ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ,ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Ανάπτυξη εφαρμογής κοινωνικών δικτύων σε Android

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

Άννας Ξενάκη

Επιβλέπων: Θεοδώρα Βαρβαρίγου
Καθηγήτρια Ε.Μ.Π

Αθήνα, Μάϊος 2013

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Ανάπτυξη εφαρμογής κοινωνικών δικτύων σε Android

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΑΝΝΑΣ ΞΕΝΑΚΗ

Επιβλέπων: Θεοδώρα Βαρβαρίγου

Καθηγήτρια Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 18η Φεβρουαρίου 1999

(Υπογραφή)	(Υπογραφή)	(Υπογραφή)
.....	.....	.....
ΘεοδώραΒαρβαρίγου	Ελευθέριος Καγιάφας	Βασίλειος Λούμος
Καθηγήτρια Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Μάϊος 2013

(Υπογραφή)

.....

Ξενάκη Άννα

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Ο σκοπός της διπλωματικής εργασίας ήταν η δημιουργία μίας εφαρμογής που θα ανασύρει δεδομένα από το δίκτυο κοινωνικής δικτύωσης LinkedIn και στη συνέχεια θα τα παρουσιάζει. Η υλοποίηση της εφαρμογής έγινε πάνω στο σύστημα Android. Το "Android" είναι ένα Λειτουργικό Σύστημα για κινητά τηλέφωνα το οποίο μας επιτρέπει να δημιουργούμε εφαρμογές γι' αυτά. Ο σχεδιασμός της εφαρμογής έγινε στη πλατφόρμα eclipse με τη βοήθεια ειδικών εργαλείων σχεδιασμού για android.

Πιο συγκεκριμένα η εφαρμογή, με την άδεια του χρήστη, ζητά δεδομένα του βασικού προφίλ του χρήστη, τις επαφές του και όσα δεδομένα πάνω σε αυτές παρέχει το LinkedIn. Στη συνέχεια προβάλει το προφίλ του χρήστη, τη λίστα με τις επαφές του χρήστη ονομαστικά και κατηγοριοποιεί τις επαφές του χρήστη με βάση το κλάδο της επιχείρησης που έχουν δηλώσει για να μπορεί ο χρήστης να βρει συνεργάτες στον κύκλο των επαφών του ανα κατηγορία.

Η εφαρμογή αυτή χρησιμοποιεί την υπάρχουσα βιβλιοθήκη του socialauth για android 1.0 καθώς και την αντίστοιχη κύρια βιβλιοθήκη που διευκολύνει την επικοινωνία με κοινωνικά δίκτυα.

Λέξεις κλειδιά: κινητό, android, LinkedIn, δίκτυα κοινωνικής δικτύωσης, εφαρμογή

Abstract

The purpose of my diplomatic work was to create an application that will retrieve data from the social network LinkedIn and then present them on a mobile phone screen. The application will run in mobile phones over Android. Android is a Linux- based operating system designed primarily for touch screen mobile devices such as smartphones and tablet computers which provides tools for creating applications. The application was designed on the eclipse platform with the assistance of android tools.

The application logs in with the user's profile in LinkedIn and requests permission for basic profile information and user's connections as well as information provided from LinkedIn for those connections. Later it presents a user profile view, a connection list view and a view with categorized connections by industry so users can find associates by category.

This application uses the library of socialauth for android and its compatible basic library for communication with social networks.

keywords: mobile phone, android, LinkedIn, social networks, application

Πίνακας Περιεχομένων

1.Εισαγωγή

Στην ενότητα αυτή θα κάνουμε μία σύντομη ιστορική αναδρομή στη κινητή τηλεφωνία εμβαθύνοντας στα smartphones και στο λειτουργικό σύστημα Android για να κατανοήσουμε καλύτερα την εφαρμογή που δημιουργήσαμε.

1.1. Κινητή Τηλεφωνία

Κινητό τηλέφωνο (ή απλά κινητό) , ονομάζεται κατά κύριο λόγο το τηλέφωνο που δεν εξαρτάται από φυσική καλωδιακή σύνδεση με δίκτυο παροχής τηλεφωνίας και δεν εξαρτάται από κάποια τοπική ασύρματη συσκευή εκπομπής ραδιοφωνικού σήματος χαμηλής συχνότητας. Τα κινητά τηλέφωνα χρησιμοποιούν τεχνολογία κυψελών (cells) και εκπέμπουν σε υψηλές συχνότητες. Για την εκπομπή και λήψη των σημάτων χρησιμοποιείται πλέον, αποκλειστικά ψηφιακή τεχνολογία με κωδικοποίηση.

Από το 1990 μέχρι το 2011 οι χρήστες των κινητών συσκευών αυξήθηκαν ραγδαία απο 12.4 εκατομμύρια σε 6 δις εκατομμύρια, προσεγγίζοντας το 87% του πληθυσμού.

Σημέρα τα κινητά τηλέφωνα υποστηρίζουν πολύ περισσότερες δυνατότητες πέρα από τη διεξαγωγή απλών κλήσεων. Κάποιες από αυτές είναι τα μηνύματα, MMS, email, internet, υπέρυθρες, bluetooth, επαγγελματικές εφαρμογές, παιχνίδια και η φωτογραφική μηχανή. Οι συσκευές που προσφέρουν όλες αυτές τις υπηρεσίες και γενικότερα ιδιότητες υπολογιστών αναφέρονται γενικότερα ως smartphones.

Η πρώτη κινητή συσκευή τηλεφώνου χειρός δημιουργήθηκε από τους John F. Mitchell και Dr Martin Cooper για τη Motorola το 1973 και ζύγιζε περίπου 1 κιλό. Το πρώτο διαφημιστικό έγινε απο το δίκτυο NTT της Ιαπωνίας το 1979 αρχικά στη πρωτεύουσα του Τοκυο. Το 1981 ακολούθησαν διαφημίσεις για το Nordic Mobile Telephone (NMT) στις Δανία, Φινλανδία,Σουηδία και Νορβηγία. Πολλές άλλες χώρες ακολούθησαν συμπεριλαμβανομένων της Αγγλίας, του Μεξικού και του Καναδά. Στις 6 Μαρτίου του 1983 η Ameritech κυκλοφόρησε στην αγορά το

μοντέλο DynaTAc το οποιο ειχε 1G, ώρες ομιλίας 0,5 και για τη λειτουργία του απαιτούσε 10 ώρες φόρτησης. Το 1991 η Radiolinja στη Φινλανδία έβγαλε τα πρώτα κινητά τηλέφωνα 2G βασισμένα στο πρότυπο GSM.

Δέκα χρόνια αργότερα το 2001 τα πρώτα κινητά 3G κυκλοφόρησαν στην Ιαπωνία από την NTT DoCoMo βασισμένα στο πρότυπο WCDMA. Στη συνέχεια ακολούθησαν 3.5G, 3G+, turbo 3G βασισμένα στο high-speed packet access (HSPA) παρέχοντας τη δυνατότητα στα δίκτυα UMTS να κάνουν ταχύτερη μεταφορά δεδομένων και μεγαλύτερη χωρητικότητα. Μέχρι το 2009 είχε γίνει σαφές ότι τα δίκτυα 3G δε μπορούσαν να υποστηρίξουν εφαρμογές που σχετίζονταν με εφαρμογές πολυμέσων του ιντερνετ (streaming media). Κατά συνέπεια άρχισαν να αναπτύσσονται τα κινητά 4ης γενιάς με έως και 10 φορές καλύτερες ταχύτητες σε σχέση με τα δίκτυα 3ης γενίας. Τα κινητά 4G στηρίζονταν σε δύο πρότυπα, το WiMAX και το LTE που παρουσιάστηκαν πρώτα από τη Σκανδιναβική TeliaSonera. Όμως οι εξελίξεις στη κινητή τηλεφωνία δε σταμάτησαν εκεί το 2005 η Android Inc. κατασκεύασε το λογισμικό για τα πρώτα κινήτα smartphones και tablets το οποίο αργότερα αγοράστηκε από τη Google. Το πρώτο κινητό με λογισμικό android πουλήθηκε τον Οκτώβριο του 2008.

1.2. Android

1.2.1. Ιστορία

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

Η Android Inc. ιδρύθηκε στο Palto Alto της Καλιφόρνια τον Οκτώβριο του 2003 από τους Andy Rubin, Rich Miner, Nick Sears, Chris White. Αρχικός σκοπός της

εταιρείας ήταν η κατασκευή λογισμικού για ψηφιακές μηχανές όταν συνειδητοποίησαν ότι δεν αφορά μεγάλο αγοραστικό κοινό και έστρεψαν τις προσπάθειες τους στη κατασκευή μιας κινητής συσκευής που να δουλεύει με λογισμικό παραπλήσιο της Symbian και της Windows Mobile (τότε δεν είχε ακόμη κυκλοφορήσει το iPhone της Apple).

Η Google αγόρασε την Android Inc. στις 17 Αυγούστου 2005. Μετά την αγορά οι Rubin, Miner και White παρέμειναν στην εταιρεία. Στη Google η ομάδα του Rubin ανέπτυξε λογισμό με βάη τον φλοιό των Linux.

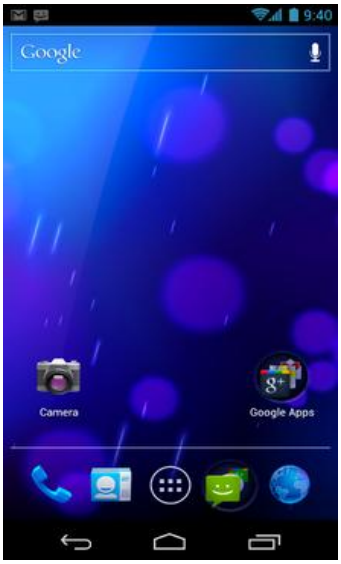
Η πρώτη παρουσίαση της πλατφόρμας Android έγινε στις 5 Νοεμβρίου 2007, παράλληλα με την ανακοίνωση της ίδρυσης του οργανισμού Open Handset Alliance, μιας κοινοπραξίας 48 τηλεπικοινωνιακών εταιριών, εταιριών λογισμικού καθώς και κατασκευής hardware, οι οποίες είναι αφιερωμένες στην ανάπτυξηκαι εξέλιξη ανοιχτών προτύπων στις συσκευές κινητής τηλεφωνίας. Το πρώτο κινητό Android ήταν το HTC Dream και κυκλοφόρησε στις 22 Οκτωβρίου 2008.

Από το 2008 και μετά το Android έβγαλε πολλές ενημερώσεις και σταδιακά βελτιώσε εκθετικά το λειτουργικό σύστημα. Κάθε καινούργια έκδοση ονομάζεται με το πρώτο γράμμα σε αλφαβητική σειρά σε σχέση με τις προηγούμενες εκδόσεις με βάση κάποιο γλυκό.

1.2.2. Διεπιφάνεια (Interface)

Η διεπιφάνεια του android στηρίζεται στον απευθείας χειρισμό της συσκευής, δηλαδή με τη χρήση οθόνης αφής ο χρήστης μπορεί με απλές κινήσεις των δακτύλων να χειρίζεται πολλές οθόνες και λειτουργίες άμεσα. Οι κινήσεις του χρήστη γίνονται αντιληπτές από τη συσκευή και η συσκευή αφήνει το χρήστη να καταλάβει πότε έγιναν αντιληπτές με ανάλογους ήχους και δονήσεις. Οι συσκευές android έχουν επίσης επυταχυνσιόμετρα, γυροσκόπια και αισθητήρες εγγύτητας που χρησιμοποιούνται σε διάφορες εφαρμογές (όπως για παράδειγμα η μετατροπή της οθόνης από κάθετη σε οριζόντια ανάλογα με το πως κρατιέται η συσκευή ή ο χειρισμός ως τιμόνι σε μία εφαρμογή με οδήγηση εικονικών αυτοκινήτων)

Τα κινητά ανοίγουν σε μία κεντρική οθόνη όπως η οθόνη ενός υπολογιστή. Η οθόνη αυτή δομείται συνήθως από εικονίδια των διαφορων εφαρμογών τα οποία με ένα πάτημα ανοίγουν την εφαρμογή. Εικονίδια που απεικονίζουν στοιχεία λειτουργίας της συσκευής (μπαταρία, ώρα, συνδεσιμότητα δικτύου κ.τ.λ) βρίσκονται στη μπάρα λειτουργιών. Ειδοποιήσεις για χαμένες κλήσεις και άλλες ειδοποιήσεις εφαρμογών εμφανίζονται σαν Pop-ups και στη συνέχεια εξαφανίζονται από τη κεντρική οθόνη και εμφανίζονται σε ξεχωριστή οθόνη που εμφανίζεται με συρσιμο του δαχτύλου προς τα κάτω, πάνω στην οθόνη.



Εικόνα 1 Αρχική οθόνη android

1.2.3. Εφαρμογές και Ανάπτυξη

Εφαρμογές σχεδιάζονται για κινητά android στη γλώσσα προγραμματισμού java με τη βοήθεια του Android Software Development Kit (SDK[1]). Το SDK περιέχει debugger, βιβλιοθήκες, προσομοιωτή βασισμένο στο QEMU (Quick emulator), αρχεία με πληροφορίες , παραδείγματα κώδικα και tutorials. Η πιο δημοφιλής πλατφόρμα που χρησιμοποιείται είναι το Eclipse το οποίο έχει τα αντίστοιχα εργαλεία για android (Android development Tools (ADT)). Άλλα εργαλεία που χρησιμοποιούνται είναι το Native Development Kit για εφαργμογές σε C ή C++ και το Google App Inventor με γραφικό περιβάλλον για αρχάριους σχεδιαστές λογισμικού.

1.2.4. Linux

Οι αρχικές εκδόσεις του android βασίζονται στο φλοιό των linux έκδοσης 2.6 και από το Android Ice Cream Sandwich 4.0 και έπειτα στην έκδοση των linux 3.x με βιβλιοθήκες και εφαρμογές γραμμένες σε C και application framework που

[1] http://developer.android.com/sdk/index.html

περιέχει βιβλιοθήκες συμβατές με τη Java βασισμένες στο Apache Harmony. Το android χρησιμοποιεί τη Dalvik virtual machine με just-in-time μεταγλώττιση για να τρέξει το εκτέλεσιμο αρχείο Dalvik που είναι μετεφρασμένο από κώδικα Java σε μορφή byte. Το android ακολουθεί την αρχιτεκτονική ARM (βασισμένη σε RISC υπολογιστές).

Ο φλοιός των linux έχει υποστεί αρχιτεκτονικές αλλαγές από τη Google εκτός του κύκλου της τυπικής ανάπτυξης του φλοιού. Δεν υποστηρίζει το πρωτόκολλο X Windows System[1] ούτε το πλήρες σύνολο των κλασικών βιβλιοθηκών της GNU και αυτό καθιστά δύσκολο να εγκαθιστούν τις υπάρχουσες εφαρμογές Linux ή βιβλιοθήκες για Android. Υποστήριξη απλής C και SDL [2] εφαρμογών είναι δυνατή με προσθήκη μίας shim[3] και χρήση JNI[4] .

Η μνήμη των κινητών android είναι χωρισμένη σε πολλά κομμάτια όπως "/system" για το λειτουργικό σύστημα και "/data" για το χρήστη και τις εγκαταστάσεις των εφαρμογών. Σε αντίθεση με τις εκδόσεις των linux οι χρήστες των android δεν έχουν δικαιώμα να πειράξουν μέρη της μνήμης όπως "/system" γιάτι είναι επιστρεπτή μόνο η ανάγνωση για λόγους ασφαλείας. Παρόλα αυτά προγραμματιστές μπορούν να επιρεάσουν το σύστημα και να επιτρέψουν τη τροποποίηση αυξάνοντας τις δυνατότητες του συστήματος ή προσθέτοντας ιούς.

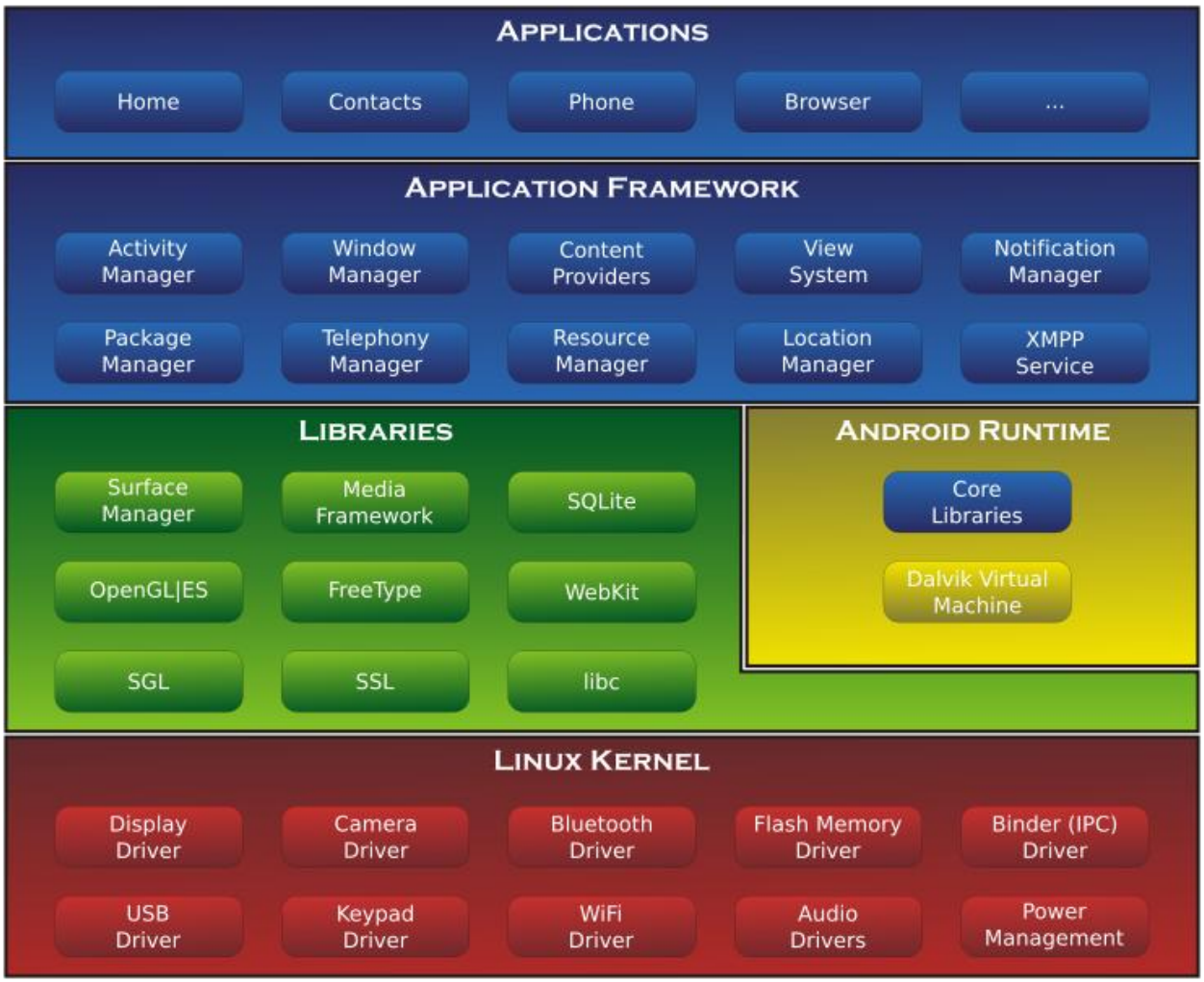
2

[1] X-Windows is a computer software system and network protocol that provides a basis for graphical user interfaces (GUIs) and rich input device capability for networked computers.

[2] Simple DirectMedia Layer (SDL) is a cross-platform, free and open source multimedia library written in C that presents a simple interface to various platforms' graphics, sound, and input devices.

[3] shim is a small library that transparently intercepts an API and changes the parameters passed, handles the operation itself, or redirects the operation elsewhere.

[4] Java Native Interface (JNI) is a programming framework that enables Java code running in a Java Virtual Machine (JVM) to call, and to be called by, native applications (programs specific to a hardware and operating system platform) and libraries written in other languages such as C, C++ and assembly.



Εικόνα 2 Περιγραφή συστήματος android

1.2.5. Διαχείρηση Μνήμης

Επειδή οι περισσότερες συσκευές android λειτουργούν με μπαταρία, είναι σχεδιασμένο να χειρίζεται τη μνήμη (RAM) και να έχει την ελάχιστη κατανάλωση μπαταρίας. Όταν κάποια εφαρμογή δε χρησιμοποιείται πλέον από το χρήστη το σύστημα περιορίζει τη μνήμη που της αφιερώνει κρατώντας την όμως ακόμη ανοιχτή. Οι εφαρμογές που παραμένουν ανοιχτές αλλά εκτός χρήσης δε χρησιμοποιούν τους πόρους της συσκευής (μπαταρία) μέχρι να επανέλθουν στο προσκήνιο. Αυτό βελτιώνει την απόκριση των εφαρμογών καθώς κάθε φορά που

ο χρήστης επιλέγει να τις χρησιμοποιήσει δε χρειάζεται να τις ανοίξει εξ'αρχής. Το android χειρίζεται τις εφαρμογές που παραμένουν στο παρασκήνιο αυτόματα. Όταν η μνήμη ελαχιστοποιείται το σύστημα θα αρχίσει να κλείνει εφαρμογές που ήταν ανενεργές για μεγάλο χρονικό διάστημα.

1.2.6. Ασφάλεια

Το android τρέχει τις διάφορες εφαρμογές στο *sandbox*, μια απομονωμένη περιοχή του συστήματος που δεν έχει πρόσβαση στο υπόλοιπο σύστημα εκτός αν έχουν δοθεί τα αντίστοιχα δικαιώματα στην εγκατάσταση από το χρήστη.

1.2.7. Licensing

Ο κώδικας του android πάρεχεται ως *free and open-source* λογισμικό . Η Google διαθέτει τα περισσότερα κομμάτια του κώδικα (συμπεριλαμβανομένων των σωρών του δικτύου και της τηλεφωνίας) υπό το license της *Apache v.2.0* και τις υπόλοιπες αλλαγές στο φλοιό των linux υπό το license της *GNU General Public License v.2.* Η *Open Handset Alliance* τις αλλαγές στο φλοιό των Linux τις διέθετε πάντα ανοιχτές σε όλους ενώ το υπόλοιπο λογισμικό του android που σχεδιάστηκε από τη Google γινόταν διαθέσιμο μόλις κυκλοφορούσε η καινούργια έκδοση. Οι εφαρμογές android που σχεδιάζονται από ιδιώτες φέρουν τα δικά τους licenses.

Στη συνέχεια ακολουθεί ένας πίνακας με πληροφορίες για όλες τις εκδόσεις android

Εκδοση	Ονομασία	Ημερομηνία Έκδοσης	API level	Κατανομή (Μαϊος 1, 2013)
4.2.x	*Jelly Bean*	Νοέμβριος 13, 2012	17	2.3%
4.1.x	*Jelly Bean*	Ιούλιος 9, 2012	16	26.1%
4.0.x	*Ice Cream Sandwich*	Δεκέμβριος 16, 2011	15	27.5%
3.2	*Honeycomb*	Ιούλιος 15, 2011	13	0.1%
3.1	*Honeycomb*	Μάϊος 10, 2011	12	
2.3.3–2.3.7	*Gingerbread*	Φεβρουάριος 9, 2011	10	38.4%
2.3–2.3.2	*Gingerbread*	Δεκέμβριος 6, 2010	9	0.1%
2.2	*Froyo*	Μάϊος 20, 2010	8	3.7%
2.0–2.1	*Eclair*	Οκτώβριος 26, 2009	7	1.7%
1.6	*Donut*	Σεπτέμβριος 15, 2009	4	0.1%
1.5	*Cupcake*	Απρίλιος 30, 2009	3	

2.Σκοπός της εργασίας

Στα πλαίσια της εργασίας αυτής δημιουργήσαμε μία εφαρμογή για κινητές συσκευές android η οποία παίρνει δεδομένα από ένα λογαριασμό LinkedIn και τα παρουσιάζει σε οθόνες. Η εφαρμογή αυτή χρησιμοποιεί την υπάρχουσα βιβλιοθήκη του socialauth. Πιο συγκεκριμένα ο χρήστης με τη βοήθεια του socialauth συνδέεται στο λογαριασμό του στο LinkedIn και δίνει στην εφαρμογή τα δικαιώματα να ανασύρει δεδομένα από το προφίλ του.

Αφού γίνει η σύνδεση και ο χρήστης μπει στο λογαριασμό του, βγαίνει ένα παράθυρο που του δίνει την επιλογή να δει το προσωπικό του προφίλ, να γράψει κάτι στο τοίχο του, να δει τη λίστα των φίλων του και να δει ποιοι από τους φίλους του ανήκουν σε κάποιο τομέα εργασίας ανά κατηγορίες.

Για το προφίλ του χρήστη ζητούνται τα δεδομένα του βασικού προφίλ όνομα, επίθετο, χώρα, τοποθεσία, url της φωτογραφίας προφίλ και e-mail. Στη συνέχεια αυτά παρουσιάζονται σε ξεχωριστή οθόνη.

Για τη λίστα των επαφών του χρήστη ζητούνται τα δεδομένα λίστας χρήστη πιο συγκεκριμένα όνομα και επίθετο. Στη συνέχεια σε ξεχωριστή οθόνη παρουσιάζεται η λίστα επαφών του χρήστη με όνομα και επίθετο.

Για τη λίστα επαφών του χρήστη με βάση το τομέα εργασίας ζητούνται και πάλι δεδομένα λίστας χρήστη και πιο συγκεκριμένα, όνομα, επίθετο, url της φωτογραφίας προφίλ, url του προφίλ της επαφής και τομέας εργασίας. Στη συνέχεια κατηγοριοποιεί της επαφές του χρήστη σε κατηγορίες με βάση το γενικότερο τομέα εργασίας (πχ. οικονομικά) κατόπιν επιλογής της πρώτης κατηγορίας ο χρήστης καλείται να επιλέξει με βάση τον τομέα εργασίας που έχουν δηλώσει οι επαφές του. Στο τελός πρασουσιάζονται τα στοιχεία της επαφής σε ξεχωριστή οθόνη.

Καθ'όλη τη διάρκεια λειτουργίας της εφαρμογής ο χρήστης έχει τη δυνατότητα να πλοηγείται εντός της εφαρμογής σε προηγούμενα παράθυρα.

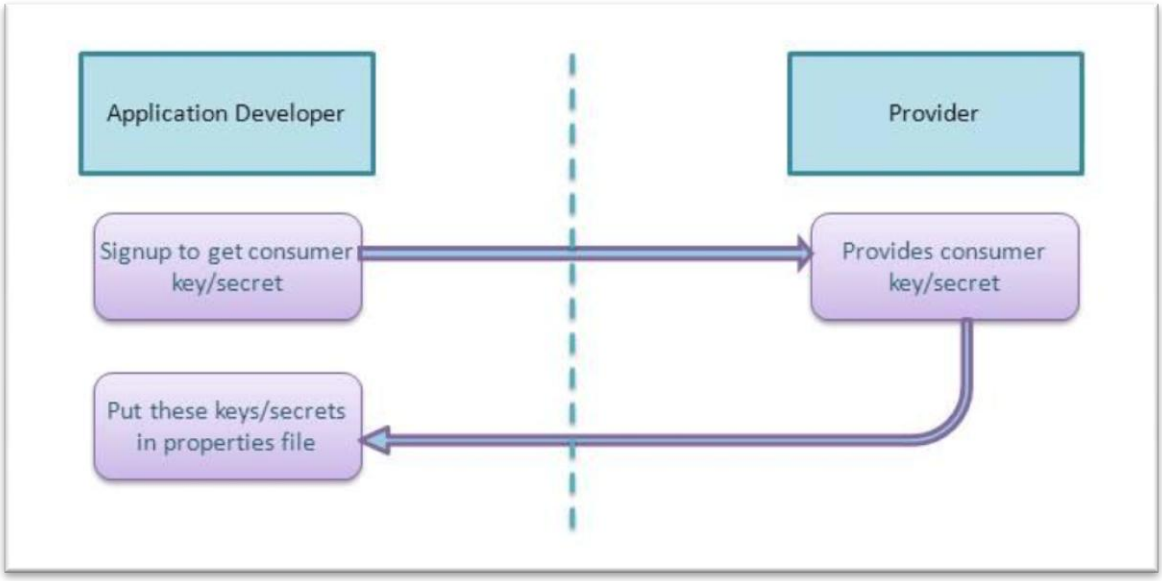
3.Συμπληρωματικά προγράμματα- εργαλεία

Για τη λειτουργία της εφαρμογής χρησιμοποιήθηκαν ειδικά εργαλεία της πλατφόρμας eclipse καθώς και υπάρχουσες βιβλιοθήκες του socialauth-android και socialauth. Επίσης χρησιμοποιήθηκαν λειτουργίες της κονσόλας Apigee, πρωτόκολλα εξουσιοδότησης.

3.1. Socialauth

Σε αυτό το σημείο πρέπει να γίνει μια αναφορά στο Socialauth και το τρόπο που δουλεύει.

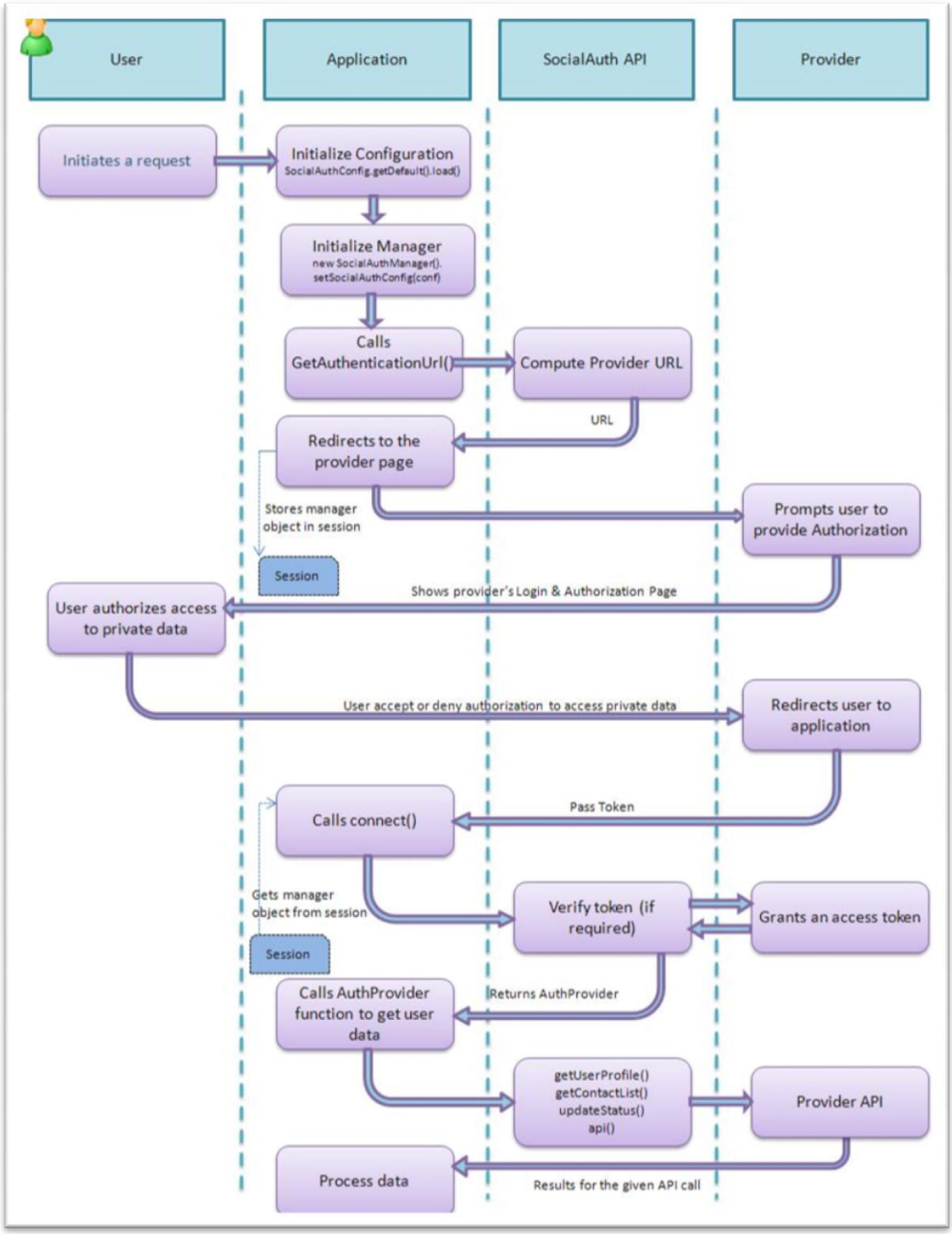
Για τη δημιουργία οποιασδήποτε εφαρμογής android που ανασύρει δεδομένα από κάποιο κοινωνικό δίκτυο είναι απαραίτητη η ανάληψη δικαιωμάτων δημιουργίας εφαρμογής από το εκάστοτε κοινωνικό δίκτυο. Για αυτό το σκοπό κάθε προγραμματιστής λαμβάνει τα αντίστοιχα κλειδιά της εφαρμογής του. Αντίστοιχα και το Socialauth και στη περίπτωσή μας το myLinkedInApp για να καταφέρουν τη σύνδεση της εφαρμογής στο κοινωνικό δίκτυο πρέπει πρώτα να κάνουν σύνδεση (Log in) με τα κλειδιά τους.



Εικόνα 3 Διαδικασία λήψης κλειδιών εφαρμογής

Αφού έχει γίνει επιτυχής σύνδεση της εφαρμογής με το κοινωνικό δίκτυο της αρεσκίας μας ζητείται από το χρήστη να συνδεθεί με το προσωπικό του λογαριασμό στο κοινωνικό δίκτυο. Τη στιγμή που ο χρήστης θα πατήσει τι θέλει να εμφανιστεί από την εφαρμογή στη συνέχεια θα σταλεί μία αίτηση GET στο κοινωνικό δίκτυο και αυτή αφού επιβεβαιώσει ότι ο χρήστης έχει όντως δικαίωμα να κάνει αυτή την αίτηση θα επιστρέψει το αποτέλεσμα της αίτησης μας. Αυτό θα το λάβει η εφαρμογή μας και θα το επεξεργαστεί ανάλογα ώστε να μπορεί να το παρουσιάσει. Όταν τελειώσει αυτή η διαδικασία θα παρουσιαστούν γραφικά τα αποτελέσματα σε καινούργια οθόνη.

Στην Εικόνα 4 Εικόνα 3 Διαδικασία λήψης κλειδιών εφαρμογήςβλέπουμε πιο παραστατικά τη διαδικασία.



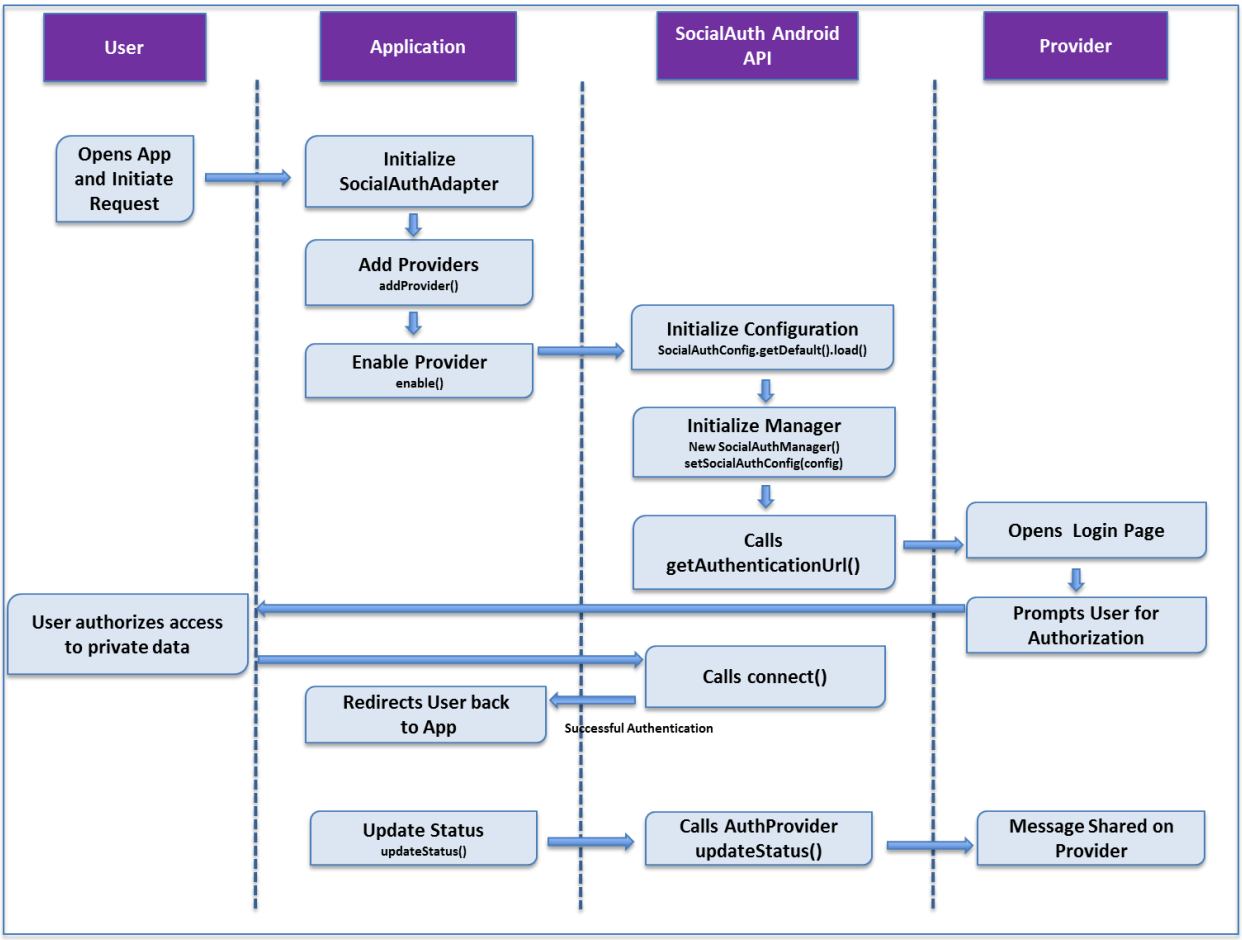
Εικόνα 4 Παραστατική λειτουργία συστήματος

3.2. Socialauth-android

Το Socialauth -android προσθέτει τα χαρακτηριστικά του Socialauth σε περιβάλλον android. Εφ'όσον προστεθεί στην εφαρμογή ακολουθεί η εξουσιοδότηση σε περιβάλλον android.

Ο χρήστης ανοίγει την εφαρμογή και επιλέγει τον πάροχο από τον οποίο θα ζητήσει την εξουσιοδότηση χρησιμοποιώντας τη βιβλιοθήκη του SocialAuth-android. Στη συνέχεια ο χρήστης προτρέπεται να εισάγει τα απαραίτητα στοιχεία για τη σύνδεση. Έπειτα ο πάροχος θα ζητήσει τα απαραίτητα δικαιώματα για να μοιραστεί τα δεδομένα του χρήστη στο κοινωνικό δίκτυο με την εφαρμογή. Με την αποδοχή έχει επιτευχθεί η εξουσιοδότηση και ο χρήστης επιστρέφει στην εφαρμογή. Τώρα ο χρήστης μπορεί να χειριστεί τα δεδομένα του μέσω της εφαρμογής.

Στην Εικόνα 5 ακολουθεί η διαδικασία σε διάγραμμα.



Εικόνα 5. Διάγραμμα λειτουργίας SocialAuth-android

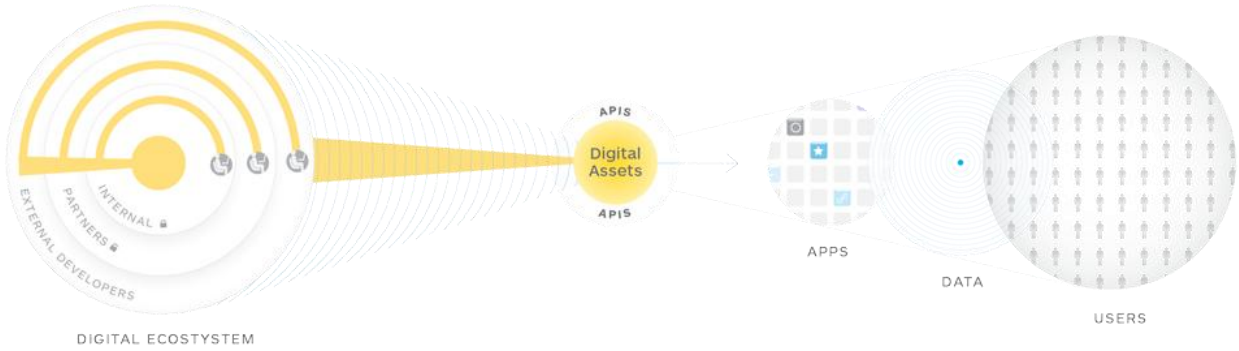
3.3. *Eclipse-Εργαλεία Σχεδιασμού Android Εφαρμογών*

Τα εργαλεία σχεδιασμού android εφαρμογών (Android Development Tools (ADT)) μπαίνουν ως πρόσθετα στη πλατφόρμα του Eclipse και διευκολύνουν το σχεδιασμό Android εφαρμογών ,φτιάχνουν το γραφικό περιβάλλον της εφαρμογής, προσθέτουν βιβλιοθήκες του Android , κάνουν έλεγχο για σφάλματα, εξάγουν υπογεγραμένα ή όχι αρχεία τύπου .apk για εγκατάσταση της εφαρμογής.

3.4. Apigee

Το Apigee είναι μια πλατφόρμα για προγραμματιστές που περιέχει στο σχεδιασμό του εργαλεία για καλύτερο, πιο λειτουργικό σχεδιασμό εφαρμογών. Πιο συγκεκριμένα διαθέτει μια πληθώρα κοινωνικών δικτύων (συνολικά 80) και μέσω της ειδικά σχεδιασμένης κονσόλας προσφέρει στο προγραμματιστή τη δυνατότητα να δει τι δεδομένα μπορεί να ζητήσει από κάθε κοινωνικό δίκτυο και να ελέγξει το αποτέλεσμα των αιτήσεων δεδομένων (requests, queries) που θα εκάνε στο πάροχο (GET, PUT, POST, DELETE).

Η Εικόνα 6 παρουσιάζει γραφικά τη λειτουργία του Apigee.



Εικόνα 6 Apigee

3.5. LinkedIn

Το LinkedIn είναι ένα κοινωνικό δίκτυο για επαγγελματίες. Δημιουργήθηκε το Δεκέμβριο του 2002 και δημοσιοποιήθηκε το Μάϊο του 2003. Η πρωταρχική του χρήση είναι για επαγγελματική δικτύωση. Από τον Ιανουάριο του 2013 το LinkedIn έχει παραπάνω από 200 εκατομμύρια χρήστες σε πάνω από 200 χώρες.

Μέσω του LinkedIn οι χρήστες έχουν τη δυνατότητα:

- Να γνωρίσουν επαγγελματικές επαφές δευτέρου βαθμού (επαφές των επαφών τους) καθώς και επαγγελματικές επαφές τρίτου βαθμού.

- Να ανεβάσουν το βιογραφικό τους ηλεκτρονικά και να παρουσιάσουν καλύτερα το επαγγελματικό τους προφίλ (εργασιακή εμπειρία και αναλυτικό του εκπαιδευτικό υπόβαθρο) και τις κοινωνικές τους δραστηριότητες.
- Να βρουν εργασία, συνεργάτες και επιχειρηματικές ευκαιρίες που προβάλλονται από τις επαφές τους.
- Ως εργοδότες να δημοσιοποιήσουν εργασιακές ευκαιρίες.
- Αν αναζητούν εργασία να κοιτάξουν τα προφίλ υπεύθυνων ανθρώπινου δυναμικού και να δουν ποιοι γνωστοί τους μπορούν να τους συστήσουν.
- Να δημοσιοποιήσουν φωτογραφίες και να δουν φωτογραφίες (με σκοπό την περιτέρω αναγνώριση των ικανοτήτων τους).
- Να ακολουθήσουν τις ενημερώσεις εταιριών και να παίρνουν ειδοποιήσεις για επαγγελματικές ευκαιρίες.
- Να αποθηκεύσουν θέσεις εργασίας που θα ήθελαν να κάνουν αίτηση και όταν παρουσιαστεί η ευκαιρία να πάρουν δράση.

Τα δεδομένα των επαφών προστατεύονται από το International Safe Harbor Privacy Principles.

Το LinkedIn εκτός των άλλων προσφέρει και κάποιες επιπλέον εφαρμογές όπως το "LinkedIn Answers" όπου χρήστες κάνουν ερωτήσεις επαγγελματικού περιεχομένου επώνυμα και δέχονται απάντησεις από άλλους επώνυμους χρήστες (πράγμα που αυξάνει την αξιοπιστία των απαντήσεων) , "LinkedIn Polls", "LinkedIn DirectAds" (μία μορφή διαφημίσεων).

Το LinkedIn επίσης υποστηρίζει τη συμμετοχή σε groups (ομάδες) ενδιαφερόντων. Οι περισσότερες ομάδες σχετίζονται με επαγγελματικά θέματα ενώ υπάρχουν και ομάδες απόφοιτων. Τα groups παρέχουν τη δυνατότητα συζήτησης η οποία όμως ελέγχεται από τους υπεύθυνους του group.

Κατά την αναζήτηση μίας εταιρείας στο LinkedIn παρουσιάζονται στατιστικά στοιχεία για την εταιρεία αυτή όπως ποσοστά ανδρών γυναικών, οι θέσεις με το μεγαλύτερο ποσοστό υπαλλήλων, η τοποθεσία των κεντρικών γραφείων τις εταιρείας ή μία λίστα από παλιούς και τωρινούς υπαλλήλους.

Τέλος, το LinkedIn προσφέρει στους χρήστες τη δυνατότητα να επαληθεύσουν τις ικανότητες που έχουν δηλώσει οι επαφές τους (endorse). Με αυτό το τρόπο πιστοποιούνται περισσότερο οι ικανότητες των μελών του LinkedIn.

3.6. OAuth - Εξουσιοδότηση

Το OAuth είναι το πιο δημοφιλές πρωτόκολλο εξουσιοδότησης που χρησιμοποιείται. Παρέχει τη δυνατότητα στους χρήστες να έχουν πρόσβαση σε πόρους που υπάρχουν σε ένα server εκ μέρους του διαχειρηστή του. Επίσης παρέχει τη δυνατότητα παραχώρησης δικαιωμάτων σε τρίτους χρήστες ώστε να μπορούν να έχουν πρόσβαση σε κάποιους πόρους που τους ανήκουν χωρίς να μοιράζονται τους κωδικούς τους με αυτούς.

Στο κλασικό μοντέλο εξουσιοδότησης ο χρήστης (client) χρησιμοποιεί τους κωδικούς του και με τη πιστοποίηση αυτών έχει πρόσβαση στους πόρους που του αντιστοιχούν στο server. Το OAuth εισάγει ένα επιπλέον ρόλο σε αυτό το μοντέλο αυτόν του διαχειριστή πόρων (resource owner). Ο χρήστης (που δεν είναι διαχειριστής πόρων) στη περίπτωση αυτή ζητάει πρόσβαση σε πόρους που χειρίζεται ο διαχειριστής και φιλοξενούνται στο server. Για αυτό το λόγο πρέπει να πιστοποιηθούν όχι μόνο ο χρήστης με τους κωδικούς του αλλά και ο διαχειριστής πόρων για αυτή την αίτηση.

3.6.1. OAuth 1.0

Το OAuth 1.0 περιγράφεται από το πρωτόκολλο *RFC5849.*

Ορολογία

- **χρήστης (client) :** Ενας χρήστης HTTP (με βάση το πρωτόκολλο RFC2616) ικανός να κάνει αιτήσεις εξουσιοδότησης OAuth.
- **διακομιστής (server):** Ενας HTTP διακομιστής με βάση το πρωτόκολλο RFC2616) ικανός να δεχτεί αιτήματα εξουσιοδότησης OAuth.
- **προστατευμένος πόρος (protected resource):** Πόρος στον οποίο αποκτά κάποιος πρόσβαση από το διακομιστή χρησιμοποιώντας αιτήσεις εξουσιοδότησης OAuth.

- **διαχειριστής πόρων(resource owner):** Μία οντότητα που έχει την ικανότητα πρόσβασης και χειρισμού των προστατευμένων πόρων αφού παρουσιάσει τα κατάλληλα διαπιστευτήρια στο διακομιστή.
- **διαπιστευτήρια(credentials):** Ενα ζευγάρι από μοναδικό όνομα και κωδικό. Το OAuth καθορίζει τρεις κλάσεις διαπιστευτηρίων : χρήστη(client), παροδικά (temporary) και τεκμήρια (token) , τα οποία χρησιμοποιούνται για να πιστοποιήσουν και να εξουσιοδοτήσουν τον χρήστη που ένα απλό αίτημα, ένα αίτημα εξουσιοδότησης και ένα αίτημα πρόσβασης αντίστοιχα.
- **τεκμήριο (token):** Ενα μοναδικό πιστοποιητικό που δίνεται από το server και χρησιμοποιείται από το χρήστη για να μπορεί να συσχετίσει τα αιτήματα με το διαχειρηστή πόρων του οποίου η εξουσιοδότηση ζητείται ή έχει ήδη αποκτηθεί από το χρήστη. Τα τεκμήρια έχουν ταιριαστό κοινό κωδικό (shared-secret) που χρησιμοποιείται από το χρήστη για να επιτευχθεί η ιδιοκτησία του τεκμηρίου και η δυνατότητα του να παρουσιάσει τον διαχειριστή πόρων.
- Σε αντιστοιχεία με τους παραπάνω όρους χρησιμοποιούνται οι :

Πελάτης : χρήστης (Consumer : client)

Πάροχος: διακομιστής (Service Provider: server)

Χρήστης : διαχειριστής πόρων (User: resource owner)

Consumer Key and Secret: πιστοποιητικά χρήστη
(Consumer Key and Secret: client credentials)

Request Token and Secret: παροδικά πιστοποιητικά
(Request Token and Secret: temporary credentials)

Access Token and Secret: πιστοποιητικά τεκμηρίων
(Access Token and Secret: token credentials)

<u>Διαδικασία-Παράδειγμα</u>

Ο Γιώργος (διαχειριστής πόρων) πρόσφατα ανέβασε κάποιες φωτογραφίες στο προσωπικό του φάκελο στο "photos.example.net" (server) και θα ήθελε να τις τυπώσει από τον ιστόχωρο "printer.example.com" (client). Πρακτικά θα κάνει εγγραφή με το λογαριασμό του στο "photos.example.net" αλλά επειδή δε θέλει να μοιραστεί τους κωδικούς του με το "printer.example.com" το "printer.example.com" θα ζητήσει κάποια πιστοποιητικά χρήστη από το "photos.example.net" .

Client Identifier: dpf43f3p2l4k3l03

Client Shared-Secret: kd94hf93k423kf44

Το "printer.example.com" θα χρησιμοποιήσει τη μεθοδο πιστοποίησης "HMAC-SHA1" όπως ορίζει το "photos.example.net".

Temporary Credential Request

https://photos.example.net/initiate

Resource Owner Authorization URI:

https://photos.example.net/authorize

Token Request URI:

https://photos.example.net/token

Προτού ζητηθεί άδεια χειρισμού φωτογραφιών από το Γιώργο πρέπει να δημιουργηθούν παροδικά πιστοποιητικά με το "photos.example.net". Ακολουθώντας το πρωτόκολλο HTTPS (RFC2818) θα γίνει αίτηση στο διακομιστή

```
POST /initiate HTTP/1.1
     Host: photos.example.net
     Authorization: OAuth realm="Photos",
        oauth_consumer_key="dpf43f3p2l4k3l03",
        oauth_signature_method="HMAC-SHA1",
        oauth_timestamp="137131200",
        oauth_nonce="wIjqoS",

oauth_callback="http%3A%2F%2Fprinter.example.com%2Fready",
        oauth_signature="74KNZJeDHnMBp0EMJ9ZHt%2FXKycU%3D"
```

Ο διακομιστής πιστοποιεί το αίτημα και απαντά με παροδικά πιστοποιητικά σε μια απάντηση HTTP

```
HTTP/1.1 200 OK
     Content-Type: application/x-www-form-urlencoded


oauth_token=hh5s93j4hdidpola&oauth_token_secret=hdhd0
244k9j7ao03&
     oauth_callback_confirmed=true
```

Ο χρήστης ζητά τα δικαιώματα από το Γιώργο

```
https://photos.example.net/authorize?oauth_token=hh5s93j4
hdidpola
```

Ο διακομιστής ζητά από το Γιώργο να κάνει εγγραφή στο σύστημα και να εγκρίνει τα δικαιώματα που έχει ζητήσει ο "printer.example.com".Ο Γιώργος εγκρίνει τα δικαιώματα και επιστρέφουμε στο προηγούμενο αίτημα μέσω του callback-URI.

```
http://printer.example.com/ready?

oauth_token=hh5s93j4hdidpola&oauth_verifier=hfdp7dh39dks9
884
```

Το link αυτό επαληθεύει ότι έχουν δοθεί τα δικαιώματα και ο χρήστης ζητά πιστοποιητικά τεκμηρίων με τα ήδη υπάρχοντα παροδικά πιστοποιητικά μέσω ενός ασφαλούς καναλιού (secure Transport Layer Seurity (TLS)).

```
POST /token HTTP/1.1
     Host: photos.example.net
     Authorization: OAuth realm="Photos",
        oauth_consumer_key="dpf43f3p2l4k3l03",
        oauth_token="hh5s93j4hdidpola",
        oauth_signature_method="HMAC-SHA1",
        oauth_timestamp="137131201",
        oauth_nonce="walatlh",
        oauth_verifier="hfdp7dh39dks9884",
        oauth_signature="gKgrFCywp7rO0OXSjdot%2FIHF7IU%3D"
```

Ο διακομιστής επαληθεύει το αίτημα και απαντά με πιστοποιητικά τεκμηρίων σε HTTP

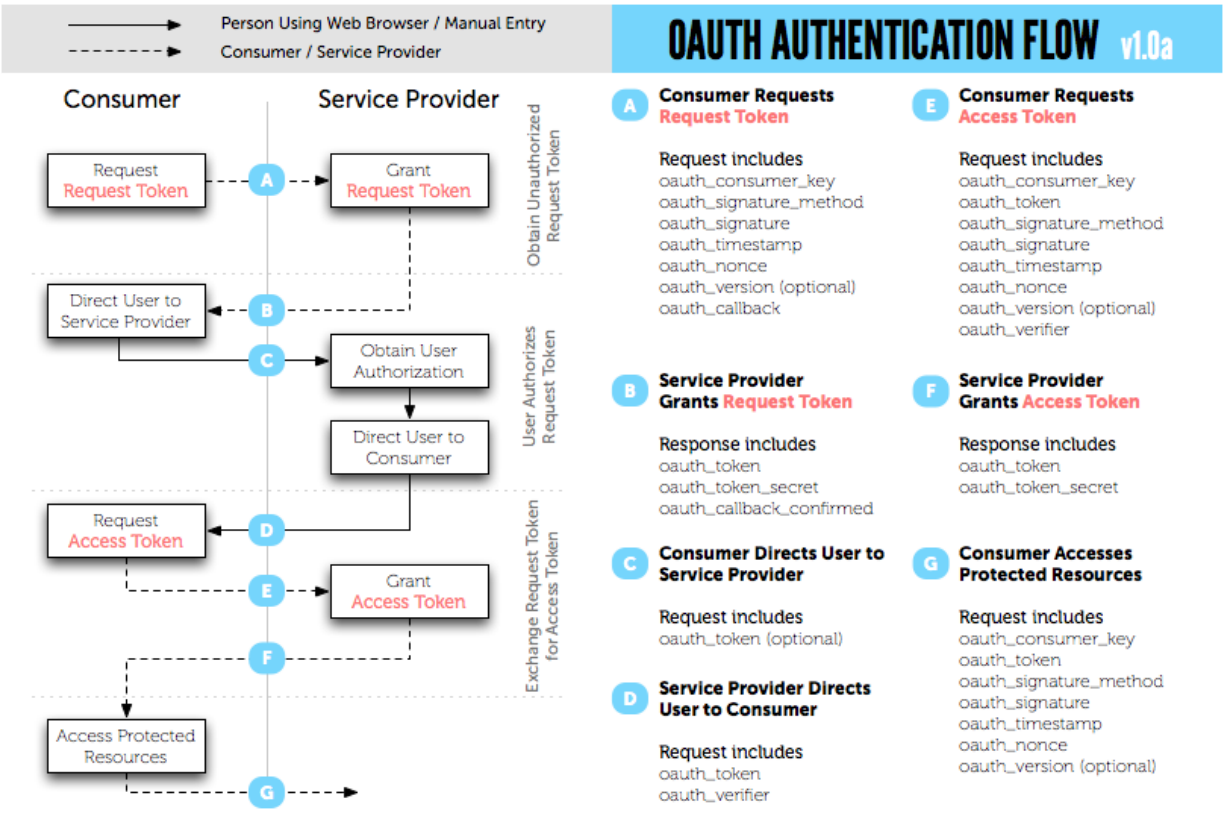
```
HTTP/1.1 200 OK
     Content-Type: application/x-www-form-urlencoded


oauth_token=nnch734d00sl2jdk&oauth_token_secret=pfkkdhi9sl
3r4s00
```

Με τα πιστοποιητικά τεκμηριών ο χρήστης μπορεί πλέον να πάρει τις προσωπικές φωτογραφίες του Γιώργου

```
GET /photos?file=vacation.jpg&size=original HTTP/1.1
     Host: photos.example.net
     Authorization: OAuth realm="Photos",
        oauth_consumer_key="dpf43f3p2l4k3l03",
        oauth_token="nnch734d00sl2jdk",
        oauth_signature_method="HMAC-SHA1",
        oauth_timestamp="137131202",
        oauth_nonce="chapoH",
        oauth_signature="MdpQcU8iPSUjWoN%2FUDMsK2sui9I%3D"
```

Ο χρήστης μπορεί να συνεχίσει να παίρνει φωτογραφίες από το φάκελο του Γιώργου μέχρι αυτός να αποσύρει τα δικαιώματα που έχει παραχωρήσει.

Στην Εικόνα 7 βλέπουμε σχηματικά τη διαδικασία

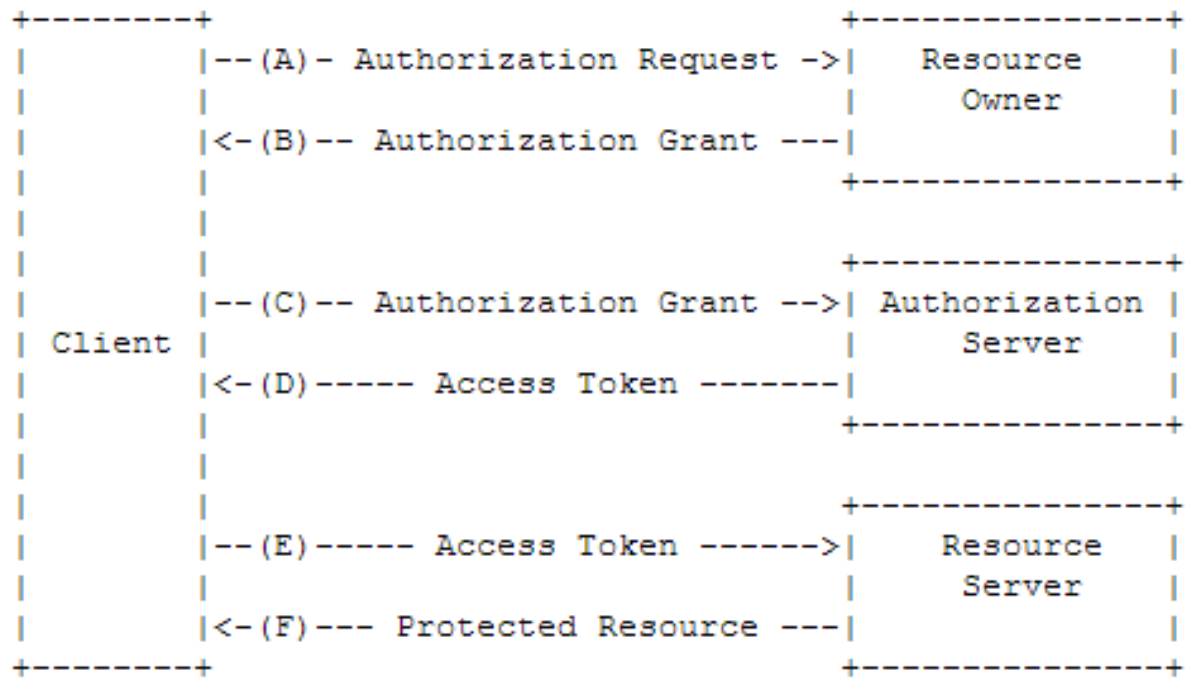


Εικόνα 7. OAuth authentication flow

3.6.2. OAuth 2.0

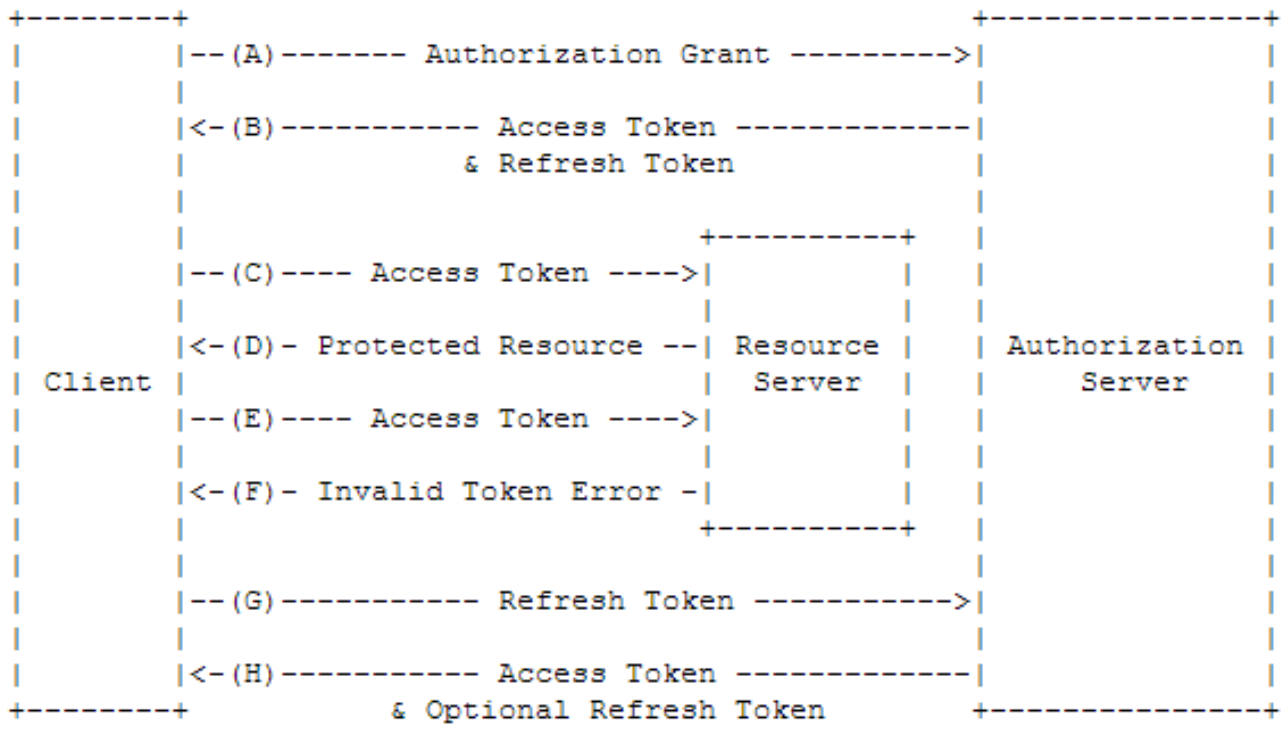
Το OAuth2 είναι η επόμενη έκδοση του OAuth και εστιάζει στην απλότητα των συνδέσεων. Δεν υποστηρίζει την υπογραφή, τη κρυπτογράφηση ή τη σύνδεση σε ασφαλή κανάλια . Απαντά αποκλειστικά σε SSL και παρέχει κάποιο επίπεδο εμπιστευτικότητας και εξουσιοδότησης. Τα πιστοποιητικά του χρειάζονται ανανέωση γιατί διαρκούν για μικρό χρονικό διάστημα. Στο OAuth2 υπάρχει διαχωρισμός ανάμεσα στο διακομιστή που χειρίζεται τις αιτήσεις OAuth και στο διακομιστή για εξουσιοδότηση. Δουλεύει περισσότερο σα πλαίσιο παρά ώς πρωτόκολλο. Το OAuth2 περιγράφεται από το πρωτόκολλο RFC6749.

Στην Εικόνα 8 περιγράφεται σχηματικά η δομή του πρωτοκόλλου.



Εικόνα 8 OAuth2 ροή πρωτόκολλου.

Στην Εικόνα 9 περιγράφεται η ανανέωση των πιστοποιητικών.



Εικόνα 9 Ανανέωση των πιστοποιητικών που έχουν λήξει

3.7. Εξουσιοδότηση στο LinkedIn

Για τη δημιουργία της εφαρμογής χρειάστηκε να πιστοποιήσουμε την εφαρμογή μας από το Linkedin και από το χρήστη για να μπορούμε να πάρουμε δεδομένα από το προφίλ του χρήστη και να τα επεξεργάστούμε με την εφαρμογή. Για το σκοπό αυτό μέσω του LinkedIn developers δηλώσαμε την εφαρμογή μας και πήραμε τα απαραίτητα κλειδιά OAuth (consumer_key, consumer_secret) . Στη συνέχεια μέσω της κλάσης LinkedInImpl.java που βρίσκεται στις βιβλιοθήκες μας παίρνουμε τα απαραίτητα πιστοποιητικά στέλνοντας τα αιτήματα μας στους συνδέσμους που ακολουθούν όπως ορίζει το πρωτόκολλο OAuth 1. 0.

OAUTH_REQUEST_TOKEN_URL:

https://api.linkedin.com/uas/oauth/requestToken

OAUTH_AUTHORIZATION_URL:

https://api.linkedin.com/uas/oauth/authenticate

OAUTH_ACCESS_TOKEN_URL:

https://api.linkedin.com/uas/oauth/accessToken

3.8. Άδειες

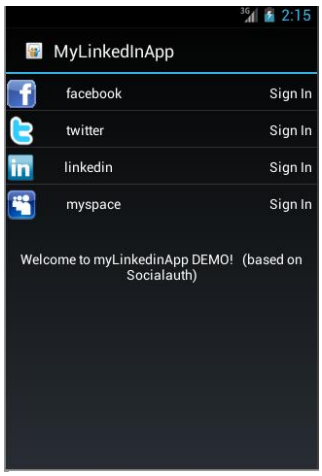
Αξίζει να αναφερθεί ότι στο LinkedIn υπάρχουν πολλά πεδία που παρότι παρέχονται οι αντίστοιχες άδειες από το χρήστη και τηρούνται όλα τα πρωτοκόλλα το κοινωνικό δίκτυο δεν επιτρέπει τη πρόσβαση σε αυτά σε κανένα διαχειριστή. Για παράδειγμα οι δεξιότητες (skills) των επαφών ενός χρήστη ή τα e-mail τους ή η ημερομηνία γεννησής τους καθώς και πολλά άλλα δε παρέχονται από το σύστημα για να γίνει η περαιτέρω επεξεργασία τους με την εφαρμογή μας. Γιαυτό το λόγο και στα πλαίσια της εφαρμογής παρουσιάζονται μόνο όσα πεδία προσφέρονται

4. Αναλυτική περιγραφή της εφαρμογής

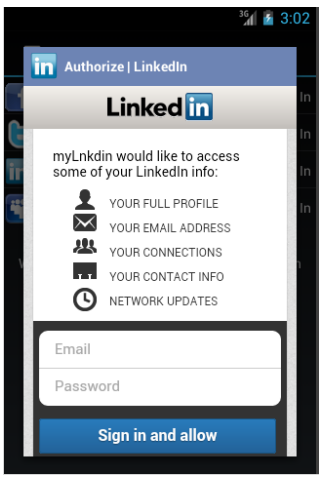
Ακολουθεί μία αναλυτική περιγραφή των κλάσεων της εφαρμογής για μεγαλύτερη κατανοήση.

4.1. Χειρισμός εφαρμογής

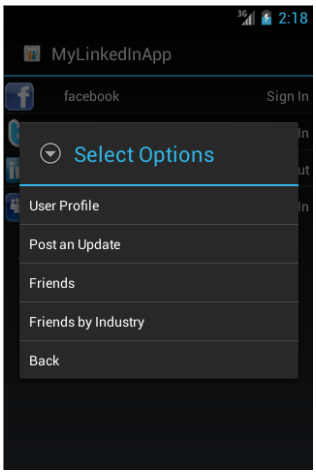
Όταν ανοίξει η εφαρμογή αρχικά παρουσιάζονται τα διαθέσιμα κοινωνικά δίκτυα προς σύνδεση (Εικόνα 10) στη περίπτωσή μας επιλέγουμε το LinkedIn και ακολουθεί μία οθόνη όπου ο χρήστης εγγράφεται στο κοινωνικό δίκτυο χρησιμοποιώντας τους κωδικούς του και παράλληλα παρέχει τα απαραίτητα δικαιώματα στην εφαρμογή μας (εικόνα 11). Στη συνέχεια εμφανίζεται ένα παράθυρο επιλογών στο οποίο ο χρήστης επιλέγει τι θελει να προβληθεί, το προφίλ του, να γράψει κάτι στο προφίλ του, να δει τη λίστα των επαφών του, και τους φίλους του κατηγοριοποιημένους με βάση το επάγγελμα, επίσης υπάρχει επιλογή για επιστροφή στη προηγούμενη οθόνη.



Εικόνα 10 Αρχική Οθόνη Εφαρμογής

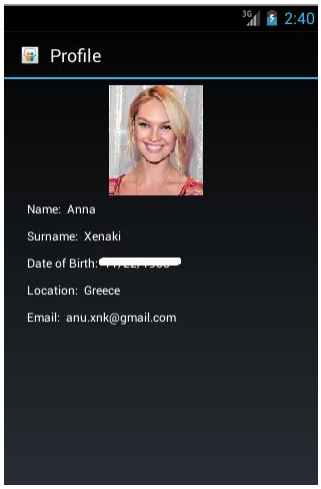


Εικόνα 11. Εγγραφή χρήστη



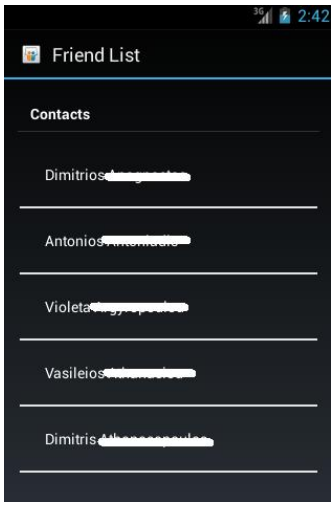
Εικόνα 12. Παράθυρο επιλογών

Αν ο χρήστης επιλέξει να προβληθεί το προφίλ του, εμφανίζεται η εικόνα 13. Για λόγους προστασίας προσωπικών δεδομένων έχουν αλλαχθεί τα πρόσωπα στις φωτογραφίες και έχουν καλυφθεί τα προσωπικά στοιχεία.



Εικόνα 13. Προφίλ χρήστη

Αν ο χρήστης επιλέξει να δει τη λίστα επαφών του θα εμφανιστεί η εικόνα 14.



Εικόνα 14. Λίστα επαφών

Εάν ο χρήστης επιλέξει να δει τις επαφές του κατηγοριοποιημένες με βάση το επάγγελμα, θα εμφανιστεί αρχικά μία οθόνη (εικόνα 15) με τις κατηγορίες επαγγελμάτων χωρισμένες σε γενικότερες κατηγορίες. Αυτές είναι :

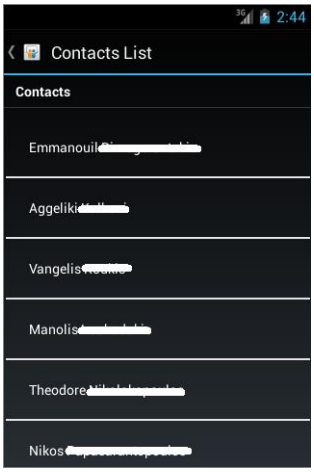
- τέχνη (art)
- γεωργία (agriculture)

- κατασκευές (constructions)
- εταιρεία (corporation)
- εκπαίδευση (education)
- χρηματοοικονομικά (finance)
- αγαθά (goods)
- κυβέρνηση (government)
- υγεία (health)
- νομοθεσία (legislation)
- παραγωγή (manufacturing)
- μέσα ενημέρωσης (media)
- οργάνωση (organisation)
- αναψυχή (recreation)
- υπηρεσίες (services)
- τεχνολογία (technology)
- μεταφορά (tranportation)
- απροσδιόριστο (undefined)

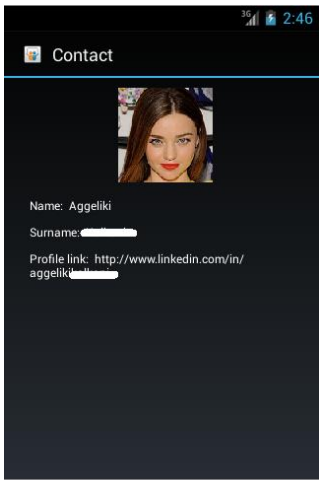
Στη συνέχεια αφού επιλέξει σε ποια από αυτές θέλει να δει αν υπαρχούν επαφές του θα εμφανιστεί μια οθόνη (εικόναι 16) με τα επαγγέλματα που ανήκουν στο συγκεκριμένο κλάδο εργασίας. Στις παρανθέσεις εμφανίζεται ο αριθμός των επαφών του χρήστη που ανήκουν στη κάθε κατηγορία. Τα επαγγέλματα ορίζονται από το LinkedIn και φτάνουν συνολικά τα 143. Με την επιλογή επαγγέλματος θα εμφανιστεί μία λίστα από επαφές με το συγκεκριμένο επάγγελμα (εικόνα 17). Ο χρήστης μπορεί να επιλέξει όποια από αυτές επιθυμεί για να δει το αναλυτικό τους προφίλ (εικόνα 18).

Εικόνα 15. Κατηγορίες επαγγελμάτων

Εικόνα 16. Επαγγέλματα



Εικόνα 17. Λίστα επαφών



Εικόνα 18. Προφίλ επαφής

4.2. Υλοποίηση

Στην ενότητα αυτή θα δώσουμε αρκετές πληροφορίες σχετικά με το πως έχουν υλοποιηθεί τα παραπάνω τμήματα. Θα εξηγήσουμε κάποια συγκεκριμένα κομμάτια κώδικα που είναι απαραίτητα για τη κατανόηση της λειτουργίας της εφαρμογής (και όχι ολόκληρες τις κλάσεις).

CustomUI.java

```
public List<Contact2> getContactList2()
    {
        try
        {
```

```
            contactsList2 = new contactTask2().execute().get();
        }
        catch (InterruptedException e)
        {
            e.printStackTrace();
        }
        catch (ExecutionException e)
        {
            e.printStackTrace();
        }

        return contactsList2;
    }
    private class contactTask2 extends AsyncTask<Void, Void, List<Contact2>>
{

        protected List<Contact2> doInBackground(Void... params) {
            try
            {
                List<Contact2> contactsMap =
LinkedInImpl2.getContactList2(adapter.getCurrentProvider());
                Log.d("SocialAuthAdapter", "Received Contact list");
                return contactsMap;
            }
            catch (Exception e)
            {
                e.printStackTrace();
                dialogListener.onError(new SocialAuthError("Contact List not
Received", e));
                return null;
            }
        }
    }
```

Για να ανακτήσουμε τη λίστα επαφών με βάση το επάγγελμα χρησιμοποιούμε τις πιστοποιήσεις που είχαμε ήδη αποκτήσει με τον adapter και εκτελούμε το αίτημα ασύγχρονα στο παρασκήνιο για να μη κολλάει η εφαρμογή κατά την αίτηση αυτή.

```
public void Events(int position , String provider)
    {

        switch(position)
        {
        case 0 :
        {

            profileMap =  adapter.getUserProfile();
```

```
            if (provider.equalsIgnoreCase("linkedin")){
                Intent intent = new Intent(CustomUI.this,
ProfileActivity.class);
                intent.putExtra("contacts", profileMap);
                startActivity(intent);
            }
            break;
        }

        case 1 :
  {

            adapter.updateStatus("SocialAuth Android" +
System.currentTimeMillis());
            Toast.makeText(CustomUI.this, "Message posted on " + provider,
Toast.LENGTH_SHORT).show();

            break;
        }

        case 2 :
        {

            List<Contact> contactsList = adapter.getContactList();
            Intent contactintent=new
Intent(getApplicationContext(),ContactList.class);

            contactintent.putExtra("contacts", (Serializable)contactsList);
            startActivity(contactintent);

            break;
        }

        case 3 :
        {

            if (provider.equalsIgnoreCase("linkedin")){
                try {

                    List<Contact2> contactsList = getContactList2();
                    IndustryLog indLog= new IndustryLog();

                    if (contactsList != null && contactsList.size() > 0)
                    {
                        for (Contact2 p : contactsList)
                        {
                            indLog.doSearchandAdd(p.getIndustry(), p);
                        }
                    }
```

```
                    Intent myintent = new
Intent(getApplicationContext(),GroupsUI.class);

                    myintent.putExtra("log", indLog);

                    myintent.putExtra("indList", (Serializable)indList);
                    startActivity(myintent);
                    break;

                } catch (Exception e) {
                    e.printStackTrace();
                }


            }else{
                dialog.dismiss();
                break;}
        }

        case 4:
        {
            if (provider.equalsIgnoreCase("linkedin")){
                dialog.dismiss();
                break;
            }
        }
```

Στο παράθυρο που βγαίνει με τις επιλογές (εικόνα 12) έχουμε ορίσει κάθε επιλογή να δημιουργεί ένα event ανάλογα με το τι πατήθηκε ξεκινώντας από το 0. Στη πρώτη επιλογή αρχικά ανακτούμε με τα αναγνωρισμένα δικαιώματα του adapter το προφίλ του χρήστη και στη συνέχεια ανοίγουμε καινούργια οθόνη με τη χρήση του Intent με πατρικό παράθυρο αυτό του CustomUI. Το καινούργιο παράθυρο θα παρουσιάζεται με βάση τα όσα γράφονται στη κλάση ProfileActivity.class. Επίσης για να είναι δυνατή η προβολή των δεδομένων πρέπει να περάσουμε τα δεδομένα στο παράθυρο που θα ανοίξει και για να το πετύχουμε αυτό τα περνάμε με τη συνάρτηση putExtras. Η επιλογή 1 αποτελεί επιλογή της βιβλιοθήκης του socialauth και δεν έχουμε επέμβει σε αυτή. Στην επιλογή 2 επαναλαμβάνουμε τη διαδικασία της επιλογής 0 χρησιμοποιώντας τον adapter του παρόχου του κοινωνικού δικτύου, που έχει πιστοποιηθεί, (στη περίπτωση μας LinkedIn) ζητούμε τη λίστα των επαφών του χρήστη (getContactList()) και αντίστοιχα τη περνάμε στο επόμενο παράθυρο (που δημιουργείται από τη κλάση ContactList.class) που δημιουργούμε με πατρικό

παράθυρο το CustomUI και πάλι. Στην επιλογή 3 επαναλαμβάνουμε τη διαδικασία της επιλογής 2 μόνο όταν ο πάροχος είναι το LinkedIn. Αποκτούμε τη λίστα των επαφών την οποία μετά περνάμε σε ένα hash table (indLog) με κλειδί το όνομα του επαγγέλματος που έχουν δηλώσει. Σε αντιστοιχία με πριν ανοίγουμε καινούργιο παράθυρο (Που δημιουργείται από τη κλάση GroupsUI.class) και περνάμε ως δεδομένο το hash table . Αν ο πάροχος δεν είναι το LinkedIn δεν εκτελείται καμία ενέργια. Στη περίπτωση 4 αν ο πάροχος είναι το LinkedIn επιστρέφει στο προηγούμενο παράθυρο. Κάποιες επιλογές έχουν ρυθμιστεί ειδικά για το LinkedIn αυτό συμβαίνει γιατί επιλέξαμε να παραμείνουν στην εφαρμογή οι υπόλοιποι πάροχοι κοινωνικών δικτύων που υπήρχαν και να διατηρήσουν τις ήδη υπάρχουσες επιλογές τους.

ProfileActivity.java

```
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_profile);

        ProfileMap= (Profile) getIntent().getSerializableExtra("contacts");

        // Loader image - will be shown before loading image
        loader = R.drawable.no_contact;

        // Imageview to show
        image = (ImageView) findViewById(R.id.myimage);

        name = (TextView) findViewById(R.id.name);
        lastname = (TextView) findViewById(R.id.lastname);
        dob=(TextView) findViewById(R.id.dob);
        email = (TextView) findViewById(R.id.email);
        location=(TextView)findViewById(R.id.location);

        name.setText("Name:  "+ProfileMap.getFirstName());
        lastname.setText("Surname:  "+ProfileMap.getLastName());
        email.setText("Email:  "+ProfileMap.getEmail());
        dob.setText("Country:  "+ ProfileMap.getDob());
        location.setText("Location:  "+ProfileMap.getLocation());
          // Image url
        image_url = ProfileMap.getProfileImageURL();
        // ImageLoader class instance
        imgLoader = new ImageLoader(getApplicationContext());

        imgLoader.DisplayImage(image_url, loader, image);
}
```

Για τη δημιουργία του προφίλ του χρήστη αρχικά διαλέγουμε το τρόπο που θα εμφανιστούν τα δεδομένα στην οθόνη (setContentView) με βάση το xml αρχείο activity_profile, στη συνέχεια αποθηκεύουμε τα δεδομένα που ήρθαν από το προήγουμενο παράθυρο στο ProfileMap συνδέουμε τις μεταβλητές της κλάσης με τα πεδία που ορίσαμε στο layout και περνάμε τιμές στα πεδία του παράθυρου. Εδω αξίζει να αναφερθεί ότι επειδή υπάρχει εικόνα που θα φορτώνει μαζί με το παράθυρο ορίσαμε κάποια προεπιλογή εικόνας για να προβάλλεται μέχρι να γίνει η φόρτωση (loader= R.drawable.no_contact η εικόνα βρίσκεται στο φάκελο drawable και ονομάζεται no_contact)

Παρακάτω θα εξηγήσουμε πως φορτώνεται η εικόνα στο προφίλ. Αρχικά πρέπει να εξηγήσουμε ότι για να φορτωθεί μία εικόνα δεν αρκεί το URL της, πρέπει να τη φορτώσουμε πρώτα σε μία προσωρινή θέση στη μνήμη (cache) και μετά να την εμφανίσουμε. Γιαυτό και αρχικά σχεδιάστηκε ο τρόπος που θα την αποθηκεύσουμε.

MemoryCache.java

```
private Map<String, SoftReference<Bitmap>>
cache=Collections.synchronizedMap(new HashMap<String,
SoftReference<Bitmap>>());

   public Bitmap get(String id){
       if(!cache.containsKey(id))
           return null;
       SoftReference<Bitmap> ref=cache.get(id);
       return ref.get();
   }

   public void put(String id, Bitmap bitmap){
       cache.put(id, new SoftReference<Bitmap>(bitmap));
   }

   public void clear() {
       cache.clear();
   }
```

Εδώ αποθηκεύουμε την εικόνα (μέθοδος put) με κάποιο αναγωριστικό id σε ένα hash table για να μπορούμε να την ανακτήσουμε πιο γρήγορα από τη μνήμη.

FileCache.java

```
    private File cacheDir;
```

```
    public FileCache(Context context){
        //Find the dir to save cached images
        if
(android.os.Environment.getExternalStorageState().equals(android.os.Environme
nt.MEDIA_MOUNTED))
            cacheDir=new
File(android.os.Environment.getExternalStorageDirectory(),"TempImages");
        else
            cacheDir=context.getCacheDir();
        if(!cacheDir.exists())
            cacheDir.mkdirs();
    }

    public File getFile(String url){
        String filename=String.valueOf(url.hashCode());
        File f = new File(cacheDir, filename);
        return f;

    }

    public void clear(){
        File[] files=cacheDir.listFiles();
        if(files==null)
            return;
        for(File f:files)
            f.delete();
    }
```

Εδώ δεσμεύουμε χώρο στη μνήμη του android για αποθήκευση εικόνων (temp images). Η getFile μας δίνει τη δυνατότητα να βρούμε στη μνήμη το φάκελο και κατ'επέκταση την εικόνα μας. Με τη μέθοδο clear απελευθερώνουμε τη μνήμη.

ImageLoader.java

```
MemoryCache memoryCache=new MemoryCache();
    FileCache fileCache;
    private Map<ImageView, String> imageViews=Collections.synchronizedMap(new
WeakHashMap<ImageView, String>());
    ExecutorService executorService;
```

Αρχικά δεσμεύουμε τους αναγκαίους πόρους.

```
public ImageLoader(Context context){
        fileCache=new FileCache(context);
        executorService=Executors.newFixedThreadPool(5);
    }
```

Έπειτα βρίσκουμε που θα αποθηκεύσουμε τις εικόνες, το χώρο που η εφαρμογή μας δεσμεύει στη μνήμη του κινητού και φτιάχνουμε εκεί μέσα το φάκελο για τις εικόνες. Στη συνέχεια ανοίγουμε threads για να μπορούμε να πραγματοποιήσουμε όλες τις εργασίες που είναι απαραίτητες.

```
int stub_id = android.R.drawable.ic_dialog_alert;
    public void DisplayImage(String url, int loader, ImageView imageView)
    {
        stub_id = loader;
        imageViews.put(imageView, url);
        Bitmap bitmap=memoryCache.get(url);
        if(bitmap!=null)
            imageView.setImageBitmap(bitmap);
        else
        {
            queuePhoto(url, imageView);
            imageView.setImageResource(loader);
        }
    }
```

Φορτώνουμε μία εικόνα stub_id σε περίπτωση που δεν ανοίξει η εικόνα που θέλουμε να εμφανιστεί (π.χ. αν δε λειτουργεί ο σύνδεσμος στον οποίο βρίσκεται). Ορίζουμε τη προεπιλεγμένη θέση εμφάνισης (imageView) και το σύνδεσμό της και δεσμεύουμε μνήμη για αυτή στο hash table. Η setImageBitmap ορίζει στη θέση εμφάνισης ότι θα εμφανιστεί εικόνα Bitmap η μέθοδος αυτή υπάρχει στη βιβλιοθήκη ImageView του android.

```
  private void queuePhoto(String url, ImageView imageView)
    {
        PhotoToLoad p=new PhotoToLoad(url, imageView);
        executorService.submit(new PhotosLoader(p));
    }
```

Η μέθοδος αυτή φροντίζει ώστε να γίνει η φόρτωση της εικόνας.

```
private class PhotoToLoad
    {
        public String url;
        public ImageView imageView;
        public PhotoToLoad(String u, ImageView i){
            url=u;
            imageView=i;
        }
    }
```

Ορίζουμε μία δομή που θα περιέχει το url και τη θέση εμφάνισης.

```
    class PhotosLoader implements Runnable {
        PhotoToLoad photoToLoad;
        PhotosLoader(PhotoToLoad photoToLoad){
            this.photoToLoad=photoToLoad;
        }

        @Override
        public void run() {
            if(imageViewReused(photoToLoad))
```

```
            return;
        Bitmap bmp=getBitmap(photoToLoad.url);
        memoryCache.put(photoToLoad.url, bmp);
        if(imageViewReused(photoToLoad))
            return;
        BitmapDisplayer bd=new BitmapDisplayer(bmp, photoToLoad);
        Activity a=(Activity)photoToLoad.imageView.getContext();
        a.runOnUiThread(bd);
    }
}
```

Παίρνουμε την εικόνα από το url και την αποθηκεύουμε στο hash table. Εδώ κρίνουμε αν η εικόνα θα χρειαστεί επεξεργασία πρωτού προσπαθήσουμε να την εμφανίσουμε. Αν ναι η επεξεργασία γίνεται σε κάποιο νήμα (thread) που ορίσαμε στην αρχή.

```
boolean imageViewReused(PhotoToLoad photoToLoad){
    String tag=imageViews.get(photoToLoad.imageView);
    if(tag==null || !tag.equals(photoToLoad.url))
        return true;
    return false;
}
```

Ελέγχουμε αν περάστηκε η εικόνα πάνω από την εικόνα που είχαμε ορίσει πριν αρχίσει η φόρτωση του url.

```
public void clearCache() {

    memoryCache.clear();
    fileCache.clear();
}
```

Εδώ καθαρίζουμε τη μνήμη για να μην υπερφορτώνεται η μνήμη της εφαρμογής κατά τη διάρκεια περιήγησης στα προφίλ των χρηστών.

```
private Bitmap getBitmap(String url)
    {
        File f=fileCache.getFile(url);

        //from SD cache
        Bitmap b = decodeFile(f);
        if(b!=null)
            return b;

        //from web
        try {
            Bitmap bitmap=null;
            URL imageUrl = new URL(url);
            HttpURLConnection conn =
(HttpURLConnection)imageUrl.openConnection();
            conn.setConnectTimeout(30000);
            conn.setReadTimeout(30000);
```

```
            conn.setInstanceFollowRedirects(true);
            InputStream is=conn.getInputStream();
            OutputStream os = new FileOutputStream(f);
            CopyStream(is, os);
            os.close();
            bitmap = decodeFile(f);
            return bitmap;
        } catch (Throwable ex){
           ex.printStackTrace();
           if (ex instanceof OutOfMemoryError)
                memoryCache.clear();
           return null;
        }
    }
```

Με αυτή τη μέθοδο ανοίγουμε το url της εικόνας και κατεβάζουμε για πρώτη φορά την εικόνα στο κινητό.

Η παρακάτω μέθοδος τροποποιεί το μέγεθος της εικόνας ώστε να περιορίζεται η δέσμευση μνήμης.

```
    private Bitmap decodeFile(File f){
        try {
            //decode image size
            BitmapFactory.Options o = new BitmapFactory.Options();
            o.inJustDecodeBounds = true;
            BitmapFactory.decodeStream(new FileInputStream(f),null,o);

            //Find the correct scale value. It should be the power of 2.
            final int REQUIRED_SIZE=70;
            int width_tmp=o.outWidth, height_tmp=o.outHeight;
            int scale=1;
            while(true){
                if(width_tmp/2<REQUIRED_SIZE || height_tmp/2<REQUIRED_SIZE)
                    break;
                width_tmp/=2;
                height_tmp/=2;
                scale*=2;
            }

            //decode with inSampleSize
            BitmapFactory.Options o2 = new BitmapFactory.Options();
            o2.inSampleSize=scale;
            return BitmapFactory.decodeStream(new FileInputStream(f), null,
o2);
        } catch (FileNotFoundException e) {}
        return null;
    }

  class BitmapDisplayer implements Runnable
    {
        Bitmap bitmap;
        PhotoToLoad photoToLoad;
```

```
        public BitmapDisplayer(Bitmap b, PhotoToLoad p){
            bitmap=b;
            photoToLoad=p;
            }
        public void run()
        {
            if(imageViewReused(photoToLoad))
                return;
            if(bitmap!=null)
                photoToLoad.imageView.setImageBitmap(bitmap);
            else
                photoToLoad.imageView.setImageResource(stub_id);
        }
    }
```

Εδώ μετά τη φόρτωση της εικόνας επιλέγουμε να τη περάσουμε ή αν δεν έχει ολοκληρωθεί η φόρτωση και η μορφοποίηση της να περαστεί η εικόνα που έχουμε ορίσει να εμφανίζεται όσο αυτή φορτώνεται (stub_id).

Για την εμφάνιση της λίστας επαφών στη δεύτερη επιλογή (εικόνα 14) δημιουργήθηκε η κλάση ContactList η οποία επεκτίνει τη κλάση του android ListActivity.

```
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_contact_list);

        ContactList= (ArrayList<Contact>)
      getIntent().getSerializableExtra("contacts");

        setListAdapter(new ContactListAdapter(this,ContactList));

    }
    @Override
    protected void onListItemClick(ListView l, View v, int position, long id)
{

    }
```

Ορίσαμε την μορφοποίηση του παράθυρου με τη μέθοδο setContentView από το αρχείο activity_contact_list. Τη λίστα τη παραλάβαμε από το προηγούμενο παράθυρο με τη μέθοδο getIntent. Τη παρουσίαση της λίστας αναθέσαμε σε έναν adapter.

```
public class ContactListAdapter  extends ArrayAdapter<Contact>{
        private final List<Contact> objects;
        private final Context context;

        public ContactListAdapter(Context context,List<Contact> objects) {
             super(context,R.layout.contact_list, objects);
```

```
            this.context=context;
            this.objects=objects;

        }
```

Ορίζουμε σε ποιό παράθυρο θα εμφανιστεί η λίστα και αποθηκεύουμε τις μεταβλητές που χρειαζόμαστε.

```
public View getView(int position, View convertView, ViewGroup parent) {
            LayoutInflater  inflater = (LayoutInflater)
context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);

  View rowView = inflater.inflate(R.layout.contact_list, parent, false);
  TextView nameView = (TextView)rowView.findViewById(R.id.contactFirstName);

  Contact u=objects.get(position);
  nameView.setText(" "+u.getFirstName()+" " +u.getLastName());
                  return rowView;
        }
```

Παραπάνω ορίζουμε τι θα περιέχει κάθε γραμμή της λίστας.

GroupsUI.java

```
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.groups_ui);

        String[] groups =getResources().getStringArray(R.array.groups_array);

        setListAdapter(new ArrayAdapter<String>(this,
android.R.layout.simple_list_item_1,groups));

        indlog=(IndustryLog) getIntent().getSerializableExtra("log");
        indList=(HashMap<String,
List<String>>)getIntent().getSerializableExtra("indList");


        for(int pass=0; pass<groups.length;pass++){
            String listItem=groups[pass];
            if (indList.containsKey(listItem)){
            List<String> thisList=indList.get(listItem) ;
            int counter=0;
            for(String temp: thisList){
                counter+=indlog.doSearchandGet(temp).size();
            }
            String says=listItem+"   "+"(" +Integer.toString(counter)+ ")" ;
            groups[pass]=says;
            }
            }

    }
@Override
```

```
    protected void onListItemClick(ListView l, View v, int position, long id)
{
        //get selected items
        pressed= (String) getListAdapter().getItem(position);
        String pressed1[]= pressed.split("\\(");
        pressed= pressed1[0];
        pressed=pressed.trim();
        Log.i("list test", "I Clicked on element " + pressed + " and it
worked!");

        Intent myintent = new
Intent(getApplicationContext(),IndustriesUI.class);

        myintent.putExtra("log", indlog);
        myintent.putExtra("indList", (Serializable)indList);
        myintent.putExtra("pressed",pressed);
        startActivity(myintent);

    }
```

Για την εμφάνιση της λίστας με τις κατηγορίες ανασύρουμε τη λίστα από το styles.xml με τη μέθοδο getResources().getStringArray(). Για να φαίνεται δίπλα στη λίστα ο αριθμός των ατόμων που ανήκουν σε αυτή μετράμε από το hash table indLog που είχαμε δημιουργήσει στην αρχή πόσοι ανήκουν σε κάθε επάγγελμα που ανήκει σε αυτή τη κατηγορία. Οτάν πατηθεί κλίκ σε κάποια κατηγορία ανοίγουμε καινούργιο παράθυρο με τη λίστα των επαγγελμάτων που ανήκουν στη κατηγορία αυτή (IndustriesUI.class).

Στην IndustriesUI.java εμφανίζουμε τη λίστα με τα επαγγέλματα και δίπλα σε κάθε όνομα τον αριθμό των επαφών που ανήκουν στο επάγγελμα αυτό. Η διαδικασία που ακολουθήθηκε είναι ίδια με την παραπάνω.

Ανάλογα με το ποιο θα πατηθεί δημιουργείται ένα νέο παράθυρο με τη λίστα των επαφών ονομαστικά (IndustrycontactList.java). Με κλίκ πάνω σε κάποια επαφή εμφανίζεται το προφίλ της με τη βοήθεια της κλάσης IndustryContact.java.

Για να αποκτηθεί η λίστα με τις επαφές στο τμήμα κατηγοριοποίησης με βάση το επάγγελμα έπρεπε να ζητήσουμε παραπάνω πεδία από αυτά στην επιλογή 2. Γιαυτό το λόγο δημιουργήσαμε μία κλαση που θα ζητούσε τα δεδομένα αυτά από το LinkedIn και αφού έπαιρνε την απάντηση σε μορφή xml θα τα πρόσθετε

στη λίστα που χρησιμοποιήσαμε σε όλα τα επόμενα παράθυρα. Αυτή η κλάση είναι η LinkedInImpl2.java .

4.3. Άδειες Λογισμικού

Για το κώδικα της εφαρμογής χρειάστηκε να ληφθούν οι κατάλληλες άδειες λογισμικού (licenses)[3]. Όλες οι άδειες ελεύθερου λογισμικού συμμορφώνονται με το "Open Source Definition" το οποίο εν συντομία ορίζει ότι το λογισμικό μπορεί να χρησιμοποιηθεί, να τροποποιηθεί και να κοινοποιηθεί. Για το κωδικά μας επιλέξαμε την MIT License[4] η οποία ορίζει τα παρακάτω :

The MIT License (MIT)

Copyright (c) <year> <copyright holders>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

[3] http://opensource.org/licenses
[4] http://opensource.org/licenses/MIT

5. Παράρτημα Α

5.1. Κώδικας εφαρμογής

strings.xml

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

<resources>

    <string name="app_name">MyLinkedInApp </string>

    <string-array name="facebook_array">
        <item>User Profile</item>
        <item>Post Message</item>
        <item>Friends</item>
        <item>Back</item>
    </string-array>
    <string-array name="twitter_array">
        <item>User Profile</item>
        <item>Send Tweet</item>
        <item>Friends</item>
        <item>Back</item>
    </string-array>

    <string-array name="linkedin_array">
        <item>User Profile</item>
        <item>Post an Update</item>
        <item>Friends</item>
        <item>Friends by Industry</item>
        <item>Back</item>
    </string-array>
   <string-array name="myspace_array">
        <item>User Profile</item>
        <item>Post an Update</item>
        <item>Friends</item>
        <item>Back</item>
    </string-array>

    <string name="groups"> Group Categories </string>

    <string-array name="groups_array">
        <item>art</item>
        <item>agriculture</item>
        <item>constructions</item>
```

```
        <item>corporation</item>
        <item>education</item>
        <item>finance</item>
        <item>goods</item>
        <item>government</item>
        <item>health</item>
        <item>legislation</item>
        <item>manufacturing</item>
        <item>media</item>
        <item>organisation</item>
        <item>recreation</item>
        <item>services</item>
        <item>technology</item>
        <item>tranportation</item>
        <item>undefined</item>
    </string-array>

    <string name="industries">Industry Categories</string>
    <string name="title_activity_groups_ui">Industry Groups</string>
    <string name="action_settings">Settings</string>
    <string name="title_activity_industries_ui">Industry Categories</string>
    <string name="title_activity_industry_contact_list">Contacts
List</string>
    <string name="hello_world">Hello world!</string>
    <string name="title_activity_industry_contact">Contact</string>
    <string name="title_activity_profile">Profile</string>
    <string name="title_activity_contact_list">Friend List</string>
    <string name="title_activity_contact">ContactActivity</string>

</resources>
```

activity_contact_list.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
   >
    <LinearLayout
    android:orientation="horizontal"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:padding="7dp"
       >
```

```
    <TextView
                android:id="@+id/Contact"
                android:layout_height="wrap_content"
                android:layout_width="wrap_content"
                android:layout_gravity="center_vertical"
                android:textColor="#ffffff"
                android:text="Contacts"
                android:textSize="14sp"
                android:layout_marginLeft="5dp"
                android:textStyle="bold"
          />
    </LinearLayout>
    <View
       android:id="@+id/view01"
       android:layout_width="fill_parent"
       android:layout_height="1dp"
       android:background="#313437" />

 <ListView
       android:id="@android:id/list"
        android:layout_width="fill_parent"
      android:layout_height="fill_parent"
      android:layout_marginTop="10dp"
      android:layout_marginLeft="2dp"
      android:layout_marginRight="2dp"
      android:layout_marginBottom="2dp"
      android:divider="#e4e4e4"
      android:scrollingCache="false"
            android:dividerHeight="2px"
        />
</LinearLayout>
```

activity_industry_contact.xml

```
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/TableLayout1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".IndustryContact" >

     <ImageView
        android:id="@+id/myimage"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_margin="10dip" />
```

```
    <TextView
        android:id="@+id/name"
        android:layout_width="137dp"
        android:layout_height="wrap_content"
        android:layout_marginLeft="5dp"
        android:padding="5dp"
        android:textColor="#ffffff"
        android:textIsSelectable="true"
        android:textSize="12sp" />

    <TextView
        android:id="@+id/lastname"
        android:layout_width="137dp"
        android:layout_height="wrap_content"
        android:layout_marginLeft="5dp"
        android:padding="5dp"
        android:textColor="#ffffff"
        android:textIsSelectable="true"
        android:textSize="12sp" />


    <TextView
        android:id="@+id/profileURL"
       android:layout_width="137dp"
        android:layout_height="wrap_content"
        android:layout_marginLeft="5dp"
        android:padding="5dp"
        android:textColor="#ffffff"
        android:textIsSelectable="true"
        android:textSize="12sp" />

    <TextView
        android:id="@+id/industry"
        android:layout_width="137dp"
        android:layout_height="wrap_content"
        android:layout_marginLeft="5dp"
        android:padding="5dp"
        android:textColor="#ffffff"
        android:textIsSelectable="true"
        android:textSize="12sp" />

</TableLayout>
```

activity_profile.xml

```
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/TableLayout1"
    android:layout_width="match_parent"
```

```
android:layout_height="match_parent"
android:orientation="horizontal"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context=".ProfileActivity" >

<ImageView
    android:id="@+id/myimage"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_margin="10dip" />

<TextView
    android:id="@+id/name"
    android:layout_width="137dp"
    android:layout_height="wrap_content"
    android:layout_marginLeft="5dp"
    android:padding="5dp"
    android:textColor="#ffffff"
    android:textIsSelectable="true"
    android:textSize="12sp" />

<TextView
    android:id="@+id/lastname"
    android:layout_width="137dp"
    android:layout_height="wrap_content"
    android:layout_marginLeft="5dp"
    android:padding="5dp"
    android:textColor="#ffffff"
    android:textIsSelectable="true"
    android:textSize="12sp" />

<Space
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="fill_vertical" />

<TextView
    android:id="@+id/dob"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="5dp"
    android:padding="5dp"
    android:textColor="#ffffff"
    android:textIsSelectable="true"
    android:textSize="12sp" />

<TextView
    android:id="@+id/location"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="5dp"
        android:padding="5dp"
        android:textColor="#ffffff"
        android:textIsSelectable="true"
        android:textSize="12sp" />

    <TextView
        android:id="@+id/email"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="5dp"
        android:padding="5dp"
        android:textColor="#ffffff"
        android:textIsSelectable="true"
        android:textSize="12sp" />

</TableLayout>
```

contact_list.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".ContactList" >

    <TextView
        android:id="@+id/contactFirstName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:padding="5dp"
      android:textColor="#ffffff"
      android:textSize="14sp"  />


</LinearLayout>
```

contact2_list.xml

```
<LinearLayout
      xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
```

```
        android:orientation="vertical"
        >
        <TextView
              android:id="@+id/conName"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:padding="5dp"
        android:textColor="#000000"
        android:textSize="12sp"
              >
        </TextView>
        <TextView
              android:id="@+id/conEmail"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:padding="5dp"
        android:textColor="#333333"
        android:textSize="10sp"
        android:visibility="gone"
              >
        </TextView>
</LinearLayout>
```

groups_ui.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".GroupsUI" >


    <ListView
       android:id="@android:id/list"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:background="#000000"
        android:drawSelectorOnTop="false" />

    <TextView
        android:id="@android:id/empty"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#ffffff"
        android:text="No data" />
```

```
</LinearLayout>
```

industries_ui.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".IndustriesUI" >


    <ListView
        android:id="@android:id/list"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:background="#000000"
        android:drawSelectorOnTop="false" />

    <TextView
        android:id="@android:id/empty"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#ffffff"
        android:text="No data" />

</LinearLayout>
```

industry_contact_list.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".IndustryContactList" >

    <TextView
        android:id="@+id/contactFisrtName"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:padding="5dp"
      android:textColor="#ffffff"
      android:textSize="14sp"  />
    <TextView
            android:id="@+id/email"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:padding="5dp"
      android:textColor="#ffffff"
      android:textSize="12sp"  />

</LinearLayout>
```

industry_contact.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <LinearLayout
    android:orientation="horizontal"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:padding="7dp"
    >

     <TextView
                  android:id="@+id/industryContact"
                  android:layout_height="wrap_content"
                  android:layout_width="wrap_content"
                  android:layout_gravity="center_vertical"
                  android:textColor="#ffffff"
                  android:text="Contacts"
                  android:textSize="14sp"
                  android:layout_marginLeft="5dp"
                  android:textStyle="bold"
            />
     </LinearLayout>
     <View
        android:id="@+id/view01"
        android:layout_width="fill_parent"
        android:layout_height="1dp"
        android:background="#313437" />
    <ListView
       android:id="@android:id/list"
        android:layout_width="fill_parent"
      android:layout_height="fill_parent"
      android:layout_marginTop="10dp"
      android:layout_marginLeft="2dp"
```

```
      android:layout_marginRight="2dp"
      android:layout_marginBottom="2dp"
      android:divider="#e4e4e4"
      android:scrollingCache="false"
            android:dividerHeight="2px"
        />
</LinearLayout>
```

list_item.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <ListView
        android:id="@android:id/list"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content" />

    <TextView

        android:id="@+id/myGroups"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:padding="10dp"
        android:textSize="16sp" >
    </TextView>

</LinearLayout>
```

main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
    <ListView
            android:id="@+id/listview"
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
      />
    <TextView
            android:id="@+id/text"
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:layout_gravity="center"
```

```
      />

</LinearLayout>
```

provider_options.xml

```
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:padding="10dp"
    android:textColor="#ffffff"
    android:textSize="14sp" >
</TextView>
```

providers_list.xml

```
<RelativeLayout
      xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      >
    <ImageView
        android:id="@+id/provider"
        android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginTop="5dp"
        >
    </ImageView>
      <TextView
            android:id="@+id/providerText"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_toRightOf="@+id/provider"
            android:layout_marginTop="5dp"
            android:layout_marginLeft="10dp"
            >
      </TextView>
      <TextView
            android:id="@+id/signstatus"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentRight="true"
            android:layout_marginTop="5dp"
            android:layout_marginLeft="10dp"
            android:layout_marginRight="5dp"
            >
      </TextView>
</RelativeLayout>
```

ContactList.java

```
package org.brickred.customui;

import java.util.ArrayList;
import java.util.List;
import android.os.Bundle;
import android.app.ListActivity;
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import org.brickred.socialauth.Contact;

public class ContactList extends ListActivity {

	// Variables
	ListView listview;
	TextView title;
	String pressed;
	List <Contact> ContactList=new ArrayList<Contact>();

	@SuppressWarnings("unchecked")
	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.activity_contact_list);
		ContactList= (ArrayList<Contact>)
getIntent().getSerializableExtra("contacts");
		setListAdapter(new ContactListAdapter(this,ContactList));

	}
	@Override
	protected void onListItemClick(ListView l, View v, int position, long
id) {

	}

//adapter for seeing industry contacts in a list
	public class ContactListAdapter  extends ArrayAdapter<Contact>{
	  private final List<Contact> objects;
	  private final Context context;
```

```
        public ContactListAdapter(Context context,List<Contact> objects) {
             super(context,R.layout.contact_list, objects);
             this.context=context;
             this.objects=objects;

        }

        @Override
            public int getCount() {
                  return objects.size();
            }


        @Override
        public View getView(int position, View convertView, ViewGroup parent)
{
            LayoutInflater  inflater = (LayoutInflater)
context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);

              View rowView = inflater.inflate(R.layout.contact_list, parent,
false);
                                TextView nameView = (TextView)
rowView.findViewById(R.id.contactFirstName);


                                Contact u=objects.get(position);
                                nameView.setText(" "+u.getFirstName()+" "
+u.getLastName());


                    return rowView;
        }
}

}
```

CustomAdapter.java

```
package org.brickred.customui;

import org.brickred.socialauth.android.SocialAuthAdapter;
import org.brickred.socialauth.android.SocialAuthAdapter.Provider;

import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Color;
import android.util.Log;
```

```
import android.view.Gravity;
import android.view.LayoutInflater;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.ImageView;
import android.widget.TextView;

/**
 * CustomAdapter for showing List. On clicking any item , it calls
authorize() method to
 * authenticate provider
 */

public class CustomAdapter extends BaseAdapter
{
    // Android Components
      private LayoutInflater mInflater;
    private Context ctx;
    private Bitmap mIcon;

    // SocialAuth Components
    SocialAuthAdapter adapter;
      private Provider[] providers = new Provider[] {Provider.FACEBOOK,
Provider.TWITTER, Provider.LINKEDIN,Provider.MYSPACE};
    private int[] images = new int[]{R.drawable.facebook,
R.drawable.twitter,R.drawable.linkedin, R.drawable.myspace};

    public CustomAdapter(Context context , SocialAuthAdapter mAdapter)
    {
        // Cache the LayoutInflate to avoid asking for a new one each time.
        ctx = context;
      mInflater = LayoutInflater.from(ctx);
      adapter = mAdapter;
    }

    /**
     * The number of items in the list is determined by the number of
speeches
     * in our array.
     */
    public int getCount()
    {
        return providers.length;
    }

    /**
     * Since the data comes from an array, just returning the index is
     * sufficent to get at the data. If we were using a more complex data
     * structure, we would return whatever object represents one row in the
```

```
     * list.
     */
    public Object getItem(int position)
    {
        return position;
    }

    /**
     * Use the array index as a unique id.
     */
    public long getItemId(int position)
    {
        return position;
    }

    /**
     * Make a view to hold each row.
     *
     * @see android.widget.ListAdapter#getView(int, android.view.View,
     *      android.view.ViewGroup)
     */
    public View getView(final int position, View convertView, ViewGroup
parent)
    {
        // A ViewHolder keeps references to children views to avoid
unneccessary calls
        // to findViewById() on each row.
        ViewHolder holder;

        // When convertView is not null, we can reuse it directly, there is
no need
        // to reinflate it. We only inflate a new View when the convertView
supplied
        // by ListView is null.
        if (convertView == null)
        {
            convertView = mInflater.inflate(R.layout.providers_list, null);

            // Creates a ViewHolder and store references to the two children
views
            // we want to bind data to.
            holder = new ViewHolder();
            holder.text = (TextView)
convertView.findViewById(R.id.providerText);
            holder.icon = (ImageView)
convertView.findViewById(R.id.provider);

holder.signText=(TextView)convertView.findViewById(R.id.signstatus);

            convertView.setTag(holder);
        }
```

```
        else
        {
            // Get the ViewHolder back to get fast access to the TextView
            // and the ImageView.
            holder = (ViewHolder) convertView.getTag();
        }

        mIcon = BitmapFactory.decodeResource(ctx.getResources(),
images[position]);

        // Bind the data efficiently with the holder.
        holder.text.setText(providers[position].toString());
        holder.text.setPadding(20, 5, 0, 5);
        holder.text.setTextColor(Color.WHITE);
        holder.text.setGravity(Gravity.CENTER_VERTICAL);

        holder.icon.setImageBitmap(mIcon);

        holder.signText.setText("Sign In");
        holder.signText.setTextColor(Color.WHITE);
        holder.signText.setPadding(0, 5, 0, 5);
        holder.signText.setGravity(Gravity.CENTER_VERTICAL);
        holder.signText.setTag(1);

        // Click Events
        holder.text.setOnClickListener(new OnClickListener() {

               public void onClick(View v) {

                            CustomUI.pos = position;

                            // This method will enable the selected
provider
                            adapter.authorize(ctx, providers[position]);
               }
            });


           holder.signText.setOnClickListener(new OnClickListener() {

           public void onClick(View v) {

            final String text = (String) ((TextView)v).getText();

            if(text.equalsIgnoreCase("sign in"))
            {
                  CustomUI.pos = position;

                            // This method will enable the selected
provider
                            adapter.authorize(ctx, providers[position]);
```

```
            }
                        else if(text.equalsIgnoreCase("sign out"))
                        {
                              // Sign Out
                              boolean status = adapter.signOut();
                              Log.d("status", String.valueOf(status));
                              if(status)
                              {
                                ((TextView)v).setText("Sign In");
                              }
                        }
                }
            });

            return convertView;
        }

        class ViewHolder
        {
            TextView text;
            ImageView icon;
            TextView signText;
        }
} // End of customAdapter
```

CustomUI.java

```
package org.brickred.customui;

import java.io.Serializable;
import java.util.Arrays;
import java.util.HashMap;
import java.util.List;
import java.util.concurrent.ExecutionException;

import org.brickred.socialauth.Contact;
import org.brickred.socialauth.Profile;
import org.brickred.socialauth.android.DialogListener;
import org.brickred.socialauth.android.SocialAuthError;
import org.brickred.socialauth.android.SocialAuthAdapter;


import org.brickred.myclasses.*;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Color;
```

```
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.Gravity;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;


/**
 *
 * Main class of the CustomUI Example for SocialAuth Android SDK. <br>
 *
 * The main objective of this example is to access social media providers
 * Facebook, Twitter and others by creating your own UI.
 *
 * Here we are creating a ListView. The ListView contains list of all
providers
 * On clicking any provider, it authorizes the provider by calling authorize
method.
 *
 * After successful authentication of provider, it receives the response in
responseListener
 * and then shows a dialog containing options  for getting user profile , for
updating
 * status and to get contact list.<br>
 *
 * @author vineeta@brickred.com
 * edited by anu.xnk@gmail.com
 */


public class CustomUI extends Activity
{

      // SocialAuth Components
      SocialAuthAdapter adapter;
      Profile profileMap;

      // Android Components
      ListView listview;
      AlertDialog dialog;

      // Variables
      boolean status;
      public static int pos;
      private DialogListener dialogListener;
      private List<Contact2> contactsList2;
```

```
      private final static List <String> art=Arrays.asList("Animation","Arts
and Crafts","Design","Fine Art","Graphic Design","Motion Pictures and
Film","Museums and Institutions","Music","Performing
Arts","Photography","Writing and Editing");
      private final static List <String>
agriculture=Arrays.asList("Dairy","Farming","Fishery","Ranching");
      private final static List <String>
constructions=Arrays.asList("Architecture & Planning","Building
Materials","Civil Engineering","Commercial Real Estate","Construction","Glass
Ceramics & Concrete","Industrial Automation","Mechanical or Industrial
Engineering","Real Estate");
      private final static List <String>
corporation=Arrays.asList("Accounting","Business Supplies and
Equipment","Commercial Real Estate","Events Services","Facilities
Services","Human Resources","Import and Export","Logistics and Supply
Chain","Management Consulting","Market Research","Marketing and
Advertising","Outsourcing/Offshoring","Professional Training &
Coaching","Program Development","Public Relations and
Communications","Security and Investigations","Staffing and
Recruiting","Translation and Localization");
      private final static List <String> education=Arrays.asList("Education
Management","E-Learning","Higher Education","Primary/Secondary
Education","Research");
      private final static List <String>
finance=Arrays.asList("Accounting","Banking","Capital Markets","Commercial
Real Estate","Financial Services","Insurance","Investment
Banking","Investment Management","Real Estate","Venture Capital & Private
Equity");
      private final static List <String> goods=Arrays.asList("Apparel &
Fashion","Cosmetics","Consumer Electronics","Consumer
Goods","Electrical/Electronic Manufacturing","Food
Production","Furniture","Import and Export","Luxury Goods &
Jewelery","Packaging and Containers","Retail","Real Estate","Sporting
Goods","Supermarkets","Tobacco","Wholesale","Wine and Spirits");
      private final static List <String> government=Arrays.asList("Aviation &
Aerospace","Biotechnology","Civil Engineering","Defense & Space","Executive
Office","Government Administration","Government Relations","International
Trade and Development","International Affairs","Judiciary","Law
Enforcement","Legislative Office","Mechanical or Industrial
Engineering","Military","Nanotechnology","Public Policy","Public
Safety","Political Organization","Research","Renewables & Environment","Think
Tanks","Telecommunications","Translation and Localization");
      private final static List <String> health=Arrays.asList("Alternative
Medicine","Biotechnology","Health, Wellness and Fitness","Hospital & Health
Care","Medical Devices","Medical Practice","Mental Health
Care","Pharmaceuticals","Veterinary");
      private final static List <String>
legislation=Arrays.asList("Alternative Dispute Resolution","Judiciary","Law
Practice","Law Enforcement","Legal Services","Legislative Office");
      private final static List <String>
manufacturing=Arrays.asList("Airlines/Aviation","Automotive","Chemicals","Mac
```

```
hinery","Mining & Metals","Oil & Energy","Paper & Forest
Products","Plastics","Railroad
Manufacture","Shipbuilding","Textiles","Utilities","Renewables &
Environment","Nanotechnology","Aviation & Aerospace","Wine and Spirits","Food
Production","Consumer Electronics","Consumer Goods","Electrical/Electronic
Manufacturing","Furniture","Packaging and Containers","Retail","Business
Supplies and Equipment","Mechanical or Industrial Engineering","Glass
Ceramics & Concrete","Industrial Automation");
      private final static List <String>
media=Arrays.asList("Animation","Arts and Crafts","Broadcast Media","Computer
Games","Design","Entertainment","Fine Art","Graphic Design","Information
Services","Libraries","Marketing and Advertising","Media Production","Motion
Pictures and Film","Museums and Institutions","Newspapers","Online
Media","Performing Arts","Photography","Printing","Publishing","Writing and
Editing");
      private final static List <String> organisation=Arrays.asList
("Alternative Dispute Resolution","Civic & Social Organization","Consumer
Services","E-Learning","Environmental Services","Fund-Raising","Individual &
Family Services","International Trade and Development","Non-Profit
Organization Management","Philanthropy","Political Organization","Program
Development","Religious Institutions","Renewables & Environment","Security
and Investigations","Think Tanks");
      private final static List <String> recreation=Arrays.asList("Arts and
Crafts","Broadcast Media","Computer Games","Entertainment","Events
Services","Fine Art","Food & Beverages","Gambling & Casinos","Health,
Wellness and Fitness","Hospitality","Leisure, Travel &
Tourism","Libraries","Media Production","Motion Pictures and Film","Museums
and Institutions","Music","Newspapers","Performing
Arts","Photography","Printing","Publishing","Recreational Facilities and
Services","Restaurants","Sports","Sporting Goods","Wine and Spirits","Writing
and Editing");
      private final static List <String> services=Arrays.asList ("Civic &
Social Organization","Consumer Services","Environmental Services","Events
Services","Facilities Services","Food Production","Food &
Beverages","Hospitality","Information Services","Individual & Family
Services","Leisure, Travel & Tourism","Libraries","Package/Freight
Delivery","Recreational Facilities and Services","Restaurants","Religious
Institutions","Security and Investigations","Translation and Localization");
      private final static List <String>
technology=Arrays.asList("Airlines/Aviation","Biotechnology","Computer &
Network Security","Computer Hardware","Computer Networking","Computer
Software","Defense & Space","Information Technology and
Services","Internet","Nanotechnology","Pharmaceuticals","Semiconductors","Tel
ecommunications","Venture Capital & Private Equity","Wireless");
      private final static List <String>
transportation=Arrays.asList("Airlines/Aviation","Hospitality","Import and
Export","International Trade and Development","Leisure, Travel &
Tourism","Maritime","Logistics and Supply Chain","Package/Freight
Delivery","Transportation/Trucking/Railroad","Warehousing");
      private final static List <String> undefined=Arrays.asList
("undefined");
```

```
	private HashMap <String, List<String>> indList=new HashMap<String,
List<String>>();



	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.main);

		// Adapter initialization
		adapter = new SocialAuthAdapter(new ResponseListener());


		listview = (ListView)findViewById(R.id.listview);
		listview.setAdapter(new CustomAdapter(this, adapter));

		// Welcome Message
		TextView textview = (TextView)findViewById(R.id.text);
		textview.setText("Welcome to myLinkedinApp DEMO!   (based on
Socialauth)");
		textview.setTextColor(Color.WHITE);
		textview.setGravity(Gravity.CENTER);
		textview.setPadding(0, 30, 0,0);




		indList.put("art", art);
		indList.put("agriculture",agriculture);
		indList.put("constructions", constructions);
		indList.put("corporation",corporation);
		indList.put("education", education);
		indList.put("finance", finance);
		indList.put("goods",goods);
		indList.put("government", government);
		indList.put("health", health);
		indList.put("legislation", legislation);
		indList.put("manufacturing", manufacturing);
		indList.put("media", media);
		indList.put("organisation",organisation);
		indList.put("recreation",recreation);
		indList.put("services",services);
		indList.put("technology", technology);
		indList.put("transportation",transportation );
		indList.put("undefined", undefined);

	}
```

```
	// To receive the response after authentication
	private final class ResponseListener implements DialogListener
	{

		public void onComplete(Bundle values) {

			Log.d("Custom-UI" , "Successful");

			// Changing Sign In Text to Sign Out
			// Code to refresh Single ListView Item : You can remove it
for your app
			View v = listview.getChildAt(pos -
listview.getFirstVisiblePosition());
			TextView pText = (TextView)
v.findViewById(R.id.signstatus);
			pText.setText("Sign Out");

			// Get the provider
			final String providerName =
values.getString(SocialAuthAdapter.PROVIDER);
			Log.d("Custom-UI", "providername = " + providerName);


			int res = getResources().getIdentifier(providerName +
"_array","array", CustomUI.this.getPackageName());

			AlertDialog.Builder builder = new
AlertDialog.Builder(CustomUI.this);
			builder.setTitle("Select Options");
			builder.setCancelable(true);
			builder.setIcon(android.R.drawable.ic_menu_more);

			builder.setSingleChoiceItems(new
ArrayAdapter<String>(CustomUI.this, R.layout.provider_options,
getResources().getStringArray(res)), 0,
					new DialogInterface.OnClickListener() {
				public void onClick(DialogInterface dialog, int item)
{

					Events(item , providerName);
				}
			});
			dialog = builder.create();
			dialog.show();

		}

		public void onError(SocialAuthError error) {
			Log.d("Custom-UI" , "Error");
		}
```

```
            public void onCancel() {
                  Log.d("Custom-UI" , "Cancelled");
            }
      }
      public List<Contact2> getContactList2()
      {
            try
            {
                  contactsList2 = new contactTask2().execute().get();
            }
            catch (InterruptedException e)
            {
                  e.printStackTrace();
            }
            catch (ExecutionException e)
            {
                  e.printStackTrace();
            }

            return contactsList2;
      }
      private class contactTask2 extends AsyncTask<Void, Void,
List<Contact2>> {

            protected List<Contact2> doInBackground(Void... params) {
                  try
                  {
                        List<Contact2> contactsMap =
LinkedInImpl2.getContactList2(adapter.getCurrentProvider());
                        Log.d("SocialAuthAdapter", "Received Contact list");
                        return contactsMap;
                  }
                  catch (Exception e)
                  {
                        e.printStackTrace();
                        dialogListener.onError(new SocialAuthError("Contact
List not Received", e));
                        return null;
                  }
            }
      }

      // Method to handle events of providers
      public void Events(int position , String provider)
      {

            switch(position)
            {
            case 0 :            // Code to print user profile details for all
providers
```

```
            {

                  profileMap =  adapter.getUserProfile();

                  if (provider.equalsIgnoreCase("linkedin")){
                        Intent intent = new Intent(CustomUI.this,
ProfileActivity.class);
                        intent.putExtra("contacts", profileMap);
                        startActivity(intent);
                  }
                  break;
            }

            case 1 :                // Code to Post Message for all providers
            {

                  adapter.updateStatus("SocialAuth Android" +
System.currentTimeMillis());
                  Toast.makeText(CustomUI.this, "Message posted on " +
provider, Toast.LENGTH_SHORT).show();

                  break;
            }

            case 2 :                // Code to get Contacts List for all
providers
            {

                  List<Contact> contactsList = adapter.getContactList();
                  Intent contactintent=new
Intent(getApplicationContext(),ContactList.class);

                  contactintent.putExtra("contacts",
(Serializable)contactsList);
                  startActivity(contactintent);

                  break;
            }

            case 3 :            //  Back to Activity
            {

                  if (provider.equalsIgnoreCase("linkedin")){
                        try {

                              List<Contact2> contactsList =
getContactList2();
                              IndustryLog indLog= new IndustryLog();

                              if (contactsList != null && contactsList.size()
> 0)
```

```
                                    {
                                          for (Contact2 p : contactsList)
                                          {

      indLog.doSearchandAdd(p.getIndustry(), p);
                                          }
                                    }


                                    Intent myintent = new
Intent(getApplicationContext(),GroupsUI.class);

                                    myintent.putExtra("log", indLog);

                                    myintent.putExtra("indList",
(Serializable)indList);
                                    startActivity(myintent);
                                    break;

                              } catch (Exception e) {
                                    // TODO Auto-generated catch block
                                    e.printStackTrace();
                              }


                        }else{
                              dialog.dismiss();
                              break;}
                  }

                  case 4:
                  {
                        if (provider.equalsIgnoreCase("linkedin")){
                              dialog.dismiss();
                              break;
                        }
                  }
                  }
      }
}
```

FileCache.java

```
package org.brickred.customui;

import java.io.File;
import android.content.Context;

public class FileCache {
```

```
    private File cacheDir;

    public FileCache(Context context){
        //Find the dir to save cached images
        if
(android.os.Environment.getExternalStorageState().equals(android.os.Environme
nt.MEDIA_MOUNTED))
            cacheDir=new
File(android.os.Environment.getExternalStorageDirectory(),"TempImages");
        else
            cacheDir=context.getCacheDir();
        if(!cacheDir.exists())
            cacheDir.mkdirs();
    }

    public File getFile(String url){
        String filename=String.valueOf(url.hashCode());
        File f = new File(cacheDir, filename);
        return f;

    }

    public void clear(){
        File[] files=cacheDir.listFiles();
        if(files==null)
            return;
        for(File f:files)
            f.delete();
    }

}
```

GroupsUI.java

```
package org.brickred.customui;

import android.os.Bundle;
import android.app.ListActivity;
import android.util.Log;
import java.io.Serializable;
import java.util.HashMap;
import java.util.List;
import org.brickred.customui.R;
import org.brickred.myclasses.IndustryLog;
import android.content.Intent;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;

public class GroupsUI extends ListActivity {
```

```
	// Variables
	String provider_name;
	IndustryLog indlog;
	ListView listview;
	TextView title;
	String pressed;
	String def;
	String listItem;
	HashMap<String, List<String>> indList=new HashMap<String,
List<String>>();

	@SuppressWarnings("unchecked")
	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.groups_ui);

		//add the list from strings.xml otherwise add "" strings
		String[] groups =
getResources().getStringArray(R.array.groups_array);

		setListAdapter(new ArrayAdapter<String>(this,
android.R.layout.simple_list_item_1,groups));

		indlog=(IndustryLog) getIntent().getSerializableExtra("log");
		indList=(HashMap<String,
List<String>>)getIntent().getSerializableExtra("indList");


		for(int pass=0; pass<groups.length;pass++){
			String listItem=groups[pass];
			if (indList.containsKey(listItem)){
			List<String> thisList=indList.get(listItem) ;
			int counter=0;
			for(String temp: thisList){
				counter+=indlog.doSearchandGet(temp).size();
			}
			String says=listItem+"   "+"(" +Integer.toString(counter)+
")" ;
			groups[pass]=says;
			}
			}

	}
@Override
	protected void onListItemClick(ListView l, View v, int position, long
id) {
		//get selected items
		pressed= (String) getListAdapter().getItem(position);
		String pressed1[]= pressed.split("\\(");
```

```
            pressed= pressed1[0];
            pressed=pressed.trim();
            Log.i("list test", "I Clicked on element " + pressed + " and it
worked!");

            Intent myintent = new
Intent(getApplicationContext(),IndustriesUI.class);

            myintent.putExtra("log", indlog);
            myintent.putExtra("indList", (Serializable)indList);
            myintent.putExtra("pressed",pressed);
            startActivity(myintent);

      }
}
```

ImageLoader.java

```
package org.brickred.customui;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.Collections;
import java.util.Map;
import java.util.WeakHashMap;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

import android.app.Activity;
import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.widget.ImageView;

public class ImageLoader {

    MemoryCache memoryCache=new MemoryCache();
    FileCache fileCache;
    private Map<ImageView, String> imageViews=Collections.synchronizedMap(new
WeakHashMap<ImageView, String>());
    ExecutorService executorService;

    public ImageLoader(Context context){
        fileCache=new FileCache(context);
        executorService=Executors.newFixedThreadPool(5);
```

```
    }

    int stub_id = android.R.drawable.ic_dialog_alert;
    public void DisplayImage(String url, int loader, ImageView imageView)
    {
        stub_id = loader;
        imageViews.put(imageView, url);
        Bitmap bitmap=memoryCache.get(url);
        if(bitmap!=null)
            imageView.setImageBitmap(bitmap);
        else
        {
            queuePhoto(url, imageView);
            imageView.setImageResource(loader);
        }
    }

    private void queuePhoto(String url, ImageView imageView)
    {
        PhotoToLoad p=new PhotoToLoad(url, imageView);
        executorService.submit(new PhotosLoader(p));
    }

    private Bitmap getBitmap(String url)
    {
        File f=fileCache.getFile(url);

        //from SD cache
        Bitmap b = decodeFile(f);
        if(b!=null)
            return b;

        //from web
        try {
            Bitmap bitmap=null;
            URL imageUrl = new URL(url);
            HttpURLConnection conn =
(HttpURLConnection)imageUrl.openConnection();
            conn.setConnectTimeout(30000);
            conn.setReadTimeout(30000);
            conn.setInstanceFollowRedirects(true);
            InputStream is=conn.getInputStream();
            OutputStream os = new FileOutputStream(f);
            CopyStream(is, os);
            os.close();
            bitmap = decodeFile(f);
            return bitmap;
        } catch (Throwable ex){
           ex.printStackTrace();
           if (ex instanceof OutOfMemoryError)
                       memoryCache.clear();
```

```
            return null;
        }
    }

    //decodes image and scales it to reduce memory consumption
    private Bitmap decodeFile(File f){
        try {
            //decode image size
            BitmapFactory.Options o = new BitmapFactory.Options();
            o.inJustDecodeBounds = true;
            BitmapFactory.decodeStream(new FileInputStream(f),null,o);

            //Find the correct scale value. It should be the power of 2.
            final int REQUIRED_SIZE=70;
            int width_tmp=o.outWidth, height_tmp=o.outHeight;
            int scale=1;
            while(true){
                if(width_tmp/2<REQUIRED_SIZE || height_tmp/2<REQUIRED_SIZE)
                    break;
                width_tmp/=2;
                height_tmp/=2;
                scale*=2;
            }

            //decode with inSampleSize
            BitmapFactory.Options o2 = new BitmapFactory.Options();
            o2.inSampleSize=scale;
            return BitmapFactory.decodeStream(new FileInputStream(f), null,
o2);
        } catch (FileNotFoundException e) {}
        return null;
    }

    //Task for the queue
    private class PhotoToLoad
    {
        public String url;
        public ImageView imageView;
        public PhotoToLoad(String u, ImageView i){
            url=u;
            imageView=i;
        }
    }

    class PhotosLoader implements Runnable {
        PhotoToLoad photoToLoad;
        PhotosLoader(PhotoToLoad photoToLoad){
            this.photoToLoad=photoToLoad;
        }

        @Override
```

```
    public void run() {
        if(imageViewReused(photoToLoad))
            return;
        Bitmap bmp=getBitmap(photoToLoad.url);
        memoryCache.put(photoToLoad.url, bmp);
        if(imageViewReused(photoToLoad))
            return;
        BitmapDisplayer bd=new BitmapDisplayer(bmp, photoToLoad);
        Activity a=(Activity)photoToLoad.imageView.getContext();
        a.runOnUiThread(bd);
    }
}

boolean imageViewReused(PhotoToLoad photoToLoad){
    String tag=imageViews.get(photoToLoad.imageView);
    if(tag==null || !tag.equals(photoToLoad.url))
        return true;
    return false;
}

//Used to display bitmap in the UI thread
class BitmapDisplayer implements Runnable
{
    Bitmap bitmap;
    PhotoToLoad photoToLoad;
    public BitmapDisplayer(Bitmap b, PhotoToLoad p){
        bitmap=b;
        photoToLoad=p;
        }
    public void run()
    {
        if(imageViewReused(photoToLoad))
            return;
        if(bitmap!=null)
            photoToLoad.imageView.setImageBitmap(bitmap);
        else
            photoToLoad.imageView.setImageResource(stub_id);
    }
}

public void clearCache() {
    memoryCache.clear();
    fileCache.clear();
}
public static void CopyStream(InputStream is, OutputStream os)
{
    final int buffer_size=1024;
    try
    {
        byte[] bytes=new byte[buffer_size];
        for(;;)
```

```
            {
              int count=is.read(bytes, 0, buffer_size);
              if(count==-1)
                  break;
              os.write(bytes, 0, count);
            }
        }
        catch(Exception ex){}
    }
}
```

IndustriesUI.java

```
package org.brickred.customui;
import android.os.Bundle;
import android.app.ListActivity;
import android.util.Log;
import java.io.Serializable;
import org.brickred.customui.R;
import org.brickred.myclasses.Contact2;
import org.brickred.myclasses.IndustryLog;
import android.content.Intent;
import java.lang.String;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;

public class IndustriesUI extends ListActivity {


      // Variables
      String provider_name;
      IndustryLog indlog;
      ListView listview;
      TextView title;
      String pressed;

      List <Contact2> myList=new ArrayList<Contact2>();
      private HashMap <String, List<String>> indList=new HashMap<String,
List<String>>();



      @SuppressWarnings("unchecked")
      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
```

```
            setContentView(R.layout.industries_ui);
            indlog=(IndustryLog) getIntent().getSerializableExtra("log");
            indList=(HashMap<String,
List<String>>)getIntent().getSerializableExtra("indList");

            provider_name = getIntent().getStringExtra("provider");
            pressed= getIntent().getStringExtra("pressed");

            List <String> thisList=new ArrayList<String>();
            if (indList.containsKey(pressed)){
                  thisList=indList.get(pressed); }
            for(int counter=0; counter<thisList.size();counter++){
                  String listItem=(String) thisList.get(counter);
                  int listcount=indlog.doSearchandGet(listItem).size() ;
                  String says=listItem+"   "+"("
+Integer.toString(listcount)+ ")" ;
                  thisList.set(counter, says);
            }

            String[]industries =new String [thisList.size()];
            thisList.toArray(industries);

            setListAdapter(new ArrayAdapter<String>(this,
android.R.layout.simple_list_item_1,industries));


      }

      protected void onListItemClick(ListView l, View v, int position, long
id) {
            //get selected items
            pressed= (String) getListAdapter().getItem(position);
            String pressed1[]= pressed.split("\\(");
            pressed= pressed1[0];
            pressed=pressed.trim();
            Log.i("industries-ui", "I Clicked on element " + pressed+ " and
it worked!");
            if (indlog==null)Log.i("industries-ui", "i lista indlog einai
keni");
            myList=indlog.doSearchandGet(pressed);
            if (myList!=null){
                  Intent myintent = new
Intent(getApplicationContext(),IndustryContactList.class);
                  myintent.putExtra("contacts",(Serializable) myList );
                  startActivity(myintent);
            }else Log.i("industries-ui", "couldnt find pressed in indlog list
!");
      }
}
```

IndustryContact.java

```
package org.brickred.customui;


import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.TextView;
import org.brickred.myclasses.*;

public class IndustryContact extends Activity {
	// Variables
	String provider_name;
	ListView listview;
	TextView title;
	TextView name;
	TextView lastname;
	TextView profileURL;
	TextView industry;
	ImageView image;
	Contact2 contact;
	String image_url;
	ImageLoader imgLoader;
	int loader;
	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.activity_industry_contact);

		contact= (Contact2) getIntent().getSerializableExtra("contact");

		// Loader image - will be shown before loading image
		loader = R.drawable.no_contact;

		// Imageview to show
		image = (ImageView) findViewById(R.id.myimage);
		name = (TextView) findViewById(R.id.name);
		lastname = (TextView) findViewById(R.id.lastname);

		profileURL=(TextView)findViewById(R.id.profileURL);
		industry=(TextView)findViewById(R.id.industry);

		name.setText("Name:  "+contact.getFirstName());
		lastname.setText("Surname:  "+contact.getLastName());

		profileURL.setText("Profile link:  "+contact.getProfileUrl() );
		// Image url
		image_url = contact.getProfileImageURL();
		// ImageLoader class instance
		imgLoader = new ImageLoader(getApplicationContext());
```

```
            // whenever you want to load an image from url
            // call DisplayImage function
            // url - image url to load
            // loader - loader image, will be displayed before getting image
            // image - ImageView

            imgLoader.DisplayImage(image_url, loader, image);

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

IndustryContactList.java

```
package org.brickred.customui;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.List;

import org.brickred.myclasses.IndustryLog;

import android.os.Bundle;
import android.app.ListActivity;
import android.content.Context;
import android.content.Intent;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;

import org.brickred.myclasses.Contact2;

public class IndustryContactList extends ListActivity {

      // Variables
      String provider_name;
      IndustryLog indlog;
      ListView listview;
      TextView title;
      String pressed;
      List <Contact2> industryContactList=new ArrayList<Contact2>();
```

```
      @SuppressWarnings("unchecked")
      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.industry_contact);

            industryContactList= (ArrayList<Contact2>)
getIntent().getSerializableExtra("contacts");

            setListAdapter(new IndustryListAdapter(this,
industryContactList));

      }
      @Override
      protected void onListItemClick(ListView l, View v, int position, long
id) {
            Contact2 contact = (Contact2) getListAdapter().getItem(position);
            Intent myintent = new
Intent(getApplicationContext(),IndustryContact.class);
            myintent.putExtra("contact",(Serializable) contact );
            startActivity(myintent);
      }

//adapter for seeing industry contacts in a list
      public class IndustryListAdapter  extends ArrayAdapter<Contact2>{
        private final List<Contact2> objects;
        private final Context context;

        public IndustryListAdapter(Context context,List<Contact2> objects) {
             super(context,R.layout.industry_contact_list, objects);
             this.context=context;
             this.objects=objects;

        }

        @Override
            public int getCount() {
                  return objects.size();
            }


        @Override
        public View getView(int position, View convertView, ViewGroup parent)
{
            LayoutInflater  inflater = (LayoutInflater)
context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);

              View rowView = inflater.inflate(R.layout.industry_contact_list,
parent, false);
                              TextView nameView = (TextView)
rowView.findViewById(R.id.contactFisrtName);
```

```
                                Contact2 u=objects.get(position);
                                nameView.setText(" "+u.getFirstName()+" "
+u.getLastName());
                    return rowView;
        }
}
}
```

MemoryCache.java

```
package org.brickred.customui;


import java.lang.ref.SoftReference;
import java.util.Collections;
import java.util.HashMap;
import java.util.Map;
import android.graphics.Bitmap;

public class MemoryCache {
   private Map<String, SoftReference<Bitmap>>
cache=Collections.synchronizedMap(new HashMap<String,
SoftReference<Bitmap>>());

   public Bitmap get(String id){
       if(!cache.containsKey(id))
           return null;
       SoftReference<Bitmap> ref=cache.get(id);
       return ref.get();
   }

   public void put(String id, Bitmap bitmap){
       cache.put(id, new SoftReference<Bitmap>(bitmap));
   }

   public void clear() {
       cache.clear();
   }
}
```

ProfileActivity.java

```
package org.brickred.customui;

import android.os.Bundle;
import android.app.Activity;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.TextView;
import org.brickred.socialauth.Profile;
```

```
public class ProfileActivity extends Activity {

	// Variables
	String provider_name;
	ListView listview;
	TextView title;
	TextView name;
	TextView lastname;
	TextView email;
	TextView location;
	TextView gender;
	TextView language;
	TextView dob;
	ImageView image;
	Profile ProfileMap;
	String image_url;
	ImageLoader imgLoader;
	int loader;

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

		ProfileMap= (Profile)
getIntent().getSerializableExtra("contacts");

		//title.setText(ProfileMap.getFirstName() +"
"+ProfileMap.getLastName());

		// Loader image - will be shown before loading image
	  loader = R.drawable.no_contact;

	  // Imageview to show
	  image = (ImageView) findViewById(R.id.myimage);

	  name = (TextView) findViewById(R.id.name);
		lastname = (TextView) findViewById(R.id.lastname);
		dob=(TextView) findViewById(R.id.dob);
		email = (TextView) findViewById(R.id.email);
		location=(TextView)findViewById(R.id.location);

		name.setText("Name:  "+ProfileMap.getFirstName());
		lastname.setText("Surname:  "+ProfileMap.getLastName());
		email.setText("Email:  "+ProfileMap.getEmail());
		dob.setText("Date of Birth:  "+ ProfileMap.getDob().toString());
		location.setText("Location:  "+ProfileMap.getLocation());
		  // Image url
	  image_url = ProfileMap.getProfileImageURL();
	  // ImageLoader class instance
```

```
        imgLoader = new ImageLoader(getApplicationContext());

        // whenever you want to load an image from url
        // call DisplayImage function
        // url - image url to load
        // loader - loader image, will be displayed before getting image
        // image - ImageView

        imgLoader.DisplayImage(image_url, loader, image);

    }
}
```

Contact2.java

```
package org.brickred.myclasses;

import java.io.Serializable;
import  org.brickred.socialauth.Contact;
import  org.brickred.socialauth.util.*;

public class Contact2 extends Contact implements Serializable{

    /**
     *
     */
    private static final long serialVersionUID = 1L;
    private String industry;
    private String profileImageURL;
    private BirthDate dob;


    public String getIndustry() {
        return industry;
    }


    public void setIndustry(final String industry) {
        this.industry = industry;
    }
    public BirthDate getDob() {
        return dob;
    }

    /**
     * Updates the date of birth
     *
     * @param dob
     *            the date of birth of user
     */
    public void setDob(final BirthDate dob) {
        this.dob = dob;
```

```
	}

	public String getProfileImageURL() {
		return profileImageURL;
	}


	public void setProfileImageURL(final String profileImageURL) {
		this.profileImageURL = profileImageURL;
	}
	@Override
	public String toString() {
		StringBuilder result = new StringBuilder();
		String NEW_LINE = System.getProperty("line.separator");
		result.append(this.getClass().getName() + " Object {" +
NEW_LINE);
		result.append(" firstName: " + super.getFirstName() + NEW_LINE);
		result.append(" lastName: " + super.getLastName() + NEW_LINE);
		result.append(" dob: ") ;
		result.append(dob.getDay());
		result.append(dob.getMonth());
		result.append(dob.getYear());
		result.append(""+ NEW_LINE);
		result.append("profileUrl: " + super.getProfileUrl() + NEW_LINE);
		result.append(" profileImageURL: " + profileImageURL + NEW_LINE);
		result.append(" industry: " + industry + NEW_LINE);

		result.append("}");
		return result.toString();
	}

}
```

IndustryLog.java

```
package org.brickred.myclasses;

import org.brickred.myclasses.Contact2;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.List;


public class IndustryLog implements Serializable {


	/**
	 *
```

```
	 */
	private static final long serialVersionUID = 1L;

	private HashMap <String, List<Contact2>> industry;

	private List<Contact2> myList=new ArrayList<Contact2>();

	private  List <String> myIndustries
=Arrays.asList("Accounting","Airlines/Aviation","Alternative Dispute
Resolution","Alternative Medicine","Animation",
			"Apparel & Fashion","Architecture & Planning","Arts and
Crafts","Automotive","Aviation & Aerospace",
			"Banking","Biotechnology", "Broadcast Media","Building
Materials","Business Supplies and Equipment",
			"Capital Markets","Chemicals","Civic & Social
Organization","Civil Engineering","Commercial Real Estate",
			"Computer & Network Security","Computer Games","Computer
Hardware","Computer Networking","Computer Software",
			"Construction","Consumer Electronics","Consumer
Goods","Consumer Services","Cosmetics","Dairy","Defense &
Space","Design","Education Management","E-Learning","Electrical/Electronic
Manufacturing","Entertainment",
			"Environmental Services","Events Services","Executive
Office","Facilities Services","Farming","Financial Services",
			"Fine Art","Fishery","Food & Beverages","Food
Production","Fund-Raising","Furniture","Gambling & Casinos",
			"Glass Ceramics & Concrete","Government
Administration","Government Relations","Graphic Design","Health, Wellness and
Fitness",
			"Higher Education","Hospital & Health
Care","Hospitality","Human Resources","Import and Export","Individual &
Family Services",
			"Industrial Automation","Information Services","Information
Technology and Services","Insurance","International Affairs","International
Trade and Development",
			"Internet","Investment Banking","Investment
Management","Judiciary","Law Enforcement","Law Practice","Legal
Services","Legislative Office",
			"Leisure, Travel & Tourism","Libraries","Logistics and
Supply Chain","Luxury Goods & Jewelery","Machinery",
			"Management Consulting","Maritime","Market
Research","Marketing and Advertising","Mechanical or Industrial Engineering",
			"Media Production","Medical Devices","Medical
Practice","Mental Health Care","Military","Mining & Metals","Motion Pictures
and Film",
			"Museums and
Institutions","Music","Nanotechnology","Newspapers","Non-Profit Organization
Management","Oil & Energy",
			"Online Media","Outsourcing/Offshoring","Package/Freight
Delivery","Packaging and Containers","Paper & Forest Products",
```

```
                  "Performing
Arts","Pharmaceuticals","Philanthropy","Photography","Plastics","Political
Organization",    "Primary/Secondary Education",
                  "Printing","Professional Training & Coaching","Program
Development","Public Policy","Public Relations and Communications",
                  "Public Safety","Publishing","Railroad
Manufacture","Ranching","Real Estate","Recreational Facilities and
Services","Religious Institutions",
                  "Renewables &
Environment","Research","Restaurants","Retail","Security and
Investigations","Semiconductors","Shipbuilding","Sporting Goods",
                  "Sports","Staffing and
Recruiting","Supermarkets","Telecommunications","Textiles","Think
Tanks","Tobacco","Translation and
Localization","Transportation/Trucking/Railroad",
                  "Utilities","Venture Capital & Private
Equity","Veterinary","Warehousing","Wholesale","Wine and
Spirits","Wireless","Writing and Editing","undefined");


      public void doSearchandAdd(String myIndustry,Contact2 myIndList){

            if (!industry.containsKey(myIndustry)){
                  myList =industry.get("undefined");
                  myList.add(myIndList);
            } else {
                  myList=industry.get(myIndustry);
                  myList.add(myIndList);
            }
      }

      public List<Contact2> doSearchandGet(String myIndustry){
            if (!industry.containsKey(myIndustry)){
                  return (new ArrayList<Contact2>());
            } else {
                  myList=industry.get(myIndustry);
                  return myList;
            }
      }
      public IndustryLog(){
            industry= new HashMap<String, List<Contact2>>();
            for (String temp : myIndustries) {
                  industry.put(temp, new ArrayList<Contact2>());
            }
      }
}
```

LinkedInImpl2.java

```
package org.brickred.myclasses;
```

```
import java.util.ArrayList;

import java.util.List;


import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import org.brickred.socialauth.AuthProvider;
import org.brickred.socialauth.exception.ServerDataException;
import org.brickred.socialauth.exception.SocialAuthException;
import org.brickred.socialauth.util.BirthDate;
import org.brickred.socialauth.util.MethodType;

import org.brickred.socialauth.util.Response;
import org.brickred.socialauth.util.XMLParseUtil;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;



public class LinkedInImpl2 {


//    private static final String SKILLS_URL=
"http://api.linkedin.com/v1/people/~/connections:(email-address,date-of-
birth)";

      private static final String SKILLS_URL=
"http://api.linkedin.com/v1/people/~/connections:(first-name,last-name,email-
address,date-of-birth,public-profile-url,picture-url,industry)";

      private static final Log LOG = LogFactory.getLog(LinkedInImpl2.class);

      public static List<Contact2> getContactList2(AuthProvider prov) throws
Exception {
            LOG.info("Fetching contacts from " + SKILLS_URL);
            Response serviceResponse = null;

            try {

                  serviceResponse =
prov.api(SKILLS_URL,MethodType.GET.toString(), null, null, null);
                  //    LOG.info("to xml
einai"+serviceResponse.getResponseBodyAsString("UTF-8"));
            } catch (Exception ie) {
                  ie.printStackTrace();
                  throw new SocialAuthException(
                              "Failed to retrieve the contacts from " +
SKILLS_URL,
```

```
                              ie);

                  }
                  Element root;
                  try {
                        root = XMLParseUtil.loadXmlResource(serviceResponse
                                    .getInputStream());
                  } catch (Exception e) {
                        throw new ServerDataException(
                                    "Failed to parse the contacts from response."
                                                + SKILLS_URL, e);
                  }
                  List<Contact2> contactSkillsList = new ArrayList<Contact2>();


                  if (root != null) {
                        NodeList pList = root.getElementsByTagName("person");
                        if (pList != null && pList.getLength() > 0) {
                              LOG.debug("Found contacts : " + pList.getLength());
                              for (int i = 0; i < pList.getLength(); i++) {
                                    Element p = (Element) pList.item(i);
                                    Contact2 cont = new Contact2();

                                    String fname = XMLParseUtil.getElementData(p,
"first-name");
                                    if (fname != null) {
                                          cont.setFirstName(fname);
                                    }
                                    String lname = XMLParseUtil.getElementData(p,
"last-name");
                                    if (lname != null) {
                                          cont.setLastName(lname);
                                    }

                                    NodeList dob = p.getElementsByTagName("date-of-
birth");
                                    if (dob != null && dob.getLength() > 0) {
                                          Element dobel = (Element) dob.item(0);
                                          if (dobel != null) {
                                                String y =
XMLParseUtil.getElementData(dobel, "year");
                                                String m =
XMLParseUtil.getElementData(dobel, "month");
                                                String d =
XMLParseUtil.getElementData(dobel, "day");
                                                BirthDate bd = new BirthDate();
                                                if (m != null) {

      bd.setMonth(Integer.parseInt(m));
                                                }
                                                if (d != null) {
```

```
      bd.setDay(Integer.parseInt(d));
                                          }
                                          if (y != null) {

      bd.setYear(Integer.parseInt(y));
                                          }
                                          cont.setDob(bd);
                                    }
                              }

                              String profileUrl =
XMLParseUtil.getElementData(p,"public-profile-url");
                              if (profileUrl != null) {
                                    cont.setProfileUrl(profileUrl);
                              }
                              String picUrl = XMLParseUtil.getElementData(p,
                                          "picture-url");
                              if (picUrl != null) {
                                    cont.setProfileImageURL(picUrl);
                              }
                              String email = XMLParseUtil.getElementData(p,
"email-address");
                              if (email != null) {
                                    cont.setEmail(email);
                              }

                              String industry =
XMLParseUtil.getElementData(p, "industry");
                              if (industry != null) {
                                    cont.setIndustry(industry);
                              }


                              contactSkillsList.add(cont);
                        }
                  } else {
                        LOG.debug("No connections were obtained from : "
                                    + SKILLS_URL);
                  }
            }return contactSkillsList;
      }
}
```

6. *Βιβλιογραφία*

[1] https://apigee.com/providers

[2] https://apigee.com/console/linkedin?authTypeVal=oauth1&afterSuccessfulAuth=true&req=%7B%22resource%22%3A%22getmyconnections%22%2C%22params%22%3A%7B%22query%22%3A%7B%7D%2C%22template%22%3A%7B%7D%2C%22headers%22%3A%7B%7D%2C%22body%22%3A%7B%22attachmentFormat%22%3A%22mime%22%2C%22attachmentContentDisposition%22%3A%22form-data%22%7D%7D%2C%22verb%22%3A%22get%22%7D

[3] http://developer.linkedin.com/documents/industry-codes

[4] http://geekswithblogs.net/bosuch/archive/2010/11/17/switching-between-activities-screens.-in-android.aspx

[5] https://developer.linkedin.com/documents/authentication#granting

[6] http://j2eeguru.blogspot.gr/2012/04/listview-listactivity-and-custom.html

[7] http://developer.android.com/guide/topics/ui/actionbar.html#Tabs

[8] http://developer.android.com/reference/android/app/ListActivity.html

[9] http://www.androidhive.info/2012/07/android-loading-image-from-url-http/

[10] http://www.vogella.com/articles/AndroidListView/article.html#listview_overview

[11] http://developer.android.com/training/basics/firstapp/building-ui.html

[12] http://developer.android.com/guide/components/fragments.html

[13] http://www.mkyong.com/android/android-tablelayout-example/

[14] http://mobile.tutsplus.com/tutorials/android/android-sdk_contact-badge/

[15] http://el.wikipedia.org/wiki/Android

[16] http://en.wikipedia.org/wiki/Mobile_phone

[17] http://en.wikipedia.org/wiki/Smartphone

[18] http://www.zdnet.com/blog/burnette/how-android-works-the-big-picture/515

[19] https://code.google.com/p/socialauth/
[20] https://code.google.com/p/socialauth-android/
[21] http://developer.android.com/tools/sdk/eclipse-adt.html
[22] http://en.wikipedia.org/wiki/OAuth
[23] http://vigrond.com/blog/2012/02/04/oauth-extension-php-and-the-tumblr-api/
[24] http://hueniverse.com/oauth/guide/terminology/
[25] http://musmato.blogspot.gr/2012/09/diy-social-auth.html
[26] http://opensource.org/licenses/MIT
[27] http://stackoverflow.com/questions/4113934/how-is-oauth-2-different-from-oauth-1
[28] http://tools.ietf.org/html/rfc5849#section-1.2
[29] http://tools.ietf.org/html/rfc6749
[30] http://en.wikipedia.org/wiki/LinkedIn
[31] http://en.wikipedia.org/wiki/Android_(operating_system)