Τ.Ε.Ι. ΠΕΛΟΠΟΝΝΗΣΟΥ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΙΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Υλοποίηση εφαρμογής για καταγραφή ηχητικών δεδομένων σε περιβάλλον Android

Φίλιππος Α. Γράψας

Επιβλέπον Καθηγητής
Αριστομένης Θανόπουλος

Σπάρτη, 2016

ΔΗΛΩΣΗ ΜΗ ΛΟΓΟΚΛΟΠΗΣ ΚΑΙ ΑΝΑΛΗΨΗΣ ΠΡΟΣΩΠΙΚΗΣ ΕΥΘΥΝΗΣ

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ενυπογράφως ότι είμαι αποκλειστικός συγγραφέας της παρούσας Πτυχιακής Εργασίας, για την ολοκλήρωση της οποίας κάθε βοήθεια είναι πλήρως αναγνωρισμένη και αναφέρεται λεπτομερώς στην εργασία αυτή. Έχω αναφέρει πλήρως και με σαφείς αναφορές, όλες τις πηγές χρήσης δεδομένων, απόψεων, θέσεων και προτάσεων, ιδεών και λεκτικών αναφορών, είτε κατά κυριολεξία είτε βάση επιστημονικής παράφρασης. Αναλαμβάνω την προσωπική και ατομική ευθύνη ότι σε περίπτωση αποτυχίας στην υλοποίηση των ανωτέρω δηλωθέντων στοιχείων, είμαι υπόλογος έναντι λογοκλοπής, γεγονός που σημαίνει αποτυχία στην Πτυχιακή μου Εργασία και κατά συνέπεια αποτυχία απόκτησης του Τίτλου Σπουδών, πέραν των λοιπών συνεπειών του νόμου περί πνευματικών δικαιωμάτων. Δηλώνω, συνεπώς, ότι αυτή η Πτυχιακή Εργασία προετοιμάστηκε και ολοκληρώθηκε από εμένα προσωπικά και αποκλειστικά και ότι, αναλαμβάνω πλήρως όλες τις συνέπειες του νόμου στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δε μου ανήκει διότι είναι προϊόν λογοκλοπής άλλης πνευματικής ιδιοκτησίας.

Όνομα και Επώνυμο Συγγραφέα (Με Κεφαλαία):

……..

Υπογραφή (Ολογράφως, χωρίς μονογραφή):

……..

Ημερομηνία (Ημέρα – Μήνας – Έτος):

……..

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον καθηγητή μου κ. Θανόπουλο Αριστομένη, για την πολύτιμη αρωγή και καθοδήγησή του καθ' όλη τη διάρκεια υλοποίησης της παρούσας πτυχιακής εργασίας.

Προς το σύνολο του ανθρώπινου δυναμικού του Τ.Ε.Ι. Πελοποννήσου, της σχολής Τεχνολογικών Εφαρμογών, του τμήματος Μηχανικών Πληροφορικής Τ.Ε. θα ήθελα εκφράσω την ευγνωμοσύνη μου για την επιστημονική γνώση, τις εμπειρίες και τα βιώματα που εισέπραξα κατά τα έτη φοίτησής μου.

Τέλος, ιδιαίτερες ευχαριστίες επιθυμώ να απευθύνω στους γονείς μου και τη σύζυγό μου για την υποστήριξη, την υπομονή και την εμπιστοσύνη τους προς εμένα, σεβόμενοι το αντικείμενο των σπουδών μου και τις επαγγελματικές μου επιλογές.

Πίνακας Περιεχομένων

1. Εισαγωγή
    1.1. Στόχοι
    1.2. Δομή
2. Εργαλεία Ανάπτυξης
    2.1. Git
    2.2. GitHub.com
    2.3. Android-Studio
    2.4. GIMP
    2.5. Inkscape
3. Stream Recorder
    3.1. Ανάπτυξη εφαρμογών για το περιβάλλον Android
        3.1.1. Δικαιώματα Πρόσβασης
        3.1.2. Fragmentation
        3.1.3. Fragmentation - Μελέτη περίπτωσης: Σύστημα αρχείων
        3.1.4. Fragmentation - Μελέτη περίπτωσης: DialogFragment & Elevation
    3.2. Εγγραφή από δέκτη ραδιοφώνου
    3.3. Εγγραφή από μικρόφωνο
    3.4. Ρυθμίσεις Εγγραφής από Μικρόφωνο
    3.5. Εγγραφή από Ροή μέσω Διαδικτύου
    3.6. StreamRecorder-URLsParser
4. Συμπεράσματα
5. Ιστογραφία
6. Παραρτήματα
    6.1. Torn Out effect (GIMP Script)
    6.2. Numbered Chapters (Google Apps Script)
    6.3. Κώδικας
        6.3.1. MediaURL.java
        6.3.2. Recorder.java
        6.3.3. MicRecorder.java
        6.3.4. StreamRecorder.java
        6.3.5. IO.java
        6.3.6. Misc.java
        6.3.7. IOV16.java
        6.3.8. IOV21.java
        6.3.9. MyLog.java
        6.3.10. FileListItem.java
        6.3.11. FavoritesURLs.java
        6.3.12. ComparableFiles.java
        6.3.13. MediaPlayerView.java
        6.3.14. MediaRecorderView.java
        6.3.15. ViewPagerListener.java

6.3.16. ComparableDocumentFile.java
6.3.17. DeleteFile.java
6.3.18. AddFavoriteURL.java
6.3.19. DeleteFavoriteURL.java
6.3.20. FavoritesURLsAdapter.java
6.3.21. Exception.java
6.3.22. IOException.java
6.3.23. NeedActivityException.java
6.3.24. NeedWorkingDirectoryException.java
6.3.25. MicRecordsFragment.java
6.3.26. StreamsRecordsFragment.java
6.3.27. FavoritesURLsActivityFragment.java
6.3.28. MyActivity.java
6.3.29. MainActivity.java
6.3.30. FavoritesURLsActivity.java
6.3.31. OnDataChanged.java
6.3.32. OnPageChangeListener.java
6.3.33. App.java

1. Εισαγωγή

Αντικείμενο της παρούσας πτυχιακής εργασίας είναι η μελέτη των δυνατοτήτων καταγραφής ηχητικών δεδομένων από διάφορες πηγές ήχου που δύναται να προσφέρει μια συσκευή με την πλατφόρμα Android καθώς και η υλοποίηση σχετικής εφαρμογής για την εκμετάλλευση των παραπάνω δυνατοτήτων. Αρχικά, εκπονήθηκε έρευνα για την εκμετάλλευση των τριών κύριων πηγών ήχου: το μικρόφωνο, τον ραδιοφωνικό δέκτη και την ροή ήχου μέσω διαδικτύου. Στην συνέχεια, αναπτύχθηκε σχετική εφαρμογή για την πλατφόρμα Android και παράλληλα πραγματοποιήθηκε η συγγραφή της παρούσας πτυχιακής εργασίας.

1.1. Στόχοι

Αρχικός στόχος της παρούσας πτυχιακής εργασίας ήταν η αναζήτηση των διαθέσιμων πηγών ήχου μιας συσκευής με την πλατφόρμα Android. Συγκεκριμένα έγινε ανάλυση και εκμετάλλευση του μικροφώνου και της ροής ήχου μέσω διαδικτύου. Η εκμετάλλευση του ραδιοφώνου δεν κατέστη δυνατή και οι λόγοι αναλύονται στο αντίστοιχο κεφάλαιο.

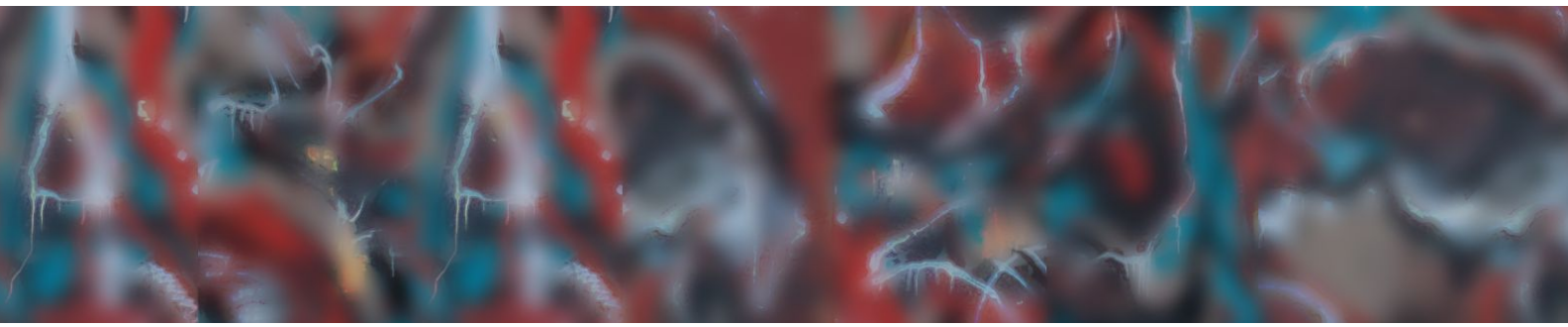
Στην συνέχεια, δεύτερος στόχος ήταν η υλοποίηση σχετικής εφαρμογής με βάση την παραπάνω μελέτη. Ταυτόχρονα όμως, η ανάπτυξη της εφαρμογής έγινε με βάση τις απαιτήσεις μιας εμπορικής εφαρμογής που απευθύνεται στο ευρύ κοινό. Έτσι δόθηκε ιδιαίτερη έμφαση στην Γραφική Διεπαφή Χρήστη και στην υποστήριξη όσο το δυνατό περισσότερων εκδόσεων της πλατφόρμας Android.

Συγχρόνως, έγινε η συγγραφή του θεωρητικού μέρους της παρούσας εργασίας. Ιδιαίτερη έμφαση δόθηκε τόσο στα κείμενα όσο και στα διαγράμματα και στα γραφικά.

1.2. Δομή

Στο κεφάλαιο «Εργαλεία ανάπτυξης» παρουσιάζονται όλα τα εργαλεία και τα προγράμματα που με την βοήθεια των οποίων υλοποιήθηκε το σύνολο του έργου και που αφορά τόσο τα εργαλεία ανάπτυξης λογισμικού όσο και τα εργαλεία για την παραγωγή των διαγραμμάτων και των γραφικών. Το κεφάλαιο «Stream Recorder» αφορά στην ανάπτυξη λογισμικού και παρουσιάζονται διεξοδικά όλες οι πλευρές της εφαρμογής της πλατφόρμας Android όσο και διάφορων βοηθητικών εργαλείων που αναπτύχθηκαν.

2. Εργαλεία Ανάπτυξης



Κατά την υλοποίηση της παρούσας πτυχιακής εργασίας, αξιοποιήθηκε σύνολο προγραμμάτων για την ανάπτυξη του λογισμικού, την συγγραφή της και τον καλλωπισμό της. Χρησιμοποιήθηκαν προγράμματα για την συγγραφή του κώδικα, για την διαχείριση των εκδόσεων του κώδικα, για την επεξεργασία ψηφιογραφικών εικόνων, για την επεξεργασία διανυσματικών εικόνων καθώς και για την επεξεργασία διαγραμμάτων και γραφημάτων. Όλα τα παραπάνω παρουσιάζονται συνοπτικά στις επόμενες ενότητες του κεφαλαίου.

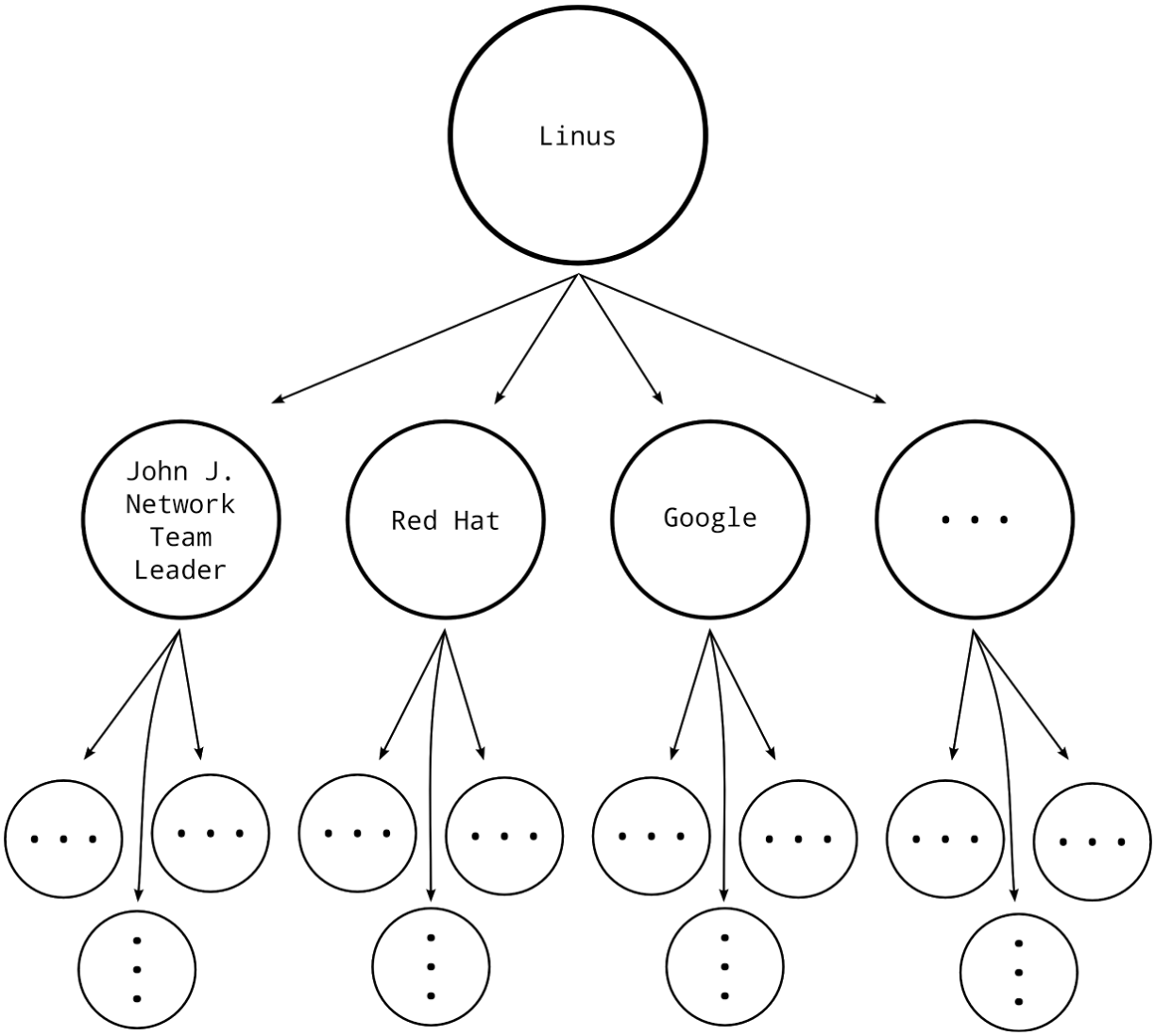
2.1. Git

Πρόκειται για ένα σύστημα ελέγχου εκδόσεων λογισμικού με την βοήθεια του οποίου γίνεται δυνατή η διατήρηση όλου του ιστορικού των αλλαγών του κώδικα και παρέχει δυνατότητες σύγκρισης μεταξύ τους, επαναφοράς τους, υπό συνθήκη συγχώνευσής τους. Επίσης, παρέχει δυνατότητες συνεργασίας ανάμεσα στους προγραμματιστές. Πρόκειται για ένα απαραίτητο σύστημα τόσο από πολύ μεγάλα όσο και από πολύ μικρά έργα.

Το σύστημα Git σχεδιάστηκε και αναπτύχθηκε το 2005 αρχικά από τον φημισμένο Λίνους Τόρβαλντς για την διαχείριση του λειτουργικού συστήματος Linux. Είναι ελεύτερο λογισμικό και είναι ανεξάρτητο πλαφόρμας. Πλέον, χρησιμοποιείται ευρέος σε όλον τον κόσμο τόσο σε μεγάλα όσο και σε μικρά έργα ανάπτυξης λογισμικού και αποτελεί την πιο διαδεδομένη λύση. Η μεγάλη πρωτοτυπία του σε σχέση με τα τότε γνωστά συστήματα ήταν ότι υλοποιήθηκε με βάση το μοντέλο των κατανεμημένων συστημάτων και δεν απαιτούσε απαραίτητα το μοντέλο πελάτη - διακομιστή για να λειτουργήσει

Το μοντέλο κατανεμημένου συστήματος πλέον φαντάζει κάτι αυτονόητο αν σκεφτεί κανείς τον τρόπο ανάπτυξης του λειτουργικού συστήματος Linux. Μπορούμε να φανταστούμε την λειτουργία του ως έναν γράφο με το παρακάτω σενάριο: Στον ριζικό κόμβο βρίσκεται ο Linus που έχει υπό ευθύνη του ολόκληρο το έργο και

συνεργάζεται είτε με κάποιους ανθρώπους που εμπιστεύεται είτε με κάποια εταιρία που συνεισφέρει στην ανάπτυξη του kernel. Κάποιος από τους συνεργάτες μπορεί να είναι ο John J. που είναι υπεύθυνος για τους οδηγούς των δικτυακών συσκευών και έχει βαθιά εξειδίκευση στον τομέα. Ο John, γνωρίζει και συνεργάζεται είτε με κάποιους ανθρώπους είτε με την ομάδα κάποιας εταιρίας που αναπτύσσει τον οδηγό για την δικτυακή της συσκευή. Και ο γράφος συνεχίζει ώστε να συμπεριλάβει τους χιλιάδες προγραμματιστές που συνεισφέρουν συστηματικά στην ανάπτυξη του kernel και τους προγραμματιστές που απλώς δημοσίευσαν ένα patch.



2.2. GitHub.com

Το GitHub.com είναι ένα κοινωνικό δίκτυο προγραμματιστών με θέμα την ανάπτυξη λογισμικού και προσφέρει υπηρεσίες φιλοξενίας αποθετηρίων Git. Το σύστημα Git όπως αναφέρθηκε στην προηγούμενη ενότητα βασίζεται στο κατανεμημένο μοντέλο. Όμως, προαιρετικά, υποστηρίζει ταυτόχρονα και το μοντέλο πελάτης - διακομιστής. Έτσι, ενώ κάθε προγραμματιστής μπορεί να έχει στο σύστημα αρχείων του το τοπικό του αποθετήριο και να στέλνει patch στον υπεύθυνο του έργου, ταυτόχρονα, ενώ έχει το τοπικό του αποθετήριο, μπορεί να στέλνει τις αλλαγές του απευθείας σε κάποιον διακομιστή που έχει πρόσβαση, είτε αυτός είναι προσωπικός είτε αυτός είναι ο κεντρικός διακομιστής του έργου.

Το GitHub.com παρέχει εργαλεία και κάνει εύκολη την συνεργασία μεταξύ προγραμματιστών για την ανάπτυξη λογισμικού. Όσο αφορά τα έργα ανοικτού ή και ελεύθερου λογισμικού, που ο κώδικάς τους είναι δημόσιος, η υπηρεσία παρέχει τα αποθετήρια δωρεάν. Όσο αφορά τα έργα που ο κώδικάς τους δεν είναι δημόσιος, παρέχει τα αποθετήρια επί πληρωμή. Επίσης, προσφέρει μη δημόσια αποθετήρια δωρεάν σε φοιτητές και σε καθηγητές σε επίπεδο εργαστηρίου.

2.3. Android-Studio

Το Android Studio είναι το επίσημο IDE (Integrated Development Environment - Ολοκληρωμένο Περιβάλλον Ανάπτυξης) για την δημιουργία εφαρμογών για την πλατφόρμα Android. Αναπτύσσεται και συντηρείται από την Google αλλά είναι βασισμένο στο IntelliJ IDEA. Πρόκειται για ένα ιδιαίτερα εύχρηστο IDE με πάρα πολλές δυνατότητες και εργαλεία που κάνει την ανάπτυξη εφαρμογών όσο το δυνατόν ευκολότερη και αντισταθμίζει κάπως την περιπλοκότητα της πλατφόρμας Android.

Όλα αυτά τα εργαλεία και οι ευκολίες έρχονται με ένα κόστος σε απαιτήσεις πόρων από τον σταθμό εργασίας τους προγραμματιστή. Εμπειρικά, απαιτούνται 8gB RAM και τουλάχιστον 4 πυρήνες KME () για την ανάπτυξη των εφαρμογών. Είναι βασισμένο στην γλώσσα προγραμματισμού Java και παρέχει ανεξαρτησία πλατφόρμας υποστηρίζοντας GNU+Linux, Windows και OS X.

2.4. GIMP

Το GIMP (GNU Image Manipulation Program) είναι πρόγραμμα επεξεργασίας ψηφιογραφικής εικόνας και απευθύνεται σε απαιτητικούς χρήστες. Η πρώτη του έκδοση δημοσιεύτηκε το 1996. Είναι μέλος του GNU Project και φυσικά είναι ελεύθερο λογισμό και δωρεάν. Επίσης, είναι ανεξάρτητο πλατφόρμας και υποστηρίζει τις GNU+Linux, OS X, Microsoft Windows, BSD, Solaris, AmigaOS 4.

Χρησιμοποιήθηκε για την δημιουργία και επεξεργασία όλων των γραφικών της παρούσας πτυχιακής εργασίας είτε αυτόνομα είτε σε συνδυασμό με το πρόγραμμα επεξεργασίας διανυσματικών εικόνων Inkscape. Αξίζει να σημειωθεί, πως έγινε επέκταση των δυνατοτήτων του μέσω προσθέτου που αναπτύχθηκε για αυτήν την πτυχιακή (βλ 4.7 Torn Out effect).

2.5. Inkscape

Το Inkscape είναι πρόγραμμα επεξεργασίας διανυσματικών εικόνων και η ανάπτυξή του ξεκίνησε το 2003 ως fork του Sodipodi. Είναι ελεύθερο λογισμικό και δωρεάν, καθώς επίσης είναι και ανεξάρτητο πλατφόρμας με υποστήριξη για GNU+Linux, Windows και Mac OS X. Υποστηρίζει την επεξεργασία διάφορων μορφών αρχείων αλλά ως κύρια μορφή χρήση του SVG (Scalable Vector Graphics), το οποίο είναι ανοικτό πρότυπο και αναπτύχθηκε από το W3C (World Wide Web Consortium).

Χρησιμοποιήθηκε για την δημιουργία όλων των διανυσματικών εικόνων και των διαγραμμάτων της παρούσας πτυχιακής εργασίας. Κατά κύριο λόγο έγινε συνδυασμός με το GIMP.

3. Stream Recorder

Ιστότοπος: https://github.com/GrapsasFilippos/StreamRecorder

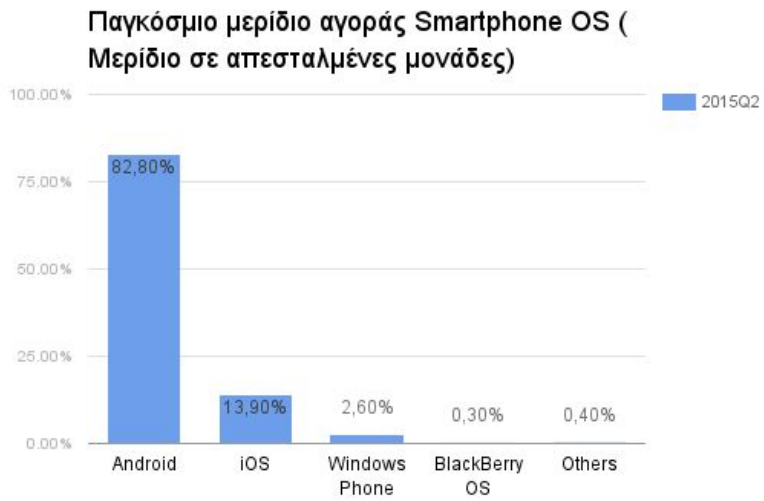
Άδεια: GPLv3 (https://www.gnu.org/licenses/gpl-3.0.txt)

Η εφαρμογή Stream Recorder δημιουργήθηκε ώστε να καλύψει το πρακτικό μέρος της πτυχιακής εργασίας. Αναπτύχθηκε για την πλατφόρμα Android και με βάση τις απαιτήσεις, στόχο έχει την εγγραφή ροής ήχου από όλες τις διαθέσιμες πηγές ήχου που μπορεί να προσφέρει μια συσκευή με την παραπάνω πλατφόρμα. Συγκεκριμένα, μελέτη πραγματοποιήθηκε για την εγγραφή από ραδιοφωνικό δέκτη όταν αυτός συμπεριλαμβάνεται στο υλικό, για την εγγραφή από το μικρόφωνο όπου είναι εφοδιασμένες όλες οι συσκευές κινητών τηλεφώνων και ταμπλετών και για την εγγραφή ροής ήχου από ραδιοφωνικούς σταθμούς μέσω διαδικτύου.

Η εφαρμογή αποθηκεύει όλες τις εγγραφές, ανεξαρτήτως πηγής, στο σύστημα αρχείων της συσκευής και αργότερα είτε δύναται να τις αναπαράγει η ίδια η εφαρμογή, είτε να γίνει η αναπαραγωγή από άλλη εφαρμογή στην ίδια συσκευή, είτε να γίνει η αναπαραγωγή από άλλη συσκευή (π.χ. PC), αφού πρώτα μεταφερθούν τα σχετικά αρχεία με την βοήθεια των εργαλείων του λειτουργικού συστήματος.

3.1. Ανάπτυξη εφαρμογών για το περιβάλλον Android

Η πλατφόρμα Android είναι η πιο διαδεδομένη στην αγορά των έξυπνων τηλεφώνων με ποσοστό 82,8% και με δεύτερη την iOS να ακολουθεί με ποσοστό 13,9% το δεύτερο τρίμηνο του 2015 σύμφωνα με την IDC[1] (International Data Corporation). Η Alphabet Inc. (γνωστή ως Google, θυγατρική πλέον της Alphabet), είναι από τις μεγαλύτερες πολυεθνικές εταιρίες στον τομέα της πληροφορικής με έντονη δραστηριοποίηση στην ανάπτυξη λογισμικού. Ως εκ τούτου, κατέχει την απαραίτητη τεχνογνωσία για τον σχεδιασμό, την ανάπτυξη και την συντήρηση της πλατφόρμας Android με τον καλύτερο δυνατό τρόπο.



[1] Smartphone OS Market Share, 2015 Q2 - http://www.idc.com/prodserv/smartphone-os-market-share.jsp

Οι εφαρμογές που αναπτύσσονται, μπορούν να διανεμηθούν στην παγκόσμια αγορά με πολύ εύκολο τρόπο μέσω του ηλεκτρονικού καταστήματος Google Play Store που είναι προεγκατεστημένο στην συντριπτική πλειοψηφία των συσκευών με Android. Αυτό, δίνει την ευκαιρία σε μικρές εταιρίες ανάπτυξης λογισμικού ή ακόμα και σε αυτόνομους προγραμματιστές, να προωθήσουν την δουλειά τους στην παγκόσμια αγορά που υπό άλλες συνθήκες θα τους ήταν ιδιαίτερα δύσκολο ή και αδύνατο. Ταυτόχρονα, υπάρχει και η δυνατότητα προώθησης των εφαρμογών από άλλα καταστήματα ή η αυτόνομη προώθηση. Όμως οι εναλλακτικοί αυτοί τρόποι προώθησης είναι αντιπαραγωγικοί καθώς το Google Play Store είναι ο πετυχημένος τρόπος.

Η παγκοσμιοποίηση της αγοράς και η ευκολία κάποιου να εισέλθει σε αυτήν, δεν αυξάνει μόνο το αγοραστικό κοινό αλλά αυξάνει αντίστοιχα και τον ανταγωνισμό. Ταυτόχρονα, η δεσπόζουσα θέση του Google Play Store σε παγκόσμια κλίμακα, το κάνει εκ των πραγμάτων δυσκίνητο και το ωθεί σε αστοχίες. Οι έλεγχοι που αφορούν στην τήρηση της νομοθεσίας, στην ασφάλεια λογισμικού και στην τήρηση των όρων χρήσης αυστηροποιούνται και εντατικοποιούνται. Αυτό σημαίνει πως εάν κάποιος μεμονωμένος προγραμματιστής βρεθεί μπροστά σε μια τέτοια αστοχία ελέγχου, θα του είναι δύσκολο να τραβήξει την προσοχή της Google για την επίλυσή της.

* Εικόνα από https://developer.android.com/

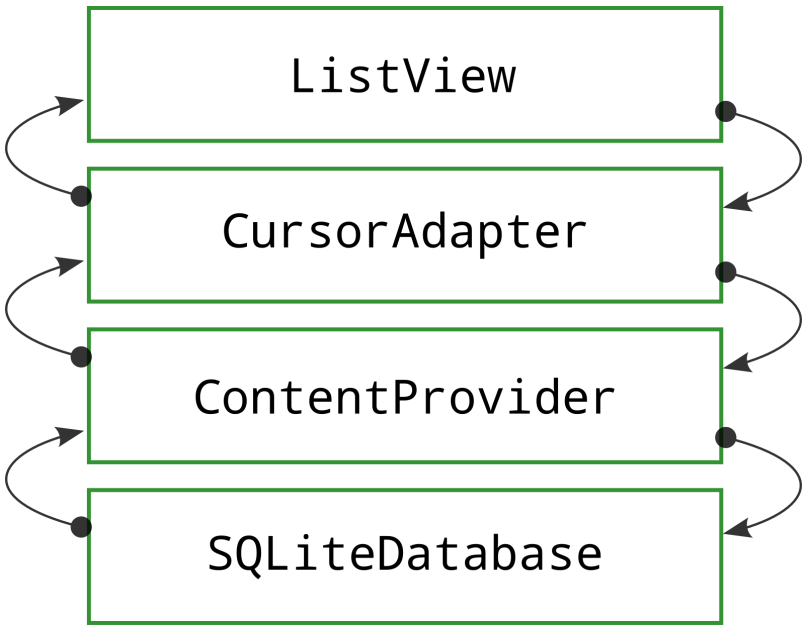
Η πλατφόρμα Android έχει ως βάση το λειτουργικό σύστημα Linux. Η Διεπαφή Προγραμματισμού Εφαρμογών (API) καθώς η ανάπτυξη των εφαρμογών γίνεται με χρήση της γλώσσας προγραμματισμού Java και η περιγραφή της Γραφικής Διεπαφής Χρήστη (GUI - Graphical User Interface) γίνεται κατά βάση με χρήση της XML. Επίσης, παρέχει υποστήριξη για native κώδικα σε C και C++ μέσω του NDK (Native Development Kit) για χρήση ή επαναχρησιμοποίηση υπαρχουσών βιβλιοθηκών καθώς και για περιπτώσεις που απαιτούνται αυξημένες επιδόσεις π.χ. σε παιχνίδια ή προσομοιώσεις φυσικής.

Για να διασφαλιστεί το όλο εγχείρημα της πλατφόρμας και η ποιότητα των εφαρμογών από προγραμματιστές από όλες της γωνιές του κόσμου, η Διεπαφή Προγραμματισμού Εφαρμογών (API) ωθεί σε ολοένα και αυξανόμενες Γραμμές Πλεύσης (Software Development Guidelines). Η ποιότητα στο λογισμικό είναι κάτι σημαντικό που όλοι θα ήθελαν, αλλά ακολουθείται από ένα κόστος στην εκμάθηση της πλατφόρμας και στην αύξηση και διατήρηση του επιπέδου της τεχνογνωσίας.

Χαρακτηριστικό παράδειγμα Γραμμής Πλεύσης θα μπορούσε να είναι η συμπλήρωση μιας απλής λίστας με δεδομένα από μια Βάση Δεδομένων. Ένας προγραμματιστής που ξεκινάει με την πλατφόρμα Android, θα έκανε τον εξής πρόχειρο σχεδιασμό στο μυαλό του: Εκτέλεση ενός SQL ερωτήματος στην ΒΔ και μιας δομή επανάληψης να συμπληρώνει την λίστα στοιχείο - στοιχείο.

Ωστόσο κάτι τέτοιο δεν ισχυεί. Η Γραμμή Πλεύσης ορίζει κάτι πολύ διαφορετικό και αρκετά πιο περίπλοκο: Η λίστα (ListView) ενημερώνεται για ένα αντικείμενο Adapter, το οποίο είναι υπεύθυνο για την παροχή των απαιτούμενων δεδομένων στην λίστα. Αυτός ο Adapter, ζητάει τα απαραίτητα δεδομένα από το ContentProvider το οποίο με την σειρά του τα ζητά από την Βάση Δεδομένων. Η επικοινωνία όλων αυτών των οντοτήτων ολοκληρώνεται με την αντίστροφη επικοινωνία για την επιστροφή των δεδομένων. Σε όλη αυτήν την υλοποίηση, θα πρέπει να προστεθούν και οι αντίστοιχες βοηθητικές οντότητες για την μεταξύ τους

επικοινωνία. Ακολουθεί ένα απλοποιημένο διάγραμμα για τον παραπάνω πρόχειρο σχεδιασμό:



Με το παραπάνω παράδειγμα γίνεται εύκολα αντιληπτό πως μια φαινομενικά απλή στην υλοποίηση απαίτηση, απαιτεί έναν σχετικά περίπλοκο σχεδιασμό και υλοποίηση. Ειδικά για κάποιον που βρίσκεται στο στάδιο εκμάθηση της πλατφόρμας και επιθυμεί να ξεκινήσει με μια απλή εφαρμογή που πιθανότατα χρειάζεται λίστα με δεδομένα από ΒΔ.

Όμως, με όλο αυτό το κόστος λύνονται πολλά προβλήματα. Αρχικά, για κάποιον με εμπειρία είναι μια καθαρή λύση (αναμενόμενη/γνώριμη/τυπική υλοποίηση).

Επίσης, ο πρώτος απλός σχεδιασμός με την απλή δομή επανάληψης, θα δημιουργούσε προβλήματα μη αποκρισιμότητας («παγώματος»/freeze) στην Γραφική Διεπαφή Χρήστη αφού και η SQL ερώτηση, εκτέλεση και το γέμισμα της λίστας θα γίνονταν στο Κύριο Νήμα (main thread/GUI thread). Κρίσιμο πρόβλημα που δίνει την εντύπωση στον χρήστη ότι η συσκευή του δεν αποκρίνεται. Με βάση

την Γραμμή Πλεύσης, το πρόβλημα είναι λυμένο αφού όλη η χρονοβόρα διαδικασία γίνεται σε ξεχωριστό νήμα.

Βέβαια, νήματα θα μπορούσαν να χρησιμοποιηθούν και στον απλό, πρώτο σχεδιασμό. Αυτό όμως θα άφηνε πολλές πιθανότητες σε κακογραμμένες εφαρμογές ή σε εφαρμογές με εκτεταμένη χρήση νημάτων να τα αφήνουν ορφανά. Το αποτέλεσμα είχε πάλι σημαντικότατο πρόβλημα. Τα σταματημένα ορφανά νήματα θα οδηγούσαν σε διαρροή μνήμης και τα μη σταματημένα θα οδηγούσαν και σε κατανάλωση επεξεργαστικής ισχύς και κατ επέκταση ενέργειας, πράγμα πολύ σημαντικό για τις φορητές συσκευές με μπαταρία.

Φυσικά, ο προγραμματιστής, στον πρώτο σχεδιασμό, θα μπορούσε να αναπτύξει κάποιου είδους απλή βιβλιοθήκη που θα αυτοματοποιούσε τις διαδικασίες ώστε να μειώσει αποτελεσματικά την πιθανότητα ορφανών νημάτων. Αυτό όμως θα απαιτούσε επιπρόσθετο κόστος και θα αύξανε την περιπλοκότητα σχετικά κοντά στην περιπλοκότητα της ήδη υπάρχουσας Γραμμής Πλεύσης της ίδιας της πλατφόρμας.

Η Γραμμή Πλεύσης στο παραπάνω παράδειγμα γίνεται ακόμα πιο ελκυστική αν ληφθεί υπόψιν πως πολλές από τις οντότητές της είναι επαναχρησιμοποιούμενες. Για παράδειγμα, η οντότητα Adapter χρησιμοποιείται για να συμπληρώσει λίστες είτε η πηγή των δεδομένων είναι μια Βάση Δεδομένων, είτε ένας πίνακας τιμών, είτε ένα αρχείο, είτε το Διαδίκτυο. Η οντότητα ContentProvider μπορεί να χρησιμοποιηθεί πολλαπλά από την ίδια την εφαρμογή για την συμπλήρωση λιστών, είτε για τον συγχρονισμό δεδομένων με άλλες συσκευές, είτε από άλλες εφαρμογές. Για παράδειγμα, οι Τηλεφωνικές Επαφές είναι προσβάσιμες από τις εφαρμογές μέσω του ειδικού προϋπάρχοντος ContentProvider.

Ένα ακόμα παράδειγμα Γραμμής Πλεύσης ακολουθεί παρακάτω (Μελέτη περίπτωσης: Σύστημα Αρχείων) που μάλιστα έχει συνδυαστεί με Fragmentation. Από μια συγκεκριμένη έκδοση Android και κάτω, η Γραμμή Πλεύσης δεν υποστηρίζεται, ενώ από μια συγκεκριμένη έκδοση Android και πάνω είναι

αναγκαστική. Αυτό σημαίνει πως μια συγκεκριμένη εφαρμογή, για να υποστηρίξει δύο διαφορετικές εκδόσεις Android, πρέπει να έχει υλοποιηθεί η Γραμμή Πλεύσης αλλά θα πρέπει να έχει υλοποιηθεί και χωρίς αυτή ταυτόχρονα για να πετύχει το ίδιο αποτέλεσμα σε δύο διαφορετικές εκδόσεις Android.

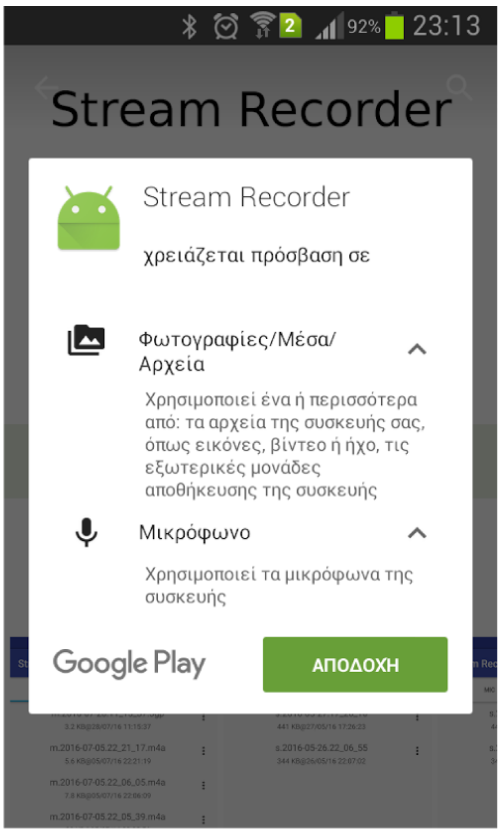
3.1.1. Δικαιώματα Πρόσβασης

Η παροχή πρόσβασης στην χρήση του μικροφώνου από τις εφαρμογές απαιτεί ιδιαίτερη προσοχή. Με την παραπάνω πρόσβαση μιας κακόβουλης εφαρμογής θα μπορούσε να οδηγήσει σε παρακολούθηση του χρήστη της συσκευής εν αγνεία του. Για αυτό τον λόγο, η πλατφόρμα ενημερώνει τον χρήστη και του ζητά την συγκατάθεσή του κάθε φορά που μια εφαρμογή κάνει χρήση του παραπάνω δικαιώματος πρόσβασης.

Όσο αφορά τις εφαρμογές που αναπτύσσονται με στόχο το APIv22 (5.1/Lollipop with an extra sugar coating on the outside!) και πίσω, ο χρήστη χρειάζεται να αξιολογήσει και να επιτρέψει ή να απορρίψει το σύνολο των δικαιωμάτων που απαιτεί η εφαρμογή κατά την εγκατάστασή της. Αν τα επιτρέψει, η εγκατάσταση της εφαρμογής προχωρά, διαφορά η εγκατάσταση ακυρώνεται. Αυτή η τεχνική δεν λειτούργησε ιδιαίτερα καλά αφού ο χρήστης συνήθως είναι ανυπόμονος και δεν αφιερώνει τον απαραίτητο χρόνο για την αξιολόγηση όλων των δικαιωμάτων κατά την εγκατάσταση των εφαρμογών, αφού συνήθως η λίστα είναι μεγάλη.

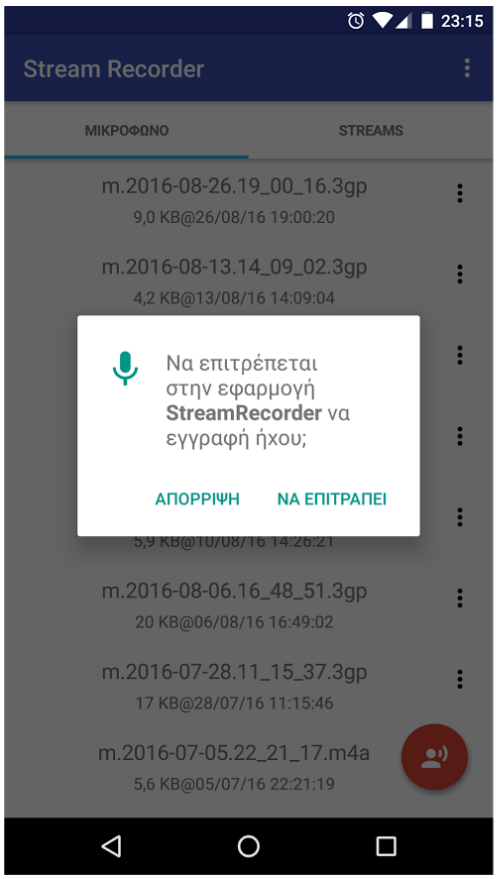
Όσο αφορά τις εφαρμογές που αναπτύσσονται με στόχο το APIv23 (6.0/M is for Marshmallow!) και πλέον, τα πράγματα έχουν βελτιωθεί. Η εγκατάσταση των εφαρμογών γίνεται άμεσα και δεν απαιτείται η αξιολόγηση και αποδοχή των δικαιωμάτων, αλλά αυτό γίνεται κατά την εκτέλεση της εφαρμογής. Έτσι, ο χρήστης, όταν εκτελεί την εφαρμογή για πρώτη φορά και όταν από την εφαρμογή απαιτηθεί η

χρήση κάποιου δικαιώματος, η πλατφόρμα ερωτά τον χρήστη για το αν παρέχει την συγκατάθεσή του για το συγκεκριμένο δικαίωμα πρόσβασης. Ο χρήστης δύναται να το επιτρέψει ή να το απορρίψει επιτόπου. Επιπλέον, ο χρήστης έχει την δυνατότητα να αλλάξει γνώμη και επιλογή από τις ρυθμίσεις της πλατφόρμας ανά πάσα στιγμή.



API level <= 22

Αίτημα συγκατάθεση κατά την εγκατάσταση.



API level >= 23

Αίτημα συγκατάθεσης αφού ο χρήστης επέλεξε ηχογράφηση και πριν αυτή ξεκινήσει για πρώτη φορά.

Αναλογία απεικόνισης 1:1

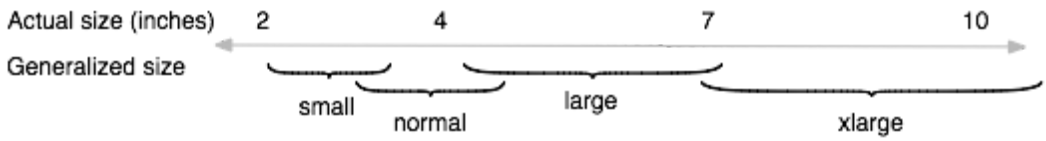
3.1.2. Fragmentation

Android Fragmentation, κατακερματισμός. Ένας πολύ διαδεδομένος όρος, αλλά τι σημαίνει; Κάποιοι αναφέρονται στις διάφορες εκδόσεις του Android, κάποιοι στους πολλούς κατασκευαστές, κάποιοι στις επεμβάσεις των κατασκευαστών στην εμφάνιση της πλατφόρμας, κάποιοι στις διάφορες αναλύσεις και αναλογίες των οθονών, κάποιοι ακόμα στα forks όπως το Amazon Fire OS ή τις διάφορες άλλες διανομές του AOSP (Android Open Source Project). Όλα αυτά είναι κάποια υπερβολή ή όχι. Είναι κάτι καλό ή κάτι κακό;

Οι διάφορες αναλύσεις, αναλογίες και μεγέθη των οθονών δεν είναι κάτι καινούριο ή κάτι που έφερε το Android. Από πάντα οι Π/Υ είχαν διαφορετικές οθόνες και κανείς δεν αναφέρθηκε σε κατακερματισμό. Σίγουρα, όσον αφορά στην σχεδίαση της Γραφικής Διεπαφής Χρήστη, απαιτείται επιπλέον χρόνος και κόστος για να προβλεφθούν όλοι οι δυνατοί συνδυασμοί. Αλλά οι σχεδιαστές και οι προγραμματιστές προσάρμοσαν τις τεχνικές τους όποτε αυτό απαιτήθηκε και οι χρήστες είχαν στη διάθεσή τους πολλές και ποικίλες επιλογές.

Στα παρακάτω δύο γραφήματα, τα στοιχεία αφορούν συσκευές που συνδέονται στο Google Play Store για το διάστημα 7 ημερών που τελειώνει την 1 Αυγούστου 2016 και παρέχονται από την Google. Επίσης, ποσοστά μικρότερα του

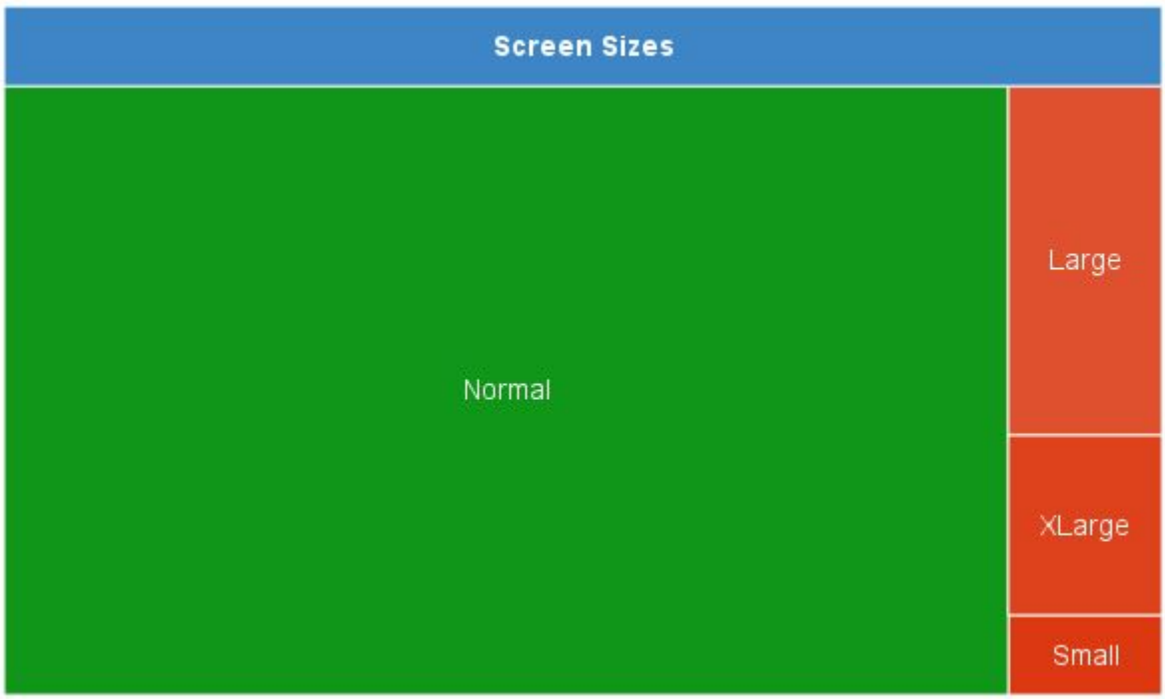
0.1% δεν εμφανίζονται. Το treemap γράφημα Screen Sizes αφορά τις παρακάτω τιμές:

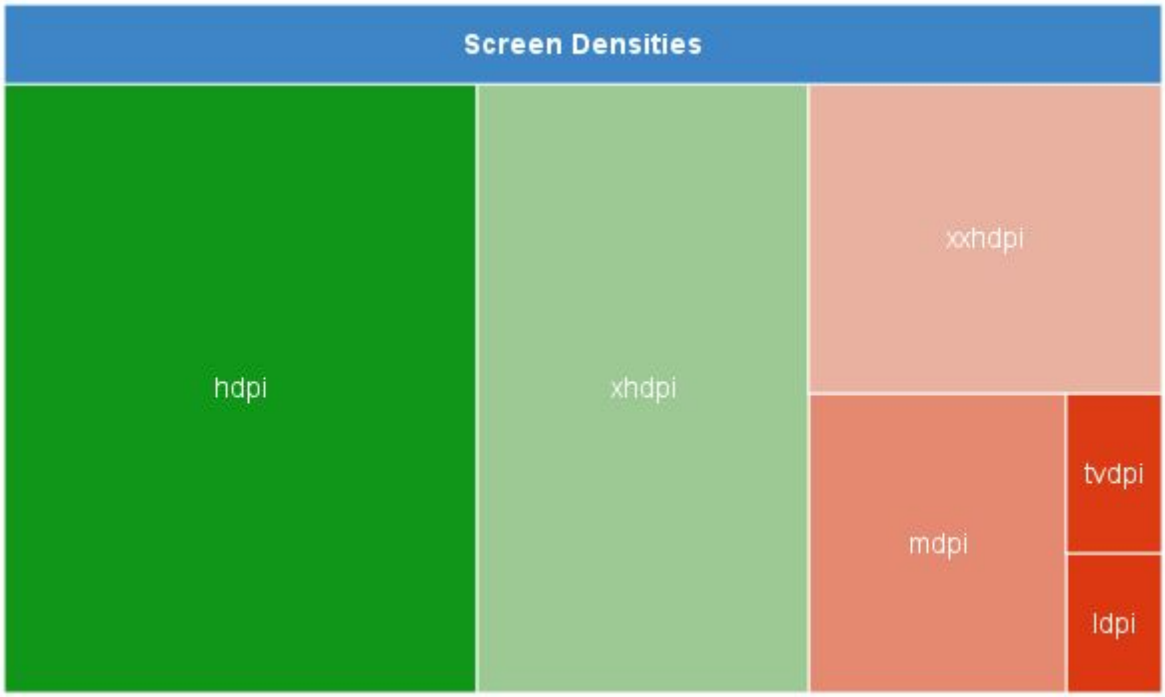


Γράφημα από https://developer.android.com/

Το treemap γράφημα Screen Densities αφορά τις παρακάτω τιμές:

1. ldpi (low) ~120dpi
2. mdpi (medium) ~160dpi
3. hdpi (high) ~240dpi
4. xhdpi (extra-high) ~320dpi
5. xxhdpi (extra-extra-high) ~480dpi
6. xxxhdpi (extra-extra-extra-high) ~640dpi





Από τους πολλούς κατασκευαστές, οι προγραμματιστές δεν επηρεάζονται ιδιαίτερα και οι χρήστες έχουν πολλές επιλογές από συσκευές, χαρακτηριστικά και τιμές για να επιλέξουν. Οι προγραμματιστές, χρησιμοποιούν την Διεπαφή Προγραμματισμού Εφαρμογών της πλατφόρμας Android και βρίσκουν την αντίστοιχη τεκμηρίωση στον αντίστοιχο ιστότοπο της πλατφόρμας και ο παραγόμενος κώδικάς τους λειτουργεί σε όλες ανεξαρτήτως τις συσκευές. Υπάρχουν βέβαια και μεμονωμένες περιπτώσεις όπου απαιτείται η χρήση API κάποιου κατατασκευαστή. Για παράδειγμα, η Samsung έχει δικό της API για την υποστήριξη της εφαρμογής από τον διαχειριστή πολλαπλών παραθύρων. Η χρήση του όμως είναι προαιρετική και όχι διαδεδομένη. Μάλιστα, κατά τον χρόνο συγγραφή της παρούσας πτυχιακής εργασίας, η νέα προς κυκλοφορία έκδοση Android 7.0 (APIv24), παρέχει αυτήν την δυνατότητα.

Όσον αφορά στις πολλαπλές εκδόσεις της πλατφόρμας Android υπάρχει πρόβλημα τόσο για τους χρήστες όσο και για τους προγραμματιστές. Αναφορικά με τους χρήστες, οι συσκευές τους δεν αναβαθμίζονται (ακόμη και όταν οι απαιτήσεις από το υλικό καλύπτονται) και «χάνουν» τις νέες δυνατότητες που προσθέτονται σε

κάνε νέα έκδοση του Android. Ταυτόχρονα, ίσως και πιο το σημαντικό, οι συσκευές τους δεν αναβαθμίζονται ακόμη και για τα διορθωμένα κενά ασφαλείας. Γεγονός που συνιστά σοβαρό πρόβλημα καθώς αυτά τα κενά ασφαλείας δημοσιεύονται με την διόρθωσή τους.

Από την πλευρά των προγραμματιστών, οι διάφορες εκδόσεις συνιστούν πρόβλημα για το οποίο έχουν αφιερωθεί οι επόμενες δύο ενότητες. Σε κάθε μια από αυτές, αναλύονται τα προβλήματα που υπήρξαν στην ανάπτυξη της εφαρμογής της παρούσας πτυχιακής εργασίας όσο και από την ανάπτυξη άλλων εφαρμογών. Στις παρακάτω δύο ενότητες δεν παρουσιάζεται η έκταση του προβλήματος αλλά η δυσκολία του.

Αν και τα προβλήματα είναι πολλά και ποικίλα αναφορικά με τον κατακερματισμό των εκδόσεων Android, οι διαδικασίες αναβάθμισης της πλατφόρμας των συσκευών από τους κατασκευαστές δεν είναι εύκολη υπόθεση και οι εμπλεκόμενοι είναι πολλοί και ίσως με διαφορετικά συμφέροντα. Σε αυτόν τον τομέα τα έχει καταφέρει μόνο η Google μέσω των συσκευών της της σειράς Nexus. Η Google ζητά την κατασκευή των συσκευών από γνωστούς κατασκευαστές όπως η Samsung, η LG, η HTC κτλ, και ύστερα αναλαμβάνει η ίδια το λογισμικό και την αναβάθμισή του.

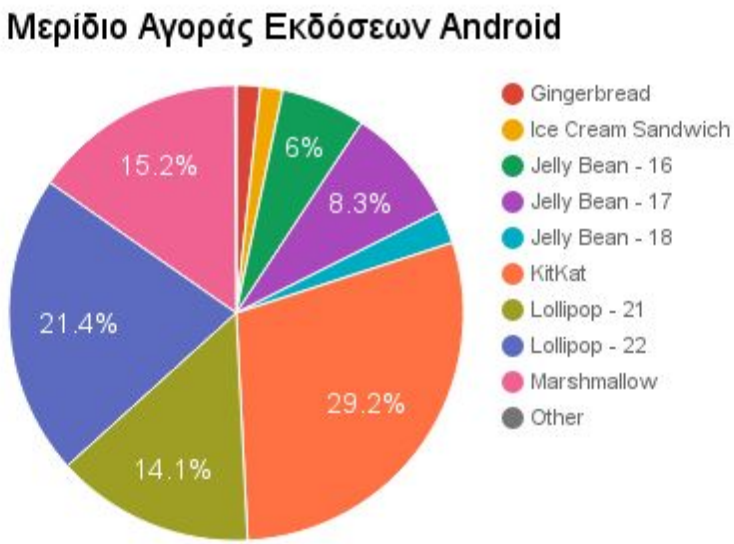
Περιληπτικά, η διαδικασία μιας αναβάθμισης της πλατφόρμας, σύμφωνα με την HTC[2]:

1. Η Google παρέχει στους κατασκευαστές το Platform Development Kit για την αξιολόγηση του νέου framework.
2. Η Google ανακοινώνει την νέα έκδοση της πλατφόρμας.
3. Η Google παρέχει τον κώδικα της νέας έκδοσης στους κατασκευαστές των ολοκληρωμένων κυκλωμάτων.
4. Η Google παρέχει τον κώδικα της νέας έκδοσης στους κατασκευαστές των συσκευών.
5. Οι κατασκευαστές ολοκληρωμένων κυκλωμάτων αποφασίζουν αν θα υποστηρίξουν την νέα έκδοση Android για τα κυκλώματα που έχουν κατασκευάσει.
6. Οι κατασκευαστές συσκευών αξιολογούν την νέα έκδοση Android.
7. Αν οι κατασκευαστές ολοκληρωμένων κυκλωμάτων αποφασίσουν να μην υποστηρίξουν την νέα έκδοση Android, η διαδικασία τελειώνει σε αυτό το σημείο και οι συσκευές δεν αναβαθμίζονται. Αν αποφασίσουν να την υποστηρίξουν, αναπτύσσουν τον νέο οδηγό και τον βελτιστοποιούν για την νέα έκδοση.
8. Οι κατασκευαστές συσκευών λαμβάνουν τους νέους οδηγούς από τους προμηθευτές τους των ολοκληρωμένων κυκλωμάτων.
9. Οι κατασκευαστές συσκευών αποφασίζουν αν θα υποστηρίξουν την νέα έκδοση Android. Αν αποφασίσουν να μην την υποστηρίξουν, η διαδικασία τελειώνει σε αυτό το σημείο και οι συσκευές δεν αναβαθμίζονται. Αν αποφασίζουν να την υποστηρίξουν, παρέχουν τους απαραίτητους πόρους για αυτήν την διαδικασία.
10. Οι κατασκευαστές συσκευών ενσωματώνουν τις τροποποιήσεις τους στην νέα έκδοση (Samsung TouchWiz, HTC Sence, κτλ).
11. Οι κατασκευαστές συσκευών συνεργάζονται με τον κάθε carrier για τις προεγκατεστημένες εφαρμογές, υπηρεσίες κτλ.

[2] HTC Software Update proccess http://www.htc.com/us/go/htc-software-updates-process/

12. Οι κατασκευαστές συσκευών προχωρούν σε εσωτερικές δοκιμές και διορθώσεις.
13. Γίνεται η αποδοχή της νέας έκδοσης από τους carriers και την Google.
14. Η αναβάθμιση δημοσιεύεται και παρέχεται στους χρήστης μέσω OTA (Over-the-air).

Ο κατακερματισμός στις εκδόσεις της πλατφόρμας Android είναι ιδιαίτερα εκτεταμένος. Ακολουθεί σχετικός πίνακας και γράφημα με τις «ζωντανές» εκδόσεις. Τα στοιχεία αφορούν συσκευές που συνδέονται στο Google Play Store για το διάστημα 7 ημερών που τελειώνει την 1 Αυγούστου 2016 και παρέχονται από την Google. Επίσης, ποσοστά μικρότερα του 0.1% δεν εμφανίζοντα.



Έκδοση	**Έκδοση API**	**Όνομα**	**Μερίδιο %**	**Ημ. Κυκλοφορίας**
2.2	8	Froyo	0.1%	20 Μαΐου 2010
2.3.3 - 2.3.7	10	Gingerbread	1.7%	6 Δεκεμβρίου 2010
4.0.3 -	15	Ice Cream	1.6%	18 Οκτωβρίου 2011

4.0.4		Sandwich		
4.1.x	16	Jelly Bean	6.0%	9 Ιουλίου 2012
4.2.x	17		8.3%	
4.3	18		2.4%	
4.4	19	KitKat	29.2%	31 Οκτωβρίου 2013
5.0	21	Lollipop	14,1%	12 Νοεμβρίου 2014
5.1	22		21,4%	
6.0	23	Marshmallow	15.2%	5 Οκτωβρίου 2015
7.0	24	Nougat	-	22 Αυγούστου 2015

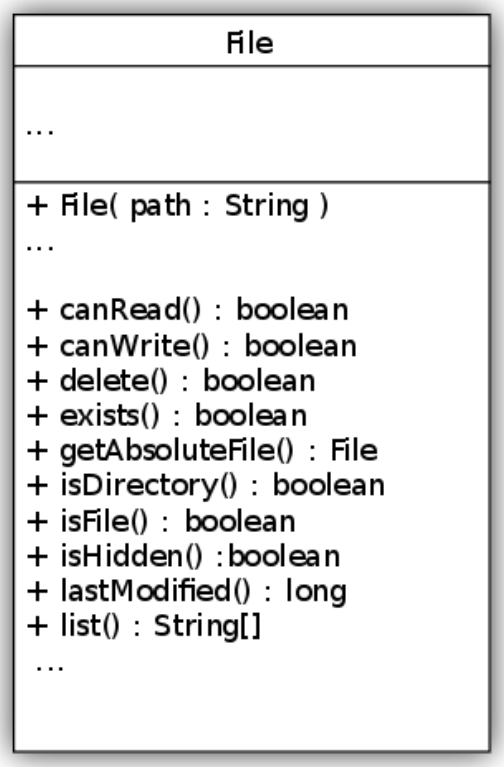
3.1.3. Fragmentation - Μελέτη περίπτωσης: Σύστημα αρχείων

Σε πολλές εφαρμογές τυγχάνει να απαιτείται η πρόσβαση στο σύστημα αρχείων της συσκευής είτε για λειτουργίες διαχείρισης αρχείων, είτε για απλή αποθήκευση οποιονδήποτε αρχείων (στιγμιότυπα οθόνης, άλλες εικόνες, φωτογραφίες/βίντεο, αρχεία ήχου κτλ) είτε για απλή ανάγνωση αρχείων. Ο κατακερματισμός σε αυτό το επίπεδο δημιουργεί αρκετές δυσκολίες στην ανάπτυξη εφαρμογών όχι τόσο γιατί η απαιτούμενη λειτουργία χρειάζεται από πολλές εφαρμογές αλλά λόγο της μεγάλης διαφοράς στη Διεπαφή Προγραμματισμού Εφαρμογών.

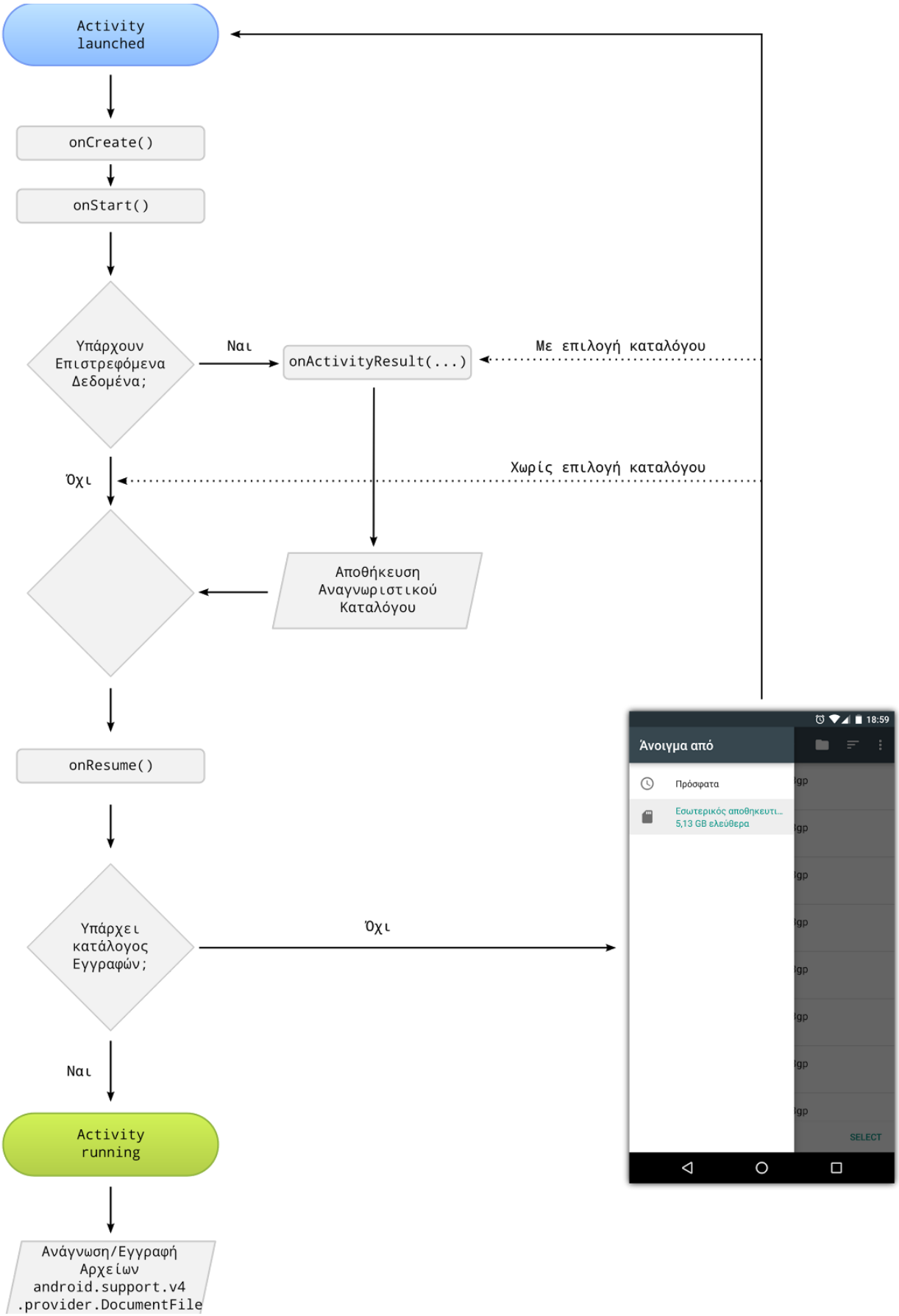
Αν η ανάγκη μιας εφαρμογής απαιτεί την πρόσβαση στο σύστημα αρχείων, τότε, κατά πάσα πιθανότητα, η ομάδα ανάπτυξης θα βρεθεί μπροστά στον κατακερματισμό. Υπάρχουν δύο εκδόσεις της Διεπαφής Προγραμματισμού Εφαρμογών για την πρόσβαση στο σύστημα αρχείων βάση της έκδοσης της πλατφόρμας Android που θα υποστηριχθεί από την εφαρμογή. Η μία έκδοση του API, καλύπτει τις εκδόσεις της πλατφόρμας μεγαλύτερες από την 5.0 (Lollipop). Σύμφωνα με τα στατιστικά στοιχεία της προηγούμενης ενότητα, αυτές οι εκδόσεις καλύπτουν το 50,7% των χρηστών. Κατά πάσα πιθανότητα, δεν θα είναι αρκετό η εφαρμογή να είναι διαθέσιμη μόνο στους μισούς χρήστες και κατ’ επέκταση θα χρειαστεί να υλοποιηθούν και οι δύο εκδόσεις του API.

Όπως φαίνεται στις ακόλουθες δύο εικόνες, η πρώτη περίπτωση είναι αρκετά απλοϊκή στην υλοποίηση και ίσως δεν έχει ληφθεί σοβαρά υπόψιν. Πάραυτα, επειδή η δεύτερη περίπτωση είναι αρκετά διαφορετική στην υλοποίηση, τα πράγματα περιπλέκονται αρκετά στο ενδιάμεσο, πολλαπλών εκδόσεων API, που θα χρειαστεί να υλοποιηθεί, αφού θα πρέπει να συνδυαστούν και τις δύο μέθοδοι.

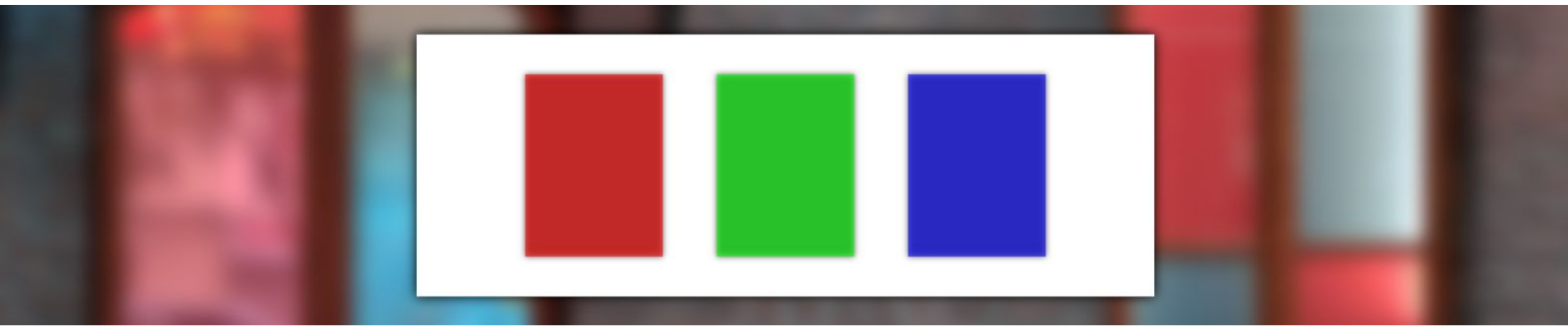
Όταν η εφαρμογή αναπτύσσεται με στόχο τις πρό-Lollipop εκδόσεις, τότε, αρκεί η αξιοποίηση της απλής κλάσης java.io.File τόσο για την διαχείριση των αρχεία όσο και για την διαχείριση των καταλόγων.



Όταν η εφαρμογή αναπτύσσεται με στόχο τις μετά-Lollipop εκδόσεις, τότε τα πράγματα είναι διαφορετικά:


Activity launched
onCreate()
onStart()
Υπάρχουν Επιστρεφόμενα Δεδομένα;
Ναι
onActivityResult(...)
Με επιλογή καταλόγου
Όχι
Χωρίς επιλογή καταλόγου
Αποθήκευση Αναγνωριστικού Καταλόγου
onResume()
Υπάρχει κατάλογος Εγγραφών;
Όχι
Ναι
Activity running
Ανάγνωση/Εγγραφή Αρχείων android.support.v4 .provider.DocumentFile
18:59
Άνοιγμα από
Πρόσφατα
Εσωτερικός αποθηκευτι...
5,13 GB ελεύθερα
SELECT

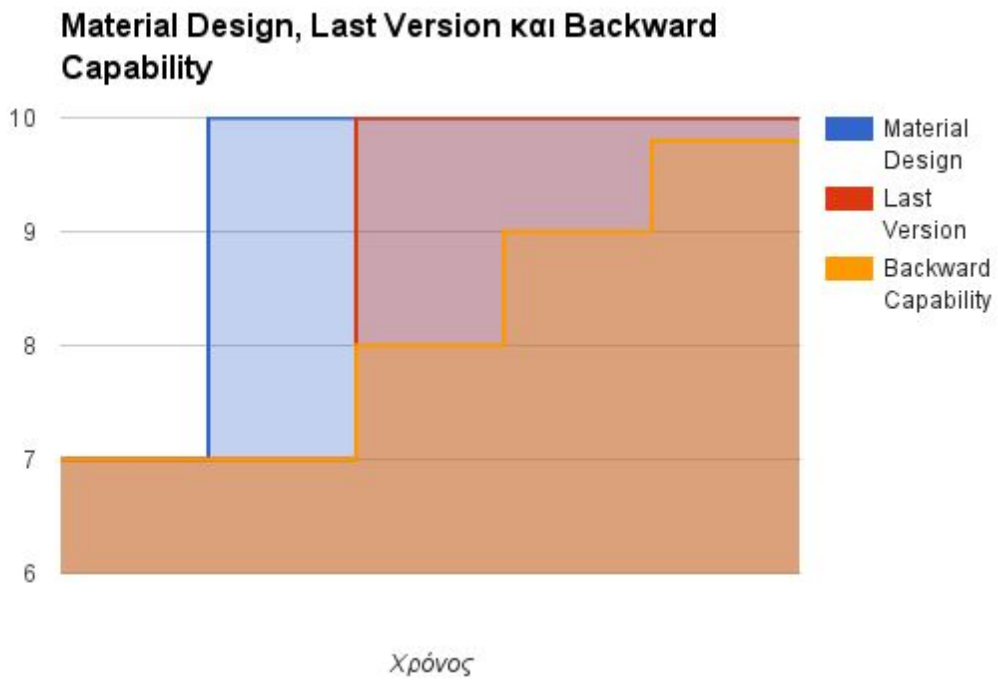
3.1.4. Fragmentation - Μελέτη περίπτωσης: DialogFragment & Elevation



Το 2014 η Google ανακοίνωσε το Material Design. Πρόκειται για ένα σύνολο σχεδιαστικών κανόνων βασισμένο «στο χαρτί και το μολύβι». Έκτοτε, το εγχείρημα, αναπτύσσεται και διαδίδεται με το πέρασμα του χρόνου. Βασίζεται σε κάρτες, σε διάταξη πίνακα, σε κινούμενα σχέδια, σε περιθώρια, στο βάθος και τον φωτισμό. Αυτοί οι σχεδιαστικοί κανόνες έχουν ακολουθηθεί στην πλατφόρμα Android τόσο στις νέες εκδόσεις όσο και στις παλαιότερες μέσω βιβλιοθηκών με backward capability (προς τα πίσω υποστήριξη). Όμως, αυτοί οι σχεδιαστικοί κανόνες, δεν έμειναν στην πλατφόρμα Android αλλά πέρασαν και στον Ιστό. Η χρήση τους πλέον έχει υιοθετηθεί και σε έργα τρίτων πέρα από αυτά της Google.

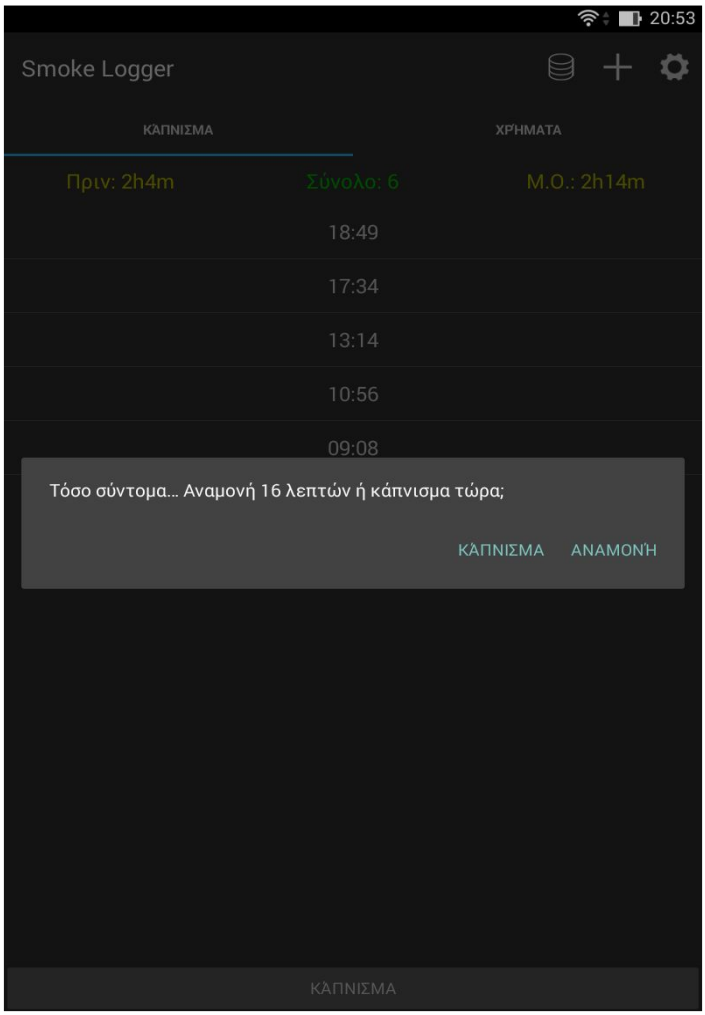
Επειδή το Material Design είναι αναπτυσσόμενο στον χρόνο και όχι κάτι σταθερό, πολλές φορές οι βιβλιοθήκες της πλατφόρμας Android τυγχάνει κατά διαστήματα να μην τους υποστηρίζουν πλήρως και μάλιστα να έχουν διαφορές στις διάφορες εκδόσεις της πλατφόρμας. Αυτό που συμβαίνει είναι το εξής: Αρχικά υπάρχει το Materia Design με κάποιους κανόνες. Σε κάποιο χρονικό σημείο, προσθέτονται κάποιοι νέοι κανόνες. Αργότερα, αυτοί οι νέοι κανόνες υποστηρίζονται στην νέα έκδοση της πλατφόρμας Android. Ταυτόχρονα, κάποιοι από αυτούς τους νέους κανόνες, όχι όλοι, προσθέτονται στην backward capability βιβλιοθήκη για τις συσκευές με παλαιότερες εκδόσεις της πλατφόρμας. Αργότερα, υποστηρίζονται

κάποιοι ακόμη κανόνες στην backward capability βιβλιοθήκη από αυτούς που δεν είχαν υποστηριχθεί αρχικά.



Το καλοκαίρι του 2015, κατά τη ανάπτυξη της εφαρμογής Smoke Logger[3], έγινε χρήση της backward capability βιβλιοθήκης ώστε να ακολουθηθούν οι νέοι κανόνες σχεδίασης ακόμη και στις παλαιότερες εκδώσεις της πλατφόρμας. Όμως, τα DialogFragmet (παράθυρα διαλόγου) δεν είχαν υποστηριχθεί. Για να ξεπεραστεί αυτή η έλλειψη, σπαταλήθηκε αρκετός χρόνος (σχετικά με την «αξία» του προβλήματος) σε έρευνα και δοκιμές και τελικά έγινε ταυτόχρονη χρήση βιβλιοθήκης τρίτου προγραμματιστή ειδικά για τα παράθυρα διαλόγου. Αρκετά αργότερα, τα DialogFragmet υποστηρίχθηκαν τελικά από την βιβλιοθήκη v7 appcompat (backward capability) και η χρήση της βιβλιοθήκης του τρίτου προγραμματιστή δεν είναι πλέων αναγκαία.

[3] https://play.google.com/store/apps/details?id=com.grapsas.smokelogger



* Στιγμιότυπο από την εφαρμογή Smoke Logger. Χρήση της βιβλιοθήκης v7 appcompat για υποστήριξη του Material Design και ταυτόχρονη χρήση βιβλιοθήκης τρίτου προγραμματιστή για την υποστήριξη του Material Design στα παράθυρα διαλόγου.

Σχεδόν ένα χρόνο μετά, κατά την ανάπτυξη της εφαρμογής Stream Recorder για την παρούσα πτυχιακή εργασία, στο Material Desing έγινε εκτεταμένη χρήση των σκιών και επιπέδων (υψώματα) μέσω της ιδιότητα Elevation. Αυτοί οι κανόνες υποστηρίχθηκαν από τις εκδόσεις 5.0 (Lollipop, APIv21) και πλέων της πλατφόρμας Android αλλά όχι και από την backward capability βιβλιοθήκη για τις παλαιότερες εκδώσει. Αυτή η έλλειψη των σκιών γίνεται ιδιαίτερα κουραστική στις παλαιότερες εκδώσεις σε συνδυασμό με το προεπιλεγμένο χρωματικό θέμα και επηρεάζει τους μισούς χρήστες (49,3%).

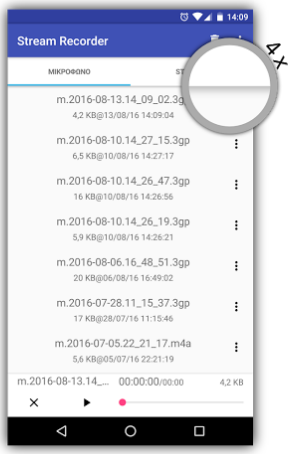
Στο παρακάτω γραφικό παρουσιάζεται το πρόβλημα της έλλειψης σκιών. Υπάρχουν στιγμιότυπα από τρεις διαφορετικές συσκευές όπου καθεμιά έχει από μια

διαφορετική έκδοση της πλατφόρμας Android. Επίσης, τα πρώτα δύο είναι από κινητά τηλέφωνα και η τρίτη από ταμπλέτα. Η αναλογία της απεικόνισης είναι 1:2 και έχει μεγενθυθεί 4 φορές η περιοχή της σκιάς.

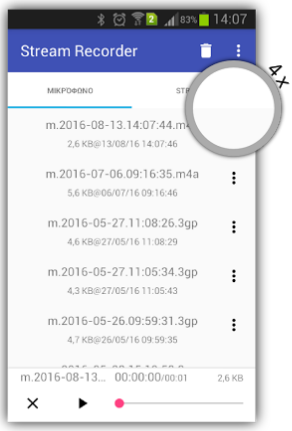
Το πρώτο στιγμιότυπο αφορά στην συσκευή LG Nexus 5x η οποία αναβαθμίζεται σχεδόν αμέσως μετά την ανακοίνωση των νέων εκδόσεων της πλατφόρμας. Είναι εγκατεστημένη η τελευταία έκδοση (στον χρόνο συγγραφής της πτυχιακής) της πλατφόρμας Android 6.0.1 / APIv23 (Marshmallow). Το φυσικό μέγεθος της οθόνης είναι 5,2 inch και η φυσική της ανάλυση Full HD (1080 χ 1920 px). Η ιδιότητα Elevation υποστηρίζεται και η εμφανιζόμενη πίπτουσα σκιά της μπάρας καρτελών την ξεχωρίζει από την κύρια περιοχή των ηχητικών εγγραφών με βάση το προεπιλεγμένο χρωματικό θέμα.

Το δεύτερο στιγμιότυπο αφορά στην συσκευή Samsung Galaxy Grand Duos η οποία δεν αναβαθμίστηκε ποτέ και ούτε πρόκειται πλέων αφού είναι συσκευή του έτους 2012. Είναι εγκατεστημένη η έκδοση της πλατφόρμας Android 4.1.2 / APIv16 (Jelly Bean). Το φυσικό μέγεθος της οθόνης είναι 5,0 inch και η φυσική της ανάλυση 480 χ 800 px. Η ιδιότητα Elevation δεν υποστηρίζεται και η μπάρα καρτελών δεν ξεχωρίζει εύκολα από την κύρια περιοχή των ηχητικών εγγραφών με βάση το προεπιλεγμένο χρωματικό θέμα αφού και τα δύο είναι χρωματικά κοντά. Στην μπάρα καρτελών χρησιμοποιείτε το χρώμα με κωδικό #FFFFFF (πλήρως λευκό) και στην κύρια περιοχή το χρώμα με κωδικό #FAFAFA (σχεδόν λευκό).

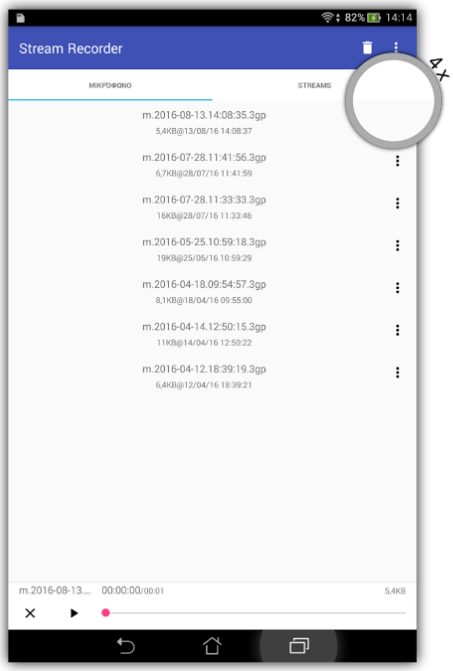
Το τρίτο στιγμιότυπο αφορά την συσκευή Asus Memo Pad 8 η οποία δεν αναβαθμίστηκε ποτέ και ούτε πρόκειται πλέων αφού είναι συσκευή του έτους 2014. Είναι εγκατεστημένη η έκδοση της πλατφόρμας Android 4.4.2 / APIv19 (KitKat). Το φυσικό μέγεθος της οθόνης είναι 8,0 inch και η φυσική της ανάλυση 800 χ 1280 px. Η ιδιότητα Elevation δεν υποστηρίζεται και η μπάρα καρτελών δεν ξεχωρίζει εύκολα από την κύρια περιοχή των ηχητικών εγγραφών με βάση το προεπιλεγμένο χρωματικό θέμα αφού και τα δύο είναι χρωματικά κοντά. Στην μπάρα καρτελών χρησιμοποιείτε το χρώμα με κωδικό #FFFFFF (πλήρως λευκό) και στην κύρια περιοχή το χρώμα με κωδικό #FAFAFA (σχεδόν λευκό).



Μοντέλο: LG Nexus 5x
Έκδοση Android: 6.0.1
Έκδοση API: 23
Μέγεθος οθόνης: 5.2'
Ανάλυση: 1920 χ 1080 px



Μοντέλο: Samsung Galaxy Grand Duos
Έκδοση Android: 4.1.2
Έκδοση API: 16
Μέγεθος οθόνης: 5'
Ανάλυση: 480 χ 800 px



Μοντέλο: Asus Memo Pad 8
Έκδοση Android: 4.4.2
Έκδοση API: 19
Μέγεθος οθόνης: 8'
Ανάλυση: 800 χ 1280 px

Αναλογία απεικόνισης 1:2

Υπάρχουν τεχνικές για να καλυφθεί η έλλειψη των σκιών αλλά απαιτείται έρευνα, δοκιμές και χρόνος για την εύρεση και την υλοποίηση κάποιας λύσης. Φυσικά υπάρχει η δυνατότητα της αλλαγής των χρωμάτων, αλλά αυτό θα έδινε την αίσθηση μιας πολύχρωμης εφαρμογής. Ταυτόχρονα, υπάρχει η πιθανότητα στο μέλλον να αναβαθμιστεί η backward capability βιβλιοθήκη και να υποστηριχθούν οι σκιές.

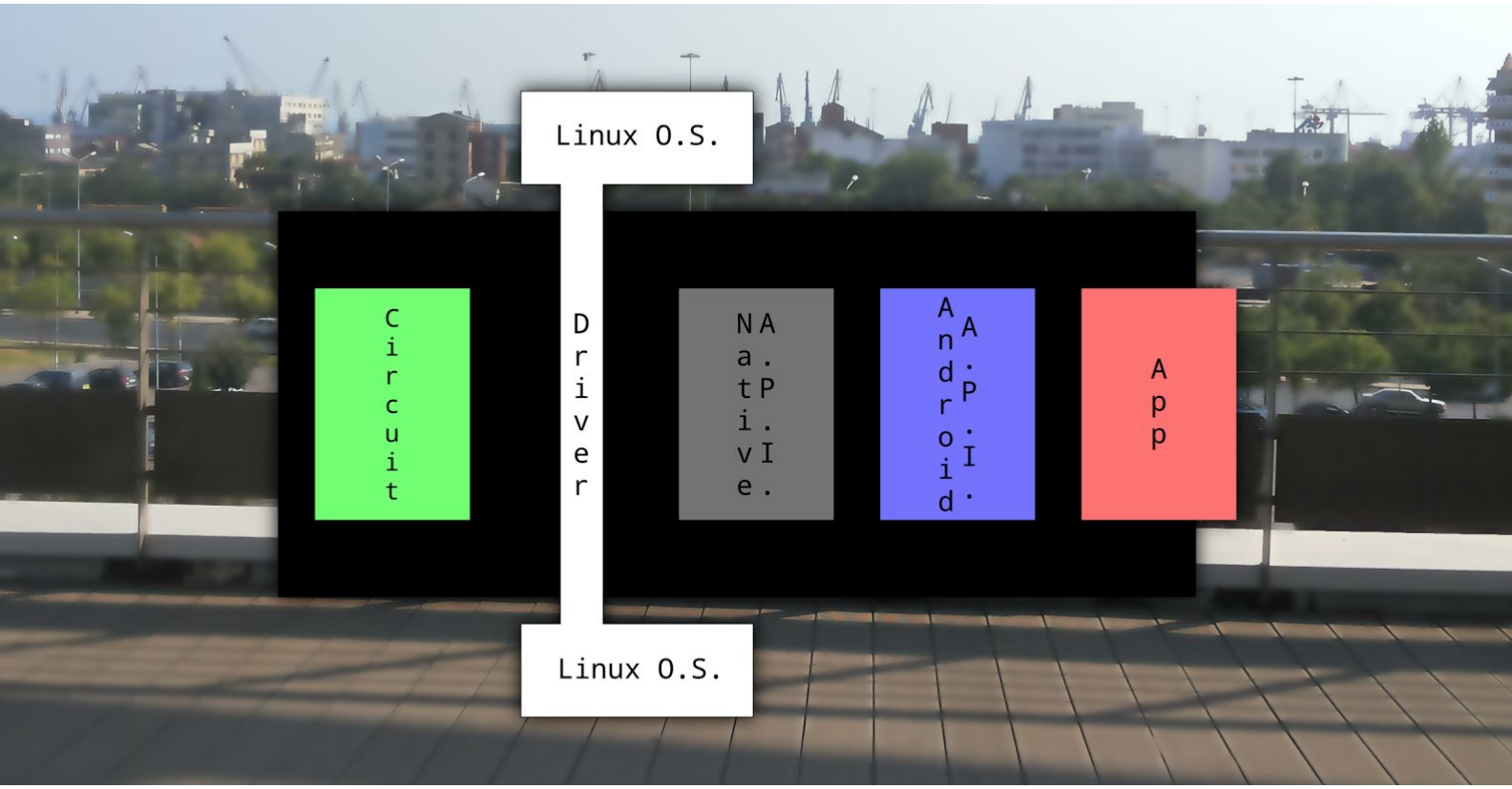
Εν γένει υπάρχει αρκετός κατακερματισμός στο πεδίο των εκδόσεων της πλατφόρμας που απαιτεί αρκετό χρόνο για την επίλυση όλων των σημείων. Αυτό το κόστος είναι βιώσιμο μόνο στην περίπτωση που υπάρχει η αντίστοιχη παραγωγικότητα στην ανάπτυξη εφαρμογών. Για κάποιον που αναπτύσσει μία εφαρμογή για παράδειγμα ανά έτος και σε συνδυασμό με την ανάπτυξη και τις

αλλαγές που εμφανίζονται σε αυτό το διάστημα, είναι δύσκολο να τελειοποιήσει όλα τα σημεία του κατακερματισμού.

3.2. Εγγραφή από δέκτη ραδιοφώνου

Η εγγραφή από ραδιοφωνικό δέκτη δεν έγινε δυνατό να υλοποιηθεί στα πλαίσια την παρούσας πτυχιακής εργασίας λόγο τεχνικών ελλείψεων από πλευράς λειτουργικού συστήματος και πλατφόρμας. Πρόκειται για ένα από τα λίγα αχαρτογράφητα κομμάτια τόσο σε επίπεδο τεκμηρίωσης όσο και σε επίπεδο προτύπων. Όσο αφορά το σύστημα Android, δεν περιλαμβάνει υποστήριξη από πλευράς βιβλιοθηκών και τεκμηρίωσης και ως εκ τούτου, στο επίπεδο της ανάπτυξης εφαρμογών δεν μπορεί να υποστηριχθεί.

Βέβαια, κυκλοφορούν πολλές συσκευές με Android από διάφορους γνωστούς κατασκευαστές, οι οποίες περιλαμβάνουν ραδιοφωνικό δέκτη. Αυτό σημαίνει πως υπάρχει ειδικό κύκλωμα ραδιοφωνικού δέκτη, έχει αναπτυχθεί οδηγός για το λειτουργικό σύστημα Linux καθώς και κάποιου είδους API (Application Programming Interface) ώστε να γίνεται δυνατό στον κατασκευαστή η δημιουργία μιας εφαρμογής η οποία να εκμεταλλεύεται το ραδιόφωνο.



Η αξιοποίησή του όμως τελειώνει εκεί. Δεν δημοσιεύεται κάποια τεκμηρίωση, κάποιο API, ή ο κώδικας του οδηγού του λειτουργικού συστήματος Linux. Οι διανομές του AOSP (Android Open Source Project) που αναπτύσσουν πολλές ομάδες προγραμματιστών με σκοπό την αντικατάσταση της προεγκατεστημένης έκδοσης Android των συσκευών, δεν παρέχουν υποστήριξη για τον ραδιοφωνικό δέκτη.

Στο ηλεκτρονικό κατάστημα/δίκτυο διανομής εφαρμογών Google Play Store, αν και υπάρχουν πολλές εφαρμογές για καθεμιά εργασία, για το συγκεκριμένο βρέθηκε μόνο μία. Ο δημιουργός την πωλεί 7,91€, σχετικά ακριβά συγκριτικά με άλλες μεγάλου ή μικρού κόστους παραγωγής εφαρμογές και αναφέρει:

«I understand the price may seem high, but this is a niche app with 10,000 hours of work required over 3+ years. How many other "Real FM Radio" apps are on Google Play ? Almost none.

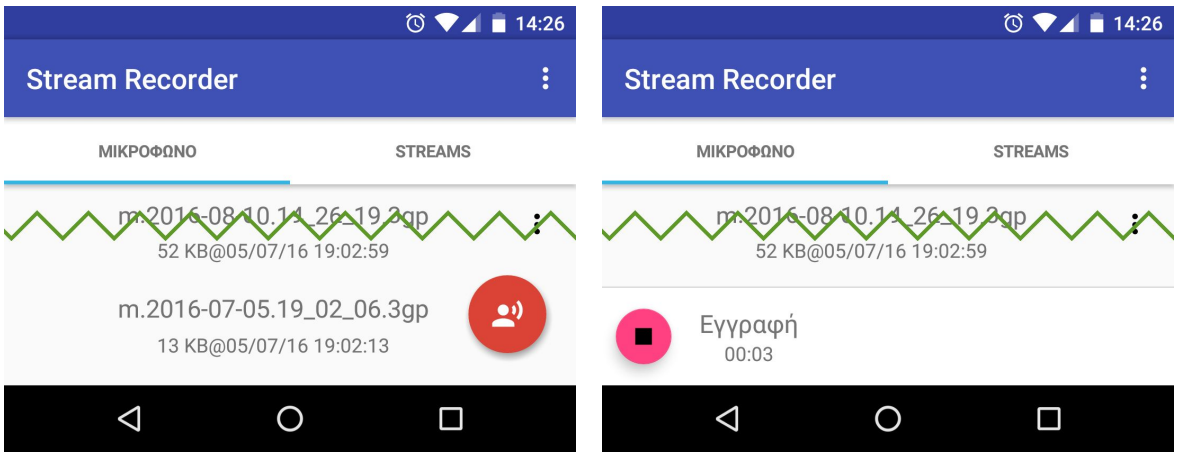
Why ? Because Android FM is extremely difficult. Google doesn't support FM radio and carriers suppress it. Intensive reverse engineering is required. In the over 3 years since Spirit was first created, nobody else has been "crazy" enough to attempt to compete in this niche (OTA radio) within a niche (AOSP ROMs) or without (all phones with usable FM).»[4]

Για τους παραπάνω λόγους, η ανάπτυξη εφαρμογής που εκμεταλλεύεται τον ραδιοφωνικό δέκτη στα πλαίσια αυτής της πρακτικής εργασίας καθίσταται πρακτικώς αδύνατη. Ίσως, η ενασχόληση ειδικά με τον ραδιοφωνικό δέκτη μιας συγκεκριμένης συσκευής ενός κατασκευαστή και η παραγωγή τεχνικών και πρακτικών που θα μπορούσαν να αξιοποιηθούν για την υποστήριξη του ραδιοφώνου σε άλλη συσκευή από άλλον ερευνητή, ίσως θα μπορούσε να αποτελέσει από μόνο του θέμα πτυχιακής εργασίας.

[4] https://play.google.com/store/apps/details?id=com.mikersmicros.fm_unlock

3.3. Εγγραφή από μικρόφωνο

Όλες οι συσκευές κινητών τηλεφώνων και ταμπλετών ενσωματώνουν μικρόφωνο. Η πλατφόρμα Android παρέχει τα απαραίτητα API (Application Programming Interface) και την απαραίτητη τεκμηρίωση ώστε να είναι εκμεταλλεύσιμο από τις εφαρμογές. Η **Stream Recorder** αξιοποιεί το μικρόφωνο ως πηγή ήχου και αποθηκεύει τα δεδομένα στο σύστημα αρχείων της συσκευής. Υποστηρίζει διάφορες μορφές ήχου και αρχείων τις οποίες μπορεί να επιλέξει ο χρήστης μέσω ειδικής Activity («παράθυρο» στην πλατφόρμα Android) ρυθμίσεων (περισσότερες λεπτομέρειες στο επόμενο κεφάλαιο).

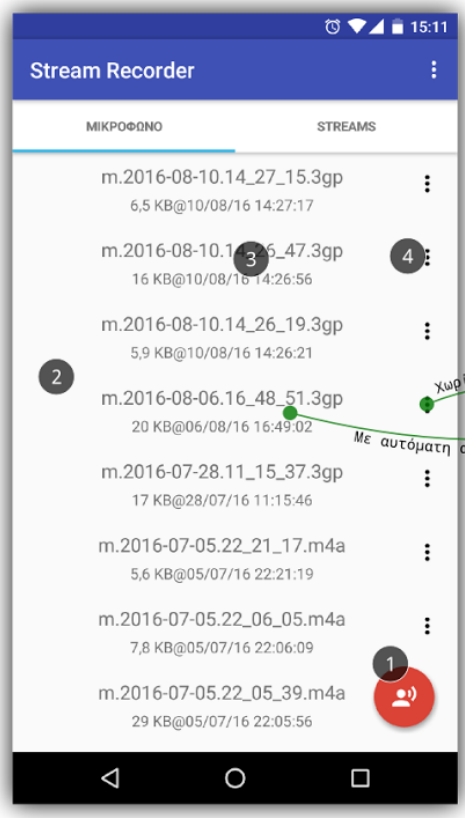


Η **εγγραφή από μικρόφωνο** είναι η ποιο εύκολη και γρήγορη ενέργεια για τον χρήστη της εφαρμογής. Στην πρώτη καρτέλα και στην πρώτη Activity, ο χρήστης αρκεί να ακουμπήσει το σχετικό FAB (FloatingActionButton) ώστε να ξεκινήσει η εγγραφή. Σε αυτό το χρονικό σημείο, το FAB εγγραφής εξαφανίζεται και στην κάτω πλευρά της Activity εμφανίζεται σχετικό widget. Σε αυτό υπάρχει FAB για την διακοπή της εγγραφής, ένδειξη ότι πραγματοποιείται ηχογράφηση καθώς και χρονόμετρο για την διάρκεια της εγγραφής. Ο χρήστης αργότερα μπορεί να ακουμπήσει το FAB διακοπής ώστε να ολοκληρωθεί η εγγραφή.

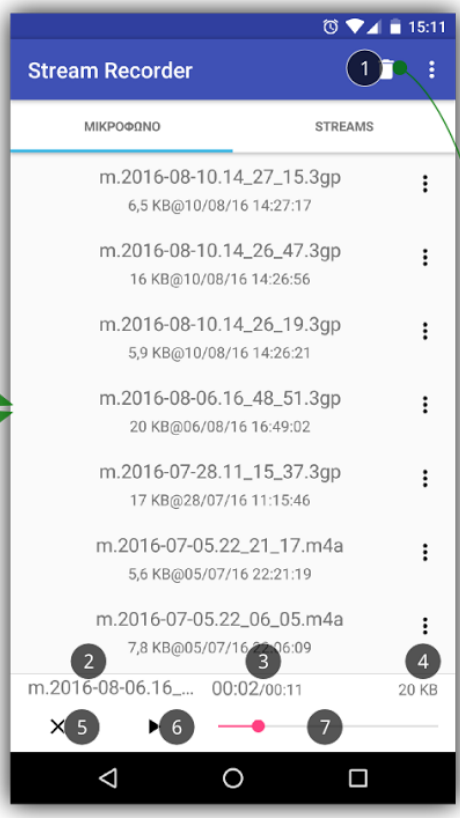
Μετά το τέλος της εγγραφής, ο χρήστης μπορείς να **αναπαράγει το ηχητικό αρχείο** από την ίδια την εφαρμογή. Για να γίνει αυτό, αρκεί να το αγγίξει από την λίστα των εγγραφών. Την αναπαραγωγή αναλαμβάνει ο ενσωματωμένος αναπαραγωγέας της εφαρμογής όπου αναπτύχθηκε για αυτόν τον σκοπό. Ο αναπαραγωγέας δύναται να αναπαράγει όλες τις μορφές ήχου που υποστηρίζει η εφαρμογή για ηχογράφηση.

Ταυτόχρονα, τα αρχεία ηχογράφησης είναι διαθέσιμα και στο **σύστημα αρχείων** της συσκευής. Αυτό σημαίνει πως ο χρήστης δύναται να το αναπαράγει από οποιοδήποτε άλλη εφαρμογή επιθυμεί. Επίσης, υπάρχει η δυνατότητα να μεταφερθούν τα αρχεία ηχογράφησης και σε άλλη συσκευή ή Π/Υ μέσω των εργαλείων που παρέχει η πλατφόρμα Android, όπως Bluetooth ή e-mail ή μεταφόρτωση στο Νέφος ή μεταφορά με χρήση καλωδίου USB.

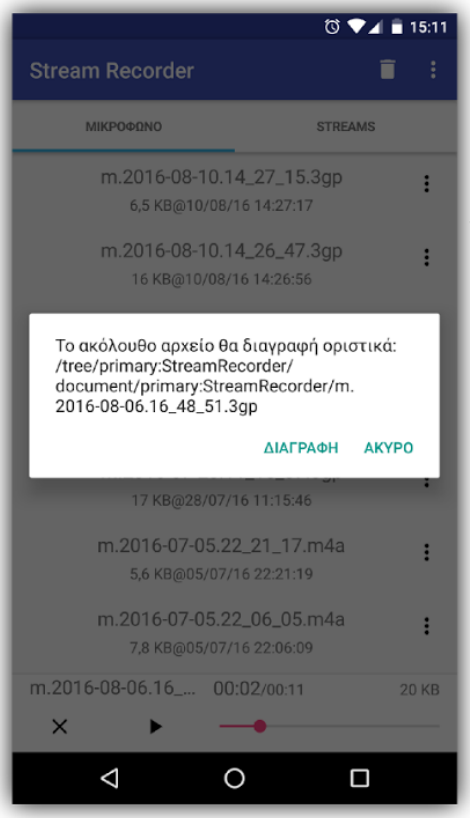
Εικόνα 1



Εικόνα 2



Εικόνα 3



Εικόνα 1:

1. FAB για την έναρξη εγγραφής.
2. Λίστα αρχείων ηχογράφησης.
3. Μεμονομένο αρχείο ηχογράφησης. Μετά το άγγιγμά του, εμφανίζεται ο ενσωματωμένος αναπαραγωγέας (εικ. 2) και ξεκινάει αυτόματα η αναπαραγωγή του ηχητικού αρχείου.
4. Ιδιότητες μεμονομένου αρχείου ηχογράφησης. Μετά το άγγιγμά του, εμφανίζεται ο ενσωματωμένος αναπαραγωγέας (εικ. 2) αλλά δεν ξεκινάει αυτόματα η αναπαραγωγή του ηχητικού αρχείου. Η αναπαραγωγή μπορεί να ξεκινήσει με το σχετικό κουμπί έναρξης.

Στην **Εικόνα 2** μεταφέρεται ο χρήστης είτε από το την επιλογή 3 είτε από την επιλογή 4 της Εικόνας 1 και υπάρχουν τα παρακάτω χαρακτηριστικά:

1. Διαγραφή ηχητικού. Εμφανίζεται στον χρήστη μήνυμα επιβεβαίωσης μέσω DialogFragment. Αν ο χρήστης επιβεβαιώσει αυτήν την ενέργεια, το αρχείο απομακρύνεται οριστικά από το σύστημα αρχείων και από την λίστα ηχογραφήσεων της εφαρμογής.
2. Όνομα αρχείου που έχει επιλεγεί είτε για αναπαραγωγή είτε για διαγγραφή.
3. Χρονικό σημείο αναπαραγωγής και συνολική διάρκεια αρχείου ηχογράφησης.
4. Μέγεθος αρχείου ηχογράφησης.
5. Αποεπιλογή αρχείου και κλείσιμο ενσωματωμένου αναπαραγωγέα (επίσης διακόπτεται τυχόν αναπαραγωγής σε εξέλιξη).
6. Έναρξη/Παύση αρχείου ηχογράφησης.
7. Χρονικό σημείο αναπαραγωγής. Γραφική αναπαράσταση του χρονικού σημείου αναπαραγωγής και συνολικής διάρκειας σε ευθύγραμμο τμήμα. Αμφίδρομη ανατροφοδότηση τόσο από τον απαραγωγέα όσο και από τον χρήστη.

3.4. Ρυθμίσεις Εγγραφής από Μικρόφωνο

Η πλατφόρμα Android παρέχει κατάλληλη Διεπαφή Προγραμματισμού Εφαρμογών (API) για την εκμετάλλευση του μικρόφωνου και για την εγγραφή του ήχου που αυτό λαμβάνει. Η εφαρμογή Stream Recorder αξιοποιεί την παραπάνω Διεπαφή για την παροχή της δυνατότητας της ηχογράφησης από το μικρόφωνο και για την αποθήκευση του ήχου στο σύστημα αρχείων όπως περιγράφεται στην προηγούμενη ενότητα.

Η εγγραφή του ήχου πραγματοποιείται με την βοήθεια της κλάσης MediaRecorder (android.media.MediaRecorder) στην οποία δηλώνεται η χρήση του μικροφώνου ως πηγής εισόδου και ως έξοδος μια διαδρομή αρχείου ή ένα FileDescriptor. Επίσης, δηλώνονται και τα υπόλοιπα επιθυμητά στοιχεία για το εξαγόμενο αρχείο όπως το container και την κωδικοποίηση.

Στην εφαρμογή StreamRecorder ως έξοδος στο MediaRecorder δηλώνεται ένα FileDescriptor αντί διαδρομής αρχείου. Αυτό επιλέγει λόγο της διαφοράς της Διεπαφής Προγραμματισμού Εφαρμογών ανάμεσα στις εκδόσεις της πλατφόρμας. Στις εκδώσεις του API level >= 21 (Lollipop), η πρόσβαση και διαχείριση των αρχείων γίνεται μέσω της κλάσης DocumentFile η οποία δεν παρέχει πληροφορίες για την διαδρομή (path) των αρχείων, όμως παρέχει το FileDescriptor του αρχείου. Στις εκδώσεις τους API level < 21, η κλασσική κλάση File (java.io.File) παρέχει και FileDescriptor μέσω της κλάσης RandomAccessFile (java.io.RandomAccessFile) εκτός της διαδρομής του αρχείου.

Η κλάση FileDescriptor (java.io.FileDescriptor) αντιπροσωπεύει το File Descriptor του λειτουργικού συστήματος Linux. Ουσιαστικά πρόκειται για έναν δείκτη αρχείου, του λειτουργικού, διαφορετικού του τύπου FILE της βιβλιοθήκης της C, που αντιπροσωπεύεται από έναν θετικό αριθμό και που οι αρνητικοί δηλώνουν κάποιου είδους σφάλμα. Εσωτερικά της βιβλιοθήκης της C, για αυτόν τον δείκτη χρησιμοποιείται ο τύπος int[5].

Όσο αφορά τις μορφές κωδικοποίησης ήχου, η **πλατφόρμα** Android υποστηρίζει για ηχογράφηση τις παρακάτω[6]:

Σταθερά	Κωδικοποίηση	Έκδοση API που προστέθηκε υποστήριξη
AAC	AAC Low Complexity (AAC-LC) audio codec	10
AAC_ELD	Enhanced Low Delay AAC (AAC-ELD) audio codec	16

[5] Linux Programmer's Manual - READ(2): [...] ssize_t read(**int fd**, void *buf, size_t count); [...].
[6] https://developer.android.com/reference/android/media/MediaRecorder.OutputFormat.html

AMR_NB	AMR (Narrowband) audio codec	1
AMR_WB	AMR (Wideband) audio codec	10
HE_AAC	High Efficiency AAC (HE-AAC) audio codec	16
VORBIS	Ogg Vorbis audio codec	21

Όσο αφορά τις μορφές αρχείων ήχου, η **πλατφόρμα** Android υποστηρίζει για ηχογράφηση τις παρακάτω[7]:

Σταθερά	Κωδικοποίηση	Έκδοση API που προστέθηκε υποστήριξη
AAC_ADTS	AAC ADTS file format	16
AMR_NB	AMR NB file format	10
AMR_WB	AMR WB file format	10
MPEG_4	MPEG4 media file format	1
RAW_AMR	AMR NB file format	3 & >= 16[8]
THREE_GPP	3GPP media file format	1
WEBM	VP8/VORBIS data in a WEBM container	21

[7] https://developer.android.com/reference/android/media/MediaRecorder.OutputFormat.html

[8] https://developer.android.com/reference/android/media/MediaRecorder.OutputFormat.html#RAW_AMR

Η εφαρμογή **StreamRecorder** έχει υποστηρίξει τους παρακάτω συνδυασμούς για ηχογράφηση από μικρόφωνο:

Κωδικοποίηση	Μορφή αρχείου
AAC-LC (Low Complexity)	● 3GPP (.3gp) ● MPEG-4 (.m4a)
HE-AAC (High Efficiency)	● 3GPP (.3gp) ● MPEG-4 (.m4a)
AAC-ELD (Enhanced Low Delay)	● 3GPP (.3gp) ● MPEG-4 (.m4a)
AMR-NB (Narrowband)	● 3GPP (.3gp)
AMR-WB (Wideband)	● 3GPP (.3gp)

Στο γραφικό που ακολουθεί παρουσιάζονται τα στιγμιότυπα από τις ρυθμίσεις της εφαρμογής. Η πρώτη επιλογή αφορά την μορφή της κωδικοποίησης ενώ το δεύτερο στιγμιότυπο αφορά την μορφή του αρχείου. Οι δύο αυτές επιλογές είναι συνδεδεμένες μεταξύ του, αφού όλες οι μορφές κωδικοποιήσης δεν είναι συμβατές με όλες τις μορφές των αρχείων. Για παράδειγμα, η κωδικοποίηση AMR-NB (Narrowband), δεν είναι συμβατή με το container MPEG-4 και κατάληξη αρχείου .m4a.

Τέλος, αξίζει να αναφερθεί ότι η παραπάνω ρυθμίσεις δεν είναι υλοποιημένες (hard-coded) στον κώδικα αλλά δημιουργούνται δυναμικά με την βοήθεια ενός JSON parser. Με αυτόν τον τρόπο, είναι δυνατή η επέκτασή τους ή τροποποίησής του με εύκολο τρόπο. Στον κώδικα του έργου υπάρχει ένα assets αρχείο με όνομα audioTypes.json το οποίο περιέχει όλες τις απαραίτητες πληροφορίες. Ακολουθεί μέρος του περιεχομένου του και ύστερα περιγραφή.

```
[
  {
    "name" : "AAC-LC (Low Complexity)",
    "encoderCode" : 3,
    "formats" : [
      {
        "formatCodec" : 1,
        "filenameExtensionName" : "3GPP (.3gp)",
        "filenameExtension" : ".3gp",
        "default" : true
      },
      {
        "formatCodec" : 2,
        "filenameExtensionName" : "MPEG-4 (.m4a)",
        "filenameExtension" : ".m4a"
      }
    ]
  },
  {
    "name" : "AMR-WB (Wideband)",
    "encoderCode" : 2,
    "formats" : [
      {
        "formatCodec" : 1,
        "filenameExtensionName" : "3GPP (.3gp)",
        "filenameExtension" : ".3gp"
      }
    ]
  }
]
```

Η δομή ρίζα της παραπάνω διαλέκτου JSON είναι ένα associative array όπου εσωκλείονται τα αντικείμενα των μορφών κωδικοποίησης. Κάθε αντικείμενο κωδικοποίησης πρέπει να περιέχει το κλειδί name όπου η τιμή του εμφανίζεται στον χρήστη. Επίσης πρέπει να περιέχει το κλειδί encoderCode όπου η τιμή του αφορά την τιμή της σταθερά της κωδικοποίησης και δηλώνεται στο MediaRecorder.

Προαιρετικά, μπορεί να υπάρξει και το κλειδί default με τιμή true, σε ένα μόνο αντικείμενο κωδικοποίησης με την βοήθεια του οποίου δηλώνεται η προεπιλεγμένη κωδικοποίηση. Αν υπάρχει σε παραπάνω από ένα αντικείμενα κωδικοποίησης, τότε προεπιλεγμένη ορίζεται η τελευταία όπου έγινε χρήση του. Τέλος, πρέπει να υπάρχει και το κλειδί formats όπου η τιμή του πρέπει να είναι ένα array και να περιέχει αντικείμενα μορφών αρχείων.

Καθένα από τα αντικείμενα μορφών αρχείων ήχου, πρέπει να έχει το κλειδί filenameExtensionName με την τιμή του να εμφανίζεται στον χρήστη. Επίσης, πρέπει να περιέχει το κλειδί formatCodec όπου η τιμή του πρέπει να είναι η τιμή της σταθεράς της μορφής αρχείου ήχου και δηλώνεται στο Media Recorder. Ακόμα, πρέπει να υπάρχει το κλειδί filenameExtension, όπου η τιμή του πρέπει να περιέχει την κατάληξη του αρχείου που θα δημιουργηθεί. Τέλος, μπορεί να περιέχει το κλειδί default με ίδια συμπεριφορά με αυτή των αντικειμένων των μορφών κωδικοποίησης ήχου.

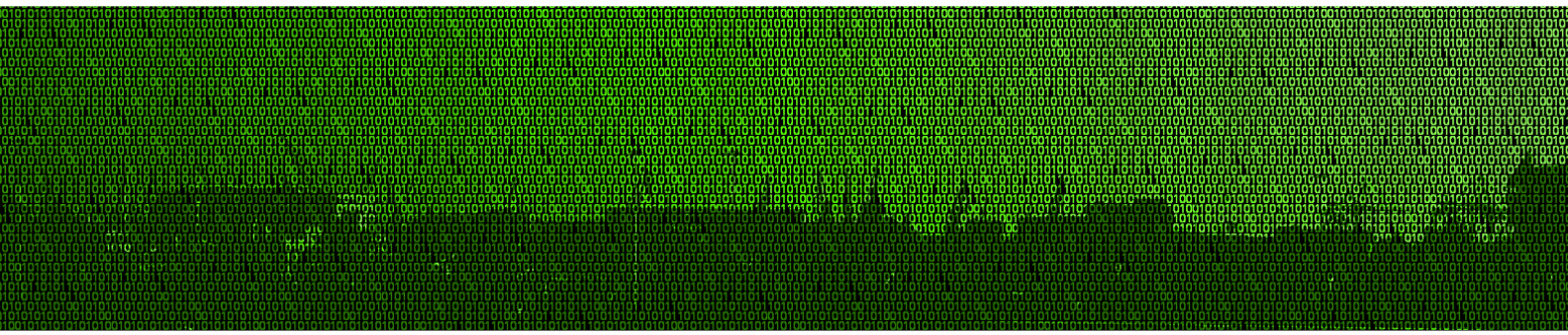
3.5. Εγγραφή από Ροή μέσω Διαδικτύου

Όλες οι συσκευές με την πλατφόρμα Android παρέχουν την δυνατότητα για **σύνδεση με το Διαδίκτυο** με περισσότερους από έναν τρόπους. Οι συνήθεις δύο τρόποι σύνδεσης με το Διαδίκτυο είναι μέσω των δικτύων κινητής τηλεφωνίας και μέσω Wi-Fi. Οι εταιρίες κινητής τηλεφωνίας παρέχουν πολλές υπηρεσίες σύνδεσης με το Διαδίκτυο είτε μεμονωμένα είτε ως μέρος από ένα σύνολο υπηρεσιών επικοινωνίας τηλεφωνίας και σύντομων γραπτών μηνυμάτων (SMS). Η τεχνολογία Wi-Fi επιτρέπει σε ηλεκτρονικές συσκευές να συνδέονται σε ένα Ασύρματο Τοπικό Δίκτυο Υπολογιστών (WLAN) μέσω του οποίου τυπικά υπάρχει διασύνδεση με το Διαδίκτυο.

Πολλοί **ραδιοφωνικοί και άλλοι σταθμοί**, εκπέμπουν το πρόγραμμά τους μέσω Διαδικτύου, είτε πρόκειται για ψυχαγωγικό είτε για ενημερωτικό περιεχόμενο. Οι ραδιοφωνικοί σταθμοί εκπέμπουν το πρόγραμμά τους και μέσω Διαδικτύου ώστε να ξεπεράσουν τους τεχνολογικά γεωγραφικούς, τους οικονομικούς και τους νομικούς περιορισμούς που τους περιορίζουν από το να αποκτήσουν κοινό σε παγκόσμια κλίμακα. Για τους υπόλοιπους, η εκπομπή μέσω Διαδικτύου αποτελεί μια οικονομική λύση τόσο από πλευράς εξοπλισμού όσο και από την πλευρά αδειών που απαιτεί το κράτος.

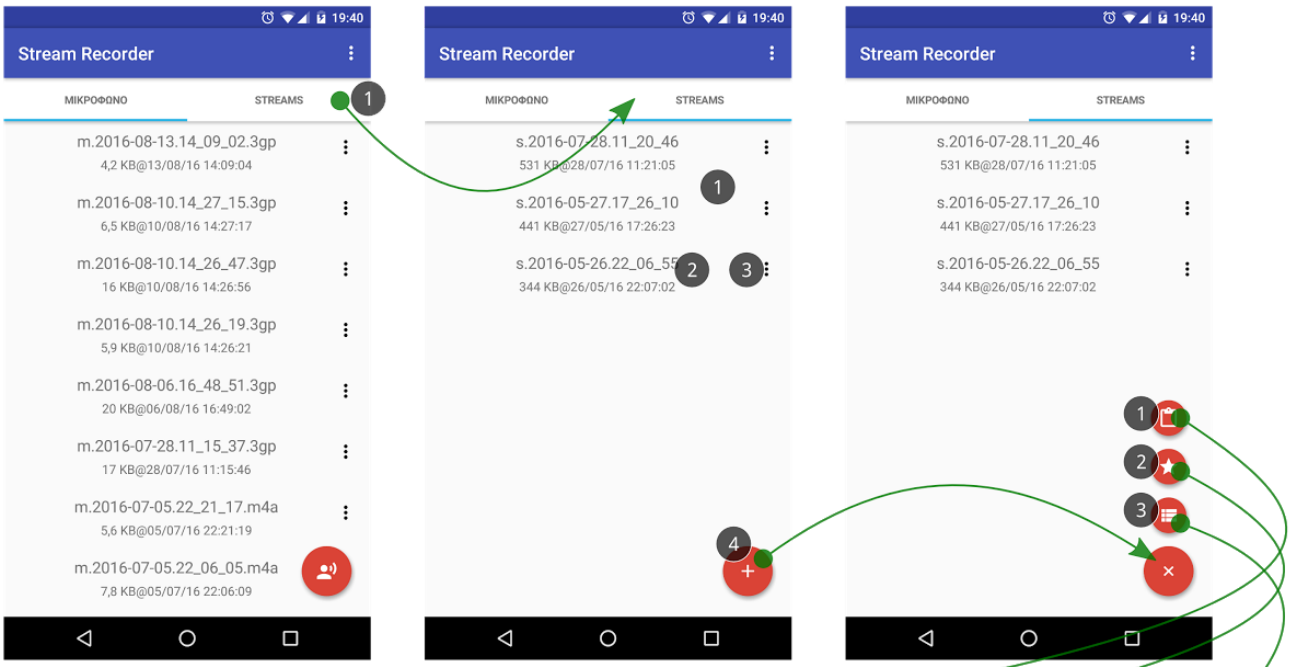
Οι σταθμοί, συνήθως, παρέχουν το πρόγραμμά τους στον τελικό χρήστη μέσω κάποιας ιστοσελίδας. Σε αυτήν την ιστοσελίδα, είτε το παρέχουν μέσω ενσωματωμένου αναπαραραγωγέα οπότε και δεν απαιτείται από τον χρήστη η εγκατάσταση κάποιου εξειδικευμένου λογισμικού είτε παρέχουν κάποιο αρχείο λίστας αναπαραγωγής πολυμέσων (PLS, M3U κ.α.). Και στις δύο περιπτώσεις, ο υπολογιστής του χρήστη καταλήγει με μία HTTP σύνδεση με τον εξυπηρετητή πολυμέσων μέσω μιας διεύθυνσης URL.

Οι σταθμοί πολλοί, οι κωδικοποιήσεις του ήχου πολλές και τα Digital Container Format πολλά. Δύσκολα θα βρεθεί στην αγορά κάποια συσκευή που θα μπορούσε να υποστηρίξει πλήρως όλους αυτούς του συνδυασμούς. Έτσι, δεν μπορεί να τους υποστηρίξει και η εφαρμογή Stream Recorder. Πάραυτα, όσο αφορά την εγγραφή από διαδικτυακή ροή ήχου, όταν αυτή παρέχεται μέσω της τυπικής HTTP σύνδεσης, **η εφαρμογή Stream Recorder μπορεί να υποστηρίξει όλους τους δυνατούς συνδυασμούς κωδικοποιήσεων και container!**

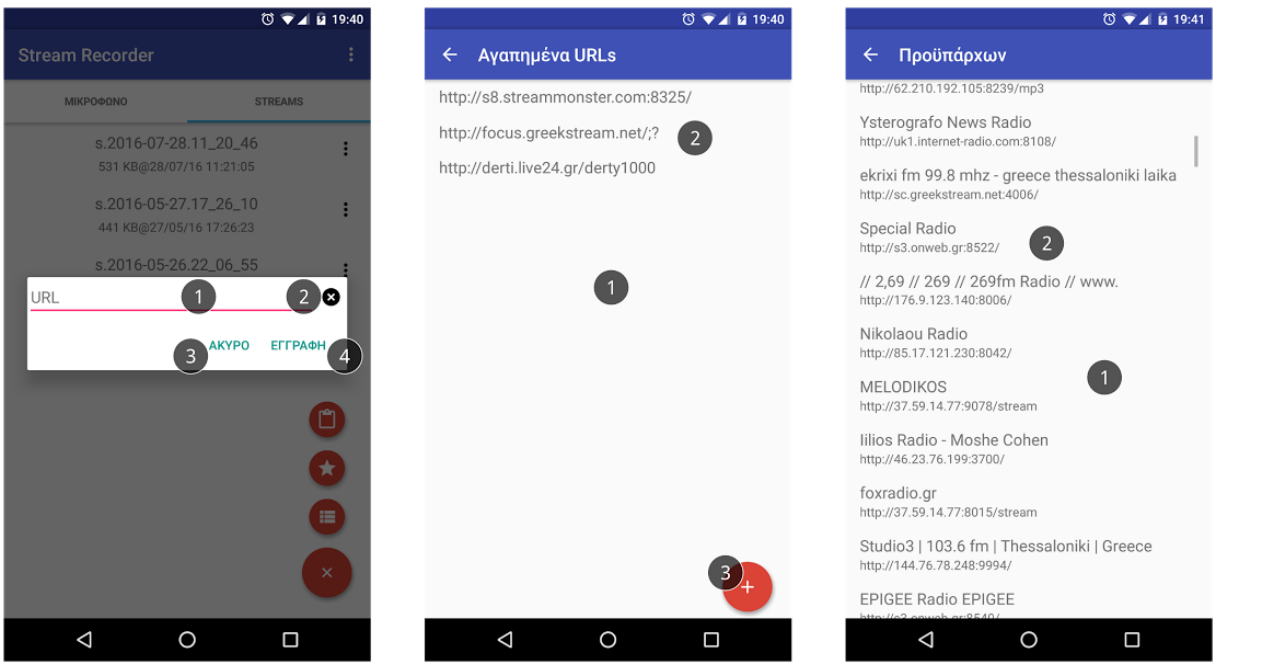


Η τεχνική που αξιοποιήθηκε για την εγγραφή από διαδικτυακή ροή ήχου είναι η αντιγραφή byte by byte. Η εφαρμογή διαβάσει byte προς byte τα δεδομένα που παρέχει ο εξυπηρετητής πολυμέσων και τα αποθηκεύει αυτούσια σε ένα αρχείο. Με αυτόν το τρόπο δεν παίζει ρόλο αν πρόκειται για ασυμπίεστη κωδικοποίηση LPCM (Linear pulse-code modulation) ή αν πρόκειται για απωλεστικής συμπίεσης κωδικοποίηση Vorbis. Το μόνο που παίζει ρόλο και απαιτείται για να λειτουργήσει η εφαρμογή είναι η σύνδεση και η επικοινωνία με τον εξυπηρετητή πολυμέσων να πραγματοποιείται μέσω τυπικής HTTP σύνδεση.

Οθόνη 1 Οθόνη 2 Οθόνη 3



Οθόνη 4 Οθόνη 5 Οθόνη 6



Αναλογία Απεικόνησης 3:4

Η εφαρμογή Stream Recorder παρέχει στον χρήστη τρεις μεθόδους για την εγγραφή από διαδικτυακή ροή ήχου. Πιο συγκεκριμένα, παρέχει την δυνατότητα στον χρήστη να επικολλήσει ένα URL ώστε να ξεκινήσει άμεσα η εγγραφή. Επίσης παρέχει βασικές λειτουργίες διαχείρισης λίστας (προσθήκη/διαγραφή) αγαπημένων URL σταθμών και την δυνατότητα να ξεκινήσει η εγγραφή με κάποιον από αυτούς. Τέλος, παρέχει μια προϋπάρχουσα λίστα αποθηκευμένων σταθμών με την δυνατότητα να ξεκινήσει η εγγραφή με κάποιον από αυτούς. Αυτή, η τελευταία λίστα, δημιουργήθηκε από την βοηθητική εφαρμογή (StreamRecorder-URLsParser, βλ αντίστοιχη ενότητα).

Ειδικότερα για την Οθόνη 1: Πρόκειται για την πρώτη καρτέλα της πρώτης Activity («παράθυρο» στην Android) που εμφανίζεται στον χρήστη με το ξεκίνημα της εφαρμογής και αφορά την εγγραφή από μικρόφωνο.

- Οθόνη 1. Σημείο 1: Με ένα άγγιγμα, ο χρήστης μεταφέρεται στην δεύτερη καρτέλα (Οθόνη 2) όπου αφορά την εγγραφή ροής ήχου από το διαδίκτυο.

Οθόνη 2:

- Οθόνη 2. Σημείο 1: Λίστα με όλες τις εγγραφές από ροή ήχου μέσω διαδικτύου. Καθεμιά από τις εγγραφές της λίστας βρίσκεται στο σύστημα αρχείων της συσκευής και είναι προσβάσημη τόσο από άλλες εφαρμογές όσο και από εργαλεία του συστήματος που επιτρέπουν την μεταφορά του αρχείου σε άλλη συσκευή ή Π/Υ.
- Οθόνη 2. Σημείο 2: Μεμονωμένη εγγραφή ήχου από ροή μέσω διαδικτύου. Με το άγγιγμά της, εμφανίζεται ο ενσωματωμένος αναπαραγωγέας της εφαρμογής Stream Recorder και ξεκινάει αυτόματα η αναπαραγωγή (βλ. εικόνα 2 κεφ. 4.3 Εγγραφή από μικρόφωνο).
- Οθόνη 2. Σημείο 3: Ιδιότητες μεμονωμένης εγγραφής ήχου από ροή μέσω διαδικτύου. Με το άγγιμά της, εμφανίζεται ο ενσωματωμένος αναπαραγωγέας της εφαρμογής Stream Recorder αλλά δεν ξεκινάει αυτόματα η αναπαραγωγή παρά μόνο εμφανίζει τις σχετικές με το ηχητικό πληροφορίες και παρέχει την δυνατότητα της μόνιμης διαγραφή του ηχητικού (βλ. εικόνα 2 κεφ. 4.3 Εγγραφή από μικρόφωνο).

- Οθόνη 2. Σημείο 4: FAB που μεταφέρει τον χρήστη στην Οθόνη 3 και παρέχει λειτουργίες σχετικές με την πηγή απ' όπου ο χρήστης δύναται να επιλέξει URL για εγγραφή ήχου από ροή μέσω διαδικτύου.

Οθόνη 3:

- Οθόνη 3. Σημείο 1: Μεταφέρει τον χρήστη στην Οθόνη 4. Πρόκειται για ένα Dialog Fragmet που δίνει την δυνατότητα στον χρήστη να εισάγει ένα URL απ' όπου επιθυμεί να πραγματοποιηθεί η εγγραφή ροής ήχου.
- Οθόνη 3. Σημείο 2: Μεταφέρει τον χρήστη στην Οθόνη 5 απ΄ όπου ο χρήστης μπορεί να επιλέξει ένα από τα αγαπημένα του URL ή να τα διαχειριστεί (προσθήκη/διαγραφή).
- Οθόνη 3. Σημείο 3: Μεταφέρει τον χρήστη στην Οθόνη 6 όπου υπάρχει μια λίστα από προεισαγμένα URLs.

Οθόνη 4:

- Οθόνη 4. Σημείο 1: Πρόκειται για ένα πεδίο κειμένου στο οποίο ο χρήστης μπορεί να εισάγει το επιθυμητό για εγγραφή URL. Επίσης, το πεδίο κειμένου συμπληρώνεται αυτόματα εάν ο χρήστης έχει URL στο Πρόχειρο (clipboard).
- Οθόνη 4. Σημείο 2: Κουμπί με το οποίο αδειάζει το πεδίο κειμένου.
- Οθόνη 4. Σημείο 3: Flat Button (συγκεκριμένο Material Design στυλ κουμπιού) όπου ακυρώνει την ενέργεια του χρήστη και τον επιστρέφει στην Οθόνη 3.
- Οθόνη 4. Σημείο 4: Flat Button που ξεκινάει την εγγραφή από το URL εισαγμένο στο πεδίο κειμένου URL.

Οθόνη 5:

- Οθόνη 5. Σημείο 1: Πρόκειται για την λίστα με τα αγαπημένα URL του χρήστη.
- Οθόνη 5. Σημείο 2: Μεμονωμένο URL. Με το άγγιγμά του ξεκινάει η εγγραφή ροής ήχου. Με το παρατεταμένο άγγιγμά του, εμφανίζεται προτροπή στον χρήστη για το αν επιθυμεί την διαγραφή του συγκεκριμένου URL.
- Οθόνη 5. Σημείο 3: Εμφανίζει στον χρήστη ένα DialogFragment που του επιτρέπει να εισάγει κάποιο URL. Στην συνέχεια είτε μπορεί απλώς να το αποθηκεύσει στην λίστα με τα αγαπημένα URLs είτε να το αποθηκεύσει στην

λίστα με τα αγαπημένα URLs και να ξεκινήσει άμεσα την εγγραφή της ροής ήχου.

Οθόνη 6:

- Οθόνη 6. Σημείο 1: Πρόκειται για την λίστα με τα προεισαγμένα URLs ροής ήχου. Η λίστα έχει δημιουργηθεί με την βοηθητική εφαρμογή StreamRecorder-URLsParser (βλ. κεφ. 4.6 StreamRecorder-URLsParser).
- Οθόνη 6. Σημείο 2: Μεμονωμένο URL. Με το άγγιγμά του, ξεκινάει άμεσα η εγγραφή της ροής ήχου από τον συγκεκριμένο σταθμό. Σημείωση: ο χρήστης δεν έχει την δυνατότητα να τροποποιήσει αυτήν την λίστα η οποία παρέχεται από προεπιλογής.

3.6. StreamRecorder-URLsParser

Ιστότοπος: https://github.com/GrapsasFilippos/StreamRecorder-URLsParser

Άδεια: GPLv3 (https://www.gnu.org/licenses/gpl-3.0.txt)

Η εφαρμογή **Stream Recorder** παρέχει δυνατότητες για την διαχείριση λίστας αγαπημένων διαδικτυακών σταθμών. Παρόλα αυτά, υπήρχε η επιθυμία να παρέχει και μία **τυπική ενσωματωμένη λίστα ελληνικών διαδικτυακών σταθμών**. Αυτή η λίστα δεν θα έπρεπε να είναι πολύ μικρή αλλά ταυτόχρονα και πολύ μεγάλη ώστε ο χρήστης να την βρει χαώδη. Αυτό θα έδινε την ευκαιρία στον νέο χρήστη να δοκιμάσει άμεσα την εφαρμογή και ταυτόχρονα την πιθανότητα να βρει τον επιθυμητό σταθμό ήδη αποθηκευμένο και έτοιμο για εγγραφή χωρίς επιπλέον διαδικασίες.

Η παραπάνω επιθυμία απαιτεί μια ιδιαίτερα μονότονη και επαναλαμβανόμενη εργασία, η οποία μάλιστα θα ήταν και αρκετά χρονοβόρα στην περίπτωση εισαγωγής 100 διαδικτυακών ραδιοφωνικών σταθμών. Η παραπάνω εργασία αποτελείται από τα παρακάτω βήμα για κάθε έναν σταθμό ξεχωριστά:

1. Λήψη του αρχείου αναπαραγωγής pls το οποίο περιέχει το URL της ροής ήχου.

2. Εισαγωγή του URL της ροής και του ονόματος του σταθμού σε κάποιο αρχείο ώστε να εισαχθεί αργότερα στην εφαρμογή

Η παραπάνω διαδικασία ολοκληρώθηκε από ένα μικρό πρόγραμμα υλοποιημένο στην γλώσσα προγραμματισμού php. Ως εισαγωγή δέχεται ένα αρχείο html και ως εξαγωγή παράγει ένα αλφαριθμητικό σε μορφή JSON και με δεδομένα τα ονόματα των σταθμών και τα URLs των ροών ήχου. Αργότερα, γίνεται η εισαγωγή του JSON αλφαριθμητικού στην εφαρμογή Android και η οποία το χρησιμοποιεί ώστε να δημιουργήσει την λίστα με τους σταθμούς. Τα βήματα που εκτελεί ο αναλυτής είναι τα παρακάτω:

1. Για κάθε διαδικτυακό ραδιοφωνικό σταθμό:
    a. Αναζητεί στην δομή html για το όνομα του σταθμού καθώς και για το URL του αρχείου .pls,
    b. πραγματοποιεί λήψη του αρχείου .pls,
    c. αναζητεί το URL της ροής ήχου,
    d. συνδέει το όνομα με το URL ήχου
2. εξάγει JSON λίστα όπου κάθε στοιχείο της περιέχει το όνομα του σταθμού καθώς και του URL της ροής ήχου.

Ο αναλυτής αρχικά υποστηρίζει html αρχεία από τον διαδικτυακό τόπο internet-radio.com. Όμως, μπορεί εύκολα να προστεθεί υποστήριξη και για άλλους διαδικτυακούς τόπους με λίστες διαδικτυακών ραδιοφωνικών σταθμών.

Έξοδος προγράμματος ευανάγνωστη από τον άνθρωπο:

```
> ./parser -d Greek\ Radio\ Stations.html
20/20 [==================================================>] 100.00% 00:00:00
'LOVELY GREEK RADIO': 'http://78.129.190.50:6144/'
'RadioIn Elladikos': 'http://213.239.206.179:8383/stream'
'Radio Thalassa - Shoutcast Streaming Technology by www.init7.net': 'http://82.197.165.143:80/'
'99.5 To Erotiko Radiofono Tis Kozanis': 'http://94.23.205.82:10432/'
'P.F.S | Athens - Greek': 'http://78.46.73.91:9171/stream'
'MINORE FM - streaming by 24SERVER': 'http://162.253.149.235:8077/stream'
'Sky Radio 99.2 - Ioannina Greece Greek Hellas': 'http://144.76.92.208:9920/stream'
'RADIO LEHOVO 971 GREECE': 'http://s4.onweb.gr:8468/'
```

```
'MAGIC FM AGRINIO': 'http://live.magicfmagrinio.net:9500/'
'ola web radio': 'http://live.isolservers.com:8300/'
'RADIO PHLIO 93.4 VOLOS': 'http://37.59.37.6:10534/'
'Radio Play 91.5 Fm Stereo (Gr Xanthi)': 'http://s6.onweb.gr:8058/'
'Greek mucik  Radiokalavryta mono laika  Greek Laika HELLAS Greece':
'http://213.239.206.179:8030/stream'
'RADIO PIERIA': 'http://s6.onweb.gr:8018/'
'Xroma FM 102': 'http://stream.xromafm.gr:59102/'
'RADIO PONTOS LELEVOSE 101,3 FM KAVALA->STUDIO THESSALONIKI GREECE  HELLAS':
'http://213.239.206.179:9652/'
'Radio Polis 99,4 (Greece)': 'http://s1.onweb.gr:8088/'
'Lakka Souli Radio www.lakkasouliradio.gr GREECE GREEK HELLAS LAIKA RADIO':
'http://eco.onestreaming.com:8286/'
'Sohos FM 88.7 Thessaloniki, Greece': 'http://37.59.32.115:2046/stream'
'Ysterografo News Radio': 'http://uk1.internet-radio.com:8108/'
```

Έξοδος προγράμματος σε μορφή JSON:

```
> ./parser -j Greek\ Radio\ Stations.html
20/20 [==================================================>] 100.00% 00:00:00
[{"title":"LOVELY GREEK RADIO","url":"http:\/\/78.129.190.50:6144\/"},{"title":"RadioIn
Elladikos","url":"http:\/\/213.239.206.179:8383\/stream"},{"title":"Radio Thalassa - Shoutcast
Streaming Technology by www.init7.net","url":"http:\/\/82.197.165.143:80\/"},{"title":"99.5 To
Erotiko Radiofono Tis Kozanis","url":"http:\/\/94.23.205.82:10432\/"},{"title":"P.F.S | Athens -
Greek","url":"http:\/\/78.46.73.91:9171\/stream"},{"title":"MINORE FM - streaming by
24SERVER","url":"http:\/\/162.253.149.235:8077\/stream"},{"title":"Sky Radio 99.2 - Ioannina Greece
Greek Hellas","url":"http:\/\/144.76.92.208:9920\/stream"},{"title":"RADIO LEHOVO 971
GREECE","url":"http:\/\/s4.onweb.gr:8468\/"},{"title":"MAGIC FM
AGRINIO","url":"http:\/\/live.magicfmagrinio.net:9500\/"},{"title":"ola web
radio","url":"http:\/\/live.isolservers.com:8300\/"},{"title":"RADIO PHLIO 93.4
VOLOS","url":"http:\/\/37.59.37.6:10534\/"},{"title":"Radio Play 91.5 Fm Stereo (Gr
Xanthi)","url":"http:\/\/s6.onweb.gr:8058\/"},{"title":"Greek mucik  Radiokalavryta mono laika
Greek Laika HELLAS Greece","url":"http:\/\/213.239.206.179:8030\/stream"},{"title":"RADIO
PIERIA","url":"http:\/\/s6.onweb.gr:8018\/"},{"title":"Xroma FM
102","url":"http:\/\/stream.xromafm.gr:59102\/"},{"title":"RADIO PONTOS LELEVOSE 101,3 FM
KAVALA->STUDIO THESSALONIKI GREECE
HELLAS","url":"http:\/\/213.239.206.179:9652\/"},{"title":"Radio Polis 99,4
(Greece)","url":"http:\/\/s1.onweb.gr:8088\/"},{"title":"Lakka Souli Radio www.lakkasouliradio.gr
GREECE GREEK HELLAS LAIKA RADIO","url":"http:\/\/eco.onestreaming.com:8286\/"},{"title":"Sohos FM
88.7 Thessaloniki, Greece","url":"http:\/\/37.59.32.115:2046\/stream"},{"title":"Ysterografo News
Radio","url":"http:\/\/uk1.internet-radio.com:8108\/"}]
```

4. Συμπεράσματα

Κατά την διάρκεια υλοποίησης της παρούσας πτυχιακής εργασίας αναπτύχθηκε εφαρμογή για την πλατφόρμα Android με την βοήθεια της οποίας γίνεται δυνατή η εγγραφή ηχητικών αρχείων από την εκμετάλλευση διάφορων πηγών ήχου.

Αρχικά, πραγματοποιήθηκε μελέτη για την εκμετάλλευση του ραδιοφωνικού δέκτη όπου διαθέτουν πολλές συσκευές κινητών και ταμπλετών. Ύστερα από διεξοδική έρευνα, διαπιστώθηκε πως κάτι τέτοιο δεν είναι δυνατό. Ως ρίζα του προβλήματος, θεωρήθηκαν οι ελλείψεις στον σχεδιασμό της πλατφόρμας στον οποίο δεν έχει γίνει καμία πρόβλεψη επί του θέματος και φυσικά δεν υπάρχει σχετική τεκμηρίωση. Βέβαια, όπως ήδη αναφέρθηκε υπάρχουν πολλές συσκευές με ραδιοφωνικό δέκτη με τον χειρισμό του να γίνεται μέσω ειδικού λογισμικού. Αυτό συμβαίνει χάρη στους κατασκευαστές συσκευών οι οποίοι όμως υλοποιούν άναρχα αυτήν την δυνατότητα και με μόνο στόχο την παροχή του χαρακτηριστικού της ακρόασης ραδιοφώνου. Συμπεριλαμβάνουν στις συσκευές τους σχετικό κύκλωμα, αναπτύσσουν τον απαραίτητο οδηγό για το λειτουργικό σύστημα Linux και την απαραίτητη εφαρμογή για τον χειρισμό του. Όλο όμως το λογισμικό είναι κλειστού κώδικα και δεν παρέχουν καμία Διεπαφή Προγραμματισμού Εφαρμογών (API) ή σχετική τεκμηρίωση. Ως εκ τούτου, οι εφαρμογές που αναπτύσσονται δεν μπορούν να εκμεταλλευτούν τον ραδιοφωνικό δέκτη για καμία λειτουργία.

Στην συνέχεια, πραγματοποιήθηκε μελέτη για την εγγραφή ηχητικών δεδομένων από διαδικτυακές ροές ήχου διάφορων ραδιοφωνικών και άλλων σταθμών παγκοσμίως. Κατά την διάρκεια της μελέτης διαπιστώθηκε πως υπάρχουν πολλοί σταθμοί οι οποίοι χρησιμοποιούν ποικίλες μορφές κωδικοποίησης ήχου. Μάλιστα, πολλές από αυτές δεν υποστηρίζονται από την πλατφόρμα Android. Έτσι, υλοποιήθηκε μια τεχνική με τη οποία τα δεδομένα αποθηκεύονται ως έχουν. Με αυτόν τον τρόπο η εφαρμογή υποστηρίζει θεωρητικά όλες τις μορφές ήχου είτε αυτές

υποστηρίζονται από την πλατφόρμα είτε όχι. Μόνη προϋπόθεση είναι η διαδικτυακή ροή ήχου να ακολουθεί το πρότυπο HTTP του Internet protocol suite.

Για να γίνει δυνατή η χρήση της παραπάνω δυνατότητας της εφαρμογής από απλούς χρήστες, χωρίς εξειδικευμένες γνώσης, οι οποίες απαιτούνται για την εύρεση των ηχητικών URLs, στην εφαρμογή StreamRecorder προστέθηκε μια λίστα με περισσότερους από 400 ελληνικούς ραδιοφωνικούς σταθμούς. Για την δημιουργία της παραπάνω λίστας, δημιουργήθηκε ένα διαδικτυακό ρομπότ το οποίο αναλύει τον ιστότοπο live24.gr και εξάγει μια λίστα η οποία μπορεί να ενσωματωθεί στον κώδικα της εφαρμογής. Με αυτόν τον τρόπο γίνεται δυνατή η χρήση της εφαρμογής από όλους τους χρήστες.

Επιπλέον των δυνατοτήτων διαχείρισης αγαπημένων σταθμών που παρέχονται από την εφαρμογή μέσω της γραφικής διεπαφής χρήσης, προστέθηκε και η δυνατότητα εισαγωγής λίστας ηχητικών URLs από αρχείο. Έτσι, οι χρήστες μπορούν να αναζητήσουν ή να δημιουργήσουν λίστες με ηχητικά URLs και να τα εισάγουν εύκολα και γρήγορα στην εφαρμογή.

Τέλος, πραγματοποιήθηκε μελέτη για την εκμετάλλευση του μικροφώνου όπου διαθέτουν, αν όχι όλες, σχεδόν όλες οι συσκευές. Σε αυτόν τον τομέα τα πράγματα ήταν αρκετά πιο ελεγχόμενα και κατ' επέκταση πιο εύκολα. Στον σχεδιασμό της πλατφόρμας έχει προβλεφθεί πλήρως η αξιοποίηση του μικροφώνου ως πηγής ήχους και παρέχεται σχετική τεκμηρίωση. Υποστηρίζονται αρκετά containers και κωδικοποιήσεις ήχου καθώς και διάφορες ρυθμίσεις για αυτές. Στην εφαρμογή γίνεται χρήση ενός υποσυνόλου ρυθμίσεων και κωδικοποιήσεων από τις διαθέσιμες της πλατφόρμας.

Σε ό,τι αφορά την ανάπτυξη εφαρμογών, η πλατφόρμα Android είναι ιδιαίτερα προσεγμένη αλλά μπορεί να γίνει και ιδιαίτερα απαιτητική για κάποιον που επιθυμεί την δημιουργία μιας σύγχρονης και ποιοτικής εφαρμογής. Οι απαιτήσεις αφορούν την εκμάθησή της και την διατήρηση της τεχνογνωσίας, αφού οι αλλαγές είναι κυριολεκτικά ραγδαίες.

Για την ανάπτυξη εφαρμογών διατίθεται το ειδικό Android Studio IDE το οποίο είναι ιδιαίτερα βολικό για τον προγραμματιστή, ενώ δεν λείπουν και άλλα IDE και εργαλεία για την μεταγλώττιση των εφαρμογών. Το «πακέτο» ολοκληρώνεται με το ηλεκτρονικό κατάστημα διανομή εφαρμογών Google Play ενώ η διανομή τους δεν περιορίζεται σε αυτό.

Ιδιαίτερο χαρακτηριστικό στην ανάπτυξη εφαρμογών αποτελεί το πλήθος των γραμμών πλεύσης (guidelines) που προωθούνται από την πλατφόρμα και που αυξάνει το κόστος για την εκμάθησή της. Παράλληλα όμως με αυτόν τον τρόπο αυξάνεται η ποιότητα των παραγόμενων εφαρμογών και η επαναχρησιμοποίηση του κώδικα. Χαρακτηριστικό παράδειγμα των παραπάνω αποτελεί η χρήση του GUI συστατικού λιστών ListView. Ενώ η φαινομενικά εύκολη λύση για ένα προγραμματιστή θα ήταν η εκτέλεση ενός ερωτήματος στην Βάση Δεδομένων και το γέμισμα της λίστας με την βοήθεια μιας δομής επανάληψη, στην πλατφόρμα Android τα πράγματα είναι πιο περίπλοκα. Η παραπάνω τεχνική κρύβει παγίδες με «πάγωμα» της γραφικής διεπαφής χρήστη έως διαρροές μνήμης. Η γραμμή πλεύσης υπαγορεύει την χρήση τεσσάρων συστατικών που επικοινωνούν σε σειρά το ένα με το άλλο (ListView, CursorAdapter, ContentProvider, SQLiteDatabase). Αυτός ο τρόπος είναι προφανώς αρκετά περίπλοκος για έναν νέο χρήστη της πλατφόρμας, όμως για κάποιον που τον έχει συνηθίσει αποτελεί μια καθαρή λύση με αρκετό επαναχρησιμοποιούμενο κώδικα.

Ένα άλλο ιδιαίτερο (αρνητικό) χαρακτηριστικό στην ανάπτυξη εφαρμογών αποτελεί ο λεγόμενος κατακερματισμός (fragmentation) των εκδόσεων της πλατφόρμας και κατ' επέκταση των εκδόσεων του API. Αν για μια εφαρμογή ο στόχος είναι η υποστήριξη περισσότερων των μισών διαθέσιμων στην αγορά συσκευών, κατά πάσα πιθανότητα θα χρειαστεί να υλοποιηθεί τουλάχιστον ένα κομμάτι δύο φορές. Μια φορά χρησιμοποιώντας την παλιά έκδοση του API και μια φορά χρησιμοποιώντας την νέα έκδοση του API. Για παράδειγμα, η πρόσβαση στο σύστημα αρχείων διαφέρει αναλόγως της έκδοσης του API. Στις μια περίπτωση αρκεί η χρήση της κλάσης File της γλώσσας προγραμματισμού Java ενώ στις

νεότερες εκδόσεις της πλατφόρμας για να περιγραφεί η υλοποίηση απαιτείται διάγραμμα μιας σελίδας με διεργασίες που πρέπει να εκτελεστούν πριν και μετά από κάποιες άλλες. Η περιπλοκότητα αυξάνεται ραγδαία όταν αυτές οι δύο τελείως διαφορετικές υλοποιήσεις πρέπει να συνδυαστούν μαζί.

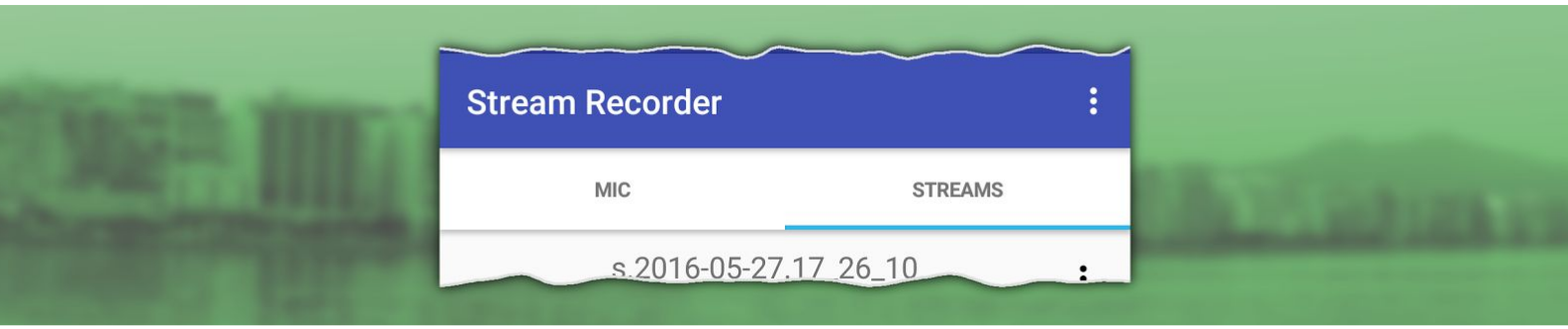
Εν κατακλείδι, η ανάπτυξη εφαρμογών για την πλατφόρμα Android απαιτεί αρκετή εξειδίκευση και αδιάκοπη ενημέρωση για τις αλλαγές που πραγματοποιούνται. Το κόστος της εξειδίκευσης είναι αξιοσέβαστο και απαιτεί την συνεχή παραγωγή εφαρμογών ώστε να δικαιολογηθεί. Κατά τα άλλα αποτελεί μια καλά σχεδιασμένη και σταθερή πλατφόρμα με αρκετά βολικά και σταθερά εργαλεία ανάπτυξης.

5. Ιστογραφία

- http://www.idc.com/prodserv/smartphone-os-market-share.jsp
- https://developer.android.com/
- http://www.htc.com/us/go/htc-software-updates-process/
- https://play.google.com/store/apps/details?id=com.grapsas.smokelogger
- https://play.google.com/store/apps/details?id=com.mikersmicros.fm_unlock
- https://developer.android.com/reference/android/media/MediaRecorder.OutputFormat.html
- https://developer.android.com/reference/android/media/MediaRecorder.OutputFormat.html
- https://developer.android.com/reference/android/media/MediaRecorder.OutputFormat.html#RAW_AMR

6. Παραρτήματα

6.1. Torn Out effect (GIMP Script)



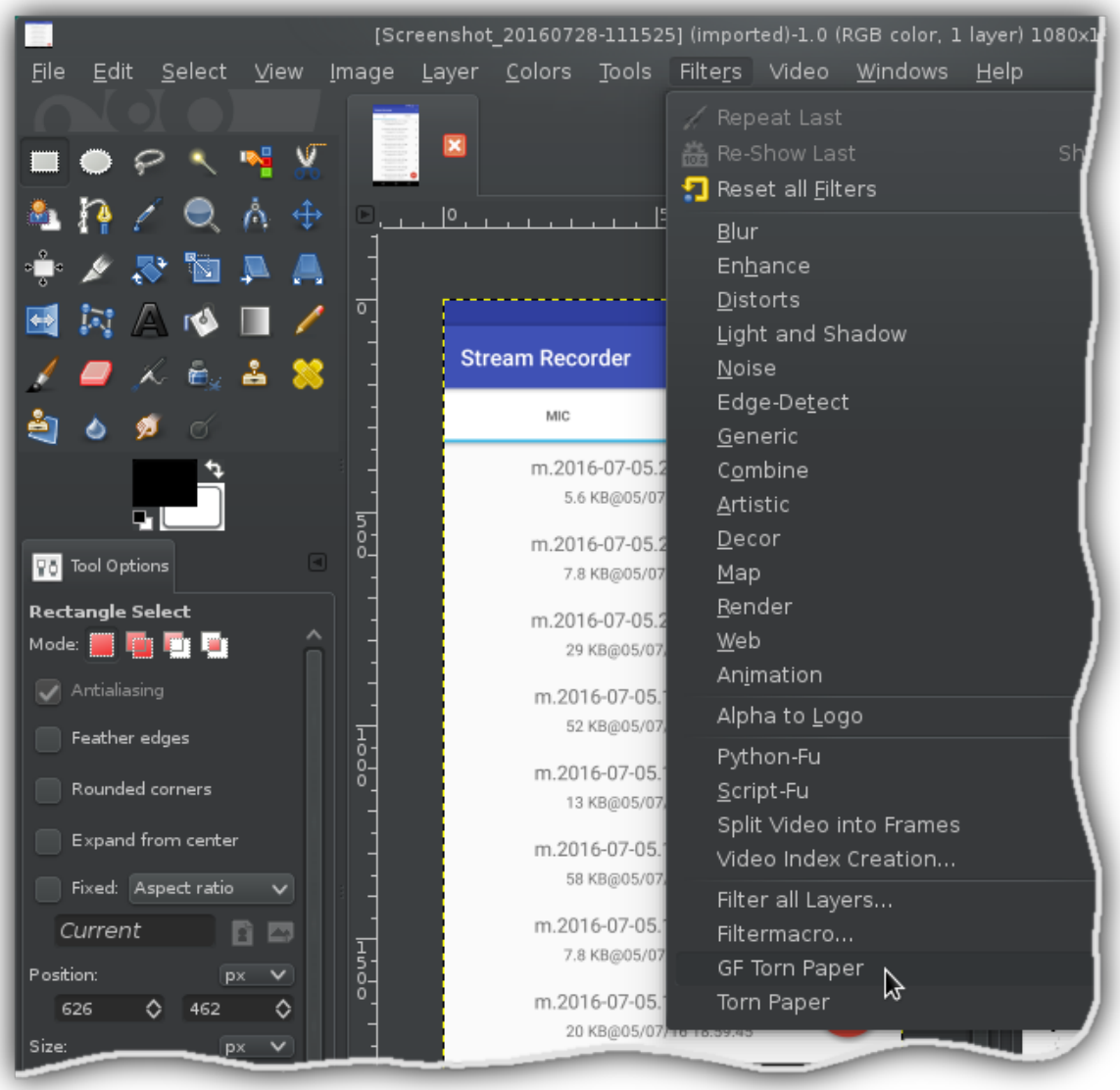
Ιστότοπος: https://github.com/GrapsasFilippos/Script_Fu_GFTornPaper

Άδεια: GPLv3 (https://www.gnu.org/licenses/gpl-3.0.txt)

Στα πλαίσια της πρακτικής εργασίας, εκτός των άλλων, δημιουργήθηκε ένα ακόμα script, με την βοήθεια του οποίου έγινε η **επεξεργασία των εικόνων** που παρουσιάζονται στο παρόν έγγραφο. Αυτό συνέβη διότι παρέστη η ανάγκη της επεξεργασίας πολλών εικόνων με έναν συγκεκριμένο τρόπο: Εφαρμογή του εφέ «σκισμένου χαρτιού» και του εφέ «πίπτουσας σκιάς». Το δεύτερο εφέ υπήρχε. Για το πρώτο, δεν βρέθηκε κάτι που να καλύπτει τις απαιτήσεις. Εναλλακτικά, η χειροκίνητη επεξεργασία των εικόνων δεν κρίθηκε συμφέρουσα διότι το εφέ «σκισμένου χαρτιού» απαιτεί αρκετό χρόνο.

Για την επεξεργασία των εικόνων, χρησιμοποιήθηκε το πρόγραμμα επεξεργασίας ψηφιογραφικών εικόνων **GIMP** (GNU Image Manipulation Program). Το GIMP είναι το πιο γνωστό λογισμικό στην κατηγορίας του. Είναι Ελεύθερου και Ανοικτού Κώδικα, παρέχοντας έτσι την ελευθερία στην χρήση, στην τροποποίηση του κώδικά του και στην διανομή του. Είναι ανεξάρτητο πλατφόρμας (cross-platform ή multi-platform) και διαθέσιμο για GNU+Linux, OS X, Windows και άλλες. Επίσης είναι ένα υψηλής ποιότητας framework για την επεξεργασία εικόνας και διαθέσιμο για πολλές γλώσσες προγραμματισμού όπως C, C++, Perl, Python, Scheme και άλλες.

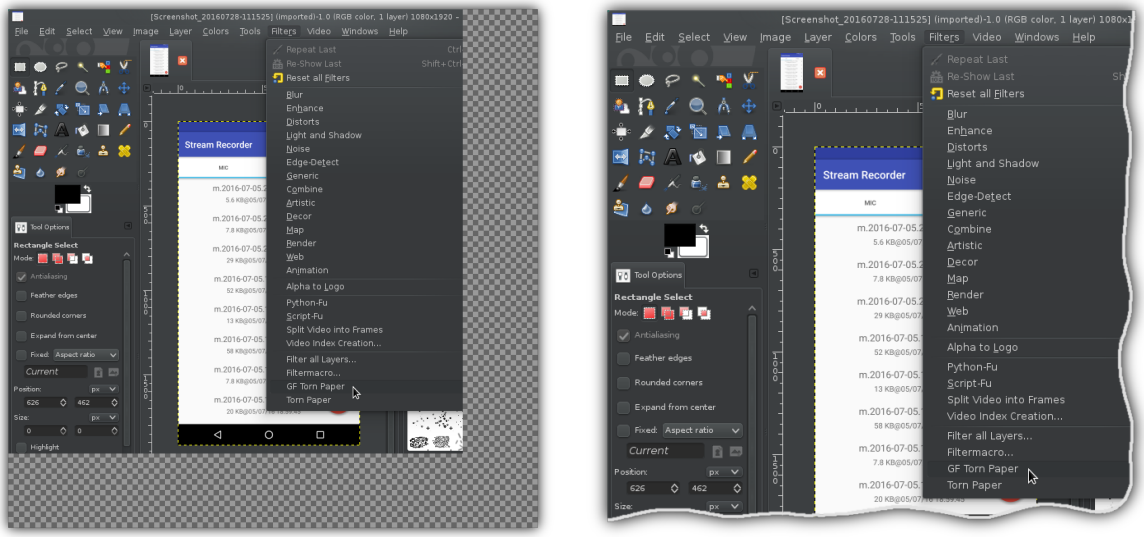
Για την δημιουργία του script χρησιμοποιήθηκε η διάλεκτος **Scheme** της γλώσσας προγραμματισμού **Lisp**. Η επιλογή της παραπάνω τεχνολογίας έγινε διότι υπήρχε αρκετή τεκμηρίωση στο διαδίκτυο τόσο από τον επίσημο ιστότοπο όσο και από διάφορους άλλους. Ακόμα και με μηδενική εμπειρία στην Scheme και στο GIMP framework, η δημιουργία του εφέ «σκισμένου χαρτιού» ολοκληρώθηκε σε σύντομο χρονικό διάστημα. Πάντως, για τους προγραμματιστές που είναι εξοικειωμένοι με τις γλώσσες προγραμματισμού C, C++, Java, Python, προτείνεται η επιλογή μιας αντίστοιχης γλώσσας προγραμματισμού αντί της Scheme για την επέκταση του GIMP.



Το script που προσθέτει το εφέ «Tron Out» στο GIMP λειτουργεί περίφημα κατά την επεξεργασία των εικόνων. Παρόλα αυτά, με την χρήση της Scheme, προστέθηκε η δυνατότητα της εφαρμογής του παραπάνω **εφέ μέσω κονσόλας**. Έτσι, εκτελείται το GIMP σε περιβάλλον κονσόλας, παρέχεται το όνομα αρχείου εισαγωγής, το όνομα αρχείου εξαγωγής στην κατάλληλη διαδικασία για την εφαρμογή του εφέ σε κάποια εικόνα. Επειδή σε περιβάλλον κονσόλας ο χρήστης δεν δύναται να επιλέξει το κομμάτι που επιθυμεί να «σκιστεί», το εφέ επιλέγει αυτόματα την επιφάνεια της εικόνας που είναι αδιαφανή.

```
> gimp -i \
```

```
-b '(gf-batch-torn-paper "/dir/TornPaper Example.xcf" "/dir/TornPaper Example.png")' \
-b '(gimp-quit 0)'
```



a b

* Αυτόματη επιλογή περιοχής «σχισίματος» βάση διάφανης περιοχής.

Επιπλέον, δημιουργήθηκε η εντολή gimp-effect-GFTornPager με τη γλώσσα προγραμματισμού php η οποία δέχεται ως είσοδο έναν κατάλογο, αναζητεί όλα τα αρχεία του προγράμματος GIMP, και εκτελεί αυτόματα την παραπάνω «μακρυά» εντολή για όλα τα αρχεία. Με το παρόν, ολοκληρώνεται η διεπαφή χρήσης του εφέ και παρέχονται όλοι οι τρόποι λειτουργίας για χρήση αναλόγως των απαιτήσεων της κάθε περίπτωσης.

- Χρήση από το γραφικό περιβάλλον του GIMP,
- χρήση από το περιβάλλον κονσόλας του GIMP,
- χρήση από το περιβάλλον κονσόλας για πολλά αρχεία,
- χρήση από άλλον πρόγραμμα/υπηρεσία μέσω περιβάλλοντος κονσόλας.

```
> gimp-effect-GFTornPager /Πτυχιακή/
gimp -i \
-b '(gf-batch-torn-paper "/Πτυχιακή/Torn Paper SS.xcf" "/Πτυχιακή/Torn Paper SS.png")' \
-b '(gf-batch-torn-paper "/Πτυχιακή/TornPaper Example.xcf" "/Πτυχιακή/TornPaper Example.png")' \
```

```
-b '(gf-batch-torn-paper "/Πτυχιακή/Torn Paper as.xcf" "/Πτυχιακή/Torn Paper as.png")' \
-b '(gimp-quit 0)'

batch command executed successfully
batch command executed successfully
batch command executed successfully
```

Ολοκληρώνοντας το παραπάνω έργο, μέσα στο πακέτο πηγαίου κώδικα (tarball) συμπεριλαμβάνεται το αρχείο Makefile με οδηγίες για το πρόγραμμα **GNU Make** ώστε να είναι δυνατή η εγκατάσταση και απεγκατάσταση του παραπάνω έργου σε συστήματα με το περιβάλλον GNU+Linux. Το gmake είναι ελεύθερο λογισμικό και μέρος του φημισμένου GNU Project. Παρέχει δυνατότητες για την κατασκευή των αρχείων μη-πηγαίου κώδικα από τα αρχεία πηγαίου κώδικα και δυνατότητες εγκατάστασης και απεγκατάστασης των προγραμμάτων που διανέμονται με αυτό.

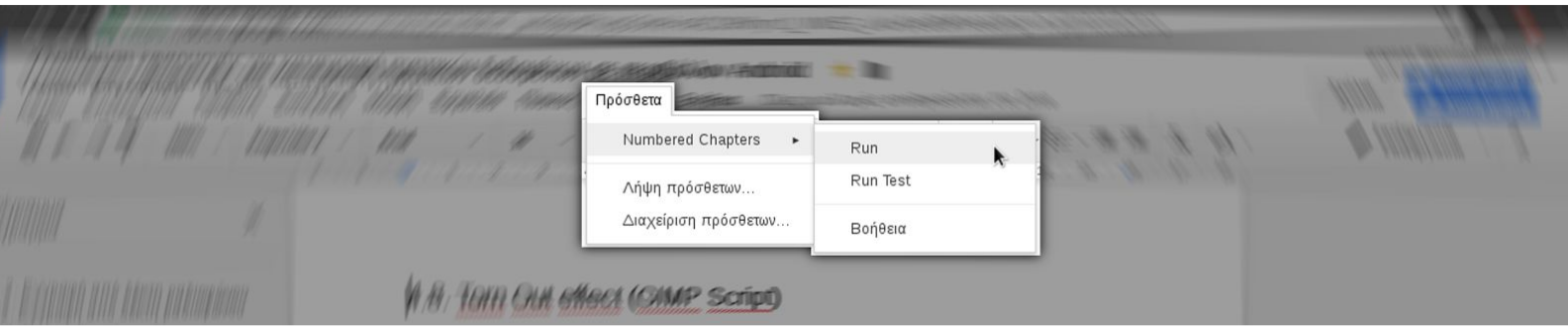
```
Script_Fu_GFTornPaper # make install
mkdir -p /usr/bin
cp ./gimp-effect-GFTornPager /usr/bin
chmod +x /usr/bin/gimp-effect-GFTornPager
mkdir -p /usr/share/gimp/2.0/scripts
cp ./GFTornPaper.scm /usr/share/gimp/2.0/scripts
cp ./GFTornPaperBatch.scm /usr/share/gimp/2.0/scripts

Script_Fu_GFTornPaper # make uninstall
rm /usr/bin/gimp-effect-GFTornPager
rm /usr/share/gimp/2.0/scripts/GFTornPaper.scm
rm /usr/share/gimp/2.0/scripts/GFTornPaperBatch.scm
```

Τέλος, στο πακέτο του πηγαίου κώδικα (tarball) συμπεριλαμβάνεται το αρχείο Script_Fu_GFTornPaper.spec με οδηγίες για την δημιουργία του αρχείου εγκατάστασης **RPM** (Red Hat Package Manager) για την διανομή του GNU+Linux OpenSUSE 13.2 για την οποία έχει φτιαχτεί και δοκιμαστεί. Αν και θεωρητικά θα μπορούσε να λειτουργήσει και σε άλλες διανομές GNU+LINUX που βασίζονται στο RPM. Το RPM αρχικά δημιουργήθηκε για την διανομή Red Hat αν και σήμερα χρησιμοποιείται από πολλές άλλες γνωστές διανομές. Πλέον είναι το πιο

διαδεδομένο σύστημα διαχείρισης πακέτων λογισμικού μαζί με αυτό της διανομής GNU+Linux Debian.

```
# zypper install ./Script_Fu_GFTornPaper-0.1.0-1.noarch.rpm
Loading repository data...
Reading installed packages...
Resolving package dependencies...

The following NEW package is going to be installed:
  Script_Fu_GFTornPaper

1 new package to install.
Overall download size: 3.6 KiB. Already cached: 0 B  After the operation, additional 3.9 KiB will
be used.
Continue? [y/n/? shows all options] (y): y
Retrieving package Script_Fu_GFTornPaper-0.1.0-1.noarch
(1/1),   3.6 KiB (  3.9 KiB unpacked)
Checking for file conflicts: ..................................[done]
(1/1) Installing: Script_Fu_GFTornPaper-0.1.0-1
.....................................................................[done]


# zypper remove Script_Fu_GFTornPaper
Loading repository data...
Reading installed packages...
Resolving package dependencies...

The following package is going to be REMOVED:
  Script_Fu_GFTornPaper

1 package to remove.
After the operation, 3.9 KiB will be freed.
Continue? [y/n/? shows all options] (y): y
(1/1) Removing Script_Fu_GFTornPaper-0.1.0-1
.....................................................................[done]
```

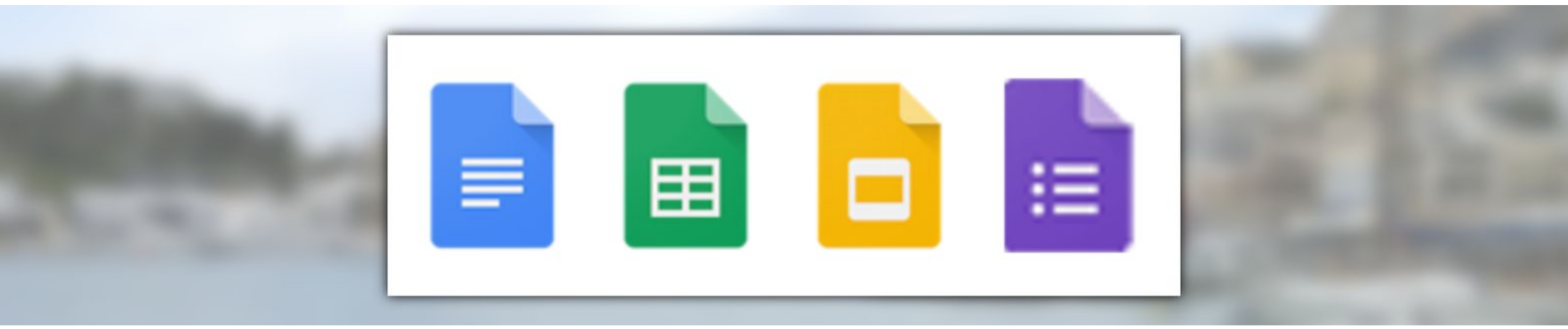
6.2. Numbered Chapters (Google Apps Script)



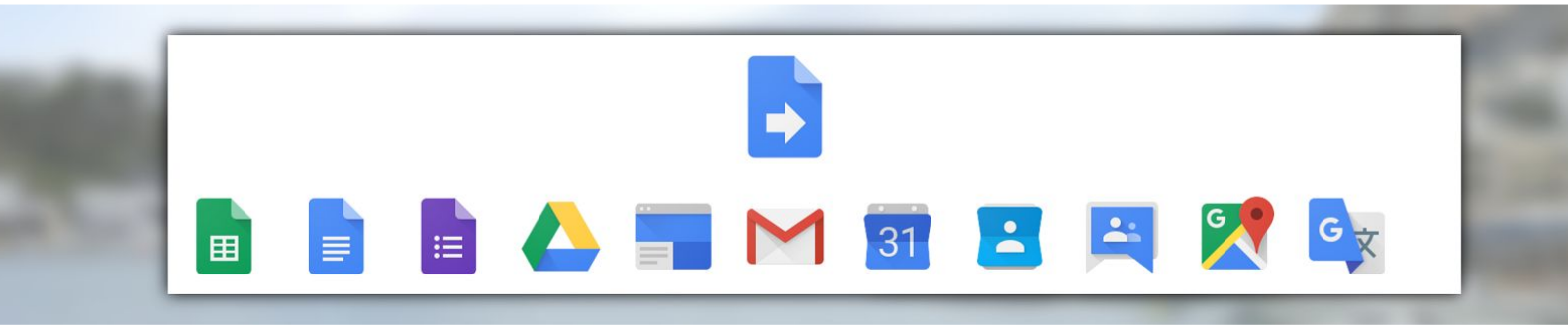
Ιστότοπος: https://gist.github.com/GrapsasFilippos/...

Άδεια: GPLv3 (https://www.gnu.org/licenses/gpl-3.0.txt)

Στα πλαίσια πάλι της πρακτικής άσκησης, υπήρχε η ανάγκη δημιουργίας ενός ακόμη script για τα Έγγραφα Google. Το παρόν έγγραφο της πτυχιακής εργασίας γράφτηκε χρησιμοποιώντας το παραπάνω πρόγραμμα επεξεργασίας κειμένου. Κάλυπτε όλες τις απαραίτητες απαιτήσεις εκτός από την αυτόματη αρίθμηση των κεφαλαίων. Η χειροκίνητη αρίθμησή τους κρίθηκε χρονοβόρα βάση του αριθμού τους και της πιθανής ανάγκης αναδιάταξής τους. Ταυτόχρονα, με χρήση του Google Apps Script παρέχεται η δυνατότητα ταχείας επέκτασης των δυνατοτήτων της σουίτας γραφείου, τουλάχιστον για απλές εργασίες.



Η σουίτα γραφείου **Google Docs**, δεν είναι Ελεύθερο Λογισμικό, το οποίο για την ώρα έχει μείνει πίσω στον τομέα του Νέφους, αλλά παρέχει αξιοπρεπή δυνατότητες. Αρχικά, καλύπτει τις απαιτήσεις ανεξαρτησίας πλατφόρμας (cross-platform), όλες οι αλλαγές αποθηκεύονται αυτόματα στο Νέφος και τα έγγραφα είναι διαθέσιμα τόσο από Προσωπικούς Υπολογιστές όσο και από Κινητές Συσκευές. Η πιο ενδιαφέρον δυνατότητα είναι η κοινή χρήση των εγγράφων και η συνεργασία με την ταυτόχρονη προβολή και επεξεργασία κάποιου αρχείου. Μάλιστα, ο ένας χρήστης βλέπει σε πραγματικό χρόνο τις αλλαγές και τις προσθήκες που κάνει ο άλλος χρήστης. Τέλος, στους Προσωπικούς Υπολογιστές είναι προσβάσιμο μέσω του Περιηγητή Ιστού και δεν απαιτείται η εγκατάσταση οποιαδήποτε άλλου λογισμικού σε τοπικό επίπεδο.



Το **Google Apps Script** είναι μια scripting γλώσσα προγραμματισμού. Είναι βασισμένη στην JavaScript και για αυτό το λόγο είναι εύκολη στην εκμάθηση, είναι διαδεδομένη και επιτρέπει την ταχεία ανάπτυξη μικρών εφαρμογών. Επίσης χρησιμοποιείται στα πρόσθετα του Google Docs. Δεν απαιτείται η εγκατάσταση κάποιου επιπρόσθετου λογισμικού σε τοπικό επίπεδο και η ανάπτυξη και αποσφαλμάτωση γίνεται μέσω του Περιηγητή Ιστού.

Το πρόσθετο **Numbered Chapters** που αναπτύχθηκε για τα Έγγραφα Google είναι ένα μικρό σενάριο. Αναζητεί όλες τις επικεφαλίδες του εγγράφου και

αφού τις βρει, τις τροποποιεί και προσθέτει στην αρχή τους την κατάλληλη αρίθμηση, αν αυτή λείπει, με βάση τα κεφάλαια και τα υποκεφάλαια. Αν υπάρχει ήδη αρίθμηση, την αναγνωρίζει, την αφαιρεί και προσθέτει την σωστή. Επίσης, υπάρχει και η δυνατότητα να μην προστεθεί αρίθμηση σε κάποιο συγκεκριμένο κεφάλαιο αν αυτό απαιτηθεί από τον χρήστη. Για να γίνει αυτό θα πρέπει το κεφάλαιο να τελειώνει με κενό χαρακτήρα. Τέλος, επειδή τα περιεχόμενα κατασκευάζονται αυτόματα από σχετική δυνατότητα που παρέχεται ήδη από τα Έγγραφα Google με βάση τις επικεφαλίδες που υπάρχουν στο έγγραφο. Η αρίθμηση εμφανίζεται και στα περιεχόμενα.

6.3. Κώδικας

6.3.1. MediaURL.java

```
1  package com.grapsas.android.streamrecorder.misc.media;
2
3
4  import android.support.annotation.NonNull;
5  import android.support.annotation.Nullable;
6
7
8  /**
9   * Created by graphi on 5/24/16.
10  */
11 public class MediaURL {
12
13    private String pTitle;
14    private String pURL;
15
16
17    public MediaURL( @Nullable String title, @NonNull String url ) {
18       this.pTitle = title;
19       this.pURL = url;
20    }
21
22
23    @Nullable
24    public String getTitle() {
25       return this.pTitle;
26    }
27
28    @NonNull
29    public String getURL() {
30       return this.pURL;
31    }
32 }
33
```

6.3.2. Recorder.java

```
1  package com.grapsas.android.streamrecorder.misc.media;
2
3
4  import java.io.FileDescriptor;
5
6
7  public interface Recorder {
8
9     boolean startRecording( FileDescriptor fd );
10    boolean stopRecording();
```

```
11
12 }
13
```

6.3.3. MicRecorder.java

```
1  package com.grapsas.android.streamrecorder.misc.media;
2
3
4  import android.media.MediaRecorder;
5  import android.support.annotation.Nullable;
6
7  import java.io.FileDescriptor;
8
9
10 public class MicRecorder implements Recorder {
11
12   private MediaRecorder recorder;
13
14   private Exception exception;
15
16
```

```
17   public boolean startRecording( FileDescriptor fd ) {
18     recorder = new MediaRecorder();
19     recorder.setAudioSource( MediaRecorder.AudioSource.MIC );
20     recorder.setOutputFormat( MediaRecorder.OutputFormat.THREE_GPP );
21     recorder.setOutputFile( fd );
22     recorder.setAudioEncoder( MediaRecorder.AudioEncoder.AMR_NB );
23
24     try {
25       recorder.prepare();
26     } catch( java.io.IOException e ) {
27       e.printStackTrace();
28       this.stopRecording();
29       this.exception = new Exception( e );
30       return false;
31     }
32
33     try {
34       recorder.start();
35     }
36     catch( IllegalStateException e ) {
37       e.printStackTrace();
38       this.stopRecording();
39       this.exception = new Exception( e );
40       return false;
41     }
42
43     return true;
44   }
45
```

```
46   public boolean stopRecording() {
47     if( this.recorder == null )
48       return true;
49     this.recorder.stop();
50     this.recorder.release();
51     this.recorder = null;
```

```
52
53          return true;
54      }
55
56      @Nullable
57      public Exception getException() {
58          return this.exception;
59      }
60
61  }
62
```

6.3.4. StreamRecorder.java

```
1   package com.grapsas.android.streamrecorder.misc.media;
2
3
4   import android.media.MediaPlayer;
5   import android.os.AsyncTask;
6   import android.os.ParcelFileDescriptor;
7   import android.support.annotation.NonNull;
8   import android.support.annotation.Nullable;
9   import android.widget.Chronometer;
10
11  import com.grapsas.android.streamrecorder.misc.Misc;
12
13  import java.io.FileDescriptor;
14  import java.io.FileOutputStream;
15  import java.io.IOException;
16  import java.io.InputStream;
17  import java.lang.ref.WeakReference;
18  import java.net.MalformedURLException;
19  import java.net.URL;
20
21  import okhttp3.OkHttpClient;
22  import okhttp3.Request;
23  import okhttp3.Response;
24
25
26  public class StreamRecorder implements Recorder {
27
28      private ThreadsConnector pTC;
29      private NetworkByteByByteCopy networkBBCp;
30
31      private String pUrl;
32      private WeakReference< Chronometer > pwChronometer;
33
34
35      public StreamRecorder( String url, Chronometer chronometer ) {
36          this.pUrl = url;
37          this.pwChronometer = new WeakReference<>( chronometer );
38      }
39
40      @Nullable
41      public Chronometer getChronometer() {
42          if( this.pwChronometer == null || this.pwChronometer.get() == null )
```

```
43          return null;
44       return this.pwChronometer.get();
45    }
46
47
```

```
48    @Override
49    public boolean startRecording( FileDescriptor fd ) {
50       if( this.networkBBCp != null  ) {
51          if( this.networkBBCp.isFinished() )
52             this.networkBBCp = null;
53          else
54             this.networkBBCp.cancel( true );
55       }
56
57       this.pTC = new ThreadsConnector( fd, pUrl );
58       if( this.getChronometer() != null )
59          this.getChronometer().setOnChronometerTickListener(
60                ( new MediaChronometer( this.pTC ) ) );
61       this.networkBBCp = new NetworkByteByByteCopy( this.pTC );
62       this.networkBBCp.execute();
63
64       return true;
65    }
66
```

```
67    @Override
68    public boolean stopRecording() {
69       if( this.getChronometer() != null )
70          this.getChronometer().setOnChronometerTickListener( null );
71       this.networkBBCp.stop();
72
73       return true;
74    }
75
76
```

```
77    private static class MediaChronometer implements Chronometer.OnChronometerTickListener {
78
79       private ThreadsConnector pTC;
80       private ParcelFileDescriptor pPFD;
81       private MediaPlayer pPlayer;
82
83
```

```
84       public MediaChronometer( ThreadsConnector threadsConnector ) {
85          this.pTC = threadsConnector;
86          try {
87             this.pPFD = ParcelFileDescriptor.dup( this.pTC.getFD() );
88          } catch( IOException e ) {
89             e.printStackTrace();
90          }
91       }
92
```

```
93       @Override
94       public void onChronometerTick( Chronometer chronometer ) {
95          String str = "";
96          str += Misc.humanBytes( this.pTC.getBytesRead(), true );
97
98          chronometer.setText( str );
```

```
99        }
100
101    }
102
103    private static class ThreadsConnector {
104
105        // TODO: keep pfd instead of fd.
106        private FileDescriptor pFileDescriptor;
107
108        private boolean pRecording;
109        private long pBytesRead;
110        private boolean pRecordingFinished = false;
111
112        private String pUrl;
113
114
115        public ThreadsConnector( @NonNull FileDescriptor fileDescriptor, @NonNull String url ) {
116            this.pUrl = url;
117            this.pFileDescriptor = fileDescriptor;
118        }
119
120        @NonNull
121        public FileDescriptor getFD() {
122            return this.pFileDescriptor;
123        }
124
125        @NonNull
126        public String getUrl() {
127            return this.pUrl;
128        }
129
130        public boolean isRecording() {
131            return this.pRecording;
132        }
133
134        public void setRecording( boolean recording ) {
135            this.pRecording = recording;
136        }
137
138        public long getBytesRead() {
139            return this.pBytesRead;
140        }
141
142        public void setBytesRead( long bytesRead ) {
143            this.pBytesRead = bytesRead;
144        }
145
146        public boolean isRecordingFinished() {
147            return this.pRecordingFinished;
148        }
149
```

```
150     public void setRecordingFinished( boolean finished ) {
151         this.pRecordingFinished = finished;
152     }
153 
154 }
155 
156 
```

```
157 private static class NetworkByteByByteCopy extends AsyncTask< Void, Void, Boolean > {
158 
159     private ThreadsConnector pTC;
160 
161     private boolean pStop = false;
162     private boolean pFinished = false;
163 
164 
```

```
165     public NetworkByteByByteCopy( ThreadsConnector threadsConnector ) {
166         this.pTC = threadsConnector;
167     }
168 
```

```
169     @Override
170     protected Boolean doInBackground( Void... params ) {
171         ParcelFileDescriptor pfd;
172         try {
173             pfd = ParcelFileDescriptor.dup( this.pTC.getFD() );
174         } catch( IOException e ) {
175             e.printStackTrace();
176             return false;
177         }
178 
179         FileOutputStream outputStream;
180         try {
181             URL url = new URL( this.pTC.getUrl() );
182             InputStream inputStream;
183 
184             OkHttpClient okHttpClient = new OkHttpClient();
185             Request request = new Request.Builder()
186                     .url( url )
187                     .build();
188             Response response = okHttpClient.newCall( request ).execute();
189             inputStream = response.body().byteStream();
190 
191             outputStream = new FileOutputStream( pfd.getFileDescriptor() );
192             this.pTC.setRecording( true );
193             int c;
194             int bytesRead = 0;
195             boolean lStop = false;
196             while( ( c = inputStream.read() ) != -1 && !lStop ) {
197                 this.pTC.setBytesRead( ++bytesRead );
198                 outputStream.write( c );
199                 lStop = this.isCancelled() || this.pStop;
200             }
201             this.pTC.setRecording( false );
202             this.pTC.setRecordingFinished( true );
203             outputStream.close();
204         } catch( MalformedURLException e ) {
205             e.printStackTrace();
206             return false;
207         } catch( IOException e ) {
```

```
208             e.printStackTrace();
209             return false;
210         }
211
212         this.pFinished = true;
213         return true;
214     }
215
```

```
216     @Override
217     protected void onCancelled( Boolean aBoolean ) {
218         super.onCancelled( aBoolean );
219         this.pTC.setRecordingFinished( true );
220     }
221
```

```
222     @Override
223     protected void onCancelled() {
224         super.onCancelled();
225         this.pTC.setRecordingFinished( true );
226     }
227
228
```

```
229     public void stop() {
230         this.pStop = true;
231         this.pTC.setRecordingFinished( true );
232     }
233
```

```
234     public boolean isFinished() {
235         return this.pFinished;
236     }
237
238 }
239
240
241 }
242
```

6.3.5. IO.java

```
1  package com.grapsas.android.streamrecorder.misc;
2
3
4  import android.net.Uri;
5  import android.os.Build;
6  import android.support.annotation.NonNull;
7  import android.support.annotation.Nullable;
8
9  import com.grapsas.android.streamrecorder.exception.NeedActivityException;
10  import com.grapsas.android.streamrecorder.exception.NeedWorkingDirectoryException;
11
12  import java.io.File;
13  import java.io.FileDescriptor;
14  import java.util.Calendar;
15
16
17  public class IO {
```

```
18
19  public static int MIC_RECORDS = 1;
20  public static int STREAM_RECORDS = 2;
21  public static int ALL_RECORDS = 3;
22
23
```

```
24  /*
25   * General tools
26   */
27  @NonNull
28  public static String generateFileName() {
29      Calendar c = Calendar.getInstance();
30
31      //noinspection UnnecessaryLocalVariable
32      String fileName = ""
33              + c.get( Calendar.YEAR ) + "-"
34              + one2tow( c.get( Calendar.MONTH ) + 1 ) + "-"
35              + one2tow( c.get( Calendar.DAY_OF_MONTH ) ) + "."
36              + one2tow( c.get( Calendar.HOUR_OF_DAY ) ) + ":"
37              + one2tow( c.get( Calendar.MINUTE ) ) + ":"
38              + one2tow( c.get( Calendar.SECOND ) );
39
40      return fileName;
41  }
42
```

```
43  @NonNull
44  public static String one2tow( int number ) {
45      return (number < 10) ? "0" + number : number+"";
46  }
47
48
```

```
49  /*
50   * Filesystem tools
51   */
52  @Nullable
53  public static FileDescriptor createNewFile( @NonNull String prefix, @NonNull String suffix ) {
54      if( Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP )
55          return IOV21.createNewFile( prefix, suffix);
56      else
57          return IOV16.createNewFile( prefix, suffix );
58  }
59
```

```
60  public static boolean removeFile( Uri uri ) {
61      if( Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP )
62          return IOV21.removeFile( uri );
63      else
64          return IOV16.removeFile( new File( uri.getPath() ) );
65  }
66
67
```

```
68  /*
69   * Records tools
70   */
71  @NonNull
72  public static FileListItem[] getRecords_FLIArray( int type ) throws
73          NeedActivityException, NeedWorkingDirectoryException {
```

```
74      if( Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP ) {
75          //noinspection UnnecessaryLocalVariable
76          FileListItem[] files = IOV21.getRecords_FLIArray( type );
77          return files;
78      }
79      else {
80          return IOV16.getRecords_FLIArray( type );
81      }
82  }
83
```

```
84  @NonNull
85  public static FileListItem[] getRecords_FLIArray() throws
86          NeedActivityException, NeedWorkingDirectoryException {
87      return getRecords_FLIArray( ALL_RECORDS );
88  }
89
90 }
91
```

6.3.6. Misc.java

```
1  package com.grapsas.android.streamrecorder.misc;
2
3
4  import android.content.Context;
5  import android.support.annotation.NonNull;
6  import android.support.annotation.Nullable;
7  import android.text.format.DateUtils;
8
9  import java.io.IOException;
10  import java.io.InputStream;
11
12
13  public final class Misc {
14
15      private static final int MIN_IN_SEC = 60;
16      private static final int HOUR_IN_SEC = 3600 /* MIN_IN_SEC * 60 */;
17
18
```

```
19      public static String fromDuration( long ms ) {
20          MyLog.d( "ms: " + ms );
21          int duration = (int) ( 10000 / DateUtils.SECOND_IN_MILLIS);
22          if (duration < 0) {
23              duration = -duration;
24          }
25
26          int h = 0;
27          int m = 0;
28
29          if (duration >= HOUR_IN_SEC) {
30              h = duration / HOUR_IN_SEC;
31              duration -= h * HOUR_IN_SEC;
32          }
33          if (duration >= MIN_IN_SEC) {
34              m = duration / MIN_IN_SEC;
35              duration -= m * MIN_IN_SEC;
36          }
```

```
37        int s = duration;
38        MyLog.d( h + ":" + m + ":" + s );
39
40        return
41              ( h != 0 ? h + ":" : "") +
42              ( m != 0 ? m + ":" : "") +
43              s;
44    }
45
```

```
46    public static String humanBytes( long bytes, boolean si ) {
47        int unit = si ? 1000 : 1024;
48        if (bytes < unit) return bytes + " B";
49        int exp = (int) (Math.log(bytes) / Math.log(unit));
50        String pre = (si ? "kMGTPE" : "KMGTPE").charAt(exp-1) + (si ? "" : "i");
51        return String.format("%.1f %sB", bytes / Math.pow(unit, exp), pre);
52    }
53
54
```

```
55    public static @Nullable String loadJSONFromAsset(
56            @NonNull Context context, @NonNull String fileName ) {
57        String json = null;
58        try {
59            InputStream is = context.getAssets().open( fileName );
60            int size = is.available();
61            byte[] buffer = new byte[size];
62            is.read( buffer );
63            is.close();
64            json = new String( buffer, "UTF-8" );
65        } catch( IOException ex ) {
66            ex.printStackTrace();
67            return null;
68        }
69        return json;
70    }
71 }
72
```

6.3.7. IOV16.java

```
1  package com.grapsas.android.streamrecorder.misc;
2
3
4  import android.os.Environment;
5  import android.support.annotation.NonNull;
6  import android.support.annotation.Nullable;
7
8  import com.grapsas.android.streamrecorder.exception.IOException;
9
10  import java.io.File;
11  import java.io.FileDescriptor;
12  import java.io.FileNotFoundException;
13  import java.io.RandomAccessFile;
14  import java.util.ArrayList;
15  import java.util.Collections;
16
17
18  public class IOV16 {
```

```
19
20  private static final String WORKING_DIRECTORY_NAME = "StreamRecorder";
21
22
```

```
23  /*
24   * Filesystem tools
25   */
26  @NonNull
27  private static String getWorkingDirectoryPath() {
28      return Environment
29              .getExternalStorageDirectory()
30              .getAbsolutePath()
31              + "/" + WORKING_DIRECTORY_NAME + "/";
32  }
33
```

```
34  private static int checkDirectory( @NonNull File directory, boolean autoCreate )
35          throws IOException {
36      if( !directory.exists() ) {
37          if( !autoCreate )
38              throw new IOException( "Directory doesn't exist", 3 );
39          else if( !directory.mkdirs() ) {
40              throw new IOException( "Unable to create directories", 1 );
41          }
42          return 2;
43
44      }
45      if( !directory.isDirectory() )
46          throw new IOException( "Isn't directory", 2 );
47
48      return 1;
49  }
50
```

```
51  private static int checkDirectory( @NonNull File directory ) throws IOException {
52      return checkDirectory( directory, true );
53  }
54
```

```
55  public static int checkWorkingDirectory( boolean autoCreate ) throws IOException {
56      File wDir = new File( getWorkingDirectoryPath() );
57
58      return checkDirectory( wDir, autoCreate );
59  }
60
```

```
61  public static int checkWorkingDirectory() throws IOException {
62      return checkWorkingDirectory( true );
63  }
64
```

```
65  @NonNull
66  private static File[] getFileList( @NonNull String directoryPath ) throws IOException {
67      File dir = new File( directoryPath );
68      checkDirectory( dir, true );
69
70      return dir.listFiles();
71  }
72
```

```
73  @NonNull
74  private static File[] getFileList() throws IOException {
75      return getFileList( getWorkingDirectoryPath() );
76  }
77
```

```
78  @Nullable
79  public static FileDescriptor createNewFile( @NonNull String prefix, @NonNull String suffix ) {
80      File wDirFile = new File( getWorkingDirectoryPath() );
81      File newFile = new File( wDirFile, prefix + IO.generateFileName() + suffix );
82
83      try {
84          if( !newFile.createNewFile() )
85              return null;
86      } catch( java.io.IOException e ) {
87          e.printStackTrace();
88          return null;
89      }
90
91      RandomAccessFile raf;
92      try {
93          raf = new RandomAccessFile( newFile, "rw" );
94      } catch( FileNotFoundException e ) {
95          e.printStackTrace();
96          return null;
97      }
98
99      FileDescriptor fd;
100     try {
101         fd = raf.getFD();
102     } catch( java.io.IOException e ) {
103         e.printStackTrace();
104         return null;
105     }
106
107     return fd;
108 }
109
```

```
110 public static boolean removeFile( File file ) {
111     return file.delete();
112 }
113
114
```

```
115 /*
116  * Records tools
117  */
118 @NonNull
119 private static ArrayList< ComparableFiles > getRecords( int type ) throws IOException {
120     File[] filesList;
121     ArrayList< ComparableFiles > comparableFiles = new ArrayList<>();
122
123     String fileName;
124     filesList = getFileList( getWorkingDirectoryPath() );
125     for( int i = 0; i < filesList.length; i++ ) {
126         fileName = filesList[ i ].getName();
127         if(
128                 ( ( type & IO.MIC_RECORDS ) == IO.MIC_RECORDS &&
129                         fileName.substring( 0, 2 ).equals( "m." ) ) ||
130                 ( ( type & IO.STREAM_RECORDS ) == IO.STREAM_RECORDS &&
```

```
131                         fileName.substring( 0, 2 ).equals( "s." ) )
132             ) {
133             comparableFiles.add( new ComparableFiles( filesList[ i ] ) );
134           }
135       }
136
137       Collections.sort( comparableFiles );
138
139       return comparableFiles;
140   }
141
```

```
142   @NonNull
143   public static FileListItem[] getRecords_FLIArray( int type ) {
144       FileListItem[] fli;
145       ArrayList< ComparableFiles > recordsAL;
146       try {
147           recordsAL = getRecords( type );
148       } catch( IOException e ) {
149           e.printStackTrace();
150           return new FileListItem[ 0 ];
151       }
152
153       fli = new FileListItem[ recordsAL.size() ];
154       for( int i = 0; i < fli.length; i++ )
155           //noinspection UnnecessaryLocalVariable
156           fli[ i ] = new FileListItem( recordsAL.remove( 0 ).getFile() );
157
158       return fli;
159   }
160
161 }
162
```

6.3.8. IOV21.java

```
1   package com.grapsas.android.streamrecorder.misc;
2
3
4   import android.annotation.TargetApi;
5   import android.app.Activity;
6   import android.content.Intent;
7   import android.content.UriPermission;
8   import android.content.res.AssetFileDescriptor;
9   import android.net.Uri;
10  import android.os.Build;
11  import android.support.annotation.NonNull;
12  import android.support.annotation.Nullable;
13  import android.support.v4.provider.DocumentFile;
14
15  import com.grapsas.android.streamrecorder.App;
16  import com.grapsas.android.streamrecorder.exception.NeedActivityException;
17  import com.grapsas.android.streamrecorder.exception.NeedWorkingDirectoryException;
18
19  import java.io.FileDescriptor;
20  import java.io.FileNotFoundException;
21  import java.util.ArrayList;
22  import java.util.Collections;
23  import java.util.List;
```

```
24
25
26  @TargetApi( Build.VERSION_CODES.LOLLIPOP)
27  public class IOV21 {
28
29    public static final int DIR_PICKER_RESULT_CODE = 1;
30
31
32    /*
33     * Filesystem tools
34     */
35    public static void launchDirectoryPicker() {
36       Activity lastActivity = App.getInstance().getLastActivity();
37       if( lastActivity == null )
38          return;
39       Intent intent = new Intent( Intent.ACTION_OPEN_DOCUMENT_TREE );
40       lastActivity.startActivityForResult( intent, DIR_PICKER_RESULT_CODE );
41    }
42
43    public static void resultDirectoryPicker( @NonNull Uri treeUri ) {
44       Activity lastActivity = App.getInstance().getLastActivity();
45       if( lastActivity == null )
46          return;
47       lastActivity.getContentResolver().takePersistableUriPermission( treeUri,
48             Intent.FLAG_GRANT_READ_URI_PERMISSION | Intent.FLAG_GRANT_WRITE_URI_PERMISSION );
49    }
50
51    @NonNull
52    private static Uri getWorkingDirectoryUri() throws
53          NeedActivityException, NeedWorkingDirectoryException {
54       Activity lastActivity = App.getInstance().getLastActivity();
55       if( lastActivity == null )
56          throw new NeedActivityException();
57       List< UriPermission > UriPermissions = lastActivity.getContentResolver().getPersistedUriPermissions();
58       if( UriPermissions.size() == 0 )
59          throw new NeedWorkingDirectoryException();
60       else {
61          //noinspection UnnecessaryLocalVariable
62          Uri wDir = UriPermissions.get( 0 ).getUri();
63          return wDir;
64       }
65    }
66
67    @NonNull
68    private static DocumentFile[] getFiles( @NonNull Uri wDirUri ) throws
69          NeedWorkingDirectoryException, NeedActivityException {
70       Activity lastActivity = App.getInstance().getLastActivity();
71       if( lastActivity == null )
72          throw new NeedActivityException();
73
74       DocumentFile wDirDF = DocumentFile.fromTreeUri( lastActivity, wDirUri );
75
76       return wDirDF.listFiles();
77    }
78
79    @NonNull
```

```
80  private static DocumentFile[] getFiles() throws
81          NeedWorkingDirectoryException, NeedActivityException {
82      return getFiles( getWorkingDirectoryUri() );
83  }
84
```

```
85  @Nullable
86  public static FileDescriptor createNewFile( @NonNull String prefix, @Nullable String suffix ) {
87      Activity lastActivity = App.getInstance().getLastActivity();
88      Uri wDirUri;
89      try {
90          wDirUri = getWorkingDirectoryUri();
91      } catch( NeedActivityException e ) {
92          e.printStackTrace();
93          return null;
94      } catch( NeedWorkingDirectoryException e ) {
95          e.printStackTrace();
96          return null;
97      }
98      if( lastActivity == null )
99          return null;
100     DocumentFile wDirDF = DocumentFile.fromTreeUri( lastActivity, wDirUri );
101
102     DocumentFile newFile = wDirDF.createFile( null, prefix + IO.generateFileName() + suffix );
103     try {
104         AssetFileDescriptor afd = lastActivity
105                 .getContentResolver()
106                 .openAssetFileDescriptor( newFile.getUri(), "w" );
107         if( afd == null )
108             throw new NullPointerException();
109         return afd.getFileDescriptor();
110     } catch( FileNotFoundException e ) {
111         e.printStackTrace();
112         return null;
113     } catch( NullPointerException e ) {
114         e.printStackTrace();
115         return null;
116     }
117 }
118
```

```
119 public static boolean removeFile( Uri uri ) {
120     Activity lastActivity = App.getInstance().getLastActivity();
121     if( lastActivity == null )
122         return false;
123     DocumentFile dFile = DocumentFile.fromSingleUri( lastActivity, uri );
124
125     return dFile.delete();
126 }
127
128
```

```
129 /*
130  * Records tools
131  */
132 @NonNull
133 private static DocumentFile[] getRecords_DF( int type) throws
134         NeedActivityException, NeedWorkingDirectoryException {
135     DocumentFile[] dFiles = getFiles();
136
137     ArrayList< ComparableDocumentFile > recordsAL = new ArrayList<>();
```

```
138         for( DocumentFile dFile : dFiles ) {
139             if(
140                 ( ( type & IO.MIC_RECORDS ) == IO.MIC_RECORDS &&
141                         dFile.getName().substring( 0, 2 ).equals( "m." ) ) ||
142                 ( ( type & IO.STREAM_RECORDS ) == IO.STREAM_RECORDS &&
143                         dFile.getName().substring( 0, 2 ).equals( "s." ) )
144                 ) {
145                 recordsAL.add( new ComparableDocumentFile( dFile ) );
146             }
147         }
148         Collections.sort( recordsAL );
149 
150         DocumentFile[] recordsArray = new DocumentFile[ recordsAL.size() ];
151         int numOfRecords = recordsAL.size();
152         for( int i = 0; i < numOfRecords; i++ )
153             recordsArray[ i ] = recordsAL.remove( recordsAL.size() - 1 ).getDFile();
154 
155         return recordsArray;
156     }
157 
```

```
158     @NonNull
159     public static FileListItem[] getRecords_FLIArray( int type ) throws
160             NeedWorkingDirectoryException, NeedActivityException {
161         DocumentFile[] dFiles = getRecords_DF( type );
162 
163         FileListItem[] fileListItems = new FileListItem[ dFiles.length ];
164         for( int i = 0; i < dFiles.length; i++ )
165             fileListItems[ i ] = new FileListItem( dFiles[ i ] );
166 
167         return fileListItems;
168     }
169 
170 }
171 
```

6.3.9. MyLog.java

```
1   package com.grapsas.android.streamrecorder.misc;
2   
3   
4   import android.support.annotation.Nullable;
5   import android.util.Log;
6   
7   public class MyLog {
8   
9       public static final String TAG = "Filippos";
10  
```

```
11      public static void v( @Nullable String message ) {
12          Log.v( TAG, message );
13      }
14  
```

```
15      public static void d( @Nullable String message ) {
16          Log.d( TAG, message );
17      }
18  
```

```
19      public static void i( @Nullable String message ) {
20          Log.i( TAG, message );
21      }
22
23      public static void w( @Nullable String message ) {
24          Log.w( TAG, message );
25      }
26
27      public static void e( @Nullable String message ) {
28          Log.e( TAG, message );
29      }
30
31      public static void wtf( @Nullable String message ) {
32          Log.wtf( TAG, message );
33      }
34
35      public static void t( @Nullable String message ) {
36          Log.v( TAG + ":Toast", message);
37      }
38  }
39
```

6.3.10. FileListItem.java

```
1   package com.grapsas.android.streamrecorder.misc;
2
3
4   import android.content.Context;
5   import android.net.Uri;
6   import android.support.annotation.NonNull;
7   import android.support.v4.provider.DocumentFile;
8   import android.text.format.Formatter;
9
10  import java.io.File;
11  import java.sql.Date;
12  import java.text.SimpleDateFormat;
13  import java.util.ArrayDeque;
14
15
16  public class FileListItem {
17
18      private Uri mUri;
19      private String mName;
20      private long mModified;
21      private long mSize;
22
23      private static final SimpleDateFormat simpleDateFormat =
24              new SimpleDateFormat( "dd/MM/yy HH:mm:ss" );
25
26      public FileListItem( @NonNull File file ) {
27          this.mUri = Uri.fromFile( file );
28          this.mName = file.getName();
29          this.mModified = file.lastModified();
30          this.mSize = file.length();
```

```
31  }
32
33  public FileListItem( @NonNull DocumentFile dFile ) {
34      this.mUri = dFile.getUri();
35      this.mName = dFile.getName();
36      this.mModified = dFile.lastModified();
37      this.mSize = dFile.length();
38  }
39
40  public void setData( @NonNull Uri uri, @NonNull String name, long modified, long size ) {
41      this.mUri = uri;
42      this.mName = name;
43      this.mModified = modified;
44      this.mSize = size;
45  }
46
47  @NonNull
48  public Uri getUri() {
49      return this.mUri;
50  }
51
52  @NonNull
53  public String getName() {
54      return this.mName;
55  }
56
57  public long getModified() {
58      return this.mModified;
59  }
60
61  @NonNull
62  public String getModifiedHuman() {
63      return simpleDateFormat.format( new Date( this.getModified() ) );
64  }
65
66  public long getSize() {
67      return this.mSize;
68  }
69
70  @NonNull
71  public String getSizeHuman( Context context ) {
72      return Formatter.formatShortFileSize( context, this.getSize() );
73  }
74
75
76  /*
77   * Tools
78   */
79  @NonNull
80  public static ArrayDeque< FileListItem > getFilesDeque(
81          @NonNull ComparableFiles[] comparableFiles ) {
82      ArrayDeque< FileListItem > fileListItems = new ArrayDeque<>();
```

```
83
84          for( int i = 0; i < comparableFiles.length; i++ )
85              fileListItems.addLast( new FileListItem( comparableFiles[ i ].getFile() ) );
86
87          return fileListItems;
88      }
89
```

```
90      @NonNull
91      public static FileListItem[] getFilesArray( @NonNull ComparableFiles[] comparableFiles ) {
92          FileListItem[] fileListItems = new FileListItem[ comparableFiles.length ];
93
94          for( int i = 0; i < comparableFiles.length; i++ )
95              fileListItems[ i ] = new FileListItem( comparableFiles[i ].getFile() );
96
97          return fileListItems;
98      }
99
```

```
100     @NonNull
101     public static FileListItem[] getFilesArray( @NonNull ComparableDocumentFile[] comparableDFile ) {
102         FileListItem[] fileListItems = new FileListItem[ comparableDFile.length ];
103
104         for( int i = 0; i < comparableDFile.length; i++ )
105             fileListItems[ i ] = new FileListItem( comparableDFile[i ].getDFile() );
106
107         return fileListItems;
108     }
109 }
110
```

6.3.11. FavoritesURLs.java

```
1   package com.grapsas.android.streamrecorder.misc;
2
3
4   import android.annotation.TargetApi;
5   import android.content.Context;
6   import android.content.SharedPreferences;
7   import android.os.Build;
8   import android.support.annotation.NonNull;
9
10  import org.json.JSONArray;
11  import org.json.JSONException;
12
13
14  public class FavoritesURLs {
15
16      public static final String PREFERENCES_FILE_NAME = "favoritesURLs";
17      public static final String URLS_KEY = "urls";
18
19      private static SharedPreferences sharedPreferences;
20
21
```

```
22      @NonNull
23      public static synchronized SharedPreferences getSharedPreferences( @NonNull Context context ) {
24          if( sharedPreferences == null )
25              sharedPreferences = context.getSharedPreferences(
```

```
26              PREFERENCES_FILE_NAME, Context.MODE_PRIVATE );
27      return sharedPreferences;
28  }
29
```

```
30  public static boolean setUrls( @NonNull Context context, @NonNull JSONArray jsonArray ) {
31      SharedPreferences.Editor editor = getSharedPreferences( context ).edit();
32      editor.putString( URLS_KEY, jsonArray.toString() );
33      editor.apply();
34
35      return true;
36  }
37
```

```
38  @NonNull
39  public static JSONArray getUrls( @NonNull Context context ) {
40      String jString = getSharedPreferences( context ).getString( URLS_KEY, "[]" );
41      JSONArray jsonArray = new JSONArray();
42      try {
43          jsonArray = new JSONArray( jString );
44      } catch( JSONException e ) {
45          e.printStackTrace();
46      }
47
48      return jsonArray;
49  }
50
```

```
51  @NonNull
52  private static JSONArray removeURLv16( @NonNull JSONArray jsonArray, int position ) {
53      JSONArray newJArray = new JSONArray();
54
55      int len = jsonArray.length();
56      try {
57          for( int i = 0; i < len; i++ )
58              if( i != position )
59                  newJArray.put( jsonArray.getString( i ) );
60      } catch( JSONException e ) {
61          e.printStackTrace();
62      }
63
64      return newJArray;
65  }
66
```

```
67  @TargetApi( Build.VERSION_CODES.KITKAT )
68  @NonNull
69  private static JSONArray removeURLv19( @NonNull JSONArray jsonArray, int position ) {
70      jsonArray.remove( position );
71      return jsonArray;
72  }
73
```

```
74  public static boolean removeURL( Context context, int position ) {
75      JSONArray jsonArray = getUrls( context );
76      JSONArray newArray;
77
78      if( Build.VERSION.SDK_INT >= Build.VERSION_CODES.KITKAT )
79          newArray = removeURLv19( jsonArray, position );
80      else
81          newArray = removeURLv16( jsonArray, position );
```

```
82
83          setUrls( context, newArray );
84
85          return true;
86      }
87  }
88
```

6.3.12. ComparableFiles.java

```
1   package com.grapsas.android.streamrecorder.misc;
2
3   import android.support.annotation.NonNull;
4
5   import java.io.File;
6
7   public class ComparableFiles implements Comparable< ComparableFiles > {
8
9       private File mFile;
10
```

```
11      public ComparableFiles( File file ) {
12          this.mFile = file;
13      }
14
```

```
15      @NonNull
16      public File getFile() {
17          return this.mFile;
18      }
19
```

```
20      @Override
21      public int compareTo( ComparableFiles another ) {
22          if( this.mFile.lastModified() == another.getFile().lastModified() )
23              return 0;
24
25          return ( this.mFile.lastModified() > another.getFile().lastModified() ) ? -1 : 1;
26      }
27
28  }
29
```

6.3.13. MediaPlayerView.java

```
1   package com.grapsas.android.streamrecorder.misc;
2
3
4   import android.media.AudioManager;
5   import android.media.MediaPlayer;
6   import android.net.Uri;
7   import android.support.annotation.NonNull;
8   import android.support.annotation.Nullable;
9   import android.support.v7.app.AppCompatActivity;
10  import android.view.View;
11  import android.view.ViewStub;
```

```
12 import android.widget.ImageButton;
13 import android.widget.TextView;
14
15 import com.grapsas.android.streamrecorder.R;
16
17 import java.io.IOException;
18 import java.lang.ref.WeakReference;
19
20
21 public class MediaPlayerView implements MediaPlayer.OnCompletionListener {
22
23   private int mStubResourceId;
24   private WeakReference< AppCompatActivity > weakActivity;
25
26   private MediaPlayer player;
27
28   private View playingView;
29   private TextView durationT;
30   private ImageButton playPauseButton;
31
32   private FileListItem fli;
33
34
```

```
35   public MediaPlayerView( AppCompatActivity activity, int stubResourceId ) {
36     this.weakActivity = new WeakReference<>( activity );
37     this.mStubResourceId = stubResourceId;
38   }
39
```

```
40   @Nullable
41   private AppCompatActivity getActivity() {
42     return weakActivity.get();
43   }
44
45
```

```
46   private void showPlayingView( @NonNull String fileName,
47                                @NonNull String fileSize, @NonNull String duration ) {
48     AppCompatActivity activity = getActivity();
49     if( activity == null )
50       return;
51     // Use recordingView for first time.
52     if( this.playingView == null ) {
53       this.playingView = ( ( ViewStub ) activity.findViewById( this.mStubResourceId ) ).inflate();
54       this.playPauseButton = (ImageButton ) this.playingView.findViewById( R.id.playPause );
55       this.durationT = (TextView ) this.playingView.findViewById( R.id.duration );
56       this.playingView.findViewById( R.id.close ).setOnClickListener( new View.OnClickListener() {
57         @Override
58         public void onClick( View v ) {
59           stopPlaying();
60         }
61       } );
62       this.playPauseButton.setOnClickListener( new View.OnClickListener() {
63         @Override
64         public void onClick( View v ) {
65           playPausePlaying();
66         }
67       } );
68     }
69     // Reuse recordingView
```

```
70      else
71          this.playingView.setVisibility( View.VISIBLE );
72
73      TextView fileNameV = (TextView) this.playingView.findViewById( R.id.fileName );
74      TextView fileSizeV= (TextView) this.playingView.findViewById( R.id.fileSize );
75
76      fileNameV.setText( fileName );
77      fileSizeV.setText( fileSize );
78      this.durationT.setText( duration );
79
80      this.triggerOnPlayerViewShow();
81  }
82
```

```
83  private void hidePlayingView() {
84      if( this.playingView == null )
85          return;
86      this.playingView.setVisibility( View.GONE );
87
88      this.triggerOnPlayerViewHide();
89  }
90
```

```
91  public void startPlayingOnlyGUI( @NonNull FileListItem fileListItem ) {
92      try {
93          this.startPlaying( fileListItem, false );
94      } catch( com.grapsas.android.streamrecorder.exception.IOException e ) {
95          e.printStackTrace();
96      }
97  }
98
99
```

```
100  /*
101   * Media Player
102   */
103  public void startPlaying( @NonNull FileListItem fileListItem, boolean flagPlay )
104          throws com.grapsas.android.streamrecorder.exception.IOException {
105      this.fli = fileListItem;
106      AppCompatActivity activity = getActivity();
107      if( activity == null )
108          return;
109      this.stopPlaying( true, false );
110
111      Uri fileUri = fileListItem.getUri();
112      this.player = new MediaPlayer();
113      this.player.setOnCompletionListener( this );
114      this.player.setAudioStreamType( AudioManager.STREAM_MUSIC );
115      try {
116          this.player.setDataSource( activity, fileUri );
117      } catch( IOException e ) {
118  //          e.printStackTrace();
119          com.grapsas.android.streamrecorder.exception.IOException e2 =
120                  new com.grapsas.android.streamrecorder.exception.IOException(
121                          "Unable to MediaPlayer.setDataSource()", 1, e );
122          throw e2;
123      }
124      try {
125          this.player.prepare();
126      } catch( IOException e ) {
127  //          e.printStackTrace();
```

```
128         com.grapsas.android.streamrecorder.exception.IOException e2 =
129               new com.grapsas.android.streamrecorder.exception.IOException(
130                     "Unable to MediaPlayer.prepare()", 2, e );
131         throw e2;
132
133       }
134       int duration = this.player.getDuration();
135       duration /= 1000;
136       String durationS = String.format(
137             "%d:%02d:%02d", duration / 3600, ( duration % 3600 ) / 60, ( duration % 60 ) );
138       this.showPlayingView(
139             fileListItem.getName(), fileListItem.getSizeHuman( activity ), durationS );
140       if( flagPlay ) {
141         this.playPausePlaying();
142       }
143       else {
144         this.triggerOnStartPlaying();
145         this.playPauseButton.setImageResource( R.drawable.ic_play_arrow_black_24dp );
146       }
147     }
148
```

```
149   public void startPlaying( @NonNull FileListItem fileListItem )
150         throws com.grapsas.android.streamrecorder.exception.IOException {
151     this.startPlaying( fileListItem, true );
152   }
153
```

```
154   public void playPausePlaying() {
155     if( this.player == null )
156       return;
157
158     if( this.player.isPlaying() ) {
159       this.player.pause();
160       this.playPauseButton.setImageResource( R.drawable.ic_play_arrow_black_24dp );
161     }
162     else {
163       this.player.start();
164       this.playPauseButton.setImageResource( R.drawable.ic_pause_black_24dp );
165     }
166
167     this.triggerOnStartPlaying();
168   }
169
```

```
170   private void stopPlaying( boolean player, boolean gui ) {
171     if( gui )
172       this.hidePlayingView();
173
174     if( !player || this.player == null )
175       return;
176     this.player.stop();
177     this.player.release();
178     this.player = null;
179
180     this.triggerOnStopPlaying();
181   }
182
```

```
183   public void stopPlaying() {
184     this.stopPlaying( true, true );
```

```
185     }
186
187
188     /*
189      * Implements MediaPlayer.OnCompletionListener
190      */
191     @Override
192     public void onCompletion( MediaPlayer mp ) {
193         this.playPauseButton.setImageResource( R.drawable.ic_play_arrow_black_24dp );
194     }
195
196
197     /*
198      * Public interface
199      */
200     public interface Events {
201         void onStartPlaying();
202         void onStopPlaying();
203         void onPlayerViewShow( FileListItem fileListItem );
204         void onPlayerViewHide();
205     }
206
207     private void triggerOnStartPlaying() {
208         AppCompatActivity activity = getActivity();
209         if( activity == null || !( activity instanceof Events ) )
210             return;
211         ( (Events) activity ).onStartPlaying();
212     }
213
214     private void triggerOnStopPlaying() {
215         AppCompatActivity activity = getActivity();
216         if( activity == null || !( activity instanceof Events ) )
217             return;
218         ( (Events) activity ).onStopPlaying();
219     }
220
221     private void triggerOnPlayerViewShow() {
222         AppCompatActivity activity = getActivity();
223         if( activity == null || !( activity instanceof Events ) )
224             return;
225         ( (Events) activity ).onPlayerViewShow( this.fli );
226     }
227
228     private void triggerOnPlayerViewHide() {
229         this.fli = null;
230         AppCompatActivity activity = getActivity();
231         if( activity == null || !( activity instanceof Events ) )
232             return;
233         ( (Events) activity ).onPlayerViewHide();
234     }
235
236 }
237
```

6.3.14. MediaRecorderView.java

```
1  package com.grapsas.android.streamrecorder.misc;
2
3
4  import android.os.SystemClock;
5  import android.support.annotation.Nullable;
6  import android.support.v7.app.AppCompatActivity;
7  import android.view.View;
8  import android.view.ViewStub;
9  import android.widget.Chronometer;
10
11 import com.grapsas.android.streamrecorder.R;
12 import com.grapsas.android.streamrecorder.activities.MainActivity;
13 import com.grapsas.android.streamrecorder.misc.media.MicRecorder;
14 import com.grapsas.android.streamrecorder.misc.media.Recorder;
15 import com.grapsas.android.streamrecorder.misc.media.StreamRecorder;
16
17 import java.io.FileDescriptor;
18 import java.lang.ref.WeakReference;
19
20
21 public class MediaRecorderView {
22
23    public static final int MIC_RECORDER = 1;
24    public static final int STREAM_RECORDER = 2;
25
26
27    private int mStubResourceId;
28    private WeakReference< AppCompatActivity > weakActivity;
29
30    private Recorder recorder;
31
32    private View recordingView;
33    private Chronometer chronometer;
34
35
-----------------------------------------------------------------------------
36    public MediaRecorderView( AppCompatActivity activity, int stubResourceId ) {
37       this.weakActivity = new WeakReference<>( activity );
38       this.mStubResourceId = stubResourceId;
39    }
40
-----------------------------------------------------------------------------
41    @Nullable
42    private AppCompatActivity getActivity() {
43       return weakActivity.get();
44    }
45
46
-----------------------------------------------------------------------------
47    private boolean createRecordingView() {
48       AppCompatActivity activity = this.getActivity();
49       if( activity == null )
50          return false;
51       if( this.recordingView != null )
52          return false;
53       this.recordingView = ( ( ViewStub ) activity.findViewById( this.mStubResourceId ) ).inflate();
54       this.recordingView.findViewById( R.id.stop ).setOnClickListener( new View.OnClickListener() {
```

```
55          @Override
56          public void onClick( View v ) {
57              stopRecording();
58          }
59      } );
60      this.chronometer = ( Chronometer ) activity.findViewById( R.id.chronometer );
61      return true;
62  }
63
```

```
64  private void showRecordingView() {
65      if( !createRecordingView() )
66          this.recordingView.setVisibility( View.VISIBLE );
67  }
68
```

```
69  private void hideRecordingView() {
70      this.recordingView.setVisibility( View.GONE );
71  }
72
73
```

```
74  /*
75   * Media Recorder
76   */
77  public void startRecording( int type ) {
78      AppCompatActivity activity = this.getActivity();
79      if( activity == null )
80          return;
81
82      if( this.recorder != null ) {
83          this.stopRecording();
84          return;
85      }
86
87      FileDescriptor fd;
88      String prefix;
89      String suffix;
90      if( type == MIC_RECORDER ) {
91          prefix = "m.";
92          suffix = ".3gp";
93          this.recorder = new MicRecorder();
94      }
95      else if( type == STREAM_RECORDER ) {
96          String url = ( (MainActivity) getActivity() ).getUrl4Rec();
97          if( url == null )
98              return;
99          prefix = "s.";
100         suffix = "";
101         createRecordingView();
102         this.recorder = new StreamRecorder( url, this.chronometer );
103     }
104     else
105         return;
106
107     fd = IO.createNewFile( prefix, suffix );
108     if( fd == null )
109         return;
110
111     if( !this.recorder.startRecording( fd ) ) {
112         this.stopRecording();
```

```
113             return;
114         }
115 
116         this.showRecordingView();
117         this.chronometer.setBase( SystemClock.elapsedRealtime() );
118         this.chronometer.start();
119         this.triggerStartRecording();
120     }
121 
122 
```

```
123     public void stopRecording() {
124         if( this.recorder == null )
125             return;
126         this.recorder.stopRecording();
127         this.recorder = null;
128 
129         this.chronometer.stop();
130         this.hideRecordingView();
131         this.triggerStopRecording();
132     }
133 
134 
```

```
135     /*
136      * Public interface
137      */
138     public interface Events {
139         void onStartRecording();
140         void onStopRecording();
141     }
142 
```

```
143     private void triggerStartRecording() {
144         AppCompatActivity activity = getActivity();
145         if( activity == null || !( activity instanceof Events ) )
146             return;
147         ( (Events) activity).onStartRecording();
148     }
149 
```

```
150     private void triggerStopRecording() {
151         AppCompatActivity activity = getActivity();
152         if( activity == null || !( activity instanceof Events ) )
153             return;
154         ( (Events) activity).onStopRecording();
155     }
156 
157 }
158 
```

6.3.15. ViewPagerListener.java

```
1   package com.grapsas.android.streamrecorder.misc;
2 
3   import android.support.annotation.Nullable;
4   import android.support.v4.view.ViewPager;
5 
6   import com.grapsas.android.streamrecorder.interfaces.OnPageChangeListener;
```

```
7
8  import java.lang.ref.WeakReference;
9
10
11 public class ViewPagerListener implements ViewPager.OnPageChangeListener {
12
13    private WeakReference< OnPageChangeListener > pListener;
14    private boolean moving;
15
16
-----------------------------------------------------------------------------
17    public ViewPagerListener( @Nullable OnPageChangeListener listener ) {
18       this.moving = false;
19       if( listener != null )
20          this.pListener = new WeakReference<>( listener );
21    }
22
-----------------------------------------------------------------------------
23    @Override
24    public void onPageScrolled( int position, float positionOffset, int positionOffsetPixels ) {
25    }
26
-----------------------------------------------------------------------------
27    @Override
28    public void onPageSelected( int position ) {
29       this.triggerStopMoving();
30    }
31
-----------------------------------------------------------------------------
32    @Override
33    public void onPageScrollStateChanged( int state ) {
34       String sState = "error";
35          if( state == ViewPager.SCROLL_STATE_IDLE )
36             sState = "SCROLL_STATE_IDLE";
37          else if( state == ViewPager.SCROLL_STATE_DRAGGING ) {
38             sState = "SCROLL_STATE_DRAGGING";
39             this.triggerStartMoving();
40          }
41          else if( state == ViewPager.SCROLL_STATE_SETTLING ) {
42             sState = "SCROLL_STATE_SETTLING";
43          }
44
45 //       MyLog.d( "ViewPagerListener.onPageScrollStateChanged( " + sState + " )" );
46
47       if( state != ViewPager.SCROLL_STATE_IDLE )
48          this.triggerStartMoving();
49       else
50          this.triggerStopMoving();
51    }
52
-----------------------------------------------------------------------------
53    private void triggerStartMoving() {
54       if( this.moving )
55          return;
56
57 //       MyLog.d( "Start Moving" );
58       this.moving = true;
59       if( this.pListener != null && this.pListener.get() != null )
60          this.pListener.get().pagerStartMoving();
61    }
62
```

```
63      private void triggerStopMoving() {
64          if( !this.moving )
65              return;
66
67  //          MyLog.d( "Stop Moving" );
68          this.moving = false;
69          if( this.pListener != null && this.pListener.get() != null )
70              this.pListener.get().pagerFinishMoving();
71      }
72
73
74  }
75
```

6.3.16. ComparableDocumentFile.java

```
1   package com.grapsas.android.streamrecorder.misc;
2
3
4   import android.support.annotation.Nullable;
5   import android.support.v4.provider.DocumentFile;
6
7
8   public class ComparableDocumentFile implements Comparable< ComparableDocumentFile > {
9
10      private DocumentFile mDFile;
11
12      public ComparableDocumentFile( DocumentFile dFile ) {
13          this.mDFile = dFile;
14      }
15
16      public long getLastModified() {
17          return mDFile.lastModified();
18      }
19
20      public DocumentFile getDFile() {
21          return this.mDFile;
22      }
23
24
25      /*
26       * Implements Comparable
27       */
28      @Override
29      public int compareTo( @Nullable ComparableDocumentFile another ) {
30          if( another == null )
31              return -1;
32          if( this.getLastModified() > another.getLastModified() )
33              return 1;
34          else if( this.getLastModified() < another.getLastModified() )
35              return -1;
36          else
37              return 0;
38      }
```

```
39 }
40
```

6.3.17. DeleteFile.java

```
1  package com.grapsas.android.streamrecorder.dialogs;
2
3
4  import android.app.Activity;
5  import android.support.v7.app.AlertDialog;
6  import android.app.Dialog;
7  import android.app.DialogFragment;
8  import android.content.DialogInterface;
9  import android.net.Uri;
10  import android.os.Bundle;
11  import android.support.annotation.NonNull;
12
13  import com.grapsas.android.streamrecorder.R;
14
15
16  public class DeleteFile extends DialogFragment {
17
18    private Uri pUri;
19
20
```

```
21    public static DeleteFile newInstance( Uri uri ) {
22       DeleteFile fragment = new DeleteFile();
23       Bundle bundle = new Bundle();
24       bundle.putString( "uri", uri.toString() );
25       fragment.setArguments( bundle );
26       return fragment;
27    }
28
29
```

```
30    @Override
31    public void onCreate( Bundle savedInstanceState ) {
32       super.onCreate( savedInstanceState );
33
34       Bundle bundle = getArguments();
35       this.pUri = Uri.parse( bundle.getString( "uri" ) );
36    }
37
```

```
38    @Override
39    public Dialog onCreateDialog( Bundle savedInstanceState ) {
40       AlertDialog.Builder builder = new AlertDialog.Builder( getActivity() );
41
42       builder.setMessage( getString( R.string.RemoveFilePermanently ) + "\n" + pUri.getPath() );
43       builder.setPositiveButton( R.string.Cancel, new DialogInterface.OnClickListener() {
44          @Override
45          public void onClick( DialogInterface dialog, int which ) {
46             Activity activity = getActivity();
47             if( activity != null && activity instanceof Response )
48                ( ( Response ) activity ).deleteResponse( false, pUri );
49          }
50       } );
51       builder.setNegativeButton( R.string.Remove, new DialogInterface.OnClickListener() {
```

```
52          @Override
53          public void onClick( DialogInterface dialog, int which ) {
54              Activity activity = getActivity();
55              if( activity != null && activity instanceof Response )
56                  ( ( Response ) activity ).deleteResponse( true, pUri );
57          }
58      } );
59
60      return builder.create();
61  }
62
63
```

```
64  public interface Response {
65      void deleteResponse( boolean delete, @NonNull Uri file );
66  }
67 }
68
```

6.3.18. AddFavoriteURL.java

```
1  package com.grapsas.android.streamrecorder.dialogs;
2
3
4  import android.content.ClipData;
5  import android.content.ClipDescription;
6  import android.content.ClipboardManager;
7  import android.content.Context;
8  import android.support.annotation.NonNull;
9  import android.support.v7.app.AlertDialog;
10  import android.app.Dialog;
11  import android.app.DialogFragment;
12  import android.content.DialogInterface;
13  import android.os.Bundle;
14  import android.view.LayoutInflater;
15  import android.view.View;
16  import android.widget.EditText;
17  import android.widget.ImageButton;
18
19  import com.grapsas.android.streamrecorder.R;
20
21
22  public class AddFavoriteURL extends DialogFragment {
23
24      public enum TYPE { FAVORITE, JUST_REC }
25
```

```
26      private final static String TYPE_KEY = "type";
27
28      protected EditText urlTV;
29      private TYPE pType;
30
31
32      public AddFavoriteURL() {}
33
```

```
34      public static AddFavoriteURL newInstance( TYPE type ) {
35          Bundle args = new Bundle();
36          args.putSerializable( TYPE_KEY, type );
```

```
37
38      AddFavoriteURL fragment = new AddFavoriteURL();
39      fragment.setArguments( args );
40
41      return fragment;
42  }
43
44
```

```
45  @Override
46  public void onCreate( Bundle savedInstanceState ) {
47      super.onCreate( savedInstanceState );
48      this.pType = (TYPE) getArguments().getSerializable( TYPE_KEY );
49  }
50
```

```
51  @Override
52  public Dialog onCreateDialog( Bundle savedInstanceState ) {
53      AlertDialog.Builder builder = new AlertDialog.Builder( getActivity() );
54      LayoutInflater inflater = getActivity().getLayoutInflater();
55      View contentView = inflater.inflate( R.layout.dialog_content_add_favorite_url, null );
56      this.urlTV = (EditText) contentView.findViewById( R.id.editText );
57      ImageButton clearB = (ImageButton) contentView.findViewById( R.id.clear );
58
59      clearB.setOnClickListener( new View.OnClickListener() {
60          @Override
61          public void onClick( View v ) {
62              urlTV.setText( "" );
63          }
64      } );
65
66      ClipboardManager clipboard = (ClipboardManager ) getActivity()
67              .getSystemService( Context.CLIPBOARD_SERVICE );
68      if(
69              clipboard.hasPrimaryClip() &&
70              clipboard.getPrimaryClipDescription()
71                      .hasMimeType( ClipDescription.MIMETYPE_TEXT_PLAIN ) ) {
72          ClipData.Item cItem = clipboard.getPrimaryClip().getItemAt(0);
73          this.urlTV.setText( cItem.getText() );
74      }
75
76      builder.setView( contentView );
77      this.initButtons( builder );
78
79      return builder.create();
80  }
81
82
```

```
83  /*
84   * GUI Tools
85   */
86  private void initButtons(  AlertDialog.Builder builder ) {
87      if( this.pType == TYPE.FAVORITE ) {
88          builder.setNeutralButton( R.string.Save, new DialogInterface.OnClickListener() {
89              @Override
90              public void onClick( DialogInterface dialog, int which ) {
91                  AddFavoriteURL.Interaction listener = ( AddFavoriteURL.Interaction ) getActivity();
92                  listener.save( urlTV.getText().toString() );
93              }
94          } );
```

```
95            builder.setPositiveButton( R.string.Save_Rec, new DialogInterface.OnClickListener() {
96               @Override
97               public void onClick( DialogInterface dialog, int which ) {
98                  AddFavoriteURL.Interaction listener = ( AddFavoriteURL.Interaction ) getActivity();
99                  listener.saveAndRec( urlTV.getText().toString() );
100              }
101           } );
102        }
103        else
104           builder.setPositiveButton( R.string.Rec, new DialogInterface.OnClickListener() {
105              @Override
106              public void onClick( DialogInterface dialog, int which ) {
107                 AddFavoriteURL.Interaction listener = ( AddFavoriteURL.Interaction ) getActivity();
108                 listener.rec( urlTV.getText().toString() );
109              }
110           } );
111
112        builder.setNegativeButton( R.string.Cancel, null );
113     }
114
115
```

```
116     /*
117      * Interfaces
118      */
119     public interface Interaction {
120        void startAddFavoriteURL();
121        void saveAndRec( @NonNull String url );
122        void save( @NonNull String url );
123        void rec( @NonNull String url );
124     }
125  }
126
```

6.3.19. DeleteFavoriteURL.java

```
1   package com.grapsas.android.streamrecorder.dialogs;
2
3
4   import android.app.Activity;
5   import android.app.Dialog;
6   import android.app.DialogFragment;
7   import android.content.DialogInterface;
8   import android.os.Bundle;
9   import android.support.annotation.NonNull;
10  import android.support.v7.app.AlertDialog;
11
12  import com.grapsas.android.streamrecorder.R;
13
14
15  public class DeleteFavoriteURL extends DialogFragment {
16
17     public static final String POSITION_KEY = "position";
18     public static final String URL_KEY = "url";
19
20     private String pUrl;
21     private int pPosition;
22
23
```

```
24    public DeleteFavoriteURL() {}
25
```

```
26    public static DeleteFavoriteURL newInstance( int position, String url ) {
27        DeleteFavoriteURL fragment = new DeleteFavoriteURL();
28        Bundle bundle = new Bundle();
29        bundle.putInt( POSITION_KEY, position );
30        bundle.putString( URL_KEY, url );
31        fragment.setArguments( bundle );
32        return fragment;
33    }
34
35
```

```
36    @Override
37    public void onCreate( Bundle savedInstanceState ) {
38        super.onCreate( savedInstanceState );
39        Bundle bundle = getArguments();
40        this.pPosition = bundle.getInt( POSITION_KEY );
41        this.pUrl = bundle.getString( URL_KEY );
42    }
43
```

```
44    @NonNull
45    @Override
46    public Dialog onCreateDialog( Bundle savedInstanceState ) {
47        AlertDialog.Builder builder = new AlertDialog.Builder( getActivity() );
48
49        builder.setMessage( getString( R.string.RemoveURL ) + "\n" + this.pUrl );
50        builder.setPositiveButton( R.string.Cancel, null);
51        builder.setNegativeButton( R.string.Remove, new DialogInterface.OnClickListener() {
52            @Override
53            public void onClick( DialogInterface dialog, int which ) {
54                Activity activity = getActivity();
55                if( activity != null && activity instanceof Interaction )
56                    ( ( Interaction ) activity ).DeleteURL( pPosition );
57            }
58        } );
59
60        return builder.create();
61    }
62
```

```
63    public interface Interaction {
64        void startDeleteFavoriteURL( int position, @NonNull String url );
65        void DeleteURL( int position );
66    }
67  }
68
```

6.3.20. FavoritesURLsAdapter.java

```
1   package com.grapsas.android.streamrecorder.adapters;
2
3
4   import android.content.Context;
5   import android.support.annotation.NonNull;
6   import android.support.annotation.Nullable;
7   import android.view.LayoutInflater;
```

```
8  import android.view.View;
9  import android.view.ViewGroup;
10 import android.widget.BaseAdapter;
11 import android.widget.TextView;
12
13 import com.grapsas.android.streamrecorder.R;
14 import com.grapsas.android.streamrecorder.activities.FavoritesURLsActivity;
15 import com.grapsas.android.streamrecorder.misc.FavoritesURLs;
16 import com.grapsas.android.streamrecorder.misc.Misc;
17 import com.grapsas.android.streamrecorder.misc.media.MediaURL;
18
19 import org.json.JSONArray;
20 import org.json.JSONException;
21 import org.json.JSONObject;
22
23
24 public class FavoritesURLsAdapter extends BaseAdapter {
25
26    private Context pContext;
27    private String pType;
28    private JSONArray pJArray;
29
30
-----------------------------------------------------------------------------
31    public FavoritesURLsAdapter( @NonNull Context context ) {
32       this.init( context, null );
33    }
34
-----------------------------------------------------------------------------
35    public FavoritesURLsAdapter( @NonNull Context context, @NonNull String type ) {
36       this.init( context, type );
37    }
38
-----------------------------------------------------------------------------
39    private void init( @NonNull Context context, @Nullable String type ) {
40       this.pContext = context;
41       this.pJArray = new JSONArray();
42       if( type == null ) {
43          this.pType = FavoritesURLsActivity.TYPE_DEFAULT;
44       }
45       else {
46          this.pType = type;
47       }
48    }
49
50
-----------------------------------------------------------------------------
51    @NonNull
52    public JSONArray getData() {
53       return this.pJArray;
54    }
55
-----------------------------------------------------------------------------
56    @Override
57    public void notifyDataSetChanged() {
58       if( this.pType.equals( FavoritesURLsActivity.TYPE_FAV ) ) {
59          pJArray = FavoritesURLs.getUrls( this.pContext );
60       }
61       else if( this.pType.equals( FavoritesURLsActivity.TYPE_PRE_EXISTS ) ) {
62          String jString = Misc.loadJSONFromAsset(
63                this.pContext, FavoritesURLsActivity.PRE_EXISTS_FILE_NAME );
```

```
64          try {
65              pJArray = new JSONArray( jString );
66          } catch( JSONException e ) {
67              e.printStackTrace();
68          }
69      }
70      super.notifyDataSetChanged();
71  }
72
```

```
73  @Override
74  public int getCount() {
75      return this.pJArray.length();
76  }
77
```

```
78  @Override
79  public MediaURL getItem( int position ) {
80      if( this.pType.equals( FavoritesURLsActivity.TYPE_FAV ) ) {
81          try {
82              return new MediaURL( null, this.pJArray.getString( position ) );
83          } catch( JSONException e ) {
84              e.printStackTrace();
85          }
86      }
87      else if( this.pType.equals( FavoritesURLsActivity.TYPE_PRE_EXISTS ) ) {
88          try {
89              JSONObject jObject = this.pJArray.getJSONObject( position );
90              return new MediaURL( jObject.getString( "title" ), jObject.getString( "url" ) );
91          } catch( JSONException e ) {
92              e.printStackTrace();
93          }
94      }
95      return null;
96  }
97
```

```
98  @Override
99  public long getItemId( int position ) {
100     return position;
101 }
102
```

```
103 @Override
104 public View getView( int position, View convertView, ViewGroup parent ) {
105     if( convertView == null ) {
106         LayoutInflater inflater = (LayoutInflater) parent.getContext()
107                 .getSystemService( Context.LAYOUT_INFLATER_SERVICE );
108         convertView = inflater.inflate( R.layout.list_item_favarites_url, parent, false );
109     }
110
111     TextView titleTV = (TextView) convertView.findViewById( R.id.title );
112     TextView urlTV = (TextView) convertView.findViewById( R.id.url );
113
114     MediaURL mediaURL = this.getItem( position );
115     String title;
116     String url;
117
118     if( mediaURL.getTitle() == null ) {
119         url = mediaURL.getURL();
120         title = url;
```

```
121             urlTV.setVisibility( View.GONE );
122         }
123         else {
124             title = mediaURL.getTitle();
125             url = mediaURL.getURL();
126             urlTV.setVisibility( View.VISIBLE );
127         }
128 
129         titleTV.setText( title );
130         urlTV.setText( url );
131 
132         return convertView;
133     }
134 }
135 
```

6.3.21. Exception.java

```
1   package com.grapsas.android.streamrecorder.exception;
2   
3   
4   public class Exception extends java.lang.Exception {
5   
6       private int code;
7   
8   
```

```
9       public Exception() {
10          super();
11      }
12  
```

```
13      public Exception( String detailMessage ) {
14          super( detailMessage );
15      }
16  
```

```
17      public Exception( String detailMessage, int code ) {
18          super( detailMessage );
19          this.code = code;
20      }
21  
```

```
22      public Exception( String detailMessage, int code, Throwable cause ) {
23          super( detailMessage, cause);
24          this.code = code;
25      }
26  
```

```
27      public int getCode() {
28          return code;
29      }
30  }
31  
```

6.3.22. IOException.java

```
1  package com.grapsas.android.streamrecorder.exception;
2
3
4  public class IOException extends com.grapsas.android.streamrecorder.exception.Exception {
5
6    public IOException() {
7      super();
8    }
9
```

```
10   public IOException( String detailMessage, int code ) {
11     super( detailMessage, code );
12   }
13
```

```
14   public IOException( String detailMessage, int code, Throwable cause ) {
15     super( detailMessage, code, cause );
16   }
17
18 }
19
```

6.3.23. NeedActivityException.java

```
1  package com.grapsas.android.streamrecorder.exception;
2
3
4  public class NeedActivityException extends java.lang.Exception {
5  }
6
```

6.3.24. NeedWorkingDirectoryException.java

```
1  package com.grapsas.android.streamrecorder.exception;
2
3
4  public class NeedWorkingDirectoryException extends java.lang.Exception {
5  }
6
```

6.3.25. MicRecordsFragment.java

```
1  package com.grapsas.android.streamrecorder.fragments;
2
3
4  import android.app.Activity;
5  import android.os.Bundle;
6  import android.support.v4.app.Fragment;
7  import android.view.LayoutInflater;
8  import android.view.View;
```

```
9  import android.view.ViewGroup;
10 import android.widget.AdapterView;
11 import android.widget.ListView;
12
13 import com.grapsas.android.streamrecorder.R;
14 import com.grapsas.android.streamrecorder.adapters.RecordsListAdapter;
15 import com.grapsas.android.streamrecorder.interfaces.OnDataChanged;
16 import com.grapsas.android.streamrecorder.misc.FileListItem;
17 import com.grapsas.android.streamrecorder.misc.IO;
18
19
20 public class MicRecordsFragment extends Fragment implements OnDataChanged {
21
22    private OnFragmentInteractionListener mListener;
23
24    private ListView listView;
25    private RecordsListAdapter adapter;
26
27
-------------------------------------------------------------------------------
28    public static MicRecordsFragment newInstance() {
29       MicRecordsFragment fragment = new MicRecordsFragment();
30       Bundle args = new Bundle();
31       fragment.setArguments( args );
32       return fragment;
33    }
34
-------------------------------------------------------------------------------
35    public MicRecordsFragment() {
36       // Required empty public constructor
37    }
38
39
-------------------------------------------------------------------------------
40    @Override
41    public View onCreateView( LayoutInflater inflater, ViewGroup container,
42                              Bundle savedInstanceState ) {
43       View rootView = inflater.inflate( R.layout.fragment_mic_records, container, false );
44
45       this.adapter = new RecordsListAdapter( getActivity(), new FileListItem[ 0 ] );
46       this.listView = ( ListView) rootView.findViewById( R.id.listView );
47       this.listView.setAdapter( adapter );
48       this.listView = (ListView) rootView.findViewById( R.id.listView );
49       this.listView.setOnItemClickListener( new AdapterView.OnItemClickListener() {
50          @Override
51          public void onItemClick( AdapterView< ? > parent, View view, int position, long id ) {
52             if( mListener != null )
53                mListener.startPlaying( ( FileListItem ) adapter.getItem( position ) );
54          }
55       } );
56
57       return rootView;
58    }
59
60
-------------------------------------------------------------------------------
61    @Override
62    public void onAttach( Activity activity) {
63       super.onAttach( activity );
64       if( activity instanceof OnFragmentInteractionListener ) {
65          mListener = ( OnFragmentInteractionListener ) activity;
```

```
66      } else {
67          throw new RuntimeException( activity.toString()
68                  + " must implement OnFragmentInteractionListener" );
69      }
70  }
71
```

```
72  @Override
73  public void onDetach() {
74      super.onDetach();
75      mListener = null;
76  }
77
```

```
78  @Override
79  public void onResume() {
80      super.onResume();
81      this.refreshListView();
82  }
83
```

```
84  private void refreshListView() {
85      if( this.mListener == null )
86          return;
87      this.adapter.refreshData( mListener.getRecords( IO.MIC_RECORDS ) );
88      this.adapter.notifyDataSetChanged();
89  }
90
91
```

```
92  /*
93   * Implements OnDataChanged
94   */
95  @Override
96  public void dataChanged() {
97      this.refreshListView();
98  }
99
100
```

```
101  /*
102   *
103   */
104  public interface OnFragmentInteractionListener {
105
106      FileListItem[] getRecords( int type );
107      void startPlaying( FileListItem fileListItem );
108
109  }
110 }
111
```

6.3.26. StreamsRecordsFragment.java

```
1  package com.grapsas.android.streamrecorder.fragments;
2
3
4  import android.app.Activity;
5  import android.os.Bundle;
```

```
6  import android.support.v4.app.Fragment;
7  import android.view.LayoutInflater;
8  import android.view.View;
9  import android.view.ViewGroup;
10 import android.widget.AdapterView;
11 import android.widget.ListView;
12
13 import com.grapsas.android.streamrecorder.R;
14 import com.grapsas.android.streamrecorder.adapters.RecordsListAdapter;
15 import com.grapsas.android.streamrecorder.interfaces.OnDataChanged;
16 import com.grapsas.android.streamrecorder.misc.FileListItem;
17 import com.grapsas.android.streamrecorder.misc.IO;
18
19
20 public class StreamsRecordsFragment extends Fragment implements OnDataChanged {
21
22    private OnFragmentInteractionListener mListener;
23
24    private ListView listView;
25    private RecordsListAdapter adapter;
26
27
```

```
28    public static StreamsRecordsFragment newInstance( String param1, String param2 ) {
29       StreamsRecordsFragment fragment = new StreamsRecordsFragment();
30       Bundle args = new Bundle();
31       fragment.setArguments( args );
32       return fragment;
33    }
34
```

```
35    public StreamsRecordsFragment() {
36       // Required empty public constructor
37    }
38
39
```

```
40    @Override
41    public View onCreateView( LayoutInflater inflater, ViewGroup container,
42                              Bundle savedInstanceState ) {
43       View rootView = inflater.inflate( R.layout.fragment_mic_records, container, false );
44
45       this.adapter = new RecordsListAdapter( getActivity(), new FileListItem[ 0 ] );
46       this.listView = ( ListView ) rootView.findViewById( R.id.listView );
47       this.listView.setAdapter( adapter );
48       this.listView = (ListView) rootView.findViewById( R.id.listView );
49       this.listView.setOnItemClickListener( new AdapterView.OnItemClickListener() {
50          @Override
51          public void onItemClick( AdapterView< ? > parent, View view, int position, long id ) {
52             if( mListener != null )
53                mListener.startPlaying( ( FileListItem ) adapter.getItem( position ) );
54          }
55       } );
56
57       return rootView;
58    }
59
```

```
60    @Override
61    public void onAttach( Activity activity) {
62       super.onAttach( activity );
```

```
63          if( activity instanceof OnFragmentInteractionListener ) {
64              mListener = ( OnFragmentInteractionListener ) activity;
65          } else {
66              throw new RuntimeException( activity.toString()
67                      + " must implement OnFragmentInteractionListener" );
68          }
69      }
70
```

```
71      @Override
72      public void onDetach() {
73          super.onDetach();
74          mListener = null;
75      }
76
```

```
77      @Override
78      public void onResume() {
79          super.onResume();
80          this.refreshListView();
81      }
82
```

```
83      private void refreshListView() {
84          if( this.mListener == null )
85              return;
86          this.adapter.refreshData( mListener.getRecords( IO.STREAM_RECORDS ) );
87          this.adapter.notifyDataSetChanged();
88      }
89
90
```

```
91      /*
92       *
93       */
94      @Override
95      public void dataChanged() {
96          this.refreshListView();
97      }
98
```

```
99      /*
100      *
101      */
102     public interface OnFragmentInteractionListener {
103
104         FileListItem[] getRecords( int type );
105         void startPlaying( FileListItem fileListItem );
106
107     }
108 }
109
```

6.3.27. FavoritesURLsActivityFragment.java

```
1   package com.grapsas.android.streamrecorder.fragments;
2
3
4   import android.support.annotation.NonNull;
```

```
5  import android.support.v4.app.Fragment;
6  import android.os.Bundle;
7  import android.view.LayoutInflater;
8  import android.view.View;
9  import android.view.ViewGroup;
10  import android.widget.AdapterView;
11  import android.widget.ListView;
12
13  import com.github.clans.fab.FloatingActionButton;
14  import com.grapsas.android.streamrecorder.R;
15  import com.grapsas.android.streamrecorder.activities.FavoritesURLsActivity;
16  import com.grapsas.android.streamrecorder.adapters.FavoritesURLsAdapter;
17  import com.grapsas.android.streamrecorder.dialogs.AddFavoriteURL;
18  import com.grapsas.android.streamrecorder.dialogs.DeleteFavoriteURL;
19  import com.grapsas.android.streamrecorder.misc.FavoritesURLs;
20
21  import org.json.JSONArray;
22
23
24  public class FavoritesURLsActivityFragment extends Fragment implements
25      AddFavoriteURL.Interaction {
26
27
28    private AddFavoriteURL.Interaction pAddListener;
29    private DeleteFavoriteURL.Interaction pDeleteListener;
30
31    private FavoritesURLsAdapter adapter;
32
33
-----------------------------------------------------------------------------
34    public FavoritesURLsActivityFragment() {
35    }
36
37
-----------------------------------------------------------------------------
38    /*
39     * Fragment overrides
40     */
41    @Override
42    public View onCreateView( LayoutInflater inflater, ViewGroup container,
43                              Bundle savedInstanceState ) {
44      View rootView = inflater.inflate( R.layout.fragment_favorites_urls, container, false );
45
46      this.initListView( rootView );
47      this.initFAB( rootView );
48
49      return rootView;
50    }
51
-----------------------------------------------------------------------------
52    @Override
53    public void onResume() {
54      super.onResume();
55      if( getActivity() instanceof AddFavoriteURL.Interaction )
56        this.pAddListener = (AddFavoriteURL.Interaction) getActivity();
57      if( getActivity() instanceof DeleteFavoriteURL.Interaction )
58        this.pDeleteListener = (DeleteFavoriteURL.Interaction) getActivity();
59    }
60
-----------------------------------------------------------------------------
61    @Override
```

```
62    public void onPause() {
63        this.pAddListener = null;
64        this.pDeleteListener = null;
65        super.onPause();
66    }
67
```

```
68    /*
69     * GUI Tools
70     */
71    @NonNull
72    private String getType() {
73        String type = ( (FavoritesURLsActivity) getActivity() ).getType();
74        return type;
75    }
76
```

```
77    private void initListView( @NonNull View rootView ) {
78        adapter = new FavoritesURLsAdapter( this.getContext(), this.getType() );
79        ListView listView = (ListView) rootView.findViewById( R.id.listView );
80        listView.setAdapter( adapter );
81        if( this.getType().equals( FavoritesURLsActivity.TYPE_FAV ) ) {
82            listView.setOnItemLongClickListener( new AdapterView.OnItemLongClickListener() {
83                @Override
84                public boolean onItemLongClick( AdapterView< ? > parent, View view, int position, long id ) {
85                    if( pDeleteListener == null )
86                        return false;
87                    pDeleteListener.startDeleteFavoriteURL( position, adapter.getItem( position ).getURL() );
88                    return true;
89                }
90            } );
91        }
92        listView.setOnItemClickListener( new AdapterView.OnItemClickListener() {
93            @Override
94            public void onItemClick( AdapterView< ? > parent, View view, int position, long id ) {
95                FavoritesURLsActivity activity = (FavoritesURLsActivity) getActivity();
96                activity.urlSelected( adapter.getItem( position ).getURL() );
97            }
98        } );
99
100       adapter.notifyDataSetChanged();
101   }
102
```

```
103   private void initFAB( @NonNull View rootView ) {
104       FloatingActionButton fab = (FloatingActionButton) rootView.findViewById( R.id.fab );
105       if( this.getType().equals( FavoritesURLsActivity.TYPE_FAV ) ) {
106           fab.setOnClickListener( new View.OnClickListener() {
107               @Override
108               public void onClick( View v ) {
109                   if( pAddListener != null )
110                       pAddListener.startAddFavoriteURL();
111               }
112           } );
113           fab.setVisibility( View.VISIBLE );
114       }
115       else {
116           fab.setVisibility( View.GONE );
117       }
118   }
119
```

```
120
121     /*
122      * Implements AddFavoriteURL.Interaction
123      */
124     public void notifyDataSetChanged() {
125         adapter.notifyDataSetChanged();
126     }
127
128
129     /*
130      * Implements AddFavoriteURL.Interaction
131      */
132     @Override
133     public void startAddFavoriteURL() {
134     }
135
136     @Override
137     public void saveAndRec( @NonNull String url ) {
138     }
139
140     @Override
141     public void save( @NonNull String url ) {
142         JSONArray jsonArray = adapter.getData();
143         jsonArray.put( url );
144         FavoritesURLs.setUrls( getContext(), jsonArray );
145         adapter.notifyDataSetChanged();
146     }
147
148     @Override
149     public void rec( @NonNull String url ) {
150     }
151 }
152
```

6.3.28. MyActivity.java

```
1  package com.grapsas.android.streamrecorder.activities;
2
3
4  import android.os.Bundle;
5  import android.support.v7.app.AppCompatActivity;
6
7  import com.grapsas.android.streamrecorder.App;
8
9
10  public class MyActivity extends AppCompatActivity {
11
12      @Override
13      protected void onCreate( Bundle savedInstanceState ) {
14          super.onCreate( savedInstanceState );
15
16          this.getApplication().registerActivityLifecycleCallbacks( App.getInstance() );
17      }
```

```
18 }
19
```

6.3.29. MainActivity.java

```
1  package com.grapsas.android.streamrecorder.activities;
2
3
4  import android.content.Context;
5  import android.content.Intent;
6  import android.net.Uri;
7  import android.os.Build;
8  import android.os.Bundle;
9  import android.support.annotation.NonNull;
10  import android.support.annotation.Nullable;
11  import android.support.design.widget.Snackbar;
12  import android.support.v4.app.Fragment;
13  import android.support.v4.app.FragmentManager;
14  import android.support.v4.app.FragmentPagerAdapter;
15  import android.support.v4.view.ViewPager;
16  import android.support.v7.widget.Toolbar;
17  import android.view.Menu;
18  import android.view.MenuItem;
19  import android.view.View;
20  import android.view.ViewGroup;
21  import android.widget.RelativeLayout;
22
23  import com.github.clans.fab.FloatingActionButton;
24  import com.github.clans.fab.FloatingActionMenu;
25  import com.grapsas.android.lib.slidingtabs.slidingtabs.SlidingTabLayout;
26  import com.grapsas.android.streamrecorder.App;
27  import com.grapsas.android.streamrecorder.R;
28  import com.grapsas.android.streamrecorder.dialogs.AddFavoriteURL;
29  import com.grapsas.android.streamrecorder.dialogs.DeleteFile;
30  import com.grapsas.android.streamrecorder.exception.NeedActivityException;
31  import com.grapsas.android.streamrecorder.exception.NeedWorkingDirectoryException;
32  import com.grapsas.android.streamrecorder.fragments.MicRecordsFragment;
33  import com.grapsas.android.streamrecorder.fragments.StreamsRecordsFragment;
34  import com.grapsas.android.streamrecorder.interfaces.OnDataChanged;
35  import com.grapsas.android.streamrecorder.interfaces.OnPageChangeListener;
36  import com.grapsas.android.streamrecorder.misc.FileListItem;
37  import com.grapsas.android.streamrecorder.misc.IO;
38  import com.grapsas.android.streamrecorder.misc.IOV16;
39  import com.grapsas.android.streamrecorder.misc.IOV21;
40  import com.grapsas.android.streamrecorder.misc.MediaPlayerView;
41  import com.grapsas.android.streamrecorder.misc.MediaRecorderView;
42  import com.grapsas.android.streamrecorder.misc.MyLog;
43  import com.grapsas.android.streamrecorder.misc.ViewPagerListener;
44
45  import java.lang.ref.WeakReference;
46  import java.util.ArrayList;
47  import java.util.List;
48
49
50  public class MainActivity extends MyActivity implements
51          MicRecordsFragment.OnFragmentInteractionListener,
52          StreamsRecordsFragment.OnFragmentInteractionListener,
53          MediaRecorderView.Events,
54          MediaPlayerView.Events,
```

```
55      DeleteFile.Response,
56      OnPageChangeListener,
57      AddFavoriteURL.Interaction {
58
59  // TODO: Add elevation
60  private RelativeLayout recordingLayout;
61  private MediaRecorderView mediaRecorderView;
62  private MediaPlayerView mediaPlayerView;
63
64  private PagerAdapter pagerAdapter;
65  private ViewPager viewPager;
66  private Snackbar snackbar;
67  private Menu pMenu;
68
69  private FloatingActionButton micFab;
70  private FloatingActionMenu sFabMenu;
71  private FloatingActionButton favFab;
72  private FloatingActionButton preExistsFab;
73  private FloatingActionButton newFab;
74
75  private FileListItem pFli;
76  private String url4Rec;
77
78
```

```
79  /*
80   * Activity Overrides
81   */
82  @Override
83  protected void onCreate( Bundle savedInstanceState ) {
84      super.onCreate( savedInstanceState );
85      setContentView( R.layout.activity_main );
86
87      Toolbar toolbar = ( Toolbar ) findViewById( R.id.toolbar );
88      setSupportActionBar( toolbar );
89
90      FragmentManager fragmentManager = getSupportFragmentManager();
91      pagerAdapter = new PagerAdapter( fragmentManager, this );
92
93      viewPager = ( ViewPager ) findViewById( R.id.viewPager );
94      if( viewPager != null )
95          viewPager.setAdapter( pagerAdapter );
96
97      SlidingTabLayout slidingTabs = (SlidingTabLayout ) findViewById(R.id.tabs);
98      if( slidingTabs != null ) {
99          slidingTabs.setDistributeEvenly( true );
100          slidingTabs.setViewPager( viewPager );
101          slidingTabs.setOnPageChangeListener( new ViewPagerListener( this ) );
102      }
103
104      this.initFabs();
105
106      this.mediaRecorderView = new MediaRecorderView( this, R.id.stub_recording );
107      this.mediaPlayerView = new MediaPlayerView( this, R.id.stub_playing );
108      this.recordingLayout = (RelativeLayout) findViewById( R.id.recordingLayout );
109  }
110
```

```
111  @Override
112  public boolean onCreateOptionsMenu( Menu menu ) {
113      getMenuInflater().inflate( R.menu.activity_main_menu, menu );
```

```
114     this.pMenu = menu;
115     return super.onCreateOptionsMenu( menu );
116 }
117
```

```
118 @Override
119 public boolean onOptionsItemSelected( MenuItem item ) {
120     switch( item.getItemId() ) {
121         case R.id.remove:
122             DeleteFile.newInstance( pFli.getUri() ).show( getFragmentManager(), null );
123             return true;
124         default:
125             return super.onOptionsItemSelected( item );
126     }
127 }
128
```

```
129 @Override
130 protected void onResume() {
131     MyLog.d( "----------------" );
132     super.onResume();
133     pagerFinishMoving();
134     if( this.getUrl4Rec() != null )
135         this.startStreamRecording();
136 }
137
```

```
138 @Override
139 protected void onPause() {
140     this.stopMicRecording();
141     this.stopPlaying();
142     super.onPause();
143 }
144
```

```
145 @Override
146 protected void onActivityResult( int requestCode, int resultCode, Intent data ) {
147     if( resultCode != RESULT_OK )
148         return;
149     switch( requestCode ) {
150         case IOV21.DIR_PICKER_RESULT_CODE:
151             App.getInstance().setLastActivity( this );
152             IOV21.resultDirectoryPicker( data.getData() );
153             if( this.snackbar != null )
154                 this.snackbar.dismiss();
155             break;
156         case FavoritesURLsActivity.RESULT_CODE:
157             String url = data.getStringExtra( FavoritesURLsActivity.RESULT_URL_KEY );
158             this.setUrl4Rec( url );
159             break;
160     }
161 }
162
163
```

```
164 /*
165  * GUI tools.
166  */
167 private void showBottomLayout() {
168     ViewGroup.LayoutParams params = this.recordingLayout.getLayoutParams();
169     params.height = ViewGroup.LayoutParams.WRAP_CONTENT;
```

```
170     this.recordingLayout.setLayoutParams( params );
171 }
172
```

```
173 private void hideBottomLayout() {
174     ViewGroup.LayoutParams params = this.recordingLayout.getLayoutParams();
175     params.height = 0;
176     this.recordingLayout.setLayoutParams( params );
177 }
178
```

```
179 private void initFabs() {
180     this.micFab = ( FloatingActionButton ) findViewById( R.id.micFab );
181     this.sFabMenu = ( FloatingActionMenu ) findViewById( R.id.sFabMenu );
182     this.favFab = ( FloatingActionButton ) findViewById( R.id.favFab );
183     this.preExistsFab = ( FloatingActionButton ) findViewById( R.id.preExistsFab );
184     this.newFab = ( FloatingActionButton ) findViewById( R.id.newFab );
185
186     this.micFab.setOnClickListener( new View.OnClickListener() {
187         @Override
188         public void onClick( View v ) {
189             startMicRecording();
190         }
191     } );
192     this.favFab.setOnClickListener( new View.OnClickListener() {
193         @Override
194         public void onClick( View v ) {
195             startFavActivity( FavoritesURLsActivity.TYPE_FAV );
196         }
197     } );
198     this.preExistsFab.setOnClickListener( new View.OnClickListener() {
199         @Override
200         public void onClick( View v ) {
201             startFavActivity( FavoritesURLsActivity.TYPE_PRE_EXISTS );
202         }
203     });
204     this.newFab.setOnClickListener( new View.OnClickListener() {
205         @Override
206         public void onClick( View v ) {
207             startAddFavoriteURL();
208         }
209     } );
210 }
211
```

```
212 private void showFab( boolean animate ) {
213     this.hideFabs();
214     switch( this.viewPager.getCurrentItem() ) {
215         case 0:
216             this.micFab.show( animate );
217             break;
218         case 1:
219             this.sFabMenu.showMenu( animate );
220             break;
221     }
222 }
223
```

```
224 private void showFab() {
225     this.showFab( false );
226 }
```

```
227
228  private void hideFabs( boolean animate ) {
229      this.micFab.hide( animate );
230      this.sFabMenu.hideMenu( animate );
231  }
232
233  private void hideFabs() {
234      this.hideFabs( false );
235  }
236
237  protected void startFavActivity( @NonNull String type ) {
238      Intent intent = new Intent( this, FavoritesURLsActivity.class );
239      Bundle bundle = new Bundle();
240      bundle.putString( FavoritesURLsActivity.TYPE_KEY, type );
241      intent.putExtras( bundle );
242      startActivityForResult( intent, FavoritesURLsActivity.RESULT_CODE );
243  }
244
245  protected void startFavActivity() {
246      this.startFavActivity( FavoritesURLsActivity.TYPE_FAV );
247  }
248
249  public void showPlayingView( View v ) {
250      FileListItem fileListItem = (FileListItem) v.getTag();
251      this.showPlayingView( fileListItem );
252  }
253
254
255  /*
256   * Tools
257   */
258  private void filesystemChanged() {
259      this.pagerAdapter.refreshDataSets();
260  }
261
262  @NonNull
263  private FileListItem[] getRecordsV21( int type ) {
264      FileListItem[] records;
265
266      try {
267          records = IO.getRecords_FLIArray( type );
268      } catch( NeedActivityException e ) {
269          MyLog.e( "NeedActivityException" );
270          e.printStackTrace();
271          records = new FileListItem[ 0 ];
272          if( this.snackbar != null )
273              this.snackbar.dismiss();
274          this.snackbar = Snackbar
275                  .make( this.micFab, "Unable to access records.", Snackbar.LENGTH_INDEFINITE );
276          this.snackbar.setAction( "Retry", new View.OnClickListener() {
277                      @Override
278                      public void onClick( View v ) {
279                          filesystemChanged();
280                      }
```

```
281                 } );
282             this.snackbar.show();
283         } catch( NeedWorkingDirectoryException e ) {
284             MyLog.e( "NeedWorkingDirectoryException" );
285             e.printStackTrace();
286             records = new FileListItem[ 0 ];
287             if( this.snackbar != null )
288                 this.snackbar.dismiss();
289             this.snackbar = Snackbar.make(
290                     this.micFab,
291                     "Need read/write permissions to a directory for saving and play records.",
292                     Snackbar.LENGTH_INDEFINITE );
293             this.snackbar.setAction( "Select", new View.OnClickListener() {
294                         @Override
295                         public void onClick( View v ) {
296                             IOV21.launchDirectoryPicker();
297                         }
298                     } );
299             this.snackbar.show();
300         }
301
302         return records;
303     }
304
```

```
305     @Override
306     public FileListItem[] getRecords( int type ) {
307         if( Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP )
308             return this.getRecordsV21( type );
309         else if( Build.VERSION.SDK_INT >= Build.VERSION_CODES.JELLY_BEAN )
310             return IOV16.getRecords_FLIArray( type );
311
312         throw new RuntimeException( "Unexpected version!" );
313     }
314
```

```
315     private void setUrl4Rec( @Nullable String url ) {
316         this.url4Rec = url;
317     }
318
```

```
319     @Nullable
320     public String getUrl4Rec() {
321         return this.url4Rec;
322     }
323
324
```

```
325     /*
326      * MediaRecorderView
327      */
328     private void startMicRecording() {
329 //      try {
330 //          this.mediaRecorderView.startRecording();
331 //      } catch( com.grapsas.android.streamrecorder.exception.IOException e ) {
332 //          e.printStackTrace();
333 //          if( e.getCode() == 1) {
334 //              Snackbar.make(
335 //                      fab,
336 //                      getString( R.string.Unable_to_create_directory_ )
337 //                      ,Snackbar.LENGTH_LONG
```

```
338 //              ).show();
339 //              //noinspection UnnecessaryReturnStatement
340 //              return;// To be sure and for the future
341 //          }
342 //          else if( e.getCode() == 2 ) {
343 //              Snackbar.make(
344 //                      fab,
345 //                      getString( R.string.Unable_to_prepare_MediaRecorder )
346 //                      ,Snackbar.LENGTH_LONG
347 //              ).show();
348 //              //noinspection UnnecessaryReturnStatement
349 //              return;// To be sure and for the future
350 //          }
351 //      } catch( IOException e ) {
352 //          e.printStackTrace();
353 //          Snackbar.make( fab, R.string.Unable_to_prepare_MediaRecorder, Snackbar.LENGTH_LONG )
354 //                  .show();
355 //      }
356 //      catch( IllegalStateException e ) {
357 //          e.printStackTrace();
358 //          Snackbar.make( fab, R.string.Unable_to_start_MediaRecorder, Snackbar.LENGTH_LONG )
359 //                  .show();
360 //      }
361
362     this.mediaRecorderView.startRecording( MediaRecorderView.MIC_RECORDER );
363 }
364
```

```
365 private  void stopMicRecording() {
366 }
367
```

```
368 private void startStreamRecording() {
369     this.mediaRecorderView.startRecording( MediaRecorderView.STREAM_RECORDER );
370 }
371
372
```

```
373 /*
374  * MediaPlayerView
375  */
376 @Override
377 public void startPlaying( FileListItem fileListItem ) {
378     try {
379         this.mediaPlayerView.startPlaying( fileListItem );
380     } catch( com.grapsas.android.streamrecorder.exception.IOException e ) {
381         e.printStackTrace();
382         int errorCode = e.getCode();
383         switch( errorCode ) {
384             case 1:
385                 Snackbar.make( this.micFab, R.string.Unable_to_set_data_source, Snackbar.LENGTH_LONG )
386                         .show();
387                 break;
388             case 2:
389                 Snackbar.make( this.micFab, R.string.Unable_to_prepare_MediaPlayer, Snackbar.LENGTH_LONG )
390                         .show();
391                 break;
392         }
393     }
394 }
395
```

```
396  private void stopPlaying() {
397      this.mediaPlayerView.stopPlaying();
398  }
399
400  private void showPlayingView( FileListItem fileListItem ) {
401      this.mediaPlayerView.startPlayingOnlyGUI( fileListItem );
402  }
403
404
405  /*
406   * Implements interface MediaRecorderView.Events
407   */
408  @Override
409  public void onStartRecording() {
410      this.hideFabs();
411      showBottomLayout();
412  }
413
414  @Override
415  public void onStopRecording() {
416      hideBottomLayout();
417      this.filesystemChanged();
418      this.showFab();
419      this.setUrl4Rec( null );
420  }
421
422
423  /*
424   * Implements interface MediaPlayerView.Events
425   */
426  @Override
427  public void onStartPlaying() {
428      this.hideFabs();
429      showBottomLayout();
430  }
431
432  @Override
433  public void onStopPlaying() {
434      hideBottomLayout();
435      this.showFab();
436  }
437
438  @Override
439  public void onPlayerViewShow( FileListItem fileListItem ) {
440      this.pFli = fileListItem;
441      this.pMenu.getItem( 0 ).setVisible( true );
442  }
443
444  @Override
445  public void onPlayerViewHide() {
446      this.pFli = null;
447      this.pMenu.getItem( 0 ).setVisible( false );
448  }
```

```
449
450
451     /*
452      * Implements interface DeleteFile.Response
453      */
454     @Override
455     public void deleteResponse( boolean delete, @NonNull Uri file ) {
456         if( !delete )
457             return;
458         String removeMessage;
459
460         if( this.snackbar != null )
461             this.snackbar.dismiss();
462
463         if( IO.removeFile( file ) ) {
464             removeMessage = getString( R.string.TheFileRemovedSuccessfully );
465             this.onStopPlaying();
466             this.filesystemChanged();
467         }
468         else
469             removeMessage = getString( R.string.FileHasntRemoved );
470
471         this.snackbar = Snackbar
472                 .make( this.micFab, removeMessage, Snackbar.LENGTH_LONG );
473         this.snackbar.show();
474     }
475
476
477     /*
478      * Implements interface OnPageChangeListener
479      */
480     @Override
481     public void pagerStartMoving() {
482         this.hideFabs();
483         this.mediaPlayerView.stopPlaying();
484     }
485
486     @Override
487     public void pagerFinishMoving() {
488         this.showFab();
489     }
490
491
492     /*
493      * Implements interface AddFavoriteURL.Interaction
494      */
495     @Override
496     public void startAddFavoriteURL() {
497         AddFavoriteURL.newInstance( AddFavoriteURL.TYPE.JUST_REC ).show( getFragmentManager(), "" );
498     }
499
500     @Override
501     public void saveAndRec( @NonNull String url ) {
502     }
503
```

```
504     @Override
505     public void save( @NonNull String url ) {
506     }
507
```

```
508     @Override
509     public void rec( @NonNull String url ) {
510         this.setUrl4Rec( url );
511         this.startStreamRecording();
512     }
513
```

```
514     /*
515      * Inner classes
516      */
517     private static class PagerAdapter extends FragmentPagerAdapter {
518
519         private Context pContext;
520         List< WeakReference< OnDataChanged > > dataSets = new ArrayList<>();
521
```

```
522         public PagerAdapter( FragmentManager fm, Context context ) {
523             super( fm );
524             this.pContext = context;
525             dataSets.add( null );
526             dataSets.add( null );
527         }
528
```

```
529         public void refreshDataSets() {
530             for( WeakReference< OnDataChanged > wdc : dataSets ) {
531                 if( wdc != null && wdc.get() != null )
532                     wdc.get().dataChanged();
533             }
534         }
535
```

```
536         @Override
537         public Fragment getItem( int position ) {
538             switch( position ) {
539                 case 0:
540                     MicRecordsFragment mFragment = new MicRecordsFragment();
541                     dataSets.set( position, new WeakReference< OnDataChanged >( mFragment ) );
542                     return mFragment;
543                 case 1:
544                     StreamsRecordsFragment sFragment = new StreamsRecordsFragment();
545                     dataSets.set( position, new WeakReference< OnDataChanged >( sFragment ) );
546                     return sFragment;
547             }
548             return null;
549         }
550
```

```
551         @Override
552         public int getCount() {
553             return 2;
554         }
555
```

```
556         @Override
557         public CharSequence getPageTitle(int position) {
```

```
558             switch( position ) {
559                 case 0:
560                     return this.pContext.getString( R.string.Mic );
561                 case 1:
562                     return this.pContext.getString( R.string.Streams );
563             }
564 
565             return super.getPageTitle(position);
566         }
567     }
568 
569 }
570 
```

6.3.30. FavoritesURLsActivity.java

```
1   package com.grapsas.android.streamrecorder.activities;
2   
3   
4   import android.content.Intent;
5   import android.os.Bundle;
6   import android.support.annotation.NonNull;
7   import android.support.annotation.Nullable;
8   import android.support.v7.widget.Toolbar;
9   import android.view.MenuItem;
10  
11  import com.grapsas.android.streamrecorder.R;
12  import com.grapsas.android.streamrecorder.dialogs.AddFavoriteURL;
13  import com.grapsas.android.streamrecorder.dialogs.DeleteFavoriteURL;
14  import com.grapsas.android.streamrecorder.fragments.FavoritesURLsActivityFragment;
15  import com.grapsas.android.streamrecorder.misc.FavoritesURLs;
16  
17  
18  public class FavoritesURLsActivity extends MyActivity implements
19          AddFavoriteURL.Interaction,
20          DeleteFavoriteURL.Interaction {
21  
22      public static final String TYPE_KEY = "type";
23      public static final String TYPE_FAV = "fav";
24      public static final String TYPE_PRE_EXISTS = "pre_exists";
25      public static final String TYPE_DEFAULT = TYPE_FAV;
26  
27      public static final int RESULT_CODE = 2;
28      public static final String RESULT_URL_KEY = "url";
29  
30      public static final String PRE_EXISTS_FILE_NAME = "urls.json";
31  
32      private String pType;
33  
34  
```

```
35      /*
36       * Activity Overrides
37       */
38      @Override
39      protected void onCreate( Bundle savedInstanceState ) {
40          super.onCreate( savedInstanceState );
41          setContentView( R.layout.activity_favorites_urls );
42  
```

```
43      Toolbar toolbar = ( Toolbar ) findViewById( R.id.toolbar );
44      setSupportActionBar( toolbar );
45      if( getSupportActionBar() != null ) {
46          getSupportActionBar().setDisplayHomeAsUpEnabled( true );
47      }
48
49      if( this.getType().equals( TYPE_FAV ) ) {
50          this.setTitle( R.string.title_activity_favorites_urls );
51      }
52      else if( this.getType().equals( TYPE_PRE_EXISTS ) ) {
53          this.setTitle( R.string.title_activity_preexists_urls );
54      }
55  }
56
```

```
57  @Override
58  public boolean onOptionsItemSelected( MenuItem item ) {
59      if( item.getItemId() == android.R.id.home )
60          finish();
61      return super.onOptionsItemSelected( item );
62  }
63
64
```

```
65  // Tools
66  @NonNull
67  public String getType() {
68      if( this.pType == null ) {
69          Bundle args = getIntent().getExtras();
70          this.pType = args.getString( TYPE_KEY );
71      }
72      if( this.pType == null ) {
73          this.pType = TYPE_DEFAULT;
74      }
75      return this.pType;
76  }
77
```

```
78  @Nullable
79  public AddFavoriteURL.Interaction getFragment( int rId ) {
80      AddFavoriteURL.Interaction fragment = (AddFavoriteURL.Interaction)
81              getSupportFragmentManager().findFragmentById( rId );
82      return fragment;
83  }
84
```

```
85  public void urlSelected( @NonNull String url ) {
86      Intent intent = new Intent();
87      intent.putExtra( RESULT_URL_KEY, url );
88      setResult( RESULT_OK, intent );
89      finish();
90  }
91
92
```

```
93  /*
94   * Implements AddFavoriteURL.Interaction
95   */
96  @Override
97  public void startAddFavoriteURL( ) {
98      AddFavoriteURL.newInstance( AddFavoriteURL.TYPE.FAVORITE ).show( getFragmentManager(), "" );
```

```
99    }
100
101   @Override
102   public void saveAndRec( @NonNull String url ) {
103      this.save( url );
104      this.rec( url );
105   }
106
107   @Override
108   public void save( @NonNull String url ) {
109      AddFavoriteURL.Interaction fragment = this.getFragment( R.id.fragment );
110      if( fragment == null )
111         return;
112      fragment.save( url );
113   }
114
115   @Override
116   public void rec( @NonNull String url ) {
117      this.urlSelected( url );
118   }
119
120   /*
121    * Implements DeleteFavoriteURL.Interaction
122    */
123   @Override
124   public void startDeleteFavoriteURL( int position, @NonNull String url ) {
125      DeleteFavoriteURL.newInstance( position, url ).show( getFragmentManager(), "" );
126   }
127
128   @Override
129   public void DeleteURL( int position ) {
130      FavoritesURLs.removeURL( this, position );
131
132      if( getFragment( R.id.fragment ) instanceof FavoritesURLsActivityFragment )
133         ( ( FavoritesURLsActivityFragment ) getFragment( R.id.fragment ) ).notifyDataSetChanged();
134   }
135 }
136
```

6.3.31. OnDataChanged.java

```
1  package com.grapsas.android.streamrecorder.interfaces;
2
3
4  public interface OnDataChanged {
5
6     void dataChanged();
7
8  }
9
```

6.3.32. OnPageChangeListener.java

```
1  package com.grapsas.android.streamrecorder.interfaces;
2
3
4  public interface OnPageChangeListener {
5
6    void pagerStartMoving();
7    void pagerFinishMoving();
8
9  }
10
```

6.3.33. App.java

```
1  package com.grapsas.android.streamrecorder;
2
3
4  import android.app.Activity;
5  import android.app.Application;
6  import android.content.Context;
7  import android.os.Bundle;
8  import android.support.annotation.Nullable;
9
10
11  import java.lang.ref.WeakReference;
12
13
14  public class App implements Application.ActivityLifecycleCallbacks {
15
16    private static App mApp;
17    private static WeakReference< Activity > weakLastActivity;
18    private static WeakReference< Context > weakPreActivityContext;
19
20
```

```
21    public static synchronized App getInstance() {
22      if( mApp == null )
23        mApp = new App();
24      return mApp;
25    }
26
```

```
27    @Nullable
28    public Context getAvailableContext() {
29      if( this.getLastActivity() != null)
30        return this.getLastActivity();
31      else
32        return this.getPreActivityContext();
33    }
34
```

```
35    public void setPreActivityContext( @Nullable Context context ) {
36      if( context == null )
37        weakPreActivityContext = null;
38      else
```

```
39          weakPreActivityContext = new WeakReference<>( context );
40      }
41
42      @Nullable
43      public Context getPreActivityContext() {
44          if( weakPreActivityContext == null )
45              return null;
46          else
47              return weakPreActivityContext.get();
48      }
49
50      public void setLastActivity( @Nullable Activity activity ) {
51          if( activity == null )
52              weakLastActivity = null;
53          else {
54              setPreActivityContext( activity.getApplicationContext() );
55              weakLastActivity = new WeakReference<>( activity );
56          }
57      }
58
59      @Nullable
60      public Activity getLastActivity() {
61          if( weakLastActivity == null )
62              return null;
63          return weakLastActivity.get();
64      }
65
66
67      /*
68       * Implements Application.ActivityLifecycleCallbacks
69       */
70      @Override
71      public void onActivityCreated( Activity activity, Bundle savedInstanceState ) {
72          setLastActivity( activity );
73      }
74
75      @Override
76      public void onActivityStarted( Activity activity ) {
77          setLastActivity( activity );
78      }
79
80      @Override
81      public void onActivityResumed( Activity activity ) {
82          setLastActivity( activity );
83      }
84
85      @Override
86      public void onActivityPaused( Activity activity ) {
87          setLastActivity( null );
88      }
89
90      @Override
91      public void onActivityStopped( Activity activity ) {}
```

```
92
93      @Override
94      public void onActivitySaveInstanceState( Activity activity, Bundle outState ) {}
95
96      @Override
97      public void onActivityDestroyed( Activity activity ) {}
98
99  }
100
```