Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων

Πτυχιακή εργασία

Τίτλος: Ανάπτυξη Εφαρμογής Διαχείρισης Συλλόγου

Μαρίνα Κουντάκη (ΑΜ: 1093)

Επιβλέπων καθηγητής : Μαλάμος Αθανάσιος

Επιτροπή Αξιολόγησης : Βασιλάκης Κωνσταντίνος
Μαλάμος Αθανάσιος
Παχουλάκης Ιωάννης

Ημερομηνία παρουσίασης:

Abstract

The purpose of this paper was to create an application for the management of an association. The following parts were implemented :

- The interface of the application in Sun Java.
- The analysis and implementation of the Database in MySql.
- The implementation of Import and Edit member data routines.
- The implementation of Searching Members routines.
- The implementation of Income - Expenses routines.
- The implementation of Fund flow analysis routines.
- Printer Output

It was developed in Sun Java a programming language which provides independence of operating system and platform and the Database was in MySQL the most popular open source DB which runs in more then 20 platforms.

We will refer at both tools with plenty details as well as the way of installation and usage. We will get familiar with the design principles and the operation of a database. We will talk about NetBeans and how to use it to create a project. We will see in details how to use the application and its components.

Περίληψη

Το αντικείμενο της Πτυχιακής ήταν η δημιουργία μίας εφαρμογής με σκοπό τη διαχείριση των εργασιών ενός Συλλόγου. Συγκεκριμένα υλοποιήθηκαν τουλάχιστον οι παρακάτω επεκτάσεις:

- Το σχεδιαστικό κομμάτι της εφαρμογής σε Sun Java.
- Η ανάλυση και δημιουργία της βάσης σε MySQL.
- Η δημιουργία βασικών ρουτινών Εισαγωγής & Τροποποίησης στοιχείων μελών.
- Η δημιουργία βασικών ρουτινών αναζήτησης μελών.
- Η δημιουργία βασικών ρουτινών Καταχώρισης Εσόδων – Εξόδων
- Η δημιουργία βασικών ρουτινών ανάλυσης ταμείου.
- Εκτυπώσεις

Η ανάπτυξη έγινε σε Sun Java μια γλώσσα που παρέχει ανεξαρτησία του λειτουργικού συστήματος και πλατφόρμας και η βάση δεδομένων με MySQL την πιο δημοφιλή βάση δεδομένων ανοικτού κώδικα η οποία λειτουργεί σε περισσότερες από 20 πλατφόρμες.
Θα αναφερθούμε και στα δύο εργαλεία με αρκετές λεπτομέρειες καθώς και τον τρόπο με τον οποίο εγκαθίστανται και δουλεύουν. Θα εξοικειωθούμε με τις αρχές σχεδιασμού και λειτουργίας βάσης δεδομένων Θα γίνει αναφορά στο εργαλείο NetBeans και στην δημιουργία ενός project με την χρήση του. Θα κάνουμε αναλυτική περιγραφή της χρήσης και των επιλογών της εφαρμογής αλλά και των δυνατοτήτων της.

Πίνακας Περιεχομένων

Πίνακας Εικόνων

Λίστα Πινάκων

1. Εισαγωγή

Οι εφαρμογές υπολογιστών που αναπτύσσονται τα τελευταία χρόνια, οποιοδήποτε και αν είναι το αντικείμενό τους και το είδος χρηστών τους, σχεδόν σίγουρα θα τρέχουν σε πολλές μηχανές. Η αυξανόμενη σημασία των υπολογιστών θέτει νέες απαιτήσεις από τα προγραμματιστικά εργαλεία και δημιουργεί απαιτήσεις για ένα νέο και συνεχώς αυξανόμενο σύνολο εφαρμογών.

Το αναπτυσσόμενο λογισμικό απαιτείται συνήθως να λειτουργεί σωστά σε πολλά διαφορετικά περιβάλλοντα και αρχιτεκτονικές υπολογιστών, καθώς και να συνεργάζεται με πολλές εφαρμογές. Το λογισμικό πρέπει να εκμεταλλεύεται τις δυνατότητες της σύγχρονης δομής υπολογιστικών συστημάτων και να είναι σε θέση να προσπελαύνει τις κατανεμημένες πηγές πληροφοριών, να συνδυάζει τις αντλούμενες πληροφορίες και να τις παρουσιάζει στο χρήστη σε κατάλληλη μορφή.

Το πρόβλημα που δημιουργείται είναι ότι οι απαιτήσεις για μεταφερσιμότητα είναι συνήθως αντικρουόμενες, ενώ στο θέμα της ασφάλειας δεν έχει δοθεί η απαραίτητη σημασία. Υπάρχουν γλώσσες προγραμματισμού οι οποίες είναι μεταφέρσιμες αλλά αργές λόγω του ότι τα προγράμματα ερμηνεύονται αντί να μεταγλωττίζονται. Οι γλώσσες αυτές είναι αρκετά διαδεδομένες τόσο λόγω της υψηλής λειτουργικότητάς τους όσο και τη μεταφερσιμότητα τους. Επίσης, υπάρχουν γλώσσες προγραμματισμού οι οποίες είναι γρήγορες αλλά η ταχύτητά τους απορρέει από το ότι είναι σχεδιασμένες για συγκεκριμένες υπολογιστικές αρχιτεκτονικές.

Από μεθοδολογικής πλευράς η ανάπτυξη λογισμικού τα τελευταία χρόνια έχει προσανατολιστεί κυρίως προς τον αντικειμενοστραφή προγραμματισμό (object-oriented programming), ο οποίος επιχειρεί να δαμάσει τη συνεχώς αυξανόμενη πολυπλοκότητα ανάπτυξης λογισμικού. Σύμφωνα με τον αντικειμενοστραφή προγραμματισμό το λογισμικό δομείται σε αυτόνομες μονάδες οι οποίες έχουν σαφή λειτουργικότητα και διαπροσωπεία.

Η Java είναι μια γλώσσα προγραμματισμού και επιχειρεί να δώσει λύση στα προβλήματα που αναφέρθηκαν παραπάνω. Αναπτύχθηκε από την εταιρεία Sun και στο χρόνο από την ανάπτυξή της έχει γνωρίσει αρκετά μεγάλη διάδοση. Αρχικά ήταν προσανατολισμένη στην ανάπτυξη λογισμικού για ηλεκτρονικές συσκευές οικιακής χρήσης, στη συνέχεια όμως εξελίχθηκε σε μια ολοκληρωμένη γλώσσα η οποία έχει αρκετά από τα χαρακτηριστικά των μοντέρνων γλωσσών προγραμματισμού και υποστηρίζει τον αντικειμενοστραφή προγραμματισμό. Η επιτυχία της γλώσσας έγκειται στο ότι μπορεί να χρησιμοποιηθεί για προγραμματισμό ασφαλών, υψηλής απόδοσης εφαρμογών οι οποίες μπορούν να τρέξουν αυτούσιες σε διαφορετικά προγραμματιστικά περιβάλλοντα και αρχιτεκτονικές, καθώς και ότι παρέχει τη δυνατότητα μεταφοράς δυναμικού περιεχομένου σε εφαρμογές πολυμέσων.

Τέλος ένας πολύ σημαντικός παράγοντας της Java είναι ότι διατίθεται δωρεάν. Παρακάτω αναφέρονται αναλυτικά τα χαρακτηριστικά της Java.

ΕΦΑΡΜΟΓΗ ΔΙΑΧΕΙΡΙΣΗΣ ΣΥΛΛΟΓΟΥ

Η **Εφαρμογή Διαχείρισης Συλλόγου** αποτελεί ένα λογισμικό διαχείρισης βάσης δεδομένων.Είναι μία εφαρμογή η οποία έχει τη δυνατότητα να προσαρμοστεί στις ανάγκες του εκάστοτε συλλόγου, σωματείου, επαγγελματικής ένωσης κλπ.

Στόχος: Η ανάπτυξη και εγκατάσταση μιας εφαρμογής που θα καλύπτει μηχανογραφικά όλες τις διαδικασίες ενός συλλόγου. Δηλαδή την συνολική παρακολούθηση πληροφοριών που σχετίζονται με την διαχείριση και τις συναλλαγές σωματείων, συλλόγων και επαγγελματικών ενώσεων.

ΒΑΣΙΚΕΣ ΛΕΙΤΟΥΡΓΙΕΣ

- Καταγραφή του μητρώου μελών με πλήρη στοιχεία μελών.
- Διαχείριση στοιχείων μελών.
- Δυναμικά φίλτρα αναζήτησης.
- Συνδρομές και οικονομικά στοιχεία.
- Δημογραφικά στοιχεία μέλους.
- Διαχείριση ταμείου.
- Ταμειακές κινήσεις με αναλυτικά αποτελέσματα.
- Εκτυπώσεις Ταμειακής Κατάστασης Εσόδων-Εξόδων.
- Εκτυπώσεις Καρτέλας Μελών, Λίστας Μελών, Κινήσεων Μέλους.

Η **Εφαρμογή Διαχείρισης Συλλόγου** ενσωματώνει εύχρηστες διαδικασίες για την **διαχείριση** των καθημερινών εργασιών **σωματείων**, **συλλόγων** ή **επαγγελματικών ενώσεων**.

Με τη χρήση απλών και λειτουργικών εντολών αναζήτησης, επιτρέπεται η **διαχείριση πλήθους μελών** και η ανίχνευση συγκεκριμένων δεδομένων από την βάση. Οι εντολές αναζήτησης ενσωματώνουν απλά ή σύνθετα ερωτήματα με χρήση πολλαπλών κριτηρίων.

Για την άμεση και ασφαλή διαχείριση των συνδρομών επιτρέπεται η καταγραφή απεριόριστων κατηγοριών με βάση την περίοδο και το ποσό πληρωμής. Η συνδρομή κάθε μέλους μπορεί να υπολογισθεί αυτόματα με βάση τον πίνακα συνδρομών, και το ιστορικό πληρωμών.

Επίσης υποστηρίζει την πλήρη καταγραφή όλων των εξόδων και άλλων ταμειακών κινήσεων.

Αποτέλεσμα: Με την εφαρμογή ο σύλλογος, έχει ένα σύστημα διαχείρισης και πληροφόρησης για όλες τις εργασίες του, τα στοιχεία που κρατούνται χειρόγραφα θα μειωθούν στο ελάχιστο και οι διαδικασίες που απαιτούν χρόνο και ιδιαίτερη προσοχή θα αυτοματοποιηθούν και θα γίνονται άμεσα και χωρίς λάθη.

2. Τι είναι η Java

Η **Java** είναι μία αντικειμενοστραφής γλώσσα προγραμματισμού που εμπεριέχει όλα τα χαρακτηριστικά της εξέλιξης της επιστήμης των υπολογιστών και χρησιμοποιείται σε ένα ευρύ φάσμα εργασιών. Επίσης επιτρέπει σε οποιοδήποτε υπολογιστή, όπου κι αν βρίσκεται αυτός, να έχει πρόσβαση και να χρησιμοποιεί μια εφαρμογή εγκατεστημένη σε κάποιο δίκτυο. Αν την συγκρίνουμε από πλευράς δομής με τις άλλες γλώσσες προγραμματισμού, η java μοιάζει περισσότερο με την **C.** Ακόμη προκάλεσε ίσως το μεγαλύτερο ενδιαφέρον σε σύγκριση με οποιαδήποτε άλλη εξέλιξη στον κόσμο του Internet.

Είναι η πρώτη που κατάφερε να συμπεριλάβει ήχο και κίνηση σε μια ιστοσελίδα. Η Java επιπλέον επιτρέπει στους χρήστες να αλληλεπιδρούν (interact) με την ιστοσελίδα. Εκτός από το να διαβάζει απλά και ίσως να συμπληρώνει μία φόρμα, ο χρήστης μπορεί τώρα να παίξει παιχνίδια, να συνομιλήσει, να λαμβάνει συνεχώς τις πιο πρόσφατες πληροφορίες και πολλά άλλα.

Ακολουθούν μερικές από τις πολλές δυνατότητες της Java:

- Ήχος ο οποίος εκτελείται όποτε ο χρήστης φορτώνει μία σελίδα
- Μουσική που παίζει στο background μιας σελίδας
- Δημιουργία κινουμένων σχεδίων
- Βίντεο
- Παιχνίδια με πολυμέσα

Η Java δεν είναι απλά μια γλώσσα προγραμματισμού του δικτύου με ειδικά χαρακτηριστικά. Παρόλο που η HotJava ήταν η πρώτη γλώσσα που συμπεριέλαβε ήχο και κίνηση, ο Microsoft Internet Explorer και ο Netscape Navigator υποστηρίζουν αυτά τα χαρακτηριστικά με πολλούς και διαφορετικούς τρόπους.Τί κάνει τη Java να ξεχωρίζει;

Η Java είναι μια γλώσσα προγραμματισμού για ποικίλες εφαρμογές. Δεν προσφέρει απλά τη δυνατότητα να προσθέσει ο χρήστης νέο περιεχόμενο στις σελίδες του (όπως συμβαίνει στο Netscape και στον Internet Explorer) αλλά επιτρέπει να προσθέσουμε και τον κώδικα που είναι απαραίτητος. Δεν χρειάζεται πλέον να περιμένουμε για να κυκλοφορήσει ο browser που θα υποστηρίξει τον συγκεκριμένο τύπο εικόνας ή το ειδικό πρωτόκολλο παιχνιδιού (special game protocol). Με τη Java εμείς στέλνουμε στους browsers το περιεχόμενο που χρειάζεται και το πρόγραμμα για να δούμε αυτό το περιεχόμενο την ίδια στιγμή.

Μέχρι τώρα έπρεπε να περιμένουμε τους αναγνώστες μας να ενημερώσουν τους browsers τους προτού χρησιμοποιήσουμε ένα νέο τύπο περιεχομένου (content type). Η ανταγωνιστικότητα της Java βρίσκεται στο ότι μπορεί να εφαρμοστεί σε οποιονδήποτε browser. Για παράδειγμα, θέλουμε να χρησιμοποιήσουμε τα αρχεία EPS στο site μας. Προηγουμένως, έπρεπε να περιμένουμε μέχρι ένας τουλάχιστον web browser να εφάρμοζε την υποστήριξη EPS. Τώρα πια δεν περιμένουμε. Αντίθετα, μπορούμε να γράψουμε τον δικό μας κώδικα για να δούμε τα αρχεία EPS και να το στείλουμε σε οποιονδήποτε πελάτη ζητά τη σελίδα μας τον ίδιο χρόνο που ζητά το αρχείο EPS.

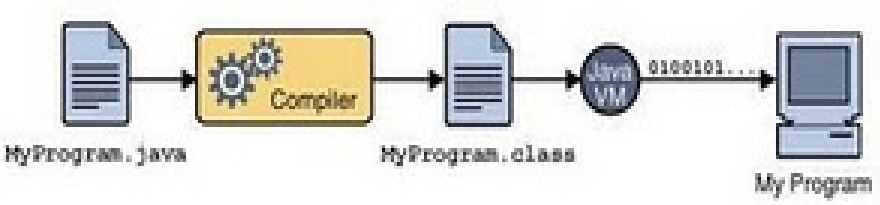
Υποθέτουμε ότι θέλουμε άτομα που να μπορούν να ψάχνουν τον ηλεκτρονικό μας κατάλογο (electronic card catalog). Η βάση δεδομένων του καταλόγου όμως υπάρχει σ' ένα μεγάλο σύστημα που δεν αναγνωρίζει την HTTP. Πριν τη Java θα μπορούσαμε να ελπίζουμε ότι κάποιος browser θα εφάρμοζε το πρωτόκολλο της κάρτας ή θα μπορούσαμε να προσπαθήσουμε να προγραμματίσουμε κάποιο ενδιάμεσο cgi-bin σε ένα UNIX BOX που θα αναγνώριζε HTTP, πράγμα που δεν είναι καθόλου εύκολο. Με τη Java, όταν ένας πελάτης θέλει να μιλήσει στον κατάλογό μας μπορούμε να του στείλουμε τον κώδικα που χρειάζεται. Η Java δεν είναι γλώσσα μόνο για τα web sites. Η Java είναι μια γλώσσα προγραμματισμού που μας επιτρέπει να κάνουμε ό,τι και οι παραδοσιακές γλώσσες, όπως η Fortran και η C++.

Ένα από τα βασικά πλεονεκτήματα της Java έναντι των περισσότερων άλλων γλωσσών είναι η ανεξαρτησία του λειτουργικού συστήματος και πλατφόρμας. Τα προγράμματα που είναι γραμμένα σε Java τρέχουνε ακριβώς το ίδιο σε Windows, Linux, Unix και Macintosh χωρίς να χρειαστεί να ξαναγίνει μεταγλώττιση (compiling) ή να αλλάξει ο πηγαίος κώδικας για κάθε διαφορετικό

λειτουργικό σύστημα. Για να επιτευχθεί όμως αυτό χρειαζόταν κάποιος τρόπος έτσι ώστε τα προγράμματα γραμμένα σε Java να μπορούν να είναι «κατανοητά» από κάθε υπολογιστή ανεξάρτητα του είδους επεξεργαστή (Intel x86, IBM, Sun SPARC, Motorola) αλλά και λειτουργικού συστήματος (Windows, Unix, Linux, BSD, MacOS). Ο λόγος είναι ότι κάθε κεντρική μονάδα επεξεργασίας κατανοεί διαφορετικό κώδικα μηχανής.

Ο compiler της java, μετατρέπει τα αρχεία μας σε bytecodes τα οποία είναι η γλώσσα μηχανής της java (προσοχή εδώ! Σε γλώσσα μηχανής της java και όχι του υπολογιστή), και δημιουργεί ένα νέο αρχείο με την κατάληξη .class. Εδώ βρίσκεται το όλο μυστικό στην συγκεκριμένη γλώσσα. Η java δημιουργεί ένα ενδιάμεσο επίπεδο μεταξύ του προγράμματος και του λειτουργικού συστήματος του υπολογιστή που το ονομάζει JVM (Java Virtual Machine). Αυτό, κατά κάποιο τρόπο ξεγελάει το πρόγραμμα μας για το ποιος είναι το CPU, η μνήμη και το λειτουργικό σύστημα. Οπότε εμείς πάντα γράφουμε κώδικα για το JVM (χωρίς να το καταλάβουμε) και όχι για την συγκεκριμένη πλατφόρμα στην οποία εργαζόμαστε.

Η παρακάτω φωτογραφία προήλθε από το site της Sun (http://java.sun.com/docs/books/tutorial/getStarted/intro/definition.html)



Εικόνα 2.α : Επισκόπηση της διαδικασίας ανάπτυξης λογισμικού.

Ας επαναλάβουμε την διαδικασία συνοπτικά.

- Κατεβάζουμε το JDK από το site της SUN και το εγκαθιστούμε στον υπολογιστή μας. Αυτό περιέχει τα εργαλεία εντολών που μας δίνουν την δυνατότητα να κάνουμε compile (έλεγχος σύνταξης εντολών) και να τρέξουμε το πρόγραμμα μας για να δοκιμάσουμε το αποτέλεσμα του κώδικά μας.

- Μετά γράφουμε το πρόγραμμα μας και το αποθηκεύουμε με κατάληξη .java. Μέχρι αυτή την στιγμή δεν γνωρίζουμε εάν υπάρχουν συντακτικά λάθη.

- Ενεργοποιούμε την διαδικασία του compilation για να εγκριθεί από το περιβάλλον της java το πρόγραμμά μας. Αν υπάρχουν λάθη τότε πρέπει να επιστρέψουμε στο αρχείο και να κάνουμε τις απαραίτητες αλλαγές πριν συνεχίσουμε. Εάν όχι, τότε δημιουργείται ένα καινούργιο αρχείο με το ίδιο όνομα όπως το πρόγραμμα μας αλλά με κατάληξη .class.

- Στον ίδιο φάκελο που έχουμε το αρχείο με την κατάληξη .java τώρα υπάρχει και ένα δεύτερο με την κατάληξη .class.

- Το δεύτερο αρχείο με την κατάληξη .class περιέχει τον κώδικα μας αλλά κωδικοποιημένο σε bytecodes. Αυτός είναι ο κώδικας που καταλαβαίνει να διαβάζει το JVM.

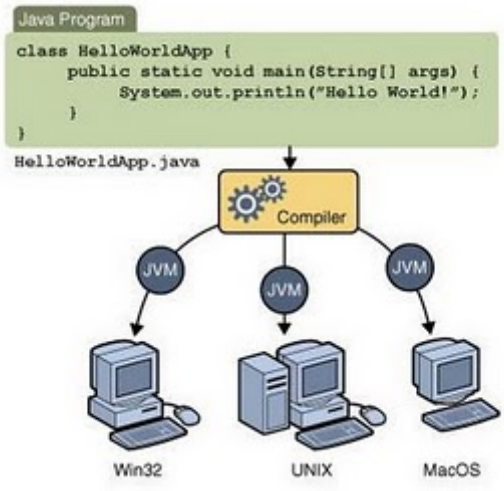
- Πρακτικά το JVM είναι το JRE πρόγραμμα που περιέχεται μέσα στο JDK πακέτο της Java.

- Τα bytecodes διαβάζονται από το JVM και μεταφράζονται στο τρέχων λειτουργικό σύστημα. Δηλαδή αναλαμβάνει την μετάφραση από εντολές java σε κώδικα μηχανής και ολοκληρώνει την εκτέλεση του προγράμματος.

Όπως ήδη αναφέρθηκε πιο πάνω, ο πρακτικός αντιπρόσωπος του JVM είναι το πρόγραμμα JRE (Java Runtime Environment) το οποίο χρειάζεται κάθε υπολογιστής για να τρέξει java προγράμματα. Το JRE περιέχεται μέσα στον compiler ή καλύτερα στο JDK πακέτο το οποίο έχουμε κατεβάσει έτσι ώστε να έχουμε την δυνατότητα και να γράψουμε αλλά και να τρέξουμε τα προγράμματα μας. Μπορούμε όμως να εγκαταστήσουμε το JRE και αυτόνομα και μάλιστα αυτή η διαδικασία είναι αναγκαία όταν είμαστε έτοιμοι να εγκαταστήσουμε την εφαρμογή μας και σε άλλους υπολογιστές. Κάθε υπολογιστής που θα τρέξει την εφαρμογή μας, θα πρέπει να έχει ήδη εγκαταστημένο το JRE πρόγραμμα για να μπορεί να δημιουργήσει το JVM. Πολλοί προγραμματιστές απλά ενσωματώνουν με την εφαρμογή τους και το JRE έτσι ώστε ο τελικός χρήστης να μην αντιμετωπίσει κανένα απολύτως πρόβλημα.

Επειδή θέλουμε το αρχείο .class να τρέχει σε οποιαδήποτε πλατφόρμα, τότε θα πρέπει να κατεβάσουμε και το ανάλογο JRE. Ας υποθέσουμε ότι έχουμε γράψει ένα πρόγραμμα σε java στον υπολογιστή μας που έχει εγκατεστημένα τα Windows σαν λειτουργικό. Ας υποθέσουμε επίσης ότι έχουμε γράψει το πρόγραμμα με τις σωστές εντολές και μετά την διαδικασία του compilation δημιουργήθηκε το αρχείο με κατάληξη .class. Για να μπορεί ένας άλλος χρήστης να τρέξει το πρόγραμμα μας στον δικό του υπολογιστή, θα πρέπει να του δώσουμε μόνο το αρχείο με την κατάληξη .class. Για να έχει την δυνατότητα να το εκτελέσει απλά θα πρέπει να κατεβάσει το σωστό JRE για την πλατφόρμα του υπολογιστή του.

Με άλλα λόγια, αν έχει LINUX τότε χρειάζεται το JRE για το LINUX ενώ εάν έχει Windows τότε χρειάζεται το Windows version του JRE. Ο λόγος που χρειάζεται μόνο το JRE και όχι όλο το πακέτο της java (δηλαδή το JDK) είναι γιατί απλά θα τρέξει το πρόγραμμα, και δεν χρειάζεται να το κάνει compile. Το .class αρχείο είναι ήδη το compile αρχείο του προγράμματος. Η παρακάτω εικόνα προήλθε από το site της SUN από την ηλεκτρονική διεύθυνση http://java.sun.com/docs/books/tutorial/getStarted/intro/definition.html.

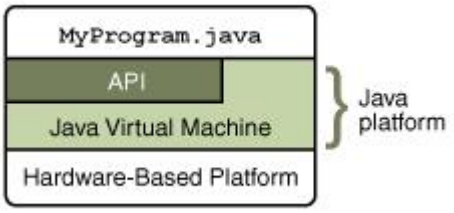


Εικόνα 2.β : Μέσω της Java VM, η ίδια αίτηση μπορεί να λειτουργεί σε πολλαπλές πλατφόρμες.

Ας ορίσουμε και την έννοια "java platform". Πλατφόρμα στον κόσμο της πληροφορικής ονομάζεται το περιβάλλον (μπορεί να είναι software ή hardware) πάνω στο οποίο "τρέχει" ένα πρόγραμμα. Ολοκληρωμένες πλατφόρμες στις οποίες στηρίζονται πολλές εφαρμογές είναι τα λειτουργικά συστήματα όπως Windows, Linux, Solaris και MAC OS. Για την java όμως η έννοια της πλατφόρμας αποκτά συγκεκριμένη έννοια. Ορίζουμε σαν πλατφόρμα την δημιουργία μιας software μόνο υποδομής η οποία στηρίζεται και εκτελείται επάνω σε hardware πλατφόρμες. Η java μάλιστα έχει χωρίσει την πλατφόρμα της σε δύο κατηγορίες:

1. Στην Java Virtual Machine για την οποία ήδη έχουμε μιλήσει και
2. Στο Java Application Programming Interface (API).

Το API είναι μια συλλογή από έτοιμα προγραμματισμένα στοιχεία που μας δίνουν την δυνατότητα να τα χρησιμοποιήσουμε και να αξιοποιήσουμε τις ικανότητες τους χωρίς να γνωρίζουμε την υποδομή τους. Τα στοιχεία αυτά είναι ομαδοποιημένα σε βιβλιοθήκες κατηγοριών για να γίνεται ευκολότερη η ταξινόμηση τους που είναι γνωστές σαν packages. Η παρακάτω εικόνα προέρχεται από το site της Sun (http://java.sun.com/docs/books/tutorial/getStarted/intro/definition.html)



Εικόνα 2.γ : Το API και Java Virtual Machine μονώνουν το πρόγραμμα από το υποκείμενο υλικό.

Είναι αντικειμενοστραφής δηλαδή χρησιμοποιεί τις κλάσεις (classes) για να οργανώσει τον κώδικα σε λογικές ενότητες. Όταν εκτελείται το πρόγραμμα δημιουργεί από τις κλάσεις αντικείμενα. Τα αντικείμενα αυτά έχουν δύο συνιστώσες: τα πεδία και τις μεθόδους. Τα πεδία περιγράφουν τι είναι το αντικείμενο ενώ οι μέθοδοι περιγράφουν τι κάνει το αντικείμενο. Οι κλάσεις μπορούν να κληρονομήσουν ιδιότητες από άλλες κλάσεις. Όμως δεν επιτρέπεται η πολλαπλή κληρονομικότητα (multiple inheritance), όπου μία κλάση έχει τη δυνατότητα να κληρονομήσει πεδία και μεθόδους από περισσότερες από μία άλλες κλάσεις.

Επίσης είναι **ασφαλής**. Έχει σχεδιαστεί από την αρχή με τέτοιο τρόπο, ώστε να παρέχει ασφάλεια εκτέλεσης του κώδικα σε δίκτυο.Αυτό είχε ως αποτέλεσμα τον περιορισμό αρκετών από τα χαρακτηριστικά που έχουν η **C** και η **C++.** Έτσι δεν υπάρχουν δείκτες, ούτε μπορεί να γίνει αυθαίρετη προσπέλαση διευθύνσεων της μνήμης.Οπότε παρέχεται προστασία από την δράση των ιών με κάποια ειδικά προγράμματα που δημιουργεί ο προγραμματιστής και λέγονται **applets.**

Τα applet είναι προγράμματα, τα οποία "κατεβάζει" από τον παγκόσμιο ιστό (WWW) και εκτελεί το πρόγραμμα πλοήγησης (browser) ενός χρήστη ο οποίος είναι συνδεδεμένος στο Internet. Αυτά δεν μπορούν να γράψουν στο σκληρό δίσκο αν δε ζητήσουν πρώτα την άδεια, ούτε και να μπλοκάρουν τον υπολογιστή μας. Χρησιμοποιεί επίσης έναν ισχυρό μηχανισμό για τον έλεγχο αναμενόμενων και μη αναμενόμενων σφαλμάτων (exception handling).

Υποστηρίζει **πολυνημάτωση** (multithreading). Ένα πρόγραμμα **Java** μπορεί να περιλαμβάνει πολλές ξεχωριστές διαδικασίες, οι οποίες να εκτελούνται συνεχώς ανεξάρτητα η μία από την άλλη. Μπορεί π.χ. από το πρόγραμμα να μεταδίδεται μία εικόνα, και συγχρόνως ο χρήστης να εισαγάγει στοιχεία από το πληκτρολόγιο.

Κάνει **συλλογή αχρήστων (garbage collection)**. Σύλλεξη σκουπιδιών είναι μία κοινή ονομασία που χρησιμοποιείται στον τομέα της πληροφορικής για να δηλώσει την ελευθέρωση τμημάτων μνήμης από δεδομένα που δε χρειάζονται και δε χρησιμοποιούνται άλλο. Αυτή η απελευθέρωση μνήμης στη Java είναι αυτόματη και γίνεται μέσω του συλλέκτη απορριμμάτων. Υπεύθυνη για αυτό είναι και πάλι η εικονική μηχανή η οποία μόλις «καταλάβει» ότι η στοίβα (heap) της μνήμης (στη Java η συντριπτική πλειοψηφία των αντικειμένων αποθηκεύονται στη στοίβα σε αντίθεση με τη C++ που αποθηκεύονται κυρίως στο σωρό - stack) κοντεύει να γεμίσει ενεργοποιεί το συλλέκτη απορριμμάτων.

Έτσι ο προγραμματιστής δε χρειάζεται να ανησυχεί για το πότε και αν θα ελευθερώσει ένα συγκεκριμένο τμήμα της μνήμης, ούτε και για δείκτες (pointers) που εστιάζουνε σε άδειο κουτάκι μνήμης. Αυτό είναι ιδιαίτερα σημαντικό αν σκεφτούμε ότι ένα μεγάλο ποσοστό κατάρρευσης των προγραμμάτων οφείλονται σε λάθος χειρισμό της μνήμης.

Μπορεί να δημιουργήσει **γρήγορο κώδικα** με κάποιους ειδικούς μεταγλωττιστές οι οποίοι μετατρέπουν τον κώδικα byte σε κώδικα γλώσσας μηχανής, ο οποίος ανταγωνίζεται σε ταχύτητα τον κώδικα της **C++** ή άλλων γλωσσών.Οι μεταγλωττιστές αυτοί ονομάζονται "just in time compilers" (JIT).

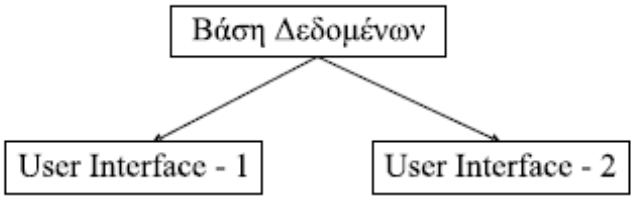
Παρέχει **βιβλιοθήκες κώδικα** για διάφορες χρήσεις, όπως τη δημιουργία γραφικών, μαθηματικές πράξεις, χειρισμό σχεσιακών βάσεων δεδομένων ,κλήση απομακρυσμένων αντικειμένων κτλ.

Με την Java μπορούμε να **δημιουργήσουμε δυναμικές ιστοσελίδες (dynamic web pages).**

Επίσης η Java είναι μια γενική γλώσσα προγραμματισμού. Θα μπορούσε να χρησιμοποιηθεί ως γλώσσα γενικού σκοπού για τη δημιουργία συνηθισμένου τύπου εφαρμογών, όπως για παράδειγμα για την κατασκευή ενός οδηγού (driver) ή για έναν εκτυπωτή.Στην ορολογία της sun ένα κανονικό πρόγραμμα σε Java χαρακτηρίζεται με τον όρο "application" (εφαρμογή) .

2.1 Εισαγωγή στον αντικειμενοστραφή προγραμματισμό

Ο αντικειμενοστραφής προγραμματισμός (OOP – Object Oriented Programming), είναι μια πραγματική φιλοσοφία όπως και ο προστακτικός ή ο λογικός προγραμματισμός. Σύμφωνα με τον OOP ένα πρόγραμμα δεν αποτελείται από τα δεδομένα και τον κώδικα που τα επεξεργάζεται αλλά από αντικείμενα (objects) τα οποία εμπεριέχουν τα δεδομένα και τα οποία (αντικείμενα) ανταλλάσσουν μεταξύ τους πληροφορίες και μηνύματα προκειμένου να επιτευχθεί ο στόχος του προγράμματος. Για παράδειγμα έστω ότι έχουμε ένα σύστημα που περιλαμβάνει μια βάση δεδομένων καθώς και user interfaces με τα οποία οι χρήστες επικοινωνούν και αλληλεπιδρούν με την Β.Δ.(Βάση Δεδομένων).



Τα UIs (User Interfaces) στέλνουν τις εντολές των χρηστών στην ΒΔ και παρουσιάζουν στην οθόνη τις απαντήσεις που λαμβάνουν. Δηλαδή όλο το πρόγραμμα αποτελείται από ένα αντικείμενο Β.Δ. και 2 αντικείμενα UI τα οποία είναι ολόιδια (αλλά εξυπηρετούν άλλους χρήστες).Συνεπώς ο κώδικας που θα γράφαμε θα περιείχε τον ορισμό ενός αντικειμένου ΒΔ, τον ορισμό ενός αντικειμένου UI και την κατασκευή μιας ΒΔ και 2 UI αντικειμένων.

Παρατήρηση

Ο κώδικας που ορίζει ένα αντικείμενο λέγεται **κλάση** (class) του αντικείμενου αυτού. Η κλάση χρησιμοποιείται σαν μήτρα για την κατασκευή πανομοιότυπων αντιγράφων – αντικειμένων. Τα αντικείμενα – αντίγραφα, λέγονται στιγμιότυπα (instances) της κλάσης αυτής και η κατασκευή και αρχικοποίηση ενός από αυτά λέγεται instantiation.

Σημείωση

Το αντικείμενο αυτό είναι μια οντότητα στη μνήμη η οποία περιέχει δεδομένα καθώς και μεθόδους μέσω των οποίων μπορούμε να αλλάξουμε τα δεδομένα ή να επικοινωνήσουμε με το

αντικείμενο. Αντίθετα η κλάση είναι απλώς ένα πρότυπο για την δημιουργία αντιγράφων του ίδιου αντικείμενου. Είναι δηλαδή ο Τύπος Δεδομένων για τις μεταβλητές.

2.2 Δομή ενός προγράμματος σε Java

<ΕΙΣΑΓΩΓΗ ΕΤΟΙΜΩΝ ΚΛΑΣΕΩΝ ΑΠΟ ΤΗ ΒΙΒΛΙΟΘΗΚΗ>
<ΟΡΙΣΜΟΣ ΝΕΑΣ ΚΛΑΣΗΣ>
<ΟΡΙΣΜΟΣ ΝΕΑΣ ΚΛΑΣΗΣ>
κ.οκ.

Μία από όλες τις κλάσεις θα παίξει το ρόλο της αφετηρίας του προγράμματος, δηλαδή θα είναι σαν την main() της C. Αναλυτικότερα, θα αναλάβει να κατασκευάσει τα αντικείμενα που χρειάζεται το πρόγραμμα, με τη χρήση των κλάσεων που ορίσαμε ή φέραμε από την βιβλιοθήκη. Βέβαια την αρχική κλάση θα την υποδείξουμε εμείς όταν θα ξεκινήσουμε τον interpreter της γλώσσας.

2.2.1 Τα αντικείμενα της Java

Όπως αναφέρθηκε και παραπάνω τα αντικείμενα περιέχουν δεδομένα και παρέχουν μεθόδους για την επεξεργασία των δεδομένων αυτών καθώς και για την επικοινωνία με άλλα αντικείμενα.

Τα δεδομένα αυτά μπορεί να είναι άλλα αντικείμενα που περιέχονται μέσα σε ένα άλλο αντικείμενο ή μπορεί να είναι κοινές μεταβλητές όπως τις ξέρουμε από την C και την PASCAL. Τις μεταβλητές και τα αντικείμενα αυτά τα ονομάζουμε πεδία ή instance variables του αντικειμένου. Σε αυτό το σημείο να παρατηρήσουμε ότι μία κλάση μοιάζει με ένα structure της C το οποίο μπορεί να περιέχει κοινές μεταβλητές ή μεταβλητές που προέκυψαν από άλλα structures.

Επίσης μία κλάση (ή ένα αντικείμενο) περιέχει και μεθόδους για την επικοινωνία του με τον έξω κόσμο, δηλαδή τα άλλα αντικείμενα. Οι μέθοδοι αυτοί υλοποιούνται σαν συναρτήσεις παρόμοιες με αυτές της C. Όταν λοιπόν κάποιος θέλει να ζητήσει κάτι από ένα αντικείμενο, (ή εναλλακτικά να στείλει ένα μήνυμα – αίτημα), δεν έχει πάρα να καλέσει – εκτελέσει την αντίστοιχη μέθοδο του αντικειμένου. Αυτό γίνεται ως εξής :

<Αντικείμενο>.<μέθοδος>(<Λίστα παραμέτρων >)

Επίσης να σημειωθεί ότι μία μέθοδος μπορεί να καλεί άλλες μεθόδους του ίδιου αντικειμένου ή να επεξεργάζεται τα πεδία του. Φυσικά μπορεί να καλεί και μεθόδους ξένων αντικειμένων εφόσον έχει κάποιον δείκτη σε αυτά.

ΣΗΜΕΙΩΣΗ

Εδώ είδαμε ένα σπουδαίο χαρακτηριστικό του OOP. Αυτό είναι το να μπορεί να κρατά τα δεδομένα του κρυφά από τα άλλα αντικείμενα αλλά και να προσφέρει μεθόδους με τις οποίες μπορούν άλλα αντικείμενα να επικοινωνούν και να αλληλεπιδρούν μαζί του. Επιπλέον ο κώδικας που υλοποιεί αυτές τις μεθόδους είναι άγνωστος σε άλλες κλάσεις ή αντικείμενα. Έτσι έχουμε την δημιουργία ενός interface επικοινωνίας ανεξάρτητου από την υλοποίηση και την εσωτερική δομή του αντικειμένου. Αυτό λέγεται implementation hiding που είναι μία μορφή data abstraction.

Ύστερα από τα παραπάνω μπορούμε να δούμε πώς ορίζουμε ένα αντικείμενο ή με άλλα λόγια πώς ορίζουμε την κλάση του.

2.2.2 Ορισμός κλάσεων στην Java

Μία κλάση της java μοιάζει αρκετά με ένα structure της C. Δηλαδή έχει την ακόλουθη δομή :

Class<class_name>{

```
<data_type or class_name>       <variable or objects>;

<returned type> <method_name> (<parameter list>){
            <method body>
 }
<data_type or class_name>       <variables or objects>;

<returned type> <method _ name> (<parameter list>){
            <method body>
     }


        κ.ο.κ.
}
```

Γενικά

Μπορούμε να δηλώνομε τα πεδία και τις μεθόδους με όποια σειρά θέλουμε. Επίσης μπορούμε να τα χρησιμοποιούμε πριν ακόμη τα δηλώσουμε. Ωστόσο συνιστάται τα πεδία να τοποθετούνται στην αρχή της κλάσης ώστε ο κώδικας να είναι πιο ευανάγνωστος. Ακόμη η τοποθέτηση κατατοπιστικών σχολίων συμβάλλει στην βελτίωση της αναγνωσιμότητας του προγράμματος και διευκολύνει την διαδικασία αποσφαλμάτωσης (debugging).

2.2.3 Τα ποιο σημαντικά χαρακτηριστικά του αντικειμενοστραφούς προγραμματισμού

Encapsulation ( )

Η διαδικασίες κρύβονται από τον χρήστη και τα ίδια τα δεδομένα προσδιορίζουν τους τρόπους διαχείρισης τους.

Polymorphism (Πολυμορφισμός)

Αντικείμενα που ανήκουν σε παρόμοιες κλάσεις μπορούν να έχουν κοινό τρόπο προσπέλασης, με αποτέλεσμα ο χρήστης να μπορεί να τα χειριστεί με τον ίδιο τρόπο χωρίς να χρειάζεται να μάθει νέες διαδικασίες.
Οι τύποι πολυμορφισμού είναι:

- Η υπερκάλυψη (overriding)
- Η υπερφόρτωση (overloading)
- Η δυναμική συσχέτιση μεθόδων (dynamic method binding)

Inheritance (Κληρονομικότητα)

Η κληρονομικότητα είναι ένα ακόμη χαρακτηριστικό του αντικειμενοστραφούς προγραμματισμού. Μπορούμε να δημιουργήσουμε ένα νέο αντικείμενο παίρνοντας ως βάση ένα άλλο ήδη υπάρχον. Το νέο αντικείμενο θα έχει τα χαρακτηριστικά του παλιού ενώ θα μπορεί να τα τροποποιήσει, να τα επεκτείνει και να προσθέσει καινούργια για να καλύψει συγκεκριμένες ανάγκες. Πρακτικά, μια κλάση κληρονομεί τα χαρακτηριστικά μιας υπάρχουσας κλάσης και προσθέτει καινούργια ή τροποποιεί τα ήδη υπάρχοντα.

2.3 Κατασκευαστές (constructors)

Κάθε κλάση διαθέτει τουλάχιστον μία μέθοδο η οποία εκτελείται κατά τη δημιουργία ενός στιγμιότυπου της, (δηλ. κατά τη δημιουργία ενός object της κλάσης αυτής).Αυτές οι μέθοδοι λέγονται **κατασκευαστές** (constructors) της κλάσης. Σκοπός των constructors είναι η δέσμευση μνήμης για την

κατασκευή του αντικειμένου καθώς και η διενέργεια κατάλληλων αρχικοποιήσεων. Μπορούν να οριστούν πολλοί constructors για μία κλάση αλλά για τη δημιουργία ενός αντικειμένου (στιγμιότυπου) μπορεί να καλέσει μόνο έναν από αυτούς. Εάν δεν οριστεί κανένας constructor τότε η γλώσσα καλεί έναν constructor της υπερκλάσης (κάποιον που δεν θέλει παραμέτρους) ή δίνει μήνυμα λάθους αν δεν βρει κάποιον τέτοιο.

2.4 Καταστροφή αντικειμένων (Finalization)

Σε ορισμένες γλώσσες όπως η C++ και η Object PASCAL υποστηρίζεται και η χρήση των **καταστροφέων (destructors)** μιας κλάσης. Προφανώς οι destructors είναι μέθοδοι που τερματίζουν την ύπαρξη ενός στιγμιότυπου της αντίστοιχης κλάσης. Ωστόσο στην JAVA το χαρακτηριστικό αυτό δεν υποστηρίζεται αφού είναι αιτία λαθών. Έχει όμως έναν άλλο πιο ασφαλέστερο τρόπο για τον τερματισμό (finalization) ενός αντικειμένου. Ο τρόπος αυτός έχει ως εξής. Το runtime system της java εξετάζει κατά διαστήματα αν για κάθε αντικείμενο υπάρχει τουλάχιστο ένας δείκτης σε αυτό. Αν δεν υπάρχει τότε το αντικείμενο είναι άχρηστο (garbage) και η μνήμη που κατέχει απελευθερώνεται.

Όμως προτού γίνει αυτό καλείται η μέθοδος void finalize() η οποία πρέπει να περιέχει κώδικα για clean-up. Για παράδειγμα αν το αντικείμενο έχει ακόμα ανοιχτές συνδέσεις στο δίκτυο τότε θα τις κλείσει. Μετά το τέλος της finalize() το αντικείμενο καταστρέφεται. Η διαδικασία του εντοπισμού των αντικειμένων – σκουπιδιών λέγεται garbage collection και ο μηχανισμός που το πραγματοποιεί είναι ενσωματωμένος στο JAVA runtime system.

2.5 Προσδιοριστές πρόσβασης (access specifiers)

Συμβαίνει πολλές φορές να θέλουμε να έχουμε κάποια δεδομένα ή μεθόδους μας ορατά στα αντικείμενα όλων των κλάσεων ή να θέλουμε κάποια πεδία να είναι ορατά μόνο στα αντικείμενα της τρέχουσας κλάσης και όχι και στις υποκλάσεις της. Σε όλες αυτές τις περιπτώσεις χρησιμοποιούμε τους access specifiers. Οι access specifiers είναι δεσμευμένες λέξεις της JAVA οι οποίες καθορίζουν το ποιος θα έχει πρόσβαση, δηλαδή το ποιος θα μπορεί να χρησιμοποιεί και να επεξεργάζεται, τις μεθόδους ή/και τα πεδία που έχουν κάποιον τέτοιο προσδιοριστή. Για αρχή ας δούμε ποιοι είναι :

- **public - Σημαίνει** ότι το συγκεκριμένο πεδίο/μέθοδος μπορεί να χρησιμοποιηθεί από οποιονδήποτε, ακόμα και από ξένο αντικείμενο.
- **protected -** Σημαίνει ότι το πεδίο/μέθοδος που έχει είναι ορατό μόνο στις μεθόδους της κλάσης αυτής καθώς και στις υποκλάσεις.
- **Private - Σημαίνει** ότι το πεδίο/μέθοδος είναι ορατό μόνο στις μεθόδους αυτής της κλάσης αλλά όχι στις υποκλάσεις της.
- **Τίποτα -** Είναι public αλλά μόνο για το τρέχων package.

2.6 Τροποποιητές (Modifiers)

Είναι δεσμευμένες λέξεις της JAVA οι οποίες προσδίδουν συγκεκριμένες ιδιότητες ή χαρακτηρίζουν τα πεδία/μεθόδους που τα έχουν. Οι πιο σημαντικοί από τους modifiers είναι οι ακόλουθοι:

- **final -** Για τα πεδία σημαίνει ότι είναι σταθερές και όχι μεταβλητές. Άρα δεν μπορούμε να τους αναθέσουμε νέα τιμή, παρά μόνο στην αρχικοποίηση της σταθεράς όταν αυτή δηλώνεται. Τέλος για τις κλάσεις σημαίνει ότι δεν μπορούμε να κατασκευάσουμε υποκλάσεις από αυτές.

- **Synchronized -** Για μεθόδους σημαίνει ότι το αντικείμενο ιδιοκτήτης της μεθόδου αυτής είναι threadsafe. Δηλαδή αν κάποιος χρησιμοποιεί αυτή την μέθοδο τότε, αποκλείει την ταυτόχρονη χρήση του αντικειμένου και από τρίτον. Ο τρίτος θα περιμένει μέχρι να ελευθερωθεί το αντικείμενο και έπειτα θα το χρησιμοποιήσει και αυτός.

  ΠΡΟΣΟΧΗ: αν κάποιος (τρίτος) θέλει να χρησιμοποιήσει την ίδια μέθοδο ενός άλλου αντικειμένου, (της ίδιας κλάσης), δεν θα έχει κανένα πρόβλημα να το κάνει αρκεί να μην χρησιμοποιείται ήδη και αυτό.
- **abstract -** Για μεθόδους σημαίνει ότι δεν θα υλοποιηθούν εδώ αλλά σε κάποια υποκλάση, έτσι δεν παρέχεται το σώμα της μεθόδου {...} αλλά στη θέση του βάζουμε ένα semicolon (;). Επίσης μία κλάση με έστω μία abstract μέθοδο θα πρέπει να δηλωθεί και αυτή ως abstract.
- **static -** Το πεδίο/μέθοδος που το έχει είναι μοναδικό για όλα τα στιγμιότυπα της κλάσης. Όλα τα αντικείμενα της κλάσης αυτής έχουν ένα τέτοιο πεδίο σαν κοινή μεταβλητή. Επίσης οι static μεταβλητές και μέθοδοι μπορούν να χρησιμοποιηθούν και χωρίς να κατασκευαστεί κάποιο αντικείμενο (στιγμιότυπο) της κλάσης. Όμως οι static μέθοδοι δεν μπορούν να δουν ή να αλλάξουν τα πεδία που δεν είναι static. Ωστόσο το αντίθετο είναι επιτρεπτό.
- **native -** Όσες μέθοδοι δηλωθούν native δεν πρέπει να έχουν σώμα. Ωστόσο ο κώδικας τους θα πρέπει να γράφει σε C και να συνδεθεί με το bytecode κατά την εκτέλεση του προγράμματος.

2.7 Διασυνδέσεις (Interfaces)

Μια διασύνδεση (interface) ορίζει έναν τρόπο συμπεριφοράς που μπορεί να υλοποιηθεί από οποιαδήποτε κλάση. Δηλώνει ένα σύνολο μεθόδων αλλά δεν προσφέρει την υλοποίηση τους. Αυτές οι μέθοδοι θεωρούνται αυτόματα εικονικές και οποιαδήποτε κλάση υλοποιεί το interface πρέπει να δώσει και υλοποίηση για όλες τις μεθόδους του. Ο λόγος για τον οποίο υπάρχουν τα interfaces είναι ότι κάθε κλάση μπορεί να έχει μόνο μία υπερκλάση αλλά μπορεί να υλοποιήσει άπειρο αριθμό interfaces.Ένα interface μοιάζει με μία abstract κλάση αλλά έχει τις εξής σημαντικές διαφορές :

- Ένα interface δεν μπορεί να παρέχει υλοποίηση για καμία μέθοδο του.
- Μια κλάση μπορεί να υλοποιήσει πολλά interfaces αλλά να κληρονομήσει μία μόνο κλάση.
- Ένα interface μπορεί επίσης να περιέχει σταθερές στον ορισμό του ή να κληρονομεί από άλλα interfaces.

2.8 Εξαιρέσεις (Exceptions)

Όταν σε ένα πρόγραμμα σε JAVA συμβεί κάποιο λάθος, για παράδειγμα περαστεί κάποια λάθος παράμετρος, τότε ο κώδικας που θα το ανιχνεύσει μπορεί να εγείρει (throw= πετάω) μία εξαίρεση. Η έγερση εξαίρεσης θα έχει ως αποτέλεσμα τον τερματισμό του thread στο οποίο συνέβη το λάθος και θα τυπωθεί κάποιο μήνυμα. Ωστόσο τα προγράμματα μπορούν να ορίσουν χειριστές εξαιρέσεων (exception handlers) οι οποίοι θα πιάνουν την εξαίρεση και θα φροντίζουν ώστε το πρόγραμμα να ανανήψει το λάθος. Μερικές από τις εξαιρέσεις προκαλούνται από το runtime system, όπως στην περίπτωση διαίρεσης με το μηδέν.

Ωστόσο, οποιαδήποτε κλάση μπορεί να ορίσει και να εγείρει δικές της εξαιρέσεις. Αυτό γίνεται ως εξής. Πρώτα δημιουργεί (με new) ένα αντικείμενο εξαίρεσης το οποίο θα πρέπει να είναι στιγμιότυπο της κλάσης Exception ή κάποιας υποκλάσης της. Έπειτα με την εντολή throw και το αντικείμενο εξαίρεσης προκαλείται exception. Η εκτέλεση του κώδικα θα διακοπεί στην εντολή throw

και ο υπόλοιπος κώδικας που ακολουθεί δεν θα εκτελεστεί. Επίσης η μέθοδος μέσα στην οποία συνέβη η εξαίρεση δεν θα επιστρέψει κάποια τιμή. Το αντικείμενο τώρα, της εξαίρεσης θα δοθεί στον κατάλληλο exception handler, απ' όπου και συνεχίζεται η εκτέλεση του προγράμματος.

Για να ορίσουμε ένα χειριστή εξαιρέσεων (exception handler) θα πρέπει να κλείσουμε τον κώδικα που μπορεί να προκαλέσει την εξαίρεση μέσα σε μια εντολή try.Μετά την try θα πρέπει να βάλουμε μία ή περισσότερες εντολές catch.Κάθε catch μπορεί να πιάνει, μία μόνο κλάση εξαίρεσης. Επίσης σε κάθε catch θα υπάρχει ο κατάλληλος κώδικας για τον χειρισμό της εξαίρεσης.

2.9 Τα βασικά των Applets στην Java

Στην Java τα Applets εκτελούνται μέσα από κάποιον Java WWW Browser.Η αναφορά σε ένα Applet γίνεται σε μια WEB σελίδα μέσω ενός ειδικού HTML tag.Όταν ο χρήστης σηκώσει σε κάποιον Browser μια WEB σελίδα που περιέχει κάποιο Applet, ο Browser κατεβάζει το Applet από τον Web Server και το εκτελεί στον τοπικό υπολογιστή.

Επειδή τα Java Applets τρέχουν μέσα από κάποιον Java Browser, έχουν το πλεονέκτημα της δομής που παρέχει ο Browser: ένα υπάρχον παράθυρο, έννοιες γραφικών και γεγονότων, και το interface που τα περιβάλει. Επιπλέον επειδή τα Applets μπορούν να κατεβούν από οπουδήποτε και να εκτελούνται τοπικά στον υπολογιστή του χρήστη, υπάρχουν περιορισμοί που εμποδίζουν τα Applets να προκαλέσουν ζημία στο τοπικό σύστημα όπως:

- Τα Applets δεν μπορούν να γράψουν ή να διαβάσουν στο τοπικό σύστημα αρχείων, εκτός από καταλόγους που πρέπει να έχει προκαθορίσει ο τοπικός χρήστης.
- Τα Applets μπορούν να επικοινωνήσουν μόνο με τον Server στον οποίο το Applet είχε αποθηκευτεί.
- Τα Applets δεν μπορούν να τρέξουν προγράμματα που υπάρχουν στο σύστημα του τοπικού χρήστη.

Επίσης για να δημιουργηθεί ένα Applet πρόγραμμα, πρέπει να δημιουργηθεί μια υποκλάση της κλάσης Applet, του java.applet πακέτου:

```
Public class my Class extends java.applet.Applet {
……..
}
```

Η κλάση Applet παρέχει συμπεριφορά που επιτρέπει στο Applet πρόγραμμα όχι μόνο να λειτουργεί μέσα στον Browser αλλά να έχει και δυνατότητες AWT για ενσωμάτωση User Interface στοιχείων, διαχείριση γεγονότων ποντικιού και πληκτρολογίου, καθώς και ζωγραφικής στην οθόνη. Παρόλο που ένα Applet πρόγραμμα μπορεί να αποτελείται από επιπλέον βοηθητικές κλάσεις, η υποκλάση της κλάσης Applet είναι αυτή που ενεργοποιεί την εκτέλεση του Applet προγράμματος.

2.9.1 Βασικές λειτουργίες των Applets

Για την δημιουργία Java εφαρμογών, η κλάση της εφαρμογής πρέπει να διαθέτει την μέθοδο main(). Όταν η εφαρμογή αρχίζει να τρέχει, εκτελείται η μέθοδος main() η οποία καθορίζει την συμπεριφορά του προγράμματος. Αντιθέτως στα Applets προγράμματα υπάρχουν διαφορετικές λειτουργίες που αντιστοιχούν σε γεγονότα που συμβαίνουν κατά την διάρκεια της ζωής του Applet (π.χ. γεγονότα αρχικοποίησης, ζωγραφικής, ποντικιού κλπ.).

Σε κάθε λειτουργία αντιστοιχεί και κάποια μέθοδος η οποία καλείται από τον Browser όταν ένα γεγονός συμβεί. Η μέθοδοι των λειτουργιών έτσι όπως ορίζονται στην Applet κλάση της Java δεν κάνουν τίποτε. Για να δώσουμε κάποια συμπεριφορά σε κάποιο γεγονός της Applet εφαρμογή μας

πρέπει να ξανα ορίσουμε την μέθοδο που αντιστοιχεί στο γεγονός μέσα στην υποκλάση της Applet που δημιουργήσαμε. Οι πέντε πιο βασικές μέθοδοι μιας applet είναι**:**

1.
Public void init () {
……
}

Η μέθοδος αυτή εκτελείται όταν η applet εφαρμογή κατεβαίνει στο τοπικό υπολογιστή για εκτέλεση. Η μέθοδος αυτή μπορεί να περιλαμβάνει την δημιουργία κάποιων αντικειμένων, τον καθορισμό παραμέτρων, το φόρτωμα εικόνων ή font κλπ.

2.
Public void start () {
……
}

Η μέθοδος αυτή καλείται αμέσως μετά την init(). Η start() καλείται επίσης όταν η applet εφαρμογή είχε προηγουμένως σταματήσει την εκτέλεση της. Για παράδειγμα μια applet εφαρμογή σταματά την εκτέλεση της όταν ο χρήστης αλλάξει μέσω ενός συνδέσμου HTML σελίδα και ξαναρχίζει την εκτέλεση της όταν ο χρήστης γυρίσει πίσω στη σελίδα της Applet εφαρμογής.

3.
Public void stop () {
……
}

Η stop() σταματά την εκτέλεση της Applet εφαρμογής και είναι το συμπλήρωμα της start(). Ο χρήστης μπορεί επίσης να καλέσει από μόνος του την stop για να σταματήσει την Applet εφαρμογή.

4.
Public void destroy () {
……
}

Δίνει την δυνατότητα στη Applet εφαρμογή να ελευθερώσει τους πόρους του συστήματος που της είχαν διατεθεί. Η μέθοδος εκτελείται λίγο πριν η Applet εφαρμογή πάψει οριστικά την εκτέλεση της ή όταν ο Browser κλείσει. Συνήθως η μέθοδος αυτή δεν χρειάζεται να οριστεί στην Applet εφαρμογή εκτός από πολύ ειδικές περιπτώσεις.

5.
Public void paint (Graphics g) {
…….
}

Με την μέθοδο αυτή η Applet εφαρμογή εμφανίζει κάτι στην οθόνη πχ. κείμενα, γραφικά, εικόνες. Η μέθοδος αυτή καλείται σε διάφορες περιπτώσεις όπως όταν η Applet εφαρμογή αρχικοποιείται, όταν μετακινείται ο Browser ή τοποθετείται πίσω από άλλο παράθυρο και μετά έρχεται πάλι μπροστά, όταν θέλουμε να δημιουργήσουμε animation οπότε και καλείται συνεχώς. Για να χρησιμοποιήσουμε την μέθοδο αυτή πρέπει προηγουμένως η κλάση των γραφικών να περιληφθεί στον κώδικα της Applet εφαρμογής.

2.10 Abstract Windowing Toolkit (AWT)

Οι κλάσεις του API της Java που χρησιμοποιούνται για την ανάπτυξη γραφικών διαπροσωπειών ανθρώπου-μηχανής αποτελούν το AWT (Abstract Windowing Toolkit). Όπως

υποδηλώνει το όνομα του είναι ένα αφηρημένο (δηλαδή ανεξάρτητο υλοποίησης) σύνολο εργαλείων για τη δημιουργία μιας απλής Γραφικής Διεπαφής Χρήστη (Graphical User Interface - GUI).Το AWT παρέχει στους προγραμματιστές της Java μια διαπροσωπεία αρκετά υψηλού επιπέδου, αντίστοιχη του MFC στα Microsoft Windons.Χρησιμοποιεί ένα σύνολο συστατικών (components), που του παρέχει η πλατφόρμα στην οποία τρέχει κάθε φορά η εφαρμογή. Μερικά χρήσιμα components όπως θα δούμε παρακάτω είναι τα κουμπιά, τα μενού, κ.λπ. Οι εφαρμογές που βασίζονται στο AWT έχουν την ίδια συμπεριφορά και εμφάνιση με τις υπόλοιπες εφαρμογές της πλατφόρμας στην οποία τρέχουν. Επίσης δεν υπάρχει καμία εξάρτηση της γλώσσας Java από το AWT και μπορούμε αν θέλουμε να υλοποιήσουμε και να χρησιμοποιήσουμε άλλα εργαλεία και βιβλιοθήκες γραφικών.

2.10.1 Βασικά στοιχεία ενός GUI

Ένα Graphical User Interface - GUI (Γραφική Διεπαφή με το Χρήστη) είναι το μέρος του προγράμματος, που φροντίζει για τον τρόπο εμφάνισης και χειρισμού του προγράμματος από τον χρήστη. Ένα GUI αποτελείται από GUI-components. Οι κλάσεις που χρησιμοποιούνται για την κατασκευή αντικειμένων τύπου GUI-components ανήκουν στο java.awt (Abstract Windowing Toolkit) package. Οι βασικότερες από αυτές είναι η κλάση Component και η κλάση Container. Επίσης κάθε κλάση που κληρονομεί την κλάση Container είναι και αυτή ένα Container. Οι κλάσεις που συνηθέστερα χρησιμοποιούνται για την κατασκευή GUI-components περιγράφονται παρακάτω **:**

Label	Εμφανίζει κείμενο που δεν μπορεί να τροποποιηθεί από τον χρήστη.
Button	Περιοχή που προξενεί ένα γεγονός (event) όταν επιλέγεται με το ποντίκι.
TextField	Περιοχή όπου ο χρήστης εισάγει δεδομένα από το πληκτρολόγιο. Σε ένα TextField μπορούμε επίσης να εμφανίζουμε πληροφορίες.
TextArea	Περιοχή όπου ο χρήστης εισάγει δεδομένα από το πληκτρολόγιο. Σε ένα TextArea μπορούμε να έχουμε πολλές γραμμές κειμένου σε αντίθεση με το TextField που μπορούμε να έχουμε μόνο μία.
Choice	Μια λίστα στοιχείων από τα οποία ο χρήστης μπορεί να επιλέξει ένα από αυτά με το ποντίκι.
Checkbox	Ένα boolean component που είναι επιλεγμένο ή μή επιλεγμένο. Με αυτό υλοποιούνται και τα Radio buttons (αμοιβαίως αποκλειόμενες επιλογές).
List	Λίστα στοιχείων από τα οποία ο χρήστης μπορεί να επιλέξει ένα από αυτά με το ποντίκι. Διπλό κλικ σε ένα στοιχείο της λίστας προξενεί ένα action event.
Panel	Είναι ένα container αντικείμενο στο οποίο μπορούν να τοποθετηθούν component αντικείμενα.
ScrollPane	Είναι ένα container αντικείμενο στο οποίο μπορεί να τοποθετηθεί ένα component, συνήθως ένα Panel, το οποίο θα εμφανίζεται στον χρήστη αλλά κι όταν αυτό δεν είναι εφικτό θα εμφανίζονται Scrollbars που θα επιτρέψουν την διολίσθηση του component έτσι ώστε να γίνεται ορατό και το τμήμα του που δεν φαίνεται.

Πίνακας 2.10.1.α Περιγραφή των κλάσεων για την κατασκευή GUI-components.

2.10.2 Δομή ενός GUI

Σύνθετα GUIs χωρίζονται σε υποπεριοχές. Για να απλοποιηθεί, λοιπόν, η ανάπτυξη ενός GUI στην Java, χωρίζεται συνήθως σε πολλά Panel components. Αξίζει να αναφερθεί ότι η κλάση Panel κληρονομεί την κλάση Container και η κλάση Applet την κλάση Panel. Έτσι τα Panels και τα Applets είναι Containers και μπορούν να έχουν Components, ακόμα και τύπου Panel. Ο κατασκευαστής της κλάσης Panel δεν παίρνει παραμέτρους. Τελικά προκύπτει μια δενδρική δόμηση του GUI που λέγεται component hierarchy ή containment hierarchy.

Σε κάθε Panel τα Components που περιέχει είναι τοποθετημένα με ένα συγκεκριμένο πλάνο ή σχέδιο (layout). Η Java παρέχει διαχειριστές πλάνων (Layout Managers) για την διευθέτηση των Component αντικειμένων μέσα σε ένα Applet ή Panel. Οι πιο συνηθισμένες κλάσεις των Layout Managers περιγράφονται παρακάτω **:**

FlowLayout	Είναι ο default Layout Manager για τα Applets και τα Panels. Τοποθετεί τα Component αντικείμενα από αριστερά προς τα δεξιά με την σειρά που προστίθονται
BorderLayout	Τοποθετεί τα Component αντικείμενα σε πέντε περιοχές: Βόρεια, Νότια, Ανατολική, Δυτική και Κεντρική (North, South, East, West, Center).
GridLayout	Τοποθετεί τα Component αντικείμενα σε γραμμές και στήλες με καθορισμένη σειρά (πρώτα γεμίζει η πρώτη γραμμή μετά η δεύτερη γραμμή κλπ).
CardLayout	Τοποθετεί τα Component αντικείμενα σε στοίβα. Κάθε Container αντικείμενο στη στοίβα μπορεί να χρησιμοποιήσει οποιονδήποτε Layout Manager. Μόνο το Container αντικείμενο που βρίσκεται στην κορυφή της στοίβας είναι ορατό.
GridBagLayout	Είναι παρόμοιος με τον GridLayout Manager. Διαφέρει από αυτόν στο ότι κάθε Component αντικείμενο που του προστίθεται μπορεί να έχει οποιοδήποτε μέγεθος (δηλαδή να καλύπτει περισσότερες από μια γραμμές και στήλες). Επίσης η σειρά με την οποία προστίθονται τα Component αντικείμενα στον GridBagLayout Manager μπορεί να είναι οποιαδήποτε.

Πίνακας 2.10.2.α : Περιγραφή των κλάσεων των Layout Managers.

2.11 Τα εργαλεία της Java

Ακολούθως παρουσιάζονται εν συντομία όλα τα εργαλεία της Java :

- **javac** Είναι ο compiler της Java. Η χρήση του στο command-line είναι javac<όνομα αρχείου>. Το javac δεν παράγει ένα αρχείο με όλο τον κώδικα, αλλά χωριστό αρχείο για κάθε κλάση. Τα αρχεία των κλάσεων ονομάζονται**:** <όνομα κλάσης>.class.
- **java** Είναι ο interpreter της Java. Η χρήση του είναι η εξής : java <κλάση>, π.χ. java myClass και όχι java myClass.class.
- **jdb** Είναι ο Java debugger.
- **javah** Κατασκευάζει C files και stub files για κάποια κλάση. Αυτά τα αρχεία είναι απαραίτητα όταν θέλουμε να υλοποιήσουμε κάποιες από τις μεθόδους της κλάσης σε C, πράγμα πολύ σπάνιο.
- **javap** Είναι ο Java disassembler.
- **javadoc** Είναι ένα πρόγραμμα για αυτόματη κατασκευή documentation**.** Είναι αρκετά χρήσιμο στην κατασκευή βοηθημάτων και τεχνικών αναφορών για εφαρμογές οποιουδήποτε μεγέθους.
- **applet viewer** Είναι ένα πρόγραμμα το οποίο μας επιτρέπει να τρέχουμε και να χρησιμοποιούμε τα διάφορα applets σε Java.Οι stand-alone εφαρμογές, ωστόσο, δεν τρέχουν σε applet viewer αλλά κατευθείαν στον java ή javaw.

2.12 Χρήσιμες διευθύνσεις Internet για την Java

Παρακάτω αναφέρονται κάποιες διευθύνσεις του internet που παρέχουν πληροφορίες και χρήσιμες οδηγίες για τη γλώσσα προγραμματισμού Java.

Java Tutorial :

http://java.sun.com/docs/books/tutorial/

Περιέχει αναλυτικό εκπαιδευτικό υλικό και ενσωματωμένα παραδείγματα για όλα τα θέματα της γλώσσας. Μπορούμε να αναφερόμαστε σε αυτό όποτε χρειάζεται να αντλήσουμε περισσότερες πληροφορίες σχετικά με κάποιο θέμα.

Java API :

http://java.sun.com/j2se/docs/api/index.html

Πολύ χρήσιμη διεύθυνση που είναι καλό να την έχουμε ανοικτή όταν προγραμματίζουμε σε Java.Περιέχει το **Application Programming Interface της Java (Java API)** δηλαδή όλες της κλάσεις που διαθέτει η Java ομαδοποιημένες ανάλογα με τις λειτουργίες που προσφέρουν σε ενότητες (πακέτα), την περιγραφή της λειτουργίας κάθε κλάσης καθώς και των μεθόδους της.

Java Platform:

http://www.oracle.com/technetwork/java/javase/downloads/index.html

Η Java πλατφόρμα με τα εργαλεία που αναφέρθηκαν παραπάνω είναι το εκτελέσιμο πρόγραμμα που πρέπει να εγκατασταθεί στον υπολογιστή μας ώστε να είμαστε σε θέση να προγραμματίζουμε σε Java.Το συγκεκριμένο εκτελέσιμο μας εγκαθιστά το JRE (Java Runtime Environment) στον Web Browser του υπολογιστή μας εάν αυτό δεν έχει ήδη εγκατασταθεί. Το JRE δίνει τη δυνατότητα στο Web Browser μας να εκτελεί java applets.

2.13 Εγκαθιστώντας τη Java

Εκδόσεις της Java σε διαφορετικά στάδια ολοκλήρωσης διατίθενται από τη Sun για Windows, Unix και MacOS. Μέχρι στιγμής δεν υπάρχουν εκδόσεις της Java για τα MIPS, Alpha or PowerPC based NT, Windows , Amiga. Το βασικό περιβάλλον της Java αποτελείται από έναν web browser, ο οποίος μπορεί να εκτελεί τις μίνι εφαρμογές της Java, έναν compiler που μετατρέπει τον πηγαίο κώδικα της Java σε κώδικα byte, κι έναν μεταφραστή της Java για να εκτελεί τα προγράμματα.

Αυτά είναι τα τρία συστατικά-κλειδιά ενός περιβάλλοντος Java. Επίσης απαραίτητος είναι ένας text editor όπως το Brief ή το BBEdit.Η Sun διαθέτει το Java Developers Kit (JDK). Περιέχει έναν applet viewer όπου θα μπορούμε να δούμε και να ελέγξουμε τις εφαρμογές μας. Το JDK περιλαμβάνει επίσης τον javac compiler, τον java interpreter, τον javaprof profiler, τον Java debugger και περιορισμένα κείμενα. Τα περισσότερα από τα κείμενα για το API και τη βιβλιοθήκη κλάσης είναι στο web site της Sun.

2.13.1 Οδηγίες εγκατάστασης σε Windows

Θα χρειαστούμε περίπου 6 MB ελεύθερα στο δίσκο για την εγκατάσταση του JDK. Εκτελούμε το αρχείο κάνοντας διπλό κλικ πάνω του στο File Manager ή επιλέγοντας Run... από το Program Manager's File menu και πληκτρολογώντας το μονοπάτι στο αρχείο. Προτείνεται να το εγκαταστήσουμε στο C:drive. Σ' αυτήν την περίπτωση τα αρχεία θα βρίσκονται στο C:\java. Θα πρέπει να προσθέσουμε το C:\java\bin directory στο path environment.

Η αρχειοθέτηση περιλαμβάνει δύο κοινά DLL's:

- MSVCRT20.DLL
- MFC30.DLL

Αυτά τα 2 αρχεία θα εγκατασταθούν στο java directory. Αν δεν έχουμε ήδη αντίγραφα αυτών στο σύστημά μας, τα αντιγράφουμε στο C:\java\bin directory. Αν τα έχουμε απλά διαγράφουμε τα επιπλέον αντίγραφα.

2.14 Τι είναι το εργαλείο NetBeans

Το εργαλείο NetBeans παρέχει μια πλατφόρμα για την ανάπτυξη διάφορων εφαρμογών που είτε θα χρησιμοποιηθούν στα πλαίσια κάποιου δικτύου είτε θα χρησιμοποιηθούν ως ανεξάρτητες (Stand-alone) εφαρμογές.

Το εργαλείο NetBeans υποστηρίζει διάφορες γλώσσες προγραμματισμού. (Java, C, C++, PHP, Python, JavaScript).Παρέχει ένα πλήρες περιβάλλον ανάπτυξης εφαρμογών (IDE: Integrated Development Enviroment).Το περιβάλλον ανάπτυξης γενικά παρέχει βοηθήματα στον προγραμματιστή τα οποία κάνουν ευκολότερη και αποδοτικότερη την ανάπτυξη των εφαρμογών. Επίσης τα βασικότερα από τα βοηθήματα αυτά αφορούν κυρίως τον Source Code Editor (συντακτικού στον κώδικα, live επισήμανση και live parsing, pop up παράθυρα βοήθειας).Ακόμη παρέχονται βοηθήματα για κατασκευή GUI άλλα και βοηθήματα διαχείρισης βάσεων δεδομένων.

3. Βάσεις Δεδομένων

Με τον όρο **βάση δεδομένων** εννοείται μία συλλογή από *συστηματικά οργανωμένα* (formatted) σχετιζόμενα δεδομένα. Ένας τηλεφωνικός κατάλογος, για παράδειγμα, θεωρείται βάση δεδομένων, καθώς αποθηκεύει και οργανώνει σχετιζόμενα τμήματα πληροφορίας, όπως είναι το όνομα και ο αριθμός τηλεφώνου. Ωστόσο, στον κόσμο των υπολογιστών, με τον όρο βάση δεδομένων αναφερόμαστε σε μια συλλογή σχετιζόμενων δεδομένων τμημάτων πληροφορίας ηλεκτρονικά αποθηκευμένων.

Αφού δημιουργήσουμε μια βάση δεδομένων με πολλές εγγραφές, τότε μπορούμε σε αυτήν να ψάξουμε και να βρούμε εύκολα και γρήγορα το/τα στοιχείο/α που επιθυμούμε. Η ηλεκτρονική βάση δεδομένων χρησιμοποιεί ιδιαίτερου τύπου λογισμικό προκειμένου να οργανώσει την αποθήκευση των δεδομένων της. Ένα πρόγραμμα που διαχειρίζεται βάσεις δεδομένων αποκαλείται *Σύστημα Διαχείρισης Βάσεων Δεδομένων (**DBMS**, DataBase Management System)*το οποίο είναι ένα σύνολο από προγράμματα που επιτρέπουν τον χειρισμό των δεδομένων μιας ή περισσοτέρων βάσεων δεδομένων που ανήκουν στο ίδιο σύστημα, και με την βοήθειά του μπορούμε να υποθηκεύσουμε, προσθέσουμε, τροποποιήσουμε, εμφανίσουμε ή και διαγράψουμε τα αποθηκευμένα δεδομένα.

Τα δεδομένα που υπάρχουν στις βάσεις δεδομένων πρέπει να είναι :

- ***Ολοκληρωμένα** (Integrated)*, δηλ. τα δεδομένα πρέπει να είναι αποθηκευμένα σε ομοιόμορφα οργανωμένα σύνολα αρχείων όπου δεν πρέπει να υπάρχει επανάληψη ή πλεονασμός (redundancy) των ίδιων στοιχείων.
- ***Καταμεριζόμενα** (Shared)*, δηλ. να μπορούν περισσότεροι του ενός χρήστες να βλέπουν και να μοιράζονται τα ίδια δεδομένα την ίδια χρονική στιγμή.

Οι στόχοι μιας βάσης δεδομένων είναι οι εξής :

- **Ο περιορισμός της πολλαπλής αποθήκευσης των ίδιων στοιχείων (redundancy).**Εάν τα ίδια δεδομένα καταχωρηθούν στη βάση δυο φορές, τότε ανακύπτουν δυο σοβαρά προβλήματα. Το πρώτο πρόβλημα είναι ότι σπαταλούμε άσκοπα αποθηκευτικό χώρο στο σκληρό δίσκο, αφού την ίδια πληροφορία την αποθηκεύουμε δυο φορές. Το δεύτερο και σοβαρότερο πρόβλημα, είναι ότι υπάρχει ο κίνδυνος δημιουργίας ασυνεπών δεδομένων (inconsistent data). Εάν κρατάμε δύο φορές την ίδια πληροφορία και η πληροφορία αυτή σε κάποια χρονική στιγμή υποστεί κάποιο είδος επεξεργασίας τότε η επεξεργασία αυτή θα πρέπει να εφαρμοσθεί και στις δύο καταχωρήσεις που αφορούν το ίδιο δεδομένο, διότι διαφορετικά, η βάση θα περιέχει δεδομένα που δεν είναι συνεπή. Για το λόγω αυτό, ένας από τους πρώτους ελέγχους που πραγματοποιούμε στη βάση αμέσως μετά το σχεδιασμό της, είναι **ο έλεγχος παρουσίας επαναλαμβανόμενων πεδίων, και η απομάκρυνσή τους, εφ' όσον υπάρχουν.**

- **Ο καταμερισμός (sharing) των ίδιων στοιχείων σ' όλους τους χρήστες.**
- **Η ομοιομορφία (uniformity) στον χειρισμό και την αναπαράσταση των δεδομένων**.
- **Η επιβολή κανόνων ασφαλείας (security).** Όπως να απαγορεύει την πρόσβαση στα δεδομένα μη εξουσιοδοτημένων ατόμων. Αυτό ισχύει κυρίως σε μεγάλες βάσεις δεδομένων

με πολλούς χρήστες, και η τεχνική που συνίσταται είναι ο καθορισμός ομάδων χρηστών (user groups) με διαφορετικά δικαιώματα πρόσβασης στον καθένα από αυτούς. Ο κάθε χρήστης λαμβάνει ένα κωδικό πρόσβασης (password) και τα καθήκοντα που μπορεί να επιτελέσει είναι εντελώς συγκεκριμένα και καθορισμένα εκ των προτέρων.

- **Η διατήρηση της ακεραιότητας (integrity) και της αξιοπιστίας (reliability) των δεδομένων.** Η βάση θα πρέπει να διαθέτει σύστημα δημιουργίας αντιγράφων ασφαλείας των δεδομένων που είναι καταχωρημένα σε αυτή (backups). Η ταυτόχρονη αποθήκευση των δεδομένων σε περισσότερους από ένα δίσκους, είναι μια εργασία επιβεβλημένη, προκειμένου να είναι δυνατή η ανάκτησή τους σε περιπτώσεις κατάρρευσης της βάσης για οποιοδήποτε λόγω**.**
- **Η ανεξαρτησία των δεδομένων (data independence) και των προγραμμάτων από τον φυσικό τρόπο αποθήκευσης των δεδομένων.**

Τα δεδομένα μιας βάσης δεδομένων αποθηκεύονται (οργανώνονται) στις εξής στοιχειώδεις μορφές :

- ***Πεδίο*** *(Field)*, είναι το μικρότερο κομμάτι δεδομένων στο οποίο μπορούμε να αναφερθούμε και περιέχει ένα μόνο χαρακτηριστικό ή ιδιότητα ενός στοιχείου της βάσης δεδομένων.

Ένα πεδίο χαρακτηρίζεται ακόμη και από το είδος των δεδομένων που μπορεί να περιέχει, όπως :

- *Αλφαριθμητικό (alphanumeric)*, μπορεί να περιέχει γράμματα, ψηφία ή και ειδικούς χαρακτήρες.
- *Αριθμητικό (numeric)*, μπορεί να περιέχει μόνο αριθμούς.
- *Αλφαβητικό (alphabetic)*, μπορεί να περιέχει μόνο γράμματα (αλφαβητικούς χαρακτήρες).
- *Ημερομηνίας (date)*, μπορεί να περιέχει μόνο ημερομηνίες.
- *Δυαδικό (binary)*, μπορεί να περιέχει ειδικού τύπου δεδομένα, όπως εικόνες, ήχους κ.ά.
- *Λογικό (logical)*, μπορεί να περιέχει μόνο μία από δύο τιμές, οι οποίες αντιστοιχούν σε δύο διακριτές καταστάσεις και μπορούν να χαρακτηρισθούν σαν 0 και 1 ή σαν αληθές (true) και ψευδές (false).
- *Σημειώσεων (memo)*, μπορεί να περιέχει κείμενο με μεταβλητό μήκος, το οποίο μπορεί να είναι και αρκετά μεγάλο και συνήθως αποθηκεύεται σαν ξεχωριστό αρχείο από το κύριο αρχείο.

- ***Εγγραφή*** *(Record)*, είναι ένα σύνολο από διαφορετικά πεδία που περιέχει όλες τις πληροφορίες για ένα στοιχείο της βάσης δεδομένων.

Όσον αφορά τις εγγραφές, χρήσιμοι ορισμοί είναι οι εξής :

- *Μήκος εγγραφής (record length)* καλείται το άθροισμα που προκύπτει από τα μήκη των πεδίων που την αποτελούν.
- *Δομή εγγραφής (record layout)* ή *γραμμογράφηση* καλείται ο τρόπος με τον οποίο οργανώνουμε τα πεδία μιας εγγραφής.

- *Διάβασμα (read)* από αρχείο σημαίνει τη μεταφορά των δεδομένων του αρχείου, που γίνεται συνήθως ανά μία εγγραφή, από το μέσο αποθήκευσης (σκληρό δίσκο ή δισκέτα) στην κεντρική μνήμη του υπολογιστή για επεξεργασία.

❖ *Γράψιμο (write)* σε αρχείο σημαίνει μεταφορά των δεδομένων του αρχείου, που γίνεται συνήθως ανά μία εγγραφή, από την κεντρική μνήμη του υπολογιστή στο μέσο αποθήκευσης (σκληρό δίσκο ή δισκέτα).

- ***Αρχείο*** *(File)*, είναι ένα σύνολο από πολλά παρόμοια στοιχεία (εγγραφές) της βάσης δεδομένων.
- ***Πρωτεύον Κλειδί*** *(Primary Key)*, είναι ένα πεδίο ή συνδυασμός πεδίων που χαρακτηρίζει μοναδικά μια εγγραφή.
- ***Κλειδί*** *(Key)*, είναι ένα πεδίο που δεν έχει κατ' ανάγκη μοναδική τιμή και που μπορούμε να το χρησιμοποιήσουμε για να κάνουμε αναζήτηση σ' ένα αρχείο.
- ***Ξένο Κλειδί*** *(Foreign Key)*, είναι μια ιδιότητα (πεδίο) που είναι πρωτεύον κλειδί σε μια οντότητα (πίνακας) αλλά που υπάρχει και σε μια άλλη οντότητα (πίνακας) σαν απλή ιδιότητα. Τα ξένα κλειδιά είναι απαραίτητα για να μπορέσουμε να κάνουμε τις συσχετίσεις (συνδέσεις, επικοινωνίες) ανάμεσα στις οντότητες (πίνακες).

Οι βάσεις δεδομένων είναι σημαντικές διότι δίνουν στη σελίδα μας την έννοια της "εφαρμογής". Δηλαδή η σελίδα μας δεν είναι απλά μία στατική σελίδα που απλά μας πληροφορεί για κάτι αλλά μπορούμε να αποθηκεύουμε δεδομένα, να τα επεξεργαζόμαστε να τα χρησιμοποιούμε γενικότερα. Έτσι, η σελίδα μας γίνεται μία δυναμική "σελίδα". Εκατομμύρια sites χρησιμοποιούν βάσεις δεδομένων. Για παράδειγμα amazon, ebay, facebook κλπ. Ολόκληρες πλατφόρμες βασίζονται σε βάσεις δεδομένων όπως η πλατφόρμα blogging "blogspot" ,"wordpress" κλπ.

Όπως αναφέραμε και παραπάνω το ΣΔΒΔ είναι ένα σύνολο από προγράμματα και υπορουτίνες που έχουν να κάνουν με τον χειρισμό της βάσης δεδομένων, όσον αφορά τη δημιουργία, τροποποίηση, διαγραφή στοιχείων, με ελέγχους ασφαλείας κ.ά.

Οι χρήστες των εφαρμογών αντλούν τα στοιχεία που τους ενδιαφέρουν από τη βάση δεδομένων χωρίς να είναι σε θέση να γνωρίζουν με ποιο τρόπο είναι οργανωμένα τα δεδομένα σ' αυτήν. Το ΣΔΒΔ παίζει τον ρόλο του μεσάζοντα ανάμεσα στον χρήστη και τη βάση δεδομένων και μόνο μέσω του ΣΔΒΔ μπορεί ο χρήστης να αντλήσει πληροφορίες από τη βάση δεδομένων. Ένα ΣΔΒΔ μπορεί να είναι εγκατεστημένο σ' έναν μόνο υπολογιστή ή και σ' ένα δίκτυο υπολογιστών και μπορεί να χρησιμοποιείται από έναν χρήστη ή και από πολλούς χρήστες.

Ένα *Σύστημα Βάσης Δεδομένων (ΣΒΔ)* ή *DBS (Data Base System)* αποτελείται από το υλικό, το λογισμικό, τη βάση δεδομένων και τους χρήστες. Είναι δηλαδή ένα σύστημα με το οποίο μπορούμε να αποθηκεύσουμε και να αξιοποιήσουμε δεδομένα με τη βοήθεια ηλεκτρονικού υπολογιστή. Αναλυτικά :

- Το *υλικό (hardware)* αποτελείται όπως είναι γνωστό από τους ηλεκτρονικούς υπολογιστές, τα περιφερειακά, τους σκληρούς δίσκους, τις μαγνητικές ταινίες κ.ά., όπου είναι αποθηκευμένα τα αρχεία της βάσης δεδομένων αλλά και τα προγράμματα που χρησιμοποιούνται για την επεξεργασία τους.
- Το *λογισμικό (software)* είναι τα προγράμματα που χρησιμοποιούνται για την επεξεργασία των δεδομένων (στοιχείων) της βάσης δεδομένων.
- Η *βάση δεδομένων (data base)* αποτελείται από το σύνολο των αρχείων όπου είναι αποθηκευμένα τα δεδομένα του συστήματος. Τα στοιχεία αυτά μπορεί να βρίσκονται αποθηκευμένα σ' έναν φυσικό υπολογιστή αλλά και σε περισσότερους. Όμως, στον χρήστη δίνεται η εντύπωση ότι βρίσκονται συγκεντρωμένα στον ίδιο υπολογιστή. Τα δεδομένα των αρχείων αυτών είναι *ενοποιημένα (data integration)*, δηλ. δεν υπάρχει πλεονασμός (άσκοπη επανάληψη) δεδομένων και *μερισμένα (data sharing)*, δηλ.

υπάρχει δυνατότητα ταυτόχρονης προσπέλασης των δεδομένων από πολλούς χρήστες. Ο κάθε χρήστης έχει διαφορετικά δικαιώματα και βλέπει διαφορετικό κομμάτι της βάσης δεδομένων, ανάλογα με τον σκοπό για τον οποίο συνδέεται.

- Οι *χρήστες (users)* μιας βάσης δεδομένων χωρίζονται στις εξής κατηγορίες :
    - *Τελικοί χρήστες (end users)*. Χρησιμοποιούν κάποια εφαρμογή για να παίρνουν στοιχεία από μια βάση δεδομένων, έχουν τις λιγότερες δυνατότητες επέμβασης στα στοιχεία της βάσης δεδομένων, χρησιμοποιούν ειδικούς κωδικούς πρόσβασης και το σύστημα τούς επιτρέπει ανάλογα πρόσβαση σε συγκεκριμένο κομμάτι της βάσης δεδομένων.
    - *Προγραμματιστές εφαρμογών (application programmers)*. Αναπτύσσουν τις εφαρμογές του ΣΒΔ σε κάποια από τις γνωστές γλώσσες προγραμματισμού.
    - *Διαχειριστής δεδομένων (data administrator – DA)*. Έχει τη διοικητική αρμοδιότητα και ευθύνη για την οργάνωση της βάσης δεδομένων και την απόδοση δικαιωμάτων πρόσβασης στους χρήστες.
    - *Διαχειριστής βάσης δεδομένων (database administrator – DBA)*. Λαμβάνει οδηγίες από τον διαχειριστή δεδομένων και είναι αυτός που διαθέτει τις τεχνικές γνώσεις και αρμοδιότητες για τη σωστή και αποδοτική λειτουργία του ΣΔΒΔ.

3.1 Η αρχιτεκτονική των ΣΔΒΔ

Όπως είδαμε νωρίτερα, ένα ΣΔΒΔ (Σύστημα Διαχείρισης Βάσης Δεδομένων) έχει σαν αποστολή τη διαχείριση των δεδομένων των αρχείων της βάσης, δηλ. την προσθήκη, διαγραφή, τροποποίηση εγγραφών, την αναζήτηση μέσα στις εγγραφές κ.ά.). Το ΣΔΒΔ δέχεται αιτήσεις από τους χρήστες των εφαρμογών και επικοινωνεί με τα αρχεία της βάσης δεδομένων για να τις διεκπεραιώσει.

Αυτή η κοινή διεπαφή (interface) των εφαρμογών με τα αρχεία αποκαλείται *λογική διεπαφή*. Οι εφαρμογές που δημιουργούμε δεν απασχολούνται με τον τρόπο που είναι αποθηκευμένα τα δεδομένα, πόσο χώρο καταλαμβάνουν κοκ και αυτή η ιδιότητα είναι γνωστή ως *ανεξαρτησία δεδομένων*.

Αυτό σημαίνει πρακτικά ότι οποιαδήποτε αλλαγή στον τρόπο οργάνωσης των αρχείων της βάσης δεδομένων δεν θα συνεπάγεται και αλλαγή στις εφαρμογές· ένα πρόβλημα που ταλαιπωρούσε πολύ τους προγραμματιστές παλαιοτέρων εποχών. Ακόμη, η προσθήκη, η κατάργηση ή και η τροποποίηση κάποιων εφαρμογών δεν θα έχει καμία επίπτωση στον τρόπο οργάνωσης των αρχείων της βάσης δεδομένων. Στα ΣΔΒΔ έχει επικρατήσει η λεγόμενη αρχιτεκτονική των τριών επιπέδων (βαθμίδων), όπου τα τρία επίπεδα είναι τα εξής :

- *Εσωτερικό επίπεδο (internal level)*, έχει να κάνει με την αποθήκευση των αρχείων στον σκληρό δίσκο, δηλ. την πραγματική ή φυσική κατάστασή τους.
- *Εξωτερικό επίπεδο (external level)*, έχει να κάνει με τους χρήστες είτε αυτοί είναι απλοί χειριστές, είτε προγραμματιστές ή και οι διαχειριστές της βάσης δεδομένων.
- *Εννοιολογικό επίπεδο (conceptual level)*, είναι ένα ενδιάμεσο επίπεδο που διασυνδέει τα δύο άλλα επίπεδα και έχει να κάνει με τη λογική σχεδίαση των αρχείων της βάσης δεδομένων.

3.2 Συσχετίσεις

Με τον όρο *συσχέτιση (relationship)* αναφερόμαστε στον τρόπο σύνδεσης (επικοινωνίας) δύο ξεχωριστών οντοτήτων, ώστε να μπορούμε να αντλούμε στοιχεία (πληροφορίες) από τον συνδυασμό τους.

3.3 Τα τρία βασικά μοντέλα

Υπάρχουν τρία βασικά μοντέλα που έχουν επικρατήσει στις βάσεις δεδομένων, το ιεραρχικό, το δικτυωτό και το σχεσιακό, και τα οποία αναπτύχθηκαν με βάση αντίστοιχες δομές.

3.3.1 Το Ιεραρχικό Μοντέλο Βάσεων Δεδομένων

Το ιεραρχικό μοντέλο (hierarchical) έχει μια ιεραρχική δομή που θυμίζει δένδρο. Οι οντότητες μοιάζουν με απολήξεις από κλαδιά δένδρων και τοποθετούνται σε επίπεδα ιεραρχίας. Τα κλαδιά παριστάνουν τις συσχετίσεις ανάμεσα στις οντότητες.

Από μια οντότητα που βρίσκεται σ' ένα ανώτερο επίπεδο εκκινούν πολλά κλαδιά, καθένα από τα οποία καταλήγει σε μια οντότητα που βρίσκεται σ' ένα χαμηλότερο επίπεδο. Αλλά, σε κάθε οντότητα που βρίσκεται σ' ένα χαμηλότερο επίπεδο αντιστοιχεί μία και μόνο μία οντότητα που βρίσκεται σ' ένα ανώτερο επίπεδο. Το μοντέλο αυτό ήταν το πρώτο που εμφανίσθηκε αλλά σήμερα θεωρείται δύσχρηστο και ξεπερασμένο.

3.3.2 Το Δικτυωτό Μοντέλο Βάσεων Δεδομένων

Και στο δικτυωτό (network) μοντέλο, τα στοιχεία τοποθετούνται σ' ένα επίπεδο ιεραρχίας, αλλά κάθε στοιχείο μπορεί να συσχετισθεί με πολλά στοιχεία είτε σ' ένα κατώτερο ή σ' ένα ανώτερο επίπεδο.

3.3.3 Το Σχεσιακό Μοντέλο Βάσεων Δεδομένων

Το σχεσιακό (relational) μοντέλο έχει επικρατήσει σήμερα στην αναπαράσταση των δεδομένων καθώς διαθέτει σημαντικά πλεονεκτήματα ως προς τα άλλα δύο και οι βάσεις δεδομένων που σχεδιάζονται σύμφωνα μ' αυτό αποκαλούνται σχεσιακές (relational databases). Με τις σχεσιακές βάσεις δεδομένων διαθέτουμε έναν σαφή, απλό και εύκολα κατανοητό τρόπο για να μπορέσουμε να αναπαραστήσουμε και να διαχειριστούμε τα δεδομένα μας. Υστερούν μόνο σε ταχύτητα υπολογισμών και σε χώρο αποθήκευσης, αλλά μόνο όταν έχουμε να κάνουμε πολύ μεγάλες βάσεις δεδομένων. Στο μοντέλο αυτό οι βάσεις δεδομένων περιγράφονται με αυστηρές μαθηματικές έννοιες και ο χρήστης βλέπει τις οντότητες και τις συσχετίσεις με τη μορφή πινάκων (tables) και σχέσεων (relations) αντίστοιχα.

Στις ***Σχεσιακές*** *(Relational)* βάσεις δεδομένων, τα δεδομένα συνδέονται μεταξύ τους με ***σχέσεις*** *(relations)*, οι οποίες προκύπτουν από τα κοινά πεδία που υπάρχουν σε διαφορετικά αρχεία. Τα αρχεία αποκαλούνται *πίνακες (tables)*, οι εγγραφές *γραμμές (rows)* και τα πεδία *στήλες (columns)*. Η ύπαρξη μιας κοινής τιμής στα πεδία δύο αρχείων καθορίζει και μια σχέση μεταξύ των γραμμών διαφορετικών πινάκων. Κάθε πεδίο του πίνακα μπορεί να πάρει ορισμένες μόνο τιμές, οι οποίες

μπορεί να καθορίζονται από τον τύπο δεδομένων της ιδιότητας, όπως ονόματα ή αριθμοί για παράδειγμα, ή και από αυτό που εκφράζει, όπως το ότι δεν μπορούμε να έχουμε αρνητικό βάρος ή αρνητικό ΑΦΜ, για παράδειγμα. Το σύνολο των αποδεκτών τιμών μιας οντότητας αποκαλείται *πεδίο ορισμού (domain)*. Οι σχεσιακές βάσεις δεδομένων έχουν το πλεονέκτημα ότι είναι λογικά κατανοητές και πολύ ευέλικτες και δεκτικές σε αλλαγές.

Όπως είναι εύκολα κατανοητό, η βασικότερη εργασία που έχουμε να κάνουμε κατά τον σχεδιασμό μιας σχεσιακής βάσης δεδομένων είναι να ορίσουμε τους πίνακες που θα χρησιμοποιήσουμε καθώς και τα πεδία που θα περιέχει ο καθένας απ' αυτούς. Η διαδικασία αυτή αποκαλείται κατασκευή του *σχήματος (schema)* μιας βάσης δεδομένων.

Οι κανόνες που πρέπει να ακολουθούμε πιστά κατά τον σχεδιασμό μιας σχεσιακής βάσης δεδομένων είναι οι εξής :

- Η κάθε οντότητα πρέπει να παριστάνεται ως ένας ξεχωριστός πίνακας.
- Η κάθε στήλη του πίνακα αντιστοιχεί σε μια ιδιότητα της οντότητας.
- Η κάθε γραμμή του πίνακα αντιστοιχεί σε μια εμφάνιση της οντότητας.
- Η κάθε γραμμή πρέπει να είναι μοναδική, δηλ. αποκλείεται να υπάρχουν δύο ή και περισσότερες γραμμές που να περιέχουν τα ίδια ακριβώς στοιχεία.
- Η σειρά εμφάνισης των γραμμών δεν έχει καμία σημασία.

3.4 Τα σχεσιακά ΣΔΒΔ (RDBMS)

Τα Σχεσιακά Συστήματα Διαχείρισης Βάσεων Δεδομένων (ΣΣΔΒΔ) ή RBMS (Relational DataBase Management Systems) αναπτύχθηκαν με βάση το σχεσιακό μοντέλο και έχουν επικρατήσει πλήρως στον χώρο. Κατά τον σχεδιασμό και τη δημιουργία μιας σχεσιακής βάσης δεδομένων, οι πίνακες αποτελούν το μοναδικό δομικό και απαραίτητο στοιχείο για μπορέσουν να αναπαρασταθούν οι πληροφορίες που περιέχονται στη βάση δεδομένων.

Για να μπορέσουμε να προσθέσουμε, διαγράψουμε ή τροποποιήσουμε τα στοιχεία που περιέχονται σε μια βάση δεδομένων, χρησιμοποιούμε ειδικές γλώσσες προγραμματισμού. Η γλώσσα που αποτελεί σήμερα ένα διεθνές πρότυπο για την επικοινωνία των χρηστών με τα Σχεσιακά ΣΔΒΔ είναι η *MYSQL.*

Τα Σχεσιακά ΣΔΒΔ τα διακρίνουμε στα *μεγάλα*, τα οποία αφορούν κυρίως μεγάλους οργανισμούς και επιχειρήσεις, έχουν τεράστιο όγκο δεδομένων και πολλούς χρήστες ταυτόχρονα, και τέτοια συστήματα είναι τα Oracle, Ingres, Informix, SQL Server κ.ά. και τα *μικρά*, τα οποία αφορούν κυρίως απλούς χρήστες, όπως είναι η Microsoft Access, η Paradox, η FoxPro κ.ά.

3.5 Το Μοντέλο Οντοτήτων - Συσχετίσεων

Το μοντέλο που έχει επικρατήσει σήμερα για να παραστήσει τις έννοιες ή τη δομή μιας βάσης δεδομένων είναι το *Μοντέλο Οντοτήτων –Συσχετίσεων (ΟΣ)*. Οι βασικές (θεμελιώδεις) έννοιες του μοντέλου αυτού είναι οι εξής :

- Οντότητες
- Ιδιότητες ή Χαρακτηριστικά
- Συσχετίσεις

Για να αναπαραστήσουμε ένα Μοντέλο Οντοτήτων – Συσχετίσεων χρησιμοποιούμε ειδικά διαγράμματα, όπου τα ορθογώνια συμβολίζουν τις οντότητες, οι ρόμβοι τις συσχετίσεις και οι ελλείψεις τις ιδιότητες. Με ευθείες γραμμές συνδέουμε τις οντότητες που συσχετίζονται με κάποιο τρόπο μεταξύ τους. Όλα τα παραπάνω αποτελούν τη λογική δομή μιας βάσης δεδομένων, μια εργασία που είναι απαραίτητο να γίνει πριν από την καταχώριση και την επεξεργασία των στοιχείων (πληροφοριών) της βάσης δεδομένων.

Το μοντέλο οντοτήτων – συσχετίσεων αποτελεί μια γενική περιγραφή των γενικών στοιχείων που απαρτίζουν μια βάση δεδομένων και απεικονίζει την αντίληψη που έχουμε για τα δεδομένα (εννοιολογικό), χωρίς να υπεισέρχεται σε λεπτομέρειες υλοποίησης.

3.5.1 Οι Οντότητες

Με τον όρο *οντότητα (entity)* αναφερόμαστε σε κάθε αντικείμενο, έννοια, πρόσωπο ή κατάσταση που έχει μια ανεξάρτητη ύπαρξη. Είναι κάτι που ξεχωρίζει και μπορούμε να συγκεντρώσουμε πληροφορίες (στοιχεία) γι' αυτό. Η οντότητα είναι αντίστοιχη με την έννοια της εγγραφής που συναντάμε στα αρχεία και στους πίνακες αλλά και με την έννοια του αντικειμένου στις σύγχρονες αντικειμενοστραφείς γλώσσες προγραμματισμού.

Παραδείγματα οντοτήτων είναι τα εξής : ΜΕΛΟΣ, ΕΣΟΔΑ, ΕΞΟΔΑ, ΚΙΝΗΣΕΙΣ ΜΕΛΟΥΣ κ.ά.

Μια βάση δεδομένων μπορεί να περιέχει πολλές διαφορετικές οντότητες, οι οποίες απεικονίζονται με ορθογώνια παραλληλόγραμμα και συσχετίζονται μεταξύ τους ανά δύο.

3.5.2 Οι Ιδιότητες (Χαρακτηριστικά) των Οντοτήτων

Με τον όρο *ιδιότητες (properties)* ή *χαρακτηριστικά (attributes)* αναφερόμαστε στα συστατικά (δομικά) στοιχεία που προσδιορίζουν (αποτελούν) μια οντότητα. Η ιδιότητα είναι αντίστοιχη με την έννοια του πεδίου που συναντάμε στα αρχεία και στους πίνακες αλλά και με την έννοια της μεταβλητής στις γλώσσες προγραμματισμού.

Για παράδειγμα, η οντότητα ΜΕΛΟΣ μπορεί να αποτελείται από τις ιδιότητες (χαρακτηριστικά) Αριθμός Μητρώου, Επώνυμο, Όνομα, Πατρώνυμο, Ειδικότητα, Έτος Γέννησης, Διεύθυνση, ΑΦΜ, Τηλέφωνο, Κινητό κ.ά.

Απ' όλες τις ιδιότητες μιας οντότητας, υπάρχει μία μόνο ιδιότητα, και σπανιότερα ένας συνδυασμός δύο ή και περισσοτέρων ιδιοτήτων, η τιμή της οποίας είναι μοναδική και προσδιορίζει την κάθε εμφάνιση (στιγμιότυπο) της οντότητας και αποκαλείται ***πρωτεύον κλειδί*** *(primary key)*. Το πρωτεύον κλειδί εμφανίζεται στα διαγράμματα με υπογράμμιση ή με έντονη γραφή ή έχει ως πρόθεμα τον χαρακτήρα #.

3.5.3 Τα κλειδιά

Όπως είδαμε και νωρίτερα, με τον όρο *κλειδί (key)* ή πιο σωστά *πρωτεύον κλειδί (primary key)* αναφερόμαστε σε μια ιδιότητα (πεδίο), ή σπανιότερα σ' ένα σύνολο ιδιοτήτων (πεδίων), η τιμή της οποίας είναι μοναδική σ' ολόκληρη την οντότητα (πίνακας). Στην πράξη, το πρωτεύον κλειδί έχει διαφορετική τιμή για κάθε εμφάνιση της οντότητας ή για κάθε γραμμή (εγγραφή) του πίνακα και ποτέ

δεν μπορεί να έχει μηδενική (κενή) τιμή (null). Προσοχή, άλλο πράγμα είναι ο αριθμός 0 και άλλο πράγμα είναι η κενή τιμή (null), δηλ. η μη ύπαρξη τιμής.

Ο συνδυασμός δύο ή και περισσοτέρων ιδιοτήτων (πεδίων) για τη δημιουργία ενός πρωτεύοντος κλειδιού αποκαλείται *σύνθετο κλειδί*. Ένα παράδειγμα σύνθετου κλειδιού θα μπορούσε να είναι ο συνδυασμός των ιδιοτήτων Επώνυμο, Όνομα και Πατρώνυμο, εφόσον φυσικά είμαστε απολύτως βέβαιοι ότι δεν υπάρχουν δύο ή και περισσότερα άτομα με κοινές τιμές στις παραπάνω ιδιότητες.

Ξένο κλειδί αποκαλείται μια ιδιότητα (πεδίο) που είναι πρωτεύον κλειδί σε μια οντότητα (πίνακας) αλλά που υπάρχει και σε μια άλλη οντότητα (πίνακας) σαν απλή ιδιότητα. Τα ξένα κλειδιά είναι απαραίτητα για να μπορέσουμε να κάνουμε τις συσχετίσεις (συνδέσεις, επικοινωνίες) ανάμεσα στις οντότητες (πίνακες).

3.5.4 Συσχετίσεις

Το τρίτο χαρακτηριστικό γνώρισμα που περιγράφει τη δομή ενός μοντέλου οντοτήτων συσχετίσεων, είναι η συσχέτιση (relationship), η οποία καθορίζει με ποιο τρόπο οι τύποι οντότητας μιας βάσης δεδομένων, συσχετίζονται μεταξύ τους.

Κάθε τύπος οντότητας που συμμετέχει σε κάποιο τύπο συσχέτισης, παίζει και ένα συγκεκριμένο **ρόλο** σε αυτή τη συσχέτιση. Το **όνομα** αυτού του ρόλου (role name) καθορίζει σε πολύ μεγάλο βαθμό **το πώς οι διάφοροι τύποι οντότητας αλληλεπιδρούν μεταξύ τους**.

3.5.5 Χαρακτηριστικά τύπων συσχέτισης

Ο κάθε τύπος συσχέτισης που ορίζεται ανάμεσα σε δύο τύπους οντότητας κατά τη λογική σχεδίαση μιας βάσης δεδομένων, χαρακτηρίζεται από δύο ιδιότητες που καθορίζουν το πλήθος και το είδος της συμμετοχής των δύο οντοτήτων στα στιγμιότυπα αυτού του τύπου συσχέτισης. Αυτές οι δύο ιδιότητες ορίζονται με τον ακόλουθο τρόπο:

Η **πολλαπλότητα (cardinality)** ενός τύπου συσχέτισης, καθορίζει **το πλήθος των στιγμιότυπων αυτού του τύπου συσχέτισης, στον οποίο μια οντότητα μπορεί να συμμετάσχει**.

Ένα είδος πολλαπλότητας του τύπου συσχέτισης που ορίζεται ανάμεσα στους τύπους οντότητας μπορεί να είναι **1:N (ένα προς πολλά)**. Εκτός από αυτού του είδους την πολλαπλότητα ένας τύπος συσχέτισης μπορεί να χαρακτηρίζεται από πολλαπλότητα **1:1 (ένα προς ένα)** ή **M:N (πολλά προς πολλά)**.

Η δεύτερη σημαντική ιδιότητα που χαρακτηρίζει ένα τύπο συσχέτισης έχει να κάνει με το εάν η συμμετοχή των δύο τύπων οντότητας σε αυτή τη συσχέτιση είναι **ολική** ή **μερική**. Ένας τύπος οντότητας χαρακτηρίζεται από **ολική συμμετοχή (total participation)** σε ένα τύπο συσχέτισης, **εάν η ύπαρξή του εξαρτάται από τη συμμετοχή του ή όχι σε αυτή τη συσχέτιση**.

Είναι τέλος σημαντικό να αναφερθεί, πως ένας τύπος συσχέτισης, εκτός από τους τύπους οντότητας μεταξύ των οποίων ορίζεται, μπορεί να χαρακτηρίζεται και από την ύπαρξη πεδίων (attributes).

3.5.6 Αδύναμοι τύποι οντότητας (Weak Entity Types)

Σε ορισμένες περιπτώσεις είναι δυνατό να ορίσουμε τύπους οντότητας οι οποίοι **δεν έχουν κάποιο πεδίο κλειδί έτσι ώστε να ξεχωρίζουν τα διάφορα στιγμιότυπά τους, αλλά αυτή η διάκριση γίνεται δια της συσχέτισής τους με κάποιον από τους υπόλοιπους τύπους οντότητας** της

βάσης. Αυτοί οι τύποι οντότητας ονομάζονται αδύναμοι τύποι οντότητας (weak entity types) και ορίζονται μόνο από ένα συνδυασμό κάποιων από τα πεδία που περιέχουν καθώς και από τη συσχέτισή τους (identifying relationship) με κάποιον τύπο οντότητας (identifying owner). Ας σημειωθεί πως σε αυτή τη συσχέτιση, οι εν λόγω τύποι οντότητας έχουν ολική συμμετοχή (total participation).

Εάν δύο ή περισσότεροι αδύναμοι τύποι οντότητας έχουν τον ίδιο **κάτοχο (owner entity)** υπάρχει ένας συνδυασμός πεδίων ο οποίος βοηθά να ξεχωρίζουμε τα στιγμιότυπα αυτού του τύπου οντότητας μεταξύ τους. Αυτός ο συνδυασμός των πεδίων λέγεται **μερικό κλειδί (partial key)**.

4. My SQL

Η βάση δεδομένων **MySQL**® έχει γίνει η πιο δημοφιλής βάση δεδομένων ανοικτού κώδικα λόγω της αυξημένης απόδοσης, υψηλής αξιοπιστίας και ευκολίας στην διαχείριση της. Η MySQL θεωρείται μια από τις πιο αξιόπιστες πλατφόρμες για την συγκέντρωση δεδομένων και για αυτό έχει κερδίσει την εκτίμηση τόσο των administrators όσο και των programmers. Διατίθεται εντελώς δωρεάν και θα την συναντήσουμε να υποστηρίζει εφαρμογές από το τομέα των τηλεπικοινωνιών, και ηλεκτρονικών καταστημάτων στο διαδίκτυο ως και συστήματα εξυπηρέτησης πελατών και μικροσυσκευές όπως PDAs. Χρησιμοποιείται σε περισσότερες από 6 εκατομμύρια εγκαταστάσεις κλιμακούμενες από μεγάλες εταιρίες μέχρι εξειδικευμένες εφαρμογές με ενσωματωμένες βάσεις δεδομένων σε όλο τον κόσμο.

Η **MySQL** καθίσταται η απόλυτη επιλογή μιας νέας γενιάς εφαρμογών που υλοποιούνται σε περιβάλλον LAMP (Linux, Apache, MySQL, PHP / Perl / Python).
Επίσης η MySQL λειτουργεί σε περισσότερες από 20 πλατφόρμες συμπεριλαμβανομένων των Linux, Windows, OS/X, HP - UX, AIX, Netware. Προϋπόθεση για να έχουμε και να λειτουργούμε MySQL βάσεις δεδομένων είναι να μας παρέχετε ένας MySQL server. Η MySQL είναι το λογισμικό του διακομιστή βάσεων δεδομένων *(*database server software*)* που χρησιμοποιούμε.

Χρήσιμα εργαλεία είναι ο MySQL Administrator που έχει σχεδιαστεί για την διαχείριση ενός MySQL εξυπηρετητή και ο MySQL Query Browser που έχει σχεδιαστεί για να μας βοηθήσει να θέτουμε αιτήματα και να αναλύουμε δεδομένα, τα οποία είναι αποθηκευμένα στην MySQL βάση δεδομένων μας.

Η MySQL ανήκει στην γενική κατηγορία των DBMS (Database Management Systems) που έχουν σαν πρωταρχικό τους ρόλο την αποθήκευση των δεδομένων, την ελεγχόμενη και ασφαλή πρόσβαση στις πληροφορίες, και την διαχείριση των δικαιωμάτων με τα οποία οι χρήστες μπορούν να επέμβουν και να αλλάξουν στοιχεία πληροφοριών. Κάθε φορά που ένας χρήστης προσπαθεί να διαχειριστεί ένα όγκο δεδομένων προσθέτοντας νέα στοιχεία ή αφαιρώντας κάποια άλλα που δεν ισχύουν πια, το DBMS είναι υπεύθυνο να ακολουθήσει αυτή την διαδικασία από την αρχή της ενεργοποίησης της μέχρι το τέλος της. Παίζει το ρόλο του "μεσάζων" γιατί απλά το DBMS δεν αφήνει το χρήστη να έχει άμεση πρόσβαση στα δεδομένα. Οι εντολές απευθύνονται αποκλειστικά στο DBMS και αφού ελεγχθούν για την εγκυρότητα τους τότε το σύστημα αναλαμβάνει την μεταφορά της πληροφορίας στο χρήστη που την αναζήτησε.

Η MySQL ανήκοντας στην κατηγορία των DBMS μπορεί να αναλάβει την διαχείριση πολλών βάσεων μαζί. Κατά την εγκατάσταση έχει ήδη δημιουργηθεί μια βάση με το όνομα mysql η οποία χρησιμοποιείται αποκλειστικά για την καταγραφή πληροφοριών του συστήματος.

Αυτές οι πληροφορίες καταγράφουν την δημιουργία νέων βάσεων, νέων πινάκων, ή ακόμα ενέργειες που έχουν ζητηθεί από τον χρήστη. Με άλλα λόγια, για οποιοδήποτε νέο στοιχείο που δημιουργείται μετά την εγκατάσταση της MySQL, πρέπει το σύστημα να ενημερώνεται και να καταγράφει την ύπαρξη του.

4.1 Γιατί MySQL

Παρέχει απλοποίηση στην πρόσβαση, βελτίωση στις επιδόσεις, την επεκτασιμότητα και την αξιοπιστία και ικανοποίηση στην εκθετική αύξηση των απαιτήσεων σε αποθήκευση με δραματικά χαμηλότερο κόστος. Επωφελούμαστε από την εξειδίκευση, τις βέλτιστες πρακτικές, την εξυπηρέτηση, την υποστήριξη και τη διαχείριση της MySQL για σύνθετα περιβάλλοντα.

Η MySQL μειώνει το συνολικό κόστος κτήσης του λογισμικού βάσης δεδομένων. Επίσης υπάρχει μείωση του κόστους αδειοδότησης της βάσης δεδομένων κατά 90 τοις εκατό, μείωση του χρόνου εκτός λειτουργίας των συστημάτων κατά 60 τοις εκατό, μείωση των δαπανών για υλικό

εξοπλισμού κατά 70 τοις εκατό και μείωση του κόστους διαχείρισης, τεχνικού σχεδιασμού και υποστήριξης έως και 50 τοις εκατό.

Παρακάτω αναφέρονται κάποια χαρακτηριστικά και κάποια πλεονεκτήματα της MySQL **:**

Επεκτασιμότητα και ευελιξία

Τα συστήματα της Sun για MySQL μπορούν να επεκταθούν, για να ικανοποιήσουν την εκθετική αύξηση του όγκου δεδομένων που χαρακτηρίζει τα εμπλουτισμένα πολυμέσα, τις Διαδικτυακές κοινότητες και άλλες υπηρεσίες Web. Εκτελείται οτιδήποτε από βαθιά ενσωματωμένες εφαρμογές με καταλαμβανόμενο χώρο μόλις 1MB, ή μαζικές "αποθήκες" δεδομένων - με terabyte πληροφοριών. Επίσης τα μοναδικά στο είδος τους συστήματα της Sun μπορούν να "επεκταθούν μέσα στο διακομιστή, ενώ η τεχνολογία εικονικοποίησης της Sun μπορεί να μειώσει τις απαιτήσεις σε κατανάλωση ρεύματος και χώρο και να επιτύχει εξοικονόμηση κόστους και μεγαλύτερο σεβασμό προς το περιβάλλον.

Υψηλή διαθεσιμότητα.

Εκτέλεση διαμορφώσεων υψηλής ταχύτητας κύριας /εξαρτώμενης αναπαραγωγής βασισμένη σε γραμμές και υβριδική αναπαραγωγή. Οι εξειδικευμένοι διακομιστές συμπλεγμάτων προσφέρουν άμεση μεταγωγή μετά από αστοχία.

Πλεονεκτήματα αποθήκευσης δεδομένων και web.

Υψηλής απόδοσης μηχανισμός ερωτημάτων. Εξαιρετικά γρήγορη δυνατότητα εισαγωγής δεδομένων. Ισχυρή υποστήριξη για εξειδικευμένες λειτουργίες web - συμπεριλαμβανομένων των γρήγορων αναζητήσεων πλήρους κειμένου.

Ισχυρή προστασία δεδομένων.

Πανίσχυροι μηχανισμοί διασφάλισης της προσπέλασης μόνο από εξουσιοδοτημένους χρήστες. Υποστήριξη SSH και SSL για ασφαλείς και σίγουρες συνδέσεις. Ισχυρή κρυπτογράφηση δεδομένων και λειτουργίες αποκρυπτογράφησης.

Ολοκληρωμένη ανάπτυξη εφαρμογών.

Υποστήριξη για υποθηκευμένες διαδικασίες, κανόνες ενεργοποίησης, λειτουργίες, προβολές, δρομείς, γλώσσες SQL προτύπου ANSI, και πολλά περισσότερα. Βιβλιοθήκες προσθέτων για την ενσωμάτωση της υποστήριξης βάσεων δεδομένων MySQL σχεδόν σε κάθε εφαρμογή.

Ευκολία διαχείρισης.

Υπάρχει χρήση του προγραμματισμού συμβάντων - αυτόματος προγραμματισμός συνήθων επαναλαμβανόμενων εργασιών βασισμένων σε SQL για εκτέλεση σε διακομιστή βάσης δεδομένων. Επίσης ο μέσος χρόνος από τη λήψη λογισμικού μέχρι την ολοκλήρωση της εγκατάστασης είναι λιγότερος από δεκαπέντε λεπτά.

Ελευθερία ανοικτού πηγαίου κώδικα και υποστήριξη 24 ώρες το εικοσιτετράωρο, 7 ημέρες την εβδομάδα.

Μπορεί να προσφέρει εικοσιτετράωρη υποστήριξη και διαθέσιμη επανόρθωση μέσω του δικτύου MySQL. Εταιρική ποιότητα και έτοιμο για εταιρική χρήση από την εγκατάσταση ως την υποστήριξη.

Το χαμηλότερο συνολικό κόστος κτήσης.

Εξοικονόμηση κόστους αδειοδότησης βάσεων δεδομένων και εξόδων για υλικό εξοπλισμού με ταυτόχρονη μείωση του χρόνου εκτός λειτουργίας.

Με γνώμονα την ταχύτητα.

Χάρη στο συνδυασμό πρωτοποριακών τεχνολογιών, της εξειδίκευσης σε συστήματα και των προηγμένων μεθόδων ρύθμισης της Sun, τα συστήματα της Sun για MySQL μας προσφέρουν μια αποτελεσματική και ολοκληρωμένη λύση MySQL. Έτσι, είναι πλέον ευκολότερο για μας να υλοποιήσουμε MySQL και να ικανοποιήσουμε απαιτήσεις για νέες υπηρεσίες Web – με αύξηση των επιδόσεων μέχρι και 3 φορές.

Απλότητα πάνω απ' όλα.

Παρέχει από τα εισαγωγικά συστήματα μέχρι τους πανίσχυρους διακομιστές, με επεξεργαστές Intel, AMD ή SPARC, μεγάλη μνήμη, ταχύτατες λειτουργίες I/O και τις πλέον πρόσφατες τεχνολογίες. Ακόμη όλα τα συστήματα της Sun για MySQL βελτιστοποιούνται και προσαρμόζονται σύμφωνα με τα εκάστοτε περιβάλλοντα MySQL. Επίσης μπορούμε να εκτελέσουμε το λειτουργικό σύστημα της επιλογής μας: Linux, Windows, Solaris 10 ή OpenSolaris.

Μείωση στους κινδύνους με το MySQL Experts.

Οι βέλτιστες πρακτικές, οι αρχιτεκτονικές αναφορές και τα blueprint της Sun, που διατίθενται προς λήψη δωρεάν, σε συνδυασμό με τις δεξιότητες ρύθμισης και υλοποίησης MySQL της Sun και τον οργανισμό επαγγελματικών υπηρεσιών της Sun, μπορούν να μας βοηθήσουν να συνθέσουμε γρήγορα και με ασφάλεια τη δική μας υποδομή Web.

4.2 Δημιουργία Βάσης

Για να δημιουργήσουμε μια βάση χρησιμοποιούμε την εντολή `create database`:

```
mysql> create database members;
```

Για να εργαστούμε με μία συγκεκριμένη βάση χρησιμοποιούμε την εντολή `use`:

```
mysql> use members;
```

Για να δημιουργήσουμε έναν πίνακα χρησιμοποιούμε την εντολή `create table`:

```
CREATE TABLE members (

Id INT (11) NULL,

name VARCHAR(255) NOT NULL,
city VARCHAR (255) NOT NULL,
state VARCHAR (50) NOT NULL,
PRIMARY KEY (id)
)
ENGINE=InnoDB;
```

Για να σβήσουμε έναν πίνακα χρησιμοποιούμε την εντολή `drop table tablename`.

Για να αλλάξουμε τον ορισμό ενός πίνακα χρησιμοποιούμε την εντολή `alter table tablename`.

4.2.1 Εισαγωγή Δεδομένων

Για την εισαγωγή δεδομένων χρησιμοποιείται η εντολή `insert into`.

Για να εισάγουμε τρία μέλη θα δώσουμε:

INSERT INTO members (name, city, state)

VALUES (‘Anna Rasoylh’, ‘Chania’, ‘PL’)

INSERT INTO members (name, city, state)

VALUES (‘Rena Zaxou’, ‘Heraklion’, ‘PL’)

INSERT INTO members (name, city, state)

VALUES (‘Maria Nikolaou’, ‘Heraklion’, ‘PL’)

4.2.2 Επιλογή Δεδομένων

Η επιλογή δεδομένων στη γλώσσα SQL γίνεται με τη χρήση της εντολής `select`.

Για να επιλέξουμε όλα τα στοιχεία για τα μέλη δίνουμε:

```
select * from members;
```

Για να επιλέξουμε όνομα μέλους δίνουμε:

```
Select member_name
From members;
```

4.2.3 Ενημερώσεις και Διαγραφές

Για να ενημερώσουμε υπάρχοντα δεδομένα στη βάση χρησιμοποιούμε την εντολή `update`.
Για να διαγράψουμε δεδομένα χρησιμοποιούμε την εντολή `delete`.

4.3 JDBC

Η τεχνολογία JDBC (σημαίνει *Java Database Connectivity*) μας επιτρέπει να έχουμε πρόσβαση σε μία σχεσιακή βάση δεδομένων μέσα από το πρόγραμμά μας. Αυτή η τεχνολογία βασίζεται στην ύπαρξη οδηγών (*drivers*), οι οποίοι είναι αρμόδιοι για την επικοινωνία μεταξύ του προγράμματος και της βάσης. Με τη χρήση του κατάλληλου οδηγού συνδεόμαστε με τη βάση, στέλνουμε εντολές και λαμβάνουμε αποτελέσματα. Επίσης οι απαραίτητες κλάσεις ορίζονται από τη βιβλιοθήκη `java.sql`.

4.3.1 Χρήση του Οδηγού

Ο οδηγός JDBC για τη βάση MySQL ονομάζεται `Connector/J`. Για να τον χρησιμοποιήσουμε πρέπει να είναι διαθέσιμος στην ιδεατή μηχανή κατά τη διάρκεια της εκτέλεσης.

Όταν αποσυμπιέσουμε το αρχείο διανομής του Connector/J θα βρούμε τον οδηγό σε ένα αρχείο της μορφής `mysql-connector-java-[version]-stable-bin.jar`.
Τοποθετούμε το αρχείο του οδηγού σε σημείο που πρέπει να θυμόμαστε να το συμπεριλάβουμε στο classpath κατά την εκτέλεση του προγράμματος. Εναλλακτικά, αν `JAVA_HOME` είναι ο κατάλογος εγκατάστασης της γλώσσας Java στο σύστημά μας και `JRE_HOME` είναι ο κατάλογος εγκατάστασης του runtime environment (συνήθως κάτω από τον κατάλογο `Program Files`), μπορούμε να αντιγράψουμε το αρχείο του οδηγού στον κατάλογο `JAVA_HOME\jre\lib\ext και στον κατάλογο \lib\ext`.

4.4 Downloading MySQL

Το πρώτο βήμα για την εγκατάσταση του MySQL για Windows είναι να κατεβάσουμε την τελευταία και πιο σταθερή έκδοση που είναι διαθέσιμη από το site της MySQL. Οι νέες εκδόσεις κυκλοφορούν συχνά, αλλά η πραγματική διαδικασία ρύθμισης θα πρέπει να παραμείνει η ίδια. Ανοίγουμε το πρόγραμμα περιήγησής μας και αναζητούμε το αρχείο εγκατάστασης στο http://dev.mysql.com/downloads/mysql/. Η οποία είναι η MySQL σελίδα για τα downloads. Από την κατηγορία Windows downloads επιλέγουμε το Windows ZIP/Setup.EXE (x86).

Για την εγκατάσταση της MySQL σε περιβάλλον Windows υπάρχουν τρία διαφορετικά πακέτα που μπορούμε να χρησιμοποιήσουμε.

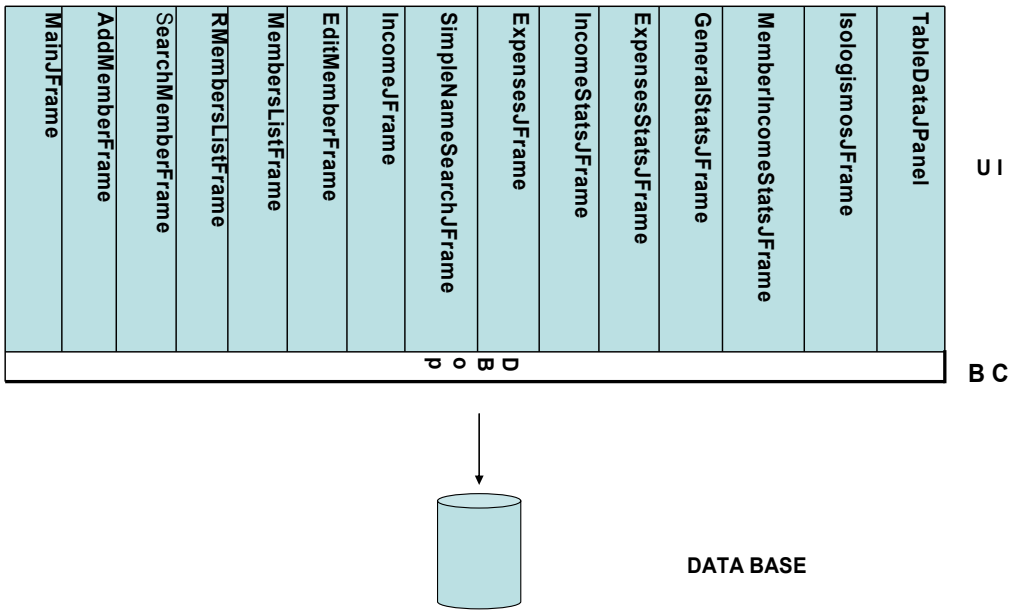
Η πρώτη μας επιλογή είναι το Windows Essentials(x86) αρχείο που έχει κατάληξη .msi (η στάνταρ κατάληξη πακέτων εγκατάστασης σε λειτουργικά συστήματα της Microsoft). Αυτό περιέχει τα βασικά στοιχεία που χρειάζεται ο μέσος χρήστης σε μια βάση δεδομένων περιλαμβανομένου και ενός γραφικού Configuration Wizard που μας βοηθάει με τις επιλογές που διαθέτει να τροποποιήσουμε την εγκατάσταση όπως εμείς θέλουμε.

Το δεύτερο πακέτο (Windows ZIP/Setup.EXE(x86)) περιλαμβάνει ότι και το πρώτο, μαζί με πολλά έξτρα στοιχεία όπως μια συλλογή από εργαλεία για μετρήσεις απόδοσης της βάσης. Το αρχείο αυτό είναι σε μορφή ZIP που σημαίνει ότι πρέπει να έχουμε ένα απλό πρόγραμμα συμπίεσης / αποσυμπίεσης αρχείων εγκαταστημένο στον υπολογιστή μας (π.χ. WinZIP, WinRAR) για να μπορέσουμε να το αποσυμπιέσουμε πριν το εγκαταστήσουμε. Τα περισσότερα από αυτά τα έξτρα εργαλεία δεν θα χρειαστούν για τον απλό χρήστη ή προγραμματιστή.

Το τελευταίο πακέτο Without installer(unzip in C:\) περιλαμβάνει ότι και τα δύο προηγούμενα εκτός του Configuration Wizard. Αυτό σημαίνει ότι δεν θα έχουμε καμιά γραφική βοήθεια κατά την εγκατάσταση και θα πρέπει να γνωρίζουμε τις κατάλληλες εντολές για να μπορέσουμε να ολοκληρώσουμε επιτυχώς την διαδικασία. Και αυτό το πακέτο έχει κατάληξη ZIP οπότε και εδώ θα χρειαστεί να αποσυμπιέσουμε το αρχείο πριν το εγκαταστήσουμε.

Κατεβάζουμε το αρχείο για την εγκατάσταση επιλέγοντας το σύνδεσμο.

5. Ανάπτυξη Εφαρμογής Διαχείρισης Συλλόγου



Εικόνα 5.α : Αρχιτεκτονική περιγραφή των τριών επιπέδων του συστήματος.

Εδώ παρουσιάζεται με πλήρη ανάλυση η δομή της εφαρμογής Διαχείρισης Συλλόγου. Η ονομασία της εφαρμογής είναι Association Management2 η οποία αποτελείται από διάφορες κλάσεις. Η κύρια κλάση είναι η DBop η οποία περιέχει όλες τις μεθόδους για την επικοινωνία με την βάση. Οι μέθοδοι οι οποίες περιέχονται είναι :

Η μέθοδος **connect ()** η οποία χρησιμοποιείται για να γίνει η σύνδεση με την βάση, έπεται η μέθοδος **connect (String table)** που χρησιμοποιείται για να γίνει η σύνδεση με την βάση και να επιστρέψει τα δεδομένα από ένα πίνακα σε ένα ResultSet, η μέθοδος **disconnect()** που χρησιμεύει για την αποσύνδεση από την βάση και την μέθοδο **getRS (String q)** που χρησιμοποιείται για να κάνει το ερώτημα που ζητάμε στη βάση. Αυτές είναι οι κύριες μέθοδοι για την επικοινωνία της εφαρμογής με την βάση.

Ακολουθούνται και μέθοδοι εξίσου σημαντικοί με τις πρώτες οι οποίες κάνουν και αυτές συγκεκριμένες δουλειές όπως η μέθοδος **getLastAA()** που παίρνει τα id από τον πίνακα μελών τα ταξινομεί και επιστρέφει το επόμενο id για την καταχώρηση νέου μέλους στην βάση, η μέθοδος **getLastAA (String table)** που η διαφορά της με την προηγούμενη είναι ότι αυτή παίρνει τα id από τον πίνακα table τα ταξινομεί και επιστρέφει το επόμενο id για νέα καταχώρηση.

Συνεχίζουμε με την μέθοδο **getLastAM (String table, String column)** που είναι ίδια με την προηγούμενη αλλά επιστρέφει την τελευταία τιμή της στήλης της οποίας ζητάμε τα δεδομένα. Ακολουθείται η μέθοδος **fillCombo (JComboBox box, String table, String field)** η οποία είναι φτιαγμένη για να γεμίσουμε ένα combobox με δεδομένα και η **SaveFields** (String table, String[] fields, Vector data) που χρησιμοποιείται για την αποθήκευση νέων καταχωρήσεων. Μετά είναι η **ErrorCatcher (Exception ex)** η οποία εκτελείτε όταν προσπαθήσουμε να καταχωρίσουμε μη αριθμητικό ποσό σε αριθμητικό πεδίο στην βάση, η μέθοδος **emptyTable (JTable ResTable)** για την αφαίρεση των δεδομένων από ένα JTable. Για να εμφανίσουμε μια λίστα με τα μέλη και κάποια στοιχεία τους τότε η μέθοδος **fillTable (JTable ResTable, int[] ReadCols, String table, String SearchCol, String SearchData)** είναι αρμόδια για αυτή την δουλειά. Η **Date2String (java.util.Date date)** μετατρέπει μια ημερομηνία java.util.Date σε java.sql.Date.

Για αναλυτική λίστα εσόδων ή εξόδων μίας συγκεκριμένης περιόδου τότε η μέθοδος **fillTableDateFiltered (JTable ResTable, int[] ReadCols, String table, String DateColName, String StartDate, String EndDate)** χρησιμοποιείται για το γέμισμα αυτής της λίστας. Η λειτουργία της μεθόδου **fillTableDateIDFiltered (JTable ResTable, int[] ReadCols, String table, String DateColName, String StartDate, String EndDate, String IDcol, String ID)** είναι ίδια με την προηγούμενη με τη μόνη διαφορά ότι αυτή χρησιμοποιεί επιπλέον ένα φίλτρο ώστε να μας δώσει μία λίστα με τις κινήσεις ενός μέλους.

Η **TableChangeId2Description (JTable ResTable, int ReadCol, String table, String DataCol)** χρησιμοποιείτε για μετατρέψουμε ένα id (πεδίο κλειδί κάποιου πίνακα) σε περιγραφή (πχ. από το id της ειδικότητας να πάρω την περιγραφή). Η μέθοδος **getNameFromId (String id, String table, String SearchCol, String DataCol)** βρίσκει το όνομα ή το επίθετο με βάση το Α.Μ. και αυτή την μέθοδο την χρησιμοποιούμε κυρίως όταν θέλουμε να καταχωρήσουμε μία συναλλαγή ενός μέλους. Η μέθοδος **IsUnique (int SearchString, String table, String SearchCol)** χρησιμοποιείται για να ελέγξουμε εάν υπάρχει μέλος με ίδιο Α.Μ. ή με ίδιο Α/Α ώστε να αποφευχθούν διπλοκαταχωρήσεις .

Για να καταχωρήσουμε αλλαγές σε μια εγγραφή ενός πίνακα χρησιμοποιούμε την μέθοδο **EditFields (String table, String SearchCol, String id, String[] fields, Vector data).** Η μέθοδος **incomeTable (JTable ResTable, String StartDate, String EndDate)** γεμίζει ένα πίνακα με τις κατηγορίες των εσόδων, πόσες φορές εμφανίστηκαν και τη έσοδα για κάθε μία σε μια συγκεκριμένη περίοδο. Η **expensesTable (JTable ResTable, String StartDate, String EndDate)** κάνει την ίδια ακριβώς δουλειά με την προηγούμενη αλλά για τα έξοδα. Την μέθοδο **Isologismos (JTable ResTable, JTable ResTable1, String Year)** την χρησιμοποιούμε για την αναλυτική κατάσταση των εσόδων και των εξόδων του έτους.

Η μέθοδος **getRecordData(String table, int id, String[] fields)** υπάρχει για να επιστρέφει μια καταχώρηση με τα δεδομένα του πίνακα μας. Η μέθοδος **Search(String table, Vector data, String[] fields, int[] ReadCols, JTable ResTable)** κάνει αναζήτηση κάποιου μέλους με βάση το /τα στοιχεία που θέλουμε.

MainJFrame

Η MainJFrame είναι το κύριο γραφικό περιβάλλον της εφαρμογής. Περιλαμβάνει τα μενού που πραγματοποιούν κάποιες λειτουργίες της. Αυτή η κλάση χρησιμοποιείται για να δημιουργεί ένα αντίγραφο ασφαλείας των δεδομένων της βάσης, μας επιτρέπει να τερματίσουμε την εφαρμογή μας, μας δίνει την δυνατότητα να ανοίγουμε τις καρτέλες καταχώρησης στοιχείων, εμφανίζει το μενού των ρυθμίσεων. Επίσης περιέχει την ρουτίνα που εμφανίζει την ώρα και την ημερομηνία στο κάτω μέρος της εφαρμογής.

AddMemberFrame

Η **AddMemberFrame** είναι ένα παράθυρο με πεδία που χρησιμοποιείται για την καταχώρηση ενός νέου μέλους στην βάση.

SearchMemberFrame

Είναι ένα παράθυρο με πεδία που χρησιμοποιείται για την εύρεση κάποιου μέλους με όποιο στοιχείο ζητηθεί.

RMembersListFrame

Αυτή η κλάση είναι ένα παράθυρο με ένα πίνακα που θα περιέχει τα δεδομένα που μας επέστρεψε η αναζήτηση της **SearchMemberFrame**.

MembersListFrame

Η **MembersListFrame** είναι ένα παράθυρο με ένα πίνακα που περιέχει όλα τα μέλη και μας δίνει την δυνατότητα εκτύπωσης του πίνακα.

EditMemberFrame

Η **EditMemberFrame** είναι ένα παράθυρο με πεδία που χρησιμοποιείται για την επεξεργασία των στοιχείων κάποιου μέλους και δίνει την δυνατότητα εκτύπωσης της καρτέλας του μέλους. Εμφανίζεται μόνο μέσω των **MembersListFrame** και **RMembersListFrame.**

IncomeJFrame

Είναι ένα παράθυρο με πεδία που χρησιμοποιείται για την καταχώρηση των εσόδων.

SimpleNameSearchJFrame

Είναι ένα παράθυρο με ένα πίνακα που χρησιμοποιείται για να κάνει γρήγορη αναζήτηση κάποιου μέλους και εμφανίζεται μέσω της **IncomeJFrame.**

ExpensesJFrame

Η **ExpensesJFrame** είναι ένα παράθυρο με πεδία που χρησιμοποιείται για την καταχώρηση των εξόδων.

IncomeStatsJFrame

Αυτή η κλάση είναι ένα παράθυρο με πίνακες. Ο πρώτος μας δίνει την δυνατότητα εμφάνισης των εσόδων μιας χρονικής περιόδου και εκτύπωσης του ενώ ο δεύτερος πίνακας περιέχει τα αποτελέσματα που επιστρέφει η μέθοδος της DBop, **incomeTable (JTable ResTable, String StartDate, String EndDate).**

ExpensesStatsJFrame

Είναι ένα παράθυρο με πίνακες. Ο πρώτος μας δίνει την δυνατότητα εμφάνισης των εξόδων μιας χρονικής περιόδου και εκτύπωσης του ενώ ο δεύτερος πίνακας περιέχει τα αποτελέσματα που επιστρέφει η μέθοδος της DBop, **expensesTable (JTable ResTable, String StartDate, String EndDate).**

GeneralStatsJFrame

Είναι ένα παράθυρο με πίνακες που μου εμφανίζει την κατάσταση του ταμείου για μία χρονική περίοδο καθώς υπάρχει και η δυνατότητα εκτύπωσης των πινάκων.

MemberIncomeStatsJFrame

Η **MemberIncomeStatsJFrame** είναι ένα παράθυρο με πίνακα η οποία χρησιμοποιείται για την ανάλυση των κινήσεων κάποιου μέλους. Επίσης υπάρχει δυνατότητα εκτύπωσης αυτού του πίνακα.

IsologismosJFrame

Αυτή η κλάση είναι ένα παράθυρο με πίνακες που περιέχουν τα έσοδα και τα έξοδα ενός έτους. Επίσης έχουμε την δυνατότητα της εκτύπωσης του κάθε πίνακα.

TableDataJPanel

Είναι ένα παράθυρο με τέσσερις καρτέλες οι οποίες είναι οι εξής : **Ειδικότητες , Κατηγορίες Εσόδων, Κατηγορίες Δαπανών, Καταστάσεις Μελών.** Καθεμία από αυτές χρησιμοποιείται για την εισαγωγή - εξαγωγή στοιχείων καθώς και την επεξεργασία των στοιχείων αυτών.

Βιβλιογραφία

1. Java «Εισαγωγή στη σύγχρονη τεχνολογία» Greanier, Todd (Συγγραφέας) Σαμαράς, Γιάννης Β. (Μεταφραστής). Εκδόσεις Γκιούρδας Μ. 2005.
2. Εισαγωγή στην Java 2 Λιακέας Γ. Εκδόσεις «ΚΛΕΙΔΑΡΙΘΜΟΣ» 2003.
3. ΕΙΣΑΓΩΓΗ ΣΤΑ ΣΥΣΤΗΜΑΤΑ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ / AN INTRODUCTION TO DATABASE SYSTEMS C.J. DATE Εκδόσεις «ΚΛΕΙΔΑΡΙΘΜΟΣ» 2003.
4. Java Hellenic User Group (JHUG). 2002
5. http://javalessons.com
6. http://www.mysql-tools.com/en/mysql-lessons.html
7. http://www.mysql.com/training/
8. http://download.oracle.com/javase/tutorial/
9. http://www.it.uom.gr/project/java/tutorial.htm
10. World Wide Web

ΠΑΡΑΡΤΗΜΑΤΑ
ΑΝΑΠΤΥΞΗΣ ΕΦΑΡΜΟΓΗΣ ΔΙΑΧΕΙΡΙΣΗΣ ΣΥΛΛΟΓΟΥ

ΠΑΡΑΡΤΗΜΑ Α : Εγκαταστάσεις προγραμμάτων σε λειτουργικά
ΠΑΡΑΡΤΗΜΑ Β : Εγχειρίδιο χρήσης του προγράμματος
ΠΑΡΑΡΤΗΜΑ Γ : API Εφαρμογής
ΠΑΡΑΡΤΗΜΑ Δ : Παρουσίαση Εφαρμογής

ΠΑΡΑΡΤΗΜΑ Α : Εγκαταστάσεις προγραμμάτων σε λειτουργικά

Παράρτημα Α1: Εγκατάσταση του NetBeans σε Windows

Μετά την ανάλυση γύρω από το JVM (Java Virtual Machine) είμαστε έτοιμοι να εγκαταστήσουμε το περιβάλλον του NetBeans μέσα από το οποίο θα αναπτύξουμε την εφαρμογή μας. Το πρώτο και απαραίτητο βήμα είναι να κατεβάσουμε από την Sun Microsystems το JDK πακέτο το οποίο περιέχει τον Java compiler μαζί με το γραφικό περιβάλλον του NetBeans (εάν έχουμε ήδη το JDK κατεβάζουμε μόνο το NetBeans) . Το πλήρης πρόγραμμα θα το βρούμε στην ακόλουθη ηλεκτρονική διεύθυνση: http://www.oracle.com/technetwork/java/javase/downloads/jdk-netbeans-jsp-142931.html

Επιλέγοντας την επιλογή ***DOWNLOAD*** (στα δεξιά της οθόνης), θα μεταφερθούμε σε μια νέα σελίδα που μας ζητάει με απλό τρόπο να ορίσουμε την πλατφόρμα στην οποία πρόκειται να εγκαταστήσουμε το NetBeans περιβάλλον και σε ποια γλώσσα επιθυμούμε να παρουσιάζονται τα μενού του συγκεκριμένου προγράμματος. Πριν επιλέξουμε το κουμπί ***CONTINUE*** θα πρέπει να συμφωνήσουμε με τους όρους άδειας που συνοδεύουν το πρόγραμμα. Αυτό πραγματοποιείται επιλέγοντας το κενό κουτάκι δίπλα από το "I agree to...".

Στο επόμενο παράθυρο που θα εμφανιστεί, απλά επιλέγουμε τον κόκκινο σύνδεσμο (link) και όταν εμφανιστεί η επιλογή αποθήκευσης από το browser, επιλέγουμε μια τοποθεσία στο σκληρό μας δίσκο στο οποίο θα αποθηκευτεί το αρχείο.

Πηγαίνουμε στο φάκελο μέσα στο οποίο βρίσκεται αποθηκευμένο το αρχείο και ξεκινάμε την διαδικασία εγκατάστασης του προγράμματος κάνοντας διπλό κλικ επάνω στο εικονίδιο.

Το πρώτο παράθυρο που θα εμφανιστεί σχεδόν αμέσως μετά από την ενεργοποίηση του προγράμματος μας ειδοποιεί ότι στα επόμενα βήματα που θα ακολουθήσουν θα εγκατασταθούν δύο προγράμματα – το Java SE Development Kit (JDK) και αμέσως μετά το περιβάλλον του NetBeans. Σε αυτό το παράθυρο απλά επιλέγουμε το ***NEXT***.

Η διαδικασία εγκατάστασης μας ενημερώνει στο αμέσως επόμενο παράθυρο ότι θα δημιουργηθεί ένας φάκελος με το όνομα Java κάτω από το directory Program Files των Windows μέσα στον οποίο θα γίνει εγκατάσταση το JDK. Δεν έχουμε κανένα απολύτως λόγω να διαφωνήσουμε με αυτή την προκαθορισμένη επιλογή και απλά επιλέγουμε το κουμπί ***NEXT*** για να συνεχιστεί η διαδικασία.

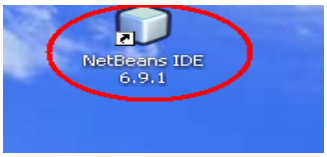
Μετά από το JDK, σειρά έχει το πρόγραμμα NetBeans να δημιουργήσει ένα φάκελο στον υπολογιστή μας μέσα στον οποίο θα εγκαταστήσει τα απαραίτητα αρχεία του. Παρατηρούμε ότι το NetBeans χρειάζεται απαραίτητα το JDK για να ολοκληρώσει την εγκατάσταση του και αυτός είναι άλλωστε ο λόγος που η δεύτερη γραμμή παρουσιάζει το directory μέσα στο οποίο θα μπορέσει να το βρει. Αυτό σημαίνει ότι αν πάμε στην ιστοσελίδα του NetBeans (http://www.netbeans.org) και κατεβάσουμε το Netbeans, για να μπορέσουμε να το εγκαταστήσουμε θα πρέπει να υπάρχει ήδη μια εγκατάσταση του JDK στον υπολογιστή μας. Το NetBeans δεν προσφέρεται σαν πακέτο από το δικό του site για αυτό άλλωστε και επιλέξαμε την ORACLE για να κάνουμε μια ολοκληρωμένη και χωρίς προαπαιτήσεις εγκατάσταση του προγράμματος. Στην δική μας περίπτωση και τα δύο προγράμματα βρίσκονται στο ίδιο πακέτο εγκατάστασης και η σειρά έχει αυτόματα οριστεί. Δεχόμαστε λοιπόν και αυτή την επιλογή και συνεχίζουμε με το κουμπί ***NEXT***.

Αμέσως μετά θα εμφανιστεί μια σύντομη περίληψη της τοποθεσίας των φακέλων στους οποίους θα εγκατασταθούν τα αρχεία των δύο προγραμμάτων. Απλά επιλέγουμε ***INSTALL*** για να ολοκληρωθεί η εγκατάσταση.

Στο τελευταίο παράθυρο εγκατάστασης του NetBeans, απλά ενημερωνόμαστε για το επιτυχής τέλος της διαδικασίας και κατά πόσο θέλουμε να συνεισφέρουμε στο NetBeans project αφήνοντας την εταιρεία να μαζεύει κάποια στατιστικά στοιχεία τα οποία θα αξιοποιήσει για την περαιτέρω βελτίωση του προγράμματος.

Αφού αποφασίσουμε ποιες επιλογές θέλουμε να υποστηρίξουμε (όλες είναι προαιρετικές), απλά συνεχίζουμε με το κουμπί ***FINISH*** για να επισημοποιήσουμε την ολοκλήρωση της

εγκατάστασης. Στην επιφάνεια εργασίας του υπολογιστή μας, θα πρέπει τώρα να εμφανίζεται ένα εικονίδιο σε σχήμα κύβου με το όνομα NetBeans.



Εικόνα Α1.α : Εικονίδιο του εργαλείου NetBeans.

Κάνοντας διπλό κλικ επάνω στο εικονίδιο του NetBeans, θα βρεθούμε στο περιβάλλον εργασίας του.

Παράρτημα Α2 : Εφαρμογή του NetBeans

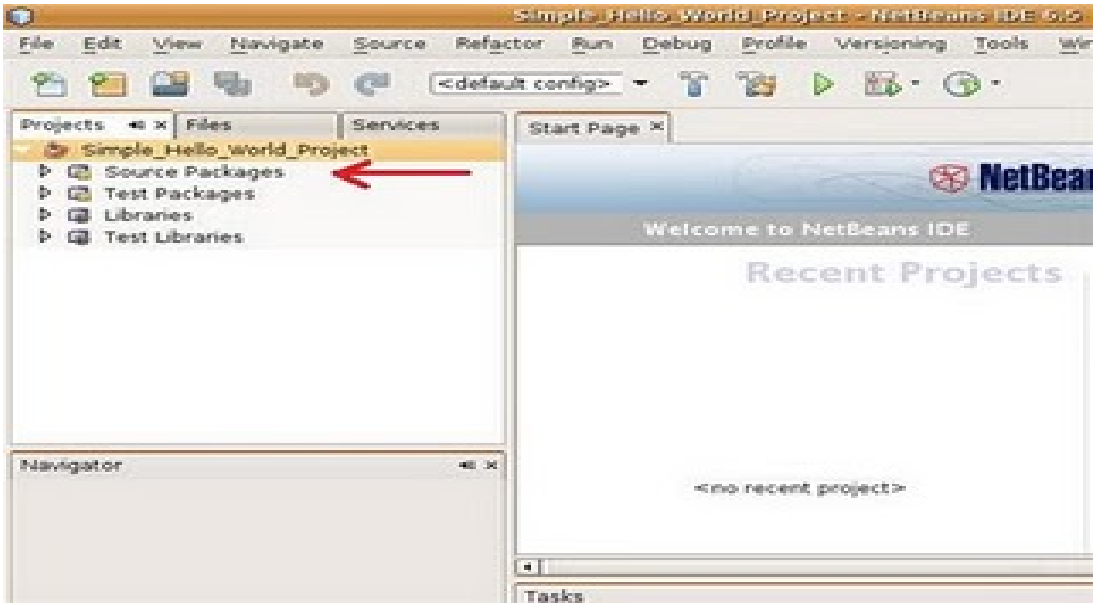
Το NetBeans δουλεύει κάτω από την έννοια του project. Δηλαδή πριν ξεκινήσουμε την δημιουργία του προγράμματος μας, πρέπει να ορίσουμε ένα project μέσα στο οποίο θα περιέχονται όλα τα αρχεία που συσχετίζονται με την δημιουργία της εφαρμογής. Έτσι, κατά την διαδικασία του compilation, όσα αρχεία συνδέονται με το συγκεκριμένο project θα συμπεριληφθούν στο τελικό αρχείο που θα δημιουργηθεί και θα έχει την κατάληξη jar. Το jar είναι το εκτελέσιμο τελικό πρόγραμμα. Η java δεν προσφέρει καταλήξεις .exe στα προγράμματα της. Αυτό όμως δεν σημαίνει ότι δεν έχουν και την ικανότητα αυτόνομης εκτέλεσης.

Όσο αφορά όμως το πρακτικό κομμάτι αυτής της διαδικασίας, με την δημιουργία ενός project, έχουμε την αντίστοιχη δημιουργία ενός φακέλου με το ίδιο όνομα στο σκληρό μας δίσκο. Κάνουμε λοιπόν κλικ στην επιλογή **File** από το κύριο μενού, επιλέγουμε **New Project** και στο καινούργιο παράθυρο που εμφανίζεται επιλέγουμε το φάκελο **Java** από τα "**Categories"**, και **Java Application** από τα "**Projects"**, και μετά συνεχίζουμε με το κουμπί **next**.

Στο επόμενο παράθυρο που εμφανίζεται έχουμε να κάνουμε κάποιες τελευταίες επιλογές πριν δημιουργηθεί ο χώρος (Project) μέσα στον οποίο θα περικλείεται το πρόγραμμα μας. Στο **Project Name** γράφουμε το όνομα που επιθυμούμε να δώσουμε στο Project μας. Δεν υπάρχουν περιορισμοί για το πως θα το ονομάσουμε, όμως αν δώσουμε ονόματα όπως Application_1, Application_2 ή Project_1 και Project_2 αργότερα δεν θα θυμόμαστε τι ακριβώς κάνει κάθε μια από τις εφαρμογές μας. Το project name να αντιπροσωπεύει ξεκάθαρα το σκοπό για τον οποίο δημιουργήθηκε η εφαρμογή. Από το **Project Location** έχουμε την δυνατότητα να αλλάξουμε την προκαθορισμένη τοποθεσία στο σκληρό δίσκο στην οποία θα αποθηκευτεί το project με όλα τα αρχεία του. Ενώ από το **Project Folder** ενημερωνόμαστε ότι ο φάκελος στον οποίο μπορούμε να αναφερόμαστε για να αντιγράψουμε, ή να αποθηκεύσουμε τα αρχεία μας θα έχει το ίδιο όνομα όπως και το project.

Πριν συνεχίσουμε με το κουμπί **FINISH**, βεβαιωνόμαστε ότι οι επιλογές "**Use Dedicated Folder for Storing Libraries**", "**Create Main Class**", και "**Set as Main Project**" δεν είναι επιλεγμένες, όπως δείχνει η παρακάτω εικόνα. Ο λόγος που αυτές τις τρεις επιλογές δεν τις θέλουμε ενεργοποιημένες είναι γιατί στα πρώτα βήματα του προγραμματισμού, δεν θέλουμε την βοήθεια από προκαθορισμένα templates.

Όταν δημιουργηθεί το Project με το όνομα "Simple_Hello_World_Project", θα εμφανιστούν τέσσερις υποκατηγορίες εκ τις οποίες η μία ονομάζεται Source Packages. Μέσα σε αυτή την κατηγορία θα δημιουργήσουμε τα πακέτα μας στα οποία θα περικλείονται τα αρχεία της εφαρμογής μας. Τα πακέτα χρησιμεύουν για την κατηγοριοποίηση και ομαδοποίηση αρχείων που εξυπηρετούν συγκεκριμένο σκοπό μέσα στην εφαρμογή μας. Για παράδειγμα, μέσα σε ένα πακέτο μπορούμε να έχουμε τις κλάσεις (αρχεία) που αφορούν την ένωση του προγράμματος μας με μια βάση δεδομένων, ενώ σε ένα δεύτερο πακέτο να έχουμε τις κλάσεις που αφορούν το γραφικό μέρος της εφαρμογής μας. Πρακτικά, στο σκληρό μας δίσκο η δημιουργία ενός πακέτου αντιστοιχεί με την δημιουργία ενός φακέλου με το ίδιο όνομα. Στην παρακάτω εικόνα βλέπουμε πως θα είναι το περιβάλλον του NetBeans μετά την δημιουργία ενός project.



Εικόνα Α2.α : Περιβάλλον του NetBeans μετά την δημιουργία ενός project.

Αναφέραμε πιο πάνω, ότι ένα πρόγραμμα, θα πρέπει να βρίσκεται μέσα σε ένα πακέτο για να μπορεί εκτελεστεί. Επίσης ένα πακέτο είναι δυνατόν να περιέχει περισσότερα από ένα αρχείο και η δημιουργία ενός πακέτου ισοβαθμεί με την δημιουργία φακέλου μέσα στον ήδη υπάρχων φάκελο του project. Οπότε εάν θέλουμε να έχουμε πάντα ένα αντίγραφο του προγράμματος μας, απλά αντιγράφουμε το φάκελο του project σε κάποιο αποθηκευτικό μέσο (flash drive ή hard drive).

Είμαστε αναγκασμένοι να δημιουργήσουμε τουλάχιστον ένα πακέτο για να μπορούμε να τρέξουμε την εφαρμογή μας. Για να δημιουργήσουμε ένα πακέτο, επιλέγουμε δεξί κλικ στο εικονίδιο **Source Packages**, επιλέγουμε **New**, και μετά **Java Package**.

Στο νέο παράθυρο που ανοίγει, μας ζητείται να δώσουμε ένα όνομα στο πακέτο που πρόκειται να δημιουργήσουμε. Αν και για το πως θα ονομάσουμε τα πακέτα δεν υπάρχει κανένας περιορισμός, ας τα ονομάζουμε με το ίδιο όνομα όπως την java εκτελέσιμη κλάση που περιέχουν.

Ονομάζουμε το πακέτο μας "**Hello_World**" (με underscore ανάμεσα στις δύο λέξεις) και αμέσως μετά συνεχίζουμε με το κουμπί **FINISH**.

Επιλέγοντας τώρα δεξί κλικ το πακέτο Hello_World, επιλέγουμε **New**, και μετά **Java class**. Στην επιλογή **Class Name** γράφουμε το όνομα του προγράμματος το οποίο είναι Hello_World (χωρίς κενό ανάμεσα στις δύο λέξεις). Τα ονόματα ανάμεσα στην κλάση και στο πακέτο δεν είναι απαραίτητο να είναι ακριβώς τα ίδια. Όμως υπάρχει μια μεγάλη διαφορά – Ενώ το NetBeans αδιαφορεί για το πως θα γράψουμε το όνομα του πακέτου, δίνει όμως μεγάλη βαρύτητα στον τρόπο εγγραφής του ονόματος της κλάσης.

Η Java είναι "ευαίσθητη" με τα πεζά και κεφαλαία γράμματα. Η λέξη Hello_World δεν είναι η ίδια με βάση τους κανόνες της java με την λέξη hello_world. Το όνομα που θα δώσουμε στην κλάση πρέπει να συμφωνεί στην γραφή με το όνομα το οποίο θα πρέπει να ξεκινήσουμε τον κώδικα μας. Όπως γράψουμε το όνομα του προγράμματος στο παράθυρο, έτσι πρέπει να το γράψουμε και μέσα στο κώδικα μας. Ο λόγος για αυτό τον αυστηρό κανόνα είναι απλός – το αρχείο θα αποθηκευτεί με το ίδιο όνομα που θα δώσουμε στην κλάση. Αφήνουμε τις υπόλοιπες επιλογές όπως είναι και συνεχίζουμε με το **FINISH**.

Κάποιες βασικές γραμμές κώδικα που εμφανίζονται στα δεξιά της οθόνης μας, είναι απαραίτητες για οποιοδήποτε πρόγραμμα java και αν γράψουμε. Απλά το NetBeans θέλει να βοηθήσει τον προγραμματιστή ετοιμάζοντας το σκελετό του προγράμματος εξοικονομώντας έτσι χρόνο και μειώνοντας την πιθανότητα λάθους – το λάθος εδώ μπορεί να γίνει όταν γράψουμε το όνομα της κλάσης μέσα στο κώδικα μας με διαφορετικό όνομα από αυτό που δώσαμε κατά την δημιουργία της. Αυτή η πρωτοβουλία που παίρνει το περιβάλλον του Netbeans μας προφυλάσσει από πολλά προβλήματα που μπορεί να αντιμετωπίσουμε αργότερα. Δεν είναι ανάγκη να διαγράψουμε καμία από αυτές τις γραμμές κώδικα. Εμείς απλά τώρα μέσα στην κλάση Hello_World θα συμπληρώσουμε τις γραμμές κώδικα που θα κάνουν αυτή την κλάση εκτελέσιμη (γιατί από μόνη της δεν είναι) και επίσης να ορίσουμε μια απλή μέθοδο η οποία θα μας δείξει ένα σύντομο μήνυμα στην οθόνη μας.

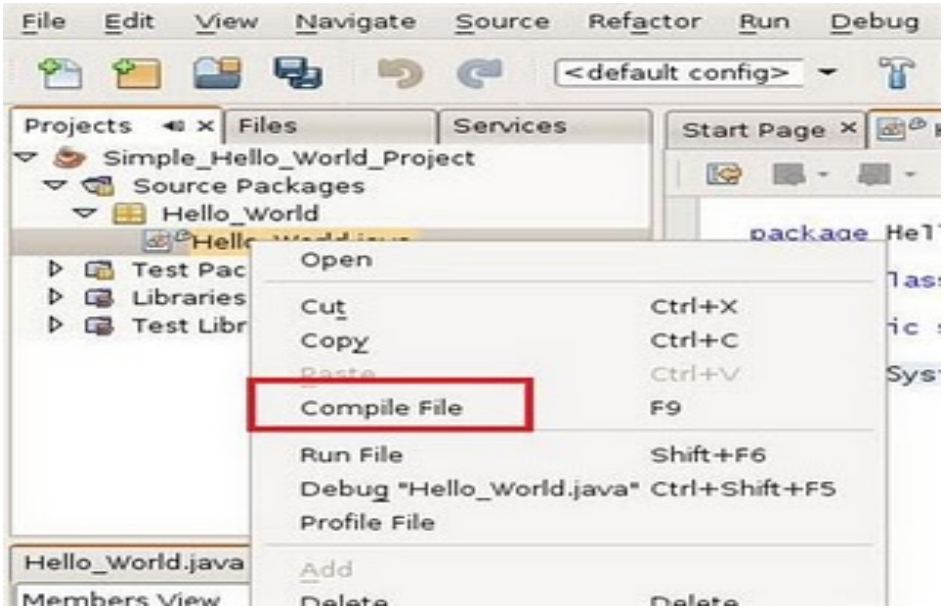
```
public class Hello_World

{

public static void main(String [ ] args)

{

System.out.println("Hello, World");

}

}
```

Το ολοκληρωμένο πρόγραμμα παρουσιάζεται στην παρακάτω εικόνα.

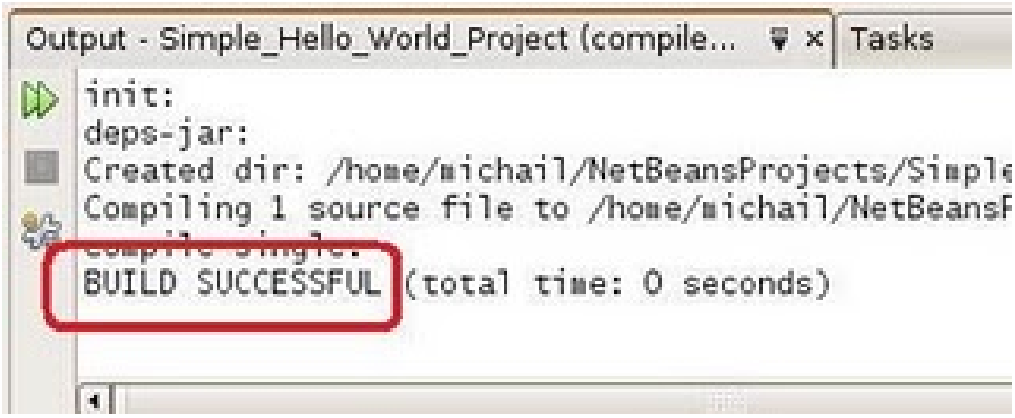
Εικόνα Α2.β : Παράθυρο σύνταξης κώδικα.

Αν κοιτάξουμε στο αριστερό μέρος της οθόνης μας, κάτω από το πακέτο Hello_World, υπάρχει ένα αρχείο Hello_World.java το οποίο συνοδεύεται από ένα μικρό εικονίδιο που μοιάζει σαν μηχανικό κλειδί. Αυτό σημαίνει ότι το πρόγραμμα μας γράφτηκε, όμως δεν έχει ελεγχθεί επίσημα για λάθη από τον Compiler της Java. Πριν όμως προχωρήσουμε σε αυτή την διαδικασία πάμε στο κύριο μενού Window και από την επιλογή Output να επιλέξουμε πάλι Output. Τώρα στο κάτω μέρος της οθόνη μας έχουμε ένα καινούργιο πλαίσιο ανοιχτό μέσα από το οποίο μπορούμε να βλέπουμε τα αποτελέσματα των προγραμμάτων μας ή ακόμα τυχών λάθη για τα οποία ο compiler έχει διαμαρτυρηθεί.

Επιλέγοντας λοιπόν με δεξί κλικ το java αρχείο και από τις επιλογές που εμφανίζονται, επιλέγουμε το **Compile File**. Αν δεν υπάρχουν συντακτικά λάθη στο πρόγραμμα μας, τότε στο πλαίσιο Output θα παρατηρήσουμε ότι η διαδικασία ελέγχου ήταν επιτυχής (BUILD SUCCESSFUL) και το πρόγραμμα μας είναι άριστα γραμμένο.



Εικόνα Α2.γ : Επιλογή Compile του προγράμματος.



Εικόνα Α2.δ : Παράθυρο αποτελέσματος ελέγχου.

Το γεγονός ότι το πρόγραμμά μας έγινε compile επίσης συμβολίζει την δημιουργία και του αρχείου .class. Επειδή όμως το NetBeans θέλει να ξεχωρίσει τα δύο αρχεία, κρατάει τα αρχεία .java στο φάκελο src ενώ δημιουργεί τα αρχεία .class στο φάκελο build.

Εικόνα Α2.ε : Φάκελος αποθηκευμένων αρχείων.

Αν γυρίσουμε πίσω στην κύρια οθόνη του NetBeans, θα προσέξουμε ότι το εικονίδιο Hello_World.java στα αριστερά της οθόνης μας, κάτω από το πακέτο Hello_World, έχει ένα καινούργιο σύμβολο που μοιάζει σαν το play κουμπί που έχουμε συνηθίσει στις ηλεκτρικές συσκευές. Αυτό σημαίνει ότι το αρχείο java είναι έτοιμο για εκτέλεση. Αυτό το σύμβολο εμφανίζεται μόνο στις κλάσεις που περιέχουν την μέθοδο main - την μέθοδο που μετατρέπει την κλάση σε εκτελέσιμη. Ο τρόπος που θα πραγματοποιήσουμε αυτή την ενέργεια είναι να κάνουμε δεξί κλικ για ακόμα μια φορά επάνω στο αρχείο, αλλά αυτή τη φορά θα επιλέξουμε Run File. Το αποτέλεσμα της εκτέλεσης του προγράμματος (“Hello, World”) θα το δούμε στο παράθυρο output.

Παράρτημα Α3 : Εγκατάσταση του NetBeans σε Ubuntu

Κατεβάσουμε από την Sun Microsystems το JDK πακέτο το οποίο παρέχει τον java compiler μαζί με το γραφικό περιβάλλον του NetBeans. Το πλήρης πακέτο θα το βρούμε στην ακόλουθη ηλεκτρονική διεύθυνση:

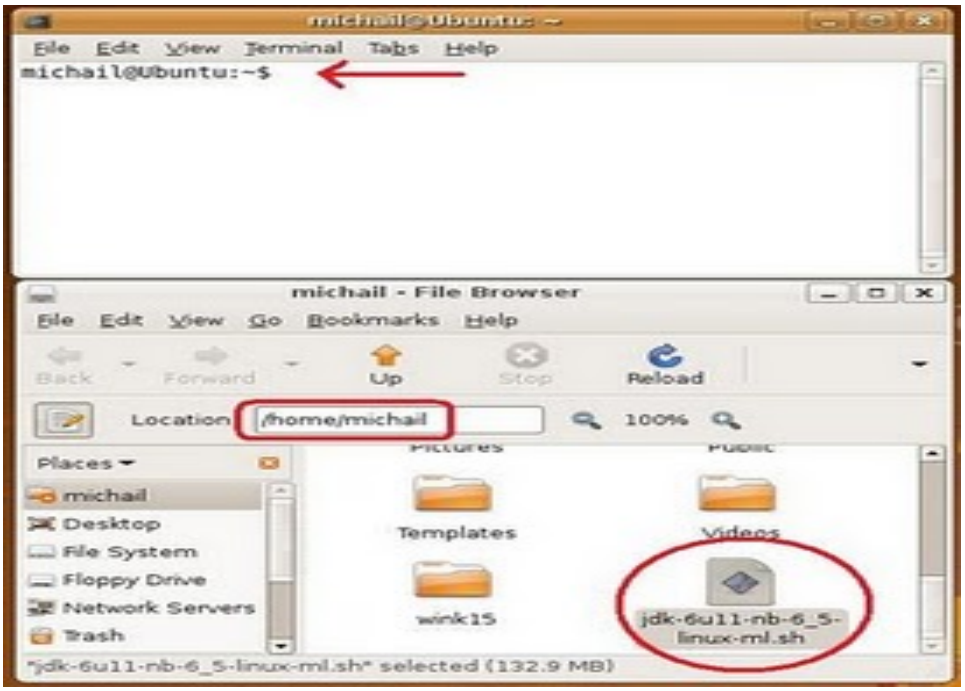
http://java.sun.com/javase/downloads/netbeans.html

Πατώντας την επιλογή **DOWNLOAD** θα μεταφερθούμε σε μια καινούργια σελίδα που μας ζητάει με απλό τρόπο να ορίσουμε την πλατφόρμα στην οποία πρόκειται να εγκαταστήσουμε το NetBeans και σε ποια γλώσσα επιθυμούμε να παρουσιάζονται τα μενού του συγκεκριμένου προγράμματος. Πριν επιλέξουμε το κουμπί **Continue**, θα πρέπει να συμφωνήσουμε με τους όρους της άδειας για το συγκεκριμένο πρόγραμμα επιλέγοντας μέσα στο κενό κουτάκι δίπλα από το "I agree....". Έχουμε επιλέξει για πλατφόρμα ανάπτυξης προγραμμάτων το λειτουργικό LINUX και σαν γλώσσα τα αγγλικά (περιέχεται μέσα στο Multi-language πακέτο).

Στην επόμενη ιστοσελίδα που μας εμφανίζεται, απλά επιλέγουμε το μπλε link του πακέτου jdk και όταν εμφανιστεί η επιλογή αποθήκευσης του browser, επιλέγουμε την τοποθεσία στο σκληρό δίσκο στο οποίο θα αποθηκεύσουμε το αρχείο.

Για τους χρήστες που δεν γνωρίζουν την διαδικασία εγκατάστασης ενός προγράμματος σε περιβάλλον LINUX, μπορούν να δουν τα απαραίτητα βήματα από τις πιο κάτω εικόνες.

Σαν πρώτο βήμα, τοποθετούμε το αρχείο που κατεβάσαμε στο home folder (δηλαδή στο φάκελο με το όνομα μας). Έτσι θα είναι ευκολότερο για μας να το βρούμε. Σαν δεύτερο βήμα ανοίγουμε το πρόγραμμα Terminal από το Accessories μενού. Η πιο κάτω εικόνα μας δείχνει πως θα πρέπει να μοιάζει η οθόνη μας με τα δύο προγράμματα ενεργοποιημένα.

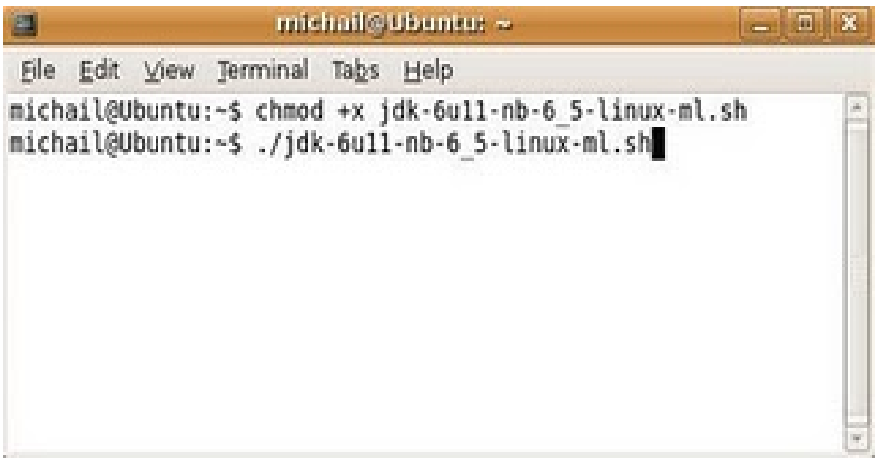


Εικόνα Α3.α : Παράθυρα Terminal και Accessories μενού.

Για να μπορέσουμε να δώσουμε την εντολή εγκατάστασης, θα πρέπει πρώτα να αλλάξουμε τα δικαιώματα κάτω από τα οποία ορίζεται η πρόσβαση μας στο αρχείο.

Εικόνα Α3.β : Διαδικασία αλλαγής δικαιωμάτων.

Τέλος μπορούμε να καλέσουμε την εντολή εγκατάστασης.



Εικόνα Α3.γ: Παράθυρο Terminal του Ubuntu.

Αμέσως μετά από την εκτέλεση της εντολής θα εμφανιστεί το πρώτο παράθυρο που μας ενημερώνει ότι πρόκειται να γίνει εγκατάσταση του JDK (δηλαδή του compiler της java), αλλά και του περιβάλλοντος Netbeans. Για να μπορέσουμε να συνεχίσουμε την εγκατάσταση του προγράμματος, δεχόμαστε το μήνυμα με το κουμπί NEXT.

Στο αμέσως επόμενο παράθυρο, πρέπει να συμφωνήσουμε με τους όρους της άδειας κάτω από τους οποίους λειτουργεί το συγκεκριμένο πρόγραμμα. Τσεκάρουμε το κουτί διπλά από το "I accept the ..."και επιλέγουμε NEXT.

Η διαδικασία εγκατάστασης μας ενημερώνει ότι θα πρέπει να δημιουργήσει ένα φάκελο στο home folder του λογαριασμού μας για να μπορέσει να πραγματοποιήσει το πρώτο βήμα που είναι η εγκατάσταση της java. Επιλέγουμε το κουμπί NEXT για να προχωρήσουμε στο επόμενο βήμα.

Μετά από την java, είναι η σειρά του Netbeans να δημιουργήσει ένα φάκελο στο σύστημα μας μέσα στον οποίο θα εγκατασταθεί. Παρατηρούμε ότι το Netbeans χρειάζεται απαραιτήτως το JDK για να ολοκληρώσει την εγκατάστασή του. Αυτό σημαίνει ότι αν πάμε στο http://www.netbeans.org και προμηθευτούμε μόνοι μας το Netbeans, για να μπορέσουμε να το εγκαταστήσουμε θα πρέπει να υπάρχει ήδη μια εγκατάστασή της java JDK στον υπολογιστή μας. Χωρίς αυτή την συνθήκη, και επειδή το Netbeans θα το αναζητήσει και δεν θα το βρει, θα ακυρωθεί η διαδικασία εγκατάστασης. Στην δική μας περίπτωση όμως και τα δύο προγράμματα βρίσκονται στο ίδιο πακέτο και η σειρά έχει αυτόματα οριστεί. Δεχόμαστε λοιπόν και αυτή την επιλογή και συνεχίζουμε με NEXT.

Μια σύντομη περίληψη της τοποθεσίας των φακέλων που θα γίνουν εγκατάστασή το JDK και το Netbeans εμφανίζεται στο επόμενο παράθυρο. Απλά επιλέγουμε INSTALL για να αρχίσει η διαδικασία εγκατάστασης του πακέτου.

Στο τελευταίο παράθυρο της διαδικασίας αυτής απλά ενημερωνόμαστε για το επιτυχής τέλος της εγκατάστασης και κατά πόσο θέλουμε να συνεισφέρουμε στο Netbeans project αφήνοντας την εταιρεία να μαζεύει κάποια στατιστικά στοιχεία τα οποία θα αξιοποιήσει για την βελτίωση του προγράμματος.

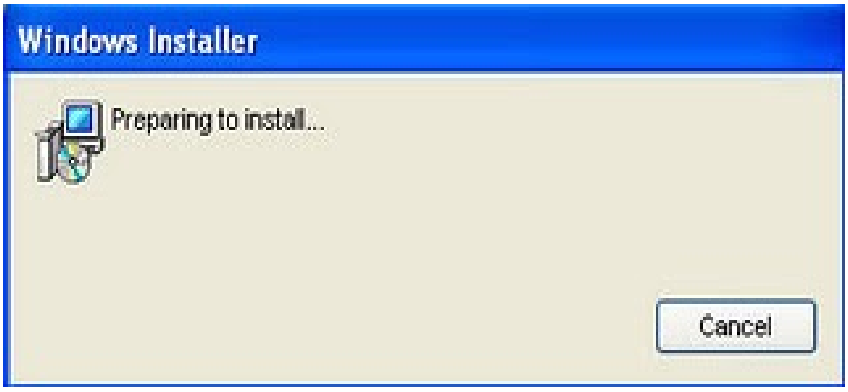
Αφού αποφασίσουμε, απλά επιλέγουμε το κουμπί FINISH για να επισημοποιήσουμε την ολοκλήρωση εγκατάστασης. Στην επιφάνεια εργασίας του υπολογιστή μας, θα πρέπει τώρα να εμφανίζεται ένα εικονίδιο με το όνομα του Netbeans.

Εδώ ολοκληρώνεται η διαδικασία εγκατάστασης του JDK και του περιβάλλοντος ανάπτυξης Java εφαρμογών Netbeans.

Παράρτημα Α4 : Εγκατάσταση της MySQL σε Windows

Αφού αποσυμπιέσουμε το ZIP αρχείο. Εκτελούμε το πρόγραμμα εγκατάστασης, κάνοντας διπλό κλικ επάνω στο Setup (εκτελούμενο αρχείο) για να αρχίσουμε την διαδικασία εγκατάστασης. Η MySQL έχει προγραμματιστεί να τρέχει σταθερά και χωρίς ιδιαίτερα προβλήματα απόδοσης σε πλατφόρμα Windows XP και Windows Server. Αν τυχόν υπάρχει κάποια προηγούμενη εγκατάσταση της MySQL στον υπολογιστή μας, πρέπει να την απεγκαταστήσουμε πριν συνεχίσουμε με τα παρακάτω βήματα. Μόνο έτσι θα μπορέσουμε να εξαλείψουμε εξ αρχής οποιαδήποτε προβλήματα που μπορεί να υπάρξουν αργότερα.

Το πρώτο παράθυρο που θα εμφανιστεί στην διαδικασία της εγκατάστασης θα είναι το παρακάτω:



Εικόνα Α4.α : Προετοιμασία εγκατάστασης.

Μετά την εμφάνιση της πρώτης εικόνας, σε λίγα δευτερόλεπτα θα εμφανιστεί το μήνυμα καλωσορίσματος της εγκατάστασης του MySQL. Σε αυτό το παράθυρο επιλέγουμε το κουμπί **NEXT**.

Έπειτα έχουμε να επιλέξουμε ανάμεσα σε τρεις επιλογές εγκατάστασης. Η πρώτη επιλογή (Typical) θα εγκαταστήσει την κύρια βάση μας με πολλά έξτρα command-line εργαλεία (δηλαδή έξτρα δυνατότητες που μπορούμε να αξιοποιήσουμε μέσα από εντολές μόνο). Αυτή είναι και η πιο συνηθισμένη μορφή εγκατάστασης για τις περισσότερες εφαρμογές για τις οποίες χρειαζόμαστε υποστήριξη από μια βάση δεδομένων.

Η δεύτερη επιλογή (Complete) θα προχωρήσει σε μια πλήρη εγκατάσταση του προγράμματος η οποία περιλαμβάνει μια πληθώρα από scripts και ένα ολοκληρωμένο documentation για να μπορούμε να ανατρέξουμε ανά πάσα στιγμή όταν χρειαστούμε κάποια βοήθεια με τις εντολές.

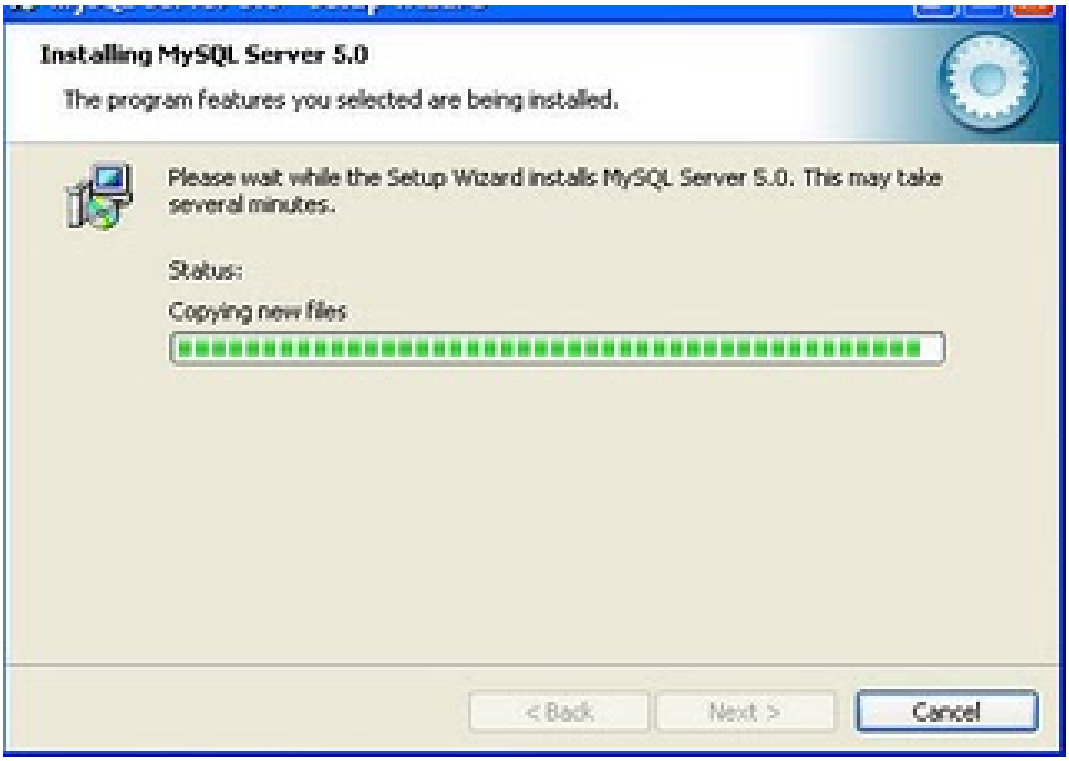
Η τρίτη και τελευταία επιλογή (Custom) αφήνει στην κρίση μας την επιλογή των διαφόρων εργαλείων και δυνατοτήτων που θέλουμε να εγκαταστήσουμε.

Εμείς θα επιλέξουμε την δεύτερη επιλογή η οποία περιλαμβάνει την ολοκληρωμένη εγκατάσταση και θα συνεχίσουμε με το κουμπί **NEXT**.

Στο επόμενο παράθυρο το πρόγραμμα μας ειδοποιεί ότι είναι έτοιμο να ξεκινήσει την διαδικασία εγκατάστασης. Στην περιγραφή που εμφανίζεται στο παραθυρικό περιβάλλον μπορούμε να δούμε μια συνοπτική εικόνα των επιλογών μας και την τοποθεσία εγκατάστασης του

προγράμματος στον σκληρό μας δίσκο. Συνιστάται να αφήσουμε αυτόν τον κατάλογο, όπως είναι η προεπιλογή. Αν το αλλάξουμε, θα πρέπει να ρυθμίσουμε τη MySQL αργότερα να επιλέγει την πορεία μας. Απλά επιλέγουμε NEXT και συνεχίζουμε.

Το επόμενο παράθυρο που θα εμφανιστεί θα είναι η διαδικασία εγκατάστασης του προγράμματος.



Εικόνα Α4.β : Παράθυρο διαδικασίας εγκατάστασης του προγράμματος.

Μετά την ολοκλήρωση του πρώτου μέρους της διαδικασίας εγκατάστασης. θα ενημερωθούμε για τα οφέλη που μπορούμε να αποκτήσουμε κάνοντας μια συνδρομή στο MySQL Enterprise. Για να ενημερωθούμε σχετικά με την συνδρομή επιλέγουμε το κουμπί **MORE**, ειδάλλως συνεχίζουμε με την διαδικασία ολοκλήρωσης της εγκατάστασης επιλέγοντας το κουμπί **NEXT**.

Μετά τις δυο ανακοινώσεις και τα οφέλη της συνδρομής, θα μας παρουσιαστεί ένα καινούργιο παράθυρο που μας ενημερώνει ότι η εγκατάσταση ολοκληρώθηκε και μπορούμε τώρα να ορίσουμε το port στο οποίο θα "ακούει" το πρόγραμμα όπως το password για τον administrator. Αφήνοντας λοιπόν την επιλογή όπως είναι συνεχίζουμε με το κουμπί **FINISH.** Έπειτα θα μεταφερθούμε στο επόμενο περιβάλλον της MySQL από το οποίο μπορούμε να ενεργοποιήσουμε διάφορες επιλογές για την συγκεκριμένη βάση που εγκαταστήσαμε.

Αφού δεχτούμε ένα ακόμα μήνυμα που μας ειδοποιεί ότι τώρα έχουμε την ευκαιρία να ορίσουμε τις δικές μας προτιμήσεις στο πως θα "τρέχει" η βάση, απλά επιλέγουμε το κουμπί **NEXT**.

Στο επόμενο παράθυρο θα βρεθούμε πάλι μπροστά σε δυο επιλογές. Η πρώτη (Detailed Configuration) δίνει την ευελιξία στον χρήστη να διαλέξει τις καταλληλότερες ρυθμίσεις της βάσης, ενώ η δεύτερη επιλογή (Standard Configuration) απλά δημιουργεί ένα περιβάλλον κατάλληλο για τις περισσότερες απαιτήσεις του εργασιακού περιβάλλοντος. Εμείς επιλέγουμε την πρώτη επιλογή.

Οι επόμενες τρεις επιλογές που θα εμφανιστούν είναι αρκετά επεξηγηματικές από μόνες τους. Αν διαλέξουμε την πρώτη επιλογή (Developer Machine), η MySQL βάση θα χρησιμοποιήσει όσο το δυνατόν λιγότερη μνήμη από τον υπολογιστή μας. Η επιλογή αυτή είναι κατάλληλη για προγραμματιστές που απλά χρειάζονται μια βάση δεδομένων για να δοκιμάσουν την λειτουργικότητα κάποιας εφαρμογής που αναπτύσσουν.

Η δεύτερη επιλογή (Server Machine) θα χειριστεί την MySQL σαν server που όμως υποθετικά δεν θα είναι ο μόνος που τρέχει στον συγκεκριμένο υπολογιστή. Συνήθως αυτή η επιλογή χρησιμοποιείται όταν στον ίδιο υπολογιστή υπάρχουν και άλλοι servers που έχουν εγκατασταθεί όπως π.χ. ένας Web Server (είναι υπεύθυνος για την διαχείριση και την παρουσίαση ιστοσελίδων). Ο λόγος που ίσως εγκαταστήσαμε την MySQL στον ίδιο υπολογιστή είναι γιατί πιθανόν υπάρχει μια Web εφαρμογή (π.χ. ένα ηλεκτρονικό κατάστημα e-commerce) το οποίο χρειάζεται μια βάση δεδομένων για την αποθήκευση στοιχείων όπως προϊόντα, πωλήσεις, και πελατολόγιο. Οπότε γίνεται μια καλύτερη κατανομή της μνήμης και του CPU όσο αφορά την MySQL σε σχέση με τους υπόλοιπους servers.

Η τρίτη επιλογή (Dedicated MySQL Server Machine) δίνει την ελευθερία στην MySQL να χρησιμοποιήσει όσους πόρους χρειάζεται από τον υπολογιστή για να ανταποκριθεί όσο το δυνατόν πιο γρήγορα στις απαιτήσεις του χρήστη. Κυρίως θα απορροφήσει ένα μεγάλο κομμάτι της μνήμης. Αυτή η επιλογή είναι κατάλληλη όταν ο υπολογιστής θα τρέχει αποκλειστικά και μόνο την MySQL και κανένα άλλο server πρόγραμμα.

Εμείς θα επιλέξουμε την δεύτερη επιλογή για τον απλό λόγω ότι πολλοί χρήστες του Internet πειραματίζονται με την εγκατάσταση Web servers όπως τον Apache και με προγράμματα ανοιχτού κώδικα (Open Source) που απαιτούν εγκατάσταση της MySQL για να προσφέρουν τις υπηρεσίες τους, όπως π.χ. την δημιουργία ενός forum. Επιλέγουμε λοιπόν το κουμπί **NEXT** με την δεύτερη επιλογή τσεκαρισμένη και προχωράμε στο επόμενο βήμα.

Αμέσως μετά θα βρεθούμε πάλι μπροστά σε τρεις επιλογές. Οι επιλογές επιδρούν άμεσα στην ενεργοποίηση/απενεργοποίηση των δυο μηχανών αποθήκευσης δεδομένων της MySQL οι οποίες είναι η InnoDB και MyISAM.

Στην πρώτη επιλογή μας, και οι δυο αυτές μηχανές είναι ενεργές και οι πόροι όσο και το φορτίο εργασίας μοιράζεται μεταξύ τους. Εάν τώρα θέλουμε και τις δυο μηχανές ενεργοποιημένες αλλά όμως μια μόνο να κάνει την δουλειά τότε διαλέγουμε την δεύτερη επιλογή. Ενώ στην τρίτη η InnoDB είναι εντελώς απενεργοποιημένη.

Επειδή στους περισσότερους αρέσει να ενεργοποιούν κάθε δυνατό εργαλείο ενός προγράμματος, θα επιλέξουμε την πρώτη επιλογή. Έτσι η βάση μας θα είναι ικανή να ανταποκριθεί σε οποιοδήποτε πρόγραμμα, εφαρμογή η παράδειγμα αναλύσουμε.

Με την επιλογή μας δώσαμε εντολή να εγκαταστήσουμε και τις δυο μηχανές διαχείρισης στοιχείων που προσφέρει η MySQL. Σαν αποτέλεσμα αυτής της επιλογής μας, η MySQL μας δίνει την δυνατότητα να μπορέσουμε να μεταφέρουμε το InnoDB σε διαφορετικό σκληρό δίσκο για ακόμα μεγαλύτερη απόδοση. Εμείς δεν χρειάζεται να αλλάξουμε το μονοπάτι εγκατάστασης του InnoDB, οπότε απλά επιλέγουμε το κουμπί **NEXT**.

Για να μην φτάσει ο server στα όρια του τόσο ώστε να μην μπορεί να εξυπηρετήσει τα εισερχόμενα αιτήματα, στο επόμενο παράθυρο μπορούμε να καθορίσουμε τον αριθμό των ενεργών ενώσεων που θα κάνουν ταυτόχρονα κάποια διεργασία επάνω στην βάση. Αν και μπορούμε να θέσουμε μόνοι μας το όριο επιλέγοντας την τρίτη επιλογή (Manual Setting), εμείς όμως θα επιλέξουμε την πρώτη επιλογή (επιτρέπει μέχρι 20 ταυτόχρονες ενώσεις να κάνουν ερωτήματα στην βάση). Αφού λοιπόν επιλέξουμε την πρώτη επιλογή συνεχίζουμε με το κουμπί **NEXT**.

Στο επόμενο παράθυρο έχουμε την επιλογή να δεχτούμε την ενεργοποίηση του TCP/IP και της θύρας (port) την οποία «ακούει» η MySQL. Αν και μπορούμε να αλλάξουμε την θύρα σε ότι αριθμό είναι κατάλληλος για το δικό μας δίκτυο, απλά θα δεχτούμε το TCP/IP και το port 3306 για την δική μας βάση (αυτές είναι οι προεπιλεγμένες ρυθμίσεις που προτείνει πάντα η MySQL). Όσο αφορά την επιλογή Enable Strict Mode, αν και το επιθυμητό είναι να την ενεργοποιήσουμε έτσι ώστε ο MySQL να συμπεριφέρεται όπως μια κοινή βάση δεδομένων, εξ αρχής δεν είναι επιλεγμένη. Ενεργοποιούμε και αυτή την επιλογή και μετά συνεχίζουμε με το κουμπί **NEXT**.

Έπειτα μας ζητείται να διαλέξουμε ποιο σετ χαρακτήρων θέλουμε η βάση μας να υποστηρίζει. Εμείς επιλέγουμε την πρώτη επιλογή (Standard Character Set) και πατάμε το κουμπί **NEXT**.

Εδώ δεχόμαστε και τις δυο επιλογές - να δηλωθεί η MySQL σαν Windows Service αλλά και να αναβαθμιστεί το Windows PATH με το BIN Directory του MySQL - έτσι ώστε από οποιοδήποτε directory του command line και αν γράφουμε τις εντολές να είναι δυνατή η εκτέλεση τους και να αναγνωρίζονται από τα Windows.

Για την ασφάλεια του Server, μας ζητείται στο επόμενο βήμα να βάλουμε το password το οποίο προστατεύει τις ρυθμίσεις και την λειτουργικότητα της βάσης. Μόνο βάζοντας το σωστό κωδικό πρόσβασης, θα αποκτήσει κάποιος το δικαίωμα να προσθέσει επιπλέον βάσεις, να προσθέσει ή να αφαιρέσει πίνακες με δεδομένα, να τροποποιήσει αποθηκευμένες πληροφορίες και να κάνει γενικώς διαχείριση όλης της MySQL βάσης. Επιλέγουμε την πρώτη επιλογή, βάζουμε το password που μας ικανοποιεί και συνεχίζουμε με το κουμπί **NEXT**.

Στο επόμενο παράθυρο εμφανίζεται μια επιβεβαίωση των επιλογών μας και το περιβάλλον περιμένει να πατήσουμε το κουμπί EXECUTE για να ολοκληρωθεί και αυτό το τελευταίο στάδιο της εγκατάστασης.

Όταν πατήσουμε το κουμπί EXECUTE, θα δούμε ότι το παράθυρο παρουσιάζει το μήνυμα Processing configuration και απλά περιμένουμε να ολοκληρωθούν οι ρυθμίσεις που επιλέξαμε.

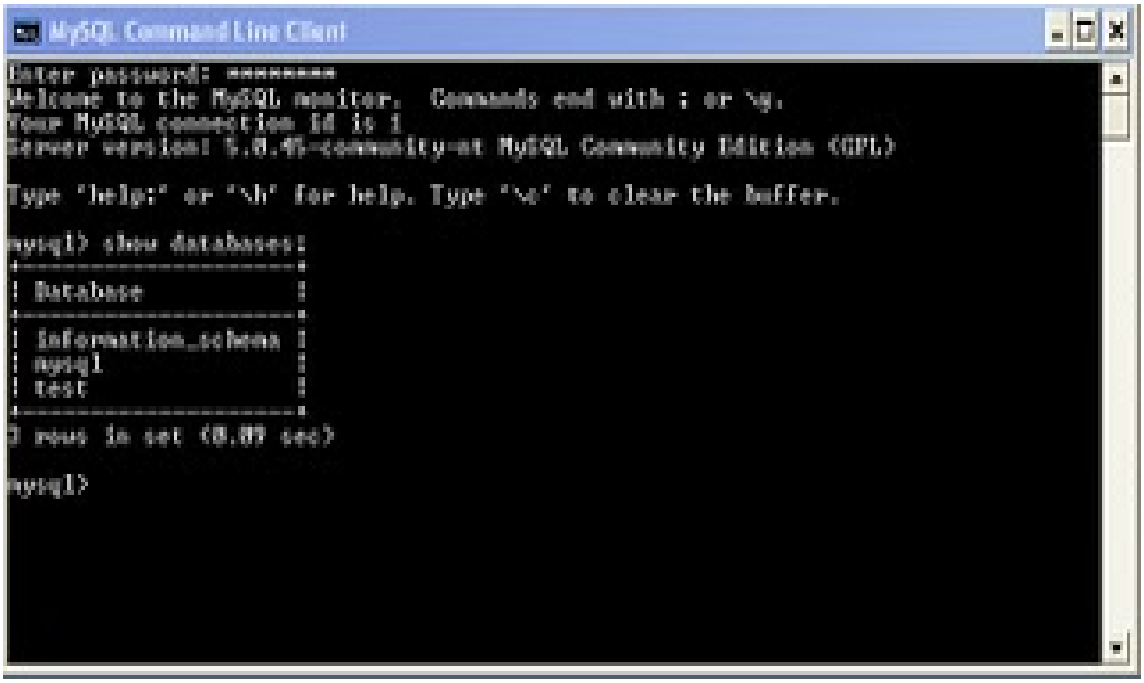
Εάν όλη η διαδικασία ολοκληρώθηκε με επιτυχία, τότε απλά θα λάβουμε ένα παράθυρο που επιβεβαιώνει αυτό ακριβώς το γεγονός.

Κλείνοντας το Wizard, θα επιθυμούσαμε να επιβεβαιώσουμε ότι η βάση λειτουργεί σωστά. Οπότε επιλέγουμε το κουμπί **START** (στα Windows), μετά **ALL PROGRAMS -->MYSQL-->MYSQL SERVER 5.0-->MYSQL COMMAND LINE CLIENT**. Λογικό είναι να μην έχουμε πρόσβαση επάνω στην βάση αφού ορίσαμε κωδικό πρόσβασης κατά την διάρκεια της εγκατάστασης. Γράφουμε λοιπόν το password και επιλέγουμε **ENTER**. Εάν έχουμε κάνει την εγκατάσταση σωστά (δεν ξεχάσαμε το password), τότε θα πρέπει η MySQL να μας παραχωρήσει την πρόσβαση στο σύστημα.

Για να δοκιμάσουμε να δούμε την επικοινωνία μας με την MySQL, απλά γράφουμε την εντολή

show databases;

Η εντολή αυτή θα μας εμφανίσει την παρακάτω εικόνα. Στα σίγουρα θα πρέπει να βλέπουμε στο αποτέλεσμα της εντολής τις databases με τα ονόματα mysql και test.



Εικόνα Α4.γ : Διαδικασία επικοινωνίας μας με την MySQL .

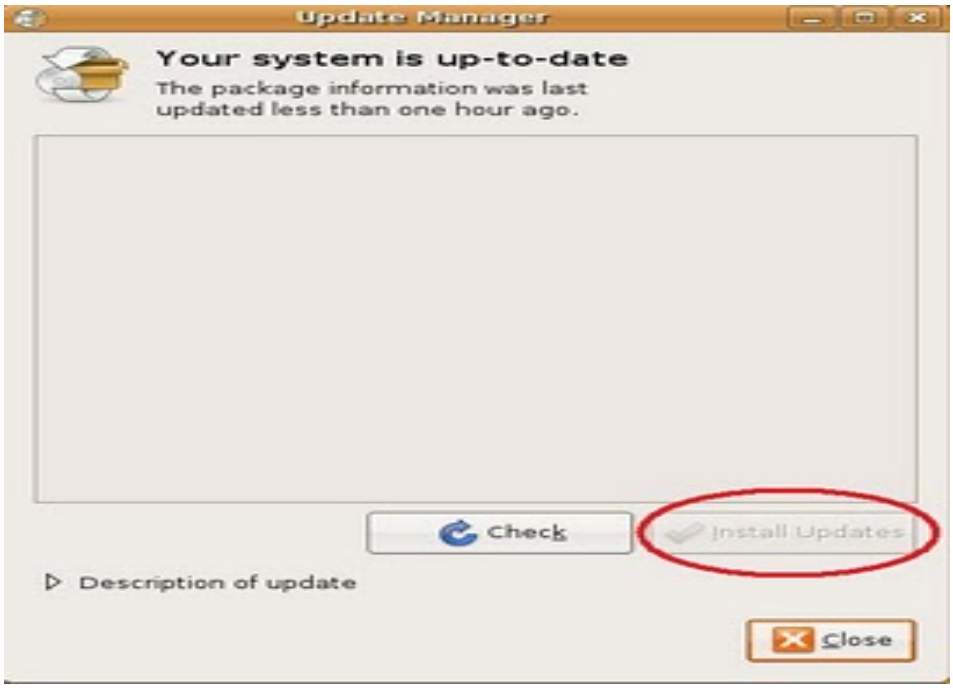
Αυτή η απλή διαδικασία είναι μια πρόχειρη απόδειξη ότι η MySQL έχει ενεργοποιηθεί χωρίς κανένα πρόβλημα, και τα Windows έχουν επιτυχημένα αναβαθμίσει το PATH.

Παράρτημα Α5 : Εγκατάσταση της MySQL σε Ubuntu

Παρακάτω θα δείξουμε ποια είναι τα βήματα εγκατάστασης της MySQL και σε λειτουργικό Ubuntu.

Το πρώτο βήμα που χρειάζεται να πραγματοποιηθεί στο Ubuntu (εδώ μιλάμε με την υπόθεση ότι είναι ήδη εγκαταστημένο το λειτουργικό) είναι να έχουμε κατεβάσει και εγκαταστήσει όλα τα τελευταία updates. Η διαδικασία αυτή είναι πολύ εύκολη. Πηγαίνουμε στο κεντρικό μενού **System**, επιλέγουμε **Administration** και μετά **Update Manager.**

Στην συνέχεια θα μας εμφανιστεί το παράθυρο του Update Manager. Εάν υπάρχουν διαθέσιμα updates, τότε θα εμφανιστούν μέσα στο τετράγωνο πλαίσιο του παραθύρου και το κουμπί Install Updates θα είναι ενεργοποιημένο. Απλά το επιλέγουμε, και το σύστημα θα κατεβάσει και θα εγκαταστήσει όλα τα διαθέσιμα updates. Εάν μας ζητηθεί να κάνουμε επανεκκίνηση στο λειτουργικό, τότε απλά δεχόμαστε το αίτημα αυτό, και μετά την επαναφορά του συστήματος θα είμαστε έτοιμοι να συνεχίσουμε με την εγκατάσταση της MySQL. Στην πιο κάτω εικόνα η οποία προέρχεται από τον υπολογιστή μου, το σύστημα είναι ενημερωμένο και δεν υπάρχουν διαθέσιμα updates οπότε δεν υπάρχει λόγος για επανεκκίνηση.



Εικόνα Α5.α : Παράθυρο του Update Manager.

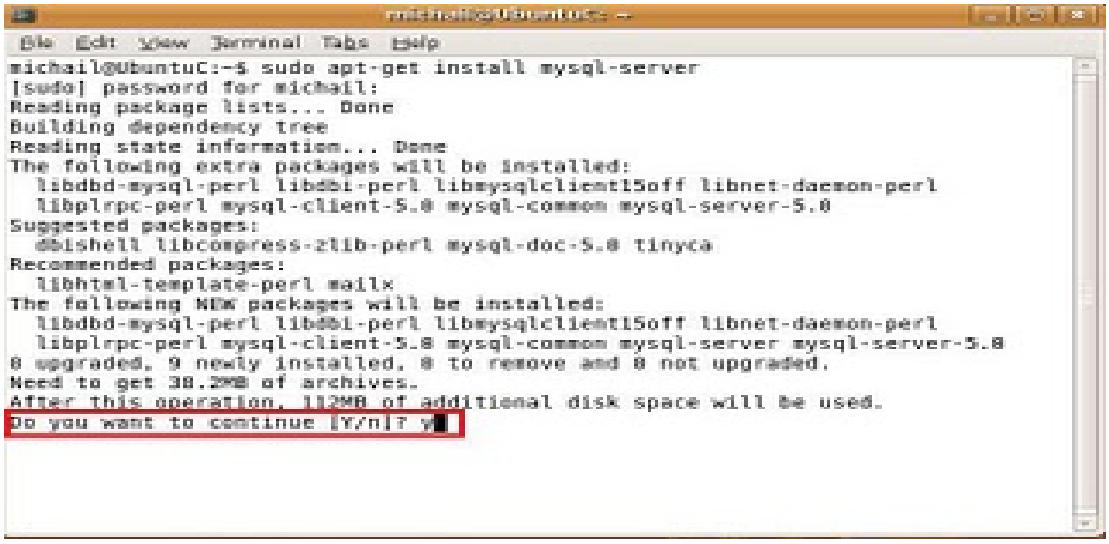
Για να εγκαταστήσουμε την MySQL χρειάζεται να καλέσουμε μέσα από το terminal την πιο κάτω εντολή:

sudo apt-get install mysql-server

Αυτό θα ενεργοποιήσει τον μηχανισμό εγκατάστασης της MySQL στο Ubuntu, το οποίο σε πολύ μικρό χρονικό διάστημα θα έχει ολοκληρωθεί. Όταν μας ζητηθεί το password, απλά γράφουμε το κωδικό πρόσβασης που έχουμε σαν sudo χρήστης στο Ubuntu. Είναι λογικό το σύστημα να μην

αφήνει κανέναν άλλο εκτός από τον Administrator να έχει δικαιώματα εγκατάστασης προγραμμάτων στο λειτουργικό.

Απαντάμε θετικά (γράφοντας απλά το γράμμα “y” και επιλέγοντας ENTER) στην ερώτηση που μας ρωτάει το σύστημα, εάν δεχόμαστε να χρησιμοποιηθούν περίπου 112MB από τον σκληρό δίσκο για την εγκατάσταση της MySQL.

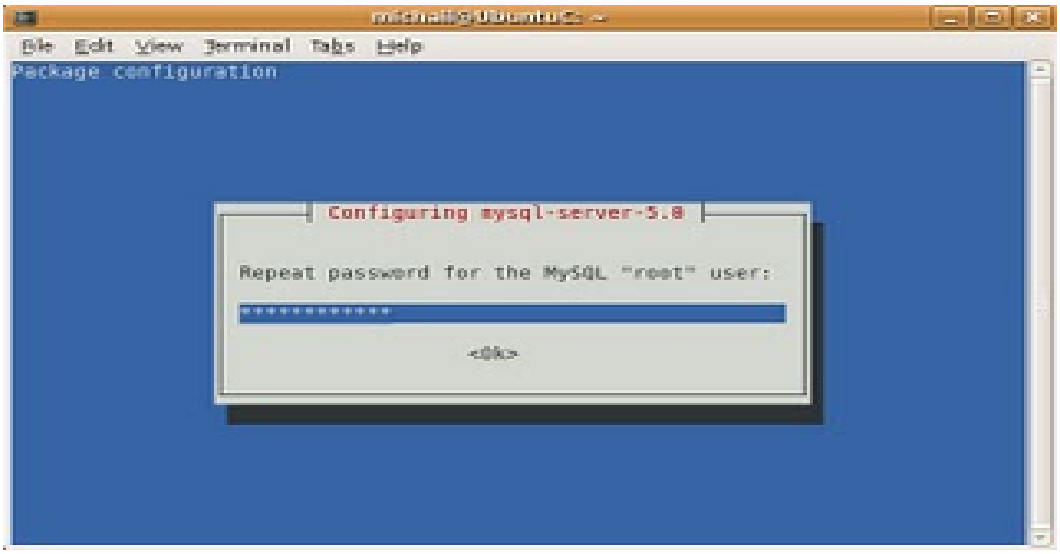


Εικόνα Α5.β: Διαδικασία εγκατάστασης της MySQL.

Αμέσως μετά την απάντηση μας, το Ubuntu ενώνεται με το Ίντερνετ για να κατεβάσει τα απαραίτητα αρχεία και να τα εγκαταστήσει.

Όταν τελειώσει με αυτή την διαδικασία, θα μας ζητηθεί να καταχωρήσουμε ένα password για το root λογαριασμό της βάσης, το οποίο φυσικά θα μας δίνει απεριόριστη πρόσβαση στις ιδιότητες της. Όταν γράψουμε τον κωδικό συνεχίζουμε με ENTER.

Για να βεβαιωθεί το σύστημα ότι γράψαμε σωστά τον κωδικό πρόσβασης μας ζητάει να το γράψουμε ακόμα μια φορά. Οπότε επαναλαμβάνουμε τον ίδιο κωδικό και πατάμε ENTER.



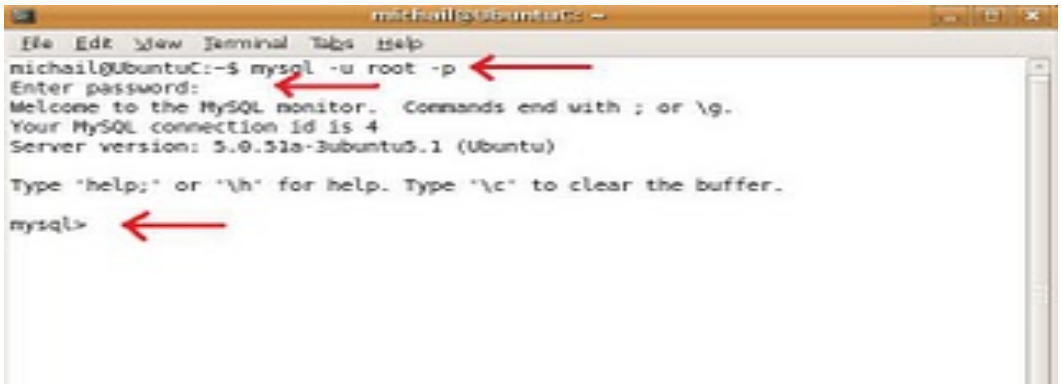
Εικόνα Α5.γ : Διαδικασία επιβεβαίωσης του συστήματος για τον κωδικό πρόσβασης.

Αφού επιβεβαιωθεί ότι ο κωδικός που γράψαμε είναι σωστός και τις δύο φορές, τότε το Ubuntu ολοκληρώνει την εγκατάσταση και μας επαναφέρει πίσω στο terminal prompt.

Αυτό που έχει απομείνει τώρα, είναι να δοκιμάσουμε την σύνδεση μας με την βάση. Αυτό γίνεται δίνοντας την εντολή:

```
mysql -u root -p
```

Με το mysql δηλώνουμε ότι θέλουμε να ενεργοποιήσουμε ένα session με την **mysql**, το **-u** δηλώνει ότι θα δώσουμε αμέσως μετά το όνομα του χρήστη που θέλει να συνδεθεί (root) και το **-p** δηλώνει ότι θα μας ζητηθεί ο κωδικός σύνδεσης όταν πατήσουμε ENTER. Πραγματικά, στην επόμενη γραμμή στο terminal μας ζητείται ο κωδικός πρόσβασης. Αφού επιτυχημένα γράψουμε και τον κωδικό, τότε το prompt αλλάζει σε mysql που συμβολίζει την επιτυχής ένωση.



Εικόνα Α5.δ : Ενεργοποίηση ενός session.

Όπως και με την εγκατάσταση στην πλατφόρμα των Windows έτσι και εδώ θα χρησιμοποιήσουμε την γνωστή εντολή **SHOW DATABASES** απλά για να βεβαιωθούμε ότι μπορούμε να επικοινωνήσουμε σωστά με την MySQL.

Για να διακόψουμε την ένωση μας με την MySQL (να σταματήσουμε το session) απλά γράφουμε την εντολή EXIT και πατάμε ENTER.

Εδώ ολοκληρώσαμε την διαδικασία εγκατάστασης της MySQL και σε λειτουργικό Ubuntu.

ΠΑΡΑΡΤΗΜΑ Β : Εγχειρίδιο χρήσης του προγράμματος

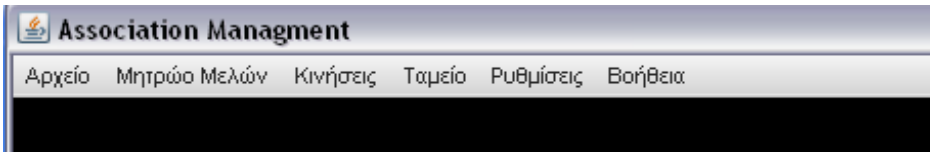
Παράρτημα Β : Εγχειρίδιο χρήσης του προγράμματος

Παράρτημα Β1 : Απαιτήσεις

Ο υπολογιστής στον οποίο θα τρέχει η εφαρμογή πρέπει να έχει εγκατεστημένα τα προγράμματα Java και MySQL.Συνίσταται η έκδοση του MySQL να είναι από 5.1 και άνω, ενώ η έκδοση της Java από 6 και άνω.

Παράρτημα Β2 : Οδηγίες χρήσης του προγράμματος

Στην αρχή του προγράμματος βρίσκονται τα μενού :

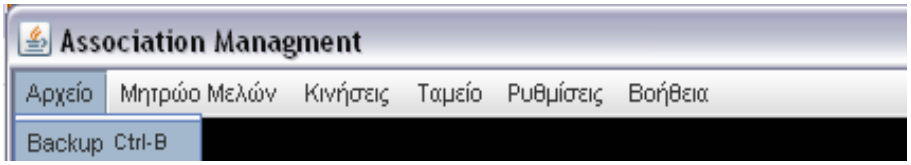


Εικόνα Β2.α : Μενού προγράμματος.

Αυτά τα μενού πραγματοποιούν κάποιες λειτουργίες της εφαρμογής.

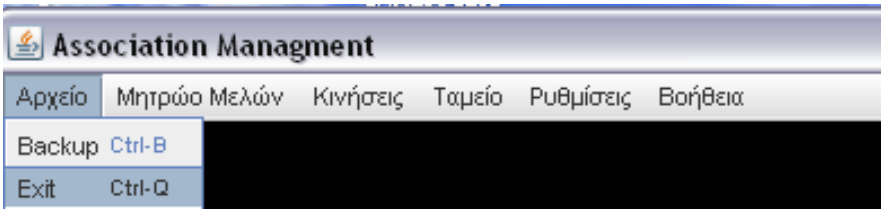
Το μενού **«Αρχείο»** έχει τις εξής επιλογές**:**

1) Αρχείο -> Backup : Αυτή η επιλογή μας δίνει την δυνατότητα να πάρουμε αντίγραφο ασφαλείας της βάσης των δεδομένων μας που αποθηκεύεται στον φάκελο της εφαρμογής.



Εικόνα Β2β : Επιλογή Backup.

2) Αρχείο -> Exit : Αυτή η επιλογή μας επιτρέπει να βγούμε από το πρόγραμμα μας.

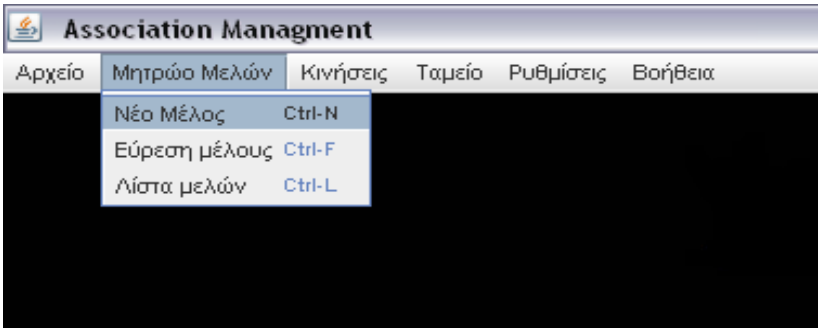


Εικόνα Β2.γ : Επιλογή exit.

Το μενού «**Μητρώο Μελών**» έχει τις εξής επιλογές**:**

1) **Μητρώο Μελών -> Νέο Μέλος :** Εμφανίζει την καρτέλα καταχώρησης νέου μέλους.

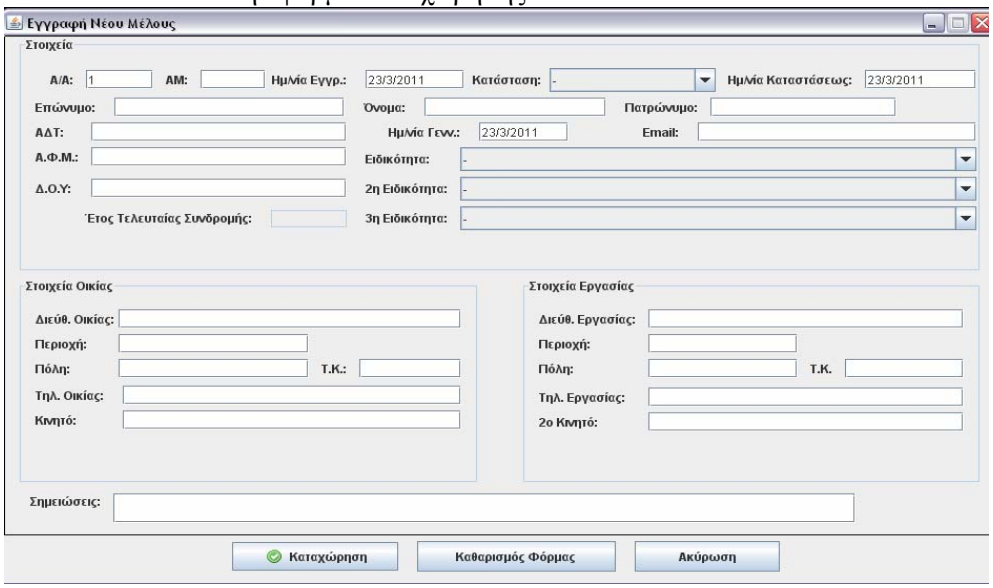
.



Εικόνα Β2.δ : Επιλογή Νέο Μέλος.

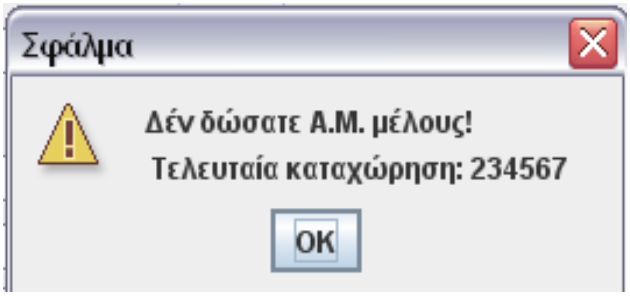
Τα κουμπιά που υπάρχουν στο κάτω μέρος του παραθύρου «Εγγραφή Νέου Μέλους» χρησιμοποιούνται για :

ΚΑΤΑΧΩΡΗΣΗ : Καταχωρεί την εγγραφή στην βάση.
ΚΑΘΑΡΙΣΜΟΣ ΦΟΡΜΑΣ : Καθαρίζει τα πεδία.
ΑΚΥΡΩΣΗ : Κλείνει την φόρμα καταχώρησης.



Εικόνα Β2.ε : Καρτέλα Εγγραφής Μέλους.

Σε περίπτωση που δεν καταχωρηθεί ΑΜ. μέλους εμφανίζεται το παρακάτω μήνυμά :



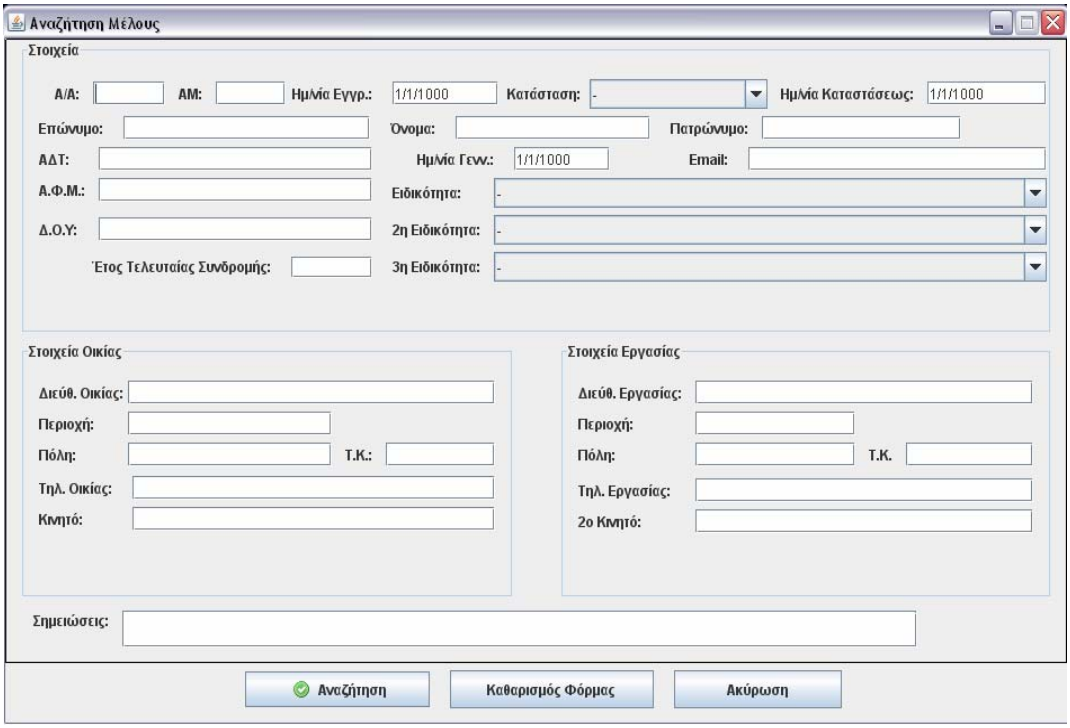
Εικόνα Β2.στ :Μήνυμά Σφάλματος.

Σε περίπτωση που καταχωρηθεί το ίδιο ΑΜ. εμφανίζεται το αντίστοιχο μήνυμά. Το μέλος μπορεί να έχει παραπάνω από μια ειδικότητα και γι' αυτό υπάρχει η δυνατότητα επιλογής τους.

2) Μητρώο Μελών -> Εύρεση Μέλους : Εμφανίζει την καρτέλα εύρεσης κάποιου μέλους με όποιο στοιχείο ζητηθεί.

Τα κουμπιά που υπάρχουν στο κάτω μέρος του παραθύρου «Αναζήτηση Μέλους» χρησιμοποιούνται για :

ΑΝΑΖΗΤΗΣΗ : Γίνεται αναζήτηση κάποιου μέλους με το/τα στοιχείο/α που έχουν ζητηθεί. Εάν δεν δώσουμε κανένα στοιχείο τότε εμφανίζεται η καρτέλα με τη λίστα των μελών.
ΚΑΘΑΡΙΣΜΟΣ ΦΟΡΜΑΣ : Καθαρίζει τα πεδία.
ΑΚΥΡΩΣΗ : Κλείνει την φόρμα καταχώρησης.

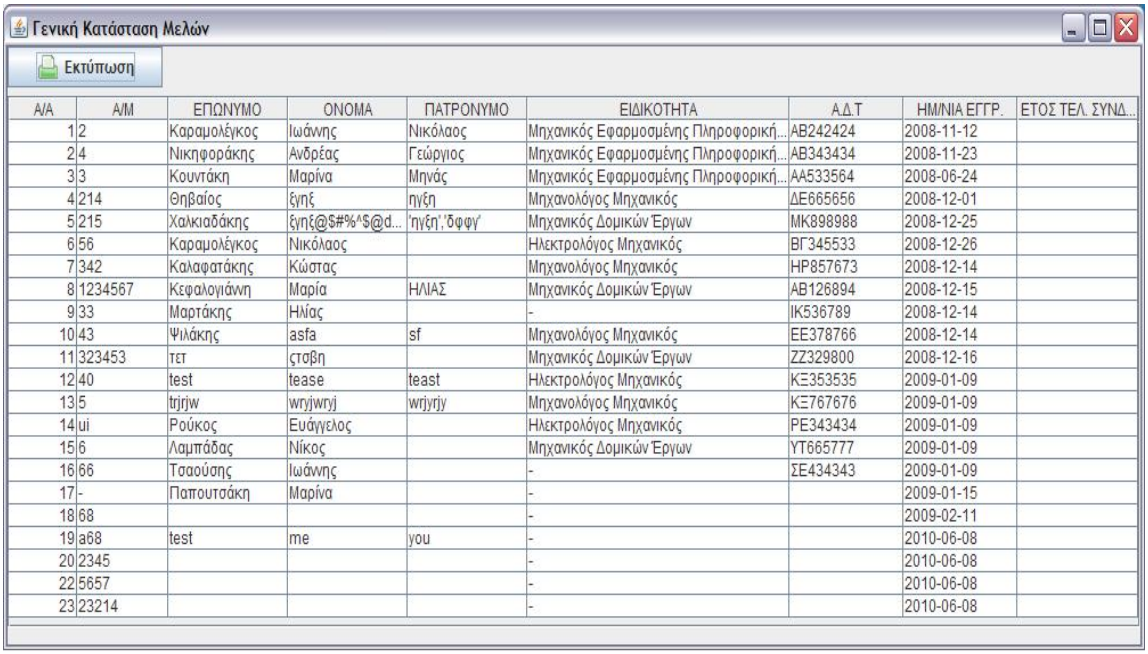


Εικόνα Β2.ζ : Καρτέλα Αναζήτησης Μέλους.

Αν δεν καταχωρηθεί κανένα στοιχείο μέλους για να γίνει η αναζήτηση τότε πατώντας το κουμπί **Αναζήτηση** εμφανίζεται η γενική λίστα των μελών.

3) Μητρώο Μελών -> Λίστα Μελών : Εμφανίζει μια λίστα με όλα τα μέλη που είναι καταχωρημένα στην εφαρμογή μας καθώς και κάποια απαραίτητα στοιχεία για το κάθε μέλος.

Το κουμπί **Εκτύπωση** που υπάρχει στο πάνω μέρος του παραθύρου χρησιμοποιείται για να εκτυπώνει την λίστα μελών όταν ζητηθεί από τον χρήστη.

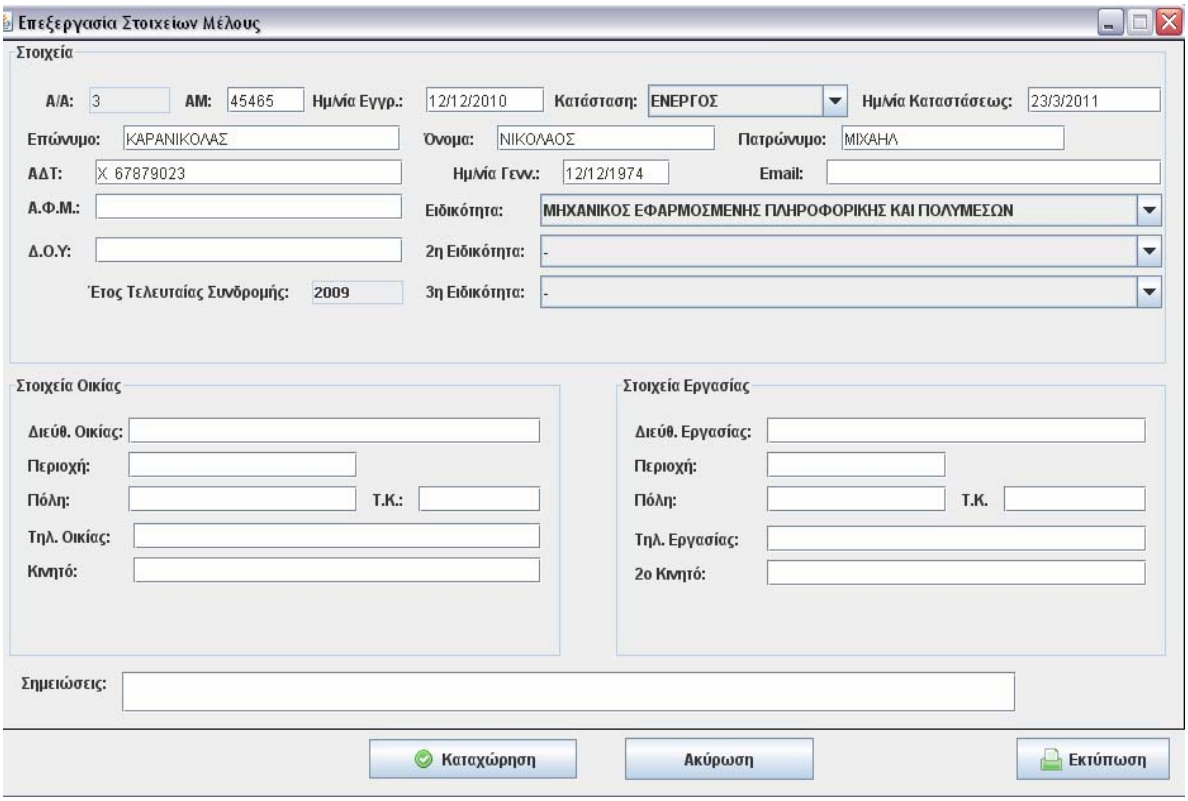


Α/Α	Α/Μ	ΕΠΩΝΥΜΟ	ΟΝΟΜΑ	ΠΑΤΡΩΝΥΜΟ	ΕΙΔΙΚΟΤΗΤΑ	Α.Δ.Τ	ΗΜ/ΝΙΑ ΕΓΓΡ.	ΕΤΟΣ ΤΕΛ. ΣΥΝΔ...
1	2	Καραμολέγκος	Ιωάννης	Νικόλαος	Μηχανικός Εφαρμοσμένης Πληροφορική...	ΑΒ242424	2008-11-12	
2	4	Νικηφοράκης	Ανδρέας	Γεώργιος	Μηχανικός Εφαρμοσμένης Πληροφορική...	ΑΒ343434	2008-11-23	
3	3	Κουντάκη	Μαρίνα	Μηνάς	Μηχανικός Εφαρμοσμένης Πληροφορική...	ΑΑ533564	2008-06-24	
4	214	Θηβαίος	ξγηξ	ηγξη	Μηχανολόγος Μηχανικός	ΔΕ665656	2008-12-01	
5	215	Χαλκιαδάκης	ξγηξ@$#%^$@d...	'ηγξη','δφφγ'	Μηχανικός Δομικών Έργων	ΜΚ898988	2008-12-25	
6	56	Καραμολέγκος	Νικόλαος		Ηλεκτρολόγος Μηχανικός	ΒΓ345533	2008-12-26	
7	342	Καλαφατάκης	Κώστας		Μηχανολόγος Μηχανικός	ΗΡ857673	2008-12-14	
8	1234567	Κεφαλογιάννη	Μαρία	ΗΛΙΑΣ	Μηχανικός Δομικών Έργων	ΑΒ126894	2008-12-15	
9	33	Μαρτάκης	Ηλίας		-	ΙΚ536789	2008-12-14	
10	43	Ψιλάκης	asfa	sf	Μηχανολόγος Μηχανικός	ΕΕ378766	2008-12-14	
11	323453	τετ	ςτσβη		Μηχανικός Δομικών Έργων	ΖΖ329800	2008-12-16	
12	40	test	tease	teast	Ηλεκτρολόγος Μηχανικός	ΚΞ353535	2009-01-09	
13	5	trjrjw	wryjwryj	wrjyrjy	Μηχανολόγος Μηχανικός	ΚΞ767676	2009-01-09	
14	ui	Ρούκος	Ευάγγελος		Ηλεκτρολόγος Μηχανικός	ΡΕ343434	2009-01-09	
15	6	Λαμπάδας	Νίκος		Μηχανικός Δομικών Έργων	ΥΤ665777	2009-01-09	
16	66	Τσαούσης	Ιωάννης		-	ΣΕ434343	2009-01-09	
17	-	Παπουτσάκη	Μαρίνα		-		2009-01-15	
18	68				-		2009-02-11	
19	a68	test	me	you	-		2010-06-08	
20	2345				-		2010-06-08	
22	5657				-		2010-06-08	
23	23214				-		2010-06-08	

Εικόνα Β2.η : Καρτέλα Λίστας Μελών.

Κάνοντας "κλικ" στο τίτλο μίας στήλης τότε τα δεδομένα ταξινομούνται με βάση την συγκεκριμένη στήλη. Π.χ. κάνοντας κλικ πάνω στο τίτλο της στήλης "ΟΝΟΜΑ" ταξινομούνται με βάση το όνομα ή αν κάνουμε κλικ πάνω στο τίτλο "Α/Α" τότε ταξινομούνται με βάση τον αύξων αριθμό.

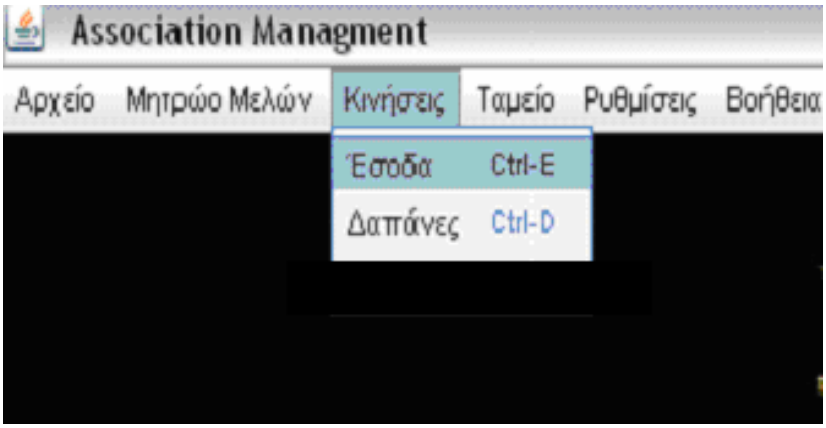
Αν κάνουμε διπλό κλικ πάνω σε ένα μέλος τότε εμφανίζεται η καρτέλα με τα στοιχεία του. Επίσης μπορούμε να κάνουμε και εκτύπωση της καρτέλας του πατώντας το κουμπί **Εκτύπωση.**



Εικόνα Β2.θ : Καρτέλα Στοιχείων Μέλους.

Το μενού «**Κινήσεις**» έχει τις εξής επιλογές**:**

1) Κινήσεις -> Έσοδα : Ανοίγει την καρτέλα για την καταχώρηση των εσόδων (π.χ. εγγραφή, συνδρομή έτους κτλ.).

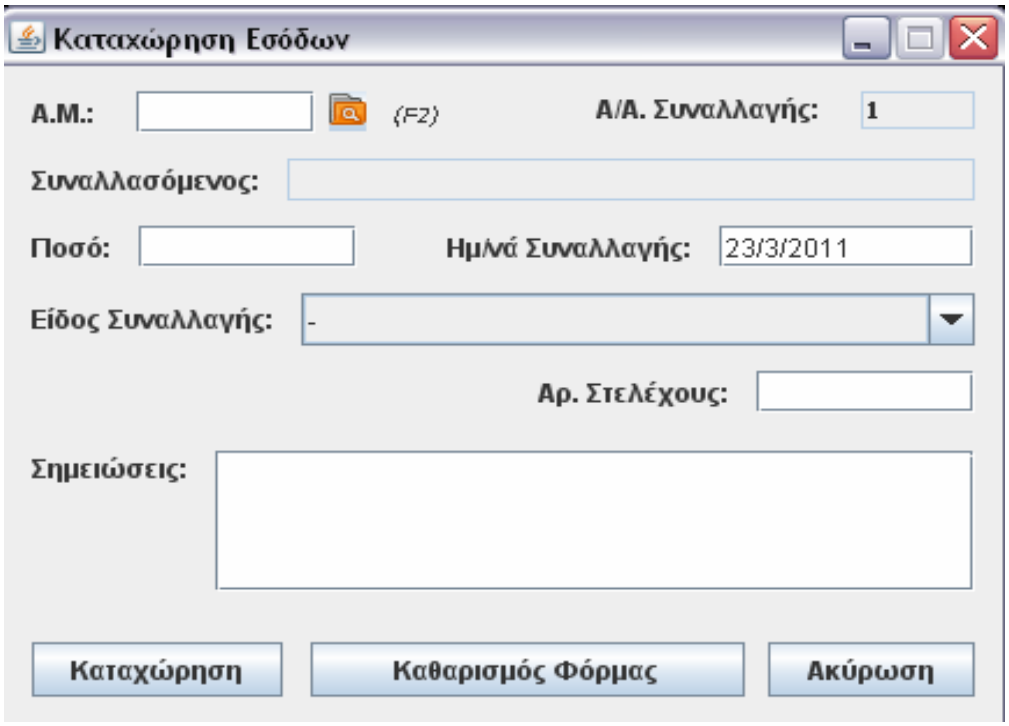


Εικόνα Β2.ι : Επιλογή Έσοδα.

Τα κουμπιά που υπάρχουν στο κάτω μέρος του παραθύρου «Καταχώρηση Εσόδων» χρησιμοποιούνται για :

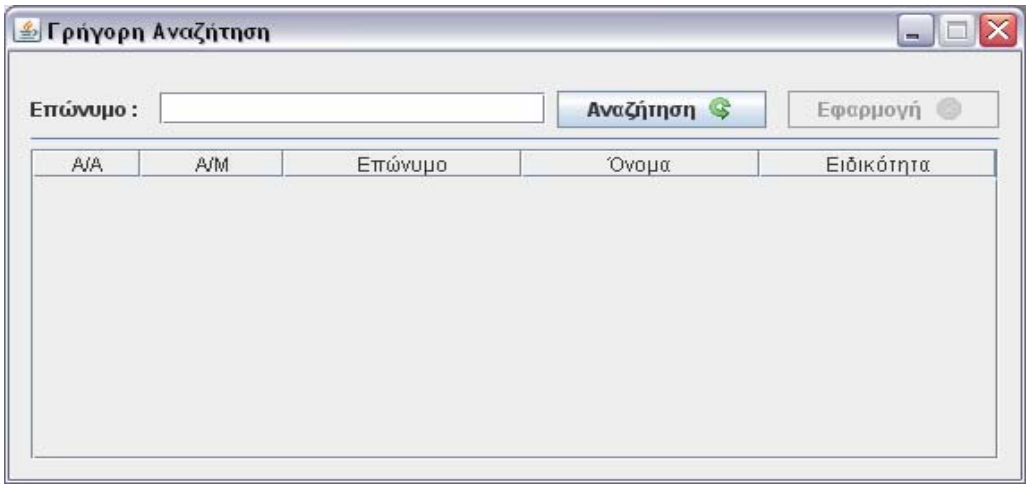
ΚΑΤΑΧΩΡΗΣΗ : Καταχωρεί την συναλλαγή στην βάση.

ΚΑΘΑΡΙΣΜΟΣ ΦΟΡΜΑΣ : Καθαρίζει τα πεδία.
ΑΚΥΡΩΣΗ : Κλείνει την φόρμα καταχώρησης.



Εικόνα Β2.ια : Καρτέλα Καταχώρησης Εσόδων.

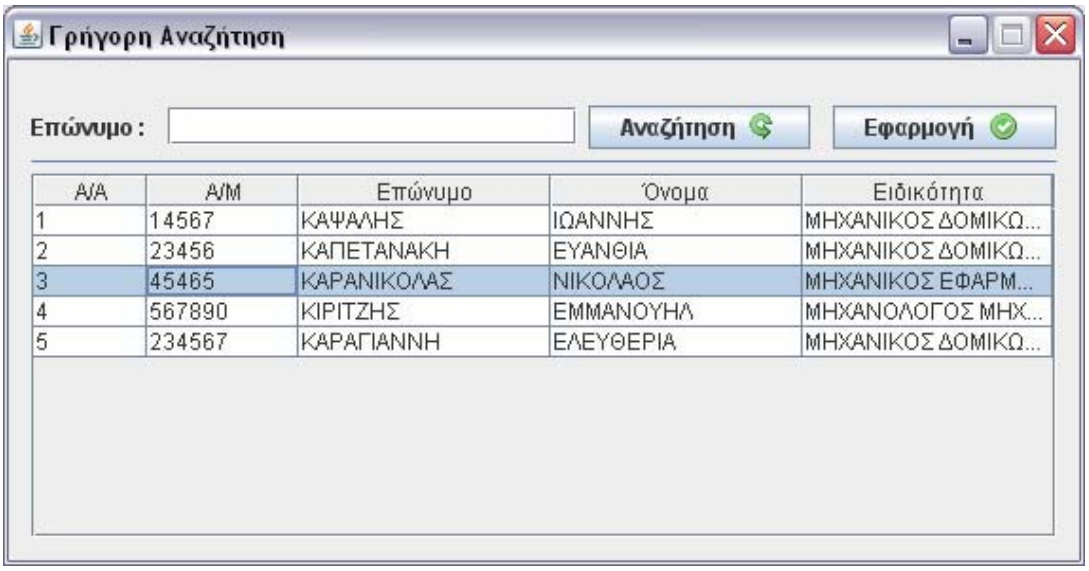
Επιλέγοντας το εικονίδιο με το φάκελο δίπλα στο Α.Μ. (ή το πλήκτρο F2) εμφανίζεται μια μάσκα γρήγορης αναζήτησης μέλους με βάση το επώνυμο.



Εικόνα Β2.ιβ : Μάσκα Αναζήτησης Μέλους.

Για εμφάνιση όλων των καταχωρίσεων το πεδίο «Επώνυμο» πρέπει να παραμείνει κενό.

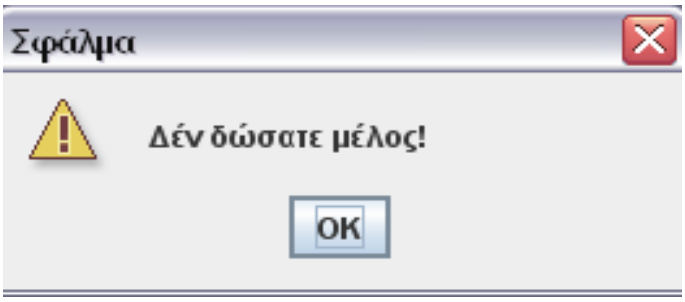
Επιλέγοντας το μέλος (με διπλό «κλικ» ή με επιλογή του και επιλέγοντας «Εφαρμογή») επιστρέφουμε στην καρτέλα καταχώρησης εξόδων με το πεδίο ΑΜ να έχει πλέον το ΑΜ του μέλους καθώς και το Ονομ/νυμο του στο πεδίο Συναλλασσόμενος.



Εικόνα Β2.ιγ : Καρτέλα Γρήγορης Αναζήτησης.

Σε περίπτωση που καταχωρηθεί Α.Μ. που δεν υπάρχει εμφανίζεται ένα μήνυμά ενημέρωσης ότι δεν υπάρχει μέλος με το συγκεκριμένο Α.Μ..

Σε περίπτωση που το πεδίο ΑΜ. μέλους είναι κενό κατά την Καταχώρηση εμφανίζεται το παρακάτω μήνυμά.



Εικόνα Β2.ιδ : Μήνυμά Σφάλματος.

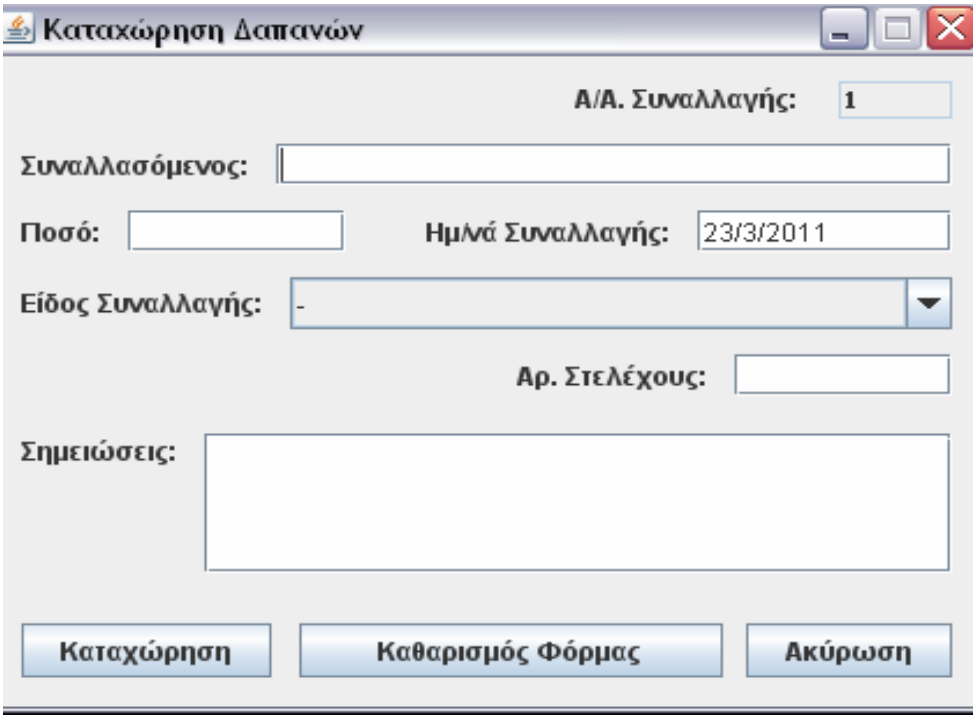
Στο «Είδος Συναλλαγής» γίνεται η επιλογή που θέλει κάθε φορά ο χρήστης (το πεδίο περιέχει δεδομένα τα οποία εισάγονται από ειδική φόρμα).

2) Κινήσεις -> Δαπάνες : Ανοίγει την καρτέλα για την καταχώρηση δαπανών (π.χ. πληρωμή ενοικίου, πληρωμή λογαριασμού ΔΕΗ κτλ.)

Τα κουμπιά που υπάρχουν στο κάτω μέρος του παραθύρου «Καταχώρηση Δαπανών» χρησιμοποιούνται για :

ΚΑΤΑΧΩΡΗΣΗ : Καταχωρεί την συναλλαγή στην βάση.

ΚΑΘΑΡΙΣΜΟΣ ΦΟΡΜΑΣ : Καθαρίζει τα πεδία.
ΑΚΥΡΩΣΗ : Κλείνει την φόρμα καταχώρησης



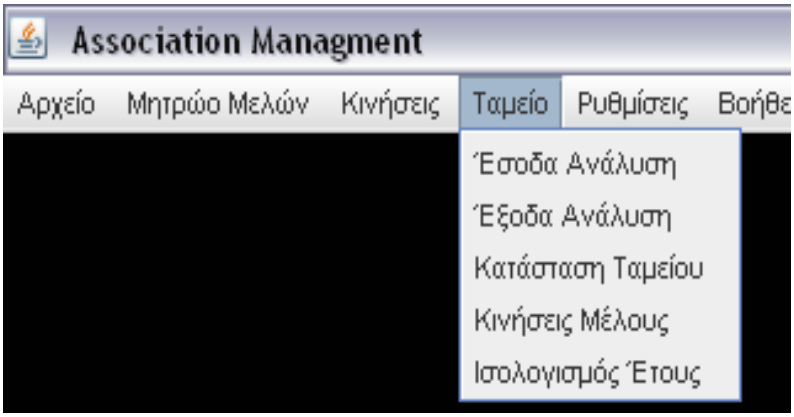
Εικόνα Β2.ιε : Καρτέλα Καταχώρησης Δαπανών.

Τα πεδία **Συναλλασσόμενος** και **Ποσό** είναι υποχρεωτικά διαφορετικά εμφανίζονται τα αντίστοιχα μηνύματα.

Στο είδος της συναλλαγής γίνεται η επιλογή που θέλει κάθε φορά ο χρήστης (το πεδίο περιέχει δεδομένα τα οποία εισάγονται από ειδική φόρμα).

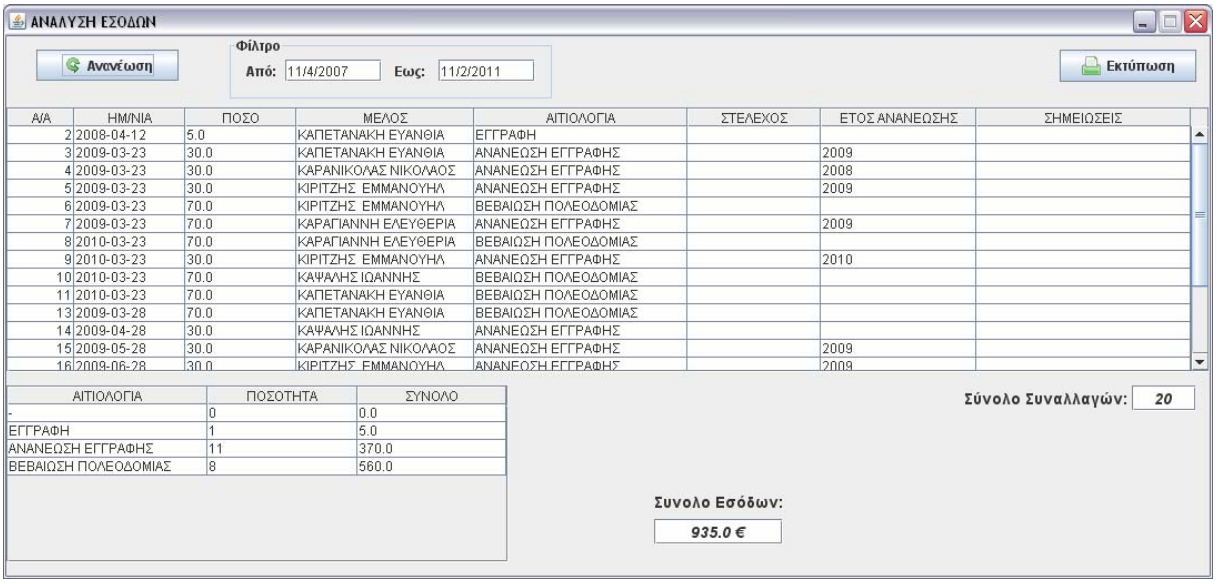
Το μενού «**Ταμείο»** έχει τις εξής επιλογές:

1) Ταμείο -> Έσοδα Ανάλυση : Ανοίγει καρτέλα με ανάλυση των εσόδων για χρονική περίοδο που θα ορίσει ο χρήστης**.**



Εικόνα Β2.ιστ : Επιλογές Μενού Ταμείο.

Το κουμπί **ΑΝΑΝΕΩΣΗ** που υπάρχει στο πάνω μέρος του παραθύρου «Ανάλυση Εσόδων» χρησιμοποιείται για να εμφανίζει την λίστα εσόδων της περιόδου που έχει (ζητηθεί) καταχωρηθεί.

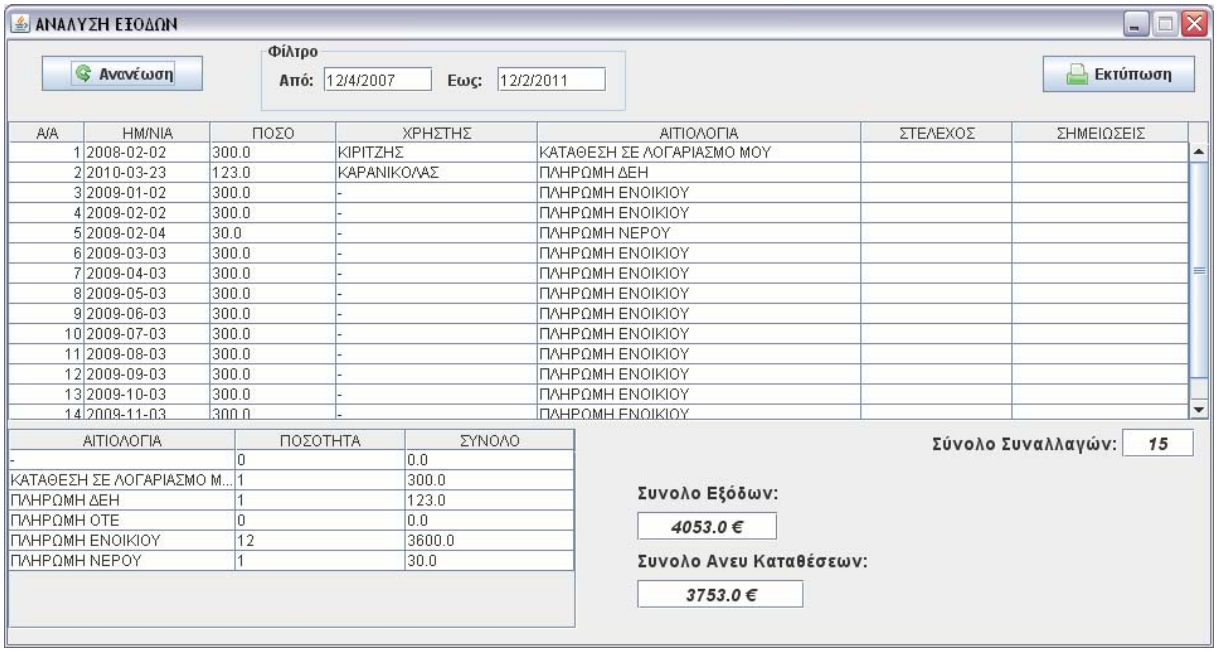


Εικόνα Β2.ιζ : Καρτέλα Ανάλυσης Εσόδων.

Στο παράθυρο εμφανίζεται αναλυτική λίστα με τις εγγραφές των εσόδων ,το σύνολο των συναλλαγών, σύντομη λίστα με τα έσοδα ανά κατηγορία και το σύνολο των εσόδων της περιόδου που έχει ζητηθεί. Μπορούμε να κάνουμε και εκτύπωση της αναλυτικής λίστας επιλέγοντας το κουμπί **Εκτύπωση.**

2) Ταμείο -> Έξοδα Ανάλυση : Ανοίγει καρτέλα με ανάλυση των εξόδων για χρονική περίοδο που θα ορίσει ο χρήστης**.**

Το κουμπί **ΑΝΑΝΕΩΣΗ** που υπάρχει στο πάνω μέρος του παραθύρου «Ανάλυση Εξόδων» χρησιμοποιείται για να εμφανίζει την λίστα εξόδων της περιόδου που έχει (ζητηθεί) καταχωρηθεί.



Εικόνα Β2.ιη : Καρτέλα Ανάλυσης Εξόδων.

Στο παράθυρο εμφανίζεται αναλυτική λίστα με τις εγγραφές των εξόδων, το σύνολο των συναλλαγών, σύντομη λίστα με τα έξοδα ανά κατηγορία και το σύνολο των εξόδων της περιόδου που έχει ζητηθεί. Επίσης υπάρχει και ένα σύνολο εξόδων που είναι το ποσό που υπάρχει στο ταμείο μετά από κατάθεση. Μπορούμε να κάνουμε και εκτύπωση της αναλυτικής λίστας επιλέγοντας το κουμπί **Εκτύπωση.**

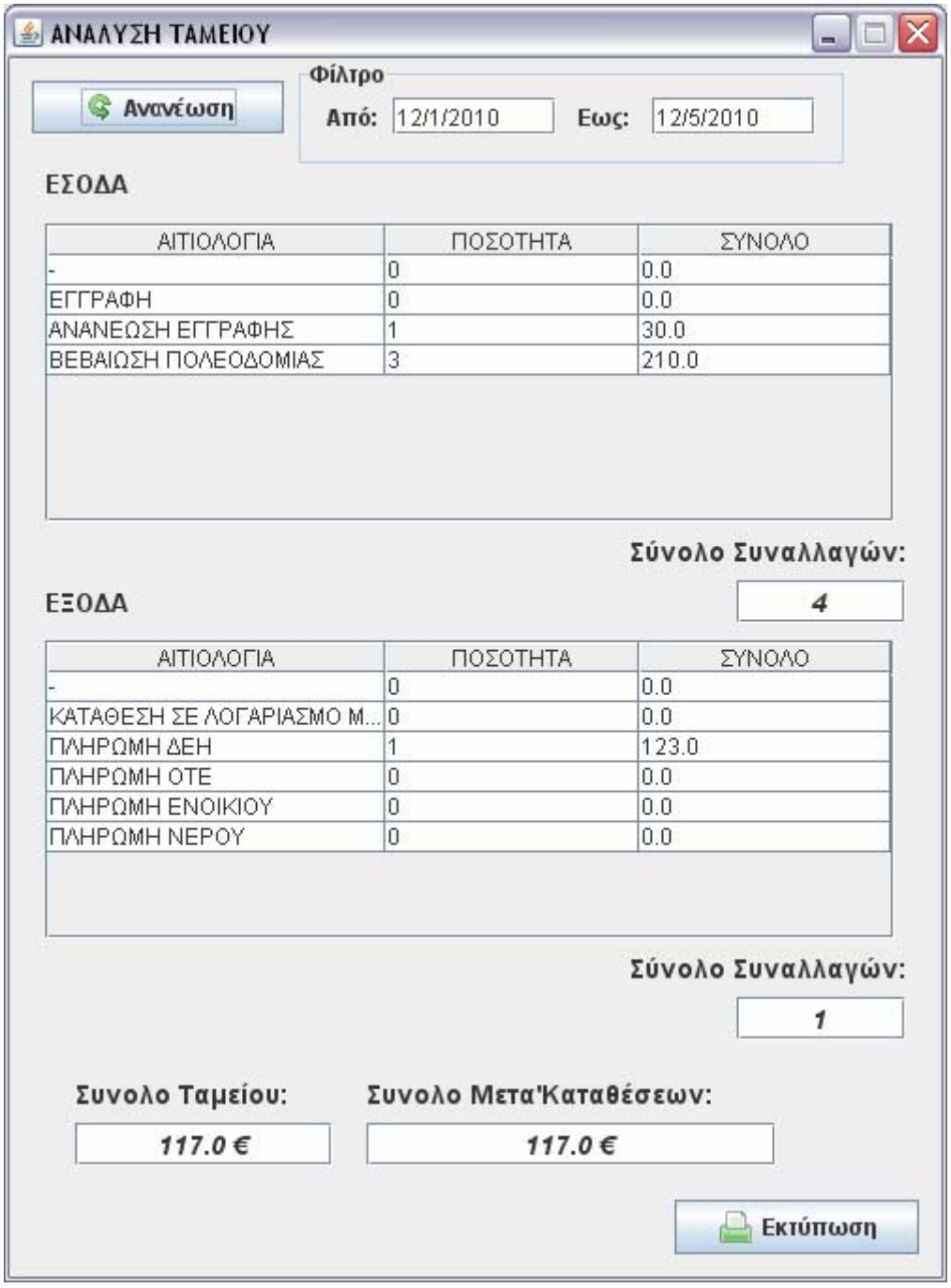
3) Ταμείο -> Κατάσταση Ταμείου : Ανοίγει καρτέλα με ανάλυση των εσόδων και των εξόδων για χρονική περίοδο που θα ορίσει ο χρήστης**.**

Το κουμπί **ΑΝΑΝΕΩΣΗ** που υπάρχει στο πάνω μέρος του παραθύρου «Ανάλυση Ταμείου» χρησιμοποιείται για να εμφανίζει τις λίστες εσόδων και εξόδων της περιόδου που έχει (ζητηθεί) καταχωρηθεί

Κάτω από τις λίστες εσόδων και εξόδων εμφανίζει και το σύνολο των συναλλαγών που έχουν καταχωρηθεί την περίοδο αυτή.

Στο τέλος του παραθύρου εμφανίζει το σύνολο του ταμείου καθώς και το σύνολο με τις καταθέσεις μου.

Μπορούμε να κάνουμε και εκτύπωση της αναλυτικής λίστας επιλέγοντας το κουμπί **Εκτύπωση.**



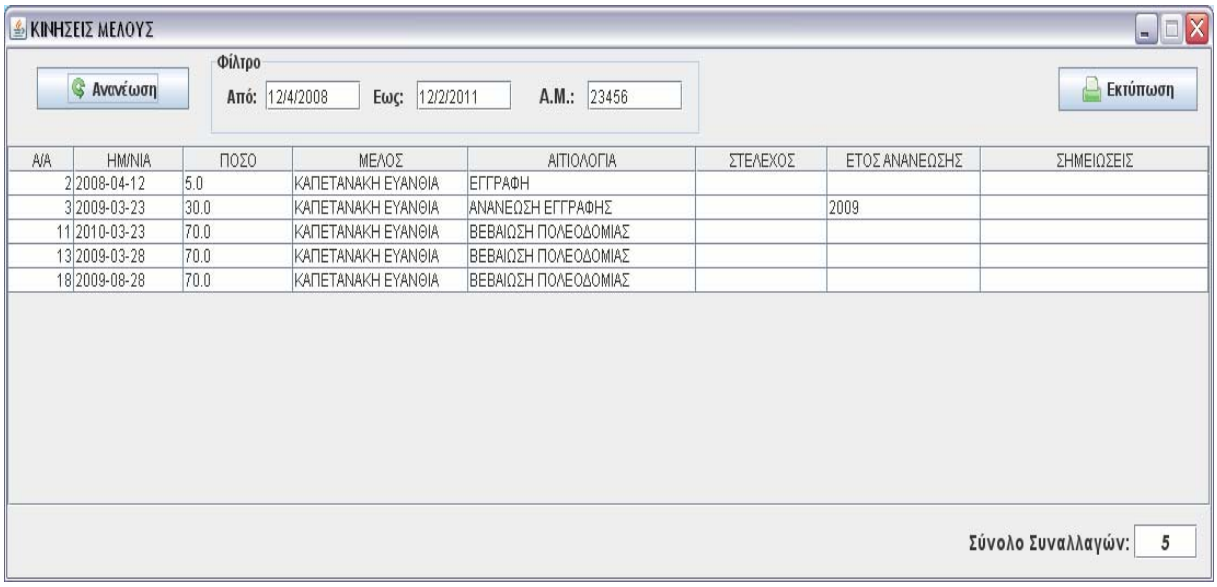
Εικόνα Β2.ιθ : Καρτέλα Ανάλυσης Ταμείου.

4) Ταμείο -> Κινήσεις Μέλους : Ανοίγει καρτέλα για αναλυτική κατάσταση συναλλαγών κάποιου μέλους.

Το κουμπί **ΑΝΑΝΕΩΣΗ** που υπάρχει στο πάνω μέρος του παραθύρου «Κινήσεις Μέλους» χρησιμοποιείται για να εμφανίζει λίστα συναλλαγών κάποιου μέλους που ορίζει ο χρήστης σε μια χρονική περίοδο.

Στο τέλος του παραθύρου εμφανίζει το σύνολο των συναλλαγών που έχουν καταχωρηθεί την περίοδο αυτή.

Μπορούμε να κάνουμε και εκτύπωση της λίστας επιλέγοντας το κουμπί **Εκτύπωση.**

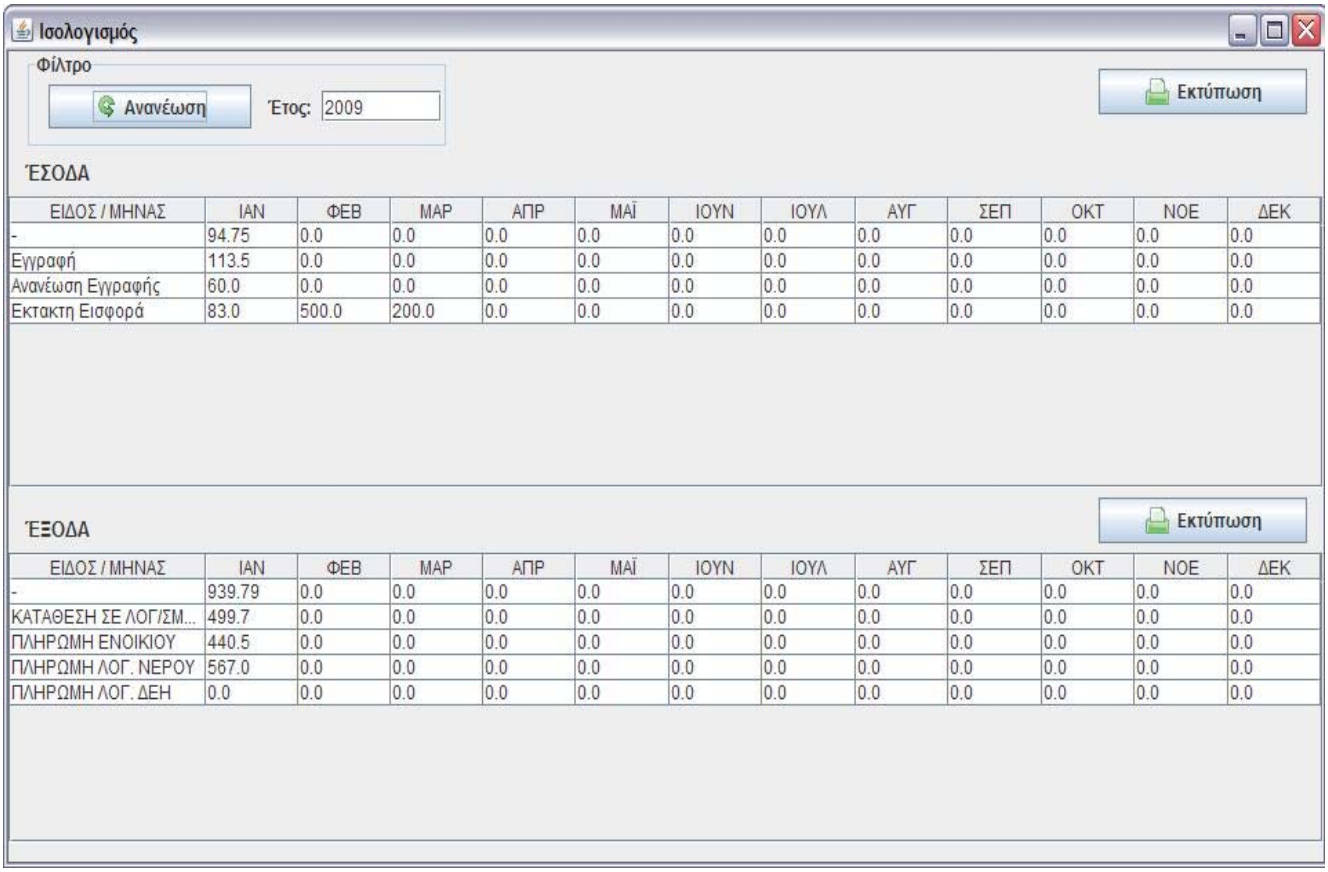


Εικόνα Β2.κ : Καρτέλα Κινήσεις Μέλους.

5) Ταμείο -> Ισολογισμός Έτους : Ανοίγει καρτέλα με αναλυτική κατάσταση των εσόδων και των εξόδων του έτους που θα ορίσει ο χρήστης**.**

Το κουμπί **ΑΝΑΝΕΩΣΗ** που υπάρχει στο πάνω μέρος του παραθύρου «Ισολογισμός» χρησιμοποιείται για να εμφανίζει έσοδα και έξοδα για κάθε μήνα του έτους που έχουμε καταχωρήσει.

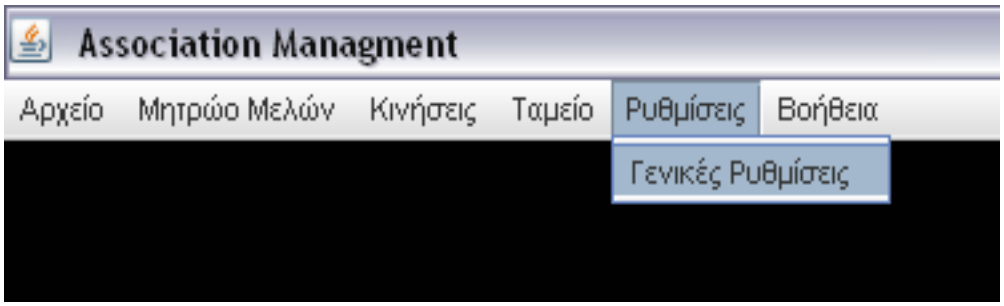
Τα κουμπιά **Εκτύπωση** μας παρέχουν εκτύπωση της κάθε λίστας.



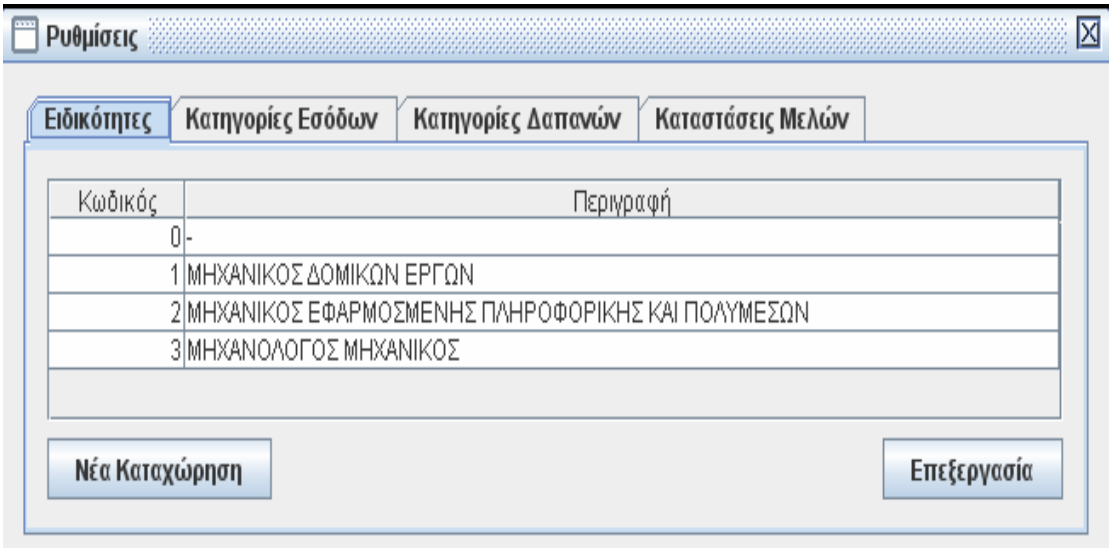
Εικόνα Β2.κα : Καρτέλα Ισολογισμού.

Το μενού «**Ρυθμίσεις**» έχει τις εξής επιλογές:

1) Ρυθμίσεις -> Γενικές Ρυθμίσεις : Ανοίγει την καρτέλα με τις κατηγορίες που μπορούμε να ρυθμίσουμε. Οι κατηγορίες στις οποίες έχουμε πρόσβαση έχουν ήδη αναφερθεί σαν επιλογές σε καρτέλες εισαγωγής δεδομένων. Πρόκειται για τις εξής κατηγορίες «Ειδικότητες – Έσοδα – Δαπάνες – Καταστάσεις Μελών»

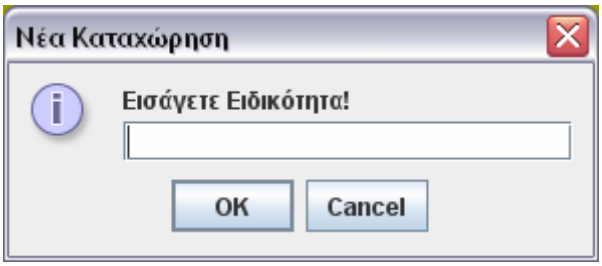


Εικόνα Β2.κβ : Επιλογή Γενικές Ρυθμίσεις.



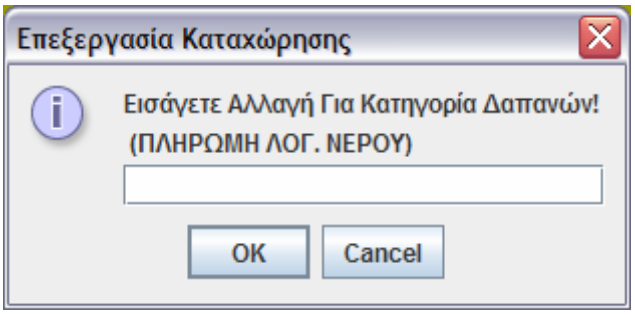
Εικόνα Β2.κγ : Καρτέλα Ρυθμίσεων.

Όταν διαλέξουμε την κατηγορία που θέλουμε και επιλέξουμε το κουμπί **Νέα Καταχώρηση** τότε εμφανίζεται το παρακάτω μήνυμά και μπορούμε να εισάγουμε τα στοιχεία που επιθυμούμε.



Εικόνα Β2.κδ : Μήνυμά Νέας Καταχώρησης.

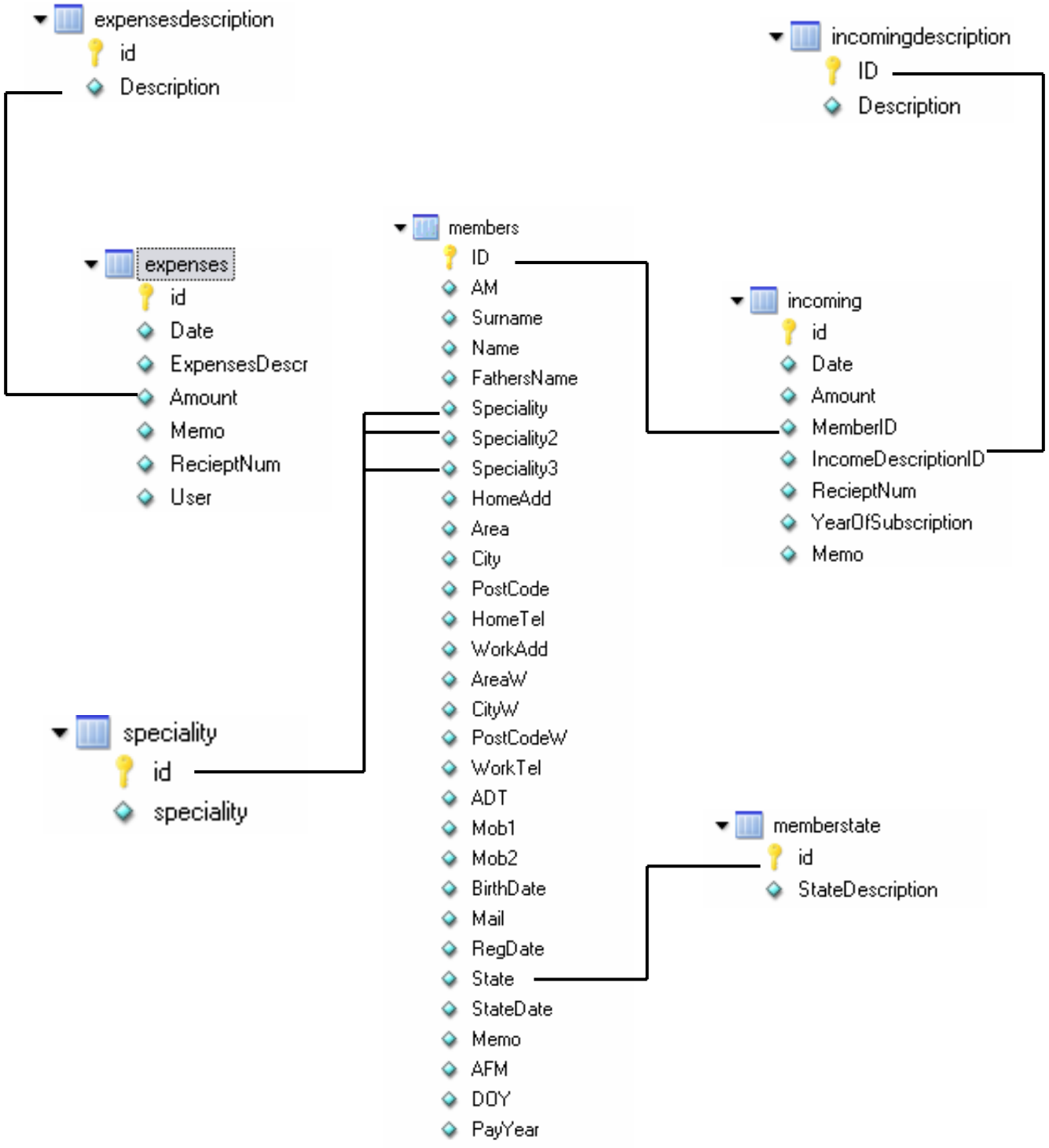
Όταν χρειάζεται αλλαγή κάποιου στοιχείου σε μία από τις κατηγορίες τότε με το κουμπί **Επεξεργασία** εμφανίζεται το παρακάτω μήνυμά :



Εικόνα Β2.κε : Μήνυμά Επεξεργασίας Καταχώρησης.

Όταν δεν επιλέξουμε την κατηγορία που επιθυμούμε να γίνει η αλλαγή τότε εμφανίζεται το αντίστοιχο μήνυμά.

Παράρτημα Β3 : ΔΟΜΗ ΒΑΣΗΣ



ΠΑΡΑΡΤΗΜΑ Γ : API Εφαρμογής

associationmanagment2

Class DBop

java.lang.Object
  └**associationmanagment2.DBop**

---

public class **DBop**
extends java.lang.Object

Field Summary

static java.sql.Connection	**conn**
static java.lang.String	**DBip**
static java.lang.String	**DBname**
static java.lang.String	**DBpass**
static java.lang.String	**DBuser**
static java.lang.String	**msg**

Constructor Summary

DBop()	

Method Summary

static java.sql.Statement	**connect**()
static java.sql.ResultSet	**connect**(java.lang.String table)
static java.lang.String	**Date2String**(java.util.Date date)
static void	**disconnect**()
static int	**EditFields**(java.lang.String table, java.lang.String SearchCol, java.lang.String id, java.lang.String[] fields, java.util.Vector data)
static void	**emptyTable**(javax.swing.JTable ResTable)

static void	**ErrorCatcher**(java.lang.Exception ex)
static void	**expensesTable**(javax.swing.JTable ResTable, java.lang.String StartDate, java.lang.String EndDate)
static void	**fillCombo**(javax.swing.JComboBox box, java.lang.String table, java.lang.String field)
static void	**fillTable**(javax.swing.JTable ResTable, int[] ReadCols, java.lang.String table, java.lang.String SearchCol, java.lang.String SearchData)
static void	**fillTable**(javax.swing.JTable ResTable, int[] ReadCols, java.lang.String table, java.lang.String SearchCol, java.lang.String SearchData, java.lang.String OrderBy)
static void	**fillTableDateFiltered**(javax.swing.JTable ResTable, int[] ReadCols, java.lang.String table, java.lang.String DateColName, java.lang.String StartDate, java.lang.String EndDate)
static void	**fillTableDateIDFiltered**(javax.swing.JTable ResTable, int[] ReadCols, java.lang.String table, java.lang.String DateColName, java.lang.String StartDate, java.lang.String EndDate, java.lang.String IDcol, java.lang.String ID)
static java.lang.String	**getLastAA**()
static java.lang.String	**getLastAA**(java.lang.String table)
static java.lang.String	**getLastAM**(java.lang.String table, java.lang.String column)
static java.lang.String	**getNameFromId**(java.lang.String id, java.lang.String table, java.lang.String SearchCol, java.lang.String DataCol)
static java.util.Vector	**getRecordData**(java.lang.String table, int id, java.lang.String[] fields)
static java.sql.ResultSet	**getRS**(java.lang.String q)
static void	**incomeTable**(javax.swing.JTable ResTable, java.lang.String StartDate, java.lang.String EndDate)
static void	**Isologismos**(javax.swing.JTable ResTable, javax.swing.JTable ResTable1, java.lang.String Year)

static boolean	**IsUnique**(int SearchString, java.lang.String table, java.lang.String SearchCol)
static boolean	**IsUnique**(java.lang.String SearchString, java.lang.String table, java.lang.String SearchCol)
static int	**SaveFields**(java.lang.String table, java.lang.String[] fields, java.util.Vector data)
static void	**Search**(java.lang.String table, java.util.Vector data, java.lang.String[] fields, int[] ReadCols, javax.swing.JTable ResTable)
static void	**TableChangeId2Description**(javax.swing.JTable ResTable, int ReadCol, java.lang.String table, java.lang.String DataCol)

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

DBname
public static java.lang.String **DBname**

DBip
public static java.lang.String **DBip**

DBuser
public static java.lang.String **DBuser**

DBpass
public static java.lang.String **DBpass**

---

conn
public static java.sql.Connection **conn**

---

msg
public static java.lang.String **msg**

Constructor Detail

DBop

```
public DBop()
```

Method Detail

connect

```
public static java.sql.Statement connect()
```

---

connect

```
public static java.sql.ResultSet connect(java.lang.String table)
```

---

disconnect

```
public static void disconnect()
```

---

getRS

```
public static java.sql.ResultSet getRS(java.lang.String q)
```

---

getLastAA

```
public static java.lang.String getLastAA()
```

---

getLastAA

```
public static java.lang.String getLastAA(java.lang.String table)
```

---

getLastAM

```
public static java.lang.String getLastAM(java.lang.String table,
                                         java.lang.String column)
```

---

fillCombo

```
public static void fillCombo(javax.swing.JComboBox box,
                             java.lang.String table,
                             java.lang.String field)
```

SaveFields

```
public static int SaveFields(java.lang.String table,
                             java.lang.String[] fields,
                             java.util.Vector data)
```

ErrorCatcher

```
public static void ErrorCatcher(java.lang.Exception ex)
```

emptyTable

```
public static void emptyTable(javax.swing.JTable ResTable)
```

fillTable

```
public static void fillTable(javax.swing.JTable ResTable,
                             int[] ReadCols,
                             java.lang.String table,
                             java.lang.String SearchCol,
                             java.lang.String SearchData)
```

fillTable

```
public static void fillTable(javax.swing.JTable ResTable,
                             int[] ReadCols,
                             java.lang.String table,
                             java.lang.String SearchCol,
                             java.lang.String SearchData,
                             java.lang.String OrderBy)
```

Date2String

```
public static java.lang.String Date2String(java.util.Date date)
```

fillTableDateFiltered

```
public static void fillTableDateFiltered(javax.swing.JTable ResTable,
                                         int[] ReadCols,
                                         java.lang.String table,
                                         java.lang.String DateColName,
                                         java.lang.String StartDate,
                                         java.lang.String EndDate)
```

fillTableDateIDFiltered

```
public static void fillTableDateIDFiltered(javax.swing.JTable ResTable,
                                           int[] ReadCols,
                                           java.lang.String table,
                                           java.lang.String DateColName,
                                           java.lang.String StartDate,
                                           java.lang.String EndDate,
                                           java.lang.String IDcol,
                                           java.lang.String ID)
```

TableChangeId2Description

```
public static void TableChangeId2Description(javax.swing.JTable ResTable,
                                             int ReadCol,
                                             java.lang.String table,
                                             java.lang.String DataCol)
```

getNameFromId

```
public static java.lang.String getNameFromId(java.lang.String id,
                                              java.lang.String table,
                                              java.lang.String SearchCol,
                                              java.lang.String DataCol)
```

IsUnique

```
public static boolean IsUnique(int SearchString,
                               java.lang.String table,
                               java.lang.String SearchCol)
```

IsUnique

```
public static boolean IsUnique(java.lang.String SearchString,
                               java.lang.String table,
                               java.lang.String SearchCol)
```

EditFields

```
public static int EditFields(java.lang.String table,
                             java.lang.String SearchCol,
                             java.lang.String id,
                             java.lang.String[] fields,
                             java.util.Vector data)
```

---

incomeTable

```
public static void incomeTable(javax.swing.JTable ResTable,
                               java.lang.String StartDate,
                               java.lang.String EndDate)
```

---

expensesTable

```
public static void expensesTable(javax.swing.JTable ResTable,
                                 java.lang.String StartDate,
                                 java.lang.String EndDate)
```

Isologismos

```
public static void Isologismos(javax.swing.JTable ResTable,
                               javax.swing.JTable ResTable1,
                               java.lang.String Year)
```

---

getRecordData

```
public static java.util.Vector getRecordData(java.lang.String table,
                                              int id,
                                              java.lang.String[] fields)
```

Search

```
public static void Search(java.lang.String table,
                         java.util.Vector data,
```

```
                        java.lang.String[] fields,
                        int[] ReadCols,
                        javax.swing.JTable ResTable)
```

Class AddMemberFrame

```
java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └java.awt.Window
              └java.awt.Frame
                  └javax.swing.JFrame
                      └associationmanagment2.AddMemberFrame
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

---

```
public class AddMemberFrame
extends javax.swing.JFrame
```

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JFrame
javax.swing.JFrame.AccessibleJFrame

Nested classes/interfaces inherited from class java.awt.Frame
java.awt.Frame.AccessibleAWTFrame

Nested classes/interfaces inherited from class java.awt.Window
java.awt.Window.AccessibleAWTWindow

Nested classes/interfaces inherited from class java.awt.Container
java.awt.Container.AccessibleAWTContainer

Nested classes/interfaces inherited from class java.awt.Component
java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy

Field Detail

memberDataJPanel1

`public MemberDataJPanel memberDataJPanel1`

Constructor Detail

AddMemberFrame

`public AddMemberFrame()`

Creates new form testJFrame

Class EditMemberFrame

```
java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └java.awt.Window
              └java.awt.Frame
                  └javax.swing.JFrame
                      └associationmanagment2.EditMemberFrame
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

---

`public class EditMemberFrame`
`extends javax.swing.JFrame`

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JFrame
javax.swing.JFrame.AccessibleJFrame

Nested classes/interfaces inherited from class java.awt.Frame
java.awt.Frame.AccessibleAWTFrame

Nested classes/interfaces inherited from class java.awt.Window
java.awt.Window.AccessibleAWTWindow

Nested classes/interfaces inherited from class java.awt.Container
java.awt.Container.AccessibleAWTContainer

Nested classes/interfaces inherited from class java.awt.Component

```
java.awt.Component.AccessibleAWTComponent,
java.awt.Component.BaselineResizeBehavior,
java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy
```

Field Detail

memberDataJPanel1

```
public static MemberDataJPanel memberDataJPanel1
```

OldAm

```
public static java.lang.String OldAm
```

Constructor Detail

EditMemberFrame

```
public EditMemberFrame()
```

Creates new form testJFrame

Method Detail

setAA

```
public static void setAA(java.lang.String aa)
```

Class ExpensesJFrame

```
java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └java.awt.Window
              └java.awt.Frame
                  └javax.swing.JFrame
                      └associationmanagment2.ExpensesJFrame
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

---

```
public class ExpensesJFrame
extends javax.swing.JFrame
```

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JFrame
javax.swing.JFrame.AccessibleJFrame

Nested classes/interfaces inherited from class java.awt.Frame
java.awt.Frame.AccessibleAWTFrame

Nested classes/interfaces inherited from class java.awt.Window
java.awt.Window.AccessibleAWTWindow

Nested classes/interfaces inherited from class java.awt.Container
java.awt.Container.AccessibleAWTContainer

Nested classes/interfaces inherited from class java.awt.Component
java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy

Field Summary

javax.swing.JTextField	**amountField**
javax.swing.JLabel	**amountLabel**
javax.swing.JFormattedTextField	**dateField**
javax.swing.JLabel	**dateLabel**
javax.swing.JTextField	**idField**
static javax.swing.JComboBox	**incomeDescrComboBox**
javax.swing.JLabel	**incomeDescriptionIDLabel**
javax.swing.JTextArea	**memoField**
javax.swing.JLabel	**memoLabel**
javax.swing.JTextField	**recieptNumField**
javax.swing.JLabel	**recieptNumLabel**

javax.swing.JLabel	**UserjLabel**
static javax.swing.JTextField	**UserjTextField**

Field Detail

UserjLabel

public javax.swing.JLabel **UserjLabel**

UserjTextField

public static javax.swing.JTextField **UserjTextField**

---

amountField

public javax.swing.JTextField **amountField**

---

amountLabel

public javax.swing.JLabel **amountLabel**

---

dateField

public javax.swing.JFormattedTextField **dateField**

---

dateLabel

public javax.swing.JLabel **dateLabel**

---

idField

public javax.swing.JTextField **idField**

---

incomeDescrComboBox

public static javax.swing.JComboBox **incomeDescrComboBox**

---

incomeDescriptionIDLabel

public javax.swing.JLabel **incomeDescriptionIDLabel**

---

memoField

public javax.swing.JTextArea **memoField**

---

memoLabel

```
public javax.swing.JLabel memoLabel
```

---

recieptNumField

```
public javax.swing.JTextField recieptNumField
```

---

recieptNumLabel

```
public javax.swing.JLabel recieptNumLabel
```

Constructor Detail

ExpensesJFrame

```
public ExpensesJFrame()
```

Creates new form IncomeJFrame

Method Detail

main

```
public static void main(java.lang.String[] args)
```

Parameters:

`args` - the command line arguments

Class ExpensesStatsJFrame

```
java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └java.awt.Window
              └java.awt.Frame
                  └javax.swing.JFrame
                      └associationmanagment2.ExpensesStatsJFrame
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

---

```
public class ExpensesStatsJFrame
extends javax.swing.JFrame
```

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JFrame

javax.swing.JFrame.AccessibleJFrame

Nested classes/interfaces inherited from class java.awt.Frame
java.awt.Frame.AccessibleAWTFrame

Nested classes/interfaces inherited from class java.awt.Window
java.awt.Window.AccessibleAWTWindow

Nested classes/interfaces inherited from class java.awt.Container
java.awt.Container.AccessibleAWTContainer

Nested classes/interfaces inherited from class java.awt.Component
java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy

Constructor Detail

ExpensesStatsJFrame

`public` **`ExpensesStatsJFrame`**`()`

Creates new form IncomeStatsJFrame

Method Detail

main

`public static void` **`main`**`(java.lang.String[] args)`

Parameters:

`args` - the command line arguments

Class GeneralStatsJFrame

```
java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └java.awt.Window
              └java.awt.Frame
                  └javax.swing.JFrame
                      └associationmanagment2.GeneralStatsJFrame
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

public class **GeneralStatsJFrame**
extends javax.swing.JFrame

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JFrame
javax.swing.JFrame.AccessibleJFrame

Nested classes/interfaces inherited from class java.awt.Frame
java.awt.Frame.AccessibleAWTFrame

Nested classes/interfaces inherited from class java.awt.Window
java.awt.Window.AccessibleAWTWindow

Nested classes/interfaces inherited from class java.awt.Container
java.awt.Container.AccessibleAWTContainer

Nested classes/interfaces inherited from class java.awt.Component
java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy

Constructor Detail

GeneralStatsJFrame

public **GeneralStatsJFrame**()

Creates new form IncomeStatsJFrame

Method Detail

main

public static void **main**(java.lang.String[] args)

Parameters:

args - the command line arguments

Class IncomeJFrame

```
java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └java.awt.Window
              └java.awt.Frame
                  └javax.swing.JFrame
                      └associationmanagment2.IncomeJFrame
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

---

```
public class IncomeJFrame
extends javax.swing.JFrame
```

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JFrame
javax.swing.JFrame.AccessibleJFrame

Nested classes/interfaces inherited from class java.awt.Frame
java.awt.Frame.AccessibleAWTFrame

Nested classes/interfaces inherited from class java.awt.Window
java.awt.Window.AccessibleAWTWindow

Nested classes/interfaces inherited from class java.awt.Container
java.awt.Container.AccessibleAWTContainer

Nested classes/interfaces inherited from class java.awt.Component
java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy

Field Summary

static java.lang.String	**aa**
javax.swing.JTextField	**amountField**

javax.swing.JLabel	**amountLabel**
javax.swing.JFormattedTextField	**dateField**
javax.swing.JLabel	**dateLabel**
static int	**id**
javax.swing.JTextField	**idField**
static javax.swing.JComboBox	**incomeDescrComboBox**
javax.swing.JLabel	**incomeDescriptionIDLabel**
static javax.swing.JTextField	**memberIDField**
javax.swing.JLabel	**memberIDLabel**
javax.swing.JTextArea	**memoField**
javax.swing.JLabel	**memoLabel**
javax.swing.JTextField	**recieptNumField**
javax.swing.JLabel	**recieptNumLabel**
static SimpleNameSearchJFrame	**SimpleNameSearch**
javax.swing.JLabel	**UserjLabel**
static javax.swing.JTextField	**UserjTextField**
javax.swing.JTextField	**yearOfSubscriptionField**
javax.swing.JLabel	**yearOfSubscriptionLabel**

Field Detail

UserjLabel

public javax.swing.JLabel **UserjLabel**

UserjTextField

public static javax.swing.JTextField **UserjTextField**

amountField
public javax.swing.JTextField **amountField**

amountLabel
public javax.swing.JLabel **amountLabel**

dateField
public javax.swing.JFormattedTextField **dateField**

dateLabel
public javax.swing.JLabel **dateLabel**

idField
public javax.swing.JTextField **idField**

incomeDescrComboBox
public static javax.swing.JComboBox **incomeDescrComboBox**

incomeDescriptionIDLabel
public javax.swing.JLabel **incomeDescriptionIDLabel**

memberIDField
public static javax.swing.JTextField **memberIDField**

memberIDLabel
public javax.swing.JLabel **memberIDLabel**

memoField
public javax.swing.JTextArea **memoField**

memoLabel
public javax.swing.JLabel **memoLabel**

recieptNumField
public javax.swing.JTextField **recieptNumField**

recieptNumLabel
public javax.swing.JLabel **recieptNumLabel**

---

yearOfSubscriptionField
public javax.swing.JTextField **yearOfSubscriptionField**

---

yearOfSubscriptionLabel
public javax.swing.JLabel **yearOfSubscriptionLabel**

SimpleNameSearch
public static SimpleNameSearchJFrame **SimpleNameSearch**

---

id
public static int **id**

---

aa
public static java.lang.String **aa**

Constructor Detail

IncomeJFrame
public **IncomeJFrame**()

Creates new form IncomeJFrame

Method Detail

main
public static void **main**(java.lang.String[] args)

Parameters:

args - the command line arguments

---

setAM
public static void **setAM**(java.lang.String am,
java.lang.String newaa)

---

setUser
public static void **setUser**(java.lang.String am)

Class IncomeStatsJFrame

```
java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └java.awt.Window
              └java.awt.Frame
                  └javax.swing.JFrame
                      └associationmanagment2.IncomeStatsJFrame
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

---

```
public class IncomeStatsJFrame
extends javax.swing.JFrame
```

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JFrame
javax.swing.JFrame.AccessibleJFrame

Nested classes/interfaces inherited from class java.awt.Frame
java.awt.Frame.AccessibleAWTFrame

Nested classes/interfaces inherited from class java.awt.Window
java.awt.Window.AccessibleAWTWindow

Nested classes/interfaces inherited from class java.awt.Container
java.awt.Container.AccessibleAWTContainer

Nested classes/interfaces inherited from class java.awt.Component
java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy

Constructor Detail

IncomeStatsJFrame

```
public IncomeStatsJFrame()
```

Creates new form IncomeStatsJFrame

Method Detail

main

```
public static void main(java.lang.String[] args)
```

Parameters:

`args` - the command line arguments

Class IsologismosJFrame

```
java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └java.awt.Window
              └java.awt.Frame
                  └javax.swing.JFrame
                      └associationmanagment2.IsologismosJFrame
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

---

```
public class IsologismosJFrame
extends javax.swing.JFrame
```

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JFrame
javax.swing.JFrame.AccessibleJFrame

Nested classes/interfaces inherited from class java.awt.Frame
java.awt.Frame.AccessibleAWTFrame

Nested classes/interfaces inherited from class java.awt.Window
java.awt.Window.AccessibleAWTWindow

Nested classes/interfaces inherited from class java.awt.Container
java.awt.Container.AccessibleAWTContainer

Nested classes/interfaces inherited from class java.awt.Component

```
java.awt.Component.AccessibleAWTComponent,
java.awt.Component.BaselineResizeBehavior,
java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy
```

Constructor Detail

IsologismosJFrame

```
public IsologismosJFrame()
```

Creates new form IncomeStatsJFrame

Method Detail

main

```
public static void main(java.lang.String[] args)
```

Parameters:

`args` - the command line arguments

Class Main

```
java.lang.Object
  └associationmanagment2.Main
```

```
public class Main
extends java.lang.Object
```

Constructor Detail

Main

```
public Main()
```

Method Detail

main

```
public static void main(java.lang.String[] args)
```

Parameters:

`args` - the command line arguments

Class MainJFrame

```
java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └java.awt.Window
              └java.awt.Frame
                  └javax.swing.JFrame
                      └associationmanagment2.MainJFrame
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

---

```
public class MainJFrame
extends javax.swing.JFrame
```

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JFrame
javax.swing.JFrame.AccessibleJFrame

Nested classes/interfaces inherited from class java.awt.Frame
java.awt.Frame.AccessibleAWTFrame

Nested classes/interfaces inherited from class java.awt.Window
java.awt.Window.AccessibleAWTWindow

Nested classes/interfaces inherited from class java.awt.Container
java.awt.Container.AccessibleAWTContainer

Nested classes/interfaces inherited from class java.awt.Component
java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy

Field Summary

static javax.swing.JFrame	**AddMember**
static javax.swing.JFrame	**CentralFrame**
static javax.swing.JFrame	**Expenses**

static javax.swing.JFrame	**ExpensesStats**
static javax.swing.JFrame	**GeneralStats**
static javax.swing.JFrame	**Income**
static javax.swing.JFrame	**IncomeStats**
static javax.swing.JFrame	**Isologismos**
static javax.swing.JFrame	**MemberIncome**
static javax.swing.JFrame	**Memberslist**
static javax.swing.JFrame	**SearchMembers**
TableDataJPanel	**tableDataJPanel1**
java.lang.String	**test**

Field Detail

tableDataJPanel1

public TableDataJPanel **tableDataJPanel1**

CentralFrame

public static javax.swing.JFrame **CentralFrame**

AddMember

public static javax.swing.JFrame **AddMember**

Income

public static javax.swing.JFrame **Income**

Expenses

public static javax.swing.JFrame **Expenses**

Memberslist
`public static javax.swing.JFrame` **`Memberslist`**

IncomeStats
`public static javax.swing.JFrame` **`IncomeStats`**

ExpensesStats
`public static javax.swing.JFrame` **`ExpensesStats`**

GeneralStats
`public static javax.swing.JFrame` **`GeneralStats`**

Isologismos
`public static javax.swing.JFrame` **`Isologismos`**

SearchMembers
`public static javax.swing.JFrame` **`SearchMembers`**

MemberIncome
`public static javax.swing.JFrame` **`MemberIncome`**

Constructor Detail

MainJFrame
`public` **`MainJFrame`**`()`

Creates new form MainJFrame

Method Detail

main
`public static void` **`main`**`(java.lang.String[] args)`

Parameters:

`args` - the command line arguments

TimeSetter
`public void` **`TimeSetter`**`()`

Class MemberDataJPanel

```
java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └javax.swing.JComponent
              └javax.swing.JPanel
                  └associationmanagment2.MemberDataJPanel
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible

---

```
public class MemberDataJPanel
extends javax.swing.JPanel
```

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JPanel
javax.swing.JPanel.AccessibleJPanel

Nested classes/interfaces inherited from class javax.swing.JComponent
javax.swing.JComponent.AccessibleJComponent

Nested classes/interfaces inherited from class java.awt.Container
java.awt.Container.AccessibleAWTContainer

Nested classes/interfaces inherited from class java.awt.Component
java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy

Field Summary

javax.swing.JTextField	**adtField**
javax.swing.JTextField	**AFMField**
javax.swing.JTextField	**amField**
javax.swing.JTextField	**areaField**
javax.swing.JTextField	**areaWField**
javax.swing.JFormattedTextField	**birthDateField**

javax.swing.JTextField	**cityField**
javax.swing.JTextField	**cityWField**
javax.swing.JTextField	**DOYField**
javax.swing.JTextField	**fathersNameField**
javax.swing.JTextField	**homeAddField**
javax.swing.JTextField	**homeTelField**
javax.swing.JTextField	**idField**
javax.swing.JTextField	**mailField**
javax.swing.JTextArea	**memoArea**
javax.swing.JTextField	**mob1Field**
javax.swing.JTextField	**mob2Field**
javax.swing.JTextField	**nameField**
javax.swing.JTextField	**PayYearField**
javax.swing.JTextField	**postCodeField**
javax.swing.JTextField	**postCodeWField**
javax.swing.JFormattedTextField	**regDateField**
javax.swing.JComboBox	**speciality1ComboBox**
javax.swing.JComboBox	**speciality2ComboBox**
javax.swing.JComboBox	**speciality3ComboBox**
javax.swing.JComboBox	**StateComboBox**
javax.swing.JFormattedTextField	**stateDateField**
javax.swing.JTextField	**surnameField**

javax.swing.JTextField	**workAddField**
javax.swing.JTextField	**workTelField**

Field Detail

AFMField
public javax.swing.JTextField **AFMField**

DOYField
public javax.swing.JTextField **DOYField**

PayYearField
public javax.swing.JTextField **PayYearField**

StateComboBox
public javax.swing.JComboBox **StateComboBox**

---

adtField
public javax.swing.JTextField **adtField**

---

amField
public javax.swing.JTextField **amField**

---

areaField
public javax.swing.JTextField **areaField**

---

areaWField
public javax.swing.JTextField **areaWField**

---

birthDateField
public javax.swing.JFormattedTextField **birthDateField**

---

cityField
public javax.swing.JTextField **cityField**

---

cityWField
public javax.swing.JTextField **cityWField**

---

fathersNameField
public javax.swing.JTextField **fathersNameField**

---

homeAddField
public javax.swing.JTextField **homeAddField**

---

homeTelField
public javax.swing.JTextField **homeTelField**

---

idField
public javax.swing.JTextField **idField**

---

mailField
public javax.swing.JTextField **mailField**

---

memoArea
public javax.swing.JTextArea **memoArea**

---

mob1Field
public javax.swing.JTextField **mob1Field**

---

mob2Field
public javax.swing.JTextField **mob2Field**

---

nameField
public javax.swing.JTextField **nameField**

---

postCodeField
public javax.swing.JTextField **postCodeField**

---

postCodeWField
public javax.swing.JTextField **postCodeWField**

---

regDateField
public javax.swing.JFormattedTextField **regDateField**

speciality1ComboBox
`public javax.swing.JComboBox` **`speciality1ComboBox`**

---

speciality2ComboBox
`public javax.swing.JComboBox` **`speciality2ComboBox`**

---

speciality3ComboBox
`public javax.swing.JComboBox` **`speciality3ComboBox`**

---

stateDateField
`public javax.swing.JFormattedTextField` **`stateDateField`**

---

surnameField
`public javax.swing.JTextField` **`surnameField`**

---

workAddField
`public javax.swing.JTextField` **`workAddField`**

---

workTelField
`public javax.swing.JTextField` **`workTelField`**

Constructor Detail

MemberDataJPanel
`public` **`MemberDataJPanel`**`()`

Creates new form MemberDataJPanel

Method Detail

clear
`public void` **`clear`**`()`

Class MemberIncomeStatsJFrame

```
java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └java.awt.Window
```

```
└java.awt.Frame
    └javax.swing.JFrame
        └associationmanagment2.MemberIncomeStatsJFrame
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

---

```
public class MemberIncomeStatsJFrame
extends javax.swing.JFrame
```

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JFrame
javax.swing.JFrame.AccessibleJFrame

Nested classes/interfaces inherited from class java.awt.Frame
java.awt.Frame.AccessibleAWTFrame

Nested classes/interfaces inherited from class java.awt.Window
java.awt.Window.AccessibleAWTWindow

Nested classes/interfaces inherited from class java.awt.Container
java.awt.Container.AccessibleAWTContainer

Nested classes/interfaces inherited from class java.awt.Component
java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy

Constructor Detail

MemberIncomeStatsJFrame

```
public MemberIncomeStatsJFrame()
```

Creates new form IncomeStatsJFrame

Method Detail

main

```
public static void main(java.lang.String[] args)
```

Parameters:

`args` - the command line arguments

Class MembersListFrame

```
java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └java.awt.Window
              └java.awt.Frame
                  └javax.swing.JFrame
                      └associationmanagment2.MembersListFrame
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

---

```
public class MembersListFrame
extends javax.swing.JFrame
```

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JFrame
javax.swing.JFrame.AccessibleJFrame

Nested classes/interfaces inherited from class java.awt.Frame
java.awt.Frame.AccessibleAWTFrame

Nested classes/interfaces inherited from class java.awt.Window
java.awt.Window.AccessibleAWTWindow

Nested classes/interfaces inherited from class java.awt.Container
java.awt.Container.AccessibleAWTContainer

Nested classes/interfaces inherited from class java.awt.Component
java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy

Field Detail

editmemb

```
public static javax.swing.JFrame editmemb
```

Constructor Detail

MembersListFrame

```
public MembersListFrame()
```

Creates new form MembersListFrame

Method Detail

main

```
public static void main(java.lang.String[] args)
```

Parameters:

`args` - the command line arguments

Class PrintUtilities

```
java.lang.Object
  └associationmanagment2.PrintUtilities
```

All Implemented Interfaces:

java.awt.print.Printable

```
public class PrintUtilities
extends java.lang.Object
implements java.awt.print.Printable
```

Constructor Detail

PrintUtilities

```
public PrintUtilities(java.awt.Component componentToBePrinted)
```

Method Detail

printComponent

```
public static void printComponent(java.awt.Component c)
```

Class RMembersListFrame

```
java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └java.awt.Window
              └java.awt.Frame
```

```
└ javax.swing.JFrame
    └ associationmanagment2.RMembersListFrame
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

---

```
public class RMembersListFrame
extends javax.swing.JFrame
```

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JFrame
javax.swing.JFrame.AccessibleJFrame

Nested classes/interfaces inherited from class java.awt.Frame
java.awt.Frame.AccessibleAWTFrame

Nested classes/interfaces inherited from class java.awt.Window
java.awt.Window.AccessibleAWTWindow

Nested classes/interfaces inherited from class java.awt.Container
java.awt.Container.AccessibleAWTContainer

Nested classes/interfaces inherited from class java.awt.Component
java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy

Field Detail

editmemb

```
public static javax.swing.JFrame editmemb
```

---

data

```
public static java.util.Vector data
```

Constructor Detail

RMembersListFrame

```
public RMembersListFrame()
```

Creates new form MembersListFrame

Method Detail

setData

```
public static void setData(java.util.Vector v)
```

---

main

```
public static void main(java.lang.String[] args)
```

Parameters:

`args` - the command line arguments

Class SearchMemberFrame

```
java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └java.awt.Window
              └java.awt.Frame
                  └javax.swing.JFrame
                      └associationmanagment2.SearchMemberFrame
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

---

```
public class SearchMemberFrame
extends javax.swing.JFrame
```

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JFrame
javax.swing.JFrame.AccessibleJFrame

Nested classes/interfaces inherited from class java.awt.Frame
java.awt.Frame.AccessibleAWTFrame

Nested classes/interfaces inherited from class java.awt.Window
java.awt.Window.AccessibleAWTWindow

Nested classes/interfaces inherited from class java.awt.Container
java.awt.Container.AccessibleAWTContainer

Nested classes/interfaces inherited from class java.awt.Component
java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy

Field Detail

memberDataJPanel1

public static MemberDataJPanel **memberDataJPanel1**

OldAm

public static java.lang.String **OldAm**

RMemberslist

public static javax.swing.JFrame **RMemberslist**

Constructor Detail

SearchMemberFrame

public **SearchMemberFrame**()

Creates new form testJFrame

Class SimpleNameSearchJFrame

java.lang.Object
  └java.awt.Component
    └java.awt.Container
      └java.awt.Window
        └java.awt.Frame
          └javax.swing.JFrame
            └**associationmanagment2.SimpleNameSearchJFrame**

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

---

public class **SimpleNameSearchJFrame**
extends javax.swing.JFrame

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JFrame
javax.swing.JFrame.AccessibleJFrame

Nested classes/interfaces inherited from class java.awt.Frame
java.awt.Frame.AccessibleAWTFrame

Nested classes/interfaces inherited from class java.awt.Window
java.awt.Window.AccessibleAWTWindow

Nested classes/interfaces inherited from class java.awt.Container
java.awt.Container.AccessibleAWTContainer

Nested classes/interfaces inherited from class java.awt.Component
java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy

Field Detail

NameTextField

```
public static javax.swing.JTextField NameTextField
```

OkButton

```
public javax.swing.JButton OkButton
```

ResultsjTable

```
public javax.swing.JTable ResultsjTable
```

SearchButton

```
public static javax.swing.JButton SearchButton
```

---

jLabel

```
public static javax.swing.JLabel jLabel
```

---

jScrollPane1

```
public javax.swing.JScrollPane jScrollPane1
```

---

jSeparator1

```
public javax.swing.JSeparator jSeparator1
```

Constructor Detail

SimpleNameSearchJFrame

```
public SimpleNameSearchJFrame()
```

Creates new form SimpleNameSearchJFrame

Method Detail

main

```
public static void main(java.lang.String[] args)
```

Parameters:

`args` - the command line arguments

---

clear

```
public static void clear()
```

Class TableDataJPanel

```
java.lang.Object
  └java.awt.Component
      └java.awt.Container
          └javax.swing.JComponent
              └javax.swing.JPanel
                  └associationmanagment2.TableDataJPanel
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible

---

```
public class TableDataJPanel
extends javax.swing.JPanel
```

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JPanel
javax.swing.JPanel.AccessibleJPanel

Nested classes/interfaces inherited from class javax.swing.JComponent
javax.swing.JComponent.AccessibleJComponent

Nested classes/interfaces inherited from class java.awt.Container
java.awt.Container.AccessibleAWTContainer

Nested classes/interfaces inherited from class java.awt.Component
java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy

Field Detail

Table

```
public javax.swing.JTable Table
```

Table1

```
public javax.swing.JTable Table1
```

Table2

```
public javax.swing.JTable Table2
```

Table3

```
public javax.swing.JTable Table3
```

---

jPanel1

```
public javax.swing.JPanel jPanel1
```

---

jPanel2

```
public javax.swing.JPanel jPanel2
```

---

jPanel3

```
public javax.swing.JPanel jPanel3
```

---

jPanel4

```
public javax.swing.JPanel jPanel4
```

---

jTabbedPane1

```
public javax.swing.JTabbedPane jTabbedPane1
```

Constructor Detail

TableDataJPanel

```
public TableDataJPanel()
```

Creates new form TableDataJPanel

ΠΑΡΑΡΤΗΜΑ Δ : Παρουσίαση Εφαρμογής Διαχείρισης Συλλόγου

Ανάπτυξη Εφαρμογής Διαχείρισης Συλλόγου

Μαρίνα Κουντάκη (ΑΜ: 1093)

Επιβλέπων καθηγητής : Μαλάμος Αθανάσιος
Επιτροπή Αξιολόγησης : Βασιλάκης Κωνσταντίνος
Μαλάμος Αθανάσιος
Παχουλάκης Ιωάννης

Ανάπτυξη Εφαρμογής Διαχείρισης Συλλόγου

- Η **Εφαρμογή Διαχείρισης Συλλόγου** ενσωματώνει εύχρηστες διαδικασίες για την **διαχείριση** των καθημερινών εργασιών **σωματείων**, **συλλόγων** ή **επαγγελματικών ενώσεων**.
- Με τη χρήση απλών και λειτουργικών εντολών αναζήτησης, επιτρέπεται η **διαχείριση πλήθους μελών** και η ανίχνευση συγκεκριμένων δεδομένων από την βάση. Οι εντολές αναζήτησης ενσωματώνουν απλά ή σύνθετα ερωτήματα με χρήση πολλαπλών κριτηρίων.

ΒΑΣΙΚΕΣ ΛΕΙΤΟΥΡΓΙΕΣ

- Καταγραφή του μητρώου μελών με πλήρη στοιχεία μελών.
- Διαχείριση στοιχείων μελών.
- Δυναμικά φίλτρα αναζήτησης.
- Συνδρομές και οικονομικά στοιχεία.
- Δημογραφικά στοιχεία μέλους.
- Διαχείριση ταμείου.
- Ταμειακές κινήσεις με αναλυτικά αποτελέσματα.
- Εκτυπώσεις Ταμειακής Κατάστασης Εσόδων-Εξόδων.
- Εκτυπώσεις Καρτέλας Μελών, Λίστας Μελών, Κινήσεων Μέλους.

Ανάπτυξη Εφαρμογής Διαχείρισης Συλλόγου

- **Αποτέλεσμα:** Με την εφαρμογή ο σύλλογος, έχει ένα σύστημα διαχείρισης και πληροφόρησης για όλες τις εργασίες του, τα στοιχεία που κρατούνται χειρόγραφα θα μειωθούν στο ελάχιστο και οι διαδικασίες που απαιτούν χρόνο και ιδιαίτερη προσοχή θα αυτοματοποιηθούν και θα γίνονται άμεσα και χωρίς λάθη.

Εργαλεία

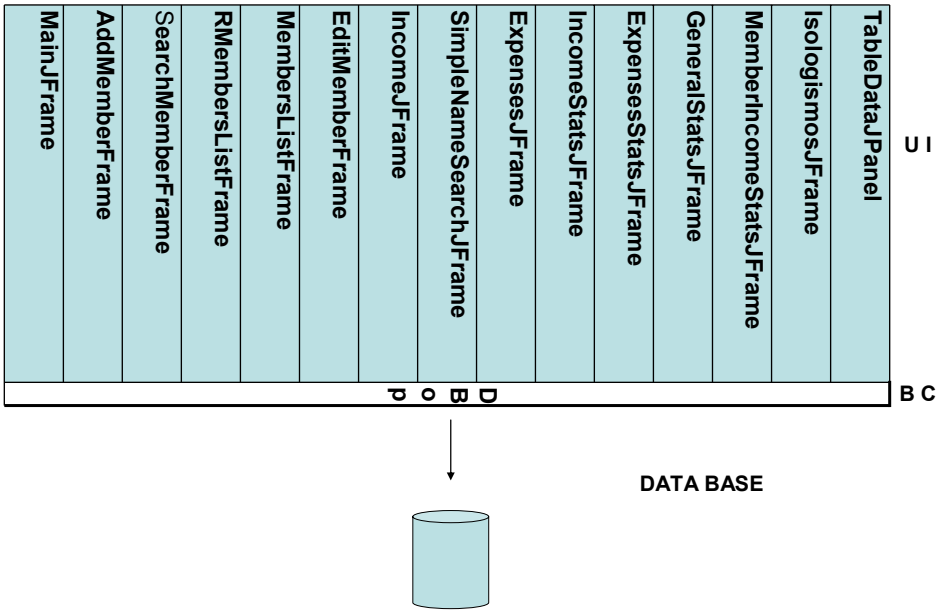
- Η ανάπτυξη έγινε σε Sun Java, γλώσσα αντικειμενοστραφή προγραμματισμού η οποία διατίθεται δωρεάν .
- Η βάση δεδομένων είναι σε MySQL την πιο δημοφιλή βάση δεδομένων ανοικτού κώδικα η οποία λειτουργεί σε περισσότερες από 20 πλατφόρμες.

Γιατί Java

Ένα από τα βασικά πλεονεκτήματα της Java έναντι των περισσότερων άλλων γλωσσών είναι η ανεξαρτησία του λειτουργικού συστήματος και πλατφόρμας. Τα προγράμματα που είναι γραμμένα σε Java τρέχουνε ακριβώς το ίδιο σε Windows, Linux, Unix και Macintosh χωρίς να χρειαστεί να ξαναγίνει μεταγλώττιση (compiling) ή να αλλάξει ο πηγαίος κώδικας για κάθε διαφορετικό λειτουργικό σύστημα.

Γιατί MySQL

Παρέχει απλοποίηση στην πρόσβαση, βελτίωση στις επιδόσεις, την επεκτασιμότητα και την αξιοπιστία και ικανοποίηση στην εκθετική αύξηση των απαιτήσεων σε αποθήκευση με δραματικά χαμηλότερο κόστος.

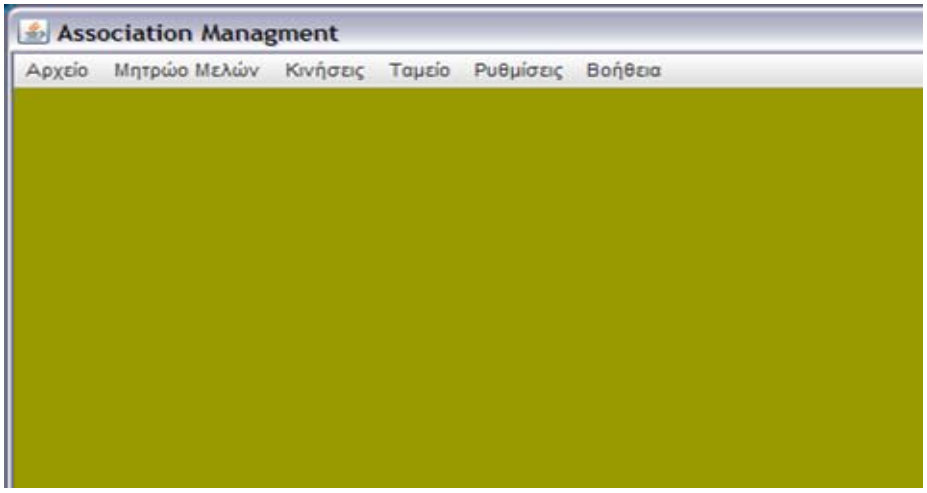


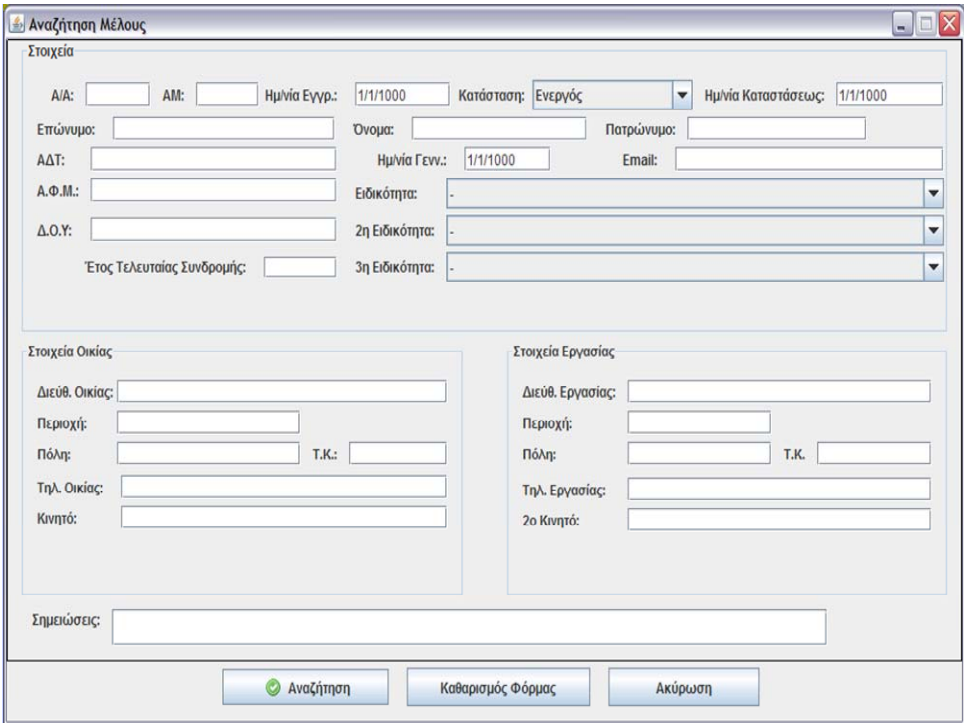
```
public static String getLastAA(String table) {
                                        //pairnei ta id apo ton pinaka table ta taxinomei me id
    String answer = new String();       // kai epistrefei to teleytaio id+1 gia nea kataxwrish
    ResultSet rs;

    rs = getRS("SELECT id FROM " + table + " ORDER BY id ");

    try {
       if (rs.first() == false) {       //Elegxei an o pinakas einai adeios
          return answer = "1";
       }
    } catch (SQLException ex) {
       Logger.getLogger(DBop.class.getName()).log(Level.SEVERE, null, ex);
    }

    try {
       rs.last();
       answer = String.valueOf(rs.getInt("ID") + 1);
    } catch (SQLException ex) {
       Logger.getLogger(DBop.class.getName()).log(Level.SEVERE, null, ex);
    }
    disconnect();
    return answer;
 }
```

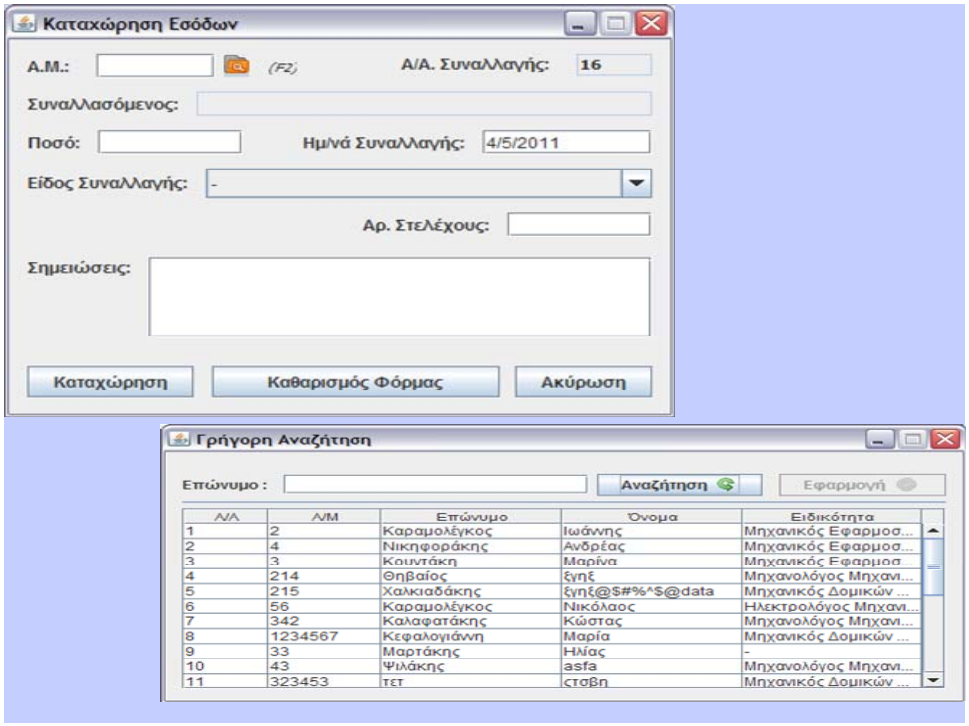
Απαιτήσεις

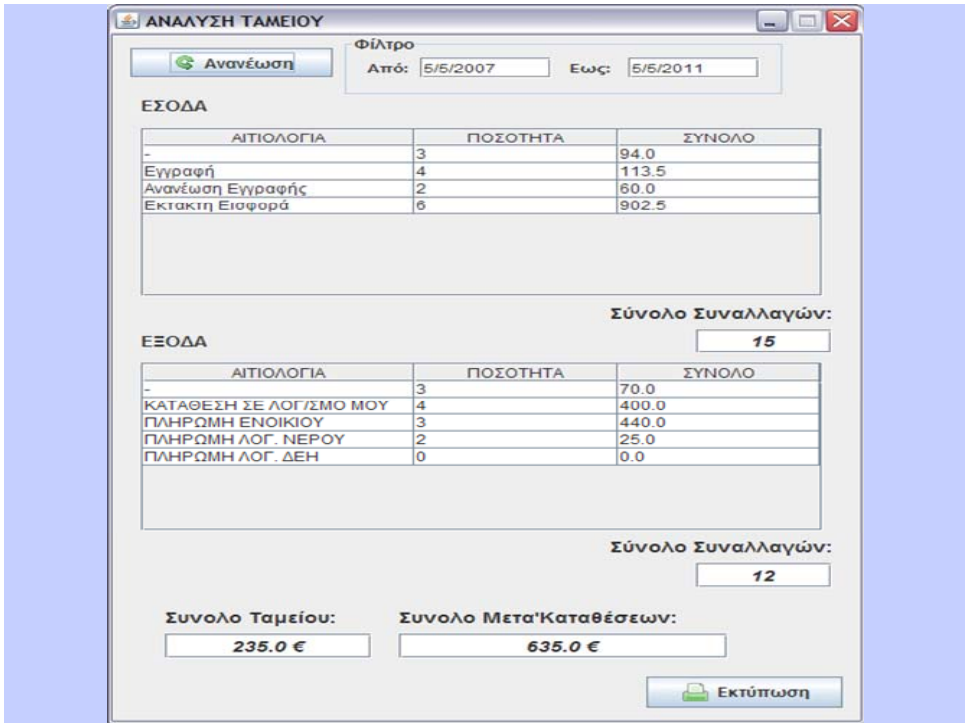
Ο υπολογιστής στον οποίο θα τρέχει η εφαρμογή πρέπει να έχει εγκατεστημένη Java και MySQL.

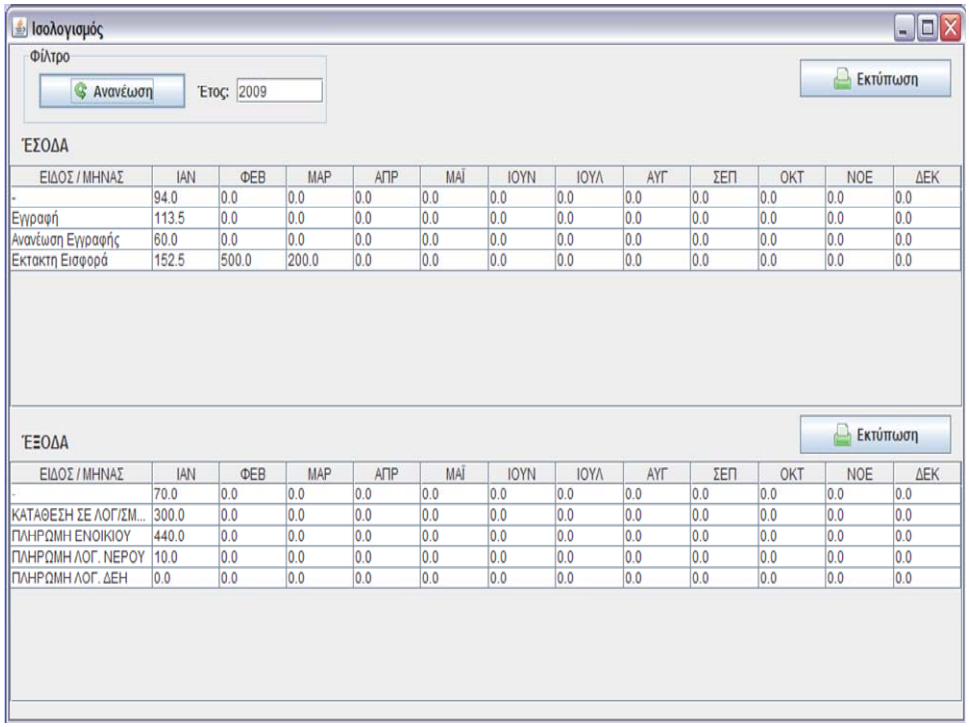
Συνίσταται η έκδοση του MySQL να είναι από 5.1 και άνω, ενώ η έκδοση της Java από 6 και άνω.


Association Managment
Αρχείο
Μητρώο Μελών
Κινήσεις
Ταμείο
Ρυθμίσεις
Βοήθεια


Αναζήτηση Μέλους
Στοιχεία
Α/Α:
ΑΜ:
Ημ/νία Εγγρ.:
1/1/1000
Κατάσταση:
Ενεργός
Ημ/νία Καταστάσεως:
1/1/1000
Επώνυμο:
Όνομα:
Πατρώνυμο:
ΑΔΤ:
Ημ/νία Γενν.:
1/1/1000
Email:
Α.Φ.Μ.:
Ειδικότητα:
Δ.Ο.Υ:
2η Ειδικότητα:
Έτος Τελευταίας Συνδρομής:
3η Ειδικότητα:
Στοιχεία Οικίας
Διεύθ. Οικίας:
Περιοχή:
Πόλη:
Τ.Κ.:
Τηλ. Οικίας:
Κινητό:
Στοιχεία Εργασίας
Διεύθ. Εργασίας:
Περιοχή:
Πόλη:
Τ.Κ.
Τηλ. Εργασίας:
2ο Κινητό:
Σημειώσεις:
Αναζήτηση
Καθαρισμός Φόρμας
Ακύρωση


Καταχώρηση Εσόδων
Α.Μ.:
(F2)
Α/Α. Συναλλαγής:
16
Συναλλασόμενος:
Ποσό:
Ημ/νά Συναλλαγής:
4/5/2011
Είδος Συναλλαγής:
-
Αρ. Στελέχους:
Σημειώσεις:
Καταχώρηση
Καθαρισμός Φόρμας
Ακύρωση
Γρήγορη Αναζήτηση
Επώνυμο :
Αναζήτηση
Εφαρμογή
Α/Α
Α/Μ
Επώνυμο
Όνομα
Ειδικότητα
1 2 Καραμολέγκος Ιωάννης Μηχανικός Εφαρμοσ...
2 4 Νικηφοράκης Ανδρέας Μηχανικός Εφαρμοσ...
3 3 Κουντάκη Μαρίνα Μηχανικός Εφαρμοσ
4 214 Θηβαίος ξγηξ Μηχανολόγος Μηχανι...
5 215 Χαλκιαδάκης ξγηξ@$#%^$@data Μηχανικός Δομικών ...
6 56 Καραμολέγκος Νικόλαος Ηλεκτρολόγος Μηχανι...
7 342 Καλαφατάκης Κώστας Μηχανολόγος Μηχανι...
8 1234567 Κεφαλογιάννη Μαρία Μηχανικός Δομικών ...
9 33 Μαρτάκης Ηλίας -
10 43 Ψιλάκης asfa Μηχανολόγος Μηχανι...
11 323453 τετ ετσβη Μηχανικός Δομικών ...


ΑΝΑΛΥΣΗ ΤΑΜΕΙΟΥ
Ανανέωση
Φίλτρο
Από: 5/5/2007
Εως: 5/5/2011
ΕΣΟΔΑ
ΑΙΤΙΟΛΟΓΙΑ ΠΟΣΟΤΗΤΑ ΣΥΝΟΛΟ
- 3 94.0
Εγγραφή 4 113.5
Ανανέωση Εγγραφής 2 60.0
Εκτακτη Εισφορά 6 902.5
Σύνολο Συναλλαγών:
15
ΕΞΟΔΑ
ΑΙΤΙΟΛΟΓΙΑ ΠΟΣΟΤΗΤΑ ΣΥΝΟΛΟ
- 3 70.0
ΚΑΤΑΘΕΣΗ ΣΕ ΛΟΓ/ΣΜΟ ΜΟΥ 4 400.0
ΠΛΗΡΩΜΗ ΕΝΟΙΚΙΟΥ 3 440.0
ΠΛΗΡΩΜΗ ΛΟΓ. ΝΕΡΟΥ 2 25.0
ΠΛΗΡΩΜΗ ΛΟΓ. ΔΕΗ 0 0.0
Σύνολο Συναλλαγών:
12
Συνολο Ταμείου:
235.0 €
Συνολο Μετα'Καταθέσεων:
635.0 €
Εκτύπωση



Ευχαριστώ

Μαρίνα Κουντάκη (ΑΜ: 1093)

Επιβλέπων καθηγητής : Μαλάμος Αθανάσιος
Επιτροπή Αξιολόγησης : Βασιλάκης Κωνσταντίνος
Μαλάμος Αθανάσιος
Παχουλάκης Ιωάννης