Ατομική Διπλωματική Εργασία

CLODA

Υλοποίηση Αρχιτεκτονικής Πληθοπορισμού για Διαχείριση Αντικειμένων σε Εσωτερικούς Χώρους

Τζιούλια Μετόχη

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2013

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

CLODA

Υλοποίηση Αρχιτεκτονικής Πληθοπορισμού για Διαχείριση Αντικειμένων σε Εσωτερικούς Χώρους

Τζιούλια Μετόχη

Επιβλέπων Καθηγητής
Δημήτρης Ζεϊναλιπούρ

Η Ατομική Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου

Μάιος 2013

Ευχαριστίες

Αρχικά, θα ήθελα να εκφράσω τις ευχαριστίες μου στον επιβλέπων καθηγητή μου κ. Δημήτρη Ζεϊναλιπούρ για όλη τη βοήθεια και καθοδήγηση που μου παρείχε κατά τη διάρκεια εκπόνησης της διπλωματικής μου εργασίας. Θα ήθελα ακόμη να τον ευχαριστήσω για το θέμα που μου ανέθεσε καθώς είναι ένα καινοτόμο θέμα και μέσα από την όλη διαδικασία υλοποίησης της συγκεκριμένης εφαρμογής κατάφερα να πάρω αρκετές γνώσεις σε διάφορα θέματα οι οποίες και θα με βοηθήσουν στη μετέπειτα πορεία μου.

Μέσω της συνεργασίας αυτής ήρθα σε επαφή με αντικείμενα που αρχικά ήταν άγνωστα για μένα αλλά με αρκετή θέληση, επιμονή και υποστήριξη κατάφερα να τα φέρω εις πέρας και να ολοκληρώσω τη διπλωματική μου εργασία.

Επιπλέον, θα ήθελα να ευχαριστήσω τον Γιώργο Λάρκου και τον Κωνσταντίνο Κώστα για τη βοήθεια και στήριξη που μου παρείχαν καθόλη τη διάρκεια της διπλωματικής μου εργασίας. Η στήριξή τους τόσο σε πρακτικό όσο και σε εμψυχωτικό επίπεδο ήταν πολύ σημαντική καθώς σε στιγμές πίεσης και δυσκολιών που αντιμετώπισα ήταν πάντα δίπλα μου και με ενθάρρυναν να συνεχίσω.

Ακόμη, θα ήθελα να ευχαριστήσω όλους τους φίλους μου οι οποίοι ήταν πάντοτε στο πλευρό μου και στήριζαν την προσπάθειά μου αυτή.

Τέλος, θα ήθελα να ευχαριστήσω την οικογένειά μου και ιδιαίτερα τους γονείς μου, οι οποίοι στάθηκαν δίπλα μου όλα αυτά τα χρόνια των σπουδών μου, και ιδιαίτερα στο τελευταίο έτος και στην εκπόνηση της διπλωματικής μου εργασίας. Θέλω ακόμη να τους ευχαριστήσω για την κατανόηση που έδειξαν στις πολύωρες απουσίες μου, και για την κάθε είδους υποστήριξη και φροντίδα που μου έδωσαν.

Περίληψη

Η διπλωματική μου εργασία αφορά στην υλοποίηση μίας πληθωποριστικής αρχιτεκτονικής διασυνδεδεμένων δημοσιοποιημένων πληροφοριών (CLODA- CrowdSourced Linked Open Data Architecture) και αποτελεί μία πρώτη προσπάθεια συνδυασμού του πληθοπορισμού, εντοπισμού και υπηρεσιών που είναι βασισμένες στην τοποθεσία με σκοπό, την παραγωγή, συλλογή, επικύρωση αλλά και τη συσχέτιση πραγματικών, γεωχωρικών και πολυδιάστατων πληροφοριών χρησιμοποιώντας έξυπνες κινητές συσκευές καθώς επίσης και οποιεσδήποτε άλλες κινητές συσκευές .

Το πρόβλημα που εντοπίστηκε ήταν η δυσκολία εύρεσης αντικειμένων που φέρουν γραμμικό κώδικα (barcode), εφόσον ακόμα και αν υπήρχαν αντικείμενα γεωτοποθετημένα στο χώρο, θα ήταν δύσκολη η ενημέρωση των διασυνδεδεμένων δημοσιοποιημένων πληροφοριών. Για αυτό το λόγο έχει υλοποιηθεί η εφαρμογή CLODA που αναφέρθηκε πιο πάνω.

Η αρχιτεκτονική αυτή έχει υλοποιηθεί ως εφαρμογή σε Android λειτουργικό σύστημα για έξυπνες κινητές συσκευές και εστιάζει στην κατασκευή ενός διευθυνσιοδοτούμενου ενιαίου αναγνωριστικού πόρου (URI), αλληλένδετων καθώς επίσης και ημι-δομημένων δεδομένων που ακολουθούν το πρότυπο των διασυνδεδεμένων δημοσιοποιημένων πληροφοριών (LOD). Ακολούθως, στην εγκυρότητα των κατασκευασμένων δεδομένων συμβάλλει η συμμετοχή πλήθους.

Η υλοποίησή μου έχει γίνει πάνω από τα Google Maps και χρησιμοποιεί τη βιβλιοθήκη Airplace, που είναι ένα πλαίσιο εσωτερικής τοποθέτησης το οποίο αναπτύχθηκε στο Πανεπιστήμιο Κύπρου.

Η αρχιτεκτονική αυτή έχει ως αποτέλεσμα να χρησιμοποιούνται τα πλεονεκτήματα καθενός από τα κομμάτια από τα οποία και αποτελείται π.χ. Εσωτερική Τοποθέτηση, Πληθωπορισμός σε κινητές συσκευές. Τα κύρια πλεονεκτήματα της εφαρμογής CLODA είναι πρώτα από όλα η ικανότητα να λειτουργεί σε εσωτερικούς χώρους, πάνω από υπάρχουσες γεωγραφικά χαρτογραφημένες τεχνολογίες (π.χ. Google Maps). Ένα ακόμη πλεονέκτημα της εφαρμογής αυτής είναι η κατανάλωση δημόσια διαθέσιμων API (π.χ. Google Shopping API). Τέλος, η χρήση της δύναμης του πλήθους αποτελεί μεγάλο πλεονέκτημα έτσι ώστε να κατασκευαστούν επικυρωμένες διασυνδεδεμένες δημοσιοποιημένες πληροφορίες (LOD) με γνώση γεωτοποθέτησης.

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή 1

Τα διασυνδεδεμένα δεδομένα περιγράφουν μία μέθοδο δημοσίευσης δομημένων δεδομένων έτσι ώστε να μπορεί να διασυνδεθεί και να γίνει πιο χρήσιμη [1], [2]. Παρόλο που ο παγκόσμιος ιστός παραδοσιακά έχει υπάρξει διασυνδεδεμένος, η απουσία της σημασιολογίας στα δημοσιοποιημένα δεδομένα, έκανε δύσκολη την άντληση σημαντικών συμπερασμάτων από αυτά τα δεδομένα. Ως αποτέλεσμα, πολλές προσπάθειες την σήμερον ημέρα, περιλαμβανομένου του CKAN, DBpedia, GeoNames κ.λ.π, δημιουργούν σημασιολογικά πλούσιες πηγές δεδομένων από τον παγκόσμιο ιστό. Τέτοια διαδικασία παρεμποδίζεται σημαντικά από την έλλειψη εργαλείων για τη δημιουργία μεγάλων όγκων μεταδεδομένων για παροχή έγκυρης σημασιολογίας σε μία λογική κλίμακα, π.χ. κάθε αντικείμενο στον κόσμο που μπορεί να σαρωθεί μέσω γραμμικού κώδικα (barcode).

Από την άλλη, ο πληθοπορισμός, αναδύεται ως ένα από τα πιο κατάλληλα μέσα για την απρόσκοπτη υλοποίηση και υιοθέτηση του προτύπου των διασυνδεδεμένων δημοσιοποιημένων δεδομένων (LOD). Επιπλέον, αποτελεί ένα μέσο για επικύρωση και ολοκλήρωση των διασυνδεδεμένων δημοσιοποιημένων πληροφοριών και την ίδια στιγμή δημιουργεί σχέσεις μεταξύ ετερογενών διασυνδεδεμένων δημοσιοποιημένων πληροφοριών (LOD) , χρησιμοποιώντας τη δύναμη του πλήθους.

Ο όρος πληθοπορισμός αναφέρεται σε ένα νέο κατανεμημένο, κοινωνικό, συνεργατικό μοντέλο επίλυσης προβλημάτων στο οποίο ένα πλήθος απροσδιορίστου μεγέθους δεσμεύεται να λύσει ένα περίπλοκο πρόβλημα μέσω μίας ανοικτής κλήσης [3].

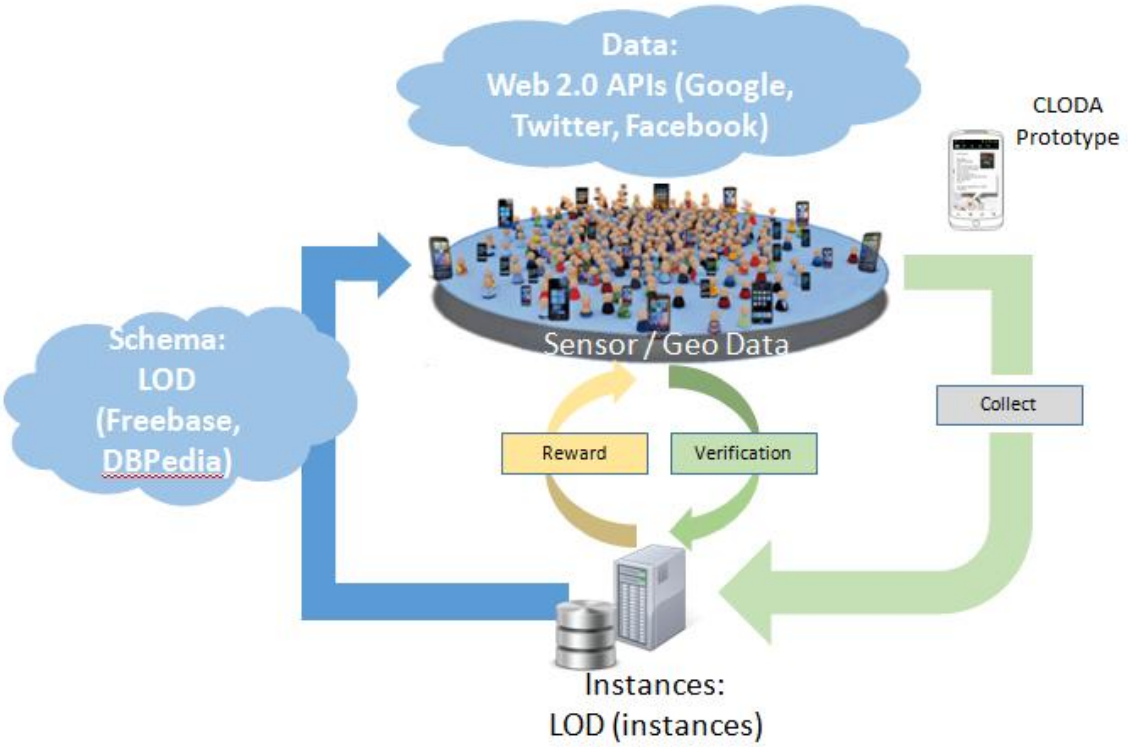
Οι κινητές έξυπνες συσκευές και οι κινητές γενικά συσκευές μπορούν να ξεδιπλώσουν όλες τις δυνατότητες του πληθοπορισμού, και να επιταχύνουν σημαντικά το ρόλο του πραγματοποιώντας το όραμα των διασυνδεδεμένων δημοσιοποιημένων πληροφοριών (LOD). Οι κινητές συσκευές σήμερα, διαθέτουν πρωτοφανείς πολύ-αισθητήριες δυνατότητες (π.χ., γεωτοποθέτηση, φως, κίνηση, αισθητήρες ακοής και όρασης) οι οποίες παρέχουν μία νέα

ποικιλία αποτελεσματικών μέσων για ευκαιριακή συλλογή δεδομένων. Ακόμη πιο σημαντικό αποτελεί και το γεγονός ότι είναι πάντα συνδεδεμένες και χρησιμοποιούνται καθημερινά για όλων των ειδών τις εργασίες.

Ακόμη μία σημαντική πτυχή που σχετίζεται με την εκπλήρωση του οράματος των διασυνδεδεμένων δημοσιοποιημένων πληροφοριών (LOD) σε έξυπνες κινητές συσκευές, είναι και το που έλαβε χώρα η συλλογή των δεδομένων. Σύμφωνα με πρόσφατες στατιστικές, οι άνθρωποι ξοδεύουν 80-90% του χρόνου τους σε εσωτερικούς χώρους όπως για παράδειγμα βιβλιοθήκες, εμπορικά κέντρα, αεροδρόμια και πανεπιστήμια, ενώ 70% των κινητών κλήσεων και 80% των συνδέσεων δεδομένων προέρχονται από εσωτερικούς χώρους [4]. Αυτό, έχει προκαλέσει ένα αυξανόμενο ενδιαφέρον στις υπηρεσίες που είναι βασισμένες στην εσωτερική τοποθέτηση (LBS) αλλά και στις εφαρμογές που αφορούν στη γνώση της γεωγραφικής θέσης, όπως για παράδειγμα πλοήγηση και καθοδήγηση σε εσωτερικούς χώρους. Προκειμένου να δημιουργηθούν τέτοιες εφαρμογές και να γίνουν ευρέως αποδεκτές, απαιτούνται εναλλακτικές λύσεις για την παροχή ακριβών και αξιόπιστων εκτιμήσεων τοποθεσιών. Η τοποθέτηση που βασίζεται σε δορυφόρους, π.χ., GPS, δεν είναι διαθέσιμη ή έχει υποβαθμιστεί σημαντικά στο εσωτερικό κτιρίων λόγω, της απόφραξης ή εξασθένησης της ισχύος του σήματος.

Η διπλωματική μου εργασία, παρουσιάζει μία αρχιτεκτονική η οποία εκμεταλλεύεται τη δύναμη του πλήθους προκειμένου να δημιουργήσει, συλλέξει και επικυρώσει πραγματικά, πολυδιάστατα δεδομένα χρησιμοποιώντας έξυπνες κινητές συσκευές και κινητές συσκευές σε εσωτερικούς αλλά και εξωτερικούς χώρους. Η αρχιτεκτονική αυτή, επιτρέπει τη δημιουργία νέων ανοικτών συνόλων δεδομένων και υποστηρίζει τους ιδιοκτήτες δεδομένων να δημοσιοποιήσουν αυτά τα δεδομένα. Αυτά τα σύνολα δεδομένων που δημοσιεύονται υπό τις αρχές των διασυνδεδεμένων δημοσιοποιημένων πληροφοριών (LOD), αναμένεται να οδηγήσουν στη δημιουργία νέων εφαρμογών και υπηρεσιών (π.χ., εσωτερική αποθήκευση και διαχείριση δεδομένων για καταστήματα, αποθήκες, δημόσια κτίρια, εμπορικά κέντρα, αεροδρόμια κ.λ.π.)

Κίνητρο για τη δημιουργία αυτής της αρχιτεκτονικής υπήρξε το γεγονός ότι παρόλο που γίνονται αρκετές προσπάθειες για δημιουργία εφαρμογών διαχείρισης αποθεμάτων (π.χ. Inventory Tracker, Barcode & Inventory Pro and Inventory Android applications), όλες αυτές, επικεντρώνονται στον ένα και μοναδικό χρήστη και όχι στα οφέλη που θα μπορούσε να αποφέρει στο πλήθος. Αποτέλεσμα αυτού είναι, όλα τα δεδομένα να αποθηκεύονται ιδιωτικά και να μην υπάρχει τρόπος να είναι προσβάσιμα και σε άλλους. Στόχος της διπλωματικής μου εργασίας είναι, να παρέχω μία αρχιτεκτονική και μία εφαρμογή για την απόκτηση επικυρωμένων διασυνδεδεμένων δημοσιοποιημένων πληροφοριών (LOD) μέσω του πληθοπορισμού.



Κεφάλαιο 1 - Εικόνα 1.1
Αρχιτεκτονική CLODA

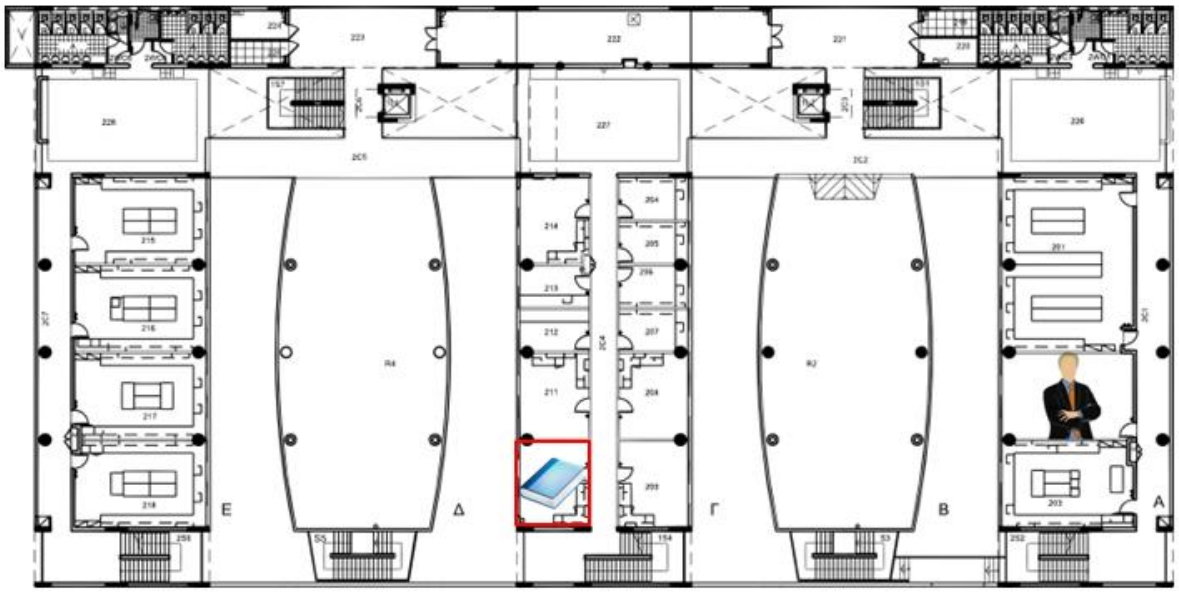
Η εφαρμογή CLODA αποτελείται από διάφορα μέρη που έχουν υλοποιηθεί κατά καιρούς όπως την εσωτερική τοποθέτηση [5] και τον απομακρυσμένο έλεγχο εφαρμογών έξυπνων κινητών συσκευών [6].

Όπως φαίνεται από την εικόνα πιο πάνω η εφαρμογή CLODA αποτελείται από διασυνδεδεμένες δημοσιοποιημένες πληροφορίες οι οποίες λαμβάνονται από το Freebase, DBpedia και άλλες βάσεις δεδομένων και σε συνδυασμό με τις υπόλοιπες πληροφορίες των αντικειμένων που λαμβάνονται από το Google Shoppinh API , αποθηκεύονται στη CouchDB βάση δεδομένων. Επιπλέον, χρησιμοποιούνται τα Google Maps για να εμφανίζεται το κτίριο της Πληροφορικής.

Στη συνέχεια ακολουθούν δύο σενάρια στα οποία διαφαίνεται η χρησιμότητα της αρχιτεκτονικής CLODA.

Σενάριο 1:

Συχνά, τόσο οι καθηγητές, οι ερευνητές καθώς επίσης και οι απλοί φοιτητές δανείζονται βιβλία για διδακτικούς και ερευνητικούς σκοπούς. Τις περισσότερες όμως φορές, το βιβλίο που χρειάζεται κάποιος είναι ήδη κρατημένο από κάποιο άλλο ή από το ίδιο το πανεπιστήμιο ή κάποιο εργαστήριο. Βέβαια, θα ήταν δύσκολο να δημιουργηθεί και να διατηρηθεί ένα σύστημα διαχείρισης και αποθήκευσης των βιβλίων σε όλη την επικράτεια του πανεπιστημίου. Με την αρχιτεκτονική CLODA, ο ιδιοκτήτης κάποιου αντικειμένου το μόνο

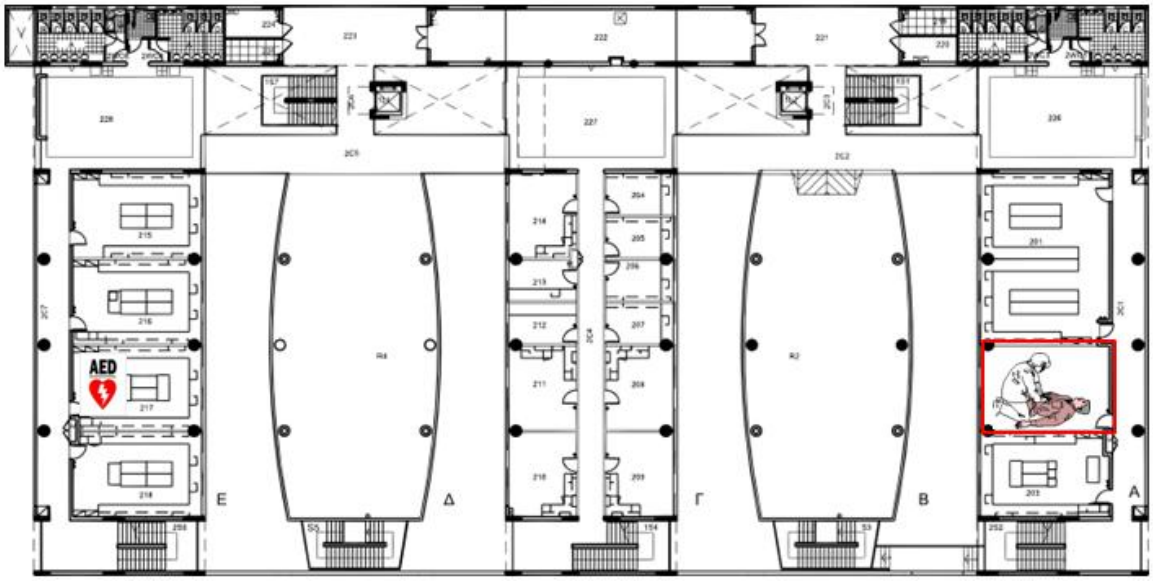


Κεφάλαιο 1 - Εικόνα 1.2
Σενάριο 1

που χρειάζεται να κάνει είναι να σαρώσει το γραμμικό κώδικα (barcode) του αντικειμένου αυτού. Τότε, αυτόματα το αντικείμενο θα τοποθετηθεί γεωγραφικά στο χάρτη, μέσα στο κτίριο και τη συγκεκριμένη αίθουσα. Οι μετά-πληροφορίες του αντικειμένου προσκομίζονται από το Σημασιολογικό ιστό (Semantic Web) όπως για παράδειγμα το Google API., και οι πληροφορίες συνδέονται και δημοσιοποιούνται σύμφωνα με το πρότυπο των διασυνδεδεμένων δημοσιοποιημένων πληροφοριών (LOD). Με αυτόν τον τρόπο, όταν ένας άλλος καθηγητής ή φοιτητής ζητάει πρόσβαση στο ίδιο αντικείμενο, η θέση του είναι ορατή σε ένα πλαίσιο χαρτογράφησης (π.χ. Google Maps), και ο χρήστης μπορεί να πλοηγηθεί εύκολα σε αυτό.

Σενάριο 2:

Το δεύτερο σενάριο αφορά ένα επείγον περιστατικό που λαμβάνει χώρα σε ένα δημόσιο χώρο (π.χ αεροδρόμιο, θέατρο κ.λ.π.). Ένας ασθενής με καρδιαγγειακά προβλήματα χρειάζεται άμεση προσοχή. Ακόμα και εάν υπάρχουν γιατροί ανάμεσα στο πλήθος, αδυνατούν να βοηθήσουν τον ασθενή εφόσον δεν έχουν τον απαραίτητο και κατάλληλο εξοπλισμό. Με την CLODA αρχιτεκτονική, χρησιμοποιώντας τη συγκεκριμένη εφαρμογή για έξυπνες κινητές συσκευές, ο χρήστης, στη συγκεκριμένη περίπτωση ένα άτομο από το πλήθος, θα μπορεί εύκολα να εντοπίσει τον κοντινότερο αυτόματο εξωτερικό απινιδωτή (AED) και ένα κουτί πρώτων βοηθειών. Ο εξοπλισμός έκτακτης ανάγκης θα έχει γεωοτοποθετηθεί από πριν μέσω ενός πράκτορα της κυβέρνησης και η ύπαρξή του θα έχει επιβεβαιωθεί από ένα αριθμό από επισκέπτες.



Κεφάλαιο 1 - Εικόνα 1.3
Σενάριο 2

Η εφαρμογή CLODA που αναπτύχθηκε, είχε ως σκοπό την ανάπτυξη μίας εύκολης ως προς τη χρήση εφαρμογής η οποία θα βοηθούσε το χρήστη να καταγράψει αντικείμενα τοποθετημένα σε διάφορους εσωτερικούς χ 18ώρους με τις σχετικές μετά-πληροφορίες αυτού του αντικειμένου που είναι διαθέσιμες εφόσον αυτό έχει σαρωθεί με τη χρήση γραμμωτού κώδικα (barcode).

Η βασική συνεισφορά δουλειάς που παρουσιάζεται στη διπλωματική αυτή εργασία είναι η εξής: Παρουσιάζεται η αρχιτεκτονική πίσω από το CLODA, ένα πλαίσιο το οποίο παρέχει πληθοποριστική κατασκευή διασυνδεδεμένων δημοσιοποιημένων πληροφοριών. Επιπλέον, γίνεται αναφορά στο τρόπο με τον οποίο μία ποικιλία από καινοτόμες τεχνολογίες μπορούν να ενσωματωθούν σε μία ενιαία εφαρμογή παρέχοντας τη δυνατότητα γεωτοποθέτησης αντικειμένων χρησιμοποιώντας συμβατικές έξυπνες κινητές συσκευές.

Το υπόλοιπο κείμενο είναι οργανωμένο ως εξής:

Στο Κεφάλαιο 2 γίνεται αναφορά στο υπόβαθρο και στη σχετική δουλειά που έχει γίνει μέχρι στιγμής όσον αφορά τις διάφορες τεχνολογίες που χρησιμοποιήθηκαν για την ανάπτυξη της συγκεκριμένης εφαρμογής. Ακολούθως, το Κεφάλαιο 3 αναφέρεται στις προδιαγραφές του συστήματος καθώς επίσης και στην περιγραφή της αρχιτεκτονικής CLODA και των διάφορων συστατικών που την αποτελούν. Στη συνέχεια, το Κεφάλαιο 4 περιγράφει αναλυτικά την υλοποίηση της εφαρμογής CLODA, δίνοντας περισσότερες πληροφορίες για τα συστατικά στοιχεία της αρχιτεκτονικής που περιγράφηκε προηγουμένως. Στο Κεφάλαιο 5, ακολουθεί η πειραματική μελέτη που έχει γίνει σχετικά με την ακρίβεια γεωτοποθέτησης ορόφων καθώς επίσης και τα αποτελέσματα ενός πειράματος που έχει γίνει προκειμένου να

διαπιστωθεί κατά πόσο και σε ποιο βαθμό η εφαρμογή αυτή υπήρξε χρήσιμη για τους χρήστες. Τέλος, στο Κεφάλαιο 6 γίνεται αναφορά στα συμπεράσματα που προέκυψαν αλλά και στις μελλοντικές επεκτάσεις που μπορεί να γίνουν με βάση αυτή την εφαρμογή.

Κεφάλαιο 2

Υπόβαθρο και Σχετική Δουλειά

2.1 Τεχνολογίες εσωτερικής τοποθέτησης

Οι άνθρωποι περνάνε σημαντικό μέρος από την καθημερινότητά τους σε εσωτερικούς χώρους και έτσι, έρχονται σε επαφή και αλληλεπιδρούν με αντικείμενα που βρίσκονται σε αυτούς. Η έρευνα όσον αφορά στην εσωτερική τοποθέτηση σημείωσε σημαντική πρόοδο τα τελευταία δέκα χρόνια, ως προς την ακρίβεια και την απαιτούμενη υποδομή, όμως, ακόμη δεν έχει προκύψει καμία κυρίαρχη λύση που να μπορεί να εφαρμοστεί. Αυτό, πιθανόν να οφείλεται στο ότι έχουμε έλλειψη από χώρους και σενάρια στα οποία η εσωτερική τοποθέτηση θα μπορούσε να φανεί χρήσιμη και αυτό γιατί μέχρι τώρα είναι περιορισμένη σε αεροδρόμια, εμπορικά κέντρα κ.ά. Στην αρχιτεκτονική CLODA και στα σενάρια που προαναφέρθηκαν, η ιδέα της εσωτερικής τοποθέτησης, αναδύεται ως το ιδανικό μέσο για

Κεφάλαιο 2 - Εικόνα 2.1
Airplace σε εσωτερικό χώρο

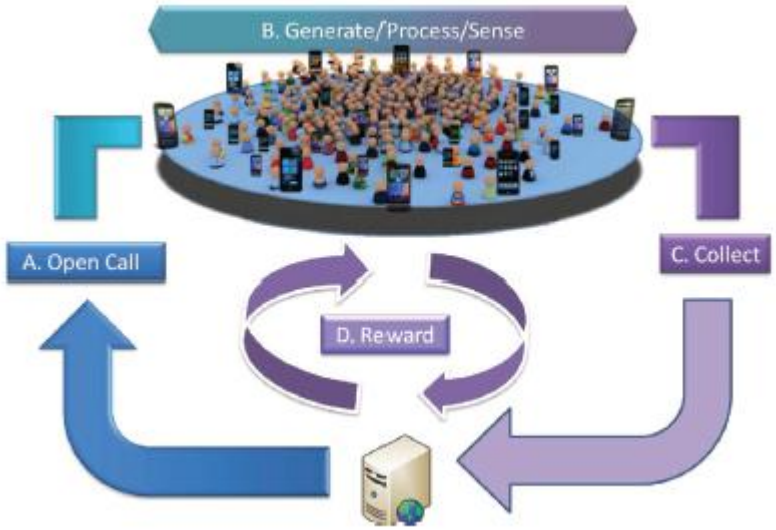
δημιουργία πληθωποριστικών διασυνδεδεμένων δημοσιοποιημένων πληροφοριών, τοποθετημένα πάνω σε χάρτη, ακόμα και σε χώρους όπου η εσωτερική τοποθέτηση μπορεί να μην φαινόταν τόσο ενδιαφέρουσα και δυνατόν να πραγματοποιηθεί.

Η σχετική βιβλιογραφία, όσον αφορά στην τοποθέτηση, είναι ευρέα και ποικιλόμορφη εφόσον εκμεταλλεύεται διάφορες τεχνολογίες. Το υποβοηθούμενο παγκόσμιο σύστημα εντοπισμού θέσης (A-GPS), είναι προφανώς παντού διαθέσιμο αλλά, είναι ακριβό όσον αφορά στην ενέργεια και επιπλέον, επηρεάζεται αρνητικά από το περιβάλλον (π.χ. συννεφιασμένες μέρες, δάση κ.λ.π.). Εκτός από το παγκόσμιο σύστημα εντοπισμού θέσης (GPS), η κοινότητα που ασχολείται με τις τοποθετήσεις [7], έχει προτείνει αρκετές ιδιωτικές λύσεις περιλαμβανομένων των υπερύθρων, Bluetooth, RFID, WANs κ.λ.π. περιλαμβάνοντας και τον συνδυασμό τους σε υβριδικά συστήματα [8]. Οι περισσότερες από αυτές τις τεχνολογίες, μπορούν να δώσουν υψηλού επιπέδου ακρίβεια τοποθέτησης αλλά, συνήθως χρειάζονται την ανάπτυξη και βαθμονόμηση ακριβών εξοπλισμών, όπως οι προσαρμοσμένοι πομποί και αντένες, οι οποίες είναι αφοσιωμένες στην τοποθέτηση.

2.2 Τεχνολογίες Πληθοπορισμού

Ο όρος πληθοπορισμός αναφέρεται, σε ένα νέο κατανεμημένο κοινωνικό συνεργατικό μοντέλο για επίλυση προβλημάτων και στο οποίο, ένα πλήθος ατόμων δεσμεύεται να επιλύσει ένα περίπλοκο πρόβλημα. Ο πληθοπορισμός μπορεί να διαχωριστεί σε δύο κατηγορίες: i) Πληθοπορισμός βασισμένος στον παγκόσμιο ιστό και ii) Κινητός πληθοπορισμός [3] . Στην CLODA αρχιτεκτονική, χρησιμοποιείται η δεύτερη κατηγορία αφού, παρέχει το πλεονέκτημα πληθοπορισμού βασισμένου σε τοποθεσία με αποτέλεσμα να ξεδιπλώνει όλες τις δυνατότητες πληθοπορισμού με επίγνωση θέσης. Οι εφαρμογές πληθοπορισμού σε έξυπνες κινητές συσκευές μπορούν να καταταγούν σε επεκτάσεις από εφαρμογές βασισμένες στον παγκόσμιο ιστό ή ως νέες εφαρμογές. Η πρώτη κατηγορία, επεκτείνεται σε χρήστες που δεν έχουν πρόσβαση σε συμβατικό σταθμό εργασίας και προσθέτει τη διάσταση της πραγματικής βασισμένης στην τοποθεσία πληροφορίας, στην υπηρεσία. Περιπτώσεις τέτοιων εφαρμογών είναι το Gigwalk, Jana και Ledlie et al. [9]. Η τελευταία κλάση περιλαμβάνει εφαρμογές για πληθωποριστική παρακολούθηση της κυκλοφορίας (π.χ. Waze) και υπολογισμό της καθυστέρησης οδικής κυκλοφορίας (VTrack [10]). Κατασκευάζουν λεπτομερή χάρτες θορύβου αφήνοντας τους χρήστες να ανεβάσουν δεδομένα που καταγράφηκαν από τα κινητά τους τηλέφωνα και μικρόφωνα (Ear-Phone [11]], NoiseTube [12]). Επιπλέον, αναγνωρίζουν τρύπες στο δρόμο επιτρέποντας στους χρήστες να μοιράζονται δόνηση και δεδομένα τοποθεσίας που λήφθηκαν από τα τηλέφωνά τους (PotHole [13]), αλλά και παιχνίδια βασισμένα σε τοποθεσίες με σκοπό τη συλλογή

γεωχωρικών δεδομένων (CityExplorer [14]). Αξιοποιούν τις έξυπνες κινητές συσκευές για συνεργατικό πρόγραμμα συμβουλευτικού οδικού σήματος (SignalGuru [15]) και τέλος, προσφέρουν νέες ευκαιρίες εργοδότησης σε εργάτες χαμηλού εισοδήματος σε αναπτυσσόμενες χώρες [16] [17] [18] και πραγματικού χρόνου λεπτομερή υπηρεσίες εσωτερικής τοποθέτησης, αξιοποιώντας την ισχύ του ραδιοσήματος (RSS), των σημείων πρόσβασης (Airplace [5]).



Κεφάλαιο 2 - Εικόνα 2.2
Πληθοπορισμός με έξυπνες κινητές συσκευές

2.3 Διασυνδεδεμένες Δημοσιοποιημένες Πληροφορίες

Οι διασυνδεδεμένες δημοσιοποιημένες πληροφορίες είναι η μέθοδος δημοσιοποίησης δομημένων δεδομένων (π.χ. RDF, XML) τα οποία μπορούν να διασυνδεθούν μέσω διάφορων ιστοσελίδων επιτρέποντας έτσι την ενοποίηση και αναζήτηση των δεδομένων [2]. Το CKAN που είναι ένα μητρώο ανοικτών δεδομένων και πακέτων περιεχομένου τα οποία και παρέχονται από το Ίδρυμα Ανοικτής Γνώσης (Open Knowledge Foundation), περιέχει τα περισσότερα από τα ανοικτά σύνολα δεδομένων που είναι δημοσίως διαθέσιμα και παρέχονται από μία ποικιλία από συγγραφείς. Για παράδειγμα το Freebase, που παρέχεται από την Google, δημιουργεί ένα γράφο οντότητας των ανθρώπων, τοποθεσίες και πράγματα ενώ, το YAGO είναι μία βάση γνώσης παραγόμενη από τη Wikipedia και τα GeoNames μεταξύ άλλων.

Εκτός από τις γενικές βάσεις δεδομένων διασυνδεδεμένων δημοσιοποιημένων πληροφοριών, υπάρχει ακόμη ένας μεγάλος αριθμός από άλλες πιο συγκεκριμένου περιεχομένου διασυνδεδεμένες δημοσιοποιημένες πληροφορίες (LOD). Περιπτώσεις τέτοιων διασυνδεδεμένων δημοσιοποιημένων πληροφοριών περιλαμβάνουν συνδεδεμένες βάσεις σχετιζόμενες με τη μουσική (π.χ. BBC Music), άλλες που σχετίζονται με τις δημοσιεύσεις

και την κοινότητα έρευνας (π.χ. DBLP), άλλες βάσεις που σχετίζονται με έργα ανοικτού κώδικα (π.χ. SECOLD), ανοικτές βιβλιοθήκες (π.χ. the Open Library), βάσεις δεδομένων ανοικτών αισθητήρων [19], πληροφορίες για πληθωποριστική διαθεσιμότητα χώρου στάθμευσης [20] και πληθωπορισιτκά καθήκοντα [21] μέσω της διαχείρισης συνδεδεμένων δεδομένων.

2.4 Μεγάλα δεδομένα-Εξόρυξη δεδομένων

Ο όρος Μεγάλα Δεδομένα αναφέρεται σε σύνολα δεδομένων των οποίων το μέγεθος και η δομή στελεχώνει τη δυνατότητα συχνά χρησιμοποιημένων σχεσιακών DBMSs να συλλάβουν, διαχειριστούν και να επεξεργαστούν τα δεδομένα μέσα σε ένα ανεκτό χρονικό διάστημα. Το μέγεθος των Μεγάλων Δεδομένων κυμαίνεται από μερικές δωδεκάδες terabytes μέχρι αρκετά petabytes σε μία μόνο βάση, και το προσδιοριστικό μοντέλο δεδομένων μπορεί να είναι οτιδήποτε από δομημένο (σχεσιακό ή πίνακας) έως και ημι-δομημένο (XML ή JSON) ή ακόμα αδόμητο (web text και log files). Οι αρχιτεκτονικές των Μεγάλων Δεδομένων είναι εξαιρετικά παράλληλες και κατανεμημένες έτσι ώστε να μπορούν να αντεπεξέλθουν στους περιορισμούς όσον αφορά στα I/O και την κεντρική μονάδα επεξεργασίας. Τέτοια συστήματα εκτελούνται σε μεσαίας κλίμακας ιδιωτικές νεφέλες, προσφέροντας υψηλότερη ασφάλεια και σε μεγάλης κλίμακας δημόσιες νεφέλες, εκθέτοντας λειτουργικές και αναλυτικές λειτουργικότητες ως αυτόνομες αλλά και ως υπηρεσίες.
Η εξέλιξη αλλά και το ιδανικό πάντρεμα των Μεγάλων Δεδομένων (Big Data) και του σημασιολογικού Ιστού (Semantic Web) είναι ικανό να παρέχει τις μετά-πληροφορίες που είναι απαραίτητες για την ευρεία υιοθέτηση του Σημασιολογικού Ιστού. Το Mega-modelling [22] προτάθηκε ως νέο μοντέλο διαχείρισης συστήματος, για την απόκτηση, σύνθεση, ένταξη, διαχείριση, αναζήτηση και εξόρυξη των δεδομένων Το GLADE [23], ένα κλιμακωτό κατανεμημένο σύστημα για αναλυτικές λειτουργίες δεδομένων μεγάλης κλίμακας, παίρνει αναλυτικές συναρτήσεις που έχουν εκφραστεί μέσω της διεπαφής User-Defined Aggregate (UDA), και εκτελεί όλα αυτά αποτελεσματικά στα δεδομένα εισόδου. Επιπρόσθετα, οι στόχοι των Μεγάλων Δεδομένων, όπως παρουσιάζονται στο [24], υπογραμμίζουν τη σημαντικότητα της ανάπτυξης νέων αναλυτικών εργαλείων ικανών να εξαγάγουν μετά-πληροφορίες, που είναι απαραίτητες για τις διασυνδεδεμένες δημοσιοποιημένες πληροφορίες (LOD), με ένα γρήγορο και αποδοτικό, από πλευράς κόστους, τρόπο από τεράστια σύνολα δεδομένων.

Κεφάλαιο 3

Προδιαγραφή Απαιτήσεων και Αρχιτεκτονική

3.1 Προδιαγραφή Απαιτήσεων

3.1.1 Στόχος Εργασίας

Ο στόχος της διπλωματικής αυτής εργασίας είναι η δημιουργία μίας αρχιτεκτονικής πληθοπορισμού για διαχείριση αντικειμένων σε εσωτερικούς χώρους. Η αρχιτεκτονική αυτή θα παρέχει τη δυνατότητα στους χρήστες να γνωρίζουν τη θέση τους μέσα στο κτίριο της Πληροφορικής ανά πάσα χρονική στιγμή, με τη βοήθεια της βιβλιοθήκης εσωτερικής τοποθέτησης AirPlace, καθώς επίσης και να γεωτοποθετούν αντικείμενα μέσα στις αίθουσες στους διάφορους ορόφους. Με αυτό τον τρόπο θα μπορούν να γνωρίζουν την ακριβή θέση του αντικειμένου και έτσι θα μπορούν να κατευθυνθούν πιο εύκολα σε αυτό. Επιπλέον, ο χρήστης θα μπορεί να μετακινεί, να διαγράφει και να μορφοποιεί τις πληροφορίες των αντικειμένων.

3.1.2 Περιγραφή

Για τη χρήση της εφαρμογής, το μόνο που χρειάζεται είναι η εγκατάστασή της σε κάποια έξυπνη κινητή συσκευή. Έτσι, ο χρήστης θα μπορεί αμέσως, με την εκκίνηση της εφαρμογής, να πλοηγηθεί στις αίθουσες των διαφόρων ορόφων του κτιρίου και να γεωτοποθετήσει ή να εντοπίσει κάποιο αντικείμενο.

3.1.2.1 Πληροφορίες αντικειμένου

Κάθε αντικείμενο εφόσον σαρωθεί με τη βοήθεια γραμμικού κώδικα (barcode), αποθηκεύεται στη βάση δεδομένων του συστήματος. Αυτό σημαίνει πως τα σημαντικότερα στοιχεία του συγκεκριμένου αντικειμένου καταγράφονται και στη συνέχεια, όταν ο χρήστης επιλέξει κάποιο αντικείμενο, προβάλλονται σε αυτόν σε ένα παράθυρο πληροφοριών. Συγκεκριμένα, με τη σάρωση του αντικειμένου, αυτό αναζητείται στο Google Shopping API και σε περίπτωση που ανευρεθεί, τότε λαμβάνονται τα σημαντικότερα του στοιχεία. Σε περίπτωση που το αντικείμενο δεν έχει ανευρεθεί ή λείπουν κάποια από τα βασικά στοιχεία του, ζητείται από το χρήστη να τα εισαγάγει. Τα στοιχεία αυτά είναι τα πιο κάτω:

Barcode: Ο γραμμικός κώδικας του αντικειμένου, που είναι μοναδικός για κάθε αντικείμενο ίδιου τύπου.
Price: Η τιμή του αντικειμένου
Title: Ο τίτλος του αντικειμένου (π.χ. οθόνη, βιβλίο – τίτλος βιβλίου)
Brand: Η μάρκα του αντικειμένου
Πέρα από αυτά τα στοιχεία, για κάθε αντικείμενο αποθηκεύεται και η θέση του στον παγκόσμιο χάρτη (γεωγραφικό μήκος και πλάτος) αλλά και ο όροφος στον οποίο βρίσκεται το συγκεκριμένο αντικείμενο

3.1.3 Λειτουργίες Συστήματος

3.1.3.1 Σάρωση Αντικειμένων

Μία από τις βασικές λειτουργίες του συστήματος είναι η σάρωση αντικειμένων και η γεωτοποθέτησή τους στους χώρους του κτιρίου. Η σάρωση των αντικειμένων επιτυγχάνεται με τη χρήση μιας εξωτερικής βιβλιοθήκης, της Zxing library με την οποία ενσωματώνεται στην εφαρμογή ένας σαρωτής γραμμικού κώδικα (barcode) σε συνδυασμό με τη camera της κινητής συσκευής.
Ο χρήστης έχει τη δυνατότητα σε περίπτωση αρκετών αποτυχημένων προσπαθειών σάρωσης κάποιου αντικειμένου να χρησιμοποιήσει χειροκίνητη σάρωση, όπου ζητείται από αυτόν να καταχωρήσει από μόνο του τον αριθμό του γραμμικού κώδικα (barcode). Στην εφαρμογή μπορεί να γίνει εύκολη εναλλαγή μεταξύ χειροκίνητης και αυτόματης σάρωσης με τη χρήση ενός κουμπιού από το μενού.
Εάν γίνει επιτυχής σάρωση του αντικειμένου, η εφαρμογή εντοπίζει τα στοιχεία του από το Google Shopping API και τα εμφανίζει στο χρήστη ο οποίος αφού τα εγκρίνει αποθηκεύονται στη βάση δεδομένων του συστήματος. Εάν όμως δεν εντοπιστούν όλα ή μερικά από τα στοιχεία του αντικειμένου, τότε ζητείται από τον χρήστη να τα εισαγάγει ο ίδιος χειροκίνητα. Σε περίπτωση που κάποιο αντικείμενο υπάρχει ήδη στη βάση δεδομένων, εμφανίζεται στο χρήστη ένα μήνυμα ενημέρωσης στο οποίο ο χρήστης ρωτάται κατά πόσο επιθυμεί να καταχωρήσει ξανά το αντικείμενο αλλά, σε διαφορετική θέση.

3.1.3.2 Εντοπισμός χρήστη στο χώρο

Η δεύτερη βασική λειτουργία της εφαρμογής είναι ο εντοπισμός της θέση του χρήστη μέσα στο κτίριο. Για την υλοποίηση αυτής της λειτουργίας χρησιμοποιείται στην εφαρμογή η βιβλιοθήκη εσωτερικής τοποθέτησης Airplace. Ο χρήστης έχει τη δυνατότητα, εφόσον

εντοπίσει κάποιο αντικείμενο το οποίο χρειάζεται και πρέπει να αποκτήσει πρόσβαση σε αυτό, να ενεργοποιήσει τη λειτουργία εντοπισμού του στο κτίριο. Συγκεκριμένα, με αυτή τη λειτουργία θα μπορεί να βλέπει σε ποιο ακριβώς σημείο βρίσκεται στο κτίριο και πόσο κοντά είναι σε σχέση με το αντικείμενο που ψάχνει. Επιπλέον, παρέχεται η δυνατότητα να εντοπίσει απλώς σε ποιο σημείο από όλο το κτίριο βρίσκεται σε κάποια στιγμή.

3.1.3.3 Αυτόματη αλλαγή ορόφου

Σημαντικό κομμάτι της εφαρμογής αυτής αποτελεί και ο αυτόματος εντοπισμός ορόφου. Για την πραγμάτωση αυτής της λειτουργίας, χρησιμοποιείται ένας αλγόριθμος ο οποίος και περιγράφεται αναλυτικά σε πιο κάτω τμήμα αυτού του κεφαλαίου. Η λειτουργία αυτή, παρέχει στο χρήστη τη δυνατότητα να κινείται στο κτίριο χωρίς να πρέπει κάθε φορά να αλλάζει ο ίδιος την κάτοψη του ορόφου ανάλογα σε ποιο όροφο βρίσκεται. Μόλις εντοπιστεί ο σωστός όροφος, αφαιρείται η προηγούμενη κάτοψη και τοποθετείται η νέα.

3.1.4 Λειτουργίες που αφορούν στη διαχείριση των αντικειμένων

Κάθε αντικείμενο το οποίο έχει σαρωθεί αλλά και κάθε άλλο αντικείμενο το οποίο πιθανό να σαρωθεί στο μέλλον, έχει κάποιες λειτουργίες.

3.1.4.1 Μορφοποίηση Αντικειμένων

Όπως αναφέρθηκε και πιο πάνω, με τη σάρωση κάθε αντικειμένου καταγράφονται στη βάση δεδομένων τα στοιχεία του. Σε περίπτωση που το αντικείμενο δεν εντοπιστεί στο Google Shopping API , ο χρήστης χειροκίνητα καταχωρεί τα στοιχεία του αντικειμένου. Αυτό έχει ως αποτέλεσμα κάποιες φορές είτε λόγω απροσεξίας ή κεκτημένης ταχύτητα, να καταχωρηθούν λανθασμένα μερικά ή και όλα τα στοιχεία. Έτσι, με αυτή τη λειτουργία, δίνεται η δυνατότητα στους χρήστες εφόσον επιλέξουν κάποιο αντικείμενο και αφού πατήσουν στο παράθυρο πληροφοριών του, να μορφοποιήσουν κάποια ή όλα τα πεδία του εκτός βέβαια από τη θέση του. Αυτό γιατί δεν είναι ανθρωπίνως δυνατό κάποιος να γνωρίζει επακριβώς σε ποια θέση αντιστοιχεί κάποιο γεωγραφικό μήκος και πλάτος.

3.1.4.1.1 Διαγραφή Αντικειμένων

Μία επιπλέον λειτουργία στη διάθεση του χρήστη είναι και η δυνατότητα διαγραφής αντικειμένων από τη βάση δεδομένων. Συγκεκριμένα, μπορεί κάποιος χρήστης κατά λάθος

να τοποθέτησε κάποιο αντικείμενο που τελικά δε χρειαζόταν ή ακόμη, μπορεί κάποιο αντικείμενο να έχει αφαιρεθεί από το χώρο στον οποίο και βρισκόταν. Έτσι, εφόσον δεν είναι πλέον χρήσιμο, ο χρήστης μπορεί να το διαγράψει.

3.1.4.1.2 Μετακίνηση Αντικειμένων

Πολλές φορές, κάποιο αντικείμενο π.χ. μία οθόνη υπολογιστή ή ένα βιβλίο, μπορεί να μετακινηθεί από την αρχική του θέση. Αυτό σημαίνει πως το σύστημα θα πρέπει να ενημερωθεί έτσι ώστε κάποιος ο οποίος ψάχνει το συγκεκριμένο αντικείμενο να μπορέσει να το εντοπίσει και όχι να το ψάχνει σε λάθος θέση. Ένας λόγος που παρέχεται αυτή η δυνατότητα στο χρήστη είναι αυτός. Επιπλέον, εφόσον δεν καθίσταται δυνατόν να γνωρίζει κάποιος τις συντεταγμένες οποιασδήποτε θέσης και μπορεί να παραστεί ανάγκη να μετακινηθεί κάποιο αντικείμενο, με αυτή τη λειτουργία πολύ εύκολα και γρήγορα μπορεί να επιτευχθεί κάτι τέτοιο. Εφόσον ο χρήστης επιλέξει τη λειτουργία αυτή, θα του εμφανιστεί ο χάρτης με το κτίριο και μόλις επιλέξει νέα τοποθεσία το αντικείμενο θα μετακινηθεί και αυτόματα, θα ενημερωθεί και η βάση δεδομένων για την αλλαγή.

3.1.4.1.3 Μενού στο Action Bar

Με την έναρξη της εφαρμογής, παρουσιάζεται στο χρήστη ο χάρτης με το κτίριο της Πληροφορικής όπως αυτός φαίνεται στα Google Maps, εμφανίζεται η κάτοψη του ορόφου στον οποίο βρίσκεται ο χρήστης καθώς επίσης κι όλα τα αντικείμενα που έχουν σαρωθεί σε αυτό τον όροφο. Στο πάνω μέρος, υπάρχει ένα ActionBar Menu το οποίο και έχει τις εξής επιλογές:

Find Me: Με την επιλογή αυτή ο χρήστης μπορεί να δει σε ποιο σημείο του κτιρίου βρίσκεται για μια στιγμή

Track Me: ΜΕ αυτή την επιλογή, ο χρήστης εντοπίζει τον εαυτό του ανά πάσα χρονική στιγμή μέσα στο κτίριο, δηλαδή καθώς κινείται.

Scan Item: Επιλέγοντας αυτό το κουμπί, ο χρήστης μπορεί να σαρώσει οποιοδήποτε αντικείμενο στο κτίριο το οποίο φέρει γραμμικό κώδικα (barcode).

Hide/Show Floorplan/Items: Η επιλογή αυτή επιτρέπει στο χρήστη να εμφανίσει και να εξαφανίσει τόσο τα αντικείμενα όσο και την κάτοψη του ορόφου. Αυτό πιθανόν να είναι χρήσιμο σε περίπτωση που κάποιος επιθυμεί να σαρώσει κάποιο αντικείμενο αλλά λόγω του ότι ήδη υπάρχουν αρκετά αντικείμενα στη συγκεκριμένη αίθουσα να μην μπορεί να δει καθαρά τα σημεία στα οποία θα μπορούσε να τοποθετήσει το συγκεκριμένο αντικείμενο.

Main Menu: Η επιλογή αυτή δίνει πρόσβαση στο χρήστη, στο κυρίως μενού της εφαρμογής.

Logger: Χρησιμοποιείται για να ληφθούν μετρήσεις στο χώρο ώστε στη συνέχεια να είναι δυνατός ο εντοπισμός του χρήστη στο κτίριο.

Preferences: Μέσω αυτής της επιλογής, μπορεί ο χρήστης να επιλέξει κατά πόσο επιθυμεί να είναι ενεργοποιημένη η λειτουργία για τον αυτόματο εντοπισμό ορόφου ή όχι, αλλά και να επιλέξει το σωστό αρχείο με τις μετρήσεις ώστε να μπορεί να εντοπίζει τον εαυτό του καθώς κινείται.

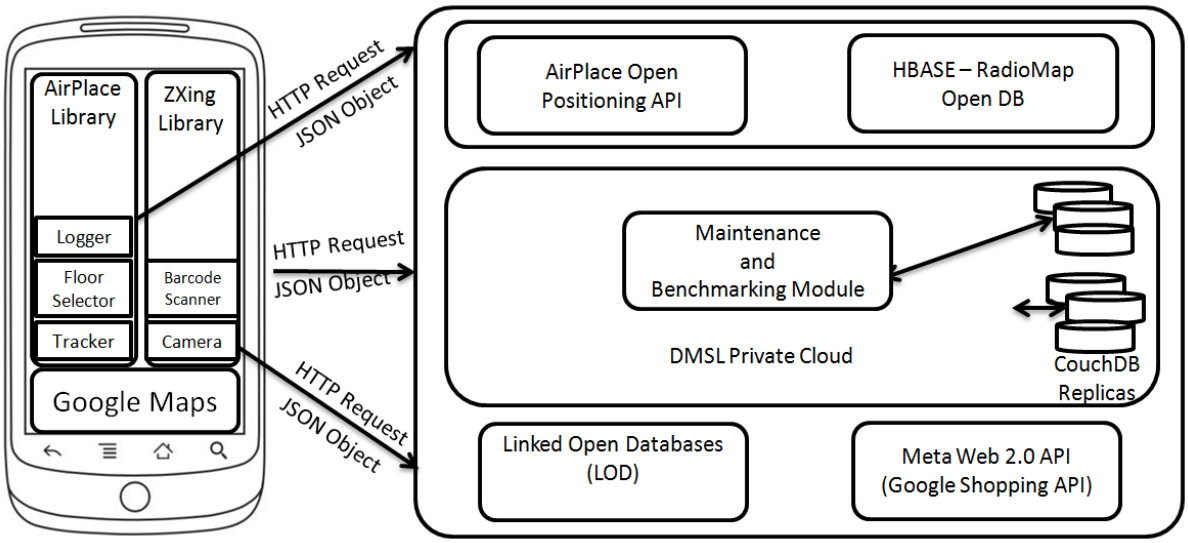
Help: Η επιλογή αυτή επεξηγεί στο χρήστη τη λειτουργία κάθε κουμπιού για να ευκολύνει τη χρήση της εφαρμογής.

About: Δίνει σχετικές πληροφορίες όσον αφορά στη δημιουργία αυτής της εφαρμογής.

Exit: Έξοδος από την εφαρμογή

3.2 Αρχιτεκτονική Συστήματος

Η αρχιτεκτονική CLODA είναι η πρώτη προσπάθεια συνδυασμού του πληθοπορισμού, της τοπικοποίησης και των υπηρεσιών βασισμένων στην τοποθεσία, για δημιουργία, συλλογή, επικύρωση και συσχέτιση πραγματικών, γεω-χωρικών και πολυδιάστατων πληροφοριών. Η CLODA έχει εμπνευστεί τόσο από το DBpedia [25], μία προσπάθεια πληθωποριστικής κοινότητας για εξαγωγή δομημένων πληροφοριών από τη Wikipedia και τη γεωγραφική βάση δεδομένων GeoNames, που είναι ακόμη μία προσπάθεια πληθωποριστικής κοινότητας



Κεφάλαιο 3 - Εικόνα 3.1
Τα δομικά στοιχεία της πλευράς του πελάτη και του εξυπηρετητή της αρχιτεκτονικής CLODA

η οποία ενσωματώνει γεωγραφικά δεδομένα όπως, ονόματα τοποθεσιών σε διαφορετικές γλώσσες, πληθυσμό κ.ά.

Κίνητρο για τη δημιουργία αυτής της αρχιτεκτονικής υπήρξε το γεγονός ότι ακόμη και αν γίνονται αρκετές προσπάθειες για δημιουργία εφαρμογών διαχείρισης αποθέματος (π.χ.

Inventory Tracker, Barcode & Inventory Pro και Inventory Android εφαρμογές), όλες αυτές επικεντρώνονται στον χρήστη σαν ξεχωριστή οντότητα και όχι στα οφέλη που θα μπορούσε να αποφέρει στο πλήθος γενικά. Ως αποτέλεσμα, όλα τα δεδομένα αποθηκεύονται ιδιωτικά για την προσωπική χρήση του κάθε χρήστη και δεν υπάρχει κάποιος τρόπος να γίνουν διαθέσιμα και σε άλλους χρήστες. Ο στόχος της CLODA αρχιτεκτονικής είναι να παρέχει τη δυνατότητα απόκτησης ανοικτών δεδομένων χρησιμοποιώντας τον πληθωπορισμό, δηλαδή το πλήθος.

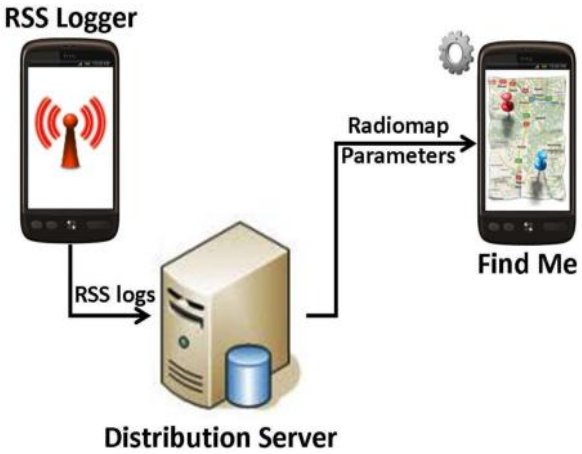
Οι χρήστες της εφαρμογής CLODA έχουν τη δυνατότητα να συνεισφέρουν με τη γεωτοποθέτηση έγκυρων μετά-πληροφοριών για διάφορα αντικείμενα, με τη χρήση των κινητών τους τηλεφώνων. Οι πληροφορίες αυτές ακολούθως, αποθηκεύονται σε μία κατανεμημένη συλλογή από βάσεις δεδομένων και είναι προσβάσιμες στο κοινό. Ιδιοκτήτης των αντικειμένων αποτελεί το πλήθος, πρόθυμο να συνεισφέρει σε ένα απόθεμα που είναι προσβάσιμο σε όλους, για ένα ηθικό όφελος. Για παράδειγμα, στο πρώτο σενάριο που προανέφερα, ο ιδιοκτήτης των αντικειμένων είναι το πλήθος (π.χ. προσωπικό, φοιτητές κ.λ.π) ενώ στο δεύτερο σενάριο, ο ιδιοκτήτης μπορεί να είναι μία κυβερνητική αρχή ή ένας δημόσιος χώρος (π.χ. αεροδρόμιο, μουσείο) πρόθυμος να συνεισφέρει στο κοινό αποθετήριο.

Η CLODA είναι οργανωμένη σε μία αρχιτεκτονική αποτελούμενη από τρία επίπεδα, οποία και είναι, η εφαρμογή σε Android λειτουργικό σύστημα, μερικοί υψηλά διαθέσιμοι εξυπηρετητές (servers) και ο συνδυασμός του τμήματος πελάτη και το αντίστοιχο τμήμα εξυπηρετητή (server), όπως φαίνονται και στην εικόνα 3.1.

Πιο κάτω θα ακολουθήσει πιο αναλυτική περιγραφή κάθε τμήματος της CLODA αρχιτεκτονικής.

3.3 Airplace Indoor Positioning

3.3.1 Περιγραφή Airplace βιβλιοθήκης



Κεφάλαιο 3 - Εικόνα 3.2
Αρχιτεκτονική Airplace

Η εφαρμογή CLODA χρησιμοποιεί τη βιβλιοθήκη Airplace [5], η οποία εκμεταλλεύεται τις τιμές της ισχύος του λαμβανόμενου σήματος (RSS), οι οποίες και λήφθηκαν μέσω της παθητικής σάρωσης των πακέτων εκπομπής που μεταδόθηκαν από γειτονικά APs ως μέρος της βασικής λειτουργικότητας του δικτύου. Για την αντιμετώπιση των δύσκολων συνθηκών διάδοσης σήματος στους εσωτερικούς χώρους, λόγω

πολλαπλών διαδρομών, αντανακλάσεων και διαθλάσεων, τα RSS αποτυπώματα (δηλαδή διανύσματα ή RSS μετρήσεις καταγεγραμμένες από APs στην περιοχή του χρήστη), συλλέγονται από προηγουμένως σε προκαθορισμένες θέσεις αναφοράς. Κάθε RSS αποτύπωμα συσχετίζεται με την αντίστοιχη θέση αναφοράς και αποθηκεύεται στο Radiomap, το οποίο καλύπτει όλη την περιοχή ενδιαφέροντος. Στο τέλος, το Radiomap αντιστοιχείται από τον πολυδιάστατο RSS χώρο στις φυσικές συντεταγμένες. Το Airplace, ξεπερνά τα μειονεκτήματα των συστημάτων εσωτερικής τοποθέτησης . Ένα σημαντικό πλεονέκτημα της εφαρμογής CLODA, σε σχέση με οποιαδήποτε άλλη υπάρχουσα εφαρμογή, είναι η ικανότητα προβολής των αντικειμένων πάνω από ένα πλαίσιο χαρτογράφησης (π.χ. Google Maps).

Η αρχιτεκτονική του Airplace φαίνεται στο πιο πάνω σχήμα, και αποτελείται από το Logger, τον Tracker (Find Me) καθώς επίσης και τον εξυπηρετητή διανομής (distribution server).

3.3.2 Logger

Η εφαρμογή CLODA αποτελείται από δύο κύρια μέρη, τον Logger και τον Tracker. Ο Logger, έχει αναπτυχθεί γύρω από το Android RSS API για σάρωση και καταγραφή δειγμάτων δεδομένων σε συγκεκριμένες περιοχές σε προκαθορισμένα διαστήματα. Αυτά τα δείγματα περιέχουν MAC διευθύνσεις και RSS επίπεδα όλων τν γειτονικών WLAN APs, καθώς επίσης και τις συντεταγμένες της περιοχής στην οποία ο χρήστης άρχισε την καταγραφή. Τα δεδομένα που συλλέγει ο χρήστης αποθηκεύονται τοπικά σε log files, τα οποία τυγχάνουν επεξεργασίας από τον εξυπηρετητή διανομής προκειμένου να παραχθούν τα κατάλληλα Radiomaps.

3.3.3 Tracker

Το δεύτερο μέρος της εφαρμογής είναι ο Tracker, ο οποίος και χρησιμοποιείται προκειμένου ο χρήστης να μπορεί ανά πάσα χρονική στιγμή να βλέπει τη θέση του στο κτίριο. Για να γίνει αυτό, χρησιμοποιούνται τα Radiomaps που παράγονται από τον εξυπηρετητή διανομής, μετά που ο χρήστης πήρε τις κατάλληλες μετρήσεις χρησιμοποιώντας τον Logger. Έχουν υλοποιηθεί αρκετοί αλγόριθμοι βασισμένοι στα αποτυπώματα (fingerprints), περιλαμβανομένου του ντετερμινιστικού K-nearest Neighbor (KNN) και του Weighted K-Nearest Neighbor (WKNN) αλγορίθμου, καθώς επίσης και του πιθανοτικού Maximum A Posteriori (MAP) και του Minimum Mean Square Error (MMSE) αλγορίθμων. Η πλατφόρμα αυτή, υποστηρίζει δύο εξαιρετικούς αλγορίθμους, τον Radial Basis Function (RBF) και τον Subtract onNegative Add on Positive (SNAP).

Περισσότερες λεπτομέρειες για το κάθε μέρος θα δοθούν σε επόμενο κεφάλαιο.

3.3.4 Εξυπηρετητής Διανομής

Ο εξυπηρετητής διανομής είναι κυρίως υπεύθυνος για την κατασκευή και διανομή του RSS Radiomap ακούγοντας για συνδέσεις από τους πελάτες που είτε συνεισφέρουν τα RSS που μάζεψαν ή, ζητώντας το Radiomap και τις παραμέτρους του αλγορίθμου για να αρχίσει η τοποθέτηση. Προκειμένου να δημιουργήσει το radiomap αρχείο, ο εξυπηρετητής αναλύει όλα τα διαθέσιμα RSS log αρχεία, τα οποία μπορεί να έχουν δοθεί από διάφορους χρήστες, και υπολογίζει τη μέση RSS τιμή ανά MAC διεύθυνση, υπολογίζοντας το μέσο όρο όλων των δειγμάτων που συλλέχθηκαν σε κάθε διακριτή περιοχή. Τέλος, ο μέσος όρος των τιμών συγκλίνει και αποθηκεύεται σε ένα μοναδικό radiomap αρχείο έτσι ώστε κάθε γραμμή να ανταποκρίνεται στη μέση RSS τιμή στη συγκεκριμένη τοποθεσία και, κάθε στήλη στην αντίστοιχη MAC διεύθυνση. Με αυτή την προεπεξεργασία (δηλαδή τον μέσο όρο), μειώνεται ο θόρυβος. Επιπλέον, υπολογίζοντας τον μέσο όρο, το RSS Radiomap, μειώνεται αποδοτικά, και έτσι μειώνεται και η καθυστέρηση για τη λήψη του Radiomap.

3.3.5 Floor Selector

3.3.5.1 Περιγραφή Αλγορίθμου Επιλογής Ορόφου

Ο αρχικός αλγόριθμος εντοπίζει κάθε φορά σε ποιο όροφο βρίσκεται ο χρήστης με βάση το μέσο όρο των rss τιμών κάθε mac address, ως προς τον αριθμό των έγκυρων mac addresses(οποιαδήποτε τιμή εκτός από -110). Αρχικά, προκειμένου να είναι δυνατός ο εντοπισμός του σωστού ορόφου, είναι απαραίτητη η δημιουργία των αρχείων .floor. Τα αρχεία αυτά, δημιουργούνται με βάση τις μετρήσεις που πήρε ο χρήστης χρησιμοποιώντας τον logger, όπου μετά την καταγραφή των rss logs και με τη βοήθεια του server παράγεται το αρχείο radiomaps-mean το οποίο και λαμβάνεται ως παράμετρος εισόδου από το πρόγραμμα που είναι υπεύθυνο για τη δημιουργία των αρχείων .floor. Αφού δημιουργηθούν αυτά τα αρχεία, εισάγονται στον κατάλογο assets της εφαρμογής που θα τα χρησιμοποιήσει.
Ο δεύτερος αλγόριθμος ακολουθεί την ίδια διαδικασία όπως περιγράφηκε πιο πάνω με τη διαφορά ότι για τη δημιουργία των .floor αρχείων, δε χρησιμοποιήθηκε ο μέσος όρος των rss values ως προς τον αριθμό των έγκυρων mac addresses αλλά, από την αρχή γινόταν ο εντοπισμός της μέγιστης τιμής rss value. Στην πιο κάτω εικόνα φαίνεται η δυνατότητα που δίνεται στο χρήστη να ενεργοποιήσει ή όχι τον αυτόματο εντοπισμό ορόφου.

Κεφάλαιο 3 - Εικόνα 3.3
Επιλογή ενεργοποίησης αυτόματης εύρεσης ορόφου στην εφαρμογή

3.3.5.2 Εύρεση Ορόφου στην εφαρμογή CLODA

Για την εύρεση του σωστού ορόφου στον οποίο βρίσκεται ο χρήστης, υπάρχει ένα service το οποίο τρέχει στο πίσω μέρος της εφαρμογής και ελέγχει κάθε 5 sec κατά πόσο ο χρήστης βλέπει το floorplan που αντιστοιχεί στον όροφο που βρίσκεται.

Αρχικά υπάρχει ένας πίνακας δύο θέσεων που αποθηκεύει στη μια θέση τον παρόντα όροφο και στην δεύτερη θέση την βαθμολογία του ορόφου, καθώς επίσης και ένας επιπλέον πίνακας στον οποίο και αποθηκεύονται οι ορόφοι οι οποίοι θα συγκριθούν αμέσως μετά.

Στη συνέχεια, διαβάζονται όλα τα .floor αρχεία από τον κατάλογο assets της εφαρμογής και αποθηκεύονται και αυτά σε ένα ArrayList, σε κάθε θέση του οποίου βρίσκεται ο αριθμός του ορόφου και ένα ArrayList για κάθε όροφο το οποίο και περιέχει τις μετρήσεις του συγκεκριμένου ορόφου, δηλαδή τα rss values.

Προκειμένου κάθε φορά να εντοπίζεται ο σωστός όροφος, υπάρχει και ένας δυσδιάστατος πίνακας ο οποίος κρατάει τον όροφο με το score του αντίστοιχα(4X2). Κάθε φορά λαμβάνονται δύο ορόφοι για σύγκριση ώστε να καθοριστεί σε ποιο όροφο βρίσκεται ο χρήστης.

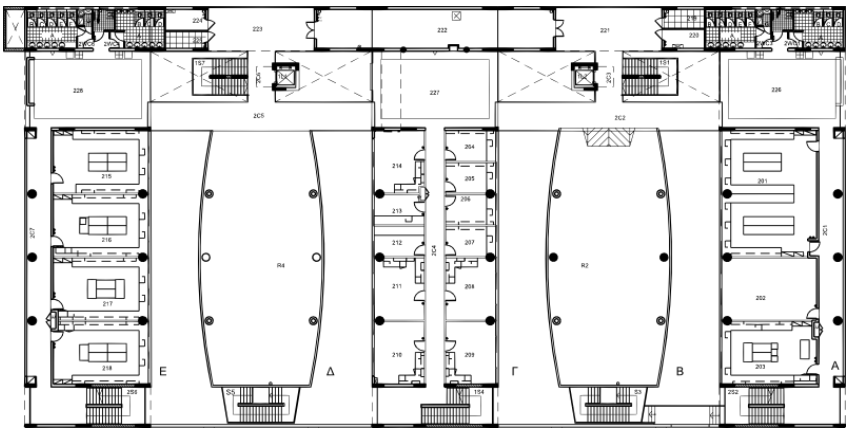
Ακολούθως, εντοπίζεται το πιο κοντινό mac address με βάση μία λίστα με τη πιο πρόσφατη καταγραφή access points. Γίνεται αναζήτηση του μέγιστου rss value και αφού εντοπιστεί

ακολουθεί ο έλεγχος των rss values κάθε ορόφου και έτσι εντοπίζεται ο όροφος στον οποίο ανήκει αυτό με τη μεγαλύτερη τιμή.

Στη συνέχεια, ελέγχεται η περίπτωση το μέγιστο rss value, που η συσκευή κατάγραψε την τρέχουσα χρονική στιγμή, να υπάρχει στα .floor αρχεία και των δύο ορόφων. Εάν σε ένα από τα δύο .floor αρχεία υπάρχει το συγκεκριμένο mac address τότε επιλέγεται ο όροφος που ανήκει το συγκεκριμένο αρχείο.

Εάν όμως το συγκεκριμένο mac address υπάρχει και στα δύο αρχεία τότε ελέγχονται το rss value που υπάρχει στα .floor αρχεία των δύο ορόφων που έχουν το ίδιο mac address. Εφόσον εντοπιστεί ο όροφος με το μεγαλύτερο rss value, αποθηκεύεται η βαθμολογία κάθε ορόφου (δηλαδή ο μέσος όρος της rss τιμής του συγκεκριμένου mac address από τον συγκεκριμένο όροφο) στον κατάλληλο πίνακα. Είναι σημαντικό να αναφερθεί ότι το rss value για το σωστό όροφο αποθηκεύεται στον πίνακα και επίσης μία τυχαία τιμή για τον άλλο όροφο ο οποίος είναι ο λανθασμένος π.χ. -800.

Στο τέλος, γίνεται έλεγχος των δύο ορόφων σε σχέση με τη μέγιστη βαθμολογία που υπήρχε από προηγουμένως, και αποθηκεύεται στον αντίστοιχο πίνακα ο όροφος και το score του. Τέλος, γίνεται έλεγχος κατά πόσο ο όροφος που εντοπίστηκε είναι ο ίδιος με τον τρέχον όροφο και αναλόγως γίνεται ή όχι αλλαγή του floorplan του κτιρίου.



Κεφάλαιο 3 - Εικόνα 3.4
Κάτοψη δευτέρου ορόφου

3.4 Βάση Ημιδομημένων Εγγράφων

3.4.1 JavaScript Open Notation - JSON

Το JSON [33], είναι μία ελαφριά μορφή ανταλλαγής δεδομένων η οποία και είναι εύκολο για τους ανθρώπους και να τη διαβάζουν αλλά και να τη γράφουν. Ακόμη, είναι εύκολο για τις μηχανές να την αναλύσουν και να τη δημιουργήσουν. Το JSON είναι μία μορφή κειμένου η οποία είναι τελείως ανεξάρτητη από οποιαδήποτε γλώσσα αλλά, χρησιμοποιεί συμβάσεις οι

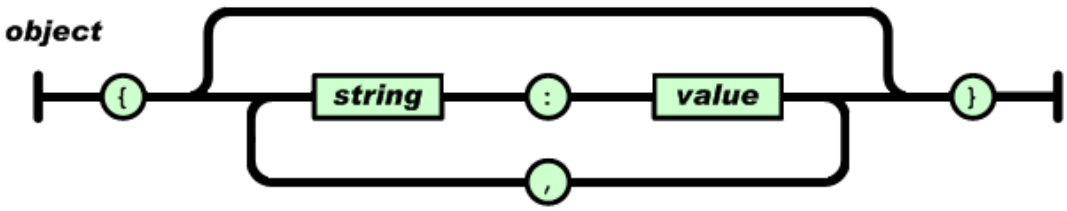
οποίες είναι γνωστές στους προγραμματιστές της C, C++, C#, Java, JavaScript, Perl, Python. Αυτές οι ιδιότητες κάνουν το JSON μία ιδανική γλώσσα ανταλλαγής δεδομένων.

Το JSON είναι κατασκευασμένο σε δύο δομές. Πρώτα, μία συλλογή από ζεύγη ονομάτων-τιμών. Σε πολλές γλώσσες, αυτό γίνεται αντιληπτό ως αντικείμενο, εγγραφή, δομή, λεξικό κ.λ.π. Δεύτερη δομή αποτελεί η ταξινομημένη λίστα από τιμές. Στις περισσότερες γλώσσες, αυτό γίνεται αντιληπτό ως πίνακας, διάνυσμα, λίστα ή ακολουθία.

Αυτές είναι παγκόσμιες δομές δεδομένων. Εικονικά, όλες οι μοντέρνες γλώσσες προγραμματισμού υποστηρίζουν αυτές τις δομές με τον ένα ή τον άλλο τρόπο. Είναι λογικό που μία μορφή δεδομένων η οποία μπορεί να ανταλλαχτεί με προγραμματιστικές γλώσσες, είναι επίσης βασισμένη σε αυτές τις δομές.

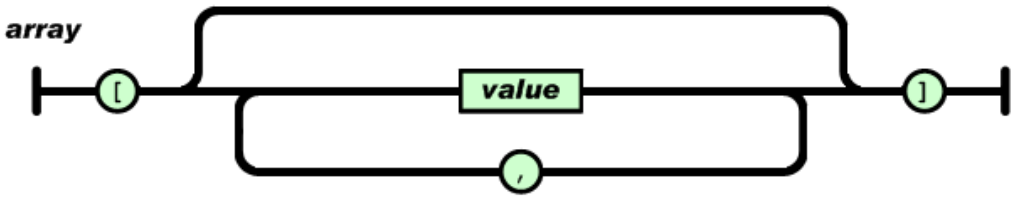
Στο JSON, παίρνουν τις πιο κάτω μορφές :

Ένα αντικείμενο είναι ένα μη ταξινομημένο σύνολο ζευγών ονόματος- τιμής. Ένα αντικείμενο αρχίζει με { και τελειώνει με }. Κάθε όνομα ακολουθείται από : και τα ζεύγη όνομα/τιμή χωρίζονται με κόμμα.



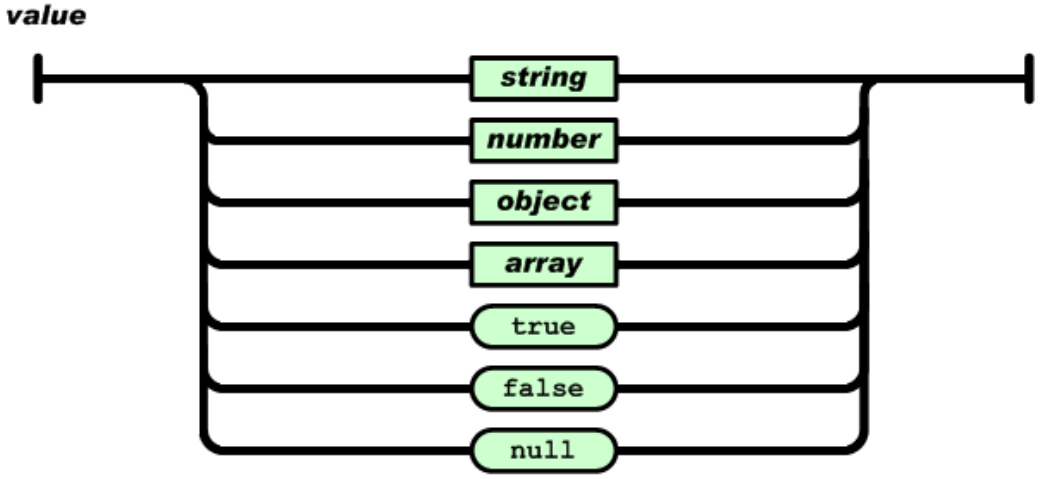
Κεφάλαιο 3 - Εικόνα 3.5
Απεικόνιση JSON αντικειμένου

Ένας πίνακας είναι μία ταξινομημένη συλλογή από τιμές, η οποία ξεκινά με [και τελειώνει με]. Οι τιμές χωρίζονται με κόμμα.



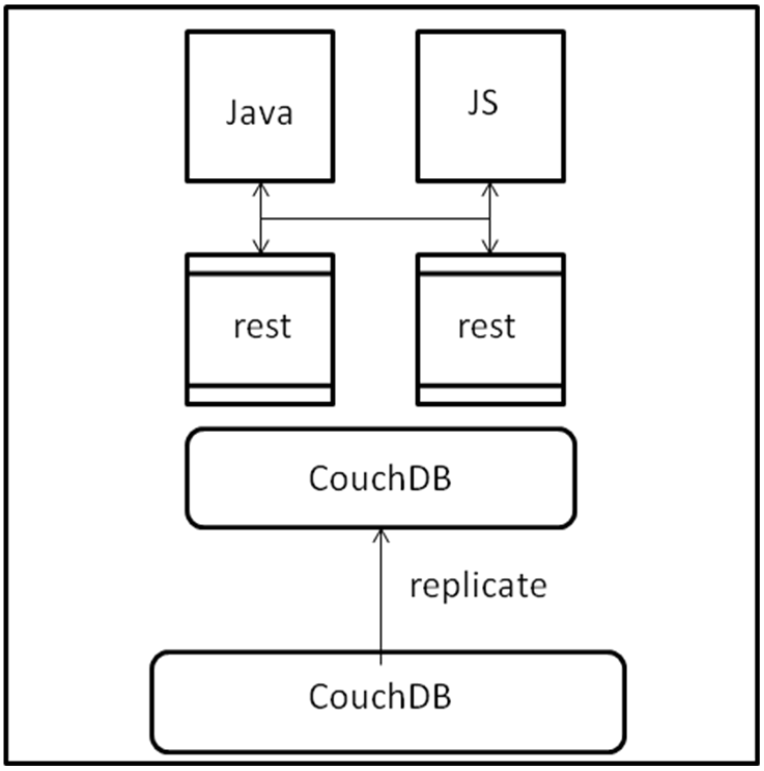
Κεφάλαιο 3 - Εικόνα 3.6
Απεικόνιση JSON πίνακα

Μία τιμή μπορεί να είναι μία συμβολοσειρά σε διπλά εισαγωγικά, ή ένας αριθμός, ή σωστό – λάθος ή Null, αντικείμενο ή πίνακας. Αυτές οι δομές μπορούν να είναι φωλιασμένες.



Κεφάλαιο 3 - Εικόνα 3.7
Απεικόνιση JSON τιμής

3.4.2 Εισαγωγή στη CouchDB



Κεφάλαιο 3 - Εικόνα 3.8
Αρχιτεκτονική CouchDB

Η CouchDB [34] είναι μία βάση δεδομένων η οποία ασπάζεται το διαδίκτυο. Τα δεδομένα σε αυτή τη βάση αποθηκεύονται σε μορφή JSON εγγράφων στα οποία και ο κάθε χρήστης μπορεί να έχει πρόσβαση από τον φυλλομετρητή του, μέσω HTTP. Προκειμένου να μεταμορφωθούν τα έγγραφα του χρήστη αλλά και να συνδυαστούν κα να χρησιμοποιηθούν για αναζήτηση συγκεκριμένων στοιχείων, χρησιμοποιείται το JavaScript. Γενικά, η CouchDB δουλεύει καλά με το μοντέρνο διαδίκτυο και τις κινητές εφαρμογές. Μπορεί ακόμη να εξυπηρετήσει εφαρμογές απευθείας από την CouchDB αλλά και να κατανήμει τα δεδομένα του χρήστη ή τις εφαρμογές του αποδοτικά, χρησιμοποιώντας την αντιγραφή (replication). Η CouchDB υποστηρίζει ρυθμίσεις με αυτόματο εντοπισμό συγκρούσεων όσον αφορά στα έγραφα που αποθηκεύονται σε αυτήν.

Συχνά, η CouchDB κατηγοριοποιείται ως “NoSQL” βάση δεδομένων, ένας όρος ο οποίος και έγινε γνωστός στα τέλη του 2009 και αρχές του 2010. Ωστόσο, αυτός ο όρος ορίζει ξεκάθαρα τον διαχωρισμό της CouchDB από τις παραδοσιακές SQL βάσεις δεδομένων. Συγκεκριμένα, η CouchDB στερείται ένα σχήμα, ή προκαθορισμένες δομές δεδομένων όπως πίνακες. Τα δεδομένα που αποθηκεύονται στη CouchDB, όπως προανέφερα είναι JSON έγγραφα, και η δομή των εγγράφων αυτών μπορεί να αλλάξει δυναμικά για να συμβαδίσει με τις εξελισσόμενες ανάγκες.

Προκειμένου να κατανοήσει κάποιος τι ακριβώς είναι η CouchDB, θα πρέπει πρώτα να καταλάβει τι δεν είναι η CouchDB [35]. Συγκεκριμένα, δεν είναι σχεσιακή βάση δεδομένων αλλά ούτε και μία αντικατάσταση για όλες τις βάσεις δεδομένων. Τέλος, η CouchDB δεν είναι αντικειμενοστρεφής βάση δεδομένων αφού ενώ, αποθηκεύει JSON αντικείμενα, δεν προορίζεται να λειτουργεί ως ένα ομοιογενές στρώμα για μία αντικειμενοστρεφή γλώσσα προγραμματισμού.

Όπως φαίνεται και από το πιο πάνω σχήμα (Εικόνα 3.8), η αρχιτεκτονική της CouchDB στηρίζεται στο RESTFull API, δηλαδή αποτελεί ένα είδος αρχιτεκτονικής η οποία χρησιμοποιείτε για το διαδίκτυο. Μέσω του HTTP πρωτοκόλλου παρέχεται η δυνατότητα POST, PUT, GET και DELETE των εγγράφων που αποθηκεύονται στη βάση αυτή. Μερικά από τα πλεονεκτήματα που προσφέρει η CouchDB είναι το γεγονός ότι μπορεί να αντιγραφεί πολύ εύκολα, είναι αξιόπιστη και είναι κοντά στο χρήστη.

Μερικά από τα βασικά χαρακτηριστικά της CouchDB είναι τα έγγραφα, οι προβολές, η έλλειψη σχήματος και το κατανεμημένο σύστημα . Αρχικά, τα έγγραφα, είναι JSON αντικείμενα τα οποία και αποτελούνται από πεδία. Τα πεδία αυτά μπορεί να είναι συμβολοσειρές, αριθμοί, ημερομηνίες ή ακόμη συσχετιζόμενοι χάρτες.

Μία CouchDB βάση δεδομένων είναι μία συλλογή από έγγραφα, τα οποία και αναγνωρίζονται από ένα μοναδικό αριθμό.

Όσον αφορά στις προβολές, αποτελούν τη μέθοδο συγκέντρωσης και υποβολής στις εγγραφές της βάσης. Δημιουργούνται δυναμικά και δεν επηρεάζουν τα έγγραφα. Ο χρήστης μπορεί να έχει όσες προβολές επιθυμεί στα ίδια δεδομένα.

Πέρα από τις SQL βάσεις δεδομένων οι οποίες είναι σχεδιασμένες για να αποθηκεύουν και να αναφέρονται σε υψηλά δομημένα δεδομένα, η CouchDB, είναι σχεδιασμένη να αποθηκεύει και να αναφέρεται σε μεγάλες ποσότητες ημί-δομημένων, εγγράφων. Επιπλέον, απλοποιεί τη δημιουργία εφαρμογών βασισμένων σε έγγραφα.

Επιπρόσθετα, σε μία SQL βάση δεδομένων, το σχήμα και η αποθήκευση των δεδομένων που είναι ήδη μέσα στη βάση, πρέπει να ενημερώνεται με βάση τις ανάγκες που προκύπτουν. Αντιθέτως, με τη CouchDB δε χρειάζεται σχήμα, επομένως νέα είδη εγγράφων με καινούργια σημασία μπορούν να αποθηκευτούν παράλληλα με την παλιά έκδοση.

Τελευταίο χαρακτηριστικό είναι το κατανεμημένο σύστημα, όπου οποιοσδήποτε αριθμός από CouchDB εξυπηρετητές μπορεί να έχει ανεξάρτητα αντίγραφα της ίδιας βάσης. Η CouchDB έχει ενσωματωμένο μηχανισμό εντοπισμού των συγκρούσεων μεταξύ των εγγράφων αλλά και διαχείρισης τους και η διαδικασία αντιγραφής είναι πολύ γρήγορη, εφόσον αντιγράφονται μόνο έγγραφα τα οποία άλλαξαν από την τελευταία φορά που έγινε κάποια αντιγραφή.
Τα πιο κύρια από αυτά τα χαρακτηριστικά αναλύονται περεταίρω στη συνέχεια.

3.4.3 Documents

Τα έγγραφα (documents) [37] αποτελούν την κεντρική δομή δεδομένων της CouchDB. Είναι αυτόνομες μονάδες δεδομένων, παρέχουν δομή και ομαδοποιούν τα εγγενή δεδομένα. Σε κάθε ένα έγγραφο μπορεί να υπάρχει απροσδιόριστος αριθμός από πεδία και δεν υπάρχουν περιορισμοί όσον αφορά στο μέγεθος των πεδίων. Εάν κάποιος επιθυμεί να έχει πρόσβαση σε 10 ή 20 έγγραφα κάθε φορά, η CouchDB παρέχει αυτή τη δυνατότητα για πρόσβαση σε συλλογή εγγράφων, προσθέτοντας τους συνδέσμους επόμενο και προηγούμενο.
Το κάθε έγγραφο είναι σε JSON μορφή το οποίο και λειτουργεί ως εξής: Άγκιστρα ({}) καλύπτουν τα αντικείμενα και τα αντικείμενα είναι ζεύγη key/value. Τα keys είναι συμβολοσειρές οι οποίες βρίσκονται μεταξύ διπλών εισαγωγικών (“”) ενώ τα values είναι συμβολοσειρά , ακέραιος, αντικείμενο ή ένας πίνακας. Τα key/values διαχωρίζονται από μία άνω κάτω τελεία (:) και πολλαπλά keys και values από κόμμα (,).
Ένα άλλο χαρακτηριστικό των εγγράφων, είναι ότι περιέχουν μετά-πληροφορίες δηλαδή, μοναδικό αριθμό εγγράφου (_id) και αριθμό αναθεώρησης (_rev). Η CouchDB δίνει τη δυνατότητα στο χρήστη να παρέχει αυτός τον μοναδικό αριθμό σε κάθε έγγραφο. Σε αντίθετη περίπτωση, η ίδια η CouchDB, κατά την εισαγωγή του εγγράφου στη βάση, του δίνει ένα μοναδικό αριθμό. Με αυτό τον τρόπο, κάθε φορά που γίνονται αλλαγές σε κάποιο έγγραφο, εντοπίζεται το συγκεκριμένο από το μοναδικό του αριθμό , και οι αλλαγές αποθηκεύονται σε ένα νέο αριθμό αναθεώρησης (_rev). Ουσιαστικά, ο αριθμός αυτός αποτελεί την ιστορία αλλαγών του συγκεκριμένου εγγράφου.

3.4.4 Views

Οι προβολές (views) [38], στην CouchDB είναι χρήσιμες για αρκετούς λόγους, ένας από τους οποίους είναι και το ότι παρέχουν φιλτράρισμα των εγγράφων μέσα στη βάση δεδομένων προκειμένου, να εντοπιστούν αυτά που σχετίζονται με μια συγκεκριμένη διεργασία. Επιπλέον, οι προβολές δίνουν τη δυνατότητα στο χρήστη να εξαγάγει πληροφορίες από τα έγγραφα και να τα παρουσιάσει με συγκεκριμένη σειρά. Μία άλλη χρησιμότητα των

προβολών είναι το ότι μπορεί ο χρήστης να κάνει όλων των ειδών τους υπολογισμούς πάνω στα δεδομένα των εγγράφων, όπως για παράδειγμα μία προβολή να επιστρέψει τον αριθμό των πωλήσεων που είχε μία επιχείρηση τον τελευταίο μήνα. Τέλος, κτίζουν αποδοτικούς δείκτες ώστε να εντοπίζονται τα έγγραφα από οποιαδήποτε τιμή ή δομή αλλά επίσης, χρησιμοποιούνται αυτοί οι δείκτες για αναπαράσταση της σχέσης μεταξύ των εγγράφων.

Οι προβολές στη CouchDB, αποτελούν το κύριο εργαλείο το οποίο χρησιμοποιείται για αναζήτηση και αναφορά στα έγγραφά της και, χωρίζονται σε δύο κατηγορίες, τις μόνιμες προβολές και τις προσωρινές [39]. Οι μόνιμες προβολές, αποθηκεύονται σε ειδικά έγγραφα τα οποία και ονομάζονται έγγραφα σχεδιασμού (design documents) και στα οποία ο χρήστης έχει πρόσβαση μέσω μίας HTTP GET αίτησης στο URI/dbName/docId/viewName, όπου το docId έχει ως πρόθεμα το _design/ έτσι ώστε η CouchDB να αναγνωρίζει το έγγραφο ως έγγραφο σχεδιασμού (design document), και το viewName έχει το πρόθεμα _view/ ώστε να αναγνωρίζεται από τη CouchDB ως προβολή (view). Οι προσωρινές προβολές δεν αποθηκεύονται στη βάση δεδομένων αλλά, εκτελούνται κατά ζήτηση του χρήστη. Προκειμένου να εκτελεστεί μία προσωρινή προβολή, ο χρήστης εκτελεί μία HTTP POST αίτηση στο URI/dbnName/_temp_view, όπου το σώμα της αίτησης περιέχει τον κώδικα της συνάρτησης προβολής. Ένα σημαντικό γνώρισμα των προσωρινών προβολών είναι το γεγονός ότι πρέπει να χρησιμοποιούνται μόνο κατά τη κατασκευή εφόσον είναι πολύ ακριβές όσον αφορά το υπολογιστικό κόστος και γίνονται ολοένα και πιο αργά όσο ο όγκος των δεδομένων στη βάση αυξάνεται. Ωστόσο, και στα δύο είδη προβολών, η προβολή καθορίζεται από μία JavaScript συνάρτηση η οποία αντιστοιχεί τα κλειδιά (keys) της προβολής στις τιμές (values).

Εξορισμού, οι προβολές δεν δημιουργούνται αλλά ούτε ενημερώνονται όταν ένα έγγραφο αποθηκευτεί, αλλά όταν κάποιος αποκτήσει πρόσβαση πάνω τους. Επιπρόσθετα, όλες οι προβολές σε ένα έγγραφο σχεδίου, ενημερώνονται όταν μία από τις προβολές στο συγκεκριμένο έγγραφο σχεδίου αναζητηθεί.

Η CouchDB παίρνει τον πηγαίο κώδικα του κάθε view, όπου η συνάρτηση είναι γραμμένη σε JavaScript και αποτελεί μία map function με παράμετρο doc, και τον τρέχει σε κάθε έγγραφο που είναι αποθηκευμένο στη βάση. Ακολούθως, το αποτέλεσμα της προβολής αποθηκεύεται σε ένα B δέντρο, το οποίο με τη σειρά του αποθηκεύεται σε δικό του αρχείο για υψηλότερη απόδοση. Το B δέντρο παρέχει γρήγορες αναζητήσεις των γραμμών με βάση το κλειδί τους.

Εκτός από τις map συναρτήσεις, υπάρχουν εναλλακτικά και οι reduce. Σκοπός μιας τέτοιας συνάρτησης είναι να μειώσει τις τιμές εισόδου. Όταν μία προβολή έχει reduce συνάρτηση, αυτή χρησιμοποιείται για την παραγωγή συνολικών αποτελεσμάτων για τη συγκεκριμένη προβολή. Στη συνάρτηση αυτή δίνεται ένα σύνολο από ενδιάμεσες τιμές και αυτή τις συνδυάζει σε μία μόνο τιμή. Οι reduce συναρτήσεις θα πρέπει να δέχονται ως είσοδο,

αποτελέσματα που δίνονται από την αντίστοιχη map συνάρτηση, καθώς επίσης και αποτελέσματα που επιστρέφονται από την ίδια τη reduce συνάρτηση. Αυτή η περίπτωση ονομάζεται rereduce. Η reduce συνάρτηση, παίρνει τρείς παραμέτρους, key, values και rereduce.

3.4.5 Design Documents

Τα έγγραφα σχεδιασμού (design documents) [40] είναι ένα ειδικό είδος εγγράφου της CouchDB και το οποίο περιέχει κώδικα εφαρμογής. Λόγω του ότι τρέχει μέσα στη βάση, το API της εφαρμογής είναι υψηλά δομημένο.

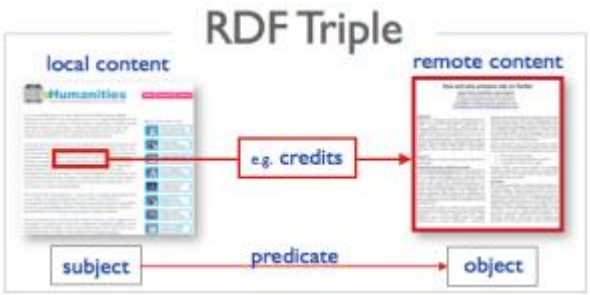
Η couchDB, έχει ένα πακέτο λογισμικού (query server) το οποίο και εκτελεί τις συναρτήσεις των εγγράφων σχεδιασμού, και είναι γραμμένο σε JavaScript. Ένα έγγραφο σχεδιασμού είναι ένα έγγραφο της CouchDB με ένα αναγνωριστικό το οποίο και ξεκινά με _design/. Τα έγγραφα σχεδιασμού είναι όπως όλα τα υπόλοιπα έγγραφα της CouchDB, δηλαδή, αναπαράγονται μαζί με τα άλλα και εντοπίζουν συγκρούσεις σε αυτά με την παράμετρο rev. Είναι κανονικά JSON έγγραφα και υποδηλώνεται αυτό από το γεγονός ότι το αναγνωριστικό του εγγράφου έχει το πρόθεμα _design/.

Η couchDB χρησιμοποιεί Map/Reduce μοντέλο για αναζήτηση. Τα ερωτήματα αυτά αποθηκεύονται στο πεδίο προβολές (views). Με αυτό τον τρόπο το Futon, δηλαδή η κονσόλα που χρησιμοποιείται για γραφικό περιβάλλον, προβάλει και επιτρέπει την τροποποίηση των Map/Reduce ερωτήσεων. Εάν οποιαδήποτε συνημμένα αρχεία, επικυρώσεις ή άλλα πεδία ενός εγγράφου σχεδιασμού αλλάξουν, η προβολή (view) δε θα παραχθεί ξανά. Αντιθέτως, εάν αλλάξει μία map ή μία reduce συνάρτηση, τότε ο δείκτης της προβολής (view index) θα διαγραφεί και ένας νέος θα δημιουργηθεί για τις νέες συναρτήσεις της προβολής.

Το γεγονός ότι η CouchDB υποστηρίζει τη δυνατότητα αντιγραφής, επιτρέπει σε μία μόνο βάση δεδομένων να φιλοξενεί πολλαπλές εφαρμογές.

3.5 Διασυνδεδεμένες δημοσιοποιημένες πληροφορίες (LOD)

Τα RDF (Resource Description Framework) [41], είναι μία γενική μέθοδος για αποσύνθεσης της γνώσης σε μικρά κομμάτια, με κάποιους κανόνες όσον αφορά στη σημασιολογία, ή έννοια αυτών των κομματιών. Στόχος είναι να υπάρχει μία



Κεφάλαιο 3 - Εικόνα 3.9
Παράδειγμα RDF τριάδας

μέθοδος που είναι τόσο απλή που να μπορεί να εκφράσει οποιοδήποτε γεγονός, αλλά ταυτόχρονα να είναι τόσο δομημένη ώστε οι εfαρμογές των υπολογιστών να μπορούν να κάνουν χρήσιμα πράγματα με τη γνώση που εκφράζεται μέσω των RDF.

Το RDF, μπορεί να συγκριθεί με το XML, το οποίο σχεδιάστηκε επίσης για να είναι απλό και εφαρμόσιμο σε οποιοδήποτε τύπο δεδομένων. Το XML είναι επίσης κάτι παραπάνω από ένα είδος αρχείου. Είναι ένα θεμέλιο για αντιμετώπιση ιεραρχικών, αυτό-περιεχόμενων εγγράφων, είτε αυτά αποθηκεύονται στο δίσκο είτε στη μνήμη.

Αυτό που κάνει το RDF να διαφέρει από το XML είναι το γεγονός ότι το RDF σχεδιάστηκε για να αντιπροσωπεύει τη γνώση σε ένα κατανεμημένο κόσμο. Το ότι σχεδιάστηκε για τη γνώση και όχι για τα δεδομένα, αυτό σημαίνει ότι ασχολείται κυρίως με τις έννοιες, δηλαδή, τα πάντα στο RDF σημαίνουν κάτι. Μπορεί να είναι μια αναφορά σε κάτι που υπάρχει στον κόσμο, ή ακόμα μία αφαιρετική ιδέα. Η έννοια της τριάδας είναι ότι με το να συνδυάζεις κάποια δεδομένα μαζί, κάποιος μπορεί να καταλήξει σε ένα είδος γνώσης.

Ένα άλλο σημαντικό χαρακτηριστικό του RDF, είναι το ότι είναι κατάλληλο για κατανεμημένες πληροφορίες. Αυτό σημαίνει ότι εφαρμογές RDF μπορούν να συναρμολογήσουν RDF αρχεία που δόθηκαν από διαφορετικούς χρήστες στο διαδίκτυο. Αυτό το επιτυγχάνει με δύο διαφορετικούς τρόπους, πρώτα ενώνοντας τα έγγραφα μαζί με βάση το κοινό λεξιλόγιο που χρησιμοποιούν, και δεύτερο, επιτρέποντας σε οποιοδήποτε έγγραφο να χρησιμοποιήσει οποιοδήποτε λεξιλόγιο. Αυτό, επιτρέπει απεριόριστη ευελιξία στην έκφραση γεγονότων για ένα ευρύ πεδίο αντικειμένων, σχεδιάζοντας πάνω σε πληροφορίες που λήφθηκαν από ένα επίσης ευρύ πεδίο πηγών.

Το RDF παρέχει μία γενική, ευέλικτη μέθοδο για αποσύνθεση οποιασδήποτε γνώσης σε μικρά κομμάτια, τις ονομαζόμενες τριάδες, με κάποιους κανόνες σχετικά με τη σημασιολογία αυτών των κομματιών.

Θεμέλιο του είναι να κομματιάζει τη γνώση σε επισημασμένο κατευθυνόμενο γράφο. Κάθε ακμή του γράφου αντιπροσωπεύει ένα γεγονός, ή τη σχέση μεταξύ δύο αντικειμένων. Ένας τέτοιος γράφος [42], αποτελείται από τριάδες RDF όπου κάθε μία από αυτές αποτελείται από ένα υποκείμενο, ένα κατηγορούμενο και ένα αντικείμενο. Ένα RDF/JSON έγγραφο παρουσιάζει ένα σύνολο από RDF τριάδες σαν σειρά από φωλιασμένες δομές δεδομένων. Ένα RDF/JSON έγγραφο αποτελείται από ένα JSON αντικείμενο ονομαζόμενο αντικείμενο ρίζας. Κάθε μοναδικό θέμα σε ένα σύνολο από τριάδες αντιπροσωπεύει σαν ένα κλειδί στο αντικείμενο ρίζας, και κανένα κλειδί δε μπορεί να εμφανιστεί πάνω από μία φορά στο κύριο αντικείμενο.

Η τιμή κάθε αντικειμένου ρίζας είναι JSON αντικείμενο του οποίου τα κλειδιά είναι τα URI των κατηγορημάτων που προκύπτουν στις τριάδες με το δοθέν υποκείμενο. Αυτά τα κλειδιά

είναι γνωστά ως κλειδιά κατηγορήματος. Και σε αυτή την περίπτωση, κανένα κλειδί κατηγορήματος δε μπορεί να εμφανιστεί περισσότερο από μία φορές σε κάθε αντικείμενο.

Γενικά, μία τριάδα (υποκείμενο, κατηγορούμενο, αντικείμενο), προβάλλεται με αυτή τη μορφή: `{ "S" : { "P" : [ O ] } }`. Το αντικείμενο Ο είναι JSON αντικείμενο με τα εξής κλειδιά:

Τύπος (Type): Ένα από τα uri

Τιμή (Value): Το uri του αντικειμένου ή άδεια ετικέτα αναλόγως του αν το αντικείμενο είναι uri, γράμμα ή Β-κόμβος.

Γλώσσα (Lang): Η γλώσσα μιας πραγματικής τιμής

Τύπος Δεδομένων (DataType): Ο τύπος δεδομένων URI μίας πραγματικής τιμής.

Κεφάλαιο 4

Υλοποίηση Συστήματος

4.1 Εισαγωγή

Οι κύριοι στόχοι της Android εφαρμογής CLODA, είναι να αναπτύξει αρχικά ένα πλαίσιο το οποίο συνδυάζει διάφορα ρεύματα δεδομένων (Online, sensor readings,user input), ώστε να δημιουργήσει δομημένες πληροφορίες οι οποίες ακολουθούν τις αρχές των διασυνδεδεμένων δημοσιοποιημένων πληροφοριών (LOD) και δεύτερο, να αναπτύξει μία εύκολη ως προς τη χρήση εφαρμογή, η οποία θα βοηθήσει τους χρήστες να καταγράψουν αντικείμενα σε εσωτερικούς χώρους περιλαμβάνοντας εσωτερική τοποθέτηση και πληροφορίες

περιβάλλοντος καθώς επίσης και online μετά-πληροφορίες διαθέσιμες για αντικείμενα με γραμμικό κώδικα (barcode).

Το πρωτότυπο της εφαρμογής παρέχει τη δυνατότητα στους χρήστες, να αναζητήσου και να γεωτοποθετήσουν αντικείμενα σε εσωτερικά και εξωτερικά περιβάλλοντα πάνω από ένα παραδοσιακό γεωγραφικό χαρτογραφικό σύστημα (π.χ. Google Maps). Η εφαρμογή CLODA, επιτρέπει στους χρήστες να μεταφορτώσουν και να προσαρμόσουν χάρτες ορόφων κτιρίων ως επίπεδα πάνω από τα Google Maps ενώ, παράλληλα θα χρησιμοποιούν το AirPlace, τη βιβλιοθήκη για εσωτερική τοποθέτηση, ώστε να μπορούν να πλοηγηθούν στους διαδρόμους και να συνεισφέρουν αντικείμενα και αποθέματα αντικειμένων στις δημόσιες διασυνδεδεμένες δημοσιοποιημένες πληροφορίες (LOD). Επιπρόσθετα, παρέχει τη δυνατότητα αυτόματης ανάκτησης βασικών πληροφοριών σχετικά με το αντικείμενο, εξάγοντας το γραμμικό κώδικα (barcode) του αντικειμένου χρησιμοποιώντας μία εξωτερική βιβλιοθήκη, τη ZXing Library, καταναλώνοντας τις μετά-πληροφορίες των αντικειμένων χρησιμοποιώντας το Google Shopping API.

4.2 Περίληψη Τεχνολογιών

4.2.1 Android Λειτουργικό Σύστημα

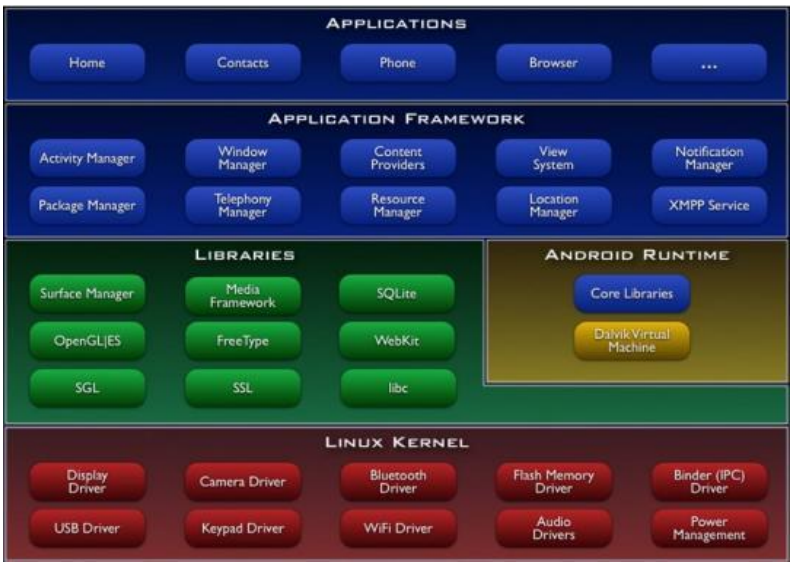
Το Android [26] είναι ένα λειτουργικό σύστημα για κινητές συσκευές όπως για παράδειγμα κινητά τηλέφωνα, tablets και netbooks. Το Android δημιουργήθηκε από την Google και είναι βασισμένο στον Linux πυρήνα. Αρχικά δημιουργήθηκε από την Android Inc. και πιο μετά από την Open Handset Alliance. Σύμφωνα με την NPD Group, πωλήσεις ανά ομάδα για έξυπνες κινητές συσκευές με Android λειτουργικό σύστημα κατατάχθηκαν δεύτερες το πρώτο τρίμηνο του 2010.

Το Android [27], είναι η δημοφιλέστερη πλατφόρμα κινητών συσκευών. Με το Android μπορείς να χρησιμοποιήσεις όλες τις Google εφαρμογές και επιπλέον να κατεβάσεις ακόμη περισσότερες από το Google Play.

Το Android αρχικά σχεδιάστηκε για οθόνες αφής σε κινητές συσκευές. Είναι ανοικτό στο κοινό (open source) και το γεγονός αυτό επιτρέπει στο λογισμικό να μπορεί να μορφοποιηθεί και να κατανεμηθεί από κατασκευαστές συσκευών. Επιπρόσθετα, το Android έχει μία μεγάλη κοινότητα από κατασκευαστές οι οποίοι γράφουν εφαρμογές που προεκτείνουν τη λειτουργικότητα των συσκευών, γραμμένες κυρίως στη γλώσσα προγραμματισμού Java. Αυτοί είναι και μερικοί λόγοι που οδήγησαν το Android να γίνει η πιο ευρέως γνωστή πλατφόρμα για έξυπνες κινητές συσκευές. Αποτέλεσμα αυτού, ήταν να σχεδιαστούν εφαρμογές για τηλεοράσεις, κονσόλες παιχνιδιών κ.λ.π. Επιπλέον, το γεγονός ότι ο κώδικας είναι ανοικτός στο κοινό, έχει ενθαρρύνει ένα μεγάλο αριθμό από κατασκευαστές να τον

χρησιμοποιήσουν ως θεμέλιο για μεγαλύτερα έργα αλλά και να εφαρμόσουν το Android σε συσκευές που αρχικά έτρεχαν σε διαφορετικό λειτουργικό σύστημα.

Το Android, έχει μία μεγάλη κοινότητα από κατασκευαστές που γράφουν εφαρμογές οι οποίες επεκτείνουν τη λειτουργικότητα των συσκευών. Μέχρι στιγμής, υπάρχουν 70.000 εφαρμογές για Android, κάτι που το κάνει το δεύτερο δημοφιλέστερο στόχο για ανάπτυξη κινητών συσκευών. Οι κατασκευαστές γράφουν κώδικα στη γλώσσα προγραμματισμού Java, ελέγχοντας τη συσκευή μέσω βιβλιοθήκες Java που δημιουργήθηκαν από την Google. Η Google κυκλοφόρησε τον περισσότερο από τον Android κώδικα κάτω από την Apache License, μία άδεια δωρεάν λογισμικού και ανοικτού κώδικα.



Κεφάλαιο 4 - Εικόνα 4.1
Android Stack

Στην εικόνα 4.1 φαίνεται η στοίβα λογισμικού του Android λειτουργικού συστήματος η οποία και αποτελείται από java εφαρμογές που τρέχουν σε ένα αντικειμενοστρεφής εφαρμογής πλαίσιο βασισμένο στη Java, πάνω από Java βιβλιοθήκες πυρήνα σε μία Dalvik εικονική μηχανή που περιέχει JIT μεταγλώττιση. Οι βιβλιοθήκες που είναι γραμμένες στη C περιέχουν το διαχειριστή επιφάνειας, την SQLite σύστημα διαχείρισης σχεσιακής βάσης, γραφικά OpenGL ES 2.0 3D, WebKit διάταξη κινητήρα, SGL γραφικά, SSL. Το Android λειτουργικό σύστημα αποτελείται από 12 εκατομμύρια γραμμές κώδικα συμπεριλαμβανομένων 3 εκατομμυρίων γραμμών από XML, 2.8 εκατομμύρια γραμμές Java και 1.75 εκατομμύρια γραμμές σε C++.

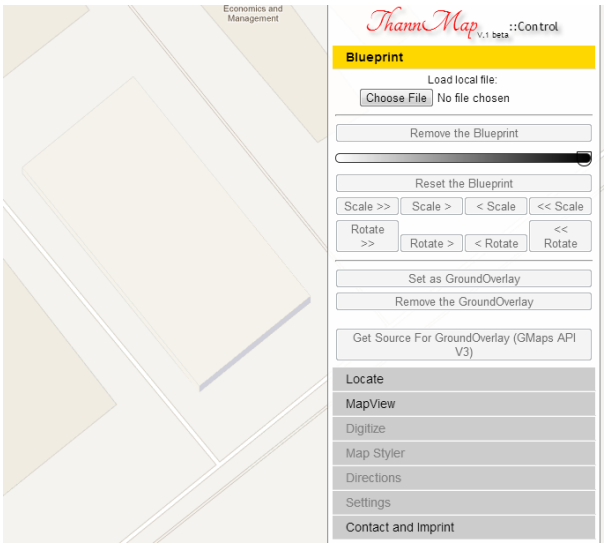
Μερικά άλλα πλεονεκτήματα του Android λειτουργικού συστήματος σε σχέση με άλλα είναι, οι πολυδιεργασίες (multitasking) όπου οι κινητές συσκευές Android μπορούν να τρέχουν πολλές εφαρμογές όπως για παράδειγμα, ενώ ακους μουσική να διαβάζει ένα άρθρο από μία ιστοσελίδα. Ένα άλλο πλεονέκτημα είναι και η ευκολία πρόσβασης σε χιλιάδες δωρεάν εφαρμογές μέσω του Google Play.

4.2.2 Google Maps - JavaScript v3 API

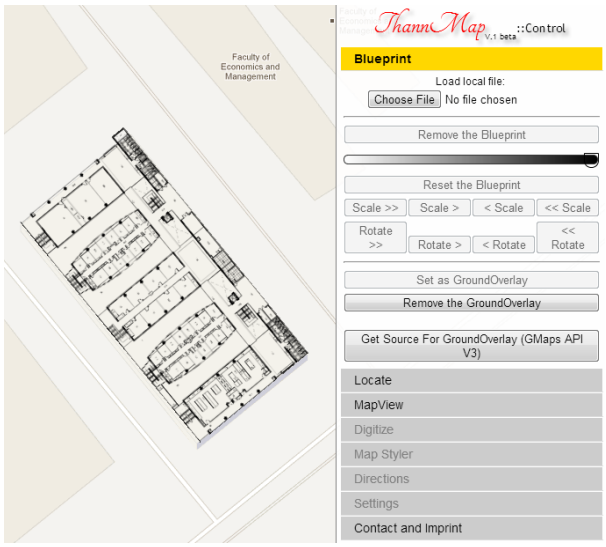
Η εφαρμογή χρησιμοποίει τα Google Maps για να εμφανίζει το κτίριο στο οποίο θα χρησιμοποιείται η εφαρμογή, και στη συγκεκριμένη περίπτωση το κτίριο της Πληροφορικής

του Πανεπιστημίου Κύπρου. Υπάρχουν δύο εναλλακτικοί τρόποι με τους οποίους θα μπορούσα να ενσωματώσω τα Google Maps στην Android εφαρμογή και οι οποίοι είναι, είτε με τη χρήση MapView είτε με τη χρήση WebView και Javascript v3 API. Όσον αφορά στην επιλογή μεταξύ MapView και Javascript v3 για χρήση των Google Maps, αρχικά, χρησιμοποίησα τη MapView κλάση του Android, όπου σε συνδυασμό με το πακέτο των Google Maps μπορούσα να προβάλω το χάρτη με το κτίριο της Πληροφορικής του Πανεπιστημίου Κύπρου. Τελικά, λόγω του ότι το zoom level στο χάρτη δεν ήταν αρκετό ώστε να μπορεί ο χρήστης να επιλέξει με ευκολία οποιοδήποτε σημείο σε οποιοδήποτε χώρο του κτιρίου το οποίο και θα αντιστοιχεί στη θέση του αντικειμένου που θέλει να σαρώσει και να αποθηκεύσει στη βάση , επέλεξα το κομμάτι της εφαρμογής που έχει να κάνει με το χάρτη να το υλοποιήσω με Javascript v3. Τόσο το MapView όσο και το Javascript v3 είναι tile based maps και σε αυτά τα tiles είναι προκαθορισμένο το zoom level και στη συγκεκριμένη περίπτωση είναι διαφορετικό στο MapView από το Javascript εφόσον πρόσφατα παρουσιάστηκε το Javascript API v3 και έχει περισσότερο zoom level σε αντίθεση με το MapView στο οποίο δεν υπήρξε κάποια αλλαγή τα τελευταία χρόνια οπόταν το zoom level έμεινε στο ίδιο μέγιστο που ήταν και πριν.

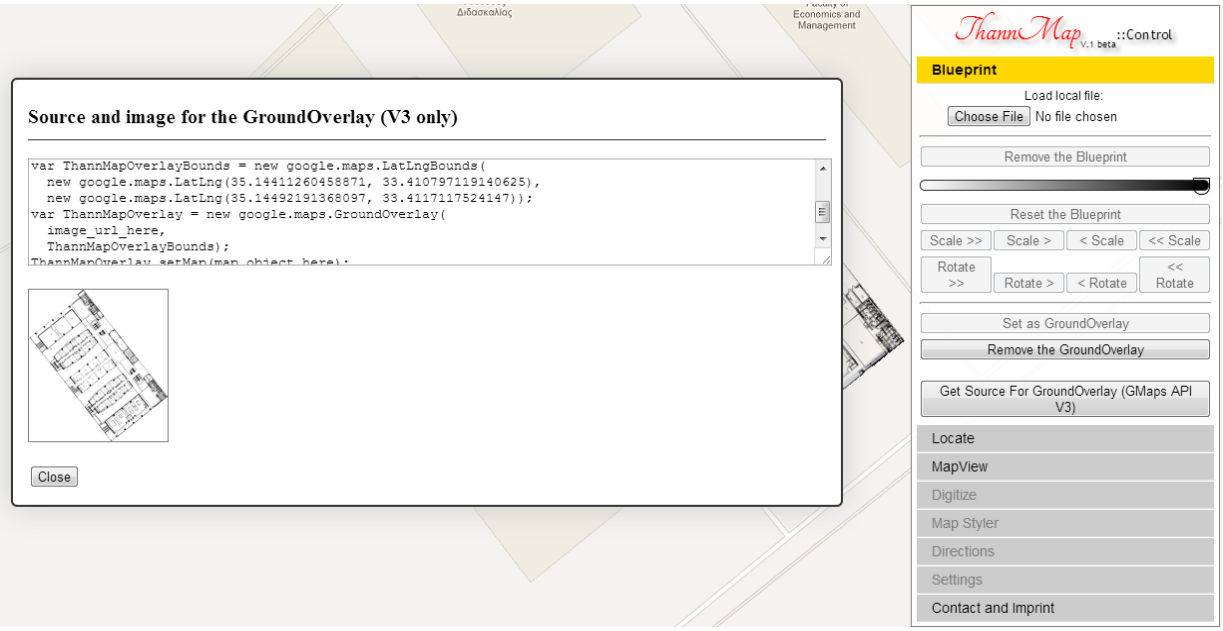
Το Google Maps JavaScript API Version 3, [28] είναι τώρα το επίσημο JavaScript API. Όπως γνωρίζουμε, το Google Maps JavaScript API επιτρέπει στους χρήστες του να ενσωματώσουν Google Maps στη δική τους προσωπική ιστοσελίδα. Η έκδοση 3, την οποία και χρησιμοποίησα για τις απαιτήσεις της διπλωματικής μου, είναι ειδικά σχεδιασμένη ώστε να είναι πιο γρήγορη και περισσότερο εφαρμόσιμη σε κινητές συσκευές, όσο και σε παραδοσιακές εφαρμογές σε φυλλομετρητές. Το API παρέχει ένα αριθμό από υπηρεσίες για διαχείριση των χαρτών και για πρόσθεση περιεχομένου στο χάρτη μέσω ενός αριθμού υπηρεσιών, επιτρέποντας στο χρήστη τη δημιουργία ισχυρών εφαρμογών για χάρτες στην προσωπική τους ιστοσελίδα. Το JavaScript Maps V3 είναι μία δωρεάν υπηρεσία, διαθέσιμη για οποιαδήποτε ιστοσελίδα η οποία είναι δωρεάν προς όλους τους καταναλωτές.



Κεφάλαιο 4 – Εικόνα 4.2
Όψη κτιρίου μέσω της ιστοσελίδας ThannMap



Κεφάλαιο 4 - Εικόνα 4.3
Όψη κτιρίου μετά την εφαρμογή της κάτοψης



Κεφάλαιο 4 - Εικόνα 4.4
Λήψη συντεταγμένων τοποθέτησης της κάτοψης

Εφόσον η εφαρμογή χρησιμοποιεί το κτίριο της Πληροφορικής, ήταν αναγκαίο να τοποθετηθούν οι κατόψεις των τεσσάρων ορόφων ώστε να μπορεί ο χρήστης να βλέπει τις αίθουσες και του διαδρόμους στους οποίους θα κινείται και θα σαρώνει αντικείμενα. Αρχικά, αφού πήρα τις κατόψεις και τια αποθήκευσα ως εικόνες, προσπάθησα να τις εφαρμόσω πάνω στο κτίριο στα Google Maps βρίσκοντας τα δύο διαγώνια σημεία. Επειδή όμως αυτό, όπως διαπίστωσα στην πορεία ήταν αρκετά δύσκολο, χρονοβόρο και δεν έδινε τα απαιτούμενα αποτελέσματα, δηλαδή οι κατόψεις δεν εφαρμόζονταν τελείως σωστά πάνω στο κτίριο, βρήκα εναλλακτική λύση. Συγκεκριμένα, χρησιμοποίησα μία ιστοσελίδα [29], η οποία σου δίνει τη δυνατότητα να εφαρμόσεις την κάτοψη οποιουδήποτε κτιρίου σε οποιοδήποτε μέρος στον κόσμο. Αφού ενσωματωθεί η κάτοψη πάνω στο κτίριο μπορείς να λάβεις τις συντεταγμένες, γεωγραφικό μήκος και πλάτος, που είναι και τα σημεία στα οποία τοποθετήθηκε η κάτοψη πάνω στο κτίριο. Τις συντεταγμένες αυτές τις χρησιμοποίησα ακολούθως στον κώδικά μου ώστε να κάθε φορά που αλλάζει ο όροφος να εφαρμόζεται στη σωστή θέση η κάτοψη του συγκεκριμένου ορόφου. Στις πιο πάνω εικόνες φαίνονται οι τρεις φάσεις για ενσωμάτωση των κατόψεων πάνω στο κτίριο.

4.2.3 Meta Web 2.0 API (Google Shopping API)

Ο όρος Web 2.0 [31] περιγράφει μία δεύτερη γενιά του Παγκόσμιου Ιστού η οποία επικεντρώνεται στην ικανότητα των ανθρώπων να συνεργάζονται και να μοιράζονται online πληροφορίες. Το Web 2.0 βασικά αναφέρεται στη μετάβαση από τις στατικές HTML σελίδες σε ένα πιο δυναμικό ιστό ο οποίος και είναι πιο οργανωμένος και είναι βασισμένος στο να προσφέρει εφαρμογές ιστού στους χρήστες. Επιπλέον βελτιωμένη λειτουργικότητα του Web 2.0 περιλαμβάνει ανοικτές πληροφορίες. Με το πέρασμα του χρόνου, το Web 2.0 έχει

χρησιμοποιηθεί περισσότερο ως όρος του μάρκετινγκ παρά ως προς της επιστήμης της πληροφορικής. Διάφορα μπλοκς, wikis, και υπηρεσίες ιστού, όλα είναι ορατά ως Web 2.0. Προηγουμένως, το Web 2.0 χρησιμοποιείτο ως συνώνυμο για το Σημασιολογικό Ιστό όμως, ενώ μοιάζουν αυτοί οι δύο όροι, δεν έχουν ακριβώς την ίδια σημασία.

Ένα μέρος της εφαρμογής CLODA αποτελεί και η αποθήκευση αντικειμένων στις διάφορες αίθουσες του κτιρίου. Τα αντικείμενα αυτά μπορεί να είναι οτιδήποτε από βιβλία μέχρι και οθόνες, πληκτρολόγια κ.λ.π. Εφόσον σαρωθεί το κάθε αντικείμενο, μέσω του Google Shopping API γίνεται αναζήτησή του με βάση το γραμμικό κώδικά του (barcode) και αναλόγως εάν αυτό το αντικείμενο είναι καταγεγραμμένο, επιστρέφονται ή όχι στοιχεία που το αφορούν όπως για παράδειγμα η τιμή, η μάρκα κ.λ.π.

Όσον αφορά στο Google Shopping API [32], η αναζήτηση σε αυτό διευκολύνει τους κατασκευαστές και πελάτες να κτίσουν πρωτοποριακές εφαρμογές χρησιμοποιώντας τα στοιχεία του προϊόντος. Το API απλοποιεί την πρόσβαση στα στοιχεία του αντικειμένου και έχει βελτιστοποιηθεί, για να δίνει όσο το δυνατό πιο σχετικά αποτελέσματα για κάθε προϊόν.

4.2.4 Zxing library - Barcode Scanner

Τα αντικείμενα που αποθηκεύονται στη βάση δεδομένων που είναι συνδεδεμένη με την εφαρμογή CLODA είναι όλα αντικείμενα τα οποία περιέχουν γραμμικό κώδικα (barcode). Προκειμένου να σαρωθούν τα αντικείμενα αυτά, έχω ενσωματώσει στη δική μου εφαρμογή μία άλλη έτοιμη εφαρμογή, το ZXing barcode scanner, το οποίο και επιτρέπει τη σάρωση γραμμικού κώδικα (barcode).

Το ZXing [30] σημαίνει zebra crossing και είναι μία ανοικτού κώδικα , πολυμορφική μίας και δύο διαστάσεων βιβλιοθήκη επεξεργασίας εικόνας από γραμμικό κώδικα (barcode). Επικεντρώνεται στη χρήση της ενσωματωμένης, στο κινητό τηλέφωνο, κάμερα έτσι ώστε να σαρώσει και να αποκωδικοποιήσει γραμμικούς κώδικες (barcodes) στη συσκευή, χωρίς οποιαδήποτε επικοινωνία με κάποιο εξυπηρετητή (server). Παρόλα αυτά, η συγκεκριμένη εφαρμογή μπορεί να χρησιμοποιηθεί και για κωδικοποίηση και αποκωδικοποίηση γραμμικού κώδικα σε desktops και εξυπηρετητές.

4.3 Περιβάλλον Διαπροσωπείας

Σε αυτό τα σημείο, θα παρουσιάσω πιο αναλυτικά το τριών επιπέδων πρωτότυπο της εφαρμογής CLODA δηλαδή, το πρωτότυπο της Android εφαρμογής, τους υψηλά διαθέσιμους εξυπηρετητές διαδικτύου και τις κατανεμημένες βάσεις δεδομένων.

Η εφαρμογή για κινητές συσκευές, κατασκευάστηκε πάνω από το λειτουργικό σύστημα Android και επιτρέπει στον χρήστη να γεωτοποθετήσει νέα αντικείμενα, να μορφοποιήσει, να αναζητήσει ή να διαγράψει ήδη υπάρχοντα αποθέματα ενώ παράλληλα, παρέχει τη δυνατότητα στο χρήστη να τοποθετεί τον εαυτό του στους εσωτερικούς χώρους. Όλες οι πληροφορίες παρουσιάζονται ως επιπρόσθετα επίπεδα πάνω από τα πραγματικά Google Maps.

Όπως ανέφερα και σε προηγούμενα σημεία, η θέση του χρήστη παρέχεται από τη βιβλιοθήκη εσωτερικής τοποθέτησης AirPlace. Πριν την πλοήγηση σε ένα εσωτερικό χώρο, η εφαρμογή φορτώνει το σχέδιο του ορόφου, το οποίο και τοποθετείται σαν επίπεδο πάνω από το χάρτη χρησιμοποιώντας to Google Maps API και ένα radiomap που ανταποκρίνεται στο κοντινότερο, ως προς το χρήστη, κτίριο. Εάν δεν είναι διαθέσιμο το radiomap, ζητείται από το χρήστη να το εισάγει στην εφαρμογή εφόσον είναι απαραίτητο για να μπορεί να πραγματοποιηθεί η τοποθέτηση του στο κτίριο.

Τα τελευταία δύο επίπεδα της αρχιτεκτονικής αποτελούν το «πίσω μερος» (back-end) της CLODA αρχιτεκτονικής. Δημιουργείται ένας αριθμός από εικονικές μηχανές οι οποίες τρέχουν CentOS, έτσι ώστε να παρέχουν ένα σύνδεσμο μεταξύ της Android εφαρμογής και της CouchDB βάσης δεδομένων. Ο λόγος πίσω από την εισαγωγή ενός ενδιάμεσου επιπέδου μεταξύ της Android εφαρμογής και των βάσεων δεδομένων ήταν, η απαίτηση ασφάλειας η οποία προέκυψε κατά τη φάση σχεδίασης όπου απαιτούσε από όλους τους χρήστες να εξακριβώσουν την ταυτότητά τους για πρόσβαση στη βάση δεδομένων. Προκειμένου να μην αποκαλυφθούν οποιεσδήποτε ευαίσθητες πληροφορίες (π.χ. διαπιστευτήρια βάσης δεδομένων) μέσω ασύρματου δικτύου, έχει προστεθεί ένα ενδιάμεσο API υπεύθυνο για τη μεσολάβηση των χρηστών της βάσης. Η επικοινωνία μεταξύ του ενδιάμεσου επιπέδου και της Android εφαρμογής επιτυγχάνεται με ένα Restful τρόπο (δηλαδή, εκδίδονται αιτήσεις HTTP από την εφαρμογή και επιστρέφονται JSON αντικείμενα από το ενδιάμεσο επίπεδο).

Τέλος, το τρίτο επίπεδο της αρχιτεκτονικής αποτελείται από πολλαπλά αντίγραφα της couchDB βάσης δεδομένων. Τα αντίγραφα της βάσης δεδομένων είναι υπεύθυνα να αποθηκεύουν τις πληροφορίες που αφορούν στο κάθε αντικείμενο που έχει γεωτοποθετηθεί από τον χρήστη αλλά επίσης, και να εξυπηρετεί οποιεσδήποτε αιτήσεις καταφθάνουν για αντικείμενα που έχουν γεωτοποθετηθεί προηγουμένως. . Πιο κάτω ακολουθούν εικόνες οι οποίες αναπαριστούν την πλήρη λειτουργία της εφαρμογής. Όπως φαίνεται και από τις εικόνες, στην εφαρμογή χρησιμοποιήθηκε το ActionBarSherlock. Συγκεκριμένα, το actionBarSherlock είναι μία προέκταση της βιβλιοθήκης στήριξης σχεδιασμένο να διευκολύνει τη χρήση του σχεδιαστικού προτύπου του action bar σε όλες τις εκδόσεις του Android με ένα μόνο API.

Η βιβλιοθήκη, αυτόματα, θα χρησιμοποιήσει το μητρικό action bar όταν είναι αναγκαίο ή, αυτόματα θα ολοκληρώσει μία προσαρμοσμένη υλοποίηση γύρω από τα σχεδιαγράμματα του χρήστη (layouts). Αυτό, επιτρέπει στο χρήστη να αναπτύξει εύκολα μία εφαρμογή με ένα action bar το οποίο είναι κατάλληλο για κάθε έκδοση του Android από 2.X και πάνω.

Αλληλεπίδραση με το action bar παρέχεται μέσω ενός μόνο API καλώντας την getSupportActionBar(). Οι μεθόδοι που παρέχονται από αυτή τη διεπαφή, αντικατοπτρίζουν ακριβώς εκείνες του μητρικού action bar.

Η ενεργοποίηση υποστήριξης είναι τόσο απλή όσο το να κληρονομούν οι δραστηριότητες (activities) από μία από τις βασικές δραστηριότητες του Sherlock αλλά και να δηλωθεί ένα θέμα στο αρχείο δήλωσης (manifest file).

Κεφάλαιο 4 - Εικόνα 4.5
Διαθέσιμα Επίπεδα Μεγέθυνσης

Στην πιο πάνω εικόνα, παρουσιάζονται μερικά από τα επίπεδα μεγέθυνσης που είναι διαθέσιμα. Στο πρώτο σχήμα φαίνεται το κτίριο της Πληροφορικής σε χαμηλό επίπεδο μεγέθυνσης. Στο δεύτερο σχήμα φαίνεται το προκαθορισμένο επίπεδο μεγέθυνσης, δηλαδή όταν ο χρήστης ξεκινήσει την εφαρμογή το κτίριο παρουσιάζεται στο συγκεκριμένο επίπεδο μεγέθυνσης. Το τελευταίο σχήμα απεικονίζει το μέγιστο βαθμό μεγέθυνσης που υποστηρίζει η εφαρμογή, και όπως φαίνεται, διακρίνονται άνετα τα διάφορα αντικείμενα μέσα στις αίθουσες των διάφορων ορόφων. Σημαντικό είναι το γεγονός ότι το βαθμό αυτό μεγέθυνσης τον έχω επιτύχει με τη βοήθεια του JavaScript σε συνδυασμό με το Webview του Android, εφόσον όπως ανέφερα και προηγουμένως, η κλάση MapView του Android υποστήριζε πολύ λιγότερο βαθμό μεγέθυνσης.

Κεφάλαιο 4 - Εικόνα 4.6
Μενού εφαρμογής Tracker

Στην Εικόνα 4.6 φαίνονται τα διάφορα μενού που είναι διαθέσιμα στο πρώτο μέρος της εφαρμογής, τον Tracker. Στο πρώτο σχήμα φαίνονται οι επιλογές που δίνονται για εμφάνιση και αφαίρεση τόσο των αντικειμένων που βρίσκονται σε κάθε όροφο, όπως επίσης και της κάτοψης των ορόφων του κτιρίου. Το δεύτερο σχήμα, αποτελεί το κύριο μενού του Tracker, το οποίο και περιέχει το Logger,δηλαδή το δεύτερο μέρος της εφαρμογής CLODA, αλλά και τις προτιμήσεις, όπου ο χρήστης της εφαρμογής μπορεί να καθορίζει υπό ποιες συνθήκες επιθυμεί να χρησιμοποιεί την εφαρμογή. Ακολουθούν οι επιλογές Βοήθεια και Σχετικά οι οποίες και περιέχουν βοήθεια σχετικά με την κάθε επιλογή που υπάρχει στην εφαρμογή και συγκεκριμένα πάνω στο ActionBar, και τη επιλογή Σχετικά αναφέρει σχετικές πληροφορίες όσον αφορά τη δημιουργία της συγκεκριμένης εφαρμογής και, κάτω από ποιες προϋποθέσεις δημιουργήθηκε.

Κεφάλαιο 4 - Εικόνα 4.7
Πρόσθεση και Αφαίρεση αντικειμένων και κάτοψης ορόφων

Η εφαρμογή CLODA δίνει στο χρήστη τη δυνατότητα να προσθέτει και να αφαιρεί τα αντικείμενα από το κτίριο. Αυτό μπορεί να διευκολύνει το χρήστη ώστε να μπορέσει να επεξεργαστεί καλύτερα την κάτοψη του συγκεκριμένου ορόφου και να εντοπίσει πιο εύκολα την αίθουσα που ψάχνει. Επιπλέον, ο χρήστης μπορεί να αφαιρεί τις κατόψεις των ορόφων ώστε να ενσωματώσει άλλους ορόφους. Το πρώτο σχήμα απεικονίζει το $2^{ο}$ όροφο του κτιρίου της Πληροφορικής όπως είναι, προβάλλοντας τα αντικείμενα που έχουν σαρωθεί σε αυτόν. Στο δεύτερο σχήμα φαίνεται το μενού επιλογών το οποίο βρίσκεται πάνω στο ActionBar ενώ, στο τρίτο σχήμα απεικονίζεται το κτίριο μόνο με την κάτοψη του συγκεκριμένου ορόφου. Το τελευταίο σχήμα δείχνει το κτίριο όπως αυτό φαίνεται στα Google Maps.

Κεφάλαιο 4 - Εικόνα 4.8
Σφάλμα στην εισαγωγή του αρχείου τοποθεσιών (radiomap)

Η διπλανή εικόνα, παρουσιάζει το μήνυμα που εμφανίζεται στον χρήστη όταν προσπαθήσει να εντοπίσει τον εαυτό του μέσα στο κτίριο είτε επιλέγοντας το κουπί Track Me από το ActionBar, είτε χρησιμοποιώντας την επιλογή Find Me από το ActionBar.

Κεφάλαιο 4 - Εικόνα 4.9
Επιλογή προτιμήσεων του χρήστη

Η εικόνα 4.9 παρουσιάζει την επιλογή Προτιμήσεις (Preferences) από το κυρίως μενού, όπου στο πρώτο σχήμα είναι ενεργοποιημένη η επιλογή Floor Mode ενώ στο δεύτερο σχήμα είναι απενεργοποιημένη. Η επιλογή αυτή, όταν είναι ενεργοποιημένη, επιτρέπει στο χρήστη καθώς κινείται, ανά πάσα χρονική στιγμή, να γνωρίζει σε ποιο όροφο βρίσκεται. Αυτό επιτυγχάνεται με τη χρήση του αλγορίθμου που περιγράφηκε σε προηγούμενο κεφάλαιο, και ο οποίος ενόσω ο χρήστης κινείται εντοπίζει μέσω των APs σε ποιο ακριβώς όροφο βρίσκεται και αμέσως φορτώνει την κατάλληλη κάτοψη πάνω στο κτίριο. Όταν η επιλογή αυτή είναι απενεργοποιημένη, ο χρήστης δε θα γνωρίζει σε ποιο όροφο βρίσκεται αλλά θα μπορεί πιο εύκολα να εναλλάσσει τις κατόψεις των ορόφων προκειμένου να βλέπει τα αντικείμενα που είναι τοποθετημένα σε κάθε αίθουσα.

Το τρίτο σχήμα απεικονίζει τον File Browser από τον οποίο ο χρήστης μπορεί να επιλέξει την τοποθεσία στην οποία βρίσκεται το αρχείο radiomap mean.txt, στο οποίο βρίσκονται και οι μετρήσεις οι οποίες θα χρησιμοποιηθούν για να δείχνουν ανά πάσα χρονική στιγμή την τοποθεσία του χρήστη καθώς αυτός κινείται μέσα στο κτίριο. Προκειμένου η εφαρμογή να λειτουργεί ορθά, θα πρέπει ο χρήστης να εισαγάγει το αρχείο με το όνομα radiomap mean.txt καθώς σε αντίθετη περίπτωση θα λαμβάνει μήνυμα λάθους.

Κεφάλαιο 4 - Εικόνα 4.10
Επιλογή ορόφων για προσαρμογή στο κτίριο

Η εικόνα 4.10 απεικονίζει την επιλογή ορόφων στο πρώτο μέρος της εφαρμογής. Συγκεκριμένα, ο χρήστης μπορεί πατώντας το κάτω μενού, να επιλέξει ποιανού ορόφου την κάτοψη επιθυμεί να τοποθετήσει πάνω στο κτίριο. Αμέσως μετά την επιλογή του, αναλόγως ποιο όροφο επέλεξε, προσαρμόζεται η κάτοψη αυτού πάνω από το κτίριο.

Κεφάλαιο 4 - Εικόνα 4.11
Σάρωση αντικειμένου

Η εικόνα 4.11, παρουσιάζει τη διαδικασία σάρωσης ενός αντικειμένου. Συγκεκριμένα, το πρώτο σχήμα παρουσιάζει το σαρωτή γραμμικού κώδικα (barcode scanner) ο οποίος και έχει ενσωματωθεί στη δική μου εφαρμογή με τη βοήθεια της εξωτερικής βιβλιοθήκης ZXing. Στο

δεύτερο σχήμα φαίνεται η σάρωση ενός βιβλίου και στο τελευταίο σχήμα, απεικονίζονται τα στοιχεία αυτού του βιβλίου όπως υπάρχουν στο Google Shopping AP|I. Ουσιαστικά, κάθε φορά που γίνεται σάρωση ενός αντικειμένου, μέσω του γραμμικό του κώδικα (barcode), γίνεται αναζήτησή του στο Google Shopping API και αναλόγως του ποια στοιχεία από αυτά που ψάχνει η εφαρμογή (π.χ. τιμή, τίτλο, μάρκα κ.λ.π) υπάρχουν, αυτά εμφανίζει στο χρήστη. Εάν κάποια από τα στοιχεία που πρέπει να αποθηκευτούν στη βάση δεν υπάρχουν τότε ζητείται από το χρήστη να τα εισαγάγει.

Κεφάλαιο 4 - Εικόνα 4.12
Σάρωση αντικειμένου που υπάρχει ήδη στη βάση δεδομένων

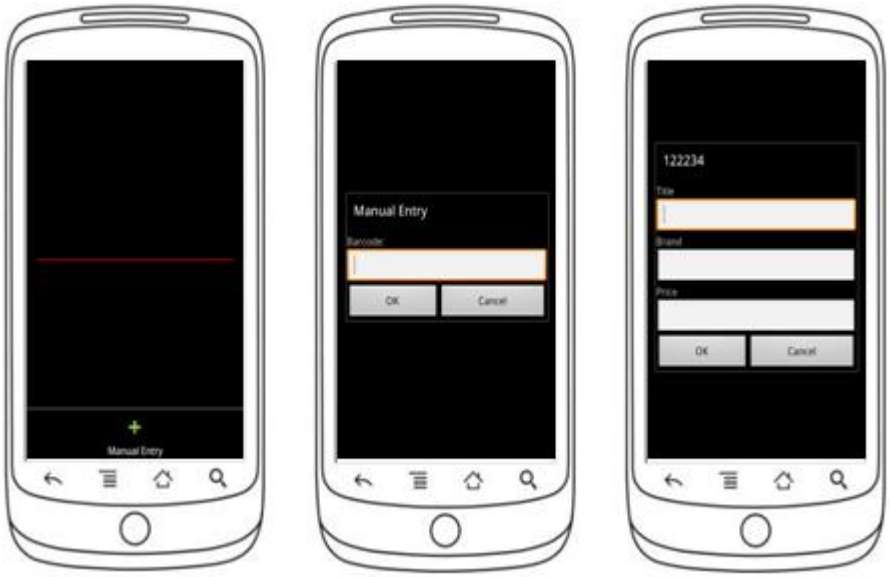
Κεφάλαιο 4 - Εικόνα 4.13
Προβολή στοιχείων αντικειμένων

Στην εικόνα 4.12 φαίνεται η σάρωση ενός βιβλίου (Σχήμα 1) ενώ, στο δεύτερο σχήμα φαίνεται το μήνυμα που εμφανίζεται στο χρήστη σε περίπτωση που το αντικείμενο που σαρώθηκε υπάρχει ήδη μέσα στη βάση δεδομένων. Σε τέτοια περίπτωση, ζητείται από το χρήστη να επιλέξει εάν θέλει να τοποθετήσει ξανά το αντικείμενο αυτό κάπου αλλού μέσα στο κτίριο και επομένως να το αποθηκεύσει ξανά στη βάση δεδομένων.

Η εικόνα 4.13 παρουσιάζει τα αντικείμενα που έχουν σαρωθεί και εμφανίζονται στο συγκεκριμένο όροφο. Το πρώτο σχήμα παρουσιάζει ένα αντικείμενο του οποίου τα στοιχεία ανευρέθηκαν στο Google Shopping API, μαζί και η εικόνα η οποία και εμφανίζεται με όλα τα υπόλοιπα στοιχεία του. Επιπλέον πληροφορία αποτελεί και η θέση του αντικειμένου δηλαδή, το γεωγραφικό μήκος και πλάτος της θέσης τους. Το δεύτερο σχήμα απεικονίζει και πάλι ένα αντικείμενο που βρίσκεται στη βάση δεδομένων με μόνη διαφορά την εικόνα. Συγκεκριμένα, σε περιπτώσεις αντικειμένων τα οποία δεν εντοπίζονται στο Google Shopping API, εμφανίζεται πάντα η εικόνα που φαίνεται σε αυτό το σχήμα.

Κεφάλαιο 4 - Εικόνα 4.14
Επιλογή Βοήθειας και Σχετικά

Στην πιο πάνω εικόνα φαίνονται οι επιλογές Βοήθεια και Σχετικά. Η επιλογή Βοήθεια, παρέχει στο χρήστη πληροφορίες που θα τον βοηθήσουν να χρησιμοποιήσει σωστά την εφαρμογή. Στο τρίτο σχήμα φαίνεται το περιεχόμενο της επιλογής Σχετικά η οποία και περιέχει πληροφορίες σχετικά με τις συνθήκες υπό τις οποίες αναπτύχθηκε αυτή η εφαρμογή.



Κεφάλαιο 4 - Εικόνα 4.15
Χειροκίνητη εισαγωγή αντικειμένου

Η εικόνα 4.15 απεικονίζει τη χειροκίνητη εισαγωγή ενός αντικειμένου. Στο πρώτο σχήμα φαίνεται η επιλογή χειροκίνητης εισαγωγής και στο δεύτερο σχήμα παρουσιάζεται η ετικέτα που εμφανίζεται στο χρήστη για να καταγράψει το γραμμικό κώδικα (barcode) του

αντικειμένου. Τέλος, στο τρίτο σχήμα παρουσιάζεται η ετικέτα που εμφανίζεται στο χρήστη για να εισαγάγει και τα υπόλοιπα στοιχεία εφόσον το συγκεκριμένο αντικείμενο δεν υπάρχει στο Google Shopping API.

Κεφάλαιο 4 - Εικόνα 4.16
Επεξεργασία αντικειμένου

Στην εικόνα 4.16 φαίνεται η διαδικασία επεξεργασίας ενός αντικειμένου που είναι αποθηκευμένο στη βάση δεδομένων. Το πρώτο σχήμα δείχνει ένα παράθυρο πληροφοριών (info window) το οποίο περιέχει τις πληροφορίες του αντικειμένου και οι οποίες είναι, ο γραμμικός κώδικας (barcode), η θέση του αντικειμένου, η τιμή του, η μάρκα και ο τίτλος του και τέλος, ο όροφος στον οποίο βρίσκεται. Στο ίδιο παράθυρο βρίσκεται μία εικόνα που δείχνει το συγκεκριμένο αντικείμενο. Ο χρήστης έχει τη δυνατότητα πατώντας πάνω στο παράθυρο πληροφοριών να επεξεργαστεί τις πληροφορίες του αντικειμένου. Συγκεκριμένα, μόλις ο χρήστης πατήσει πάνω στο παράθυρο πληροφοριών, εμφανίζεται η εικόνα που παρουσιάζεται στο δεύτερο σχήμα. Στη συνέχεια, ο χρήστης μπορεί να πατήσει στο κουμπί επεξεργασίας, που βρίσκεται πάνω στο ActionBar και έτσι, θα μπορεί να κάνει οποιεσδήποτε αλλαγές επιθυμεί σε οποιοδήποτε από τα πεδία εκτός από την τοποθεσία, εφόσον του δίνεται ξεχωριστή δυνατότητα μετακίνησης του αντικειμένου. Μόλις ολοκληρώσει τις αλλαγές, ο χρήστης μπορεί να πατήσει του κουμπί αποθήκευσης και οι αλλαγές τότε θα αποθηκευτούν στη βάση δεδομένων. Πέρα από την επεξεργασία του αντικειμένου, ο χρήστης πατώντας το δεύτερο κουμπί από το ActionBar, μπορεί να διαγράψει το αντικείμενο από τη βάση δεδομένων. Τέλος, παρέχεται η δυνατότητα στο χρήστη να μετακινήσει το αντικείμενο του από την παρούσα θέση σε κάποια άλλη. Για να το επιτύχει αυτό, μπορεί να επιλέξει το τρίτο κουμπί από το ActionBar. Ακολούθως, εμφανίζεται στο χρήστη ο χάρτης του κτιρίου και ένα μήνυμα ενημερώνοντας τον πως το αντικείμενο θα μετακινηθεί στο σημείο που αυτός θα

αγγίξει. Μόλις ο χρήστης αγγίξει σε οποιοδήποτε σημείο πάνω στο κτίριο, το αντικείμενο αυτόματα θα μετακινηθεί στη νέα του θέση και η αλλαγή θα ενημερωθεί και στη βάση δεδομένων.

Στην πιο κάτω εικόνα (Εικόνα 4.17), φαίνεται η διαδικασία της επεξεργασίας στο πρώτο σχήμα, αλλά και το γεγονός ότι δεν καθίσταται δυνατόν να αλλάξει ο χρήστης την τοποθεσία του αντικειμένου από μόνος του, εφόσον δεν είναι δυνατόν να γνωρίζει το γεωγραφικό μήκος και πλάτος μίας πιθανής θέσης για το αντικείμενο. Για αυτό το λόγο, όπως φαίνεται και στο δεύτερο σχήμα, το συγκεκριμένο πεδίο δεν είναι επεξεργάσιμο. Ωστόσο, εάν ο χρήστης επιθυμεί να μετακινήσει το αντικείμενο, μπορεί άνετα να το πράξει όπως αναφέρθηκε πιο πάνω.

Κεφάλαιο 4 - Εικόνα 4.17
Αδύνατη η αλλαγή της θέσης του αντικειμένου χειροκίνητα

Κεφάλαιο 4 - Εικόνα 4.18
Μενού επεξεργασίας

Η εικόνα 4.18, απεικονίζει το μενού επιλογών που παρουσιάζεται στον χρήστη όταν επιλέξει να επεξεργαστεί κάποιο αντικείμενο. Όπως φαίνεται και από το πρώτο σχήμα, ο χρήστης μπορεί είτε να επιστρέψει στο πρώτο μέρος της εφαρμογής, δηλαδή τον Tracker, είτε να πάρει σχετική βοήθεια για τη χρησιμότητα καθενός από τα κουμπιά που υπάρχουν πάνω στο ActionBar.

Κεφάλαιο 4 - Εικόνα 4.19
Logger

Κεφάλαιο 4 - Εικόνα 4.20
Σφάλμα – Μη εύρεση φακέλου για αποθήκευση αρχείου μετρήσεων

Στην εικόνα 4.19 παρουσιάζεται το δεύτερο μέρος της εφαρμογής CLODA, ο Logger. Συγκεκριμένα, ο Logger, όπως αναφέρθηκε και σε προηγούμενο κεφάλαιο, χρησιμοποιείται για να ληφθούν μετρήσεις οι οποίες αποθηκεύονται σε ένα αρχείο το λεγόμενο rss log και το οποίο στη συνέχεια τυγχάνει κάποιας επεξεργασίας από τον εξυπηρετητή διανομής (distribution server), και ακολούθως παράγεται ένα αρχείο radiomap mean το οποίο και χρησιμοποιείται από το πρώτο μέρος της εφαρμογής, τον Tracker, προκειμένου ο χρήστης να μπορεί να εντοπίζει τον εαυτό του ανά πάσα χρονική στιγμή μέσα στο κτίριο.

Στο πρώτο σχήμα της εικόνας φαίνεται το περιβάλλον του Logger, ενώ στο δεύτερο σχήμα, απεικονίζεται η διαδικασία λήψης μετρήσεων, όπου αφού ο χρήστης επέλεξε κάποιο σημείο, παίρνει μετρήσεις και στις 4 κατευθύνσεις (0,90,180,270). Στο τελευταίο σχήμα παρουσιάζεται ένα μήνυμα επιτυχίας στο χρήστη εάν τα δείγματα έχουν καταγραφεί με επιτυχία.

Η εικόνα 4.20 παρουσιάζει το μήνυμα σφάλματος που εμφανίζεται στον χρήστη όταν αυτός προσπαθήσει να αποθηκεύσει μετρήσεις χωρίς πρώτα να καθορίσει το φάκελο στον οποίο θα αποθηκευτεί αυτό το αρχείο.

Κεφάλαιο 4 - Εικόνα 4.21
Καθορισμός προτιμήσεων χρήστη

Η εικόνα 4.21 απεικονίζει την επιλογή προτιμήσεων στο δεύτερο μέρος της εφαρμογής, τον Logger. Το πρώτο σχήμα παρουσιάζει το κυρίως μενού του Logger το οποίο και αποτελείται από τον Tracker, δηλαδή δυνατότητα να επιστρέψει ο χρήστης στο πρώτο μέρος της εφαρμογής, τον Tracker. Επιπλέον, το κυρίως μενού περιέχει τη Βοήθεια και τα Σχετικά που θα επεξηγηθούν πιο κάτω και τέλος, την επιλογή προτιμήσεις όπου ο χρήστης μπορεί να θέσει τις ρυθμίσεις που επιθυμεί. Αυτό σημαίνει ότι μπορεί να επιλέξει τον αριθμό των δειγμάτων που επιθυμεί να λαμβάνει κάθε φορά και που όπως φαίνεται από το τρίτο σχήμα μπορεί να επιλέξει μεταξύ 5, 10, 15, 20, 25 και 30 δειγμάτων. Επιπρόσθετα, ο χρήστης μπορεί να καθορίσει το διάστημα μεταξύ των δειγμάτων και μπορεί να επιλέξει από 0.5, 1 ή 2 δευτερόλεπτα.

Εικόνα 4.22
Καθορισμός προτιμήσεων χρήστη

Η πιο πάνω εικόνα (Εικόνα 4.22), απεικονίζει επιπλέον ρυθμίσεις που μπορεί να επιλέξει ο χρήστης και οι οποίες είναι οι εξής. Αρχικά ο χρήστης καλείται να επιλέξει σε ποιο φάκελο στο σύστημα αρχείων επιθυμεί να αποθηκευτεί το αρχείο με τις μετρήσεις. Ακολούθως, πρέπει να ορίσει ένα όνομα στο συγκεκριμένο αρχείο αποτελεσμάτων ώστε στη συνέχεια να μπορεί να το εντοπίσει εύκολα και να το φορτώσει στον εξυπηρετητή προκειμένου να λάβει το radiomap αρχείο ποτ θα δείχνει τη θέση του χρήστη στο κτίριο καθώς αυτός κινείται σε αυτό.

Κεφάλαιο 4 - Εικόνα 4.23
Επιλογή Βοήθεια και Σχετικά

Στην εικόνα 4.23 φαίνονται οι επιλογές Βοήθεια και Σχετικά όπου, η επιλογή Βοήθεια παρέχει στο χρήστη οδηγίες για τη σωστή χρήση του Logger, προκειμένου να ληφθούν σωστά οι μετρήσεις και τα αποτελέσματα να είναι εξίσου ορθά. Η επιλογή Σχετικά αφορά λεπτομέρειες της εφαρμογής CLODA, όπως για παράδειγμα κάτω από ποιες συνθήκες δημιουργήθηκε.

Κεφάλαιο 4 - Εικόνα 4.24
Επιλογή Ορόφου στο Logger

Η εικόνα 4.24 παρουσιάζει τη δυνατότητα εφαρμογής κάτοψης ορόφου πάνω από το κτήριο της Πληροφορικής, στο δεύτερο μέρος της εφαρμογής, τον Logger, έτσι ώστε να διευκολυνθεί η καταγραφή των μετρήσεων. Το πρώτο σχήμα απεικονίζει το κτίριο της Πληροφορικής όπως αυτό φαίνεται μέσω των Google Maps. Όταν ο χρήστης επιλέξει το κάτω μενού του παρέχεται η δυνατότητα να εφαρμόσει την κάτοψη οποιουδήποτε ορόφου επιθυμεί. Μπορεί να επιλέξει ένα από τους τέσσερις ορόφους αναλόγως σε ποιον από αυτούς βρίσκεται και επιθυμεί να πάρει μετρήσεις. Στο τελευταίο σχήμα φαίνεται το κτίριο της Πληροφορικής πάνω από τα Google Maps μετά από την εφαρμογή ενός από τους τέσσερις ορόφους.

Κεφάλαιο 4 - Εικόνα 4.25
Μήνυμα για έξοδο από εφαρμογή λόγω έλλειψης σύνδεσης με το wifi

Κεφάλαιο 4 - Εικόνα 4.26
Μήνυμα εξόδου από εφαρμογή

Στην εικόνα 4.25 φαίνεται το μήνυμα που εμφανίζεται στο χρήστη σε περίπτωση που είτε δεν υπάρχει σύνδεση με το Wifi προτού αυτός ενεργοποιήσει την εφαρμογή, είτε καθώς τη χρησιμοποιεί χάνεται για κάποιο λόγο η σύνδεση. Συγκεκριμένα, ενημερώνεται ο χρήστης για τη μη διαθεσιμότητα Wifi σύνδεσης και επιπλέον ότι η εφαρμογή θα τερματίσει εφόσον χρειάζεται Wifi προκειμένου να λειτουργεί ορθά.

Η εικόνα 4.26 παρουσιάζει το μήνυμα που παρουσιάζεται στον χρήστη όταν αυτός επιλέξει να τερματίσει την εφαρμογή. Ο χρήστης επιλέγει κατά πόσο επιθυμεί να εγκαταλείψει ή όχι την εφαρμογή και πράττει αναλόγως επιλέγοντας ναι ή όχι αντίστοιχα.

Κεφάλαιο 4 - Εικόνα 4.27
FindMe και TrackMe λειτουργία

Στην πιο πάνω εικόνα φαίνονται οι λειτουργίες FindMe και TrackMe αντίστοιχα. Συγκεκριμένα, όταν ο χρήστης επιλέξει το TrackMe, τότε η μπλε τελεία που παρουσιάζεται στο χάρτη, αποτελεί και τη θέση στην οποία βρίσκεται ο χρήστης τη χρονική εκείνη στιγμή. Όσον αφορά στο TrackMe, όταν ο χρήστης το επιλέξει, μπορεί ανά πάσα χρονική στιγμή να γνωρίζει τη θέση του στο κτίριο. Έτσι, μπορεί να κατευθύνεται σε μία αίθουσα και να βλέπει ακριβώς τη διαδρομή που ακολουθεί κάτι που είναι πολύ χρήσιμο εφόσον εάν γνωρίζει που βρίσκεται κάποιο αντικείμενο που ψάχνει, θα τον βοηθήσει να φτάσει γρήγορα στον προορισμό του καθώς διαρκώς θα βλέπει κατά πόσο κινείται προς τη σωστή κατεύθυνση.

4.4 Βάση Δεδομένων

4.4.1 Sag Library For PHP

```php
<?php
/*
 * A real quick example to get and update a doc.
 */
require_once('./src/Sag.php');
$sag = new Sag('127.0.0.1', '5984');

// Select the database that holds our blog's data.
$sag->setDatabase('blog');

try {
  //Get a blog post (a StdClass)...
  $post = $sag->get('postID')->body;

  //...update its info...
  $post->views++;

  //..and send it back to the couch.
  if(!$sag->put($post->_id, $post)->body->ok) {
    error_log('Unable to log a view to CouchDB.');
  }
}
catch(SagCouchException $e) {
  //The requested post doesn't exist - oh no!
  if($e->getCode() == "404") {
    $e = new Exception("That post doesn't exist.");
  }

  throw $e;
}
catch(SagException $e) {
  //We sent something to Sag that it didn't expect.
  error_log('Programmer error: '.$e->getMessage());
}
?>
```

Κεφάλαιο 4 - Εικόνα 4.28
Παράδειγμα για λήψη και ενημέρωση ενός εγγράφου με τη χρήση της Sag βιβλιοθήκης

Η Sag [36] είναι μία συλλογή από βιβλιοθήκες οι οποίες ενώνονται στη CouchDB. Βασική αρχή της είναι η απλότητα, δημιουργώντας μία ισχυρή διεπαφή με πολύ μικρή επιβάρυνση και η οποία μπορεί να ενσωματωθεί με τη δομή οποιασδήποτε εφαρμογής.

Δεν έχει κάποια ιδιαίτερα πλαίσια, κλάσεις για έγγραφα ή λανθασμένες κλήσεις HTTP. Επιπλέον, η χρήση είναι εύκολη εφόσον εξακολουθούν να χρησιμοποιούνται συναρτήσεις όπως η get(), post() και put(). Η Sag βιβλιοθήκη, παρέχει ακόμη αυτόματο έλεγχο δηλαδή, κάθε φορά που κυκλοφορεί μία νέα έκδοση, αυτόματα ελέγχεται σε ορισμένες βάσεις μία από τις οποίες είναι και η CouchDB. Τέλος, παρέχει το ίδιο API στην php και JSBorn.

4.4.2 Στοιχεία Υλοποίησης συστατικών στοιχείων

4.4.2.1 Documents

Τα έγγραφα όπως ανέφερα και προηγουμένως, είναι το βασικό συστατικό στοιχείο της CouchDB βάσης δεδομένων. Πιο κάτω φαίνεται ένα παράδειγμα ενός JSON εγγράφου. Όπως είναι φανερό και από την εικόνα, τα πεδία που αποτελούν το συγκεκριμένο έγγραφο είναι 10 και τα οποία είναι το _id και _rev που δημιουργούνται αυτόματα από τη βάση, αλλά και τα

υπόλοιπα 8 (barcode, brand, floor, image, pointX, pointY, price, title) τα οποία αποτελούν στοιχεία του κάιε αντικειμένου που καταχωρείται στη βάση δεδομένων του συστήματος.


```
{
   "_id": "379916e9f02d1ff1a6bc3125af00e505",
   "_rev": "3-72c75fe2547202f9804183c686f76acb",
   "barcode": "4006381105330",
   "brand": "Stabilo",
   "floor": "1",
   "image": "http://www.hyatts.com/eCom/images/N/N47055.jpg",
   "pointX": "33.411027789115906",
   "pointY": "35.144588540720655",
   "price": "1.19",
   "title": "Stabilo Point 88-55 Violet"
}
```


Κεφάλαιο 4 - Εικόνα 4.29
Δείγμα JSON εγγράφου

Στην πιο κάτω εικόνα, φαίνεται ένα σύνολο από όλα τα έγγραφα που είναι καταχωρημένα στη βάση δεδομένων couchDB, όπως ακριβώς παρουσιάζονται στη διεπαφή Futon.

Κεφάλαιο 4 - Εικόνα 4.30
<key,value> έγγραφα όπως φαίνονται στη couchDB

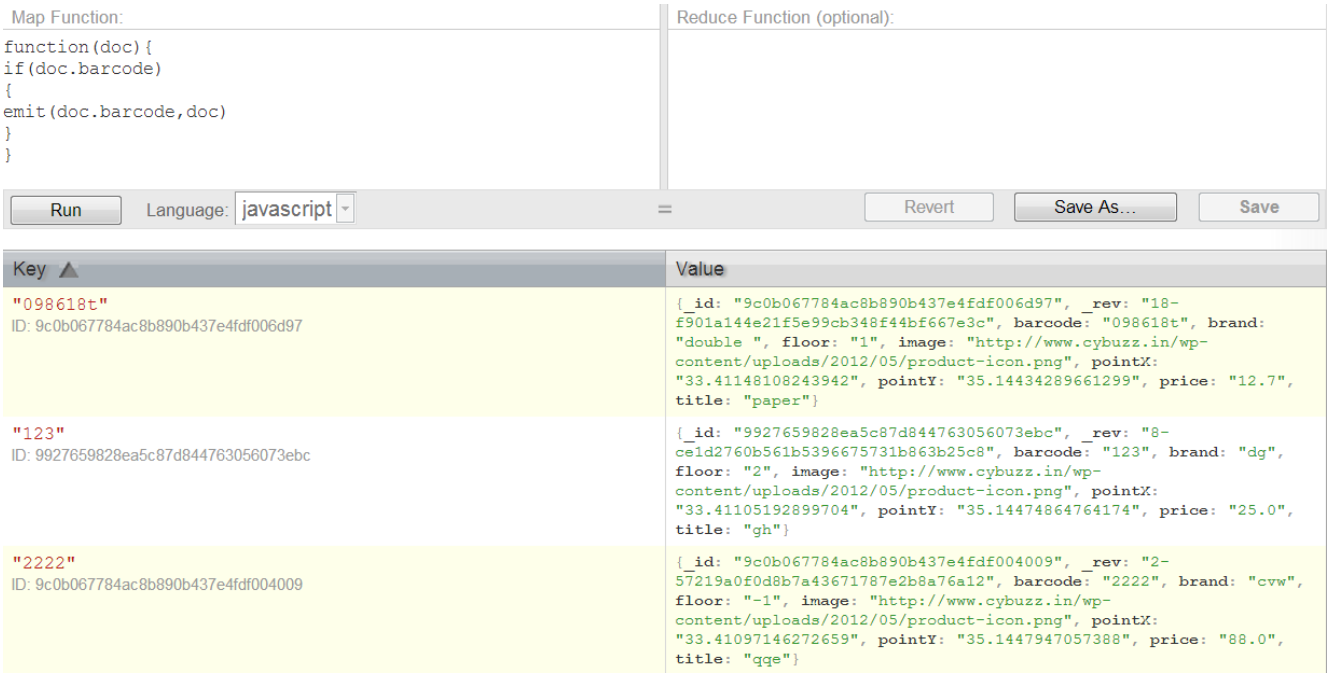
Προκειμένου να αποθηκευτούν τα έγγραφα τη βάση δεδομένων, εφόσον πρώτα ληφθούν τα στοιχεία του αντικειμένου και δημιουργηθεί ένα JSON αντικείμενο με αυτά τα δεδομένα, χρησιμοποιείται η php. Συγκεκριμένα, μέσω της παραμέτρου $_POST λαμβάνεται το JSON αντικείμενο το οποίο στη συνέχεια, με τη βοήθεια συναρτήσεων την Sag βιβλιοθήκης που περιγράφηκε πιο πάνω, αποστέλλεται το JSON αντικείμενο με τα δεδομένα στη βάση δεδομένων crowdspace. Πιο κάτω φαίνεται κομμάτι από τον κώδικα ο οποίος πραγματοποιεί εισαγωγή ενός JSON αντικειμένου μέσα στη βάση δεδομένων.

```
$bones = new Bones();
$product=json_decode($_POST['value']);
$bones->couch->post($product);
```

Κεφάλαιο 4 - Εικόνα 4.31
Εισαγωγή JSON αντικειμένου ως έγγραφο στη CouchDB

Επιπλέον, χρησιμοποιείται η συνάρτηση της php json_decode για αποκωδικοποίηση του JSON αντικοειμένου. Μέσω της συνάρτησης post που καλείται από το πεδίο couch της μεταβλητής bones, αποθηκεύονται οι πληροφορίες του αντικειμένου σαν ένα καινούργιο έγγραφο στη βάση δεδομένων CouchDB.

4.4.2.2 Views



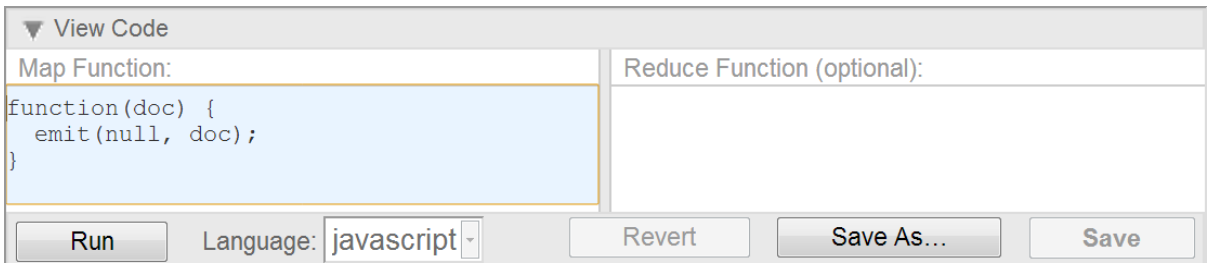
Κεφάλαιο 4 - Εικόνα 4.32
Συνάρτηση μίας προβολής και το αποτέλεσμα της εκτέλεσής της (Design View)

Στην πιο πάνω εικόνα, βλέπουμε μία μόνιμη προβολή (design view) και το αποτέλεσμα αυτής με βάση τα έγγραφα που είναι αποθηκευμένα στη βάση.

Συγκεκριμένα, η συνάρτηση είναι γραμμένη σε JavaScript και αποτελεί μία map function. Όλες οι map συναρτήσεις έχουν μία παράμετρο doc. Αυτό, είναι ένα έγγραφο στη βάση δεδομένων. Στη συγκεκριμένη περίπτωση, η map συνάρτηση ελέγχει κατά πόσο το έγγραφο περιέχει το χαρακτηριστικό barcode, και στη συνέχεια καλεί την ενσωματωμένη συνάρτηση emit() με αυτό το χαρακτηριστικό ως παράμετρο αλλά και ολόκληρο το έγγραφο. Η

συνάρτηση emit() πάντα παίρνει δύο παραμέτρους εκ των οποίων η πρώτη είναι το κλειδί (key), και η δεύτερη η τιμή (value). Η συνάρτηση emit(key,value) έχει ως αποτέλεσμα τη δημιουργία μίας εγγραφής στο αποτέλεσμα της προβολής (view result). Η CouchDB, παίρνει οτιδήποτε περάσουμε μέσω της emit() και το αποθηκεύει σε μία λίστα, της οποίας κάθε γραμμή περιλαμβάνει το κλειδί και την τιμή της. Σημαντικό είναι και το γεγονός ότι η λίστα αυτή είναι ταξινομημένη με βάσει το key, όπου στη συγκεκριμένη περίπτωση είναι το barcode.

Η εικόνα πιο κάτω απεικονίζει ένα temporary view όπως αυτό παρουσιάζεται μέσω του Futon. Συγκεκριμένα, ο χρήστης υλοποιεί τη συνάρτηση που επιθυμεί και μπορεί να την τρέξει και ακολούθως να την αποθηκεύσει σαν design document.



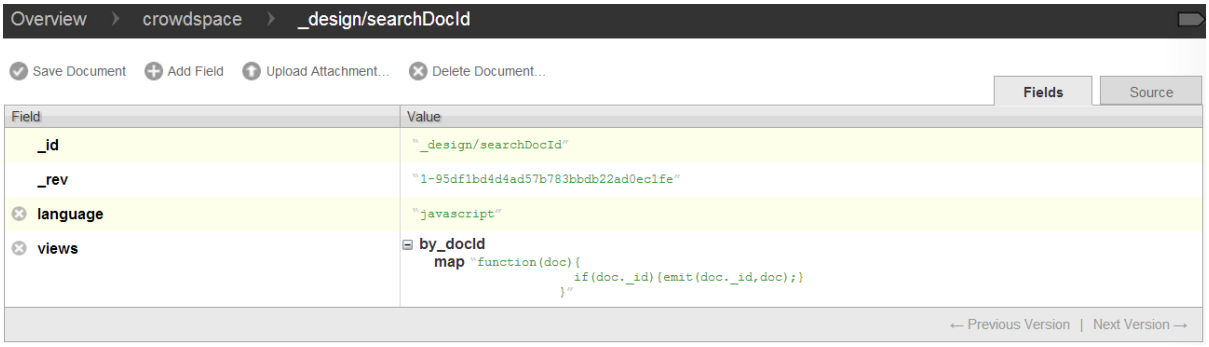
Κεφάλαιο 4 - Εικόνα 4.33
Temporary View

4.4.2.3 Design Documents

Η εικόνα που ακολουθεί παρουσιάζει μία λίστα με όλα τα design documents όπως αυτά φαίνονται στη Futon διεπαφή.

Κεφάλαιο 4 - Εικόνα 4.34
Λίστα με όλα τα Design Documents

Στις δύο επόμενες εικόνες φαίνεται το searchDocId,ένα δείγμα από ένα design document όπως ακριβώς παρουσιάζεται στο Futon. Το συγκεκριμένο έγγραφο σχεδιασμού επιστρέφει όλα τα αναγνωριστικά όλων των εγγράφων που είναι αποθηκευμένα στη βάση.



Κεφάλαιο 4 - Εικόνα 4.35
searchDocId – Παράδειγμα ενός design Document

```
{
    "_id": "_design/searchDocId",
    "_rev": "1-95df1bd4d4ad57b783bbdb22ad0ec1fe",
    "language": "javascript",
    "views": {
        "by_docId": {
            "map": "function(doc){\n\t\tif(doc._id){emit(doc._id,doc);}\n\t      }"
        }
    }
}
```

Κεφάλαιο 4 - Εικόνα 4.36
Δείγμα design document

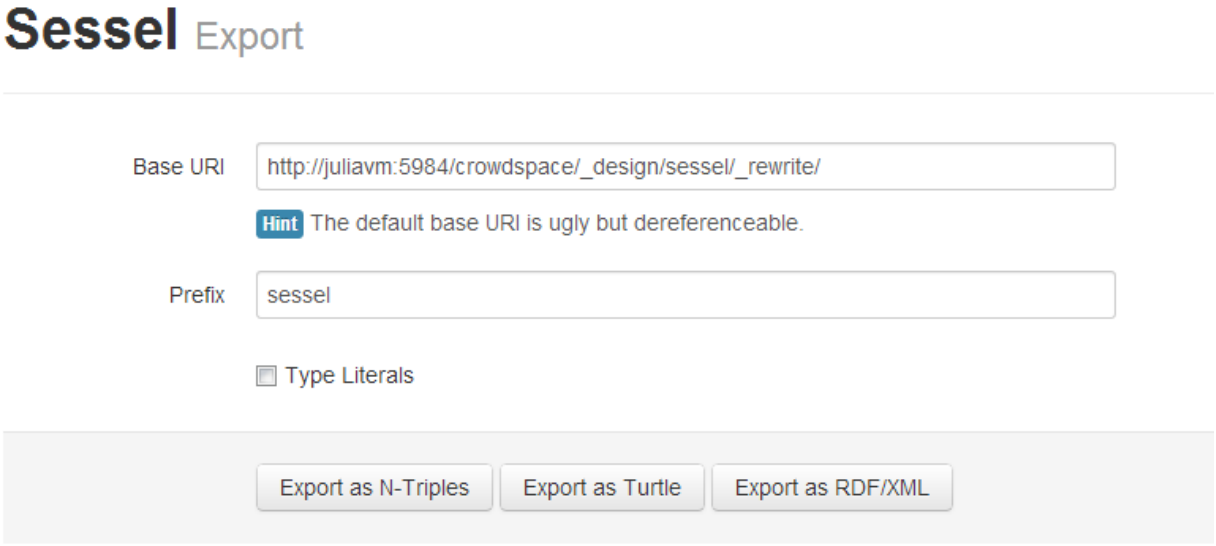
4.5 Ανοικτά Δεδομένα

4.5.1 Μετατροπή των JSON εγγράφων της CouchDB σε RDF

Λόγω των δυνατοτήτων που παρέχουν τα RDF, κρίθηκε αναγκαίο να μετατραπούν όλα τα JSON έγγραφα που είναι αποθηκευμένα στη CouchDB, σε RDF. Προκειμένου να γίνει αυτό, χρησιμοποίησα το Sessel.

Το Sessel [43], είναι μία εφαρμογή (CouchApp) κατάλληλη για couchDB βάση δεδομένων, και η οποία παράγει rdf τριάδες από JSON έγγραφα, τα οποία στη συνέχεια μπορούν να εξαχθούν σε διάφορες μορφές ή να αναζητηθούν μέσω του SPARQL, το οποίο και θα επεξηγηθεί περεταίρω στη συνέχεια.

Το Sessel, σχεδιάστηκε για να μπορεί να αντιγραφεί ή να ενσωματωθεί σε μία ήδη υπάρχουσα CouchDB βάση δεδομένων, η οποία και περιέχει τα δεδομένα που πρέπει να εξαχθούν σε rdf.



Κεφάλαιο 4 - Εικόνα 4.37
Γραφικό Περιβάλλον Sessel για εξαγωγή δεδομένων σε RDF

Προκειμένου να εξαχθούν τα JSON έγγραφα από τη βάση δεδομένων σε rdf, χρησιμοποίηση το πιο πάνω γραφικό περιβάλλον, ονομαζόμενο Sessel Export. Όπως φαίνεται και από την εικόνα, αναλόγως σε τι μορφή επιθυμεί ο κάθε χρήστης να εξαγάγει τα ήδη υπάρχοντα έγγραφά του, επιλέγει το κατάλληλο κουμπί. Με την επιλογή της εξαγωγής των εγγράφων σε rdf, οι τύποι του JSON αντιστοιχούνται σε τύπους toy XML. Συγκεκριμένα, τόσο οι συμβολοσειρές όσο και οι πίνακες, αντικείμενα, αριθμοί, αντιστοιχούνται σε xsd:string και οι αριθμοί σε xsd:integer.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns:sessel="http://juliavm:5984/crowdspace/_design/sessel/_rewrite/"
  xmlns:sesselVocab="http://juliavm:5984/crowdspace/_design/sessel/_rewrite/vocab/#">
  <rdf:Description rdf:about="http://juliavm:5984/crowdspace/_design/sessel/_rewrite/9c0b067784ac8b890b437e4fdf00215d">
    <sesselVocab:barcode rdf:datatype="xsd:string">aaaaasss</sesselVocab:barcode>
  </rdf:Description>
  <rdf:Description rdf:about="http://juliavm:5984/crowdspace/_design/sessel/_rewrite/9c0b067784ac8b890b437e4fdf00215d">
    <sesselVocab:brand rdf:datatype="xsd:string">fh</sesselVocab:brand>
  </rdf:Description>
  <rdf:Description rdf:about="http://juliavm:5984/crowdspace/_design/sessel/_rewrite/9c0b067784ac8b890b437e4fdf00215d">
    <sesselVocab:floor rdf:datatype="xsd:string">2</sesselVocab:floor>
  </rdf:Description>
  <rdf:Description rdf:about="http://juliavm:5984/crowdspace/_design/sessel/_rewrite/9c0b067784ac8b890b437e4fdf00215d">
    <sesselVocab:image rdf:datatype="xsd:string">http://www.cybuzz.in/wp-content/uploads/2012/05/product-icon.png</sesselVocab:image>
  </rdf:Description>
  <rdf:Description rdf:about="http://juliavm:5984/crowdspace/_design/sessel/_rewrite/9c0b067784ac8b890b437e4fdf00215d">
    <sesselVocab:pointX rdf:datatype="xsd:string">33.411274552345276</sesselVocab:pointX>
  </rdf:Description>
  <rdf:Description rdf:about="http://juliavm:5984/crowdspace/_design/sessel/_rewrite/9c0b067784ac8b890b437e4fdf00215d">
    <sesselVocab:pointY rdf:datatype="xsd:string">35.14440650096201</sesselVocab:pointY>
  </rdf:Description>
  <rdf:Description rdf:about="http://juliavm:5984/crowdspace/_design/sessel/_rewrite/9c0b067784ac8b890b437e4fdf00215d">
    <sesselVocab:price rdf:datatype="xsd:string">55.0</sesselVocab:price>
  </rdf:Description>
  <rdf:Description rdf:about="http://juliavm:5984/crowdspace/_design/sessel/_rewrite/9c0b067784ac8b890b437e4fdf00215d">
    <sesselVocab:title rdf:datatype="xsd:string">pol</sesselVocab:title>
  </rdf:Description>
```

Κεφάλαιο 4 - Εικόνα 4.38
Δείγμα JSON εγγράφων σε RDF μορφή

Πιο πάνω (Εικόνα 4.38), φαίνεται δείγμα από τα έγγραφα της βάσης δεδομένων μου σε rdf μορφή. Συγκεκριμένα, η πιο κάτω εικόνα απεικονίζει σε rdf τα στοιχεία ενός JSON εγγράφου, δηλαδή τα στοιχεία ενός αντικειμένου.

Με τη σύνδεση του Sessel CouchApp με τη δική μου CouchDB βάση δεδομένων, δημιουργήθηκε μία νέα προβολή στο γραφικό περιβάλλον, Futon. Η προβολή αυτή έχει το όνομα triples και όταν επιλεχθεί, φαίνονται τα στοιχεία κάθε αντικειμένου (εγγράφου) σε τριάδες, όπως φαίνεται και στην Εικόνα 4.30. Αυτό αποτελεί και το κλειδί (key). Τιμή (value), είναι ο τύπος των στοιχείων της τριάδας.

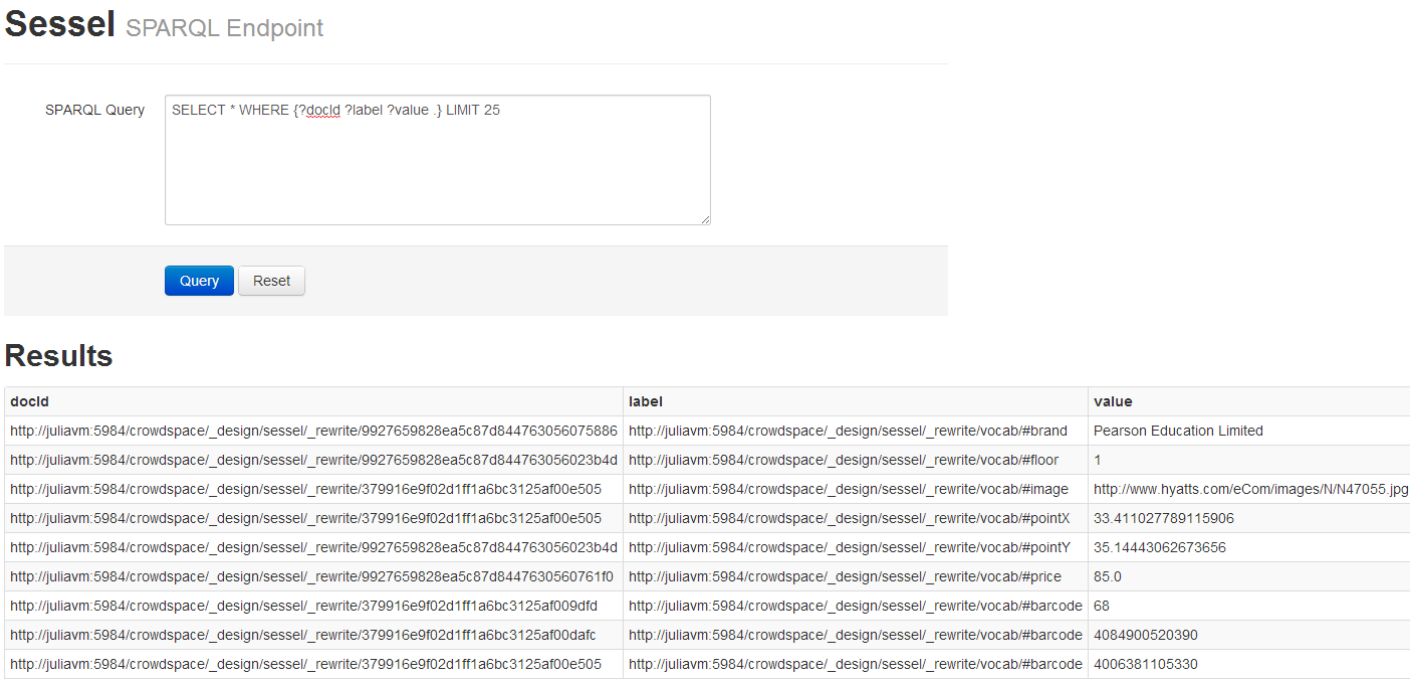
Key ▲	Value
["379916e9f02d1ff1a6bc3125af00dafc", "floor", ""0\""] ID: 379916e9f02d1ff1a6bc3125af00dafc	{objectType: "string"}
["379916e9f02d1ff1a6bc3125af00dafc", "image", ""http://www.cybuzz.in/wp-content/uploads/2012/05/product-icon.png\""] ID: 379916e9f02d1ff1a6bc3125af00dafc	{objectType: "string"}
["379916e9f02d1ff1a6bc3125af00dafc", "pointX", ""33.41106668114662\""] ID: 379916e9f02d1ff1a6bc3125af00dafc	{objectType: "string"}
["379916e9f02d1ff1a6bc3125af00dafc", "pointY", ""35.14475084088505\""] ID: 379916e9f02d1ff1a6bc3125af00dafc	{objectType: "string"}
["379916e9f02d1ff1a6bc3125af00dafc", "price", ""2.4\""] ID: 379916e9f02d1ff1a6bc3125af00dafc	{objectType: "string"}
["379916e9f02d1ff1a6bc3125af00dafc", "title", ""pens\""] ID: 379916e9f02d1ff1a6bc3125af00dafc	{objectType: "string"}
["379916e9f02d1ff1a6bc3125af00e505", "barcode", ""4006381105330\""] ID: 379916e9f02d1ff1a6bc3125af00e505	{objectType: "string"}
["379916e9f02d1ff1a6bc3125af00e505", "brand", ""Stabilo\""] ID: 379916e9f02d1ff1a6bc3125af00e505	{objectType: "string"}
["379916e9f02d1ff1a6bc3125af00e505", "floor", ""1\""] ID: 379916e9f02d1ff1a6bc3125af00e505	{objectType: "string"}
["379916e9f02d1ff1a6bc3125af00e505", "image", ""http://www.hyatts.com/eCom/images/N/N47055.jpg\""] ID: 379916e9f02d1ff1a6bc3125af00e505	{objectType: "string"}

Κεφάλαιο 4 - Εικόνα 4.39
Triples View in CouchDB

Το SPARQL [44], είναι ένα πρωτόκολλο το οποίο αποτελεί μέσο για μεταφορά SPARQL αναζητήσεων από πελάτες σε επεξεργαστές. Το SPARQL πρωτόκολλο περιγράφεται σε δύο τρόπους. Πρώτα, ως μία αφαιρετική διεπαφή ανεξάρτητη από οποιαδήποτε συγκεκριμένη υλοποίηση και δεύτερο σαν HTTP και SOAP συνδέσεις αυτής της διεπαφής.

Όπως ανέφερα και προηγουμένως, το RDF είναι ένας κατευθυνόμενος με ετικέτες γράφος που αναπαριστά πληροφορίες στο διαδίκτυο [45]. Αυτή η προδιαγραφή καθορίζει τη σύνταξη και σημασιολογία της γλώσσας αναζήτησης SPARQL για RDF. Το SPARQL, μπορεί να χρησιμοποιηθεί για να εκφράσει αναζητήσεις σε διάφορες πηγές δεδομένων, είτε τα δεδομένα αποθηκεύτηκαν σαν RDF ή προβλήθηκαν μέσω ενδιάμεσου λογισμικού. Επιπλέον, το SPARQL περιέχει δυνατότητες για αναζήτηση απαιτούμενων αλλά και προαιρετικών προτύπων γράφων μαζί με τους συνδέσμους και τους διαχωρισμούς τους.

Τέλος, το SPARQL υποστηρίζει επεκταμένο έλεγχο των τιμών καθώς επίσης και περιορισμό των αναζητήσεων μέσω RDF γράφου. Τα αποτελέσματα των S{ARQL αναζητήσεων μπορεί να είναι είτε σύνολα ή RDF γράφοι.



docId	label	value
http://juliavm:5984/crowdspace/_design/sessel/_rewrite/9927659828ea5c87d844763056075886	http://juliavm:5984/crowdspace/_design/sessel/_rewrite/vocab/#brand	Pearson Education Limited
http://juliavm:5984/crowdspace/_design/sessel/_rewrite/9927659828ea5c87d844763056023b4d	http://juliavm:5984/crowdspace/_design/sessel/_rewrite/vocab/#floor	1
http://juliavm:5984/crowdspace/_design/sessel/_rewrite/379916e9f02d1ff1a6bc3125af00e505	http://juliavm:5984/crowdspace/_design/sessel/_rewrite/vocab/#image	http://www.hyatts.com/eCom/images/N/N47055.jpg
http://juliavm:5984/crowdspace/_design/sessel/_rewrite/379916e9f02d1ff1a6bc3125af00e505	http://juliavm:5984/crowdspace/_design/sessel/_rewrite/vocab/#pointX	33.411027789115906
http://juliavm:5984/crowdspace/_design/sessel/_rewrite/9927659828ea5c87d844763056023b4d	http://juliavm:5984/crowdspace/_design/sessel/_rewrite/vocab/#pointY	35.14443062673656
http://juliavm:5984/crowdspace/_design/sessel/_rewrite/9927659828ea5c87d8447630560761f0	http://juliavm:5984/crowdspace/_design/sessel/_rewrite/vocab/#price	85.0
http://juliavm:5984/crowdspace/_design/sessel/_rewrite/379916e9f02d1ff1a6bc3125af009dfd	http://juliavm:5984/crowdspace/_design/sessel/_rewrite/vocab/#barcode	68
http://juliavm:5984/crowdspace/_design/sessel/_rewrite/379916e9f02d1ff1a6bc3125af00dafc	http://juliavm:5984/crowdspace/_design/sessel/_rewrite/vocab/#barcode	4084900520390
http://juliavm:5984/crowdspace/_design/sessel/_rewrite/379916e9f02d1ff1a6bc3125af00e505	http://juliavm:5984/crowdspace/_design/sessel/_rewrite/vocab/#barcode	4006381105330

Κεφάλαιο 4 - Εικόνα 4.40
Αποτέλεσμα SPARQL αναζήτησης στη δική μου βάση δεδομένων

Στην πιο πάνω εικόνα, φαίνεται το γραφικό περιβάλλον του Sessel SPARQL Endpoint, το οποίο και χρησιμοποιήθηκε για την εκτέλεση μίας αναζήτησης στη βάση δεδομένων με τα αντικείμενα. Όπως φαίνεται, η συγκεκριμένη αναζήτηση παρουσιάζει τα έγγραφα της βάσης δεδομένων σε τριάδες οι οποίες, έχουν τα ονόματα docId, το οποίο είναι και το αναγνωριστικό κάθε εγγράφου της βάσης, το label, δηλαδή τα στοιχεία του αντικειμένου (π.χ. τίτλος, μάρκα κ.λ.π) και τέλος, το value που περιέχει την τιμή κάθε χαρακτηριστικού, πεδίου του κάθε εγγράφου.

4.5.2 DBpedia

Η DBpedia [46] είναι μία πληθωποριστική κοινή προσπάθεια για εξαγωγή δομημένων πληροφοριών από τη Wikipedia, και καθιστά αυτές τις πληροφορίες διαθέσιμες στο διαδίκτυο. Η DBpedia, επιτρέπει στο χρήστη να δημιουργεί έξυπνα ερωτήματα προς τη Wikipedia, και να συνδέει τα διάφορα σύνολα δεδομένων από το διαδίκτυο στα δεδομένε της Wikipedia. Αυτή η τεχνολογία, μπορεί να εμπνεύσει νέους μηχανισμούς πλοήγησης, σύνδεσης και βελτίωσης της ίδιας της εγκυκλοπαίδειας.

Οι βάσεις γνώσης παίζουν πολύ σημαντικό ρόλο στην ενίσχυση της νοημοσύνης του διαδικτύου και της αναζήτησης επιχειρήσεων αλλά και στην ολοκλήρωση υποστηρικτικών πληροφοριών. Σήμερα, οι περισσότερες βάσεις γνώσης καλύπτουν μόνο συγκεκριμένους τομείς, και δημιουργούνται από σχετικά μικρές ομάδες μηχανικών γνώσης, και είναι πολυδάπανες στη διατήρηση καθώς οι τομείς αυτού αλλάζουν. Την ίδια στιγμή, η Wikipedia έχει γίνει μία από τις κεντρικές πηγές γνώσης της ανθρωπότητας, και η οποία διατηρείται από χιλιάδες συντελεστές.

Η DBpedia αξιοποιεί αυτή τη γιγαντιαία πηγή γνώσης εξάγοντας δομημένες πληροφορίες από τη Wikipedia και κάνοντάς τις προσβάσιμες στο διαδίκτυο κάτω από όρους. Για παράδειγμα, η Αγγλική έκδοση της DBpedia περιγράφει 3.77 εκατομμύρια πράγματα από τα οποία οι 764.000 είναι άτομα, 573.000 τόποι, 192.00 οργανισμοί κ.λ.π.

Επιπρόσθετα, παρέχονται τοπικές εκδόσεις της DBpedia σε 111 γλώσσες. Όλες αυτές οι εκδόσεις μαζί, περιγράφουν 20.8 εκατομμύρια πράγματα, από τα οποία τα 10.5 εκατομμύρια συμπίπτουν με έννοιες από την Αγγλική DBpedia.

Το ολοκληρωμένο σύνολο δεδομένων της DBpedia διαθέτει ετικέτες και αποσπάσματα για 10.3 εκατομμύρια μοναδικά πράγματα σε 111 διαφορετικές γλώσσες, 8 εκατομμύρια συνδέσμους σε εικόνες και 24.4 εκατομμύρια HTML συνδέσμους σε εξωτερικές ιστοσελίδες, 27.2 εκατομμύρια συνδέσμοι δεδομένων σε εξωτερικά RDF σύνολα δεδομένων, 55.8 εκατομμύρια συνδέσμοι σε κατηγορίες της Wikipedia. Το σύνολο δεδομένων αποτελείται από 1.89 δισεκατομμύρια κομμάτια πληροφοριών (RDF τριάδων).

Η DBpedia βάση γνώσεων έχει αρκετά πλεονεκτήματα σε σχέση με ήδη υπάρχουσες βάσεις γνώσεων. Αρχικά, καλύπτει πολλούς τομείς, αντιπροσωπεύει πραγματική συμφωνία κοινότητας και αυτόματα εξελίσσεται καθώς η Wikipedia αλλάζει, και είναι πραγματικά πολύγλωσση. Επιπλέον, επιτρέπει στο χρήστη να ρωτήσει ενδιαφέροντα πράγματα όπως για παράδειγμα, να ζητήσει όλους τους Ιταλούς μουσικούς από τον 18^{ο} αιώνα.

Ένα άλλο βασικό στοιχείο της είναι και το ότι περιέχει πληροφορίες για γεωγραφικές τοποθεσίες και είναι εσωτερικά συνδεδεμένη με άλλα γεωγραφικά δεδομένα όπως για παράδειγμα geonames, geocoordinates.

Ωστόσο, η DBpedia, όπως όλα έχει και κάποια μειονεκτήματα, όπως για παράδειγμα η χαμηλή ποιότητα των δεδομένων της αλλά και η ασφάλεια καθώς δε συνάδει πλήρως η DBpedia με τα δεδομένα της.

Μερικά επιπλέον χαρακτηριστικά της DBpedia είναι το γεγονός ότι χρησιμοποιεί τα rdf ως ένα ευέλικτο μοντέλο δεδομένων για να αναπαριστά εξαγόμενες πληροφορίες και για δημοσίευση τους στο διαδίκτυο. Επιπρόσθετα, χρησιμοποιεί το SPARQL για εκτέλεση αναζητήσεων στα δεδομένα. Κάθε οντότητα του DBpedia συνδέεται απευθείας σε ένα άρθρο της Wikipedia και κάθε πόρος της περιγράφεται από μία ετικέτα, μικρή ή μεγάλη περιγραφή,

σύνδεσμο στηναντίστοιχη σελίδα αλλά και σύνδεσμο σε εικόνα που περιγράφει το αντικείμενο. Τέλος, η DBpedia Μπορεί να είναι προσβάσιμη online ,έσω του SPARQL αλλά και ως διασυνδεδεμένες δημοσιοποιημένες πληροφορίες.

Κεφάλαιο 5

Πειραματική Μελέτη

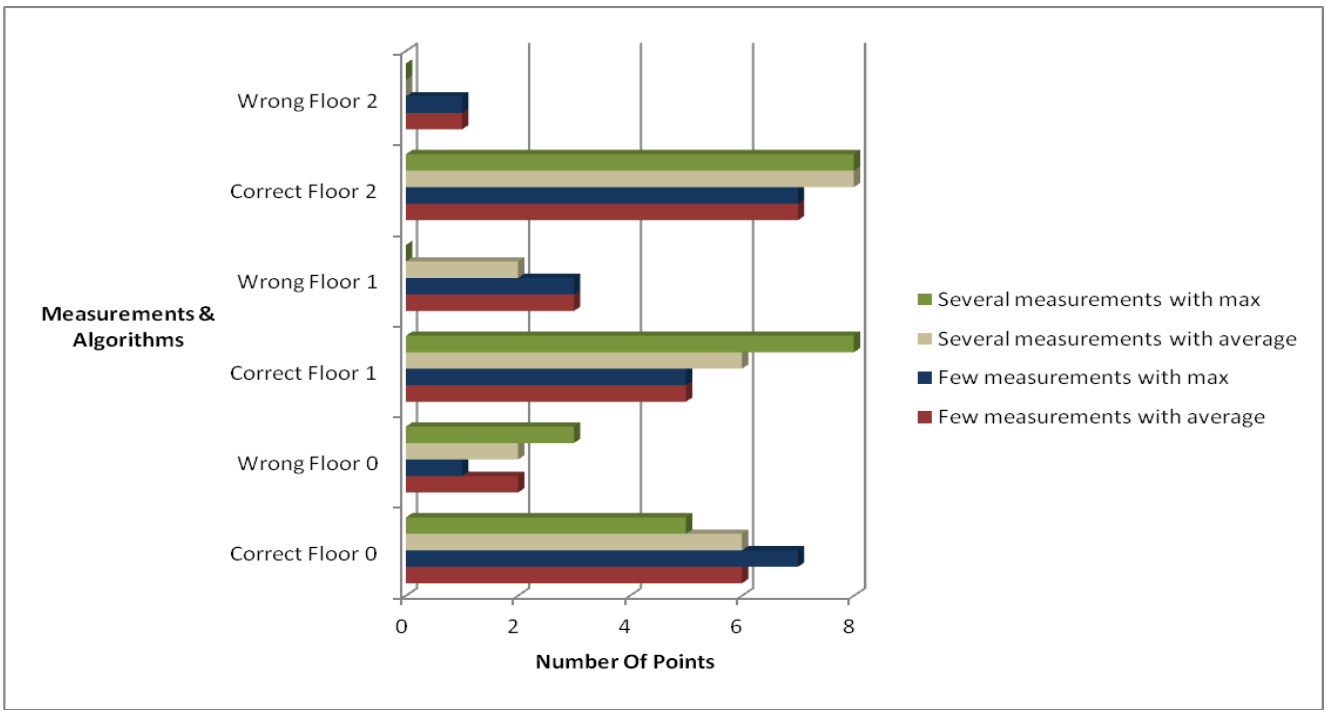
5.1 Ακρίβεια Γεωτοποθέτησης Ορόφων

5.1.1 Αποτελέσματα

Προκειμένου να διαπιστώσω ποιος από τους δύο αλγορίθμους δίνει καλύτερα αποτελέσματα, έλεγξα και τους δύο με τις ήδη υπάρχουσες μετρήσεις από τους τρεις ορόφους, που ήταν αρκετές, αλλά και με επιπλέον μετρήσεις που πήρα από κάθε ένα από τους 4 ορόφους, λιγότερες σε αριθμό ούτως ώστε να παρατηρήσω κατά πόσο είναι δυνατόν να εντοπίζει το σωστό όροφο με κάποιο από τους δύο αλγορίθμους με λιγότερες όμως μετρήσεις. Αρχικά δημιούργησα τα .floor αρχεία και για τις 4 περιπτώσεις υπό μελέτη και έλεγξα την ακρίβεια εμφάνισης του σωστού ορόφου σε 8 τυχαία σημεία σε κάθε ένα από τους 3 ορόφους.

Στην πιο πάνω γραφική παράσταση φαίνονται τα αποτελέσματα των 4 καταστάσεων υπό μελέτη, όπου στον άξονα χ βρίσκεται ο αριθμός των σημείων που χρησιμοποιήθηκαν για έλεγχο των διαφόρων καταστάσεων, δηλαδή των 8 τυχαίων σημείων σε κάθε όροφο. Στον άξονα ψ, φαίνεται ο αριθμός των σημείων στα οποία παρουσιάστηκε επιτυχία ή αποτυχία αντίστοιχα στην εύρεση του σωστού ορόφου, σε κάθε ένα από αυτούς.

Όπως φαίνεται και από την πιο πάνω γραφική παράσταση, τα καλύτερα αποτελέσματα δίνει ο αλγόριθμος που υπολογίζει τη μέγιστη τιμή σε συνδυασμό με τις μετρήσεις που είναι περισσότερες και αυτό γιατί από τα 8 τυχαία σημεία στα οποία έλεγξα τον αλγόριθμο, στους



Κεφάλαιο 5 - Εικόνα 5.1
Αποτελέσματα Πειραματικής Μελέτης

ορόφους 1 και 2 το ποσοστό επιτυχίας ήταν 100% και στον όροφο 0 η επιτυχία ήταν 62.5%. Ο αλγόριθμος με το μέσο όρο και όταν οι μετρήσεις είναι περισσότερες, είναι ο δεύτερος καλύτερος με μικρή διαφορά από τον αλγόριθμο με τη μέγιστη τιμή σε λιγότερες όμως μετρήσεις. Συγκεκριμένα, τα ποσοστά επιτυχίας του δεύτερου στη σειρά αλγορίθμου ήταν 75% στους ορόφους 0 και 1 ενώ στον όροφο 2 η επιτυχία έφτασε το 100%. Αντιθέτως, ο αλγόριθμος με τη μέγιστη τιμή σε λιγότερες μετρήσεις σημείωσε επιτυχία 87,5% στον όροφο 0 και 2 και 62,5% στον πρώτο όροφο. Λιγότερη ακρίβεια ως προς την εύρεση του σωστού ορόφου είχε ο αλγόριθμος με το μέσο όρο των rss values για λιγότερες μετρήσεις. Παρόλο που ο συγκεκριμένος αλγόριθμος είχε σχεδόν τα ίδια αποτελέσματα με τον αλγόριθμο με μέγιστο rss value και λιγότερες μετρήσεις, ο δεύτερος ήταν πιο ακριβής. Συγκεκριμένα, στα σημεία όπου εμφανιζόταν λανθασμένος όροφος, στην πρώτη περίπτωση είτε παρέμενε στο λανθασμένο όροφο είτε γινόταν συνεχής εναλλαγή μεταξύ διαφόρων ορόφων, μαζί και του σωστού. Αντιθέτως, στον αλγόριθμο με το μέγιστο και με λιγότερες μετρήσεις εάν εμφάνιζε λανθασμένο όροφο δεν παρέμενε σε αυτό αλλά αμέσως μετά εμφάνιζε τον σωστό και παρέμενε σε αυτόν. Ο αλγόριθμος με τη λιγότερη επιτυχία σημείωσε ποσοστά 75%, 62,5% και 87,5 στους ορόφους 0-2 αντίστοιχα. Ενώ σχετικά είναι καλά αποτελέσματα σε σχέση με το ποσοστό επιτυχίας των υπολοίπων δεν είναι ικανοποιητικά.

5.1.2 Συγκριτική αξιολόγηση και διαχείριση

Κεφάλαιο 5 - Εικόνα 5.2
SmartLab

Ο επαναπρογραμματισμός έξυπνων κινητών συσκευών και ο έλεγχος των εφαρμογών καθώς και η συλλογή δεδομένων είναι κουραστική και χρονοβόρα διαδικασία η οποία και δημιουργεί σημαντικές προκλήσεις. Για το σκοπό αυτό, χρησιμοποίησα το SmartLab, μία νέα ανοικτή υποδομή νεφέλης, ως υπηρεσία (IaaS) η οποία επιτρέπει λεπτό έλεγχο τόσο στις πραγματικές όσο και στις εικονικές έξυπνες κινητές συσκευές μέσω, μίας διαισθητικής διεπαφής βασισμένης στο διαδίκτυο. Η υπάρχουσα υποδομή του SmartLab είναι ιδανική για σενάρια που απαιτούν λεπτό και χαμηλού επιπέδου έλεγχο πάνω σε πραγματικές έξυπνες κινητές συσκευές π.χ. βάσεις δεδομένων και αποθήκευση, ασφάλεια, αλλά, και για σενάρια που απαιτούν τη δέσμευση φυσικών αισθητήρων και γεωτοποθεσιών. Το SmartLab,έχει χρησιμοποιηθεί κατά τη διαδικασία ανάπτυξης του πρωτοτύπου της Android εφαρμογής CLODA.

Το SmartLab διευκόλυνε την ανάπτυξη, τον έλεγχο και την επίδειξη της εφαρμογής CLODA σε σημαντικό βαθμό.

Αρχικά, επέτρεψε τη μετακίνηση στο κτίριο εντοπίζοντας την τοποθεσία του χρήστη ενώ παράλληλα προβαλλόταν η οθόνη της κινητής συσκευής σε ένα απομακρυσμένο φυλλομετρητή μέσω του SmartLab. Η συγκεκριμένη ρύθμιση έχει αποδειχτεί ιδιαίτερα χρήσιμη εφόσον η πλειονότητα των υφιστάμενων AndroidScreenCapture είναι βασισμένη στα usb και είναι επίσης αναποτελεσματική. Επιπλέον, το SmartLab, επέτρεψε τη συλλογή και σύγκριση RSS δεικτών από διαφορετικά WiFi σύνολα τσιπ, κάτι που είναι σημαντικό για

τις RSS μετρήσεις και δε θα ήταν δυνατό να γίνει με προσομοιωτές. Τέλος, το SmartLab επέτρεψε τον έλεγχο του παραγόμενου APK σε διάφορες συσκευές.

5.2 Δοκιμή εφαρμογής από χρήστες

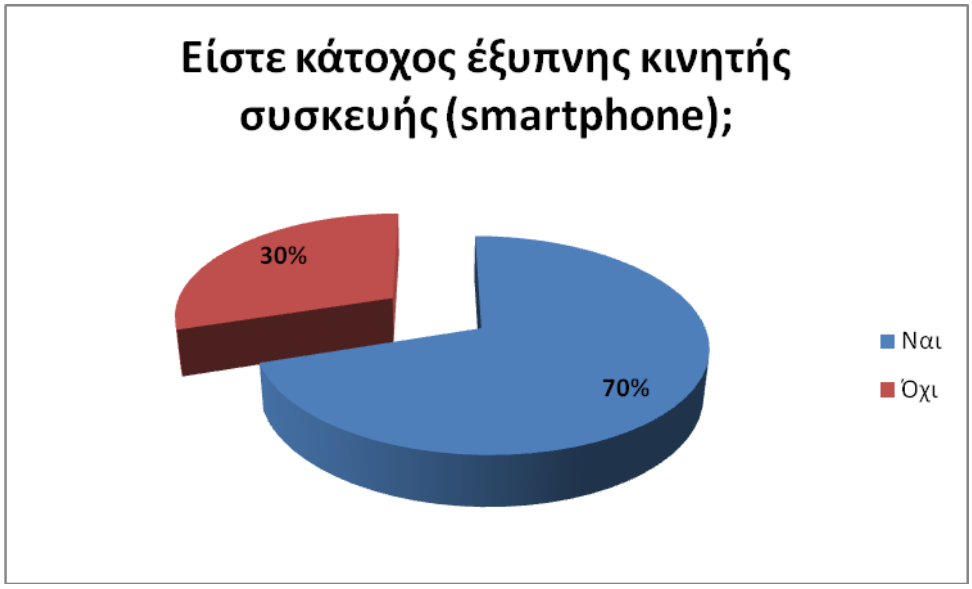
Μετά την υλοποίηση της εφαρμογής CLODA, κρίθηκε αναγκαίο να δοκιμαστεί από ένα σύνολο χρηστών προκειμένου να διαπιστωθεί κατά πόσο υπήρξε χρήσιμη και εάν ανταποκρινόταν στις απαιτήσεις των χρηστών εκείνη τη στιγμή. Μετά τη δοκιμή από τους χρήστες, δόθηκε ένα ερωτηματολόγιο στον κάθε ένα από αυτούς προκειμένου να ληφθούν κάποιες πληροφορίες σχετικά με το πώς ήταν η αλληλεπίδρασή τους με την εφαρμογή. Το ερωτηματολόγιο αποτελείται από 11 ερωτήσεις και δίνεται στο παράρτημα Β. Στη συνέχεια, ακολουθούν τα αποτελέσματα από αυτή τη πειραματική μελέτη.

5.2.1 Αποτελέσματα User Study

Για την πειραματική αυτή μελέτη χρησιμοποίησα ένα δείγμα από 30 άτομα εκ των οποίων οι 15 ήταν άντρες και οι 15 γυναίκες. Πιο κάτω φαίνονται τα αποτελέσματα από το ερωτηματολόγιο.

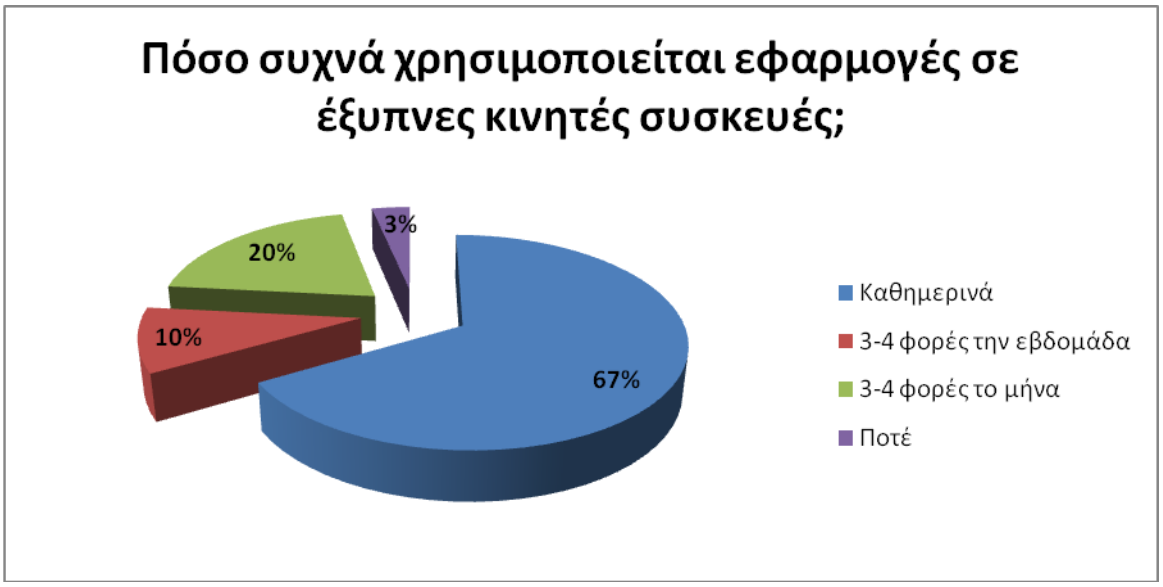
Κεφάλαιο 5 - Εικόνα 5.3
Ηλικία Συμμετεχόντων

Στην πιο πάνω εικόνα φαίνεται η ηλικία των ατόμων που συμμετείχαν στην πειραματική μελέτη. Όπως φαίνεται και από τη γραφική παράσταση είναι όλοι νέοι μεταξύ 19 και 25 ετών και είναι όλοι φοιτητές στο Πανεπιστήμιο Κύπρου.



Κεφάλαιο 5 - Εικόνα 5.4
Ερώτηση 1

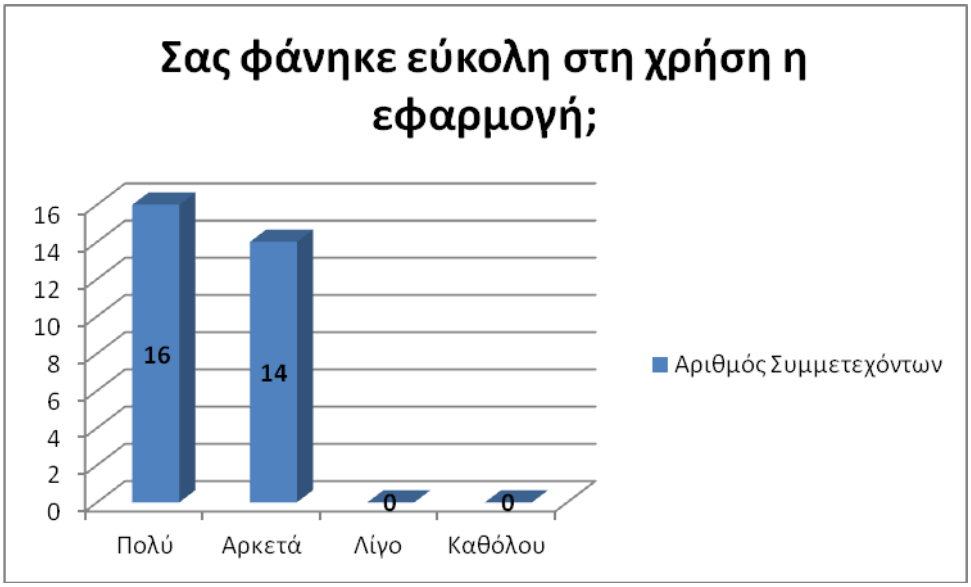
Στην πιο πάνω εικόνα φαίνονται τα ποσοστά όπως αυτά προέκυψαν από τις απαντήσεις των χρηστών στα ερωτηματολόγια, όσον αφορά στο κατά πόσο ή όχι είναι κάτοχοι έξυπνων κινητών συσκευών. Όπως μπορεί να διαπιστώσει κανείς, το 70%, δηλαδή 21 άτομα, δήλωσε πως κατέχει έξυπνη κινητή συσκευή ενώ, το υπόλοιπο 30% δεν έχει στην κατοχή του μια τέτοια συσκευή. Τα αποτελέσματα ήταν αναμενόμενα εφόσον στη σημερινή εποχή οι πλείστοι άνθρωποι έχουν στην κατοχή τους έξυπνες κινητές συσκευές.



Κεφάλαιο 5 - Εικόνα 5.5
Ερώτηση 2

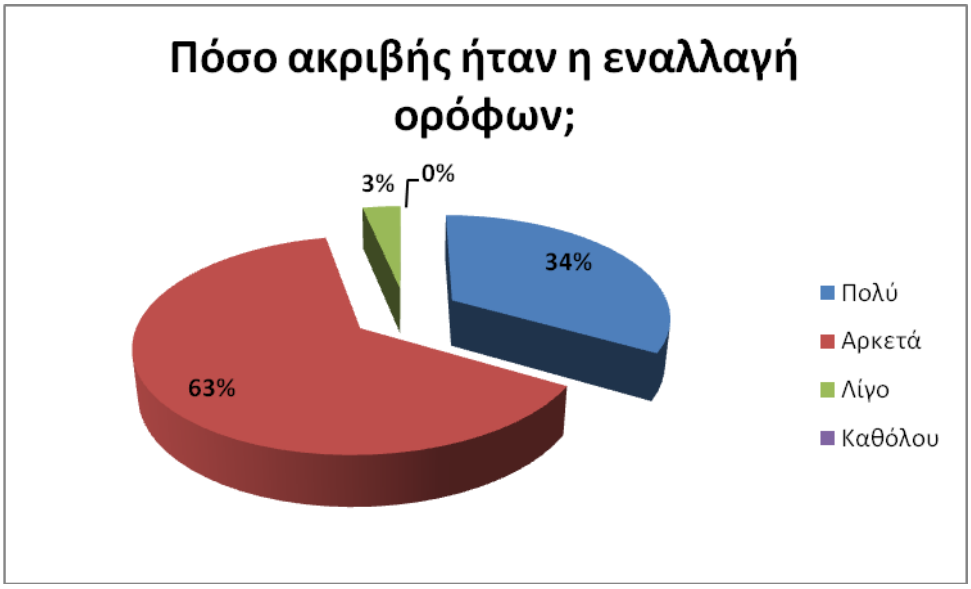
Η εικόνα 5.5 παρουσιάζει τα αποτελέσματα στην ερώτηση 2 του ερωτηματολογίου η οποία και αποσκοπούσε στο να διαπιστωθεί πόσο συχνά οι κάτοχοι έξυπνων κινητών συσκευών

χρησιμοποιούν τις διάφορες εφαρμογές που είτε είναι ήδη εγκατεστημένες στη συσκευή τους είτε, τις λαμβάνουν από την αγορά εφαρμογών. Συγκεκριμένα, η πλειονότητα δηλαδή, το 67% των χρηστών (20 άτομα) δήλωσε πως οι εφαρμογές αυτές αποτελούν αναπόσπαστο κομμάτι της καθημερινότητάς τους. Σημαντικό είναι το γεγονός ότι 20% (6 άτομα) των ερωτηθέντων δήλωσε πως η χρήση εφαρμογών ανέρχεται στις 3-4 φορές το μήνα, ποσοστό σχετικά ψηλό στις τις μέρες μας, όπου οι περισσότεροι ασχολούνται με έξυπνες κινητές συσκευές. Ακολουθεί ένα ποσοστό 10% το οποίο και χρησιμοποιεί εφαρμογές 3-4 φορές την εβδομάδα και τέλος, ένα ποσοστό 3% το οποίο δήλωσε πως δεν ασχολείται καθόλου με εφαρμογές.



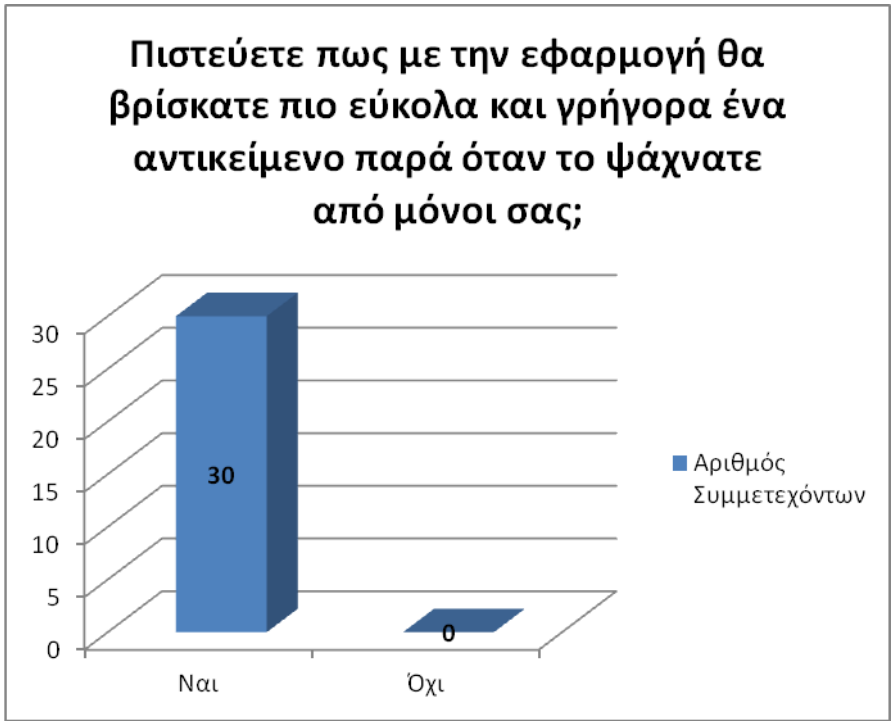
Κεφάλαιο 5 - Εικόνα 5.6
Ερώτηση 4

Στην ερώτηση 4 ζητήθηκε από τους χρήστες να απαντήσουν πόσο εύκολη φάνηκε σε αυτούς η χρήση της εφαρμογής. Όπως φαίνεται και από την εικόνα 5.6, 16 από τους 30 ερωτηθέντες δήλωσαν πως η εφαρμογή ήταν πολύ εύκολη για αυτούς να τη χρησιμοποιήσουν ενώ, 14 άτομα δήλωσαν πως η ευκολία στη χρήση ήταν αρκετή. Όλοι οι χρήστες δήλωσαν πως γενικά η εφαρμογή ήταν εύκολη στη χρήση και αυτό οδηγεί στο συμπέρασμα πως η διεπαφή και ο τρόπος υλοποίησης των διάφορων λειτουργιών ήταν όντως φιλικά προς το χρήστη.



Κεφάλαιο 5 - Εικόνα 5.7
Ερώτηση 5

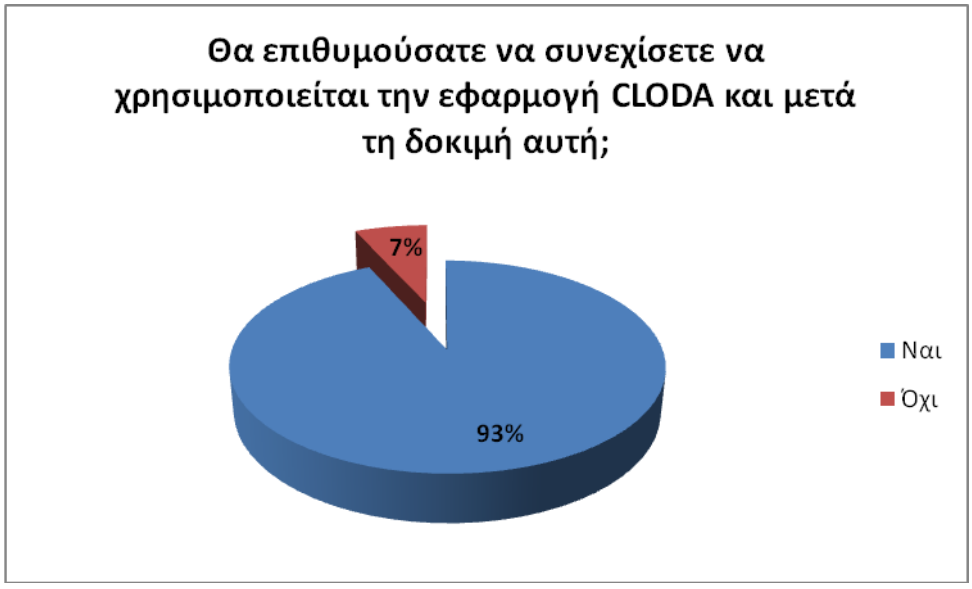
Η πιο πάνω εικόνα παρουσιάζει τα αποτελέσματα των χρηστών στην ερώτηση 5, κατά πόσο η εναλλαγή των ορόφων ήταν ακριβής. Όπως φαίνεται, το 63% (19 άτομα) είναι αρκετά ικανοποιημένο, ένα ποσοστό 34%(10 άτομα) ήταν πολύ ικανοποιημένο και, 3% υπήρξε λιγότερο ικανοποιημένο. Συμπέρασμα αυτού είναι πως ο αλγόριθμος για την εναλλαγή ορόφων θα πρέπει να βελτιωθεί ώστε να γίνει πιο ακριβής και σωστός εντοπισμός του κατάλληλου ορόφου.



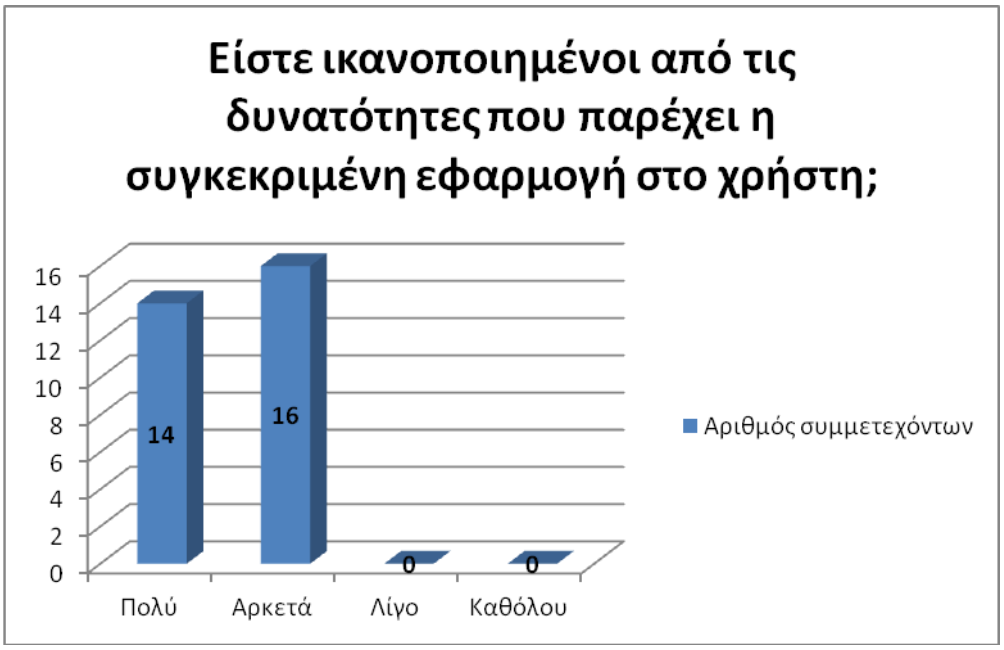
Κεφάλαιο 5 - Εικόνα 5.8
Ερώτηση 6

Η πιο πάνω εικόνα παρουσιάζει τι δήλωσαν οι χρήστες ως προς το κατά πόσο θα εύρισκαν πιο εύκολα κάποιο αντικείμενο χρησιμοποιώντας την εφαρμογή CLODA παρά από όταν το έψαχναν από μόνοι τους. Όλοι οι χρήστες ανέφεραν πως η εύρεση ενός αντικειμένου θα ήταν πιο εύκολη. Αυτό σημαίνει πως το γεγονός ότι τα διάφορα αντικείμενα είναι ήδη καταχωρημένα στο χώρο, διευκολύνει τους χρήστες στο να εντοπίσουν πιο εύκολα κάποιο αντικείμενο που ψάχνουν. Επιπλέον, αυτό δηλώνει πως υπάρχει ανάγκη για τέτοιου είδους εφαρμογές οι οποίες θα διευκολύνουν ολοένα και περισσότερο τους χρήστες σε καθημερινές τους δραστηριότητες.

Η πιο κάτω εικόνα (Εικόνα 5.9) δείχνει τα αποτελέσματα στην ερώτηση 8, κατά πόσο δηλαδή οι χρήστες θα ήθελαν να συνεχίσουν τη χρήση της εφαρμογής CLODA και μετά τη διεξαγωγή του πειράματος. Ένα ποσοστό 93% δήλωσε πως θα επιθυμούσε να συνεχίσει να χρησιμοποιεί την εφαρμογή ενώ, ένα ποσοστό 7% δήλωσε πως όχι. Μέσα από τα αποτελέσματα αυτής της ερώτησης φαίνεται πως οι χρήστες κρίνουν αναγκαία την ύπαρξη μίας τέτοιας εφαρμογής που θα τους βοηθά στην καθημερινότητά τους όσον αφορά στην εσωτερική τοποθέτηση αλλά και στον εντοπισμό αντικειμένων στο χώρο, εφόσον δεν υπάρχουν σήμερα αρκετές εφαρμογές που να συνδυάζουν αυτά τα δύο και ειδικά την εσωτερική τοποθέτηση η οποία είναι και ένα κομμάτι το οποίο μελετάται ακόμη.



Κεφάλαιο 5 - Εικόνα 5.9
Ερώτηση 8



Κεφάλαιο 5 - Εικόνα 5.10
Ερώτηση 11

Στην πιο πάνω εικόνα φαίνονται τα αποτελέσματα στην ερώτηση 11 κατά πόσο οι χρήστες ήταν ικανοποιημένοι από τις δυνατότητες που παρέχει η εφαρμογή CLODA. Οι 14 από τους 30 χρήστες δήλωσαν πολύ ικανοποιημένοι και οι υπόλοιποι 16 αρκετά ικανοποιημένοι. Αυτό σημαίνει πως εάν η εφαρμογή διατεθεί στο κοινό θα έχει απήχηση εφόσον ένα δείγμα ατόμων υπήρξε ικανοποιημένο από αυτή.

Μέσω του συγκεκριμένου ερωτηματολογίου διαπιστώθηκε πως οι περισσότεροι χρήστες (22 άτομα) γνώριζαν τη σημασία του όρου εσωτερική τοποθέτηση ενώ μόλις 8 άτομα δήλωσαν πως δε γνώριζαν τον όρο αυτό. Αυτό δείχνει πως παρόλο που οι περισσότεροι ήξεραν την έννοια του όρου αυτού, θα πρέπει να ενημερώνονται ολοένα και περισσότερα άτομα για τις νέες αυτές τεχνολογίες και τις δυνατότητες που μπορούν να παρέχουν στο χρήστη ώστε, να αναπτυχθούν περισσότερες εφαρμογές βασισμένες στην εσωτερική τοποθέτηση των χρηστών σε οποιοδήποτε χώρο.

Επιπρόσθετα, ένα άλλο συμπέρασμα που προκύπτει από αυτή την πειραματική μελέτη είναι, ότι η εφαρμογή αυτή φάνηκε χρήσιμη σε όλους τους χρήστες που τη δοκίμασαν καθώς και οι 30 συμμετέχοντες σε ερώτηση κατά πόσο πιστεύουν πως η εφαρμογή είναι χρήσιμη και γιατί, απάντησαν θετικά. Συγκεκριμένα, δήλωσαν πως θα μπορούσε κάποιος να εντοπίσει εύκολα συσκευές μέσα στο κτίριο της Πληροφορικής, όπως για παράδειγμα εκτυπωτές. Μερικοί δήλωσαν πως με τη βοήθεια της εφαρμογής αυτής θα μπορούσαν να εντοπίζουν

εύκολα και γρήγορα οποιοδήποτε αντικείμενο αλλά και να γνωρίζουν τι ακριβώς υπάρχει σε κάθε αίθουσα. Ακόμη, κάποιοι δήλωσαν πως είναι χρήσιμη η εφαρμογή εφόσον το κτίριο είναι αρκετά μεγάλο και κάθε χώρος περιέχει πολλά αντικείμενα. Υπήρξαν και μερικοί οι οποίοι δήλωσαν πως είναι χρήσιμη για πρωτοετής και Erasmus φοιτητές οι οποίοι δεν γνωρίζουν καλά τις εγκαταστάσεις και θα τους διευκόλυνε πολύ στο να μην σπαταλούν το χρόνο τους ψάχνοντας αίθουσες διδασκαλίας και αντικείμενα.

Μία ακόμη ερώτηση που έγινε στους συμμετέχοντες ήταν, σε ποιους άλλους τομείς θα ήταν χρήσιμη μία τέτοια εφαρμογή. Ορισμένοι δήλωσαν πως θα ήταν χρήσιμη στο στρατό και συγκεκριμένα στις αποθήκες με τα διάφορα πυρομαχικά και όπλα όπου θα ήταν καλό να είναι όλα καταγεγραμμένα και αυτοί που πρέπει, να γνωρίζουν που βρίσκονται ώστε εάν χρειαστεί να μπορέσουν άμεσα να τα χρησιμοποιήσουν. Μία άλλη πρόταση ήταν και ο τομέας του εμπορίου, όπου δηλαδή κάποιος θα μπορούσε να καταγράψει τις τιμές σε κάποια καταστήματα ώστε οι χρήστες να μπορούν να συγκρίνουν. Τέλος, υπήρξε και μία πρόταση η εφαρμογή να χρησιμοποιηθεί για προσωπικούς σκοπούς δηλαδή στο σπίτι.

Μία τελευταία και ενδιαφέρουσα ερώτηση ήταν κατά πόσο θα μπορούσε η εφαρμογή να χρησιμοποιηθεί και σε άλλους δημόσιους χώρους, εκτός από το Πανεπιστήμιο Κύπρου. Και εδώ οι απαντήσεις ήταν πολύ θετικές καθώς και οι 30 συμμετέχοντες δήλωσαν πως ναι μπορεί να επεκταθεί και πρότειναν μερικούς χώρους. Ένας από αυτούς είναι τα εμπορικά κέντρα και οι υπεραγορές για τον εντοπισμό καταστημάτων αλλά και προϊόντων μέσα σε αυτά, εύκολα και γρήγορα χωρίς οποιαδήποτε ταλαιπωρία. Άλλος χώρος ήταν και οι δημόσιες βιβλιοθήκες αλλά και τα νοσοκομεία, στα οποία θα μπορούσε όλος ο εξοπλισμός να είναι καταγεγραμμένος σε μία εφαρμογή και όταν κάποιος ιατρός ή άλλο προσωπικό χρειαστεί κάτι σε άμεση ανάγκη, να μπορεί να το εντοπίσει γρήγορα και να δράσει άμεσα και αποτελεσματικά.

Στο τέλος του ερωτηματολογίου ζητήθηκε από τους χρήστες να δώσουν οποιεσδήποτε εισηγήσεις ή και σχόλια σχετικά με την εφαρμογή. Μέσω αυτού, έχουν προκύψει μερικά σημαντικά στοιχεία. Έχει προταθεί η επέκταση της εφαρμογής ώστε να μπορεί να βοηθήσει και άτομα με προβλήματα όρασης, με την εγκατάσταση φωνητικής αναγνώρισης στην εφαρμογή. Επίσης, κάποιος δήλωσε πως θα επιθυμούσε πιο γρήγορο τρόπο αναζήτησης των διάφορων αντικειμένων, ενώ κάποιος άλλος εισηγήθηκε την προσομοίωση σε 3D τεχνολογία και τη εισαγωγή επαυξημένης πραγματικότητας (augmented reality). Μία άλλη πρόταση ήταν και η επέκταση της εφαρμογής και στα υπόλοιπα κτίρια του Πανεπιστημίου Κύπρου αλλά και η δυνατότητα αποθήκευσης αντικειμένων που δεν έχουν γραμμικό κώδικα. Μία τελευταία πρόταση ήταν και η κατηγοριοποίηση των αντικειμένων με διαφορετικά χρώματα ώστε να μπορεί να γίνεται πιο εύκολο φιλτράρισμα και εντοπισμός των αντικειμένων.

Κεφάλαιο 6

Η εφαρμογή CLODA έχει δημιουργηθεί με μία πρωτοποριακή αρχιτεκτονική για πληθωποριστικές διασυνδεδεμένες δημοσιοποιημένες πληροφορίες (LOD). Η αρχιτεκτονική αυτή, χρησιμοποιεί πλεονεκτήματα καθενός από τα κομμάτια από τα οποία και αποτελείται π.χ. Εσωτερική Τοποθέτηση, Πληθοπορισμός σε κινητές συσκευές, προκειμένου να παρέχει ένα βιώσιμο πλαίσιο για μία συλλογή διασυνδεδεμένων δημοσιοποιημένων πληροφοριών (LOD). Τα κύρια πλεονεκτήματα της εφαρμογής CLODA είναι πρώτα από όλα η ικανότητα να λειτουργεί σε εσωτερικούς χώρους, πάνω από υπάρχουσες γεωγραφικά χαρτογραφημένες τεχνολογίες (π.χ. Google Maps). Ένα ακόμη πλεονέκτημα της εφαρμογής αυτής είναι η κατανάλωση δημόσια διαθέσιμων API (π.χ. Google Shopping API). Τέλος, η χρήση της δύναμης του πλήθους αποτελεί μεγάλο πλεονέκτημα έτσι ώστε να κατασκευαστούν επικυρωμένες διασυνδεδεμένες δημοσιοποιημένες πληροφορίες (LOD) με γνώση γεωτοποθέτησης.

Ως μελλοντικά σχέδια, θα μπορούσαν να διερευνηθούν τα πλεονεκτήματα της εισαγωγής κινήτρων για τους χρήστες έτσι ώστε να ενισχυθεί η συνεργασία μεταξύ αυτών και να ενθαρρυνθούν περισσότερα άτομα ώστε να συνεισφέρουν στις δημοσίως διαθέσιμες διασυνδεδεμένες δημοσιοποιημένες πληροφορίες (LOD), παρέχοντας ταυτόχρονα εγγυήσεις για την εγκυρότητα των δεδομένων που έχουν συλλεχθεί. Επιπλέον, θα μπορούσε να επεκταθεί η συγκεκριμένη εφαρμογή σε περισσότερους τομείς (π.χ. εμπόριο, εκπαιδευτικό τομέα, στρατιωτικό κ.λ.π.) αφού όπως φάνηκε και από την πειραματική μελέτη θα είχε απήχηση στο κοινό.

Βιβλιογραφία

[1] T. Berners-Lee, J. Hendler, and O. Lassila, “The Semantic Web,” *In Scientific American*, pp. 29–37.

[2] C. Bizer, T. Heath, and T. Berners-Lee, “Linked Data - The Story So Far,” *Int. J. Semantic Web Inf. Syst.*, vol. 5, no. 3, pp. 1–22, 2009.

[3] G. Chatzimilioudis, A. Konstantinidis, C. Laoudias, and D. Zeinalipour-Yazti, “Crowdsourcing with Smartphones,” *IEEE Internet Computing*, vol. 16, no. 5, pp. 36–44, Sep. 2012.

[4] “Source: Strategy Analytics.” [Online]. Available: http://www.strategyanalytics.com.

[5] C. Laoudias, G. Constantinou, M. Constantinides, S. Nicolaou, D. Zeinalipour-Yazti, and C. G. Panayiotou, “The Airplace Indoor Positioning Platform for Android Smartphones,” *2012 IEEE 13th International Conference on Mobile Data Management*, pp. 312–315, Jul. 2012.

[6] A. Konstantinidis and C. Costa, “Demo : A Programming Cloud of Smartphones,” p. 4503, 2012.

[7] Y. Gu, A. Lo, and I. Niemegeers, “A survey of indoor positioning systems for wireless personal networks,” *In IEEE Communications Surveys & Tutorials 09*.

[8] C.-L. Li, C. Laoudias, G. Larkou, G. Chatzimilioudis, D. Zeinalipour-Yazti, and C. . Panayiotou, “Hybrid Indoor Positioning on Multi-Sensor Android Smartphones,” *In IPIN `12*.

[9] J. Ledlie, B. Odero, E. Minkov, I. Kiss, and J. Polifroni, “Crowd translator: on building localized speech recognizers through micropayments,” *In ACM SIGOPD Operating Systems Review*, 2010.

[10] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson, “VTrack: accurate, energy-aware road traffic delay estimation using mobile phones,” *In ACM SenSys`09*.

[11] R. . Rana, C.-T. Chou, S. Kanhere, N. Bulusu, and W. Hu, “Ear-phone: an end-to-end participatory urban noise mapping system,” *In IPSN`10*.

[12] M. Stevens and E. . Hondt, “Crowdsourcing of Pollution Data using Smartphones,” *In UbiComp`10*.

[13] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan, “The pothole patrol: using a mobile sensor network for road surface monitoring,” *In MobiSys’08*.

[14] S. Matyas, C. Matyas, C. Schlieder, P. Kiefer, H. Mitarai, and M. Kamata, “Designing location-based mobile games with a purpose: collecting geospatial data with CityExplorer,” *In International Conference on Advances in Computer Entertainment Technology*, 2008.

[15] E. Koukoumidis, L. . Peh, and M. R. Martonosi, “SignalGuru: leveraging mobile phones for collaborative traffic signal schedule advisory,” *In MobiSys’11*.

[16] A. Gupta, W. Thies, E. Cutrell, and R. Balakrishnan, “mClerk: enabling mobile crowdsourcing in developing regions,” *In CHI’12*.

[17] T. Yan, M. Marzilli, R. Holmes, D. Ganesan, and M. Corner, “mCrowd: a platform for mobile crowdsourcing,” *In SenSys’09*.

[18] N. Eagle, “txteagle: Mobile Crowdsourcing,” *In IDGD’09*.

[19] D. Le-Phuoc, J. X. Parreira, M. Hausenblas, Y. Han, and M. Hauswirth, “Live linked open sensor database,” *In I-SEMANTICS’10*.

[20] J. Kopecky and J. Domingue, “ParkJam: Crowdsourcing Parking Availability Information with Linked Data,” *In ESWC’12*.

[21] E. Simperl, B. Norton, and D. Vrandecic, “Crowdsourcing Tasks within Linked Data Management,” *In ESWC’12*.

[22] S. Ceri, E. Valle, D. Pedreschi, and R. Trasarti, “Mega-modeling for Big Data Analytics,” *In Conceptual Modeling*, 2012.

[23] Y. Cheng, C. Qin, and F. Rusu, “GLADE: big data analytics made easy,” *In SIGMOD’12*.

[24] S. Chaudhuri, “What next?: a half-dozen data management research goals for big data and the cloud,” *In PODS’12*.

[25] C. Bizer, J. Lehmann, G. Kobilarov, C. Becker, R. Cyganiak, and S. Hellman, “A Crystalization Point for the Web of Data,” *In WWW`09*.

[26] “Source:” [Online]. Available: http://forum.xda-developers.com/wiki/Android.

[27] “Source:” [Online]. Available: http://www.android.com/about/.

[28] “Source:” [Online]. Available: https://developers.google.com/maps/documentation/javascript/ .

[29] “Source: ThannMap.” [Online]. Available: thannmap.appspot.com.

[30] “Source:” [Online]. Available: https://code.google.com/p/zxing/.

[31] “Source:” [Online]. Available: http://www.webopedia.com/TERM/W/Web_2_point_0.html.

[32] “Source:” [Online]. Available: https://developers.google.com/shopping-search/.

[33] “Source:” [Online]. Available: http://www.json.org/.

[34] “Source:” [Online]. Available: http://couchdb.apache.org/.

[35] “Source:” [Online]. Available: http://wiki.apache.org/couchdb/Introduction.

[36] “Source:” [Online]. Available: http://www.saggingcouch.com/.

[37] “Source:” [Online]. Available: http://guide.couchdb.org/draft/documents.html.

[38] “Source:” [Online]. Available: http://guide.couchdb.org/editions/1/en/views.html.

[39] “Source:” [Online]. Available: http://wiki.apache.org/couchdb/Introduction_to_CouchDB_views.

[40] “Source:” [Online]. Available: http://guide.couchdb.org/editions/1/en/design.html.

[41] “Source:” [Online]. Available: http://www.rdfabout.com/intro/#Why we need a new standard for the Semantic Web.

[42] “Source:” [Online]. Available: http://www.w3.org/TR/2004/REC-rdf-primer-20040210/#rdfmodel.

[43] “Source:” [Online]. Available: https://github.com/agrueneberg/Sessel.

[44] “Source:” [Online]. Available: http://www.w3.org/TR/rdf-sparql-protocol/#intro.

[45] “Source:” [Online]. Available: http://www.w3.org/TR/rdf-sparql-query/.

[46] “Source:” [Online]. Available: http://dbpedia.org/About.

Παράρτημα Α

Ερωτηματολόγιο

Ηλικία Φύλο (Α/Θ)

☐ ☐

Ερωτήσεις

1. Είστε κάτοχος έξυπνης κινητής συσκευής (smartphone);

   - Ναι
   - Όχι

2. Πόσο συχνά χρησιμοποιείται εφαρμογές για έξυπνες κινητές συσκευές;

   - Καθημερινά
   - 3-4 φορές την εβδομάδα
   - 3-4 φορές το μήνα
   - Ποτέ

3. Γνωρίζετε τι σημαίνει ο όρος εσωτερική τοποθέτηση (Indoor positioning);

   - Ναι
   - Όχι

4. Σας φάνηκε εύκολη στη χρήση η εφαρμογή;

   - Πολύ
   - Αρκετά
   - Λίγο
   - Καθόλου

5. Πόσο ακριβής ήταν η εναλλαγή ορόφων;

   - Πολύ
   - Αρκετά
   - Λίγο
   - Καθόλου

6. Πιστεύετε πως με την εφαρμογή θα βρίσκατε πιο εύκολα και γρήγορα ένα αντικείμενο παρά όταν το ψάχνατε από μόνοι σας;

   - ○ Ναι
   - ○ Όχι

7. Πιστεύετε πως η εφαρμογή αυτή είναι χρήσιμη και γιατί;

   - ○ Ναι
   - ○ Όχι

   Σχόλια

8. Θα επιθυμούσατε να συνεχίσετε να χρησιμοποιείτε την εφαρμογή CLODA και μετά τη δοκιμή αυτή;

   - ○ Ναι
   - ○ Όχι

9. Σε ποιους άλλους τομείς πιστεύετε θα ήταν χρήσιμη μία τέτοια εφαρμογή;

10. Πιστεύετε πως η εφαρμογή θα ήταν χρήσιμη εάν επεκτεινόταν και σε άλλους δημόσιους χώρους εκτός του Πανεπιστημίου Κύπρου (π.χ. υπεραγορές, εμπορικά κέντρα κ.λ.π.) και γιατί;

    - Ναι
    - Όχι

    Σχόλια

11. Είστε ικανοποιημένοι από τις δυνατότητες που παρέχει η συγκεκριμένη εφαρμογή στο χρήστη;

    - Πολύ
    - Αρκετά
    - Λίγο
    - Καθόλου

    Εισηγήσεις/ Γενικά Σχόλια

Παράρτημα Β

Κώδικας

Πιο κάτω ακολουθεί ο κώδικας της εφαρμογής

```
1: <resources>
2:
3:     <style name="AppTheme" parent="android:Theme.Light" />
4:
5: </resources>
```

```
    1: <?xml version="1.0" encoding="utf-8"?>
    2: <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    3:     android:id="@+id/helpwindowLogger"
    4:     android:layout_width="match_parent"
    5:     android:layout_height="match_parent"
    6:     android:orientation="vertical" >
    7:
    8:     <TextView
    9:         android:id="@+id/tHelpLogger"
   10:         android:layout_width="match_parent"
   11:         android:layout_height="match_parent"
   12:         android:text="\n\n>Adjust a floorplan of one of the four floors (-1,0,
1,2).
   13:         \n\n>Choose the samples number and sample interval from menu -> prefer
ences.
   14:         \n\n>Choose the folder to store RSS log file and the name of the file
with the RSS log records again from menu->preferences.
   15:         \n\n>Finally, touch on the screen and press save button in order to sa
ve the measurements. Take measurements in all four directions (0,90,180,270 degrees).
   16:         \n\n" />
   17:
   18: </LinearLayout>
```

```
    1: <?xml version="1.0" encoding="utf-8"?>
    2: <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    3:     android:id="@+id/helpwindowItemMenu"
    4:     android:layout_width="match_parent"
    5:     android:layout_height="match_parent"
    6:     android:orientation="vertical" >
    7:
    8:     <TextView
    9:         android:id="@+id/tHelpOptionsItem"
   10:         android:layout_width="match_parent"
   11:         android:layout_height="match_parent"
   12:         android:text=">EDIT ITEM\n\nPress the edit icon in order to be able to
 edit the attributes of the item,all but the location.\nWhen you edit the desired attr
ibute press save.
   13:         \n\n>DELETE ITEM\n\nPress the delete icon in order to permanently dele
te the selected item.\n\n>MOVE ITEM\n\nPress the move icon icon in order to change the
 location of the item.\nWhen the move icon is pressed the map with the computer scienc
e building appears.\nSelect the new position of the item by touching on the desired lo
cation.\nFinally, the item is moved to the new location.
   14:         \n\n" />
   15:
   16: </LinearLayout>
```

```
    1: <?xml version="1.0" encoding="utf-8"?>
    2: <menu xmlns:android="http://schemas.android.com/apk/res/android" >
    3:
    4:     <!-- <item android:id="@+id/DownloadRadioMap" android:icon="@drawable/down
load" android:title="Download RadioMap"></item> -->
    5:     <item
    6:         android:id="@+id/logger"
    7:         android:icon="@drawable/location2"
    8:         android:title="@string/menu_logger">
    9:     </item>
   10:     <item
   11:         android:id="@+id/Prefs"
   12:         android:icon="@android:drawable/ic_menu_preferences"
   13:         android:title="@string/menu_preferences">
   14:     </item>
   15:     <item
   16:         android:id="@+id/Help"
   17:         android:icon="@android:drawable/ic_menu_help"
   18:         android:title="@string/menu_help">
   19:     </item>
   20:     <item
   21:         android:id="@+id/About"
   22:         android:icon="@android:drawable/ic_menu_info_details"
   23:         android:title="@string/menu_about">
   24:     </item>
   25:     <item
   26:         android:id="@+id/Exit"
   27:         android:icon="@android:drawable/ic_menu_close_clear_cancel"
   28:         android:title="@string/menu_exit">
   29:     </item>
   30:     <item
   31:         android:id="@+id/FloorPlinEna"
   32:         android:title="@string/FloorPlinEna">
   33:     </item>
   34:     <item
   35:         android:id="@+id/FloorZero"
   36:         android:title="@string/FloorZero">
   37:     </item>
   38:     <item
   39:         android:id="@+id/FloorOne"
   40:         android:title="@string/FloorOne">
   41:     </item>
   42:     <item
   43:         android:id="@+id/FloorTwo"
   44:         android:title="@string/FloorTwo">
   45:     </item>
   46:
   47: </menu>
```

```
    1: <?xml version="1.0" encoding="utf-8"?>
    2: <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    3:     android:id="@+id/helpwindow"
    4:     android:layout_width="match_parent"
    5:     android:layout_height="match_parent"
    6:     android:orientation="vertical" >
    7:
    8:     <TextView
    9:         android:id="@+id/tHelp"
   10:         android:layout_width="match_parent"
   11:         android:layout_height="match_parent"
   12:         android:text=">FIND ME\nThe find me icon shows your current location i
n the building.
   13:         \n\n>TRACK ME\nThe track me icon displays your location while walking
in the building.
   14:         \n\n>SCAN ITEM\nThe scan item icon allows you to scan the barcode of a
n item.\nIt also allows you to manally enter a barcode through the application‘s menu.
\nWhen a barcode is scanned succesfully you will hear a beep sound.
   15:         \n\n>HIDE/SHOW FLOORPLAN/ITEMS\nThe hide/show floorplan/items icon giv
es you the ability to hide or show the floorplans and items-markers accordingly.
   16:         \n\n>PREFERENCES\nSelect the radiomap file you wish to import for trac
king.\nEnable floor mode which will change the floorpan to the current floor.
   17:         \n\n" />
   18:
   19: </LinearLayout>
```

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="300dp"
    android:layout_height="wrap_content"
    android:orientation="vertical" >

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginRight="7dp"
        android:orientation="vertical" >

        <TextView
            android:id="@+id/title"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Title" />

        <EditText
            android:id="@+id/etTitle"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:inputType="text" >
        </EditText>

        <TextView
            android:id="@+id/Brand"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Brand" />

        <EditText
            android:id="@+id/etBrand"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:inputType="text" >
        </EditText>

        <TextView
            android:id="@+id/Price"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Price" />

        <EditText
            android:id="@+id/etPrice"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:inputType="phone|numberDecimal" >
        </EditText>

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal"
            android:weightSum="100" >

            <Button
                android:id="@+id/ok"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="50"
                android:text="OK" />

            <Button
                android:id="@+id/cancel"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="50"
                android:text="Cancel" />
        </LinearLayout>
    </LinearLayout>

</LinearLayout>
```

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <RelativeLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:gravity="fill" >

        <TextView
            android:id="@+id/cwd"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentLeft="true"
            android:layout_marginRight="5dp"
            android:text="Current Location:" />

        <ScrollView
            android:layout_width="fill_parent"
            android:layout_height="50dip"
            android:layout_toRightOf="@id/cwd" >

            <TextView
                android:id="@+id/pwd"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content" />
        </ScrollView>
    </RelativeLayout>

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

        <ListView
            android:id="@android:id/list"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:layout_above="@+id/select_file_folder" />

        <Button
            android:id="@+id/select_file_folder"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_alignParentBottom="true" />
    </RelativeLayout>

</LinearLayout>
```

```
 1: <?xml version="1.0" encoding="UTF-8"?>
 2: <!--
 3:  Copyright (C) 2008 ZXing authors
 4:
 5:  Licensed under the Apache License, Version 2.0 (the "License");
 6:  you may not use this file except in compliance with the License.
 7:  You may obtain a copy of the License at
 8:
 9:       http://www.apache.org/licenses/LICENSE-2.0
10:
11:  Unless required by applicable law or agreed to in writing, software
12:  distributed under the License is distributed on an "AS IS" BASIS,
13:  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
14:  See the License for the specific language governing permissions and
15:  limitations under the License.
16: -->
17: <resources>
18:
19:     <item name="decode" type="id"/>
20:     <item name="decode_failed" type="id"/>
21:     <item name="decode_succeeded" type="id"/>
22:     <item name="launch_product_query" type="id"/>
23:     <item name="quit" type="id"/>
24:     <item name="restart_preview" type="id"/>
25:     <item name="return_scan_result" type="id"/>
26:     <item name="search_book_contents_failed" type="id"/>
27:     <item name="search_book_contents_succeeded" type="id"/>
28:
29: </resources>
```

```
<?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:minHeight="60dp"
    android:orientation="vertical"
    android:weightSum="100" >

    <ScrollView
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent" >

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_weight="20"
            android:orientation="horizontal"
            android:weightSum="100" >

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:layout_weight="50"
                android:orientation="vertical" >

                <TextView
                    android:id="@+id/tBarcode"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_marginLeft="16dp"
                    android:layout_marginTop="5dp"
                    android:text="Barcode: "
                    android:textAppearance="?android:attr/textAppearanceLarge"
 />

                <EditText
                    android:id="@+id/eBarcode"
                    android:layout_width="fill_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginLeft="16dp"
                    android:layout_marginRight="16dp"
                    android:layout_marginTop="5dp"
                    android:textAppearance="?android:attr/textAppearanceMedium
"
                    android:textColor="@color/lightblue" />

                <TextView
                    android:id="@+id/tLocation"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_marginLeft="16dp"
                    android:layout_marginTop="5dp"
                    android:text="Location: "
                    android:textAppearance="?android:attr/textAppearanceLarge"
 />

                <EditText
                    android:id="@+id/eLocation"
                    android:layout_width="fill_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginLeft="16dp"
                    android:layout_marginRight="16dp"
                    android:editable="true"
                    android:textAppearance="?android:attr/textAppearanceMedium
"
                    android:textColor="@color/lightblue" >
                </EditText>

                <TextView
                    android:id="@+id/tPrice"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_marginLeft="16dp"
                    android:layout_marginTop="5dp"
                    android:text="Price: "
                    android:textAppearance="?android:attr/textAppearanceLarge"
 />

                <EditText
                    android:id="@+id/ePrice"
                    android:layout_width="fill_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginLeft="16dp"
                    android:layout_marginRight="16dp"
                    android:editable="true"
                    android:textAppearance="?android:attr/textAppearanceMedium
"
                    android:textColor="@color/lightblue" />

                <TextView
                    android:id="@+id/tBrand"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_marginLeft="16dp"
                    android:layout_marginTop="5dp"
                    android:text="Brand: "
                    android:textAppearance="?android:attr/textAppearanceLarge"
 />

                <EditText
                    android:id="@+id/eBrand"
                    android:layout_width="fill_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginLeft="16dp"
                    android:layout_marginRight="16dp"
                    android:editable="true"
                    android:textAppearance="?android:attr/textAppearanceMedium
"
                    android:textColor="@color/lightblue" />

                <TextView
                    android:id="@+id/tTitle"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_marginLeft="16dp"
                    android:layout_marginTop="5dp"
                    android:text="Title: "
                    android:textAppearance="?android:attr/textAppearanceLarge"
 />

                <EditText
```

```
  114:                     android:id="@+id/eTitle"
  115:                     android:layout_width="fill_parent"
  116:                     android:layout_height="wrap_content"
  117:                     android:layout_marginLeft="16dp"
  118:                     android:layout_marginRight="16dp"
  119:                     android:editable="true"
  120:                     android:textAppearance="?android:attr/textAppearanceMedium
"
  121:                     android:textColor="@color/lightblue" />
  122:
  123:                 <TextView
  124:                     android:id="@+id/tFloor"
  125:                     android:layout_width="wrap_content"
  126:                     android:layout_height="wrap_content"
  127:                     android:layout_marginLeft="16dp"
  128:                     android:layout_marginTop="5dp"
  129:                     android:text="Floor: "
  130:                     android:textAppearance="?android:attr/textAppearanceLarge"
 />
  131:
  132:                 <EditText
  133:                     android:id="@+id/eFloor"
  134:                     android:layout_width="fill_parent"
  135:                     android:layout_height="wrap_content"
  136:                     android:layout_marginLeft="16dp"
  137:                     android:layout_marginRight="16dp"
  138:                     android:editable="true"
  139:                     android:textAppearance="?android:attr/textAppearanceMedium
"
  140:                     android:textColor="@color/lightblue" />
  141:             </LinearLayout>
  142:         </LinearLayout>
  143:     </ScrollView>
  144:
  145: </LinearLayout>
```

```
 1: <?xml version="1.0" encoding="utf-8"?>
 2: <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
 3:     android:layout_width="fill_parent"
 4:     android:layout_height="fill_parent" >
 5:
 6:     <WebView
 7:         android:id="@+id/map"
 8:         android:layout_width="match_parent"
 9:         android:layout_height="match_parent"
10:         android:layout_below="@+id/linearLayout1" />
11:
12:     <LinearLayout
13:         android:id="@+id/linearLayout4"
14:         android:layout_width="wrap_content"
15:         android:layout_height="wrap_content"
16:         android:layout_alignParentBottom="true"
17:         android:layout_alignParentRight="true" >
18:     </LinearLayout>
19:
20:     <LinearLayout
21:         android:id="@+id/linearLayout3"
22:         android:layout_width="wrap_content"
23:         android:layout_height="wrap_content"
24:         android:layout_above="@+id/linearLayout4"
25:         android:layout_alignParentRight="true"
26:         android:orientation="vertical" >
27:
28:         <TextView
29:             android:id="@+id/detectedAPs"
30:             android:layout_width="wrap_content"
31:             android:layout_height="wrap_content"
32:             android:layout_alignParentLeft="true"
33:             android:layout_marginTop="21dp"
34:             android:background="#80FFFFFF"
35:             android:text="AP: "
36:             android:textColor="#353535"
37:             android:textSize="12.5dip"
38:             android:textStyle="bold" />
39:
40:         <TextView
41:             android:id="@+id/trackingInfoLabel"
42:             android:layout_width="140dip"
43:             android:layout_height="fill_parent"
44:             android:layout_weight="0.0"
45:             android:background="#80FFFFFF"
46:             android:maxLines="5"
47:             android:paddingLeft="2dip"
48:             android:singleLine="false"
49:             android:text="Heading: \nX:\nY:"
50:             android:textColor="#353535"
51:             android:textSize="12.5dip"
52:             android:textStyle="bold" />
53:
54:         <TextView
55:             android:id="@+id/tFloor"
56:             android:layout_width="wrap_content"
57:             android:layout_height="wrap_content"
58:             android:background="#80FFFFFF"
59:             android:textColor="#353535"
60:             android:textSize="12.5dip"
61:             android:textStyle="bold" />
62:     </LinearLayout>
63:
64: </RelativeLayout>
```

```
 1: <?xml version="1.0" encoding="utf-8"?>
 2: <PreferenceScreen xmlns:android="http://schemas.android.com/apk/res/android"
 3:     android:key="pref_screen_key" >
 4:
 5:     <PreferenceCategory android:title="@string/radiomap_file_settings" >
 6:         <Preference
 7:             android:key="radiomap_file"
 8:             android:summary="@string/radiomap_file_summary"
 9:             android:title="@string/radiomap_file_title" />
10:     </PreferenceCategory>
11:     <PreferenceCategory android:title="Mode" >
12:         <CheckBoxPreference
13:             android:defaultValue="true"
14:             android:key="modeFloor"
15:             android:summaryOff="Floor Mode Off"
16:             android:summaryOn="Floor Mode On"
17:             android:title="Floor Mode" />
18:     </PreferenceCategory>
19:
20: </PreferenceScreen>
```

```
 1: <?xml version="1.0" encoding="utf-8"?>
 2: <menu xmlns:android="http://schemas.android.com/apk/res/android" >
 3:
 4:     <item
 5:         android:id="@+id/Prefs"
 6:         android:icon="@android:drawable/ic_menu_preferences"
 7:         android:title="Preferences">
 8:     </item>
 9:     <item
10:         android:id="@+id/Help"
11:         android:icon="@android:drawable/ic_menu_help"
12:         android:title="Help">
13:     </item>
14:     <item
15:         android:id="@+id/About"
16:         android:icon="@android:drawable/ic_menu_info_details"
17:         android:title="About">
18:     </item>
19:
20: </menu>
```

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

    <string name="app_name">CLODA</string>
    <string name="prefs_name">Airplace Tracker Preferences</string>
    <string name="app_choose_folder">Choose folder</string>
    <string name="building_settings">Building Settings</string>
    <string name="building_title">Floor Plan</string>
    <string name="building_summary">The floor plan image of building</string>
    <string name="radiomap_file_settings">Radiomap Settings</string>
    <string name="radiomap_download_title">RadioMap Download Settings</string>
    <string name="radiomap_download_summary">Configure parameters to download radio map</string>
    <string name="radiomap_file_title">Radiomap File</string>
    <string name="radiomap_file_summary">The radiomap that is currently used to estimate
                position</string>
    <string name="download_connection_title">Connection Settings</string>
    <string name="serverIP_title">Server Address</string>
    <string name="serverIP_summary">The IP address of the server</string>
    <string name="serverPORT_title">Port number</string>
    <string name="serverPORT_summary">The non-IANA reserved port</string>
    <string name="downloading_file_title">Downloading Settings</string>
    <string name="test_data_file_title">Test Data File</string>
    <string name="test_data_file_summary">The data file to use for offline mode statistics</string>
    <string name="parentDir">..</string>
    <string name="homeDir">.</string>
    <string name="rootDir">/</string>
    <string name="strMinusOne">"-1"</string>
    <string name="strZero">"0"</string>
    <string name="strFirst">"1"</string>
    <string name="strSecond">"2"</string>
    <string name="strFindMe">Find Me</string>
    <string name="strTrackMe">Track Me</string>
    <string name="strtRecording">Start Recording</string>
    <string name="sampling_title">Sampling Settings</string>
    <string name="samples_title">Samples Number</string>
    <string name="samples_summary">The number of samples to record</string>
    <string name="samples_interval_title">Samples Interval</string>
    <string name="samples_interval_summary">Time between two consecutive samplings</string>
    <string name="storing_title">Storing Settings</string>
    <string name="folder_browser_title">Folder</string>
    <string name="folder_browser_summary">The folder to store RSS log files</string>
    <string name="filename_title">Filename</string>
    <string name="filename_summary">The file contains RSS log records</string>
    <string name="uploading_title">Uploading Settings</string>
    <string name="server_IP_title">Server Address</string>
    <string name="server_IP_summary">The server\'s IP address</string>
    <string name="server_PORT_title">Port number</string>
    <string name="server_PORT_summary">The server\'s port</string>
    <string name="upload_file_title">File</string>
    <string name="upload_file_summary">The file to upload</string>
    <string name="menu_about">About</string>
    <string name="menu_help">Help</string>
    <string name="menu_exit">Exit</string>
    <string name="menu_logger">Logger</string>
    <string name="menu_preferences">Preferences</string>
    <string name="strHelpMenu">"> 1. Put a product's barcode inside the scanner's rectangle (steady handling of the phone and good lighting make results faster).\n\n"
                                                                > "2. You can also manually enter a barcode through the applications menu.\n\n"
                                                                > "3. When a barcode is scanned successfully you will hear a \"beep\" sound (if you can't hear it, you might need to adjust your phone's sound volume).\n\n"</string>
    <string name="strAbutMenu">"CrowdSpace Context Scanner implemented \n2012\n\nAffiliation:\n"
                                                                "Data Management Systems Laboratory\n"
                                                                "Dept. of Computer Science\n"
                                                                "University of Cyprus\n"
                                                                "P.O. Box 20537\n"
                                                                "1678 Nicosia, CYPRUS\n"
                                                                "Web: http://dmsl.cs.ucy.ac.cy/\n"
                                                                "Email: dmsl@cs.ucy.ac.cy\n"
                                                                "Tel: +357-22-892755\n"
                                                                "Fax: +357-22-892701"</string>
    <string name="msg_camera_framework_bug">Sorry, the Android camera encountered a problem. You may need to restart the device.</string>

    <string-array name="samplesArray">
        <item>5 samples</item>
        <item>10 samples</item>
        <item>15 samples</item>
        <item>20 samples</item>
        <item>25 samples</item>
        <item>30 samples</item>
    </string-array>
    <string-array name="samplesValues">
        <item>5</item>
        <item>10</item>
        <item>15</item>
        <item>20</item>
        <item>25</item>
        <item>30</item>
    </string-array>
    <string-array name="intervalArray">
        <item>0.5 sec</item>
        <item>1 sec</item>
        <item>2 sec</item>
    </string-array>
    <string-array name="intervalValues">
        <item>500</item>
        <item>1000</item>
        <item>2000</item>
    </string-array>

    <string name="image_desc">Image</string>
    <string name="Floors">Floors</string>
    <string name="FloorPlinEna">Floor -1</string>
    <string name="FloorZero">Floor 0</string>
    <string name="FloorOne">Floor 1</string>
```

```
103:     <string name="FloorTwo">Floor 2</string>
104:
105: </resources>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
 Copyright (C) 2008 ZXing authors

 Licensed under the Apache License, Version 2.0 (the "License");
 you may not use this file except in compliance with the License.
 You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

 Unless required by applicable law or agreed to in writing, software
 distributed under the License is distributed on an "AS IS" BASIS,
 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 See the License for the specific language governing permissions and
 limitations under the License.
-->
<resources>

    <color name="contents_text">#ff000000</color>
    <color name="encode_view">#ffffffff</color>
    <color name="help_button_view">#ffcccccc</color>
    <color name="help_view">#ff404040</color>
    <color name="possible_result_points">#c0ffff00</color>
    <color name="result_image_border">#ffffffff</color>
    <color name="result_minor_text">#ffc0c0c0</color>
    <color name="result_points">#c000ff00</color>
    <color name="result_text">#ffffffff</color>
    <color name="result_view">#b0000000</color>
    <color name="sbc_header_text">#ff808080</color>
    <color name="sbc_header_view">#ffffffff</color>
    <color name="sbc_list_item">#fffff0e0</color>
    <color name="sbc_layout_view">#ffffffff</color>
    <color name="sbc_page_number_text">#ff000000</color>
    <color name="sbc_snippet_text">#ff4b4b4b</color>
    <color name="share_text">#ff000000</color>
    <color name="status_text">#ffffffff</color>
    <color name="transparent">#00000000</color>
    <color name="viewfinder_frame">#ff000000</color>
    <color name="viewfinder_laser">#ffff0000</color>
    <color name="viewfinder_mask">#60000000</color>
    <color name="green">#00FF00</color>
    <color name="side_navigation_background">#303030</color>
    <color name="side_navigation_outside_background">#7000</color>
    <color name="side_navigation_list_divider_color">#4f4f4f</color>
    <color name="side_navigation_item_text_color">#f3f3f3</color>
    <color name="white">#FFFFFF</color>
    <color name="yellow">#FFFF00</color>
    <color name="fuchsia">#FF00FF</color>
    <color name="red">#FF0000</color>
    <color name="silver">#C0C0C0</color>
    <color name="gray">#808080</color>
    <color name="olive">#808000</color>
    <color name="purple">#800080</color>
    <color name="maroon">#800000</color>
    <color name="aqua">#00FFFF</color>
    <color name="lime">#00FF00</color>
    <color name="teal">#008080</color>
    <color name="blue">#0000FF</color>
    <color name="navy">#000080</color>
    <color name="black">#000000</color>
    <color name="orange">#FEC680</color>
    <color name="dark_orange">#FFBB3D</color>
    <color name="lightblue"> #87CEFA</color>
    <color name="sherlockBlue">   #fff3f3f3</color>
    <color name="darkGray">#cdc9c9</color>

</resources>
```

```
 1: <?xml version="1.0" encoding="utf-8"?>
 2: <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
 3:     android:layout_width="fill_parent"
 4:     android:layout_height="wrap_content"
 5:     android:orientation="horizontal" >
 6:
 7:     <ImageView
 8:         android:id="@+id/icon"
 9:         android:layout_width="wrap_content"
10:         android:layout_height="wrap_content"
11:         android:src="@drawable/icon" />
12:
13:     <TextView
14:         android:id="@+id/file"
15:         android:layout_width="wrap_content"
16:         android:layout_height="wrap_content" />
17:
18: </LinearLayout>
```

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

    <PreferenceCategory android:title="@string/sampling_title" >
        <ListPreference
            android:defaultValue="5"
            android:entries="@array/samplesArray"
            android:entryValues="@array/samplesValues"
            android:key="samples_num"
            android:summary="@string/samples_summary"
            android:title="@string/samples_title" />
        <ListPreference
            android:defaultValue="1000"
            android:entries="@array/intervalArray"
            android:entryValues="@array/intervalValues"
            android:key="samples_interval"
            android:summary="@string/samples_interval_summary"
            android:title="@string/samples_interval_title" />
    </PreferenceCategory>
    <PreferenceCategory android:title="@string/storing_title" >
        <Preference
            android:key="folder_browser"
            android:summary="@string/folder_browser_summary"
            android:title="@string/folder_browser_title" />

        <EditTextPreference
            android:defaultValue="rss-log"
            android:key="filename_log"
            android:summary="@string/filename_summary"
            android:title="@string/filename_title" >
        </EditTextPreference>

        <CheckBoxPreference
            android:defaultValue="false"
            android:key="write_mode"
            android:summaryOff="Append mode"
            android:summaryOn="Overwrite mode"
            android:title="Write Mode" />
    </PreferenceCategory>

</PreferenceScreen>
```

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

    <WebView
        android:id="@+id/map"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true" />

    <LinearLayout
        android:id="@+id/linearLayout2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:orientation="vertical" >

        <TextView
            android:id="@+id/detectedAPs"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentLeft="true"
            android:layout_marginTop="21dp"
            android:background="#80FFFFFF"
            android:text="AP: "
            android:textColor="#353535"
            android:textSize="12.5dip"
            android:textStyle="bold" />

        <TextView
            android:id="@+id/trackingInfoLabel"
            android:layout_width="140dip"
            android:layout_height="fill_parent"
            android:layout_weight="0.0"
            android:background="#80FFFFFF"
            android:maxLines="5"
            android:paddingLeft="2dip"
            android:singleLine="false"
            android:text="Heading: \nX:\nY:"
            android:textColor="#353535"
            android:textSize="12.5dip"
            android:textStyle="bold" />
    </LinearLayout>

</RelativeLayout>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
 Copyright (C) 2008 ZXing authors

 Licensed under the Apache License, Version 2.0 (the "License");
 you may not use this file except in compliance with the License.
 You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

 Unless required by applicable law or agreed to in writing, software
 distributed under the License is distributed on an "AS IS" BASIS,
 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 See the License for the specific language governing permissions and
 limitations under the License.
-->
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <SurfaceView
        android:id="@+id/preview_view"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent" />

    <com.google.zxing.client.android.ViewfinderView
        android:id="@+id/viewfinder_view"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:background="@color/transparent" />

    <LinearLayout
        android:id="@+id/Scanner"
        android:layout_width="fill_parent"
        android:layout_height="50dp"
        android:layout_gravity="top"
        android:layout_margin="7dp"
        android:gravity="top"
        android:orientation="horizontal"
        android:visibility="visible"
        android:weightSum="10" >
    </LinearLayout>

</FrameLayout>
```

```
    1: <?xml version="1.0" encoding="utf-8"?>
    2: <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    3:     android:id="@+id/aboutwindow"
    4:     android:layout_width="match_parent"
    5:     android:layout_height="match_parent"
    6:     android:orientation="vertical" >
    7:
    8:     <TextView
    9:         android:id="@+id/tAbout"
   10:         android:layout_width="match_parent"
   11:         android:layout_height="wrap_content"
   12:         android:layout_weight="0.64"
   13:         android:text="About CLODA: Crowdsourced Linked Open Data\n\nCrowdSourc
ed Linked Open Data implemented in 2012-2013 as a part of my dissertation project\n\n
   14:                                     Affiliation:\n\n
   15:                                     Data Management Systems Laboratory\n
   16:                                     Dept. of Computer Science\n
   17:                                     University of Cyprus\n
   18:                                     P.O. Box 20537\n
   19:                                     1678 Nicosia, CYPRUS\n
   20:                                     Web: http://dmsl.cs.ucy.ac.cy/\n
   21:                                     Email: dmsl@cs.ucy.ac.cy\n
   22:                                     Tel: +357-22-892755\n
   23:                                     Fax: +357-22-892701\n\n
   24:                                     Development:\n
   25:                                     Julia Metochi\n
   26:                                     Undergraduate Student at University Of\n
   27:                                     Cyprus\n\n
   28:                                     Supervisor:\n
   29:                                     Dr. Demetris Zeinalipour

   30:                                                 \n\n" />
   31:
   32: </LinearLayout>
```

```
 1: <?xml version="1.0" encoding="utf-8"?>
 2: <resources>
 3:
 4:     <dimen name="side_navigation_width">200dp</dimen>
 5:     <dimen name="side_navigation_item_padding_topbottom">12dp</dimen>
 6:     <dimen name="side_navigation_item_padding_leftright">8dp</dimen>
 7:     <dimen name="side_navigation_item_image_height">24dp</dimen>
 8:     <dimen name="side_navigation_item_image_width">24dp</dimen>
 9:     <dimen name="side_navigation_item_image_padding">2dp</dimen>
10:     <dimen name="side_navigation_item_text_size">18sp</dimen>
11:     <dimen name="side_navigation_item_text_padding_left">10dp</dimen>
12:
13: </resources>
```

```
<?php

/*
 *
 * User: Julia
 *
 */

error_reporting(E_ALL);
ini_set('display_errors','1');



require_once "./lib/sag/src/Sag.php";
require_once "./lib/bones.php";


$bones = new Bones();
if(isset($_POST['byFloor'])){
    $floor=$_POST['byFloor'];
   $floorSearch='"' . $floor . '"';
    $arr=array();
    $result=$bones->couch->get('_design/FloorItems/_view/itemsPerFloor?key=' .
$floorSearch)->body->rows;
    if($result){

        echo json_encode($result);
    }

}
```

```
 1: <?php
 2: /*
 3:  *
 4:  * User: Julia
 5:  *
 6:  */
 7: error_reporting(E_ALL);
 8: ini_set('display_errors','1');
 9: $couch_dsn = "http://localhost:5984/";
10: $couch_db = "crowdspace";
11:
12: include_once('./lib/couch.php');
13: include_once('./lib/couchClient.php');
14: include_once('./lib/couchDocument.php');
15:
16:
17: $client = new couchClient($couch_dsn,$couch_db);
18:
19:
20:         $info = $client->getDatabaseInfos();
21:
22:
23: print_r($info);
24:
25: ?>
26:
```

```
    1: <?php
    2:
    3: /*
    4:  *
    5:  * User: Julia
    6:  *
    7:  */
    8:
    9: error_reporting(E_ALL);
   10: ini_set('display_errors','1');
   11:
   12:
   13: require_once "./lib/sag/src/Sag.php";
   14: require_once "./lib/bones.php";
   15:
   16:
   17: $bones = new Bones();
   18: if(isset($_POST['searchByDocId'])){
   19:     $docId=$_POST['searchByDocId'];
   20:     $docIdSearch='"' . $docId . '"';
   21:
   22:     $arr=array();
   23:     $result=$bones->couch->get('_design/searchDocId/_view/by_docId?key=' . $do
cIdSearch)->body->rows;
   24:     if($result){
   25:         foreach ($bones->couch->get('_design/searchDocId/_view/by_docId?key='
. $docIdSearch)->body->rows as $_post){
   26:             $prodBarcode=$_post->value->barcode;
   27:             $title=$_post->value->title;
   28:             $brand=$_post->value->brand;
   29:             $price= $_post->value->price;
   30:             $pointX=$_post->value->pointX;
   31:             $pointY=$_post->value->pointY;
   32:             $image=$_post->value->image;
   33:             $floor=$_post->value->floor;
   34:             $docId=$_post->value->_id;
   35:             $docRev=$_post->value->_rev;
   36:
   37:             $arr[]=array('barcode' => $prodBarcode,'title' => $title,'brand' =
> $brand,'price' => $price,'pointX'=>$pointX,'pointY'=>$pointY,'image'=>$image,'floor'
=>$floor, 'docId'=>$docId,'docRev'=>$docRev);
   38:
   39:
   40:         }
   41:         echo json_encode($arr,JSON_FORCE_OBJECT);
   42:     }
   43:     else{
   44:
   45:         $arr[]=array('status' => 'fail');
   46:         echo json_encode($arr,JSON_FORCE_OBJECT);
   47:     }
   48:
   49: }
   50: ?>
```

```php
<?php
/**
 * Created by JetBrains PhpStorm.
 * User: Julia
 * Date: 27/3/2013
 * Time: 12:15 Ï\200Î¼
 * To change this template use File | Settings | File Templates.
 */
require_once "./lib/sag/src/Sag.php";
require_once "./lib/bones.php";

$bones = new Bones();
$docId=$_POST['docId'];
$docRev=$_POST['docRev'];
echo $docRev;

try {
 $bones->couch->delete($docId,$docRev);
}
catch(SagCouchException $e) {
  $bones->error500($e);
}
?>
```

```php
<?php

/*
 *
 * User: Julia
 *
 */

require_once "./lib/sag/src/Sag.php";
require_once "./lib/bones.php";

error_reporting(E_ALL);
ini_set('display_errors','1');

$bones = new Bones();


$product=json_decode($_POST['value']);
$bones->couch->post($product);



?>
```

```php
<?php

/*
 *
 * User: Julia
 *
 */

require_once "./lib/sag/src/Sag.php";
require_once "./lib/bones.php";

$bones = new Bones();


$docId=$_POST['docId'];
$itemJson=json_decode($_POST['item']);

$response=$bones->couch->put($docId,$itemJson);

echo $response;
?>
```

```
    1: <?php
    2:
    3: /*
    4:  *
    5:  * User: Julia
    6:  *
    7:  */
    8:
    9: error_reporting(E_ALL);
   10: ini_set('display_errors','1');
   11:
   12:
   13: require_once "./lib/sag/src/Sag.php";
   14: require_once "./lib/bones.php";
   15:
   16:
   17: $bones = new Bones();
   18: if(isset($_POST['search'])){
   19: $barcode=$_POST['search'];
   20: $barcodeSearch='"' . $barcode . '"';
   21:
   22: $arr=array();
   23:     $result=$bones->couch->get('_design/searchBarcode/_view/by_barcode?key=' .
 $barcodeSearch)->body->rows;
   24:     if($result){
   25:         foreach ($bones->couch->get('_design/searchBarcode/_view/by_barcode?ke
y=' . $barcodeSearch)->body->rows as $_post){
   26:             $prodBarcode=$_post->value->barcode;
   27:             $title=$_post->value->title;
   28:             $brand=$_post->value->brand;
   29:             $price= $_post->value->price;
   30:             $pointX=$_post->value->pointX;
   31:             $pointY=$_post->value->pointY;
   32:             $image=$_post->value->image;
   33:             $floor=$_post->value->floor;
   34:             $docId=$_post->value->_id;
   35:             $docRev=$_post->value->_rev;
   36:
   37:              $arr[]=array('barcode' => $prodBarcode,'title' => $title,'brand'
=> $brand,'price' => $price,'pointX'=>$pointX,'pointY'=>$pointY,'image'=>$image,'floor
'=>$floor, 'docId'=>$docId,'docRev'=>$docRev);
   38:
   39:
   40:
   41:     }
   42:         echo json_encode($arr,JSON_FORCE_OBJECT);
   43:     }
   44:     else{
   45:
   46:         $arr[]=array('status' => 'fail');
   47:         echo json_encode($arr,JSON_FORCE_OBJECT);
   48:     }
   49:
   50: }
   51: ?>
```

```
<!DOCTYPE html>
<html>

<head>

    <meta name="viewport" content="initial-scale=1.0, user-scalable=no">
    <meta charset="utf-8">
    <title>CrowdSpace Context Scanner</title>

    <META HTTP-EQUIV="Pragma" CONTENT="no-cache">
    <META HTTP-EQUIV="Expires" CONTENT="-1">
</head>
<body style="margin:0px; padding:0px;" onload="initialize()">
<style type="text/css">
    html { height: 100% }
    body { height: 100%; margin: 0; padding: 0 }
    #map_canvas { height: 100% }
</style>

<script src="https://maps.googleapis.com/maps/api/js?key=AIzaSyB0hxkB5QPcRgJ5OokCeSOa154CnuJYKBs&sensor=true"></script>
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js"></script>
<script src="//ajax.googleapis.com/ajax/libs/jqueryui/1.10.2/jquery-ui.min.js"></script>
<script type="text/javascript">
console.firebug=true;

var overlay;
var lon,lat,updateMarker;
var  lonA = 33.410649597644806, latA = 35.14498551757748;
var  lonC = 33.4115669131279, latC = 35.14415427652711;
var swBound;
var neBound ;
var bounds;
var map,  markers = [];
USGSOverlay.prototype = new google.maps.OverlayView();
var srcImage,curLon,curLat;

var items_array=[];
var Imarkers=[],ImarkersCouch=[];
var itemsJSON,itemsJSONcouch;
var itemsCouch;
/*************************************************************************************/

/************************JQUERY*****************************/


$(document).ready(function() {

});

$(document).on('click', '.item', function(){
    myAndroidFunctions.ItemMenu($(this).attr('id').toString());

});



/************************JQUERY*****************************/


function initialize() {
    var ucy = new google.maps.LatLng(35.144569,33.411107);
    var mapOptions = {
        zoom: 19,
        center: ucy,
        panControl: false,
        zoomControl: true,
        scrollWheel: true,
        disableDefaultUI: true,
        mapTypeId: google.maps.MapTypeId.ROADMAP
    };

    map = new google.maps.Map(document.getElementById('map_canvas'),
            mapOptions);

    // Photograph courtesy of the U.S. Geological Survey
    srcImage = 'images/';

      google.maps.event.addListener(map, 'click', marker);

}

/*************************************************************************************/
function toast(){
    myAndroidFunctions.showToast("Hello");
}

/*************************************************************************************/
 /*****************************************************ADD MARKER*********************************/

function marker(event){
    curLon=event.latLng.lng();
    curLat=event.latLng.lat();
    clearOverlays();
    addMarker(event.latLng);
    sendCurLonLat(curLon,curLat);
}
/*************************************************************************************/
//blue
var pinColor = "c4d4f3";


var pinImage = new google.maps.MarkerImage("http://chart.apis.google.com/chart?chst=d_map_pin_letter&chld=%E2%80%A2|" + pinColor,
        new google.maps.Size(21, 34),
        new google.maps.Point(0,0),
        new google.maps.Point(10, 34));


// Add a marker to the map and push to the array.
function addMarker(location) {

    marker = new google.maps.Marker({
        position: location,
        icon: "
```

```
rhtAAAESElEQVRYw+2XT2wUZRTAfyMDvJVWZ0xLuqEV5sBhG3roamJa7cFN1FDSGHsQgXjkQPZkevJCjFHPHjj
oofFAykEQYoj1QLrBmGxvcCBhEDBjYmGX8GeGOGVesh98HmapmHjolpVw2G/y8r7kzZv5zTfve+99jrWW53m8w
HM+eoA9wB5gD/B/Hu5GHR3HcQABtgNFwGs/7yFwH2gCt4AH9imqgbMRX8dxCsBOz/PenJya3D++Z3wsGAm2KYo
gRHeiLLwY/lZfrp9qNpvngd+ttavPBNBxHA94Y3pm+ujs3tny6NioDA0P4Xt+foMBVSVaibgcXm7Vfq5dPXHmx
GcYzgN3O15Na+26BfBE5ED1cLWxVFuyjUbDWmutbdn/1HEc2/py3c59MnfX87wjwEAn77PW0glcAXjv4IGDN5d
qS2sgWStb00/Kk7al2pKtHqnew+UjoL8TQLeDDbFz4vWJo/tm9hUrUxXUKBi4mCg/rTRYbiREKQR9MFH02DfsM
z7gAUplqkKSJH54Pfyqdq523XGcC9baR91MMyIibwWl4LXx8fE1uNMrytylBvMxRIMB7C4RDQbMxzB3qcHpPxL
UgBql8naFYDh41RvwZoCXup0HtxeHi/vLpfLWYFcABup3lGN/xkSuT6G/CAgZCgiFfp/I9TkWxVxMFIwirjAxN
bGpOFTcD+zoNmBRXBkr7Snlq4eyeEeJNDdmRsmMUtDH8zxgI+CHppJ7QHmsjLgy1AngehO1h4v4vg8K6kL9dra
WVjAKZGQU1hyytq1+O0N2CWoSvAEP4EXg5W4DbsWwRURQowgQpgqugIkRhHwx43Zx0TUdaR6DilBwQdHNbWNXA
Vuq+jCOY7whDyhQciFSJb/4B8okOTgJghAIqIHMxGiqkPKoXQ67GoNJptlqeCVEFdCMsg+o5pIq2taYttZ8Ptk
HmclAIQxDYo0fAEm3AZtJmlytL9fRNCE2ynSfUJQ83aiJQTPUxDmziREDRZTKoIAqSZpQ/7WOpnoPuNltwFuqe
qp2rtaKrkVoqgSiVEc8ypK3MWJAFMRkeMCoQDXwCERJUiWOlYXj81ZVzwAr3Y7BBxh+STS5tnByYbR6uIrv+0x
6gr/b40IshIkSaUYgBUqeUB4USgKaKPFfMfPfzZOoNlT1VCe/eN3djOM424C90iffzs7MvnLo40P4/YKIj4j8+
1Pb1UPTHO70yQUWTi6uapp8Chy31nYfsA05ABwQkc8r71S82Q9mKQ4UkT6hsPlxPwFZK0NTRTVh/vgCtXO1VTX
6NYZvgBudtFydAjrAIPCuuPIFLiPTe6dfKO0pEYwEFKRAZjKiKCK8ErJ4dtGqoYnRL4EfgYa1dt0p5mk66n6gB
LyPy4dt6D5gE/AISIEYw1nge+AScH8jrb+z0eOC4zib2l3Jjrb4wBag1d4EN4EbQGKtNc/0TNI7dvYAe4A9wB5
gD3Bt/A1VrekRV9CDowAAAABJRU5ErkJggg==",
```

```
  114:         map: map
  115:
  116:     });
  117:
  118:     markers.push(marker);
  119: }
  120:
  121: /*************************************************************************
**********************************/
  122: function sendCurLonLat(curLon,curLat){
  123:     myAndroidFunctions.getCurLatLon(curLon,curLat);
  124: }
  125:
  126: /*************************************************************************
**************************************/
  127: function clearOverlays(){
  128:     setAllMap(null);
  129:     markers=[];
  130: }
  131:
  132: /*************************************************************************
**************************************/
  133:
  134: function setAllMap(map) {
  135:     for (var i = 0; i < markers.length; i++) {
  136:         markers[i].setMap(map);
  137:     }
  138: }
  139: /*************************************************************END OF ADD MARKER
 **************************************/
  140:
  141:
  142: /*************************************************************************
**************************************/
  143: function viewUniMap(floor){
  144:
  145:     switch (floor){
  146:         case -1:
  147:
  148:             latA=35.14412137763014, lonA=33.410782366991045
  149:             latC=35.1449175272035, lonC=33.41167688369751;
  150:             srcImage+='-1_big.png';
  151:
  152:             break
  153:         case 0:
  154:             latA=35.14411479784918,lonA=33.410791754722595;
  155:             latC=35.14490546438928,lonC=33.41168090701103;
  156:             srcImage+='0.png';
  157:
  158:             break
  159:         case 1:
  160:
  161:             latA=35.14411589447936, lonA=33.4107930958271;
  162:             latC=35.14490985086738,lonC= 33.41168358922005;
  163:             srcImage+='1.png';
  164:
  165:             break
  166:         case 2:
  167:
  168:             latA=35.14411260458871, lonA=33.41078907251358;
  169:             latC=35.14490985086738,lonC= 33.41168358922005;
  170:             srcImage+='2.png';
  171:
  172:             break
  173:     }
  174:
  175:     swBound = new google.maps.LatLng(latA,lonA );
  176:     neBound = new google.maps.LatLng(latC,lonC );
  177:     bounds = new google.maps.LatLngBounds(swBound, neBound);
  178:     overlay = new USGSOverlay(bounds, srcImage, map);
  179: }
  180: /*************************************************************************
**************************************/
  181: function TrackMeMarker(lon,lat){
  182:
  183:     updateMarker=new google.maps.LatLng(lat,lon);
  184:     clearOverlays();
  185:     addMarker(updateMarker);
  186: }
  187: /*************************************************************************
**************************************/
  188: function USGSOverlay(bounds, image, map) {
  189:
  190:     // Now initialize all properties.
  191:     this.bounds_ = bounds;
  192:     this.image_ = image;
  193:     this.map_ = map;
  194:
  195:     // We define a property to hold the image's
  196:     // div. We'll actually create this div
  197:     // upon receipt of the add() method so we'll
  198:     // leave it null for now.
  199:     this.div_ = null;
  200:
  201:     // Explicitly call setMap on this overlay
  202:     this.setMap(map);
  203: }
  204:
  205: USGSOverlay.prototype.onAdd = function() {
  206:
  207:     // Note: an overlay's receipt of add() indicates that
  208:     // the map's panes are now available for attaching
  209:     // the overlay to the map via the DOM.
  210:
```

```
211:     // Create the DIV and set some basic attributes.
212:     var div = document.createElement('div');
213:     div.style.border = 'none';
214:     div.style.borderWidth = '0px';
215:     div.style.position = 'absolute';
216:
217:     // Create an IMG element and attach it to the DIV.
218:     var img = document.createElement('img');
219:     img.src = this.image_;
220:     img.style.width = '100%';
221:     img.style.height = '100%';
222:     div.appendChild(img);
223:
224:
225:     // Set the overlay's div_ property to this DIV
226:     this.div_ = div;
227:
228:     // We add an overlay to a map via one of the map's panes.
229:     // We'll add this overlay to the overlayImage pane.
230:     var panes = this.getPanes();
231:     panes.mapPane.appendChild(this.div_);
232: }
233:
234: USGSOverlay.prototype.draw = function() {
235:
236:     // Size and position the overlay. We use a southwest and northeast
237:     // position of the overlay to peg it to the correct position and size.
238:     // We need to retrieve the projection from this overlay to do this.
239:     var overlayProjection = this.getProjection();
240:
241:     // Retrieve the southwest and northeast coordinates of this overlay
242:     // in latlngs and convert them to pixels coordinates.
243:     // We'll use these coordinates to resize the DIV.
244:     var sw = overlayProjection.fromLatLngToDivPixel(this.bounds_.getSouthWest(
));
245:     var ne = overlayProjection.fromLatLngToDivPixel(this.bounds_.getNorthEast(
));
246:
247:     // Resize the image's DIV to fit the indicated dimensions.
248:     var div = this.div_;
249:     div.style.left = sw.x + 'px';
250:     div.style.top = ne.y + 'px';
251:     div.style.width = (ne.x - sw.x) + 'px';
252:     div.style.height = (sw.y - ne.y) + 'px';
253: }
254:
255: USGSOverlay.prototype.onRemove = function() {
256:
257:     this.div_.parentNode.removeChild(this.div_);
258: }
259:
260: // Note that the visibility property must be a string enclosed in quotes
261: USGSOverlay.prototype.hide = function() {
262:     if (this.div_) {
263:         this.div_.style.visibility = 'hidden';
264:     }
265: }
266:
267: USGSOverlay.prototype.show = function() {
268:     if (this.div_) {
269:         this.div_.style.visibility = 'visible';
270:     }
271: }
272:
273: USGSOverlay.prototype.toggle = function() {
274:
275:     if (this.div_) {
276:         if (this.div_.style.visibility == 'hidden') {
277:             this.show();
278:         } else {
279:             this.hide();
280:         }
281:     }
282: }
283:
284: /*************************************************************************
*************************************/
285: function hideShowOverlay(){
286:     overlay.setMap();
287:     srcImage = 'images/';
288:     clearOverlays();
289:     google.maps.event.addListener(map, 'click', marker);
290: }
291:
292: //MARKERS ON ITEM POSITIONS
293: /*************************************************************************
*************************************/
294: function getItemsInJSON(itemsJSON){
295:
296:
297:     var itemsStringArray = "["+itemsJSON+"]";
298:
299:     showItemsOnFloor(map,itemsStringArray);
300: }
301: /*************************************************************************
*************************************/
302:
303: function getItemsInJsonFromCouch(itemsJSONcouch){
304:
305:     var itemsStrArray = "["+itemsJSONcouch+"]";
306:     showCouchItemsOnFloor(map,itemsStrArray);
307:
308:
309:
310: }
311: /*************************************************************************
*************************************/
312: function showCouchItemsOnFloor(map,itemsCouch){
313:     var jsonArray = JSON.parse(itemsCouch);
314:     var infoWindow=new google.maps.InfoWindow();
315:
316:     for(var i=0;i<jsonArray.length;i++){
317:
318:         var brand=jsonArray[i].value.brand;
319:         var price=jsonArray[i].value.price;
320:         var title=jsonArray[i].value.title;
321:         var x=jsonArray[i].value.pointX;
322:         var y=jsonArray[i].value.pointY;
323:         var iPos=new google.maps.LatLng(y,x);
324:         var barcode=jsonArray[i].value.barcode;
325:         var floor=jsonArray[i].value.floor;
326:         var image=jsonArray[i].value.image;
```

```
  327:         var doc_id=jsonArray[i].value._id;
  328:         var doc_rev=jsonArray[i].value._rev;
  329:
  330:
  331:         var item_info="barcode:"+barcode+"@"+"Lon:"+x+"@"+"Lat:"+y+"@"+"Price:
"+price+"@"+"Brand:@"+brand+"@"+"Title:@"+title.replace(" ","@")+"@"+"Floor:"+floor+"@
"+"_id:"+doc_id+"@"+"_rev:"+doc_rev;
  332:
  333:         var markerItemCouch=new google.maps.Marker({
  334:             position: iPos,
  335:             map:map
  336:         });
  337:
  338:
  339:         var doc=doc_id+"@"+doc_rev;
  340:         markerItemCouch.html = '<div class="item" id='+doc+'>   ITEM INFO <img
 src='+image+' style="float: right" width=100; height=120" /><br><br>Barcode: '+barcod
e+'<br>Pos: '+iPos+'<br>Price: '+price+'<br>Brand: '+brand+'<br>Title: '+title+'<br>Fl
oor: '+floor+'<br><br>TAP INFO WINDOW TO VIEW OPTIONS<br> </div> ' ;
  341:         ImarkersCouch.push(markerItemCouch);
  342:
  343:         google.maps.event.addListener(markerItemCouch,"click",function(){
  344:
  345:             infoWindow.setContent(this.html);
  346:             infoWindow.open(map,this);
  347:         });
  348:
  349:     }
  350:
  351: }
  352: /*************************************************************************
**************************************/
  353: function showItemsOnFloor(map,items){
  354:
  355:
  356:     var jsonArray = JSON.parse(items);
  357:     var infoWindow=new google.maps.InfoWindow();
  358:
  359:     for(var i=0;i<jsonArray.length;i++){
  360:
  361:
  362:         var brand=jsonArray[i].brand;
  363:         var price=jsonArray[i].price;
  364:         var title=jsonArray[i].title;
  365:         var x=jsonArray[i].pointX;
  366:         var y=jsonArray[i].pointY;
  367:         var iPos=new google.maps.LatLng(y,x);
  368:         var barcode=jsonArray[i].barcode;
  369:         var floor=jsonArray[i].floor;
  370:         var image=jsonArray[i].image;
  371:         var doc_id=jsonArray[i].doc_id;
  372:         var doc_rev=jsonArray[i].doc_rev;
  373:
  374:         var item_info="barcode:"+barcode+"@"+"Lon:"+x+"@"+"Lat:"+y+"@"+"Price:
"+price+"@"+"Brand:@"+brand+"@"+"Title:@"+title+"@"+"Floor:"+floor+"@"+"_id:"+doc_id+"
@"+"_rev:"+doc_rev;
  375:
  376:
  377:
  378:         var markerItem=new google.maps.Marker({
  379:             position: iPos,
  380:             map:map
  381:
  382:         });
  383:
  384:         var doc=doc_id+"@"+doc_rev;
  385:         markerItem.html = '<div class="item" id='+doc+'>   ITEM INFO <img src=
'+image+' style="float: right" width=100; height=120" /><br><br>Barcode: '+barcode+'<b
r>Pos: '+iPos+'<br>Price: '+price+'<br>Brand: '+brand+'<br>Title: '+title+'<br>Floor:
'+floor+'<br><br>TAP INFO WINDOW TO VIEW OPTIONS<br></div>';
  386:         Imarkers.push(markerItem);
  387:         google.maps.event.addListener(markerItem,"click",function(){
  388:
  389:             infoWindow.setContent(this.html);
  390:             infoWindow.open(map,this);
  391:         });
  392:     }
  393: }
  394: /*************************************************************************
**************************************/
  395: function clearMarkers() {
  396:
  397:     if(Imarkers){
  398:         for(var i = 0; i < Imarkers.length; i++) {
  399:
  400:             Imarkers[i].setMap(null);
  401:         }
  402:     }
  403:     if(ImarkersCouch){
  404:         for(var i = 0; i < ImarkersCouch.length; i++) {
  405:
  406:             ImarkersCouch[i].setMap(null);
  407:         }
  408:     }
  409:
  410:
  411:
  412:
  413:
  414:
  415:
  416: }
  417: /*************************************************************************
**************************************/
  418: function clearMarkers() {
  419:
  420:     for(var i = 0; i < Imarkers.length; i++) {
  421:
  422:         Imarkers[i].setMap(null);
  423:     }
  424:     for(var i = 0; i < ImarkersCouch.length; i++) {
  425:
  426:         ImarkersCouch[i].setMap(null);
  427:     }
  428:
  429:
  430:
  431:
  432:
  433: }
  434:
  435: </script>
```

```
436:
437:
438:
439:
440: <div id="map_canvas">
441:
442: </div>
443:
444:
445: </body>
446: </html>
```

```java
/*
 * Copyright (c) 2011, KIOS Research Center and Data Management Systems Lab,
 * University of Cyprus. All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without modification,
 * are permitted provided that the following conditions are met:
 *
 *    * Redistributions of source code must retain the above copyright notice, this
 *      list of conditions and the following disclaimer.
 *    * Redistributions in binary form must reproduce the above copyright notice,
 *      this list of conditions and the following disclaimer in the documentation
 *      and/or other materials provided with the distribution.
 *    * Neither the name of the Sony Ericsson Mobile Communication AB nor the names
 *      of its contributors may be used to endorse or promote products derived from
 *      this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
 * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.
 * IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT,
 * INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING,
 * BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
 * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE
 * OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
 * OF THE POSSIBILITY OF SUCH DAMAGE.
 */

package dmsl.crowdspace.airplace;

import android.app.AlertDialog;
import android.app.ListActivity;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.net.Uri;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.ViewGroup;
import android.widget.*;

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

import dmsl.crowdspaceSidemenu.MainActivity;
import dmsl.crowdspaceSidemenu.R;

/**
 * Custom file browser for selecting Image, folder, file through the preferences
 *
 * @author KIOS Research Center and Data Management Systems Lab, University of
 *         Cyprus
 *
 */
public class AndroidFileBrowserTracker extends ListActivity implements
                OnClickListener {

        // Enum for the Display Mode
        private enum DISPLAYMODE {
                ABSOLUTE, RELATIVE;
        }

        private final DISPLAYMODE displayMode = DISPLAYMODE.ABSOLUTE;
        /** All the entries that will be shown for a directory */
        private List<String> directoryEntries = new ArrayList<String>();
        /** The current directory */
        private File currentDirectory = null;
        /** The home directory */
        private final File homeDirectory = new File("/");
        /** File chosen */
        private File file = null;
        /** Current working directory */
        private TextView pwd = null;
        /** Button for selecting file or folder */
        private Button select_file_folder = null;
        private int selectFolder = 1;
        /** Path of selected folder */
        private String folder_path = null;
        /** Path of selected file */
        private String file_path = null;
        /** Preference attributes */
        private SharedPreferences sharedPreferences = null;

        /** Layout file browser for preferences */
        @Override
        public void onCreate(Bundle icicle) {

                super.onCreate(icicle);
                setContentView(R.layout.main_choose_file_or_directory);

                pwd = (TextView) findViewById(R.id.pwd);
                select_file_folder = (Button) findViewById(R.id.select_file_folder);

                select_file_folder.setOnClickListener(this);

                // Get flags for browsing and indoor or outdoor
                Bundle extras = getIntent().getExtras();
                if (extras != null) {
                        selectFolder = extras.getInt("to_Browse");
                }

                sharedPreferences = getSharedPreferences(
```

```
                                MainActivity.SHARED_PREFS_INDOOR, MODE_PRIVATE
);

                // Get folder path selected before
                folder_path = sharedPreferences.getString("folder_browser", ""
).trim();

                // Get file of radiomap selected before

                switch (selectFolder) {
                case 1:
                        select_file_folder.setText("Save in this folder");
                        break;

                case 2:

                        file_path = sharedPreferences.getString("radiomap_file
", "").trim();
                        select_file_folder.setText("Use radiomap");

                        if (file_path != null && !file_path.equals("")
                                        && new File(file_path).canRead())
                                pwd.setText(file_path);

                        break;

                case 3:

                        file_path = sharedPreferences.getString("test_data_fil
e", "")
                                        .trim();
                        select_file_folder.setText("Use test data");

                        if (file_path != null && !file_path.equals("")
                                        && new File(file_path).canRead())
                                pwd.setText(file_path);

                        break;
                }

                if (folder_path == null || folder_path.equals("")) {
                        browseToHome();
                } else {
                        currentDirectory = new File(makePath(folder_path));
                        browseTo(currentDirectory);
                }
        }

        /**
         * Sets home directory (/) as the current directory
         *
         * */
        private void browseToHome() {
                currentDirectory = new File("/");
                browseTo(homeDirectory);
        }

        /**
         * Make a path if it exists, can be read and is a directory
         *
         * @param path
         *            the given path to check
         *
         * @return the constructed path
         * */
        private String makePath(String path) {

                if (path.length() == 0) {
                        return "/";
                }

                File check = new File(path);
                if (!check.exists() || !check.canRead() || !check.isDirectory(
)) {
                        return "/";
                }

                if (check.isDirectory())
                        return path;

                return "/";
        }

        /**
         * Browse to given directory and set it as the current directory
         *
         * @param aDirectory
         */
        private void browseTo(final File aDirectory) {

                if (aDirectory.isDirectory()) {
                        this.currentDirectory = aDirectory;
                        fill(aDirectory.listFiles());

                        if (selectFolder == 1)
                                pwd.setText(currentDirectory.getAbsolutePath()
);

                }
        }

        /**
         * Fill the directory entries with the files of the directory plus the
         * home-directory entry and the go-one-level-up entry
         *
         * @param files
         */
        private void fill(File[] files) {

                // Clear list
                this.directoryEntries.clear();

                // Add the "~" for home directory
                this.directoryEntries.add(getString(R.string.homeDir));

                // And the ".." for parent directory
                if (this.currentDirectory.getParent() != null)
                        this.directoryEntries.add(getString(R.string.parentDir
));

                switch (this.displayMode) {

                case ABSOLUTE:
```

```
                        for (File file : files) {
                                this.directoryEntries.add(file.getAbsolutePath
());
                        }
                        break;
                case RELATIVE:
                        int currentPathStringLenght = this.currentDirectory
                                        .getAbsolutePath().length();
                        for (File file : files) {
                                this.directoryEntries.add(file.getAbsolutePath
().substring(
                                                currentPathStringLenght));
                        }
                        break;
                }

                MyCustomAdapter directoryList = new MyCustomAdapter(this,
                                R.xml.file_row, this.directoryEntries);

                this.setListAdapter(directoryList);
        }

        /**
         * Actions to be taken when a directory entry is selected
         */
        @Override
        protected void onListItemClick(ListView l, View v, int position, long
id) {

                file = new File(directoryEntries.get(position));

                if (file.isDirectory()) {
                        if (file.canRead()) {
                                if (file.getName().equals(getString(R.string.p
arentDir))) {
                                        browseTo(currentDirectory.getParentFil
e());
                                } else if (file.getName().equals(getString(R.s
tring.homeDir))) {
                                        browseTo(homeDirectory);
                                } else
                                        browseTo(file);
                        } else {
                                showAlert("Read Permission Denied", "Warning",
 this);
                        }
                } else if (!file.isDirectory() && selectFolder == 1) {
                        showAlert("Not a directory", "Warning", this);
                } else {
                        pwd.setText(file.getAbsolutePath());
                }
                super.onListItemClick(l, v, position, id);
        }

        /**
         * Creates an Alert box
         *
         * @param message
         *            the message to show to user
         *
         * @param title
         *            the title of alert box
         *
         * @param ctx
         *            the context
         * */
        private void showAlert(String message, String title, Context ctx) {
                // Create a builder
                AlertDialog.Builder builder = new AlertDialog.Builder(ctx);
                builder.setTitle(title);

                // Add buttons and listener
                builder.setMessage(message).setCancelable(false)
                                .setPositiveButton("OK", new DialogInterface.O
nClickListener() {
                                        public void onClick(DialogInterface di
alog, int id) {
                                                dialog.cancel();
                                        }
                                });

                // Create the dialog
                AlertDialog ad = builder.create();

                // Show
                ad.show();
        }

        /**
         * Handle actions when completing file browser
         */
        public void onClick(View v) {

                switch (v.getId()) {

                case R.id.select_file_folder:

                        // Select a path mode
                        if (selectFolder == 1) {

                                if (currentDirectory.canWrite()) {
                                        Intent data = new Intent();
                                        data.setData(Uri.parse(currentDirector
y.getAbsolutePath()));
                                        setResult(RESULT_OK, data);
                                        AndroidFileBrowserTracker.this.finish(
);
                                } else
                                        showAlert("Write Permission Denied", "
Warning", this);

                        }
                        // Select a file mode
                        else {
                                file = new File(pwd.getText().toString());
                                if (file.exists()) {
                                        if (file.canRead()) {
                                                Intent data = new Intent();
                                                data.setData(Uri.parse(pwd.get
Text().toString()));
                                                setResult(RESULT_OK, data);
                                                AndroidFileBrowserTracker.this
```

```
.finish();
  330:                                        } else
  331:                                                showAlert("Read Permission Den
ied", "Warning", this);
  332:                                } else
  333:                                        showAlert("File not available", "Warni
ng", this);
  334:                        }
  335:                        break;
  336:                }
  337:        }
  338:
  339:        /**
  340:         * Exit file browser when back button is pressed
  341:         * */
  342:        @Override
  343:        public void onBackPressed() {
  344:                finish();
  345:        }
  346:
  347:        /**
  348:         * Class that sets icons in the file browser.
  349:         */
  350:        public class MyCustomAdapter extends ArrayAdapter<String> {
  351:
  352:                List<String> myList;
  353:
  354:                /**
  355:                 * Constructor
  356:                 *
  357:                 * @param context
  358:                 * @param textViewResourceId
  359:                 * @param objects
  360:                 */
  361:                public MyCustomAdapter(Context context, int textViewResourceId
,
  362:                                List<String> objects) {
  363:
  364:                        super(context, textViewResourceId, objects);
  365:                        myList = objects;
  366:                }
  367:
  368:                /**
  369:                 * Set icons.
  370:                 */
  371:                @Override
  372:                public View getView(int position, View convertView, ViewGroup
parent) {
  373:
  374:                        View row = convertView;
  375:                        // Check If Row Is Null
  376:                        if (row == null) {
  377:                                // Make New Layoutinflater
  378:                                LayoutInflater vi = (LayoutInflater) getSystem
Service(Context.LAYOUT_INFLATER_SERVICE);
  379:                                row = vi.inflate(R.xml.file_row, parent, false
);
  380:                        }
  381:                        TextView label = (TextView) row.findViewById(R.id.file
);
  382:                        String stringFile = myList.get(position).toString();
  383:
  384:                        // Change The Symbol Of The Home Directory
  385:                        if (stringFile.equals(getString(R.string.homeDir)))
  386:                                label.setText("~");
  387:                        else
  388:                                label.setText(stringFile);
  389:
  390:                        File file = new File(stringFile);
  391:                        ImageView icon = (ImageView) row.findViewById(R.id.ico
n);
  392:
  393:                        if (file.isDirectory())
  394:                                icon.setImageResource(R.drawable.directory);
  395:                        else
  396:                                icon.setImageResource(FindDrawable(stringFile)
);
  397:
  398:                        return row;
  399:                }
  400:
  401:                /**
  402:                 * Find the right icon for each file type
  403:                 *
  404:                 * @param file
  405:                 *            the file to get image source
  406:                 *
  407:                 * @return the id of image source
  408:                 * */
  409:                private int FindDrawable(String file) {
  410:                        if (file.endsWith(".txt")) {
  411:                                return R.drawable.txt;
  412:                        } else if (file.endsWith(".pdf")) {
  413:                                return R.drawable.pdf;
  414:                        } else if (file.endsWith(".exe")) {
  415:                                return R.drawable.exe;
  416:                        } else if (file.endsWith(".apk")) {
  417:                                return R.drawable.apk;
  418:                        } else if (file.endsWith(".png")) {
  419:                                return R.drawable.png;
  420:                        } else if (file.endsWith(".gif")) {
  421:                                return R.drawable.gif;
  422:                        } else if (file.endsWith(".jpg")) {
  423:                                return R.drawable.jpg;
  424:                        } else if (file.endsWith(".rar")) {
  425:                                return R.drawable.rar;
  426:                        } else if (file.endsWith(".zip")) {
  427:                                return R.drawable.zip;
  428:                        } else if (file.endsWith(".gz")) {
  429:                                return R.drawable.gz;
  430:                        } else if (file.endsWith(".mp3") || file.endsWith(".wa
v")
  431:                                        || file.endsWith(".amp")) {
  432:                                return R.drawable.sound;
  433:                        } else if (file.endsWith(".mp4") || file.endsWith(".av
i")
  434:                                        || file.endsWith(".flv")) {
  435:                                return R.drawable.video;
  436:                        } else
  437:                                return R.drawable.txt;
  438:                }
  439:        }
```

```
440: }
```

```
    1: /*
    2:  * Copyright (c) 2011, KIOS Research Center and Data Management Systems Lab,
    3:  * University of Cyprus. All rights reserved.
    4:  *
    5:  * Redistribution and use in source and binary forms, with or without modifica
tion,
    6:  * are permitted provided that the following conditions are met:
    7:  *
    8:  *    * Redistributions of source code must retain the above copyright notice,
 this
    9:  *      list of conditions and the following disclaimer.
   10:  *    * Redistributions in binary form must reproduce the above copyright noti
ce,
   11:  *      this list of conditions and the following disclaimer in the documentat
ion
   12:  *      and/or other materials provided with the distribution.
   13:  *    * Neither the name of the Sony Ericsson Mobile Communication AB nor the
names
   14:  *      of its contributors may be used to endorse or promote products derived
 from
   15:  *      this software without specific prior written permission.
   16:  *
   17:  * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 AND
   18:  * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLI
ED
   19:  * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISC
LAIMED.
   20:  * IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DI
RECT,
   21:  * INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDI
NG,
   22:  * BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF US
E,
   23:  * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEOR
Y OF
   24:  * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIG
ENCE
   25:  * OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF A
DVISED
   26:  * OF THE POSSIBILITY OF SUCH DAMAGE.
   27:  */
   28:
   29: package dmsl.crowdspace.airplace;
   30:
   31: import android.content.Context;
   32: import android.content.IntentFilter;
   33: import android.net.wifi.ScanResult;
   34: import android.net.wifi.WifiManager;
   35: import android.widget.TextView;
   36: import android.widget.Toast;
   37:
   38: import java.util.List;
   39: import java.util.Timer;
   40: import java.util.TimerTask;
   41:
   42: /**
   43:  * Custom WiFi manager used to scan and get scan results.
   44:  *
   45:  * @author KIOS Research Center and Data Management Systems Lab, University of
   46:  *         Cyprus
   47:  *
   48:  */
   49: public class SimpleWifiManager {
   50:
   51:         /** WiFi manager used to scan and get scan results */
   52:         private final WifiManager mainWifi;
   53:
   54:         /**
   55:          * Intent with the SCAN_RESULTS_AVAILABLE_ACTION action will be broadc
ast to
   56:          * asynchronously announce that the scan is complete and results are
   57:          * available.
   58:          */
   59:         private final IntentFilter wifiFilter;
   60:
   61:         /** Timer to perform new scheduled scans */
   62:         private final Timer timer;
   63:
   64:         /** Task to perform a scan */
   65:         private TimerTask WifiTask;
   66:
   67:         /** Application context */
   68:         private final Context mContext;
   69:
   70:         /** Text View to show the scan results */
   71:         private TextView scanResults;
   72:
   73:         /** If Scanning, true or false */
   74:         private Boolean isScanning;
   75:
   76:         /**
   77:          * Constructor
   78:          *
   79:          * @param context
   80:          *            Application context
   81:          */
   82:         public SimpleWifiManager(Context context) {
   83:                 mContext = context;
   84:                 isScanning = new Boolean(false);
   85:                 mainWifi = (WifiManager) mContext
   86:                                 .getSystemService(Context.WIFI_SERVICE);
   87:                 wifiFilter = new IntentFilter(WifiManager.SCAN_RESULTS_AVAILAB
LE_ACTION);
   88:                 timer = new Timer();
   89:         }
   90:
   91:         /**
   92:          * Sets the TextView to show scan results
   93:          *
   94:          * @param scanResults
   95:          *            TextView to set for scan results
   96:          * */
   97:         public void setScanResultsTextView(TextView scanResults) {
   98:                 this.scanResults = scanResults;
   99:         }
  100:
  101:         /**
  102:          * @return if the WiFi manager performs a scan
  103:          * */
  104:         public Boolean getIsScanning() {
  105:                 return isScanning;
```

```
106:         }
107:
108:         /**
109:          * Sets isScanning
110:          *
111:          * @param isScanning
112:          *            Boolean to set for isScanning
113:          * */
114:         public void setIsScanning(Boolean isScanning) {
115:
116:                 synchronized (isScanning) {
117:                         this.isScanning = isScanning;
118:                 }
119:         }
120:
121:         /**
122:          * @return the results of the current scan
123:          * */
124:         public List<ScanResult> getScanResults() {
125:                 return mainWifi.getScanResults();
126:         }
127:
128:         /**
129:          * Starts the Access Points Scanning
130:          *
131:          * @param receiverWifi
132:          *            WifiReceiver to register with WiFi manager
133:          * @param samples_interval
134:          *            Interval used to perform a new scan
135:          * */
136:         public void startScan(WifiReceiver receiverWifi, String samples_interv
al) {
137:
138:                 synchronized (isScanning) {
139:                         if (isScanning) {
140:                                 return;
141:                         }
142:                         isScanning = true;
143:                 }
144:
145:                 if (samples_interval.equals("n/a") || samples_interval.equals(
""")) {
146:                         toastPrint(
147:                                         "Samples interval not specified\nGo to
 Menu->Preferences->Sampling Settings",
148:                                         Toast.LENGTH_LONG);
149:
150:                         synchronized (isScanning) {
151:                                 isScanning = false;
152:                         }
153:                         return;
154:                 }
155:
156:                 toastPrint("Start scanning for near Access Points...",
157:                                 Toast.LENGTH_SHORT);
158:
159:                 enableWifi();
160:
161:                 mContext.registerReceiver(receiverWifi, wifiFilter);
162:
163:                 if (WifiTask != null) {
164:                         WifiTask.cancel();
165:                 }
166:
167:                 if (timer != null) {
168:                         timer.purge();
169:                 }
170:
171:                 WifiTask = new TimerTask() {
172:
173:                         @Override
174:                         public void run() {
175:                                 mainWifi.startScan();
176:                         }
177:                 };
178:
179:                 timer.schedule(WifiTask, 0, Long.parseLong(samples_interval));
180:
181:         }
182:
183:         /**
184:          * Stops the Access Points Scanning
185:          *
186:          * @param receiverWifi
187:          *            WifiReceiver to unregister
188:          * */
189:         public void stopScan(WifiReceiver receiverWifi) {
190:
191:                 synchronized (isScanning) {
192:                         if (!isScanning) {
193:                                 return;
194:                         }
195:                         isScanning = false;
196:                 }
197:
198:                 disableWifi();
199:
200:                 mContext.unregisterReceiver(receiverWifi);
201:                 toastPrint("Scanning Stopped", Toast.LENGTH_SHORT);
202:
203:                 if (scanResults != null)
204:                         scanResults.setText("AP detected: " + 0);
205:
206:         }
207:
208:         /**
209:          * Enables WiFi
210:          * */
211:         private void enableWifi() {
212:                 if (!mainWifi.isWifiEnabled())
213:                         if (mainWifi.getWifiState() != WifiManager.WIFI_STATE_
ENABLING)
214:                                 mainWifi.setWifiEnabled(true);
215:         }
216:
217:         /**
218:          * Disables WiFi
219:          * */
220:         public void disableWifi() {
221:                 if (mainWifi.isWifiEnabled())
222:                         if (mainWifi.getWifiState() != WifiManager.WIFI_STATE_
DISABLING)
```

```
223:                                mainWifi.setWifiEnabled(false);
224:        }
225:
226:        /**
227:         * Prints toast message to user
228:         * */
229:        protected void toastPrint(String textMSG, int duration) {
230:                Toast.makeText(this.mContext, textMSG, duration).show();
231:        }
232:
233: }
```

```java
package dmsl.ItemScan;

import java.io.BufferedInputStream;
import java.io.DataInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.ArrayList;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import dmsl.crowdspaceSidemenu.MainActivity;
import dmsl.crowdspaceSidemenu.MyJavaScriptInterface;

import android.util.Log;

/**
 *
 * @author Julia Metochi
 *
 *         This class implements methods that retrieve information from the
 *         Google Shopping API via an HTTP request and by parsing a JSON Objec
t.
 *
 */
public class GoogleShoppingAPIHelperClass {

        private String SHOPPING_API_KEY = "AIzaSyB0hxkB5QPcRgJ5OokCeSOa154CnuJ
YKBs";

        /**
         * Sends an HTTP request to Google API for Shopping and retrieves a JS
ON
         * String of a product, based on the barcode number given in the searc
h
         * criteria.
         *
         * @param barcodeContents
         *            The product's barcode which we use to find that product.
         * @return The JSON String that holds information about the product.
         */
        private String getJsonStringFromGoogleShopping(String barcodeContents)
 {
                URL u;
                InputStream is = null;
                DataInputStream dis = null;
                String s;
                StringBuilder sb = new StringBuilder();
                String jsonString = null;
                try {
                        u = new URL(
                                        "https://www.googleapis.com/shopping/s
earch/v1/public/products?key="
                                                        + SHOPPING_API_KEY + "
&country=US&restrictBy=gtin:"
                                                        + barcodeContents + "&
startIndex=1&maxResults=1");
                        is = u.openStream();
                        dis = new DataInputStream(new BufferedInputStream(is))
;
                        while ((s = dis.readLine()) != null) {
                                sb = sb.append(s + "\n");
                        }
                } catch (MalformedURLException mue) {
                        mue.printStackTrace();
                } catch (IOException ioe) {
                        ioe.printStackTrace();
                } finally {
                        if (is != null) {
                                try {
                                        is.close();
                                } catch (IOException ioe) {
                                        ioe.printStackTrace();
                                }
                        }
                }

                jsonString = sb.toString();
                return jsonString;
        }

        /*************************************************************************
*************************************************************/

        /**
         * This method receives a JSON String and gets any information we want
 from
         * it (e.g. Title, price, description, image of the product etc).
         *
         * @param barcodeContents
         *            The unique Global Trade Item Number of the product.
         * @return The title of the product we are searching for.
         */
        public ArrayList<String> getGoogleShoppingInfo(String barcodeContents)
 {

                String jsonString = getJsonStringFromGoogleShopping(barcodeCon
tents);
                String productTitle = null;
                String productBrand = null;
                String img = null;
                float productPrice;
                String pr = null;
                ProductObject probj;
                ArrayList<String> itemInfo = new ArrayList<String>();

                if (jsonString != null) {
                        try {

                                JSONObject jsonObject = new JSONObject(jsonStr
ing);

                                JSONArray itemsArray = jsonObject.getJSONArray
("items");
                                JSONObject itemObject = itemsArray.getJSONObje
ct(0);
                                JSONObject productObject = itemObject.getJSONO
bject("product");
                                if (itemObject.has("product")) {
```

```
  106:
  107:                                        if (productObject.has("title")) {
  108:                                                productTitle = productObject.g
etString("title");
  109:
  110:                                        } else
  111:                                                productTitle = null;
  112:                                        itemInfo.add(productTitle);
  113:
  114:                                        if (productObject.has("brand")) {
  115:                                                productBrand = productObject.g
etString("brand");
  116:
  117:                                        } else
  118:                                                productBrand = null;
  119:                                        itemInfo.add(productBrand);
  120:
  121:                                        JSONArray inventories = productObject
  122:                                                        .getJSONArray("invento
ries");
  123:                                        JSONObject inventoryObject = inventori
es.getJSONObject(0);
  124:                                        if (inventoryObject.has("price")) {
  125:                                                pr = inventoryObject.getString
("price");
  126:
  127:                                        } else
  128:                                                pr = null;
  129:                                        itemInfo.add(pr);
  130:
  131:                                        JSONArray image = productObject.getJSO
NArray("images");
  132:                                        JSONObject imageObject = image.getJSON
Object(0);
  133:                                        if (imageObject.has("link")) {
  134:                                                img = imageObject.getString("l
ink");
  135:
  136:                                        } else
  137:                                                img = null;
  138:                                        itemInfo.add(img);
  139:                                } else
  140:                                        return null;
  141:
  142:                                return itemInfo;
  143:
  144:                        } catch (JSONException je) {
  145:                                je.printStackTrace();
  146:                        }
  147:                }
  148:                return null;
  149:        }
  150:
  151:        /**********************************************************************
*************************************************************/
  152:
  153:        /**
  154:         * Search couchdb to check if the item exists
  155:         *
  156:         * @param jsonString
  157:         *            item info in json string
  158:         * @return
  159:         */
  160:        public String checkIfExistsInCouch(String jsonString) {
  161:
  162:                String msg = null;
  163:
  164:                if (jsonString != null) {
  165:
  166:                        JSONObject jsonObject = null;
  167:                        try {
  168:                                jsonObject = new JSONObject(jsonString);
  169:                        } catch (JSONException e1) {
  170:                                e1.printStackTrace();
  171:                        }
  172:
  173:                        try {
  174:
  175:                                JSONObject statusObj = jsonObject.optJSONObjec
t("0");
  176:
  177:                                msg = statusObj.getString("status");
  178:
  179:                        } catch (JSONException e) {
  180:                                e.printStackTrace();
  181:                        }
  182:
  183:                }
  184:
  185:                return msg;
  186:        }
  187:
  188:        /**********************************************************************
*************************************************************/
  189:
  190:        /**
  191:         * Retrun the information of the item if it is found in couchDB
  192:         *
  193:         * @param jsonString
  194:         *            item info i json string
  195:         * @return
  196:         */
  197:        public ProductObject productFoundInCouch(String jsonString) {
  198:
  199:                String barcode, title, brand, price, pointX, pointY, image, fl
oor, docId, docRev;
  200:                ProductObject pr = null, pr2 = null;
  201:
  202:                if (jsonString != null) {
  203:
  204:                        JSONObject jsonObject = null;
  205:                        try {
  206:                                jsonObject = new JSONObject(jsonString);
  207:                        } catch (JSONException e1) {
  208:                                e1.printStackTrace();
  209:                        }
  210:
  211:                        try {
  212:
  213:                                JSONObject Obj = jsonObject.optJSONObject("0")
;
  214:                                barcode = Obj.getString("barcode");
```

```
  215:                                title = Obj.getString("title");
  216:                                brand = Obj.getString("brand");
  217:                                price = Obj.getString("price");
  218:                                pointX = Obj.getString("pointX");
  219:                                pointY = Obj.getString("pointY");
  220:                                image = Obj.getString("image");
  221:                                floor = Obj.getString("floor");
  222:                                docId = Obj.getString("docId");
  223:                                docRev = Obj.getString("docRev");
  224:                                pr = new ProductObject(title, brand, Float.par
seFloat(price),
  225:                                                       barcode, Double.parseDouble(po
intX),
  226:                                                       Double.parseDouble(pointY), In
teger.parseInt(floor),
  227:                                                       image, docId, docRev);
  228:
  229:                        } catch (JSONException e) {
  230:                                e.printStackTrace();
  231:                        }
  232:
  233:                }
  234:
  235:                return pr;
  236:        }
  237:
  238: }
```

```
    1: /*
    2:  * Copyright (c) 2011, KIOS Research Center and Data Management Systems Lab,
    3:  * University of Cyprus. All rights reserved.
    4:  *
    5:  * Redistribution and use in source and binary forms, with or without modifica
tion,
    6:  * are permitted provided that the following conditions are met:
    7:  *
    8:  *    * Redistributions of source code must retain the above copyright notice,
 this
    9:  *      list of conditions and the following disclaimer.
   10:  *    * Redistributions in binary form must reproduce the above copyright noti
ce,
   11:  *      this list of conditions and the following disclaimer in the documentat
ion
   12:  *      and/or other materials provided with the distribution.
   13:  *    * Neither the name of the Sony Ericsson Mobile Communication AB nor the
names
   14:  *      of its contributors may be used to endorse or promote products derived
 from
   15:  *      this software without specific prior written permission.
   16:  *
   17:  * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 AND
   18:  * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLI
ED
   19:  * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISC
LAIMED.
   20:  * IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DI
RECT,
   21:  * INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDI
NG,
   22:  * BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF US
E,
   23:  * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEOR
Y OF
   24:  * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIG
ENCE
   25:  * OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF A
DVISED
   26:  * OF THE POSSIBILITY OF SUCH DAMAGE.
   27:  */
   28:
   29: package dmsl.crowdspace.airplace;
   30:
   31: import android.content.BroadcastReceiver;
   32: import android.content.Context;
   33: import android.content.Intent;
   34:
   35: /**
   36:  * Custom BroadcastReceiver to read RSS
   37:  *
   38:  * @author KIOS Research Center and Data Management Systems Lab, University of
   39:  *         Cyprus
   40:  *
   41:  */
   42: public abstract class WifiReceiver extends BroadcastReceiver {
   43:
   44:         /**
   45:          * On receive do nothing.
   46:          */
   47:         @Override
   48:         public void onReceive(Context paramContext, Intent paramIntent) {
   49:
   50:         }
   51:
   52: }
```

```
    1: package dmsl.crowdspace.airplace;
    2:
    3: /*
    4:  * Copyright (c) 2011, KIOS Research Center and Data Management Systems Lab,
    5:  * University of Cyprus. All rights reserved.
    6:  *
    7:  * Redistribution and use in source and binary forms, with or without modifica
tion,
    8:  * are permitted provided that the following conditions are met:
    9:  *
   10:  *    * Redistributions of source code must retain the above copyright notice,
 this
   11:  *      list of conditions and the following disclaimer.
   12:  *    * Redistributions in binary form must reproduce the above copyright noti
ce,
   13:  *      this list of conditions and the following disclaimer in the documentat
ion
   14:  *      and/or other materials provided with the distribution.
   15:  *    * Neither the name of the Sony Ericsson Mobile Communication AB nor the
names
   16:  *      of its contributors may be used to endorse or promote products derived
 from
   17:  *      this software without specific prior written permission.
   18:  *
   19:  * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 AND
   20:  * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLI
ED
   21:  * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISC
LAIMED.
   22:  * IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DI
RECT,
   23:  * INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDI
NG,
   24:  * BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF US
E,
   25:  * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEOR
Y OF
   26:  * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIG
ENCE
   27:  * OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF A
DVISED
   28:  * OF THE POSSIBILITY OF SUCH DAMAGE.
   29:  */
   30:
   31: import dmsl.crowdspaceSidemenu.R;
   32: import android.app.Activity;
   33: import android.content.Intent;
   34: import android.content.SharedPreferences;
   35: import android.content.SharedPreferences.OnSharedPreferenceChangeListener;
   36: import android.database.Cursor;
   37: import android.net.Uri;
   38: import android.os.Bundle;
   39: import android.preference.Preference;
   40: import android.preference.Preference.OnPreferenceChangeListener;
   41: import android.preference.PreferenceActivity;
   42: import android.preference.Preference.OnPreferenceClickListener;
   43: import android.provider.MediaStore.MediaColumns;
   44:
   45: public class Prefs_logger extends PreferenceActivity implements
   46:                 OnSharedPreferenceChangeListener {
   47:
   48:         private static final int SELECT_IMAGE = 7;
   49:         private static final int SELECT_PATH = 8;
   50:         private static final int SELECT_FILE = 9;
   51:
   52:         @Override
   53:         protected void onCreate(Bundle savedInstanceState) {
   54:
   55:                 super.onCreate(savedInstanceState);
   56:
   57:                 getPreferenceManager().setSharedPreferencesName("Indoor_Prefer
ences");
   58:
   59:                 addPreferencesFromResource(R.xml.preferences_logger);
   60:                 getPreferenceManager().getSharedPreferences()
   61:                                 .registerOnSharedPreferenceChangeListener(this
);
   62:
   63:                 getPreferenceManager().findPreference("folder_browser")
   64:                                 .setOnPreferenceClickListener(new OnPreference
ClickListener() {
   65:
   66:                                         public boolean onPreferenceClick(Prefe
rence preference) {
   67:
   68:                                                 Intent i = new Intent(getBaseC
ontext(),
   69:                                                                 AndroidFileBro
wserLogger.class);
   70:
   71:                                                 Bundle extras = new Bundle();
   72:                                                 extras.putBoolean("to_Browse",
 true);
   73:
   74:                                                 i.putExtras(extras);
   75:
   76:                                                 startActivityForResult(i, SELE
CT_PATH);
   77:                                                 return true;
   78:                                         }
   79:                                 });
   80:
   81:         }
   82:
   83:         @Override
   84:         public void onActivityResult(int requestCode, int resultCode, Intent d
ata) {
   85:
   86:                 super.onActivityResult(requestCode, resultCode, data);
   87:
   88:                 SharedPreferences customSharedPreference;
   89:
   90:                 customSharedPreference = getSharedPreferences("Indoor_Preferen
ces",
   91:                                 MODE_PRIVATE);
   92:
   93:                 switch (requestCode) {
   94:
   95:                 case SELECT_IMAGE:
   96:                         if (resultCode == Activity.RESULT_OK) {
   97:                                 Uri selectedImage = data.getData();
```

```
                                String RealPath;
                                SharedPreferences.Editor editor = customShared
Preference.edit();
                                RealPath = getRealPathFromURI(selectedImage);
                                editor.putString("image_custom", RealPath);
                                editor.commit();
                        }
                        break;
                case SELECT_PATH:
                        if (resultCode == Activity.RESULT_OK) {
                                Uri selectedFolder = data.getData();
                                String path = selectedFolder.toString();
                                SharedPreferences.Editor editor = customShared
Preference.edit();
                                editor.putString("folder_browser", path);
                                editor.commit();
                        }
                        break;
                case SELECT_FILE:
                        if (resultCode == Activity.RESULT_OK) {
                                Uri selectedFile = data.getData();
                                String file = selectedFile.toString();
                                SharedPreferences.Editor editor = customShared
Preference.edit();
                                editor.putString("upload_file", file);
                                editor.commit();
                        }
                        break;
                }
        }

        public String getRealPathFromURI(Uri contentUri) {
                String[] proj = { MediaColumns.DATA };
                Cursor cursor = managedQuery(contentUri, proj, null, null, nul
l);
                int column_index = cursor.getColumnIndexOrThrow(MediaColumns.D
ATA);
                cursor.moveToFirst();
                return cursor.getString(column_index);
        }

        @Override
        protected void onResume() {
                super.onResume();
                // Set up a listener whenever a key changes
                getPreferenceScreen().getSharedPreferences()
                                .registerOnSharedPreferenceChangeListener(this
);
        }

        @Override
        protected void onDestroy() {
                // Unregister the listener whenever a key changes
                getPreferenceManager().getSharedPreferences()
                                .unregisterOnSharedPreferenceChangeListener(th
is);
                super.onDestroy();
        }

        @Override
        protected void onPause() {
                super.onPause();
                // Unregister the listener whenever a key changes
                getPreferenceScreen().getSharedPreferences()
                                .unregisterOnSharedPreferenceChangeListener(th
is);
        }

        public void onSharedPreferenceChanged(SharedPreferences sharedPreferen
ces,
                        String key) {
        }

}
```

```
    1: package dmsl.ItemScan;
    2:
    3: import dmsl.crowdspaceSidemenu.R;
    4: import android.app.Activity;
    5: import android.content.Intent;
    6: import android.os.Bundle;
    7: import android.view.View;
    8: import android.widget.Button;
    9: import android.widget.EditText;
   10: import android.widget.TextView;
   11: import android.widget.Toast;
   12:
   13: /**
   14:  * This activity is called when the user wishes to manually input a barcode.
   15:  *
   16:  * @author Julia Metochi
   17:  *
   18:  */
   19: public class AskForManualEntryActivity extends Activity {
   20:
   21:         public static final int MAX_BARCODE_LENGTH = 30;
   22:
   23:         @Override
   24:         protected void onCreate(Bundle savedInstanceState) {
   25:                 super.onCreate(savedInstanceState);
   26:                 setContentView(R.layout.prompt);
   27:
   28:                 this.setTitle("Manual Entry");
   29:
   30:                 final EditText etTitle = (EditText) findViewById(R.id.etTitle)
;
   31:                 EditText etPrice = (EditText) findViewById(R.id.etPrice);
   32:                 EditText etBrand = (EditText) findViewById(R.id.etBrand);
   33:
   34:                 TextView tvTitle = (TextView) findViewById(R.id.title);
   35:                 TextView tvPrice = (TextView) findViewById(R.id.Price);
   36:                 TextView tvBrand = (TextView) findViewById(R.id.Brand);
   37:
   38:                 tvTitle.setText("Barcode:");
   39:                 etPrice.setVisibility(View.GONE);
   40:                 tvPrice.setVisibility(View.GONE);
   41:                 etBrand.setVisibility(View.GONE);
   42:                 tvBrand.setVisibility(View.GONE);
   43:
   44:                 etPrice.requestFocus();
   45:
   46:                 Button bOk = (Button) findViewById(R.id.ok);
   47:                 Button bCancel = (Button) findViewById(R.id.cancel);
   48:
   49:                 bOk.setOnClickListener(new View.OnClickListener() {
   50:
   51:                         @Override
   52:                         public void onClick(View v) {
   53:
   54:                                 String strBarcode = etTitle.getText().toString
();
   55:
   56:                                 if (strBarcode == null || strBarcode.length()
== 0) {
   57:                                         Toast.makeText(getApplicationContext()
,
   58:                                                 "Please enter a barcod
e.", Toast.LENGTH_SHORT)
   59:                                                 .show();
   60:                                 } else if (strBarcode.length() > MAX_BARCODE_L
ENGTH) {
   61:                                         Toast.makeText(
   62:                                                 getApplicationContext(
),
   63:                                                 "The barcode is too lo
ng. Maximum length is "
   64:                                                         + MAX_
BARCODE_LENGTH + ".",
   65:                                                 Toast.LENGTH_SHORT).sh
ow();
   66:                                 } else {
   67:                                         Intent i = new Intent();
   68:                                         Bundle b = new Bundle();
   69:                                         // Data to be passed to the main activ
ity is put in the
   70:                                         // bundle.
   71:                                         b.putString("strBarcode", strBarcode);
   72:                                         i.putExtras(b);
   73:                                         setResult(RESULT_OK, i);
   74:                                         finish();
   75:                                 }
   76:
   77:                         }
   78:                 });
   79:
   80:                 bCancel.setOnClickListener(new View.OnClickListener() {
   81:
   82:                         @Override
   83:                         public void onClick(View v) {
   84:                                 setResult(RESULT_CANCELED);
   85:                                 finish();
   86:                         }
   87:                 });
   88:         }
   89:
   90: }
```

```
    1: package dmsl.FloorMode;
    2:
    3: import java.util.ArrayList;
    4:
    5: import com.cy.wifi.algorithms.LogRecord;
    6:
    7: public class FloorProcedure {
    8:         private ArrayList<ArrayList<RMHelp>> arrayRMHelp = null;
    9:
   10:         /**
   11:          * Take an arrayList of arrayLists. Each arraList is the content of th
e help
   12:          * files, of the two floors that I want to find me!
   13:          *
   14:          * @param arrayList
   15:          *            of arrayLists of RMHelp lists
   16:          */
   17:         public FloorProcedure(ArrayList<ArrayList<RMHelp>> array) {
   18:                 this.arrayRMHelp = array;
   19:         }
   20:
   21:         /**
   22:          * Find the nearest mac Address of the current live received AP's. Aft
er
   23:          * that decide in which floor is this AP and change to this floor.
   24:          *
   25:          * @param the
   26:          *            current results of the detected AP's
   27:          * @return the result of the two check in .floor files
   28:          */
   29:         public String findTheMAC(ArrayList<LogRecord> lr) {
   30:                 ArrayList<LogRecordFloor> lrf = new ArrayList<LogRecordFloor>(
);
   31:                 Double maxRSS = 0.0;
   32:                 Double minRSS = 0.0;
   33:                 String strRe = "";
   34:                 boolean flagEq = true;
   35:
   36:                 /**
   37:                  * Take the lr type LogRecord and convert it to LogRecordFloor
   38:                  */
   39:
   40:                 for (LogRecord current : lr) {
   41:                         lrf.add(new LogRecordFloor(current.getBssid(), current
.getRss(), 2));
   42:                 }
   43:
   44:                 // Find the strongest Access Point
   45:                 boolean flag = true;
   46:                 LogRecordFloor maxMac = null;
   47:                 for (int i = 0; i < lrf.size(); ++i) {
   48:                         if (flag) {
   49:                                 maxMac = lrf.get(i);
   50:                                 flag = false;
   51:                         }
   52:                         if (maxMac.getRss() < lrf.get(i).getRss()) {
   53:                                 maxMac = lrf.get(i);
   54:                         }
   55:                 }
   56:
   57:                 /*
   58:                  * Find where it exists the maxMac
   59:                  */
   60:                 for (int i = 0; i < arrayRMHelp.size(); ++i) {
   61:                         for (int j = 0; j < arrayRMHelp.get(i).size(); ++j) {
   62:                                 ArrayList<RMHelp> temp = arrayRMHelp.get(i);
   63:
   64:                                 if (arrayRMHelp.get(i).get(j).getMacAddress()
   65:                                                 .equals(maxMac.getBssid())) {
   66:                                         temp.get(j).setVisible(true);
   67:                                 }
   68:                         }
   69:                 }
   70:
   71:                 /* Check the case of existing the maxMac in both .floor files
*/
   72:                 for (int j = 0; j < arrayRMHelp.get(0).size(); ++j) {
   73:                         RMHelp temp0 = arrayRMHelp.get(0).get(j);
   74:
   75:                         for (int z = 0; z < arrayRMHelp.get(1).size(); ++z) {
   76:                                 RMHelp temp1 = arrayRMHelp.get(1).get(z);
   77:
   78:                                 /**
   79:                                  * Check if one macAddress belongs to the two
floors
   80:                                  */
   81:                                 if (temp0.getMacAddress().equals(temp1.getMacA
ddress())) {
   82:                                         /**
   83:                                          * Find the macAddress from the curren
t results if exist
   84:                                          */
   85:                                         if (maxMac.getBssid().equals(temp0.get
MacAddress())) {
   86:                                                 /**
   87:                                                  * Decide in which floor you w
ill label the macAddress
   88:                                                  */
   89:                                                 if (temp0.getRss() > temp1.get
Rss()) {
   90:                                                         // lrf.get(i).setFloor
(false, 1);
   91:                                                         flagEq = false;
   92:                                                         maxRSS = temp0.getRss(
);
   93:                                                         minRSS = temp1.getRss(
);
   94:                                                         strRe = "1," + maxRSS.
intValue() + " 2,"
   95:                                                                         + minR
SS.intValue();
   96:                                                 } else {
   97:                                                         flagEq = false;
   98:                                                         maxRSS = temp1.getRss(
);
   99:                                                         minRSS = temp0.getRss(
);
  100:                                                         strRe = "1," + minRSS.
intValue() + " 2,"
  101:                                                                         + maxR
SS.intValue();
  102:                                                 }
```

```
  103:                                        }
  104:                                }
  105:                        }
  106:                }
  107:
  108:                if (flagEq) {
  109:                        for (int i = 0; i < arrayRMHelp.size(); ++i) {
  110:                                for (int j = 0; j < arrayRMHelp.get(i).size();
 ++j) {
  111:                                        ArrayList<RMHelp> temp = arrayRMHelp.g
et(i);
  112:                                        for (RMHelp current : temp) {
  113:                                                if (temp.get(j).isVisible()) {
  114:                                                        if (i == 0) {
  115:                                                                return "1," +
temp.get(j).getRss().intValue()
  116:
+ " 2," + (-800);
  117:                                                        } else if (i == 1) {
  118:                                                                return "1," +
(-800) + " 2,"
  119:
+ temp.get(j).getRss().intValue();
  120:                                                        }
  121:
  122:                                                }
  123:                                        }
  124:                                }
  125:                        }
  126:                }
  127:
  128:                if (strRe.equals("")) {
  129:                        return "1," + (-800) + " 2," + (-800);
  130:                }
  131:
  132:                return strRe;
  133:
  134:        }
  135: }
```

```java
package dmsl.crowdspaceSidemenu;

/**
 * @author Julia Metochi
 *
 */
public class Coordinates {
	private double lon, lat;

	/**
	 *
	 * @param lon
	 *            Set longitude
	 */
	public void setLon(double lon) {
		this.lon = lon;

	}

	/**
	 *
	 * @param lat
	 *            set latitude
	 */
	public void setLat(double lat) {
		this.lat = lat;

	}

	/**
	 * @return return longitude
	 */
	public double getLon() {
		return this.lon;
	}

	/**
	 *
	 * @return return latitude
	 */
	public double getLat() {
		return this.lat;
	}

}
```

```
    1: package dmsl.ItemScan;
    2:
    3: import java.text.NumberFormat;
    4:
    5: import android.content.Context;
    6:
    7: /**
    8:  * This class is an object which represents a product. This representation is
    9:  * used through out the program.
   10:  *
   11:  * @author Julia Metochi
   12:  *
   13:  */
   14: public class ProductObject {
   15:
   16:         private long id = 0;
   17:         private String title = null;
   18:         private String brand = null;
   19:         private float price = 0;
   20:         private String barcode = null;
   21:         private double pointX = 0.0;
   22:         private double pointY = 0.0;
   23:         private int floor;
   24:         private String image = null, docId = null, docRev = null;
   25:
   26:         private NumberFormat form = NumberFormat.getInstance();
   27:
   28:         /**
   29:          * Constructor
   30:          *
   31:          * @param t
   32:          *            Title
   33:          * @param p
   34:          *            Price
   35:          * @param b
   36:          *            Global Trade Item Number (Barcode number)
   37:          * @param la
   38:          *            Geo Latitude
   39:          * @param lo
   40:          *            Geo Longitude
   41:          */
   42:         public ProductObject(String t, String br, float p, String b, double x,
   43:                         double y, int floor, String image, String docId, Strin
g docRev) {
   44:
   45:                 setTitle(t);
   46:                 setBrand(br);
   47:                 setPrice(p);
   48:                 setBarcode(b);
   49:                 setPointX(x);
   50:                 setPointY(y);
   51:                 setFloor(floor);
   52:                 setImage(image);
   53:                 setDocId(docId);
   54:                 setDocRev(docRev);
   55:
   56:         }
   57:
   58:         /*************************************************************************
*************************************************************/
   59:
   60:         public void setDocId(String docId) {
   61:                 this.docId = docId;
   62:         }
   63:
   64:         /*************************************************************************
*************************************************************/
   65:
   66:         public void setDocRev(String docRev) {
   67:                 this.docRev = docRev;
   68:         }
   69:
   70:         /*************************************************************************
*************************************************************/
   71:
   72:         public void setId(long id) {
   73:                 this.id = id;
   74:         }
   75:
   76:         /*************************************************************************
*************************************************************/
   77:
   78:         public void setTitle(String title) {
   79:                 this.title = title;
   80:         }
   81:
   82:         /*************************************************************************
*************************************************************/
   83:
   84:         public void setBrand(String brand) {
   85:                 this.brand = brand;
   86:         }
   87:
   88:         /*************************************************************************
*************************************************************/
   89:
   90:         public void setPrice(float price) {
   91:                 this.price = price;
   92:         }
   93:
   94:         /*************************************************************************
*************************************************************/
   95:
   96:         public void setBarcode(String barcode) {
   97:                 this.barcode = barcode;
   98:         }
   99:
  100:         /*************************************************************************
*************************************************************/
  101:
  102:         public void setImage(String img) {
  103:                 this.image = img;
  104:         }
  105:
  106:         /*************************************************************************
*************************************************************/
  107:
  108:         public String getImage() {
  109:                 return this.image;
  110:         }
  111:
  112:         /*************************************************************************
```

```
**********************************************************/
  113:
  114:        public String getTitle() {
  115:                return this.title;
  116:        }
  117:
  118:        /*************************************************************************
**********************************************************/
  119:
  120:        public String getBrand() {
  121:                return this.brand;
  122:        }
  123:
  124:        /*************************************************************************
**********************************************************/
  125:
  126:        public String getBarcode() {
  127:                return this.barcode;
  128:        }
  129:
  130:        /*************************************************************************
**********************************************************/
  131:
  132:        public float getPrice() {
  133:                return this.price;
  134:        }
  135:
  136:        /*************************************************************************
**********************************************************/
  137:
  138:        public String getDocId() {
  139:                return this.docId;
  140:        }
  141:
  142:        /*************************************************************************
**********************************************************/
  143:
  144:        public String getDocRev() {
  145:                return this.docRev;
  146:        }
  147:
  148:        /*************************************************************************
**********************************************************/
  149:
  150:        public long getId() {
  151:                return this.id;
  152:        }
  153:
  154:        /*************************************************************************
**********************************************************/
  155:
  156:        public void changeId(long newId) {
  157:                this.id = newId;
  158:        }
  159:
  160:        /*************************************************************************
**********************************************************/
  161:
  162:        public void changeTitle(String newT) {
  163:                this.title = newT;
  164:        }
  165:
  166:        /*************************************************************************
**********************************************************/
  167:
  168:        public void changeBrand(String newBr) {
  169:                this.brand = newBr;
  170:        }
  171:
  172:        /*************************************************************************
**********************************************************/
  173:
  174:        public void changePrice(float newP) {
  175:                this.price = newP;
  176:        }
  177:
  178:        /*************************************************************************
**********************************************************/
  179:
  180:        public void setPointX(double x) {
  181:                this.pointX = x;
  182:        }
  183:
  184:        /*************************************************************************
**********************************************************/
  185:
  186:        public void setPointY(double y) {
  187:                this.pointY = y;
  188:        }
  189:
  190:        /*************************************************************************
**********************************************************/
  191:
  192:        public double getPointX() {
  193:                return this.pointX;
  194:        }
  195:
  196:        /*************************************************************************
**********************************************************/
  197:
  198:        public double getPointY() {
  199:                return this.pointY;
  200:        }
  201:
  202:        /*************************************************************************
**********************************************************/
  203:
  204:        public void setFloor(int floor) {
  205:                this.floor = floor;
  206:        }
  207:
  208:        /*************************************************************************
**********************************************************/
  209:
  210:        public int getFloor() {
  211:                return this.floor;
  212:        }
  213:
  214:        /*************************************************************************
**********************************************************/
  215:
  216: }
```

```
    1: /*
    2:  * Copyright (c) 2011, KIOS Research Center and Data Management Systems Lab,
    3:  * University of Cyprus. All rights reserved.
    4:  *
    5:  * Redistribution and use in source and binary forms, with or without modifica
tion,
    6:  * are permitted provided that the following conditions are met:
    7:  *
    8:  *    * Redistributions of source code must retain the above copyright notice,
 this
    9:  *      list of conditions and the following disclaimer.
   10:  *    * Redistributions in binary form must reproduce the above copyright noti
ce,
   11:  *      this list of conditions and the following disclaimer in the documentat
ion
   12:  *      and/or other materials provided with the distribution.
   13:  *    * Neither the name of the Sony Ericsson Mobile Communication AB nor the
names
   14:  *      of its contributors may be used to endorse or promote products derived
 from
   15:  *      this software without specific prior written permission.
   16:  *
   17:  * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 AND
   18:  * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLI
ED
   19:  * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISC
LAIMED.
   20:  * IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DI
RECT,
   21:  * INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDI
NG,
   22:  * BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF US
E,
   23:  * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEOR
Y OF
   24:  * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIG
ENCE
   25:  * OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF A
DVISED
   26:  * OF THE POSSIBILITY OF SUCH DAMAGE.
   27:  */
   28:
   29: package dmsl.crowdspace.airplace;
   30:
   31: import java.util.Observable;
   32:
   33: /**
   34:  * Observable class that verifies boolean value.
   35:  *
   36:  * @author KIOS Research Center and Data Management Systems Lab, University of
   37:  *         Cyprus
   38:  *
   39:  */
   40: public class BooleanObservable extends Observable {
   41:
   42:         private boolean isTracking;
   43:
   44:         public boolean get() {
   45:                 return isTracking;
   46:         }
   47:
   48:         public void setBoolean(boolean track) {
   49:
   50:                 isTracking = track;
   51:                 setChanged();
   52:         }
   53: }
```

```
    1: /*
    2:  * Copyright (c) 2011, KIOS Research Center and Data Management Systems Lab,
    3:  * University of Cyprus. All rights reserved.
    4:  *
    5:  * Redistribution and use in source and binary forms, with or without modifica
tion,
    6:  * are permitted provided that the following conditions are met:
    7:  *
    8:  *    * Redistributions of source code must retain the above copyright notice,
 this
    9:  *      list of conditions and the following disclaimer.
   10:  *    * Redistributions in binary form must reproduce the above copyright noti
ce,
   11:  *      this list of conditions and the following disclaimer in the documentat
ion
   12:  *      and/or other materials provided with the distribution.
   13:  *    * Neither the name of the Sony Ericsson Mobile Communication AB nor the
names
   14:  *      of its contributors may be used to endorse or promote products derived
 from
   15:  *      this software without specific prior written permission.
   16:  *
   17:  * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 AND
   18:  * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLI
ED
   19:  * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISC
LAIMED.
   20:  * IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DI
RECT,
   21:  * INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDI
NG,
   22:  * BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF US
E,
   23:  * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEOR
Y OF
   24:  * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIG
ENCE
   25:  * OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF A
DVISED
   26:  * OF THE POSSIBILITY OF SUCH DAMAGE.
   27:  */
   28:
   29: package dmsl.crowdspace.airplace;
   30:
   31: import android.graphics.drawable.Drawable;
   32:
   33: public class IconifiedText implements Comparable<IconifiedText> {
   34:
   35:         private String mText = "";
   36:         private Drawable mIcon;
   37:         private boolean mSelectable = true;
   38:
   39:         public IconifiedText(String text, Drawable bullet) {
   40:                 mIcon = bullet;
   41:                 mText = text;
   42:         }
   43:
   44:         public boolean isSelectable() {
   45:                 return mSelectable;
   46:         }
   47:
   48:         public void setSelectable(boolean selectable) {
   49:                 mSelectable = selectable;
   50:         }
   51:
   52:         public String getText() {
   53:                 return mText;
   54:         }
   55:
   56:         public void setText(String text) {
   57:                 mText = text;
   58:         }
   59:
   60:         public void setIcon(Drawable icon) {
   61:                 mIcon = icon;
   62:         }
   63:
   64:         public Drawable getIcon() {
   65:                 return mIcon;
   66:         }
   67:
   68:         /** Make IconifiedText comparable by its name */
   69:         public int compareTo(IconifiedText other) {
   70:                 if (this.mText != null)
   71:                         return this.mText.compareTo(other.getText());
   72:                 else
   73:                         throw new IllegalArgumentException();
   74:         }
   75: }
```

```
    1: /*
    2:  * Copyright (c) 2011, KIOS Research Center and Data Management Systems Lab,
    3:  * University of Cyprus. All rights reserved.
    4:  *
    5:  * Redistribution and use in source and binary forms, with or without modifica
tion,
    6:  * are permitted provided that the following conditions are met:
    7:  *
    8:  *    * Redistributions of source code must retain the above copyright notice,
 this
    9:  *      list of conditions and the following disclaimer.
   10:  *    * Redistributions in binary form must reproduce the above copyright noti
ce,
   11:  *      this list of conditions and the following disclaimer in the documentat
ion
   12:  *      and/or other materials provided with the distribution.
   13:  *    * Neither the name of the Sony Ericsson Mobile Communication AB nor the
names
   14:  *      of its contributors may be used to endorse or promote products derived
 from
   15:  *      this software without specific prior written permission.
   16:  *
   17:  * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 AND
   18:  * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLI
ED
   19:  * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISC
LAIMED.
   20:  * IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DI
RECT,
   21:  * INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDI
NG,
   22:  * BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF US
E,
   23:  * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEOR
Y OF
   24:  * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIG
ENCE
   25:  * OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF A
DVISED
   26:  * OF THE POSSIBILITY OF SUCH DAMAGE.
   27:  */
   28:
   29: package dmsl.crowdspace.airplace;
   30:
   31: import java.io.BufferedReader;
   32: import java.io.FileReader;
   33: import java.io.IOException;
   34: import java.io.InputStreamReader;
   35: import java.io.PrintWriter;
   36: import java.net.Socket;
   37:
   38: import android.app.ProgressDialog;
   39: import android.os.Handler;
   40: import android.os.Message;
   41:
   42: public class UploadingSettings extends Thread {
   43:
   44:         private String IP;
   45:         private String PORT;
   46:         private Socket socket = null;
   47:         private String filename;
   48:
   49:         private String errMsg;
   50:         private ProgressDialog dialog;
   51:
   52:         public UploadingSettings(String IP, String PORT, String filename,
   53:                         ProgressDialog dialog) {
   54:                 this.filename = filename;
   55:                 this.PORT = PORT;
   56:                 this.IP = IP;
   57:                 this.dialog = dialog;
   58:         }
   59:
   60:         public boolean connect() {
   61:                 try {
   62:                         socket = new Socket(IP, Integer.parseInt(PORT));
   63:                         return true;
   64:                 } catch (NumberFormatException e) {
   65:                         errMsg = "Specify IP/PORT to upload";
   66:                 } catch (Exception e) {
   67:                         errMsg = e.getMessage();
   68:                 }
   69:                 return false;
   70:         }
   71:
   72:         public String getErrMsg() {
   73:                 return errMsg;
   74:         }
   75:
   76:         public void run() {
   77:
   78:                 try {
   79:
   80:                         PrintWriter out = new PrintWriter(socket.getOutputStre
am(), true);
   81:                         BufferedReader in = new BufferedReader(new InputStream
Reader(
   82:                                         socket.getInputStream()));
   83:
   84:                         String inputLine, outputLine;
   85:
   86:                         inputLine = in.readLine();
   87:
   88:                         if (!inputLine.equalsIgnoreCase("+OK READY")) {
   89:                                 out.close();
   90:                                 in.close();
   91:                                 socket.close();
   92:                                 errMsg = "Server not available right now";
   93:                                 handler.sendEmptyMessage(0);
   94:                                 return;
   95:                         }
   96:
   97:                         outputLine = "UPLOAD rsslog";
   98:                         out.println(outputLine);
   99:
  100:                         inputLine = in.readLine();
  101:
  102:                         if (!inputLine.equalsIgnoreCase("+OK UPLOAD")) {
  103:                                 out.close();
  104:                                 in.close();
  105:                                 socket.close();
```

```
  106:                                errMsg = "Server not available right now";
  107:                                handler.sendEmptyMessage(0);
  108:                                return;
  109:                        }
  110:
  111:                        BufferedReader reader = new BufferedReader(new FileRea
der(filename));
  112:
  113:                        while ((outputLine = reader.readLine()) != null) {
  114:                                out.println(outputLine);
  115:                        }
  116:
  117:                        reader.close();
  118:                        out.close();
  119:                        in.close();
  120:                        socket.close();
  121:                        dialog.dismiss();
  122:                        return;
  123:
  124:                } catch (IOException e) {
  125:                        errMsg = e.getMessage();
  126:                        handler.sendEmptyMessage(0);
  127:                }
  128:        }
  129:
  130:        final Handler handler = new Handler() {
  131:                public void handleMessage(Message msg) {
  132:                        dialog.dismiss();
  133:                }
  134:        };
  135:
  136: }
```

```
    1: /*
    2:  * Copyright (c) 2011, KIOS Research Center and Data Management Systems Lab,
    3:  * University of Cyprus. All rights reserved.
    4:  *
    5:  * Redistribution and use in source and binary forms, with or without modifica
tion,
    6:  * are permitted provided that the following conditions are met:
    7:  *
    8:  *    * Redistributions of source code must retain the above copyright notice,
 this
    9:  *      list of conditions and the following disclaimer.
   10:  *    * Redistributions in binary form must reproduce the above copyright noti
ce,
   11:  *      this list of conditions and the following disclaimer in the documentat
ion
   12:  *      and/or other materials provided with the distribution.
   13:  *    * Neither the name of the Sony Ericsson Mobile Communication AB nor the
names
   14:  *      of its contributors may be used to endorse or promote products derived
 from
   15:  *      this software without specific prior written permission.
   16:  *
   17:  * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 AND
   18:  * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLI
ED
   19:  * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISC
LAIMED.
   20:  * IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DI
RECT,
   21:  * INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDI
NG,
   22:  * BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF US
E,
   23:  * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEOR
Y OF
   24:  * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIG
ENCE
   25:  * OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF A
DVISED
   26:  * OF THE POSSIBILITY OF SUCH DAMAGE.
   27:  */
   28:
   29: package dmsl.crowdspace.airplace;
   30:
   31: import java.io.File;
   32: import java.util.ArrayList;
   33: import java.util.List;
   34:
   35: import dmsl.crowdspaceSidemenu.R;
   36:
   37: import android.app.AlertDialog;
   38: import android.app.ListActivity;
   39: import android.app.ProgressDialog;
   40: import android.content.Context;
   41: import android.content.DialogInterface;
   42: import android.content.Intent;
   43: import android.content.SharedPreferences;
   44: import android.net.Uri;
   45: import android.os.Bundle;
   46: import android.os.Environment;
   47: import android.view.LayoutInflater;
   48: import android.view.View;
   49: import android.view.ViewGroup;
   50: import android.view.View.OnClickListener;
   51: import android.widget.ArrayAdapter;
   52: import android.widget.Button;
   53: import android.widget.ImageView;
   54: import android.widget.ListView;
   55: import android.widget.TextView;
   56:
   57: public class AndroidFileBrowserLogger extends ListActivity implements
   58:                 OnClickListener {
   59:         // Enum For The Display Mode You Want
   60:         private enum DISPLAYMODE {
   61:                 ABSOLUTE, RELATIVE;
   62:         }
   63:
   64:         private final DISPLAYMODE displayMode = DISPLAYMODE.ABSOLUTE;
   65:
   66:         private List<String> directoryEntries = new ArrayList<String>();
   67:         private File currentDirectory;
   68:         private File homeDirectory;
   69:         private File file;
   70:         private TextView pwd;
   71:         private Button select_file_folder;
   72:         private boolean selectFolder = true;
   73:         private String IP;
   74:         private String PORT;
   75:         private String folder_path;
   76:         private String file_path;
   77:         private SharedPreferences sharedPreferences;
   78:
   79:         @Override
   80:         public void onCreate(Bundle icicle) {
   81:
   82:                 super.onCreate(icicle);
   83:                 setContentView(R.layout.main_choose_file_or_directory);
   84:
   85:                 pwd = (TextView) findViewById(R.id.pwd);
   86:                 select_file_folder = (Button) findViewById(R.id.select_file_fo
lder);
   87:                 select_file_folder.setOnClickListener(this);
   88:
   89:                 Bundle extras = getIntent().getExtras();
   90:                 if (extras != null) {
   91:                         selectFolder = extras.getBoolean("to_Browse");
   92:                 }
   93:
   94:                 sharedPreferences = getSharedPreferences("Indoor_Preferences",
   95:                                 MODE_PRIVATE);
   96:
   97:                 folder_path = sharedPreferences.getString("folder_browser", ""
);
   98:                 file_path = sharedPreferences.getString("upload_file", "");
   99:                 IP = sharedPreferences.getString("serverIP", "");
  100:                 PORT = sharedPreferences.getString("serverPORT", "");
  101:
  102:                 if (selectFolder)
  103:                         select_file_folder.setText("Save in this folder");
  104:                 else {
  105:                         select_file_folder.setText("Upload File");
```

```
106:
107:                         if (file_path != null && !file_path.trim().equals("")
108:                                         && new File(file_path).canRead())
109:                                 pwd.setText(file_path);
110:                 }
111:
112:                 if (folder_path == null || folder_path.equals("")) {
113:                         browseToHome();
114:                 } else {
115:                         currentDirectory = new File(makePath(folder_path));
116:                         homeDirectory = new File("/");
117:                         browseTo(currentDirectory);
118:                 }
119:         }
120:
121:         // This Function Browses To The Root-Directory Of The File-System.
122:         private void browseToHome() {
123:
124:                 currentDirectory = new File(makePath("/"));
125:
126:                 if (currentDirectory == null || isMount())
127:                         currentDirectory = new File("/");
128:
129:                 homeDirectory = currentDirectory.getAbsoluteFile();
130:                 browseTo(currentDirectory);
131:
132:         }
133:
134:         // check that the sdcard is mounted
135:         static public boolean isMount() {
136:                 String state = Environment.getExternalStorageState();
137:                 if (Environment.MEDIA_MOUNTED.equals(state)) {
138:                         return true;
139:                 }
140:                 return false;
141:         }
142:
143:         private String makePath(String path) {
144:                 if (path.length() == 0) {
145:                         return "/";
146:                 }
147:
148:                 File check = new File(path);
149:                 if (!check.exists() || !check.canRead()) {
150:                         return "/";
151:                 }
152:                 if (check.isDirectory())
153:                         return path;
154:                 else {
155:                         return check.getParent();
156:                 }
157:         }
158:
159:         private void browseTo(final File aDirectory) {
160:                 if (aDirectory.isDirectory()) {
161:                         this.currentDirectory = aDirectory;
162:                         fill(aDirectory.listFiles());
163:
164:                         if (selectFolder)
165:                                 pwd.setText(currentDirectory.getAbsolutePath()
);
166:                 }
167:         }
168:
169:         private void fill(File[] files) {
170:                 this.directoryEntries.clear();
171:                 // Add the "~" == "home directory"
172:                 // And the ".." == 'Up one level'
173:                 this.directoryEntries.add(getString(R.string.homeDir));
174:                 if (this.currentDirectory.getParent() != null)
175:                         this.directoryEntries.add(getString(R.string.parentDir
));
176:
177:                 switch (this.displayMode) {
178:
179:                 case ABSOLUTE:
180:                         for (File file : files) {
181:                                 this.directoryEntries.add(file.getAbsolutePath
());
182:                         }
183:                         break;
184:                 case RELATIVE: // On relative Mode, we have to add the current
-path to
185:                         // the beginning
186:                         int currentPathStringLenght = this.currentDirectory
187:                                         .getAbsolutePath().length();
188:                         for (File file : files) {
189:                                 this.directoryEntries.add(file.getAbsolutePath
().substring(
190:                                                 currentPathStringLenght));
191:                         }
192:                         break;
193:                 }
194:
195:                 MyCustomAdapter directoryList = new MyCustomAdapter(this,
196:                                 R.xml.file_row, this.directoryEntries);
197:
198:                 this.setListAdapter(directoryList);
199:         }
200:
201:         @Override
202:         protected void onListItemClick(ListView l, View v, int position, long
id) {
203:                 // TODO Auto-generated method stub
204:                 file = new File(directoryEntries.get(position));
205:
206:                 if (file.isDirectory()) {
207:                         if (file.canRead()) {
208:                                 if (file.getName().equals(getString(R.string.p
arentDir))) {
209:                                         browseTo(currentDirectory.getParentFil
e());
210:                                 } else if (file.getName().equals(getString(R.s
tring.homeDir))) {
211:                                         browseTo(homeDirectory);
212:                                 } else
213:                                         browseTo(file);
214:                         } else {
215:                                 showAlert("Read Permission Denied", "Warning",
 this);
216:                         }
217:                 } else if (!file.isDirectory() && selectFolder) {
```

```
  218:                         showAlert("Not a directory", "Warning", this);
  219:                 } else {
  220:                         pwd.setText(file.getAbsolutePath());
  221:                 }
  222:                 super.onListItemClick(l, v, position, id);
  223:
  224:         }
  225:
  226:         public static void showAlert(String message, String title, Context ctx
) {
  227:                 // Create a builder
  228:                 AlertDialog.Builder builder = new AlertDialog.Builder(ctx);
  229:                 builder.setTitle(title);
  230:                 // add buttons and listener
  231:
  232:                 builder.setMessage(message).setCancelable(false)
  233:                                 .setPositiveButton("OK", new DialogInterface.O
nClickListener() {
  234:                                         public void onClick(DialogInterface di
alog, int id) {
  235:                                                 dialog.cancel();
  236:                                         }
  237:                                 });
  238:
  239:                 // Create the dialog
  240:                 AlertDialog ad = builder.create();
  241:                 // show
  242:                 ad.show();
  243:         }
  244:
  245:         public void onClick(View v) {
  246:
  247:                 switch (v.getId()) {
  248:
  249:                 case R.id.select_file_folder:
  250:
  251:                         if (selectFolder) {
  252:
  253:                                 if (currentDirectory.canWrite()) {
  254:                                         Intent data = new Intent();
  255:                                         data.setData(Uri.parse(currentDirector
y.getAbsolutePath()));
  256:                                         setResult(RESULT_OK, data);
  257:                                         finish();
  258:                                 } else
  259:                                         showAlert("Write Permission Denied", "
Warning", this);
  260:
  261:                         } else {
  262:                                 file = new File(pwd.getText().toString());
  263:                                 if (file.exists()) {
  264:                                         if (file.canRead()) {
  265:
  266:                                                 Intent data = new Intent();
  267:                                                 data.setData(Uri.parse(pwd.get
Text().toString()));
  268:                                                 setResult(RESULT_OK, data);
  269:
  270:                                                 ProgressDialog dialog = Progre
ssDialog.show(
  271:                                                                 AndroidFileBro
wserLogger.this, "",
  272:                                                                 "Uploading. Pl
ease wait...", true, false);
  273:
  274:                                                 UploadingSettings us = new Upl
oadingSettings(IP, PORT,
  275:                                                                 file.getAbsolu
tePath(), dialog);
  276:
  277:                                                 if (us.connect()) {
  278:                                                         us.start();
  279:                                                 } else {
  280:                                                         showAlert(us.getErrMsg
(), "Error", this);
  281:                                                         dialog.dismiss();
  282:                                                 }
  283:                                         } else
  284:                                                 showAlert("Read Permission Den
ied", "Warning", this);
  285:                                 } else
  286:                                         showAlert("No file selected", "Warning
", this);
  287:                                 break;
  288:                         }
  289:                 }
  290:         }
  291:
  292:         /**
  293:          * Control when back button is pressed
  294:          * */
  295:         @Override
  296:         public void onBackPressed() {
  297:                 finish();
  298:         }
  299:
  300:         // Public Inner Class Which Help Me To Put Png Image For Each File Tha
t Is
  301:         // Different Type
  302:         public class MyCustomAdapter extends ArrayAdapter<String> {
  303:                 List<String> myList;
  304:
  305:                 public MyCustomAdapter(Context context, int textViewResourceId
,
  306:                                 List<String> objects) {
  307:
  308:                         super(context, textViewResourceId, objects);
  309:                         myList = objects;
  310:                         // TODO Auto-generated constructor stub
  311:                 }
  312:
  313:                 @Override
  314:                 public View getView(int position, View convertView, ViewGroup
parent) {
  315:                         // TODO Auto-generated method stub
  316:                         // super.getView(position, convertView, parent);
  317:                         View row = convertView;
  318:                         // Check If Row Is Null
  319:                         if (row == null) {
  320:                                 // Make New Layoutinflater
  321:                                 LayoutInflater vi = (LayoutInflater) getSystem
Service(Context.LAYOUT_INFLATER_SERVICE);
```

```
  322:                                row = vi.inflate(R.xml.file_row, parent, false
);
  323:                        }
  324:                        TextView label = (TextView) row.findViewById(R.id.file
);
  325:                        String stringFile = myList.get(position).toString();
  326:                        // Change The Symbol Of The Home Directory
  327:                        if (stringFile.equals(getString(R.string.homeDir)))
  328:                                label.setText("~");
  329:                        else
  330:                                label.setText(stringFile);
  331:
  332:                        File file = new File(stringFile);
  333:                        ImageView icon = (ImageView) row.findViewById(R.id.ico
n);
  334:
  335:                        if (file.isDirectory())
  336:                                icon.setImageResource(R.drawable.directory);
  337:                        else
  338:                                icon.setImageResource(FindDrawable(stringFile)
);
  339:                        return row;
  340:                }
  341:
  342:                // Find The Right Icon For Each File Type
  343:                private int FindDrawable(String file) {
  344:                        if (file.endsWith(".txt")) {
  345:                                return R.drawable.txt;
  346:                        } else if (file.endsWith(".pdf")) {
  347:                                return R.drawable.pdf;
  348:                        } else if (file.endsWith(".exe")) {
  349:                                return R.drawable.exe;
  350:                        } else if (file.endsWith(".apk")) {
  351:                                return R.drawable.apk;
  352:                        } else if (file.endsWith(".png")) {
  353:                                return R.drawable.png;
  354:                        } else if (file.endsWith(".gif")) {
  355:                                return R.drawable.gif;
  356:                        } else if (file.endsWith(".jpg")) {
  357:                                return R.drawable.jpg;
  358:                        } else if (file.endsWith(".rar")) {
  359:                                return R.drawable.rar;
  360:                        } else if (file.endsWith(".zip")) {
  361:                                return R.drawable.zip;
  362:                        } else if (file.endsWith(".gz")) {
  363:                                return R.drawable.gz;
  364:                        } else if (file.endsWith(".mp3") || file.endsWith(".wa
v")
  365:                                        || file.endsWith(".amp")) {
  366:                                return R.drawable.sound;
  367:                        } else if (file.endsWith(".mp4") || file.endsWith(".av
i")
  368:                                        || file.endsWith(".flv")) {
  369:                                return R.drawable.video;
  370:                        } else
  371:                                return R.drawable.txt;
  372:                }
  373:        }
  374: }
```

```
/*
 ** CaptureActivity.java
 **
 ** Copyright (C) 2012 Julia Metochi
 **
 ** This program is free software: you can redistribute it and/or modify
 ** it under the terms of the GNU General Public License as published by
 ** the Free Software Foundation, either version 3 of the License, or
 ** at your option) any later version.
 **
 ** This program is distributed in the hope that it will be useful,
 ** but WITHOUT ANY WARRANTY; without even the implied warranty of
 ** MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 ** GNU General Public License for more details.
 **
 ** You should have received a copy of the GNU General Public License
 ** along with this program.  If not, see <http://www.gnu.org/licenses/>.
 */

package com.google.zxing.client.android;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.lang.reflect.Type;
import java.text.NumberFormat;
import java.util.ArrayList;
import java.util.Collection;
import java.util.List;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.apache.http.protocol.HTTP;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import android.app.Activity;
import android.app.AlertDialog;
import android.app.Dialog;
import android.content.ComponentName;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.IntentSender.SendIntentException;
import android.database.SQLException;
import android.database.sqlite.SQLiteDatabase;
import android.graphics.Bitmap;
import android.location.Criteria;
import android.location.Location;
import android.location.LocationManager;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.provider.Settings;
import android.util.Log;
import android.view.KeyEvent;
import android.view.Menu;
import android.view.MenuItem;
import android.view.SurfaceHolder;
import android.view.SurfaceView;
import android.view.View;
import android.view.Window;
import android.view.WindowManager;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

import com.google.gson.Gson;
import com.google.gson.reflect.TypeToken;

import com.google.zxing.BarcodeFormat;
import com.google.zxing.Result;
import com.google.zxing.client.android.camera.CameraManager;

import dmsl.ItemScan.GoogleShoppingAPIHelperClass;
import dmsl.ItemScan.ObjectToSend;
import dmsl.ItemScan.ProductObject;
import dmsl.crowdspaceSidemenu.MainActivity;
import dmsl.crowdspaceSidemenu.R;

/**
 * This activity opens the camera and does the actual scanning on a background
 * thread. It draws a viewfinder to help the user place the barcode correctly
 * and shows feedback as the image processing is happening.
 *
 * When a product is found, its price and title are shown on the screen. Also a
 * google map is shown with the user's current location and it can be expanded
 * to full screen in order to show points on the map where the specific produc
t
 * was scanned before.
 *
 * (ZXing Authors)
 *
 * @author dswitkin@google.com (Daniel Switkin)
 * @author Sean Owen
 *
 *
 */
public final class CaptureActivity extends Activity implements
                SurfaceHolder.Callback {

        private ArrayList<String> itemInfo = new ArrayList<String>();
        private ArrayList<ProductObject> listProdObj = new ArrayList<ProductOb
ject>();
        private static final String TAG = CaptureActivity.class.getSimpleName(
```

```
);
	public static ProductObject editItem;

	// Menu components
	private static final int MANUALENTRY_ID = Menu.FIRST;
	private static final int HELP_ID = Menu.FIRST + 1;
	private static final int ABOUT_ID = Menu.FIRST + 2;
	private static final int EXIT_ID = Menu.FIRST + 3;

	public static boolean DELETE_ITEM = false;
	public static boolean MOVE_ITEM = false;
	public static boolean ADD_ITEM = false;

	private CameraManager cameraManager;
	private CaptureActivityHandler handler;
	private Result savedResultToShow;
	private ViewfinderView viewfinderView;
	private boolean hasSurface;
	private Collection<BarcodeFormat> decodeFormats;
	private String characterSet;
	private InactivityTimer inactivityTimer;
	private BeepManager beepManager;

	// onActivityResult codes
	private static final int USER_INPUT_FOR_RESULT = 0;
	private static final int USER_EDIT = 2;
	private static final int USER_MANUAL_ENTRY = 3;

	// GeoLocation components
	private double geoLongitude;
	private double geoLatitude;
	private String provider;
	private LocationManager locationManager;

	public static JSONArray arrayObj, arrayObjIdRev;
	static int w = 0;

	public static int itemIsScanned = 0;
	public int BACK = 0;
	public static List<String> listItems = new ArrayList<String>();

	public static final String defaultImg = "http://www.cybuzz.in/wp-content/uploads/2012/05/product-icon.png";

	/**
	 * This method receives a number and formats it in order to have "n" digits
	 * after the decimal point notation.
	 *
	 * @param number
	 *            The real number to be formatted.
	 * @param n
	 *            The number of decimal points the new real number will have.
	 * @return The new formatted number in a string form.
	 */
	public String formatToNDigits(double number, int n) {
		NumberFormat form = NumberFormat.getInstance();
		form.setMinimumFractionDigits(n);
		form.setMaximumFractionDigits(n);
		return form.format(number);
	}

	/**
	 * Checks if the Wi-Fi is on and if an internet connection is available.
	 *
	 * @return True if a connection is available, False otherwise.
	 */
	private boolean isNetworkAvailable() {
		ConnectivityManager connectivityManager = (ConnectivityManager) getSystemService(Context.CONNECTIVITY_SERVICE);
		NetworkInfo activeNetworkInfo = connectivityManager
				.getActiveNetworkInfo();
		return activeNetworkInfo != null
				&& activeNetworkInfo.isConnectedOrConnecting();
	}

	ViewfinderView getViewfinderView() {
		return viewfinderView;
	}

	public Handler getHandler() {
		return handler;
	}

	CameraManager getCameraManager() {
		return cameraManager;
	}

	// ********************************************************************************************
	@Override
	public void onCreate(Bundle icicle) {
		super.onCreate(icicle);
		Window window = getWindow();
		window.addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
		setContentView(R.layout.capture);
		hasSurface = false;
		inactivityTimer = new InactivityTimer(this);
		beepManager = new BeepManager(this);

		if (!isNetworkAvailable()) {
			AlertDialog.Builder dialog = new AlertDialog.Builder(this);
			dialog.setTitle("Wi-Fi is turned off/No internet connection available");
			dialog.setMessage("Scanner needs an internet connection for all its functions to work properly.");
			dialog.setPositiveButton("Ok",
					new DialogInterface.OnClickListener() {

						@Override
						public void onClick(DialogInterface dialog, int which) {
							// TODO Auto-generated method stub
							Intent intent = new In
```

```
tent();
  227:                                                    intent.setComponent(ne
w ComponentName(
  228:                                                                        "com.a
ndroid.settings",
  229:                                                                        "com.a
ndroid.settings.WirelessSettings"));
  230:                                                    startActivity(intent);
  231:                                            }
  232:                                    });
  233:                    dialog.show();
  234:            }
  235:
  236:            initializeViews();
  237:    }
  238:
  239:    // ************************************************************************
*****************************************************
  240:    @Override
  241:    protected void onResume() {
  242:            super.onResume();
  243:
  244:            cameraManager = new CameraManager(getApplication());
  245:
  246:            viewfinderView = (ViewfinderView) findViewById(R.id.viewfinder
_view);
  247:            viewfinderView.setCameraManager(cameraManager);
  248:
  249:            handler = null;
  250:
  251:            resetStatusView();
  252:
  253:            SurfaceView surfaceView = (SurfaceView) findViewById(R.id.prev
iew_view);
  254:            SurfaceHolder surfaceHolder = surfaceView.getHolder();
  255:            if (hasSurface) {
  256:                    // The activity was paused but not stopped, so the sur
face still
  257:                    // exists. Therefore
  258:                    // surfaceCreated() won't be called, so init the camer
a here.
  259:                    initCamera(surfaceHolder);
  260:            } else {
  261:                    // Install the callback and wait for surfaceCreated()
to init the
  262:                    // camera.
  263:                    surfaceHolder.addCallback(this);
  264:                    surfaceHolder.setType(SurfaceHolder.SURFACE_TYPE_PUSH_
BUFFERS);
  265:            }
  266:
  267:            beepManager.updatePrefs();
  268:
  269:            inactivityTimer.onResume();
  270:            if (!isNetworkAvailable()) {
  271:                    AlertDialog.Builder dialog = new AlertDialog.Builder(t
his);
  272:                    dialog.setTitle("Wi-Fi is turned off/No internet conne
ction available");
  273:                    dialog.setMessage("Scanner needs an internet connectio
n for all its functions to work properly.");
  274:                    dialog.setPositiveButton("Ok",
  275:                                    new DialogInterface.OnClickListener()
{
  276:
  277:                                            @Override
  278:                                            public void onClick(DialogInte
rface dialog, int which) {
  279:                                                    // TODO Auto-generated
 method stub
  280:                                                    Intent intent = new In
tent();
  281:                                                    intent.setComponent(ne
w ComponentName(
  282:                                                                        "com.a
ndroid.settings",
  283:                                                                        "com.a
ndroid.settings.WirelessSettings"));
  284:                                                    startActivity(intent);
  285:                                            }
  286:                                    });
  287:                    dialog.show();
  288:            }
  289:
  290:    }
  291:
  292:    // ************************************************************************
*****************************************************
  293:
  294:    @Override
  295:    protected void onPause() {
  296:            if (handler != null) {
  297:                    handler.quitSynchronously();
  298:                    handler = null;
  299:            }
  300:            inactivityTimer.onPause();
  301:            cameraManager.closeDriver();
  302:            if (!hasSurface) {
  303:                    SurfaceView surfaceView = (SurfaceView) findViewById(R
.id.preview_view);
  304:                    SurfaceHolder surfaceHolder = surfaceView.getHolder();
  305:                    surfaceHolder.removeCallback(this);
  306:            }
  307:            super.onPause();
  308:
  309:    }
  310:
  311:    // ************************************************************************
*********************************************
  312:    @Override
  313:    protected void onDestroy() {
  314:            inactivityTimer.shutdown();
  315:            super.onDestroy();
  316:    }
  317:
  318:    // ************************************************************************
*********************************************
  319:    @Override
  320:    public boolean onCreateOptionsMenu(Menu menu) {
  321:            super.onCreateOptionsMenu(menu);
  322:            menu.add(Menu.NONE, MANUALENTRY_ID, Menu.NONE, "Manual Entry")
.setIcon(
```

```
                                android.R.drawable.ic_input_add);

                return true;
        }

        // ********************************************************************
*************************************************
        @Override
        public boolean onOptionsItemSelected(MenuItem item) {
                Intent intent = new Intent(Intent.ACTION_VIEW);
                intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_WHEN_TASK_RESET);
                switch (item.getItemId()) {

                case MANUALENTRY_ID:
                        Intent i = new Intent("dmsl.ItemScan.ASKFORMANUALENTRY
ACTIVITY");
                        startActivityForResult(i, USER_MANUAL_ENTRY);
                        break;

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

        // ********************************************************************
*************************************************
        @Override
        public boolean onKeyDown(int keyCode, KeyEvent event) {
                switch (keyCode) {
                case KeyEvent.KEYCODE_BACK:
                        BACK = 1;
                        finish();
                        return true;
                }
                return super.onKeyDown(keyCode, event);
        }

        // ********************************************************************
*************************************************
        private void decodeOrStoreSavedBitmap(Bitmap bitmap, Result result) {
                // Bitmap isn't used yet -- will be used soon
                if (handler == null) {
                        savedResultToShow = result;
                } else {
                        if (result != null) {
                                savedResultToShow = result;
                        }
                        if (savedResultToShow != null) {
                                Message message = Message.obtain(handler,
                                                R.id.decode_succeeded, savedRe
sultToShow);

                                handler.sendMessage(message);
                        }
                        savedResultToShow = null;
                }
        }

        // ********************************************************************
*************************************************
        @Override
        public void surfaceCreated(SurfaceHolder holder) {
                if (holder == null) {
                        Log.e(TAG,
                                        "*** WARNING *** surfaceCreated() gave
 us a null surface!");
                }
                if (!hasSurface) {
                        hasSurface = true;
                        initCamera(holder);
                }
        }

        // ********************************************************************
*************************************************
        @Override
        public void surfaceDestroyed(SurfaceHolder holder) {
                hasSurface = false;
        }

        // ********************************************************************
*************************************************
        @Override
        public void surfaceChanged(SurfaceHolder holder, int format, int width
,
                        int height) {

        }

        // ********************************************************************
*************************************************
        /**
         * A valid barcode has been found, so the data is displayed on the scr
een
         * and the preview is restarted so that the user can scan a new barcod
e.
         *
         * @param rawResult
         *            The contents of the barcode.
         * @param barcode
         *            A greyscale bitmap of the camera data which was decoded.
         *            (Currently not used, might be used in future extensions.
)
         */
        public void handleDecode(Result rawResult, Bitmap barcode) {

                final String[] itemOptions = { "Move item to selected location
",
                                "Add item to selected location", "Delete item"
 };

                new AlertDialog.Builder(this);

                final String barcodeText = rawResult.getText();

                String prodTitle = null;
                ProductObject testProduct = null;

                inactivityTimer.onActivity();
                beepManager.playBeepSoundAndVibrate();

                restartPreviewAfterDelay(2500L);
```

```
  429:
  430:                String exists = null;
  431:                exists = searchCouchDb(barcodeText);
  432:                if (!(exists.equalsIgnoreCase("fail"))) {
  433:                        GoogleShoppingAPIHelperClass pr = new GoogleShoppingAP
IHelperClass();
  434:                        editItem = pr.productFoundInCouch(exists);
  435:                }
  436:
  437:                if (exists.equalsIgnoreCase("fail") && isNetworkAvailable()) {
  438:                        // If the barcode was not matched in our database we s
earch for
  439:                        // a title in Google for Shopping API.
  440:                        GoogleShoppingAPIHelperClass googleShopHelper = new Go
ogleShoppingAPIHelperClass();
  441:                        itemInfo = googleShopHelper.getGoogleShoppingInfo(barc
odeText);
  442:
  443:                        googleShopHelper = null; // Garbage Collector will fre
e this memory
  444:
// on its next run.
  445:                }
  446:                // if product not found in database
  447:                if (exists.equalsIgnoreCase("fail")) {
  448:                        // if product exists in google shhopping API
  449:                        if (itemInfo != null) {
  450:                                prodTitle = itemInfo.get(0).toString();
  451:                                if (prodTitle != null) {
  452:
  453:                                        Intent i = new Intent("dmsl.ItemScan.A
SKFORUSERINPUT");
  454:                                        Bundle b = new Bundle();
  455:                                        b.putString("barcode", barcodeText);
  456:                                        b.putString("title", prodTitle);
  457:                                        b.putString("brand", itemInfo.get(1));
  458:                                        b.putString("price", itemInfo.get(2));
  459:                                        // b.putString("image", itemInfo.get(3
));
  460:                                        i.putExtras(b);
  461:                                        startActivityForResult(i, USER_INPUT_F
OR_RESULT);
  462:
  463:                                }
  464:                        }
  465:
  466:                        // Product was not found,not even the title.User must
input all info
  467:                        else {
  468:                                Intent i = new Intent("dmsl.ItemScan.ASKFORINP
UTACTIVITY");
  469:                                Bundle b = new Bundle();
  470:                                b.putString("barcode", barcodeText);
  471:                                i.putExtras(b);
  472:                                startActivityForResult(i, USER_INPUT_FOR_RESUL
T);
  473:
  474:                        }
  475:
  476:                } else {
  477:                        Toast.makeText(getApplicationContext(), "Item exists i
n database",
  478:                                        Toast.LENGTH_SHORT).show();
  479:                        BACK = 1;
  480:
  481:                        AlertDialog.Builder aboutBuilder = new AlertDialog.Bui
lder(this);
  482:                        aboutBuilder.setTitle("ITEM ALREADY EXISTS");
  483:                        aboutBuilder
  484:                                        .setMessage("Do you want to add it aga
in to the selected position?");
  485:
  486:                        aboutBuilder.setPositiveButton("Yes",
  487:                                        new DialogInterface.OnClickListener()
{
  488:
  489:                                                @Override
  490:                                                public void onClick(DialogInte
rface dialog, int which) {
  491:                                                        ADD_ITEM = true;
  492:                                                        finish();
  493:                                                }
  494:
  495:                                        });
  496:
  497:                        aboutBuilder.setNegativeButton("No", null);
  498:
  499:                        aboutBuilder.show();
  500:                }
  501:
  502:        }
  503:
  504:        // ********************************************************************
*************************************************************
  505:        @Override
  506:        protected void onActivityResult(int requestCode, int resultCode, Inten
t data) {
  507:                // TODO Auto-generated method stub
  508:                super.onActivityResult(requestCode, resultCode, data);
  509:                JSONObject object = null;
  510:                if (resultCode == RESULT_OK) {
  511:                        if (requestCode == USER_INPUT_FOR_RESULT) {
  512:                                Bundle extras = data.getExtras();
  513:                                String title = extras.getString("title");
  514:                                String brand = extras.getString("brand");
  515:                                float price = extras.getFloat("price");
  516:                                String barcode = extras.getString("barcode");
  517:                                String img = null;
  518:                                String docId = null;
  519:                                String docRev = null;
  520:                                if (itemInfo != null) {
  521:                                        if (itemInfo.get(3) != null) {
  522:                                                img = itemInfo.get(3);
  523:                                        } else {
  524:                                                img = defaultImg;
  525:                                        }
  526:                                } else
  527:                                        img = defaultImg;
  528:                                ProductObject product = new ProductObject(titl
e, brand, price,
  529:                                                barcode, MainActivity.itemLon,
 MainActivity.itemLat,
```

```
                                                MainActivity.currentFloor, img
, docId, docRev);

                                sendProductToCouch(product);
                                listProdObj.add(product);

                        }

                        else if (requestCode == USER_MANUAL_ENTRY) {
                                Bundle extras = data.getExtras();
                                String manualBarcode = extras.getString("strBa
rcode");
                                handleDecode(new Result(manualBarcode, null, n
ull, null), null);
                        }

                } else if (resultCode == RESULT_CANCELED
                                && !(requestCode == USER_EDIT || requestCode =
= USER_MANUAL_ENTRY)) {

                }
        }

        // ************************************************************************
*************************************************
        // Search couchDB to check if item already exists
        public String searchCouchDb(String barcode) {

                String result = null;
                InputStream inputStream = null;
                HttpClient httpclient = new DefaultHttpClient();
                HttpPost httppost = new HttpPost(
                                "http://juliavm.in.cs.ucy.ac.cy/search.php");
                List<NameValuePair> nvps = new ArrayList<NameValuePair>();
                nvps.add(new BasicNameValuePair("search", barcode));
                try {
                        httppost.setEntity(new UrlEncodedFormEntity(nvps, HTTP
.UTF_8));
                } catch (UnsupportedEncodingException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }

                String ifFound = null;
                try {
                        HttpResponse response = httpclient.execute(httppost);
                        // Save response in InputStream
                        HttpEntity entity = response.getEntity();
                        inputStream = entity.getContent();

                        // Convert response
                        BufferedReader reader = new BufferedReader(new InputSt
reamReader(
                                        inputStream, "iso-8859-1"), 8);

                        StringBuilder stringBuilder = new StringBuilder();
                        String line = null;

                        while ((line = reader.readLine()) != null) {
                                stringBuilder.append(line + "\n");
                        }

                        // Close Stream
                        inputStream.close();

                        result = stringBuilder.toString();
                        result = result.replaceAll("(\\r|\\n)", "");
                        GoogleShoppingAPIHelperClass h = new GoogleShoppingAPI
HelperClass();
                        ifFound = h.checkIfExistsInCouch(result);

                        System.out.println();

                } catch (ClientProtocolException e) {
                        e.printStackTrace();
                } catch (IOException e) {
                        e.printStackTrace();
                        Toast.makeText(this, "No internet access", Toast.LENGT
H_LONG)
                                        .show();
                        AlertDialog.Builder aboutBuilder = new AlertDialog.Bui
lder(this);
                        aboutBuilder.setTitle("Closing Application");
                        aboutBuilder
                                        .setMessage("No internet access...CLOD
A will be closed!");

                        aboutBuilder.setPositiveButton("Close",
                                        new DialogInterface.OnClickListener()
{

                                                public void onClick(DialogInte
rface dialog, int which) {

                                                        finish();
                                                }
                                        });

                        aboutBuilder.show();

                }

                if (ifFound != null)
                        return ifFound;
                else
                        return result;
        }

        // ////////////////////////////////////////////////////////////////////
////////////////////////////////////////
        public void deleteItemFromCouch(String doc_id, String doc_rev) {
                HttpClient httpclient = new DefaultHttpClient();
                HttpPost httpDelete = new HttpPost(
                                "http://juliavm.in.cs.ucy.ac.cy/delete.php");
                List<NameValuePair> nvps = new ArrayList<NameValuePair>();
                nvps.add(new BasicNameValuePair("docId", doc_id));
                nvps.add(new BasicNameValuePair("docRev", doc_rev));

                try {
                        httpDelete.setEntity(new UrlEncodedFormEntity(nvps, HT
TP.UTF_8));
                } catch (UnsupportedEncodingException e) {
                        e.printStackTrace();
```

```
  637:                }
  638:                try {
  639:                        HttpResponse response = httpclient.execute(httpDelete)
;
  640:                        System.out.println(response);
  641:                } catch (ClientProtocolException e) {
  642:                        e.printStackTrace();
  643:                } catch (IOException e) {
  644:                        e.printStackTrace();
  645:                        Toast.makeText(this, "No internet access", Toast.LENGT
H_LONG)
  646:                                        .show();
  647:                        AlertDialog.Builder aboutBuilder = new AlertDialog.Bui
lder(this);
  648:                        aboutBuilder.setTitle("Closing Application");
  649:                        aboutBuilder
  650:                                        .setMessage("No internet access...CLOD
A will be closed!");
  651:
  652:                        aboutBuilder.setPositiveButton("Close",
  653:                                        new DialogInterface.OnClickListener()
{
  654:
  655:                                                public void onClick(DialogInte
rface dialog, int which) {
  656:
  657:                                                        finish();
  658:                                                }
  659:                                        });
  660:
  661:                        aboutBuilder.show();
  662:                        return;
  663:                }
  664:
  665:        }
  666:
  667:        // ************************************************************************
*****************************************************
  668:
  669:        // If item does not exist in couchDB store its info
  670:        public void sendProductToCouch(ProductObject product) {
  671:
  672:                Gson gson = new Gson();
  673:                ObjectToSend obj = new ObjectToSend(product);
  674:                String json = gson.toJson(obj);
  675:
  676:                arrayObj = new JSONArray();
  677:                arrayObj.put(json);
  678:
  679:                ObjectToSend objWithDocIdRev = new ObjectToSend(product, 0);
  680:                String jsonIdRev = gson.toJson(objWithDocIdRev);
  681:                arrayObjIdRev = new JSONArray();
  682:                arrayObjIdRev.put(jsonIdRev);
  683:
  684:                HttpClient httpclient = new DefaultHttpClient();
  685:                HttpPost httppost = new HttpPost(
  686:                                "http://juliavm.in.cs.ucy.ac.cy/post.php");
  687:                List<NameValuePair> nvps = new ArrayList<NameValuePair>();
  688:                nvps.add(new BasicNameValuePair("value", json));
  689:                try {
  690:                        httppost.setEntity(new UrlEncodedFormEntity(nvps, HTTP
.UTF_8));
  691:                } catch (UnsupportedEncodingException e) {
  692:                        e.printStackTrace();
  693:                }
  694:
  695:                try {
  696:                        HttpResponse response = httpclient.execute(httppost);
  697:                        System.out.println(response);
  698:                } catch (ClientProtocolException e) {
  699:                        e.printStackTrace();
  700:                } catch (IOException e) {
  701:                        e.printStackTrace();
  702:                        Toast.makeText(this, "No internet access", Toast.LENGT
H_LONG)
  703:                                        .show();
  704:                        AlertDialog.Builder aboutBuilder = new AlertDialog.Bui
lder(this);
  705:                        aboutBuilder.setTitle("Closing Application");
  706:                        aboutBuilder
  707:                                        .setMessage("No internet access...CLOD
A will be closed!");
  708:
  709:                        aboutBuilder.setPositiveButton("Close",
  710:                                        new DialogInterface.OnClickListener()
{
  711:
  712:                                                public void onClick(DialogInte
rface dialog, int which) {
  713:
  714:                                                        finish();
  715:                                                }
  716:                                        });
  717:
  718:                        aboutBuilder.show();
  719:                        return;
  720:                }
  721:                finish();
  722:        }
  723:
  724:        // ************************************************************************
*****************************************************
  725:
  726:        /**
  727:         * Initialization of the Views in capture.xml.
  728:         */
  729:        private void initializeViews() {
  730:
  731:                // tvScannedItems = (TextView) findViewById(R.id.tvItemsScanne
d);
  732:                // tvItemsScanned = (TextView) findViewById(R.id.tvItemsScanne
d);
  733:        }
  734:
  735:        // ************************************************************************
*****************************************************
  736:
  737:        private void initCamera(SurfaceHolder surfaceHolder) {
  738:                if (surfaceHolder == null) {
  739:                        throw new IllegalStateException("No SurfaceHolder prov
ided");
  740:                }
```

```
741:                 if (cameraManager.isOpen()) {
742:                         Log.w(TAG,
743:                                         "initCamera() while already open -- la
te SurfaceView callback?");
744:                         return;
745:                 }
746:                 try {
747:                         cameraManager.openDriver(surfaceHolder);
748:                         // Creating the handler starts the preview, which can
also throw a
749:                         // RuntimeException.
750:                         if (handler == null) {
751:                                 handler = new CaptureActivityHandler(this, dec
odeFormats,
752:                                                 characterSet, cameraManager);
753:                         }
754:                         decodeOrStoreSavedBitmap(null, null);
755:                 } catch (IOException ioe) {
756:                         Log.w(TAG, ioe);
757:                         displayFrameworkBugMessageAndExit();
758:                 } catch (RuntimeException e) {
759:                         // Barcode Scanner has seen crashes in the wild of thi
s variety:
760:                         // java.?lang.?RuntimeException: Fail to connect to ca
mera service
761:                         Log.w(TAG, "Unexpected error initializing camera", e);
762:                         displayFrameworkBugMessageAndExit();
763:                 }
764:         }
765:
766:         // ************************************************************************
*************************************************
767:
768:         private void displayFrameworkBugMessageAndExit() {
769:                 AlertDialog.Builder builder = new AlertDialog.Builder(this);
770:                 builder.setTitle(getString(R.string.app_name));
771:                 builder.setMessage(getString(R.string.msg_camera_framework_bug
));
772:                 builder.setPositiveButton("OK", new FinishListener(this));
773:                 builder.setOnCancelListener(new FinishListener(this));
774:                 builder.show();
775:
776:         }
777:
778:         /**
779:          * Restarts the scanner
780:          *
781:          * @param delayMS
782:          *            The time in which it will restart (in milliseconds).
783:          */
784:         public void restartPreviewAfterDelay(long delayMS) {
785:                 if (handler != null) {
786:                         handler.sendEmptyMessageDelayed(R.id.restart_preview,
delayMS);
787:                 }
788:                 resetStatusView();
789:         }
790:
791:         // ************************************************************************
*************************************************
792:         private void resetStatusView() {
793:                 viewfinderView.setVisibility(View.VISIBLE);
794:         }
795:
796:         // ************************************************************************
*************************************************
797:         public void drawViewfinder() {
798:                 viewfinderView.drawViewfinder();
799:         }
800:
801:         // ************************************************************************
*************************************************
802:         @Override
803:         public void finish() {
804:                 // TODO Auto-generated method stub
805:
806:                 super.finish();
807:                 if (BACK == 1) {
808:                         itemIsScanned = 0;
809:                 } else {
810:                         itemIsScanned = 1;
811:                 }
812:
813:         }
814:
815: }
```

```java
package dmsl.crowdspaceSidemenu;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.util.ArrayList;
import java.util.List;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.apache.http.protocol.HTTP;
import org.json.JSONArray;

import com.actionbarsherlock.app.SherlockActivity;
import com.actionbarsherlock.view.Menu;
import com.actionbarsherlock.view.MenuItem;
import com.actionbarsherlock.view.SubMenu;
import com.google.gson.Gson;
import com.google.zxing.client.android.CaptureActivity;

import dmsl.ItemScan.GoogleShoppingAPIHelperClass;
import dmsl.ItemScan.ObjectToSend;
import dmsl.ItemScan.ProductObject;
import dmsl.crowdspace.airplace.Prefs_logger;

import android.app.Activity;
import android.app.AlertDialog;
import android.app.Dialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Color;
import android.os.Bundle;
import android.text.InputType;
import android.view.Gravity;
import android.view.KeyEvent;
import android.view.View;
import android.view.ViewGroup;
import android.view.Window;
import android.view.View.OnClickListener;
import android.view.WindowManager;
import android.view.inputmethod.InputMethodManager;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;

/**
 *
 * @author Julia Metochi
 *
 */
public class ItemMenu extends SherlockActivity {
        ImageButton delete, edit, move;
        public AlertDialog.Builder aboutBuilder;
        public static int DELETE_ITEM = 0;
        public static int EDIT_ITEM = 0;
        public static int MOVE_ITEM = 0;
        private Dialog dialog = null;
        static EditText barcode, location, price, brand, title, floor;
        static ProductObject prObj;
        public static JSONArray arrayItem;

        @Override
        protected void onCreate(Bundle savedInstanceState) {

                super.onCreate(savedInstanceState);
                setContentView(R.layout.item_menu);
                barcode = (EditText) findViewById(R.id.eBarcode);
                location = (EditText) findViewById(R.id.eLocation);
                price = (EditText) findViewById(R.id.ePrice);
                brand = (EditText) findViewById(R.id.eBrand);
                title = (EditText) findViewById(R.id.eTitle);
                floor = (EditText) findViewById(R.id.eFloor);

                // Find document in couch
                String itemInfo = searchByDocId(MyJavaScriptInterface.doc[0]);
                GoogleShoppingAPIHelperClass gHelp = new GoogleShoppingAPIHelperClass();

                // Get product info and display them to user
                prObj = gHelp.productFoundInCouch(itemInfo);
                if (prObj != null) {
                        barcode.setText(prObj.getBarcode());
                        location.setText("x: " + String.valueOf(prObj.getPointX())
                                        + "\ny: " + String.valueOf(prObj.getPointY()));
                        price.setText(String.valueOf(prObj.getPrice()));
                        brand.setText(prObj.getBrand());
                        title.setText(prObj.getTitle());
                        floor.setText(String.valueOf(prObj.getFloor()));

                        // Make keyboard disappear
                        this.getWindow().setSoftInputMode(
                                        WindowManager.LayoutParams.SOFT_INPUT_STATE_ALWAYS_HIDDEN);

                        setFocusFalse(barcode);
                        setFocusFalse(location);
                        setFocusFalse(price);
                        setFocusFalse(brand);
                        setFocusFalse(title);
                        setFocusFalse(floor);

                }
        }

        /*************************************************************************************************/

        /**
```

```
  118:          * Delete Item From CouchDB
  119:          *
  120:          * @param doc_id
  121:          * @param doc_rev
  122:          */
  123:         public void deleteItemFromCouch(String doc_id, String doc_rev) {
  124:                 HttpClient httpclient = new DefaultHttpClient();
  125:                 HttpPost httpDelete = new HttpPost(
  126:                                 "http://juliavm.in.cs.ucy.ac.cy/delete.php");
  127:                 List<NameValuePair> nvps = new ArrayList<NameValuePair>();
  128:                 nvps.add(new BasicNameValuePair("docId", doc_id));
  129:                 nvps.add(new BasicNameValuePair("docRev", doc_rev));
  130:
  131:                 try {
  132:                         httpDelete.setEntity(new UrlEncodedFormEntity(nvps, HT
TP.UTF_8));
  133:                 } catch (UnsupportedEncodingException e) {
  134:                         e.printStackTrace();
  135:                 }
  136:                 try {
  137:                         HttpResponse response = httpclient.execute(httpDelete)
;
  138:                         System.out.println(response);
  139:                 } catch (ClientProtocolException e) {
  140:                         e.printStackTrace();
  141:                 } catch (IOException e) {
  142:                         e.printStackTrace();
  143:                         // NoInternet = 1;
  144:                         Toast.makeText(this, "No internet access", Toast.LENGT
H_LONG)
  145:                                         .show();
  146:                         finish();
  147:                         System.exit(-1);
  148:                         // return;
  149:                 }
  150:
  151:         }
  152:
  153:         /*************************************************************************
*************************************************/
  154:
  155:         /**
  156:          * Update item info in couchDB
  157:          *
  158:          * @param docId
  159:          * @param item
  160:          */
  161:         public void updateDocument(String docId, ProductObject item) {
  162:
  163:                 String result = null;
  164:                 InputStream inputStream = null;
  165:                 Gson gson = new Gson();
  166:                 ObjectToSend obj = new ObjectToSend(item, 0);
  167:                 String json = gson.toJson(obj);
  168:
  169:                 HttpClient httpclient = new DefaultHttpClient();
  170:                 HttpPost httppost = new HttpPost(
  171:                                 "http://juliavm.in.cs.ucy.ac.cy/updateDocument
.php");
  172:                 List<NameValuePair> nvps = new ArrayList<NameValuePair>();
  173:                 nvps.add(new BasicNameValuePair("item", json));
  174:                 nvps.add(new BasicNameValuePair("docId", docId));
  175:                 try {
  176:                         httppost.setEntity(new UrlEncodedFormEntity(nvps, HTTP
.UTF_8));
  177:                 } catch (UnsupportedEncodingException e) {
  178:                         e.printStackTrace();
  179:                 }
  180:
  181:                 try {
  182:                         HttpResponse response = httpclient.execute(httppost);
  183:                         System.out.println(response);
  184:
  185:                         // Save response in InputStream
  186:                         HttpEntity entity = response.getEntity();
  187:                         inputStream = entity.getContent();
  188:
  189:                         // Convert response
  190:                         BufferedReader reader = new BufferedReader(new InputSt
reamReader(
  191:                                         inputStream, "iso-8859-1"), 8);
  192:
  193:                         StringBuilder stringBuilder = new StringBuilder();
  194:                         String line = null;
  195:
  196:                         while ((line = reader.readLine()) != null) {
  197:                                 stringBuilder.append(line + "\n");
  198:                         }
  199:                         // Close Stream
  200:                         inputStream.close();
  201:
  202:                         result = stringBuilder.toString();
  203:                         result = result.replaceAll("(\\r|\\n)", "");
  204:                 } catch (ClientProtocolException e) {
  205:                         e.printStackTrace();
  206:                 } catch (IOException e) {
  207:                         e.printStackTrace();
  208:                         Toast.makeText(this, "No internet access", Toast.LENGT
H_LONG)
  209:                                         .show();
  210:                         finish();
  211:                         System.exit(-1);
  212:                 }
  213:         }
  214:
  215:         /*************************************************************************
*************************************************/
  216:
  217:         /**
  218:          * Go Back
  219:          */
  220:         @Override
  221:         public boolean onKeyDown(int keyCode, KeyEvent event) {
  222:                 switch (keyCode) {
  223:                 case KeyEvent.KEYCODE_BACK:
  224:
  225:                         finish();
  226:                         return true;
  227:                 }
  228:                 return super.onKeyDown(keyCode, event);
  229:         }
  230:
```

```
  231:         /*****************************************************************
*************************************************/
  232:
  233:         /**
  234:          * Search CouchDB using the document id
  235:          *
  236:          * @param docId
  237:          * @return
  238:          */
  239:         public String searchByDocId(String docId) {
  240:
  241:                 String result = null;
  242:                 InputStream inputStream = null;
  243:                 HttpClient httpclient = new DefaultHttpClient();
  244:                 HttpPost httppost = new HttpPost(
  245:                                 "http://juliavm.in.cs.ucy.ac.cy/searchByDocId.
php");
  246:                 List<NameValuePair> nvps = new ArrayList<NameValuePair>();
  247:                 nvps.add(new BasicNameValuePair("searchByDocId", docId));
  248:                 try {
  249:                         httppost.setEntity(new UrlEncodedFormEntity(nvps, HTTP
.UTF_8));
  250:                 } catch (UnsupportedEncodingException e) {
  251:                         // TODO Auto-generated catch block
  252:                         e.printStackTrace();
  253:                 }
  254:
  255:                 String ifFound = null;
  256:                 try {
  257:                         HttpResponse response = httpclient.execute(httppost);
  258:                         // Save response in InputStream
  259:                         HttpEntity entity = response.getEntity();
  260:                         inputStream = entity.getContent();
  261:
  262:                         // Convert response
  263:                         BufferedReader reader = new BufferedReader(new InputSt
reamReader(
  264:                                         inputStream, "iso-8859-1"), 8);
  265:
  266:                         StringBuilder stringBuilder = new StringBuilder();
  267:                         String line = null;
  268:
  269:                         while ((line = reader.readLine()) != null) {
  270:                                 stringBuilder.append(line + "\n");
  271:                         }
  272:                         // Close Stream
  273:                         inputStream.close();
  274:
  275:                         result = stringBuilder.toString();
  276:                         result = result.replaceAll("(\\r|\\n)", "");
  277:                         GoogleShoppingAPIHelperClass h = new GoogleShoppingAPI
HelperClass();
  278:                         ifFound = h.checkIfExistsInCouch(result);
  279:
  280:                         System.out.println();
  281:
  282:                 } catch (ClientProtocolException e) {
  283:                         e.printStackTrace();
  284:                 } catch (IOException e) {
  285:                         e.printStackTrace();
  286:                         Toast.makeText(this, "No internet access", Toast.LENGT
H_LONG)
  287:                                                         .show();
  288:                         finish();
  289:                         System.exit(-1);
  290:                 }
  291:
  292:                 if (ifFound != null)
  293:                         return ifFound;
  294:                 else
  295:                         return result;
  296:         }
  297:
  298:         /*****************************************************************
*************************************************/
  299:
  300:         /**
  301:          * set focus of editText to true
  302:          *
  303:          * @param eText
  304:          */
  305:         public void setFocusTrue(EditText eText) {
  306:                 eText.setEnabled(true);
  307:                 eText.setTextColor(Color.BLACK);
  308:                 eText.setBackgroundColor(Color.WHITE);
  309:                 eText.setFocusable(true);
  310:                 eText.setFocusableInTouchMode(true);
  311:         }
  312:
  313:         /*****************************************************************
*************************************************/
  314:
  315:         /**
  316:          * Set focus of editText to false
  317:          *
  318:          * @param eText
  319:          */
  320:         public void setFocusFalse(EditText eText) {
  321:                 eText.setEnabled(false);
  322:                 eText.setTextColor(Color.parseColor("#ff33b5e5"));
  323:                 eText.setBackgroundColor(Color.BLACK);
  324:                 eText.setFocusable(false);
  325:                 eText.setFocusableInTouchMode(false);
  326:         }
  327:
  328:         /*****************************************************************
*************************************************/
  329:
  330:         /**
  331:          * Hide keyboard
  332:          *
  333:          * @param eText
  334:          */
  335:         public void hideKeyboard(EditText eText) {
  336:                 InputMethodManager mgr = (InputMethodManager) getSystemService
(Context.INPUT_METHOD_SERVICE);
  337:                 mgr.hideSoftInputFromWindow(eText.getWindowToken(), 0);
  338:         }
  339:
  340:         /*****************************************************************
*************************************************/
  341:
```

```
  342:         /**
  343:          * Create Menu
  344:          */
  345:         @Override
  346:         public boolean onCreateOptionsMenu(Menu menu) {
  347:                 getSupportActionBar().setHomeButtonEnabled(false);
  348:                 getSupportActionBar().setTitle("");
  349:
  350:                 /************************************** EDIT ITEM *********
**************************************************************/
  351:                 final SubMenu subMenu1 = menu.addSubMenu("Edit Item");
  352:                 MenuItem subMenu1Edit = subMenu1.getItem();
  353:                 subMenu1Edit.setIcon(android.R.drawable.ic_menu_edit);
  354:                 subMenu1Edit.setShowAsAction(MenuItem.SHOW_AS_ACTION_ALWAYS
  355:                                 | MenuItem.SHOW_AS_ACTION_WITH_TEXT);
  356:                 subMenu1Edit
  357:                                 .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  358:
  359:                                         @Override
  360:                                         public boolean onMenuItemClick(MenuIte
m item) {
  361:                                                 if (subMenu1.getItem().isCheck
ed()) {
  362:                                                         subMenu1.getItem().set
Icon(
  363:                                                                         androi
d.R.drawable.ic_menu_edit);
  364:                                                         String itemInfo = sear
chByDocId(MyJavaScriptInterface.doc[0]);
  365:                                                         GoogleShoppingAPIHelpe
rClass gHelp = new GoogleShoppingAPIHelperClass();
  366:                                                         prObj = gHelp.productF
oundInCouch(itemInfo);
  367:                                                         String docId = prObj.g
etDocId(), docRev = prObj
  368:                                                                         .getDo
cRev();
  369:                                                         double pX = prObj.getP
ointX(), pY = prObj
  370:                                                                         .getPo
intY();
  371:                                                         String img = prObj.get
Image();
  372:                                                         ProductObject editedIt
em = new ProductObject(String
  373:                                                                         .value
Of(title.getText()), String
  374:                                                                         .value
Of(brand.getText()), Float.valueOf(
  375:                                                                         Float.
parseFloat(String.valueOf(price
  376:
        .getText()))).floatValue(), String
  377:                                                                         .value
Of(barcode.getText()), pX, pY,
  378:                                                                         Intege
r.valueOf(
  379:
        String.valueOf(floor.getText()))
  380:
        .intValue(), img, docId, docRev);
  381:
  382:                                                         EDIT_ITEM = 1;
  383:                                                         setFocusFalse(barcode)
;
  384:                                                         hideKeyboard(barcode);
  385:                                                         setFocusFalse(location
);
  386:                                                         hideKeyboard(location)
;
  387:                                                         setFocusFalse(price);
  388:                                                         hideKeyboard(price);
  389:                                                         setFocusFalse(brand);
  390:                                                         hideKeyboard(brand);
  391:                                                         setFocusFalse(title);
  392:                                                         hideKeyboard(title);
  393:                                                         setFocusFalse(floor);
  394:                                                         hideKeyboard(floor);
  395:
  396:                                                         subMenu1.getItem().set
Checked(false);
  397:                                                         updateDocument(docId,
editedItem);
  398:
  399:                                                 } else {
  400:                                                         setFocusTrue(barcode);
  401:                                                         setFocusTrue(location)
;
  402:                                                         location.setKeyListene
r(null);
  403:                                                         setFocusTrue(price);
  404:                                                         setFocusTrue(brand);
  405:                                                         setFocusTrue(title);
  406:                                                         setFocusTrue(floor);
  407:
  408:                                                         subMenu1.getItem().set
Icon(
  409:                                                                         androi
d.R.drawable.ic_menu_save);
  410:                                                         EDIT_ITEM = 1;
  411:                                                         subMenu1.getItem().set
Checked(true);
  412:
  413:                                                 }
  414:
  415:                                                 return false;
  416:                                         }
  417:                                 });
  418:
  419:                 /************************************** END OF EDIT ITEM **
*************************************************************/
  420:
  421:                 /************************************** DELETE ITEM *******
**************************************************************/
  422:
  423:                 SubMenu subMenu2 = menu.addSubMenu("Delete Item");
  424:                 MenuItem subMenu2Delete = subMenu2.getItem();
  425:                 subMenu2Delete.setIcon(android.R.drawable.ic_menu_delete);
  426:                 subMenu2Delete.setShowAsAction(MenuItem.SHOW_AS_ACTION_ALWAYS
  427:                                 | MenuItem.SHOW_AS_ACTION_WITH_TEXT);
  428:                 subMenu2Delete
```

```
  429:                                .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  430:
  431:                                        @Override
  432:                                        public boolean onMenuItemClick(MenuIte
m item) {
  433:                                                Toast.makeText(ItemMenu.this,
"Delete",
  434:                                                               Toast.LENGTH_S
HORT).show();
  435:                                                DELETE_ITEM = 1;
  436:                                                deleteItemFromCouch(MyJavaScri
ptInterface.doc[0],
  437:                                                               MyJavaScriptIn
terface.doc[1]);
  438:
  439:                                                finish();
  440:
  441:                                                return false;
  442:                                        }
  443:
  444:                                });
  445:
  446:                    /************************************* END OF DELETE ITEM
*********************************************************************/
  447:
  448:                    /************************************* MOVE ITEM *********
*************************************************************/
  449:
  450:                    SubMenu subMenu3 = menu.addSubMenu("Move Item");
  451:                    MenuItem subMenu3Move = subMenu3.getItem();
  452:                    subMenu3Move.setIcon(R.drawable.move_item);
  453:                    subMenu3Move.setShowAsAction(MenuItem.SHOW_AS_ACTION_ALWAYS
  454:                                    | MenuItem.SHOW_AS_ACTION_WITH_TEXT);
  455:                    subMenu3Move
  456:                                    .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  457:
  458:                                            @Override
  459:                                            public boolean onMenuItemClick(MenuIte
m item) {
  460:                                                    Toast.makeText(ItemMenu.this,
"Move",
  461:                                                                   Toast.LENGTH_S
HORT).show();
  462:                                                    MOVE_ITEM = 1;
  463:                                                    Toast.makeText(ItemMenu.this,
"Select new location",
  464:                                                                   Toast.LENGTH_L
ONG).show();
  465:
  466:                                                    finish();
  467:
  468:                                                    return false;
  469:                                            }
  470:
  471:                                    });
  472:                    /************************************* END OF MOVE ITEM **
*************************************************************/
  473:
  474:                    SubMenu subMenu4 = menu.addSubMenu("Main Menu");
  475:                    subMenu4.add("Tracker");
  476:                    subMenu4.add("Help");
  477:
  478:                    MenuItem subMenu4Main = subMenu4.getItem();
  479:                    subMenu4Main
  480:                                    .setIcon(R.drawable.abs__ic_menu_moreoverflow_
normal_holo_dark);
  481:                    subMenu4Main.setShowAsAction(MenuItem.SHOW_AS_ACTION_ALWAYS
  482:                                    | MenuItem.SHOW_AS_ACTION_WITH_TEXT);
  483:
  484:                    // Tracker
  485:                    MenuItem subMenu4Tracker = subMenu4.getItem(0);
  486:                    subMenu4Tracker.setIcon(android.R.drawable.ic_menu_mylocation)
;
  487:
  488:                    subMenu4Tracker
  489:                                    .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  490:                                            @Override
  491:                                            public boolean onMenuItemClick(MenuIte
m item) {
  492:
  493:                                                    finish();
  494:
  495:                                                    return false;
  496:                                            }
  497:                                    });
  498:
  499:                    // Help
  500:                    MenuItem subMenu4Help = subMenu4.getItem(1);
  501:                    subMenu4Help.setIcon(android.R.drawable.ic_menu_help);
  502:
  503:                    subMenu4Help
  504:                                    .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  505:                                            @Override
  506:                                            public boolean onMenuItemClick(MenuIte
m item) {
  507:                                                    dialog = new Dialog(ItemMenu.t
his);
  508:                                                    dialog.setContentView(R.layout
.help_options_item);
  509:                                                    Window w = dialog.getWindow();
  510:                                                    w.setGravity(Gravity.FILL_HORI
ZONTAL
  511:                                                                    | Gravity.FILL
_VERTICAL);
  512:                                                    w.setLayout(ViewGroup.LayoutPa
rams.FILL_PARENT,
  513:                                                                    ViewGroup.Layo
utParams.FILL_PARENT);
  514:                                                    dialog.setTitle("CLODA: Help")
;
  515:                                                    LinearLayout v = (LinearLayout
) dialog
  516:                                                                    .findViewById(
R.id.helpwindowItemMenu);
  517:                                                    if (v != null) {
  518:                                                            v.setClickable(true);
  519:                                                            v.setOnClickListener(n
ew OnClickListener() {
```

```
  520:                                                                public void on
Click(View v) {
  521:                                                                        dialog
.dismiss();
  522:                                                                }
  523:                                                        });
  524:                                                }
  525:                                                dialog.show();
  526:
  527:                                                return false;
  528:                                        }
  529:                                });
  530:
  531:                /************************************* END OF MAIN MENU **
************************************************************/
  532:
  533:                return super.onCreateOptionsMenu(menu);
  534:        }
  535: }
```

```
    1: package dmsl.ItemScan;
    2:
    3: import dmsl.crowdspaceSidemenu.R;
    4: import android.app.Activity;
    5: import android.content.Intent;
    6: import android.os.Bundle;
    7: import android.view.View;
    8: import android.widget.Button;
    9: import android.widget.EditText;
   10: import android.widget.TextView;
   11: import android.widget.Toast;
   12:
   13: /**
   14:  *
   15:  * @author Julia Metochi
   16:  *
   17:  *         This activity is called when neither the product's name or product'
s
   18:  *         price is known. It asks the user for them and passes user's input t
o
   19:  *         the main activity.
   20:  *
   21:  */
   22: public class AskForUserInput extends Activity {
   23:
   24:         public static final int MAX_TITLE_LENGTH = 60;
   25:         public static final int MAX_PRICE = 1000000;
   26:
   27:         boolean bFlag = false;
   28:         boolean pFlag = false;
   29:
   30:         public AskForUserInput() {
   31:         }
   32:
   33:         @Override
   34:         protected void onCreate(Bundle savedInstanceState) {
   35:                 super.onCreate(savedInstanceState);
   36:                 setContentView(R.layout.prompt);
   37:
   38:                 final EditText etTitle = (EditText) findViewById(R.id.etTitle)
;
   39:                 final EditText etBrand = (EditText) findViewById(R.id.etBrand)
;
   40:                 final EditText etPrice = (EditText) findViewById(R.id.etPrice)
;
   41:
   42:                 TextView tvTitle = (TextView) findViewById(R.id.title);
   43:                 TextView tvBrand = (TextView) findViewById(R.id.Brand);
   44:                 TextView tvPrice = (TextView) findViewById(R.id.Price);
   45:
   46:                 Bundle getB = getIntent().getExtras();
   47:                 final String barcode = getB.getString("barcode");
   48:                 final String title = getB.getString("title");
   49:                 final String brand = getB.getString("brand");
   50:                 final String price = getB.getString("price");
   51:                 final float pr = Float.parseFloat(price);
   52:
   53:                 if (brand != null) {
   54:                         tvBrand.setText(brand);
   55:                         etBrand.setVisibility(View.GONE);
   56:                 } else
   57:                         bFlag = true;
   58:                 if (price != null) {
   59:                         tvPrice.setText("$" + price);
   60:                         etPrice.setVisibility(View.GONE);
   61:                 } else
   62:                         pFlag = true;
   63:
   64:                 tvTitle.setText(title);
   65:                 etTitle.setVisibility(View.GONE);
   66:
   67:                 Button bOk = (Button) findViewById(R.id.ok);
   68:                 Button bCancel = (Button) findViewById(R.id.cancel);
   69:
   70:                 bOk.setOnClickListener(new View.OnClickListener() {
   71:
   72:                         @Override
   73:                         public void onClick(View v) {
   74:
   75:                                 boolean success = false;
   76:                                 if (bFlag || pFlag) {
   77:                                         if (bFlag) {
   78:                                                 if (etBrand.getText().toString
().length() == 0) {
   79:                                                         // If any of the field
s are left empty a toast
   80:                                                         // message
   81:                                                         // appears
   82:                                                         // and the activity st
ill waits for valid input.
   83:                                                         Toast.makeText(getAppl
icationContext(),
   84:                                                                         "One o
r all of the fields are empty.",
   85:                                                                         Toast.
LENGTH_SHORT).show();
   86:                                                 }
   87:                                         }
   88:                                         if (pFlag) {
   89:                                                 if (etPrice.getText().toString
().length() == 0) {
   90:                                                         // If any of the field
s are left empty a toast
   91:                                                         // message
   92:                                                         // appears
   93:                                                         // and the activity st
ill waits for valid input.
   94:                                                         Toast.makeText(getAppl
icationContext(),
   95:                                                                         "One o
r all of the fields are empty.",
   96:                                                                         Toast.
LENGTH_SHORT).show();
   97:                                                 } else {
   98:                                                         try {
   99:                                                                 String strPric
e = etPrice.getText().toString()
  100:
.replace(',', '.');
  101:
  102:                                                                 if (pr >= MAX_
PRICE) {
```

```
  103:                                                                  Toast.
makeText(
  104:
        getApplicationContext(),
  105:
        "Extreme Value, only prices below "
  106:
                        + MAX_PRICE
  107:
                        + " are accepted.",
  108:
        Toast.LENGTH_SHORT).show();
  109:                                                                  succes
s = true;
  110:                                                                  }
  111:                                                          } catch (NumberFormatE
xception nfe) {
  112:                                                                  // Indication
that an error occurred.
  113:                                                                  success = fals
e;
  114:                                                          }
  115:                                                          if (!success) {
  116:                                                                  Toast.makeText
(getApplicationContext(),
  117:
"Invalid price format",
  118:
Toast.LENGTH_SHORT).show();
  119:                                                          }
  120:                                                  }
  121:                                          }
  122:                                  }
  123:
  124:                                  // else {
  125:
  126:                                  Intent i = new Intent();
  127:                                  Bundle b = new Bundle();
  128:
  129:                                  // Data to be passed to the main activity is p
ut in
  130:                                  // the bundle.
  131:
  132:                                  b.putString("title", title);
  133:                                  if (bFlag)
  134:                                          b.putString("brand", etBrand.getText()
.toString());
  135:                                  else
  136:                                          b.putString("brand", brand);
  137:                                  if (pFlag) {
  138:                                          float timi = Float.parseFloat(etPrice.
getText().toString());
  139:                                          b.putFloat("price", timi);
  140:                                  } else
  141:                                          b.putFloat("price", pr);
  142:                                  b.putString("barcode", barcode);
  143:                                  i.putExtras(b);
  144:                                  setResult(RESULT_OK, i);
  145:                                  success = true;
  146:                                  finish();
  147:
  148:                          }
  149:                  });
  150:
  151:                  bCancel.setOnClickListener(new View.OnClickListener() {
  152:
  153:                          @Override
  154:                          public void onClick(View v) {
  155:                                  setResult(RESULT_CANCELED);
  156:                                  finish();
  157:                          }
  158:                  });
  159:          }
  160:
  161: }
```

```
    1: package dmsl.FloorMode;
    2:
    3: import com.cy.wifi.algorithms.LogRecord;
    4:
    5: public class LogRecordFloor extends LogRecord {
    6:         private boolean[] floor;
    7:
    8:         public LogRecordFloor(String bssid, int rss, int noOfFloors) {
    9:                 super(bssid, rss);
   10:                 this.floor = new boolean[noOfFloors];
   11:
   12:                 for (int i = 0; i < this.floor.length; ++i) {
   13:                         this.floor[i] = true;
   14:                 }
   15:         }
   16:
   17:         /**
   18:          * Set in which floor the macAddress belongs at final
   19:          *
   20:          * @param validFloor
   21:          * @param floor
   22:          */
   23:         public void setFloor(boolean validFloor, int floor) {
   24:                 this.floor[floor] = validFloor;
   25:         }
   26:
   27:         /**
   28:          * Get the result of the specific number of the floor
   29:          *
   30:          * @param noOfFloor
   31:          * @return
   32:          */
   33:         public boolean getFloor(int noOfFloor) {
   34:                 return floor[noOfFloor];
   35:         }
   36:
   37:         public String toString() {
   38:                 String str = new String();
   39:                 str = this.getBssid() + " " + String.valueOf(this.getRss()) +
"\n"
   40:                                 + " " + this.floor[0] + " " + this.floor[1];
   41:                 return str;
   42:         }
   43: }
```

```
    1: /*
    2:  * Copyright (c) 2011, KIOS Research Center and Data Management Systems Lab,
    3:  * University of Cyprus. All rights reserved.
    4:  *
    5:  * Redistribution and use in source and binary forms, with or without modifica
tion,
    6:  * are permitted provided that the following conditions are met:
    7:  *
    8:  *    * Redistributions of source code must retain the above copyright notice,
 this
    9:  *      list of conditions and the following disclaimer.
   10:  *    * Redistributions in binary form must reproduce the above copyright noti
ce,
   11:  *      this list of conditions and the following disclaimer in the documentat
ion
   12:  *      and/or other materials provided with the distribution.
   13:  *    * Neither the name of the Sony Ericsson Mobile Communication AB nor the
names
   14:  *      of its contributors may be used to endorse or promote products derived
 from
   15:  *      this software without specific prior written permission.
   16:  *
   17:  * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 AND
   18:  * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLI
ED
   19:  * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISC
LAIMED.
   20:  * IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DI
RECT,
   21:  * INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDI
NG,
   22:  * BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF US
E,
   23:  * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEOR
Y OF
   24:  * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIG
ENCE
   25:  * OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF A
DVISED
   26:  * OF THE POSSIBILITY OF SUCH DAMAGE.
   27:  */
   28:
   29: package dmsl.crowdspace.airplace;
   30:
   31: public class LogRecord {
   32:
   33:         private long ts;
   34:         private double lng;
   35:         private double lat;
   36:         private float heading;
   37:         private String bssid;
   38:         private int rss;
   39:
   40:         public LogRecord(long ts, double lat, double lng, float heading,
   41:                         String bssid, int rss) {
   42:                 super();
   43:                 this.ts = ts;
   44:                 this.lng = lng;
   45:                 this.lat = lat;
   46:                 this.heading = heading;
   47:                 this.bssid = bssid;
   48:                 this.rss = rss;
   49:         }
   50:
   51:         public String toString() {
   52:                 String str = new String();
   53:                 str = String.valueOf(ts) + " " + String.valueOf(lat) + " "
   54:                                 + String.valueOf(lng) + " " + String.valueOf(h
eading) + " "
   55:                                 + String.valueOf(bssid) + " " + String.valueOf
(rss) + "\n";
   56:                 return str;
   57:         }
   58:
   59: }
```

```
    1: /*
    2:  * Copyright (c) 2011, KIOS Research Center and Data Management Systems Lab,
    3:  * University of Cyprus. All rights reserved.
    4:  *
    5:  * Redistribution and use in source and binary forms, with or without modifica
tion,
    6:  * are permitted provided that the following conditions are met:
    7:  *
    8:  *    * Redistributions of source code must retain the above copyright notice,
 this
    9:  *      list of conditions and the following disclaimer.
   10:  *    * Redistributions in binary form must reproduce the above copyright noti
ce,
   11:  *      this list of conditions and the following disclaimer in the documentat
ion
   12:  *      and/or other materials provided with the distribution.
   13:  *    * Neither the name of the Sony Ericsson Mobile Communication AB nor the
names
   14:  *      of its contributors may be used to endorse or promote products derived
 from
   15:  *      this software without specific prior written permission.
   16:  *
   17:  * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 AND
   18:  * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLI
ED
   19:  * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISC
LAIMED.
   20:  * IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DI
RECT,
   21:  * INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDI
NG,
   22:  * BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF US
E,
   23:  * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEOR
Y OF
   24:  * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIG
ENCE
   25:  * OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF A
DVISED
   26:  * OF THE POSSIBILITY OF SUCH DAMAGE.
   27:  */
   28:
   29: package dmsl.crowdspace.airplace;
   30:
   31: import android.content.Context;
   32: import android.graphics.drawable.Drawable;
   33: import android.widget.ImageView;
   34: import android.widget.LinearLayout;
   35: import android.widget.TextView;
   36:
   37: public class IconifiedTextView extends LinearLayout {
   38:
   39:         private TextView mText;
   40:         private ImageView mIcon;
   41:
   42:         public IconifiedTextView(Context context, IconifiedText aIconifiedText
) {
   43:                 super(context);
   44:
   45:                 /*
   46:                  * First Icon and the Text to the right (horizontal), not abov
e and
   47:                  * below (vertical)
   48:                  */
   49:                 this.setOrientation(HORIZONTAL);
   50:
   51:                 mIcon = new ImageView(context);
   52:                 mIcon.setImageDrawable(aIconifiedText.getIcon());
   53:                 // left, top, right, bottom
   54:                 mIcon.setPadding(0, 2, 5, 0); // 5px to the right
   55:
   56:                 addView(mIcon, new LinearLayout.LayoutParams(LayoutParams.WRAP
_CONTENT,
   57:                                 LayoutParams.WRAP_CONTENT));
   58:
   59:                 mText = new TextView(context);
   60:                 mText.setText(aIconifiedText.getText());
   61:
   62:                 /* Now the text (after the icon) */
   63:                 addView(mText, new LinearLayout.LayoutParams(LayoutParams.WRAP
_CONTENT,
   64:                                 LayoutParams.WRAP_CONTENT));
   65:         }
   66:
   67:         public void setText(String words) {
   68:                 mText.setText(words);
   69:         }
   70:
   71:         public void setIcon(Drawable bullet) {
   72:                 mIcon.setImageDrawable(bullet);
   73:         }
   74: }
```

```
    1: package dmsl.ItemScan;
    2:
    3: /**
    4:  * Create JSON Object to send it to couchDB
    5:  *
    6:  * @author Julia Metochi
    7:  *
    8:  */
    9: public class ObjectToSend {
   10:         private String title = null, barcode = null, brand = null, price = nul
l,
   11:                         pointX = null, pointY = null, floor = null, image = nu
ll,
   12:                         _id = null, _rev = null;
   13:
   14:         public ObjectToSend(ProductObject obj) {
   15:                 barcode = obj.getBarcode();
   16:                 title = obj.getTitle();
   17:                 brand = obj.getBrand();
   18:                 price = Float.toString(obj.getPrice());
   19:                 pointX = Double.toString(obj.getPointX());
   20:                 pointY = Double.toString(obj.getPointY());
   21:                 floor = String.valueOf(obj.getFloor());
   22:                 image = obj.getImage();
   23:
   24:         }
   25:
   26:         /*****************************************************************
*************************************************************/
   27:
   28:         public ObjectToSend(ProductObject obj, int num) {
   29:                 barcode = obj.getBarcode();
   30:                 title = obj.getTitle();
   31:                 brand = obj.getBrand();
   32:                 price = Float.toString(obj.getPrice());
   33:                 pointX = Double.toString(obj.getPointX());
   34:                 pointY = Double.toString(obj.getPointY());
   35:                 floor = String.valueOf(obj.getFloor());
   36:                 image = obj.getImage();
   37:                 _id = obj.getDocId();
   38:                 _rev = obj.getDocRev();
   39:         }
   40:
   41: }
```

```
  1: package dmsl.crowdspaceSidemenu;
  2:
  3: import java.io.File;
  4: import java.io.FileNotFoundException;
  5: import java.io.FileOutputStream;
  6: import java.io.IOException;
  7: import java.util.ArrayList;
  8: import java.util.Calendar;
  9: import java.util.Date;
 10: import java.util.List;
 11:
 12: import com.actionbarsherlock.app.SherlockActivity;
 13: import com.actionbarsherlock.view.Menu;
 14: import com.actionbarsherlock.view.MenuItem;
 15: import com.actionbarsherlock.view.SubMenu;
 16: import com.google.zxing.client.android.CaptureActivity;
 17:
 18: //import com.cy.wifi.algorithms.LogRecord;
 19:
 20: import dmsl.crowdspace.airplace.BooleanObservable;
 21: import dmsl.crowdspace.airplace.Heading;
 22: import dmsl.crowdspace.airplace.LogRecord;
 23: import dmsl.crowdspace.airplace.Prefs;
 24: import dmsl.crowdspace.airplace.Prefs_logger;
 25: import dmsl.crowdspace.airplace.SimpleWifiManager;
 26: import dmsl.crowdspace.airplace.WifiReceiver;
 27:
 28: import android.app.Activity;
 29: import android.app.Dialog;
 30: import android.app.ProgressDialog;
 31: import android.content.Context;
 32: import android.content.Intent;
 33: import android.content.SharedPreferences;
 34: import android.net.wifi.ScanResult;
 35: import android.os.Bundle;
 36: import android.os.Handler;
 37: import android.os.Message;
 38: import android.preference.PreferenceManager;
 39: import android.view.Gravity;
 40: //import android.view.Menu;
 41: //import android.view.MenuInflater;
 42: //import android.view.MenuItem;
 43: import android.view.View;
 44: import android.view.ViewGroup;
 45: import android.view.Window;
 46: import android.view.View.OnClickListener;
 47: import android.webkit.WebSettings;
 48: import android.webkit.WebView;
 49: import android.widget.Button;
 50: import android.widget.LinearLayout;
 51: import android.widget.TextView;
 52: import android.widget.Toast;
 53:
 54: /**
 55:  *
 56:  * @author Julia Metochi
 57:  *
 58:  */
 59: public class MainActivityLogger extends SherlockActivity {
 60:
 61:         WebView mWebview;
 62:         static int floor;
 63:         private Dialog dialog = null;
 64:         private static double curLon = 0;
 65:         private static double curLat = 0;
 66:         // WiFi manager
 67:         private SimpleWifiManager wifi;
 68:         // WiFi Receiver
 69:         private WifiReceiver receiverWifi;
 70:         // TextView showing the current scan results
 71:         private TextView scanAPs;
 72:
 73:         // TextView trackingInfo
 74:         private static TextView info;
 75:         public Heading h;
 76:
 77:         // Preferences name for indoor and outdoor
 78:         public static final String SHARED_PREFS_INDOOR = "Indoor_Preferences";
 79:         private SharedPreferences prefs;
 80:
 81:         // By default tracker selected
 82:         public static int flag = -1;
 83:
 84:         public static Coordinates LonLat = new Coordinates();
 85:
 86:         // ProgressDialog
 87:         private ProgressDialog progressDialog;
 88:         private static final int PROGRESS_DIALOG = 0;
 89:
 90:         // Flag to show if there is an ongoing progress
 91:         private Boolean inProgress;
 92:
 93:         private String samples_num = null;
 94:
 95:         private String filename_rss, folder_path;
 96:         private boolean recordingWifi = false;
 97:
 98:         private final BooleanObservable trackMe = new BooleanObservable();
 99:
100:         // The latest scan list of APs
101:         ArrayList<LogRecord> latestScanList;
102:
103:         // Samples list
104:         private final ArrayList<ArrayList<LogRecord>> samples = new ArrayList<
ArrayList<LogRecord>>();
105:
106:         SubMenu subMenu1;
107:         static final Handler myHandler = new Handler() {
108:
109:                 /**
110:                  * Handle message send from javaScript
111:                  */
112:                 @Override
113:                 public void handleMessage(Message msg) {
114:                         // TODO Auto-generated method stub
115:                         super.handleMessage(msg);
116:                         LonLat = (Coordinates) msg.obj;
117:                         curLon = LonLat.getLon();
118:                         curLat = LonLat.getLat();
119:                         updateInfoLabel();
120:                 }
121:
```

```
  122:         };
  123:
  124:         /*********************************************************************
*****************************************************/
  125:
  126:         /** Called when the activity is first created. */
  127:         @Override
  128:         public void onCreate(Bundle savedInstanceState) {
  129:
  130:                 super.onCreate(savedInstanceState);
  131:                 setContentView(R.layout.main_logger);
  132:                 curLat = 0;
  133:                 curLon = 0;
  134:                 h = new Heading(this);
  135:                 PreferenceManager.setDefaultValues(this, "Indoor_Preferences",
  136:                                 MODE_PRIVATE, R.xml.preferences, true);
  137:                 prefs = getSharedPreferences("Indoor_Preferences", MODE_PRIVAT
E);
  138:                 mWebview = (WebView) findViewById(R.id.map);
  139:                 info = (TextView) findViewById(R.id.trackingInfoLabel);
  140:
  141:                 final MyJavaScriptInterface myJavaScriptInterface = new MyJava
ScriptInterface(
  142:                                 this);
  143:                 mWebview.addJavascriptInterface(myJavaScriptInterface,
  144:                                 "myAndroidFunctions");
  145:                 mWebview.getSettings().setJavaScriptEnabled(true);
  146:                 mWebview.loadUrl("http://juliavm.in.cs.ucy.ac.cy/uniBuilding.h
tml");
  147:                 updateInfoLabel();
  148:
  149:                 // Create new receiver to get broadcasts
  150:                 receiverWifi = new SimpleWifiReceiver();
  151:
  152:                 scanAPs = (TextView) findViewById(R.id.detectedAPs);
  153:
  154:                 // WiFi manager to manage scans
  155:                 wifi = new SimpleWifiManager(MainActivityLogger.this);
  156:                 wifi.setScanResultsTextView(scanAPs);
  157:                 // ----------------------------------------------------------
----------------
  158:
  159:                 updateInfoLabel();
  160:
  161:         }
  162:
  163:         /*********************************************************************
*****************************************************/
  164:
  165:         /**
  166:          * Update label on the bottom right of the screen
  167:          */
  168:         public static void updateInfoLabel() {
  169:                 StringBuilder sb = new StringBuilder();
  170:                 int head = (int) Heading.azimuth;
  171:                 sb.append("H: ");
  172:                 sb.append(head);
  173:                 sb.append("\nX: ");
  174:                 sb.append(curLon);
  175:                 sb.append("\nY: ");
  176:                 sb.append(curLat);
  177:                 if (info != null) {
  178:                         info.setText(sb.toString());
  179:                 }
  180:
  181:         }
  182:
  183:         /*********************************************************************
*****************************************************/
  184:
  185:         /**
  186:          * onPause
  187:          */
  188:         @Override
  189:         protected void onPause() {
  190:                 // TODO Auto-generated method stub
  191:                 super.onPause();
  192:                 h.pause();
  193:         }
  194:
  195:         /*********************************************************************
*****************************************************/
  196:
  197:         /**
  198:          * onResume
  199:          */
  200:         @Override
  201:         protected void onResume() {
  202:                 super.onResume();
  203:                 updateInfoLabel();
  204:                 h.resume();
  205:         }
  206:
  207:         /*********************************************************************
*****************************************************/
  208:
  209:         /**
  210:          * onRestart
  211:          */
  212:         @Override
  213:         protected void onRestart() {
  214:                 super.onRestart();
  215:                 flag = 1;
  216:         }
  217:
  218:         /*********************************************************************
*****************************************************/
  219:
  220:         /**
  221:          * Create the dialog
  222:          *
  223:          * @param id
  224:          *            the id of dialog to create
  225:          * */
  226:         @Override
  227:         protected Dialog onCreateDialog(int id) {
  228:                 switch (id) {
  229:                 case PROGRESS_DIALOG:
  230:                         progressDialog = new ProgressDialog(MainActivityLogger
.this);
  231:                         progressDialog.setProgressStyle(ProgressDialog.STYLE_H
ORIZONTAL);
```

```
  232:                         progressDialog.setMessage("Scanning in progress...");
  233:                         progressDialog.setCancelable(false);
  234:                         return progressDialog;
  235:                 default:
  236:                         return null;
  237:                 }
  238:         }
  239:
  240:         /*************************************************************************
*************************************************************/
  241:
  242:         /**
  243:          * Prepare the dialog. Sets progress to 0 and max to samples number
  244:          *
  245:          * @param id
  246:          *            the id of dialog to prepare
  247:          * @param dialog
  248:          *
  249:          * */
  250:         @Override
  251:         protected void onPrepareDialog(int id, Dialog dialog) {
  252:                 switch (id) {
  253:                 case PROGRESS_DIALOG:
  254:                         progressDialog.setProgress(0);
  255:                         progressDialog.setMax(Integer.parseInt(samples_num));
  256:                         break;
  257:                 }
  258:         }
  259:
  260:         /*************************************************************************
*************************************************************/
  261:
  262:         /**
  263:          * Create Options Menu
  264:          */
  265:         @Override
  266:         public boolean onCreateOptionsMenu(Menu menu) {
  267:
  268:                 getSupportActionBar().setHomeButtonEnabled(false);
  269:                 getSupportActionBar().setTitle("Logger");
  270:
  271:                 /************************************* START RECORDING ***
*************************************************************/
  272:                 subMenu1 = menu.addSubMenu("Start Recording");
  273:                 MenuItem subMenu1ItemStartRecording = subMenu1.getItem();
  274:                 subMenu1ItemStartRecording.setIcon(android.R.drawable.ic_menu_
save);
  275:                 subMenu1ItemStartRecording
  276:                                 .setShowAsAction(MenuItem.SHOW_AS_ACTION_ALWAY
S
  277:                                                 | MenuItem.SHOW_AS_ACTION_WITH
_TEXT);
  278:                 subMenu1ItemStartRecording
  279:                                 .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  280:
  281:                                         @Override
  282:                                         public boolean onMenuItemClick(MenuIte
m item) {
  283:                                                 startWifiRecording();
  284:                                                 return false;
  285:                                         }
  286:                                 });
  287:
  288:                 /************************************** END OF START RECO
RDING *****************************************************/
  289:
  290:                 /************************************* MAIN MENU *********
*************************************************************/
  291:                 SubMenu subMenu2 = menu.addSubMenu("Main Menu");
  292:                 subMenu2.add("Tracker");
  293:                 subMenu2.add("Preferences");
  294:                 subMenu2.add("Help");
  295:                 subMenu2.add("About");
  296:
  297:                 MenuItem subMenu2ItemMainMenu = subMenu2.getItem();
  298:                 subMenu2ItemMainMenu
  299:                                 .setIcon(R.drawable.abs__ic_menu_moreoverflow_
normal_holo_dark);
  300:                 subMenu2ItemMainMenu.setShowAsAction(MenuItem.SHOW_AS_ACTION_A
LWAYS
  301:                                 | MenuItem.SHOW_AS_ACTION_WITH_TEXT);
  302:
  303:                 // Tracker
  304:                 MenuItem subMenu2Tracker = subMenu2.getItem(0);
  305:                 subMenu2Tracker.setIcon(android.R.drawable.ic_menu_mylocation)
;
  306:
  307:                 subMenu2Tracker
  308:                                 .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  309:                                         @Override
  310:                                         public boolean onMenuItemClick(MenuIte
m item) {
  311:
  312:                                                 finish();
  313:
  314:                                                 return false;
  315:                                         }
  316:                                 });
  317:
  318:                 // Preferences
  319:                 MenuItem subMenu2Prefs = subMenu2.getItem(1);
  320:                 subMenu2Prefs.setIcon(android.R.drawable.ic_menu_preferences);
  321:
  322:                 subMenu2Prefs
  323:                                 .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  324:                                         @Override
  325:                                         public boolean onMenuItemClick(MenuIte
m item) {
  326:
  327:                                                 Intent prefs = new Intent(Main
ActivityLogger.this,
  328:                                                                 Prefs_logger.c
lass);
  329:                                                 startActivity(prefs);
  330:
  331:                                                 return false;
  332:                                         }
  333:                                 });
  334:
```

```
  335:                // Help
  336:                MenuItem subMenu2Help = subMenu2.getItem(2);
  337:                subMenu2Help.setIcon(android.R.drawable.ic_menu_help);
  338:
  339:                subMenu2Help
  340:                        .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  341:                                @Override
  342:                                public boolean onMenuItemClick(MenuIte
m item) {
  343:                                        dialog = new Dialog(MainActivi
tyLogger.this);
  344:                                        dialog.setContentView(R.layout
.help_logger);
  345:                                        Window w = dialog.getWindow();
  346:                                        w.setGravity(Gravity.FILL_HORI
ZONTAL
  347:                                                        | Gravity.FILL
_VERTICAL);
  348:                                        w.setLayout(ViewGroup.LayoutPa
rams.FILL_PARENT,
  349:                                                        ViewGroup.Layo
utParams.FILL_PARENT);
  350:                                        dialog.setTitle("CrowdSpace: H
elp");
  351:                                        LinearLayout v = (LinearLayout
) dialog
  352:                                                        .findViewById(
R.id.helpwindowLogger);
  353:                                        if (v != null) {
  354:                                                v.setClickable(true);
  355:                                                v.setOnClickListener(n
ew OnClickListener() {
  356:                                                        public void on
Click(View v) {
  357:                                                                dialog
.dismiss();
  358:                                                        }
  359:                                                });
  360:                                        }
  361:                                        dialog.show();
  362:
  363:                                        return false;
  364:                                }
  365:                        });
  366:
  367:                // About
  368:                MenuItem subMenu2About = subMenu2.getItem(3);
  369:                subMenu2About.setIcon(android.R.drawable.ic_menu_info_details)
;
  370:
  371:                subMenu2About
  372:                        .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  373:                                @Override
  374:                                public boolean onMenuItemClick(MenuIte
m item) {
  375:                                        dialog = new Dialog(MainActivi
tyLogger.this);
  376:                                        dialog.setContentView(R.layout
.about);
  377:                                        Window w = dialog.getWindow();
  378:                                        w.setGravity(Gravity.FILL_HORI
ZONTAL
  379:                                                        | Gravity.FILL
_VERTICAL);
  380:                                        w.setLayout(ViewGroup.LayoutPa
rams.FILL_PARENT,
  381:                                                        ViewGroup.Layo
utParams.FILL_PARENT);
  382:                                        dialog.setTitle("CrowdSpace: A
bout");
  383:                                        LinearLayout v = (LinearLayout
) dialog
  384:                                                        .findViewById(
R.id.aboutwindow);
  385:                                        if (v != null) {
  386:                                                v.setClickable(true);
  387:                                                v.setOnClickListener(n
ew OnClickListener() {
  388:                                                        public void on
Click(View v) {
  389:                                                                dialog
.dismiss();
  390:                                                        }
  391:                                                });
  392:                                        }
  393:                                        dialog.show();
  394:                                        return false;
  395:                                }
  396:                        });
  397:
  398:                /************************************** END OF MAIN MENU **
************************************************************/
  399:
  400:                /************************************** FLOORS ***********
************************************************************/
  401:                SubMenu subMenu3 = menu.addSubMenu("Floors");
  402:                subMenu3.add("Floor -1");
  403:                subMenu3.add("Floor  0");
  404:                subMenu3.add("Floor  1");
  405:                subMenu3.add("Floor  2");
  406:
  407:                // Floor -1
  408:                MenuItem subMenu3ItemFloorPlinEna = subMenu3.getItem(0);
  409:                subMenu3ItemFloorPlinEna
  410:                        .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  411:
  412:                                @Override
  413:                                public boolean onMenuItemClick(MenuIte
m item) {
  414:                                        floor = -1;
  415:                                        mWebview.loadUrl("javascript:h
ideShowOverlay();");
  416:
  417:                                        mWebview.loadUrl("javascript:v
iewUniMap(" + floor
  418:                                                        + ");");
  419:                                        return false;
  420:                                }
  421:                        });
```

```
  422:
  423:                // Floor 0
  424:                MenuItem subMenu3ItemFloorZero = subMenu3.getItem(1);
  425:                subMenu3ItemFloorZero
  426:                                .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  427:
  428:                                        @Override
  429:                                        public boolean onMenuItemClick(MenuIte
m item) {
  430:                                                floor = 0;
  431:                                                mWebview.loadUrl("javascript:h
ideShowOverlay();");
  432:                                                mWebview.loadUrl("javascript:v
iewUniMap(" + floor
  433:                                                                + ");");
  434:                                                return false;
  435:                                        }
  436:                                });
  437:
  438:                // Floor 1
  439:                MenuItem subMenu3ItemFloorOne = subMenu3.getItem(2);
  440:                subMenu3ItemFloorOne
  441:                                .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  442:
  443:                                        @Override
  444:                                        public boolean onMenuItemClick(MenuIte
m item) {
  445:                                                floor = 1;
  446:                                                mWebview.loadUrl("javascript:h
ideShowOverlay();");
  447:                                                // mWebview.loadUrl("javascrip
t:initialize();");
  448:                                                mWebview.loadUrl("javascript:v
iewUniMap(" + floor
  449:                                                                + ");");
  450:                                                return false;
  451:                                        }
  452:                                });
  453:
  454:                // Floor 2
  455:                MenuItem subMenu3ItemFloorTwo = subMenu3.getItem(3);
  456:                subMenu3ItemFloorTwo
  457:                                .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  458:
  459:                                        @Override
  460:                                        public boolean onMenuItemClick(MenuIte
m item) {
  461:                                                floor = 2;
  462:                                                mWebview.loadUrl("javascript:h
ideShowOverlay();");
  463:                                                mWebview.loadUrl("javascript:v
iewUniMap(" + floor
  464:                                                                + ");");
  465:                                                return false;
  466:                                        }
  467:                                });
  468:
  469:                return super.onCreateOptionsMenu(menu);
  470:        }
  471:
  472:        /*************************************************************************
**************************************************/
  473:
  474:        /**
  475:         * Select from Menu
  476:         */
  477:        @Override
  478:        public boolean onOptionsItemSelected(MenuItem item) {
  479:
  480:                switch (item.getItemId()) {
  481:
  482:                case R.id.Prefs:
  483:                        Intent prefs = new Intent(this, Prefs_logger.class);
  484:                        startActivity(prefs);
  485:                        return true;
  486:
  487:                case R.id.About:
  488:                        dialog = new Dialog(this);
  489:                        dialog.setContentView(R.layout.about);
  490:                        Window w = dialog.getWindow();
  491:                        w.setGravity(Gravity.FILL_HORIZONTAL | Gravity.FILL_VE
RTICAL);
  492:                        w.setLayout(ViewGroup.LayoutParams.FILL_PARENT,
  493:                                        ViewGroup.LayoutParams.FILL_PARENT);
  494:                        dialog.setTitle("CrowdSpace Context Scanner");
  495:                        LinearLayout v = (LinearLayout) dialog
  496:                                        .findViewById(R.id.aboutwindow);
  497:                        if (v != null) {
  498:                                v.setClickable(true);
  499:                                v.setOnClickListener(new OnClickListener() {
  500:                                        public void onClick(View v) {
  501:                                                dialog.dismiss();
  502:                                        }
  503:                                });
  504:                        }
  505:                        dialog.show();
  506:                        return true;
  507:
  508:                case R.id.Help:
  509:                        dialog = new Dialog(this);
  510:                        dialog.setContentView(R.layout.help);
  511:                        w = dialog.getWindow();
  512:                        w.setGravity(Gravity.FILL_HORIZONTAL | Gravity.FILL_VE
RTICAL);
  513:                        w.setLayout(ViewGroup.LayoutParams.FILL_PARENT,
  514:                                        ViewGroup.LayoutParams.FILL_PARENT);
  515:                        dialog.setTitle("CrowdSpace Context Scanner");
  516:                        v = (LinearLayout) dialog.findViewById(R.id.helpwindow
);
  517:                        if (v != null) {
  518:                                v.setClickable(true);
  519:                                v.setOnClickListener(new OnClickListener() {
  520:                                        public void onClick(View v) {
  521:                                                dialog.dismiss();
  522:                                        }
  523:                                });
  524:                        }
  525:                        dialog.show();
  526:                        return true;
```

```
  527:
  528:                }
  529:                return false;
  530:        }
  531:
  532:        /*****************************************************************
*********************************************************/
  533:
  534:        /**
  535:         * start wifi recording
  536:         */
  537:        public void startWifiRecording() {
  538:                samples_num = prefs.getString("samples_num", "n/a");
  539:                String samples_interval = (String) prefs.getString("samples_in
terval",
  540:                                "n/a");
  541:
  542:                if (samples_num.equals("n/a") || samples_num.trim().equals("")
) {
  543:                        Toast.makeText(
  544:                                        this,
  545:                                        "Samples number not specified\nGo to M
enu::Preferences::Sampling Settings::Samples Number",
  546:                                        Toast.LENGTH_LONG).show();
  547:                        recordingWifi = false;
  548:                        return;
  549:                }
  550:
  551:                folder_path = (String) prefs.getString("folder_browser", "n/a"
);
  552:                if (folder_path.equals("n/a") || folder_path.equals("")) {
  553:                        Toast.makeText(
  554:                                        this,
  555:                                        "Folder path not specified\nGo to Menu
::Preferences::Storing Settings::Folder",
  556:                                        Toast.LENGTH_LONG).show();
  557:                        recordingWifi = false;
  558:                        return;
  559:
  560:                } else if ((!(new File(folder_path).canWrite()))) {
  561:                        Toast.makeText(
  562:                                        this,
  563:                                        "Folder path is not writable\nGo to Me
nu::Preferences::Storing Settings::Folder",
  564:                                        Toast.LENGTH_LONG).show();
  565:
  566:                        recordingWifi = false;
  567:                        return;
  568:                }
  569:
  570:                filename_rss = (String) prefs.getString("filename_log", "n/a")
;
  571:                if (filename_rss.equals("n/a") || filename_rss.equals("")) {
  572:                        Toast.makeText(
  573:                                        this,
  574:                                        "Filename of RSS log not specified\nGo
 to Menu::Preferences::Storing Settings::Filename",
  575:                                        Toast.LENGTH_LONG).show();
  576:
  577:                        recordingWifi = false;
  578:                        return;
  579:                }
  580:
  581:                if (!wifi.getIsScanning()) {
  582:
  583:                        if (samples_interval.equals("n/a")
  584:                                        || samples_interval.trim().equals(""))
 {
  585:                                Toast.makeText(
  586:                                                this,
  587:                                                "Samples interval not specifie
d\nGo to Menu::Preferences::Sampling Settings::Samples Interval",
  588:                                                Toast.LENGTH_LONG).show();
  589:                                recordingWifi = false;
  590:                                return;
  591:                        }
  592:                        subMenu1.getItem().setCheckable(false);
  593:                        wifi.startScan(receiverWifi, samples_interval);
  594:                }
  595:
  596:                Toast toast = Toast.makeText(getApplicationContext(),
  597:                                "Start Recording WiFi Fingerprint", Toast.LENG
TH_SHORT);
  598:                toast.show();
  599:
  600:                subMenu1.getItem().setCheckable(false);
  601:                showDialog(PROGRESS_DIALOG);
  602:
  603:        }
  604:
  605:        final Handler handler = new Handler() {
  606:
  607:                public void handleMessage(Message msg) {
  608:                        int total = msg.arg1;
  609:                        progressDialog.setProgress(total);
  610:
  611:                        // Check to dismiss
  612:                        if (total >= progressDialog.getMax()) {
  613:                                dismissDialog(PROGRESS_DIALOG);
  614:                                progressDialog.setProgress(0);
  615:                        }
  616:
  617:                }
  618:        };
  619:
  620:        /*****************************************************************
*********************************************************/
  621:
  622:        /**
  623:         * Write to log file
  624:         */
  625:        public void write_to_log() {
  626:                String header = "# Timestamp, X, Y, HEADING, MAC Address of AP
, RSS\n";
  627:                ArrayList<LogRecord> writeRecords;
  628:                LogRecord writeLR;
  629:                int N;
  630:
  631:                synchronized (samples) {
  632:
  633:                        try {
  634:
```

```
  635:                                    Boolean write_mode = prefs.getBoolean("write_m
ode", false);
  636:
  637:                                    N = Integer.parseInt(samples_num);
  638:
  639:                                    File root = new File(folder_path);
  640:
  641:                                    if (root.canWrite()) {
  642:
  643:                                            FileOutputStream fos = new FileOutputS
tream(new File(root,
  644:                                                    filename_rss), !write_
mode);
  645:
  646:                                            for (int i = 0; i < N; ++i) {
  647:                                                    fos.write(header.getBytes());
  648:                                                    writeRecords = samples.get(i);
  649:                                                    for (int j = 0; j < writeRecor
ds.size(); ++j) {
  650:                                                            writeLR = writeRecords
.get(j);
  651:                                                            fos.write(writeLR.toSt
ring().getBytes());
  652:                                                    }
  653:                                            }
  654:
  655:                                            if (!samples.isEmpty()) {
  656:
  657:                                                    Toast.makeText(
  658:                                                            this,
  659:                                                            ((N < samples.
size()) ? N : samples.size())
  660:
+ " Samples Recorded on\n"
  661:
+ Calendar.getInstance().getTime()
  662:
                .toString(), R.drawable.info)
  663:                                                            .show();
  664:
  665:                                                    samples.clear();
  666:
  667:                                                    subMenu1.getItem().setCheckabl
e(true);
  668:
  669:                                            }
  670:
  671:                                            fos.close();
  672:                                    }
  673:                            } catch (ClassCastException cce) {
  674:                                    Toast.makeText(this, "Error: " + cce.getMessag
e(),
  675:                                            R.drawable.error).show();
  676:                            } catch (NumberFormatException nfe) {
  677:                                    Toast.makeText(this, "Error: " + nfe.getMessag
e(),
  678:                                            R.drawable.error).show();
  679:                            } catch (FileNotFoundException fnfe) {
  680:                                    Toast.makeText(this, "Error: " + fnfe.getMessa
ge(),
  681:                                            R.drawable.error).show();
  682:                            } catch (IOException ioe) {
  683:                                    Toast.makeText(this, "Error: " + ioe.getMessag
e(),
  684:                                            R.drawable.error).show();
  685:                            }
  686:
  687:                    }
  688:
  689:            }
  690:
  691:            /*************************************************************************
*************************************************/
  692:
  693:            /**
  694:             *
  695:             * @author Julia Metochi
  696:             *
  697:             */
  698:            public class SimpleWifiReceiver extends WifiReceiver {
  699:                    @Override
  700:                    public void onReceive(Context context, Intent intent) {
  701:                            try {
  702:                                    if (intent == null || context == null
  703:                                                    || intent.getAction() == null)
  704:                                            return;
  705:
  706:                                    List<ScanResult> wifiList = wifi.getScanResult
s();
  707:                                    scanAPs.setText("AP : " + wifiList.size());
  708:                                    int m = 0;
  709:
  710:                                    if (subMenu1.getItem().isCheckable())
  711:                                            return;
  712:
  713:                                    ArrayList<LogRecord> Records = new ArrayList<L
ogRecord>();
  714:                                    Records.clear();
  715:
  716:                                    Date date = new Date();
  717:                                    long timestamp = date.getTime();
  718:
  719:                                    if ((curLon != 0 && curLat != 0) && !wifiList.
isEmpty()) {
  720:
  721:                                            for (int i = 0; i < wifiList.size(); i
++) {
  722:
  723:                                                    LogRecord lr = new LogRecord(t
imestamp, curLat, curLon,
  724:                                                            Heading.azimut
h, wifiList.get(i).BSSID,
  725:                                                            wifiList.get(i
).level);
  726:                                                    Records.add(lr);
  727:                                            }
  728:
  729:                                            synchronized (samples) {
  730:
  731:                                                    samples.add(0, Records);
  732:
  733:                                                    Message msg = handler.obtainMe
```

```
ssage();
  734:                                                  msg.arg1 = samples.size();
  735:                                                  handler.sendMessage(msg);
  736:
  737:                                                  if (samples.size() >= Integer.
parseInt(samples_num)) {
  738:                                                          write_to_log();
  739:                                                  }
  740:                                          }
  741:                                  }
  742:                          } catch (RuntimeException e) {
  743:                                  return;
  744:                          }
  745:                  }
  746:          }
  747:          /*****************************************************************
*************************************************/
  748: }
```

```java
package dmsl.crowdspaceSidemenu;

import java.io.BufferedReader;
import java.io.File;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.util.ArrayList;
import java.util.List;
import java.util.Observable;
import java.util.Observer;
import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.apache.http.protocol.HTTP;
import org.json.JSONArray;
import org.json.JSONException;

import com.actionbarsherlock.app.SherlockActivity;
import com.actionbarsherlock.view.Menu;
import com.actionbarsherlock.view.MenuItem;
import com.actionbarsherlock.view.SubMenu;
import com.cy.wifi.algorithms.Algorithms;
import com.cy.wifi.algorithms.LogRecord;
import com.cy.wifi.radiomap.RadioMap;

import com.google.gson.Gson;
import com.google.gson.JsonArray;
import com.google.zxing.client.android.CaptureActivity;

import dmsl.ItemScan.GoogleShoppingAPIHelperClass;
import dmsl.ItemScan.ObjectToSend;
import dmsl.ItemScan.ProductObject;
import dmsl.crowdspace.airplace.BooleanObservable;
import dmsl.crowdspace.airplace.DownloadingSettings;
import dmsl.crowdspace.airplace.FloorService;
import dmsl.crowdspace.airplace.Heading;
import dmsl.crowdspace.airplace.Prefs;
import dmsl.crowdspace.airplace.SimpleWifiManager;
import dmsl.crowdspace.airplace.WifiReceiver;

import android.app.Activity;
import android.app.ActivityManager;
import android.app.ActivityManager.RunningServiceInfo;
import android.app.AlertDialog;
import android.app.Dialog;
import android.app.ProgressDialog;
import android.content.BroadcastReceiver;
import android.content.ComponentName;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.IntentFilter;
import android.content.SharedPreferences;
import android.content.SharedPreferences.OnSharedPreferenceChangeListener;
import android.graphics.Paint.Cap;
import android.graphics.SumPathEffect;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.net.NetworkInfo.State;
import android.net.wifi.ScanResult;
import android.net.wifi.WifiManager;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.preference.PreferenceManager;
import android.util.Log;
import android.view.Gravity;
import android.view.KeyEvent;
import android.view.MenuInflater;
import android.view.View;
import android.view.ViewGroup;
import android.view.Window;
import android.view.View.OnClickListener;
import android.view.WindowManager.LayoutParams;
import android.webkit.ConsoleMessage;
import android.webkit.WebChromeClient;
import android.webkit.WebSettings;
import android.webkit.WebView;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;
import android.widget.ToggleButton;

/**
 *
 * @author Julia Metochi
 *
 */

public class MainActivity extends SherlockActivity implements
                OnSharedPreferenceChangeListener {

        public static WebView mWebview;
        public static int floor;
        private Dialog dialog = null;
        // ProgressDialog
        private ProgressDialog progressDialog;
        private static final int PROGRESS_DIALOG = 0;

        // Number of samples
        private String samples_num = null;

        // Coordinates(lon,lat)
        private static double curLon = 0;
        private static double curLat = 0;

        // Lon & Lat of scanned item
        public static double itemLon = 0, itemLat = 0;

        // WiFi manager
        private SimpleWifiManager wifi;
        // WiFi Receiver
```

```
123:         private WifiReceiver receiverWifi;
124:         // TextView showing the current scan results
125:         private TextView scanAPs;
126:         public static TextView curFloor;
127:
128:         // TextView trackingInfo
129:         private static TextView info;
130:         public Heading h;
131:
132:         // Preferences name for indoor and outdoor
133:         public static final String SHARED_PREFS_INDOOR = "Indoor_Preferences";
134:         // By default tracker selected
135:         public static int flag = -1;
136:
137:         public static Coordinates LonLat = new Coordinates();
138:
139:         // Flag to show if there is an ongoing progress
140:         private Boolean inProgress;
141:
142:         // Read RadioMap
143:         private RadioMap RM;
144:
145:         // Read Algorithm
146:         private Algorithms algo;
147:
148:         private final BooleanObservable trackMe = new BooleanObservable();
149:
150:         // The latest scan list of APs
151:         public static ArrayList<LogRecord> latestScanList;
152:
153:         static boolean bFind = false;
154:
155:         // Algorithm Selection
156:         private String algorithmSelection;
157:
158:         // Filename of radiomap to use for positioning
159:         private String filename_radiomap;
160:
161:         private JSONArray itemsPerFloor;// = new JSONArray();
162:         static boolean bImg = false;
163:         static boolean internetCon = true;
164:
165:         static final Handler myHandler = new Handler() {
166:
167:                 @Override
168:                 public void handleMessage(Message msg) {
169:                         // TODO Auto-generated method stub
170:                         super.handleMessage(msg);
171:                         LonLat = (Coordinates) msg.obj;
172:                         curLon = LonLat.getLon();
173:                         curLat = LonLat.getLat();
174:                         itemLat = curLat;
175:                         itemLon = curLon;
176:                         updateInfoLabel();
177:                 }
178:
179:         };
180:
181:         // Current floor
182:         public static int currentFloor = -100;
183:         public static String newId;
184:
185:         int currentFloorForFindMe = -100;
186:
187:         AccelerationServiceReceiver accelerationReceiver = null;
188:         BroadcastReceiver checkForInternet = null;
189:         // Flag for Floor mode on or Floor mode off
190:         public static Boolean isFloorMode = true;
191:
192:         // Flag for offline or online mode
193:         private boolean isOffline;
194:
195:         private SharedPreferences Preferences;
196:         private Boolean preFM;
197:
198:         SubMenu subMenu2;
199:
200:         /*************************************************************************
*************************************************/
201:
202:         private boolean isNetworkAvailable() {
203:                 ConnectivityManager connectivityManager = (ConnectivityManager
) getSystemService(Context.CONNECTIVITY_SERVICE);
204:                 NetworkInfo activeNetworkInfo = connectivityManager
205:                                 .getActiveNetworkInfo();
206:                 return activeNetworkInfo != null
207:                                 && activeNetworkInfo.isConnectedOrConnecting()
;
208:         }
209:
210:         /*************************************************************************
*************************************************/
211:
212:         /** Called when the activity is first created. */
213:         @Override
214:         public void onCreate(Bundle savedInstanceState) {
215:
216:                 super.onCreate(savedInstanceState);
217:                 setContentView(R.layout.main);
218:
219:                 h = new Heading(this);
220:                 mWebview = (WebView) findViewById(R.id.map);
221:                 info = (TextView) findViewById(R.id.trackingInfoLabel);
222:                 final MyJavaScriptInterface myJavaScriptInterface = new MyJava
ScriptInterface(
223:                                 this);
224:                 mWebview.addJavascriptInterface(myJavaScriptInterface,
225:                                 "myAndroidFunctions");
226:                 mWebview.getSettings().setJavaScriptEnabled(true);
227:
228:                 mWebview.loadUrl("http://juliavm.in.cs.ucy.ac.cy/uniBuilding.h
tml");
229:                 curFloor = (TextView) findViewById(R.id.tFloor);
230:                 updateInfoLabel();
231:
232:                 mWebview.setWebChromeClient(new WebChromeClient() {
233:                         public boolean onConsoleMessage(ConsoleMessage cm) {
234:                                 Log.d("MY_SCANNER ",
235:                                                 cm.message() + " -- From line
" + cm.lineNumber()
236:                                                                 + " of " + cm.
sourceId());
```

```
  237:                                    return true;
  238:                            }
  239:                    });
  240:
  241:                    // Create the Radio map
  242:                    RM = new RadioMap();
  243:
  244:                    // Create new Algorithm Object
  245:                    algo = new Algorithms();
  246:
  247:                    wifi = new SimpleWifiManager(getApplicationContext());
  248:
  249:                    // Create new receiver to get broadcasts
  250:                    receiverWifi = new SimpleWifiReceiver();
  251:
  252:                    scanAPs = (TextView) findViewById(R.id.detectedAPs);
  253:
  254:                    inProgress = false;
  255:
  256:                    latestScanList = new ArrayList<LogRecord>();
  257:
  258:                    // Configure preferences
  259:                    Preferences = PreferenceManager.getDefaultSharedPreferences(th
is);
  260:
  261:                    PreferenceManager.setDefaultValues(this, SHARED_PREFS_INDOOR,
  262:                                    MODE_PRIVATE, R.xml.preferences, true);
  263:                    Preferences = MainActivity.this.getSharedPreferences(
  264:                                    SHARED_PREFS_INDOOR, MODE_PRIVATE);
  265:
  266:                    currentFloor = -100;
  267:                    Preferences.registerOnSharedPreferenceChangeListener(this);
  268:                    onSharedPreferenceChanged(Preferences, "modeIn");
  269:                    isFloorMode = Preferences.getBoolean("modeFloor", false);
  270:                    SharedPreferences.Editor editor = Preferences.edit();
  271:                    editor.putBoolean("modeFloor", true);
  272:                    editor.commit();
  273:                    preFM = true;
  274:
  275:                    wifi.startScan(receiverWifi, "2000");
  276:
  277:            }
  278:
  279:            /*************************************************************************
*************************************************/
  280:
  281:            // show items of chosen floor
  282:            public void showItemsByFloor(int floor) {
  283:
  284:                    String result = null;
  285:                    String sFloor = String.valueOf(floor);
  286:                    InputStream inputStream = null;
  287:                    HttpClient httpclient = new DefaultHttpClient();
  288:                    HttpPost httppost = new HttpPost(
  289:                                    "http://juliavm.in.cs.ucy.ac.cy/itemsByFloor.p
hp");
  290:                    List<NameValuePair> nvps = new ArrayList<NameValuePair>();
  291:                    nvps.add(new BasicNameValuePair("byFloor", sFloor));
  292:                    try {
  293:                            httppost.setEntity(new UrlEncodedFormEntity(nvps, HTTP
.UTF_8));
  294:                    } catch (UnsupportedEncodingException e) {
  295:
  296:                            e.printStackTrace();
  297:                    }
  298:
  299:                    try {
  300:                            HttpResponse response = httpclient.execute(httppost);
  301:                            // Save response in InputStream
  302:                            HttpEntity entity = response.getEntity();
  303:                            inputStream = entity.getContent();
  304:
  305:                            // Convert response
  306:                            BufferedReader reader = new BufferedReader(new InputSt
reamReader(
  307:                                            inputStream, "iso-8859-1"), 8);
  308:
  309:                            StringBuilder stringBuilder = new StringBuilder();
  310:                            String line = null;
  311:
  312:                            while ((line = reader.readLine()) != null) {
  313:                                    stringBuilder.append(line + "\n");
  314:                            }
  315:                            // Close Stream
  316:                            inputStream.close();
  317:
  318:                            result = stringBuilder.toString();
  319:                            result = result.replaceAll("(\\r|\\n)", "");
  320:
  321:                    } catch (ClientProtocolException e) {
  322:
  323:                            e.printStackTrace();
  324:                    } catch (IOException e) {
  325:                            // e.printStackTrace();
  326:                            Toast.makeText(this, "No internet access", Toast.LENGT
H_LONG)
  327:                                            .show();
  328:                            AlertDialog.Builder aboutBuilder = new AlertDialog.Bui
lder(
  329:                                            MainActivity.this);
  330:                            aboutBuilder.setTitle("Closing Application");
  331:                            aboutBuilder
  332:                                            .setMessage("No internet access...CLOD
A will be closed!");
  333:
  334:                            aboutBuilder.setPositiveButton("Close",
  335:                                            new DialogInterface.OnClickListener()
{
  336:
  337:                                                    public void onClick(DialogInte
rface dialog, int which) {
  338:
  339:                                                            finish();
  340:                                                    }
  341:                                            });
  342:
  343:                            aboutBuilder.show();
  344:                            return;
  345:
  346:                    }
  347:
  348:                    // Get json from couchDB
```

```
349:                 try {
350:                         itemsPerFloor = new JSONArray(result);
351:                 } catch (JSONException e) {
352:                         e.printStackTrace();
353:                 }
354:                 if (itemsPerFloor != null) {
355:                         // Json with /
356:                         JSONArray jArray = new JSONArray();
357:                         for (int i = 0; i < itemsPerFloor.length(); i++) {
358:                                 try {
359:                                         jArray.put(itemsPerFloor.get(i).toStri
ng());
360:                                 } catch (JSONException e) {
361:                                         e.printStackTrace();
362:                                 }
363:
364:                         }
365:                         int k = 0;
366:                         // Send JSON items to js
367:                         mWebview.loadUrl("javascript:getItemsInJsonFromCouch("
368:                                         + jArray.toString() + ");");
369:                 }
370:
371:         }
372:
373:         /**************************************************************************
*************************************************/
374:
375:         // Update label with current longitude, latitude, heading
376:         public static void updateInfoLabel() {
377:                 StringBuilder sb = new StringBuilder();
378:                 int head = (int) Heading.azimuth;
379:                 sb.append("H: ");
380:                 sb.append(head);
381:                 sb.append("\nX: ");
382:                 sb.append(curLon);
383:                 sb.append("\nY: ");
384:                 sb.append(curLat);
385:                 sb.append("\nFloor: ");
386:                 sb.append(MainActivity.currentFloor);
387:                 if (info != null) {
388:                         info.setText(sb.toString());
389:                 }
390:
391:         }
392:
393:         /**************************************************************************
*************************************************/
394:
395:         @Override
396:         protected void onPause() {
397:
398:                 Intent intent = new Intent("dmsl.crowdspace.airplace.FloorServ
ice");
399:                 this.stopService(intent);
400:
401:                 if (accelerationReceiver != null) {
402:
403:                         unregisterReceiver(accelerationReceiver);
404:                         accelerationReceiver = null;
405:                 }
406:
407:                 super.onPause();
408:                 h.pause();
409:         }
410:
411:         /**************************************************************************
*************************************************/
412:
413:         @Override
414:         protected void onResume() {
415:
416:                 if (accelerationReceiver == null) {
417:
418:                         accelerationReceiver = new AccelerationServiceReceiver
();
419:                         IntentFilter movementFilter;
420:                         movementFilter = new IntentFilter(FloorService.MOVEMEN
T_UPDATE);
421:                         registerReceiver(accelerationReceiver, movementFilter)
;
422:                 }
423:
424:                 if (ItemMenu.EDIT_ITEM == 1) {
425:                         mWebview.loadUrl("javascript:clearMarkers();");
426:                         showItemsByFloor(MainActivity.currentFloor);
427:                         ItemMenu.EDIT_ITEM = 0;
428:                 }
429:                 super.onResume();
430:
431:                 // Return to tracker again
432:                 flag = -1;
433:                 // Reset longitude and latitude to zero
434:                 curLat = 0;
435:                 curLon = 0;
436:                 updateInfoLabel();
437:                 if (CaptureActivity.itemIsScanned == 1) {
438:                         CaptureActivity.itemIsScanned = 0;
439:                         sendJsonArrayOfItems();
440:
441:                 }
442:
443:                 h.resume();
444:
445:         }
446:
447:         /**************************************************************************
*************************************************/
448:
449:         // Send JSONArray of items to Javascript and show them in infowindow f
or
450:         // each item
451:         public void sendJsonArrayOfItems() {
452:
453:                 // if (CaptureActivity.arrayObj != null) {
454:                 if (CaptureActivity.ADD_ITEM) {
455:                         ItemMenu itMenu = new ItemMenu();
456:                         String itemInfo = itMenu.searchByDocId(newId.replace("
\"", ""));
457:                         GoogleShoppingAPIHelperClass gHelp = new GoogleShoppin
gAPIHelperClass();
458:                         ProductObject prObj = gHelp.productFoundInCouch(itemIn
```

```
fo);
  459:                          Gson gson = new Gson();
  460:                          ObjectToSend obj = new ObjectToSend(prObj, 0);
  461:                          String json = gson.toJson(obj);
  462:
  463:                          JSONArray arrayObj = new JSONArray();
  464:                          arrayObj.put(json);
  465:                          mWebview.loadUrl("javascript:getItemsInJSON(" + arrayO
bj.toString()
  466:                                          + ");");
  467:                          CaptureActivity.ADD_ITEM = false;
  468:                  } else {
  469:                          CaptureActivity.arrayObjIdRev.toString();
  470:                          mWebview.loadUrl("javascript:getItemsInJSON("
  471:                                          + CaptureActivity.arrayObjIdRev.toStri
ng() + ");");
  472:                  }
  473:                  mWebview.loadUrl("javascript:clearMarkers();");
  474:                  showItemsByFloor(MainActivity.currentFloor);
  475:
  476:          }
  477:
  478:          /*************************************************************************
*************************************************/
  479:
  480:          @Override
  481:          protected void onRestart() {
  482:                  if (accelerationReceiver == null) {
  483:
  484:                          IntentFilter movementFilter;
  485:                          movementFilter = new IntentFilter(FloorService.MOVEMEN
T_UPDATE);
  486:                          accelerationReceiver = new AccelerationServiceReceiver
();
  487:                          registerReceiver(accelerationReceiver, movementFilter)
;
  488:                  }
  489:                  if (!isNetworkAvailable()) {
  490:                          AlertDialog.Builder dialog = new AlertDialog.Builder(t
his);
  491:                          dialog.setTitle("Wi-Fi is turned off/No internet conne
ction available");
  492:                          dialog.setMessage("CLOD needs an internet connection f
or all its functions to work properly.");
  493:                          dialog.setPositiveButton("Ok",
  494:                                          new DialogInterface.OnClickListener()
{
  495:
  496:                                                  @Override
  497:                                                  public void onClick(DialogInte
rface dialog, int which) {
  498:                                                          // TODO Auto-generated
 method stub
  499:                                                          Intent intent = new In
tent();
  500:                                                          intent.setComponent(ne
w ComponentName(
  501:                                                                          "com.a
ndroid.settings",
  502:                                                                          "com.a
ndroid.settings.WirelessSettings"));
  503:                                                                          startActivity(intent);
  504:                                                  }
  505:                                          });
  506:                          dialog.show();
  507:                  }
  508:
  509:                  if (CaptureActivity.ADD_ITEM) {
  510:
  511:                          CaptureActivity item = new CaptureActivity();
  512:                          CaptureActivity.editItem.setPointX(MainActivity.itemLo
n);
  513:                          CaptureActivity.editItem.setPointY(MainActivity.itemLa
t);
  514:                          CaptureActivity ca = new CaptureActivity();
  515:                          item.sendProductToCouch(CaptureActivity.editItem);
  516:                          String test = ca.searchCouchDb(CaptureActivity.editIte
m
  517:                                          .getBarcode());
  518:                          String[] ja = test.split(",");
  519:                          ArrayList<String> jas = new ArrayList<String>();
  520:                          for (int i = 0; i < ja.length; i++) {
  521:                                  jas.add(ja[i]);
  522:                          }
  523:                          String[] newItemId = jas.get(jas.size() - 2).split(":"
);
  524:                          jas.clear();
  525:                          for (int i = 0; i < newItemId.length; i++) {
  526:                                  jas.add(newItemId[i]);
  527:                          }
  528:                          newId = jas.get(1);
  529:                          System.out.println(test);
  530:
  531:                  }
  532:                  if (ItemMenu.DELETE_ITEM == 1 || ItemMenu.EDIT_ITEM == 1) {
  533:                          mWebview.loadUrl("javascript:clearMarkers();");
  534:                          showItemsByFloor(MainActivity.currentFloor);
  535:                          ItemMenu.DELETE_ITEM = 0;
  536:                          ItemMenu.EDIT_ITEM = 0;
  537:                  }
  538:
  539:                  super.onRestart();
  540:                  curLat = 0;
  541:                  curLon = 0;
  542:
  543:                  updateInfoLabel();
  544:
  545:          }
  546:
  547:          /*************************************************************************
*************************************************/
  548:
  549:          @Override
  550:          protected void onStart() {
  551:                  super.onStart();
  552:
  553:          }
  554:
  555:          /*************************************************************************
*************************************************/
  556:
  557:          @Override
```

```
  558:         protected void onDestroy() {
  559:                 super.onDestroy();
  560:         }
  561:
  562:         /*************************************************************************
**************************************************/
  563:
  564:         @Override
  565:         protected void onStop() {
  566:
  567:                 super.onStop();
  568:
  569:         }
  570:
  571:         /*************************************************************************
**************************************************/
  572:
  573:         @Override
  574:         public boolean onCreateOptionsMenu(Menu menu) {
  575:
  576:                 getSupportActionBar().setHomeButtonEnabled(false);
  577:                 getSupportActionBar().setDisplayShowTitleEnabled(false);
  578:
  579:                 /************************************ FIND ME ************
************************************************************/
  580:                 SubMenu subMenu1 = menu.addSubMenu("Find Me");
  581:                 subMenu2 = menu.addSubMenu("Track Me");
  582:                 MenuItem subMenu1ItemFindMe = subMenu1.getItem();
  583:                 subMenu1ItemFindMe.setIcon(android.R.drawable.ic_menu_myplaces
);
  584:                 subMenu1ItemFindMe.setShowAsAction(MenuItem.SHOW_AS_ACTION_ALW
AYS
  585:                                 | MenuItem.SHOW_AS_ACTION_WITH_TEXT);
  586:                 subMenu1ItemFindMe
  587:                                 .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  588:
  589:                                         @Override
  590:                                         public boolean onMenuItemClick(MenuIte
m item) {
  591:                                                 // bTrackMe.setChecked(false);
  592:                                                 trackMe.setBoolean(false);
  593:                                                 trackMe.notifyObservers();
  594:                                                 bFind = true;
  595:                                                 subMenu2.getItem().setChecked(
false);
  596:                                                 subMenu2.getItem().setIcon(
  597:                                                                 android.R.draw
able.ic_menu_mylocation);
  598:                                                 FindMe_Method();
  599:                                                 return false;
  600:                                         }
  601:                                 });
  602:                 /************************************ END OF FIND ME ****
************************************************************/
  603:                 /************************************ TRACK ME ***********
************************************************************/
  604:
  605:                 MenuItem subMenu2ItemTrackMe = subMenu2.getItem();
  606:
  607:
  608:
  609:                 subMenu2ItemTrackMe.setIcon(android.R.drawable.ic_menu_mylocat
ion);
  610:
  611:                 subMenu2ItemTrackMe.setShowAsAction(MenuItem.SHOW_AS_ACTION_AL
WAYS
  612:                                 | MenuItem.SHOW_AS_ACTION_WITH_TEXT);
  613:
  614:                 subMenu2ItemTrackMe
  615:                                 .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  616:
  617:                                         @Override
  618:                                         public boolean onMenuItemClick(MenuIte
m item) {
  619:
  620:                                                 if (subMenu2.getItem().isCheck
ed()) {
  621:                                                         subMenu2.getItem().set
Icon(
  622:                                                                         androi
d.R.drawable.ic_menu_mylocation);
  623:                                                         trackMe.setBoolean(fal
se);
  624:                                                         trackMe.notifyObserver
s();
  625:                                                         subMenu2.getItem().set
Checked(false);
  626:
  627:                                                 } else {
  628:                                                         subMenu2.getItem().set
Icon(R.drawable.track_me_on);
  629:                                                         trackMe.setBoolean(tru
e);
  630:                                                         trackMe.notifyObserver
s();
  631:                                                         subMenu2.getItem().set
Checked(true);
  632:                                                 }
  633:
  634:                                                 return false;
  635:                                         }
  636:                                 });
  637:
  638:                 /************************************ END OF TRACK ME ***
************************************************************/
  639:
  640:                 /************************************ SCAN ITEM **********
************************************************************/
  641:                 SubMenu subMenu3 = menu.addSubMenu("Scan Item");
  642:                 MenuItem subMenu3ItemScanItem = subMenu3.getItem();
  643:                 subMenu3ItemScanItem.setIcon(android.R.drawable.ic_menu_camera
);
  644:                 subMenu3ItemScanItem.setShowAsAction(MenuItem.SHOW_AS_ACTION_A
LWAYS
  645:                                 | MenuItem.SHOW_AS_ACTION_WITH_TEXT);
  646:                 subMenu3ItemScanItem
  647:                                 .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  648:
  649:                                         @Override
```

```
  650:                                                public boolean onMenuItemClick(MenuIte
m item) {
  651:                                                        if (!subMenu2.getItem().isChec
ked()) {
  652:                                                                Intent scanItem = new
Intent(MainActivity.this,
  653:                                                                                  Captur
eActivity.class);
  654:                                                                startActivity(scanItem
);
  655:                                                        }
  656:
  657:                                                        else {
  658:                                                                Toast.makeText(MainAct
ivity.this,
  659:                                                                               "Turn
Track Me off and try again",
  660:                                                                               Toast.
LENGTH_LONG).show();
  661:                                                        }
  662:                                                        return false;
  663:                                                }
  664:                                        });
  665:
  666:                    /************************************** END OF SCAN ITEM **
*****************************************************************/
  667:
  668:                    /************************************** SHOW/HIDE ITEMS &
FLOORPLANS ***********************************************************/
  669:                    final SubMenu subMenu4 = menu.addSubMenu("Show/Hide Overlays")
;
  670:
  671:                    subMenu4.add("Hide Items");
  672:                    subMenu4.add("Show Items");
  673:                    subMenu4.add("Hide FloorPlan");
  674:                    subMenu4.add("Show FloorPlan");
  675:                    MenuItem subMenu4ItemHideOverlays = subMenu4.getItem();
  676:                    subMenu4ItemHideOverlays.setIcon(android.R.drawable.ic_menu_ma
pmode);
  677:                    subMenu4ItemHideOverlays.setShowAsAction(MenuItem.SHOW_AS_ACTI
ON_ALWAYS
  678:                                    | MenuItem.SHOW_AS_ACTION_WITH_TEXT);
  679:
  680:                    // Hide items
  681:                    MenuItem subMenu4HideItems = subMenu4.getItem(0);
  682:
  683:                    subMenu4HideItems
  684:                                    .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  685:
  686:                                            @Override
  687:                                            public boolean onMenuItemClick(MenuIte
m item) {
  688:
  689:                                                    mWebview.loadUrl("javascript:c
learMarkers();");
  690:                                                    return false;
  691:                                            }
  692:                                    });
  693:
  694:                    subMenu4HideItems
  695:                                    .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  696:
  697:                                            @Override
  698:                                            public boolean onMenuItemClick(MenuIte
m item) {
  699:
  700:                                                    mWebview.loadUrl("javascript:c
learMarkers();");
  701:
  702:                                                    return false;
  703:                                            }
  704:                                    });
  705:
  706:                    // Show items
  707:                    MenuItem subMenu4ShowItems = subMenu4.getItem(1);
  708:                    subMenu4ShowItems
  709:                                    .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  710:
  711:                                            @Override
  712:                                            public boolean onMenuItemClick(MenuIte
m item) {
  713:
  714:                                                    showItemsByFloor(MainActivity.
currentFloor);
  715:
  716:                                                    return false;
  717:                                            }
  718:                                    });
  719:
  720:                    // Hide FloorPlan
  721:                    MenuItem subMenu4HideFloorPlan = subMenu4.getItem(2);
  722:
  723:                    subMenu4HideFloorPlan
  724:                                    .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  725:
  726:                                            @Override
  727:                                            public boolean onMenuItemClick(MenuIte
m item) {
  728:
  729:                                                    mWebview.loadUrl("javascript:h
ideShowOverlay();");
  730:                                                    bImg = true;
  731:                                                    return false;
  732:                                            }
  733:                                    });
  734:
  735:                    // Show FloorPlan
  736:                    MenuItem subMenu4ShowFloorPlan = subMenu4.getItem(3);
  737:
  738:                    subMenu4ShowFloorPlan
  739:                                    .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  740:
  741:                                            @Override
  742:                                            public boolean onMenuItemClick(MenuIte
m item) {
  743:                                                    if (bImg == true) {
  744:                                                            mWebview.loadUrl("java
```

```
script:viewUniMap("
  745:                                                                    + Main
Activity.currentFloor + ");");
  746:                                                    bImg = false;
  747:                                            }
  748:
  749:                                            return false;
  750:                                    }
  751:                            });
  752:
  753:                /************************************** END OF SHOW/HIDE *
*****************************************************************/
  754:
  755:                /************************************** MAIN MENU *********
*****************************************************************/
  756:                SubMenu subMenu5 = menu.addSubMenu("Main Menu");
  757:                subMenu5.add("Logger");
  758:                subMenu5.add("Preferences");
  759:                subMenu5.add("Help");
  760:                subMenu5.add("About");
  761:                subMenu5.add("Exit");
  762:
  763:                subMenu2.getItem().setChecked(false);
  764:                subMenu2.getItem().setIcon(android.R.drawable.ic_menu_mylocati
on);
  765:
  766:                MenuItem subMenu5ItemMainMenu = subMenu5.getItem();
  767:                subMenu5ItemMainMenu
  768:                                .setIcon(R.drawable.abs__ic_menu_moreoverflow_
normal_holo_dark);
  769:                subMenu5ItemMainMenu.setShowAsAction(MenuItem.SHOW_AS_ACTION_A
LWAYS
  770:                                | MenuItem.SHOW_AS_ACTION_WITH_TEXT);
  771:
  772:                // Logger
  773:                MenuItem subMenu5Logger = subMenu5.getItem(0);
  774:                subMenu5Logger.setIcon(R.drawable.location2);
  775:
  776:                subMenu5Logger
  777:                                .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  778:                                        @Override
  779:                                        public boolean onMenuItemClick(MenuIte
m item) {
  780:
  781:                                                flag = 1;
  782:                                                Intent logger = new Intent(Mai
nActivity.this,
  783:                                                                MainActivityLo
gger.class);
  784:                                                startActivity(logger);
  785:
  786:                                                return false;
  787:                                        }
  788:                                });
  789:
  790:                // Preferences
  791:                MenuItem subMenu5Prefs = subMenu5.getItem(1);
  792:                subMenu5Prefs.setIcon(android.R.drawable.ic_menu_preferences);
  793:
  794:                subMenu5Prefs
  795:                                .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  796:                                        @Override
  797:                                        public boolean onMenuItemClick(MenuIte
m item) {
  798:
  799:                                                Intent prefs = new Intent(Main
Activity.this,
  800:                                                                Prefs.class);
  801:                                                startActivity(prefs);
  802:
  803:                                                return false;
  804:                                        }
  805:                                });
  806:
  807:                // Help
  808:                MenuItem subMenu5Help = subMenu5.getItem(2);
  809:                subMenu5Help.setIcon(android.R.drawable.ic_menu_help);
  810:
  811:                subMenu5Help
  812:                                .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  813:                                        @Override
  814:                                        public boolean onMenuItemClick(MenuIte
m item) {
  815:                                                dialog = new Dialog(MainActivi
ty.this);
  816:                                                dialog.setContentView(R.layout
.help);
  817:                                                Window w = dialog.getWindow();
  818:
  819:                                                w.setGravity(Gravity.FILL_HORI
ZONTAL
  820:                                                                | Gravity.FILL
_VERTICAL);
  821:                                                w.setLayout(LayoutParams.FILL_
PARENT,
  822:                                                                ViewGroup.Layo
utParams.FILL_PARENT);
  823:                                                dialog.setTitle("CLODA: Help")
;
  824:                                                LinearLayout v = (LinearLayout
) dialog
  825:                                                                .findViewById(
R.id.helpwindow);
  826:                                                if (v != null) {
  827:                                                        v.setClickable(true);
  828:                                                        v.setOnClickListener(n
ew OnClickListener() {
  829:                                                                public void on
Click(View v) {
  830:                                                                        dialog
.dismiss();
  831:                                                                }
  832:                                                        });
  833:                                                }
  834:                                                dialog.show();
  835:
  836:                                                return false;
  837:                                        }
  838:
```

```
                                });

                // About
                MenuItem subMenu5About = subMenu5.getItem(3);
                subMenu5About.setIcon(android.R.drawable.ic_menu_info_details)
;

                subMenu5About
                                .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
                                        @Override
                                        public boolean onMenuItemClick(MenuIte
m item) {
                                                dialog = new Dialog(MainActivi
ty.this);
                                                dialog.setContentView(R.layout
.about);
                                                Window w = dialog.getWindow();
                                                w.setGravity(Gravity.FILL_HORI
ZONTAL
                                                                | Gravity.FILL
_VERTICAL);
                                                w.setLayout(ViewGroup.LayoutPa
rams.FILL_PARENT,
                                                                ViewGroup.Layo
utParams.FILL_PARENT);
                                                dialog.setTitle("CLODA: About"
);
                                                LinearLayout v = (LinearLayout
) dialog
                                                                .findViewById(
R.id.aboutwindow);
                                                if (v != null) {
                                                        v.setClickable(true);
                                                        v.setOnClickListener(n
ew OnClickListener() {
                                                                public void on
Click(View v) {
                                                                        dialog
.dismiss();
                                                                }
                                                        });
                                                }
                                                dialog.show();
                                                return false;
                                        }
                                });

                // Exit
                MenuItem subMenu5Exit = subMenu5.getItem(4);
                subMenu5Exit.setIcon(android.R.drawable.ic_menu_close_clear_ca
ncel);

                subMenu5Exit
                                .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
                                        @Override
                                        public boolean onMenuItemClick(MenuIte
m item) {
                                                finish();

                                                return false;
                                        }
                                });
                        }
                });

                /************************************** END OF MAIN MENU **
************************************************************/

                /************************************** FLOORS ***********
************************************************************/
                SubMenu subMenu6 = menu.addSubMenu("Floors");
                subMenu6.add("Floor -1");
                subMenu6.add("Floor  0");
                subMenu6.add("Floor  1");
                subMenu6.add("Floor  2");

                // Floor -1
                MenuItem subMenu6ItemFloorPlinEna = subMenu6.getItem(0);
                subMenu6ItemFloorPlinEna
                                .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {

                                        @Override
                                        public boolean onMenuItemClick(MenuIte
m item) {
                                                floor = -1;
                                                Intent intent = new Intent(
                                                                "dmsl.crowdspa
ce.airplace.FloorService");
                                                stopService(intent);
                                                SharedPreferences.Editor edito
r = Preferences.edit();
                                                editor.putBoolean("modeFloor",
 false);
                                                editor.commit();
                                                currentFloor = floor;

                                                itemsPerFloor = null;
                                                mWebview.loadUrl("javascript:h
ideShowOverlay();");

                                                mWebview.loadUrl("javascript:c
learMarkers();");

                                                mWebview.loadUrl("javascript:v
iewUniMap(" + floor
                                                                + ");");
                                                showItemsByFloor(floor);
                                                return false;
                                        }
                                });

                // Floor 0
                MenuItem subMenu6ItemFloorZero = subMenu6.getItem(1);
                subMenu6ItemFloorZero
                                .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {

                                        @Override
                                        public boolean onMenuItemClick(MenuIte
m item) {
                                                floor = 0;
```

```
  931:                                                    Intent intent = new Intent(
  932:                                                            "dmsl.crowdspa
ce.airplace.FloorService");
  933:                                                    stopService(intent);
  934:                                                    SharedPreferences.Editor edito
r = Preferences.edit();
  935:                                                    editor.putBoolean("modeFloor",
 false);
  936:                                                    editor.commit();
  937:                                                    currentFloor = floor;
  938:
  939:                                                    itemsPerFloor = null;
  940:                                                    mWebview.loadUrl("javascript:h
ideShowOverlay();");
  941:
  942:                                                    mWebview.loadUrl("javascript:c
learMarkers();");
  943:
  944:                                                    mWebview.loadUrl("javascript:v
iewUniMap(" + floor
  945:                                                            + ");");
  946:                                                    showItemsByFloor(floor);
  947:                                                    return false;
  948:                                            }
  949:                                    });
  950:
  951:                    // Floor 1
  952:                    MenuItem subMenu6ItemFloorOne = subMenu6.getItem(2);
  953:                    subMenu6ItemFloorOne
  954:                                    .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  955:
  956:                                            @Override
  957:                                            public boolean onMenuItemClick(MenuIte
m item) {
  958:                                                    floor = 1;
  959:                                                    Intent intent = new Intent(
  960:                                                            "dmsl.crowdspa
ce.airplace.FloorService");
  961:                                                    stopService(intent);
  962:                                                    SharedPreferences.Editor edito
r = Preferences.edit();
  963:                                                    editor.putBoolean("modeFloor",
 false);
  964:                                                    editor.commit();
  965:                                                    currentFloor = floor;
  966:
  967:                                                    itemsPerFloor = null;
  968:                                                    mWebview.loadUrl("javascript:h
ideShowOverlay();");
  969:
  970:                                                    mWebview.loadUrl("javascript:c
learMarkers();");
  971:
  972:                                                    mWebview.loadUrl("javascript:v
iewUniMap(" + floor
  973:                                                            + ");");
  974:                                                    showItemsByFloor(floor);
  975:                                                    return false;
  976:                                            }
  977:                                    });
  978:
  979:                    // Floor 2
  980:                    MenuItem subMenu6ItemFloorTwo = subMenu6.getItem(3);
  981:                    subMenu6ItemFloorTwo
  982:                                    .setOnMenuItemClickListener(new MenuItem.OnMen
uItemClickListener() {
  983:
  984:                                            @Override
  985:                                            public boolean onMenuItemClick(MenuIte
m item) {
  986:                                                    floor = 2;
  987:                                                    Intent intent = new Intent(
  988:                                                            "dmsl.crowdspa
ce.airplace.FloorService");
  989:                                                    stopService(intent);
  990:                                                    SharedPreferences.Editor edito
r = Preferences.edit();
  991:                                                    editor.putBoolean("modeFloor",
 false);
  992:                                                    editor.commit();
  993:                                                    currentFloor = floor;
  994:
  995:                                                    itemsPerFloor = null;
  996:                                                    mWebview.loadUrl("javascript:h
ideShowOverlay();");
  997:
  998:                                                    mWebview.loadUrl("javascript:c
learMarkers();");
  999:
 1000:                                                    mWebview.loadUrl("javascript:v
iewUniMap(" + floor
 1001:                                                            + ");");
 1002:                                                    showItemsByFloor(floor);
 1003:                                                    return false;
 1004:                                            }
 1005:                                    });
 1006:
 1007:                    return super.onCreateOptionsMenu(menu);
 1008:            }
 1009:
 1010:            /*****************************************************************
*************************************************/
 1011:
 1012:            @Override
 1013:            public boolean onKeyDown(int keyCode, KeyEvent event) {
 1014:
 1015:                    switch (keyCode) {
 1016:                    case KeyEvent.KEYCODE_BACK:
 1017:
 1018:                            setResult(RESULT_CANCELED);
 1019:                            AlertDialog.Builder aboutBuilder = new AlertDialog.Bui
lder(this);
 1020:                            aboutBuilder.setTitle("Closing Application");
 1021:                            aboutBuilder.setMessage("Are you sure you want to exit
 CLODA?");
 1022:                            aboutBuilder.setPositiveButton("Yes",
 1023:                                            new DialogInterface.OnClickListener()
{
 1024:
 1025:                                            public void onClick(DialogInte
rface dialog, int which) {
```

```
1026:                                                                   finish();
1027:                                                           }
1028:                                                   });
1029:                                   aboutBuilder.setNegativeButton("Cancel", null);
1030:                                   aboutBuilder.show();
1031:                                   return true;
1032:                           case KeyEvent.KEYCODE_FOCUS:
1033:                           case KeyEvent.KEYCODE_CAMERA:
1034:                           case KeyEvent.KEYCODE_SEARCH:
1035:                                   // Handle these events so they don't launch the Camera
 app
1036:                                   return true;
1037:                           }
1038:                           return super.onKeyDown(keyCode, event);
1039:                   }
1040:
1041:                   /************************************************************************
**************************************************/
1042:
1043:                   /**
1044:                    * Create the dialog
1045:                    *
1046:                    * @param id
1047:                    *            the id of dialog to create
1048:                    * */
1049:                   @Override
1050:                   protected Dialog onCreateDialog(int id) {
1051:                           switch (id) {
1052:                           case PROGRESS_DIALOG:
1053:                                   progressDialog = new ProgressDialog(MainActivity.this)
;
1054:                                   progressDialog.setProgressStyle(ProgressDialog.STYLE_H
ORIZONTAL);
1055:                                   progressDialog.setMessage("Scanning in progress...");
1056:                                   progressDialog.setCancelable(false);
1057:                                   return progressDialog;
1058:                           default:
1059:                                   return null;
1060:                           }
1061:                   }
1062:
1063:                   /************************************************************************
**************************************************/
1064:
1065:                   /**
1066:                    * Prepare the dialog. Sets progress to 0 and max to samples number
1067:                    *
1068:                    * @param id
1069:                    *            the id of dialog to prepare
1070:                    * @param dialog
1071:                    *
1072:                    * */
1073:                   @Override
1074:                   protected void onPrepareDialog(int id, Dialog dialog) {
1075:                           switch (id) {
1076:                           case PROGRESS_DIALOG:
1077:                                   progressDialog.setProgress(0);
1078:                                   progressDialog.setMax(Integer.parseInt(samples_num));
1079:                                   break;
1080:                           }
1081:                   }
1082:
1083:                   /************************************************************************
**************************************************/
1084:
1085:                   /**
1086:                    * Handler that receives messages from the thread and update the progr
ess
1087:                    * dialog
1088:                    */
1089:                   final Handler handler = new Handler() {
1090:
1091:                           public void handleMessage(Message msg) {
1092:                                   int total = msg.arg1;
1093:                                   progressDialog.setProgress(total);
1094:
1095:                                   // Check to dismiss
1096:                                   if (total >= progressDialog.getMax()) {
1097:                                           dismissDialog(PROGRESS_DIALOG);
1098:                                           progressDialog.setProgress(0);
1099:                                   }
1100:
1101:                           }
1102:                   };
1103:
1104:                   /************************************************************************
**************************************************/
1105:
1106:                   private void popup_msg(String msg, String title, int imageID) {
1107:
1108:                           AlertDialog.Builder alert_box = new AlertDialog.Builder(this);
1109:                           alert_box.setTitle(title);
1110:                           alert_box.setMessage(msg);
1111:                           alert_box.setIcon(imageID);
1112:
1113:                           alert_box.setNeutralButton("Hide",
1114:                                           new DialogInterface.OnClickListener() {
1115:                                                   public void onClick(DialogInterface di
alog, int which) {
1116:                                                           dialog.cancel();
1117:                                                   }
1118:                                           });
1119:
1120:                           AlertDialog alert = alert_box.create();
1121:                           alert.show();
1122:                   }
1123:
1124:                   /************************************************************************
**************************************************/
1125:
1126:                   // Starts the appropriate positioning algorithm
1127:                   private boolean FindMe_Method() {
1128:
1129:                           algorithmSelection = (String) Preferences.getString("Algorithm
s", "1")
1130:                                           .trim();
1131:                           filename_radiomap = (String) Preferences.getString("radiomap_f
ile", "")
1132:                                           .trim();
1133:
1134:                           long startTime = System.currentTimeMillis();
1135:
```

```
1136:
1137:                // Check that radiomap file is readable
1138:                if (filename_radiomap.equals("")) {
1139:
1140:                        popup_msg(
1141:                                        "Radiomap file not specified\nGo to Me
nu::Preferences::Radiomap Settings::Radiomap File",
1142:                                        "User Error", R.drawable.error);
1143:                        return false;
1144:
1145:                } else if ((!(new File(filename_radiomap).canRead()))) {
1146:
1147:                        popup_msg(
1148:                                        "Radiomap file is not readable\nGo to
Menu::Preferences::Radiomap Settings::Radiomap File",
1149:                                        "User Error", R.drawable.error);
1150:                        return false;
1151:                }
1152:
1153:                // Set in progress (true)
1154:                synchronized (inProgress) {
1155:                        if (inProgress == true)
1156:                                return false;
1157:                        inProgress = true;
1158:                }
1159:
1160:                // Error reading Radio Map
1161:                if (!RM.ConstructRadioMap(new File(filename_radiomap))) {
1162:
1163:                        popup_msg(
1164:                                        "Error while reading radio map.\nDownl
oad new Radio Map and try again",
1165:                                        "User Error", R.drawable.error);
1166:
1167:                        // Unset in progress (false)
1168:                        synchronized (inProgress) {
1169:                                inProgress = false;
1170:                        }
1171:
1172:                        return false;
1173:                }
1174:
1175:                if (latestScanList.isEmpty()) {
1176:
1177:                        popup_msg(
1178:                                        "No Access Point Received.\nWait for a
 scan first and try again.",
1179:                                        "Warning", R.drawable.warning);
1180:
1181:                        // Unset in progress (false)
1182:                        synchronized (inProgress) {
1183:                                inProgress = false;
1184:                        }
1185:
1186:                        return false;
1187:                }
1188:
1189:                if (!calculatePosition()) {
1190:
1191:                        popup_msg(
1192:                                        "Can't find location. Check that radio
map file refers to the same area.",
1193:                                                "Error", R.drawable.error);
1194:
1195:                        // Unset in progress (false)
1196:                        synchronized (inProgress) {
1197:                                inProgress = false;
1198:                        }
1199:
1200:                        return false;
1201:                }
1202:
1203:                // Unset in progress (false)
1204:                synchronized (inProgress) {
1205:                        inProgress = false;
1206:                }
1207:
1208:                long stopTime = System.currentTimeMillis();
1209:                long elapsedTime = stopTime - startTime;
1210:                System.err.println("Elapsed Time: " + elapsedTime);
1211:
1212:                return true;
1213:
1214:        }
1215:
1216:        /*************************************************************************
*************************************************/
1217:
1218:        // Show marker according to user movements on map
1219:        public void updateMap(double lon, double lat) {
1220:                mWebview.loadUrl("javascript:TrackMeMarker(" + lon + "," + lat
+ ");");
1221:
1222:        }
1223:
1224:        /*************************************************************************
*************************************************/
1225:
1226:        // Calculate the position of the user on map
1227:        private boolean calculatePosition() {
1228:                int degrees = 360;
1229:                int num_orientations = 4;
1230:                int range = degrees / num_orientations;
1231:                float deviation = range / 2;
1232:
1233:                float Hheading = Heading.azimuth;
1234:                int heading = (int) (((Math.round(Hheading + deviation) % degr
ees) / range) % num_orientations)
1235:                                        * range;
1236:                boolean result = algo.ProcessingAlgorithms(latestScanList, RM,
1237:                                        algo.readWeights(RM.getRadiomapMean_File(), he
ading), heading);
1238:
1239:                if (!result) {
1240:                        return false;
1241:                }
1242:
1243:                float x, y;
1244:
1245:                // lat
1246:                x = algo.getX();
1247:                // lon
```

```
 1248:                 y = algo.getY();
 1249:
 1250:                 curLat = x;
 1251:                 curLon = y;
 1252:                 updateInfoLabel();
 1253:                 updateMap(y, x);
 1254:
 1255:                 return true;
 1256:
 1257:         }
 1258:
 1259:         /*************************************************************************
*************************************************/
 1260:
 1261:         /**
 1262:          *
 1263:          * Class that takes the new floor if and only if the FloorService indi
cates
 1264:          * a new floor
 1265:          *
 1266:          */
 1267:
 1268:         public class AccelerationServiceReceiver extends BroadcastReceiver {
 1269:
 1270:                 // This method receives broadcast messages. Be sure to modify
 1271:                 // AndroidManifest.xml file in order to enable message receivi
ng
 1272:                 @Override
 1273:                 public void onReceive(Context context, Intent intent) {
 1274:                         // Take the new floor that must change
 1275:                         double floor = intent.getDoubleExtra(FloorService.MSG_
SEND, 0);
 1276:
 1277:                         // Assign the new floor
 1278:                         currentFloor = (int) floor;
 1279:                         // unregisterReceiver(accelerationReceiver);
 1280:
 1281:                         // Change the architecture map
 1282:                         onSharedPreferenceChanged(Preferences, "change_floor_i
mage_custom");
 1283:
 1284:                         // Change the radio-map, parameters file etc.
 1285:                         toastPrint("Floor changed to " + currentFloor, Toast.L
ENGTH_LONG);
 1286:
 1287:                 }
 1288:         }
 1289:
 1290:         /*************************************************************************
*************************************************/
 1291:
 1292:         /*
 1293:          * Method used to print pop up message to user
 1294:          */
 1295:         private void toastPrint(String textMSG, int duration) {
 1296:                 Toast.makeText(this, textMSG, duration).show();
 1297:         }
 1298:
 1299:         /*************************************************************************
*************************************************/
 1300:
 1301:         public void onSharedPreferenceChanged(SharedPreferences prefs, String
key) {
 1302:
 1303:                 if (key == null)
 1304:                         return;
 1305:
 1306:                 if (key.equals("modeIn")) {
 1307:                         isOffline = Preferences.getBoolean("modeIn", false);
 1308:
 1309:                         if (!isOffline) {
 1310:
 1311:                                 // Enables the WiFi if is in Online Mode
 1312:                                 wifi.startScan(receiverWifi, "2000");
 1313:                         }
 1314:
 1315:                 } else if (key.equals("change_floor_image_custom")) {
 1316:                         Toast.makeText(this, "Changing floorplan", Toast.LENGT
H_LONG)
 1317:                                         .show();
 1318:                         itemsPerFloor = null;
 1319:                         mWebview.loadUrl("javascript:hideShowOverlay();");
 1320:
 1321:                         mWebview.loadUrl("javascript:clearMarkers();");
 1322:
 1323:                         mWebview.loadUrl("javascript:viewUniMap(" + currentFlo
or + ");");
 1324:                         showItemsByFloor(currentFloor);
 1325:
 1326:                 } else if (key.equals("modeFloor")) {
 1327:                         isFloorMode = Preferences.getBoolean("modeFloor", fals
e);
 1328:
 1329:                         if (isFloorMode) {
 1330:                                 // Must take the currentFloor to inform the Fl
oorService
 1331:                                 // Start service if is not started
 1332:                                 System.out.println("floorMode: " + isFloorMode
 1333:                                                 + " currentFloor: " + currentF
loor + " floor: "
 1334:                                                 + currentFloorForFindMe);
 1335:
 1336:                                 /**
 1337:                                  * Check if we are in the same area of the cur
rent results We do
 1338:                                  * that to load all the necessary items (radio
-map, floorplan
 1339:                                  * etc) for default option. If we are in the v
alid area we print
 1340:                                  * a message that all the data were loaded suc
cessfully.
 1341:                                  */
 1342:
 1343:                                 Toast.makeText(this, "FLOOR MODE ON", Toast.LE
NGTH_LONG).show();
 1344:
 1345:                                 Intent intent = new Intent(
 1346:                                                 "dmsl.crowdspace.airplace.Floo
rService");
 1347:                                 this.startService(intent);
 1348:
 1349:                         } else if (!isFloorMode) {
```

```
 1350:                                  Toast.makeText(this, "FLOOR MODE OFF", Toast.L
ENGTH_LONG)
 1351:                                                  .show();
 1352:
 1353:                                  Intent intent = new Intent(
 1354:                                                  "dmsl.crowdspace.airplace.Floo
rService");
 1355:                                  this.stopService(intent);
 1356:                          }
 1357:                  }
 1358:          }
 1359:
 1360:          /*************************************************************************
*************************************************/
 1361:          public class SimpleWifiReceiver extends WifiReceiver {
 1362:                  public void onReceive(Context c, Intent intent) {
 1363:
 1364:                          try {
 1365:                                  if (intent == null || c == null || intent.getA
ction() == null)
 1366:                                          return;
 1367:
 1368:                                  String action = intent.getAction();
 1369:
 1370:                                  if (!action.equals(WifiManager.SCAN_RESULTS_AV
AILABLE_ACTION))
 1371:                                          return;
 1372:
 1373:                                  List<ScanResult> wifiList = wifi.getScanResult
s();
 1374:                                  scanAPs.setText("AP: " + wifiList.size());
 1375:
 1376:                                  // Set in progress (true)
 1377:                                  synchronized (inProgress) {
 1378:                                          if (inProgress == true)
 1379:                                                  return;
 1380:                                          inProgress = true;
 1381:                                  }
 1382:
 1383:                                  latestScanList.clear();
 1384:                                  LogRecord lr = null;
 1385:
 1386:                                  // If we receive results, add them to latest s
can list
 1387:                                  if (wifiList != null && !wifiList.isEmpty()) {
 1388:                                          for (int i = 0; i < wifiList.size(); i
++) {
 1389:                                                  lr = new LogRecord(wifiList.ge
t(i).BSSID,
 1390:                                                                  wifiList.get(i
).level);
 1391:                                                  latestScanList.add(lr);
 1392:                                          }
 1393:                                          if (preFM) {
 1394:                                                  onSharedPreferenceChanged(Pref
erences, "modeFloor");
 1395:                                                  System.out.println("IN preFM")
;
 1396:                                                  preFM = false;
 1397:                                          }
 1398:                                  }
 1399:
 1400:                                  // Unset in progress (false)
 1401:                                  synchronized (inProgress) {
 1402:                                          inProgress = false;
 1403:
 1404:                                  }
 1405:
 1406:                                  if (trackMe.get()) {
 1407:                                          if (!FindMe_Method()) {
 1408:                                                  subMenu2.getItem().setChecked(
false);
 1409:                                                  subMenu2.getItem().setIcon(
 1410:                                                                  android.R.draw
able.ic_menu_mylocation);
 1411:                                                  trackMe.setBoolean(false);
 1412:                                                  trackMe.notifyObservers();
 1413:                                          }
 1414:                                  }
 1415:
 1416:                          } catch (RuntimeException e) {
 1417:                                  return;
 1418:                          }
 1419:
 1420:                  }
 1421:
 1422:          }
 1423:
 1424:          /*************************************************************************
*************************************************/
 1425: }
```

```
    1: package dmsl.crowdspaceSidemenu;
    2:
    3: import com.google.zxing.client.android.CaptureActivity;
    4:
    5: import dmsl.ItemScan.GoogleShoppingAPIHelperClass;
    6: import dmsl.ItemScan.ProductObject;
    7:
    8: import android.app.Activity;
    9: import android.content.Context;
   10: import android.content.Intent;
   11: import android.content.ClipData.Item;
   12: import android.os.Handler;
   13: import android.os.Message;
   14: import android.widget.Toast;
   15:
   16: /**
   17:  *
   18:  * @author Julia Metochi
   19:  *
   20:  */
   21: public class MyJavaScriptInterface extends Activity {
   22:
   23:         Context mContext;
   24:         public static int menuItem = 0;
   25:         public static String[] doc;
   26:
   27:         MyJavaScriptInterface(Context c) {
   28:                 mContext = c;
   29:         }
   30:
   31:         public void showToast(String toast) {
   32:                 Toast.makeText(mContext, toast, Toast.LENGTH_SHORT).show();
   33:         }
   34:
   35:         /**
   36:          * Get current longitude and latitude sent directly form Google Maps t
hrough
   37:          * JavaScript
   38:          *
   39:          * @param lon
   40:          * @param lat
   41:          */
   42:         public void getCurLatLon(double lon, double lat) {
   43:
   44:                 Coordinates point = new Coordinates();
   45:
   46:                 Message msg = new Message();
   47:                 point.setLat(lat);
   48:                 point.setLon(lon);
   49:
   50:                 msg.obj = point;
   51:                 if (MainActivity.flag == 1) {
   52:                         MainActivityLogger.myHandler.sendMessage(msg);
   53:                 } else if (MainActivity.flag == -1) {
   54:                         MainActivity.myHandler.sendMessage(msg);
   55:                         if (ItemMenu.MOVE_ITEM == 1) {
   56:                                 ItemMenu it = new ItemMenu();
   57:                                 String itemInfo = it
   58:                                                 .searchByDocId(MyJavaScriptInt
erface.doc[0]);
   59:                                 GoogleShoppingAPIHelperClass gHelp = new Googl
eShoppingAPIHelperClass();
   60:                                 ProductObject prObj = gHelp.productFoundInCouc
h(itemInfo);
   61:                                 if (prObj != null) {
   62:                                         prObj.setPointX(MainActivity.itemLon);
   63:                                         prObj.setPointY(MainActivity.itemLat);
   64:                                         ProductObject itemNewLocation = new Pr
oductObject(
   65:                                                         prObj.getTitle(), prOb
j.getBrand(),
   66:                                                         prObj.getPrice(), prOb
j.getBarcode(),
   67:                                                         prObj.getPointX(), prO
bj.getPointY(),
   68:                                                         prObj.getFloor(), prOb
j.getImage(),
   69:                                                         prObj.getDocId(), prOb
j.getDocRev());
   70:
   71:                                         ItemMenu.EDIT_ITEM = 1;
   72:                                         it.updateDocument(prObj.getDocId(), it
emNewLocation);
   73:                                         MainActivity.mWebview.loadUrl("javascr
ipt:clearMarkers();");
   74:                                         MainActivity ma = new MainActivity();
   75:                                         ma.showItemsByFloor(MainActivity.curre
ntFloor);
   76:
   77:                                         ItemMenu.EDIT_ITEM = 0;
   78:                                         ItemMenu.MOVE_ITEM = 0;
   79:                                 }
   80:                         }
   81:                 }
   82:
   83:         }
   84:
   85:         /**********************************************************************
**********************************************************/
   86:
   87:         /**
   88:          * Call activity ItemMenu to edit,delete or move item
   89:          *
   90:          * @param docInfo
   91:          */
   92:         public void ItemMenu(String docInfo) {
   93:                 doc = docInfo.split("@");
   94:
   95:                 Intent itemMenu = new Intent(mContext, ItemMenu.class);
   96:                 mContext.startActivity(itemMenu);
   97:         }
   98:
   99: }
```

```
    1: /*
    2:  * Copyright (c) 2011, KIOS Research Center and Data Management Systems Lab,
    3:  * University of Cyprus. All rights reserved.
    4:  *
    5:  * Redistribution and use in source and binary forms, with or without modifica
tion,
    6:  * are permitted provided that the following conditions are met:
    7:  *
    8:  *    * Redistributions of source code must retain the above copyright notice,
 this
    9:  *      list of conditions and the following disclaimer.
   10:  *    * Redistributions in binary form must reproduce the above copyright noti
ce,
   11:  *      this list of conditions and the following disclaimer in the documentat
ion
   12:  *      and/or other materials provided with the distribution.
   13:  *    * Neither the name of the Sony Ericsson Mobile Communication AB nor the
names
   14:  *      of its contributors may be used to endorse or promote products derived
 from
   15:  *      this software without specific prior written permission.
   16:  *
   17:  * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 AND
   18:  * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLI
ED
   19:  * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISC
LAIMED.
   20:  * IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DI
RECT,
   21:  * INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDI
NG,
   22:  * BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF US
E,
   23:  * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEOR
Y OF
   24:  * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIG
ENCE
   25:  * OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF A
DVISED
   26:  * OF THE POSSIBILITY OF SUCH DAMAGE.
   27:  */
   28:
   29: package dmsl.crowdspace.airplace;
   30:
   31: import dmsl.crowdspaceSidemenu.MainActivity;
   32: import dmsl.crowdspaceSidemenu.R;
   33: import android.app.Activity;
   34: import android.app.AlertDialog;
   35: import android.content.Context;
   36: import android.content.DialogInterface;
   37: import android.content.Intent;
   38: import android.content.SharedPreferences;
   39: import android.database.Cursor;
   40: import android.net.ConnectivityManager;
   41: import android.net.NetworkInfo;
   42: import android.net.Uri;
   43: import android.os.Bundle;
   44: import android.preference.Preference;
   45: import android.preference.Preference.OnPreferenceClickListener;
   46: import android.preference.PreferenceActivity;
   47: import android.provider.MediaStore.MediaColumns;
   48: import android.widget.Toast;
   49:
   50: /**
   51:  * Defines the behavior of the preferences menu.
   52:  *
   53:  * @author KIOS Research Center and Data Management Systems Lab, University of
   54:  *         Cyprus
   55:  *
   56:  */
   57: public class Prefs extends PreferenceActivity implements
   58:                 SharedPreferences.OnSharedPreferenceChangeListener {
   59:
   60:         private static final int SELECT_IMAGE = 7;
   61:         private static final int SELECT_PATH = 8;
   62:         private static final int SELECT_FILE = 9;
   63:
   64:         /**
   65:          * Build preference menu when the activity is first created.
   66:          */
   67:         @Override
   68:         protected void onCreate(Bundle savedInstanceState) {
   69:
   70:                 super.onCreate(savedInstanceState);
   71:
   72:                 // Load the appropriate preferences
   73:                 getPreferenceManager().setSharedPreferencesName(
   74:                                 MainActivity.SHARED_PREFS_INDOOR);
   75:
   76:                 addPreferencesFromResource(R.xml.preferences);
   77:
   78:                 getPreferenceManager().getSharedPreferences()
   79:                                 .registerOnSharedPreferenceChangeListener(this
);
   80:
   81:                 // Custom button to choose radio map file to use for positioni
ng
   82:                 getPreferenceManager().findPreference("radiomap_file")
   83:                                 .setOnPreferenceClickListener(new OnPreference
ClickListener() {
   84:
   85:                                         public boolean onPreferenceClick(Prefe
rence preference) {
   86:
   87:                                                 Intent i = new Intent(getBaseC
ontext(),
   88:                                                                 AndroidFileBro
wserTracker.class);
   89:
   90:                                                 Bundle extras = new Bundle();
   91:                                                 // Send flag to browse for fol
der false, it is a file
   92:                                                 // selection.
   93:                                                 extras.putInt("to_Browse", 2);
   94:
   95:                                                 i.putExtras(extras);
   96:
   97:                                                 startActivityForResult(i, SELE
CT_FILE);
   98:                                                 return true;
   99:                                         }
```

```
  100:                                });
  101:
  102:                getPreferenceManager().findPreference("modeFloor")
  103:                                .setOnPreferenceClickListener(new OnPreference
ClickListener() {
  104:
  105:                                        public boolean onPreferenceClick(Prefe
rence preference) {
  106:                                                System.out.println("Enable: "
+ preference.isEnabled()
  107:                                                                + " " + prefer
ence.isPersistent() + " "
  108:                                                                + preference.i
sSelectable());
  109:                                                if (MainActivity.isFloorMode)
{
  110:                                                        popup_msg(
  111:                                                                        "Be su
re that all the radio-maps, test-files and floor plans have the right settings.",
  112:                                                                        "Warni
ng of floor mode!", R.drawable.info);
  113:
  114:                                                }
  115:
  116:                                                return true;
  117:
  118:                                        }
  119:                                });
  120:
  121:        }
  122:
  123:        private void popup_msg(String msg, String title, int imageID) {
  124:
  125:                AlertDialog.Builder alert_box = new AlertDialog.Builder(this);
  126:                alert_box.setTitle(title);
  127:                alert_box.setMessage(msg);
  128:                alert_box.setIcon(imageID);
  129:
  130:                alert_box.setNeutralButton("Hide",
  131:                                new DialogInterface.OnClickListener() {
  132:                                        public void onClick(DialogInterface di
alog, int which) {
  133:                                                dialog.cancel();
  134:                                        }
  135:                                });
  136:
  137:                AlertDialog alert = alert_box.create();
  138:                alert.show();
  139:        }
  140:
  141:        /**
  142:         * Actions to be taken on Preference menu exit.
  143:         */
  144:        @Override
  145:        public void onActivityResult(int requestCode, int resultCode, Intent d
ata) {
  146:                super.onActivityResult(requestCode, resultCode, data);
  147:
  148:                SharedPreferences customSharedPreference;
  149:
  150:                customSharedPreference = getSharedPreferences(
  151:                                MainActivity.SHARED_PREFS_INDOOR, MODE_PRIVATE
);
  152:
  153:                switch (requestCode) {
  154:
  155:                case SELECT_IMAGE:
  156:                        if (resultCode == Activity.RESULT_OK) {
  157:                                Uri selectedImage = data.getData();
  158:                                String RealPath;
  159:                                SharedPreferences.Editor editor = customShared
Preference.edit();
  160:                                RealPath = getRealPathFromURI(selectedImage);
  161:                                editor.putString("image_custom", RealPath);
  162:                                editor.commit();
  163:                        }
  164:                        break;
  165:                case SELECT_PATH:
  166:                        if (resultCode == Activity.RESULT_OK) {
  167:                                Uri selectedFolder = data.getData();
  168:                                String path = selectedFolder.toString();
  169:                                SharedPreferences.Editor editor = customShared
Preference.edit();
  170:                                editor.putString("folder_browser", path);
  171:                                editor.commit();
  172:                        }
  173:                        break;
  174:                case SELECT_FILE:
  175:                        if (resultCode == Activity.RESULT_OK) {
  176:                                Uri selectedFile = data.getData();
  177:                                String file = selectedFile.toString();
  178:                                SharedPreferences.Editor editor = customShared
Preference.edit();
  179:                                editor.putString("radiomap_file", file);
  180:                                editor.commit();
  181:                        }
  182:                        break;
  183:                }
  184:
  185:        }
  186:
  187:        /**
  188:         * @param contentUri
  189:         * @return absolute path given a URI
  190:         */
  191:        public String getRealPathFromURI(Uri contentUri) {
  192:                String[] proj = { MediaColumns.DATA };
  193:                Cursor cursor = managedQuery(contentUri, proj, null, null, nul
l);
  194:                int column_index = cursor.getColumnIndexOrThrow(MediaColumns.D
ATA);
  195:                cursor.moveToFirst();
  196:                return cursor.getString(column_index);
  197:        }
  198:
  199:        /**
  200:         * Sets up a listener when the preference activity is resumed.
  201:         */
  202:        @Override
  203:        protected void onResume() {
  204:                super.onResume();
  205:                // Set up a listener whenever a key changes
```

```
  206:                 getPreferenceScreen().getSharedPreferences()
  207:                                 .registerOnSharedPreferenceChangeListener(this
);
  208:         }
  209:
  210:         /**
  211:          * Unregisters the listener before the preference activity is destroye
d.
  212:          */
  213:         @Override
  214:         protected void onDestroy() {
  215:                 super.onDestroy();
  216:                 // Unregister the listener whenever a key changes
  217:                 getPreferenceManager().getSharedPreferences()
  218:                                 .unregisterOnSharedPreferenceChangeListener(th
is);
  219:         }
  220:
  221:         /**
  222:          * Unregisters the listener when a preference activity is paused.
  223:          */
  224:         @Override
  225:         protected void onPause() {
  226:                 super.onPause();
  227:                 // Unregister the listener whenever a key changes
  228:                 getPreferenceScreen().getSharedPreferences()
  229:                                 .unregisterOnSharedPreferenceChangeListener(th
is);
  230:         }
  231:
  232:         /**
  233:          * Does nothing when a shared preference is changed.
  234:          */
  235:         public void onSharedPreferenceChanged(SharedPreferences arg0, String a
rg1) {
  236:         }
  237:
  238: }
```

```java
package dmsl.FloorMode;

public class RMHelp {
	private String macAddress;
	private Double rss;
	private int times;
	private boolean visible;

	/**
	 * Constructor of RMHelp class to upload the .floor file
	 *
	 * @param macAddress
	 * @param rss
	 * @param times
	 */
	public RMHelp(String macAddress, Double rss, int times) {
		this.setMacAddress(macAddress);
		this.setRss(rss);
		this.setTimes(times);
		this.setVisible(false);
	}

	/**
	 * Constructor that takes only two parameters
	 *
	 * @param macAddress
	 * @param rss
	 */
	public RMHelp(String macAddress, Double rss) {
		this.setMacAddress(macAddress);
		this.setRss(rss);
	}

	public void setMacAddress(String macAddress) {
		this.macAddress = macAddress;
	}

	public String getMacAddress() {
		return macAddress;
	}

	public void setRss(Double rss) {
		this.rss = rss;
	}

	public Double getRss() {
		return rss;
	}

	public void setTimes(int times) {
		this.times = times;
	}

	public int getTimes() {
		return times;
	}

	public String toString() {
		return macAddress + " " + rss + " " + times + " " + visible;
	}

	public void setVisible(boolean visible) {
		this.visible = visible;
	}

	public boolean isVisible() {
		return visible;
	}

}
```

```
    1: /*
    2:  * Copyright (c) 2011, KIOS Research Center and Data Management Systems Lab,
    3:  * University of Cyprus. All rights reserved.
    4:  *
    5:  * Redistribution and use in source and binary forms, with or without modifica
tion,
    6:  * are permitted provided that the following conditions are met:
    7:  *
    8:  *    * Redistributions of source code must retain the above copyright notice,
 this
    9:  *      list of conditions and the following disclaimer.
   10:  *    * Redistributions in binary form must reproduce the above copyright noti
ce,
   11:  *      this list of conditions and the following disclaimer in the documentat
ion
   12:  *      and/or other materials provided with the distribution.
   13:  *    * Neither the name of the Sony Ericsson Mobile Communication AB nor the
names
   14:  *      of its contributors may be used to endorse or promote products derived
 from
   15:  *      this software without specific prior written permission.
   16:  *
   17:  * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 AND
   18:  * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLI
ED
   19:  * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISC
LAIMED.
   20:  * IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DI
RECT,
   21:  * INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDI
NG,
   22:  * BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF US
E,
   23:  * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEOR
Y OF
   24:  * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIG
ENCE
   25:  * OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF A
DVISED
   26:  * OF THE POSSIBILITY OF SUCH DAMAGE.
   27:  */
   28:
   29: package dmsl.crowdspace.airplace;
   30:
   31: public class WeightRecord {
   32:
   33:         private float wx;
   34:         private float wy;
   35:
   36:         public WeightRecord(float weightX, float weightY) {
   37:                 this.wx = weightX;
   38:                 this.wy = weightY;
   39:         }
   40:
   41:         public float getWeightX() {
   42:                 return wx;
   43:         }
   44:
   45:         public float getWeightY() {
   46:                 return wy;
   47:         }
   48: }
```

```
    1: package dmsl.ItemScan;
    2:
    3: import dmsl.crowdspaceSidemenu.R;
    4: import android.app.Activity;
    5: import android.content.Intent;
    6: import android.os.Bundle;
    7: import android.view.View;
    8: import android.widget.Button;
    9: import android.widget.EditText;
   10: import android.widget.Toast;
   11:
   12: /**
   13:  *
   14:  * @author Julia Metochi
   15:  *
   16:  *         This activity is called when neither the product's name or product's
   17:  *         price is known. It asks the user for them and passes user's input to
   18:  *         the main activity.
   19:  *
   20:  */
   21: public class AskForInputActivity extends Activity {
   22:
   23:         public static final int MAX_TITLE_LENGTH = 60;
   24:         public static final int MAX_PRICE = 1000000;
   25:
   26:         @Override
   27:         protected void onCreate(Bundle savedInstanceState) {
   28:                 super.onCreate(savedInstanceState);
   29:                 setContentView(R.layout.prompt);
   30:
   31:                 final EditText etTitle = (EditText) findViewById(R.id.etTitle);
   32:                 final EditText etBrand = (EditText) findViewById(R.id.etBrand);
   33:                 final EditText etPrice = (EditText) findViewById(R.id.etPrice);
   34:
   35:                 Bundle getB = getIntent().getExtras();
   36:                 String barcode = getB.getString("barcode");
   37:
   38:                 this.setTitle(barcode);
   39:
   40:                 etTitle.requestFocus();
   41:
   42:                 Button bOk = (Button) findViewById(R.id.ok);
   43:                 Button bCancel = (Button) findViewById(R.id.cancel);
   44:
   45:                 bOk.setOnClickListener(new View.OnClickListener() {
   46:
   47:                         @Override
   48:                         public void onClick(View v) {
   49:                                 // TODO Auto-generated method stub
   50:
   51:                                 boolean success = false;
   52:
   53:                                 if (etPrice.getText().toString().length() == 0
   54:                                                 || etTitle.getText().toString().length() == 0
   55:                                                 || etBrand.getText().toString().length() == 0) {
   56:                                         // If any of the fields are left empty a toast message
   57:                                         // appears
   58:                                         // and the activity still waits for valid input.
   59:                                         Toast.makeText(getApplicationContext(),
   60:                                                         "One or all of the fields are empty.",
   61:                                                         Toast.LENGTH_SHORT).show();
   62:                                 } else {
   63:                                         // Some users may use ',' instead of '.' when it comes to
   64:                                         // decimal
   65:                                         // point notation so we cover this fact so Float.parseFloat
   66:                                         // works
   67:                                         // properly.
   68:                                         String strPrice = etPrice.getText().toString()
   69:                                                         .replace(',', '.');
   70:
   71:                                         try {
   72:                                                 float p = Float.parseFloat(strPrice);
   73:                                                 String t = etTitle.getText().toString();
   74:                                                 String br = etBrand.getText().toString();
   75:
   76:                                                 if (p >= MAX_PRICE) {
   77:                                                         Toast.makeText(
   78:                                                                         getApplicationContext(),
   79:                                                                         "Extreme Value, only prices below "
   80:                                                                                         + MAX_PRICE + " are accepted.",
   81:                                                                         Toast.LENGTH_SHORT).show();
   82:                                                         success = true;
   83:                                                 } else if (t.length() > MAX_TITLE_LENGTH) {
   84:                                                         Toast.makeText(
   85:                                                                         getApplicationContext(),
   86:                                                                         "Product's Title too long. Please remove "
   87:                                                                                         + (t.length() - MAX_TITLE_LENGTH)
   88:                                                                                         + " characters.",
   89:                                                                         Toast.LENGTH_SHORT).show();
   90:                                                         success = true;
   91:                                                 } else {
   92:                                                         Bundle getB = getIntent().getExtras();
   93:                                                         String barcode = getB.
```

```
getString("barcode");
   94:
   95:                                                  Intent i = new Intent(
);
   96:                                                  Bundle b = new Bundle(
);
   97:
   98:                                                  // Data to be passed t
o the main activity is put in
   99:                                                  // the bundle.
  100:
  101:                                                  b.putString("title", t
);
  102:                                                  b.putString("brand", b
r);
  103:                                                  b.putFloat("price", p)
;
  104:                                                  b.putString("barcode",
 barcode);
  105:                                                  i.putExtras(b);
  106:                                                  setResult(RESULT_OK, i
);
  107:                                                  success = true;
  108:                                                  finish();
  109:                                          }
  110:
  111:                                  } catch (NumberFormatException nfe) {
  112:                                          // Indication that an error oc
curred.
  113:                                          success = false;
  114:                                  } finally {
  115:                                          if (!success) {
  116:                                                  // If Float.parseFloat
 didn't work normally because
  117:                                                  // of invalid input,
  118:                                                  // a toast message app
ears and the activity still
  119:                                                  // waits for valid
  120:                                                  // input.
  121:                                                  Toast.makeText(getAppl
icationContext(),
  122:                                                                  "Inval
id price format", Toast.LENGTH_SHORT)
  123:                                                                  .show(
);
  124:                                          }
  125:                                  }
  126:
  127:                          }
  128:                  }
  129:          });
  130:
  131:          bCancel.setOnClickListener(new View.OnClickListener() {
  132:
  133:                  @Override
  134:                  public void onClick(View v) {
  135:                          setResult(RESULT_CANCELED);
  136:                          finish();
  137:                  }
  138:          });
  139:  }
  140:
  141: }
```

```
    1: package dmsl.crowdspace.airplace;
    2:
    3: import dmsl.crowdspaceSidemenu.MainActivity;
    4: import dmsl.crowdspaceSidemenu.MainActivityLogger;
    5: import android.app.Activity;
    6: import android.content.Context;
    7: import android.hardware.GeomagneticField;
    8: import android.hardware.Sensor;
    9: import android.hardware.SensorEvent;
   10: import android.hardware.SensorEventListener;
   11:
   12: import android.hardware.SensorManager;
   13:
   14: /**
   15:  * @author Julia Metochi
   16:  *
   17:  */
   18:
   19: // Determine the heading of the phone
   20: public class Heading implements SensorEventListener {
   21:
   22:         private SensorManager mSensorManager = null;
   23:         private MainActivity mainAct;
   24:         public static float azimuth;
   25:         private Sensor orientation;
   26:
   27:         /*********************************************************************
**********************************************/
   28:
   29:         public Heading(Activity activity) {
   30:
   31:                 mSensorManager = (SensorManager) activity
   32:                                 .getSystemService(Context.SENSOR_SERVICE);
   33:
   34:         }
   35:
   36:         /*********************************************************************
**********************************************/
   37:
   38:         public void pause() {
   39:
   40:                 mSensorManager.unregisterListener(this);
   41:         }
   42:
   43:         /*********************************************************************
**********************************************/
   44:
   45:         public void resume() {
   46:                 registerSensors();
   47:         }
   48:
   49:         /*********************************************************************
**********************************************/
   50:
   51:         private void registerSensors() {
   52:                 // get orientation sensor
   53:                 orientation = mSensorManager.getDefaultSensor(Sensor.TYPE_ORIE
NTATION);
   54:
   55:                 // check if the device has a default orientation sensor
   56:                 if (orientation == null) {
   57:                         return;
   58:                 }
   59:
   60:                 if (orientation != null) {
   61:                         mSensorManager.registerListener(this, orientation,
   62:                                         SensorManager.SENSOR_DELAY_GAME);
   63:                 }
   64:
   65:         }
   66:
   67:         /*********************************************************************
**********************************************/
   68:
   69:         // Save heading every time Sensor detects motion
   70:         public void onSensorChanged(SensorEvent event) {
   71:
   72:                 azimuth = event.values[0];
   73:
   74:                 if (MainActivity.flag == -1)
   75:                         MainActivity.updateInfoLabel();
   76:                 else if (MainActivity.flag == 1) {
   77:                         MainActivityLogger.updateInfoLabel();
   78:                 }
   79:
   80:         }
   81:
   82:         /*********************************************************************
**********************************************/
   83:
   84:         public void onAccuracyChanged(Sensor sensor, int accuracy) {
   85:
   86:         }
   87:
   88: }
```

```
    1: package dmsl.FloorMode;
    2:
    3: import java.io.BufferedReader;
    4: import java.io.DataInputStream;
    5: import java.io.FileInputStream;
    6: import java.io.FileNotFoundException;
    7: import java.io.IOException;
    8: import java.io.InputStreamReader;
    9: import java.util.ArrayList;
   10:
   11: public class ParseRMHelp {
   12:         private BufferedReader br = null;
   13:         private String nameOfFile = null;
   14:         private ArrayList<RMHelp> arrayRMH = new ArrayList<RMHelp>();
   15:
   16:         private ArrayList<String> inArray = new ArrayList<String>();
   17:
   18:         public ParseRMHelp(BufferedReader buf) throws IOException {
   19:
   20:                 br = buf;
   21:                 String line = null;
   22:                 while ((line = br.readLine()) != null) {
   23:                         inArray.add(line);
   24:                 }
   25:
   26:                 br.close();
   27:                 this.createRMHelp();
   28:         }
   29:
   30:         private void createRMHelp() {
   31:
   32:                 for (String str : inArray) {
   33:                         String[] temp = str.split(", ");
   34:                         arrayRMH.add(new RMHelp(temp[0], Double.valueOf(temp[1
]), Integer
   35:                                         .valueOf(temp[2])));
   36:                 }
   37:         }
   38:
   39:         public ArrayList<RMHelp> getArrayRMH() {
   40:                 return arrayRMH;
   41:         }
   42: }
```

```java
/*
 * Copyright (c) 2011, KIOS Research Center and Data Management Systems Lab,
 * University of Cyprus. All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without modification,
 * are permitted provided that the following conditions are met:
 *
 *    * Redistributions of source code must retain the above copyright notice, this
 *      list of conditions and the following disclaimer.
 *    * Redistributions in binary form must reproduce the above copyright notice,
 *      this list of conditions and the following disclaimer in the documentation
 *      and/or other materials provided with the distribution.
 *    * Neither the name of the Sony Ericsson Mobile Communication AB nor the names
 *      of its contributors may be used to endorse or promote products derived from
 *      this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
 * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.
 * IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT,
 * INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING,
 * BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
 * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE
 * OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
 * OF THE POSSIBILITY OF SUCH DAMAGE.
 */

package dmsl.crowdspace.airplace;

import android.os.Handler;

import java.io.*;
import java.net.Socket;

/**
 * Settings needed to communicate with server in order to download radiomap
 *
 * @author KIOS Research Center and Data Management Systems Lab, University of
 *         Cyprus
 *
 */
public class DownloadingSettings extends Thread {

        /** Server IP */
        private final String IP;
        /** Server Port */
        private final String PORT;
        /** Filename of Radiomap on the server to download */
        private final String filename_radiomap_download;
        /** Folder where the Radiomap should be downloaded */
        private final String folder_path;

        private String errMsg;
        private final Handler handler;

        /**
         * Constructor
         *
         * @param IP
         *            the IP address of radiomap distribution server
         *
         * @param PORT
         *            the port that radiomap distribution server is listening
         * */
        public DownloadingSettings(String IP, String PORT, String folder_path,
                        String filename, Handler handler) {
                this.filename_radiomap_download = filename;
                this.PORT = PORT;
                this.IP = IP;
                this.folder_path = folder_path;
                this.handler = handler;
        }

        public String getErrMsg() {
                return errMsg;
        }

        /**
         * Establishes a connection on IP/PORT and download radiomap
         *
         * */
        public void run() {

                Socket connection = null;
                FileOutputStream fos = null;
                File root = new File(folder_path);

                String radiomap_mean = filename_radiomap_download;
                String rbf_weights = radiomap_mean + "-rbf-weights";
                String parameters = radiomap_mean + "-parameters";

                try {

                        // Create new socket
                        connection = new Socket(IP, Integer.parseInt(PORT));

                        // Check that path is writable
                        if (root.canWrite()) {
                                fos = new FileOutputStream(new File(root, radiomap_mean), false);
                        } else {
                                errMsg = "Directory: "
                                                + root.getAbsolutePath()
                                                + " is not writable.\nYou may need an external memory card";
                                handler.sendEmptyMessage(-2);
```

```
  106:                            return;
  107:                    }
  108:
  109:                    PrintWriter out = new PrintWriter(connection.getOutput
Stream(),
  110:                                    true);
  111:                    BufferedReader in = new BufferedReader(new InputStream
Reader(
  112:                                    connection.getInputStream()));
  113:                    String inputLine, outputLine;
  114:
  115:                    inputLine = in.readLine();
  116:
  117:                    if (!inputLine.equalsIgnoreCase("+OK READY")) {
  118:                            fos.close();
  119:                            out.close();
  120:                            in.close();
  121:                            connection.close();
  122:                            errMsg = "Server not ready.\nTry again later."
;
  123:                            handler.sendEmptyMessage(-2);
  124:                            return;
  125:                    }
  126:
  127:                    outputLine = "GET radiomap";
  128:                    out.println(outputLine);
  129:                    inputLine = in.readLine();
  130:
  131:                    if (!inputLine.startsWith("RADIOMAP")) {
  132:                            fos.close();
  133:                            out.close();
  134:                            in.close();
  135:                            connection.close();
  136:                            errMsg = inputLine + "";
  137:                            handler.sendEmptyMessage(-2);
  138:                            return;
  139:                    }
  140:
  141:                    inputLine = inputLine.replaceFirst("RADIOMAP ", "");
  142:                    fos.write((inputLine + "\n").getBytes());
  143:
  144:                    // Get files: radiomap, parameters and rbf weights
  145:                    while ((inputLine = in.readLine()) != null) {
  146:                            if (inputLine.compareTo("null") == 0
  147:                                            || inputLine.startsWith("CORRU
PTED"))
  148:                                    break;
  149:
  150:                            if (inputLine.equalsIgnoreCase("PARAMETERS"))
{
  151:                                    fos = new FileOutputStream(new File(ro
ot, parameters),
  152:                                                    false);
  153:                            } else if (inputLine.equalsIgnoreCase("RBF_WEI
GHTS")) {
  154:                                    fos = new FileOutputStream(new File(ro
ot, rbf_weights),
  155:                                                    false);
  156:                            } else {
  157:                                    fos.write((inputLine + "\n").getBytes(
));
  158:                            }
  159:                    }
  160:
  161:                    fos.close();
  162:                    out.close();
  163:                    in.close();
  164:                    connection.close();
  165:
  166:                    errMsg = null;
  167:                    handler.sendEmptyMessage(-2);
  168:
  169:            } catch (Exception e) {
  170:                    errMsg = "Error: " + e.getMessage();
  171:                    handler.sendEmptyMessage(-2);
  172:            }
  173:
  174:    }
  175:
  176: }
```

```
    1: package dmsl.crowdspace.airplace;
    2:
    3: import java.io.BufferedReader;
    4: import java.io.File;
    5: import java.io.IOException;
    6: import java.io.InputStreamReader;
    7: import java.util.ArrayList;
    8:
    9: import dmsl.FloorMode.FloorProcedure;
   10: import dmsl.FloorMode.ParseRMHelp;
   11: import dmsl.FloorMode.RMHelp;
   12: import dmsl.crowdspaceSidemenu.MainActivity;
   13:
   14: import android.app.Service;
   15: import android.content.Intent;
   16: import android.content.res.AssetManager;
   17: import android.os.Environment;
   18: import android.os.Handler;
   19: import android.os.IBinder;
   20: import android.widget.Toast;
   21:
   22: public class FloorService extends Service {
   23:         private final int PERIODIC_EVENT_TIMEOUT = 5000;
   24:         public static final String MOVEMENT_UPDATE = "dmsl.airplace.FloorServi
ce.action.MOVEMENT_UPDATE";
   25:         public static final String MSG_SEND = "dmsl.airplace.FloorService.MSG_
SEND";
   26:         private Handler periodicEvent;
   27:         private int floor = 0;
   28:
   29:         @Override
   30:         public IBinder onBind(Intent intent) {
   31:                 return null;
   32:         }
   33:
   34:         @Override
   35:         public void onCreate() {
   36:
   37:                 periodicEvent = new Handler();
   38:                 periodicEvent.postDelayed(doPeriodicTask, PERIODIC_EVENT_TIMEO
UT);
   39:                 this.floor = MainActivity.currentFloor;
   40:
   41:         }
   42:
   43:         private Runnable doPeriodicTask = new Runnable() {
   44:
   45:                 public void run() {
   46:
   47:                         /**
   48:                          * Find first where the user Is
   49:                          */
   50:                         // Array max contains 1st the no of floor 2nd the scor
e of floor
   51:                         int[] max = { MainActivity.currentFloor, -800 };
   52:
   53:                         // Stores the current floors that every time compares
   54:
   55:                         ArrayList<Integer> tempFloors = new ArrayList<Integer>
();
   56:
   57:                         String resultOfFloors = "";
   58:
   59:                         // Take the names of all .floor files and take the con
tent of all
   60:                         // the .floor files (sort)
   61:                         ArrayList<ArrayList<RMHelp>> arrayRMHelp = readTheFloo
rFiles(tempFloors);
   62:                         int[][] floorsScore = new int[tempFloors.size()][2];
   63:                         for (int i = 0; i < tempFloors.size(); ++i) {
   64:                                 floorsScore[i][0] = tempFloors.get(i);
   65:                         }
   66:
   67:                         ArrayList<ArrayList<RMHelp>> arrayRMHelpTemp = new Arr
ayList<ArrayList<RMHelp>>();
   68:
   69:                         for (int i = 0; i < arrayRMHelp.size() - 1; ++i) {
   70:                                 arrayRMHelpTemp.add(arrayRMHelp.get(i));
   71:                                 arrayRMHelpTemp.add(arrayRMHelp.get(i + 1));
   72:                                 FloorProcedure findFloor = new dmsl.FloorMode.
FloorProcedure(
   73:                                                 arrayRMHelpTemp);
   74:                                 boolean res = MainActivity.latestScanList.isEm
pty();
   75:                                 if (res) {
   76:                                         int g = 0;
   77:                                         return;
   78:                                 }
   79:
   80:                                 // Take the score of the two floors that are i
n competition and
   81:                                 // the max
   82:
   83:                                 resultOfFloors = findFloor
   84:                                                 .findTheMAC(MainActivity.lates
tScanList);
   85:                                 // Parse the result
   86:                                 String[] tempStr = resultOfFloors.split(" ");
   87:
   88:                                 // Assign the score to each floor
   89:                                 String[] temp = tempStr[0].split(",");
   90:                                 floorsScore[i][1] = Integer.valueOf(temp[1]);
   91:
   92:                                 temp = tempStr[1].split(",");
   93:                                 floorsScore[i + 1][1] = Integer.valueOf(temp[1
]);
   94:
   95:                                 arrayRMHelpTemp.clear();
   96:
   97:                         }
   98:
   99:                         /**
  100:                          * Find the highest score
  101:                          */
  102:                         for (int i = 0; i < tempFloors.size(); ++i) {
  103:                                 if (floorsScore[i][1] > max[1]) {
  104:
  105:                                         max[0] = floorsScore[i][0];
  106:                                         max[1] = floorsScore[i][1];
  107:                                 }
  108:                         }
  109:
```

```
  110:                        // Inform the main thread if and ONLY if the view must
 change!
  111:                        if (max[0] != floor && !MainActivity.latestScanList.is
Empty()) {
  112:                                informFindMeForChanges((double) max[0]);
  113:                                floor = MainActivity.currentFloor = max[0];
  114:                        }
  115:                        periodicEvent.postDelayed(doPeriodicTask, PERIODIC_EVE
NT_TIMEOUT);
  116:                }
  117:
  118:        };
  119:
  120:        /**
  121:         * Inform the FindMe class if the floor is changed
  122:         *
  123:         * @param newValue
  124:         *            new floor
  125:         */
  126:
  127:        private void informFindMeForChanges(double newValue) {
  128:                Intent intent = new Intent(MOVEMENT_UPDATE);
  129:                intent.putExtra(MSG_SEND, newValue);
  130:                sendBroadcast(intent);
  131:        }
  132:
  133:        /**
  134:         * Method that reads all the .floor files. Files that will help the de
vice
  135:         * to decide in which floor is the user
  136:         *
  137:         * @param tempFloors
  138:         *            the no of all the floors
  139:         * @return floorFiles a list of the name of the files
  140:         */
  141:        private ArrayList<ArrayList<RMHelp>> readTheFloorFiles(
  142:                        ArrayList<Integer> tempFloors) {
  143:                ArrayList<ArrayList<RMHelp>> rmh = new ArrayList<ArrayList<RMH
elp>>();
  144:                File sdCardDir = Environment.getExternalStorageDirectory();
  145:                ArrayList<String> floorFiles = new ArrayList<String>();
  146:                String files;
  147:                String floor;
  148:
  149:                AssetManager assetManager = getAssets();
  150:                String[] listOfFiles = null;
  151:                try {
  152:                        listOfFiles = assetManager.list("");
  153:                } catch (IOException e1) {
  154:                        // TODO Auto-generated catch block
  155:                        e1.printStackTrace();
  156:                }
  157:
  158:                for (int i = 0; i < listOfFiles.length; i++) {
  159:
  160:                        files = listOfFiles[i];
  161:                        if (files.endsWith(".floor")) {
  162:                                floor = files.substring(files.length() - 8, fi
les.length() - 6);
  163:
  164:                                if (floor.contains("-")) {
  165:                                        tempFloors.add(Integer.valueOf("-" + f
loor.substring(1)));
  166:                                } else {
  167:                                        tempFloors.add(Integer.valueOf(floor.s
ubstring(1)));
  168:                                }
  169:                                try {
  170:                                        BufferedReader reader = new BufferedRe
ader(
  171:                                                        new InputStreamReader(
getAssets().open(files)));
  172:
  173:                                        ParseRMHelp newRMFloor = new ParseRMHe
lp(reader);
  174:                                        rmh.add(newRMFloor.getArrayRMH());
  175:                                } catch (IOException e) {
  176:                                        e.printStackTrace();
  177:                                }
  178:                                floorFiles.add(files);
  179:                        }
  180:
  181:                }
  182:                return rmh;
  183:        }
  184:
  185:        @Override
  186:        public void onDestroy() {
  187:                periodicEvent.removeCallbacks(doPeriodicTask);
  188:                super.onDestroy();
  189:        }
  190:
  191:        @Override
  192:        public void onStart(Intent intent, int startid) {
  193:        }
  194:
  195: }
```