ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΗΛΕΚΤΡΟΝΙΚΗΣ

Θέμα διπλωματικής εργασίας:

ΕΝΣΩΜΑΤΩΜΕΝΟ ΣΥΣΤΗΜΑ ΓΙΑ ΠΑΡΟΧΗ ΤΟΥΡΙΣΤΙΚΩΝ ΠΛΗΡΟΦΟΡΙΩΝ

ΚΟΝΤΟΜΑΝΟΥ ΑΙΚΑΤΕΡΙΝΗ

Εξεταστική επιτροπή:

.....	.....	.....
Παπαευσταθίου Ιωάννης Επίκουρος Καθηγητής (Επιβλέπων)	Δόλλας Απόστολος Καθηγητής	Πνευματικάτος Διονύσιος Αναπληρωτής Καθηγητής

Χανιά, Μάρτιος 2008

ΕΥΧΑΡΙΣΤΙΕΣ

Ολοκληρώνοντας τη διπλωματική αυτή εργασία, θα ήθελα να ευχαριστήσω όλους όσους μου συμπαραστάθηκαν κατά τη διάρκεια της φοίτησής μου στο Πολυτεχνείο Κρήτης.

Ιδιαίτερα θα ήθελα να ευχαριστήσω τον καθηγητή μου κ. Ιωάννη Παπαευσταθίου, για την επίβλεψη καθώς και την σημαντική καθοδήγησή του κατά την διάρκεια της διπλωματικής μου εργασίας.

Ακόμη θα ήθελα να ευχαριστήσω τους καθηγητές κ. Απόστολο Δόλλα και Διονύσιο Πνευματικάτο για τον χρόνο που αφιέρωσαν να διαβάσουν το κείμενο αυτό.

Επίσης, θα ήθελα να ευχαριστήσω τον Ερευνητή, Διπλωματούχο Ηλεκτρονικό Μηχανικό & Μηχανικό Υπολογιστών καθώς και στέλεχος του Εργαστηρίου Διανεμημένων Πληροφοριακών Συστημάτων και Εφαρμογών, Γιώργο Ανέστη, για την πολύτιμη βοήθεια, καθοδήγηση και στήριξη την οποία μου παρείχε καθ' όλη τη διάρκεια της εκπόνησης της εργασίας.

Επιπλέον, νιώθω την ανάγκη να ευχαριστήσω την οικογένειά μου – τους γονείς και τον αδερφό μου – αλλά και τους φίλους μου για την στήριξη που μου παρείχαν όλα αυτά τα χρόνια.

Κατερίνα Κοντομάνου,
Χανιά, Πολυτεχνείο Κρήτης,
Μάρτιος 2008

ΠΕΡΙΛΗΨΗ

Σκοπός της παρούσας διπλωματικής εργασίας είναι ο σχεδιασμός και η ανάπτυξη μιας εφαρμογής τουριστικού περιεχομένου για φορητές συσκευές και συγκεκριμένα για κινητά τηλέφωνα.

Πιο συγκεκριμένα, η εφαρμογή αυτή, είναι μια εφαρμογή πλοήγησης από σημείο σε σημείο ενός χάρτη και απόκτησης της σχετικής πληροφορίας του σημείου.

Ουσιαστικά, ο χρήστης της φορητής συσκευής έχει στην διάθεσή του έναν τουριστικό οδηγό της πόλης των Χανίων. Ο χάρτης αυτός φέρει πάνω του κάποια σημεία ενδιαφέροντος, τα οποία όταν επιλεγούν από το χρήστη, δίνουν την ζητούμενη χωρική πληροφορία.

Η γλώσσα που χρησιμοποιήθηκε για την ανάπτυξη της εφαρμογής είναι η J2ME (JAVA2 MICRO EDITION), η οποία αποτελεί εξέλιξη της Java που ξέρουμε μέχρι στιγμής. Η γλώσσα αυτή προσφέρει ανεξαρτησία πλατφόρμας και λειτουργικού συστήματος. Η δημιουργία του κώδικα έγινε με τη βοήθεια του εργαλείου J2ME Wireless Toolkit 2.2 και του προγράμματος NetBeans 5.5.1 .

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ : J2ME, Java, MIDlet, χωρική πληροφορία, γεωγραφικά δεδομένα, σημεία ενδιαφέροντος, πλοήγηση, κινητά τηλέφωνα.

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΙΝΑΚΕΣ – ΕΙΚΟΝΕΣ – ΔΙΑΓΡΑΜΜΑΤΑ

ΚΕΦΑΛΑΙΟ 2 : Γνωριμία με την J2ME

ΚΕΦΑΛΑΙΟ 3 : ΣΧΕΤΙΚΗ ΕΡΓΑΣΙΑ

ΚΕΦΑΛΑΙΟ 4 : ΗΛΕΚΤΡΟΝΙΚΟΣ ΤΟΥΡΙΣΤΙΚΟΣ ΟΔΗΓΟΣ ΤΗΣ ΠΟΛΗΣ ΤΩΝ ΧΑΝΙΩΝ

ΚΕΦΑΛΑΙΟ 5 : ΕΓΧΕΙΡΙΔΙΟ ΔΗΜΙΟΥΡΓΙΑΣ ΠΑΡΟΜΟΙΑΣ ΕΦΑΡΜΟΓΗΣ

1

ΕΙΣΑΓΩΓΗ

1.1 Αντικείμενο της διπλωματικής.

Η ραγδαία ανάπτυξη της τεχνολογίας και η σημαντική έξαρση του φαινομένου της ζήτησης παροχής υπηρεσιών σε όλα τα επίπεδα, έχει σαν αποτέλεσμα την δημιουργία ολοένα και περισσότερων εφαρμογών που διευκολύνουν την ζωή του ανθρώπου.

Οι φορητές συσκευές και ειδικά τα κινητά τηλέφωνα έχουν εισβάλλει στην καθημερινή μας ζωή σε τέτοιο βαθμό που δεν νοείται άνθρωπος που να μην έχει στην κατοχή του τουλάχιστον ένα από αυτά. Οι απαιτήσεις του, όσο αφορά τις εφαρμογές του τηλεφώνου που έχει στα χέρια του είναι μεγάλες. Στις μέρες μας τα κινητά τηλέφωνα είναι στην ουσία, μικρογραφίες των ηλεκτρονικών υπολογιστών με σαφώς πολύ μικρότερο επεξεργαστή και πολύ λιγότερους πόρους. Αφού λοιπόν λειτουργούν σαν μικροί ηλεκτρονικοί υπολογιστές μπορούν να δέχονται συνεχώς νέες εφαρμογές που καθιστούν τη ζωή μας πιο εύκολη και που ακόμα και οι ίδιοι οι χρήστες μπορούν να δημιουργήσουν.

Μια τέτοια εφαρμογή που καθιστά πιο εύκολη τη ζωή του χρήστη της φορητής συσκευής είναι και το αντικείμενο της παρούσας διπλωματικής εργασίας. Σκοπός αυτής της εργασίας είναι η δημιουργία ενός ηλεκτρονικού τουριστικού οδηγού της πόλης των Χανίων. Με το άνοιγμα της εφαρμογής, ο χρήστης θα μπορεί να δει στην οθόνη του κινητού του τηλεφώνου έναν αστικό χάρτη της πόλης των Χανίων. Ο χάρτης αυτός φέρει πάνω του κάποια σημεία ενδιαφέροντος χωρισμένα σε κατηγορίες. Πιο συγκεκριμένα τα σημεία αυτά αποκαλούνται σημεία ενδιαφέροντος διότι προσελκύουν το ενδιαφέρον του χρήστη είτε αυτός επισκέπτεται την πόλη για διακοπές είτε όχι. Ο χάρτης απεικονίζει που ακριβώς βρίσκονται οι έδρες των Taxi, οι στάσεις των Αστικών λεωφορείων, τα μουσεία, οι έξοδοι από την πόλη καθώς και κάποια σημαντικά σημεία της πόλης. Ο χρήστης μπορεί να πλοηγηθεί σε αυτά τα σημεία και έχει την δυνατότητα να επιλέξει ένα από αυτά έτσι ώστε να πάρει την απαραίτητη χωρική πληροφορία.

Η εφαρμογή αναπτύχθηκε με σκοπό να τρέχει σε κινητά τηλέφωνα και να είναι συμβατή με όλα τα μοντέλα και μάρκες τηλεφώνων που είναι Java - enabled. Επίσης, όπως είναι λογικό,δημιουργήθηκε με βάση τους περιορισμένους πόρους της συσκευής, που δεν είναι άλλοι από τον μικρό επεξεργαστή, την μικρή μνήμη και το μικρό μέγεθος οθόνης. Για τον λόγο αυτό χρησιμοποιήθηκε η γλώσσα προγραμματισμού Java 2 Micro Edition (J2ME) που είναι μια άλλη έκδοση της Java - ειδική για τέτοιου είδους εφαρμογές - και η οποία παρέχει ανεξαρτησία πλατφόρμας και λειτουργικού συστήματος.

1.2 Δομή της εργασίας

Ο τόμος της διπλωματικής εργασίας αποτελείται από 8 κεφάλαια. Στα κεφάλαια αυτά γίνεται αναλυτική περιγραφή της διαδικασίας η οποία ακολουθήθηκε κατά την εκπόνηση της εργασίας. Συγκεκριμένα, περιγράφονται έννοιες σχετικές με το θέμα και επιπλέον, παρουσιάζεται η γλώσσα προγραμματισμού η οποία χρησιμοποιήθηκε, τα προβλήματα που αντιμετωπίστηκαν, ο τρόπος υλοποίησης της εφαρμογής και τέλος, παρουσιάζεται ένα manual με οδηγίες προς αυτούς που θέλουν να δημιουργήσουν κάτι παρόμοιο.

Πιο αναλυτικά, το $1^{ο}$ κεφάλαιο είναι εισαγωγικό και περιγράφει σε γενικές γραμμές το αντικείμενο της εργασίας.

Στο $2^{ο}$ κεφάλαιο γίνεται μια αναφορά στη γλώσσα προγραμματισμού J2ME και παραθέτονται οι διαφορές της σε σχέση με τις άλλες εκδόσεις της Java (J2SE, J2EE) και σε επίπεδο δυνατότητας προγραμματισμού αλλά και σε επίπεδο λογισμικού.

Στο $3^{ο}$ κεφάλαιο παρουσιάζονται κάποιες παρόμοιες δουλειές που έχουν υλοποιηθεί από άλλους ερευνητές.

Στο $4^{ο}$ κεφάλαιο το οποίο είναι και το πιο εκτενές, γίνεται η ανάλυση του κώδικα και η παρουσίαση της εφαρμογής.

Το $5^{ο}$ κεφάλαιο παραθέτει ένα εγχειρίδιο με οδηγίες προς αυτούς που θέλουν να ασχοληθούν με τέτοιου είδους εφαρμογές.

Στο 6^{ο} κεφάλαιο παρουσιάζονται κάποιες προτάσεις για βελτιστοποίησεις και επεκτάσεις που μπορούν να γίνουν πάνω στην παρούσα εφαρμογή έτσι ώστε να είναι πιο εύχρηστη και λειτουργική.

Στο 7^{ο} κεφάλαιο παρουσιάζεται η βιβλιογραφία που χρησιμοποιήθηκε για την διαικπεραίωση της διπλωματικής εργασίας καθώς επίσης και κάποια χρήσιμα links.

Στο 8^{ο} κεφάλαιο παρουσιάζονται κάποια γνωστά προγραμματιστικά περιβάλλοντα τα οποία μπορεί να χρησιμοποιήσει κάποιος για την υλοποίηση μιας J2ME εφαρμογής.

2

ΓΝΩΡΙΜΙΑ ΜΕ ΤΗΝ J2ME

Στο κεφάλαιο αυτό θα γίνει μια γνωριμία με την Java 2 Micro Edition. Θα παρουσιαστεί η έννοια της J2ME, η δομή της, τα χαρακτηριστικά της, η προσφορά της, καθώς και οι διαφορές της με τις άλλες δύο εκδόσεις της Java.

2.1 Τι είναι η Java 2 Micro Edition (J2ME)

2.1.1 Ιστορική αναδρομή και ορισμός

Η **Java** μετρά 13 χρόνια ύπαρξης από τη στιγμή που παρουσιάστηκε για πρώτη φορά το 1995 από την Sun Microsystems. Πρόκειται για μια γλώσσα προγραμματισμού, ειδικά σχεδιασμένη για χρήση σε κατανεμημένα περιβάλλοντα, όπως το Internet. Δημιουργήθηκε με την προοπτική να είναι απλή και εύκολη στη χρήση, επιβάλλοντας μια ολοκληρωτικά αντικειμενοστραφή αντιμετώπιση. Τα προγράμματα της Java εξ αιτίας του γεγονότος ότι είναι συμπαγή και αξιόπιστα,μπορούν να «μεταφερθούν» σε οποιαδήποτε συσκευή είτε ασύρματα, είτε μέσω του Διαδικτύου. Στη συνέχεια, ο κώδικας του προγράμματος μπορεί να

εκτελεστεί, αρκεί να υπάρχει ήδη εγκατεστημένη μια εικονική μηχανή (Virtual Machine), ικανή να μεταφράσει τις εντολές σε εκτελέσιμο αρχείο. Αυτό σημαίνει, ότι οι ιδιαιτερότητες της κάθε αρχιτεκτονικής και πλατφόρμας, αναγνωρίζονται και αντιμετωπίζονται τοπικά, καθώς εκτελείται το πρόγραμμα. Η γλώσσα προγραμματισμού της εταιρείας Sun MicroSystems γοήτευσε τους κατασκευαστές κινητών τηλεφώνων, και έτσι θέλησαν να δώσουν στους χρήστες τους, πρόσβαση σε μια ευρεία γκάμα εφαρμογών, αλλά και να «εμπλουτίσουν» τις συσκευές τους με νέες δυνατότητες. Η Sun Microsystems έπρεπε να εξελίξει την Java, ώστε να μπορεί να ενσωματωθεί ακόμη και σε συσκευές με ανίσχυρους επεξεργαστές και ελάχιστη μνήμη. Ανταποκρινόμενη στις απαιτήσεις των καιρών, υποχρεώθηκε να παρουσιάσει τρεις διαφορετικές εκδόσεις της Java, που θα ικανοποιούσαν κάθε επιθυμία: Micro, Standard και Enterprise. Η Standard έκδοση της Java έχει ενσωματωθεί στο πυρήνα πολλών λειτουργικών συστημάτων, όπως το Linux, αλλά και στους περισσότερους σύγχρονους browsers των Windows, όπως ο Internet Explorer και ο Netscape Navigator. Η Micro έκδοση της γλώσσας προγραμματισμού έχει ήδη ενσωματωθεί σε κινητά τηλέφωνα, pagers, set-top boxes, συστήματα πλοήγησης, αλλά και φορητούς υπολογιστές χειρός.

Η πλατφόρμα της αμερικανικής εταιρείας προσφέρει όλα τα πλεονεκτήματα της Java, όπως τη «φορητότητα» του κώδικα, την ευκολία προγραμματισμού, την εγγύηση λειτουργίας του κώδικα σε όλες ανεξαρτήτως τις συσκευές, αλλά και τη συμβατότητα με την Standard και την Enterprise edition. Μια εφαρμογή, που έχει αναπτυχθεί σε **J2ME,**

μπορεί να εκτελεστεί σε οποιαδήποτε από τις προαναφερθέντες συσκευές, αρκεί οι προγραμματιστές της να μην έχουν «επέμβει» στις «ιδιαιτερότητες» του περιβάλλοντος εργασίας της. Για παράδειγμα, ένα παιχνίδι, που ενσωματώνει έγχρωμα γραφικά και ψηφιακούς ήχους, δεν θα μπορέσει να εκτελεστεί σε ένα κινητό με ασπρόμαυρη οθόνη και χωρίς τις δυνατότητες εκείνες, που θα εξασφαλίσουν την αναπαραγωγή του ήχου.

Αν δίναμε έναν ορισμό για το τι ακριβώς είναι η J2ME και ποια τα χαρακτηριστικά της αυτός θα ήταν ο εξής : ***« Η Java 2 Micro Edition (J2ME) είναι η πλατφόρμα ανάπτυξης εφαρμογών για μικρές συσκευές. Παρέχει ένα εύρωστο, ευέλικτο περιβάλλον για την ανάπτυξη εφαρμογών που τρέχουν σε μια ευρεία γκάμα από μικρές συσκευές, όπως κινητά τηλέφωνα ή PDAs. Οι εφαρμογές που γράφονται σε J2ME είναι ανεξάρτητες από τη συσκευή στην οποία τρέχουν. Το μόνο που χρειάζεται είναι η συσκευή να υποστηρίζει Java. Η J2ME έχει αναπτυχθεί σε εκατομμύρια συσκευές, υποστηρίζεται από τους περισσότερους κατασκευαστές φορητών συσκευών και χρησιμοποιείται από πολλές εταιρείες ανά τον κόσμο.»***

Ιστορικά αξίζει να αναφερθεί, ότι η Motorola ήταν η πρώτη εταιρεία που ενδιαφέρθηκε για τα πλεονεκτήματα της πλατφόρμας Java 2 Micro Edition, ενσωματώνοντας την σε κινητά τηλέφωνα, που διατέθηκαν στην αμερικανική αγορά στα τέλη του 2000. Το Accompli 008 ήταν επισήμως το πρώτο smart- phone με τη συγκεκριμένη πλατφόρμα, ενώ ακολούθησε η Nokia με το 9210 Communicator και η Siemens με το SL45i. Πλέον,

όλες σχεδόν οι συσκευές που διατίθενται στην αγορά ενσωματώνουν την J2ME ή κάποια άλλη έκδοση της πλατφόρμας Java 2.

2.1.2 Διαμορφώσεις – Προφίλ – Προαιρετικά Πακέτα

2.1.2.1 Διαμορφώσεις – Configurations

Οι ***διαμορφώσεις*** είναι προδιαγραφές που αφορούν τον καθορισμό μιας virtual machine και ενός βασικού συνόλου βιβλιοθηκών που παρέχουν τα απαραίτητα **A**pplication **P**rogramming **I**nterfaces (APIs) τα οποία μπορούν να χρησιμοποιηθούν από μια ομάδα συσκευών. Παρέχουν τη βασική λειτουργικότητα για μια γκάμα συσκευών οι οποίες έχουν παρόμοια χαρακτηριστικά. Για παράδειγμα, μια διαμόρφωση μπορεί να έχει σχεδιαστεί για συσκευές οι οποίες έχουν λιγότερο από 512 KB μνήμης και περιορισμένη συνδεσιμότητα. Η virtual machine η οποία αναφέρθηκε παραπάνω, είναι είτε μια πλήρης Java Virtual Machine (JVM) είτε ένα υποσύνολο αυτής. Το σύνολο των APIs είναι συνήθως ένα υποσύνολο των APIs της Java SE. Προς το παρόν, υπάρχουν δύο διαμορφώσεις για την J2ME οι οποίες διαχωρίζονται με βάση κάποιους περιορισμούς μνήμης. Αυτές είναι :

- η Connected Limited Device Configuration (CLDC)
- και η Connected Device Configuration (CDC).

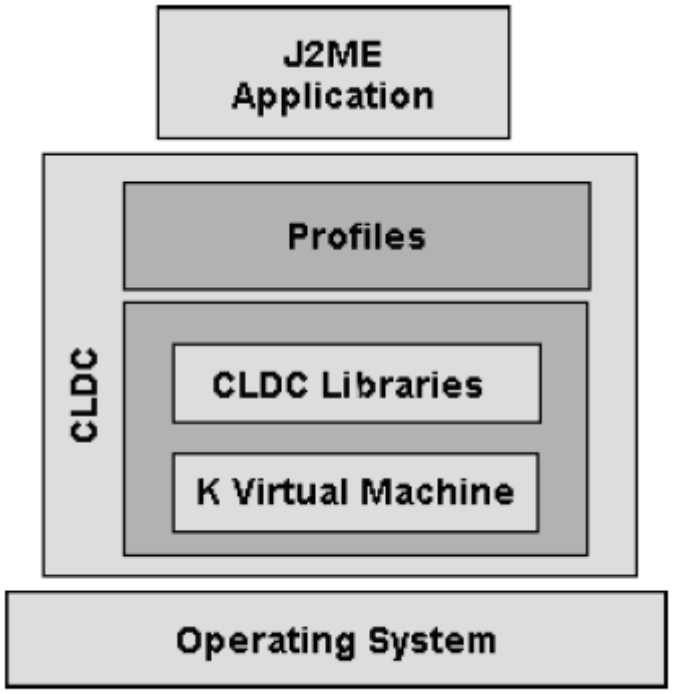
Οι δύο αυτές διαμορφώσεις αντιπροσωπεύουν και δύο διαφορετικές κατηγορίες συσκευών.

Πιο αναλυτικά :

Η *Connected Limited Device Configuration (CLDC*), υποστηρίζει προσωπικές, κινητές συσκευές, οι οποίες αποτελούν μια μικρή υποκατηγορία των συσκευών που υποστηρίζει η άλλη διαμόρφωση της J2ME, η CDC. Πρόκειται για μικρές ασύρματες συσκευές με περιορισμένη επεξεργαστική ισχύ, μνήμη και γραφική έξοδο, όπως τα κινητά τηλέφωνα ή τα PDAs. Τα χαρακτηριστικά των συσκευών που υποστηρίζει η CLDC διαμόρφωση είναι τα ακόλουθα :

- 160 – 512 KB συνολικά διαθέσιμη μνήμη για την Java Platform
- 16-bit ή 32-bit επεξεργαστής 16MHz ή ανώτερος
- Χαμηλή κατανάλωση ενέργειας (αφού η παροχή ενέργειας γίνεται με χρήση μπαταρίας)
- Παροδική συνδεσιμότητα δικτύου (συχνά ασύρματου) με πιθανώς περιορισμένο εύρος ζώνης.

Στόχος της CLDC διαμόρφωσης είναι να ορίσει μία πρότυπη Java πλατφόρμα για τις συσκευές με τα παραπάνω γνωρίσματα. Η Virtual Machine που ορίζει η παραπάνω διαμόρφωση είναι η KVM (Kilobyte Virtual Machine),η οποία ονομάζεται έτσι επειδή χρησιμοποιεί μόνο μερικά KB τρέχουσας μνήμης .



ΕΙΚΟΝΑ 2.1 : CLDC

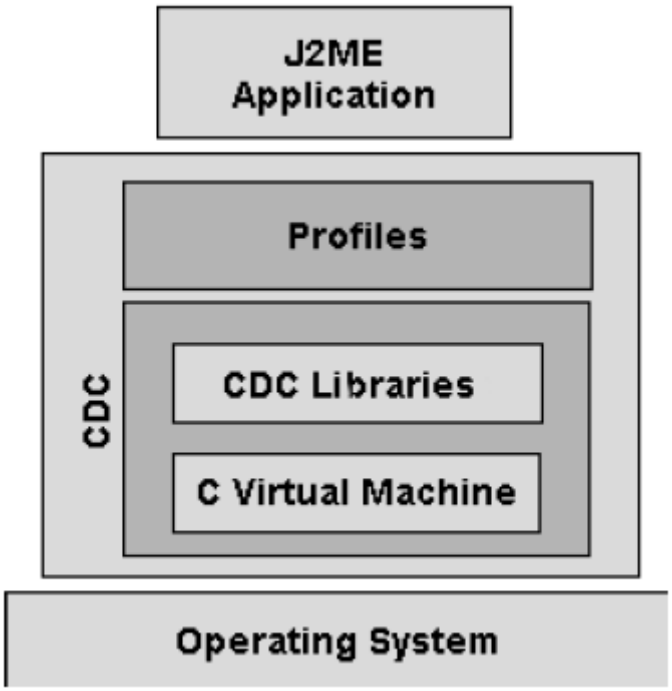
Η *Connected Device Configuration (CDC)* υποστηρίζει συσκευές που

- είναι μόνιμα συνδεδεμένες σε δίκτυο
- η συνολική διαθέσιμη μνήμη τους (συμπεριλαμβάνοντας και μνήμη RAM και μνήμη ROM) είναι 2 MB ή περισσότερο.
- Ο επεξεργαστής είναι 32-bitος.

Μεγαλύτερες συσκευές δηλαδή από πλευράς μνήμης και ταχύτητας επεξεργασίας καθώς και από πλευράς συνδεσιμότητας. Παραδείγματα τέτοιων συσκευών είναι τα set-top boxes, Internet appliances, embedded servers καθώς και “high-end mobile devices”.

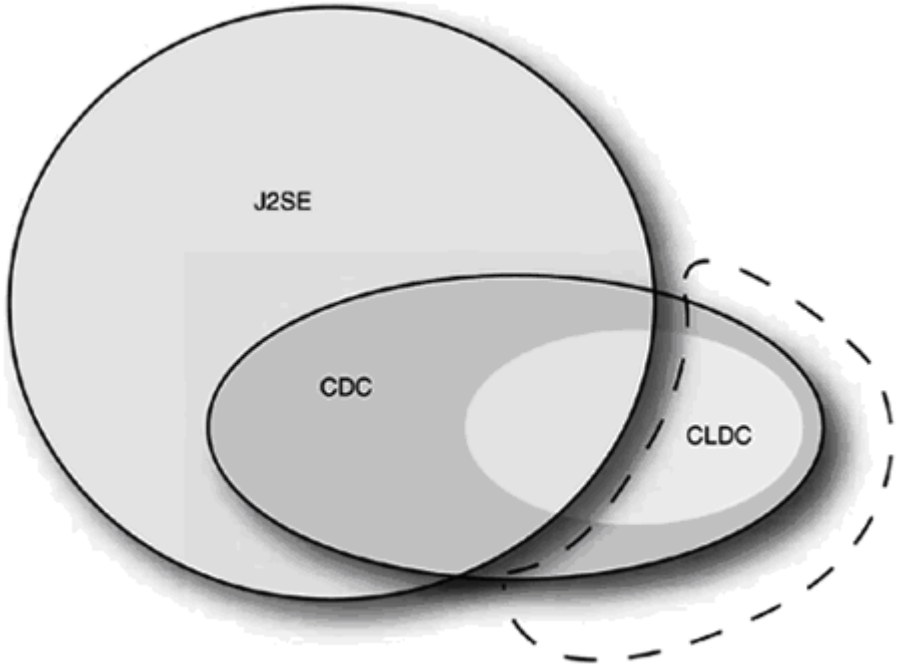
Η διαμόρφωση αυτή έχει την δυνατότητα να μας παρέχει πιο πολλά χαρακτηριστικά της Standard Edition .Επίσης το πιο ενδιαφέρον από όλα

είναι ότι ο αριθμός των APIs που περιλαμβάνονται στην CDC διαμόρφωση είναι σημαντικά μεγαλύτερος από αυτών που περιλαμβάνονται στην CLDC . Αυτό σημαίνει ότι ο χρήστης της CDC έχει πολύ μεγάλη λειτουργικότητα στην εφαρμογή που υλοποιεί.
Η CDC διαμόρφωση ορίζει την CVM (Compact Virtual Machine) .



ΕΙΚΟΝΑ 2.2 : CDC

Η CLDC διαμόρφωση είναι μεν διαφορετική από την CDC αλλά συγχρόνως αποτελεί και υποσύνολό της. Οι δύο διαφορετικές διαμορφώσεις είναι ανεξάρτητες η μία από την άλλη, παρ' όλα αυτά όμως δεν μπορούν να χρησιμοποιηθούν μαζί για να ορίσουν μία πλατφόρμα.
Η εικόνα που ακολουθεί δείχνει την σχέση μεταξύ των 2 διαμορφώσεων και της J2SE πλατφόρμας .



. ΕΙΚΟΝΑ 2.3 : Σχέση CDC, CLDC και J2SE Platform

2.1.2.2 Προφίλ – Profiles

Τα προφίλ συμπληρώνουν μια διαμόρφωση, προσθέτοντας συγκεκριμένα APIs για να δημιουργήσουν ένα runtime environment για να μπορούν να τρέχουν εφαρμογές σε μια συγκεκριμένη κατηγορία συσκευών. Ένα ***προφίλ*** είναι ένα σύνολο από APIs υψηλότερου επιπέδου τα οποία ορίζουν το μοντέλο κύκλου ζωής της εφαρμογής, τη διεπαφή χρήστη, τη μόνιμη αποθήκευση και την πρόσβαση στις ιδιαίτερες ιδιότητες μιας συσκευής. Ένα ευρέως αποδεκτό παράδειγμα είναι ο συνδυασμός της διαμόρφωσης CLDC με το προφίλ Mobile Information Device Profile (MIDP) για την παροχή μιας ολοκληρωμένου περιβάλλοντος εφαρμογών Java για κινητά τηλέφωνα και άλλες συσκευές με παρόμοια

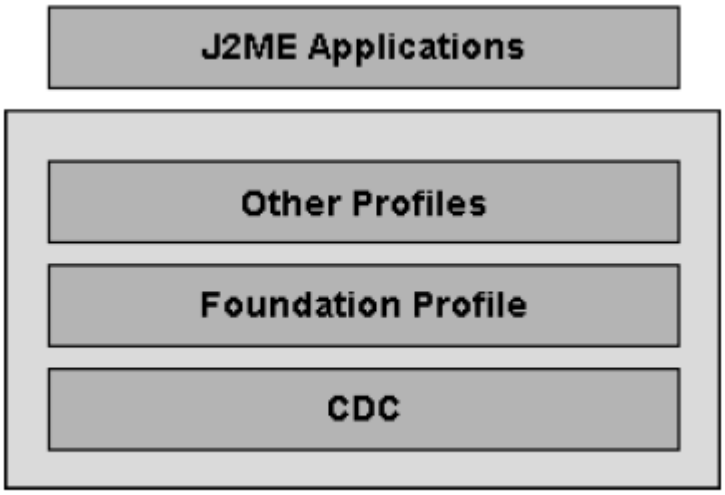
χαρακτηριστικά. Ο συνδυασμός αυτός είναι και ο συνδυασμός που χρησιμοποιήθηκε για την υλοποίηση της παρούσας εφαρμογής.

Το *Mobile Information Device Profile (MIDP)* βασίζεται πάνω στη διαμόρφωση CLDC και ήταν το πρώτο προφίλ που κατασκευάστηκε και συνεπώς το πρώτο Java ME περιβάλλον εφαρμογών. Το MIDP προσδιορίζει κινητά τηλέφωνα, ενώ οι εφαρμογές που γράφονται πάνω σε αυτό αποκαλούνται MIDlets. Η έκδοση 1.0 (JSR 37) ήταν η πρώτη έκδοση του MIDP, και προσέφερε βασικές λειτουργίες, όπως η διασύνδεση του χρήστη και η ασφάλεια του δικτύου. Η τρέχουσα έκδοση 2.0 προσφέρει βελτιωμένη διεπαφή χρήστη, δυνατότητες πολυμέσων και δημιουργία παιχνιδιών, και ακόμα βελτιωμένες δυνατότητες δικτυακής διασύνδεσης. Οι συσκευές στις οποίες στοχεύει το MIDP λέγονται Mobile Information Devices (MIDs) (π.χ. κινητά τηλέφωνα, pagers) και έχουν τα ακόλουθα χαρακτηριστικά :

- Μέγεθος οθόνης περίπου (τουλάχιστον) 96x54 pixels
- Βάθος οθόνης 1 bit
- Πληκτρολόγιο χειρός, ή χειριστήριο αφής της συσκευής
- 128 KB σταθερής μνήμης για MIDP components
- 8 KB σταθερής μνήμης για αποθήκευση δεδομένων της εφαρμογής
- Ασύρματη συνδεσιμότητα 2 δρόμων
- 32 Kb μεταβλητή runtime μνήμη για τη Java.

Το *Foundation Profile* βασίζεται πάνω στη διαμόρφωση CDC και στην ουσία αποτελεί μία επέκταση αυτής της διαμόρφωσης. Το Foundation Profile δρα σαν επέκταση της διαμόρφωσης για να πετύχει μια λειτουργικότητα παρόμοια αυτής της Java 2 Standard Edition.

Εκτός από τα MIDP και Foundation Profile υπάρχουν και κάποια άλλα προφίλ. Υπάρχει το *Personal Profile* το οποίο επίσης βασίζεται στην CDC διαμόρφωση και που παρέχει ένα περιβάλλον με πλήρη γραφική υποστήριξη. Σκοπός αυτών που το δημιούργησαν ήταν να εξασφαλίσουν μία πλατφόρμα κατάλληλη για Web applets. Τέλος υπάρχει και το RMI προφίλ που βασίζεται και αυτό στην CDC διαμόρφωση. Τα δύο τελευταία προφίλ προαπαιτούν την υλοποίηση του Foundation Profile.



ΕΙΚΟΝΑ 2.4 : Foundation Profile

Γιατί είναι αναγκαίο για την J2ME να χωριστεί σε διαμορφώσεις και προφίλ;

Η ιδέα πίσω από τις διαμορφώσεις και τα προφίλ είναι να κάνει την J2ME όσο το δυνατόν πιο ευέλικτη. Αντίθετα με τους επιτραπέζιους υπολογιστές οι οποίοι λίγο - πολύ είναι ίδιοι μεταξύ τους στα πλαίσια των βασικών ικανοτήτων τους, οι μικρές υπολογιστικές συσκευές διαφέρουν όσον αφορά το θέμα τον δυνατοτήτων τους. Για παράδειγμα οι οθόνες των φορητών συσκευών παρουσιάζουν μεγάλες διαφορές στο μέγεθός τους, ενώ υπάρχουν και συσκευές που δεν έχουν καθόλου οθόνη.

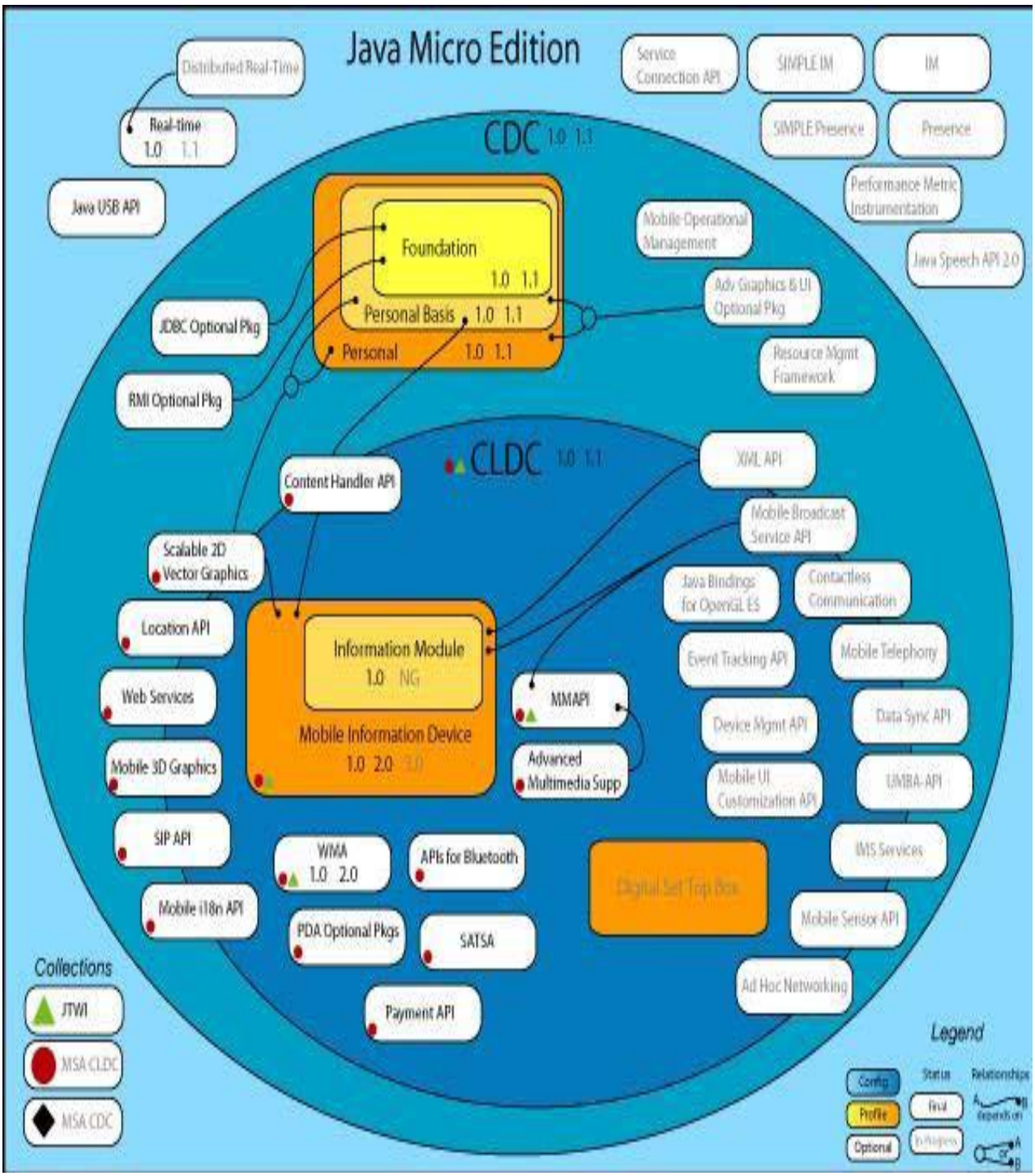
Για το λόγο αυτό, τα J2ME API χρειάζεται να είναι εξαιρετικά ευέλικτα. Προσαρτώντας τα J2ME API στις διαμορφώσεις και τα προφίλ η εταιρία Sun επιτρέπει στο API να αποκτήσει ή να χάσει κάποια χαρακτηριστικά, το οποίο εξαρτάται κάθε φορά από τον ειδικό τύπο της υπολογιστικής συσκευής. Για παράδειγμα η CLDC και το MIDP περιγράφουν ένα σετ από J2ME APIs εφαρμόσιμα σε ασύρματες φορητές συσκευές.

.

2.1.2.3 Προαιρετικά πακέτα – optional packages

Τα προαιρετικά πακέτα επεκτείνουν την πλατφόρμα Java ME προσθέτοντας λειτουργικότητα στην στοίβα τεχνολογιών η οποία περιλαμβάνει την διαμόρφωση CLDC ή CDC και ένα σχετικό προφίλ. Έχοντας δημιουργηθεί για να καλύψουν πολύ ειδικές απαιτήσεις εφαρμογής, τα προαιρετικά πακέτα προσφέρουν APIs για τη χρησιμοποίηση υπαρχόντων και νέων τεχνολογιών όπως η συνδεσιμότητα με βάσεις δεδομένων, αποστολή μηνυμάτων ασύρματα, πολυμέσα, 3Δ γραφικά, και web services.

Η τρέχουσα σχέση μεταξύ των διαμορφώσεων, προφίλ και προαιρετικών πακέτων που υπάρχουν φαίνεται στο παρακάτω διάγραμμα.

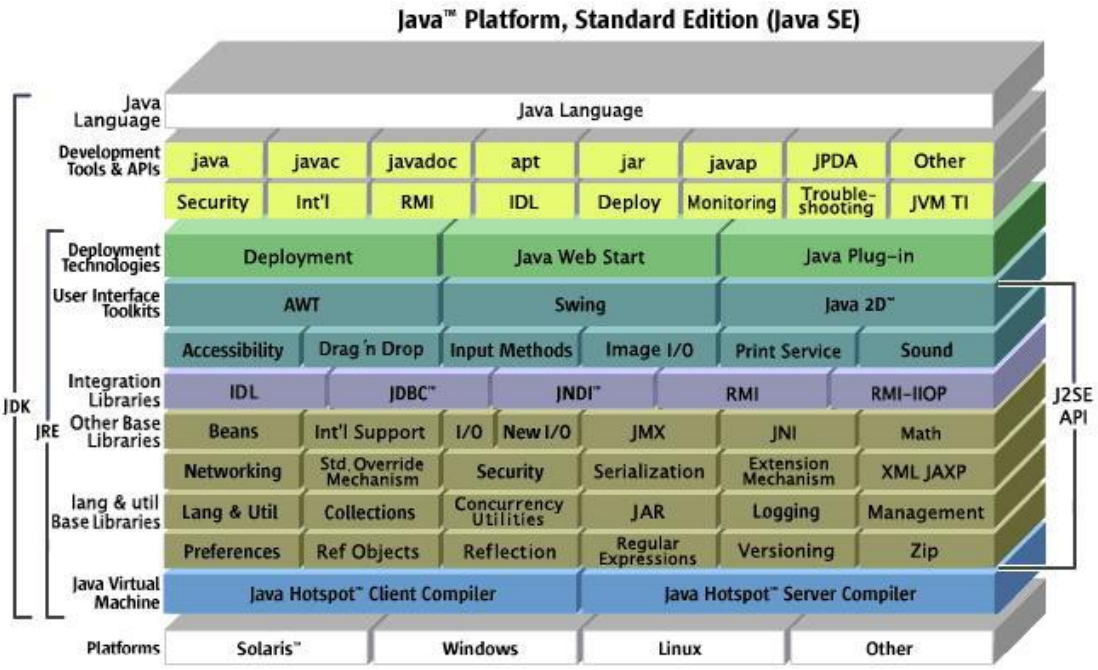


EIKONA 2.5 : Java Platform Micro Edition

2.2 Διαφορές μεταξύ των 3 διαφορετικών εκδόσεων της Java.

Όπως έχει ήδη αναφερθεί και παραπάνω η αμερικάνικη εταιρία Sun Microsystems προσφέρει την Java σε 3 διαφορετικές μορφές, κάθε μία από τις οποίες στοχεύει και σε διαφορετικό τομέα της τεχνολογίας. Το γεγονός αυτό αποτελεί το κύριο κριτήριο της διαφοροποίησής τους. Οι μορφές αυτές είναι οι εξής :

- ***J2SE (Java 2 platform Standard Edition)***. Είναι το standard πακέτο της Java το οποίο χρησιμοποιείται τόσο για εφαρμογές του Παγκόσμιου Ιστού, όσο και για κανονικές προγραμματιστικές εφαρμογές. Μας επιτρέπει να αναπτύξουμε εφαρμογές Java για desktops και servers, καθώς επίσης και εφαρμογές για ενσωματωμένα (embedded) περιβάλλοντα και περιβάλλοντα πραγματικού χρόνου (real time). Αποτελεί θεμέλιο για τη δημιουργία της Java Platform Enterprise Edition (Java EE). Οι τεχνολογίες που περιλαμβάνει η J2SE φαίνονται στο ακόλουθο σχήμα:



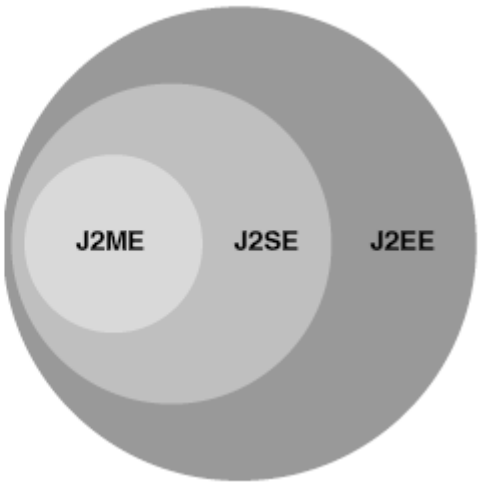
ΕΙΚΟΝΑ 2.6 : Java Platform Standard Edition

- ***J2EE (Java 2 platform Enterprise Edition)***. Το πακέτο περιλαμβάνει μια σειρά από τεχνολογίες κάτω από ενιαία αρχιτεκτονική και στοχεύει στη δημιουργία επιχειρηματικών εφαρμογών ηλεκτρονικού εμπορίου βασισμένων στον Παγκόσμιο Ιστό. Μερικές από τις τεχνολογίες που περιλαμβάνονται στην πλατφόρμα αυτή είναι οι EJB (Enterprise Java Beans), JSP (Java Server Pages), Java Servlets, Java Mail, JAXP για προσπέλαση εγγράφων XML κτλ.
- ***J2ME (Java 2 platform Micro Edition)***. Το πακέτο αποτελεί μια "ελαφρή" έκδοση για αποκλειστική χρήση σε εμπορικές συσκευές και εργαλεία, όπως έξυπνες πιστωτικές κάρτες, κινητά τηλέφωνα, εικονοτηλέφωνα, συστήματα πλοήγησης αυτοκινήτων, χειριστήρια ασύρματου ελέγχου συσκευών κτλ. Απευθύνεται δηλαδή σε συσκευές με περιορισμένη μνήμη και ισχύ επεξεργαστή.

Πρέπει επίσης να αναφερθεί ότι η Enterprise έκδοση είναι ένα υπερσύνολο της Standard έκδοσης, γεγονός το οποίο σημαίνει ότι η J2EE περιέχει όλη τη λειτουργικότητα της J2SE και επιπλέον την enterprise υπολογιστική υποστήριξη.

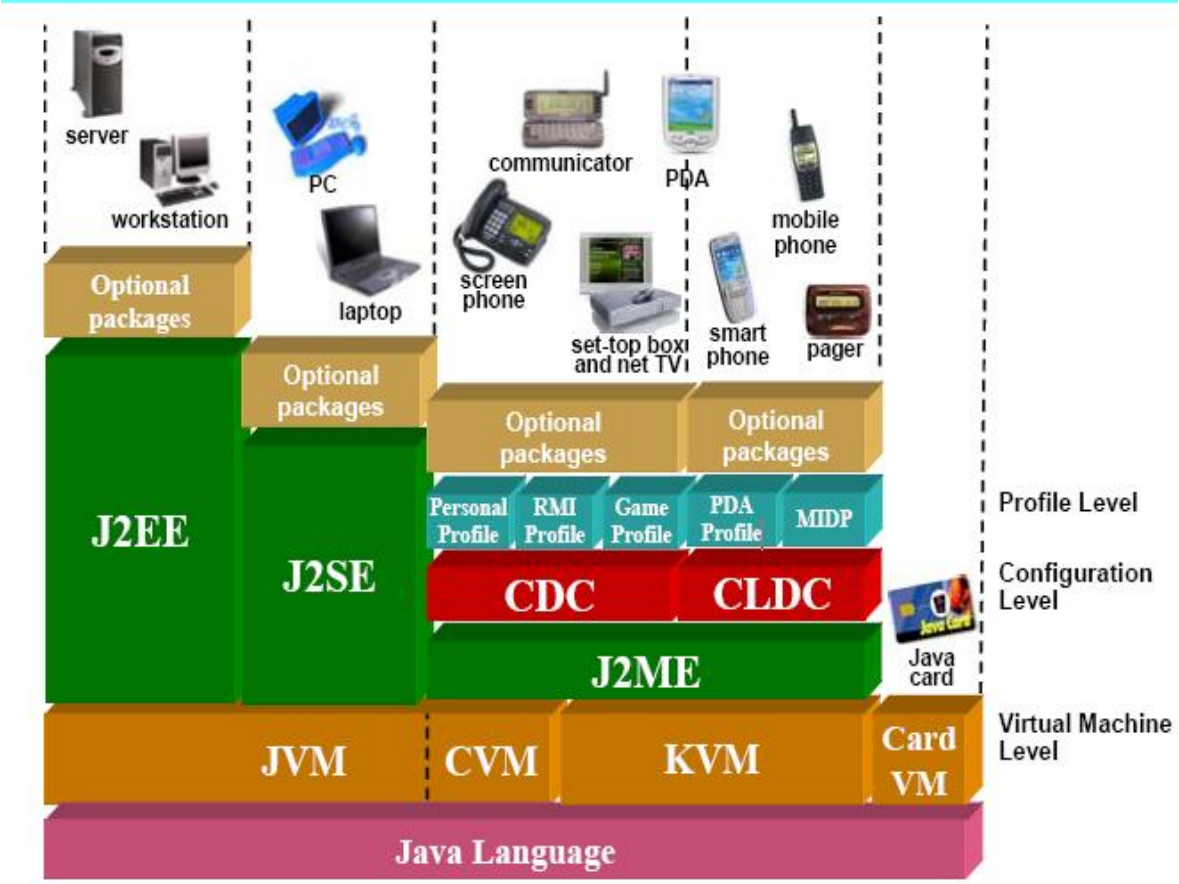
Σχηματικά η σχέση μεταξύ των τριών εκδόσεων της Java φαίνεται παρακάτω :

ΕΙΚΟΝΑ 2.7 : Σχέση μεταξύ της J2ME, J2SE, J2EE



Η επόμενη μεγάλη διαφορά ανάμεσα στις τρείς εκδόσεις της Java είναι οι εικονικές μηχανές (virtual machines). Η J2SE και η J2EE χρησιμοποιούν την Java Virtual Machine (JVM) ενώ η J2ME χρησιμοποιεί τις KVM (Kilobyte Virtual Machine) που απευθύνεται σε κινητά τηλέφωνα και CVM (C Virtual Machine) που απευθύνεται σε PDAs και πιο δυνατές συσκευές. Ο διαχωρισμός των virtual machines γίνεται με βάση τις διαμορφώσεις που περιγράφηκαν παραπάνω. Η εικόνα που φαίνεται

παρακάτω δείχνει όλες τις διαφορές ανάμεσα στις 3 εκδόσεις που περιγράφηκαν παραπάνω :



ΕΙΚΟΝΑ 2.8 : Διαφορές μεταξύ J2ME, J2SE, J2EE

Λόγω αυτής της ιδιαιτερότητας που παρουσιάζουν οι τρεις εκδόσεις, υπάρχουν περιορισμένες βιβλιοθήκες που μπορούν να χρησιμοποιηθούν. Μερικές βιβλιοθήκες που χρησιμοποιούνται και από τη J2SE/EE υπάρχουνε και στην J2ME (όπως Sockets) αλλά γενικώς μία εφαρμογή που τρέχει σε J2ME δεν τρέχει αναγκαστικά και σε J2SE.

Κάποια από τα packages της J2SE υπάρχουν και στην J2ME αλλά σαφώς περιρισμένα. Για παράδειγμα η CLDC διαμόρφωση που είναι και η διαμόρφωση που χρησιμοποιούμε ορίζει μόνο τα 4 ακόλουθα packages :

- java.io: Περιέχει αρκετά λιγότερες κλάσεις σε σχέση με το java.io package της J2SE. Περιέχει μόνο τις κλάσεις που χρειάζονται για τα data input και data output χρησιμοποιούμενα streams.
- java.lang: «υπο - package» του java.lang package της J2SE. Περιέχει μόνο τις κλάσεις που είναι βασικές για την γλώσσα Java όπως οι wrapper classes για τους τύπους των δεδομένων.
- java.util: «υπο - package» του java.util package της J2SE. Περιέχει κλάσεις όπως οι Calender, Vector, Date και Random.
- javax.microedition.io: Είναι μια καινούρια κλάση που ορίζει το Generic Connection Framework.

Επίσης παρ' όλο που αυτή η διαμόρφωση υποστηρίζει αρκετές runtime Exceptions, τα runtime Errors που υποστηρίζει είναι μόνο τα εξής :

- java.lang.Error
- java.lang.OutOfMemoryError
- java.lang.VirtualMachineError

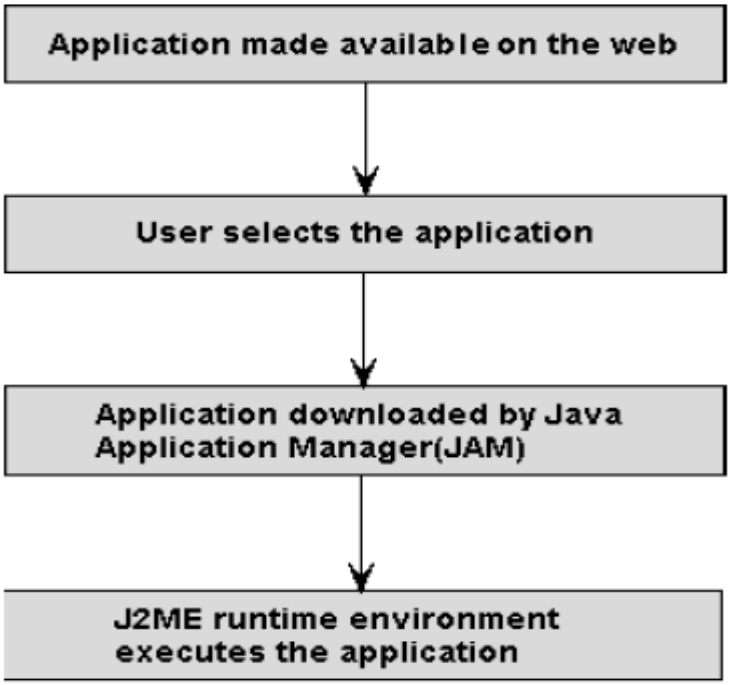
2.3 Προσφορά της J2ME

Συμπερασματικά από ότι έχει αναπτυχθεί μέχρι στιγμής, μπορούμε να πούμε ότι η Java 2 Micro Edition μας βοηθά να τρέχουμε εφαρμογές που είτε έχουμε δημιουργήσει μόνοι μας είτε κάποιοι άλλοι για μας. Η προσφορά της έγκειται στο γεγονός ότι μπορούμε να μετατρέψουμε το

κινητό μας τηλέφωνο για παράδειγμα σε έναν μικρό υπολογιστή με τις εφαρμογές της αρεσκείας μας .

Επίσης η J2ME υλοποιεί τεχνολογίες όπως η WAP, cHTML (Compact HTML) και i – mode. Μπορεί να κάνει για τις μικρές και περιορισμένες συσκευές (από άποψη μνήμης και επεξεργαστή) ότι η J2SE και η J2EE για τα τα desktop και τα servers systems. Όπως μπορούμε να κάνουμε download και να τρέξουμε ένα applet σε έναν browser που υποστηρίζει HTML ή XML, έτσι μπορούμε να τρέξουμε ένα Spotlet ή ένα MIDlet σε έναν browser που υποστηρίζει WML ή I- mode.

Με τη βοήθεια της J2ME μπορούμε να φορτώσουμε και να τρέξουμε σε μια οποιαδήποτε φορητή συσκευή,όποια εφαρμογή θέλουμε τόσο απλά όσο δείχνει το διάγραμμα που ακολουθεί :



ΕΙΚΟΝΑ 2.9 : Downloading & running a J2ME application

3

ΣΧΕΤΙΚΗ ΕΡΓΑΣΙΑ

3.1 Εισαγωγή

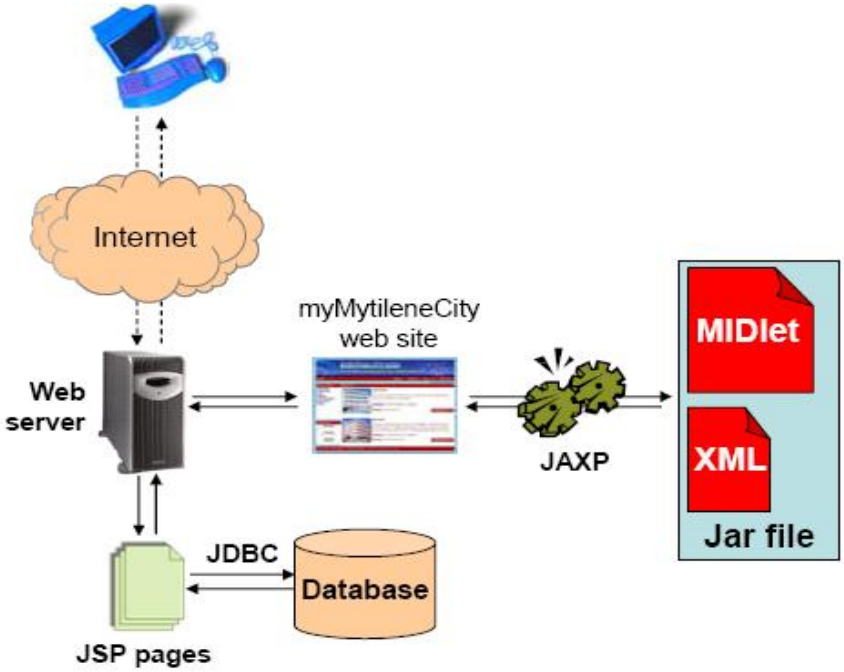
Στο κεφάλαιο αυτό θα γίνει μια αναφορά σε σχετικές εργασίες ή εφαρμογές που έχουν παρόμοιο θέμα με την παρούσα διπλωματική και έχουν αναπτυχθεί και παρουσιαστεί από άλλους ερευνητές.

3.2 Σχετικές εφαρμογές

Ο ηλεκτρονικός τουριστικός οδηγός της πόλης της Μυτιλήνης είναι δυναμική δημιουργία μιας J2ME εφαρμογής έπειτα από αλληλεπίδραση με ένα web site. Δημιουργήθηκε από μία ομάδα ερευνητών(Μιχάλης Κεντέρης, Δαμιανός Γαβαλάς, Δάφνη Οικονόμου) του Πανεπιστημίου Αιγαίου, του τμήματος Πολιτιστικής Τεχνολογίας και Επικοινωνιών, με σκοπό να αναδείξει την λειτουργικότητα, ευκολία και προσφορά της πλατφόρμας J2ME. Πρόκειται για μια εφαρμογή που τρέχει σε κινητά τηλέφωνα και στόχος της είναι να διευκολύνει αυτόν που θέλει να

επισκεφτεί την πρωτεύουσα του νησιού. Παρουσιάζει τις δυνατότητες του νησιού όσον αφορά τη διασκέδαση και τα αξιοθέατα και διαθέτει δυνατότητες πλοήγησης στο χρήστη.

Πιο αναλυτικά η σχεδίαση της εφαρμογής αυτής ακολουθεί δύο βήματα. Στο πρώτο βήμα, ο χρήστης που ενδιαφέρεται για έναν ειδικό τουριστικό προορισμό επισκέπτεται ένα web site, το οποίο περιλαμβάνει πληροφορίες σχετικά με εστιατόρια, χώρους στέγασης, αξιοθέατα, δρώμενα, κτλ. Ο χρήστης προσθέτει τις πληροφορίες του προσωπικού του ενδιαφέροντος στο «web καλάθι » του (αφηρημένη έννοια του προσωπικού λογαριασμού), το οποίο μπορεί να αποθηκεύσει και να ξαναχρησιμοποιήσει στο μέλλον. Όταν ο χρήστης γεμίσει αυτό το ειδικό καλάθι με τις πληροφορίες που θέλει να έχει στο κινητό του τηλέφωνο, το περιεχόμενο του καλαθιού μετατρέπεται σε ένα XML format. Μετά από αυτό, το σύστημα αυτόματα δημιουργεί μία εφαρμογή (ικανή να εκτελεστεί σε κινητό τηλέφωνο) ενσωματώνοντας το επιλεγμένο από τον χρήστη XML – based περιεχόμενο του καλαθιού. (Εικόνα 3.1)

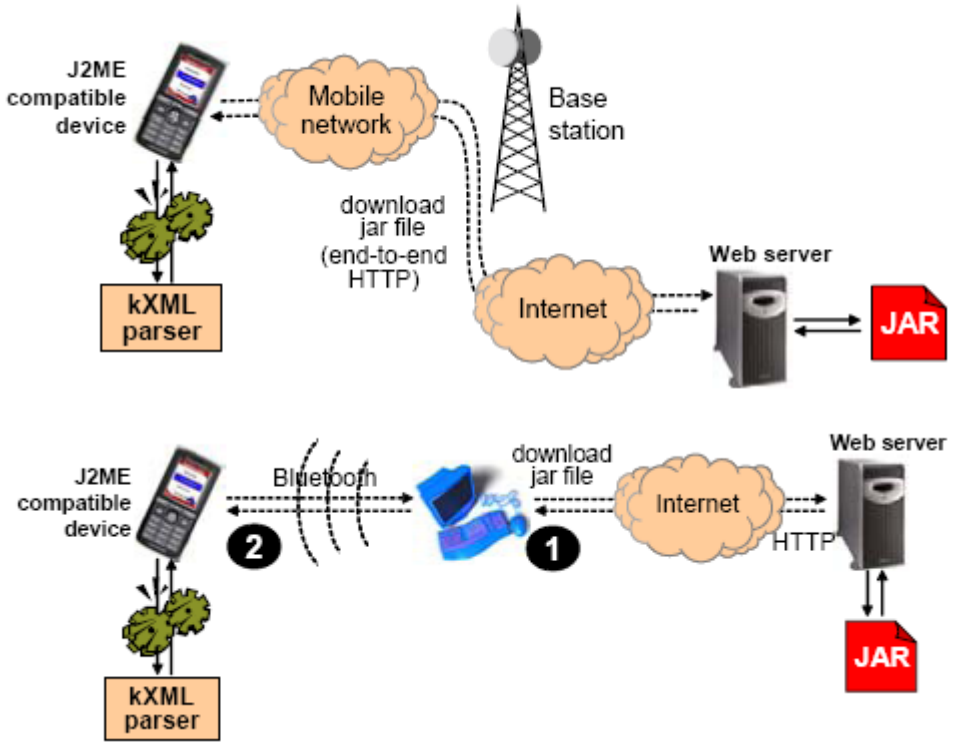


ΕΙΚΟΝΑ 3.1 : Δυναμική δημιουργία μιας J2ME εφαρμογής έπειτα από αλληλεπίδραση με το myMytileneCity web site.

Στο δεύτερο βήμα, ο χρήστης κατεβάζει το ήδη δημιουργημένο .jar αρχείο (το οποίο σώζεται προσωρινά στον web server) στην κινητή του συσκευή. Αυτό μπορεί να γίνει με τρεις τρόπους οι οποίοι περιγράφονται παρακάτω :

- ✓ έχοντας λάβει μια ειδοποίηση με ένα SMS (το οποίο περιλαμβάνεε ένα link), ο χρήστης κατεβάζει την εφαρμογή απ' ευθείας από τον web server.
- ✓ Άμεσο download του .jar αρχείου στην φορητή του συσκευή (διαμέσου end – to – end HTTP)
- ✓ Κατέβασμα του .jar αρχείου σε 2 φάσεις : πρώτα στον υπολογιστή και μετά στην κινητή συσκευή (μέσω Bluetooth ή μέσω καλωδίου).

Τα σενάρια που περιγράφηκαν παραπάνω φαίνονται και στην εικόνα που ακολουθεί :



ΕΙΚΟΝΑ 3.2 : Σενάρια download της εφαρμογής στο τηλέφωνο του χρήστη

Όταν ολοκληρωθεί η μεταφορά του .jar αρχείου στην J2ME- συμβατή συσκευή, τότε η MIDlet εφαρμογή θα μπορεί να εκτελεστεί χωρίς να υπάρχει απαραίτητη ασύρματη σύνδεση. Στο μέλλον μπορεί ο χρήστης να χρησιμοποιήσει σύνδεση για να αναβαθμίσει την εφαρμογή του.

Οι εικόνες που φαίνονται στη συνέχεια δείχνουν κάποια στιγμιότυπα της οθόνης του κινητού τηλεφώνου του χρήστη μετά το άνοιγμα της εφαρμογής. Στην εικόνα 3.3 φαίνεται ο χάρτης της Μυτηλήνης και κάποια σημεία ενδιαφέροντος, τα οποία ο χρήστης μπορεί να επιλέξει για να λάβει την απαραίτητη πληροφορία. Στην εικόνα 3.4 φαίνεται το menu της εφαρμογής με τις πιθανές επιλογές και στις εικόνες 3.5, 3.6 και 3.7 φαίνονται οι πληροφορίες που είναι καταχωρημένες για το επιλεγμένο ξενοδοχείο. (Κάτι αντίστοιχο θα παρουσιαστεί αργότερα στα πλαίσια της παρούσας διπλωματικής εργασίας.) Επίσης πρέπει να σημειωθεί ότι το

μενού και τα σημεία ενδιαφέροντος έχουν δημιουργηθεί δυναμικά όπως συμβαίνει και στην εργασία που κρατάτε στα χέρια σας.

ΕΙΚΟΝΑ 3.3

ΕΙΚΟΝΑ 3.4

ΕΙΚΟΝΑ 3.5

ΕΙΚΟΝΑ 3.6

ΕΙΚΟΝΑ 3.7

Εικόνες από προσομοιωτή κινητού τηλεφώνου που εκτελεί την myMytileneCity guide εφαρμογή.

3.3 Συμπεράσματα

Οι κύριες προοπτικές της ερευνητικής εφαρμογής που παρουσιάστηκε ήταν:

- Να καταστήσει δυνατή την δημιουργία κινητών, προσωποποιημένων τουριστικών εφαρμογών με πλούσιο και προσαρμοσμένο περιεχόμενο
- Να ελαχιστοποιήσει την απαίτηση ασύρματης συνδεσιμότητας .(Μετά το download και την εγκατάσταση της εφαρμογής στην φορητή συσκευή του χρήστη δεν απαιτείται περαιτέρω σύνδεση δικτύου. Αυτό θα γίνει μόνο σε περίπτωση αναβάθμισης της συσκευής.)
- Να τροφοδοτήσει την δυναμική εφαρμογή με καινούριο περιεχόμενο κάνοντας ελάχιστη την παρέμβαση του χρήστη.

4

ΗΛΕΚΤΡΟΝΙΚΟΣ ΤΟΥΡΙΣΤΙΚΟΣ ΟΔΗΓΟΣ ΤΗΣ ΠΟΛΗΣ ΤΩΝ ΧΑΝΙΩΝ

4.1 Εισαγωγή

Οι φορητοί ηλεκτρονικοί τουριστικοί οδηγοί έχουν εισβάλλει στην προσωπική μας ζωή με σκοπό να την διευκολύνουν. Δεν σκοπεύουν στην κατάργηση των τουριστικών οδηγών και των βιβλίων με οδηγίες ξενάγησης που όλοι ξέρουμε, αλλά στην παροχή πληροφοριών για μέρη που δεν έχουν πληθώρα έγγραφων τουριστικών οδηγών. Η αντίληψη όλων μας, λέει ότι οι φορητοί ηλεκτρονικοί τουριστικοί οδηγοί μπορούν να είναι πιο παραστατικοί, πιο πλούσιοι από άποψη περιεχομένου και να προκαλούν μεγαλύτερο ενδιαφέρον στο χρήστη από ότι οι συμβατικοί οδηγοί.

Στο κεφάλαιο αυτό θα γίνει μια εκτενής παρουσίαση της ανάλυσης και της σχεδίασης μιας τέτοιας εφαρμογής . Ο *ηλεκτρονικός τουριστικός*

οδηγός της πόλης των Χανίων είναι το αντικείμενο του παρόντος κεφαλαίου, το οποίο χωρίζεται σε δύο μέρη. Στο πρώτο μέρος, παρουσιάζεται η λειτουργία της εφαρμογής συνοδευόμενη από εικόνες που δείχνουν την οθόνη ενός προσομοιωτή κινητού τηλεφώνου στα διάφορα στάδια της εφαρμογής. Στο δεύτερο μέρος του κεφαλαίου γίνεται η ανάλυση της σχεδίασης και η ανάλυση του κώδικα της εφαρμογής.

4.2 Παρουσίαση της εφαρμογής

Στο παρόν κεφάλαιο δίνεται στον αναγνώστη με λεπτομέρεια η λειτουργία της εφαρμογής όταν εγκατασταθεί στο κινητό του τηλέφωνο και εφόσον επιλεγεί από το menu των Java εφαρμογών που είναι ήδη εγκατεστειμένες στο τηλέφωνο.

Όπως ήδη έχει αναφερθεί με το άνοιγμα της εφαρμογής, εμφανίζεται στην οθόνη του κινητού του χρήστη ένας χάρτης της πόλης των Χανίων, φέροντας πάνω του κάποια σημεία ενδιαφέροντος. Λέγοντας σημεία ενδιαφέροντος εννοούμε κάποια σημεία τα οποία όταν επιλεγούν μετέπειτα δίνουν την αντίστοιχη πληροφορία. Τα σημεία χωρίζονται σε κατηγορίες, οι οποίες διαχωρίζονται μεταξύ τους με βάση το χρώμα, το μέγεθος, την επιγραφή και την πληροφορία που περικλείουν. Έτσι με την έναρξη της εφαρμογής βλέπουμε τον χάρτη με τις εξής κατηγορίες σημείων :

- TAXI STATIONS (κίτρινο χρώμα και επιγραφή **ΤΑΧΙ)**
- MUSEUMS (ροζ χρώμα και επιγραφή **Μ**)

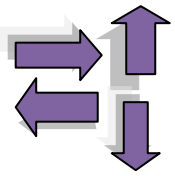
- BUS STATIONS (μπλέ χρώμα και επιγραφή **BUS**)
- INTERESTING POINTS & EXITS (κόκκινο χρώμα και επιγραφή **i**)

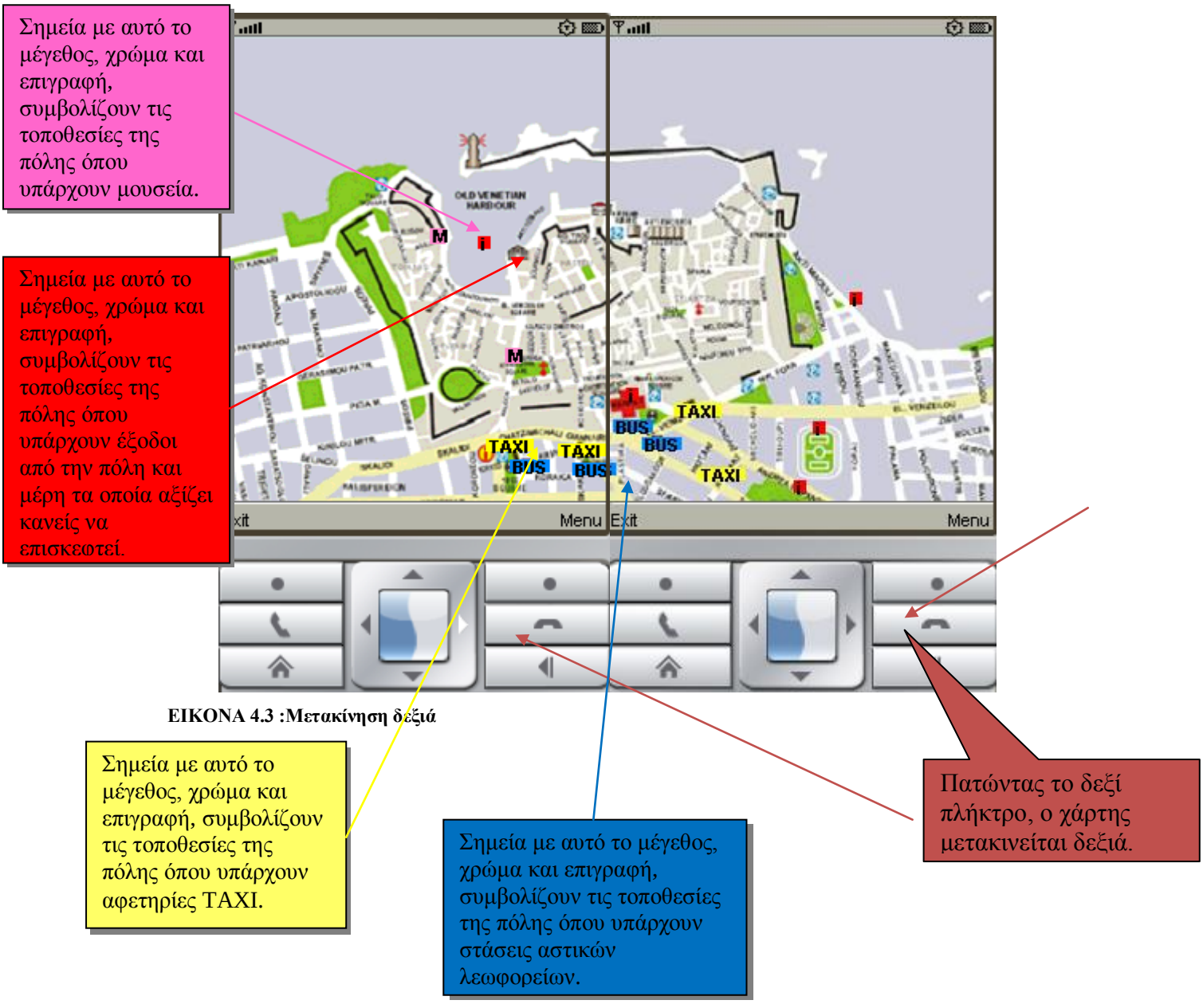
Οι εικόνες που ακολουθούν παρουσιάζουν όσα προαναφέρθηκαν.

ΕΙΚΟΝΑ 4.1 : Επιλογή της εφαρμογής

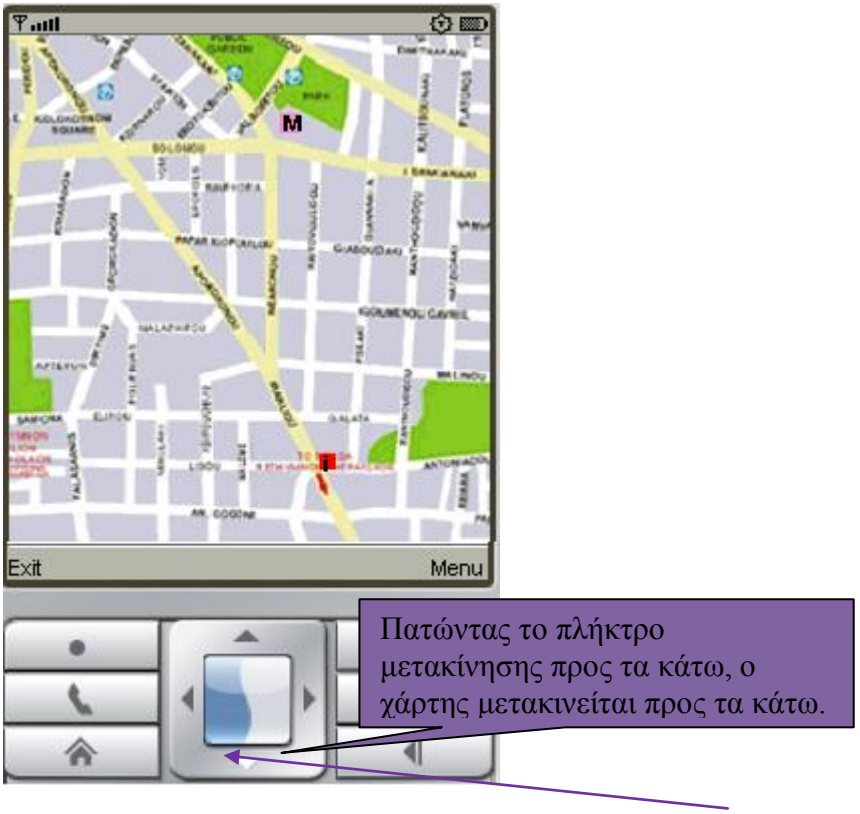
Αμέσως μόλις ανοίξει η εφαρμογή, στην οθόνη του κινητού τηλεφώνου φαίνεται ένα κομμάτι του χάρτη.

ΕΙΚΟΝΑ 4.2 : Άνοιγμα της εφαρμογής

Η περιήγηση σε όλο το χάρτη γίνεται με τα πλήκτρα  όπως φαίνεται για παράδειγμα παρακάτω :



ΕΙΚΟΝΑ 4.3 :Μετακίνηση δεξιά

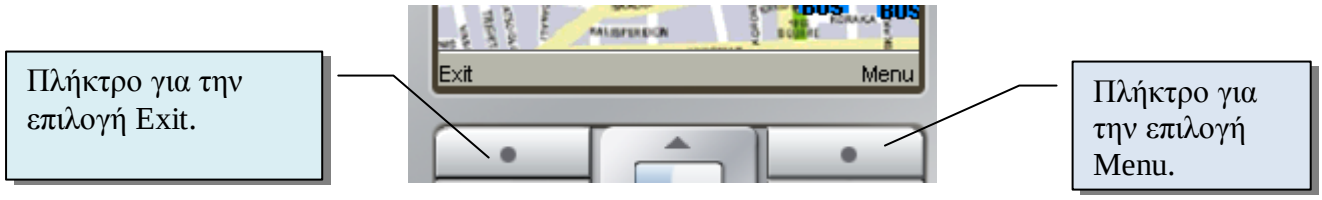


ΕΙΚΟΝΑ 4.4 : Μετακίνηση προς τα κάτω

ΕΙΚΟΝΑ 4.5 : Μετακίνηση προς τα αριστερά

ΕΙΚΟΝΑ 4.6 : Μετακίνηση προς τα πάνω

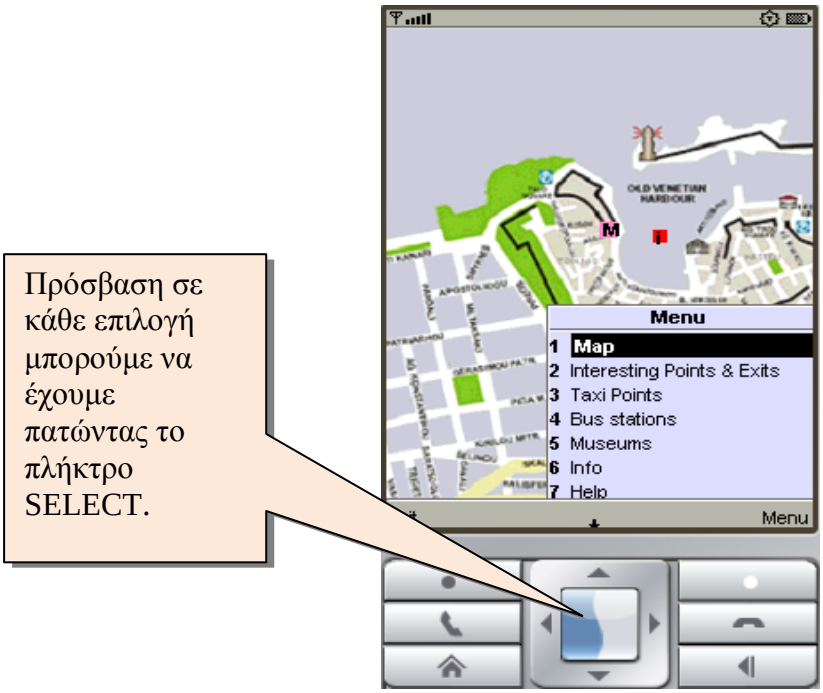
Συνεχίζοντας με την παρουσίαση της εφαρμογής, στο κάτω μέρος της οθόνης φάινονται δύο επιλογές. Η επιλογή Exit και η επιλογή Menu.



ΕΙΚΟΝΑ 4.7 : Επιλογές Exit και Menu

Όταν ο χρήστης επιλέξει την εντολή ***Exit***, επιλέγει να τερματίσει την εφαρμογή.

Όταν πατήσει το πλήκτρο που αντιστοιχεί στην εντολή ***Menu,*** τότε ανοίγει το μενού της εφαρμογής και οδηγείται σε έναν αριθμό από επιλογές και στην κατάσταση που δείχνει η επόμενη εικόνα :



ΕΙΚΟΝΑ 4.8 : Άνοιγμα του μενού της εφαρμογής

1. Η πρώτη επιλογή ονομάζεται ***Map.*** Όταν επιλεγεί από το χρήστη, εμφανίζει το χάρτη με όλες τις κατηγορίες των σημείων που περιγράφηκαν προηγουμένως. Μια απλή περιήγηση στο χάρτη είναι ό,τι μπορεί να κάνει ο χρήστης πατώντας αυτήν την επιλογή.
2. Η δεύτερη επιλογή ονομάζεται ***Interesting Points & Exits*** . Με αυτήν την επιλογή μπορούμε να περιηγηθούμε μόνο στα σημεία του χάρτη που ανήκουν στην κατηγορία αυτή και περιλαμβάνουν, αξιοθέατα και εξόδους από την πόλη. Επίσης δίνεται και η δυνατότητα επιλογής σημείου. Η οθόνη δείχνει το κομμάτι του χάρτη που περιέχει το πιο αριστερό "Intersting Point &Exit" σημείο, το οποίο και γίνεται πράσινο. Γενικά πρέπει να σημειωθεί ότι το πράσινο είναι το χρώμα επιλογής σημείου. Η μετακίνηση μεταξύ των σημείων γίνεται πατώντας τα πλήκτρα -> και <- . Για να

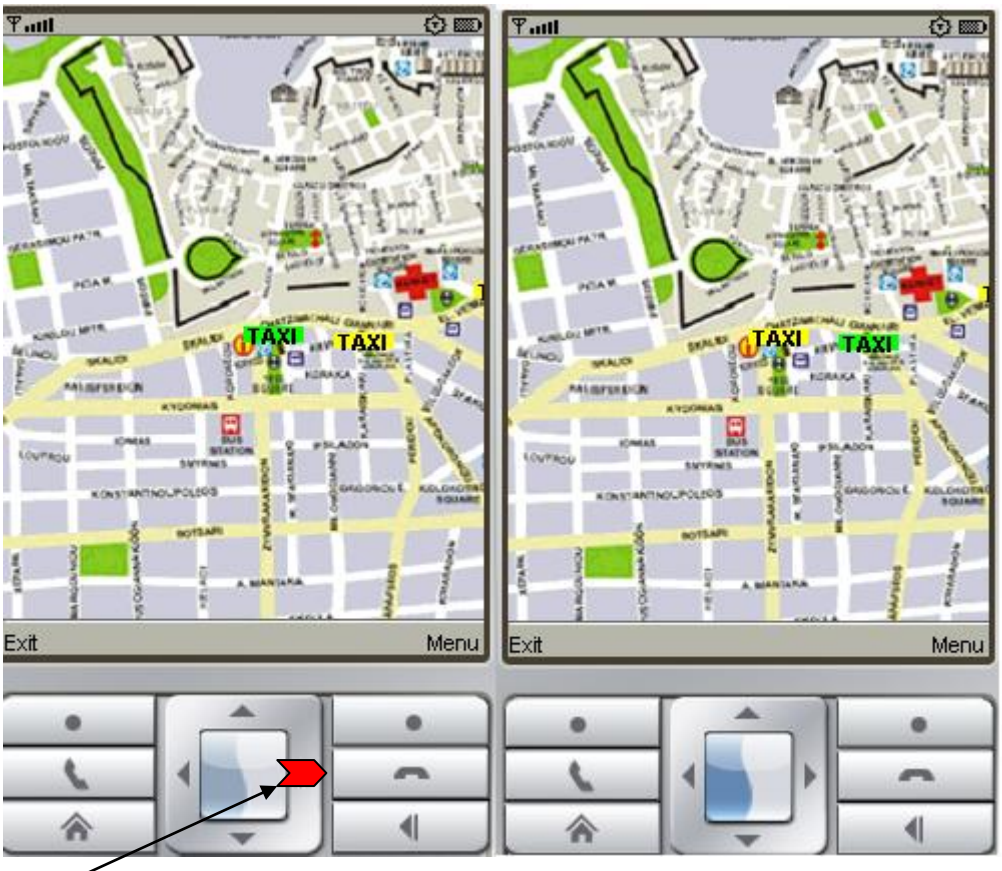
μην υπάρχουν συγχύσεις θεωρήθηκε οτι η μετακίνηση απο σημείο σε σημείο γίνεται μόνο πατώντας τα πλήκτρα δεξιάς και αριστερής μετακίνησης. Για να γίνει πιο κατανοητή αυτή η δυνατότητα, παρουσιάζονται η επόμενη εικόνα και τα σχόλια που τη συνοδεύουν :

Όταν πρασινίσει το σημείο, θεωρείται ότι το σημείο αυτό έχει επιλεγεί για να δούμε αργότερα κάποιες πληροφορίες που είναι καταχωρημένες.

Όταν πατήσουμε το πλήκτρο της δεξιάς μετακίνησης τότε μεταφερόμαστε από το σημείο που ήμασταν στο αμέσως πιο κοντινό σημείο που βρίσκεται δεξιά του. Το προηγούμενο σημείο ανακτά το παλιό του χρώμα και το καινούριο έχει πρασινίσει και συνάμα επιλεγεί.

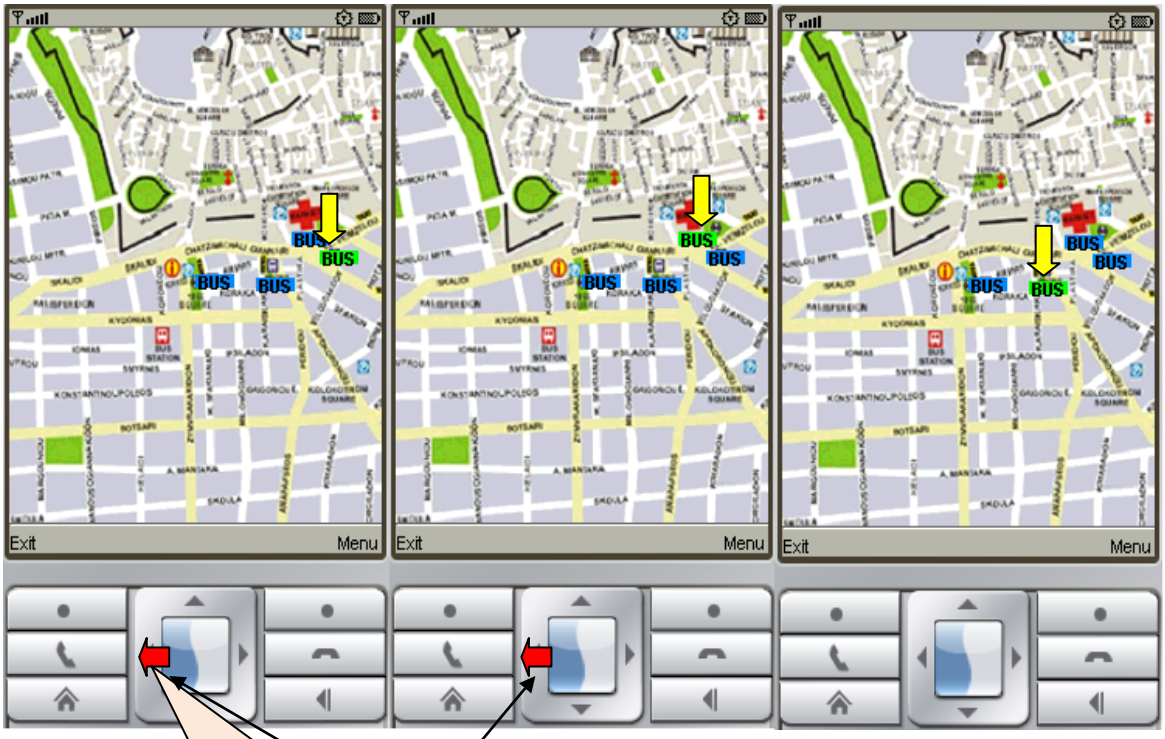
ΕΙΚΟΝΑ 4.9 : Επιλογή Interesting Points & Exits.

3. Η τρίτη επιλογή ονομάζεται ***Taxi Points***. Με αυτήν την επιλογή μπορούμε να περιηγηθούμε μόνο στα σημεία του χάρτη που ανήκουν στην κατηγορία αυτή και περιλαμβάνουν, τις τοποθεσίες της πόλης όπου υπάρχουν έδρες Taxi. Επίσης και εδώ, δίνεται δυνατότητα επιλογής σημείου. Η οθόνη δείχνει το κομμάτι του χάρτη που περιέχει το πιο αριστερό "Taxi Point" σημείο, το οποίο και γίνεται πράσινο. Εξακολουθεί να ισχύει ο ίδιος τρόπος περιήγησης όπως και στην προηγούμενη επιλογή.



ΕΙΚΟΝΑ 4.10 : Επιλογή Taxi Points

4. Η τέταρτη επιλογή ονομάζεται ***Bus Stations***. Με αυτήν την επιλογή μπορούμε να περιηγηθούμε μόνο στα σημεία του χάρτη που ανήκουν στην κατηγορία αυτή και που περιλαμβάνουν, τις τοποθεσίες της πόλης όπου υπάρχουν στάσεις αστικών λεωφορείων. Επίσης και εδώ, δίνεται δυνατότητα επιλογής σημείου. Η οθόνη δείχνει το κομμάτι του χάρτη που περιέχει το πιο αριστερό "Bus Station" σημείο, το οποίο και γίνεται πράσινο. Εξακολουθεί να ισχύει ο ίδιος τρόπος περιήγησης όπως και στις προηγούμενες επιλογές.



ΕΙΚΟΝΑ 4.11 : Επ[illegible]ς Stations

5. Η πέμπτη επιλογή ονομάζεται ***Museums***. Με αυτήν την επιλογή μπορούμε να περιηγηθούμε μόνο στα σημεία του χάρτη που ανήκουν στην κατηγορία αυτή και που περιλαμβάνουν, τις τοποθεσίες της πόλης όπου υπάρχουν μουσεία. Επίσης και εδώ, δίνεται δυνατότητα επιλογής σημείου. Η οθόνη δείχνει το κομμάτι του χάρτη που περιέχει το πιο αριστερό "Museums" σημείο, το οποίο και γίνεται πράσινο. Εξακολουθεί να ισχύει ο ίδιος τρόπος περιήγησης και επιλογής σημείου όπως και προηγουμένως.

ΕΙΚΟΝΑ 4.12 : Επιλογή Museums

6. Η έκτη επιλογή έχει να κάνει με την πληροφορία που είναι καταχωρημένη για κάθε σημείο, της κάθε κατηγορίας και ονομάζεται ***Info.*** Οταν ο χρήστης επιλέξει κάποιο σημείο της αρεσκείας του, με την διαδικασία που έγινε ήδη γνωστή, τότε πατώντας άπο το menu των επιλογών της εφαρμογής, την εντολή Info, θα εμφανιστεί στην οθόνη του κινητού τηλεφώνου ένα μήνυμα με τις ζητούμενες πληροφορίες, όπως δείχνει η εικόνα που ακολουθεί :

ΕΙΚΟΝΑ Έχουμε επιλέξει ένα σημείο της κατηγορίας Museums.

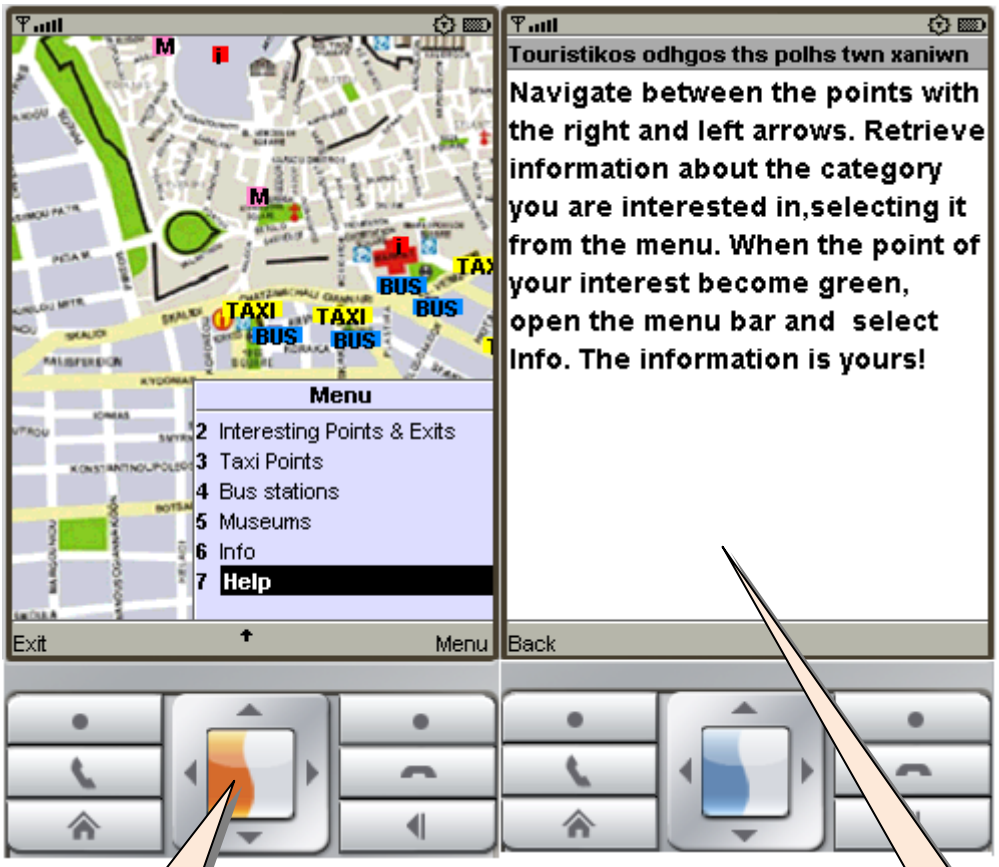
Επιλογή απο το Menu bar της εντολής Info.

Πληροφορία που είναι καταχωρημένη για το επιλεγμένο σημείο.

Εκτός από την πληροφορία που αφορά το επιλεγμένο σημείο, στην οθόνη του κινητού τηλεφώνου φαίνεται μία νέα εντολή που ονομάζεται ***Back*** και η οποία όταν τεθεί σε λειτουργία επαναφέρει το χρήστη στην προηγούμενη κατάσταση της οθόνης :

ΕΙΚΟΝΑ 4.14 : Εντολή Back

7. Η έβδομη και τελευταία προς το παρόν εντολή του Menu επιλογών της εφαρμογής ονομάζεται Help. Όπως δηλώνει και το όνομά της όταν την επιλέξει ο χρήστης, εμφανίζεται στην οθόνη της συσκευής ένα κείμενο που βοηθά τον χρήστη να καταλάβει τι πρέπει να κάνει και με ποιο τρόπο :



ΕΙΚΟΝΑ 4.15 : Ε[illegible] Help

Επιλογή της εντολής Help.

Εμφάνιση του κειμένου βοηθείας.

Καθώς τελείωσε η περιγραφή της λειτουργίας της εφαρμογής, πρέπει να αναφερθεί ότι η εφαρμογή έχει δημιουργηθεί με δυναμικό τρόπο, γεγονός που σημαίνει ότι το μενού των επιλογών μπορεί να διαφοροποιηθεί αρκετά η και να αλλάξει τελείως σε σχέση με αυτό που μόλις παρουσιάστηκε. Η δυναμική υλοποίηση μας προσφέρει την δυνατότητα εισαγωγής στο μενού και επομένως και στην λειτουργία της εφαρμογής καινούριων κατηγοριών σημείου, με έναν «εύκολο τρόπο», ο οποίος θα αναπτυχθεί με λεπτομέρεια στις σελίδες που ακολουθούν. Οι κατηγορίες των σημείων που περιγράφηκαν παραπάνω, ήταν ενδεικτικές για να καταλάβει ο αναγνώστης τι θα αντιμετωπίσει όταν κατεβάσει, εγκαταστήσει και ανοίξει την παρούσα εφαρμογή. Έτσι λοιπόν αν θέλουμε π.χ. να προσθέσουμε τους χώρους στάθμευσης οχημάτων στον χάρτη που έχουμε στην οθόνη μας, μπορούμε με έναν «μαγικό τρόπο» που περιορίζεται μόνο στην εισαγωγή κάποιων στοιχείων σε ένα αρχείο να έχουμε τα αποτελέσματα που ακολουθούν :

ΕΙΚΟΝΑ 4.16 : Εισαγωγή της κατηγορίας Parkings.

4.3 Ανάλυση του κώδικα της εφαρμογής

Πριν την ανάλυση του κώδικα της εφαρμογής και για να γίνει κατανοητή αυτή σε κάθε αναγνώστη, γίνεται στις παραγράφους που ακολουθούν μία εξοικείωση με κάποιες έννοιες που είναι βασικές για κάθε MIDlet και που ακολουθούν όλες οι εφαρμογές που δημιουργούνται.

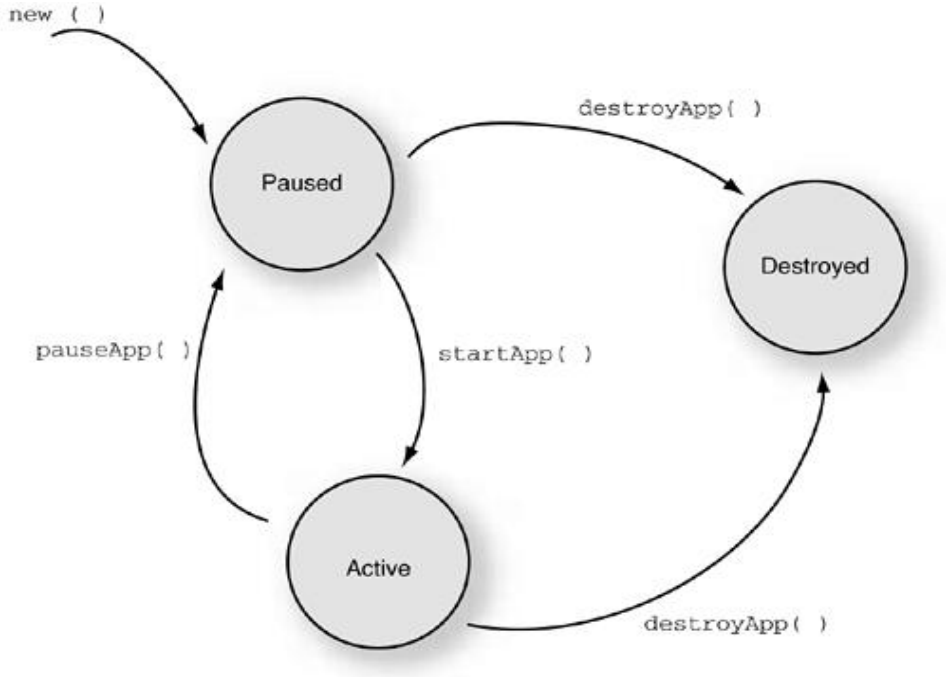
4.3.1 Το Λογισμικό Διαχείρισης Εφαρμογής - Application Management Software (AMS)

Όλες οι J2ME εφαρμογές – MIDlets και άλλες- τρέχουν κάτω από τον έλεγχο μιας Java Virtual Machine. Σε ένα κινητό τηλέφωνο δεν υπάρχει ένα shell εντολών με το οποίο μπορεί να θέσει κανείς σε λειτουργία μια Java εφαρμογή. Η έναρξη, ο τερματισμός και η διαχείριση της εκτέλεσης μιας J2ME εφαρμογής ελέγχεται από το *Application Management Software* (AMS) το οποίο υπάρχει στην συσκευή. Στην πραγματικότητα, το AMS ελέγχει ολόκληρο τον κύκλο ζωής της εφαρμογής, από την εγκατάσταση και την αναβάθμισή της εως την διαγραφή της .

4.3.2 Το μοντέλο κατάστασης ενός MIDlet

Όλα τα MIDlets μεταφέρονται σε διαφορετικές καταστάσεις κατά την διάρκεια της ζωής τους. Οι προδιαγραφές που θέτει το προφίλ MIDP, καθορίζουν και τις καταστάσεις μετάβασης του MIDlet. Η εικόνα που ακολουθεί, δείχνει ένα διάγραμμα καταστάσεων μετάβασης που αναπαριστά αυτές τις καταστάσεις του MIDlet καθώς και τα γεγονότα

που προκαλούνται εξ' αιτίας αυτής της μεταφοράς .



ΔΙΑΓΡΑΜΜΑ 4.1 : Το μοντέλο κατάστασης ενός MIDlet

Οι μέθοδοι startApp(), pauseApp() και destroyApp() επιτρέπουν στο MIDlet να αλλάξει την κατάσταση στην οποία βρίσκεται. Στην πραγματικότητα το application management software (AMS) κάνει τις απαραίτητες μεταβάσεις καλώντας αυτές τις μεθόδους. Ενα MIDlet από μόνο του δεν είναι ικανό να κάνει μετάβαση από την μία κατάσταση στην άλλη, παρόλο που μπορεί να ζητήσει αλλαγή κατάστασης από το AMS.

Όπως φαίνεται και στο διάγραμμα οι καταστάσεις στις οποίες μπορεί να μεταβεί ένα MIDlet είναι τρείς :

- **Paused:** το MIDlet δεν εκτελείται. Δεν μπορεί να εκτελεστεί μέχρι να μεταβεί σε active κατάσταση.

- **Active:** το MIDlet είτε είναι έτοιμο να εκτελεστεί, είτε εκτελείται. Το νήμα που ελέγχει το MIDlet ίσως να μην είναι σε active κατάσταση, αλλά το MIDlet είναι ακόμη ενεργό.
- **Destroyed:** το MIDlet δεν εκτελείται και δεν μπορεί άλλο πια να μεταβεί σε άλλες καταστάσεις.

Όταν το AMS δημιουργήσει ένα MIDlet, θεωρείται ότι το MIDlet βρίσκεται στην paused κατάσταση.

Ο πίνακας που ακολουθεί παρουσιάζει την λειτουργία των μεθόδων που χρησιμοποιούνται για την μετάβαση στις 3 καταστάσεις και τη δουλειά του AMS.

Όνομα μεθόδου της κλάσης MIDlet	**Περιγραφή**
protected abstract void destroyApp()	Το AMS δίνει σήμα στο MIDlet να σταματήσει.Το MIDlet μπαίνει στην destroyed κατάσταση.
void notifyDestroyed()	Το MIDlet ζητά να μπει στην destroyed κατάσταση.
void notifyPaused()	Το MIDlet ζητά να γίνει ανενεργό και να μπεί στην paused κατάσταση.
protected abstract void pauseApp()	Το AMS στέλνει σήμα στο MIDlet να σταματήσει : το MIDlet θα μπεί στην paused κατάσταση.
void resumeRequest()	Το MIDlet να ξαναμπεί στην active κατάσταση.
protected abstract void startApp()	Το AMS στέλνει σήμα στο MIDlet ότι είναι πλέον ενεργό.

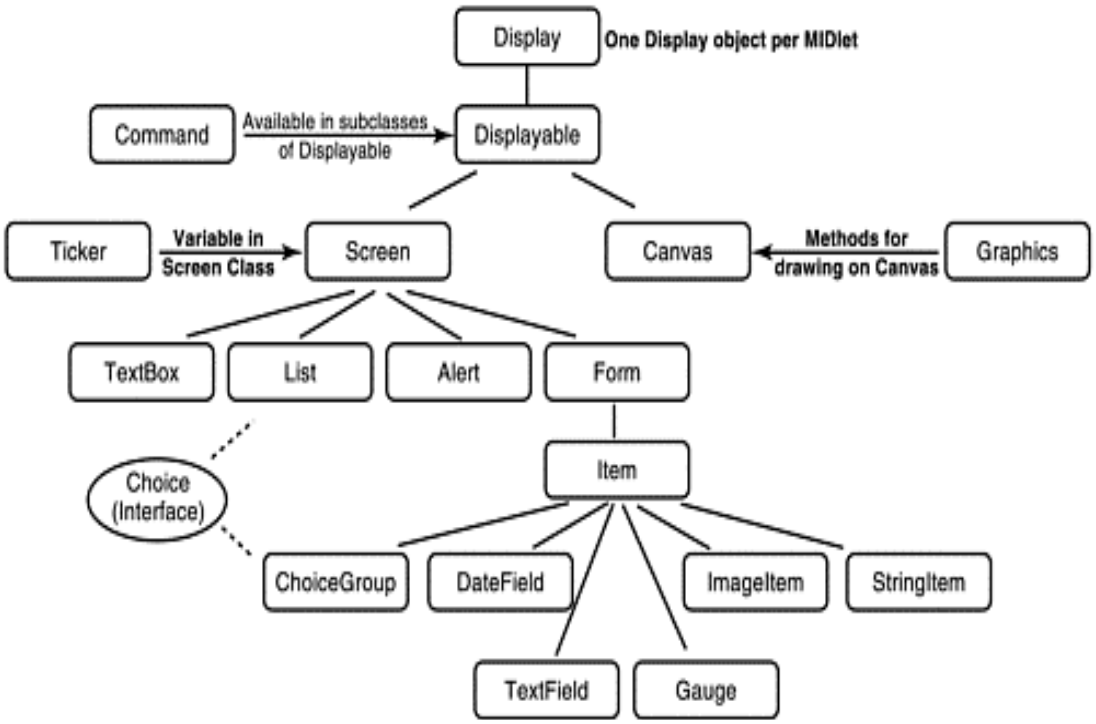
ΠΙΝΑΚΑΣ 4.1 : Μέθοδοι αλλαγής κατάστασης ενός MIDlet.

4.3.3 Οι έννοιες Display και Displayable

Για να προχωρήσουμε παρακάτω καλό θα ήταν να οριστούν και να διαφοροποιηθούν οι έννοιες **Display** και **Displayable**.

Η **Display** είναι μια κλάση η οποία ανήκει στο package java.lang.Object. Ένα αντικείμενο μιας τέτοιας κλάσης διαχειρίζεται όλα τα Displayable συστατικά. Στην πραγματικότητα είναι αυτό που «θέτει σε εμφάνιση» στην οθόνη του κινητού τηλεφώνου (και γενικά μιας φορητής συσκευής) ένα Displayable component. Αξιοπρόσεκτο είναι το γεγονός ότι μόνο ένα αντικείμενο Display υπάρχει σε κάθε MIDlet.

Η **Displayable** είναι μία αφηρημένη κλάση, η οποία επίσης ανήκει στο package java.lang.Object , και ο σκοπός της είναι διαφορετικός από αυτόν της κλάσης Display. Καθορίζει τι θα εμφανίζεται στην οθόνη κάθε δεδομένη χρονική στιγμή. Φαίνεται ότι υπάρχει μια συγγενική σχέση μεταξύ των Display και Displayable, αλλά επειδή όπως αναφέρθηκε έχουν διαφορετικούς σκοπούς, δεν υπάρχει καμία σχέση κληρονομικότητας. Ενα MIDlet μπορεί να έχει πολλά διαφορετικά Displayable components αλλά μόνο ένα Displayable αντικείμενο μπορεί να εμφανίζεται στην οθόνη κάθε δεδομένη χρονική
στιγμή. Παρακάτω φαίνεται η ιεραρχία των Displayable κλάσεων (που αναφέρθηκαν παραπάνω ως components).



ΔΙΑΓΡΑΜΜΑ 4.2 : Ιεραρχία Displayable κλάσεων.

Στο διάγραμμα φάινεται ότι μπορούν να υπάρξουν δύο κατηγορίες Displayable αντικειμένων σε κάθε MIDlet. Η **Screen** και η **Canvas**.

Η **Screen** κλάση από μόνη της δεν είναι κάτι που μπορεί να δει κάποιος. Είναι μία parent αφηρημένη κλάση για τα συστατικά που έχουν πραγματική εμφάνιση και φαίνονται στην οθόνη όπως είναι τα TextBox, List, Alert, Form.

Η **Canvas** είναι μία κλάση που περιέχει μεθόδους για την δημιουργία γραφικών και για τον χειρισμό γεγονότων όπως για παράδειγμα το πάτημα ενός πλήκτρου της φορητής συσκευής.

```
Display (public class Display)
Displayable (public abstract class Displayable)
```

Screen (public abstract class Screen extends Displayable)
TextBox (public class TextBox extends Screen)
List (public class List extends Screen implements Choice)
Alert (public class Alert extends Screen)
Form (public class Form extends Screen)
Item (public abstract class Item)
ChoiceGroup (public class ChoiceGroup extends Item implements Choice)
DateField (public class DateField extends Item)
TextField (public class TextField extends Item)
Gauge (public class Gauge extends Item)
ImageItem (public class ImageItem extends Item)
StringItem (public class StringItem extends Item)
Canvas (public abstract class Canvas extends Displayable)
Command (public class Command)
Ticker (public class Ticker)
Graphics (public class Graphics)
Choice (public interface Choice)

Όπως φαίνεται και στο διάγραμμα η **Form** είναι μια διακριτή υποκλάση της Screen, η οποία χρησιμοποιήθηκε και στην παρούσα εφαρμογή. Είναι η μοναδική κλάση που μπορεί να συναθροίζει αντικείμενα, και αυτό την κάνει ένα είδος container. Μπορεί να εμπεριέχει τρείς τύπους αντικειμένων : Strings, Images και Items. Δεν μπορεί να εμπεριέχει άλλο αντικείμενο Displayable κάθε είδους, ούτε καν αντικείμενο Screen ή Form.

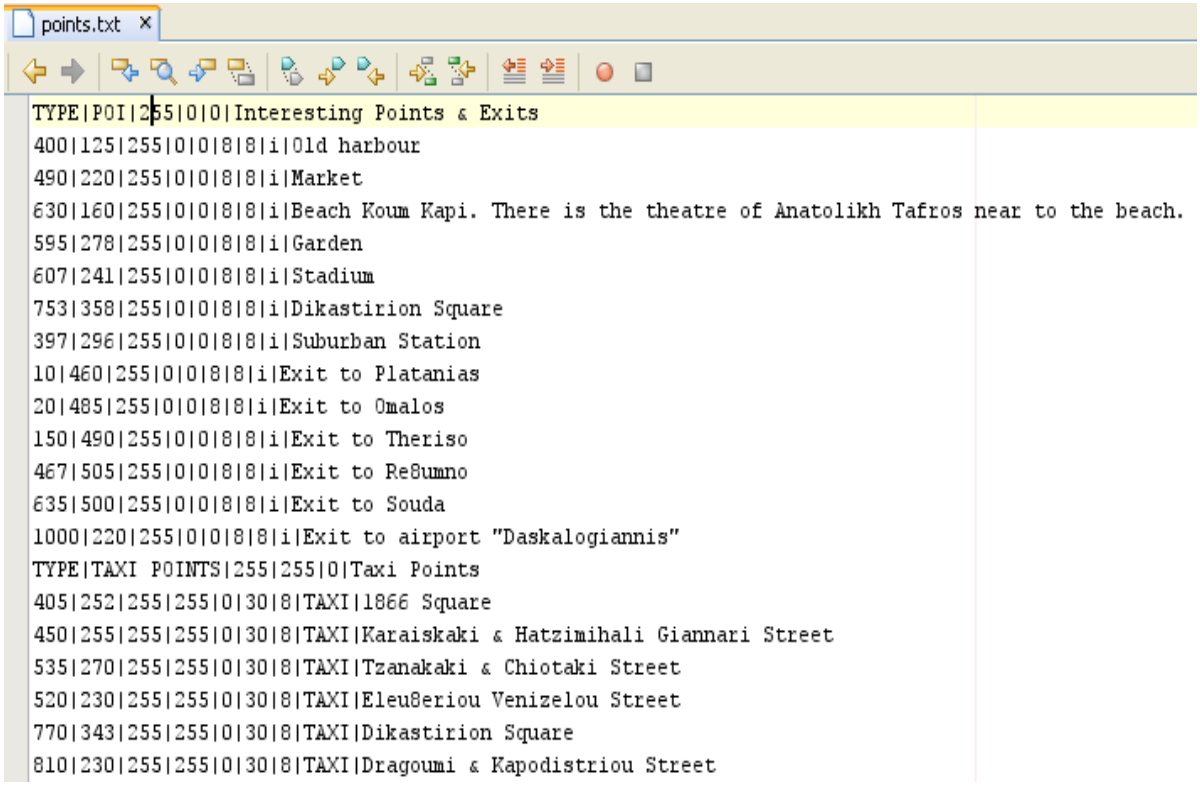
Τέλος η **Item** ιεραρχία, ορίζει τα συστατικά που φαίνονται. Όλες οι διακριτές υποκλάσεις της Item μπορούν να εμφανιστούν στην οθόνη της συσκευής. Δεν μπορούν όμως να εμφανιστούν ανεξάρτητα όπως τα top level συστατικά της κλάσης Screen, παρα μόνο με την βοήθεια ενός Form αντικειμένου.

4.3.4 Λεπτομέρειες Υλοποίησης

Στο σημείο αυτό περιγράφεται η διαδικασία υλοποίησης της εφαρμογής, ο κώδικας της οποίας δίνεται στο CD το οποίο συνοδεύει το παρόν κείμενο της διπλωματικής εργασίας. Θα πρέπει να αναφερθεί ότι ο κώδικας της εφαρμογής δημιουργήθηκε με το πρόγραμμα NetBeans 5.5.1 το οποίο μπορεί να ενσωματώσει και emulators πραγματικών κινητών τηλεφώνων.

Κατά την κατασκευή της εφαρμογής δημιουργήθηκε το **package map**, το οποίο περιέχει πέντε κλάσεις και ένα αρχείο. Η βασική φιλοσοφία της υλοποίησης της εφαρμογής η οποία δημιουργήθηκε με δυναμικό τρόπο, έγκειται στα εξής : υπάρχει ένα αρχείο κειμένου το οποίο περιέχει κωδικοποιημένες πληροφορίες για τα σημεία ενδιαφέροντος που θέλουμε να τοποθετήσουμε πάνω στο χάρτη. Το αρχείο αυτό διαβάζεται από μία μέθοδο του προγράμματος, η οποία επεξεργάζεται τις γραμμές του αρχείου μια προς μία, διαχωρίζει την πληροφορία και έπειτα με κάποιες άλλες μεθόδους που θα αναλυθούν λεπτομερώς στη συνέχεια της παραγράφου έχουμε το αποτέλεσμα που παρουσιάστηκε προηγουμένως.

Ένα παράδειγμα της δομής του κειμένου φαίνεται στην σελίδα που ακολουθεί :

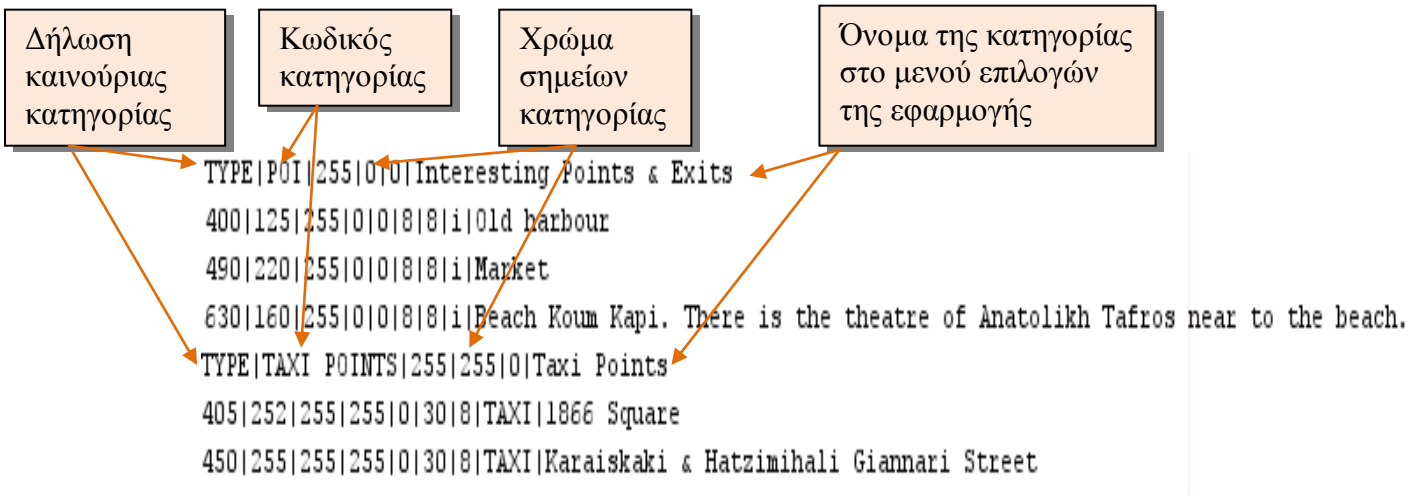


ΕΙΚΟΝΑ 4.17 : Παράδειγμα της δομής του αρχείου κειμένου points.txt

Το αρχείο κειμένου ονομάζεται ***points.txt*** και δομείται ως εξής:

Κάθε γραμμή του κειμένου περιέχει πληροφορίες για τα σημεία ενδιαφέροντος που βλέπουμε τοποθετημένα πάνω στο χάρτη : όταν θέλουμε να δηλώσουμε μια νέα κατηγορία έχουμε ορίσει η γραμμή του κειμένου να ξεκινά με την λέξη TYPE, και όλα τα επόμενα είδη πληροφορίας που θέλουμε να δώσουμε να ξεχωρίζουν μεταξύ τους με το διαχωριστικό σύμβολο | .Οι υπόλοιπες πληροφορίες που δίνουμε κατά την δήλωση της κατηγορίας είναι, ο κωδικός της κατηγορίας, το χρώμα όλων των σημείων που ανήκουν σ΄αυτήν και η επιγραφή της που θέλουμε να φαίνεται στο μενού επιλογών της εφαρμογής. Αφού δηλωθεί η κατηγορία, στις αμέσως επόμενες γραμμές γράφονται τα σημεία που θέλουμε να δημιουργήσουμε δίνοντας πληροφορία για τις συντεταγμένες

πάνω στις οποίες θέλουμε να τοποθετηθούν, το χρώμα, τις διαστάσεις τους, το μύνημα που θέλουμε να φέρουν και την επιγραφή τους.



ΕΙΚΟΝΑ 4.18 : Δήλωση κατηγορίας

Το χρώμα δίνεται με την μορφή RGB (r,g,b) γι' αυτό και δηλώνεται με 3 αριθμούς. Για παράδειγμα το κόκκινο χρώμα που αντιπροσωπεύει τα σημεία της κατηγορίας *Interesting points & Exits* ορίζεται από την αλληλουχία των αριθμών 255|0|0 .

Η εικόνα που ακολουθεί κάνει πιο κατανοητή τη δομή του αρχείου. Έτσι για παράδειγμα αν θέλαμε να τοποθετήσουμε ένα σημείο της κατηγορίας *Museums* μετά την δήλωση της κατηγορίας όπως δείξαμε παραπάνω θα γράφαμε στο αρχείο points.txt :

ΕΙΚΟΝΑ 4.19 : Δημιουργία ενός σημείου της κατηγορίας *Museums*

Το αρχείο παίζει πολύ σημαντικό ρόλο στην δυναμική υλοποίηση του όλου συστήματος. Όταν λέμε δυναμική υλοποίηση εννοούμε ότι αν κάποιος θελήσει να αλλάξει κατηγορίες σημείων,ή να τοποθετήσει καινούριες, ή να προσθέσει κάποια καινούρια σημεία, μπορεί να το κάνει πολύ εύκολα πειράζοντας το αρχείο μόνο και όχι τον κώδικα. Αυτός είναι ο «μαγικός τρόπος» που αναφέρθηκε στην παράγραφο 4.2 (εικόνα 4.16) για την δημιουργία νέας κατηγορίας.

Συνεχίζουμε παρουσιάζοντας τις πέντε κλάσεις και τις μεθόδους που περιέχουν και που αναφέρθηκαν προηγουμένως.

Class MapMidlet

Η κλάση MapMidlet είναι το visual MIDlet της εφαρμογής. Είναι η κλάση μέσα από την οποία καλείται ένα αντικείμενο μιας κλάσης που θα δούμε παρακάτω και που υλοποιεί όλα αυτά που φαίνονται στην

οθόνη του κινητού μας τηλεφώνου. Αποτελείται από έναν constructor και από εννιά μεθόδους.

- Ο constructor, όπως ξέρουμε φέρει το όνομα της κλάσης : ονομάζεται δηλαδή ***MapMidlet()*** και αυτό που κάνει στην συγκεκριμένη περίπτωση είναι να δημιουργεί ένα αντικείμενο της κλάσης ImageCanvas.
- Συνεχίζουμε με τη μέθοδο ***initialize()*** , η οποία αρχικοποιεί το User Interface (UI) της εφαρμογής και που συγκεκριμένα με την εντολή getDisplay().setCurrent(imgCanvas), ορίζει να εμφανιστεί στην οθόνη του κινητού μας τηλεφώνου το αντικείμενο της κλάσης ImageCanvas που δημιουργήθηκε από τον constructor.
- Η μέθοδος ***getDisplay()*** επιστρέφει ένα instance της Display που περιγράφηκε στην αρχή της παραγράφου 4.3.2.
- Η μέθοδος ***exitMIDlet()*** εκτελεί την έξοδο της εφαρμογής
- Η μέθοδος ***get_exitCommand()*** επιστρέφει ένα instance της εντολής Exit έτσι ώστε αν θέλουμε να χρησιμοποιήσουμε την εντολή exit να μην την δημιουργούμε, να την ορίζουμε ξανά και να την προσθέτουμε στο κομμάτι του κώδικα που τη χρειαζόμαστε. Με μία απλή κλήση της μεθόδου get_exitCommand() τα παραπάνω θα γίνονται αυτόματα.
- Η μέθοδος get_form1() επιστρέφει ένα instance της κλάσης Form αφού πρώτα το δημιουργήσει.
- Η μέθοδος ***startApp()*** περιορίζει τη δράση της μόνο στην κλήση της μεθόδου initialize() που περιγράφηκε παραπάνω. Επίσης με αυτήν εδώ τη μέθοδο το AMS στέλνει σήμα στο MIDlet ότι είναι σε active κατάσταση.

- Με τη μέθοδο ***pauseApp()*** το AMS στέλνει σήμα στο MIDlet ότι βρίσκεται σε pause κατάσταση.
- Με τη μέθοδο ***destroyApp ()*** το AMS στέλνει σήμα στο MIDlet ότι βρίσκεται σε destroyed κατάσταση και ότι πλέον έχει σταματήσει.

Class Category

Η κλάση Category αποτελείται από 2 constructors και από 4 μεθόδους. Είναι η πιο μικρή κλάση του project και ορίζει την ύπαρξη μιας κατηγορίας σημείων . Από τους 2 constructors ο ένας δεν παίρνει ορίσματα, δημιουργεί ένα instance της κλάσης και αρχικοποιεί το χρώμα και το όνομα της κατηγορίας που την αντιπροσωπεύουν. Ο δεύτερος constructor δημιουργεί ουσιαστικά μία κατηγορία σημείων με το όνομα και το χρώμα που δίνονται κάθε φορά (από το αρχείο,όπως περιγράφηκε παραπάνω), αφού παίρνει σαν ορίσματα, ένα String (για το όνομα) και ένα RGB(για το χρώμα).

Οι μέθοδοι που περιέχει η κλάση χωρίζονται σε δύο κατηγορίες : τις Getters και τις Setters. Ονομάζονται έτσι επειδή οι μεν setters θέτουν κάτι ίσο με κάτι άλλο (από το αγγλικο set = θέτω), και οι δε getters που παίρνουν – επιστρέφουν κάτι (από το αγγλικό get = παίρνω).

<u>*Setters:*</u>

- Η μέθοδος ***setColor(RGB c)*** θέτει το χρώμα της κατηγορίας ίσο με αυτό που του δίνει η RGB παράμετρος.
- Η μέθοδος ***setTypeName(String tn)*** θέτει το όνομα της κατηγορίας ίσο με αυτό που του δίνει η String παράμετρος.

Getters:

- Η μέθοδος ***getColor()*** επιστρέφει το χρώμα της κατηγορίας στο σημείο του προγράμματος από όπου και εκλήθη.
- Η μέθοδος ***getTypeName()*** επιστρέφει το όνομα της κατηγορίας των σημείων στο σημείο του προγράμματος από όπου και εκλήθη.

Class PointOfInterest

Η κλάση PointOfInterest αποτελείται από 2 constructors και από 18 μεθόδους. Εννέα setters και εννέα getters μεθόδους. Ορίζει την ύπαρξη ένος σημείου οποιασδήποτε κατηγορίας . Ο constructor με τα κενά ορίσματα *PointOfInterest(),* δημιουργεί ένα instance της κλάσης και αρχικοποιεί τα στοιχεία που περιγράφουν ένα σημείο όπως :

- ✓ Τις συντεταγμένες του σημείου
- ✓ Το χρώμα
- ✓ Την επιγραφή
- ✓ Το μήνυμα που αφορά το σημείο και που εμφανίζεται όταν επιλεγεί η εντολή Info.
- ✓ Τον κωδικό της κατηγορίας
- ✓ Το όνομα της κατηγορίας
- ✓ Το πλάτος του σημείου
- ✓ Το ύψος του σημείου
- ✓ Και τέλος μια boolean μεταβλητή που ενημερώνει αν το σημείο είναι ορατό στο χάρτη ή όχι.

Ο δεύτερος constructor *PointOfInterest(Point p, RGB c, String l, String m, String tc,String tn,int w,int h)* έχει ως ορίσματα τις πληροφορίες που αναφέρθηκαν παραπάνω (συντεταγμένες, χρώμα κλπ) και δημιουργεί

ένα instance με τα στοιχεία που παίρνει από το αρχείο ή τον προγραμματιστή.

Setters

- ✓ Η μέθοδος ***setPos(Point p)*** θέτει τις συντεταγμένες του σημείου ίσες με αυτές που του δίνει η παράμετρος τύπου Point.
- ✓ Η μέθοδος ***setColor(RGB c)*** θέτει το χρώμα του σημείου ίσο με αυτό που του δίνει η παράμετρος τύπου RGB.
- ✓ Η μέθοδος ***setLabel(String l)*** θέτει την επιγραφή του σημείου ίση με αυτή που του δίνει η παράμετρος τύπου String.
- ✓ Η μέθοδος ***setMsg(String s)*** θέτει το μήνυμα που αφορά το σημείο ίσο με αυτό που του δίνει η παράμετρος τύπου String.
- ✓ Η μέθοδος ***setTypeCode(String tc)*** θέτει το κωδικό όνομα της κατηγορίας του σημείου ίσο με αυτό που του δίνει η παράμετρος τύπου String.
- ✓ Η μέθοδος ***setTypeName(String tn)*** θέτει το όνομα της κατηγορίας του σημείου ίσο με αυτό που του δίνει η παράμετρος τύπου String.
- ✓ Η μέθοδος ***setVisible(boolean v)*** θέτει το σημείο ορατό στο χάρτη της εφαρμογής ή μή ορατό. Αν η Boolean μεταβλητή είναι αληθής τότε το σημείο είναι ορατό, ενώ αν είναι ψευδής το σημείο δεν φαίνεται στο χάρτη.
- ✓ Η μέθοδος ***setPointWidth(int w)*** θέτει το πλάτος του σχήματος του σημείου ίσο με αυτό που του δίνει η παράμετρος τύπου int.
- ✓ Η μέθοδος ***setPointHeight(int h)*** θέτει το ύψος του σχήματος του σημείου ίσο με αυτό που του δίνει η παράμετρος τύπου int.

Getters

- ✓ Η μέθοδος ***getPos()*** επιστρέφει τις συντεταγμένες του σημείου στο μέρος εκείνο του προγράμματος από όπου και εκλήθη.
- ✓ Η μέθοδος ***getColor()*** επιστρέφει το χρώμα του σημείου στο μέρος εκείνο του προγράμματος από όπου και εκλήθη.
- ✓ Η μέθοδος ***getLabel()*** επιστρέφει την επιγραφή του σημείου στο μέρος εκείνο του προγράμματος από όπου και εκλήθη.
- ✓ Η μέθοδος ***getMsg()*** επιστρέφει το μήνυμα που αφορά το σημείο, στο μέρος εκείνο του προγράμματος από όπου και εκλήθη.
- ✓ Η μέθοδος ***getTypeCode()*** επιστρέφει τον κωδικό της κατηγορίας του σημείου, στο μέρος εκείνο του προγράμματος από όπου και εκλήθη.
- ✓ Η μέθοδος ***getTypeName()*** επιστρέφει το όνομα της κατηγορίας του σημείου, στο μέρος εκείνο του προγράμματος από όπου και εκλήθη.
- ✓ Η μέθοδος ***getVisible()*** επιστρέφει F (false) ή T(true), διαψεύδοντας η επαληθεύοντας ότι το σημείο είναι ορατό στο χάρτη ή οχι, στο μέρος εκείνο του προγράμματος από όπου και εκλήθη.
- ✓ Η μέθοδος ***getPointWidth()*** επιστρέφει το πλάτος του σχήματος του σημείου στο μέρος εκείνο του προγράμματος από όπου και εκλήθη.
- ✓ Η μέθοδος ***getPointHeight ()*** επιστρέφει το ύψος του σχήματος του σημείου, στο μέρος εκείνο του προγράμματος από όπου και εκλήθη.

Class MyForm1

Η κλάση MyForm1 αποτελείται από έναν constructor και από δύο μεθόδους. Η κλάση αυτή ορίζει τη φόρμα που βλέπουμε στην οθόνη της συσκευής όταν από το μενού των επιλογών επιλέξουμε Info. Ο constructor MyForm(MapMidlet midlet) παίρνει σαν όρισμα το midlet στο οποίο θα προσαρτηθεί η φόρμα και δημιουργεί μια εντολή back και ένα ticker(κυλιόμενο μήνυμα στο πάνω μέρος του Form και άρα και της οθόνης του κινητού μας τηλεφώνου).

- ✓ Η μέθοδος ***setInfoValue(String info)*** καθορίζει τα χαρακτηριστικά της γραμματοσειράς που θα χρησιμοποιηθεί για την σύνταξη του μηνύματος και που θα φανεί στην οθόνη, και επίσης θέτει το μήνυμα ίσο με αυτό που δίνει η παράμετρος String.
- ✓ Η μέθοδος ***commandAction(Command c, Displayable d)*** παίρνει σαν ορίσματα μια εντολή καθώς και το displayable(Canvas η Form) για το οποίο θα ισχύει η εντολή αυτή. Στην προκειμένη περίπτωση η μέθοδος αυτή ελέγχει αν η εντολή είναι η back. Αν ισχύει το παραπάνω, με το που επιλεγεί η εντολή back, από την κατάσταση Form στην οποία βρισκόμαστε και βλέπουμε στην οθόνη (μήνυμα που αφορά το σημείο), μεταφερόμαστε στην προηγούμενη κατάσταση της οθόνης και βλέπουμε και πάλι το χάρτη(Canvas).

Class ImageCanvas

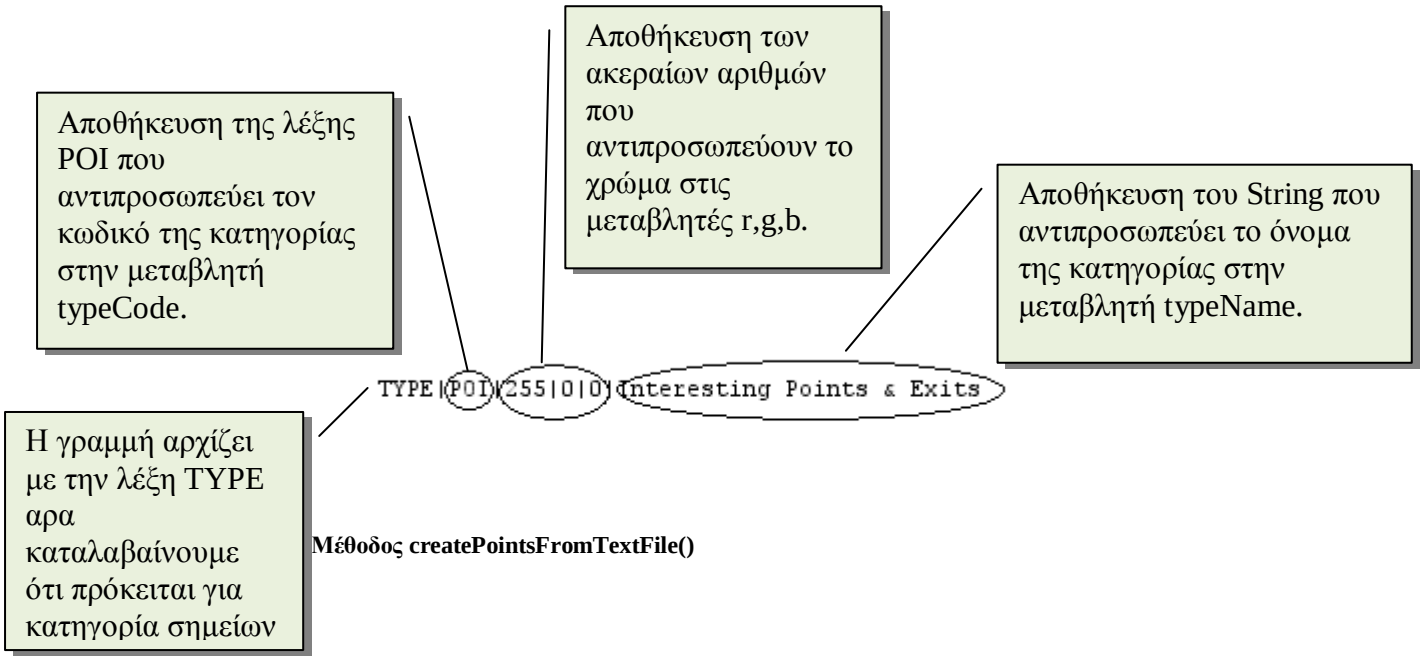
Η κλάση ImageCanvas είναι η πιο εκτενής κλάση της εφαρμογής και στην πραγματικότητα, μέσα σ΄ αυτήν δημιουργούνται σχεδόν όλα από όσα βλέπουμε στην οθόνη της φορητής μας συσκευής. Παρακάτω γίνεται

μια λεπτομερής ανάλυση των μεθόδων που υλοποιήθηκαν προκειμένου να έχουμε το γνωστό αποτέλεσμα. Η κλάση ImageCanvas αποτελείται από έναν constructor και από πέντε μεθόδους.

Ο constructor ImageCanvas() πάιρνει σαν όρισμα το midlet στο οποίο θα προσαρτηθεί (MapMidlet), κάνει τις απαραίτητες αρχικοποιήσεις, δημιουργεί εντολές, δεσμεύει χώρο και δημιουργεί το image ChaniaTown.png που απεικονίζει το χάρτη της πόλης των Χανίων και τέλος καλεί τις μεθόδους που δημιουργούν τα σημεία από το .txt αρχείο και το μενού των επιλογών της εφαρμογής.

- ✓ Η μέθοδος ***createPointsFromTextFile(String fileName)*** δέχεται σαν όρισμα το όνομα του αρχείου που περιέχει τις πληροφορίες για τα σημεία και τις κατηγορίες τους και που στην προκειμένη περίπτωση ονομάζεται points.txt. Η λειτουργία της μεθόδου έγκειται στα εξής : όλες οι γραμμές του αρχείου φορτώνονται και αποθηκεύονται σε έναν buffer από όπου και επεξεργάζονται μία – μία. Αν η γραμμή αρχίζει από τη λέξη TYPE, τότε καταλαβαίνουμε ότι πρόκειται για μία νέα κατηγορία και αρχίζει μια διαδικασία συλλογής πληροφοριών για την κατηγορία αυτή. Το πρόγραμμα διαβάζει την γραμμή μέχρι να βρεί τον διαχωριστικό χαρακτήρα | - όπως έχει αναφερθεί και κατά την παρουσίαση της δομής του αρχείου – ή μεχρι να βρεθεί το τέλος της γραμμής (-1). Όταν βρεί το διαχωριστικό |, αποθηκεύει σε μία μεταβλητή αυτό που έχει διαβάσει εως τώρα και συνεχίζει διαβάζοντας και αποθηκεύοντας σε διαφορετικές μεταβλητές κάθε φορά, μεχρι να βρει και πάλι | ή το τέλος της γραμμής.

Κάτι ανάλογο ισχύει όταν η γραμμή ξεκινά με αριθμό. Μόνο που τότε το πρόγραμμα «καταλαβαίνει» ότι πρόκειται για εισαγωγή σημείου και με ανάλογο τρόπο αποθηκεύει τις απαραίτητες πληροφορίες .



Μέθοδος createPointsFromTextFile()

Αφού συλλεχθούν τα στοιχεία για όλα τα σημεία και τις κατηγορίες τους, στην συνέχεια δεσμεύεται χώρος στη μνήμη για να δημιουργηθούν αυτά τα σημεία.

- ✓ Η μέθοδος ***createCommands()*** παίρνει το όνομα της κατηγορίας των σημείων που δημιουργήθηκε και που αποθηκεύτηκε προηγουμένως στην μεταβλητή typeName και το καρφιτσώνει στο μενού επιλογών της εφαρμογής.

- ✓ Ο ρόλος της μεθόδου ***paint()*** έγκειται στη δημιουργία γραφικών και περιλαμβάνει το «ζωγράφισμα» του χάρτη, το «ζωγράφισμα»των σημείων πάνω στο χάρτη καθώς και των επιγραφών πάνω στο σχήμα που συμβολίζει το σημείο ενδιαφέροντος. Επίσης με τη μέθοδο αυτή μπορούμε να μετακινούμε τις συντεταγμένες του χάρτη οι οποίες φαίνονται στην οθόνη της συσκευής, έτσι ώστε με το πάτημα των πλήκτρων μετατόπισης να μπορούμε να έχουμε οπτική επαφή με κάθε μέρος του χάρτη.
- ✓ Η μέθοδος ***commandAction(Command c, Displayable d)*** παίρνει σαν όρισμα Commands και Dislayable. Η μέθοδος αυτή είναι διαφορετική από τη μέθοδο commandAction που παρουσιάστηκε στην κλάση MyForm1 και αφορά την κλάση ImageCanvas. Λειτουργεί ως εξής :
    - Αν η εντολή που επιλέχτηκε στο μενού επιλογών από το χρήστη είναι η *Exit*, τότε η εφαρμογή τερματίζεται.
    - Αν επιλέχτηκε η εντολή *Map* τότε εμφανίζεται ο χάρτης με όλες τις κατηγορίες των σημείων. Ο χρήστης έχει τη δυνατότητα πλοήγησης σε όλο το χάρτη χωρίς όμως να μπορεί να επιλέξει κάποιο από όλα τα σημεία.
    - Αν επιλέχτηκε μία από τις κατηγορίες που δημιουργήθηκαν, ο χρήστης μπορεί να εξερευνήσει όλα τα σημεία της κάθε κατηγορίας και επίσης μπορεί να τα επιλέξει. Η επιλογή γίνεται όταν το σημείο που μας ενδιαφέρει γίνει πράσινο με τη βοήθεια των πλήκτρων κατεύθυνσης.

- Αν επιλέχτηκε η εντολή Info τότε η μέθοδος commandAction() καλεί τον constructor της κλάσης MyForm1 και δημιουργεί ένα instance της κλάσης αυτής έτσι ώστε να μπορούμε να δούμε στην οθόνη την πληροφορία του σημείου που επιλέξαμε.
- Τέλος αν επιλέχτηκε η εντολή Help, τότε η μέθοδος commandActıon() και πάλι δημιουργεί ένα instance της κλάσης MyForm1, έτσι ώστε να δούμε στην οθόνη το κείμενο βοηθείας.

✓ Η μέθοδος ***keyPressed(int keyCode)*** παίρνει σαν όρισμα το πλήκτρο (κάθε πλήκτρο αντιστοιχεί σε έναν κωδικό ακέραιο αριθμό) που πατήθηκε από τον χρήστη της συσκευής. Ο ρόλος της είναι ο πιο βασικός για τη λειτουργία της εφαρμογής. Πριν εξηγηθεί η η λειτουργία της μεθόδου όμως, πρέπει να πούμε ότι έχουμε δημιουργήσει μια μεταβλητή mode στην οποία αποθηκεύουμε αν η εφαρμογή μας είναι σε κατάσταση Map (όπου γίνεται και η περιήγηση σε όλο το χάρτη), ή αν βρίσκεται σε κατάσταση επιλογής κατηγορίας και μόνο (όπου η περιήγηση γίνεται μόνο ανάμεσα στα σημεία της κατηγορίας). Ο περιορισμός δεν ισχύει για τις επιλογές Info, Exit, Help.

- Αν το mode της εφαρμογής είναι Map τότε η μέθοδος κάνει δυνατή την περιήγηση σε όλο το χάρτη. Αν για παράδειγμα πατηθεί το πλήκτρο UP τότε ο χρήστης θα δει στην οθόνη του κινητού τηλεφώνου, το πάνω μέρος του χάρτη σε σχέση με το μέρος του χάρτη που έβλεπε πριν πατηθεί το πλήκτρο αυτό. Αυτό επιτυγχάνεται με την αλλαγή των

συντεταγμένων από τις οποίες αρχίζει να ζωγραφίζει η μέθοδος paint().

- Αν έχει επιλεγεί οποιαδήποτε εντολή από το μενού των επιλογών που αφορά τις κατηγορίες τότε η εφαρμογή αλάζει κατάσταση και η μέθοδος κάνει δυνατή μόνο την περιήγηση από σημείο σε σημείο της ίδιας κατηγορίας. Πρέπει να υπενθυμιστεί ότι όταν έχουμε επιλέξει κάποια απο τις κατηγορίες η μετακίνηση απο σημείο σε σημείο γίνεται μόνο με την βοήθεια των πλήκτρων RIGHT και LEFT. Η υλοποίηση της μετακίνησης απο σημείο σε σημείο γίνεται με τον εξής τρόπο : Βρισκόμαστε σε ένα σημείο το οποίο είναι πράσινο,ανεξάρτητα με το χρώμα της κατηγορίας του, και θέλουμε να δούμε ποιο είναι το επόμενο σημείο που υπάρχει στα δεξιά του. Όταν πατηθεί το πλήκτρο RIGHT τότε, ψάχνουμε όλα τα σημεία των οποίων οι συντεταγμένες x, είναι πιο μεγάλες σε αριθμό από την συντεταγμένη x στην οποία βρίσκεται το ήδη επιλεγμένο σημείο. Αφού τις βρούμε, τότε ψάχνουμε ποιο σημείο έχει την πιο κοντινή απόσταση απο το σημείο που είμαστε. Όταν το βρούμε, το σημείο αυτό γίνεται πράσινο και το προηγούμενο ανακτά και πάλι το χρώμα της κατηγορίας του. Κάτι αντίστοιχο συμβαίνει και όταν πατήσουμε το πλήκτρο LEFT.

Η keyPressed( ), ήταν η τελευταία μέθοδος της κλάσης ImageCanvas, και κάπως έτσι τελειώνει η ανάλυση του κώδικα της εφαρμογής. Για καλύτερη κατανόηση του κώδικα της εφαρμογής, μπορείτε να ανατρέξετε στον κώδικα που δίνεται σε CD μαζί με το παρόν κείμενο συνοδευόμενος από σχόλια. Για τυχόν απορίες, μπορείτε να επικοινωνήσετε μαζί μου, στέλνοντας e-mail στο katerinakontomanou@yahoo.gr .

4.4 Σχόλια – Παρατηρήσεις

Η υλοποίηση της εφαρμογής που παρουσιάστηκε παραπάνω δεν ήταν εύκολη υπόθεση. Πραγματοποιήθηκαν πολλές αλλαγές για να φτάσουμε στο επιθυμητό αποτέλεσμα. Το κυριότερο πρόβλημα που αντιμετωπίστηκε κατά την δημιουργία του κώδικα της εφαρμογής ήταν ότι αρχικά ο κώδικας είχε δημιουργηθεί με στατικό τρόπο. Αυτό σημαίνει ότι όλη η πληροφορία που υπάρχει στο αρχείο points.txt ήταν αποθηκευμένη εντός του κώδικα. Το γεγονός αυτό δυσχαίραινε την εισαγωγή καινούριων σημείων, ή κατηγοριών, την αλλαγή ή την εισαγωγή καινούριων πληροφοριών, επειδή κάτι τέτοιο θα καθιστούσε απαραίτητη την αλλαγή του κώδικα της εφαρμογής και σαφώς δεν θα ήταν εύκολο να γίνει από κάποιον που δεν ξέρει να προγραμματίζει. Επιπλέον η στατική υλοποίηση της εφαρμογής έκανε τον κώδικά της αχανή από την άποψη ότι έπιανε πολλές γραμμές κώδικα. Χαρακτηριστικό παράδειγμα αποτελεί η κλάση ImageCanvas, η οποία με την στατική υλοποίηση έπιανε 1300 περίπου γραμμές κώδικα, ενώ με την

δυναμική υλοποίηση, η έκτασή της περιορίστικε σε λίγες εκατοντάδες γραμμές κώδικα.

Επίσης πρέπει να αναφερθεί ως μεγίστης σημασίας παρατήρηση ότι η μνήμη που καταλαμβάνει η παρούσα εφαρμογή είναι 938 KB ενώ ο χάρτης που περιέχεται στην εφαρμογή αυτή, είναι 918 KB. Καταλαμβάνει δηλαδή πάνω από το 95% της μνήμης της εφαρμογής. Επίσης πρέπει να σημειωθεί ότι η μνήμη που απαιτείται για την εφαρμογή αυτή δεν είναι σταθερή αλλά μεταβάλλεται ανάλογα με τις προσθήκες, τις αλλαγές και τις διαγραφές στο αρχείο κειμένου. Δεν θα απαιτείται όμως μνήμη για την εφαρμογή λιγότερη από 918 KB πού είναι και το μέγεθος (σε μνήμη) του .png αρχείου του χάρτη.

4.5 Επαλήθευση σωστής λειτουργίας της εφαρμογής

Στην παράγραφο αυτή, αναφέρεται ότι για να φτάσουμε στο επιθυμητό αποτέλεσμα, έγιναν πολλές επαληθεύσεις για να πιστοποιηθεί ότι η εφαρμογή αυτή λειτουργεί σωστά και ότι δεν υπάρχει περίπτωση το σύστημα να καταρρεύσει:

- Έτσι λοιπόν δοκιμάστηκαν πολλές φορές όλες οι επιλογές που ανήκουν στο menu της εφαρμογής με διαφορετική σειρά κάθε φορά και με πολύ μεγάλη προσοχή.

- Ακόμη, επιλέχτηκαν όλα τα σημεία κάθε κατηγορίας ένα προς ένα πολλές φορές για να δούμε αν υπάρχει πρόβλημα με το χρώμα της κατηγορίας, με την πληροφορία που φέρουν όλα τα σημεία ενδιαφέροντος πάνω τους, ή με την θέση τους στο χάρτη.

- Επιπλέον δοκιμάστηκε η εισαγωγή, διαγραφή ή ενημέρωση, σημείου αλλά και κατηγορίας στο αρχείο points.txt, έτσι ώστε η αλλαγή του αρχείου κειμένου να συνεπάγεται και την αντίστοιχη αλλαγή στον χάρτη που βλέπουμε στην οθόνη του κινητού μας τηλεφώνου. Για παράδειγμα, αν θέλουμε να εισάγουμε μια κατηγορία δύο σημείων με το όνομα Parkings, επιγραφή P, χρώμμα μωβ, με διαστάσεις και κείμενο της αρεσκείας μας στο αρχείο, να βλέπουμε και την αντίστοιχη προσθήκη στο χάρτη και το μενού της εφαρμογής μας (Εικόνα 4.16).

- Ακόμη πιστοποιήθηκε ότι η μνήμη που καταλαμβάνει η εφαρμογή ανήκει στις προδιαγραφές μνήμης που ορίζει το δίδυμο της διαμόρφωσης και του προφίλ για τα κινητά τηλέφωνα και δεν υπάρχει πρόβλημα το σύστημα να καταρρεύσει.

- Τέλος δοκιμάστηκε και η ορθή λειτουργία της πλοήγησης. Αν δηλαδή με το πάτημα των πλήκτρων μετακίνησης γίνεται και η αντίστοιχη μετακίνηση στο χάρτη (πάτημα πλήκτρου δεξιάς μετακίνησης => μετακίνηση του χάρτη προς τα δεξιά). Επίσης δοκιμάστηκε και η πλοήγηση από σημείο σε σημείο.

Αφού δοκιμάστηκαν όλα τα παραπάνω, επαληθεύτηκε η σωστή λειτουργία της εφαρμογής.

5

ΕΓΧΕΙΡΙΔΙΟ ΔΗΜΙΟΥΡΓΙΑΣ ΠΑΡΟΜΟΙΑΣ ΕΦΑΡΜΟΓΗΣ

5.1 Εισαγωγή

Το κεφάλαιο αυτό παρουσιάζει ένα εγχειρίδιο, το οποίο μπορεί να χρησιμοποιήσει κάποιος που ενδιαφέρεται να δημιουργήσει μια παρόμοια εφαρμογή για κινητά τηλέφωνα βασιζόμενη στο προφίλ MIDP και την διαμόρφωση CLDC. Παρουσιάζει κλάσεις και μεθόδους που χρησιμοποιήθηκαν για την παρούσα εφαρμογή, αλλά και κλάσεις και μεθόδους που δεν χρησιμοποιήθηκαν αλλά είναι πολύ χρήσιμες για παρόμοια projects η και για προσθήκη στην ήδη υπάρχουσα εφαρμογή.

5.2 Προγραμματίζοντας με την J2ME

Στις παραγράφους που ακολουθούν παρουσιάζεται μια συνολική εικόνα για την δημιουργία ενος MIDlet με σχόλια και παραδείγματα που βοηθούν τον αναγνώστη να μάθει να προγραμματίζει με την J2ME.

5.2.1 MIDlets και MIDlet states

Το προφίλ MIDP όπως έχει αναφερθεί σε προηγούμενο κεφάλαιο προσδιορίζει κινητά τηλέφωνα, ενώ οι εφαρμογές που γράφονται πάνω σε αυτό αποκαλούνται MIDlets. Τα MIDlets είναι κάτι παρόμοιο με τα γνωστά από την J2SE applets. Κάθε MIDP εφαρμογή είναι ένα MIDlet, και σε κάθε MIDlet χρησιμοποιείται η κλάση javax.microedition.midlet.MIDlet. Επίσης όπως και στα applets, έτσι και στα MIDlets, πρέπει αναγκαστικά να υλοποιηθούν και κάποιες μεθόδοι, όπως είναι οι startApp(), pauseApp(), destroyApp(). Εύκολα μπορεί να μαντέψει κανείς από τα ονόματά τους, ότι αυτές οι μέθοδοι ορίζονται για να διευκολύνουν το Application Management Software (AMS) να διαχειριστεί τον κύκλο ζωής ενός MIDlet. Ένα MIDlet κατα την διάρκεια της ζωής του μπορεί να βρεθεί σε οποιαδήποτε από τις επόμενες καταστάσεις(συνήθως με την σειρά που παρουσιάζονται):

- **Loaded**: Συμβαίνει όταν το MIDlet έχει δημιουργηθεί χρησιμοποιώντας την λέξη κλειδί *new*. Το MIDlet μπορεί να υπάρξει μόνο μια φορά σε αυτήν την κατάσταση κατά την διάρκεια της ζωής του. Σε περίπτωση εξαίρεσης το MIDlet καταστρέφεται.
- **Paused**: Συμβαίνει εφόσον το MIDlet έχει αρχικοποιηθεί.
- **Active**: Στην κατάσταση αυτή το MIDlet εισέρχεται κάθε φορά μετά την κατάσταση paused. Συμβαίνει όταν έχει αρχικοποιηθεί και λειτουργεί κανονικά.
- **Destroyed**: Συμβαίνει όταν το MIDlet έχει καταστραφεί.

5.2.2 Τι είναι τα MIDlet suites

Όπως οι J2SE κλάσεις μπορούν να «πακεταριστούν» σε ένα μόνο .jar αρχείο, έτσι μπορούν και τα MIDlets. Όταν γίνει αυτό, το .jar αρχείο ονομάζεται *MIDlet suite.* Ένα MIDlet suite δημιουργείται αν τα MIDlets πρέπει να κάνουν share ένα κομμάτι κώδικα ή δεδομένων. Σημαντικό να αναφερθεί είναι ότι MIDlets που ανήκουν σε διαφορετικά suites δεν μπορούν να αλληλεπιδράσουν μεταξύ τους κατευθείαν. Αυτό το γεγονός παρέχει ασφάλεια στο MIDlet suite.

Το όνομα, η έκδοση, και ο δημιουργός του MIDlet suite προσδιορίζονται κατά την δημιουργία του προγράμματος δίνοντας κάποιες εισόδους. Οι είσοδοι αυτές, επίσης περιγράγουν τις ελάχιστες προδιαγραφές της διαμόρφωσης και του προφίλ. Μερικά από τα χαρακτηριστικά του MIDlet είναι υποχρεωτικό να προσδιοριστούν και κάποια άλλα όχι. Όλα μαζί υποχρεωτικά και προαιρετικά συνοψίζονται παρακάτω :

- **MicroEdition – Configuration**: το όνομα και η έκδοση της J2ME διαμόρφωσης που απαιτείται για να τρέξει το MIDlet suite. (προαιρετικό)
- **MicroEdition – Profile :** το όνομα και η έκδοση του J2ME προφίλ που απαιτείται για να τρέξει το MIDlet suite.(υποχρεωτικό)
- **MIDlets :** το όνομα, το εικονίδιο, και η κύρια κλάση από κάθε MIDlet που ανήκει στο suite.(υποχρεωτικό)
- **MIDlet – Data –Size :** ο ελάχιστος αριθμός bytes μόνιμης αποθήκευσης που χρειάζεται το MIDlet για να τρέξει.(προαιρετικό)

- **MIDlet – Description:** μια περιγραφή του MIDlet suite. (προαιρετικο)
- **MIDlet - Icon:** το μονοπάτι του .png αρχείου μέσα στο .jar αρχείο που χρησιμοποιείται από το AMS για να αναγνωρίσει το MIDlet suite. (προαιρετικό)
- **MIDlet – Info – URL:** ένα URL που περιγράφει το MIDlet suite με λεπτομέρεια. (προαιρετικό)
- **MIDlet – Name:** το όνομα του MIDlet suite. (υποχρεωτικό)
- **MIDlet – Vendor:** ο πωλητής του suite. (υποχρεωτικό)
- **MIDlet – Version:** ο αριθμός έκδοσης του suite. (υποχρεωτικό)

5.2.3 MIDP API

Το MIDP API(Application Programming Interface) είναι το σύνολο των πακέτων που προσδιορίζει το προφίλ MIDP και που χρησιμοποιείται για την επίτευξη της επικοινωνίας μεταξύ προγράμματος και χρήστη. Αποτελείται από τα εξής πακέτα :

- ***Core packages (java.io, java.lang, java.util*)** : αυτά τα πακέτα προσδιορίζουν την βάση πάνω στην οποία στηρίζονται η διαμόρφωση CLDC και το προφίλ MIDP.
- ***javax.microedition.lcdui***: ένα user interface API που χρησιμοποιείται για Mobile Information συσκευές όπως τα κινητά τηλέφωνα.
- **javax.microedition.rms**: παρέχει έναν μηχανισμό για τα MIDlets, ο οποίος αποθηκεύει συνεχώς δεδομένα και αργότερα τα ανακτεί.

- **javax.microedition.midlet:** το πακέτο αυτό ορίζει τις MIDP εφαρμογές και τις αλληλεπιδράσεις μεταξύ των εφαρμογών, όπως επίσης και το περιβάλλον στο οποίο τρέχουν αυτές οι εφαρμογές.
- **javax.microedition.io:** είναι ένα πακέτο το οποίο παρέχει δικτυακή υποστήριξη.

Τα core packages είναι τα ίδια και στην CLDC διαμόρφωση. Σκοπός τους είναι να ορίζουν την γλώσσα Java, όπως χρησιμοποιείται στα CLDC και MIDP.

5.2.4 Το πακέτο του κύκλου ζωής της εφαρμογής

Το πακέτο javax.microedition.midlet είναι για το προφίλ MIDP ό,τι και το Applet package για την Standard έκδοση. Περιλαμβάνεται κυρίως για να ορίσει πως οι MIDP εφαρμογές θα πρέπει να οργανώνονται και πως θα πρέπει να αλληλεπιδρούν με το περιβάλλον τους. Αυτός είναι και ο λόγος που το πακέτο αυτό έχει μόνο μία κλάση, η οποία ονομάζεται MIDlet. Όπως το applet, έτσι και το MIDlet είναι μία εφαρμογή. Κάθε MIDP εφαρμογή οφείλει να περιέχει αυτή την κλάση έτσι ώστε το AMS να μπορεί να ελέγχει την εφαρμογή.

Όπως είδαμε και νωρίτερα, το AMS διαχειρίζεται τις μεθόδους που δημιουργούν, ξεκινούν, σταματούν και καταστρέφουν ένα MIDlet. Η δυνατότητα για ένα MIDlet να έχει πολλές καταστάσεις είναι πολύ σημαντική επειδή επιτρέπει πολλές εφαρμογές να τρέχουν την ίδια στιγμή. Εκτός από τις μεθόδους που κληρονομεί, αυτή η κλάση έχει και τις ακόλουθες μεθόδους :

- startApp
- pauseApp
- destroyApp
- notifyDestroyed
- notifyPaused
- getAppProperty
- resumeRequest

Για να αποκτηθεί μια καλύτερη ιδέα για την χρήση αυτής της κλάσης και των μεθόδων της, παρατίθεται ο ακόλουθος κώδικας, ο οποίος αναπαριστά τον σκελετό μιας MIDP εφαρμογής :

```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;

/**
 * Skeleton Application illustrating the use of MIDlet Class..
 */
public class first extends MIDlet implements CommandListener {

 private Command quit;  // The Quit button
 private Display ourDisplay;  // Declaring the display

// Initialize the Display and place system controls in the Constructor..
 public first()
 {
  ourDisplay = Display.getDisplay(this);
   quit = new Command("Quit", Command.SCREEN, 2);
 }

 /**
  * Initialize all the classes to be used in the program here (startApp()).. 
 */
 public void startApp()
 {

   ...
   ...

 }
```

```
// If the Application needs to be paused temporarily.
 public void pauseApp()
 {

   ...
   ...

 }
// Clean up when the application is destroyed..

 public void destroyApp(boolean unconditional)
 {

   ...
   ...

 }

// Event handling routine..

 public void commandAction(Command c, Displayable s)
   {
     if (c == quit)        // If Quit button is pressed..
      {
            notifyDestroyed(); // Call the destroyApp method..
      }
   }
}
```

5.2.5 Πακέτο διεπαφής του χρήστη (User Interface Package)

Η παράγραφος αυτή παρουσιάζει το σύνολο των κλάσεων και των μεθόδων που χρησιμοποιεί ο χρήστης για να δει στην οθόνη του κάτι αντίστοιχο αυτής της εφαρμογής.

Το MIDP προφίλ ορίζει το δικό του GUI (Graphical User Interface) πακέτο. Το πακέτο αυτό ονομάζεται javax.microedition.lcdui και είναι ιδανικό για μικρές φορητές συσκευές όπως τα κινητά τηλέφωνα. Το προφίλ MIDP δεν ήταν δυνατό να χρησιμοποιήσει το πακέτο AWT που

χρησιμοποείται στην Standard Edition για την δημιουργία γραφικών γιατί είναι πολύ «βαρύ» για τόσο μικρές συσκευές (από άποψη μνήμης και επεξεργαστή). Το πακέτο αυτό είναι ιδανικό για την δημιουργία παιχνιδιών, αλλά μπορεί να χρησιμοποιηθεί και για άλλους σκοπούς. Έχει σχεδιαστεί με τέτοιο τρόπο ώστε, και η μεταφερσιμότητα αλλά και η ειδική λειτουργικότητα της συσκευής να επιτυγχάνονται στον μέγιστο βαθμό.

Το UI (User Interface) μοντέλο το οποίο χρησιμοποιείται στο MIDP προφίλ βασίζεται στα εξής:

Ένα MIDlet έχει ένα ***Display*** (φανταστείτε οθόνη) όπως έχει αναφερθεί και σε προηγούμενο κεφάλαιο, πάνω στο οποίο μόνο ένα αντικείμενο της κλάσης ***Displayable*** μπορεί να εμφανιστεί . Υπάρχουν δύο είδη αντικειμένων αυτής της κλάσης – Canvas και Screen. Η κλάση Canvas είναι κατάλληλη για χαμηλού επιπέδου UI αντικείμενα και χρησιμοποιείται για την δημιουργία γραφικών(χρησιμοποιώντας την κλάση Graphics). Αντιθέτως η κλάση Screen είναι κατάλληλη για την παροχή ενός συνόλου UI αντικειμένων που χρησιμοποιούνται πολύ συχνά, όπως είναι τα TextField, List, TextBox, ChoiceGroup, StringItem κ.τ.λ.. Η κλάση Screen έχει επίσης μια υποκλάση,η οποία ονομάζεται Form, και στην οποία μπορούν να προστεθούν Items (StringItem, ImageItem, TextField, DateField, Gauge και ChoiceGroup). Οι κλάσεις που είναι διαθέσιμες στο MIDP UI package παρουσιάζονται με αλφαβητική σειρά παρακάτω :

- Alert
- Canvas
- ChoiceGroup
- Command
- DateField
- Display
- Displayable
- Font
- Form
- Gauge
- Graphics
- Image
- ImageItem
- Item
- List
- Screen
- StringItem
- TextBox
- TextField
- Ticker

Επίσης τα interfaces που παρουσιάζονται σε αυτό το πακέτο είναι :

- Choice
- CommandListener
- ItemStateListener

Τα πιο συχνά χρησιμοποιούμενα interfaces και κλάσεις από τα παραπάνω εξηγούνται στις σελίδες που ακολουθούν.

To interface CommandListener

Στο προφίλ MIDP ο χειρισμός γεγονότων(πάτημα ένος πλήκτρου, απλευθέρωση ενός πλήκτρου) βασίζεται σε έναν ακροατή (listener). Κάθε Displayable αντικείμενο χρειάζεται έναν και μόνο έναν ακροατή. Ο CommandListener είναι ένας ακροατής για γεγονότα υψηλού επιπέδου. Η μέθοδος με την οποία θέτουμε σε λειτουργία τον ακροατή γεγονότων είναι η *Displayable.setCommandListener()* . Επίσης έχει μόνο μία μέθοδο η οποία ονομάζεται *commandAction()*. Ένα παράδειγμα ορισμού του CommandListener και της μεθόδου commandAction() φαίνεται στον κώδικα που ακολουθεί :

```
// Implementing the interface...
public class TextFieldCheck extends MIDlet implements CommandListener {

// Declaring Buttons and initializing them to null...
 Command ok = null;
 Command quit = null;

// Handling the event.

public void commandAction(Command c, Displayable d) {
// Event handling for the Button
   if ( c == ok )
   {
     // Code for tasks to be done on OK button press...
   }
   if ( c == quit )
   {
    // Code for tasks to be done on Quit button press...
   }
 }
```

Ένα κομμάτι του κώδικα της παρούσας εφαρμογής που χρησιμοποιεί τον Command Listener φαίνεται παρακάτω :

```
/*orismos ths MyForm1 class*/

class MyForm1 extends Form implements CommandListener
{
    private MapMidlet midlet;
    private Command back ;
    private String str = "Electronic tourist map of Chania";
    private Ticker ticker = new Ticker(str);

    private StringItem s1 = new StringItem("","");


    public MyForm1(MapMidlet midlet)
    {

        super("Information :");
        this.midlet = midlet;
        back = new Command("Back",Command.BACK,1); //dhmiourgia ths entolhs Back
        append(s1);

        setTicker(ticker);      //pros8esh toy Ticker (kuliomeno mhnuma)

        addCommand(back);      // pros8esh ths entolhw back sto Displayable
        setCommandListener(this);  // 8etete o CommandListener se leitourgia
    }

    public void setInfoValue(String info) {
       s1.setFont(Font.getFont(Font.FACE_PROPORTIONAL,Font.STYLE_BOLD,Font.SIZE_LARGE));
       s1.setText(info);
    }

    /* orismos ths me8odou commnandAction , etsi wste otan epilegei h entolh back na
     epistrepsoume sthn prohgoumenh katastash ths o8onhs kai na ksanaginei trexon displayable
     to ImageCanvas*/
    public void commandAction(Command c, Displayable d)
    {

        if (c == back){
        midlet.getDisplay().setCurrent(midlet.getImgCanvas());
        }
    }
```

Η κλάση Alert

Η κλάση αυτή χρησιμοποιείται όταν θέλουμε να δείξουμε ένα μήνυμα, μια ειδοποίηση, ή οποιαδήποτε άλλη πληροφορία στο χρήστη για ένα

ορισμένο χροσνικό διάστημα ή μεχρι να το ακυρώσει ο χρήστης. Μπορεί να περιέχει συμβολοσειρές, εικόνες κ.α. και επίσης μπορει να χειριστεί γεγονότα όπως κάθε άλλο είδος της κλάσης Screen. Οι πιο συχνα χρησιμοποιούμνες μέθοδοι που είναι διαθέσιμες γι αυτήν την κλάση είναι οι εξής :

Μέθοδος	***Λειτουργία***
getDefaultTimeout ()	επιστρέφει το προκαθορισμένο χρονικό διάστημα που εμφανίζεται ένα Alert.
getTimeout ()	επιστρέφει το χρονικό διάστημα κατά το οποίο εμφαίζεται το Alert.
setTimeout()	θέτει το διάστημα κατα το οποίο θα εμφανίζεται το Alert
appendString()	προσθήκη συμβολοσειράς
appendImage()	προσθήκη εικόνας
appendItem()	προσθήκη ενός Item

ΠΙΝΑΚΑΣ 5.1 : Μέθοδοι της κλάσης Alert.

Ένα παράδειγμα χρήσης των μεθόδων αυτών για τη δημιουργία ενός Alert φαίνεται στον κώδικα που ακολουθεί :

```
// Popping an Alert...
// Parameters..
// 1. Title of the Alert..
// 2. Text of Alert.
// 3. Image if required.
// 4 Type of Alert (INFO, WARNING,ERROR, CONFIRMATION, ALARM).


Alert alert = new Alert ("Warning", "You have entered Wrong serial number",
                         null, AlertType.WARNING);
// Making this alert a Modal alert..
     alert.setTimeout(Alert.FOREVER);
//  or else...
// Making this alert disappear after certain milliseconds..
     alert.setTimeout(10);
```

Η κλάση ChoiceGroup

Αυτή η κλάση παρέχει στον προγραμμτιστή έναν τρόπο προσθήκης UI components, τα οποία μπορούν να επιλεγούν από το χρήστη. Χρησιμοποιώντας αυτήν την κλάση μπορεί να δημιουργήσει ραδιοπλήκτρα αλλά και πλαίσια ελέγχου (checkboxes). Ποιο από τα δύο θα δημιουργηθεί εξαρτάται στην τιμή μιας ChoiceType παραμέτρου. Η παράμετρος αυτή αν έχει την τιμή EXCLUSIVE, ο προγραμματιστής θα μπορεί να δημιουργήσει checkboxes, και αν έχει την τιμή MULTIPLE, θα μπορεί να δημιουργήσει ραδιοπλήκτρα. Οι μέθοδοι που παρέχονται για αυτήν την κλάση είναι υλοποιήσεις μεθόδων στο interface Choice και είναι οι εξής :

Μέθοδος	***Λειτουργία***
appendElement(String stringElement, Image imageElement)	προσθέτει ένα element στο ChoiceGroup.
deleteElement(int index)	διαγράφει το element στο οποίο αναφέρεται ο αριθμός index.
getImage(int i)	επιστρέφει το image του element στο οποίο αναφέρεται ο αριθμός i.
getSelectedFlags(boolean[] selectedArray_return)	εξετάζει την κατάσταση του ChoiceGroup και επιστρέφει την κατάσταση όλων των elements στον πίνακα boolean[] selectedArray_return.
getSelectedIndex()	επιστρέφει έναν δείκτη(ακέραιος αριθμός) για ένα επιλεγμένο element στο ChoiceGroup.
getSize()	επιστρέφει τον αριθμό των elements στο ChoiceGroup.
getString(int i)	επιστρέφει το String του element που δείχνει ο δείκτης i.
insertElement(int index, String stringElement, Image imageElement)	εισάγει ένα element στο ChoiceGroup.
isSelected()	επιστρέφει μια μεταβλητή τύπου boolean για το αν ένα element είναι επιλεγμένο η όχι.

setElement(int index, String stringElement, Image imageElement)	εισάγει την συμβολοσειρά και την εικόνα στο element που δείχνει ο αριθμός index.
setSelectedFlags(boolean[] selectedArray)	προσπαθεί να εισάγει την κατάσταση του επιλεγμένου element στον πίνακα selectedArray.
setSelectedIndex(int index, Boolean selected)	για τα ChoiceGroup αντικείμενα τύπου MULTIPLE, ορίζει την κατάσταση ενός επιλεγμένου element.

ΠΙΝΑΚΑΣ 5.2 :Μέθοδοι της κλάσης ChoiceGroup

Παρακάτω φαίνεται ένα κομμάτι κώδικα που δημιουργεί ραδιοπλήκτρα:

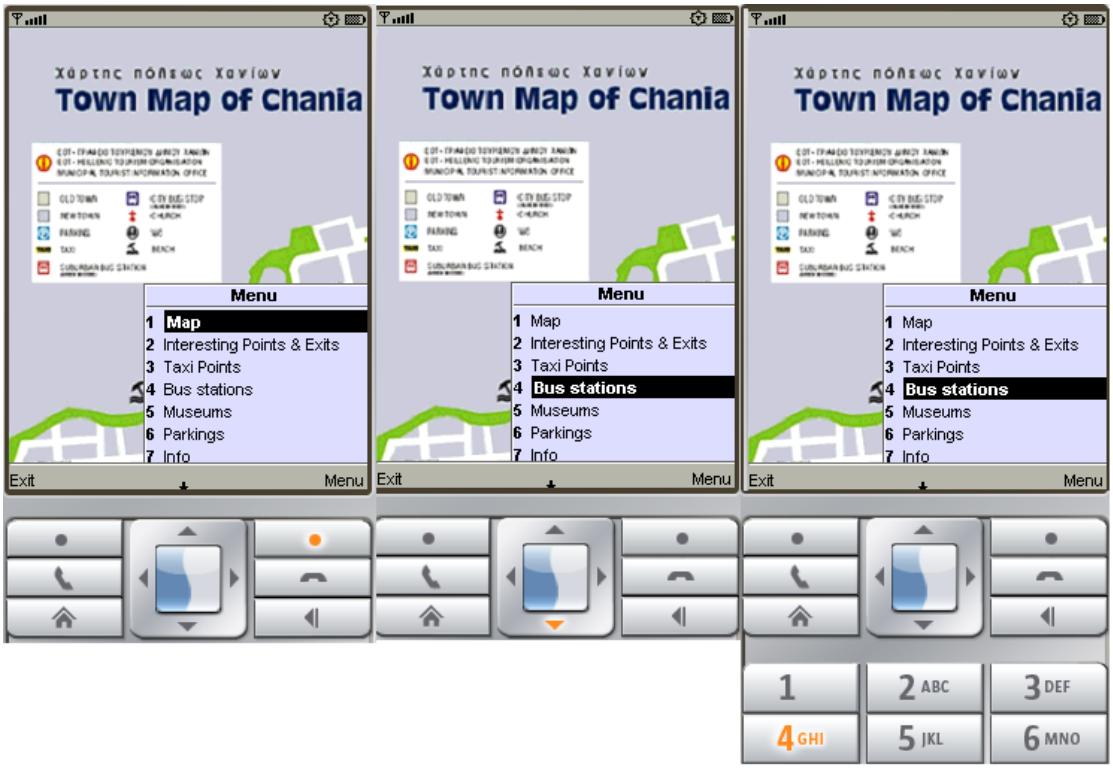
```
// Declaring Form and ChoiceGroup objects and initializing them to null...
Form ui_holder = null;

ChoiceGroup radiobutton_type = null;

String[] name = {"a","b","c"};
Image[] img = null;

// Initialize the ChoiceGroup..
// Parameter 1 ---   Title
// Parameter 2 ---   Type of ChoiceGroup (Exclusive for RadioButtons)
// Parameter 3 ---   Label of the radio buttons displayed
// Parameter 4 ---   Images for the radio button label if required

radiobutton_type = new ChoiceGroup("Choices..",ChoiceGroup.EXCLUSIVE,name,img);

// Adding the ChoiceGroup to the Form...
ui_holder.append(radiobutton_type);

// Event Handling..
// To get the String of the selected radio button...
radiobutton_type.getString(radiobutton_type.getSelectedIndex());
```

Η κλάση Command

Η κλάση Command χρησιμοποιείται για να δημιουργήσει UI αντικείμενα τα οποία συμπεριφέρονται ως κουμπιά μιας που το MIDP δεν ορίζει την κλάση Button. Σε μια Command, περιέχεται μόνο το όνομά της, η προτεραιότητα της και το είδος της. Η λειτουργία του αντικειμένου Command εξαρτάται από τον CommandListener, ο οποίος προστίθεται στο Screen, στο οποίο τοποθετείται το αντικείμενο. Οι παράμετροι που χρειάζεται να περαστούν στον constructor ενός αντικειμένου Command είναι :

- η ετικέτα για το Command button
- η θέση(είδος) του Command button
- και η προτεραιότητα της εντολής σε σχέση με άλλες εντολές.

Η τρίτη παράμετρος λειτουργεί ως εξής : αν ο χρήστης επιλέξει το πλήκτρο που αντιστοιχεί στο μενού μιας εφαρμογής, τότε εμφανίζεται μία λίστα από επιλογές (comnands). Στη συνέχεια μπορεί να επιλέξει την ετικέτα της εντολής που θέλει να χρησιμοποιήσει είτε πατώντας τα βέλη, είτε πληκτρολογώντας τον αριθμό που προηγείται της εντολής της αρεσκείας του :



ΕΙΚΟΝΑ 5.1 : Λειτουργία της τρίτης παραμέτρου του Constructor Command()

Για την κλάση αυτή υπάρχει μόνο μία διαθέσιμη μέθοδος, η οποία ονομάζεται toString() και επιστρέφει μια String αναπαράσταση του αντικειμένου. Ακολουθεί παράδειγμα δημιουργίας κουμπιών (commands).

```
 Form ui_holder = null;

// Declaring Command objects and initializing them to null...
 Command ok = null;
 Command quit = null;
// Instantiating the Form...
 ui_holder = new Form("User Interface - TextField");
...
 ok = new Command("Ok",Command.SCREEN,3);
 quit = new Command("Quit",Command.SCREEN,2);

// Adding Command Button to the Form
   ui_holder.addCommand(ok);
   ui_holder.addCommand(quit);
...
  // Invoking Action Listener
  ui_holder.setCommandListener(this);

  public void commandAction(Command c, Displayable d) {
 // Event handling for the Button
  if ( c == ok )
  {
  ...
  }
  if ( c == quit )
  {
   ...
  }
 }
```

Η κλάση Display

Αυτή η κλάση όπως είναι ήδη γνωστό αναπαριστά την οθόνη της φορητής συσκευής. Για να φανεί ένα αντικείμενο στην οθόνη, θα πρέπει αρχικά να ορίσουμε την display, να την «πάρουμε» και στην συνέχεια να την κάνουμε current με το αντικείμενο που δείχνει η παράμετρος της μεθόδου setCurrent. Οι μέθοδοι που είναι διαθέσιμες σε αυτήν την κλάση είναι οι :

Μέθοδος	*Λειτουργία*
getDisplay(MIDlet m)	επιστρέφει το αντικείμενο Display, το οποίο είναι μοναδικό για κάθε MIDlet.
isColor()	επιστρέφει πληροφορία σχετικά με τα χρώματα που υποστηρίζει η συσκευή.
numColors()	επιστρέφει τον αριθμό των χρωμάτων (αν η isColor() επιστρέφει true) ή τον αριθμό των graylevels (αν η isColor() επιστρέφει false) που μπορούν να αναπαρασταθούν στην οθόνη.
getCurrent()	επιστρέφει το τρέχον αντικείμενο Displayable για το συγκεκριμένο MIDlet.
setCurrent(Displayable nextDisplayable)	θέτει ένα διαφορετικό Displayable αντικείμενο να φαίνεται στην οθόνη.

ΠΙΝΑΚΑΣ 5.3 : Μέθοδοι της κλάσης Display

Ένα παράδειγμα χρήσης της κλάσης αυτής φαίνεται στο κομμάτι κώδικα που ακολουθεί, το οποίο ανήκει στην παρούσα εφαρμογή:

```
/** This method initializes UI of the application.
   */
  private void initialize() {
          getDisplay().setCurrent(imgCanvas);
  }
  /**
   * This method should return an instance of the display.
   */
  public Display getDisplay() {
    return Display.getDisplay(this);
  }
   /**
   * This method should exit the midlet.
   */
  public void exitMIDlet() {
    getDisplay().setCurrent(null);
    destroyApp(true);
    notifyDestroyed();
  }
```

Η κλάση Form

Το πακέτο lcdui περιέχει την κλάση Form. Ένα Form είναι ένα είδος Screen, πάνω στο οποίο μπορούν να προστεθούν άλλα UI αντικείμενα. Μπορούμε να δημιουργήσουμε ένα Form είτε δίνοντας ένα String που αντιπροσωπεύει τον τίτλο του Form είτε δίνοντας ένα String για τον τίτλο αλλά και έναν πίνακα με τα αντικείμενα που θέλουμε να προστεθούν στο Form. Οι διαθέσιμες μέθοδοι είναι οι εξής :

Μέθοδος	***Λειτουργία***
append(Image img)	προσθέτει στη Form ένα item που περιέχει μία εικόνα.
append(Item item)	προσθέτει ένα item στη Form.
append(String str)	προσθέτει στη Form ένα item που περιέχει μία συμβολοσειρά.
insert(int itemNum, Item item)	εισάγει ένα item σύμφωνα με την προτεραιότητα που της δίνει ο αριθμός itemNum
set(int itemNum, Item item)	θέτει ένα καινούριο item στη θέση ενός παλιού item.
get(int itemNum)	επιστρέφει το item της δεδομένης θέσης.
size()	επιστρέφει τον αριθμό των items που υπάρχουν στη Form.
setItemStateListener(ItemStateListener iListener)	θέτει τον ItemStateListener για τη Form, αντικαθιστώντας οποιονδήποτε προηγούμενο ItemStateListener.

ΠΙΝΑΚΑΣ 5.4 : Μέθοδοι της κλάσης Form

Παρακάτω ακολουθεί ένα κομμάτι κώδικα στο οποίο δημιουργείται ένα Form και αρχικοποιείται σε null. Σε αυτό το Form δίνεται ένας τίτλος String. Στη συνέχεια προστίθεται ένα TextField καλώντας την μέθοδο append().

```
// Declaring the form...
Form myform = null;
...
// Declaring the Display and initializing it as null...
private Display show = null;
...
// Getting the Display unique to this MIDlet...
show = Display.getDisplay(this);
// Initializing the form with a string title...
myform = new Form("User Interface - TextField");
// Declaring and initializing a TextField...
tx = new TextField("MyField","Type here...",70,0);
// Adding the TextField to the form...
myform.append(tx);
// Showing the form, which is a Displayable object, on the screen...
show.setCurrent(myform);
```

Η κλάση Gauge

Συνεχίζουμε με την κλάση Gauge. Μερικές φορές θέλουμε να χρησιμοποιήσουμε ένα bar προόδου για να βλέπουμε την εξέλιξη μιας διαδικασίας ή την φόρτωση ενός παιχνιδιού κ.τ.λ. Αυτός είναι και ο σκοπός αυτής της κλάσης. Χρησιμοποιείται για την δημιουργία τέτοιων bars προόδου. Οι μέθοδοι που υπάρχουν σε αυτήν την κλάση και μια υλοποίηση ενος Gauge παραδείγματος φαίνονται παρακάτω :

Μέθοδος	***Λειτουργία***
getValue()	επιστρέφει την τρέχουσα τιμή του Gauge αντικειμένου.
setValue(int value)	θέτει την τρέχουσα τιμή του Gauge αντικειμένου.
getMaxValue()	επιστρέφει την μεγαλύτερη τιμή του Gauge αντικειμένου.
setMaxValue(int value)	θέτει την μεγαλύτερη τιμή του Gauge αντικειμένου

ΠΙΝΑΚΑΣ 5.5 : Μέθοδοι της κλάσης Gauge

```
/**
This class demonstrates the use of the Gauge MIDP UI
class.
@see javax.microedition.lcdui.Gauge
*/
public class GaugeDemo extends Form
implements CommandListener
{
...
```

```
private String gauge1Label = new String("Interactive gauge");
private Gauge interactiveGauge =
new Gauge("Interactive", true, 50, 15);
private String gauge2Label = new String("Non-interactive");
private Gauge staticGauge = new Gauge("Static", false, 50, 25);
/**
Constructor.
*/
public GaugeDemo()
{
super("Gauge Demo");
append(gauge1Label);
append(interactiveGauge);
append(gauge2Label);
append(staticGauge);
addCommand(back);
setCommandListener(this);
instance = this;
}
...
}
```

Το αποτέλεσμα που φαίνεται στην οθόνη μετά την εκτέλεση του παραπάνω κώδικα είναι αυτό που φαίνεται στην εικόνα 5.2 .

ΕΙΚΟΝΑ 5.2 : Παράδειγμα για την κλάση Gauge.

Η κλάση Graphics.

Η κλάση Graphics είναι η πιο εκτενής και περιέχει πάρα πολλές μεθόδους, οι οποίες παρουσιάζονται παρακάτω. Αρκετές από τις μεθόδους αυτές χρησιμοποιήθηκαν και στην παρούσα εφαρμογή έτσι ώστε να έχουμε το αποτέλεσμα που παρουσιάστηκε στο προηγούμενο κεφάλαιο. Η κλάση Graphics παρέχει μια πολύ καλή λειτουργικότητα για την εμφάνιση 2D γραφικών σε συσκευές με περιορισμένη επεξεργαστική ισχύ αλλά και περιορισμένη μνήμη. Οι μέθοδοι της κλάσης παρουσιάζονται παρακάτω :

Μέθοδοι	***Λειτουργία***
clipRect(int x, int y, int width, int height)	Κόβει το τρέχων clip με ένα προκαθορισμένου μήκους ορθογώνιο.
copyArea(int x_src, int y_src, int width, int height, int x_dest, int y_dest, int anchor)	Αντιγράφει τα περιεχόμενα απο μία περιοχή που περικλείεται από το ορθογώνιο (x_src, y_src, width, height) σε μία άλλη περιοχή της οποίας ο δείκτης (από που ξεκινάει δηλαδή) τοποθετείται στο σημείο (x_dest, y_dest).
drawArc(int x, int y, int width, int height, int startAngle, int arcAngle)	Ζωγραφίζει το περίγραμμα ενός κυκλικού ή ελλειπτικού τόξου χρησιμοποιώντας το ήδη προκαθορισμένο χρώμα και στυλ γραμμής.
drawChar(char character, int x, int y, int anchor)	Ζωγραφίζει έναν χαρακτήρα χρησιμοποιώντας το τρέχον χρώμα και συμβολοσειρά.
drawChars(char[] data, int offset, int length, int x, int y, int anchor)	Ζωγραφίζει τους χαρακτήρες που περιέχονται στον πίνακα data[] χρησιμοποιώντας το τρέχον χρώμα και συμβολοσειρά.
drawImage(Image img, int x, int y, int anchor)	Ζωγραφίζει μία εικόνα.
drawLine(int x1, int y1, int x2, int y2)	Ζωγραφίζει μία γραμμή μεταξύ των συντεταγμένων (x1,y1) και (x2,y2) χρησιμοποιώντας το τρέχον χρώμα και στυλ γραμμής.
drawRect(int x, int y, int width, int height) void	Ζωγραφίζει το περίγραμμα ενός ορθογωνίου χρησιμοποιώντας το τρέχον χρώμα και στυλ γραμμής.
drawRegion(Image src, int x_src, int y_src, int width,	Αντιγράφει μια περιοχή ενός καθορισμένου image, την μετατρέπει(περιστροφή, αντανάκλαση) και την

int height, int transform, int x_dest, int y_dest, int anchor)	βάζει σε μια καθορισμένη τοποθεσία.
drawRoundRect(int x, int y, int width, int height, int arcWidth, int arcHeight)	Ζωγραφίζει το περίγραμμα ενός καθορισμένου ορθογωνίου με στογγυλεμένη γωνία χρησιμοποιώντας το τρέχον χρώμα και στυλ γραμμής .
drawString(String str, int x, int y, int anchor)	Ζωγραφίζει ένα καθορισμένο String χρησιμοποιώντας το τρέχον χρώμα και γραμματοσειρά.
drawSubstring(String str, int offset, int len, int x, int y, int anchor)	Ζωγραφίζει ένα καθορισμένο String χρησιμοποιώντας το τρέχον χρώμα και γραμματοσειρά.
fillArc(int x, int y, int width, int height, int startAngle, int arcAngle)	Γεμίζει με το τρέχον χρώμα ένα καθορισμένο κυκλικό ή ελλειπτικό τόξο.
fillRect(int x, int y, int width, int height)	Γεμίζει με το τρέχον χρώμα ένα καθορισμένο ορθογώνιο.
fillRoundRect(int x, int y, int width, int height, int arcWidth, int arcHeight)	Γεμίζει με το τρέχον χρώμα ένα καθορισμένο ορθογώνιο με στρογγυλεμένες γωνίες.
fillTriangle(int x1, int y1, int x2, int y2, int x3, int y3)	Γεμίζει με το τρέχον χρώμα ένα καθορισμένο τρίγωνο.
getBlueComponent()	Επιστρέφει τον αριθμό που αντιπροσωπεύει την ποσότητα του μπλέ χρώματος στο τρέχον χρώμα.
getClipHeight()	Επιστρέφει το ύψος της τρέχουσας clipping περιοχής.(το clip είναι ένα σύνολο από pixels)
getClipWidth()	Επιστρέφειτο πλάτος της τρέχουσας clipping περιοχής.
getColor()	Επιστρέφει το τρέχον χρώμα
getFont()	Επιστρέφει την τρέχουσα γραμματοσειρά.
getGrayScale()	Επιστρέφει την τρέχουσα τιμή του GrayScale του χρώματος που χρησιμοποιείται για τις εργασίες φωτοσκίασης.
getGreenComponent()	Επιστρέφει τον αριθμό που αντιπροσωπεύει την ποσότητα του πράσινου χρώματος στο τρέχον χρώμα.
getRedComponent()	Επιστρέφει τον αριθμό που αντιπροσωπεύει την ποσότητα του κόκκινου χρώματος στο τρέχον χρώμα.
getStrokeStyle()	Επιστρέφει το στύλ της γραμμής που χρησιμοποιείται για τις εργασίες σχεδίασης.
getTranslateX()	Επιστρέφει την X συντεταγμένη της μετετοπισμένης αρχής των συντεταγμένων γι αυτό το γραφικό περιβάλλον.
getTranslateY()	Επιστρέφει την Y συντεταγμένη της μετετοπισμένης αρχής των συντεταγμένων γι αυτό

	το γραφικό περιβάλλον.
setClip(int x, int y, int width, int height)	Τοποθετεί το τρέχον clip στο ορθογώνιο που καθορίζεται από τις δεδομένες συντεταγμένες.
setColor(int RGB)	Θέτει καινούριο χρώμα που καθορίζεται από τις RGB τιμές.
setColor(int red, int green, int blue)	Θέτει καινούριο χρώμα που καθορίζεται από τις RGB τιμές.
setFont(Font font)	Θέτει καινούρια γραμματοσειρά.
setGrayScale(int value)	Θέτει καινούρια τιμή στο grayscale.
setStrokeStyle(int style)	Θέτει καινούριο στυλ γραμμής για όλες τις εργασίες σχεδίασης.
translate(int x, int y)	Μεταφέρει την αρχή των συντεταγμένων του γραφικού περιβάλλοντος στο σημείο (x, y) στο τρέχον σύστημα συντεταγμένων.

ΠΙΝΑΚΑΣ 5.6 : Μέθοδοι της κλάσης Graphics

Η συνάρτηση paint() της παρούσας εφαρμογής δέχεται σαν όρισμα ένα αντικείμενο της κλάσης Graphics, οπότε χρησιμοποιεί αρκετές από τις μεθόδους που προαναφέρθηκαν, ο κώδικας της οποίας παρατίθεται παρακάτω :

```
protected void paint(Graphics g)  {

  if (im != null)  {
    // Clear the background
    g.setColor(255, 255, 255);
    g.fillRect(0, 0, getWidth(), getHeight());
    // Translate coordinates
    g.translate(translatex, translatey);
    // Always draw at 0,0
    g.drawImage(im, 0, 0, Graphics.LEFT | Graphics.TOP);
  }
  // draw all points
  PointOfInterest p;
  for (int i = 0; i < allPoints.size(); ++i) {
    p = (PointOfInterest)allPoints.elementAt(i);

    //draw point
    if (p.getVisible()) {
      g.setColor(p.getColor().getRed(), p.getColor().getGreen(), p.getColor().getBlue());
      g.fillRect((int)p.getPos().getX(), (int)p.getPos().getY(), p.getPointWidth(),p.getPointHeight());
```

```
            //draw label
             g.setColor(0,0,0);
             g.setFont(Font.getFont(Font.FACE_SYSTEM,Font.STYLE_BOLD,Font.SIZE_MEDIUM));

             g.drawString(p.getLabel(), (int)p.getPos().getX()+2, (int)p.getPos().getY()-3,
Graphics.TOP|Graphics.LEFT);
          }
       }
    }
```

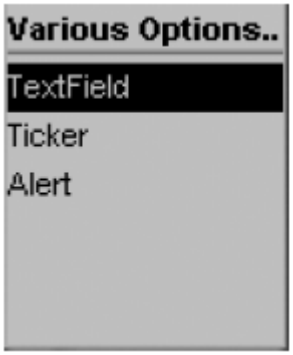
Η κλάση List

Η κλάση List χρησιμοποιείται για να δημιουργήσουμε αντικείμενα τα οποία μπορούν να επιλεγούν στο user interface της κάθε εφαρμογής. Δημιουργείται δηλαδή μία λίστα από αντικείμενα τα οποία μπορούν να επιλεγούν. Οι μέθοδοι που υπάρχουν σε αυτήν την κλάση είναι οι μέθοδοι που υλοποιούνται στο Interface Choice.

Μέθοδοι	***Λειτουργία***
size()	Επιστρέφει τον αριθμό των elements που υπάρχουν στη λίστα.
getString(int element Num)	Επιστρέφει την συμβολοσειρά του element στο οποίο αναφέρεται ο καθορισμένος ακέραιος αριθμός.
getImage(int elementNum)	Επιστρέφει την εικόνα του element στο οποίο αναφέρεται ο καθορισμένος ακέραιος αριθμός.
append(String stringPart, Image imagePart)	Προσθέτει ένα element στη λίστα.
insert(int elementNum, String stringPart, Image imagePart)	Εισάγει ένα element στην λίστα σύμφωνα με τον αριθμό προτεραιότητας elementNum.
delete(int elementNum)	Καταστρέφει το element που αντιστοιχεί στον καθορισμένο αριθμό.

setElement(int elementNum, String stringPart, Image imagePart)	Θέτει καινούρια συμβολοσειρά και εικόνα στο element που υποδεικνύεται από τον καθορισμένο αριθμό elementNum αντικαθιστώντας τα παλιά.
isSelected(int elementNum)	Επιστρέφει μία boolean μεταβλητή που ποδεικνύει αν το element είναι επιλεγμένο ή όχι.
getSelectedIndex()	Επιστρέφει τον δείκτη του element στη λίστα που έχει επιλεγεί.
getSelectedFlags(boolean[] selectedArray_return)	Εξετάζει την κατάσταση της λίστας και επιστρέφει τις καταστάσεις των elements και τις αποθηκεύει σε μορφή Boolean στον πίνακα `selectedArray_return`.
setSelectedIndex(int elementNum, boolean selected)	Θέτει την επιλεγμένη κατάσταση ενός element.
setSelectedFlags(boolean[] selectedArray)	Θέτει τις επιλεγμένες καταστάσεις όλων των elements της λίστας.

ΠΙΝΑΚΑΣ 5.7 : Μέθοδοι της κλάσης List.



ΕΙΚΟΝΑ 5.3 : List αντικειμένων

Η κλάση StringItem

Η κλάση StringItem χρησιμοποιείται για να προσθέτουμε συμβολοσειρές σε ένα Form. Έχει μόλις δύο μεθόδους. Η μία επιστρέφει την συμβολοσειρά που προστέθηκε και είναι η getText(), και η άλλη θέτει την συμβολοσειρά(το κείμενο) και είναι η setText().Ο κώδικας που ακολουθεί δείχνει ένα παράδειγμα της χρήσης των μεθόδων αυτών :

```
// Declaring variable for StringItem class.
StringItem string_item = null;
// Initializing the variable

// parameter 1 .. Label to be displayed
// parameter 1 .. Text along with the Label.

string_item = new StringItem("User Entered ..", "");


// Adding String item to the form (place holder)
ui_holder.append(string_item);


// Changing the Textual context of the StringItem

  string_item.setText(textcheck.getString());
```

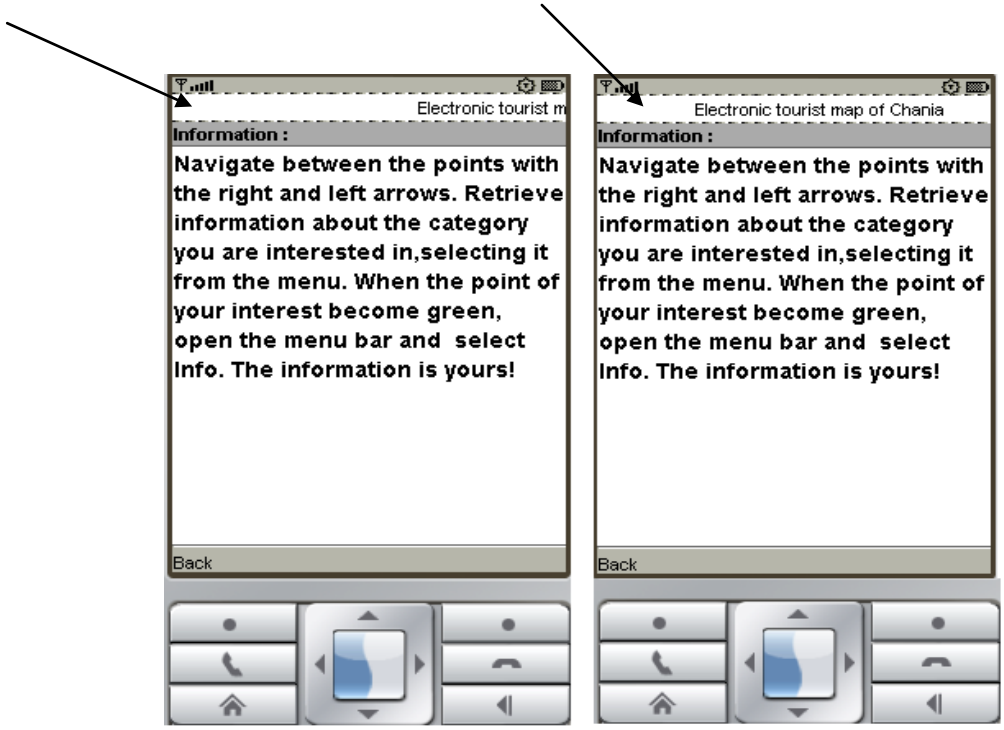
Η κλάση Ticker

Η κλάση αυτή μας παρέχει ένα τρόπο να δημιουργούμε κυλιόμενα μηνύματα στο πάνω μέρος της οθόνης της φορητής μας συσκευής. Οι μέθοδοι που περιλαμβάνει η κλάση αυτή είναι η getString() η οποία επιστρέφει το μήμυμα του Ticker αντικειμένου, και η setString() που θέτει καινούριο μήνυμα στο Ticker αντικαθιστώντας το παλιό. Η κλάση Ticker έχει χρησιμοποιηθεί και στην παρούσα εφαρμογή, ενώ ο κώδικας που υλοποιεί το δικό μας κυλιόμενο μήνυμα είναι ο εξής :

```
...
...
 private String str = "Electronic tourist map of Chania";
 private Ticker ticker = new Ticker(str);

 private StringItem s1 = new StringItem("","");


 public MyForm1(MapMidlet midlet)
 {

    super("Information :");
    this.midlet = midlet;
    back = new Command("Back",Command.BACK,1); //dhmiourgia ths entolhs Back
    append(s1);
```

```
    setTicker(ticker);     //pros8esh toy Ticker (kuliomeno mhnuma)

     addCommand(back);     // pros8esh ths entolhs back sto Displayable
     setCommandListener(this);  // 8etetai o CommandListener se leitourgia
  }
...
...
```

Η εικόνα 5.4 δείχνει το κυλιόμενο μήνυμα της εφαρμογής μας που δημιουργήθηκε με την βοήθεια του παραπάνω κώδικα:



ΕΙΚΟΝΑ 5.4 : Παράδειγμα για την κλάση Ticker

Η κλάση TextField

Η κλάση αυτή μας παρέχει την δυνατότητα δημιουργίας ενός πεδίου κειμένου στο οποίο ο χρήστης μπορεί να εισάγει ένα δικό του κείμενο. Οι παράμετροι που απαιτούνται είναι η ετικέτα, το αρχικό κείμενο, το μέγιστο μέγεθος του κειμένου και οι περιορισμοί εισόδου οι οποίοι συνοψίζονται στους εξής :

- TextField.ANY : ο χρήστης μπορεί να εισάγει ό,τι κείμενο θέλει.

- TextField.EMAILADDR : ο χρήστης επιτρέπεται να εισάγει e-mail διευθύνσεις.
- TextField.PASSWORD : το κείμενο που εισάγεται δεν είναι ορατό.
- TextField.PHONEMUMBER : ο χρήστης μπορεί να εισάγει μόνο αριθμούς τηλεφώνων.

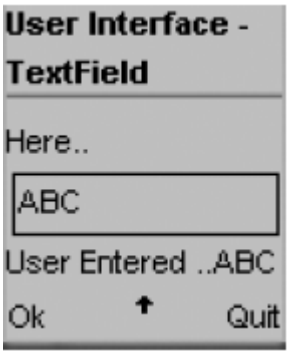
Οι μέθοδοι που μπορούμε να χρησιμοποιήσουμε με αυτήν την κλάση φαίνονται στον πίνακα που ακολουθεί :

Μέθοδος	***Λειτουργία***
getString()	Επιστρέφει το κείμενο του TextField σε μορφή String.
setString(String text)	Θέτει καινούρια περιεχόμενα στο TextField αντικαθιστώντας τα παλιά.
getChars(char[] data)	Επιστρέφει τα περιεχόμενα του TextField σε έναν πίνακα από Chars.
setChars(char[] data, int offset, int length)	Θέτει τα περιεχόμενα του TextField από έναν πίνακα τύπου Char, αντικαθιστώντας τα παλιά.
size()	Επιστρέφει τον αριθμό των χαρακτήρων που είναι αποθηκευμένα στο TextField.
setSize(int size)	Ορίζει τον μέγιστο αριθμό χαρακτήρων από τον οποίο πρέπει να αποτελείται ένα TextField.
getConstraints()	Επιστρέφει τον τύπο των περιορισμών εισόδου του TextField.
setConstraints(int constraints)	Θέτει τον περιορισμό εισόδου του TextField.

ΠΙΝΑΚΑΣ 5.8 : Μέθοδοι της κλάσης TextField.

Ο κώδικας που ακολουθεί προσθέτει ένα TextField σε ένα Form :

```
// Declaring a variable of the type TextField
 TextField textcheck = null;

 textcheck = new TextField("Enter the Text Here..","",50,0);

// Adding TextField to the form (place holder)
  ui_holder.append(textcheck);

// Event Handling..

public void commandAction(Command c, Displayable d) {
// Event handling for the Button
   if ( c == ok )
   {
    System.out.println(textcheck.getString());
   }
 }
```



EIKONA 5.5 : TextField

Η κλάση TextBox

Η κλάση αυτή μας δίνει την δυνατότητα δημιουργίας ενός πεδίου κειμένου, στο οποίο ο χρήστης μπορεί να γράψει ένα δικό του κείμενο. Όπως ακριβώς δηλαδή και στην περίπτωση της κλάσης TextField. Η μόνη διαφορά που έχουν αυτές οι δύο κλάσεις έγκειται στο γεγονός ότι η κλάση TextField είναι Item, οπότε και πρέπει να προστεθεί σε ένα Form. Το TextBox είναι ένα component το οποίο μπορεί απευθείας να προστεθεί στο Screen. Οι περιορισμοί εισόδου που ισχύουν στα

αντικείμενα της TextField ισχύουν και για τα αντικείμενα της TextBox, όπως επίσης και οι μέθοδοι.

Ακολουθεί ένα κομμάτι κώδικα το οποίο υλοποιεί ένα TextBox, το οποίο ονομάζεται ourBox:

```
String s = "Hello From J2ME";
...
private Display ourDisplay;  // Declaring the display
private Form ourForm = null;
...
ourDisplay = Display.getDisplay(this);
...
ourForm = new Form("Our First");
TextBox ourBox = new TextBox("J2ME Application", s, 256, 0);
ourForm.append(ourBox);
...
ourForm.setListener(this);

ourDisplay.setCurrent(ourForm);
```

5.2.6 Διαχείριση αρχείων

Στην ροή της παραγράφου αυτής δίνονται κάποιες συμβουλές για την ανάγνωση, δημιουργία και διαχείριση ενός αρχείου. Τα αρχεία ισοδυναμούν με την δυναμική υλοποίηση μιας εφαρμογής. Όπως έχει αναφερθεί η παρούσα εφαρμογή έχει υλοποιηθεί με δυναμικό τρόπο γεγονός που σημαίνει ότι αν κάποιος θέλει να εισάγει καινούρια στοιχεία ή να διαγράψει κάποια από τα ήδη υπάρχοντα, σε έναν κώδικα που έχει ήδη υλοποιηθεί, αντί να «πειράξει» τον κώδικα, κάνει τις εισαγωγές ή τις διαγραφές από ένα αρχείο. Πώς το κάνουμε όμως αυτό να δουλεύει?

Η J2ME μας παρέχει το java.io Package το οποίο περιέχει κλάσεις και μεθόδους που μας δίνουν την δυνατότητα είτε να διαβάζουμε από ένα

αρχείο, είτε να γράφουμε σε αυτό (π.χ. έξοδος της εφαρμογής: ένα αρχείο που είναι αποτέλεσμα της εκτέλεσης της εφαρμογής.) Στην παράγραφο αυτή θα παρουσιαστούν οι κλάσεις και οι μέθοδοι που χρησιμοποιήθηκαν για την υλοποίηση του ηλεκτρονικού τουριστικού οδηγού της πόλης των Χανίων και που πραγματοποιούν ανάγνωση από ένα αρχείο .

Η κλάση InputStream

Η κλάση αυτή είναι ίδια με αυτήν της J2SE και ανήκει στο java.io Paclage. Χρησιμοποιείται για τις ροές δεδομένων εισόδου. Είναι μια superclass των κλάσεων που αναπαριστούν μια ροή δεδομένων εισόδου που αποτελείται από bytes. Οι εφαρμογές που χρειάζεται να ορίσουν μια υπο – κλάση της InputStream πρέπει πάντα να παρέχουν και μία μέθοδο που να επιστρέφει το επόμενο byte εισόδου. Κάτι τέτοιο δεν χρησιμοποιήθηκε στην παρούσα εφαρμογή, και επομένως δεν θα αναλυθεί περαιτέρω.

Οι μέθοδοι που παρέχει η κλάση InputStream φαίνονται στον πίνακα που ακολουθεί :

Μέθοδοι της κλάσης InputStream	
`available``()`	Επιστρέφει έναν αριθμό από bytes τα οποία μπορούν να διαβαστούν ή να προσπεραστούν.
`close``()`	Κλείνει την ροή δεδομένων εισόδου (input Stream) και απελευθερώνει όσους πόρους του συστήματος είχαν δεσμευθεί με το stream.
`mark``(int readlimit)`	Σημειώνει την τρέχουσα θέση μέσα στο input strem.
`markSupported``()`	Ελέγχει αν το παρόν input stream υποστηρίζει τις μεθόδους `mark` και `reset`.
`read``()`	Διαβάζει το επόμενο byte από το input stream.

read(byte[] b)	Διαβάζει έναν αριθμό από bytes του από το input stream και τα αποθηκεύει σε έναν πίνακα – buffer b.
read(byte[] b, int off, int len)	Διαβάζει μέχρι έναν αριθμό len bytes δεδομένων από το input stream μέσα σε έναν πίνακα από bytes.
skip(long n)	Προσπερνά και απορρίπτει n bytes δεδομένων από tos input stream.

ΠΙΝΑΚΑΣ 5.9 : Μέθοδοι της κλάσης Input Stream

Μερικές ιδιαίτερες μέθοδοι

Η μέθοδος **getClass()** είναι μία μέθοδος που περιλαμβάνει η κλάση Object η οποία είναι υπερκλάση όλων των κλάσεων, ενώ η **getResourceAsStream(fileName)** είναι μία μέθοδος που περιλαμβάνεται στην κλάση Class η οποία αναπαριστά όλες τις κλάσεις και τα interfaces σε μία Java εφαρμογή.

getClass().getResourceAsStream(fileName)

Οι δύο αυτές μέθοδοι χρησιμοποιούμενες μαζί επιστρέφουν το Input Stream με το όνομα που δίνει η παράμετρος filename .

Ο κώδικας που ακολουθεί ανήκει στην εφαρμογή για την οποία γίνεται λόγος σε όλο το παρόν κείμενο και δείχνει την χρήση κάποιων από όσα αναφέρθηκαν παραπάνω :

```
private void createPointsFromTextFile(String fileName) {
    Vector lines = new Vector();
    String typeCode="", typeName="";

    InputStream is = this.getClass().getResourceAsStream(fileName);

    String mode= "";
    try {
      StringBuffer sb = new StringBuffer();
      String line, tmp, type="", msg="", label="";
      int x = -1, y = -1;
      int r=0, g=0, b=0;
      int width=0, height=0;
      int chr;
```

```
/** Read file and insert its rows in a vector */
        while ( (chr = is.read()) != -1 ) {
          if ( (char)chr != '\n') {
            sb.append((char) chr);
          }
          else {
            line = sb.toString();
            lines.addElement(line);
            sb = new StringBuffer();
          }
        }
.....

catch (Exception e) {
        System.out.println("Unable to create stream");
      }
    }
```

Για καλύτερη κατανόηση των όσων αναφέρθηκαν στο εγχειρίδιο αυτό ανατρέψτε στην βιβλιογραφία που παρατίθεται στο τέλος του κειμένου. Το εγχειρίδιο αυτό δημιουργήθηκε για να δώσει όλες τις κλάσεις και τις μεθόδους που μπορεί να χρησιμοποιήσει κάποιος για να υλοποιήσει μια τέτοια εφαρμογή ή να βελτιστοποιήσει την ήδη υπάρχουσα!

6

ΜΕΛΛΟΝΤΙΚΕΣ ΕΠΕΚΤΑΣΕΙΣ

6.1 Εισαγωγή

Στο κεφάλαιο αυτό αναφέρονται κάποιες επεκτάσεις ή βελτιστοποιήσεις που μπορούν να γίνουν πάνω στην παρούσα εφαρμογή έτσι ώστε να επιτευχθεί μεγαλύτερη λειτουργικότητα της εφαρμογής.

6.2 Μελλοντική δουλειά

Το σύστημα το οποίο υλοποιήθηκε μπορούμε να πούμε ότι αποτελεί μία εναλλακτική προσέγγιση στην δημιουργία εφαρμογών που έχουν ως αντικείμενο ενασχόλησης την παρουσίαση ενός τουριστικού οδηγού. Είναι σαφές ότι μπορούν να γίνουν αρκετές επεκτάσεις πάνω σε αυτήν την αρχική έκδοση της εφαρμογής, έτσι ώστε το σύστημα αυτό να μπορεί να αποτελέσει έναν πλήρη ηλεκτρονικό τουριστικό οδηγό της πόλης των Χανίων και όχι μόνο.

Από πλευράς βελτιώσεων του υπάρχοντος κώδικα, μπορούν να γίνουν οι εξής επεκτάσεις :

- Δυνατότητα εισαγωγής στην εφαρμογή, περισσότερων χαρτών (διαφορετικών πόλεων) με την ανάλογη εισαγωγή σημείων και κατηγοριών - γεγονός που σημαίνει την προσθήκη κώδικα που διαφοροποιεί τους χάρτες μεταξύ τους χωρίς να χάνεται η πληροφορία που αφορά τα σημεία ενδιαφέροντος τα οποία φέρει πάνω του ο καθένας από αυτούς.
- Βελτίωση της απεικόνισης των χαρτών στην οθόνη, καθώς επίσης και παροχή της δυνατότητας μεγέθυνσης (zoom) κάποιου «κομματιού» του χάρτη, το οποίο εκτιμάται ότι θα επιλεγεί από τον χρήστη. Για παράδειγμα, αν στο παρόν σύστημα, ο χρήστης επιλέξει ένα οποιοδήποτε σημείο ενδιαφέροντος που ανήκει σε μία από τις υπάρχουσες κατηγορίες της εφαρμογής, με την προτεινόμενη βελτίωση, να μπορεί να κάνει zoom σε αυτό το κομμάτι του χάρτη, έτσι ώστε να μπορεί να προσανατολιστεί καλύτερα βλέποντας καθαρά τις οδούς που βρίσκονται κοντά στο σημείο της επιλογής του.
- Μείωση του μεγέθους της εφαρμογής με μία απλή μείωση του μεγέθους των αρχείων με πιθανή συμπίεση τους, που συνεπάγεται και αλλαγή στον κώδικα έτσι ώστε να διαβάζει συμπιεσμένα αρχεία.
- Προσθήκη επιπλέον χαρακτηριστικών στην εφαρμογή, όπως για παράδειγμα προσθήκη φωτογραφιών των σημείων οι οποίες θα παρουσιάζονται κατά την επιλογή των σημείων αυτών από το χρήστη, έτσι ώστε να μπορεί να γίνεται ταυτοποίηση του σημείου που επιλέχτηκε στο χάρτη με το σημείο στο οποίο δύναται να βρεθεί στην πραγματικότητα ο χρήστης.

- Γενική βελτίωση στο User Interface του χρήστη, ώστε να γίνει όσο το δυνατόν πιο εύχρηστο.
- Τέλος προσθήκη δυνατότητας λήψης πληροφορίας μετά από σύνδεση με ένα δίκτυο. Για παράδειγμα αν θέλει ο χρήστης να γνωρίζει τις καιρικές συνθήκες της πόλης που δείχνει ο χάρτης, να υπάρχει στο menu των επιλογών της εφαρμογής μια τέτοια εντολή, η οποία μετά από επιλογή της από τον χρήστη, να καθιστά δυνατή μία σύνδεση με ένα δίκτυο, να λαμβάνει την πληροφορία και να την παρέχει στο χρήστη της εφαρμογής . Όλα αυτά βέβαια να γίνονται στον βέλτιστο χρόνο απόκρισης.

Αυτές είναι ίσως οι κυριότερες επεκτάσεις που μπορούν να γίνουν στο μέλλον, ώστε να έχουμε πιο εμπλουτισμένους, πιο εύχρηστους και πιο ολοκληρωμένους ηλεκτρονικούς τουριστικούς οδηγούς.

7

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. Γιώργος Λιακέας, « Εισαγωγή στην JAVA2 – Ένας ολοκληρωμένος και εύχρηστος οδηγός της γλώσσας», Εκδόσεις Κλειδάριθμος, 2003.

2. Rogers Cadenhead, «Μάθετε την JAVA2 σε 24 ώρες», Εκδόσεις Γκιούρδας, web site : http://www.java24hours.com , 2003.

3. Vartan Piroumian, "Wireless J2ME™ Platform Programming", March 25, 2003, ISBN : 0-13-044914-8.

4. James Keogh, "J2ME: The Complete Reference", 2003, ISBN 0-07-222710-9.

5. Michael Morrison, "Teach Yourself Wireless Java with J2ME in 21 days", June 2001, ISBN: 0-672-32142-4.

6. DreamTech Software Team, "Wireless Programming with J2ME™", web site : www.dreamtechsoftware.com, 2002, ISBN: 0-7645-4885-9.

7. John W. Muchow, "Core J2ME™ Technology & MIDP", December 21, 2001, ISBN: 0-13-066911-3.

8. James White & David Hemphill, "Java 2 Micro Edition – Java in small things", 2002, ISBN 1-930110-33-2.

9. Qusay Mahmoud, "Learning Wireless Java", December 2001, ISBN: 0-59600-243-2.

10. Vikram Goyal, "Pro Java ME MMAPI : Mobile Media API for Java Micro Edition", 2006, ISBN-13: 978-1-59059-639-5, ISBN-10: 1-59059-639-0.

11. Rogger Riggs, Antero Taivalsaari, Jim Van Peursem, Jyri Huopaniemi, Mark Patel, Aleksi Uotila, Jim Holliday, " Programming Wireless Devices with the Java 2 Platform, Micro Edition, Second Edition", June 13, 2003, ISBN: 0-321-19798-4.

12. Kim Topley, "J2ME in a Nutshell", web site: http://www.oreilly.com/catalog/j2meanut/, March 2002, ISBN: 0-596-00253-X.

13. Michael Kroll, Stefan Haustein, "Java™ 2 Micro Edition Application Development", June 25, 2002, ISBN: 0-672-32095-9.

14. Neil Rhodes & Julie McKeehan, "Palm Programming: The Developer's Guide", December 1998, ISBN: 0-91126-109-5.

15. Martin de Jode, “Programming Java 2 Micro Edition on Symbian OS – A developer’s guide to MIDP 2.0”, 2004, ISBN: 1-56592-525-4 .

16. Nicholas Pleis, “Palm OS Game Programming”, 2002.

17. David Fox, Roman Verhosek, “Micro Java™ Game Development”, April 18, 2002.

18. Tino Pyssysalo, “Programming for the Series 60 Platform Symbian OS”, 2003.

19. Daryl Wilding-McBride, “Java Development on PDAs”, June 05, 2003.

20. Jonathan Knudsen, “Wireless Java Developing with J2ME”, 2003

21. Cynthia Bloch, Annette Wagner, “MIDP Style Guide for the Java™ 2 Platform, Micro Edition”, June 10, 2003

22. Michael Juntao Yuan, “Enterprise J2ME: Developing Mobile Java Applications”, October 23, 2003.

23. Ralph Barbagallo, “Wireless Game Development in Java with MIDP 2.0”, 2004.

24. Michael Kenteris, Damianos Gavalas, Daphne Economou, “Mobile Electronic Guides for the Masses: Optimizing Tourists Mobile Devices” (paper), September 27, 2007.

25. Γαβαλάς Δαμιανός, «Δίκτυα Υπολογιστών – Διάλεξη #10 : J2ME», Πανεπιστήμιο Αιγαίου – www.aegean.gr .

26. Σταυρουλάκης Γεώργιος, «Εφαρμογές πλοήγησης για φορητές συσκευές.» http://www.dbnet.ece.ntua.gr/pubs/uploads/DIPL-2006-18.pdf

27. www.developer.com/java/j2me/article.php/10934_1561591_6

28. http://java.sun.com/j2me/

29. http://java.sun.com/j2me/docs/

30. http://forum.java.sun.com/forum.jsp?forum=50

31. http://wireless.java.sun.com/

32. http://www.javamobiles.com/

33. http://www.oreilly.com/catalog/j2meanut/.

34. www.**netbeans**.org

35. www.**netbeans**.org/kb/50/mobility.html

36. www.**netbeans**.org/kb/trails/mobility.html

8

Μερικά J2ME Περιβάλλοντα Προγραμματισμού.

Στο κεφάλαιο αυτό παρουσιάζονται ονομαστικά κάποια άλλα J2ME περιβάλλοντα προγραμματισμού, καθώς και οι ηλεκτρονικές διευθύνσεις από τις οποίες μπορούν όσοι ενδιαφέρονται, να κατεβάσουν και να γνωρίσουν τα προγραμματιστικά αυτά περιβάλλοντα.

1. **The J2ME Wireless Toolkit – www.sun.com**

2. **NetBeans Mobility Pack 5.0** - www.**netbeans**.org

3. **Borland JBuilder MobileSet** - http://www.borland.co.uk/jbuilder/.

4. **Nokia Developer's Suite** - http://americas.forum.nokia.com/java/.

5. **Siemens Mobility Toolkit** - http://www.siemens-mobile.de/mobile/

6. **Sony Ericsson J2ME SDK –** http://developer.sonyericsson.com