναι μια ηλεκτρική συσκευή με την οποία επιτυγχάνεται η μεταβίβαση της φωνής, δηλαδή παρέχεται η δυνατότητα συνομιλίας ανάμεσα σε δυο πρόσωπα που βρίσκονται σε απόσταση. Ο εφευρέτης του είναι ο Γκραχαμ Μπελ.

Το 1940, μετά τον δεύτερο παγκόσμιο πόλεμο ξεκίνησε η ιστορία του κινητού τηλεφώνου. Το κινητό τηλέφωνο είναι μία ασύρματη συσκευή που συνδέεται σε ένα τηλεφωνικό δίκτυο και προσφέρεται στο ευρύ κοινό από κάποια δημόσια ή ιδιωτική υπηρεσία[2]. Υπήρχαν βέβαια οι ασύρματοι που θεωρούνται μία αρχέγονη μορφή του, και πολλά πυροσβεστικά πλοία και ρυμουλκά είχαν τέτοιους στη Νέα Υόρκη το 1930. Ωστόσο, η Αμερική που είχε μείνει ανέπαφη μετά τον πόλεμο και με τις δύο μεγάλες ιδιωτικές εταιρείες τις, την Bell Telephone Laboratories και την Motorola corporation, ξεκίνησε μία έρευνα και ανάπτυξη στη ράδιοτηλεπικοινωνία. Αυτή η έρευνα κυμάνθηκε γύρω από τα δίκτυα κινητής τηλεπικοινωνίας που σιγά σιγά εξελίχθηκαν και σε άλλες χώρες.

Το πρώτο κινητό τηλέφωνο παγκοσμίως, κυκλοφόρησε από την Motorola το 1973. Ήταν γνωστό ως DynaTAC και ήταν ένα άκρως μεγάλο τηλέφωνο που ζύγιζε περίπου ένα κιλό και είχε μπαταρία που διαρκούσε από 30 έως 60 λεπτά [3]. Το κόστος του ήταν 3995$, ένα τεράστιο κόστος που πρόσφερε μόνο την δυνατότητα να μιλάς καθώς κινείσαι, και η ζήτησή τους παρέμεινε χαμηλή μέχρι το 1990.

Αργότερα, όταν ήρθε η δεύτερη γενιά των ψηφιακών δικτύων και η αξιοπιστία των κλήσεων βελτιώθηκε, βελτιώθηκαν και τα κινητά τηλέφωνα. Το μέγεθός τους ελαττώθηκε, οι επιδόσεις τους αυξήθηκαν και η τιμή τους έπεσε. Αυτό συνέβαλε στο να γίνουν δημοφιλέστερα και, καθώς η τεχνολογία εξελισσόταν, προστέθηκαν επιπλέον χαρακτηριστικά πέρα από την απλή τηλεφωνική κλήση όπως η δυνατότητα αποστολής γραπτών μηνυμάτων, παιχνίδια, ημερολόγια, αριθμομηχανές κ.α.

## 2.2 Ιστορική αναδρομή

*Η ιστορία του λειτουργικού και οι ανάγκες που εξυπηρετεί.*

Το πρώτο Smartphone που κυκλοφόρησε ήταν το Simon της IBM το 1993, το οποίο αξιοποιούσε την τεχνολογία touchscreen και μπορούσε να στείλει και να λάβει fax και email [4].

Στις 15 Αυγούστου του 1996, η Nokia παρουσίασε το Communicator, ένα κινητό τηλέφωνο GSM και υπολογιστή χειρός. Είχε ένα πληκτρολόγιο QWERTY, ενσωματωμένο πρόγραμμα επεξεργασίας λέξεων και ημερολόγιο. Εκτός από την αποστολή και λήψη φαξ, με το 9000 θα μπορούσατε να ελέγξετε τα e-mail σας ενώ είχε επίσης περιορισμένη πρόσβαση στο internet [2].

Τον Οκτώβριο του 2000 η Sharp λάνσαρε το πρώτο κινητό τηλέφωνο με ενσωματωμένη κάμερα. Το J-SH04 κινητό τηλέφωνο παρείχε στους χρήστες μία ψηφιακή κάμερα και επέτρεπε να στέλνουν και να λαμβάνουν εικόνες μέσο email.

Οι δυνατότητές τους ωστόσο ήταν έλαχιστες σε σχέση με τα σημερινά μοντέλα που διαθετουν οθόνες πολυεπαφής με δυνατότητα έλεγχου της συσκευής με χειρονομίες, ενσωματωμένες εφαρμογές, υπηρεσίες Web, συστήματα πλοήγησης, και λειτουργικά συστήματα όπως το Android που αναπτύχθηκε από την Android Inc, και αγοράστηκε από την Google το 2005 [8].

Τα σημερινά κινητά τηλέφωνα προσεγγίζουν περισσότερο την έννοια ενός προσωπικού υπολογιστή χειρός και ονομάζονται πλέον έξυπνα τηλέφωνα (smartphones). Τα smartphones είναι ένα κύριο εργαλείο επικοινωνίας και ως εκ τούτου οι άνθρωποι τα κουβαλάνε μαζί τους καθημερινά. Εξοπλίζουν τον χρήστη με πρόσβαση στο Internet, μουσική, εγγραφή και προβολή βίντεο, δυνατότητες πλοήγησης κ.α. Ακόμα θεωρούνται ως μια πύλη στη ζωή του καθενός αφού παρέχουν πρόσβαση σε emails, μέσα μαζικής δικτύωσης και σε ανταλλαγή μηνυμάτων [5].

Τα smartphones έχουν λειτουργίες όπως βίντεοκλήση, σύνδεση σε ασύρματα δίκτυα και στο διαδίκτυο, τρισδιάστατα γραφικά και πολυμέσα, επεξεργαστές πολλών πυρήνων, δυνατότητες λειτουργίας ως Wi-Fi Hotspots, δέκτες δορυφορικών σημάτων (GPS), αισθητήρες για επαφή της συσκευής με το περιβάλλον της και πολλές άλλες που αυξάνονται και βελτιώνονται συνεχώς.

Υπάρχουν επιταχυνσιόμετρα σε κάθε συσκευή μετά από την αρχική ιδέα να εισαχθούν για να ενισχύσουν τη διασύνδεση του χρήστη με την κάμερα. Χρησιμοποιούνται για να καθορίσουν αυτόματα τις συντεταγμένες στις οποίες ο χρήστης κρατά το τηλέφωνο και χρησιμοποιούν την πληροφορία αυτή ώστε αυτόματα να αναπροσανατολίσουν την οθόνη μεταξύ οριζόντιου και κατακόρυφου προσανατολισμού ή προσανατολίζουν σωστά φωτογραφίες που τραβήχτηκαν κατά τη διάρκεια της προβολής στο τηλέφωνο[6].

Όλα αυτά τα τεχνικά χαρακτηριστικά συνεισφέρουν στην ανάπτυξη ριζοσπαστικών εφαρμογών, όπως το Facebook που μπορούν να κοινοποιεί και καταγράφει τις δραστηριότητες ενός ατόμου σε πραγματικό χρόνο. Με το επιταχυνσιόμετρο είναι δυνατή η ανάπτυξη εφαρμογών που καταγράφουν τις φυσικές κινήσεις των χρηστών που φέρουν τα τηλέφωνα. Αναγνωρίζουν για παράδειγμα αν κάποιος τρέχει ή περπατά. Το GPS μπορεί να χρησιμοποιηθεί για την ανίχνευση της θέση ενός ατόμου και για την δημιουργία υπηρεσιών που βασίζονται σε αυτή.

Με δεδομένο επίσης τα app stores που είναι διαθέσιμα, οι εφαρμογές είναι διαθέσιμες σε ένα τεράστιο κοινό και έτσι επιτρέπεται η συλλογή και η ανάλυση δεδομένων πέρα από κάθε δυνατή κλίμακα στο παρελθόν.

Σίγουρα όμως, ένα από τα πιό ουσιώδη μέρη που απαρτίζουν τις κινητές συσκευές του σήμερα είναι το λειτουργικό τους σύστημα. Το λειτουργικό σύστημα είναι κομμάτι κάθε κινητού τηλεφώνου. Είναι υπεύθυνο για την διαχείριση των πόρων(επεξεργαστή, μνήμη κτλ.) της συσκευής, ελέγχει την εκτέλεση όλων των εφαρμογών στο σύστημα και χάρης αυτού το κινητό επικοινωνεί με το περιβάλλον αλλά και με τον χρήστη.

## 2.3 Διαφορές μεταξύ εκδόσεων android

- Android 1.6 Donut

Το Android 1.6 (Donut) περιλαμβάνει ένα επανασχεδιασμένο πλαίσιο αναζήτησης που παρέχει μια γρήγορη και αποτελεσματική για τους χρήστες αναζήτηση σε πολλαπλές πηγές, όπως σελιδοδείκτες του προγράμματος περιήγησης, επαφές του χρήστη που είναι αποθηκευμένες στη συσκευή, αλλά και αναζητήσεις στο διαδίκτυο. Και όλα αυτά κατευθείαν από την αρχική οθόνη. Η αποτελεσματικότητα της αναζήτησης βασίζεται στο ειδικά διαμορφωμένο σύστημα που μαθαίνει σύμφωνα με τα clicks των προηγούμενων αναζητήσεων ποια αποτελέσματα σχετίζονται περισσότερο σε αυτή και έπειτα τα παρουσιάζει στον χρήστη. Άλλο ένα στοιχείο που σχετίζεται με την απελευθέρωση του Android 1.6 Donut ήταν και η βελτίωση του Google Play που αύξησε την συνολική εμπειρία του χρήστη βοηθώντας τον να ανακαλύψει πιο εύκολα αυτό που έψαχνε. Πάλι μέσο στης αρχικής οθόνης δόθηκε η δυνατότητα στους χρήστες να επιλέγουν ευκολότερα τις εφαρμογές που τους ενδιέφεραν χωρίζοντας αυτές σε κατηγορίες οι όποιες περιείχαν τους τίτλους των εφαρμογών. Κάθε κατηγορία εμφάνιζε τους πιο δημοφιλής τίτλους επί πληρωμή, τους πιο δημοφιλείς δωρεάν τίτλους και αυτούς που απλά υπήρχαν. Επίσης για κάθε τίτλο ήταν δυνατό να δει ο χρήστης στιγμιότυπα της εφαρμογής όπως ακόμα και σχόλια που την περιέγραφαν από όλους χρήστες.

- Android 2.0 Eclair

Το Android 2.0 (Eclair) περιλάμβανε και αυτό αρκετές νέες ενημερώσεις καθιστώντας τους προγραμματιστές δυνατούς ώστε να μπορούν να δημιουργήσουν εφαρμογές που συγχρονίζονται με συμπληρωματικές πηγές δεδομένων. Το λογισμικό είναι σε θέση να δεχτεί πολλαπλούς λογαριασμούς ηλεκτρονικού ταχυδρομείου έτσι ώστε να παρέχει την παρακολούθησή και την ενημέρωση των σε πραγματικό χρόνο. Ένα επιπλέον χαρακτηριστικό της πλατφορμας ονομαζεται Quick Contact. Είναι μία μέθοδος που επιτρέπει την γρήγορη επικοινωνία με άλλους χρήστες. Πατώντας πάνω στην εικόνα μιας τηλεφωνικής επαφής εμφανίζεται ένα παραθυράκι με διάφορες διαθέσιμες επιλογές επικοινωνίας όπως: ηλεκτρονικό ταχυδρομείο, τηλεφωνική κλήση, γραπτό μήνυμα κ.α. Ωστόσο και άλλες εφαρμογές όπως το ηλεκτρονικό ταχυδρομείο, τα Μηνύματα, το Ημερολόγιο μπορούν να χρησιμοποιήσουν την μέθοδο αυτή εμφανίζοντας το παραθυράκι που αναφέραμε, όταν μια φωτογραφία επαφής πατηθεί. Επιπλέον χαρακτηριστικά που εξελιχθήκαν ήταν: η κάμερα, εξοπλίστηκε με ψηφιακό ζουμ και παρείχε υποστήριξη για ενσωματωμένο flash, και το πληκτρολόγιο, με ανανεωμένο

πλαίσιο εργασιών που παρείχε την δυνατότητα στον χρήστη να γράφει και με τα δύο του χέρια διαβεβαιώνοντας τον ότι αυτό που πληκτρολογούσε δεν θα χανόταν. Ο περιηγητής βελτιώθηκε επίσης παρέχοντας υποστήριξη σε HTML5 που συνοδεύεται με ικανότητες χρήσης βάσεων δεδομένων, από τη μεριά του πελάτη, χρησιμοποιώντας SQL, παρουσίαση βίντεο σε πλήρη οθόνη, και υποστήριξη χρήσης μνήμης cache για εργασία χωρίς σύνδεση.

- Android 2.2 Froyo

Μια από τις πρώτες εκδόσεις είναι και η Android 2.2 (Froyo) η οποία εκδόθηκε το Μάιο του 2010. Εισήγαγε την τεχνολογία Android Cloud to Device Messaging(C2DM). Η υπολογιστική νέφους(cloud computing) επιτρέπει την χρησιμοποίηση λογισμικού και δεδομένων που είναι αποθηκευμένα στο ”σύννεφο”- δηλαδή προσπελαύνονται απομακρυσμένοι υπολογιστές (ή διακομιστές) μέσω του Internet και ο χρήστης έχει πρόσβαση σε δεδομένα κατά απαίτησή του. Αύτη η δυνατότητα επεκτείνει τους πόρους των Android συσκευών δίνοντας αποθηκευτικό χώρο στους χρήστες εκτός αυτών. Ο χρήστης εξοικονομεί έτσι χρήματα αφού δεν είναι αναγκασμένος να αγοράσει ακριβό υλικό για να διασφαλίσει αρκετό χώρο αποθήκευσης. Ωστόσο η τεχνολογία C2DM επίσημα καταργήθηκε στις 26 Ιουνίου 2012 και θα κλείσει εντελώς από τις 30 Ιουλίου 2015. Οι προγραμματιστές C2DM ενθαρρύνονται να χρησιμοποιήσουν το Google Cloud Messaging (GCM) ανταυτού. Αλλά στην πραγματικότητα το C2DM δεν ήταν η μόνη αιτία εξοικονόμησης υπολογιστικών πόρων. Το Android 2.2 εξοπλίστηκε με υπολογιστική ισχύ διπλάσια - έως και πενταπλάσια σε σχέση με το Android 2.1. Η απόδοση της μνήμης του βελτιώθηκε κατά είκοσι φορές και αυτό οδηγεί, με την σειρά του, σε ταχύτερη εναλλαγή εφαρμογών και ομαλότερη διαχείριση τους σε συσκευές με περιορισμένη μνήμη. Το Android Android 2.2 Froyo παρείχε επιπλέον ένα χαρακτηριστικό που μετέτρεπε τις συσκευές σε φορητά Hotspots. Ορισμένες συσκευές, όπως το Nexus One μπορούσαν να μετατραπούν σε ένα φορητό Wi-Fi hotspot που παρείχε δυνατότητα σε έως και 8 συσκευές να συνδεθούν με αυτό. Η συσκευή με το λογισμικό Android χρησιμοποιούσε το κινητό τηλέφωνο ως μία σύνδεση 3G για φορητούς υπολογιστές Windows ή Linux. Έπειτα η σύνδεση μοιραζότανε μεταξύ των συσκευών.

- Android 2.3 Gingerbread

Ακολουθεί η Android 2.3 (Gingerbread) που δόθηκε αυτή το 2010 και πρόσθεσε περισσότερες βελτιώσεις χρήστη που στόχευαν στη απλότητα και την ταχύτητα, όπως ένα ανασχεδιασμένο πληκτρολόγιο, βελτιωμένες δυνατότητες πλοήγησης, αυξημένη αποδοτικότητα ισχύος και πολλές άλλες. Η διαχείριση του χρήστη εξευγενίστηκε με πολλούς τρόπους σε όλο το σύστημα, πράγμα που καθιστά ευκολότερη την προσαρμογή του χρήστη, και του παρέχει ευελιξία κινήσεων. Ένα απλοποιημένο οπτικό θέμα ζωντάνευσε την μπάρα κοινοποίησης, τα μενού, και άλλα μέρη του UI. Ακόμα, αλλαγές στο μενού και τις ρυθμίσεις κατέστησαν στον χρήστη ευκολότερη την πλοήγηση, τον έλεγχο των δυνατοτήτων του συστήματος και της συσκευής[7]. Το πληκτρολόγιο του Android ανασχεδιάστηκε, τα πλήκτρα πήραν διαφορετικό σχήμα και επανατοποθετήθηκαν, έτσι ώστε να μπορούν να πατηθούν πιο εύκολα. Η έκδοση αυτή προσθέτει υποστήριξη και σε πολλούς νέους τύπους αισθητήρων, συμπεριλαμβανομένου του γυροσκοπίου, του ανιχνευτή περιστροφής, του ανιχνευτή γραμμικής επιτάχυνσης, του ανιχνευτή βαρύτητας, και αισθητήρες βαρόμετρα. Εφαρμογές, πλέον, μπορούν να χρησιμοποιήσουν αυτούς τους αισθητήρες σε συνδυασμό με τυχόν άλλους αισθητήρες που

υπάρχουν στη συσκευή, για να παρακολουθούν τρισδιάστατα την κίνηση της συσκευής και την αλλαγή του προσανατολισμού της με υψηλή ακρίβεια. Αυτό έχει φέρει στην αγορά και πολλές εφαρμογές παιχνίδια που μπορούν και χρησιμοποιούν αυτές τις ενδείξεις για να αναγνωρίζουν πολύπλοκες χειρονομίες και τις κινήσεις του χρήστη, όπως η κλίση, η περιστροφή, η ώθηση, έτσι ώστε για παράδειγμα να προσομοιώνουν την οδήγηση ενός αμαξιού. Αλλά ένα από τα κυριότερα νέα χαρακτηριστικά της έκδοσης που αξίζει να αναφερθούμε ήταν οι επικοινωνίες κοντινού επιπέδου NFC (Near-Field communication). Οι επικοινωνίες κοντινού επιπέδου NFC είναι ένα πρότυπο ασύρματων επικοινωνιών που επιτρέπει την επικοινωνία ανάμεσα σε 2 συσκευές οι οποίες απέχουν μεταξύ τους λίγα εκατοστά του μέτρου. Το NFC μπορεί να χρησιμοποιηθεί για πληρωμές και για πολλές άλλες εργασίες πολλά υποσχόμενες αλλά και με ίσως πρωτοφανείς υπηρεσίες όπως το Android Pay.

- Android 3.0-3.2 Honeycomb

Η έκδοση Android 3.0 - 3.2 (Honeycomb) ήταν η αμέσως επόμενη έκδοση. Η έκδοση αυτή περιλαμβάνει βελτιώσεις στην διεπαφή χρήστη, ειδικά για συσκευές με μεγάλες οθόνες όπως τα tablets. Μία από αυτές είναι και τα fragments τα οποία περιγράφουν τμήματα διεπαφής χρήστη μίας εφαρμογής και μπορούν να συνδυάζονται σε μία οθόνη ή και να χρησιμοποιούνται σε πολλές οθόνες. Το Android 3.0 είναι κτισμένο από τα πράγματα που οι άνθρωποι αγαπούν περισσότερο στο Android - εξευγενισμένα multitasking, πολλές κοινοποιήσεις, προσαρμογή αρχικής οθόνης, widgets, και πολλά άλλα[7]. Πέντε κύριες οθόνες υπάρχουν έτσι ώστε ο χρήστης να έχει πρόσβαση σε ό,τι θέλει. Οι οθόνες αυτές παραμετροποιούνται από τους χρήστες που έχουν την δυνατότητα να προσθέσουν το δικό τους φόντο εργασίας και συντομεύσεις των εφαρμογών που θέλουν. Επίσης πλέον κάθε αρχική οθόνη παρέχει μια συντόμευση που ανοίγει όλες τις εγκαταστημένες εφαρμογές. Το πληκτρολόγιο επανασχεδιάστηκε, και σε αυτή την έκδοση, έτσι ώστε να προσαρμόζεται και στις ανάγκες μίας μεγάλης οθόνης και να είναι φιλικό με τον χρήστη. Νέες επιλογές συνδεσιμότητας εμφανίστηκαν όπως ένα πρωτόκολλο(MTP/PTP) για μεταφορά εικόνων και πολυμέσων που μέσο USB έδινε την δυνατότητα στον χρήστη να μεταφέρει τις εικόνες του και τα πολυμέσα του από τον υπολογιστή του στο κινητό του και αντίστροφα. Ένα επιπλέον χαρακτηριστικό που προστέθηκε σε αυτόν το τομέα ήταν και οι συνδέσεις Bluetooth που υποστηρίζουν την σύνδεση με εξωτερικές συσκευές και παρέχουν ενημερώσεις για αυτές, μέσο της συσκευής Android στον χρήστη, λόγου χάρη το επίπεδο της μπαταρίας των Bluetooth ακουστικών.

- Android 4.0 Ice Cream Sandwich

Η έκδοση Android 4.0 - 4.0.4 (Ice Cream Sandwich) εκδόθηκε το 2011 και συνενώνει την έκδοση Android 2.3 και 3.0 σε ένα λειτουργικό σύστημα που μπορεί να χρησιμοποιηθεί σε όλες τι συσκευές Android. Ωστόσο, η δύναμη του Android 4.0 στηρίζεται στο Multitasking**.** Οι χρήστες είχαν πλέον την δυνατότητα να πηδούν από την μία εφαρμογή στην άλλη μέσο μιας λίστας που εμφανίζει τις ενεργές εφαρμογές. Η έκδοση είχε επίσης πολλά νέα χαρακτηριστικά διεπαφής χρήστη. Παρέχει ένα “δίσκο” στον χρήστη, τοποθετημένο στην αρχική οθόνη έτσι ώστε να καθιστά δυνατή την τοποθέτηση φακέλων και εφαρμογών μέσα σε αυτόν. Κατά κόρον οι αρχικές οθόνες στο Android 4.0 σχεδιάστηκαν έτσι ώστε να παρέχουν ένα πλούσιο περιεχόμενο και να είναι απόλυτα προσαρμόσιμες. Οι χρήστες μπορούν να ενσωματώσουν στις αρχικές τους οθόνες εφαρμογές μέσω διαδραστικών widgets. Τα Widgets επιτρέπουν στους χρήστες να

ελέγχουν το ηλεκτρονικό ταχυδρομείο τους, να ξεφυλλίζουν ένα ημερολόγιο, να ακούνε μουσική, και πολλά άλλα. Τα widgets χρησιμοποιούνται κυρίως για εφαρμογές που ονομάζονται “ζωντανές εφαρμογές“(live applications) οι οποίες είναι ενεργές πάντα χωρίς κάποιος να τις ανοίξει. Ένα χαρακτηριστικό παράδειγμα είναι και η εφαρμογή του καιρού. Οι χρήστες μπορούν να επεκτείνουν αυτά τα παράθυρα για να δείξουν περισσότερο περιεχόμενο ή να τα συρρικνώσουν για να εξοικονομήσουν χώρο.

Η συγκεκριμένη έκδοση πρόσθεσε επίσης και αρκετές API και αρκετά χαρακτηριστικά που χαρακτηρίζονται ως καινοτομίες.

★ Android Beam που χρησιμοποιεί την NFC σύνδεση και επιτρέπει να ακουμπάτε 2 συσκευές Android, για να μπορούν να μοιραστούν περιεχόμενο όπως αγαπημένες εφαρμογές, επαφές, μουσική, βίντεο και σχεδόν τα πάντα..

★ Wi-Fi Direct που επιτρέπει την σύνδεση πολλαπλών συσκευών Android και την επικοινωνία τους σε μεγάλες αποστάσεις σε σχέση με τη χρήση Bluetooth.

★ API για διευκόλυνση προσπέλασης για χρήστες με προβλήματα (π.χ. άτομα με προβλήματα όρασης)

★ Android@Home framework ένα πλαίσιο εργασίας για δημιουργία εφαρμογών που ελέγχουν οικιακές συσκευές.

★ Face Unlock που εισάγει σε μια εντελώς νέα προσέγγιση στην ασφάλεια μιας συσκευής, καθιστώντας τη συσκευή του κάθε ατόμου, ακόμη πιο προσωπική. Δίνει μία νέα επιλογή κλειδώματος οθόνης που επιτρέπει στους χρήστες να ξεκλειδώνουν την συσκευή τους με την ανίχνευση του προσώπου τους. Χρησιμοποιεί τη τεχνολογία state-of-the-art αναγνώρισης προσώπου.

- Android 4.1- 4.3 Jelly Bean

Στην συνέχεια η επόμενη έκδοση Android 4.1- 4.3 Jelly Bean που εκδόθηκε το 2012 περιλαμβάνει βελτιωμένη ασφάλεια αλλά και επιλογές για προγραμματιστές, όπως αρκετά νέα χαρακτηριστικά ανίχνευσης και αποσφαλμάτωσης. Παρέχει τη λειτουργία που συμπληρώνει αυτόματα αριθμούς και γράμματα όταν αρχίζεις να πληκτρολογείς. Επίσης παρέχει ανίχνευση τοποθεσίας μέσω σύνδεσης Wi-Fi, έξυπνη υποστήριξη Bluetooth γνωστή και ως Bluetooth Low-Energy αλλά και σύνδεση Bluetooth AVRPC 1.3 που μπορείς για παράδειγμα να εμφανίσεις ονόματα τραγουδιών στο στερεοφωνικό του αυτοκινήτου.

Μερικές ακόμα διεπαφές εφαρμογών που προστέθηκαν στην έκδοση Android 4.3 Jelly Bean είναι:

★ Widget στην οθόνη κλειδώματος που εμφανίζονται στην οθόνη του χρήστη όταν η συσκευή είναι κλειδωμένη.

★ Photo Sphere το οποίο εκμεταλλεύεται τα χαρακτηριστικά των πανοραμικών φωτογραφιών και επιτρέπει στους χρήστες να τραβάνε φωτογραφίες 360 μοιρών.

★ Υποστήριξη γλώσσας που παρέχει νέα χαρακτηριστικά τα οποία βοηθούν τις εφαρμογές να υποστηρίζονται σε όλον τον κόσμο.

- Android 4.4 Kit-Kat

Ακλουθεί το Android 4.4 Kit-Kat, που εκδόθηκε τον Οκτώβριο του 2013 και περιλαμβάνει αρκετές βελτιώσεις που καθιστούν δυνατή την εκτέλεση του λειτουργικού συστήματος σε όλες τις συσκευές Android συμπεριλαμβανομένων και των παλαιότερων συσκευών με περιορισμούς στη μνήμη [8]. Η δυνατότητα που δίνεται σε χρήστες να κάνουν ενημέρωση σε Kit Kat θα μειώσει τον "τεμαχισμό" των εκδόσεων του Android στην αγορά. Το Android 4.4 εκτοξεύει την απόδοση του συστήματος στο πιο υψηλό επίπεδο όλων των εποχών με τη βελτιστοποίηση της μνήμης και τη βελτίωση της αφής σας, έτσι ώστε να ανταποκρίνεται ταχύτερα και με μεγαλύτερη ακρίβεια από ποτέ. Αυτό σημαίνει ότι μπορείτε να ακούσετε μουσική ενώ περιηγείστε στον ιστό, χωρίς κανένα πρόβλημα[7]. Ακόμη μία δυνατότητα που παρέχεται στον χρήστη είναι ότι τώρα μπορεί να εκτυπώσει φωτογραφίες, έγγραφα και ιστοσελίδες από το κινητό ή το tablet του. Ο χρήστης μπορεί να εκτυπώσει έγγραφα σε οποιονδήποτε εκτυπωτή που συνδέεται με το Google Cloud Print, σε εκτυπωτές HP ePrint, και σε άλλους εκτυπωτές που έχουν εφαρμογές στο Google Play Store. Αλλά αυτό που αξίζει σίγουρα να σημειώσουμε είναι ότι η συγκεκριμένη έκδοση υποστηρίζει ασφαλές συναλλαγές βασισμένες σε NFC μέσω Host Card Emulation (HCE). Το HCE εξομοιώνει ειδικές έξυπνες κάρτες βασισμένες στο ISO/IEC 7816 που χρησιμοποιούν ένα συγκεκριμένο πρωτόκολλο για συναλλαγές. Συσκευές που υποστηρίζουν την τεχνολογία NFC θα περιλαμβάνουν την λειτουργία Tap & Pay για εύκολες πληρωμές με χρήση HCE.

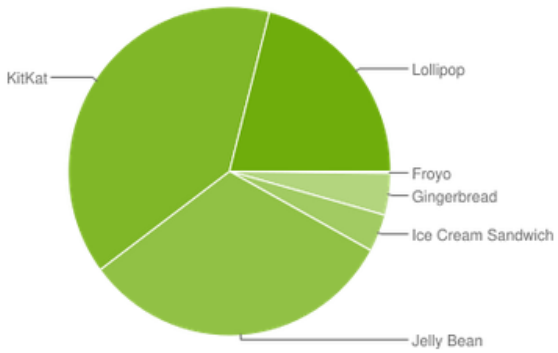
- Android 5 Lollipop

Και καταλήγουμε στην τελευταία έκδοση Android Lollipop. Σας καλώς ορίζουμε στην έκδοση Android Lollipop. Την μεγαλύτερη και πιο φιλόδοξη απελευθέρωση για το Android! Έχει γραφικά υψηλής απόδοσης, προηγμένη συνδεσιμότητα, νέους τύπους αισθητήρων κ.α. Με το Lollipop, το tablet σας παρουσιάζει τα εισερχόμενά σας μαζί με το μήνυμα που έχετε ανοίξει, ενώ το ρολόι σας δείχνει νέα μηνύματα που φτάνουν. Αυτή η έκδοση είναι γεμάτη με νέα χαρακτηριστικά για τους χρήστες και χιλιάδες νέα API για τους προγραμματιστές. Επεκτείνει το Android ακόμη περισσότερο, από κινητά τηλέφωνα, ταμπλέτες, και wearables, σε τηλεοράσεις και αυτοκίνητα[9].

Ακόμα η συγκεκριμένη έκδοση εξοπλίζει το χρήστη με πολλά νέα και χρήσιμα χαρακτηριστικά. Ένα από αυτά είναι η εξοικονόμηση μπαταρίας που παρέχει στον χρήστη ενημέρωση για την διάρκεια που απομένει μέχρι η μπαταρία να χρειαστεί φόρτιση, τον ενημερώνει για το υπολειπόμενο χρόνο που χρειάζεται μέχρι να φορτίσει, και του δίνει και την δυνατότητα να πάρει παράταση 90 λεπτών για χρήση της συσκευής. Επίσης το Lollipop παρέχει επιπλέον ένα χαρακτηριστικό που επιτρέπει πολλαπλούς χρήστες στην ίδια συσκευή. Με την προϋπόθεση ότι οι συσκευές στις οποίες αναφερόμαστε τρέχουν την Android Lollipop έκδοση, κάθε χρήστης μπορεί να συνδέεται από μια άλλη συσκευή στο δικό του τηλέφωνο και μπορεί για παράδειγμα

να πραγματοποιεί κλήσεις. Το προαναφερθέν χαρακτηριστικό παρέχει και την δυνατότητα της σύνδεσης χρήστη ως επισκέπτη, με αποτέλεσμα αν κάποιος θέλει να μοιραστεί την συσκευή του να μην μοιράζει το υλικό του και τις προσωπικές του πληροφορίες.

| Version | Codename | API | Distribution |
|---|---|---|---|
| 2.2 | Froyo | 8 | 0.2% |
| 2.3.3 - 2.3.7 | Gingerbread | 10 | 4.1% |
| 4.0.3 - 4.0.4 | Ice Cream Sandwich | 15 | 3.7% |
| 4.1.x | Jelly Bean | 16 | 12.1% |
| 4.2.x | | 17 | 15.2% |
| 4.3 | | 18 | 4.5% |
| 4.4 | KitKat | 19 | 39.2% |
| 5.0 | Lollipop | 21 | 15.9% |
| 5.1 | | 22 | 5.1% |




*Data collected during a 7-day period ending on September 7, 2015.*
*Any versions with less than 0.1% distribution are not shown.*

> **Note:** This data is gathered from the new Google Play Store app, which supports Android 2.2 and above, so devices running older versions are not included. However, in August, 2013, versions older than Android 2.2 accounted for about 1% of devices that *checked in* to Google servers (not those that actually visited Google Play Store).

*Εικόνα 8 : Στατιστικά δεδομένα εκδόσεων Android.*

## 2.4 Η ιστορία του Iphone

Η Apple ενώ το 1984 επαναστάτησε με τα Macintosh στους προσωπικούς υπολογιστές το 1990, η εταιρεία, άρχισε να μην τα πάει καλά. Ωστόσο, αυτό άλλαξε με την επιστροφή του Steve Jobs. Υπό την επίβλεψή του η Apple έκανε κάποιες καινοτομίες μια εκ των οποίων ήταν το iPod, ένα αριστούργημα φορητού σχεδιασμού. Επεσήμανε ένα απλό interface, που έδινε στον χρήστη πρόσβαση σε χιλιάδες μουσικά κομμάτια και την δυνατότητα να τα κουβαλάει μαζί του όλη την ώρα[10]. Το 2005, ο διευθύνων σύμβουλος της Apple Steve Jobs συνέλαβε την ιδέα να χρησιμοποιήσει μια οθόνη αφής multi-touch για να αλληλεπιδράσει με έναν υπολογιστή με έναν τρόπο κατά τον οποίο θα μπορούσε να πληκτρολογήσει απευθείας στην οθόνη, αφαιρώντας ουσιαστικά το φυσικό πληκτρολόγιο και το ποντίκι. Το 2006 άρχισαν φήμες και εικασίες για την επόμενη καινοτομία της εταιρείας: ένα κινητό τηλέφωνο όπως το iPod που θα ονομαζόταν iPhone. Αυτό ήταν που χάρισε στο αρχείο της Apple μια τεχνολογική καινοτομία και ένα εξαιρετικό σχεδιασμό χρήστη του εικοστού πρώτου αιώνα. Ήταν κάτι καινούργιο και διαφορετικό, ένα έξυπνο τηλέφωνο φιλικό προς τον χρήστη. Η εταιρεία δημιούργησε τελικά τη συσκευή κατά τη διάρκεια μιας μυστικοπαθείς και άνευ προηγουμένου συνεργασία με την AT & T. Το κόστος ανάπτυξης της συνεργασίας εκτιμάται ότι ήταν $ 150 εκατομμύρια σε μια περίοδο τριάντα μηνών. Με την έκδοσή του το 2007 η ανταπόκριση του κόσμου ήταν τεράστια, το ίδιο και ο ενθουσιασμός του, όπως και οι πωλήσεις που επηρεάστηκαν από αυτόν τον ξέφρενο ενθουσιασμό. Το ενδιαφέρον συνέχισε να αυξάνεται και οι πωλήσεις με την καινούργια έκδοση του iPhone 3G το 2008 υπολογίζεται ότι φτάσανε το ένα εκατομμύριο μέσα στις πρώτες τρείς ημέρες.

## 2.5 Διαφορές μεταξύ εκδόσεων Iphone

- Η πρώτη έκδοση το 2007 συμπεριλάμβανε 620MHz ARM CPU που ήταν underclocked για να βελτιώσουν την επίδοση της μπαταρίας και να μειώσουν την θερμότητα. Περιείχε 128MB δυναμικής RAM(DRAM) και από 4GB έως 16GB Flash memory. Το λογισμικο του δεν είχε κάποιο συγκεκριμένο όνομα, απλά ήταν γνωστό.

- Η δεύτερη 3G έκδοση το 2008 ήταν παρόμοια με τη πρώτη. Είχε τον ίδιο επεξεργαστή ARM, 128MB δυναμικής RAM(DRAM) και από 4GB έως 16GB Flash memory. Η κύρια διαφορά τους ήταν στο GPS και στην δικτύωση [10]. Αποτελεί την δεύτερη γενιά των iPhone και συμπεριλαμβάνει νέα χαρακτηριστικά όπως δικτύωση 3G και μηχανισμό πλοήγησης GPS. Η συσκευή είχε το λογισμικό iPhone OS 2.0 το οποίο παρουσίαζε για

πρώτη φορά το App Store - ένα κατάστημα για τις εφαρμογές που υπήρχαν. Το iPhone 3G διέθετε μια εσωτερική επαναφορτιζόμενη μπαταρία, ισχύος 1150 mAh που ήταν σε θέση να προσφέρει έως και έξι ώρες περιήγησης στο διαδίκτυο μέσω του Wi-Fi, ή πέντε ώρες μέσω 3G σύνδεσης, ή 25 ώρες αναπαραγωγής ήχου. Εναλλακτικά, μπορούσε να παρέχει 300 ώρες χρόνο αναμονής. Τέλος το iPhone 3G παρέμεινε στην αγορά και μετά τον ερχομό του διαδόχου του ως μια επιλογή χαμηλού κόστους και εξοπλίστηκε και με το iPhone OS 3.0.

- Το 2009 η Apple λάνσαρε το 3GS iPhone που είχε μεγαλύτερο χώρο αποθήκευσης από τα προηγούμενα iPhones και καλύτερη κάμερα. Επίσης την ίδια χρονολογία βγήκε και η έκδοση 3.0 του λογισμικου. Τα νέα χαρακτηριστικά του iPhone 3GS ήταν κυρίως εσωτερικές αλλαγές όσον αφορά την ταχύτητα. Για αυτό προστέθηκε και το ¨S¨ στην ονομασία του μοντέλου ώστε να δηλώσει την ταχύτητα “Speed”. Το iPhone 3GS ήταν 2 φορές ταχύτερο από τον προκάτοχό του. Αν και πέρα από τις αναβαθμίσεις, κυρίως όσον αφορά τις επιδόσεις, διάφορα χαρακτηριστικά του λογισμικού εισήχθησαν επίσης αποκλειστικά για το iPhone 3GS, όπως βιντεοκάμερα, Έλεγχος Voice και ψηφιακή πυξίδα. Όλα τα αποκλειστικά χαρακτηριστικά του ενσωματώθηκαν στο iPhone 4[11].

- Ένα χρόνο αργότερα ανακοινώθηκε από τον Steve Jobs το iPhone 4. Το μοντέλο αυτό είχε 2 κάμερες, μία στο μπροστά μέρος ώστε να εξυπηρετεί για παράδειγμα στην πραγματοποίηση Video κλήσεων, και μία στο πίσω μέρος της συσκευής. Ακόμα είχε αυτό που αποκαλούσε η Apple Retina display: ανάλυση 960 επί 640 pixels και αντίθεση 800 προς 1. Το ίδιο έτος ο Jobs ανακοίνωσε και το νέο όνομα του λειτουργικού συστήματος του iPhone: iOS[12]. Έτσι, το iPhone 4 συνδυάζοντας το iOS4 με το νέο εισαχθέντα A4 system-on-chip που αποτελείται από έναν ARM Cortex-A8 επεξεργαστή με ενσωματωμένη GPU PowerVR SGX535, ήταν σε θέση να υποστηρίζει multitasking. Επίσης το μοντέλο αυτό διέθετε, 512 MB DRAM ώστε να αυξήσει τις επιδόσεις του στο multitasking και γυροσκόπιο που συμπληρώνει το επιταχυνσιόμετρο, έναν αισθητήρα που έχει παρουσία από το αρχικό iPhone, το οποίο ανιχνεύει την επιτάχυνση της συσκευής, το κούνημα, τις δονήσεις, ή την πτώση, παρακολουθώντας την γραμμική επιτάχυνση κατά μήκος ενός από τους τρεις άξονες (X, Y και Z). Και έτσι συνδέοντας την πληροφορία από το επιταχυνσιόμετρο και το γυροσκόπιο, παρέχει λεπτομερείς και ακριβείς πληροφορίες σχετικά με την κίνηση της συσκευής στο χώρο επιτρέποντας τη συσκευή να αναγνωρίζει περίπου πόσο μακριά, γρήγορα, και σε ποια κατεύθυνση έχει μετακινηθεί στο διάστημα.

- Ακολούθησε η 5.0 έκδοση του iOS και το iPhone 4S με ένα A5 system-on-chip, που αποτελείται από έναν dual-core επεξεργαστή, τον ARM Cortex-A9 MPCore που μπορεί να προσαρμόζει δυναμικά τη συχνότητα ώστε να περιορίζει την κατανάλωση της μπαταρίας, και μία GPU(Graphics Processor Unit). Επίσης Το iPhone 4S διαθέτει 512 MB DDR2 RAM, μέγιστο μέγεθος διαθέσιμου χώρου αποθήκευσης 64 GB με 2 εναλλακτικές επιλογές, αυτές των 32GB και των 16GB, διέθετε ταχύμετρο και κάμερα 8Mp που μπορεί να τραβήξει βίντεο 1080p.

Το μοντέλο αυτό διαθέτει, επίσης, οθόνη πολυεπαφής που αναγνωρίζει πολλαπλές κινήσεις, multi-touch χειρονομίες, και βασίζει την διεπαφή του χρήστη του iOS στην έννοια του άμεσου χειρισμού. Το iPhone 4S κυκλοφόρησε με iOS 5, τα όποια κυκλοφόρησαν στις 12 Οκτωβρίου 2011 δύο ημέρες πριν από την απελευθέρωση της συσκευής. Το λειτουργικό σύστημα της Apple, εισήγαγε και χαρακτηριστικά, όπως το:

- ★ iCloud

- ★ iMessage ένα εξειδικευμένο πρόγραμμα ανταλλαγής μηνυμάτων που επιτρέπει την απεριόριστη ανταλλαγή γραπτών μηνυμάτων με άλλες Apple iOS 5 συσκευές. Η υπηρεσία αυτή υποστηρίζει την εισαγωγή κειμένου με φωνητική υπαγόρευση, αλλά και ένταξη πολυμέσων σε μηνύματα κειμένου, και παρέχει και αναφορές παράδοσης, ώστε να ενημερώνει τους χρήστες για την κατάσταση των μηνυμάτων.

- ★ Κέντρο ειδοποιήσεων(Notification Center), μια δυνατότητα του iOS, η οποία παρέχει μια επισκόπηση των καταχωρήσεων από τις εφαρμογές. Εμφανίζει ειδοποιήσεις μέχρι ο χρήστης να ολοκληρώσει μια σχετική δράση, αντί να απαιτεί άμεση επίλυση. Οι χρήστες μπορούν να επιλέξουν ποιες εφαρμογές εμφανίζονται στο Κέντρο ειδοποιήσεων, και πώς αντιμετωπίζονταν [13].

- ★ Υπενθυμίσεις(Reminders) ένα συστατικό του iOS, που επιτρέπει στους χρήστες να ρυθμίσουν ειδοποιήσεις για τον εαυτό τους, αλλά και λίστες από ειδοποιήσεις.

- ★ Siri που επιτρέπει στον χρήστη να δίνει φωνητικές εντολές στη συσκευή του. Το siri είναι ένα νέο αυτοματοποιημένο σύστημα φωνητικού ελέγχου που χρησιμοποιείτε για να δίνει την στους χρήστες να χρησιμοποιούν την συσκευή τους ακόμα και όταν τα χέρια τους είναι δεσμευμένα, για παράδειγμα όταν οδηγούν, αλλά για να διευκολύνει χρήστες με προβλήματα στην όραση και στο γράψιμο.

- Ακολούθησε το iPhone 5, με μεγαλύτερη οθόνη από τα 4 και 4s. Το iPhone 5 είχε σώμα φτιαγμένο από αλουμίνιο και έτσι ήταν λεπτότερο και ελαφρύτερο από τα προηγούμενα μοντέλα, είχε αναλογία οθόνης που σχεδόν έφτανε την 16:9, και A6 system - on - chip. Παρείχε υποστήριξη LTE, μία τεχνολογία αιχμής που χρησιμοποιείται για την ασύρματη επικοινωνία και δικτύωση των κινητών συσκευών, σε υψηλές ταχύτητες που φτάνουν κοντά σε αυτές των 4G, χρησιμοποιούσε την Sony φωτογραφική μηχανή των 8MP, που χρησιμοποίησε και το iPhone 4S και εξοπλίστηκε με τον A6 (SoC) που φέρεται να χρησιμοποιεί έναν 1,3 GHz, σχεδιασμένο από την Apple, ARMv7-A dual-core επεξεργαστή, που ονομάζεται Swift, και μία ολοκληρωμένη 266 MHz triple-core PowerVR SGX543MP3 μονάδα επεξεργασίας γραφικών (GPU)[14]. Η λειτουργική μνήμη(LPDDR2-1066 eDRAM) του iPhone 5 διπλασιάστηκε, από 512 MB σε 1 GB, οι διαθέσιμοι αποθηκευτικοί χώροι ορίστηκαν σε 16, 32 και 64 GB, και plug-in κάρτες

μνήμης δεν υποστηρίχτηκαν από τη συσκευή, όπως στο 4S[15]. Από την αρχή της κυκλοφορίας του, το iPhone 5 είχε iOS 6 λειτουργικό σύστημα, το οποίο κυκλοφόρησε στις 19 Σεπτέμβρη του 2012, και η κυκλοφορία της συσκευής, επίσημα σταμάτησε από την Apple στις 10 Σεπτεμβρίου 2013 με την ανακοίνωση των διαδόχων του.

- Τα iPhone 5S και iPhone 5C κυκλοφόρησαν στις 20 και στις 10 Σεπτεμβρίου του 2014. Στην αρχή της κυκλοφορίας τους και τα δύο μοντέλα ήταν εξοπλισμένα με το iOS7, ωστόσο τους παρέχεται η δυνατότητα για αναβάθμιση νεοτέρων εκδόσεων όπως αυτής των iOS 8. Ο σκελετός τους είναι φτιαγμένος από ανθρακόνιμα αλλά διαφέρουν σε μερικά σημεία μεταξύ τους. Το 5c χρησιμοποιεί τον A6 αντί του A7 που χρησιμοποίει το 5s, δεν περιλαμβάνει Touch ID, δεν υποστηρίζει OpenGL ES 3.0, την Burst iSight κάμερα ούτε λειτουργία εγγραφής βίντεο σε αργή κίνηση και, σε γενικές γραμμές, τα τεχνικά του χαρακτηριστικά είναι σχεδόν όμοια με αυτά του iPhone 5. Από την άλλη μεριά, το iPhone 5s χρησιμοποιεί τον A7 chip. Ο A7 είναι ένα σύστημα 64 - bit SoC που έχει σχεδιαστεί από την Apple, η οποία αναφέρει ότι είναι έως και δύο φορές πιο γρήγορος και έχει μέχρι και διπλάσια ισχύ γραφικών σε σύγκριση με τον προκάτοχό του, τον Apple A6. Αν και δεν είναι ο πρώτος 64-bit επεξεργαστής ARM, είναι ο πρώτος που ενσωματώθηκε σε εμπορικά smartphone και tablets[16]. Επίσης το 5s χρησιμοποίει έναν επεξεργαστή κίνησης, τον M7 co-processor. Αυτός ο συνεπεξεργαστής συλλέγει και επεξεργάζεται τα δεδομένα από την πυξίδα, το επιταχυνσιόμετρο και το γυροσκόπιο, και είναι σε θέση να πληροφορήσει τις εφαρμογές αν για παράδειγμα ο χρήστης οδηγεί, τρέχει περπατάει κ.ο.κ, αποφορτίζοντας την εργασία αυτή από το τσιπ A7 για βελτίωση της απόδοσή του. Τέλος το iPhone 5s περιλαμβάνει και τα χαρακτηριστικά που προαναφέραμε συν μερικών ακόμα:

  - ★ CoreMotion APIs

  - ★ Touch ID, τοποθετήστε το δάχτυλό σας στο κουμπί Home, και ακριβώς αμέσως το iPhone σας ξεκλειδώνει. Διαβάζει το δακτυλικό σας αποτύπωμα και ξέρει ποιος είστε, με επιπλέον δυνατότητα εγγραφής πολλαπλών δακτυλικών αποτυπωμάτων.

  - ★ Burst mode, που βοηθά να πάρετε φωτογραφίες εαυτού και ομαδικά πορτρέτα από τη συνεχή λήψη 10 φωτογραφίες ανά δευτερόλεπτο. Αναλύοντας κάθε λήψη σε πραγματικό χρόνο, συγκρίνει την ευκρίνεια, και ακόμη ανιχνεύει αν κάποιος έχει κλειστά τα μάτια του και στη συνέχεια, προτείνει μεμονωμένες φωτογραφίες ή σειρά φωτογραφιών που μπορεί να σας αρέσουν καλύτερα.

  - ★ iSight, το χαρακτηριστικό αυτό σας επιτρέπει να τραβάτε HD βίντεο στα 120 fps σε 720p ανάλυση σε slow-motion. Μπορείτε ακόμη και να καταγράψει και στη συνέχεια να επιλέξτε το τμήμα που θέλετε να επιβραδύνει[17].

- το iPhone 6 και το iPhone 6 Plus. Τα δύο νέα μοντέλα iPhone κυκλοφόρησαν στις 19 Σεπτεμβρίου, δύο ημέρες μετά τη δημοσιοποίηση, στις 17 Σεπτεμβρίου, του νέου iPhone λειτουργικού συστήματος, του iOS 8[18]. Τα 2 νέα μόντέλα διαθέτουν ακόμα

μεγαλύτερες οθόνες: 4.7 ίντσες για το iPhone 6 και 5.5 για το iPhone 6 Plus, και είναι ότι πιο καινούργιο κυκλοφορεί, μέχρι στιγμής, από την Apple. Τα μοντέλα που θεωρούνται ως ένα, περιλαμβάνουν έναν Apple A8 SoC που εκτοξεύει τις επιδόσεις της CPU και της GPU στα ύψη, έτσι ώστε να ανταποκρίνεται σε παιχνίδια με γραφικά υψηλής ευκρίνειας και σε αναπαραγωγή βίντεο υψηλών ταχυτήτων εναλλαγής καρέ. Διαθέτουν, αυτή την φορά, έναν M8 συνεπεξεργαστή που μετρά συνεχώς τα δεδομένα από το επιταχυνσιόμετρο, την πυξίδα, το γυροσκόπιο, και ένα καινούριο βαρόμετρο που έχει προστεθεί στην συσκευή, και τώρα κάνει ακόμα περισσότερα, μετρώντας τα βήματα σας και αλλαγές στην απόσταση, και στο υψόμετρό σας. Είναι ακόμη πιο λεπτά από τα προηγούμενα μοντέλα και πλέον έχει ένα πλήκτρο κλειδώματος στην δεξιά άκρη τους. Τέλος, το iPhone σε συνδυασμό με το iOS8 διαθέτει πολλά καινούργια και ανανεωμένα χαρακτηριστικά:

- ★ Messages , καθιστά εύκολη την σύλληψη ήχων - της φωνή σας, ενός τραγουδιού ή ενός μεγάλου γέλιου σας - και να τους κάνουν μέρος της συνομιλίας, στέλνει ένα βίντεο γρήγορα και εύκολα αλλά, και πολλά μαζί μόνο μιας. Παρέχει την δυνατότητα να μοιραστείτε την τοποθεσία σας στο κέντρο της συζήτησης αλλά και πολλές ακόμα.

- ★ Photos, έχει την δυνατότητα να βρίσκει τις φωτογραφίες που ψάχνετε εύκολα, και να επεξεργάζεται τις φωτογραφίες

- ★ εμφανίζει επιλογές λέξεων ή φράσεων που πιθανότατα θα πληκτρολογούσατε στη συνέχεια, με βάση τις προηγούμενες συνομιλίες που έχουν γίνει.

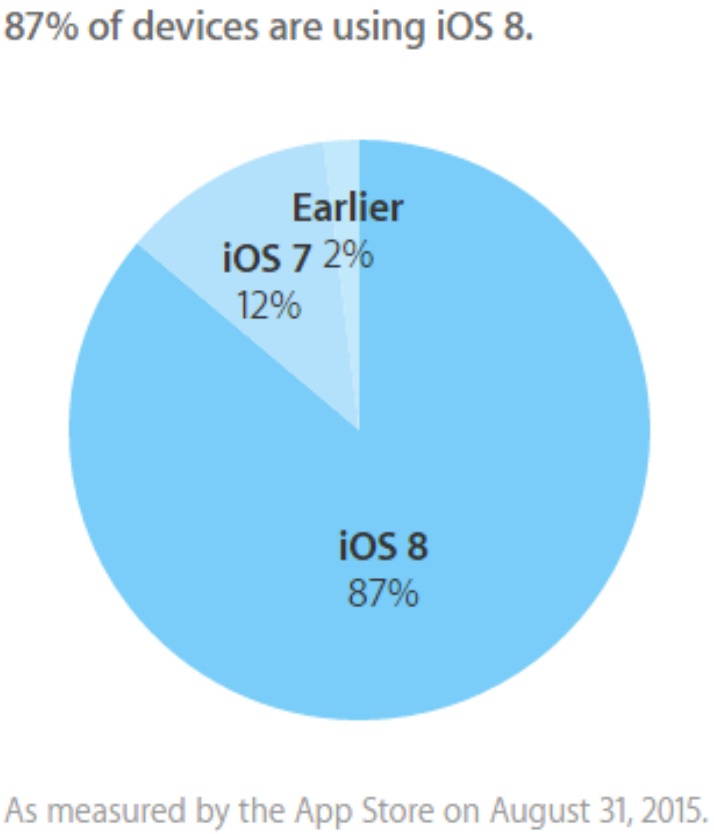
- ★ Family Sharing, που επιτρέπει σε - έως και έξι άτομα σε μιά οικογένεια, να μοιραστούν τις αγορές του άλλου, από το iTunes, iBooks, και το App Store.

- ★ iCloud Drive, μέσο αυτού μπορείτε να αποθηκεύετε με ασφάλεια όλες σας τις παρουσιάσεις, υπολογιστικά φύλλα, αρχεία PDF, εικόνες, και άλλα είδη εγγράφων στο iCloud. Έτσι είναι πάντα προσβάσιμα από το iPhone, το iPad, και σε όλες τις συσκευές Mac που διαθέτετε.

- ★ Health, ώστε να δίνει την δυνατότητα σε εφαρμογές να παρακολουθούν τον καρδιακό ρυθμό σας, και άλλες πληροφορίες για την υγεία και την φυσική σας κατάσταση.

- ★ Spotlight, “έξυπνα” δίνει προτάσεις από τις καταχωρήσεις της Wikipedia, κοντινά μέρη που σας ενδιαφέρουν, ειδήσεις που σας αφορούν, όταν ψάχνετε για μια επαφή ή εφαρμογές, ή όταν ψάχνετε για απαντήσεις.

- ★ iPhone, iPad, και Mac, παρέχει σύνδεση σε πολλαπλές συσκευές της Apple δίνοντας δυνατότητες όπως ιδιοκτήτες iphone να απαντάνε σε τηλεφωνικές κλήσεις μέσω του mac υπολογιστή τους.




*Εικόνα 9 : Στατιστικά εκδόσεων iOS*

## 2.6 Windows Phone


Ένα διαδεδομένο λειτουργικό σύστημα για smartphones είναι και το Windows Phone. Το δημιούργησε η microsoft ως συνέχεια των Windows Mobile που δεν κυκλοφορούν πια. Το Windows Phone χαρακτηρίζεται από το Metro User Interface. Το Metro στυλ είναι απλοϊκό αλλά ταυτόχρονα, και μοντέρνο στην εμφάνισή του , γι' αυτό κανονικά λέγετε και "Modern". Τετράγωνα ή μεγαλύτερα ορθογώνια πλακίδια (tiles) διακοσμούν την αρχική επιφάνεια του κινητού, μέσω των οποίων ο χρήστης μπορεί να μεταβεί σε διάφορες εφαρμογές του κινητού[19]. Τα πλακίδια - πλακάκια προσαρμόζονται με βάση το υλικό που ενδιαφέρει τον χρήστη και ενημερώνονται αυτόματα ώστε να δείχνουν αναρτήσεις, ενημερώσεις κατάστασης και tweets σε πραγματικό χρόνο.

- Τα Windows Phone 7 ήταν η πρώτη έκδοση του λειτουργικού που κυκλοφόρησε το 2010. Στην συνέχεια υπέστη αρκετές βελτιώσεις που τις διαδέχτηκαν ανανεωμένες εκδόσεις εκ των οποίων η τελευταία ήταν η έκδοση 7.8. Μία από αυτές που αξίζει να σημειώσουμε είναι έκδοση 7.5 Mango η οποία διέθετε έναν Windows Explorer περιηγητή, ίδιο με αυτόν που κυκλοφορούσε και για υπολογιστές, σύνδεση Window Live SkyDrive και επίσης υποστήριζε και multitasking. Αρκετά βασικά χαρακτηριστικά των Windows Phone 7 ήταν οργανωμένα σε "Hubs", τα οποία συνδυάζουν το περιεχόμενο που υπάρχει στο κινητό με αυτό του διαδικτύου μέσω της συνεργασίας Windows Phone με δημοφιλή κοινωνικά δίκτυα όπως το Facebook, το Windows Live, και το Twitter. Υπάρχουν αρκετά "Hubs" όπως το Pictures Hub και το People Hub που δείχνει τις επαφές που συγκεντρώνει από διάφορες πηγές, συμπεριλαμβανομένων των Windows Live, του Facebook και του Gmail. Έπειτα παρέδωσε τη σκυτάλη στο Windows Phone 8, ωστόσο οι συσκευές με το συγκεκριμένο λειτουργικό σύστημα δεν ήταν δυνατό να αναβαθμιστούν. Η συγκεκριμένη έκδοση παρείχε χαρακτηριστικά όπως:

    - ★ Music + Video Hub, που είναι μία συσκευή αναπαραγωγής στο Windows Phone σας. Μπορεί να χρησιμοποιηθεί για να δημιουργήσετε λίστες αναπαραγωγής, για να ακούσετε ραδιόφωνο FM, και να εγγραφείτε σε podcasts.

★ People Hub, κάτι περισσότερο από ένα βιβλίο διευθύνσεων στο τηλέφωνό σας. Σας κρατά συνδεδεμένους με τα κοινωνικά δίκτυα και σας βοηθά να μείνετε σε επαφή με τους ανθρώπους που σας ενδιαφέρουν περισσότερο.

★ Marketplace Hub, ένα εικονικό κατάστημα της Microsoft για εφαρμογές, παιχνίδια, μουσική, ταινίες, τηλεοπτικές εκπομπές, podcasts και για το τηλέφωνό σας.

★ Office hub, οργανώνει όλες τις εφαρμογές και τα έγγραφα του Microsoft Office. Το Microsoft Office Mobile παρέχει λειτουργικότητα μεταξύ του Windows Phone και της desktop έκδοσης του Microsoft Office. Word Mobile, Excel Mobile, PowerPoint Mobile, OneNote Mobile και το SharePoint Workspace Mobile επιτρέπουν στις περισσότερες μορφές αρχείων του Microsoft Office να είναι ορατές και να μπορούν άμεσα να επεξεργαστούν σε μια συσκευή Windows Phone. Το Microsoft Office μπορεί επίσης να ανοίγει αρχεία από το SkyDrive και το Office 365, καθώς και αρχεία που είναι αποθηκευμένα τοπικά στο τηλέφωνο [19].

- Τα Windows Phone 8 κυκλοφόρησαν το 2012, και μέσω αυτών η Microsoft επέκτεινε τις δυνατότητες του λογισμικού για να είναι συμβατό με συσκευές που έχουν μεγάλες οθόνες και πολυεπεξεργαστές. Επίσης παρείχε υποστήριξη σε NFC λειτουργίες, που μπορούν να χρησιμοποιηθούν κυρίως για ανταλλαγή περιεχόμενου και εκτέλεση των πληρωμών. Και τέλος εισήγαγε μια κατάσταση, η οποία λειτουργεί ως ένα είδος "λειτουργία επισκέπτη", και ονομάζεται Kids Corner. Ο χρήστης επιλέγει ποιες εφαρμογές και πια παιχνίδια θα εμφανίζονται. Όταν είναι ενεργοποιημένη η κατάσταση αυτή, προφυλάσσονται εφαρμογές και παιχνίδια που είναι εγκατεστημένα στη συσκευή και δεν είναι πλέον διαθέσιμα στον χρήστη επισκέπτη.

- Έπειτα κυκλοφόρησαν τον Απρίλιο του 2014 τα Windows Phone 8.1 με ανανεωμένες εφαρμογές όπως το Battery Saver, το Storage Sense που επιτρέπει στον χρήστη την μεταφορά αρχείων από τη μνήμη του κινητού στη microSD και αντίστροφα. Το Windows Phone είναι το μοναδικό τηλέφωνο με το Xbox built-in. Στο Xbox Live μπορείτε να μείνετε συνδεδεμένοι με τα επιτεύγματα του παιχνιδιού σας, τους φίλους σας, και όλα αυτά μέσω του τηλεφώνου σας. Η συγκεκριμένη έκδοση παρέχει χαρακτηριστικά όπως:

  ★ Music + Video Hub, αναβαθμισμένο και πιο απλό από ποτέ δίνει την δυνατότητα να απολαμβάνετε την αγαπημένη σας μουσική, τηλεοπτικές εκπομπές, ταινίες, podcasts, και ραδιοφωνικούς σταθμούς FM.

  ★ Me Hub, διατηρεί όλα τα κοινωνικά μέσα μαζικής ενημέρωσης σε ένα μόνο κόμβο όπου παρέχει την δυνατότητα σε χρήστες να δουν news feeds από κοινωνικά δίκτυα. Παρέχεται υποστήριξη σε κοινωνικά δίκτυα στο όπως το Facebook, το Foursquare, το Linkedin, και το Twitter.

★ Cortana, είναι ο προσωπικός βοηθός σας στο Windows Phone σας. Είναι εκεί για να βοηθήσει, κάνοντας τα πράγματα ευκολότερα για σας, και να σας κρατήσει ενήμερους για τα πράγματα που έχουν σημασία για εσάς.

## 2.7 Σύγκριση μεταξύ των κινητών συσκευών

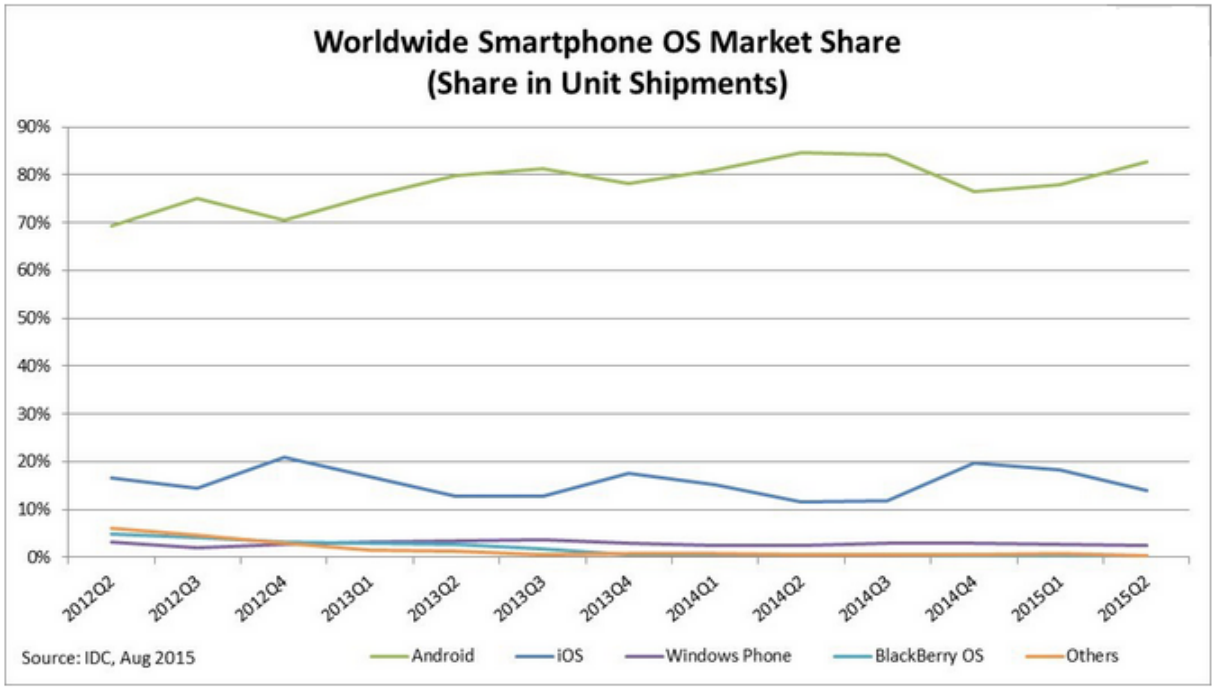
Τόσο οι πόροι - το υλικό όσο και το λογισμικό στις φορητές συσκευές έχουν βελτιωθεί σημαντικά τα τελευταία χρόνια. Οι τρεις ηγέτες της αγορά είναι: 1)το Android από την Google, με βάση το Linux λειτουργικό σύστημα, 2) το Windows Phone λειτουργικό σύστημα της Microsoft 3) και το iOS, μία πλατφόρμα από την Apple.

Η πλατφόρμα Android είναι η πιο συναρπαστική και έχει την πρώτη θέση στα κινητά λειτουργικά σύστημα που επιτρέπουν στους προγραμματιστές να παράγουν νέες εφαρμογές[20]. Το Android διευκολύνει τη χρήση των 2D και 3D γραφικών βιβλιοθηκών, παρέχει μια προσαρμοσμένη μηχανή SQL για επίμονη αποθήκευση και προηγμένες δυνατότητες δικτύου, όπως 3G, 4G και WLAN. Οι εφαρμογές εξελίσσονται συνεχώς δεδομένου ότι το Android είναι ένα λειτουργικό σύστημα ανοιχτού κώδικα, και η κοινότητα συμβάλλει στην εξέλιξη του περιβάλλοντος προγραμματισμού, του λειτουργικού συστήματος και των API εφαρμογών.

Το Windows Phone προσφέρει και αυτό λειτουργίες όπως δικτύωση, αισθητήρες και ενσωμάτωση με την κάμερα. Οι εφαρμογές των Windows Phone γράφονται σε .NET διαμορφωμένο κώδικα όπως C#. Αυτό αναλαμβάνει την ευθύνη για πολλές επιρρεπής σε λάθη και συχνά πολύπλοκες εργασίες, όπως ελέγχους αυθεντικοποίησης και ασφάλειας, διαχείρισης της μνήμης και καταστροφής των περιττών αντικειμένων. Το Windows Phone, επίσης, υποστηρίζει 2D και 3D γραφικά με την βοήθεια της πλατφόρμας XNA. Ωστόσο το IDE που χρησιμοποιείται για την δημιουργία των εφαρμογών, το Visual Studio, δεν είναι πάντα δωρεάν. Αυτό ίσως να είναι και η αιτία ενός μεγάλου μειονεκτήματος της πλατφόρμας. Δεν έχει καλό Documentation ώστε να στηρίξει ή ακόμα και να δελεάσει τον προγραμματιστή, κάτι το οποίο το παρέχουν οι 2 μεγάλοι ανταγωνιστές του σε υπερθετικό βαθμό.

Όσο για το iOS, οι εφαρμογές του αναπτύσονταν με την χρήση της προγραμματιστικής γλώσσας Objective-C χρησιμοποιώντας τη βιβλιοθήκη Cocoa Touch. Αλλά, πλέον η Apple εισήγαγε μια νέα γλώσσα ανάπτυξης, σχεδιασμένη από την ίδια την Apple, την Swift. Με την Swift, η Apple δίνει στους προγραμματιστές της περισσότερη δύναμη, αφού σε σύγκριση με την Objective-C είναι 2.6 φορές γρηγορότερη. Το μειονέκτημα της Apple είναι ότι η ανάπτυξη των εφαρμογών iOS προϋποθέτει ότι ο υπολογιστής που χρησιμοποιείτε τρέχει Mac OS. Ωστόσο το iOS παρέχει πληθώρα από διεπαφές προγραμματισμού εφαρμογών και πολλά υποσχόμενες.

Τέλος, εντύπωση κάνει η επέκταση των λειτουργικών συστημάτων iOS και Android σε ρολόγια και αυτοκίνητα, που σε αυτόν τον τομέα τα Windows Phone υστερούν.




| Period | Android | iOS | Windows Phone | BlackBerry OS | Others |
|---|---|---|---|---|---|
| 2015Q2 | 82.8% | 13.9% | 2.6% | 0.3% | 0.4% |
| 2014Q2 | 84.8% | 11.6% | 2.5% | 0.5% | 0.7% |
| 2013Q2 | 79.8% | 12.9% | 3.4% | 2.8% | 1.2% |
| 2012Q2 | 69.3% | 16.6% | 3.1% | 4.9% | 6.1% |

Source: IDC, Aug 2015

*Εικόνα 10 : Παγκόσμιο μερίδιο αγοράς λειτουργικών συστημάτων*

## 2.8 Διαδικασία ανεβάσματος

Για να δημοσιεύσεις το iOS SDK πρόγραμμά σου πρέπει να εγγραφείς στην Apple.

- Αρχικά πρέπει να εγγραφείς ως προγραμματιστής στο developer.apple.com. Κατά την σύνδεση θα ερωτηθείς για το τι εφαρμογές φτιάχνεις και θα πρέπει να συμφωνήσεις με τους όρους χρήσης για να συνεχίσεις. Μετά ειδοποιείσαι από την Apple για την επιβεβαίωση της εγγραφής σου και παίρνεις το OK για το πρόγραμμά σου. Σε αυτό το σημείο σου ζητείτε και να πληρώσεις μία αμοιβή, προσωρινά 100$ ή 300$ για να γίνεις ένας πλήρης προγραμματιστής[10].

- Στο δεύτερο βήμα, στην ιστοσελίδα developer.apple.com, πατάμε πάνω στο Member Center, βρίσκεται στην πάνω δεξιά γωνία, για να συνδεθούμε.

- Αμέσως μετά, πατάμε στην επιλογή Manage your Certificates, Apps Ids, devices and provisioning profiles, ώστε να δημιουργήσουμε ένα αίτημα πιστοποίησης για την εφαρμογή μας. Για να γίνει αυτό παράγουμε ένα Certification Sign Request(CSR) μέσω του Keychain Access του mac υπολογιστή μας(αν έχουμε!). Ανοίγουμε λοιπόν το KeyChain Access, πατάμε πάνω του, στην γραμμή εργασιών και στην λίστα που μας εμφανίζεται, επιλέγουμε Certificate Assistant > Request a Certificate From a Certificate Authority... Ύστερα στο παράθυρο του developer.apple που βρισκόμαστε ακολουθούμε: Certificates > Development tab, ανεβάζουμε την αίτηση που μόλις φτιάξαμε και περιμένουμε μέχρι να γίνει αποδεχτή.

- Μόλις η αίτησή μας γίνει αποδεχτή από τον διαχειριστή, που στην συγκεκριμένη περίπτωση είμαστε εμείς οι ίδιοι, κατεβάζουμε την πιστοποίηση ως εξής:
  Certificate > Development, και διπλοκλικάρουμε download WWDR Intermediate Certificate. Με την ολοκλήρωση της λήψης, βρίσκουμε την πιστοποίηση και ξανά κάνουμε διπλό κλικ σ' αυτή για να την εγκαταστήσουμε.

- Έπειτα στο παραθυράκι Devices προσθέτουμε τις συσκευές, στις οποίες θέλουμε να κτίσουμε την εφαρμογή μας.

- Δημιουργούμε ένα App ID που είναι αναγκαίο για κάθε εφαρμογή, επιλέγουμε:
  Identifiers > App IDs και κλικάρουμε στο +. Στο παράθυρο που μας εμφανίζεται εισάγουμε το όνομα της εφαρμογής στην φόρμα που βρίσκετε στην περιγραφή. Στην περιοχή του App ID suffix επιλέγουμε το Explicit App ID, και εισάγουμε Bundle id, βρίσκοντάς το ως εξής:
  - ανοίγουμε το Xcode,
  - ανοίγουμε την εφαρμογή,
  - πατάμε πάνω της στη γραμμή εργασιών του Xcode
  - τοποθετούμε το όνομα της κάτω από το targets,

    - πατάμε στο info που βρίσκεται και αυτό στην γραμμή εργασιών του Xcode και
    - αντίγραφουμε το Bundle Identifier.

- Δημιουργούμε ένα provision profil
    - Επιλέγουμε Provision Profile > Distribution
    - πατάμε πάνω στο +.
    - επιλέγουμε το App Store, και πατάμε Continue
    - επιλέγομε το App id που δημιουργήσαμε στα προηγούμενα βήματα και πατάμε Continue
    - επιλέγουμε το πιστοποιητικό που δημιουργήσαμε και πατάμε Continue
    - εισάγουμε ένα όνομα profil(δεν έχει σημασία τι όνομα θα επιλέξουμε) και πατάμε Generate

- Κατεβάζουμε το provision profil που μόλις δημιουργήσαμε

- Έπειτα το βρίσκουμε στον υπολογιστή μας
    - διπλο κλικάρουμε επάνω του
    - πατάμε πάνω στο add to library

- Ανοίγουμε το Xcode
    - επιλέγουμε την εφαρμογή που θα ανεβάσουμε στο App store
    - πατάμε πάνω της στη γραμμή εργασιών του Xcode
    - τοποθετούμε το όνομα της κάτω από το targets,
    - πατάμε στο info που βρίσκεται και αυτό στην γραμμή εργασιών του Xcode
    - Στην περιοχή του Configuration βεβαιωνόμαστε ότι υπάρχουν (ή τα δημιουργούμε αν δεν υπάρχουν) μόνο τα εξείς 3 Configurations: Release, Debug, Distribution
    - πάλι όπως και πριν, τοποθετούμε το όνομα της εφαρμογής μας κάτω από το targets, και πατάμε πάνω του
    - πατάμε στο Build Settings που βρίσκεται και αυτό στην γραμμή εργασιών του Xcode
    - και στην περιοχή Code Signing τοποθετούμε σε όλες τις επιλογές το πιστοποιητικό που δημιουργήσαμε και το Provision Profile που δημιουργήσαμε.
    - έπειτα πατάμε πάνω στο όνομα της εφαρμογής μας που βρίσκεται στη γραμμή εργασιών του Xcode και επιλέγουμε το project μας
    - και πάλι, στην περιοχή Code Signing τοποθετούμε σε όλες τις επιλογές το πιστοποιητικό που δημιουργήσαμε και το Provision Profile που δημιουργήσαμε.
    - έπειτα πατάμε πάνω στο όνομα της εφαρμογής μας που βρίσκεται στη γραμμή εργασιών
    - στο πάνω μέρος του Xcode(στην μπάρα με τα εργαλεία) αλλάζουμε το προσομοιωτή μας σε iOs Device
    - και στο πάνω μέρος του Xcode, στη γραμμή με τα εργαλεία, επιλέγουμε Product > Archive

- Έπειτα , στην ιστοσελίδα developer.apple.com, πατάμε πάνω στο Member Center, πατάμε την επιλογή του iTunes Connect.

- Επιλέγουμε Manage your Apps. > Add New και συμπληρώνουμε την φόρμα βάζοντας στο πεδίο Bundle Id το προηγούμενο Bundle Id που δημιουργήσαμε, προσθέτοντας τις φωτογραφίες του εικονιδίου της εφαρμογής.

- Ανοίγουμε το Archive αρχείο που δημιουργήσαμε και με το πρόγραμμα Organizer και πατάμε Distribute..
    - επιλέγουμε Submit to the iOS App Store
    - και πατάμε Next
    - και τέλος Submit

Για την δημοσίευση τις εφαρμογής στο Google Play πρέπει να εγγραφείς στο htttp://play.google.com/apps/publish. Πληρώνεις μόνο μία φορά εγγραφή $25. Σε αντίθεση με άλλες δημοφιλείς κινητές πλατφόρμες το Google Play δεν έχει διαδικασία έγκρισης για ανέβασμα εφαρμογών[8]. Απαιτεί όμως την αποδοχή των όρων χρήσης και την μη παραβίαση τους από την εφαρμογή. Πριν ανεβάσεις την εφαρμογή σου πρέπει να υπογράψεις ψηφιακά το αρχείο .apk χρησιμοποιώντας ένα ψηφιακό πιστοποιητικό, που προσδιορίζει τον συγγραφέα της εφαρμογής. Τέλος συνδέεσαι στο Google Play και ανεβάζεις την εφαρμογή. Υπάρχουν και άλλες θέσεις αγορών εφαρμογών Android όπως το Amazon Appstore και Samsung Apps αλλά με Google Play να είναι σίγουρα το πιο διαδεδομένο

Η Microsoft διαθέτει 2 τύπους λογαριασμών τον Ατομικό λογαριασμό(14 EUR) με βασικές δυνατότητες εφαρμογής και τον Εταιρικό λογαριασμό(75 EUR) με ευρείες δυνατότητες εφαρμογής, όπου δυνατότητες εφαρμογής είναι οι προγραμματιστικές ανάγκες για πρόσβαση σε πόρους του χρήστη, δηλαδή φωτογραφίες και συσκευές όπως η κάμερα[21].

# 3. Ροές ειδήσεων

## 3.1 Δικτυακά δεδομένα Rss

Η απαρχή του RSS ήταν ένα σύστημα που λεγόταν Meta Content Framework (MCF). Εμφανίστηκε το 1995 και είχε ως στόχο να περιγράψει αντικείμενα, με τα στοιχεία τους και τις συσχετίσεις μεταξύ τους. Το MCF, που ήταν μία πειραματική έρευνα της Apple, το χρησιμοποίησαν αρκετές ιστοσελίδες για να περιγράψουν το περιεχόμενό τους. Αργότερα άρχισε να αλλάζει σε XML μορφή και αυτό το έργο ονομάστηκε RDF(Resource Description Framework).

Το RDF ήταν μία γλώσσα γενικού σκοπού που σκόπευε στην αναπαράσταση των πληροφοριών που υπήρχαν στο διαδίκτυο και ήταν η βάση της έννοιας Semantic Web, ένα όραμα της W3C's που θέλει τους υπολογιστές να μπορούν να καταλαβαίνουν τις πληροφορίες που βρίσκονται στο διαδίκτυο. Με την ολοκλήρωση του έργου, όταν η το RDF διατυπώθηκε σε XML, η εταιρία NetScape λάνσαρε ένα έργο που ονομάστηκε "My netscape NetWork" και μαζί του ήρθε και το RSS, συντομογραφία του RDF Site Summary[22], το οποίο μετά αποστρατεύτηκε από την Netscape και το παρέλαβε η κοινότητα.

Η ονομασία της RSS τεχνολογίας είναι συγκεχυμένη. Δεν γνωρίζουμε αν το RSS σημαίνει Really Simple Syndication, RDF Site Summary ή Rich Site Summary. Το RSS είναι ένα σύνολο από πρωτόκολλα διανομών περιεχομένου στο διαδίκτυο και αναδημοσίευσης τους, που χρησιμοποιείται από ιστοσελίδες ειδήσεων και blogs για να ανακοινώσουν πρόσφατες προσθήκες στο περιεχόμενό τους[23]. Ο κάθε φάκελος RSS περιέχει μία λίστα από αντικείμενα εκ των οποίων κάθε αντικείμενο έχει ένα τίτλο, μία περίληψή και ένα URL. Όταν ένας φάκελος RSS ανανεώνεται, όλες οι ιστοσελίδες και οι περιηγητές που είναι εγγεγραμμένοι σε αυτόν ανανεώνονται επίσης[24]. Αυτό κάνει τα δικτυακά δεδομένα Rss έναν δημοφιλή μηχανισμό που παραδίδει νέα-ειδήσεις σε εγγεγραμμένους χρήστες, συμβάλει στην αύξηση της επισκεψιμότητας της ιστοσελίδας και βελτιώνει τη σχέση ιστοσελίδας και χρήστη.

Το Rss μπορεί να θεωρηθεί επίσης και ως ένας πράκτορας που ελέγχει περιοδικά τα Url του κάθε χρήστη για δει αν έχουν ανανεωθεί. Και όταν τελικά το περιεχόμενο τους ανανεώνεται μετά από τον τελευταίο έλεγχο, ειδοποιεί τον χρήστη. Οι χρήστες εγγράφονται στις ενημερώσεις αυτές χρησιμοποιώντας έναν aggregator ή ένα πρόγραμμα ανάγνωσης ειδήσεων που καταγράφει την ιστοσελίδα σε τακτική βάση, ίσως μία φορά την εβδομάδα, ίσως μία φορά κάθε τέταρτο της ώρας. Οι aggregators εμφανίζουν τα feeds και επιτρέπουν στους χρήστες να τα οργανώσουν όπως θέλουν και να έχουν πρόσβαση στις σχετικές ιστοσελίδες, όταν αυτές είναι διαθέσιμες[25].

Παράδειγμα RSS αρχείου:

```
<?xml version="1.0" encoding="UTF-8" ?>
<rss version="2.0">

<channel>
  <title>W3Schools Home Page</title>
  <link>http://www.w3schools.com</link>
  <description>Free web building tutorials</description>
  <item>
    <title>RSS Tutorial</title>
    <link>http://www.w3schools.com/webservices</link>
    <description>New RSS tutorial on W3Schools</description>
  </item>
  <item>
    <title>XML Tutorial</title>
    <link>http://www.w3schools.com/xml</link>
    <description>New XML tutorial on W3Schools</description>
  </item>
</channel>

</rss>
```

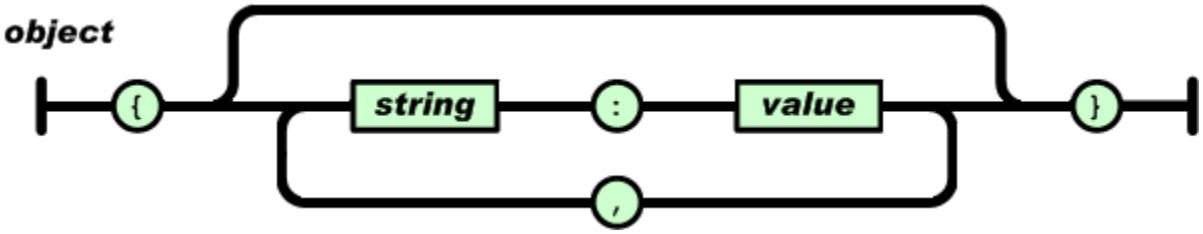
## 3.2 Δικτυακά δεδομένα Json

Ο Douglas Crockford ήταν ο πρώτος που καθόρισε και διέδωσε τη μορφή JSON. Το ακρώνυμο επινοήθηκε από τη State Software, μια εταιρεία που ιδρύθηκε από τον Crockford, τον Chip Morningstar και τον Robert F. Napiltonia τον Απρίλιο του 2001 και χρηματοδοτήθηκε από την Tesla Ventures[26].

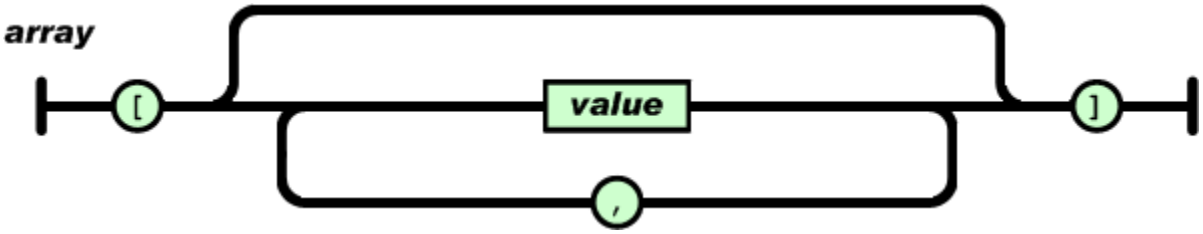
Τα διαδικτυακά δεδομένα JavaScript Object Notation (JSON) έχουν μορφή κειμένου και είναι μία διαδικασία ανταλλαγής δεδομένων σε ανεξάρτητη γλώσσα. Είναι εύκολο να διαβαστούν και να γραφούν από ανθρώπους και επίσης εύκολο να παραχθούν και να αναλυθούν από υπολογιστές. Αναπαριστούν δομές δεδομένων σειριακά με αλφαριθμητικά, αριθμούς, booleans, null, αντικείμενα και πίνακες. Αναπτύχθηκαν από την ανάγκη επικοινωνίας σε πραγματικό χρόνο χωρίς κατάσταση(statefull) μεταξύ διακομιστή και πελάτη. Μια τέτοια υπηρεσία μεταξύ πελάτη - διακομιστή υλοποιείται χρησιμοποιώντας HTTP ή HTTPS πρωτόκολλο[27]. Ο διακομιστής χρησιμοποιείται ως ένας τομέας αποθήκευσης δεδομένων που

είναι σε θέση να παρέχει REST web services, δηλαδή να λαμβάνει αιτήματα από τον http πελάτη και να επιστρέφει πληροφορίες που βρίσκονται στο διαδίκτυο σε μορφή JSON.

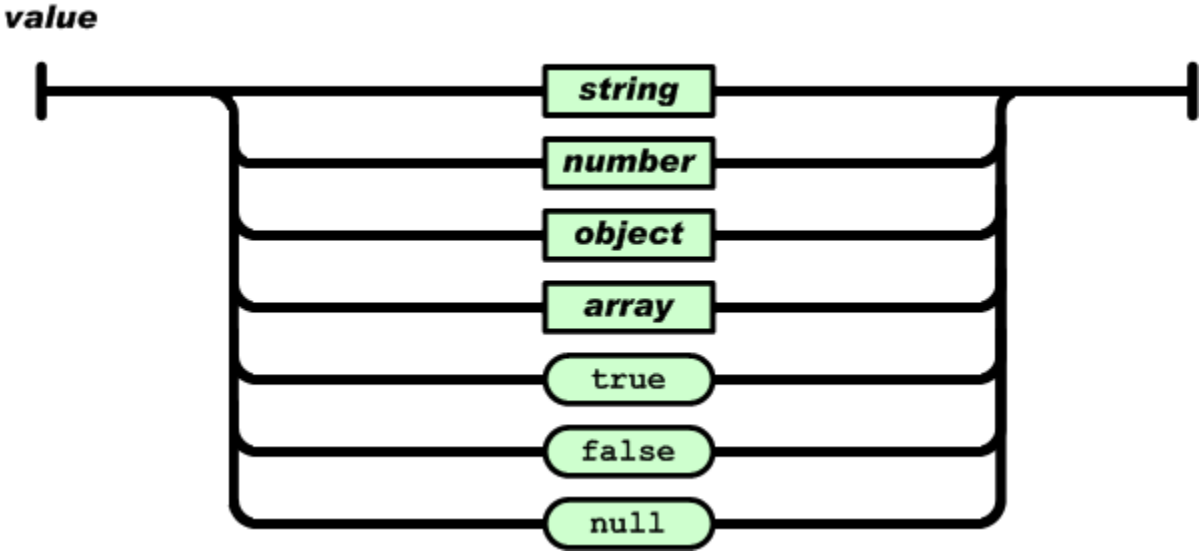
Τα JSON εκτιμάται ότι αναλύονται μέχρι εκατό φορές ταχύτερα από ότι τα XML διαδικτυακά δεδομένα στους σύγχρονους browsers, αλλά παρά τους ισχυρισμούς αυτούς της αξιοσημείωτης απόδοσης, τα επιχειρήματα εναντίον των JSON περιλαμβάνουν την έλλειψη της υποστήριξης namespace, έλλειψη επικύρωσης των εισροών και μειονεκτήματα επεκτάσιμα των προηγουμένων[28].



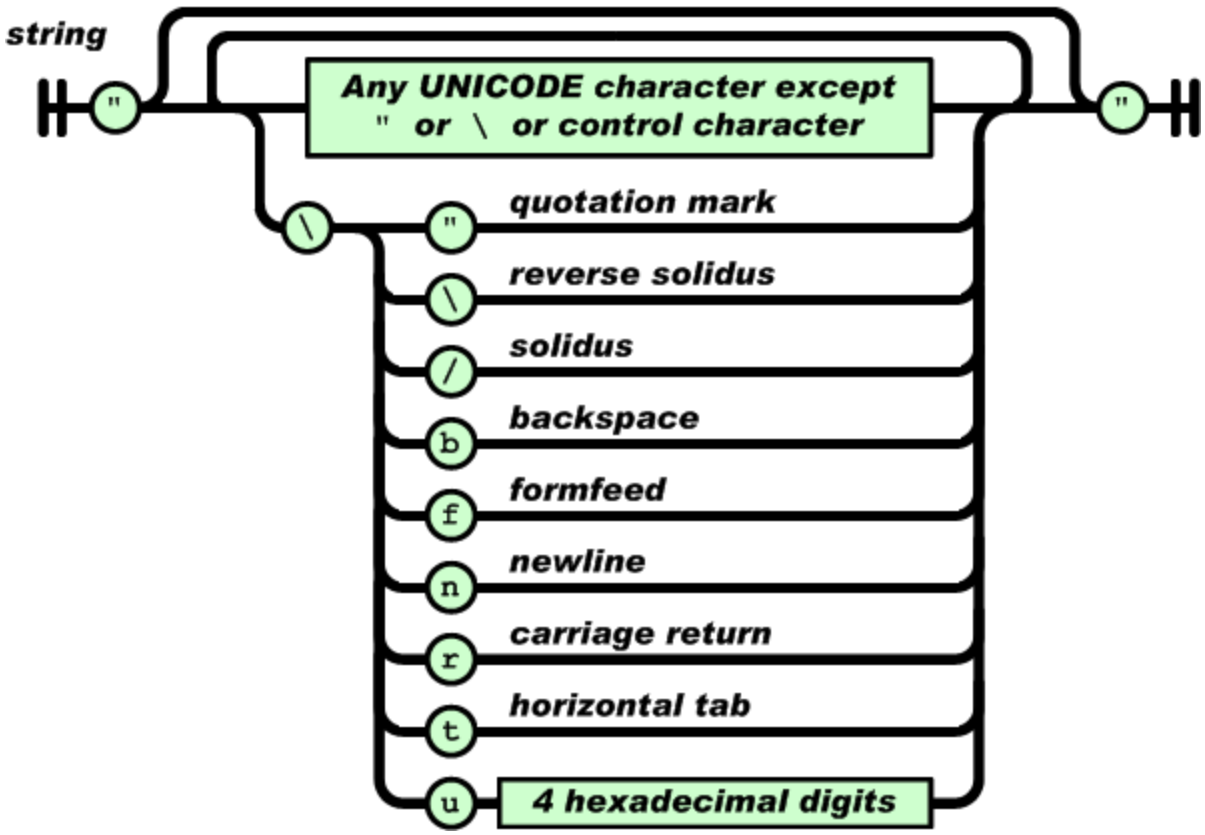

*Εικόνα 11: JSON Object*



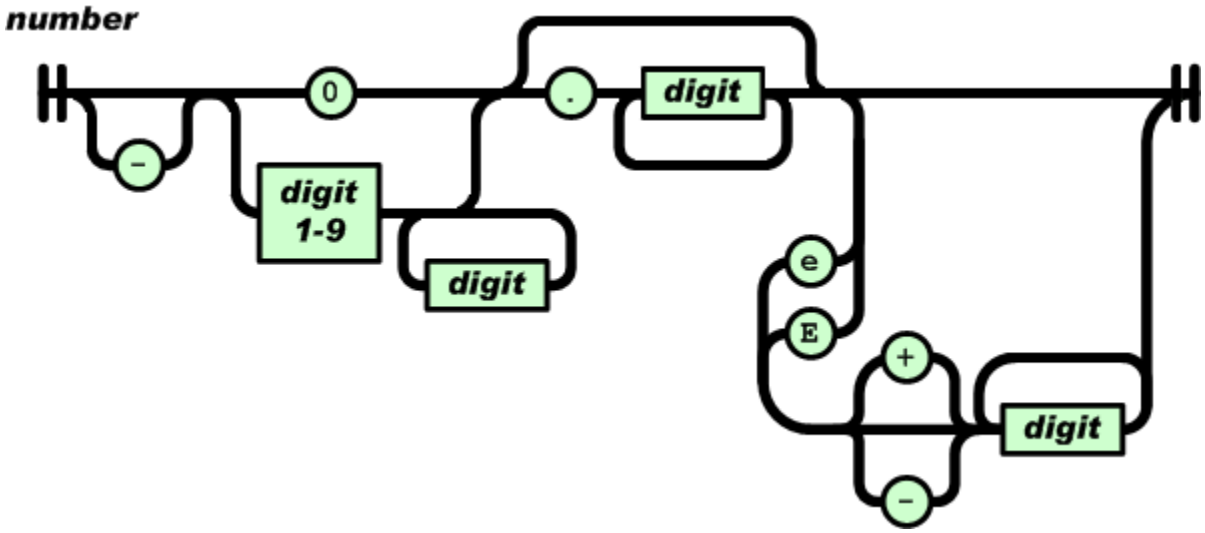

*Εικόνα 12: JSON Array*




*Εικόνα 13: JSON Value*




*Εικόνα 14: JSON String*




*Εικόνα 15: JSON Number*

## 3.3 Δικτυακά Δεδομένα Xml

Η επιτυχία του διαδικτύου, εν μέρη, βασίστηκε στην HTML. Ο κάθε χρήστης ακολουθώντας τους συνδέσμους που παρείχε η HTML γλώσσα μπορούσε πλοηγηθεί και να πάρει την πληροφορία που επιθυμούσε. Παρόλα αυτά, το κύριο πλεονέκτημά της - η απλότητα - είναι ταυτόχρονα και ένα μειονέκτημα.

Η HTML έχει να κάνει κυρίως με την παρουσίαση των πληροφοριών, τις παραγράφους, τις γραμματοσειρές, που μόνο οι άνθρωποι μπορούν να νοήσουν το περιεχόμενό τους. Από την άλλη πλευρά η ανάπτυξη του ηλεκτρονικού εμπορίου παρουσίασε την ανάγκη για μία γλώσσα που θα μπορούσε να περιγράψει και το περιεχόμενο των πληροφοριών που υπάρχει σε ένα έγγραφο στο διαδίκτυο με έναν τρόπο που θα ήταν κατανοητός στους υπολογιστές. Τα καλά νέα ήταν ότι αυτή η γλώσσα υπήρχε και ονομαζόταν SGML αλλά τα κακά νέα ήταν ότι ήταν αρκετά πολύπλοκη και ασύμβατη για το Internet.

Το 1998, μία καινούργια μετά γλώσσα αναπτύχθηκε με το όνομα XML, eXtensible Markup Language, αντικατέστησε την SGML και θεωρήθηκε ως θεμέλιο για τη επόμενη γενιά του Internet. Η XML έχει μορφή απλού κειμένου και είναι εύκολο να γραφτεί και να διαβαστεί. Μπορεί να δημιουργηθεί ακόμα και με ένα απλό σημειωματάριο, αρκεί η προέκταση .xml. Η xml χρησιμοποιεί ετικέτες(φράσεις μέσα σε τριγωνικές αγκύλες) για να δηλώσει την σημασία των δεδομένων που περιέχει. Για την ακρίβεια οι ετικέτες εμφανίζονται σε ζευγάρια, αρχικής-τελικής ετικέτας που πρέπει να ταιριάζουν μεταξύ τους ακριβώς, με την μόνη διαφορά ότι περιέχεται μία κάθετη παύλα στην τελική ετικέτα μετά το άνοιγμα τη πρώτης αγκύλης[29]. Ο συνδυασμός αρχικής ετικέτας, τελικής ετικέτας και του περιεχομένου τους καλείτε στοιχείο. Μερικά στοιχεία περιέχουν χαρακτηριστικά (attributes) που απαρτίζονται από ζευγάρια ονόματος - τιμής, βρίσκονται στην αρχική ετικέτα και αναπαριστούν επιπλέον πληροφορία.

Η XML έχει να κάνει με την αναπαράσταση δεδομένων και δίνει την δυνατότητα και στους υπολογιστές να κατανοήσουν περισσότερα για την σημασία των δεδομένων που αναπαριστούν. Για παράδειγμα αν το 100 σημαίνει 100$ μπορεί να αναπαρασταθεί σε XML ως <Price currency="dollar">100</Price> ενώ αν ήταν ταχύτητα θα μπορούσε να γραφεί <Speed>100</Speed>. Έτσι το 100 δεν είναι πια ένας απλός αριθμός αλλά κάτι που έχει σημασία. Αυτό κάνει την XML μία διαλειτουργική γλώσσα μεταξύ υπολογιστών και λογισμικών μέσω του διαδικτύου, δηλαδή οι εφαρμογές πλέον μπορούν να επικοινωνούν διαδικτυακά στέλνοντας αρχεία XML.

Ένα χαρακτηριστικό πλεονέκτημα της XML είναι και τα Namespaces. Η χρήση τους δεν είναι υποχρεωτική όμως χρησιμοποιούνται συχνά. Για παράδειγμα ας υποθέσουμε ότι έχουμε δυο αρχεία XML από κάποια ξενοδοχεία. Στην περίπτωση που τα αρχεία αυτά περιέχουν και τα δύο την ετικέτα <Foods>, εάν συνδυαστούν δεν θα προκύψει, μία σωστή ερμηνεία για το στοιχείο αυτό. Όταν δύο XML αρχεία που περιέχουν ταυτόσημα στοιχεία από διαφορετικές πηγές συγχωνευθούν, αυτά τα στοιχεία λέγεται ότι συγκρούονται[30]. Τα Namespaces δηλώνονται προσθέτοντας το xmlns:*κάποιο στοιχείο*, όπου το *κάποιο στοιχείο* δηλώνει την ταυτότητα του namespace. Με την χρήση των Namespace το κάθε στοιχείο μπορεί να παραμείνει μοναδικό.

Ειδικό παράδειγμα χρήσης Namespace:

```
<Hotels xmlns="www.paradisehotel.com">
     <breakfast_menu>
     </breakfast_menu>
</Hotels>
```

Γενικό παράδειγμα χρήσης XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<breakfast_menu>
<food>
  <name>Belgian Waffles</name>
  <price>$5.95</price>
  <description>Our famous Belgian Waffles with plenty of real
maple syrup</description>
  <calories>650</calories>
</food>
<food>
  <name>French Toast</name>
  <price>$4.50</price>
  <description>Thick slices made from our homemade sourdough
bread</description>
  <calories>600</calories>
</food>
<food>
  <name>Homestyle Breakfast</name>
  <price>$6.95</price>
  <description>Two eggs, bacon or sausage, toast, and our ever-
popular hash browns</description>
  <calories>950</calories>
</food>
</breakfast_menu>
```

# 4. Ανάπτυξη εφαρμογών σε android

## 4.1 Ιστορική αναδρομή

Μέχρι και το τέλος του 2014, το επίσημο ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) για εφαρμογές Android, ήταν το Eclipse. Χρησιμοποιώντας το Android Development Tools (ADT) Plugin, το Eclipse υποστήριζε πλήρως την ανάπτυξη εφαρμογών Android, ωστόσο μία δοκιμαστική έκδοση του Android Studio ήταν παράλληλα διαθέσιμη.

Πλέον κυκλοφορεί η σταθερή έκδοση του Android Studio. Και σήμερα, το Android Studio είναι το επίσημο IDE για την ανάπτυξη εφαρμογών Android, με βάση το IntelliJ IDEA. Το intelliJ IDEA είναι ένα έξυπνο πρόγραμμα επεξεργασίας κώδικα ικανό για προηγμένη ολοκλήρωση κώδικα, Refactor και για ανάλυση κώδικα που σας βοηθά να γίνετε ένας παραγωγικός προγραμματιστής Android εφαρμογών.

Όμως πάνω από τις δυνατότητες που θα περιμένατε από το IntelliJ, το Android Studio προσφέρει και πολλές ακόμα δυνατότητες:

- προβολή της μνήμης και της CPU οθόνη ώστε να μπορείτε πιο εύκολα να παρακολουθείτε την απόδοση και τη χρήση της μνήμης της εφαρμογής σας, για να παρακολουθείτε τη χρήση της CPU, να βρείτε deallocated αντικείμενα, να εντοπίσει διαρροές μνήμης και να παρακολουθείτε την ποσότητα της μνήμης που η συνδεδεμένη συσκευή χρησιμοποιεί.

- Το εργαλείο Android build που χρησιμοποιείτε για να δημιουργήσετε, να δοκιμάσετε, να τρέξετε και να συσκευάσετε σε πακέτα τις εφαρμογές σας. Αυτό το σύστημα κατασκευής αντικαθιστά το ant σύστημα που χρησιμοποιείται με το Eclipse ADT. Μπορεί να τρέξει ως ένα ολοκληρωμένο εργαλείο από το μενού Android Studio και ανεξάρτητα από τη γραμμή εντολών.

Η ανάπτυξη στο SDK του Android ακολουθεί την συνολική ανάπτυξη του Android. Το SDK υποστηρίζει επίσης παλαιότερες εκδόσεις της πλατφόρμας Android σε περίπτωση που οι προγραμματιστές θέλουν να στοχεύσουν τις εφαρμογές τους σε παλαιότερες συσκευές.

## 4.2 Εγκατάσταση του android plugin σε eclipse

Απαραίτητη προϋπόθεση για την εγκατάσταση του plugin είναι η προϋπάρχουσα εγκατάσταση του eclipse που απαιτεί :

- Έκδοση Eclipse 3.7.2(Indigo) ή μεγαλύτερη
- Eclipse JDT plugin που περιλαμβάνεται στα περισσότερα Eclipse IDE πακέτα.
- JDK 6 ή 7

Στη συνέχεια για να προσθέσουμε το ADT plugin:

1. Ξεκινάεμε το Eclipse, και επιλέγουμε Help > Install New Software.
2. Πατάμε το Add, στην πάνω αριστερή γωνία.
3. Στον διάλογο που εμφανίζεται, εισάγουμε “ADT Plugin" στο πεδίο όνομα και στο πεδίο Τοποθεσία το URL: https://dl-ssl.google.com/android/eclipse/
4. Πατάμε το OK
5. Στον διάλογο: Διαθέσιμο Λογισμικό, επιλέγουμε το πλαίσιο ελέγχου δίπλα από τα Εργαλεία ανάπτυξης(Developer Tools) και πατάμε το Next
6. Στο επόμενο παράθυρο που εμφανίζεται μία λίστα με τα εργαλεία που πρόκειται να κατεβάσουμε πατάμε next.
7. Διαβάζουμε τους όρους χρήσης και πατάμε Finish.
8. Όταν η εγκατάσταση τελειώσει, παρακινάμε το Eclipse.

Μόλις το Eclipse ανοίξει προσδιορίζουμε την τοποθεσία του Android SDK:

1. Στο παράθυρο υποδοχής με το μήνυμα “Καλώς ορίσατε κτλ...” που εμφανίζεται, επιλέγουμε Use existing SDKs.
2. Επιλέγουμε την τοποθεσία του Android SDK που κατεβάσαμε παραπάνω και πατάμε Next.

Όλα είναι έτοιμα. Ενημερώνουμε τα εργαλεία της πλατφόρμας κατεβάζοντας τα πακέτα των καινούργιων εκδόσεων πατώντας το SDK Manager στη μπάρα εργαλείων και αρχίζουμε να προγραμματίζουμε...

## 4.3 Χρήση android για πολυμέσα

Χρησιμοποιώντας τις κάμερες των κινητών συσκευών, οι χρήστες μοιράζονται τις εμπειρίες τους και οι εφαρμογές γίνονται μέρος της ζωής των χρηστών. Το Android απλοποιεί την εγγραφή βίντεο όπως προαναφέραμε. Οι εφαρμογές, μιας και οι περισσότερες συσκευές με λογισμικό Android έχουν ήδη μια εφαρμογή κάμερας που καταγράφει βίντεο, κάνουν χρήση της υπάρχουσας εφαρμογή κάμερας, ελέγχουν τη συσκευή τροποποιώντας τις ρυθμίσεις της κάμερας και τραβούν φωτογραφίες και βίντεο..

Το Android πλαίσιο εργασιών (framework) πολυμέσων περιλαμβάνει υποστήριξη για την αναπαραγωγή ποικίλων από συνήθεις τύπους πολυμέσων, έτσι ώστε να μπορείτε να ενσωματώσετε εύκολα ήχο, βίντεο και εικόνες σε εφαρμογές σας. Μπορείτε να αναπαράγετε ήχο ή βίντεο από τα αρχεία πολυμέσων που είναι αποθηκευμένα στους πόρους της εφαρμογής σας, από αυτόνομα αρχεία στο σύστημα αρχείων, ή από ένα ρεύμα δεδομένων που φθάνει μέσω μιας σύνδεσης δικτύου[31].

Το Android παρέχει μία λειτουργία εστίασης που διαχειρίζεται την αναπαραγωγή ήχου. Επειδή σε μία συσκευή υπάρχουν πολλαπλές εφαρμογές για αναπαραγωγή ηχητικού υλικού, το Android για να εμποδίσει την ταυτόχρονη αναπαραγωγή ήχου χρησιμοποιεί την εστίαση ήχου για να επιτρέψει μόνο σε μία εφαρμογή να αναπαράγει ήχο και κατ' επέκταση μόνο η εφαρμογή που είναι εστιασμένη αναπαράγει ήχο. Συνεπάγεται ότι οι εφαρμογές δέχονται και λαμβάνουν αιτήματα για να δώσουν και να λάβουν την εστίαση και έτσι πρέπει να διαχειρίζονται και το τι θα κάνουν σε περίπτωση που χάνουν τη σκυτάλη.

Οι εφαρμογές μπορούν να αιτηθούν να πάρουν τη σκυτάλη προσωρινά, για μικρά αποσπάσματα ήχου όπως οδηγίες πλοήγησης, και μόνιμα, όταν για παράδειγμα παίζουν ένα μουσικό κομμάτι. Οι παροδικές αναπαραγωγές ήχου παρέχουν μία επιπλέον επιλογή η οποία λέγεται "ducking" και διευθετεί, αν είναι αποδεχτό και από την εφαρμογή αναπαραγωγής ήχου που βρίσκεται στο προσκήνιο, την ελάττωση της έντασης της εφαρμογής που χάνει την εστίαση μέχρι να την ξαναπάρει πίσω.

Μία ακόμα επιλογή που παρέχει το Android στους χρήστες του είναι ότι τους δίνει έλεγχο στα πολυμέσα που αναπαράγουν μέσω της συσκευής και καθιστά δυνατή την αυξομείωση των ήχων της συσκευής. Μέσω των κουμπιών του hardware υλικού της συσκευής android, ο χρήστης της συσκευής μπορεί και ρυθμίζει τους ήχους του τηλεφώνου ανάλογα με το πώς την χρησιμοποιεί την συγκεκριμένη στιγμή. Το Android διατηρεί μια ξεχωριστή ροή ήχου, για την μουσική που παίζει, τους συναγερμούς, τις ειδοποιήσεις, τις εισερχόμενες κλήσεις, τους ήχους του συστήματος, τον ήχο κατά τη διάρκεια κλήσης, και των τόνων DTMF. Αυτό γίνεται κυρίως για να επιτρέψει στους χρήστες να ελέγχουν την ένταση του κάθε ρεύματος ανεξάρτητα[32]. Με αυτόν τον τρόπο ο χρήστης αν ακούει μουσική πατώντας να χαμηλώσει τον ήχο χαμηλώνει τον ήχο της αναπαραγωγής του μουσικού κομματιού που ακούει, αν μιλάει στο τηλέφωνο χαμηλώνει τον ήχο του ακουστικού τηλεφώνου, κ.ο.κ.

## 4.4 Αποθήκευση δεδομένων σε android προγράμματα

Το Android παρέχει αρκετούς τρόπους για αποθήκευση μόνιμων δεδομένων οι οποίοι διαχωρίζονται από το αν τα δεδομένα που θα αποθηκευτούν θα είναι ιδιωτικά(private) ή δημόσια(public) και από το μέγεθός τους[33].

Οι επιλογές που προκύπτουν είναι οι εξείς:

- Shared Preferences
  Επιτρέπει την αποθήκευση και την ανάκτηση ζευγαριών κλειδιού-τιμής primitive τύπου δεδομένων. Χρησιμοποιείται για μικρές ποσότητες δεδομένων.

- Intrernal Storage
  Αποθηκεύει απευθείας στον εσωτερικό αποθηκευτικό χώρο της συσκευής. Από προεπιλογή, τα δεδομένα που αποθηκεύονται στον εσωτερικό χώρο αποθήκευσης είναι ιδιωτικά και άλλες εφαρμογές δεν έχουν πρόσβαση σ' αυτά. Όταν η εφαρμογή απεγκαθίσταται τα δεδομένα διαγράφονται[34].

- External Storage
  Είναι ο κοινόχρηστος αποθηκευτικός χώρος που μοιράζεται και από άλλες εφαρμογές και που μπορεί να είναι είτε μια αφαιρούμενη κάρτα μνήμης όπως μία SD card ή ένας αποθηκευτικός χώρος στο κινητό. Συνήθως χρησιμποποιείται για αποθήκευση πολυμέσων.

- SQLite Database
  Το Android υποστηρίζει SQLite Βάσεις δεδομένων. Όλες οι βάσεις δεδομένων που δημιουργείτε είναι προσβάσιμες σε οποιαδήποτε κλάση της εφαρμογή κατά όνομα, αλλά δεν είναι προσβάσιμες σε άλλες εφαρμογές.

- Network Connection
  Αποθηκεύει-ανακτά δεδομένα σε διακομιστή στο internet χρησιμοποιώντας το δίκτυο(όταν είναι διαθέσιμο).

# 5. Η προτεινόμενη εφαρμογή

## 5.1 Πηγές ειδήσεων

Η εφαρμογή τραβάει τα δεδομένα - ειδήσεις από έναν php διακομιστή που περιέχει τα δεδομένα που χρειαζόμαστε. Τα δεδομένα αυτά υπάρχουν σε wordpress ιστοσελίδες στο internet όπως για παράδειγμα είναι το https://gkiouzelis.wordpress.com/ ,το https://athensofmyheart.wordpress.com/ και το https://orthodoxyislove.wordpress.com/ . Ωστόσο ο διακομιστής που είναι κατά κύριο λόγο ένας Document Object Model (DOM) αναλυτής που φιλτράρει τα html δεδομένα των παραπάνω σελίδων και εξάγει σε μορφή JSON, υλοποιήθηκε για μια πιο ευέλικτη διαχείριση του υλικού. Στην συνέχεια υλοποιείται ένα Rest Web Service όπου ο πελάτης είναι η εφαρμογή μας “Orthodox Heart”. Η εφαρμογή στέλνει http αιτήματα(requests) και λαμβάνει απαντήσεις(responses), σχετικά, με τις διαθέσιμες ιστοσελίδες, με τα άρθρα - ειδήσεις που υπάρχουν στην κάθε μια και με το περιεχόμενο της κάθε είδησης, με την βοήθεια του Retrofit. Οι απαντήσεις είναι σε μορφή JSON όπως προείπαμε, και η εφαρμογή μας που τα λαμβάνει τα επεξεργάζεται και τα παρουσιάζει στον χρήστη.

## 5.2 Παραδείγματα χρήσεως της εφαρμογής

Αυτή είναι η προτεινόμενη εφαρμογή μας. Ονομάζεται Orthodox Heart όπως φαίνεται στην εικόνα 16.


*Εικόνα 16: Εικονίδιο Εφαρμογής*

Με την εκκίνηση της εφαρμογής, εμφανίζεται το κυρίως μενού που περιέχει τις ιστοσελίδες χριστιανικών ειδήσεων όπως φαίνεται και παρακάτω. Ο χρήστης έχει τη δυνατότητα να επιλέξει ποιό site τον ενδιαφέρει και εν συνεχεία να δει το περιεχόμενό του.


*Εικόνα 17: Μενού εφαρμογής*

Με την επιλογή της ιστοσελίδας από το μενού εφαρμογής εμφανίζεται ένα επιπλέον μενού το οποίο περιέχει τα διαθέσιμα άρθρα ειδήσεων. Αρχικά εμφανίζονται τα πιο πρόσφατα άρθρα ειδήσεων. Έπειτα δίνεται η δυνατότητα στο χρήστη να εμφανίσει περισσότερα άρθρα όπως φαίνεται στην εικόνα 19. Έτσι ώστε να μην φορτώνεται μεγάλος όγκος δεδομένων στο κινητό εκτός εάν το απαιτήσει ο χρήστης.

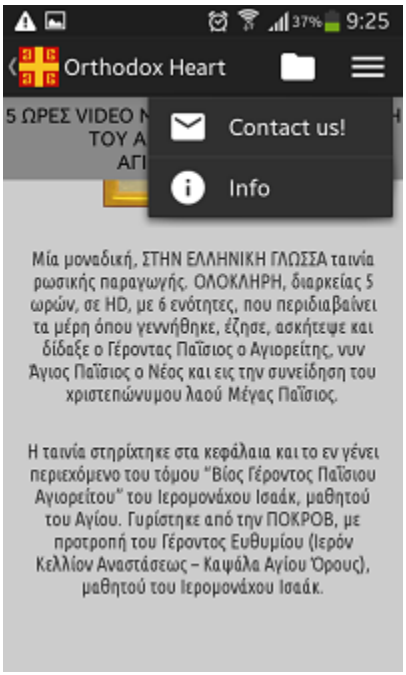

*Εικόνες 18-19 : Μενού ειδήσεων*

Κατά την επιλογή μιας είδησης-άρθρου εμφανίζονται οι παρακάτω εικόνες. Η εφαρμογή εμφανίζει ένα τίτλο με την επιλεγμένη είδηση. Όπως βλέπουμε και στις εικόνες 20-21 υπάρχει ένα floating button με το εικονίδιο της δισκέτας. Σε περίπτωση που ο χρηστής το επιλέξει, η εφαρμογή αποθηκεύει μόνο το κείμενο και τα URL του συγκεκριμένου άρθρου.


*Εικόνες 20-21: Παρουσίαση άρθρου*

Εφόσον οι ιστοσελίδες περιέχουν ως επί το πλείστον φωτογραφίες και βίντεο, η εφαρμογή έχει τη δυνατότητα να τα παρουσιάζει στο χρήστη. Επίσης, για κάθε URL που υπάρχει στην είδηση, δημιουργείται ένα κουμπί το οποίο όταν επιλεγεί:
1) Σε περίπτωση που ο σύνδεσμος που περιέχει είναι ένα site του αρχικού μενού επαναφέρει το χρήστη στο μενού ειδήσεων του site αυτού.
2) Σε διαφορετική περίπτωση, ο χρήστης παραπέμπεται στον προεπιλεγμένο περιηγητή της συσκευής.

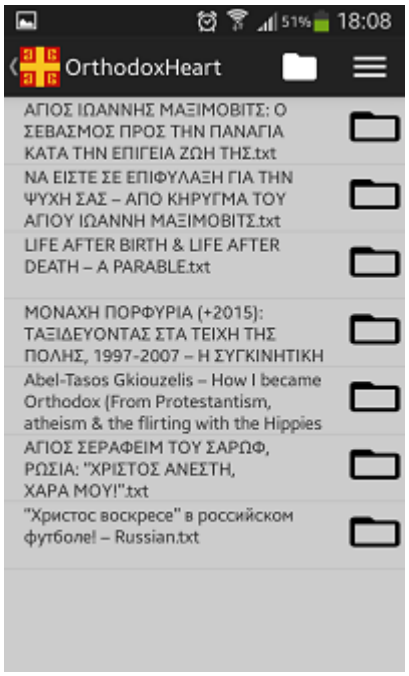
Στην εικόνα 22 εικονίζεται το Action Bar το οποίο είναι κοινό σε όλη την εφαρμογή. Όπως φαίνεται δίπλα από το εικονίδιο της εφαρμογής υπάρχει ένα βελάκι που σε περίπτωση επιλογής του, ο χρήστης επιστρέφει στο αρχικό μενού. Επίσης δίνεται η δυνατότητα μέσω της μπάρας αυτής κάθε χρήστης να πλοηγηθεί στις επιμέρους δυνατότητες της εφαρμογής.



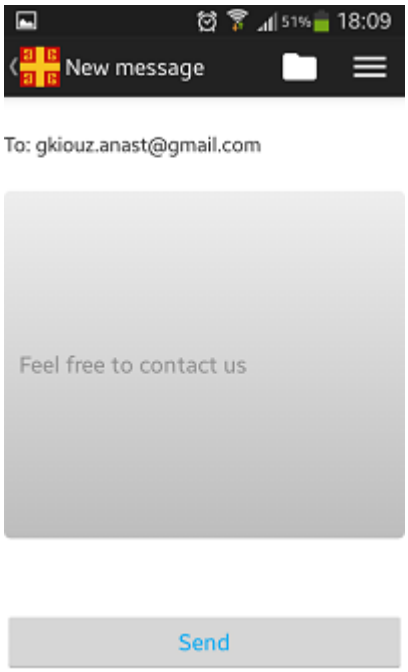

*Εικόνα 22: Action Bar*

Οι επιμέρους δυνατότητες της εφαρμογής παρουσιάζονται παρακάτω με τις εικόνες 23,24,25.

Στην εικόνα 23 βλέπουμε μία από τις δυνατότητες, που είναι η παρουσίαση των αποθηκευμένων άρθρων. Έτσι δίνεται η δυνατότητα στο χρήστη να έχει πρόσβαση στα άρθρα οποιαδήποτε στιγμή ακόμα και χωρίς σύνδεση στο internet.




*Εικόνα 23: Αποθηκευμένα άρθρα*




*Εικόνα 24: Δημιουργία μηνύματος*

Μία ακόμα επιλογή είναι η δημιουργία μηνύματος, όπως φαίνεται στην εικόνα 24, ώστε να μπορεί να επικοινωνήσει ο χρήστης της εφαρμογής με τον συντάκτη των ειδήσεων.

Και τέλος, ο χρήστης μπορεί να δει πληροφορίες για την εφαρμογή όπως φαίνεται παρακάτω στην εικόνα 25.


*Εικόνα 25: Πληροφορίες*

## 5.3 Μελλοντικές επεκτάσεις

Η εφαρμογή θα μπορεί στο μέλλον να αποθηκεύει τα δεδομένα σε έναν Content Provider. Αυτό θα διευκολύνει μία τυχόν αναζήτηση. Επίσης αν τυχόν δημιουργηθούν άλλες δυνατότητες στην εφαρμογή θα μπορούσε να δημιουργηθεί ως κεντρική σελίδα ένα κεντρικό μενού όπου ο χρήστης θα μπορεί να πλοηγηθεί ευκολότερα στις δυνατότητες της εφαρμογής. Για παράδειγμα, θα μπορούσε η εφαρμογή να παρέχει την δυνατότητα στους χρήστες να βρουν διαθέσιμα μέρη και ώρες για εξομολόγηση με την απεικόνισή τους σε χάρτες Google. Κατά συνέπεια θα δημιουργηθεί το κεντρικό μενού με επιλογές: 1.Παρουσίαση ειδήσεων, 2.Ώρες εξομολόγησης κ.α. Ακόμα μία μελλοντική επέκταση θα μπορούσε να είναι η βαθμολόγηση των άρθρων από τους χρήστες με ίσως κάποιου είδους αυθεντικοποίηση. Επίσης θα μπορούσε να δημιουργηθεί και μια αυτόματη ενημέρωση που θα ειδοποιεί τους χρήστες για την προσθήκη νέων ειδήσεων. Αυτό θα ήταν ενδιαφέρον να υλοποιηθεί με την τεχνολογία GCM(Google Cloud Messaging) όπου στέλνει δεδομένα από το διακομιστή μας στις συσκευές των χρηστών μας, και να λαμβάνει μηνύματα από συσκευές χρησιμοποιώντας την ίδια σύνδεση. Και καθώς το Android γίνεται πιο συναρπαστικό μέρα με τη μέρα πολλές μελλοντικές επεκτάσεις θα μπορούν να δημιουργηθούν!

# Παράρτημα (ο κώδικας)

```
package gr.teiep.orthodoxheart.ops;


import gr.teiep.orthodoxheart.provider.ArticleData;
import gr.teiep.orthodoxheart.provider.ArticleData.ArticleElement;
import gr.teiep.orthodoxheart.provider.WebServiceProxy;
import gr.teiep.orthodoxheart.view.ArticleActivity;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStreamWriter;
import java.io.Writer;
import java.lang.ref.WeakReference;
import java.util.ArrayList;

import retrofit.Callback;
import retrofit.RestAdapter;
import retrofit.RetrofitError;
import retrofit.client.Response;
import android.app.Activity;
import android.os.Environment;
import android.util.Log;



public class ArticleOps   implements Callback<ArticleData> {


	private WebServiceProxy mWebServiceProxy;

	private WeakReference<ArticleActivity> mActivity;

	private ArrayList<ArticleElement> mResults;


	public void onConfiguration(Activity activity) {

		mActivity = new WeakReference<>((ArticleActivity) activity);
	}

	public void downloadArticle(String fileName, StringBuilder sb) {

		File dir = new File(
				Environment.getExternalStorageDirectory(),
				"/OrthodoxHeartArticles"
		);

		try{
```

```
                if (!dir.exists())
                        if(!dir.mkdir())
                                mActivity
                                .get()
                                .makeToast(
                                                "Make sure you have memory " +
                                                "space in sdCard."
                                );

                File myfile = new File(
                        Environment.getExternalStorageDirectory(),
                        "/OrthodoxHeartArticles/"+ fileName + ".txt");

                Writer out = new BufferedWriter(new OutputStreamWriter(
                        new FileOutputStream(myfile),"UTF8"));

                out.append(sb);

                out.flush();
                out.close();

                mActivity.get().makeToast(
                                "The article was saved successfully."
        );
        }
        catch (IOException e) {
                e.printStackTrace();
                mActivity.get().makeToast(
                                "We 're sorry, an error ocurrued"
        );
        }

  }

  @SuppressWarnings("deprecation")
private void handleResults(ArticleData results,
        String reason) {


        mResults =
                        new ArrayList<ArticleElement>(
                                        results.getmWebPages()
        );
        int i,j;
        for(i=0;i<mResults.size();i++)
        {
                for(j=0;j<mResults.size()-1;j++)
                {
                        ArticleElement a1=mResults.get(j);
                        ArticleElement a2=mResults.get(j+1);
                        if(a1.geteTurn()>a2.geteTurn())
                        {
                                mResults.set(j, a2);
                                mResults.set(j+1, a1);
                        }
```

```
            }
        }

        mActivity.get().displayResults(mResults,
           reason);

    }

    public void findArticleContet(String selectedArticle) {

        RestAdapter restadapter = new RestAdapter
                            .Builder()
                    .setEndpoint(
                                mWebServiceProxy
                                .sOrthodox_Heart_Service_URL_Retro
                                )
                            .build();

             mWebServiceProxy =
                            restadapter.create(WebServiceProxy.class);

           mWebServiceProxy.getArticleData(selectedArticle, this);
    }

    /**
     * Called by Retrofit for handling error.
     */
    @Override
    public void failure(RetrofitError error) {
        Log.v("",
              "Retrofit failure");

    }

    /**
     * Called by Retrofit for handling success result.
     */
    @Override
    public void success(ArticleData results,
                        Response response) {
        Log.v("",
              "Retrofit success");


        handleResults(results,
                "");
    }
}
```

```
package gr.teiep.orthodoxheart.ops;


import gr.teiep.orthodoxheart.provider.ArticleData;
import gr.teiep.orthodoxheart.provider.ArticleData.ArticleElement;
import gr.teiep.orthodoxheart.provider.WebServiceProxy;
import gr.teiep.orthodoxheart.view.ArticleActivity;
import gr.teiep.orthodoxheart.view.FilesActivity;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStreamWriter;
import java.io.Writer;
import java.lang.ref.WeakReference;
import java.util.ArrayList;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.StatusLine;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.DefaultHttpClient;


import retrofit.Callback;
import retrofit.RestAdapter;
import retrofit.RetrofitError;
import retrofit.client.Response;
import android.app.Activity;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.Environment;
import android.util.Log;



public class FilesOps {



	private WeakReference<FilesActivity> mActivity;

	private ArrayList<String> mResults;



	public void onConfiguration(Activity activity) {

		mActivity = new WeakReference<>((FilesActivity) activity);
```

```
}

public void getFiles() {
       ArrayList<String> myFiles = new ArrayList<String>();
       File dir = new File(
                     Environment.getExternalStorageDirectory(),
                           "/OrthodoxHeartArticles"
       );

       File[] files = dir.listFiles();
       if (files.length == 0)
              mActivity.get().displayResults(myFiles,
                                                  "No files were found");
           else {
               for (int i=0; i<files.length; i++)
                   myFiles.add(files[i].getName());
           }
       mActivity.get().displayResults(myFiles,
          "");

  }

public void getArticleContent(String articleName) {

       File file = new File(
                     Environment.getExternalStorageDirectory(),
                     "/OrthodoxHeartArticles/" + articleName
       );

       StringBuilder articleContent = new StringBuilder();

       try {
           BufferedReader br =
                     new BufferedReader(new FileReader(file));
           String line;

           while ((line = br.readLine()) != null) {
              articleContent.append(line);
           }
           br.close();

           mActivity.get().setArticleContent(articleContent);
       }
       catch (IOException e) {
           //You'll need to add proper error handling here
       }
}

public boolean deleteArticleContent(String articleName) {

       File file =
                     new File(Environment.getExternalStorageDirectory(),
                                  "/OrthodoxHeartArticles/" + articleName
       );
```

```
            return file.delete();
        }
}


package gr.teiep.orthodoxheart.ops;

import gr.teiep.orthodoxheart.provider.Database;
import gr.teiep.orthodoxheart.provider.MainIndexData;
import gr.teiep.orthodoxheart.provider.MainIndexData.WebPage;
import gr.teiep.orthodoxheart.provider.WebServiceProxy;
import gr.teiep.orthodoxheart.view.MainActivity;

import java.lang.ref.WeakReference;
import java.util.ArrayList;

import retrofit.Callback;
import retrofit.RestAdapter;
import retrofit.RetrofitError;
import retrofit.client.Response;
import android.app.Activity;
import android.util.Log;



public class HttpOps implements Callback<MainIndexData> {


        private WeakReference<MainActivity> mActivity;

        private ArrayList<WebPage> mResults;

        private WebServiceProxy mWebServiceProxy;

        Database db = null;

        private static int version = 1;


        public void onConfiguration(Activity activity) {

                mActivity    = new WeakReference<>((MainActivity) activity);
                db                  = new Database(activity, null, version);

                mWebServiceProxy = new RestAdapter
                                        .Builder()
                                .setEndpoint(
                                                mWebServiceProxy
                                                .sOrthodox_Heart_Service_URL_Retro
                                                )
                                        .build()
                                        .create(WebServiceProxy.class);
        }

         public void findWebSites() {
```

```
        db.clearData();

       mWebServiceProxy.getMainIndexData(this);
    }

public void doBulkInsertIntoWebPages
               (ArrayList<WebPage> arrayList) {
   db.bulkInsertIntoWebPages(arrayList);
}

public ArrayList<WebPage> selectAllWebPages() {
   return db.getResults();
}

public void insertIntoWebPages( int id,
                                                  String name,
                                                  String url,
                                                  String country)
{
   db.insertIntoWebPages(id, name, url, country);
}

public boolean isOrthoxHeartSite() {

   return db.doesWebPageExist(
               mActivity
               .get()
               .preSelectedWebPage);
}

/**
 * Called by Retrofit for handling error.
 */
@Override
public void failure(RetrofitError error) {
    Log.v("",
          "Retrofit failure");
}

/**
 * Called by Retrofit for handling success result.
 */
@Override
public void success(MainIndexData results,
                    Response response) {
    Log.v("",
          "Retrofit success");


    handleResults(results,
            "");
}

private void handleResults(MainIndexData results,
```

```
            String reason) {
        mResults = results.getmWebPages();

        doBulkInsertIntoWebPages(results.getmWebPages());

                // Try to display the results.
                mActivity.get().displayResults(results.getmWebPages(),
                                               reason);
        }
}


package gr.teiep.orthodoxheart.ops;

import gr.teiep.orthodoxheart.provider.WebPageData;
import gr.teiep.orthodoxheart.provider.WebPageData.WebPageArticle;
import gr.teiep.orthodoxheart.provider.WebServiceProxy;
import gr.teiep.orthodoxheart.view.WebPageActivity;

import java.lang.ref.WeakReference;
import java.util.ArrayList;

import retrofit.Callback;
import retrofit.RestAdapter;
import retrofit.RetrofitError;
import retrofit.client.Response;
import android.app.Activity;
import android.util.Log;



public class WebPageOps
                implements Callback<WebPageData> {


        private WeakReference<WebPageActivity> mActivity;

        private ArrayList<WebPageArticle> mResults;

        private WebServiceProxy mWebServiceProxy;

        private boolean hasMoreArticle;



        public void onConfiguration(Activity activity) {

                mActivity    =
                             new WeakReference<>((WebPageActivity) activity);
                mResults    = new ArrayList<WebPageArticle>();
                hasMoreArticle = true;
        }


        public void findWebPageContet(String selectedWebPage) {
```

```
        RestAdapter restadapter = new RestAdapter
                    .Builder()
              .setEndpoint(
                          mWebServiceProxy
                          .sOrthodox_Heart_Service_URL_Retro)
                    .build();

        mWebServiceProxy = restadapter.create(WebServiceProxy.class);

      mWebServiceProxy.getWebPageData(selectedWebPage, this);
  }

/**
 * Called by Retrofit for handling error.
 */
@Override
public void failure(RetrofitError error) {
    Log.v("",
          "Retrofit failure");

}

/**
 * Called by Retrofit for handling success result.
 */
@Override
public void success(WebPageData results,
                    Response response) {
    Log.v("",
          "Retrofit success");


    handleResults(results,
            "Retrofit Succeses");
}

private void handleResults(WebPageData results,
        String reason) {

  mResults.addAll(results.getmWebPageArticles());

  if(results.getmNextPage()!=null)
  {
        mActivity
              .get()
              .setMoreAriclesPage(results.getmNextPage());
  }
  else {

        turnHasMoreArticleToFalse();
  }

  // Try to display the results.
  mActivity.get().displayTitle(results.getmSiteTitle());
```

```
            // Try to display the results.
            mActivity.get().displayResults(mResults,
                                           reason);
      }

    private void turnHasMoreArticleToFalse() {
      mActivity.get().setMoreAriclesPage("");
    }
}
```

```
package gr.teiep.orthodoxheart.provider;

import java.util.ArrayList;

import com.google.gson.annotations.SerializedName;


/**
 * Αυτή η κλάση είναι ένα Plain Old Java Object. Χρησιμοποιείται για
 * ταιρίαζει αυτόματα τα δεδομένα που λαμβάνονται από τον διακομιστή.
 * Για παράδειγμα μια κλήση στο
 * http://orthodoxismyheart.esy.es/MainServise%20.php
 * που θα επεστρέπει τα παρακάτω δεδομένα Json
 *
 * {"articleElements":[
 *                                     {"turn":0,
 *                                      "content":"",
 *
"src":"https:\/\/gkiouzelis.files.wordpress.com\",
 *                                      "element":"img"},
 *                                     {"turn":1,
 *
"content":"\u79c1\u306f\u65e5\u672c\u304c\u5927\u597d\u304d",
 *                                      "src":"",
 *                                      "element":"p"}
 *                                     ]
 * }
 *
 * με την βοήθεια της  βιβλιοθήκη Retrofit θα τα μετατραπούν
 * αυτόματα σε τα Json δεδομένα σε αντικείμενα WebPageData.
 */
public class ArticleData {

      /*
     * Αυτά τα πεδία αποθηκεύου την κατάσταση του ArticleData.
     * Χρησιμοποιούνε το @SerializedName annotation για να
     * καθορίσουμε τα ονόματα ανάμεσα στα Json ονόματα και των
     * πεδίων στην κλάση.  Εάν τα ονόματα αυτών των πεδίων ήταν
     * ίδια με τα ονόματα των Json δεν θα χρειαζόταν να
     * χρησιμοποιήσουμε αυτό το annotation.
```

```
 */
@SerializedName("articleElements")
private ArrayList<ArticleElement> mWebPages =
   new ArrayList<ArticleElement>();


public ArticleData(ArrayList<ArticleElement> mWebPages) {
         super();
         this.mWebPages = mWebPages;
   }


   public ArrayList<ArticleElement> getmWebPages() {
         return mWebPages;
   }


   public void setmWebPages(
               ArrayList<ArticleElement> mWebPages
   )
   {
         this.mWebPages = mWebPages;
   }

   /**
 * Inner class representing article html elements
 */
public static class ArticleElement {


   @SerializedName("turn")
    private int eTurn;
   @SerializedName("content")
    private String eContent;
   @SerializedName("element")
    private String eElement;
   @SerializedName("src")
    private String eBm;


   public ArticleElement(    int eTurn,
                                          String eContent,
                                          String eElement,
                                          String eBm) {
         super();
         this.eTurn = eTurn;
         this.eContent = eContent;
         this.eElement = eElement;
         this.eBm = eBm;
   }

         public ArticleElement(    int eTurn,
                                                String eContent,
                                                String eElement) {
               super();
```

```
                this.eTurn = eTurn;
                this.eContent = eContent;
                this.eElement = eElement;
        }


        /*     setters & getters*/

        public int geteTurn() {
                return eTurn;
        }
        public void seteTurn(int eTurn) {
                this.eTurn = eTurn;
        }
        public String geteContent() {
                return eContent;
        }
        public void seteContent(String eContent) {
                this.eContent = eContent;
        }
        public String geteElement() {
                return eElement;
        }
        public void seteElement(String eElement) {
                this.eElement = eElement;
        }
        public String geteBm() {
                return eBm;
        }
        public void seteBm(String eBm) {
                this.eBm = eBm;
        }

    }
}
```

```
package gr.teiep.orthodoxheart.provider;

import gr.teiep.orthodoxheart.provider.MainIndexData.WebPage;

import java.util.ArrayList;

import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteDatabase.CursorFactory;
import android.database.sqlite.SQLiteOpenHelper;

public class Database extends SQLiteOpenHelper{

    private Context mcontext;
    private SQLiteDatabase database;
```

```
public Database(Context context, CursorFactory factory,
            int version) {

      super(context,
                  DatabaseNamesContract.DATABASE_NAME,
                  factory,
                  version);

      mcontext=context;
      database=this.getWritableDatabase();
}

@Override
public void onCreate(SQLiteDatabase db) {

      final String SQL_CREATE_WEB_PAGES_TABLE =
            "CREATE TABLE "
            + DatabaseNamesContract.TABLE_WEB_PAGES
            + " ("
            + DatabaseNamesContract.COLUMN_ID
            + " INTEGER PRIMARY KEY, "
            + DatabaseNamesContract.COLUMN_NAME
            + " TEXT NOT NULL, "
            + DatabaseNamesContract.COLUMN_URL
            + " TEXT NOT NULL, "
            + DatabaseNamesContract.COLUMN_COUNTRY
            + " TEXT NOT NULL"
            + " );";
      db.execSQL(SQL_CREATE_WEB_PAGES_TABLE);

}

public ArrayList<WebPage> getResults()
{
      ArrayList<WebPage> x=new ArrayList<WebPage>();
      Cursor cursor=database.rawQuery("select * "
                  +"from " + DatabaseNamesContract.TABLE_WEB_PAGES,
                  null);
      if(cursor.getCount()==0)
      {
            cursor.close();
            return null;
      }
      int idIndex=
                  cursor
                  .getColumnIndex(DatabaseNamesContract.COLUMN_ID);
      int nameIndex =
                  cursor
                  .getColumnIndex(DatabaseNamesContract.COLUMN_NAME);
      int urlIndex =
                  cursor
                  .getColumnIndex(DatabaseNamesContract.COLUMN_URL);
      int countryIndex =
                  cursor
                  .getColumnIndex(
```

```
                                        DatabaseNamesContract.COLUMN_COUNTRY
                );
                cursor.moveToFirst();
                do
                {
                        WebPage webSite;
                        webSite=new WebPage(
                                        cursor.getInt(idIndex),
                                        cursor.getString(nameIndex),
                                        cursor.getString(urlIndex),
                                        cursor.getString(countryIndex));

                        x.add(webSite);
                }while(cursor.moveToNext());
                cursor.close();
                return x;

        }

        public void insertIntoWebPages( int id,
                                                                String name,
                                                                String url,
                                                                String country)
        {
                database.execSQL("insert into "
                        + DatabaseNamesContract.TABLE_WEB_PAGES
                        + " values("
                        + id + ","
                        +"'" + name + "','" + url + "','" + country + "')");
        }

        public boolean doesWebPageExist( String url)
        {
                Cursor cursor=database.rawQuery("select * from "
                        + DatabaseNamesContract.TABLE_WEB_PAGES
                        + " where "+DatabaseNamesContract.COLUMN_URL
                        +"='" + url + "';",null);
                if(cursor.getCount()==0)
                {
                        cursor.close();
                        return false;
                }
                else
                        return true;
        }

//      public void updateWebPages( int id,
//                                                      String name,
//                                                      String url,
//                                                      String country)
//      {
//              database.execSQL("update "
//                      + DatabaseNamesContract.TABLE_WEB_PAGES
//                      + " set '" + DatabaseNamesContract.COLUMN_IS_CHECKED
//                      + "'=" +country
```

```
//                  + " where "+DatabaseNamesContract.COLUMN_ID
//                  +"=" + id + ";");
//     }

    public void bulkInsertIntoWebPages(ArrayList<WebPage> arrayList) {

       // Begins a transaction in EXCLUSIVE mode.
       database.beginTransaction();
       int returnCount = 0;

             try {

                   for (WebPage listItem : arrayList) {
                   insertIntoWebPages(
                            listItem.getSiteId(),
                            listItem.getSiteName(),
                            listItem.getSiteUrl(),
                            listItem.getSiteCountry());
                   returnCount++;
               }

               // Marks the current transaction as successful.
               database.setTransactionSuccessful();
             } finally {
               // End a transaction.
               database.endTransaction();
             }
       }


       @Override
       public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
             db.execSQL("drop table " + DatabaseNamesContract.TABLE_WEB_PAGES );
             onCreate(db);
       }

       public void clearData()
       {
             database.execSQL("delete from "+ DatabaseNamesContract.TABLE_WEB_PAGES);
       }

       public static final class DatabaseNamesContract {

           public static final String DATABASE_NAME =
                   "orthdxheart.db";

             /**
              * Name of the database table.
              */
             public static final String TABLE_WEB_PAGES =
                     "web_pages_table";


             public static final String COLUMN_ID = "id";
             public static final String COLUMN_NAME = "name";
```

```
            public static final String COLUMN_URL = "url";
            public static final String COLUMN_COUNTRY = "country";

    }

}
```

```
package gr.teiep.orthodoxheart.provider;

import gr.teiep.orthodoxheart.provider.WebPageData.WebPageArticle;

import java.util.ArrayList;

import com.google.gson.annotations.SerializedName;

import android.graphics.Bitmap;

/**
 * Αυτή η κλάση είναι ένα Plain Old Java Object. Χρησιμοποιείται για
 * ταιρίαζει αυτόματα τα δεδομένα που λαμβάνονται από τον διακομιστή.
 * Για παράδειγμα μια κλήση στο
 * http://orthodoxismyheart.esy.es/ArticleContentServise.php
 * ?selectedArticle=
 * https://gkiouzelis.wordpress.com/2015/09/15/japan-of-my-heart/
 * που θα επεστρέπει τα παρακάτω δεδομένα Json
 *
 * {"webPages":[{
 *                              "id":"1",
 *                              "name":"france of my heart",
 *                              "url":"http:\/\/franceofmyheart.wordpress.com",
 *                              "country":"global"
 *                         },
 *                         {      "id":"2",
 *                              "name":"gkiouzelis",
 *                              "url":"https:\/\/gkiouzelis.wordpress.com\/",
 *                              "country":"global"},
 *                         }
 * ]}
 *
 *
 * με την βοήθεια της  βιβλιοθήκη Retrofit θα τα μετατραπούν
 * αυτόματα σε τα Json δεδομένα σε αντικείμενα WebPageData.
 */
public class MainIndexData {

      /*
     * Αυτά τα πεδία αποθηκεύου την κατάσταση του MainIndexData.
     * Χρησιμοποιούνε το @SerializedName annotation για να
     * καθορίσουμε τα ονόματα ανάμεσα στα Json ονόματα και των
     * πεδίων στην κλάση.  Εάν τα ονόματα αυτών των πεδίων ήταν
     * ίδια με τα ονόματα των Json δεν θα χρειαζόταν να
     * χρησιμοποιήσουμε αυτό το annotation.
     */
```

```
@SerializedName("webPages")
private ArrayList<WebPage> mWebPages = new ArrayList<WebPage>();


public MainIndexData(ArrayList<WebPage> mWebPages) {
        super();
        this.mWebPages = mWebPages;
  }

  public ArrayList<WebPage> getmWebPages() {
        return mWebPages;
  }

  public void setmWebPages(ArrayList<WebPage> mWebPages) {
        this.mWebPages = mWebPages;
  }


  /**
 * Inner class representing webSite fields
 */
public static class WebPage {


  @SerializedName("id")
    private int siteId;
  @SerializedName("name")
    private String siteName;
  @SerializedName("url")
    private String siteUrl;
  @SerializedName("country")
    private String siteCountry;


        public WebPage(int siteId, String siteName, String siteUrl,
                    String siteCountry) {
              super();
              this.siteId = siteId;
              this.siteName = siteName;
              this.siteUrl = siteUrl;
              this.siteCountry = siteCountry;
        }

        public int getSiteId() {
              return siteId;
        }

        public void setSiteId(int siteId) {
              this.siteId = siteId;
        }

        public String getSiteName() {
              return siteName;
        }
```

```
	public void setSiteName(String siteName) {
		this.siteName = siteName;
	}

	public String getSiteUrl() {
		return siteUrl;
	}

	public void setSiteUrl(String siteUrl) {
		this.siteUrl = siteUrl;
	}

	public String getSiteCountry() {
		return siteCountry;
	}

	public void setSiteCountry(String siteCountry) {
		this.siteCountry = siteCountry;
	}
    }
}

package gr.teiep.orthodoxheart.provider;

import java.util.ArrayList;
import java.util.List;
import java.util.List;

import android.graphics.Bitmap;
import android.os.Parcel;
import android.os.Parcelable;

import com.google.gson.annotations.SerializedName;

/**
 * Αυτή η κλάση είναι ένα Plain Old Java Object. Χρησιμοποιείται για
 * ταιριάζει αυτόματα τα δεδομένα που λαμβάνονται από τον διακομιστή.
 * Για παράδειγμα μια κλήση στο
 * http://orthodoxismyheart.esy.es/PageContentServise.php
 * ?selectedWebPage=https://gkiouzelis.wordpress.com/
 * που θα επεστρέπει τα παρακάτω δεδομένα Json
 *
 * {"authorTitle":"ABEL-TASOS GKIOUZELIS:
 * MY HEART SITES ST JOHN MAXIMOVITCH& ST BRIGID OF KILDARE",
 *  "nextPage":"https:\/\/gkiouzelis.wordpress.com\....",
 *  "article":[
 *                  {"header_title":"I LOVE JAPAN  JAPAN OF MY HEART",
 *                   "header_url":"https:\/\/gkiouzelis.wordpress.com..",
 *                   "img":"https:\/\/gkiouzelis.files.wordpress.com...",
 *                   "context":""}
 *             ]
 * }
 *
 * με την βοήθεια της  βιβλιοθήκη Retrofit θα τα μετατραπούν
 * αυτόματα σε τα Json δεδομένα σε αντικείμενα WebPageData.
```

```
 */
public class WebPageData  {
	/*
     * Αυτά τα πεδία αποθηκεύου την κατάσταση του WebPageData.
     * Χρησιμοποιούνε το @SerializedName annotation για να
     * καθορίσουμε τα ονόματα ανάμεσα στα Json ονόματα και των
     * πεδίων στην κλάση.  Εάν τα ονόματα αυτών των πεδίων ήταν
     * ίδια με τα ονόματα των Json δεν θα χρειαζόταν να
     * χρησιμοποιήσουμε αυτό το annotation.
     */

	@SerializedName("nextPage")
    private String mNextPage;

	@SerializedName("authorTitle")
    private String mSiteTitle;


	@SerializedName("article")
    private ArrayList<WebPageArticle> mWebPageArticles =
	new ArrayList<WebPageArticle>();


	public WebPageData(String mNextPage,
			ArrayList<WebPageArticle> mWebPageArticles) {
		super();
		this.mNextPage = mNextPage;
		this.mWebPageArticles = mWebPageArticles;
	}

	public String getmNextPage() {
		return mNextPage;
	}

	public void setmNextPage(String mNextPage) {
		this.mNextPage = mNextPage;
	}

	public String getmSiteTitle() {
		return mSiteTitle;
	}

	public void setmSiteTitle(String mSiteTitle) {
		this.mSiteTitle = mSiteTitle;
	}

	public ArrayList<WebPageArticle> getmWebPageArticles() {
		return mWebPageArticles;
	}

	public void setmWebPageArticles(
			ArrayList<WebPageArticle> mWebPageArticles
	) {
		this.mWebPageArticles = mWebPageArticles;
	}
```

```
    /**
     * Inner class representing the articles in the webSite
     */
    public static class WebPageArticle {
      @SerializedName("header_title")
        private String mTitle;
      @SerializedName("header_url")
        private String mUrl;
      @SerializedName("img")
        private String mImg;

      private Bitmap mBitmap;


            public Bitmap getmBitmap() {
                  return mBitmap;
            }


            public void setmBitmap(Bitmap mBitmap) {
                  this.mBitmap = mBitmap;
            }


            public String getmTitle() {
                  return mTitle;
            }


            public void setmTitle(String mTitle) {
                  this.mTitle = mTitle;
            }


            public String getmUrl() {
                  return mUrl;
            }


            public void setmUrl(String mUrl) {
                  this.mUrl = mUrl;
            }


            public String getmImg() {
                  return mImg;
            }


            public void setmImg(String mImg) {
                  this.mImg = mImg;
            }
      }
}
```

```java
package gr.teiep.orthodoxheart.provider;

import gr.teiep.orthodoxheart.provider.ArticleData.ArticleElement;
import retrofit.Callback;
import retrofit.http.GET;
import retrofit.http.Query;

/**
 * Interface defining methods used by RetroFit to access current
 * weather data from the Weather Service web service.
 */
public interface WebServiceProxy {
    /**
     * URL to the Web Search web service to use with the Retrofit
     * service.
     */
    final String sOrthodox_Heart_Service_URL_Retro =
        "http://orthodoxismyheart.esy.es";



    /**
     * Method used to query the Orthodox heart web service for the
     * page content.  The annotations enable
     * Retrofit to convert the selectedWebPage parameter into an HTTP
     * request, which would look something like this:
     * http://orthodoxismyheart.esy.es
     * ?selectedWebPage=selectedWebPage/PageContentServise.php
     *
     * @param selectedWebPage
     * @param callback
     * @return WebPageData
     */
    @GET("/PageContentServise.php")
    public void getWebPageData
        (@Query("selectedWebPage") String selectedWebSite,
         Callback<WebPageData> callback);

    /**
     * Method used to query the Orthodox heart web service for the
     * MainIndex which is the stored orthodox heart web sites.  The
     * annotations enable Retrofit to convert the @a location
     * parameter into an HTTP request, which would look something
     * like this:
     * http://orthodoxismyheart.esy.es/MainServise%20.php
     *
     * @param callback
     * @return MainIndexData
     */
    @GET("/MainServise%20.php")
    public void getMainIndexData (Callback<MainIndexData> callback);
```

```
    /**
     * Method used to query the Orthodox heart web service for the
     * article content.  The annotations enable
     * Retrofit to convert the selectedArticle parameter into an HTTP
     * request, which would look something like this:
     * http://orthodoxismyheart.esy.es/ArticleContentServise.php?
     * selectedArticle=selectedArticle
     *
     * @param selectedWebPage
     * @param callback
     * @return WebPageData
     */
    @GET("/ArticleContentServise.php")
    public void getArticleData
             (@Query("selectedArticle") String selectedArticle,
         Callback<ArticleData> callback);
}
```

```
package gr.teiep.orthodoxheart.utils;

import gr.teiep.orthodoxheart.R;
import gr.teiep.orthodoxheart.provider.MainIndexData.WebPage;

import java.util.ArrayList;

import android.app.Activity;
import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Color;
import android.text.Html;
import android.view.Display;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.TextView;

public class FilesListAdapter extends ArrayAdapter{
       public ArrayList<String> item=null;
       Bitmap flagBm;
       int i=0;

       Context mcontext=null;

       public FilesListAdapter(Context context,
                    int textViewResourceId,
                    ArrayList<String> t
       )
       {
           super(context, textViewResourceId, t);
           mcontext=context;
```

```
    item= new ArrayList<String>(t);
}


public View getView(final int index,
                                View convertView,
                                ViewGroup viewGroup
)
{
        Display display =
                    ((Activity) mcontext).getWindowManager()
                                                        .getDefaultDisplay();
        int width = display.getWidth();
        int height = display.getHeight();

        LinearLayout l=new LinearLayout(mcontext);


        TextView t=new TextView(mcontext);
        l.addView(t);
        LinearLayout.LayoutParams tparams =
                    (LinearLayout.LayoutParams)t.getLayoutParams();
        tparams.width = 75 * width/100;
        tparams.height=height/10;
        tparams.leftMargin= 5 * width/100;
        t.setLayoutParams(tparams);

        t.setText(Html.fromHtml(item.get(index).toString()));


        ImageView arrowImg=new ImageView(mcontext);
        l.addView(arrowImg);
        Bitmap arrowBm = BitmapFactory.decodeResource(
                                mcontext.getResources(),
                                R.drawable.ic_folder_open_black_48dp
                                );
        double arrowImgRatio=arrowBm.getWidth()*1.0/arrowBm.getHeight();
        LinearLayout.LayoutParams arrawImParams =
                    (LinearLayout.LayoutParams)arrowImg.getLayoutParams();
        arrawImParams.leftMargin = 5 * width/100;
        arrawImParams.width      = 15 * width/100;
        arrawImParams.height     = 15 * width/100;

        arrowBm = arrowBm.createScaledBitmap(
                    arrowBm, 20*width/100,
                    arrawImParams.height,
                    true
                    );
        arrowImg.setImageBitmap(arrowBm);
        l.setBackgroundColor(Color.LTGRAY);
        return l;
 }

 public ArrayList<String> getItem(){
        return item;
```

```
        }
}


package gr.teiep.orthodoxheart.utils;



import gr.teiep.orthodoxheart.R;
import gr.teiep.orthodoxheart.view.FilesActivity;
import gr.teiep.orthodoxheart.view.SendMailActivity;
import gr.teiep.orthodoxheart.view.TextDisplayActivity;
import android.app.ActionBar;
import android.app.Activity;
import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Typeface;
import android.graphics.drawable.ColorDrawable;
import android.os.Bundle;
import android.provider.Settings;
import android.util.DisplayMetrics;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;


public class GenericActivity
       extends Activity {

       private Context mActivity;
       public static int WIDTH   = 0;
       public static int HEIGHT  = 0;

       public static Typeface    TYPE = null;
       public static int         FONT_SIZE;
       Intent i = null;


    public void onCreate(Bundle savedInstanceState, Context mContext) {
        // Call up to the super class.
        super.onCreate(savedInstanceState);
        mActivity=mContext;
        final DisplayMetrics metrics =
                  getResources().getDisplayMetrics();
        WIDTH = metrics.widthPixels;
        HEIGHT = metrics.heightPixels;
        FONT_SIZE=26;
           if(HEIGHT<976)  FONT_SIZE=16;
           TYPE = Typeface.createFromAsset(
                     mContext.getAssets(),
                     "fonts/ubuntu.ttf"
                     );
           ActionBar mActionBar = getActionBar();
```

```
//          showNoConnectionDialog(mActivity);
//          mActionBar.setBackgroundDrawable(
//                  new ColorDrawable(0xff888888)
//          );
    }

        @Override
        public boolean onCreateOptionsMenu(Menu menu) {

                    // Get a reference to the MenuInflater
                    MenuInflater inflater = getMenuInflater();

                    // Inflate the menu using activity_menu.xml
                    inflater.inflate(R.menu.activity_menu, menu);

                    // Return true to display the menu
                    return true;
        }

        @Override
        public boolean onOptionsItemSelected(MenuItem item) {

              switch (item.getItemId()) {
              case R.id.folder_open:

                    i = new Intent(mActivity, FilesActivity.class);
                    startActivity(i);
                    // return value true indicates that the menu click has
                    // been handled
                    return true;

              case R.id.form1:

                    i = new Intent(mActivity, SendMailActivity.class);
                    startActivity(i);
                    // return value true indicates that the menu click has
                    // been handled
                    return true;

                    case R.id.form2:
                    i = new Intent(mActivity, TextDisplayActivity.class);
                    startActivity(i);
                    // return value true indicates that the menu click has
                    // been handled
                    return true;

                    default:
                          return super.onOptionsItemSelected(item);
                    }
        }

        public static void showNoConnectionDialog(Context ctx1) {
          final Context ctx = ctx1;
          AlertDialog.Builder builder = new AlertDialog.Builder(ctx);
          builder.setCancelable(true);
```

```
        builder.setMessage(
                    "Go to Intenet Settings or read your downloads"
        );
        builder.setTitle(
                    "Connect to the internet?"
        );
        builder.setPositiveButton("Connect",
                    new DialogInterface.OnClickListener()
        {
            public void onClick(DialogInterface dialog,
                    int which
            )
            {
                ctx.startActivity(
                        new Intent(Settings.ACTION_WIRELESS_SETTINGS)
                );
            }
        });
        builder.setNegativeButton(
                    "Open folder",
                    new DialogInterface.OnClickListener()
        {
            public void onClick(DialogInterface dialog, int which) {
                return;
            }
        });
        builder
        .setOnCancelListener(new DialogInterface.OnCancelListener()
        {
            public void onCancel(DialogInterface dialog) {
                return;
            }
        });

        builder.show();
    }

}


package gr.teiep.orthodoxheart.utils;



import gr.teiep.orthodoxheart.R;
import gr.teiep.orthodoxheart.view.FilesActivity;
import gr.teiep.orthodoxheart.view.SendMailActivity;
import gr.teiep.orthodoxheart.view.TextDisplayActivity;
import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.graphics.Typeface;
import android.os.Bundle;
import android.view.Display;
import android.view.Menu;
```

```
import android.view.MenuInflater;
import android.view.MenuItem;

import com.google.android.youtube.player.YouTubeBaseActivity;


public class GenericYouTubeActivity
       extends YouTubeBaseActivity {

       private Context mActivity;
       public static int WIDTH   = 0;
       public static int HEIGHT  = 0;

       public static Typeface    TYPE = null;
       public static int         FONT_SIZE;
       Intent i = null;



    public void onCreate(Bundle savedInstanceState, Context mContext)
    {
        // Call up to the super class.
        super.onCreate(savedInstanceState);
        mActivity=mContext;
        Display display =
                   ((Activity) mActivity)    .getWindowManager()
                                                    .getDefaultDisplay();
        WIDTH = display.getWidth();
        HEIGHT = display.getHeight();
        FONT_SIZE=26;
           if(HEIGHT<976)  FONT_SIZE=16;
           TYPE=Typeface.createFromAsset(
                    mContext.getAssets(),
                    "fonts/ubuntu.ttf"
                    );

    }

       @Override
       public boolean onCreateOptionsMenu(Menu menu) {

                    // Get a reference to the MenuInflater
                    MenuInflater inflater = getMenuInflater();

                    // Inflate the menu using activity_menu.xml
                    inflater.inflate(R.menu.activity_menu, menu);

                    // Return true to display the menu
                    return true;
       }

       @Override
       public boolean onOptionsItemSelected(MenuItem item) {
```

```
		switch (item.getItemId()) {
		case R.id.folder_open:

			i = new Intent(mActivity, FilesActivity.class);
			startActivity(i);
			// return value true indicates that the menu click has
			// been handled
			return true;

		case R.id.form1:

			i = new Intent(mActivity, SendMailActivity.class);
			startActivity(i);
			// return value true indicates that the menu click has
			// been handled
			return true;

			case R.id.form2:
			i = new Intent(mActivity, TextDisplayActivity.class);
			startActivity(i);
			// return value true indicates that the menu click has
			// been handled
			return true;

			default:
				return super.onOptionsItemSelected(item);
			}
	}
}


package gr.teiep.orthodoxheart.utils;

import gr.teiep.orthodoxheart.R;
import gr.teiep.orthodoxheart.provider.MainIndexData.WebPage;

import java.util.ArrayList;

import android.app.Activity;
import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Color;
import android.text.Html;
import android.view.Display;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.TextView;

public class MainIndexListAdapter extends ArrayAdapter{

	public ArrayList<WebPage> item=null;
```

```
Bitmap flagBm;
int i=0;

Context mcontext=null;
public MainIndexListAdapter(Context context,
                                                int textViewResourceId,
                                                ArrayList<WebPage> t
)
{
       super(context, textViewResourceId, t);
       mcontext=context;
       item= new ArrayList<WebPage>(t);
}

public View getView(final int index,
                                View convertView,
                                ViewGroup viewGroup
)
{
        Display display =
                     ((Activity) mcontext)     .getWindowManager()

.getDefaultDisplay();
        int width = display.getWidth();
        int height = display.getHeight();

        LinearLayout l=new LinearLayout(mcontext);

        ImageView im=new ImageView(mcontext);
        l.addView(im);
        if(item.get(index).getSiteCountry() != null
                     && item.get(index)
                     .getSiteCountry().equals("global") )
              flagBm = BitmapFactory
                                .decodeResource(mcontext.getResources(),

R.drawable.mb_world);
        else if(item.get(index).getSiteCountry() != null
                     && item.get(index)
                     .getSiteCountry().equals("greece") )
              flagBm = BitmapFactory
                                .decodeResource(mcontext.getResources(),

R.drawable.flagofgreece);
        else if(item.get(index).getSiteCountry() != null
                     && item.get(index)
                     .getSiteCountry().equals("france") )
              flagBm = BitmapFactory
                                .decodeResource(mcontext.getResources(),

R.drawable.flagoffrance);
        else if(item.get(index).getSiteCountry() != null
                     && item.get(index)
                                .getSiteCountry().equals("italy") )
              flagBm = BitmapFactory
```

```
                                .decodeResource(mcontext.getResources(),

R.drawable.flagofitaly);
        else if( item.get(index).getSiteCountry() != null
                        && item.get(index)
                                .getSiteCountry().equals("germany"))
                flagBm = BitmapFactory
                                .decodeResource(mcontext.getResources(),

R.drawable.flagofgermany);
        double ratio = flagBm.getWidth() * 1.0 / flagBm.getHeight();
        LinearLayout.LayoutParams imparams =
                        (LinearLayout.LayoutParams)im.getLayoutParams();
        imparams.width=15*width/100;
        imparams.height=(int)(imparams.width*ratio);
        imparams.leftMargin = 3*width/100;


        flagBm=flagBm.createScaledBitmap(     flagBm,
                                                                15*width/100,

imparams.height,
                                                                true);
        im.setImageBitmap(flagBm);


        TextView t=new TextView(mcontext);
        l.addView(t);
        LinearLayout.LayoutParams tparams =
                        (LinearLayout.LayoutParams)t.getLayoutParams();
        tparams.width = 64 * width/100;
        tparams.height=imparams.height;
        tparams.leftMargin= 3 * width/100;
        t.setLayoutParams(tparams);

        t.setText(Html.fromHtml(item.get(index).getSiteName()));


        ImageView arrowImg=new ImageView(mcontext);
        l.addView(arrowImg);
        Bitmap arrowBm = BitmapFactory.decodeResource(
                                        mcontext.getResources(),
                                        R.drawable.rightarrow
                                        );
        double arrowImgRatio =
                        arrowBm.getWidth() *1.0/arrowBm.getHeight();

        LinearLayout.LayoutParams arrawImParams =
                        (LinearLayout.LayoutParams)arrowImg

.getLayoutParams();
        arrawImParams.width      = 15 * width/100;
        arrawImParams.height     =
                        (int)(imparams.width * arrowImgRatio);
```

```
                arrowBm = arrowBm.createScaledBitmap(  arrowBm,

        15*width/100,

        arrawImParams.height,
                                                                                true);
                arrowImg.setImageBitmap(arrowBm);

                l.setBackgroundColor(Color.LTGRAY);
                return l;
         }

        public ArrayList<WebPage> getItem(){
                return item;
        }
}


package gr.teiep.orthodoxheart.utils;

import gr.teiep.orthodoxheart.R;
import gr.teiep.orthodoxheart.provider.WebPageData.WebPageArticle;

import java.util.ArrayList;

import android.app.Activity;
import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Color;
import android.view.Display;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.TextView;

import com.squareup.picasso.Picasso;

public class WebPageAdapter extends     ArrayAdapter {

        ArrayList<WebPageArticle> item=null;

        Context mcontext=null;

        public WebPageAdapter(    Context context,
                                                int textViewResourceId,
                                                ArrayList< WebPageArticle > t
        )
        {
               super(context, textViewResourceId, t);
               mcontext=context;
               item= new ArrayList<WebPageArticle>(t);
        }
```

```
 public View getView(     int index,
                                        View convertView,
                                        ViewGroup viewGroup)
 {
        Display display =
                     ((Activity) mcontext).getWindowManager()
                     .getDefaultDisplay();
        int width = display.getWidth();
        int height = display.getHeight();


        LinearLayout l=new LinearLayout(mcontext);
        final ImageView im1=new ImageView(mcontext);
        Picasso.with(mcontext)
                    .load(item.get(index).getmImg())
                    .resize(26 * width/100, 14 * height/100)
                    .into(im1);
        l.addView(im1);
        LinearLayout.LayoutParams im1params =
                     (LinearLayout.LayoutParams)im1.getLayoutParams();
        im1params.topMargin      = 2*height /100;
        im1params.bottomMargin   = 2*height/100;
        im1params.leftMargin     = 2*width/100;
        im1params.height = 14 * height/100;
        im1params.width  = 26 * width/100;
        im1.setLayoutParams(im1params);



        LinearLayout secondLayout= new LinearLayout(mcontext);
        secondLayout.setOrientation(LinearLayout.VERTICAL);
        l.addView(secondLayout);
        TextView t=new TextView(mcontext);
        secondLayout.addView(t);
        LinearLayout.LayoutParams tparams =
                     (LinearLayout.LayoutParams)t.getLayoutParams();
        tparams.height = 10 * height/100;
        tparams.width      = 66 * width/100;
        tparams.topMargin  = 2*height /100;
        tparams.leftMargin = 2*width /100;
        t.setLayoutParams(tparams);
        t.setText(item.get(index).getmTitle());


        ImageView arrowImg=new ImageView(mcontext);
        secondLayout.addView(arrowImg);
        Bitmap arrowBm =
                     BitmapFactory
                     .decodeResource(   mcontext.getResources(),

R.drawable.arrowpencilleftlarge
                                                          );
        double arrowImgRatio =
                     arrowBm.getWidth()*1.0/arrowBm.getHeight();
```

```
                LinearLayout.LayoutParams arrawImParams =
                                (LinearLayout.LayoutParams)arrowImg.getLayoutParams();
                arrawImParams.width      = 4 * width/100;
                arrawImParams.height     =
                                (int)(arrawImParams.width * arrowImgRatio);
                arrawImParams.leftMargin = 54 * width/100;
                arrowBm = arrowBm.createScaledBitmap(  arrowBm,

        4*width/100,

        arrawImParams.height,
                                                                                true
                );
                arrowImg.setImageBitmap(arrowBm);


                l.setBackgroundColor(Color.LTGRAY);
                return l;
         }

         public ArrayList<WebPageArticle> getItem(){
                return item;
         }
}



package gr.teiep.orthodoxheart.view;



import gr.teiep.orthodoxheart.R;
import gr.teiep.orthodoxheart.ops.ArticleOps;
import gr.teiep.orthodoxheart.provider.ArticleData.ArticleElement;
import gr.teiep.orthodoxheart.utils.GenericYouTubeActivity;

import java.util.ArrayList;

import android.app.ProgressDialog;
import android.content.Intent;
import android.graphics.Color;
import android.os.Bundle;
import android.util.DisplayMetrics;
import android.util.Log;
import android.view.Gravity;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.ScrollView;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.youtube.player.YouTubeInitializationResult;
```

```
import com.google.android.youtube.player.YouTubePlayer;
import com.google.android.youtube.player.YouTubePlayerView;
import com.squareup.picasso.Picasso;


public class ArticleActivity extends GenericYouTubeActivity
             implements YouTubePlayer.OnInitializedListener
{
       ArticleOps   articleOps;
       LinearLayout mainLayout   = null;
       LinearLayout l1           = null;

       Button       button       = null;
       TextView     header       = null;
       TextView     txt          = null;
       Bundle       bundle       = null;
       ScrollView   mScrollView  = null;


       private YouTubePlayerView youTubeView = null;
       private static final int        RECOVERY_DIALOG_REQUEST = 1;
       private static final String     DEVELOPER_KEY =
                    "this is a private key";
       private FloatingActionButton mDownloadArticle;
       private static String YOUTUBE_VIDEO_CODE      = null;
       private StringBuilder articleContent          = null;

       private String selectedArticle         = null;
       private String selectedArticleName     = null;
       private ProgressDialog proDialog;


       private void startLoading(String message) {
           proDialog = new ProgressDialog(this);
           proDialog.setMessage(message);
           proDialog.setProgressStyle(ProgressDialog.STYLE_SPINNER);
           proDialog.setCancelable(false);
           proDialog.show();
       }

       private void stopLoading() {
           proDialog.dismiss();
           proDialog = null;
       }

       public void makeHeader()
       {
           header=new TextView(this);
           header.setTextSize(FONT_SIZE);
           header.setText(selectedArticleName);
           header.setGravity(Gravity.CENTER);
           header.setBackgroundColor(Color.GRAY);
           header.setTextColor(Color.BLACK);
           mainLayout.addView(header);
           LinearLayout.LayoutParams hparams =
```

```
            (LinearLayout.LayoutParams)header
            .getLayoutParams();
    hparams.height = 4 * HEIGHT/32;
      header.setLayoutParams(hparams);
}


public void makeYouTubeView(String url)
{
      YOUTUBE_VIDEO_CODE = url;
      youTubeView = new YouTubePlayerView(this);
      l1.addView(youTubeView);
      youTubeView.initialize(DEVELOPER_KEY, this);
}
public void makeTextView(String txt)
{
      this.txt=new TextView(this);
      l1.addView(this.txt);
      this.txt.setTextSize(FONT_SIZE);
      this.txt.setTypeface(TYPE);
      this.txt.setText(txt);
      this.txt.setGravity(Gravity.CENTER);
      this.txt.setPadding(WIDTH /20,
                                    HEIGHT /40,
                                    WIDTH  /20,
                                    HEIGHT /40);
}

public void makeButton(String txt)
{

      button=new Button(this);
      button.setText(txt);
      l1.addView(button);
      button.setPadding(  WIDTH  /20,
                                    HEIGHT /30,
                                    WIDTH  /20,
                                    HEIGHT /30);
      button.setOnClickListener(new OnClickListener()
      {

            @Override
            public void onClick(View v) {

                  Intent I = new Intent(   ArticleActivity.this,

MainActivity.class);
                     I.putExtra("ButtonContent",
                                    button.getText());
                       startActivity(I);
            }
      });
}
```

```
   public void makeImageView(String url)
   {

          ImageView im1=new ImageView(this);
          Picasso.with(this)
                 .load(url)//.resize(getWidth()/2, getHeight()/2)
                 .into(im1);
          l1.addView(im1);
          LinearLayout.LayoutParams im1params =
                 (LinearLayout.LayoutParams)im1.getLayoutParams();
          im1params.topMargin=HEIGHT/30;
          im1params.bottomMargin=HEIGHT/30;
          im1.setLayoutParams(im1params);

   }

   @SuppressWarnings("deprecation")
private void createPlusFabButton() {
    final DisplayMetrics metrics =
        getResources().getDisplayMetrics();
    final int position =
        (metrics.widthPixels / 4) + 5;

    mDownloadArticle =
        new FloatingActionButton
        .Builder(this)
        .withDrawable(getResources()
                      .getDrawable(R.drawable.ic_save_white_24dp))
        .withButtonColor(0xff000000)
        .withGravity(Gravity.TOP | Gravity.RIGHT)
        .withMargins(0,HEIGHT/32,0,0)
        .create();

    // Download the current article when user clicks the button.
    mDownloadArticle.setOnClickListener(new OnClickListener() {
            public void onClick(View v) {

                downloadArticle(selectedArticleName, articleContent);
            }
        });
   }

   public void downloadArticle(String fileName,
                                                 StringBuilder articleContent
   ) {

          articleOps.downloadArticle(fileName, articleContent);
   }


private void findArticleContet(String selectedArticle) {

   startLoading("Loading article content...");
       articleOps.findArticleContet(selectedArticle);
}
```

```
   public void makeToast(String message) {
          Toast.makeText(this,
                                    message,
                                    Toast.LENGTH_SHORT).show();
   }

   @Override
   protected void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState,this);
          mainLayout=new LinearLayout(this);
          setContentView(mainLayout);
          mainLayout.setOrientation(LinearLayout.VERTICAL);
          mainLayout.setBackgroundColor(Color.LTGRAY);
          bundle = getIntent().getExtras();
          selectedArticle = bundle.getString("selectedArticle");
          selectedArticleName =
                       bundle.getString("selectedArticleName");
          articleOps = new ArticleOps();
       makeHeader();
       createPlusFabButton();
       mScrollView = new ScrollView(this);
          mainLayout.addView(mScrollView);
          l1 = new LinearLayout(this);
          mScrollView.addView(l1);
          l1.setOrientation(LinearLayout.VERTICAL);
       articleOps.onConfiguration(ArticleActivity.this);
       findArticleContet(selectedArticle);
   }

/**
 * Hook method called by Android when this Activity becomes
 * invisible.
 */
@Override
protected void onDestroy() {
    // Always call super class for necessary operations when
    // stopping.
    super.onDestroy();
    Log.d(      "articleActivity",
                "onDestroy()-the activity is about to be destroyed");
}

@Override
protected void onStop() {
    // Always call super class for necessary operations when
    // stopping.
    super.onStop();

    Log.d("articleActivity",
                "onStop()-the activity is about to be stoped");
}


   public void displayResults(ArrayList<ArticleElement> listItems,
```

```
        String errorMessage) {

        if (listItems == null
        || listItems.size() == 0)
              Log.d("","");
        else {
              Log.d("",
              "displayResults() with number of acronyms = "
              + listItems.size());

              articleContent = new StringBuilder();

              for(int i=0; i<listItems.size(); i++) {



                    if(listItems.get(i).geteElement()
                                .equals("p"))
                    {
                          articleContent.append("<p>"
                                      + listItems.get(i).geteContent()
                                      + "<p>");
                          makeTextView(listItems.get(i).geteContent());
                    }
                    if(listItems.get(i).geteElement()
                                .equals("a"))
                    {
                          articleContent.append("<p> <a href=\""
                                      + listItems.get(i).geteContent()
                                      + "\">"
                                      + listItems.get(i).geteContent()
                                      + "</a><p>");
                          makeButton(listItems.get(i).geteContent());
                    }
                    else if(listItems.get(i).geteElement()
                                .equals("yt"))
                    {
                          makeYouTubeView(listItems.get(i).geteBm());
                    }
                    else if(listItems.get(i).geteElement()
                                .equals("img"))
                    {
                          makeImageView(listItems.get(i).geteBm());
                    }

              }
        }
        stopLoading();
    }

   @Override
public void onInitializationFailure(
         YouTubePlayer.Provider provider,
        YouTubeInitializationResult errorReason
) {
```

```
        if (errorReason.isUserRecoverableError()) {
            errorReason.getErrorDialog(
                    this,
                    RECOVERY_DIALOG_REQUEST)
                    .show();
        } else {
            String errorMessage = String.format(
                    "dfasfdaseeerrroorrrr", errorReason.toString());
            Toast.makeText(this,
                    errorMessage,
                    Toast.LENGTH_LONG)
                    .show();
        }
    }

    @Override
    public void onInitializationSuccess(
                    YouTubePlayer.Provider provider,
                    YouTubePlayer player,
                    boolean wasRestored) {
        if (!wasRestored) {

            // loadVideo() will auto play video
            // Use cueVideo() method, if you don't want to play it
             // automatically
            player.cueVideo(YOUTUBE_VIDEO_CODE);

            // Hiding player controls
//            player.setPlayerStyle(PlayerStyle.CHROMELESS);
        }
    }

    @Override
    protected void onActivityResult(int requestCode,
                                                    int resultCode,
                                                    Intent data) {
        if (requestCode == RECOVERY_DIALOG_REQUEST) {
            // Retry initialization if user performed a recovery
             // action
             youTubeView.initialize(DEVELOPER_KEY, this);
        }
    }
}
```

```
package gr.teiep.orthodoxheart.view;


import gr.teiep.orthodoxheart.ops.FilesOps;
import gr.teiep.orthodoxheart.utils.FilesListAdapter;
import gr.teiep.orthodoxheart.utils.GenericActivity;
```

```
import java.util.ArrayList;

import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Color;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.ViewGroup.LayoutParams;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.Button;
import android.widget.LinearLayout;
import android.widget.ListView;
import android.widget.Toast;


public class FilesActivity extends GenericActivity
{

	LinearLayout mainLayout   = null;
	LinearLayout subLayout    = null;
	ListView           list         = null;
	FilesListAdapter   adapter      = null;
	Button readAtricleButton        = null;
	Button deleteAtricleButton      = null;
	StringBuilder articleContent    = null;
	FilesOps filesOps;
	String articleName;

	public void makeList()
	{
		list=new ListView(this);
		mainLayout.addView(list);
		LinearLayout.LayoutParams listparams =
				(LinearLayout.LayoutParams)list.getLayoutParams();
		listparams.width=WIDTH;
		listparams.height= 15 * HEIGHT/20;
		list.setLayoutParams(listparams);
		list.setOnItemClickListener(new OnItemClickListener()
		{

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

				articleName =adapter.getItem(position).toString();
				if(subLayout==null)
					makeButtons();
				else
				{
					if(subLayout.getVisibility()==View.GONE)
						showButtons();
```

```
                        else

                              hideButtons();
                  }
            }
      });
}

public void makeButtons()
{
      LayoutParams params = new LayoutParams(
                  LayoutParams.WRAP_CONTENT,
                  LayoutParams.WRAP_CONTENT);
      subLayout=new LinearLayout(this);
 addContentView(subLayout,params);
 LinearLayout l1=new LinearLayout(this);
 subLayout.addView(l1);
 LinearLayout.LayoutParams l1params =
            (LinearLayout.LayoutParams)l1.getLayoutParams();
 l1params.topMargin=HEIGHT/2;
 l1params.leftMargin=WIDTH/10;
 readAtricleButton=new Button(this);
 readAtricleButton.setText("Read");
 readAtricleButton.setOnClickListener(new OnClickListener() {

            @Override
            public void onClick(View v) {
                  Intent I = new Intent(    FilesActivity.this,

TextDisplayActivity.class);
                  Bundle extras = new Bundle();
                  extras.putString("headerContent", articleName);
                  getArticleContent(articleName);
                  String bodyContent = articleContent.toString();
                  extras.putString("bodyContent", bodyContent);
                I.putExtras(extras);
                  startActivity(I);
            }
      });
 l1.addView(readAtricleButton);
 deleteAtricleButton=new Button(this);
 deleteAtricleButton.setText("Delete");
 l1.addView(deleteAtricleButton);
 deleteAtricleButton.setOnClickListener(new OnClickListener()
 {
      @Override
      public void onClick(View v)
      {
            AlertDialog.Builder alert = new
                        AlertDialog.Builder(FilesActivity.this);
            alert.setTitle("Delete article?");
            alert.setMessage("You 're going to delete this article.");
            alert.setPositiveButton(
                        "Delete",
                        new DialogInterface.OnClickListener()
```

```
                {
                        @Override
                        public void onClick(DialogInterface dialog,int which)
                        {

                                if(filesOps.deleteArticleContent(articleName))
                                        Toast.makeText(     FilesActivity.this,
                                                                "Article was
deleted.",
                                        Toast.LENGTH_SHORT).show();

                                else
                                        Toast.makeText(     FilesActivity.this,
                                                                "Article was not
deleted.",
                                                Toast.LENGTH_SHORT).show();

                                adapter.notifyDataSetChanged();
                                dialog.dismiss();
                        }
                });
                alert.setNegativeButton(
                                "Cancel",
                                new DialogInterface.OnClickListener()
                {
                        @Override
                        public void onClick(DialogInterface dialog,int which)
                        {
                                dialog.dismiss();
                        }
                });
                alert.show();
            }
          });
        }


        @Override
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState,this);
                mainLayout=new LinearLayout(this);
                setContentView(mainLayout);
                mainLayout.setOrientation(LinearLayout.VERTICAL);
                mainLayout.setBackgroundColor(Color.LTGRAY);
                filesOps = new FilesOps();
                makeList();
                filesOps.onConfiguration(FilesActivity.this);
                filesOps.getFiles();
        }

    /**
     * Hook method called by Android when this Activity becomes
     * invisible.
     */
    @Override
```

```
    protected void onDestroy() {
        // Always call super class for necessary operations when
        // stopping.
        super.onDestroy();
        Log.d(      "articleActivity",
                    "onDestroy()-the activity is about to be destroyed");
    }


      public void displayResults(ArrayList<String> arrayList,
            String errorMessage) {

            if (arrayList == null
            || arrayList.size() == 0)
                  Log.d("","");
            else {
                  Log.d("",
                  "displayResults() with number of files = "
                  + arrayList.size());

                  adapter=new FilesListAdapter(
                              FilesActivity.this,
                              android.R.layout.simple_list_item_1,
                              arrayList);
                  list.setAdapter(adapter);

            }
    }

      private void getArticleContent(String articleName) {

            filesOps.getArticleContent(articleName);
      }

      public void setArticleContent(StringBuilder sb) {

            articleContent = sb;
      }

      private void showButtons()
      {
            subLayout.setVisibility(View.VISIBLE);
      }

      private void hideButtons()
      {
            subLayout.setVisibility(View.GONE);
      }

}

package gr.teiep.orthodoxheart.view;

import android.animation.AnimatorSet;
import android.animation.ObjectAnimator;
```

```
import android.annotation.SuppressLint;
import android.app.Activity;
import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.drawable.BitmapDrawable;
import android.graphics.drawable.Drawable;
import android.view.Gravity;
import android.view.MotionEvent;
import android.view.View;
import android.view.ViewGroup;
import android.view.animation.AccelerateInterpolator;
import android.view.animation.OvershootInterpolator;
import android.widget.FrameLayout;

/**
 * CustomView that shows how to create a Floating Action Button, as
 * per Google's Material Design principles.
 */
public class FloatingActionButton extends View {
    /**
     * An interpolator where the change flings forward and overshoots
     * the last value then comes back.
     */
    final static OvershootInterpolator overshootInterpolator =
        new OvershootInterpolator();

    /**
     * An interpolator where the rate of change starts out slowly and
     * and then accelerates.
     */
    final static AccelerateInterpolator accelerateInterpolator =
        new AccelerateInterpolator();

    /**
     * Gets access to application-specific resources.
     */
    Context context;

    /**
     * Paints used to draw the Button in Canvas.
     */
    Paint mButtonPaint;
    Paint mDrawablePaint;

    /**
     * Bitmap of the icon present in Floating Action Button
     */
    Bitmap mBitmap;

    /**
     * Boolean to indicate if the Button is hidden or not.
     */
```

```
boolean mHidden = false;


/**
 * Constructor that initializes the Floating
 * Action Button.
 *
 * @param context
 */
public FloatingActionButton(Context context) {
    super(context);
    this.context = context;
    init(Color.WHITE);
}

/**
 * Sets the Color of FloatingActionButton.
 *
 * @param FloatingActionButtonColor
 */
public void setFloatingActionButtonColor(
         int FloatingActionButtonColor
) {
    init(FloatingActionButtonColor);
}

/**
 * Sets the Icon of FloatingActionButton.
 *
 * @param FloatingActionButtonDrawable
 */
public void setFloatingActionButtonDrawable(
         Drawable FloatingActionButtonDrawable
) {
    mBitmap =
    ((BitmapDrawable) FloatingActionButtonDrawable).getBitmap();
    invalidate();
}

/**
 * Initialize all the Resources needed before drawing.
 *
 * @param FloatingActionButtonColor
 */
public void init(int FloatingActionButtonColor) {
    setWillNotDraw(false);
    setLayerType(View.LAYER_TYPE_SOFTWARE, null);

    mButtonPaint = new Paint(Paint.ANTI_ALIAS_FLAG);
    mButtonPaint.setColor(FloatingActionButtonColor);
    mButtonPaint.setStyle(Paint.Style.FILL);
    mButtonPaint.setShadowLayer(10.0f,
                                0.0f,
                                3.5f,
                                Color.argb(100,
```

```
                                            0,
                                            0,
                                            0));
    mDrawablePaint = new Paint(Paint.ANTI_ALIAS_FLAG);

    invalidate();
}

/**
 * Hook method called to draw the View on the Canvas.
 *
 *@param canvas
 */
@Override
protected void onDraw(Canvas canvas) {
    setClickable(true);
    canvas.drawCircle(getWidth() / 2,
                      getHeight() / 2,
                      (float) (getWidth() / 2.6),
                      mButtonPaint);
    canvas.drawBitmap(mBitmap,
                                 (getWidth() - mBitmap.getWidth()) / 2,
                      (getHeight() - mBitmap.getHeight()) / 2,
                      mDrawablePaint);
}

/**
 * Hook method called when View is Touched.
 *
 * @param event
 */
@SuppressLint("ClickableViewAccessibility")
@Override
public boolean onTouchEvent(MotionEvent event) {
    if (event.getAction() == MotionEvent.ACTION_UP) {
        setAlpha(1.0f);
        showFloatingActionButton();
    } else if (event.getAction() == MotionEvent.ACTION_DOWN) {
        setAlpha(0.6f);
        hideFloatingActionButton();
    }
    return super.onTouchEvent(event);
}

/**
 * Hides the Floating Action Button with some Animation.
 */
public void hideFloatingActionButton() {
    if (!mHidden) {
        ObjectAnimator scaleX = ObjectAnimator.ofFloat(      this,

  "scaleX",

  1,
```

```
   0);
        ObjectAnimator scaleY = ObjectAnimator.ofFloat(      this,

   "scaleY",

   1,

   0);
        AnimatorSet animSetXY = new AnimatorSet();
        animSetXY.playTogether(scaleX, scaleY);
        animSetXY.setInterpolator(accelerateInterpolator);
        animSetXY.setDuration(100);
        animSetXY.start();
        mHidden = true;
    }
}

/**
 * Shows the Floating Action Button with some Animation.
 */
public void showFloatingActionButton() {
    if (mHidden) {
        ObjectAnimator scaleX = ObjectAnimator.ofFloat(      this,

   "scaleX",

   0,

   1);
        ObjectAnimator scaleY = ObjectAnimator.ofFloat(      this,

   "scaleY",

   0,

   1);
        AnimatorSet animSetXY = new AnimatorSet();
        animSetXY.playTogether(scaleX, scaleY);
        animSetXY.setInterpolator(overshootInterpolator);
        animSetXY.setDuration(200);
        animSetXY.start();
        mHidden = false;
    }
}

/**
 * @return True if the View is hidden.
 */
public boolean isHidden() {
    return mHidden;
}

/**
 * Builder pattern used to Build the Floating
```

```
 * Action Button.
 */
static public class Builder {
    private FrameLayout.LayoutParams params;
    private final Activity activity;
    int gravity = Gravity.BOTTOM | Gravity.END; // default bottom right
    Drawable drawable;
    int color = Color.WHITE;
    int size = 0;
    float scale = 0;

    /**
     * Constructor used to initialize the Builder.
     *
     * @param context
     */
    public Builder(Activity context) {
        scale = context.getResources().getDisplayMetrics().density;
        size = convertToPixels(72, scale);
        params = new FrameLayout.LayoutParams(size, size);
        params.gravity = gravity;

        this.activity = context;
    }

    /**
     * Sets the Gravity of the View.
     *
     * @param gravity
     * @return Builder
     */
    public Builder withGravity(int gravity) {
        this.gravity = gravity;
        return this;
    }

    /**
     * Sets the Margins of the View.
     *
     * @param left
     * @param top
     * @param right
     * @param bottom
     *
     * @return Builder
     */
    public Builder withMargins(    int left,
                                   int top,
                                   int right,
                                   int bottom) {
        params.setMargins(
                        convertToPixels(left, scale),
                        convertToPixels(top, scale),
                        convertToPixels(right, scale),
                        convertToPixels(bottom, scale));
```

```
    return this;
}

/**
 * Sets the Drawable used by the View.
 *
 * @param drawable
 * @return Builder
 */
public Builder withDrawable(final Drawable drawable) {
    this.drawable = drawable;
    return this;
}

/**
 * Sets the color used by the View.
 *
 * @param color
 * @return Builder
 */
public Builder withButtonColor(final int color) {
    this.color = color;
    return this;
}

/**
 * Sets the size of the View.
 *
 * @param size
 * @return Builder
 */
public Builder withButtonSize(int size) {
    size = convertToPixels(size, scale);
    params = new FrameLayout.LayoutParams(size, size);
    return this;
}

/**
 * Creates the Floating Action Button.
 *
 * @return FloatingActionButton
 */
public FloatingActionButton create() {
    final FloatingActionButton button =
            new FloatingActionButton(activity);
    button.setFloatingActionButtonColor(this.color);
    button.setFloatingActionButtonDrawable(this.drawable);
    params.gravity = this.gravity;
    ViewGroup root =
            (ViewGroup) activity
            .findViewById(android.R.id.content);
    root.addView(button, params);
    return button;
}
```

```
        /**
         * Calculate and scale the values to fit
         * the larger devices.
         *
         * @param dp
         * @param scale
         * @return
         */
        private int convertToPixels(int dp, float scale){
            return (int) (dp * scale + 0.5f) ;
        }
    }
}
```

```
package gr.teiep.orthodoxheart.view;



import gr.teiep.orthodoxheart.ops.HttpOps;
import gr.teiep.orthodoxheart.provider.MainIndexData.WebPage;
import gr.teiep.orthodoxheart.utils.GenericActivity;
import gr.teiep.orthodoxheart.utils.MainIndexListAdapter;

import java.net.URISyntaxException;
import java.util.ArrayList;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.ActivityNotFoundException;
import android.content.ComponentName;
import android.content.Intent;
import android.graphics.Color;
import android.net.Uri;
import android.os.Bundle;
import android.util.Log;
import android.view.Display;
import android.view.Gravity;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.Button;
import android.widget.LinearLayout;
import android.widget.ListView;
import android.widget.TextView;


public class MainActivity extends GenericActivity {


       LinearLayout mainLayout = null;
       HttpOps httpOps;
       ListView list = null;
```

```
        Button goButton     = null;
        TextView header     = null;
        String imagepath    = null;
        Bundle bundle       = null;
        MainIndexListAdapter adapter = null;


        public  String preSelectedWebPage = null;
        private ProgressDialog proDialog;
        private void startLoading(String message) {
            proDialog = new ProgressDialog(MainActivity.this);
            proDialog.setMessage(message);
            proDialog.setProgressStyle(ProgressDialog.STYLE_SPINNER);
            proDialog.setCancelable(false);
            proDialog.show();
        }

        private void stopLoading() {
            proDialog.dismiss();
            proDialog = null;
        }

        public void makeHeader()
        {
            header=new TextView(this);
            header.setTextSize(FONT_SIZE+5);
            header.setText("Select Web Page");
            header.setGravity(Gravity.CENTER);
            header.setBackgroundColor(Color.GRAY);
            header.setTextColor(Color.BLACK);
            mainLayout.addView(header);
            LinearLayout.LayoutParams hparams =
                     (LinearLayout.LayoutParams)header.getLayoutParams();
            hparams.height = 3 * HEIGHT/20;
              header.setLayoutParams(hparams);
        }

        public void makeList()
        {
               list=new ListView(this);
               mainLayout.addView(list);
               LinearLayout.LayoutParams listparams =
                            (LinearLayout.LayoutParams)list.getLayoutParams();
//             listparams.width=WIDTH;
//             listparams.height = 17 * HEIGHT/20;
//             list.setLayoutParams(listparams);
               list.setOnItemClickListener(new OnItemClickListener()
               {

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

                            Intent I = new Intent(    MainActivity.this,
```

```
WebPageActivity.class);
                    Bundle extras = new Bundle();
                    extras
                    .putString(
                                "title",
                                adapter.getItem()
                                .get(position).getSiteName().toString());
                    extras.putString(
                                "selectedWebPage",
                                adapter.getItem()
                                .get(position).getSiteUrl().toString());
                I.putExtras(extras);
                    startActivity(I);
            }
      });
}

@Override
public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState, this);
      mainLayout=new LinearLayout(this);
      setContentView(mainLayout);
      mainLayout.setOrientation(LinearLayout.VERTICAL);


      httpOps = new HttpOps();
      httpOps.onConfiguration(this);
      if(getIntent().getExtras()!= null) {
            bundle = getIntent().getExtras();
            preSelectedWebPage = bundle.getString("ButtonContent");
            if(httpOps.isOrthoxHeartSite())
            {
                  Intent I = new Intent(    MainActivity.this,

WebPageActivity.class);
                I.putExtra("selectedWebPage",
                        preSelectedWebPage);
                startActivity(I);
            }
            else {
                  Uri uri = Uri.parse(preSelectedWebPage);
                  Intent intent = new Intent(Intent.ACTION_VIEW, uri);
                  startActivity(intent);
            }
      }
    makeHeader();
    makeList();
    findWebSites();
}

 public void displayResults(ArrayList<WebPage>  arrayList,
      String errorMessage) {

      if ( arrayList == null
```

```
            ||  arrayList.size() == 0)
                  Log.d("","");
            else {
                  Log.d("",
                  "displayResults() with number of acronyms = "
                  +  arrayList.size());

                  // Add the results to the Adapter and notify changes.
                  adapter= new MainIndexListAdapter(
                              MainActivity.this,
                              android.R.layout.simple_list_item_1, arrayList
                  );
                  list.setAdapter(adapter);
            }
            stopLoading();
      }

      private void findWebSites() {
            startLoading("Loading Othodox Heart sites...");
            httpOps.findWebSites();
      }
}


package gr.teiep.orthodoxheart.view;

import gr.teiep.orthodoxheart.R;
import gr.teiep.orthodoxheart.utils.GenericActivity;
import android.app.ActionBar.LayoutParams;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Color;
import android.os.Bundle;
import android.text.Html;
import android.text.InputType;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;


public class SendMailActivity extends GenericActivity {
      LinearLayout mainLayout=null;
      LinearLayout btnLayout=null;
      EditText emailBody=null;
      TextView textReciver=null;
      Button mEmailSend;
      private String e="gkiouz.anast@gmail.com";


    @Override
```

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState, this);
    mainLayout=new LinearLayout(this);
    mainLayout.setOrientation(LinearLayout.VERTICAL);
    mainLayout.setBackgroundColor(Color.WHITE);
    setContentView(mainLayout);
       getActionBar().setTitle("New message");

    textReciver = new TextView(this);
    mainLayout.addView(textReciver);

    textReciver.setText("To: gkiouz.anast@gmail.com");
    textReciver.setTextSize(FONT_SIZE);
         LinearLayout.LayoutParams textReciverParams=
                     (LinearLayout.LayoutParams)     textReciver

   .getLayoutParams();
         textReciverParams.bottomMargin= HEIGHT/20;
         textReciverParams.topMargin = HEIGHT/20;
         textReciver.setLayoutParams(textReciverParams);

    emailBody=new EditText(this);
    emailBody.setHint("Feel free to contact us");
    mainLayout.addView(emailBody);
    LinearLayout.LayoutParams lp =
               new LinearLayout.LayoutParams(
                           LayoutParams.MATCH_PARENT,
                           LayoutParams.WRAP_CONTENT,
                           1.0f);
    lp.bottomMargin= HEIGHT/9;
    emailBody.setLayoutParams(lp);
    emailBody.setSingleLine(false);
    emailBody.setInputType(InputType.TYPE_TEXT_FLAG_MULTI_LINE);
    emailBody.setBackgroundResource(R.drawable.edit_text_border);

    mEmailSend = new Button(this);
    mainLayout.addView(mEmailSend);
    mEmailSend.setText("Send");
    mEmailSend.setTextColor(Color.rgb(3, 169, 244));

    // Send the email when user clicks the button.
    mEmailSend.setOnClickListener(new OnClickListener()
    {
         @Override
         public void onClick(View v)
         {
               AlertDialog.Builder alert = new
                           AlertDialog.Builder(SendMailActivity.this);
               alert.setTitle("Send email?");
               alert.setMessage("You 're going to send this email.");
               alert.setPositiveButton(
                           "Send",
                           new DialogInterface.OnClickListener()
               {
                     @Override
```

```
                        public void onClick(DialogInterface dialog,
                                                int which)
                        {
                                Intent i=
                                        new Intent(Intent.ACTION_SEND);
                        i.setType("text/plain");
                        i.putExtra(  Intent.EXTRA_EMAIL ,
                                        new String[]{e});
                        i.putExtra(  Intent.EXTRA_SUBJECT,
                                        "teiep");
                        String myMessage =
                                    "<h1>Your message</h1>"+emailBody;
                        i.putExtra(  Intent.EXTRA_TEXT,
                                        Html.fromHtml(myMessage));
                        try {
                                startActivity(
                                        Intent
                                        .createChooser(i,"Send mail...")
                                );
                        }
                        catch (
                        android.content.ActivityNotFoundException ex
                        )
                        {
                        Toast
                        .makeText(
                                SendMailActivity.this,
                                "There aren't email clients installed.",
                                Toast.LENGTH_SHORT).show();
                        }
                                dialog.dismiss();
                        }
                    });
                    alert.setNegativeButton("Cancel",
                                new DialogInterface.OnClickListener()
                    {
                            @Override
                            public void onClick(DialogInterface dialog,
                                                int which)
                            {
                                dialog.dismiss();
                            }
                    });
                    alert.show();
                }
            });
        }

}


package gr.teiep.orthodoxheart.view;
```

```
import gr.teiep.orthodoxheart.utils.GenericActivity;
import android.graphics.Color;
import android.os.Bundle;
import android.text.Html;
import android.view.Gravity;
import android.view.ViewGroup.LayoutParams;
import android.widget.Button;
import android.widget.LinearLayout;
import android.widget.ScrollView;
import android.widget.TextView;

public class TextDisplayActivity extends GenericActivity {


	TextView header=null;
	LinearLayout mainLayout=null;
	ScrollView scrollView=null;
	TextView body=null;
	Bundle bundle=null;
	String headerContent=null;
	String bodyContet=null;


	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState, this);
		mainLayout=new LinearLayout(this);
		setContentView(mainLayout);
		mainLayout.setOrientation(LinearLayout.VERTICAL);
		scrollView = new ScrollView(this);
		mainLayout.addView(scrollView);
		mainLayout.setBackgroundColor(Color.LTGRAY);

		if (getIntent().getExtras() != null)
		{
			bundle = getIntent().getExtras();
			headerContent = bundle.getString("headerContent");
			bodyContet = bundle.getString("bodyContent");
			displayArticle();
		}
		else {
			displayInfo();
		}
	}

	public void makeHeader(String txt)
	{
	    header=new TextView(this);
	    header.setTextSize(FONT_SIZE+5);
	    header.setText(txt);
	    header.setGravity(Gravity.CENTER);
	    header.setBackgroundColor(Color.GRAY);
	    header.setTextColor(Color.BLACK);
	    mainLayout.addView(header);
	    LinearLayout.LayoutParams hparams=
```

```
                (LinearLayout.LayoutParams)header.getLayoutParams();
        hparams.height = 3 * HEIGHT/20;
           header.setLayoutParams(hparams);
    }

    public void makeBody(String txt)
    {
           body=new TextView(this);
           scrollView.addView(body);
           body.setTextSize(FONT_SIZE);
           body.setTypeface(TYPE);
           body.setText(Html.fromHtml(txt));
           body.setGravity(Gravity.CENTER);
           body.setPadding(WIDTH/20, HEIGHT/40, WIDTH/20, HEIGHT/40);
    }

    private void displayArticle() {

           makeHeader(headerContent);
           makeBody(bodyContet);
    }

    private void displayInfo() {

           getActionBar().setTitle("Info");
           makeBody("<h1>ABEL-TASOS GKIOUZELIS: MY HEART SITES &#8211;"
                        +" ST JOHN MAXIMOVITCH &amp; ST BRIGID OF KILDARE,"
                        +" IRELAND꒰,,•¨* 微笑 &#8211; 미소 &#8211; スマイル "
                        +"&#8211; SMILE &#8211; AOIBH GHÁIRE</h1>"
                        + "<h2>PAINTING LEAVES IN ORTHODOXY &#8211; "
                        + "EASTERN ORTHODOX CHURCH ꒰,,•¨* Abel-Tasos "
                        + "Gkiouzelis &#8211; https://gkiouzelis.wordpress.com "
                        + "&#8211; Email: gkiouz.anast@gmail.com &#8211; "
                        + "Feel free to email me…!</h2>");
    }
}


package gr.teiep.orthodoxheart.view;



import gr.teiep.orthodoxheart.ops.WebPageOps;
import gr.teiep.orthodoxheart.provider.WebPageData.WebPageArticle;
import gr.teiep.orthodoxheart.utils.GenericActivity;
import gr.teiep.orthodoxheart.utils.WebPageAdapter;

import java.util.ArrayList;

import android.app.ProgressDialog;
import android.content.Intent;
import android.graphics.Color;
import android.os.Bundle;
import android.text.Html;
```

```
import android.text.TextUtils;
import android.util.Log;
import android.view.Gravity;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.Button;
import android.widget.LinearLayout;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;


public class WebPageActivity extends GenericActivity {

	LinearLayout mainLayout   = null;
	LinearLayout l                  = null;

	WebPageOps webPageOps;
	ListView list       = null;
	TextView header     = null;
	Bundle bundle       = null;
	Button listFooterBtn            = null;
	WebPageAdapter adapter          = null;
	public String selectedWebPage   = null;
	private String moreAriclesPage  = null;

	private ProgressDialog proDialog;
	private void startLoading(String message) {
	    proDialog = new ProgressDialog(WebPageActivity.this);
	    proDialog.setMessage(message);
	    proDialog.setProgressStyle(ProgressDialog.STYLE_SPINNER);
	    proDialog.setCancelable(false);
	    proDialog.show();
	}

	private void stopLoading() {
	    proDialog.dismiss();
	    proDialog = null;
	}

	public void makeHeader()
	{
	    header=new TextView(this);
	    header.setBackgroundColor(Color.GRAY);
	    header.setTextColor(Color.BLACK);
	    header.setTextSize(FONT_SIZE);
//	    header.setText();
	    header.setGravity(Gravity.CENTER);
	    mainLayout.addView(header);
	    LinearLayout.LayoutParams hparams =
	             (LinearLayout.LayoutParams)header.getLayoutParams();
	      hparams.height = 3 * HEIGHT/20;
	      header.setLayoutParams(hparams);
```

```
            header.setEllipsize(TextUtils.TruncateAt.END);

        }


        public void makeList()
        {
                list=new ListView(this);
                mainLayout.addView(list);
                LinearLayout.LayoutParams listparams =
                                (LinearLayout.LayoutParams)list.getLayoutParams();
//              listparams.width=WIDTH;
//              listparams.height = 15 * HEIGHT/20;
//              list.setLayoutParams(listparams);
                listFooterBtn = new Button(this);
                listFooterBtn.setText("Show more...");
                listFooterBtn.setBackgroundColor(Color.LTGRAY);
                list.addFooterView(listFooterBtn);
                l =new LinearLayout(this);
                listFooterBtn.setOnClickListener(new OnClickListener()
                {
                        @Override
                        public void onClick(View v) {
                                if(getMoreAriclesPage()!=
                                                null && getMoreAriclesPage()!=""
                                )
                                        findWebPageContet(getMoreAriclesPage());
                                else {

                                        Toast.makeText(
                                                        WebPageActivity.this,
                                                        "There are no more articles to
load...",
                                                        Toast.LENGTH_SHORT)
                                                        .show();
                                }
                        }
                });
                list.setOnItemClickListener(new OnItemClickListener()
                {

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

                                Intent I = new Intent(    WebPageActivity.this,

        ArticleActivity.class);
                            // Store the selected article name and the selected
                                // article url to send to the ArticleActivity.
                                Bundle extras = new Bundle();
                                extras.putString(
                                                "selectedArticleName",
                                                adapter.getItem()
                                                .get(position).getmTitle().toString());
```

```
                        extras.putString("selectedArticle",
                                    adapter.getItem()
                                    .get(position).getmUrl().toString());
                    I.putExtras(extras);
                        startActivity(I);
                }
        });
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState, this);
        mainLayout=new LinearLayout(this);
        setContentView(mainLayout);
        mainLayout.setOrientation(LinearLayout.VERTICAL);
        bundle = getIntent().getExtras();
        selectedWebPage = bundle.getString("selectedWebPage");
//      getActionBar().setTitle(bundle.getString("title"));
        webPageOps = new WebPageOps();
      makeHeader();
      makeList();
      webPageOps.onConfiguration(WebPageActivity.this);
      findWebPageContet(selectedWebPage);
    }


     public void displayResults(ArrayList<WebPageArticle> mResults,
          String errorMessage) {

          if (mResults == null
          || mResults.size() == 0)
                Log.d("","");
          else {
                Log.d("",
                "displayResults() with number of articles = "
                + mResults.size());

                // Add the results to the Adapter and notify changes.
                adapter=
                            new WebPageAdapter(
                                        WebPageActivity.this,
                                        android.R.layout.simple_list_item_1,
                                        mResults
                );
                list.setAdapter(adapter);
          }
          stopLoading();
     }

     public void displayTitle(String siteTitle) {

           header.setText(siteTitle);
     }

     private voidfindWebPageContet(String selectedWebPage) {
```

```
                startLoading("Loading articles...");
                webPageOps.findWebPageContet(selectedWebPage);
        }

        public void setMoreAriclesPage(String moreAriclesPage)
        {
                this.moreAriclesPage = moreAriclesPage;
        }

        public String getMoreAriclesPage()
        {
                return this.moreAriclesPage;
        }
}
```

activitymenu.xml

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


       <item
         android:id="@+id/folder_open"
         android:icon="@drawable/ic_folder_white_48dp"
         android:title="@string/folder_open"
         android:focusable="true"
         android:showAsAction="ifRoom|withText">
    </item>
    <item
        android:id="@+id/action_settings"
        android:orderInCategory="100"
        android:icon="@drawable/menuopen"
        android:title="@string/menu_open"
        android:focusable="true"
        android:showAsAction="ifRoom|withText">
        <menu>
            <item android:id="@+id/form1"
                      android:title="@string/contact_us"
                      android:icon="@drawable/ic_mail_white_24dp"/>
            <item android:id="@+id/form2"
                  android:title="@string/info"
                  android:icon="@drawable/ic_info_white_24dp"/>
        </menu>
    </item>

</menu>
```

AndroidManifest.xml

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

    <uses-sdk
        android:minSdkVersion="16"
        android:targetSdkVersion="21" />

    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />



    <application
        android:allowBackup="true"
        android:icon="@drawable/flag_of_byzantio"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".view.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

                <activity
            android:name=".view.WebPageActivity"
            android:parentActivityName=".view.MainActivity" >
        <!-- Parent activity meta-data to support API level 7+ -->
        <meta-data
            android:name="android.support.PARENT_ACTIVITY"
            android:value=".view.MainActivity" />
        </activity>

        <activity
            android:name=".view.ArticleActivity"
            android:parentActivityName=".view.MainActivity" >
        <!-- Parent activity meta-data to support API level 7+ -->
        <meta-data
            android:name="android.support.PARENT_ACTIVITY"
            android:value=".view.MainActivity" />
        </activity>

         <activity
            android:name=".view.SendMailActivity"
```

```
            android:parentActivityName=".view.MainActivity" >
        <!-- Parent activity meta-data to support API level 7+ -->
        <meta-data
            android:name="android.support.PARENT_ACTIVITY"
            android:value=".view.MainActivity" />
        </activity>
         <activity
            android:name=".view.FilesActivity"
             android:parentActivityName=".view.MainActivity" >
        <!-- Parent activity meta-data to support API level 7+ -->
        <meta-data
            android:name="android.support.PARENT_ACTIVITY"
            android:value=".view.MainActivity" />
        </activity>
           <activity
            android:name=".view.TextDisplayActivity"
             android:parentActivityName=".view.MainActivity" >
        <!-- Parent activity meta-data to support API level 7+ -->
        <meta-data
            android:name="android.support.PARENT_ACTIVITY"
            android:value=".view.MainActivity" />
        </activity>
    </application>

</manifest>
```

edit_text_border.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle"
    android:thickness="0dp" >

    <corners android:radius="5dp" />

    <gradient
        android:angle="270"
        android:endColor="#BDBDBD"
        android:startColor="#EEEEEE"
        android:type="linear" />

</shape>
```

Strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">Orthodox Heart</string>
    <string name="hello_world">Hello world!</string>
```

```
    <string name="info">Info</string>
    <string name="contact_us">Contact us!</string>
    <string name="menu_open"></string>
    <string name="folder_open"></string>


</resources>
```

# Βιβλιογραφία


1. Vladimir Maskov (2015) Implementing REST Client for Android

2. TOM FARLEY Mobile telephone history http://www.privateline.com/archive/TelenorPage_022-034.pdf

3. Morgan Payton (2014) The Impact of Mobile Technology
And how it has made a change to society
School of Computing and Mathematics
University of Derby

4. Θεοδωρογιαννάκης Ι. Παναγιώτης (2012) Ανάπτυξη εφαρμογής σε περιβάλλον Google Android, Πανεπιστήμιο Πατρών.

5. Denzil Ferreira, Anind K. Dey, Vassilis Kostakos
Understanding Human-Smartphone Concerns: A Study of Battery Life
http://www.ee.oulu.fi/~vassilis/files/papers/pervasive11.pdf
6. Nicholas D. Lane et all, A Survey of Mobile Phone Sensing
https://www.cs.cornell.edu/~tanzeem/pubs/mobile_phone_survey.pdf

7. http://www.android.com

8. Paul Deitel, Harvey Deitel, Abbey Deitel (2014) Android προγραμματισμός Δεύτερη έκδοση, *Μ.Γκιούρδας*

9. http://developer.android.com/about/versions/lollipop.html

10. Cristopher Allen, Shannon Appelcline (2009) iPhone in Action, *Manning Publications Co.*

11. https://en.wikipedia.org/wiki/IPhone_3GS

12. Tracy V. Wilson, Wesley Fenlon (2009) How the iPhone Works

13. https://en.wikipedia.org/wiki/Notification_Center

14. https://en.wikipedia.org/wiki/Apple_A6

15. https://en.wikipedia.org/wiki/IPhone_5

16. https://en.wikipedia.org/wiki/Apple_A7

17. http://www.apple.com

18. Jonathan Mosen, Bill Holton (2014) iPhone 6 and iOS 8: A Look at Accessibility with the Help of iOS Without the Eye

19. ΒΙΔΑΛΗ ΙΑΚΩΒΙΝΑ (2014) Ανάπτυξη εφαρμογής για εύκολη χρήση μέσα σε οχήματα για κινητά Windows Phone 8
ΑΝΩΤΑΤΟ ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΘΕΣΣΑΛΙΑΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

19. https://en.wikipedia.org/wiki/Windows_Phone_7

20. Tor-Morten Grønli et all (2014) Mobile application platform heterogeneity : Android vs Windows Phone vs iOS vs Firefox OS

21. https://msdn.microsoft.com/en-us/library/windows/apps/jj863494.aspx#account_markets

22. Ben Hammersley (2003) Content Syndication with RSS First Edition, *O'reilly*

23. Maged N. Kamel Boulos and Steve Wheeler (2007)
The emerging Web 2.0 social software: an enabling suite of sociable technologies in health and health care education

24. Howard Justin Glaser, David Robert Draeger (2008) Really simple syndication (rss) feed customization

25. Judith Wusteman (2004) RSS: the latest feed First published in Library Hi Tech Volume 22 Number 4

26. https://en.wikipedia.org/wiki/JSON

27. Norio Kobayashi et all (2011) Semantic-JSON: a lightweight web service interface for Semantic Web contents integrating multiple life science databases.

28. Nurzhan Nurseitov et all, Comparison of JSON and XML Data Interchange Formats: A Case Study
http://www.cs.montana.edu/izurieta/pubs/caine2009.pdf

29. Vladimir Geroimenko et all (2006) Visualizing the Semantic Web, XML-Based Internet an Information Visualization Second Edition, *Spinger*

30. Robert Eckstein (1999) XML Pocket Reference First Edition, *O'Reilly*

31. http://developer.android.com/guide/topics/media/mediaplayer.html

32. http://developer.android.com/training/managing-audio/volume-playback.html

33. http://developer.android.com/intl/zh-cn/sdk/installing/installing-adt.html

34. http://developer.android.com/guide/topics/data/data-storage.html

https://developer.apple.com/library/ios/documentation/IDEs/Conceptual/AppDistributionGuide/SubmittingYourApp/SubmittingYourApp.html#//apple_ref/doc/uid/TP40012582-CH9-SW2