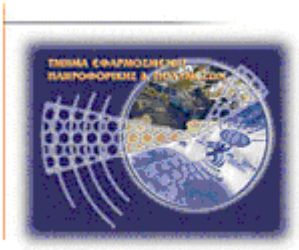
Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων



Πτυχιακή εργασία

ΈΝΑ ΕΥΦΥΕΣ ΣΥΣΤΗΜΑ ΠΡΟΒΛΕΨΗΣ ΤΙΜΩΝ ΜΕΤΟΧΩΝ

Σαρικάκη Αργυρώ (ΑΜ : 2660)

Φιλιππάκη Ιωάννα (ΑΜ: 2700)

Επιβλέπων καθηγητής: Παπαδάκης Νίκος

Ευχαριστίες

Θα θέλαμε να ευχαριστήσουμε όλους όσους στάθηκαν στο πλευρό μας όλο αυτό το χρονικό διάστημα. Πρώτα απ' όλα τις οικογένειές μας που μας στάθηκαν οικονομικά και ψυχολογικά παρέχοντάς μας την κατάλληλη στήριξη στις σπουδές μας. Θα θέλαμε να ευχαριστήσουμε επίσης όλο το κοινωνικό μας περίγυρο φίλους, συνάδελφους, διδάσκοντες που με τις γνώσεις τους και τη βοήθεια τους μας παρείχαν τα απαιτούμενα εφόδια για το μέλλον. Τέλος θα θέλαμε να αποδώσουμε τις ευχαριστίες μας στον εισηγητή και τον υπεύθυνο για την πτυχιακή μας εργασία κ. Νίκο Παπαδάκη για την ευκαιρία που μας έδωσε να ασχοληθούμε με το συγκεκριμένο θέμα.

Abstract:

The purpose of this paper was to create an intelligent stock prices forecasting system, using the java programming language and MySQL database. The latest was used in order to store the necessary information for each stock. The whole system offers to users the ability to directly being informed about the performance of stock of their interest. The following sections will introduce step by step the process of creating the aforementioned application and how it can be used.

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 analytically how to use the application and its components.

Περίληψη

Αντικείμενο της πτυχιακής εργασίας ήταν να κατασκευαστεί ένα ευφυΐες σύστημα πρόβλεψης τιμών μετοχών με την χρήση της γλώσσας προγραμματισμού java καθώς και την χρήση βάσης δεδομένων σε Mysql, για την αποθήκευση πληροφοριών για κάθε μετοχή. Το σύστημα θα προσφέρει στους χρήστες τη δυνατότητα να ενημερώνονται άμεσα για την απόδοση της μετοχής που τους ενδιαφέρει, καθώς και να έχουν πρόσβαση στις ιστορικές τιμές διαφόρων μεταβλητών όπως τιμή κλεισίματος, ανοίγματος κ.α.. Θα επεξηγήσουμε πως ακριβώς υπολογίζονται οι τιμές πρόβλεψης των μέτοχων χρησιμοποιώντας διάφορες τεχνικές τα αποτελέσματα των οποίων στην πραγματικότητα χρησιμοποιούνται για να υπολογιστεί η απόδοση της μετοχής. Στη συνέχεια θα παρουσιάσουμε βήμα προς βήμα τη διαδικασία δημιουργίας της εφαρμογής και πως μπορεί αυτή να χρησιμοποιηθεί περιγράφοντας αναλυτικά όλα τα σενάρια χρήσης.

Η ανάπτυξη έγινε σε Java, μια γλώσσα προγραμματισμού που παρέχει ανεξαρτησία από το λειτουργικό σύστημα και πλατφόρμας υλοποίησης σε συνδυασμό με την βάση δεδομένων 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 των βάσεων δεδομένων. Θα γίνει αναφορά στο δημοφιλές εργαλείο NetBeans καθώς με αυτό δημιουργήσαμε το σύστημα μας. Θα κάνουμε αναλυτική περιγραφή της χρήσης και των επιλογών της εφαρμογής αλλά και των δυνατοτήτων της.

Περιεχόμενα

1.Εισαγωγή 1

Στόχος συστήματος πρόβλεψης των μετοχών 2

2.Εισαγωγή στην Java 3

Η Java σαν Γλώσσα Προγραμματισμού 3

Η Java Πλατφόρμα 6

2.1 Εισαγωγή στον αντικειμενοστραφή προγραμματισμό 10

2.2 Δομή ενός προγράμματος σε Java 10

2.2.1 Τα αντικείμενα της Java 10

2.2.2 Ορισμός κλάσεων στην Java 11

2.2.3Τα ποιο σημαντικά χαρακτηριστικά του αντικειμενοστραφούς προγραμματισμού 12

2.3 Κατασκευαστές (constructors) 12

2.4 Καταστροφή αντικειμένων (Finalization) 12

2.5 Προσδιοριστές πρόσβασης (access specifiers) 13

2.6 Τροποποιητές (Modifiers) 14

2.7 Διασυνδέσεις (Interfaces) 14

2.8 Εξαιρέσεις (Exceptions) 15

2.9 Τα βασικά των Applets στην Java 15

2.10 Abstract Windowing Toolkit (AWT) 18

2.10.1 Βασικά στοιχεία ενός GUI 18

2.10.2 Δομή ενός GUI 19

2.11 Τα εργαλεία της Java 19

2.12 Βασικά στοιχεία ενός UML 20

2.13 Τι είναι το εργαλείο NetBeans 24

2.14 Τι είναι Xampp 25

3.Γενικές αρχές Βάσεων Δεδομένων 26

3.1 Εισαγωγή στις Βάσεις Δεδομένων 26

3.2 Τα Δεδομένα και οι Πληροφορίες 26
3.3 Η Οργάνωση Αρχείων 27
3.4 Προβλήματα της Οργάνωσης Αρχείων 28
3.5 Οι Βάσεις Δεδομένων και τα ΣΔΒΔ (DBMS) 28
3.5.1 Η Αρχιτεκτονική των ΣΔΒΔ 30
3.5.2 Οι Οντότητες (Entities) 30
3.5.3 Οι Ιδιότητες (Attributes) 30
3.5.4 Τα Στιγμιότυπα (Snapshots) 31
3.5.5 Το Πρωτεύον Κλειδί (Primary Key) 31
3.5.6 Οι Συσχετίσεις (Relationships) 31
3.5.7 Τα τρία βασικά μοντέλα 31
3.5.7.1 Το Ιεραρχικό Μοντέλο Βάσεων Δεδομένων 31
3.5.7.2 Το Δικτυωτό Μοντέλο Βάσεων Δεδομένων 32
3.5.7.3 Το Σχεσιακό Μοντέλο Βάσεων Δεδομένων 32
3.6 Τα Σχεσιακά ΣΔΒΔ (RDBMS) 32
3.7 Το Μοντέλο Οντοτήτων–Συσχετίσεων 34
3.7.1 Οι Οντότητες 34
3.7.2 Τα Κλειδιά 34
3.7.3 Οι Συσχετίσεις Μεταξύ Οντοτήτων 35
3.8 MySQL 36
3.8.1 Γιατί MySQL 36
3.9 Δημιουργία βάσεων δεδομένων 38
3.9.1 Δημιουργία πινάκων 39
3.9.2 Εισαγωγή δεδομένων 39
3.9.3 Ενημέρωση δεδομένων 39
3.9.4 Ερωτήματα 39
3.10 JDBC 41
3.10.1 Χρήση του Οδηγού 41

3.10.2 Επικοινωνία με τη Βάση 41
4.Μετοχές και Δείκτες Τεχνικής Ανάλυσης 42
4.1 Ανάλυση Μετοχών 42
4.2 Ερμηνεία δεικτών τεχνικής ανάλυσης 43
4.2.1 Τεχνική Ανάλυση Μετοχής 43
4.2.2 Τεχνικοί Δείκτες 44
4.2.3 Κινητός Μέσος Όρος 46
4.3 Κριτήρια για την αξιολόγηση της ακρίβειας των προβλέψεων 51
5.Ανάπτυξη Συστήματος Πρόβλεψης Τιμών Μετοχών 57
5.1 UML (Unified Modeling Language) – Class Diagrams 58
5.2 Εγχειρίδιο Χρήσης 74
5.3 Αξιολόγηση της ακρίβειας των προβλέψεων των μετοχών 92
6. Συμπεράσματα και Μελλοντική Εργασία 96
Βιβλιογραφία 97
ΠΑΡΑΡΤΗΜΑΤΑ 99
ΠΑΡΑΡΤΗΜΑ Α : ΕΡΓΑΛΕΙΑ ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΑΝ 100
Α1: Εγκατάσταση του Xampp 101
Α2: Εγκατάσταση JDK 104
Α3:Εγκατάσταση Netbeans 110
Α4: Εγκατάσταση MySQL Workbench 112
ΠΑΡΑΡΤΗΜΑ Β: ΧΡΗΣΙΜΕΣ ΒΙΒΛΙΟΘΗΚΕΣ 116
Β1:Εισαγωγή Mysql-Connector-java 5.1.22 117
Β2: Comma-separated Files (CSV) 118
Β3:Εισαγωγή του Jfreechart. 119
ΠΑΡΑΡΤΗΜΑ Γ: Χρήσιμες διευθύνσεις Internet για την Java 120
Γ1: Χρήσιμες Διευθύνσεις Internet για την Java 121
ΠΑΡΑΡΤΗΜΑ Δ : Παρουσίαση Εφαρμογής «Ένα Ευφυές Σύστημα Πρόβλεψης Τιμών Μετοχών» 122

Πίνακας Εικόνων

Εικόνα 1: Java Platform 6

Εικόνα 2: Java Application 6

Εικόνα 3: Επισκόπηση της διαδικασίας ανάπτυξης λογισμικού 7

Εικόνα 4: Μέσω της Java VM, η ίδια αίτηση μπορεί να λειτουργεί σε πολλαπλές πλατφόρμες. 8

Εικόνα 5: Το API και Java Virtual Machine μονώνουν το πρόγραμμα από το υποκείμενο υλικό. 9

Εικόνα 6: User interfaces – Β.Δ 10

Εικόνα 7: Παράδειγμα Συσχέτισης 23

Εικόνα 8: Σύστημα Β.Δ 28

Εικόνα 9: Σύστημα διαχείρισης βάσεων δεδομένων 29

Εικόνα 10: UML Diagram Εφαρμογής 58

Εικόνα 11: Class DBManager 60

Εικόνα 12: Class FiltersPanel 64

Εικόνα 13: Class DailyPrice 65

Εικόνα 14: Class StockMarket 66

Εικόνα 15: Class Stock 66

Εικόνα 16: Class DailyPriceModel 67

Εικόνα 17: Class DailyPricePanel 68

Εικόνα 18: Class StockModel 69

Εικόνα 19: Class StockPanel 70

Εικόνα 20: Class StockChartPanel 70

Εικόνα 21: Class StockMarketGUI 71

Εικόνα 22: Class Utils 72

Εικόνα 23: Class DailyPriceComparator 73

Εικόνα 24: Class Sector 73

Εικόνα 25: Μενού προγράμματος 74

Εικόνα 26: Επιλογή Create a new Sector 74

Εικόνα 27: Επιλογή Rename a selected Sector 74

Εικόνα 28: Επιλογή Delete selected Sector 75
Εικόνα 29: Stock Market 75
Εικόνα 30: Πίνακας DailyPrices 76
Εικόνα 31: DailyPrices 76
Εικόνα 32: Stocks 76
Εικόνα 33: Επιλογή Sector 77
Εικόνα 34: Επιλογή "φίλτρου" και χρονικής περιόδου 78
Εικόνα 35: Επιλογή Δείκτη Πρόβλεψης 78
Εικόνα 36: Επιλογή Δείκτη πρόβλεψης 78
Εικόνα 37: Επιλογή Χρονικού Διαστήματος 78
Εικόνα 38: Draw Button 79
Εικόνα 39: Εμφάνιση Γραφικής Παράστασης 79
Εικόνα 40: Εμφάνιση ποσοστών πρόβλεψης 79
Εικόνα 41: Zoom Γραφικής Παράστασης 80
Εικόνα 42: Μεγεθυμένη Γραφική Παράσταση 80
Εικόνα 43: Create Sector 81
Εικόνα 44: Εμφάνιση κατηγορίας 81
Εικόνα 45: Error message for Sector 81
Εικόνα 46: Error Message for delete Sector 82
Εικόνα 47: Error Message for rename Sector 82
Εικόνα 48: Inserting a new Stock for Selected Sector 82
Εικόνα 49: Message "Done" 83
Εικόνα 50: Stocks-Sectors 83
Εικόνα 51: Message for Stock 83
Εικόνα 52: Message for Rename stock 84
Εικόνα 53: Message for Delete stock 84
Εικόνα 54: Error Message for Stock 84
Εικόνα 55: Επιλογή Import CSV data to selected Stock 85

Εικόνα 56: Παράθυρο διαλόγου επιλογής CSV αρχείου 86

Εικόνα 57: Message Done for CSV files..... 86

Εικόνα 58: Εμφάνιση DailyPrices..... 86

Εικόνα 59: Επιλογή Συνάρτησης-Φίλτρου..... 87

Εικόνα 60: Επιλογή Συγκεκριμένου Φίλτρου 87

Εικόνα 61: Επιλογή Χρονικού Διαστήματος 87

Εικόνα 62: Εμφάνιση πρόβλεψης τιμών μετοχής 8887

Εικόνα 63: Simple moving average πρόβλεψη 88

Εικόνα 64: Επιλογή Συγκεκριμένου διαστήματος από τον πίνακα των DailyPrices 89

Εικόνα 65: Cumulative moving average πρόβλεψη 90

Εικόνα 66: Weighted moving average πρόβλεψη 90

Εικόνα 67: Exponential moving average value..... 91

Εικόνα 68: Exponential moving average πρόβλεψη 91

Λίστα Πινάκων

Πίνακας 1: Περιγραφή των κλάσεων για την κατασκευή GUI-components 18

Πίνακας 2: Περιγραφή των κλάσεων των Layout Managers...... 19

Πίνακας 3 Μέθοδος Initialize..... 62

1.Εισαγωγή

Η σύγχρονη ζωή έχει φέρει τους υπολογιστές και τις εφαρμογές τους σε πρώτο πλάνο. Τα λογισμικά πρέπει να δουλεύουν σωστά σε πολλά λειτουργικά περιβάλλοντα και σε πολλές αρχιτεκτονικές υπολογιστών. Πρέπει επίσης να εκμεταλλεύονται τις δυνατότητες των σύγχρονων υπολογιστικών συστημάτων, να είναι σε θέση να λαμβάνουν πληροφορίες, να τις επεξεργάζονται και να τις παρουσιάζουν στον χρήστη με όσο το δυνατό πιο εύχρηστο και κατανοητό τρόπο. Βεβαία μια σύγχρονη εφαρμογή θα πρέπει να έχει την δυνατότητα επέκτασης, ευελιξίας και ασφάλειας.

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 με την πάροδο του χρόνου 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. Ακόμα 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.

Στόχος συστήματος πρόβλεψης των μετοχών

Το Σύστημα πρόβλεψης τιμών μετοχών αποτελεί ένα λογισμικό διαχείρισης βάσης δεδομένων σε συνδυασμό με τον αντικειμενοστραφή προγραμματισμό. Είναι μία εφαρμογή η οποία έχει τη δυνατότητα πρόβλεψης τιμών μετοχών, η οποία θα μπορούσε να αποφέρει σημαντικά κέρδη στους επενδυτές τους.

Οι προβλέψεις είναι ένα βασικό στοιχείο των χρηματοπιστωτικών αγορών για το λόγο αυτό, η εφαρμογή έχει προσαρμοστεί στις ανάγκες των εκάστοτε επενδυτών που δραστηριοποιούνται στον χώρο του χρηματιστηρίου για την ορθή πρόβλεψη των τιμών και των αποδόσεων των μετοχών.

Είναι σημαντικό για έναν επενδυτή να παρακολουθεί από νωρίτερα τις τιμές των μετοχών και να παρατηρεί την πορεία της αγοράς, ούτως ώστε αν κινηθεί ανοδικά κατά τις αρχές του νέου έτους, να είναι σε θέση, με έγκαιρες τοποθετήσεις, να εκμεταλλευτεί την αναποτελεσματικότητα αυτή της αγοράς και να καρπωθεί τα οφέλη από την εμφάνιση του φαινομένου. Έτσι λοιπόν έγινε μια προσπάθεια, αφενός να διαπιστωθεί σε τι ποσοστό οι προβλέψεις που γίνονται είναι επιτυχείς και αφετέρου να διερευνηθεί αν η απόδοση των μετοχών είναι γραμμική συνάρτηση των προβλέψεων.

Επομένως, σαν πρωταρχικός στόχος τέθηκε η ανάπτυξη και εγκατάσταση μιας εφαρμογής που θα καλύπτει μηχανογραφικά τις ανάγκες μιας εταιρείας, μιας επιχείρησης ή ενός επενδυτή. Πιο συγκεκριμένα, την εισαγωγή μετοχών σε ένα σύστημα διαχείρισης και την παρακολούθηση αυτών, μέσα από ειδικού ελέγχους, ώστε να διαπιστωθεί αν μια μετοχή μπορεί να προσφέρει μελλοντικά κέρδη.

Ένας ακόμα στόχος ήταν να προσφερθεί αξιοπιστία. Αυτό επιτεύχθηκε ακολουθώντας μια σειρά από ελέγχους και μαθηματικούς υπολογισμούς. Τέλος, σημαντικός παράγοντας είναι η δημιουργία και παροχή δυνατοτήτων. Ο χρήστης πέρα από την παρακολούθηση των πληροφοριών επιθυμεί παραπάνω λειτουργικές δυνατότητες για παράδειγμα, η ομαδοποίηση μετοχών από πληθώρα χρηματιστηρίων ή επιλογή μετοχών συγκεκριμένου χρονικού διαστήματος και η προβολή στατιστικών δεδομένων και γραφημάτων.

Βασικές Λειτουργίες Εφαρμογής

1. Καταγραφή δεδομένων κάθε μετοχής Date, Open, High, Low, Close, Volume.
2. Καταγραφή μετοχών με το id και το όνομα της μετοχής.
3. Δημιουργία, μετονομασία, διαγραφή Sectors στην εφαρμογή.
4. Δημιουργία, μετονομασία, διαγραφή, εισαγωγή μετοχών στην εφαρμογή.
5. Επιλογή δεικτών τεχνικής ανάλυσης για την πρόβλεψη των τιμών των μετοχών.
6. Επιλογή συγκεκριμένου χρονικού διαστήματος για την πρόβλεψη τιμών των μετοχών.
7. Εμφάνιση γραφικής παράστασης με την πρόβλεψη τιμών των μετοχών.
8. Εμφάνιση ποσοστών αποτυχίας πρόβλεψης.

Αποτέλεσμα: Με την εφαρμογή οι επενδυτές που ασχολούνται με το χρηματιστήριο, έχουν στη διάθεση τους ένα σύστημα πρόβλεψης τιμών μετοχών που θα τους βοηθήσει μέσω μαθηματικών υπολογισμών σχετικών με την τιμή της μετοχής που χρησιμοποιείται στην προσπάθεια να εκτιμηθούν μελλοντικές αλλαγές στην τιμή της μετοχής. Η αλλαγή των προσδοκιών των επενδυτών συχνά είναι απότομη και βασίζεται σε ειδήσεις γύρω από την εταιρεία. Όταν η μετοχή είναι σε ανοδική πορεία οι αγοραστές είναι πιο πολλοί από τους πωλητές. Όταν η μετοχή είναι σε καθοδική πορεία αυτοί που πουλάνε είναι πιο πολλοί από αυτούς που αγοράζουν.

2.Εισαγωγή στην Java

Η γλώσσα προγραμματισμού Java είναι ένα προϊόν της Sun Microsystems Inc.. Ξεκίνησε σαν μέρος ενός μεγαλύτερου σχεδίου που αφορούσε την ανάπτυξη λογισμικού για καταναλωτικά ηλεκτρονικά. Πρόκειται για μικρές, αξιόπιστες, φορητές, πραγματικού χρόνου συσκευές που στην αρχή βασιζόντουσαν στην C++. Αρκετά προβλήματα όμως παρουσιάστηκαν και η γλώσσα C++ δεν μπόρεσε να εφαρμοστεί τελικά. Χρειάστηκε να αναπτυχθεί μία νέα γλώσσα: η Java. Η Java, στην τελική της μορφή, βρήκε περαιτέρω εφαρμογή στην επίλυση μερικών προβλημάτων του σημερινού προγραμματισμού, όπως animation, την αλληλεπίδραση πραγματικού χρόνου (real-time interaction) και την εξερεύνηση του Web (Web browsing).

Η Java είναι δύο πράγματα: (α) γλώσσα προγραμματισμού και (β) πλατφόρμα.

Η Java σαν Γλώσσα Προγραμματισμού

Η Java χαρακτηρίζεται από τα εξής: απλή, αντικειμενοστραφής, συμβατή με δικτυακά πρωτόκολλα, ουδέτερη της υποκείμενης αρχιτεκτονικής, φορητή, ασφαλής, υψηλής απόδοσης, δυναμική, σταθερή, interpreted και multithreaded. Στις ακόλουθες παραγράφους θα αναλύσουμε καθεμία από αυτές τις έννοιες.

Απλή

Στόχος της ομάδας της Sun που ανέπτυξε την Java, ήταν μια γλώσσα εύκολη στην χρήση, που δεν απαιτεί πολλή εξάσκηση και εκπαίδευση. Οι περισσότεροι προγραμματιστές στις μέρες μας δουλεύουν είτε με την C είτε με την C++. Έτσι, μολονότι η C++ δεν ήταν η κατάλληλη για το αρχικό σχέδιο, η Java σχεδιάστηκε βάσει της C++, με σκοπό να γίνει όσο το δυνατόν περισσότερο κατανοητή.

Η Java παραλείπει πολλά από τα σπανίως χρησιμοποιούμενα και δυσκολονόητα χαρακτηριστικά της C++, που δεν ωφελούν και πολύ την ευελιξία της γλώσσας. Προστέθηκαν διεργασίες, όπως η αυτόματη συλλογή των "σκουπιδιών" (automatic garbage collection), διευκολύνοντας τον προγραμματισμό σε Java. Μια κοινή πηγή πολυπλοκότητας της C++ και της C είναι η διαχείριση της μνήμης. Με την καινούργια διεργασία της αυτόματης συλλογής "σκουπιδιών", που συνιστάται από την περιοδική αποδέσμευση της μνήμης που δεν χρησιμοποιείται, μεγάλο μέρος από την δουλεία των προγραμματιστών αυτοματοποιείται και μειώνονται τα bugs.

Ένα πλεονέκτημα της Java που οφείλεται στην απλότητα της είναι ότι το μέγεθος των απαραίτητων εργαλείων. Ο Java interpreter και οι βασικές βιβλιοθήκες είναι μικρές και ο κώδικάς της Java είναι τόσο περιορισμένος σε μέγεθος που μπορεί άνετα να τρέξει σε οποιαδήποτε μικρή μηχανή και να κατέβει από το δίκτυο.

Αντικειμενοστραφής

Λέγοντας ότι μία γλώσσα προγραμματισμού είναι αντικειμενοστραφής, εννοούμε ότι η τεχνική σχεδιασμού ενός προγράμματος συγκεντρώνεται σε αντικείμενα. Ένα αντικείμενο είναι ο συνδυασμός δεδομένων, διαδικασιών και λειτουργιών με βασική ιδιότητα την απόκρυψη του συνδυασμού αυτού. Το κάθε αντικείμενο, δηλαδή, αντιμετωπίζεται σαν ένα "μαύρο κουτί". Τα αντικείμενα δεν είναι ανεξάρτητα μεταξύ τους, αλλά βρίσκονται σε σχέση αλληλεξάρτησης με τα υπόλοιπα. Υπάρχει η έννοια της κληρονομικότητας μεταξύ των αντικειμένων, δηλαδή ένα αντικείμενο μπορεί να κληρονομήσει δεδομένα από άλλα αντικείμενα.

Οι γλώσσες αντικειμενοστραφή προγραμματισμού είναι γλώσσες υψηλού επιπέδου, αφαιρετικές, αποτελεσματικές, γρήγορες και χρησιμοποιούνται για την δημιουργία μεγάλων και σημαντικών εφαρμογών. Οι αντικειμενοστραφής ευκολίες της Java είναι ίδιες με αυτές της C++, με επεκτάσεις από την Objectine C.

Συμβατή με Δίκτυα

Η Java έχει μια μεγάλη βιβλιοθήκη από ρουτίνες για την επιτυχημένη συνεργασία με τα πρωτόκολλα HTTP και FTP. Κατ 'αυτόν τον τρόπο, οι δικτυακές συνδέσεις δημιουργούνται ευκολότερα από ότι με την C ή την C++. Τα προγράμματα σε Java μπορούν να έχουν πρόσβαση μέσω δικτύου σε αντικείμενα , με την ίδια άνεση που ένας χρήστης προσπελάζει ένα τοπικό σύστημα αρχείων.

Σταθερή

Η Java προορίζεται για την σύνταξη προγραμμάτων που θα είναι αξιόπιστα από όλες τις πλευρές. Δίνεται έμφαση στον από νωρίς έλεγχο για πιθανά προβλήματα και στον έλεγχο σε πραγματικό χρόνο και στην εξάλειψη καταστάσεων που προκαλούν λάθη.

Η μεγαλύτερη διαφορά μεταξύ Java και C/C++ είναι το γεγονός ότι η Java έχει ένα μοντέλο δεικτών που εξαφανίζει την πιθανότητα της επαναχρησιμοποίησης της μνήμης και την καταστροφή των δεδομένων. Αντί για αριθμητικούς δείκτες (pointer arithmetic), η Java έχει πραγματικούς πίνακες (true arrays). Οι προγραμματιστές της Java δεν έχουν να φοβηθούν την ακούσια (ή μη) τροποποίηση της μνήμης, γιατί δεν υπάρχουν δείκτες (pointers). Εξάλλου, τα προγράμματα σε Java δεν μπορούν να αποκοτήσουν μη εγκεκριμένη πρόσβαση στην μνήμη.

Ασφαλής

Η Java προορίζεται για χρήση σε ανοικτά, δικτυωμένα περιβάλλοντα. Γι' αυτό το λόγο, ιδιαίτερη προσοχή έχει δοθεί στην ασφάλεια που παρέχει η γλώσσα. Η Java επιτρέπει την κατασκευή προγραμμάτων ελεύθερων από ιούς και η τροποποίηση τους είναι αδύνατη. Οι τεχνικές πιστοποίησης ταυτότητας βασίζονται στην ασύμμετρη κρυπτογραφία.

Υπάρχει μεγάλη σχέση μεταξύ του τρόπου διαχείρισης της μνήμης και της παρεχόμενης ασφάλειας. Αλλαγές στην σημασιολογία των δεικτών της μνήμης κάνουν αδύνατη την μη έγκυρη πρόσβαση στα δεδομένα της μνήμης ή της πρόσβασης των δεδομένων των αντικειμένων. Με αυτόν τον τρόπο καταπολεμούνται οι περισσότεροι ιοί.

Φορητή

Το γεγονός ότι είναι ανεξάρτητη της υποκείμενης πλατφόρμας αποτελεί μεγάλο μέρος του ότι είναι φορητή, άλλα υπάρχουν και άλλα σημεία που χαρακτηρίζουν την φορητότητα της.
Σε αντίθεση με την C/C++ δεν υπάρχουν καθόλου χαρακτηριστικά που εξαρτούνται από την CPU του υπολογιστή. Έτσι, τα μεγέθη των πρωταρχικών τύπων δεδομένων είναι καθορισμένα και η συμπεριφορά τους είναι παντού η ίδια. Για παράδειγμα, "int" σημαίνει πάντα έναν 32 bit ακέραιο και "float" πάντα αντιπροσωπεύει έναν 32 bit floating αριθμό.

Interpreted

Τα Java bytecodes μεταφράζονται σε πραγματικό χρόνο σε εντολές μηχανής που εξαρτώνται από την εκάστοτε πλατφόρμα, και δεν αποθηκεύονται πουθενά. Η διαδικασία είναι γρήγορη και πιο αποτελεσματική. Μαζί με τα bytecodes μεταφέρονται πληροφορίες που μπορούν να χρησιμοποιηθούν κατά την εκτέλεση και παρέχουν την βάση για τους ελέγχους που πραγματοποιεί ο συνδετής (linker). Επίσης τα προγράμματα γίνονται πιο επιδεκτικά σε debugging διαδικασίες.

Υψηλής Απόδοσης

Η διαδικασία παραγωγής των εντολών μηχανής είναι απλή και γρήγορη. Ο κώδικας που προκύπτει είναι αποτελεσματικός. Ο μεταγλωττιστής από την μεριά του εφαρμόζει αυτόματη κατανομή των καταχωρητών (automatic register allocation) όταν παράγει τα bytecodes. Η τελική μορφή του κώδικα (εκτελέσιμη δυαδική μορφή) είναι μικρή σε μέγεθος και ταχύτατη στην εκτέλεση.

Multithreaded

Τα προγράμματα σε Java έχουν την δυνατότητα να αντιμετωπίζουν πολλές καταστάσεις διαδικασίες ταυτόχρονα. Σε αντίθεση, η C και C++ είναι single-threaded γλώσσες. Τα πλεονεκτήματα του multithreading είναι η καλύτερη πραγματικού χρόνου συμπεριφορά και η καλύτερη αλληλεπιδραστική ανταπόκριση.

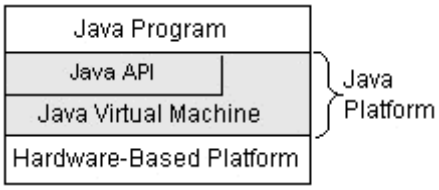
Δυναμική

Η Java είναι πιο δυναμική γλώσσα από την C ή C++. Έχει αναπτυχθεί για να προσαρμοστεί σε ένα εξελισσόμενο περιβάλλον. Οι βιβλιοθήκες εργαλείων αναπτύσσονται ελεύθερα με την πρόσθεση νέων μεθόδων και μεταβλητών, χωρίς να επηρεάζονται οι ήδη υπάρχουσες εφαρμογές.

Η Java Πλατφόρμα

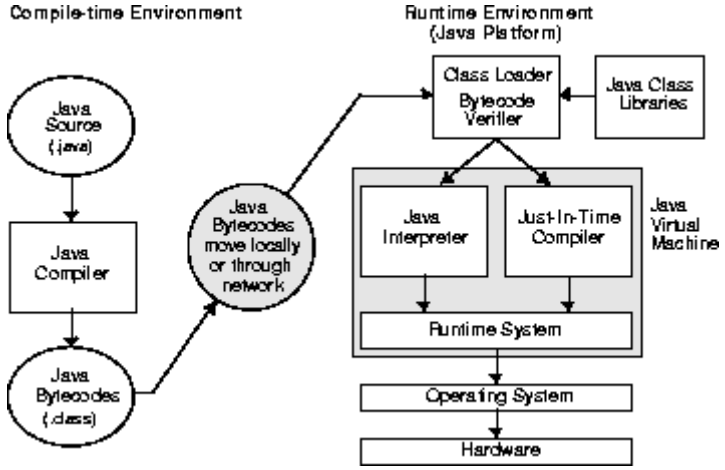
Πλατφόρμα είναι το hardware ή software περιβάλλον όπου τρέχει ένα πρόγραμμα. Η Java πλατφόρμα διαφέρει από τις άλλες πλατφόρμες, γιατί είναι μία software-only πλατφόρμα που τρέχει πάνω από άλλες hardware πλατφόρμες. Οι υπόλοιπες πλατφόρμες περιγράφονται σαν συνδυασμός hardware και software.
Η Java πλατφόρμα έχει δύο στοιχεία: **την Java Virtual Machine (JVM)** και **το Java Application Programming Interface (Java API)**.

Java Virtual Machine (JVM): Η Java παρέχει την δυνατότητα "write once, run everywhere" μέσω της JVM. Η JVM εφαρμόζεται πάνω από το λειτουργικό σύστημα της μηχανής και τα προγράμματα σε Java τρέχουν πάνω από την virtual machine. Σκοπός της JVM είναι η απομόνωση του προγράμματος από τις διαφορές μεταξύ των υποκείμενων λειτουργικών συστημάτων και CPUs. Η JVM ήταν αρχικά διαθέσιμη σε Web browsers και αυτήν την στιγμή υπάρχουν εκδόσεις για τα περισσότερα λειτουργικά συστήματα, όπως Windows 3.1/95/98/NT, MacOS και OS/2 Warp.



Εικόνα 1: Java Platform

Java Application Programming Interface (Java API): Είναι μια συλλογή από έτοιμα λογισμικά εργαλεία που προσφέρουν πολλές χρήσιμες δυνατότητες (π.χ. Graphical User Interface GUI). Το Java API είναι ομαδοποιημένο σε βιβλιοθήκες συσχετιζόμενων εργαλείων.
Τα προγράμματα σε Java, λόγω της συμβατότητας τους με όλα είδη υπολογιστών, μπορεί να είναι λίγο πιο αργά σε εκτέλεση από ότι τα προγράμματα σε τοπική γλώσσα μηχανής. Η χρήση, όμως, just-in-time-compilers εξαλείφει και αυτό το πρόβλημα. Παρακάτω βλέπουμε το ολοκληρωμένο μοντέλο της Java πλατφόρμας.



Εικόνα 2: Java Application

Χαρακτηριστικά της Γλώσσας

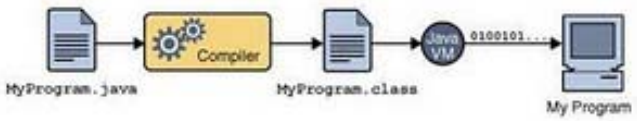
Η Java έχει αρκετά χαρακτηριστικά που προστατεύουν την ασφάλεια και την ακεραιότητα του συστήματος και που αποτρέπουν μερικές συνηθισμένες επιθέσεις. Τα Java προγράμματα δεν μπορούν να ορίζουν τους δικούς τους δείκτες στην μνήμη του συστήματος, ούτε έχουν απευθείας

πρόσβαση στην φυσική μνήμη. Η γλώσσα ελέγχει όλες τις ενέργειες του applet ώστε να μην μπορεί να του επιτρέπεται να δημιουργήσει τους δικούς του class loader ή security managers. Έχει, επιπλέον, ειδικούς ελέγχους για κακομεταχείριση δεικτών και μνήμης.

Αρχιτεκτονική

Η Java έχει σχεδιαστεί για να υποστηρίζει δικτυακές εφαρμογές. Ένα δίκτυο, όμως, αποτελείται από ποικιλία διαφορετικών συστημάτων, με διαφορετικές CPU και λειτουργικά συστήματα. Για να μπορούν οι Java εφαρμογές να εκτελούνται παντού στο δίκτυο, το πρόγραμμα Java πρέπει να περάσει από δύο διαδικασίες ώστε να καταλήξει σε εκτελέσιμη μορφή. Πρώτα ο μεταγλωττιστής, μετατρέπει τον πηγαίο κώδικα του προγράμματος σε μία ενδιάμεση γλώσσα που καλείται Java bytecodes. Τα Java bytecodes είναι ανεξάρτητα της πλατφόρμας και με χρήση του ερμηνευτή (interpreter) κάθε bytecode εντολή μετατρέπεται σε κατάλληλη δυαδική μορφή για να τρέξει στον εκάστοτε υπολογιστή. Η μεταγλώττιση (compilation) συμβαίνει μόνο μια φορά για κάθε Java πρόγραμμα, η ερμηνεία (interpretation) γίνεται κάθε φορά που το πρόγραμμα εκτελείται. Το παρακάτω σχήμα επιδεικνύει πως λειτουργεί αυτή η φιλοσοφία.

Τα Java bytecodes μπορούμε να τα φανταστούμε σαν την γλώσσα μηχανής για την Java Virtual Machine (JVM). Κάθε Java ερμηνευτής (π.χ. ένας Web browser που μπορεί να τρέχει applets) είναι μια λογισμική εφαρμογή του της Java Virtual Machine. Η JVM αναλαμβάνει να μετατρέψει τα bytecodes σε κατάλληλη εκτελέσιμη μορφή, ανάλογα με το υποκείμενο software και hardware.
Η τεχνική που περιγράφηκε παραπάνω καλείται "write once, run anywhere". Το Java πρόγραμμα μεταγλωττίζεται μία φορά σε Java bytecodes με τον μεταγλωττιστή της Java. Έπειτα, τα bytecodes μπορούν να τρέξουν σε οποιαδήποτε μηχανή που έχει μία εφαρμοσμένη JVM (ο ερμηνευτής).



Εικόνα 3: Επισκόπηση της διαδικασίας ανάπτυξης λογισμικού

Η 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

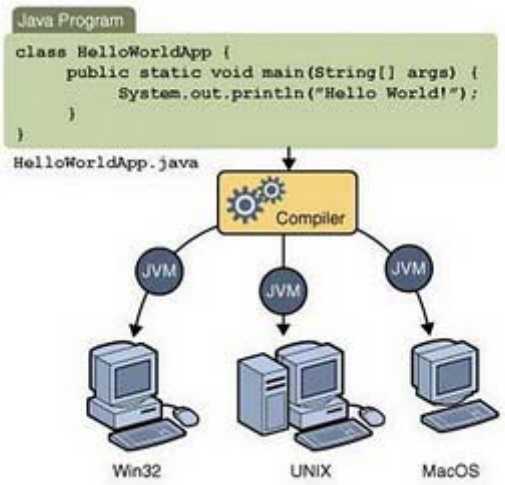
προγράμματα. Το JRE περιέχεται μέσα στον compiler ή καλύτερα στο JDK πακέτο το οποίο έχουμε κατεβάσει έτσι ώστε να έχουμε την δυνατότητα και να γράψουμε αλλά και να τρέξουμε τα προγράμματα μας.

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.



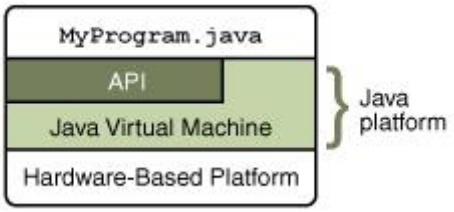
Εικόνα 4: Μέσω της Java VM, η ίδια αίτηση μπορεί να λειτουργεί σε πολλαπλές πλατφόρμες.

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 όπως αναφέραμε και προηγουμένως 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

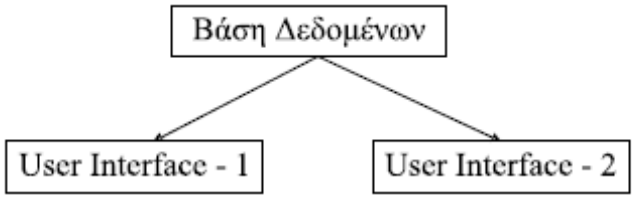
http://java.sun.com/docs/books/tutorial/getStarted/intro/definition.html



Εικόνα 5: Το API και Java Virtual Machine μονώνουν το πρόγραμμα από το υποκείμενο υλικό.

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).



Εικόνα 66: User interfaces – Β.Δ

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.

Παρατηρήσεις: 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.
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>
κοκ.

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.

Σημείωση: 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).

2.2.3Τα ποιο σημαντικά χαρακτηριστικά του αντικειμενοστραφούς προγραμματισμού

Encapsulation()

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

Polymorphism()

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.

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 ξανά 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 1: 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: 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 Βασικά στοιχεία ενός UML

Τυπικός ορισμός:

Η Ενοποιημένη Γλώσσα Μοντελοποίησης (Unified Modeling Language, στο εξής UML) είναι μία γραφική γλώσσα γενικού σκοπού, η οποία χρησιμοποιείται για τον προσδιορισμό, οπτικοποίηση, ανάπτυξη και τεκμηρίωση των κατασκευασμάτων (artifacts) ενός συστήματος λογισμικού.

Πρακτικά:

Η UML είναι μία γλώσσα μοντελοποίησης (σύνολο από διαγράμματα). (Δεν είναι γλώσσα προγραμματισμού)

Χρήση:

με οποιαδήποτε διαδικασία
σε οποιοδήποτε στάδιο ανάπτυξης
για οποιαδήποτε τεχνολογική πλατφόρμα

Μοντελοποίηση

- Ένα μοντέλο είναι μία αναπαράσταση σε ένα συγκεκριμένο μέσο, κάποιας οντότητας στο ίδιο ή σε άλλο μέσο.
- Το μοντέλο αναπαριστά τα σημαντικά (από μία ορισμένη οπτική γωνία), στοιχεία της οντότητας που μοντελοποιείται, απλοποιώντας ή παραλείποντας τα υπόλοιπα.
- Το μέσο επάνω στο οποίο αναπτύσσεται το μοντέλο είναι βολικό για επεξεργασία.

Πλεονεκτήματα ενός μοντέλου

- Ακριβής καθορισμός των απαιτήσεων έτσι ώστε όλοι οι εμπλεκόμενοι να τις κατανοούν με κοινό τρόπο. Μοντέλα ενός συστήματος λογισμικού κατασκευάζονται ώστε οι αναλυτές, προγραμματιστές, διαχειριστές έργων, πελάτες και τελικοί χρήστες να κατανοήσουν το σύστημα από την οπτική γωνία που τους ενδιαφέρει.
- Μελέτη του τρόπου σχεδίασης. Το μοντέλο ενός συστήματος λογισμικού παρέχει τη δυνατότητα διερεύνησης διαφόρων αρχιτεκτονικών, με οικονομικό τρόπο, αρκετά πριν από την κωδικοποίηση.
- Παραγωγή χρήσιμων πρωτοτύπων. Το μοντέλο ενός συστήματος λογισμικού μπορεί να χρησιμοποιηθεί για την παραγωγή πρωτοτύπων τα οποία με τη σειρά τους θα αξιοποιηθούν για τη διερεύνηση των απαιτήσεων.
- Διαχείριση περίπλοκων συστημάτων. Τα σύγχρονα συστήματα λογισμικού χαρακτηρίζονται από υψηλή πολυπλοκότητα όσον αφορά το μέγεθος τους, τον αριθμό των λειτουργιών τους, τον αριθμό των μονάδων που τα απαρτίζουν, το μέγεθος της ομάδας ανάπτυξης κ.ο.κ. Ένα μοντέλο μπορεί να συμβάλει στην αντιμετώπιση της πολυπλοκότητας εισάγοντας διάφορα επίπεδα αφαίρεσης, αποκρύπτοντας τον τεράστιο όγκο των λεπτομερειών.

UML

Η UML αποτυπώνει τόσο τη στατική δομή, όσο και τη δυναμική συμπεριφορά ενός συστήματος. Ένα αντικειμενοστραφές σύστημα μοντελοποιείται ως μία συλλογή αντικειμένων που αλληλεπιδρούν για την εκτέλεση μίας λειτουργίας η οποία είναι τελικά αξιοποιήσιμη από τον χρήστη του συστήματος.

Η στατική δομή καθορίζει τα είδη των αντικειμένων που είναι σημαντικά για το σύστημα καθώς και τις συσχετίσεις μεταξύ τους. Η δυναμική συμπεριφορά προσδιορίζει την εξέλιξη των αντικειμένων σε σχέση με τον χρόνο και την επικοινωνία μεταξύ τους.

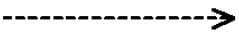
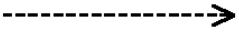
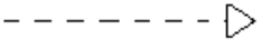
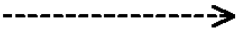
Περιπτώσεις Χρήσης

- Τα διαγράμματα περιπτώσεων χρήσης στη UML χρησιμοποιούνται για τη μοντελοποίηση της συμπεριφοράς ενός συστήματος, υποσυστήματος ή κλάσης, όπως αυτή γίνεται αντιληπτή από τον εξωτερικό χρήστη.
- Τα διαγράμματα περιπτώσεων χρήσης διαμερίζουν τη λειτουργικότητα του συστήματος σε συναλλαγές που έχουν νόημα για τους χαρακτήρες (actors) - ιδανικούς χρήστες του συστήματος. Τα επιμέρους τμήματα της λειτουργικότητας ονομάζονται περιπτώσεις χρήσης.
- Το σύνολο των περιπτώσεων χρήσης συνιστούν τη συμπεριφορά του συστήματος. Ο τυπικός ορισμός μιας περίπτωσης χρήσης είναι μία ακολουθία συναλλαγών που πραγματοποιείται από το σύστημα για την παραγωγή μετρήσιμων αποτελεσμάτων που έχουν νόημα για τον χρήστη.
- Σε κάθε διάγραμμα περίπτωσης χρήσης απεικονίζεται ένας χρήστης του συστήματος (άνθρωπος ή άλλο σύστημα) ως ένα σχηματικό ανθρωπάκι (stick person).
- Η ίδια η περίπτωση χρήσης, ως σύνολο λειτουργιών που έχουν κάποιο νόημα για το χρήστη, απεικονίζεται ως μία έλλειψη.
- Ο χρήστης ενεργοποιεί μία περίπτωση χρήσης αναμένοντας την εκτέλεση κάποια συμπεριφοράς. Η συσχέτιση μεταξύ χρήστη και περίπτωσης χρήσης απεικονίζεται με μία ακμή μεταξύ τους ενώ η φορά της ενεργοποίησης με τη χρήση προσανατολισμένης ακμής.
- Η απόφαση σχετικά με τις απαιτήσεις που πρόκειται να υλοποιηθούν στο σύστημα, είναι αρμοδιότητα και δικαίωμα των πελατών
- Ένας χαρακτήρας που αλληλεπιδρά με το σύστημα υπό διαφορετικό ρόλο κάθε φορά, αναγνωρίζεται ως διαφορετικός χρήστης. (Π.χ. ένας καθηγητής μπορεί να λειτουργεί είτε ως καθηγητής (Faculty) είτε ως πρόεδρος τμήματος).
- Δράσεις που ενεργοποιούνται από το ίδιο το σύστημα δεν καταγράφονται ως περιπτώσεις χρήσης
- Ο ορισμός μιας περίπτωσης χρήσης περιλαμβάνει όλες τις δυνατές συμπεριφορές που εμπερικλείει (κανονική και εναλλακτική συμπεριφορά)
- Η άποψη του συστήματος βάσει ενός μοντέλου περιπτώσεων χρήσης θεωρεί ότι η εκτέλεση κάθε περίπτωσης χρήσης είναι ανεξάρτητη από τις υπόλοιπες (ορθογωνικότητα), παρόλο που η υλοποίηση του συστήματος μπορεί να συνεπάγεται έμμεσες εξαρτήσεις μεταξύ τους λόγω κοινών αντικειμένων.
- Η αναλυτική περιγραφή μιας περίπτωσης χρήσης προσδιορίζεται με άλλους τύπους διαγραμμάτων της UML όπως τα διαγράμματα καταστάσεων, διαγράμματα αλληλεπίδρασης (διαγράμματα ακολουθίας ή/και διαγράμματα συνεργασίας) ή και άτυπες περιγραφές κειμένου.

Στατική Άποψη

- Η στατική άποψη ενός μοντέλου είναι θεμελιώδης στη UML καθώς αποτυπώνει την αρχιτεκτονική του συστήματος (μονάδες + μεταξύ τους σχέσεις)
- Σε ένα αντικειμενοστραφές σύστημα τα δομικά του στοιχεία είναι οι κλάσεις και οι σχέσεις μεταξύ των κλάσεων επιτρέπουν τη συνεργασία των αντικειμένων τους
- Τα διαγράμματα κλάσεων αποτυπώνουν τη στατική δομή
- Πολύ συχνά, τα διαγράμματα κλάσεων είναι το μόνο είδος διαγραμμάτων που χρησιμοποιείται λόγω των πληροφοριών που παρέχει σχετικά με τον κώδικα

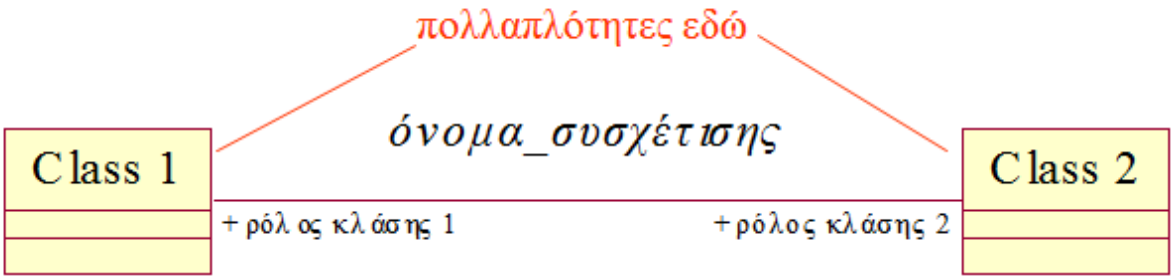
Σχέσεις

Σχέση	***Λειτουργία***	***Συμβολισμός***
Συσχέτιση	Περιγραφή μιας διασύνδεσης μεταξύ αντικειμένων	
Εξάρτηση	Μία σχέση εξάρτησης μεταξύ δύο στοιχείων ενός μοντέλου	
Ροή	Μία συσχέτιση μεταξύ δύο εκδόσεων ενός αντικειμένου σε διαδοχικές χρονικές στιγμές	
Γενίκευση	Μία συσχέτιση μεταξύ μιας γενικής περιγραφής και μιας ειδικότερης περίπτωσης - υποδηλώνει κληρονομικότητα	
Υλοποίηση	Σχέση μεταξύ μιας προδιαγραφής και της υλοποίησής της	
Χρήση	Μία κατάσταση κατά την οποία ένα στοιχείο του μοντέλου απαιτεί την ύπαρξη κάποιου άλλου για την ορθή λειτουργία του	

Συσχέτιση (Association)

Οι συσχετίσεις παρέχουν τις διασυνδέσεις βάσει των οποίων τα αντικείμενα διαφόρων κλάσεων μπορούν να αλληλεπιδρούν και να ανταλλάσσουν μηνύματα.
Οι συσχετίσεις είναι η "κόλλα" του συστήματος: χωρίς συσχετίσεις υπάρχουν μόνο ανεξάρτητες κλάσεις που δεν συνεργάζονται.

- Κάθε συσχέτιση έχει δύο άκρα τα οποία είναι δυνατόν να ονομαστούν ώστε να προσδιορίζουν ρόλους.
- Η σημαντικότερη ιδιότητα των άκρων είναι η πολλαπλότητα (multiplicity) που υποδηλώνει τον αριθμό των στιγμιοτύπων μιας κλάσης που μπορούν να συσχετιστούν με ένα στιγμιότυπο μιας άλλης κλάσης.
- Μία συσχέτιση συμβολίζεται με μία συνεχή γραμμή που συνδέει τις κλάσεις που συμμετέχουν.



Εικόνα 7: Παράδειγμα Συσχέτισης

Πολλαπλότητες

*	Οποιοσδήποτε αριθμός αντικειμένων
1	Ακριβώς ένα αντικείμενο
n	Ακριβώς n αντικείμενα (όπου n ακέραιος)
0..1	Μηδέν ή ένα αντικείμενα (υποδηλώνει ότι η συσχέτιση είναι προαιρετική)
n..m	Εύρος τιμών με μικρότερη δυνατή τιμή n και μέγιστη τιμή m
n, m	Διακριτοί συνδυασμοί (π.χ. ένα αυτοκίνητο μπορεί να έχει 2 ή 4 πόρτες)

2.13 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.

Το NetBeans IDE είναι ένα περιβαλλοντικό ανάπτυγμα IDE - ένα εργαλείο στους προγραμματιστές για να γράψουν, να κάνουν compile, debug και να αναπτύξουν προγράμματα. Είναι γραμμένο σε Java - αλλά μπορεί να υποστηρίξει όλες τις γλώσσες προγραμματισμού. Υπάρχει επίσης ένας μεγάλος αριθμός υπομονάδων (modules) που βοηθάνε στην επέκταση της λειτουργικότητας του NetBeans IDE. Το NetBeans IDE είναι ένα ελεύθερο προϊόν δίχως περιορισμούς στον τρόπο χρησιμοποίησής του.

Διαθέσιμο επίσης είναι το NetBeans Platform ένα εκτατό θεμέλιο αποτελούμενο από υπομονάδες (modular) που χρησιμοποιείται σαν βάση λογισμικού για τη δημιουργία μεγάλων επιτραπέζιων (desktop) εφαρμογών. Οι ISV συνεργάτες διαθέτουν προσθήκες, επιπρόσθετα προγράμματα (pluins) που εύκολα συνενώνονται στο Platform και μπορούν επίσης να χρησιμοποιηθούν για την ανάπτυξη άλλων εργαλείων και λύσεων.

Και τα δύο τα προϊόντα είναι ανοιχτής πηγής (open source) και ελεύθερα για εμπορική ή μη χρήση. Ο κώδικας πηγής (source code) είναι διαθέσιμος για επαναχρησιμοποίηση κάτω από το Common Development and Distribution License (CDDL).

2.14 Τι είναι Xampp

- Το όνομα του Xampp είναι ένα ακρωνύμιο των:
- X (σημαίνει cross-platform=που λειτουργεί σε πολλές πλατφόρμες)
- Apache HTTP Server
- MySQL
- PHP
- Perl

Το XAMPP αποτελεί στην ουσία ένα πακέτο, το οποίο περιλαμβάνει τις τελευταίες εκδόσεις του Apache, της PHP και της MySQL, ενώ περιλαμβάνει επίσης και άλλα τρία χρήσιμα εργαλεία, που θα χρειαστούμε στην συνέχεια (PhpMyAdmin, Filezilla Server, Mercury Mail). Το XAMPP διατίθεται και αυτό δωρεάν από την σελίδα http://www.apachefriends.org για διάφορα λειτουργικά συστήματα (Linux, Windows, Solaris, Mac).

Αρχιτεκτονική

Η εφαρμογή που αναπτύχθηκε έχει σχεδιαστεί έχοντας ως βάση την μέγιστη δυνατή λειτουργικότητα. **Έτσι έχει προσεχθεί ώστε να είναι:**
ανεξάρτητη λειτουργικού συστήματος (δηλαδή μπορεί να εγκατασταθεί σε οποιοδήποτε λειτουργικό σύστημα), προσπελάσιμη μέσω οποιουδήποτε φυλλομετρητή (browser), πλήρως σπονδυλωτή στη δομή της, δηλαδή να χρησιμοποιεί αρθρώματα (modules) για τις διάφορες λειτουργίες που εκτελεί προσιτή στη διαχείριση από τον διδάσκοντα.

Οι παραπάνω ιδιότητες-στόχοι οδηγούν σε ένα σύστημα υλικού (H/W) και λογισμικού (S/W) το οποίο αποτελείται από: τη βάση δεδομένων που περιέχει όλες τις απαραίτητες πληροφορίες για τη λειτουργία του συστήματος, τα αποθηκευτικά μέσα, δηλαδή τους υπολογιστές που φιλοξενούν το υλικό του μαθήματος, το περιβάλλον εργασίας, δηλαδή το λογισμικό διεπαφής που επεξεργάζεται τις πληροφορίες και κάνει δυνατή την αλληλεπίδραση των χρηστών με το εκπαιδευτικό υλικό.

Η εφαρμογή είναι βασισμένη στο πρότυπο τύπου «πελάτη-εξυπηρετητήή» (client-server). Εγκαθίσταται σε οποιοδήποτε λειτουργικό σύστημα, υποστηρίζει web server τύπου Apache ή Microsoft IIS, ενώ στηρίζεται εξολοκλήρου σε περιβάλλοντα «ανοιχτού κώδικα» (open source) για την λειτουργία της. Για την ανάπτυξη του ιστογενούς περιβάλλοντος της εφαρμογής και των αλγορίθμων της χρησιμοποιήθηκε η γλώσσα PHP (Pre Hypertext Processor) .

3.Γενικές αρχές Βάσεων Δεδομένων

3.1 Εισαγωγή στις Βάσεις Δεδομένων

Η αλματώδης ανάπτυξη της επιστήμης της πληροφορικής και των επικοινωνιών τα τελευταία χρόνια έχει καταστήσει την πληροφορία ως ένα από τα βασικότερα και πολυτιμότερα αγαθά. Είναι κοινός τόπος σήμερα η εκτίμηση ότι το αγαθό της πληροφορίας είναι επιθυμητό απ' όλους τους εργαζόμενους αλλά και τους εκπαιδευόμενους, ώστε να είναι πιο αποδοτικοί, ανταγωνιστικοί αλλά και παραγωγικοί στην εργασία τους.

Τα συστήματα βάσεων δεδομένων τα χρησιμοποιούμε για να μπορούμε να αποθηκεύσουμε, να επεξεργαστούμε αλλά και να εκμεταλλευτούμε αποδοτικά αυτόν τον τεράστιο όγκο των πληροφοριών που αυξάνονται με αλματώδεις ρυθμούς καθημερινά.

3.2 Τα Δεδομένα και οι Πληροφορίες

Με τον όρο πληροφορία αναφερόμαστε συνήθως σε ειδήσεις, γεγονότα και έννοιες που αποκτάμε από την καθημερινή μας επικοινωνία και τα θεωρούμε ως αποκτηθείσα γνώση, ενώ τα δεδομένα μπορούν να είναι μη κατάλληλα επεξεργασμένα και μη ταξινομημένα σύνολα πληροφοριών. Ένας αυστηρός ορισμός για το τι είναι δεδομένα και τι είναι πληροφορία, σύμφωνα με την επιτροπή ANSI των ΗΠΑ, είναι ο εξής :

- Δεδομένα (data) είναι μια παράσταση, όπως γράμματα, αριθμοί, σύμβολα κ.ά. στα οποία μπορούμε να δώσουμε κάποια σημασία (έννοια).

- Πληροφορία (information) είναι η σημασία που δίνουμε σ' ένα σύνολο από δεδομένα, τα οποία μπορούμε να επεξεργαστούμε βάσει προκαθορισμένων κανόνων και να βγάλουμε έτσι κάποια χρήσιμα συμπεράσματα. Με τις πληροφορίες περιορίζεται η αβεβαιότητα που έχουμε για διάφορα πράγματα και βοηθιόμαστε έτσι στο να λάβουμε σωστές αποφάσεις.

Τα δεδομένα μπορούν να θεωρηθούν ως τρόποι αναπαράστασης εννοιών και γεγονότων που μπορούν να υποστούν διαχείριση και επεξεργασία. Η συλλογή και αποθήκευση ενός τεράστιου όγκου δεδομένων όπως απαιτούν οι κοινωνικές συνθήκες σήμερα, δεν λύνει τελείως το πρόβλημα της σωστής οργάνωσης και ταξινόμησης των δεδομένων. Τα δεδομένα θα πρέπει να οργανωθούν με τέτοιο τρόπο έτσι ώστε να μπορούμε να τα εντοπίζουμε και να τα αξιοποιούμε εύκολα και γρήγορα και τη στιγμή που τα χρειαζόμαστε.

Ένα κλασικό παράδειγμα μη σωστής οργάνωσης δεδομένων θα ήταν για παράδειγμα ο τηλεφωνικός κατάλογος της πόλης της Θεσσαλονίκης, όπου οι συνδρομητές δεν θα ήταν καταχωρημένοι αλφαβητικά σύμφωνα με το επώνυμο και το όνομά τους, αλλά εντελώς τυχαία. Ένας τέτοιος τηλεφωνικός κατάλογος θα περιείχε μια τεράστια ποσότητα δεδομένων αλλά θα ήταν ουσιαστικά άχρηστος.

Ένα άλλο παράδειγμα μη σωστής οργάνωσης δεδομένων θα ήταν μια πολύ μεγάλη βιβλιοθήκη με χιλιάδες τόμους βιβλίων και χωρίς να διαθέτει κάποιο υποτυπώδες σύστημα οργάνωσης και ταξινόμησης των βιβλίων. Ούτε οι υπάλληλοι της βιβλιοθήκης θα μπορούσαν να κάνουν τη δουλειά τους αλλά ούτε και οι επισκέπτες θα μπορούσαν να αξιοποιήσουν την πληθώρα των πληροφοριών που περιέχονται στα βιβλία. Εκτός λοιπόν από τη μόνιμη αποθήκευση των δεδομένων, χρειαζόμαστε και κάποιους τρόπους ευέλικτης και αποδοτικής οργάνωσής τους.

3.3 Η Οργάνωση Αρχείων

Ο πιο γνωστός τρόπος οργάνωσης δεδομένων με τη χρήση ηλεκτρονικών υπολογιστών είναι σε αρχεία εγγραφών.

Ένα αρχείο (file) θα μπορούμε να το χαρακτηρίσουμε σαν ένα σύνολο που αποτελείται από οργανωμένα ομοειδή στοιχεία. Τα στοιχεία ενός αρχείου μπορούμε να τα οργανώσουμε σε λογικές ενότητες και το σύνολο των στοιχείων που περιέχει μια λογική ενότητα καλείται εγγραφή (record). Το κάθε στοιχείο της εγγραφής καλείται πεδίο (field). Το πεδίο αποτελεί και τη μικρότερη δυνατή υποδιαίρεση των στοιχείων ενός αρχείου. Ένα πεδίο χαρακτηρίζεται από τον μέγιστο αριθμό των χαρακτήρων (bytes) που απαιτούνται για την καταχώρησή του στη μνήμη του υπολογιστή και που αποκαλείται μήκος του πεδίου (field length).

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.

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).

3.4 Προβλήματα της Οργάνωσης Αρχείων

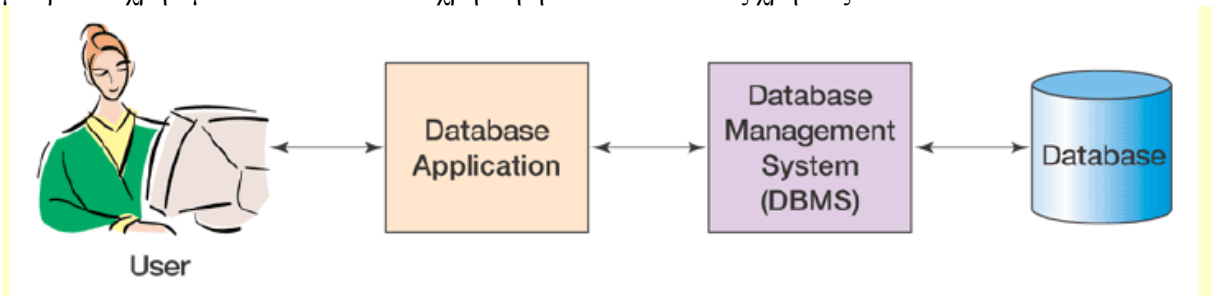
- Πλεονασμός των δεδομένων (data redundancy).
- Ασυνέπεια των δεδομένων (data inconsistency).
- Αδυναμία μερισμού δεδομένων (data sharing).
- Αδυναμία προτυποποίησης.

3.5 Οι Βάσεις Δεδομένων και τα ΣΔΒΔ (DBMS)

Για να δοθεί μια λύση σ' όλα τα παραπάνω προβλήματα, και με βάση το γεγονός ότι η χρήση των ηλεκτρονικών υπολογιστών και συνεπώς η ηλεκτρονική καταχώρηση και επεξεργασία δεδομένων αυξήθηκε κατακόρυφα ήδη από τη δεκαετία του '70 στις μεγάλες επιχειρήσεις και άρα είχαμε πάρα πολλές εφαρμογές να επεξεργάζονται δεδομένα σε πάρα πολλά αρχεία ταυτόχρονα, προτάθηκε η συνένωση όλων των αρχείων μιας εφαρμογής. Εκτός, όμως, από τη συνένωση των αρχείων, απαιτείτο και μια σωστή οργάνωσή τους. Δημιουργήθηκαν έτσι οι Τράπεζες Πληροφοριών ή Βάσεις Δεδομένων (Data Bases).

Μια Βάση Δεδομένων (ΒΔ) είναι ένα σύνολο αρχείων με υψηλό βαθμό οργάνωσης τα οποία είναι συνδεδεμένα μεταξύ τους με λογικές σχέσεις, έτσι ώστε να μπορούν να χρησιμοποιούνται από πολλές εφαρμογές και από πολλούς χρήστες ταυτόχρονα. Υπάρχει ένα ειδικό λογισμικό το οποίο μεσολαβεί ανάμεσα στις αρχεία δεδομένων και τις εφαρμογές που χρησιμοποιούν οι χρήστες και αποκαλείται Σύστημα Διαχείρισης Βάσης Δεδομένων (ΣΔΒΔ) ή DBMS (Data Base Management System). Το ΣΔΒΔ είναι στην ουσία 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.



Εικόνα 8: Σύστημα Β.Δ

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.

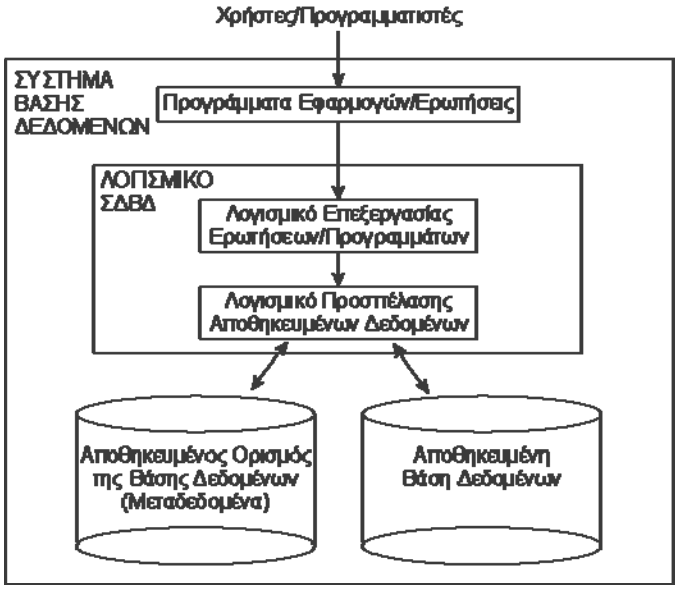
- Οι χρήστες (users) μιας βάσης δεδομένων χωρίζονται στις εξής κατηγορίες :

- 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.



Εικόνα 9: Σύστημα διαχείρισης βάσεων δεδομένων

3.5.1 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.

3.5.2 Οι Οντότητες (Entities)

Με τον όρο ***οντότητα (entity)*** εννοούμε ένα αντικείμενο, ένα πρόσωπο, μια κατάσταση και γενικά οτιδήποτε μπορεί να προσδιορισθεί σαν ανεξάρτητη ύπαρξη (αυτόνομη μονάδα του φυσικού κόσμου). Για παράδειγμα, σε μια βάση δεδομένων μιας εμπορικής εταιρείας, οντότητες μπορεί να είναι οι εργαζόμενοι, οι πελάτες, οι προμηθευτές, οι παραγγελίες, τα είδη της αποθήκης (προϊόντα) κ.ά.

Το ***Μοντέλο Οντοτήτων Συσχετίσεων (Entity Relationship Model, ER Model)*** είναι μια διαγραμματική αναπαράσταση της δομής μιας βάσης δεδομένων και χρησιμοποιείται κατά τη φάση του λογικού σχεδιασμού της βάσης. Δηλαδή, δεν ασχολείται με τον τρόπο που αποθηκεύονται τα δεδομένα της βάσης, αλλά με την ταυτοποίηση των δεδομένων και με τον τρόπο με τον οποίο αυτά συσχετίζονται μεταξύ τους.

Θα δούμε ένα παράδειγμα μιας εταιρείας, η οποία περιέχει δεδομένα που αφορούν τους υπαλλήλους της (employees), τα τμήματά της (departments) και τα έργα (projects) που έχουν αναλάβει αυτά τα τμήματα. Ένα τμήμα της εταιρείας μπορεί να εποπτεύει ένα ή περισσότερα έργα (projects) και ένας υπάλληλος ανήκει σ' ένα μόνο τμήμα της εταιρείας αλλά μπορεί να απασχολείται ταυτόχρονα σε πολλά έργα, τα οποία δεν είναι υποχρεωτικό να παρακολουθούνται από το ίδιο τμήμα.

3.5.3 Οι Ιδιότητες (Attributes)

Με τον όρο ***ιδιότητα*** ή ***χαρακτηριστικό*** ή και ***πεδίο (attribute)*** μιας οντότητας, αναφερόμαστε σ' ένα από τα συστατικά της στοιχεία που την περιγράφουν και την κάνουν να ξεχωρίζει από τα άλλα στοιχεία της ίδιας οντότητας. Για παράδειγμα, η οντότητα ΠΕΛΑΤΗΣ μπορεί να έχει ως ιδιότητες (χαρακτηριστικά) τον κωδικό, το επώνυμο, το όνομα, τη διεύθυνση, το τηλέφωνο, το ΑΦΜ κ.ά., με τη βοήθεια των οποίων μπορούμε να ξεχωρίσουμε τους πελάτες μεταξύ τους.
Επίσης, η οντότητα ΠΑΡΑΓΓΕΛΙΑ μπορεί να έχει ως ιδιότητες (χαρακτηριστικά) τον κωδικό, τον αριθμό παραστατικού, την ημερομηνία, τον κωδικό πελάτη, το προϊόν κ.ά., με τη βοήθεια των οποίων

μπορούμε να ξεχωρίσουμε τις παραγγελίες μεταξύ τους Οι πληροφορίες αυτές αποτελούν το ***σχήμα (schema)*** της οντότητας.

3.5.4 Τα Στιγμιότυπα (Snapshots)

Το κάθε διαφορετικό (αυτόνομο) στοιχείο μιας οντότητας αποκαλείται ***στιγμιότυπο (snapshot)*** ή και ***εμφάνιση*** της οντότητας. Για παράδειγμα, στην οντότητα ΠΕΛΑΤΗΣ, άλλο στιγμιότυπο είναι ο πελάτης με επώνυμο Παπαδόπουλος και άλλο στιγμιότυπο είναι ο πελάτης με επώνυμο Σουμπάσης.

3.5.5 Το Πρωτεύον Κλειδί (Primary Key)

Πρωτεύον κλειδί ή ***πεδίο κλειδί*** *(primary key)* μιας οντότητας καλείται εκείνη η ιδιότητα (ή ο συνδυασμός ιδιοτήτων) που έχει μοναδική τιμή για όλα τα στιγμιότυπα (εμφανίσεις) της οντότητας. Για παράδειγμα, στην οντότητα ΠΕΛΑΤΗΣ πρωτεύον κλειδί είναι ο κωδικός πελάτη, στην οντότητα ΠΑΡΑΓΓΕΛΙΑ πρωτεύον κλειδί μπορεί να είναι ο κωδικός παραγγελίας ή ο αριθμός παραστατικού κοκ.

Υπάρχουν περιπτώσεις όπου το πεδίο κλειδί ενός τύπου οντότητας μπορεί να μην είναι απλό αλλά σύνθετο, να αποτελείται δηλαδή από πολλά απλά πεδία και τότε η συνθήκη της μοναδικότητας για την τιμή του κλειδιού δεν εφαρμόζεται σε κάθε πεδίο του σύνθετου κλειδιού αλλά στο σύνολο του συνδυασμού αυτών των πεδίων.

3.5.6 Οι Συσχετίσεις (Relationships)

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

Για παράδειγμα, η οντότητα ΓΙΑΤΡΟΣ συσχετίζεται με την οντότητα ΑΣΘΕΝΗΣ αλλά και με την οντότητα ΚΛΙΝΙΚΗ στη βάση δεδομένων ενός νοσοκομείου. Μπορούμε να δεχθούμε ότι ένας γιατρός παρακολουθεί (συσχετίζεται με) πολλούς ασθενείς, αλλά ένας ασθενής παρακολουθείται από (συσχετίζεται με) έναν μόνο γιατρό και επίσης ένας γιατρός συσχετίζεται με (ανήκει σε) μία μόνο κλινική, αλλά μια κλινική συσχετίζεται με (απασχολεί) πολλούς γιατρούς.

3.5.7 Τα τρία βασικά μοντέλα

3.5.7.1 Το Ιεραρχικό Μοντέλο Βάσεων Δεδομένων

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. 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.

3.5.7.2 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.

3.5.7.3 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 με 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.

Ένας ***πίνακας (table)*** αποτελείται από γραμμές (rows) και στήλες (columns), όπου τοποθετούμε τα στοιχεία σε οριζόντια και κάθετη μορφή. Η κάθε στήλη του πίνακα χαρακτηρίζει κάποια ιδιότητα της οντότητας και αποκαλείται ***χαρακτηριστικό (attribute)*** ή ***πεδίο (field)***, ενώ η κάθε γραμμή του πίνακα περιέχει όλες τις πληροφορίες (στήλες) που αφορούν ένα στοιχείο της οντότητας και αποκαλείται ***πλειάδα (tuple)*** ή ***εγγραφή (record)*.**

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)*.

3.6 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 που αποκαλούνται ***γλώσσες ερωταπαντήσεων (query languages).*** Η γλώσσα που αποτελεί σήμερα ένα διεθνές πρότυπο για την επικοινωνία των χρηστών με τα Σχεσιακά ΣΔΒΔ είναι η ***SQL (Structured Query Language)*** ή ***Δομημένη Γλώσσα Ερωτημάτων.*** Μπορεί να λειτουργήσει αυτόνομα αλλά και σε συνεργασία μ' άλλες γλώσσες προγραμματισμού.

Μια άλλη, φιλική προς τον χρήστη γλώσσα προγραμματισμού για να μπορούμε να υποβάλουμε ερωτήματα σε σχεσιακές βάσεις δεδομένων και να λαμβάνουμε απαντήσεις είναι η ***QBE (Query By Example),*** η οποία χρησιμοποιεί φόρμες για τη γραφική απεικόνιση των ερωτημάτων μας.
Σήμερα, υπάρχουν εξελιγμένα εργαλεία διαχείρισης σε γραφικό και φιλικό προς τον χρήστη περιβάλλον για να κάνουμε τα εξής :

- Δημιουργία πινάκων
- Δημιουργία φορμών
- Δημιουργία ερωτημάτων
- Δημιουργία εκθέσεων (αναφορών)

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.

3.7 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.

3.7.1 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 : ΠΕΛΑΤΗΣ, ΠΑΡΑΓΓΕΛΙΑ, ΠΡΟΜΗΘΕΥΤΗΣ, ΑΠΟΘΗΚΗ, ΜΑΘΗΤΗΣ, ΚΑΘΗΓΗΤΗΣ, ΑΘΛΗΤΗΣ, ΑΓΩΝΙΣΜΑ, ΧΩΡΑ, ΠΟΛΕΙΣ κ.ά.
Μια βάση δεδομένων μπορεί να περιέχει πολλές διαφορετικές οντότητες, οι οποίες απεικονίζονται με ορθογώνια παραλληλόγραμμα και συσχετίζονται μεταξύ τους ανά δύο.

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.

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 #.

Στο διάγραμμα οντοτήτων–συσχετίσεων οι ιδιότητες απεικονίζονται με σχήματα ελλειπτικής μορφής, τα οποία ενώνονται με ευθείες γραμμές με την οντότητα στην οποία ανήκουν.

3.7.2 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).

3.7.3 Οι Συσχετίσεις Μεταξύ Οντοτήτων

Ο σωστός σχεδιασμός και προσδιορισμός των οντοτήτων και των ιδιοτήτων τους αποτελούν τα θεμελιώδη βήματα για τη σωστή σχεδίαση και υλοποίηση μιας βάσης δεδομένων. Μια συσχέτιση συνδέει δύο ή και περισσότερες οντότητες μεταξύ τους και παριστάνεται στο διάγραμμα οντοτήτων–συσχετίσεων μ' έναν ρόμβο.

Οι συσχετίσεις είναι απαραίτητες για να μπορέσουμε να αντλήσουμε πληροφορίες που αφορούν δύο ή και περισσότερες οντότητες, όπως για παράδειγμα ποιοι πελάτες έκαναν παραγγελίες κάποια συγκεκριμένη χρονική περίοδο (συσχέτιση ΠΑΡΑΓΓΕΛΝΕΙ) ή ποιοι αθλητές ανήκουν σε ποιους συλλόγους (συσχέτιση ΑΝΗΚΕΙ) ή ποιοι αθλητές έλαβαν μέρος σε αγωνίσματα μια συγκεκριμένη χρονιά (συσχέτιση ΣΥΜΜΕΤΕΧΕΙ) κοκ.

Όταν οι οντότητες που συμμετέχουν σε μια συσχέτιση είναι δύο, η συσχέτιση αποκαλείται ***διμελής*** ή ***δυαδική***. Ο βαθμός μιας συσχέτισης είναι ίσος με το πλήθος των οντοτήτων που συμμετέχουν σ' αυτήν.

Όταν σχεδιάζουμε μια βάση δεδομένων, θα πρέπει να εκχωρούμε ιδιότητες μόνο στις οντότητες και να έχουμε τις συσχετίσεις απλά και μόνο για να κατανοούμε τις λογικές συνδέσεις ανάμεσα στις οντότητες

Οι Διμελείς Συσχετίσεις

Οι διμελείς συσχετίσεις μεταξύ οντοτήτων είναι αυτές που θα μας απασχολήσουν ιδιαίτερα και υπάρχουν τρία βασικά είδη συνδέσεων σ' αυτές, τα εξής :

- ***Ένα-προς-ένα (1:1)***, όπου μια εμφάνιση της μιας οντότητας συνδέεται με μία και μόνο μία εμφάνιση της άλλης οντότητας.
- ***Ένα-προς-πολλά (1:M)***, όπου μια εμφάνιση της μιας οντότητας συνδέεται με πολλές εμφανίσεις της άλλης οντότητας αλλά κάθε εμφάνιση της δεύτερης οντότητας συνδέεται με μία και μόνο μία εμφάνιση της πρώτης οντότητας.
- ***Πολλά-προς-πολλά (M:N)***, όπου σε μια εμφάνιση της μιας οντότητας αντιστοιχούν πολλές εμφανίσεις της άλλης οντότητας και σε κάθε εμφάνιση της δεύτερης οντότητας αντιστοιχούν πολλές εμφανίσεις της πρώτης οντότητας.

Το Διάγραμμα Οντοτήτων Συσχετίσεων

Για να μπορέσουμε να διαμορφώσουμε το διάγραμμα οντοτήτων συσχετίσεων, θα πρέπει να ακολουθήσουμε τα εξής βήματα :

- Να ορίσουμε τις οντότητες (πίνακες) που θα ανήκουν στη βάση δεδομένων που θέλουμε να δημιουργήσουμε.
- Να ορίσουμε τις ιδιότητες (πεδία) και τα πρωτεύοντα κλειδιά της κάθε οντότητας (πίνακα).
- Να ορίσουμε τις συσχετίσεις ανάμεσα στις οντότητες.
- Δημιουργούμε το διάγραμμα οντοτήτων συσχετίσεων, όπου θα απεικονίσουμε τις οντότητες, τις ιδιότητές τους και τις συσχετίσεις τους.

3.8 MySQL

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.

3.8.1 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.

3.9 Δημιουργία βάσεων δεδομένων

Για να φτιάξουμε μία νέα βάση δεδομένων πρέπει να έχουμε δικαίωμα από τον διαχειριστή να το κάνουμε. Δικαιώματα δημιουργίας και διαγραφής σε μία βάση δεδομένων έχει μόνον ο υπερχρήστης. Για να δημιουργήσει ο υπερχρήστης μία νέα βάση δεδομένων αρκεί να δώσει

create database name;

όπου το name είναι το όνομα της βάσεως. Στην συνέχεια θα πρέπει να δώσει δικαιώματα και σε μας με την χρήση της εντολής

grant all on name.* to user;

Αφού γίνουν όλα αυτά για να χρησιμοποιήσουμε μία βάση δεδομένων απαιτείται να δώσουμε την εντολή use name στην γραμμή εντολών της mysql. Αν την πρώτη φορά που ζητήσουμε να χρησιμοποιήσουμε την βάση πάρουμε ένα μήνυμα λάθους σχετικά με δικαιώματα, τότε μπορούμε να εισάγουμε την εντολή connect name και ύστερα use name και το πρόβλημα έχει λυθεί. Αν όλα τα παραπάνω ακούγονται πολύπλοκα μπορούμε να χρησιμοποιήσουμε την προκαθορισμένη βάση δεδομένων test, για την οποία δεν χρειάζεται να ζητήσουμε κάτι από τον διαχειριστή συστήματος. Ωστόσο είναι κοινή βάση και έτσι ότι φτιάξουμε στην συνέχεια θα μπορούν να το δουν και να το αλλάξουν και άλλοι χρήστες του συστήματος.

3.9.1 Δημιουργία πινάκων

Αφού συνδεθούμε σε μία βάση δεδομένων το επόμενο βήμα είναι να κάνουμε κάποια εργασία σε πίνακες που διαθέτει. Αν δεν έχει πίνακες μπορούμε να φτιάξουμε με την εντολή create table. Αν για παράδειγμα θέλουμε ένα πίνακα για την περιγραφή φοιτητών μπορούμε να γράψουμε:

create table student(name varchar(20), lastname varchar(20), code int);

3.9.2 Εισαγωγή δεδομένων

Η εισαγωγή δεδομένων στην βάση μπορεί να γίνει είτε με την εντολή insert είτε από κάποιο αρχείο. Με την εντολή insert μπορούμε να γράψουμε για το παράδειγμα του πίνακα φοιτητών:

insert into student(name,lastname,code) values('giannis','papadopoulos',122);

Αν έχουμε να περάσουμε κάποιες χιλιάδες ονόματα η παραπάνω διαδικασία γίνεται κουραστική. Αυτό που μπορούμε να κάνουμε είναι να γράψουμε τις πλειάδες που θέλουμε σε ένα αρχείο κειμένου, έστω students.txt, και στην συνέχεια να φορτώσουμε αυτά τα στοιχεία στην βάση δεδομένων

load data local infile 'students.txt' into table student;

Στο αρχείο δεδομένων τα πεδία στην κάθε πλειάδα χωρίζονται μεταξύ τους με TAB.

3.9.3 Ενημέρωση δεδομένων

Πολλές φορές χρειάζεται να αλλάξουμε κάποια από τα δεδομένα που έχουμε εισάγει νωρίτερα σε κάποιον πίνακα. Η αλλαγή αυτή γίνεται με την χρήση της εντολής update. Για το παράδειγμα των φοιτητών που είδαμε προηγουμένως μία έγκυρη εντολή ενημερώσεως θα μπορούσε να είναι:

update students set name='kostas' where name='giannis';

3.9.4 Ερωτήματα

Η εντολή που χρησιμοποιούμε για να εκτελέσουμε κάποια ερωτήματα σε μία βάση δεδομένων είναι η select. Η πλήρης (όσο γίνεται) σύνταξή της είναι

select field1, field2,...
from tablename
where boolean – condition;

Όλα τα δεδομένα

Για να πάρουμε όλα τα δεδομένα από έναν πίνακα χρησιμοποιούμε την σύνταξη

select * from tablename;

Συγκεκριμένες πλειάδες

Για να ανακτήσουμε συγκεκριμένες πλειάδες από έναν πίνακα αρκεί να ορίσουμε κάποια λογική συνθήκη στην ενότητα where της εντολής select. Για παράδειγμα για να εμφανίσουμε τους φοιτητές με κωδικό μεγαλύτερο του 500 γράφουμε

```
select *
from student
where code>500;
```

Συγκεκριμένες στήλες

Για να επιλέξουμε συγκεκριμένες στήλες από έναν πίνακα πρέπει να προσδιορίσουμε τις στήλες αυτές στην κεφαλή του select. Για παράδειγμα για να εμφανίσουμε μόνον τα ονόματα των φοιτητών γράφουμε:

```
select name
from student;
```

Ταξινόμηση πλειάδων

Είναι χρήσιμο στην έξοδο των αποτελεσμάτων από μία επιλογή να δούμε τα αποτελέσματα ταξινομημένα προκειμένου να αποφασίσουμε πιο εύκολα. Αυτό γίνεται βάζοντας το επίθεμα ORDER BY COLUMN στο τέλος της εντολής select. Column είναι το όνομα του πεδίου κατά το οποίο επιθυμούμε ταξινόμηση. Για παράδειγμα αν θέλουμε να ταξινομήσουμε τους φοιτητές ως προς το επίθετό τους γράφουμε:

```
select *
from student
order by lastname;
```

Ταίριασμα προτύπων

Με το ταίριασμα προτύπων κάνουμε πράξεις μεταξύ αλφαριθμητικών προκειμένου να βρίσκουμε πλειάδες στις οποίες υπάρχει ταίριασμα κάποιου ονόματος με ένα συγκεκριμένο πρότυπο αλφαριθμητικού. Για παράδειγμα για να βρούμε όλους τους φοιτητές που το όνομά τους τελειώνει σε 'ΟΣ' μπορούμε να γράψουμε

```
select *
from student
where name like '%ΟΣ';
```

Το σύμβολο % συμβολίζει το οτιδήποτε.

Διαγραφή

Για να διαγράψουμε κάποιες πλειάδες από μία βάση δεδομένων πρέπει να χρησιμοποιήσουμε την εντολή delete, που συντάσσεται ως εξής:

```
delete from tablename where condition;
```

Αν για παράδειγμα θέλουμε να σβήσουμε τους φοιτητές που ο κωδικός τους είναι αρνητικός (λάθος στην εισαγωγή), τότε μπορούμε να γράψουμε:

```
delete from student where code<0;
```

Αν θέλουμε να διαγράψουμε ολόκληρο τον πίνακα student, τότε δίνουμε

```
drop table student;
```

3.10 JDBC

Η τεχνολογία JDBC (σημαίνει Java Database Connectivity) μας επιτρέπει να έχουμε πρόσβαση σε μία σχεσιακή βάση δεδομένων μέσα από το πρόγραμμά μας. 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. Οι απαραίτητες κλάσεις ορίζονται από τη βιβλιοθήκη **java.sql**.

3.10.1 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]-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 Netbeans 7.0.1 είναι ο κατάλογος εγκατάστασης της γλώσσας Java στο σύστημά μας, μπορούμε να αντιγράψουμε το αρχείο του οδηγού στον κατάλογο C:\Program Files\NetBeans 7.0.1\profiler στον κατάλογο στον κατάλογο \lib.
Αφού έχουμε φορτώσει τον κατάλληλο οδηγό μπορούμε να συνδεθούμε χρησιμοποιώντας την κλάση java.sql.DriverManager. Για να πραγματοποιηθεί η σύνδεση πρέπει να ορίσουμε τουλάχιστον το μηχάνημα όπου είναι εγκατεστημένη η βάση, το όνομα της βάσης και το όνομα και τον κωδικό του χρήστη.

3.10.2 Επικοινωνία με τη Βάση

Για να στείλουμε εντολές στη βάση χρησιμοποιούμε την κλάση java.sql.Statement. Αντικείμενα της κλάσης αυτής μας επιστρέφουν τα αντικείμενα της κλάσης java.sql.Connection.

Για να μεταβάλλουμε τη βάση (δηλαδή να μεταβάλλουμε τον ορισμό των πινάκων, να ορίσουμε νέους πίνακες ή να σβήσουμε κάποιους πίνακες, να προσθέσουμε, να σβήσουμε ή να μεταβάλλουμε εγγραφές) χρησιμοποιούμε τη μέθοδο executeUpdate της κλάσης java.sql.Statement. Η μέθοδος executeUpdate δέχεται ένα string με την εντολή SQL που θέλουμε να εκτελέσουμε και επιστρέφει τον αριθμό των γραμμών που επηρεάστηκαν αν η εντολή είναι insert, update, delete, ή 0 αν η εντολή δεν επιστρέφει τίποτε.

Για να αναζητήσουμε στοιχεία στη βάση χρησιμοποιούμε τη μέθοδο executeQuery της κλάσης java.sql.Statement. Η μέθοδος επιστρέφει ένα αντικείμενο της κλάσης java.sql.ResultSet με τα αποτελέσματα της αναζήτησής μας. Αφού ολοκληρώσουμε την επικοινωνία μας με τη βάση κλείνουμε το αντικείμενο java.sql.Statement με τη μέθοδο close.

4.Μετοχές και Δείκτες Τεχνικής Ανάλυσης

4.1 Ανάλυση Μετοχών

Μετοχές είναι η συμμετοχή στην ιδιοκτησία κάποιας εταιρείας. Οι μετοχές είναι οι πλέον εμπορεύσιμες αξίες. Είναι η μονάδα που διαιρείται το μετοχικό κεφάλαιο μίας εταιρείας, παρέχει στον κάτοχο της δικαίωμα ψήφου και δικαίωμα συμμετοχής στα κέρδη που προκύπτουν από τις εργασίες της. Επίσης συμμετέχουν αναλογικά σε νέες εκδόσεις κεφαλαίου (rights issues).

- Η **κοινή μετοχή** είναι ο πιο συνηθισμένος τύπος μετοχής και περιλαμβάνει όλα τα βασικά δικαιώματα ενός μετόχου, όπως δικαίωμα συμμετοχής στα κέρδη, στην έκδοση νέων μετοχών, στο προϊόν της εκκαθάρισης, καθώς και δικαίωμα ψήφου στη Γενική Συνέλευση της εταιρείας και συμμετοχής στη διαχείρισή της.
- Η **προνομιούχος μετοχή** προσφέρει απλά ένα προβάδισμα έναντι των κατόχων κοινών μετοχών, στη λήψη μερίσματος και στη λήψη του προϊόντος της εκκαθάρισης σε περίπτωση διάλυσης της επιχείρησης, αλλά συνήθως στερείται του δικαιώματος ψήφου και συμμετοχής στη διαχείριση της επιχείρησης.

- Η **ονομαστική αξία** της μετοχής προκύπτει κατά την πρώτη έκδοση των μετοχών, διαιρώντας την αξία του μετοχικού κεφαλαίου της εταιρείας με τον αριθμό μετοχών, που εξέδωσε αρχικά. Η ονομαστική αξία μπορεί να αλλάξει μετά από απόφαση της Γενικής Συνέλευσης της εταιρείας.
- Η **λογιστική αξία** της μετοχής αντικατοπτρίζει την πραγματική αξία της και προκύπτει διαιρώντας τα ίδια κεφάλαια της εταιρείας με τον αριθμό μετοχών της εταιρείας σε κυκλοφορία.
- Η **χρηματιστηριακή αξία** της μετοχής, διαμορφώνεται καθημερινά ,μέσω της προσφοράς και της ζήτησης.

Ωφελήματα Μετόχων

Ο επενδυτής βλέπει το χρηματιστήριο ως μια εναλλακτική μορφή τοποθέτησης των χρημάτων που αποταμιεύει, με σκοπό την επιδίωξη ικανοποιητικής απόδοσης. Απόδοση που συνήθως είναι υψηλότερη από αυτήν που προσφέρουν άλλες επενδύσεις όπως οι τραπεζικές καταθέσεις και τα κρατικά ομόλογα.

Μία εταιρεία παρέχει στους εγγεγραμμένους μετόχους της διάφορα ωφελήματα όπως μέρισμα, προμέρισμα, δωρεάν μετοχές (bonus issue), ΔΑΜ (warrants), rights issue κ.α. σε μία συγκεκριμένη ημερομηνία (record day). Οι τίτλοι διαπραγματεύονται χωρίς τα ωφελήματα που παρέχονται οκτώ μέρες πριν την πιο πάνω ημερομηνία, σύμφωνα με τους κανόνες διαπραγμάτευσης του Χρηματιστηρίου. Ένας τίτλος όταν διαπραγματεύεται με ωφέλημα χρησιμοποιείται το πρόθεμα "cum" ενώ χωρίς το σχετικό ωφέλημα χρησιμοποιείται το πρόθεμα "ex". Δηλαδή εάν πρόκειται για παραχώρηση μερίσματος, ο τίτλος της μετοχής που θα διαπραγματεύεται με μέρισμα θα αναφέρεται ως "cum-dividend" ενώ αν διαπραγματεύεται χωρίς μέρισμα θα αναφέρεται ως "ex –dividend".

4.2 Ερμηνεία δεικτών τεχνικής ανάλυσης

4.2.1 Τεχνική Ανάλυση Μετοχής

Τεχνική ανάλυση μιας μετοχής ή ενός Δείκτη, ή μιας ολόκληρης οργανωμένης αγοράς, είναι η μελέτη της κίνησης των τιμών, όπως αυτή απεικονίζεται σε ένα διάγραμμα τιμής- χρόνου, με στόχο τον χαρακτηρισμό της συμπεριφοράς της και την πρόβλεψη της μελλοντικής της κίνησης. Στηρίζεται στα γραφήματα της Αναλυτικής Γεωμετρίας και της Στατιστικής, τα οποία όμως έχουν εκλαϊκευτεί, για να μπορεί να τα καταλαβαίνει ο μέσος επενδυτής. Η Τεχνική Ανάλυση δεν ασχολείται με τα θεμελιώδη στοιχεία μιας εταιρείας ή μιας αγοράς, όπως είναι οι ισολογισμοί, ο λόγος P/E, ή τα κέρδη, αλλά μόνο με τα γραφήματα των μετοχών (ή των δεικτών).

Βασική υπόθεση της τεχνικής ανάλυσης είναι πως ότι είναι γνωστό για μια εταιρεία είναι ήδη ενσωματωμένο στην τιμή της. Για την τεχνική ανάλυση δεν έχει σημασία τι προκάλεσε την άνοδο ή κάθοδο της τιμής μιας μετοχής, αλλά πόσο θα διαρκέσει αυτή η μεταβολή, είτε προς τα πάνω είτε προς τα κάτω. Επίσης βασική παραδοχή της τεχνικής ανάλυσης αποτελεί το γεγονός ότι η τιμή μιας μετοχής απεικονίζει τα οικονομικά δεδομένα, τις πληροφορίες και τα νέα της και επηρεάζεται σημαντικά από τα ανθρώπινα συναισθήματα και την ψυχολογία που επικρατεί κάθε περίοδο.

Η Τεχνική Ανάλυση μελετάει τις μεταβολές τιμών από το παρελθόν και προσπαθεί να εντοπίσει έγκαιρα τις μεταβολές που θα γίνουν στο μέλλον και κυρίως το σημείο αντιστροφής μιας κίνησης τιμών.

Πρέπει να γίνει κατανοητό σε όποιον δραστηριοποιείται στην αγορά, με εργαλείο την Τεχνική Ανάλυση ότι οι αγορές δεν κινούνται από τα γεγονότα, αλλά από τις προσδοκίες των ανθρώπων. Οι δε τιμές των μετοχών στο ταμπλό του χρηματιστηρίου δεν απεικονίζουν το "σήμερα" αλλά το "αύριο" με ορίζοντα περίπου 3-6 μηνών ή και περισσότερο.

Κάθε αγορά μετοχών, παραγώγων, αξιών ή/και καταναλωτικών ειδών γεννά, με τη συμπεριφορά της, στους εμπλεκόμενους τα απολύτως απαραίτητα ανθρώπινα συναισθήματα της απληστίας, του φόβου, της ανάγκης μιμητισμού των άλλων και ένταξης σε μεγαλύτερα σύνολα, τα οποία αποτελούν τις αιτίες αποτυχημένης δραστηριοποίησης στην αγορά.

Η Τεχνική Ανάλυση, θεωρεί ότι τα ανθρώπινα συναισθήματα απεικονίζονται στους σχηματισμούς των τιμών, και προσπαθεί να απομονώσει τις ψυχολογικές επιδράσεις της αγοράς, εστιάζοντας στα διαγράμματα και τους σχηματισμούς που δημιουργούνται σε αυτά.

Εργαλεία τεχνικής ανάλυσης

Τα εργαλεία της τεχνικής ανάλυσης είναι οι δείκτες. Ένας δείκτης είναι ένας μαθηματικός υπολογισμός που στηρίζεται στην τιμή της μετοχής ή τον όγκο συναλλαγών της ή και στα δύο. Τα ονόματα των κυριοτέρων δεικτών είναι: Κινητός Μέσος Όρος 5 -30 και 200 ημερών, Στοχαστικός δείκτης, Λωρίδες Bollinger κ.λπ. Σήμερα με τη χρήση των υπολογιστών, η τεχνική ανάλυση είναι πολύ πιο εύκολη και χρήσιμη για τον υπολογισμό της κίνησης των τιμών των μετοχών. Επίσης σήμερα υπάρχουν αρκετά προγράμματα Τεχνικής Ανάλυσης όπως είναι το Metastock, τα οποία εφ' όσον εφοδιαστούν με τα ιστορικά στοιχεία μιας μετοχής ή ενός δείκτη, παρουσιάζουν σε γραφήματα όλους τους σημαντικότερος δείκτες Τεχνικής Ανάλυσης

Κυριότερος μεσοπρόθεσμος δείκτης αγορών

Είναι ο λεγόμενος "Κινητός μέσος των 200 ημερών" Θεωρείται ο σημαντικότερος δείκτης για μετοχές, αλλά και για ολόκληρες αγορές. Η κίνηση του έχει διαπιστωθεί ότι επηρεάζεται τόσο από τα οικονομικά στοιχεία, όσο και από τις πολιτικές εξελίξεις (όταν πρόκειται για οργανωμένες αγορές όπως είναι το Χρηματιστήριο Αθηνών), ακόμα και από την γενικότερη ψυχολογία του επενδυτικού κοινού.

Συμπερασματικά

Οποιαδήποτε μετοχή και οποιοσδήποτε δείκτης έχει 50% πιθανότητα να ανέβει και 50% να πέσει. Όποια μέθοδο και όποιο δείκτη και αν χρησιμοποιήσει κανείς δεν μπορεί να προβλέψει την κίνηση των χρηματαγορών με ακρίβεια. Η τεχνική ανάλυση λειτουργεί με πιθανότητες και αποσκοπεί στο να βελτιώσει αυτές τις πιθανότητες προς όφελος του επενδυτή. **Γενικά αν μία μέθοδος πρόβλεψης της κίνησης των τιμών των μετοχών έχει πιθανότητες επιτυχίας 60% θεωρείται επιτυχημένη.**

Ένα σύστημα προβλέψεων λειτουργεί ως εξής : Αρχικά έχουμε τη συλλογή των δεδομένων που αποτελούν τις εισροές του συστήματος. Εν συνεχεία θα πρέπει να γίνει επιλογή της μεθόδου προβλέψεως που θα χρησιμοποιηθεί για την επεξεργασία των δεδομένων μας. **Προκειμένου να επιλέξουμε την καταλληλότερη μέθοδο πρόβλεψης θα πρέπει να λάβουμε υπόψη μας μεταξύ άλλων:** 1) την ακρίβεια των προβλέψεων 2) την ευστάθεια της μεθόδου πρόβλεψης 3) την αντικειμενικότητα στην επεξεργασία των δεδομένων 4) τον απαιτούμενο χρόνο για τη διαμόρφωση προβλέψεων και 5) το κόστος εφαρμογής της μεθόδου.

Επιπροσθέτως εκτός από τα ανωτέρω κριτήρια αξιολόγησης των μεθόδων πρόβλεψης θα πρέπει να λάβουμε υπόψη μας και τυχόν περιορισμούς που υπάρχουν στη λειτουργία ενός συστήματος προβλέψεων. Οι σπουδαιότεροι από αυτούς είναι οι ακόλουθοι : 1) ο διαθέσιμος χρόνος για την προετοιμασία μιας πρόβλεψης 2) η έλλειψη στοιχείων 3) η ποιότητα και η αξιοπιστία των διαθέσιμων στοιχείων 4) η έλλειψη εξειδικευμένου προσωπικού και 5) τα διαθέσιμα μέσα για την επεξεργασία των στοιχείων (υπολογιστές, ειδικά προγράμματα κ.τ.λ). Εφόσον έχουμε επιλέξει την καταλληλότερη μέθοδο προβλέψεων, την εφαρμόζουμε και προβαίνουμε στη διαμόρφωση προβλέψεων που αφορούν τη μελλοντική εξέλιξη των τιμών της υπό εξέταση μεταβλητής.

4.2.2 Τεχνικοί Δείκτες

Πιο ειδικά, με τον όρο τεχνική ανάλυση εννοείται η συστηματική μελέτη και επεξεργασία διαγραμμάτων, με στόχο την όσο δυνατή ασφαλέστερη και ακριβέστερη πρόβλεψη των τιμών των μετοχών.

Ένας δείκτης τεχνικής ανάλυσης είναι ένας μαθηματικός υπολογισμός που έχει να κάνει είτε με την τιμή της μετοχής είτε με τον όγκο συναλλαγών της. Ο κινητός μέσος όρος είναι ένα παράδειγμα δείκτη ανάλυσης. Ουσιαστικά είναι ένας μαθηματικός υπολογισμός σχετικός με την τιμή της μετοχής που χρησιμοποιείται στην προσπάθεια να εκτιμηθούν μελλοντικές αλλαγές στην τιμή της μετοχής. **Η αλλαγή των προσδοκιών των επενδυτών συχνά είναι απότομη και βασίζεται σε ειδήσεις γύρω από την εταιρεία. Όταν η μετοχή είναι σε ανοδική πορεία οι αγοραστές είναι πιο πολλοί από τους πωλητές. Όταν η μετοχή είναι σε καθοδική πορεία αυτοί που πουλάνε είναι πιο πολλοί από αυτούς που αγοράζουν.**

Όταν γίνεται λόγος για την κατεύθυνση μιας μετοχής εννοείται μια σταθερή πορεία αλλαγής της τιμής της μετοχής προς τα πάνω ή προς τα κάτω (που είναι συνδεδεμένη με τις προσδοκίες των επενδυτών). Η κατεύθυνση μιας μετοχής διαφέρει από τα επίπεδα στήριξης ή αντίστασης, γιατί κατεύθυνση σημαίνει αλλαγή, ενώ αντίσταση ή στήριξη σημαίνει εμπόδιο στην αλλαγή.

Η τεχνική ανάλυση επιδιώκει να εντοπίζει έγκαιρα την έναρξη τάσης των τιμών και να εκμεταλλεύεται την τάση αυτή για την επίτευξη κέρδους. Οι πιο κερδοφόρες αγοραπωλησίες πραγματοποιούνται μέσα σε τάση, διότι η κατεύθυνση της τιμής είναι συγκεκριμένη και προβλέψιμη με μεγάλη πιθανότητα επιτυχίας.

Για το λόγο αυτό έχουν αναπτυχθεί ειδικοί τεχνικοί δείκτες που δείχνουν:

1.Το ξεκίνημα μιας νέας τάσης των τιμών (ανοδική ή καθοδική)
2.Την ταχύτητα και την ορμή αυτής της τάσης
3.Πόσο «ώριμη» είναι η τάση και πότε έχουμε τερματισμό της τάσης

4.Εάν η τάση πρόκειται να αναστραφεί.

Γνωρίζοντας αυτούς τους τέσσερις βασικούς παράγοντες της τάσης, μπορεί να καθοριστεί επακριβώς πότε θα αγοραστεί και θα πουληθεί τη μετοχή ώστε να τηρείται πιστά ο κανόνας «αγόραζε φθηνά, πούλα ακριβά».

Οι τεχνικοί δείκτες που μετρούν αυτούς τους τέσσερις παράγοντες και δίνουν μια ολοκληρωμένη η εικόνα της τάσης των τιμών, ονομάζονται δείκτες τάσης (trend-following indicators). Σύμφωνα με την αγγλική ονομασία τους: οι δείκτες αυτοί «ακολουθούν» την τάση. Αυτό στην πράξη σημαίνει πως οι δείκτες αυτοί δουλεύουν σωστά μόνο όταν οι τιμές των μετοχών είναι μέσα σε περιβάλλον τάσης. Επομένως, δε λειτουργούν σωστά όταν βρίσκονται σε ατασικές-πλευρικές φάσεις των τιμών, δηλαδή μέσα σε ζώνες συναλλαγών ή σχηματισμούς τιμών (τρίγωνα, σφήνες κ.λπ.)

4.2.3 Κινητός Μέσος Όρος

Γενικά

Ο απλούστερος και δημοφιλέστερος δείκτης τάσης των τιμών είναι ο κινητός μέσος. Πολλοί άλλοι σημαντικοί δείκτες τάσης βασίζονται στην αρχή λειτουργίας του κινητού μέσου. Με τον κινητό μέσο είναι δυνατό να απαλειφτούν έντονες καθημερινές διακυμάνσεις της τιμής της μετοχής. Με αυτό τον τρόπο ο δείκτης δίνει την ομαλοποιημένη τάση (smoothed trend) της μετοχής.

Ο κινητός μέσος δίνει ακριβή σήματα αγοράς και πώλησης, καθώς η τάση της τιμής ξεκινάει, εξελίσσεται και ωριμάζει. Επίσης με τη χρήση του μπορεί να αφαιρεθεί η τάση από τις τιμές (detrending) και να μελετηθεί η κυκλική συμπεριφορά της μετοχής.

Επίσης ο κινητός μέσος έχει και μια άλλη ερμηνεία που αφορά περισσότερο τη ψυχολογία των επενδυτών. Είναι ένας δείκτης που μας δίνει τη μέση τιμή κτήσης της μετοχής των τελευταίων ημερών. Όταν ο η τιμή κλεισίματος βρίσκεται πάνω από τον κινητό μέσο, η μέση τιμή κτήσης της μετοχής είναι χαμηλότερη από την τρέχουσα και οι επενδυτές (τουλάχιστον αυτοί που τοποθετήθηκαν πρόσφατα) διατηρούν κλίμα ευφορίας και αισιοδοξίας από την άνοδο των κερδών τους. Άρα η αγορά δείχνει σθένος. Το αντίθετο συμβαίνει όταν η τρέχουσα τιμή κλεισίματος βρίσκεται χαμηλότερα από τον κινητό μέσο, δηλαδή από τη πρόσφατη μέση τιμή κτήσης της μετοχής. Το κλίμα είναι σαφώς αρνητικό με τους επενδυτές να υφίστανται ζημίες κατά μέσο όρο.

Συνεπώς συγκεκριμένοι κανόνες συναλλαγών που προκύπτουν από τη χρήση του κινητού μέσου, οι οποίοι είναι οι εξής:

1. Η ανοδική διάσπαση του ΚΜ από την τιμή κλεισίματος δίνει σήμα ενάρξεως ανοδικής τάσης και αγοράς της μετοχής. Στην περίπτωση ανοδικής τάσης, ο ΚΜ παρέχει στήριξη στην τιμή όταν αυτή επιχειρεί να κάνει διορθωτικές πτώσεις.
2. Η πτωτική διάσπαση του ΚΜ από την τιμή κλεισίματος σηματοδοτεί έναρξη πτωτικής τάσης και μας δίνει το σήμα πώλησης της μετοχής. Στην πτωτική τάση, ο ΚΜ παρέχει το επίπεδο αντίστασης της τιμής, σε κάθε περίπτωση ανοδικής αντίδρασης της.
3. Η μεταβολή της κλίσης του ΚΜ επιβεβαιώνει την αναστροφή της τάσης. Δηλαδή, όταν ένας ανοδικός ΚΜ αντιστρέφεται και αποκτά αρνητική κλίση, επιβεβαιώνει το γεγονός πως η προϋπάρχουσα ανοδική τάση έχει ήδη αντιστραφεί σε πτωτική. Το σήμα αυτό λειτουργεί μόνο ως επιβεβαίωση του γεγονότος αντιστροφής, διότι έρχεται καθυστερημένα λόγω της χρονικής υστέρησης του ΚΜ.
4. Όσο μεγαλύτερης ο αριθμός των ημερών στον ΚΜ, τόσο πιο μακροχρόνια είναι η τάση που αναλύει και τόσο πιο αξιόπιστα είναι τα σήματα αγοραπωλησιών που μας δίνει.
5. Ο ΚΜ δεν προβλέπει την τάση, απλά ακολουθεί την τάση. Πρόκειται για ένα είδος «κινητής» γραμμής τάσης.

<u>Αποκλίσεις</u>

Όταν ο δείκτης Σύγκλισης-Απόκλισης αποκλίνει σε πορεία από την τιμή της μετοχής, αυτό είναι ένδειξη ότι η τωρινή κατεύθυνση της μετοχής είναι πιθανό να αντιστραφεί. Μια αρνητική απόκλιση (bearish divergence) συμβαίνει όταν ο δείκτης πέφτει όλο και πιο χαμηλά, ενώ οι τιμές δεν

ακολουθούν την ίδια πορεία. Μια θετική απόκλιση (bullish divergence) συμβαίνει όταν ο δείκτης ανεβαίνει όλο και πιο ψηλά, ενώ η μετοχή όχι. Και οι δύο αυτές αποκλίσεις θεωρούνται πιο σημαντικές όταν συμβαίνουν σε επίπεδα υπερ-αγοράς ή υπερ-πώλησης.

Υπάρχουν διάφοροι τύποι δεικτών «ΚΙΝΗΤΟΥ ΜΕΣΟΥ ΟΡΟΥ» (μαθηματικές συναρτήσεις για των υπολογισμό των μελλοντικών τιμών μετοχών) όπως:

- **<u>ΑΠΛΟΣ ΚΙΝΗΤΟΣ ΜΕΣΟΣ ΟΡΟΣ (Simple moving average)</u>**

Ο απλός κινητός μέσος όρος είναι ένας αριθμητικός μέσος όρος δεδομένων τιμών. Υπολογίζεται προσθέτοντας την τιμή κάθε διαστήματος και διαιρώντας το σύνολο με τον αριθμό των διαστημάτων που καλύπτει ο κινητός μέσος όρος. Ο μέσος αυτός αποκαλείται «κινητός» γιατί κάθε φορά που γίνεται διαθέσιμη μια νέα παρατήρηση, μπορεί να υπολογιστεί και να χρησιμοποιηθεί ως πρόβλεψη ένας νέος αριθμητικός μέσος. Για παράδειγμα, για να υπολογίσουμε τον κινητό μέσο όρο 25 ημερών προσθέτουμε τις τιμές κλεισίματος των τελευταίων 25 ημερών ενός τίτλου και στη συνέχεια διαιρούμε διά 25.

$$SMA = \frac{p_M + p_{M-1} + \cdots + p_{M-(n-1)}}{n}$$

$$SMA_{\text{today}} = SMA_{\text{yesterday}} - \frac{p_{M-n}}{n} + \frac{p_M}{n}$$

Ή

Οι προβλέψεις μιας χρονοσειράς Yt, όπου t=1,2,…,n, χρησιμοποιώντας τη μέθοδο του απλού κινητού μέσου, δίνονται από τον ακόλουθο μαθηματικό τύπο :

$$\hat{Y}t+1 = Mt+1 = 1 / m * (\Sigma jYt-j+1) \text{ , όπου } j=1,2,\ldots,m$$

Όπου :
$\hat{Y}t+1$ = η πρόβλεψη για την περίοδο (t+1)
Mt+1 = ο απλός κινητός μέσος για την περίοδο (t+1) και
m = ο αριθμός των περιόδων που χρησιμοποιούνται για τον υπολογισμό του απλού κινητού μέσου

Με τη μέθοδο του απλού κινητού μέσου μπορούμε να εξομαλύνουμε τις τιμές της χρονοσειράς. Αναλυτικότερα, εφαρμόζοντας την μαθηματική σχέση που αναπτύξαμε παραπάνω στις παρατηρήσεις της χρονοσειράς εξομαλύνουμε τις τελευταίες (n – m) παρατηρήσεις της χρονοσειράς. Οι τιμές που προκύπτουν χρησιμοποιούνται ως προβλέψεις των αντίστοιχων περιόδων και εν συνεχεία, βάσει αυτών, προσδιορίζουμε τις τιμές των σφαλμάτων της πρόβλεψης.
Ένας εναλλακτικός τρόπος για τη δημιουργία προβλέψεων, στην περίπτωση που η τιμή του m είναι γνωστή, είναι ο εξής :

$$\hat{Y}t+1 = \hat{Y}t + Yt / m – Yt-m / m$$

Ο ανωτέρω μηχανισμός μας δείχνει τον τρόπο με τον οποίο η μέθοδος του απλού κινητού μέσου αναπροσαρμόζει τις προβλέψεις της χρονοσειράς κάθε φορά που μια νέα παρατήρηση γίνεται διαθέσιμη.
Οι κυριότερες αντιρρήσεις που έχουν εκφραστεί αναφορικά με τη χρησιμοποίηση της μεθόδου του απλού κινητού μέσου είναι οι εξής :

- ✓ Η ίση στάθμιση των δεδομένων κάθε μιας εκ των προηγούμενων περιόδων μπορεί να μην εκφράζει ικανοποιητικά την παρούσα κατάσταση.

- ✓ Η αύξηση του αριθμού των καλυπτόμενων χρονικών περιόδων n, έχει ως αποτέλεσμα την μεγαλύτερη εξομάλυνση των διακυμάνσεων, αλλά κάνει τη μέθοδο λιγότερο ευαίσθητη στις πραγματικές αλλαγές των δεδομένων.
- ✓ Ο απλός κινητός μέσος παρουσιάζει μία χρονική υστέρηση σε αλλαγές των πραγματικών τιμών των δεδομένων λόγω μακροχρόνιας τάσεως ή άλλων αιτιών.
- ✓ Η μέθοδος του απλού κινητού μέσου απαιτεί εκτεταμένα αρχεία για δεδομένα παρελθουσών περιόδων.
- ✓ Με τη μέθοδο του απλού κινητού μέσου είναι δυνατή η πρόβλεψη μόνο για την επόμενη χρονική περίοδο.

• **ΑΘΡΟΙΣΤΙΚΟΣ ΚΙΝΗΤΟΣ ΜΕΣΟΣ ΟΡΟΣ (Cumulative moving average)**

Ο αθροιστικός κινητός μέσος είναι και αυτός ένας αριθμητικός μέσος όρος δεδομένων τιμών. Υπολογίζεται προσθέτοντας τις τιμές κάθε διαστήματος και διαιρώντας το σύνολο με τον αριθμό των διαστημάτων που καλύπτει ο κινητός μέσος όρος.

$$CA_i = \frac{x_1 + \cdots + x_i}{i}.$$

$CA_{i-1} = \frac{x_{i+1} + iCA_i}{i+1}$; όπου το CA_0 μπορεί να θεωρηθεί ίσο με το 0.

• **ΣΤΑΘΜΙΣΜΕΝΟΣ ΚΙΝΗΤΟΣ ΜΕΣΟΣ ΟΡΟΣ (Weighted moving average)**

Η μέθοδος του σταθμισμένου κινητού μέσου είναι, όπως και αυτή του απλού κινητού μέσου, μια πολύ απλή μέθοδος προβλέψεων. Μια από τις αντιρρήσεις που έχουν εκφραστεί για τον απλό κινητό μέσο είναι η ίση στάθμιση των δεδομένων κάθε μιας εκ των προηγούμενων περιόδων, η οποία μπορεί να μην εκφράζει ικανοποιητικά την παρούσα κατάσταση. Το μειονέκτημα αυτό μπορούμε να το εξουδετερώσουμε χρησιμοποιώντας το σταθμισμένο κινητό μέσο. Η λογική και η μεθοδολογία της μεθόδου αυτής είναι ακριβώς η ίδια με αυτή του απλού κινητού μέσου με τη μόνη διαφορά ότι εδώ σταθμίζουμε τα δεδομένα των περιόδων που εξετάζουμε, χρησιμοποιώντας διαφορετικούς συντελεστές βαρύτητας-στάθμισης ανάλογα με το χρονικό ορίζοντα απόκτησης των δεδομένων. Συνήθως, τα δεδομένα των πιο πρόσφατων περιόδων τα σταθμίζουμε χρησιμοποιώντας συντελεστές βαρύτητας που είναι μεγαλύτεροι από αυτούς που χρησιμοποιούνται για τη στάθμιση δεδομένων πιο μακρινών περιόδων. Το άθροισμα όλων των συντελεστών βαρύτητας ισούται με τη μονάδα, κάτι που ισχύει και στην περίπτωση του απλού κινητού μέσου.

Η επιλογή των συντελεστών βαρύτητας εξαρτάται από την κρίση και την εμπειρία του εκάστοτε αναλυτή. Γενικά θα μπορούσαμε να πούμε ότι αν το πρόσφατο, σε σχέση με το απώτερο, παρελθόν θεωρείται καλύτερος δείκτης πρόβλεψης του μέλλοντος, τότε μεγαλύτεροι συντελεστές βαρύτητας θα πρέπει να δίνονται στις πιο πρόσφατες παρατηρήσεις. Ωστόσο, αν η χρονολογική σειρά που εξετάζουμε διακρίνεται από μεγάλη μεταβλητότητα, τότε ίσως είναι ορθότερο να χρησιμοποιούμε ίσους συντελεστές βαρύτητας
για κάθε παρατήρηση.

Ο τύπος που μας δίνει τον σταθμισμένο κινητό μέσο είναι ο ακόλουθος:

Σταθμισμένος Κινητός Μέσος =
Σ (συντελεστής βαρύτητας για την περίοδο t)*(παρατήρηση της περιόδου t)/
Σ (συντελεστές βαρύτητας)

Ή με την μαθηματική του μορφή:

$$\mathrm{WMA}_M = \frac{np_M + (n-1)p_{M-1} + \cdots + 2p_{(M-n+2)} + p_{(M-n+1)}}{n + (n-1) + \cdots + 2 + 1}$$

Αξίζει να σημειωθεί ότι τα μειονεκτήματα που αναφέραμε στην προηγούμενη ενότητα και που αφορούν τον απλό κινητό μέσο, ισχύουν, με εξαίρεση αυτό της ίσης στάθμισης, και για τον σταθμισμένο κινητό μέσο.

- **ΕΚΘΕΤΙΚΟΣ ΜΕΣΟΣ ΟΡΟΣ (Exponential moving average)**

Στα πλαίσια της παρουσίασης του απλού κινητού μέσου, που αποτελεί μια πολύ απλή μέθοδο προβλέψεων, διαπιστώσαμε ότι ένα από τα μειονεκτήματα της μεθόδου είναι η ίση βαρύτητα που δίνει, κατά τον υπολογισμό των προβλέψεων, σε κάθε παρατήρηση ανεξάρτητα από τόσο κοντά ή μακριά βρίσκεται σε σχέση με την προβλεπόμενη περίοδο. Ένας τρόπος για την αντιμετώπιση αυτού του προβλήματος είναι η χρήση της μεθόδου του σταθμισμένου κινητού μέσου. Μία άλλη μέθοδος που μπορεί να αντιμετωπίσει το ανωτέρω πρόβλημα και που τυγχάνει ευρείας αποδοχής σε διάφορες περιπτώσεις που απαιτούνται προβλέψεις οικονομικών μεγεθών, είναι η μέθοδος της απλής εκθετικής εξομάλυνσης.

Σύμφωνα με τη μέθοδο αυτή οι προβλέψεις δημιουργούνται βάσει κάποιου σταθμικού μέσου όρου, έτσι ώστε να δίνεται διαφορετική βαρύτητα σε κάθε παρατήρηση. Αναλυτικότερα, με τη μέθοδο αυτή δίνεται μεγαλύτερη βαρύτητα στις πιο πρόσφατες παρατηρήσεις, σε σχέση πάντοτε με την προβλεπόμενη περίοδο, από αυτή που δίνεται στις πιο απομακρυσμένες. Για να μπορέσουμε να κατανοήσουμε τον τρόπο λειτουργίας της μεθόδου αυτής θα θεωρήσουμε ότι οι προβλέψεις της χρονοσειράς που εξετάζουμε δημιουργούνται ως εξής :

$$\hat{Y}t+1= \alpha Yt + \alpha(1-\alpha)Yt-1+\alpha(1-\alpha)2Yt-2+\ldots\ldots..$$

Τη σχέση αυτή θα την ονομάσουμε σχέση (1). Η παράμετρος α της σχέσης αυτής ονομάζεται σταθερά εξομάλυνσης (smoothing constant) και λαμβάνει τιμές από μηδέν έως και ένα, δηλαδή $0 \leq \alpha \leq 1$. Από τη σχέση (1) γίνεται αντιληπτό ότι η πρόβλεψη $\hat{Y}t+1$ προκύπτει ως ένας σταθμικός μέσος όρος των παρατηρήσεων της χρονοσειράς, καθώς το άθροισμα όλων των συντελεστών βαρύτητας ισούται με τη μονάδα. Επιπροσθέτως, διαπιστώνουμε ότι όσο αυξάνεται ο αριθμός των χρονικών περιόδων μεταξύ της πρόβλεψης και της πραγματικής τιμής της χρονοσειράς τόσο οι συντελεστές βαρύτητας μειώνονται και μάλιστα εκθετικά. Γι' αυτό και η υπό εξέταση μέθοδος καλείται απλή «εκθετική» εξομάλυνση. Γίνεται λοιπόν κατανοητό, ότι όσο μεγαλύτερη είναι η τιμή της σταθεράς εξομάλυνσης α τόσο πιο μεγάλη είναι η βαρύτητα που δίνεται στις πιο πρόσφατες παρατηρήσεις και τόσο πιο μικρή ή και μηδαμινή είναι η βαρύτητα που δίνεται στις πιο απομακρυσμένες παρατηρήσεις.

Με βάση τη σχέση (1) η πρόβλεψη για την περίοδο t, $\hat{Y}t$, που γίνεται στην αρχή της περιόδου αυτής, θα είναι :

$$\hat{Y}t= \alpha Yt-1 + \alpha(1-\alpha)Yt-2+\alpha(1-\alpha)2Yt-3+\ldots..$$

Τη σχέση αυτή θα την ονομάσουμε σχέση (2). Αν πολλαπλασιάσουμε και τα δύο μέλη της σχέσης (2) με (1-α) και εν συνεχεία την αφαιρέσουμε από τη σχέση (1), τότε προκύπτει η εξής σχέση :

$$\hat{Y}t+1= \alpha Yt+ (1-\alpha)\hat{Y}t$$

Η ανωτέρω σχέση αποτελεί την μαθηματική έκφραση της μεθόδου της απλής εκθετικής εξομάλυνσης για t = 2,3,….,n και με αρχική συνθήκη $\hat{Y}2=Y1$.

Γίνεται σαφές ότι η πρόβλεψη $\hat{Y}t+1$ της περιόδου (t+1) είναι ένας σταθμικός μέσος όρος της πραγματικής τιμής Yt και της προβλεπόμενης τιμής $\hat{Y}t$ της περιόδου t με συντελεστές βαρύτητας α και (1-α) αντίστοιχα. Αυτό σημαίνει ότι αν η τιμή της σταθεράς εξομάλυνσης α είναι α=0,3, τότε η πρόβλεψη της επόμενης περιόδου προσδιορίζεται κατά 30% από την πραγματική τιμή και κατά 70%

από την προβλεπόμενη της τρέχουσας περιόδου. Να σημειωθεί πως όταν η τιμή της σταθεράς εξομάλυνσης λαμβάνει τις ακραίες τιμές ένα και μηδέν τότε : για α=1 η πρόβλεψη της επόμενης περιόδου (t+1) είναι η πραγματική τιμή της τρέχουσας περιόδου t και για α=0 η πρόβλεψη της επόμενης περιόδου (t+1) είναι ίση με την πρόβλεψη της τρέχουσας περιόδου t.

Παρατηρούμε ότι όσο πιο μικρές είναι οι τιμές της σταθεράς α τόσο περισσότερο εξομαλύνονται οι παρατηρήσεις της χρονοσειράς και όσο πιο μεγάλες είναι οι τιμές της σταθεράς α τόσο πιο γρήγορα αντιδρά η εν λόγω μέθοδος στις πραγματικές μεταβολές των παρατηρήσεων της χρονοσειράς.

Συνεπώς τίθεται ένα ερώτημα: Μεγαλύτερη εξομάλυνση των παρατηρήσεων της χρονοσειράς και άρα μικρή τιμή για το α ή μεγαλύτερη ανταπόκριση στις πραγματικές μεταβολές των παρατηρήσεων και άρα μεγάλη τιμή για το α ; Η απάντηση στο ερώτημα αυτό δεν είναι καθόλου εύκολη. Θα λέγαμε ότι σπουδαίο ρόλο διαδραματίζει η κρίση του ερευνητή και η τυχόν προηγούμενη εμπειρία που έχει για τη συγκεκριμένη χρονολογική σειρά. Βέβαια, το πιο σωστό είναι η «άριστη» τιμή για το α να προσδιορίζεται από τα δεδομένα της χρονοσειράς. Συγκεκριμένα από τις τιμές του α, $0 \leq \alpha \leq 1$, επιλέγουμε εκείνη που ελαχιστοποιεί την τιμή του κριτηρίου MSE ή κάποιου άλλου κριτηρίου.

Αυτό μπορεί να γίνει εύκολα και γρήγορα με τη χρήση ενός σύγχρονου υπολογιστικού πακέτου. Ωστόσο, να σημειωθεί πως η «άριστη» τιμή του α προσδιορίζεται βάσει ενός συγκεκριμένου αριθμού παρατηρήσεων. Έτσι όταν ο αριθμός αυτός μεταβάλλεται καλό θα είναι να προβαίνουμε σε επανεκτίμηση της τιμής της σταθεράς α, ώστε οι προβλέψεις που προκύπτουν να είναι όσο το δυνατόν πιο αξιόπιστες.

Η μαθηματική σχέση που εκφράζει τη μέθοδο της απλής εκθετικής εξομάλυνσης μπορεί να λάβει και την ακόλουθη μορφή :

$$\hat{Y}t+1 = \hat{Y}t + \alpha(Yt - \hat{Y}t) = \hat{Y}t + \alpha et$$

Η ανωτέρω σχέση μας δείχνει ότι η πρόβλεψη της περιόδου (t+1) είναι ίση με την πρόβλεψη της περιόδου t συν το σφάλμα της πρόβλεψης et, πολλαπλασιασμένο με την τιμή της σταθεράς εξομάλυνσης α. Κατά συνέπεια όσο μεγαλύτερη είναι η τιμή του α τόσο πιο μεγάλη βαρύτητα δίνεται στο σφάλμα της πρόβλεψης.

Τα κυριότερα πλεονεκτήματα της μεθόδου που την κάνουν ευρέως χρησιμοποιούμενη είναι:

1. η μεγάλη ακρίβεια πρόβλεψης,
2. η ευκολία υπολογισμού,
3. η απαίτηση ελάχιστων δεδομένων για τον υπολογισμό.

Ο Γενικός τύπος υπολογισμού του εκθετικού μέσου όρου είναι:

$$S_1 = Y_1$$

Για $t > 1, \quad S_t = \alpha \cdot Y_t + (1 - \alpha) \cdot S_{t-1}$

$$\mathrm{EMA}_{\mathrm{today}} = \mathrm{EMA}_{\mathrm{yesterday}} + \alpha \times (\mathrm{price}_{\mathrm{today}} - \mathrm{EMA}_{\mathrm{yesterday}})$$

Σημείωση: Όσο μικρότερη η τιμή του α τόσο μεγαλύτερη εμπιστοσύνη δείχνουμε στην αρχική μας πρόβλεψη.
Συνήθεις τιμές του α: 0.1 – 0.3

4.3 Κριτήρια για την αξιολόγηση της ακρίβειας των προβλέψεων

Στην ενότητα αυτή θα παρουσιάσουμε τα πιο γνωστά κριτήρια που υπάρχουν για την αξιολόγηση των μεθόδων προβλέψεων. Σκοπός της εφαρμογής των κριτηρίων αυτών είναι η **επιλογή της πιο κατάλληλης και αξιόπιστης μεθόδου πρόβλεψης**. Τα κριτήρια αυτά βασίζονται, στις αποκλίσεις που παρουσιάζονται ανάμεσα στις προβλεπόμενες και στις πραγματικές τιμές της χρονοσειράς. Όσο μικρότερες είναι οι αποκλίσεις αυτές τόσο πιο αξιόπιστη θεωρείται η μέθοδος πρόβλεψης που εφαρμόστηκε. Η απόκλιση ανάμεσα στην προβλεπόμενη και την πραγματική τιμή για μια περίοδο t όπου t = 1,2,…..,n καλείται «Σφάλμα Πρόβλεψης» και ορίζεται ως εξής :

$$et = Yt - \hat{Y}t$$

Όπου : Yt = πραγματική τιμή της περιόδου t και
$\hat{Y}t$ = προβλεπόμενη τιμή της περιόδου t

Η ανωτέρω σχέση εκφράζει για κάθε περίοδο t τη διαφορά μεταξύ της πραγματικής τιμής (Yt) και της αντίστοιχης προβλεπόμενης τιμής ($\hat{Y}t$) που προήλθε από τη μέθοδο πρόβλεψης που εφαρμόστηκε. Γίνεται λοιπόν κατανοητό ότι για να προσδιορίσουμε την αξιοπιστία μιας μεθόδου πρόβλεψης θα πρέπει να μελετήσουμε τη διαχρονική συμπεριφορά που παρουσιάζουν οι τιμές των σφαλμάτων της πρόβλεψης.

Αξίζει να σημειωθεί ότι τα κριτήρια που θα εξετάσουμε στη συνέχεια μπορούν να χρησιμοποιηθούν όχι μόνο για την αξιολόγηση μιας μεθόδου πρόβλεψης, αλλά και για την επιλογή της καλύτερης μεταξύ δύο ή και περισσοτέρων εναλλακτικών μεθόδων προβλέψεων.

ΜΕΣΟ ΣΦΑΛΜΑ (MEAN ERROR – ME)

Ένα πολύ απλό μέτρο για την αξιολόγηση της ακρίβειας μιας μεθόδου πρόβλεψης είναι το μέσο σφάλμα. Το μέσο σφάλμα ορίζεται ως εξής : είναι το άθροισμα των τιμών του σφάλματος της πρόβλεψης διαιρούμενο με τον αριθμό των περιόδων n στις οποίες έγιναν προβλέψεις. Με άλλα λόγια:

$$ME = 1 / n * (\Sigma et) \text{ , Όπου } t = 1,2,\ldots,n$$

Η μονάδα μέτρησης του κριτηρίου αυτού είναι η ίδια με εκείνη των τιμών της χρονοσειράς και έτσι η ερμηνεία του είναι εύκολη. Το σοβαρό μειονέκτημα όμως που παρουσιάζει είναι ότι η τιμή που λαμβάνει επηρεάζεται από τις θετικές και αρνητικές τιμές του σφάλματος. Για να αποφύγουμε το πρόβλημα αυτό μπορούμε να καταφύγουμε στο επόμενο κριτήριο που είναι η μέση απόλυτη απόκλιση.

ΜΕΣΟ ΑΠΟΛΥΤΟ ΣΦΑΛΜΑ (MEAN ABSOLUTE ERROR).

Είναι το σφάλμα ως προς τον μέσο όρο των απολύτων τιμών της απόκλισης των πραγματικών τιμών από τις προβλεπόμενες τιμές. Το σφάλμα αυτό δίνεται από τον τύπο:

$$\mathrm{MAE} - \frac{1}{n}\sum_{i=1}^{n}|f_i - y_i| - \frac{1}{n}\sum_{i=1}^{n}|e_i|\,.$$

Σημείωση: Οι θετικές αποκλίσεις εξουδετερώνονται από τις αρνητικές, έτσι ώστε να μπορεί να εμφανιστεί τελικά μικρό μέσο σφάλμα αν και έχουν σημειωθεί στην πραγματικότητα πολύ μεγάλες (θετικές και αρνητικές) αποκλίσεις.

ΜΕΣΗ ΑΠΟΛΥΤΗ ΑΠΟΚΛΙΣΗ (MEAN ABSOLUTE DEVIATION –MAD)

Πρόκειται για ένα απλό και εύχρηστο κριτήριο για την αξιολόγηση της ακρίβειας μιας μεθόδου πρόβλεψης. Η μέση απόλυτη απόκλιση εκφράζει την μέση τιμή των απόλυτων αποκλίσεων μεταξύ των προβλεπόμενων και πραγματικών τιμών μιας χρονοσειράς και ορίζεται ως το άθροισμα των απόλυτων τιμών του σφάλματος της πρόβλεψης ως προς τον αριθμό των περιόδων n για τις οποίες έγιναν προβλέψεις. Δηλαδή :

$$\mathrm{MAD} = \mathrm{median}_i\left(\left| X_i - \mathrm{median}_j(X_j) \right| \right),$$

ή

MAD = 1 / n * (Σt IetI), Όπου t =1,2,…,n

Πχ ας εξετάσουμε τα δεδομένα (1, 1, 2, 2, 4, 6, 9). Έχει μια διάμεση τιμή 2. Οι απόλυτες αποκλίσεις περίπου για το 2 είναι (1, 1, 0, 0, 2, 4, 7) οι οποίες με τη σειρά τους έχουν μία διάμεση τιμή 1 (επειδή οι απόλυτες ταξινόμηση αποκλίσεις είναι (0, 0, 1, 1, 2, 4, 7). Έτσι, η μέση απόλυτη απόκλιση για αυτά τα δεδομένα είναι 1.

Στα θετικά του κριτηρίου καταλογίζονται : α) το γεγονός ότι η μονάδα μέτρησης του, όπως και στην περίπτωση του μέσου σφάλματος, είναι η ίδια με εκείνη των τιμών της χρονοσειράς και έτσι η ερμηνεία του είναι εύκολη β) επειδή χρησιμοποιούμε τις απόλυτες τιμές του σφάλματος της πρόβλεψης, το MAD είναι ανεξάρτητο από το εάν οι τιμές των προβλέψεων είναι μικρότερες (υποεκτίμηση) ή μεγαλύτερες (υπερεκτίμηση) των πραγματικών τιμών και γ) η σοβαρότητα του σφάλματος, δηλαδή το κόστος που δημιουργείται από το σφάλμα της πρόβλεψης, σχετίζεται γραμμικά με το μέγεθος του σφάλματος και κατά συνέπεια η σοβαρότητα του σφάλματος είναι η ίδια, είτε προέρχεται από λίγα και μεγάλα σφάλματα είτε από πολλά και μικρά σφάλματα, που έχουν το ίδιο συνολικό άθροισμα απόλυτων τιμών.

Η ΜΕΣΗ ΑΠΟΛΥΤΗ ΠΟΣΟΣΤΙΑΙΑ ΑΠΟΚΛΙΣΗ (Percent Absolute Deviation, PMAD) δίνεται από τον τύπο:

$$PMAD = \frac{\sum_{t=1}^{N} |E_t|}{\sum_{t=1}^{N} |Y_t|}$$

ΜΕΣΟ ΣΦΑΛΜΑ ΤΕΤΡΑΓΩΝΟΥ (MEAN SQUARED ERROR – MSE)

Μια άλλη τεχνική που είναι γενικά παραδεκτή για την αξιολόγηση των μεθόδων της εκθετικής εξομάλυνσης (καθώς και άλλων) είναι το μέσο σφάλμα τετραγώνου. Το μέσο σφάλμα τετραγώνου είναι η μέση τιμή των τετραγώνων των αποκλίσεων μεταξύ των προβλεπόμενων και των πραγματικών τιμών μιας χρονοσειράς και ορίζεται ως το άθροισμα των τετραγώνων των σφαλμάτων ως προς τον αριθμό των χρονικών περιόδων n στις οποίες έγιναν προβλέψεις. Δηλαδή :

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2.$$

ή

MSE = 1 / n * (Σt (et)2) , Όπου t =1,2,…,n

Ορισμένες ουσιώδεις παρατηρήσεις που μπορούν να γίνουν σχετικά με το υπό εξέταση κριτήριο είναι οι ακόλουθες : α) η μονάδα μέτρησης εις την οποία είναι εκφρασμένο το MSE είναι η ίδια με αυτή των τιμών της χρονοσειράς αλλά υψωμένη στο τετράγωνο. Το γεγονός αυτό δυσχεραίνει σημαντικά την κατανόηση και την ερμηνεία του κριτηρίου. Για το λόγο αυτό πολύ συχνά χρησιμοποιείται η τετραγωνική ρίζα μέσου σφάλματος τετραγώνου (Root Mean Squared Error – RMSE) που εκφράζεται στις ίδιες μονάδες μέτρησης με τις τιμές της χρονοσειράς. Η σχέση που υπάρχει μεταξύ MSE και RMSE θα μπορούσαμε να πούμε ότι είναι αντίστοιχη της σχέσεως που υπάρχει μεταξύ τυπικής απόκλισης και διακύμανσης. β) ο τρόπος με τον οποίο υπολογίζεται το MSE δίνει μεγάλη βαρύτητα στα μεγάλα παρά στα μικρά σφάλματα. Κατά συνέπεια η ύπαρξη προβλέψεων που απέχουν πολύ από τις αντίστοιχες πραγματικές τιμές γίνεται περισσότερο αισθητή με το κριτήριο MSE από ότι με το κριτήριο MAD, γιατί το σφάλμα της πρόβλεψης υψώνεται στο τετράγωνο. γ) Το κριτήριο MSE θεωρείται ότι είναι στατιστικά περισσότερο αξιόπιστο από το κριτήριο MAD.

ΜΕΣΟ ΑΠΟΛΥΤΟ ΠΟΣΟΣΤΙΑΙΟ ΣΦΑΛΜΑ (MEAN ABSOLUTE PERCENTAGE ERROR – MAPE)

Ένα άλλο κριτήριο αξιολόγησης των μεθόδων προβλέψεων είναι το μέσο απόλυτο ποσοστιαίο σφάλμα, το οποίο εξετάζει τη συμπεριφορά της απόλυτης τιμή του σφάλματος της πρόβλεψης σε σχέση με την πραγματική τιμή της χρονοσειράς. Το μέσο απόλυτο ποσοστιαίο σφάλμα ορίζεται ως το άθροισμα των απόλυτων τιμών των σφαλμάτων των χρονοσειρών προς τις αντίστοιχες πραγματικές τιμές της χρονοσειράς, διαιρούμενο με τον αριθμό των χρονικών περιόδων n στις οποίες έγιναν προβλέψεις. Δηλαδή :

$$\mathrm{M} = \frac{100\%}{n} \sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right|,$$

όπου At η πραγματική τιμή και Ft η εκτιμώμενη τιμή.

Σημείωση: μετράει το πόσο έξω πέφτουν οι προβλέψεις ως ποσοστά της πραγματικής τιμής της μεταβλητής

MAPE = 1 / n * [Σt(IetI / Yt)], Όπου t =1,2,…,n

Το συγκεκριμένο κριτήριο δείχνει πόσο μεγάλο είναι το σφάλμα της πρόβλεψης σε σχέση με τις πραγματικές τιμές της χρονοσειράς. Όσο μικρότερη είναι η τιμή που λαμβάνει τόσο πιο καλή θεωρείται η μέθοδος πρόβλεψης. Επειδή το κριτήριο MAPE είναι απαλλαγμένο από μονάδες μέτρησης, χρησιμοποιείται για να συγκρίνουμε την ακρίβεια μιας ή περισσοτέρων μεθόδων πρόβλεψης σε δύο ή περισσότερες χρονοσειρές.

ΜΕΣΟ ΠΟΣΟΣΤΙΑΙΟ ΣΦΑΛΜΑ (MEAN PERCENTAGE ERROR –MPE)

Όταν θέλουμε να προσδιορίσουμε κατά πόσο οι προβλεπόμενες τιμές είναι συστηματικά μεγαλύτερες ή μικρότερες από τις αντίστοιχες πραγματικές τιμές της χρονοσειράς, δηλαδή κατά πόσο η μέθοδος πρόβλεψης που εφαρμόζουμε είναι μεροληπτική ή όχι, μπορούμε να εφαρμόσουμε το μέσο ποσοστιαίο σφάλμα. Ο τρόπος υπολογισμού του κριτηρίου αυτού είναι ακριβώς ίδιος με τον τρόπο υπολογισμού του μέσου απόλυτου ποσοστιαίου σφάλματος, με τη μόνη διαφορά ότι εδώ δεν χρησιμοποιούμε τις απόλυτες αλλά τις πραγματικές τιμές του σφάλματος της πρόβλεψης. Δηλαδή :

MPE = 1 / n * [Σt(et / Yt)], Όπου t =1,2,…,n

Σημείωση: μέσο σφάλμα επί τοις 100 για Ν περιόδους.

Όταν οι τιμές που λαμβάνει το κριτήριο αυτό είναι κοντά στο μηδέν, τότε η μέθοδος πρόβλεψης θεωρείται αμερόληπτη. Αντιθέτως, μεγάλες τιμές του κριτηρίου καταδεικνύουν μεγάλη μεροληψία. Αρνητική τιμή του MPE σημαίνει ότι η μέθοδος πρόβλεψης παρέχει υπερεκτιμημένες προβλέψεις σε σχέση με τις πραγματικές τιμές, ενώ θετική τιμή του MPE σημαίνει ότι οι τιμές της χρονοσειράς είναι υποεκτιμημένες.

ΡΙΖΑ ΜΕΣΟΥ ΤΕΤΡΑΓΩΝΙΚΟΥ ΣΦΑΛΜΑΤΟΣ (Root Mean squared error, RMSE) δίνεται από τον τύπο:

$$RMSE = \sqrt{\frac{\sum_{t=1}^{N} E_t^2}{N}}$$

Συμπερασματικά, για να προσδιορίσουμε την αξιοπιστία μιας συγκεκριμένης μεθόδου πρόβλεψης, θα πρέπει να μελετήσουμε τη διαχρονική συμπεριφορά των τιμών των σφαλμάτων της πρόβλεψης. Αυτό γίνεται με την εφαρμογή διάφορων κριτηρίων, σύμφωνα με τα οποία αξιολογούμε τη χρησιμοποιούμενη μέθοδο πρόβλεψης. Κάθε ένα από τα κριτήρια αυτά ορίζεται από μία συγκεκριμένη συναρτησιακή σχέση των σφαλμάτων της πρόβλεψης και μπορεί να χρησιμοποιηθεί όχι μόνο για την αξιολόγηση μιας μεθόδου πρόβλεψης αλλά και για την επιλογή της “ καλύτερης ” μεταξύ δύο ή περισσοτέρων εναλλακτικών μεθόδων προβλέψεων.

Παρατήρηση: Τους μαθηματικούς τύπους στους οποίους αναφερθήκαμε παραπάνω και χρησιμοποιήσαμε στην εφαρμογή μας τους βρήκαμε μέσω του internet από την ιστοσελίδα του Wikipedia: http://en.wikipedia.org/wiki/Moving_average

Σύγκριση Μεθόδων Πρόβλεψης Τιμών Μετοχών

Moving – Cumulative average

Πλεονεκτήματα

- Εύκολα κατανοητή
- Υπολογίζεται εύκολα
- Παρέχει σταθερές προβλέψεις

Μειονεκτήματα

- Προϋποθέτει την αποθήκευση πολλών από τα προηγούμενα σημεία δεδομένων: τουλάχιστον οι N περίοδοι χρησιμοποιούνται για τον υπολογισμό του κινούμενου μέσου όρου
- Υστερεί σε σχέση με την τάση
- Αγνοεί πολύπλοκες σχέσεις των δεδομένων

Weighted Moving Averages

Αυτή η μέθοδος εξετάζει δεδομένα του παρελθόντος και προσπαθεί να αποδίδουν λογικά σημασία σε ορισμένα στοιχεία σε σχέση με άλλα στοιχεία

- Συντελεστές βαρύτητας πρέπει να προσθέσετε σε ένα
- Μπορεί να σταθμίσουμε τα τελευταία υψηλότερα από ό, τι παλαιότερα ή συγκεκριμένα στοιχεία πάνω από τα άλλα

- Σε περίπτωση πρόβλεψη στελέχωσης, θα μπορούσαμε να χρησιμοποιήσουμε τα δεδομένα από τις τελευταίες τέσσερις εβδομάδες, όπου οι Τρίτες είναι που θα προβλέψεις.
- Στάθμιση τις Τρίτες είναι: Τ-1 είναι .25; Τ-2 είναι .20; Τ-3 είναι .15; Τ-4 είναι .10 και Μέσος όρος όλων των άλλων ημερών ζυγίζεται .30

.**Exponential Smoothing Method**

Ένας τύπος του σταθμισμένου κινητού μέσου όρου που ισχύει για μείωση βαρών ιστορικών δεδομένων

1. . Νέα Πρόγνωση = a (πιο πρόσφατη παρατήρηση) + (1 - α) (τελευταία πρόβλεψη)
   ή
2. Νέα Πρόγνωση = τελευταία πρόβλεψη - ένα (το τελευταίο σφάλμα πρόβλεψης)

όπου 0 <α <1 και γενικά είναι μικρό για τη σταθερότητα των προβλέψεων (περίπου 0,1 - 0,2)
Σε σύμβολα:

$$F_{t+1} = a D_t + (1 - a) F_t$$

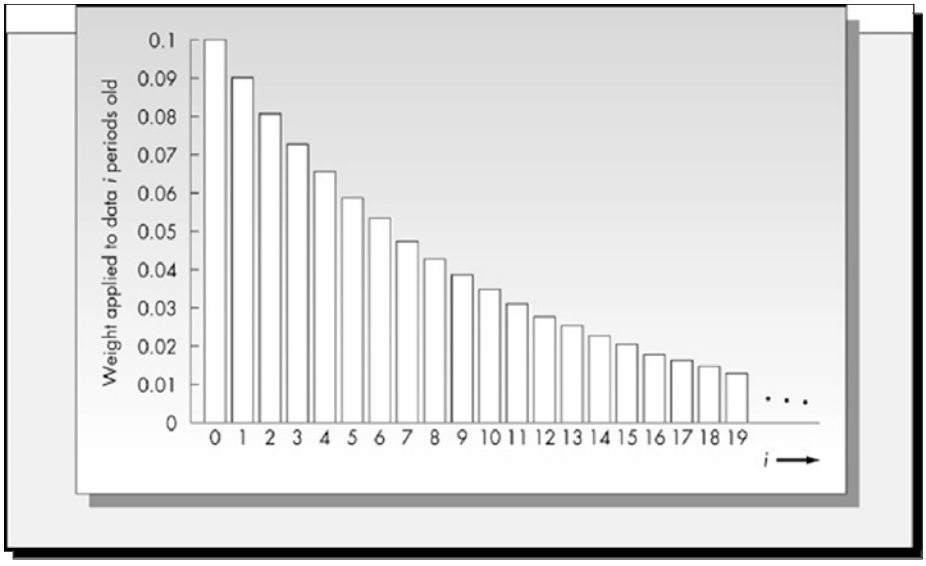
$$= a D_t + (1 - a)(a D_{t-1} + (1 - a) F_{t-1})$$

$$= a D_t + (1 - a)(a) D_{t-1} + (1 - a)2 (a) D_{t-2} + \ldots$$

Ως εκ τούτου, η μέθοδος αυτή εφαρμόζεται ένα σύνολο εκθετικά φθίνουσα βαρών σε ιστορικά δεδομένα. Είναι εύκολο να αποδειχθεί ότι το άθροισμα των βαρών είναι ακριβώς ένα.

$$\{Or : F_{t+1} = F_t - a (F_t - D_t) \}$$

Weights in Exponential Smoothing:



Επίδραση της α τιμής για την Πρόβλεψη

- Μικρές τιμές του α σημαίνει ότι η προβλεπόμενη τιμή θα είναι σταθερές (δείχνουν χαμηλή μεταβλητότητα)

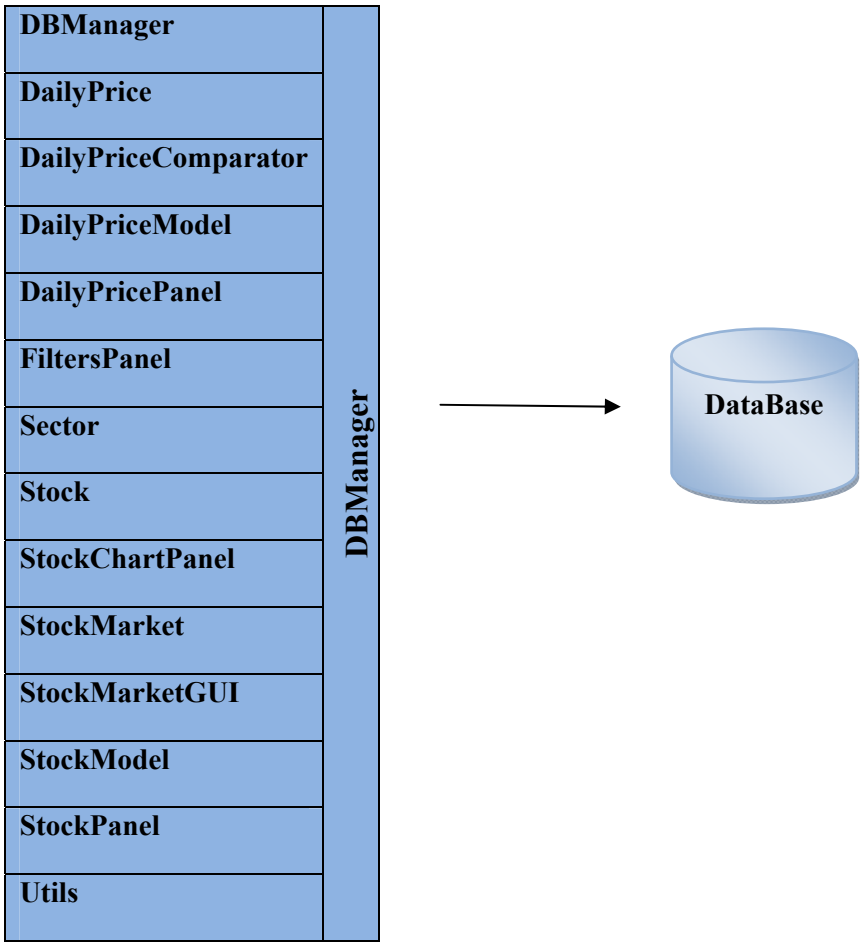
- Χαμηλή α αυξάνει την υστέρηση των προβλέψεων για τα πραγματικά δεδομένα, εάν η τάση είναι παρούσα
- Μεγάλες τιμές α σημαίνει ότι η πρόβλεψη θα παρακολουθούν πιο στενά την πραγματική χρονολογική σειρά

Ας δούμε ένα παράδειγμα:

Jan 23.3 Feb 72.3 Mar 30.3 Apr 15.5
And the January Forecast was: 25
Using α = .15
Forecast for Feb: αDjan + (1- α)Fjan = .15*23.3 + (.85)*25 = 24.745
Forecast for Mar: αDfeb + (1- α)Ffeb = .15*72.3 + (.85)*24.745 = 31.88
Apr: αDmar + (1- α)Fmar = .15*30.3 + .85*31.88 = 31.64
May: αDapr + (1- α)Fapr = .15*15.5 + .85*31.64 = 29.22

5.Ανάπτυξη Συστήματος Πρόβλεψης Τιμών Μετοχών

Package stockmarket:



Εδώ παρουσιάζεται με πλήρη ανάλυση η δομή της εφαρμογής πρόβλεψης μετοχών. Η ονομασία της εφαρμογής **είναι StockMarket** η οποία αποτελείται από διάφορες κλάσεις.

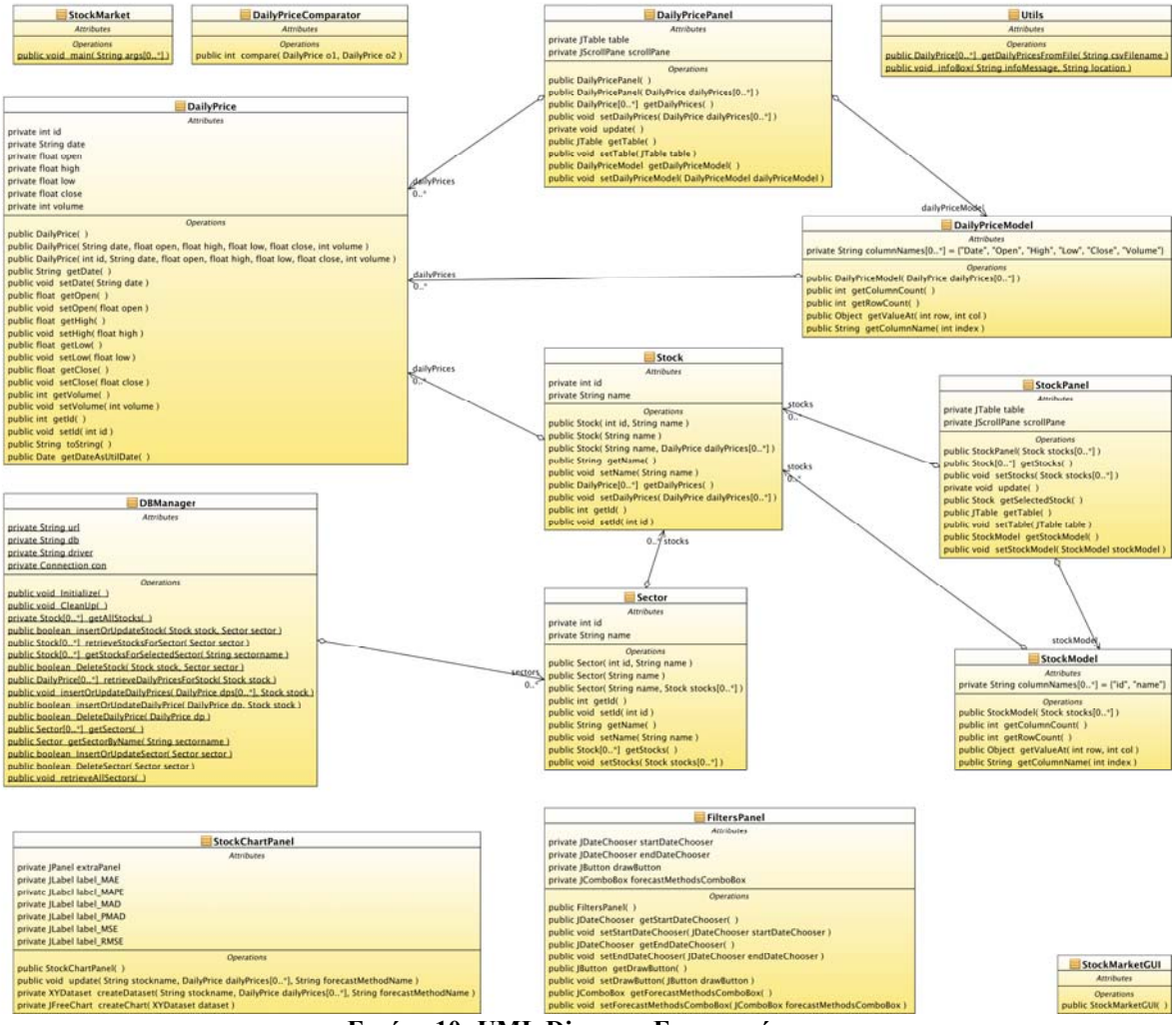
5.1 UML (Unified Modeling Language) – Class Diagrams

UML: Αναπαράσταση με οπτικό τρόπο (visualization) τμημάτων λογισμικού

Class Diagrams: Περιγράφουν την δομή ενός συστήματος με το να απεικονίζουν τις κλάσεις, τα πεδία και τις μεθόδους τους καθώς και τις σχέσεις μεταξύ των διάφορων κλάσεων

Γενικά:

• Οι κλάσεις, τα αντικείμενα και οι μεταξύ τους συσχετίσεις είναι τα πρωταρχικά στοιχεία μοντελοποίησης στην αντικειμενοστραφή θεώρηση
• Οι κλάσεις και τα αντικείμενα περιγράφουν τι υπάρχει μέσα στο σύστημα που περιγράφουμε
• Οι συσχετίσεις μεταξύ τους περιγράφουν πως δομούνται το ένα συστατικό σε σχέση με το άλλο
• Η ταξινόμηση χρησιμοποιείται εδώ και χρόνια για απλοποιημένη περιγραφή πολύπλοκων συστημάτων
• Όταν η υλοποίηση του συστήματος γίνεται σε μια αντικειμενοστραφή γλώσσα, οι κλάσεις και οι συσχετίσεις τους μεταμορφώνονται απ' ευθείας σε κώδικα προγραμματισμού.



Εικόνα 10: UML Diagram Εφαρμογής

Σε αυτό το κομμάτι της εργασίας μας θα αναφέρουμε τον τρόπο με τον οποίο δουλεύουν οι συνδέσεις στο UML διάγραμμα μας. Όπως και στις βάσεις δεδομένων όπου υπάρχουν συνδέσεις μεταξύ των διάφορων οντοτήτων έτσι και στην περίπτωση του uml υπάρχουν συνδέσεις μεταξύ των κλάσεων μας, γιατί η μια κλάση μπορεί να σχετίζεται με κάποια άλλη.

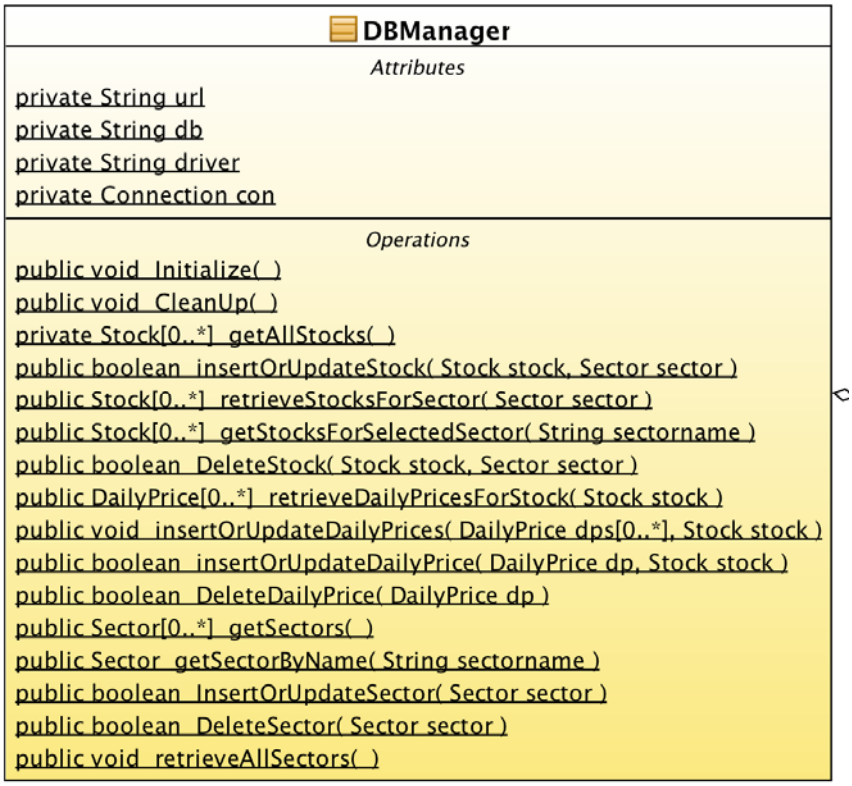
Αρχικά, λοιπόν βλέπουμε πως η **κλάση DBManager συνδέεται μέσω ενός composition με την κλάση Sector**. Αυτό σημαίνει πως η κλάση DBManager περιέχει μέσα της, από 0-πολλά Sectors (κατηγορίες). Λογικό είναι πως και η κλάση Sector θα περιέχει μέσα της, από 0-πολλές μετοχές. Όντως αυτό πραγματοποιείται αφού η DBManager εξαρτάται από Sectors και τα Sectors εξαρτώνται από Stocks (μετοχές). Αυτό το βλέπουμε και μέσα από την **σύνδεση που υπάρχει ανάμεσα στην κλάση Sector και στην κλάση Stock**. Τέλος, **υπάρχει η σύνδεση μεταξύ της κλάσης Stock με την κλάση DailyPrice**. Πολύ λογικό να υπάρχει το composition αφού η μετοχή αυτή κάθε αυτή περιέχει μέσα πολλές τιμές. Αυτές οι τιμές είναι τα DailyPrices γι' αυτό και συμβολίζονται με (0-*). Η χρησιμότητα αυτών των διασυνδέσεων είναι πολύ βασική, αν δεν υπήρχαν δεν θα υπήρχε δομή στην εφαρμογή μας και δεν θα έβγαινε ένα ολοκληρωμένο αποτέλεσμα.

Η κλάση DailyPricePanel είναι μια κλάση η οποία δημιουργεί ένα Panel. Μέσα σε αυτό το Panel τοποθετεί έναν πίνακα από DailyPrices, όπου μέσα σε αυτόν τον πίνακα εφαρμόζεται ένα DailyPriceModel για να υπάρχει ένα συγκεκριμένο μοντέλο (μία συγκεκριμένη δομή στον πίνακα από DailyPrices). Γι' αυτό υπάρχει και η σχετική σύνδεση ανάμεσα στην κλάση DailyPriceModel και στην κλάση DailyPrices.

Η κλάση StockPanel είναι ένα Panel που περιέχει μέσα του ένα πίνακα από μετοχές. Με τη σειρά της η StockPanel περιέχει ένα StockModel που μας δείχνει ότι το μοντελο που θα τοποθετούνται οι μετοχές είναι συμφωνα με το id και το name. Συμπερασματικά η κλάση StockModel θα περιέχει μέσα της πολλές μετοχές.

Παρακάτω αναλύουμε κάθε κλάση ξεχωριστα, εξηγούμε ποιος είναι ο ρόλος ύπαρξης της.

Η κλάση DBManager



Εικόνα 11: Class DBManager

Η κύρια κλάση είναι η **DBManager** η οποία περιέχει όλες τις μεθόδους για την επικοινωνία με την βάση και την χωρίζουμε σε 3 κατηγορίες. Στην κάθε μία κατηγορία περιέχονται οι κατάλληλες μεθόδου για την διαχείριση των μετοχών (**Stocks**), των ημερήσιων στοιχείων μιας μετοχής (**DailyPrices)** και των θεματικών κατηγοριών (**Sectors**).

Η μέθοδος **initialize()** χρησιμοποιείται για να γίνει η σύνδεση με την βάση. Η static μέθοδος **Class.forName** (static σημαίνει εδώ πως δεν χρειάζεται να φτιάξουμε ένα πραγματικό αντικείμενο της κλάσης Class για να χρησιμοποιήσουμε τη μέθοδο αυτή) χρησιμοποιείται για το φόρτωμα του οδηγού. Σημειώνεται εδώ πως πρέπει να βρούμε ποιος συγκεκριμένος οδηγός είναι ο κατάλληλος για τη βάση δεδομένων που χρησιμοποιούμε. Αν η κλάση δεν βρεθεί τότε προκαλείται μία **ClassNotFoundException** και το μπλοκ catch θα εμφανίσει ένα μήνυμα λάθους και θα τερματίσει το πρόγραμμα.

Αν το φόρτωμα του οδηγού ήταν επιτυχές στη συνέχεια θα γίνει σύνδεση με την βάση δεδομένων. Αυτό επιτυγχάνεται με την δήλωση:

Connection con = DriverManager.getConnection(url+db,"root","epp")

Δηλώνεται ένα νέο αντικείμενο Connection και χρησιμοποιείται η μέθοδος getConnection(), η οποία έχει τρία ορίσματα. Το δεύτερο είναι το όνομα χρήστη της βάσης δεδομένων και το τρίτο είναι ο αντίστοιχος κωδικός πρόσβασης. Το πρώτο όρισμα είναι ένα Ενιαίος Εντοπιστής Πόρων (Uniform Resource Locator, URL): **το URL είναι μία μορφή διεύθυνσης για αντικείμενα ή υπηρεσίες που υπάρχουν στον Παγκόσμιο Ιστό.** Παρακάτω παρουσιάζεται το URL της βάσης δεδομένων:

url = "jdbc:mysql://localhost:3306/";

Το πρώτο τμήμα του URL δηλώνει το πρωτόκολλο επικοινωνίας με τον διακομιστή βάσης δεδομένων (εδώ δηλώνεται το jdbc). Το επόμενο τμήμα προσδιορίζει σε τί τύπο βάσης δεδομένων αναφέρεται, στην περίπτωση του παραπάνω URL η βάση δεδομένων είναι μία συμβατή με το πρότυπο mysql βάση δεδομένων. Στη συνέχεια προσδιορίζεται ο υπολογιστής στον ποίο βρίσκεται η βάση δεδομένων.

Στο επόμενο κομμάτι του κώδικα η μέθοδος **createStatement** δημιουργεί μία δήλωση που μπορεί να χρησιμοποιηθεί **για την αποστολή δηλώσεων SQL στην βάση δεδομένων** και η μέθοδος **executeQuery** εκτελεί την δήλωση SQL που αποτελεί το όρισμά της. Στην συνέχεια μπορούμε να επεξεργαστούμε την απάντηση του ερωτήματος. Όπως είπαμε, η απάντηση θα έχει τη μορφή πίνακα.

Ο πίνακας αυτός αντιπροσωπεύεται στον κώδικα Java από ένα αντικείμενο **ResultSet** και όπως αναφέραμε πρέπει να διατρέξουμε τις γραμμές του μια - μια. Η μέθοδος next του αντικειμένου ResultSet θα προχωράει το δείκτη του ResultSet στην επόμενη εγγραφή και θα επιστρέφει την τιμή true όταν υπάρχει επόμενη διαθέσιμη εγγραφή. Πάντα μόνο μια εγγραφή του ResultSet είναι διαθέσιμη, συνεπώς πρέπει να χρησιμοποιήσουμε την next και να προχωράμε εγγραφή - εγγραφή μέχρι να φτάσουμε στην τελευταία. Μέσα στον κύκλο επεξεργασίας από την πρώτη ως την τελευταία εγγραφή χρησιμοποιούνται δύο μέθοδοι, η **getInt** και η **getString**. Καθεμιά από αυτές παίρνει μια ακέραιη παράμετρο που δηλώνει τον αριθμό στήλης του πίνακα του αποτελέσματος και θα επιστρέψει τα περιεχόμενα της τρέχουσας εγγραφής στην στήλη που έχει δηλωθεί. Ο τύπος δεδομένων που επιστρέφεται από την κλήση της μεθόδου είναι ο τύπος που δηλώνεται από το όνομα της μεθόδου. Για παράδειγμα, η getString(1) θα επιστρέψει τα περιεχόμενα της πρώτης στήλης της τρέχουσας εγγραφής του ResultSet με τη μορφή string. Παρόμοια, η getInt(2) θα επιστρέψει τον ακέραιο που βρίσκεται στην δεύτερη (τελευταία) στήλη της τρέχουσας σειράς του ResultSet. Πρέπει να σημειώσουμε πως πρέπει να προσέχουμε να ταιριάζουν οι συναρτήσεις που χρησιμοποιούμε με τους τύπους δεδομένων που υπάρχουν στη βάση δεδομένων. Επίσης εκτός από αυτές τις δυο υπάρχουν και άλλες αντίστοιχες μέθοδοι όπως οι getFloat, getDate κ.λ.π.

Τέλος, δημιουργείται ένας πίνακας μετοχών ***stocks = new ArrayList<>()*** και κάθε φορά που γίνεται η σύνδεση ανακτώνται οι μετοχές με την μέθοδο ***retrieveAllStocks().***

Μετά την επιτυχή δημιουργία ενός URL αντικειμένου η σύνδεση με αυτό επιτυγχάνεται με κλήση της μεθόδου openConnection. Όταν γίνεται η σύνδεση, αρχικοποιείται ένας δεσμός επικοινωνίας μεταξύ του δικού μας προγράμματος σε Java και του URL. Ακολουθεί η μέθοδος **CleanUp()** που χρησιμεύει στο να κλείσει την σύνδεση.

```
public static void Initialize() {
        url = "jdbc:mysql://localhost:3306/";
        db = "stockmarket";
        driver = "com.mysql.jdbc.Driver";
        con = null;

        try{
            Class.forName(driver);
            con = DriverManager.getConnection(url+db,"root","epp");
        }
        catch(ClassNotFoundException | SQLException e)
        {
            System.out.println("Unable to connect to db. Error: " +
e.getMessage());
        }
        sectors = new ArrayList<>();
        retrieveAllSectors();
    }

    public static void CleanUp() {
        try {
            con.close();
        } catch (SQLException ex) {
            Logger.getLogger(DBManager.class.getName()).log(Level.SEVERE, null,
ex);
        }
                                            }
```

Πίνακας 3 Μέθοδος Initialize

A) <u>Στην κατηγορία Stocks υλοποιούνται οι εξής μέθοδοι:</u>

Η μέθοδος **getAllStocks()** είναι μέθοδος με την οποία παίρνουμε όλες τις μετοχές για όλα τα Sector δηλαδή επιστρέφει τις μετοχές από τη βάση και τις προσθέτουμε στην εφαρμογή μας.

Η μέθοδος **insertOrUpdateStock(Stock stock , Sector sector)** χρησιμεύει στην εισαγωγή ή την ενημέρωση μιας μετοχής στην βάση. Πιο συγκεκριμένα, παίρνει σαν όρισμα μια μετοχή τύπου Stock και μια κατηγορία τύπου Sector στην όποια ανήκει και στη συνέχεια δημιουργεί ερωτήματα προς τη βάση ***"SELECT * FROM stock where id = " + stock.getId()*** φέρνοντας από αυτήν όλες τις μετοχές με το συγκεκριμένο id. Αν υπάρχει ήδη τότε προχωρά στην ενημέρωση των στοιχείων της αλλιώς προχωρά στην δημιουργία της ισοδύναμης εγγραφής στην βάση. Η ***stock.getId()*** μας επιστρέφει το id μετοχής που έχουμε εισάγει και στη συνέχεια βάζει το όνομα της μετοχής και το id στην επόμενη γραμμή του πίνακα των μετοχών. Αν υπάρχει ήδη η μετοχή τότε την κάνει update στη βάση και μετά μας το εμφανίζει ενώ αν η μετοχή δεν υπάρχει τότε την εισάγει στη βάση ***pstatement = con.prepareStatement(INSERT_SQL,Statement.RETURN_GENERATED_KEYS)*** και μας επιστρέφει τα γενόμενα κλειδιά. Αν συμβεί κάποιο λάθος μας εμφανίζει μήνυμα με τη βοήθεια της εντολής. ***System.out.println("SQL statement is not executed! Error: " + s.getMessage ())***.

Η μέθοδος **retrieveStocksForSector(Sector sector)** είναι εκείνη που θα μιλήσει με τη βάση για το που ανήκει το συγκεκριμένο stock δηλαδή σε ποια κατηγορία. Για παράδειγμα, θέτουμε ερωτήματα ***SELECT * FROM stock where sectorid = " + sector.getId()*** και μας επιστρέφει το id και το name της μετοχής και για την συγκεκριμένη κατηγορία Τέλος ανακτά τα ιστορικά στοιχεία της επιλεγμένης μετοχής ***stock.setDailyPrices(retrieveDailyPricesForStock(stock));.***

Η μέθοδος **getStocksForSelectedSector(String sectorname)** μας βοηθάει στην κατηγοριοποίηση. Αν επιλέξουμε την κατηγορία All Sectors μας εμφανίζει όλες τις μετοχές, ενώ αν επιλέξουμε κάποια άλλη κατηγορία θα μας επιστρέψει τις μετοχές της συγκεκριμένης κατηγορίας που επιλέξαμε.

Η μέθοδος **DeleteStock(Stock stock, Sector sector)** μας επιτρέπει να διαγράψουμε μετοχές. Πιο συγκεκριμένα, στέλνει ένα ερώτημα στη βάση δεδομένων λέγοντας να διαγράψει αυτό που του ζητάμε ***String DELETE_SQL = "DELETE FROM stock WHERE id = ?";.***

Β) Στην δεύτερη κατηγορία ανήκουν οι μέθοδοι των DailyPrices:

Με την μέθοδο **retrieveDailyPricesForStock(Stock stock)** στέλνουμε ένα query στη βάση δεδομένων λέγοντας της να μας φέρει τα ιστορικά στοιχεία των μετοχών που ζητάμε, ταξινομημένα με βάση την ημερομηνία ***statement = con.prepareStatement("SELECT * FROM dailyprice where stockid = ? ORDER BY date").*** Αν αυτό δεν γίνει εφικτό μας πετάει exception.

Η μέθοδος **insertOrUpdateDailyPrices(ArrayList<DailyPrice> dps, Stock stock)** παίρνει σαν όρισμα ένα πίνακα από ιστορικά στοιχεία και μια μετοχή. Για κάθε γραμμή του πίνακα, δηλαδή για κάθε DailyPrice, καλεί την μέθοδο ***insertOrUpdateDailyPrice(dp,stock)*** και κάνει update τα dailyPrices αν η μετοχή υπάρχει ήδη ή εισαγωγή dailyPrices αν η μετοχή δεν υπάρχει.

Η συγκεκριμένη μέθοδος **insertOrUpdateDailyPrice(DailyPrice dp, Stock stock)** ψάχνει όλο το πίνακα με τα ιστορικά στοιχεία και αν βρει την μετοχή που του ζητάμε την κάνει update για να ανανεώσει κάποια πιθανή τιμή. Αν τυχόν δεν την βρει τότε κάνει εισαγωγή της μετοχής με τα ιστορικά της στοιχεία

Η μέθοδος **DeleteDailyPrice(DailyPrice dp)** χρησιμεύει στο να διαγράψουμε ένα dailyPrice. Αυτό πραγματοποιείται θέτοντας ένα ερώτημα στη βάση. Σε αυτό το ερώτημα καθιστούμε σαφές πιο dailyPrice θέλουμε να διαγράψουμε δίνοντας το id του, ***String DELETE_SQL = "DELETE FROM dailyPrice WHERE id = ?".*** Αν κάτι πάει στραβά τότε θα μας εμφανίσει μήνυμα λάθους μέσω του exception που υπάρχει.

Γ) Η τελευταία κατηγορία ανήκει στα Sectors δηλαδή τις κατηγορίες που έχουμε χωρίσει τις μετοχές μας για παράδειγμα υπάρχει η κατηγορία Banking, Capital goods κα. Παρακάτω υλοποιούνται οι παρακάτω μέθοδοι:

Η μέθοδος **getSectors()** λειτουργεί ουσιαστικά σαν έναν getter, όπου μας επιστρέφει όλους τους Sectors.

Η μέθοδος **getSectorByName(String sectorname)** χρησιμεύει στο να επιστρέφει τα Sectors ανάλογα με το όνομα κατηγορίας που έχουμε επιλέξει. Αφού ψάξει στον πίνακα με τα Sectors, αν δεν βρει αυτό που του ζητάμε επιστρέφει null.

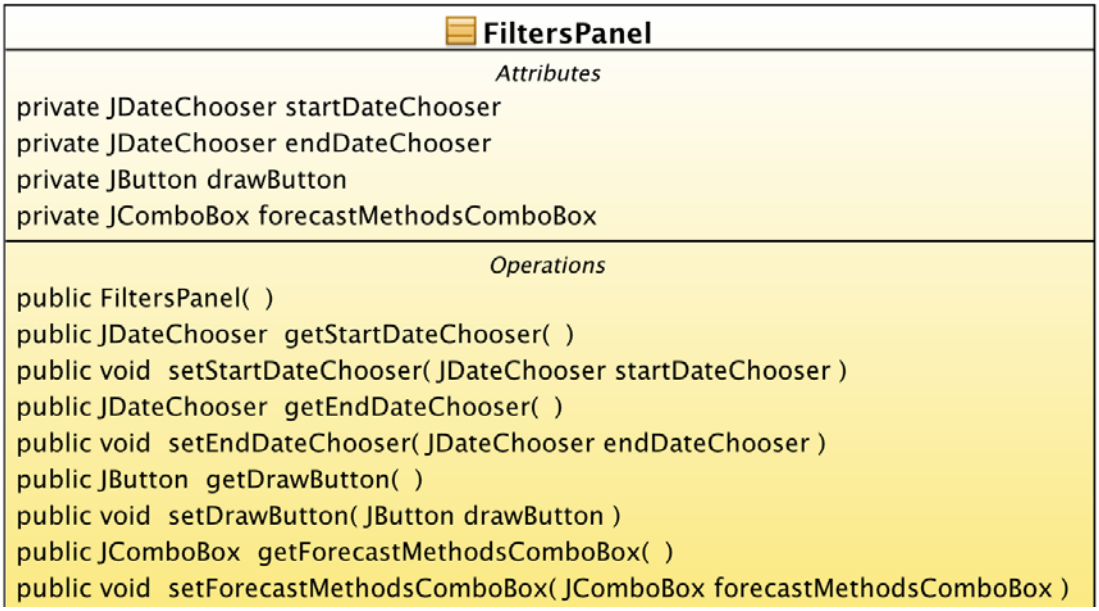
Η μέθοδος **InsertOrUpdateSector(Sector sector)** λειτουργεί το ίδιο όπως και η InsertOrUpdateDailyPrices. Πιο συγκεκριμένα, ψάχνει μέσα στη βάση το Sector που ζητάμε αν το βρει κάνει update, για να ανανεώσει τυχών αλλαγές, ειδάλλως κάνει insert.

Όπως και στα Dailyprices έτσι κι εδώ έχουμε τη δυνατότητα να διαγράψουμε ένα sector. Αυτό γίνεται με τη βοήθεια της μεθόδου **DeleteSector(Sector sector),** όπου θέτοντας του το ερώτημα στην βάση, διαγράφει την κατηγορία που επιλέξαμε.

Η συγκεκριμένη μέθοδος **retrieveAllSectors()** υλοποιείται για το σκοπό του ότι «μιλάει» με την βάση και παίρνει όλα τα δεδομένα(id και name) του πίνακα sectors μέσω μιας επανάληψης while.

Κάνουμε χρήση των exception, σε περίπτωση που έχουμε κάνει κάποιο λάθος, ώστε να μας ενημερώσει.

Η κλάση FiltersPanel



Εικόνα 12: Class FiltersPanel

Η κλάση **FilterPanel** έχει τα εξής χαρακτηριστικά, ένα ***startDateChooser*** και ένα ***endDateChooser*** τύπου ***JDateChooser***, όπου βάση αυτών επιλέγουμε την χρονική περίοδο της πρόβλεψης. Επίσης έχει ένα ***drawButton*** τύπου ***JButton*** που βοηθάει στην εμφάνιση της γραφικής παράστασης των τιμών των μετοχών στο Historical Chart. Τέλος ένα ***forecastMethodsComboBox*** τύπου ***JComboBox*** που με την βοήθεια του επιλέγουμε το φίλτρο, που με βάση αυτό θα ζωγραφιστεί η πρόβλεψη των τιμών της μετοχής που επιλέξαμε. Οι μέθοδοι/κατασκευαστές που έχει αυτή η κλάση είναι η ***FiltersPanel()*** που μας βοηθά να εισάγουμε τα παραπάνω χαρακτηριστικά στο γραφικό περιβάλλον της εφαρμογή μας. Οι υπόλοιπες μέθοδοι μας βοηθούν ώστε να διαβάσουμε ή να αλλάξουμε τις τιμές των χαρακτηριστικών της κλάσης. Η κλάση κληρονομεί από το JPanel (είναι ένα container αντικείμενο στο οποίο μπορούν να τοποθετηθούν component αντικείμενα) αφού και η ίδια είναι ένα μικρό Panel που θέλουμε να πάρει τις ιδιότητες και τις μεθόδους του JPanel και να τις χρησιμοποιήσουμε κατάλληλα στην υλοποίηση της εφαρμογής μας.

Η κλάση DailyPrice

Εικόνα 13: Class DailyPrice

Η κλάση **DailyPrice** περιέχει τα δεδομένα της μετοχής, δηλαδή το id της μετοχής, την ημερομηνία που άνοιξε η μετοχή (**date**), την τιμή που πήρε η μετοχή τη στιγμή που άνοιξε (**open**),την υψηλότερη και την χαμηλότερη τιμή που πήρε η μετοχή μια συγκεκριμένη ημερομηνία (**high** και **low**), την τιμή που πήρε η τιμή την στιγμή που έκλεισε (**close**) και το πλήθος των μετοχών (**volume**). Αυτά είναι τα δεδομένα που αποθηκεύονται στην βάση. Ουσιαστικά είναι ένας πίνακας με ιστορικά στοιχεία.

Η παραπάνω κλάση έχει τρεις constructor **DailyPrice**(), ένα για την περίπτωση που ο πίνακας με τα ιστορικά στοιχεία της μετοχής είναι κενός, έναν δεύτερο για την περίπτωση που του εισάγουμε τα δεδομένα και έναν τρίτο που του έχουμε δώσει τα δεδομένα και θέλουμε να πάρουν και ένα **id**. Επίσης έχουμε και μεθόδους που μας βοηθούν ώστε να διαβάσουμε ή να αλλάξουμε (getters/setters) τις τιμές των ιδιοτήτων της κλάσης (*π.χ με την **getDate()** διαβάζουμε την τιμή του **Date** ενώ με την μέθοδο **setDate(String date)** αλλάζουμε την τιμή του **Date***).

Αντί για την μέθοδο ***toString()***, μπορούμε να χρησιμοποιήσουμε **System.out.println()** για να τυπώσουμε κάποιο αντικείμενο. Όμως για καλά αποτελέσματα η κλάση μας πρέπει να έχει μια μέθοδο **toString()**, που να μορφοποιεί τα δεδομένα των αντικειμένων με ένα λογικό τρόπο και να επιστρέφει ένα **string**. Στη συγκεκριμένη περίπτωση τυπώνει στην οθόνη τα χαρακτηριστικά της μετοχής (Date,Open,Close,High,Low,Volume) .

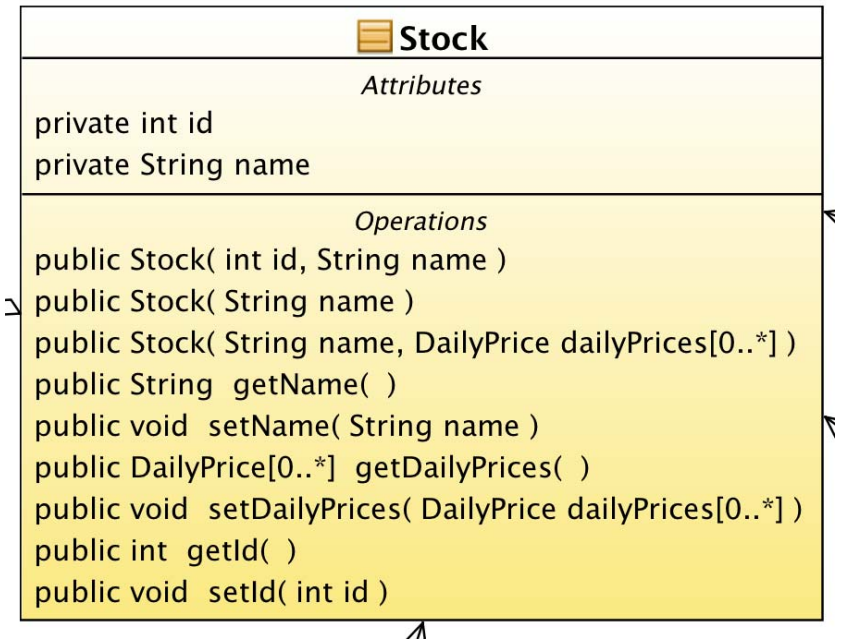
Τέλος έχουμε μία extra μέθοδο την **getDateAsUtilDate()** η οποία μετατρέπει την ημερομηνία της μετοχής από απλό κείμενο σε στιγμιότυπο της κλάσης **java.util.Date** με την βοήθεια του μεταφραστή **InputFormatter**.

Η κλάση StockMarket

Εικόνα 14: Class StockMarket

Η κλάση **StockMarket** είναι η main δηλαδή το entry point του προγράμματος μας που αρχικοποιεί την σύνδεση με την βάση δεδομένων και δημιουργεί το γραφικό περιβάλλον της διεπαφής μας. Βάζουμε έναν ***windowstatelinstener()*** ο οποίος "ακούει" αν το παράθυρο κλείσει για να τερματίσει σωστά την σύνδεση με την βάση.

Η κλάση Stock

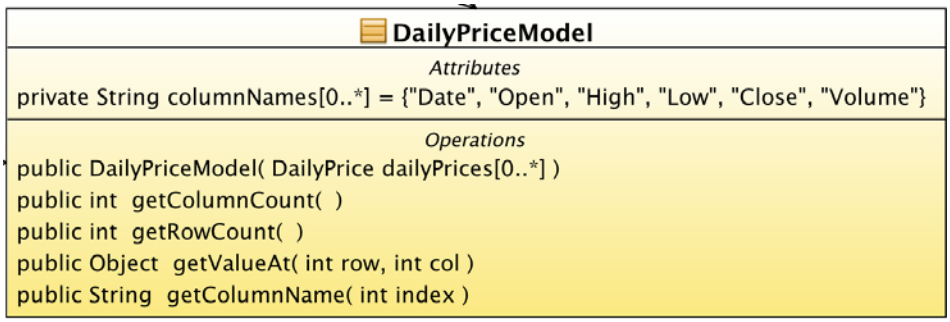


Εικόνα 15: Class Stock

Η κλάση **Stock** αντιπροσωπεύει την μετοχή. Περιέχει το **όνομα**, το **id** της μετοχής και ένα πίνακα από **dailyPrices** που θα πάνε στην βάση. Έχουμε ένα constructor με μοναδικό όρισμα το name, για την περίπτωση όπου ο χρήστης πληκτρολογεί το όνομα της μετοχής και στη συνέχεια πατάει το κουμπί create a new Stock (αυτό γίνεται κατά την επιλογή του χρήστη να εισάγει μια νέα μετοχή στην βάση). Ο δεύτερος όταν έχουμε εισάγει το όνομα της μετοχής και έχει πάρει και ένα **id** αλλά δεν έχουν εισαχθεί ακόμα τα δεδομένα της και ένας τρίτος constructor χρησιμοποιείται όταν έχουμε ήδη εισάγει και αποθηκεύσει την μετοχή μαζί με τα ιστορικά της δεδομένα (dailyPrices).

Τέλος υπάρχουν και οι μέθοδοι που μας βοηθούν να διαβάσουμε και να αλλάξουμε (getters/setters) τις ιδιότητες της κλάσης.

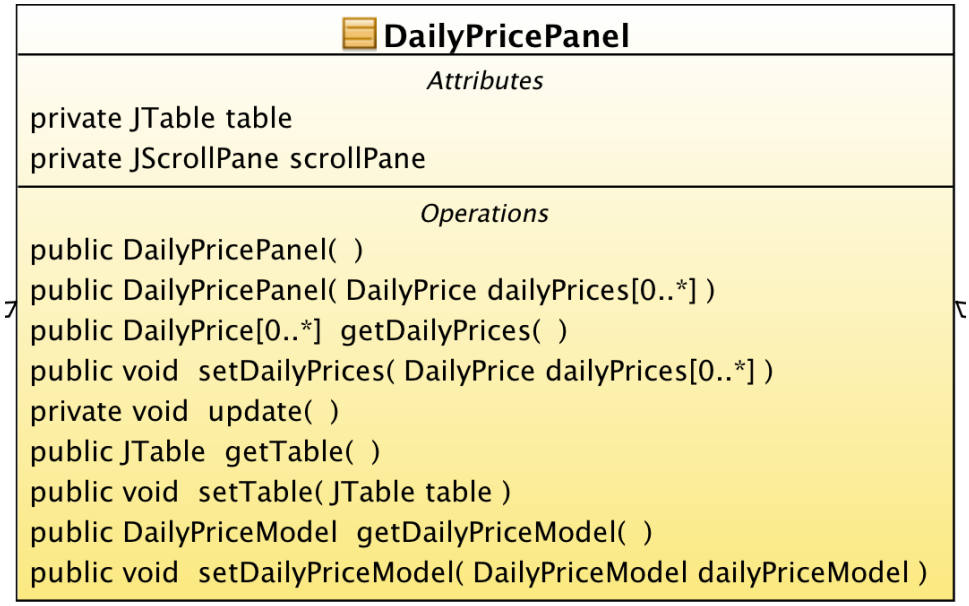
Η κλάση DailyPriceModel



Εικόνα 16: Class DailyPriceModel

Η κλάση **DailyPriceModel** κληρονομεί από την κλάση **AbstractTableModel**, η Sun είχε προβλέψει την ανάγκη να γράψει ένα μοντέλο πίνακα (**TableModel**) για να αποφύγει τα προβλήματα ορισμού μεγέθους ενός πίνακα. Με την βοήθεια της **AbstractTableModel** η οποία υλοποιεί την διεπαφή (**interface**) **TableModel** και χειρίζεται τους listeners για εμάς. Οι μέθοδοι που υλοποιούνται από την **AbstractTableModel** είναι η ***getColumnCount()*** , η ***getRowCount(),*** η ***getValueAt(int row, int col)*** και η ***getColumnName(int index),*** ελέγχοντας έτσι όλα τα στοιχεία του πίνακα. Αρχικά ορίσαμε που θα αποθηκευτούν τα δεδομένα. Χρησιμοποιήσαμε ένα **java.util.ArrayList** για την αποθήκευση των δεδομένων και βάση αυτών των δεδομένων θα "ζωγραφιστεί" ο πίνακας ***JTable private String[] columnNames = {"Date", "Open", "High", "Low", "Close", "Volume"};.*** Στη συνέχεια με τη βοήθεια μιας switch δώσαμε τις τιμές που θα πάρει κάθε γραμμή και στήλη. Τέλος προσθέσαμε τον πίνακα ArrayList στο table model και στη συνέχεια εφαρμόσαμε τις μεθόδους που χρειαζόμασταν.

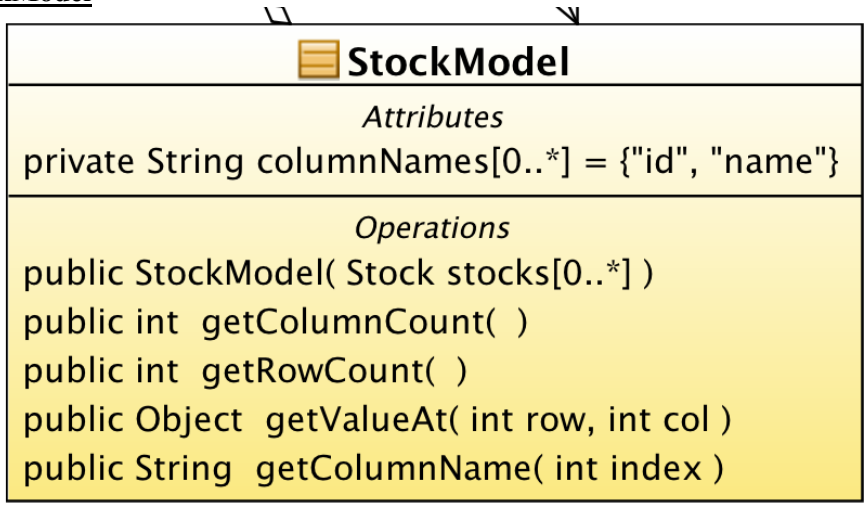
Η κλάση DailyPricePanel



Εικόνα 17: Class DailyPricePanel

Η κλάση **DailyPricePanel** κληρονομεί από την κλάση **JPanel**, είναι ένα component με ένα πίνακα που περιέχει τα δεδομένα των μετοχών. Σ' αυτό το σημείο αρχίζει η γραφική αναπαράσταση του προγράμματος μας και πιο συγκεκριμένα η εμφάνιση του πίνακα με τις τιμές των μετοχών. Χρησιμοποιώντας τους κατάλληλους layout managers δίνουμε την διαμόρφωση του παραθύρου μας ***this.setLayout(new BorderLayout());*** . Ορίζουμε τις διαστάσεις του και ενεργοποιούμε την λειτουργία scrolling στο παράθυρο ***this.scrollPane = new JScrollPane(table).*** Έπειτα υλοποιούμε μια σειρά από μεθόδους όπως ***getDailyPrices()*** όπου παίρνουμε τις τιμές από τη βάση και στη συνέχεια με την μέθοδο ***setDailyPrices()*** τις βάζουμε στο πρόγραμμα μας. Τέλος, κάνουμε update για την περίπτωση που έχουν αλλάξει τα ιστορικά της στοιχεία, για παράδειγμα όταν επιλέγουμε διαφορετική μετοχή.

Η κλάση StockModel

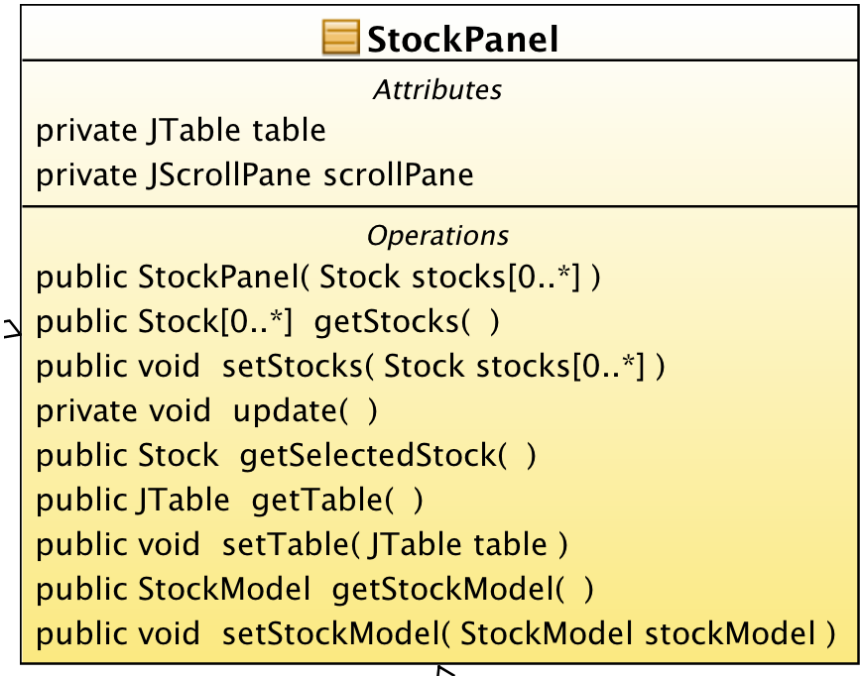


Εικόνα 18: Class StockModel

Η κλάση **stockModel** κληρονομεί από την **AbstractTableModel** τα πεδία και τις μεθόδους της. Ορίσαμε έναν **ArrayList** πίνακα όπου εκεί θα αποθηκεύονται οι μετοχές που εισάγουμε δηλαδή το id και το όνομα της μετοχής. Με την βοήθεια της **getValueAt**() βάζουμε τις τιμές στον πίνακα όπου στη πρώτη στήλη αποθηκεύεται το **id** και στη δεύτερη το **name** της μετοχής.

```
if(col==0) {
    return stocks.get(row).getId();
}
else if(col ==1) {
    return stocks.get(row).getName();
}
```

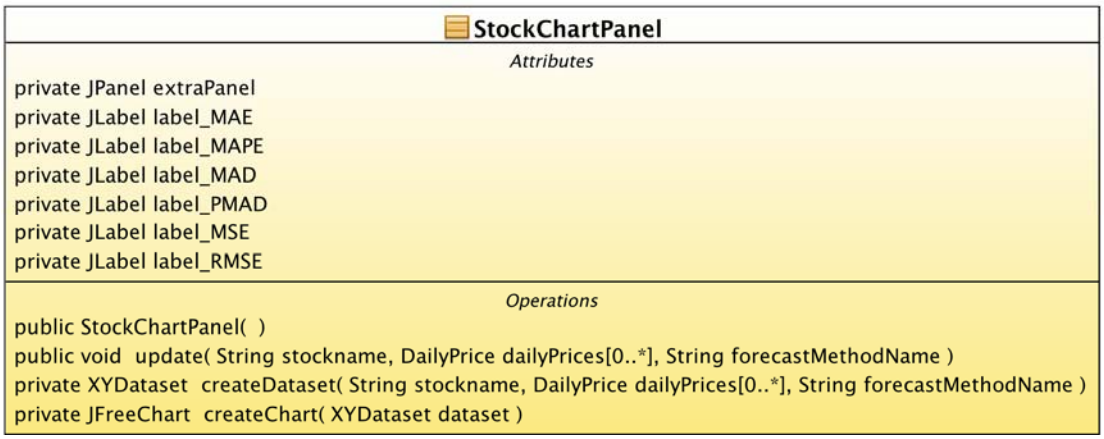
Η κλάση StockPanel



Εικόνα 19: Class StockPanel

Όπως στην κλάση **DailyPricePanel** έτσι και στην κλάση **StockPanel** δημιουργούμε το γραφικό περιβάλλον των μετοχών. Έτσι λοιπόν ορίζουμε ένα layout manager **this.setLayout(new BorderLayout());** και τον ορίζουμε στο κέντρο. Φτιάχνουμε επίσης τις διαστάσεις που επιθυμούμε και θέτουμε τη δυνατότητα scrolling. Στις επόμενες μεθόδους παίρνουμε τις μετοχές από την βάση και τις βάζουμε στο πρόγραμμα μας. Αυτό γίνετε με τη βοήθεια των μεθόδων **getStocks()** και **setStocks(ArrayList<Stock> stocks)**. Με την μέθοδο **getSelectedStock()** μαθαίνουμε την επιλεγμένη μετοχή από τον πίνακα με τις μετοχές. Το ότι εμφανίζει **exception** είναι κάτι το οποίο το αξιοποιούμε στην περίπτωση όπου μόλις άνοιξε το πρόγραμμα και το πρώτο πράγμα που έκανε ο χρήστης ήταν να κάνει import ιστορικά δεδομένα από αρχείο, χωρίς να έχει πρώτα επιλέξει μετοχή. Έπειτα χρησιμοποιούμε και την μέθοδο **update()** για να ανανεώσουμε τον πίνακα των μετοχών σε περίπτωση που έχουμε προσθέσει, μετονομάσει ή διαγράψει κάποια μετοχή. Τέλος, υπάρχουν και οι μέθοδοι που μας βοηθούν να διαβάσουμε ή να τροποποιήσουμε **(getters/setters)** τα χαρακτηριστικά της κλάσης.

Η κλάση StockChartPanel



Εικόνα 20: Class StockChartPanel

Η κλάση **StockChartPanel** κληρονομεί από την κλάση **JPanel** είναι ένα component που περιέχει την γραφική παράσταση της πρόβλεψης των τιμών των μετοχών. Αυτό το πετυχαίνουμε με την βοήθεια της **JFreeChart** βιβλιοθήκης. Στην κλάση αυτή έχουμε ορίσει ένα **GridLayout** manager προσθέτοντας σε αυτόν ετικέτες (Labels) όπως την **MAE, MAPE, MAD, PMAD, MSE** και **RMSE** για την εμφάνιση των ποσοστών αποτυχίας για το συγκεκριμένο δείκτη τεχνικής ανάλυσης (φίλτρο πρόβλεψης) που έχουμε επιλέξει.

Όπως και στις παραπάνω κλάσεις έτσι και εδώ χρησιμοποιούμε την μέθοδο **update()** για την περίπτωση που επιλέξουμε κάποια άλλο δείκτη τεχνικής ανάλυσης (φίλτρο πρόβλεψης). Η επόμενη μέθοδος που χρησιμοποιούμε είναι η ***createDataset()*** που παίρνει σαν ορίσματα το όνομα της μετοχής, ένα πίνακα με τα ιστορικά στοιχεία της μετοχής και το όνομα του δείκτη τεχνικής ανάλυσης (φίλτρου) που χρησιμοποιούνται για την πρόβλεψη. Σε αυτή την μέθοδο πραγματοποιούνται οι μαθηματικές πράξεις κάθε δείκτη. Τέλος, με την βοήθεια της μεθόδου CreateChart() δημιουργούμε την γραφική παράσταση της εφαρμογής μας ***final JFreeChart chart = ChartFactory.createTimeSeriesChart("Historical Chart", "Day", "Close", dataset, true, true, true);*** και με την οποία ορίζουμε επίσης στον άξονα τον X να φαίνονται οι ημερομηνίες. Η μέθοδος αυτή επιστρέφει την τελική μορφή της γραφικής παράστασης.

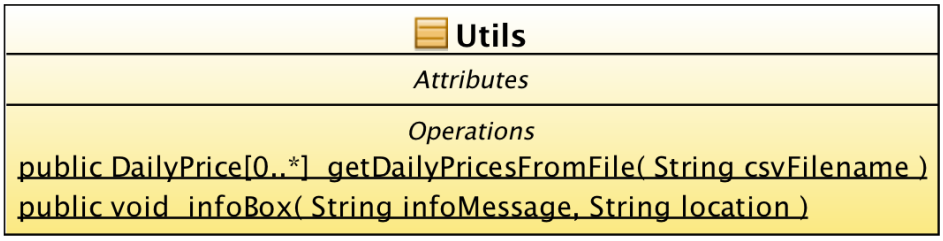
Η κλάση StockMarketGUI

Εικόνα 21: Class StockMarketGUI

Η κλάση **StockMarketGUI** είναι το κύριο γραφικό περιβάλλον της εφαρμογής μας και κληρονομεί από την **JFrame** (top level container), ορίζουμε ένα τίτλο, το μέγεθος του παραθύρου, ένα κύριο menubar ***JMenuBar menubar = new JMenuBar();*** που περιέχει το menu των κατηγοριών και το menu των μετοχών με τις επιλογές που διαθέτουν. Αυτές, οι επιλογές, είναι η δημιουργία, η μετονομασία, η διαγραφή ενός **Sector** ή **Stock** αντίστοιχα και η εισαγωγή των μετοχών. Στη συνέχεια, δημιουργούμε ένα combo box ***final JComboBox sectorComboBox = new JComboBox();*** στο οποίο προσθέτουμε τις κατηγορίες των μετοχών ***sectorComboBox.addItem(sector.getName());.*** Μέσω ενός **ActionListener** όταν επιλέγουμε κάποια κατηγορία μας εμφανίζει τις μετοχές που ανήκουν στη συγκεκριμένη κατηγορία. Στο παράθυρο μας δημιουργούμε ένα στιγμιότυπο τύπου **DailyPricePanel**, ένα **StockChartPanel** και ένα filtersPanel. Από το filtersPanel παίρνουμε το κουμπί Draw και του προσθέτουμε έναν **ActionListener** για να εκτελεστεί κάποια ενέργεια όταν το κάνουμε ‘‘κλικ’’.

Το κουμπί **“Import CSV data to selected stock”** εμφανίζει ένα file dialog με το οποίο ο χρήστης επιλέγει ένα αρχείο τύπου .csv το οποίο περιέχει τα ιστορικά στοιχεία μιας μετοχής και κατόπιν τα διαβάζει και προσθέτει στην βάση τα αντίστοιχα DailyPrices

Η κλάση Utils



Εικόνα 22: Class Utils

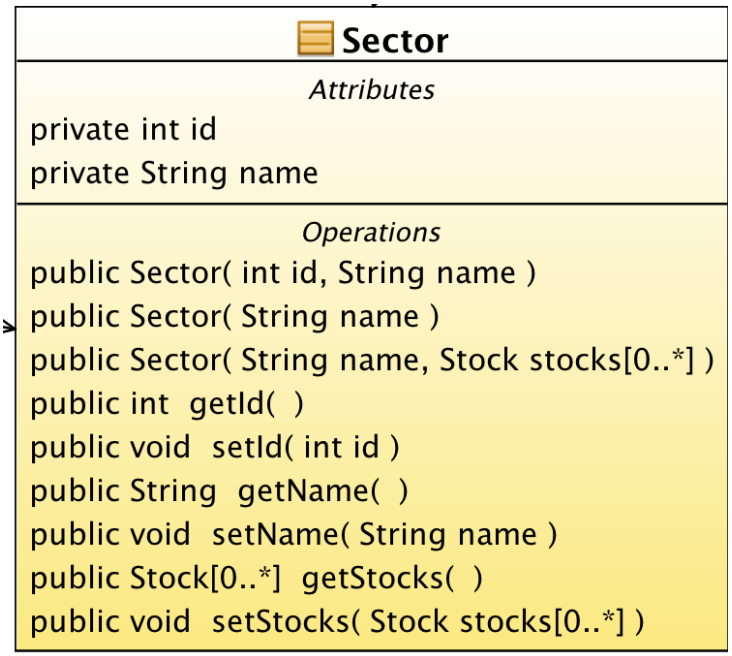
Η κλάση **Utils** χρησιμεύει στην εισαγωγή των .csv αρχείων μέσω της java και την αποστολή των δεδομένων που περιέχουν στη βάση δεδομένων. Επαναληπτικά διαβάζουμε κάθε γραμμή του αρχείου και όταν το αρχείο φτάσει στο τέλος του η μέθοδος **readNext**() επιστρέφει null, η συνθήκη γίνεται false και ως εκ τούτου σταματάει η επανάληψη και επιστρέφει τα dailyPrices. Επίσης έχουμε μία μέθοδο την **infoBox**() που μέσω αυτής εμφανίζουμε τα μηνύματα διαλόγου "Done" , "Den dialekses stock" ή άλλα μηνύματα μέσω ενός pop-up window.

Κλάση DailyPriceComparator

Εικόνα 23: Class DailyPriceComparator

Η κλάση **DailyPriceComparator** χρησιμεύει για να συγκρίνει τις ημερομηνίες των μετοχών, αυτό υλοποιείται από την μέθοδο **compare()** που δέχεται σαν όρισμα δύο στιγμιότυπα τύπου **DailyPrice**. Παίρνει από κάθε στιγμιότυπο την ημερομηνία ώστε να βρει την προγενέστερη ημερομηνία της μετοχής. Για το λόγο ότι η ημερομηνία στη java είναι τύπου **String** προσπαθεί να δημιουργήσει ένα τύπου **java.util.Date** αντίστοιχο της java.
Η μέθοδος αυτή επιστρέφει το αποτέλεσμα της σύγκρισης στη βάση, η βάση τα βρίσκει και τα στέλνει πίσω. Αυτή η μέθοδος χρειάζεται όταν θέλουμε να ταξινομήσουμε τις πληροφορίες για κάθε μετοχή γιατί μπορεί η βάση να μην τα στείλει ταξινομημένα.

Η κλάση Sector



Εικόνα 24: Class Sector

Με την κλάση Sector κατηγοριοποιούμε τις μετοχές. Έτσι δηλώνουμε ένα **id** που αντιστοιχεί στη βάση, ένα name και λίστα από μετοχές οι οποίες ανήκουν σε αυτή την κατηγορία. Έχουμε τρεις κατασκευαστές για τρεις περιπτώσεις α) στην αρχή όταν θέλουμε να προσθέσουμε ένα όνομα κατηγορίας (**Sector**), β) όταν έχουμε δώσει το όνομα και ξέρουμε το **id**, πριν προσθέσουμε τις μετοχές και γ) όταν έχουμε προσθέσει το όνομα της κατηγορίας και θέλουμε να προσθέσουμε και τις μετοχές. Τέλος, υπάρχουν και οι μέθοδοι που μας βοηθούν να διαβάσουμε ή να τροποποιήσουμε τα χαρακτηριστικά της κλάσης (**getters/setters**).

5.2 Εγχειρίδιο Χρήσης

Η εφαρμογή μας έχει σχεδιαστεί με σκοπό την πρόβλεψη τιμών μετοχών. Παρακάτω παρουσιάζουμε το γραφικό περιβάλλον της και τον τρόπο λειτουργίας του.

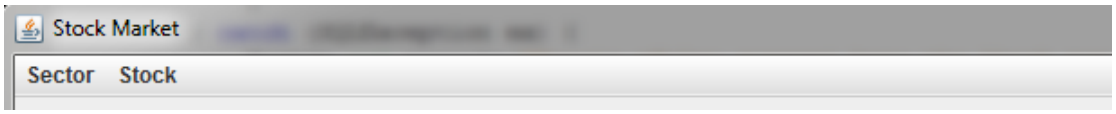
Απαιτήσεις

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.

Επίσης, για να τρέξει η συγκεκριμένη εφαρμογή που δημιουργήσαμε σε οποιονδήποτε υπολογιστή, θα πρέπει να εισάγουμε την βάση στο σύστημα μας, δηλαδή το αρχείο **stock.sql** μέσω του προγράμματος Mysql Workbench. Ακολουθώντας τα παρακάτω βήματα: Mysql Workbench → Server Administration → Data Import/Restore → Import from Self-Contained File → Επιλέγουμε το αρχείο .sql → Start Import, με αυτό τον τρόπο πετυχαίνουμε την επικοινωνία της Java με την Βάση.

Οδηγίες χρήσης του προγράμματος

Στην αρχή του προγράμματος βρίσκεται το κυρίως μενού :



Εικόνα 25: Μενού προγράμματος

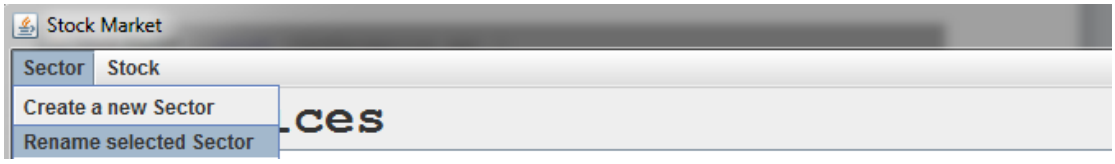
Αυτά τα μενού πραγματοποιούν κάποιες λειτουργίες της εφαρμογής.

- Το μενού **«Sector»** έχει τις εξής επιλογές**:**

1) **Sector -> Create a new Sector :** Αυτή η επιλογή μας δίνει την δυνατότητα να δημιουργήσουμε μία νέα κατηγορία μετοχών στην εφαρμογή

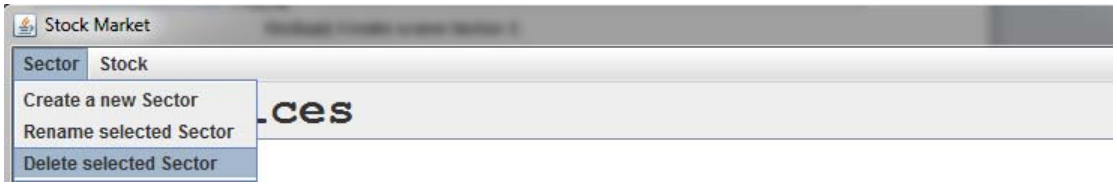
Εικόνα 26: Επιλογή Create a new Sector

2) **Sector -> Rename a selected Sector:** Αυτή η επιλογή μας επιτρέπει να μετονομάσουμε το sector που θα επιλέξουμε.



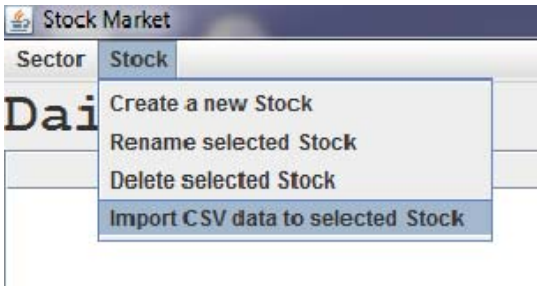
Εικόνα 27: Επιλογή Rename a selected Sector

3) **Sector -> Delete selected Sector:** Αυτή η επιλογή μας επιτρέπει να διαγράψουμε το sector που θα επιλέξουμε.



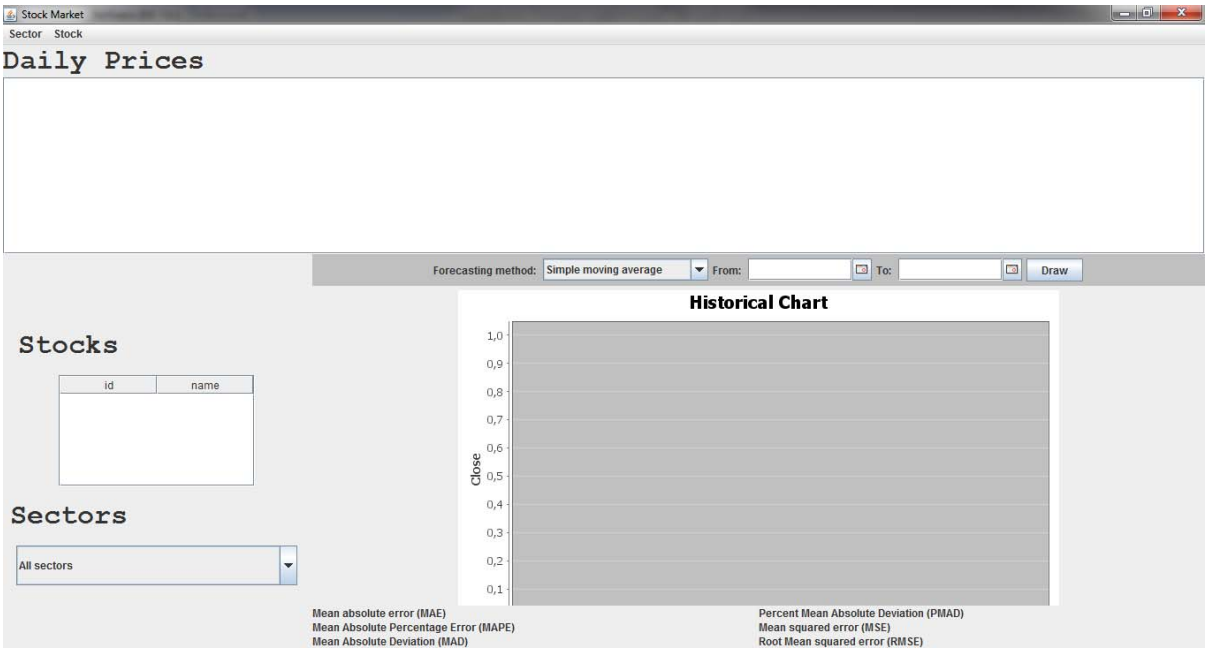
Εικόνα 2828: Επιλογή Delete selected Sector

Αντίστοιχα και για το μενού «**Stock**» υπάρχουν οι ίδιες επιλογές και μία επιπλέον η **Import CSV data to selected Stock** με την οποία μπορούμε να κάνουμε εισαγωγή ενός CSV αρχείου. Ένα CSV αρχείο περιέχει τα ιστορικά στοιχεία της μετοχής.



Περιβάλλον Συστήματος Πρόβλεψης Τιμών Μετοχών

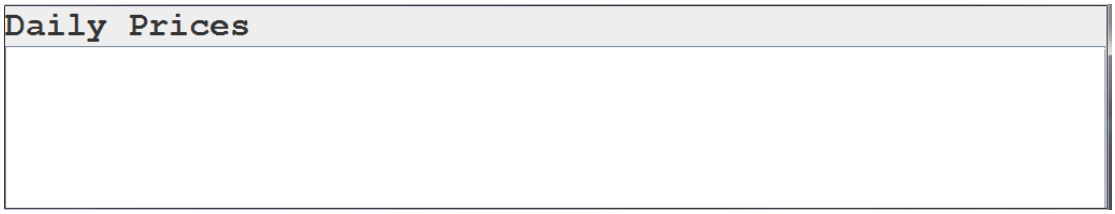
- **Stock Market:** Το όνομα του παραθύρου της εφαρμογής μας.



Εικόνα 29: Stock Market

Πιο ειδικά:

- **Daily Prices:** Εμφανίζει μία λίστα με τα ιστορικά στοιχεία κάθε μετοχής.



Εικόνα 30: Πίνακας DailyPrices

Date	Open	High	Low	Close	Volume
13-Jan-12	0.0	0.0	0.0	1.74	0
17-Jan-12	1.78	1.79	1.76	1.78	10149958
18-Jan-12	1.78	1.88	1.78	1.88	17120793
19-Jan-12	1.99	2.02	1.96	1.99	42796407
20-Jan-12	1.96	2.01	1.96	2.0	23342229
23-Jan-12	2.02	2.05	1.99	2.01	23281475
24-Jan-12	1.92	1.95	1.89	1.94	17509794
25-Jan-12	1.79	1.82	1.77	1.81	22129670
26-Jan-12	1.85	1.87	1.81	1.84	11423768
27-Jan-12	1.8	1.84	1.8	1.83	5527169
30-Jan-12	1.79	1.8	1.78	1.78	6924036

Εικόνα 31: DailyPrices

<u>Πιο συγκεκριμένα,</u>

Date: Η ημερομηνία.
Open: Η τιμή που πήρε η μετοχή την στιγμή που άνοιξε.
High: Η μέγιστη τιμή της μετοχής.
Low: Η ελάχιστη τιμή της μετοχής.
Close: Η τελευταία τιμή που πήρε η μετοχή για την συγκεκριμένη ημερομηνία.
Volume: Το πλήθος μετοχών.

- **Stocks:** Εμφανίζει μια λίστα με όλες τις μετοχές που είναι καταχωρημένες στην εφαρμογή μας ανάλογα την κατηγορία που θα επιλέξουμε.

Εικόνα 32: Stocks

- **Sectors:** Οι κατηγορίες των μετοχών, παρακάτω επιλέγουμε την κατηγορία που μας ενδιαφέρει και μας εμφανίζει στη περιοχή Stocks την λίστα με τις μετοχές που έχει αυτή η κατηγορία.

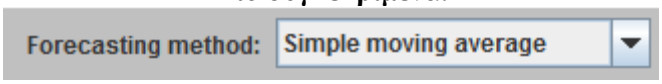
Εικόνα 33: Επιλογή Sector

- **Forecasting method:** Επιλογή Φίλτρου – Δείκτη πρόβλεψης τιμών της μετοχής που έχουμε επιλέξει από τη λίστα μετοχών stocks.

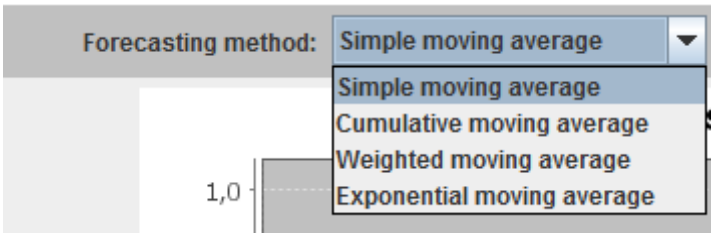
(*Forecasting method: Τεχνική ανάλυση τιμών μετοχών είναι ένας μαθηματικός υπολογισμός σχετικός με την τιμή της μετοχής που χρησιμοποιείται στην προσπάθεια να εκτιμηθούν μελλοντικές αλλαγές στην τιμή της μετοχής.)*

Εικόνα 34: Επιλογή "φίλτρου" και χρονικής περιόδου

Πιο συγκεκριμένα:



Εικόνα 35: Επιλογή Δείκτη Πρόβλεψης



Εικόνα 36: Επιλογή Δείκτη πρόβλεψης

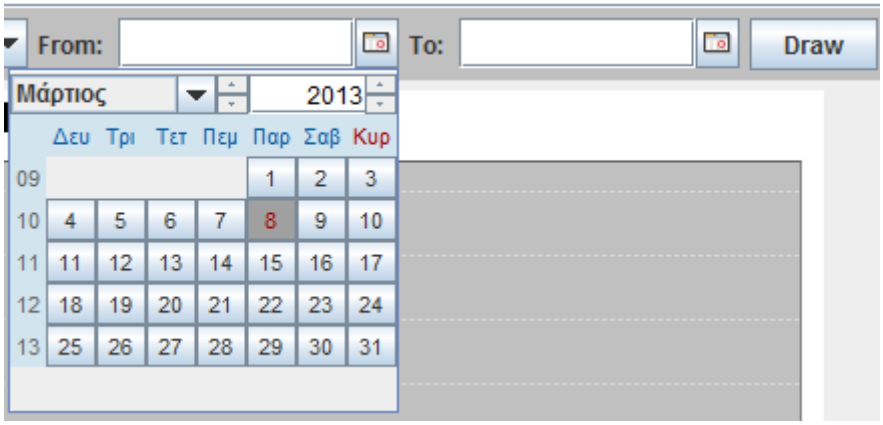
- **From: - To:** Επιλογή χρονικής περιόδου πρόβλεψης Τιμών Μετοχών

Εικόνα 37: Επιλογή Χρονικού Διαστήματος

Βασική παράμετρος του δείκτη που θα επιλέξουμε είναι η περίοδος υπολογισμού του. Και εδώ επιλέγουμε τη συγκεκριμένη χρονικής περίοδο που μας ενδιαφέρει.
Ο πιο συνηθισμένος Κινητός Μέσος Όρος είναι αυτός των 200 ημερών, μόνο όμως η επένδυση είναι για το πολύ μακρινό μέλλον. Αν γίνονται αγορές και πωλήσεις μετοχών καθημερινά, τότε ο Μέσος Όρος των 15 ή 25 ημερών είναι πιο χρήσιμος.

- **Κουμπί '' Draw'':**

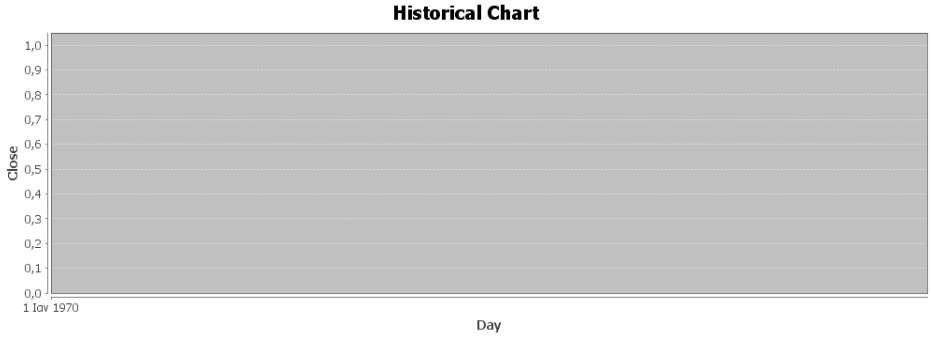
Το κουμπί Draw που υπάρχει στη μέση του παραθύρου «Stock Market» χρησιμοποιείται για να εμφανίζει τη γραφική παράσταση της μετοχής που έχουμε επιλέξει καθώς και την πρόβλεψη των τιμών της για την περίοδο που έχει (ζητηθεί) καταχωρηθεί.



Εικόνα 38: Draw Button

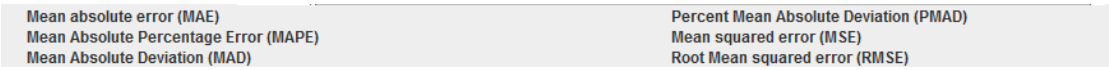
- **Historical Chart**

Στη συνέχεια του προγράμματος μας βλέπουμε την γραφική παράσταση (Historical Chart) των τιμών που πήρε η μετοχή μας για ένα χρονικό διάστημα και την μελλοντική πρόβλεψη.



Εικόνα 39: Εμφάνιση Γραφικής Παράστασης

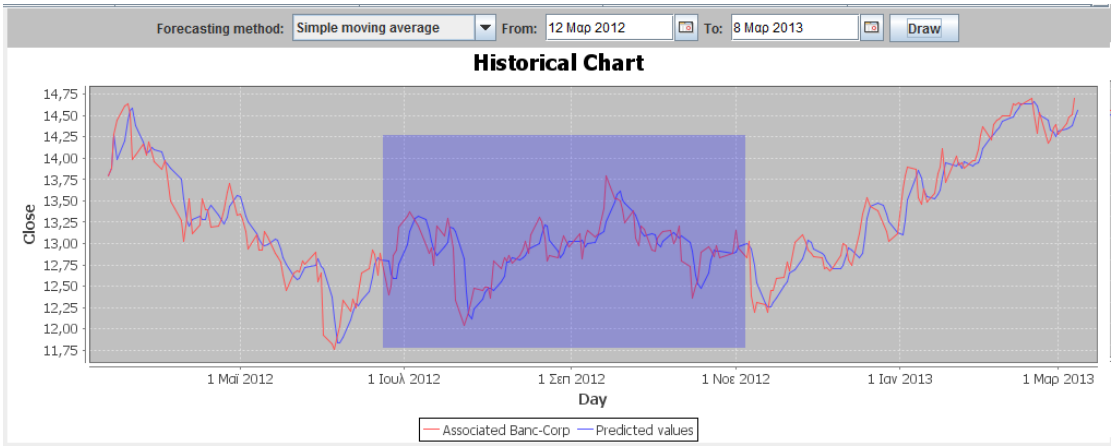
- **Ποσοστά αποτυχίας:**



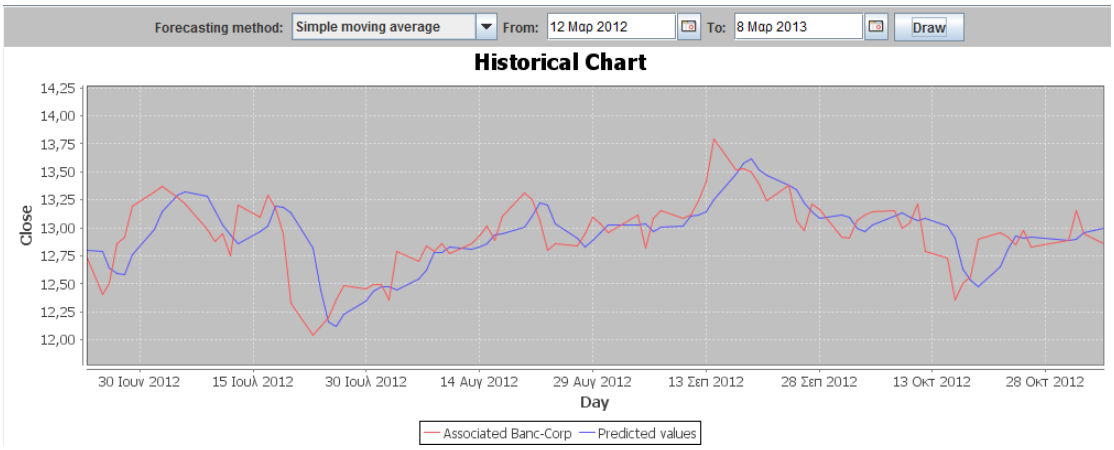
Εικόνα 40: Εμφάνιση ποσοστών πρόβλεψης

- **Historical Chart Zoom:**

Επίσης μπορούμε να μεγεθύνουμε μια περιοχή του γραφήματος μας για να δούμε την πρόβλεψη για την χρονική περίοδο που μας ενδιαφέρει.



Εικόνα 41: Zoom Γραφικής Παράστασης



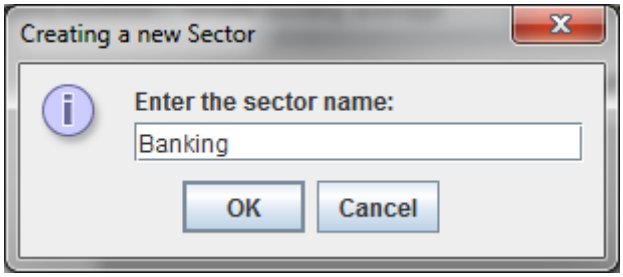
Εικόνα 42: Μεγεθυμένη Γραφική Παράσταση

Βήματα χρήσης εφαρμογής

Βήμα 1:

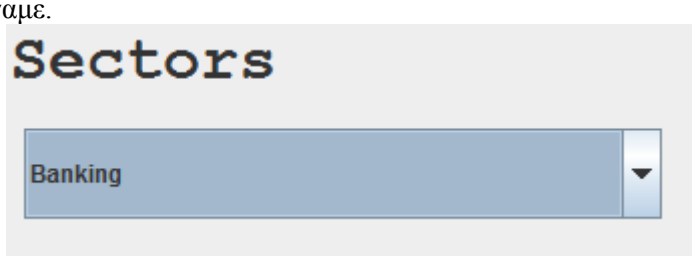
Από το μενού του προγράμματος επιλέγουμε **Sector-> Creating a new Sector**, πληκτρολογούμε στο πεδίο κειμένου το όνομα της κατηγορίας και πατάμε ‘’OK’’.

Μας εμφανίζει μήνυμα **‘’Done”** αν όλα πήγαν καλά.



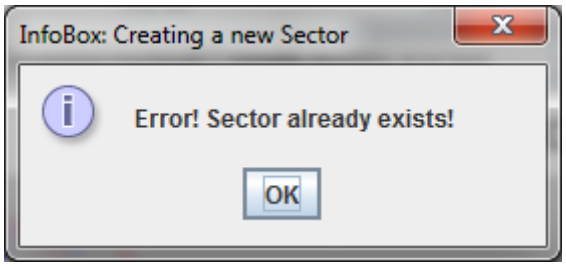
Εικόνα 43: Create Sector

Παρατηρούμε ότι εμφανίζεται στις κατηγορίες **‘’Sectors’’** της εφαρμογής το όνομα της κατηγορίας που εισάγαμε.



Εικόνα 44: Εμφάνιση κατηγορίας

Παρατήρηση: *Αν εισάγω όνομα Sector που υπάρχει ήδη εμφανίζεται το παρακάτω μήνυμα.*

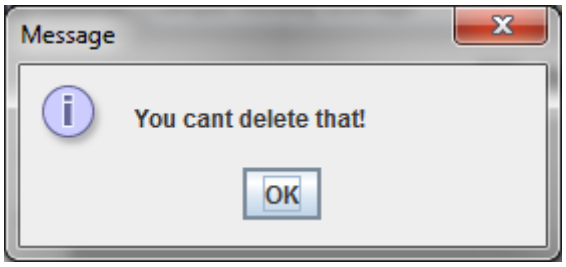


Εικόνα 45: Error message for Sector

ΠΑΡΑΤΗΡΗΣΕΙΣ:

Διαγραφή υπάρχουσας κατηγορίας γίνεται επιλέγοντας **Sector-> Delete selected Sector**

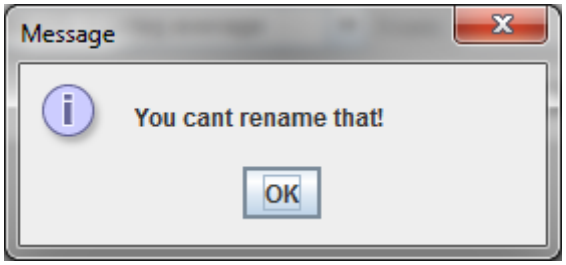
Παρατήρηση:*Πρέπει πρώτα να επιλέξουμε την κατηγορία που θέλουμε να διαγράψουμε γιατί αλλιώς μας εμφανίζεται το παρακάτω μήνυμα.*



Εικόνα 46: Error Message for delete Sector

Μετονομασία υπάρχουσας κατηγορίας γίνεται επιλέγοντας **Sector-> Rename selected Sector**

Παρατήρηση: *Πρέπει πρώτα να επιλέξουμε την κατηγορία που θέλουμε να μετονομάσουμε γιατί αλλιώς μας εμφανίζεται το παρακάτω μήνυμα.*



Εικόνα 47: Error Message for rename Sector

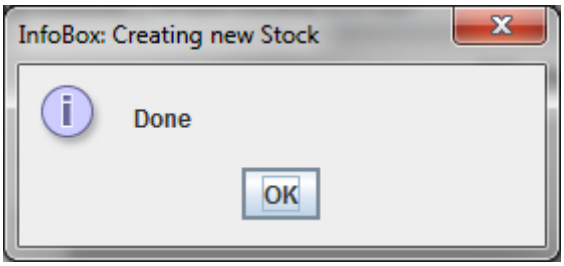
Βήμα 2:

Στη συνέχεια, αφού έχουμε επιλέξει την κατηγορία της μετοχής, που θέλουμε να εισάγουμε, από το μενού του προγράμματος επιλέγουμε **Stock-> Create a new Stock** . Πληκτρολογούμε στο πεδίο κειμένου που μας εμφανίζεται το όνομα της μετοχής που θέλουμε να εισάγουμε και πατάμε **''OK''**.

(Για να δημιουργήσουμε μια μετοχή πρέπει αρχικά να επιλέξουμε σε ποια κατηγορία θα ανήκει η συγκεκριμένη μετοχή.)

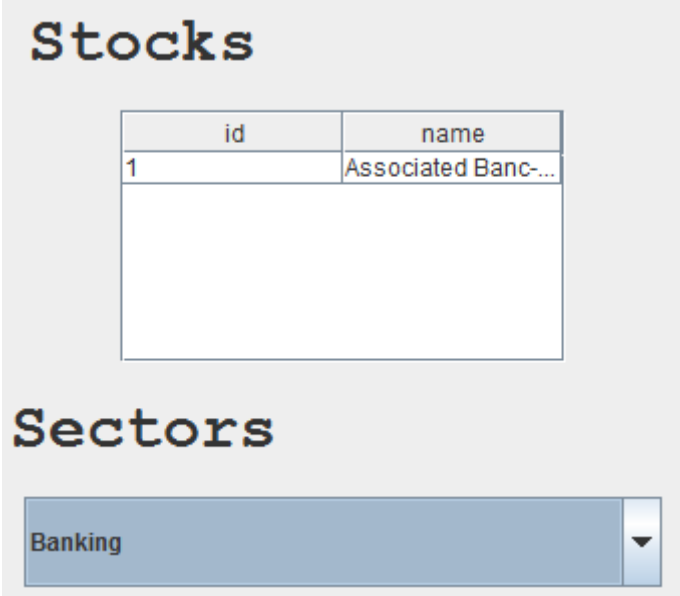
Εικόνα 48: Inserting a new Stock for Selected Sector

*Εμφανίζεται μήνυμα **"Done"** αν όλα πήγαν καλά.*



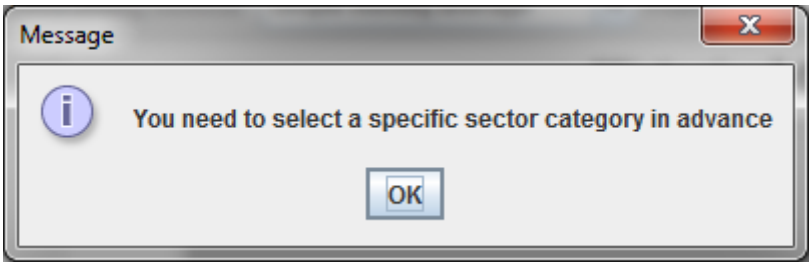
Εικόνα 4949: Message "Done"

Βλέπουμε στις μετοχές **''Stocks''** της εφαρμογής, το όνομα της μετοχής που εισάγαμε και το id που της δώθηκε στη συγκεκριμένη κατηγορία.



Εικόνα 5050: Stocks-Sectors

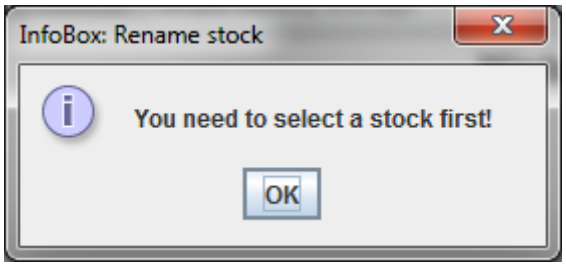
Παρατήρηση: *Αν δεν επιλέξουμε πρώτα την κατηγορία που ανήκει η μετοχή και κάνουμε την δημιουργία νέας μετοχής εμφαφνίζεται το παρακάτω μήνυμα λάθους.*



Εικόνα 5151: Message for Stock

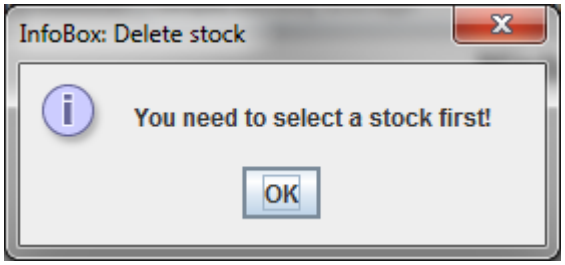
Η **Μετονομασία** μιας μετοχής γίνεται επιλέγοντας από το μενού της εφαρμογής **Stock-> Rename Selected Stock.**

Παρατήρηση: *Για να μετονομάσουμε μια μετοχή πρέπει αρχικά να επιλέξουμε τη συγκεκριμένη μετοχή. Αν δεν την επιλέξουμε εμφανίζεται το παρακάτω μήνυμα.*



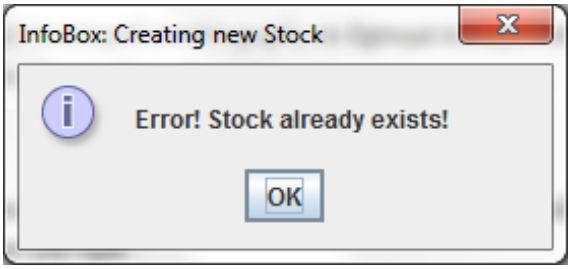
Εικόνα 5252: Message for Rename stock

Η **Διαγραφή** μιας μετοχής γίνεται επιλέγοντας από το μενού της εφαρμογής **Stock->Delete Selected Stock.**
Παρατήρηση: *Για να διαγράψουμε μια μετοχή πρέπει αρχικά να επιλέξουμε τη συγκεκριμένη μετοχή. Αν δεν την επιλέξουμε εμφανίζεται το παρακάτω μήνυμα.*



Εικόνα 5353: Message for Delete stock

Τέλος, **αν εισάγουμε ένα όνομα μετοχής που υπάρχει ήδη** στον πίνακα μετοχών μας , εμφανίζεται το ακόλουθο μήνυμα.

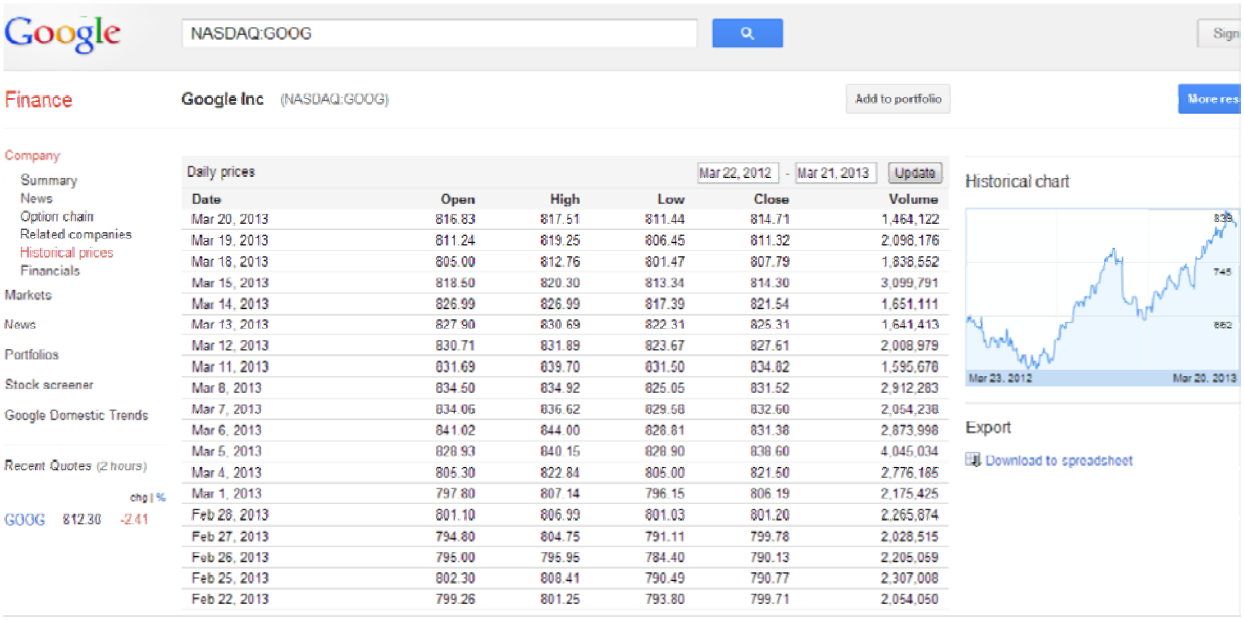


Εικόνα 54: Error Message for Stock

Βήμα 3:

Για να πάρουμε τα ιστορικά στοιχεια μιας μετοχής δηλαδή τις τιμές open, close, high, low και volume για μία χρονική περίοδο μπορούμε για διευκόλυνση μας να κατεβάσουμε αυτές τις τιμές από την ακόλουθη σελίδα:

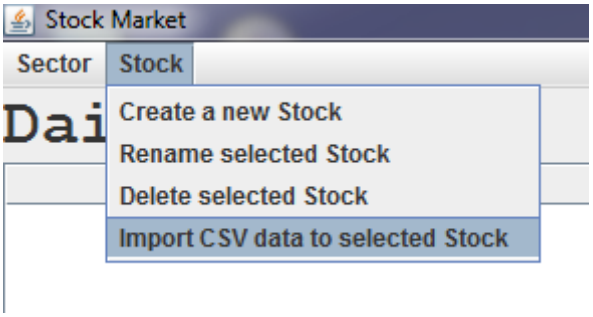
http://www.google.com/finance/historical?q=NASDAQ%3AGOOG&ei=9FpLUeDLBqmrwAPllQE



Date	Open	High	Low	Close	Volume
Mar 20, 2013	816.83	817.51	811.44	814.71	1,464,122
Mar 19, 2013	811.24	819.25	806.45	811.32	2,096,176
Mar 18, 2013	805.00	812.76	801.47	807.79	1,838,552
Mar 15, 2013	818.50	820.30	813.34	814.30	3,099,791
Mar 14, 2013	826.99	826.99	817.39	821.54	1,651,111
Mar 13, 2013	827.90	830.69	822.31	825.31	1,641,413
Mar 12, 2013	830.71	831.89	823.67	827.61	2,008,979
Mar 11, 2013	831.69	839.70	831.50	834.82	1,595,678
Mar 8, 2013	834.50	834.92	825.05	831.52	2,912,283
Mar 7, 2013	834.06	836.62	829.58	832.60	2,054,238
Mar 6, 2013	841.02	844.00	828.81	831.38	2,873,998
Mar 5, 2013	828.93	840.15	828.90	838.60	4,045,034
Mar 4, 2013	805.30	822.84	805.00	821.50	2,776,185
Mar 1, 2013	797.80	807.14	796.15	806.19	2,175,425
Feb 28, 2013	801.10	806.99	801.03	801.20	2,265,874
Feb 27, 2013	794.80	804.75	791.11	799.78	2,028,515
Feb 26, 2013	795.00	795.95	784.40	790.13	2,205,059
Feb 25, 2013	802.30	808.41	790.49	790.77	2,307,008
Feb 22, 2013	799.26	801.25	793.80	799.71	2,054,050

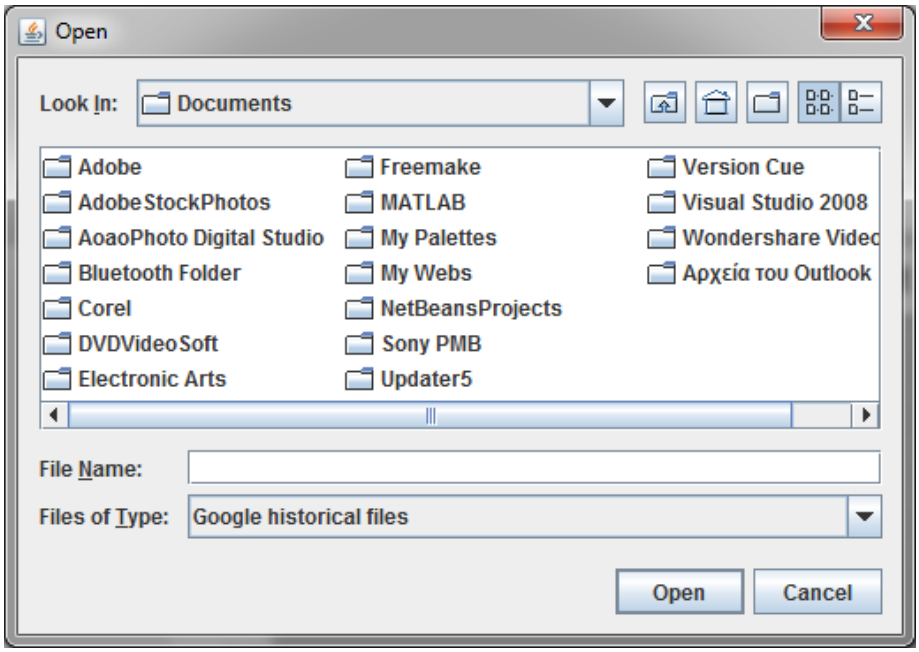
Εκεί πληκτρολογούμε το όνομα της μετοχής ή την εταιρία που μας ενδιαφέρει και πατώντας αναζήτηση, μας εμφανίζονται οι τιμές που έχει πάρει η μετοχή για όποια χρονική περίοδο εμείς επιλέξουμε. Επίσης μας εμφανίζει και την γραφική παράσταση της μετοχής με τις τιμές που είχε πάρει για το διάστημα ενός χρόνου. Τέλος, για να πάρουμε αυτές τις τιμές μετοχών και να τις χρησιμοποιήσουμε στη βάση μας πατάμε **Download to spreadsheet** και κατεβαίνει στον υπολογιστή μας το CSV αρχείο με τα ιστορικά δεδομένα της μετοχής που θα μας βοηθήσουν μέσω μαθηματικών υπολογισμών να βρούμε πιθανές επόμενες τιμές που μπορεί να πάρει η μετοχή.

Έπειτα για να εισάγω το αρχείο .csv που έχω κατεβάσει και περιέχει τα ιστορικά στοιχεία της συγκεκριμένης μετοχής, επιλέγουμε από το μενού το προγράμματος **Stock-> Import CSV data to selected Stock**



Εικόνα 55: Επιλογή Import CSV data to selected Stock

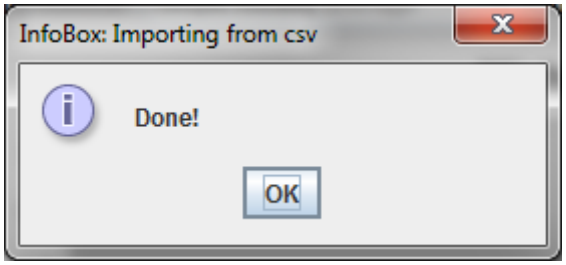
Παρακάτω εμφανίζεται το παράθυρο διαλόγου:



Εικόνα 56: Παράθυρο διαλόγου επιλογής CSV αρχείου

Από εκεί επιλέγουμε τον φάκελο που έχουμε αποθηκεύσει την μετοχή δηλαδή το **.csv** αρχείο και πατάμε το κουμπί **open**. Αν θέλουμε να βγούμε από το παράθυρο διαλόγου πατάμε το κουμπί cancel .

Αν όλα πήγαν καλά εμφανίζεται μήνυμα "Done!".



Εικόνα 57: Message Done for CSV files

Βλέπουμε στις ημερήσιες τιμές **''Daily Prices''** της εφαρμογής τα ιστορικά στοιχεία της μετοχής (τις τιμές στις οποίες κυμάνθηκε για ένα χρόνο) που εισάγαμε.

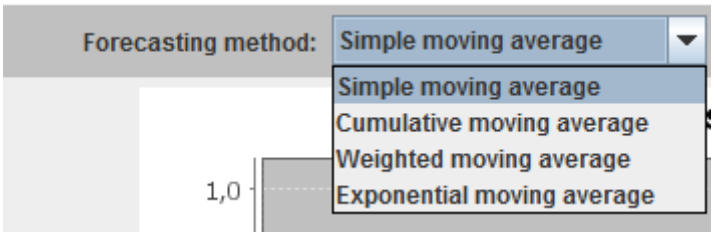
Εικόνα 58: Εμφάνιση DailyPrices

Βήμα 4:

Στο επόμενο Βήμα επιλέγουμε το **''φίλτρο-δείκτη''** πρόβλεψης τιμών της μετοχής, που έχουμε επιλέξει από τη λίστα μετοχών **''Stocks''**.

Εικόνα 59: Επιλογή Συνάρτησης-Φίλτρου

Πιο συγκεκριμένα:



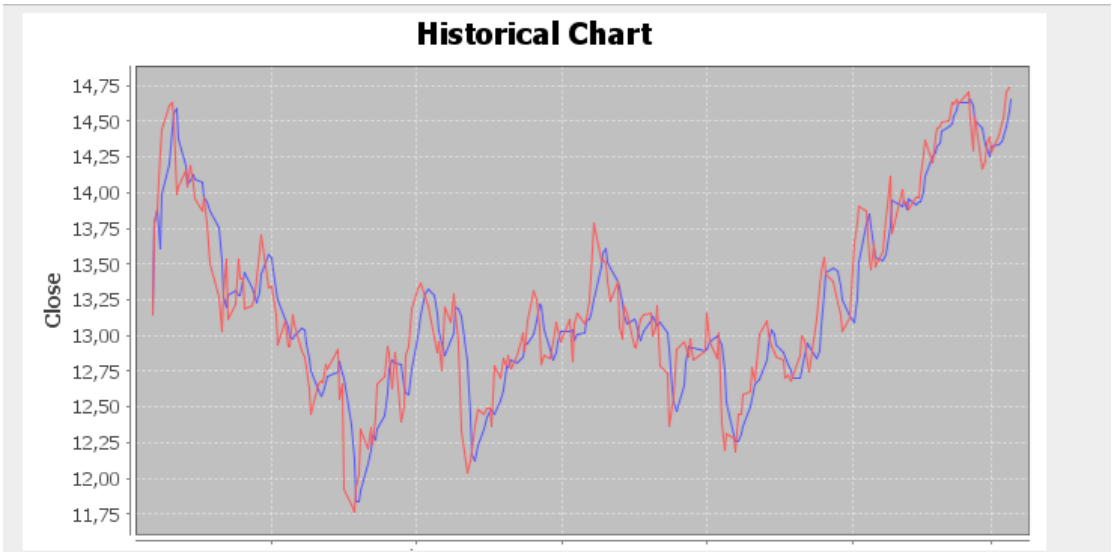
Εικόνα 60: Επιλογή Συγκεκριμένου Φίλτρου

Βήμα 5:

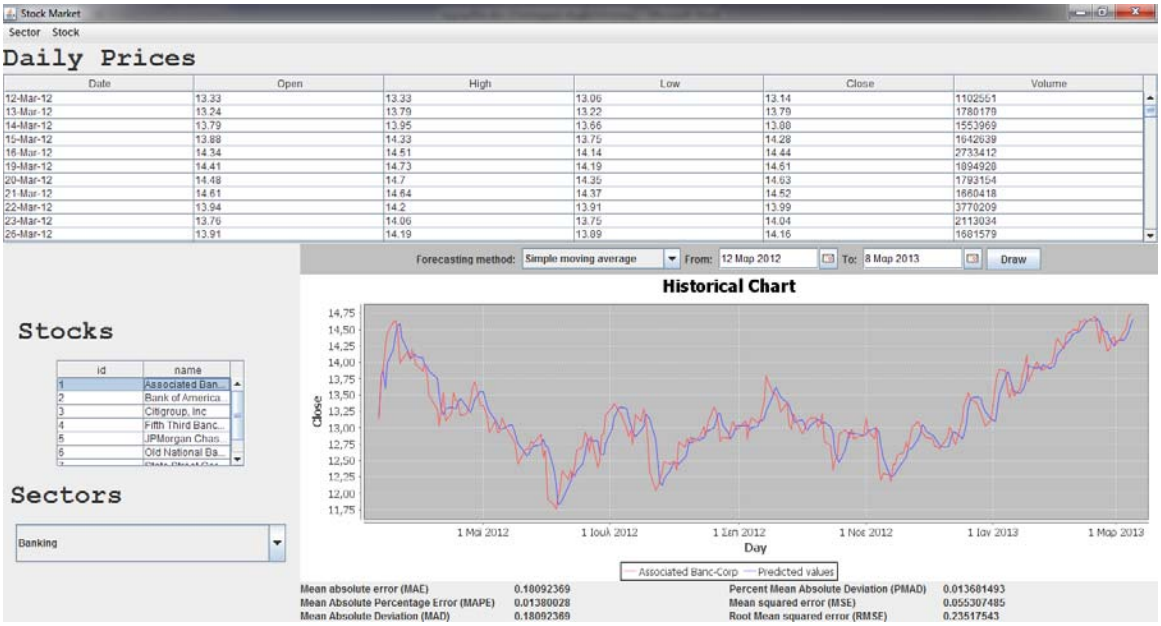
Τέλος, επιλέγουμε την περίοδο που μας ενδιαφέρει για την πρόβλεψη των τιμών της μετοχής και πατάμε το κουμπί Draw για να ζωγραφιστεί η γραφική μας παράσταση.

Εικόνα 61: Επιλογή Χρονικού Διαστήματος

Το κουμπί **Draw** που υπάρχει στη μέση του παραθύρου «Stock Market» χρησιμοποιείται για να εμφανίζει τη γραφική παράσταση των ιστορικών τιμών της μετοχής που έχουμε επιλέξει καθώς και την πρόβλεψη των τιμών της για την περίοδο που έχει (ζητηθεί) καταχωρηθεί.



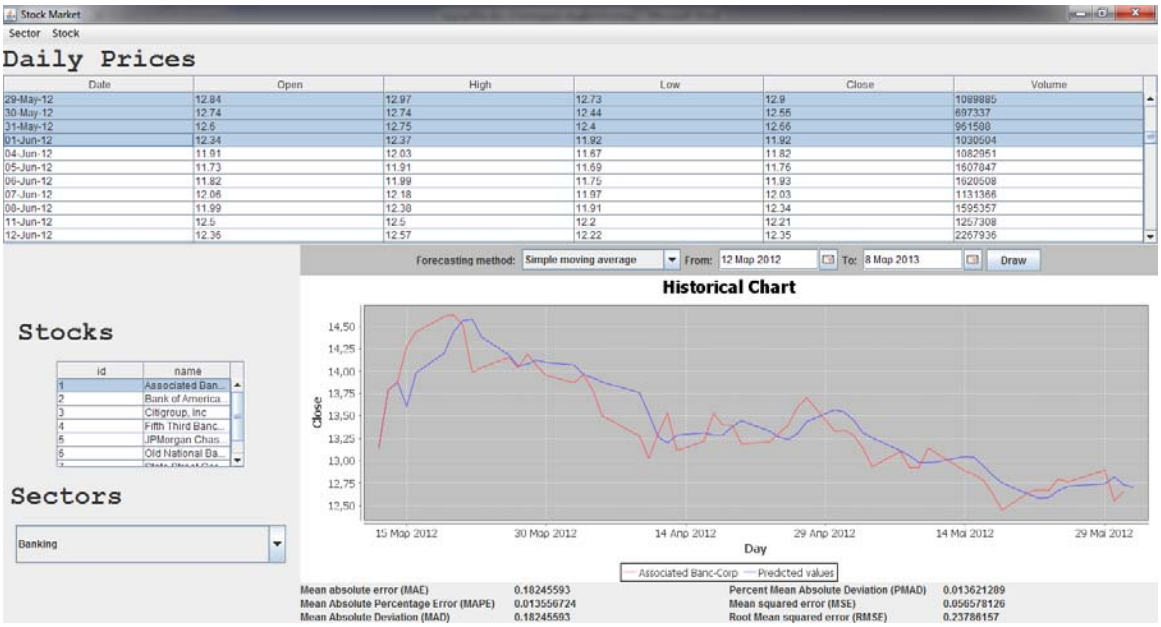
Εικόνα 62: Εμφάνιση πρόβλεψης τιμών μετοχής



Εικόνα 6363: Simple moving average πρόβλεψη

Σημείωση: Αν θέλουμε μπορούμε να επιλέξουμε από τα Daily Prices συγκεκριμένες ημερήσιες τιμές της μετοχής για να γίνει η πρόβλεψη.

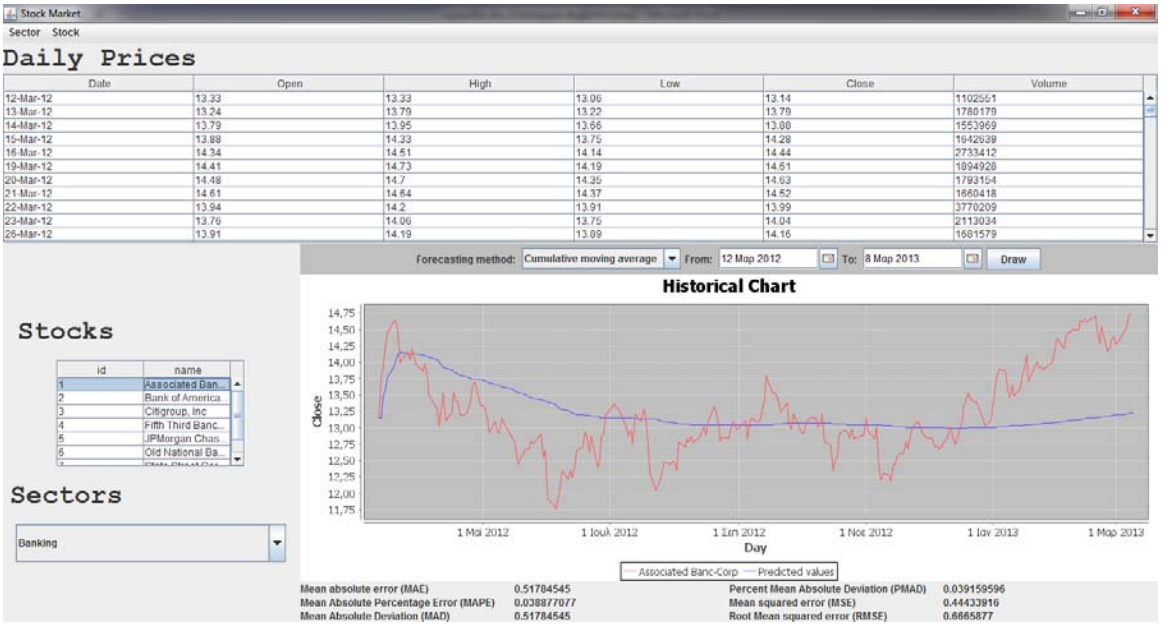
Παράδειγμα: Επιλογή περιόδου 12 Μαρτίου 2012 – 1Ιουνίου 2012



Εικόνα 64: Επιλογή Συγκεκριμένου διαστήματος από τον πίνακα των DailyPrices

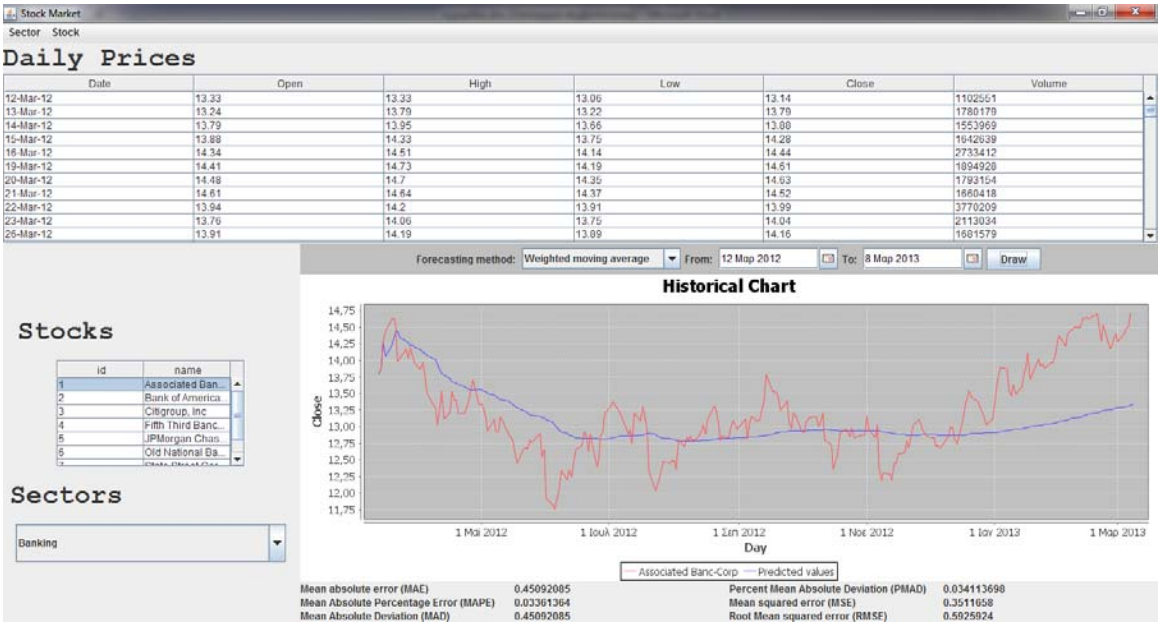
Σημείωση: *Η κόκκινη γραμμή στην γραφική παράσταση της εφαρμογή μας αντιπροσωπεύει τις τιμές που έκλεισε η μετοχή, ενώ η μπλε τις τιμές που βρήκαμε μέσω μαθηματικών υπολογισμών με την βοήθεια των Daily prices και αντιπροσωπεύουν τις πιθανές τιμές που μπορεί να πάρει η μετοχή.*

- **Cumulative moving average - Αθροιστικός κινητός μέσος όρος:**



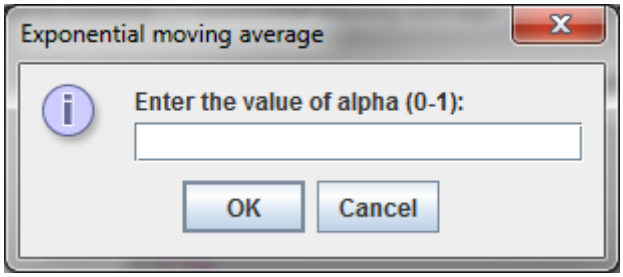
Εικόνα 65: Cumulative moving average πρόβλεψη

- **Weighted moving average - Σταθμισμένος μέσος όρος:**

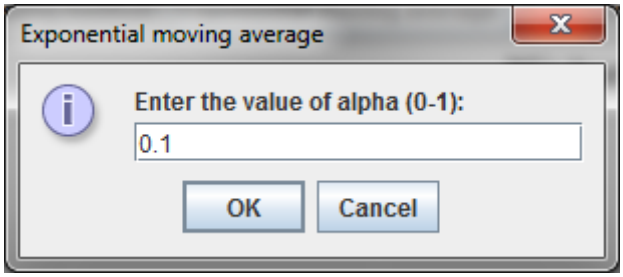


Εικόνα 66: Weighted moving average πρόβλεψη

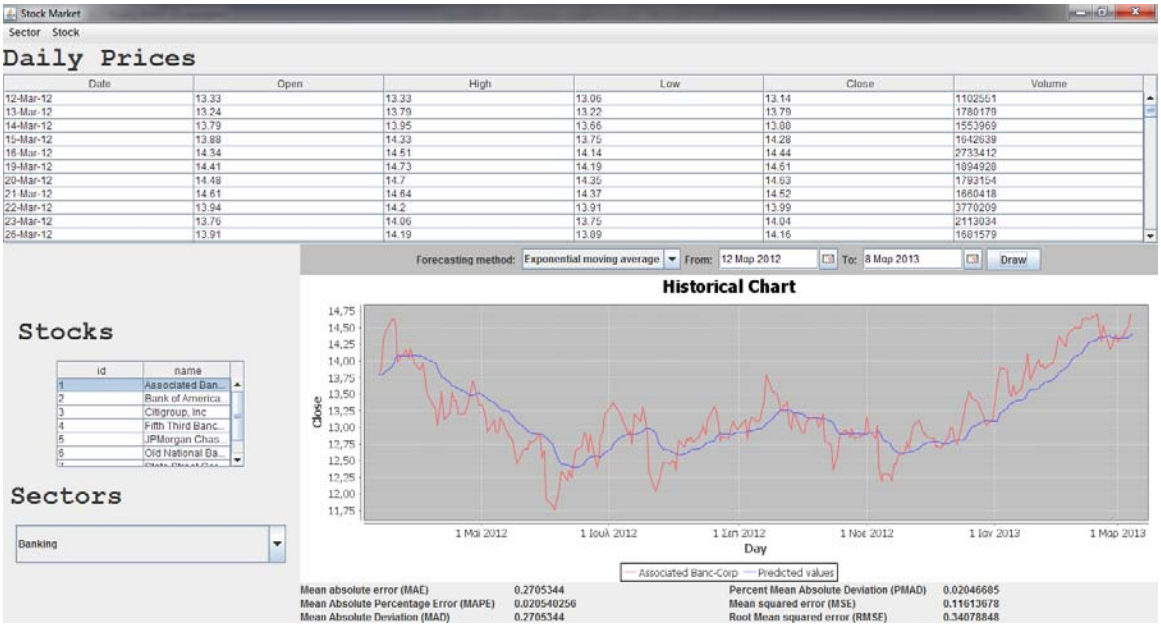
- **Exponential moving average – Εκθετικός μέσος όρος**



Εικόνα 67: Exponential moving average value



Σημείωση: Όσο μικρότερη η τιμή του alpha τόσο μεγαλύτερη εμπιστοσύνη δείχνουμε στην αρχική μας πρόβλεψη.
Συνήθεις τιμές του alpha: 0.1 – 0.3



Εικόνα ~~68~~68: Exponential moving average πρόβλεψη

5.3 Αξιολόγηση της ακρίβειας των προβλέψεων των μετοχών

Πρόβλεψη τιμών μετοχής Toyota Motor που ανήκει στην κατηγορία Transportation με την μέθοδο **Simple Moving Average** για την περίοδο ενός έτους (12 Μαρτίου 2012 – 12 Μαρτίου 2013)

Ιστορικά Δεδομένα

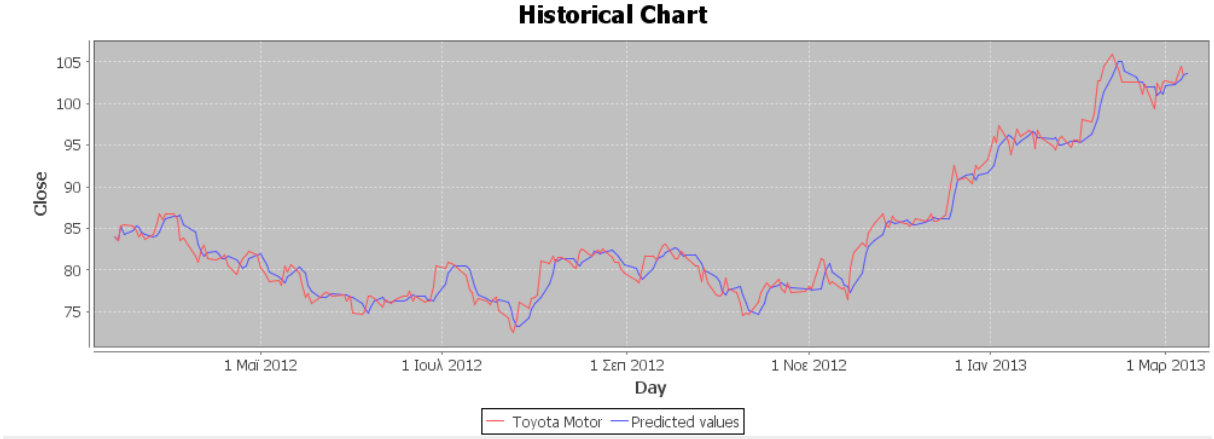
Sector Stock

Daily Prices

Date	Open	High	Low	Close	Volume
12-Mar-12	83.18	83.18	82.61	82.78	271336
13-Mar-12	83.1	83.99	83.09	83.97	307973
14-Mar-12	84.06	84.2	83.31	83.54	344688
15-Mar-12	85.1	85.43	84.79	85.34	591235
16-Mar-12	85.77	85.85	85.34	85.4	368183
19-Mar-12	85.2	85.28	84.79	85.24	316026
20-Mar-12	84.43	84.86	84.1	84.66	303727
21-Mar-12	84.07	84.23	83.79	83.96	283480
22-Mar-12	83.97	84.72	83.96	84.43	516879
23-Mar-12	83.36	83.76	82.97	83.76	437466
26-Mar-12	84.19	84.59	83.8	84.35	559696

Forecasting method: Simple moving average From: 12 Μαρ 2012 To: 8 Μαρ 2013 Draw

Γραφική Παράσταση



Mean absolute error (MAE)	1.1013085	Percent Mean Absolute Deviation (PMAD)	0.013116623
Mean Absolute Percentage Error (MAPE)	0.01314576	Mean squared error (MSE)	2.091684
Mean Absolute Deviation (MAD)	1.1013085	Root Mean squared error (RMSE)	1.4462656

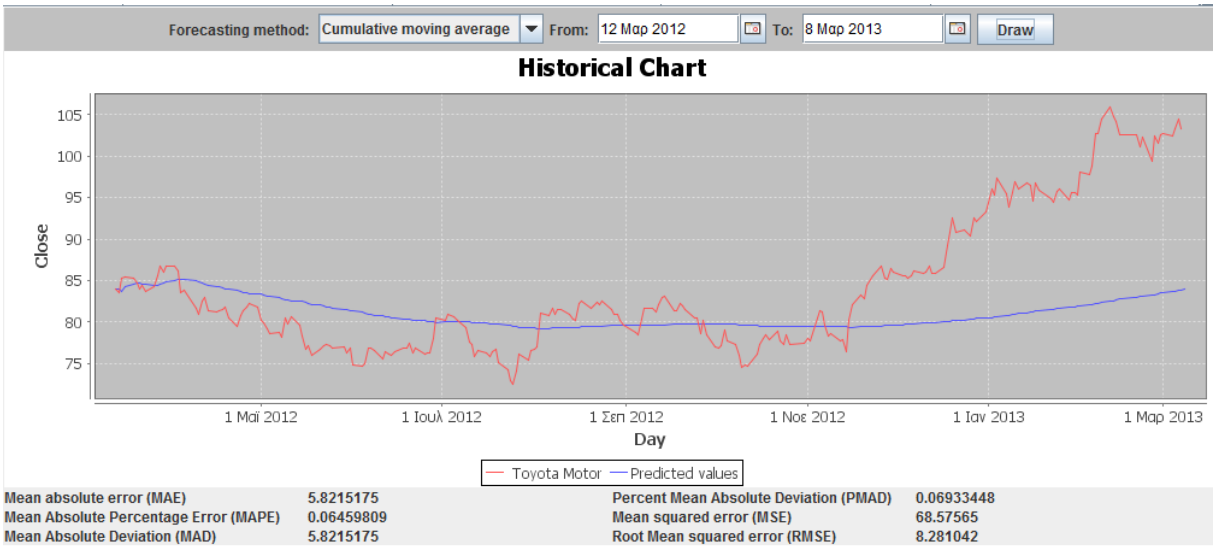
Εξομάλυνση και Πρόβλεψη των τιμών μετοχών

Για την αξιολόγηση του υποδείγματος θα χρησιμοποιήσουμε τα κριτήρια αξιολόγησης των μεθόδων προβλέψεων. Θα βρούμε τις αποκλίσεις των προβλεπόμενων τιμών από τις πραγματικές τιμές και θα υπολογίσουμε τις τιμές των κριτηρίων MAD, MAPE, MAD, PMAD, MSE και RMSE.

Οι τιμές των κριτηρίων είναι αρκετά καλές και δείχνουν ότι οι προβλεπόμενες τιμές με τη μέθοδο Simple moving average πλησιάζουν αρκετά τις πραγματικές τιμές της χρονοσειράς.

MAE: 1.1
MAPE: 0.013
MAD: 1.1
PMAD: 0.013
MSE: 2.09
RMSE: 1.446

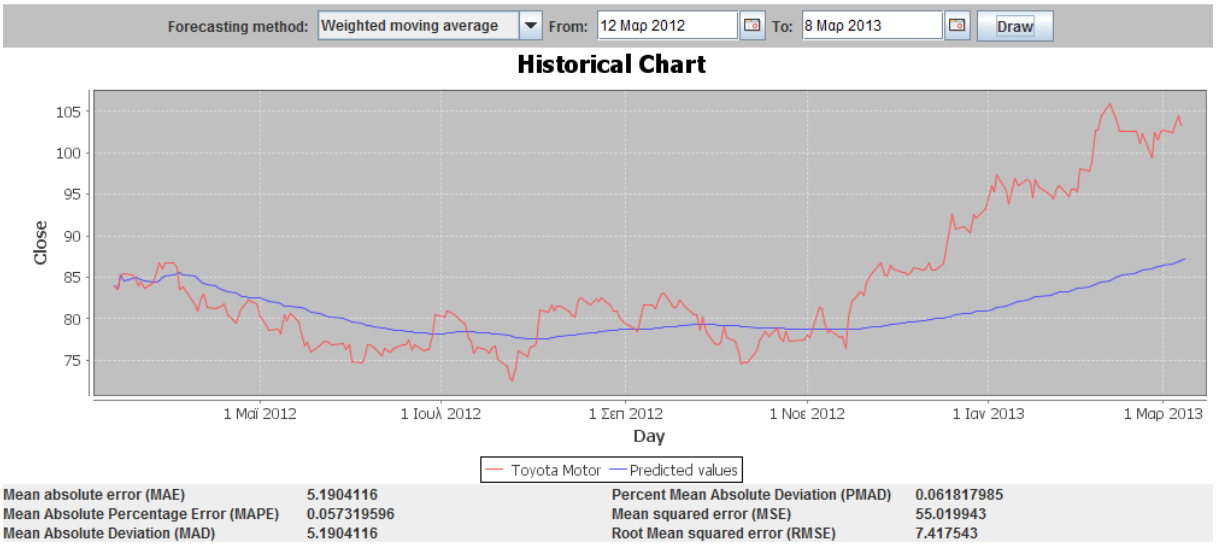
Πρόβλεψη τιμών μετοχής Toyota Motor που ανήκει στην κατηγορία Transportation με την μέθοδο **Cumulative Moving Average** για την περίοδο ενός έτους (12 Μαρτίου 2012 – 12 Μαρτίου 2013)



MAE: 5.82
MAPE: 0.064
MAD: 5.82
PMAD: 0.069
MSE: 68.575
RMSE: 8.28

Παρατηρούμε ότι η μέθοδος Cumulative moving average προβλέπει με όχι και τόσο μεγάλη ακρίβεια τις μελλοντικές τιμές της χρονοσειράς. Παρατηρούμε ότι οι πραγματικές τιμές αρχίζουν να αποκλίνουν σε μεγάλο βαθμό από τις προβλεφθείσες από τον Νοέμβριο του 2012.

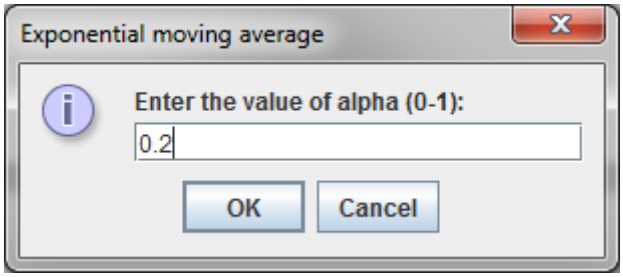
Πρόβλεψη τιμών μετοχής Toyota Motor που ανήκει στην κατηγορία Transportation με την μέθοδο **Weighted Moving Average** για την περίοδο ενός έτους (12 Μαρτίου 2012 – 12 Μαρτίου 2013)

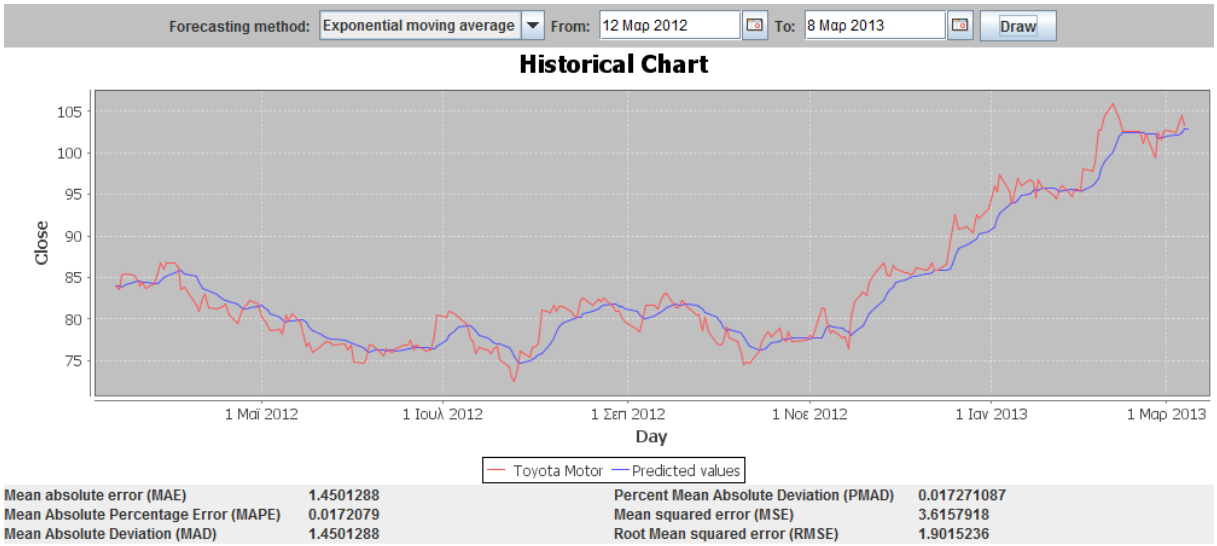


MAE: 5.19
MAPE: 0.057
MAD: 5.19

PMAD: 0.0618
MSE: 55.0199
RMSE: 7.4175

Πρόβλεψη τιμών μετοχής Toyota Motor που ανήκει στην κατηγορία Transportation με την μέθοδο **Exponential Moving Average** για την περίοδο ενός έτους (12 Μαρτίου 2012 – 12 Μαρτίου 2013)





MAE: 1.45
MAPE: 0.017
MAD: 1.45
PMAD: 0.017
MSE: 3.61579
RMSE: 1.90

Για να έχουμε μια πληρέστερη εικόνα του σφάλματος σε βάθος χρόνου χρησιμοποιούμε τα μεγέθη της Μέσης Απόλυτης Απόκλισης (mean absolute deviation, MAD) ή/και του Μέσου Τετραγωνισμένου Σφάλματος (mean squared error, MSE). Όσο μικρότερη είναι η τιμή των μεγεθών αυτόν, τόσο μεγαλύτερη η ακρίβεια. **Με τα μεγέθη αυτά μπορούμε να ελέγξουμε την ακρίβεια των μεθόδων πρόβλεψης και να επιλέξουμε τη βέλτιστη.**

Είναι προφανές ότι η ακρίβεια της μεθόδου Simple moving average είναι καλύτερη τόσο με τη μορφή του MAD όσο και του MSE αφού έχουν και τα δύο μικρότερη τιμή από τα αντίστοιχα των μεθόδων Cumulative, Weighted και Exponential moving average.
Δεύτερη καλύτερη βάση των τιμών είναι η Exponential moving average.

Σύγκριση της Moving Average και Exponential Smoothing

Ομοιότητες:

- Και οι δύο μέθοδοι κατάλληλες για σταθερές σειρές
- Και οι δύο μέθοδοι εξαρτώνται από μία μόνο παράμετρο
- Και οι δύο μέθοδοι δεν συμβαδίζουν με μια τάση
- Κάποιος μπορεί να επιτύχει την ίδια κατανομή του σφάλματος πρόβλεψης με τον καθορισμό:

a = 2/ (N + 1) or N = (2 - a)/ a

Διαφορές:

- ES μεταφέρει όλα τα τελευταία ιστορικά δεδομένα
- MA αποβάλλει "κακά" δεδομένα μετά από N περιόδους
- MA απαιτεί από όλους τα N σημεία δεδομένων του παρελθόντος για τον υπολογισμό νέα εκτίμηση πρόβλεψης, ενώ απαιτεί μόνο ES την τελευταία πρόβλεψη και τελευταία παρατήρηση της «ζήτησης» για να συνεχίσει.

Ακόμα κι αν υπάρχουν σαφείς διαφορές μεταξύ των απλού κινητού μέσου όρου και του εκθετικού κινητού μέσου όρου, ο ένας δεν είναι απαραιτήτως καλύτερος από τον άλλο. Ο εκθετικός κινητός μέσος όρους έχει λιγότερη καθυστέρηση και ως εκ τούτου είναι πιο ευαίσθητος στις πρόσφατες τιμές και στις πρόσφατες μεταβολές των τιμών. Επίσης ο εκθετικός κινητός μέσος όρος, θα γίνει πριν από ένα απλό κινούμενο μέσο όρο.

Ο απλός μέσος όρος από την άλλη πλευρά, αντιπροσωπεύει έναν αληθινό μέσο όρο τιμών για ολόκληρο το χρονικό διάστημα. Ως εκ τούτου, ο απλός κινητός μέσος όρος μπορεί να είναι καταλληλότερος για τον εντοπισμό στήριξης. Αξίζει να αναφέρουμε πως ο απλός μέσος όρος, εξαρτάται από τους στόχους, το αναλυτικό ύφος και το χρονικό ορίζοντα.

Χαρακτηριστικά, θα πρέπει να πειραματιστούμε και με τα δύο είδη κινητού μέσου όρου καθώς και τα διάφορα χρονικά πλαίσια για να βρούμε το καλύτερο . Επίσης, μία από τις πιο χρήσιμες τεχνικές είναι ο σταθμισμένος κινητός μέσος όρος. Ένας κινητός μέσος όρος παίρνει μια σειρά από προηγούμενες τιμές κλεισίματος, τις προσθέτει, και διαιρεί με τον αριθμό των ημερών του συγκεκριμένου χρονικού διαστήματος. Το αποτέλεσμα είναι ένας δείκτης που δείχνει τη γενική κατεύθυνση των τιμών, χωρίς την ακραία αστάθεια που μπορεί να στρεβλώσει την ανάγνωσή μας.

Μερικοί επενδυτές προτιμούν απλούς κινητούς μέσους όρους για μεγάλες χρονικές περιόδους για τον εντοπισμό μακροπρόθεσμης αλλαγής τάσης.

Συμβουλές: Το κόλπο είναι να επιλέξετε τα κατάλληλα χρονικά διαστήματα. Είναι σοφό να ελέγξετε το σύστημα σας αν έχει επιλέξει εκτενώς τα ιστορικά δεδομένα για την ασφάλεια των συναλλαγών που σκοπεύετε να κάνετε.

6. Συμπεράσματα και Μελλοντική Εργασία

Το βασικότερο συμπέρασμα που μπορεί να εξαχθεί είναι πως η Java και η Mysql συνεργάζονται άψογα μεταξύ τους για την δημιουργία σύνθετων αλλά και πολύ χρήσιμων συστημάτων. Μάλιστα στις μέρες μας είναι από τα πλέον χρησιμοποιούμενα εργαλεία για την ανάπτυξη εφαρμογών με υψηλά κριτήρια αξιοπιστίας και απόδοσης. Η ανάπτυξη μιας πτυχιακής βοηθά τον σπουδαστή να εμπεδώσει θεωρητικά και τεχνικά θέματα τα οποία διδάχθηκε κατά την διάρκεια των σπουδών του, δημιουργώντας τις προϋποθέσεις να τα χρησιμοποιήσει στην μετέπειτα σταδιοδρομία του.

Ο σχεδιασμός και η υλοποίηση της εφαρμογής εστιάζει στην υποστήριξη των επενδυτών που ασχολούνται με την χρηματιστηριακή αγορά. Προσφέρει στους επενδυτές τη δημιουργία ενός περιβάλλοντος με κατηγοριοποιημένες μετοχές, δίνει την δυνατότητα να διαγραφούν, να μετονομαστούν κατηγορίες μετοχών αλλά και να εισαχθούν νέες. Επίσης τις ίδιες λειτουργίες ισχύουν και για τις μετοχές. Μπορούν με εύκολο τρόπο να εισάγουν στην εφαρμογή ιστορικά δεδομένα μετοχών από έγκυρη και συνεχώς ενημερωμένη πηγή και να επιλέξουν τέσσερις διαφορετικούς τρόπους ώστε να υπολογιστούν πιθανές επόμενες τιμές των μετοχών για ένα χρονικό διάστημα που θα επιλέξει, δείχνοντας βέβαια και τις αποκλίσεις των τιμών που υπάρχουν (σφάλματα πρόβλεψης). Έτσι αποκτούν μία γενικότερη εικόνα για τις μελλοντικές τιμές της μετοχής αλλά και για την γενική πρόοδο της μετοχής από την στιγμή που άνοιξε μέσω της γραφικής παράστασης των τιμών της μετοχής που εμφανίζεται στην εφαρμογή μας.

Οι επενδυτές με την βοήθεια της εφαρμογής μας και μέσω των τεχνικών δεικτών μπορούν να δουν το ξεκίνημα μιας νέας τάσης των τιμών (ανοδική ή καθοδική), την ταχύτητα και την ορμή αυτής της τάσης, πόσο «ώριμη» είναι η τάση και πότε έχουμε τερματισμό της τάσης, εάν η τάση πρόκειται να αναστραφεί. Γνωρίζοντας αυτούς τους τέσσερις βασικούς παράγοντες της τάσης, μπορεί να καθοριστεί επακριβώς πότε θα αγοραστεί και θα πουληθεί τη μετοχή ώστε να τηρείται πιστά ο κανόνας «αγόραζε φθηνά, πούλα ακριβά».

Μελλοντική Εργασία

Η εφαρμογή που αναπτύξαμε μπορεί μελλοντικά να εξελιχθεί και να συνδυάζει περισσότερες λειτουργίες. Την εφαρμογή αυτή θα μπορούσαμε να την αναπτύξουμε σε επόμενο στάδιο και σε ιστοσελίδα που ο χρήστης θα είχε την δυνατότητα να την επισκεφτεί και εύκολα να ενημερωθεί για τις τάσεις των μετοχών. Μερικές μελλοντικές προτάσεις για την εξέλιξη της εφαρμογής μας είναι ότι ο χρήστης θα μπορεί να εγγραφεί στην εφαρμογή και με τον τρόπο αυτό θα του επιστρέφει μόνο τις μετοχές που τον ενδιαφέρουν, αλλά και να τον ενημερώνει αυτόματα η εφαρμογή, μέσω ενός SMS μηνύματος για τις αλλαγές των τιμών της μετοχής. Επίσης θα μπορούσαν να βρεθούν καλύτερες μέθοδοι για τον υπολογισμό των πιθανών τιμών που μπορεί να πάρει η μετοχή, ώστε να έχουμε και μικρότερα ποσοστά αποτυχίας, έτσι θα μπορεί ο επενδυτής να εμπιστευτεί πλήρως την εφαρμογή μας.

Επίσης θα μπορούσαμε να προγραμματίσουμε την εφαρμογή μας έτσι ώστε η βάση να ενημερώνεται αυτόματα κάθε μέρα και να έχουμε τις ημερήσιες τιμές συνεχώς, χωρίς την παρέμβαση του χρήστη με σκοπό την εξυπηρέτηση του. Το τελευταίο επειδή δεν θα χρειάζεται η διαδικασία της εύρεσης, αποθήκευσης και εισαγωγής των CSV αρχείων. Τέλος, θα μπορούσαμε να εισάγουμε περισσότερες πληροφορίες για κάθε μετοχή ώστε ο κάθε χρήστης να έχει μια συνολική εικόνα για την πορεία της μετοχής αλλά και της εταιρίας. Τέλος μια ακόμη ιδέα θα μπορούσε να ήταν η καταγραφή των κινήσεων του χρήστη σαν επενδυτή έτσι ώστε να του προσφέρει το ιστορικό των κινήσεων του, ποιες και πόσες μετοχές έχει στην διάθεση του έτσι ώστε να του εμφανίζει πληροφορίες και στατιστικά προβλέψεων που θα αφορούν τις επενδύσεις του.

Βιβλιογραφία

1. [Wikipedia]: http://el.wikipedia.org/wiki/JDBC
2. [Wikipedia]: http://en.wikipedia.org/wiki/Moving_average
3. http://www.islab.demokritos.gr/gr/html/ptixiakes/kostas-aris_ptyxiakh/Phtml/java.htm
4. http://www.scribd.com
5. http://anamorfosi.teiser.gr/ekp_yliko/e-notes/Data/database/main.htm
6. http://osarena.net/hacks-guides/mathenontas-tin-java-mathima-1o.html
7. http://dide.flo.sch.gr/Plinet/Tutorials/Tutorials-DataBasesTheory.html
8. http://www.mykosmos.gr/loc_mk/xrimatistirio.asp#metoxi
9. http://viralpatel.net/blogs/java-read-write-csv-file/
10. http://www.google.com/finance/historical?q=GOOG
11. http://java.sun.com/docs/books/tutorial/getStarted/intro/definition.html
12. http://netbeans.org/index_el.html
13. http://users.uom.gr/~kat/poe/notes/ch01_kat.pdf
14. http://openarchives.gr/search/%CE%A0%CF%81%CF%8C%CE%B2%CE%BB%CE%B5%CF%88%CE%B7%20%CF%84%CE%B9%CE%BC%CF%8E%CE%BD%20%CE%BC%CE%B5%CF%84%CE%BF%CF%87%CF%8E%CE%BD
15. [Wikipedia]http://el.wikipedia.org/wiki/XAMPP
16. http://www.informit.com/articles/article.aspx?p=332278&seqNum=2
17. http://www.markets.com/el/education/technical-analysis/math-indicators.html
18. http://www.in.gr/stocks/tahelp/kmo.htm
19. http://www.investorguide.com/sector
20. http://stockcharts.com/school/doku.php?id=chart_school%3Atechnical_indicators%3Amoving_averages
21. http://stockcharts.com/school/doku.php?id=chart_school:technical_indicators:moving_averages
22. Μιχάλης Βαϊδάνης, Αρχές Διοίκησης και Οργάνωση Παραγωγής: Σημειώσεις «Πρόβλεψης»
23. ΓΕΩΡΓΙΟΣ Σ. ΠΑΝΑΓΙΩΤΟΥ, ΠΡΟΒΛΕΨΕΙΣ ΠΩΛΗΣΕΩΝ ΤΩΝ Ι.Χ. ΑΥΤΟΚΙΝΗΤΩΝ ΣΕ ΔΕΚΑΠΕΝΤΕ ΧΩΡΕΣ – ΜΕΛΗ ΤΗΣ ΕΥΡΩΠΑΪΚΗΣ ΕΝΩΣΗΣ ΓΕΩΡΓΙΟΣ Σ. ΠΑΝΑΓΙΩΤΟΥ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ 2005
24. Διπλωματική εργασία της Γεωργίας Μαργιά, Ανάλυση και Πρόβλεψη Χρονοσειρών
25. http://www.it.uom.gr/teaching/ince_2e_gr/Text/C5/Programmingdatabases_6.htm
26. http://www0.dmst.aueb.gr/louridas/lectures/dais/jdbc/jdbc.html

ΠΑΡΑΡΤΗΜΑΤΑ
ΈΝΑ ΕΥΦΙΕΣ ΣΥΣΤΗΜΑ ΠΡΟΒΛΕΨΗΣ ΤΙΜΩΝ ΜΕΤΟΧΩΝ

ΠΑΡΑΡΤΗΜΑ Α : ΕΡΓΑΛΕΙΑ ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΑΝ

ΠΑΡΑΡΤΗΜΑ Β: ΧΡΗΣΙΜΕΣ ΒΙΒΛΙΟΘΗΚΕΣ

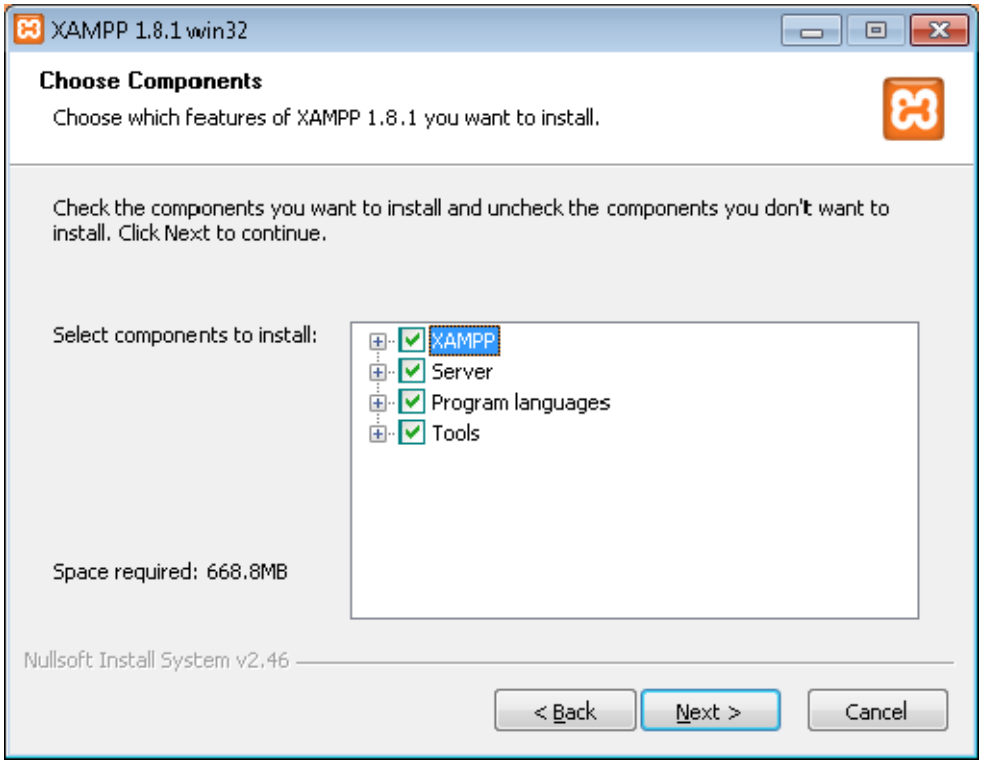
ΠΑΡΑΡΤΗΜΑ Γ: Χρήσιμες διευθύνσεις Internet για την Java

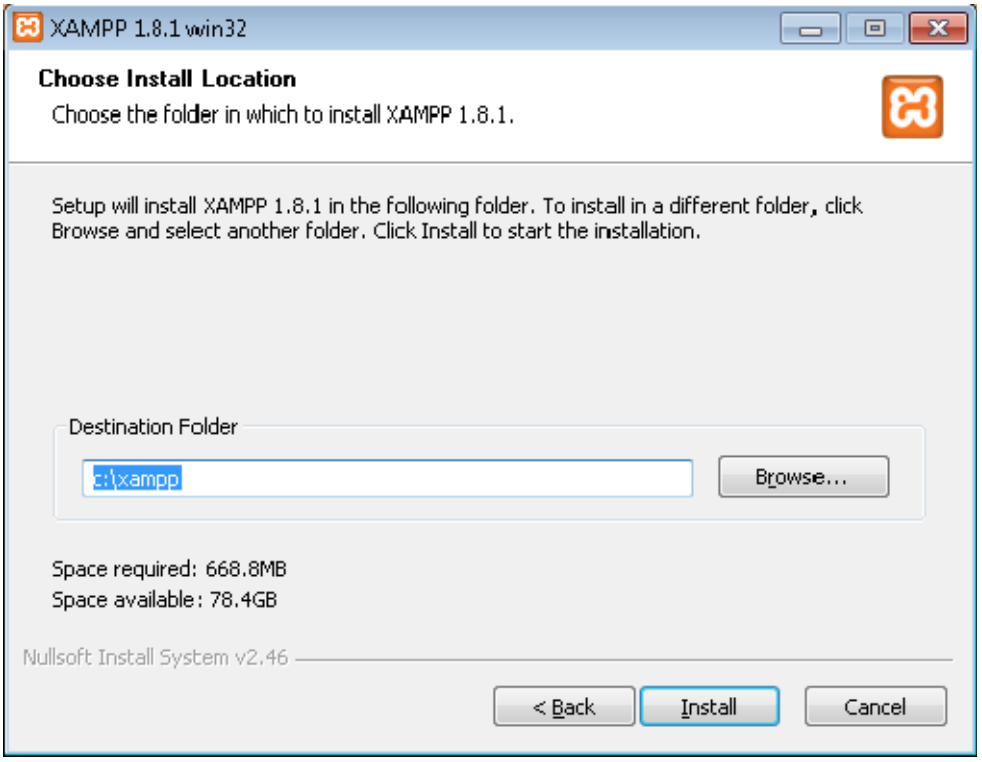
ΠΑΡΑΡΤΗΜΑ Δ : Παρουσίαση Εφαρμογής «Ένα Ευφυές Σύστημα Πρόβλεψης Τιμών Μετοχών».

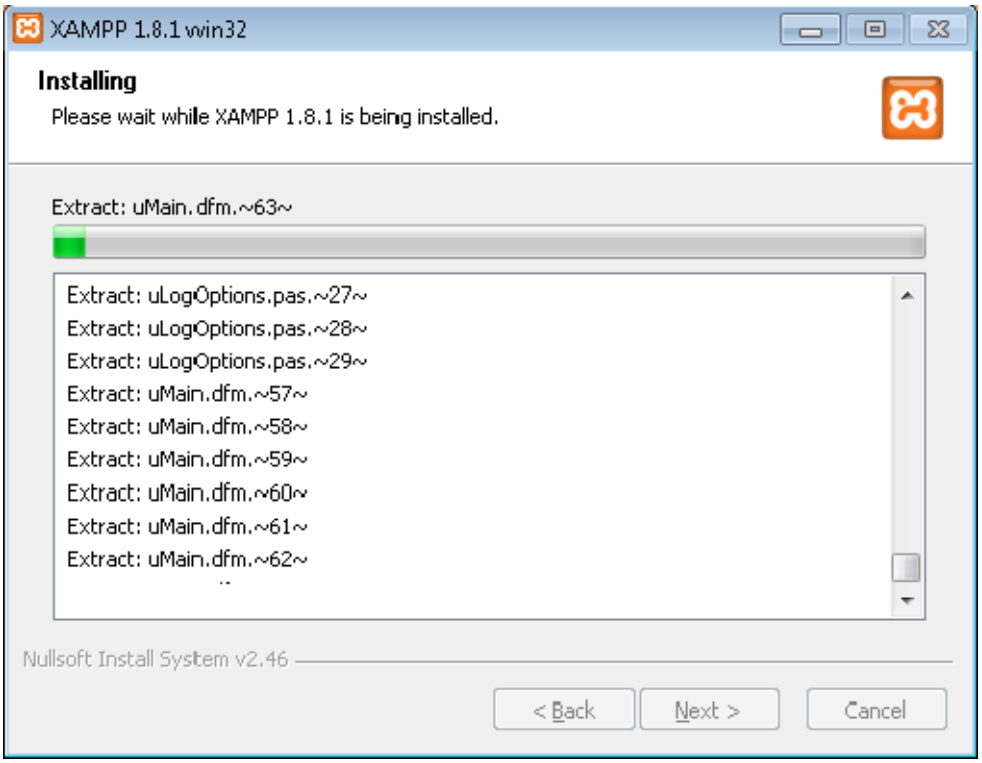
ΠΑΡΑΡΤΗΜΑ Α : ΕΡΓΑΛΕΙΑ ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΑΝ

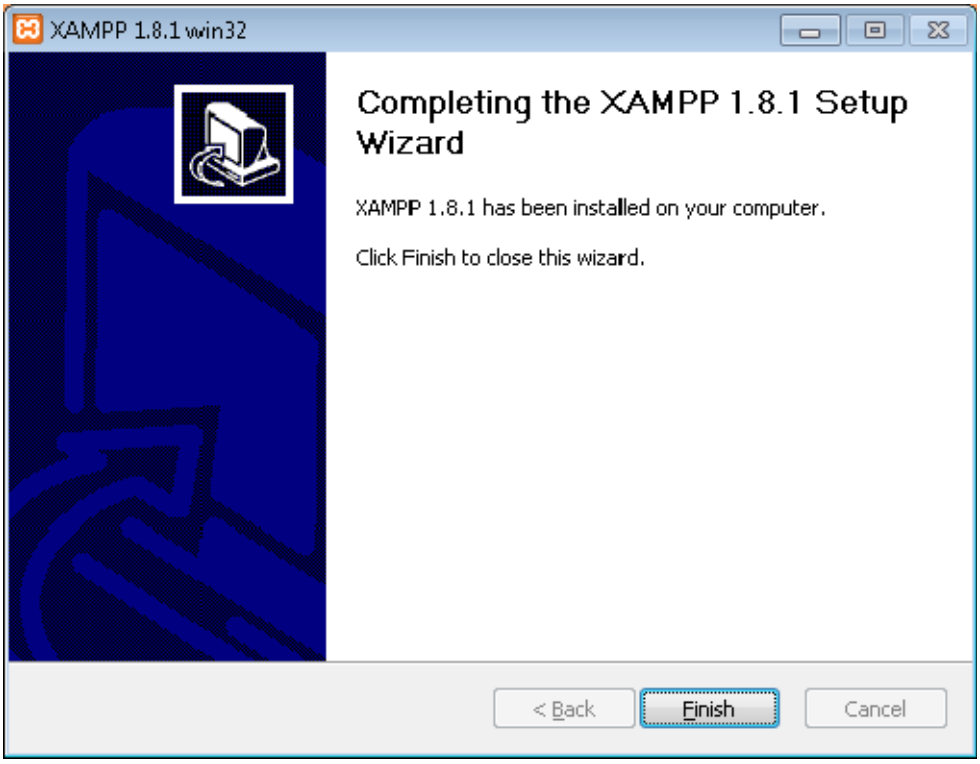
Α1: Εγκατάσταση του Xampp

Κατεβάζουμε το Xampp από την ιστοσελίδα http://www.apachefriends.org/en/xampp-windows.html και στη συνέχεια ακολουθούμε τις οδηγίες εγκατάστασης:

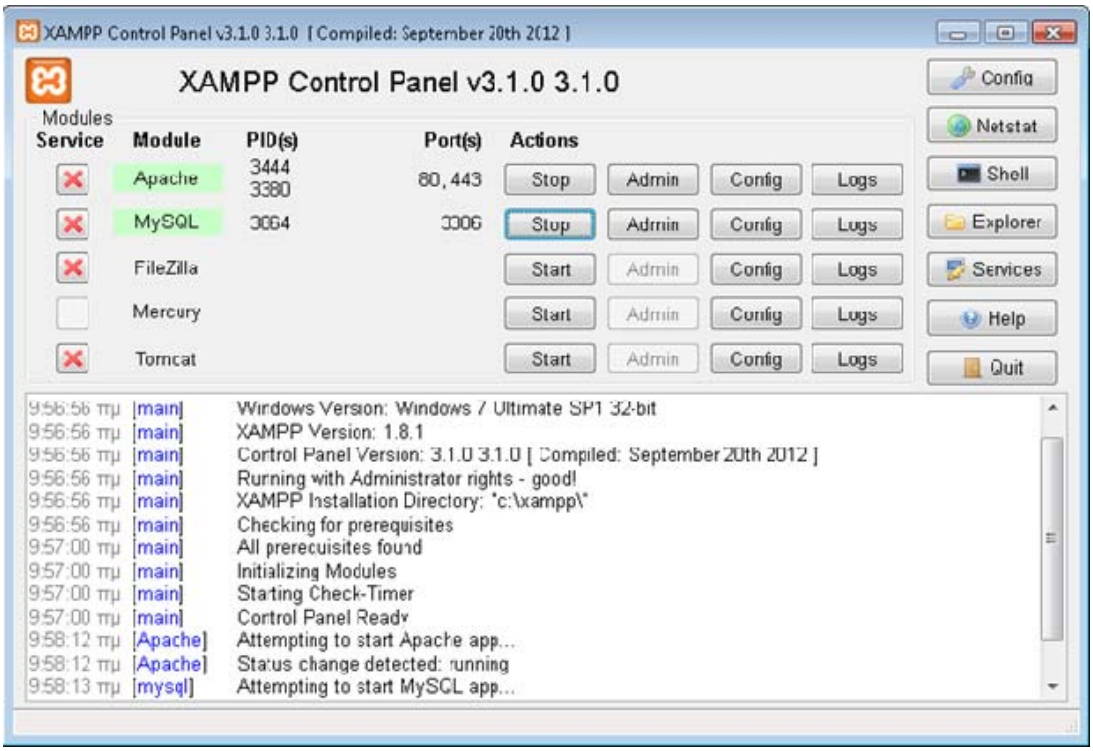



XAMPP 1.8.1 win32
Choose Install Location
Choose the folder in which to install XAMPP 1.8.1.
Setup will install XAMPP 1.8.1 in the following folder. To install in a different folder, click Browse and select another folder. Click Install to start the installation.
Destination Folder
c:\xampp
Browse...
Space required: 668.8MB
Space available: 78.4GB
Nullsoft Install System v2.46
< Back
Install
Cancel


XAMPP 1.8.1 win32
Installing
Please wait while XAMPP 1.8.1 is being installed.
Extract: uMain.dfm.~63~
Extract: uLogOptions.pas.~27~
Extract: uLogOptions.pas.~28~
Extract: uLogOptions.pas.~29~
Extract: uMain.dfm.~57~
Extract: uMain.dfm.~58~
Extract: uMain.dfm.~59~
Extract: uMain.dfm.~60~
Extract: uMain.dfm.~61~
Extract: uMain.dfm.~62~
Nullsoft Install System v2.46
< Back
Next >
Cancel



Περιβάλλον Control Panel Xampp



Α2: Εγκατάσταση JDK

Σε περίπτωση που θέλουμε να τρέξουμε προγράμματα και εφαρμογές java χρειαζόμαστε να εγκαταστήσουμε το περιβάλλον της Java καθώς και τα εργαλεία που υποστηρίζει, όπως τον compiler,τον debugger και εργαλεία για την εκτέλεση προγραμμάτων java. Για να εγκαταστήσετε το JDK (Java Development Toolkit)στον υπολογιστή χρειάζεται να κατεβάσουμε το jdk από την java sun Microsystems. Παρακάτω εμφανίζονται με τη σειρά τα βήματα που πρέπει να ακολουθηθούν.

1. Αρχικά μεταβαίνουμε στο ακόλουθο link: http://www.oracle.com/technetwork/java/javase/downloads/index.html για κατεβάσουμε όποια έκδοση jdk προτιμούμε.

2. Στη συνέχεια αφού επιλέξουμε την έκδοση που επιθυμούμε εμφανίζεται το μήνυμα για αποδοχή της άδεια ή όχι. Επιλέγουμε λοιπόν ότι αποδεχόμαστε τους όρους και επιλέγουμε το λειτουργικό σύστημα στο οποίο θα τρέχει το jdk. Εμείς επιλέγουμε τα Windows.

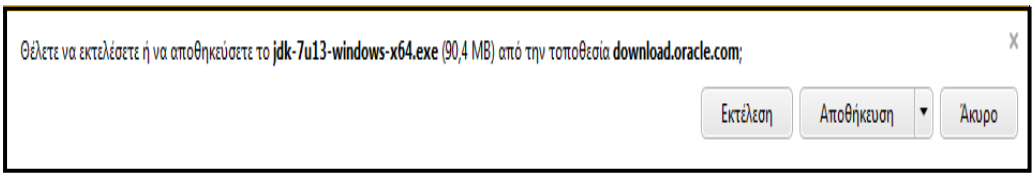
Java SE Development Kit 7u13

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

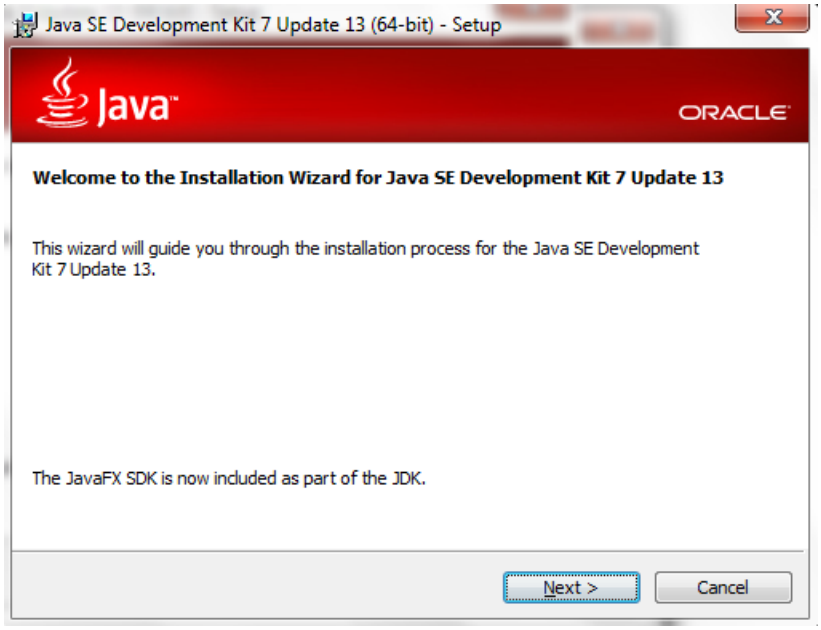
Thank you for accepting the Oracle Binary Code License Agreement for Java SE; you may now download this software.

Product / File Description	File Size	Download
Linux x86	106.64 MB	jdk-7u13-linux-i586.rpm
Linux x86	92.97 MB	jdk-7u13-linux-i586.tar.gz
Linux x64	104.77 MB	jdk-7u13-linux-x64.rpm
Linux x64	91.69 MB	jdk-7u13-linux-x64.tar.gz
Mac OS X x64	143.71 MB	jdk-7u13-macosx-x64.dmg
Solaris x86 (SVR4 package)	135.55 MB	jdk-7u13-solaris-i586.tar.Z
Solaris x86	91.95 MB	jdk-7u13-solaris-i586.tar.gz
Solaris x64 (SVR4 package)	22.54 MB	jdk-7u13-solaris-x64.tar.Z
Solaris x64	14.96 MB	jdk-7u13-solaris-x64.tar.gz
Solaris SPARC (SVR4 package)	135.83 MB	jdk-7u13-solaris-sparc.tar.Z
Solaris SPARC	95.28 MB	jdk-7u13-solaris-sparc.tar.gz
Solaris SPARC 64-bit (SVR4 package)	22.89 MB	jdk-7u13-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	17.58 MB	jdk-7u13-solaris-sparcv9.tar.gz
Windows x86	88.74 MB	jdk-7u13-windows-i586.exe
Windows x64	90.41 MB	jdk-7u13-windows-x64.exe

3. Μετά, θα μας ζητήσει να το αποθηκεύσουμε. Επιλέγουμε το φάκελο που θα το αποθηκεύσουμε και πατάμε Εκτέλεση.

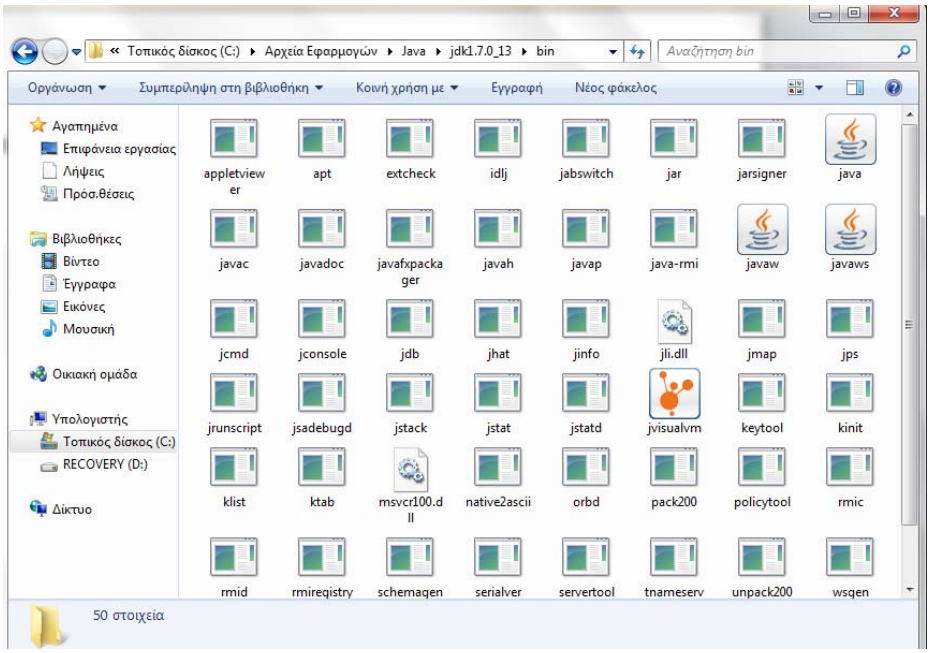


4. Αφού ολοκληρωθεί το κατέβασμα, πρέπει να το εγκαταστήσουμε στον υπολογιστή μας. Αυτό γίνεται απλά ακολουθώντας τα βήματα της εγκατάστασης .



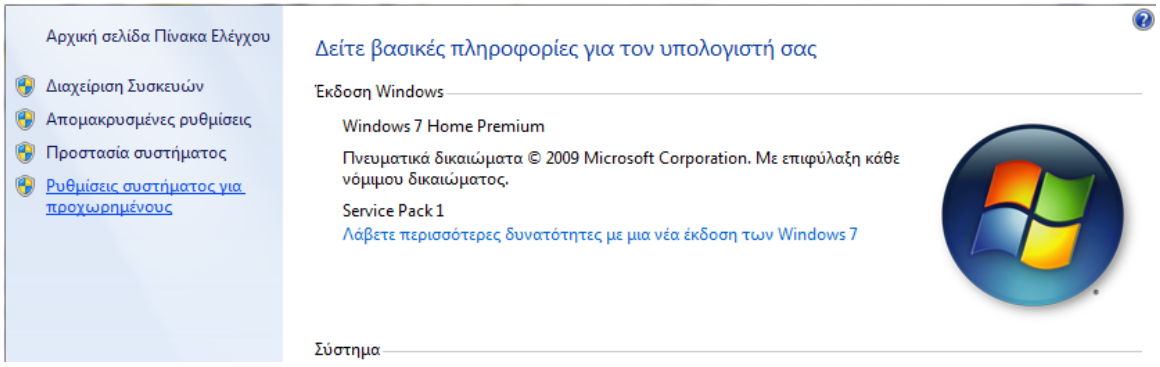
5. Η εγκατάσταση ολοκληρώθηκε με επιτυχία !

6. Τώρα είστε έτοιμοι να εκτελέσετε οποιοδήποτε αρχείο java.
7. Για να είναι σε θέση, εκτός από το να τρέξει, να μεταγλωττίσει ένα αρχείο θα πρέπει να εκτελέσετε ορισμένα επιπλέον βήματα. Συγκεκριμένα, θα πρέπει να βρείτε τον κατάλογο java που έχει εγκατασταθεί στο σύστημά σας. Συνήθως στο c: \ \ Files pogram \ java \ jdk1.7.0 \ bin.

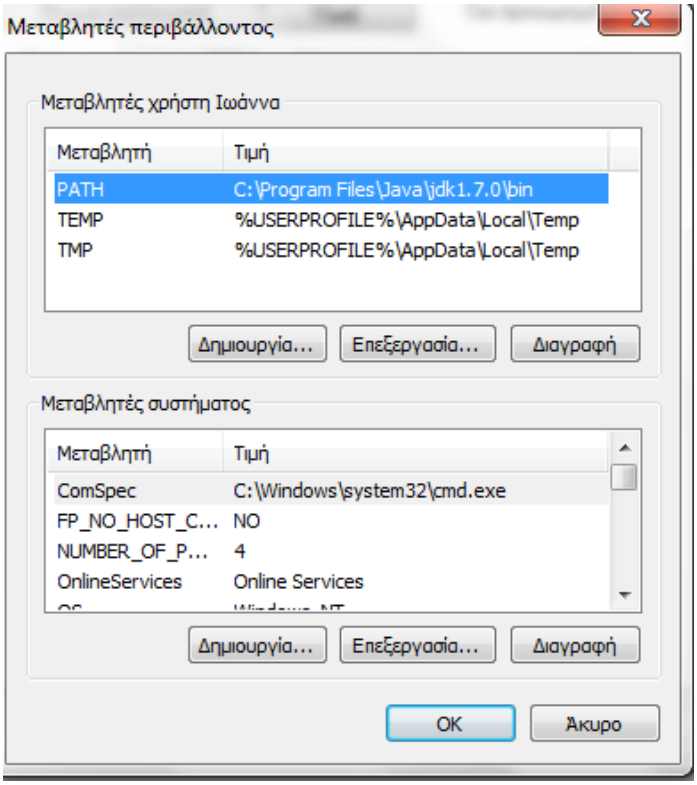


8. Στον υπολογιστή μου που βρίσκεται στην επιφάνεια εργασίας πατάμε δεξί κλικ και επιλέγουμε Ιδιότητες. Εναλλακτικά εάν ο υπολογιστής μας δεν βρίσκεται στην επιφάνεια

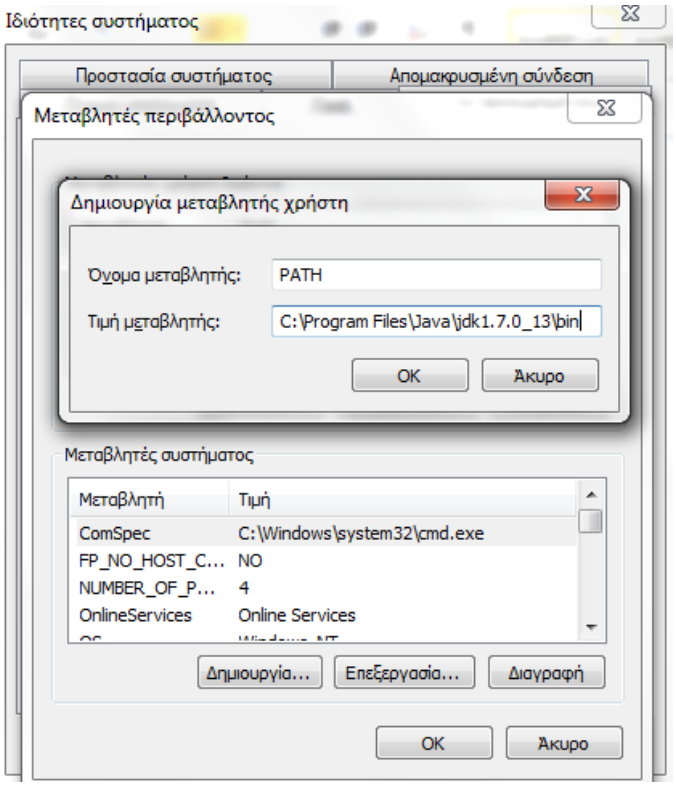
εργασίας από την Έναρξη επιλέγουμε υπολογιστής, στην συνέχεια επιλέγουμε Ιδιότητες Συστήματος → Ρυθμίσεις για Προχωρημένους.



9. Στο παράθυρο που εμφανίζεται πατάμε το κουμπί Μεταβλητές Περιβάλλοντος… και εμφανίζεται το ακόλουθο παράθυρο.



10. Πατάμε το κουμπί Δημιουργία… Δίνουμε σαν όνομα μεταβλητής “PATH” και Τιμή μεταβλητής το path που αναφέραμε στο βήμα 7.



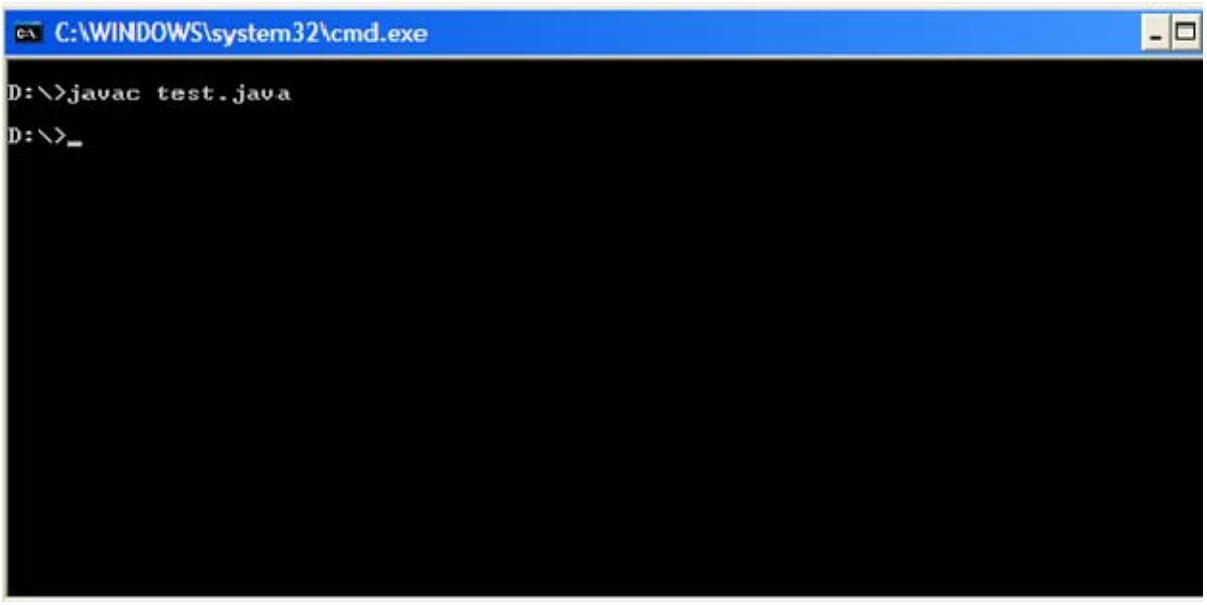
11. Πατήστε OK. Έχετε εγκαταστήσει με επιτυχία το JDK! Τώρα μπορείτε επίσης να μεταγλωττίσετε ένα αρχείο java.
12. Για να ελέγξετε αν το jdk έχει εγκατασταθεί σωστά στον υπολογιστή σας δημιουργείστε ένα αρχείο με όνομα test.java και προσθέστε τον παρακάτω κώδικα.
    public class test {

        public static void main(String args[]) {

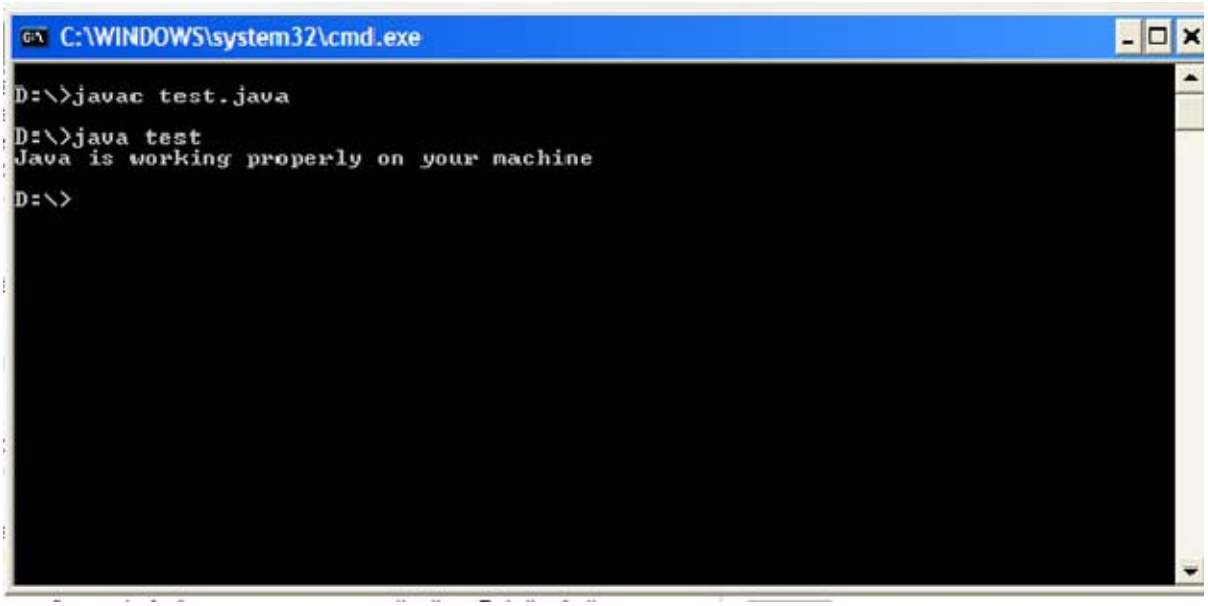
            System.out.println("**Java is working properly on your machine**");

    }

13. Ανοίξτε ένα cmd και εκτελέστε την εντολή **javac test.java**. javac είναι ένα εργαλείο που χρησιμοποιείται για να μεταγλωττίσετε ένα αρχείο java. Από τη στιγμή που συντάσσει ένα αρχείο java με επιτυχία θα δημιουργήσει αυτόματα ένα αρχείο που ονομάζεται test.class στον ίδιο κατάλογο.



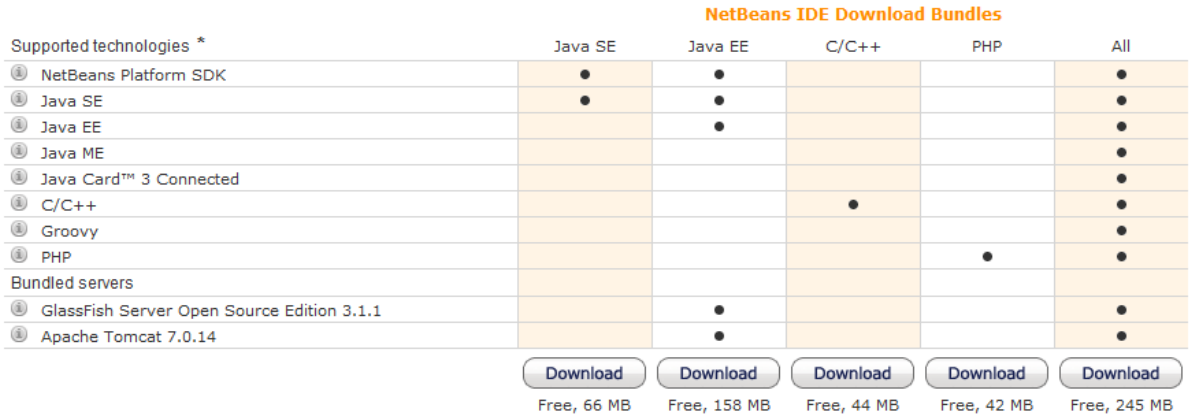
14. Συγχαρητήρια!!! Έχετε εγκαταστήσει την java με επιτυχια.



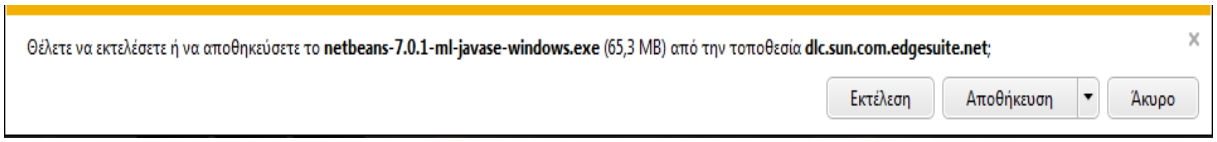
Α3:Εγκατάσταση Netbeans

Το Netbeans είναι ένα χρήσιμο εργαλείο, αφού μας προσφέρει μια πλατφόρμα για να δημιουργούμε εφαρμογές και προγράμματα. Για να το εγκαταστήσουμε λοιπόν, στον υπολογιστή μας ακολουθούμε τα παρακάτω βήματα.

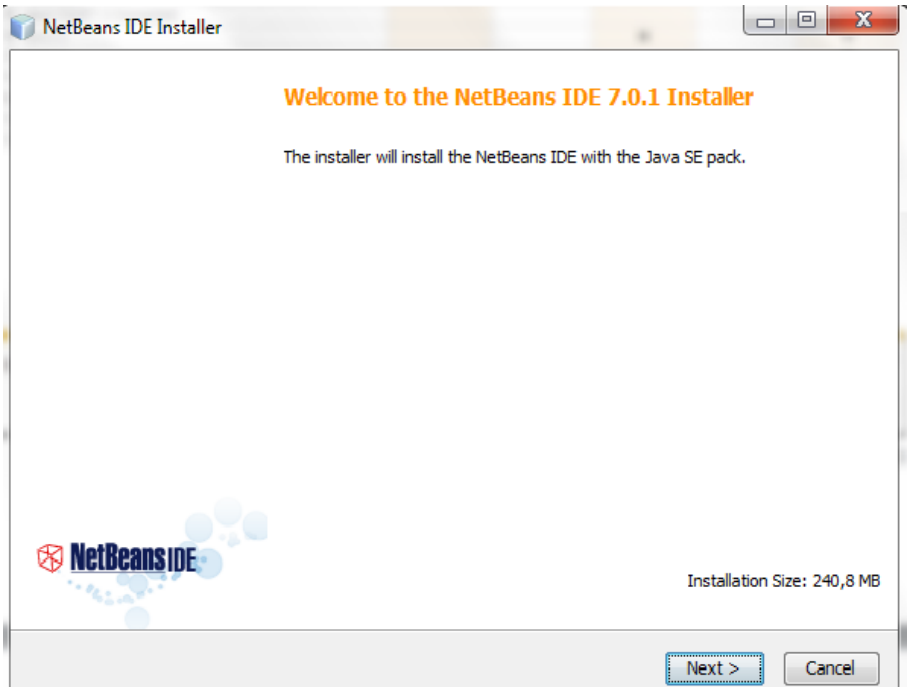
1. Πάμε στο ακόλουθο link http://netbeans.org/downloads/7.0.1/ και επιλέγουμε την έκδοση που επιθυμούμε. Πατάμε το κουμπί Download για να το κατεβάσουμε.



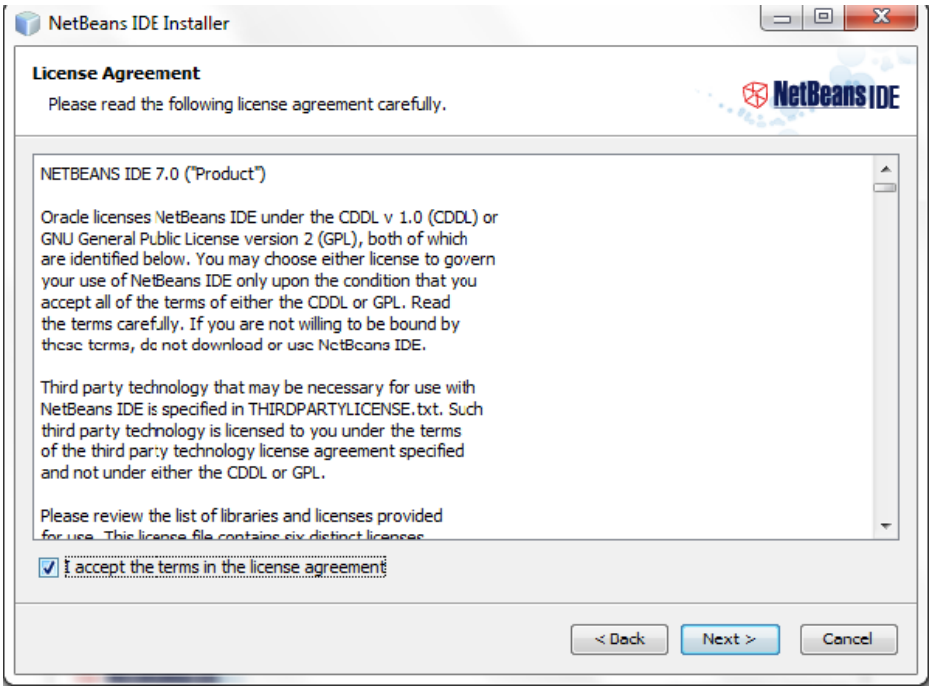
2. Εμφανίζεται το μήνυμα όπου πρέπει να επιλέξουμε την τοποθεσία αποθήκευσης στον υπολογιστή μας. Αφού ολοκληρωθεί η λήψη πατάμε το κουμπί εκτέλεση.



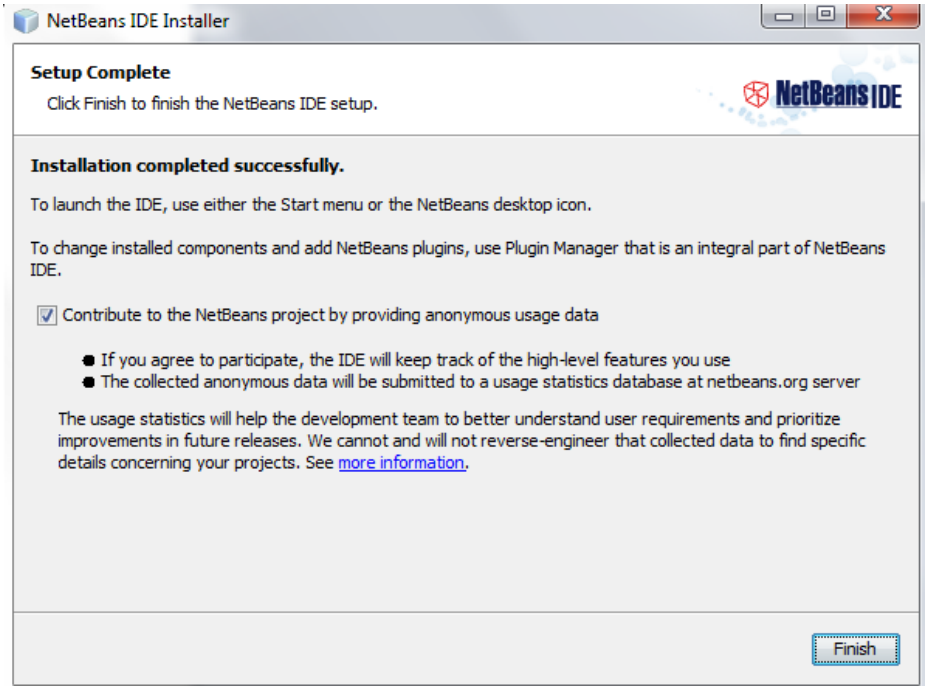
3. Στην συνέχεια ακολουθούμε τα βήματα του οδηγού εγκατάστασης. Είναι πάρα πολύ απλό!



4. Στη συνέχεια τσεκάρουμε ότι αποδεχόμαστε τους όρους εγκατάστασης και συνεχίζουμε.

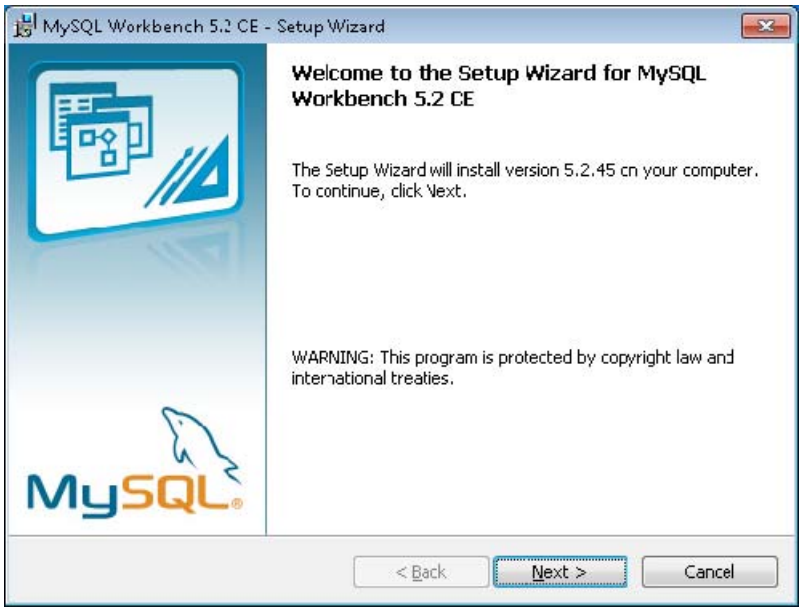


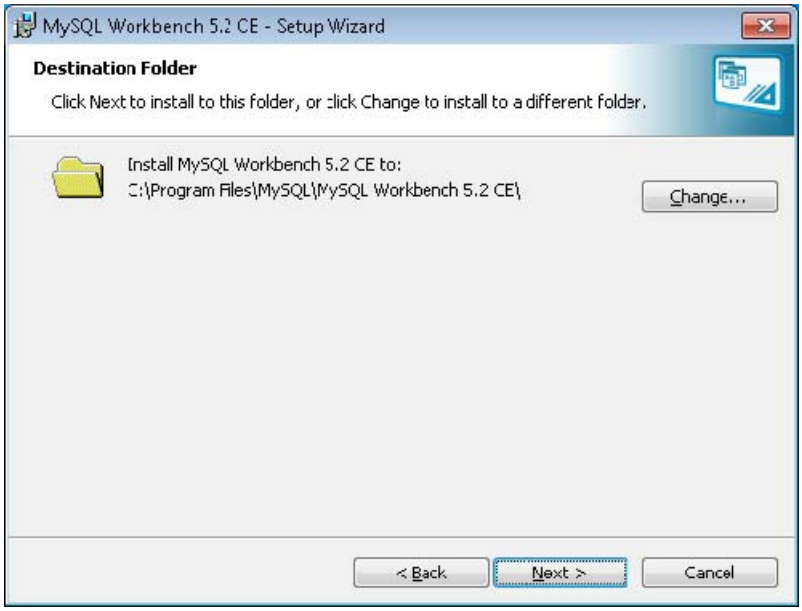
5. Σε αυτό το σημείο πατάμε το κουμπί Finish για να τελειώσουμε την εγκαταστάτη. Συγχαρητήρια ο Netbeans εγκαταστάθηκε με επιτυχία!!

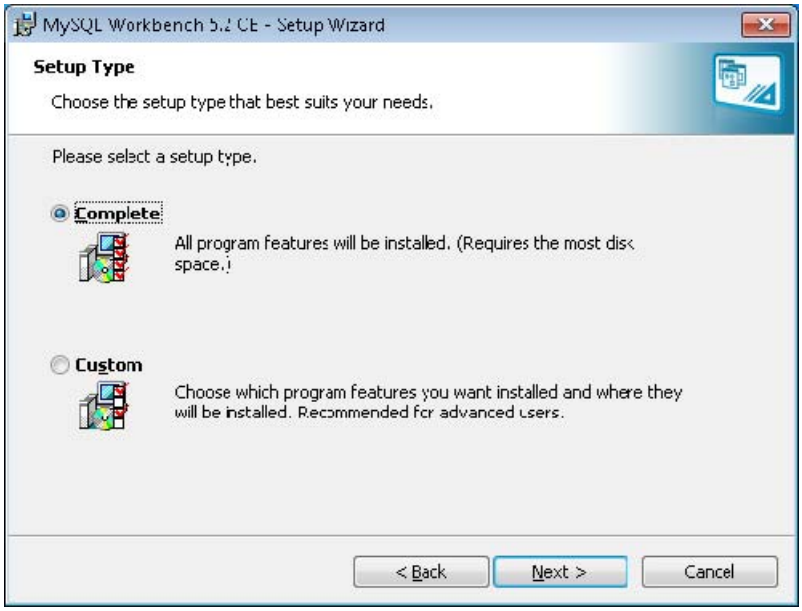


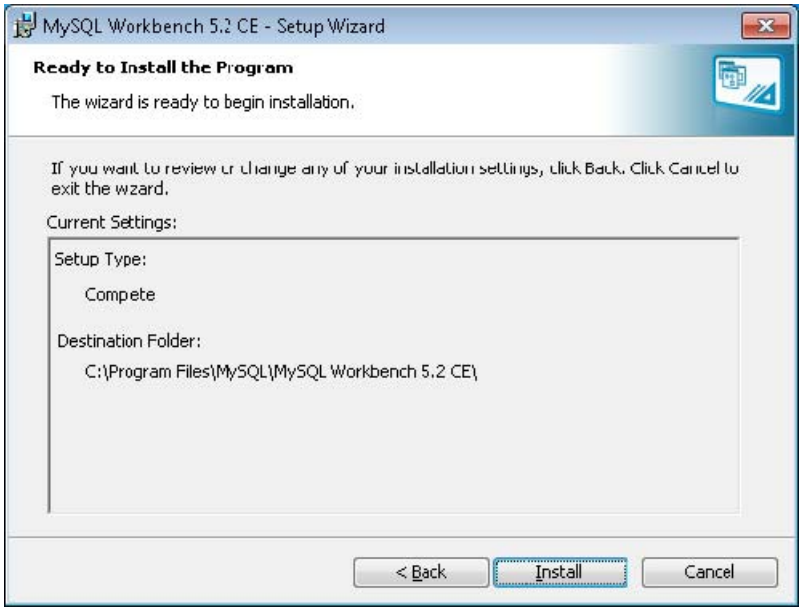
Α4: Εγκατάσταση MySQL Workbench

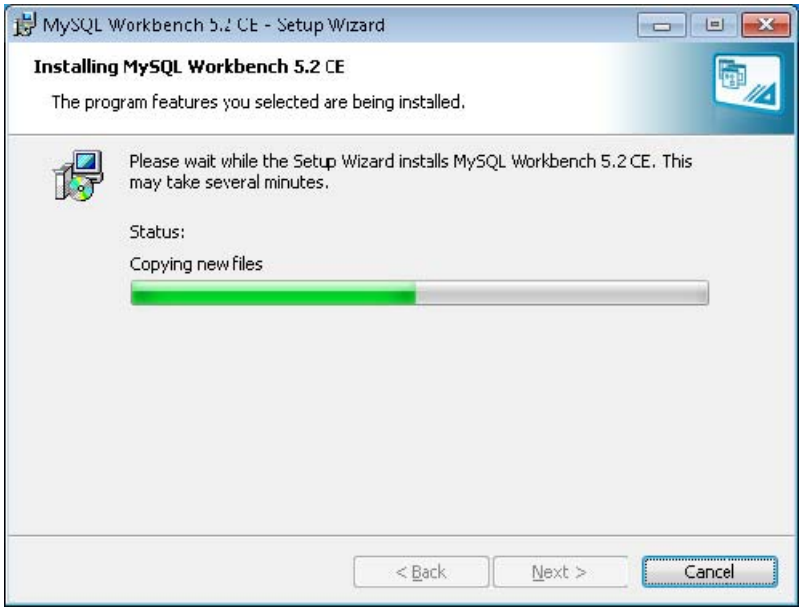
Κατεβάζουμε το αρχείο από την ιστοσελίδα http://www.mysql.com/downloads/workbench/ και ακολουθούμε τις παρακάτω οδηγίες εγκατάστασης:

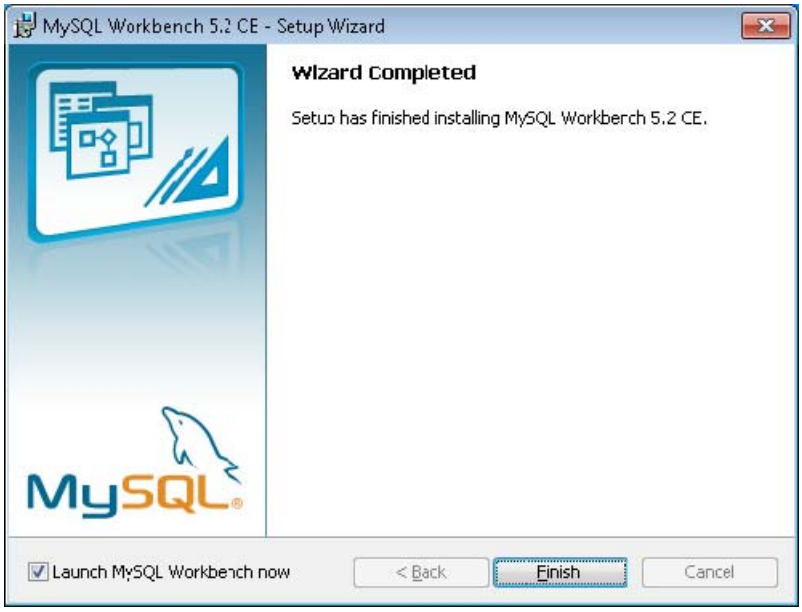




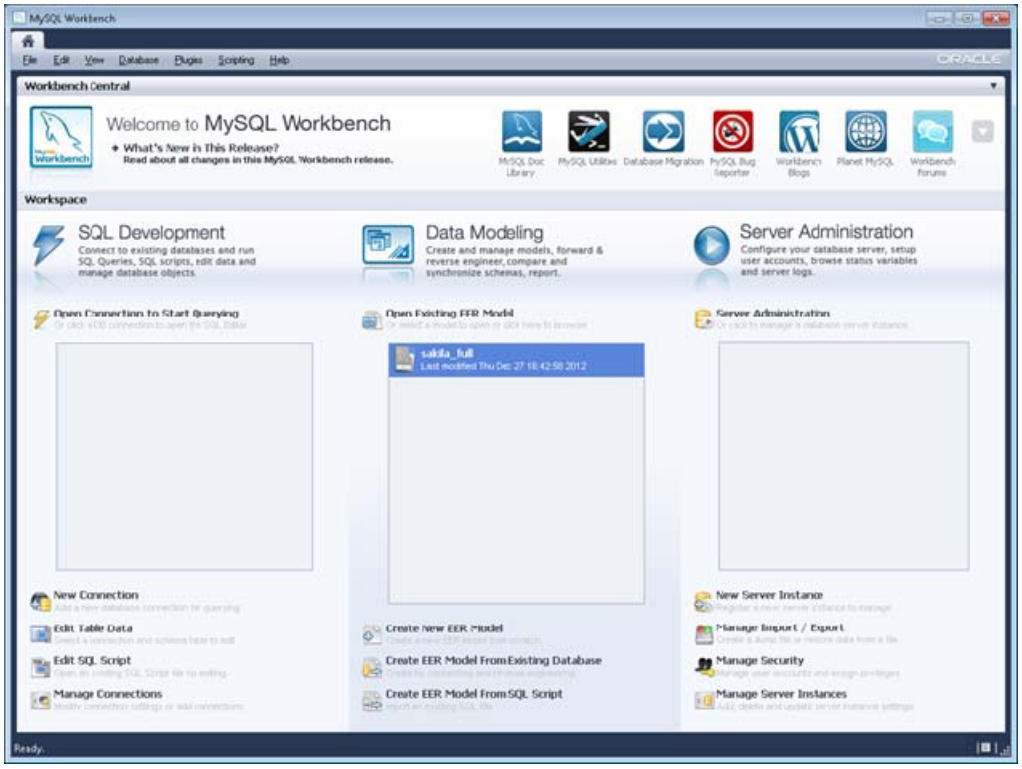

MySQL Workbench 5.2 CE - Setup Wizard
Setup Type
Choose the setup type that best suits your needs.
Please select a setup type.
Complete
All program features will be installed. (Requires the most disk space.)
Custom
Choose which program features you want installed and where they will be installed. Recommended for advanced users.
< Back
Next >
Cancel


MySQL Workbench 5.2 CE - Setup Wizard
Ready to Install the Program
The wizard is ready to begin installation.
If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.
Current Settings:
Setup Type:
Compete
Destination Folder:
C:\Program Files\MySQL\MySQL Workbench 5.2 CE\
< Back
Install
Cancel


MySQL Workbench 5.2 CE - Setup Wizard
Installing MySQL Workbench 5.2 CE
The program features you selected are being installed.
Please wait while the Setup Wizard installs MySQL Workbench 5.2 CE. This may take several minutes.
Status:
Copying new files
< Back
Next >
Cancel


MySQL Workbench 5.2 CE - Setup Wizard
Wizard Completed
Setup has finished installing MySQL Workbench 5.2 CE.
MySQL
Launch MySQL Workbench now
< Back
Finish
Cancel

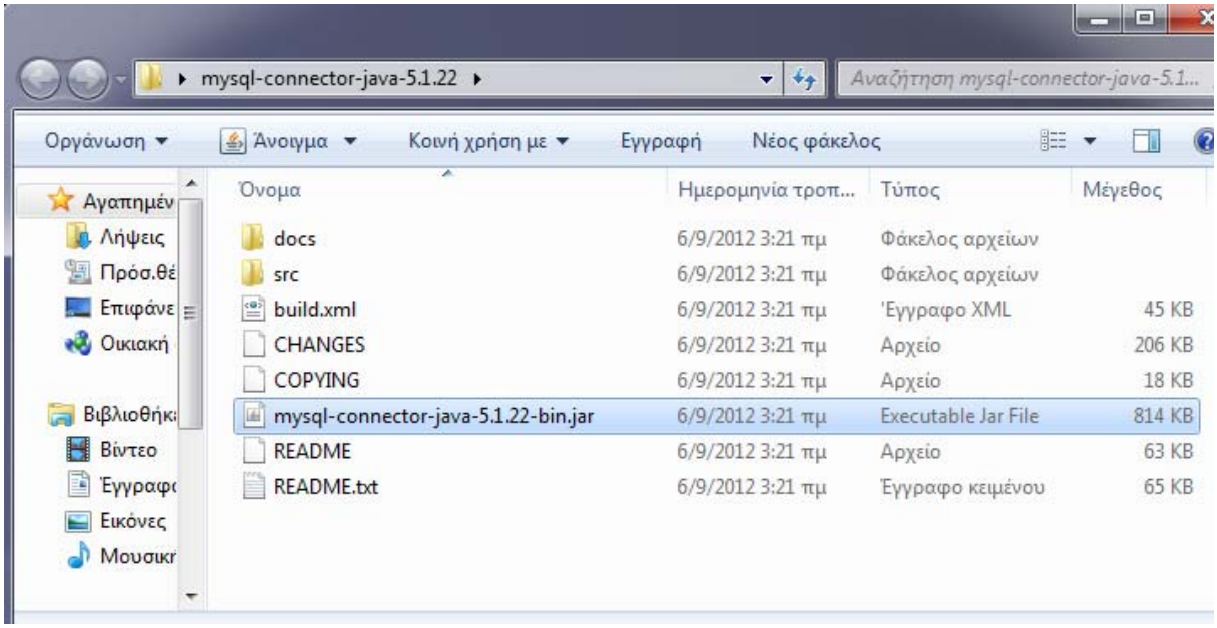
Περιβάλλον MySQL Workbench



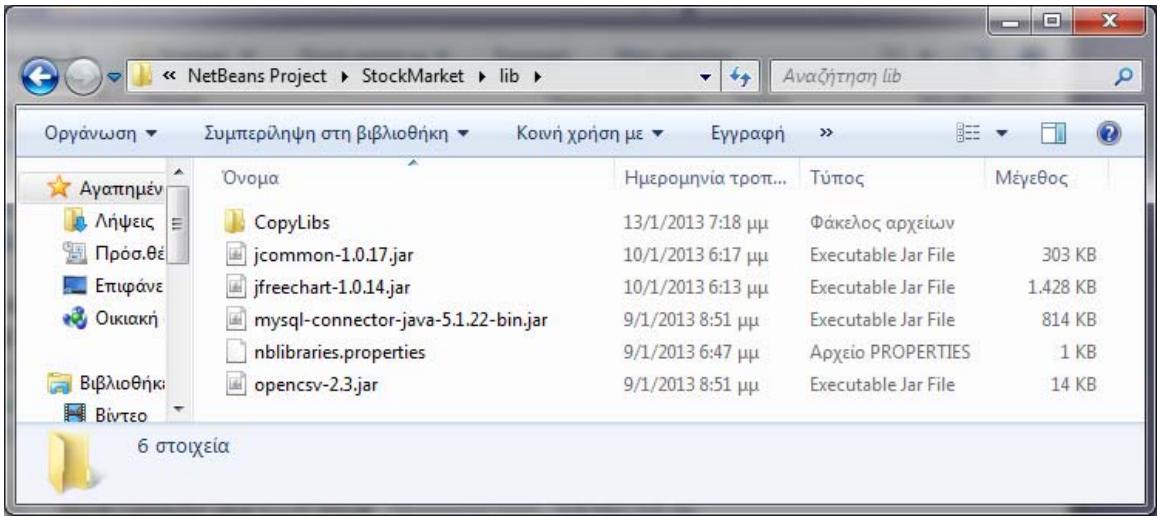
ΠΑΡΑΡΤΗΜΑ Β: ΧΡΗΣΙΜΕΣ ΒΙΒΛΙΟΘΗΚΕΣ

Β1:Εισαγωγή Mysql-Connector-java 5.1.22

Κατεβάζουμε το αρχείο από την ιστοσελίδα http://downloads.mysql.com/archives.php?p=mysql-connector-java-5.1&o=other σε μορφή .zip εξάγουμε όλα τα αρχεία που υπάρχουν σε αυτό και επιλέγουμε το αρχείο **mysql-connector-java-5.1.22-bin.jar**



Στη συνέχεια το εισάγουμε το αρχείο στο φάκελο εγκατάστασης του Netbeans στην θέση **C:\Program Files\NetBeans 7.0.1\profiler** στον κατάλογο **\lib** ή στο φάκελο αποθήκευσης του project μας όπου και αν βρίσκεται στον υπολογιστή στον κατάλογο lib πιο συγκεκριμένα **C:\Users\vaio\Desktop\ΠΤΥΧΙΑΚΗ\ NetBeans Project\StockMarket\lib**



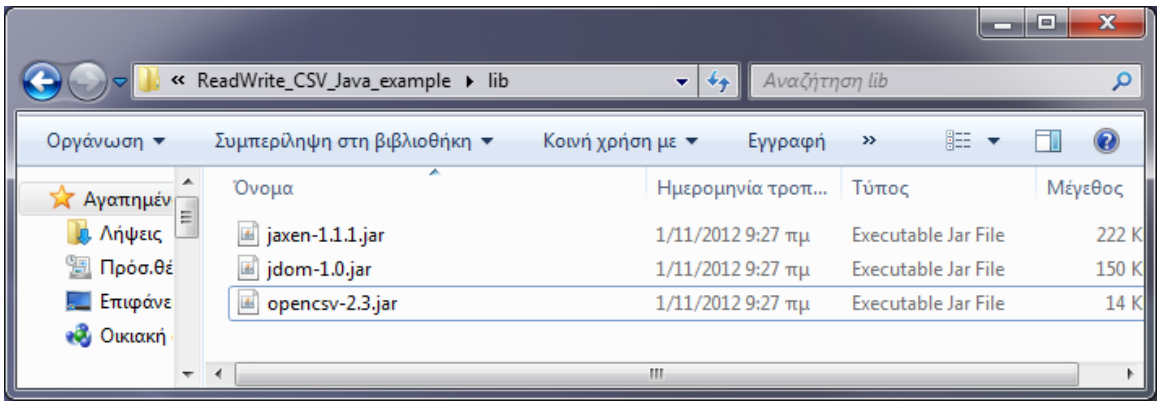
Το αρχείο αυτό πετυχαίνει τη σύνδεση της java με την βάση δεδομένων.

B2: Comma-separated Files (CSV)

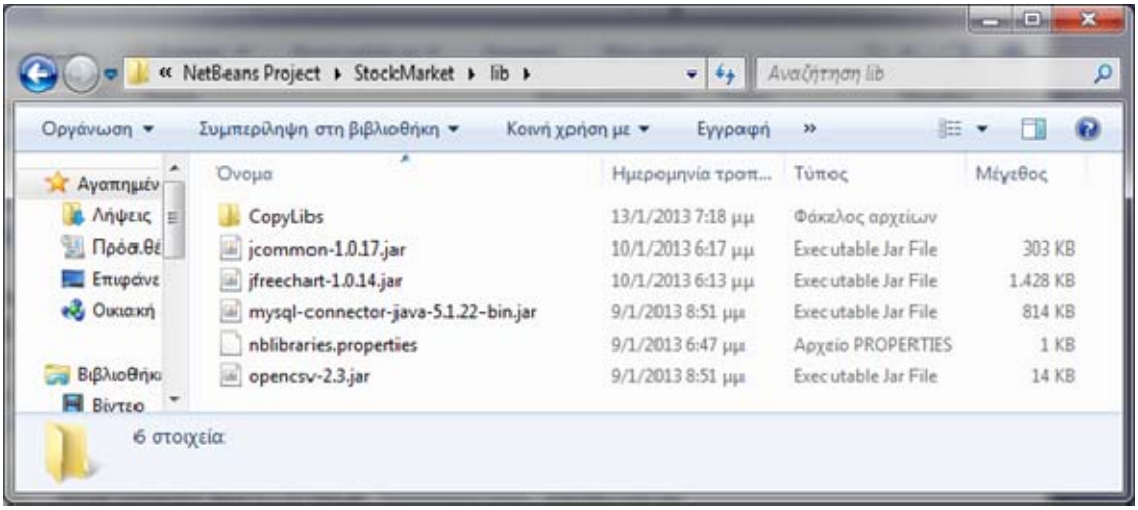
Η java εν γένει δεν υποστηρίζει CSV αρχεία αλλά βασίζεται σε βιβλιοθήκες. Η Opencsv είναι μία από τις καλύτερες Βιβλιοθήκες που είναι διαθέσιμες γι' αυτό το λόγο. Είναι ανοιχτού κώδικα και διατίθεται στην αγορά με άδεια Apache 2.0 το οποίο το καθιστά δυνατό για εμπορική χρήση.
Πριν αναλύσουμε ένα CVS αρχείο θα χρειαστούμε ορισμένα εργαλεία:

Tools & Technologies
1. Java JDK 1.5 or above
2. OpenCSV library v1.8 or above
3. Eclipse 3.2 above

Κατεβάζουμε το αρχείο ReadWrite_CSV_Java_example.zip (356 KB) από την ιστοσελίδα http://viralpatel.net/blogs/java-read-write-csv-file/ σε μορφή .zip εξάγουμε τα αρχεία που υπάρχουν σε αυτό και επιλέγουμε από τον κατάλογο lib το αρχείο opencsv-2.3.jar

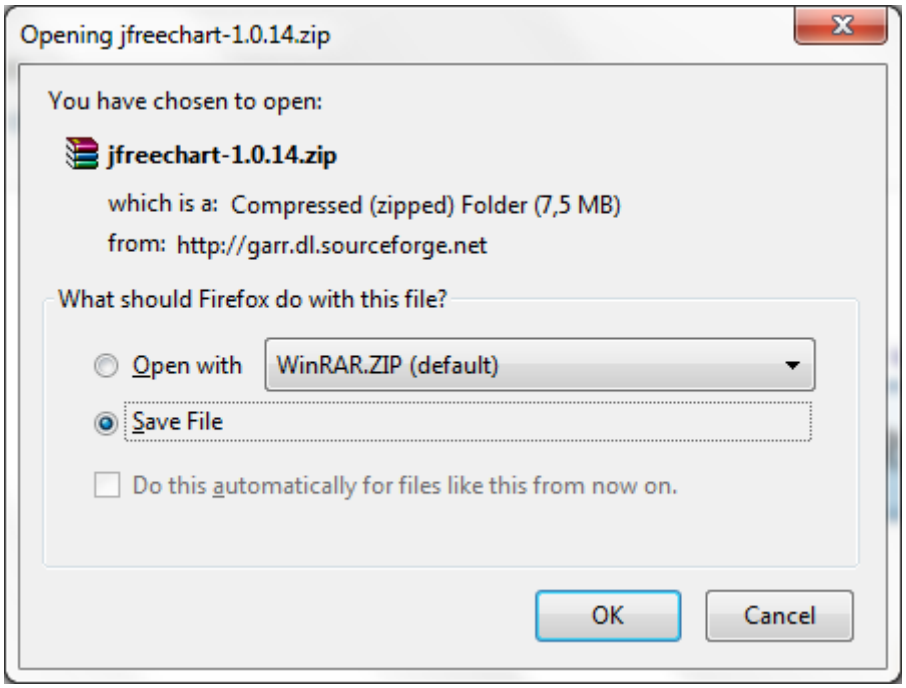


Στη συνέχεια το εισάγουμε το αρχείο στο φάκελο εγκατάστασης του Netbeans στην θέση C:\Program Files\NetBeans 7.0.1\profiler στον κατάλογο \lib ή στο φάκελο αποθήκευσης του project μας όπου και αν βρίσκεται στον υπολογιστή στον κατάλογο lib πιο συγκεκριμένα C:\Users\vaio\Desktop\ΠΤΥΧΙΑΚΗ\ NetBeans Project\StockMarket\lib

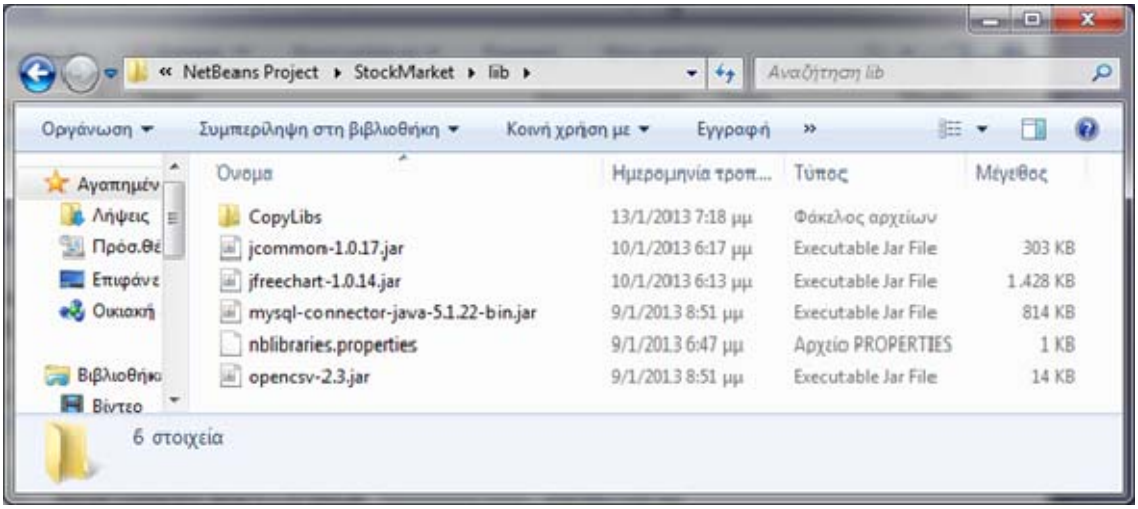


Β3:Εισαγωγή του Jfreechart.

Για να εμφανίζονται οι μετοχές μας σε μορφή γραφήματος ακολουθούμε την παρακάτω διαδικασία. Εισάγουμε την παρακάτω διεύθυνση: http://sourceforge.net/projects/jfreechart/files/1.%20JFreeChart/1.0.14/jfreechart-1.0.14.zip/download στο internet, το αποθηκεύουμε σε ένα σημείο της επιλογής μας και το αρχείο μας αρχίζει να κατεβαίνει.



Στη συνέχεια το εισάγουμε το αρχείο στο φάκελο εγκατάστασης του Netbeans στην θέση C:\Program Files\NetBeans 7.0.1\profiler στον κατάλογο \lib ή στο φάκελο αποθήκευσης του project μας όπου και αν βρίσκεται στον υπολογιστή στον κατάλογο lib πιο συγκεκριμένα C:\Users\vaio\Desktop\ΠΤΥΧΙΑΚΗ\ NetBeans Project\StockMarket\lib



ΠΑΡΑΡΤΗΜΑ Γ: Χρήσιμες διευθύνσεις Internet για την Java

Γ1: 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.

The Java Tutorials - Trail: Learning the Java Language:

Lesson: Object-Oriented Programming Concepts
http://download.oracle.com/javase/tutorial/java/concepts/index.html

Lesson: Interface and Inheritance
http://download.oracle.com/javase/tutorial/java/IandI/subclasses.html

The Java Tutorials - Trail: Creating a GUI With JFC/Swing:

Lesson: A Brief Introduction to the Swing Package
http://download.oracle.com/javase/tutorial/ui/overview/index.html

Lesson: Using Swing Components
http://download.oracle.com/javase/tutorial/uiswing/components/index.html

ΠΑΡΑΡΤΗΜΑ Δ : Παρουσίαση Εφαρμογής «Ένα Ευφυές Σύστημα Πρόβλεψης Τιμών Μετοχών»

ΕΝΑ ΕΥΦΥΕΣ ΣΥΣΤΗΜΑ ΠΡΟΒΛΕΨΗΣ ΤΙΜΩΝ ΜΕΤΟΧΩΝ

Ιωάννα Φιλιππάκη (ΑΜ: 2700)
Αργυρώ Σαρικάκη (ΑΜ: 2660)

Επιβλέπων καθηγητής: Νίκος Παπαδάκης
Επιτροπή Αξιολόγησης: Νίκος Παπαδάκης
Χαράλαμπος Μανιφάβας
Ηλίας Χατζάκης

Ένα Ευφυές Σύστημα Πρόβλεψης Τιμών Μετοχών

- Το Σύστημα πρόβλεψης τιμών μετοχών αποτελεί ένα λογισμικό διαχείρισης βάσης δεδομένων σε συνδυασμό με τον αντικειμενοστραφή προγραμματισμό (Java). Είναι μία εφαρμογή η οποία έχει τη δυνατότητα πρόβλεψης τιμών μετοχών, η οποία θα μπορούσε να αποφέρει σημαντικά κέρδη στους επενδυτές τους.
- Οι προβλέψεις είναι ένα βασικό στοιχείο των χρηματοπιστωτικών αγορών για το λόγο αυτό, η εφαρμογή έχει προσαρμοστεί στις ανάγκες των εκάστοτε επενδυτών που δραστηριοποιούνται στον χώρο του χρηματιστηρίου για την ορθή πρόβλεψη των τιμών και των αποδόσεων των μετοχών.

Βασικές Λειτουργίες

- Καταγραφή ημερήσιων δεδομένων κάθε μετοχής Date, Open, High, Low, Close, Volume.
- Καταγραφή μετοχών.
- Δημιουργία, μετονομασία, διαγραφή Sectors στην εφαρμογή.
- Δημιουργία, μετονομασία, διαγραφή, εισαγωγή μετοχών στην εφαρμογή.
- Επιλογή δεικτών τεχνικής ανάλυσης για την πρόβλεψη των τιμών των μετοχών.
- Επιλογή συγκεκριμένου χρονικού διαστήματος για την πρόβλεψη τιμών των μετοχών.
- Εμφάνιση γραφικής παράστασης με την πρόβλεψη τιμών των μετοχών.
- Εμφάνιση ποσοστών αποτυχίας πρόβλεψης.

Ένα Ευφυές Σύστημα Πρόβλεψης Τιμών Μετοχών

- **Αποτέλεσμα:** Με την εφαρμογή οι επενδυτές που ασχολούνται με το χρηματιστήριο, έχουν στη διάθεση τους ένα σύστημα πρόβλεψης τιμών μετοχών που θα τους βοηθήσει να μεγιστοποιήσουν τα κέρδη τους. Η αλλαγή των προσδοκιών των επενδυτών συχνά είναι απότομη και βασίζεται σε ειδήσεις γύρω από την εταιρεία.

- Πιο συγκεκριμένα, όταν η μετοχή είναι σε ανοδική πορεία οι αγοραστές είναι πιο πολλοί από τους πωλητές. Όταν η μετοχή είναι σε καθοδική πορεία αυτοί που πουλάνε είναι πιο πολλοί από αυτούς που αγοράζουν.

Εργαλεία

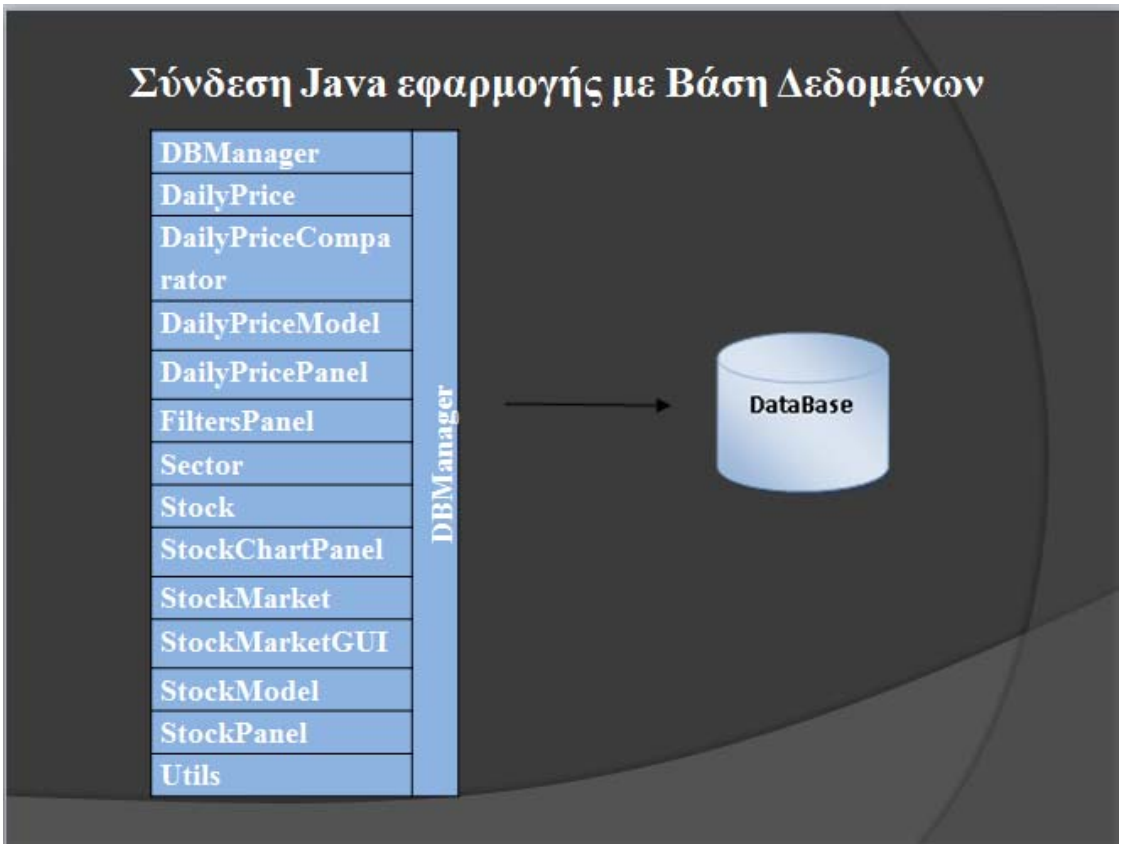
- Η ανάπτυξη έγινε σε Oracle Java, γλώσσα αντικειμενοστραφή προγραμματισμού η οποία διατίθεται δωρεάν.
- Η βάση δεδομένων είναι σε MySQL την πιο δημοφιλή βάση δεδομένων ανοικτού κώδικα η οποία λειτουργεί σε περισσότερες από 20 πλατφόρμες.

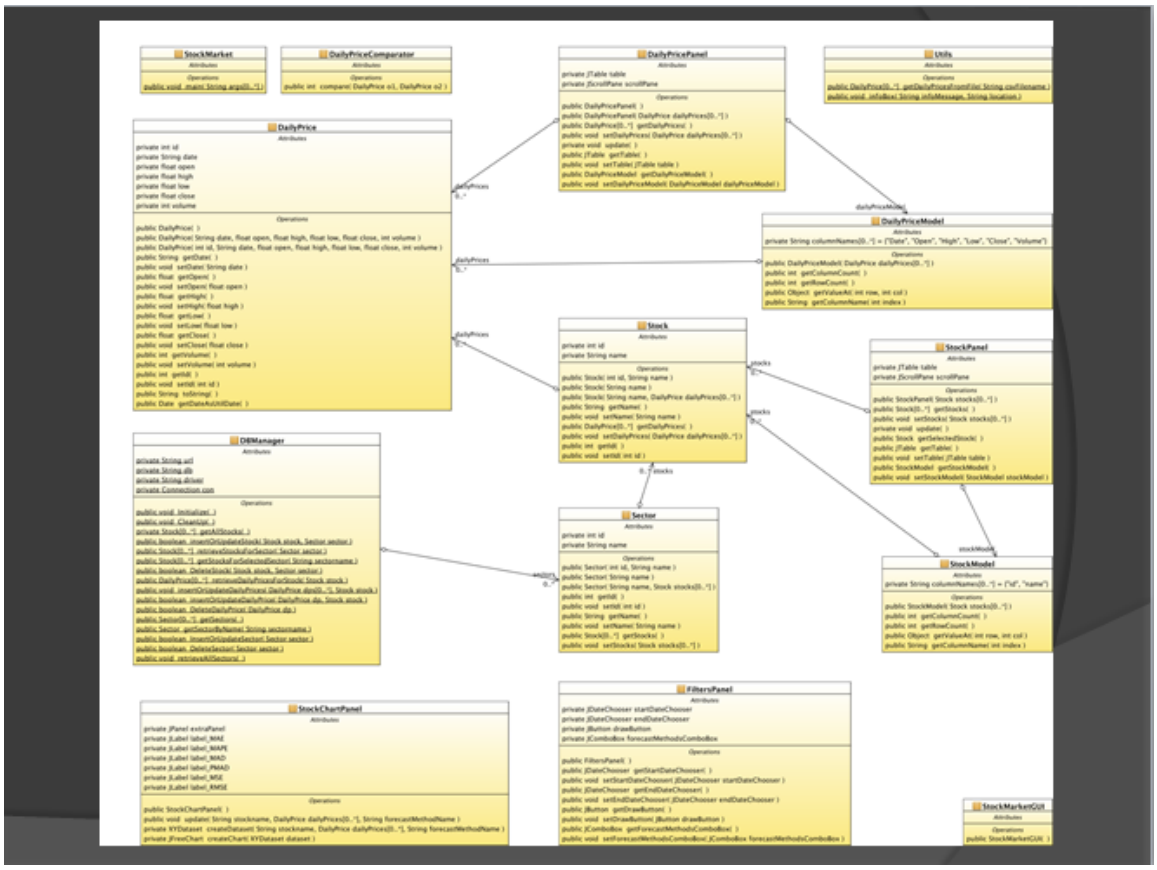
Γιατί Java

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.
MySQL


Σύνδεση Java εφαρμογής με Βάση Δεδομένων
DBManager
DailyPrice
DailyPriceComparator
DailyPriceModel
DailyPricePanel
FiltersPanel
Sector
Stock
StockChartPanel
StockMarket
StockMarketGUI
StockModel
StockPanel
Utils
DBManager
DataBase



Ιδιότητες – Μέθοδοι
Class DBManager

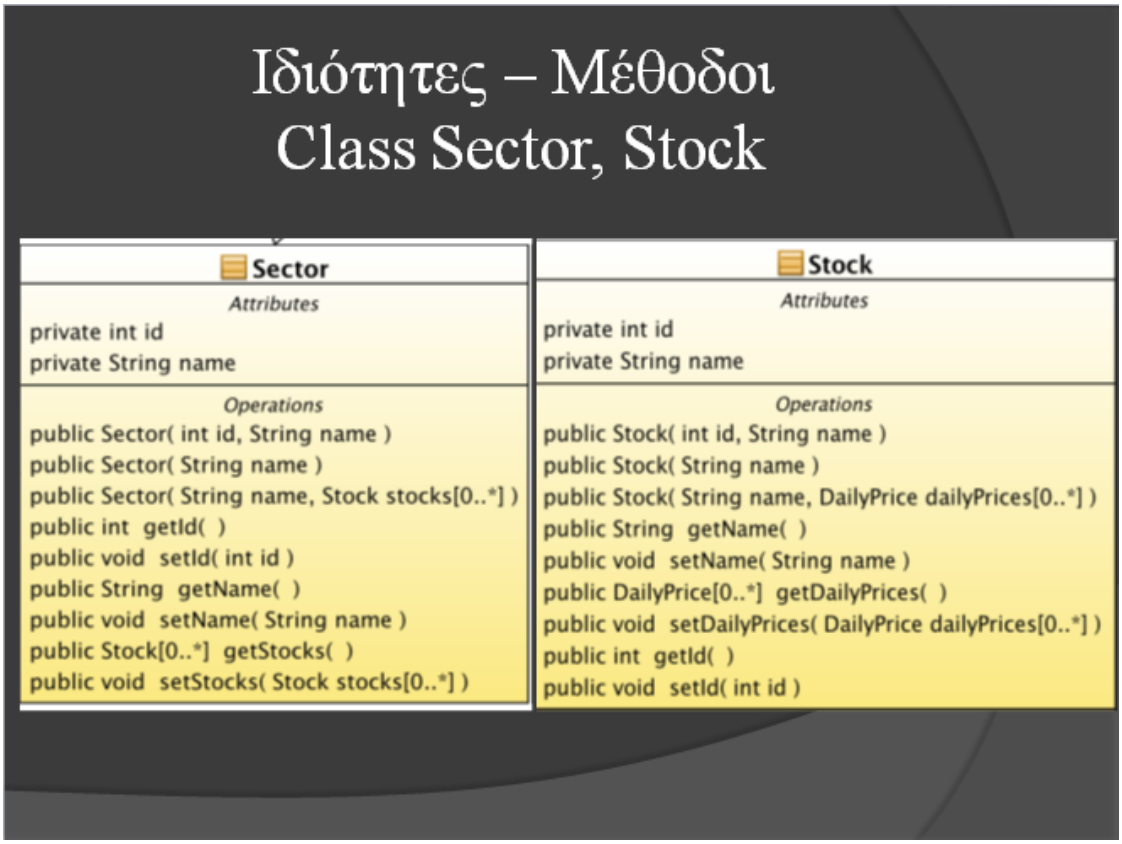
DBManager

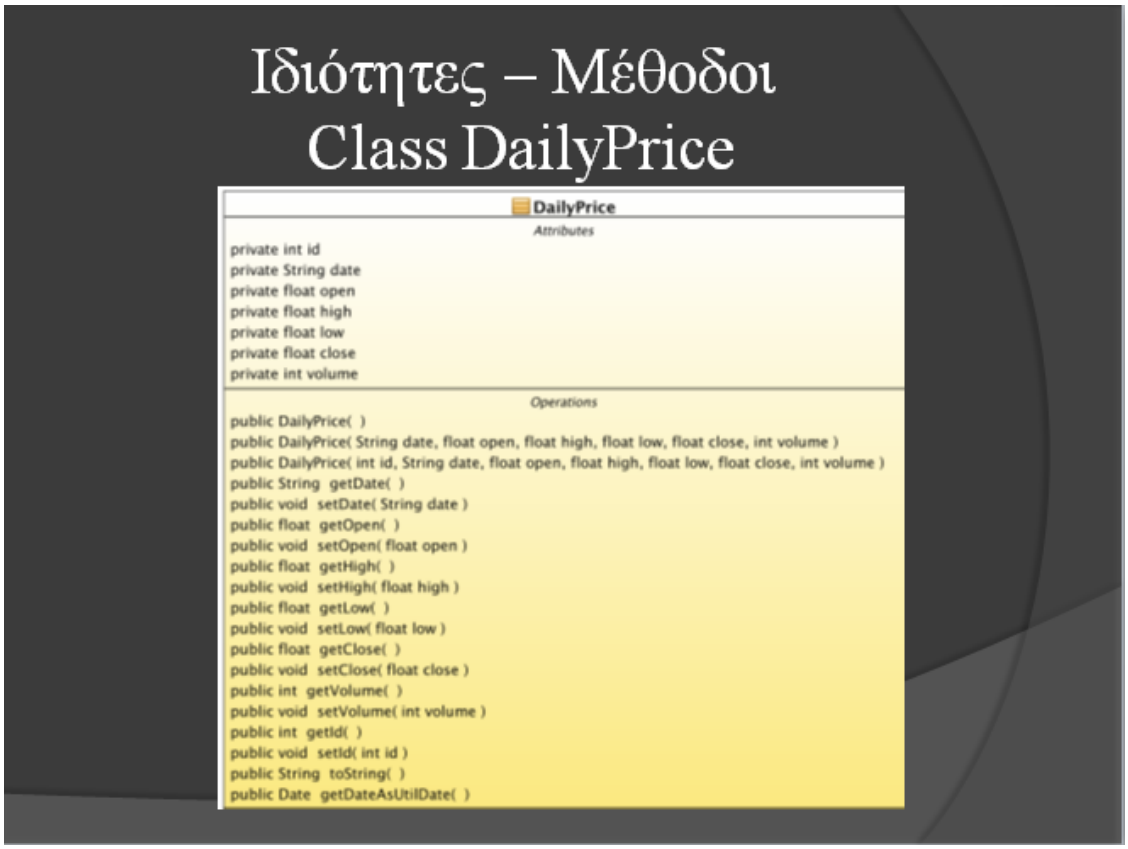
Attributes

private String url
private String db
private String driver
private Connection con

Operations

public void Initialize( )
public void CleanUp( )
private Stock[0..*] getAllStocks( )
public boolean insertOrUpdateStock(Stock stock, Sector sector)
public Stock[0..*] retrieveStocksForSector(Sector sector)
public Stock[0..*] getStocksForSelectedSector(String sectorname)
public boolean DeleteStock(Stock stock, Sector sector)
public DailyPrice[0..*] retrieveDailyPricesForStock(Stock stock)
public void insertOrUpdateDailyPrices(DailyPrice dps[0..*], Stock stock)
public boolean insertOrUpdateDailyPrice(DailyPrice dp, Stock stock)
public boolean DeleteDailyPrice(DailyPrice dp)
public Sector[0..*] getSectors( )
public Sector getSectorByName(String sectorname)
public boolean InsertOrUpdateSector(Sector sector)
public boolean DeleteSector(Sector sector)
public void retrieveAllSectors( )


Ιδιότητες – Μέθοδοι
Class Sector, Stock
Sector
Attributes
private int id
private String name
Operations
public Sector(int id, String name)
public Sector(String name)
public Sector(String name, Stock stocks[0..*])
public int getId( )
public void setId(int id)
public String getName( )
public void setName(String name)
public Stock[0..*] getStocks( )
public void setStocks(Stock stocks[0..*])
Stock
Attributes
private int id
private String name
Operations
public Stock(int id, String name)
public Stock(String name)
public Stock(String name, DailyPrice dailyPrices[0..*])
public String getName( )
public void setName(String name)
public DailyPrice[0..*] getDailyPrices( )
public void setDailyPrices(DailyPrice dailyPrices[0..*])
public int getId( )
public void setId(int id)

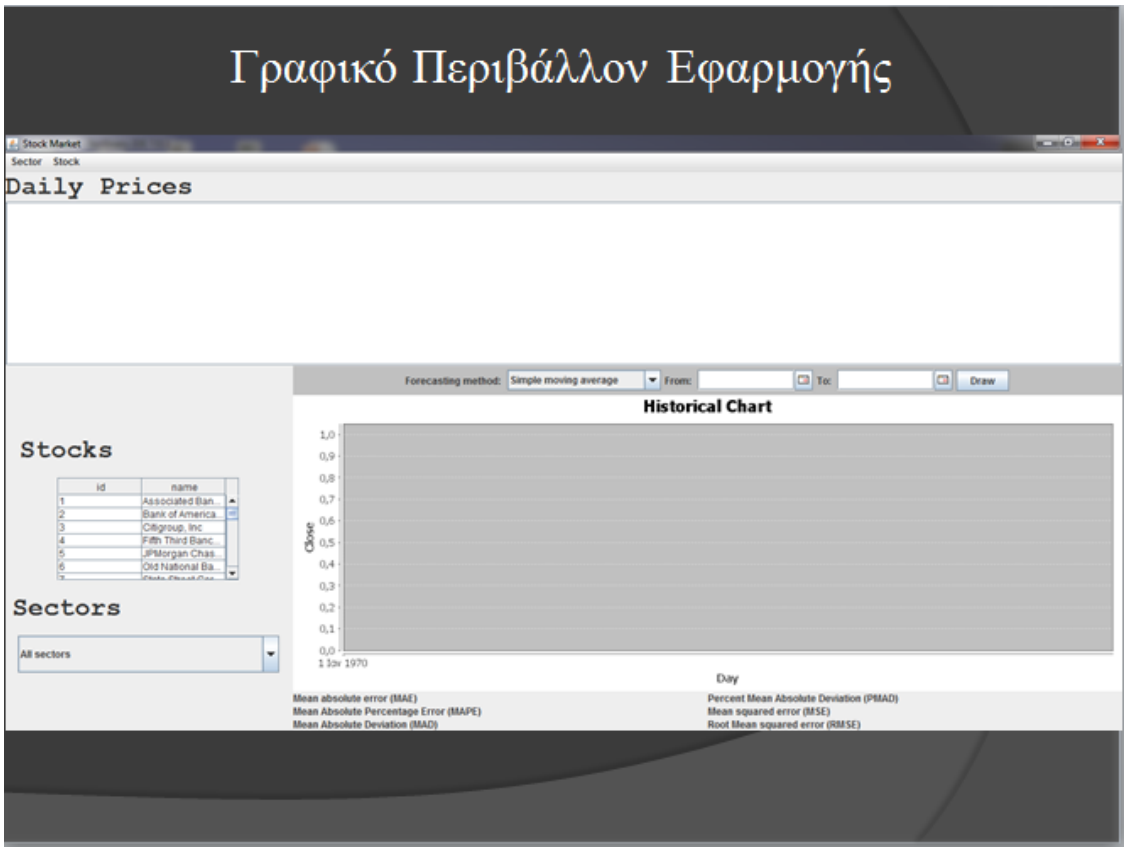

Ιδιότητες – Μέθοδοι
Class DailyPrice
DailyPrice
Attributes
private int id
private String date
private float open
private float high
private float low
private float close
private int volume
Operations
public DailyPrice( )
public DailyPrice(String date, float open, float high, float low, float close, int volume)
public DailyPrice(int id, String date, float open, float high, float low, float close, int volume)
public String getDate( )
public void setDate(String date)
public float getOpen( )
public void setOpen(float open)
public float getHigh( )
public void setHigh(float high)
public float getLow( )
public void setLow(float low)
public float getClose( )
public void setClose(float close)
public int getVolume( )
public void setVolume(int volume)
public int getId( )
public void setId(int id)
public String toString( )
public Date getDateAsUtilDate( )

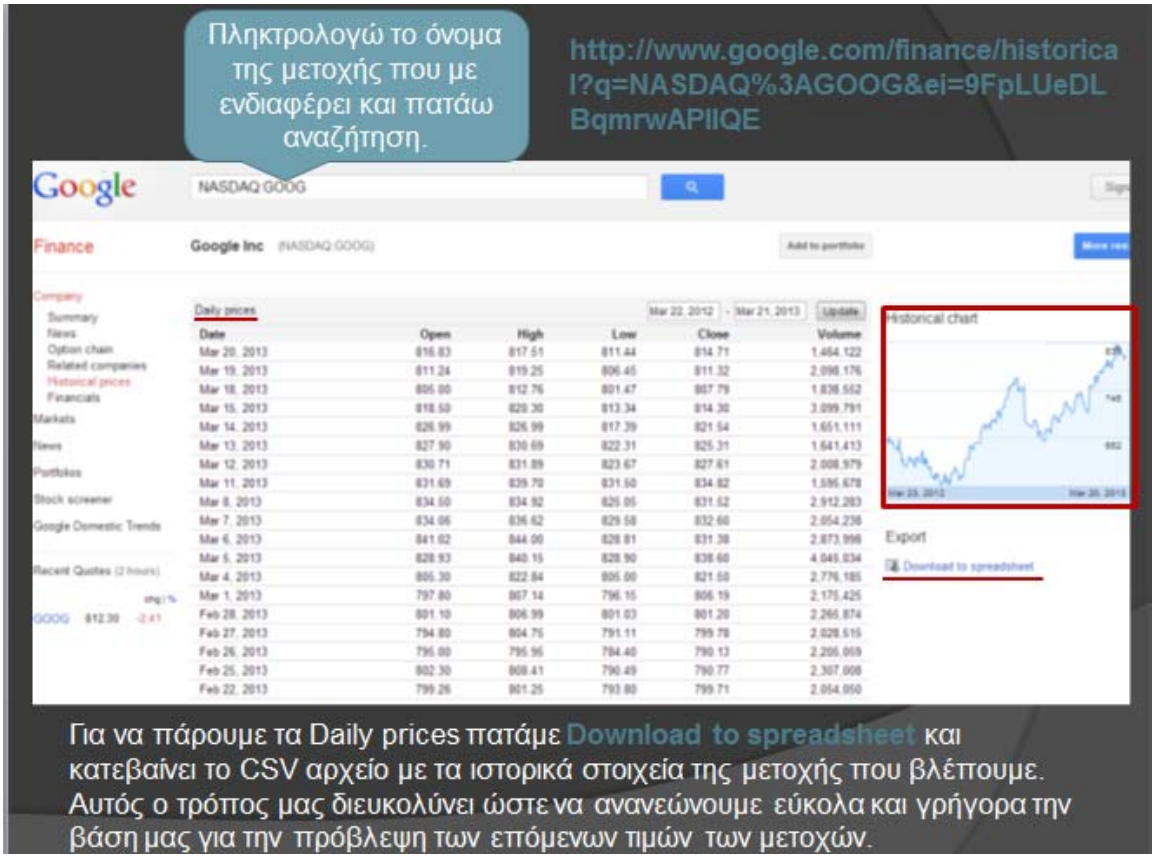
```
public class DBManager
{
    private static String url;
    private static String db;
    private static String driver;
    private static Connection con;
    private static ArrayList<Sector> sectors;

    public static void Initialize() {
        url = "jdbc:mysql://localhost:3306/";
        db = "stockmarket";
        driver = "com.mysql.jdbc.Driver";
        con = null;

        try{
            Class.forName(driver);
            con = DriverManager.getConnection(url+db,"root","epp");
        }
        catch(ClassNotFoundException | SQLException e)
        {
            System.out.println("Unable to connect to db. Error: " + e.getMessage());
        }
        sectors = new ArrayList<>();
        retrieveAllSectors();
    }

    public static void CleanUp() {
        try {
            con.close();
        } catch (SQLException ex) {
            Logger.getLogger(DBManager.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
```

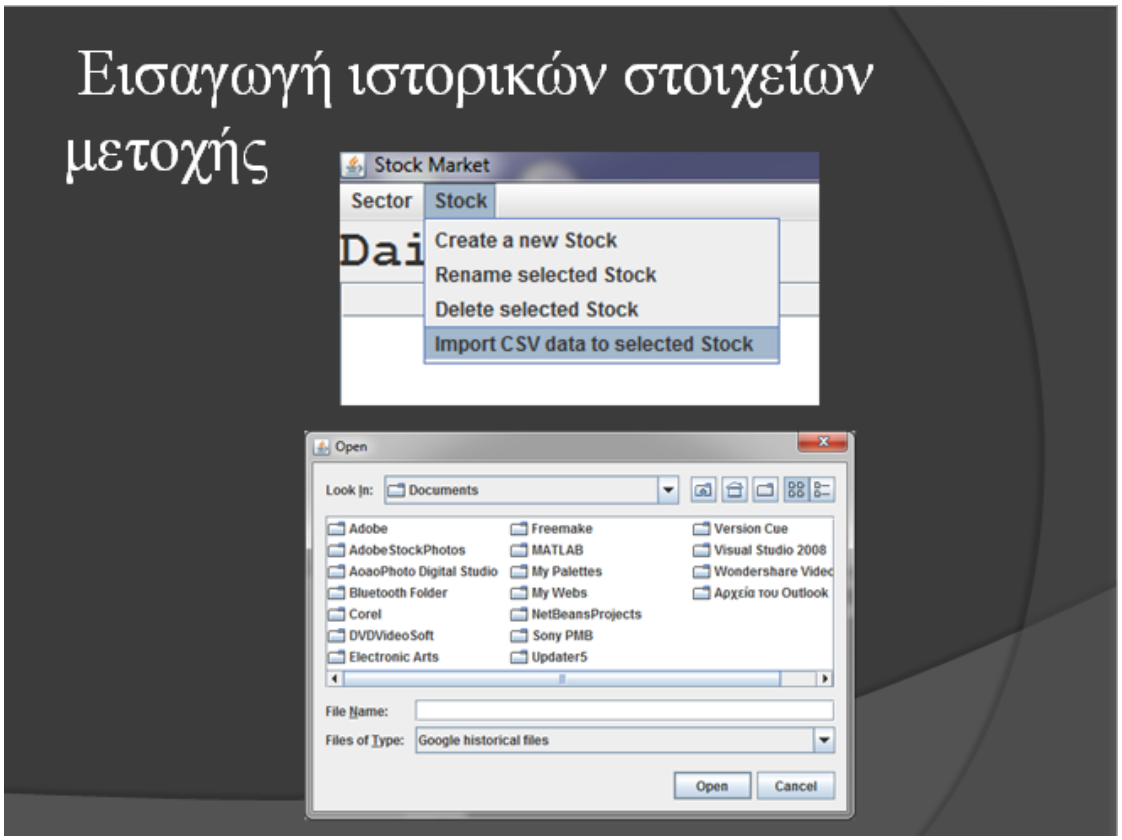
Απαιτήσεις

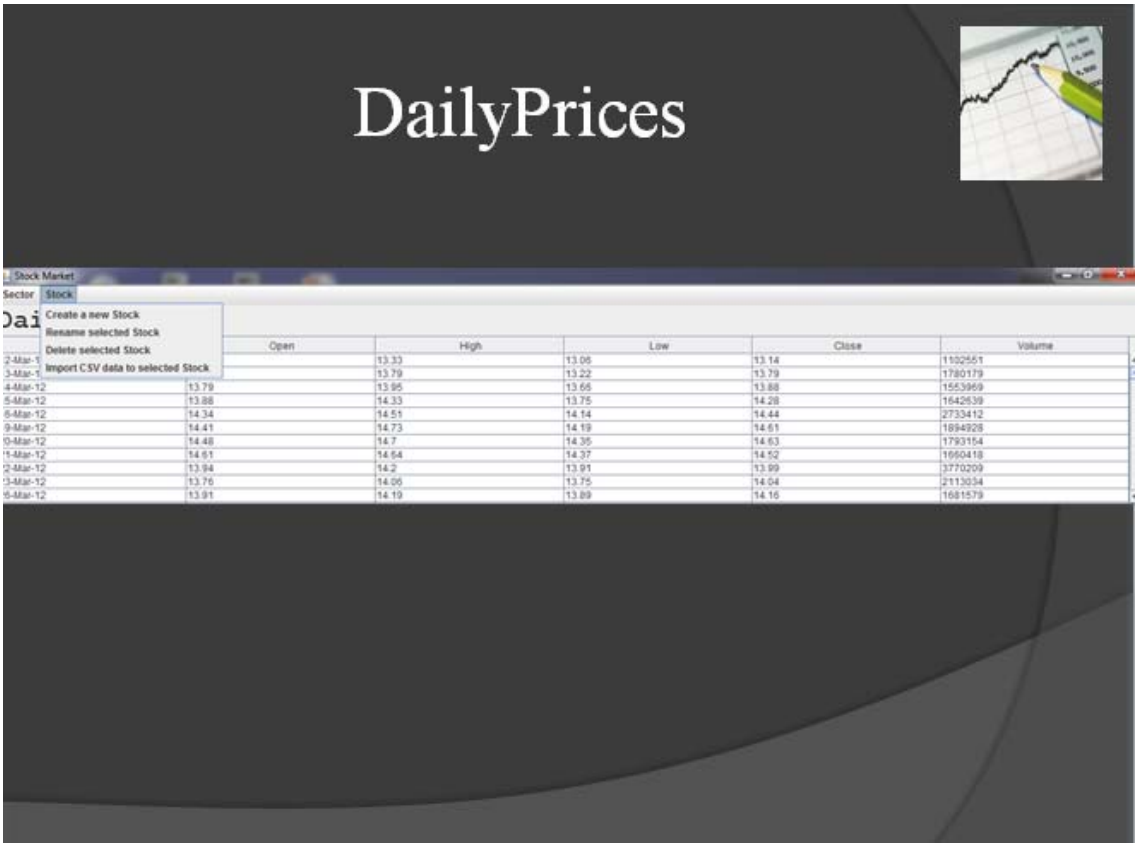
Ο υπολογιστής στον οποίο θα τρέχει η εφαρμογή πρέπει να έχει εγκατεστημένη Java και MySQL.

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


Γραφικό Περιβάλλον Εφαρμογής
Stock Market
Sector Stock
Daily Prices
Stocks
id
name
Sectors
All sectors
Forecasting method:
Simple moving average
From:
To:
Draw
Historical Chart
Close
Day
Mean absolute error (MAE)
Mean Absolute Percentage Error (MAPE)
Mean Absolute Deviation (MAD)
Percent Mean Absolute Deviation (PMAD)
Mean squared error (MSE)
Root Mean squared error (RMSE)

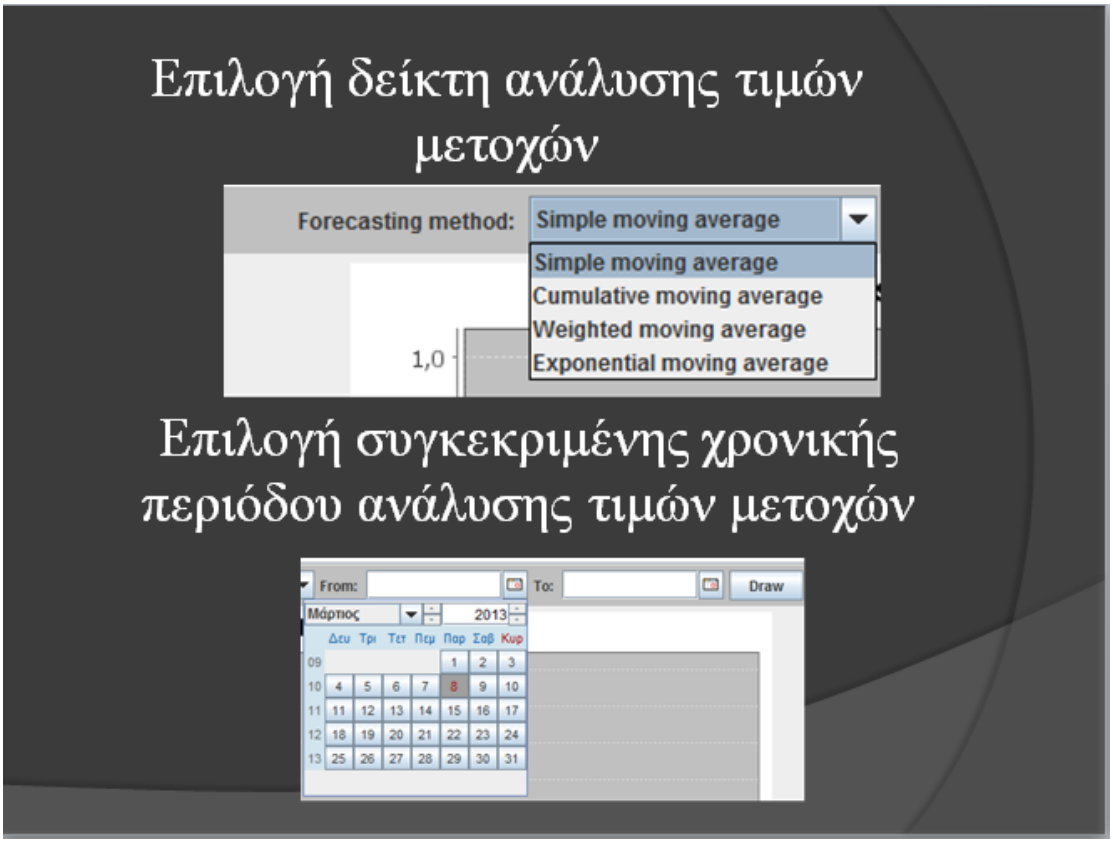

Πληκτρολογώ το όνομα της μετοχής που με ενδιαφέρει και πατάω αναζήτηση.
http://www.google.com/finance/historical?q=NASDAQ%3AGOOG&ei=9FpLUeDLBqmrwAPIQE
Google
NASDAQ:GOOG
Finance
Google Inc (NASDAQ:GOOG)
Daily prices
Historical chart
Export
Download to spreadsheet
Για να πάρουμε τα Daily prices πατάμε Download to spreadsheet και κατεβαίνει το CSV αρχείο με τα ιστορικά στοιχεία της μετοχής που βλέπουμε. Αυτός ο τρόπος μας διευκολύνει ώστε να ανανεώνουμε εύκολα και γρήγορα την βάση μας για την πρόβλεψη των επόμενων τιμών των μετοχών.


Εισαγωγή ιστορικών στοιχείων μετοχής
Stock Market
Sector
Stock
Create a new Stock
Rename selected Stock
Delete selected Stock
Import CSV data to selected Stock
Open
Look In:
Documents
Adobe
Adobe StockPhotos
AoaoPhoto Digital Studio
Bluetooth Folder
Corel
DVDVideoSoft
Electronic Arts
Freemake
MATLAB
My Palettes
My Webs
NetBeansProjects
Sony PMB
Updater5
Version Cue
Visual Studio 2008
Wondershare Videc
Αρχεία του Outlook
File Name:
Files of Type:
Google historical files
Open
Cancel


DailyPrices
Stock Market
Sector
Stock
Create a new Stock
Rename selected Stock
Delete selected Stock
Import CSV data to selected Stock
Open
High
Low
Close
Volume

\


Stocks - Sectors
Stocks
id
name
1
Associated Ban...
2
Bank of America...
3
Citigroup, Inc
4
Fifth Third Banc...
5
JPMorgan Chas...
6
Old National Ba...
Sectors
All sectors

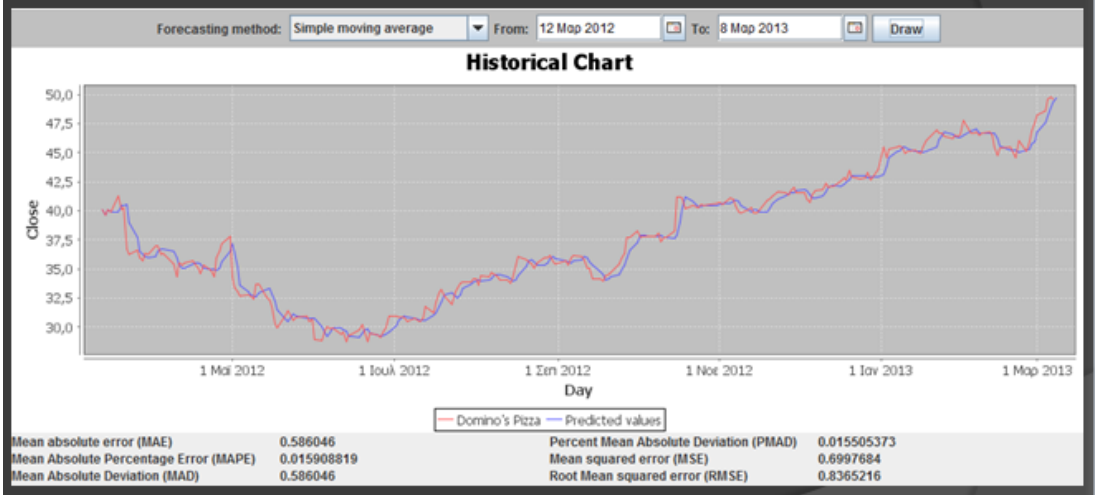

Επιλογή δείκτη ανάλυσης τιμών μετοχών
Forecasting method:
Simple moving average
Simple moving average
Cumulative moving average
Weighted moving average
Exponential moving average
1,0
Επιλογή συγκεκριμένης χρονικής περιόδου ανάλυσης τιμών μετοχών
From:
To:
Draw
Μάρτιος
2013
Δευ Τρι Τετ Πεμ Παρ Σαβ Κυρ

FORECASTING METHOD

Τεχνική ανάλυση τιμών μετοχών, με τον όρο τεχνική ανάλυση εννοείται η συστηματική μελέτη και επεξεργασία διαγραμμάτων, με στόχο την όσο δυνατή ασφαλέστερη και ακριβέστερη πρόβλεψη των τιμών των μετοχών.

Η τεχνική αυτή ανάλυση ουσιαστικά είναι ένας μαθηματικός υπολογισμός σχετικός με την τιμή της μετοχής που χρησιμοποιείται στην προσπάθεια να εκτιμηθούν μελλοντικές αλλαγές στην τιμή της μετοχής.

Historical Chart – Simple moving average

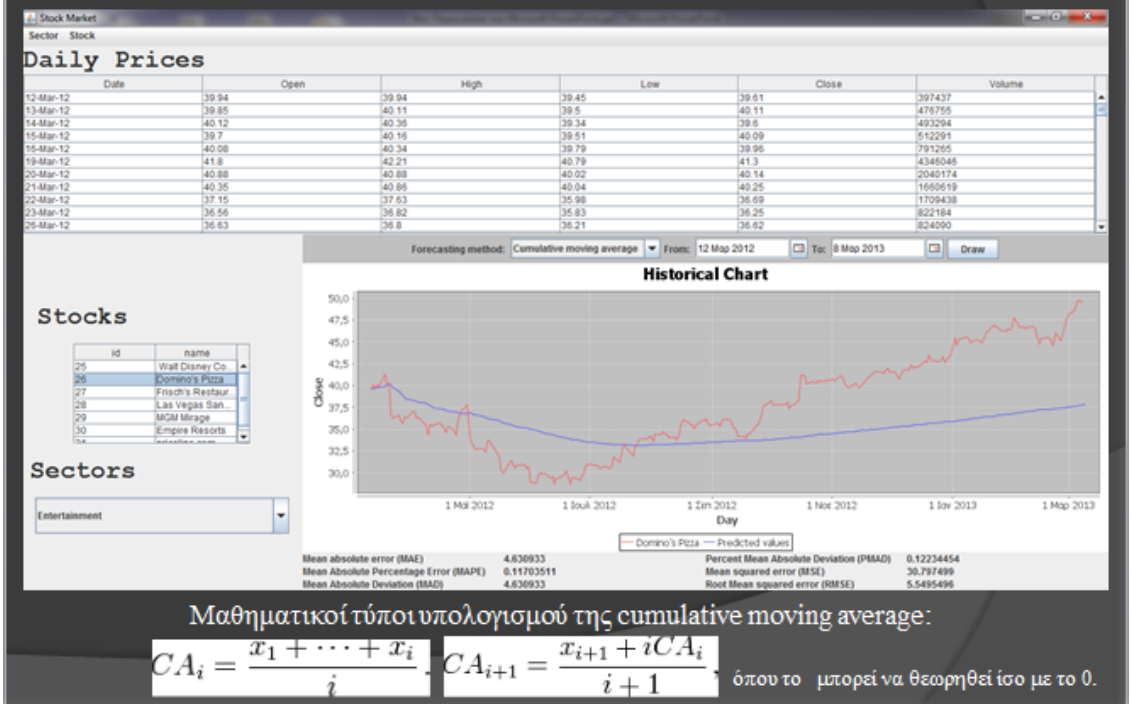


Μαθηματικοί τύποι υπολογισμού της simple moving average:

$$SMA = \frac{p_M + p_{M-1} + \cdots + p_{M-(n-1)}}{n}$$

$$SMA_{\text{today}} = SMA_{\text{yesterday}} - \frac{p_{M-n}}{n} + \frac{p_M}{n}$$

Cumulative moving average

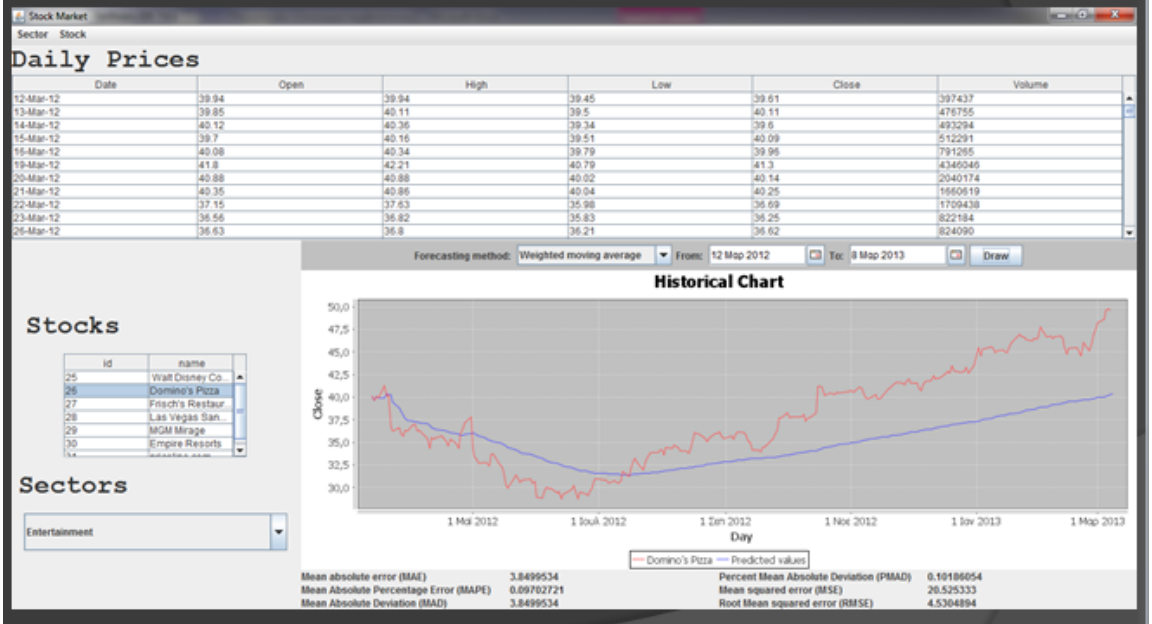


Μαθηματικοί τύποι υπολογισμού της cumulative moving average:

$$CA_i = \frac{x_1 + \cdots + x_i}{i}, \quad CA_{i+1} = \frac{x_{i+1} + iCA_i}{i+1}$$

όπου το μπορεί να θεωρηθεί ίσο με το 0.

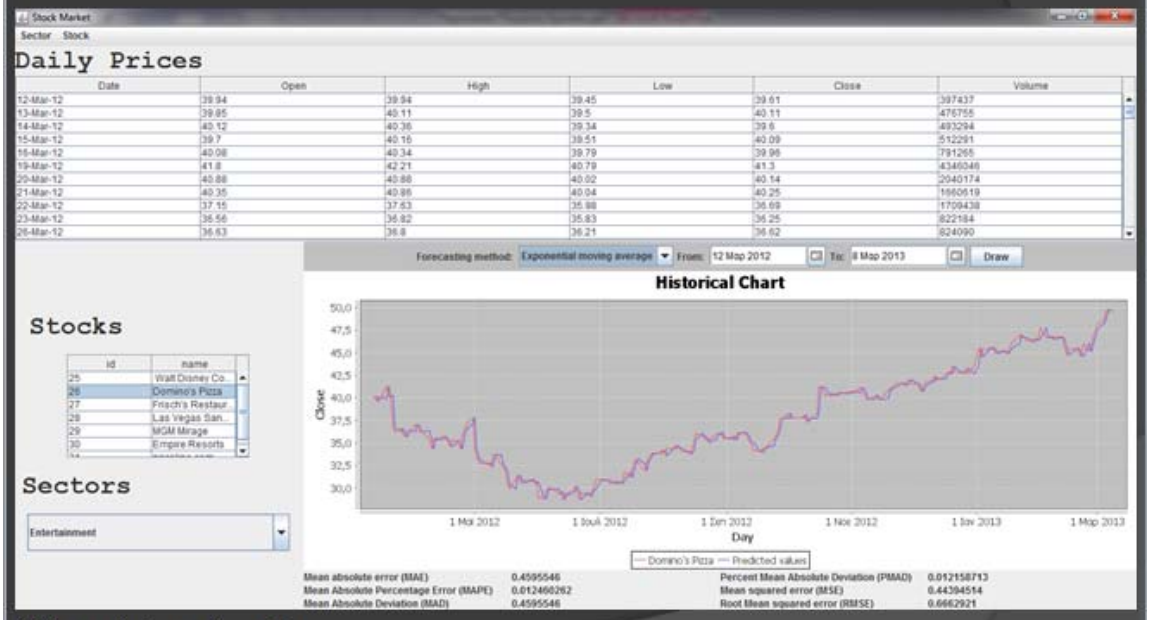
Weighted moving average



Μαθηματικός τύπος υπολογισμού της weighted moving average:

$$\mathrm{WMA}_M = \frac{np_M + (n-1)p_{M-1} + \cdots + 2p_{(M-n+2)} + p_{(M-n+1)}}{n + (n-1) + \cdots + 2 + 1}$$

Exponential moving average



Μαθηματικός τύπος υπολογισμού της exponential moving average:

$$\mathrm{EMA}_{\mathrm{today}} = \mathrm{EMA}_{\mathrm{yesterday}} + \alpha \times (\mathrm{price}_{\mathrm{today}} - \mathrm{EMA}_{\mathrm{yesterday}})$$

Ποσοστά αποτυχίας

Mean absolute error (MAE)	0.586046	Percent Mean Absolute Deviation (PMAD)	0.015505373
Mean Absolute Percentage Error (MAPE)	0.015908819	Mean squared error (MSE)	0.6997684
Mean Absolute Deviation (MAD)	0.586046	Root Mean squared error (RMSE)	0.8365216

Το μέσο απόλυτο σφάλμα (Mean Absolute Error). Είναι το σφάλμα ως προς τον μέσο όρο των απολύτων τιμών της απόκλισης των πραγματικών τιμών από τις προβλεπόμενες τιμές. Το σφάλμα αυτό δίνεται από τον τύπο:

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|f_i - y_i| = \frac{1}{n}\sum_{i=1}^{n}|e_i|$$

Το μέσο απόλυτο ποσοστιαίο σφάλμα (Mean absolute percentage error) γνωστό και ως μέση απόλυτη ποσοστιαία απόκλιση (MAPE) είναι ένα μέτρο ακρίβειας μιας μεθόδου για την κατασκευή προσαρμοσμένων τιμών χρονοσειρών σε στατιστικά στοιχεία, ειδικά στην εκτίμηση των τάσεων. Εκφράζει συνήθως την ακρίβεια ως ποσοστό, και ορίζεται από τον τύπο:

$$\mathrm{M} = \frac{100\%}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right|$$

όπου At η πραγματική τιμή και Ft η εκτιμώμενη τιμή.

Ποσοστά Αποτυχίας

Η μέση απόλυτη απόκλιση (Mean Absolute Deviation, MAD) είναι ένα ισχυρό μέτρο της μεταβλητότητας ενός δείγματος ποσοτικών δεδομένων. Για ένα σύνολο δεδομένων X1, X2, ..., Xn, η MAD ορίζεται ως η διάμεση τιμή των απόλυτων αποκλίσεων από το μέσο των δεδομένων:

$$\mathrm{MAD} = \mathrm{median}_i\left(\ |X_i - \mathrm{median}_j(X_j)|\ \right)$$

Το μέσο τετραγωνικό σφάλμα (Mean Squared Error, MSE). Είναι το σφάλμα ως προς την μέση τιμή της τετραγωνικής απόκλισης των πραγματικών τιμών από τις προβλεπόμενες τιμές. Το σφάλμα αυτό δίνεται από τον τύπο:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2$$

Η μέση απόλυτη ποσοστιαία απόκλιση (Percent Absolute Deviation, PMAD) δίνεται από τον τύπο:

$$PMAD = \frac{\sum_{t=1}^{N}|E_t|}{\sum_{t=1}^{N}|Y_t|}$$

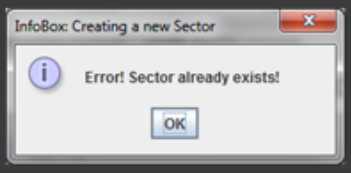
Ρίζα μέσου τετραγωνικού σφάλματος (Root Mean squared error, RMSE) δίνεται από τον τύπο:

$$RMSE = \sqrt{\frac{\sum_{t=1}^{N}E_t^2}{N}}$$

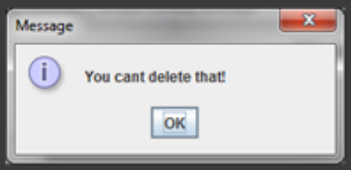
Πηγή:
[Wikipedia]http://en.wikipedia.org/wiki/Moving_average

Sectors

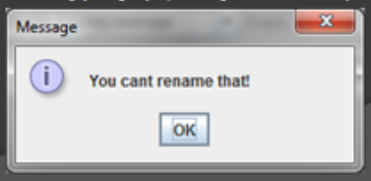
Αν εισάγω όνομα Sector που υπάρχει ήδη εμφανίζεται το παρακάτω μήνυμα.



Πρέπει πρώτα να επιλέξουμε την κατηγορία που θέλουμε να διαγράψουμε γιατί αλλιώς μας εμφανίζεται το παρακάτω μήνυμα.

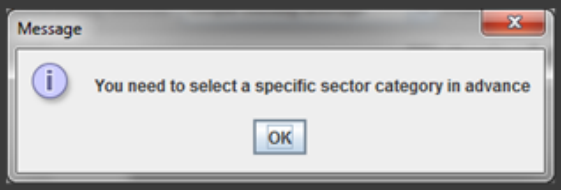


Πρέπει πρώτα να επιλέξουμε την κατηγορία που θέλουμε να μετονομάσουμε γιατί αλλιώς μας εμφανίζεται το παρακάτω μήνυμα.

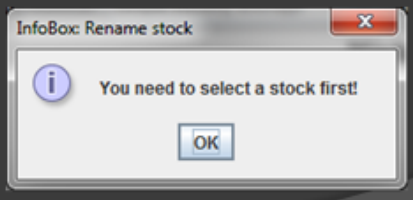


Stocks

Αν δεν επιλέξουμε πρώτα την κατηγορία που ανήκει η μετοχή και κάνουμε την δημιουργία νέας μετοχής εμφανίζεται το παρακάτω μήνυμα λάθους.

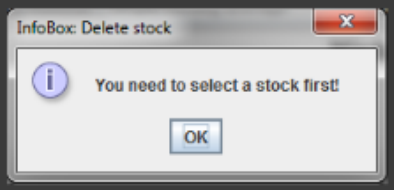


Για να μετονομάσουμε μια μετοχή πρέπει αρχικά να επιλέξουμε τη συγκεκριμένη μετοχή. Αν δεν την επιλέξουμε εμφανίζεται το παρακάτω μήνυμα.

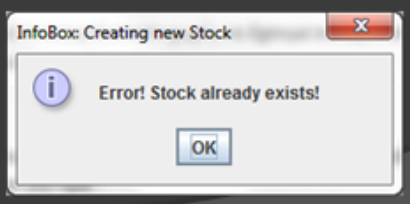


Stocks

Για να διαγράψουμε μια μετοχή πρέπει αρχικά να επιλέξουμε τη συγκεκριμένη μετοχή. Αν δεν την επιλέξουμε εμφανίζεται το παρακάτω μήνυμα.



Αν εισάγουμε ένα όνομα μετοχής που υπάρχει ήδη στον πίνακα μετοχών μας , εμφανίζεται το ακόλουθο μήνυμα



ΕΥΧΑΡΙΣΤΟΥΜΕ

Ιωάννα Φιλιππάκη (ΑΜ: 2700)
Αργυρώ Σαρικάκη (ΑΜ: 2660)

Επιβλέπων καθηγητής: Νίκος Παπαδάκης
Επιτροπή Αξιολόγησης: Νίκος Παπαδάκης
Χαράλαμπος Μανιφάβας
Ηλίας Χατζάκης