Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής

Πρόγραμμα Μεταπτυχιακών Σπουδών

«Πληροφορική»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**«Εφαρμογή ενημέρωσης των χρηστών για τις λίμνες της Ελλάδας σε πλατφόρμα Android»** **«Implementation of user information for lakes in Greece platform to Android»**
Ονοματεπώνυμο Φοιτητή	**Γεράσιμος Κουτσομπίνας**
Πατρώνυμο	**Δημήτριος**
Αριθμός Μητρώου	**ΜΠΠΛ/10054**
Επιβλέπων	**Κατερίνα Καμπάση**

Ημερομηνία Παράδοσης **Οκτώβριος 2013**

Ευχαριστίες

Η παρούσα μεταπτυχιακή διατριβή πραγματοποιήθηκε στα πλαίσια του Προγράμματος Μεταπτυχιακών Σπουδών στην Πληροφορική του Πανεπιστημίου Πειραιώς.

Πρωτίστως, θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα, κύριο Ευθύμιο Αλέπη, για την πολύτιμη καθοδήγησή του, την προτροπή, τη συνεργασία και το αληθινό του ενδιαφέρον. Επίσης, ευχαριστώ θερμά όλους τους ανθρώπους που στη διάρκεια των σπουδών μου συνέβαλλαν με τέτοιο τρόπο ώστε να με πείσουν πως η μόρφωση είναι μια διαρκής και αέναη διαδικασία.

Τέλος, οφείλω θερμές ευχαριστίες στην οικογένεια μου για την στήριξη και σε όσους από το φιλικό και συναδελφικό περιβάλλον με πραγματικό ενδιαφέρον στήριξαν την προσπάθειά μου.

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Κατερίνα Καμπάση Καθηγήτρια	Μαρία Βίρβου Καθηγήτρια	Γεώργιος Τσιχριντζής Καθηγητής

Πίνακας περιεχομένων

1 Περίληψη

Η εργασία εκπονήθηκε στα πλαίσια του μεταπτυχιακού, με σκοπό την έρευνα πάνω στην περιβαλλοντική πληροφορική που υποστηρίζει την χρήση υπολογιστών για τη προστασία του περιβάλλοντος. Για την ακρίβεια μια εφαρμογή που εκτελείται στο emulator του Android και διαθέτει μια βάση δεδομένων σε SQLite που θα εμφανίζονται οι λίμνες.

Στην εφαρμογή που αναπτύσσεται στην συνέχεια της διατριβής χρησιμοποιούμε διάφορα προγράμματα, ώστε να είναι λειτουργική, εύχρηστη αλλά και φιλική προς τους χρήστες. Τα προγράμματα που εγκαταστάθηκαν στο λειτουργικό του laptop είναι Windows7, Eclipse java IDE Version 3.8.2, Εργαλεία Android SDK, SUN JAVA SE Development Kit(JDK) 7 Update 25 αλλά και το sqlitebrowser. Η χρήση των πιο πάνω προγραμμάτων αναπτύσσεται πιο κάτω αναλυτικά και με κάθε λεπτομέρεια από την στιγμή που ξεκίνησαν μέχρι και σήμερα με βάση όσα καταγράφονται στο internet.

Στη συνέχεια, έχουμε αναλύσει και συνοψίσει τα δυο ακόλουθα paper, το Location Based User Modeling in Adaptive Mobile Learning for Environmental Awareness και το Mobile Applications for Industrial Environmental Protection. Αφορούν δυο paper, τα οποία αναπτύσσουν συστήματα τα οποία αλληλεπιδρούν με τους χρήστες για την προστασία του περιβάλλοντος. Ειδικά, τα συστήματα αυτά χρησιμοποιούν κινητές συσκευές, ένα κεντρικό σύστημα διαχείρισης αλλά και την βοήθεια των χρηστών.

Επίσης, δημιουργήσαμε και ένα help με σκοπό την κατανόηση και την εύκολη χρήση της εφαρμογής από άτομα που δεν είναι εξοικειωμένα με τις σύγχρονες συσκευές αλλά και τα σύγχρονα λειτουργικά όπως το Android στην προκειμένη περίπτωση. Έχουμε σχεδιάσει βήμα βήμα την κάθε λειτουργία που εκτελεί η εφαρμογή, με τα κατάλληλα print screen που δείχνουν κάθε φορά την λειτουργία που εκτελεί ο emulator του Android, για να καταλάβει ο αναγνώστης πώς να χρησιμοποιεί το πρόγραμμα προς όφελος της προστασίας του περιβάλλοντος.

Παρακάτω, αναπτύσσεται ο κώδικας που έχουμε υλοποιήσει για να τρέχει το πρόγραμμα. Η σχεδίαση της εφαρμογής διαθέτει τέσσερις οθόνες, την splash, menu, row αλλά και την simple_spinner_item. Η προσθήκη πόρων της εφαρμογής, την διαμορφώνει ουσιαστικά καλλωπιστικά για να είναι εύχρηστη. Ακόμη, γίνεται επεξεργασία του Αρχείου Manifest του Android αλλά και η Υλοποίηση των Δραστηριοτήτων Εφαρμογής, που συνδέει το user interface με την βάση δεδομένων SQLite για να πάρει ζωή η εφαρμογή μας.

Τελικά, η εργασία είχε σαν κύριο σκοπό να βοηθήσει στην προστασία του περιβάλλοντος από εξωγενείς παράγοντες που το απειλούν, αλλά και την ανάπτυξη του προγράμματος σε Android, η οποία είναι μια σύγχρονη πλατφόρμα.

Abstract

The work was performed as part of the Master, to research on environmental information technology that supports the use of computers for environmental protection. In fact an application that runs in the Android emulator and has a database in SQLite to display the lakes.

In the application developed in the thesis then use various programs to be functional, easy to use and user friendly. The programs that were installed in the operating of this laptop is Windows7, Eclipse java IDE Version 3.8.2, Tools Android SDK, SUN JAVA SE Development Kit (JDK) 7 Update 25 but the sqlitebrowser. The use of the above programs developed in detail below and in detail by the time we started up to date with what is recorded on the internet.

Then, we analyze and summarize the two following paper, the Location Based User Modeling in Adaptive Mobile Learning for Environmental Awareness and Mobile Applications for Industrial Environmental Protection. Involving two papers, which develop systems that interact with users to protect the environment. Specifically, these systems use mobile devices, a central management system and the help of users.

We also created a help for understanding and ease of use of the application by persons who are not familiar with modern appliances and modern functional such as Android in this case. We have designed step by step each function it performs the app, with appropriate print screen showing each time the function performed by the emulator of Android, for the reader to understand how to use the program in favor of environmental protection.

Below, we develop the code that we have implemented to run the program. The application design has four screens, the splash, menu, row and also simple_spinner_item. Adding resources to the application, the forms essentially ornamental to be handy. Still being worked on the Android Manifest File and implementation of enforcement activity, connecting the user interface to the SQLite database to get life our application.

Ultimately, the work had as main objective to help protect the environment by external factors threatening it, and program development in Android, which is a modern platform.

2 Εισαγωγή

Στα πλαίσια της μεταπτυχιακής διατριβής θα αναπτύξουμε μια εφαρμογή σε android που θα εμφανίζει τις λίμνες της Ελλάδας καθώς θα δίνει και την επιφάνεια καθεμιάς από αυτές. Η συγκεκριμένη εφαρμογή γίνεται με σκοπό την προστασία των λιμνών από εξωγενής παράγοντες. Ένα χρήσιμο εργαλείο ώστε να επιτευχθούν οι στόχοι που θέτουμε και ειδικότερα για την προστασία αυτού του νευραλγικού χώρου όπως οι λίμνες είναι η Περιβαλλοντική Πληροφορική (Environmental Informatics ή Enviromatics) αποτελεί αναπόσπαστο κομμάτι της Εφαρμοσμένης Πληροφορικής και υποστηρίζει μεθοδολογικά την χρήση υπολογιστών για την προστασία του περιβάλλοντος. Στο πλαίσιο αυτό, επιστρατεύονται μέθοδοι, τεχνικές και εργαλεία από την Επιστήμη Υπολογιστών για την ανάλυση, υποστήριξη και εδραίωση διαδικασιών επεξεργασίας πληροφορίας που συμβάλλουν στην διερεύνηση, την αποφυγή και τον περιορισμό της υποβάθμισης και της καταστροφής του περιβάλλοντος. Μια μεγάλη πρόκληση για την Περιβαλλοντική Πληροφορική είναι η ολοκλήρωση ετερογενών συστημάτων σε ένα κατανεμημένο Σύστημα Περιβαλλοντικής Πληροφορίας (ΣΠΕ). Η πρόκληση αυτή απαιτεί:

- τη δημιουργία νέων μεθόδων πλοήγησης και επεξεργασίας δεδομένων

- την παροχή υπηρεσιών μετα-δεδομένων (metadata) και συν-δεδομένων (co-data)
- την αποτελεσματική χρήση των υπαρχόντων δομών και δεδομένων

Με τον τρόπο αυτό η Περιβαλλοντική Πληροφορική θα μπορέσει να υποστηρίξει τους υπεύθυνους λήψης αποφάσεων στην προσπάθεια τους να συσχετίσουν τη γνώση για το περιβάλλον με κοινωνικούς, οικονομικούς και οικολογικούς στόχους για το μέλλον. Ταυτόχρονα, θα είναι δυνατή η παροχή απευθείας υπηρεσιών στους πολίτες και η βελτίωση της ζωής τους στα πλαίσια της βιώσιμης ανάπτυξης.

Υπηρεσίες και εφαρμογές για χρήστες με κινητά υπολογιστικά συστήματα, φορητές συσκευές, έξυπνα τηλέφωνα (σε λειτουργικά συστήματα, όπως το Android, το iOS, το Symbian, και το Windows Mobile) αναπτύσσονται ραγδαία με σκοπό την παροχή της πληροφορίας με βάση τη θέση του χρήστη ή τον χώρο στον οποίο βρίσκεται. Επίσης, η ανάγκη για υπηρεσίες εντοπισμού θέσης στον κλάδο των μεταφορών και το εμπόριο, σε συνθήκες έκτακτων αναγκών και αντιμετώπισης καταστροφών καθώς και σε υπηρεσίες πληροφορικής στο χώρο της υγείας, έχει επιταχύνει την ανάπτυξη συστημάτων εντοπισμού θέσης.
Τα συστήματα εύρεσης θέσης ποικίλουν με βάση τη χρήση εξειδικευμένης υποδομής και υλισμικό (hardware), την ακρίβεια που παρέχουν, τη λειτουργία τους σε εσωτερικούς ή εξωτερικούς χώρους, την παροχή λειτουργιών για ταυτοποίηση,

αναγνώριση και κατηγοριοποίηση, καθώς και το κόστος τους. Το Global Positioning System (GPS) χρησιμοποιεί δορυφόρους και είναι το πιο διαδεδομένο σύστημα εύρεσης θέσης σε εξωτερικούς χώρους, ενώ τα GSM τηλέφωνα παρέχουν επίσης πληροφορία θέσης της συσκευής. Η ευρεία εξάπλωση των ασύρματων δικτύων (π.χ., WiFi), με το χαμηλό κόστος εγκατάστασής τους, τα καθιστούν μια ελκυστική επιλογή για την ανάπτυξη συστημάτων εντοπισμού θέσης πέρα των τηλεπικοινωνιακών υπηρεσιών που παρέχουν. Επίσης, περιβάλλοντα με διατάξεις αισθητήρων διαφόρων τεχνολογιών (π.χ., RFID, ultrasonic, WiFi, πίεσης, κάμερα) αναπτύσσονται για τον ακριβέστερο υπολογισμό της θέσης.

Σημαντικές αγορές και πεδία εφαρμογών αποτελούν τα γεωγραφικά συστήματα πληροφοριών, το εμπόριο, η διαχείριση στόλου οχημάτων, “Vehicle-to-Vehicle” επικοινωνία , η διαχείριση αποθήκης εμπορευμάτων, το περιβάλλον, οι καλλιέργειες, η ενέργεια, η πολιτισμική πληροφορική, η πολιτική προστασία, οι μεταφορές, η υγεία (e-health), ο τουρισμός, οι υπηρεσίες προς κινητούς χρήστες, η ψυχαγωγία (home entertainment, games, home appliances), η κοινωνική δικτύωση, και τα περιβάλλοντα διάχυτης νοημοσύνης.

Περιβάλλον Ανάπτυξης Εφαρμογής

Ο κώδικας της συγκεκριμένης εργασίας γράφτηκε χρησιμοποιώντας τα παρακάτω περιβάλλοντα ανάπτυξης:

- Windows 7
- Eclipse java IDE Version 3.8.2
- Εργαλεία Android SDK
- SUN JAVA SE Development Kit(JDK) 7 Update 25

Windows 7

Τα Windows 7 (προηγουμένως γνωστά με τις κώδικες ονομασίες Blackcomb και Vienna) είναι ο διάδοχος των Windows Vista. Τα Windows 7 κυκλοφόρησαν στις 22 Οκτωβρίου 2009 και σε 32-bit και σε 64-bit εκδόσεις.

Η Microsoft είχε διατηρήσει άκρα μυστικότητα για τα Windows 7, καθώς είχε δώσει έμφαση στην κυκλοφορία και στην ανάπτυξη των Windows Vista, δηλώνοντας ότι δεν θέλει να υποσχεθεί χαρακτηριστικά των Windows 7 που τελικά δεν θα ενσωματωθούν, παρόλο που σε προηγούμενα λειτουργικά είχε αποκαλύψει κάποια χαρακτηριστικά πριν την κυκλοφορία τους. Ως αποτέλεσμα, γνωρίζουμε πολύ λίγα για τα Windows 7, αλλά δημόσιες ομιλίες και παρουσιάσεις από συνεργάτες της Microsoft έχουν δώσει εικόνα για τα χαρακτηριστικά τους. Μάλιστα, πληροφορίες που δημοσιεύτηκαν στο διαδίκτυο από άτομα στα οποία δόθηκε η έκδοση M1 των Windows 7 έχουν δώσει περισσότερες λεπτομέρειες.

Κατά τη διάρκεια της κεντρικής του ομιλίας στη διεθνή έκθεση Consumers Electronics Show 2009, στο Λ. Βέγκας, ο Steve Ballmer, Chief Executive Officer της Microsoft, ανακοίνωσε τη διάθεση της δοκιμαστικής (beta) έκδοσης του λειτουργικού συστήματος Windows 7**,** καθώς και τη διάθεση της νέας έκδοσης του Windows Live, της πλατφόρμας υπηρεσιών και εφαρμογών για την επικοινωνία στο διαδίκτυο. Κατά την άποψη της εταιρείας τα νέα Windows 7 και το Windows Live θα διευκολύνουν ακόμα περισσότερο την επικοινωνία, την κοινή χρήση αρχείων και τη διεκπεραίωση εργασιών.

Η πρώτη δομή των Windows 7 ονομάστηκε «Milestone 1 (M1)», σύμφωνα με αναφορές στο TG Daily, και έχει αριθμό έκδοσης 6.1.6519.1. Δόθηκε σε συνεργάτες της Microsoft τον Ιανουάριο του 2008 σε εκδόσεις x86 και x64. Αν και δεν έχει επιβεβαιωθεί από τη Microsoft, έχουν κυκλοφορήσει αναφορές και εικόνες από διάφορες πηγές. Τα χαρακτηριστικά που περιλαμβάνονται είναι οι μικροεφαρμογές, που ενσωματώθηκαν στην Εξερεύνηση των Windows, μία μικροεφαρμογή για το Windows Media Center, η δυνατότητα καρφιτσώματος και ξεκαρφιτσώματος στοιχείων από το Μενού Έναρξη και τον Κάδο Ανακύκλωσης, βελτιωμένα χαρακτηριστικά πολυμέσων και ένα νέο πρόγραμμα προβολής εγγράφων XPS. Οι αναφορές επίσης δείχνουν ότι ένα εργαλείο αποστολής πληροφοριών στη Microsoft Windows 7 εμφανίζει κάποια άλλα χαρακτηριστικά: την ικανότητα αποθήκευσης ρυθμίσεων λογαριασμού του Windows Live στον Internet Explorer, νέες εκδόσεις της Αριθμομηχανής, της Ζωγραφικής και του WordPad, που χρησιμοποιούν το Windows Presentation Foundation. Επίσης, αναφέρεται ότι η διάρκεια εγκατάστασης του λειτουργικού συστήματος διαρκεί 10 λεπτά. Αλλαγές στο περιβάλλον εργασίας έγιναν στις τελευταίες δοκιμαστικές εκδόσεις των Windows 7, όπως η οθόνη εκκίνησης, θυμίζοντας τις προ-XP εκδόσεις των Windows.

Επόμενες εκδόσεις

Η έκδοση Milestone 2 (M2) κυκλοφόρησε τον Απρίλιο του 2008 και η έκδοση Milestone 3 (M3) κυκλοφόρησε τον Σεπτέμβριο του 2008 , ενώ η έκδοση για τους κατασκευαστές υπολογιστών (RTM) κυκλοφόρησε το δεύτερο μισό του 2009. Η πρώτη beta έκδοση των Windows 7 έγινε διαθέσιμη στο ευρύ κοινό στις 7 Ιανουαρίου 2009 ενώ η Release Candidate 1 κυκλοφόρησε τέλος Μαΐου του ίδιου έτους.

Ο Μπιλ Γκέιτς τον Απρίλιο του 2008 σχολίασε σε μια συνέντευξη τύπου ότι μια νέα έκδοση θα έρθει «του χρόνου ή κάπου εκεί». Σύμφωνα με πρόσθετο σχόλιο της Microsoft, ο Γκέιτς αναφερόταν μόνο σε εκδόσεις alpha ή beta των Windows 7[11].

MinWin

Μία μικρή διαφοροποίηση του Windows kernel, γνωστή ως MinWin, βρίσκεται σε ανάπτυξη για χρήση με τα Windows 7. Με αυτό, γίνονται προσπάθειες για να χρησιμοποιείται λιγότερη μνήμη και χώρος στο σκληρό δίσκο. Το MinWin χρησιμοποιεί περίπου 25 MB στο σκληρό δίσκο και 40 MB μνήμης. Δεν χρησιμοποιεί γραφικό περιβάλλον χρήσης, αλλά μια γραμμή εντολών πλήρους οθόνης. Περιλαμβάνει υποσυστήματα και συστήματα δικτύου I/O. Το MinWin παρουσιάστηκε για πρώτη φορά στις 13 Οκτωβρίου του 2007 από τον Eric Traut. Το σύστημα παρουσίασης περιελάμβανε μια εικόνα του λειτουργικού συστήματος, που είχε δημιουργηθεί από 100 αρχεία, στα οποία λειτουργούσε ένας server HTTP.

Το όνομα MinWin είχε αρχικά χρησιμοποιηθεί ως ονομασία για τον πυρήνα του server του Windows Server 2008. Όμως, αυτά τα δύο διαφέρουν πολύ.

Μέθοδοι εισαγωγής δεδομένων

Στις 11 Δεκεμβρίου του 2007, ο Hilton Locke, που εργαζόταν στην ομάδα του Tablet PC της Microsoft, ανέφερε ότι τα Windows 7 θα φέρουν νέα χαρακτηριστικά αφής. Αναφέρει: «Εάν είχατε εντυπωσιαστεί από τα «χαρακτηριστικά αφής» του iPhone, θα μείνετε άναυδοι με το τι θα συμβεί στα Windows 7. Ελπίζουμε να πείσουμε περισσότερες OEM ότι η Τεχνολογία Αφής των Windows θα ανεβάσει τις πωλήσεις τους.

Άλλες μέθοδοι εισαγωγής δεδομένων, που θα υποστηρίζει το λειτουργικό σύστημα, όπως είπε και ο Μπιλ Γκέιτς, είναι η αναγνώριση ομιλίας και η οπτική αναγνώριση χαρακτήρων.

Αντιμονοπωλιακός έλεγχος

Η ανάπτυξη των Windows 7 έχει ήδη προσελκύσει αντιμονοπωλιακούς ρυθμιστές, οι οποίοι επιβλέπουν τις ενέργειες της Microsoft βάσει της αρχής του Θέματος μονοπωλίου της Microsoft από τις Ηνωμένες Πολιτείες του 2001. Σύμφωνα με αναφορές, μια τριμελής επιτροπή άρχισε να αξιολογεί τα νέα Windows τον Φεβρουάριο του 2008.Ο Michael Gartenberg, ένας αναλυτής της JupiterResearch υπέβαλε το ερώτημα «η πρόκληση (της Microsoft) για τα Windows 7 θα είναι πώς θα συνεχίσουν να προσθέτουν χαρακτηριστικά που θα θέλουν οι καταναλωτές και δεν θα παραβιάζουν τους αντιμονοπωλιακούς κανόνες;

Java

Ιστορία

Στις αρχές του 1991, η *Sun* αναζητούσε το κατάλληλο εργαλείο για να αποτελέσει την πλατφόρμα ανάπτυξης λογισμικού σε μικρο-συσκευές (έξυπνες οικιακές συσκευές έως πολύπλοκα συστήματα παραγωγής γραφικών). Τα εργαλεία της εποχής ήταν γλώσσες όπως η C++ και η C. Μετά από διάφορους πειραματισμούς προέκυψε το συμπέρασμα ότι οι υπάρχουσες γλώσσες δεν μπορούσαν να καλύψουν τις ανάγκες τους. Ο "πατέρας" της Java, James Gosling, που εργαζόταν εκείνη την εποχή για την Sun, έκανε ήδη πειραματισμούς πάνω στη C++ και είχε παρουσιάσει κατά καιρούς κάποιες πειραματικές γλώσσες (C++ ++) ως πρότυπα για το νέο εργαλείο που

αναζητούσαν στην *Sun*. Τελικά μετά από λίγο καιρό κατέληξαν με μια πρόταση για το επιτελείο της εταιρίας, η οποία ήταν η γλώσσα *Oak*. Το όνομά της το πήρε από το ομώνυμο δένδρο (βελανιδιά) το οποίο ο Gosling είχε έξω από το γραφείο του και έβλεπε κάθε μέρα.

Από την Oak στη Java

Η *Oak* ήταν μία γλώσσα που διατηρούσε μεγάλη συγγένεια με την C++. Παρόλα αυτά είχε πολύ πιο έντονο αντικειμενοστρεφή (*object oriented*) χαρακτήρα σε σχέση με την C++ και χαρακτηριζόταν για την απλότητα της. Σύντομα οι υπεύθυνοι ανάπτυξης της νέας γλώσσας ανακάλυψαν ότι το όνομα *Oak* ήταν ήδη κατοχυρωμένο οπότε κατά την διάρκεια μιας εκ των πολλών συναντήσεων σε κάποιο τοπικό καφέ αποφάσισαν να μετονομάσουν το νέο τους δημιούργημα σε Java που εκτός των άλλων ήταν το όνομα της αγαπημένης ποικιλίας καφέ για τους δημιουργούς της. Η επίσημη εμφάνιση της *Java* αλλά και του *HotJava* (πλοηγός με υποστήριξη *Java*) στη βιομηχανία της πληροφορικής έγινε το Μάρτιο του 1995 όταν η *Sun* την ανακοίνωσε στο συνέδριο *Sun World* 1995. Ο πρώτος μεταγλωττιστής (*compiler*) της ήταν γραμμένος στη γλώσσα C από τον James Gosling. Το 1994, ο A.Van Hoff ξαναγράφει τον μεταγλωττιστή της γλώσσας σε *Java*, ενώ το Δεκέμβριο του 1995 πρώτες οι IBM, Borland, Mitsubishi Electronics, Sybase και Symantec ανακοινώνουν σχέδια να χρησιμοποιήσουν τη *Java* για την δημιουργία λογισμικού. Από εκεί και πέρα η Java ακολουθεί μία ανοδική πορεία και είναι πλέον μία από τις πιο δημοφιλείς γλώσσες στον χώρο της πληροφορικής. Στις 13 Νοεμβρίου του 2006 η Java έγινε πλέον μια γλώσσα ανοιχτού κώδικα(GPL) όσον αφορά το μεταγλωττιστή (javac) και το πακέτο ανάπτυξης (JDK, Java Development Kit).

Η εξαγορά από την Oracle και το μέλλον της Java

Στις 27 Απριλίου 2010 η εταιρία λογισμικού Oracle Corporation ανακοίνωσε ότι μετά από πολύμηνες συζητήσεις ήρθε σε συμφωνία για την εξαγορά της Sun Microsystems και των τεχνολογιών (πνευματικά δικαιώματα/ πατέντες) που η δεύτερη είχε στην κατοχή της ή δημιουργήσει. Η συγκεκριμένη συμφωνία θεωρείται σημαντική για το μέλλον της Java και του γενικότερου οικοσυστήματος τεχνολογιών γύρω από αυτή μιας και ο έμμεσος έλεγχος της τεχνολογίας και η εξέλιξη της περνάει σε άλλα χέρια.

Τα χαρακτηριστικά της Java

Ένα από τα βασικά πλεονεκτήματα της Java έναντι των περισσότερων άλλων γλωσσών είναι η ανεξαρτησία του λειτουργικού συστήματος και πλατφόρμας. Τα προγράμματα που είναι γραμμένα σε *Java* τρέχουνε ακριβώς το ίδιο σε Windows, Linux, Unix και Macintosh (σύντομα θα τρέχουν και σε Playstation καθώς και σε άλλες κονσόλες παιχνιδιών) χωρίς να χρειαστεί να ξαναγίνει μεταγλώττιση (compiling) ή να αλλάξει ο πηγαίος κώδικας για κάθε διαφορετικό λειτουργικό σύστημα. Για να επιτευχθεί όμως αυτό χρειαζόταν κάποιος τρόπος έτσι ώστε τα προγράμματα γραμμένα σε Java να μπορούν να είναι «κατανοητά» από κάθε υπολογιστή ανεξάρτητα του είδους επεξεργαστή (Intel x86, IBM, Sun SPARC, Motorola) αλλά και λειτουργικού συστήματος (Windows, Unix, Linux, BSD, MacOS). Ο λόγος είναι ότι κάθε κεντρική μονάδα επεξεργασίας κατανοεί διαφορετικό κώδικα μηχανής. Ο συμβολικός κώδικας (*assembly*) που μεταφράζεται και εκτελείται σε Windows είναι διαφορετικός από αυτόν που μεταφράζεται και εκτελείται σε έναν υπολογιστή Macintosh. Η λύση δόθηκε με την ανάπτυξη της *Εικονικής Μηχανής* (*Virtual Machine* ή VM ή ΕΜ στα ελληνικά).

Η εικονική μηχανή της Java

Αφού γραφεί κάποιο πρόγραμμα σε Java, στη συνέχεια μεταγλωτίζεται μέσω του μεταγλωττιστή javac, ο οποίος παράγει έναν αριθμό από αρχεία .class (κώδικας byte ή bytecode). Ο κώδικας byte είναι η μορφή που παίρνει ο πηγαίος κώδικας της Java όταν μεταγλωττιστεί. Όταν πρόκειται να εκτελεστεί η εφαρμογή σε ένα μηχάνημα, το Java Virtual Machine που πρέπει να είναι εγκατεστημένο σε αυτό θα αναλάβει να διαβάσει τα αρχεία .class. Στη συνέχεια τα μεταφράζει σε γλώσσα μηχανής που να υποστηρίζεται από το λειτουργικό σύστημα και τον επεξεργαστή, έτσι ώστε να εκτελεστεί (να σημειωθεί εδώ ότι αυτό συμβαίνει με την παραδοσιακή Εικονική Μηχανή (Virtual Machine). Πιο σύγχρονες εφαρμογές της εικονικής Μηχανής μπορούν και μεταγλωττίζουν εκ των προτέρων τμήματα bytecode απευθείας σε κώδικα μηχανής (εγγενή κώδικα ή native code) με αποτέλεσμα να βελτιώνεται η ταχύτητα). Χωρίς αυτό δε θα ήταν δυνατή η εκτέλεση λογισμικού γραμμένου σε Java. Πρέπει να σημειωθεί ότι η JVM είναι λογισμικό που εξαρτάται από την πλατφόρμα, δηλαδή για κάθε είδος λειτουργικού συστήματος και αρχιτεκτονικής επεξεργαστή υπάρχει διαφορετική έκδοση του. Έτσι υπάρχουν διαφορετικές JVM για Windows, Linux, Unix, Macintosh, κινητά τηλέφωνα, παιχνιδομηχανές κλπ.

Οτιδήποτε θέλει να κάνει ο προγραμματιστής (ή ο χρήστης) γίνεται μέσω της εικονικής μηχανής. Αυτό βοηθάει στο να υπάρχει μεγαλύτερη ασφάλεια στο σύστημα γιατί η εικονική μηχανή είναι υπεύθυνη για την επικοινωνία χρήστη - υπολογιστή. Ο προγραμματιστής δεν μπορεί να γράψει κώδικα ο οποίος θα έχει καταστροφικά αποτελέσματα για τον υπολογιστή γιατί η εικονική μηχανή θα τον ανιχνεύσει και δε θα επιτρέψει να εκτελεστεί. Από την άλλη μεριά ούτε ο χρήστης μπορεί να κατεβάσει «κακό» κώδικα από το δίκτυο και να τον εκτελέσει. Αυτό είναι ιδιαίτερα χρήσιμο για μεγάλα κατανεμημένα συστήματα όπου πολλοί χρήστες χρησιμοποιούν το ίδιο πρόγραμμα συγχρόνως.

Ο συλλέκτης απορριμμάτων (Garbage Collector)

Ακόμα μία ιδέα που βρίσκεται πίσω από τη *Java* είναι η ύπαρξη του συλλέκτη απορριμμάτων (*Garbage Collector*). Συλλογή απορριμμάτων είναι μία κοινή ονομασία που χρησιμοποιείται στον τομέα της πληροφορικής για να δηλώσει την ελευθέρωση τμημάτων μνήμης από δεδομένα που δε χρειάζονται και δε χρησιμοποιούνται άλλο. Αυτή η απελευθέρωση μνήμης στη Java είναι αυτόματη και γίνεται μέσω του συλλέκτη απορριμμάτων. Υπεύθυνη για αυτό είναι και πάλι η εικονική μηχανή η οποία μόλις «καταλάβει» ότι ο σωρός (heap) της μνήμης (στη Java η συντριπτική πλειοψηφία των αντικειμένων αποθηκεύονται στο σωρό σε αντίθεση με τη C++ όπου αποθηκεύονται κυρίως στη στοίβα) κοντεύει να γεμίσει ενεργοποιεί το συλλέκτη απορριμμάτων. Έτσι ο προγραμματιστής δε χρειάζεται να ανησυχεί για το πότε και αν θα ελευθερώσει ένα συγκεκριμένο τμήμα της μνήμης, ούτε και για σφάλματα δεικτών. Αυτό είναι ιδιαίτερα σημαντικό γιατί είναι κοινά τα σφάλματα προγραμμάτων που οφείλονται σε λανθασμένο χειρισμό της μνήμης.

Επιδόσεις

Παρόλο που η εικονική μηχανή προσφέρει όλα αυτά (και όχι μόνο) τα πλεονεκτήματα, η *Java* αρχικά ήταν πιο αργή σε σχέση με άλλες προγραμματιστικές γλώσσες υψηλού επιπέδου (high-level) όπως η C και η C++. Εμπειρικές μετρήσεις στο παρελθόν είχαν δείξει ότι η C++ μπορούσε να είναι αρκετές φορές γρηγορότερη

από την Java. Ωστόσο γίνονται προσπάθειες από τη Sun για τη βελτιστοποίηση της εικονικής μηχανής, ενώ υπάρχουν και άλλες υλοποιήσεις της εικονικής μηχανής από διάφορες εταιρίες (όπως της IBM), οι οποίες μπορεί σε κάποια σημεία να προσφέρουν καλύτερα και σε κάποια άλλα χειρότερα αποτελέσματα. Επιπλέον με την καθιέρωση των μεταγλωττιστών JIT (Just In Time), οι οποίοι μετατρέπουν τον κώδικα byte απευθείας σε γλώσσα μηχανής, η διαφορά ταχύτητας από τη C++ έχει μικρύνει κατά πολύ.

Οι τελευταίες εκδόσεις του javac με τη χρήση της τεχνολογίας Hot Spot έχουν καταφέρει αξιόλογες επιδόσεις που πλησιάζουν ή και ξεπερνούν σε μερικές περιπτώσεις τον εγγενή κώδικα.

Android

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

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

Χαρακτηριστικά

Τωρινά χαρακτηριστικά και λειτουργίες:

Λειτουργίες Οθόνης	Η πλατφόρμα είναι προσαρμόσιμη σε μεγαλύτερη ανάλυση (VGA), διδιάστατες ψηφιακές γραφικές βιβλιοθήκες, τρισδιάστατα γραφικά βασισμένα στην OpenGL ES 1.0 έκδοση χαρακτηριστικών, καθώς και παραδοσιακές απεικονίσεις οθόνης "έξυπνων" συσκευών κινητής τηλεφωνίας.

Αποθήκευση Δεδομένων	Χρήση βάσης δεδομένων SQLite για τις ανάγκες αποθήκευσης
Συνδεσιμότητα	Το Android υποστηρίζει τεχνολογίες συνδεσιμότητας συμπεριλαμβανομένου GSM/EDGE, CDMA, EV-DO, UMTS, Bluetooth, και Wi-Fi.
Αποστολή μηνυμάτων	SMS και MMS είναι οι διαθέσιμοι τρόποι ανταλλαγής μηνυμάτων.
Περιήγηση στον Ιστό	Για την περιήγηση στον ιστό το Android διαθέτει φυλλομετρητή βασισμένο στην ανοιχτή τεχνολογία WebKit.
Υποστήριξη Java	Λογισμικό γραμμένο στην Java είναι δυνατόν να μεταγλωττιστεί και να εκτελεστεί στην εικονική μηχανή Dalvik, η οποία αποτελεί εξειδικευμένη υλοποίηση εικονικής μηχανής, σχεδιασμένης για χρήση σε φορητές συσκευές, παρόλο που δεν είναι πρότυπη εικονική μηχανή Java.
Υποστήριξη Πολυμέσων	Το λειτουργικό Android υποστηρίζει τις ακόλουθα μορφές ήχου, στατικής και κινούμενης εικόνας: H.263, H.264 (σε 3GP ή MP4container), MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, MP3, MIDI, OGG Vorbis, WAV, JPEG, PNG, GIF, BMP.[8]
Επιπλέον υποστήριξη υλικού	Το λειτουργικό Android μπορεί να συνεργαστεί με κάμερες στατικής ή κινούμενης εικόνας, οθόνες αφής, GPS, αισθητήρες επιτάχυνσης, μαγνητόμετρα, δισδιάστατους καθώς και τρισδιάστατους επιταχυντές γραφικών.
Περιβάλλον Ανάπτυξης Λογισμικού	Περιλαμβάνει ένας προσομοιωτή συσκευής, εργαλεία για διόρθωση σφαλμάτων, μνήμη και εργαλεία ανάλυσης της απόδοσης του εκτελέσιμου λογισμικού καθώς και ένα επιπρόσθετο για το Eclipse IDE.
Αγορά και Εγκατάσταση	Παρόμοια με το App Store του iPhone OS, το Android Market είναι ένας κατάλογος εφαρμογών που μπορούν να μεταφορτωθούν και εγκατασταθούν στην συσκευή άμεσα μέσω ασύρματων καναλιών,

Εφαρμογών	χωρίς την χρήση υπολογιστή. Αρχικά μόνο δωρεάν εφαρμογές ήταν δυνατόν να εγκατασταθούν. Εφαρμογές επί πληρωμή ήταν μετέπειτα διαθέσιμες στο Android Market στις ΗΠΑ ύστερα από τις 19 Φεβρουαρίου 2009.[9]
Οθόνη Αφής Πολλαπλών Σημείων	Το λειτουργικό Android είχε εξ ορισμού υποστήριξη για οθόνες πολλαπλών σημείων αλλά η δυνατότητα αυτή έχει κλειδωθεί σε επίπεδο πυρήνα (πιθανόν για αποφυγή παραβιάσεων των πατεντών λογισμικού της Apple στις τεχνολογίες οθονών αφής[10]). Κυκλοφορεί μια ανεπίσημη τροποποίηση (mod) που έχει αναπτυχθεί για να υποστηρίζει πολλαπλή επαφή (multi-touch), αλλά απαιτεί δικαιώματα πρόσβασης υπερχρήστη (superuser) στη συσκευή για να γραφεί στη μνήμη flash ένας πυρήνας που να μην είναι υπογεγραμμένος (unsigned kernel).[11]

Ιστορικό Ενημερώσεων

Παρόλο που το Android είναι ένα προϊόν ελεύθερου λογισμικού, ένα κομμάτι της ανάπτυξης του λογισμικού συνεχίζεται σε ιδιωτικό παρακλάδι. Για να έρθει αυτό το λογισμικό σε κοινή θέαση δημιουργήθηκε ένα παρακλάδι του μόνο ανάγνωσης, εν ονόματι "Cupcake". Το Cupcake συνήθως συγχέεται με τον τίτλο μιας ενημέρωσης, σε αντίθεση με όσα δηλώνει η ίδια η Google στην ιστοσελίδα ανάπτυξης του Android: "το Cupcake αποτελεί ακόμη ένα έργο σε εξέλιξη, όχι μια επίσημη έκδοση." Αξιοσημείωτες αλλαγές στο λειτουργικό Android θα παρουσιαστούν στο cupcake και περιλαμβάνουν αλλαγές στο σύστημα διαχείρισης των μεταφορτώσεων (download manager), το framework, Bluetooth, το λογισμικό συστήματος, το ραδιόφωνο και το σύστημα τηλεφωνίας, εργαλεία προγραμματισμού, το κυρίως σύστημα και διάφορες εφαρμογές, καθώς και πληθώρες διορθώσεις σφαλμάτων.

Στις 30 Απριλίου 2009, κυκλοφόρησε η επίσημη ενημέρωση έκδοσης 1.5 για το Android. Αποτελείται από πολλά νέα χαρακτηριστικά και βελτιώσεις στο γραφικό περιβάλλον:

- Δυνατότητα καταγραφής κινούμενης εικόνας με την χρήση της αντίστοιχης λειτουργίας του τηλεφώνου
- Μεταφόρτωση αρχείων βίντεο στο YouTube και εικόνων στο Picasa κατευθείαν από το τηλέφωνο
- Επανασχεδιασμένο λογισμικό πληκτρολογίου με λειτουργία αυτόματης συμπλήρωσης κειμένου
- Δυνατότητα αυτόματης σύνδεσης ασύρματης συσκευής ακουστικού Bluetooth εφόσον εντοπιστεί σε μια συγκεκριμένη απόσταση
- Νέα widgets και φάκελοι που μπορούν να τοποθετηθούν στην επιφάνεια εργασίας
- Εφέ αλλαγής οθονών και μενού
- Διευρυμένη λειτουργία αντιγραφής/επικόλλησης για να περιλαμβάνει δικτυακές διευθύνσεις

Έκδοση	Κωδική ονομασία	Ημερομηνία	API level	Διανομή (4 Μαρτίου 2013)
4.2.x	*Jelly Bean*	13 Νοεμβρίου 2012	17	1.6%
4.1.x	*Jelly Bean*	9 Ιουλίου 2012	16	14.9%
4.0.x	*Ice Cream Sandwich*	16 Δεκεμβρίου 2011	15	28.6%
3.2	*Honeycomb*	15 Ιουλίου 2011	13	0.9%
3.1	*Honeycomb*	10 Μαΐου 2011	12	0.3%
2.3.3-2.3.7	*Gingerbread*	9 Φεβρουαρίου 2011	10	44%
2.3-2.3.2	*Gingerbread*	6 Δεκεμβρίου 2010	9	0.2%
2.2	*Froyo*	20 Μαΐου 2010	8	7.6%
2.0-2.1	*Eclair*	26 Οκτωβρίου 2009	7	1.9%
1.6	*Donut*	15 Σεπτεμβρίου 2009	4	0.2%

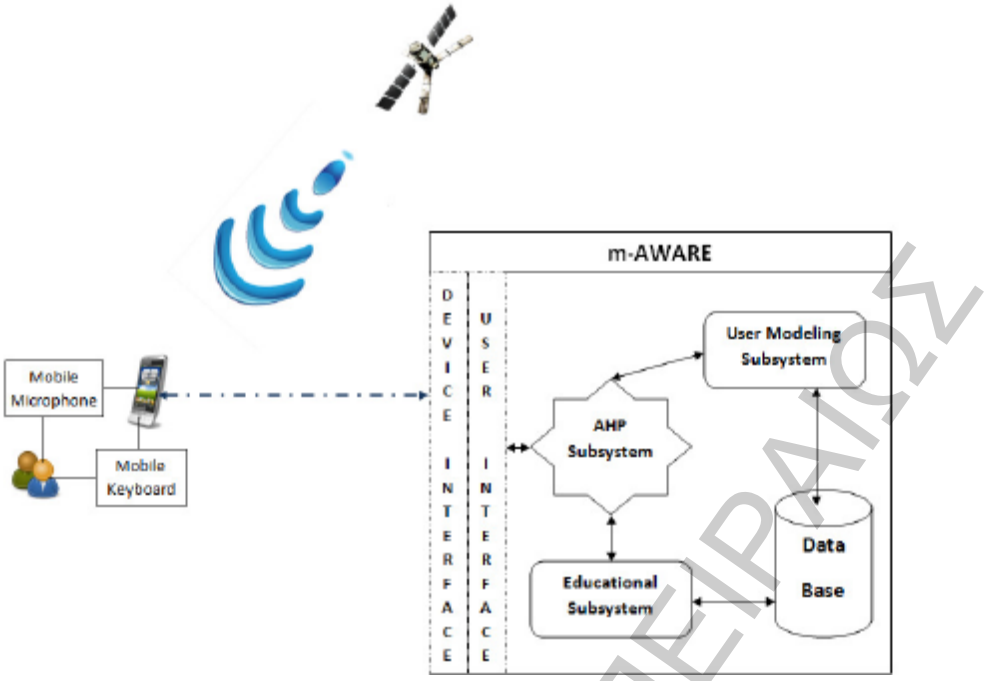
3 Παρουσίαση Εφαρμογών

LOCATION BASED USER MODELING IN ADAPTIVE MOBILE LEARNING FOR ENVIRONMENTAL AWARENESS

Alepis, E., Virvou, M. & **Kabassi, K.** (2011). Location based user modelling in adaptive mobile learning for environmental awareness.In M. Escalona Cuaresma, B. Shishkov, J. Cordeiro (Eds.): ICSOFT 2011 - Proceedings of the 6th International Conference on Software and Data Technologies, Volume 1, SciTePress 2011, ISBN 978-989-8425-76-8, pp. 214-217

Σύνοψη

Στην εισαγωγή του paper δίνεται ο γενικός όρος της Περιβαλλοντικής πληροφορικής αλλά και η ανάγκη γέννησής της μέσα από τα προβλήματα που παρουσιάζονται στο περιβάλλον από τις ανθρώπινες δραστηριότητες και παρεμβάσεις. Για την αντιμετώπιση αυτών των δυσχερών αποτελεσμάτων για το περιβάλλον δημιουργήθηκαν νέες εφαρμογές υπολογιστών που έχουν την δυνατότητα της λήψης αποφάσεων, καθώς και πληροφορίες για την περιβαλλοντική προστασία. Σε αυτή την εργασία παρουσιάζεται ένα νέο προσαρμοστικό σύστημα κινητής μάθησης που ενσωματώνει τους τέσσερεις πιο κάτω υπολογισμούς τεχνολογιών, όπως τα αντικειμενοστραφή δεδομένα αναπαράστασης, το Internet και το web wide world για την αναγνώριση λειτουργιών, την τηλεπισκόπηση και το GIS και τέλος τους προσαρμοστικούς πράκτορες. Το σύστημα που έχει κεντρικό χαρακτήρα σε αυτήν την εφαρμογή είναι το m-AWARE. Το AHP είναι υποσύστημα του κεντρικού συστήματος το οποίο είναι πολυκριτηριακό μοντέλο λήψης αποφάσεων, ουσιαστικά είναι ένας νοήμον μηχανισμός για τον καθορισμό των πληροφοριών. Η αρχιτεκτονική που στηρίχτηκε αυτό το σύστημα για τα διαθέσιμα δεδομένα είναι το αντικειμενοστραφές. Η περιβαλλοντική εκπαίδευση για τα κινητά έχει αναπτυχθεί περισσότερο στις μέρες μας χάρη στην πρόοδο της ασύρματης επικοινωνίας. Υπάρχει αυξημένο ενδιαφέρον για τα κινητά , τα κινητά δίκτυα αλλά και για τις location-based υπηρεσίες. Με τα κινητά διευκολύνουμε τον τρόπο ζωής μας γιατί μπορούμε να τα χρησιμοποιήσουμε δημιουργικά κατά τον ελεύθερο χρόνο μας, αφού έχουμε ενσωματώσει στην λειτουργία του κινητό την εκπαιδευτική τεχνολογία. Με βάση τα συμπερασμάτων των συγγραφέων η χρήση των κινητών οδηγών μπορούν να οδηγήσουν σε αύξηση της περιβαλλοντικής γνώσης και έτσι να αυξηθούν τα κίνητρα των παιδιών για την περιβαλλοντική εκπαίδευση, γιατί ένας σημαντικός παράγων για την επίτευξη αυτού του στόχου είναι το περιβάλλον εκπαίδευσης. Το σύστημα βασίζεται στην mobile technology και ενσωματώνει την θεωρία των Adaptive Hypermedia Systems, το Geographic Information Systems και το Global Positioning Systems. Επιπλέον το σύστημα ενσωματώνει ευφυής μεθόδους διδασκαλίας, όπως το m-AWARE.



Μας δίνονται πληροφορίες για το πώς ξεκίνησε το GIS, τον όρο του αλλά και την χρήση της θέσης ως βασική μεταβλητή δείκτη. Η geolocation μας πληροφορεί για θέση των χρηστών Η/Υ και συσκευών. Στην συγκεκριμένη εφαρμογή το m-AWARE ενσωματώνει το GIS ως υποσύστημα, έτσι έχουμε ένα προσαρμοσμένο εκπαιδευτικό περιβάλλον εργασίας του χρήστη που περιέχει περιβαλλοντικές πληροφορίες για την συγκεκριμένη θέση μέσα σε ένα προκαθορισμένο εύρος.

Το μοντέλο λήψεως αποφάσεων μέσα από την AHP είναι ένα από τις πιο δημοφιλής μεθόδους Multi Criteria Decision Making (MCDM). Η AHP μέθοδος αποτελείται από τα ακόλουθα βήματα:

- Developing a goal hierarchy (Ανάπτυξη μίας ιεραρχίας στόχου)
- Setting up a pair wise comparison matrix of criteria(Σύσταση ενός ζεύγους συνετού πίνακα σύγκρισης των κριτηρίων)
  Ο αλγόριθμος που χρησιμοποιείται παρουσιάζεται παρακάτω:
  1. Κανονικοποίηση κάθε στήλης διαιρώντας κάθε κύτταρο με το σύνολο της στήλης
  2. Συνοψίζοντας κάθε μία από τις σειρές σε μία νέα στήλη
  3. Κανονικοποίηση η νέα στήλη αντιπροσωπεύει το ιδιοδιάνυσμα, το οποίο περιέχει το βάρος του κάθε χαρακτηριστικού
- Ranking the relative importance between alternatives (Κατάταξη της σχετικής σημασίας μεταξύ των ζευγών των εναλλακτικών λύσεων)
- Checking consistency of the comparisons(έλεγχος συνέπειας των συγκρίσεων)
- Calculating AHP values(Υπολογισμός των τιμών AHP)

Το εκπαιδευτικό σύστημα αποτελείται από τρία υποσυστήματα: το user modeling subsystem, το educating subsystem και το AHP subsystem. Τόσο το πρώτο όσο και το δεύτερο υποσύστημα αποθηκεύονται στην βάση δεδομένων του κυρίου

συστήματος, ενώ το σύστημα λήψης αποφάσεων είναι υπεύθυνο για την προκύπτον διεπαφή δημιουργείται για τον κάθε χρήστη κατά την αλληλεπίδραση του με m-AWARE.

Επομένως το προκύπτον σύστημα είναι ένα εξελιγμένο σύστημα, το οποίο ενσωματώνει νέα ευρήματα στον τομέα των Mobile Computing, User Modeling, GIS και Περιβαλλοντικής Πληροφορικής, καθώς προσφέρει και την δυνατότητα για προώθηση και βελτίωση της περιβαλλοντικής ευαισθητοποίησης. Μια μελλοντική σκέψη για την εξέταση της χρησιμότητας είναι η αξιολόγηση του m-AWARE.

Mobile Applications for Industrial Environmental Protection

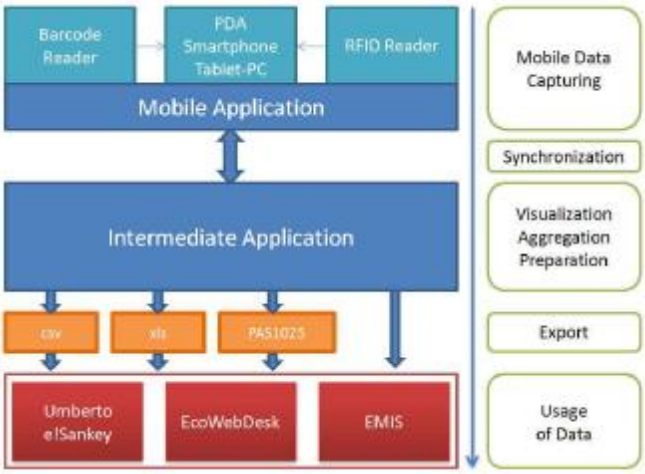
Tobias Ziep, Peter Krehahn, Volker Wohlgemuth
HTW Berlin, University of Applied Sciences,
Department of Engineering II, Industrial Environmental Informatics Unit,
Wilhelminenhofstraße 75A, D-12459 Berlin

Σύνοψη

Στην αρχή του paper μας αναφέρει πληροφορίες για την ανάπτυξη το λογισμικό των εφαρμογών των κινητών ώστε να βοηθηθούν οι μικρές και οι μεσαίες επιχειρήσεις να συγκεντρώνουν απαραίτητα δεδομένα για την προστασία του περιβάλλοντος. Οι επιχειρήσεις αυτές, η βιομηχανική περιβαλλοντική προστασία απαιτεί πολύ μεγάλα ποσά με αποτέλεσμα αυτές να μην μπορούν να ανταποκριθούν. Οι δυσκολίες δεν έγκειται μόνο στο οικονομικό σκέλος αλλά και νομικές υποχρεώσεις υποβολής εκθέσεων που θα πρέπει να πληρούν ώστε να αποθηκεύουν με σωστό τρόπο τα δεδομένα σύμφωνα με την εσωτερική διαδικασία. Αυτά τα δεδομένα που χρησιμοποιούνται από τις SME επιχειρήσεις είναι από πένα και χαρτί . Η κυβέρνηση της Γερμανίας χρηματοδότησε το πρόγραμμα MOEBIUS, έτσι ανέπτυξαν λογισμικό διαθέσιμο και κινητά για να συλλέγουν τα δεδομένα . Το λογισμικό έχει σχεδιαστεί έτσι ώστε να είναι εύκολο στις αλλαγές για να μπορεί να προσαρμοστεί ανάλογα στις απαιτήσεις των επιχειρήσεων ώστε να ταιριάζουν στις ειδικές ανάγκες τους.

Οι υποστηριζόμενες πτυχές της βιομηχανικής περιβαλλοντικής προστασίας για τις SME επιχειρήσεις είναι το Waste, Material/Material flow, Legal Compliance και Energy and Water. Το waste (απόβλητα) αφορά για την συλλογή δεδομένων για το πόσα και τα είδη των αποβλήτων στις εταιρίες. Το Material/Material flow (Υλικό / Υλικό ροής) αναλύει απλές μεθόδους διαχείρισης της ροής υλικών, όπως Material Stream Mapping (MSM) and Value Stream Mapping (VSM) στην πρώτη θέση. Η Legal Compliance (Νομική Συμμόρφωση) υποστηρίζει τι συμμόρφωση με βάση τις νομικές κατευθυντήριες γραμμές με τι συλλογή δεδομένων για την τεκμηρίωση και την υποβολή εκθέσεων σκοπού. Το *Energy and Water* (Ενέργεια και το Νερό) εστιάζεται στο εννοιολογικό σχεδιασμό και την ανάπτυξη λύσεων για τη συλλογή των δεδομένων της ενέργειας (πεπιεσμένου αέρα, ηλεκτρικό ρεύμα, θέρμανση, φυσικό αέριο) και υδρόμετρα.

Η βασική αρχιτεκτονική του λογισμικού στηρίζεται στο λογισμικό με τις κινητές συσκευές και σε μια ενδιάμεση εφαρμογή που είναι ένας desktop υπολογιστής. Σκοπός της κινητής εφαρμογής είναι συλλογή δεδομένων από τεχνολογίες όπως η αναγνώριση από barcode, RFID και PDA smartphone Tablet, έτσι τα δεδομένα που συλλέγονται μεταφέρονται στην ενδιάμεση συσκευή. Στην εφαρμογή κινητών γίνεται η συλλογή των δεδομένων κάθε αφορά από άλλο άτομο. Ο τελικός χρήστης έχει ένα πλεονέκτημα χρησιμοποιώντας την συσκευή και αυτό είναι η αλλαγή της υφιστάμενης διαδικασίας. η συσκευή θα πρέπει να είναι απλά σχεδιασμένη ώστε να είναι ευκολόχρηστη. Η ενδιάμεση συσκευή με την κινητή συσκευή έχουν έναν αμφίδρομο συγχρονισμό μεταξύ των λογισμικών τους. Η PAS1025 είναι μια προδιαγραφή XML για τη μεταφορά δεδομένων μεταξύ του προγραμματισμού των επιχειρηματικών πόρων (ERP) λογισμικό και τα συστήματα περιβαλλοντικής διαχείρισης πληροφοριών (EMIS). Η ενδιάμεση εφαρμογή προσφέρει διαφορετικές όψεις για μεταβλητές εργασίες, επομένως χρειάζεται το λογισμικό να έχει υψηλή επεκτασιμότητα για μελλοντικές εξελίξεις στον τομέα της SME επιχείρησης. Μια πρόκληση για την αποθήκευση των δεδομένων είναι ή σε ένα στιγμιότυπο της ενδιάμεσης συσκευής ή σε μια βάση δεδομένων server, με το δεύτερο να έχει αβαντάζ στις μεγάλες επιχειρήσει.



Τα παραδείγματα που αναλύονται πιο κάτω αφορά λογισμικό που καλείται Meter. Το λογισμικό παρέχει την λειτουργικότητα για την καταγραφή των τιμών της κατανάλωσης ενέργειας και νερού από τις αντίστοιχες μετρήσεις. Το εν λόγω παράδειγμα αφορά μια μεγάλη επιχείρηση μετάλλων που έχει μπει στο πρόγραμμα, της οποίας οι διεργασίες αναλύονται για να προσδιορίσουν τις βέλτιστες πρακτικές. Το Meter καθορίζει ένα σχέδιο συλλογής δεδομένων, το οποίο αποτελείται από ένα σύνολο μέτρων και έναν υπάλληλο υπεύθυνο για την συλλογή των δεδομένων. Ο υπάλληλος εισέρχεται στο λογισμικό όπου υπάρχει μια λίστα με τις τρέχουσες εργασίες ελέγχου. Ενώ, τα μέτρα προσδιορίζονται με κάμερα μέσω ειδικών barcodes για την συλλογή δεδομένων και οι τιμές είναι εξαγώγιμες για χρήση σε μορφή xml. Ένα άλλο παράδειγμα είναι το κινητό θέσης στο τμήμα της νομικής συμμόρφωσης όπου οι επιχειρήσεις είναι υποχρεωμένες για την συντήρηση των εργασιών συντήρησης σε ψυκτικές συσκευές. Αυτό θα πρέπει να γίνεται με βάση τη κείμενη νομοθεσία της Γερμανίας, για συντήρηση και τη εποπτεία θα πρέπει να ενημερώνονται οι δικαστικές αρχές. Το λογισμικό που χρησιμοποιείται στο κινητό για την συντήρηση είναι ένα tablet-PC, έτσι επιτρέπει στους χρήστες να εισάγουν δεδομένα και δημιουργεί ερωτηματολόγια με βάση της πληροφορίες και ταχύτερους χρόνους αντίδρασης.

Η ανάπτυξη της εφαρμογής των κινητών θα πρέπει να είναι όσο το δυνατόν εύκολο στη χρήση του λογισμικού του κινητού με την κατάλληλη αλληλεπίδραση για απλούς χρήστες και να μεταθέσουν λειτουργικότητα που δεν είναι απαραίτητη στην ενδιάμεση εφαρμογή. Χρησιμοποιούμε μικρές συσκευές για συλλογή μεμονωμένων

τιμών, για καλύτερη ποιότητα δεδομένων χρησιμοποιούμε το RFID ή το barcode, σημαντικός παράγων είναι ο συγχρονισμός μεταξύ των συσκευών γιατί έτσι μόνο οι νέες τιμές εισάγονται και προετοιμάζονται για περεταίρω χρήση. Ωστόσο, υπάρχει πρόβλημα στα λειτουργικό σύστημα του κινητού και η διεπαφή χρήστη μπορεί να γίνει μόνο με Silverlight/WPF. Οι επιχειρήσεις όπως Microsoft ή Apple ενδιαφέρονται για πωλήσεις στην καταναλωτική αγορά και δεν τους ενδιαφέρει η ανάπτυξη των εξατομικευμένων επιχειρηματικών εφαρμογών. Σε αυτόν τον τομέα των εφαρμογών μόνο το Windows Embedded Compact 7 έχει δημιουργηθεί, ενώ το λογισμικό για το Android είναι ανοικτού κώδικα σε linux και ουσιαστικά χρειάζεται εκ νέου υλοποίηση.

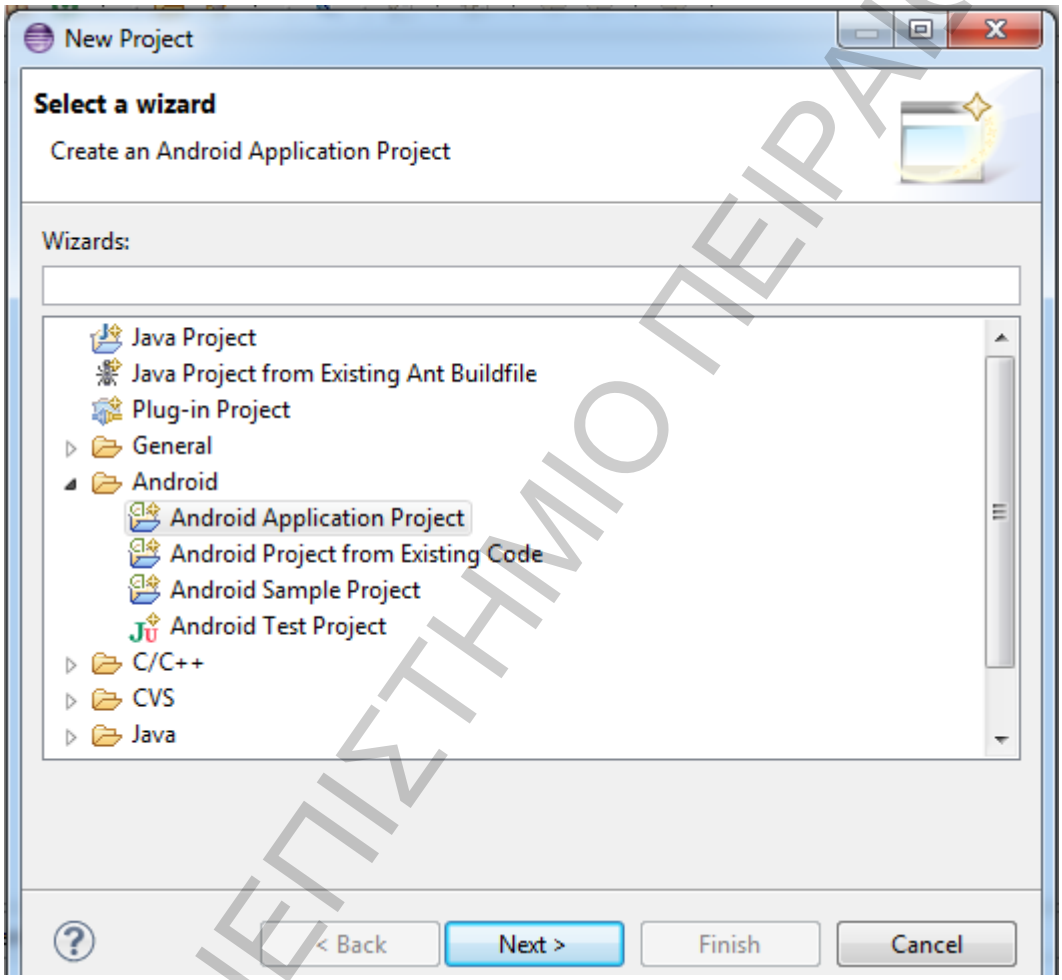
Ουσιαστικά, στο paper παρουσιάζεται η χρήση κινητών τηλεφώνων για βελτιστοποίηση και αυτοματοποίηση των διαδικασιών ώστε να διαχειρίζονται τα δεδομένα για την βιομηχανική περιβαλλοντική προστασία. Μέσω αυτών των δεδομένων οι εταιρίες θα μπορούν να επεκτείνουν ή τροποποιήσουν το λογισμικό αναλόγως των αναγκών τους. Σε δεύτερη φάση χρησιμοποιεί την ενδιάμεση εφαρμογή που είναι επεκτάσιμη εφαρμογή και χρησιμοποιεί το Empinia λογισμικό. Το επόμενο βήμα είναι ο περαιτέρω σχεδιασμός και εφαρμογή της ενδιάμεσης εφαρμογής, το οποίο είναι βασικό στοιχείο του MOEBIUS.

4 Παρουσίαση και Χρήση Εφαρμογής

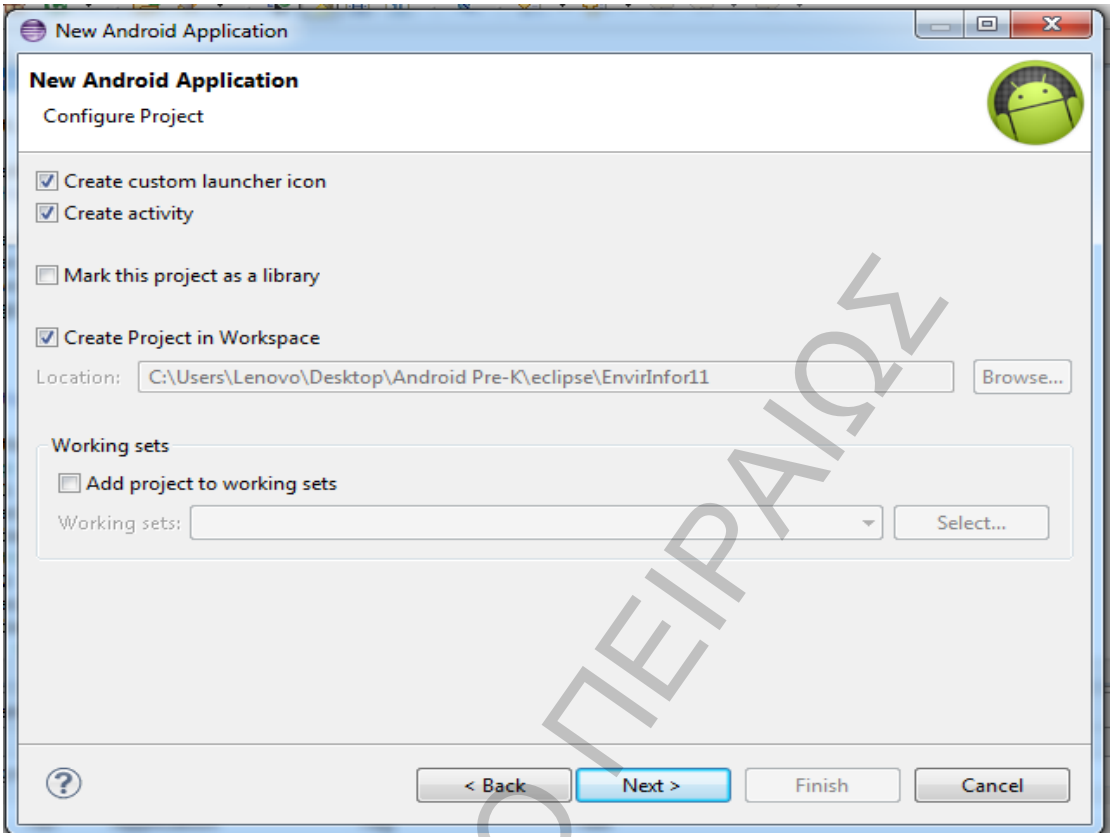
Δημιουργία Έργου Android

Ανοίγουμε το Eclipse και κάνουμε τα παρακάτω βήματα για να δημιουργήσουμε το έργο μας:

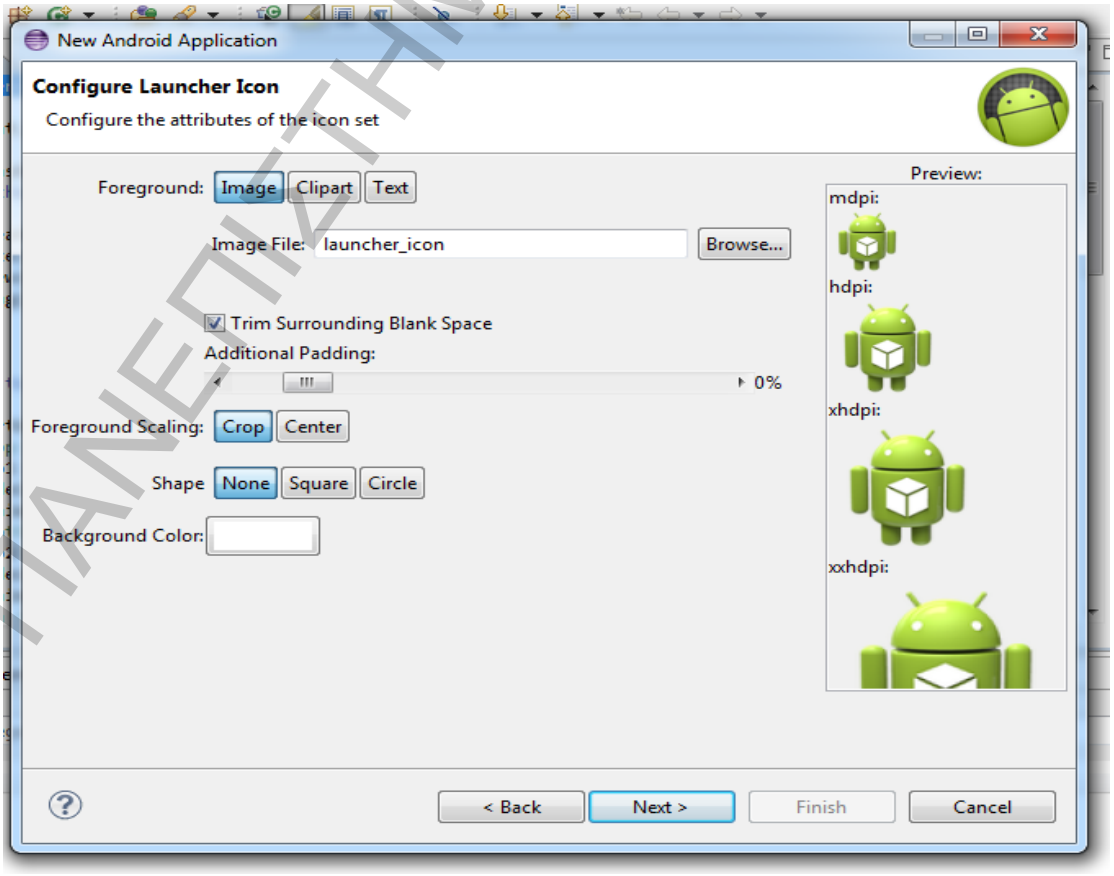
1. Επιλέγουμε File, New, Project κάνουμε κλικ πάνω στο project , εμφανίζεται το New Project και επιλέγουμε στο φάκελο Android την επιλογή Android Application Project πατώντας το Next.



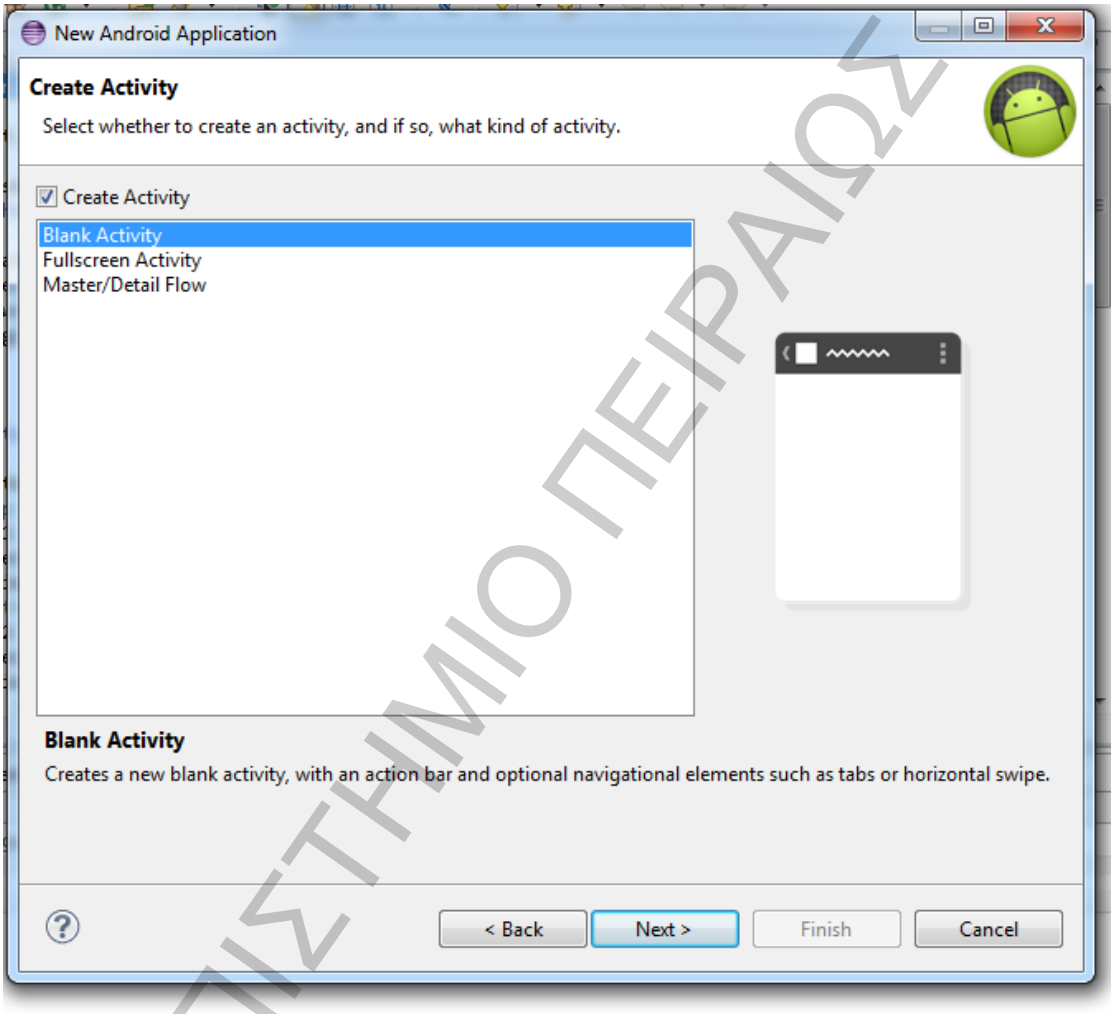
2. Μας οδηγεί στη φόρμα New Android Application, συμπληρώνουμε την φόρμα με τα στοιχεία της εφαρμογής μας που το όνομά της είναι EnvirInfor1, το package είναι com.example.envirinfor1, την έκδοση του Android την κάνουμε 4.3 , πατάμε Next.



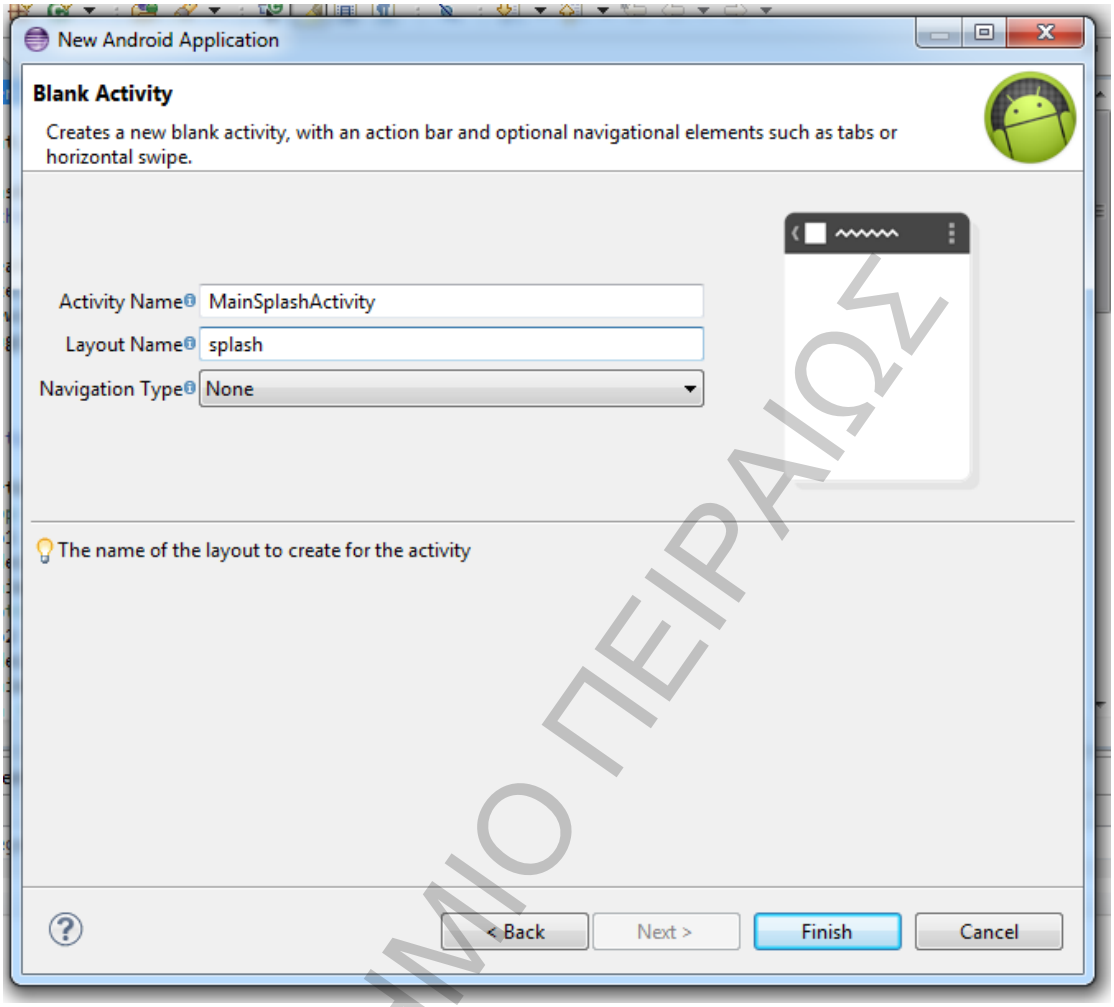
3. Επίσης πατάμε Next και οδηγούμαστε πιο κάτω.



4. Εδώ δίνουμε την διεύθυνση της εικόνας που έχουμε επιλέξει για λογότυπο στην εφαρμογή μας, στην προκειμένη περίπτωση είναι η images.png και μετά πατάμε Next



5. Επιλέγουμε Blank Activity και πατάμε Next.

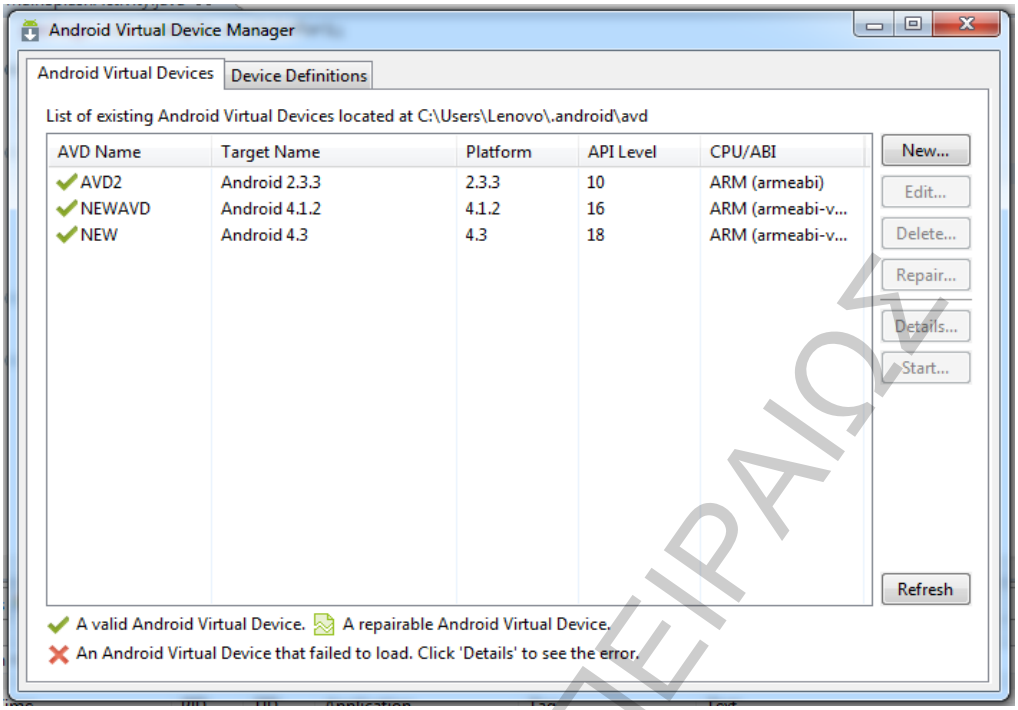


6. Στο Activity Name το κάνουμε MainSplashActivity ενώ το LayoutName σε Splash και πατάμε το finish ώστε να μπούμε στην εφαρμογή μας.

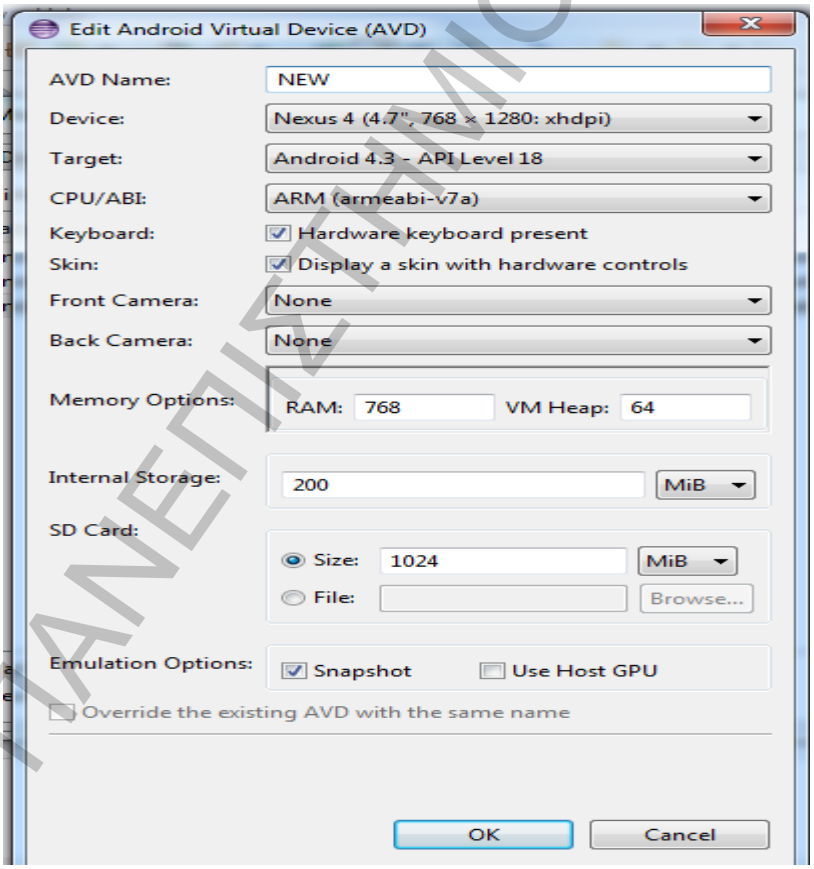
Διαχείριση Εικονικών Συσκευών Android

Για να μπορέσουμε να τρέξουμε μια εφαρμογή μέσα στον εξομοιωτή του Android, πρέπει να διαμορφώσουμε μια εικονική συσκευή του Android(AVD). Το προφίλ AVD περιγράφει τον τύπο της συσκευής που θέλουμε να προσομοιώσει ο εξομοιωτής, περιλαμβανομένου και ποια πλατφόρμα Android υποστηρίζει. Μπορούμε να καθορίσουμε διαφορετικά μεγέθη και αναλύσεις οθόνης και μπορούμε να καθορίσουμε, αν ο εξομοιωτής έχει κάρτα SD και, αν ναι, την χωρητικότητά της. Τα βήματα που ακολουθήσαμε για την δημιουργία του AVD της εφαρμογής μας:

1. Εκκίνηση του AVD Manager μέσα από το Eclipse, κάνοντας κλικ στο μικρό πράσινο εικονίδιο του Android στη γραμμή εργαλείων. Εμφανίζεται το παρακάτω εικονίδιο.



2. Κάνουμε κλικ στο κουμπί New για να δημιουργήσουμε μια νέα AVD, έτσι παρουσιάζεται η πιο κάτω μορφή.



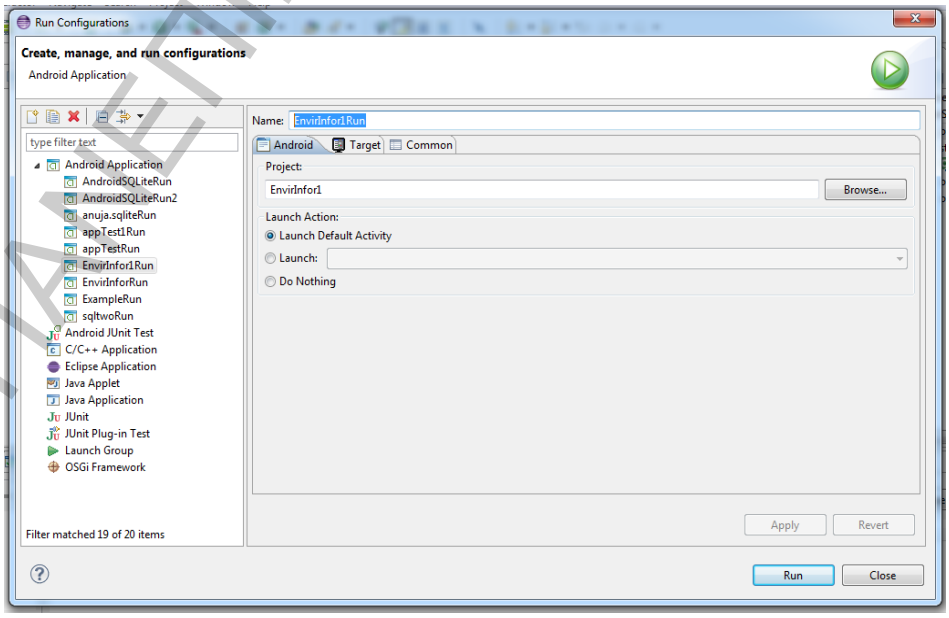
3. Επιλέγουμε ένα όνομα για το AVD και το ονομάζουμε NEW.

4. Στο Device είναι το είδος της εικονικής συσκευής που χρησιμοποιούμε και το όνομά της Nexus 4(4.7¨, 768*1280:xhdi)
5. Επιλέγουμε ένα στόχο δόμησης. Για να υποστηρίξουμε το Android 4.3, θα πρέπει να επιλέξουμε το στοιχείο δόμησης με όνομα Android 4.3- API Level 18.
6. Το CPU/ABI το θέτουμε ARM(armeabi-v7a)
7. Κάνουμε κλικ στο skin και το keyboard, όπου αυτές οι επιλογές ελέγχει τις διάφορες οπτικές εμφανίσεις του εξομοιωτή αλλά και το πληκτρολόγιο.
8. Το Memory Options διαθέτει την μνήμη RAM και VM Heap, όπου το πρώτο το δίνουμε τιμή 768, ενώ το δεύτερο σε 64.
9. Το Internal Storage δίνουμε τιμή 200 MiB
10. Τέλος, η SD card καταλαμβάνει χώρο στον σκληρό δίσκο, οπότε επιλέγουμε ένα λογικό μέγεθος που είναι 1024 MiB.

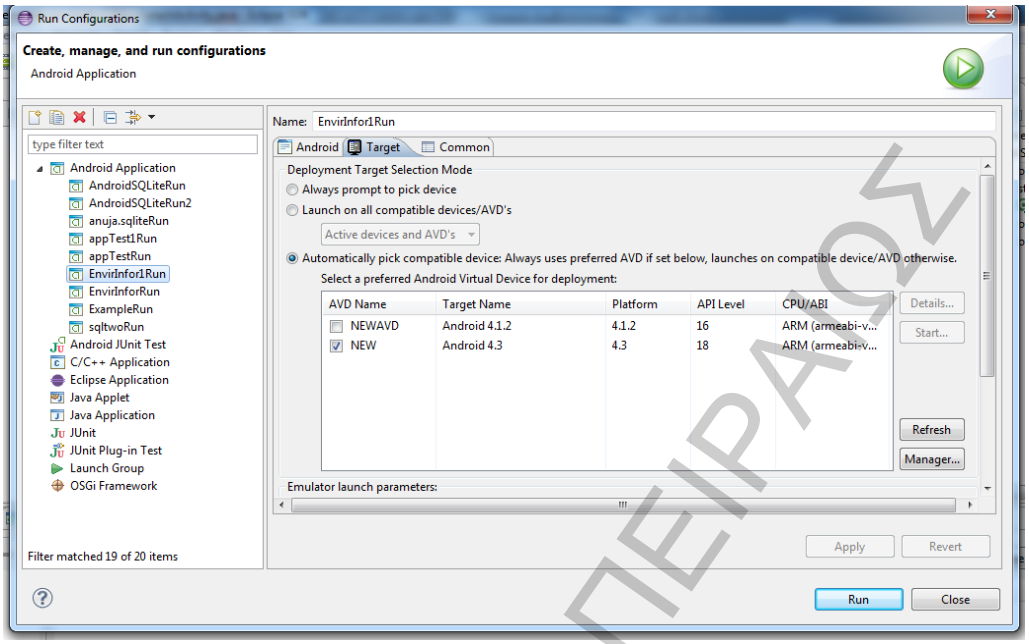
Δημιουργία Εκτέλεσης του Android

Για να εκκινήσουμε την εφαρμογή μας να δημιουργήσουμε μια διαμόρφωση Run για το έργο μας μέσα στο Eclipse. Κάνουμε το παρακάτω βήματα:

1. Μέσα στο Eclipse επιλέγουμε το Run και εν συνεχεία επιλέγουμε το Run Configuration.
2. Κάνουμε διπλό κλικ στο εικονίδιο Android Application για να δημιουργήσουμε μια νέα καταχώριση.
3. Επεξεργαζόμαστε την νέα καταχώριση που τώρα έχει όνομα New_configuration.
4. Αλλάζουμε το όνομα της διαμόρφωσης σε EnvirInfor1Run.
5. Θέτουμε το έργο κάνοντας κλικ στο κουμπί Browse και επιλέγοντας το έργο EnvirInfor1.



6. Στην καρτέλα Target , επιλέγουμε το πλαίσιο δίπλα στην AVD και είναι το NEW.

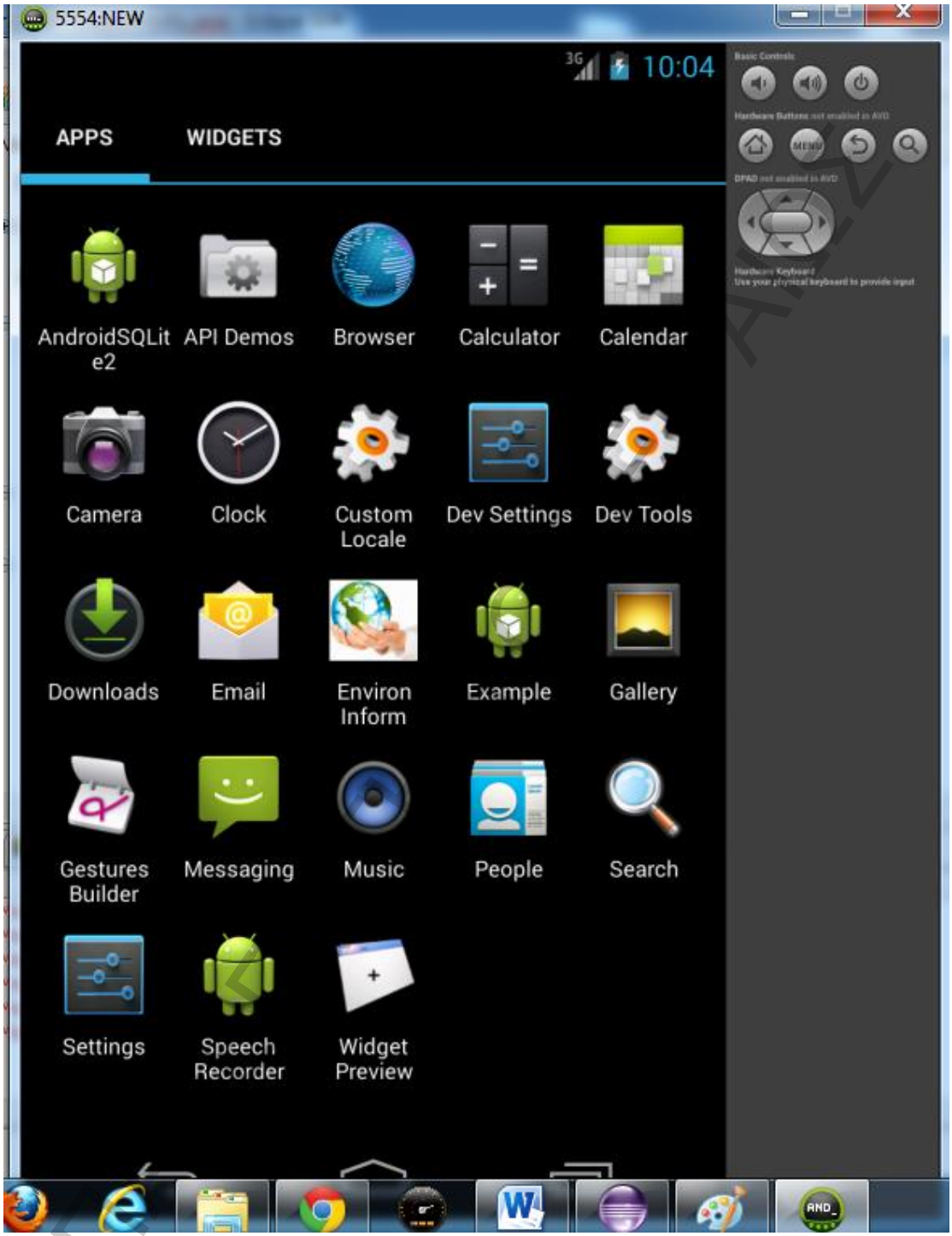


7. Εφαρμόζουμε τις αλλαγές κάνοντας κλικ στο κουμπί Apply και στη συνέχεια πατάμε Run.

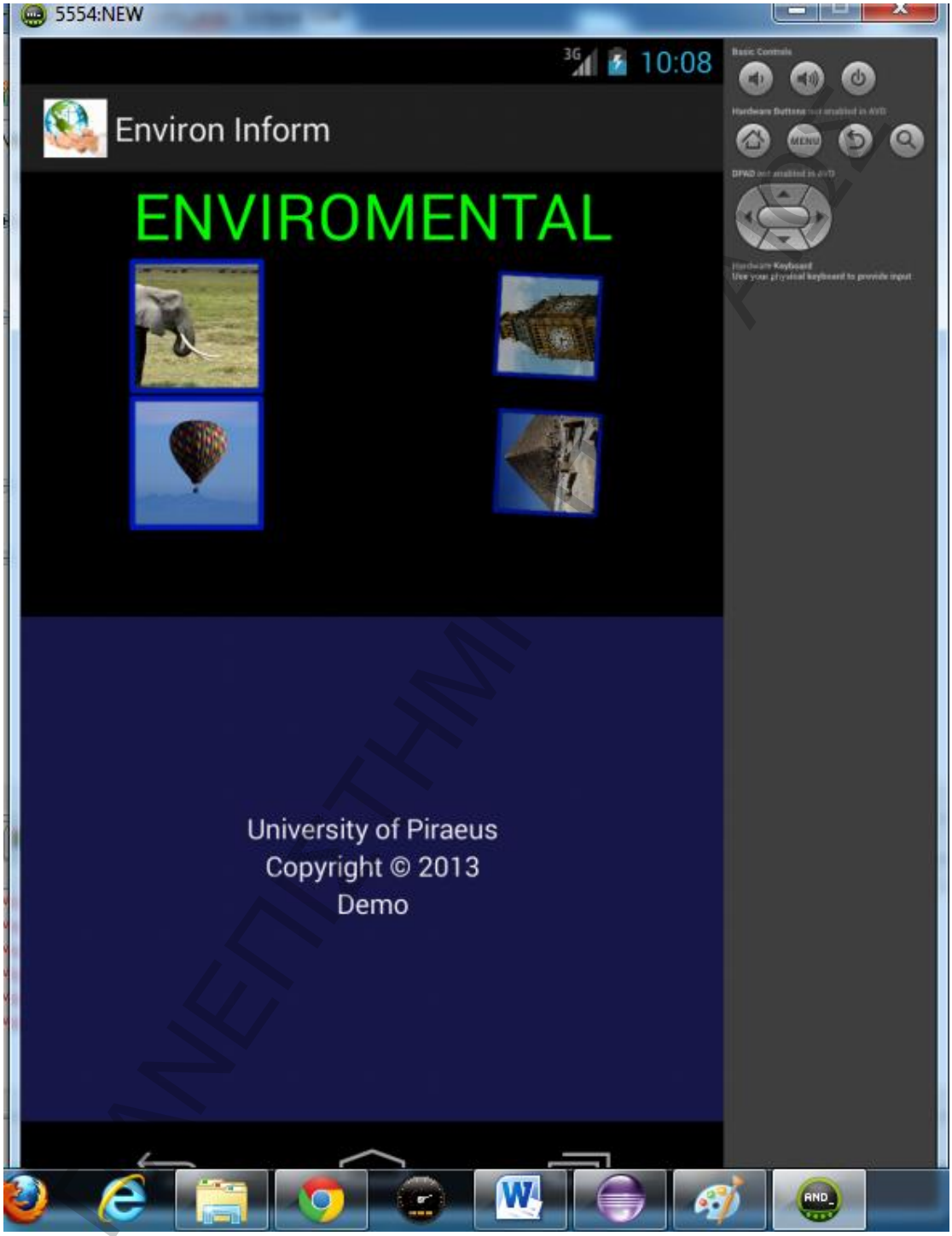
Προβολή εφαρμογής

Έχοντας τρέξει το πρόγραμμα του EnvirInfor1 στο eclipse εμφανίζεται στον emulator με το όνομα NEW και έχει την παρακάτω μορφή:

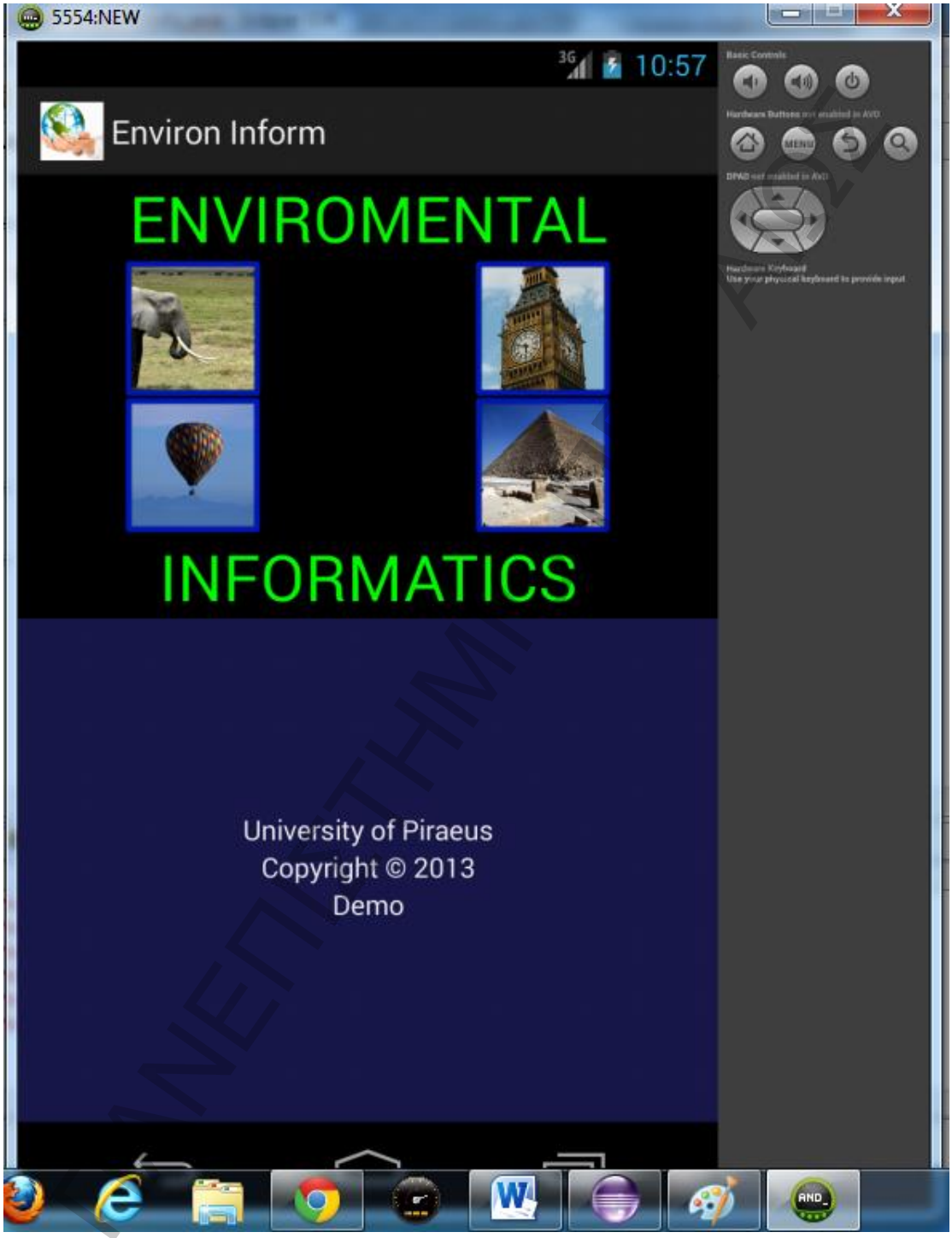
Μπαίνοντας στο κυρίως μενού του emulator συναντάμε την εφαρμογή με όνομα Environ Inform που έχουμε δημιουργήσει, με την ακόλουθη μορφή:



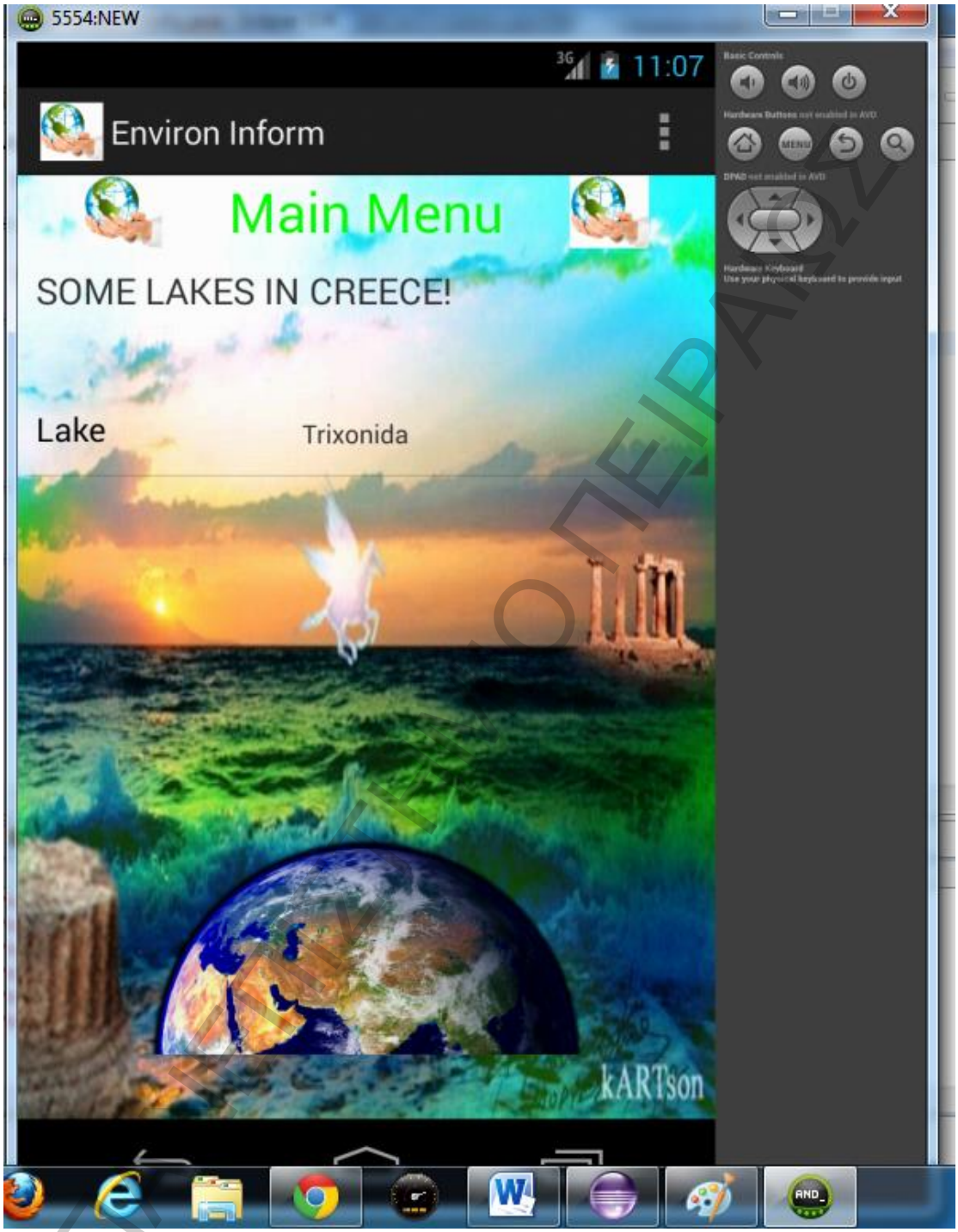
Στην συνέχεια κάνοντας κλικ πάνω στο εικονίδιο της εφαρμογής βλέπουνε να ξεκινάει το πρόγραμμα να τρέχει και σε πρώτη φάση να παίρνει την παρακάτω μορφή:



Στην πιο πάνω εικόνα φαίνεται η εφαρμογή όταν είναι σε κίνηση η οποία κρατάει συνολικά 5 sec και στην συνέχεια πηγαίνει στο κυρίως μενού, πριν όμως πάει παίρνει την τελική μορφή, σύμφωνα με το παρακάτω print screen.

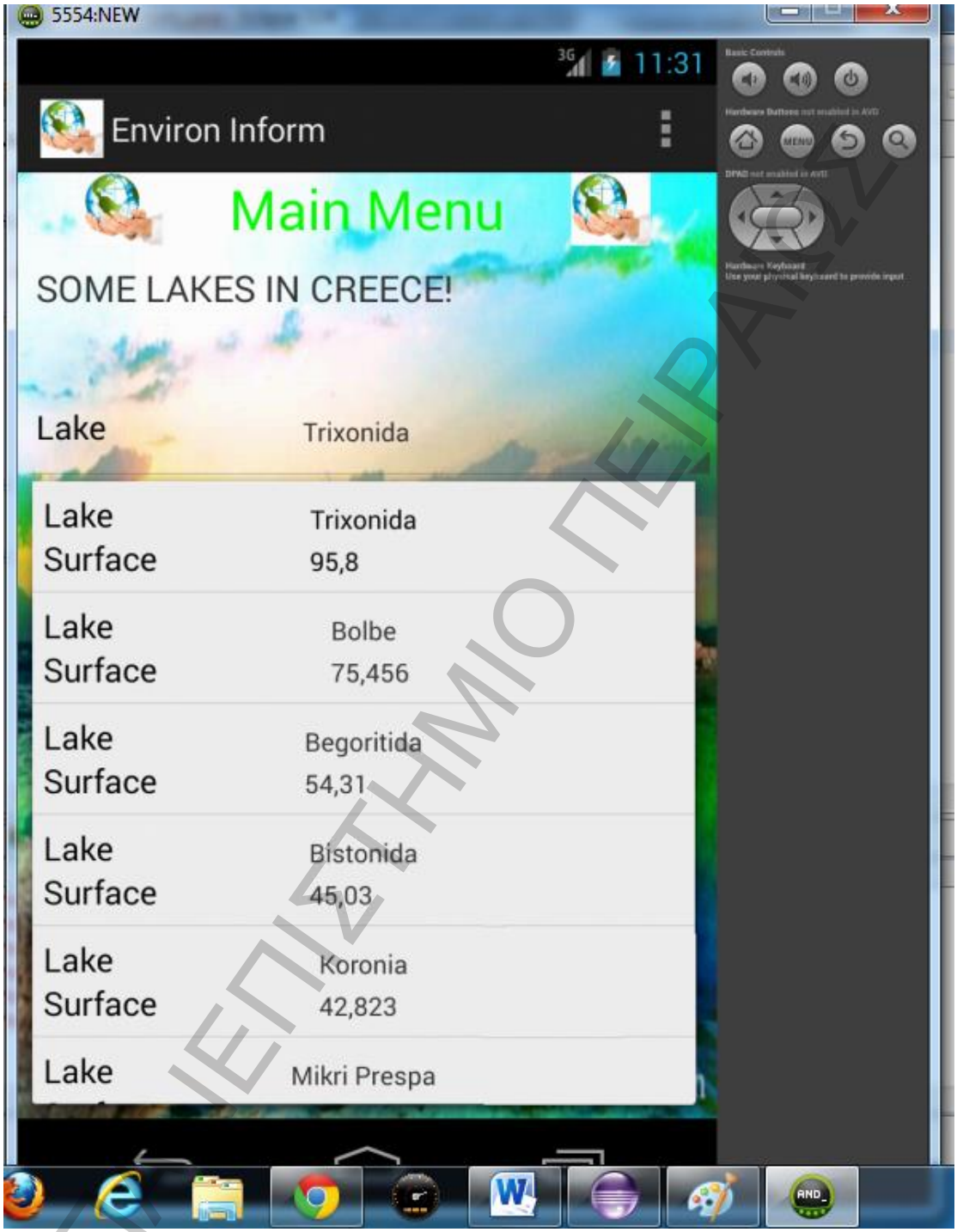


Όταν τελειώσει το πρώτο μέρος της οθόνης εισέρχεται στο κυρίως μενού και παίρνει την ακόλουθη μορφή:

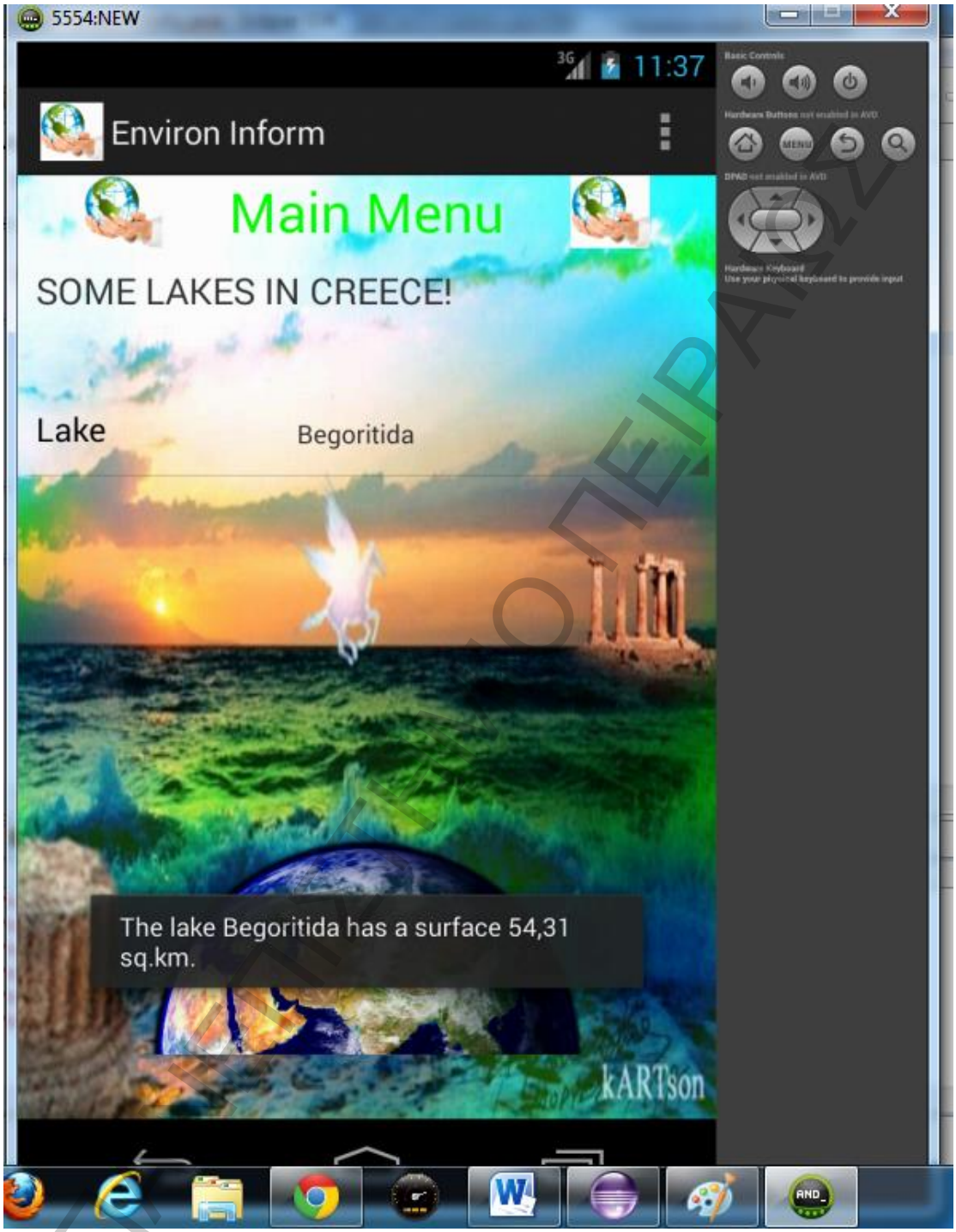


Στο συγκεκριμένο σημείο θα πρέπει να πούμε ότι η εφαρμογή μας παρουσιάζει τις μεγαλύτερες λίμνες τις Ελλάδας, κάνει αναφορά στο όνομα, της κάθε μιας αλλά και στο εμβαδό τους. Όταν κάνουμε κλικ πάνω στο spinner εμφανίζονται όλες οι λίμνες που είναι στο κατάλογο μέσα στην βάση δεδομένων. Επιλέγουμε την λίμνη της αρεσκείας μας και βλέπουμε σε μήνυμα ότι η λίμνη πχ Τριχωνίδα έχει εμβαδό πχ 95,8

τ.χλμ. Έτσι μόλις φύγει το μήνυμα από την οθόνη της εφαρμογής το spinner παίρνει ως αρχική τιμή την λίμνη που επιλέξαμε.



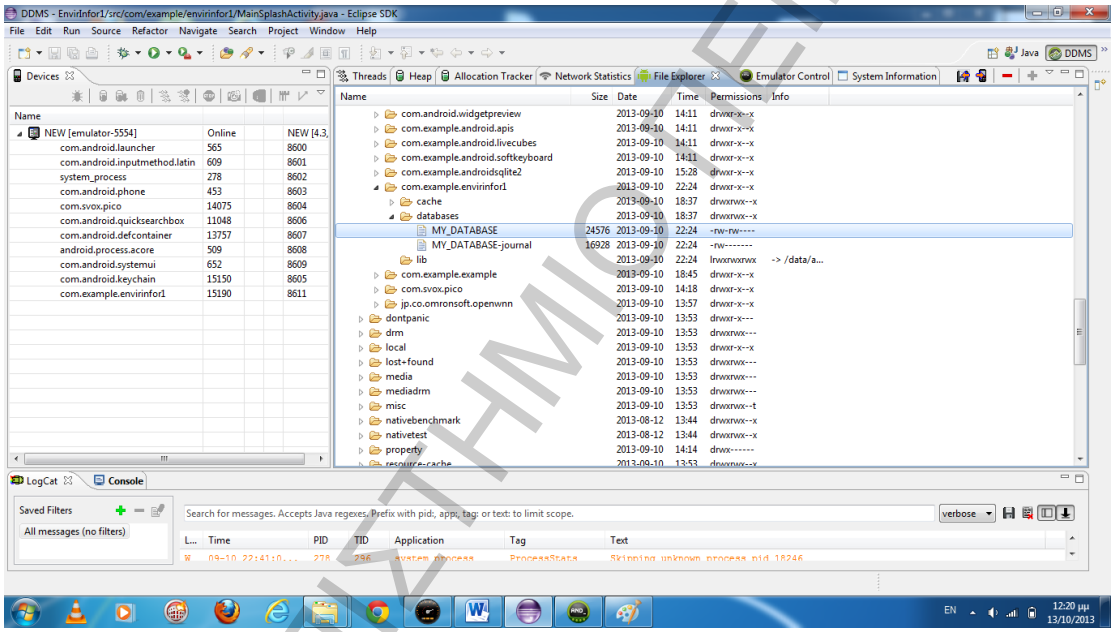
Αφού κάνουμε κλικ πάνω στην λίμνη που επιθυμούμε βγαίνει το αντίστοιχο μήνυμα και γίνεται αντικατάσταση της λίμνης.



Δόμηση της βάσης δεδομένων με sqlitebrowser

Στη γραμμή εργαλείων του Eclipse οδηγούμε τον κέρσορα πάνω στο κουμπί του DDMS και αλλάζουμε περιβάλλον. Η υπηρεσία Dalvik Debug Monitor Service(DDMS) είναι ένα βοήθημα αποσφαλμάτωσης, που ενοποιείται μέσα στο Eclipse μέσω μιας ειδικής όψης του Eclipse. Η όψη DDMS παρέχει αρκετά χρήσιμα για αλληλεπίδραση με εξομοιωτές και συσκευές και για αποσφαλμάτωση εφαρμογών. Τα χαρακτηριστικά του DDMS διαιρούνται γενικά σε πέντε λειτουργικές περιοχές:

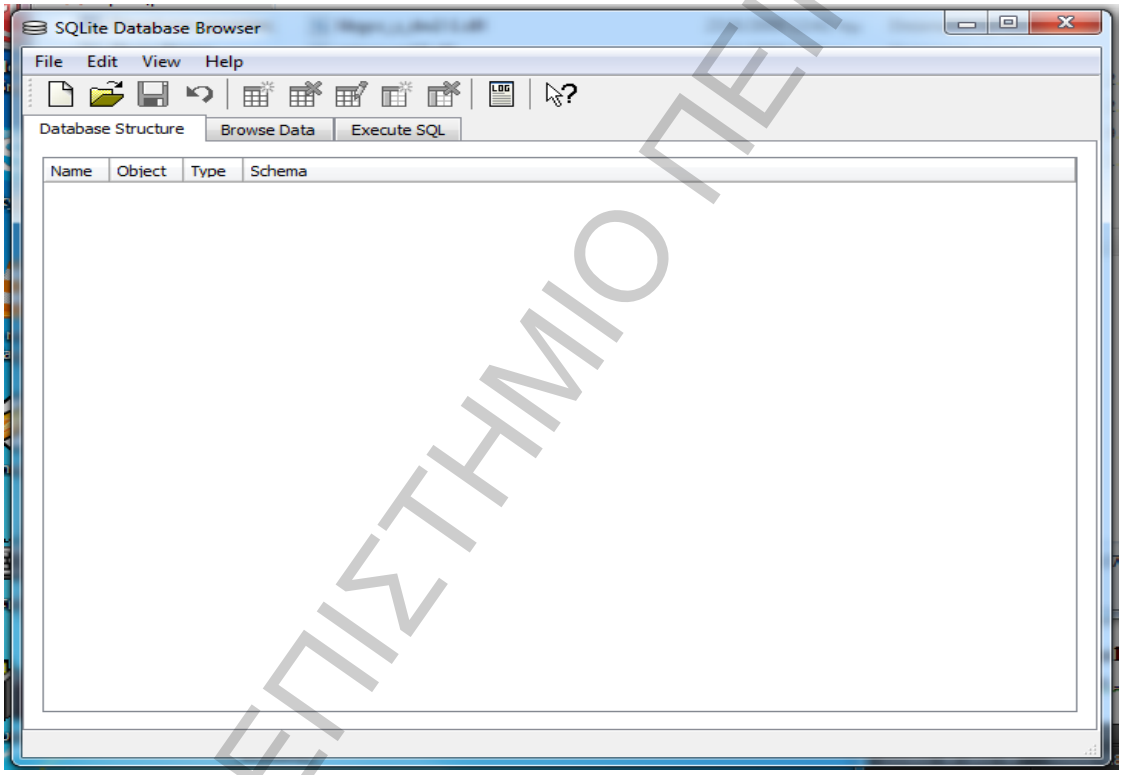
- Διαχείριση εργασιών
- Διαχείριση αρχείων
- Αλληλεπίδραση με εξομοιωτή
- Εισοδοχή
- Συλλήψεις οθόνης



Αφού έχουμε εισέλθει στο περιβάλλον του DDMS, κάνουμε κλικ πάνω στο File Explorer. Στη συνέχεια επιλέγουμε το φάκελο data, τον ίδιο φάκελο επιλέγουμε και πιο κάτω. Μετέπειτα, κλικάρουμε το φάκελο com.exampe.envirinfor1 και επιλέγοντας το φάκελο databases κάνουμε κλικ στο MY_DATABASE.

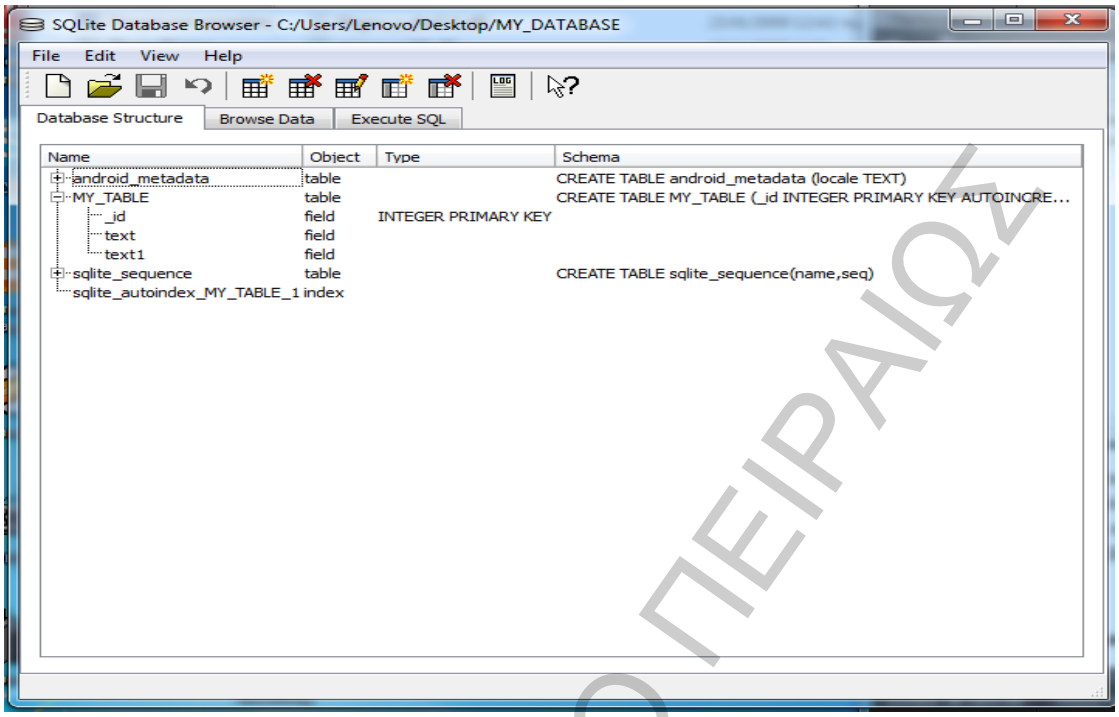
Για να μπορέσουμε να αποθηκεύσουμε το αρχείο βάσης MY_DATABASE, επιλέγουμε το  για να αποθηκεύσουμε το αρχείο σε ένα σημείο του υπολογιστή, ώστε να το χρησιμοποιήσουμε με ευκολία. Το αποθηκεύουμε στην επιφάνεια εργασίας με το ίδιο όνομα και το αρχείο παίρνει την παρακάτω μορφή:

Αφού μπούμε στο πρόγραμμα του SQLite Database Browser 2.0b1, παίρνει την πιο κάτω μορφή

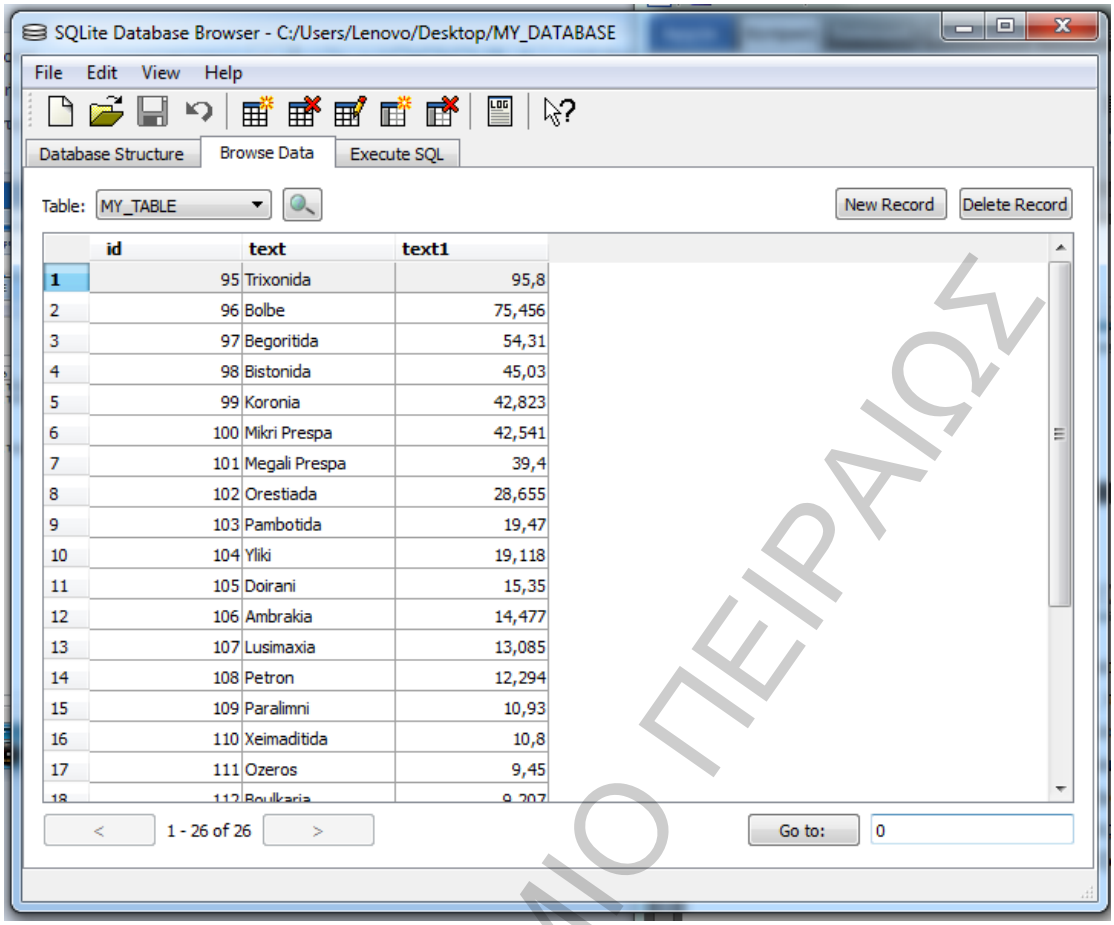


Έχοντας μπει στο επιθυμητό πρόγραμμα επιλέγουμε File , Open Database, έτσι μας δίνει την δυνατότητα να επιλέξουμε το αρχείο της αρεσκείας μας, που εμείς με την

σειρά μας το είχαμε οδηγήσει στην Επιφάνεια Εργασίας. Έχοντας επιλέξει το MY_DATABASE το πρόγραμμα παίρνει τη παρακάτω μορφή



Η πιο πάνω εικόνα είναι ένα print screen από την επιλεγόμενη καρτέλα Database Structure. Το Database Structure μας πληροφορεί για το όνομα του πίνακα που είναι MY_TABLE, καθώς και για τις στήλες. Το _id είναι το κύριο κλειδί του πίνακα και είναι integer, η άλλη στήλη αποθηκεύει τα ονόματα των λιμνών και η τελευταία τα τ. χλμ του εμβαδού της κάθε λίμνης. Όμως, υπάρχει δίπλα μια άλλη καρτέλα η Browse Data, η οποία μας πληροφορεί για όλες τις εγγραφές που διαθέτει η βάση μας. Για να δούμε τις εγγραφές θα πρέπει να επιλέξουμε MY_TABLE.

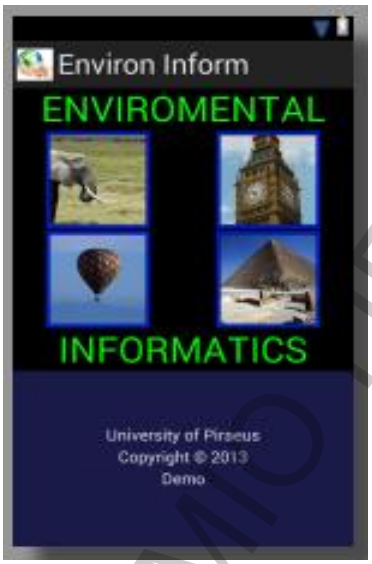


5 Δόμηση Εφαρμογής Android

Σχεδίαση της EnvirInform εφαρμογής στο Android

Η σχεδίαση της εφαρμογής EnvirInform διαθέτει τέσσερις οθόνες:

- Splash.xml- Λειτουργεί ως μια οθόνη εκκίνησης και περιλαμβάνει το λογότυπο , καθώς και την έκδοση της εφαρμογής.



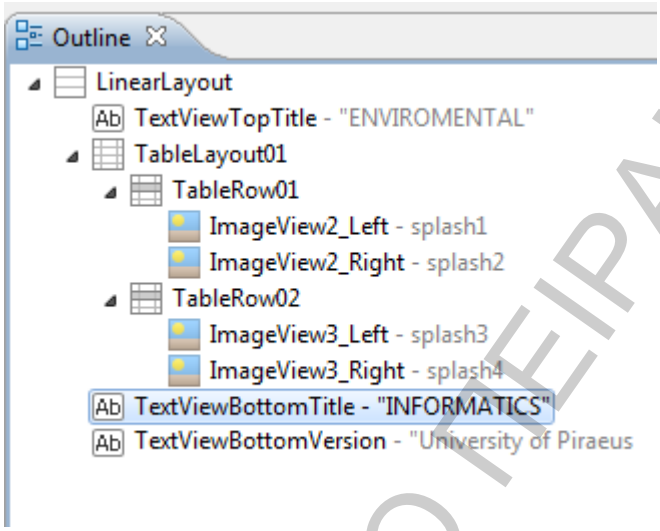
Επίσης, εμπεριέχεται και ο κώδικας του παραπάνω xml αρχείου.

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@android:color/black"
    android:orientation="vertical">

    <TextView android:layout_height="wrap_content"
              android:layout_width="match_parent"
              android:textSize="@dimen/logo_size"
              android:gravity="top|center"
              android:layout_gravity="center_vertical|center_horizontal"
              android:textColor="@color/logo_color"
              android:text="@string/app_logo_top"
              android:id="@+id/TextViewTopTitle"/>

     <TableLayout android:layout_height="wrap_content"
                  android:layout_width="match_parent"
                  android:id="@+id/TableLayout01"
                  android:stretchColumns="*">

            <TableRow android:layout_height="wrap_content"
                      android:layout_width="wrap_content"
```

```
android:layout_gravity="center_vertical|center_horizontal"
                    android:id="@+id/TableRow01">

                <ImageView android:layout_height="wrap_content"
                           android:layout_width="wrap_content"

android:layout_gravity="center_vertical|center_horizontal"
                           android:id="@+id/ImageView2_Left"
                           android:src="@drawable/splash1"/>

                <ImageView android:layout_height="wrap_content"
                           android:layout_width="wrap_content"

android:layout_gravity="center_vertical|center_horizontal"
                           android:id="@+id/ImageView2_Right"
                           android:src="@drawable/splash2"/>

        </TableRow>

        <TableRow android:layout_height="wrap_content"
                  android:layout_width="wrap_content"

android:layout_gravity="center_vertical|center_horizontal"
                  android:id="@+id/TableRow02">

                <ImageView android:layout_height="wrap_content"
                           android:layout_width="wrap_content"

android:layout_gravity="center_vertical|center_horizontal"
                           android:id="@+id/ImageView3_Left"
                           android:src="@drawable/splash3"/>

                <ImageView android:layout_height="wrap_content"
                           android:layout_width="wrap_content"

android:layout_gravity="center_vertical|center_horizontal"
                           android:id="@+id/ImageView3_Right"
                           android:src="@drawable/splash4"/>
        </TableRow>

   </TableLayout>

    <TextView android:layout_height="wrap_content"
              android:layout_width="match_parent"
              android:textSize="@dimen/logo_size"
              android:gravity="center"
              android:textColor="@color/logo_color"
              android:text="@string/app_logo_bottom"
              android:id="@+id/TextViewBottomTitle"></TextView>

    <TextView android:background="@color/version_bkgrd"
              android:layout_height="match_parent"
              android:layout_width="match_parent"
              android:textSize="@dimen/version_size"
              android:gravity="center"
              android:layout_gravity="center_vertical|center_horizontal"
              android:textColor="@color/version_color"
              android:text="@string/app_version_info"
```

```
            android:id="@+id/TextViewBottomVersion"
            android:lineSpacingExtra="@dimen/version_spacing"></TextView>

</LinearLayout>
```

Στη συνέχεια φαίνεται πως ακριβώς είναι διαμορφωμένη η διάταξη μέσω του outline.



Ενημέρωση της Διάταξης της splash.xml

Χρησιμοποιούμε ένα μηχανισμό ελέγχου LinearLayout και θέτουμε το χαρακτηριστικό background σε *@android:color/black και το orientation σε vertical.*

Προσθέτουμε ένα μηχανισμό ελέγχου TextView με όνομα TextViewTopTitle και του δίνουμε το όνομα Enviromental

Στη συνέχεια τοποθετούμε ένα μηχανισμό ελέγχου TableLayout με όνομα TableLayout01, με τα χαρακτηριστικά που φαίνονται από τον κώδικα που βρίσκεται πιο πάνω. Θέτουμε το χαρακτηριστικό stretchColumns σε *, για να επεκτείνετε οποιοδήποτε στοιχείο, όταν χρειάζεται, ώστε να χωρά στην οθόνη.

Μέσα στον μηχανισμό TableLayout προσθέτουμε ένα θυγατρικό μηχανισμό ελέγχου TableRow. Μέσα σε αυτόν το μηχανισμό προσθέτουμε δυο μηχανισμούς ελέγχου ImageView. Για τον πρώτο μηχανισμό θέτουμε το χαρακτηριστικό scr στον σχεδιάσιμο πόρο splash1.png με όνομα @drawable/splash1. Το ίδιο κάνουμε και για τον δεύτερο μηχανισμό. Δημιουργούμε ένα δεύτερο TableRow, πάλι προσθέτουμε μηχανισμούς ελέγχου ImageView για τα splash3.png και splash4.png . Παραθέτοντας τις αντίστοιχες εικόνες που χρησιμοποιούμε στην εφαρμογή μας .

Splash1.png

Splash2.png

Splash3.xml

Splash4.png

Προσθέτουμε ένα άλλο μηχανισμό ελέγχου TextView με όνομα TextViewBottomTitle μέσα στο πατρικό LinearLayout. Κάνουμε το χαρακτηριστικό του text στην κατάλληλη συμβολοσειρά, το χαρακτηριστικό του textColor στον κατάλληλο πόρο χρώματος και το χαρακτηριστικό του textSize στον πόρο διάστασης, που δημιουργήσαμε για αυτόν τον σκοπό.

Τέλος, προσθέτουμε ένα άλλο μηχανισμό ελέγχου TextView με όνομα TextViewBottomVersion. Κάνουμε το χαρακτηριστικό του text στην κατάλληλη συμβολοσειρά, το χαρακτηριστικό του textColor στον κατάλληλο πόρο χρώματος και το χαρακτηριστικό του textSize στον πόρο διάστασης, που δημιουργήσαμε. Επίσης θέτουμε το χαρακτηριστικό του background στον πόρο χρώματος που έχουμε σημειώσει στο string.xml και το lineSpacingExtra στην τιμή του πόρου διάστασης του διαστίχου.

Προσθήκη Πόρων Κίνησης

Για την splash οθόνη δημιουργούμε τρείς προσαρμοσμένες κινήσεις σε xml και τις αποθηκεύουμε μέσα σε έναν κατάλογο πόρων /res/anim με fade_in.xml, fade2_in.xml και custom_anim.xml. εν συνεχεία δίνουμε τα παραπάνω xml αρχεία.

fade_in.xml

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

<set android:shareInterpolator="false"
      xmlns:android="http://schemas.android.com/apk/res/android">

   <alpha android:duration="2500"
          android:toAlpha="1.0"
          android:fromAlpha="0.0">
   </alpha>

</set>
```

Η πρώτη κίνηση fade_in.xml εξασθενίζει το στόχο της από μία τιμή alpha 0(διαφανές) σε μια τιμή alpha 1(αδιαφανές), σε μια χρονική περίοδο 2,5 δευτερόλεπτα και επειδή δεν υπάρχει ενσωματωμένος επεξεργαστής θα πρέπει να δημιουργήσουμε την κατάλληλη xml για την ακολουθία της κίνησης.

Το ίδιο ισχύει και για την αντίστοιχη κίνηση fade_in2.xml εκτός του ότι το χαρακτηριστικό startOffset πρέπει να τεθεί στα 2,5 δευτερόλεπτα, αυτό σημαίνει ότι η κίνηση στην πραγματικότητα διαρκεί 5 δευτερόλεπτα.

Το αρχείο fade_in2.xml

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

<set android:shareInterpolator="false"
     xmlns:android="http://schemas.android.com/apk/res/android">

     <alpha android:startOffset="2500"
            android:duration="2500"
            android:toAlpha="1.0"
            android:fromAlpha="0.0">
     </alpha>

</set>
```

Τέλος, δημιουργήσαμε μία ακολουθία κίνησης για το γραφικό TableLayout. Το σύνολο της κίνησης περιέχει πολλαπλές και ταυτόχρονες λειτουργίες, όπως μία περιστροφή, λίγη αλλαγή μεγέθους (κλιμάκωση) και μία εναλλαγή alpha.

Το αρχείο custom_anim.xml

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


<set android:shareInterpolator="false"
    xmlns:android="http://schemas.android.com/apk/res/android">

    <rotate android:duration="2000"
            android:pivotY="50%"
            android:pivotX="50%"
            android:toDegrees="360"
            android:fromDegrees="0"/>

       <alpha android:duration="2000"
              android:toAlpha="1.0"
              android:fromAlpha="0.0">

       </alpha>

           <scale android:duration="2000"
                  android:pivotY="50%"
                  android:pivotX="50%"
                  android:toYScale="1.0"
                  android:toXScale="1.0"
                  android:fromYScale=".1"
                  android:fromXScale=".1"/>
</set>
```

Σχεδίαση της Οθόνης του Main Menu

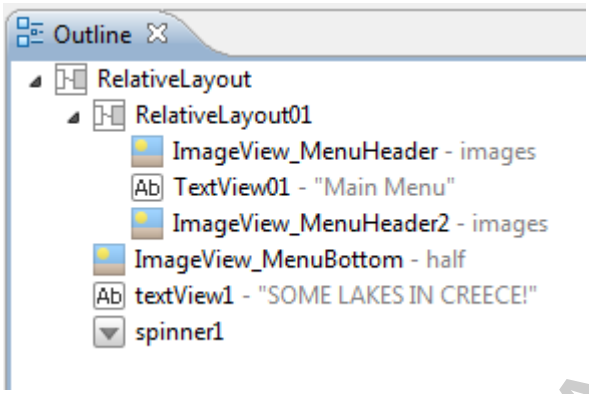
- Menu.xml- μας δίνει τη δυνατότητα μέσω του snipper να επιλέξουμε ποιας λίμνης την επιφάνεια θέλουμε να δούμε.

Το αρχείο menu.xml περιέχει τον παρακάτω κώδικα.

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

    <RelativeLayout
        android:id="@+id/RelativeLayout01"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true" >

        <ImageView
            android:id="@+id/ImageView_MenuHeader"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignBottom="@+id/TextView01"
            android:layout_alignParentLeft="true"
            android:layout_alignParentTop="true"
            android:layout_toLeftOf="@+id/TextView01"
            android:src="@drawable/images" />

        <TextView
            android:id="@+id/TextView01"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentTop="true"
```

```
            android:layout_centerHorizontal="true"
            android:layout_gravity="fill_horizontal|center"
            android:shadowColor="@android:color/white"
            android:shadowDx="0"
            android:shadowDy="0"
            android:shadowRadius="10"
            android:text="@string/menu"
            android:textColor="@color/title_color"
            android:textSize="@dimen/screen_title_size" />

        <ImageView
            android:id="@+id/ImageView_MenuHeader2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignBottom="@+id/ImageView_MenuHeader"
            android:layout_alignParentRight="true"
            android:layout_alignParentTop="true"
            android:layout_toRightOf="@+id/TextView01"
            android:src="@drawable/images" />
    </RelativeLayout>

    <ImageView
        android:id="@+id/ImageView_MenuBottom"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:adjustViewBounds="true"
        android:scaleType="centerInside"
        android:src="@drawable/half" />

    <TextView
        android:id="@+id/textView1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/RelativeLayout01"
        android:padding="10dp"
        android:text="@string/some_text"
        android:textSize="20sp" />

    <Spinner
        android:id="@+id/spinner1"
        android:layout_width="match_parent"
        android:layout_height="55dp"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/textView1"
        android:layout_marginTop="28dp" />

</RelativeLayout>
```

Στη συνέχεια φαίνεται πως είναι διαμορφωμένη η διάταξη μέσω του outline.



Ενημέρωση της Διάταξης της menu.xml

Σχεδίαση της Επικεφαλίδας της Οθόνης με το RelativeLayout

Το RelativeLayout επιτρέπει σε κάθε θυγατρική προβολή να τοποθετηθεί σχετικά προς την πατρική διάταξη ή προς μηχανισμούς ελέγχου μίας άλλης θυγατρικής προβολής. Έτσι μπορούμε εύκολα να περιγράψουμε την επικεφαλίδα ως ένα μηχανισμό ελέγχου RelativeLayout με τρείς θυγατρικές διατάξεις:

- Ένα μηχανισμό ελέγχου ImageView στοιχισμένο στο επάνω αριστερό τμήμα του πατρικού μηχανισμού ελέγχου
- Ένα μηχανισμό ελέγχου TextView στοιχισμένο επάνω και στο κέντρο του πατρικού μηχανισμού ελέγχου
- Ένα μηχανισμό ελέγχου ImageView στοιχισμένο στο επάνω δεξιό τμήμα του πατρικού μηχανισμού ελέγχου
- Ένας μηχανισμός ελέγχου ImageView στοιχισμένο στο RelativeLayout01 κάτω από τους τρεις προηγούμενους μηχανισμούς.
- Ενώ ο μηχανισμός ελέγχου textView1 είναι η επικεφαλίδα στο κυρίως μενού και κρέμεται κάτω από τους άλλους μηχανισμούς
- Τέλος, υπάρχει και ο μηχανισμός spinner οπου εμφανίζονται οι λίμνες, καθώς και το εμβαδό

Ο μηχανισμός ελέγχου Spinner είναι βασικά η έκδοση για την πλατφόρμα Android μιας αναπτυσσόμενης λίστας. Ο μηχανισμός ελέγχου Spinner μοιάζει με μια αναπτυσσόμενης λίστας, όταν κλείνει, αλλά όταν ενεργοποιηθεί, εμφανίζει ένα παράθυρο επιλογής, αντί να σχεδιάζει μια αναπτυσσόμενη λίστα στην κύρια οθόνη.

Θέτουμε το χαρακτηριστικό του background σε @drawable/deka.

Προσθέτουμε ένα δεύτερο μηχανισμό ελέγχου RelativeLayout, θέτουμε το χαρακτηριστικό του android:layout_width σε "wrap_content" και το χαρακτηριστικό του android:layout_height σε "wrap_content" επίσης το χαρακτηριστικό android:layout_alignParentTop σε "true", έτσι ώστε η επικεφαλίδα να τοποθετείται στην κορυφή του πατρικού RelativeLayout.

Τώρα μέσα στο μηχανισμό RelativeLayout προσθέτουμε ένα μηχανισμό ImageView, θέτουμε τα χαρακτηριστικά android:layout_alignParentLeft σε"true" και android:layout_alignParentTop σε "true".

Θέτουμε το χαρακτηριστικό scr της εικόνας στο γραφικό "@drawable/images".

Image.png

Επίσης μέσα στο μηχανισμό μας, προσθέτουμε ένα άλλο μηχανισμό TextView για το κείμενο του τίτλου. Τα χαρακτηριστικά text, textSize, textColor του μηχανισμού μας τα θέτουμε στους πόρους που δημιουργήσαμε.

Ολοκληρώνουμε το μηχανισμό ελέγχου RelativeLayout τοποθετώντας ένα ακόμα , ImageView, θέτουμε τα χαρακτηριστικά android:layout_alignParentLeft σε "true" και android:layout_alignParentTop σε "true". Θέτουμε το χαρακτηριστικό scr της εικόνας στο γραφικό "@drawable/images".

Έξω από το RelativeLayout της επικεφαλίδας, αλλά μέσα στο πατρικό προσθέτουμε το μηχανισμό spinner και ImageView. Αρχίζουμε προσθέτοντας ένα μηχανισμό ελέγχου με όνομα ImageView_MenuBotton, όπου θέτουμε το χαρακτηριστικό scr της εικόνας στο γραφικό "@drawable/half", το χαρακτηριστικό του android:layout_width σε "match_parent" και το χαρακτηριστικό του android:layout_height σε "wrap_content" ώστε ο μηχανισμός να γεμίσει το κάτω μέρος της οθόνης. Επιπρόσθετα, θέτουμε τα χαρακτηριστικά android:layout_alignParentBottom, android:layout_alignParentLeft, android:adjustViewBounds σε "true",το χαρακτηριστικό android:scaleType σε"centerInside"και το χαρακτηριστικό scr της εικόνας στο γραφικό @drawable/half".

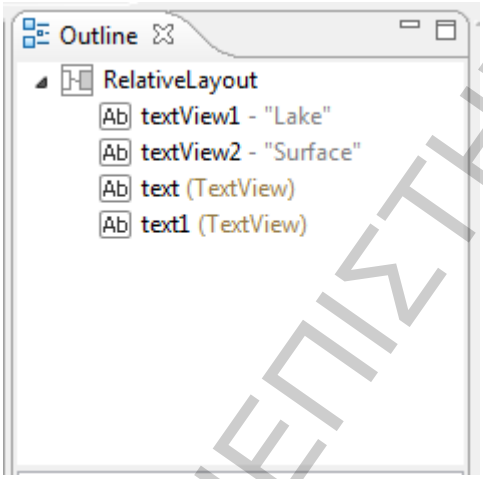
half.png

Το αρχείο row.xml περιέχει τον παρακάτω κώδικα.

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

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="6dip" >

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textView2"
        android:layout_alignParentTop="true"
        android:text="@string/lake"
        android:textAppearance="?android:attr/textAppearanceMedium" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/textView1"
        android:text="@string/surface"
        android:textAppearance="?android:attr/textAppearanceMedium" />

    <TextView
        android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/textView1"
        android:layout_centerHorizontal="true"
        android:text="@string/TextView" />

    <TextView
        android:id="@+id/text1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/textView2"
        android:layout_alignBottom="@+id/textView2"
        android:layout_alignLeft="@+id/text"
        android:text="@string/TextView1" />


</RelativeLayout>
```

Η διαμορφωμένη διάταξη μέσω του outline



Χρησιμοποιούμε ένα μηχανισμό ελέγχου RelativeLayout ώστε να εμφανίζονται με μορφή κειμένου τα δεδομένα που αποθηκεύουμε μέσα στην βάση δεδομένων μας θέτουμε το χαρακτηριστικό του android:layout_width σε "match_parent" και το χαρακτηριστικό του android:layout_height σε "match_parent", επίσης το χαρακτηριστικό android:padding το κάνουμε 6dip. Επομένως, έχουμε ένα πατρικό μηχανισμό και τους θυγατρικούς :

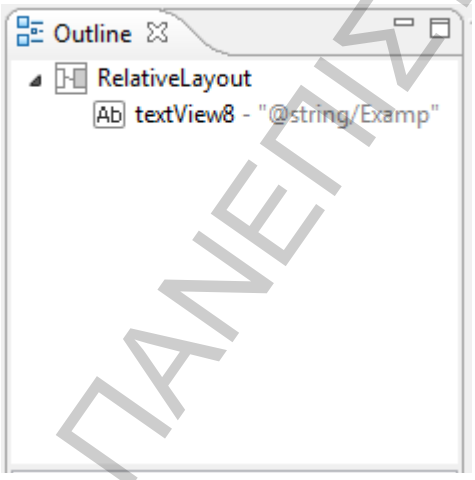
- Ένας μηχανισμός ελέγχου textView1 που βρίσκεται στο επάνω αριστερό σημείο του RelativeLayout και έχει σταθερό όνομα Lake για όποια τιμή και αν πάρει οποιοσδήποτε μηχανισμός

- Το ίδιο ισχύει και για τον μηχανισμό textView2 που βρίσκεται στο κάτω αριστερό σημείο και έχει σταθερό όνομα Surface
- Ο μηχανισμός ελέγχου text βρίσκεται στο επάνω δεξιό σημείο της πατρικής διάταξης και δεξιά από το textView1, αλλά η τιμή του είναι μεταβλητή κάθε φορά που επιλέγουμε την λίμνη της αρεσκείας μας κάνοντας κλικ στο ενδιαφερόμενο σημείο.
- Τέλος, ο μηχανισμός ελέγχου text1 βρίσκεται στο κάτω δεξιά σημείο του RelativeLayout. Η τιμή του αλλάζει αναλόγως την επιλογή μας, εδώ όμως, μας εμφανίζει την επιφάνεια σε τ.χλμ .

Το αρχείο simple_spinner_item.xml περιέχει τον παρακάτω κώδικα

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

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="6dip" >

    <TextView
        android:id="@+id/textView8"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:text="@string/Examp"
        android:textAppearance="?android:attr/textAppearanceMedium" />

</RelativeLayout>
```

Η διαμορφωμένη διάταξη μέσω του outline



Χρησιμοποιούμε ένα μηχανισμό ελέγχου RelativeLayout ώστε να εμφανίζονται με μορφή κειμένου τα δεδομένα που αποθηκεύουμε μέσα στην βάση δεδομένων μας θέτουμε το χαρακτηριστικό του android:layout_width σε "match_parent" και το χαρακτηριστικό του android:layout_height σε "match_parent", επίσης το χαρακτηριστικό android:padding το κάνουμε 6dip.

Προσθήκη Πόρων της Εφαρμογής

Οι πόροι διαιρούνται σε δυο τύπους: πόροι συστήματος και πόροι εφαρμογής. Οι πόροι ορίζονται από τον προγραμματιστή μέσα στα αρχεία έργου του Android και είναι ειδικοί για την εφαρμογή. Οι πόροι συστήματος είναι κοινοί πόροι, που ορίζονται από την πλατφόρμα του Android και είναι προσπελάσιμοι σε όλες τις εφαρμογές μέσω Android SDK . Μπορούμε να προσπελάσουμε και τους δυο τύπους πόρων κατά τον χρόνο εκτέλεσης. Επίσης φορτώνονται πόροι μέσα στον κώδικά μας μέσα από μια δραστηριότητα. Μπορούμε επίσης να αναφερθούμε σε πόρους μέσα από άλλους πόρους. Για παράδειγμα, μπορούμε να αναφερθούμε σε πολυάριθμούς πόρους συμβολοσειρών, διαστάσεων και χρώματος μέσα από έναν πόρο διάταξης XML, για να ορίσουμε τις ιδιότητες και τα χαρακτηριστικά συγκεκριμένων μηχανισμών ελέγχου, όπως είναι τα χρώματα παρασκηνίου και το κείμενο που θα εμφανιστεί.

Οι πόροι εφαρμογής δημιουργούνται και αποθηκεύονται μέσα στα αρχεία έργου του Android, μέσα στον υποκατάλογο /res. Χρησιμοποιώντας μια σωστά καθορισμένη, αλλά ευέλικτη δομή καταλόγου, οι πόροι οργανώνονται, ορίζονται και μεταγλωττίζονται μαζί με το πακέτο της εφαρμογής. Οι πόροι εφαρμογής δεν μπορούν να χρησιμοποιηθούν από το υπόλοιπο σύστημα Android.

Ο ορισμός των δεδομένων εφαρμογής ως πόροι (και όχι κατά τον χρόνο εκτέλεσης μέσα στον κώδικα) είναι μια σωστή προγραμματιστική τεχνική. Η ομαδοποίηση πόρων εφαρμογής και η μεταγλώττισή τους μέσα στο πακέτο της εφαρμογής έχει τα παρακάτω πλεονεκτήματα:

- Ο κώδικας είναι σαφέστερος και πιο ευανάγνωστος, γεγονός που οδηγεί σε λιγότερα σφάλματά
- Οι πόροι οργανώνονται ανά τύπο και είναι εγγυημένα μοναδικοί
- Οι πόροι τοποθετούνται σε βολικές θέσεις, για να μπορεί να γίνει προσαρμογή ανάλογα με την τηλεφωνική συσκευή
- Η τοπικοποίηση και διεθνοποίηση γίνονται εύκολα

Η πλατφόρμα Android υποστηρίζει διάφορους τύπους πόρων, οι οποίοι μπορούν να συνδυαστούν για να δημιουργήσουν διάφορους τύπους εφαρμογών.
Οι εφαρμογές Android μπορούν να περιλαμβάνουν πολλά και διαφορετικά είδη πόρων. Παρακάτω αναφέρουμε μερικούς από τους συνηθέστερους τύπους πόρων:

- Συμβολοσειρές, χρώματα και διαστάσεις
- Αρχεία σχεδιάσιμων γραφικών
- Αρχεία διάταξης
- Ακατέργαστα αρχεία όλων των τύπων

Οι τύποι πόρων ορίζονται με ειδικές σημάνσεις XML και οργανώνονται μέσα σε ειδικά ονομασμένους καταλόγους του έργου. Ορισμένοι υποκατάλογοι /res, όπως τους καταλόγους /drawable,/layout και /values, δημιουργούνται προεπιλεγμένα όταν δημιουργείται ένα νέο έργο Android, αλλά άλλοι πρέπει να προστεθούν από τον προγραμματιστή, όταν χρειάζεται.
Τα αρχεία πόρων που αποθηκεύονται μέσα σε υποκαταλόγους /res, πρέπει να ακολουθούν τους παρακάτω κανόνες:

- Τα ονόματα αρχείων πόρων πρέπει να γράφονται με πεζά γράμματα

- Τα ονόματα αρχείων πόρων μπορούν να περιέχουν μόνο γράμματα, αριθμούς, υπογραμμίσεις και τελείες
- Τα ονόματα αρχείων πόρων (και τα χαρακτηριστικά ονομάτων XML) πρέπει να είναι μοναδικά

Έχοντας σχεδιάσει τη διάταξη, πρέπει να δημιουργήσουμε τους πόρους σχεδιάσιμων, συμβολοσειράς, χρωμάτων και διάστασης που χρησιμοποιούμε μέσα στις διατάξεις της εφαρμογής.

Μπορούμε να χρησιμοποιήσουμε πόρους συμβολοσειράς οπουδήποτε χρειάζεται η εφαρμογή μας να εμφανίσει κείμενο. Ορίζουμε πόρου συμβολοσειράς με την σήμανση <string>, τους αναγνωρίζουμε με την ιδιότητά name και τους αποθηκεύουμε μέσα στο αντίστοιχο αρχείο πόρων.
Το αρχείο xml για τους πόρους συμβολοσειράς μέσα στο /res/values/strings.xml είναι το ακόλουθο:

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

<resources>

    <string name="app_name">Environ Inform</string>
    <string name="menu">Main Menu</string>
    <string name="action_settings">Settings</string>
    <string name="some_hint"> Type here to filter&#8230;</string>
    <string name="app_logo_top">ENVIROMENTAL</string>
    <string name="app_logo_bottom">INFORMATICS</string>
    <string name="app_version_info">University of Piraeus\nCopyright ©
2013\nDemo </string>
    <string name="some_text">SOME LAKES IN CREECE!</string>
    <string name="lake">Lake</string>
    <string name="surface">Surface</string>
     <string name="TextView"></string>
     <string name="TextView1"></string>
     <string name="Examp"></string>
</resources>
```

Εφαρμόζουμε πόρους χρωμάτων σε μηχανισμούς ελέγχου της οθόνης. Ορίζουμε πόρους χρώματος με την σήμανση <color>, τους αναγνωρίζουμε με το χαρακτηριστικό name και του αποθηκεύουμε στο αντίστοιχο αρχείο πόρων.
Μπορούμε να προσθέσουμε ένα νέο αρχείο XML, όπως αυτό, επιλέγοντας File, New, Android XML File και μετά να συμπληρώσουμε το προκύπτον παράθυρο διαλόγου με τον τύπο αρχείου. Αυτή η ενέργεια καθορίζει αυτόματα τον αναμενόμενο φάκελο και τύπο αρχείου για το έργο Android. Το αρχείο xml για τους πόρους χρωμάτων μέσα στο /res/values/colors.xml είναι το ακόλουθο:

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

<resources>

    <color name="logo_color">#FF00FF00</color>
    <color name="version_color">#f0f0f0</color>
    <color name="version_bkgrd">#1a1a48</color>
    <color name="title_color">#FF00FF00</color>
```

```
    <color name="title_glow">#F00</color>
    <color name="menu_color">#FFFF0F</color>
    <color name="menu_glow">#F00</color>

</resources>
```

Για τον καθορισμό του μεγέθους ενός μηχανισμού ελέγχου διεπαφής χρήστη, σαν τους μηχανισμούς ελέγχου Button ή TextView, πρέπει να καθορίσουμε διαφορετικά είδη διαστάσεων. Οι πόροι διαστάσεων είναι χρήσιμοι για μεγέθη γραμματοσειρών, για μεγέθη εικόνων και άλλες φυσικές μετρήσεις ή μετρήσεις σε pixels. Ορίζουμε πόρους διαστάσεων με την σήμανση <dimen>, τους αναγνωρίζουμε με την ιδιότητα name και το αποθηκεύουμε μέσα στο αρχείο πόρου. Το αρχείο xml για τους πόρους διαστάσεων των κειμένων μέσα στο /res/values/dimens.xml είναι το ακόλουθο:

```
<resources>

    <!-- Default screen margins, per the Android Design guidelines. -->
    <dimen name="logo_size">35dp</dimen>
    <dimen name="version_size">15dp</dimen>
    <dimen name="version_spacing">3dp</dimen>
    <dimen name="screen_title_size">30dp</dimen>
    <dimen name="menu_item_size">34dp</dimen>
</resources>
```

Το αρχείο xml για τους πόρους του στυλ των κειμένων μέσα στο /res/values/styles.xml είναι το ακόλουθο:

```
<resources>

    <!--
        Base application theme, dependent on API level. This theme is
replaced
        by AppBaseTheme from res/values-vXX/styles.xml on newer devices.
    -->
    <style name="AppBaseTheme" parent="android:Theme.Light">
        <!--
            Theme customizations available in newer API levels can go in
            res/values-vXX/styles.xml, while customizations related to
            backward-compatibility can go here.
        -->
    </style>

    <!-- Application theme. -->
    <style name="AppTheme" parent="AppBaseTheme">
        <!-- All customizations that are NOT specific to a particular API-
level can go here. -->
    </style>

</resources>
```

Επεξεργασία του Αρχείου Manifest του Android

Το αρχείο manifest του Android είναι το κεντρικό αρχείο διαμόρφωσης για την εφαρμογή μας. Το σύστημα Android χρησιμοποιεί τις πληροφορίες αυτού του αρχείου για να κάνει τα παρακάτω:

- Εγκατάσταση και ενημέρωση του πακέτου της εφαρμογής
- Εμφάνιση λεπτομερειών εφαρμογής σε χρήστες
- Εκκίνηση δραστηριοτήτων εφαρμογής
- Διαχείριση αδειών εφαρμογής
- Χειρισμός αρκετών άλλων προχωρημένων διαμορφώσεων εφαρμογής, που περιλαμβάνουν λειτουργία ως πάροχο υπηρεσιών ή ως πάροχο περιεχομένου

Χρησιμοποιούμε αυτήν την καρτέλα για να προσπελάσουμε τον επεξεργαστή XML

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.envirinfor1"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="15"
        android:targetSdkVersion="18" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/images"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.example.envirinfor1.MainSplashActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER"
/>
            </intent-filter>
        </activity>
        <activity android:name="MainMenuActivity"/>
    </application>

</manifest>
```

Η διαμόρφωση βασικών ρυθμίσεων εφαρμογής για το αρχείο Manifest του Android, αρχίζει πάντα με μια επικεφαλίδα XML όπως η παρακάτω: <?xml version="1.0" encoding="utf-8"?>.

Η ονομασία πακέτων , ορίζει τις λεπτομέρειες της εφαρμογής μέσα στην εμβέλεια της σήμανσης <manifest>, θέτουμε αυτήν την τιμή χρησιμοποιώντας το χαρακτηριστικό package ως εξής:

<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.example.envirinfor1"
  android:versionCode="1"

android:versionName="1.0" >

Οι εκδόσεις μιας εφαρμογής χρησιμοποιούνται για δύο σκοπούς:

- Για οργάνωση και παρακολούθηση των χαρακτηριστικών της εφαρμογής
- Για διαχείριση αναβαθμίσεων της εφαρμογής

Για αυτό το λόγο η σήμανση manifest έχει δυο ξεχωριστά χαρακτηριστικά έκδοσης: ένα όνομα έκδοσης και έναν κωδικό έκδοσης.

Επίσης χρησιμοποιεί την σήμανση <uses_sdk> για να καθοριστεί το ελάχιστο SDK που απαιτείται να έχει η συσκευή για να δομηθεί και να εκτελεστεί σωστά η εφαρμογή : <uses-sdk
android:minSdkVersion=*"15"*
android:targetSdkVersion=*"18"* />
Μέσα στη σήμανση του <application> γίνεται η ονομασία της εφαρμογής, θέτουμε το χαρακτηριστικό android:label σε ένα πόρο συμβολοσειράς, η *"@string/app_name"* μέσα στο αρχείο strings.xml δίνει το όνομα της εφαρμογής.

Το χαρακτηριστικό android:icon είναι ένας πόρος drawable όπου θέτουμε το εικονίδιο της εφαρμογής ως εξής: android:icon=*"@drawable/images"*

Πρέπει να καταχωρίσουμε κάθε δραστηριότητα μέσα στην ενότητα Application Nodes της καρτέλας Application. Κάθε δραστηριότητα έχει τη δική της σήμανση <activity> μέσα στην προκύπτουσα XML. Στην δικιά μας εφαρμογή υπάρχουν δύο κλήσεις δραστηριοτήτων.

Υλοποίηση Δραστηριοτήτων Εφαρμογής

Για να υλοποιηθεί η βασική κλάση Activity, αντιγράφοντας απλώς το αρχείο πηγαίου κώδικα με όνομα MainActivity.java, ονομάζουμε αυτό το νέο αρχείο κλάσης MainActivity και αποθηκεύουμε το αρχείο. Αυτή έχει την παρακάτω μορφή:

```
package com.example.envirinfor;

import android.app.Activity;

public class MainActivity extends Activity {
      public static final String GAME_PREFERENCES="GamePrefs";
}
```

Για να δηλώσουμε το όνομα του συνόλου των προτιμήσεών μας μέσα στην βασική κλάση MainActivity, ώστε να είναι εύκολα προσπελάσιμες σε όλες τις δευτερεύουσες κλάσεις: public static final String *GAME_PREFERENCES*="GamePrefs";

MainSplashActivity

```
package com.example.envirinfor1;

import android.content.Intent;
```

```
import android.os.Bundle;
import android.view.animation.Animation;
import android.view.animation.Animation.AnimationListener;
import android.view.animation.AnimationUtils;
import android.view.animation.LayoutAnimationController;
import android.widget.TableLayout;
import android.widget.TableRow;
import android.widget.TextView;

public class MainSplashActivity extends MainActivity {
    /** Καλείται όταν η δραστηριότητα δημιουργείται πρώτη φορά. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.splash);
        startAnimating();
    }

    /**
     * Η βοηθητική μέθοδος για να ξεκινήσει το animation στην splash οθόνη
     */
    private void startAnimating() {
        // Fade in εφαρμογή στον τίτλο
        TextView logo1 = (TextView) findViewById(R.id.TextViewTopTitle);
        Animation fade1 = AnimationUtils.loadAnimation(this,
R.anim.fade_in);
        logo1.startAnimation(fade1);
        // Fade in2 του κάτω μέρος του τίτλου μετά από μια ενσωματωμένη
καθυστέρηση.
        TextView logo2 = (TextView) findViewById(R.id.TextViewBottomTitle);
        Animation fade2 = AnimationUtils.loadAnimation(this,
R.anim.fade_in2);
        logo2.startAnimation(fade2);
        // Μετάβαση στο Main Menu όταν το κάτω μέρος του τίτλου ολοκληρώσει
το animating
        fade2.setAnimationListener(new AnimationListener() {
            public void onAnimationEnd(Animation animation) {
                // Το animation έχει τελειώσει, η μετάβαση στην οθόνη του
Main Menu
                startActivity(new Intent(MainSplashActivity.this,
MainMenuActivity.class));
                MainSplashActivity.this.finish();
            }

            public void onAnimationRepeat(Animation animation) {
            }

            public void onAnimationStart(Animation animation) {
            }
        });
        // Φόρτωση των animations για όλα τα views εντός του TableLayout
        Animation spinin = AnimationUtils.loadAnimation(this,
R.anim.custom_anim);
        LayoutAnimationController controller = new
LayoutAnimationController(spinin);
        TableLayout table = (TableLayout) findViewById(R.id.TableLayout01);
        for (int i = 0; i < table.getChildCount(); i++) {
            TableRow row = (TableRow) table.getChildAt(i);
            row.setLayoutAnimation(controller);
```

```
        }
    }

    @Override
    protected void onPause() {
        super.onPause();
        // Διακοπή του animation
        TextView logo1 = (TextView) findViewById(R.id.TextViewTopTitle);
        logo1.clearAnimation();
        TextView logo2 = (TextView) findViewById(R.id.TextViewBottomTitle);
        logo2.clearAnimation();
        TableLayout table = (TableLayout) findViewById(R.id.TableLayout01);
        for (int i = 0; i < table.getChildCount(); i++) {
            TableRow row = (TableRow) table.getChildAt(i);
            row.clearAnimation();
        }
    }

    @Override
    protected void onResume() {
        super.onResume();

        // Ξεκινώντας το animating στην αρχή έτσι έχουμε την πλήρη splash
οθόνη
        startAnimating();
 }
}
```

Δημιουργούμε την MainSplashMenu που είναι επέκταση της MainActivity και φτιάχνουμε την public void onCreate που καλείται όταν η δραστηριότητα δημιουργείται πρώτη φορά, η setContentView(R.layout.*splash*) καλείται όταν το τεμάχιο πρέπει να δημιουργήσει τη splash προβολή. Εν συνεχεία, καλείται η startAnimating() η οποία είναι μια βοηθητική μέθοδο για να ξεκινήσει το animation της splash οθόνης.

Η εφαρμογή της κίνησης fade_in στον μηχανισμό ελέγχου TextView του τίτλου, που καλείται TextViewTopTitle. Ανακτούμε ένα στιγμιότυπο του μηχανισμού ελέγχου TextView μέσα στην μέθοδο onCreate() της κλάσης, φορτώνουμε τον πόρο κίνησης μέσα στο αντικείμενο Animation και καλούμε την μέθοδο startAnimation() :
TextView logo1 = (TextView) findViewById(R.id.*TextViewTopTitle*);
Animation fade1 = AnimationUtils.*loadAnimation*(this, R.anim.*fade_in*);
logo1.startAnimation(fade1);

Η εφαρμογή της κίνησης fade_in2, είναι παρόμοια με την fade_in αλλά, αφορά το κάτω μέρος του τίτλου μετά από μια ενσωματωμένη καθυστέρηση. Εν συνεχεία, γίνεται μετάβαση Main Menu όταν το κάτω μέρος του τίτλου ολοκληρώσει το animating.

Αφού το animation έχει ολοκληρωθεί, γίνεται μετάβαση στην οθόνη του Main Menu, ουσιαστικά μιλάμε για την εναλλαγή δραστηριότητας ανάμεσα στην MainSplashActivity και MainMenuActivity, όταν ολοκληρωθεί η κίνηση. Για να γίνει αυτό, δημιουργούμε ένα νέο μηχανισμό ελέγχου Intent, ώστε να εκκινεί την κλάση και τον μεταβιβάζουμε μέσα στην μέθοδο startActivity(). Κατόπιν, καλούμε την μέθοδο finish() της MainMenuActivity επειδή δεν θέλουμε να την κρατήσουμε στην

στοίβα δραστηριοτήτων. Η κίνηση fade_in2 διαρκεί 5 δευτερόλεπτα ουσιαστικά είναι η κίνηση που θέλουμε να εκκινεί την εναλλαγή.

Αυτό το κάνουμε δημιουργώντας ένα αντικείμενο, που περιέχει ειδοποιήσεις για τον κύκλο ζωής της κίνησης, όπως είναι start, end και repeat. Μόνο η μέθοδος onAnimationEnd() πρέπει να υλοποιηθεί όπου εισάγουμε κώδικα για την εκκίνηση της νέας δραστηριότητας. Αυτό που περιγράψαμε φαίνεται στον παρακάτω κώδικα:

```
fade2.setAnimationListener(new AnimationListener() {
        public void onAnimationEnd(Animation animation) {
           startActivity(new Intent(MainSplashActivity.this,
MainMenuActivity.class));
          MainSplashActivity .this.finish();
        }
```

Εκτός της εφαρμογής κινήσεων σε μεμονωμένους μηχανισμούς ελέγχου View, εφαρμόζονται και στους θυγατρικούς μηχανισμούς ελέγχου View μέσα σε ένα πατρικό μηχανισμό ελέγχου, χρησιμοποιώντας ένα αντικείμενο LayoutAnimationController.

Για να κινήσουν οι μηχανισμοί ελέγχου με αυτόν τον τρόπο, πρέπει να φορτώσουμε την κίνηση, να δημιουργήσουμε ένα στιγμιότυπο ενός LayoutAnimationController, να το διαμορφώσουμε και μετά να το μεταβιβάσουμε στη μέθοδο setLayoutAnimation() της διάταξης. Στο συγκεκριμένο σημείο φορτώνεται η κίνηση custom_anim, δημιουργούμε ένα LayoutAnimationController που το εφαρμόζει σε κάθε TableRow μέσα στον μηχανισμό ελέγχου TableLayout. Αυτό φαίνεται στον παρακάτω σημείο του κώδικα :

```
Animation spinin = AnimationUtils.loadAnimation(this, R.anim.custom_anim);
    LayoutAnimationController controller = new
LayoutAnimationController(spinin);
    TableLayout table = (TableLayout) findViewById(R.id.TableLayout01);
    for (int i = 0; i < table.getChildCount(); i++) {
      TableRow row = (TableRow) table.getChildAt(i);
      row.setLayoutAnimation(controller);
    }
```

Με τον τρόπο αυτό δεν χρειάζεται να καλέσουμε την μέθοδο startAnimation() επειδή το χειρίζεται το LayoutAnimationController, έτσι η κίνηση εφαρμόζεται σε κάθε θυγατρική προβολή, αλλά η καθεμία αρχίζει σε διαφορετικό χρόνο.

Για διακοπεί η κίνηση, μέσα στην μέθοδο ειδοποίησης onPause() της δραστηριότητας καλούμε την μέθοδο clearAnimation(). Η διακοπή κινήσεων LayoutAnimationController δεν είναι διαφορετική από την διακοπή μεμονωμένων κινήσεων.

```
protected void onPause() {
    super.onPause();
    TextView logo1 = (TextView) findViewById(R.id.TextViewTopTitle);
    logo1.clearAnimation();
    TextView logo2 = (TextView) findViewById(R.id.TextViewBottomTitle);
```

```
        logo2.clearAnimation();
        TableLayout table = (TableLayout) findViewById(R.id.TableLayout01);
        for (int i = 0; i < table.getChildCount(); i++) {
            TableRow row = (TableRow) table.getChildAt(i);
            row.clearAnimation();
        }
    }
```

MainMenuActivity

Η κλάση MainMenuActivity.java έχει την παρακάτω μορφή :

```
package com.example.envirinfor1;

import com.example.envirinfor1.R;
import com.example.envirinfor1.SQLiteAdapter;
import android.os.Bundle;
import android.view.Menu;
import android.database.Cursor;
import android.widget.Spinner;
import android.support.v4.widget.SimpleCursorAdapter;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Toast;
import android.widget.AdapterView.OnItemSelectedListener;

public class MainMenuActivity extends MainActivity {

	private SQLiteAdapter mySQLiteAdapter;
	private SimpleCursorAdapter cursorAdapter;

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

		mySQLiteAdapter = new SQLiteAdapter(this);
		mySQLiteAdapter.open();
		mySQLiteAdapter.deleteAll();
		mySQLiteAdapter.insertLake();
		displaySpinner();

	}

	private void displaySpinner(){

		Cursor mcursor = mySQLiteAdapter.queueAll();


		final String[] from1 = new String[]{
				SQLiteAdapter.KEY_CONTENT,
				SQLiteAdapter.KEY_CONTENT1
				};
		int[] to = new int[]{
				R.id.text,
				R.id.text1
				};
```

```
            cursorAdapter = new SimpleCursorAdapter(
                    this,R.layout.row,mcursor,from1,to,0
                    );

             Spinner spinner = (Spinner) findViewById(R.id.spinner1);

            spinner.setAdapter(cursorAdapter);
            spinner.setOnItemSelectedListener(new OnItemSelectedListener()
              {
                   public void onItemSelected(AdapterView<?> spinner,
                   View view, int pos, long id)
                   {

                    Cursor cursor = (Cursor)spinner.getItemAtPosition(pos);

                    String surfaceText =
                    cursor.getString(cursor.getColumnIndexOrThrow("text"));
                    String surfaceText1 =
                   cursor.getString(cursor.getColumnIndexOrThrow("text1"));
                    Toast.makeText(getApplicationContext(),
                   "The lake " + surfaceText+" has a surface"
+surfaceText1+ "sq.km.", Toast.LENGTH_SHORT).show();
                   }


public void onNothingSelected(AdapterView<?> arg0) {}
               });

        }

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

}
```

Δημιουργούμε την MainMenuActivity που είναι επέκταση της MainActivity καλούμε την SQLiteAdapter με την mySQLiteAdapter και την κάνουμε private. Φτιάχνουμε την public void onCreate που καλείται όταν η δραστηριότητα δημιουργείται πρώτη φορά, η setContentView(R.layout.*menu*) καλείται όταν το τεμάχιο πρέπει να δημιουργήσει τη menu προβολή. Ουσιαστικά, μπαίνουμε στο κυρίως μενού της εφαρμογής μας. Στη συνέχεια δημιουργούμε και ανοίγουμε μια βάση δεδομένων SQLite : mySQLiteAdapter = new SQLiteAdapter(this); την οποία την έχουμε δημιουργήσει μέσα στην SQLiteAdapter, μετά καλείται η συνάρτηση open() για να ξεκινήσει η εγγραφή των δεδομένων, επίσης καλούμε την συνάρτηση deletAll(), για να καθαρίσουμε το πεδίο της βάσης έτσι ώστε να ξεκινήσει η εγγραφή των δεδομένων της εφαρμογής. Τα δεδομένα που εισάγουμε μέσα στη βάση είναι στο _id είναι το κλειδί του πίνακα μας και είναι αριθμός, το text που είναι αποθηκευμένα τα ονόματα των λιμνών και τέλος, το text1 που είναι αποθηκευμένες οι επιφάνειες των

λιμνών που έχουμε εισάγει στην εφαρμογή. Στην συνέχεια καλούμε την συνάρτηση mySQLiteAdapter.insertLake() η οποία εισάγει τα δεδομένα στο πίνακα που έχουμε δημιουργήσει, αυτό δεν γίνεται δυναμικά αλλά μέσα από τον κώδικα χειροκίνητα. Τέλος, καλείται η συνάρτηση displaySpinner().

Την οποία, την δημιουργούμε πιο κάτω, την καλούμε private void. Καλούμε την συνάρτηση Cursor queueAll() όπου δημιουργούμε τον πίνακα KEY_ID και KEY_CONTENT και KEY_CONTENT1 στην ουσία έναν πίνακα τριών στηλών, για να καταχωρηθούν οι καταχωρήσεις που κάναμε. Προχωρώντας καλούμε το σημείο του πίνακα που θα κάνουμε την εισχώρηση των δεδομένων που είναι το KEY_CONTENT και το KEY_CONTENT1. Πιο κάτω καλούμε με το int το text και το text1, είναι τα σημεία όπου θα εμφανίζονται μέσω του spinner οι συμβολοσειρές που θα επιλέγουμε κάθε φορά για να ανακτήσουμε τα δεδομένα μας. Το text είναι ένα TextView που εμφανίζεται με την μορφή android:id=”@+id/text” είναι μέσα στο row.xml. Το ίδιο ισχύει και για το text1 αλλά έχει τη μορφή android:id=”@+id/text1.

```
Cursor mcursor = mySQLiteAdapter.queueAll();
        final String[] from1 = new String[]{
                SQLiteAdapter.KEY_CONTENT,
                SQLiteAdapter.KEY_CONTENT1  };
        int[] to = new int[]{
                R.id.text,
                R.id.text1
                };
```

Για να μπορέσουμε να επιλέξουμε μέσα από το spinner την συμβολοσειρά με την λίμνη της αλλά και την επιφάνεια της αρεσκείας μας με τον κέρσορα χρησιμοποιούμε την SimpleCursorAdapter cursorAdapter =new SimpleCursorAdapter(this, R.layout.*row*, cursor, from, to,0); spinner.setAdapter(cursorAdapter);
Στη συνέχεια καλούμε την findViewById για να καλέσουμε το spinner όπου θα εμφανίζεται οι λίμνες στην εφαρμογής.

```
Spinner spinner = (Spinner) findViewById(R.id.spinner1);
```

Τέλος, έχουμε την τελευταία λειτουργία που κάνει η εφαρμογή μας ώστε να μας εμφανίζει κάθε φορά το επιθυμητό μήνυμα, αναλόγως ποια λίμνη έχουμε επιλέξει. Ουσιαστικά, όλη η μέθοδος καλείται spinner.setOnItemSelectedListener(new OnItemSelectedListener() και ενεργοποιεί το spinner ώστε να υπακούει στο κλικάρισμα του επιλεγόμενου σημείου, εντός του spinner. Το public void onItemSelected(AdapterView<?> spinner, View view, int pos, long id) μας εισάγει στη βάση δεδομένων για να μπορέσουμε να ανακτήσουμε τα δεδομένα που θέλουμε για την εύρυθμη λειτουργία της εφαρμογής. Αυτό γίνεται με την Cursor cursor = (Cursor) spinner.getItemAtPosition(pos).
Η String surfaceText =cursor.getString (cursor.getColumnIndexOrThrow("text")) και η String surfaceText1 =cursor.getString (cursor.getColumnIndexOrThrow("text1")); ανακτούν ειδικά τα επιλεγόμενα σημεία text και text1 με τη μορφή συμβολοσειράς και με την συνάρτηση Toast.*makeText*(getApplicationContext(), "The lake " + surfaceText+" has a surface "+surfaceText1+" sq.km.", Toast.*LENGTH_SHORT*).show(), μας δίνει το κατάλληλο μήνυμα με τις απαραίτητες πληροφορίες που αναζητούσαμε από την εφαρμόγη.

SQLiteAdapter

Οι εφαρμογές Android μπορούν να έχουν μια τοπικά προσπελάσιμη ιδιωτική βάση δεδομένων, που βασίζεται στην SQLite. Οι σχεσιακές βάσεις δεδομένων SQLite είναι ελαφρού τύπου, βασιζόμενες σε αρχεία-ιδανικές για κινητές συσκευές. Το Android SDK περιλαμβάνει αρκετές χρήσιμες κλάσεις διαχείρισης βάσεων δεδομένων SQLite. Μπορούμε να βρούμε την υποστήριξη SQLite για την πλατφόρμα Android στο πακέτο android.database.sqlite. Εδώ μπορούμε να βρούμε βοηθητικές κλάσεις για τα παρακάτω:

- Δημιουργία, δημιουργία εκδόσεων και διαχείριση βάσεων δεδομένων
- Δόμηση σωστών ερωτημάτων SQL
- Βρόχο μέσα στα αποτελέσματα ερωτήματος με αντικείμενα Cursor
- Επεξεργασία συναλλαγών βάσεων δεδομένων
- Χειρισμό εξειδικευμένων εξαιρέσεων βάσεων δεδομένων

Η κλάση SQLiteAdapter.java έχει την παρακάτω μορφή:

```
package com.example.envirinfor1;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;

public class SQLiteAdapter {

private static final String MYDATABASE_NAME = "MY_DATABASE";
private static final String MYDATABASE_TABLE = "MY_TABLE";
private static final int MYDATABASE_VERSION = 1;

public static final String KEY_ID = "_id";
public static final String KEY_CONTENT = "text";
public static final String KEY_CONTENT1 = "text1";

private static final String SCRIPT_CREATE_DATABASE =
        "CREATE TABLE IF NOT EXISTS " + MYDATABASE_TABLE + " (" +
            KEY_ID +" INTEGER PRIMARY KEY AUTOINCREMENT, " +
          KEY_CONTENT +","+
          KEY_CONTENT1 +","+
            "UNIQUE ("+KEY_CONTENT+"));";

private static final String TAG = "SQLiteAdapter";
private SQLiteHelper sqLiteHelper;
private SQLiteDatabase sqLiteDatabase;

private final Context mContext;

private static class SQLiteHelper extends SQLiteOpenHelper {
```

```
SQLiteHelper(Context context) {
 super(context,MYDATABASE_NAME,null,MYDATABASE_VERSION);
}

@Override
public void onCreate(SQLiteDatabase db) {
 // TODO Auto-generated method stub
       Log.w(TAG,SCRIPT_CREATE_DATABASE);
       db.execSQL(SCRIPT_CREATE_DATABASE);
}

@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
 // TODO Auto-generated method stub
       Log.w(TAG, "Upgrading database from version " + oldVersion + " to "
                  + newVersion + ", which will destroy all old data");
       db.execSQL("DROP TABLE IF EXISTS "+MYDATABASE_TABLE);
    onCreate(db);
}
}


public SQLiteAdapter(Context c){
       this.mContext=c;
}

public SQLiteAdapter open() throws android.database.SQLException {
sqLiteHelper = new SQLiteHelper( mContext);
sqLiteDatabase = sqLiteHelper.getWritableDatabase();
return this;
}

public void close() {
        if (sqLiteHelper!= null) {
              sqLiteHelper.close();
        }
       }

public long insert(String text,String text1){

ContentValues contentValues = new ContentValues();
contentValues.put(KEY_CONTENT, text);
contentValues.put(KEY_CONTENT1, text1);
return sqLiteDatabase.insert(MYDATABASE_TABLE, null, contentValues);
}

public boolean deleteAll(){
int doneDelete=0;
doneDelete=sqLiteDatabase.delete(MYDATABASE_TABLE, null, null);
Log.w(TAG, Integer.toString(doneDelete));
return doneDelete > 0;
}

public Cursor queueAll(){
String[] from = new String[]{KEY_ID, KEY_CONTENT,KEY_CONTENT1};
Cursor cursor = sqLiteDatabase.query(MYDATABASE_TABLE, from,
  null, null, null, null, null);
if(cursor!=null){
       cursor.moveToFirst();
```

```
        }
return cursor;
}

public void insertLake() {

          insert("Trixonida","95,8");
          insert("Bolbe","75,456");
          insert("Begoritida","54,31");
          insert("Bistonida","45,03");
          insert("Koronia","42,823");
          insert("Mikri Prespa","42,541");
          insert("Megali Prespa","39,4");
          insert("Orestiada","28,655");
          insert("Pambotida","19,47");
          insert("Yliki","19,118");
          insert("Doirani","15,35");
          insert("Ambrakia","14,477");
          insert("Lusimaxia","13,085");
          insert("Petron","12,294");
          insert("Paralimni","10,93");
          insert("Xeimaditida","10,8");
          insert("Ozeros","9,45");
          insert("Boulkaria","9,207");
          insert("Dustos","5,165");
          insert("Pikrolimni","3,772");
          insert("Stumfalia","3,545");
          insert("Mitrikou","2,524");
          insert("Saltini","1,986");
          insert("Zazari","1,845");
          insert("Kaiafa","1,68");
          insert("Morfi","0,97");

        }

}
```

Δημιουργούμε την κλάση SQLiteAdapter και την καλούμε public, μέσα στην συγκεκριμένη κλάση ξεκινάμε την δημιουργία της βάσης της εφαρμογής μας σύμφωνα με τις προδιαγραφές της SQLite, που πρέπει να εφαρμόσουμε ώστε να την κατασκευάσουμε με σωστό τρόπο και να είναι λειτουργική για τους σκοπούς της εφαρμογής. Αρχικά δίνουμε τα ονόματα ώστε να τα χρησιμοποιήσουμε για την κατασκευή του προγράμματος και ορίζοντας ως public static final String:

private static final String *MYDATABASE_NAME* = "MY_DATABASE";
private static final String *MYDATABASE_TABLE* = "MY_TABLE";
private static final int *MYDATABASE_VERSION* = 1;

public static final String *KEY_ID* = "_id";
public static final String *KEY_CONTENT* = "text";
public static final String *KEY_CONTENT1* = "text1";

εκτός του public static final int *MYDATABASE_VERSION* = 1; που το καλεί int γιατί είναι ακέραιος και ουσιαστικό μας δίνει την version.

Δημιουργούμε τον πίνακα MYDATABASE_TABLE και το θέτουμε private static final, τον κάνουμε με τρεις στήλες, στη μια το KEY_ID που είναι ακέραιος και παράλληλα το κυρίως κλειδί του πίνακα και στην άλλη το KEY_CONTENT και τέλος το KEY_CONTENT1 όπου αποθηκεύουμε τα στοιχεία που θέλουμε σε μορφή κειμένου.

```
private static final String SCRIPT_CREATE_DATABASE =
    "CREATE TABLE IF NOT EXISTS " + MYDATABASE_TABLE + " (" +
            KEY_ID +" INTEGER PRIMARY KEY AUTOINCREMENT, " +
        KEY_CONTENT +","+
        KEY_CONTENT1 +","+
       "UNIQUE ("+KEY_CONTENT+"));";
```

Στη συνέχεια δημιουργούμε την SQLiteHelper η οποία είναι επέκταση της SQLiteOpenHelper , με την public SQLiteHelper(Context context) {super (context, *MYDATABASE_NAME*,null,*MYDATABASE_VERSION*)}, δημιουργούμε έναν constructor,λαμβάνει και κρατά μια αναφορά του περασμένου πλαισίου προκειμένου να έχουν πρόσβαση στα assets στοιχεία και πόρους. Η βάση δεδομένων στην πραγματικότητα δεν δημιουργήθηκε ή ανοίχθηκε έως ότου η getWritableDatabase () καλεστεί. Το context για να ανοίξει ή να δημιουργήσει την βάση, το name του αρχείου βάσης δεδομένων, το factory για να χρησιμοποιήσει για την δημιουργία δρομέα αντικείμενων και το version ακέραιός αριθμός της βάσης δεδομένων.

Η public void onCreate(SQLiteDatabase db){ Log.*w*(*TAG*,*SCRIPT_CREATE_DATABASE*);db.execSQL(*SCRIPT_CREATE_ DATABASE*);} Καλείται όταν η βάση δεδομένων δημιουργείται για πρώτη φορά. Αυτό είναι όπου η δημιουργία των πινάκων και ο αρχικός πληθυσμός των πινάκων πρέπει να γίνει. Η db είναι η βαση δεδομένων και η επόμενη εντολή μας δημιουργεί την βάση όπως την κατασκευάσαμε πιο πάνω. Το Android παρέχει μια βοηθητική κλάση καταγραφής ημερολογίου, που καλείται android.util.Log. τα μηνύματα ημερολογίου κατατάσσονται βάσει σοβαρότητας και διεξοδικότητας και τα σφάλματα είναι τα πιο σοβαρά. Η Log.w είναι αυτή που χρησιμοποιούμε στην εφαρμογή και καταγράφει προειδοποιήσεις. Η πρώτη παράμετρος κάθε μεθόδου Log είναι μια συμβολοσειρά που καλείται σήμανση (tag). Μια συνηθισμένη προγραμματιστική πρακτική του Android είναι να ορίζει μία καθολική στατική συμβολοσειρά, που παριστά την συνολική εφαρμογή ή την συγκεκριμένη δραστηριότητα, μέσα στην εφαρμογή, έτσι μπορούμε να δημιουργήσουμε φίλτρα ημερολογίου, προκειμένου να περιορίζουν την έξοδο ημερολογίου σε συγκεκριμένα δεδομένα.

Η public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) καλείται όταν η βάση δεδομένων πρέπει να αναβαθμιστεί. Η εφαρμογή θα πρέπει να χρησιμοποιεί αυτή τη μέθοδο για να ρίξει τους πίνακες, προσθέστε τους πίνακες, ή να κάνουμε οτιδήποτε άλλο που χρειάζεται για να αναβαθμίσουν στην νέα έκδοση του σχήματος. Πάλι θα χρησιμοποιήσουμε την Log.w με σκοπό να φιλτράρουμε τα μηνύματα του ημερολογίου με βάση την συμβολοσειρά σήμανσης, για να μπορέσουμε να αναβαθμίσουμε από μια πιο παλιάς έκδοσης σε πιο νέα και καταστροφής της παλιάς.

Στη συνέχεια, καλείται η public SQLiteAdapter(Context c){this.mContext=c;} η οποία ενεργοποιεί τη βάση δεδομένων, καλώντας ένα αντικείμενο το mContext.Η

συνάρτηση public SQLiteAdapter open() την χρησιμοποιούμε για να καλέσουμε το βοηθητικό(SQLiteHelper) αντικείμενο της βάσης και μέσω αυτού καλείται getWritableDatabase() ώστε να δημιουργήσει και/ή να ανοίξει μια βάση δεδομένων που θα χρησιμοποιηθεί για ανάγνωση και γραφή, θα πρέπει να αναφέρουμε ακόμη οτι η συνάρτησή μας είναι επέκταση της android.database.SQLException, που την χρησιμοποιούμε όταν δεν μπορεί να ανοίξει η βάση μας.

Θα πρέπει να φροντίσουμε μετά την χρήση της παραπάνω συνάρτησης να την κλείσουμε για να μπορούμε να την χρησιμοποιούμε όποτε είναι αναγκαίο να την ανοίξουμε και να ξαναεγγράψουμε δεδομένα στην βάση μας γιατί σε αντίθετη περίπτωση δεν θα είναι εγγράψιμη και θα επιτρέπει μόνο το διάβασμά της. Αυτό το επιτυγχάνουμε με την public void close(){if (sqLiteHelper!= null){sqLiteHelper. close();}}
Για να τοποθετήσουμε τα στοιχεία μέσα στην βάση χρησιμοποιούμε την συνάρτηση insert(String text, String text1) καλώντας ένα ContentValues που δημιουργεί ένα σύνολο τιμών που αντιγράφονται από το σύνολο δεδομένων και με την συνάρτηση, τα τοποθετούμε μέσα στο κατασκευασμένο contentValues, είναι ένα αντικείμενο του αρχικού. Οι συναρτήσεις που τοποθετούν τα δεδομένα μέσα στην βάση δεδομένων μας, στη μια στήλη τις λίμνες και στην άλλη την επιφάνεια της κάθε μιας, είναι η contentValues.put(*KEY_CONTENT*, text) για την πρώτη στήλη και η contentValues.put(*KEY_CONTENT1*, text1) για την δεύτερη στήλη. Εν συνεχεία, τα επιστρέφει στην βάση μας και αυτό γίνεται με την εντολή return sqLiteDatabase.insert(*MYDATABASE_TABLE*, null, contentValues).

Την deleteAll() την χρησιμοποιούμε για να διαγράφουμε τα δεδομένα της βάσης, θέτουμε το doneDelete=0 και καλούμε την βάση να κλείσει μέσω της sqLiteDatabase.delete(*MYDATABASE_TABLE*, null, null) και αν η συνάρτηση επιστρέψει doneDelete>0 τότε κλείνει τη βάση. Καθώς υπάρχει και η Log.w μας εμφανίζει τη συμβολοσειρά σήμανσης για το doneDelete.
Η συνάρτηση insertLake() εισάγουμε τα δεδομένα της βάσης μας με την εντολή insert("Trixonida","95,8").

Lake

Η κλάση lake.java έχει την παρακάτω μορφή:

```
package com.example.envirinfor1;

public class Lake {
        String text = null;
        String text1 = null;


        public String getText() {
         return text;
        }
        public void setText(String text) {
         this.text = text;
```

```
        }
        public String getText1() {
         return text1;
        }
        public void setText1(String text1) {
         this.text1 = text1;
}
}
```

Το POJO σχετίζεται με το πώς πρόκειται να εφαρμόσει το σύνολο των δεδομένων που εμείς δημιουργήσαμε. Απλά είναι μια κλάση με public getters και setters για να αποκτήσουμε πρόσβαση σε αυτά τα χαρακτηριστικά. Ουσιαστικά, συνδέει το User Interface με το σύνολο δεδομένων.

6 Συμπεράσματα και Μελλοντικές επεκτάσεις

Στην μεταπτυχιακή διατριβή, αναπτύξαμε πτυχές της Περιβαλλοντικής Πληροφορικής για την προστασία του περιβάλλοντος από εξωγενείς παράγοντες που απειλούν την οικολογική ισορροπία του πλανήτη μας και βοηθά στην καλύτερη οικολογική συνείδηση των ανθρώπων μέσα από εύχρηστες κινητές συσκευές.

Σε πρώτη φάση αναπτύξαμε τα προγράμματα που χρησιμοποιήθηκαν για να τρέξουμε την εφαρμογή. Γράψαμε για το καθένα ξεχωριστά, για το πότε ξεκίνησαν, από ποιους αλλά και την ιστορική τους διαδρομή μέσα στο πέρασμα των χρόνων. Επίσης, πωλήσεις που έγιναν μεταξύ των εταιριών για να φθάσουν στην τωρινής τους μορφή καθώς και την εξέλιξη των μοντέλων αναδρομικά για την καλύτερη διεπαφή μεταξύ των συσκευών και χρηστών.

Εν συνεχεία, μελετήσαμε κάποια paper που μας βοήθησαν να καταλάβουμε πως λειτουργεί η Περιβαλλοντική Πληροφορική σε παγκόσμιο επίπεδο, με βάση συστημάτων που ανέπτυξαν καθηγητές πανεπιστημιακού επιπέδου. Τα συστήματα που αναπτύχθηκαν είχαν σαν κεντρική φιλοσοφία, ένα κεντρικό σύστημα ελέγχου όπου θα αποθηκεύονται τα δεδομένα στην βάση. Τα δεδομένα εισέρχονται μέσα στο σύστημα από περιφερειακές κινητές συσκευές που έχουν εφοδιαστεί με το κατάλληλο λογισμικό.

Στην εφαρμογή που έχουμε δημιουργήσει, χρησιμοποιήσαμε το λογισμικό Android και έχει σαν σκοπό να αποθηκεύει τις λίμνες της Ελλάδος καθώς και την επιφάνεια της καθεμιάς. Πρώτα όμως, φτιάξαμε μια κεντρική οθόνη που είναι εισαγωγική και μας οδηγεί μέσα στην εφαρμογή μας. Σε αυτή τοποθετήσαμε διάφορα λογότυπα, αλλά και κινήσεις που κάναμε για να είναι εντυπωσιακή και να προσελκύει το χρήστη. Στο κεντρικό μενού εμφανίζεται το όνομα της λίμνης, όταν κλικάρουμε πάνω στο spinner εμφανίζονται όλες οι επιλογές που διαθέτουμε, επιλέγοντας τη λίμνη της αρεσκείας μας, που στο κάτω μέρος βρίσκεται και η επιφάνειά της σε τ.χλμ. . Εφόσον επιλέξουμε την λίμνη τότε εμφανίζεται ένα μήνυμα στο κάτω μέρος της οθόνης μας που μας παρουσιάζει διεξοδικά την επιλογή μας. Τα δεδομένα μας, τα έχουμε αποθηκευμένα στην βάση δεδομένων που διαθέτει το Android, την SQLite. Πριν από όλα αυτά, για τη εύκολη χρήση της εφαρμογής αφιερώσαμε αρκετό χρόνο για να αναπτύξουμε το help ώστε να μπορεί ο χρήστη να αλληλεπιδρά εύκολα με την εφαρμογή και όχι μόνο, αλλά και την βάση μας μέσω της sqlitebrowse.

Αυτό που θα μπορούσε να γίνει, για να βελτιώσουμε την εφαρμογή και να προστατεύσουμε τις λίμνες από εξωγενείς απειλές αλλά και φυσικές καταστροφές. Να τοποθετήσουμε σένσορες σε διάφορα σημεία της λίμνης, κάμερες ώστε να υπάρχει άμεση πρόσβαση. Κάθε φορά που γίνεται παραβίαση θα ενημερώνεται η βάση δεδομένων των χρηστών. Θα έχουμε επεκτείνει τη βάση μας και εκτός από τα στοιχεία που είναι διαθέσιμα για ενημερωτικούς λόγους, θα προστίθενται δυναμικά απευθείας από τους σένσορες τα δεδομένα που συλλέγουν. Η κινητή εφαρμογή θα

ενημερώνει τον χρήστη και αυτό με τη σειρά θα τα στέλνει σε ένα κεντρικό σύστημα, όπου θα συλλέγονται και αξιολογούνται τα δεδομένα, αν είναι απειλητικά για την λίμνη θα ενημερώνονται οι κατάλληλες αρχές ώστε να επέμβει άμεσα. Τα υπόλοιπα είτε θα καταστρέφονται είτε θα αποθηκεύονται για περεταίρω έρευνες.

Βιβλιογραφία

1. Alepis, E., Virvou, M. & Kabassi, K. (2011). Location based user modelling inadaptive mobile learning for environmental awareness. In M. Escalona Cuaresma, B.Shishkov, J. Cordeiro (Eds.): ICSOFT 2011 - Proceedings of the 6th International Conference on Software and Data Technologies, Volume 1, SciTePress 2011, ISBN 978-989-8425-76-8, pp. 214-217
2. Tobias Ziep, Peter Krehahn, Volker Wohlgemuth, HTW Berlin, University of Applied Sciences, Department of Engineering II, Industrial Environmental Informatics Unit, Wilhelminenhofstraße 75A, D-12459 Berlin
3. Lauren Darcey, Shane Conder, Ανάπτυξη Εφαρμογών για το Android,Εκδόσεις Μ.Γκιουρδά
4. Paul Deitel ,Harvey Deteil, Java Προγραμματισμός, όγδοη έκδοση, Εκδόσεις Μ. Γκιούρδας
5. Lang, C., Rey, U., Wohlgemuth, V., Genz, S., & Pawlytsch, S. (2003). PAS 1025 – Austausch umweltrelevanter Daten zwischen ERP-Systemen und betrieblichenUmweltinformationssystemen. Berlin: Beuth Verlag.
6. The European Parliament and the Council of the European Union. (2006). Regulation (EC) No 842/2006 of the European Parliament and of the Council of 17 May 2006 on certain fluorinated green housegases (Text with EEA relevance).
7. Anonymous. Android platform architecture. , (2009). Viewed on May 5, 2011,http://www.android.com/media/ # # android platform-architecture-23 , visited 15 November 2010.
8. Mark H. G, Michael P. R, (2011), Smart smartphone development: iOS versus android, Proceedings of the 42nd ACM technical symposium on Computer science education, New York, NY, USA 2011
9. http://en.wikipedia.org/wiki/Geographic_information_system
10. http://android-er.blogspot.gr/2011/06/simple-example-using-androids-sqlite.html
11. http://kallurubrothers.blogspot.gr/2013/04/spinnerview-in-android.html
12. http://www.mysamplecode.com/2012/07/android-listview-cursoradapter-sqlite.html
13. http://www.iseis.org/jei/abstract.asp?no=200300001
14. http://mobiforge.com/designing/story/understanding-user-interface-android-part-3-more-views
15. http://scholar.google.gr/scholar
16. http://www.java2s.com/Code/Android/Database/InsertDataintodatabase.htm