ΑΛΕΞΑΝΔΡΕΙΟ ΤΕΙ ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Ανάπτυξη εφαρμογής παροχής πληροφοριών και υπηρεσιών του τμήματος Πληροφορικής στην πλατφόρμα Android

Των φοιτητών
Βαρσάμη Ιωάννη &
Παναγιωτόπουλου Βασίλη
Αρ. Μητρώου: 07/3189 & 08/3335

Επιβλέπων καθηγητής

Θεσσαλονίκη 2012

Πρόλογος

Η παρούσα πτυχιακή εργασία πραγματοποιήθηκε στο Αλεξάνδρειο Τεχνολογικό Ίδρυμα Θεσσαλονίκης, στο τμήμα Πληροφορικής της Σχολής Τεχνολογικών Εφαρμογών. Είναι η εργασία δυο φοιτητών του τμήματος με σκοπό την ανάπτυξη, σε mobile περιβάλλον (πλατφόρμα Android), μίας εφαρμογής για το ίδιο το τμήμα και τις υπηρεσίες που προσφέρει στους φοιτητές του.

Οι κυριότεροι λόγοι που μας οδήγησαν να επιλέξουμε αυτό το θέμα ήταν οι εξής: η παρουσία του τμήματος μας σε μια πλατφόρμα κινητής τηλεφωνίας, η προσπάθεια και η εξοικείωση με το αντικείμενο του προγραμματισμού, και φυσικά, η τεράστια επέκταση των «*έξυπνων κινητών τηλεφώνων*» (smartphones) και των αντίστοιχων εφαρμογών που αξιοποιούν τα πλεονεκτήματα τους, μέσα σε ένα συνεχώς ανεπτυγμένο τεχνολογικό πεδίο.

Στην παρούσα, λοιπόν, εργασία, αρχικά, περιγράφουμε την πλατφόρμα του λειτουργικού συστήματος για να εισάγουμε τον αναγνώστη σε ένα αρχικό τεχνολογικό πλαίσιο. Έπειτα, περιγράφουμε το βασικό εργαλείο ανάπτυξης που χρησιμοποιήσαμε για την ανάπτυξη και τον προγραμματισμό της. Τέλος, αναλύουμε και εξηγούμε το τεχνολογικό υπόβαθρο, τα χαρακτηριστικά, και τις λειτουργίες που έχουμε ενσωματώσει στην εφαρμογή.

Ευχαριστούμε τον καθηγητή μας κ. Παναγιώτη Σφέτσο για την ανάθεση της πτυχιακής εργασίας και για την βοήθεια που μας έδωσε κατά την διάρκεια της. Επίσης, σημαντική ήταν η προφορική βοήθεια συμφοιτητών του τμήματος μας για το επιθυμητό περιεχόμενο και τις λειτουργίες της εφαρμογής.

Περίληψη

Η παρούσα πτυχιακή εργασία αφορά την ανάπτυξη εφαρμογής κινητής τηλεφωνίας στην πλατφόρμα Android για το τμήμα Πληροφορικής του Αλεξάνδρειου Τεχνολογικού Ιδρύματος Θεσσαλονίκης.

Κατόπιν συζητήσεων και μελετών για θέματα τεχνολογικά, περιεχομένου, λειτουργικότητας, σχεδιασμού, και αναγκών της εφαρμογής, καταλήξαμε σε μια συγκεντρωτική, διατεταγμένη ανάλογα με τις ανάγκες του χρήστη, διεπαφή χρήστη. Πιο συγκεκριμένα, απευθύνεται σε ενδιαφερόμενους, εντός και εκτός σχολής, που θέλουν να μάθουν για το τμήμα και τις βασικότερες λειτουργίες του. Επίσης, παρέχει στους φοιτητές του τμήματος πρόσβαση στις βασικότερες διαδικτυακές του υπηρεσίες. Πιλοτικά, παρέχονται δύο επιπλέον υπηρεσίες προς κάθε χρήστη της εφαρμογής: συνομιλία μεταξύ των χρηστών και ενημέρωση για την πληρότητα της γραμμής αστικής συγκοινωνίας 52.

Με χρήση του περιβάλλοντος ανάπτυξης Eclipse, αναπτύξαμε, στην έκδοση 2.1 του λειτουργικού συστήματος Android, μία εφαρμογή χωρισμένη σε τέσσερις (4) καρτέλες. Μία αρχική οθόνη, μία καρτέλα που περιέχει πληροφορίες για το τμήμα Πληροφορικής, μία καρτέλα με παρεχόμενες διαδικτυακές υπηρεσίες του τμήματος, και, τέλος, μία καρτέλα με επιπρόσθετες πιλοτικές υπηρεσίες.

Αναλυτικότερα, στις πληροφορίες του τμήματος ενσωματώσαμε στοιχεία για το τμήμα, τις σπουδές, το πτυχίο, την διοικητική και εκπαιδευτική υποστήριξη προς τον φοιτητή, την οργάνωση των σπουδών, φωτογραφίες, κ.α. Στις υπηρεσίες προς τους φοιτητές, ενσωματώσαμε τις ανακοινώσεις του τμήματος, το σύνολο των βαθμολογιών του φοιτητή, τον υπολειπόμενο αριθμό μαθημάτων για το πτυχίο, την δήλωση μαθημάτων σε ηλεκτρονική μορφή (pdf), τα στοιχεία επικοινωνίας των καθηγητών του τμήματος και τα στοιχεία του φοιτητή.

Το σύνολο του περιεχομένου της εφαρμογής είναι διαθέσιμο χωρίς να απαιτείται καμία πιστοποίηση από τον χρήστη. Εξαίρεση αποτελεί η τρίτη καρτέλα, όπου οι φοιτητικές υπηρεσίες απαιτούν τα στοιχεία λογαριασμού του διαδικτυακού συστήματος ενημέρωσης (Hydra) και του συστήματος δηλώσεων του τμήματος (Pithia).

Περίληψη στα Αγγλικά (Abstract)

The present study is about the development of a mobile application based on the Android platform which concerns the Department of Informatics of Alexander Technological Educational Institute.

Following discussions on technological, functional, and design issues we ended up with a summarized user interface, arranged according to the user's needs. More specifically, it is addressing everyone, being or not student of the department, providing information about the department and its functions. Moreover, it provides to the students the basic online services they need. Also, there are two additional services for every user; the one is about chatting among the users and the other about informing on the occupancy of the line 52 of the public transportation.

Using the software development environment Eclipse, based on the 2.1 version of the Android operating system, we have developed an application divided in four tabs. Firstly, the home tab, and then, a tab which is providing all the information about the department, a tab with the basic web services of the department for the students and, finally, a tab consisting on the additional testing services we have built.

In more detail, the information about the department include some figures about the department and its studies, the degree, the administrative and educational support for the student, the structure of studies, pictures, etc. The information for the students include the announcements of the department, the marks and the remaining subjects, a .pdf file of the subjects taken this semester, the figures of the student, and, finally, contact information of all the department’s professors.

The content of the application is available for anyone to read without the need of any validation method. However, a student must add the personal login and password in order to have access to the services of the third tab.

Ευρετήριο Περιεχομένων

Ευρετήριο Σχημάτων και Πινάκων

Εισαγωγή

Η επικρατέστερη ιδέα για το θέμα την πτυχιακής εργασίας προέκυψε μετά από μία εβδομάδα συζητήσεων, κυρίως με βάση το ενδιαφέρον που μπορεί αυτή να έχει για εμάς. Το ίδιο το πεδίο του προγραμματισμού εφαρμογών για λειτουργικά συστήματα κινητής τηλεφωνίας προσφέρει ένα τεράστιο εύρος επιλογών οπότε, πριν μπούμε σε οποιαδήποτε τεχνολογική λεπτομέρεια, έπρεπε να αποφασίσουμε ποια θα είναι η κατάλληλη επιλογή.

Μεταξύ των διαφόρων ιδεών υπήρχαν προτάσεις για διακομιστές διαδικτυακών υπηρεσιών (μουσικής, βίντεο, πολυμεσικές εφαρμογές γενικότερα), για δικής μας προσαρμογής φυλλομετρητή (web browser), για έκδοση για κινητά γνωστών διαδικτυακών υπηρεσιών, για πρόγραμμα διαχείρισης λήψεων (download manager), για εφαρμογή διαδικτυακής συνομιλίας (chat client) κ.α. Τελικώς, καταλήξαμε σε μία πιο ενδιαφέρουσα και ίσως περισσότερο πρωτότυπη επιλογή: την ανάπτυξη εφαρμογής για το τμήμα μας, συγκεκριμένα, το τμήμα Πληροφορικής του Αλεξάνδρειου Τεχνολογικού Ιδρύματος Θεσσαλονίκης.

Υπήρχε η γνώση παρόμοιας εφαρμογής πανεπιστημίου του εξωτερικού από όπου και μπόρεσε η ιδέα να αναλυθεί εκτενέστερα από μια απλή αναφορά. Επίσης, κατανοήσαμε την σημαντικότητα της επιλογής αυτής καθότι η πτυχιακή εργασία θα μπορούσε, μέσω της επαφής της με το ίδιο το τμήμα, να δώσει αφορμή και σε άλλους φοιτητές να ασχοληθούν με αυτόν τον τομέα του προγραμματισμού. Πιστεύουμε ότι, από την τεράστια ανάπτυξη που έχει επιτευχθεί στα κινητά λειτουργικά συστήματα, θα πρέπει να μάθουμε την σημαντικότητα που του αναλογεί στο πεδίο της πληροφορικής και να ενσωματώσουμε ανάλογες προσπάθειες στους διαγωνισμούς και στο πρόγραμμα μας.

Την ιδέα ενίσχυσε επίσης η άποψη ότι οι υπηρεσίες του τμήματος θα έπρεπε να είναι περισσότερο λειτουργικές και εύχρηστες, αν όχι συνολικά σε μια μόνο διαδικτυακή πύλη, σε κάτι πιο συγκεντρωτικό. Εδώ επίσης είχε επιρροή και η εμπειρία από αντίστοιχα συστήματα πανεπιστημίων του εξωτερικού από την οποία έγινε αντιληπτή η χειρότερη κατάσταση στην οποία βρισκόμαστε.

Έτσι, εφόσον τα παραπάνω συζητήθηκαν, καταλήξαμε στο θέμα της εφαρμογής του τμήματος μας και ξεκινήσαμε να ασχολούμαστε με τις λειτουργίες και τις ειδικότερες λεπτομέρειες που την αφορούν.

Από τις πρώτες επιλογές που έπρεπε να γίνουν ήταν αυτή της πλατφόρμας ανάπτυξης, του λειτουργικού συστήματος, δηλαδή, για το οποίο θα προγραμματίζαμε την εφαρμογή. Τη δεδομένη στιγμή, τα κυρίαρχα λειτουργικά συστήματα ήταν δύο: το Android της Google και το iOS της Apple. Τα υπόλοιπα γνωστά λειτουργικά συστήματα, όπως BlackBerry OS της RIM, Bada της Samsung, Symbian OS της Nokia, Windows Phone της Microsoft, WebOS της Hewlett-Packard, ήταν εξαρχής δευτερεύουσες επιλογές γιατί βασική παράμετρος ήταν η ευρεία χρήση της εφαρμογής. Η πιθανή ευκολία ανάπτυξης σε μία λιγότερο δημοφιλή πλατφόρμα θα εκμηδενίζονταν από τον μικρό αριθμό χρηστών της. Κάνοντας μια προσπάθεια που απευθύνεται στους φοιτητές και σε όσο το δυνατόν περισσότερους αποδέκτες, επιλέγουμε να μην θέτουμε περισσότερους των φυσικών περιορισμούς.

Ανάμεσα στα λειτουργικά συστήματα Android και iOS επιλέξαμε το πρώτο για τους εξής λόγους: υπάρχουσα εξοικείωση με την γλώσσα προγραμματισμού Java -που χρησιμοποιείται από το Android-, δωρεάν χρήση του λογισμικού –σε αντίθεση με το επί πληρωμή iOS-, ανάπτυξη σε τυπικούς προσωπικούς υπολογιστές –σε αντίθεση με τον συγκεκριμένο τύπο υπολογιστή (Macintosh) που απαιτεί το iOS-, η μεγαλύτερη εξάπλωση των συσκευών Android καθώς και η προσβασιμότητα των χρηστών σε πολλές συσκευές, όλων σχεδόν των οικονομικών επιπέδων.

Έχοντας λύσει το ζήτημα του λειτουργικού συστήματος επιλέξαμε το περιβάλλον ανάπτυξης Eclipse το οποίο είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης και αποτελεί ελεύθερο λογισμικό ανοιχτού κώδικα. Με την χρήση των απαραίτητων επεκτάσεων, παρέχει την δυνατότητα ανάπτυξης και προσομοίωσης εφαρμογών για την πλατφόρμα Android. Επίσης, δίνει την δυνατότητα για διάφορες επιπρόσθετες επεκτάσεις που βοηθούν τον προγραμματισμό.

Επόμενο βήμα των συζητήσεων ήταν η διεπαφή χρήστη (user interface), η φιλοσοφία, και η λειτουργικότητα της εφαρμογής. Υπάρχουν πολλές δυνατότητες όσον αφορά το πως θα δομηθεί μια διεπαφή: ακολουθία μενού, σειρές εικονιδίων, κεντρικά τοποθετημένη κουμπιά με αντιστοίχηση σε φόρμες περιεχομένων, καθώς και επιλογές για τον αριθμό των καρτελών που θα υπάρχουν. Κατόπιν αναζήτησης και συζήτησης καταλήξαμε σε μία διάταξη χωρισμένη σε τέσσερις καρτέλες καθεμία από τις οποίες θα αφορά ξεχωριστή κατηγορία ύλης.

Πιο συγκεκριμένα, η πρώτη καρτέλα θα ήταν η αρχική οθόνη της εφαρμογής, έχοντας, πιθανώς, λογότυπα και κείμενο που την χαρακτηρίζουν. Επίσης, μπορούσε να ενσωματωθεί κι άλλη λειτουργία στο κάτω μέρος της καρτέλας αφού πρώτα ενσωματωθούν τα παραπάνω. Απαραίτητα χαρακτηριστικά για την αρχική οθόνη ορίστηκαν η αναφορά στο Ίδρυμα (Αλεξάνδρειο Τεχνολογικό Ίδρυμα Θεσσαλονίκης) και ενσωμάτωση του αντίστοιχου λογότυπου, αναφορά στο τμήμα μας (Τμήμα Πληροφορικής), το όνομα της εφαρμογής (καταλήξαμε στο *it.teithe.app*) και ένα κατάλληλο λογότυπο που θα δημιουργούσαμε εμείς.

Στη δεύτερη καρτέλα, ή αλλιώς, την καρτέλα των πληροφοριών, θα εισαγάγαμε όλες τις πληροφορίες που θα αποφασίζαμε και θα αφορούσαν αποκλειστικά το τμήμα Πληροφορικής. Ειδικότερα, οποιεσδήποτε χρήσιμες πληροφορίες που θα κρίναμε πως είναι απαραίτητες για την πληροφόρηση ενός ενδιαφερομένου, είτε φοιτεί είτε όχι στο δικό μας τμήμα. Η εφαρμογή μπορεί κάλλιστα να είναι ένα εργαλείο για τον ίδιο τον φοιτητή που έχει άμεσο ενδιαφέρον για το τμήμα και τις υπηρεσίες του καθώς επίσης και για τον εξωτερικό ενδιαφερόμενο που θέλει να μάθει για το τμήμα, το πρόγραμμα του, για προσωπικούς λόγους. Η επιλογή συγκεκριμένων πληροφοριών που θα ενσωματώναμε στη καρτέλα αυτή θα γινόταν μεταγενέστερα, καθώς την αναπτύσσαμε προγραμματιστικά, αν και υπήρχε μια αρχική ιδέα των περιεχομένων της.

Στη τρίτη καρτέλα, ή αλλιώς, την καρτέλα των υπηρεσιών, θα εισαγάγαμε τις διαδικτυακές υπηρεσίες που προσφέρει το τμήμα μας. Αυτή η καρτέλα αφορά τους φοιτητές του ίδιου του τμήματος και η πρόσβαση, λοιπόν, περιορίζεται μόνο στους χρήστες που έχουν πιστοποιήσει τα στοιχεία τους (συγκεκριμένα, login και password των αντίστοιχων υπηρεσιών). Σε αυτό το θέμα υπήρξε αρκετή συζήτηση

στο ποια θα είναι ακριβώς τα περιεχόμενα της καρτέλας. Κρίνοντας με βάση την εμπειρία μας, τις δικές μας απόψεις και τις γνώμες άλλων φοιτητών, αποφασίσαμε να ενσωματώσουμε τις δύο σημαντικότερες διαδικτυακές υπηρεσίες του τμήματος: το σύστημα ανακοινώσεων Hydra και το σύστημα δήλωσης μαθημάτων Pithia. Το πρώτο είναι από τα πλέον βασικά και πολυσύχναστα συστήματα του τμήματος οπότε οι ανακοινώσεις αυτές θα ήταν σίγουρα μέρος της εφαρμογής μας. Το δεύτερο, περιλαμβάνει τις δηλώσεις των φοιτητών, τις βαθμολογίες κ.α. και, αν και δεν είναι όσο πολυσύχναστο είναι το σύστημα ανακοινώσεων, είναι εξίσου σημαντικό και χρήσιμο. Με βάση αυτά τα δύο συστήματα μαζέψαμε αρκετές ιδέες για το περιεχόμενο της καρτέλας. Το τμήμα, προφανώς, προσφέρει πολλές άλλες υπηρεσίες και δυνατότητες αλλά, σαν αρχική βάση, μείναμε στα προαναφερθέντα δύο. Αφήσαμε, φυσικά, ανοιχτό το ενδεχόμενο να προσθέσουμε αργότερα υπηρεσίες από άλλα συστήματα του τμήματος.

Στην τέταρτη και τελευταία καρτέλα, την καρτέλα των επιπρόσθετων υπηρεσιών, είχαμε σκοπό να ενσωματώσουμε διάφορες ιδέες που μας προέκυπταν και είχαν να κάνουν με υπηρεσίες που δεν παρέχονταν από το Ίδρυμα ή το τμήμα. Δηλαδή, περισσότερο πρωτότυπες ιδέες που θέλαμε να αναπτύξουμε και οι οποίες θα βοηθούσαν τον τελικό χρήστη. Επηρεαζόμενοι από άλλες υπάρχουσες εφαρμογές κινητής τηλεφωνίας, καταλήξαμε, με τον καιρό, σε δύο ιδέες. Η μία ήταν η δυνατότητα των χρηστών να συνομιλούν μεταξύ τους με δημόσια μηνύματα, στα πρότυπα ενός καναλιού IRC (διαδικτυακή συνδιάλεξη). Αυτό θα διευκόλυνε φοιτητές με κοινά ενδιαφέροντα και προβλήματα, είτε του πεδίου μας είτε προσωπικά. Υπάρχουν, φυσικά, πολλές εφαρμογές κοινωνικής δικτύωσης ή ηλεκτρονικής αλληλογραφίας αλλά πιστεύουμε ότι οι συνομιλίες μέσα στην ίδια την εφαρμογή θα διευκολύνουν συζητήσεις του αντικειμένου μας και θα ήταν, αν μη τι άλλο, ενδιαφέρουσες. Η άλλη υπηρεσία που σκεφτήκαμε ήταν η δυνατότητα ενημέρωσης και καταχώρισης της πληρότητας της γραμμής αστικής συγκοινωνίας που αφορά το ΑΤΕΙΘ, την γραμμή 52 του ΟΑΣΘ. Η συγκεκριμένη υπηρεσία ήταν και μία προγραμματιστική πρόκληση αλλά και μία χρήσιμη εφαρμογή για τους φοιτητές που μετακινούνται συχνά με την αστική συγκοινωνία.

Ο χρήστης θα μπορεί να ενημερωθεί για την κατάσταση στην γραμμή 52, αν είναι δηλαδή γεμάτο (αντίστοιχη πληρότητα 100%) ή άδειο και κατά πόσο, ώστε να γνωρίζει τις ώρες και τις συχνότητες πολυκοσμίας στο αστικό λεωφορείο. Η εμπειρία μας έχει δείξει ότι είναι μια σημαντική παράμετρος, άλλωστε, η διαδρομή από τον σταθμό έως τη Σίνδο, όπου είναι το ΑΤΕΙΘ, είναι χρονοβόρα και, σε κάποιες περιπτώσεις, κουραστική.

Όλα τα παραπάνω ενσωματώθηκαν σε μία διάταξη που έχεις ως εξής: τέσσερις καρτέλες με την πλοήγηση σε αυτές να βρίσκεται στο κάτω μέρος της οθόνης και η καθεμία να έχει το αντίστοιχο εικονίδιο και κείμενο. Η διάταξη των ίδιων των καρτελών διαμορφώθηκε σε μορφή πλεγμάτων εικονιδίων (grid icons), διάταξη η οποία είναι αρκετά δημοφιλής ώστε να είναι, πλέον, οικεία στο χρήστη.

Αφού ολοκληρώσαμε τις αποφάσεις για την δομή και την λειτουργία της εφαρμογής, ξεκινήσαμε την πρακτική ενασχόληση. Εγκαταστήσαμε το λογισμικό Eclipse και προσθέσαμε τα αναγκαία πρόσθετα: αυτό του Android, για την ανάπτυξη της εφαρμογής, και το πρόσθετο του GitHub που προσφέρει την δυνατότητα εξ αποστάσεως συγγραφής κώδικα από πολλούς προγραμματιστές. Έτσι, δουλεύαμε ταυτόχρονα, κατανέμοντας τις εργασίες σε δύο μέρη. Το GitHub είναι μία διαδικτυακή υπηρεσία φιλοξενίας για project ανάπτυξης λογισμικού που βασίζεται στο Git, μία πλατφόρμα που βοηθά την κατανεμημένη εργασία.

Για εξάσκηση στον προγραμματισμό Android διαβάσαμε και συμβουλευτήκαμε το δημοφιλές βιβλίο *Hello, Android: Introducing Google's Mobile Development Platform* του Ed Burnette. Λόγω προσωπικού ενδιαφέροντος, είχαμε επίσης λίγη εμπειρία που ήταν χρήσιμη στο ξεκίνημα της συγγραφής. Η κύρια πηγή πληροφοριών και λύσεων στα συνεχή προβλήματα που προέκυπταν στην πορεία ήταν το διαδίκτυο, δηλαδή ιστοσελίδες και forum συζητήσεων οπού αναλύονται και επιλύονται θέματα προγραμματισμού Android. Επίσης, άλλη βασική πηγή πληροφοριών ήταν τα κείμενα τεκμηρίωσης του λειτουργικού συστήματος (αναλυτική διαδικτυακή επεξήγηση όλων των λειτουργιών της γλώσσας προγραμματισμού) και τα παραδείγματα κώδικα που παρέχει και τα οποία εξετάσαμε μέσα από το περιβάλλον Eclipse.

Όσον αφορά τον προγραμματισμό στην γλώσσα Java, το τμήμα Πληροφορικής έχει τρία μαθήματα αφιερωμένα στην εκμάθηση του. Έτσι, ήμασταν εξοικειωμένοι με τα απαιτούμενα του προγραμματισμού εφαρμογών Android, τουλάχιστον στο μέτρο που αφορά την γενικότερη φιλοσοφία και χρήση της γλώσσας Java.

Με αυτά τα εφόδια, αναπτύξαμε τον κεντρικό σκελετό της εφαρμογής, δομημένο όπως προαναφέραμε. Οι εργασίες ήταν κατανεμημένες και είχαμε εβδομαδιαίες συναντήσεις για συζητήσεις που αφορούσαν εξειδικευμένες λύσεις πάνω στην υλοποίηση των λειτουργιών της εφαρμογής.

Για την δεύτερη καρτέλα, στην οποία ενσωματώσαμε τις πληροφορίες για το τμήμα, έπρεπε να αποφασίσουμε αν τα δεδομένα θα ήταν στατικά (αποθηκευμένα τοπικά στην ίδια την εφαρμογή) ή δυναμικά (προσκόμιση τους μέσω διαδικτυακής σύνδεσης). Για λόγους διαθεσιμότητας και ευκολίας επιλέξαμε την πρώτη λύση: ακόμα κι αν τα ασύρματα δίκτυα είναι πλέον παντού, η απαίτηση για αυτά παραμένει ένας υπαρκτός περιορισμός στην πρόσβαση των πληροφοριών. Το σύνολο, λοιπόν, των περιεχομένων της καρτέλας των πληροφοριών θα είναι διαθέσιμο στατικά, κάτι το οποίο αφαιρεί την δυνατότητα δυναμικής ενημέρωσης (σε περιπτώσεις αλλαγής των πληροφοριών στην πηγή) αλλά τις προσφέρει, από την άλλη, ανεξαρτήτως δικτυακών περιορισμών. Εξαίρεση αποτελεί η λίστα των μαθημάτων όλων των εξαμήνων για την οποία επιλέξαμε δυναμική υλοποίηση. Η πηγή, τέλος, των πληροφοριών αυτών είναι η επίσημη ιστοσελίδα του τμήματος Πληροφορικής (*http://www.it.teithe.gr*). Εδώ υπάρχει το θέμα της ενημέρωσης και επικαιροποίησης κάποιων πληροφοριών αλλά θα αναφερθούμε σε αυτό αργότερα.

Θέματα που προέκυψαν από την ανάπτυξη των υπηρεσιών στη τρίτη καρτέλα ήταν κυρίως περιεχομένου και όχι τεχνικών λεπτομερειών. Οι διαδικτυακές υπηρεσίες είναι συγκεκριμένες και άρα οι αποφάσεις μας είχαν να κάνουν με την λειτουργικότητα και την επιλογή ανάμεσα σε αυτές.

Κατά την διάρκεια της ανάπτυξης των επιπρόσθετων υπηρεσιών της τέταρτης καρτέλας προέκυψαν πολλές λεπτομέρειες που έπρεπε να συζητηθούν. Η υπηρεσία της πληρότητας της αστικής γραμμής 52 ήταν μια ιδέα που παρόμοια

της είχε ήδη υλοποιηθεί, στο σύνολο των αστικών γραμμών του ΟΑΣΘ, από άλλη εφαρμογή. Εμείς έπρεπε πλέον να αποφασίσουμε πως θα υλοποιήσουμε την δική μας σκέψη, εξυπηρετώντας του φοιτητές του τμήματος και, γενικότερα, τον κάθε χρήστη. Τελικά, απορρίψαμε την ιδέα να αλληλεπιδρά η λειτουργία αυτή με μία άλλη εφαρμογή/βάση δεδομένων και επιλέξαμε μια αυτόνομη υλοποίηση στην οποία τα δεδομένα θα αποθηκεύονταν και θα ανακτούνται από δική μας βάση. Είναι μία κλειστή μεν αλλά λειτουργική για τον σκοπό που θέλουμε να καταφέρουμε. Έτσι, τα δεδομένα πληρότητας που θα στέλνει ο χρήστης θα αποθηκεύονται στην κατάλληλη, εσωτερικά μορφοποιημένη, μορφή και θα εμφανίζονται στην υπηρεσία της εφαρμογής με ανάκτηση από τους πίνακες τους. Τα βήματα που εκτελούνται δηλαδή θα είναι τα εξής: ο χρήστης θα έχει στην οθόνη του δύο περιοχές, ή μια για την εμφάνιση της πληρότητας και η δεύτερη για την καταχώριση τιμής. Η εμφάνιση περιλαμβάνει μία μπάρα προόδου όπου αναπαριστάται γραφικά η πληρότητα της τελευταίας ενημέρωσης από χρήστη και ένα σύνολο των τελευταίων ενημερώσεων ώστε να έχει μια εικόνα για την συνολική κατάσταση της πληρότητας. Στη δεύτερη περιοχή, αυτή της καταχώρισης, θα υπάρχει μία μπάρα όπου ο χρήστης θα θέτει την αριθμητική τιμή της πληρότητας (επί τοις εκατό), δύο επιλογές όπου θα ορίζει την αφετηρία του (αν ανέβηκε στο αστικό λεωφορείο από το ΑΤΕΙΘ ή από τον Νέο Σιδηροδρομικό Σταθμό) και, τέλος, ένα πλήκτρο για να στείλει την καταχώριση του. Όσον αφορά το θέμα της καταχώρισης όμως, πολλά προβλήματα μπορούσαν να προκύψουν, τα περισσότερα από τα οποία έχουν να κάνουν με την σωστή χρήση της υπηρεσίας από τον χρήστη. Για παράδειγμα, μπορεί να γίνει αλόγιστη χρήση, στέλνοντας αντίθετες καταχωρίσεις. Στο μέτρο που αυτό δεν αφορά την ορθή χρήση από μεριάς του χρήστη (κάτι το οποίο δεν μπορούμε να το ελέγξουμε), υλοποιήσαμε έναν περιορισμό ο οποίος δεν του επιτρέπει (μέσω του ελέγχου της IP διεύθυνσης) να στείλει πολλαπλές καταχωρίσεις. Αυτό το μέτρο, μερικώς, λύνει το πρόβλημα της αλόγιστης χρήσης.

Όσον αφορά της υλοποίηση της υπηρεσίας διαδικτυακής συνομιλίας μεταξύ χρηστών (chat service) ακολουθήσαμε μια παρόμοια τεχνική λύση. Βασιστήκαμε στην δική μας βάση δεδομένων για την αποθήκευση – ανάκτηση των πληροφοριών. Εδώ, οι χρήστες θα στέλνουν το κείμενο που έχουν συγγράψει στο κάτω μέρος της οθόνης το οποίο θα αποθηκεύεται ως εγγραφή στον αντίστοιχο

πίνακα της βάσης. Πολλαπλά μηνύματα αποθηκεύονται σε πολλαπλές σειρές. Το κάθε μήνυμα/σειρά περιλαμβάνει τα όνομα του χρήστη, την ώρα και την ημερομηνία αποστολής και το κείμενο του. Το όνομα του (username), σε περίπτωση που έχει δώσει τα στοιχεία του στις ρυθμίσεις (για την πρόσβαση στις υπηρεσίες του τμήματος, στην τρίτη καρτέλα) και αν ο ίδιος το επιθυμεί, είναι το επίσημο που υπάρχει στον λογαριασμό του (ονοματεπώνυμο φοιτητή). Σε οποιαδήποτε άλλη περίπτωση, αν ο φοιτητής δεν θέλει να εμφανίζεται το όνομα του ή αν δεν έχει δώσει τα στοιχεία του ή αν είναι απλός χρήστης της εφαρμογής, ως όνομα εμφανίζεται η λέξη *anonymous*, στα πρότυπα αντίστοιχων υπηρεσιών. Συνοπτικά, η διεπαφή χρήστη αυτής της λειτουργίας είναι η γραμμή συγγραφής κειμένου στο κάτω άκρο της οθόνης και το ιστορικό των συνομιλιών στο μεγαλύτερο μέρος που απομένει. Η εμφάνιση επιλέξαμε να είναι με χρονολογική σειρά ως προς το νεότερο μήνυμα: ό,τι καινούριο εμφανίζεται στην κορυφή της οθόνης.

Απώτερος σκοπός αυτής της πτυχιακής εργασίας είναι να βοηθήσει, να εμπνεύσει, και να προτρέψει τους φοιτητές του τμήματος Πληροφορικής να ασχοληθούν με το συγκεκριμένο πεδίο.

Είτε μέσω της χρήσης της εφαρμογής είτε μέσω της ενημέρωσης για αυτήν την πτυχιακή, θέλουμε οι φοιτητές να ενδιαφερθούν και να ψάξουν για τον προγραμματισμό για κινητές συσκευές (έξυπνα κινητά - smartphones). Είναι ένα συνεχώς αναπτυσσόμενο πεδίο του κλάδου μας με μέλλον και αρκετή ζήτηση ώστε να υπάρξει καλή σταδιοδρομία.

Ο αριθμός έξυπνων κινητών έχει ξεπεράσει το ένα δισεκατομμύριο[1], οι εφαρμογές περισσότερες από ένα εκατομμύριο[2] και η συνολική αξία προβλέπεται ότι θα αντιστοιχεί σε πάνω από εκατό δισεκατομμύρια δολάρια το 2017[3]. Η χρήση τους έχει εισχωρήσει σε κάθε κοινωνική περίσταση, είναι λοιπόν ευκαιρία για εμάς, για την εξάσκηση της εφευρετικότητας και της πρωτοτυπίας.

Σαφώς, εγγύτερος στόχος αυτής της πτυχιακής είναι η εξάσκηση μας στο αντικείμενο που μας ενδιαφέρει. Θεωρούσαμε τα παραπάνω δεδομένα μία πραγματικότητα που μπορεί μας ωφελήσει. Εφόσον, επίσης, θεωρούσαμε το πεδίο αρκετά ενδιαφέρον ώστε να μονοπωλήσει τις σκέψεις μας για το θέμα της

πτυχιακής εργασίας, η κατάληξη μας στον προγραμματισμό έξυπνων κινητών ήταν συνειδητή και ώριμη.

Παράλληλα, επειδή αυτό αποτελεί απλά μία βραχυπρόθεσμη –στο πρώτο στάδιο- θεώρηση και αφορά μόνο την δική μας εξέλιξη, δίνουμε και δώσαμε περισσότερη σημασία στον αντίκτυπο της εργασίας μας στους φοιτητές. Η ενασχόληση μας, με το πέρας της πτυχιακής και της φοίτησης μας, θα είναι κάτι παρελθοντικό εάν δεν έχει κάποια σχέση και στόχευση που αφορά συνολικά την δουλειά που κάνει το τμήμα και οι φοιτητές του.

Έτερος στόχος ήταν, προφανώς, η άμεση βοήθεια προς τους φοιτητές του τμήματος Πληροφορικής και όσους ενδιαφέρονται γι’ αυτό. Έχοντας την εμπειρία των διαδικτυακών υπηρεσιών του τμήματος, μπορέσαμε και υλοποιήσαμε μια χρήσιμη και εύχρηστη κινητή εφαρμογή που θα εξυπηρετεί παράλληλα με τις υπάρχουσες υποδομές. Αυτό που καταφέραμε είναι μακριά από το ιδανικό, μιας και μπορεί να γίνει πολύ καλύτερη και εκτενέστερη ενσωμάτωση υπηρεσιών, αυτό όμως αποτελεί αφορμή για τους επόμενους φοιτητές. Θα αναφέρουμε το παρόν θέμα και παρακάτω, στις προτάσεις μας για μελλοντική προσπάθεια.

Μέσα από την συνολική δουλειά μας, αποκομίσαμε γνώσεις προγραμματισμού, σχεδιασμού, συνεργασίας, επιδιώξεις που δεν ήταν άμεσοι στόχοι αλλά προκύπτουν φυσικά από κάθε συνεργατική εργασία. Πλέον, μετά από την ολοκλήρωση την πτυχιακής εργασίας, έχουμε μία εξοικείωση ή οποία είναι και θα είναι χρήσιμη στις μελλοντικές ασχολίες μας.

Ανάμεσα στα προβλήματα που αντιμετωπίσαμε κατά την διάρκεια της ανάπτυξης της εφαρμογής ήταν αυτό της διαθεσιμότητας των διαδικτυακών υπηρεσιών του τμήματος. Το σύνολο των λειτουργιών που συμπεριλήφθηκαν στην τρίτη καρτέλα χρησιμοποιούν την ανταλλαγή διαδικτυακών μηνυμάτων ανάμεσα στην εφαρμογή και τους εξυπηρετητές του συστήματος ανακοινώσεων (Hydra) και του συστήματος δήλωσης μαθημάτων του τμήματος (Pithia). Η πρόσβαση και η αλληλεπίδραση με αυτές τις υπηρεσίες ήταν κάποιες φορές προβληματική, δυσκολεύοντας όχι μόνο την λειτουργία καθεαυτή αλλά και την διαδικασία εντοπισμού σφαλμάτων στον κώδικα (αποσφαλμάτωση – debugging). Προέκυπταν δηλαδή λανθασμένες εκτιμήσεις για τις αιτίες των σφαλμάτων και την καταλληλότητα των προγραμματιστικών τεχνικών. Το συγκεκριμένο πρόβλημα χρειάστηκε υπομονή και κατανόηση, συνεχίσαμε τις δοκιμές μέσω της ίδιας της εφαρμογής και, παράλληλα, μέσω τυπικών υπολογιστών, δοκιμές σε άλλες χρονικές στιγμές και σε άλλα κομμάτια κώδικα που εκτελούσαν παρόμοιες λειτουργίες, επανεισαγωγή στοιχείων φοιτητή για τις αντίστοιχες υπηρεσίες, δοκιμές με διαφορετικούς λογαριασμούς φοιτητών κ.ο.κ.

Επίσης, αιτία δυσλειτουργιών ήταν και η ποιότητα του σήματος στην διαδικτυακή σύνδεση (Wi-Fi). Κάτι τέτοιο δεν προβλεπόταν από την λειτουργία του κώδικα καθεαυτή, η ακολουθία εκτέλεσης εντολών δεν εξετάζει επιμέρους λεπτομέρειες, αντίθετα, θεωρεί δεδομένο ότι είτε υπάρχει συνεχώς και μόνιμα η απαιτούμενη σύνδεση είτε ότι δεν υπάρχει καμία πρόσβαση. Δηλαδή, σε περιπτώσεις αδύναμου σήματος χρειάζονταν διαφορετική προγραμματιστική αντιμετώπιση. Ήταν διαφορετικός ο χειρισμός σφαλμάτων σε περίπτωση διακοπής, διαφορετικός σε περίπτωση αργής σύνδεσης και ανταλλαγής δεδομένων, διαφορετικός σε περίπτωση χαμηλής ποιότητας σήματος και διαφορετικός όταν δεν υπήρχε καμία πηγή πρόσβασης. Όλες αυτές οι περιπτώσεις χειρίστηκαν ανάλογα, προσαρμόζοντας τον προγραμματισμό σε πιθανά προβλήματα πραγματικής χρήσης (τι θα γίνει αν...).

Ένα εγγενές πρόβλημα που προκύπτει από τον προγραμματισμό είναι αυτό της συμβατότητας. Η εφαρμογή μας αναπτύχθηκε στην έκδοση του λειτουργικού συστήματος Andoird 2.1 (Eclair). Καθώς υπάρχουν νέες όπως και παλαιότερες εκδόσεις Android, μπορούν να δημιουργηθούν προβλήματα προγραμματιστικής

υλοποίησης: τρόποι ανάπτυξης κάποιας λειτουργίας μπορεί κάλλιστα να έχουν σταματήσει να χρησιμοποιούνται (προέκυπτε όταν συμβουλευόμασταν λύσεις από παλιότερες διαδικτυακές υλοποιήσεις), μπορεί να έχουν αλλάξει στον τρόπο συγγραφής και λειτουργίας (διαφορετικά ονόματα κλάσεων, μεθόδων, πακέτων, αρμοδιοτήτων), μπορεί, τέλος, να προτείνεται μια νέα μεθοδολογία ανάπτυξης, κατάλληλη για υποστήριξη μελλοντικών εκδόσεων του λειτουργικού συστήματος.

Περνώντας σε άλλο θέμα, όπως αναφέραμε παραπάνω, η εργασία που κάναμε είναι απλά η δική μας προσπάθεια υλοποίησης των υπηρεσιών που θεωρήσαμε απαραίτητες για μία τέτοια εφαρμογή. Μπορούν να υπάρξουν αμέτρητες τροποποιήσεις και επέκτασης της, ενσωματώνοντας περισσότερες και αρτιότερα υλοποιημένες λειτουργίες. Εμείς, φτάσαμε μέχρι το επίπεδο που θεωρήσαμε επαρκές για ένα ικανοποιητικό αποτέλεσμα. Αυτό δεν είναι παρά το σημείο εκκίνησης για μελλοντικές προσπάθειες, για τις οποίες θα αναφέρουμε μερικές ιδέες παρακάτω.

Όσον αφορά τις υπόλοιπες διαδικτυακές υπηρεσίες που παρέχονται στους φοιτητές, έχουμε προτάσεις για το σύστημα μαθημάτων του Blackboard. Οι υπηρεσίες του Blackboard αφορούν ανακοινώσεις, υλικό μαθημάτων, οδηγίες και γενικότερα πληροφορίες που έχουν προστεθεί από καθηγητές του τμήματος. Παράλληλα με αυτό, κάποιοι καθηγητές επιλέγουν την προσωπική τους ιστοσελίδα (aetos.it.teithe.gr/~xxx) για τις ίδιες λειτουργίες που προσφέρει το Blackboard. Οπότε, συνοψίζοντας, έχουμε αυτές τις δύο πηγές φοιτητικής ενημέρωσης. Μια συγκεντρωτική υλοποίηση όλου αυτού του όγκου πληροφοριών σε μία μόνο υπηρεσία, στην οποία ο φοιτητής θα έχει να επιλέξει μόνο το μάθημα που τον ενδιαφέρει από το κάθε εξάμηνο φοίτησης, θα ήταν ένα εξαιρετικό βήμα για την διευκόλυνση των φοιτητών. Φυσικά, αυτή η εργασία θα απαιτούσε πολύ χρόνο και προσπάθεια αλλά είναι μία πολύ καλή ιδέα για επέκταση της παρούσας ή ακόμα και ως ανάπτυξη μίας αυτόνομης εφαρμογής για το τμήμα μας.

Μία άλλη ιδέα είναι να ενσωματωθεί μια έκδοση του ηλεκτρονικού ταχυδρομείο (webmail) που παρέχει το τμήμα, έτσι ώστε να μπορούν οι φοιτητές να το χειριστούν εύκολα και γρήγορα με την συσκευή τους. Η λειτουργία αυτή μπορεί να συνδέεται με την λίστα (υπηρεσία που υπάρχει στη δική μας

υλοποίηση) των καθηγητών και των φοιτητών, κάτι που θα διευκολύνει κατά πολύ την επικοινωνία.

Επίσης, μπορεί να ενσωματωθεί μία έκδοση του φόρουμ των φοιτητών του τμήματος (steki). Μπορεί να γίνει εύκολα με άνοιγμα μιας νέας φόρμας φυλλομετρητή σε συγκεκριμένο παράθυρο της εφαρμογής. Άλλες υπηρεσίες που μπορούν να ενσωματωθούν είναι αυτή της live κάμερας του τμήματος και αυτή της βιβλιοθήκης του Ιδρύματος.

Μία τελευταία και απαιτητική πρόταση για μελλοντική εργασία είναι η επέκταση της εφαρμογής ώστε να απευθύνεται σε όλους τους φοιτητές του Ιδρύματος και όχι μόνο του τμήματος Πληροφορικής. Αυτό, φυσικά, θα απαιτείσαι μεγάλη προσπάθεια, ο όγκος και η δυσκολία θα είναι πολλαπλάσια, αλλά είναι μια ιδέα που θα βοηθήσει, κατά αναλογία, πολλαπλάσιο αριθμό ενδιαφερομένων. Μπορεί να υλοποιηθεί με μια κοινή βάση δεδομένων, κοινή λίστα καθηγητών, χωρισμένη ανά τμήμα, με διαδικασία πιστοποίησης για κάθε τμήμα ξεχωριστά.

Συμπερασματικά, η παρούσα πτυχιακή εργασία αποτελεί το αποτέλεσμα συγκέντρωσης χρήσιμων πληροφοριών και υπηρεσιών διαχωρισμένων με βάση τον χρήστη. Τα τρία πλαίσια στα οποία αναπτύσσεται είναι, αρχικά, αυτό που απευθύνεται προς κάθε ενδιαφερόμενο, της πληροφόρησης, δηλαδή, για το τμήμα και ό,τι αυτό παρέχει, έπειτα είναι αυτό της παροχής διαδικτυακών υπηρεσιών προς του φοιτητές που είναι ενεργά μέλη και θα βοηθηθούν από τα εργαλεία που αναπτύξαμε και, τέλος, είναι οι επιπρόσθετες υπηρεσίες που είναι προσβάσιμες από όλους, αποτελούν δική μας ιδέα και υλοποίηση, και στηρίζονται στην χρήση και αξιοποίηση από τους τελικούς αποδέκτες.

Ξεκινήσαμε, αναπτύξαμε, και ολοκληρώσαμε την πτυχιακή μας εργασία συνεργατικά, με διαμοιρασμένες αρμοδιότητες, με συχνές συναντήσεις και συζητήσεις για θέματα που αφορούσαν κάθε πτυχή της εφαρμογής. Χρησιμοποιήσαμε το περιβάλλον Eclipse και το πρόσθετο της διαδικτυακής υπηρεσίας GitHub για να γίνει εύκολη η μεταξύ μας συνεργασία. Απευθυνθήκαμε σε φοιτητές στις περιπτώσεις που θεωρήσαμε πως δεν αρκεί η προσωπική μας εμπειρία, έτσι, ζητήσαμε την γνώμη των μελλοντικών χρηστών της εφαρμογής για

θέματα υλοποίησης, περιεχομένου, τεχνικής, και χειριστήκαμε όπως κρίναμε ορθό τις πληροφορίες που αποκομίσαμε.

Λύσαμε με συνεχείς προσπάθειες και δοκιμές όσα προβλήματα προέκυπταν κατά την διάρκεια της ανάπτυξης της εφαρμογής: σε αυτές, μεγαλύτερη και χρησιμότερη βοήθεια ήταν οι διαδικτυακές κοινότητες που φιλοξενούσαν υλικό, παραδείγματα, απορίες και λύσεις που αφορούσαν όποιο πρόβλημα μας προέκυπτε. Επίσης, τα επεξηγηματικά αρχεία που προσφέρει η πλατφόρμα Android καθώς και μερικά βιβλία για την πλατφόρμα βοήθησαν να ξεπεραστούν παρόμοια προβλήματα.

Σκοπός μας ήταν και είναι η επιρροή προς τους φοιτητές του τμήματος μας. Θέλουμε να επεκταθεί η εξάσκηση και ο προγραμματισμός σε πλατφόρμες έξυπνων κινητών τηλεφώνων στο τμήμα Πληροφορικής καθώς αποτελεί ένα συνεχώς αναπτυσσόμενο και εξαιρετικά ενδιαφέρον πεδίο του κλάδου μας. Υπάρχουν διάφορες δράσεις που ωθούν μια μερίδα των φοιτητών να ασχοληθούν σχετικά αλλά νομίζουμε πως μια καθολικότερη αντιμετώπιση θα ήταν καταλληλότερη. Σαφώς, μέσα από την παρούσα πτυχιακή εργασία, εμείς αποκομίσαμε πολλά, εξοικειωθήκαμε και μάθαμε τον προγραμματισμό Android, καταφέραμε να συνεργαστούμε και να μάθουμε μέσα από την συνεργατική εργασία, θέματα που πιστεύουμε πως θα είναι χρήσιμα στην μετέπειτα επαγγελματική μας πορεία.

Πιθανές μελλοντικές ιδέες επέκτασης της εργασίας μας θα είναι το καλύτερο δυνατό επακόλουθο της δουλειάς μας. Προτείναμε και αναλύσαμε κάποιες σχετικές ιδέες και, από εδώ και πέρα, βασιζόμαστε στην διάθεση και την εφευρετικότητα των συναδέλφων μας.

Η βασική δομή της πτυχιακής εργασίας είναι η εξής: τρία συγκεντρωτικά κεφάλαια που αφορούν αντίστοιχα το λειτουργικό σύστημα Android, το περιβάλλον ανάπτυξης Eclipse, και την ίδια την εφαρμογή με την παράθεση και επεξήγηση των λειτουργιών και του κώδικα της.

Στο πρώτο κεφάλαιο θα εισάγουμε τον αναγνώστη στο θεωρητικό υπόβαθρο που χρειάζεται για να κατανοήσει και να μάθει το λειτουργικό σύστημα Android. Δεν θα αναφέρουμε περιττές και πολύ εξειδικευμένες λεπτομέρειες του, θα αρκεστούμε σε μία επαρκή ανάλυση των σημαντικότερων δεδομένων που θα πρέπει να γνωρίζει ο αναγνώστης.

Στο δεύτερο κεφάλαιο θα παρουσιάσουμε το περιβάλλον ανάπτυξης Eclipse και θα αναλύσουμε την λειτουργία και τον τρόπο χρήσης του. Είναι το βασικότερο εργαλείο προγραμματισμού για εφαρμογές Android οπότε η επεξήγηση του είναι πολύ σημαντική για τους ενδιαφερόμενους. Σε ένα μέρος του κεφαλαίου αναπτύσσουμε και την βασική λειτουργία του προσθέτου που αφορά την διαδικτυακή υπηρεσία GitHub η οποία δίνει την δυνατότητα συνεργατικού προγραμματισμού.

Στο τρίτο κεφάλαιο αναλύουμε όλες τις λειτουργίες που ενσωματώσαμε στην εφαρμογή μας. Παραθέτουμε τον κώδικα και όπου είναι απαραίτητο αναφέρουμε λεπτομερώς τι εκτελεί κάθε σύνολο εντολών. Η μορφή που διαλέξαμε ακολουθεί την σειρά των υπηρεσιών της εφαρμογής: έκαστο υποκεφάλαιο αφορά μία συγκεκριμένη υπηρεσία, όπου αυτή επεξηγείται περιληπτικά και έπειτα παραθέτεται και επεξηγείται με την σειρά του και ο κώδικας της.

Στο τέλος της πτυχιακής εργασίας περιλαμβάνεται ένας οδηγός χρήσης λογισμικού στον οποίον εξηγείται το σύνολο των λειτουργιών σε κάθε επιλογή της διεπαφής χρήστη. Συγκεκριμένα, είναι οι τέσσερις καρτέλες που αναπτύξαμε, με όλες τις υπηρεσίες τους, και η διεπαφή με τις απαραίτητες επιλογές/ρυθμίσεις της εφαρμογής. Όλες οι επεξηγήσεις συνοδεύονται από εικόνες, τα αντίστοιχα στιγμιότυπα των λειτουργιών που θα συναντήσει ο χρήστης κατά την χρήση της εφαρμογής.

Κεφάλαιο 1: Το Λειτουργικό Σύστημα Android

Εισαγωγή

Το λειτουργικό σύστημα Android είναι η πιο δημοφιλής και διαδεδομένη πλατφόρμα για έξυπνα κινητά τηλέφωνα και φορητές συσκευές. Είναι βασισμένη στο Linux και αναπτύσσεται από τις Google και Open Handset Alliance. Ο προγραμματισμός γίνεται στη γλώσσα Java αλλά υπάρχουν κι άλλες υποστηριζόμενες επιλογές (C, C++). Σχεδιάστηκε και αφορά κυρίως συσκευές με οθόνη αφής και αποτελεί λογισμικό ανοιχτού κώδικα: ο κώδικας του έχει διατεθεί υπό την άδεια Apache[5].

Στο παρόν κεφάλαιο θα παρουσιάσουμε την ιστορική διαδρομή, τις αντίστοιχες εκδόσεις του λειτουργικού, και κάποια βασικά στοιχεία της αρχιτεκτονικής του ως μία απαραίτητα εισαγωγή για την κατανόηση της πλατφόρμας στην οποία δουλέψαμε και στην οποία θα απευθυνθούν όσοι ενδιαφερθούν για τον προγραμματισμό παρόμοιων εφαρμογών.

Υποκεφάλαιο 1.1: Ιστορικά Στοιχεία

Η Android Inc ιδρύθηκε στη Silicon Valley, στην Καλιφόρνια, τον Οκτώβριο του 2003, με την ιδέα της παροχής μιας κινητής πλατφόρμας που είναι περισσότερο ενήμερη για την τοποθεσία του χρήστη και τις προτιμήσεις του.

Η Google εξαγόρασε την Android Inc τον Αύγουστο του 2005 ως θυγατρική πλήρους ιδιοκτησίας της Google Inc. Σκοπός της ήταν να παρέχει μια πλήρως ανοικτή πλατφόρμα, υποστηριζόμενη από τις υπηρεσίες και τις τεχνολογίες της Google, τόσο για τους χρήστες όσο και για τους προγραμματιστές εφαρμογών.

Τον Νοέμβριο του 2007, ιδρύθηκε η Open Handset Alliance ως κοινοπραξία για να αναπτύξει ένα ανοικτό πρότυπο για τις κινητές συσκευές. Η Open Handset Alliance ξεκίνησε τη πορεία της ανακοινώνοντας την πλατφόρμα Android. Σε λιγότερο από ένα χρόνο, νέα μέλη άρχισαν να συμμετέχουν σε αυτήν την κοινοπραξία.

Το Android έγινε μια ανοιχτού λογισμικού πρωτοβουλία που καθοδηγείται από την Google κάτω από την υποστήριξη της Open Handset Alliance. Ο στόχος του Android, ως έργο ανοικτού κώδικα, είναι να παράσχει μια ανοικτή πλατφόρμα για τη βελτίωση της εμπειρίας των χρηστών κινητών τηλεφώνων.

Το Android είναι η πρώτη πλήρης, ανοικτή, και δωρεάν πλατφόρμα κινητής τηλεφωνίας:

- Πλήρης: Η πλατφόρμα Android είναι μία ισχυρή, ασφαλής, εύκολα αναβαθμιζόμενη, κινητή πλατφόρμα με ένα ολοκληρωμένο πλαίσιο και σαφώς καθορισμένες διασυνδέσεις. Επιτρέπει σε προγραμματιστές εφαρμογών να αναπτύξουν πλήρως και να συνδυάζουν τις εφαρμογές τους με τις δυνατότητες της πλατφόρμας. Επίσης, παρέχει συμβατότητα και τα απαραίτητα προγράμματα πιστοποίησης, ώστε οι κατασκευαστές να μπορούν να σχεδιάζουν εξαιρετικά συμβατές συσκευές.

- Ανοιχτή: Το σύνολο της πλατφόρμας Android έχει αναπτυχθεί και διατεθεί κάτω από τους όρους που προβλέπονται από την άδεια ανοικτού κώδικα Apache (Apache License). Το Android δεν κάνει διάκριση μεταξύ προεγκατεστημένων και εφαρμογών τρίτων κατασκευαστών (third-party applications). Οι προγραμματιστές έχουν πλήρη πρόσβαση στα χαρακτηριστικά και τις υπηρεσίες της συσκευής κατά την διάρκεια ανάπτυξης των εφαρμογών.

- Δωρεάν: Η πλατφόρμα Android δεν χρεώνει την αδειοδότηση, τα δικαιώματα, τη συμμετοχή, ή την πιστοποίηση για την ανάπτυξη εφαρμογών για την πλατφόρμα. Ο πηγαίος κώδικας της πλατφόρμας Android και το κιτ ανάπτυξης λογισμικού της παρέχεται δωρεάν σε προγραμματιστές εφαρμογών. Η πλατφόρμα ανάπτυξης λογισμικού (η πλατφόρμα Eclipse που αναλύουμε στο επόμενο κεφάλαιο) είναι διαθέσιμη σε πολλά λειτουργικά συστήματα υπολογιστών (Linux, Windows, Mac OS X, Solaris), επιτρέποντας τους προγραμματιστές να αναπτύξουν εφαρμογές που χρησιμοποιούν το λειτουργικό σύστημα της επιλογής τους.

Σήμερα, το Android είναι μία από τις σημαντικότερες πλατφόρμες στην αγορά κινητής τηλεφωνίας. Σύμφωνα με πρόσφατες αναλύσεις της αγοράς, κατά μέσο όρο, 400 χιλιάδες συσκευές Android ενεργοποιούνται καθημερινά και περισσότερα από 400 εκατομμύρια συσκευές έχουν ήδη ενεργοποιηθεί. Το Android, σήμερα, έχει περίπου το 56% του μεριδίου της αγοράς κινητής τηλεφωνίας παρουσιάζοντας συνεχή ανοδική πορεία.

Υποκεφάλαιο 1.2: Οι εκδόσεις του Android

Η πρώτη beta έκδοση της πλατφόρμας Android κυκλοφόρησε στις 5 Νοεμβρίου του 2007. Από τότε, έχουν γίνει μια σειρά από ενημερώσεις και διορθώσεις σφαλμάτων. Αν και οι διορθώσεις σφαλμάτων είναι συνήθως φανερές από την οπτική γωνία του προγραμματιστή εφαρμογών, οι νέες ενημερώσεις του λειτουργικού Android συνήθως σημαίνουν αλλαγές και προσθήκες στο πλαίσιο API. Για το λόγο αυτό, εκτός από τους αριθμούς εκδόσεων της πλατφόρμας Android, ένας δεύτερος αριθμός έκδοσης, που ονομάζεται επίπεδο API (API level), χρησιμοποιείται για να προσδιορίσει το πλαίσιο API (Application programming interface) που υποστηρίζεται.

Από τον Απρίλιο του 2009, κάθε έκδοση του Android έχει διατεθεί με μία κωδική ονομασία επιδορπίου, όπως Cupcake, Eclair, Froyo, Gingerbread κ.α. Αυτό εισήγαγε ένα τρίτο σύστημα προσδιοριστικών εκδόσεων στην πλατφόρμα, κάνοντας τα πράγματα ακόμα πιο αινιγματικά για τους προγραμματιστές εφαρμογών Android που έρχονται για πρώτη φορά σε επαφή με αυτή. Όσον αφορά την ανάπτυξη εφαρμογών Android, φράσεις που χρησιμοποιούν συχνά οι προγραμματιστές είναι, μεταξύ άλλων, οι παρακάτω: «η εφαρμογή απαιτεί έκδοση Eclair και πάνω», «αυτή η μέθοδος απαιτεί τουλάχιστον επίπεδο API 9», και «αναβάθμισα το κινητό μου τηλέφωνο μου στην έκδοση Android 2.1». Το να κατανοεί ο προγραμματιστής ποια έκδοση και ποιο επίπεδο API αναφέρεται, καθώς και ποια νέα APIs αποτελούν μέρος ποιας έκδοση της πλατφόρμας Android, μπορεί εύκολα να γίνει μια περίπλοκη διαδικασία. Για να γίνει περισσότερο κατανοητό, παρακάτω παρουσιάζουμε (Σχήμα 1) μια λίστα με τις εκδόσεις ως αναφορά στα τρία επίπεδα του συστήματος εκδόσεων.

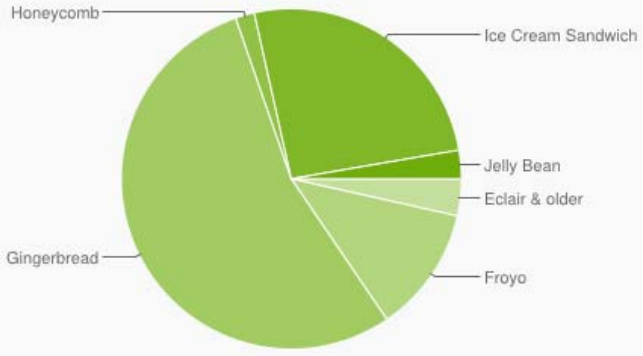
Release Date	Platform Version	API Level	Codename
November 5, 2007	Beta		
September 23, 2008	Android 1.0	1	
February 9, 2009	Android 1.1	2	
April 30, 2009	Android 1.5	3	Cupcake
September 15, 2009	Android 1.6	4	Donut
October 26, 2009	Android 2.0	5	Éclair
December 3, 2009	Android 2.0.1	6	Éclair
January 12, 2009	Android 2.1	7	Éclair
May 20, 2010	Android 2.2	8	Froyo
January 18, 2011	Android 2.2.1	8	Froyo
January 22, 2011	Android 2.2.2	8	Froyo
November 21, 2011	Android 2.2.3	8	Froyo
December 6, 2010	Android 2.3	9	Gingerbread
February 9, 2011	Android 2.3.3	10	Gingerbread
July 25, 2011	Android 2.3.5	10	Gingerbread
September 2, 2011	Android 2.3.6	10	Gingerbread
February 22, 2011	Android 3.0	11	Honeycomb
May 10, 2011	Android 3.1	12	Honeycomb
July 15, 2011	Android 3.2	13	Honeycomb
September 20, 2011	Android 3.2.1	13	Honeycomb
August 30, 2011	Android 3.2.2	13	Honeycomb
October 19, 2011	Android 4.0.1	14	Ice Cream Sandwich
November 28, 2011	Android 4.0.2	14	Ice Cream Sandwich
December 16, 2011	Android 4.0.3	15	Ice Cream Sandwich
February 4, 2012	Android 4.0.4	15	Ice Cream Sandwich

Σχέδιο 1: Λίστα εκδόσεων Android

Όπως φαίνεται στο Σχέδιο 1, υπάρχουν 15 επίπεδα API που πρέπει να έχουμε υπόψη κατά την ανάπτυξη των εφαρμογών. Το επίπεδο API καθορίζει το μέγεθος του κοινού στο οποίο απευθύνεται οπότε θα πρέπει να γίνει επιλογή κατόπιν σκέψης διότι είναι πολύ σημαντικό για την προοπτική μιας νέας εφαρμογής.

Η αγορά, όσον αφορά τα κινητά τηλέφωνα Android, είναι εξαιρετικά κατακερματισμένη. Βλέποντας τις ημερομηνίες κυκλοφορίας, μπορούμε να υποθέσουμε ότι το μεγαλύτερο μέρος της βάσης των χρηστών Android χρησιμοποιεί τουλάχιστον την έκδοση Android 3.0, δεδομένου ότι είναι διαθέσιμη εδώ και ένα περίπου ενάμιση χρόνο. Ωστόσο, αυτό δεν ισχύει. Λόγω του κατακερματισμού, οι ημερομηνίες δεν δίνουν μία σαφή εικόνα των εκδόσεων που είναι σε χρήση για κάθε χρονική στιγμή. Στο Σχέδιο 2 βλέπουμε την τελευταία έκδοση του διαγράμματος χρήσης για τις εκδόσεις της πλατφόρμας Android (http://developer.android.com/resources/dashboard/platformversions.html).

Version	Codename	API	Distribution
1.5	Cupcake	3	0.1%
1.6	Donut	4	0.3%
2.1	Eclair	7	3.1%
2.2	Froyo	8	12%
2.3 - 2.3.2	Gingerbread	9	0.3%
2.3.3 - 2.3.7		10	53.9%
3.1	Honeycomb	12	0.4%
3.2		13	1.4%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	25.8%
4.1	Jelly Bean	16	2.7%



Σχέδιο 2: Διάγραμμα χρήσης εκδόσεων Android

Όπως βλέπουμε στο Σχέδιο 2, το μεγαλύτερο μέρος της βάσης των χρηστών Android χρησιμοποιεί αυτήν τη στιγμή την έκδοση Android 2.3.3-7 Gingerbread. Αυτό σημαίνει ότι μία εφαρμογή θα πρέπει να υποστηρίζει τουλάχιστον το επίπεδο API 10, προκειμένου να μπορεί να απευθυνθεί στην πλειοψηφία των χρηστών. Επίσης, σημαίνει ότι ο προγραμματιστής δεν θα είναι σε θέση να χρησιμοποιήσει τις πιο πρόσφατες δυνατότητες API που έχουν εισαχθεί στις νεότερες εκδόσεις της πλατφόρμας. Στη δική μας εφαρμογή, η ανάπτυξη γίνεται στην έκδοση Android 2.1.

Η ποικιλία των εκδόσεων είναι ένα κοινό πρόβλημα για τους προγραμματιστές Android. Οι περισσότεροι προγραμματιστές αναπτύσσουν πακέτα για διαφορετικά επίπεδα API. Αυτό επιλύει το πρόβλημα, αλλά σημαίνει επίσης ότι πρέπει να διατηρούνται διαφορετικοί κλάδοι κώδικα.

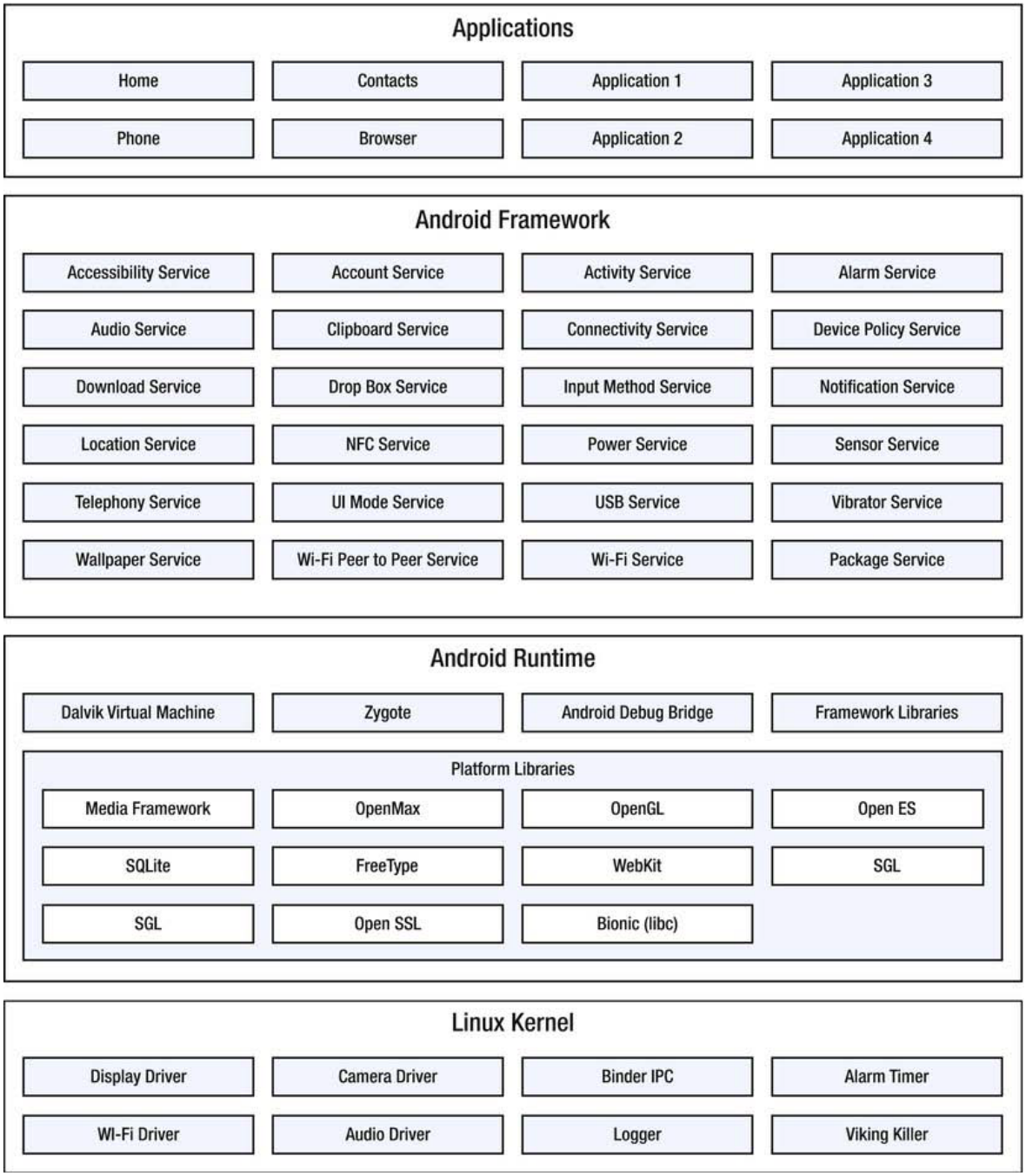
Τον Μάρτιο του 2011, η Google παρουσίασε το Πακέτο Στήριξης (Support Package) ως λύση για το πρόβλημα των εκδόσεων. Το Πακέτο Στήριξης είναι ένα σύνολο από στατικές βιβλιοθήκες που επιτρέπει στους προγραμματιστές εφαρμογών Android την ανάπτυξη εφαρμογών που υποστηρίζουν πολλαπλές εκδόσεις της πλατφόρμας. Ο κύριος στόχος του πακέτου στήριξης είναι η απλούστευση της διαδικασίας της υποστήριξης πολλαπλών εκδόσεων του Android από μια ενιαία βάση κώδικα. Περισσότερες πληροφορίες σχετικά με το Πακέτο Στήριξης υπάρχουν στη παρακάτω ιστοσελίδα του Android: http://developer.android.com/sdk/compatibility-library.html.

Υποκεφάλαιο 1.3: Η αρχιτεκτονική του Android

Το Android είναι περισσότερο μία πλήρη στοίβα λογισμικού για κινητές συσκευές παρά ένα λειτουργικό σύστημα. Είναι ένας συνδυασμός εργαλείων και τεχνολογιών που έχουν προσεκτικά βελτιστοποιηθεί για τις ανάγκες της κινητής τηλεφωνίας.

Το Android βασίζεται στον αποδεδειγμένα ικανό πυρήνα του Linux με σκοπό την παροχή λειτουργικότητας για το λειτουργικό σύστημα. Για τις εφαρμογές του χρήστη, το Android βασίζεται στην τεχνολογία της εικονικής μηχανής Java (Java Virtual Machine) με τη χρησιμοποίηση της εικονικής μηχανής Dalvik (Dalvik Virtual Machine – Υποκεφάλαιο 1.3.3). Η διαδικασία εφαρμογής Android Zygote (Υποκεφάλαιο 1.3.2), μέσω της προφόρτισης υπηρεσιών (service preloading) και του διαμοιρασμού πόρων (resource sharing), ενισχύει τους χρόνους εκκίνησης εφαρμογής και επιτρέπει την αποτελεσματική χρήση των περιορισμένων πόρων μνήμης σε κινητές πλατφόρμες. Όλες αυτές οι επιτυχημένες τεχνολογίες παίζουν ένα σημαντικό ρόλο στην επιτυχία της πλατφόρμας Android, όπως απεικονίζεται στο Σχήμα 3. Εκτός από αυτά τα εργαλεία και τις τεχνολογίες, το Android παρέχει ένα μοναδικό περιβάλλον που έχει σχεδιαστεί για την παροχή ομαλής λειτουργίας προς τους τελικούς χρήστες, ενώ παρέχει ένα εξορθολογισμένο περιβάλλον ανάπτυξης εφαρμογών για τους προγραμματιστές.

Η ύλη του παρόντος υποκεφαλαίου είναι μακροσκελής διότι περιλαμβάνει την επεξήγηση της εσωτερικής δομής του λειτουργικού συστήματος. Έτσι, αποφασίσαμε να παραθέσουμε την απεικόνιση της αρχιτεκτονικής της πλατφόρμας Android (Σχέδιο 3) μαζί με μία συνοπτική αναφορά των κυριότερων μερών της. Για μία εκτενέστερη ανάλυση, παραπέμπουμε στο βιβλίο *Android Apps with Eclipse* που παραθέτουμε στη βιβλιογραφία.



Σχέδιο 3: Αρχιτεκτονική Android

Υποκεφάλαιο 1.3.1: Επίπεδο Αφαίρεσης Υλικού

Το Επίπεδο Αφαίρεσης Υλικού (Hardware Abstraction Layer) είναι το επίπεδο που μεσολαβεί στις αλληλεπιδράσεις του υλικού (hardware) με τις αντίστοιχες υπηρεσίες που προσφέρει στο λογισμικό (software). Δίνει, δηλαδή, ως αφαιρετική επίπεδο, την δυνατότητα προγραμματισμού και ελέγχου του υλικού χωρίς να απαιτείται επακριβής γνώση των λεπτομερειών του.

Το Android, για το Επίπεδο Αφαίρεσης Υλικού, βασίζεται στον πυρήνα του Linux το οποίο, επίσης, του παρέχει την λειτουργικότητα του λειτουργικού συστήματος. Κατά τη διάρκεια της ανάπτυξης του Android έχουν γίνει πολλές βελτιώσεις στον κώδικα του πυρήνα του Linux (Linux Kernel) προκειμένου να ικανοποιεί τις ανάγκες των κινητών τηλεφώνων.

Τα πιο αξιοσημείωτα χαρακτηριστικά είναι τα ακόλουθα:

- Alarm timer: Δυνατότητα προγραμματισμού μελλοντικών ειδοποιήσεων από όλες τις εφαρμογές
- Paranoid network security: Μέθοδοι ασφάλειας για την πρόσβαση σε δικτυακούς πόρους
- Binder: Μηχανισμός που επιτρέπει την επικοινωνία μεταξύ των εφαρμογών και αυτών, επιπλέον, με το λειτουργικό σύστημα
- Wakelocks: Δικλίδα ελέγχου της κατάστασης του λειτουργικού συστήματος για την ολοκλήρωση εκτελούμενου κώδικα
- Android shared memory (Ashmem): Κοινή μνήμη σε επίπεδο πυρήνα
- Process shared memory (Pmem): Κοινή μνήμη για τις λειτουργίες των εφαρμογών και των διαδικασιών τους
- Low memory killer (Viking Killer): Μηχανισμός για την προτεραιότητα των εφαρμογών στους πόρους της διαθέσιμης μνήμης
- Logger: Μηχανισμός καταγραφής πληροφοριών του συστήματος και της αλληλεπίδρασης των εφαρμογών με αυτό

Υποκεφάλαιο 1.3.2: Zygote

Στα περισσότερα UNIX-οειδή λειτουργικά συστήματα, η εφαρμογή που είναι γνωστή ως Init θεωρείται ως η μητρική όλων των διαδικασιών. Η Init μπαίνει σε λειτουργία όταν ο πυρήνας έχει ξεκινήσει με επιτυχία. Ο κύριος ρόλος της είναι να ξεκινήσει μια σειρά από άλλες διεργασίες με βάση τη διαμόρφωση του συστήματος.

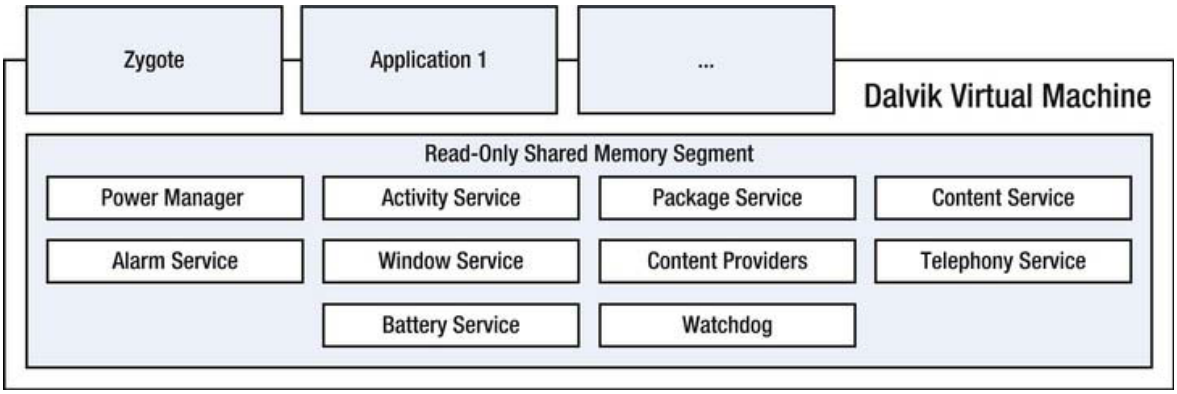
Η Zygote, γνωστή ως η «διαδικασία εφαρμογών» (app process), είναι μία από τις βασικές διαδικασίες που ξεκινούν από την Init. Ο ρόλος της στο πλαίσιο της πλατφόρμας Android είναι παρόμοιος με αυτόν της Init. Ο πρώτος στόχος της είναι να ξεκινήσει μια νέα εικονική μηχανή Dalvik (Υποκεφάλαιο 1.3.3) και να προετοιμάσει τις βασικές υπηρεσίες του Android, όπως οι παρακάτω:

- Power service
- Activity service
- Package service
- Content service
- Alarm service
- Window service
- Content providers
- Telephony service
- Battery service
- Watchdog

Μετά την εκκίνηση αυτών των υπηρεσιών, η Zygote αρχίζει να δουλεύει για την δεύτερη εργασία της, από την οποία προέρχεται και το όνομα της.

Στο Android, κάθε εφαρμογή τρέχει σε μία δική της διαδικασία της εικονικής μηχανής, αφιερωμένα για την εκτέλεση της. Επιπλέον, οι εφαρμογές Android βασίζονται σε ένα σύνολο από τάξεις (Classes) και αντικείμενα δεδομένων (Data objects) που πρέπει πρώτα να φορτωθούν στη μνήμη ώστε να εκπληρώσουν την αποστολή τους. Αυτό απαιτεί μια μεγάλη επιβάρυνση κατά την έναρξη μιας νέας εφαρμογής. Παρ' όλα αυτά, το Android θα πρέπει να κρατήσει τον χρόνο εκκίνησης όσο το δυνατόν συντομότερο προκειμένου να παρέχει γρήγορη απόκριση για τις ανάγκες του χρήστη. Με την χρήση της διακλάδωσης, η Zygote επιλύει αυτό το πρόβλημα με ένα γρήγορο και αποτελεσματικό τρόπο.

Στην επιστήμη των υπολογιστών, διακλάδωση ονομάζεται η λειτουργία της κλωνοποίησης μιας υπάρχουσας διαδικασίας. Η νέα διαδικασία έχει ένα ακριβές αντίγραφο όλων των τμημάτων της μνήμης της μητρικής διαδικασίας. Ωστόσο, οι δύο διεργασίες εκτελούνται ανεξάρτητα (Σχέδιο 4).



Σχέδιο 4: Η Zygote μαζί με άλλες εφαρμογές

Η Zygote φορτώνει τα αντικείμενα του Android runtime και περιμένει τις αιτήσεις για την έναρξη νέων εφαρμογών. Όταν μια νέα αίτηση φτάνει, αντί της έναρξης μιας νέας διαδικασίας εικονικής μηχανής, απλά διακλαδώνεται. Αυτό επιτρέπει στη νέα εφαρμογή να ξεκινήσει πολύ γρήγορα, διατηρώντας παράλληλα το αποτύπωμα της μνήμης σε χαμηλά επίπεδα.

Υποκεφάλαιο 1.3.3: Εικονική Μηχανή Dalvik

Η Java είναι μια γενικής χρήσης, αντικειμενοστρεφής γλώσσα προγραμματισμού που είναι ειδικά σχεδιασμένη για την ανάπτυξη εφαρμογών και την λειτουργίας τους ανεξαρτήτως πλατφόρμας. Στόχος είναι ο προγραμματισμός εφαρμογών μία φορά έτσι ώστε να είναι εκτελέσιμες σε όλες τις πλατφόρμες. Η Java το επιτυγχάνει με την μεταγλώττιση του κώδικα της εφαρμογής σε μία ενδιάμεση γλώσσα, η οποία μπορεί να ερμηνευθεί ανεξάρτητα από την πλατφόρμα, που ονομάζεται bytecode. Κατά τη διάρκεια της εκτέλεσης, το bytecode εκτελείται μέσω μίας άλλης οντότητας της Java που ονομάζεται Εικονική Μηχανή Java (Java Virtual Machine).

Οι εικονικές μηχανές είναι εγγενείς εφαρμογές που τρέχουν στον υπολογιστή και ερμηνεύουν (interpretation) το bytecode. Για να βελτιστοποιηθεί η εκτέλεση των πολύπλοκων εφαρμογών, οι περισσότερες υλοποιήσεις εικονικής μηχανής υποστηρίζουν επίσης την λειτουργία just-in-time (JIT), η οποία επιτρέπει την ταυτόχρονη μετάφραση από bytecode στον εγγενή κώδικα μηχανής. Αυτό επιτρέπει στις χρονοβόρες εφαρμογές να εκτελούνται πολύ πιο γρήγορα, αφού η ερμηνεία του bytecode χρειάζεται μόνο κατά την έναρξη της εκτέλεσης της εφαρμογής.

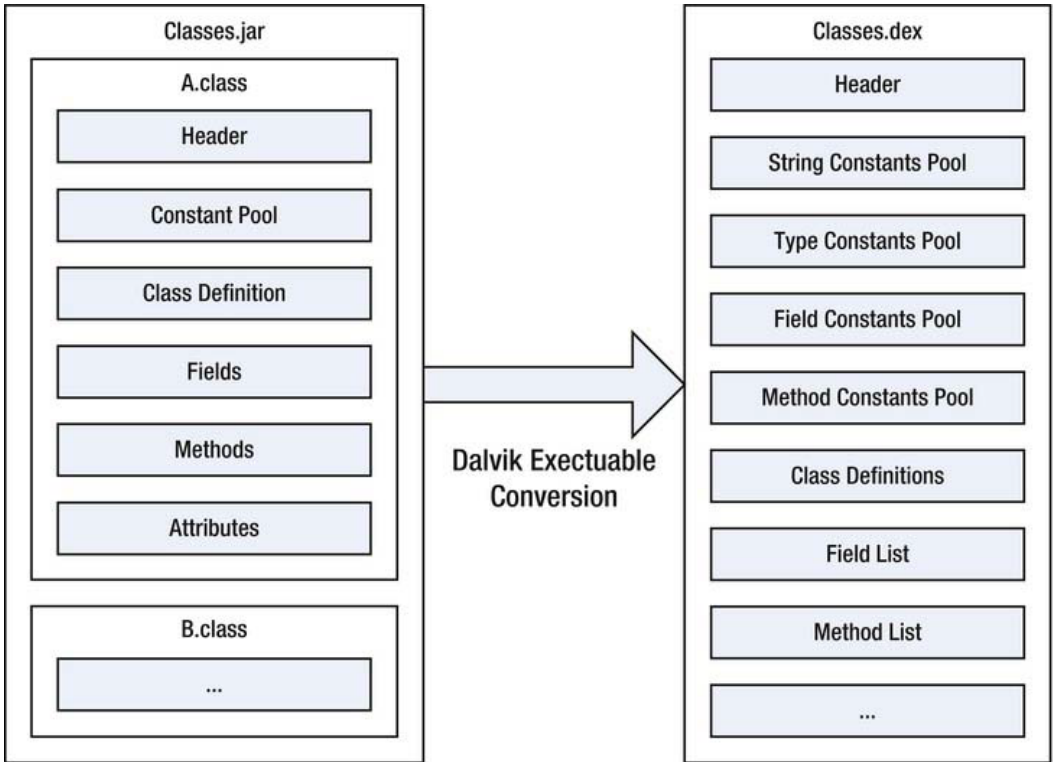
Μία από τις μεγαλύτερες προκλήσεις που αντιμετωπίζουν οι περισσότερες πλατφόρμες κινητών τηλεφώνων είναι η έλλειψη εφαρμογών. Για να λυθεί αυτό το πρόβλημα, το Android βασίζεται στην ήδη καταξιωμένη γλώσσα προγραμματισμού Java, η οποία έχει ήδη μια πολύ μεγάλη κοινότητα προγραμματιστών, καθώς και εφαρμογές, εργαλεία, εξαρτήματα, και ότι γενικότερα διευκολύνει την ανάπτυξη εφαρμογών.

Το Android βασίζεται επίσης σε μία αρκετά προσαρμοσμένη εφαρμογή εικονικής μηχανής που αφορά τις ανάγκες της κινητής τηλεφωνίας. Η Εικονική Μηχανή Dalvik (Dalvik Virtual Machine) είναι η προσαρμοσμένη στο Android έκδοση της εικονική μηχανή της Java (JVM) για κινητές πλατφόρμες.

Η εικονική μηχανή Dalvik είναι πολύ διαφορετική από τις υπόλοιπες υλοποιήσεις της Java Virtual Machine. Οι περισσότερες εφαρμογές εικονικής μηχανής στην πλατφόρμα των προσωπικών υπολογιστών έχουν αναπτυχθεί με την βασισμένη-στην-στοίβα τεχνική. Η DVM υλοποιήθηκε με την βασισμένη-στους-καταχωρητές τεχνική λόγω των αναγκών της κινητής πλατφόρμας. Η δεύτερη αυτή τεχνική απαιτεί επιπλέον οδηγίες για την ερμηνεία. Ωστόσο, ο πραγματικός αριθμός των οδηγιών που εκτελούνται είναι μικρότερος σε σύγκριση με την πρώτη. Το γεγονός αυτό καθιστά την τεχνική των καταχωρητών μια πολύ καλύτερη επιλογή για τα κινητά περιβάλλοντα στα οποία υπάρχει ανεπαρκής υπολογιστική ισχύς.

Δεδομένου ότι η DVM απαιτεί ένα διαφορετικό είδος του bytecode προς ερμηνεία, δεν υποστηρίζει τα βασικά αρχεία κλάσεων της Java αλλά βασίζεται σε δική του μορφή, η οποία ονομάζεται εκτελέσιμο Dalvik (DEX, Dalvik Executable). Η πλατφόρμα ανάπτυξης λογισμικού Android συνοδεύεται με ένα σύνολο εργαλείων για την επεξεργασία των μεταγλωττισμένων αρχείων τύπου Java σε μορφή DEX.

Η μορφή DEX παρέχει επίσης έναν πιο συμπαγή τρόπο για την αποθήκευση μεταγλωττισμένου κώδικα εφαρμογών Java στην κινητή πλατφόρμα. Βασικές εφαρμογές Java σχηματίζονται με πολλαπλά αρχεία τάξεων (java classes) που αποθηκεύονται ξεχωριστά. Το DEX συγχωνεύει όλα τα αρχεία των τάξεων σε ένα μεγάλο αρχείο DEX, όπως φαίνεται στο Σχέδιο 5. Αυτό ελαχιστοποιεί το αποτύπωμα του κώδικα εφαρμογής.



Σχέδιο 5: Μετατροπή βασικών κλάσεων Java σε μορφή DEX

Η μορφή DEX επιτρέπει σε μεταβλητές, πεδία, και σταθερές μεθόδων, και ό,τι άλλο περιλαμβάνεται στον κώδικα, να αποθηκευθούν σε ένα μόνο σημείο, χρησιμοποιώντας ευρετήρια για αυτές τις λίστες αντί των τυπικών ονομάτων. Αυτό μειώνει το μέγεθος των αρχείων τάξεων σχεδόν κατά 50%.

Η πλατφόρμα Android εκτελεί κάθε εφαρμογή στη δική του διαδικασία εικονικής μηχανής. Αυτό θέτει υψηλές απαιτήσεις στην πλατφόρμα, καθώς οι πολλαπλές εικονικές μηχανές αναμένεται να λειτουργούν ταυτόχρονα εντός περιορισμένου εύρους υπολογιστικών πόρων. Η εικονική μηχανική Dalvik είναι ειδικά ρυθμισμένη ώστε να λειτουργεί σε αυτές τις συνθήκες περιβάλλοντος.

Υποκεφάλαιο 1.3.4: Σύστημα Αρχείων

Το σύστημα αρχείων είναι ένα πολύ σημαντικό κομμάτι του λειτουργικού συστήματος. Ειδικά για κινητές πλατφόρμες, το σύστημα αρχείων έχει σημαντικό ρόλο στην ικανοποίηση των απαιτήσεων του λειτουργικού συστήματος.

Οι κινητές συσκευές βασίζονται σε μνήμες αποθήκευσης τύπου flash. Το Android έχει ως κύριο σύστημα αρχείων το Yet Another Flash File System (YAFFS2). Το YAFFS2 είναι μια ανοιχτού κώδικα υλοποίηση συστήματος αρχείων που έχει σχεδιαστεί και προγραμματιστεί από τον Charles Manning για το λειτουργικό σύστημα Linux. Το YAFFS2 είναι ένα υψηλής απόδοσης σύστημα αρχείων ειδικά σχεδιασμένο ώστε να λειτουργεί με μνήμες NAND. Κυριότερη προτεραιότητα του είναι η ακεραιότητα των δεδομένων.

Επιπρόσθετα του συστήματος αρχείων, η δομή που αφορά το πως τα αρχεία του λειτουργικού συστήματος και τα συστατικά του οργανώνονται έχει σημαντικό ρόλο στο Android. Οι κινητές πλατφόρμες αναμένεται να είναι εύκολα αναβαθμίσιμες και επίσης εξαιρετικά ασφαλείς προκειμένου για την προστασία των εμπιστευτικών πληροφοριών των χρηστών. Το Android αντιμετωπίζει αυτή την απαίτηση βασιζόμενο σε μία οργάνωση πολλών διαμερισμάτων (partitions). Χρησιμοποιώντας διαφορετικά μέρη του λειτουργικού συστήματος σε διαφορετικά διαμερίσματα, το Android παρέχει ένα υψηλό επίπεδο ασφάλειας και επίσης κάνει την πλατφόρμα εύκολα αναβαθμίσιμη.

Τα διαμερίσματα που χρησιμοποιούνται εξαρτώνται από τον κατασκευαστή της συσκευής. Τα κυριότερα είναι τα εξής:

- /boot: Αυτό το διαμέρισμα περιλαμβάνει το φορτωτή εκκίνησης (boot loader) και τον πυρήνα Linux που απαιτείται για την εκκίνηση της συσκευής. Αυτό το διαμέρισμα δεν είναι εγγράψιμο από τις εφαρμογές των χρηστών, δεδομένου ότι η τροποποίηση του περιεχομένου του μπορεί να προκαλέσει βλάβη στην εκκίνηση της συσκευής.

- /system: Αυτό το τμήμα περιέχει όλα τα αρχεία του συστήματος Android και τις προεγκατεστημένες εφαρμογές του. Κατά τη διάρκεια μιας αναβάθμισης, το διαμέρισμα αυτό αντικαθίσταται από την πιο πρόσφατη έκδοση του στην πλατφόρμα Android. Αυτό το διαμέρισμα δεν είναι εγγράψιμο από τις εφαρμογές του χρήστη, αν και η εφαρμογή Android Market μπορεί να το κάνει προσωρινά εγγράψιμο για να ενημερώνεται με τις προεγκατεστημένες εφαρμογές.

- /recovery: Αυτό το διαμέρισμα διατηρεί ένα αρχείο επαναφοράς, το οποίο είναι ένα εναλλακτικό διαμέρισμα εκκίνησης. Παρέχει λειτουργίες συντήρησης προκειμένου να ανακτηθεί το σύστημα ή να γίνουν άλλες εργασίες, όπως η δημιουργία αντιγράφων ασφαλείας του συστήματος. Αυτό το διαμέρισμα επίσης δεν είναι εγγράψιμο από τις εφαρμογές των χρηστών.

- /data: Αυτό το διαμέρισμα περιέχει τις εφαρμογές του χρήστη και επίσης τα δεδομένα του, όπως οι επαφές, τα μηνύματα και οι ρυθμίσεις. Όταν η συσκευή επαναφέρεται (reset), το διαμέρισμα αυτό διαγράφεται.

- /cache: Το διαμέρισμα αυτό χρησιμοποιείται για την αποθήκευση συχνά προσβάσιμων αρχείων. Στις περισσότερες συσκευές Android, το /cache είναι ένα εικονικό διαμέρισμα αποθηκευμένο στη μνήμη RAM. Τα περιεχόμενα αυτού του διαμερίσματος παύουν να υπάρχουν όταν η συσκευή επανεκκινείται.

- /sdcard: Αυτό είναι ένα κυρίως ένα σημείο προσάρτησης, και όχι τόσο ένα διαμέρισμα. Η κάρτα SD (Secure Digital - φορητή μνήμη) που είναι τοποθετημένη στη συσκευή προσαρτάται κάτω από αυτό το όνομα. Αυτό το σημείο προσάρτησης δεν είναι πάντα διαθέσιμο στις εφαρμογές, δεδομένου ότι μπορεί να είναι άμεσα προσαρτημένο σε έναν προσωπικό υπολογιστή όταν η συσκευή είναι συνδεδεμένη μέσω USB.

Υποκεφάλαιο 1.3.5: Ασφάλεια

Όπως και σε άλλες κινητές πλατφόρμες, η μεγαλύτερη απαίτηση για το Android, από την πλευρά των χρηστών, είναι η ασφάλεια και η ακεραιότητα των εφαρμογών και των δεδομένων τους. Το Android έχει σχεδιαστεί με γνώμονα αυτήν την έννοια της ασφάλειας.

Η αρχιτεκτονική του Android παρέχει ασφάλεια σε πολλαπλά στρώματα της πλατφόρμας. Αυτό το εκτεταμένο πλαίσιο ασφάλειας αφορά επίσης και τους προγραμματιστές κατά τις λειτουργίες εκτέλεσης του Android. Οι προγραμματιστές μπορούν εύκολα να βασίζονται στα σχετικά APIs ώστε να παρέχουν ένα υψηλό επίπεδο ασφάλειας για την εφαρμογή τους και τα δεδομένα που χρησιμοποιεί. Οι λιγότερο εξοικειωμένοι προγραμματιστές προστατεύονται, όπως και να έχει, από τις προεπιλεγμένες ρυθμίσεις ασφαλείας.

Το Android παρέχει ένα υψηλό επίπεδο ασφάλειας με τη χρήση πολλαπλών χαρακτηριστικών ασφαλείας τόσο από το υλικό όσο και από το λογισμικό. Αν και είναι σχεδιασμένο να λειτουργεί σε μια ποικιλία από πλατφόρμες υλικού, το Android μπορεί, επιπλέον, να εκμεταλλευτεί τις δυνατότητες ασφαλείας που προσφέρει συγκεκριμένος τύπος υλικού.

Η πλατφόρμα Android είναι χτισμένη πάνω στον πυρήνα του Linux. Ο πυρήνας του Linux καθεαυτός έχει ήδη χρησιμοποιηθεί σε πολλά, απαιτητικά σε θέματα ασφάλειας, περιβάλλοντα για πολλά χρόνια. Παρέχει, επίσης, στο Android πολλά βασικά χαρακτηριστικά ασφαλείας, όπως τα εξής:

- Μοντέλο αδειών με βάση τον χρήστη
- Απομόνωση διαδικασιών (process isolation)
- Ασφαλής μηχανισμός IPC (επικοινωνία εφαρμογών-συστήματος)
- Δυνατότητα να αφαιρεθεί περιττή λειτουργικότητα από τον πυρήνα

Ο πυρήνας του Linux έχει σχεδιαστεί για πλατφόρμες πολλών χρηστών (multiuser). Παρά το γεγονός ότι το Android είναι ένα μεμονωμένου-χρήστη (user-based) περιβάλλον, χρησιμοποιεί τα πλεονέκτημα του μοντέλου αδειών με βάση τον χρήστη. Το Android εκκινεί τις εφαρμογές σε μία διαδικασία εικονικής μηχανής (sandbox), και τις μεταχειρίζεται σαν διαφορετικούς χρήστες στο σύστημα. Στηριζόμενο στο user-based μοντέλο αδειών, το Android προστατεύει εύκολα το σύστημα, αποκλείοντας την πρόσβαση των εφαρμογών στα δεδομένα και την μνήμη άλλων εφαρμογών.

Με το ίδιο μοντέλο, προστατεύονται επίσης οι υπηρεσίες και οι υλικοί πόροι. Κάθε ένας από αυτούς τους πόρους έχει τη δική του ομάδα προστασίας. Κατά την ανάπτυξη των εφαρμογών, μία εφαρμογή ζητά πρόσβαση σε αυτούς τους πόρους. Εάν η αίτηση χορηγείται από το χρήστη, η εφαρμογή γίνεται μέλος της ομάδας αυτών των πόρων. Αν μία εφαρμογή δεν είναι μέλος μίας συγκεκριμένης ομάδας πόρων, δεν επιτρέπεται να έχει πρόσβαση σε οποιονδήποτε από αυτούς.

Εκτός από τα χαρακτηριστικά ασφαλείας που παρέχονται από το λειτουργικό σύστημα, το Android επίσης ενισχύει την ασφάλεια των εκτελέσιμων της πλατφόρμας χρησιμοποιώντας τον μηχανισμό ProPolice για την προστασία τους από τις επιθέσεις υπερχείλισης στοίβας (stack buffer overflow attacks).

Η προστασία του συστήματος αρχείων είναι επίσης ένα από τα νέα χαρακτηριστικά του Android, διαθέσιμα από την έκδοση 3.0. Επιτρέπει στο Android την κρυπτογράφηση του συνόλου των μέσων αποθήκευσης με τη χρήση του αλγορίθμου AES-128. Αυτό αποτρέπει την πρόσβαση τρίτων στα δεδομένα του χρήστη, η οποία είναι δυνατή μόνο μέσω του κλειδιού ασφαλείας που χρησιμοποιείται.

Η διαχείριση της συσκευής είναι ένα από τα υπόλοιπα χαρακτηριστικά ασφάλειας που είναι διαθέσιμα από την έκδοση 2.2. Επιτρέπει στους διαχειριστές να εφαρμόσουν πολιτικές ασφαλείας από απόσταση και να σβήσουν τα δεδομένα όταν η συσκευή έχει χαθεί ή κλαπεί.

Υποκεφάλαιο 1.4: Ανάπτυξη και Διανομή

Επειδή η πλατφόρμα Android είναι μια δωρεάν πλατφόρμα, δεν υπάρχει χρέωση αδειών, δικαιωμάτων, συμμετοχής, ή πιστοποίησης, ώστε να αναπτύξει και να διανείμει ο προγραμματιστής τις εφαρμογές του για την πλατφόρμα.

Η πλατφόρμα Android επιτρέπει στους προγραμματιστές εφαρμογών να αποφασίσουν τον τρόπο διανομής και τιμολόγησης των εφαρμογών τους. Οι προγραμματιστές εφαρμογών μπορούν να διανέμουν τις εφαρμογές του ως δωρεάν λογισμικό, σε δοκιμαστική χρήση, με έσοδα από διαφημίσεις, ή υπό πληρωμή.

Η πλατφόρμα Android χρησιμοποιεί ένα προεπιλεγμένο διαδικτυακό κατάστημα, το Google Play, γνωστό παλιότερα ως Android Market, το οποίο είναι ένα ηλεκτρονικό κατάστημα που αναπτύχθηκε από την Google για τις ανάγκες των συσκευών Android. Σε αντίθεση με την πλατφόρμα Android, η εφαρμογή του Android Market δεν είναι ανοιχτού λογισμικού. Είναι διαθέσιμη μόνο για συσκευές που πληρούν τις απαιτήσεις συμβατότητας της Google. Το λογισμικό για τον πελάτη υπάρχει προεγκατεστημένο στις συσκευές Android με το όνομα Market. Οι χρήστες μπορούν να χρησιμοποιήσουν αυτήν την εφαρμογή για την αναζήτηση και το κατέβασμα εφαρμογών Android. Επίσης, η εφαρμογή Market ελέγχει τακτικά ώστε να παρέχει τις νεότερες εκδόσεις των ήδη εγκατεστημένων εφαρμογών του χρήστη.

Οι προγραμματιστές εφαρμογών χρησιμοποιούν το τμήμα του διακομιστή του Android Market. Οι προγραμματιστές μπορούν να ανεβάσουν (upload) τις εφαρμογές τους στο Android Market μέσω της διαδικτυακής διεπαφής του (web-based).

Το Android Market χρησιμοποιεί μια σειρά δοκιμών για τις διαθέσιμες εφαρμογές αλλά δεν αναλαμβάνει καμία ευθύνη για τις εφαρμογές που κατεβάζουν οι χρήστες από το Android Market. Κατά τη διάρκεια της εγκατάστασης, η εφαρμογή του Android Market εμφανίζει μια λίστα δικαιωμάτων που ζητάει η εφαρμογή και ο χρήστης και μόνο δίνει την άδεια πριν προχωρήσει με την διαδικασία της εγκατάστασης.

Αν και οι περισσότερες συσκευές Android διατίθενται με την προεγκατεστημένη εφαρμογή του Android Market της Google, υποστηρίζονται και άλλες επιλογές διανομής εφαρμογών από την πλατφόρμα. Το GetJar και το Amazon Appstore είναι δύο εναλλακτικές λύσεις για αυτήν την διανομή εφαρμογών Android.

Επίλογος

Στο κεφάλαιο αυτό κάναμε μία περιεκτική και συνοπτική αναφορά στο λειτουργικό σύστημα Android. Δεν το περιγράψαμε σε βάθος καθότι θεωρούμε ότι αυτό αφορά τον ενδιαφερόμενο που θέλει να ερευνήσει λεπτομερώς την πλατφόρμα και όχι τον αναγνώστη της πτυχιακής.

Παραπέμπουμε τον ενδιαφερόμενο στην μελέτη της αρχιτεκτονικής του λειτουργικού συστήματος καθώς αυτό θα του δώσει μια ξεκάθαρη οπτική των εργαλείων που θα έχει στη διάθεση του. Φυσικά, δεν είναι απαραίτητη αυτή η μελέτη για την ανάπτυξη εφαρμογών Android αλλά ενδείκνυται στον προγραμματιστή που θέλει να ασχοληθεί σε βάθος με την πλατφόρμα.

Στο επόμενο κεφάλαιο θα αναπτύξουμε τα θέματα που αφορούν το επίσημο περιβάλλον ανάπτυξης εφαρμογών Android, το Eclipse.

Κεφάλαιο 2: Το περιβάλλον ανάπτυξης Eclipse

Εισαγωγή

Το περιβάλλον ανάπτυξης Eclipse είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης λογισμικού που χρησιμοποιείται, ως επί των πλείστον, στην ανάπτυξη εφαρμογών για το λειτουργικό σύστημα Android. Το Eclipse δεν προσφέρει απλά την δυνατότητα επεξεργασίας κώδικα. Παρέχει την υποδομή για προσθήκη επεκτάσεων κάθε προγραμματιστικής ανάγκης, είναι μια σουίτα που εξυπηρετεί την ανάπτυξη εφαρμογών και όλες τις λειτουργίες που τις αφορούν. Από αυτή την προοπτική, το κεφάλαιο αυτό είναι πολύ σημαντικό, καθώς θα καθορίσει το πλαίσιο για το επόμενο κεφάλαιο, θέτοντας την βάση για την κατανόηση της εργασίας μας και της ορθής λειτουργίας του περιβάλλοντος από κάθε ενδιαφερόμενο.

Υποκεφάλαιο 2.1: Ιστορικά Στοιχεία

Το 1995, η Sun Microsystems κυκλοφόρησε την πρώτη δημόσια εφαρμογή της γλώσσα προγραμματισμού Java για το κοινό. Στο εξής, η Java, ουσιαστικά, θα διαιρούσε την προγραμματιστική κοινότητα σε δύο ομάδες: η μία θα επικεντρωνόταν σε τεχνολογίες και εργαλεία της Microsoft, και η άλλη θα επικεντρωνόταν στην πλατφόρμα της νεοφερμένης Java.

Το περιβάλλον ανάπτυξης Visual Studio ήταν το εργαλείο της Microsoft που παρέχει πρόσβαση σε όλες τις τεχνολογίες της Microsoft με έναν ολοκληρωμένο τρόπο. Υπήρχαν πολλά επιτυχημένα εργαλεία ανάπτυξης στην αγορά για την Java αλλά δεν ήταν τόσο στενά ενσωματωμένα όσο οι τεχνολογίες της Microsoft. Στα τέλη της δεκαετίας του 1990, η IBM (International Business Machines Corporation) ήταν ένας σημαντικός παράγοντας για την Java. Ο κύριος στόχος της IBM κατά αυτήν την περίοδο ήταν να φέρει τους προγραμματιστές πιο κοντά στο ενδιάμεσο λογισμικό (middleware) της Java. Η IBM γνώριζε ότι το ιδανικό περιβάλλον ανάπτυξης πρέπει να αποτελείται από έναν ετερογενή συνδυασμό: των εργαλείων από την IBM, των εργαλείων «τρίτων εταιριών» (third-party), και από εργαλεία ενσωματωμένα από τους χρήστες. Το εργαστήριο Object Technology International (OTI) της IBM και οι παράγοντες πίσω από την οικογένεια προϊόντων VisualAge (οικογένεια λογισμικού για περιβάλλοντα ανάπτυξης) της ίδιας, είχαν ήδη εκτεταμένη εμπειρία ανάπτυξης ολοκληρωμένων περιβαλλόντων ανάπτυξης.

Ως πρώτο βήμα, η VisualAge για την έκδοση Java Micro Edition αναπτύχθηκε ως ένα εκ νέου υλοποιημένο ολοκληρωμένο περιβάλλον ανάπτυξης, αποκλειστικά με τη χρήση της γλώσσας προγραμματισμού Java. Αργότερα, ο κώδικας αυτής της έκδοσης της VisualAge χρησιμοποιήθηκε ως βάση για την πλατφόρμα Eclipse.

Η IBM γνώριζε ήδη ότι έχοντας μόνο τα προϊόντα της ίδιας για αυτή τη νέα πλατφόρμα δεν θα κατάφερνε μία ευρεία υιοθέτηση της από την κοινότητα των προγραμματιστών. Το κλειδί για την προσπέραση αυτού του γεγονότος και την επιτυχία της νέας προσπάθειας (Eclipse) ήταν η ενσωμάτωση εργαλείων από «τρίτες εταιρίες». Το 2001, η IBM αποφάσισε να υιοθετήσει την Γενική Άδεια

Δημόσιας Χρήσης (General Public License) και το αντίστοιχο λειτουργικό μοντέλο για την πλατφόρμα Eclipse. Η IBM, μαζί με οκτώ άλλους οργανισμούς, ίδρυσε την κοινοπραξία Eclipse. Η βασική αρχή λειτουργίας της κοινοπραξίας ήταν να οδηγήσει την διαφήμιση και προώθηση για την πλατφόρμα Eclipse, αφήνοντας το τον έλεγχο και την ανάπτυξη του πηγαίου κώδικα του Eclipse στην κοινότητα του ανοικτού κώδικα.

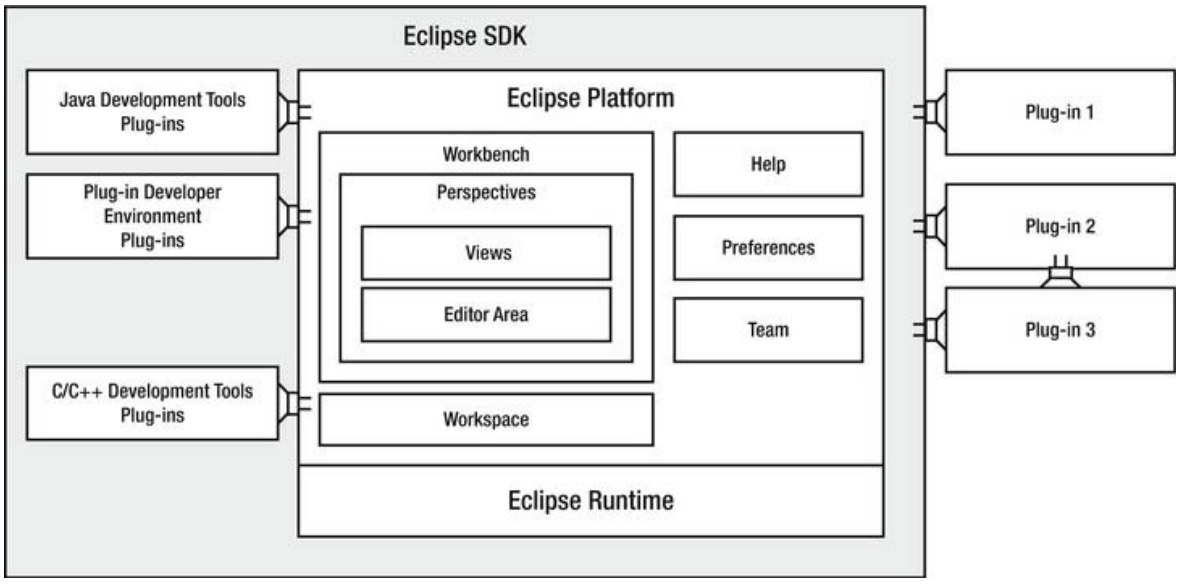
Το 2003, η πλατφόρμα Eclipse, με το ταχέως αναπτυσσόμενο σύνολο των επεκτάσεων ανοιχτού λογισμικού και αντίστοιχων εμπορικών, γινόταν δημοφιλής στους προγραμματιστές. Το 2004, το Eclipse Foundation, μία μη κερδοσκοπική οργάνωση με το δικό της επαγγελματικό και ανεξάρτητο προσωπικό, ανέλαβε τον πλήρη έλεγχο της πλατφόρμας Eclipse. Το Eclipse είναι πλέον το μεγαλύτερο και σημαντικότερο περιβάλλον ανάπτυξης για την γλώσσα προγραμματισμού Java. Λόγω της ιδιαίτερης και επεκτάσιμης αρχιτεκτονική του, χρησιμοποιείται επίσης ως περιβάλλον ανάπτυξης για πολλές άλλες γλώσσες προγραμματισμού.

Υποκεφάλαιο 2.2: Αρχιτεκτονική Eclipse

Ως προγραμματιστές Android, δεν είναι απαραίτητο να αλληλεπιδράσουμε με τα εσωτερικά χαρακτηριστικά της πλατφόρμας Eclipse. Ωστόσο, γνωρίζοντας την αρχιτεκτονική του θα καταστήσει πολύ πιο εύκολο να αντιληφθούμε και να κατανοήσουμε πώς λειτουργεί ο Eclipse σε γενικές γραμμές.

Η πλατφόρμα Eclipse έχει σχεδιαστεί κυρίως για τη δημιουργία ολοκληρωμένων περιβαλλόντων ανάπτυξης. Είναι μια εξαιρετικά επεκτάσιμη πλατφόρμα, και όχι απλά ένα προσαρμοσμένο εργαλείο για ένα συγκεκριμένο σύνολο εργασιών.

Η πλατφόρμα Eclipse καθορίζει τους μηχανισμούς και τους κανόνες, και επιτρέπει στα εργαλεία να χτίσουν πάνω τους, παρέχοντας μια σειρά σαφώς καθορισμένων διεπαφών προγραμματισμού εφαρμογών (APIs). Η πλατφόρμα είναι δομημένη γύρω από την έννοια των προσθέτων (plug-ins), όπως φαίνεται στο Σχήμα 1.



Σχέδιο 6: Επισκόπηση αρχιτεκτονικής της πλατφόρμας Eclipse

Τα πρόσθετα (plug-ins) είναι η μικρότερη μονάδα της πλατφόρμας Eclipse. Είναι δομημένες δέσμες του κώδικα που προσφέρουν ένα σύνολο λειτουργιών στην πλατφόρμα. Τα πρόσθετα αυτά μπορούν να αναπτυχθούν, να διανεμηθούν, και να αποσταχθούν μεμονωμένα. Η πλατφόρμα Eclipse επιτρέπει τα πρόσθετα να είναι επεκτάσιμα επίσης. Αυτά, μπορούν να παρέχουν ένα σύνολο από σημεία επέκτασης, μέσα από μια καλά καθορισμένη διεπαφή προγραμματισμού εφαρμογών (API), έτσι ώστε άλλα πρόσθετα να επεκτείνουν τις λειτουργίες τους.

Κάθε υποσύστημα στην πλατφόρμα Eclipse βασίζεται σε μια σειρά από πρόσθετα. Για παράδειγμα, τα βασικά εργαλεία ανάπτυξης της Java (Java Development Tools) είναι ένα σύνολο από πρόσθετα που παρέχουν το σύνολο των δυνατοτήτων ανάπτυξης στην γλώσσα Java με ολοκληρωμένο τρόπο. Τα πρόσθετα αυτών των εργαλείων ανάπτυξης Java είναι επίσης επεκτάσιμα.

Βασισμένοι στην ίδια λογική, σε επόμενο υποκεφάλαιο, θα αναλύσουμε τα σχετικά πρόσθετα της πλατφόρμας Eclipse που παρέχουν τα απαραίτητα εργαλεία για την ανάπτυξη εφαρμογών σε περιβάλλον Android. Αυτά, δηλαδή, επεκτείνουν με την σειρά τους τα υπάρχοντα εργαλεία ανάπτυξης της Java, προκειμένου να επεκτείνουν και την λειτουργικότητα της.

Ο πυρήνας της πλατφόρμας Eclipse, γνωστός ως Eclipse runtime, είναι υπεύθυνός για την παροχή της υποδομής όπου τα πρόσθετα μπορούν να λειτουργήσουν και να διαλειτουργήσουν μεταξύ τους. Το Eclipse runtime παρέχει επίσης τα απαιραίτητα εργαλεία που κάνουν την ανάπτυξη νέων προσθέτων ευκολότερη για τους προγραμματιστές.

Κατά την διάρκεια ανάπτυξης της εφαρμογής μας, χρησιμοποιήσαμε την έκδοση Indigo 3.7.2. Η τελευταία έκδοση του Eclipse που κυκλοφορεί, την στιγμή της συγγραφής της πτυχιακής, είναι η Juno 4.2.

Υποκεφάλαιο 2.3: Εγκατάσταση και διαμόρφωση

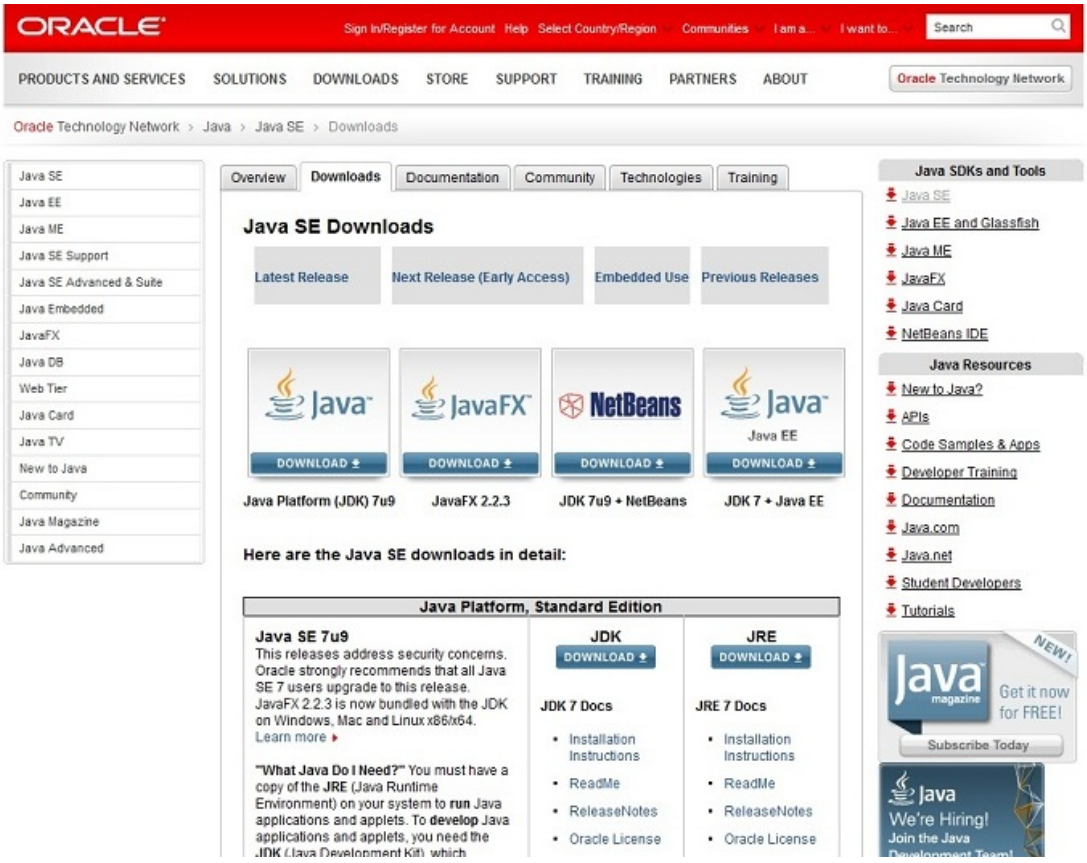
Σε αυτό το υποκεφάλαιο θα δείξουμε τα βήματα για την εγκατάσταση του απαραίτητου λογισμικού. Αποτελεί την ίδια διαδικασία που ακολουθήσαμε κι εμείς, με το πέρας της οποίας ο χρήστης είναι έτοιμος να ξεκινήσει την ανάπτυξη μιας εφαρμογής Android.

Πιο συγκεκριμένα, αρχικά, θα εγκαταστήσουμε το Java Development Kit (SDK), την βάση μας στην γλώσσα προγραμματισμού και το απαραίτητο συστατικό για την λειτουργία του περιβάλλοντος ανάπτυξης Eclipse. Έπειτα, θα κάνουμε μια σχετική ρύθμιση που απαιτεί η Java και θα εγκαταστήσουμε το Eclipse. Τέλος, θα ενσωματώσουμε τα εργαλεία για την ανάπτυξη της εφαρμογής: το πρόσθετο Android Development Tools (ADT) για τα εργαλεία ανάπτυξης Android και το πρόσθετο EGit που μας δίνει την δυνατότητα χρήσης των υπηρεσιών του GitHub.

Οι οδηγίες που ακολουθούν υποδεικνύουν την διαδικασία σε λειτουργικό σύστημα Windows 7. Το σύνολο των παραπάνω εργαλείων είναι διαθέσιμο σε όλα τα γνωστά λειτουργικά συστήματα, όπως: διανομές Linux, Mac OS X, Microsoft Windows, Solaris.

Υποκεφάλαιο 2.3.1: Εγκατάσταση Java SDK

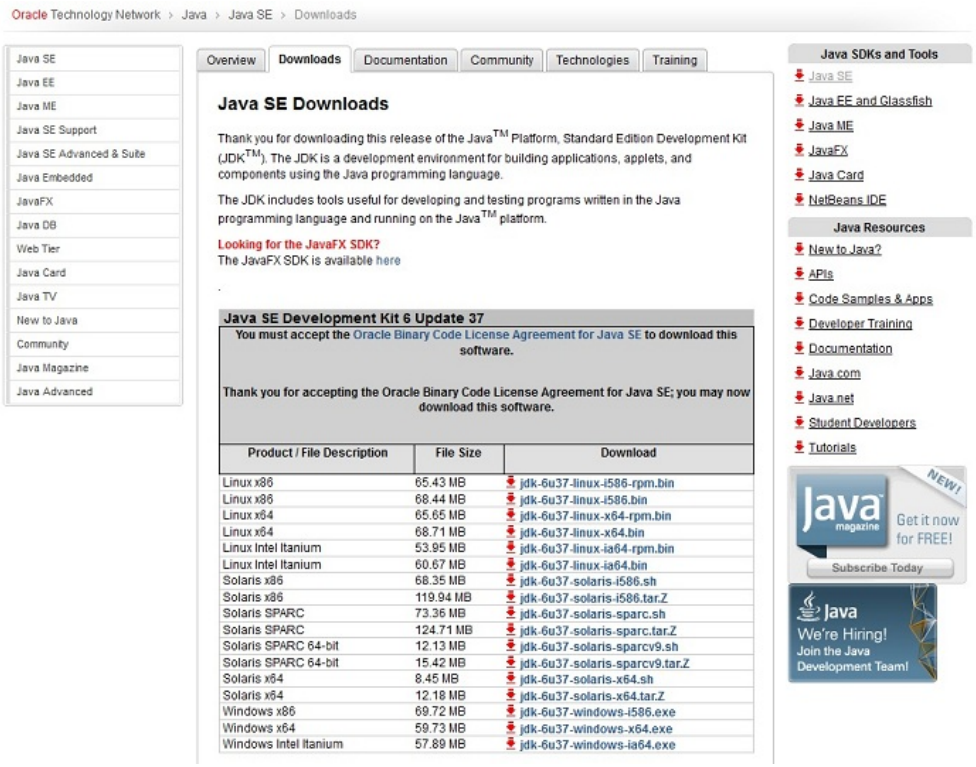
Χρησιμοποιώντας τον φυλλομετρητή μας, πηγαίνουμε στην ιστοσελίδα της Oracle: http://www.oracle.com/technetwork/java/javase/downloads/index.html (Σχέδιο 2) όπου υπάρχουν οι εκδόσεις του SDK. Την στιγμή της συγγραφής της πτυχιακής, η τελευταία έκδοση του SDK 6 ήταν η 37 (update 37).



Σχέδιο 7: Ιστοσελίδα της Oracle για το Java SDK

Επιλέγουμε και κατεβάζουμε (Download) την τελευταία έκδοση του Java SE 6 η οποία βρίσκεται στη λίστα της παραπάνω ιστοσελίδας.

Μετά την επιλογή, οδηγούμαστε σε μία λίστα με τις εκδόσεις του SDK για τα διάφορα λειτουργικά συστήματα (Σχέδιο 3). Κατεβάζουμε αυτή που χρειαζόμαστε.

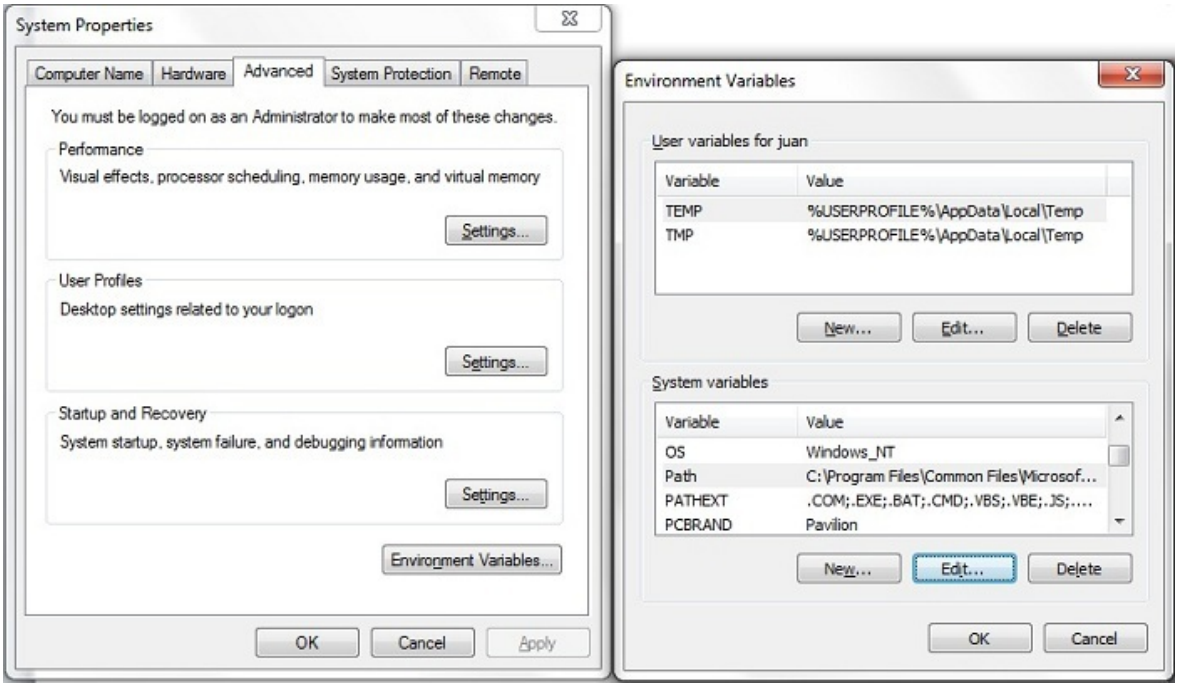


Σχέδιο 8: Εκδόσεις του SDK για κάθε ΛΣ

Η εγκατάσταση που ακολουθείται για το αρχείο που προκύπτει είναι η τυπική (Σχέδιο 4). Στο λειτουργικό σύστημα Windows, μετά την εγκατάσταση απαιτείται η ρύθμιση μίας παραμέτρου.

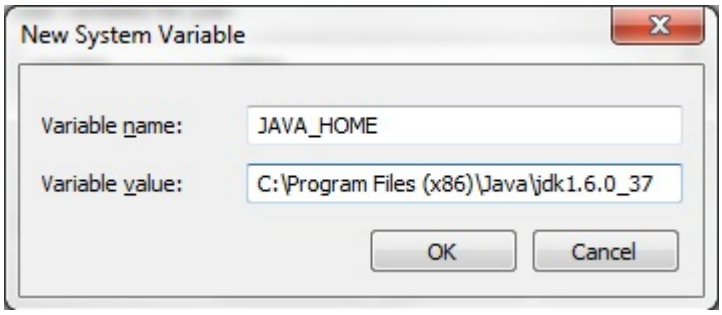
Σχέδιο 9: Εγκατάσταση Java SDK

Απαιτείται τώρα να συνδέσουμε τις βιβλιοθήκες της Java με την διαδρομή που χρειάζεται το λειτουργικό σύστημα. Ακολουθούμε την εξής διαδικασία: επιλέγουμε την Εκτέλεση (Run) από το κεντρικό μενού και εισάγουμε στο κενό την εντολή *sysdm.cpl*. Αυτό μας οδηγεί στο παράθυρο των ρυθμίσεων συστήματος (Σχέδιο 5) όπου στην καρτέλα *Για προχωρημένους* (Advanced) επιλέγουμε το Μεταβλητές Περιβάλλοντος (Environment Variables).



Σχέδιο 10: Ρυθμίσεις συστήματος - Μεταβλητές περιβάλλοντος

Έπειτα επιλέγουμε το κουμπί Δημιουργία (New) και εισάγουμε στα δύο πεδία: JAVA_HOME και την διαδρομή εγκατάστασης της Java στον υπολογιστή μας αντίστοιχα (Σχέδιο 6).

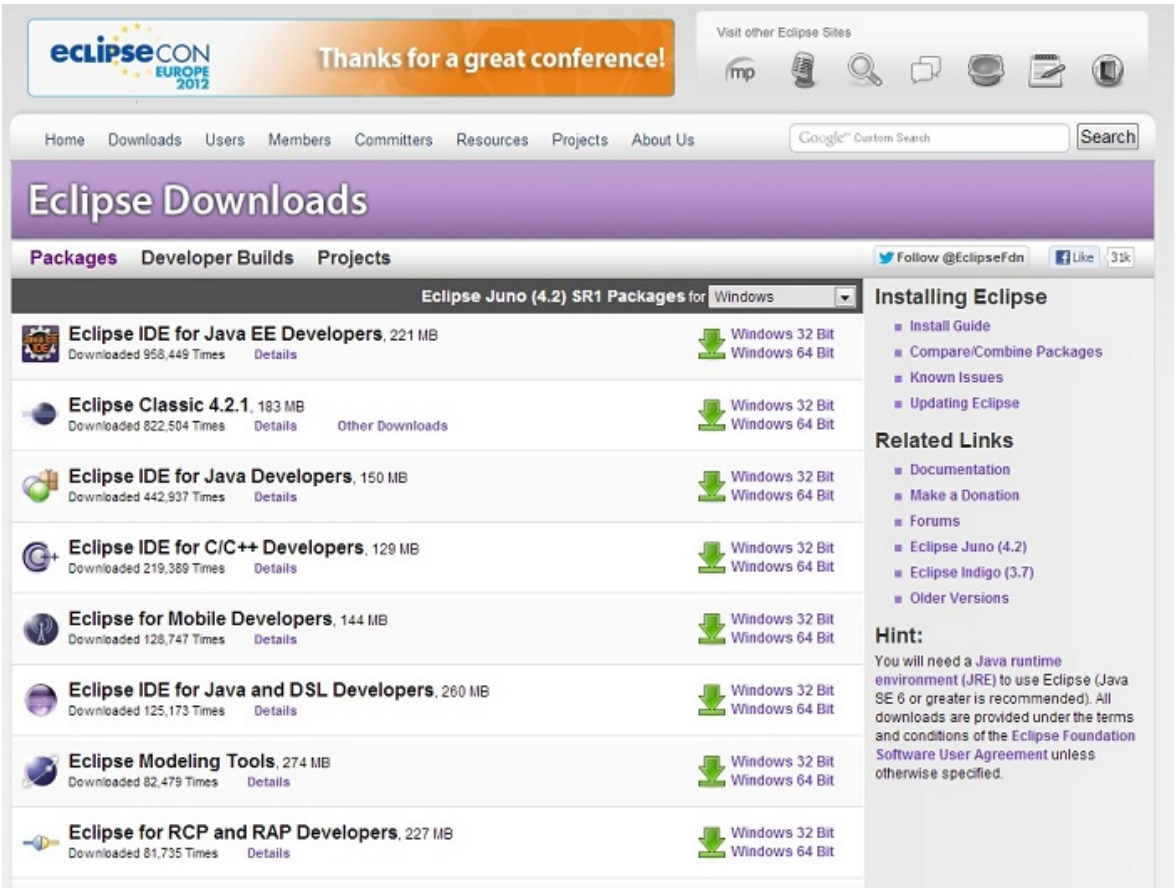


Σχέδιο 11: Προσθήκη μεταβλητής περιβάλλοντος

Τέλος, αφού πατήσουμε *OK*, επιλέγουμε *Επεξεργασία* (Edit) της μεταβλητής συστήματος Path και επεκτείνουμε το δεύτερο πεδίο με το κείμενο *;%JAVA_HOME%\bin*.

Υποκεφάλαιο 2.3.2: Εγκατάσταση Eclipse

Για την εγκατάσταση του Eclipse κατεβάζουμε την τελευταία έκδοση από την ιστοσελίδα http://www.eclipse.org/downloads/. Εμείς χρησιμοποιήσαμε την έκδοση 3.7.2 (Indigo). Κατά την συγγραφή της πτυχιακής εργασίας, τελευταία διαθέσιμη έκδοση ήταν η 4.2.1. Επιλέγουμε την έκδοση που χρειαζόμαστε (υπάρχουν επιλογές για Windows, Linux, Mac OS X). Για την παρουσίαση θα δείξουμε την διαδικασία για λειτουργικό σύστημα Windows (Σχέδιο 7).

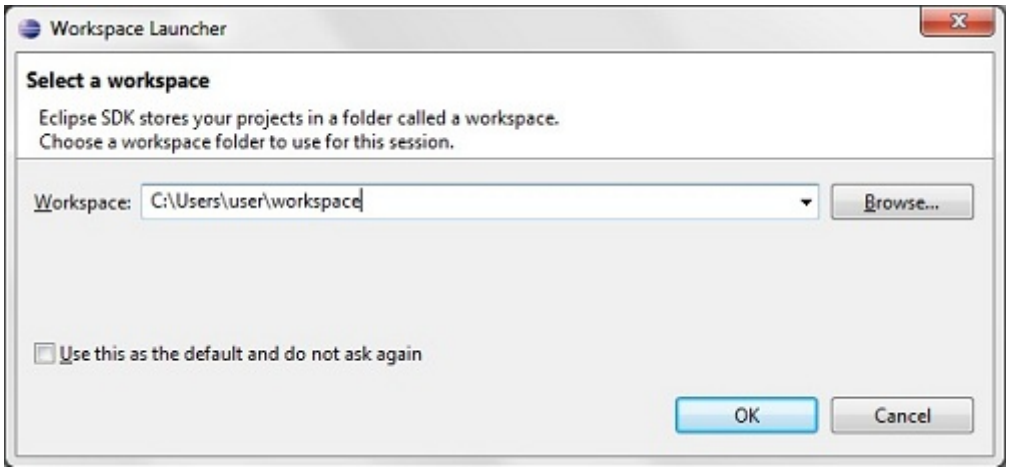


Σχέδιο 12: Κατέβασμα του Eclipse

Το αρχείο που προκύπτει είναι συμπιεσμένο (.zip). Το μεταφέρουμε στον δίσκο C: και το αποσυμπιέζουμε.

Με το πέρας της διαδικασίας το εκτελέσιμο αρχείο για την εφαρμογή θα είναι, στην περίπτωση μας, στον φάκελο C:\eclipse\, με το όνομα *eclipse*. Μπορούμε να δημιουργήσουμε μια συντόμευση του στην Επιφάνεια Εργασίας, για ευκολότερη πρόσβαση.

Η πρώτη εκκίνηση του Eclipse θα ζητήσει μια διαδρομή όπου θα αποθηκεύει τις εργασίες που αναπτύσσονται με αυτό. Επιλέγουμε OK (Σχέδιο 8) και το περιβάλλον είναι έτοιμο για χρήση.



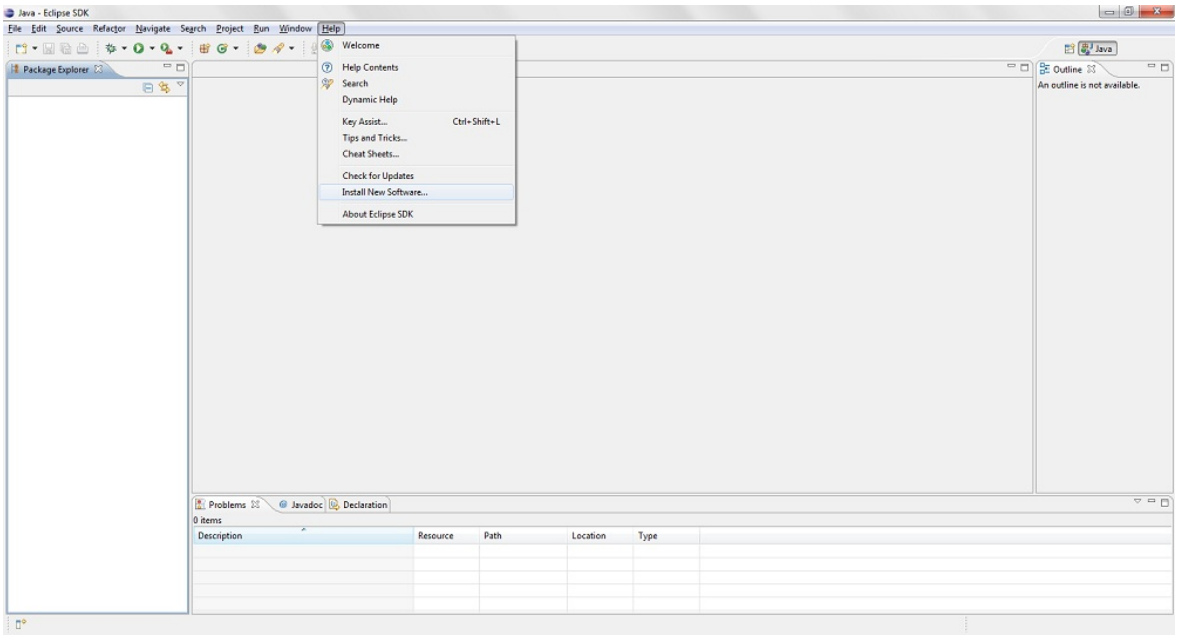
Σχέδιο 13: Ορισμός διαδρομής εργασιών

Υποκεφάλαιο 2.3.3: Ενσωμάτωση Andoird Development Tools

Αν και το περιβάλλον Eclipse περιλαμβάνει τα εργαλεία για την ανάπτυξη εφαρμογών στην γλώσσα προγραμματισμού Java, απαιτούνται κάποια επιπρόσθετα εργαλεία για τον προγραμματισμό Android. Σε αυτό το υποκεφάλαιο θα δείξουμε την διαδικασία ενσωμάτωσης των Andoird Development Tools (ADT) στα εργαλεία του Eclipse.

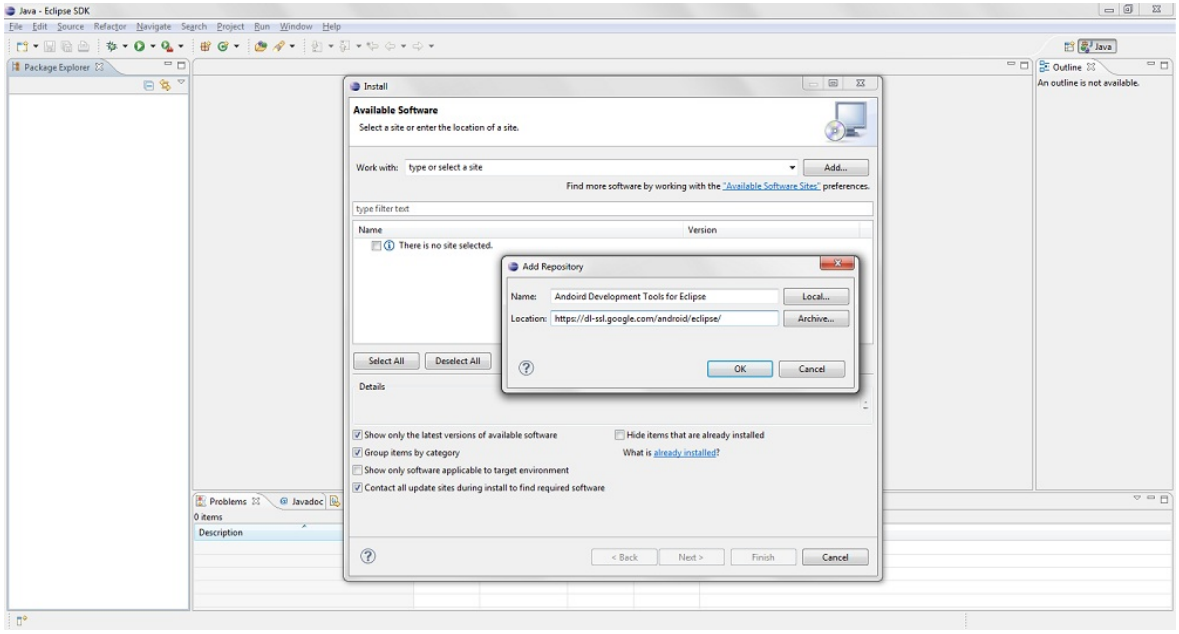
Τα ADT είναι ένα πρόσθετο, στην λογική που αναπτύξαμε στο υποκεφάλαιο 2.2, που επεκτείνει τις υπάρχουσες δυνατότητες. Θα χρησιμοποιήσουμε τον οδηγό εγκατάστασης του Eclipse για την ενσωμάτωση τους, η οποία έχει ως εξής:

Επιλέγουμε (Σχέδιο 9) από το κεντρικό μενού Βοήθεια (Help), Εγκατάσταση Νέου Λογισμικού (Install New Software).



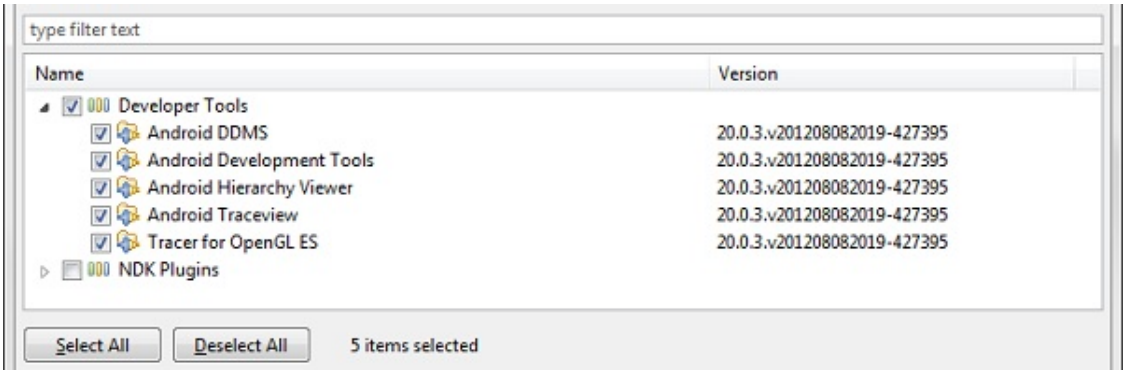
Σχέδιο 14: Επιλογή για εγκατάσταση ADT

Έπειτα, επιλέγουμε Προσθήκη (Add), και στο παράθυρο που εμφανίζεται, εισάγουμε ένα όνομα για το πρόσθετο (π.χ. Andoird Development Tools for Eclipse) και την διεύθυνση https://dl-ssl.google.com/android/eclipse/ αντίστοιχα στα δύο πεδία (Σχέδιο 10).



Σχέδιο 15: Εισαγωγή πηγής λογισμικού

Επιλέγουμε OK και στο παράθυρο μας εμφανίζεται μία λίστα με το όνομα Developer Tools. Την επιλέγουμε (τσεκάροντας έτσι όλα τα περιεχόμενα της – Σχέδιο 11) και προχωράμε επιλέγοντας *Επόμενο* (Next) δύο φορές.

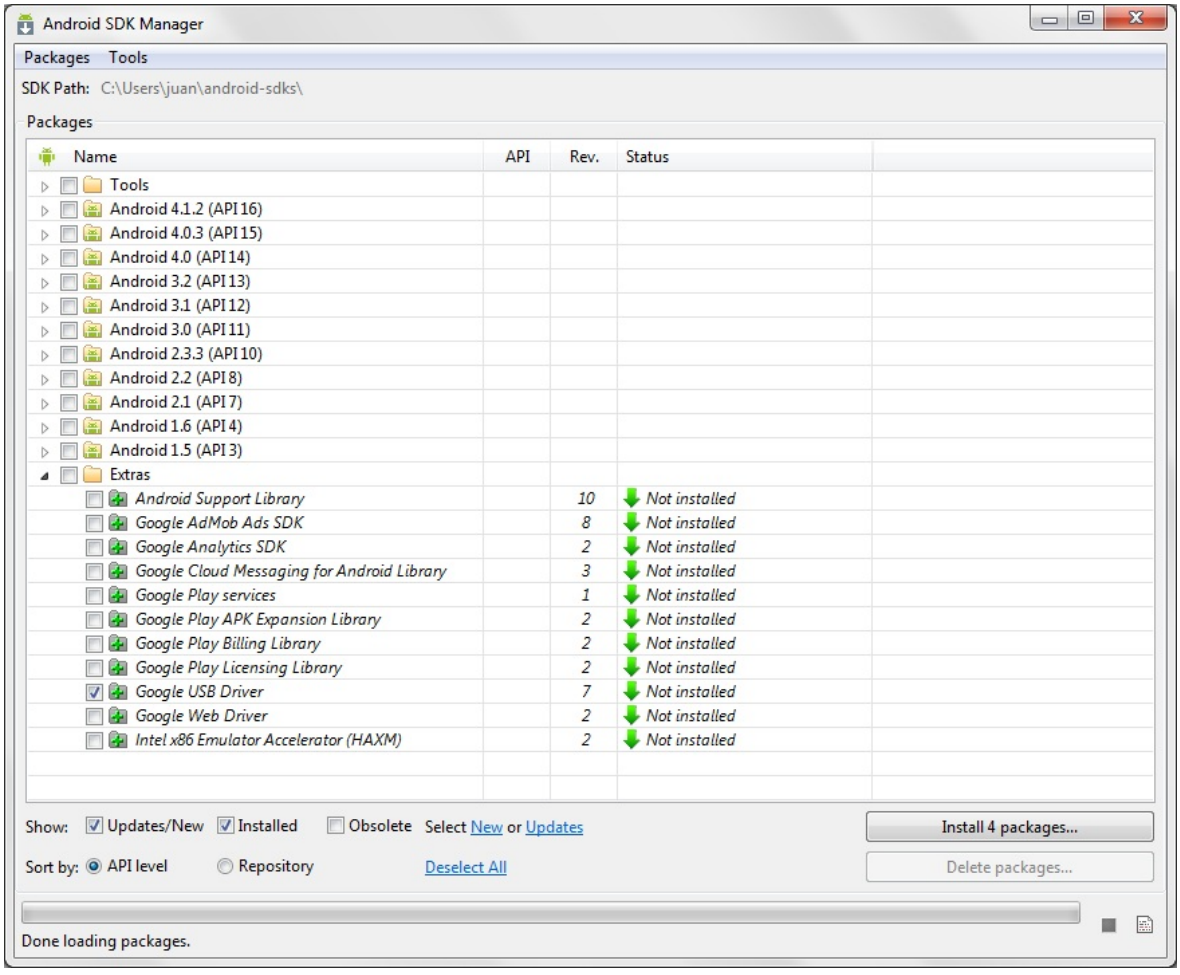


Σχέδιο 16: Επιλογή εγκατάστασης εργαλείων ανάπτυξης

Στο επόμενο παράθυρο θα πρέπει να αποδεχτούμε τους όρους χρήσης του λογισμικού που εγκαθιστάμε κάνοντας την σχετική επιλογή και, έπεται, επιλέγουμε *Τέλος* (Finished).

Ο Eclipse θα προχωρήσει στην διαδικασία ενσωμάτωσης και αφού την ολοκληρώσει θα απαιτήσει επανεκκίνηση της εφαρμογής. Επιλέγουμε *Επανεκκίνηση Τώρα* (Restart Now) και η διαδικασία έχει ολοκληρωθεί.

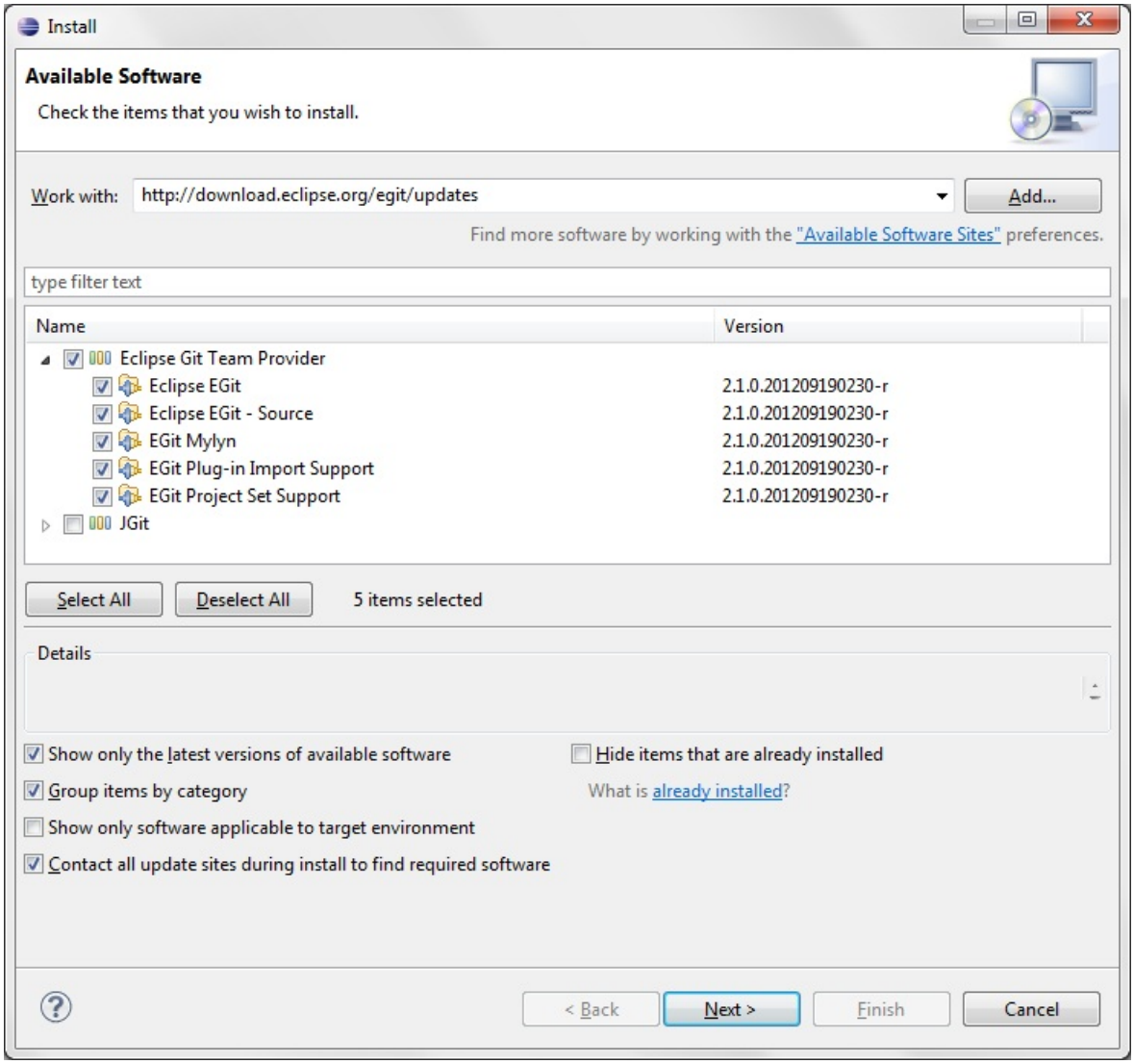
Τέλος, στην επιλογή Παράθυρο (Window), Android SDK Manager (Σχέδιο 12) επιλέγουμε την έκδοση του Android στην οποία θέλουμε να αναπτύξουμε την εφαρμογή μας και μετά επιλέγουμε *Εγκατάσταση Πακέτων*. Εμείς, για την πτυχιακή εργασία, χρησιμοποιήσαμε την έκδοση 2.1.



Σχέδιο 17: Επιλογή έκδοσης Android

Υποκεφάλαιο 2.3.4: Ενσωμάτωση EGit Plugin

Ακολουθώντας την διαδικασία που περιγράψαμε στο υποκεφάλαιο 2.3.3 (Σχέδιο 10 & 11), επιλέγουμε τα απαραίτητα πακέτα για το EGit (Σχέδιο 13) και τα εγκαθιστούμε.



Σχέδιο 18: Εγκατάσταση EGit

Ένας οδηγός για τα πρώτα βήματα συνεργατικού προγραμματισμού με χρήση της υπηρεσίας Git (εμείς χρησιμοποιήσαμε το GitHub) υπάρχει στη διεύθυνση http://www.vogella.com/articles/EGit/article.html.

Υποκεφάλαιο 2.4: Χαρακτηριστικά περιβάλλοντος Eclipse

Τα βασικά χαρακτηριστικά του περιβάλλοντος Eclipse είναι τα εξής μέρη: Menus, Toolbars, Toolbars, Editors, Views, Projects, Perspectives.

Menus: είναι η τυπική διάταξη των μενού, γνωστή σε όλα τα λειτουργικά συστήματα. Στο περιβάλλον Eclipse, εκτός από το κεντρικό οριζόντιο μενού στο πάνω μέρος, υπάρχει παρόμοια λειτουργία στο παράθυρο των Project καθώς και σε κάθε project ξεχωριστά –για εξειδικευμένες λειτουργίες κάθε εργασίας, και σε κάθε καρτέλα του κεντρικού επεξεργαστή κώδικα (Editors). Τα μενού αυτά παρέχουν τις βασικές εντολές που θα χρειαστεί ο χρήστης.

Toolbars: είναι συντομεύσεις που βοηθούν τον χρήστη στις πιο συχνά εκτελούμενες ενέργειες. Στο περιβάλλον του Eclipse, βρίσκονται στην τυπική οριζόντια στοίχιση, κάτω από το κεντρικό μενού, στο πάνω μέρος του παραθύρου των Project, και για κάθε ελαχιστοποιημένο παράθυρο στο δεξί άκρο της διεπαφής.

Editors: αποτελεί το κύριο συστατικό του περιβάλλοντος και αφορά την επεξεργασία κάθε αρχείου του project. Κάθε τύπος αρχείου (java, xml, text κοκ.) έχει τον κατάλληλο χειρισμό από το Eclipse. Εδώ γίνεται η συγγραφή του κώδικα, ο σχεδιασμός των διεπαφών, η επεξεργασία διαγραμμάτων κ.α.

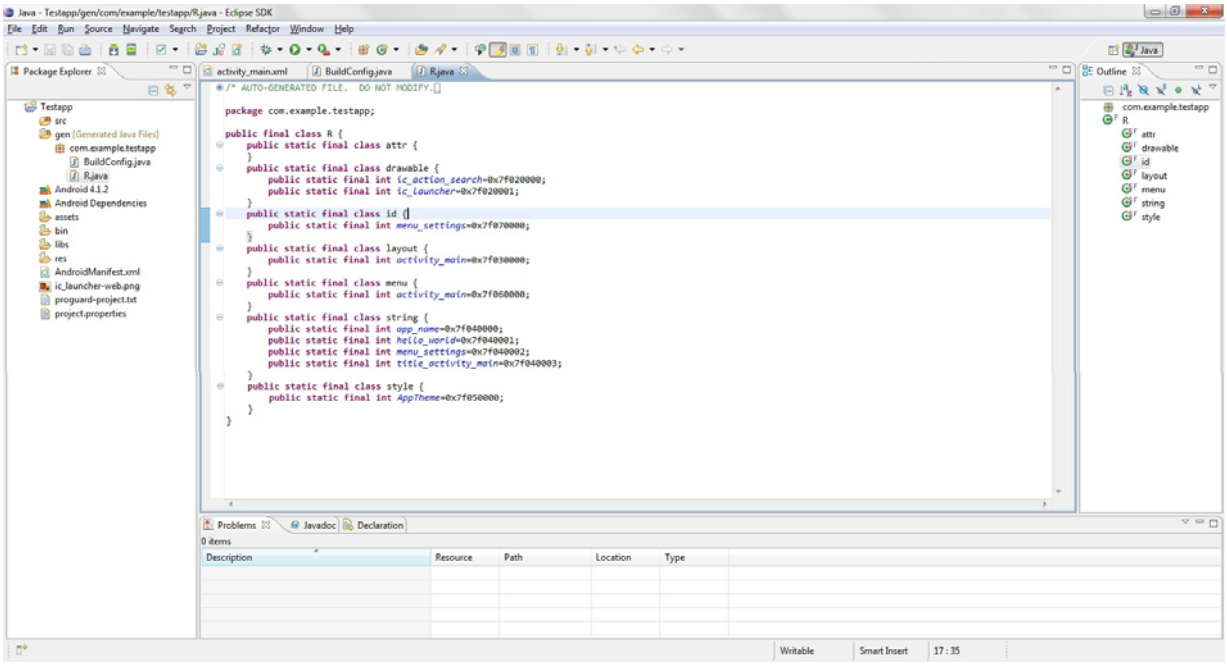
Views: είναι εργαλεία του Eclipse που παρέχουν εναλλακτικές εμφανίσεις για την εφαρμογή και τους επεξεργαστές κειμένου, δίνοντας πληροφορίες ανάλογα με τις ανάγκες του χρήστη. Υπάρχουν πολλές επιλογές και βρίσκονται στο μενού Window, Show View.

Projects: είναι το παράθυρο στα αριστερά της διεπαφής που περιλαμβάνει τα project (μια εφαρμογή για παράδειγμα) και τα αντίστοιχα αρχεία τους (κώδικας, πολυμεσικό υλικό, ρυθμίσεις). Η εμφάνιση τους είναι δενδρική (tree view).

Perspectives: αποτελούν ένα σύνολο προεπιλεγμένων δομών εμφάνισης που εξυπηρετούν αντίστοιχες ανάγκες. Κεντρικές είναι αύτη του προγραμματισμού (Java) και αυτή της αποσφαλμάτωσης (Debug). Βρίσκεται στην πάνω δεξιά περιοχή του περιβάλλοντος.

Υποκεφάλαιο 2.4.1: Το περιβάλλον Eclipse

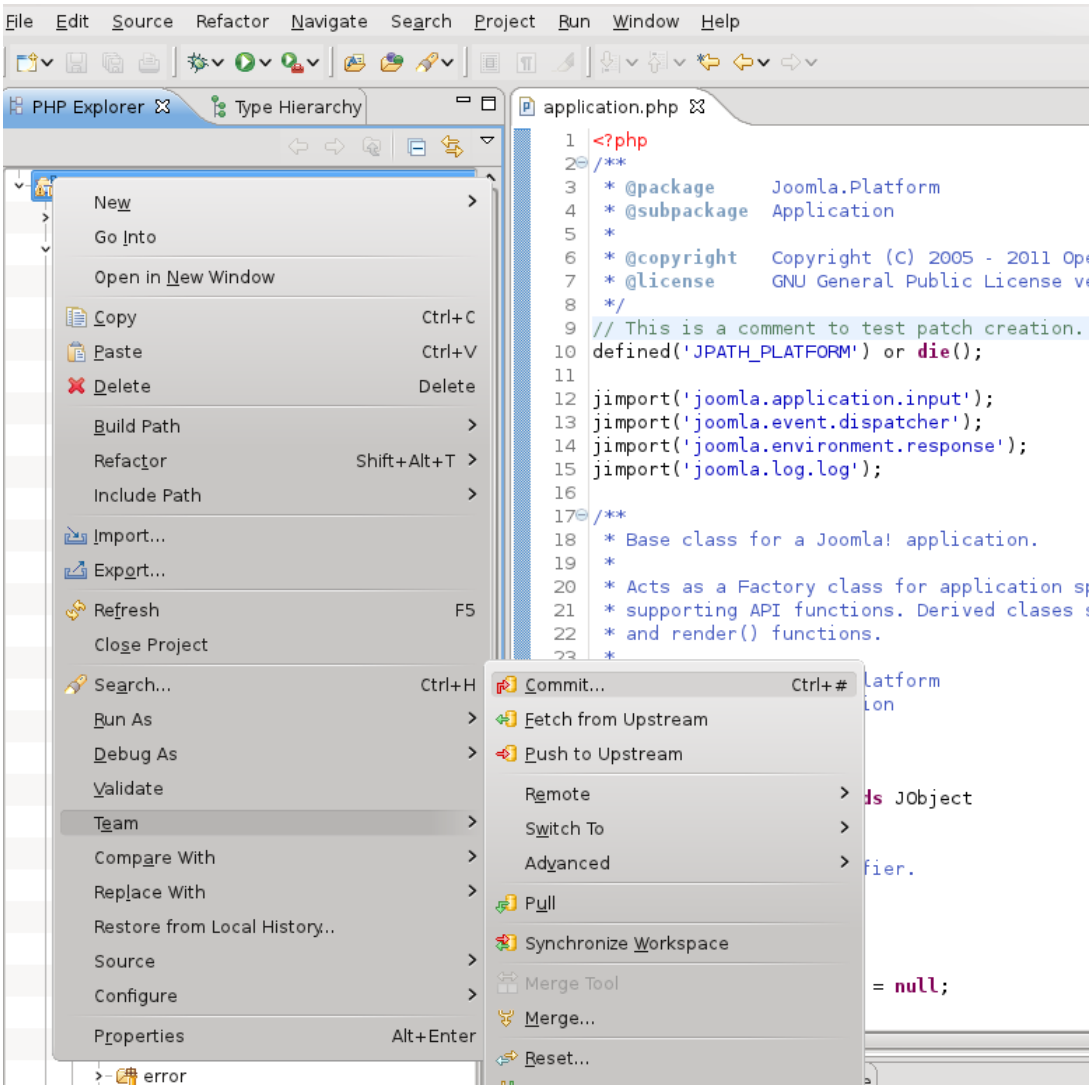
Το σύνολο των μερών που αναλύσαμε παραπάνω παρουσιάζεται στην εικόνα που ακολουθεί (Σχέδιο 14). Στο πάνω μέρος, το κεντρικό μενού λειτουργιών. Αμέσως μετά η μπάρα των συντομεύσεων (toolbar) και ακολουθούν αριστερά το παράθυρο του project με μια λίστα των αρχείων του, το κεντρικό παράθυρο επεξεργασίας/προγραμματισμού, και, τέλος, στα δεξιά, ένα πρόσθετο παράθυρο από τα πολλά που διατίθενται (Views).



Σχέδιο 19: Περιβάλλον Eclipse

Υποκεφάλαιο 2.4.2: Οι λειτουργίες του EGit plugin

Αφού έχουμε ενσωματώσει την υπηρεσία Git της επιλογής μας[4], έχουμε τις εξής βασικές λειτουργίες μέσα από το περιβάλλον του Eclipse: *Commit* (η κατάθεση δηλαδή των αλλαγών που έχουμε κάνει στην εφαρμογή), *Pull* (η προσκόμιση των αλλαγών που έχουν κατατεθεί από άλλους), και *Push to Upstream* (αποστολή αλλαγών στον εξυπηρετητή ώστε να διατεθεί και στους άλλους προγραμματιστές). Οι επιλογές αυτές υπάρχουν στην επιλογή *Team* κάνοντας δεξί κλικ στο όνομα του Project μας (της εφαρμογής) στο αριστερό μέρος της διεπαφής του Eclipse (Σχέδιο 15).



Σχέδιο 20: Επιλογές υπηρεσίας Git

Επίλογος

Στο κεφάλαιο αυτό αναφερθήκαμε στο περιβάλλον ανάπτυξης Eclipse. Υποδείξαμε τον τρόπο εγκατάστασης και λειτουργίας του ώστε να υπάρχει μία βασική γνώση για αυτό το εύχρηστο εργαλείο προγραμματισμού που αποτελεί ελεύθερο λογισμικό / λογισμικό ανοιχτού κώδικα.

Είναι το πιο διαδεδομένο λογισμικό για ανάπτυξη εφαρμογών στην πλατφόρμα Android και αυτό μας παρέχει μια μεγάλη κοινότητα που απασχολείται και βοηθά τους χρήστες σε όποια ανάγκη προκύψει. Υπάρχει μεγάλος αριθμός διαδικτυακών βοηθημάτων που αφορούν το Eclipse και προτείνουμε στους χρήστες του να εξερευνήσουν την δουλειά των μελών που δραστηριοποιούνται με αυτό.

Στην βιβλιογραφία της πτυχιακής εργασίας έχουμε προσθέσει ένα πολύ καλό βιβλίο που θα φανεί πολύ χρήσιμο σε έναν αρχάριο χρήστη.

Κεφάλαιο 3: Η εφαρμογή

Εισαγωγή

Στο κεφάλαιο αυτό θα εστιάσουμε στα τεχνικά χαρακτηριστικά της εφαρμογής αναλύοντας κάθε τμήμα της ξεχωριστά. Θα αναφερθούμε κυρίως στον κώδικα της εφαρμογής επεξηγώντας τον κατάλληλα.

Αρχικά θα αναφερθούμε στον μηχανισμό που αναπτύξαμε για να έχουμε πρόσβαση στις υπηρεσίες του τμήματός μας καθώς δεν υπάρχει κάποιο API (Application Programming Interface) το οποίο να μας δίνει αυτή τη δυνατότητα, και στη συνέχεια θα αναφερθούμε στον κώδικα κάθε κλάσης ξεχωριστά. Επειδή είναι αδύνατον να συμπεριλάβουμε όλον τον κώδικα της εφαρμογής στο παρόν έγγραφο, θα παρουσιάσουμε τα μέρη του κώδικα που θεωρούμε ότι είναι σημαντικά να απαραίτητα.

Πέρα από τον μηχανισμό που αναπτύξαμε για τις υπηρεσίες του τμήματός μας, θα αναφερθούμε και στο API που αναπτύξαμε για τις επιπρόσθετες υπηρεσίες (πληρότητα γραμμής 52, συνομιλία φοιτητών του τμήματός μας – Υποκεφάλαιο 3.6) όπου εκεί κάνουμε χρήση των PHP και MySQL.

Ευελπιστούμε το κεφάλαιο αυτό να βοηθήσει τους μελλοντικούς φοιτητές του τμήματος μας να αντιληφθούν την δομή και τις λειτουργίες της εφαρμογής ώστε να μπορέσουν να την επεκτείνουν και να την βελτιστοποιήσουν καθώς οι υπηρεσίες του τμήματός μας ενδέχεται να αλλάξουν μελλοντικά.

Υποκεφάλαιο 3.1: Υπηρεσίες

Υποκεφάλαιο 3.1.1: Περιγραφή

Εφόσον δεν είχαμε στη διάθεση μας κάποιο API για να μπορούμε να αλληλεπιδρούμε με τους διάφορους servers του τμήματός μας ώστε να χρησιμοποιήσουμε τις υπηρεσίες του, αναγκαστήκαμε να γράψουμε δικές μας κλάσεις οι οποίες να μας δίνουν αυτήν την δυνατότητα.

Μέσω των κλάσεων αυτών μπορούμε να χειριστούμε την είσοδο του χρήστη χρησιμοποιώντας την υποδομή που υπάρχει στους servers (aetos, pithia) καθώς και την λήψη δεδομένων τα οποία θα εμφανίζονται στην οθόνη της συσκευής του χρήστη.

Υποκεφάλαιο 3.1.2: Υλοποίηση μηχανισμού για την πρόσβαση στις υπηρεσίες hydra και pithia

Το interface LoginServiceDelegate

Το interface αυτό περιέχει τις βασικές callback μεθόδους που θα χρειαστούμε στη συνέχεια για να συνδεθούμε με τις υπηρεσίες (hydra, pithia). Ο κώδικας του interface φαίνεται παρακάτω:

```
1. package org.teitheapp.classes;
2.
3. public interface LoginServiceDelegate {
4.     public void loginSuccess(String cookie, String am, String
   surname, String name, int loginMode);
5.     public void loginFailed(int status, int loginMode);
6.     public void netError(String errMsg);
7. }
```

Αρχείο: LoginServiceDelegate.java

Η μέθοδος loginSuccess θα εκτελεστεί όταν υπάρξει επιτυχής είσοδος στο σύστημα. Η μέθοδος αυτή μας παρέχει τα παρακάτω:

- cookie (String): είναι το αλφαριθμητικό το οποίο θα χρησιμοποιούμε στα requests προς τον server. Είναι αυτό που κρατάει ο φυλλομετρητής μας (browser) όταν συνδεόμαστε μέσω αυτού
- am (String): είναι ο αριθμός μητρώου (σε αλφαριθμητική μορφή) του φοιτητή που συνδέθηκε στο σύστημα
- surname (String): είναι το επώνυμο του φοιτητή που συνδέθηκε στο σύστημα
- name (String): είναι το όνομα του φοιτητή που συνδέθηκε στο σύστημα
- loginMode (int): είναι ο αριθμός που δηλώνει σε ποιά υπηρεσία συνδέθηκε ο χρήστης (hydra, pithia) σύμφωνα με τον παρακάτω κώδικα στον οποίο φαίνεται ο ορισμός των σταθερών:

```
8. public final static int LOGIN_MODE_HYDRA = 1;
9. public final static int LOGIN_MODE_PITHIA = 2;
```

Αρχείο: LoginService.java

Η μέθοδος loginFailed θα εκτελεστεί όταν υπάρξει αποτυχία εισόδου στο σύστημα. Η μέθοδος αυτή μας παρέχει τα παρακάτω:

- status (int): είναι ένας ακέραιος που δηλώνει το είδος του σφάλματος, τα οποία ορίζονται ως εξής:

```
1. public final static int RESPONSE_BADUSERPASS = 3;
2. public final static int RESPONSE_SERVICEUNAVAILABLE = 4;
3. public final static int RESPONSE_TIMEOUT = 5;
```

Αρχείο: LoginService.java

Τιμή 3: Λάθος όνομα χρήστη/λάθος κωδικός πρόσβασης
Τιμή 4: Η υπηρεσία δεν είναι διαθέσιμη αυτή τη στιγμή
Τιμή 5: Υπέρβαση του χρονικού ορίου ολοκλήρωσης

- loginMode (int): είναι ο αριθμός που δηλώνει σε ποιά υπηρεσία συνδέθηκε ο χρήστης (hydra, pithia).

Η μέθοδος netError θα εκτελεστεί όταν υπάρξει αποτυχία σύνδεσης με το δίκτυο, π.χ. κακό σήμα στο ασύρματο δίκτυο. Η μέθοδος αυτή μας παρέχει τα παρακάτω:

- errMsg (String): είναι η συμβολοσειρά που μας επιστρέφει η βιβλιοθήκη της java όταν υπάρξει πρόβλημα στη σύνδεση

Η κλάση LoginService

Η κλάση αυτή μας παρέχει τις μεθόδους που χρειαζόμαστε για να συνδεθούμε στο hydra ή στο pithia.

Δομητής / Constructor

```
1. public LoginService(int loginMode, String user, String pass,
   LoginServiceDelegate delegate) {
2.     super();
3.     LOGIN_MODE = loginMode;
4.     this.user = user;
5.     this.pass = pass;
6.     this.delegate = delegate;
7. }
```

Αρχείο: LoginService.java

Ο δομητής δέχεται τέσσερεις παραμέτρους:

- loginMode(int): ο ακέραιος που δηλώνει σε ποια υπηρεσία επιθυμούμε να συνδεθούμε
- user(String): το username του φοιτητή. Π.χ. it123456
- pass(String): ο κωδικός του φοιτητή
- delegate(LoginServiceDelegate): η κλάση η οποία θα υλοποιήσει το interface LoginServiceDelegate

Αφού έχουμε δημιουργήσει ένα instance της κλάσης μπορούμε να κάνουμε login στο σύστημα καλώντας τη μέθοδο login(). Η μέθοδος αυτή δημιουργεί ένα instance του AsyncTask, μιας κλάσης η οποία μας επιτρέπει να τρέχουμε κώδικα στο background. Μόλις ολοκληρωθεί, μπορούμε να κάνουμε τις ενέργειες που θέλουμε.

Στην περίπτωσή μας έχουμε δημιουργήσει μια υποκλάση (subclass) της κλάσης μας (LoginService) η οποία επεκτείνει την AsyncTask και υλοποιεί τις μεθόδους doInBackground(...) και onPostExecute(...)

```
1. protected String[] doInBackground(Void... params) {
2.
3.             String strResponse = null;
4.             String strCookie =  null;
5.
6.             try {
7.                 HttpPost post = null;
8.
9.                 List<NameValuePair> nameValuePairs = new
   ArrayList<NameValuePair>();
10.                   String encoding = "utf-8";
11.
12.
13.                   if (LOGIN_MODE == LOGIN_MODE_HYDRA) {
14.                       strCookie = new DefaultHttpClient().execute(new
   HttpGet(new URI(Constants.URL_HYDRA_LOGIN))).getFirstHeader("Set-
   Cookie").getValue().split(";")[0];
15.
16.                       Trace.i("cookieee", strCookie);
17.
18.                       post = new HttpPost(new
   URI(Constants.URL_HYDRA_LOGIN));
19.
20.                       post.addHeader("Cookie", strCookie);
21.
22.                       nameValuePairs.add(new BasicNameValuePair("am",
   user));
23.                       nameValuePairs.add(new
   BasicNameValuePair("pass", pass));
24.                       nameValuePairs.add(new
   BasicNameValuePair("login", "Login"));
25.                   }
26.
27.                   else if (LOGIN_MODE == LOGIN_MODE_PITHIA) {
28.                       encoding = "windows-1253";
29.
30.                       HttpGet get = new HttpGet(new
   URI(Constants.URL_PITHIA_LOGIN));
31.
32.                       post = new HttpPost(new
   URI(Constants.URL_PITHIA_LOGIN));
33.
34.                       DefaultHttpClient defaultHttpClient = new
   DefaultHttpClient();
35.
36.                       HttpResponse response =
   defaultHttpClient.execute(get);
37.
38.   if
   (Net.readStringFromInputStream(response.getEntity().getContent(),
   encoding).contains("Ανακοίνωση")) {
```

```
39.                             return new String[] {null, "service
   unavailable"};
40.                         }
41.
42.                         strCookie = response.getFirstHeader("Set-
   Cookie").getValue().split(";")[0];
43.                         post.addHeader("Host", "pithia.teithe.gr");
44.                         post.addHeader("Content-Type",
45.                                 "application/x-www-form-urlencoded");
46.                         post.addHeader("Cookie", strCookie);
47.                         nameValuePairs.add(new
   BasicNameValuePair("userName",
48.                                 user));
49.                         nameValuePairs.add(new
   BasicNameValuePair("pwd", pass));
50.
51.                         nameValuePairs.add(new
   BasicNameValuePair("submit1","Είσοδος"));
52.                         nameValuePairs.add(new
   BasicNameValuePair("loginTrue", "login"));
53.
54.                         encoding = "windows-1253";
55.                     }
56.
57.                     HttpParams httpParameters = new BasicHttpParams();
58.
   HttpConnectionParams.setConnectionTimeout(httpParameters,
   Constants.TIMEOUT_CONNECTION);
59.                     HttpConnectionParams.setSoTimeout(httpParameters,
   Constants.TIMEOUT_SOCKET_CONNECTION);
60.
61.                     DefaultHttpClient defaultHttpClient = new
   DefaultHttpClient(httpParameters);
62.                     post.setEntity(new
   UrlEncodedFormEntity(nameValuePairs));
63.
64.                     HttpResponse response =
   defaultHttpClient.execute(post);
65.
66.
67.                     InputStream data =
   response.getEntity().getContent();
68.
69.                     strResponse = Net.readStringFromInputStream(data,
   encoding);
70.
71.                 } catch (SocketTimeoutException e) {
72.                     return new String[]{null, "timeout"};
73.                 } catch (Exception e) {
74.                     return new String[]{e.toString(), "neterror"};
75.                 }
76.                 return new String[] {strCookie, strResponse};
77.             }
78.         }
79.     }
80.
```

Αρχείο: LoginService.java

Στη γραμμή 9 ορίζουμε την μεταβλητή που θα κρατάει τις τιμές (post values) που θα στείλουμε στον server (username, password). Στις γραμμές 10-25 ελέγχουμε αν ο χρήστης θέλει να συνδεθεί με το hydra και κάνουμε τις κατάλληλες ενέργειες ώστε να στείλουμε τα κλειδιά (keys) που είναι απαραίτητα στον server.

Στις γραμμές 27-55 κάνουμε τις απαραίτητες ενέργειες για να στείλουμε τα σωστά κλειδιά, εφόσον θέλει ο χρήστης να συνδεθεί στο pithia.

Στις γραμμές 57-75 παίρνουμε την απάντηση του server ελέγχοντας τυχόν προβλήματα στην σύνδεση (try/catch).

Η μέθοδος αυτή επιστρέφει έναν πίνακα τύπου String που περιλαμβάνει το cookie και το response σε αλφαριθμητική μορφή. Ο πίνακας αυτός περνιέται σαν παράμετρος στην onPostExecute η οποία εκτελείται όταν ολοκληρωθεί η doInBackground. Οι εντολές που υπάρχουν μέσα στην onPostExecute εκτελούνται στο κεντρικό thread (main thread). Αυτός είναι και ο λόγος ύπαρξης της AsyncTask, να εκτελεί δηλαδή τις εντολές που θέλουμε στο thread που πρέπει.

Παρακάτω φαίνεται ο κώδικας της onPostExecute:

```
1. protected void onPostExecute(String[] result) {
2.
3.             String lastIpAddressField = "last_ip";
4.
5.             if (result[1].equals("timeout")) {
6.
7.                 delegate.loginFailed(RESPONSE_TIMEOUT,
   LOGIN_MODE);
8.                 return;
9.             }
10.
11.               if (result[1].endsWith("neterror")) {
12.                   delegate.netError(result[0]);
13.                   return;
14.               }
15.
16.               if (LOGIN_MODE == LOGIN_MODE_HYDRA) {
17.
```

```
18.                     if (result[1].contains("Bad
   username/password")) {
19.
   delegate.loginFailed(RESPONSE_BADUSERPASS, LOGIN_MODE);
20.                     } else {
21.
22.                         String pattern = "<div
   class=\"txt\">([^<]+)<\\/div>";
23.
24.                         // Create a Pattern object
25.                         Pattern r = Pattern.compile(pattern);
26.
27.                         // Now create matcher object.
28.                         Matcher m = r.matcher(result[1]);
29.
30.                         String am, name, surName;
31.
32.                         m.find();
33.                         am = m.group(1).trim();
34.                         m.find();
35.                         String namePart =
   m.group(1).trim().replace(" of ", " ");
36.                         String[] parts = namePart.split("\\W");
37.
38.                         surName = parts[0];
39.                         name = parts[1];
40.
41.                         String cookie = result[0];
42.
43.                         //Insert login data to database
44.                         Date date = new java.util.Date();
45.
46.                         DatabaseManager dbManager = new
   DatabaseManager((Context)delegate);
47.
48.                         String cookieFieldName = "hydra_cookie";
49.                         String studentFieldName =
   "hydra_student";
50.
51.                         //Remove previous login sessions from db
52.
   dbManager.deleteSetting(studentFieldName);
53.                         dbManager.deleteSetting(cookieFieldName);
54.
   dbManager.deleteSetting(lastIpAddressField);
55.
56.                         dbManager.insertSetting(new
   Setting(studentFieldName, am + ";"
57.                                 + surName + ";" + name));
58.                         dbManager.insertSetting(new
   Setting(cookieFieldName, cookie + " "
59.                                 + date.getTime()));
60.                         dbManager.insertSetting(new
   Setting(lastIpAddressField, Net.getLocalIpAddress()));
61.
```

```
62.                         dbManager.close();
63.
64.                         delegate.loginSuccess(cookie, am,
   surName, name, LOGIN_MODE);
65.                     }
66.                 }
67.
68.                 else if (LOGIN_MODE == LOGIN_MODE_PITHIA) {
69.
70.                     if (result[1].equals("service unavailable"))
   {
71.
   delegate.loginFailed(RESPONSE_SERVICEUNAVAILABLE, LOGIN_MODE);
72.                         return;
73.                     }
74.
75.                     else if (result[1].contains("Λάθος όνομα
   χρήστη")
76.                             || result[1].contains("Λάθος κωδικός
   πρόσβασης")) {
77.
78.
   delegate.loginFailed(RESPONSE_BADUSERPASS, LOGIN_MODE);
79.
80.                     } else {
81.
82.                         String am, name, surName;
83.                         Matcher m = null;
84.
85.
86.
87.                         Document doc = Jsoup.parse(result[1]);
88.                         Elements elements =
   doc.getElementsMatchingOwnText("Επώνυμο");
89.
90.
91.
92.                         Trace.i("elements", "a" + elements.get(0)
   + "a");
93.
94.                         m = Pattern.compile("<td
   class=\"tableBold\">AEM: </td>[^<]+<td
   colspan=\"3\">([^<]+)").matcher(result[1]);
95.                         m.find();
96.                         am = m.group(1).trim();
97.
98.                         m = Pattern.compile("<td
   class=\"tableBold\">Επώνυμο:
   </td>[^<]+<td>([^<]+)").matcher(result[1]);
99.                         m.find();
100.                        surName = m.group(1).trim();
101.
102.                        m = Pattern.compile("<td
   class=\"tableBold\">Όνομα:
   </td>[^<]+<td>([^<]+)").matcher(result[1]);
```

```
103.                    m.find();
104.                    name = m.group(1).trim();
105.
106.                    String cookie = result[0];
107.
108.
109.                    //Insert login data to database
110.                    Date date = new java.util.Date();
111.
112.                    DatabaseManager dbManager = new
    DatabaseManager((Context)delegate);
113.
114.                    String cookieFieldName = "pithia_cookie";
115.                    String studentFieldName =
    "pithia_student";
116.
117.                    //Remove previous login sessions from db
118.
    dbManager.deleteSetting(studentFieldName);
119.                    dbManager.deleteSetting(cookieFieldName);
120.
    dbManager.deleteSetting(lastIpAddressField);
121.
122.                    dbManager.insertSetting(new
    Setting(studentFieldName, am + ";"
123.                            + surName + ";" + name));
124.                    dbManager.insertSetting(new
    Setting(cookieFieldName, cookie + " "
125.                            + date.getTime()));
126.                    dbManager.insertSetting(new
    Setting(lastIpAddressField, Net.getLocalIpAddress()));
127.
128.                    dbManager.close();
129.                    dbManager = null;
130.
131.                    delegate.loginSuccess(cookie, am,
    surName, name, LOGIN_MODE);
132.                }
133.
134.
135.            }
136.    }
```

Αρχείο: LoginService.java

Στις γραμμές 5-7 ελέγχουμε αν η doInBackground μας έχει επιστρέψει timeout και αν αληθεύει καλούμε την loginFailed της delegate κλάσης.

Στις γραμμές 11-14 ελέγχουμε αν υπήρξε κάποιο πρόβλημα με την σύνδεση και καλούμε την netError της delegate κλάσης.

Στις γραμμές 16-64 εκτελείται το κομμάτι κώδικα που αναφέρεται στο hydra. Ελέγχουμε αν ο server έχει επιστρέψει μήνυμα σφάλματος, π.χ. λάθος κωδικός και καλούμε την loginFailed της delegate κλάσης. Στην συνέχεια με χρήση των κανονικών παραστάσεων (regular expressions) κάνουμε ανάλυση (parse) της απάντησης του hydra για να πάρουμε τον αριθμό μητρώου και το ονοματεπώνυμο του φοιτητή. Τέλος, αποθηκεύουμε τα παραπάνω στοιχεία μαζί με το cookie και την ημερομηνία σε μορφή unix timestamp στην βάση δεδομένων ώστε να μπορούμε να τα χρησιμοποιήσουμε στη συνέχεα.

Στις γραμμές 68-131 κάνουμε την ίδια διαδικασία για την pithia.

Υποκεφάλαιο 3.2: Ανακοινώσεις Hydra

Υποκεφάλαιο 3.2.1: Περιγραφή

Η λειτουργία αυτού του Activity είναι να κατεβάζει τις ανακοινώσεις που βρίσκονται στο http://hydra.it.teithe.gr/s/ και να τις αποθηκεύει τοπικά σε μία βάση δεδομένων (sqlite). Τα βήματα που ακολουθούνται για την ανάκτηση των ανακοινώσεων συνοψίζονται στα εξής:

1. Ελέγχουμε αν έχει περάσει το χρονικό όριο σύνδεσης (login timeout)
2. Σε περίπτωση που δεν έχει περάσει χρησιμοποιούμε το cookie που είναι αποθηκευμένο στην βάση για να ανακτήσουμε τις ανακοινώσεις. Αν έχει περάσει κάνουμε login στο σύστημα
3. Κατεβάζουμε τις ανακοινώσεις και τις αποθηκεύουμε στην βάση δεδομένων ώστε να είναι διαθέσιμες ακόμα και όταν ο χρήστης της εφαρμογής δεν είναι συνδεδεμένος σε κάποιο δίκτυο

Επίσης έχουμε δημιουργήσει και ένα service το οποίο θα ελέγχει για νέες ανακοινώσεις χωρίς ο χρήστης να είναι αναγκασμένος να ανοίξει την εφαρμογή. Το service αυτό τρέχει στο παρασκήνιο (background) και ελέγχει κάθε 5 λεπτά, 15 λεπτά, 30λεπτά, 1 ώρα ή 5 ώρες αν υπάρχει νέα ανακοίνωση ενημερώνοντας τον χρήστη με σχετική ειδοποίηση (notification).

Στα επόμενα υποκεφάλαια εξηγούμε τις βασικές λειτουργίες του Activity και στην συνέχεια του Service.

Υποκεφάλαιο 3.2.2: Κώδικας & επεξήγηση

Ο παρακάτω κώδικας βρίσκεται στη μέθοδο onCreate και ελέγχει αν έχει ξεπεραστεί το χρονικό όριο σύνδεσης στο hydra:

```
1.          Long time =
    Long.parseLong(dbManager.getSetting("hydra_cookie")
2.                  .getText().split("\\s")[1]);
3.          int minutesElapsed = (int) (TimeUnit.MILLISECONDS
4.                  .toSeconds(new java.util.Date().getTime()) -
    TimeUnit.MILLISECONDS
5.                  .toSeconds(time)) / 60;
6.
7.          Trace.i("time", minutesElapsed + "");
8.
9.
10.           // Re-login if required
11.           if (minutesElapsed > Constants.HYDRA_LOGIN_TIMEOUT ||
    !dbManager.getSetting("last_ip").getText().equals(Net.getLocalI
    pAddress())) {
12.
13.
14.               LoginService ls = new
    LoginService(LoginService.LOGIN_MODE_HYDRA,
15.                       preferences.getString("hydra_login",
    null),
16.                       preferences.getString("hydra_pass",
    null), this);
17.               ls.login();
18.
19.               if (!updateInBackground) {
20.
21.                   dialog = ProgressDialog.show(this, "",
22.
    getResources().getString(R.string.login_loading), true);
23.               }
24.
25.           } else {
26.               announcementsDownloader = new
    DownloadWebPageTask();
27.               announcementsDownloader.execute();
28.
29.               if (!updateInBackground) {
30.                   dialog = ProgressDialog.show(this, "",
31.
    getResources().getString(R.string.reading_data), true);
32.               }
33.
34.               dbManager.close();
35.       }
```

Αρχείο:HydraAnnouncements.java

Η λειτουργία του είναι να ελέγχει αν το χρονικό όριο έχει ξεπεραστεί, σύμφωνα με την σταθερά που έχουμε ορίσει στην κλάση Constants, και αν ισχύει η συνθήκη στην γραμμή 11 τότε δημιουργούμε ένα instance της κλάσης LoginService που περιγράψαμε προηγουμένως και περιμένουμε να κληθεί η ζητούμενη callback μέθοδος.

Αν δεν έχει ξεπεραστεί το χρονικό όριο τότε δημιουργούμε ένα instance τύπου DownloadWebPageTask για να κατεβάσουμε τις ανακοινώσεις.

Παρακάτω θα εξηγήσουμε την λειτουργία της doInBackground η οποία είναι το 90% του Activity:

```
36.          protected Integer doInBackground(Void... params) {
37.
38.              List<NameValuePair> nameValuePairs = new
   ArrayList<NameValuePair>();
39.              nameValuePairs.add(new
   BasicNameValuePair("Cookie", cookie));
40.
41.              String cookie =
   dbManager.getSetting("hydra_cookie").getText()
42.                      .split("\\s")[0];
43.
44.              int count = 0;
45.
46.              try {
47.                  HttpGet get = new HttpGet(new URI(
48.                          Constants.URL_HYDRA_ANNOUNCEMENTS));
49.
50.                  get.addHeader("Cookie", cookie);
51.
52.                  DefaultHttpClient defaultHttpClient = new
   DefaultHttpClient();
53.                  HttpResponse response =
   defaultHttpClient.execute(get);
54.
55.                  String curString;
56.                  StringBuffer data = new StringBuffer();
57.
58.                  InputStream is =
   response.getEntity().getContent();
59.                  InputStreamReader isr = new
   InputStreamReader(is, "utf8");
60.
61.                  Integer totalLength = 0;
62.
```

```
63.                     while( (curString =
   Net.readStringFromInputStream(isr, 512)) != null ) {
64.                         if (isCancelled()) {
65.                             Trace.i("cancel", "cancelled!");
66.                             return null;
67.                         }
68.
69.                         totalLength += 512;
70.
71.                         if (totalLength > 1024 * 50) {
72.                             break;
73.                         }
74.
75.                         Trace.i("512 bytes: ", curString);
76.
77.                         data.append(curString);
78.                     }
79.
80.                     isr.close();
81.
82.
83.                     //String data =
   Net.readStringFromInputStream(response
84.                     //      .getEntity().getContent(), "utf8");
85.
86.                     Document doc = Jsoup.parse(data.toString());
87.
88.                     Elements rows =
   doc.getElementsByClass("data").tagName("tr");
89.
90.                     String announcementTitle, announcementBody,
   announcementCategory, announcementAuthor,
   announcementAttachmentLink, announcementDate;
91.
92.                     announcements = new
   ArrayList<Announcement>();
93.
94.                     int step = 1;
95.                     int order = 0;
96.                     int startingOrder = 0;
97.
98.                     if (dbManager.getNumberOfAnnouncements() > 0)
   {
99.                         order =
   dbManager.getAnnouncementMinimumOrder() - 1;
100.                        startingOrder = order;
101.                        step = -1;
102.                    }
103.
104.                    for (int i = 4; i < rows.size()-1; i++) {
105.
106.                        Element el = rows.get(i);
107.
108.                        announcementBody =
   el.attr("onmouseover");
```

```
109.
110.                       String pattern = "return
   overlib\\(\'(.+?)\',TEXTCOLOR.+";
111.                       announcementBody =
   announcementBody.replaceAll(pattern,
112.                                   "$1");
113.
114.                       announcementTitle =
   announcementBody.replaceAll("<div
   class=\"title\">([^>]+)</div>.*", "$1");
115.                       announcementTitle =
   announcementTitle.replace("\\r", "");
116.                       announcementTitle =
   announcementTitle.replace("&amp;", "&");
117.                       announcementTitle =
   announcementTitle.replace("&quot;", "\"");
118.                       announcementTitle =
   announcementTitle.replace("\\'", "\'");
119.
120.                       announcementBody =
   announcementBody.replaceAll("<div
   class=\"title\">[^>]+</div>(.*)", "$1");
121.                       announcementBody =
   announcementBody.replace("\\r", "");
122.                       announcementBody =
   announcementBody.replace("\\'", "\'");
123.                       announcementBody =
   announcementBody.replace("&quot;", "\"");
124.                       announcementBody =
   announcementBody.replace("&amp;", "&");
125.
126.
127.                       //Replace all links with 'link'
128.                       Pattern p = Pattern.compile("(https?://[^
   <)]+)([ <)])");
129.
130.                       Matcher m = p.matcher(announcementBody);
131.
132.                       while (m.find()) {
133.                           String url = m.group(1);
134.                           announcementBody =
   announcementBody.replace(url, "<a href=\"" + url +
   "\">link</a>");
135.                       }
136.
137.                       Trace.i("data", announcementBody);
138.
139.                       Elements children =
   el.getElementsByTag("td");
140.
141.                       announcementCategory =
   children.get(0).text();
142.                       announcementAuthor =
   children.get(1).text();
```

```
143.                     announcementDate =
   children.get(2).text();
144.
145.                     Trace.i("ann", announcementCategory + " "
146.                             + announcementAuthor + " " +
   announcementTitle);
147.
148.                     // Export attachment link
149.                     announcementAttachmentLink =
   children.get(4).html();
150.
151.                     pattern = ".* <a href=\"([^\"]+)\"
   class=\"veh-link\" target=\"_blank\">[^\\<]+</a>";
152.
153.                     announcementAttachmentLink =
   announcementAttachmentLink
154.                             .replaceAll(pattern, "$1");
155.
156.                     announcementAttachmentLink =
   announcementAttachmentLink.replace("&amp;", "&");
157.
158.                     Announcement newAnnouncement = new
   Announcement(announcementBody,
159.                             announcementCategory,
   announcementAuthor,
160.                             announcementTitle,
   announcementAttachmentLink,
161.                             announcementDate, order);
162.
163.                     order += step;
164.
165.                     announcements.add(newAnnouncement);
166.
167.                 }
168.
169.
170.
171.
172.                 //Trace.i("number",
   dbManager.getNumberOfAnnouncements() +"");
173.
174.                 //Add the required announcements to database
175.                 //diff = announcements.size() -
   (int)dbManager.getNumberOfAnnouncements();
176.
177.                 //dbManager.removeAllAnnouncements();
178.
179.                 ArrayList<Announcement> addedAnnouncements =
   new ArrayList<Announcement>();
180.
181.                 for (int i = 0; i < announcements.size();
   i++) {
182.                     Announcement thisAnnouncement =
   announcements.get(i);
183.
```

```
184.                     if
   (dbManager.announcementExists(thisAnnouncement)) {
185.                         break;
186.                     }
187.
188.                     count++;
189.
190.
   dbManager.insertAnnouncement(thisAnnouncement);
191.                     addedAnnouncements.add(thisAnnouncement);
192.                 }
193.
194.                 //Fix order
195.                 if (step == -1) {
196.
197.                     //startingOrder =
   addedAnnouncements.get(addedAnnouncements.size()-1).getOrder();
198.
199.                     fixOrder(addedAnnouncements);
200.
201.                     for (Announcement ann:
   addedAnnouncements) {
202.                         Trace.i("starting order: ",
   startingOrder + "");
203.                         ann.setOrder(startingOrder--);
204.                         dbManager.updateAnnouncement(ann);
205.                     }
206.                 }
207.
208.                 announcements = null;
209.
210.                  Editor editor = preferences.edit();
211.                  editor.putLong("hydra_last_fetch", new
   Date().getTime());
212.                  editor.commit();
213.
214.                 //System.gc();
215.                 //Trace.i("announcements inserted to db:" ,
   numberToAdd + "");
216.
217.
218.
219.                 // Trace.i("data", tables.size() + "");
220.             } catch (Exception e) {
221.                 // TODO Auto-generated catch block
222.                 e.printStackTrace();
223.             }
224.
225.             // Trace.i("childData", childData.size() + "");
226.
227.             return new Integer(count);
228.         }
```

Αρχείο:HydraAnnouncements.java

Όπως έχουμε εξηγήσει και σε προηγούμενο υποκεφάλαιο, η doInBackground εκτελείται στο παρασκήνιο και μόλις ολοκληρωθεί καλείται η onPostExecute. Στις γραμμές 3-45 κάνουμε τις απαραίτητες ενέργειες για να στείλουμε το κατάλληλο αίτημα (request) στον server ώστε να πάρουμε τα δεδομένα. Στην γραμμή 51 δημιουργούμε ένα instance της κλάσης Jsoup την οποία χρησιμοποιούμε για να αναλύσουμε τον html κώδικα ώστε να πάρουμε το κάθε πεδίο της ανακοίνωσης ξεχωριστά (χωρίς το framework Jsoup αυτό θα ήταν πολύ δυσκολότερο).

Στις επόμενες γραμμές αρχικοποιούμε τις μεταβλητές και ελέγχουμε τις συνθήκες έτσι ώστε οι ανακοινώσεις που αποθηκεύονται στην βάση να είναι με την σωστή σειρά (η πρώτη ανακοίνωση να είναι και η πιο πρόσφατη). Στη συνέχεια μέσα από έναν βρόγχο επανάληψης παίρνουμε μία-μία τις ανακοινώσεις σε html μορφή και ξεχωρίζουμε τα πεδία τους αφαιρώντας οποιουσδήποτε «βρώμικους» html χαρακτήρες υπάρχουν μέσα τους (, &quit;, κ.α.). Αφού πάρουμε έναν ικανοποιητικό αριθμό ανακοινώσεων (δεν τις παίρνουμε όλες για λόγους αποδοτικότητας) ελέγχουμε, αρχίζοντας από την πιο πρόσφατη, αν υπάρχει στην βάση η τρέχουσα ανακοίνωση. Αν υπάρχει, σταματάμε εκεί, αποθηκεύοντας στην βάση τις καινούριες ανακοινώσεις, ενημερώνοντας παράλληλα τον χρήστη για τον αριθμό των νέων ανακοινώσεων.

Ο χρήστης βλέπει τις ανακοινώσεις από την πιο πρόσφατη μέχρι την πιο παλιά που έχει ληφθεί από τον server. Πατώντας τα σχετικά βελάκια της διεπαφής φορτώνεται η αντίστοιχη ανακοίνωση από την βάση στην μνήμη και εμφανίζεται σε ένα WebView. Για λόγους ευκολίας έχουμε μετατρέψει οποιοδήποτε link βρίσκεται μέσα στην ανακοίνωση (που ξεκινάει δηλαδή με http:// και τελειώνει σε .com .gr κ.ο.κ.) σε html σύνδεσμο ώστε, όταν το επιλέγει ο χρήστης, να τον οδηγεί στην αντίστοιχη σελίδα επιλέγοντας τον φυλλομετρητή της επιλογής του.

Στην περίπτωση που υπάρχει συνημμένο στην ανακοίνωση, ο χρήστης μπορεί να το κατεβάσει πατώντας το κουμπί ρυθμίσεων του κινητού του. Στην συνέχεια θα του εμφανιστεί ένα παράθυρο διαλόγου στο οποίο μπορεί να πληκτρολογήσει την διαδρομή που θέλει για την αποθήκευση του συνημμένου.

Παρακάτω φαίνεται ο κώδικας για το onOptionsItemSelected event, για την ανάκτηση, δηλαδή, του συνημμένου:

```
1. public boolean onOptionsItemSelected(MenuItem item) {
2.             switch (item.getItemId()) {
3.                 // For "Title only": Examples of matching an ID
   with one assigned in
4.                 //                    the XML
5.                 case R.id.menu_attachment:
6.                     Intent intent = new Intent();
7.                     intent.setClass(this,
   DownloadAttachment.class);
8.                     intent.putExtra("url",
   selectedAnnouncement.getAttachmentUrl());
9.                     startActivity(intent);
10.                       return true;
11.             }
12.
13.             return false;
14.         }
```

Αρχείο:HydraAnnouncements.java

Η μέθοδος αυτή εκτελείται όταν ο χρήστης πατήσει στο κουμπί ‘Κατέβασε το συνημμένο’ της εφαρμογής. Αυτό που κάνει είναι να δημιουργεί ένα νέο Activity και να περνάει σαν παράμετρο τον σύνδεσμο του συνημμένου. Το Activity αυτό δημιουργεί ένα instance της κλάσης AsyncTask για να κάνει λήψη του συνημμένου στον φάκελο που έχει επιλέξει ο χρήστης.

Υποκεφάλαιο 3.3: Πληροφορίες διδάσκοντα

Υποκεφάλαιο 3.3.1: Περιγραφή

Μέσω αυτής της υπηρεσίας ο χρήστης-φοιτητής έχει τη δυνατότητα να δει πληροφορίες του διδάσκοντα που τον ενδιαφέρει. Όταν ο χρήστης επιλέξει κάποιον καθηγητή από την λίστα, τα στοιχεία του εμφανίζονται σε νέο παράθυρο όπου περιλαμβάνονται τα παρακάτω στοιχεία:

- Επώνυμο
- Όνομα
- Ρόλος
- Τηλέφωνο
- E-Mail

Τα πεδία τηλέφωνο και e-mail είναι clickable, δηλαδή μπορεί να τα επιλέξει ο χρήστης για να κάνει κλήση ή για να στείλει e-mail στον καθηγητή αντίστοιχα. Επιπλέον, ο χρήστης έχει τη δυνατότητα να αναζητήσει καθηγητή με βάση το όνομα ή το επώνυμό του.

Υποκεφάλαιο 3.3.2: Κώδικας & επεξήγηση

Όπως και στις ανακοινώσεις του Hydra έτσι και εδώ αρχικά ελέγχουμε για το αν έχει ξεπεραστεί το χρονικό όριο από την τελευταία φορά που έκανε login ο χρήστης και κάνουμε τις απαραίτητες ενέργειες για να αποκτήσουμε το σωστό session.

Παρακάτω φαίνεται ο κώδικας που εκτελείται στο παρασκήνιο και κάνει parse τα δεδομένα:

```
1. protected Void doInBackground(Void... params) {
2.
3.             Bundle bundle = new Bundle();
4.             String cookie =
   dbManager.getSetting("hydra_cookie").getText().split("\\s")[0];
5.
6.             Trace.i("Cookie", cookie);
7.
8.             try {
9.                 HttpGet get = new HttpGet(new
   URI(Constants.URL_HYDRA_TEACHER_INFO));
10.
11.                get.addHeader("Cookie", cookie);
12.
13.                DefaultHttpClient defaultHttpClient = new
   DefaultHttpClient();
14.                HttpResponse response =
   defaultHttpClient.execute(get);
15.
16.                String data =
   Net.readStringFromInputStream(response.getEntity().getContent()
   , "utf8");
17.
18.                Trace.i("data", data);
19.
20.                Document doc = Jsoup.parse(data);
21.
22.
23.                Elements tableRows =
   doc.getElementsByClass("data");
24.
25.
26.
27.                for (int i = 3; i < tableRows.size(); i++) {
28.                    Element curRow = tableRows.get(i);
29.
30.                    Elements children =
   curRow.getElementsByTag("td");
31.
```

```
32.                         String name, surname, role, phone, email;
33.
34.                         //Trace.i("row", curRow.html());
35.                         surname = children.get(0).text();
36.                         name = children.get(1).text();
37.                         role = children.get(2).text();
38.                         phone = children.get(3).text();
39.                         email = children.get(4).text();
40.
41.                         teachers.add(new Teacher(surname, name,
   role, phone, email));
42.
43.                         Trace.i("teacher: ", "name: " + name + ",
   surname: " + surname + ", role: " + role + ", phone: " + phone
   + ", email: " + email);
44.                     }
45.
46.
47.
48.                 } catch (Exception e) {
49.                     // TODO Auto-generated catch block
50.                     e.printStackTrace();
51.                 }
52.
53.                 dialog.dismiss();
54.
55.                 return null;
56.         }
```

Αρχείο:TeacherInfo.java

Στις γραμμές 7-41 εκτελούμε έναν βρόγχο και παίρνουμε όλους τους καθηγητές αποθηκεύοντάς τους στον πίνακα teachers.

Στην onPostExecute αρχικοποιούμε την λίστα με τον πίνακα που δημιουργήσαμε στην doInBackground:

```
1. final MySimpleArrayAdapter adapter = new
   MySimpleArrayAdapter(TeacherInfo.this);
2. list.setAdapter(adapter);
```

Αρχείο:TeacherInfo.java

Ορίζουμε τους listeners για την αναζήτηση (search) και την επιλογή (click):

```
1. filterText.addTextChangedListener(new TextWatcher() {
2.                 public void onTextChanged(CharSequence s, int
   start, int before, int count) {
3.                     adapter.filter();
4.                     adapter.notifyDataSetChanged();
5.                 }
6.
7.                 // @Override
8.                  public void beforeTextChanged(CharSequence s,
   int start, int count,
9.                    int after) {
10.                   }
11.
12.                   //@Override
13.                   public void afterTextChanged(Editable s) {
14.                   }
15.               });
16.
17.               list.setOnItemClickListener(new
   OnItemClickListener() {
18.
19.                   public void onItemClick(AdapterView<?>
   parent, View view,
20.                       int position, long id) {
21.
22.                       Intent intent = new
   Intent(TeacherInfo.this, ViewTeacher.class);
23.                       intent.putExtra("teacher",
   (Serializable)teachersClone.get(position));
24.                       startActivity(intent);
25.                   }
26.           });
```

Αρχείο:TeacherInfo.java

Στη γραμμή 22 δημιουργούμε ένα νέο Intent το οποίο θα μας εμφανίσει τις πληροφορίες του καθηγητή.

Στη γραμμή 23 περνάμε σαν παράμετρο στο Intent που δημιουργήσαμε το αντικείμενο του πίνακα teachersClone το οποίο έχει επιλέξει ο χρήστης.

Υποκεφάλαιο 3.4: Τα στοιχεία μου

Υποκεφάλαιο 3.4.1: Περιγραφή

Με την υπηρεσία αυτή ο χρήστης μπορεί να δει τα στοιχεία του με τα οποία είναι γραμμένος στο Ίδρυμα (ΑΤΕΙΘ). Οι πληροφορίες λαμβάνονται από το http://pithia.it.teithe.gr/unistudent και περιλαμβάνουν τις παρακάτω πληροφορίες:

- Επώνυμο
- Όνομα
- ΑΕΜ (Αριθμός μητρώου)
- Τμήμα
- Εξάμηνο
- Πρόγραμμα σπουδών
- Στοιχεία εγγραφής (Ακαδ. Έτος, Περίοδος, εξάμηνο)

Υποκεφάλαιο 3.4.2: Κώδικας & επεξήγηση

Παρακάτω φαίνεται ο κώδικας που εκτελείται στο παρασκήνιο για τη λήψη των δεδομένων από το pithia:

```
1. protected Bundle doInBackground(Void... params) {
2.
3.             Bundle bundle = new Bundle();
4.             String cookie =
   dbManager.getSetting("pithia_cookie").getText().split("\\s")[0]
   ;
5.
6.
7.             try {
8.                 HttpGet get = new HttpGet(new
   URI(Constants.URL_PITHIA_MYINFO));
9.
10.                    get.addHeader("Cookie", "login=True; " +
   cookie);
11.
12.                    DefaultHttpClient defaultHttpClient = new
   DefaultHttpClient();
13.                    HttpResponse response =
   defaultHttpClient.execute(get);
14.
15.                    String data =
   Net.readStringFromInputStream(response.getEntity().getContent()
   , "windows-1253");
16.
17.
18.                    Document doc = Jsoup.parse(data);
19.
20.                    Elements tableRows =
   doc.getElementsByClass("tableBold");
21.                    Elements tableRowsRegistration =
   doc.getElementsByAttributeValue("width",
   "80%").get(0).getElementsByClass("tablecell");
22.
23.                    String surname, name, aem, department,
   semester, program, registration_info;
24.
25.                    surname =
   tableRows.get(10).siblingElements().get(0).text();
26.                    name =
   tableRows.get(11).siblingElements().get(0).text();
27.                    aem =
   tableRows.get(12).siblingElements().get(0).text();
28.                    department =
   tableRows.get(13).siblingElements().get(0).text();
29.                    semester =
   tableRows.get(14).siblingElements().get(0).text();
```

```
30.                     program =
   tableRows.get(15).siblingElements().get(0).text();
31.
32.
33.                     Trace.i("dsa",
   tableRowsRegistration.get(1).text());
34.
35.                     //registration_info = "";
36.
37.                     registration_info = "Ακάδ. έτος: " +
   tableRowsRegistration.get(0).text() + ", Περίοδος: " +
   tableRowsRegistration.get(1).text() + ", Εξάμηνο: " +
   tableRowsRegistration.get(2).text() ;
38.
39.                     bundle.putString("surname", surname);
40.                     bundle.putString("name", name);
41.                     bundle.putString("aem", aem);
42.                     bundle.putString("department", department);
43.                     bundle.putString("semester", semester);
44.                     bundle.putString("program", program);
45.                     bundle.putString("registration_info",
   registration_info);
46.
47.
48.
49.
50.                 } catch (Exception e) {
51.                     // TODO Auto-generated catch block
52.                     e.printStackTrace();
53.                 }
54.
55.                 dialog.dismiss();
56.                 return bundle;
57.
58.
59.         }
```

Αρχείο:MyInfo.java

Στην γραμμή 18 δημιουργούμε ένα instance της κλάσης Jsoup με παράμετρο τα δεδομένα που πήραμε από το pithia.

Στην γραμμή 20 παίρνουμε τις γραμμές που περιέχουν τα στοιχεία του φοιτητή και βρίσκονται στην ιεραρχία του html εγγράφου έχοντας την κλάση “TableBold”.

Στην γραμμή 21 παίρνουμε τις γραμμές που περιέχουν τα στοιχεία εγγραφής του φοιτητή και βρίσκονται στην ιεραρχία του html εγγράφου έχοντας width: 80% και κλάση tablecell.

Στις γραμμές 25-37 δημιουργούμε τις μεταβλητές παίρνοντας τα στοιχεία που μας ενδιαφέρουν.

Στις γραμμές 39-45 δημιουργούμε ένα bundle με τις μεταβλητές αυτές για να το επιστρέψουμε στην onPostExecute η οποία θα εμφανίσει τις πληροφορίες με τα στοιχεία του φοιτητή στην οθόνη.

Παρακάτω φαίνεται η onPostExecute που θα εκτελεστεί στο main thread για να εμφανίσει τα στοιχεία στην οθόνη:

```
1. protected void onPostExecute(Bundle bundle) {
2.             setContentView(R.layout.my_info);
3.             TextView tvSurname =
   (TextView)findViewById(R.id.my_info_surname);
4.             TextView tvName =
   (TextView)findViewById(R.id.my_info_name);
5.             TextView tvAem =
   (TextView)findViewById(R.id.my_info_aem);
6.             TextView tvDepartment =
   (TextView)findViewById(R.id.my_info_department);
7.             TextView tvSemester =
   (TextView)findViewById(R.id.my_info_semester);
8.             TextView tvProgram =
   (TextView)findViewById(R.id.my_info_program);
9.             TextView tvRegistrationInfo =
   (TextView)findViewById(R.id.my_info_resistration);
10.
11.                tvSurname.setText(bundle.getString("surname"));
12.                tvName.setText(bundle.getString("name"));
13.                tvAem.setText(bundle.getString("aem"));
14.
   tvDepartment.setText(bundle.getString("department"));
15.                tvSemester.setText(bundle.getString("semester"));
16.                tvProgram.setText(bundle.getString("program"));
17.
   tvRegistrationInfo.setText(bundle.getString("registration_info"
   ));
18.         }}
```

Αρχείο:MyInfo.java

Υποκεφάλαιο 3.5: Η δήλωσή μου

Υποκεφάλαιο 3.5.1: Περιγραφή

Με την υπηρεσία αυτή ο χρήστης μπορεί να δει τα μαθήματα που έχει δηλώσει το τρέχον εξάμηνο. Τα μαθήματα εμφανίζονται σε μορφή λίστας.

Υποκεφάλαιο 3.5.2: Κώδικας & επεξήγηση

Ο κώδικας της doInBackground φαίνεται παρακάτω:

```
1.     protected ArrayList<String> doInBackground(Void... params)
   {
2.
3.             ArrayList<String> arrCourses = new
   ArrayList<String>();
4.             String cookie =
   dbManager.getSetting("pithia_cookie").getText()
5.                     .split("\\s")[0];
6.
7.             try {
8.                 HttpGet get = new HttpGet(new URI(
9.                         Constants.URL_PITHIA_MY_DECLARATION));
10.
11.                  get.addHeader("Cookie", "login=True; " +
   cookie);
12.
13.                  DefaultHttpClient defaultHttpClient = new
   DefaultHttpClient();
14.                  HttpResponse response =
   defaultHttpClient.execute(get);
15.
16.                  String data =
   Net.readStringFromInputStream(response
17.                          .getEntity().getContent(), "windows-
   1253");
18.
19.                  Document doc = Jsoup.parse(data);
20.
21.                  Elements courses =
   doc.getElementsByAttributeValueStarting(
22.                          "onmouseover", "underline");
```

```
23.
24.                 for (int i = 0; i < courses.size(); i++) {
25.                     arrCourses.add(courses.get(i).text());
26.                 }
27.
28.             } catch (Exception e) {
29.                 // TODO Auto-generated catch block
30.                 e.printStackTrace();
31.             }
32.
33.             dialog.dismiss();
34.             return arrCourses;
35.
36.     }
```

Αρχείο: MyCourseDeclaration.java

Στη γραμμή 18 δημιουργούμε ένα instance της κλάσης Jsoup με παράμετρο τα δεδομένα που πήραμε από το pithia.

Στη γραμμή 39 παίρνουμε όλα τα στοιχεία (elements) που έχουν την τιμή underline στο χαρακτηριστικό (attribute) onmouseover.

Παρακάτω φαίνεται η μέθοδος onPostExecute που θα εκτελεστεί στο main thread για να εμφανίσει τα στοιχεία στην οθόνη:

```
1. protected void onPostExecute(ArrayList<String> arrCourses) {
2.             setContentView(R.layout.my_declaration);
3.             ListView listView = (ListView)
   findViewById(R.id.my_declaration_listview);
4.
5.             Trace.i("rows", arrCourses.size() + "");
6.
7. ArrayAdapter<String> adapter = new
   ArrayAdapter<String>(MyCourseDeclaration.this,
   android.R.layout.simple_list_item_1, android.R.id.text1,
   arrCourses.toArray(new String[arrCourses.size()]));
8.             listView.setAdapter(adapter);
9.
10.         }
```

Αρχείο: MyCourseDeclaration.java

Στη γραμμή 7 δημιουργούμε ένα instance του ArrayAdapter με την λίστα arrCourses που επιστράφηκε από την doInBackground.

Υποκεφάλαιο 3.6 Βαθμολογίες και αριθμός μαθημάτων

Υποκεφάλαιο 3.6.1: Περιγραφή

Οι υπηρεσίες αυτές χρησιμοποιούν την ίδια ακριβώς λογική με τις παραπάνω κι έτσι δεν χρειάζεται η περαιτέρω ανάλυση τους. Οι υπηρεσίες αυτές είναι:

- Οι βαθμολογίες μου
- Αριθμός μαθημάτων για πτυχίο

Στις βαθμολογίες εμφανίζεται στον χρήστη μία λίστα με το σύνολο των βαθμολογιών του στα μαθήματα κάθε εξαμήνου ενώ στον αριθμό μαθημάτων εμφανίζεται ο αριθμός των εναπομεινάντων μαθημάτων για την απόκτηση του πτυχίου.

Οι διαδικασίες που εκτελούνται σε αυτές τις υπηρεσίες είναι οι εξής:

1. Έλεγχος χρονικού ορίου σύνδεσης στις υπηρεσίες hydra/pithia
2. Επανασύνδεση στο σύστημα σε περίπτωση που ο χρόνος αυτός έχει ξεπεραστεί
3. Λήψη δεδομένων με την doInBackground της AsyncTask
4. Εμφάνιση των δεδομένων στην οθόνη της συσκευής με τον κατάλληλο τρόπο με την χρήση της onPostExecute της AsyncTask

Για την ανάγκη των υπηρεσιών Πληρότητα γραμμής 52 και Συνομιλία φοιτητών επιλέξαμε να δημιουργήσουμε ένα API γραμμένο σε PHP. Το API αυτό μας δίνει πρόσβαση στην βάση ώστε να εισάγουμε και να εξάγουμε δεδομένα από αυτήν.

Υποκεφάλαιο 3.7: Επιπρόσθετες υπηρεσίες

Υποκεφάλαιο 3.7.1: Περιγραφή

Οι υπηρεσίες που αναλύονται σε αυτό το υποκεφάλαιο αποτελούν υλοποίηση ιδεών που μας προέκυψαν για την εξυπηρέτηση επιπρόσθετων αναγκών των φοιτητών. Πρόκειται για δύο λειτουργίες που δεν περιλαμβάνονται στις διαθέσιμες υπηρεσίες του τμήματος και, πιο συγκεκριμένα, είναι οι εξής:

- Πληρότητα Γραμμής 52
- Συνομιλία φοιτητών

Η πληρότητα της γραμμής 52 αφορά της γραμμή της αστικής συγκοινωνίας του ΟΑΣΘ που εξυπηρετεί τους φοιτητές για την πρόσβαση στο Ίδρυμα. Ενημερώνει και δίνει την επιλογή για τον ορισμό της πληρότητας του αστικού λεωφορείου ώστε να πληροφορούνται σχετικά οι φοιτητές.

Η συνομιλία των φοιτητών είναι μια υπηρεσία ανταλλαγής μηνυμάτων (chatting) και εξυπηρετεί την επικοινωνία και, γενικότερα, την αλληλεπίδραση μεταξύ των φοιτητών.

Υποκεφάλαιο 3.7.2: Πληρότητα γραμμής 52

Για την πληρότητα γραμμής 52 ο τρόπος χρήσης του API περιγράφεται στον παρακάτω πίνακα:

Μέθοδος	**Παράμετροι**	**Λειτουργία**
κενό	κενό	επιστρέφει όλον τον πίνακα με τις καταχωρήσεις
add	**starting_point** 1: από ΝΣΣ 2: από ΤΕΙ **progress** το ποσοστό πληρότητας	δημιουργεί μία νέα καταχώριση πληρότητας
update	**id** το id της εγγραφής **starting_point** 1: από ΝΣΣ 2: από ΤΕΙ **progress** το ποσοστό πληρότητας	ενημερώνει μία εγγραφή στον πίνακα
delete	**id** το id της εγγραφής	διαγράφει μία εγγραφή από τον πίνακα

Πίνακας 1: Υπηρεσία πληρότητας

script url: http://aetos.it.teithe.gr/~vpanag/teitheapp/json.php?action=bus_line

Το sql ερώτημα για την δημιουργία του πίνακα φαίνεται παρακάτω:

```
1. CREATE TABLE `bus_line` (
2.  `id` int(11) NOT NULL AUTO_INCREMENT,
3.  `update_time` datetime NOT NULL,
4.  `starting_point` int(11) NOT NULL,
5.  `progress` int(11) NOT NULL,
6.  PRIMARY KEY (`id`)
7. )
```

Ο php κώδικας που αναφέρεται στην πληρότητα φαίνεται παρακάτω:

```
1.     if ($action == "bus_line") {
2.         if ($mode == "") {
3.             $sql = "SELECT `id`,UNIX_TIMESTAMP(`update_time`)
   as `update_time`, `starting_point`, `progress`
4.                     FROM `bus_line` ORDER BY `update_time` DESC
   limit 5";
5.             $result = mysql_query($sql, $link);
6.
7.             $bus_lines = array();
8.
9.
10.
11.                 while ($row = mysql_fetch_assoc($result)) {
12.                     $json_row = array();
13.
14.                     $json_row['id'] = (int)$row['id'];
15.                     $json_row['starting_point'] =
   (int)$row['starting_point'];
16.                     $json_row['update_time'] =
   (int)$row['update_time'];
17.                     $json_row['progress'] =
   (int)$row['progress'];
18.
19.                     array_push($bus_lines, $json_row);
20.                 }
21.                 echo json_encode($bus_lines);
22.             }
23.             else if ($mode == "add") {
24.                 $starting_point = $_GET['starting_point'];
25.                 $progress = $_GET['progress'];
26.
27.                 $sql = "INSERT INTO `bus_line` (`update_time`,
   `starting_point`, `progress`) VALUES (now(), $starting_point,
   $progress)";
28.                 Database::execute($sql, $link);
29.
30.                 if (mysql_affected_rows($link) > 0) {
31.                     $json['status'] = "success";
32.                     $json['id'] = mysql_insert_id();
```

```
33.                     echo json_encode($json);
34.                 } else {
35.                     $json['status'] = "failed";
36.                     echo json_encode($json);
37.                 }
38.
39.             }
40.
41.             else if ($mode == "update") {
42.                 $id = $_GET['id'];
43.                 $starting_point = $_GET['starting_point'];
44.                 $progress = $_GET['progress'];
45.
46.                 $sql = "UPDATE `bus_line` SET `starting_point` =
   $starting_point,
47.                                                 `update_time` =
   now(),
48.                                                 `progress` =
   $progress WHERE `id` = $id";
49.                 Database::execute($sql, $link);
50.
51.                 if (mysql_affected_rows($link) > 0) {
52.                     $json['status'] = "success";
53.                     $json['id'] = (int)$id;
54.                     echo json_encode($json);
55.                 } else {
56.                     $json['status'] = "failed";
57.                     echo json_encode($json);
58.                 }
59.
60.             }
61.
62.             else if ($mode == "delete") {
63.                 $id = $_GET['id'];
64.                 $sql = sprintf("DELETE FROM `bus_line` WHERE `id`
   = %s", $id);
65.
66.                 Database::execute($sql, $link);
67.                 if (mysql_affected_rows($link) > 0) {
68.                     $json['status'] = "success";
69.
70.                     echo json_encode($json);
71.                 } else {
72.                     $json['status'] = "failed";
73.                     echo json_encode($json);
74.                 }
75.             }
76.     }
```

Υποκεφάλαιο 3.7.3: Συνομιλία φοιτητών

Για την συνομιλία φοιτητών ο τρόπος χρήσης του API περιγράφεται στον παρακάτω πίνακα:

Μέθοδος	**Παράμετροι**	**Λειτουργία**
κενό	κενό	επιστρέφει όλον τον πίνακα με τις καταχωρήσεις
add	**student_name** όνομα φοιτητή **text** κείμενο	δημιουργεί μία νέα καταχώρηση συνομιλίας

Πίνακας 2: Υπηρεσία συνομιλίας

script url: http://aetos.it.teithe.gr/~vpanag/teitheapp/json.php?action=chat

Το sql ερώτημα για την δημιουργία του πίνακα φαίνεται παρακάτω

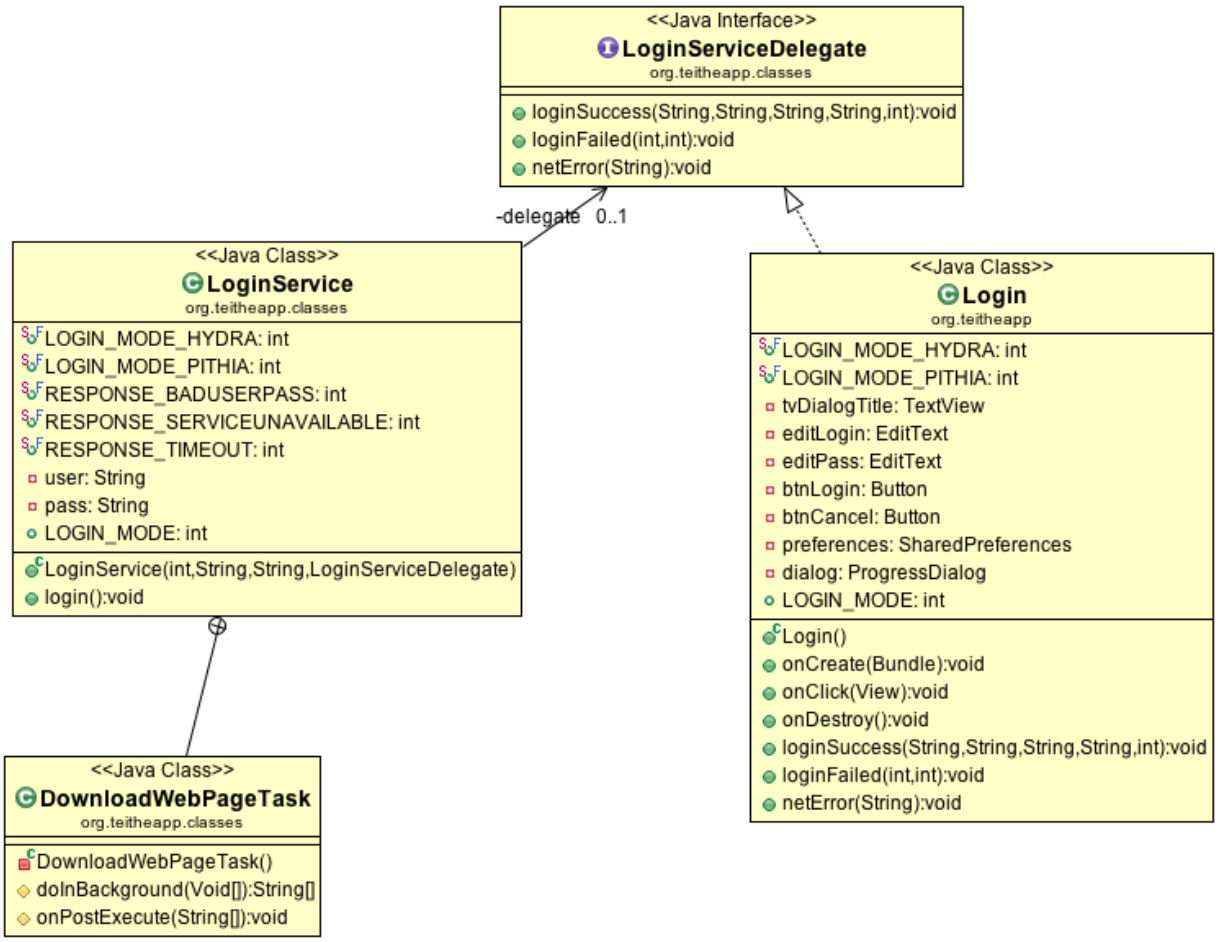
```
1. CREATE TABLE `chat` (
2.  `id` int(11) NOT NULL AUTO_INCREMENT,
3.  `student_name` varchar(55) COLLATE utf8_unicode_ci NOT NULL,
4.  `text` text COLLATE utf8_unicode_ci NOT NULL,
5.  `update_time` datetime NOT NULL,
6.  PRIMARY KEY (`id`,`student_name`)
7. )
```

Ο php κώδικας που αναφέρεται στην πληρότητα φαίνεται παρακάτω:

```
1. else if ($action == "chat") {
2.         if ($mode == "") {
3.             $sql = "SELECT `id`, UNIX_TIMESTAMP(`update_time`)
   as `update_time`, `student_name`, `text`
4.                     FROM `chat` ORDER BY `update_time` DESC";
5.             $result = mysql_query($sql, $link);
6.
7.             $chat_rows = array();
8.
9.             while ($row = mysql_fetch_assoc($result)) {
```

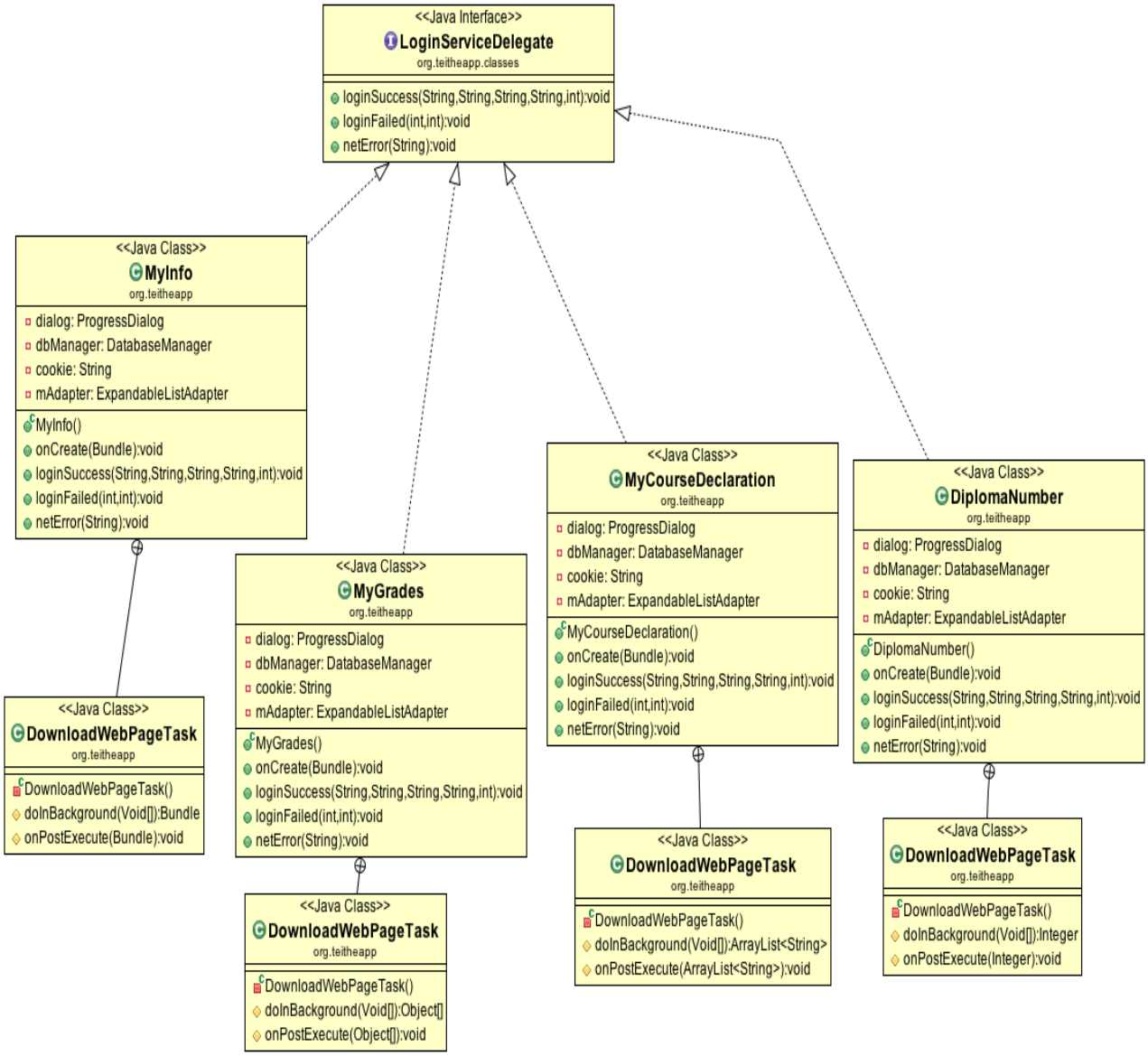
```
10.                     $json_row = array();
11.
12.                     $json_row['id'] = (int)$row['id'];
13.                     $json_row['student_name'] =
   $row['student_name'];
14.                     $json_row['text'] = $row['text'];
15.                     $json_row['update_time'] =
   (int)$row['update_time'];
16.
17.                     array_push($chat_rows, $json_row);
18.                 }
19.                 echo json_encode($chat_rows);
20.             }
21.             else if ($mode == "add") {
22.                 $student_name = $_GET['student_name'];
23.                 $text = $_GET['text'];
24.
25.
26.                 $sql = "INSERT INTO `chat` (`student_name`,
   `text`, `update_time`) VALUES ('$student_name', '$text',
   now())";
27.                 Database::execute($sql, $link);
28.
29.                 if (mysql_affected_rows($link) > 0) {
30.                     $json['status'] = "success";
31.                     echo json_encode($json);
32.                 } else {
33.                     $json['status'] = "failed";
34.                     echo json_encode($json);
35.                 }
36.
37.         }
```

Υποκεφάλαιο 3.8: Διαγράμματα κλάσεων



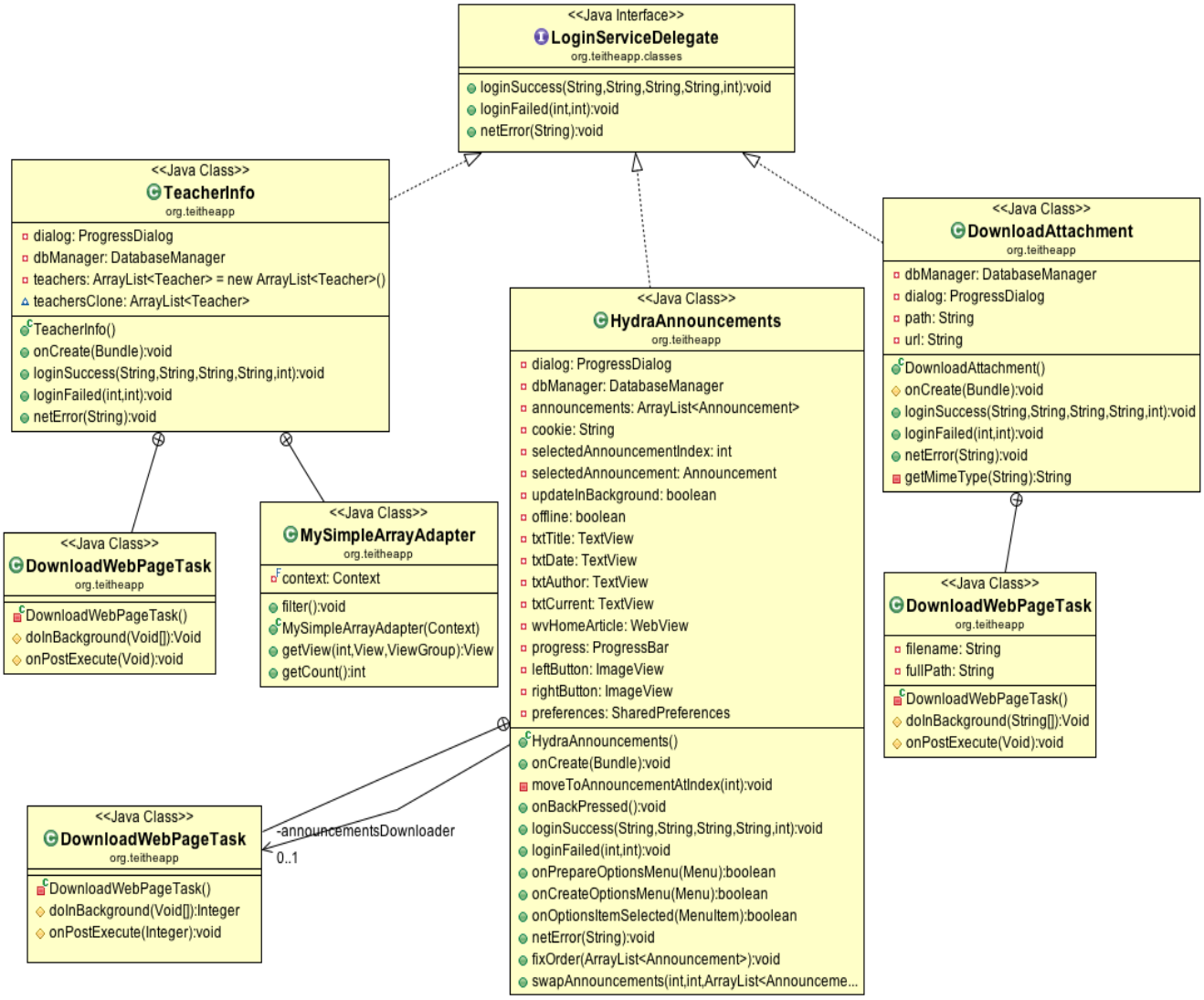
Σχέδιο 21: ΔΚ μηχανισμού εισόδου στα συστήματα Hydra, Pithia

Οι κλάσεις LoginService και Login υλοποιούν το LoginServiceDelegate.



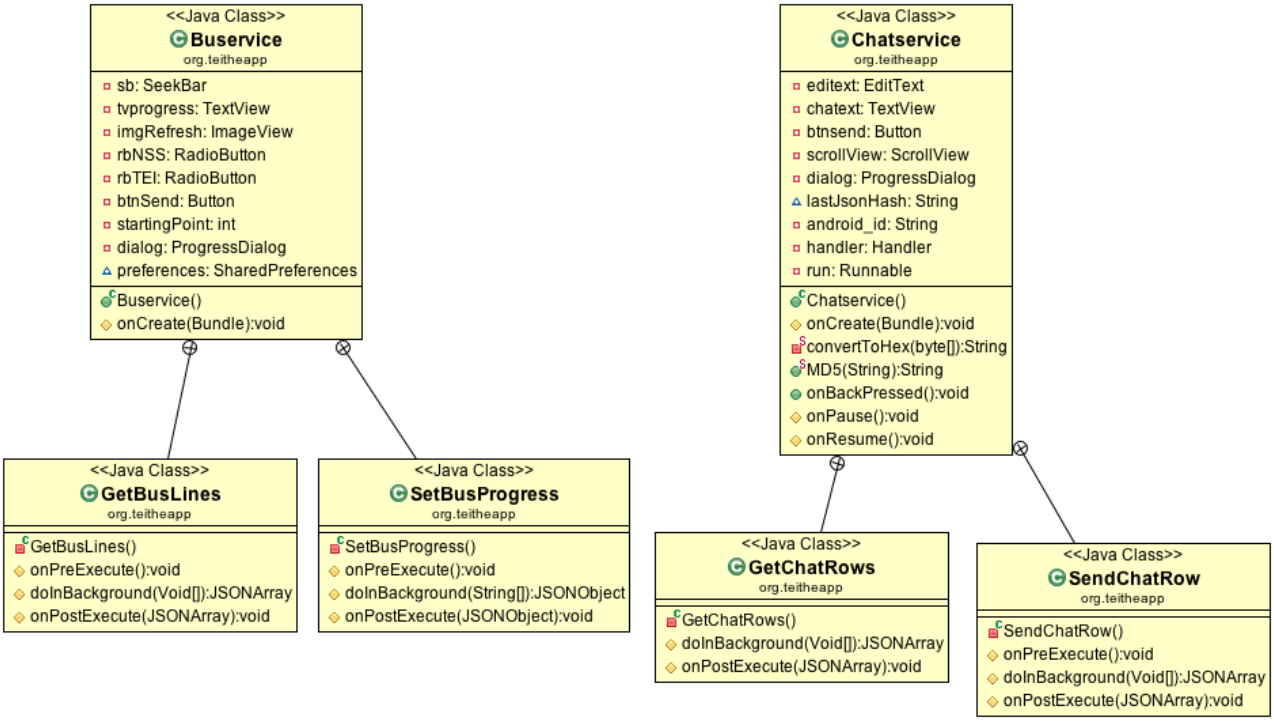
Σχέδιο 22: ΔΚ υπηρεσιών Pithia

Όλες οι υπηρεσίες της pithia που υλοποιούν το LoginServiceDelegate έχουν την DownloadWebPageTask ως υποκλάση η οποία αναλαμβάνει την λήψη των δεδομένων.



Σχέδιο 23: ΔΚ υπηρεσιών Hydra

Η κλάση DownloadAttachment χρησιμοποιείται για την λήψη του συνημμένου.



Σχέδιο 24: ΔΚ επιπρόσθετων υπηρεσιών

Επιπρόσθετες υπηρεσίες: οι υποκλάσεις της Buservice χρησιμοποιούνται για την λήψη και για τον ορισμό της πληρότητας όπως γίνεται αντίστοιχα για την Chatservice όπου εκεί χρησιμοποιούνται για την λήψη και τον ορισμό των μηνυμάτων.

Επίλογος

Στο κεφάλαιο αυτό αναφερθήκαμε στα σημαντικότερα σημεία της εφαρμογής όσον αφορά τον κώδικα και εξηγήσαμε την λειτουργία του χρησιμοποιώντας αρκετά παραδείγματα και αναφέροντας τα ονόματα των αρχείων για να μπορέσει ο αναγνώστης να τα βρει εύκολα. Επίσης, ενσωματώσαμε διαγράμματα κλάσεων για την κατανόηση κάποιων συσχετισμών.

Όπως αναφέραμε και στην αρχή του κεφαλαίου είναι αδύνατο να συμπεριλάβουμε όλον τον κώδικα της εφαρμογής στο έγγραφο. Για οποιονδήποτε ενδιαφέρεται να μελετήσει τον κώδικα και πιθανότατα να διορθώσει τυχόν λάθη ή ακόμη και να συμβάλει στην επέκτασή του μπορεί να επισκευθεί την σελίδα: https://github.com/billp/teitheapp.

Το github είναι μία υπηρεσία που φιλοξενεί έργα ανοιχτού κώδικα και η οποία μας βοήθησε πολύ στον συγχρονισμό μας για την ολοκλήρωση της εφαρμογής.

Συμπεράσματα

Αυτή η πτυχιακή εργασία περιλαμβάνει την δουλειά ενός εξαμήνου και είναι υλοποιημένη με τα δικά μας κριτήρια και δυνατότητες. Ως εκ τούτου, δεν είναι παρά μέρος αυτού που θα μπορούσε να πραγματοποιηθεί για τις ανάγκες των φοιτητών και των ενδιαφερομένων του τμήματος μας.

Έτσι, συμπερασματικά, θα αναφέρουμε κάποια σημεία της εφαρμογής που μπορούν να επεκταθούν, ποσοτικά και ποιοτικά. Αυτά απευθύνονται σε όποιους ενδιαφέρονται να ασχοληθούν περαιτέρω με το αντικείμενο που μας απασχόλησε.

Όσον αφορά το εύρος της εφαρμογής, εμείς καλύψαμε το τμήμα Πληροφορικής. Ένα επόμενο θεμιτό βήμα είναι να γίνει μία προσπάθεια συμπερίληψης όλων των τμημάτων του Ιδρύματος ώστε κάτω από μία καθολική δομή να υπάρξει το σύνολο των αναγκών των φοιτητών του. Μία ποσοτική, δηλαδή, επέκταση. Αυτή μπορεί να περιλαμβάνει τις πληροφορίες που παρέχει το κάθε τμήμα ξεχωριστά καθώς και τις υπηρεσίες που παρέχονται καθολικά στους φοιτητές (σύστημα Blackboard, σύστημα δηλώσεων μαθημάτων, δομές του Ιδρύματος κ.α.).

Επίσης, όσον αφορά την δομή των περιεχομένων, μπορεί να στηρίζεται στον υλοποίηση μας, με μία δυναμική προσκόμιση πληροφοριών για κάθε τμήμα. Δηλαδή, η δομή που αναπτύξαμε να είναι το πλαίσιο για κάθε επιμέρους τμήμα ενώ η επιλογή του κάθε τμήματος από τον χρήστη να προηγείται. Πιο συγκεκριμένα, να υπάρχουν γενικές πληροφορίες και υπηρεσίες που αφορούν το Ίδρυμα συνολικά και έπειτα από επιλογή του χρήστη να μεταβαίνει σε μία δομή παρόμοια της δικής μας η οποία θα αφορά το κάθε τμήμα ειδικότερα.

Βιβλιογραφία

Ed Burnette (2010), Hello, Android: Introducing Google's Mobile Development Platform, United States of America, Pragmatic Programmers, LLC

Onur Cinar (2012), Android Apps with Eclipse, Apress

Lauren Darcey (2011), Sams Teach Yourself Android Application Development in 24 Hours (2nd Edition), Sams Publishing

Παραρτήματα

[1] Mobile Milestone: The Number of Smartphones in Use Passed 1 Billion in Q3, Says Strategy Analytics - http://techcrunch.com/2012/10/16/mobile-milestone-the-number-of-smartphones-in-use-passed-1-billion-in-q3-says-strategy-analytics/

[2] Total Count of Smartphone Apps to Surpass the Big Million soon - http://www.wizardjournal.com/tech-news/smartphone-apps-pass-1-million.html

[3] Global Smartphone Apps Market to Reach US$101.2 Billion by the year 2017 - http://www.electronics.ca/presscenter/articles/1644/1/Global-Smartphone-Apps-Market-to-Reach-US1012-Billion-by-the-year-2017/Page1.html

[4] Git with Eclipse (EGit) - Tutorial - http://www.vogella.com/articles/EGit/article.html

[5] Licensing of Distributions - http://www.apache.org/licenses/

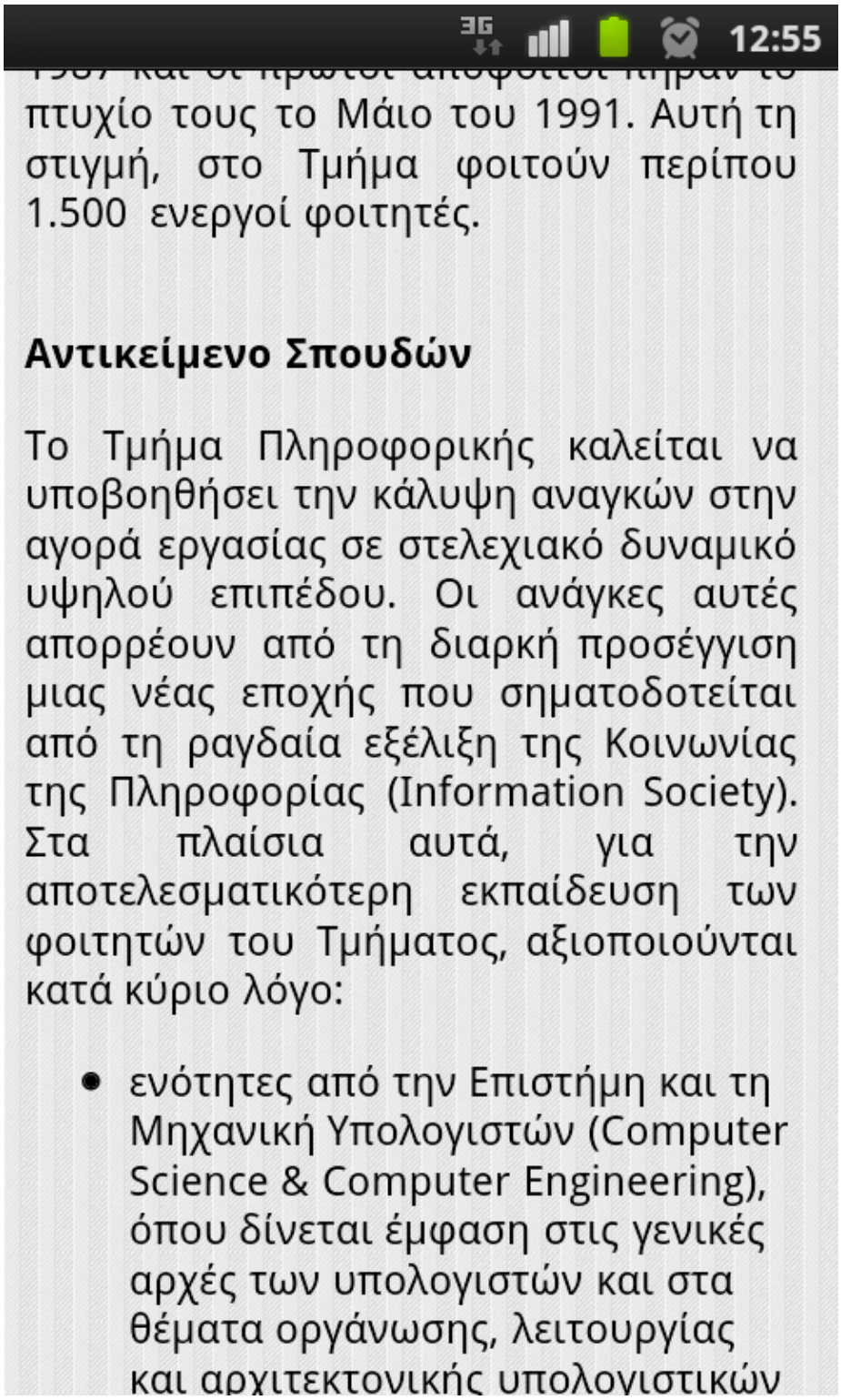
Οδηγός Χρήσης Λογισμικού

Αρχική Οθόνη

Σχέδιο 25: Αρχική οθόνη εφαρμογής

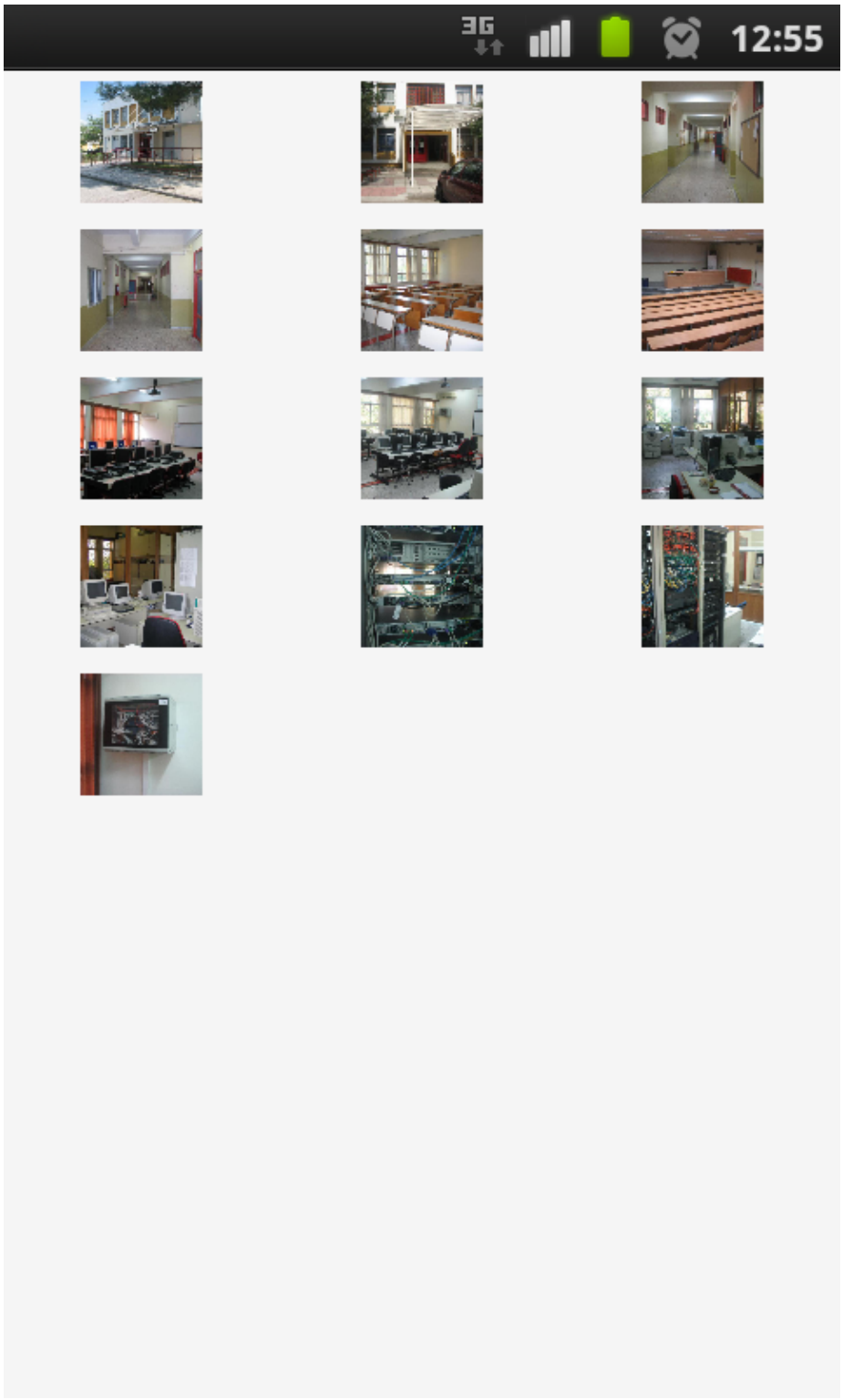
Σχέδιο 26: Καρτέλα Πληροφοριών

Στις **Πληροφορίες** μπορείτε να βρείτε όλα όσα αφορούν το τμήμα Πληροφορικής, από πληροφορίες μαθημάτων/οργάνωσης μέχρι ιστοσελίδες φοιτητών/καθηγητών.



Σχέδιο 27: Οθόνη εμφάνισης πληροφοριών

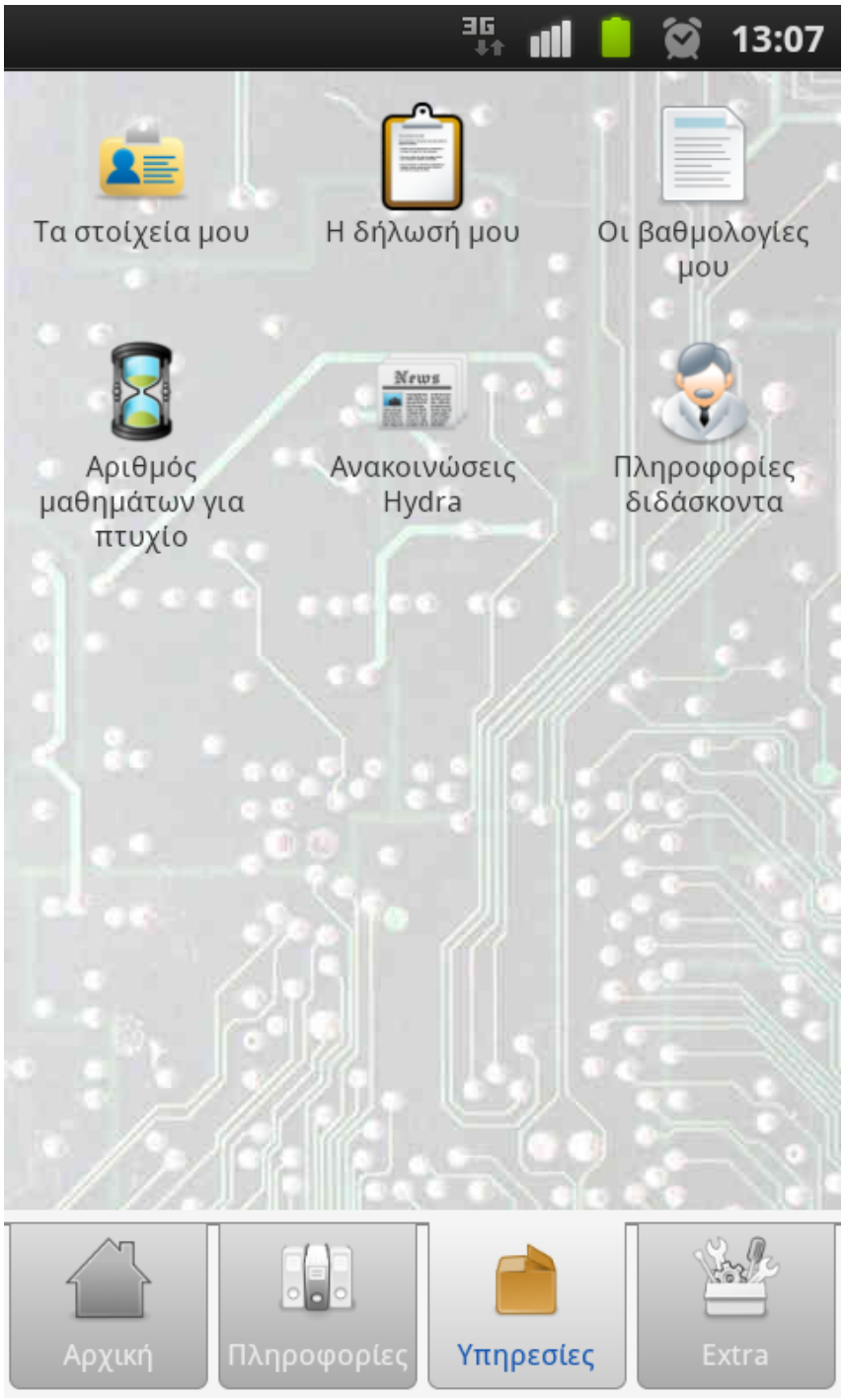
Η οθόνη **‘Το Τμήμα’** στην καρτέλα Πληροφορίες



Σχέδιο 28: Οθόνη φωτογραφιών

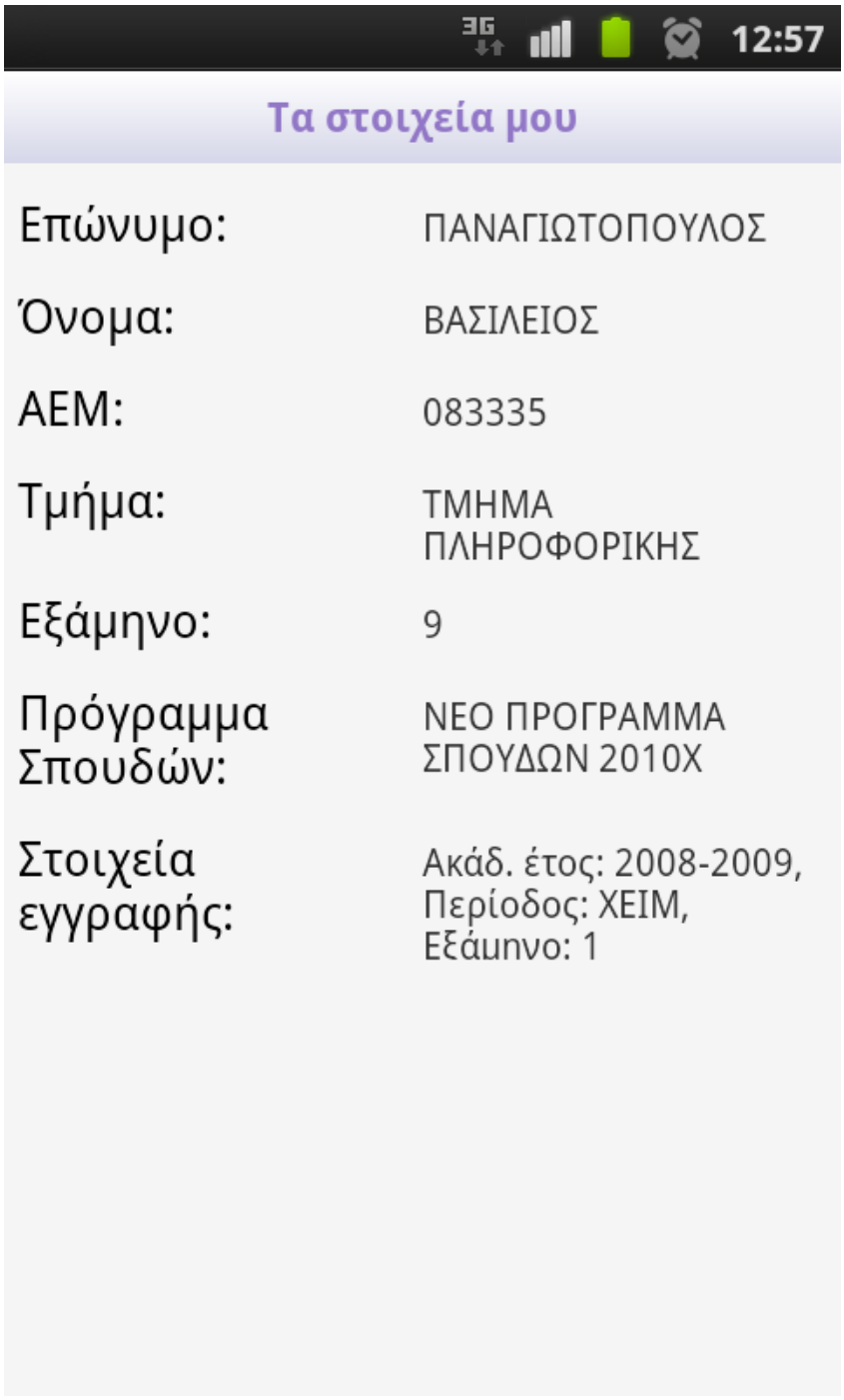
Η οθόνη **‘Φωτογραφίες’** στην καρτέλα Πληροφορίες

Καρτέλα Υπηρεσιών



Σχέδιο 29: Καρτέλα υπηρεσιών

Στις υπηρεσίες μπορούν οι φοιτητές να ενημερωθούν για νέες ανακοινώσεις, να αναζητήσουν πληροφορίες για κάποιον διδάσκοντα, να δουν τις βαθμολογίες τους, κ.α. Η χρήση των υπηρεσιών αυτών απαιτεί σύνδεση στις υπηρεσίες hydra/pithia κάτι το οποίο γίνεται πατώντας το κουμπί των ρυθμίσεων της συσκευής (βρίσκεται κάτω αριστερά ή κάτω δεξιά ανάλογα με το μοντέλο και τον κατασκευαστή).

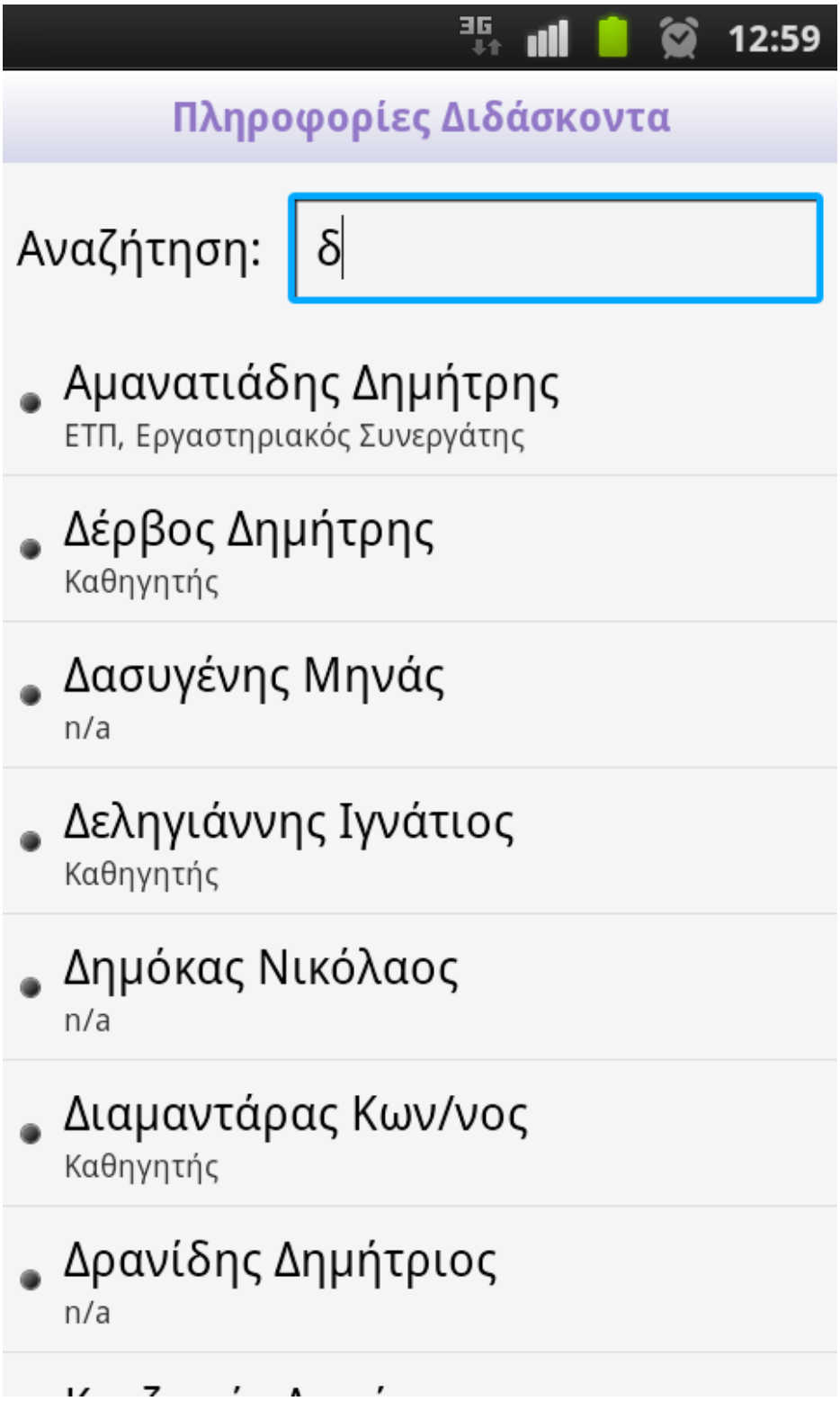


Σχέδιο 30: Οθόνη εμφάνισης προσωπικών στοιχείων

Με την υπηρεσία '**Τα στοιχεία μου**' ο φοιτητής μπορεί να δει πληροφορίες όπως ο αριθμός μητρώου, το τμήμα, το τυπικό εξάμηνο καθώς και πληροφορίες εγγραφής (απαιτεί σύνδεση στο pithia).

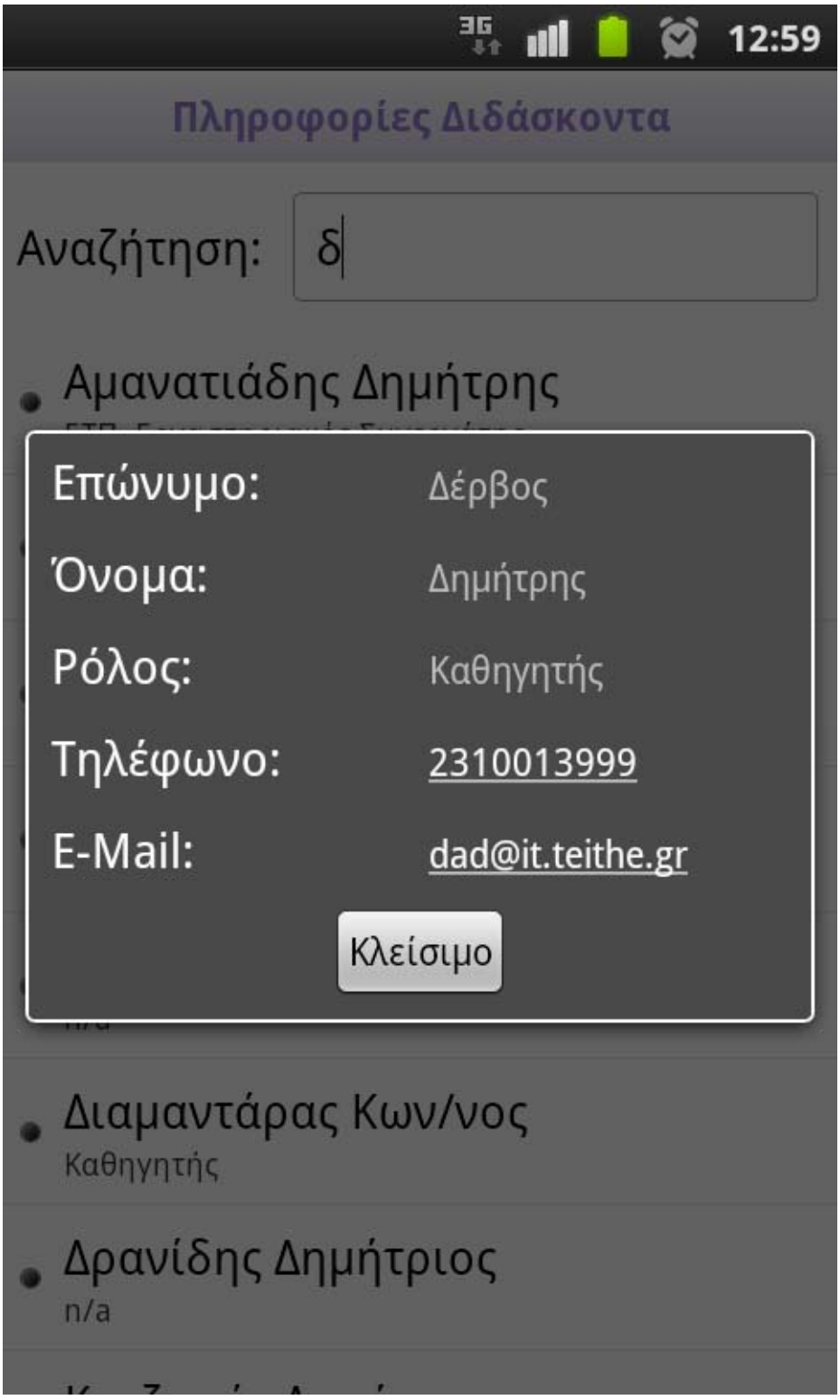
Σχέδιο 31: Οθόνη βαθμολογιών φοιτητή

Με την υπηρεσία '**Οι βαθμολογίες μου**' ο φοιτητής μπορεί να δει όλες τις βαθμολογίες του καθώς και τον μέσο όρο, μέχρι στιγμής, όπως εμφανίζεται στο pithia (απαιτεί σύνδεση στο pithia).



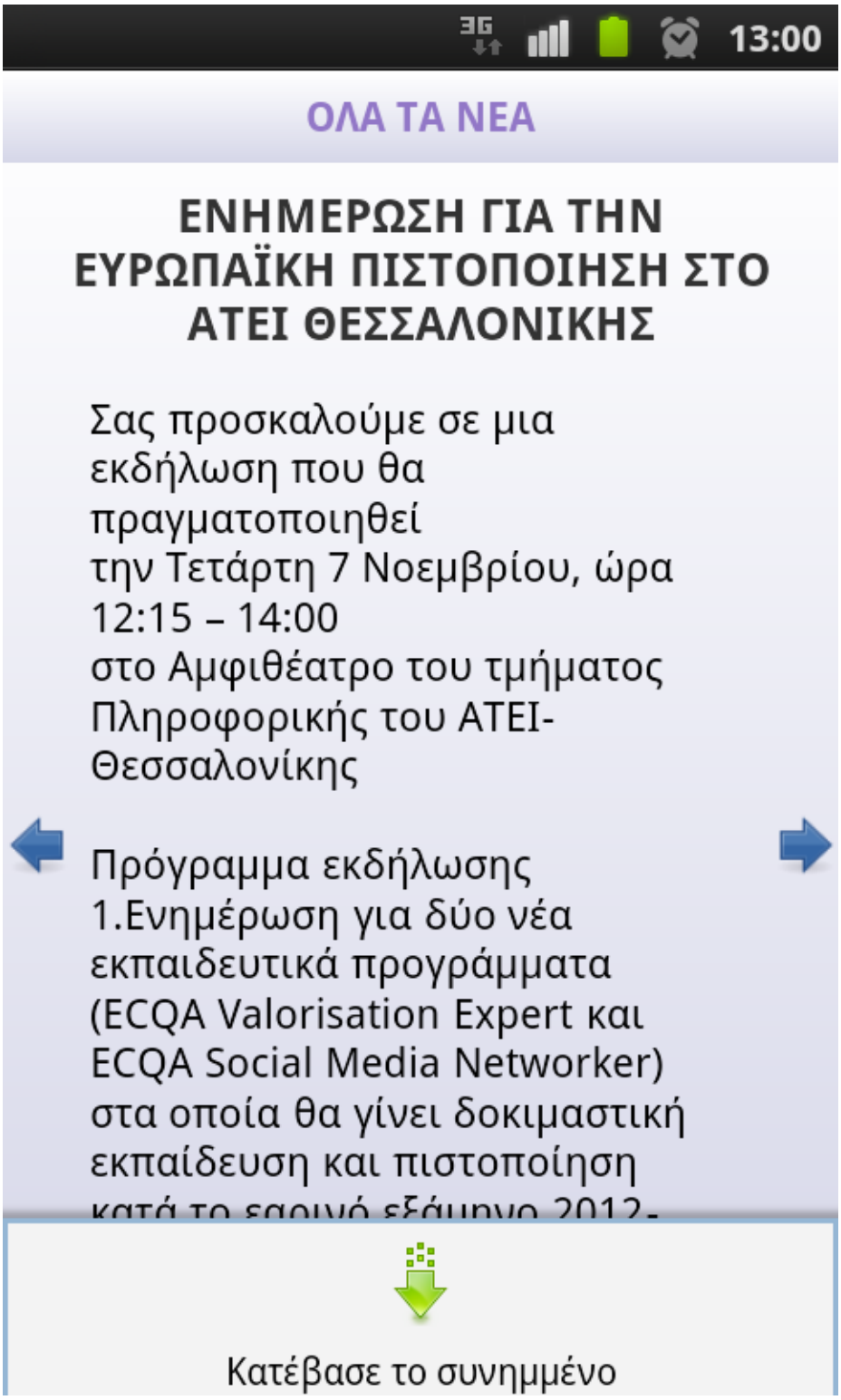
Σχέδιο 32: Οθόνη πληροφοριών διδασκόντων

Με την υπηρεσία '**Πληροφορίες Διδάσκοντα**' ο φοιτητής μπορεί να αναζητήσει πληροφορίες για οποιονδήποτε διδάσκοντα με βάση το όνομα ή το επίθετο του (απαιτεί σύνδεση στο hydra).



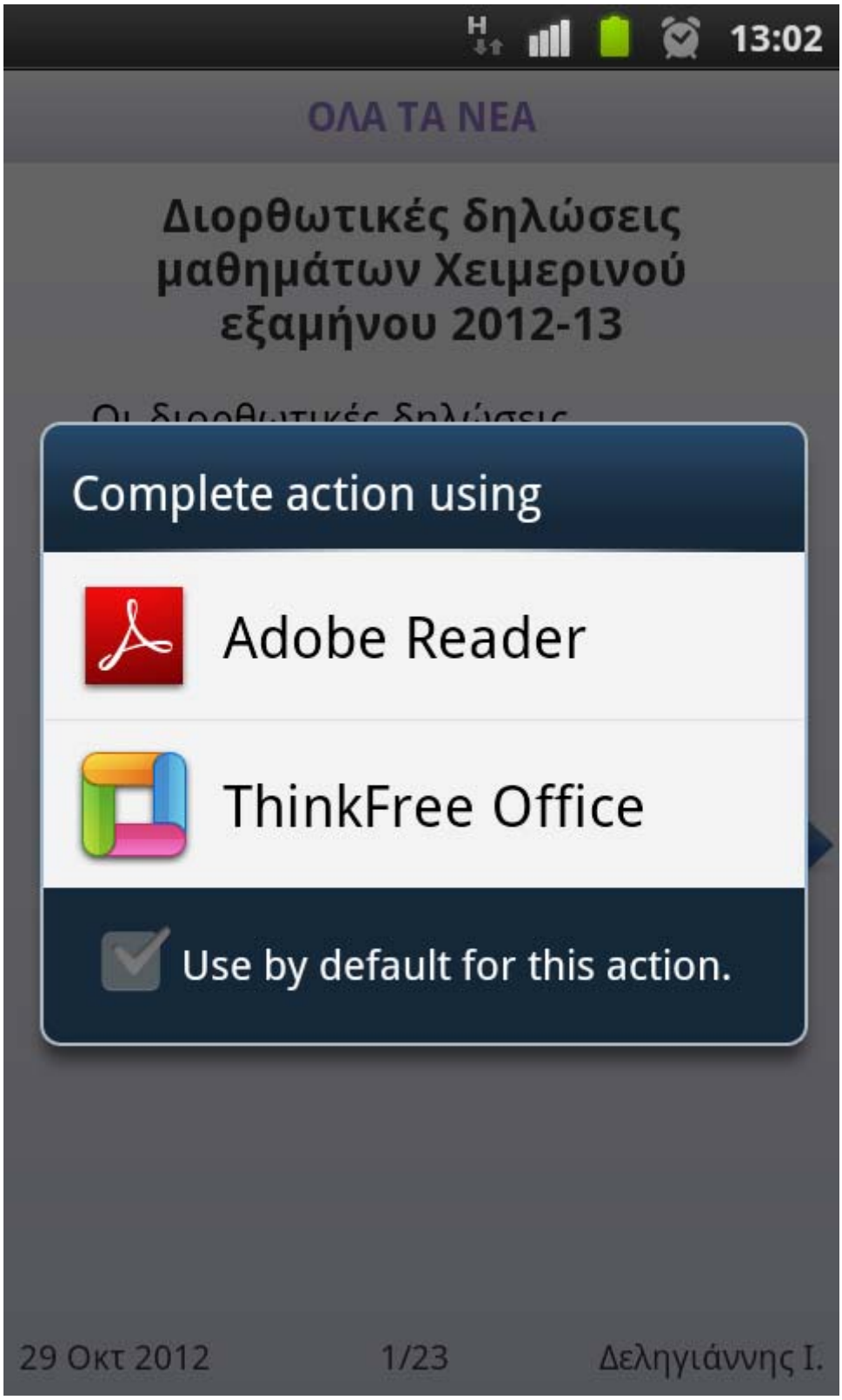
Σχέδιο 33: Οθόνη επιλογής διδάσκοντα

Οι πληροφορίες που εμφανίζονται κατά την επιλογή ενός διδάσκοντα από την λίστα. Τα πεδία τηλέφωνο και e-mail είναι επιλέξιμα (clickable) δηλαδή μπορεί να τα επιλέξει ο χρήστης είτε για να πάρει τηλέφωνο είτε για να στείλει e-mail (απαιτεί σύνδεση στο hydra).



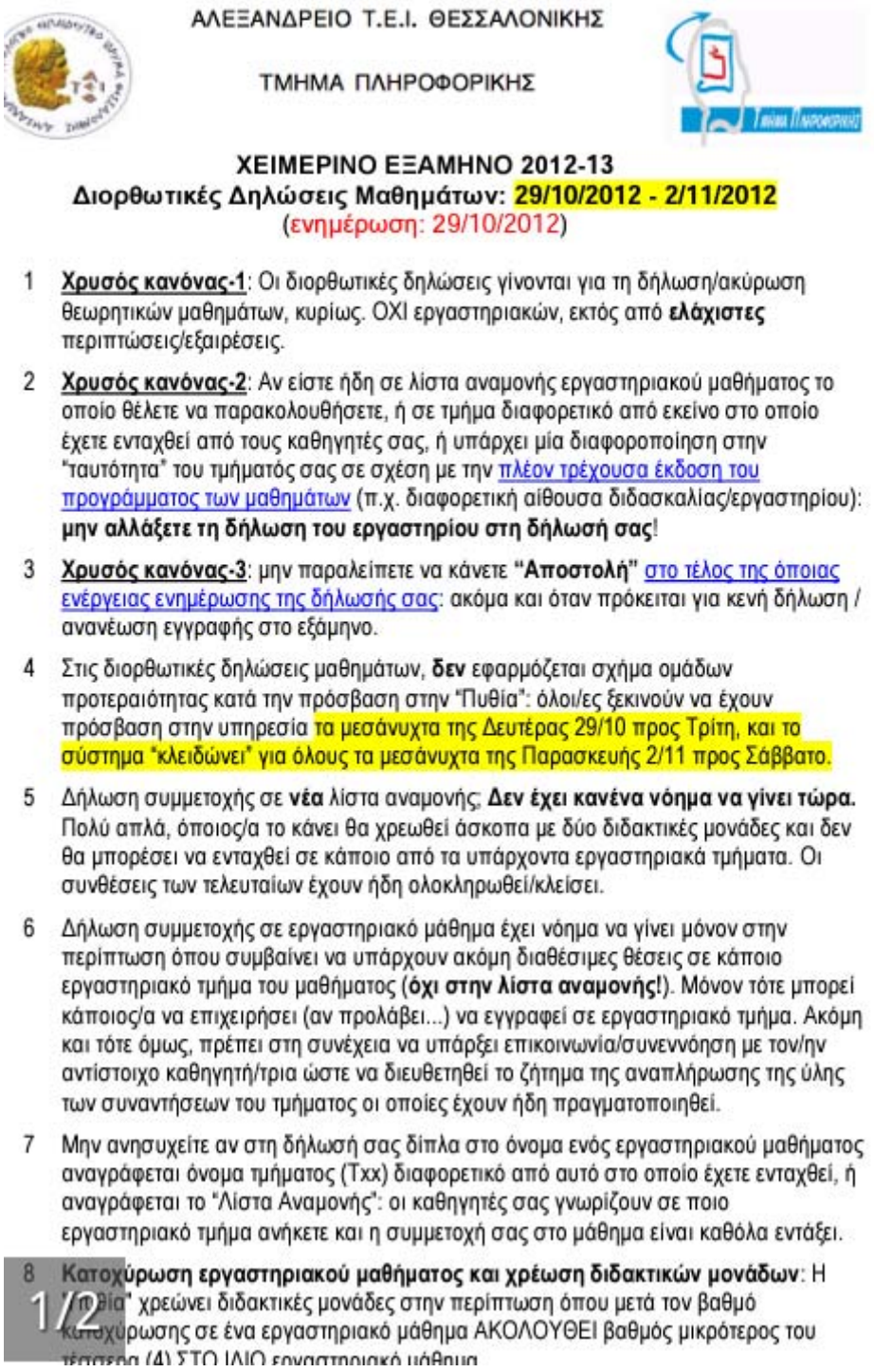
Σχέδιο 34: Οθόνη ανακοινώσεων υπηρεσίας Hydra

Στις ανακοινώσεις της Hydra ο χρήστης μπορεί να ενημερωθεί για τις πιο πρόσφατες ανακοινώσεις καθώς και να κατεβάσει το συνημμένο, εφόσον αυτό υπάρχει, πατώντας το option button του κινητού (απαιτεί σύνδεση στο hydra).



Σχέδιο 35: Οθόνη λήψης συνημμένου

Παραπάνω φαίνεται το παράθυρο διαλόγου κατά την λήψη του συνημμένου και το άνοιγμα του με την κατάλληλη εφαρμογή.



ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΧΕΙΜΕΡΙΝΟ ΕΞΑΜΗΝΟ 2012-13

Διορθωτικές Δηλώσεις Μαθημάτων: 29/10/2012 - 2/11/2012

(ενημέρωση: 29/10/2012)

1 **Χρυσός κανόνας-1**: Οι διορθωτικές δηλώσεις γίνονται για τη δήλωση/ακύρωση θεωρητικών μαθημάτων, κυρίως. ΟΧΙ εργαστηριακών, εκτός από **ελάχιστες** περιπτώσεις/εξαιρέσεις.

2 **Χρυσός κανόνας-2**: Αν είστε ήδη σε λίστα αναμονής εργαστηριακού μαθήματος το οποίο θέλετε να παρακολουθήσετε, ή σε τμήμα διαφορετικό από εκείνο στο οποίο έχετε ενταχθεί από τους καθηγητές σας, ή υπάρχει μία διαφοροποίηση στην "ταυτότητα" του τμήματός σας σε σχέση με την πλέον τρέχουσα έκδοση του προγράμματος των μαθημάτων (π.χ. διαφορετική αίθουσα διδασκαλίας/εργαστηρίου): **μην αλλάξετε τη δήλωση του εργαστηρίου στη δήλωσή σας**!

3 **Χρυσός κανόνας-3**: μην παραλείπετε να κάνετε **"Αποστολή"** στο τέλος της όποιας ενέργειας ενημέρωσης της δήλωσής σας: ακόμα και όταν πρόκειται για κενή δήλωση / ανανέωση εγγραφής στο εξάμηνο.

4 Στις διορθωτικές δηλώσεις μαθημάτων, **δεν** εφαρμόζεται σχήμα ομάδων προτεραιότητας κατά την πρόσβαση στην "Πυθία": όλοι/ες ξεκινούν να έχουν πρόσβαση στην υπηρεσία τα μεσάνυχτα της Δευτέρας 29/10 προς Τρίτη, και το σύστημα "κλειδώνει" για όλους τα μεσάνυχτα της Παρασκευής 2/11 προς Σάββατο.

5 Δήλωση συμμετοχής σε **νέα** λίστα αναμονής: **Δεν έχει κανένα νόημα να γίνει τώρα.** Πολύ απλά, όποιος/α το κάνει θα χρεωθεί άσκοπα με δύο διδακτικές μονάδες και δεν θα μπορέσει να ενταχθεί σε κάποιο από τα υπάρχοντα εργαστηριακά τμήματα. Οι συνθέσεις των τελευταίων έχουν ήδη ολοκληρωθεί/κλείσει.

6 Δήλωση συμμετοχής σε εργαστηριακό μάθημα έχει νόημα να γίνει μόνον στην περίπτωση όπου συμβαίνει να υπάρχουν ακόμη διαθέσιμες θέσεις σε κάποιο εργαστηριακό τμήμα του μαθήματος (**όχι στην λίστα αναμονής!**). Μόνον τότε μπορεί κάποιος/α να επιχειρήσει (αν προλάβει...) να εγγραφεί σε εργαστηριακό τμήμα. Ακόμη και τότε όμως, πρέπει στη συνέχεια να υπάρξει επικοινωνία/συνεννόηση με τον/την αντίστοιχο καθηγητή/τρια ώστε να διευθετηθεί το ζήτημα της αναπλήρωσης της ύλης των συναντήσεων του τμήματος οι οποίες έχουν ήδη πραγματοποιηθεί.

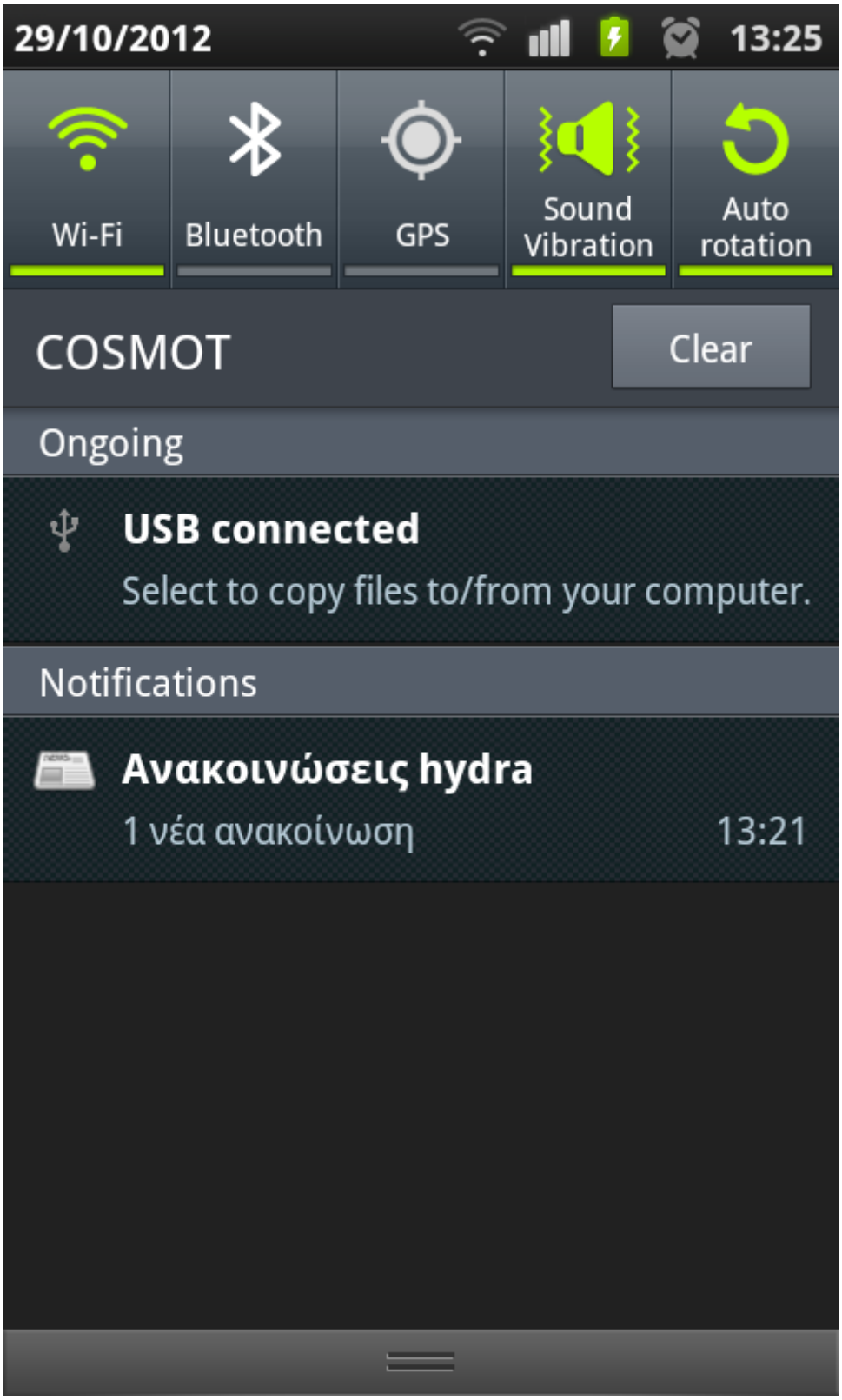
7 Μην ανησυχείτε αν στη δήλωσή σας δίπλα στο όνομα ενός εργαστηριακού μαθήματος αναγράφεται όνομα τμήματος (Τxx) διαφορετικό από αυτό στο οποίο έχετε ενταχθεί, ή αναγράφεται το "Λίστα Αναμονής": οι καθηγητές σας γνωρίζουν σε ποιο εργαστηριακό τμήμα ανήκετε και η συμμετοχή σας στο μάθημα είναι καθόλα εντάξει.

8 **Κατοχύρωση εργαστηριακού μαθήματος και χρέωση διδακτικών μονάδων**: Η "Πυθία" χρεώνει διδακτικές μονάδες στην περίπτωση όπου μετά τον βαθμό κατοχύρωσης σε ένα εργαστηριακό μάθημα ΑΚΟΛΟΥΘΕΙ βαθμός μικρότερος του τέσσερα (4) ΣΤΟ ΙΔΙΟ εργαστηριακό μάθημα

1/2

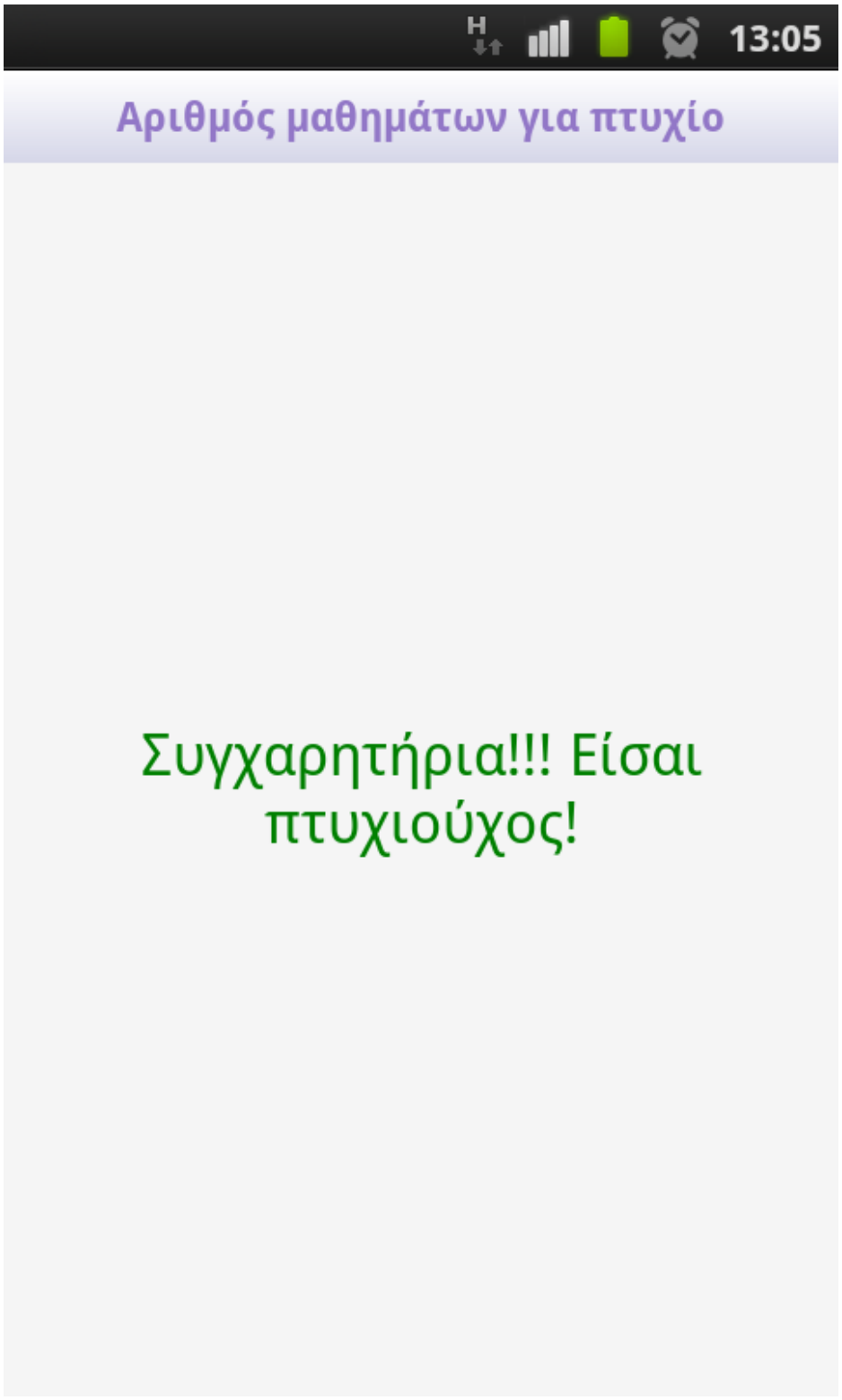
Σχέδιο 36: Οθόνη εμφάνισης συνημμένου

Η εμφάνιση του συνημμένου στην εφαρμογή adobe reader όπως φαίνεται στη συσκευή.



Σχέδιο 37: Οθόνη εμφάνισης ειδοποιήσεων

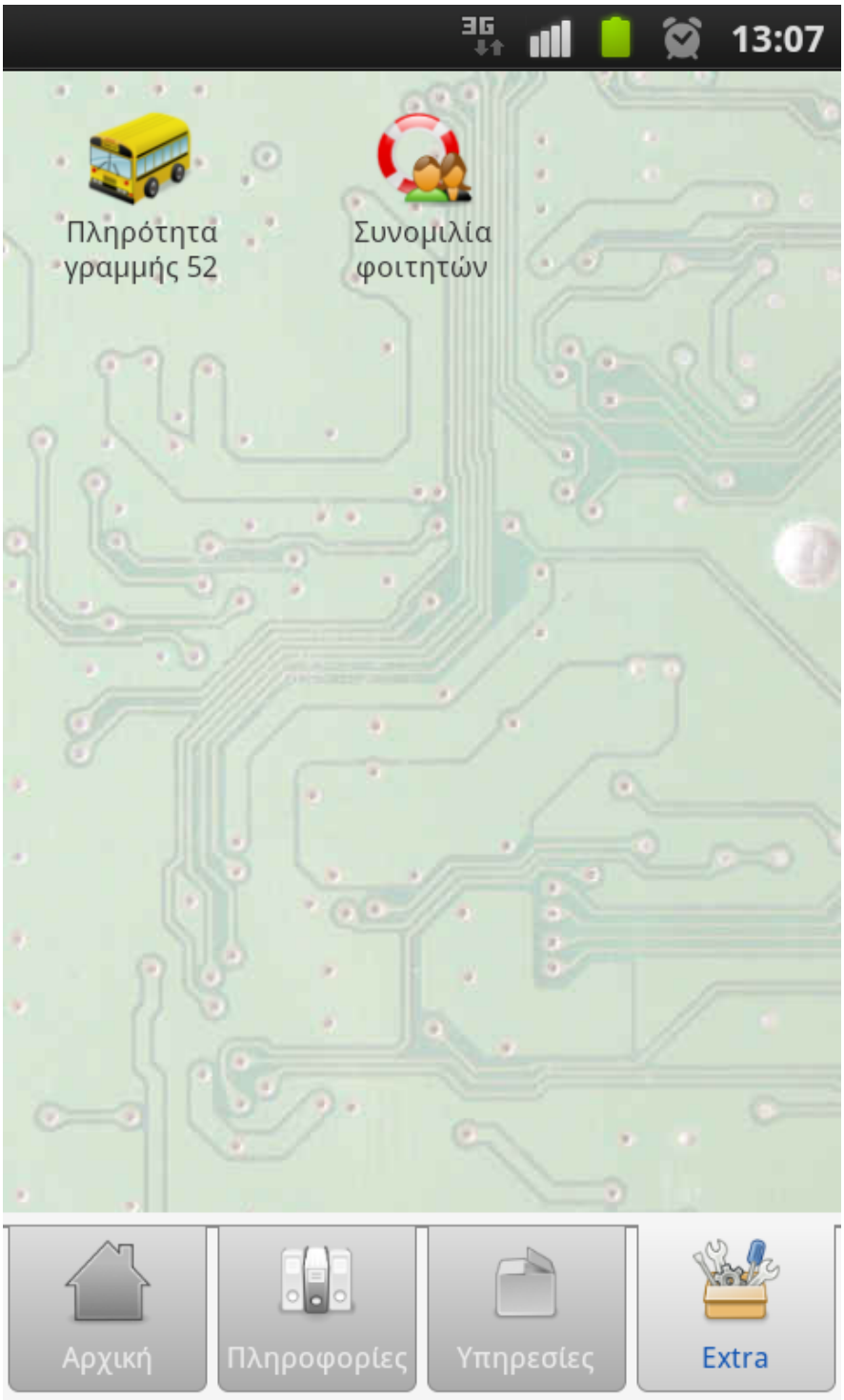
Ενημέρωση με ειδοποίηση (notification) στη συσκευή όταν υπάρχει νέα ανακοίνωση στο hydra (ενεργοποιείται από τις ρυθμίσεις και απαιτεί τα στοιχεία του λογαριασμού στην hydra).



Σχέδιο 38: Οθόνη εμφάνισης εναπομεινάντων μαθημάτων

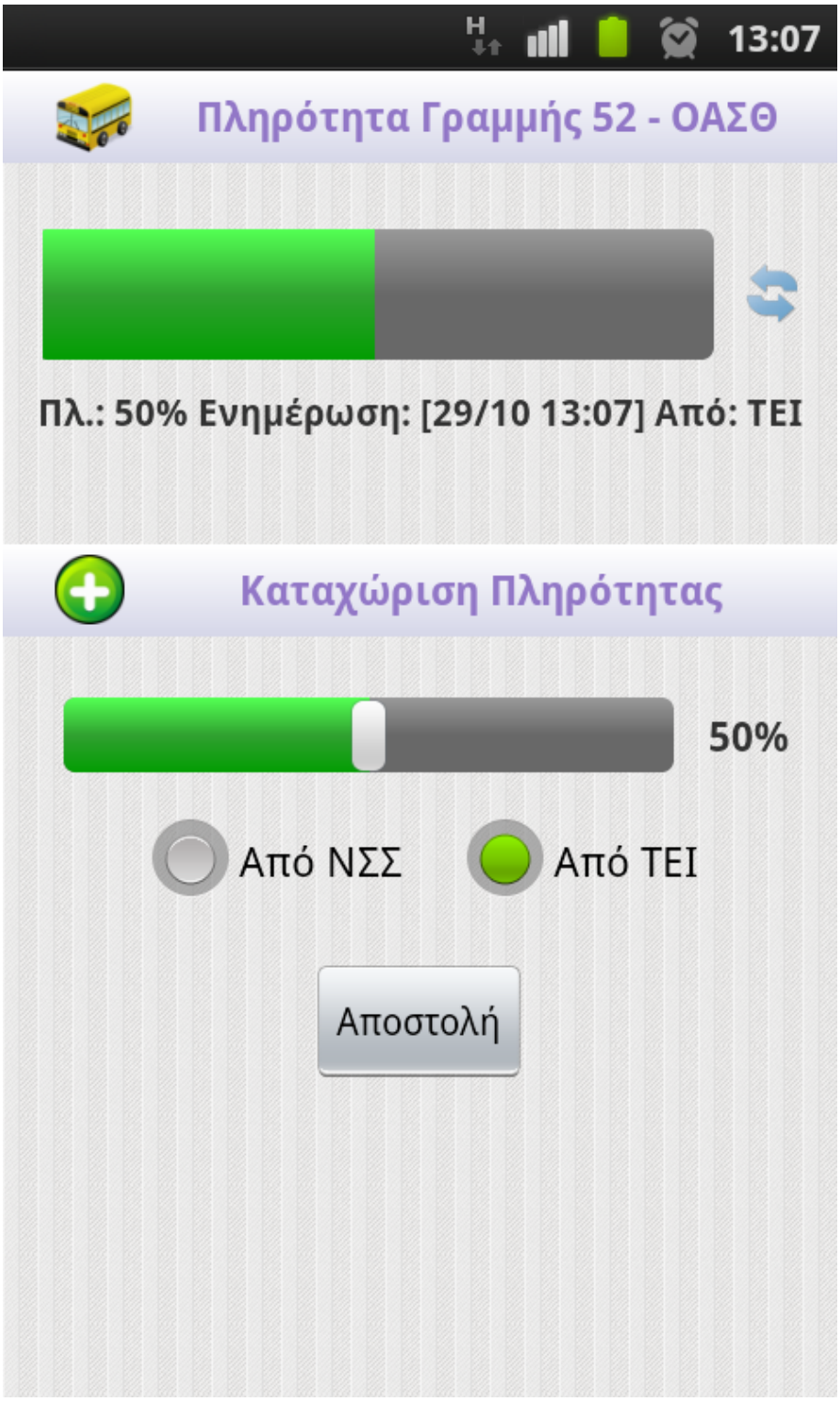
Με την υπηρεσία '**Αριθμός μαθημάτων για πτυχίο**' ο φοιτητής μπορεί να ενημερωθεί για τον αριθμό μαθημάτων που του απομένουν για να αποκτήσει το πτυχίο. Στην περίπτωσή μας ο παραπάνω φοιτητής έχει περάσει όλα τα μαθήματα, 35 στο σύνολο (απαιτεί σύνδεση pithia).

Καρτέλα επιπρόσθετων υπηρεσιών



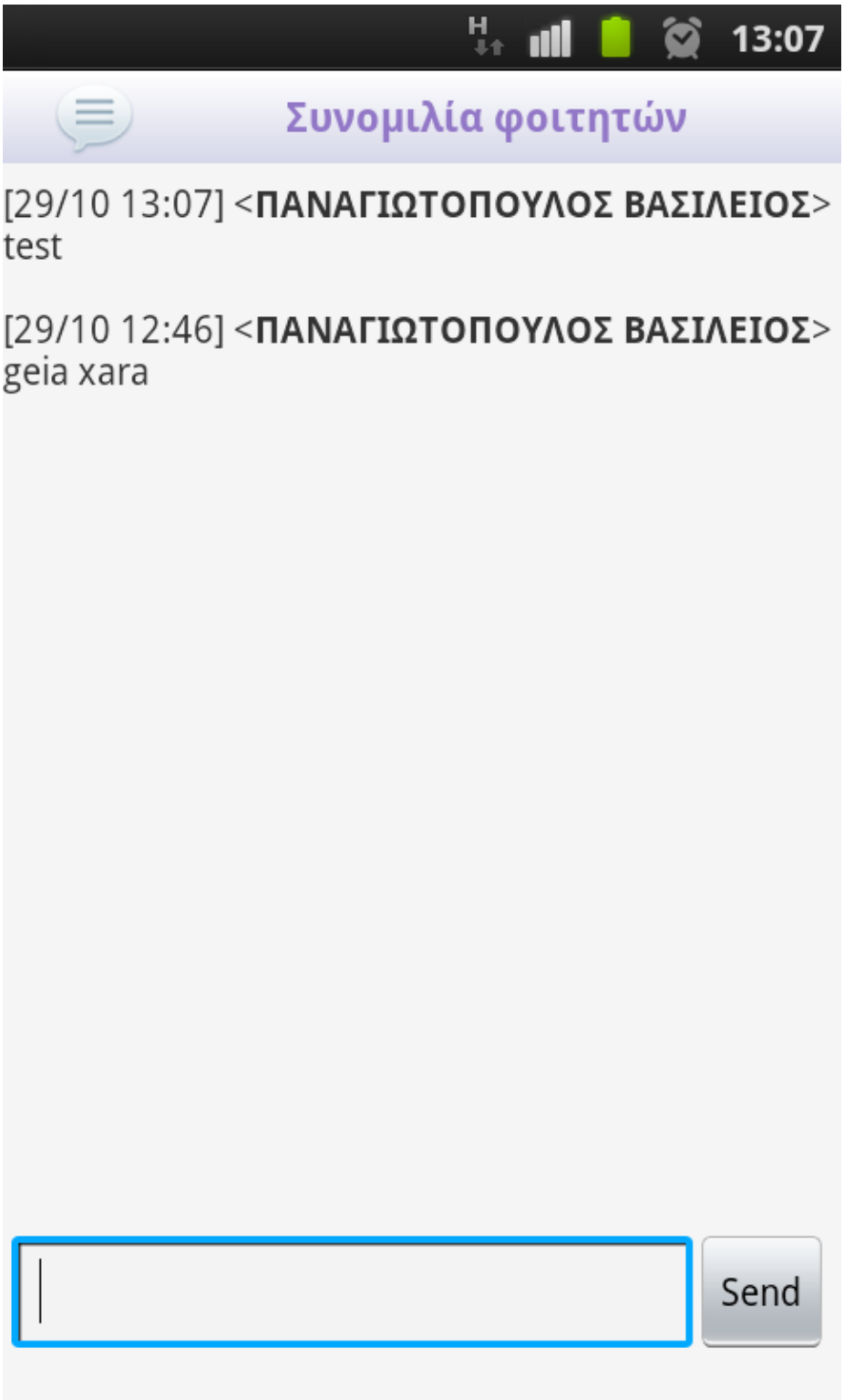
Σχέδιο 39: Καρτέλα πρόσθετων υπηρεσιών

Στις επιπρόσθετες υπηρεσίες οι φοιτητές μπορούν να ορίζουν και να ενημερώνονται για την πληρότητα της γραμμής 52 και, επίσης, να συνομιλούν μεταξύ τους για φοιτητικά θέματα.



Σχέδιο 40: Οθόνη πληρότητας γραμμής 52

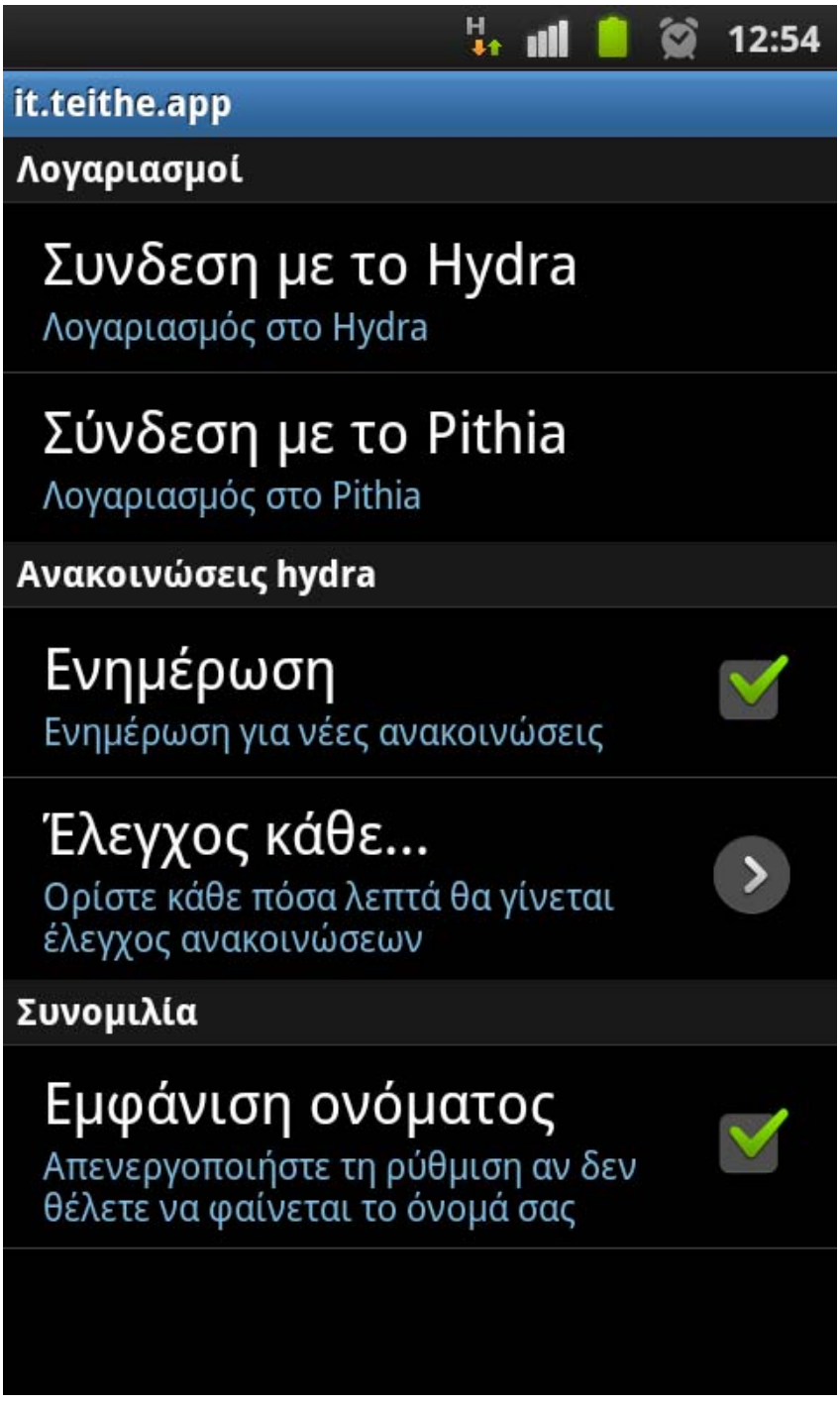
Στην οθόνη '**Πληρότητα Γραμμής 52 – ΟΑΣΘ**' ο φοιτητής μπορεί να ορίσει το ποσοστό πληρότητας του αστικού λεωφορείου είτε από τον Σταθμό προς το ΤΕΙ είτε το αντίστροφο. Επίσης, ενημερώνεται για τις τελευταίες ενημερώσεις πληρότητας των χρηστών.



Σχέδιο 41: Οθόνη συνομιλίας φοιτητών

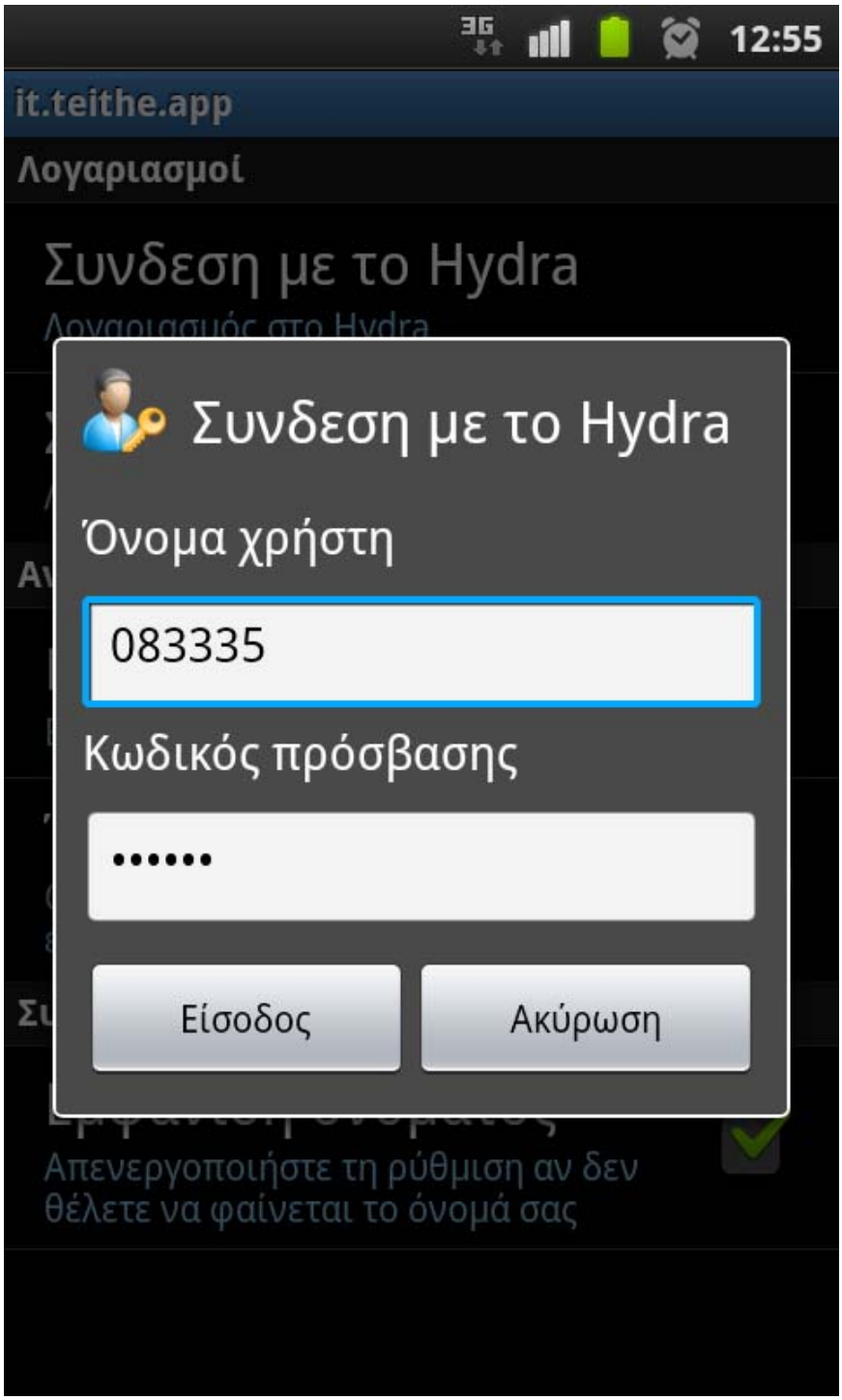
Στην οθόνη '**Συνομιλία φοιτητών**' οι φοιτητές μπορούν να ανταλλάσσουν μηνύματα για φοιτητικά θέματα (η εμφάνιση του ονόματος είναι προαιρετική και μπορεί να απενεργοποιηθεί από τις ρυθμίσεις).

Ρυθμίσεις εφαρμογής



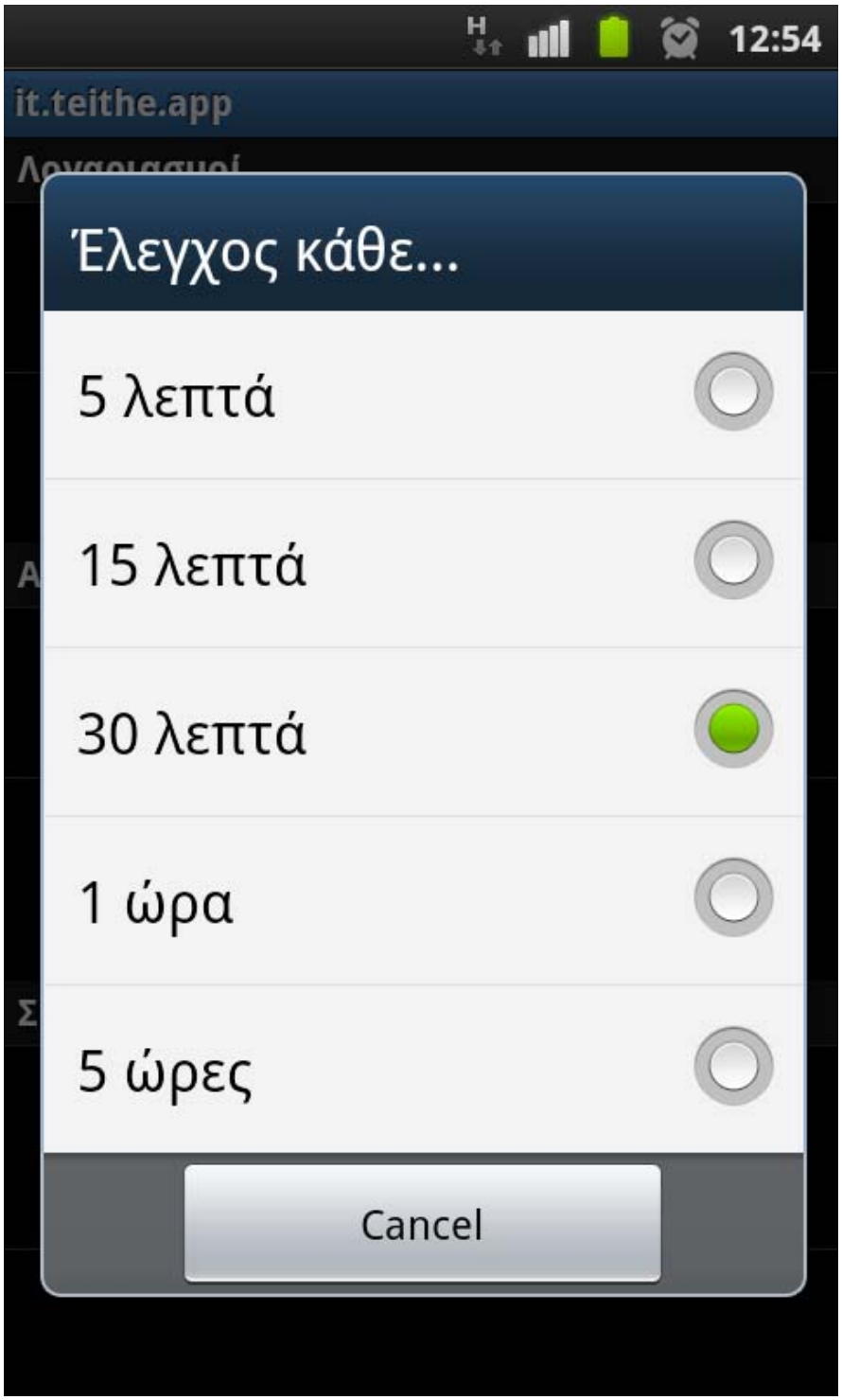
Σχέδιο 42: Οθόνη ρυθμίσεων εφαρμογής

Στις ρυθμίσεις ο χρήστης μπορεί να ορίσει τα στοιχεία για τους λογαριασμούς που έχει στο hydra και στο pithia (απαιτούνται για την χρήση των υπηρεσιών στη τρίτη καρτέλα), να ενεργοποιήσει την ενημέρωση για νέες ανακοινώσεις καθώς και να ορίσει κάθε πότε θα γίνεται ο έλεγχος αυτός. Τέλος, μπορεί να ενεργοποιήσει την επιλογή της εμφάνισης του Ονοματεπώνυμου του στη συνομιλία.



Σχέδιο 43: Οθόνη εισαγωγής προσωπικών στοιχείων - Hydra

Παραπάνω φαίνεται το παράθυρο διαλόγου για την σύνδεση στο hydra.



Σχέδιο 44: Οθόνη επιλογής χρόνου για έλεγχο ενημερώσεων

Παραπάνω φαίνεται το παράθυρο για την επιλογή του χρόνου που θα απαιτείται για κάθε επαναλαμβανόμενο έλεγχο για νέες ανακοινώσεις.

Περί της εφαρμογής

Σχέδιο 45: Οθόνη στοιχείων εφαρμογής

Οθόνη με συνοπτική περιγραφή της εφαρμογής (About box).