ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

Ανάπτυξη εφαρμογής παραγγελιοληψίας σε Android

Development of an ordering application in Android

Διπλωματική Εργασία

του

Χρήστου Κυρίτση

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

Ανάπτυξη εφαρμογής παραγγελιοληψίας σε Android

Development of an ordering application in Android

Διπλωματική Εργασία

του

Χρήστου Κυρίτση

Υπεύθυνος καθηγητής: **Αλκιβιάδης Γ. Ακρίτας**
Καθηγητής ΠΘ

Επιβλέπων Καθηγητής: **Γεώργιος Σταμούλης**
Καθηγητής ΠΘ

Βόλος, 2015

Εγκρίθηκε από τη διμελή εξεταστική επιτροπή την ημερομηνία εξέτασης.

(Υπογραφή)	*(Υπογραφή)*
(……)	(……)
ΚΥΡΙΟΣ ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ ΠΘ	ΔΕΥΤΕΡΕΥΩΝ ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ ΠΘ

Βόλος, Σεπτέμβριος 2015

(Υπογραφή)

.....

Κυρίστης Χρήστος

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών,
Πανεπιστημίου Θεσσαλίας

Στην οικογένεια μου και στους φίλους μου…

Ευχαριστίες

Με την περάτωση της παρούσας εργασίας, θα ήθελα να ευχαριστήσω θερμά τους επιβλέποντες καθηγητές μου, κύριο Αλκιβιάδη Ακρίτα και κύριο Γεώργιο Σταμούλη, για την υποστήριξη και εμπιστοσύνη που επέδειξαν στο πρόσωπό μου, την άριστη συνεργασία και τις ουσιώδεις υποδείξεις που διευκόλυναν την εκπόνηση της διπλωματικής μου εργασίας.

Ακόμα, θα ήθελα να ευχαριστήσω τους φίλους και συμφοιτητές μου, για την απαράμιλλη βοήθεια και τη στήριξη που μου παρείχαν κατά τη διάρκεια αυτών των ετών.

Τέλος, ένα μεγάλο ευχαριστώ στην οικογένεια μου, της οποίας η συμπαράσταση και υποστήριξη με βοήθησαν καθ' όλη τη διάρκεια των σπουδών μου.

Χρήστος Κυρίτσης
Βόλος, 2015

Περίληψη

Όλοι παρατηρούμε πλέον, ότι η τεχνολογία έχει εισέλθει στην ζωή και την καθημερινότητα μας σε έναν πολύ σημαντικό βαθμό. Άνθρωποι όλου του φάσματος των ηλικιών, χρησιμοποιούν υπολογιστές, κινητά τηλέφωνα και άλλες ηλεκτρονικές συσκευές με σκοπό, αν μη τι άλλο, την διευθέτηση ενός μεγάλου ποσοστού των υποχρεώσεών τους. Έτσι, γίνεται ευθέως αντιληπτό ότι η τεχνολογία, η οποία επιδέχεται καθημερινή και ταχύτατη εξέλιξη, διευκολύνει τη ζωή μας και προσδίδει ένα μεγαλύτερο επίπεδο ποιότητας σε αυτή.

Η παρούσα πτυχιακή εργασία έχει ως στόχο την υλοποίηση μιας εφαρμογής σε περιβάλλον Android. Η δημιουργία της εφαρμογής αυτής έγινε με την χρήση του λογισμικού Android Studio, το οποίο είναι ένα ολοκληρωμένο περιβάλλον μέσα από το οποίο μπορούμε να γράψουμε και να εκτελέσουμε κώδικα. Είναι ελεύθερης διανομής (freeware) και ανοικτού κώδικα (open source).

Η εφαρμογή θα μπορεί να χρησιμοποιηθεί από όλους τους χρήστες κινητών με λειτουργικό Android. Αφορά στη δημιουργία εφαρμογής /προϊόντος, που παρέχει τη δυνατότητα στον πελάτη/χρήστη να παραγγέλνει καφέ και άλλα είδη online, μέσω smartphone, από μια καφετέρια, η οποία θα έχει προμηθευτεί την εν λόγω εφαρμογή εξυπηρέτησης πελατών, χωρίς ο ίδιος να βρίσκεται στο χώρο της.

Περιεχόμενα

1. Εισαγωγή

1.1 Έξυπνα τηλέφωνα/Smartphones

Τα smartphones, αποτελούν την φυσική εξέλιξη των κλασικών συσκευών κινητής τηλεφωνίας. Δεν υπάρχει ένας γενικά αποδεκτός ορισμός τους, αλλά γενικά θα λέγαμε πως ένα smartphone είναι μια συσκευή τηλεπικοινωνίας, η οποία έχει επιπρόσθετα την δυνατότητα να πραγματοποιήσει κάποιες από τις εργασίες που εκτελούν οι προσωπικοί υπολογιστές, όπως την λήψη και αποστολή emails, την επεξεργασία κειμένων κ.λ.π. Τα smartphones είναι αποτέλεσμα της σύζευξης των κλασσικών κινητών τηλεφώνων με τα Personal Digital Assistants (PDA), τα οποία ήταν στην πράξη ηλεκτρονικές φορητές ατζέντες, που μπορούσαν να επικοινωνήσουν με τον υπολογιστή για ανταλλαγή στοιχείων.

Στην ιστορία των smartphones, η εταιρία Research in Motion (RIM) κατέχει μια ιδιαίτερη θέση, αφού από το 2002 διαθέτει στην αγορά το BlackBerry, όνομα το οποίο για αρκετά χρόνια αποτελούσε τον ορισμό του «smartphone». Το εργονομικό «qwerty» πληκτρολόγιο και η δυνατότητα ασφαλούς λήψης και αποστολής email κατευθείαν μέσω του δικτύου κινητής τηλεφωνίας, κατέστησε το BlackBerry το κατεξοχήν μοντέλο κινητού το οποίο μπορούσε να ανταποκριθεί στις απαιτήσεις των ατόμων που χρειάζεται να δουλεύουν εν κινήσει.

Ίσως το κυριότερο χαρακτηριστικό που ξεχωρίζει τα smartphones, πέραν της εμφάνισης, είναι το λειτουργικό σύστημα που χρησιμοποιούν. Ίσως να φαίνεται λίγο περίεργο για τους χρήστες οι οποίοι δεν είχαν συνηθίσει να επιλέγουν κάποιο κινητό με κριτήριο το λογισμικό, εντούτοις αποτελεί μια σημαντική παράμετρο, η οποία προδιαγράφει τις πραγματικές δυνατότητες της συσκευής. Το λειτουργικό σύστημα μπορεί να παρέχεται από εταιρία διαφορετική από την κατασκευάστρια της συσκευής. Μέσω αυτού, ο χρήστης μπορεί να εγκαταστήσει στην συσκευή του εφαρμογές οι οποίες θα είναι συμβατές με το λειτουργικό. Αυτό σίγουρα είναι μια βελτίωση από άλλες τεχνολογίες στα κινητά, όπου λ.χ. κάποιες εφαρμογές σε Java μπορεί να μην έτρεχαν σε διαφορετικά μοντέλα κινητών. Μέσω του λειτουργικού συστήματος λοιπόν, μπορούν οι εφαρμογές να εκμεταλλευτούν την τεχνολογία που χρησιμοποιεί η συσκευή smartphone. Κάποιες κατασκευάστριες εταιρίες, όπως η Apple και RIM, χρησιμοποιούν τα δικά τους, «κλειστά» λειτουργικά, με αποτέλεσμα οι εφαρμογές που μπορούν να εγκατασταθούν να έχουν περιορισμούς. Αντίθετα, άλλες εταιρίες που

χρησιμοποιούν λειτουργικά ανοικτού κώδικα όπως είναι το Android, Linux (το Symbian θα εξελιχθεί σύντομα σε λειτουργικό ανοικτού κώδικα), μπορούν να δώσουν περισσότερη δύναμη στον προγραμματιστή εφαρμογών για το smartphone.

Ανάμεσα σε άλλες, οι κυριότερες δυνατότητες των smartphones είναι η αποστολή και λήψη email, η πρόσβαση στο διαδίκτυο, η λήψη φωτογραφιών ή και βίντεο υψηλής ευκρίνειας, η οθόνη αφής, η ύπαρξη πλήρους «qwerty» πληκτρολογίου (με πλήκτρα ή δυνητικό στην οθόνη), καθώς και η δυνατότητα ανάγνωσης κειμένων. Εξίσου σημαντικά χαρακτηριστικά είναι η δυνατότητα αναπαραγωγής μουσικής και η εμφάνιση φωτογραφιών και βίντεο στην οθόνη. Επιπρόσθετες δυνατότητες αποτελεί η ύπαρξη συστήματος εντοπισμού θέσης (π.χ. A-GPS). Πρόσφατα διάφορα smartphones παρέχονται με εσωτερικό επιταχυσιόμετρο, προσφέροντας την δυνατότητα για πιο εξελιγμένο και φυσικό χειρισμό της συσκευής.

Επιλέγοντας κάποιο smartphone, οι παράμετροι που μπορεί να επηρεάσουν την απόφαση είναι το υλικό μέρος, στο οποίο περιλαμβάνεται η αισθητική και ο εργονομικός σχεδιασμός, η ποιότητα της οθόνης, η ευκρίνεια της κάμερας, καθώς και ο χρόνος αυτονομίας. Μια ακόμη σημαντική παράμετρος είναι η ύπαρξη εφαρμογών για το λειτουργικό σύστημα, από παιχνίδια μέχρι εξειδικευμένες υπηρεσίες. Μάλιστα πολλές εταιρίες ξεκίνησαν να δημιουργούν ηλεκτρονικά καταστήματα εφαρμογών, όπως το App Store για τα κινητά που τρέχουν το iPhone Operating System της Apple και το Ovi Store για κινητά που τρέχουν το Symbian OS, το οποίο υποστηρίζεται από την Nokia.

Το 2014 ήταν η χρονιά των smartphones, με τις πωλήσεις στους τελικούς χρήστες να ξεπερνούν, σύμφωνα με τα στοιχεία της *Gartner*, το ένα δισεκατομμύριο, αγγίζοντας τα 1,2 δισεκατομμύρια, παρουσιάζοντας αύξηση κατά 28,4% σε σχέση με το 2013. Η Samsung και η Apple είναι όπως αναμενόταν οι εταιρείες με τις περισσότερες πωλήσεις στη διάρκεια της χρονιάς, πετυχαίνοντας μαζί σχεδόν 500 εκατομμύρια πωλήσεις και συγκεντρώνοντας το 40,1% της παγκόσμιας αγοράς. Συγκεκριμένα, η Samsung με 307,6 εκατομμύρια πωλήσεων βρέθηκε στην πρώτη θέση των κατασκευαστών, με ποσοστό 24,7% και η Apple, με 191,4 εκατομμύρια πωλήσεις στη δεύτερη θέση, με ποσοστό 15,4%. Στην τρίτη θέση βρίσκεται η Lenovo, αρκετά μακριά από τις δύο πρώτες, με 81,4 εκατομμύρια πωλήσεις (μαζί με τις πωλήσεις της Motorola, μετά την εξαγορά της) και ποσοστό 6,5% επί των παγκόσμιων πωλήσεων. Ακολουθούν η Huawei και η LG, με 21 εκατομμύρια και 18,5

εκατομμύρια πωλήσεις αντίστοιχα. Αξίζει να σημειωθεί ότι οι υπόλοιποι κατασκευαστές πραγματοποίησαν συνολικά 538,7 εκατομμύρια πωλήσεις σε τελικούς χρήστες. Όσο αναφορά τις πωλήσεις βάσει λειτουργικού συστήματος, το Android είναι ο απόλυτος κυρίαρχος για το 2014. Συγκεκριμένα, κατέχει το 80,4% της αγοράς, με το iPhone να ακολουθεί με το 15,4%. Μόλις 35 εκατομμύρια συσκευές είναι τα smartphones που πουλήθηκαν το 2014 με Windows Phone, που είναι αρκετές για την τρίτη θέση και το 2,8% της αγοράς. Τέλος πωλήθηκαν 7,9 εκατομμύρια smartphones με το λειτουργικό της BlackBerry (0,6%) ενώ στην αγορά βρέθηκαν και 5,7 εκατομμύρια συσκευές (0,5%) με άλλο λειτουργικό.

Εικόνα 1. Έξυπνα τηλέφωνα

1.2 Ορισμός Συστήματος Παραγγελιοληψίας

Αρχικά, αξίζει να σημειωθεί ότι ο τρόπος που γίνεται μια οποιαδήποτε παραγγελία σήμερα, διαφέρει σημαντικά σε σχέση με αυτόν που ήταν καθιερωμένος τα προηγούμενα χρόνια. Όσο η τεχνολογία εξελίσσεται και μπαίνει ολοένα και πιο δυναμικά στις ζωές μας, πολλά πράγματα τείνουν να ακολουθούν και να προσαρμόζονται στην εξέλιξη αυτή.

Ένα σύστημα παραγγελιοληψίας, είναι ουσιαστικά ένα σύνολο από προγράμματα και εφαρμογές, με σκοπό την απομακρυσμένη επικοινωνία δύο μηχανημάτων με συγκεκριμένο τρόπο. Όπως και στα συστήματα τηλεπικοινωνιών, υπάρχει μία εφαρμογή πομπός και μία εφαρμογή δέκτης, και η επικοινωνία αυτή μπορεί να είναι μονόδρομη ή και αμφίδρομη. Έτσι, η επικοινωνία αυτή επιτυγχάνεται μέσω της αποστολής πληροφορίας πάνω από το δίκτυο, πράγμα που προϋποθέτει την ενσωμάτωση κατάλληλων μηχανισμών ασφαλείας και άλλων πρακτικών. Πιο συγκεκριμένα, μέσω κατάλληλης διεπαφής, ο χρήστης καλείται να επιλέξει τα προϊόντα που επιθυμεί, να συμπληρώσει τα στοιχεία που απαιτούνται για την διεξαγωγή της παραγγελίας και να πραγματοποιήσει την αποστολή της. Από την άλλη πλευρά, ο υπάλληλος της εκάστοτε εταιρίας που χρησιμοποιεί τέτοιου είδους σύστημα παραγγελιοληψίας, θα πρέπει με τη βοήθεια συγκεκριμένης εφαρμογής, να ενημερώνεται για τις νέες παραγγελίες που φτάνουν στο κατάστημα, τις οποίες πρέπει να διοχετεύει στους κατάλληλους ανθρώπους που είναι υπεύθυνοι για την εκτέλεση της.

1.3 Γενική περιγραφή παρόντος συστήματος

Το σύστημα που κατασκευάστηκε και αναλύεται στην παρούσα διπλωματική εργασία αφορά στη δημιουργία εφαρμογής/προϊόντος, που παρέχει τη δυνατότητα στον πελάτη/χρήστη να παραγγέλνει καφέ online ,μέσω smartphone, από μια καφετέρια καφέ και άλλα είδη που αυτή προσφέρει, η οποία θα έχει προμηθευτεί την εν λόγω εφαρμογή εξυπηρέτησης πελατών, χωρίς ο ίδιος να βρίσκεται στο χώρο της. Ο πελάτης/χρήστης αρχικά θα πρέπει μέσω internet να κατεβάσει την εφαρμογή από το Google Play και στη συνέχεια με σχετικά απλή διαδικασία θα επιλέγει το προϊόν που επιθυμεί από το κατάστημα της αρεσκείας του. Η χρέωση των προϊόντων που θα παραγγέλνει ο πελάτης/χρήστης θα γίνεται είτε μέσω χρεωστικής κάρτας είτε με μετρητά.

Επομένως, η εφαρμογή iOrder επιγραμματικά είναι:

1.μια εφαρμογή για smartphones, που θα δίνει τη δυνατότητα στο χρήστη να παραγγέλνει online τον καφέ ή το snack του, ενώ είναι καθ οδόν.
2.μια διαδικτυακή πλατφόρμα για τα συνεργαζόμενα καταστήματα, που μέσω αυτής θα εξυπηρετούν με αυτόν τον τρόπο τους πελάτες τους.

Η διαδικασία λειτουργίας της θα είναι η εξής: ο χρήστης θα εισέρχεται στην εφαρμογή, θα επιλέγει το κατάστημα από το οποίο θέλει να παραγγείλει. Στη συνέχεια, θα εμφανίζεται ο αντίστοιχος κατάλογος της επιχείρησης και αφού επιλέξει τα προϊόντα που επιθυμεί, θα εμφανίζεται το συνολικό τους κόστος, των οποίων η χρέωση θα γίνεται με τη χρήση τόσο χρεωστικής κάρτας όσο και με μετρητά, ανάλογα με την προτίμηση του πελάτη. Θα ενημερώνεται ταυτόχρονα για το χρόνο που απομένει για να ετοιμαστεί η παραγγελία του. Σε κάθε παραγγελία θα αποδίδεται ένα ξεχωριστό αναγνωριστικό, το οποίο θα χρησιμοποιείται από το χρήστη ως αποδεικτικό στο κατάστημα για τη λήψη της παραγγελίας του. Η εφαρμογή θα διατίθεται από το Google Play για Android κινητά.

Στα πλαίσια της παρούσας εργασίας θα παρουσιάσουμε αναλυτικά τη διαδικασία που ακολουθήσαμε για την ανάπτυξη και τη σχεδίαση της εφαρμογής του χρήστη smartphone.

2. Το λειτουργικό Android

2.1 Γενικά για το Android

Το Android είναι λειτουργικό σύστημα για συσκευές κινητής τηλεφωνίας το οποίο τρέχει τον πυρήνα του λειτουργικού Linux. Αρχικά αναπτύχθηκε από την Google και αργότερα από την [Handset Alliance | Open Handset Alliance]. Επιτρέπει στους κατασκευαστές λογισμικού να συνθέτουν κώδικα με την χρήση της γλώσσας προγραμματισμού Java, ελέγχοντας την συσκευή μέσω βιβλιοθηκών λογισμικού ανεπτυγμένων από την Google. Το Android είναι κατά κύριο λόγο σχεδιασμένο για συσκευές με οθόνη αφής, όπως τα έξυπνα τηλεφωνα και τα τάμπλετ, με διαφορετικό περιβάλλον χρήσης για τηλεοράσεις (Android TV), αυτοκίνητα (Android Auto) και ρολόγια χειρός (Android Wear). Παρόλο που έχει αναπτυχθεί για συσκευές με οθόνη αφής, έχει χρησιμοποιηθεί σε κονσόλες παιχνιδιών, ψηφιακές φυτογραφικές μηχανές, συνηθισμένους Η/Υ (π.χ. το HP Slate 21) και σε άλλες ηλεκτρονικές συσκευές.

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

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

Εικόνα 2. Το λογότυπο του Android

2.2 Ιστορική αναδρομή

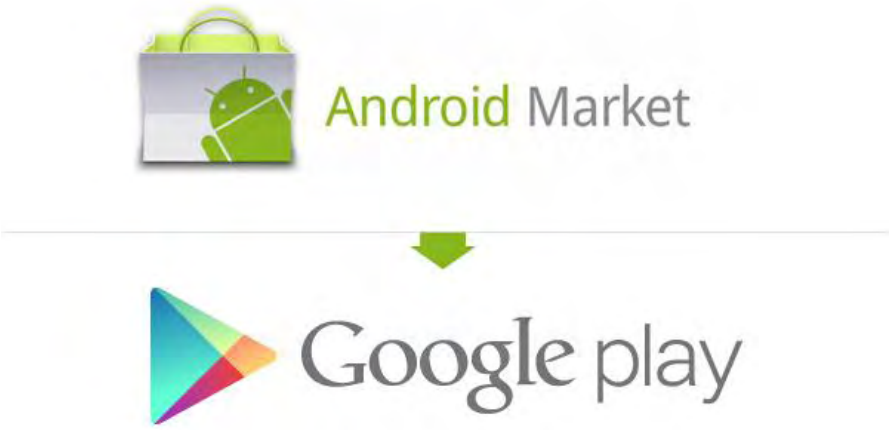
Οι Rubin, Miner, Sears και White δημιουργήσαν το Android, το 2003 στην Καλιφόρνια. Η δυνατότητα εντοπισμού και αναγνώρισης της τοποθεσίας του ιδιοκτήτη ενός έξυπνου τηλεφώνου οποιαδήποτε στιγμή και αν χρειαστεί ήταν η ανάγκη-αφορμή που οδηγησε στην δημιουργία του λειτουργικού αυτού. Στην αρχή δούλευαν κρυφά όσπου το 2005, ο Andy Rubin αποφάσισε να συνεργαστεί με την Google ώστε να χρησιμοποιηθεί ως η κύρια μηχανη αναζήτησης για το T-Mobile Sidekick, μια φερέλπιδα συσκευή κινητού, την οποία είχε αναπτύξει με την ομάδα συνεργατών του. Εν συνεχεία, ζήτησε να συναντηθεί με τον Larry Page, ο οποίος είναι ο ένας από τους δύο ιδρυτές της Google. Σε αυτήν τη συνάντηση ο Rubin παρουσίασε το Android ως ένα εν δυνάμει παγκόσμιο ανοικτό λειτουργικό σύστημα που θα άλλαζε για πάντα τον τρόπο που διαντιδρούνε οι χρήστες με το κινητό σας, τονίζοντας, ταυτόχρονα, τη σταθερή υπεροχή που παρατηρείται στις συνήθειες του αγοραστικού κοινού των κινητών τηλεφώνων, σε αντιδιαστολή με τις πωλήσεις ηλεκτρονικών υπολογιστών.
Την ίδια στιγμή, ο Page δεν ήθελε να γίνει απλώς ο υποστηρικτής του Android, ήθελε να γίνει ο ιδιοκτήτης του. Ο Andy βρήκε «την καλή», την ώρα που ένας ισχυρός παίκτης εμφανίστηκε στο προσκήνιο και έθεσε έτσι τους όρους του ανταγωνισμού σε άλλο επίπεδο. Ο καινούργιος «παίκτης» δεν είναι άλλος από εκείνον που τελικά λάνσαρε το καλοκαίρι του 2005, το iPhone της Apple.

Ο επιχειρηματικός-τεχνολογικός κόσμος περίμενε πως η Google θα απαντούσε με ένα gPhone, αλλά αυτό δεν έγινε, διότι έγινε κάτι άλλο, πολύ σημαντικότερο. Το Φθινόπωρο του 2005 ανακοινώνεται ότι 34 εταιρίες, όπως η Texas Instruments, η Intel, η T-Mobile και η Sprint Nextel, ενώνουν τις δυνάμεις τους με την Google για τη δημιουργία μιας πλατφόρμας ανοιχτού κώδικα που θα έχει ενσωματωμένο το λογισμικό Linux και θα εκπροσωπείται από μια νέα συστάδα εταιριών που θα καλείται Open Handset Alliance.

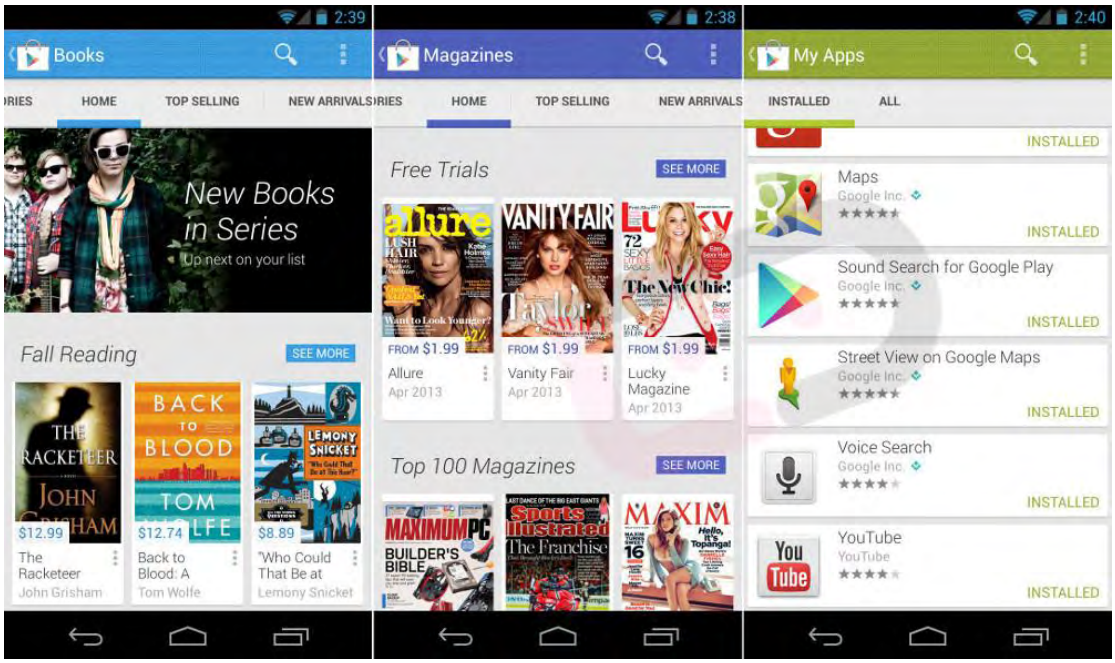
Στο χορό δεν άργησαν να μπουν και άλλες εταιρείες, όπως η HTC, η Motorola και η LG, ανακοινώνοντας την πρόθεσή τους να δώσουν προς πώληση στην αγορά smartphones με λειτουργικό σύστημα Android σε διάφορα σχήματα και μεγέθη, με τα οποία θα μπορεί να έχει ο χρήστης να ενσωματώνει στο κινητό του πλήθος εφαρμογών.

Στις 28 Αύγουστου 2008 η Google ανακοινώσε το Android Market (όπου από 6 Μάρτιου 2012 ονομάζετε Google Play) και σταδιακά άρχισε να υποστηρίζει εφαρμογές επί πληρωμή σε Οι Android συσκευές

άρχιζαν να γίνονται γνωστές και να αρέσουν στο κοινό, με αποτέλεσμα όλο και περισσότεροι διαφορές χώρες ξεχωριστά. κατασκευαστές να στραφούν προς την πλατφόρμα της Google.



Εικόνα 3. Το λογότυπο του Android Market και του Google Play



Εικόνα 4. Screenshot από το Google Play

2.3 Εξέλιξη Android

Η ιστορία του λειτουργικού συστήματος Android ξεκινά τον Νοέμβριο του 2007 με την πρώτη δοκιμαστική (beta) έκδοσή του. Η πρώτη εμπορική έκδοση του Android κυκλοφόρησε έναν χρόνο και έναν μήνα αργότερα, τον Σεπτέμβριο του 2009. Έκτοτε το Android έχει περάσει από πολλές εκδόσεις οι οποίες έχουν διορθώσει ατέλειές του και έχουν προσθέσει μια πληθώρα νέων χαρακτηριστικών. Στο παρόν κεφάλαιο θα κάνουμε μια σύντομη ανασκόπηση στις διάφορες εμπορικές εκδόσεις του λειτουργικού συστήματος καθώς και των βασικών χαρακτηριστικών που κάθε έκδοση εισήγαγε.

2.3.1 Android 1.0 και Android 1.1

Το Android 1.0 ήταν, όπως αναφέραμε, η πρώτη εμπορική έκδοση του λειτουργικού συστή- ματος και κυκλοφόρησε στις 23 Σεπτεμβρίου του 2008. Η πρώτη συσκευή που κυκλοφόρησε με λειτουργικό σύστημα Android ήταν της HTC το μοντέλο Dream. Τα χαρακτηριστικά που εισήγαγε αυτή η έκδοση συνοψίζονται στον παρακάτω πίνακα.

ΧΑΡΑΚΤΗΡΙΣΤΙΚΟ	ΠΕΡΙΓΡΑΦΗ
Android Market	Εισαγωγή της εφαρμογής του Android Market μέσω της οποίας ήταν δυνατή η αγορά νέων εφαρμογών καθώς και η αναβάθμιση ήδη υπαρχόντων
Android Web Browser	Εισαγωγή ενός web browser που μπορεί να απεικονίσει HTML και XHTML σελίδες
Κάμερα	Υποστήριξη για χρήση κάμερας αλλά έλλειψη επιλογών αλλαγής ανάλυσης, ισορροπίας λευκού κ.λπ
Φάκελοι	Υποστήριξη χρήσης φακέλων για την καλύτερη οργάνωση των εφαρμογών που βρίσκονταν στην επιφάνεια εργασίας
Email	Υποστήριξη σύνδεσης σε εξυπηρετητές email με πρωτόκολλα POP3, IMAP4 και SMTF
Google Mail	Υποστήριξη συγχρονισμού των email από τον λογαριασμό της Google με χρήση της εφαρμογής Gmail
Google Contacts	Υποστήριξη συγχρονισμού των επαφών στον λογαριασμό Google με την εφαρμογή People στην κινητή συσκευή

Google Calendar	Υποστήριξη συγχρονισμού των γεγονότων στον λογαριασμό Google με την εφαρμογή Calendar στην συσκευή
Google Maps	Υποστήριξη προβολής χαρτών της υπηρεσίας Google Maps και λήψης πληροφοριών πλοήγησης με χρήση GPS
Google Sync	Υποστήριξη συγχρονισμού των email, των επαφών και των γεγονότων από στον λογαριασμό Google στις αντίστοιχες εφαρμογές
Google Search	Υποστήριξη αναζήτησης στην συσκευή για εφαρμογές, επαφές ημερολόγια κ.λπ
Google Talk	Υποστήριξη της εφαρμογής Google Talk για ανταλλαγή μηνυμάτων μέσω του διαδικτύου
SMS, MMS	Υποστήριξη αποστολής SMS και MMS
Αναπαραγωγή Πολυμέσων	Δυνατότητα αναπαραγωγής πολυμεσικού περιεχομένου από διάφορες πηγές (αρχεία, διαδίκτυο) αλλά με έλλειψη υποστήριξης αναπαραγωγής βίντεο μέσω Bluetooth
Notifications Bar	Εισαγωγή της μπάρας ενημερώσεων του Android με δυνα- τότητα ρύθμισης ήχου, δόνησης και του LED ενημέρωσης
Voice Dialer	Εισαγωγή δυνατότητας κλήσης τηλεφώνου χωρίς να είναι αναγκαία η πληκτρολόγηση του αριθμού
Wallpapers	Δυνατότητα χρήσης οποιασδήποτε φωτογραφίας για φόντο της επιφάνειας εργασίας της συσκευής
YouTube	Εφαρμογή αναπαραγωγής βίντεο από την υπηρεσία YouTube
Wi-Fi & Bluetooth	Δυνατότητα σύνδεσης με Wi-Fi και Bluetooth

Η επόμενη έκδοση του Android ήταν η 1.1 και κυκλοφόρησε τον Φεβρουάριο του 2009, αρχικά μόνο για το HTC Dream. Η νέα αυτή έκδοση διόρθωσε σφάλματα που είχαν βρεθεί στην προηγούμενη έκδοση και εισήγαγε κάποια νέα χαρακτηριστικά. Τα νέα αυτά χαρακτηριστικά φαίνονται στον παρακάτω πίνακα.

ΧΑΡΑΚΤΗΡΙΣΤΙΚΟ	ΠΕΡΙΓΡΑΦΗ
Google Maps Enhancement	Παροχή επιπλέον πληροφοριών καθώς και κριτικών όταν κάποιος αναζητά επιχειρήσεις στην εφαρμογή με του χάρτες
In Call Enhancements	Αύξηση του χρόνου σβησίματος της οθόνης όταν γίνεται χρήση του μεγαφώνου και εισαγωγή δυνατότητας

	εμφάνι- σης και απόκρυψης του πληκτρολογίου
Mail Enhancements	Δυνατότητα αποθήκευσης στα mail αρχείων

2.3.2 Android 1.5 Cupcake (API Level: 3)

Την κυκλοφορία του Android 1.1 ακολούθησε, τρείς μήνες αργότερα, το Android 1.5 με την κωδική ονομασία Cupcake. Έκτοτε, όλες οι επόμενες εκδόσεις έφεραν σαν κωδικές ονομασίες ονόματα γλυκών. Η κυκλοφορία του Android Cupcake έγινε στις 30 Απριλίου του 2009 και ήταν βασισμένο στον πυρήνα του Linux 2.6.27. Τα νέα χαρακτηριστικά του λειτουργικού συστήματος φαίνονται παρακάτω.

ΧΑΡΑΚΤΗΡΙΣΤΙΚΟ	ΠΕΡΙΓΡΑΦΗ
Πληκτρολόγιο	Υποστήριξη πληκτρολογίων από άλλους κατασκευαστές με δυνατότητα πρόβλεψης λέξεων κατά την πληκτρολόγηση και δημιουργία λεξικού με λέξεις που ορίζει ο χρήστης
Widgets	Εισαγωγή των Widgets, μικρογραφιών μιας εφαρμογής που μπορούν να ενσωματωθούν στην επιφάνεια εργασίας και να ανανεώνονται σε τακτά χρονικά διαστήματα
Video Codecs	Δυνατότητα εγγραφής και αναπαραγωγής βίντεο στα φορμά MPEG-4 και 3GP
Auto-pairing & Stereo over Bluetooth	Αυτόματη ανίχνευση και σύνδεση με Bluetooth ακουστικά ή ηχεία και δυνατότητα αναπαραγωγής στέρεο ήχου
Copy & Paste in Browser	Εισαγωγή δυνατότητας copy και paste στον browser του Android
Εικόνες στις αγαπημένες επαφές	Δυνατότητα επιλογής εικόνων του χρήστη για εμφάνιση στις επαφές που βρίσκονται στα αγαπημένα
Λίστα Κλήσεων	Αναγραφή ημερομηνίας και ώρας στις καταχωρήσεις της λίστας και δυνατότητα επίσκεψης της επαφής στην οποία αναφέρεται η συγκεκριμένη καταχώρηση
	Δυνατότητα μετάβασης από την μια

Animated Transitions	οθόνη της εφαρμογής στην άλλη με χρήση εφέ (fade in, fade out κ.λπ)
Auto-Rotation	Αυτόματη προσαρμογή της οθόνης ανάλογα με την κλίση της συσκευής
Animation Εκκίνησης	Εισαγωγή του animation κατά την εκκίνηση του λειτουργικού
Μεταφόρτωση στο YouTube	Δυνατότητα μεταφόρτωσης βίντεο κατευθείαν στο YouTube
Μεταφόρτωση στο Picasa	Δυνατότητα μεταφόρτωσης εικόνων κατευθείαν στην υπηρεσία Picasa

2.3.3 Android 1.6 Donut (API Level: 4)

Η επόμενη έκδοση του Android ήταν η 1.6 και είχε κωδική ονομασία Donut. Ήταν βασισμένη στον πυρήνα του Linux 2.6.29 και είχε τα χαρακτηριστικά που φαίνονται παρακάτω.

ΧΑΡΑΚΤΗΡΙΣΤΙΚΟ	ΠΕΡΙΓΡΑΦΗ
Δυνατότητα Αναζήτησης	Βελτίωση της φωνητικής και μη αναζήτησης ώστε να περιλαμβάνει το ιστορικό των επισκέψεων στο διαδίκτυο, τις επαφές και το διαδίκτυο (με χρήση του google.com)
Περιεχόμενα Αναζήτησης	Παρέχεται πλέον η δυνατότητα στους προγραμματιστές να περιλαμβάνουν το περιεχόμενο των εφαρμογών τους στην δυνατότητα αναζήτησης του Android
Πολυγλωσσική Σύνθεση Ομιλίας	Δυνατότητα σύνθεσης ομιλίας σε διαφορετικές γλώσσες ώστε κάθε συσκευή Android να μπορεί να διαβάσει μια πρόταση ή ένα κείμενο
Android Market	Ευκολότερη αναζήτηση εφαρμογών στην αγορά της Google και προβολή στιγμιότυπων από την εφαρμογή για καλύτερη επιλογή από τους χρήστες
Gallery, Camera, Camcorder	Βελτίωση των επιδόσεων τον εφαρμογών πολυμέσων που είχαν πρόσβαση σε περιφερειακά όπως η

	κάμερα και οι κάρτες μνήμης της συσκευής
Gallery	Δυνατότητα επιλογής περισσότερων τις μιας φωτογραφιών για διαγραφή
Τεχνολογίες Συνδεσιμότητας	Αναβάθμιση των τεχνολογιών που χρησιμοποιούνταν για CDMA/EVDO, 802.1x και VPN
Αναλύσεις Οθόνης	Υποστήριξη για οθόνες ανάλυσης WVGA

2.3.4 Android 2.0/2.1 Eclair (API Level: 7)

Η επόμενη έκδοση του Android που κυκλοφόρησε ήταν η 2.0 βασισμένη στην ίδια έκδοση του πυρήνα του Linux που βασίστηκε και η 1.6 (την 2.6.29) και είχε τα παρακάτω χαρακτηριστικά.

ΧΑΡΑΚΤΗΡΙΣΤΙΚΟ	ΠΕΡΙΓΡΑΦΗ
Γενικευμένος Μηχανισμός Συγχρονισμού	Πλέον ο χρήστης μπορεί να εισάγει πολλαπλούς λογαριασμούς από διάφορες υπηρεσίες και οι αντίστοιχες εφαρμογές μπορούν να χρησιμοποιούν τα στοιχεία αυτά για να συγχρονίζουν το περιεχόμενό τους
Exchange Mail Server	Υποστήριξη του Microsoft Exchange Email Server για συγχρονισμό email
Αναβάθμιση Bluetooth	Υποστήριξη του πρωτοκόλλου Bluetooth 2.0
People App	Εισαγωγή της δυνατότητα να επιλέξει ο χρήστης την κλήση, την αποστολή email ή την αποστολή SMS σε μια επαφή απλά πατώντας την εικόνα της
Αναζήτηση στα SMS & MMS	Δυνατότητα αναζήτησης στα μηνύματα του χρήστη και αυτόματη διαγραφή μηνυμάτων που χρονολογικά περνούν κάποιο καθορισμένο όριο
Camera	Υποστήριξη flash, ψηφιακό ζουμ, scene mode λειτουργίας, ρύθμιση ισορροπίας λευκού, εισαγωγή εφέ χρώματος και δυνατότητα macro focus
Εικονικό Πληκτρολόγιο	Βελτίωση της ταχύτητας δακτυλογράφησης στο πληκτρολόγιο με χρήση έξυπνου λεξικού που μαθαίνει από

	τις πληκτρολογήσεις του χρήστη
Android Browser	Βελτίωση της διεπαφής του περιηγητή και εισαγωγή υποστήριξης HTML5 και double-tap ζουμ
Calendar	Βελτίωση της εμφάνισης της ατζέντας και δυνατότητα πρό- σκλησης επιπλέων ατόμων σε κάποιο γεγονός
Βελτιώσεις Βασιζόμενες στο Υλικό	Βελτιώσεις του λειτουργικού για επίτευξη καλύτερων επιδόσεων και ανανέωση της διεπαφής χρήστη
Αναλύσεις Οθόνης	Υποστήριξη περισσότερων μεγεθών και αναλύσεων οθονών με καλύτερο συντελεστή αντίθεσης
Google Maps	Αναβάθμιση του Google Maps στην έκδοση 3.1.2
MotionEvent class (SDK)	Η κλάση MotionEvent αναβαθμίστηκε ώστε να αναγνωρίζει και πολλαπλά αγγίγματα (multitouch)
Live Wallpapers	Κίνηση του φόντου της επιφάνειας εργασίας καθώς ο χρήστης αλλάζει οθόνες

2.3.5 Android 2.2 Froyo (API Level: 8)

Η έκδοση 2.2 του Android είναι από τις ποιο δημοφιλείς εκδόσεις του και είναι, μέχρι και σήμερα, η δεύτερη πιο διαδεδομένη. Για τον λόγο αυτό από δω και στο εξής θα αναφερόμαστε στα χαρακτηριστικά της κάθε έκδοσης χωριστά για αυτά που αφορούν τους τελικούς χρήστες και χωριστά σε αυτά που αφορούν στις προγραμματιστικές δυνατότητες που προσφέρονται. Έτσι, η έκδοση Froyo του Android έχει τα εξής χαρακτηριστικά.

ΧΑΡΑΚΤΗΡΙΣΤΙΚΟ	ΠΕΡΙΓΡΑΦΗ
Βελτιωμένη διαχείριση μνήμης και καλύτερες επιδόσεις	Ο διερμηνευτής Dalvik έγινε πέντε φορές πιο γρήγορος συγκρινόμενος με τον διερμηνευτή της έκδοσης 2.1 του Android, η μηχανή V8 του Chrome βοηθάει τον browser να φορτώνει γρηγορότερα σελίδες με JavaScript περιεχόμενο και αλλαγές στην διαχείριση μνήμης στο επίπεδο του πυρήνα αύξησαν ακόμη περισσότερο τις επιδόσεις

Προστασία Συσκευής με Κωδικό	Οι χρήστες μπορούν να κλειδώνουν την συσκευή με κωδικό για να αποτρέψουν την πρόσβαση τρίτων σε προσωπικά δεδομένα
Remote Wipe	Δυνατότητα επαναφοράς της συσκευής στην εργοστασιακή της κατάσταση και διαγραφή όλων των προσωπικών δεδομένων σε περίπτωση απώλειας ή κλοπής της
Framework Πολυμέσων	Το Stagefright Media Framework παρέχει την δυνατότητα αναπαραγωγής βίντεο καθώς και streaming βίντεο μέσω του HTTP. Στην δεύτερη περίπτωση η αναπαραγωγή μπορεί να αρχίσει και πριν την ολοκλήρωση της λήψης του αρχείου με το πολυμεσικό περιεχόμενο
Tethering & Wi-Fi Hotspot	Με τις δυνατότητες αυτές ο χρήστης μπορεί να χρησιμοποιήσεις την κινητή συσκευή του σαν ένα σημείο σύνδεσης στο διαδίκτυο συνδεόμενος σε αυτήν με ένα απλό USB καλώδιο

2.3.6 Android 2.3 Gingerbread (API Level: 9)

Η έκδοση 2.3 του Android κυκλοφόρησε τον Δεκέμβριο του 2010 (με μια αναβάθμιση τον Φεβρουάριο του ’11 το Android 2.3.3). Στην έκδοση αυτή σχεδιάστηκε από την αρχή το εικονικό πληκτρολόγιο, βελτιώθηκαν οι δυνατότητες πλοήγησης και έγιναν βήματα προς την καλύτερη διαχείριση ενέργειας. Παρακάτω βλέπουμε τα χαρακτηριστικά αυτής της έκδοσης.

ΧΑΡΑΚΤΗΡΙΣΤΙΚΟ	ΠΕΡΙΓΡΑΦΗ
Διαχείριση ενέργειας	Το Android έχει το δικαίωμα να τερματίσει οποιαδήποτε εφαρμογή τρέχει στο παρασκήνιο και καταναλώνει πολύ ενέργεια ή τρέχει στο προσκήνιο για περισσότερο χρόνο του κανονικού (συνήθως πέντε δευτερόλεπτα) προκειμένου διασφαλίσει την μέγιστη διάρκεια λειτουργίας
	Η συντόμευση διαχείρισης εφαρμογών

Διαχείριση Εφαρμογών	στο μενού επιλογών της επιφάνειας εργασίας δίνει την δυνατότητα στον χρήστη να δει ποιες εφαρμογές εκτελούνται. Για κάθε εφαρμογή υπάρχουν διαθέσιμες πληροφορίες που αφορούν τον χώρο που χρησιμοποιεί στην μνήμη ή σε κάποιο αποθηκευτικό μέσο, τους περιορισμούς στους οποίους υπόκειται η εφαρμογή κα. Ο χρήστης έχει την δυνατότητα να τερματίσει την εφαρμογή
Near Field Communication (NFC)	Το πρωτόκολλο NFC είναι ένα πρότυπο ασύρματης επικοινωνίας μεταξύ δυο συσκευών ή μιας συσκευής και μίας καρτέλας, παρόμοιο με το γνωστό RFID. Το Android δίνει την δυνατότητα στην συσκευή να λειτουργίσει ως αναγνώστης πληροφορίας ή ως εγγραφέας πληροφορίας
Βελτιωμένες λειτουργίες για αντιγραφή και επικόλληση	Με ένα απλό άγγιγμα μιας λέξης αυτή επιλέγεται. Η επιλογή μπορεί να τροποποιηθεί, ώστε να περιλαμβάνει και γειτονικές λέξεις, σέρνοντας τους δείκτες που θα εμφανιστούν, στα επιθυμητά σημεία. Αγγίζοντας την επιλεγμένη περιοχή αυτή αντιγράφεται και είναι έτοιμοι για να επικολληθεί σε κάποιο άλλο σημείο πιέζοντας παρατεταμένα και επιλέγοντας επικόλληση από το εμφανιζόμενο μενού
Κλήσεις με Camera	Οι εφαρμογές έχουν πρόσβαση σε όλες τις κάμερες της συσκευής και έτσι μπορούν να τις χρησιμοποιήσουν για κλήσεις μέσω του διαδικτύου. Το Android υποστηρίζει το πρωτόκολλο SIP και έτσι αν οι χρήστες διαθέτουν λογαριασμούς σε κάποιον πάροχο αντίστοιχης υπηρεσίας μπορούν να χρησιμοποιήσουν τις κινητές συσκευές τους σαν SIP τηλέφωνα
Εφαρμογή Downloads	Με την εφαρμογή Downloads οι χρήστες μπορούν να δουν όλα τα αρχεία που έχουν λάβει στην συσκευή τους από

	email, από τον περιηγητή κ.λπ

2.3.7 Android 3.0 Honeycomb

Το Android 3.0 ήταν η πρώτη έκδοση του Android που αφορούσε αποκλειστικά tablets. Οι σημαντικότερες διαφοροποιήσεις έγιναν στο επίπεδο της διεπαφής χρήστη η οποία σχεδιάστηκε με στόχο μεγαλύτερη ευχρηστία σε μεγαλύτερες οθόνες. Από την άποψη του χρήστη οι βελτιώσεις αφορούσαν τα παρακάτω.

ΧΑΡΑΚΤΗΡΙΣΤΙΚΟ	ΠΕΡΙΓΡΑΦΗ
Διεπαφή χρήστη	Ελκυστική 3D-looking διεπαφή χρήστη πλήρως ρυθμιζόμενη ως προς το περιεχόμενό της
Επανασχεδιασμένο πληκτρολόγιο	Σχεδιασμός εκ νέου του πληκτρολογίου λόγο τις μεγαλύτερης διαθέσιμης επιφάνειας στα tablets
Μπάρα συστήματος	Γρήγορη επιλογή συχνά χρησιμοποιούμενων λειτουργιών διαθέσιμες πάντα το κάτω μέρος της οθόνης
Μπάρα εφαρμογής (Action Bar)	Μπάρα με επιλογές ειδικά για τρέχουσα εφαρμογή, που βρίσκεται στο πάνω μέρος της οθόνης
Multitasking	Δυνατότητα επισκόπησης των ταυτόχρονα εκτελούμενων εφαρμογών και μετάβαση σε οποιαδήποτε από αυτές ο χρή στης επιθυμεί
Επιλογές Συνδεσιμότητας	Δυνατότητα σύνδεσης του tablet με εξωτερικές συσκευές όπως πληκτρολόγιο με χρήση είτε USB, είτε Bluetooth
Photo Transfer Protocol (PTP) & Media Transfer Protocol (MTP)	Με χρήση των πρωτοκόλλων αυτών γίνεται δυνατή η μεταφορά πολυμέσων από συσκευή σε συσκευή με μεγαλύτερη ευκολία χρησιμοποιώντας εφαρμογές που τα υποστηρίζουν
Bluetooth Tethering	Δυνατότητα χρήσης της σύνδεσης της κινητής συσκευής από κάποιον προσωπικό υπολογιστή που συνδέετε σε αυτήν μέσω bluetooth
Περιηγητής	Εισαγωγή tabs, όπως και στους προσωπικούς υπολογιστές, αφού πλέον υπάρχει ο διαθέσιμος χώρος. Παρέχει υποστίριξη JavaScript και δυνατότητες συγχρονισμού με τον λογα-ριασμό Google του χρήστη
	Επανασχεδιασμένη διεπαφή χρήστη για

Camera	τον έλεγχο της κάμερας με περισσότερες επιλογές άμεσα διαθέσιμες και την δυνατότητα λήψης time-lapse βίντεο
Επαφές	Λόγο και πάλι του μεγαλύτερου διαθέσιμου μεγέθους οι επισκόπηση των επαφών και η αναζήτησή τους είναι ευκολότερες από ποτέ χάρη στην διεπαφή δυο στηλών (αριστερά οι επαφές και δεξιά οι λεπτομέρειες της τρέχουσας επιλεγμένης)

2.3.8 Android 4.0 Ice cream sandwich (API Level: 15)

Η έκδοση του Android ονόματι Ice Cream Sandwich ήρθε για να ενώσει του δυο κόσμους συσκευών. Αυτόν των tablets και αυτόν των smart phones. Η έκδοση αυτή κυκλοφόρησε στις 19 Οκτωβρίου 2011 και ήταν βασισμένη στον πυρήνα του Linux 3.0.1. Σε αυτήν την έκδοση έγιναν πολλές βελτιώσεις σε είδη υπάρχουσες εφαρμογές και χαρακτηριστικά του συστήματος όπως ολοκλήρωση των κοινωνικών δικτύων στην εφαρμογή των επαφών, επιτάχυνση του UI από το υλικό και εγγραφή 1080p βίντεο. Επιπλέον διορθώθηκαν αρκετά σφάλματα στον κώδικα και έγιναν πολλές βελτιώσεις που διασφάλισαν την σταθερότητα της έκδοσης αυτής.

2.3.9 Android 4.1 Jelly Bean (API Level: 16)

Παρουσιάστηκε στις 9/7/2012. Κάποια χαρακτηριστικά της έκδοσης αυτής είναι:

- Ανανεωμένο σύστημα ειδοποιήσεων
- δυνατότητα χρήσης εξωτερικής συσκευής USB ήχου
- Βελτιωμένη φωνητική αναζήτηση
- Δυνατότητα χρήσης της υπηρεσίας Google Wallet
- OpenGL ES 3.0 υποστήριξη, που βελτιώνει τα γραφικά ενός παιχνιδιού

2.3.10 Android 4.4 Kit Kat (API Level: 19)

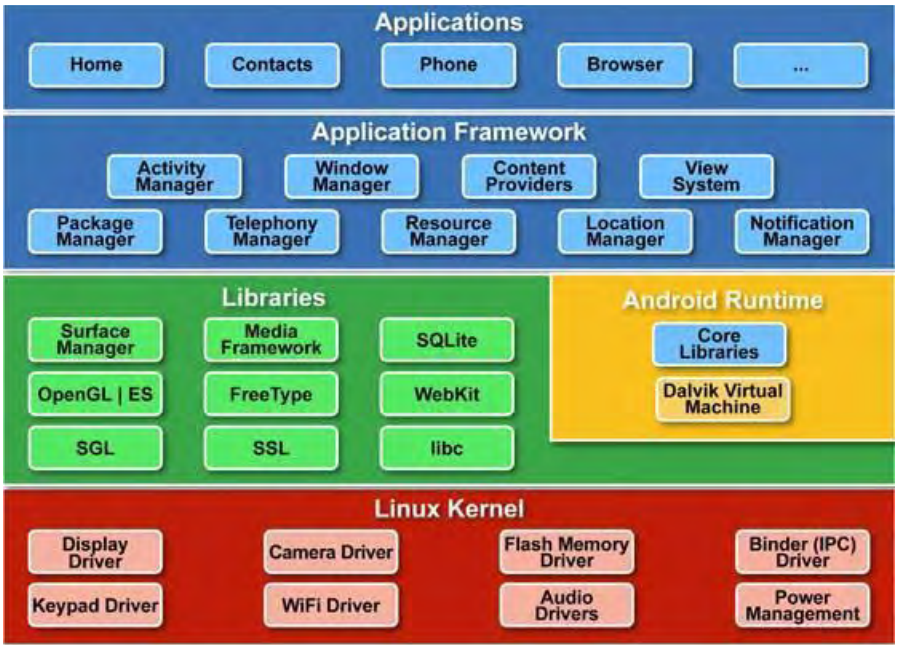
Παρουσιάστηκε στις 31/10/2013. Κάποια χαρακτηριστικά της έκδοσης αυτής είναι:

- Αρκεί να πεις “Ok Google” για να ξεκινήσεις τη φωνητική λειτουργία
- Κατά την αναπαραγωγή μουσικής ή videos, στο κλείδωμα οθόνης εμφανίζει κάποιο είδος “τέχνης”
- Γρηγορότερο multitasking
- Με το Hangouts τοποθετούνται όλα τα μηνύματα σε ένα μέρος
- Υποστηρίζεται η ασύρματη εκτύπωση για τους εκτυπωτές που είναι συνδεδεμένοι στο Google Cloud Print

Εικόνα 5. Η εξέλιξη των εκδόσεων Android

2.4 Αρχιτεκτονική Android

Το λειτουργικό σύστημα Android είναι μια στοίβα στοιχείων λογισμικού που σε γενικές γραμμές χωρίζονται σε πέντε ενότητες και τέσσερα κύρια στρώματα, όπως φαίνεται παρακάτω στο διάγραμμα της αρχιτεκτονικής.



Εικόνα 6. Διάγραμμα αρχιτεκτονικής Android

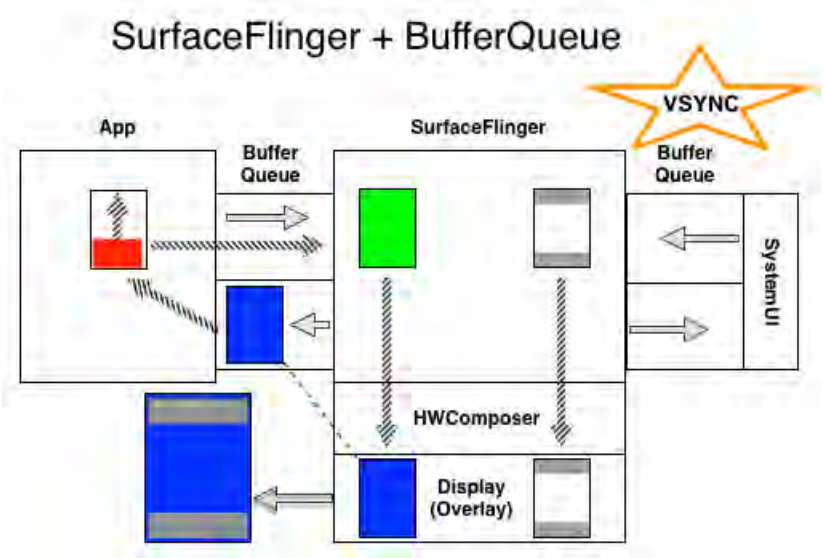
Το Android αποτελείται από τα ακόλουθα στρώματα:

- εφαρμογές (γραμμένες σε Java, σε εκτέλεση Dalvik)
- πλαίσιο των υπηρεσιών και βιβλιοθήκες (γραμμένα κυρίως σε Java)
    - οι εφαρμογές και οι περισσότεροι κωδικοί πλαισίου εκτελούνται σε μια εικονική μηχανή
- βιβλιοθήκες και υπηρεσίες (γραμμένες σε C ή C ++)
- ο πυρήνας του Linux, ο οποίος περιλαμβάνει:
    - τους οδηγούς για το υλικό, τη δικτύωση, την πρόσβαση στο σύστημα αρχείων και στο εσωτερικό των διαδικασιών επικοινωνίας

Η αρχιτεκτονική του Android περιλαμβάνει τα εξής επίπεδα, πηγαίνοντας από το υψηλότερο στο χαμηλότερο:

- **Επίπεδο Εφαρμογών (Applications):** Το Android εξαρχής περιέχει ένα σύνολο από βασικές εφαρμογές που περιλαμβάνουν ένα email client, ένα πρόγραμμα για SMS μηνύματα, ημερολόγιο, χάρτες (Google Maps), περιηγητή ιστού, πρόγραμμα για δομημένη αποθήκευση των επαφών και άλλα. Όλες οι εφαρμογές είναι γραμμένες στη γλώσσα προγραμματισμού Java.
- **Επίπεδο Πλαισίου Εφαρμογών (Applications Framework):** Ακολουθώντας μια ανοικτή πλατφόρμα ανάπτυξης, το Android προσφέρει στους προγραμματιστές τη δυνατότητα να κατασκευάσουν πλούσιες και καινοτόμες εφαρμογές. Οι προγραμματιστές είναι ελεύθεροι να αξιοποιήσουν πλήρως το hardware της συσκευής, να έχουν πρόσβαση σε υπηρεσίες εντοπισμού θέσης, να τρέξουν υπηρεσλιες στο background, να θέσουν χρονοδιακόπτες για εμφάνιση ειδοποιήσεων και πολλά άλλα. Επίσης, έχουν πλήρη πρόσβαση στο ίδιο πλαίσιο από APIs που έχουν οι βασικές εφαρμογές του Android. Η αρχιτεκτονική είναι διαμορφωμένη με τέτοιο τρόπο που κάθε εφαμογή μπορεί να χρησιμοποιήσει τις δυνατότητες μιας άλλης και επίσης με τέτοιο τρόπο που δίνει τη δυνατότητα στο χρήστη να αλλάξει τα συστατικά κάθε εφαρμογής. Κάτω από το πλαίσιο των εφαρμογών υπάρχει ένα σύστημα από υπηρεσίες και συστήματα τα οποία περιλαμβάνουν:
    - Ένα σύνολο από γραφικά στοιχεία (Views) για τη δημιουργία γραφικού περιβάλλοντος συμπεριλαμβανομένων λιστών (lists), πλεγμάτων (grids), κουτιών κειμένων (text boxes), κουμπιών (buttons) και άλλων.
    - Ένα διαχειριστή περιεχομένου (Content Manager), ο οποίος επιτρέπει στις εφαρμογές την πρόσβαση σε δεδομένα άλλων εφαρμογών ή το διαμοιρασμό των δικών τους δεδομένων με άλλες εφαρμογές.
    - Ένα διαχειριστή πόρων (Resource Manager) για την πρόσβαση στους πόρους όπως strings, εικόνες, layout files.
    - Ένα διαχειριστή ειδοποιήσεων (Notification Manager), ο οποίος επιτρέπει την προβολή ειδοποιήσεων στη μπάρα κατάστασης (status bar).
    - Ένα διαχειριστή δραστηριοτήτων (Activity Manager), ο οποίος διαχειρίζεται τον κύκλο ζωής των εφαρμογών.

- **Επίπεδο Βιβλιοθηκών (Libraries):** Το οποίο περιλαμβάνει ένα σύνολο από βιβλιοθήκες γραμμένες σε C/C ++, οι οποίες χρησιμοποιούνται από διάφορα στοιχεία του συστήματος του Android. Οι δυνατότητες που προσφέρουν αυτές οι βιβλιοθήκες είναι η προσβασιμότητα στους προγραμματιστές μέσου του επιπέδου πλαισίου εφαρμογής.
- **Επίπεδο Εκτέλεσης (Android Runtime):** Το οποίο αποτελείται από ένα σύνολο από βασικές βιβλιοθήκες και τη Dalvik Virtual Machine.
- **Πυρήνας του Linux:** Το Android βασίζεται στον πυρήνα Linux έκδοση 2.6 για βασικές υπηρεσίες συστήματος, όπως ασφάλεια, διαχείριση μνήμης, διαχείριση διεργασιών, στοίβα δικτύου, και οδηγούς συσκευών. Ο πυρήνας λειτουργεί, επίσης, ως ένα ενδιάμεσο επίπεδο αφαίρεσης μεταξύ της στοίβας λογισμικού και του υλικού.



Εικόνα 7. Λειτουργία του Android

3. Εργαλεία Ανάπτυξης

Όπως έχουμε δει μέχρι τώρα οι εφαρμογές για το λειτουργικό σύστημα Android γράφονται στην γλώσσα προγραμματισμού Java. Απαραίτητο είναι, για όποιον επιθυμεί να αναπτύξει εφαρμογές για Android, να διαθέτει λειτουργικό σύστημα Windows, Linux ή Mac OS. Επιπρόσθετα, χρειάζονται τα παρακάτω εργαλεία, στις αντίστοιχες εκδόσεις του λειτουργικού στο οποίο θα πραγματοποιηθεί η διαδικασία ανάπτυξης:

1. Java Development Kit
2. Android Studio
3. Android SDK

Όλα τα απαραίτητα εργαλεία είναι διαθέσιμα δωρεάν και η εγκατάστασή τους είναι αρκετά εύκολη και σύντομη διαδικασία (η χρονική διάρκεια της εγκατάστασης εξαρτάται σημαντικά από την ταχύτητα της σύνδεσης στο διαδίκτυο).

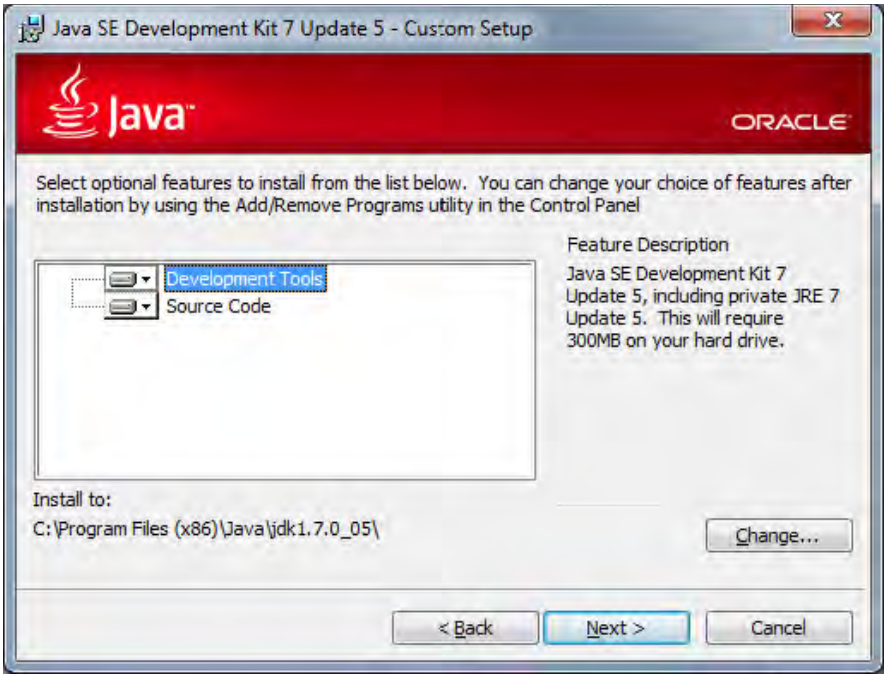
3.1 Εγκατάσταση του Java Development Kit

Για την συγγραφή οποιασδήποτε εφαρμογής Java είναι απαραίτητη η ύπαρξη του Java Development Kit (JDK). Το JDK περιλαμβάνει ένα σύνολο εργαλείων ανάπτυξης προγραμμάτων για την γλώσσα προγραμματισμού Java όπως είναι ο java (φορτωτής Java εφαρμογών), ο javac (ο μεταφραστής του Java κώδικα σε Java bytecodes), ο javah (δημιουργός C stubs για συγγραφή native μεθόδων), ο jar (ο γνωστός Java archiver) και άλλα. Το JDK είναι το υποσύνολο του Java SDK που είναι απαραίτητο για τον προγραμματισμό και την εκτέλεση εφαρμογών σε Java.Το Java SDK περιλαμβάνει επιπλέον στοιχεία όπως application servers, επιπλέον debuggers και documentation.

Για την ανάπτυξη εφαρμογών για το λειτουργικό σύστημα Android, το JDK είναι αρκετό. Η τρέχουσα έκδοσή του είναι η 7 (update 5) και το αρχείο εγκατάστασης είναι διαθέσιμο από τον ιστότοπο της Oracle (στο παρόν κεφάλαιο θα αποφύγουμε να παραθέσουμε υπερσυνδέσμους για τα διάφορα εργαλεία που θα παρουσιαστούν διότι αυτοί ενδέχεται να αλλάξουν από τον εκάστοτε πάροχο).

Για την εγκατάσταση, επιλέγουμε από τον ισότοπο το αρχείο εγκατάστασης που αντιστοιχεί στο λειτουργικό μας σύστημα, το αποθηκεύουμε και το εκτελούμε ακολουθώντας τα παρακάτω βήματα:

1. Στην πρώτη οθόνη επιλέγουμε τα κομμάτια του JDK που θέλουμε να εγκαταστήσουμε καθώς και την διαδρομή στον δίσκο στην οποία θα πραγματοποιηθεί η εγκατάσταση. Στην περίπτωσή μας οι προεπιλεγμένες τιμές είναι ικανοποιητικές.



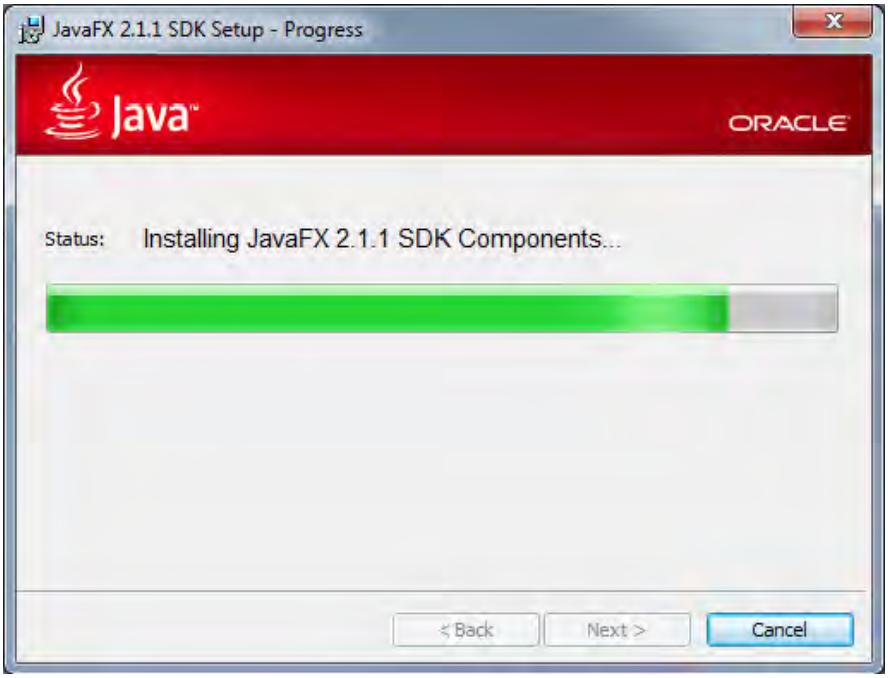
2. Αναμένουμε να ολοκληρωθεί η αντιγραφή των νέων αρχείων και να γίνει η διαγραφή των backup αρχείων

3. Μετά την ολοκλήρωση του βήματος 2 θα εμφανιστεί μήνυμα που θα ενημερώνει για την επιτυχή εγκατάσταση του JDK και θα ζητά να εγγραφούμε στην Oracle. Αγνοούμε την εγγραφή και πατάμε το κουμπί **Continue.**

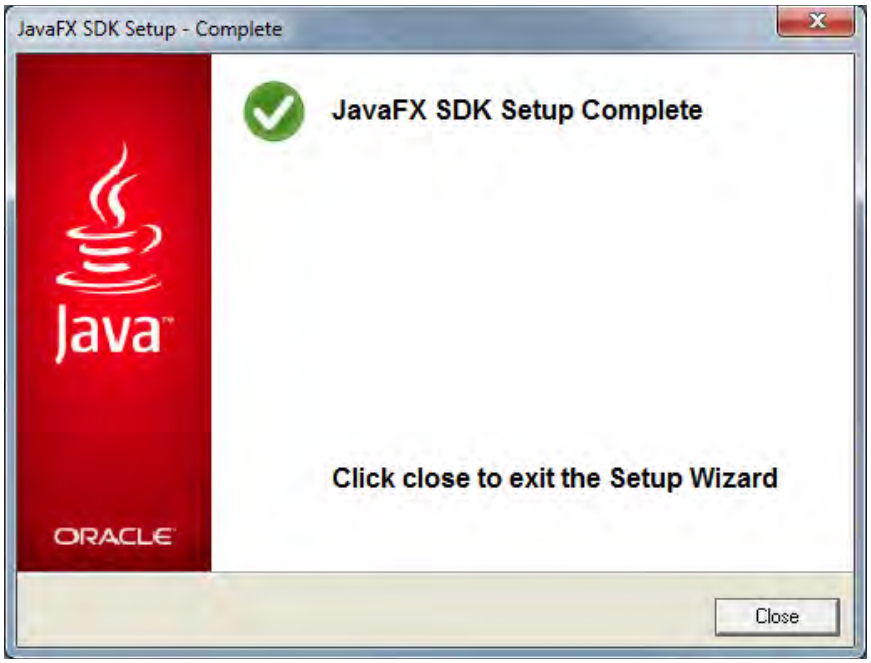
4. Στο επόμενο βήμα θα ξεκινήσει αυτόματα η εγκατάσταση του Java FX SDK. Επιλέγουμε **Next**

5. Στο επόμενο πλαίσιο διαλόγου λαμβάνουμε κάποιες πληροφορίες για τον χώρο που θα δεσμεύσει η τρέχουσα εγκατάσταση και μας δίνεται η δυνατότητα να επιλέξουμε την διαδρομή στον δίσκο που αυτή θα πραγματοποιηθεί. Επιλέγουμε **Next.**

6. Αναμένουμε την ολοκλήρωση της εγκατάστασης.



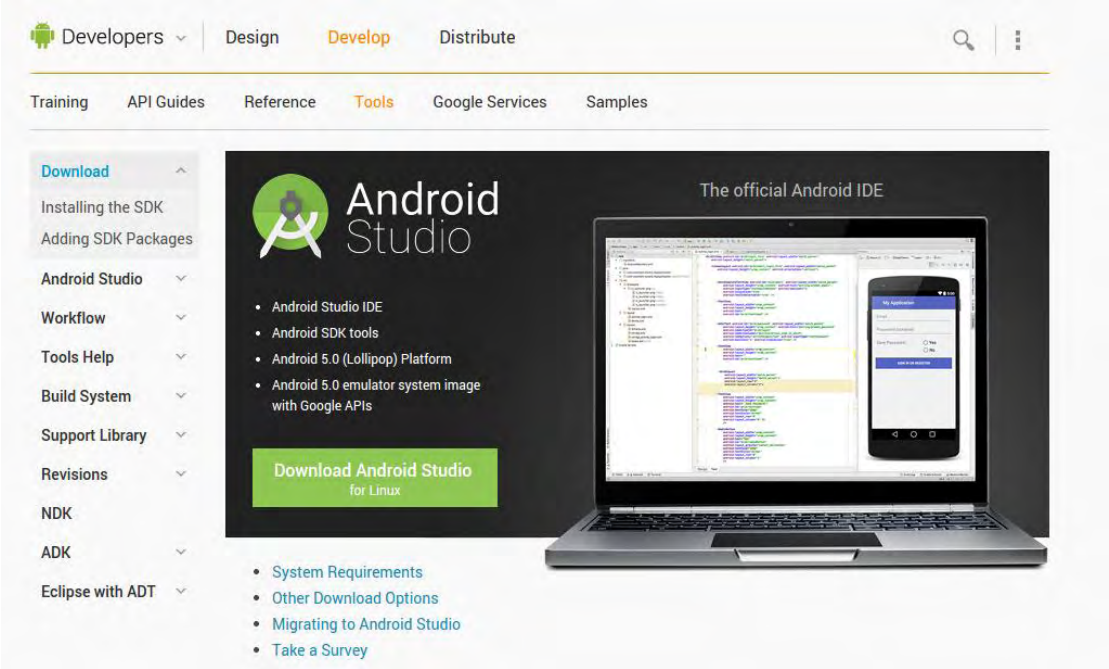
7. Μετά την ενημέρωσή μας για την επιτυχή εγκατάσταση του Java FX SDK επιλέγουμε **Close** και η εγκατάσταση του JDK έχει ολοκληρωθεί επιτυχώς.



3.2 Εγκατάσταση του Android Studio

3.2.1 Λήψη του Android Studio

Το βασικό εργαλείο που θα χρησιμοποιήσουμε για την ανάπτυξη της εφαρμογής είναι το Android Studio. Μπορείτε να το κατεβάσετε από την ακόλουθη ιστοσελίδα: http://developer.android.com/sdk/index.html Κλικάρετε το download button εάν ήδη έχει επιλεγεί σωστά το λειτουργικό σας σύστημα (πχ "Download Android Studio for Windows"), διαφορετικά επιλέξτε το "Other Download Options" και επιλέξτε το κατάλληλο package.

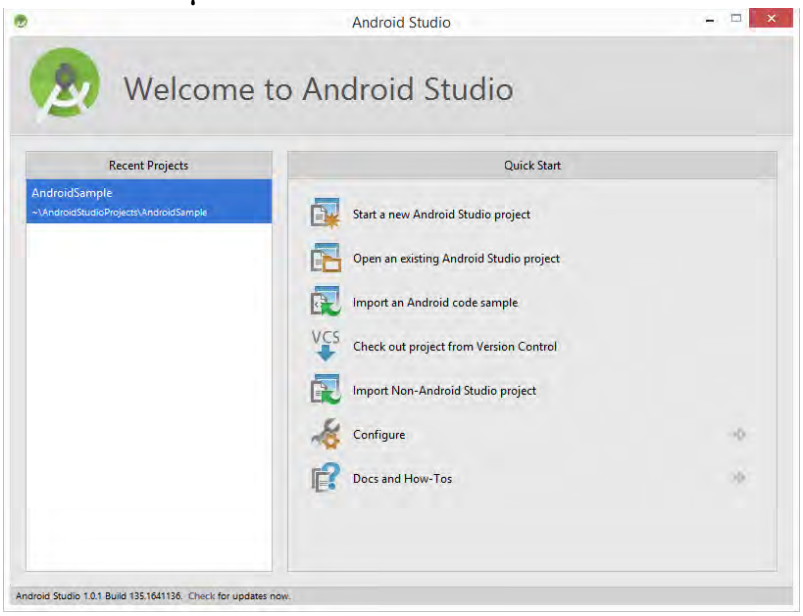


Εικόνα 8. Οθόνη λήψης Android Studio

3.2.2 Εγκατάσταση του Android Studio

Μετά τη λήψη, τα ακριβή βήματα για να εγκαταστήσετε το Android Studio διαφέρουν ανάλογα με το λειτουργικό σας σύστημα.

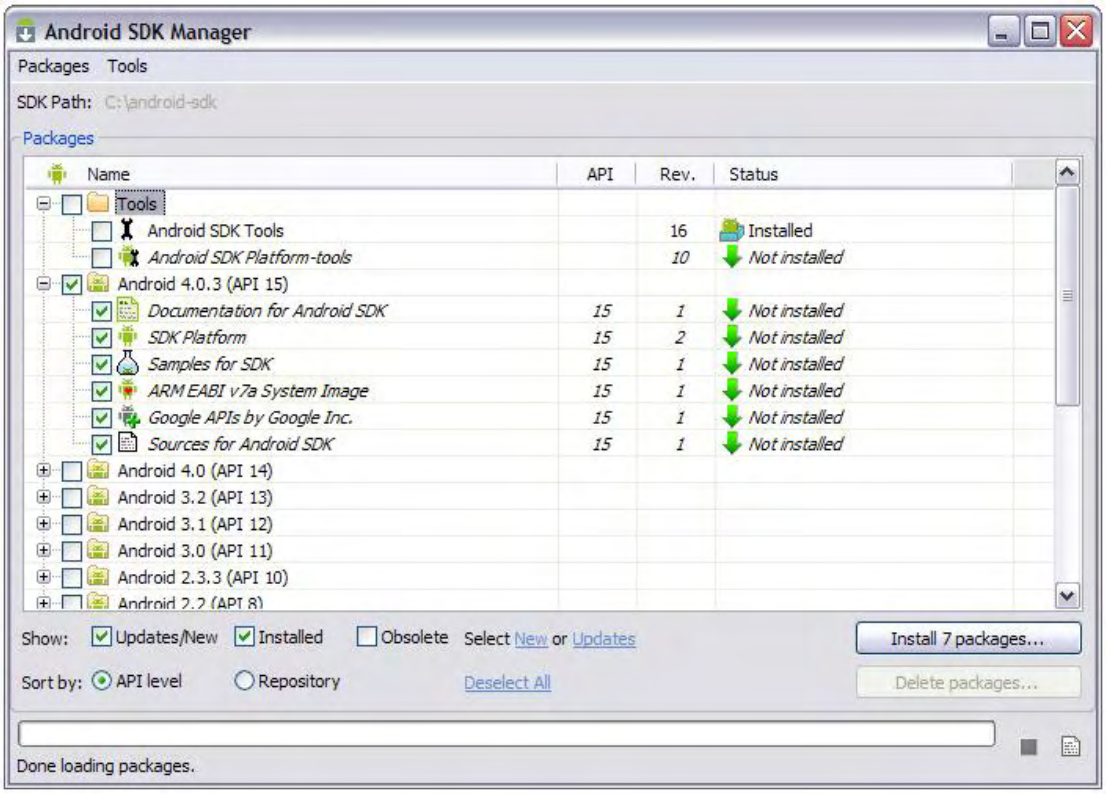
Εγκατάσταση σε Windows: Εντοπίστε το εκτελέσιμο αρχείο εγκατάστασης (που ονομάζεται Androidstudiobundle .exe) σε ένα παράθυρο του Windows Explorer και κάντε διπλό κλικ σε αυτό για να ξεκινήσει η διαδικασία εγκατάστασης. Κάντε κλικ στο κουμπί YES στο παράθυρο διαλόγου, εάν εμφανίζεται. Μόλις εμφανιστεί ο οδηγός εγκατάστασης Android Studio, προχωρήστε μέσα από τις διάφορες οθόνες για να διαμορφώσετε την εγκατάσταση έτσι ώστε να καλύπτει τις απαιτήσεις σας όσον αφορά την τοποθεσία του συστήματος αρχείων του Android Studio κτλ. Αν και δεν υπάρχουν αυστηροί κανόνες για το πού θα πρέπει να εγκατασταθεί στο σύστημα το Android Studio, θα υποθέσουμε ότι η εγκατάσταση έγινε σε ένα υποφάκελο του home folder που ονομάζεται Android-στούντιο. Μόλις οι επιλογές έχουν ρυθμιστεί, κάντε κλικ στο κουμπί Εγκατάσταση για να ξεκινήσει η διαδικασία εγκατάστασης. Στις εκδόσεις των Windows με μενού Έναρξη, το μόλις εγκατεστημένο Android Studio μπορεί να βρίσκεται σε αυτό το μενού. Στα Windows 8, το εκτελέσιμο μπορεί να καρφώθηκε στη γραμμή εργασιών. Σημειώστε ότι το εκτελέσιμο παρέχεται σε 2 εκδόσεις, 32bit (studio) και 64bit (studio64). Εάν χρησιμοποιείτε ένα σύστημα 32bit να είστε σίγουρος για να χρησιμοποιήσετε το σωστό εκτελέσιμο.



Εικόνα 9. Welcome Screen Android Studio

3.3 Εγκατάσταση Android SDK

Τα βήματα που έχουν πραγματοποιηθεί μέχρι στιγμής έχουν εγκαταστήσει την Java, το Android Studio IDE και το τρέχον σύνολο των προεπιλεγμένων πακέτων του Android SDK. Πριν προχωρήσετε, αξίζει να αφιερώσετε κάποιο χρόνο, για να δείτε ποια πακέτα είναι εγκατεστημένα και να εγκαταστήσετε τα πακέτα που λείπουν. Αυτή η εργασία μπορεί να εκτελεστεί με τη χρήση του Android SDK Manager, η οποία μπορεί να ξεκινήσει μέσα από το εργαλείο Android Studio, επιλέγοντας το Configure > SDK Manager από το Android Studio welcome dialog. Μετά το άνοιγμα, το εργαλείο SDK Manager θα εμφανιστεί, όπως στο παρακάτω σχήμα:



Εικόνα 10. SDK Manager Android Studio

Μέσα στο Android SDK Manager, βεβαιωθείτε πως είναι επιλεγμένα τα ακόλουθα πακέτα:

- Tools > Android SDK Tools
- Tools > Android SDK Platformtools
- Tools > Android SDK Buildtools
- SDK Platform (most recent version)> SDK Platform

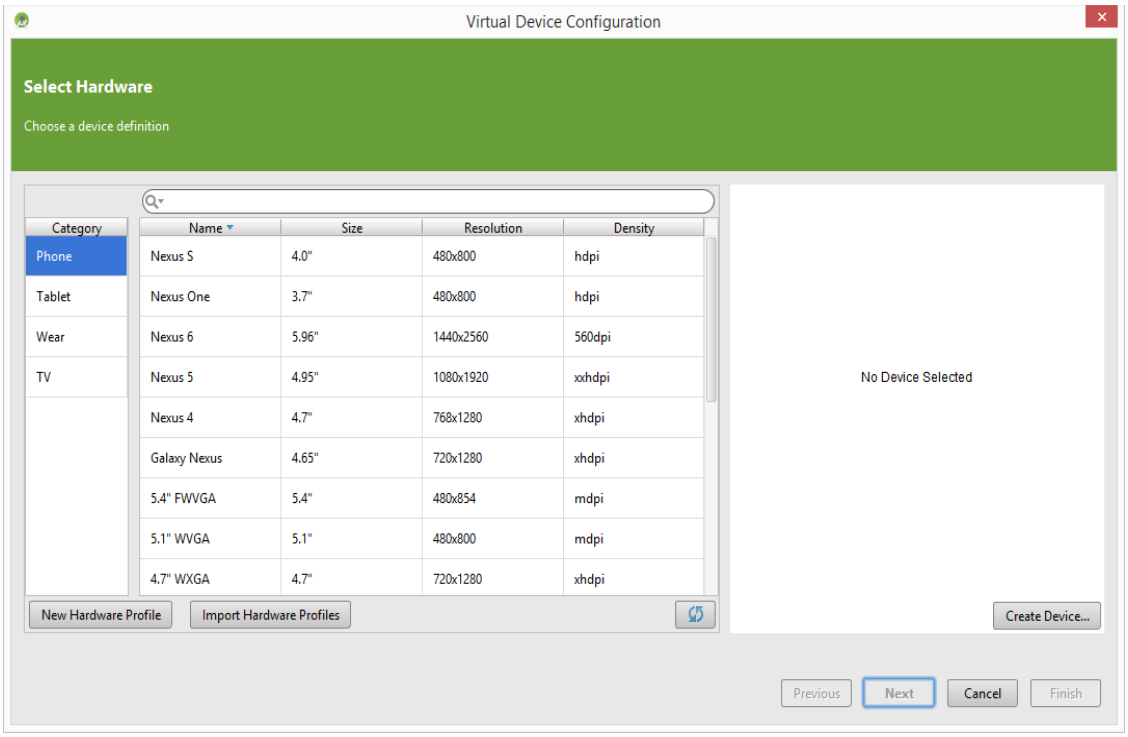
- SDK Platform (most recent version) > ARM EABI v7a System Image
- Extras > Android Support Repository
- Extras > Android Support Library
- Extras > Google Repository
- Extras > Google USB Driver (Required on Windows systems only)
- Extras > Intel x86 Emulator Accelerator (HAXM installer)

Σε περίπτωση που κάποιο από τα παραπάνω πακέτα φαίνεται ως μη αγκατεστημένο, επιλέξτε το και κάντε κλικ στο κουμπί Install Packages για να ξεκινήσει η διαδικασία εγκατάστασης. Στο παράθυρο διαλόγου που θα εμφανιστεί, αποδεχθείτε τις άδειες χρήσης και κάντε κλικ στο κουμπί install. Στη συνέχεια, το SDK Manager θα αρχίσει να κατεβάζει και να εγκαθιστά τα πακέτα που επιλέξατε. Καθώς προχωρά η εγκατάσταση, η γραμμή προόδου θα εμφανιστεί στο κάτω μέρος του παραθύρου του διαχειριστή υποδεικνύοντας την κατάσταση της εγκατάστασης

3.4 Δημιουργία εικονικής συσκευής Android

Το Android Virtual Device (AVD) είναι ένα εικονικό smartphone που τρέχει Android το οποίο μπορούμε εύκολα να εκτελέσουμε οποιαδήποτε εφαρμογή δημιουργήσουμε. Το πλεονέκτημα της AVD είναι ότι μπορούμε να δούμε πώς θα προβάλλεται το λογισμικό που δημιουργήσαμε σε διαφορετικά smartphones με Android.

Για την δημιουργία μιας τέτοιας εικονικής συσκευής πηγαίνουμε στο πρόγραμμα Android Studio στην επιλογή Tools > Android > AVD Manager. Στο παράθυρο εμφανίζονται οι επιλογές μας και διαλέγουμε τα χαρακτηριστικά του smartphone που επιθυμούμε να προσομοιώσουμε.

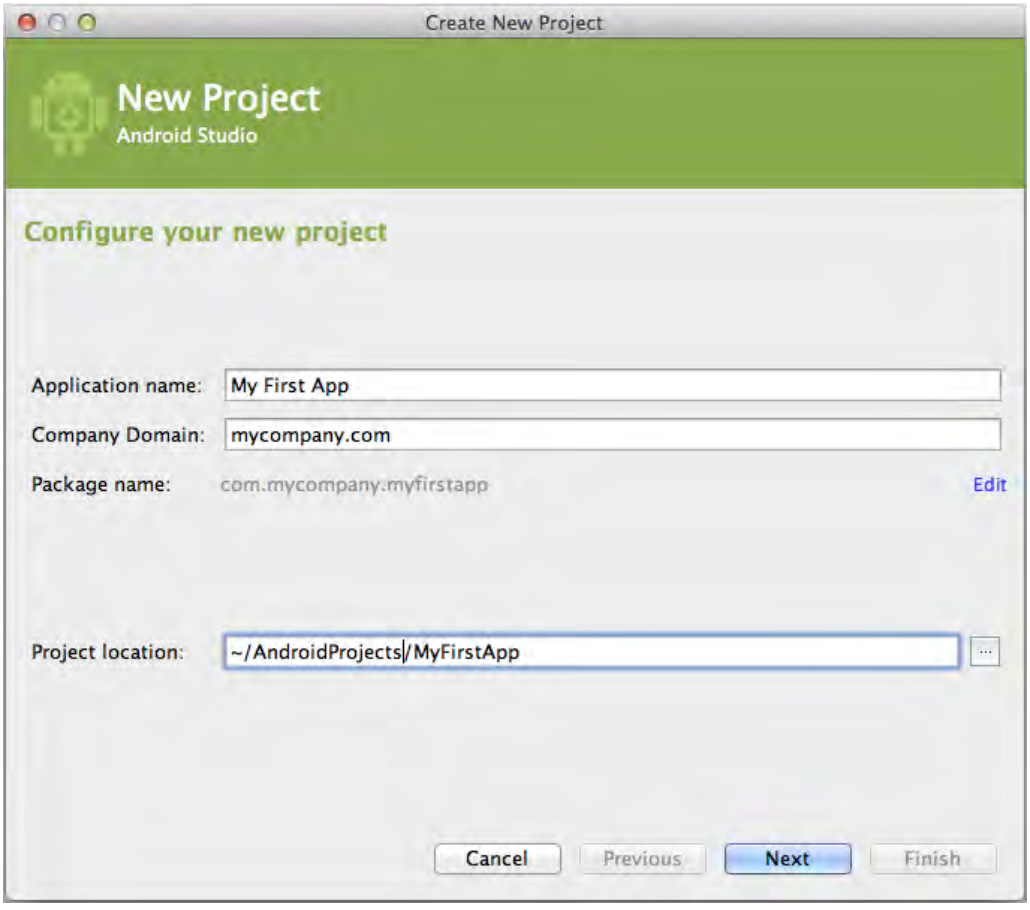


Εικόνα 11. Android Virual Device

3.5 Δημιουργία νέου Project

1. Στο Android Studio, δημιουργείστε ένα νέο Project:
   a. Αν δεν έχετε κάποιο Project ήδη ανοιχτό, στην οθόνη καλωσορίσματος, πατήστε New Project
   b. Αν έχετε κάποιο Project ανοιχτό, από το File Menu, επιλέξτε New Project.

2. Στην καρτέλα Create New Project, συμπληρώστε τα πεδία, όπως φαίνονται στην εικόνα παρακάτω, και πατήστε Next.



Εικόνα 12. Οθόνη δημιουργίας New Project

3. Στην επιλογή Select the form factors your app will run on, τσέκαρε το κουτί για Phone and Tablet.

4. Για Minimum SDK, επιλέξτε API 8: Android 2.2 (Froyo)

5. Αφήστε όλες τις υπόλοιπες επιλογές (TV, Wear και Glass) μη επιλεγμένες και πατήστε Next.

6. Στην επιλογή Add an activity to <template>, επιλέξτε Blank Activity και πατήστε Next.

7. Στην επιλογή Choose options for your new file, αλλάξτε το Activity Name σε MyActivity. Το Layout Name σε activity_my και το Title σε MyActivity. Το Menu Resource Name είναι menu_my.

8. Πατήστε το Finish κουμπί για να δημιουργήσετε το Project.

Το Android Project σας, τώρα, είναι μια βασική εφαρμογή "Hello World", που περιέχει κάποια default αρχεία.

4. Βασικά στοιχεία μιας εφαρμογής

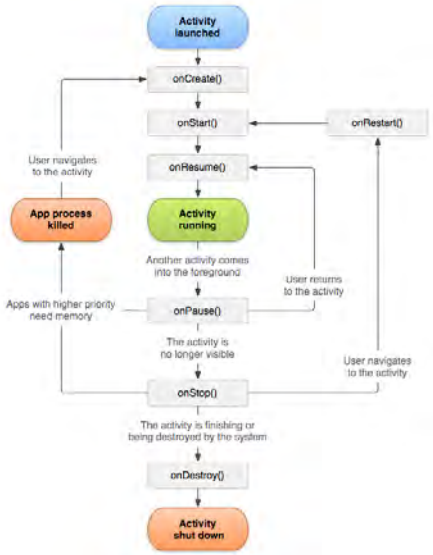
Τα δομικά στοιχεία μιας εφαρμογής Android μπορούμε να τα διακρίνουμε σε τέσσερις κατηγορίες: στις δραστηριότητες (Activities), στις υπηρεσίες (Services), στους παρόχους περιεχομένου (Content providers) και τέλος, στους καθολικούς παραλήπτες μηνυμάτων (Broadcast Receives).

4.1 Activities

Είναι το σημαντικότερο στοιχείο μιας εφαρμογής. Κάθε Activity αποτελεί ένα παράθυρο της εφαρμογής, ουσιαστικά μια διαφορετική οθόνη, και είναι αυτό που φαίνεται στο χρήστη. Εδώ φορτώνεται το γραφικό περιβάλλον της εφαρμογής.

Η πρώτη δραστηριότητα που βλέπει ο χρήστης είναι η Main Activity, από την οποία έχει πρόσβαση και σε όλες τις υπόλοιπες. Ουσιαστικά πολλές δραστηριότητες μαζί μπορούν να συνδέονται μεταξύ τους.

Όταν ο χρήστης καλέσει μια νέα οθόνη ουσιαστικά καλείται μια νέα Activity, με το πλήκτρο back μπορεί να επιστρέψει στην προηγούμενη Activity (οθόνη), δηλαδή έχουμε μια μορφή στοίβας. Δεν είναι απαραίτητο όλες οι δρατηριότητες να είναι αποθηκευμένες στη στοίβα. Ένα παράδειγμα στο οποίο είναι απαραίτητο να μην είναι αποθηκευμένη στη στοίβα είναι όταν ο χρήστης πατήσει το back και ενώ είναι στη Main Activity θα πρέπει να βγει από την εφαρμογή.



Εικόνα 13. Διάγραμμα ζωής Activity

4.2 Services

Εκτός από την Activity με την οποία έχει άμεση επαφή ο χρήστης, από πίσω κάποιο στοιχείο της εφαρμογής εκτελεί κάποιες διεργασίες. Αυτή είναι η Service, η οποία μπορεί να εκτελείται από μια διαφορετική εφαρμογή από αυτήν που αρχικά την κάλεσε, ενώ δεν έχει κάποιο user interface.

Παράδειγμα υπηρεσίας είναι η εφαρμογή πλοήγησης. Χωρίς απαραίτητα να βλέπουμε κάποιο γραφικό περιβάλλον, θέλουμε να τρέχει η εφαρμογή ενώ παράλληλα να μπορούμε να χρησιμοποιήσουμε το κινητό τηλέφωνο για μια εισερχόμενη κλήση. Στην περίπτωση αυτή, θέλουμε να τρέχει η εφαρμογή, ώστε όταν χρειαστεί να την τρέξουμε, να μη τη φορτώσουμε από την αρχή.

Δύο είναι οι βασικές λειτουργίες που έχουν άμεση ισχύ με τις Services. Η startService() και η bindService(). Η πρώτη καλείται από κάποια Activity για να ξεκινήσει η Service και συνεχίζει να εκτελείται ακόμη και όταν η Activity που την κάλεσε έχει τερματιστεί. Ένα παράδειγμα που χρησιμοποιείται η startService είναι όταν θέλουμε να κατεβάσουμε ένα αρχείο από το Google Play. Μπορεί εμείς να έχουμε κλείσει την εφαρμογή, αλλά το αρχείο συνεχίζει να κατεβαίνει. Η δεύτερη είναι «δεμένη» από μια Activity. Όταν για κάποιο λόγο τερματιστεί η Activity, τερματίζεται και η Service.

Τέλος, πρέπει να τονίσουμε ότι είναι απαραίτητο να δηλώνεται ο τερματισμός της Service, διότι συνεχίζει να τρέχει καταναλώνοντας υπολογιστικούς πόρους.

4.3 Content Providers

Οι Content Providers διαχειρίζονται αποθηκευτικούς χώρους δεδεμένων που είναι προσπελάσιμοι από περισσότερες εφαρμογές. Είναι ο μοναδικός τρόπος με τον οποίο μπορούν να αποθηκευτούν δεδομένα από αρκετές εφαρμογές. Για παράδειγμα, το ενδεχόμενο να χρησιμοποιηθούν οι επαφές ενός τηλεφώνου από μια άλλη εφαρμογή είναι πολύ πιθανό, οπότε και θα πρέπει οι επαφές να αποθηκευτούν σε κάποιο συγκεκριμένο χώρο, ώστε μελλοντικά να μπορεί μια άλλη εφαρμογή να τις επεξεργαστεί. Τα αρχεία βίνετο εικόνας και ήχου είναι, επίσης, αρχεία τα οποία πρέπει και αυτά να αποθηκευτούν σε κάποιο αντίστοιχο χώρο.

Στο πλαίσιο αυτό υπάρχει ενσωματωμένη βάση δεδομένων στο λειτουργικό σύστημα Android (SQLite Database), στην οποία μπορούν να αποθηκεύουν ή να διαβάζουν δεδομένα οι Content Providers.

4.4 Broadcast Receivers

Οι Broadcast Receivers ενημερώνονται για κάποιο συγκεκριμένο γεγονός από το λειτουργικό σύστημα και τότε ενεργοποιούνται. Τέτοια γεγονότα μπορεί να είναι ότι η οθόνη έχει σβήσει, ότι η στάθμη της μπαταρίας είναι χαμηλή κλπ. Για παράδειγμα αρκετές εφαρμογές λήψης φωτογραφιών απενεργοποιούν το flash, σε περίπτωση χαμηλής μπαταρίας για λόγους εξοικονόμησης ενέργειας. Ενημερώνονται, δηλαδή, από τον Broadcast Receiver για το γεγονός αυτό και πράττουν ανάλογα. Οι Broadcast Receivers δεν παρέχουν user interface, αλλά μπορούν να ενημερώσουν το χρήστη για κάποιο γεγονός μέσω του status bar notifications.

4.5 Resources

Στα resources μιας εφαρμογής ορίζεται το layout των activities, οι διάφορες εικόνες και λεκτικά που χρησιμοποιούνται στα activities. Σε κάθε activity αντιστοιχεί ένα Layout αρχείο, το οποίο περιγράφει τη θέση των διάφορων αντικειμένων στην οθόνη. Το layout αρχείο είναι ένα αρχείο XML. Στην πράξη το αρχείο αυτό διαμορφώνεται από κατάλληλους γραφικούς editors που προσφέρονται από ολοκληρωμένα περιβάλλοντα ανάπτυξης, όπως το Android Studio. Στην ενότητα για την διεπαφή χρήστη αναφέρθηκε ότι η διάταξη των γραφικών στοιχείων δηλώνεται σε αρχεία xml. Συγκεκριμένα, στο project της εφαρμογής μας υπάρχει ο φάκελος res/layout/ στον οποίο τοποθετούμε όλα τα αρχεία

xml που αφορούν στο user interface της εφαρμογής μας (το res προέρχεται από το resources). Η εντολή που χρησιμοποιείται για να δηλωθεί το αρχείο xml που θα χρησιμοποιηθεί σε μία activity είναι η setContentView().

4.6 User Interface

Το χρήστη δεν τον ενδιαφέρει τι συμβαίνει πίσω από την εφαρμογή που τρέχει. Τον ενδιαφέρει το γραφικό περιβάλλον και η ευχρηστία της εφαρμογής. User Interface και λειτουργικότητα μιας εφαρμογής είναι δύο αλληλένδετοι όροι. Μεγάλες εταιρίες εφαρμογών δίνουν μεγάλη βάση στο σχεδιασμό του περιβάλλοντος εργασίας και μάλιστα δεν είναι λίγες οι φορές που είναι δυσκολότερος ο σχεδιασμός μιας εφαρμογής από την υλοποίησή της.

4.6.1 Layout

Η διάταξη των γραφικών στοιχείων της οθόνης ονομάζονται layouts. Ένα layout μπορεί να περιλαμβάνει πολλά διατεταγμένα layouts.

Τα layouts χωρίζονται σε τέσσερα διαφορετικά είδη. Στην **LinearLayout** (γραμμική διάταξη), **RelativeLayout** (σχετική διάταξη), **FrameLayout** (διάταξη πλαισίου) και **TableLayout** (διάταξη πίνακα).

Το **LinearLayout** αποτελεί διάταξη στοιχείων σε οριζόντια ή κατακόρυφη σειρά. Όταν δηλώνουμε δύο στοιχεία σε ένα οριζόντιο LinearLayout τότε αυτά θα εμφανίζονται στην οθόνη σε οριζόντια θέση το ένα δίπλα στο άλλο με τη σειρά που δηλώθηκαν. Αντίστοιχα, θα διαταχθούν σε μια κατακόρυφη θέση.

Στο **RelativeLayout** υπάρχει μεγαλύτερη ελευθερία στη διάταξη των φραφικών, εφόσον η θέση του γραφικού ορίζεται με βάση τη θέση του προηγούμενου. Ωστόσο, το ενδεχόμενο λάθος δεν επιλύεται εύκολα.

Το **FrameLayout** αποτελεί την απλούστερη διάταξη στοιχείων. Είναι απλά ένας κενός χώρος τον οποίο μπορούμε να γεμίσουμε με κάποιο αντικείμενο, π.χ. μία εικόνα.

Τέλος, το **TableLayout** αποτελεί διάταξη πίνακα, δηλαδή μπορεί να διατάσσει τα παιδιά του (children) σε σειρές και στήλες.

Σε όλα τα Layouts μπορούμε να ρυθμίσουμε αρκετές παραμέτρους, όπως μέγεθος (πλάτος και ύψος), οριζόντια ή κατακόρυφη

διάταξη, βαρύτητα (layout gravity) και άλλες, προκειμένου να καταλήξουμε στο αποτέλεσμα που θέλουμε ευκολότερα και πιο γρήγορα.

Η δήλωση των Layouts μπορεί να γίνει με δύο τρόπους:

1) μέσω αρχείων .xml
2) μέσα στις Activities

Τα αρχεία .xml αποτελούν στατικό τρόπο δημιουργίας των γραφικών στοιχείων, αποθηκεύονται σε συγκεκριμένο φάκελο του Project και τα καλούμε μέσω των Activities. Είναι εύκολα στη συγγραφή και στην κατανόηση. Ωστόσο, πολλές φορές δε γνωρίζουμε τη διάταξη των γραφικών. Στις περιπτώσεις αυτές δημιουργούμε δυναμικά τα Layouts μέσα στις Activities, από όπου ορίζουμε και τη διάταξή τους.

4.6.2 Notifications

Σε πολλές περιπτώσεις θέλουμε να ενημερώσουμε το χρήστη για κάποιο γεγονός ή αποτέλεσμα σχετικό με την εφαρμογή μας. Ορισμένα από αυτά τα γεγονότα απαιτούν κάποια απάντηση από το χρήστη και άλλα όχι.

Για παράδειγμα, όταν ο χρήστης αποθηκεύει ένα αρχείο, θα θέλαμε να δει κάποιο μήνυμα ότι το αρχείο αποθηκεύτηκε επιτυχώς. Ή όταν η εφαρμογή μας τρέχει στο background και θέλει να ενημερώσει το χρήστη για κάποιο γεγονός, θα πρέπει να στείλει κάποια ειδοποίηση την οποία ο χρήστης να μπορεί να "ανοίξει" όταν αυτός επιθυμεί. Στην πρώτη περίπτωση χρησιμοποιούμε toast notification, ενώ στη δεύτερη status bar notification

Toast Notification

Το toast notification είναι ένα μήνυμα που εμφανίζεται για λίγα δευτερόλεπτα στο παράθυρο που βρίσκεται ο χρήστης, οποιασδήποτε εφαρμογής και αν είναι αυτό. Ο χώρος που καταλαμβάνει είναι ο ελάχιστος απαιτούμενος ώστε το μήνυμα να είναι εμφανές, ενώ ο χρήστης μπορεί όσο εμφανίζεται το μήνυμα, να αλληλεπιδρά με την activity στην οποία βρίσκεται. Δεν υπάρχει κάποια επιλογή σε αυτή την ειδοποίηση, παρά μόνο ενημέρωση, δηλαδή ο χρήστης δε μπορεί να αλληλεπιδράσει με την ειδοποίηση. Χρησιμοποιείται συνήθως για μικρά μηνύματα που δεν απαιτούν κάποια ενέργεια από το χρήστη, όπως για παράδειγμα "Το αρχείο αποθηκεύτηκε επιτυχώς", "Το ξυπνητήρι ορίστηκε στις ..." κλπ.

Status Bar Notification

Το status bar notification, όπως φανερώνει και το όνομά της, είναι μία ειδοποίηση η οποία εμφανίζεται στη status bar του κινητού τηλεφώνου μας, και την οποία μπορούμε να ανοίξουμε είτε βρισκόμαστε στο κεντρικό μενού του τηλεφώνου μας, είτε σε κάποια εφαρμογή. Αντίθετα με την toast, η status bar notification μπορεί να επιλεχθεί και να ξεκινήσει κάποια λειτουργία ανάλογα με τις ενέργειες που έχουμε ορίσει στον κώδικα της εφαρμογής. Για παράδειγμα, όταν κατεβάζουμε ένα αρχείο από το διαδίκτυο, όταν η λήψη ολοκληρωθεί, θα θέλαμε να επιλέξουμε την ειδοποίηση αυτή και με τον τρόπο αυτό είτε να ανοίξουμε το φάκελο που βρίσκεται το αρχείο, είτε να το τρέξουμε. Τις περισσότερες φορές οι toast notifications ενεργοποιούνται από activities, ενώ οι status bar notifications από services.

4.7 Google Maps

Το Google Maps API είναι μια web εφαρμογή υπηρεσιών, η οποία παρέχεται από την Google και προσφέρει οδικούς χάρτες και υπηρεσίες πλοήγησης σε δικτυακούς τόπους ή σε εφαρμογές Android. Οι υπηρεσίες που προσφέρονται είναι οι εξής:

- οδικοί χάρτες
- πλοήγηση διαδρομής (με αυτοκίνητο, με δημόσια μέσα μεταφοράς ή με τα πόδια)
- εντοπισμός επιχειρήσεων σε χώρες σε όλο τον κόσμο.

Το Google Maps API είναι δωρεάν για εμπορική χρήση, υπό τον όρο ότι το site/εφαρμογή στην οποία χρησιμοποιείται θα είναι προσβάσιμο στο κοινό χωρίς να χρεώνει τον χρήστη του για κάθε πρόσβαση, και δεν παράγουν περισσότερα από 25.000 προσβάσεις χάρτη ανά ημέρα. Η ερώτηση στο Google Maps API γίνεται με την αποστολή ενός αιτήματος HTTP GET στην Web εφαρμογή, και επιστρέφει την απάντηση σε μορφή XML ή JSON. Η εφαρμογή η οποία περιγράφεται σε αυτή την πτυχιακή χρησιμοποιεί μηνύματα JSON όπως περιγράφονται στη συνέχεια.

Οι εφαρμογές Android χρησιμοποιούν μια ειδικά σχεδιασμένη για αυτές έκδοση του API, το Google Maps Android API v2. Για να χρησιμοποιηθεί το Google Maps API από μία εφαρμογή Android απαιτείται η απόκτηση ενός "κλειδιού" (Google Maps API Key) από τον δημιουργό της εφαρμογής που το χρησιμοποιεί και η εισαγωγή του στον κώδικα της εφαρμογής. Για την απόκτηση είναι απαραίτητη η δημιουργία λογαριασμού Google και η εγγραφή στο Google APIs Console (https://code.google.com/apis/console/).

Στη συνέχεια για τη χρήση του εργαλείου πρέπει να οριστεί στο xml αρχείο Manifest της εφαρμογής (AndroidManifest.xml) το κλειδί το οποίο έχει δοθεί από την Google με την εισαγωγή του παρακάτω αντικειμένου σαν "παιδί" στοιχείου <application>:

```
<meta-data
android:name="com.google.android.maps.v2.API_KEY"
android:value="your_api_key"/>
```

Στο παραπάνω αντικαθίσταται το "your_api_key" με το κλειδί το οποίο έχει αποκτηθεί από το Google APIs Console.

Για τη χρήση του Google Maps Android API v2 χρειάζεται ο καθορισμός στην εφαρμογή δικαιωμάτων πρόσβασης σε λειτουργείες του συστήματος. Ο καθορισμός ενός δικαιώματος γίνεται με την προσθήκη ενός στοιχείου <uses-permission> σαν παιδί του αντικειμένου <manifest> στο αρχείο AndroidManifest.xml της εφαρμογής με το συντακτικό:

<uses-permission android:name="permission_name"/>

Τα απαραίτητα δικαιώματα που χρειάζεται να καθοριστούν για τη χρήση του Google Maps API v2 είναι:

- **android.permission.INTERNET:** Χρησιμοποιεί το API για τη λήψη του χάρτη απο τους Google Maps servers στη συσκευή.
- **android.permission.ACCESS_NETWORK_STATE:** Επιτρέπει στο API να ελέγξει τη κατάσταση των συνδέσεων τις συσκευής, έτσι ώστε να καθορίσει αν μπορεί να γίνει λήψη δεδομένων.
- **android.permission.WRITE_EXTERNAL_STORAGE:** Επιτρέπει στο API να καταχωρίσει τα δεδομένα του χάρτη στον χώρο αποθήκευσης της συσκευής.
- **com.google.android.providers.gsf.permission.READ_GSERVICES:** Επιτρέπει στο API την πρόσβαση σε web υπηρεσίες της Google.

Εικόνα 14. Google Maps

4.8 JavaScript Object Notation (JSON)

Το JavaScript Object Notation, το οποίο είναι γνωστό και ως JSON, είναι ένα ελαφρύ πρότυπο ανταλλαγής δεδομένων. Το JSON είναι ένα πρότυπο κειμένου, το οποίο είναι τελείως ανεξάρτητο από γλώσσες προγραμματισμού αλλά χρησιμοποιεί πρακτικές οι οποίες είναι γνωστές στους προγραμματιστές της οικογένειας προγραμματισμού C, συμπεριλαμβανομένων των C, C++, C#, Java, JavaScript και πολλών άλλων.

Το JSON προσφέρεται ως εναλλακτική λύση στην XML, όμως δεν μπορεί να την αντικαταστήσει καθότι δεν υποστηρίζει Schema validation. Η χρήση της είναι πολύ εύκολη υπόθεση καθότι το JSON έχει και αυτό self-documented format, το οποίο περιγράφει τη δομή των δεδομένων. Ο κώδικας που χρησιμοποιείται έχει πολύ μικρό μέγεθος πράγμα που το καθιστά ιδανικό για κινητές συσκευές λόγω του ότι στην πλειοψηφία τους οι πάροχοι κινητής τηλεφωνίας ακολουθούν την πολιτική της ογκοχρέωσης.

4.9 AsyncTask

Η AsyncTask επιτρέπει την σωστή και εύκολη χρήση του UI Thread. Η κλάση αυτή μας επιτρέπει να εκτελέσουμε λειτουργίες στο παρασκήνιο και να δημοσιεύονται τα αποτελέσματα στο UI Thread χωρίς να χρειάζεται να τα χειριστούν τα threads.

Η AsyncTask έχει σχεδιαστεί για να είναι μια κλάση αρωγός γύρω από το Thread και τον Handler και δεν αποτελεί ένα γενικό πλαίσιο threading. Τα AsyncTasks θα πρέπει ιδανικά να χρησιμοποιηθούν για μικρής διάρκειας χειρισμούς (λίγα δευτερόλεπτα το πολύ). Εάν πρέπει να κρατήσει το thread για μεγάλο χρονικό διάστημα, συνιστάται να χρησιμοποιήσουμε τα διάφορα APIs που παρέχονται από την java.util.concurrentpackage όπως Thread Pool Executor και Future Task. Στην δική μας εφαρμογή η AsyncTask ήταν αρκετή για τις διεργασίες που εκτελούμε.

Η AsyncTask έχει τέσσερα στάδια:

- **onPreExecute**: Αυτή η μέθοδος εκτελείται πριν την doInBackground.
- **doInBackground**: Η doInBackground χρησιμοποιείται για υπολογισμούς που παίρνουν ώρα να εκτελεστούν, δηλαδή είναι το βασικό βήμα για να εκτελέσουμε τον κώδικα που θέλουμε. Οι παράμετροι της AsyncTask παίρνονται από αυτό το στάδιο και το αποτέλεσμα επιστρέφεται στο επόμενο βήμα που αναλύουμε παρακάτω.
- **onPostExecute**: Αυτή η μέθοδος εκτελείται αφότου η doInBackground ολοκληρώσει την κλήση της. Το αποτέλεσμα από την doInBackround περνιέται σε αυτή τη μέθοδο.
- **onProgressUpdate**: Αυτή η μέθοδος καλείται καλώντας την publishProgress ανά πάσα στιγμή ενώ εκτελείται η doInBackground. Συνήθως η onProgressUpdate χρησιμοποιείται για να ενημερώσουμε το UI Thread με κάποια μπάρα προόδου ενώ εκτελείται η doInBackground.

4.9.1 Το πλεονέκτημα της AsyncTask

Το λειτουργικό Android εφαρμόζει ένα μοντέλου ενιαίου Thread και οποτεδήποτε μια Android εφαρμογή ξεκινάει, δημιουργείται και ένα αντίστοιχο Thread. Τώρα, υποθέτοντας ότι εκτελούμε μια λειτουργία

δικτύου με το πάτημα ενός κουμπιού στην εφαρμογή μας. Όταν πατήσουμε το κουμπί θα γίνει μια αίτηση στον server και έτσι πρέπει να περιμένουμε την απάντηση του server. Εξαιτίας του ενιαίου μοντέλου στο Thread, όση ώρα περιμένουμε την απάντηση του server, η εφαρμογή μας δεν θα ανταποκρίνεται. Έτσι χρησιμοποιούμε την AsyncTask για να αποφύγουμε λειτουργίες που παίρνουν αρκετό σχετικά χρόνο να εκτελεστούν και κάνουν το user interface της εφαρμογής μας να μην ανταποκρίνεται.

Έτσι δημιουργούμε ένα νέο Thread και τρέχουμε εκεί τη λειτουργία μας κάνοντας το βασικό Thread της εφαρμογής να παραμένει ανταποκρίσιμο στο χρήστη. Η κλάση AsyncTask περιέχει δηλαδή μεθόδους για να τρέξουμε μια λειτουργία που απαιτεί κάποιο χρόνο, στο παρασκήνιο, αλλά ταυτόχρονα μπορεί να επικοινωνεί και με το Thread της εφαρμογής και να συγχρονίζεται με αυτό ή και να το ενημερώνει όταν χρειάζεται.

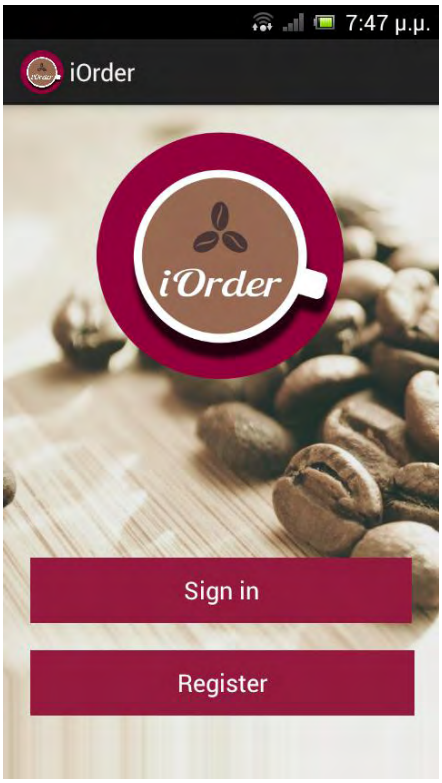
5. Η εφαρμογή iOrder

Σε αυτό το κεφάλαιο θα παρουσιάσουμε τη λειτουργία της εφαρμογής μέσα από κάποια screenshots και ενδεικτικά μέσα από κάποια κομμάτια κώδικα.

5.1 Αρχική οθόνη

Αρχικά εμφανίζεται στο χρήστη η παρακάτω οθόνη με το λογότυπο της εφαρμογής και τα εξής κουμπιά:

1. **Sign in:** μέσω του οποίου ο χρήστης μπορεί να πραγματοποιήσει είσοδο, εφόσον έχει εγγραφεί παλαιότερα στην εφαρμογή
2. **Register:** μέσω του οποίου μπορεί να κάνει εγγραφή για πρώτη φορά στην εφαρμογή



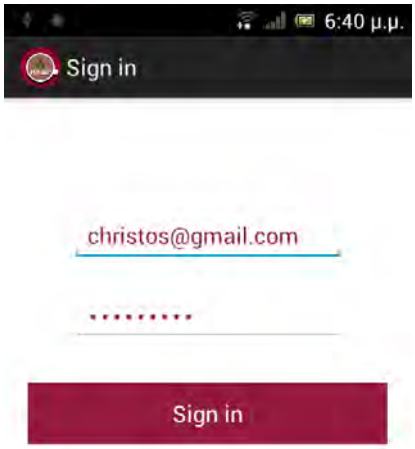
Εικόνα 15. Αρχική Οθόνη iOrder

Ο κώδικας είναι ο εξής:

```
package activities;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import mycompany.iorder.R;

public class LoginActivity extends Activity {

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


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

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar
will
        // automatically handle clicks on the Home/Up button, so
long
        // as you specify a parent activity in
AndroidManifest.xml.
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            return true;
        }
        return super.onOptionsItemSelected(item);
    }



    //connect LoginActivity with NewAccountActivity
    public void newAccount(View view)
    {
        Intent intent = new Intent(this,
NewAccountActivity.class);
        startActivity(intent);
    }

    //connect LoginActivity with Sign_inActivity
    public void sign_in_via_iorder(View view)
    {
        Intent intent = new Intent(this, Sign_inActivity.class);
        startActivity(intent);
    }
}
```

5.2 Οθόνη Sign in

Ο χρήστης προκειμένου να εισέλθει στην εφαρμογή πρέπει να εισάγει το email και το password, που είχε ορίσει ο ίδιος κατά την εγγραφή του.



Εικόνα 16. iOrder sign in

Ο κώδικας είναι ο εξής:

```
package activities;
import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import com.google.gson.Gson;
import org.apache.http.HttpResponse;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONObject;
import java.io.BufferedReader;
```

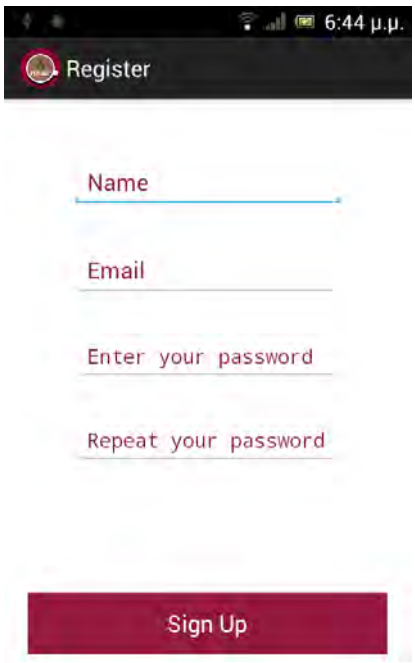
```
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;

import models.Credentials;
import models.User;
import models.iOrder;
import mycompany.iorder.R;

public class Sign_inActivity extends Activity /*implements
View.OnClickListener*/ {

    User user;

    private EditText email_given, pass_given;
    private Button ok;

    // Progress Dialog
    private ProgressDialog pDialog;

    private static final String LOGIN_URL =
"https://myiorderapp.herokuapp.com/api/v1/users/sessions/signin";

    private static final String TAG_SUCCESS = "success";
    private static final String TAG_MESSAGE = "message";

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

        email_given = ((EditText) findViewById(R.id.email_signin));
        pass_given = ((EditText) findViewById(R.id.pass_signin));

        email_given.setText("kir.christos@gmail.com");
        pass_given.setText("app12345");

        if (email_given.equals("") || pass_given.equals(""))
          Toast.makeText(Sign_inActivity.this,
R.string.error_all_fields_are_required, Toast.LENGTH_LONG).show();

        ok = ((Button)findViewById(R.id.buttonSign_in));
        ok.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                switch (v.getId()) {
                    case R.id.buttonSign_in:
                        new HttpAsyncTask().execute();
                }
            }
        });
    }



    public String POST(String url, User users) {
        InputStream inputStream = null;
        String result = "";
        try {

            // 1. create HttpClient
```

```
            HttpClient httpclient = new DefaultHttpClient();

            // 2. make POST request to the given URL
            HttpPost httpPost = new HttpPost(url);

            String json = "";

            // 3. build jsonObject
            JSONObject jsonObject = new JSONObject();

            jsonObject.accumulate("email", email_given.getText());
            jsonObject.accumulate("password", pass_given.getText());

            // 4. convert JSONObject to JSON to String
            json = jsonObject.toString();

            // 5. set json to StringEntity
            StringEntity se = new StringEntity(json);

            // 6. set httpPost Entity
            httpPost.setEntity(se);

            // 7. Set some headers to inform server about the type of
the content
            httpPost.setHeader("Accept", "application/json");
            httpPost.setHeader("Content-type", "application/json");

            // 8. Execute POST request to the given URL
            HttpResponse httpResponse = httpclient.execute(httpPost);

            // 9. receive response as inputStream
            inputStream = httpResponse.getEntity().getContent();

            // 10. convert inputstream to string
            if (inputStream != null &&
httpResponse.getStatusLine().getStatusCode()==201)
                result = convertInputStreamToString(inputStream);
            else
                result = "Authentication Failure";

        } catch (Exception e) {
            Log.d("InputStream", e.getLocalizedMessage());
        }

        // 11. return result
        return result;
    }


    class HttpAsyncTask extends AsyncTask<String, Void, String> {
        @Override
        protected String doInBackground(String... urls) {

            user = new User();

            user.setEmail(email_given.toString());
            user.setPassword(pass_given.toString());


            return POST(LOGIN_URL, user);
        }
```

```
        // onPostExecute displays the results of the AsyncTask.
        @Override
        protected void onPostExecute(String result) {

            if(result != "Authentication Failure" && result != null)
            {
                iOrder iorder = ((iOrder) getApplicationContext());

                String x = result.replace("\\","");
                Credentials credentials = new
Gson().fromJson(x.substring(1,x.length()-1), Credentials.class);
                iorder.setUserId(credentials.getId().toString());

iorder.setAuthenticationToken(credentials.getToken());
                Toast.makeText(getBaseContext(), "Data Sent!",
Toast.LENGTH_LONG).show();
                sign_in();
            }
            else
            {
                Toast.makeText(getBaseContext(), "Authentication
Failure", Toast.LENGTH_LONG).show();
            }
        }
    }

        private String convertInputStreamToString(InputStream
inputStream) throws IOException {
            BufferedReader bufferedReader = new BufferedReader(new
InputStreamReader(inputStream));
            String line = "";
            String result = "";
            while ((line = bufferedReader.readLine()) != null)
                result += line;

            inputStream.close();
            return result;

        }




    //connect Sign_inActivity with OrderActivity
    public void sign_in()
    {
        Intent intent = new Intent(this, MapsActivity.class);
        startActivity(intent);
    }

}
```

5.3 Οθόνη Register

Ο χρήστης προκειμένου να εγγραφεί στην εφαρμογή πρέπει να συμπληρώσει τα παρακάτω πεδία: όνομα, email, password και να επαναλάβει την εισαγωγή του password του. Το email του για να είναι έγκυρο θα πρέπει να περιέχει τουλάχιστον έξι χαρακτήρες, ενώ ο κωδικός που θα εισάγει θα πρέπει να έχει τουλάχιστον οκτώ χαρακτήρες. Τέλος, για να μπορέσει να ολοκληρωθεί η εγγραφή ο χρήστης θα πρέπει να συμπληρώσει υποχρεωτικά όλα τα πεδία.



Εικόνα 17. iOrder register

Ο κώδικας είναι ο εξής:

```
package activities;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.os.AsyncTask;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import org.json.JSONObject;
import mycompany.iorder.R;
import models.User;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
```

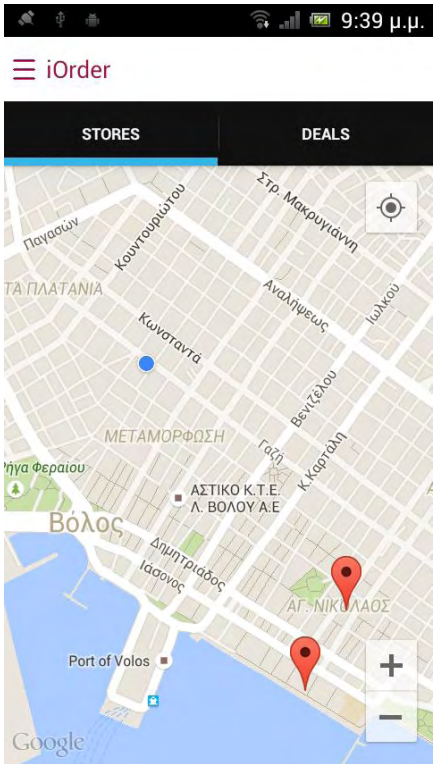
```
import java.io.InputStreamReader;
import org.apache.http.HttpResponse;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.DefaultHttpClient;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.util.Log;
import android.widget.TextView;

public class NewAccountActivity extends Activity  {
    private String name = null;
    private String password = null;
    private String password_confirmation = null;
    private String email = null;
    TextView tvIsConnected;
    Button signupPost;
    User user;

    @Override
    protected void onCreate(Bundle savedInstanceState) {


        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_new_account);

        Button ok = (Button) findViewById(R.id.button_sign_up);
        ok.setOnClickListener(new View.OnClickListener() {

            public void onClick(View v) {


                tvIsConnected = (TextView)
findViewById(R.id.textView2);
                name = ((EditText)
findViewById(R.id.name_signup)).getText().toString();
                email = ((EditText)
findViewById(R.id.email_signup)).getText().toString();
                password = ((EditText)
findViewById(R.id.pass_signup)).getText().toString();
                password_confirmation = ((EditText)
findViewById(R.id.repassword)).getText().toString();
                signupPost = (Button)
findViewById(R.id.button_sign_up);

                if (name.equals("") || email.equals("") ||
password.equals("")){
                    Toast.makeText(NewAccountActivity.this,
R.string.error_all_fields_are_required, Toast.LENGTH_LONG).show();
                }
                else if (!password.equals(password_confirmation)){
                    Toast.makeText(NewAccountActivity.this,
R.string.preferences_passwords_do_not_match,
Toast.LENGTH_LONG).show();

                }
                else{
                    new
HttpAsyncTask().execute("https://myiorderapp.herokuapp.com/api/v1/use
rs/signup");
```

```
                    Intent intent = new
Intent(NewAccountActivity.this, MapsActivity.class);
                    startActivity(intent);
                }



                // check if you are connected or not
                if (isConnected()) {
                    tvIsConnected.setBackgroundColor(0xFF00CC00);

                } else {

                }
            }
        });
    }

    public String POST(String url, User users) {
        InputStream inputStream = null;
        String result = "";
        try {

            // 1. create HttpClient
            HttpClient httpclient = new DefaultHttpClient();

            // 2. make POST request to the given URL
            HttpPost httpPost = new HttpPost(url);

            String json = "";

            // 3. build jsonObject
            JSONObject jsonObject = new JSONObject();
            jsonObject.accumulate("name", users.getName());
            jsonObject.accumulate("email", users.getEmail());
            jsonObject.accumulate("password", users.getPassword());
            jsonObject.accumulate("password_confirmation",
users.getRepassword());

            // 4. convert JSONObject to JSON to String
            json = jsonObject.toString();

            // 5. set json to StringEntity
            StringEntity se = new StringEntity(json);

            // 6. set httpPost Entity
            httpPost.setEntity(se);

            // 7. Set some headers to inform server about the type of
the content
            httpPost.setHeader("Accept", "application/json");
            httpPost.setHeader("Content-type", "application/json");

            // 8. Execute POST request to the given URL
            HttpResponse httpResponse = httpclient.execute(httpPost);

            // 9. receive response as inputStream
            inputStream = httpResponse.getEntity().getContent();

            // 10. convert inputstream to string
            if (inputStream != null)
```

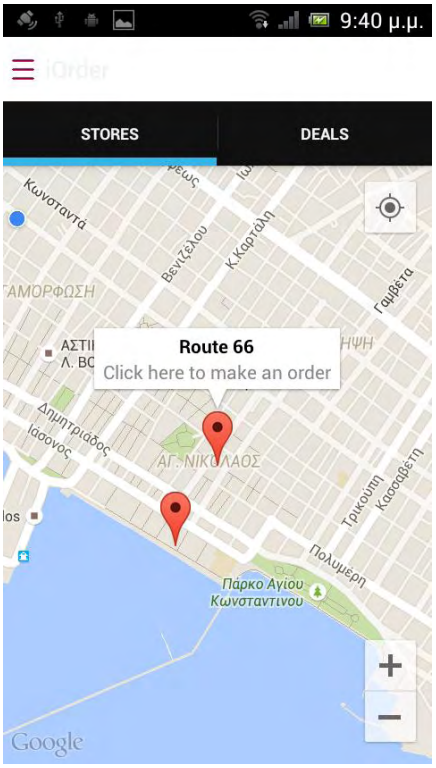
```
                result = convertInputStreamToString(inputStream);
            else
                result = "Did not work!";

        } catch (Exception e) {
            Log.d("InputStream", e.getLocalizedMessage());
        }

        // 11. return result
        return result;
    }

    public boolean isConnected() {
        ConnectivityManager connMgr = (ConnectivityManager)
getSystemService(Activity.CONNECTIVITY_SERVICE);
        NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();
        if (networkInfo != null && networkInfo.isConnected())
            return true;
        else
            return false;
    }

    class HttpAsyncTask extends AsyncTask<String, Void, String> {
        @Override
        protected String doInBackground(String... urls) {

            user = new User();
            user.setName(name.toString());
            user.setEmail(email.toString());
            user.setPassword(password.toString());
            user.setRepassword(password_confirmation.toString());
            return POST(urls[0], user);
        }

        // onPostExecute displays the results of the AsyncTask.
        @Override
        protected void onPostExecute(String result) {
            Toast.makeText(getBaseContext(), "Data Sent!",
Toast.LENGTH_LONG).show();

            tvIsConnected.setText(result.toString());
        }

    }
        private String convertInputStreamToString(InputStream
inputStream) throws IOException {
            BufferedReader bufferedReader = new BufferedReader(new
InputStreamReader(inputStream));
            String line = "";
            String result = "";
            while ((line = bufferedReader.readLine()) != null)
                result += line;

            inputStream.close();
            return result;

        }
}
```

5.4 Οθόνη Maps

Η εφαρμογή εντοπίζει την τρέχουσα θέση του χρήστη στο χάρτη, εφόσον είναι συνδεδεμένος στο Internet. Στη συνέχεια, ο χρήστης μπορεί να δει ποια είναι τα καταστήματα από τα οποία μπορεί να παραγγείλει, καθώς εμφανίζονται στο χάρτη με markers. Αφού αποφασίσει από ποιο κατάστημα επιθυμεί να κάνει την παραγγελία του μπορεί να πατήσει στο αντίστοιχο marker, το οποίο θα τον ανακατευθύνει στην αντίστοιχη οθόνη με τα διαθέσιμα προϊόντα του μαγαζιού.



Εικόνα 18. iOrder map



Εικόνα 19. iOrder store marker

Ο κώδικας είναι ο εξής:

```
package activities;
import android.app.ActionBar;
import android.app.Fragment;
import android.app.FragmentManager;
import android.app.FragmentTransaction;
import android.content.Intent;
import android.content.res.Configuration;
import android.graphics.Color;
import android.graphics.drawable.ColorDrawable;
import android.os.Bundle;
import android.support.v4.app.ActionBarDrawerToggle;
import android.support.v4.app.FragmentActivity;
import android.support.v4.view.GravityCompat;
import android.support.v4.view.ViewPager;
import android.support.v4.widget.DrawerLayout;
```

```
import android.text.Html;
import android.util.Log;
import android.view.Gravity;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.app.ActionBar.Tab;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import adapters.TabsPagerAdapter;
import fragments.VersionFragment;
import mycompany.iorder.R;
import activities.TopUpActivity;

public class MapsActivity extends FragmentActivity implements
ActionBar.TabListener {

    private ViewPager viewPager;
    private ActionBar actionBar;
    private TabsPagerAdapter mAdapter;

    private DrawerLayout drawerLayout;
    private ListView drawerList;
    private ActionBarDrawerToggle drawerToggle;

    private CharSequence drawerTitle;
    private CharSequence title;

    String[] andoridVeriosnArray;

    // Tab titles
    private String[] tabs = { "Stores", "Deals"};

    @Override
    public  void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_maps);

        getActionBar().setIcon(android.R.color.transparent);
        getActionBar().setBackgroundDrawable(new
ColorDrawable(Color.parseColor("#FFFFFF")));
        getActionBar().setTitle(Html.fromHtml("<font
color='#951A3F'>iOrder</font>"));

        // Initilization
        viewPager = (ViewPager) findViewById(R.id.pager);
        actionBar = getActionBar();
        mAdapter = new TabsPagerAdapter(getSupportFragmentManager());

        viewPager.setAdapter(mAdapter);
        actionBar.setHomeButtonEnabled(false);
        actionBar.setNavigationMode(ActionBar.NAVIGATION_MODE_TABS);

        // Adding Tabs
        for (String tab_name : tabs) {
            actionBar.addTab(actionBar.newTab().setText(tab_name)
```

```
                    .setTabListener(this));
        }

        /**
         * on swiping the viewpager make respective tab selected
         * */
        viewPager.setOnPageChangeListener(new
ViewPager.SimpleOnPageChangeListener() {

            @Override
            public void onPageSelected(int position) {
                // on changing the page
                // make respected tab selected
                actionBar.setSelectedNavigationItem(position);
            }

            @Override
            public void onPageScrolled(int arg0, float arg1, int
arg2) {
            }

            @Override
            public void onPageScrollStateChanged(int arg0) {
            }
        });

        title = drawerTitle = getTitle();

        andoridVeriosnArray = new String[] { "Top Up", "Alarm",
"Coffee Beans", "Log Out"};

        drawerLayout = (DrawerLayout)
findViewById(R.id.drawerLayout);
        drawerList = (ListView) findViewById(R.id.drawerList);

        // Set shadow to drawer
        drawerLayout.setDrawerShadow(R.drawable.drawer_shadow,
                GravityCompat.START);

        // Set adapter to drawer
        drawerList.setAdapter(new ArrayAdapter<String>(this,
                R.layout.drawer_list_item, andoridVeriosnArray));

        // set up click listener on drawer
        drawerList.setOnItemClickListener(new
DrawerItemClickListener());

        // set app icon to behave as action to toggle navigation
drawer
        getActionBar().setDisplayHomeAsUpEnabled(true);

        // ActionBarDrawerToggle ties together the the interactions
the sliding
        // drawer and the app icon
        drawerToggle = new ActionBarDrawerToggle(this, // Host
Activity
                drawerLayout, // layout container for navigation
drawer
                R.drawable.ic_drawer, // Application Icon
                R.string.drawer_open, // Open Drawer Description
                R.string.drawer_close) // Close Drawer Description
```

```
        {
            public void onDrawerClosed(View view) {
                getActionBar().setTitle(title);
                invalidateOptionsMenu();
            }

            public void onDrawerOpened(View drawerView) {
                getActionBar().setTitle(drawerTitle);
                invalidateOptionsMenu();
            }
        };
        drawerLayout.setDrawerListener(drawerToggle);
    }

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

    @Override
    public boolean onPrepareOptionsMenu(Menu menu) {

        return super.onPrepareOptionsMenu(menu);

    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {

        // Listen Toggle state of navigation Drawer
        if (drawerToggle.onOptionsItemSelected(item)) {
            return true;
        }
        return super.onOptionsItemSelected(item);

    }

    // Click Event of navigation drawer item click
    private class DrawerItemClickListener implements
            ListView.OnItemClickListener {
        @Override
        public void onItemClick(AdapterView<?> parent, View view, int
position,
                                long id) {
            drawerLayout.closeDrawers();
            Bundle args = new Bundle();
            args.putString("name", andoridVeriosnArray[position]);
            Intent i = new Intent(MapsActivity.this,
TopUpActivity.class);
            switch(position){
                case 0:
                    i = new Intent(MapsActivity.this,
TopUpActivity.class);
                    break;
                case 1:
                    i = new Intent(MapsActivity.this,
AlarmActivity.class);
```

```
                    break;
                case 2:
                    i = new Intent(MapsActivity.this,
CoffeeBeansActivity.class);
                    break;
                case 3:
                    i = new Intent(MapsActivity.this, LogOut.class);
                    break;

            }
            startActivity(i);
        }
    }

    @Override
    protected void onPostCreate(Bundle savedInstanceState) {
        super.onPostCreate(savedInstanceState);
        // Sync the toggle state after onRestoreInstanceState has
occurred.
        drawerToggle.syncState();
    }

    @Override
    public void onConfigurationChanged(Configuration newConfig) {
        super.onConfigurationChanged(newConfig);
        // Pass any configuration change to the drawer toggls
        drawerToggle.onConfigurationChanged(newConfig);
    }

    @Override
    public void onTabReselected(Tab tab, FragmentTransaction ft) {
        Log.d("onTabReselected", String.valueOf(tab.getPosition()));
    }

    @Override
    public void onTabSelected(Tab tab, FragmentTransaction ft) {
        // on tab selected
        // show respected fragment view
        Log.d("onTabSelected", String.valueOf(tab.getPosition()));
        viewPager.setCurrentItem(tab.getPosition());
    }

    @Override
    public void onTabUnselected(Tab tab, FragmentTransaction ft) {
        Log.d("onTabUnselected", String.valueOf(tab.getPosition()));
    }


}
```

```
package fragments;
import activities.Order_Activity;
import android.content.Context;
import android.content.Intent;
import android.location.Criteria;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentTransaction;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.FrameLayout;
import asynctasks.StoresAsyncTask;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;
import interfaces.OnStoresTaskCompletedInterface;
import java.util.ArrayList;
import java.util.HashMap;

import models.Store;
import mycompany.iorder.R;

public class StoresFragment extends Fragment implements
LocationListener, OnStoresTaskCompletedInterface {

    // Keys for storing activity state in the Bundle.
    protected final static String LOCATION_KEY = "location-key";
    private GoogleMap googleMap;
    private FrameLayout mapContainer;
    private SupportMapFragment mapFragment;
    private ArrayList<Store> stores;
    private StoresAsyncTask storesAsyncTask;
    private HashMap<LatLng, String[]> markersMap;

    public StoresFragment()
    {
    }

    private void SetUpShops()
    {
        markersMap = new HashMap<LatLng, String[]>();
        for (final Store store: stores) {

            MarkerOptions markerOptions = new
MarkerOptions().position(new
LatLng(Double.parseDouble(store.getLatitude()),
Double.parseDouble(store.getLongitude()))).title(store.getName()).sni
ppet("Click here to make an order");

            googleMap.addMarker(markerOptions);
            String[] storeValues = {store.getId(), store.getName()};
            markersMap.put(new
```

```
LatLng(Double.parseDouble(store.getLatitude()),
Double.parseDouble(store.getLongitude())), storeValues);
            googleMap.setOnInfoWindowClickListener(new
GoogleMap.OnInfoWindowClickListener() {

                @Override
                public void onInfoWindowClick(Marker marker) {
                    Intent intent = new Intent(getActivity(),
Order_Activity.class);
                    for (HashMap.Entry<LatLng,String[]> entry :
markersMap.entrySet()) {
                        Log.d("hash position:" ,
entry.getKey().toString());
                        Log.d("marker position:" ,
marker.getPosition().toString());
                        Log.d("hashvalues  n:" , entry.getValue()[0]
+ "," + entry.getValue()[1]) ;
                        if
(entry.getKey().equals(marker.getPosition())){
                            intent.putExtra("storeId",
entry.getValue()[0]);
                            intent.putExtra("storeName",
entry.getValue()[1]);
                        }
                    }
                    startActivity(intent);

                }
            });
        }

    }

    private void setUpMap()
    {
        googleMap.setMyLocationEnabled(true);
        LocationManager locationManager =
(LocationManager)getActivity().getSystemService(Context.LOCATION_SERV
ICE);
        Criteria criteria = new Criteria();
        String provider = locationManager.getBestProvider(criteria,
false);
        Location location =
locationManager.getLastKnownLocation(provider);
        if(location != null){
            LatLng userLocation = new
LatLng(location.getLatitude(),location.getLongitude());

googleMap.moveCamera(CameraUpdateFactory.newLatLngZoom(userLocation,
15));
            googleMap.animateCamera(CameraUpdateFactory.zoomTo(15));
        }
    }

    private void setUpMapIfNeeded()
    {
        if (googleMap == null)
        {
            googleMap = mapFragment.getMap();
            if (googleMap != null)
            {
```

```
                if (stores == null)
                {
                    storesAsyncTask = new StoresAsyncTask();
                    storesAsyncTask.storesDelegate = this;
                    storesAsyncTask.execute(new Context[0]);
                }
                setUpMap();
            }
        }
    }
    @Override
    public void onCreate(Bundle bundle)
    {
        super.onCreate(bundle);
        setRetainInstance(true);

((LocationManager)getActivity().getSystemService(Context.LOCATION_SER
VICE)).requestLocationUpdates(LocationManager.GPS_PROVIDER,100,0,this
);
    }
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
viewgroup, Bundle bundle)
    {
        View v = inflater.inflate(R.layout.fragment_stores,
viewgroup, false);
        mapContainer = (FrameLayout)v.findViewById(R.id.mapStores);
        return v;
    }

    public void onLocationChanged(Location location)
    {
        googleMap.clear();
        MarkerOptions markeroptions = new MarkerOptions();
        markeroptions.position(new LatLng(location.getLatitude(),
location.getLongitude()));
        markeroptions.title("my position");
        googleMap.addMarker(markeroptions);
        googleMap.animateCamera(CameraUpdateFactory.newLatLngZoom(new
LatLng(location.getLatitude(), location.getLongitude()), 16F));
    }

    public void onProviderDisabled(String s)
    {
    }

    public void onProviderEnabled(String s)
    {
    }
    @Override
    public void onResume()
    {
        super.onResume();
        if (mapFragment != null)
        {
            setUpMapIfNeeded();
        }
    }
    @Override
    public void onStart()
    {
```

```
        super.onStart();
        mapFragment = null;
        mapFragment = new SupportMapFragment();
        FragmentTransaction fragmenttransaction =
getChildFragmentManager().beginTransaction();
        fragmenttransaction.add(R.id.mapStores, mapFragment);
        fragmenttransaction.commit();
    }

    public void onStatusChanged(String s, int i, Bundle bundle)
    {
    }

    public void onStoresTaskCompletedInterface(ArrayList arraylist)
    {
        if (arraylist != null)
        {
            stores = arraylist;
            SetUpShops();
        }
    }

    @Override
    public void onDestroyView() {
        // TODO Auto-generated method stub
        super.onDestroyView();

        try {
            Fragment fragment = (getFragmentManager()
                    .findFragmentById(R.id.map));
            FragmentTransaction ft =
getActivity().getSupportFragmentManager()
                    .beginTransaction();
            ft.remove(fragment);
            ft.commit();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

```
package asynctasks;
import android.content.Context;
import android.os.AsyncTask;
import android.util.Log;
import org.apache.http.HttpResponse;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.HashMap;
import interfaces.OnStoresTaskCompletedInterface;
import models.Store;

public class StoresAsyncTask extends AsyncTask<Context,
ArrayList<HashMap<String, String>>, String> {

    private static final String STORES_PATH =
"https://myiorderapp.herokuapp.com/api/v1/stores";
    private static final String TAG_ID = "id";
    private static final String TAG_NAME = "name";
    private static final String TAG_LATITUDE = "latitude";
    private static final String TAG_LONGITUDE = "longitude";
    public OnStoresTaskCompletedInterface storesDelegate = null;

    public StoresAsyncTask() {
        this.storesDelegate = storesDelegate;
    }

    @Override
    protected String doInBackground(Context... urls) {
        return GET(STORES_PATH);
    }

    // onPostExecute displays the results of the AsyncTask.
    @Override
    protected void onPostExecute(String result) {
        try {
            // Getting JSON Array from URL
            JSONObject json = new JSONObject(result);

            JSONArray stores = json.getJSONArray("stores");

            ArrayList<Store> storeList = new ArrayList<Store>();

            for (int i = 0; i < stores.length(); i++) {
                Store store = new Store();
                JSONObject c = stores.getJSONObject(i);
                // Storing  JSON item in a Variable
                store.setId(c.getString(TAG_ID));
                store.setName(c.getString(TAG_NAME));
                store.setLatitude(c.getString(TAG_LATITUDE));
                store.setLongitude(c.getString(TAG_LONGITUDE));
                storeList.add(store);
            }
```

```
            if (storesDelegate != null)

storesDelegate.onStoresTaskCompletedInterface(storeList);
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }


    public static String GET(String url) {
        InputStream inputStream = null;
        String result = "";
        try {
            HttpClient httpclient = new DefaultHttpClient();

            // make GET request to the given URL
            HttpResponse httpResponse = httpclient.execute(new
HttpGet(url));

            // receive response as inputStream
            inputStream = httpResponse.getEntity().getContent();

            // convert inputstream to string
            if (inputStream != null)
                result = convertInputStreamToString(inputStream);
            else
                result = "Did not work!";

        } catch (Exception e) {
            Log.d("InputStream", e.getLocalizedMessage());
        }

        return result;
    }

    private static String convertInputStreamToString(InputStream
inputStream) throws IOException {
        BufferedReader bufferedReader = new BufferedReader(new
InputStreamReader(inputStream));
        String line = "";
        String result = "";
        while ((line = bufferedReader.readLine()) != null)
            result += line;

        inputStream.close();
        return result;

    }
}
```

5.5 Οθόνη προϊόντων επιλεγμένου καταστήματος

Στη συγκεκριμένη οθόνη ο χρήστης μπορεί να βρει όλα τα διαθέσιμα προϊόντα που προσφέρει το κατάστημα και να επιλέξει αυτά που επιθυμεί για την παραγγελία του.

Εικόνα 20. Επιλογή προϊόντων από κατάστημα

Ο κώδικας είναι ο εξής:

```
package activities;
import adapters.OrderedItemsAdapter;
import models.Product;
import models.iOrder;
import mycompany.iorder.R;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
```

```
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpGet;
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.util.EntityUtils;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
import android.content.DialogInterface;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.ListAdapter;
import android.widget.ListView;
import android.widget.SimpleAdapter;
import android.widget.TextView;
import android.widget.Toast;
import android.app.Activity;

import com.google.gson.Gson;

public class Order_Activity extends Activity {

    EditText etResponse;
    TextView tvIsConnected;
    ListView list;
    private ListView favOrderedItems;
    private iOrder iorder;
    private ArrayList<Product> favOrderedItemsList;
    private String favOrderedItemsString;

    // Hashmap for ListView

    private ArrayList<HashMap<String, String>> productList = new
ArrayList<HashMap<String, String>>();

    //JSON Node Names
    private static final String TAG_PRODUCTS = "products";
    private static final String TAG_ID = "id";
    private static final String TAG_TITLE = "title";
    private static final String TAG_PRICE = "price";

    public static String shopName, shopId, productId, userId;
    public TextView mTextView2, mTextView;

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

```
        list=(ListView)findViewById(R.id.list);

        Intent intent = getIntent();

        shopName = intent.getStringExtra("storeName");
        mTextView = (TextView) findViewById(R.id.textView);
        mTextView.setText(shopName);
        shopId = intent.getStringExtra("storeId");

        iorder = ((iOrder) getApplicationContext());
        userId = iorder.getUserId();
        iorder.setStoreId(shopId);
        // call AsynTask to perform network operation on separate
thread
        new
HttpAsyncTask().execute("https://myiorderapp.herokuapp.com/api/v1/sto
res/"+shopId+"/products");

    }


    public static String GET(String url){
        InputStream inputStream = null;
        String result = "";
        try {

            // create HttpClient
            HttpClient httpclient = new DefaultHttpClient();

            // make GET request to the given URL
            HttpResponse httpResponse = httpclient.execute(new
HttpGet(url));

            // receive response as inputStream
            inputStream = httpResponse.getEntity().getContent();

            // convert inputstream to string
            if(inputStream != null)
                result = convertInputStreamToString(inputStream);
            else
                result = "Did not work!";

        } catch (Exception e) {
            Log.d("InputStream", e.getLocalizedMessage());
        }

        return result;
    }

    private static String convertInputStreamToString(InputStream
inputStream) throws IOException{
        BufferedReader bufferedReader = new BufferedReader( new
InputStreamReader(inputStream));
        String line = "";
        String result = "";
        while((line = bufferedReader.readLine()) != null)
            result += line;

        inputStream.close();
        return result;
```

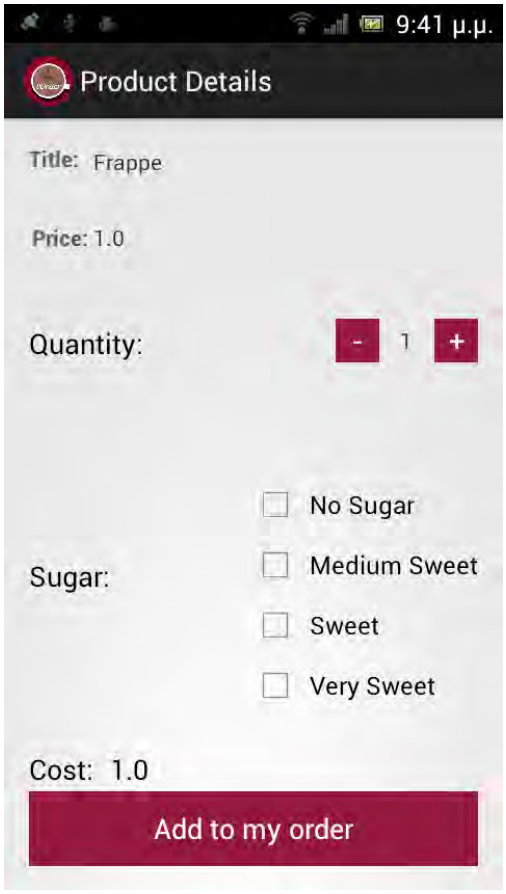
```
    }

    public boolean isConnected(){
        ConnectivityManager connMgr = (ConnectivityManager)
getSystemService(Activity.CONNECTIVITY_SERVICE);
        NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();
        if (networkInfo != null && networkInfo.isConnected())
            return true;
        else
            return false;
    }
    private class HttpAsyncTask extends AsyncTask<String, Void,
String> {
                    @Override
                    protected String doInBackground(String... urls) {

                        return GET(urls[0]);
                    }
                    // onPostExecute displays the results of the
AsyncTask.
                    @Override
                    protected void onPostExecute(String result) {
                        Toast.makeText(getBaseContext(), "Received!",
Toast.LENGTH_LONG).show();


                        try {
                                // Getting JSON Array from URL
                                JSONObject json = new
JSONObject(result);
                                String str = "";

                                JSONArray products =
json.getJSONArray("products");

                                for(int i = 0; i < products.length();
i++){

                                    JSONObject c =
products.getJSONObject(i);

                                    // Storing  JSON item in a
Variable
                                    String id = c.getString(TAG_ID);
                                    final String title =
c.getString(TAG_TITLE);
                                    final String price =
c.getString(TAG_PRICE);

                                    // Adding value HashMap key =>
value
                                    HashMap<String, String> map = new
HashMap<String, String>();
                                    map.put(TAG_ID, id);
                                    map.put(TAG_TITLE, title);
                                    map.put(TAG_PRICE, price);
                                    productList.add(map);


                                    final ListAdapter adapter = new
SimpleAdapter(Order_Activity.this, productList,
```

```
R.layout.activity_single_product,
                                                    new String[] {
TAG_ID,TAG_TITLE, TAG_PRICE }, new int[] {
                                                    R.id.id,R.id.titleText,
R.id.priceText})
                                        {
                                        };


list.setAdapter(adapter);

                                        AdapterView.OnItemClickListener
onListClick = new AdapterView.OnItemClickListener(){

                                            @Override
                                            public void
onItemClick(AdapterView<?> parent, View view,
                                                                        final
int position, long id) {

                                                runOnUiThread(new
Runnable() {
                                                    public void run() {


                                                        Intent i = new
Intent(Order_Activity.this, ProductDetails.class);

i.putExtra("title",
productList.get(position).get("title").toString());

i.putExtra("price",
productList.get(position).get("price").toString());
                                                        i.putExtra("id",
productList.get(position).get("id").toString());

                                                        startActivity(i);


                                                    }
                                                });
                                            }
                                        };


list.setOnItemClickListener(onListClick);
                                }
                            } catch (JSONException e) {
                                e.printStackTrace();
                            }
                        }
    }

    public String POST(String url, iOrder iorder) {
        InputStream inputStream = null;
        String result = "";
        try {

            // 1. create HttpClient
            HttpClient httpclient = new DefaultHttpClient()
```

```
            // 2. make POST request to the given URL
            HttpPost httpPost = new HttpPost(url);
            String json = "";

            // 3. build jsonObject
            JSONObject jsonObject = new JSONObject();
            jsonObject.accumulate("user_id", userId);
            jsonObject.accumulate("shop_id", shopId);
            jsonObject.accumulate("product_id", productId);

            // 4. convert JSONObject to JSON to String
            json = jsonObject.toString();

            // 5. set json to StringEntity
            StringEntity se = new StringEntity(json);

            // 6. set httpPost Entity
            httpPost.setEntity(se);

            // 7. Set some headers to inform server about the type of
the content
            httpPost.setHeader("Accept", "application/json");
            httpPost.setHeader("Content-type", "application/json");

            // 8. Execute POST request to the given URL
            HttpResponse httpResponse = httpclient.execute(httpPost);

            // 9. receive response as inputStream
            inputStream = httpResponse.getEntity().getContent();

            // 10. convert inputstream to string
            if (inputStream != null)
                result = convertInputStreamToString(inputStream);
            else
                result = "Did not work!";

        } catch (Exception e) {
            Log.d("InputStream", e.getLocalizedMessage());
        }

        // 11. return result
        return result;
    }

    class HttpAsyncTask2 extends AsyncTask<String, Void, String> {
        @Override
        protected String doInBackground(String... urls) {
            Log.d("HEllooooooo", "yearpdragons");

            return POST(urls[0], iorder);
        }

        // onPostExecute displays the results of the AsyncTask.
        @Override
        protected void onPostExecute(String result) {
            Toast.makeText(getBaseContext(), "Data Sent!",
Toast.LENGTH_LONG).show();
        }
    }
}
```

5.6 Οθόνη λεπτομερειών παραγγελίας χρήστη

Στη δεδομένη οθόνη ο χρήστης μπορεί να καθορίσει τις λεπτομέρειες της παραγγελίας του. Συγκεκριμένα, βλέπει το κόστος που έχει το εκάστοτε προϊόν, μπορεί να αυξομοιώσει την ποσότητα του προϊόντος, να καθορίσει την ποσότητα της ζάχαρης για το ρόφημά του.



Εικόνα 21. Καθορισμός λεπτομερειών παραγγελίας

Ο κώδικας είναι ο εξής:

```
package activities;
import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.ImageButton;
import android.widget.TextView;
import android.widget.Toast;
import java.util.ArrayList;
import models.Product;
import models.iOrder;
```

```
import mycompany.iorder.R;

public class ProductDetails extends Order_Activity {

    private int i = 1;
    private int counter = 1;
    private static String title, price, cost, productId, sugarChoice;
    private TextView mTextView1, mTextView2, mTextView3, mTextView4;
    private static double temp, myNum, finalMyNum;
    private static final String TAG_ID = "id";
    private Button button, button2;
    private Button addOrder;
    private CheckBox noSugar, medSweet, sweet, verySweet;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_product_details);
        medSweet = (CheckBox) findViewById(R.id.mediumSweet);
        sweet = (CheckBox) findViewById(R.id.sweet);
        verySweet = (CheckBox) findViewById(R.id.verySweet);
        noSugar = (CheckBox) findViewById(R.id.noSugar);
        mTextView1 = (TextView) findViewById(R.id.counter);
        mTextView2 = (TextView) findViewById(R.id.titleProduct);
        mTextView3 = (TextView) findViewById(R.id.priceProduct);
        mTextView4 = (TextView) findViewById(R.id.costNum);
        button = (Button) findViewById(R.id.plusButton);
        button2 = (Button) findViewById(R.id.minusButton);
        addOrder = (Button) findViewById(R.id.addToOrder);
        Intent intent = getIntent();
        title = intent.getStringExtra("title");
        productId = intent.getStringExtra("id");
        price = intent.getStringExtra("price");
        cost = intent.getStringExtra(("price"));
        mTextView1.setText(Integer.toString(i));
        mTextView4.setText(cost);
        mTextView2.setText(title);
        mTextView3.setText(price);
        myNum = Double.parseDouble(mTextView4.getText().toString());
        finalMyNum = myNum;

        button.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {

                // Perform action on click
                i = (i + 1);
                counter = i;

                temp = Double.parseDouble(Integer.toString(i)) *
finalMyNum;
                mTextView1.setText(Integer.toString(i));
                mTextView4.setText(Double.toString(temp) + "€");
            }
        });

        button2.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                // Perform action on click
                if (i > 1) {
                    i = i - 1;
```

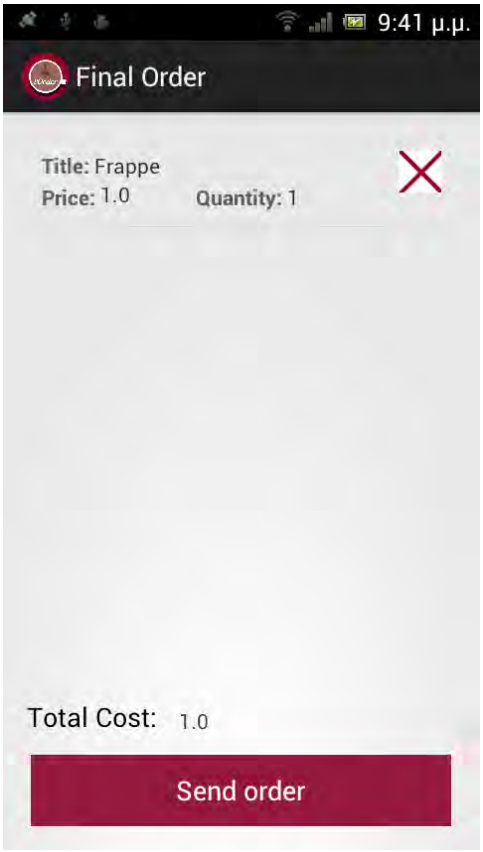
```
                    counter = i;
                    mTextView1.setText(Integer.toString(i));
                    double temp = i;
                    temp = temp * finalMyNum;
                    mTextView4.setText(Double.toString(temp) + "€");
                }
            }
        });

        noSugar.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                noSugar();
            }
        });
        medSweet.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                mediumSugar();
            }
        });
        verySweet.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                verySweetSugar();
            }
        });
        sweet.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                sweetSugar();
            }
        });

        addOrder.setOnClickListener(new View.OnClickListener() {

            public void onClick(View v) {
                Add_to_my_order();
            }
        });
    }

    @Override
    protected void onDestroy() {

        super.onDestroy();
        Toast.makeText(getApplicationContext(), "16. onDestroy()",
Toast.LENGTH_SHORT).show();

        title = null;
        price = null;

        mTextView2.setText(title);
        mTextView3.setText(price);
        mTextView4.setText("");

    }

    @Override
    public void onRestart() {
        super.onRestart();
```

```
        Toast.makeText(getApplicationContext(), "5. onRestart()",
Toast.LENGTH_SHORT).show();
    }

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

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            return true;
        }

        return super.onOptionsItemSelected(item);
    }


    //connect ProductDetails with FinalOrder Activity
    public void Add_to_my_order() {
        Intent intent = new Intent(this, FinalOrder.class);

        ArrayList<Product> tempList;
        iOrder iorder = ((iOrder) getApplicationContext());
        Product productToOrder = new Product();
        productToOrder.setDetails(sugarChoice);
        productToOrder.setQuantity(Integer.toString(counter));
        productToOrder.setPrice(price);
        productToOrder.setTitle(title);
        productToOrder.setProductId(productId);
        if (iorder.getProducts() == null)
            tempList = new ArrayList<Product>();
        else
            tempList = iorder.getProducts();
        tempList.add(productToOrder);
        iorder.setProducts(tempList);

        startActivity(intent);
    }

    public void Back(View view) {
        super.onBackPressed();
    }


    public void noSugar() {
        if (noSugar.isChecked()) {
            sugarChoice = "No Sugar";
            medSweet.setChecked(false);
            sweet.setChecked(false);
            verySweet.setChecked(false);
        }
    }
```

```
    public void mediumSugar() {
        if (medSweet.isChecked()) {
            sugarChoice = "Medium Sugar";
            sweet.setChecked(false);
            verySweet.setChecked(false);
            noSugar.setChecked(false);
        }
    }

    public void sweetSugar() {
        if (sweet.isChecked()) {
            sugarChoice = "Sweet Sugar";
            medSweet.setChecked(false);
            verySweet.setChecked(false);
            noSugar.setChecked(false);
        }
    }

    public void verySweetSugar() {
        if (verySweet.isChecked()) {
            sugarChoice = "Very Sweet Sugar";
            medSweet.setChecked(false);
            sweet.setChecked(false);
            noSugar.setChecked(false);
        }
    }

}
```

5.7 Οθόνη τελικής παραγγελίας

Εδώ ο χρήστης βλέπει συγκεντρωτικά την παραγγελία του πριν τη στείλει στο κατάστημα. Μπορεί να δει τα προϊόντα που έχει επιλέξει μέχρι στιγμής, το κόστος ανά προϊόν, την ποσότητα ανά προϊόν και το συνολικό κόστος της παραγγελίας του. Αν θέλει να αφαιρέσει κάποιο προϊόν από την παραγγελία του, μπορεί να το πραγματοποιήσει πατώντας το κουμπί X. Τέλος, πατώντας το κουμπί "Send Order" αποστέλνει την παραγγελία του στο κατάστημα.



Εικόνα 22. Τελική παραγγελία

Ο κώδικας είναι ο εξής:

```
package activities;
import android.app.Activity;
import android.content.Intent;
import android.database.DataSetObserver;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.ListView;
```

```
import android.widget.TextView;
import android.widget.Toast;
import com.google.gson.Gson;
import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
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.util.EntityUtils;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
import java.io.IOException;
import java.io.InputStream;
import java.util.ArrayList;
import java.util.List;
import adapters.OrderedItemsAdapter;
import models.Product;
import models.iOrder;
import mycompany.iorder.R;

public class FinalOrder extends Activity {

    private iOrder iorder;
    private ArrayList<Product> orderedItemsList;
    private ListView orderedItems;
    private OrderedItemsAdapter orderedItemsAdapter;
    private TextView mTextView;
    private double totalCost;
    private String finalOrderString;
    private String shopId;


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

        mTextView = (TextView) findViewById(R.id.totalCost);
        iorder = ((iOrder) getApplicationContext());
        orderedItems = (ListView) findViewById(R.id.orderedItems);
        orderedItemsList = new ArrayList<Product>();
        orderedItemsList = iorder.getProducts();
        orderedItemsAdapter = new OrderedItemsAdapter(this,
orderedItemsList);
        DataSetObserver dataSetObserver = new DataSetObserver() {
            @Override
            public void onChanged() {
                super.onChanged();
                Ref();
            }
        };
        orderedItemsAdapter.registerDataSetObserver(dataSetObserver);
        orderedItems.setAdapter(orderedItemsAdapter);
        for (Product p : orderedItemsList) {
            totalCost += Double.parseDouble(p.getPrice()) *
Double.parseDouble(p.getQuantity());
```

```
        }
//          totalCost = OrderedItemsAdapter.refreshTotalCost();
        mTextView.setText(Double.toString(totalCost));
        Button addToStore = (Button) findViewById(R.id.button);
        addToStore.setOnClickListener(new View.OnClickListener() {

            public void onClick(View v) {
                new
HttpAsyncTask().execute("https://myiorderapp.herokuapp.com/api/v1/ord
ers/new_order");
                Intent intent = new Intent(FinalOrder.this,
MapsActivity.class);
                startActivity(intent);
            }
        });

    }

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

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            return true;
        }
        return super.onOptionsItemSelected(item);
    }

    public String POST(String url, iOrder iorder) {
        InputStream inputStream = null;
        String result = "";

        // Create a new HttpClient and Post Header
        HttpClient httpclient = new DefaultHttpClient();
        HttpPost httppost = new HttpPost(url);

        JSONArray productArray = new JSONArray();

            // Add your data
            List<NameValuePair> nameValuePairs = new
ArrayList<NameValuePair>(1);
            nameValuePairs.add(new BasicNameValuePair("user_id",
iorder.getUserId()));
            nameValuePairs.add(new BasicNameValuePair("store_id",
iorder.getStoreId()));
            nameValuePairs.add(new BasicNameValuePair("auth_token",
iorder.getAuthenticationToken()));
            nameValuePairs.add(new BasicNameValuePair("total",
Double.toString(totalCost)));
            for (Product p : orderedItemsList) {
                JSONObject productJson = new JSONObject();
```

```
                try {
                     productJson.put("id", p.getProductId());
                     productJson.put("quantity", p.getQuantity());
                     productJson.put("details", p.getDetails());
                }catch(JSONException e){
                    e.printStackTrace();
                }
                productArray.put((productJson));
            }
            nameValuePairs.add(new
BasicNameValuePair("order_details", new
Gson().toJson(productArray.toString())));
        try {
            httppost.setEntity(new
UrlEncodedFormEntity(nameValuePairs));
            HttpResponse response = httpclient.execute(httppost);
            HttpEntity entity = response.getEntity();
            String content = EntityUtils.toString(entity);
            finalOrderString = content;
            StringEntity se = new StringEntity(finalOrderString);
            httppost.setEntity(se);
        }catch (IOException e) {
            // TODO Auto-generated catch block
        }


        return result;
    }


    class HttpAsyncTask extends AsyncTask<String, Void, String> {
        @Override
        protected String doInBackground(String... urls) {

            return POST(urls[0], iorder);
        }

        // onPostExecute displays the results of the AsyncTask.
        @Override
        protected void onPostExecute(String result) {
            Toast.makeText(getBaseContext(), "Data Sent!",
Toast.LENGTH_LONG).show();
        }
    }

    public void Ref() {
        Double newCost = 0.0;
        for (Product p : orderedItemsList) {
            newCost += Double.parseDouble(p.getPrice()) *
Double.parseDouble(p.getQuantity());
        }
//        totalCost = OrderedItemsAdapter.refreshTotalCost();
        mTextView.setText(Double.toString(newCost));
    }
}
```

6. Συμπεράσματα

Από τα παραπάνω κεφάλαια καταλήγουμε ότι η πλατφόρμα της Google, Android, είναι και παρέχει τα εξής:

1. Είναι ένα ολοκληρωμένο λειτουργικό σύστημα για κινητές συσκευές
2. Αποτελεί μια ολοκληρωμένη πλατφόρμα ανάπτυξης εφαρμογών και προσφέρει πληθώρα εργαλείων και μεθόδων.
3. Παρέχει πολλά και τεκμηριωμένα κείμενα ανάπτυξης εγαρμογών και του τρόπου λειτουργίας του συστήματος.
4. Μπορεί να καλύψει τις ανάγκες τόσο των μέσων όσο και των εξειδικευμένων χρηστών.
5. Οι κατασκευαστές των κινητών συσκευών έχουν τη δυνατότητα να προσαρμόσουν ανάλογα με τις ανάγκες τους το λειτουργικό σύστημα Android χωρίς κόστος.

Μέσα από την ενασχόληση των παραπάνω τεχνολογιών, που μελετήσαμε στα προηγούμενα κεφάλαια, καταλήγουμε στην επίτευξη του στόχου μας. Στόχος μας ήταν, η εφαρμογή iOrder, που αναπτύχθηκε, να λειτουργεί γενικότερα σε συσκευές που υποστηρίζονται από το λειτουργικό Android. Συνεπώς, η εφαρμογή δοκιμάστηκε τόσο σε εικονικές συσκευές όσο και σε πραγματικά τηλέφωνα, όπως περιγράψαμε παραπάνω. Επιπλέον, το iOrder θα μπορούσε να υπάρξει με μεγάλη επιτυχία στο Google Play.

Βιβλιογραφία

Βιβλιογραφικές πηγές

1) **Paul Deitel, Harvey Deitel, Abbey Deitel, Michael Morgano** *2011. Android forProgrammers, An App-Driven Approach. 1st Edition, Prentice Hall, USA.*
2) **Erik T. Ray** *2003. Learning XML. 2nd Edition, O'Reilly, USA.*
3) **Peter Morville, Louis Rosenfeld** *2006. Information Architecture for thw World Wide Web.3rd Edition, O'Reilly, USA.*
4) **Paul Deitel, Harvey Deitel** *2011. Java, How to Program. 9th Edition, Prentice Hall, USA.*
5) **Bill McCarty, Stephen Gilbert** *1998. Object-oriented Design in Java. 1st Edition, Waite Group Press, USA.*
6) **Reto Meier** *2012. Professional Android 4 Application Development. Updated Edition, John Wiley & Sons, USA.*
7) **Zigurd Mednieks, Laird Dornin, G. Blake Meike, MasumiNakamura** *2011. Programming Android: Java Programming for the New Generation of Mobile Devices. 1st Edition, O'Reilly,USA.*
8) **Wei-Meng Lee** *2011. Beginning Android Application Development. 1st Edition, John Wiley & Sons, USA.*
9) **Wei-Meng Lee** *2012. Beginning Android 4 Application Development. 1st Edition, John Wiley & Sons, USA.*

Διαδικτυακές πηγές

1) **Google Inc.**, *www.google.com*
2) **Stack Overflow**, *www.stackoverflow.com*
3) **Google Android Developers**, *developer.android.com*
4) **Google Inc. IO Sessions**,*developers.google.com/events/io/sessions*
5) **Wikipedia**, *www.wikipedia.com*