Τεχνολογικό Εκπαιδευτικό Ίδρυμα ΚΡΗΤΗΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

Τμήμα Μηχανικών Πληροφορικής

Πτυχιακή Εργασία

ΕΦΑΡΜΟΓΗ ANDROID ΓΙΑ ΚΑΘΟΔΗΓΗΣΗ ΧΡΗΣΤΗ ΣΕ ΠΕΡΙΠΤΩΣΗ ΕΚΤΑΚΤΗΣ ΑΝΑΓΚΗΣ

ΟΝΟΜΑΤΑ ΦΟΙΤΗΤΩΝ:

1. Μαρένα Χέκτορ ΑΜ: 3933
2. Κόλα Βιτόριο ΑΜ: 3953
3. Μπίμπα Φλορίν ΑΜ: 3621

ΕΠΙΒΛΕΠΟΝΤΕΣ

Παπαδάκης Νικόλαος

Περίληψη

Η πτυχιακή αυτή εργασία έχει στόχο να περιγράψει την κατασκευή μιας εφαρμογής πάνω σε λειτουργικό σύστημα Android η οποία θα έχει ως κύριο στόχο να ειδοποιήσει υπηρεσίες έκτακτης ανάγκης όπως Αστυνομία, Πυροσβεστική ή Ε.Κ.Α.Β., μέσω μηνύματος SMS ή ακόμα και να δείξει σε χάρτη στον συνδεδεμένο στην εφαρμογή χρήστη, το κοντινότερο σημείο που βρίσκεται η υπηρεσία που θέλει αυτός να καλέσει.

Θα χρησιμοποιηθούνε τεχνολογίες της Google όπως GoogleMaps για να δείχνουμε στον χρήστη τον χάρτη, GooglePlaces για την εύρεση της κάθε τοποθεσίας και GoogleDirections για την εύρεση της διαδρομής στον χάρτη. Τα δεδομένα του χρήστη θα διαχειρίζεται το Σύστημα Διαχείρισης Σχεσιακών Βάσεων Δεδομένων MySQL. Η εφαρμογή Android έχει όνομα **Emergency App** και η αρχιτεκτονική της ακολουθεί το μοτίβο **MVP** (**M**odel**V**iew**P**resenter).

Θεματική Περιοχή: Android εφαρμογή

Summary

This thesis is intended to describe the creation of an application installed on an Android operating system, that has as a main goal to send an emergency call to services like Police, Fire Station and Hospitals, through SMS or even showing to the end user a map with the direction for the shortest location of the service the user called.

We will use Google technologies like Google Maps for showing the map to the user, Google Places for finding every location we want and Google Directions for finding the direction on the map. User data will be managed by the Relational Database Management System MySQL. The Android application's name is **Emergency App** and its architecture follows the **MVP** Pattern (**M**odel**V**iew**P**resenter).

Subject Area: Android application

ΠΕΡΙΕΧΟΜΕΝΑ

Εισαγωγή

Στις μέρες μας οι κινητές συσκευές αποτελούν αναπόσπαστο κομμάτι της καθημερινότητας μας καθώς χωρίς αυτές δεν θα μπορούσαμε να επικοινωνήσουμε με τα συγγενικά μας πρόσωπα ή στην περίπτωση της εργασίας αυτής να καλέσουμε για βοήθεια. Η αρχική ιδέα για την δημιουργία της συγκεκριμένης

εφαρμογής γεννήθηκε από το γεγονός ότι δεν υπάρχει αρκετός χρόνος να καλέσεις για βοήθεια και σε πολλές περιπτώσεις δεν μπορείς να καλέσεις όπως για παράδειγμα σε περίπτωση ληστείας.

Η εφαρμογή

Η εφαρμογή που πραγματεύεται η εργασία αυτή επιτρέπει μέσω κινητής συσκευής Android επικοινωνία με υπηρεσίες έκτακτης ανάγκης σε συνδεδεμένο με την εφαρμογή, χρήστη, με το πάτημα ενός κουμπιού. Σε περίπτωση που ο χρήστης δεν έχει λογαριασμό στην εφαρμογή, μπορεί να κάνει εγγραφή μέσω αυτής. Η εφαρμογή έχει τη δυνατότητα να λάβει το γεωγραφικό σημείο στο οποίο βρίσκεται ο χρήστης, τα οποία και θα στείλει στην περίπτωση που ο τελευταίος επιλέξει να στείλει ειδοποίηση στην αντίστοιχη υπηρεσία έκτακτης ανάγκης. Δίνεται επίσης και η επιλογή να δει σε χάρτη το κοντινότερο σημείο που αναζητά ο χρήστης μέσω της υπηρεσίας GoogleMaps. Οι υπηρεσίες έκτακτης ανάγκης για τις οποίες έχει υλοποιηθεί η εφαρμογή είναι οι παρακάτω:

- Αστυνομία
- Πυροσβεστική
- Ε.Κ.Α.Β.

Παρακάτω θα δούμε διάφορα κομμάτια της εφαρμογής ως σενάρια επικοινωνίας μελετώντας τα.

Μελέτη σεναρίων επικοινωνίας

Οι υπηρεσίες έκτακτης ανάγκης

Οι υπηρεσίες αυτές επιλέχθηκαν διότι καλύπτουν ένα ευρύ φάσμα περιπτώσεων έκτακτης ανάγκης, ταυτόχρονα με το γεγονός ότι είναι αυτές που διαθέτουν υπηρεσία αποστολής SMS (**S**hort**M**essage**S**ervice). Η αστυνομία συγκεκριμένα δίνει τη δυνατότητα στους πολίτες να επικοινωνήσουν με τα τμήματα με ηλεκτρονικό μήνυμα στη διεύθυνση amdrasi@astynomia.gr αλλά χάριν ενιαίας μεθόδου επικοινωνίας με τις υπηρεσίες, χρησιμοποιήσαμε μόνο την υπηρεσία SMS.Ο πίνακας των αριθμών για τις αντίστοιχες υπηρεσίες δίνεται στον πίνακα παρακάτω:

Υπηρεσία	Αριθμός
Αστυνομία	100
Πυροσβεστική	199
Ε.Κ.Α.Β.	166

Σύμφωνα με τις υπηρεσίες αυτές, η αποστολή μηνύματος είναι ανέξοδη. Όταν παραληφθεί το μήνυμα από τις υπηρεσίες αυτές, εμφανίζεται σε υπολογιστή το περιεχόμενο και ειδοποιούνται αναλόγως. Σε αστυνομικό τμήμα ανοίγεται αμέσως φάκελος συμβάντος και δίνεται εντολή στο πλησιέστερο περιπολικό να μεταβεί στο σημείο που έχει αναφερθεί για να ελέγξει τι έχει συμβεί ή, αν έχει συμβεί στην επαρχία, ενημερώνονται άμεσα οι κατά τόπους αρμόδιες αρχές. Το περιεχόμενο του μηνύματος εμφανίζεται σε οθόνη υπολογιστή στο κέντρο επιχειρήσεων της Άμεσης Δράσης. Σε περίπτωση που ο χειριστής του υπολογιστή χρειάζεται κάποια διευκρίνιση στέλνει στον πολίτη που έστειλε το SMS γραπτό μήνυμα στο κινητό τηλέφωνό του ζητώντας περισσότερες πληροφορίες.

Σε γενικές γραμμές έτσι δουλεύει και το αντίστοιχο σύστημα της πυροσβεστικής αλλά και της Άμεσης Βοήθειας. Τα συστήματα αυτά είναι διαθέσιμα στην Ελλάδα τα τελευταία 13 χρόνια, λίγο μετά τους Ολυμπιακούς Αγώνες καθώς και γίναμε έτσι η τρίτη χώρα στην Ευρώπη που διαθέτουμε τέτοιο σύστημα μαζί με την Γαλλία και την Φιλανδία. Το σύστημα αυτό μπορεί να εξυπηρετήσει και άτομα με ειδικές ανάγκες όπως βαρήκοους και κωφούς.

Σύνδεση χρήστη

Για να χρησιμοποιήσει κάποιος την εφαρμογή αυτή θα πρέπει να είναι συνδεμένος με τον λογαριασμό που θα έχει δημιουργήσει πιο πριν με τα αληθινά στοιχεία του βοηθώντας έτσι την κάθε υπηρεσία να γνωρίζει τα στοιχεία του ατόμου που ζητάει βοήθεια. Η εφαρμογη ζητάει από τον χρήστη τα παρακάτω στοιχεία.

- Όνομα
- Επίθετο
- Χώρα
- Πόλη
- Διεύθυνση
- Τηλέφωνο
- Ημερομηνία Γέννησης
- Όνομα χρήστη
- Κωδικός πρόσβασης

Με αυτά τα στοιχεία μπορούμε ακόμα και σε περίπτωση που δεν βρεθεί ο χρήστης στο σημείο που δήλωσε ότι βρίσκεται, να βρούμε περεταίρω στοιχεία για εκείνον αφού γίνεται καταγραφή από την εφαρμογή της κάθε κλήσης με ακριβή ώρα και ημερομηνία αυτής.

Η ειδοποίηση της υπηρεσίας

Το περιβάλλον χρήσης θα δίνει απλές και κατανοητές επιλογές στον χρήστη ο οποίος με το πάτημα ενός κουμπιού έχει τη δυνατότητα να ειδοποιήσει την υπηρεσία. Συγκεκριμένα η κάθε κλήση θα σημαίνει αποστολή μηνύματος SMS στην αντίστοιχη υπηρεσία το οποίο έχουμε επιλέξει να περιέχει τα παρακάτω στοιχεία:

- Όνομα χρήστη
- Επώνυμο χρήστη
- Γεωγραφικό πλάτος
- Γεωγραφικό μήκος

Σημειώνεται ότι δεν υπάρχει χρέωση για την αποστολή μηνύματος. Η γεωγραφική θέση του χρήστη είναι απαραίτητη για την αντίστοιχη υπηρεσία ώστε να βρει άμεσα την θέση αυτού.

Διαδρομή για την υπηρεσία

Παράλληλα με την ειδοποίηση της υπηρεσίας από τον χρήστη μέσω SMS, δίνεται η δυνατότητα σε αυτόν να δει την διαδρομή από το σημείο που βρίσκεται στο κοντινότερο σημείο που βρίσκεται η αντίστοιχη υπηρεσία. Εδώ θα χρησιμοποιήσουμε κάποια εργαλεία που μας δίνει η Google. Συγκεκριμένα θα χρησιμοποιήσουμε τα εξής εργαλεία:

- Google Maps API
- Google Places API
- Google Directions API

Αυτά τα εργαλεία παρέχονται επίσης ελεύθερα από την Google που σημαίνει ότι δεν υπάρχει χρέωση αλλά θα πρέπει ως Developers σε λογαριασμό Google να δηλώσουμε ότι θα χρησιμοποιήσουμε τα αντίστοιχα εργαλεία μέσω της σελίδας https://console.developers.google.com.

Για την αποθήκευση των δεδομένων χρήστη θα χρησιμοποιήσουμε το Σύστημα Διαχείρισης Σχεσιακών Βάσεων Δεδομένων MySQL. Πρόκειται για ένα σύστημα ανοιχτού κώδικα υπό την άδεια GNUGPL (**G**eneral**P**ublic**L**icense), που επιβεβαιώνει ότι μπορεί οποιοσδήποτε προγραμματιστής να διαβάσει, εκτελέσει, μοιράσει και διαμορφώσει τον κώδικα μιας εφαρμογής.

Λίγα λόγια για την GNUGPL

Αρχικά γράφτηκε από τον RichardStallman για το GNUProject και συντηρείται από την Oracle. Το GNUProject ένα project που ξεκίνησε τον Σεπτέμβριο του 1983 στο MIT και στοχεύει στο αποτέλεσμα να δώσει στους χρήστες υπολογιστών την ελευθερία και τον έλεγχο πάνω στις υπολογιστικές τους μηχανές. Όταν δημιουργείται παράγωγο λογισμικό υπό αυτή την άδεια, μπορεί να διανεμηθεί μόνο υπό την ίδια άδεια. Η τελευταία έκδοση αυτής είναι η GNU GPLv3.

PHP

‘’ Η **PHP** (Hypertext Preprocessor) είναι μια γλώσσα προγραμματισμού ανοιχτού κώδικα, για τη δημιουργία σελίδων με δυναμικό περιεχόμενο.Η πιο συχνή χρήση της γλώσσας αυτής είναι η ανάκτηση δεδομένων από βάσεις δεδομένων SQL, η επεξεργασία τους και η εμφάνιση των αποτελεσμάτων στους χρήστες. Συγκρίνοντας την PHP με την HTML θα λέγαμε ότι η βασική τους διαφορά είναι ότι η PHP είναι δυναμική γλώσσα προγραμματισμού που σημαίνει ότι οι χρήστες μπορούν να δημιουργήσουν περιεχόμενο σε μια ιστοσελίδα. Χρησιμεύει επίσης για την δημιουργία δικτυακής υπηρεσίας (Web Service), ενός τομέα, δηλαδή, που είναι ικανός να εκτελέσει λειτουργίες πάνω στον διακομιστή και να μεταβιβάσει το αποτέλεσμα στον χρήστη. Το ότι είναι μια server-side γλώσσα μας δημιουργεί την επιθυμία να την χρησιμοποιήσουμε για την υλοποιήση της εφαρμογής και την ομαλή επικοινωνία με την βάση δεδομένων. Σήμερα σχεδόν όλοι οι web servers υποστηρίζουν την εκτέλεση κώδικα PHP.’’ www.wikipedia.com

ΙΣΤΟΡΙΚΑ ΓΕΓΟΝΟΤΑ

Γράφτηκε το 1994 , εφευρέθηκε από τον Rasmus Lerdorf και δημοσιεύτηκε το 1995.Ξεκίνησε ως ένα σύνολο εργαλείων για τη δημιουργία και επεξεργασία ιστοσελίδων. Η γλώσσα προγραμματισμού με την οποία γράφτηκε ήταν η C.

Εκδόσεις PHP

June 8, 1995 -> PHP 2.0
1997 -> PHP/FI 2 (update του PHP 2.0)
1998 -> PHP 3
2000 -> PHP 4
2004 -> PHP 5

Εισαγωγή Στο Λειτουργικό Σύστημα Android

Στο κεφάλαιο αυτό θα κάνουμε μια εισαγωγή στα smartphones και στην συνέχεια θα περιγράψουμε την πλατφόρμα Android καθώς η εφαρμογή που θα περιγράψουμε αφορά κατά κύριο λόγο λειτουργικά συστήματα Android.

SMARTPHONES

Η κινητή τηλεφωνία όταν δημιουργήθηκε είχε διαφορετική έννοια και σημασία απ' ότι σήμερα. Τα κινητά τηλέφωνα τότε δημιουργήθηκαν αποκλειστικά και μόνο για την τηλεφωνική επικοινωνία δύο ατόμων . Στην συνέχεια προστέθηκαν και τα μυνηματα. Σήμερα η έννοια του τηλεφώνου έχει αλλάξει αποκτώντας τρομερές δυνατότητες.
Με τον όρο smartphone θα μπορούσαμε να χαρακτηρίσουμε τον συνδυασμό ενός κινητού τηλεφώνου όπως το ξέρουμε με τον προσωπικό μας υπολογιστή μέσω ενός προχωρημένου λειτουργικού συστήματος που συνδυάζει αυτά τα δυο. Συνήθως είναι σε μέγεθος τσέπης για την διευκόλυνση της μεταφοράς και του χειρισμού με ένα χέρι. Όλα τα smartphones σήμερα περιέχουν γραφικό περιβάλλον με έγχρωμη οθόνη. Τον Οκτώμβριο του 2008 δημιουργήθηκε το πρώτο smartphone με android έκδοση. Περιείχε ιδιόμορφα σχεδιαστικά στοιχεία όπως το swing-out πληκτρολόγιο. Το android είναι ένα open-source λειτουργικό σύστημα που δημιουργήθηκε από τον Andy Rubin και τώρα ανήνει την Google.

Android

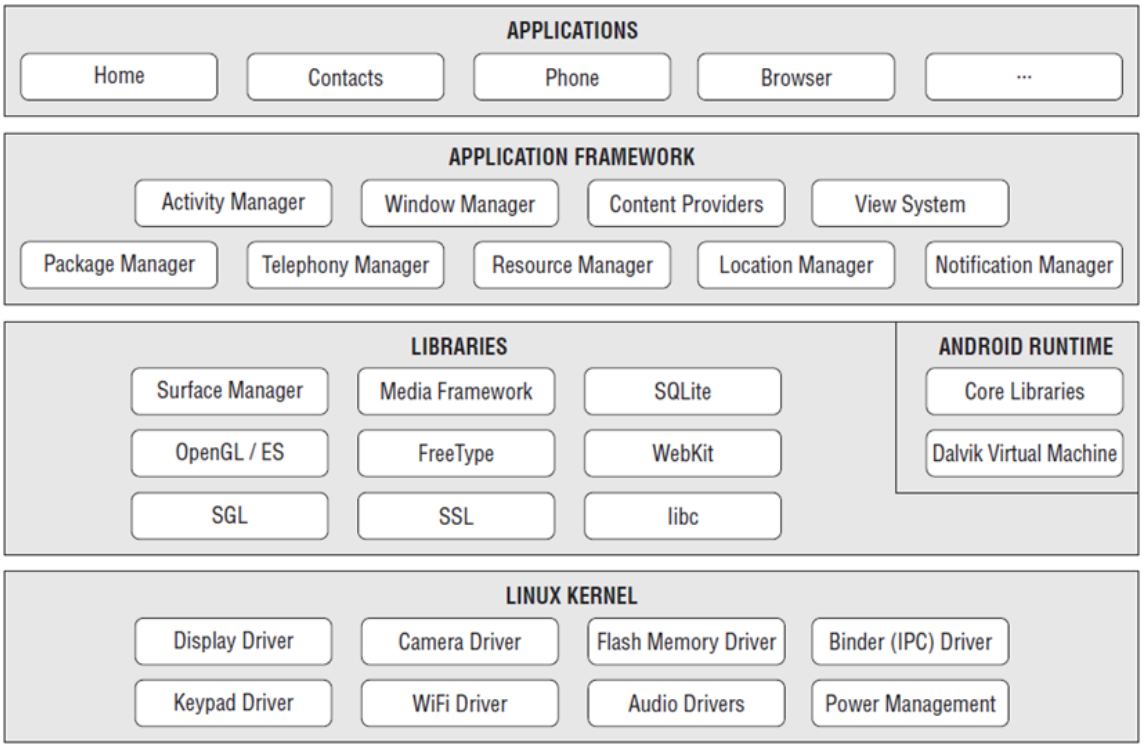
Τι είναι το Android OS

Η λέξη "Android" αναφέρεται σε ένα λειτουργικό σύστημα για κινητά τηλέφωνα και όχι μόνο (βασισμένο στο Linux) το οποίο αναπτύσσεται από την Google. Είναι λογισμικό ανοιχτού κώδικα, το οποίο σημαίνει οτι οποιοσδήποτε μπορεί να πάρει τον πηγαίο κώδικα και να τον χρησιμοποιήσει / παραμετροποιήσει . Αφού λοιπόν είναι ελεύθερα διαθέσιμο, κατασκευαστές hardware όπως η HTC, η Motorola και η Samsung παίρνουν τον κώδικά του και το χρησιμοποιούν ως βάση για να χτίσουν πάνω του τις δικές τους ιδιόκτητες υλοποιήσεις που τρέχουν στα κινητά τους τηλέφωνα. Εκτός από τα smartphone εξυπηρετεί και μία τεράστια γκάμα από tablet όπως το Nexus 7. Η ποικιλομορφία του όμως δεν σταματά εδώ, αφού μπορεί να τρέξει σ' όλων των ειδών τις συσκευές που υποστηρίζονται στο Linux από κάμερες, κονσόλες παιχνιδιών και τηλεοράσεις μέχρι συστήματα αυτοκινήτων, ρολόγια και γυαλιά.Το Android όμως δεν είναι ίδιο σε όλες τις συσκευές. Ο μεγάλος ανταγωνισμός κάνει τις εταιρίες να το παραμετοποιούν με δικές τους ιδέες προσπαθώντας να προσελκύσουν περισσότερους καταναλωτές.

Πότε παρουσιάστηκε

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

Αρχιτεκτονική συστήματος Android



Πυρήνας Linux

Στο κατώτερο επίπεδο, το οποίο επικοινωνεί με το υλικό, βρίσκεται μία τροποποιημένη έκδοση του πυρήνα Linux. Διαθέτει μία μεγάλη ποικιλία προγραμμάτων οδήγησης.

Από τον Απρίλιο του 2014, οι συσκευές Android χρησιμοποιούν κυρίως τις εκδόσεις 3.4 ή 3.10 του πυρήνα Linux. Ο πυρήνας παρέχει βασικές λειτουργίες του συστήματος, όπως είναι η διαχείριση της δικτύωσης, των διεργασιών, της μνήμης και συσκευών όπως η κάμερα, το πληκτρολόγιο και η οθόνη.

Βιβλιοθήκες

Πάνω από τον πυρήνα υπάρχει μια σειρά βιβλιοθηκών γραμμένες σε εγγενή κώδικα C ή C++, στις οποίες περιλαμβάνεται η μηχανή προγράμματος περιήγησης WebWebKit ,η γνωστή βιβλιοθήκη libc, η βάση δεδομένων SQLite, η οποία είναι χρήσιμη για την αποθήκευση δεδομένων και την κοινή χρήση τους με άλλες εφαρμογές, βιβλιοθήκες για την αναπαραγωγή ήχου και βίντεο, βιβλιοθήκες SSL υπεύθυνες για την ασφαλή μετάδοση δεδομένων στο Internet κλπ.

Το περιβάλλον εκτέλεσης του Android

Κάθε εφαρμογή που τρέχει σε ένα λειτουργικό android το κάνει μέσα από δικιά του διεργασία της εικονικής μηχανής Dalvik Virtual Machine. Η εικονική μηχανή Dalvik αναπτύχθηκε από την Google και βασίζεται στον πυρήνα linux. Μέχρι την έκδοση 4.4 του Android χρησιμοποιούνταν η Dalvik Virtual Machine, ενώ από την 5ηέκδοση και έπειτα η Dalvik Virtual Machine έχει αντικατασταθεί από την εικονική μηχανή Android Runtime(ART). 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 . 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 JavaStandardEdition

Το framework για τις εφαρμογές

Το framework παρέχει πολλες υπηρεσίες οι οποίες ειναι με την μορφή βιβλιοθηκών Java που εχουν δημιουργηθεί και σχεδιαστεί ειδικά για το λειτουργικό σύστημα του Android.

Εφαρμογές

Οι εφαρμογές ειναι προγράμματα τα οποία λειτουργούν με την χρήση των frameworks και βρίσκονται στο ανώτερο επίπεδο. Μια εφαρμογή μπορει να είναι σε οποιοδήποτε μορφή και για οποιαδήποτε χρήση δηλαδή μπορεί να έχουμε απο εφαρμογές υπολογιστικές μέχρι εφαρμογές διασκεδαστικού περιεχομένου οπως για παράδειγμα τα παιχνίδια .

Χαρακτηριστικά Android

‘’Η πλατφόρμα είναι προσαρμόσιμη σε πολλές ανάλυσεις οθόνης (από VGA μέχρι 4K), δισδιάστατες ψηφιακές γραφικές βιβλιοθήκες, τρισδιάστατα γραφικά βασισμένα στην OpenGLs 3.0+ έκδοση χαρακτηριστικών, καθώς και παραδοσιακές απεικονίσεις οθόνης "έξυπνων" συσκευών κινητής τηλεφωνίας. Μπορεί να συνεργαστεί με κάμερες στατικής ή κινούμενης εικόνας, οθόνες αφής, GPS,

αισθητήρες επιτάχυνσης, μαγνητόμετρα, δισδιάστατους καθώς και τρισδιάστατους επιταχυντές γραφικών.Για την περιήγηση στον ιστό το Android διαθέτει φυλλομετρητή βασισμένο στην ανοιχτή τεχνολογία WebKit. 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). Από το Android 2.2 και ύστερα οι multi-touch displays έγιναν νόρμα.'', https://el.wikipedia.org/wiki/Android

Μνήμη & Αποθήκευση

Στο κινητό μας τηλέφωνο η στο τάμπλετ μας έχουμε 2 μέρη που ειναι χωρισμένη ο χώρος αποθήκευσης μας. Στο πρώτο μέρος εχουμε να κάνουμε με την λειτουργεία των εφαρμογών μας και στο δεύτερο μέρος με την αποθήκευση των δεδομένων και προγραμμάτων στην συσκευή μας . Το AndroidOS έχει σχεδιαστεί έτσι ώστες οταν ένα πρόγραμαμα ειναι σε λειτουργεία να καταναλώνει οσο πιο λίγο RAM μπορεί έτσι ώστες να έχουμε περισσότερη οικονομία στην μπαταρία και στον αποθηκευτικό μας χώρο.

Accelerometer

Στις περισσότερς συσκευές Android υπάρχουν ενσωματεμένοι αισθητήρες που με τρούν την κίνηση.Το επιταχυνσιόμετρο (accelerometer) είναι ένας απο αυτούς, είναι δηλαδή μια συσκευή που έχει την ικανότητα να μετρά δυνάμεις επιτάχυνσης. Η μέτρηση των δυνάμεων που δέχεται η συσκευή μπορεί να χρησιμοποιηθεί για να αναγνωρίσει ο χρήστης προς ποια κατεύθυνση μετακινεί τη συσκευή. Ετσι το επιταχυνσιόμετρο θα είναι αρκετά χρήσιμο στον χρήστη στις εφαρμογές που θα χρειάζονται δεδομένα κίνησης και τοποθεσίας οπως για παράδειγμα στα παιχνίδια. Η τιμή της επιτάχυνσης εκφράζεται σαν ένα διάνυσμα 3 διαστάσεων το οποίο αναπαριστά τις τιμές των επιταχύνσεων στους άξονες X, Y και Z.''

GPS

Το Global Positioning System(GPS) ή Παγκόσμιο Συστήματος Εντοπισμού Θέσης ειναι ένα συστημα το οποίο εξυπηρετει στην πλοήγηση του χρήστη και στον εντοπισμό της γεωγραφικής του θέσης. Το σύστημα αυτο χρησιμοποιείται μεταξύ άλλων και στις συσκευές Android. Έτσι μια συσκευή Andoid έχει πλήρη επίγνωση εύκολα και γρήγορα της θέσης του στο χώρο. Το σύστημα GPS αποτελείται απο τουλάχιστον 24 δορυφόρους οι οποίοι λαμβάνουν σήμα απο της συσκευές που χρησιμοποιούν το GPS και με διάφορους αλγορίθμους και μαθηματικές πράξεις βρίσκουν την ακριβή γεωγραφική θέση των συσκευών. Το κινητό τηλέφωνο έχει τη καλύτερη δυνατή απόδοση όταν ο χρήστης βρίσκεται σε ανοιχτό χώρο διότι οι πιθανότητες στο να γίνουν παρεμβολές είναι αρκετά μειωμένες απο το να βρίσκεται ο χρήστης σε κλειστό χώρο , έτσι ειναι ευκολότερο για τους δορυφόρους να εντοπίσουν την ακριβή γεωγραφική θέση. Επίσης μπορεί να εντοπίσει το κοντινότερο σήμα WI-FI καθώς και τη κοντινότερη κεραία κινητής τηλεφωνίας. Επίσης μπορεί να χρησιμοποήσεις τα δεδομένα και τις συντεταγμένες απο τους δορυφόρους του GPS για την γεωγραφική τοποθεσία της συσκευής σε διάφορες εφαρμογές.

Ενημέρωση Λογισμικού

Κατα περιόδους, κυκλοφορούν πιο βελτιωμένες εκδόσης του λογισμικού android, αυτο κυρίως γίνεται για να βελτιωθούν και οι επιδόσεις του τηλεφώνου με σκοπό πάντα την μέγιστη αξιοποίησης του υλισμικού και να προστεθούν νέες υπηρεσίες που ενδεχομένως να έχουν ζητηθεί απο τον ίδιο τον χρήστη. Οταν κυκλοφορεί μια νέα έκδοση λειτουργικού της android ο χρήστης ενημερώνεται απο την συσκευή και παρατρύνεται να προχωρησει στην αναβάθμηση του λειτουργικού του συστήματος. Έτσι ο χρήστης αποκτάει την νέα έκδοση και εκμεταλέυεται τις καινούργιες υπηρεσίες της.

Επικοινωνία με το Android OS

Με τα κινητά τηλέφωνα η εποικωνία έχει γίνει πολύ ευκολότερη σε σχέση με τα προηγούμενα χρόνια διότι οι άνθρωποι πλέον έχουν την δυνατότητα να επικοινωνούν μεταξύ τους και οταν αυτοι βρίσκονται σε μεγάλες αποστάσεις . Αυτο όμως δεν εμπόδισε την ανθρωπότητα να προσπαθήσει να βελτιώσει τον τρόπο που γίνοταν οι επικοινωνία μέσω τον κινητων τηλεφώνων. Πλέον με το λειτουργικό Android εγινε πολύ πιο ευκολή και διασκεδαστική η επικοινωνία για το με τα άτομα που τον ενδιαφέρουν. Έτσι ο χρήστης δεν έχει μόνο τον γραπτό ή τον φωνιτικό τρόπο επικοινωνίας μέσο του κινητού τού αλλα μπορεί να στέλνει φωτογραφίες ή να τις ανεβάζει στο διαδύκτιο, να κάνει βιντεοκλήσεις και να ψάχνει πληροφορίες μέσω του ίντερνετ με μόνο λίγα πατήματα στην οθόνη αφης του. Έχοντας το λειτουργικό Android εισχωρίσει στις κινητές συσκευές οι δυνατότητες και οι ιδέες για πως να γίνει η καθημερινότητα μας πιο ευκολη έχουν πληθύνει σε σημαντικό βαθμό. Υπάρχει πλέον οι δυνατότητα της διαχείρισης λογαριασμών email και λογαριασμών άλλων υπηρεσιών μέσω του κινητού μας τηλεφώνου απο οπουδήποτε χωρίς να χρειάζεται πρόσβαση σε ηλεκτρονικό υπολογιστή .

Ψυχαγωγία στο Android OS

Ο χρήστης με την χρήση του Android δεν έχει μονο την δυνατότητα απλά να απαθανατίσει μια ιδιαίτερη στιγμή για εκείνον , αλλα μπορεί πλέον με την χρήση του κινητού του τηλεφώνου να την μοιραστεί μέσω email ή αλλων υπηρεσιών .Επίσης έχει την δυνατότητα να την ανεβάσει ακομα και σε κοινωνικά δίκτυα χωρίς ωστόσο να χρειάζεται ιδιαίτερες γνώσεις . Μέσω του κινητού του μπορεί να δημιουργήσει ένα άλμπουμ φωτογραφιών της αρεσκείας του και να έχει εκει συγκεντρωμένες όλες τις φωτογραφίες που ο ίδιος θέλει. Με το Android πλέον και με την χρήση διάφορων εφαρμογών υπάρχει η δυνατότητα τροποποίησης μια φωτογραφίας με διάφορους και πολλούς τρόπους(χρώματα,κείμενα,μοντάζ) . Ιδιαίτερα μεγάλο κομμάτι του θέματος της ψυχαγωγίας είναι και η δυνατότητα που έχει πλέον ο χρήστης να ακούσει μέσω του κινητού του την μουσική που ο ίδιος επιθυμεί ή ακομα και να δεί τα βίντεοκλιπ που ο ίδιος θέλει απο όπου θέλει.

Ελάχιστα Χαρακτηριστικά Υλικού για Android OS

Το πιο σύνηθες πρόβλημα που αντιμετωπίζουν οι προγραμματιστές εφαρμογών android, ειναι οι διαφορές που έχουν οι συσκευές κινητών τηλεφώνων μεταξύ τους . Ένα πρόβλημα που πολλές φορές μπορει να αντιμετωπίσει ένας προγραμαμτιστής ειναι οι διαφορές στις οθόνες των συσκευών, ιδιαίτερα τις διαφορές στο μέγεθος για παράδειγμα μια εφαρμογή να εμφανίζεται καλύτερα σε συσκευή με μεγαλύτερη οθόνη ενω να μην εμφανίζεται το ίδιο σε συσκευές οθόνες μίκροτερου μεγέθους. Επίσης μεγάλες ειναι οι διαφορές στην υλισμική δομή που μπορεί να έχουν οι συσκευές μεταξύ τους, σε κάποιες συσκευές μπορει ο επεξεργαστης να είναι πιο ισχυρός ενώ σε άλλες πιο αδύναμος και αργός, μια συσκευεί μπορει να εχει επιταχυνσιόμετρο ενώ κάποι άλλη όχι. Έτσι για να μην υπάρχουν όλα αυτα τα προβλήματα και για να γίνεται οι δουλειά του προγραμματιστή πιο ευκολη

και να υπάρχει πιο γρήγορη ανάπτυξη στον τομέα θεσπίστηκάν οι ελάχιστες αυτες προδιαγραφές, για να υπάρχει μια ομοιομορφία και να μην χρειάζεται για παράδειγμα μια εφαρμογή να βγει σε πέντε ή και έξι ακόμα εκδόσεις.

Εδώ φένονται οι ελάχιστες προδιαγραφές που πρέπει να έχει μια συσκευή Android για να τρέξει ομαλά μια εφαρμογή σε μια συσκευή Android .

- **Chipset:**Βασισμένο σε ARM
- **Μνήμη:** 128MBRAM / 256MB External Flash. Το σύστημα Android μπορεί να εκκινήσει σε λειτουργία συντήρησης με λιγότερη μνήμη αλλά είναι κάτι που δεν προτείνεται.
- **Χώρος αποθήκευσης:**Mini / MicroSD. Δεν είναι απαραίτητο για την λειτουργία του συστήματος αλλά προτείνεται.
- **Κύρια οθόνη:**QVGA TFT LCD με 16-bit ανάλυση χρωμάτων. Το γραφικό περιβάλλον του Androidστοχεύει σε συσκευές με οθόνη όχι μικρότερη από 2.8 ίντσες.
- **Πλήκτρα πλοήγησης:**Τρία πλήκτρα που θα αφορούν την ενεργοποίηση της συσκευής και την αυξομοίωση του ήχου.
- **USB:**Mini-B USB
- **Bluetooth:**Εκδοση 1.2

Εκδόσεις Android

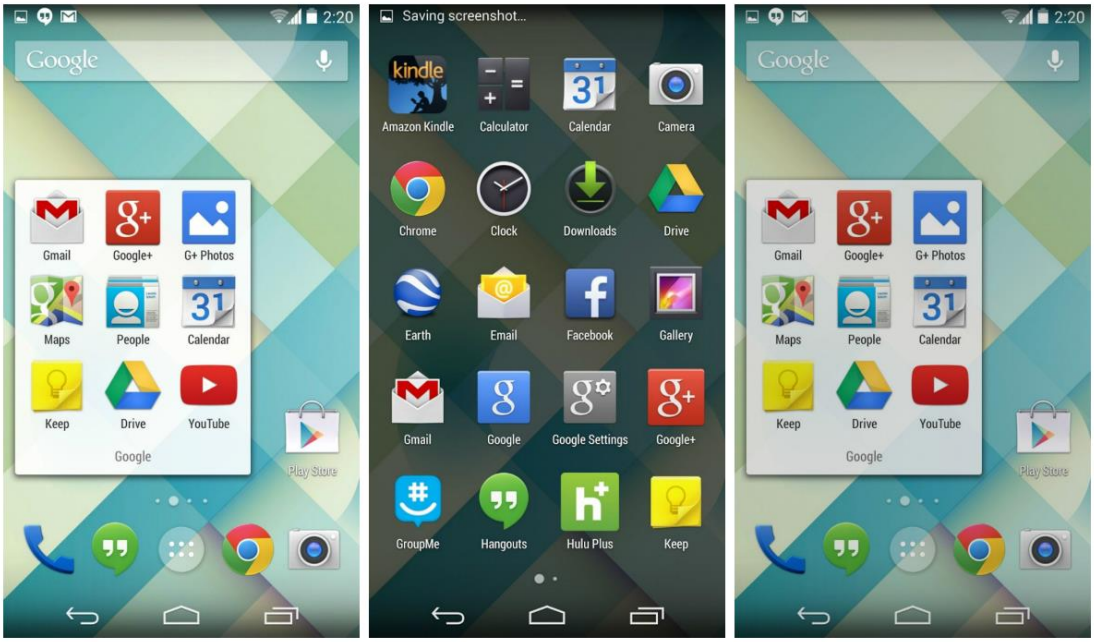
- Android 0.9 Aug 2008 22
- Android 1 Apple pie Sep 2008 23
- Android 1.1 Banana bread Feb 2009 09
- Android 1.5 Cupcake Apr 2009 30
- Android 1.6 Donut Sep 2009 15
- Android 2.0 Éclair Oct 2009 26
- Android 2.0.1 Éclair Dec 2009 03
- Android 2.1 Éclair Jan 2010 12
- Android 2.2 Froyo May 2010 20
- Android 2.3 Gingerbread Dec 2010 06
- Android 2.3.3 Gingerbread Feb 2011 09
- Android 3.0 Honeycomb Feb 2011 22
- Android 3.1 Honeycomb May 2011 10
- Android 2.3.4 Gingerbread May 2011 10
- Android 3.2 Honeycomb Jul 2011 15
- Android 2.3.5 Gingerbread Jul 2011 25
- Android 2.3.6 Gingerbread Sep 2011 02
- Android 3.2.1 Honeycomb Sep 2011 20

- Android 2.3.7 Gingerbread Sep 2011 21
- Android 3.2.2 Honeycomb Sep 2011 30
- Android 4.0 Ice Cream Sandwich Oct 2011 18
- Android 4.0.1 Ice Cream Sandwich Oct 2011 19
- Android 4.0.2 Ice Cream Sandwich Nov 2011 28
- Android 3.2.4 Honeycomb Dec 2011 15
- Android 4.0.3 Ice Cream Sandwich Dec 2011 16
- Android 3.2.6 Honeycomb Feb 2012 15
- Android 4.0.4 Ice cream Sandwich Mar 2012 28
- Android 4.1 Jelly Bean Jul 2012 09

- Android 4.1.1 Jelly Bean Jul 2012 23
- Android 4.1.2 Jelly Bean Oct 2012 09
- Android 4.2 Jelly Bean Nov 2012 13
- Android 4.2.1 Jelly Bean Nov 2012 27
- Android 4.2.2 Jelly Bean Feb 2013 11
- Android 4.3 Jelly Bean Jul 2013 24
- Android 4.4 KitKat Oct 2013 31
- Android 4.4.1 KitKat Dec 2013 05
- Android 4.4.2 KitKat Dec 2013 09
- Android 4.4.3 KitKat Apr 2014 14
- Android 4.4.4 KitKat Jun 2014 23
- Android 5.0 Lollipop Oct 2014 17
- Android 5.0.1 Lollipop Dec 2014 02
- Android 5.0.2 Lollipop Dec 2014 19
- Android 5.1 Lollipop Mar 2015 09
- Android 5.1.1 Lollipop Apr 2015 21
- Android 6 Marshmallow Oct 2015 05
- Android 6.0.1 Marshmallow Dec 2015 07
- Android 7.0 Nougat Aug 2016 22
- Android 7.1 Nougat Oct 2016 04

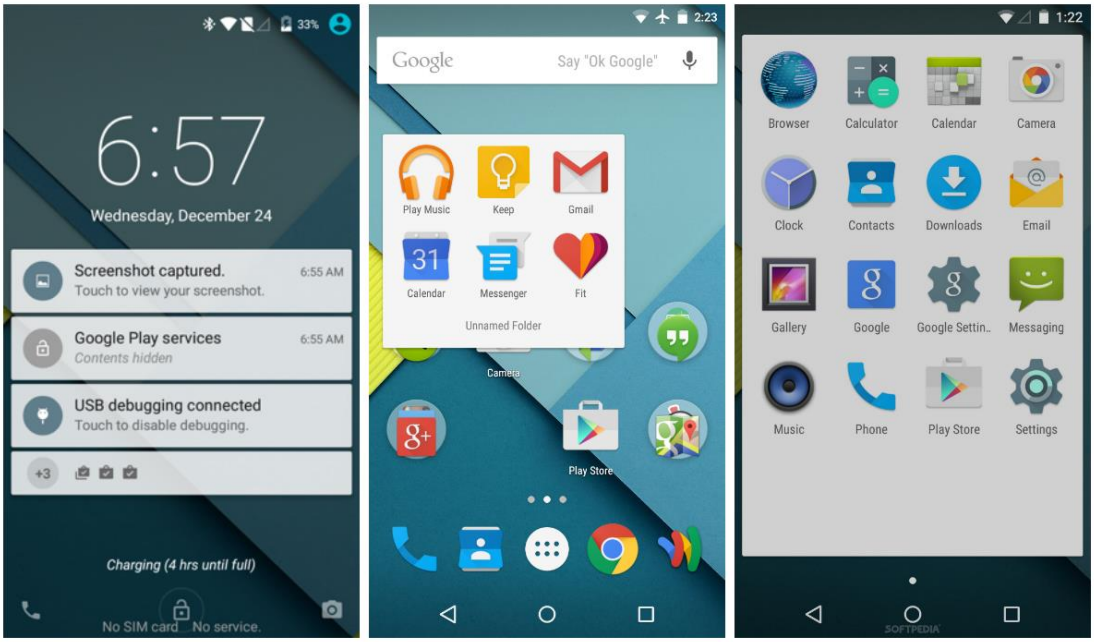
Ανάλυση εκδόσεων

KitKat (Android 4.0)



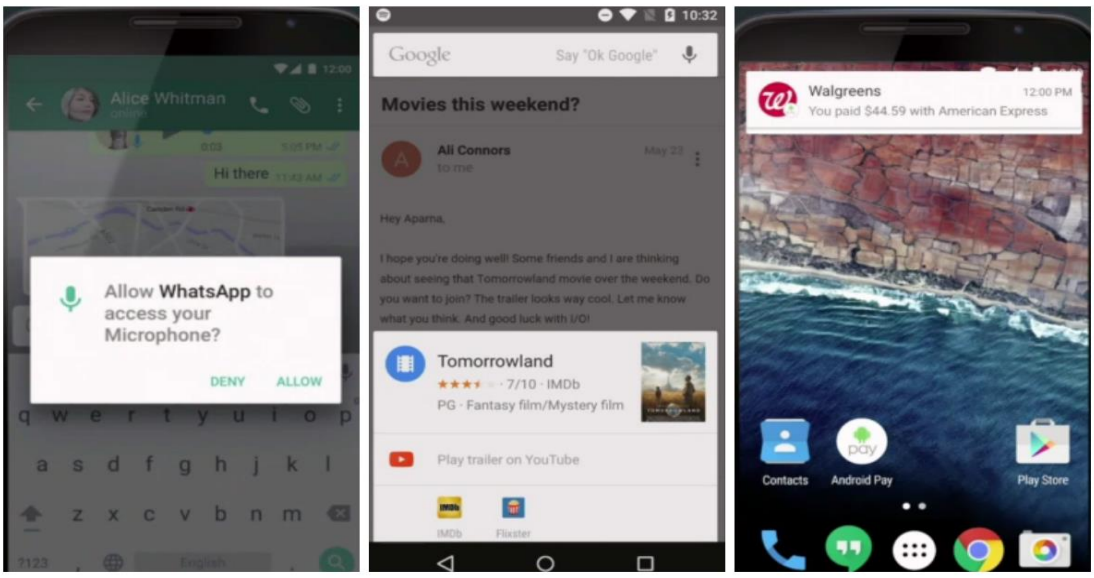
Android 4.4-4.4.4 KitKat είναι ένα λειτουργικό σύστημα βασισμένο κυρίως σε συσκευές android το οποίο δημοσιεύτηκε τον Σεπτέμβριο 3,2013 και επικεντρώθηκε κυρίως στην βελτιστοποίηση της απόδοσης σε καινούριες συσκευές που είχαν περιορισμένους πόρους διαθέσιμους. Δηλαδή μόνο όσες συσκευές είχαν μνήμη 512Mb θα μπορούσαν να τρέξουν με αποτελεσματικότητα την συγκεκριμένη έκδοση.

Lollipop (Android 5.0)



Android 5.0 – 5.1.1 "Lollipop" δημοσιεύτηκε στις 25 Ιουνίου του 2014 από την Google αλλά έγινε διαθέσιμη για εγκατάσταση και χρησιμοποίηση στις 12 Νοεμβρίου του 2014 για επιλεγμένες συσκευές της Google όπως για παράδειγμα τις Nexus συσκευές και τις συσκευές που είχαν εγκαταστημένη την Google Play . Κύριο χαρακτηριστικό της συγκεκριμένης έκδοσης είναι το περιβάλλον εργασίας χρήστη χτισμένο γύρω από μια διαδραστική γλώσσα. Εικόνες , κινούμενα σχέδια και το multitasking επανασχεδιάστηκαν με το παραπάνω περιβάλλον εργασίας.

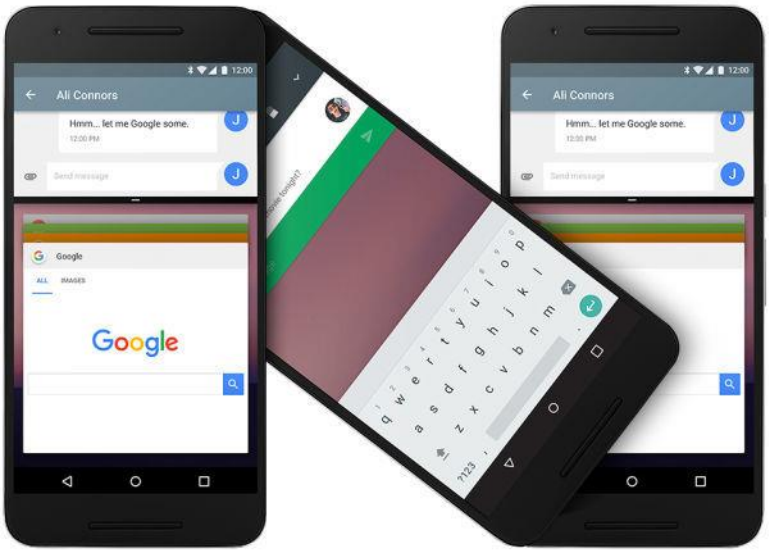
Marshmallow (Android 6.0)



Android 6.0 – 6.0.1 "Marshmallow" είναι η έκτη έκδοση στην ιστορία του Android . Δημοσιεύτηκε τον Μαιο του 2015 και έγινε διαθέσιμη τον Οκτώβριο του 2015 . Η συγκεκριμένη έκδοση αφορούσε κυρίως την βελτιστοποίηση της υπάρχουσας εως τότε

διεπαφής του χρήστη και όχι τόσο τον επανασχεδιασμό . Έβαλε σε λειτουργία τον σένσορα για τα δακτυλικά αποτυπώματα και βελτίωσε την ζωή της μπαταρίας που σημαίνει ότι μπορεί να διαρκούσε και μια μέρα η μπαταρία χωρίς φόρτιση .

Nougat (Android 7.0)



Android 7.0 - 7.1.1 “Nougat” είναι η έβδομη και τελευταία έκδοση του Android λειτουργικού συστήματος και έγινε διαθέσιμη για χρησιμοποίηση στις 22 Αυγούστου του 2016 με τις συσκευές Nexus να δέχονται πρώτες την συγκεκριμένη αναβάθμιση. Η πρώτη μεγάλη αλλαγή είναι η υποστήριξη διπλών οθονών που σημαίνει ότι οι εφαρμογές μπορούν να τρέχουν δύο εφαρμογές σε μία όψη.

Ασφάλεια Android

Στις μέρες μας η ασφάλεια των smartphones παίζει σημαντικό ρόλο στην ζωή μας διότι υπάρχουν πολλές εφαρμογές οι οποίες άλλες προσπαθούν να υποκλέψουν τα προσωπικά μας δεδομένα για τις δώσουν σε τρίτους ή να υποκλέψουν στοιχεία του e-banking μας και υπάρχουν άλλες εφαρμογές οι οποίες στόχο έχουν να καταστρέψουν την κινητή μας συσκευή.

Το android σύστημα από άποψη ασφάλειας είναι πολύ ασφαλές διότι έχει πολλά επίπεδα προστασίας για να κρατήσει έναν ιό μακριά και απαιτεί την έγκριση του χρήστη για να κάνει οποιεσδήποτε αλλαγές στο σύστημα ή για να πάρει δεδομένα από αυτό. Το android είναι προσπαθεί να προστατέψει τον χρήστη αλλά παράλληλα του δίνει την δυνατότητα να έχει την τελευταία λέξη σε ποιόν θα δώσει έγκριση .

Μέχρι στιγμής ο χειρότερος ιός στην ιστορία του android ενεργεί μολύνοντας την συσκευή όταν λάβουμε ένα MMS μήνυμα. Σε αντίθεση με άλλους ιούς ο συγκεκριμένος δεν είναι απαραίτητο να ενεργοποιηθεί όταν ανόιξουμε το μήνυμα . Όταν το μήνυμα παραληφθεί ο ιός ενεργοποιεί κώδικα ο οποίος δίνει τον απόλυτο έλεγχο της android συσκευής δηλαδή θα μπορεί ο επιτιθέμενος να κάνει τα πάντα από το να αντιγράψει δεδομένα μέχρι να πάρει τον έλεγχο της συσκευής και να ελέγξει την κάμερα ή το μικρόφωνο.

Λογαριασμός root

Λέγοντας ότι κάποια συσκευή είναι rooted εννοούμε τον χρήστη εκείνον ο οποίος έχει περισσότερα δικαιώματα από κάποιον άλλον . Σε πολλές των περιπτώσεων με το rooting ο χρήστης μπορεί να αξιοποιήσει στο έπακρο τους διαθέσιμους πόρους της συσκευής του αλλά ταυτόχρονα να κάνει ζημιά σε κάποιο από τα κατώτερα επίπεδα του android.

Οι κατασκευαστές των android με την εγκατάσταση του λειτουργικού αποφάσισαν να μην διαθέσουν τις συσκευές ως rooted αποφασίζοντας ποίες εντολές και διαδικασίες θα είναι διαθέσιμες στον χρήστη και ποίες όχι . Ακόμα εγκατέστησαν εφαρμογές που δεν μπορείς να τις απενεργοποιήσεις ή να τις διαγράψεις. Οι περισσότεροι χρήστες δεν έχουν την γνώση να ωφεληθούν στο έπακρο από το Rooting ή να διαχειριστούν την συσκευή με αποτελεσματικό τρόπο όταν τους δοθεί πλήρη προσβασιμότητα .

Με το Rooting μπορούν οι χρήστες να επωφεληθούν από την εγκατάσταση εξαιρετικών εφαρμογών οι οποίες δεν είναι αναγκαίο να υπάρχουν στο Google Play Store αλλά και εφαρμογών που υπάρχουν σε αυτό και το οποίο προυποθέτει να είναι η συσκευή Root.

Υπάρχουν πολλές εφαρμογές οι οποίες προυποθέτουν Rooting και βοηθάνε στο underclock και στο overclock της συσκευής που σημαίνει ότι μπορεί να αλλάξει η συχνότητα του επεξεργαστή κάνοντάς τον πιο γρήγορο ή ποιο αργό βοηθώντας την διάρκεια της μπαταρίας.

Οι κίνδυνοι για τις συσκευές Root είναι πολλοί . Δεν μπορείς να έχεις όλα αυτά τα οφέλη χωρίς να ρισκάρεις τίποτα. Στην πραγματικότητα οι κίνδυνοι είναι πραγματικά μεγάλοι που μπορούν να φτάσουν εώς την καταστροφή του λογισμικού.

Λίγα λόγια για τις συσκευές Android

Home: Το κουμπί Home είναι αρκετά χρήσιμο διότι βοηθάει τον χρήστη να επιστρέφει στην οθόνη έναρξης χωρίς να έχει σημασία σε πιο φάκελο ή σημείο βρίσκεται στο κινητό του τηλέφωνο. Υπάρχει επίσης η δυνατότητα με το κουμπί Home πατώντας παρατεταμένα να ενεργοποιήσει τις φωνιτικές εντολές .

Back: Το Κουμπί Back οταν πατηθεί μεταφέρει τον χρήστη ένα "κόμβο" πίσω απο εκει που βρίσκεται τώρα. Αυτο επιτυγχάνεται με το τηλέφωνό να έχει αποθηκευσεί όλη την "διαδρομή" που έχει ακουλουθήσει ο χρήστης δηλαδή όλες τις εφαρμογές και τις τοποθεσίες web που επισκέφτηκε από την τελευταία φορά που κλειδώθηκε η οθόνη και τον μεταφέρει έναν κόμβο πίσω κάθε φορά που πατάει το κουμπί Back, μέχρι να φτάσει στην αρχική οθόνη

.

Options: Το κουμπί που δεξια του home θα εμφανίσει τυχόν ρυθμίσεις που μπορεί να έχει το κινητό (διαφορετικά σε κάθε συσκευή android). Όπως και το Home έτσι και αυτο έχει ξεχωριστή ιδιότητα σε περίπτωση που πατηθεί παρατεταμένα , έτσι αν το κουμπί options πατηθεί παρατεταμένα μας εμφανίζει τις εφαρμογές που τρέχουν στην εφαρμογή εκείνη την χρονική στιγμή.

VolumeUP / DOWN: Αυτο το κουμπί είναι "υπέυθυνο" για την μεταβολή της έντασης του ήχου. Συνήθως ειναι διπλό κουμπί ,δηλαδή έχει την ιδιότητα να πατίεται απο δυο μεριές πχ αν πατηθεί απο την πάνω μεριά μεταβάλει την ένταση του ήχου προς τα πάνω και αν πατηθεί απο την κάτω μεριά μεταβάλει
την ένταση του ήχου προς τα κάτω. Το κουμπί αυτο μπορεί να πατηθεί και σε συνδιασμό με άλλα κουμπιά για να γίνει μια προκαθορισμένη ενέργεια.

ON / OFF: Αυτό το κουμπί αφορά το αυτονόητο, την ενεργοποίηση ή απενεργοποίηση της συσκευής Android καθώς και αποτελεί πλήκτρο κλειδώματος οθόνης ή εναλλαγή μεταξύ αθόρυβου προφίλ και κανονικής λειτουργίας ήχων συσκευής.

Java

Αντικειμενοστραφής Προγραμματισμός

Στον τομέα της πληροφορικής αντικειμενοστραφή προγραμματισμός ονομάζομε μια μεθοδολογιά κατα την οποία γίνεται ανάπτυξη προγραμμάτων υποστηριζόμενη απο κατάλληλες γλέωσσες προγραμματισμού. Στον Αντικειμενοστραφή Προγραμματισμό στρεφόμαστε στα αντικείμενα τα οποία δημιουργούνται από κατάλληλη επεξεργασία των δεδομένων μελών από τις συναρτήσεις μέλη μιας κλάσης.

- Ενθυλάκωση
- Πολυμορφισμός
- Κληρονομικότητα
- Επαναχρησιμοποίηση

Κλάσεις

Κεντρική ιδέα στον αντικειμενοστραφή προγραμμτισμό είναι η κλάση(class).
Κλάση είναι ένα πρότυπο που ορίζει τη μορφή ενός αντικειμένου. Καθορίζει τα δεδομένα και το κώδικα που δρα πάνω σε αυτά τα δεδομένα. Τα αντικείμενα είναι στιγμιότυπα της κλάσης στην οποία ανήκουν. Ότι ανήκει μέσα σε μία κλάση αποτελεί μέλος της κλάσης.Πρακτικά είναι ένας τύπος δεδομένων ή αλλιώς το προσχέδιο μιας δομής δεδομένων. Οι περισσότερες κλάσεις περιέχουν κώδικα και δεδομένα. Γενικότερα μια κλάση μπορεί να έχει ως μέλη τα εξής:

- Μεταβλητές στιγμιότυπου
- Στατικές μεταβλητές
- Σταθερές

- Μεθόδους
- Κατασκευαστές
- Καταστροφείς
- Δεικτοδότες
- Συμβάντα
- Τελεστές
- Ιδιότητες

Ο καθοριστής προσπέλασης μέλους ο οποίος καθορίζει το ποιος θα μπορέσει να χρησιμοποιήσει μια μεταβλητή ή μια μέθοδο μιας κλάσεις, είναι απαραίτητος μηχανισμός για όλες τις κλάσεις .Στις κλάσεις έχουμε 4 καθοριστές προσπέλασης:

- **Public**(δημόσια): το μέλος μπορεί να προσπελαύνεται από κάθε κώδικα που βρίσκεται μέσα σε ένα πρόγραμμα, φυσικά και εκτός της κλάσης.
- **Private**(ιδιωτική): το μέλος μπορεί να προσπελαστεί μόνο από άλλα μέλη της κλάσης στην οποία βρίσκεται.
- **Protected** (προστατευμένη)

Επίσης, οι μέθοδοι είναι ένα σημαντικό "ατού" του αντικειμενοστραφούς σχεδιασμού. Οι μέθοδοι κλάσης είναι συναρτήσεις που χειρίζονται τα δεδομένα της και σε πολλές περιπτώσεις παρέχουν πρόσβαση σε αυτά. Ουσιαστικά, άλλα μέρη του προγράμματος θα αλληλοεπιδρούν με μία κλάση μέσω των μεθόδων της. Τέλος, μια μέθοδος υλοποιεί μια συγκεκριμένη λειτουργία, μπορεί να χρειάζεται ορίσματα για να δουλέψει και μετά το πέρας λειτουργίας της,να επιστρέφει κάτι σε αυτόν που τη κάλεσε. Ακόμη, ο κατασκευαστής είναι μία μέθοδος, η οποία αρχικοποιεί ένα αντικείμενο της κλάσης όταν αυτό δημιουργείται, δηλαδή δίνει αρχικές τιμές στα δεδομένα του. Έχει το ίδιο όνομα με τη κλάση αλλά δεν έχουν ρητό επιστρεφόμενο τύπο. Συνήθως η προσπέλαση είναι public για να μπορεί να κληθεί εκτός της κλάσης και μπορεί να περιέχει ή όχι ορίσματα. Όλες οι κλάσεις έχουν κατασκευαστές χωρίς ορίσματα, ακόμα και αν δε δηλώσουμε ρητά εμείς κάποιον (σε αυτή τη περίπτωση το κάνει ο μεταγλωττιστής και αρχικοποιεί όλες τις μεταβλητές -μέλη με τις προεπιλεγμένες τιμές τους). Επιπλέον, μια καθοριστική μέθοδος μπορεί να θεωρηθεί και ο καταστροφέας. Είναι μια μέθοδος - μέλος, η οποία απελευθερώνει τη μνήμη που δεσμεύθηκε για τη δημιουργία ενός αντικειμένου (από το τελεστή new). Στην ουσία καταστρέφει το αντικείμενο. Ο καταστροφέας καλείται πριν από τη λεγόμενη συλλογή απορριμμάτων, κάτι που γίνεται στο παρασκήνιο (χωρίς τη συμμετοχή του προγραμματιστή), έτσι ώστε να απελευθερωθεί η μνήμη.

Ενθυλάκωση

Ενθυλάκωση λέμε έναν μηχανισμό προγραμματισμού ο οποιός ειναι υπεύθηνος για να συνδέει τον κώδικα με τα δεδομένα που χειρίζεται και τα κρτάει και τα δύο ασφαλή απο εξωτερικές παρεμβολές και κακομεταχείριση.Η ενθυλάκωση αφήνει μία κλάση να διαχωριστεί και να αναλυθεί σε μικτρότερα κομμάτια προγραμμάτων, κομμάτια τα οποία ειναι ανεξάρτητα. Κάθε κομμάτι προγραμμάτων ειναι υπεύθηνο να κάνει την εργασία του ανεξάρτητα απο τα άλλα κομμάτια. Τα δεδομένα και ο κώδικας των προγραμμάτων συνδέονται με τρόπο τέτοιο ώστε να δημιουργείται ένα "μαύρο κουτί"στο οποίο βρίσκονται ολα τα δεδομένα και ο κώδικας που χρειάζονται. Έτσι δημιουργείται ένα αντικείμενο. Μέσα σε αυτό ο κώδικας και τα δεδομένα είναι ιδιωτικά – δηλαδή σημαίνει ότι μπορούν να προσπελαστούν μόνο από άλλα μέρη του αντικειμένου ή δημόσια (public)- που σημαίνει ότι μπορούν να προσπελαστούν και από κομμάτι του προγράμματος που είναι εκτός από το αντικείμενο. Τόσο τα δεδομένα όσο και ο κώδικας έχουν την ονομασία μέλη της κλάσης. Τα δεδομένα τα λέμε μεταβλητές μέλους ενώ,ο κώδικας μέθοδοι μέλους.

Πολυμορφισμός

Με τον πολυμορφισμό δύνεται η δυνατότητα στο ίδιο αντικείμενο, όταν τρέχει η εφαρμογή, να μνημονεύεται σε διαφορετικές κλάσεις και να επηρεάζεται με διαφορετικό τρόπο σε διαφορετικά αντικείμενα. Κλάσεις οι οποίες είναι διαφορετικές μπορούν να έχουν συναρτήσεις (συμπεριφορές) με ίδια όνομασια και ίδιο βασικό στόχο, αλλά με διαφορετική εφαρμογή. Για παράδειγμα, σε ένα αμάξι ο συμπλέκτης ,το γκάζι και του φρένο υποστηρίζουν τη συνάρτηση πάτησε. Είναι διαφορετική όμως η εφαρμογή της λειτουργίας σε κάθε πεντάλ.

Κληρονομικότητα

Η κληρονομικότητα χρειάζεται για να μπορούμε να έχουμε κάποια κοινά χαρακτηριστικά σε διαφορετικές κλάσεις ή αλλιώς να μπορούμε να έχουμε κλάσεις οι οποίες θα είναι σε κάποια μορφή υποκατηγοριών άλλων κλάσεων . Η βασική αρχή σ' αυτή τη διαίρεση είναι ότι κάθε υποκατηγορία έχει κοινά χαρακτηριστικά με την κλάση από την οποία προέρχεται. Για παράδειγμα,εργαζόμενοι στον τομέα παραγωγής , εργαζόμενοι στον τομέα του λογιστικού τμήματος και εργαζόμενοι στον τομέα πληροφορικής ενός εργοστασίου έχουν όλοι κοινο χαρακτηριστικό το οποίο ειναι οτι όλοι ειναι εργαζόμενοι στο ίδιο εργοστάσιο.Εκτός απ' αυτά τα χαρακτηριστικά, που είναι κοινά για όλα τα μέλη της αρχικής κλάσης, κάθε υποκατηγορία έχει και άλλα διαφορετικά χαράκτηριστικά σε σχέση με τις άλλες κατηγορίες. Για παράδειγμα οι εργαζόμενοι στον τομέα παραγωγής έχουν διαφορετικό μισθό και ωράριο απο τους εργαζομένους του τμήματος πληροφορική, όμως δεν παύουν και οι δυο κατηγορίες να έχουν κοινά χαρακτηριστικά που τα έχουν κληρονομήσει απο την κατηγορία των εργαζομένων του εργοστασίου.

Επαναχρησιμοποίηση

Μια κλάση, δεν μπορεί να χρησιμοποιηθεί μόνο για ένα πρόγραμμα . Μπορούμε επίσης να την διανέμουμε και σε άλλα προγράμματα για χρήση. Αυτο ονομάζεται επαναχρησιμοποίηση.

Μία κλάση, αφού δημιουργηθεί, μπορεί να διανεμηθεί για χρήση σε πολλά προγράμματα. Αυτό καλείται επαναχρησιμοποίηση και είναι σαν τις βιβλιοθήκες συναρτήσεων που χρησιμοποιούν οι διαδικαστικές γλώσσες. Με την έννοια της κληρονομίκοτητας πλέον μπορούμε να κάνουμε και άλλα πράγματα με την επαναχρησιμοποίηση. Μπορούμε πλέον πολύ εύκολα να πάρουμε μια κλάση που ήδη υπάρχει χωρίς να την τροποποιήσουμε να της βάλουμε και να της προσθέσουμε περισσότερα χαρακτηριστικά. Η νέα κλάση θα κληρονομήσει τις δυνατότητες της παλιάς, αλλά μπορεί να χρησιμοποιήσει και τα νέα δικά της χαρακτηριστικά.

Λίγα λόγια για την γλώσσα Java

''Η Java είναι μια γλώσσα αντικειμενοστραφούς προγραμματισμού η οποία σχεδιάστηκε και συντηρείται από την SunMicrosystems. Ξεκίνησε το 1991 σαν μέρος ενός μεγαλύτερου σχεδίου που αφορούσε την ανάπτυξη λογισμικού για καταναλωτικά ηλεκτρονικά. Πρόκειται για μικρές, αξιόπιστες, φορητές, πραγματικού χρόνου συσκευές που στην αρχή βασιζόντουσαν στην C++. Αρκετά προβλήματα όμως παρουσιάστηκαν και η γλώσσα C++ δεν μπόρεσε να εφαρμοστεί τελικά. Χρειάστηκε να αναπτυχθεί μία νέα γλώσσα: η Java. 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 γλώσσα *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 αντικειμενοστραφή 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 αγαπημένης ποικιλίας καφέ για τους δημιουργούς της. Η επίσημη εμφάνιση της Java αλλά 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 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.'' Wikipedia

Χαρακτηριστικά της Java

Απλή

Η ομάδας της Sun που ανέπτυξε την Java είχε σαν στόχο μια γλώσσα εύκολη στην χρήση ώστε να μην απαιτεί αρκετό από το χρόνο του χρήστη να την μάθει. Παρόλο που οι περισσότεροι developers δούλευαν πάνω σε C και C++ οι γλώσσες αυτές δεν ηταν κατάλληλες για το αρχικό σχέδιο παρόλα αυτά η java σχεδιάστηκε με βάση την C++ και ο στόχος της ηταν να γίνει οσο το δυνατον ποιο κατανοητη στον χρήστη.

Η Java προσπαθεί να φέρει όσο δυνατόν την γλώσσα πιο κοντά στον χρήστη αφερώντας έτσι μερικά χαρακτηριστικα της C++.
Μία από τις διεργασίες που προστλέθηκαν είναι η automatic garbage collection είναι μια διεργασία στην οποία διευκολύνει κατά πολύ τον προγραμματισμό σε Java καθώς σταματάει ο χρήστης να ασχολείτε με την διαχείριση της μνημης. Ο χρήστης πλέον δεν χρειάζεται στην αποδεσμευει την μνήμη καθώς αυτό γίνεται αυτόματα αυτό το σύστημα με αυτό τον τρόπο μέγαλο μερος από την δουλειά που έπρεπε να κάνει ο προγραμματιστης γίνετε αυτόματα και αυτό έχει ως αποτέλεσμα την μείωση τον bugs.

Ο κώδικας της java περιορίζεται σε μικρές σε μέγεθος απαραίτητων εργαλείων και βιβλιοθηκών . Αυτό είναι ένα πλεονέκτημα καθώς κάνοντας τον κώδικα πιο απλό και πιο μικρό σε μέγεθος μπορει να κατέβει από το δίκτυο και να τρέξει σε μικρές μηχανές χώρις απολύτος κάποιο πρόβλημα

Αντικειμενοστραφής

Η java ανήκει στην οικεγένεια της αντικειμενοστρέφιας καθώς η τεχνική που χρησιμοποιεί για τον σχεδιασμό ενός προγράμματος συγκεντρώνεται σε αντικείμενα. Τα αντικείμενα που σχεδιαζονται στην Java είναι ο συνδιασμός διαδικασιών , δεδομένων ,διαδικασιών και λειτουργιών.

Τα αντικείμενα δεν είναι ανεξάρτητα αλλά το ένα έχει μια εξάρτηση με το άλλο μπόρει να κληρονομήσει δεδομένα και ιδιότητες από τα άλλα αντικείμενα .

Ο προγραμματισμός αυτός είναι υψηλού επιπέδου χρησιμοποιείτε επί των πλείστων σε σημαντικές και μεγάλες εφαρμογές καθώς είναι γρήγορος, αφαιρετικός και αποτελεσματικός. Η Java περιέχει τις ευκολίες της C++ με επεκτάσεις από την Objective C.

Συμβατή με Δίκτυα

Η Java έχει μεγάλη βιβλιοθήκη από ρουτίνες για την συνεργασία με τα πρωτόκολλα HTTP και FTP. Με αυτόν τον τρόπο η java προσφέρει ευκολότερη δικτυακή σύνδεση σε σχέση με την C και C++ .Τα προγράμματα σε Java μπορούν να έχουν πρόσβαση σε αντικείμενα με την ίδια άνεση που ένας χρήστης προσπελάζει ένα τοπικό σύστημα αρχείων ακόμα και μέσω του δικτύου.

Σταθερή

Ο προγραμματισμός σε Java δίνει την δυνατότητα σύνταξης προγραμμάτων που θα είναι αξιόπιστα από όλες τις πλευρές καθώς μπορει να γίνει από νωρίς έλεγχος για πιθανά προβλήματα σε πραγματικό χρόνο , αποφεύγοντας με αυτόν τον τρόπο να προκληθούν σοβαρά λάθη .

Η Java υπερτερεί απέναντι στην C και C++ με το μοντέλο δεικτών το οποίο έχει εξαλείφοντας με τον τρόπο αυτό την πιθανότητα της επαναχρησιμοποίησης της μνήμης που είχε ως αποτέλεσμα την καταστροφή των δεδομένων.Οι προγραμματιστές δεν χρειάζεται να ανυσηχούν για την πιθανή τροποποιήση της μνήμης γιατί η Java χρησιμοποιεί πραγματικούς πίνακες σε αντίθεση με τους αριθμητικούς δείκτες δεν επιτρέπει την μην εγκεκριμένη πρόσβαση .

Ασφαλής

Ιδιαίτερη προσοχή έχει δοθέι στην ασφάλεια της java καθώς και η χρήση της θα γίνεται και σε ανοιχτά δικτυωμένα περιβάλλοντα. Ακολουθεί το πρότυπο ασύμμετρης κρυπτογράφισης και επιτρέπει την δημιουργία προγραμμάτων “virus free” .

Από την στιγμή που η γλώσσα αυτή εμποδίζει την μην εγκεκριμένη πρόσβαση στην μνήμη καταπολεμά άμεσα μεγάλο ποσοστό από τους ιούς.

Ουδέτερη της Υποκείμενης Αρχιτεκτονικής

Για να μπορεί η Java να υποστιρίζει δικτυακές εφαρμογές παντού στο δίκτυο και να μη επηρεάζεται από την ποικιλία διαφορετικών συστημάτων (λειτουργικών συστημάτων “OS” και διαφορετικών CPU) .Το πρόγραμμα θα πρέπει να περάσει από δύο φάσεις για να μπορέσει να προκύψει η εκτελέσιμη μορφή του.

Στην πρώτη φάση γίνεται η μεταγλώττιση του πηγαίου κώδικα σε Java bytecodes που στην ουσία είναι μια ενδίαμεση γλώσσα πριν το εκτελέσιμο που θέλουμε, τα Java bytcodes δεν σχετίζοναι με την πλατφόρμα οπότε δεν περιορίζεται από λειτουργικά συστήματα και διαφορετικά CPU. Με την βοήθεια του Interpreter (ερμηνευτής) γίνεται η μετατροπή κάθε εντολής σε ανάλογη δυαδική μορφή

του κάθε συστήματος η διαδικασία αυτή γίνεται κάθε φορά που εκτελείται το πρόγραμμα ενώ η μεταγλώττιση (compilation) συμβαίνει μια και μόνο φορά.

Η Java χαρακτηρίζεται από την τεχνική (write once , run anywhere) , τα προγράμματα αυτά μεταγλωττίζονται μια φορά σε Java bytecodes και μπορούν να εκτελεστούν σε οποιαδήποτε σύστημα εμπεριέχει JVM(Java Virtual Machine ¨Interpreter¨).

Φορητή

Η Java για να μπορέσει να να είναι εκτελέσιμη σε ανεξάρτητα σύστημα το βοηθάει επίσης ότι έχει καθορισμένο το τύπο δεδομένων ανεξάρτητα την CPU του υπολογιστή σε αντίθεση με την C και C++ που αυτά προσδιοριζονται ανάλογα με την CPU και λειτουργικό σύστημα.

Σε συνδιασμό με το γεγονος ότι είναι ανεξάρτητη της υποκείμενης πλατφόρμας που και αυτό αποτελεί μέρος του ότι η γλώσσα είναι φορητή.

Interpreter

Ο Interpreter (ερμηνευτής) μετραφάζει τα Java bytecodes ανάλογα την πλατφόρμα που έχουμε , αυτό γίνετε σε πραγματικό χρόνο σε δυαδικό σύστημα σε γλώσσα μηχανής τα δεδομένα αυτά δεν αποθηκεύονται σε κάποια μνήμη του συστήματος μας.

Τα bytecodes αυτά μεταφέρουν πληροφορίες οι οποίες είναι απαραίτητες κατά της εκτέλη καθώς και βοηθούν στην πραγματοποίηση έλεγχο που εκτελεί ο συνδετής (linker).

Υψηλής Απόδοσης

Υπάρχει υψηλή απόδοση καθώς η διαδικασία στην οποία παράγει τις εντολές μηχανής είναι γρήγορη και απλή. Το τελικό αποτέλεσμα που προκύπτει μετά την μεταγλώττιση γίνεται με αυτόματη κατανομή καταχωρητών με αυτόν τον τρόπο η τελική μορφή του κώδικα έχει υψηλή ταχύτητα εκτέλεσης σε αυτό παίζει ρόλο και το μικρό μέγεθος του κώδικα .

Multithreaded

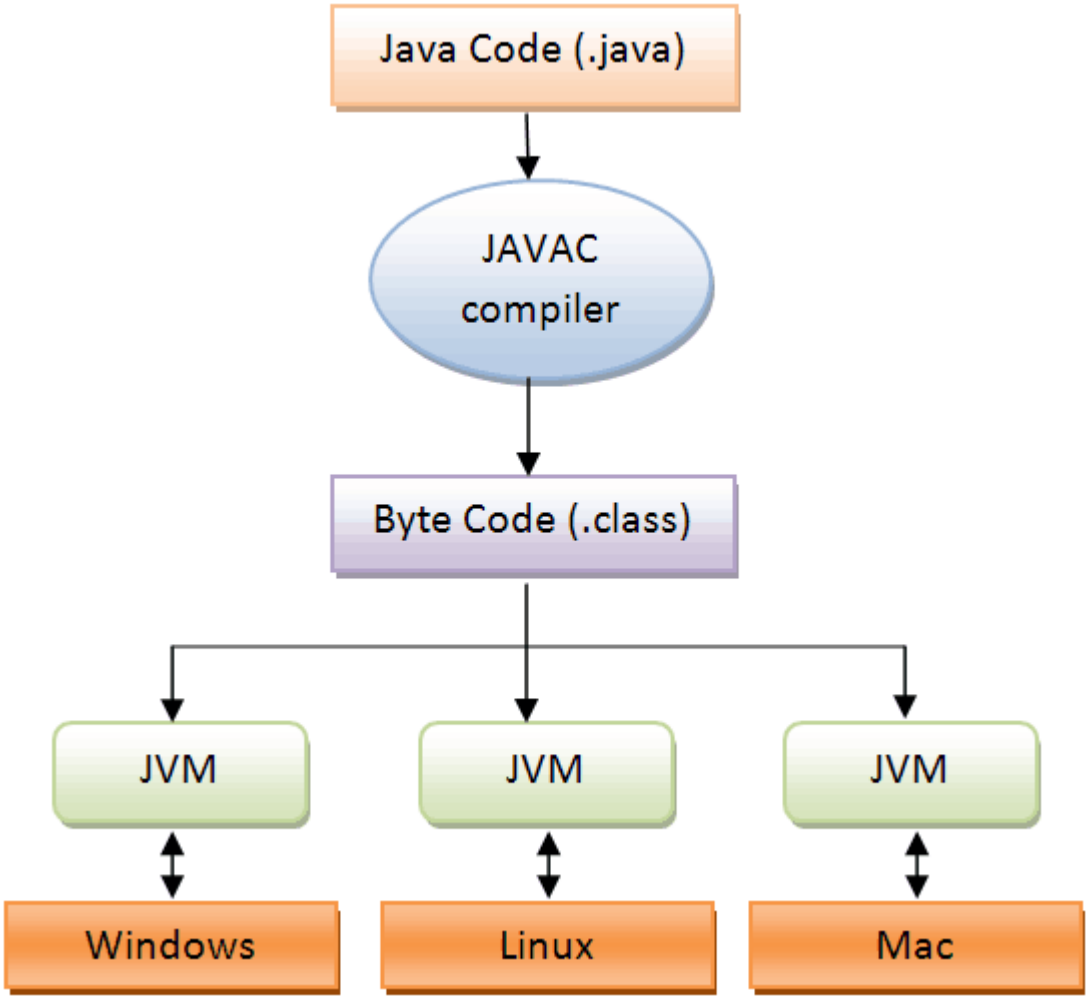
Η java δίνει την δυνατότητα να μπορούν να εκτελεστούν πολλές διαδικασίες ταυτόχρονα αυτό έχει μεγάλο πλεονέκτημα απέναντι σε C και C++ που εκτελούν μια διαδικασία κάθε φορά καθώς και η java μπορεί να αντιμετωπίσει πολλές καταστάσεις ταυτόχρονα .

Δυναμική

Η java έχει αναπτυχθεί με τέτοιο τρόπο ώστε να δίνει την δυνατότητα να είναι προσαρμοζόμενη για να μπορεί να εξελίσσεται ανάλογα με το περιβάλλον .

Με αυτόν τον τρόπο χωρίς να επηρεάζονται οι υπάρχουσες ήδη εφαρμογες μπορεί να γίνει προσθήκη κάποιας βιβλιοθήκης η νέων μεθόδων.

Η εικονική μηχανή της Java



Για να εκτελεστεί κάποιο πρόγραμμα σε Java θα πρέπει να περάσει κάποιες διαδικασίες, μια από της διαδικασίες είναι η μεταγλώττιση του σε κώδικα byte η bytecode αυτό γίνεται μέσω του μεταγλωττιστή (JavaC).Για να εκτέλεση το bytecode αναλαμβάνει το JVM (Java Virtual Machine) του κάθε συστήματος να διαβάσει τα αρχεία .class.

Στην επόμενη φάση το bytecode μεταφράζεται σε γλώσσα μηχανής ανάλογα με το λειτουργικό μας σύστημα για να μπορεί να εκτελεστεί αυτό συμβαίνει σε κλασική εικονική μηχανή καθώς και υπάρχουν πιο σύγχρονες JVM στις οποίες γίνετε μεταγλώττιση του bytecode σε κώδικα μηχανής βελτιώνοντας με αυτόν τον τρόπο την ταχύτητα σε μεγάλο ποσοστό. Χωρίς την JVM θα ηταν αδύνατη η εκτέλεση των προγραμμάτων γραμμένα σε Java. Βέβαια το JVM δεν είναι ανεξάρτητης πλάτφορμας και κάθε λειτουργικό σύστημα θα πρέπει να έχει το δικο του JVM. Έτσι υπάρχουν διαφορετικές JVM για (Linux, Windows, παιχνιδομηχανές, Macintosh, κινητά τηλέφωνα, Unix, κλπ).

Επιδόσεις της Java

Οι επιδόσεις της Java λόγο της εικονικής μηχανής είναι χαμηλότερες σε σχέση με τις επιδόσεις της C και της C++ διότι αυτές οι γλώσσες είναι υψηλού επιπέδου .

Καθημερινά γίγονται προσπάθειες για την βελτιστοποιηση της JVM είτε από την μεριά της Sun είτε από άλλες εταιρίες που προσπαθούν να φτιάξου ένα βελτιωμένο JVM για να προσφέρουν καλύτερα αποτελέσματα οι οποίες μπορεί σε κάποια σημεία να προσφέρουν καλύτερα και σε κάποια άλλα χειρότερα.

Πλέον με τους μεταγλωττιστες Just in time (JIT) που έχουν καθιερωθεί τα τελευταία χρόνια η διαφορά σε ταχύτητα σε σχέση με την C++ έχει ελαχιστοποιηθεί καθώς και η μεταγλώττιση γίνετε σε γλώσσα μηχανής.

Στις τελευταίες εκδόσεις της η java με την τεχνολογία του HOTSPOT πλησιάζει και ξεπερνάει πολλές φορές τον εγγενή κώδικα (native code).

Java Packages

Σε προγράμματα με χιλιάδες γραμμές είναι αρκετά δύσκολο να βρέθουν μοναδικά ονόματα για κάθε κλάση καθώς δεν θα πρέπει να χρησιμοποιηθούν ονόματα κλάσεων της βιβλιοθήκης της Java η ακόμα και ονόματα που μπορεί να δωθούν από διαφορετικά άτομα τα οποία εργάζονται στο ίδιο project .

Για το πρόβλημα που προαναφέραμε στην γλώσσα της java έρχεται να αντιμετωπίσει αυτό το πρόβλημα τα javaPackages στα οποία με δίαφορους τρόπους λύνουν αυτό το πρόβλημα δίνοντας συσχέτιση μεταξύ του με βάση τον σκοπό , την κληρονομικότητά τους η την εμβέλεια .Τα javaPackages πλέον επιτρέπουν την οργάνωση των κλάσεων σε μονάδες μιώνοντας ετσι προβλήματα από διενέξεις ονομάτων και μπορούν να χρησιμοποιηθούν για ταυτοποίηση των κλάσεων.

XML

Λίγα λόγια για την XML

Η XML είναι μια γλώσσα για την δόμηση δεδομένων με την εννοια αυτή εννοούμε μια συλλογή στοιχείων δεδομένων , στην ουσία η xml είναι ένα σύνολο κανόνων για την το σχεδιασμό μορφών κειμένου για την διευκόλυνση στην δόμηση των δεδομένων μας.

Η XML μπορεί να να χρησιμοποιεί ετικέτες και γνωρίσματα όπως και η HTML αλλά διαφέρει στο ότι η xml κανει διευκρίνιση για κάθε ετικέτα.Αν και η μορφή του XML είναι μορφή κειμένου και μπορει να διαβαστεί χωρίς το πρόγραμμα το οποίο το παρήγαγε δεν προορίζεται για ανάγνωση βέβαια δίνεται η δυνατότητα με έναν απλό διορθωτή να κάνεται κάποια εκσφαλμάτωση εάν προκύψει.

Η xml είναι σχετίκα καινούρια αλλα συνδιάζει διαφορετικές τεχνολογίες όπως (xlink,Xpointer,CCS,κτλπ) και δεν χρειάζεται άδεια χρήσης καθώς επίσης λειτουργεί ανεξάρτητα του συστήματος υλικού και είναι ευρείας υποστήριξης.

Android Studio

Στην έρευνα την οποία διεξήγαμε όσον αφόρα το πρόγραμμα το οποίο θα χρησιμοποιήσουμε για να προγραμματίσουμε για την δημιουργία της εφαρμογής μας βρίκαμε αρκετά προγράμματα αλλά και δύο που ξεχώριζαν από τα υπόλοιπα αυτά τα δύο είναι το Android Studio και το Eclipse.

Αν και το eclipse ήταν ευρέως διαδεδομένο γιατί είχε γραφίκο περιβάλλον στο οποίο σου έδιδε πολλές δυνατότητες ακόμα και σε άπειρους χρήστες αλλά σε περιορίζεσαι πολύ σε σχέση με το Android studio το οποίο απευθείνεται σε προγραμματιστές καθώς και από το 2013 που ανακοινώθηκε από την Google είναι διαθέσιμο δωρεάν για όλα τα λειτουργικά συστήματα.

Ο λόγος λοιπόν που προτιμήσαμε το android studio έναντι στο eclipse ηταν ότι είναι αποκλειστικά για προγραμματισμό Android και σήμερα είναι το κυριο IDE της Google για εφαρμογές.

VS

SPECIAL SLIDE : "ANDROID STUDIO VS. ECLIPSE ADT"

FEATURE	ECLIPSE ADT	ANDROID STUDIO
Build system	Ant	Gradle
Maven-based build dependencies	No	Yes
Build variants and multiple-APK generation (great for Android Wear)	No	Yes
Advanced Android code completion and refactoring	No	Yes
Graphical layout editor	Yes	Yes
APK signing and keystore management	Yes	Yes
NDK support	Yes	Coming soon

Gradle

Το Gradle είναι χτισμένο πάνω στις έννοιες του Apache Ant και Apache maven και ανήκει στο λογισμικό ανοιχτού κώδικα.

Βασίζεται στην Groovy η οποία ανήκει σε γλώσσα προγραμματισμού αντικειμενοστρέφιας και είναι διαδεδομένη σαν γλώσσα scripting Java projects.

Έχει πολλά θετικά όπως επαναχρησιμοποίηση του ήδη υπάρχων κώδικα και πόρων , διαφορετικές λειτουργίες πάνω στο ίδιο project με την δημιουργια πολλαπλών apks , υποστηρίζει πολλαπλές γλώσσες και προσαρμόζεται έυκολα .

Την μετατροπή του κώδικα της εφαρμογής σε εκτελέσιμο APK την αναλαμβάνει το Gradle.Σε όλα τα Projects θα πρέπει να υπάρχουν δυο φάκελοι (SRC και RES) όπου το SRC αντιπροσωπεύει τον κώδικα που έχουμε γράψει (¨κλάσεις + μέθοδοι¨) και στον φάκελο RES όλες τις πηγές της εφαρμογής συνήθως XML κώδικας.

Το Gradle αναλαμβάνει να μετατρέψει τον java κώδικα σε dex κώδικα ο οποίος είναι σε κατάληλη μορφή για να μπορέσει να εκτελεστεί σε android OS.

Σε κάθε Project στο android studio θα πρέπει να υπάρχουν δύο Gradle scripts:

- Ενα σχετικά με το πως θα εκτελεστεί και τι ρυθμίσεις θα έχει.

- Ενα που αφορά το project που εκτελούμε (έκδοση εφαρμογής κλπ)

Κατασκευή της εφαρμογής

Σε αυτό το σημείο θα περιγράψουμε τα σενάρια κατασκευής της εφαρμογής και θα αναλύσουμε το κάθε κομμάτι του ξεχωριστά.

Αποθήκευση δεδομένων χρήστη

Βάση δεδομένων

Για την αποθήκευση δεδομένων χρήστη θα κατασκευάσουμε μια βάση δεδομένων που θα ανταποκρίνεται στη σύνδεση του χρήστη με την εφαρμογή. Η βάση μας θα είναι απλή αφού οι συναλλαγές που γίνονται αφορούν την οντότητα του χρήστη.

Για να λειτουργήσει η εφαρμογή από οποιοδήποτε μέρος και όχι μόνο τοπικά θα πρέπει να ανεβάσουμε τα αρχεία μας ,δηλαδή την βάση μας και τα αρχεία php σε έναν online σέρβερ.

Για τον σκοπό αυτό θα χρησιμοποιήσουμε την ιστοσελίδα του δωρεάν
Hosting 000webhost.com.

Κάνωντας εγγραφή θα πρέπει να εισάγουμε τα αρχεία php και το sql αρχείο στο phpmyadmin.
Το URL της ιστοσελίδας μας είναι sosmanagerapp.000webhostapp.com το οποίο θα το τοποθετήσουμε στην εφαρμογή για το httpPost.

Οι πίνακες που θα εισάγουμε στην βάση δεδομένων είναι :

- **Users:** Σε αυτόν τον πίνακα αποθηκεύονται οι χρήστες που έχουν εγγραφεί για να χρησιμοποιηθούν στην συνέχεια οι πληροφορίες αυτές για την είσοδο στην εφαρμογή κατά κύριο λόγο.
- **SaveCalls:** Σε αυτόν τον πίνακα αποθηκεύονται οι κλήσεις έκτακτης ανάγκης που έγιναν για να τις εμφανίσουμε στην ιστοσελίδα.

Όπως φαίνεται η σχέση που υπάρχει μεταξύ των δύο πινάκων είναι το γνώρισμα **UsersID** το οποίο αποτελεί ξένο κλειδί με αναφορά στο γνώρισμα **ID** από τον πίνακα των πελατών.

Η σχέση αυτή είναι ***ένα προς πολλά***, αφού ένας χρήστης μπορεί να πραγματοποιήσει διάφορες κλήσεις σε υπηρεσίες ενώ δεν μπορεί μια κλήση να προέρχεται από παραπάνω από έναν, χρήστες.

Στην βάση μας έχουμε αποθηκευμένες κάποιες διαδικασίες για να επικοινωνεί η εφαρμογή με την βάση και να διαβάσει ή να εισάγει πληροφορίες σε αυτήν.

Κώδικας Users

```
--
-- Δομή πίνακα για τον πίνακα `users`
--

CREATE TABLE `users` (
  `ID` int(11) NOT NULL,
  `FirstName` text NOT NULL,
  `LastName` text NOT NULL,
  `Country` text NOT NULL,
  `City` text NOT NULL,
  `Address` text NOT NULL,
  `BirthDate` text NOT NULL,
  `PhoneNo` text NOT NULL,
  `UserName` text NOT NULL,
  `PassWord` text NOT NULL,
  `DateCreated` text NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE utf8_unicode_ci;

--
```

Κώδικας saveCalls

```
--
-- Δομή πίνακα για τον πίνακα `saveCalls`
--

CREATE TABLE `saveCalls` (
  `ID` int(11) NOT NULL,
  `ClientID` int(11) NOT NULL,
  `ServiceID` int(11) NOT NULL,
  `ServiceName` text NOT NULL,
  `ClientLongitude` text NOT NULL,
  `ClientLatitude` text NOT NULL,
  `Message` text NOT NULL,
  `DateCreated` text NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE utf8_unicode_ci;
```

ΔΙΑΔΙΚΑΣΙΕΣ

selectAll

Με την διαδικασία αυτήν επιστρέφεται από την βάση όλοι οι εγγεγραμένοι.

Κώδικας:

```
BEGIN
	SELECT *
	FROM Persons;
END
```

signUp

Με την διαδικασία αυτήν εισάγωνται στην βάση οι πληροφορίες εγγραφής.Αυτή η διαδικασία έχει εννιά ορίσματα. Τα ορίσματα είναι τα παρακάτω:

- inFirstName
- inLastName
- inCountry
- inCity
- inAddress
- inBirthDate
- inPhoneNo
- inUserName
- inPassWord

Κώδικας:

```
BEGIN
	INSERT INTO Persons (FirstName, LastName, Country, City, Address, BirthDate, PhoneNo, UserName, PassWord, DateCreated)
	VALUES (inFirstName, inLastName, inCountry, inCity, inAddress, inBirthDate, inPhoneNo, inUserName, MD5(inPassWord), DATE_FORMAT(NOW(),'%d/%m/%Y'));

	SELECT 0;
END
```

signIn

Η διαδικασία αυτή παίρνει ως ορίσματα το Username και Password και ελέγχει αν υπάρχουν στον πίνακα Persons .Εαν υπάρχουν επιστρέφει το ID του αλλίως το αλφαριθμητικο NotFound.

Κώδικας:

```
BEGIN
	SELECT ID INTO @outID FROM tbl_clients
	WHERE (tbl_clients.UserName = inUserName) AND (tbl_clients.PassWord = MD5(inPassWord));

	IF @outID IS NOT NULL
	THEN
		SELECT @outID AS 'Result';
	END IF;
```

```
	IF @outID IS NULL
	THEN
		SELECT ‘ClientNotFound' AS 'Result';
	END IF;
END
```

Save

Με την διαδικασία αυτήν αποθηκεύεται στην βάση κάθε κλήση έκτακτης ανάγκης που πραγματοποιεί ο χρήστης .Αποθηκεύεται το ID του χρήστη και της υπηρεσίας , οι συντεταγμένες της γεωγραφικής θέσης καθώς και το μήνυμα.
Τα στοιχεία αυτά αποθηκεύονται στον πίνακα saveCalls και τα ορίσματα της διαδικασίας είναι τα παρακάτω:

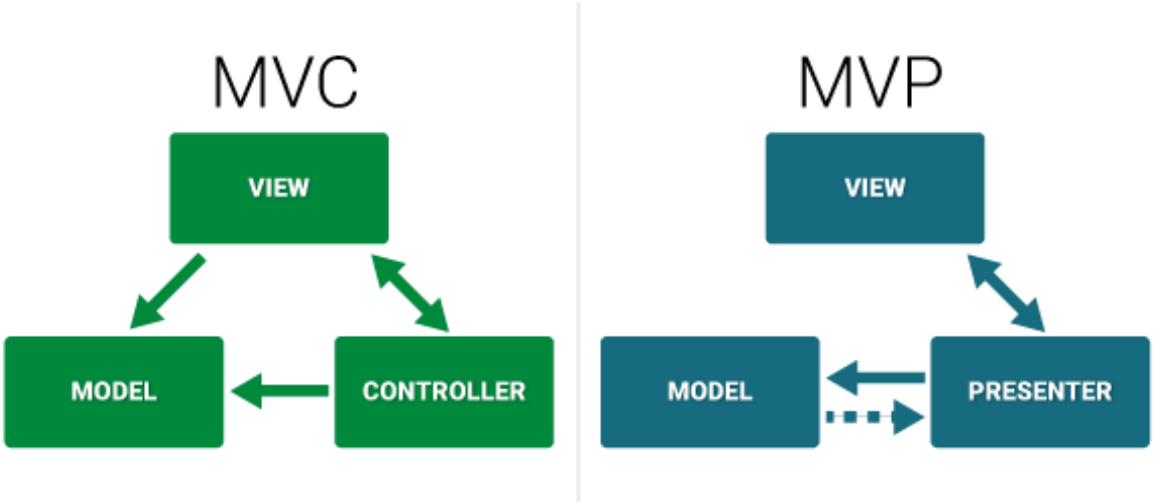
- inClientID
- inServiceID
- inServiceName
- inClientLongitude
- inClientLatitude
- inMessage

Κώδικας:

```
BEGIN
	INSERT INTO saveCalls (ClientID, ServiceID, ServiceName, ClientLongitude, ClientLatitude, Message, DateCreated)
  VALUES (inClientID, inServiceID, inServiceName, inClientLongitude, inClientLatitude, inMessage, DATE_FORMAT(NOW(),'%d/%m/%Y'));
END
```

Η εφαρμογή

Η εφαρμογή ‘EmergencyApp’ ακολουθεί το μοτίβο **MVP** (**M**odel**V**iew**P**resenter), το οποίο αποτελεί μετεξέλιξη του μοτίβου **MVC**(**M**odel**V**iew**C**ontroller). Παρακάτω έχουμε γραφική αναπαράσταση του μοτίβου MVP και MVC αντίστοιχα.



Model

Το μοντέλο είναι ο τομέας αυτός που σε κάθε περίπτωση περιλαμβάνει τις οντότητες που αλληλοεπιδρούν με το σύστημα και αναλαμβάνει την διαδικασία επικοινωνίας με εξωτερικούς από την εφαρμογή, τομείς του συστήματος, όπως βάσεις δεδομένων.

Βέβαια, ο τομέας των μοντέλων δεν περιλαμβάνει μόνο οντότητες όπως πελάτες, προϊόντα κλπ, αλλά και αντικείμενα που πραγματοποιούν συναλλαγές με βάση δεδομένων.

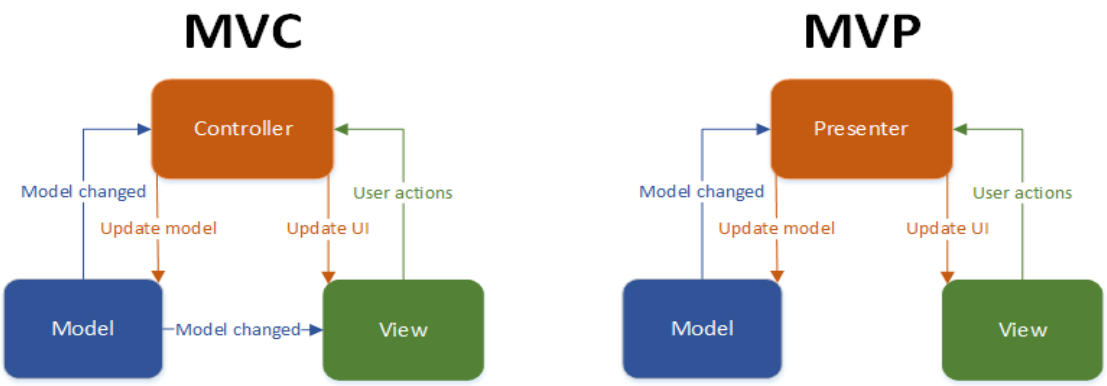
View

Ο τομέας των όψεων όταν μιλάμε για το μοτίβο MVC είναι πολύ πιο ‘παθητικός’ στη λειτουργία του απ’ ότι στο MVP και η διαφορά βρίσκεται στο εξής. Στο μοτίβο MVC, τα δεδομένα καλούνται να εισαχθούν από τον Controller στο View ενώ στο μοτίβο MVP δεν συμβαίνει έτσι ακριβώς. Τα δεδομένα εισάγονται στο View και οι λειτουργίες που θα γίνουν σύμφωνα με αυτά, εξαρτώνται από τον Presenter. Για να απλοποιήσουμε και να συγκεκριμενοποιήσουμε την εξήγηση θα αναφέρουμε ότι μπορούμε να φτιάξουμε ως View για μια λειτουργία ένα αρχείο XML όπου θα ορίζονται τα UIControls που χρησιμοποιούμε, και αυτό θα μπορούμε να το διαχειριστούμε φτιάχνοντας μια κλάση που – ουσιαστικά – ολοκληρώνει την λειτουργία του View όσον αφορά βασικές διαδικασίες όπως την ρύθμιση προβολής κάθε όψης.

Presenter

Εδώ έχουμε τη βασικότερη διαφοροποίηση μεταξύ του MVC και του MVP. Ο Controller του MVC θα πάρει τα δεδομένα χρήστη, θα τα επεξεργαστεί σε συνεργασία με τους υπόλοιπους τομείς του συστήματος και θα παράγει το αποτέλεσμα. Στο μοτίβο MVP ο Presenter παίρνει τα δεδομένα από την όψη του χρήστη, επικοινωνεί με το μοντέλο για να παράγει το αποτέλεσμα και το επιστρέφει στον

τομέα των όψεων. Βασική διαφορά είναι ότι οι τομείς στο μοτίβο MVC επικοινωνούν μεταξύ τους. Μπορεί, δηλαδή το μοντέλο να επικοινωνήσει με την όψη. Στο μοτίβο MVP αυτό δεν επιτρέπεται, η επικοινωνία του μοντέλου και του Presenter είναι αποκλειστική, όπως και η επικοινωνία της όψης με τον Presenter. Παρακάτω φαίνονται σχηματικά οι διαφορές των δύο αυτών μοτίβων.



Μοτίβο Σχεδίασης

Τα μοτίβα σχεδίασης με το πέρασμα των χρόνων αποδεικνύεται ότι είναι μία καλή λύση που έχει εφαρμοστεί με επιτυχία στην επίλυση ενός επαναλαμβανόμενου προβλήματος σχεδίασης συστημάτων λογισμικού. Τα πρότυπα σχεδίασης ορίζονται τόσο σε επίπεδο μακροσκοπικής σχεδίασης όσο και σε επίπεδο υλοποίησης, ενώ με τη χρήση τους ένας προγραμματιστής αντικαθιστά πρακτικώς μεγάλα τμήματα του κώδικα του με μαύρα κουτιά. Πρόκειται για αφαιρέσεις υψηλού επιπέδου που αποτελούν πλήρη υποσυστήματα, καταλλήλως ρυθμισμένα για την επίλυση συγκεκριμένων προβλημάτων σχεδίασης λογισμικού και έτοιμα για χρήση.

Κατά τα τέλη της δεκαετίας του 1970 ένας αρχιτέκτονας ονόματι Κρίστοφερ Αλεξάντερ επιχείρησε να βρει και να καταγράψει αποδεδειγμένα ποιοτικούς σχεδιασμούς στον τομέα των κατασκευών. Έτσι μελέτησε πολλές διαφορετικές κατασκευές που εξυπηρετούσαν τον ίδιο σκοπό και προσπάθησε να ανακαλύψει κοινά στοιχεία, τα οποία κατηγοριοποίησε σε *σχεδιαστικά πρότυπα*. Το 1987 η ιδέα της εύρεσης σχεδιαστικών προτύπων εφαρμόστηκε για πρώτη φορά στη μηχανική λογισμικού και μέχρι τα μέσα της δεκαετίας του '90 η εν λόγω έννοια είχε καθιερωθεί και εξαπλωθεί, προσανατολισμένη πλέον προς τον αντικειμενοστραφή προγραμματισμό. Ιδιαίτερα σημαντική εξέλιξη προς αυτή την κατεύθυνση υπήρξε η έκδοση του βιβλίου *Design Patterns: Elements of Reusable Object-Oriented Software* των Erich Gamma, Richard Helm, Ralph Johnson και John Vlissides, το 1994.

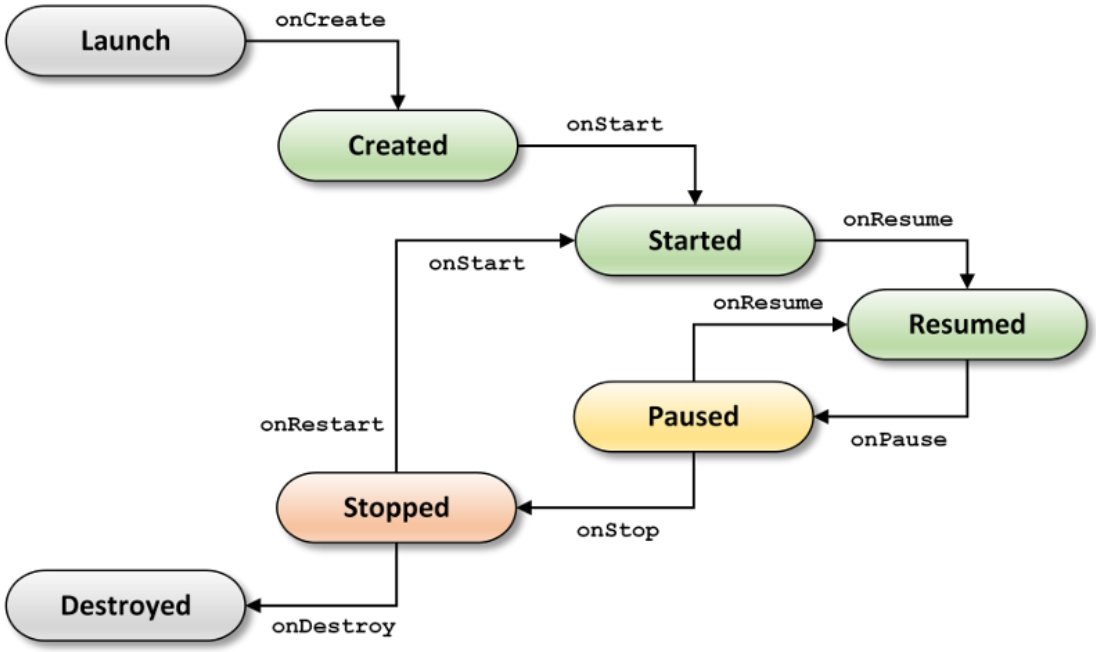
Σε γενικές γραμμές τα μοτίβα σχεδίασης μας βοηθάνε στα παρακάτω:

- Οργάνωση του κώδικα σε τομείς
- Ευκολία στη πραγματοποίηση testing
- Διευκόλυνση στην συντήρηση του κώδικα

Κώδικας της εφαρμογής

Σε αυτό το κεφάλαιο θα δώσουμε μια αναλυτική περιγραφή του κώδικα της εφαρμογής. Η εφαρμογή η οποία υλοποιήθηκε αποτελείται από τέσσερα Activities. Με τον όρο Activity εννοούμε κάθε οθόνη που εμφανίζεται στον χρήστη και μπορεί να αλληλεπιδρά με αυτήν.

Κρατάει κάθε φορά το περιεχόμενο της εφαρμογής καθώς μπορούμε να μεταβιβάσουμε δεδομένα από το ένα activity στο άλλο. Παρακάτω φαίνεται ο κύκλος ζωής κάθε activity.



Τα activities που υπάρχουν στην εφαρμογή αυτή είναι τα παρακάτω:

- MainActivity: Αυτό το activity περιέχει δύο textFields που δίνουν την δυνατότητα στον χρήστη να εισάγει το Username και το Password του. Ακόμα περιέχει δύο buttons το ένα για να κάνει είσοδο στην εφαρμογή και να ελεγχθούν τα στοιχεία που έδωσε ο χρήστης και το άλλο για να κάνει εγγραφή.
- SignUpActivity: Αυτό το activity είναι υπεύθυνο για την εγγραφή του χρήστη και ενεργοποιείται όταν ο χρήστης πατήσει το signUp κουμπί που βρίσκεται στο MainActivity. Για να γίνει εγγραφή ο χρήστης θα πρέπει να εισάγει τα παρακάτω στοιχεία.
  -Όνομα
  -Επίθετο
  -Χώρα
  -Πόλη
  -Διεύθυνση Κατοικίας
  -Αριθμός Τηλεφώνου
  -Ημερομηνία Γέννησης
  -Όνομα Χρήστη
  -Κωδικός Πρόσβασης

  Όταν ολοκληρώσει την εγγραφή πατώντας το κουμπί signUp και αφού η εφαρμογή ελέγξει αν τα στοιχεία που έδωσε είναι σωστά και καλύπτουν τις ανάγκες της εφαρμογής με μεθόδους που θα δούμε παρακάτω θα ενεργοποιήσει την αρχική οθόνη δηλαδή το MainActivity.
- EmergencyCallerActivity: Το activity αυτό ενεργοποιείται όταν ο χρήστης έχει κάνει είσοδο στην εφαρμογή αφού έχει προηγηθεί έλεγχος και εξακρίβωση στοιχείων. Στην οθόνη εμφανίζονται τρία buttons ένα για κάθε υπηρεσία.

- GoogleMapsActivity: Στο activity αυτό εμφανίζεται η τοποθεσία του χρήστη σε ένα googlemapfragment και η κοντινότερη υπηρεσία έκτακτης ανάγκης επιλέγει ο χρήστης καθώς και η διαδρομή που θα ακολουθήσει για την υπηρεσία.

Ανάλυση MVP Εφαρμογής

Model

IExecuteStoredProcedure.java

Το interface αυτό υλοποιείται από την κλάση Option και User . Οι μέθοδοι που περιέχει το Interface θα πρέπει να υλοποιηθούν από τις δύο κλάσεις διότι οι κλάσεις αυτές δεν έχουν αναφερθεί ως abstract.

```
package com.example.komabi.emergencyapp.Model.Interfaces;


import java.util.concurrent.ExecutionException;
import org.json.JSONException;
import java.util.List;

public interface IExecuteStoredProcedure<N extends IExecuteStoredProcedure<N>>
{
    List<N> getList() throws ExecutionException, InterruptedException, JSONException;
    N getListById(String inputModelID) throws ExecutionException, InterruptedException, JSONException;
}
```

Οι δύο μέθοδοι που περιέχει το interface αυτό και πρέπει να υλοποιηθούν είναι :

List<N>getList(): Η μέθοδος αυτή χρησιμοποιείται από τις κλάσεις αυτές για ζητήσει από την βάση δεδομένων συγκεκριμένα μοντέλα και να τα επιστρέψει στην εφαρμογή ως λίστα.

N getListById(String inputID): Με την μέθοδο αυτή επιστρέφονται στην εφαρμογή συγκεκριμενα μοντέλα ανάλογα με το ID που έχει πάρει ως όρισμα.

Κώδικας Υλοποίησης

```
public List<Option> getList()
{
    return (optionList);
}

@Override
public Option getListById(String inputModelId)
{
    for (Option eachOption : getList())
    {
        if (String.valueOf(eachOption.getOptionID()).equals(inputModelId)) return eachOption;
    }
    return null;
}
```

IAsyncModel

Αυτό το interface υλοποιείται από την κλάση AsyncModel η οποία επεκτείνει την κλάση AsyncTask η οποία χρησιμοποιείται σε περιπτώσεις που εκτελούμε μεγάλες λειτουργίες χωρίς να απασχολούμε το κύριο νήμα που χρησιμοποιείται για την διεπαφή χρήστη (User Inderface).

ILocationTracker

Το interface αυτό χρησιμοποιείται για να εντοπίσουμε την γεωγραφική τοποθεσία της συσκευής . Έχουμε κάνει την εισαγωγή της κλάσης android.location.Location όπου είναι μια κλάση δεδομένων που αντιπροσωπεύει μια γεωγραφική τοποθεσία. Η τοποθεσία που προκύπτει
Από τον LocationManager έχει αξιόπιστες τιμές. Στην διεπαφή αυτή έχουμε για υλοποίηση δύο μεθόδους για να ξεκινήσει ο εντοπισμός τοποθεσίας και μια μέθοδο που παίρνει ως όρισμα έναν Listener για τον επανυπολογισμό τις τοποθεσίας . Οι επόμενες δύο μέθοδοι επιστρέφουν boolean μεταβλητές αν έχουμε αποτέλεσμα από την ζητούμενη τοποθεσία .Στις τελευταίες δύο μεθόδους επιστρέφεται η τελευταία έγκυρη τοποθεσία.

Κώδικας Υλοποιήσης

```
package com.example.komabi.emergencyapp.Model.Interfaces;
import android.location.Location;

public interface IFindLocation
{
    interface LocationUpdateListener
    {
        void onUpdate(Location oldLoc, long oldTime, Location newLoc, long newTime);
    }

    void start();
    void start(LocationUpdateListener update);
    void stop();

    boolean hasLocation();
    boolean hasPossiblyStaleLocation();
    Location getLocation();
    Location getPossiblyStaleLocation();
}
```

Model Classes

DbAsyncBackground

Η κλάση αυτή επεκτείνει την AsyncModel που σημαίνει ότι θα εκτελεστεί μια ενέργεια ενώ το κύριο νήμα (main thread) τρέχει στην εφαρμογή. Είναι γνωστό ότι οι περισσότερες εφαρμογές θα πρέπει να

είναι συνδεδεμένες στο δυαδίκτιο για να κάνουν HTTP Request. Στην εφαρμογή που αναπτύξαμε αυτό είναι αναγκαίο διότι θα πρέπει να επικοινωνεί η εφαρμογή με την βάση δεδομένων που βρίσκεται στον web server.

Κώδικας Υλοποίησης

```
class DbAsyncBackground extends AsyncModel
{

    DbAsyncBackground()
    {

    }

    @Override
    protected String doInBackground(String... strings)
    {
        if(android.os.Debug.isDebuggerConnected()) android.os.Debug.waitForDebugger();

        try
        {
            String sqlProcedure = strings[0];
            HttpClient httpClient = new DefaultHttpClient(); // Dimiourgia HTTP client
            HttpPost httpPost = new HttpPost("https://sosmanagerapp.000webhostapp.com/action.php"); // Dimiourgia HTTP Post
            List<NameValuePair> data = new ArrayList<>();// Dimiourgia post parametrwn
            data.add(new BasicNameValuePair("action", sqlProcedure));// key and value pair
            UrlEncodedFormEntity encodedValuesToSend = new UrlEncodedFormEntity(data);// URL kwdikopoihsh twn Post Parametrwn
            httpPost.setEntity(encodedValuesToSend);
            HttpResponse httpResponse = httpClient.execute(httpPost);//Dhmiourgia HTTP Request
            InputStream inStream = httpResponse.getEntity().getContent();
            InputStreamReader inStreamReader = new InputStreamReader(inStream);
            BufferedReader buffReader = new BufferedReader(inStreamReader);

            StringBuilder strBuilder = new StringBuilder();
            String tmpBuilder;
            while ((tmpBuilder = buffReader.readLine()) != null)
            {
```

Σύμφωνα με τον παραπάνω κώδικα δημιουργήσαμε τον constructor της κλάσης και υλοποιήσαμε την μέθοδο doInBackground . Στην συνέχεια δημιουργήσαμε ένα HTTP request καθώς και έναν string builder για τα δεδομένα που εισέρχονται στην εφαρμογή.

ExecuteStoredProcedure

Αυτή η κλάση είναι υπεύθυνη για το χτίσιμο του ερωτήματος ή της αποθηκευμένης διαδικασίας μαζί με τα ορίσματά του και αποστέλλεται μέσω asynctask στη βάση δεδομένων. Η απάντηση που θα λάβουμε επιστρέφεται στην εφαρμογη ως JSONArray. Η κλάση αυτή υλοποιεί την IExecuteStoredProcedure.

Η συγκεκριμένη κλάση έχει ως ιδιότητες ένα μοντέλο ασύγχρονης κλήσης και μια μεταβλητή τύπου String που αποθηκεύει το αποτέλεσμα που επέστρεψε η βάση δεδομένων το οποίο αποτέλεσμα χρησιμοποείται στην εφαρμογή για να ελέγξουμε για παράδειγμα αν ένας χρήστης είναι εγγεγραμένος ώστε να του δώσουμε πρόσβαση στην εφαρμογή.

Κώδικας Υλοποίησης

```
abstract class ExecuteStoredProcedure
{

    private DbAsyncBackground caller;
    private String result;

    JSONArray ExecuteStoredProcedure(String execName) throws ExecutionException, InterruptedException, JSONException
    {
        String storedProcedureQuery = "CALL " + execName + "()";

        caller = new DbAsyncBackground();
        result = caller.execute(storedProcedureQuery).get();

        return (new JSONArray(result));
    }

    JSONArray ExecuteStoredProcedure(String execName, List<String> sqlParams) throws ExecutionException, InterruptedException, JSONException
    {
        String storedProcedureQuery = "CALL " + execName + "(";

        int i = 1;
        for (String param : sqlParams)
        {
            if (i != sqlParams.size()) storedProcedureQuery += "'" + param + "',";
            else storedProcedureQuery += "'" + param + "')";
            i++;
        }

        caller = new DbAsyncBackground();
        result = caller.execute(storedProcedureQuery).get();

        return (new JSONArray(result));
```

User

Η κλάση αυτή επεκτείνει το μοντέλο executeStoredProcedure και υλοποιεί τις μεθόδους της αντίστοιχης διεπαφής . Έχει ως ιδιότητες που είναι όλα τύπου String τα παρακάτω:

- Κωδικός Χρήστη
- Όνομα
- Επίθετο
- Χώρα Διαμονής
- Πόλη Διαμονής
- Διεύθυνση
- Ημερομηνία Γέννησης
- Αριθμός τηλεφώνου
- Όνομα Χρήστη
- Κωδικός πρόσβασης

Οι δύο τελευταίες ιδιότητες είναι σημαντικές και θα πρέπει να θυμάται ο χρήστης τι τιμές τοποθέτησε διότι η λανθασμένη εισαγωγή των στοιχείων θα
Αποτρέψει την είσοδο στην εφαρμογή.

Κώδικας Υλοποίησης

```
package com.example.komabi.emergencyapp.Model.Classes;

import android.util.Log;
import org.json.JSONArray;
import org.json.JSONException;
import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.ExecutionException;
import com.example.komabi.emergencyapp.Model.Interfaces.IExecuteStoredProcedure;


public class User extends ExecuteStoredProcedure implements IExecuteStoredProcedure<User>
{
    /*
        Current.Class Properties
    */
    private String ID;
    private String FirstName;
    private String LastName;
    private String Country;
    private String City;
    private String Address;
    private String BirthDate;
    private String PhoneNo;
    private String UserName;
    private String PassWord;
    private String DateCreated;
```

Στον παραπάνω κώδικα έχουμε την δήλωση όλων των μεταβλητών τύπου String που χρησιμοποιούνται στην κλάση User.
Παρατηρούμε ότι έχουμε τέσσερις constructors:

- Χωρίς ορίσματα
- Username + Password ως ορίσματα
- Όλα τα στοιχεία από την εγγραφή ως ορίσματα
- Όλα τα στοιχεία από την βάση ως όρισμα

Η περίπτωση του πρώτου constructor μας χρησιμεύει σε περιπτώσεις όπως να πάρουμε λίστα από όλους τους χρήστες ή να ζητήσουμε κάτι γενικά από τις λειτουργίες που έχει ο χρήστης. Η δεύτερη περίπτωση εξυπηρετεί ώστε να ελέγξουμε για έναν χρήστη δυναμικά αν είναι χρήστης της εφαρμογής. Η τρίτη και η τέταρτη περίπτωση χρησιμεύει για την εγγραφή του χρήστη στη βάση ή εξαγωγή λίστας χρηστών από αυτή, ώστε να δίνουμε άμεσα όλα τα στοιχεία ενός νέου χρήστη.

```
    public User(String inFN, String inLN, String inCo, String inCi, String inAd, String inBD, String
    {
        FirstName = inFN;
        LastName = inLN;
        Country = inCo;
        City = inCi;
        Address = inAd;
        BirthDate = inBD;
        PhoneNo = inPN;
        UserName = inUN;
        PassWord = inPW;
    }
public User(String inId, String inFN, String inLN, String inCo, String inCi, String inAddr, String
    {
        ID = inId;
        FirstName = inFN;
        LastName = inLN;
        Country = inCo;
        City = inCi;
        Address = inAddr;
        BirthDate = inBD;
        PhoneNo = inPN;
        UserName = inUN;
        PassWord = inPW;
        DateCreated = inDC;
    }
    public User()
    {

    }
public User(String inUN, String inPW)
    {
        UserName = inUN;
        PassWord = inPW;
    }
```

Μετακινήστε το δ

```
    public String isMember() throws ExecutionException, InterruptedException, JSONException
    {
        List<String> sqlParams = new ArrayList<>();
        sqlParams.add(getUserName());
        sqlParams.add(getPassWord());
        return (ExecuteStoredProcedure("signIn", sqlParams).getJSONObject(0).getString("Result"));
    }



    public void insertUser() throws ExecutionException, InterruptedException, JSONException
    {
        List<String> sqlParams = new ArrayList<>();
        sqlParams.add(getFirstName());
        sqlParams.add(getLastName());
        sqlParams.add(getCountry());
        sqlParams.add(getCity());
        sqlParams.add(getAddress());
        sqlParams.add(getBirthDate());
        sqlParams.add(getPhoneNo());
        sqlParams.add(getUserName());
        sqlParams.add(getPassWord());
        ExecuteStoredProcedure("signUp", sqlParams);
```

Η μέθοδος insertUser καλείται όταν γίνεται εγγραφή νέου χρήστη και έτσι τα δεδομένα που εισάγει ο χρήστης εγγράφονται στην βάση δεδομένων.

Η μέθοδος isMember δεν παίρνει ως όρισμα κάτι αλλά επιστρέφει το αποτέλεσμα από την βάση. Στην περίπτωση που ο χρήστης είναι εγγεγραμένος επιστρέφει σαν αποτέλεσμα τον κωδικό του αλλιώς επιστρέφει δεν βρέθηκε που σημαίνει ότι δεν μπορεί να κάνει είσοδο στην εφαρμογή.

```
    public String getAddress()
    {
        return Address;
    }
   public String getBirthDate()
    {
        return BirthDate;
    }
    public String getCity()
    {
        return City;
    }
    public String getCountry()
    {
        return Country;
    }
    public String getDateCreated()
    {
        return DateCreated;
    }
    public String getFirstName()
    {
        return FirstName;
    }
    public String getID()
    {
        return ID;
    }
    public String getLastName()
    {
        return LastName;
    }
    public String getPassWord()
    {
        return PassWord;
    }
```

Οι getters του μοντέλου User.

Option

Η κλάση αυτή δημιουργήθηκε με σκοπό να δίνει την επιλογή στον χρήστη να επιλέξει την υπηρεσία έκτακτης ανάγκης και να αποθηκεύσει την επιλογή αυτή σε μια λίστα. Επεκτείνει την ήδη υπάρχουσα κλάση ExecuteStoredProcedure και υλοποιεί τις μεθόδους της αντίστοιχης διεπαφής.

```
public class Option extends ExecuteStoredProcedure implements IExecuteStoredProcedure<Option>
{

    private int OptionID;
    private String OptionName;
    private String OptionGPName;
    private String OptionNo;
    private String OptionMailAddress;
    private static List<Option> optionList = new ArrayList<>();


    public Option()
    {

    }

    public Option(int inOptionID, String inOptionName, String inOptionGPName, String inOptionNo, String inOptionMailAddr)
    {
        OptionID = inOptionID;
        OptionName = inOptionName;
        OptionGPName = inOptionGPName;
        OptionNo = inOptionNo;
        OptionMailAddress = inOptionMailAddr;
    }

    public static void initializeOptionList()
    {
        optionList.add(new Option(R.id.fireBtn, "Fire", "fire_station", "199", "firemail@gmail.com"));
        optionList.add(new Option(R.id.policeBtn, "Police", "police", "100", "policemail@gmail.com"));
        optionList.add(new Option(R.id.ambulanceBtn, "Ambulance", "hospital", "166", "ambulancemail@gmail.com"));
    }
```

Το μοντέλο αυτό αποτελείται από δύο constructors ο οποιός ο δεύτερος παίρνει ως ορίσματα τα παρακάτω :

- Κωδικό υπηρεσίας
- Όνομα υπηρεσίας
- Όνομα υπηρεσίας που θα χρησιμοποιήσουμε για το google directions
- Αριθμό τηλεφώνου
- Διεύθυνση email

Σε αυτή την κλάση δεν επικοινωνούμε απευθείας με την βάση αλλά αποθηκεύονται οι μεταβλητές στατικά μέσα στο αντικείμενο.

```
public void saveOptionCall(String inCID, String inSID, String inSN, String inCLo, String inCLa, String inMsg) throws ExecutionException, Interrupt
{
    List<String> sqlParams = new ArrayList<>();
    sqlParams.add(inCID);
    sqlParams.add(inSID);
    sqlParams.add(inSN);
    sqlParams.add(inCLo);
    sqlParams.add(inCLa);
    sqlParams.add(inMsg);
    ExecuteStoredProcedure("save", sqlParams);
}
```

Παραπάνω έχουμε την μέθοδο saveOptionCall η οποία αποθηκεύει σε λίστα τις παραπάνω μεταβλητές και καλεί την μέθοδο ExecuteStoredProcedure η οποία με την σειρά της χτίζει το query για επικοινωνία με την βάση δεδομένων.

Για να αρχικοποιήσουμε τη λίστα με τις υπηρεσίες, καλούμε την παρακάτω στατική συνάρτηση

Option.initializeOptionList();

Η οποία θα δώσει στοιχεία στη λίστα με τις υπηρεσίες έκτακτης ανάγκης. Αυτό το κάνουμε διότι θέλουμε ως ID της κάθε υπηρεσίας να δώσουμε το ID του UIControl που αντιστοιχεί στην κάθε υπηρεσία από την αρχή της εφαρμογής, επειδή μας διευκολύνει σε διαδικασίες αναγνώρισης της κάθε υπηρεσίας μοναδικά σε όλη την έκταση της εφαρμογής.

LocParseJSON

Σκοπός του συγκεκριμένου μοντέλου είναι να μετατρέψει ένα JSONObject σε πίνακα ίδιο τύπο με αυτόν που λαμβάνουμε από την υπηρεσία της Directions της Google. Επεκτείνει το μοντέλο ExecuteStoredProcedure και δεν επεκτείνει κάποια διεπαφή . Επίσης, δίνει όλες τις κατάλληλες λειτουργίες που θα δείξουν την διαδρομή που θα ακολουθήσει ο χρήστης στον χάρτη από το σημείο που βρίσκεται στο σημείο που θέλει.

Κώδικας Υλοποιήσης

```
public class LocParceJSON extends ExecuteStoredProcedure {
    /**
     * Receives a JSONObject and returns a list of lists containing latitude and longitude
     */
    public List<List<HashMap<String, String>>> parse(JSONObject jObject) {
        List<List<HashMap<String, String>>> routes = new ArrayList<>();
        JSONArray jRoutes;
        JSONArray jLegs;
        JSONArray jSteps;

        try {
            jRoutes = jObject.getJSONArray("routes");

            /** Traversing all routes */
            for (int i = 0; i < jRoutes.length(); i++) {
                jLegs = ((JSONObject) jRoutes.get(i)).getJSONArray("legs");
                List path = new ArrayList<>();

                /** Traversing all legs */
                for (int j = 0; j < jLegs.length(); j++) {
                    jSteps = ((JSONObject) jLegs.get(j)).getJSONArray("steps");

                    /** Traversing all steps */
                    for (int k = 0; k < jSteps.length(); k++) {
                        String polyline;
                        polyline = (String) ((JSONObject) ((JSONObject) jSteps.get(k)).get("polyline"
                        List<LatLng> list = decodePoly(polyline);

                        /** Traversing all points */
                        for (int l = 0; l < list.size(); l++) {
                            HashMap<String, String> hm = new HashMap<>();
                            hm.put("lat", Double.toString(list.get(l).latitude));
                            hm.put("lng", Double.toString(list.get(l).longitude));
                            path.add(hm);
                        }
                    }
                    routes.add(path);
                }
```

```
    /* *
     * Method to decode polyline points
     * Courtesy : http://jeffreysambells.com/2010/05/27/decoding-polylines-from-google-maps-direction-api-with-java
     * */
    private List<LatLng> decodePoly(String encoded) {
        List<LatLng> poly = new ArrayList<>();
        int index = 0, len = encoded.length();
        int lat = 0, lng = 0;

        while (index < len) {
            int b, shift = 0, result = 0;
            do {
                b = encoded.charAt(index++) - 63;
                result |= (b & 0x1f) << shift;
                shift += 5;
            } while (b >= 0x20);
            int dlat = ((result & 1) != 0 ? ~(result >> 1) : (result >> 1));
            lat += dlat;

            shift = 0;
            result = 0;
            do {
                b = encoded.charAt(index++) - 63;
                result |= (b & 0x1f) << shift;
                shift += 5;
            } while (b >= 0x20);
            int dlng = ((result & 1) != 0 ? ~(result >> 1) : (result >> 1));
            lng += dlng;

            LatLng p = new LatLng((((double) lat / 1E5)), (((double) lng / 1E5)));
            poly.add(p);
        }

        return poly;
    }

    public static PolylineOptions getPoly(List<List<HashMap<String, String>>> inList) {
```

findLocation

Η κλάση αυτή επεκτείνει την ήδη υπάρχουσα executeStoredProcedure και διαθέτει μεθόδους που εκτελούνται σε διάφορα σημεία του κύκλου ζωής της εφαρμογής και αφορούν την εύρεση τοποθεσίας του χρήστη.

Κώδικας Υλοποίησης

```
package com.example.komabi.emergencyapp.Model.Classes;
import android.content.Context;

public class FindLocation extends ExecuteStoredProcedure
{
    private static LocationChanged Locationer;

    public static void initLocation(Context inContext)
    {
        Locationer = new LocationChanged(inContext);
        Locationer.start();
    }

    public static void stop()
    {
        Locationer.stop();
    }

    public static String getLongitude()
    {
        if (Locationer.hasLocation()) return (String.valueOf(Locationer.getLocation().getLongitude()));
        else return (String.valueOf(Locationer.getPossiblyStaleLocation().getLongitude()));
    }

    public static String getLatitude()
    {
        if (Locationer.hasLocation()) return (String.valueOf(Locationer.getLocation().getLatitude()));
        else return (String.valueOf(Locationer.getPossiblyStaleLocation().getLatitude()));
    }
}
```

LocAsyncBackgound

Αυτή η κλάση δεν υλοποιεί πραγματικά τίποτα εκτός της αντίστοιχης διεπαφής 'IAsyncModel' αλλά επεκτείνει την κλάση AsyncTask<String, String, String> της βιβλιοθήκης Android API, η οποία μας βοηθάει να πραγματοποιούμε ασύγχρονες κλήσεις σε συστήματα ή βάσεις δεδομένων.

Συγκεκριμένα, αυτή η κλάση μας βοηθάει να εκτελούμε διαδικασίες στο παρασκήνιο χωρίς να χρειάζεται να επεξεργαστούμε νήματα. Μας δίνει τρεις βασικές μεθόδους τις οποίες αναγκάζεται κάθε αντικείμενο που επεκτείνει αυτή την κλάση και δεν είναι αφηρημένο, να υλοποιήσει, οι οποίες καθορίζουν και τον βασικό κύκλο ζωής τους. Οι μέθοδοι είναι οι παρακάτω;

- onPreExecute(): Είναι μια μέθοδος στην οποία θα εισάγουμε ότι θέλουμε να συμβεί πριν εκτελεστεί η βασική διαδικασία που ορίζουμε στην μέθοδο doInBackground().
- doInBackground(String… params): Μέσα σε αυτή τη μέθοδο συμβαίνει η βασική διαδικασία που θέλουμε να εκτελέσουμε. Ανάλογαμε τις παραμέτρους που δίνουμε σαν ορίσματα, εκτελείται η διαδικασία και μας δίνεται το ανάλογο αποτέλεσμα.
- onPostExecute(String result):Εδώ επιστρέφεται το αποτέλεσμα από την διαδικασία που ορίστηκε στην μέθοδο doInBackground().

Κώδικας Υλοποίησης

```
public LocAsyncBackground(Location inStartLocation, Location inEndLocation)
{
    startLocation = inStartLocation;
    endLocation = inEndLocation;
}

@Override
protected String doInBackground(String... strings)
{
    if(android.os.Debug.isDebuggerConnected()) android.os.Debug.waitForDebugger();

    try
    {

        HttpClient httpClient = new DefaultHttpClient();
        HttpPost httpPost = new HttpPost("https://maps.googleapis.com/maps/api/directions/json?" +
                "origin=" + startLocation.getLatitude() + "," + startLocation.getLongitude() +
                "&destination=" + endLocation.getLatitude() + "," + endLocation.getLongitude() +
                "&sensor" + "false");
        HttpResponse httpResponse = httpClient.execute(httpPost);
        InputStream inStream = httpResponse.getEntity().getContent();
        InputStreamReader inStreamReader = new InputStreamReader(inStream);
        BufferedReader buffReader = new BufferedReader(inStreamReader);

        StringBuilder strBuilder = new StringBuilder();
        String tmpBuilder;

        while ((tmpBuilder = buffReader.readLine()) != null)
        {
            strBuilder.append(tmpBuilder);
        }

        return (strBuilder.toString());
    }
    catch (Exception x)
```

Παρατηρούμε ότι έχουμε έναν constructor ώστε να αρχικοποιούμε το μοντέλο αυτό, όπου στην περίπτωσή μας είναι η αρχική τοποθεσία και η τελική τοποθεσία. Το συγκεκριμένο μοντέλο αφορά την αποστολή αίτησης στην υπηρεσία Directions της Google ώστε να παραλάβουμε την ακριβή διαδρομή που πρέπει να κάνουμε για να πάμε από την αρχική τοποθεσία στην τελική τοποθεσία.

Υλοποιείται η μέθοδος doInBackground(); η οποία εκτελεί τις κατάλληλες λειτουργίες για να πραγματοποιηθεί το αίτημά μας. Επειδή το αποτέλεσμα θα γυρίσει ως τύπος String, δεν υλοποιούμε άλλες μεθόδους διότι θα πάρουμε το αποτέλεσμα με την παρακάτω μορφή:

```
String Result = (new LocAsyncBackground()).execute(startL, endL).get();
```

Παρατηρούμε ότι έχουμε φτιάξει το μοντέλο αυτό με τρόπο τέτοιο ώστε να το χρησιμοποιούμε πιο εύκολα ως αντικείμενο που θα κάνει την κλήση χωρίς να ασχολείται με λειτουργίες όπως διαχείριση του αποτελέσματος ή προετοιμασία αρχικών συνθηκών. Εδώ σημειώνεται ότι όλα τα μοντέλα ίδιου τύπου ακολουθούν το ίδιο μοτίβο. Τα υπόλοιπα μοντέλα που επεκτείνουν την κλάση ‘BaseAsyncCaller’ είναι τα παρακάτω:

DbAsyncBackground: Αφορά τη λειτουργία αποστολής αίτησης για μια αποθηκευμένη διαδικασία και την παραλαβή το αποτελέσματος, το οποίο επεξεργάζεται το αντικείμενο που κάλεσε τη λειτουργία αυτή.
PlacesAsyncBackground: Αφορά την λειτουργία αποστολής αιτήματος στην υπηρεσία Places της Google ώστε να λάβουμε βάσει κάποιων παραμέτρων τοποθεσίες που βρίσκονται κοντά στη τοποθεσία του χρήστη και είναι ενός συγκεκριμένου τύπου, της οποίας το αποτέλεσμα επεξεργάζεται το αντικείμενο που την κάλεσε.

LocationTracker -LocationChanged

Το μοντέλο locationTracker υλοποιεί τις μεθόδους της διεπαφής iFindLocation και αυτό κρίνεται αναγκαστικό διότι δεν έχουμε δηλώσει την κλάση ως abstract . Ανάλογα με την περίπτωση του παρόχου δηλαδή αν ο πάροχος είναι το GPS ή το διαδίκτυο , τα οποία ελέγχονται με δίαφορες συνθήκες μέσα στις μεθόδους , εκτελούνται και οι κατάλληλες ενέργειες. Το μοντέλο LocationChanged χρησιμοποιεί το παραπάνω μοντέλο για τον έλεγχο του παρόχου.

Κώδικας Υλοποίησης

```
public void start()
{
    if (isRunning) return;

    isRunning = true;
    if (ActivityCompat.checkSelfPermission(currentContext, Manifest.permission.ACCESS_FINE_LOCATION) != P
    {
        Toast.makeText(currentContext, "need to enable GPS Location", Toast.LENGTH_LONG).show();
        return;
    }
    locMgr.requestLocationUpdates(provider, MIN_UPDATE_TIME, MIN_UPDATE_DISTANCE, this);
    lastLocation = null;
    lastTime = 0;
}

@Override
public void start(LocationUpdateListener update)
{
    start();
    locUpdListener = update;
}

@Override
public void stop()
{
    if(isRunning)
    {
        if (ActivityCompat.checkSelfPermission(currentContext, Manifest.permission.ACCESS_FINE_LOCATION)
        {
            return;
        }
        locMgr.removeUpdates(this);
        isRunning = false;
        locUpdListener = null;
    }
}
```

```
@Override
public boolean hasLocation()
{
    return ((lastLocation != null) && (System.currentTimeMillis() - lastTime <= 5 * MIN_UPDATE_TIME));
}

@Override
public boolean hasPossiblyStaleLocation()
{
    if(lastLocation != null) return true;
    if (ActivityCompat.checkSelfPermission(currentContext, Manifest.permission.ACCESS_FINE_LOCATION) !=
    {

    }
    return (locMgr.getLastKnownLocation(provider)!= null);
}

@Override
public Location getLocation()
{
    if(lastLocation == null) return null;
    if(System.currentTimeMillis() - lastTime > 5 * MIN_UPDATE_TIME) return null; //stale
    return lastLocation;
}

@Override
public Location getPossiblyStaleLocation()
{
    if(lastLocation != null) return lastLocation;
    if (ActivityCompat.checkSelfPermission(currentContext, Manifest.permission.ACCESS_FINE_LOCATION) !=
    {

    }
    return locMgr.getLastKnownLocation(provider);
}
```

Η κύρια διαφορά των δύο μοντέλων είναι η onUpdate μέθοδος που υλοποιείται στην onChanged κλάση και κάνει επανυπολογισμό της τοποθεσίας αν δεν υπάρχει τοποθεσία ή ο πάροχος είναι ίδιος ή αν ο πάροχος είναι πιο αξιόπιστος στην τοποθεσία.

```
@Override
public void onUpdate(Location oldLoc, long oldTime, Location newLoc, long newTime)
{
    boolean update = false;
    //kanoume update mono an den uparxei topothesia h o paroxos einai idios h o paroxos einai pio aksiopistos stin topothesia
    if(lastLoc == null) update = true;
    else if(lastLoc.getProvider().equals(newLoc.getProvider())) update = true;
    else if(newLoc.getProvider().equals(LocationManager.GPS_PROVIDER)) update = true;
    else if (newTime - lastTime > 5 * 60 * 1000) update = true;

    if(update)
    {
        if(listener != null) listener.onUpdate(lastLoc, lastTime, newLoc, newTime);
        lastLoc = newLoc;
        lastTime = newTime;
    }
}
}
```

Η μέθοδος αυτή παίρνει ως ορίσματα την παλιά τοποθεσία, την παλιά ώρα υπολογισμού τοποθεσίας την καινούρια τοποθεσία και την καινούρια ώρα υπολογισμού τοποθεσιας.

View

Στο παράρτημα αυτό θα παρουσιάσουμε τις κλάσεις από τις οποίες αποτελείται η όψη της εφαρμογής. Η σημασία των όψεων είναι καθοριστική διότι με αυτές αλληλεπιδρά ο χρήστης και έτσι η σχεδίαση τους καθιστάται αναγκαία έτσι ώστε κάθε χρήστης να μπορέσει να αλληλεπιδράσει με την εφαρμογή είτε αυτός είναι εξοικειωμένος με την τεχνολογία είτε όχι.

IActivity

Πρόκειται για την μια και μοναδική διεπαφή η οποία ορίζει όλες τις βασικές λειτουργίες που θα εκτελέσει κάθε activity .

```
package com.example.komabi.emergencyapp.View.Interfaces;

import android.support.v4.app.ActivityCompat;

public interface IActivity extends ActivityCompat.OnRequestPermissionsResultCallback
{
    void initComps();
}
```

Η διεπαφή αυτή επιτρέπει στο activity να διαχειριστεί τυχόν άδειες που μπορεί να χρειαστεί η εφαρμογή κατά τη διάρκεια της λειτουργίας της. Διαθέτει μια μέθοδο η οποία πρέπει να υλοποιηθεί από όλα τα activities η οποία είναι η initComps() και αφορά συνήθως λειτουργίες αρχικοποίησης των αντικειμένων που διαθέτει κάθε όψη, όπως UIControls.

Η διεπαφή αυτή υλοποιείται από δύο κλάσεις οι οποίες είναι οι παρακάτω.

- StartActivity
- BaseMapActivity

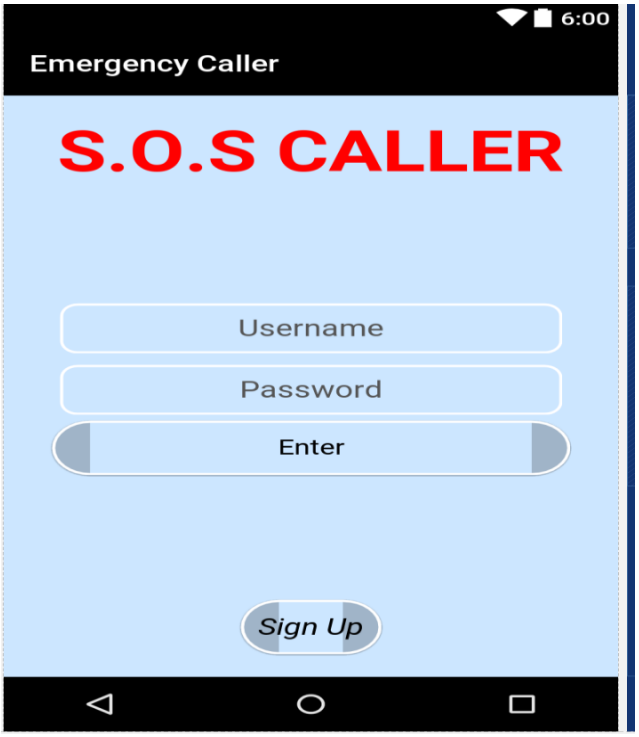
StartActivity

Η κλάση αυτή απευθύνεται σε λειτουργίες ενός τυπικού activity αφού είναι υπεύθυνο για τις βασικές λειτουργίες αυτού. Συγκεκριμένα, ανταποκρίνεται στους βασικούς κύκλους ζωής ενός activity καθώς φαίνεται στον παρακάτω κώδικα.

```
public abstract class StartActivity extends Activity implements IActivity
{


    @Override

    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);




        initComps();
    }

    @Override
    protected void onStart()
    {
        super.onStart();
        FindLocation.initLocation(getApplicationContext());
    }

    @Override
    protected void onStop()
    {
        super.onStop();
        FindLocation.stop();
    }
}
```

Η κλάση αυτή επιλέχθηκε να είναι abstract ώστε να μην είναι απαραίτητο να υλοποιούνται οι μέθοδοι του 'IActivity' άμεσα από αυτήν αλλά από τις κλάσεις που κληρονομούν αυτή την κλάση ('StartActivity'). Σκοπός αυτού είναι να εκτελούμε σε κάθε σημείο του κύκλου ζωής ενός activity τις λειτουργίες που θέλουμε καθολικά σε όλα και να αφήνουμε μετά στο κάθε activity που κληρονομεί αυτή την κλάση να εκτελεί τις ιδιαίτερες λειτουργίες που διαθέτει.

MainActivity



Στιγμιότυπο Οθόνης Main Activity

Στο παραπάνω στιγμιότυπο απεικονίζεται η όψη η οποία κάνει την εμφάνιση της όταν 'τρέξει' η εφαρμογή. Η όψη αυτή αποτελείται από ένα textView το οποίο είναι ο τίτλος της εφαρμογής (S.O.S CALLER). Αποτελείται ακόμα από δύο textFields στα οποία εισάγει ο χρήστης το όνομα χρήστη και τον κωδικό πρόσβασης. Τέλος αποτελείται από δύο buttons τα οποία το ένα είναι για την είσοδο του χρήστη και το άλλο για την εγγραφή με το οποίο πατώντας το θα βρεθεί σε άλλο activity το signUpActivity. Η όψη αυτή έχει σαν κύριο layout ένα relativeLayout και κάθε στοιχείο μέσα στο layout τοποθετείται σε TableRow.

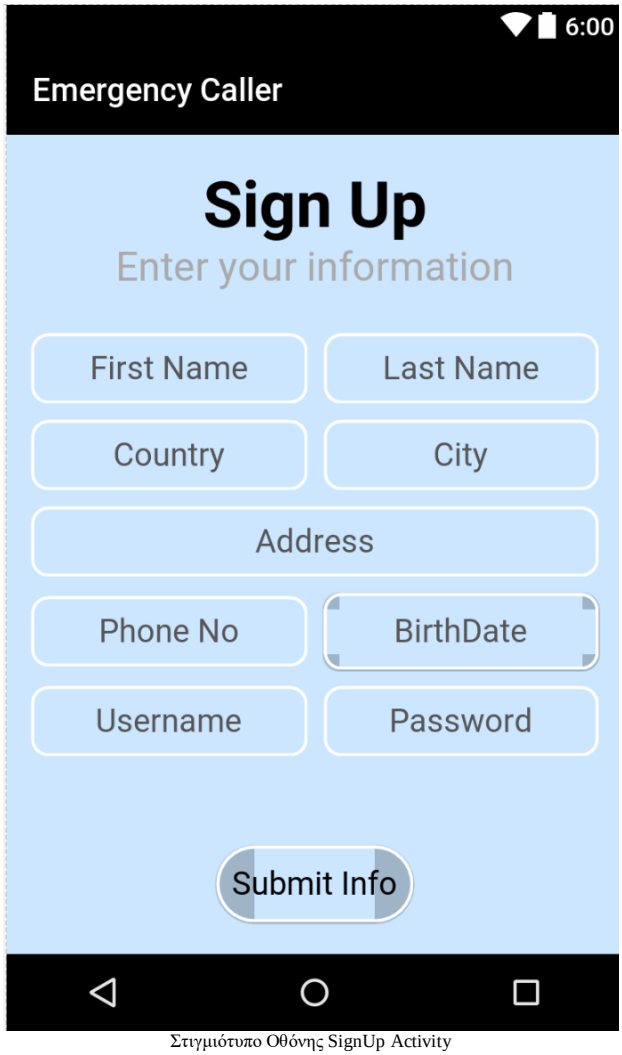
MainActivity.xml

```
<TableLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:padding="20dp">

    <TableRow>
        <TextView
            android:id="@+id/sosTitle"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:textAlignment="center"
            android:fontFamily="sans-serif"
            android:textStyle="bold"
            android:textColor="#F00"
            android:textSize="50sp"
            android:text="@string/sosTitleStr"
            />
    </TableRow>

    <TableRow>

        <TextView
            android:id="@+id/managerTitle"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:textAlignment="center"
            android:fontFamily="monospace"
            android:textColor="#777"
            android:textSize="30sp"
            android:text="@string/managerTitleStr"
            />
    </TableRow>

</TableLayout>
```

SignUpActivity

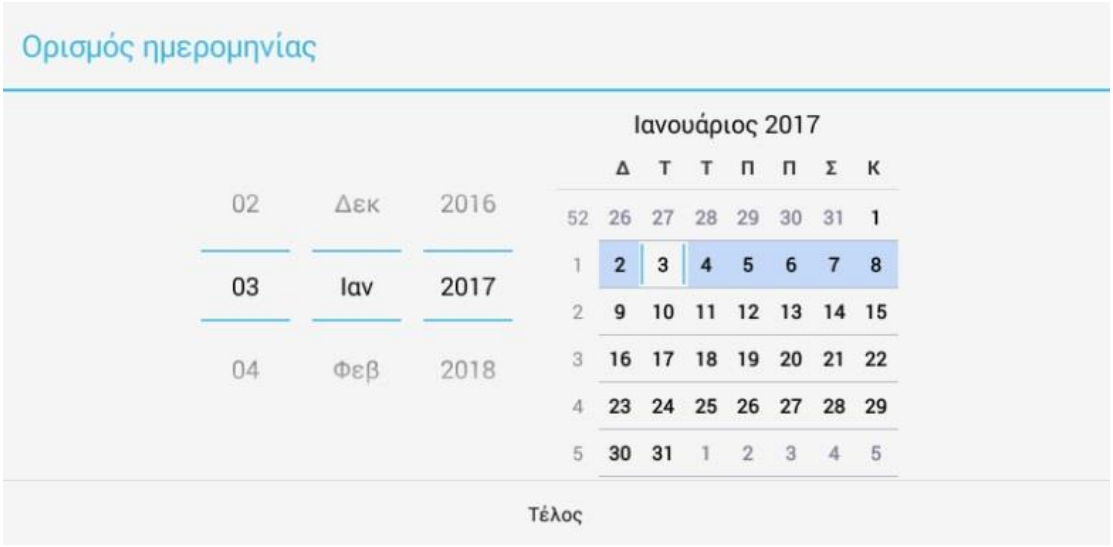
Η συγκεκριμένη όψη αντιπροσωπεύει την όψη με την οποία θα αλληλεπιδράσει ο χρήστης για να πραγματοποιήσει την εγγραφή του στην εφαρμογή. Ουσιαστικά κάθε καινούριος χρήστης θα περάσει αναγκαστικά από αυτήν την όψη διότι χωρίς να κάνει εγγραφή δεν θα μπορέσει να πραγματοποιήσει είσοδο στην εφαρμογή. Η όψη αυτή ακολουθεί το ίδιο μοτίβο με την προηγούμενη όψη δηλαδή έχει την ίδια δομή ως προς το layout.

Περιέχει δύο textViews και εννέα textFields καθώς και ένα button.



Στιγμιότυπο Οθόνης SignUp Activity

Να τονίσουμε ότι στην περίπτωση του κωδικού ο κωδικός που θα πρέπει να εισάγει ο χρήστης αντιμετωπίζει κάποιους περιορισμούς καθώς θα πρέπει να είναι μεγαλύτερος από οχτώ ψηφία. Το ίδιο συμβαίνει και με το όνομα χρήστη καθώς ελέγχεται αν το όνομα που τοποθέτησε υπάρχει στην βάση και ανάλογα εμφανίζει αντίστοιχο μήνυμα ώστε να εισάγει άλλο.



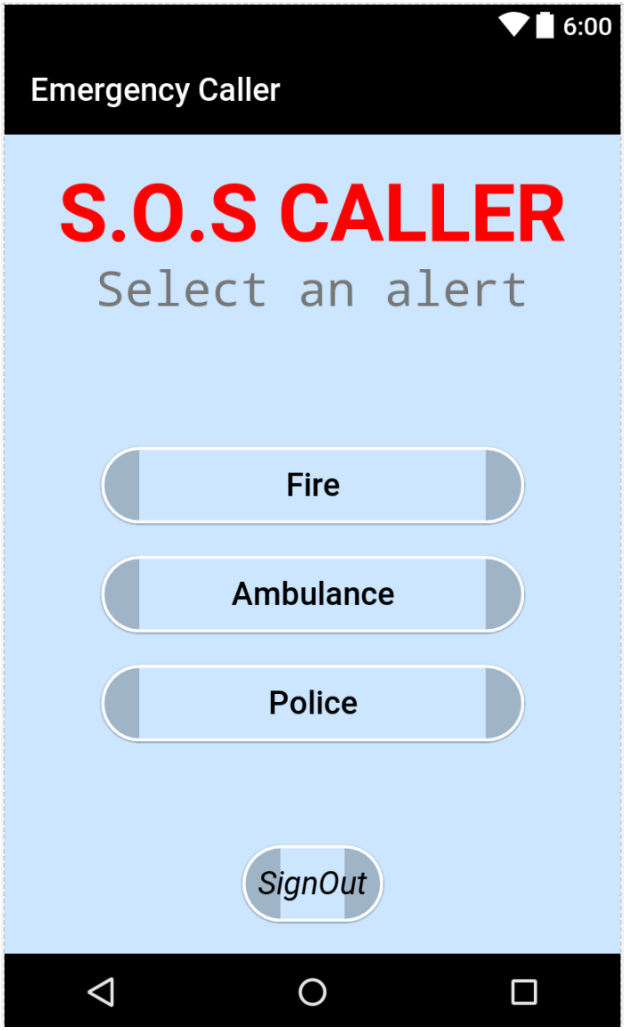
Στιγμιότυπο Διαλόγου Επιλογής Ημερομηνίας

Αφού συμπληρώσει όλα τα στοιχεία του ο χρήστης και γίνει η εγγραφή επιτυχώς, μεταφέρεται στην αρχική οθόνη της εφαρμογής που είναι το 'MainActivity'. Σε περίπτωση μη εισαγωγής κάποιου στοιχείου, θα ειδοποιηθεί με ανάλογο μήνυμα από την εφαρμογή.

SignUpActivity.xml

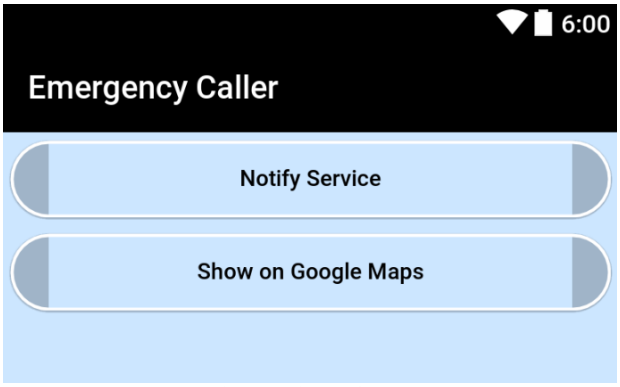
```
            <Button
                android:id="@+id/inputBirthDate"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:paddingTop="10dp"
                android:paddingBottom="10dp"
                android:paddingLeft="20dp"
                android:paddingRight="20dp"
                android:layout_margin="5dp"
                android:textAlignment="center"
                android:textSize="20sp"
                android:fontFamily="sans-serif"
                android:textColor="#000"
                android:textColorHint="#555"
                android:hint="@string/inputBDateHintStr"
                android:background="@drawable/styling_text_input_normal"
                />
        </TableRow>
<TableRow>
            <EditText
                android:id="@+id/inputUserName"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:paddingTop="10dp"
                android:paddingBottom="10dp"
                android:paddingLeft="20dp"
                android:paddingRight="20dp"
                android:layout_margin="5dp"
                android:textAlignment="center"
                android:textSize="20sp"
                android:fontFamily="sans-serif"
                android:inputType="text"
                android:textColor="#000"
                android:textColorHint="#555"
                android:hint="@string/inputUserNameHintStr"
                android:background="@drawable/styling_text_input_normal"
```

SosCallerActivity



Στιγμιότυπο Οθόνης SOS Caller Activity

Η όψη αυτή είναι ίσως η σημαντικότερη διότι αντιπροσωπεύει την λειτουργία της εφαρμογής. Συγκεκριμένα εμπεριέχει δύο textViews και τέσσερα buttons , τα οποία τα τρία είναι για την κάθε υπηρεσία ξεχωριστά και το τέταρτο είναι για την έξοδο από την εφαρμογή. Επιλέγωντας μια από τις παραπάνω υπηρεσίες ο χρήστης θα του εμφανιστεί η επόμενη όψη όπως θα δούμε παρακάτω.



Στιγμιότυπο Διαλόγου Επιλογής Εξυπηρέτησης Χρήστη

Ο χρήστης έχει δύο προφανείς επιλογές:

- Ειδοποίηση της υπηρεσίας
- Επίδειξη σημείου σε Google Maps

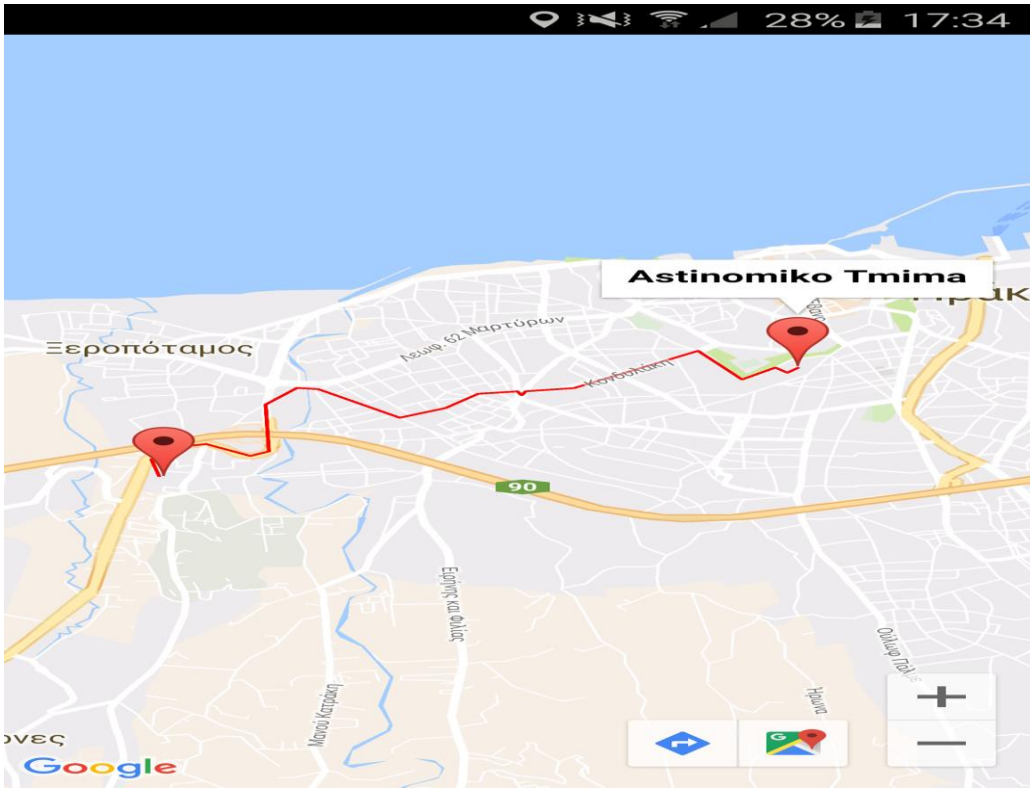
Με την πρώτη επιλογή, η εφαρμογή θα στείλει μήνυμα στην αντίστοιχη υπηρεσία με την παρακάτω μορφή:

«
Είμαι ο [Όνομα Χρήστη] [Επίθετο Χρήστη]
Η τοποθεσία μου είναι:
Γεωγραφικό Πλάτος: [Γ. Πλάτος Χρήστη]
Γεωγραφικό Μήκος: [Γ. Μήκος Χρήστη]
»

Να υπενθυμίσουμε ότι η αποστολή μηνύματος προς τις συγκεκριμένες υπηρεσίες είναι δωρεάν .

Παρατηρούμε ότι δίνονται τα απαραίτητα στοιχεία στην αντίστοιχη υπηρεσία ώστε να υπάρχει τρόπος να επικοινωνήσουν σε περίπτωση που χρειαστεί περεταίρω επικοινωνία. Ο αριθμός τηλεφώνου φαίνεται στην υπηρεσία αφού το μήνυμα χρησιμοποιεί πόρους του συστήματος του χρήστη για την αποστολή του. Κατά την αποστολή, ο χρήστης θα ειδοποιηθεί για την κατάσταση αποστολή και την κατάσταση παράδοσης με ανάλογο μήνυμα.

Με την δεύτερη επιλογή ο χρήστης θα μεταφερθεί σε νέα όψη και θα εμφανιστεί η τοποθεσία του με την βοήθεια της GoogleMaps καθώς και η τοποθεσία του προορισμού δηλαδή της υπηρεσίας που έχει επιλέξει να ειδοποιείσει.



Dialog_soscall.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#cce6ff">

    <Button
        android:id="@+id/notifyServiceBtn"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="5dp"
        android:textAllCaps="false"
        android:textColor="#000"
        android:text="@string/notifyServiceStr"
        android:background="@drawable/styling_button_normal"
        />

    <Button
        android:id="@+id/showGMapBtn"
        android:layout_below="@id/notifyServiceBtn"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="5dp"
        android:textAllCaps="false"
        android:textColor="#000"
        android:text="@string/showGMapStr"
        android:background="@drawable/styling_button_normal"
        />

</RelativeLayout>
```

Presenter

Στο κομμάτι αυτό θα παρουσιάσουμε τον παρουσιαστή (Presenter) ο οποίος συνδέεται με κάθε όψη έτσι κάθε όψη θα έχει τον δικό του παρουσιαστή για να κάνει τις συναλλαγές του. Για την εφαρμογή μας στο κομμάτι του παρουσιαστή θα έχουμε τις παρακάτω κλάσεις οι οποίες καλύπτουν στο έπακρο τις λειτουργίες της εφαρμογής.

- AlertDialogg
- CheckLoginSignUp
- ClickEvent
- EmergencyCaller
- GoogleMaps
- SignUpSubmit
- IAlertDialog

Παρακάτω θα δώσουμε μια αναλυτική περιγραφή των παραπάνω κλάσεων.

IAlertDialog

Η διεπαφή αυτή είναι η μια και μοναδική στον τομέα των παρουσιαστών και ο σκοπός της είναι κάθε παρουσιαστής να υιοθετήσει και να επεκτείνει την παραπάνω διεπαφή ώστε να είναι υλοποιήσει τις μεθόδους της. Οι δύο πρώτες μέθοδοι αφορούν την εμφάνιση ανάλογων μηνυμάτων στους χρήστες η alertdialogg μέθοδος εμφανίζει ένα alertdialog και η isNetworkAvailable επιστρέφει αληθές ή ψευδές αν ο χρήστης είναι συνδεμένος σε δίκτυο ή όχι.

Κώδικας Υλοποίησης

```
package com.example.komabi.emergencyapp.Presenter.Interfaces;

import android.net.ConnectivityManager;
import android.content.Context;

public interface IAlertDialog
{
    void printMessage(String inMsg, Context inContext);
    void printException(Exception x, Context inContext);
    void alertDialogg(String diagTitle, String diagMsg, Context inContext);
    boolean isNetworkAvailable(ConnectivityManager inConnMgr);
}
```

AlertDialogg

Η κλάση αυτή επεκτείνει την διεπαφή IAlertDialog και σκοπός της είναι να υλοποιήσει όλες τις μεθόδους της διεπαφής.

Κώδικας Υλοποίησης

```
abstract class AlertDialogg implements IAlertDialog
{

    @Override
    public void printMessage(String inMsg, Context inContext)
    {
        Toast.makeText(inContext, inMsg, Toast.LENGTH_LONG).show();
    }

    @Override
    public void printException(Exception x, Context inContext)
    {
        Toast.makeText(inContext, "EXCEPTION '" + x.getClass().getName() + "'\nWITH STACK TRACE '" + x.getStackTr
    }

    @Override
    public void alertDialogg(String diagTitle, String diagMsg, Context inContext)
    {
        AlertDialog.Builder warningDialogBuilder;

        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP) warningDialogBuilder = new AlertDialog.Builder
        else warningDialogBuilder = new AlertDialog.Builder(inContext);
        warningDialogBuilder.setPositiveButton("Ok", new DialogInterface.OnClickListener()
        {
            @Override
            public void onClick(DialogInterface dialog, int id)
            {
                dialog.cancel();
            }
        })
                .setTitle(diagTitle)
                .setMessage(diagMsg)
                .setCancelable(false);
        AlertDialog warningDialog = warningDialogBuilder.create();
        warningDialog.show();
    }
```

ClickEvent

Η κλάση αυτή επεκτείνει την AlertDialogg και υλοποιεί την υλοποιεί τη μέθοδο onClick(View v),που ανήκει στη διεπαφή View.OnClickListener

Στην κλάση αυτή έχουμε τοποθετήσει μια switch η οποία ελέγχει τις περιπτώσεις των κλικ των κουμπιών της εφαρμογής. Κάθε κουμπί έχει την δικιά του περίπτωση (case) η οποία εκτελείται από την αναγνώριση του button που πατήθηκε . Η αναγνώριση του button γίνεται από τον listener που χρησιμοποιεί κάθε κουμπί μέσω του ID του . Στα περισσότερα από τα κουμπιά που έχουμε τοποθετήσει στην εφαρμογή η κοινή διαδικασία είναι να ανοίγουν κάποιο άλλο activity διαφορετικό από αυτό που βρίσκεται ο χρήστης.

Κώδικας Υλοποίησης

```
public class ClickEvent extends AlertDialogg implements View.OnClickListener
{

    private Activity currentAct;


    public ClickEvent(Activity inAct)
    {
        currentAct = inAct;
    }

    public SharedPreferences sp;

    @Override
    public void onClick(View v)
    {

        switch (v.getId())
        {
            case (R.id.submitBtn):
            {
                MainActivity currAct = ((MainActivity) currentAct);
                (new CheckLoginSignUp(currAct)).checkLoginCredentials(currAct.getInputUserNameET().getText().toStr
                        currAct.getInputPassWordET().getText().toString());

                break;
            }
            case (R.id.signUpBtn):
            {
                MainActivity currAct = ((MainActivity) currentAct);
                (new CheckLoginSignUp(currAct)).signUpUser();

                break;
            }
            case (R.id.signOutBtn):
            {

                SharedPreferences sp= currentAct.getSharedPreferences("login",MODE_PRIVATE);
```

```
            SharedPreferences.Editor e=sp.edit();
            e.clear();
            e.commit();

            Intent Signout = new Intent(currentAct,MainActivity.class);
            currentAct.startActivity(Signout);

            break;

        }
        case (R.id.inputBirthDate):
        {
            SignUpActivity currAct = ((SignUpActivity) currentAct);
            currAct.showDialog(800);
            currAct.getInBirthDateBTN().setText(new StringBuilder().append(Calendar.getInstance().get(Calendar
                    .append("/")
                    .append(Calendar.getInstance().get(Calendar.MONTH) + 1)
                    .append("/")
                    .append(Calendar.getInstance().get(Calendar.YEAR)));

            break;
        }
        case (R.id.signUpSubmitBtn):
        {
            SignUpActivity currAct = ((SignUpActivity) currentAct);
            (new SignUpSubmit(currAct)).submitUsersInformation();

            break;
        }
        case (R.id.notifyServiceBtn):
        {
            EmergencyCallerActivity currAct = ((EmergencyCallerActivity) currentAct);
            (new EmergencyCaller(currAct)).notifyOption();

            break;
        }
```

CheckLoginSignUp

Ο παρουσιαστής αυτός είναι υπεύθυνος για τις λειτουργίες που εκτελούνται στην όψη mainActivity στην οποία να υπενθυμίσουμε έχουμε τοποθετήσει δυο κουμπιά ένα για είσοδο και ένα για εγγραφή. Έτσι στον παρουσιαστή που αντιπροσωπεύει την συγκεκριμένη όψη θα έχουμε δύο κύριες μεθόδους ανάλογα μια για κάθε κουμπί. Οι μέθοδοι είναι οι παρακάτω :

Void checkLoginCredentials(String uN, String pW): Η μέθοδος αυτή παίρνει τα στοιχεία που εισήγαγε ο χρήστης στην κύρια όψη της εφαρμογής και αναλαμβάνει να χρησιμοποιήσει τις μεθόδους που υλοποιούνται στο μοντέλο του χρήστη.

Void signUpClient(): Η μέθοδος αυτή εκτελείται όταν ο χρήστης επιλέξει να κάνει εγγραφή και είναι υπεύθυνη για την μεταφορά του χρήστη στην οθόνη για την εγγραφή του χρήστη με την οποία είναι συνδεδεμένος ο presenter της οθόνης αυτής.

Κώδικας Υλοποίησης

```
class CheckLoginSignUp extends AlertDialogg
{


   private MainActivity currentAct;


   CheckLoginSignUp(MainActivity inAct)
   {
       currentAct = inAct;
   }


   void checkLoginCredentials(String inUserName, String inPassWord)
   {
       try
       {
           if (isNetworkAvailable((ConnectivityManager) currentAct.getSystemService(Context.CONNECTIVITY_SERVICE)))
           {

               if ((currentAct.getInputUserNameET().getText().toString().equals("")) || (currentAct.getInputUserNameET().getText().toString().equals("
               {
                   printMessage(currentAct.getResources().getString(R.string.emptyInputsStr), currentAct.getApplicationContext());
               }
               else
               {
                   String CurrentUserID = (new User(inUserName, inPassWord)).isMember();

                   currentAct.getInputUserNameET().setText(null);
                   currentAct.getInputPassWordET().setText(null);

                   if (CurrentUserID.equals("ClientNotFound")) printMessage(currentAct.getResources().getString(R.string.clientAuthFailedStr), current
                   else
                       else
                       {
                           //check username and password are correct and then add them to SharedPreferences

                               SharedPreferences.Editor e=currentAct.sp.edit();
                               e.putString("username",inUserName);
                               e.putString("password",inPassWord);
                               e.commit();




                               Intent sosCallerIntent = new Intent(currentAct.getApplicationContext(), EmergencyCallerActivity.class);
                           Bundle intentBundle = new Bundle();
                           intentBundle.putString("CID", CurrentUserID);
                           sosCallerIntent.putExtras(intentBundle);

                           currentAct.startActivity(sosCallerIntent);
                       }
                   }
               }
               else alertDialogg(
                       currentAct.getResources().getString(R.string.noInternetStr),
                       currentAct.getResources().getString(R.string.noInternetTitleStr),
                       currentAct.getApplicationContext());
           }
           catch (Exception x)
           {
               printException(x, currentAct.getApplicationContext());
           }
       }
```

```
    void signUpUser()
    {
        try
        {
            if (isNetworkAvailable((ConnectivityManager) currentAct.getSystemService(Context.CONNECTIVITY_SERVICE)))
            {
                currentAct.startActivity(new Intent(currentAct.getApplicationContext(), SignUpActivity.class));
            }
            else alertDialogg(
                    currentAct.getResources().getString(R.string.noInternetStr),
                    currentAct.getResources().getString(R.string.noInternetTitleStr),
                    currentAct.getApplicationContext());
        }
        catch (Exception x)
        {
            printException(x, currentAct.getApplicationContext());
        }
    }
}
```

SignUpSubmit

Ο παρουσιαστής αυτός είναι υπεύθυνος να πάρει τα δεδομένα που εισήγαγε ο χρήστης να ελέγξει αν είναι κάποιο στοιχείο κενό ή αν ο κωδικός είναι μικρότερος από 8 αλφαριθμητικά στοιχεία και να εκτελέσει την μέθοδο insertUser() που βρίσκεται στο μοντέλο Client μεταβιβάζοντας όλα τα στοιχεία που έδωσε ο χρήστης.

Κώδικας Υλοποιησης

```
if ((inFirstName.equals("")) || (inLastName.equals("")) || (inCountry.equals("")) || (inCity.equals("")) ||
        (inAddress.equals("")) || (inBirthDate.equals("")) || (inPhoneNo.equals("")) || (inUserName.equals("")) ||
        (inPassWord.equals("")))
{
    printMessage(currentAct.getResources().getString(R.string.emptyInputsStr), currentAct.getApplicationContext());
}
else
{
    if (inPassWord.length() < 8)
    {
        currentAct.getInPassWordET().setText(null);
        printMessage(currentAct.getResources().getString(R.string.passEightCharsStr), currentAct.getApplicationContext());
    }
    else
    {
        (new User(inFirstName, inLastName, inCountry, inCity, inAddress, inBirthDate, inPhoneNo, inUserName, inPassWord)).insertUser();

        currentAct.getInFirstNameET().setText(null);
        currentAct.getInLastNameET().setText(null);
        currentAct.getInCountryET().setText(null);
        currentAct.getInCityET().setText(null);
        currentAct.getInAddressET().setText(null);
        currentAct.getInBirthDateBTN().setText(null);
        currentAct.getInPhoneNoET().setText(null);
        currentAct.getInUserNameET().setText(null);
        currentAct.getInPassWordET().setText(null);

        currentAct.startActivity(new Intent(currentAct.getApplicationContext(), MainActivity.class));
    }
}

Toast.makeText(currentAct, "No Internet Connection Please Get Online ", Toast.LENGTH_LONG).show();
```

EmergencyCaller

Ο παρουσιαστής αυτός αντιπροσωπεύει την όψη EmergencyCallerActivity η οποία είναι η σημαντικότερη όψη διότι υλοποιεί την κύρια λειτουργία της εφαρμογής. Όπως κάθε παρουσιαστής έτσι και αυτός κατά την κατασκευή του παίρνει ως όρισμα τν αντίστοιχη όψη. Οι μέθοδοι που υλοποιεί είναι οι παρακάτω :

Void promptUsersByOption() : Είναι υπεύθυνο για να δείξει την επιλογή στον χρήστη σχετικά με τον τρόπο που θα εξυπηρετηθεί από την αντίστοιχη υπηρεσία. Εδώ θα τεθεί και το static πεδίο του παρουσιαστή που είναι το ServiceID που επιλέχθηκε και χρησιμεύει στις παρακάτω μεθόδους.

Void showOnGoogleMaps(); : Θα μεταφέρει τον χρήστη στο activity που δείχνει την διαδρομή του κοντινότερου σημείου που ψάχνει, χρησιμοποιώντας το ServiceID που προαναφέραμε. Η διαδικασία θα είναι να ψάξουμε με τον τύπο της υπηρεσίας τις τοποθεσίες που υπάρχουν σε μια συγκεκριμένη ακτίνα. Αφού τις βρούμε, θα φτιάξουμε μια λίστα με αυτές, θα τις ταξινομήσουμε ανάλογα με την απόσταση που έχουν από το σημείο που βρίσκεται ο χρήστης και θα επιστρέψουμε αυτή που θα βρούμε πιο κοντά. Αμέσως μετά θα ζητήσουμε την διαδρομή από το GoogleDirections, το οποίο και θα εμφανίσουμε μέσω του αντικειμένου PolylineOptions στον χάρτη.

Void notifyService(); :Είναι υπεύθυνο για την ειδοποίηση της υπηρεσίας μέσω ανάλογου μηνύματος σχετικά με την τοποθεσία και βασικά στοιχεία του χρήστη καθώς και αυτή η κλήση θα εγγραφεί στη βάση μόνο αν σταλεί επιτυχώς.

Κώδικας Υλοποίησης

```
import android.content.Intent;
import android.util.Log;
import android.widget.Toast;

import com.example.komabi.emergencyapp.R;

class EmergencyCaller extends AlertDialogg
{
    private static int OptionID;
    private EmergencyCallerActivity currentAct;

    EmergencyCaller(EmergencyCallerActivity inAct)
    {
        currentAct = inAct;
        Option.initializeOptionList();
    }

 void promptUsersByOption()
    {
        try
        {

            if (isNetworkAvailable((ConnectivityManager) currentAct.getSystemService(Context.CONNECTIVITY_S
            {
                String optionName = (new Option()).getListById(String.valueOf(OptionID)).getOptionName();
                currentAct.promptUsersByOption(optionName);
            }
            else
                Toast.makeText(currentAct, "No Internet Connection,Please Get Online ", Toast.LENGTH_LONG).

        }
        catch (Exception x)
        {

            printException(x, currentAct.getApplicationContext());
        }
    }
```

```
  void showOnGoogleMaps()
  {
      try
      {

          if (isNetworkAvailable((ConnectivityManager) currentAct.getSystemService(Context.CONNECTIVITY_SERVICE)))
          {

              String OptionGPName = (new Option()).getListById(String.valueOf(OptionID)).getOptionGPName();
              Intent gMapsIntent = new Intent(currentAct.getApplicationContext(), GoogleMapsActivity.class);
              gMapsIntent.putExtra("OptionGPName", OptionGPName);
              //locMgr = (LocationManager)currentAct.getSystemService(Context.LOCATION_SERVICE);
              //boolean enabled = locMgr.isProviderEnabled(LocationManager.GPS_PROVIDER);

              currentAct.startActivity(gMapsIntent);
          }
          else
              Toast.makeText(currentAct, "No Internet Connection,Please Get Online ", Toast.LENGTH_LONG).show();
      }
      catch (Exception x)
      {
          printException(x, currentAct.getApplicationContext());
      }
  }
```

```
void notifyOption()
{
    try
    {
        if (isNetworkAvailable((ConnectivityManager) currentAct.getSystemService(Context.CONNECTIVITY_SERVICE))
        {

            String drReceiverStr = "DeliveryReportReceiver";
            String sReceiverStr = "SentReceiver";
            String optionName = (new Option()).getListById(String.valueOf(OptionID)).getOptionName();
            String optionNo = (new Option()).getListById(String.valueOf(OptionID)).getOptionNo();
            String OptionMail = (new Option()).getListById(String.valueOf(OptionID)).getOptionMailAddress();
            String currentLong = FindLocation.getLongitude();
            String currentLat = FindLocation.getLatitude();
            User currentUsers = (new User()).getListById(currentAct.getIntent().getExtras().getString("CID"));

            String messageToSend = "I am " + currentUsers.getFirstName() + " " + currentUsers.getLastName() +
                    "\nMy location is:\nLatitude: " + currentLat + "\nLongitude: " + currentLong;
            String messageTitle = "SOS:: Longitude: " + currentLong + " Latitude: " + currentLat;

          PendingIntent drPendInt = PendingIntent.getBroadcast(currentAct, 0, (new Intent(drReceiverStr)), 0);
          PendingIntent sPendInt = PendingIntent.getBroadcast(currentAct, 0, (new Intent(sReceiverStr)), 0);


            (SmsManager.getDefault()).sendTextMessage(
                    optionNo,
                    null,
                    messageToSend,
                    drPendInt,
                    sPendInt);


            Toast.makeText(currentAct, "SMS MESSAGE WAS SUCCESSFULLY SENDED", Toast.LENGTH_LONG).show();

            (new Option()).saveOptionCall(
                    currentAct.getIntent().getExtras().getString("CID"),
                    String.valueOf(OptionID),
                    optionName.toString(),
```

GoogleMaps

Αυτός ο παρουσιαστής είναι υπεύθυνος για τη διαδικασία παρουσίασης της διαδρομής στον χρήστη αφού συνδέεται με το αντίστοιχο activity. Διαθέτει την παρακάτω μέθοδο:

Void adjustMap (GoogleMap inmap); : Που θα γίνει η ουσιαστική εκτέλεση της λειτουργίας εύρεσης και επίδειξης της διαδρομής που περιγράψαμε παραπάνω.

Διαθέτει ταυτόχρονα μια μέθοδο με την χρήση της οποίας γίνεται η εύρεση του κοντινότερου σημείου στο σημείο που βρίσκεται ο χρήστης.

Κώδικας Υλοποίησης

```
public void adjustMap(GoogleMap inMap)
{
    inMap.getUiSettings().setZoomControlsEnabled(true);

    LatLng startPoint = new LatLng(Double.parseDouble(FindLocation.getLatitude()),
    Double.parseDouble(FindLocation.getLongitude()));


    Location currentLocation = new Location("Your Location Now");
    currentLocation.setLatitude(startPoint.latitude);
    currentLocation.setLongitude(startPoint.longitude);

    inMap.addMarker(new MarkerOptions().position(startPoint).title("You"));

    try
    {
        PlacesAsyncBackground pAsyncCall = new PlacesAsyncBackground(currentAct.getIntent().getExtras().getStri
        JSONObject placesResponseJSON = new JSONObject(pAsyncCall.execute().get());
        JSONArray placesResultJSON = placesResponseJSON.getJSONArray("results");

        ArrayList<Location> locationList = new ArrayList<>();

        int x = placesResultJSON.length();

        for (int i = 0; i < x; i++)
        {
Location eachLocation = new Location(placesResultJSON.getJSONObject(i).getString("name"));
eachLocation.setLatitude(Double.parseDouble(placesResultJSON.getJSONObject(i).getJSONObject("geometry").getJSONObj
eachLocation.setLongitude(Double.parseDouble(placesResultJSON.getJSONObject(i).getJSONObject("geometry").getJSONOb
locationList.add(eachLocation);
        }

        Location closestLocation = getClosestLocation(locationList, currentLocation);

        inMap.addMarker(
                new MarkerOptions().position(new LatLng(closestLocation.getLatitude(), closestLocation.getLongitude()
                        .title(closestLocation.getProvider())
        );

        LocAsyncBackground dAsyncCall = new LocAsyncBackground(currentLocation, closestLocation);
        JSONObject directionsJSONArray = new JSONObject(dAsyncCall.execute().get());
        List<List<HashMap<String, String>>> routesList = (new LocParceJSON()).parse(directionsJSONArray);
        inMap.addPolyline(LocParceJSON.getPoly(routesList));

    }
    catch (Exception x)
    {
        printException(x, currentAct.getApplicationContext());
    }

    inMap.moveCamera(CameraUpdateFactory.newLatLng(startPoint));
    inMap.animateCamera(CameraUpdateFactory.zoomTo(12));
}
private Location getClosestLocation(ArrayList<Location> inLocList, Location currentLocation)
{
    Location closestLocation = null;
    float minDistance = 100000;

    for (Location eachLocation : inLocList)
    {
        if (currentLocation.distanceTo(eachLocation) < minDistance)
        {
            closestLocation = eachLocation;
            minDistance = currentLocation.distanceTo(eachLocation);
        }
    }
    return closestLocation;
```

Επικοινωνία με τη βάση

Για να πραγματοποιηθεί η επικοινωνία με την βάση θα φτιάξουμε ένα Web Service με την χρήση της γλώσσας PHP που θα εκτελεί τις αποθηκευμένες διαδικασίες και θα επιστρέφει το αποτέλεσμα στην εφαρμογή. Προτιμάται αυτή η διαδικασία διότι παρέχει ασφάλεια για τις πληροφορίες σύνδεσης με τη βάση και κρατάει κρυφή όλη τη διαδικασία από την ίδια την εφαρμογή. Ο κώδικας δίνεται παρακάτω.

Παρατηρείται ότι μόνο εάν οριστεί η μεταβλητή ‘action’ και έρθει στο Web Service μέσω μεθόδου POST, θα διεξαχθεί η συναλλακτική διαδικασία με τη βάση. Σε άλλη περίπτωση θα δώσει στην αντίστοιχη έξοδο ένα μήνυμα αποτυχίας με μορφή HTML.

```php
<?php
    if (isset($_POST["action"]))
    {

        $serverName = "localhost";
        $dbUserName = "root";
        $dbPassWord = "";
        $dbName = "id641894_sos";

        $temp = Array();

        $dbConn = mysqli_connect($serverName, $dbUserName, $dbPassWord, $dbName) or die(mysqli_error($dbCon));


        $sqlR = mysqli_query($dbConn,$_POST["action"])or die("Error: ".mysqli_error($dbConn));

        while ($row = mysqli_fetch_array($sqlR))
        {
            array_push($temp, $row);
        }

        mysqli_close($dbConn);
        header('Content-Type: application/json');
        echo json_encode($temp);
    }
    else
    {

    }
?>
```

Σενάρια Μελλοντικής Βελτίωσης

Στο άμεσο μέλλον αν υποθέσουμε ότι η εφαρμογή αυτή γίνει διαθέσιμη προς το κοινό για την εξυπηρέτησή τους θα μπορούσαμε να κάνουμε τις εξής αλλαγές ώστε η εφαρμογή να είναι πιο αξιόπιστη σε διάφορα θέματα .

- Θα μπορούσαμε να προσθέσουμε ακεραιότητα των στοιχείων του χρήστη δηλαδή κάθε χρήστης όταν πραγματοποιεί εγγραφή στην εφαρμογή τα δεδομένα να στέλνονται κατά κύριο λόγο στην αστυνομία ώστε να ελέγξει αν τα στοιχεία είναι σωστά ώστε να αποφύγουμε την άσκοπη χρήση της εφαρμογής από ψεύτικους χρήστες.

- Θα μπορούσαμε να αναπτύξουμε μια πλατφόρμα για την κάθε υπηρεσία ώστε να βλέπουν σε πραγματικό χρόνο τις κλήσεις έκτακτης ανάγκης των χρηστών.

- Θα μπορούσαμε να στέλνουμε την τοποθεσία του οχήματος της κάθε υπηρεσίας αφότου πραγματοποιηθεί η κλήση ώστε να γνωρίζει ο χρήστης πότε πλησιάζει το όχημα .

- Θα μπορούσαμε να εφαρμόσουμε στην εφαρμογή φωνητικές εντολές με συγκεκριμένες λέξεις κλειδιά όπως για παράδειγμα «Βοήθεια ‘Υπηρεσία’ » για άτομα με ειδικές ανάγκες ή ηλικιωμένους.

Αναφορές και βιβλιογραφία

Αναφορές

http://www.gabesechansoftware.com/location-tracking/
http://www.dmst.aueb.gr/dds/ism/oo/indexw.htm
https://el.wikipedia.org/wiki/Java
https://en.wikipedia.org/wiki/XML
https://en.wikipedia.org/wiki/Android_Studio
https://en.wikipedia.org/wiki/PHP
https://en.wikipedia.org/wiki/Gradle
https://docs.oracle.com/javase/7/docs/api/
https://developer.android.com/reference/packages.html
http://www.w3schools.com/
https://en.wikipedia.org/wiki/Unified_Modeling_Language
http://www.dmst.aueb.gr/dds/ism/oo/indexw.htm
https://en.wikipedia.org/wiki/Android_KitKat
http://gizmodo.com/a-history-of-android-from-cupcake-to-m_____-1707432419

http://thetechhacker.com/2013/09/24/easily-root-android-device-one-click-using-framaroot/

Βιβλία

Android Programming - The Big Nerd Ranch Guide - E Book
Android 6 Studio Development Essentials - Neil Smyth - E Book
Java Programming for Android Developers for Dummies - E Book
Android Phones For Dummies - E Book
PHP for the Web - Visual Quickstart Guide - E Book