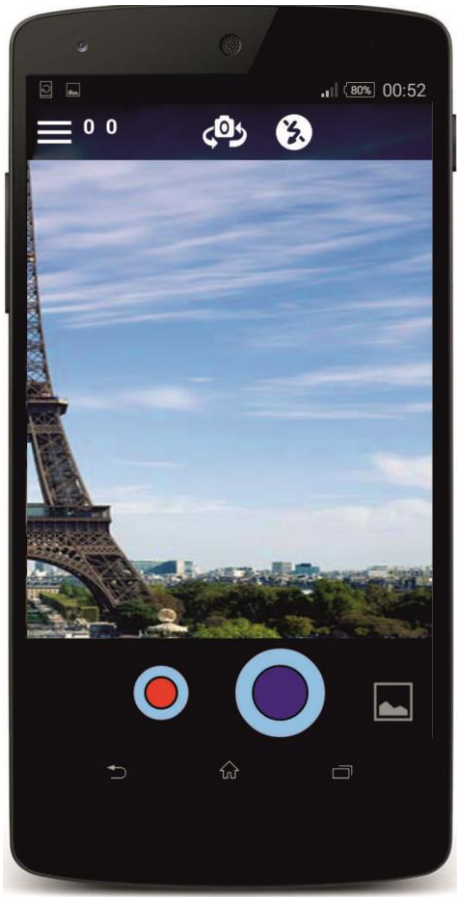
ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Εφαρμογή διαχείρισης φωτογραφιών σε περιβάλλον Android



Του φοιτητή

Αβραάμ Πιπερίδης

Αρ. Μητρώου: 134101

Επιβλέπων καθηγητής

Δεληγιάννης Ιγνάτιος

Θεσσαλονίκη 2017

ΠΡΟΛΟΓΟΣ

Open Source Android Camera Application

Εφαρμογή για την πλατφόρμα Android .Εφαρμογή φωτογραφικής μηχανής με τα εξής χαρακτηριστικά.

https://github.com/avraampiperidis/AndroidCamera

ΠΕΡΙΛΗΨΗ

- Ανάλυση λήψη φωτογραφιών
  - Επιλογή ανάλυσης από αυτές που υποστηρίζει η συσκευή.
- λήψη φωτογραφιών με διάφορα εφέ / μάσκες (mono,negative,sepia),
  - εφαρμογή εφέ στις φωτογραφίες οπος(invert,greyScale,Sepia με πέρασμα

τιμών,contrast,brightness,emboss,engrave,boost,snowEffect,shading filter,reflection, και color filters)

- λήψη φωτογραφιών αναλόγως το σκηνικό (πορτρέτο , πεδιάδας, νύχτα, νυχτερινό πορτρέτο, παραλία, χιόνι, πυροτεχνήματα, αθλήματα, πάρτι,)
- δυνατότητες φλας και focus (auto flash, λειτουργία φακού ...,auto focus, macro focus)
- λήψη φωτογραφίας από μπροστινή και πίσω κάμερα
- Gallery για την προβολή των φωτογραφιών
- δυνατότητα περικοπής και περιστροφής των φωτογραφιών
- δυνατότητα επεξεργασίας των φωτογραφιών για εφέ
- υποστήριξη σε εκατοντάδες μοντέλα (συσκευές android)
- Εμφάνιση κλήση / μιρες της συσκευής
- Open Source ο καθένας να μπορεί να την επεκτείνει

Επιπλέον.

- Έμφαση στην δημιουργία Unit tests και Integration tests.
- Unit test , Integration test , Stress Test στο cloud σε δεκάδες διαφορετικές συσκευές (Amazon web services,testing services Student Pack).Και μελέτη των αποτελεσμάτων
- UML Diagrams και flow charts/use cases.
- Έμφαση στα UI/UX elements για ευχάριστη χρίση από τον τελικό χρήστη.
- Χρήση Java native interface , ανάθεση βαριών δουλειών σε native C++ μέσα από την java για εξηκονομιση μνήμης του jvm, και αποφυγή OutOfMemoryException.
- Η Όλη ανάπτυξη της εφαρμογής ξεκίνησε και γίνετε αποκλειστικά στο android studio.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΙΣΑΓΩΓΗ

Στόχος της παρούσας πτυχιακής εργασίας είναι η απόκτηση των γνώσεων που απαιτούνται για την ανάπτυξη εφαρμογών για το λειτουργικό Android. Η εργασία περιλαμβάνει θεωρητικό και πρακτικό μέρος. Για το πρακτικό μέρος αναπτύχθηκε μία εφαρμογή φωτογραφική μηχανή.

Εισαγωγή στις τεχνολογίες

Αναφορα και μια εισαγωγη στις βασικες τεχνολογίες που χρισιμοποιουνται απο την εφαρμογη και το γενικο τεχνολογικο πλαισιο.

Εισαγωγη στις εξεις τεχνολογιες.Java, Android, jvm, dalvik, testing, uml, IDE

Αντικειμενοστρεφής προγραμματισμός

Στην πληροφορική **αντικειμενοστρεφή προγραμματισμό** (object-oriented programming), ή ΑΠ, ονομάζουμε ένα προγραμματιστικό υπόδειγμα το οποίο εμφανίστηκε στα τέλη της δεκαετίας του 1960 και καθιερώθηκε κατά τη δεκαετία του 1990, αντικαθιστώντας σε μεγάλο βαθμό το παραδοσιακό υπόδειγμα του δομημένου προγραμματισμού. Πρόκειται για μία μεθοδολογία ανάπτυξης προγραμμάτων, υποστηριζόμενη από κατάλληλες γλώσσες προγραμματισμού, όπου ο χειρισμός σχετιζόμενων δεδομένων και των διαδικασιών που επενεργούν σε αυτά γίνεται από κοινού, μέσω μίας δομής δεδομένων που τα περιβάλλει ως αυτόνομη οντότητα με ταυτότητα και δικά της χαρακτηριστικά. Αυτή η δομή δεδομένων καλείται *αντικείμενο* και αποτελεί πραγματικό στιγμιότυπο στη μνήμη ενός σύνθετου, και πιθανώς οριζόμενου από τον χρήστη, τύπου δεδομένων ονόματι *κλάση*. Η κλάση προδιαγράφει τόσο δεδομένα όσο και τις διαδικασίες οι οποίες επιδρούν επάνω τους· αυτή υπήρξε η πρωταρχική καινοτομία του ΑΠ.

Έτσι μπορεί να οριστεί μία προδιαγραφή δομής αποθήκευσης (π.χ. μία κλάση «τηλεόραση») η οποία να περιέχει τόσο ιδιότητες (π.χ. μία μεταβλητή «τρέχον κανάλι») όσο και πράξεις ή χειρισμούς επί αυτών των ιδιοτήτων (π.χ. μία διαδικασία «άνοιγμα της τηλεόρασης»). Στο εν λόγω παράδειγμα κάθε υλική τηλεόραση (κάθε αντικείμενο αποθηκευμένο πραγματικά στη μνήμη) αναπαρίσταται ως ξεχωριστό, «φυσικό» στιγμιότυπο αυτής της πρότυπης, ιδεατής κλάσης. Επομένως μόνο τα αντικείμενα καταλαμβάνουν χώρο στη μνήμη του υπολογιστή ενώ οι κλάσεις αποτελούν απλώς «καλούπια». Οι αιτίες που ώθησαν στην ανάπτυξη του ΑΠ ήταν οι ίδιες με αυτές που οδήγησαν στην ανάπτυξη του δομημένου προγραμματισμού (ευκολία συντήρησης, οργάνωσης, χειρισμού και επαναχρησιμοποίησης κώδικα μεγάλων και πολύπλοκων εφαρμογών), όμως τελικώς η αντικειμενοστρέφεια επικράτησε καθώς μπορούσε να αντεπεξέλθει σε προγράμματα πολύ μεγαλύτερου όγκου και πολυπλοκότητας.

Java

Η Java είναι μια αντικειμενοστρεφής γλώσσα προγραμματισμού που σχεδιάστηκε από την εταιρεία πληροφορικής Sun Microsystems.

Σχήμα 1 Το λογότυπο της γλώσσας Java.

Στις αρχές του 1991, η Sun αναζητούσε το κατάλληλο εργαλείο για να αποτελέσει την πλατφόρμα ανάπτυξης λογισμικού σε μικρο-συσκευές (έξυπνες οικιακές συσκευές έως πολύπλοκα συστήματα παραγωγής γραφικών). Τα εργαλεία της εποχής ήταν γλώσσες όπως η C++ και η C. Μετά από διάφορους πειραματισμούς προέκυψε το συμπέρασμα ότι οι υπάρχουσες γλώσσες δεν μπορούσαν να καλύψουν τις ανάγκες τους. Ο "πατέρας" της Java, James Gosling, που εργαζόταν εκείνη την εποχή για την Sun, έκανε ήδη πειραματισμούς πάνω στη C++ και είχε παρουσιάσει κατά καιρούς κάποιες πειραματικές γλώσσες (C++ ++,που μετέπειτα ονομαστικέ C#) ως πρότυπα για το νέο εργαλείο που αναζητούσαν στην Sun. Τελικά μετά από λίγο καιρό κατέληξαν με μια πρόταση για το επιτελείο της εταιρίας, η οποία ήταν η γλωσσα Oak. Το όνομά της το πήρε από το ομώνυμο δένδρο (βελανιδιά) το οποίο ο Gosling είχε έξω από το γραφείο του και έβλεπε κάθε μέρα.

Η Oak ήταν μία γλώσσα που διατηρούσε μεγάλη συγγένεια με την C++. Παρόλα αυτά είχε πολύ πιο έντονο αντικειμενοστρεφή (object oriented) χαρακτήρα σε σχέση με την C++ και χαρακτηριζόταν για την απλότητα της. Σύντομα οι υπεύθυνοι ανάπτυξης της νέας γλώσσας ανακάλυψαν ότι το όνομα Oak ήταν ήδη κατοχυρωμένο οπότε κατά την διάρκεια μιας εκ των πολλών συναντήσεων σε κάποιο τοπικό καφέ αποφάσισαν να μετονομάσουν το νέο τους δημιούργημα σε Java που εκτός των άλλων ήταν το όνομα της αγαπημενης ποικιλιας καφέ για τους δημιουργούς της. Η επίσημη εμφάνιση της Java αλλα και του HotJava (πλοηγός με υποστήριξη Java) στη βιομηχανία της πληροφορικής έγινε το Μάρτιο του 1995 όταν η Sun την ανακοίνωσε στο συνέδριο Sun World 1995. Ο πρώτος μεταγλωττιστής (compiler) της ήταν γραμμένος στη γλώσσα C από τον James Gosling. Το 1994, ο A.Van Hoff ξαναγράφει τον μεταγλωττιστή της γλώσσας σε Java, ενώ το Δεκέμβριο του 1995 πρώτες οι IBM, Borland, Mitsubishi Electronics, Sybase και Symantec ανακοινώνουν σχέδια να χρησιμοποιήσουν τη Java για την δημιουργία λογισμικού. Από εκεί και πέρα η Java

ακολουθεί μία ανοδική πορεία και είναι πλέον μία από τις πιο δημοφιλείς γλώσσες στον χώρο της πληροφορικής. Στις 13 Νοεμβρίου του 2006 η Java έγινε πλέον μια γλώσσα ανοιχτού κώδικα (GPL) όσον αφορά το μεταγλωττιστή (javac) και το πακέτο ανάπτυξης (JDK, Java Development Kit).

Στις 27 Απριλίου 2010 η εταιρία λογισμικού Oracle Corporation ανακοίνωσε ότι μετά από πολύμηνες συζητήσεις ήρθε σε συμφωνία για την εξαγορά της Sun Microsystems και των τεχνολογιών (πνευματικά δικαιώματα/ πατέντες) που η δεύτερη είχε στην κατοχή της ή δημιουργήσει. Η συγκεκριμένη συμφωνία θεωρείται σημαντική για το μέλλον της Java και του γενικότερου οικοσυστήματος τεχνολογιών γύρω από αυτή μιας και ο έμμεσος έλεγχος της τεχνολογίας και η εξέλιξη της περνάει σε άλλα χέρια.

Ένα από τα βασικά πλεονεκτήματα της Java έναντι των περισσότερων άλλων γλωσσών είναι η ανεξαρτησία του λειτουργικού συστήματος και πλατφόρμας. Τα προγράμματα που είναι γραμμένα σε Java τρέχουνε ακριβώς το ίδιο σε Windows, Linux, Unix και Macintosh (σύντομα θα τρέχουν και σε Playstation καθώς και σε άλλες κονσόλες παιχνιδιών) χωρίς να χρειαστεί να ξαναγίνει μεταγλώττιση (compiling) ή να αλλάξει ο πηγαίος κώδικας για κάθε διαφορετικό λειτουργικό σύστημα.

Ακόμα μία ιδέα που βρίσκεται πίσω από τη Java είναι η ύπαρξη του συλλέκτη απορριμμάτων (Garbage Collector). Συλλογή απορριμμάτων είναι μία κοινή ονομασία που χρησιμοποιείται στον τομέα της πληροφορικής για να δηλώσει την ελευθέρωση τμημάτων μνήμης από δεδομένα που δε χρειάζονται και δε χρησιμοποιούνται άλλο.

Παρόλο που η εικονική μηχανή προσφέρει όλα αυτά (και όχι μόνο) τα πλεονεκτήματα, η *Java* αρχικά ήταν πιο αργή σε σχέση με άλλες προγραμματιστικές γλώσσες υψηλού επιπέδου (high-level) όπως η C και η C++. Εμπειρικές μετρήσεις στο παρελθόν είχαν δείξει ότι η C++ μπορούσε να είναι αρκετές φορές γρηγορότερη από την Java. Ωστόσο γίνονται προσπάθειες από τη Sun για τη βελτιστοποίηση της εικονικής μηχανής, ενώ υπάρχουν και άλλες υλοποιήσεις της εικονικής μηχανής από διάφορες εταιρίες (όπως της IBM), οι οποίες μπορεί σε κάποια σημεία να προσφέρουν καλύτερα και σε κάποια άλλα χειρότερα αποτελέσματα. Επιπλέον με την καθιέρωση των μεταγλωττιστών JIT (Just In Time), οι οποίοι μετατρέπουν τον κώδικα byte απευθείας σε γλώσσα μηχανής, η διαφορά ταχύτητας από τη C++ έχει μικρύνει κατά πολύ.

Οι τελευταίες εκδόσεις του javac με τη χρήση της τεχνολογίας Hot Spot έχουν καταφέρει αξιόλογες επιδόσεις που πλησιάζουν ή και ξεπερνούν σε μερικές περιπτώσεις τον εγγενή κώδικα.

Η εικονική μηχανή της Java (JVM)

Αφού γραφεί κάποιο πρόγραμμα σε Java, στη συνέχεια μεταγλωτίζεται μέσω του μεταγλωττιστή javac, ο οποίος παράγει έναν αριθμό από αρχεία .class (κώδικας byte ή bytecode). Ο κώδικας byte είναι η μορφή που παίρνει ο πηγαίος κώδικας της Java όταν μεταγλωττιστεί. Όταν πρόκειται να εκτελεστεί η εφαρμογή σε ένα μηχάνημα, το Java Virtual Machine που πρέπει να είναι εγκατεστημένο σε αυτό θα αναλάβει να διαβάσει τα αρχεία .class. Στη συνέχεια τα μεταφράζει σε γλώσσα μηχανής που να υποστηρίζεται από το λειτουργικό σύστημα και τον επεξεργαστή, έτσι ώστε να εκτελεστεί (αυτό συμβαίνει με την παραδοσιακή Εικονική Μηχανή (Virtual Machine). Πιο σύγχρονες εφαρμογές της εικονικής Μηχανής μπορούν και μεταγλωττίζουν εκ των προτέρων τμήματα bytecode απευθείας σε κώδικα μηχανής (εγγενή κώδικα ή native code) με αποτέλεσμα να βελτιώνεται η ταχύτητα). Χωρίς αυτό δε θα ήταν δυνατή η εκτέλεση λογισμικού γραμμένου σε Java. Η JVM είναι

λογισμικό που εξαρτάται από την πλατφόρμα, δηλαδή για κάθε είδος λειτουργικού συστήματος και αρχιτεκτονικής επεξεργαστή υπάρχει διαφορετική έκδοση του. Έτσι υπάρχουν διαφορετικές JVM για Windows, Linux, Unix, Macintosh, κινητά τηλέφωνα, παιχνιδομηχανές κλπ.

Οτιδήποτε θέλει να κάνει ο προγραμματιστής (ή ο χρήστης) γίνεται μέσω της εικονικής μηχανής. Αυτό βοηθάει στο να υπάρχει μεγαλύτερη ασφάλεια στο σύστημα γιατί η εικονική μηχανή είναι υπεύθυνη για την επικοινωνία χρήστη - υπολογιστή. Ο προγραμματιστής δεν μπορεί να γράψει κώδικα ο οποίος θα έχει καταστροφικά αποτελέσματα για τον υπολογιστή γιατί η εικονική μηχανή θα τον ανιχνεύσει και δε θα επιτρέψει να εκτελεστεί. Από την άλλη μεριά ούτε ο χρήστης μπορεί να κατεβάσει «κακό» κώδικα από το δίκτυο και να τον εκτελέσει. Αυτό είναι ιδιαίτερα χρήσιμο για μεγάλα κατανεμημένα συστήματα όπου πολλοί χρήστες χρησιμοποιούν το ίδιο πρόγραμμα συγχρόνως.

Η εικονική μηχανή Java δεν έχει καμία γνώση όσο αναφορά την γλώσσα προγραμματισμού Java.
Γνωρίζει μόνο το πρότυπο αρχείου κλάσης (class file format) το οποίο περιέχει εντολές της εικονικής μηχανής (bytecodes), έναν πίνακα συμβόλων καθώς και διάφορες βοηθητικές πληροφορίες.

Οι σχεδιαστές της εικονικής μηχανής Java περιγράφουν τις διάφορες λειτουργίες στο παρακάτω πρότυπο:

http://docs.oracle.com/javase/specs/jvms/se7/html/ index.html

Για να υλοποιησετε μία σωστή εικονική μηχανή Java, αρκεί να διαβάσετε σωστά το πρότυπο αρχείου κλάσης και να μπορεί η μηχανή σας να εκτελεί τις λειτουργίες που περιγράφονται στο παραπάνω κείμενο.

Το πρότυπο της εικονικής μηχανής δεν αναφέρει πουθενά θέματα υλοποίησης, όπως π.χ πως υλοποιείται η κάθε εντολή του JVM με χαμηλότερου επιπέδου εντολές ή πως λειτουργεί ο συλλέκτης σκουπιδιών, κ.τ.λ.

Υπάρχουν διάφορες υλοποιήσεις JVM:

http://en.wikipedia.org/wiki/List_of_Java_virtual_machines

http://en.wikipedia.org/wiki/Free_Java_implementations

http://openjdk.java.net/

Όπως και η γλώσσα προγραμματισμού Java, το JVM υποστηρίζει δύο ειδών τύπους:

βασικούς τύπους (primitive types) αναφορές

(references)

Το JVM υποθέτει πως το μεγαλύτερο μέρος του έλεγχου τύπων έχει πραγματοποιηθεί κατά την διάρκεια της μεταγλώττισης.

Για αυτό το λόγο οι εντολές της εικονικής μηχανής έχουν εκδόσεις ανά τύπο, π.χ iadd για την πρόσθεση ακεραίων και fadd για την πρόσθεση αριθμών κινητής υποδιαστολής.

- byte, 8-bit προσημασμένοι ακέραιοι σε συμπλήρωμα ως προς 2
- short, 16-bit προσημασμένοι ακέραιοι σε συμπλήρωμα ως προς 2
- int, 32-bit προσημασμένοι ακέραιοι σε συμπλήρωμα ως προς 2
- long, 64-bit προσημασμένοι ακέραιοι σε συμπλήρωμα ως προς 2
- char, 16-bit μη-προσημασμένοι ακέραιοι σε Unicode
- float, 32-bit single precision IEEE 754 double,
- 64-bit double precision IEEE 754
- boolean, true ή false (υλοποιούνται με int και 0 για false, 1 για true)

Το JVM παρέχει υποστήριξη για αντικείμενα.

Ένα αντικείμενο είναι είτε ένα δυναμικά δημιουργημένο στιγμιότυπο μίας κλάσης είτε ένας πίνακας.

Η πρόσβαση στα αντικείμενα γίνεται μέσω του τύπου *αναφορά* (reference).

Dalvik

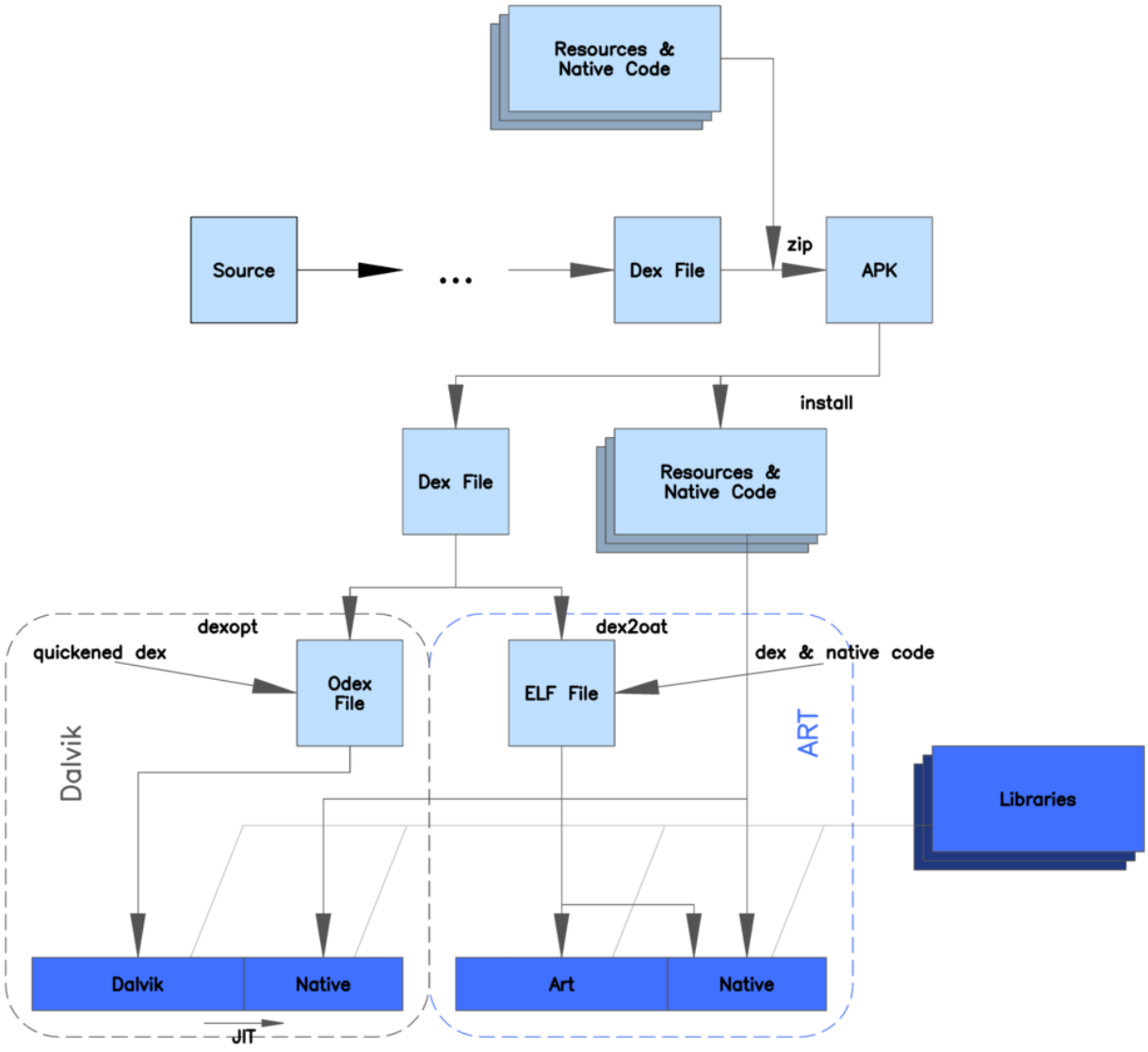
Το περιβάλλον εκτέλεσης (runtime) του Android

Το περιβάλλον εκτέλεσης βρίσκεται και αυτό στο δεύτερο επίπεδο μετρώντας από κάτω προς τα πάνω. Μέχρι την έκδοση 4.4 του Android χρησιμοποιούνταν η Dalvik Virtual Machine, ενώ από την 5η έκδοση και έπειτα η Dalvik Virtual Machine έχει αντικατασταθεί από την εικονική μηχανή Android Runtime (ART). Και οι δύο εικονικές μηχανές είναι ειδικά σχεδιασμένες και βελτιστοποιημένες για το Android.

Η εικονική μηχανή Dalvik επιτρέπει σε κάθε εφαρμογή να εκτελείται στην δική της διεργασία, με το δικό της στιγμιότυπο (instance) της Dalvik. Σε αντίθεση με την Dalvik η οποία χρησιμοποιεί just-in-time μεταγλώττιση, η ART χρησιμοποιεί ahead-of-time μεταγλώττιση κατά την εγκατάσταση της εφαρμογής στην συσκευή. Έτσι επιτυγχάνεται καλύτερη απόδοση εφόσον δεν χάνεται χρόνος για να γίνει η μεταγλώττιση του κώδικα όταν τρέχει η εφαρμογή. Επιπλέον, η ART χρησιμοποιεί βελτιωμένο συλλέκτη σκουπιδιών (garbage collector). Στα μειονεκτήματα της ART είναι ότι απαιτείται περισσότερος χρόνος για να γίνει η εγκατάσταση,

και ότι ο κώδικας μηχανής που προκύπτει από την μεταγλώττιση είναι μεγαλύτερος σε μέγεθος οπότε απαιτείται περισσότερος χώρος για την αποθήκευσή του.

Το περιβάλλον εκτέλεσης παρέχει επίσης μία σειρά από βιβλιοθήκες οι οποίες επιτρέπουν στους προγραμματιστές να γράφουν εφαρμογές χρησιμοποιώντας πολλές από τις κλάσεις οι οποίες υπάρχουν στην Java Standard Edition.



A comparison of Dalvik and ART architectures

Android

Το Android είναι λειτουργικό σύστημα για συσκευές κινητής τηλεφωνίας το οποίο τρέχει τον πυρήνα του λειτουργικού Linux. Αρχικά αναπτύχθηκε από την Google και αργότερα από την Open Handset Alliance .[1] Επιτρέπει στους κατασκευαστές λογισμικού να

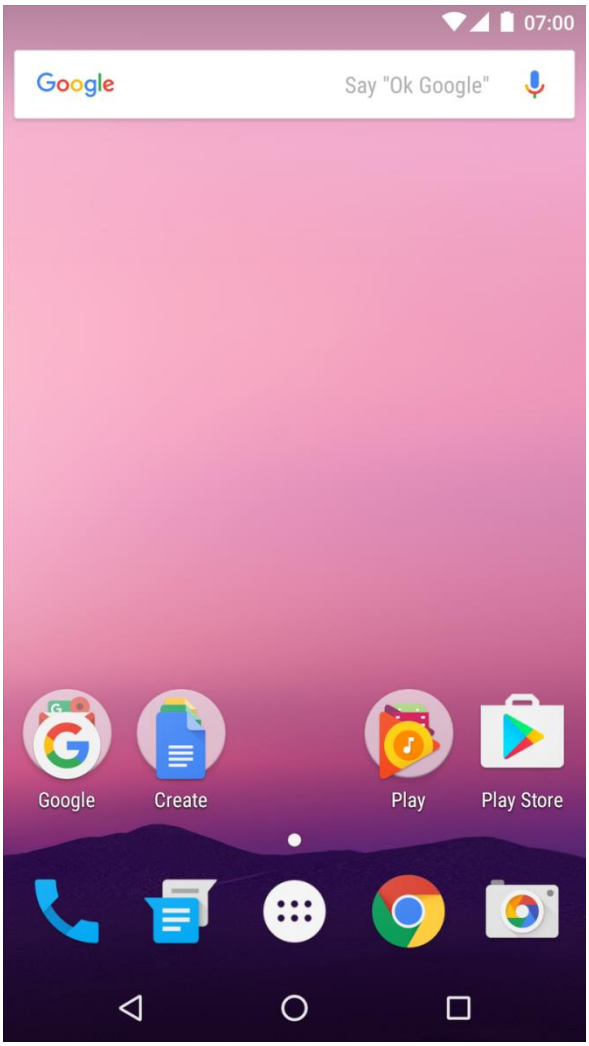
συνθέτουν κώδικα με την χρήση της γλώσσας προγραμματισμού Java, ελέγχοντας την συσκευή μέσω βιβλιοθηκών λογισμικού ανεπτυγμένων από την Google.[2] Το Android είναι κατά κύριο λόγο σχεδιασμένο για συσκευές με οθόνη αφής, όπως τα έξυπνα τηλέφωνα και τα τάμπλετ, με διαφορετικό περιβάλλον χρήσης για τηλεοράσεις (Android TV), αυτοκίνητα (Android Auto) και ρολόγια χειρός (Android Wear). Παρόλο που έχει αναπτυχθεί για συσκευές με οθόνη αφής, έχει χρησιμοποιηθεί σε κονσόλες παιχνιδιών, ψηφιακές φωτογραφικές μηχανές, συνηθισμένους Η/Υ (π.χ. το HP Slate 21) και σε άλλες ηλεκτρονικές συσκευές.

Το Android είναι το πιο ευρέως διαδεδομένο λογισμικό στον κόσμο. Οι συσκευές με Android έχουν περισσότερες πωλήσεις από όλες τις συσκευές Windows, iOS και Mac OS X μαζί.

Η πρώτη παρουσίαση της πλατφόρμας Android έγινε στις 5 Νοεμβρίου 2007, παράλληλα με την ανακοίνωση της ίδρυσης του οργανισμού Open Handset Alliance, μιας κοινοπραξίας 48 τηλεπικοινωνιακών εταιριών, εταιριών λογισμικού καθώς και κατασκευής hardware, οι οποίες είναι αφιερωμένες στην ανάπτυξη και εξέλιξη ανοιχτών προτύπων στις συσκευές κινητής τηλεφωνίας.[3][4] Η Google δημοσίευσε το μεγαλύτερο μέρος του κώδικα του Android υπό τους όρους της Apache License, μιας ελεύθερης άδειας λογισμικού. Το λογότυπο για το λειτουργικό σύστημα Android είναι ένα ρομπότ σε χρώμα πράσινου μήλου και σχεδιάστηκε από τη γραφίστρια Ιρίνα Μπλόκ.[5]

Η τελευταία έκδοση καλείται Android 7.1.2 Nougat και φέρνει σημαντικές αλλαγές.

Android logo



Η αρχική οθόνη του Android 7.1.2 (με κάποιες εφαρμογές της Google)

Χαρακτηριστικά

Λειτουργίες Οθόνης	Η πλατφόρμα είναι προσαρμόσιμη σε πολλές ανάλυσεις οθόνης (από VGA μέχρι 4K), δισδιάστατες ψηφιακές γραφικές βιβλιοθήκες, τρισδιάστατα γραφικά βασισμένα στην OpenGL ES 3.0+ έκδοση χαρακτηριστικών, καθώς και παραδοσιακές απεικονίσεις οθόνης "έξυπνων" συσκευών κινητής τηλεφωνίας.
Αποθήκευση Δεδομένων	Χρήση βάσης δεδομένων SQLite για τις ανάγκες αποθήκευσης
Συνδεσιμότητα	Το Android υποστηρίζει τεχνολογίες συνδεσιμότητας συμπεριλαμβανομένου GSM/EDGE/UMTS/HSPA/HSPA+/LTE,

	3G, 4G, CDMA, EV-DO, Bluetooth, NFC, και Wi-Fi.
Αποστολή μηνυμάτων	SMS και MMS είναι οι διαθέσιμοι τρόποι ανταλλαγής μηνυμάτων.
Περιήγηση στον Ιστό	Για την περιήγηση στον ιστό το Android διαθέτει φυλλομετρητή βασισμένο στην ανοιχτή τεχνολογία WebKit. Και άλλοι φυλλομετρητές είναι διαθέσιμοι από το Google play
Υποστήριξη Java	Λογισμικό γραμμένο στην Java είναι δυνατόν να μεταγλωττιστεί και να εκτελεστεί στην εικονική μηχανή Dalvik, η οποία αποτελεί εξειδικευμένη υλοποίηση εικονικής μηχανής, σχεδιασμένης για χρήση σε φορητές συσκευές, παρόλο που δεν είναι πρότυπη εικονική μηχανή Java.
Υποστήριξη Πολυμέσων	Το λειτουργικό Android υποστηρίζει τις ακόλουθα μορφές ήχου, στατικής και κινούμενης εικόνας: H.263, H.264 (σε 3GP ή MP4 container), MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, MP3, MIDI, OGG Vorbis, WAV, JPEG, PNG, GIF, BMP.
Επιπλέον υποστήριξη υλικού	Το λειτουργικό Android μπορεί να συνεργαστεί με κάμερες στατικής ή κινούμενης εικόνας, οθόνες αφής, GPS, αισθητήρες επιτάχυνσης, μαγνητόμετρα, δισδιάστατους καθώς και τρισδιάστατους επιταχυντές γραφικών.
Περιβάλλον Ανάπτυξης Λογισμικού	Περιλαμβάνει ένας προσομοιωτή συσκευής, εργαλεία για διόρθωση σφαλμάτων, μνήμη και εργαλεία ανάλυσης της απόδοσης του εκτελέσιμου λογισμικού καθώς και ένα επιπρόσθετο για το Android Studio.
Αγορά και Εγκατάσταση Εφαρμογών	Παρόμοια με το App Store του iOS, το Google Play είναι ένας κατάλογος εφαρμογών που μπορούν να μεταφορτωθούν και εγκατασταθούν στην συσκευή άμεσα μέσω ασύρματων καναλιών, χωρίς την χρήση υπολογιστή. Αρχικά μόνο δωρεάν εφαρμογές ήταν δυνατόν να εγκατασταθούν. Εφαρμογές επί πληρωμή ήταν μετέπειτα διαθέσιμες στο Google play στις ΗΠΑ ύστερα από τις 19 Φεβρουαρίου 2009.
Οθόνη Αφής Πολλαπλών Σημείων	Το λειτουργικό Android υποστηρίζει

	οθόνες αφής πολλαπλών σημείων αλλά η δυνατότητα αυτή είχε κλειδωθεί σε επίπεδο πυρήνα (πιθανόν για αποφυγή παραβιάσεων των πατεντών λογισμικού της Apple στις τεχνολογίες οθονών αφής). Κυκλοφορούσε μια ανεπίσημη τροποποίηση (mod) που έχει αναπτυχθεί για να υποστηρίζει πολλαπλή επαφή (multi-touch), αλλά απαιτούσε δικαιώματα πρόσβασης υπερχρήστη (superuser) στη συσκευή για να γραφεί στη μνήμη flash ένας πυρήνας που να μην είναι υπογεγραμμένος (unsigned kernel).

Τωρινά χαρακτηριστικά και λειτουργίες

Linux

Το **Linux** (*Λίνους*)[6] ή **GNU/Linux** (*Γκνού/Λίνους*),[7] είναι ένα λειτουργικό σύστημα που αποτελείται από ελεύθερο λογισμικό. Η χρήση του είναι παρόμοια με αυτή του Unix, αλλά όλος ο πηγαίος κώδικας του έχει γραφτεί από την αρχή ως ελεύθερο λογισμικό υπό την ελεύθερη άδεια χρήσης GNU General Public License.

Το Linux μπορεί να εγκατασταθεί και να λειτουργήσει σε μεγάλη ποικιλία υπολογιστικών συστημάτων, από μικρές συσκευές όπως κινητά τηλέφωνα[8][9] μέχρι μεγάλα υπολογιστικά συστήματα και υπερυπολογιστές.[10][11]Τον Ιούνιο του 2014, 97% των 500 ισχυρότερων υπερυπολογιστών χρησιμοποιούν κάποια διανομή Linux.[12] Το Linux χρησιμοποιείται κατά κόρον σε διακομιστές, αφού η καταγεγραμμένη χρήση του σε αυτούς για το 2008 ανέρχεται σε 60% του συνόλου της αγοράς. Σε ότι αφορά τους προσωπικούς υπολογιστές, η δημοφιλία των λειτουργικών συστημάτων Mac OS X ή Microsoft Windows είναι υψηλότερη, ενώ το αντίστοιχο ποσοστό του Linux είναι σχεδόν 2%.[13] Τα τελευταία χρόνια παρατηρείται άνοδος του Linux και σε προσωπικούς υπολογιστές, χάρη στην πολύ καλύτερη υποστήριξη και συμβατότητα με τα διάφορα συστήματα και υλικά υπολογιστών απ'ότι στο παρελθόν, καθώς και την αναβαθμισμένη αισθητικά και χρηστικά λειτουργικότητα των διάφορων διανομών. Επίσης διανομές Linux είναι εξαιρετικά δημοφιλής στα παλαιότερα ή μικρής επεξεργαστικής ισχύς μηχανήματα (όπως netbook[14][15]), καθώς συχνά έχει πολύ χαμηλότερες απαιτήσεις επεξεργαστικής ισχύος, μνήμης, και αποθηκευτικού χώρου σε σχέση με άλλα λειτουργικά συστήματα.

Το Linux συχνά προσφέρεται στο χρήστη σε διάφορες *διανομές Linux*. Χαρακτηριστικό των διανομών είναι η μεγάλη δυνατότητα παραμετροποίησης και επιλογής που προσφέρουν καθώς κάθε μια απευθύνεται σε διαφορετικό τύπο χρηστών. Ανάλογα με την φιλοσοφία που ακολουθεί κάθε διανομή μπορεί να δίνει μεγαλύτερη βάση στη φιλικότητα προς τον χρήστη, στις εφαρμογές πολυμέσων, την ευκολία παραμετροποίησης, απλότητα του συστήματος, μόνο ελεύθερο λογισμικό, χαμηλές απαιτήσεις σε πόρους, και άλλα.

Δημιουργός του πυρήνα Linux είναι ο Λίνους Τόρβαλντς, από το όνομα του οποίου προήλθε και η ονομασία Linux. Ο Τόρβαλντς άρχισε να αναπτύσσει ένα αρχικό πυρήνα το 1991 χρησιμοποιώντας κώδικα από το ακαδημαϊκό λειτουργικό σύστημα MINIX του Άντριου Τάνενμπάουμ, το οποίο και μετεξέλιξε ανεξάρτητα, και κατόπιν υιοθέτησε τα προγράμματα και βιβλιοθήκες του λειτουργικού συστήματος GNU του Ρίτσαρντ Στάλλμαν. Πάνω στον αρχικό πυρήνα του Τόρβαλντς έχουν εργαστεί χιλιάδες χρήστες, κοινότητες αλλά και εταιρείες. Λόγω της συνύπαρξης του πυρήνα Linux και του συστήματος GNU στο σχηματισμό του Linux ως λειτουργικό σύστημα, συχνά το σύστημα αυτό αναφέρεται ώς GNU/Linux, όπως προτιμά το Ίδρυμα Ελεύθερου Λογισμικού.[16][17]

IDE Android Studio

Το Android Studio είναι ένα ολοκληρωμένο προγραμματιστικό περιβάλλον (IDE) για ανάπτυξη εφαρμογών στην πλατφόρμα Android. Ανακοινώθηκε στις 16 Μαΐου 2013 στο συνέδριο Google I/O από την Google Product Manager, Katherine Chou. Το Android Studio είναι διαθέσιμο ελεύθερα με την άδεια Apache License 2.0.[18]

Το Android Studio ήταν διαθέσιμο σε πρώιμο στάδιο για προεπισκόπηση ξεκινώντας από την έκδοση 0.1 τον Μάιο του 2013, έπειτα ξεκίνησε το δοκιμαστικό στάδιο από την έκδοση 0.8 που βγήκε τον Ιούνιο του 2014[19]. Η πρώτη σταθερή έκδοση βγήκε το Δεκέμβριο του 2014, με την έκδοση 1.0.[20]

Βασισμένο στο λογισμικό της JetBrains' IntelliJ IDEA, το Android Studio σχεδιάστηκε αποκλειστικά για προγραμματισμό Android [4]. Είναι διαθέσιμο για Windows, Mac OS X και Linux[19][21], και αντικατέστησε τα Eclipse Android Development Tools (ADT) ως το κύριο IDE της Google για ανάπτυξη εφαρμογών Android.

Android Studio Logo

Unified Modeling Language

Η Unified Modeling Language (UML, μη δόκιμη απόδοση στην Ελληνική γλώσσα: Ενοποιημένη Γλώσσα Μοντελοποίησης) πλέον είναι η πρότυπη γλώσσα μοντελοποίησης στη μηχανική λογισμικού. Χρησιμοποιείται για τη γραφική απεικόνιση, προσδιορισμό, κατασκευή και τεκμηρίωση των στοιχείων ενός συστήματος λογισμικού. Μπορεί να χρησιμοποιηθεί σε διάφορες φάσεις ανάπτυξης, από την ανάλυση απαιτήσεων ως τον έλεγχο ενός ολοκληρωμένου συστήματος. Αποτελείται από ένα σύνολο προσυμφωνημένων όρων, συμβόλων και διαγραμμάτων που επιτρέπουν:

- την εμφάνιση των ορίων ενός συστήματος και των βασικών λειτουργιών του, χρησιμοποιώντας «περιπτώσεις χρήσης» (use-cases) και «actors».
- την επεξήγηση της πραγματοποίησης των περιπτώσεων χρήσης με «διαγράμματα αλληλεπίδρασης».
- την αναπαράσταση μιας στατικής δομής ενός συστήματος χρησιμοποιώντας «διαγράμματα κλάσεων».
- τη μοντελοποίηση της συμπεριφοράς των αντικειμένων με «διαγράμματα καταστάσεων».
- τη μοντελοποίηση της εργασιακής ροής με «διαγράμματα δραστηριοτήτων».
- την αποκάλυψη της υλοποίησης της αρχιτεκτονικής με «διαγράμματα συστατικών» και «ανάπτυξης».
- την επέκταση της λειτουργικότητας με «στερεότυπα».

Τα διαγράμματα κλάσεων της UML ορίζουν γεωμετρικά σχήματα ως συμβολισμούς για τα αντικείμενα, τις κλάσεις και τις διασυνδέσεις, ενώ διαφόρων τύπων γραμμές χρησιμοποιούνται για να συνδέουν αυτά τα σχήματα και να υποδηλώνουν έτσι τον τρόπο που κληρονομούν, συνεργάζονται ή εξαρτώνται μεταξύ τους. Τα αντικείμενα της ίδιας κλάσης αναπαριστώνται με ένα μόνο γεωμετρικό σχήμα. Όταν ένα αντικείμενο χρησιμοποιεί κώδικα κάποιας άλλης κλάσης (π.χ. καλώντας μία μέθοδό της), σύμφωνα με το πρότυπο της UML υπάρχει μία «εξάρτηση» (dependency) μεταξύ τους η οποία αναπαρίσταται με μία διακεκομμένη γραμμή. Αυτή η εξάρτηση μπορεί να είναι «συσχέτιση» (association), ένας τύπος εξάρτησης που υπονοεί πραγματική συνύπαρξη στη μνήμη στιγμιοτύπων των συμμετεχόντων κλάσεων κατά τον χρόνο εκτέλεσης, «συνάθροιση» (aggregation), ένας τύπος συσχέτισης ο οποίος σημαίνει ότι το ένα αντικείμενο μπορεί να περιέχει στιγμιότυπα της άλλης κλάσης ως γνωρίσματα του, ή «σύνθεση» (composition), ένας πιο ισχυρός τύπος συνάθροισης που υπονοεί πως ο χρόνος ζωής των αντικειμένων είναι κοινός (δημιουργούνται και καταστρέφονται στη μνήμη ταυτόχρονα). Καθεμία από αυτές τις σχέσεις συμβολίζεται οπτικά με έναν διαφορετικό τύπο γραμμής μεταξύ των συμμετεχόντων κλάσεων, ενώ μπορεί να υπάρχουν και εξαρτήσεις οι οποίες δεν είναι καν συσχετίσεις (π.χ. όταν ένα αντικείμενο καλεί μία στατική μέθοδο κάποιας κλάσης).

Τί είναι ένα Mobile App

Μια φορητή εφαρμογή (ή αλλιώς mobile app) είναι μια εφαρμογή λογισμικού σχεδιασμένη να τρέχει σε smartphone, υπολογιστές tablet και άλλες φορητές συσκευές. Είναι διαθέσιμες στο κοινό μέσω πλατφορμών διανομής εφαρμογών, οι οποίες συνήθως λειτουργούν από τον ιδιοκτήτη του φορητού λειτουργικό συστήματος, όπως το Apple App Store, Google Play, BlackBerry App World.Τα Mobile apps, αρχικά είχαν στόχο την προσφορά στην γενική παραγωγικότητα του κοινού και την ανάκτηση πληροφοριών, συμπεριλαμβανομένων εφαρμογών για e-mail, ημερολόγιο, κατάλογο επαφών, χρηματιστηριακές αγορές και πληροφορίες για τον καιρό. Ωστόσο, η δημόσια ζήτηση και η διαθεσιμότητα των εργαλείων ανάπτυξης οδήγησε με γρήγορους ρυθμούς σε επέκταση και άλλων κατηγοριών, όπως παιχνίδια, αυτοματισμούς εργοστασίων, GPS και location-based υπηρεσίες, banking, εξέλιξη παραγγελιών, καθώς και στις αγορές εισιτηρίων.

Πώς ξεκίνησαν όλα.

Το Motorola DynaTAC 8000X ήταν το πρώτο εμπορικά διαθέσιμο κινητό τηλέφωνο. Πρωτοβγήκε στην αγορά το 1983, με διαστάσεις 33*4,5*9 εκατοστά, βάρος λίγο παραπάνω από το 1 κιλό και τιμή $3.995 συν τα μηνιαία πάγια και χρεώσεις ανά λεπτό, επέτρεπε κλήσεις συνολικής διάρκειας μισής ώρας. Αυτό, όπως και τα υπόλοιπα επερχόμενα πρώτα κινητά, δεν μπορούν να θεωρηθούν συσκευές πολλών δυνατοτήτων, παρόλο που οι περισσότερες συσκευές νέας γενιάς έχουν κληρονομήσει πολλά πλήκτρα από αυτά (SEND, END, CLEAR). Αντίθετα περιοριζόταν στην λήψη και πραγματοποίηση κλήσεων και στην διαχείριση επαφών μέσω μιας απλής εφαρμογής η οποία συνήθως ήταν δύσχρηστη.

Motorola DynaTAC 8000X: Το πρώτο κινητό διαθέσιμο στην αγορά

Τα κινητά τηλέφωνα πρώτης γενιάς σχεδιάζονταν και αναπτύσσονταν από τις ίδιες τις κατασκευάστριες εταιρίες των συσκευών. Ο ανταγωνισμός ήταν σκληρός και τα μυστικά του εμπορίου επισφαλώς σφραγισμένα. Οι κατασκευαστές δεν ήθελαν να εκθέσουν τα "εσωτερικά" των συσκευών τους και έτσι συνήθως η ανάπτυξη του λογισμικού τους γινόταν σε πολύ στενούς κύκλους. Οι προγραμματιστές της εποχής, αν δεν ανήκαν σε αυτούς τους κύκλους, δεν είχαν την ευκαιρία να γράψουν εφαρμογές για τις συσκευές αυτές, σε αντίθεση με σήμερα.
Αυτή ήταν η περίοδος που είχαν κάνει την εμφάνισή τους για πρώτη φορά τα παιχνίδια σε κινητά τηλέφωνα. Η Nokia έγινε γνωστή βάζοντας το βιντεο-παιχνίδι Snake από την δεκαετία του '70, σε μερικές από τις πρώτες μονόχρωμες συσκευές της. Ακολούθησαν και άλλοι κατασκευαστές εγκαθιστώντας παιχνίδια, όπως τα Pong, Tetris και Τρίλιζα.
Αυτά τα πρώτα τηλέφωνα μπορεί να είχαν ψεγάδια όμως κατάφεραν κάτι πολύ σημαντικό: να αλλάξουν τον τρόπο με τον οποίο ο κόσμος σκεφτόταν για την επικοινωνία. Όσο οι τιμές τους πέφταν, οι μπαταρίες βελτιωνόταν και οι περιοχές κάλυψης διευρύνονταν, τόσο και πιο πολλοί άνθρωποι άρχισαν να έχουν πάνω τους τέτοιες συσκευές χειρός. Σύντομα τα κινητά τηλέφωνα έπαψαν να είναι μια καινοτομία στον τρόπο επικοινωνίας.
Οι ανάγκες των πελατών, με τον καιρό, έφερναν πιέσεις για περισσότερες καινοτομίες και παιχνίδια, όμως υπήρχε ένα βασικό πρόβλημα. Οι κατασκευαστές δεν είχαν τα κίνητρα και τους πόρους για να φτιάξουν κάθε εφαρμογή που ζητούσαν οι χρήστες. Έπρεπε να βρουν έναν τρόπο να τους παρέχουν μια "πύλη" που να προσφέρει υπηρεσίες ψυχαγωγίας και πληροφοριών χωρίς να επιτρέπουν την πρόσβαση στον προγραμματισμό των συσκευών. Για την επίλυση, λοιπόν, αυτού του προβλήματος, έγινε απόπειρα επιστράτευσης του Διαδικτύου.

Wireless Application Protocol (WAP)

Όπως αποδείχτηκε, η απευθείας πρόσβαση στο Διαδίκτυο δεν μπόρεσε να ταιριάξει στις προδιαγραφές των τότε φορητών συσκευών. Οι ιστοσελίδες ήταν γεμάτες χρώματα, πληθώρα κειμένων, εικόνων και άλλων ειδών πολυμέσων. Η βάση αυτών των ιστοσελίδων ήταν το JavaScript και το Flash, καθώς και άλλες τεχνολογίες που σκοπό είχαν να βελτιστοποιήσουν την εμπειρία του χρήστη και συχνά ήταν σχεδιασμένες για να τρέχουν σε αναλύσεις 800*600 pixel και υψηλότερες. Όταν το πρώτο clamshell κινητό, το Motorola StarTAC, λανσαρίστηκε στην αγορά το 1996, είχε οθόνη LCD, μόλις, 10 ψηφίων. Παράλληλα η Nokia είχε κυκλοφορήσει ένα από τα πρώτα slider κινητά, το 8110, το οποίο μπορούσε να προβάλει κείμενο 13 χαρακτήρων ανά γραμμή. Τα κινητά αυτά, με τις εξαιρετικά μικρές οθόνες χαμηλών αναλύσεων και τον περιορισμένο αποθηκευτικό τους χώρο και υπολογιστικές δυνατότητες, δεν μπορούσαν να διαχειριστούν απαιτητικές διεργασίας με πολλά δεδομένα, με τον τρόπο που μέχρι στιγμής έκανε ένας συμβατικός browser. Εκτός αυτού, το κόστος για το απαιτούμενο bandwidth ήταν απαγορευτικό. Το Πρωτόκολλο Ασύρματων Εφαρμογών (WAP) έκανε την εμφάνισή του απευθυνόμενο στα προηγούμενα προβλήματα. Με απλά λόγια, το WAP ήταν μια ξεγυμνωμένη έκδοση του βασικού πρωτοκόλλου του Διαδικτύου, HTTP. Σε αντίθεση με τους συμβατικούς browser, οι WAP-browser

είχαν σχεδιαστεί για να τρέχουν μέσα από την μνήμη και για τους περιορισμούς του bandwidth του τηλεφώνου. Ιστοσελίδες τρίτων προσέφεραν σελίδες WAP γραμμένες σε γλώσσα mark-up που ονομαζόταν Wireless Markup Language (WML) που προβαλλόταν στον WAP-browser ενός
κινητού. Η πλοήγηση μέσα σε αυτές γινόταν με παρόμοιο τρόπο όπως γινόταν και σε αυτές του Διαδικτύου, απλά η φιλοσοφία τους ήταν πολύ πιο απλή.
Η λύση που έφερε το WAP κατάφερε να πάρει να την πίεση για περισσότερες εφαρμογές ,από πλευράς πελατών, από τις κατασκευάστριες εταιρίες και να την μετατοπίσει στους 7 προγραμματιστές, γεγονός που καθιστούσε αυτούς υπεύθυνους για την παροχή περισσότερων υπηρεσιών στους πελάτες. Έτσι για πρώτη φορά, οι προγραμματιστές είχαν την ευκαιρία να αναπτύξουν εφαρμογές για χρήστες κινητών τηλεφώνων.

Οι περισσότερες από τις πρώτες WAP σελίδες ήταν επεκτάσεις ήδη γνωστών ιστοσελίδων, όπως CNN.com, ESPN.com, που απλά έψαχναν νέους τρόπους να πάρουν την πρωτιά από τους
ανταγωνιστές τους. Ως αποτέλεσμα, οι χρήστες είχαν πρόσβαση σε Ειδήσεις, Χρηματιστήριο και αποτελέσματα αγώνων από το κινητό τους. Ωστόσο, η διαφήμιση WAP εφαρμογών ήταν δύσκολη, καθώς δεν υπήρχε κάποιος έτοιμος μηχανισμός. Μερικές από τις πιο πετυχημένες εφαρμογές ήταν εφαρμογές προσωποποίησης του κινητού, που για πρώτη φορά επέτρεπαν στους χρήστες να
κατεβάζουν ringtone και wallpaper.Από την άλλη πλευρά, ένας WAP browser συνήθως ήταν δύσχρηστος και αργός. Τα URL ήταν μεγάλα και δύσκολα να τυπωθούν με τα πληκτρολόγια των κινητών και οι σελίδες δύσκολες στην πλοήγηση και απευθυνόταν σε όλα τα είδη οθονών με ακριβώς τον ίδιο τρόπο με αποτέλεσμα η εμπειρία του χρήστη να είναι μέτρια και καθόλου ελκυστική. Αυτό σε συνδυασμό με τις εμπορικές ελλείψεις οδήγησαν το WAP σε αποτυχία στις περισσότερες περιοχές του κόσμου. Αποτυχία έγινε όμως και για τους προγραμματιστές, όταν οι πάροχοι δικτύων τηλεφωνίας, για διαφημιστικούς λόγους, άρχισαν να επιτρέπουν την πρόσβαση μόνο σε συγκεκριμένες σελίδες, με αποτέλεσμα οι περισσότερες εφαρμογές που ανέπτυσσαν να μην φτάνουν στο κοινό.

Φορητές Πλατφόρμες

Καθώς οι μνήμες γινόταν φθηνότερες, οι μπαταρίες καλύτερες και καθώς πολλά PDA και άλλες ενσωματωμένες συσκευές άρχισαν να τρέχουν συμπυκνωμένες εκδόσεις διάφορων γνωστών λειτουργικών συστημάτων, οι κατασκευάστριες εταιρείες κινητών συνειδητοποίησαν πως θα πρέπει αν αλλάξουν τακτική για να παραμείνουν στην αγορά. Με αυτό το σκεπτικό, εμφανίστηκαν στο
προσκήνιο διάφορες φορητές πλατφόρμες, για τις οποίες οι προγραμματιστές γράφουν ακόμη και σήμερα εφαρμογές. Νέα smartphone άρχισαν να τρέχουν Palm (Garnet OS) και RIM BlackBerry
OS. Η Sun Microsystems έβγαλε στο τραπέζι την δημοφιλή πλατφόρμα της Java και αναφάνηκε η J2ME (Java Micro Edition). Άλλες πλατφόρμες όπως το Symbian OS, αναπτύχθηκαν από κατασκευάστριες κινητών όπως Nokia, Sony Ericsson, Motorola και Samsung, ενώ το Apple iPhone OS προστέθηκε στην λίστα το 2008. Μία τέτοια πλατφόρμα είναι και το Android.

Λίγα λόγια για το Android

Το Android είναι λειτουργικό σύστημα για συσκευές κινητής τηλεφωνίας και άλλες συσκευές το

οποίο τρέχει τον πυρήνα του λειτουργικού Linux. Αρχικά αναπτύχθηκε από την Google και αργότερα από την Open Handset Alliance. Επιτρέπει στους κατασκευαστές λογισμικού να συνθέτουν κώδικα με την χρήση της γλώσσας προγραμματισμού Java, ελέγχοντας την συσκευή
μέσω βιβλιοθηκών λογισμικού ανεπτυγμένων από την Google.
Η πρώτη παρουσίαση της πλατφόρμας Android έγινε στις 5 Νοεμβρίου 2007, παράλληλα με την ανακοίνωση της ίδρυσης του οργανισμού Open Handset Alliance, μιας κοινοπραξίας 48 τηλεπικοινωνιακών εταιριών, εταιριών λογισμικού καθώς και κατασκευής hardware, οι οποίες είναι αφιερωμένες στην ανάπτυξη και εξέλιξη ανοιχτών προτύπων στις συσκευές κινητής τηλεφωνίας. Η Google δημοσίευσε το μεγαλύτερο μέρος του κώδικα του Android υπό τους όρους της Apache
License, μιας ελεύθερης άδειας λογισμικού.

Ιστορία

Το 2005 η Google αγόρασε την αρχική εταιρία Android Inc. Που βρισκόταν στο Palo Alto της California. Κάποια από τα άτομα που πήγαν εκεί για να δουλέψουν για την Google ήταν ο Rick Miner, ο συνιδρυτής της Wildfire Communications, ο Andy Rubin, συνιδρυτής της Danger, ο Chris
White, μηχανικός στο WebTv και ο Nick Sears ο πρώην αντιπρόεδρος της T-Mobile. Όλοι αυτοί ξεκίνησαν να αναπτύσσουν ένα Linux-based λειτουργικό σύστημα για κινητά τηλέφωνα και άλλες κινητές συσκευές. Ο στόχος τους ήταν ένα ευέλικτο λειτουργικό σύστημα καθώς και αναβαθμίσιμο.
Στις 5 Νοεμβρίου του 2007, διάφορες εταιρίες συνεργάστηκαν για να δημιουργήσουν την Open Handset Alliance. Οι πιο γνωστές εταιρίες που συνεργάστηκαν είναι:

• Motorola, Inc.
- HTC Corporation

• Intel Corporation
• Qualcomm
• Sprint Nextel
• T- Mobile
• NVIDIA Corporation
• Toshiba
• Samsung Electronics
• LG Electronics, Inc.
• Vodafone
• Sony Ericsson
• Texas Instruments

Ξεκίνησαν με στόχο να αναπτύξουν open standards για mobile συσκευές και έτσι το πρώτο τους προϊόν ήταν η πλατφόρμα Android που ήταν χτισμένη πάνω στον πυρήνα του Linux. Σχεδιάστηκε ώστε να εξυπηρετεί τις εταιρίες κινητής τηλεφωνίας, τους κατασκευαστές συσκευών και τους
προγραμματιστές εφαρμογών.
Το Android SDK παρουσιάστηκε ανεπίσημα για πρώτη φορά τον Νοέμβριο του 2007. Τον

Σεπτέμβριο του 2008 η T-Mobile ανακοίνωσε την διαθεσιμότητα του T-Mobile G1, του πρώτου Smartphone βασισμένο στο λειτουργικό Android. Λίγες μέρες αργότερα η Google ανακοίνωσε την 10
διαθεσιμότητα του Android SDK Release Candidate 1.0. Το Android ήταν διαθέσιμο σαν opensource λογισμικό από τον Οκτώβριο του 2008. Μέσω του Apache, ιδιωτικές εταιρίες μπορούσαν
να προσθέσουν τις δικές τους εφαρμογές και επεκτάσεις και να τις πουλήσουν χωρίς να υποχρεούνται να τις υποβάλλουν στην open-source κοινότητα. Στα τέλη του 2008 η Google ανακοίνωσε μια συσκευή με το όνομα Android Dev Phone 1 που είχε την δυνατότητα να τρέχει Android εφαρμογές χωρίς να είναι άμεσα συνδεδεμένη με κάποιο δίκτυο κινητής τηλεφωνίας. Ο
στόχος της συσκευής αυτής ήταν να επιτρέψει στους προγραμματιστές να πειραματιστούν με μια πραγματική συσκευή με Android χωρίς κάποιο συμβόλαιο με εταιρία κινητής τηλεφωνίας.

Γιατί Ανάπτυξη σε Android

Το Android είναι μια ολοκληρωμένη, ανοιχτή και ελεύθερη πλατφόρμα για κινητά τηλέφωνα που περιλαμβάνει ένα λειτουργικό σύστημα (OS), το απαραίτητο ενδιάμεσο λογισμικό, βιλιοθήκες και βασικές εφαρμογές. Το Android SystemDevelopmentKit παρέχει στους προγραμματιστές όλα τα εργαλεία και APIs για να αρχίσουν να αναπτύσσουν λογισμικό για την πλατφόρμα Android χρησιμοποιώντας τη γλώσσα προγραμματισμού Java.

i) Λειτουργικότητα και ευελιξία

Το Android είναι μια μοναδική πλατφόρμα που επιτρέπει την ανάπτυξη εφαρμογών λογισμικού το οποίο εκμεταλλεύεται πλήρως τις δυνατότητες μιας συμβατής συσκευής. Για παράδειγμα, οι προγραμματιστές εφαρμογών είναι ελεύθεροι να δημιουργήσουν εφαρμογές που χρησιμοποιούν οποιαδήποτε από τις βασικές λειτουργίες του τηλεφώνου όπως η αποστολή SMS, τηλεφωνικές κλήσεις, τη λήψη φωτογραφιών, το GPS κτλ. Έτσι διευκολύνονται στην ανάπτυξη πιο περίπλοκου
και πιο πλούσια λειτουργικού λογισμικού. Αυτό το λειτουργικό σύστημα κινητών τηλεφώνων (ή άλλων μικρών φορητών συσκευών που συνδέονται στο διαδίκτυο) στηρίζεται στον ελεύθερο πυρήνα του Linux. Επιπλέον, η πλατφόρμα ανάπτυξης Android είναι μια πλατφόρμα multi tasking, πράγμα που σημαίνει ότι κάθε εφαρμογή μπορεί να τρέξει στο τηλέφωνο ταυτόχρονα κάποια άλλη χωρίς να επηρεαστεί η απόδοσή τους, και αυτό είναι καλύτερο από το να περιορίζεται σε μία εφαρμογή κάθε φορά. Το Android είναι μια πλατφόρμα ανοικτού κώδικα, πράγμα που σημαίνει ότι μπορεί εύκολα να επεκταθεί και να τροποποιηθεί για να συμβαδίζει και να υιοθετεί τις τελευταίες τεχνολογίες και εξελίξεις. Το γεγονός ότι και η πηγή της πλατφόρμας είναι ανοικτή διασφαλίζει ότι η ανάπτυξη το Android θα έχει συνεχή πρόοδο και θα εξελίσσεται αφού ένας μεγάλος αριθμός
ικανών android προγραμματιστών εργάζεται για τη δημιουργία ελεύθερων για χρήση προηγμένων εργαλείων λογισμικού.

ii) Πλήρης παραμετροποίηση

Δεν υπάρχει διαφορά μεταξύ των λειτουργιών / εφαρμογών οι οποίες είναι ενσωματωμένες στο τηλεφώνου απο τις εφαρμογές που δημιουργούνται και προστίθενται από τρίτους προγραμματιστές Android. Οι τελευταίες μπορούν και έχουν την ίδια πρόσβαση σε όλες τις κύριες λειτουργίες της συσκευής κάτι που επιτρέπει στους τελικούς χρήστες να απολαμβάνουν ένα ευρύ φάσμα εφαρμογών Android που μπορούν να χρησιμοποιηθούν για

τη σχεδόν απεριόριστους σκοπούς. Με συσκευές χτισμένες στην πλατφόρμα Android, οι χρήστες έχουν τη δυνατότητα να προσαρμόσουν πλήρως τη συσκευή τους ανάλογα με τις ανάγκες και τις απαιτήσεις τους. Τυχόν εφαρμογές ακόμα
και οι βασικές λειτουργίες μπορεί να τροποποιηθούν ή να αντικατασταθούν πλήρως απο άλλες. Για παράδειγμα, ο χρήστης μπορεί να χρησιμοποιήσει την επιθυμητή του εφαρμογή για να εμφανίσει τις φωτογραφίες που είναι αποθηκευμένες στο τηλέφωνό του, ή για να έχει πρόσβαση στην αλληλογραφία του.

iii) Διαδραστικότητα

Οι προγραμματιστές Android μπορούν να δημιουργήσουν πολύπλοκες καινοτόμες εφαρμογές με σχεδόν απεριόριστη λειτουργικότητα. Για παράδειγμα, μια εφαρμογή μπορεί να μεταδώσει τα δεδομένα από το κινητό σας με το διαδίκτυο (κάτι που μπορεί να περιλαμβάνει το ημερολόγιο σας και τις προγραμματισμένες εκδηλώσεις, λίστα με τις επαφές, τις φωτογραφίες σας και ακόμη και την τρέχουσα θέση σας, αλλά και παραγγελίες, τιμολόγια κτλ) και να λάβει όλα όσα μπορεί να χρειαστεί online και να εμφανίζονται στην οθόνη της συσκευής.

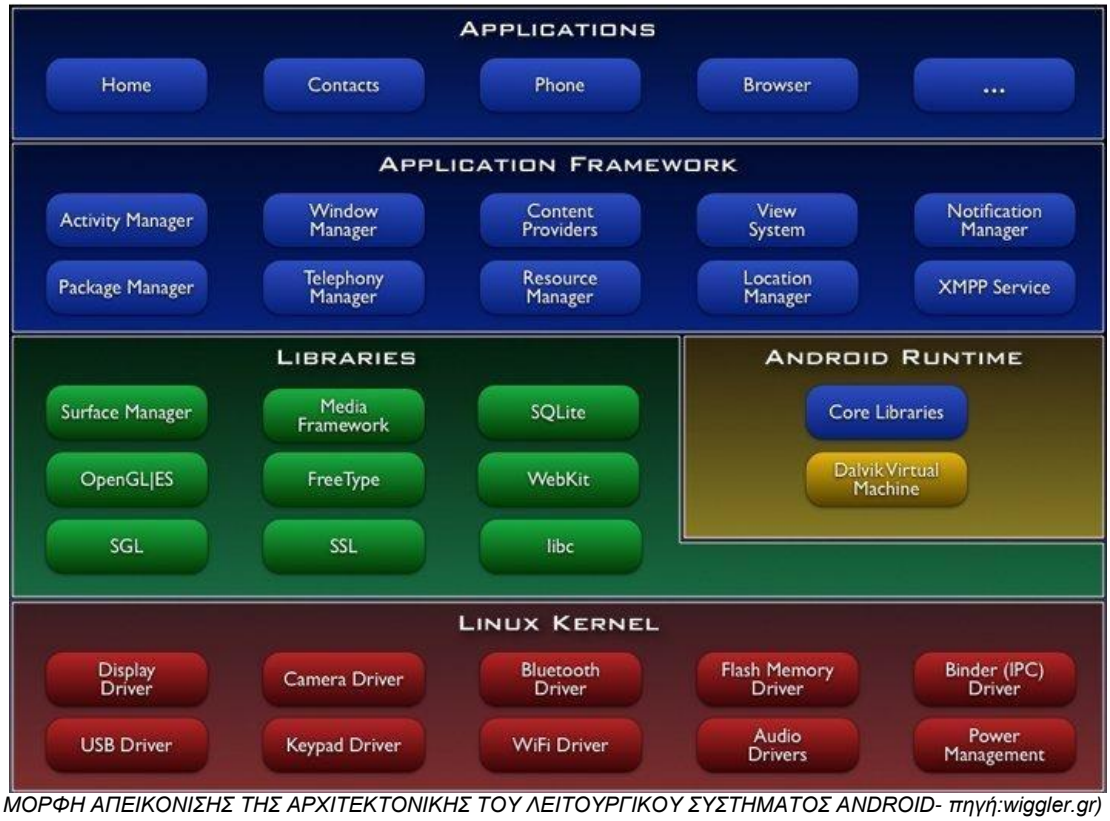
iv) Απλούστερη ανάπτυξη ε φαρμογών κ ινητών

Η πλατφόρμα παρέχει στο καθένα που ασχολείται με την ανάπτυξη εφαρμογών τη δυνατότητα χρησιμοποίησης μια μεγάλης ποικιλίας από βιβλιοθήκες και τα χρήσιμα εκείνα εργαλεία που μπορούν να χρησιμοποιηθούν για τη δημιουργία του πιο εξελιγμένου λογισμικού. Αυτή η ολοκληρωμένη δέσμη από έτοιμα εργαλεία αυξάνει σημαντικά την παραγωγικότητα των προγραμματιστών Android εφαρμογών και τους βοηθά να δημιουργήσουν εκπληκτικά πλούσιο λογισμικό γρηγορότερα και με λιγότερα λάθη.

Η αρχιτεκτονική του Android

“Η Αρχιτεκτονική του Android”

Όπως έχει ήδη αναφερθεί, το Android αποτελεί μία στοίβα λογισμικού. Βασική αρχή της αρχιτεκτονικής Android είναι η επαναχρησιμοποίηση κώδικα, με τον οποίο δίνεται η δυνατότητα δημοσίευσης και μοιράσματος δραστηριοτήτων, υπηρεσιών και δεδομένων με άλλες εφαρμογές, πάντοτε στα πλαίσια ασφαλείας. Με αυτόν τον τρόπο οδηγούμαστε στην επέκταση και βελτιστοποίηση ήδη υπάρχοντων εφαρμογών, καθώς και τη δημιουργία νέων. Το Android δεν είναι απλά ένα λειτουργικό σύστημα, είναι μία στοιβα λογισμικού που αποτελείται από το λειτουργικό σύστημα , τις υπηρεσίες διασύνδεσης με τις εφαρμογές (middleware) και τις κύριες εφαρμογές (core) που είναι ο e-mail client, η εφαρμογή διαχείρισης SMS, το ημερολόγιο, ο browser, η εφαρμογή διαχείρισης επαφών και άλλες που είναι προεγκατεστημένες.



ΜΙΑ ΜΟΡΦΗ ΑΠΕΙΚΟΝΙΣΗΣ ΤΗΣ ΑΡΧΙΤΕΚΤΟΝΙΚΗΣ ΤΟΥ ΛΕΙΤΟΥΡΓΙΚΟΥ ΣΥΣΤΗΜΑΤΟΣ ANDROID- πηγή:wiggler.gr)

Η αρχιτεκτονική του Android παρέχει:

- Κανένα περιορισμό στις εφαρμογές.
- Παράλληλη λειτουργία πολλαπλών εφαρμογών.
- Ισότητα για κάθε εφαρμογή.

Η στοίβα του Android αποτελείται από πέντε επίπεδα, τα οποία είναι τα εξής:

Α) Επίπεδο Εφαρμογών (Applications)

ΑΠΕΙΚΟΝΙΣΗ ΤΟΥ ΕΠΙΠΕΔΟΥ ΕΦΑΡΜΟΓΩΝ, ΟΠΩΣ ΑΝΑΠΑΡΙΣΤΑΤΑΙ ΣΤΗ ΣΤΟΙΒΑ

ΛΟΓΙΣΜΙΚΟΥ ΤΗΣ ΑΡΧΙΤΕΚΤΟΝΙΚΗΣ ΤΟΥ ΛΕΙΤΟΥΡΓΙΚΟΥ ΣΥΣΤΗΜΑΤΟΣ ANDROID – πηγή: pcmonster.gr

Μερικές βασικές εφαρμογές που περιλαμβάνονται στο σύστημα Android όπως οι e-mail clients, πρόγραμμα για SMS μηνύματα, ημερολόγιο, χάρτες (Google Maps), περιηγητής ιστού και πρόγραμμα για δομημένη αποθήκευση των επαφών, χρησιμοποιούν τη Java ως γλώσσα προγραμματισμού τους.

Β) Επίπεδο Πλαισίου Εφαρμογών (Application Framework)

ΑΠΕΙΚΟΝΙΣΗ ΤΟΥ ΕΠΙΠΕΔΟΥ ΠΛΑΙΣΙΟΥ ΕΦΑΡΜΟΓΩΝ, ΟΠΩΣ ΑΥΤΗ ΑΝΑΠΑΡΙΣΤΑΤΑΙ ΣΤΗΝ ΑΠΕΙΚΟΝΙΣΗ ΤΗΣ ΣΤΟΙΒΑΣ ΛΟΓΙΣΜΙΚΟΥ ΤΟΥ ΛΕΙΤΟΡΓΙΚΟΥ ΣΥΣΤΗΜΑΤΟΣ ANDROID – πηγή: pcmonster.gr

Αυτό το επίπεδο μας παρέχει υψηλού επιπέδου δομικές μονάδες που δίνουν τη δυνατότητα στους προγραμματιστές να κατασκευάσουν πλούσιες και καινοτόμες εφαρμογές. Με τη δυνατότητα των προγραμματιστών να εκμεταλλεύονται το hardware των συσκευών, έχουν πρόσβαση σε υπηρεσίες εντοπισμού θέσης, δυνατότητα να θέτουν χρονοδιακόπτες για εμφάνιση ειδοποιήσεων, πλήρη πρόσβαση στο ίδιο πλαίσιο από APIs που έχουν βασικές εφαρμογές του Android, καθώς και πολλά άλλα. Η διαμόρφωση της αρχιτεκτονικής αυτού του επιπέδου γίνεται με τέτοιο τρόπο, έτσι ώστε κάθε εφαρμογή να μπορεί να χρησιμοποιεί τις δυνατότητες μίας άλλης εφαρμογής και ταυτόχρονα ο χρήστης να επεμβαίνει στα συστατικά καθεμίας από αυτές.

Ο τρόπος με τον οποίο είναι οργανωμένα τα API στο επίπεδο αυτό ακολουθεί τη λογική του διαχειριστή (manager).

Τα σημαντικότερα δομικά στοιχεία του πλαισίου εφαρμογών είναι:

- **Διαχειριστής περιεχομένου (Content Manager)**

Ο οποίος επιτρέπει στα δεδομένα να διαμοιράζονται από μία εφαρμογή σε μία άλλη εφαρμογή. Τέτοια δεδομένα μπορεί να είναι οι επαφές του χρήστη και οι βάσεις δεδομένων των εφαρμογών.

- **Διαχειριστής πόρων (Resource Manager)**

Ο οποίος επιτρέπει την πρόσβαση στους πόρους, όπως strings, εικόνες, layout files,πίνακες χαρακτήρων. Οι πόροι είναι οτιδήποτε υπάρχει σε ένα πρόγραμμα και δεν είναι κώδικας.

- **Διαχειριστής τοποθεσίας (Location Manager)**

Με τον οποίο έχουμε τη δυνατότητα να γνωρίζουμε που βρίσκεται η συσκευή που χρησιμοποιεί το λογισμικό Android ανά πάσα στιγμή.

- **Διαχειριστής ειδοποιήσεων (Notification Manager)**

Ο οποίος επιτρέπει στο χρήστη να ενημερώνεται για γεγονότα που συμβαίνουν. Τα γεγονότα εμφανίζονται στη μπάρα κατάστασης (status bar),με τη μορφή toast μηνυμάτων τα οποία εμφανίζονται στο κάτω μέρος της οθόνης, με τη δόνηση του κινητού και την ενεργοποίηση της οθόνης.

- **Διαχειριστής δραστηριοτήτων (Activity Manager)**

Ο οποίος ελέγχει το χρόνο ζωής των εφαρμογών και επιτρέπει στο χρήστη να πλοηγείται σε προηγούμενες οθόνες εφαρμογών.

- **Σύστημα προβολών (View System)**

Αποτελεί ένα σύνολο αντικειμένων GUI τα οποία χρησιμοποιούνται στο σχεδιασμό μίας εφαρμογής. Χαρακτηριστικά παραδείγματα προβολών αποτελούν τα πεδία εισαγωγής κειμένου, το πλέγμα, οι λίστες, κλπ. Η κλάση View παρέχει ένα σύνολο μεθόδων, οι οποίες εκτελούνται όταν ανιχνευθεί κάποιο συμβάν από το χρήστη.

- **Ικανότητα δικτύωσης (Connectivity Manager)**

Παρέχει πληροφορίες για τις δυνατές συνδέσεις μίας συσκευής, καθώς και την κατάσταση κάθε σύνδεσης.

Χαρακτηριστικά παραδείγματα πακέτων για τη διαχείριση συνδέσεων και μεταφορά δεδομένων σε διαφορα δίκτυα είναι:

- ***Java.net**** -> βιβλιοθήκες της Java για τη δημιουργία συνδέσεων με τη χρήση του πρωτοκόλλου HTTP.

- ***Android.net**->** βιβλιοθήκες του Android για την επέκταση των δυνατοτήτων των πακέτων της java.net.

- ***Android.net.http**->** βιβλιοθήκες του Android για τη διαχείριση ασφαλών συνδέσεων με τα πρωτόκολλα HTTP.

- ***Org.apache**->** πακέτα του Apache Foundation για HTTP συνδέσεις.,

- ***Android.telephony**->** πακέτα του Android για πρόσβαση σε δίκτυα τηλεφωνίας (GSM και DMA).

- ***Android.net.wifi*->** πακέτα android για σύνδεση με τη χρήση Wifi.

Γ) Επίπεδο Χρόνου Εκτέλεσης Εφαρμογής (Android Runtime)

ΑΠΕΙΚΟΝΙΣΗ ΤΟΥ ΕΠΙΠΕΔΟΥ ΧΡΟΝΟΥ ΕΚΤΕΛΕΣΗΣ ΕΦΑΡΜΟΓΗΣ, ΟΠΩΣ ΑΥΤΗ ΑΠΕΙΚΟΝΙΖΕΤΑΙ ΣΤΗΝ ΑΝΑΠΑΡΑΣΤΑΣΗ ΤΗΣ ΣΤΟΙΒΑΣ ΛΟΓΙΣΜΙΚΟΥ ΤΟΥ ΛΕΙΤΟΥΡΓΙΚΟΥ ΣΥΣΤΗΜΑΤΟΣ ANDROID –πηγή: pcmonster.gr

Αυτό το επίπεδο αποτελείται από ένα σύνολο βασικών βιβλιοθηκών και την Dalvik Virtual Machine.

Η Dalvik Virtual Machine είναι μια εικονική μηχανή Java για φορητές συσκευές, από την Google. Σε αυτή εκτελείται ο κώδικας bytecode των εφαρμογών. Κάθε εφαρμογή δεν έχει επαφή με άλλη, παρόλο που εκτελούνται ταυτόχρονα. Με τη χρήση της Dalvik εκτελούνται αρχεία
.dex, τα οποία βρίσκονται σε συμπιεσμένη μορφή, εξοικονομώντας χώρο στη μνήμη και δίνοντας τη δυνατότητα να τρέχουν πολλές εικονικές μηχανές ταυτόχρονα στο σύστημα. Χαρακτηριστικό παράδειγμα μηχανισμού του Android για τη σωστή διαχείριση μνήμης είναι το garbage collector και το Zygote.
Τέλος, ένα σημαντικό χαρακτηριστικό του Android Runtime είναι η ύπαρξη του JIT (just in time) μεταφραστή, που χρησιμοποιεί τη μετάφραση από bytecodes σε κώδικα μηχανής ώστε να αυξηθεί η ταχύτητα εκτέλεσης των

Δ) Επίπεδο Βιβλιοθηκών (Libraries)

ΑΠΕΙΚΟΝΙΣΗ ΤΟΥ ΕΠΙΠΕΔΟΥ ΒΙΒΛΙΟΘΗΚΩΝ, ΟΠΩΣ ΑΥΤΗ ΑΠΕΙΚΟΝΙΖΕΤΑΙ ΣΤΗΝ ΑΝΑΠΑΡΑΣΤΑΣΗ ΤΗΣ ΣΤΟΙΒΑΣ ΛΟΓΙΣΜΙΚΟΥ ΤΟΥ ΛΕΙΤΟΥΡΓΙΚΟΥ ΣΥΣΤΗΜΑΤΟΣ ANDROID – πηγή: pcmonster.gr

Οι βιβλιοθήκες είναι γραμμένες στις γλώσσες προγραμματισμού C και C++ και μεταγλωττίστηκαν για τη συγκεκριμένη αρχιτεκτονική υλικού που χρησιμοποιείται από διάφορα συστήματα Android. Οι κύριες βιβλιοθήκες που συναντάμε στο Android είναι οι ακόλουθες:

1. ***System C library:*** πρόκειται για μία ενσωμάτωση της standard βιβλιοθήκης συστήματος της C η οποία έχει τροποποιηθεί ώστε να είναι κατάλληλη για κινητές συσκευές που βασίζονται στο Linux. Βασικός της στόχος είναι η μείωση της απαιτούμενης μνήμης.

2. ***Surface Manager*****:** αναφέρεται στο υποσύστημα προβολής συνθέτοντας

πολυδιάστατα επίπεδα γραφικών που προέρχονται από πολλές εφαρμογές. Είναι ο διαχειριστής σχεδιαστικών επιφανειών. Μία σχεδιαστική επιφάνεια είναι η οθόνη της συσκευής ή η περιοχή της μνήμης που έχει δεσμευθεί για να κρατήσει τα δεδομένα προς απεικόνιση.Ο διαχειριστής σχεδιαστικών επιφανειών σε συνδυασμό με το διαχειριστή παραθύρων (window manager), αποφασίζουν για τη σειρά σχεδίασης των παραθύρων των εφαρμογών έτσι ώστε να υπάρχει σωστή αλληλουχία ανάλογα με τις επιλογές του χρήστη. Επιπλέον, καθορίζουν στην περίπτωση που κάποια παράθυρα είναι ημιδιάφανα, τότε αυτά που βρίσκονται από πίσω τους να είναι μερικώς ορατά.

3. ***SQLite*:** πρόκειται για μία πολύ ισχυρή βάση δεδομένων. Αφορά μια μηχανή ανοικτού κώδικα διαχείρισης σχεσιακών βάσεων δεδομένων που παρέχει ένα μηχανισμό για ανάγνωση, εγγραφή, τροποποιήση και άλλες διαδικασίες στα δεδομένα.

4. ***Βιβλιοθήκες 3D*:** οι βιβλιοθήκες αυτές χρησιμοποιούν τρισδιάστατη επιτάχυνση υλικού και λογισμικού που βασίζεται στα APIs του OpenGL ES 1.

5. ***Βιβλιοθήκες πολυμέσων*:** υποστηρίζει την αναπαραγωγή και την εγγραφή πολλαπλών μέσων εικόνας και ήχου. Αποτελεί το λεγόμενο Media Framework του Android που είναι υπεύθυνο για τη σωστή αναπαραγωγή ήχου και βίντεο σε μία συσκευή. Υποστηρίζει την κωδικοποίηση του ήχου και του βίντεο ή τα διάφορα format AAC, AMR, MP2, MP3, WMA, MPEG1, MPEG2, MPEG4, VC-1,VC-2, VC-3. Βασικό μέρος αποτελεί η διαχείριση μνήμης που απαιτείται κατά τη διαδικασία κωδικοποίησης ή αποκωδικοποίησης.

6. ***SGL:*** πρόκειται για μία μηχανή που προσφέρει τη δυνατότητα σχεδίασης δυσδιάστατων γραφικών.

7. ***FreeType*:** προσφέρει ευκρίνεια στα γραφικά που χρησιμοποιούνται στα bitmaps και στις γραμματοσειρές των εφαρμογών. Συγκεκριμένα, η ποιότητα απεικόνισης μίας διανυσματικής γραμματοσειράς δε μειώνεται καθώς αυξάνεται το μέγεθός της.

8. ***LibWebCore*:** υποστηρίζει την πλοήγηση στο διαδίκτυο και χρησιμοποιείται από τον browser του Android και τις Web Views που είναι ενσωματωμένες στις εφαρμογές.

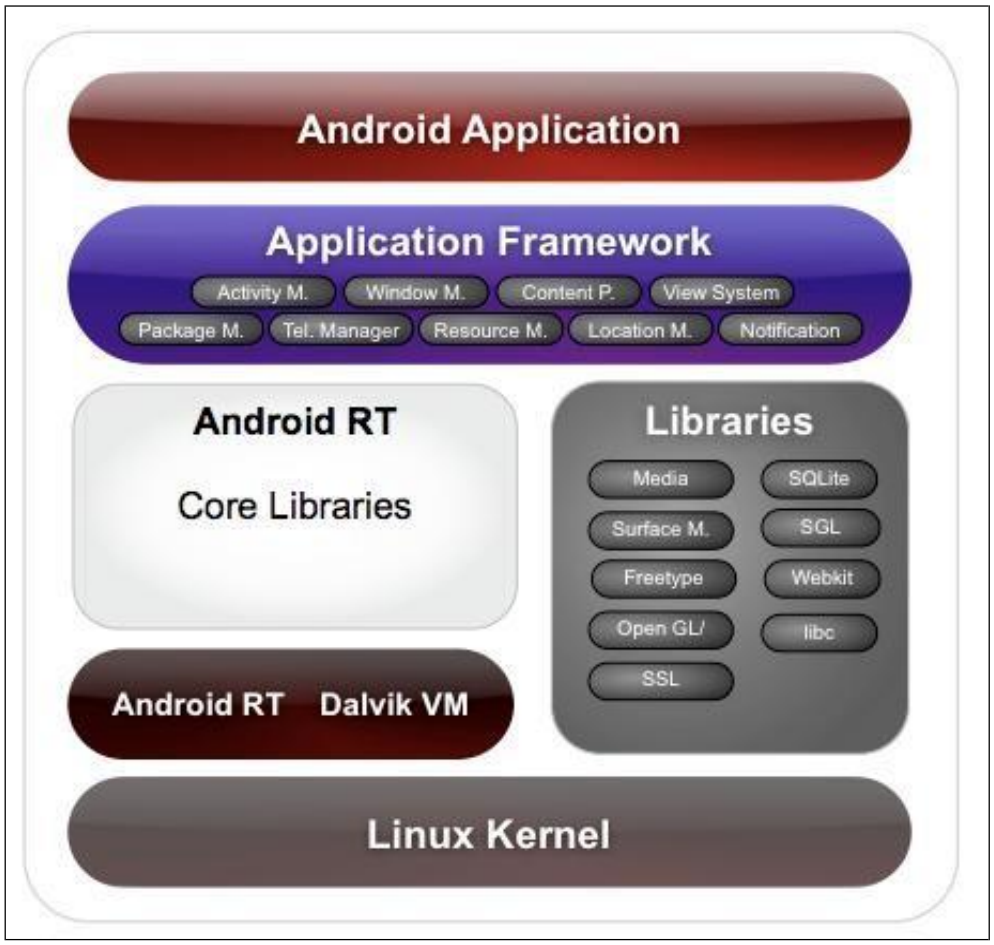
9. ***WebKit:*** είναι η μηχανή διάταξης γραφικών που δίνει τη δυνατότητα στον πλοηγητή του διαδικτύου να απεικονίσει τις σελίδες που ο χρήστης επισκέπτεται με γραφικό τρόπο, με τον κώδικα HTML που του παρέχει ο ιστότοπος.

Ε) Επίπεδο Πυρήνα του Linux

ΑΠΕΙΚΟΝΙΣΗ ΤΟΥ ΕΠΙΠΕΔΟΥ ΠΥΡΗΝΑ ΤΟΥ LINUX, ΟΠΩΣ ΑΥΤΟΣ ΑΠΕΙΚΟΝΙΖΕΤΑΙ ΣΤΗΝ ΑΝΑΠΑΡΑΣΤΑΣΗ ΤΗΣ ΣΤΟΙΒΑΣ ΛΟΓΙΣΜΙΚΟΥ ΤΟΥ ΛΕΙΤΟΥΡΓΙΚΟΥ ΣΥΣΤΗΜΑΤΟΣ ANDROID-πηγή: pcmonster.gr

Το λογισμικό του Android βασίζεται στον πυρήνα Linux, έκδοση 2.6(και στην έκδοση 3.0.1 για το Android 4.0) και του επιτρέπει να μπορεί να χρησιμοποιηθεί σε ένα μεγάλο εύρος πλατφορμών στο μέλλον. Ειδικότερα χρησιμοποιείται για τη διαχείριση μνήμης, τη διαχείριση διεργασιών, τις λειτουργίες δικτύου, ασφάλεια του λειτουργικού συστήματος και οδηγίες

υλικού (hardware drivers). Ο πυρήνας του Linux δρα στην περίπτωση του Android ως το επονομαζόμενο επίπεδο αφαίρεσης υλικού (hardware abstraction layer). Στο επίπεδο αυτό ανήκουν οι οδηγοί για την οθόνη, για τις κάρτες μνήμης, για τον ήχο, για το υποσύστημα ασύρματης σύνδεσης στο διαδίκτυο (Wi-Fi), για την κάμερα, καθώς και όποιο άλλο περιφερειακό συναντάμε στις συσκευές κινητών.



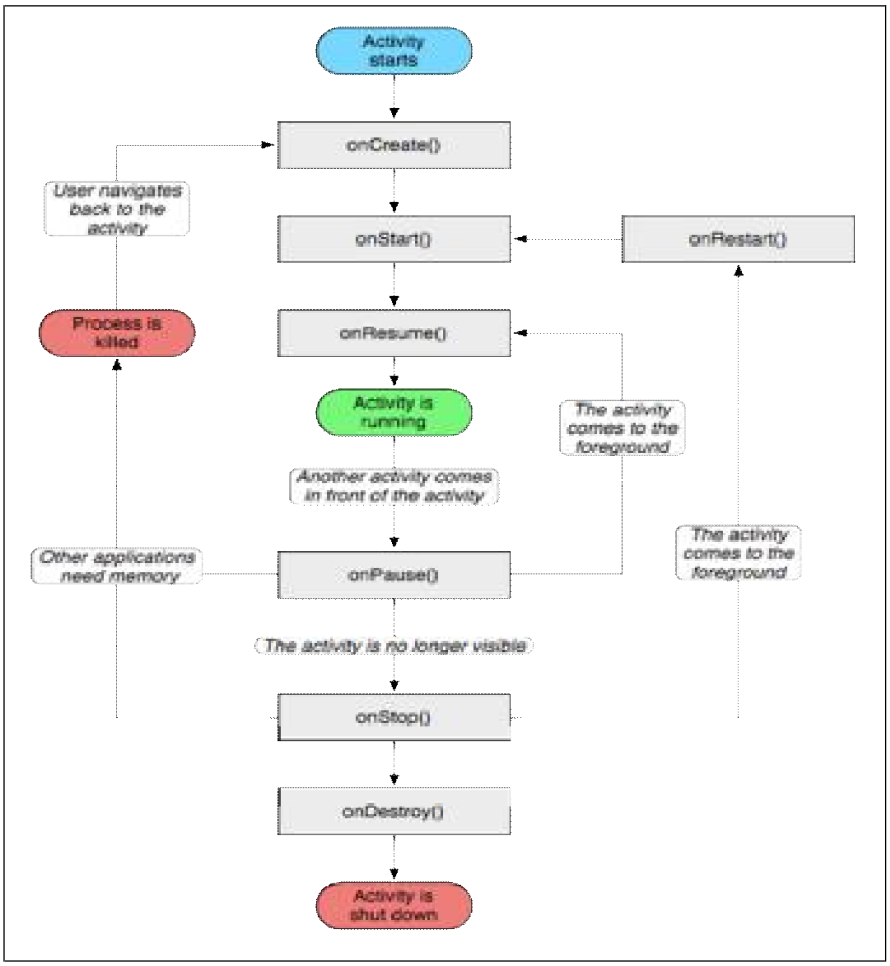
Οι Εφαρμογές του Android

Κάθε εφαρμογή του Android δομείται από κάποια στοιχεία που συνδέονται χρησιμοποιώντας ένα XML αρχείο, το οποίο περιγράφει κάθε στοιχείο και τον τρόπο που αλληλεπιδρά με τα άλλα.

Τα στοιχεία αυτά μπορεί να είναι ένας συνδυασμός από τα παρακάτω:

- **ΔΡΑΣΤΗΡΙΟΤΗΤΑ(ACTIVITY)**

- Η Δραστηριότητα αποτελεί το πιο κοινό από τα δομικά στοιχεία μιας εφαρμογής και είναι το επίπεδο παρουσίασης εφαρμογών (presentation layer). Πρόκειται για μία απλή οθόνη της εφαρμογής. Κάθε δραστηριότητα υλοποιείται σαν μία κλάση που επεκτείνει τη Βασική Κλάση Δραστηριότητας (Activity Base Class). Στη συγκεκριμένη κλάση προβάλλεται μία διεπαφή χρήστη (user interface) που αποτελείται από όψεις(views) και ανταποκρίνεται σε συμβάντα (events). Στην ανάπτυξη Desktop εφαρμογών, μια δραστηριότητα είναι ισοδύναμη με μία φόρμα (form). Ένα σύνολο από Activities αποτελούν την εφαρμογή μας, αλλά κάθε Activity είναι αυτοτελής και ανεξάρτητη η μία από την άλλη. Μία εφαρμογή μπορεί να κάνει χρήση μιας ήδη υπάρχουσας Activity σε περίπτωση που το χρειαστεί, όπως είναι η



χρήση της φωτογραφικής μηχανής που υπάρχει ήδη σαν εφαρμογή στο κινητό μας.

- **ΥΠΗΡΕΣΙΑ (SERVICE)**

- Η υπηρεσία είναι ένας κώδικας που τρέχει για ένα μεγάλο χρονικό διάστημα και χωρίς διεπαφή χρήστη. Σκοπός της είναι να ενημερώνει τις πηγές δεδομένων και τις ορατές δραστηριότητες με την ενεργοποίηση ειδοποιήσεων. Π.χ μία εφαρμογή media player μπορεί να συνεχίζει να παίζει μουσική ακόμη και αν το κύριο παράθυρο της εφαρμογής δε βρίσκεται στο προσκήνιο. Είναι σύνηθες ένα Activity να αναθέτει σε ένα Service μία εργασία χρονοβόρα, προκειμένου να μην υπάρξει κάποια απώλεια στην ανταποκρισιμότητα της διεπαφης χρήστη.

- **ΠΑΡΟΧΟΣ ΠΕΡΙΕΧΟΜΕΝΟΥ (CONTENT PROVIDER)**

Ο πάροχος περιεχομένου χρησιμοποιείται ώστε μία εφαρμογή να μοιράζεται με άλλες εφαρμογές, όταν χρειάζεται. Είναι μία κλάση στην οποία υλοποιούνται μέθοδοι που επιτρέπουν στις εφαρμογές την αποθήκευση και την επαναφορά δεδομένων συγκεκριμένου τύπου που χειρίζεται ο πάροχος περιεχομένου. Μερικά από τα συνηθέστερα δεδομένα που μοιράζονται μεταξύ των Content Providers είναι οι επαφές του χρήστη και οι βάσεις δεδομένων SQLite μιας εφαρμογής. Τα δεδομένα μπορεί να είναι αποθηκευμένα στην κάρτα μνήμης του κυνητού, σε ένα απομακρυσμένο εξυπηρετητή ή σε μία βάση δεδομένων.

- **ΠΡΟΘΕΣΗ ΚΑΙ ΦΙΛΤΡΟ ΠΡΟΘΕΣΕΩΣ(INTENTS)**

Η πρόθεση είναι ένα αντικείμενο το οποίο περιγράφει τι θέλει να κάνει μία εφαρμογή. Βασικά στοιχεία της είναι ποια ενέργεια θέλει η εφαρμογή για να εκτελεστεί και ποια δεδομένα θα χρησιμοποιήσει για να την εκτελέσει. Εν αντιθέσει, το φίλτρο προθέσεως είναι ένα αντικείμενο που περιγράφει τι είδους προθέσεις είναι δυνατόν να εξυπηρετηθούν.

- **ΔΕΙΚΤΕΣ ΜΕΤΑΔΟΣΗΣ (BROADCAST RECEIVERS)**

Πρόκειται για μία υπηρεσία που αντιλαμβάνεται ορισμένα γεγονότα του συστήματος και αναλαμβάνει να ενημερώσει το σύστημα και τις υπόλοιπες εφαρμογές. Δεν έχει γραφικό περιβάλλον και ενημερώνει τον χρήστη μέσω της μπάρας ειδοποιήσεων. Τα δίαφορα μηνύματα και ανακοινώσεις στέλνονται από το ίδιο το λειτουργικό σύστημα ανά τακτά χρονικά διαστήματα. Ένα πολύ χαρακτηριστικό παράδειγμα είναι η ανακοίνωση που εμφανίζεται όταν η μπαταρία της συσκευής μας είναι χαμηλή.

ΚΑΤΑΚΕΡΜΑΤΙΣΜΟΣ ΤΟΥ ANDROID

Όπως αναφέρθηκε και στα προηγούμενα κεφάλαια, το Android χάρη στη δυναμική του έχει κατακτήσει ένα μεγάλο μερίδιο της αγοράς και είναι διαθέσιμο σε δεκάδες συσκευές κινητής τηλεφωνίας. Οι συσκευές κυκλοφορούν από τις κατασκευάστριες εταιρίες είτε με την νεότερη έκδοση του λειτουργικού συστήματος, είτε με την αμέσως προηγούμενη, με την αντίστοιχη αναβάθμιση στην νεότερη. Βέβαια, οι αναβαθμίσεις δεν είναι πάντα επαρκείς, εξαιτίας του στόχου των κατασκευαστών για ευκολία και κέρδος, οι οποίοι προτιμούν να κυκλοφορούν μόνο συσκευές που περιλαμβάνουν τις νεότερες εκδόσεις.

Το πρόβλημα αυτό έρχονται να αντιμετωπίσουν οι διάφοροι προγραμματιστές του Android, οι οποίοι παρέχουν και υποστηρίζουν οι ίδιοι τις διάφορες αναβαθμίσεις, γνωστές ως “Custom Roms”.

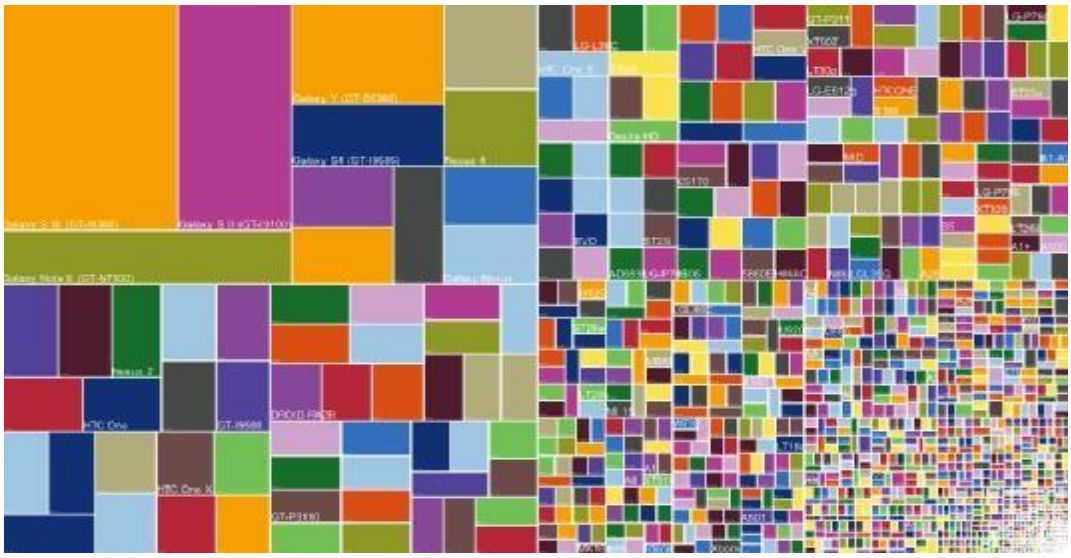
Πιο αναλυτικά, εκατομμύρια συσκευές ανά τον κόσμο λειτουργούν με τη βοήθεια του Android. Αυτό έχει ως αποτέλεσμα το λειτουργικό να τρέχει σε αμέτρητες διαφορετικές αναλύσεις, άλλες για κινητό, άλλες για tablet κλπ.
Η ποικιλομορφία αυτή οδηγεί στον κατακερματισμό της πλατφόρμας. Έτσι δυσκολεύονται πολύ οι προγραμματιστές, καθώς κάθε αναβάθμιση που καλούνται να πραγματοποιήσουν πρέπει να γίνει λαμβάνοντας υπόψιν όλες τις ιδιαιτερότητες των μοντέλων.

Μια άκρως ενδιαφέρουσα μελέτη δόθηκε στη δημοσιότητα, παρουσιάζοντας με ιδανικό τρόπο την ποικιλομορφία του λειτουργικού συστήματος Android. Έχει διεξαχθεί από την Open Signal Maps, λαμβάνοντας δεδομένα από 681.900 χρήστες που χρησιμοποίησαν την εφαρμογή της εταιρίας τον περασμένο χρόνο.

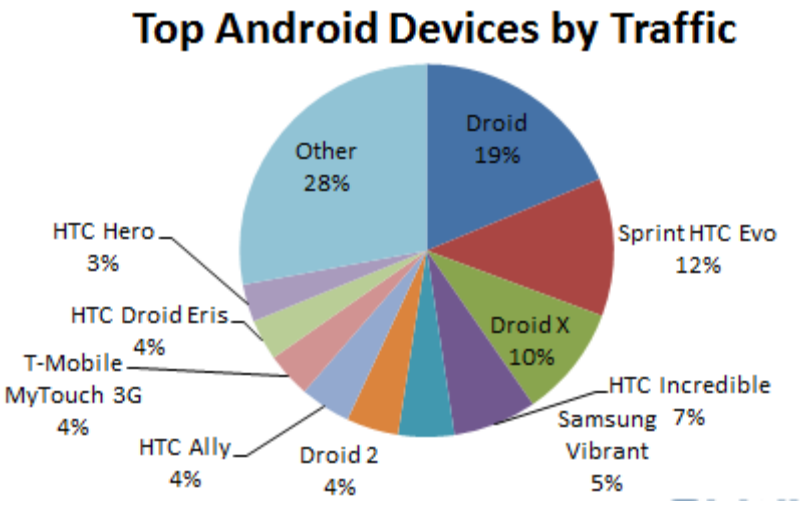
Τα αποτελέσματα συγκέντρωσαν δεδομένα από 4.000 διαφορετικές συσκευές.

Στο παρακάτω διάγραμμα παρουσιάζεται η κατανομή των συσκευών. Αξίζει να δοθεί μεγάλη βάση στο εντυπωσιακό κομμάτι που καταλαμβάνει η Samsung, αντιπροσωπεύοντας το 40% του συνόλου,ενώ μόνο το Samsung S II το 10%.

Ο κατακερματισμός είναι ένα τεράστιο πρόβλημα για το οποίο διαμαρτύρονται οι προγραμματιστές του Android, αφού είναι υποχρεωμένοι να ελέγχουν τη δουλεία τους πάνω σε δεκάδες συσκευές. Οι μεγάλες εταιρίες προφανώς διαθέτουν τα ποσά για τέτοιες επενδύσεις, αλλά εταιρίες 5-6 ατόμων μπορούν να προσφέρουν εξαιρετικές εφαρμογές στο iOS. Βέβαια σε αντίθεση με τους χρήστες iOS, που αν διαπιστώσουν προβλήματα σε μία εφαρμογή γράφουν ένα αρνητικό σχόλιο και την εγκαταλείπουν , οι Android χρήστες τείνουν να είναι ικανοποιημένοι όταν βρίσκουν εφαρμογές που απλά λειτουργούν στη συσκευή τους, ακόμα και αν έχουν μία ή δύο δυσλειτουργίες.



Ο κατακερματισμός του Android φαίνεται να αυξάνεται, οδηγώντας σε πολυπλοκότητα και αυξημένο κόστος τους προγραμματιστές.



https://www.androidheadlines.com/2010/11/android-device-usage-by-phone-og-droid-still-1.html

Η ΕΞΕΛΙΞΗ ΤΟΥ ANDROID

Συμφώνα με προηγούμενη αναφορά , το Android πρόκειται για ένα λειτουργικό σύστημα ανοικτού τύπου. Λόγω της open source φύσης που παρουσιάζει μπόρεσε σε μικρό χρονικό διάστημα να κυκλοφορήσει πολλές εκδόσεις .

Οι κύριες εκδόσεις παρουσιάζονται παρακάτω :

“Android 1.0”

Η πρώτη συσκευή που τρέχει σε Android 1.0 είναι η HTC Dream G1 , με τα ακόλουθα χαρακτηριστικά :

- Υποστήριξη HTML , XHTML ιστοσελίδων ,POP3 ,IMAP4 KAI SMPT
- Multitasking , instant messaging , GRS , Wi-Fi , Bluetooth, η αγορά νέων εφαρμογών καθώς και η αναβάθμιση των ήδη υπάρχοντων.
- Android Market -> η εφαρμογή μέσω της οποίας είναι δυνατή η αγορά νέων εφαρμογών καθώς και η αναβάθμιση των ήδη υπάρχοντων.
- Επιτρέπεται η ομαδοποίηση των εφαρμογών σε ένα ενιαίο φάκελο στην αρχική σελίδα.
- Δυνατότητα συγχρονισμού του Google με πολλές εφαρμογές.
- Φωνητική κλήση: που επιτρέπει την κλήση και τη διάθεση των τηλεφωνημάτων χωρίς την πληκτρολόγηση ονόματος ή αριθμού (Voicer Dialer).
- Εφαρμογές : Ξυπνητήρι , Αριθμομηχανή , dialer , Εικόνες , Ρυθμίσεις.
- Χάρτες της Google με το Latitude (Street View καθώς και δορυφορικές εικόνες) και κατεύθυνση της οδήγησης με χρήση GPS.
- Υποστήριξη της εφαρμογής Google Talk για ανταλλαγή μηνυμάτων μέσω του διαδικτύου.
- Δυνατότητα χρήσης οποιασδήποτε φωτογραφίας για φόντο της επιφάνειας εργασίας της συσκευής.

“Android 1.5 (Cupcake)”

Η έκδοση Cupcake βασίζεται στο Linux Kernel 2.6 και 2.7. Το Linux Kernel (πυρήνας του Linux) είναι ένα λειτουργικό σύστημα σαν το Unix, που χρησιμοποιείται από μία ποικιλία λειτουργικών συστημάτων που βασίζονται σε αυτό, τα οποία είναι συνήθως με τη μορφή των διανομών Linux. Το Linux Kernel είναι ένα χαρακτηριστικό παράδειγμα του ελεύθερου και ανοιχτού κώδικα λογισμικού.

Ο πυρήνας του Linux διατίθεται σύμφωνα με την άδεια GNU (General Public License version 2 – GRLv2) και έχει αναπτυχθεί παγκοσμίως.

Αρχικά, σχεδιάστηκε και αναπτύχθηκε το 1991 από τον Φιλανδό φοιτητή της επιστήμης των υπολογιστών Linus Torvalds. Το Linux γρήγορα προσαρμόστηκε από προγραμματιστές και χρήστες, με κώδικες από άλλα ελεύθερα λειτουργικά συστήματα.

Το **Android 1.5** παρουσιάστηκε στις 30 Απριλίου του 2009 , με τα ακόλουθα χαρακτηριστικά:

- Παγκόσμιο πλαίσιο αναζήτησης.
- Επανασχεδιασμός του Android Market : περιήγηση στις κατηγορίες και στα φίλτρα (Top free , Top paid , Just in). Το Android Market ή το Android Central, είναι το αποθετήριο της Google για εφαρμογές Android. Το Market προσφέρει δεκάδες χιλιάδες εφαρμογές κινητής τηλεφωνίας για συσκευές Android (συσκευές που χρησιμοποιούν το Android mobile λειτουργικό σύστημα), πολλές από τις οποίες είναι ελεύθερες για κατέβασμα και χρήση από την Android συσκευή.

- Εναλλαγές μεταξύ κάμερας και βίντεο σε μορφή MPEG-4 και 3GP.
- Ταχύτερη εκκίνηση της κάμερας, γρηγορότερη λήψη εικόνας, καθώς και πιο ολοκληρωμένη συλλογή φωτογραφιών.
- Γρηγορότερη απόκτηση της θέσης στο GPS (powered by SURL AGPS).
- Προστέθηκε Auto-Pairing και στερεοφωνική υποστήριξη για Bluetooth (AZPP και
- AVRCP προφίλ).
- Πληκτρολόγιο αφής (πάνω στην οθόνη).
- Δυνατότητα άμεσου Upload (μεταφόρτωσης) των βίντεο στο YouTube και το Picasa.
- Εισαγωγή των μικρογραφιών (Widgets) μιας εφαρμογής που μπορούν να ενσωματωθούν στην επιφάνεια εργασίας και να ανανεώνονται σε τακτά χρονικά διαστήματα.
- Δυνατότητα επιλογής εικόνων του χρήστη για εμφάνιση στις επαφές που βρίσκονται στα αγαπημένα.
- Δυνατότητα μετάβασης από τη μία οθόνη της εφαρμογής στην άλλη με χρήση εφέ (fade in, fade out).

“Android 2.2 (Froyo)”

Ακολουθεί το Android 2.2, με το όνομα Froyo, το Μάιο του 2010. Η έκδοση Froyo ανάμεσα σε άλλες αλλαγές περιλαμβάνει :

- Ο χρήστης μπορεί να διαθέτει πολλαπλούς λογαριασμούς.
- Βελτιστοποιήσεις στην ταχύτητα , τη μνήμη και την απόδοση.
- Ενημέρωση του Android Market : αυτόματες ενημερώσεις , εγκατάσταση εφαρμογών στην κάρτα SD.
- Νέες συμβουλές για Widget στην αρχική οθόνη.
- Υποστήριξη του Android Cloud. Το Android Cloud παρέχει ένα ευρύ φάσμα από αυτόματα κλιμακόμενα χαρακτηριστικά δικτύου και αντικείμενα δεδομένων για την εφαρμογή μας.
- Δυνατότητα ανεβάσματος αρχείων στο πρόγραμμα περιήγησης.
- Βελτιωμένη υποστήριξη του Microsoft Exchange. Το Microsoft Exchange είναι διακομιστής συνεργατικής επικοινωνίας που βασίζεται στην εφαρμογή ηλεκτρονικού

ταχυδρομείου και προορίζεται για επιχειρήσεις. Μπορεί να γίνει εύκολη εγκατάσταση του σε μία συσκευή Android. Μετά την εγκατάσταση μπορούμε να έχουμε πρόσβαση και συγχρονισμό στο e-mail, στο ημερολόγιο και στις επαφές.

- Υποστήριξη USB και Hotspot , φωνητική κλήση και ανταλλαγή επικοινωνίας μέσω
- Bluetooth.
- Πολύγλωσσο πληκτρολόγιο.
- Adobe Flash 10.1 .
- Δυνατότητα του χρήστη να κλειδώνει τη συσκευή του με τη χρήση κωδικού ώστε να αποτρέπεται η πρόσβαση σε τρίτους.
- Οι εφαρμογές πλέον μπορούν να εγκατασταθούν και σε άλλες μνήμες της συσκευής εκτός της κύριας-εσωτερικής μνήμης της.

“Android 2.3 (Gingerbread)”

Το Δεκέμβριο του 2010, έχουμε την εμφάνιση της καινούριας έκδοσης Android 2.3 (Gingerbread). Οι αλλαγές που έχουν γίνει είναι οι ακόλουθες:

- Βελτιώσεις UI (User Interface= περιβάλλον χρήστη ή διασύνδεση χρήστη) για την απλότητα και την ταχύτητα. Το UI είναι το σύνολο των γραφικών στοιχείων που εμφανίζονται στην οθόνη της συσκευής Android και χρησιμοποιούνται για την αλληλεπίδραση του χρήστη με τη συσκευή αυτή. Με απλά λόγια παρέχουν εργαλεία και ενδείξεις που κάνουν πιο εύκολη τη ζωή του χρήστη της συσκευής Android.
- Νέο πληκτρολόγιο για ταχύτερη εισαγωγή κειμένου.
- Δυνατότητα, με ένα πάτημα, επιλογής λέξεων και άμεση αντιγραφή/επικόλληση.
- Near Field Communication (NFC), βελτιωμένη διαχείριση ενέργειας, με τερματισμό εφαρμογών που τρέχουν στο προσκήνιο και καταναλώνουν πολύ ενέργεια.
- Διαχείριση κατεβάσματος αρχείων και εφαρμογών.
- Κλήσεις μέσω διαδικτύου.
- Νέα ηχητικά εφέ: ισοστάθμιση του αναπαραγόμενου ήχου, ενίσχυση των μπάσων, δημιουργία ηχώ.
- Υποστήριξη για WebMD/VP8 αναπαραγωγή βίντεο και AAC (Advanced Audio Coding Files) κωδικοποίηση αρχείων. Με την AAC κωδικοποίηση παρέχεται υψηλότερη απόδοση

κωδικοποίησης, τόσο για σταθερά όσο και παροδικά σήματα. Αυτή η μορφή κωδικοποίησης διατηρεί μία ποιότητα σχεδόν πανομοιότυπη με αυτή της αρχικής ηχητικής πηγής.

“Android 3.0 (HONEYCOMB)”

Η έκδοση Android 3.0 (Honeycomb) βρισκόταν στη διάθεση των χρηστών και προγραμματιστών από το Φεβρουάριο του 2011, λίγες μέρες μετά την επανέκδοση του Android
2.3.3 και προοριζόταν αποκλειστικά για ταμπλέτες (tablet). Τα Tablet pc είναι υπολογιστές που λειτουργούν χωρίς πληκτρολόγιο, με τη χρήση ειδικών μολυβιών ή δαχτύλων. Επίσης, δίνουν τη δυνατότητα σύνδεσης στο διαδίκτυο. Απαιτούν βέβαια την ύπαρξη λειτουργικού συστήματος (Android στην περίπτωση μας).

Μερικά από τα χαρακτηριστικά του είναι:

- Ειδικά βελτιωμένη έκδοση για tablet και συσκευές με οθόνες μεγαλύτερου μεγέθους.
- Επιτάχυνση υλικού, υποστήριξη για multi-core επεξεργαστές (επεξεργαστές πολλαπλών πυρήνων) και δυνατότητα κρυπτογράφησης όλων των δεδομένων του χρήστη.
- Εξευγενισμένο multi-tasking, πολλαπλές κοινοποιήσεις, προσαρμογή αρχικής οθόνης, widgets και ανασχεδιασμένο πληκτρολόγιο.
- Υποστήριξη για video chat, πολλαπλές καρτέλες περιήγησης, φόρμες αυτόματης συμπλήρωσης και προσθήκη νέας λειτουργίας “incognito”, που επιτρέπει ανώνυμη περιήγηση.
- Βελτίωση του Bluetooth.
- Ενσωματωμένη υποστήριξη για το Medial Picture Transfer Protocol (PTP). Το PTP είναι το πωτόκολλο που επιτρέπει τη μεταφορά εικόνων από ψηφιακές κάμερες σε drivers (πχ tablets, pc) χωρίς τη χρήση driver. Έχει οριστικοποιηθεί ως ISO 15740.

- Δυνατότητα προβολής άλμπουμ και άλλων συλλογών σε λειτουργία πλήρους οθόνης.
- Εκ νέου σχεδιασμός του πληκτρολογίου ώστε να προσαρμόζεται στη μεγαλύτερη οθόνη των tablets.

"Android 4.0 (Ice Cream Sandwich)"

Η έκδοση αυτή ανακοινώθηκε το 2011 στο San Francisco και αποτελεί την προσπάθεια της εταιρίας για δημιουργία ενός ενιαίου λειτουργικού συστήματος για όλες τις συσκευές.Η έκδοση αυτή είναι βασισμένη στον πυρήνα του Linux 3.0.1. Οι ιδιότητες του παρατίθενται παρακάτω:

- Αύξηση στην ταχύτητα και την απόδοση.
- Εικονικά πλήκτρα.
- Ευκολότερη δημιουργία φακέλων με στυλ drag-and-drop. Με το drag-and-drop κάνοντας κλικ σε ένα εικονικό αντικείμενο μπορούμε να του αλλάξουμε θέση ή να το μετατρέψουμε σε άλλο εικονικό αντικείμενο. Γενικότερα, μπορούμε να δημιουργήσουμε διαφόρων τύπων εικόνες μεταξύ δύο διαφορετικών αφηρημένων αντικειμένων.
- Δυνατότητα επιτάχυνσης ή επιβράδυνσης των μηνυμάτων στον τηλεφωνητή.
- Βελτιωμένη διόρθωση λάθους στο πληκτρολόγιο, σε πραγματικό χρόνο ομιλίας, για υπαγόρευση κειμένου.
- Δυνατότητα πρόσβασης σε εφαρμογή απευθείας από την οθόνη κλειδώματος (αρχική οθόνη).
- Νέες καρτέλες προστίθενται στον Browser (επιτρέπει έως και 16 καρτέλες).
- Νέα δυνατότητα για τερματισμό εφαρμογών που χρησιμοποιούν μνήμη στο προσκήνιο.
- Βελτίωση της εφαρμογής της κάμερας (μηδενική υστέρηση κλείστρου, λειτουργία πανοραμικής λήψης, δυνατότητα μεγέθυνσης κατά τη διάρκεια της εγγραφής, εγγραφή βίντεο 1080p) για τις βασικές συσκευές Android.
- Ενσωματωμένος επεξεργαστής φωτογραφιών.

"Android 4.4 (KitKat)"

Το Android 4.4, με το ψευδόνυμο KitKat, είναι από τις σημαντικότερες εκδόσεις του Android. Παρουσιάστηκε το Σεπτέμβριο του 2013 μαζί με την παρουσίαση ενός τεράστιου αγάλματος KitKat έξω από το κτίριο της Google. Βέβαια μέχρι τον Οκτώβριο του 2013 δεν είχε γίνει γνωστό τι καινούργιο παρουσίαζε αυτή η νέα εκδοση.

Το Android 4.4 άλλαξε τα γραφικά παρουσιάζοντας ένα ελαφρύτερου χρώματος σχέδιο, με πιο εκλεπτισμένα εικονίδια και γενικότερα επανασχεδιασμένη αρχική οθόνη. Τα νέα χαρακτηριστικά του είναι:

- Μπάρες που κρύβονται και εμφανίζονται στην οθόνη ανάλογα με τις ειδοποιήσεις.
- Βελτίωση του Google Map- δυνατότητα απευθείας σύνδεσης με επιχειρήσεις και επίδειξη διαδρομών.
- Ιδιαίτερα βελτιωμένη ασύρματη υποστήριξη.

Η πρώτη συσκευή που κυκλοφόρησε με το Android KitKat είναι η LG- made Nexus 5. Όλες οι συσκευές μέχρι τις αρχές του 2014 είχαν αναβαθμιστεί σε KitKat, καθώς και νέες έκαναν την εμφάνισή τους το πρώτο τρίμηνο του 2014.

Το αρχικό όνομα που προοριζόταν για αυτή την έκδοση λογισμικού ήταν ‘’Key Lime Pie’’,
ωστόσο ύστερα με συμωνία που πραγματοποιήθηκε με την Nestle εδραιώθηκε το όνομα KitKat.

“Android 5.0 (Lollipop)“

Το Android 5.0 έκανε την επίσημη παρουσίασή του τον Ιούνιο του 2014. Η Google το καθιέρωσε ως Android Lollipop και η παρουσίαση του συνέπεσε με την κυκλοφορία των Nexus 6 και Nexus 9.

Οι διαφοροποιήσεις σε σχέση με τις προηγούμενες εκδόσεις είναι:

- Εντελώς νέα αισθητική με πιο bold χρώματα και γραφικά, η κίνηση είναι φυσική και τα χρώματα πιο ρεαλιστικά. Ας επισημάνουμε βέβαια και την ύπαρξη ιδιαίτερα καλοχεδιασμένων σκιών.
- Διάθεση πάνω από 5,000 APIs.
- Δυνατότητα ευελιξίας ανεξάρτητα από το μέγεθος οθόνης της συσκευής, προσφέροντας την ίδια εμπειρία ανεξαρτήτου μαγέθους.
- Βελτίωση της μπαταρίας, προσφέροντας επιπλέον παράταση 90 λεπτών κατά τη χρήση με το Battery Saver Featuring. Επιπλέον, δίνεται η δυνατότητα προβολής του ποσοστού της μπαταρίας που απομένει και του ποσοστού που έχει η μπαταρία κατά τη φόρτιση, μέσω του Project Volta,όπως παρουσιάστηε από την Google.
- Προηγμένη μορφή ασφάλειας με το Kill Switch που με τη χρήση ενός κωδικού μπορούμε να προκαλέσουμε αχρήστευση της συσκευής μας, για όποιον δεν γνωρίζει τον κωδικό.
- Η αποκρυπτογράφηση γίνεται by default.
- Δυνατότητα του χρήστη να κάνει προσωπική διαχείριση των notifications, επιλέγοντας ποιες και πότε θα εμφανίζονται στην οθόνη.
- Η διεπαφή αποκτά νέο χαρακτήρα, γνωστή ως Moonshine, κάνοντας τα απλά τηλέφωνα Android να υιοθετούν το look του Nexus 5, που διαφέρει εντελώς από τις υπόλοιπες συσκευές Nexus.
- Το περιβάλλον της συσκευής γίνεται πιο παιχνιδιάρικο.

Android 6.0-6.0.1 "Marshmallow"

Το **Android 6.0-6.0.1 "Marshmallow"** (με την κωδική ονομασία **Android M** κατά τη διάρκεια της ανάπτυξης του)[22] είναι η έκτη κύρια έκδοση του λειτουργικού συστήματος Android από την Google. Παρουσιάστηκε τον Μάιο του 2015 στο συνέδριο του Google I/O, και κυκλοφόρησε επίσημα τον Οκτώβριο του 2015.[23]

Το Marshmallow επικεντρώνεται κυρίως στη βελτίωση της συνολικής εμπειρίας του χρήστη σε σύγκριση με τον προκάτοχο του, Lollipop,[24] εισήγαγε μια νέα αρχιτεκτονική στα δικαιώματα των εφαρμογών, νέα APIs για συμφραζόμενα βοηθούς (που χρησιμοποιείται για πρώτη φορά από ένα νέο χαρακτηριστικό το "Now on Tap" που παρέχει το πλαίσιο με τα ευαίσθητα αποτελέσματα αναζήτησης), ένα νέο σύστημα διαχείρισης ενέργειας που μειώνει την δραστηριότητα του παρασκηνίου όταν μια συσκευή δεν χρησιμοποιείται, εγγενή υποστήριξη για την αναγνώριση δακτυλικών αποτυπωμάτων και USB τύπου-C, τη δυνατότητα να μεταφοράς των δεδομένων και τις εφαρμογές σε μια κάρτα microSD, και άλλες εσωτερικές αλλαγές.

Κατά τον Δεκέμβριο του 2016, το 24% των συσκευών με πρόσβαση στο Google Play τρέχει το Android 6.0.

Android 7.0-7.1.1

Το **Android 7.0-7.1.1 "Nougat"** (με την κωδική ονομασία **Android N** κατά τη διάρκεια της ανάπτυξης του)[25] είναι η έβδομη σημαντική έκδοση του λειτουργικού συστήματος Android από την Google. Κυκλοφόρησε για πρώτη φορά ως beta build στις 9 Μαρτίου 2016[26] και επίσημα στις 22 Αυγούστου 2016. Οι συσκευές Nexus ήταν οι πρώτες ενημερώθηκαν με το νέο λογισμικό, αν και το LG V20 ήταν το πρώτο νέο έξυπνο τηλέφωνο που κυκλοφόρησε με Nougat.[27]

Το Nougat εισάγει σημαντικές αλλαγές στο λειτουργικό σύστημα και την πλατφόρμα ανάπτυξης του, συμπεριλαμβανομένης της δυνατότητας να εμφανίσει πολλαπλές εφαρμογές στην οθόνη με προβολή διαίρεσης της οθόνης, υποστήριξη για τις απαντήσεις ενσωματωμένη με τις κοινοποιήσεις, καθώς και περιβάλλον Java βασισμένο στο OpenJDK αλλά και υποστήριξη για τις απόδοσης γραφικών Vulkan API, και απρόσκοπτες ενημερώσεις συστήματος για υποστηριζόμενες συσκευές.

Δοκιμή (Testing)

Εισαγωγή στο automated unit testing και το framework JUnit

Τι είναι το automated unit testing;

Το automated unit testing είναι μια πρακτική που έγινε ευρέως γνωστή μέσα από τη μεθοδολογία ανάπτυξης eXtreme Programming αλλά ακολουθείται ευρέως στις περισσότερες σύγχρονες μεθοδολογίες ανάπτυξης λογισμικού.

Η φιλοσοφία πίσω από αυτή την πρακτική βασίζεται στο γεγονός ότι η παραγωγικότητα σε μια ομάδα ανάπτυξης έχει άμεση σχέση με τη σταθερότητα και την αξιοπιστία του κώδικα. Η σταθερότητα και η αξιοπιστία του κώδικα εξασφαλίζονται με το testing.

Ποιος όμως είναι υπεύθυνος για το testing; Αν είναι ένα μέλος της ομάδας τότε πρέπει, εκτός από το να έχει απίστευτα αποθέματα πειθαρχίας και υπομονής, να ενημερώνεται διαρκώς για όλες τις αλλαγές που γίνονται. Επίσης σημαίνει ότι υπάρχει ένα λιγότερο άτομο για την ανάπτυξη του κώδικα.

Τη λύση στο παραπάνω πρόβλημα έρχεται να δώσει το automated unit testing. Η πρακτική αυτή απαιτεί από τον προγραμματιστή που θα γράψει τον κώδικα να γράψει και ένα ή περισσότερα unit tests για αυτόν, δηλαδή κώδικα που να τεστάρει συγκεκριμένα classes και μεθόδους τους ως προς το αποτέλεσμά τους.

Αυτά τα unit tests εξασφαλίζουν ότι ο κώδικας που έχει γραφεί όχι μόνο γίνεται compile αλλά και λειτουργεί έτσι όπως θέλουμε να λειτουργεί.

Το πλεονέκτημα του automated unit testing είναι ότι δεν χρειάζεται κανείς να θυμάται τι πρέπει να ελέγξει στον κώδικα. Τα unit tests είναι εκεί και περιμένουν κάποιον να τα εκτελέσει.

Υπάρχουν διάφορα frameworks για τη συγγραφή unit tests με πιο δημοφιλές το JUnit το οποίο μάλιστα συνεργάζεται πολύ καλά με το Ant. Έτσι είναι δυνατό να γράψουμε μια σειρά από tests για τον κώδικά μας, να προσθέσουμε ορισμένα tags στο build file του Ant και με αυτό τον τρόπο κάθε φορά που κάνουμε compile να τρέχουν και όλα (ή όσα θέλουμε) τα unit tests. Αν όλα τα unit tests τρέξουν επιτυχημένα αυτό σημαίνει ότι ο κώδικάς μας λειτουργεί όπως θα έπρεπε.

Γιατί να χρησιμοποιώ automated unit testing;

Η συνήθης δικαιολογία για να μην γράψει κανείς unit tests είναι ότι δεν υπάρχει χρόνος. Αυτό βέβαια μπορεί να αμφισβητηθεί εύκολα. Θα πρέπει να καταλάβουμε ότι τα unit tests δεν είναι απλά επιπλέον κώδικας αλλά μέρος του κώδικα που γράφουμε για να υλοποιήσουμε τις λειτουργίες ενός συστήματος.

Μάλιστα υπάρχει μια ακόμη πρακτική που έρχεται να συμπληρώσει το automated unit testing και ονομάζεται Test first programming. Η πρακτική αυτή μας λέει ότι για οποιαδήποτε λειτουργία θέλουμε να προσθέσουμε στον κώδικά μας πρέπει πρώτα να γράψουμε τουλάχιστον ένα unit test που να την ελέγχει και έπειτα να γράψουμε τον κώδικα που την υλοποιεί. Χαρακτηριστικά στη θεωρεία του eXtreme Programming αναφέρεται ότι λειτουργία του συστήματος για την οποία δεν έχουν γραφτεί unit tests θεωρείται ότι απλά δεν υπάρχει στο σύστημα.

Ένας τέτοιος τρόπος ανάπτυξης μπορεί στην αρχή να φαίνεται περίεργος και δύσκολος αλλά έχει τα εξής πλεονεκτήματα:

- Εξασφαλίζει ότι όλες οι λειτουργίες του συστήματός μας είναι τεσταρισμένες.

- Εξασφαλίζει πιο καλή σχεδίαση του προγραμματιστικού interface (API) των classes του συστήματός μας. Κάτι τέτοιο συμβαίνει γιατί αναγκαζόμαστε να χρησιμοποιήσουμε τον κώδικά μας σαν να είμαστε κάποιος άλλος προγραμματιστής. Οπότε αναγκαστικά τα public members του κάθε class μας πρέπει να είναι πολύ καλά σχεδιασμένα για να μπορούμε να τα χρησιμοποιήσουμε.

- Παρέχει πρώτης τάξεως παραδείγματα για το πώς πρέπει να χρησιμοποιηθεί ένα class ή και ένα ολόκληρο σύστημα. Αυτό οφείλεται στο ότι έχουμε ήδη γράψει unit tests που χρησιμοποιούν τα classes μας και έτσι υπάρχουν έτοιμα παραδείγματα τα οποία μπορεί να διαβάσει κάποιος άλλος προγραμματιστής της ομάδας (και όχι μόνο) για να μπορέσει να χρησιμοποιήσει τα classes μας.

- Με την πάροδο του χρόνου και όσο το σύστημά μας θα μεγαλώνει δημιουργείται μια ολόκληρη βιβλιοθήκη από unit tests τα οποία είναι και η απόδειξη ότι το σύστημά μας λειτουργεί ορθά. Αυτό μας δίνει τη δυνατότητα να κάνουμε όσες αλλαγές θέλουμε σε όποια classes θέλουμε αφού μετά από κάθε τέτοια αλλαγή μπορούμε να τρέξουμε όλα τα unit tests και αυτά θα μας δείξουν αν η αλλαγή μας «έσπασε» ένα τμήμα του συστήματος ή αν το σύστημα εξακολουθεί να λειτουργεί σωστά. Έτσι λοιπόν το automated unit testing μας παρέχει ένα δίχτυ ασφαλείας το οποίο μας επιτρέπει να κάνουμε refactoring του κώδικά μας γνωρίζοντας ότι το σύστημά μας εξακολουθεί να λειτουργεί όπως πρέπει.

Τι είναι το JUnit

Το JUnit είναι ένα framework ή βιβλιοθήκη ανοικτού κώδικα την οποία χρησιμοποιούν χιλιάδες προγραμματιστών σε Java για να γράψουν unit tests. Πλεονέκτημα του JUnit είναι η απλότητά του η οποία δίνει τη δυνατότητα σε ένα προγραμματιστή να γράψει το πρώτο του πλήρες unit test μέσα σε πολύ μικρό χρονικό διάστημα (της τάξεως της μιας ώρας).

Ένα ακόμη πλεονέκτημα του JUnit είναι ότι υπάρχει υποστήριξη για αυτό σε ένα άλλο πολύ δημοφιλές εργαλείο : το Ant. Έτσι αν χρησιμοποιηθούν ταυτόχρονα μας παρέχουν ένα πλήρες αυτοματοποιημένο περιβάλλον για unit testing.

Τόσο για το JUnit όσο και για το Ant, λόγω της δημοτικότητάς τους στις τάξεις των προγραμματιστών, υπάρχει πλήρης υποστήριξη στα πιο δημοφιλή ολοκληρωμένα περιβάλλοντα ανάπτυξης (IDEs) όπως το Eclipse, Netbeans, JDeveloper κλπ.

Πώς χρησιμοποιείται το JUnit;

Το JUnit εισάγει δύο έννοιες :

- TestCase: πρόκεται για ένα class το οποίο περιέχει μεθόδους οι οποίες ελέγουν το σύστημα
- TestSuite : πρόκεται για ένα σύνολο από TestCases ή και άλλες TestSuites τα οποία ομαδοποιούνται με σκοπό την καλύτερη οργάνωσή τους και την εκτέλεσή τους σαν σύνολα (δεν θα αναφερθούμε περαιτέρω).

Για να γράψει κανείς ένα TestCase πρέπει να κάνει τα εξής:

1. Να δημιουργήσει ένα class το οποίο θα κληρονομεί από το junit.framework.TestCase.
2. Να γράψει μέσα σε αυτό το class μια σειρά από public μεθόδους οι οποίες να αρχίζουν με test (π.χ. testAdd(), testInsert() κλπ), να μην επιστέφουν κάποια τιμή (void) και να μην παίρνουν καμία παράμετρο.
3. Αν πρέπει να χρησιμοποιηθούν κάποιοι πόροι (resources) μέσα σε αυτές τις μεθόδους, όπως για παράδειγμα ένα Connection σε μια βάση δεδομένων, θα πρέπει να κάνει override τη μέθοδο setUp() και μέσα σε αυτή να δημιουργήσει το instance του Connection.
4. Αντίστοιχα αν έχει δημιουργήσει κάποιους πόρους τους οποίους θα πρέπει να ελευθερώσει στο τέλος τότε θα πρέπει να κάνει override τη μέθοδο tearDown().

Το JUnit χρησιμοποιεί reflection για να ψάξει μεθόδους μέσα σε ένα class οι οποίες να είναι public, να ξεκινούν από test, να μην επιστρέφουν αποτέλεσμα (void) και να μην παίρνουν παραμέτρους.

Για κάθε μία από αυτές τις μεθόδους γίνονται τα παρακάτω:

1. Πριν την εκτέλεση της μεθόδου εκτελείται η setUp().
2. Εκτελείται η μέθοδος π.χ. testInsert().

Μετά την εκτέλεση της μεθόδου εκτελείται η tearDown().

Αυτοματοποιημένη δοκιμή

Στρατηγική δοκιμών στο Android

Η αυτοματοποιημένη δοκιμή των εφαρμογών είναι ιδιαίτερα σημαντική λόγω της μεγάλης ποικιλίας συσκευών που κυκλοφορούν στην αγορά. Επειδή δεν είναι δυνατόν να δοκιμάσουμε μία εφαρμογή με όλες τις δυνατές ρυθμίσεις μιας συσκευής, αποτελεί συνηθισμένη πρακτική η εκτέλεση δοκιμών με συνηθισμένες ρυθμίσεις.

Η εκτέλεση δοκιμών μας βοηθάει να βελτιώσουμε και να συντηρήσουμε μία εφαρμογή.

Πώς μπορούμε να δοκιμάσουμε τις εφαρμογές

Στο Android, οι δοκιμές βασίζονται στο JUnit. Μπορούμε να διαχωρίσουμε τις δοκιμές σε αυτές που χρειάζονται μόνο την JVM και σε αυτές που χρειάζονται το λειτουργικό σύστημα. Αν είναι εφικτό, είναι προτιμότερο να τρέξουμε τις δοκιμές απευθείας στην JVM, λόγω του ότι ο χρόνος που απαιτείται για την εκτέλεσή τους είναι πολύ μικρότερος σε σχέση με τον χρόνο που απαιτείται για την εγκατάσταση και την εκτέλεσή τους σε μία συσκευή Android.

Σημειώνεται ότι η εφαρμογή που δοκιμάζεται καλείται *εφαρμογή υπό δοκιμή*.

Δοκιμές μονάδων (unit tests) και δοκιμές ενσωμάτωσης (integration tests) στο Android

Η δοκιμή μονάδας δοκιμάζει μόνο τη λειτουργικότητα κάποιου στοιχείου.

Ας υποθέσουμε για παράδειγμα ότι ένα κουμπί σε μία Δραστηριότητα χρησιμοποιείται για την εκκίνηση μιας άλλης Δραστηριότητας. Μία δοκιμή μονάδας θα καθόριζε αν η αντίστοιχη πρόθεση (intent) δημιουργήθηκε σωστά, και όχι αν ξεκίνησε η δεύτερη Δραστηριότητα.

Μία δοκιμή ενσωμάτωσης θα έλεγχε επίσης, αν η Δραστηριότητα ξεκίνησε σωστά.

Το Android παρέχει διαφορετικές κλάσεις για δοκιμές μονάδων και δοκιμές ενσωμάτωσης.

Android και JUnit 3

Στη παρούσα φάση, το API δοκιμών του Android βασίζεται στο JUnit 3 και όχι στο JUnit 4.

Η Google εργάζεται για την αναβάθμιση του framework δοκιμών, με την οποία θα διαχωριστεί το framework δοκιμών από το framework εφαρμογών. Με αυτήν την αναβάθμιση σχεδιάζουν επίσης να αναβαθμίσουν το framework δοκιμών έτσι ώστε να βασίζεται στο JUnit 4.

Με το JUnit 3 οι κλάσεις δοκιμών κληρονομούν από την κλάση junit.framework.TestCase του JUnit 3.

Στο JUnit 3, οι μέθοδοι δοκιμών πρέπει να ξεκινούν με το πρόθεμα test. Πρέπει να καλείται η μέθοδος διαμόρφωσης setUp() και η τελική μέθοδος εκκαθάρισης teardown().

Τι πρέπει να υποβάλλουμε σε δοκιμή σε μία εφαρμογή

Ο ακόλουθος πίνακας περιλαμβάνει τις σημαντικές περιοχές τις οποίες πρέπει να υποβάλλουμε σε δοκιμή σε μία εφαρμογή.

να υποβάλλουμε σε δοκιμή σε μία εφαρμογή. **Περιοχή υπό δοκιμή**	**Περιγραφή**
Γεγονότα από τον κύκλο ζωής μιας Δραστηριότητας	Θα πρέπει να ελέγξουμε κατά πόσον οι Δραστηριότητες χειρίζονται καλά τα γεγονότα που συμβαίνουν κατά τον κύκλο ζωής τους. Θα πρέπει επίσης να ελέγξουμε αν η εφαρμογή συμπεριφέρεται σωστά όταν αλλάζουν οι ρυθμίσεις (configuration), και αν επανέρχονται στην πρότερη κατάσταση (state) τα στοιχεία της διεπαφής χρήστη.
Λειτουργίες του συστήματος αρχείων και της βάσης δεδομένων	Η ανάγνωση και η εγγραφή από και προς το σύστημα αρχείων θα πρέπει να ελεγχθεί, συμπεριλαμβανομένων των λειτουργιών των βάσεων δεδομένων.
Διαφορετικές ρυθμίσεις της συσκευής	Θα πρέπει επίσης να ελέγξουμε κατά πόσον η εφαρμογή συμπεριφέρεται καλά με διαφορετικές ρυθμίσεις της συσκευής.

Προϋποθέσεις δοκιμής

Στο Android αποτελεί καλή πρακτική να έχουμε μία μέθοδο με όνομα testPreconditions() η οποία δοκιμάζει (tests) τις προϋποθέσεις όλων των δοκιμών. Αν αυτή η μέθοδος αποτύχει, αυτό σημαίνει ότι δεν ισχύουν οι υποθέσεις στις οποίες βασίζονται οι δοκιμές.

Για ποιες δοκιμές απαιτείται το λειτουργικό σύστημα προκειμένου να τρέξουν

Δοκιμή τυπικών κλάσεων της Java

Αν οι κλάσεις δεν καλούν το Android API, μπορούμε να χρησιμοποιήσουμε το framework δοκιμών JUnit (ή οποιοδήποτε άλλο framework δοκιμών για Java) χωρίς περιορισμούς.

Το πλεονέκτημα αυτής της μεθόδου είναι ότι μπορούμε να χρησιμοποιήσουμε οποιοδήποτε framework δοκιμών για Java και ότι η ταχύτητα εκτέλεσης της δοκιμής μονάδας είναι πολύ μεγάλη σε σχέση με την ταχύτητα που έχουν οι δοκιμές που χρειάζονται το λειτουργικό σύστημα.

Δοκιμή Java κλάσεων οι οποίες χρησιμοποιούν το API του Android

Αν θέλουμε να δοκιμάσουμε κώδικα ο οποίος χρησιμοποιεί το API του Android, θα πρέπει να τρέξουμε αυτές τις δοκιμές σε μία συσκευή Android. Δυστυχώς, αυτό κάνει την εκτέλεση των δοκιμών να κρατάει περισσότερο.

Άγκιστρα

Instrumentation

Το API για τις δοκιμές παρέχει άγκιστρα στον κύκλο ζωής των στοιχείων λογισμικού και της εφαρμογής. Αυτά τα άγκιστρα απαρτίζουν το instrumentation API και επιτρέπουν στις δοκιμές να ελέγχουν γεγονότα από τον κύκλο ζωής (life cycle events) και την αλληλεπίδραση του χρήστη.

Υπό κανονικές συνθήκες μία εφαρμογή μπορεί μόνο να αντιδράσει σε τέτοια γεγονότα. Για παράδειγμα, όταν το Android δημιουργεί μία Δραστηριότητα καλείται η μέθοδος onCreate() για την Δραστηριότητα. Άλλη περίπτωση είναι όταν ο χρήστης πατήσει κάποιο κουμπί ή πλήκτρο οπότε και καλείται ο αντίστοιχος κώδικας. Μέσω του instrumentation μπορούμε να προκαλέσουμε τέτοια γεγονότα.

Μόνο μία κλάση δοκιμών που χρησιμοποιεί το instrumentation API μας επιτρέπει να στείλουμε γεγονότα πλήκτρων (key events) ή αφής (touch events) στην εφαρμογή υπό δοκιμή.

Για παράδειγμα, μία δοκιμή μπορεί να καλέσει την μέθοδο getActivity() η οποία εκκινεί μία Δραστηριότητα και επιστρέφει την Δραστηριότητα υπό δοκιμή. Έπειτα, μπορούμε να καλέσουμε την μέθοδο finish() και εν συνεχεία την μέθοδο getActivity() και πάλι, για να ελέγξουμε εάν η εφαρμογή επανάφερε σωστά την κατάσταση της Δραστηριότητας.

Το instrumentation API μας επιτρέπει να τρέξουμε το πρότζεκτ δοκιμών και το κανονικό πρότζεκτ στην ίδια διεργασία προκειμένου το πρότζεκτ δοκιμών να μπορεί να καλεί απευθείας μεθόδους του πρότζεκτ υπό δοκιμή.

Πώς διενεργεί δοκιμές το Android

Οι κλάσεις του συστήματος φορτώνουν και εκκινούν το πακέτο δοκιμών (test package), τερματίζουν τυχόν διεργασίες οι οποίες τρέχουν κάποιο στιγμιότυπο της εφαρμογής υπό δοκιμή, και στη συνέχεια φορτώνουν ένα νέο στιγμιότυπο της εφαρμογής υπό δοκιμή. Ακολούθως, δίνουν τον έλεγχο στον InstrumentationTestRunner, ο οποίος τρέχει κάθε κλάση δοκιμών που υπάρχει στο πακέτο δοκιμών.

Ούτε οι κλάσεις του συστήματος, ούτε ο InstrumentationTestRunner τρέχουν την εφαρμογή υπό δοκιμή. Αυτό το κάνουν οι κλάσεις δοκιμών απευθείας. Είτε καλούν μεθόδους στην εφαρμογή υπό δοκιμή, είτε καλούν τις μεθόδους που παρέχονται από το instrumentation API, οι οποίες προκαλούν γεγονότα από τον κύκλο ζωής και την αλληλεπίδραση του χρήστη, στην εφαρμογή υπό δοκιμή.

Δοκιμή Δραστηριοτήτων

Κύκλος ζωής δραστηριοτήτων και instrumentation

Όταν χρησιμοποιούμε το instrumentation API για να δοκιμάσουμε δραστηριότητες, οι μέθοδοι από τον κύκλο ζωής δεν καλούνται αυτόματα, μόνο η μέθοδος onCreate() καλείται αν καλέσουμε την μέθοδο startActivity(). Μπορούμε να καλέσουμε απευθείας τις υπόλοιπες μεθόδους μέσω των βοηθητικών μεθόδων getInstrumentation().callActivityOn*.

Δοκιμές μονάδων για Δραστηριότητες

Για να δοκιμάσουμε μία Δραστηριότητα σε απομόνωση, μπορούμε να χρησιμοποιήσουμε την κλάση ActivityUnitTestCase.

Αυτή η κλάση μας επιτρέπει να ελέγξουμε την διάταξη καθώς και απομονωμένες μεθόδους μιας Δραστηριότητας, όπως επίσης να ελέγξουμε αν οι προθέσεις (intents) δημιουργούνται κανονικά. Η πρόθεση δεν στέλνεται στο σύστημα, αλλά μπορούμε να χρησιμοποιήσουμε την μέθοδο getStartedActivityIntent() για να αποκτήσουμε πρόσβαση σε μία πιθανή πρόθεση και να επαληθεύσουμε τα δεδομένα.

Η κλάση ActivityUnitTestCase εκκινεί την Δραστηριότητα σε ένα IsolatedContext, δηλαδή η Δραστηριότητα είναι απομονωμένη ως επί το πλείστον από το σύστημα.

Εφόσον αυτή η δοκιμή τρέχει σε ένα IsolatedContext, η δοκιμή πρέπει να εκκινήσει την Δραστηριότητα. Η Δραστηριότητα δεν εκκινείται αυτόματα από το σύστημα.

```
Intent intent = new Intent(getInstrumentation().getTargetContext(),
MainActivity.class);

startActivity(intent, null, null);

// μετά από αυτήν την κλήση μπορούμε να πάρουμε την Δραστηριότητα με την μέθοδο

// getActivity()
```

Δοκιμές ενσωμάτωσης για Δραστηριότητες

Λειτουργικές δοκιμές για μία Δραστηριότητα μπορούν να γραφούν με την κλάση ActivityInstrumentationTestCase2. Αυτή η δοκιμή χρησιμοποιεί την πλήρη υποδομή του συστήματος Android και μας επιτρέπει να αλληλεπιδράσουμε με διαφορετικά στοιχεία. Η επικοινωνία με την υποδομή του συστήματος γίνεται μέσω της κλάσης Instrumentation στην οποία μπορούμε να αποκτήσουμε πρόσβαση μέσω της μεθόδου getInstrumentation(). Αυτή η κλάση μας επιτρέπει να στείλουμε γεγονότα πληκτρολογίου και γεγονότα κλικ.

Αν επιθυμούμε να θέσουμε τιμές απευθείας, πρέπει να χρησιμοποιήσουμε την μέθοδο runOnUiThread() της Δραστηριότητας, δεδομένου ότι μόνο το νήμα διεπαφής χρήστη (UI thread) μπορεί να τροποποιήσει την διεπαφή χρήστη. Αν όλες οι προτάσεις εντός της μεθόδου αλληλεπιδρούν με το νήμα διεπαφής 60

χρήστη, μπορούμε να χρησιμοποιήσουμε το σχόλιο @UiThreadTest στην μέθοδο. Σε αυτήν την περίπτωση δεν μπορούμε να χρησιμοποιήσουμε μεθόδους οι οποίες δεν τρέχουν στο κύριο νήμα / νήμα διεπαφής χρήστη.

Οι δοκιμές οι οποίες βασίζονται στην ActivityInstrumentationTestCase2 εκκινούν την Δραστηριότητα στο τυπικό context, όπως όταν ένας χρήστης εκκινεί την εφαρμογή.

Μπορούμε να χρησιμοποιήσουμε την Instrumentation.invokeMenuActionSync(context, MENU_ID,0) για να προκαλέσουμε μία ενέργεια στο μενού.

Δοκιμή της αρχικής κατάστασης

Αποτελεί καλή πρακτική η δοκιμή της αρχικής κατάστασης της εφαρμογής πριν ξεκινήσει η βασική Δραστηριότητα, έτσι ώστε να είμαστε σίγουροι ότι πληρούνται οι συνθήκες δοκιμής της Δραστηριότητας.

Δοκιμές διαχείρισης της κατάστασης

Προτείνεται να γράψουμε δοκιμές οι οποίες ελέγχουν αν διατηρείται η κατάσταση μιας Δραστηριότητας ακόμα και αν αυτή έχει παυθεί ή τερματιστεί από το σύστημα.

Η κλάση ActivityInstrumentationTestCase2 χρησιμοποιεί την κλάση Instrumentation, η οποία μας επιτρέπει να καλούμε απευθείας τα άγκιστρα στον κύκλο ζωής των Δραστηριοτήτων. Για παράδειγμα, μπορούμε να καλέσουμε τις μεθόδους onPause() και onDestroy(), και στη συνέχεια την onCreate() για να ελέγξουμε αν η κατάσταση διατηρείται.

Δοκιμές της διεπαφής χρήστη

Συνολική δοκιμή της διεπαφής χρήστη

Οι λειτουργικές δοκιμές της διεπαφής χρήστη δοκιμάζουν ολόκληρη την εφαρμογή και όχι μεμονωμένα στοιχεία της.

uiautomator

Το Android SDK περιέχει την βιβλιοθήκη Java *uiautomator* για την δημιουργία δοκιμών της διεπαφής χρήστη και επιπλέον παρέχει μία μηχανή για την εκτέλεση αυτών των δοκιμών. Τα εργαλεία αυτά είναι διαθέσιμα από το API 16 και μετά.

Τα πρότζεκτ δοκιμών που χρησιμοποιούν το uiautomator αποτελούν μεμονωμένα πρότζεκτ Java. Χρησιμοποιούν την βιβλιοθήκη JUnit 3 και επιπλέον τα αρχεία uiautomator.jar και android.jar από τον φάκελο android-sdk/platforms/api-version πρέπει να προστεθούν στη διαδρομή δόμησης (build path).

Το uiautomator παρέχει την κλάση UiDevice για την επικοινωνία με την συσκευή, την κλάση UiSelector για αναζήτηση στοιχείων στην οθόνη, και το UiObject για τα διάφορα στοιχεία της διεπαφής χρήστη. Η κλάση UiCollection επιτρέπει την ταυτόχρονη επιλογή ενός αριθμού στοιχείων της διεπαφής χρήστη, και η κλάση UiScrollable επιτρέπει την κύλιση (scroll) σε κάποιο View για την εύρεση κάποιου στοιχείου.

Google Services

Αξιοποιώντας τα Google Services μπορούμε να εμπλουτίσουμε εύκολα την εφαρμογή μας με διάφορα χαρακτηριστικά, προκειμένου να προσελκύσουμε χρήστες ενός μεγαλύτερου εύρους συσκευών, χρησιμοποιώντας γνωστές υπηρεσίες της Google. Ακολουθεί μία αναφορά ορισμένων εξ' αυτών:

Games: Με τα Google Play Game Services μπορούμε να δώσουμε στους χρήστες των παιχνιδιών μας μια πιο κοινωνική εμπειρία. Υπάρχει η δυνατότητα προσθήκης επιτευγμάτων, πινάκων με τους κορυφαίους παίκτες (leaderboards), multiplayer σε πραγματικό χρόνο, και άλλων δημοφιλών χαρακτηριστικών. Οι παίκτες μπορούν να εισέρχονται χρησιμοποιώντας τα διαπιστευτήρια χρήστη που έχουν στο Google+ και να μοιράζονται την εμπειρία τους από το παιχνίδι με φίλους.

Location: Με τα location APIs είναι ευκολότερη η δημιουργία εφαρμογών που γνωρίζουν τη θέση του χρήστη, εφόσον δεν χρειάζεται ο προγραμματιστής να επικεντρωθεί στις λεπτομέρειες της τεχνολογίας που χρησιμοποιείται για τον εντοπισμό της θέσης (location technology). Επιπλέον, επιτρέπουν την ελαχιστοποίηση της κατανάλωσης ενέργειας χρησιμοποιώντας όλες τις δυνατότητες του υλικού της συσκευής.

 Geofencing:

Επιτρέπει στην εφαρμογή μας να θέσει γεωγραφικά όρια γύρω από συγκεκριμένες τοποθεσίες και να δέχεται ειδοποιήσεις όταν ο χρήστης μπαίνει ή φεύγει από αυτές τις περιοχές.

 Activity recognition:

Είναι χρήσιμο για μία εφαρμογή να γνωρίζει τι κάνει ο χρήστης εκείνη την στιγμή προκειμένου να εμφανίσει το κατάλληλο περιεχόμενο. Με το Activity recognition API είναι εύκολος ο έλεγχος της τρέχουσας δραστηριότητας του χρήστη—κάθεται ακίνητος, περπατάει, κάνει ποδήλατο, είναι σε κάποιο όχημα—με πολύ αποτελεσματική χρήση της μπαταρίας.

Maps: Το Google Maps API μας επιτρέπει να χρησιμοποιήσουμε στην εφαρμογή μας τους χάρτες που παρέχει η Google και να τους προσαρμόσουμε με δείκτες (markers), να ελέγξουμε την οπτική του χρήστη, και να σχεδιάσουμε γραμμές και σχήματα.

Google+: Χρησιμοποιώντας το Google+ για το Android, μπορούμε να πιστοποιήσουμε τον χρήστη της εφαρμογής μας. Αφού γίνει η πιστοποίηση,

μπορούμε επιπλέον να έχουμε πρόσβαση στο δημόσιο προφίλ και στο κοινωνικό γράφημα του χρήστη.

Drive: Με το Google Drive Android API μπορούμε να δώσουμε στους χρήστες πρόσβαση στα αρχεία τους, όπου και αν βρίσκονται, από οποιαδήποτε συσκευή. Το API κάνει εύκολη την αποθήκευση και τον συγχρονισμό των αρχείων του χρήστη μεταξύ της συσκευής και του cloud.

Google Play In-app Billing: Με το Google Play In-app Billing είναι εφικτή η πώληση ψηφιακού περιεχομένου από τις εφαρμογές μας. Αυτό το ψηφιακό περιεχόμενο μπορεί να πωληθεί με μία και μοναδική χρέωση ή με διαφορετικές χρεώσεις οι οποίες μπορεί να γίνονται και μέσω συνδρομών. Μπορούμε να χρησιμοποιήσουμε αυτήν την υπηρεσία για να πουλήσουμε μία μεγάλη ποικιλία περιεχομένου, στο οποίο περιλαμβάνεται περιεχόμενο το οποίο μπορεί να κατεβάσει ο χρήστης, όπως αρχεία πολυμέσων ή φωτογραφίες, εικονικό περιεχόμενο όπως επίπεδα παιχνιδιών ή φίλτρα, premium υπηρεσίες και χαρακτηριστικά κ.ά.

Ads: Το Google Mobile Ads επιτρέπει την ενσωμάτωση διαφημίσεων σε μία εφαρμογή. Υποστηρίζει μία μεγάλη ποικιλία διαφημίσεων χάρη στο μεγάλο όγκο των διαφημιστών που συνεργάζονται με την Google (πάνω από ένα εκατομμύριο). Υπάρχει η δυνατότητα προβολής διαφημίσεων με βάση την τοποθεσία που βρίσκεται ο χρήστης (location-based ads), οι οποίες μπορούν να αποφέρουν περισσότερα έσοδα σε σχέση με τις «συμβατικές» διαφημίσεις.

Google Cloud Messaging (**GCM**): Το GCM για Android επιτρέπει την επικοινωνία ανάμεσα στον διακομιστή μας και την εφαρμογή μέσω ασύγχρονων μηνυμάτων. Δεν χρειάζεται να μας απασχολεί ο χειρισμός χαμηλού επιπέδου (low level) πτυχών αυτής της επικοινωνίας, όπως είναι η κατασκευή των μηνυμάτων και της ουράς. Χρησιμοποιώντας αυτήν την υπηρεσία, μπορούμε να υλοποιήσουμε εύκολα ένα σύστημα ειδοποιήσεων για την εφαρμογή μας.

Το 2012 η Google άρχισε να διαχωρίζει τα Google Services από το Android, παρέχοντας με αυτό τον τρόπο τη δυνατότητα αναβάθμισης τους από το Google Play Store χωρίς να απαιτείται αναβάθμιση του λειτουργικού.

Πώς δουλεύουν τα Google Play Services

Όταν η Google παρουσίασε τα Google Play Services στο Google I/O 2012, είπε ότι η πλατφόρμα αποτελείται από ένα στοιχείο υπηρεσιών το οποίο τρέχει στην συσκευή και μία βιβλιοθήκη πελάτη (client library) η οποία περιλαμβάνεται στην εφαρμογή.

Αυτό σημαίνει ότι τα Google Play Services δουλεύουν χάρη σε δύο βασικά στοιχεία: την βιβλιοθήκη πελάτη και το Google Play Services APK.

Βιβλιοθήκη πελάτη: Η βιβλιοθήκη πελάτη των Google Play Services περιλαμβάνει τις διασυνδέσεις (interfaces) που χρησιμοποιούνται από την εφαρμογή για κάθε Google Service. Η βιβλιοθήκη επιτρέπει στους χρήστες να εξουσιοδοτήσουν την εφαρμογή με πρόσβαση σε αυτά τα services χρησιμοποιώντας τα διαπιστευτήριά τους. Η βιβλιοθήκη πελάτη αναβαθμίζεται από την Google με καινούρια χαρακτηριστικά και υπηρεσίες. Μπορούμε να αναβαθμίσουμε την βιβλιοθήκη παρέχοντας μία νεότερη έκδοση της εφαρμογής 63

μας, το οποίο βέβαια δεν είναι απαραίτητο εάν δεν συμπεριλαμβάνουμε τα νεότερα χαρακτηριστικά που υποστηρίζει.

Google Play Services APK: Το Google Play Services Android Package (APK) τρέχει παρασκηνιακά (background) σαν υπηρεσία στο Android. Χρησιμοποιώντας την βιβλιοθήκη πελάτη, η εφαρμογή αποκτάει πρόσβαση σε αυτήν την υπηρεσία, η οποία είναι αυτή που εκτελεί τις διάφορες ενέργειες ενώ τρέχει η εφαρμογή. Το APK δεν είναι εγγυημένο ότι θα είναι εγκατεστημένο σε όλες τις συσκευές. Σε περίπτωση που δεν είναι εγκατεστημένο μπορεί κάποιος χρήστης να το προμηθευτεί από το Google Play Store. Κατ' αυτό τον τρόπο, δεν χρειάζεται να αναβαθμίζουμε την εφαρμογή μας κάθε φορά που αναβαθμίζονται τα Google Play Services από την Google. Εφόσον το Google Play services APK παρέχεται μέσω του Google Play Store, οι αναβαθμίσεις που δέχονται οι υπηρεσίες δεν εξαρτώνται από τις αναβαθμίσεις που γίνονται στο λειτουργικό από τους κατασκευαστές ή τους παρόχους κινητής τηλεφωνίας. Προκειμένου να αναβαθμίζεται αυτόματα το Google Play Services APK μέσω του Google Play Store, πρέπει να είναι εγκατεστημένη η εφαρμογή Google Play Store και η έκδοση του Android να είναι η 2.3 ή κάποια νεότερη. Μολονότι τα Google Play Services δεν αποτελούν μέρος της πλατφόρμας Android, υποστηρίζονται από τις περισσότερες συσκευές Android. Όποια συσκευή έχει εγκατεστημένη την έκδοση 2.2 του Android ή κάποια νεότερη, μπορεί να εγκαταστήσει οποιαδήποτε εφαρμογή που κάνει χρήση των Google Play Services.

CODE COMMENTS

determine the space between first two fingers

com.protectsoft.simplecam. CameraPreview. getFingerSpacing

//check if shutter sound can be disabled

// In fact, in many places (most of Europe, Japan, parts of US, etc.) it's illegal to disable the shutter sound

com.protectsoft.simplecam. CameraFeatures. setDevicehardwareSupport

CharSequence[] of all flash modes supported by camera

* or null if flash is not supported

com.protectsoft.simplecam. CameraFeatures. getSupportedFlashoptions

CharSequence[] of all camera supported effects

* or null if camera dosent support effects

com.protectsoft.simplecam. CameraFeatures. getSupportedCameraEffects

CharSequence[] of all the scene modes the camera supports

* or null if camera dont support scene modes

com.protectsoft.simplecam. CameraFeatures. getSupportedCameraScenes

CharSequence[] of all the focus modes supported by camera

* or null if no focus modes supported

com.protectsoft.simplecam. CameraFeatures. getSupportedFocusModes

the cameras supported resolutions/megapixels

* in the format of 1920x1080. or null

com.protectsoft.simplecam. CameraFeatures. getSupportedPictureSizes

CharSequence[] of the whitebalances supported by camera

* or null if none is supported

com.protectsoft.simplecam. CameraFeatures. getSupportedWhiteBalances

@param data the byte[] array that contains the image

* @param reqWidth the imageview width

* @param reqHeight the imageview height

* @return fixed size bitmap for the given imageview

com.protectsoft.simplecam.Utils. BitmapUtils. decodeSampledBitmapFromByteArray

@param res the filepath to the image

* @param reqWidth the imageview width

* @param reqHeight the imageview height

* @return fixed size bitmap for the given imageview

com.protectsoft.simplecam.Utils. BitmapUtils. decodeSampledBitmapFromFile

com.protectsoft.simplecam.Utils. BitmapUtils. calculateInSampleSize

@param options the BitmapFactoryOptions
** @param reqWidth the imageview width*
** @param reqHeight the imageview height*
** @return int how many times the origila bitmap's size will be divided to be fit to the imageview*

*/**
 * initialize create folders/file
 /

com.protectsoft.simplecam.Utils. MediaFileUtils. initializeFolder

* *
** returns the path + filename ! isn't creating it!*
** *
** @param type file will be image or video/*
** @return the File name with the path.*

com.protectsoft.simplecam.Utils. MediaFileUtils. getOutputMediaFile

@return home file path

com.protectsoft.simplecam.Utils. MediaFileUtils. getHomeFile

*two possible Throwables! OutOfMemoryError Exception(when dalvik runs out of memory because of big bitmaps data) ->subclass of Error
*
** and Unsatisfiedlinkerror ->subclass of Error if failed to load cpp library <-this comes from JniBitmapHolder*
** @param rotation the current rotation of the camera ,for the picture taken from camera to be rotated properly*
** @param data the byte[] array image captured from camera*
** @param picturefile the path+imagename to be saved*
** @param con application context*
** @throws Throwable to inform calling method by catching it and taking different action(edit image with java instead cpp)
*

com.protectsoft.simplecam.Utils. MediaFileUtils. jnirotateImageAndWriteToFile

```
//android bug
//refresh saved picture to internall storage otherwise it will not be accessible from usb
MediaScannerConnection.scanFile(con, new String[]{picturefile.getAbsolutePath()}, null, null);
Constants.lastpicturefiletaken = picturefile;
   BucketFiles.addPictureFile(picturefile);


//throwable catched becase cpp/so library failed to load!
} catch (Throwable ex) {
   //in that case throw new throwable to inform calling method ny catching it and taking different action(edit image with java instead cpp)
   // or OutOfMemoryError
   throw new Throwable();
```

```
<p>
 *  usealy this method gets called if the above throw Exception
 * </p>
 *
 * @param rotation the current rotation of the camera ,for the picture taken from camera to be rotated properly
 * @param data the byte[] array image captured from camera
 * @param picturefile the path+imagename to be saved
 * @param con application context
 *  @see #jnirotateImageAndWriteToFile
```

com.protectsoft.simplecam.Utils. MediaFileUtils. rotateImageAndWriteToFile

```
<p>
*     this method havent tested!
* </p>
* @param dp the imageview dp
```

** @param density phones screen density*
** @return the right pixel value for the imageview
*

com.protectsoft.simplecam.Utils. MediaFileUtils. dpToPx

*@param data the data as a byte[] array to be tested if there is enough free space in the folder to be saved
*
** compares the given data plus 1mb for safety*
** @return true false*

com.protectsoft.simplecam.Utils. MediaFileUtils. isFreeSpaceAvailable

@param profileQuality the camera quality
** @return the birate of camera record video*
** used in recording video for calculating free space*

com.protectsoft.simplecam.Utils. MediaFileUtils. getBitrate

* *
** this method havent tested in many devices*
** *
** @param orientation the current orientation of device*
** @return 0,90,180,270
*

com.protectsoft.simplecam.Utils. OrientationUtils. getCurrentOrientation

@param orientation the current device's orientation
** @return return only LinearLayout.HORIZONTAL 0 or LinearLayout.VERTICAL 1;*

com.protectsoft.simplecam.Utils. OrientationUtils. getHorizontalOrVertical

* *
** sets the camera DisplayOrientation for the preview*
** *
***
** @param cameraid 0 or 1 usualy front or back camera*
** @param cm the camera instance*
** @param display*

com.protectsoft.simplecam.Utils. OrientationUtils. setCameraDisplayOrientation

Extends Android ImageView to include pinch zooming, panning, fling and double tap zoom.

com.protectsoft.simplecam.PreviewImage. TouchImageView

```
//
// SuperMin and SuperMax multipliers. Determine how much the image can be
// zoomed below or above the zoom boundaries, before animating back to the
// min/max zoom boundary.
//
private static final float SUPER_MIN_MULTIPLIER = .75f;
private static final float SUPER_MAX_MULTIPLIER = 1.25f;

//
   // Scale of image ranges from minScale to maxScale, where minScale == 1
   // when the image is stretched to fit view.
   //
   private float normalizedScale;

   //
   // Matrix applied to image. MSCALE_X and MSCALE_Y should always be equal.
   // MTRANS_X and MTRANS_Y are the other values used. prevMatrix is the
matrix
   // saved prior to the screen rotating.
   //
private Matrix matrix, prevMatrix;

//
// Size of view and previous view size (ie before rotation)
```

//
private int viewWidth, viewHeight, prevViewWidth, prevViewHeight;

//
// Size of image when it is stretched to fit view. Before and After rotation.
//
private float matchViewWidth, matchViewHeight, prevMatchViewWidth, prevMatchViewHeight;

com.protectsoft.simplecam.PreviewImage. TouchImageView

Returns false if image is in initial, unzoomed state. False, otherwise.
** @return true if image is zoomed*

com.protectsoft.simplecam.PreviewImage. TouchImageView. isZoomed

Return a Rect representing the zoomed image.
** @return rect representing zoomed image*

com.protectsoft.simplecam.PreviewImage. TouchImageView. getZoomedRect

Save the current matrix and view dimensions
** in the prevMatrix and prevView variables.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. savePreviousImageValues

Get the max zoom multiplier.
** @return max zoom multiplier.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. getMaxZoom

Set the max zoom multiplier. Default value: 3.
** @param max max zoom multiplier.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. setMaxZoom (float max)

Get the min zoom multiplier.
** @return min zoom multiplier.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. getMinZoom

Get the current zoom. This is the zoom relative to the initial
** scale, not the original resource.*
** @return current zoom multiplier.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. getCurrentZoom

Set the min zoom multiplier. Default value: 1.
** @param min min zoom multiplier.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. setMinZoom(float min)

Reset zoom and translation to initial state.

com.protectsoft.simplecam.PreviewImage. TouchImageView. resetZoom

Set zoom to the specified scale. Image will be centered by default.
** @param scale*

com.protectsoft.simplecam.PreviewImage. TouchImageView. setZoom(float scale)

Set zoom to the specified scale. Image will be centered around the point
** (focusX, focusY). These floats range from 0 to 1 and denote the focus point*
** as a fraction from the left and top of the view. For example, the top left*
** corner of the image would be (0, 0). And the bottom right corner would be (1, 1).*
** @param scale*
** @param focusX*
** @param focusY*

com.protectsoft.simplecam.PreviewImage. TouchImageView. setZoom(float scale, float focusX, float focusY)

Set zoom to the specified scale. Image will be centered around the point
** (focusX, focusY). These floats range from 0 to 1 and denote the focus point*
** as a fraction from the left and top of the view. For example, the top left*
** corner of the image would be (0, 0). And the bottom right corner would be (1, 1).*
** @param scale*
** @param focusX*
** @param focusY*
** @param scaleType*

com.protectsoft.simplecam.PreviewImage. TouchImageView. setZoom(float scale, float focusX, float focusY, ScaleType scaleType)

Set zoom parameters equal to another TouchImageView. Including scale, position,
** and ScaleType.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. setZoom(TouchImageView img)

Return the point at the center of the zoomed image. The PointF coordinates range
** in value between 0 and 1 and the focus point is denoted as a fraction from the left*
** and top of the view. For example, the top left corner of the image would be (0, 0).*
** And the bottom right corner would be (1, 1).*
** @return PointF representing the scroll position of the zoomed image.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. getScrollPosition

Set the focus point of the zoomed image. The focus points are denoted as a fraction from the
** left and top of the view. The focus points can range in value between 0 and 1.*
** @param focusX*
** @param focusY*

com.protectsoft.simplecam.PreviewImage. TouchImageView. setScrollPosition(float focusX, float focusY)

Performs boundary checking and fixes the image matrix if it
** is out of bounds.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. fixTrans

When transitioning from zooming from focus to zoom from center (or vice versa)
** the image can become unaligned within the view. This is apparent when zooming*
** quickly. When the content size is less than the view size, the content will often*
** be centered incorrectly within the view. fixScaleTrans first calls fixTrans() and*
** then makes sure the image is centered correctly within the view.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. fixScaleTrans

If the normalizedScale is equal to 1, then the image is made to fit the screen. Otherwise,
** it is made to fit the screen according to the dimensions of the previous image matrix. This*
** allows the image to maintain its zoom after rotation.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. fitImageToView

Set view dimensions based on layout params
***
** @param mode*
** @param size*
** @param drawableWidth*

com.protectsoft.simplecam.PreviewImage. TouchImageView. setViewSize

After rotating, the matrix needs to be translated. This function finds the area of image
** which was previously centered and adjusts translations so that is again the center, post-rotation.*
***
** @param axis Matrix.MTRANS_X or Matrix.MTRANS_Y*
** @param trans the value of trans in that axis before the rotation*
** @param prevImageSize the width/height of the image before the rotation*
** @param imageSize width/height of the image after rotation*
** @param prevViewSize width/height of view before rotation*

** @param viewSize width/height of view after rotation*
** @param drawableSize width/height of drawable*

com.protectsoft.simplecam.PreviewImage. TouchImageView. translateMatrixAfterRotate(int axis, float trans, float prevImageSize, float imageSize, int prevViewSize, int viewSize, int drawableSize)

Gesture Listener detects a single click or long click and passes that on
** to the view's listener.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. GestureListener

Responsible for all touch events. Handles the heavy lifting of drag and also sends
** touch events to Scale Detector and Gesture Detector.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. PrivateOnTouchListener

ScaleListener detects user two finger scaling and scales image.

com.protectsoft.simplecam.PreviewImage. TouchImageView. ScaleListener

DoubleTapZoom calls a series of runnables which apply
** an animated zoom in/out graphic to the image.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. DoubleTapZoom

Interpolate between where the image should start and end in order to translate
** the image so that the point that is touched is what ends up centered at the end*
** of the zoom.*
** @param t*

com.protectsoft.simplecam.PreviewImage. TouchImageView. translateImageToCenterTouchPosition(float t)

Interpolate the current targeted zoom and get the delta
** from the current zoom.*
** @param t*

com.protectsoft.simplecam.PreviewImage. TouchImageView. calculateDeltaScale

This function will transform the coordinates in the touch event to the coordinate
** system of the drawable that the imageview contain*
** @param x x-coordinate of touch event*
** @param y y-coordinate of touch event*
** @param clipToBitmap Touch event may occur within view, but outside image content. True, to clip return value*
** to the bounds of the bitmap size.*
** @return Coordinates of the point touched, in the coordinate system of the original drawable.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. transformCoordTouchToBitmap(float x, float y, boolean clipToBitmap)

Inverse of transformCoordTouchToBitmap. This function will transform the coordinates in the
** drawable's coordinate system to the view's coordinate system.*
** @param bx x-coordinate in original bitmap coordinate system*
** @param by y-coordinate in original bitmap coordinate system*
** @return Coordinates of the point in the view's coordinate system.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. transformCoordBitmapToTouch(float bx, float by)

Fling launches sequential runnables which apply
** the fling graphic to the image. The values for the translation*
** are interpolated by the Scroller.*

com.protectsoft.simplecam.PreviewImage. TouchImageView. Fling

native class and method calling c++ methods from jni/JniBitmapOperationsLibrary.cpp for heavy image processing

com.protectsoft.simplecam.jni.bitmapoperations. JniBitmapHolder

*flips a bitmap horizontally, as such:
*
*
* *<pre>*
* *123 321*
* *456 => 654*
* *789 987*
* *</pre>*

com.protectsoft.simplecam.jni.bitmapoperations. JniBitmapHolder. flipBitmapHorizontal

*Flips the bitmap on the vertically, as such:
*
*
* *<pre>*
* *123 789*
* *456 => 456*
* *789 123*
* *</pre>*

com.protectsoft.simplecam.jni.bitmapoperations. JniBitmapHolder. flipBitmapVertical

Returns the color of the cluster. If options.enableDominantColor is true, return the
* *dominant color around the provided point. Return the color of the point itself otherwise.*
* *The dominant color algorithm is based on simple counting search, so use with caution.*
*
* *@param pixels the bitmap*
* *@param pixelX the x coordinate of the reference point*
* *@param pixelY the y coordinate of the reference point*
* *@param opts additional options*
* *@return the color of the cluster*

com.protectsoft.simplecam.editimage. getPixelColor(@NonNull Bitmap pixels, int pixelX, int pixelY, @NonNull PixelateLayer opts)

arraylist with all the filePath picture taken and saved from camera capture

com.protectsoft.simplecam.bucket. BucketFiles

//// TODO: delete not working properly
//on android 4.2 and above delete is disabled!
// actualy remove delete functionality! leave it for photo album apps and etc.

com.protectsoft.simplecam.bucket. BucketFiles. deleteFileFromFileSystem

Μαθαίνοντας Design Patterns Model – View – ViewModel

Τί είναι το MVVM;

Το Model – View – ViewModel είναι ένα πρότυπο σχεδίασης για το σχεδιασμό διεπαφών χρήστη, το οποίο έχει επηρροές τόσο από το Model View Presenter, όσο και από το Model View Controller. Είναι γνωστό και ως Presentation Model (σχεδόν ίδιο)(PM στο εξής), όπως το έχει καταγράψει ο Martin Fowler. Το Model – View – ViewModel (MVVM στο εξής) είναι επίσης ένας καθιερωμένος τρόπος, για τη δημιουργία εφαρμογών που βασίζονται και σε XAML και επιλέχθηκε για την ευελιξία που παρέχει στον προγραμματιστή κατά την ανάπτυξη (decoupling View-Model). Το design pattern που περιγράφεται στη συνέχεια με τη βοήθεια του παραδείγματος σε silverlight, παρουσιάζεται έχοντας κατα νου το ισχυρό πλαίσιο databinding του silverlight (η αντιστοιχία στοιχείου οθόνης με κανονικό Property ενός object).

Ξεκινώντας την περιγραφή του MVVM, αναφέρουμε ότι παρέχει (όπως και το PM) ένα επίπεδο αφαίρεσης του View (της εικόνας δηλαδή που φαίνεται στον χρήστη / XAML) στο οποίο ορίζεται τόσο η κατάστασή του, όσο και η συμπεριφορά του. Έστω ότι έχουμε την οθόνη διαχείρισης καταστημάτων μίας εταιρίας πώλησης παπουτσιών. Υπάρχουν δύο εκδόσεις για την εν λόγω οθόνη, μία που παρουσιάζει όλα τα καταστήματα (datagrid με μία λίστα) και μία που ενεργοποιείται όταν αλλάζουμε ιδιότητες από κάποιο κατάστημα (αρκετά controls, όπως textboxes κτλ με τα οποία μπορούμε να καταχωρούμε πληροφορίες για ένα selected store από την προηγούμενη σελίδα). Οι δύο οθόνες λέγονται αντίστοιχα Master και Detail σελίδες. Έστω ότι έχουμε την Detail σελίδα να αναπτύξουμε. Στη συνέχεια περιγράφεται πώς το MVVM μπορεί να διαχωρίσει τις αρμοδιότητες, για διαφορετικές ενέργειες που λαμβάνουν χώρα πίσω από την οθόνη.

Τα τρία στοιχεία που περιγράφονται είναι τρία διαφορετικά αντικείμενα στη μνήμη και έχουν πεδία και μεθόδους. Έτσι λοιπόν έχουμε το View (τονίζω πάλι ότι

αναφερόμαστε σε XAML αρχείο) το οποίο περιέχει διάφορα controls με τα οποία μπορεί να αλληλεπιδράσει ο χρήστης. Κάθε control που περιέχεται, μπορεί να εμφανίσει πληροφορία είτε σε μορφή κειμένου είτε σε κάτι πιο πολυμεσικό.

Έστω ότι έχουμε ένα πεδίο κειμένου (textboxes) και έναν επιλογέα με μία μοναδική επιλογή κάθε φορά (combobox). Τα δύο αυτά controls, δένουν τις τιμές τους με ιδιότητες (properties) του ViewModel μέσω μηχανισμού databinding. Προγραμματιστικά το databinding ανάμεσα στο View και στο ViewModel (που περιέχει όλη την πληροφορία που δείχνει/ανανεώνει/… το View) γίνεται πάρα πολύ εύκολα, μιας και θέτουμε το ViewModel αντικείμενο, ως το πλαίσιο δεδομένων (datacontext) του View (βλ. παράδειγμα). Αν κάποια τιμή, σε κάποιο property αλλάξει, τότε η ίδια ιδιότητα ενεργοποιεί ένα γεγονός (κάνει raise ένα event propertychanged) που υποδηλώνει ότι άλλαξε τιμή και έτσι ενημερώνει το View ότι άλλαξε η τιμή για το συγκριμένο user control (για παράδειγμα ένα textbox – πεδίο εισαγωγής κειμένου). Κάθε πληροφορία δηλαδή, που είναι databound με ένα control όταν αλλάξει, ενημερώνει το UI ότι άλλαξε και ότι θα πρέπει να ανανεωθεί. Για να μπορεί ένα αντικείμενο (το ViewModel στην περίπτωσή μας) να αλληλεπιδρά με αυτόν τον τρόπο θα πρέπει να υλοποιεί το INotifyPropertyChanged interface. Έτσι λοιπόν ανανεώνεται και η αντίστοιχη πληροφορία στον χρήστη. Σε αντίθετη περίπτωση, όταν ένας χρήστης πατήσει ένα κουμπί, μία εντολή (Commanding) θα εκτελεστεί στο ViewModel και θα πραγματοποιήσει όλες τις αλλαγές στα δεδομένα του μοντέλου (τα οποία περιέχονται στο ίδιο το αντικείμενο του ViewModel).

Το Model είναι το αντικείμενο το οποίο καλείται από το ViewModel για λειτουργίες πρόσβασης στα δεδομένα. Στην περίπτωσή μας έχουμε δομικές μονάδες που περιγράφουν το πεδίο της εφαρμογής (customers, orders, κτλ). Αξίζει να κρατήσουμε τρία πράγματα:

- Το View ΔΕΝ γνωρίζει το Model (γιατί δεν απαιτείται κάτι τέτοιο εννοιολογικά)
- Το ViewModel ΔΕΝ γνωρίζει τίποτα για View (εξαρτάται, ίσως είναι θέμα προθέσεων του προγραμματιστή)
- Το Model ΔΕΝ γνωρίζει τίποτα για View

Πρότυπα Σχεδίασης

Όλα ξεκίνησαν από …..

την Αρχιτεκτονική !!

- Christopher Alexander, *The Timeless Way of Building*, Oxford University Press, New York, 1979: "Είναι η ποιότητα μία αντικειμενική ιδιότητα? "
- Αν αποδεχθεί κανείς ότι είναι δυνατό να αναγνωρίσει και να περιγράψει ένα σχέδιο καλής ποιότητας τότε:

(Alexander): Τι υπάρχει σε ένα σχέδιο καλής ποιότητας το οποίο δεν υπάρχει σε ένα σχέδιο κακής ποιότητας ?

- Ο Alexander μελετώντας πληθώρα αρχιτεκτονικών κατασκευασμάτων, παρατήρησε ότι οι καλές κατασκευές είχαν κοινά στοιχεία μεταξύ τους
- Τα κοινά στοιχεία συνήθως αφορούν κοινές λύσεις ή λύσεις σε κοινά προβλήματα
- Ο Alexander κατανόησε ότι οι δομές δεν μπορούσαν να διαχωριστούν από το πρόβλημα το οποίο προσπαθούν να επιλύσουν. Για το λόγο αυτό αναζήτησε διαφορετικές δομές που σχεδιάστηκαν για να επιλύσουν το ίδιο πρόβλημα
- Ο Alexander όρισε την έννοια του προτύπου ως "Μία λύση ενός προβλήματος μέσα σε συγκεκριμένο πλαίσιο" (…κάθε πρότυπο περιγράφει ένα πρόβλημα που εμφανίζεται συνέχεια στο περιβάλλον και στη συνέχεια περιγράφει τον πυρήνα της λύσης κατά τέτοιο τρόπο ώστε η λύση να μπορεί να εφαρμοστεί εκατομμύρια φορές)

Κάθε πρότυπο περιλαμβάνει:

- το όνομα του προτύπου

- το σκοπό του προτύπου, το πρόβλημα που επιλύει
- τον τρόπο επίλυσης
- τους περιορισμούς που πρέπει να ληφθούν υπόψη

Adapter (Προσαρμογέας)

- Κατηγορία: Structural
- Σκοπός: *Η μετατροπή της διασύνδεσης μιας κλάσης σε μία άλλη που αναμένει το πρόγραμμα πελάτης*. *Ο προσαρμογέας επιτρέπει τη συνεργασία κλάσεων, η οποία σε διαφορετική περίπτωση θα ήταν αδύνατη λόγω ασύμβατων διασυνδέσεων.*
- Συνώνυμα: Wrapper

Συχνά ο κώδικας μιας κλάσης προσφέρεται για επαναχρησιμοποίηση, αλλά αυτή δεν είναι δυνατή λόγω του ότι τα προγράμματα που επιθυμούν να χρησιμοποιήσουν τις λειτουργίες της, αναμένουν διαφορετική διασύνδεση.

Έστω ότι μία κλάση Σχεδίασης είναι σε θέση να σχεδιάσει γραμμές, αλλά απαιτεί ως παραμέτρους τις συντεταγμένες στη μορφή (x1, y1, x2, y2) ενώ τα προγράμματα πελάτες είναι σε θέση να παρέχουν τις συντεταγμένες στη μορφή (x1, x2, y1, y2).

Συνήθως τα προγράμματα πελάτες δεν είναι δυνατόν να τροποποιηθούν (καθώς βρίσκονται ήδη εγκατεστημένα).

Η κλάση Σχεδίασης είναι επιθυμητό να χρησιμοποιηθεί χωρίς τροποποίηση (καθώς οποιαδήποτε επέμβαση στον κώδικα μιας μεθόδου είναι δυνατόν να προκαλέσει σφάλματα στις υπόλοιπες μεθόδους).

Εδώ βρίσκει εφαρμογή το πρότυπο "Προσαρμογέας".

Bridge (Γέφυρα)

- Κατηγορία: Structural
- Σκοπός: *Η αποσύνδεση μιας αφαίρεσης από την υλοποίησή της, ώστε να μπορούν να μεταβάλλονται ανεξάρτητα .*

- Συνώνυμα: Handle/Body
- Όταν μία αφαίρεση μπορεί να έχει περισσότερες από μία υλοποιήσεις, ο συνήθης τρόπος οργάνωσης είναι με τη χρήση κληρονομικότητας.
- Με τον όρο αφαίρεση νοείται μία αφηρημένη κλάση που ορίζει μία διασύνδεση (ένα σύνολο υπογραφών), ενώ υλοποιήσεις είναι οι συγκεκριμένες παράγωγες κλάσεις οι οποίες υλοποιούν τις μεθόδους της αφηρημένης κλάσεις.
- Πρόβλημα: Αφαίρεση και υλοποιήσεις συνδέονται μόνιμα, καθιστώντας δύσκολη την επέκταση, τροποποίηση και επαναχρησιμοποίηση αφαιρέσεων και υλοποιήσεων ανεξάρτητα.

Το πρότυπο "Γέφυρα" είναι σχετικά δύσκολο στην κατανόησή του· Χρησιμοποιείται ωστόσο σε πληθώρα περιπτώσεων όπου εντοπίζονται:

- Μεταβολές στην αφαίρεση μιας έννοιας
- Μεταβολές στον τρόπο υλοποίησης της έννοιας αυτής

Η Γέφυρα αντίκειται επίσης στη συνήθη τάση χειρισμού αντίστοιχων καταστάσεων μόνο με κληρονομικότητα. Ικανοποιεί όμως δύο από τους βασικούς κανόνες της αντικειμενοστρεφούς κοινότητας: "Εντοπίστε αυτό που μεταβάλλεται και ενσωματώστε το", και "προτιμήστε σύνθεση αντικειμένων από κληρονομικότητα κλάσεων".

Singleton (Μοναδιαίο)

- Κατηγορία: Creational

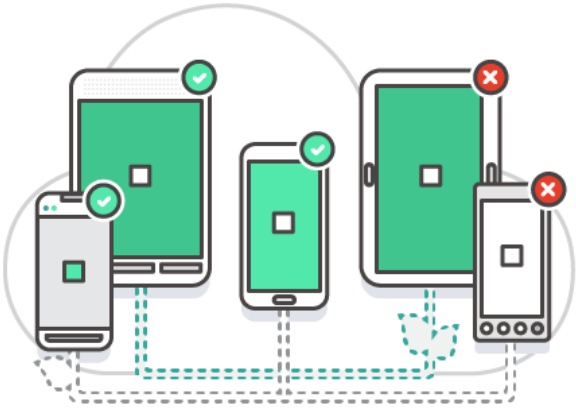
Σκοπός: *Η εξασφάλιση ότι μία κλάση θα έχει μόνο ένα στιγμιότυπο και παρέχει ένα καθολικό σημείο πρόσβασης σε αυτό*

Κλάσεις και στιγμιότυπα: Σχέση ένα-προς-πολλά. Τα αντικείμενα δημιουργούνται δεσμεύοντας χώρο στη μνήμη όποτε κρίνεται σκόπιμο και διαγράφονται από τη μνήμη όταν τερματιστεί η χρήση τους.

Ορισμένες φορές, απαιτείται η ύπαρξη κλάσεων από τις οποίες παράγεται ένα μόνο αντικείμενο. Το αντικείμενο αυτό συνήθως δημιουργείται κατά την έναρξη της εφαρμογής και διαγράφεται με το πέρας της.

Ο ρόλος του μοναδικού αυτού αντικειμένου είναι η διαχείριση των υπολοίπων αντικειμένων της εφαρμογής και για το λόγο αυτό, αποτελεί λογικό σφάλμα να δημιουργηθούν περισσότερα του ενός τέτοια αντικείμενα-διαχειριστές (managers ή controllers).

ΜΕΛΈΤΗ ΠΕΡΊΠΤΩΣΗΣ AWS DEVICE FARM APP CLOUD TESTING



https://aws.amazon.com/device-farm/

Τι είναι το Farm AWS Device

Το Device Farm είναι μια υπηρεσία δοκιμών εφαρμογών που σας δίνει τη δυνατότητα να δοκιμάσετε και να αλληλεπιδράσετε με τις εφαρμογές σας Android, iOS και Web σε πραγματικά, φυσικά τηλέφωνα και tablet που φιλοξενούνται από την υπηρεσία Amazon Web Services (AWS). Υπάρχουν δύο βασικοί τρόποι χρήσης του Device Farm:

Αυτοματοποιημένη δοκιμή εφαρμογών χρησιμοποιώντας ποικίλα διαθέσιμα πλαίσια δοκιμών

Απομακρυσμένη πρόσβαση συσκευών στις οποίες μπορείτε να φορτώνετε, να εκτελείτε και να αλληλεπιδράτε με εφαρμογές σε πραγματικό χρόνο.

Το Farm Farm σας επιτρέπει να ανεβάσετε τις δικές σας δοκιμές ή να χρησιμοποιήσετε ενσωματωμένες δοκιμές συμβατότητας χωρίς δέσμες ενεργειών. Επειδή οι δοκιμές εκτελούνται αυτόματα παράλληλα, οι δοκιμές σε πολλές συσκευές ξεκινούν μέσα σε λίγα λεπτά.

Μια αναφορά δοκιμής που περιέχει αποτελέσματα υψηλού επιπέδου, αρχεία καταγραφής χαμηλού επιπέδου, στιγμιότυπα από pixel σε pixel και δεδομένα απόδοσης ενημερώνονται με την ολοκλήρωση των δοκιμών.

Το Device Farm υποστηρίζει τη δοκιμή των εγγενών και υβριδικών εφαρμογών Android, iOS και Fire OS, συμπεριλαμβανομένων εκείνων που δημιουργούνται με το PhoneGap, το Titanium, το Xamarin, το Unity και άλλα πλαίσια. Υποστηρίζει την απομακρυσμένη πρόσβαση των εφαρμογών Android για διαλογικές δοκιμές.

Υποστηριζόμενους τύπους δοκιμών και ενσωματωμένες δοκιμές

Το Farm Farm παρέχει επί του παρόντος υποστήριξη για τους ακόλουθους τύπους δοκιμών:

Για Android:

- Appium Java JUnit
- Appium Java TestNG
- Appium Python
- Τσότρα
- Συσκευές (JUnit, Espresso, Robotium ή οποιαδήποτε τεστ με όργανα)
- UI Automator
- Εξερευνητής
-

Για το iOS:

- Appium Java JUnit
- Appium Java TestNG
- Appium Python
- Τσότρα

- UI Automation
- XCTest (συμπεριλαμβανομένου του KIF)
- XCTest UI

Για εφαρμογές Web:

- Appium Java JUnit
- Appium Java TestNG
- Appium Python

Αλληλεπίδραση απομακρυσμένης πρόσβασης

Η απομακρυσμένη πρόσβαση σάς επιτρέπει να κάνετε ταλάντωση, χειρονομία και αλληλεπίδραση με μια συσκευή μέσω του προγράμματος περιήγησης ιστού σας σε πραγματικό χρόνο. Υπάρχουν πολλές περιπτώσεις όπου η αλληλεπίδραση σε πραγματικό χρόνο με μια συσκευή είναι χρήσιμη. Για παράδειγμα, εκπρόσωποι εξυπηρέτησης πελατών μπορούν να καθοδηγήσουν τους πελάτες μέσω του τρόπου χρήσης ή ρύθμισης της συσκευής τους. Μπορούν επίσης να περπατήσουν τους πελάτες μέσω του τρόπου χρήσης των εφαρμογών που εκτελούνται σε μια συγκεκριμένη συσκευή. Μπορείτε να εγκαταστήσετε εφαρμογές σε μια συσκευή που εκτελείται σε μια περίοδο απομακρυσμένης πρόσβασης και στη συνέχεια να αναπαράγονται προβλήματα πελατών ή αναφερθέντα σφάλματα.

Κατά τη διάρκεια μιας περιόδου απομακρυσμένης πρόσβασης, το Device Farm συλλέγει λεπτομέρειες σχετικά με τις ενέργειες που πραγματοποιούνται καθώς αλληλεπιδράτε με τη συσκευή. Τα αρχεία καταγραφής με αυτές τις λεπτομέρειες και μια καταγραφή βίντεο της περιόδου σύνδεσης παράγονται στο τέλος της συνεδρίας για την αναθεώρησή σας.

Αρχικά, υποστηρίζεται περιορισμένος αριθμός συσκευών Android και Fire OS για απομακρυσμένη πρόσβαση. Ωστόσο, ο κατάλογος των συσκευών θα αυξηθεί κατά τη διάρκεια της περιόδου beta και καθώς οι νέες συσκευές εισέρχονται στην αγορά.

Ορολογία

Το Device Farm εισάγει τους ακόλουθους όρους που καθορίζουν τον τρόπο οργάνωσης των πληροφοριών:

project

Ένας λογικός χώρος εργασίας που περιέχει τρεξίματα, ένα τρέξιμο για κάθε δοκιμή μιας εφαρμογής σε σχέση με μία ή περισσότερες συσκευές. Τα έργα σας επιτρέπουν να οργανώνετε χώρους εργασίας με τον τρόπο που επιλέγετε. Για παράδειγμα, μπορεί να υπάρχει ένα έργο ανά τίτλο εφαρμογής ή μπορεί να υπάρχει ένα έργο ανά πλατφόρμα. Μπορείτε να δημιουργήσετε τόσα πολλά έργα που χρειάζεστε.

run

Μια συγκεκριμένη κατασκευή της εφαρμογής σας, με ένα συγκεκριμένο σύνολο δοκιμών, που θα εκτελεστεί σε ένα συγκεκριμένο σύνολο συσκευών. Μια εκτέλεση παράγει μια αναφορά που περιέχει πληροφορίες σχετικά με τα αποτελέσματα της εκτέλεσης. Μια εκτέλεση περιέχει μία ή περισσότερες εργασίες. Για περισσότερες πληροφορίες, ανατρέξτε στην ενότητα Εκτέλεση.

report

Περιέχει πληροφορίες σχετικά με μια εκτέλεση, η οποία αποτελεί ζήτημα για το Farm Farm να δοκιμάσει μια εφαρμογή έναντι μιας ή περισσότερων συσκευών. Για περισσότερες πληροφορίες, ανατρέξτε στην ενότητα Αναφορές.

job

Ένα αίτημα για το Farm Farm να δοκιμάσει μια ενιαία εφαρμογή σε μία μόνο συσκευή. Μια εργασία περιέχει μία ή περισσότερες σουίτες.

meter

Η μέτρηση αναφέρεται στη χρέωση για συσκευές και ενδέχεται να εμφανιστούν αναφορές σε "μετρημένες συσκευές" ή "μη μετρημένες συσκευές" στην αναφορά τεκμηρίωσης και API. Για περισσότερες πληροφορίες σχετικά με την τιμολόγηση, ανατρέξτε στο AWS Device Farm Pricing.

suite

Η ιεραρχική οργάνωση των δοκιμών σε μια δοκιμαστική συσκευασία. Μια σουίτα περιέχει μία ή περισσότερες δοκιμές.

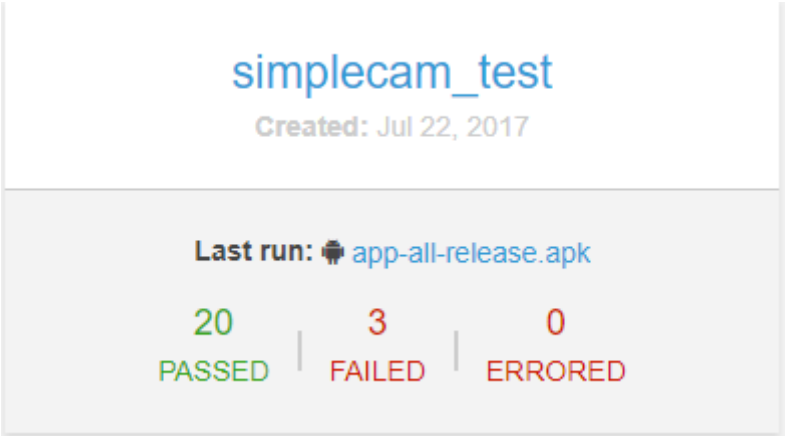
test

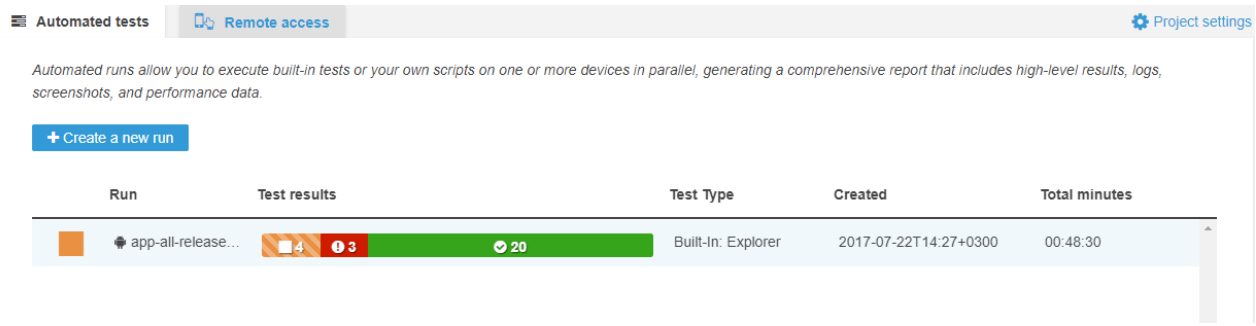
Μια μεμονωμένη δοκιμή μέσα σε μια δοκιμαστική συσκευασία.

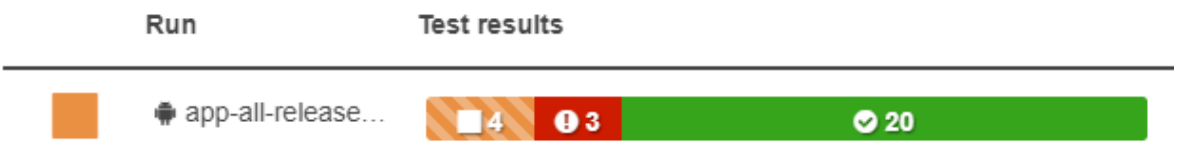
session

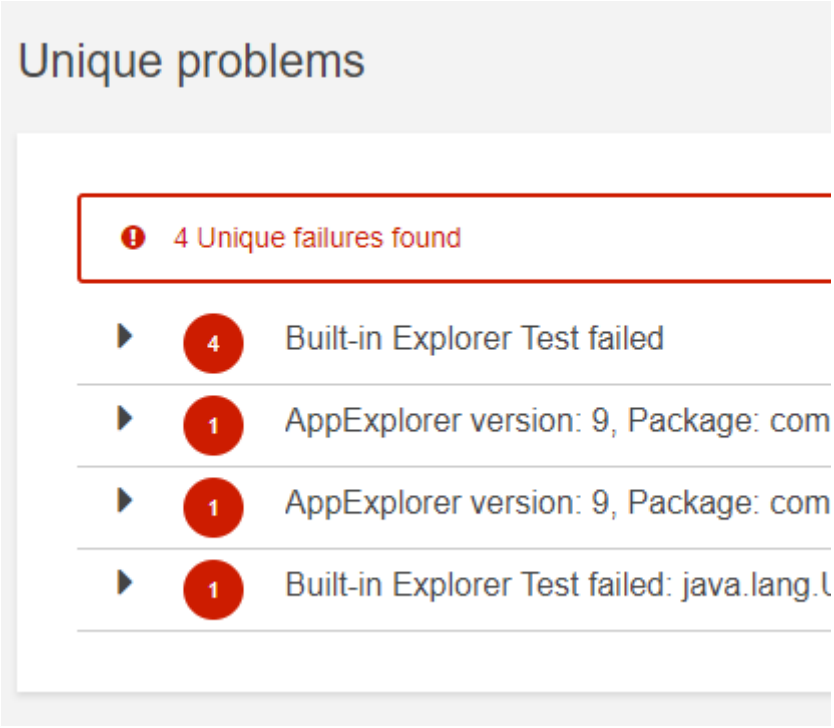
Μια διαδραστική περίοδος με μια μοναδική συσκευή στην κονσόλα.

Case Study , Simple cam android app Test.




Automated tests
Remote access
Project settings
Automated runs allow you to execute built-in tests or your own scripts on one or more devices in parallel, generating a comprehensive report that includes high-level results, logs, screenshots, and performance data.
+ Create a new run
Run
Test results
Test Type
Created
Total minutes
app-all-release...
4
3
20
Built-In: Explorer
2017-07-22T14:27+0300
00:48:30

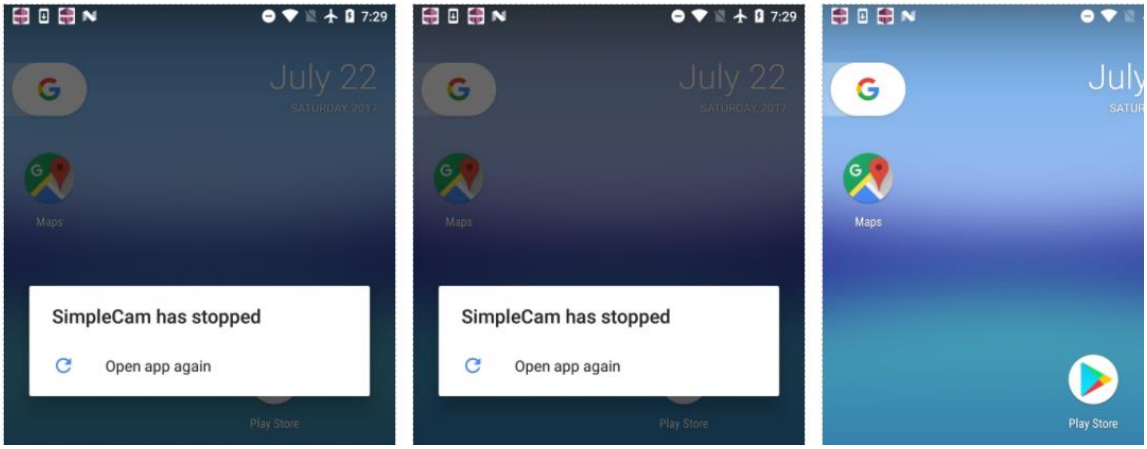

Run
Test results
app-all-release...
4
3
20


Unique problems
4 Unique failures found
4
Built-in Explorer Test failed
1
AppExplorer version: 9, Package: com
1
AppExplorer version: 9, Package: com
1
Built-in Explorer Test failed: java.lang.U

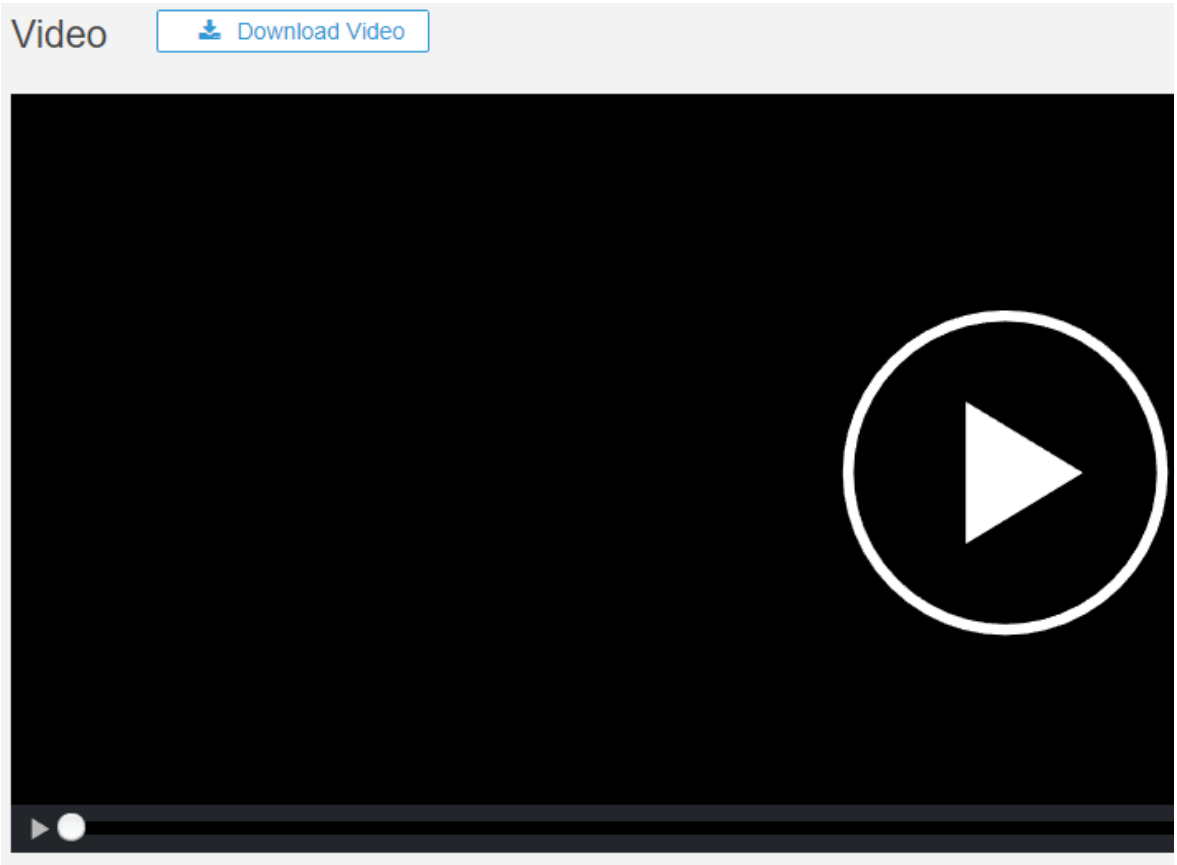
Devices

	Device		Test results	Total minut
	Amazon Fire (2015)	5.1	1 2	00:03:03
	Amazon Fire Phone	4.4.4	1 2	00:07:35
	Google Pixel	7.1	1 2	00:03:40
	HTC One A9 (Unlocked)	6.0.1	1 2	00:03:54
	HTC One M8 (Verizon)	4.4.4	1 2	00:08:06
	LG G4 (Verizon)	5.1	1 2	00:08:01
	LGE LG V20 (T-Mobile)	7.0	3	00:03:25
	Samsung Galaxy Note 3 (Sprint)	5.0	1 2	00:07:50
	Samsung Galaxy Note5 SM-N920C	6.0.1	3	00:02:51

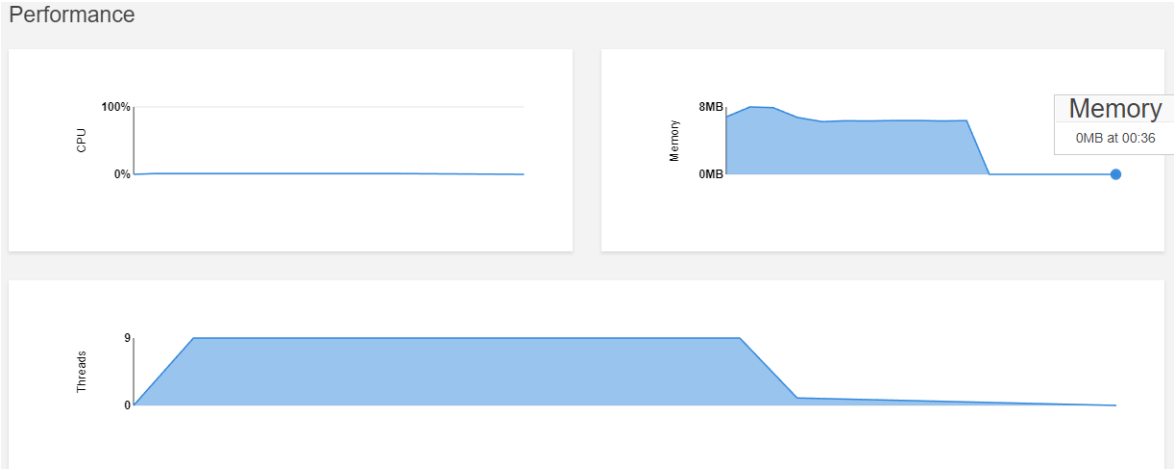
Google Pixel 7.1

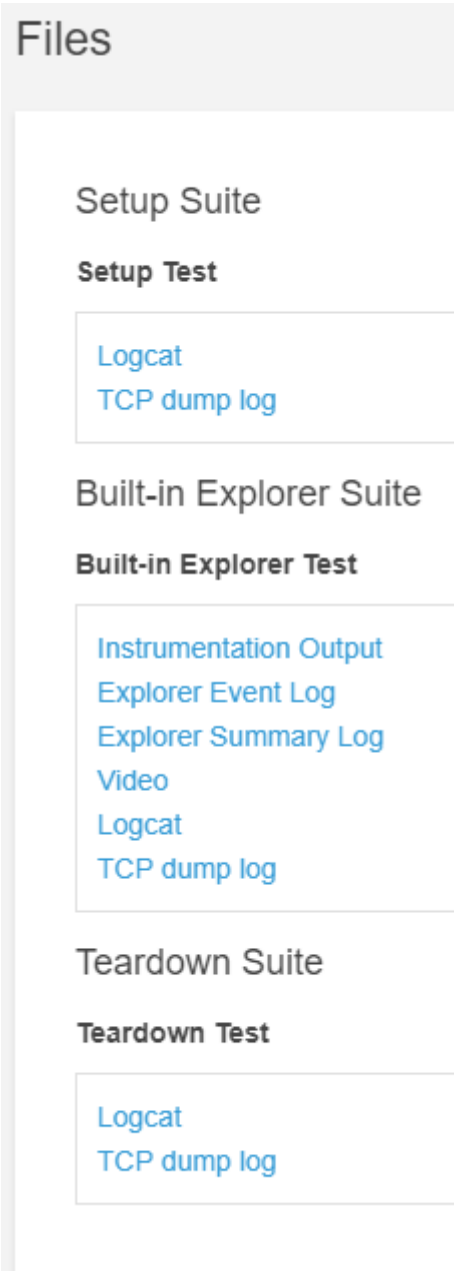


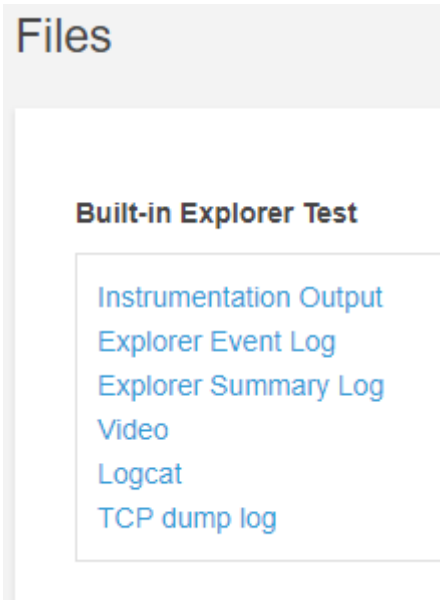
	HTC One A9 (Unlocked)	6.0.1	1	2



Suite	Test results
Setup Suite	1
Built-in Explorer Suite	1
Teardown Suite	1


Performance
CPU
100%
0%
Memory
8MB
0MB
Memory
0MB at 00:36
Threads
9
0


Files
Setup Suite
Setup Test
Logcat
TCP dump log
Built-in Explorer Suite
Built-in Explorer Test
Instrumentation Output
Explorer Event Log
Explorer Summary Log
Video
Logcat
TCP dump log
Teardown Suite
Teardown Test
Logcat
TCP dump log




```
2794  --------- beginning of crash
2795  07-22 04:29:44.405 17444 17444 E AndroidRuntime: FATAL EXCEPTION: main
2796  07-22 04:29:44.405 17444 17444 E AndroidRuntime: Process: com.protectsoft.simplecam, PID: 17444
2797  07-22 04:29:44.405 17444 17444 E AndroidRuntime: java.lang.RuntimeException: Unable to start activity
```

```
Caused by: java.lang.NullPointerException:

at com.protectsoft.simplecam.MainActivity.onCreate(MainActivity.java:125)
```

```
123             //get/set camera parameters
124             if (CameraFeatures.params == null) {
125                 CameraFeatures.params = camera.getParameters();
```

ΒΙΒΛΙΟΓΡΑΦΙΑ - ΑΝΑΦΟΡΕΣ

Γενικά για το Android:

http://www.tutorialspoint.com/android/android_overview.htm

Android Camera SourceCode:

https://github.com/avraampiperidis/AndroidCamera

Android NDK:

https://developer.android.com/tools/sdk/ndk/index.html

AndroidManifest.xml

http://developer.android.com/guide/topics/manifest/manifest-intro.html

Android Developer Tools:

http://developer.android.com/tools/help/adt.html

Android Studio:

http://developer.android.com/training/basics/firstapp/running-app.html

Android Runtime:

https://source.android.com/devices/tech/dalvik/index.html

http://www.techotopia.com/index.php/An_Overview_of_the_Android_Architecture

http://anandtech.com/show/8231/a-closer-look-at-android-runtime-art-in-android-l/

Application Components:

http://developer.android.com/guide/components/fundamentals.html

https://source.android.com/devices/tech/security/overview/app-security.html

Learning Android, O'Reilly (Marko Gargenta and Masumi Nakamura)

Διεπαφή Χρήστη:

http://developer.android.com/guide/topics/ui/overview.html

Γραφικά και Animations:

http://developer.android.com/guide/topics/graphics/index.html

Hardware Accelerated 2D Rendering for Android, Feb 19, 2013 / Android Builders Summit (Jim Huang)

Ασφάλεια:

http://source.android.com/devices/tech/security/overview/index.html

Δοκιμή:

http://www.vogella.com/tutorials/AndroidTesting/article.html

https://developer.android.com/tools/testing/testing_android.html

Google Services:
http://androidmobileapps.wikispaces.com/file/detail/Google+Play+Services.potx/529546152

https://developer.android.com/google/index.html

http://developer.android.com/google/play-services/index.html

https://developer.android.com/google/play-services/games.html

https://developer.android.com/google/play-services/location.html

https://developer.android.com/google/play-services/drive.html

https://developer.android.com/google/play/billing/index.html

https://developer.android.com/google/play-services/ads.html

http://developer.android.com/google/play-services/setup.html

http://developer.android.com/sdk/installing/adding-packages.html

[1] Open Handset Alliance (5 Νοεμβρίου 2007). *Industry Leaders Announce Open Platform for Mobile Devices*. Δελτίο τύπου. Ανακτήθηκε στις 5 Νοεμβρίου 2007.

[2] Shankland, Stephen (12 Νοεμβρίου 2007). «Google's Android parts ways with Java industry group». *CNET News*.

[3] «Open Handset Alliance». Open Handset Alliance. Ανακτήθηκε στις 6 Νοεμβρίου 2007.

[4] Jackson, Rob (10 Δεκεμβρίου 2008). «Sony Ericsson, HTC Androids Set For Summer 2009». *Android Phone Fans*. Ανακτήθηκε στις 3 Σεπτεμβρίου 2009.

[5] «Android Overview». Open Handset Alliance. Ανακτήθηκε στις 23 Αυγούστου 2008.

[6] Linus Torvalds says

[7] https://www.gnu.org/pronunciation/pronunciation.html

[8] IBM (October 2001). «Linux Watch». Ανακτήθηκε στις 2009-09-29.

[9] Linux Devices (January 2010). «Trolltech rolls "complete" Linux smartphone stack». Αρχειοθετήθηκε από το πρωτότυποστις 2012-05-25. Ανακτήθηκε στις 2009-09-29.

[10] Computerworld, Patrick Thibodeau. «IBM's newest mainframe is all Linux». Ανακτήθηκε στις 2009-02-22.

[11] Lyons, Daniel. «Linux rules supercomputers». Ανακτήθηκε στις 2007-02-22.

[12] «Operating system Family / Linux». Top500.org. Ανακτήθηκε στις 9 Σεπτεμβρίου 2014.

[13] «Linux Takes 1.7% Desktop Marketshare». 28 Αύγουστος 2014. Ανακτήθηκε στις 17 Μαιου 2015.

[14] The Economist (December 2008). «Small is beautiful». Ανακτήθηκε στις 2008-12-21.

[15] The Developer-network (January 2010). «Smartbook Playing Field Wide Open for Linux». Ανακτήθηκε στις 2008-12-21.

[16] Weeks, Alex (2004). «1.1». *Linux System Administrator's Guide* (version 0.9 έκδοση). Ανακτήθηκε στις 2007-01-18.

[17] «The GNU Operating System». Gnu.org. Ανακτήθηκε στις 2009-04-17.

[18] «Android Studio Plugin». *Android googlesource*. Ανακτήθηκε στις 25 Απριλίου 2015.

[19] «Download Studio». *Android Developers*. Android Developers.

[20] «Google releases Android Studio 1.0, the first stable version of its IDE». *venturebeat*. 8 Δεκεμβρίου 2014.

[21] «Download Android Studio IDE For Windows, OS X And Linux». *redmondpie*. 16 Μαΐου 2013.

[22] «Android M's name is Marshmallow». *The Verge*. Vox Media. Ανακτήθηκε στις August 17, 2015.

[23] «Get ready for the sweet taste of Android 6.0 Marshmallow». *Official Android Blog*. Google. Ανακτήθηκε στις 2015-10-06

[24] Chester, Brandon. «Google Announces Android M At Google I/O 2015». *Anandtech*. Purch, Inc.. Ανακτήθηκε στις May 28, 2015.

[25] Opam, Kwame (June 30, 2016). «Android N is now Android Nougat». The Verge. Ανακτήθηκε στις October 1, 2016.

[26] Amadeo, Ron (March 10, 2016). «Surprise! The Android N Developer Preview is out right now». *Ars Technica*. Condé Nast. Ανακτήθηκε στις July 1, 2016.

[27] «Android 7.0 Nougat: a more powerful OS, made for you». Android Developers Blog. August 22, 2016. Ανακτήθηκε στις August 23, 2016.