ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πτυχιακή Εργασία

«Δημιουργία και χρήση Βάσεων Δεδομένων στην Java – Υλοποίηση με το εργαλείο Netbeans»

Του φοιτητή
Enzelberger August
Αρ. Μητρώου: 03/2166

Επιβλέπων καθηγητής
Δρ. Σφέτσος Παναγιώτης

Θεσσαλονίκη 2010

Περίληψη

Η εργασία αυτή αφορά την γλώσσα προγραμματισμού Java και συγκεκριμένα της χρήσης του JDBC για την διαχείριση βάσεων δεδομένων, και της Swing για τον σχεδιασμό ενός γραφικού περιβάλλοντος. Για την ανάπτυξη του λογισμικού χρησιμοποιήθηκε το εργαλείο προγραμματισμού Netbeans, του οποίου η χρήση παρουσιάζεται όσον αφορά το πώς δημιουργείται μια εφαρμογή C.R.U.D. (Create, Read, Update, Delete) που χρησιμοποιεί το JDBC API, χρησιμοποιώντας κάποιες από τις λειτουργίες που παρέχει, ειδικά στην σχεδίαση ενός γραφικού περιβάλλοντος.

Abstract

The present work is about the Java programming language, in particular about the use of JDBC for database management, and Swing for designing graphical user interfaces. Netbeans, a programming tool, was used for the development of the software, and it’s use is presented concerning the creation of a C.R.U.D. (Create, Read, Update, Delete) application with the JDBC API, using some of its facilities, especially it’s graphical user interface design environment.

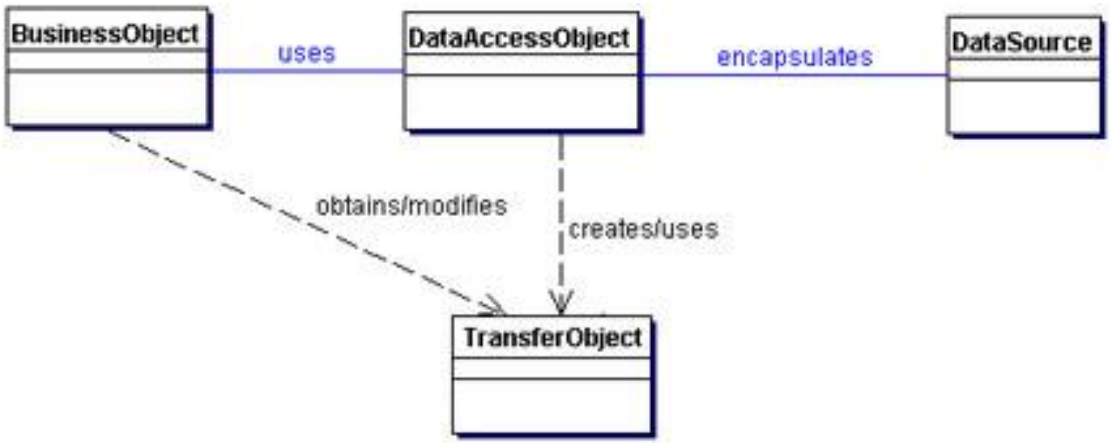
Περιεχόμενα

Εισαγωγή

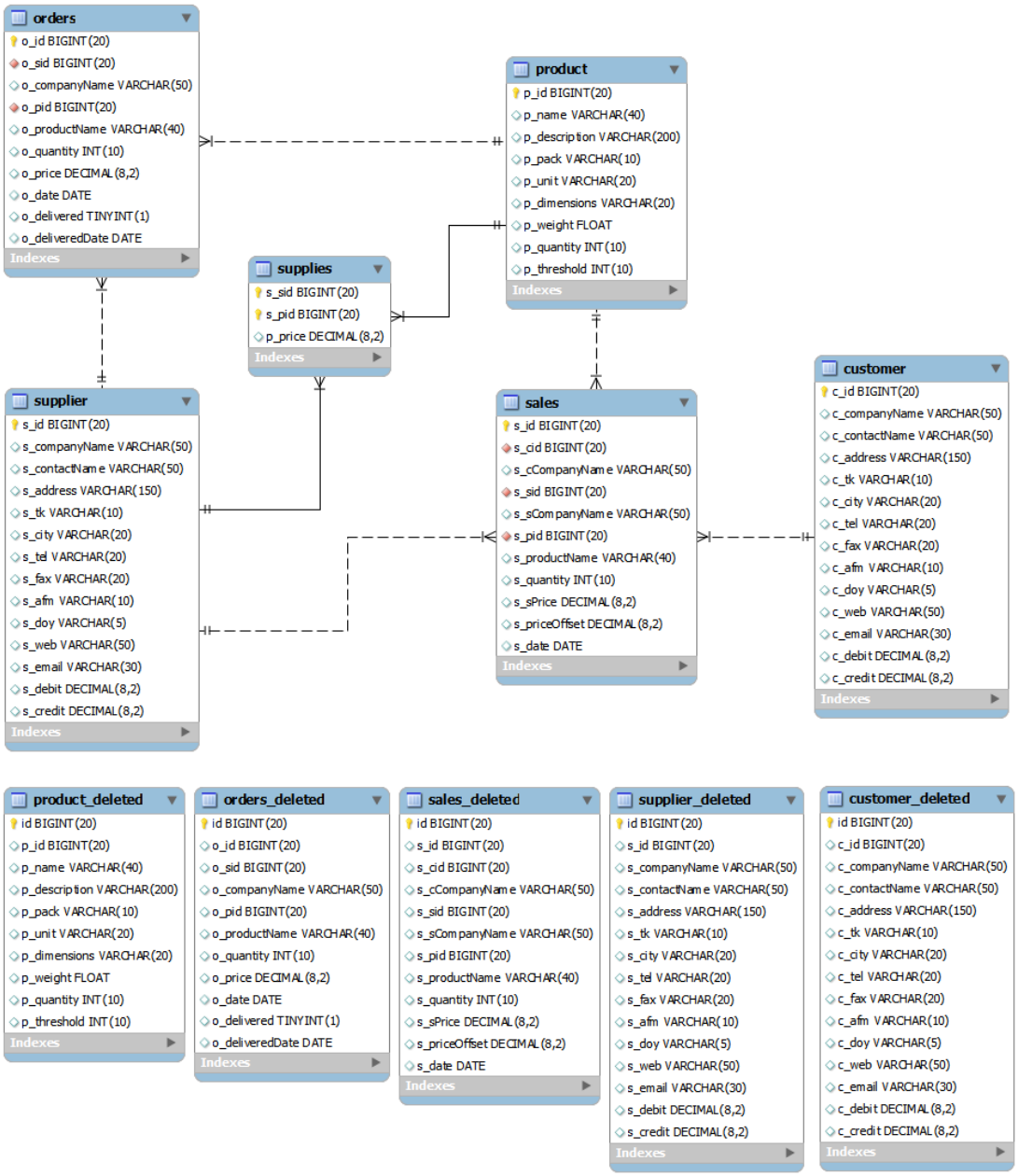
Για την εργασία αυτή δημιουργήθηκαν δύο εφαρμογές. Η πρώτη, με το όνομα Apothiki, ανήκει στην κατηγορία εφαρμογών W.M.S. (Warehouse Management System), είναι δηλαδή μια εφαρμογή διαχείρισης των δεδομένων μιας αποθήκης, ενώ η δεύτερη είναι μια εφαρμογή που δημιουργεί δυναμικά ένα γραφικό περιβάλλον CRUD, για την βάση δεδομένων με την οποία συνδέεται.

Η πρώτη εφαρμογή (Apothiki), βασίζεται σε ένα συγκεκριμένο σχήμα βάσης δεδομένων (Σχήμα 1-2) που μοντελοποιεί τα βασικά στοιχεία μιας αποθήκης, το οποίο και δημιουργεί. Περιλαμβάνονται στο σχήμα σχετικοί πίνακες για την αποθήκευση των πελατών, των προμηθευτών, των προϊόντων, των παραγγελιών και των πωλήσεων.

Κατά την φάση της σχεδίασης επιλέχθηκε το pattern DAO (Data Access Object). [8] Το pattern αυτό εξασφαλίζει την ανεξαρτησία του γραφικού περιβάλλοντος από τον κώδικα που απαιτείται για την επικοινωνία με την πηγή των δεδομένων. Κάποιες λεπτομέρειες του pattern προσαρμόστηκαν στις ανάγκες της εφαρμογής, κυρίως για την επίτευξη της ενσωμάτωσης συγκεκριμένης λογικής, την απλούστευση του παραγόμενου κώδικα και κατά συνέπεια την αύξηση της απόδοσης της εφαρμογής. Στο σχήμα 1-1 φαίνεται το διάγραμμα τάξεων με τις σχέσεις των αντικειμένων του pattern DAO.

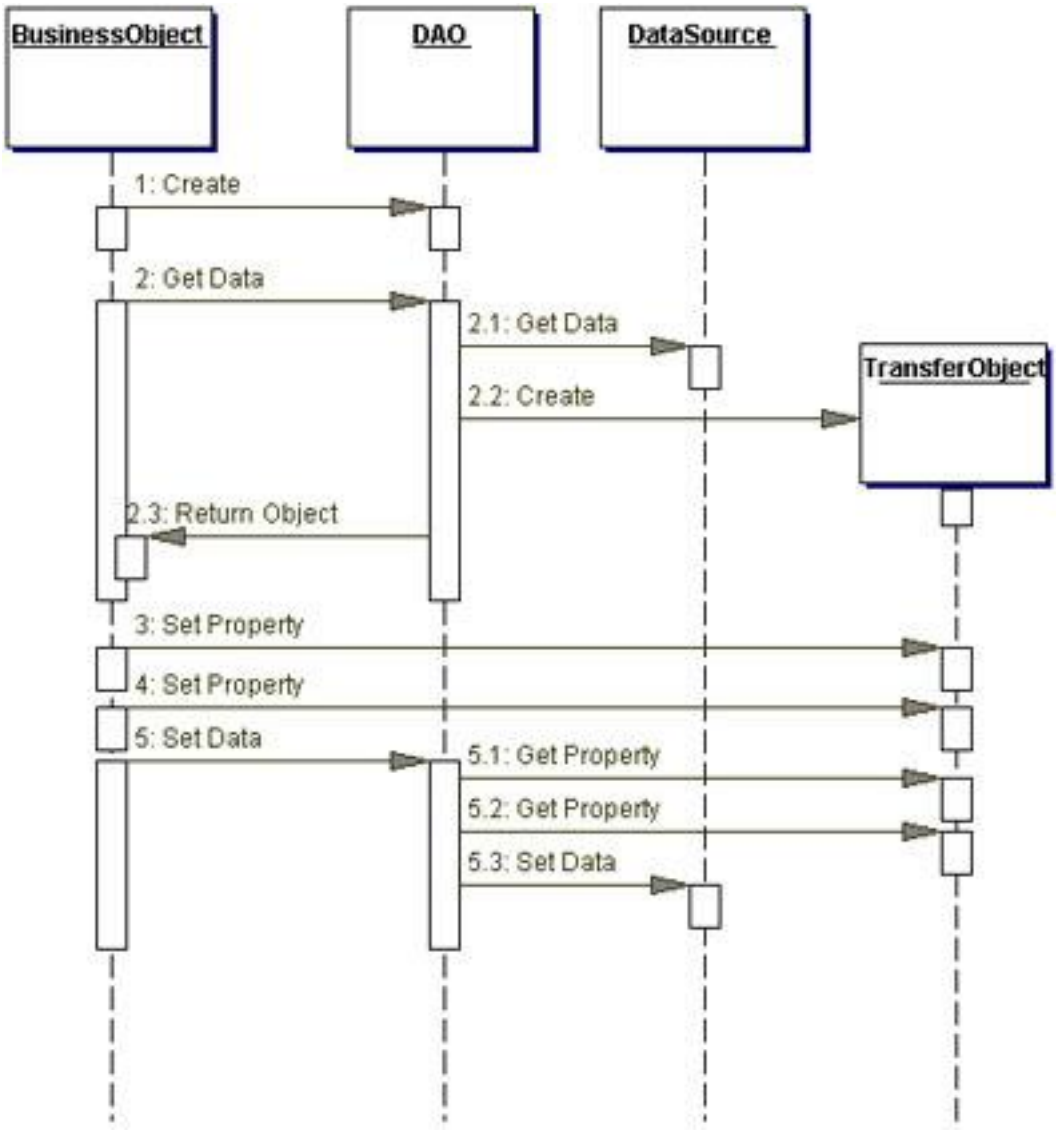


Σχήμα 1-1. Το pattern Data Access Object



Σχήμα 1-2: Διάγραμμα οντοτήτων-σχέσεων της εφαρμογής Apothiki.

Το αντικείμενο DAO είναι υπεύθυνο για την αποθήκευση και την ανάκτηση των δεδομένων. Σε αντίθεση με την προδιαγραφή, επιλέχθηκε να είναι υπεύθυνη η εφαρμογή για την διαχείριση της σύνδεσης με την πηγή δεδομένων (σύνδεση/αποσύνδεση) χρησιμοποιώντας αντικείμενα μιας τάξεως που αναπτύχθηκε ειδικά για την διαχείριση συνδέσεων JDBC. Η σύνδεση, αφού δημιουργηθεί επιτυχώς, παραδίδεται ως παράμετρος σε όλα τα αντικείμενα DAO. Στο σχήμα 1-3 φαίνεται το διάγραμμα ακολουθίας που δείχνει την αλληλεπίδραση μεταξύ των αντικειμένων του pattern DAO.



Σχήμα 1-3. Το διάγραμμα ακολουθίας του Data Access Object

BusinessObject

Το αντικείμενο BusinessObject αντιπροσωπεύει τον πελάτη δεδομένων. Είναι το αντικείμενο που απαιτεί την πρόσβαση στην πηγή δεδομένων για να ανακτήσει και να αποθηκεύσει δεδομένα.

DataAccessObject

Το αντικείμενο DataAccessObject είναι το βασικό αντικείμενο αυτού του pattern. Αφαιρεί την υποκείμενη υλοποίηση της πρόσβασης στα δεδομένα για το αντικείμενο BusinessObject, ώστε να παρέχει διαφανή πρόσβαση στην πηγή δεδομένων. Το αντικείμενο BusinessObject αποστέλλει εντολές

ανάκτησης και αποθήκευσης δεδομένων στο αντικείμενο DataAccessObject.

DataSource

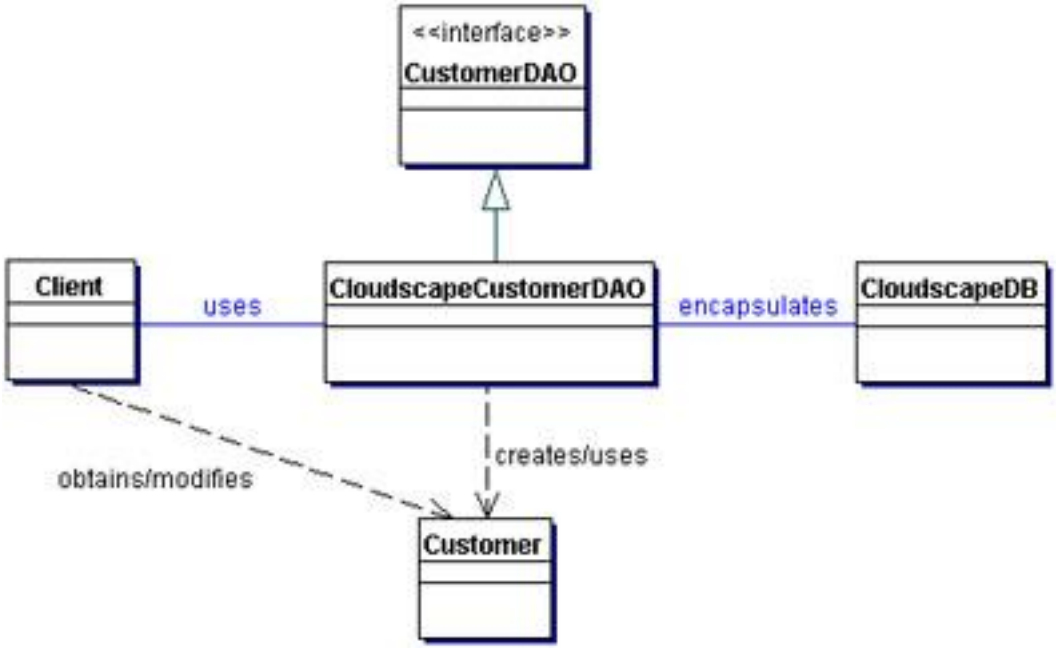
Το DataSource αντιπροσωπεύει την πηγή δεδομένων. Η πηγή δεδομένων μπορεί να είναι μια βάση δεδομένων όπως πχ ένα RDBMS, OODBMS κτλ, αλλά και κάποιο άλλο σύστημα (πχ. κεντρικός Η/Υ), υπηρεσία (πχ. γραφείο πιστωτικών καρτών), ή κάποιο άλλο είδος χώρου αποθήκευσης.

TransferObject

Το αντικείμενο TransferObject αντιπροσωπεύει ένα αντικείμενο μεταφοράς που χρησιμοποιείται ως φορέας δεδομένων. Το αντικείμενο DataAccessObject μπορεί να χρησιμοποιήσει ένα αντικείμενο μεταφοράς για να επιστρέψει δεδομένα στον πελάτη. Για την εφαρμογή επιλέχθηκε στο σημείο αυτό να χρησιμοποιηθεί το αντικείμενο ResultSet της Java για την ανάκτηση των δεδομένων, διότι τα αποτελέσματα των ερωτημάτων επιστρέφονται στην μορφή αυτή, αλλά και επειδή τα αντικείμενα στα οποία επιστρέφονται τα αποτελέσματα αυτά είναι αντικείμενα Jtable της Java, των οποίων τα μοντέλα δέχονται ως όρισμα αντικείμενα ResultSet.
Το αντικείμενο DataAccessObject μπορεί επίσης να λάβει δεδομένα από τον πελάτη με ένα αντικείμενο μεταφοράς, ώστε να καταχωρήσει δεδομένα στην πηγή δεδομένων. Η λειτουργία αυτή υλοποιήθηκε στην εφαρμογή με την χρήση αντικειμένων που αντιπροσωπεύουν εγγραφές των αντίστοιχων πινάκων της βάσης δεδομένων.

Υλοποίηση του pattern Data Access Object

Το σχήμα 1-4 δείχνει το διάγραμμα τάξεων μιας υλοποίησης του Data Access Object, όσον αφορά την διαχείριση των δεδομένων των πελατών μιας επιχείρησης.



Σχήμα 1-4. Διάγραμμα τάξεων μιας υλοποίησης του Data Access Object

Συγκεκριμένα η εφαρμογή Apothiki για την διαχείριση των πελατών (και όλων των υπόλοιπων οντοτήτων παρομοίως) περιλαμβάνει μια τάξη Customer για την δημιουργία αντικειμένων μεταφοράς (Transfer Object), ένα Interface CustomerDAO και μια τάξη MySQLCustomerDAO που το υλοποιεί για την βάση δεδομένων MySQL.

Παράδειγμα 1.1

Customer.java

```
public class Customer {
    private String cId;
    private String cCompanyName;
    private String cContactName;
    private String cAddress;
    private String cTk;
    private String cCity;
    private String cTel;
    private String cFax;
    private String cAfm;
```

```
    private String cDoy;
    private String cWeb;
    private String cEmail;
    private String cDebit;
    private String cCredit;
...
...
...
```

CustomerDAO.java

```
public interface CustomerDAO {
    // Interface that all CustomerDAOs must support
    public SQLException insertCustomer(Customer c, MyDBConnection conn);
    public SQLException deleteCustomer(Customer c, MyDBConnection conn);
    public ResultSet findCustomer(Customer c, MyDBConnection conn);
    public SQLException updateCustomer(Customer c, MyDBConnection
conn);
    public ResultSet selectCustomersRS(MyDBConnection conn);
}
```

MySQLCustomerDAO.java

```
public class MySQLCustomerDAO implements CustomerDAO {


    public SQLException insertCustomer(Customer c, MyDBConnection conn)
{
        String SQL="INSERT INTO customer (";
        SQL+=" c_id, c_companyName, c_contactName, c_address, c_tk,
c_city, c_tel, c_fax, c_afm, c_doy, c_web, c_email, c_debit, c_credit) VALUES
('";
        SQL+= c.getCId()+"', '"+c.getCCompanyName()+"',
'"+c.getCContactName()+"', '"+c.getCAddress()+"', '"+c.getCTk()+"',
'"+c.getCCity()+"', '"+c.getCTel()+"', '"+c.getCFax()+"', '"+c.getCAfm()+"',
'"+c.getCDoy()+"', '"+c.getCWeb()+"', '"+c.getCEmail()+"', '"+c.getCDebit()+"',
'"+c.getCCredit()+"');";
```

```
        return new SQLExecution(SQL,conn).update();
    }
...
...
...
```

Το ίδιο σετ αρχείων έχει δημιουργηθεί για κάθε πίνακα της βάσης δεδομένων τον οποίο θα πρέπει να διαχειριστεί η εφαρμογή.

Τα πεδία της τάξης Customer (αλλά και των υπόλοιπων παρόμοιων τάξεων) είναι Strings, αφού το μόνο που κάνουν είναι να μεταφέρουν δεδομένα από την εφαρμογή στην βάση δεδομένων. Ούτως η άλλως όλες οι εντολές SQL αποστέλλονται ως Strings στην βάση δεδομένων, που σημαίνει πως οποιεσδήποτε αριθμητικές μεταβλητές θα έπρεπε σε κάθε περίπτωση να μετατραπούν σε Strings. Εάν χρειαστεί να γίνει κάποια αριθμητική πράξη με κάποια από τα πεδία, αυτά μπορούν να μετατραπούν από την εφαρμογή και έπειτα να εκτελεστεί η πράξη. Για όλα τα πεδία πρέπει να δημιουργηθούν μέθοδοι get/set, κάτι που με το Netbeans μπορεί να γίνει αυτόματα πολύ εύκολα (refactor->encapsulate fields).

Οι μέθοδοι του Interface CustomerDAO χωρίζονται σε δύο κατηγορίες: αυτές που επιστρέφουν ως αποτέλεσμα εγγραφές στην εφαρμογή και αυτές που επεξεργάζονται τα δεδομένα της βάσης δεδομένων και επιστρέφουν το αποτέλεσμα της επεξεργασίας. Για την πρώτη κατηγορία επιλέχθηκε να χρησιμοποιηθεί το Result Set της Java, στο οποίο αποθηκεύονται οι εγγραφές που επιστρέφονται από τα ερωτήματα SELECT, αφού χρησιμοποιείται απευθείας από τα Table Model που χρειάζονται τα JTable για την εμφάνιση των περιεχομένων των πινάκων στο γραφικό περιβάλλον. Για την δεύτερη κατηγορία επιλέχθηκε το SQLException που περιλαμβάνει μεταξύ άλλων τον κωδικό και το μήνυμα του σφάλματος σε περίπτωση που η εκτέλεση ήταν ανεπιτυχής, ή ισούται με null εάν η εκτέλεση ήταν επιτυχής.

Τέλος, η υλοποίηση του Interface περιλαμβάνει για κάθε μέθοδο τον απαραίτητο κώδικα για τον σχηματισμό των εντολών SQL που θα αποσταλούν στην βάση δεδομένων. Όποτε η εφαρμογή χρειάζεται επικοινωνία με την βάση δεδομένων, καλεί κάποια μέθοδο των υλοποιημένων

DAOs δίνοντας ως παράμετρο το αντικείμενο μεταφοράς και την σύνδεση.
Για την δεύτερη εφαρμογή, όπου όλα όσα αφορούν το σχήμα της βάσης δεδομένων είναι άγνωστα (όνομα της βάσης δεδομένων, αριθμός και ονομασία πινάκων, πεδία πινάκων κτλ), η ανεξαρτησία της διεπιφάνειας από τον κώδικα SQL εξασφαλίζεται από την τάξη MySQLGenerator. Η τάξη αυτή αναπτύχθηκε ώστε να παράγει αυτόματα τον απαιτούμενο κώδικα SQL για τις εντολές SELECT, INSERT, UPDATE, DELETE για όλους τους πίνακες οποιουδήποτε σχήματος βάσης δεδομένων.
Ο Constructor της δέχεται ως ορίσματα το όνομα της βάσης, το όνομα του πίνακα για τον οποίο θα παραχθεί κώδικας και μια απλά συνδεδεμένη λίστα με τα ονόματα των στηλών του πίνακα αυτού. Οι μέθοδοι δέχονται ως ορίσματα έναν πίνακα String με τις τιμές των πεδίων και την σύνδεση. Χρησιμοποιώντας τα παραπάνω στοιχεία ο κώδικας SQL παράγεται δυναμικά.

Παράδειγμα 1.2

SQLGenerator.java

```
public interface SQLGenerator {
   public SQLException insert(String[] values, MyDBConnection conn);
   public SQLException delete(String[] values, MyDBConnection conn, boolean makeCopy);
   public ResultSet find(String[] values, MyDBConnection conn);
   public SQLException update(String[] values, MyDBConnection conn);
   public ResultSet selectAllRS(MyDBConnection conn);
}
```

MySQLGenerator.java

```
public class MySQLGenerator implements SQLGenerator{

   String dbName;
   String tableName;
   LinkedList tableColNames;
   int colnum;
```

```
   public MySQLGenerator(String dbName, String tableName, LinkedList tableColNames) {
       this.dbName=dbName;
       this.tableName=tableName;
       this.tableColNames=tableColNames;
       this.colnum=tableColNames.size();
   }

   public SQLException insert(String[] values, MyDBConnection conn) {
       String SQL="INSERT INTO "+dbName+"."+tableName+"(";
       if(!values[0].isEmpty())SQL+=tableColNames.get(0)+",";
       for(int i=1; i<colnum; i++) {
           SQL+=tableColNames.get(i);
           if(i<colnum-1) SQL+=",";
       }
       SQL+=") VALUES ('";
       if(!values[0].isEmpty())SQL+=values[0]+"','";
       for(int i=1; i<colnum; i++) {
           SQL+=values[i];
           if(i<colnum-1) SQL+="','";
       }
       SQL+="');";
       return new SQLExecution(SQL,conn,view).update();
   }
...
...
```

Με την ίδια λογική κατασκευάστηκε μια παρόμοια τάξη, η τάξη TableModelGenerator. Η τάξη αυτή παράγει αυτόματα TableModel, τα οποία χρησιμοποιούνται από τους πίνακες JTable όπου εμφανίζονται οι εγγραφές των πινάκων της βάσης δεδομένων. Έτσι μπορεί να κατασκευαστεί δυναμικά κατά τον χρόνο εκτέλεσης ένα TableModel για οποιονδήποτε πίνακα, ώστε να μπορούν να παρουσιαστούν τα περιεχόμενά του στο γραφικό περιβάλλον.

Παράδειγμα 1.3

TableModelGenerator.java

```
public class TableModelGenerator extends AbstractTableModel{

    private int colnum;
    private int rownum;
    private String[] colNames;
    private  ArrayList<String[]> ResultSets;

    /** Creates a new instance of TableModelGenerator */
    public TableModelGenerator(LinkedList fields, ResultSet rs) {
        colnum=fields.size();
        colNames = new String[colnum];
        ResultSets=new ArrayList<String[]>();
        for(int i=0; i<colnum; i++)
                colNames[i]=fields.get(i).toString();

        try{
            String[] row = new String[colnum];
            while(rs.next()){
                for(int i=0; i<row.length; i++) {
                    row[i]=rs.getString(colNames[i]);
                }
                ResultSets.add(row);
                row = new String[colnum];
            }
        }
        catch(Exception e){
            System.out.println("Exception in TableModelGenerator");
        }
    }
...
```

```
...
...
```

Και οι δύο εφαρμογές αναπτύχθηκαν για την βάση δεδομένων MySQL, χρησιμοποιώντας το εργαλείο NetBeans. Το NetBeans παρέχει πολλές λειτουργίες στον προγραμματιστή, διευκολύνοντας και επιταχύνοντας σημαντικά το έργο του. Η σχεδίαση του γραφικού περιβάλλοντος, τα εργαλεία για την ενσωμάτωση της μετάφρασης της γλώσσας του γραφικού περιβάλλοντος σε άλλες ώστε να είναι πολύγλωσση, η αυτόματη παραγωγή κώδικα όπου αυτό είναι δυνατό (δήλωση, αρχικοποίηση και παραμετροποίηση Swing components, event listeners κτλ) είναι μόνο λίγες από τις ευκολίες που παρέχονται.
Οι δύο εφαρμογές που αναπτύχθηκαν έχουν τις εξής δυνατότητες:

Apothiki

- Διαχείριση πελατών, προμηθευτών, προϊόντων, παραγγελιών και πωλήσεων.
- Αυτόματη ενημέρωση credit/debit κατά την αποθήκευση παραγγελιών και πωλήσεων.
- Ενημέρωση σε περίπτωση που η ποσότητα ενός προϊόντος φτάσει στο κρίσιμο σημείο.
- Ανάθεση προμήθευσης ενός προϊόντος σε έναν ή και περισσότερους προμηθευτές (ο καθένας πιθανώς με διαφορετική τιμή).

MySQL Swing Interface Generator

- Δυναμική δημιουργία καρτέλας για κάθε πίνακα.
- Δυναμική προσθήκη label και textfield στην καρτέλα για κάθε πεδίο του πίνακα.
- Δυναμική προσθήκη κουμπιών New, Save, Update, Delete, Search, Refresh
- Δυναμική παραγωγή SQL κώδικα για την λειτουργία των παραπάνω κουμπιών (αρχείο MySQLGenerator.java)
- Καρτέλα για την εκτέλεση SQL εντολών.

Λειτουργίες που υποστηρίζονται και από τις δύο εφαρμογές

- Επιλογή γλώσσας (Ελληνικά/Αγγλικά).
- Καρτέλα στην οποία εμφανίζονται όλες οι εντολές SQL που αποστέλλονται στην βάση δεδομένων, μαζί με το αποτέλεσμά της εκτέλεσης.
- Σάρωση των διαθέσιμων „Look and Feel“ του συστήματος και δυναμική δημιουργία μενού για την επιλογή του επιθυμητού στυλ.
- Εξαγωγή των δεδομένων του πίνακα της επιλεγμένης καρτέλας σε αρχείο text, excel ή html.
- Αναζήτηση εγγραφών χρησιμοποιώντας οποιονδήποτε συνδυασμό πεδίων.
- Δημιουργία/διαγραφή βάσεων δεδομένων.
- Δημιουργία/διαγραφή χρηστών.
- Επεξεργασία δικαιωμάτων των χρηστών.

Κεφάλαιο 1: Η Java και το JDBC

Εισαγωγή

Η Java και το περιβάλλον της εισήχθησαν το 1995 από την Sun Microsystems, Inc. Σχεδιάστηκε έτσι ώστε να ανταπεξέλθει στις προκλήσεις της ανάπτυξης σε μη-συμβατά αλλά δικτυωμένα περιβάλλοντα. Είναι μια γλώσσα προγραμματισμού η οποία είναι ιδιαίτερα ευάρμοστη σε εφαρμογές Internet και intranet.

Η Standard Edition του περιβάλλοντος χρόνου εκτέλεσής της περιέχει μια ποικιλία APIs (Application Programming Interfaces – Διεπαφές Προγραμματισμού Εφαρμογών) για γραφικές διεπαφές χρήστη (GUIs), χειρισμό κατανεμημένων αντικειμένων, είσοδο/έξοδο, μαθηματικές ρουτίνες, δικτύωση, ασφάλεια και προσπέλαση βάσεων δεδομένων. Αυτή η ποικιλία σε APIs επιτρέπει στους προγραμματιστές να γράψουν ισχυρές εφαρμογές χωρίς να πρέπει να αγοράσουν και να μάθουν πολλές βιβλιοθήκες τρίτων. [1]

1.1 Γιατί Java

Η Java είναι ανεξάρτητη πλατφόρμας (platform independent) και αυτός είναι ένας από τους κύριους λόγους για τους οποίους έγινε διάσημη. Το ότι είναι ανεξάρτητη πλατφόρμας σημαίνει πως τα προγράμματα Java μπορούν να τρέξουν σε πολλούς διαφορετικούς τύπους υπολογιστών. Ένα πρόγραμμα Java τρέχει σε οποιονδήποτε υπολογιστή έχει εγκατεστημένο το Περιβάλλον Χρόνου Εκτέλεσης Java ή αλλιώς JRE (Java Runtime Environment). Υπάρχουν JRE για σχεδόν κάθε τύπο υπολογιστή, όπως υπολογιστές με οποιαδήποτε έκδοση των Windows, Linux ή Unix, υπολογιστές Macintosh, τεράστιους υπολογιστές μεγάλης ισχύος καθώς και κινητά τηλέφωνα.

Πριν την Java, άλλες γλώσσες προγραμματισμού, όπως η C και η C++, προσπάθησαν να είναι ανεξάρτητες πλατφόρμας παραδίδοντας μεταγλωττιστές (compilers) για διαφορετικές πλατφόρμες. Ο σκοπός ήταν να μπορεί κάποιος να μεταγλωττίσει το πρόγραμμά του σε διαφορετικές εκδόσεις, συμβατές με την κάθε πλατφόρμα, κάτι το οποίο δυστυχώς δεν λειτούργησε ποτέ καλά.

Η Java αντί να ακολουθήσει την ίδια τακτική, να βασιστεί δηλαδή σε πολλούς

διαφορετικούς μεταγλωττιστές για την κάθε πλατφόρμα, βασίστηκε στην Εικονική Μηχανή Java ή JVM (Java Virtual Machine). Η JVM είναι μια υποθετική πλατφόρμα, ένας υπολογιστής που ουσιαστικά δεν έχει κατασκευαστεί με υλικό. Είναι στην πραγματικότητα ένας προσομοιωτής (emulator) ο οποίος μπορεί να εκτελέσει προγράμματα Java.

Ο μεταγλωττιστής της Java δεν μεταγλωττίζει το πρόγραμμα στη γλώσσα μηχανής του υπολογιστή στον οποίο θα τρέξει. Αντί αυτού, το πρόγραμμα μεταγλωττίζεται στη γλώσσα μηχανής της JVM, το λεγόμενο bytecode. Τέλος, το JRE εκτελεί το bytecode στην JVM. Λόγω της JVM, ένα πρόγραμμα Java μπορεί να εκτελεστεί σε οποιονδήποτε υπολογιστή έχει εγκατεστημένο το JRE, χωρίς να πρέπει να μεταγλωττιστεί ξανά.

Αν και παρουσιάζονται κάποια προβλήματα όταν μεταφέρεται κώδικας Java από πλατφόρμα σε πλατφόρμα, είναι πολύ μικρά σε σύγκριση με την μεταφορά κώδικα C++ σε διαφορετικές πλατφόρμες όσον αφορά τον πολυνηματισμό, την δικτύωση, την διεπαφή χρήστη καθώς και τα APIs βάσεων δεδομένων. [2]

Η Java έχει ομοιότητες με την C++, κάτι που βοηθάει στην εκμάθηση της για κάποιον με λίγες γνώσεις C++, αλλά και σημαντικές διαφορές. Προσφέρονται διάφορα επιπλέον χαρακτηριστικά, όπως αυτόματη συλλογή απορριμμάτων από τον garbage collector ο οποίος αποδεσμεύει στιγμιότυπα αντικειμένων όταν αυτά δεν χρειάζονται πια, αναφορές αντικειμένων αντί για δείκτες αριθμητικών διευθύνσεων και τοπικός ή μη-τοπικός πολυνηματισμός.

Σίγουρα πρόκειται για μια αντικειμενοστραφή γλώσσα προγραμματισμού, καθώς υποστηρίζεται πλήρως η ενθυλάκωση, ο πολυμορφισμός, η κληρονομικότητα και οι δυναμικές δεσμεύσεις. Το πλεονέκτημα μιας τέτοιας προσέγγισης είναι πως το προγραμματιστικό μοντέλο έρχεται πιο κοντά στο πραγματικό μοντέλο, κάνοντας πιο εύκολη την υλοποίηση. Επίσης με τον τρόπο αυτό ενισχύεται και η επαναχρησιμοποίηση του λογισμικού. [1]

Οι προγραμματιστές Java κατασκευάζουν προγράμματα χρησιμοποιώντας αντικείμενα που αντιπροσωπεύουν αντικείμενα του πραγματικού κόσμου ή αφηρημένες δομές.

Όλα τα αντικείμενα έχουν δύο βασικά χαρακτηριστικά:

- Έχουν δεδομένα, ή αλλιώς κατάσταση (state). Για παράδειγμα μια ταινία έχει δεδομένα όπως τον τίτλο της ταινίας, τον συγγραφέα. τον

σκηνοθέτη κτλ.

- Έχουν συμπεριφορά, που σημαίνει πως μπορούν να εκπληρώσουν εργασίες χρησιμοποιώντας τις λεγόμενες μεθόδους. Για παράδειγμα ένα αυτοκίνητο μπορεί να έχει τις μεθόδους start, stop, drive κτλ. Μερικές μέθοδοι επιτρέπουν απλά την πρόσβαση στα δεδομένα του αντικειμένου, όπως πχ. μια ταινία μπορεί να έχει την μέθοδο getTitle με την οποία επιστρέφεται ο τίτλος της ταινίας.

Οι τάξεις είναι στενά συνδεδεμένες με τα αντικείμενα καθώς μια τάξη χρησιμοποιείται για την κατασκευή αντικειμένων. Η τάξη περιγράφει τα δεδομένα και τις μεθόδους που αποτελούν το αντικείμενο, καθορίζοντας έτσι την κατάσταση και την συμπεριφορά του.
Για παράδειγμα κατά την κατασκευή ενός προγράμματος εικονικής ταινιοθήκης θα πρέπει προφανώς να κατασκευαστούν αντικείμενα που αντιπροσωπεύουν τις πραγματικές ταινίες. Για την κατασκευή των αντικειμένων αυτών μπορεί να χρησιμοποιηθεί μια τάξη Ταινία, η οποία κατά την εκτέλεση του προγράμματος θα κατασκευάσει ένα αντικείμενο για κάθε ταινία της ταινιοθήκης. [2]

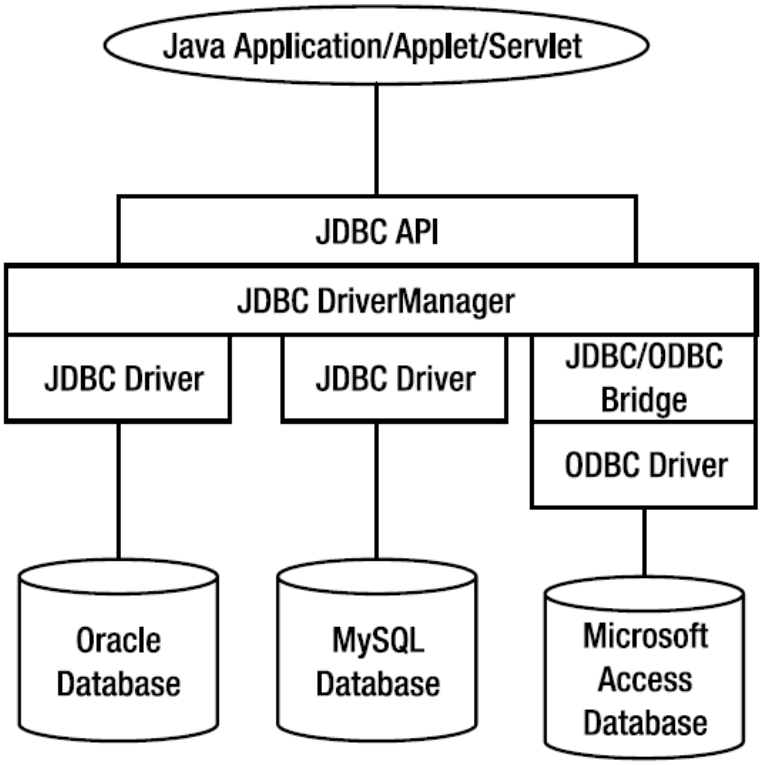
1.2 Τί είναι το JDBC;

Το JDBC (Java Database Connectivity) είναι ένα API βάσεων δεδομένων που αφαιρεί σχεδόν όλες τις πτυχές της διασύνδεσης προγραμμάτων με βάσεις δεδομένων και καθιστά εύκολη την εναλλαγή μεταξύ βάσεων δεδομένων όπως πχ από mySQL σε DB2 ή σε Oracle.
Υπάρχουν πολλά σημεία όπου μπορεί να χρησιμοποιηθεί το JDBC για την ενσωμάτωση βάσεων δεδομένων, όπως στην μεριά πελάτη (client side) , σε οποιαδήποτε σειρά εξυπηρετητών (server tier), ακόμα και σε smartcards και Palmtops, καθώς υπάρχουν πλέον “ελαφριές” βάσεις δεδομένων και για αυτές τις πλατφόρμες. Το JDBC είναι μια προδιαγραφή διεπαφής, σχεδιασμένο να είναι ανεξάρτητο πλατφόρμας και πραγματικά φορητό. Ανάλογα με το τι συμβαίνει και πως, από την μεριά της εφαρμογής, το σύνολο μπορεί να είναι εν μέρει ή πλήρως φορητό.
Το JDBC παρέχει όλα τα απαραίτητα για την ονομασία, σύνδεση, ερώτηση, διαχείριση συναλλαγών, αντιστοίχιση τύπων, metadata βάσεων δεδομένων,

δείκτες και διάφορες άλλες ευκολίες με τρόπο πλήρως ανεξάρτητο των βάσεων δεδομένων. Παρόλα αυτά υπάρχουν διάφορες πτυχές της ενσωμάτωσης με βάσης δεδομένων που δεν πρέπει να υποτιμηθούν. Δεν γίνεται πάντα να υπάρχει πλήρης έλεγχος του περιβάλλοντος στο οποίο θα τρέξουν οι εφαρμογές. Μερικές από αυτές τις πτυχές είναι:

- Η συμμόρφωση του οδηγού JDBC στο standard API
- Ο τύπος του οδηγού JDBC, αναλόγως εάν ο οδηγός είναι γραμμένος πλήρως σε Java ή χρησιμοποιεί τοπικές βιβλιοθήκες
- Το αν ο οδηγός JDBC είναι ασφαλής όσον αφορά τον πολυνηματισμό
- Η δυνατότητα του οδηγού JDBC να περάσει διάφορα πρωτόκολλα δικτύου από HTTP
- Το σύστημα ονομασίας βάσεων δεδομένων του οδηγού JDBC
- Οι περιορισμοί του οδηγού JDBC και της βάσης δεδομένων σε τύπους δεδομένων και μέγιστα μεγέθη τύπων
- Η στρατηγική κλειδώματος βάσεων δεδομένων [2]

Στο σχήμα 1.5 φαίνεται πώς ένα πρόγραμμα Java μπορεί μέσω του JDBC να προσπελάσει διάφορες βάσεις δεδομένων:



Σχήμα 1-5. Εφαρμογή βάσεων δεδομένων Java που χρησιμοποιεί το JDBC [3]

1.3 Τα συστατικά του JDBC

Στο σχήμα 1-1 μπορούμε να διακρίνουμε τα εξής συστατικά του JDBC:

- Την εφαρμογή
- Το JDBC API
- Τον driver manager
- Τον driver
- Την πηγή δεδομένων (data source)

Η εφαρμογή καλεί μεθόδους του JDBC για να αποστείλει εντολές SQL στην βάση δεδομένων και να ανακτήσει τα αποτελέσματα. Μπορεί να εκτελέσει τις παρακάτω λειτουργίες:

- Να ζητήσει την σύνδεση με μια πηγή δεδομένων
- Να αποστείλει εντολές SQL στην πηγή δεδομένων
- Να καθορίσει αποθηκευτικούς χώρους και τύπους δεδομένων για τα σετ αποτελεσμάτων (result sets)

- Να ζητήσει αποτελέσματα
- Να επεξεργαστεί σφάλματα
- Να ελέγχει τις δοσοληψίες ζητώντας την εκτέλεση των λειτουργιών commit και rollback
- Να κλείσει την σύνδεση

Το JDBC API περιέχει δύο μεγάλα σετ διεπαφών. Το πρώτο είναι το JDBC API που αφορά προγραμματιστές εφαρμογών, ενώ το δεύτερο είναι το χαμηλού επιπέδου JDBC driver API που αφορά προγραμματιστές οδηγών.
Ο κύριος σκοπός του driver manager είναι να φορτώνει συγκεκριμένους οδηγούς για την εφαρμογή του χρήστη. Μπορεί επίσης να εκτελέσει τις παρακάτω λειτουργίες:

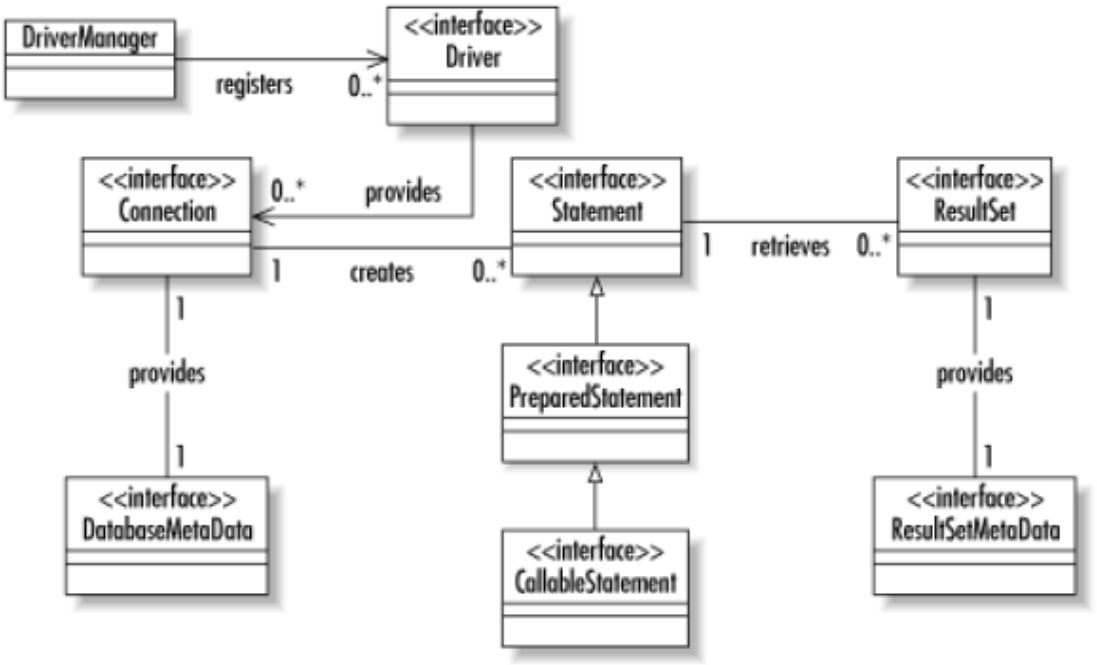
- Να εντοπίσει έναν οδηγό για μια συγκεκριμένη βάση δεδομένων
- Να επεξεργαστεί κλήσεις αρχικοποίησης JDBC
- Να παρέχει σημεία εισόδου σε συναρτήσεις JDBC για κάθε συγκεκριμένο οδηγό
- Να εκτελεί επιβεβαίωση παραμέτρων και ακολουθίας για κλήσεις JDBC

Ο driver επεξεργάζεται κλήσεις μεθόδων JDBC, αποστέλλει εντολές SQL σε μια συγκεκριμένη πηγή δεδομένων και επιστρέφει τα αποτελέσματα στην εφαρμογή. Όταν είναι απαραίτητο, ο driver μεταφράζει ή/και βελτιστοποιεί τις αιτήσεις έτσι ώστε να συμμορφώνονται στο συντακτικό που υποστηρίζει το συγκεκριμένο DBMS. Ο driver εκτελεί τα παρακάτω:

- Εδραιώνει μια σύνδεση σε μια πηγή δεδομένων
- Αποστέλλει αιτήσεις στην πηγή δεδομένων
- Επιστρέφει αποτελέσματα στην εφαρμογή του χρήστη
- Μορφοποιεί τα σφάλματα σε standard JDBC κωδικούς σφαλμάτων
- Χειρίζεται cursors
- Αρχικοποιεί δοσοληψίες

Τέλος η πηγή δεδομένων αποτελείται από τα δεδομένα που θέλει να προσπελάσει ο χρήστης και τις συσχετιζόμενες με αυτά παραμέτρους, όπως τον τύπο του DBMS και το επίπεδο δικτύου (εάν υπάρχει) που χρησιμοποιείται για την πρόσβαση στο DBMS. [1]

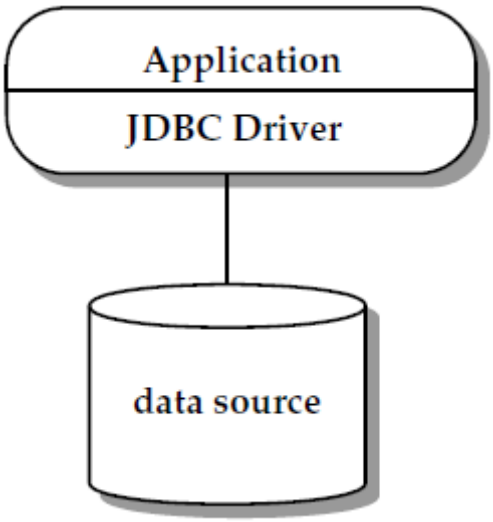
Στο σχήμα 1-6 φαίνεται το διάγραμμα κλάσεων UML των βασικών κλάσεων και διεπαφών του JDBC API:



Σχήμα 1-6. Οι βασικές τάξεις και διεπαφές του JDBC API [4]

1.4 Τα μοντέλα Two-Tier και Three-Tier

Το JDBC API υποστηρίζει το μοντέλο δύο επιπέδων (two-tier) καθώς και το μοντέλο τριών επιπέδων (three-tier) για την προσπέλαση μιας βάσης δεδομένων. Στο μοντέλο two-tier, ένα Java applet ή εφαρμογή επικοινωνεί απευθείας με την πηγή δεδομένων. Αυτό απαιτεί έναν οδηγό JDBC που να μπορεί να επικοινωνήσει με την συγκεκριμένη πηγή δεδομένων. Οι εντολές του χρήστη αποστέλλονται στην βάση δεδομένων ή σε κάποια άλλη πηγή δεδομένων και τα αποτελέσματα των εντολών επιστρέφονται στον χρήστη. Η πηγή δεδομένων μπορεί να βρίσκεται σε ένα άλλο μηχάνημα στο οποίο ο χρήστης συνδέεται μέσω ενός δικτύου. Αυτή η διάταξη αποκαλείται διάταξη πελάτη/εξυπηρετητή (client/server), με το μηχάνημα του χρήστη ως πελάτη και την μηχανή που στεγάζει την πηγή δεδομένων ως εξυπηρετητή. Το δίκτυο μπορεί να είναι ένα εσωτερικό δίκτυο π.χ. μιας επιχείρησης (intranet) ή το Διαδίκτυο (Internet). [5] Στο σχήμα 1-7 φαίνεται το μοντέλο two-tier:



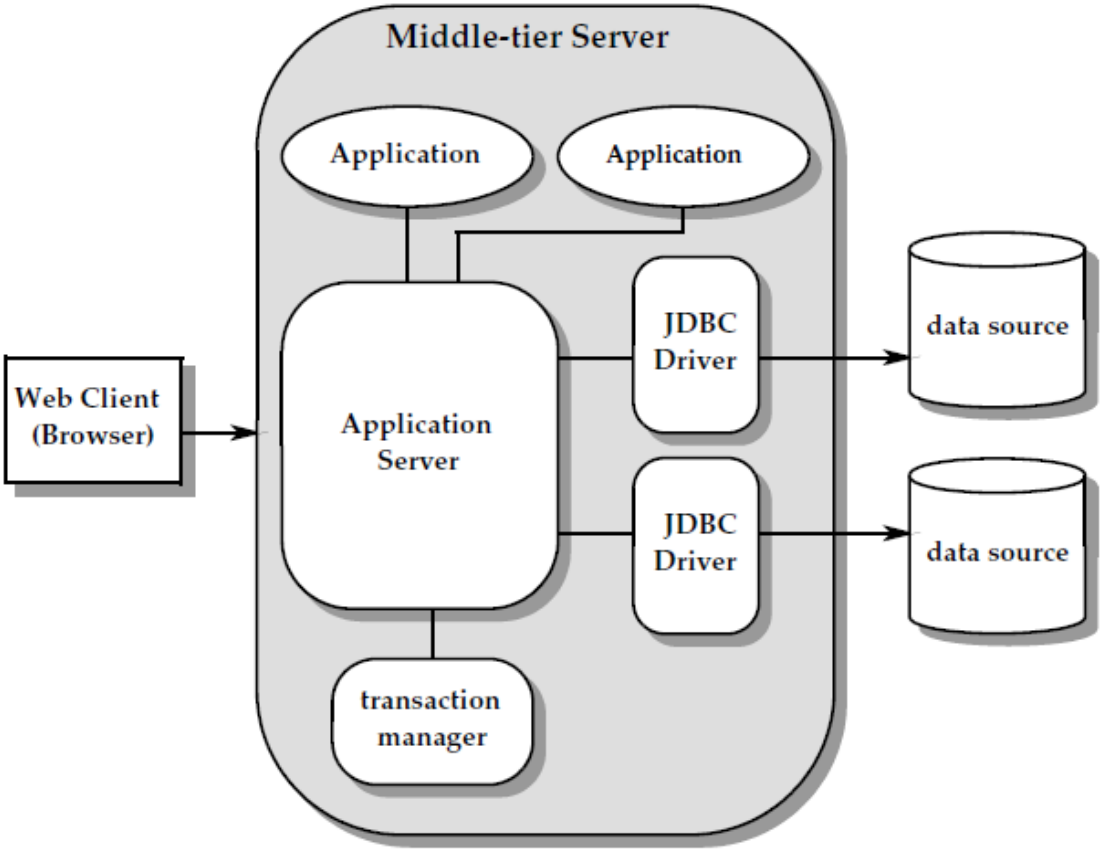
Σχήμα 1-7: Το μοντέλο two-tier [7]

Στο μοντέλο three-tier οι εντολές αποστέλλονται σε ένα μέσο επίπεδο (middle-tier) υπηρεσιών, το οποίο μετά τις αποστέλλει στην πηγή δεδομένων. Η πηγή δεδομένων επεξεργάζεται τις εντολές και επιστρέφει τα αποτελέσματα στο middle-tier, το οποίο τα επιστρέφει στον χρήστη. [5] Η αρχιτεκτονική αυτή είναι σχεδιασμένη να παρέχει βελτιωμένη απόδοση, κλιμάκωση (scalability) και διαθεσιμότητα. Η λειτουργικότητα κατανέμεται στα τρία επίπεδα κατά αυτό τον τρόπο:

- Client tier: ένα λεπτό επίπεδο που υλοποιεί την λογική παρουσίασης (presentation logic) για ανθρώπινη αλληλεπίδραση. Προγράμματα Java, περιηγητές Ιστού και PDAs είναι τυπικές υλοποιήσεις ενός client-tier. Ο πελάτης επικοινωνεί με την εφαρμογή middle-tier χωρίς να χρειάζεται να γνωρίζει την υποδομή ή οποιεσδήποτε συναρτήσεις της πηγής δεδομένων.
- Middle-tier server: Ο εξυπηρετητής middle-tier περιλαμβάνει τα παρακάτω:
    - Εφαρμογές για την αλληλεπίδραση με τον πελάτη και την υλοποίηση της επιχειρηματικής λογικής (business logic). Εάν περιλαμβάνεται αλληλεπίδραση με κάποια πηγή δεδομένων σε κάποια εφαρμογή, αυτή θα επιτυγχάνεται χρησιμοποιώντας αφαιρέσεις υψηλότερου επιπέδου, όπως αντικείμενα DataSource και λογικές συνδέσεις αντί του χαμηλού επιπέδου API του οδηγού.

- Έναν εξυπηρετητή εφαρμογών για την υποστήριξη της υποδομής ενός μεγάλου εύρους φάσματος εφαρμογών. Αυτό μπορεί να περιλαμβάνει διαχείριση και pooling φυσικών συνδέσεων, διαχείριση δοσοληψιών, και απόκρυψη διαφορών μεταξύ διαφορετικών οδηγών JDBC.
- Έναν οδηγό JDBC ο οποίος επιτρέπει την σύνδεση σε υποκείμενες πηγές δεδομένων. Κάθε οδηγός JDBC υλοποιεί στο ανώτερο επίπεδο το standard JDBC API, επάνω από οποιαδήποτε χαρακτηριστικά υποστηρίζονται από την υποκείμενη πηγή δεδομένων. Το επίπεδο οδηγού μπορεί να αποκρύπτει συντακτικές διαφορές μεταξύ της standard SQL:2003 και της τοπικής διαλέκτου που υποστηρίζεται από την πηγή δεδομένων. Εάν η πηγή δεδομένων δεν είναι ένα σχεσιακό σύστημα διαχείρισης βάσεων δεδομένων, τότε ο οδηγός υλοποιεί το σχεσιακό επίπεδο που χρησιμοποιείται από τον εξυπηρετητή εφαρμογών.

- Data source tier: Στο αυτό το tier βρίσκονται τα δεδομένα. Μπορεί να περιλαμβάνει σχεσιακά συστήματα διαχείρισης βάσεων δεδομένων, legacy συστήματα αρχείων, αντικειμενοστραφή συστήματα διαχείρισης βάσεων δεδομένων, αποθήκες δεδομένων (data warehouses), λογιστικά φύλλα ή άλλους τρόπους πακεταρίσματος και παρουσίασης δεδομένων. Η μόνη απαίτηση είναι ένας αντίστοιχος οδηγός που να υποστηρίζει το JDBC API. [7]



Σχήμα 1-8: Το μοντέλο three-tier [7]

1.5 Τύποι οδηγών JDBC

Υπάρχουν τέσσερις κατηγορίες οδηγών JDBC. Η Sun έχει ονομάσει τις κατηγορίες type 1-4. [4] Μερικοί οδηγοί είναι γραμμένοι σε Java και είναι φορητοί ενώ άλλοι είναι μόνο εν μέρει γραμμένοι σε Java και χρησιμοποιούν βιβλιοθήκες για να επικοινωνήσουν με κάποια συγκεκριμένη βάση. Ένας ειδικός τύπος είναι η γέφυρα JDBC-ODBC, η οποία επιτρέπει την χρήση οδηγών ODBC για την δημιουργία συνδέσεων σε βάσεις δεδομένων που βρίσκονται σε πλατφόρμες Microsoft Windows ή UNIX. Οι τύποι των οδηγών φαίνονται στον πίνακα 1-1:

Πίνακας 1-1: Τύποι οδηγών JDBC [1]

Τυπος	**Περιγραφή**
Τύπος 1	Οδηγοί γέφυρας (για παράδειγμα η γέφυρα JDC-ODBC)
Τύπος 2	Οδηγοί γραμμένοι εν μέρει σε Java και σε εγγενή κώδικα

Τύπος 3	Οδηγοί πελάτη Java που χρησιμοποιούν ένα μεσολαβητή στην μεριά του server
Τύπος 4	Οδηγοί πελάτη Java που συνδέονται απευθείας στην βάση δεδομένων

1.5.1 Οδηγοί Τύπου 1

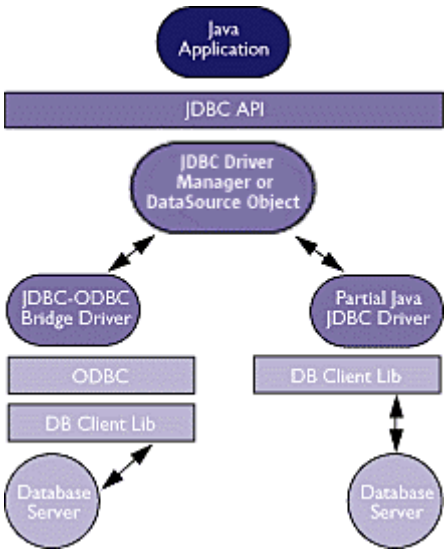
Ένας οδηγός JDBC τύπου 1 δεν υλοποιεί ένα πρωτόκολλο επικοινωνίας για κάποια συγκεκριμένη βάση. Παρέχει μόνο μια αντιστοίχηση στο API μιας άλλης βάσης δεδομένων. Η γέφυρα JDBC-ODBC είναι ένας τέτοιος οδηγός. Επιτρέπει την χρήση υπαρχόντων οδηγών ODBC για την σύνδεση σε κάποια βάση δεδομένων χωρίς να υποχρεώνει τους προγραμματιστές να χρησιμοποιήσουν κλήσεις ODBC από το πρόγραμμά τους. Στις περισσότερες περιπτώσεις, ο οδηγός ODBC βασίζεται σε έναν εγγενή οδηγό, ο οποίος με τη σειρά του βασίζεται σε έναν οδηγό Net. Ο ρόλος του εγγενή οδηγού είναι να παρέχει λειτουργική πρόσβαση σε κάποια συγκεκριμένη μηχανή βάσης δεδομένων και ο ρόλος του οδηγού Net είναι να παρέχει ένα πρωτόκολλο επικοινωνίας πχ για το TCP/IP ή τους μηχανισμούς ενδοεπικοινωνίας διεργασιών (Inter-Process Communication - IPC). Στο σχήμα 1-9 και αριστερά φαίνονται τα επίπεδα των οδηγών τύπου 1.

Το πλεονέκτημα ενός οδηγού αυτού του τύπου είναι πως επιτρέπει την γρήγορη δημιουργία σύνδεσης σε μια βάση δεδομένων, εφόσον είναι διαθέσιμος ένας οδηγός ODBC. Το μειονέκτημα είναι πως η χρήση του ODBC δεν είναι πάντα ο πιο γρήγορος τρόπος επικοινωνίας με την βάση δεδομένων καθώς εμπλέκονται πολλά επίπεδα κώδικα: η γέφυρα JDBC-ODBC, ο ODBC driver manager, ο οδηγός ODBC και η βιβλιοθήκη Net που χρησιμοποιεί ο οδηγός ODBC. [1]

1.5.2 Οδηγοί τύπου 2

Με οδηγούς τύπου 2 ένα εγγενές API προωθεί τις κλήσεις σε μια τοπικά εγκατεστημένη βιβλιοθήκη που μπορεί να έχει αναπτυχθεί σε C ή κάποια άλλη γλώσσα προγραμματισμού, και παρέχεται από τον κατασκευαστή της βάσης δεδομένων. [1]

Οι κλήσεις JDBC μετατρέπονται σε κλήσεις του API πελάτη διαφόρων συστημάτων διαχείρισης βάσεων δεδομένων. Αυτός ο τύπος οδηγού απαιτεί την φόρτωση κάποιας βιβλιοθήκης DLL (για Windows) ή .so (για UNIX) σε κάθε μηχανή πελάτη. Οι οδηγοί τύπου 2 είναι οι πιο αποδοτικοί καθώς χρησιμοποιούν βιβλιοθήκες οι οποίες είναι θεωρητικά βελτιστοποιημένες ως προς την βάση δεδομένων για την οποία δημιουργήθηκαν, αλλά επειδή οι βιβλιοθήκες είναι εγγενής, η λύση δεν θα είναι φορητή. Η λύση αυτή απαιτεί πλήρη εγκατάσταση και ρύθμιση των βιβλιοθηκών. Στο σχήμα 1-9 και δεξιά φαίνεται η δομή ενός οδηγού τύπου 2. [6]



Σχήμα 1-9: Άριστερα: Οδηγός Τύπου 1 - Δεξια: Οδηγός Τύπου 2 [5]

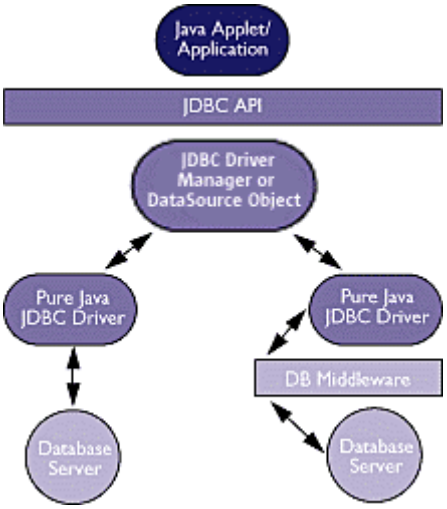
1.5.3 Οδηγοί τύπου 3

Οι οδηγοί τύπου 3 είναι γραμμένοι πλήρως σε Java και χρησιμοποιούν ένα πρωτόκολλο δικτύου το οποίο είναι ανεξάρτητο του συστήματος διαχείρισης βάσεων δεδομένων. Είναι εξαιρετικά φορητοί λόγω του ότι είναι γραμμένοι πλήρως σε Java και μη-απαιτητικοί σε πόρους καθώς υλοποιούν μόνο ένα λεπτό επίπεδο επικοινωνίας με ένα πρόγραμμα μεσολαβητή που βρίσκεται στον εξυπηρετητή. [1] Αυτό το είδος οδηγού μεταφράζει τις κλήσεις JDBC στο πρωτόκολλο του κατασκευαστή του μεσολαβητή, οι οποίες μετά μεταφράζονται από έναν εξυπηρετητή-μεσολαβητή σε κάποιο πρωτόκολλο ΣΔΒΔ. Ο μεσολαβητής παρέχει συνδεσιμότητα σε πολλές διαφορετικές βάσεις δεδομένων. [5] Στο σχήμα 1-10 και δεξιά φαίνεται η δομή του οδηγού τύπου

3.

1.5.4 Οδηγοί τύπου 4

Οι οδηγοί τύπου 4 είναι εγγενούς πρωτοκόλλου Java οδηγοί, οι οποίοι υλοποιούν σε Java όλα τα επίπεδα που είναι απαραίτητα για την επικοινωνία με την βάση δεδομένων. Είναι πλήρως φορητοί επειδή δεν χρησιμοποιούν τοπικές βιβλιοθήκες ή άλλο εγγενή κώδικα. Οι οδηγοί τύπου 4 είναι οι πιο φορητοί επειδή εμπεριέχουν μόνο τάξεις Java. Είναι ικανοί να συνδεθούν στην βάση δεδομένων χωρίς πρόσθετο λογισμικό στην μεριά του πελάτη ή του εξυπηρετητή, καθώς υλοποιούν το πρωτόκολλο της βάσης δεδομένων. Είναι οι πιο εύχρηστοι όσον αφορά την εγκατάσταση και την διανομή. [1] Οι κλήσεις JDBC μεταφράζονται στο πρωτόκολλο του ΣΔΒΔ, επιτρέποντας έτσι απευθείας κλήσεις από την μηχανή πελάτη στον εξυπηρετητή ΣΔΒΔ. [5] Στο σχήμα 1-10 και αριστερά φαίνεται η δομή του οδηγού τύπου 4.



Σχήμα 1-10: Αριστερά: Οδηγοί Τύπου 4 – Δεξιά: Οδηγοί Τύπου 3

1.6 Τα πακέτα του JDBC

Το JDBC αποτελείται από δύο πακέτα: το κυρίως πακέτο java.sql, που ονομάζεται JDBC core API και το πακέτο επέκτασης javax.sql, που ονομάζεται JDBC Optional Package API. Τα πακέτα αυτά παρέχουν το API για την πρόσβαση και την επεξεργασία δεδομένων που είναι αποθηκευμένα σε κάποια πηγή δεδομένων (συνήθως μια σχεσιακή βάση δεδομένων) χρησιμοποιώντας την γλώσσα προγραμματισμού Java. Το πακέτο javax.sql επεκτείνει την λειτουργικότητα του JDBC API από ένα API μεριάς-πελάτη (client-side API) σε ένα API μεριάς-εξυπηρετητή (server-side API). [5] Οι

οδηγοί JDBC πρέπει να υλοποιούν τις διεπαφές του πακέτου java.sql ώστε να συμμορφώνονται στο JDBC αλλά δεν είναι υποχρεωμένοι να υλοποιούν το πακέτο javax.sql. [1]

1.6.1 Το πακέτο java.sql

Το πακέτο java.sql εμπεριέχει API για τα παρακάτω: [5]

- Δημιουργία σύνδεσης με μια βάση δεδομένων μέσω του DriverManager:
    - Τάξη DriverManager: δημιουργεί μια σύνδεση με έναν οδηγό
    - Τάξη SQLPermission: παρέχει άδεια όταν κώδικας που τρέχει μέσα σε έναν Security Manager, όπως ένα applet, επιχειρήσει να εδραιώσει μια ροή καταγραφής (logging stream) μέσω του DriverManager.
    - Διεπαφή Driver: Παρέχει το API για για την καταχώρηση και την σύνδεση οδηγών βασισμένων στην τεχνολογία JDBC (οδηγοί JDBC). Γενικά χρησιμοποιείται μόνο από την τάξη DriverManager.
    - Τάξη PropertyInfo: Παρέχει τις ιδιότητες ενός οδηγού JDBC, δεν χρησιμοποιείται από τον γενικό χρήστη.
- Αποστολή δηλώσεων SQL (SQL Statements) στην βάση δεδομένων:
    - Statement: χρησιμοποιείται για την αποστολή βασικών δηλώσεων SQL
    - PreparedStatement: χρησιμοποιείται για την αποστολή προετοιμασμένων δηλώσεων (prepared statements) ή βασικών δηλώσεων SQL
    - CallableStatement: χρησιμοποιείται για την κλήση αποθηκευμένων διαδικασιών (procedures) της βάσης δεδομένων
    - Διεπαφή Connection: παρέχει μεθόδους (methods) για την δημιουργία δηλώσεων και την διαχείριση συνδέσεων και των ιδιοτήτων τους
    - Savepoint: παρέχει αποθήκευση σημείων σε μια δοσοληψία
- Ανάκτηση και ανανέωση των αποτελεσμάτων ενός ερωτήματος:
    - Διεπαφή ResultSet

- Τυποποιημένη αντιστοίχηση τύπων SQL σε τάξεις και διεπαφές της γλώσσας προγραμματισμού Java:
    - Διεπαφή Array: αντιστοίχηση για το SQL ARRAY
    - Διεπαφή Blob: αντιστοίχηση για το SQL BLOB
    - Διεπαφή Clob: αντιστοίχηση για το SQL CLOB
    - Τάξη Date: αντιστοίχηση για το SQL DATE
    - Διεπαφή NClob: αντιστοίχηση για το SQL NCLOB
    - Διεπαφή Ref: αντιστοίχηση για το SQL REF
    - Διεπαφή RowId: αντιστοίχηση για το SQL ROWID
    - Διεπαφή Struct: αντιστοίχηση για το SQL STRUCT
    - Διεπαφή SQLXML: αντιστοίχηση για το SQL XML
    - Τάξη Time: αντιστοίχηση για το SQL TIME
    - Τάξη Timestamp: αντιστοίχηση για το SQL TIMESTAMP
    - Τάξη Types: παρέχει σταθερές για τύπους SQL
- Προσαρμοσμένη αντιστοίχηση ενός ορισμένου από τον χρήστη τύπου SQL (User Defined Type – UDT) σε μια τάξη της γλώσσας προγραμματισμού Java:
    - Διεπαφή SQLData: προσδιορίζει την αντιστοίχηση ενός UDT σε ένα στιγμιότυπο αυτής της τάξης
    - Διεπαφή SQLInput: παρέχει μεθόδους για την ανάγνωση των ιδιοτήτων ενός UDT από μια ροή
    - Διεπαφή SQLOutput: παρέχει μεθόδους για την εγγραφή των ιδιοτήτων ενός UDT σε μια ροή
- Metadata
    - Διεπαφή DatabaseMetaData: παρέχει πληροφορίες για την βάση
    - Διεπαφή ResultSetMetaData: παρέχει πληροφορίες για τις στήλες ενός αντικειμένου ResultSet
    - Διεπαφή ParameterMetaData: παρέχει πληροφορίες για τις παραμέτρους εντολών PreparedStatement
- Exceptions
    - SQLException: προκαλείται από τις περισσότερες μεθόδους όταν υπάρχει κάποιο πρόβλημα στην πρόσβαση των

δεδομένων και από κάποιες άλλες μεθόδους για άλλους λόγους

- SQLWarning: προκαλείται για να υποδείξει μια προειδοποίηση
- DataTruncation: προκαλείται για να υποδείξει πως τα δεδομένα μπορεί να έχουν περικοπεί
- BatchUpdateException: προκαλείται για να υποδείξει πως δεν εκτελέστηκαν επιτυχώς όλες οι εντολές μιας δέσμης ανανέωσης (batch update)

1.6.2 Το πακέτο javax.sql

Το πακέτο javax.sql παρέχει τα παρακάτω: [5]

1. Την διεπαφή DataSource ως εναλλακτική του DriverManager για την δημιουργία μιας σύνδεσης με μια πηγή δεδομένων
2. Connection Pooling και Statement pooling
3. Κατανεμημένες δοσοληψίες (Distributed Transactions)
4. Rowsets

1.6.3 Το αντικειμένο DataSource

Το πακέτο javax.sql επιτρέπει την επιλογή του τρόπου με τον οποίο θα δημιουργηθεί μια σύνδεση με την πηγή δεδομένων. Η τάξη DriverManager, ο αρχικός μηχανισμός, είναι ακόμα έγκυρος και εάν χρησιμοποιείται σε κάποιο κώδικα, αυτός θα συνεχίσει να λειτουργεί. Όμως ο καινούργιος μηχανισμός DataSource είναι προτιμότερος καθώς έχει πολλά πλεονεκτήματα έναντι του μηχανισμού DriverManager. Αυτά είναι τα δύο βασικά πλεονεκτήματα της χρήσης ενός αντικειμένου DataSource για την δημιουργία μιας σύνδεσης:

- Μπορούν να γίνουν αλλαγές στις ιδιότητες μιας πηγής δεδομένων που σημαίνει πως δεν είναι απαραίτητο να γίνουν αλλαγές στον κώδικα της εφαρμογής όταν κάτι αλλάξει στην πηγή δεδομένων ή τον οδηγό.
- Μπορούν να χρησιμοποιηθούν οι κατανεμημένες δοσοληψίες καθώς και το Connection και Statement pooling, μέσω ενός αντικειμένου DataSource που έχει υλοποιηθεί να λειτουργεί με την υποδομή middle-tier. Συνδέσεις που έχουν δημιουργηθεί μέσω του DriverManager δεν έχουν τις δυνατότητες του Connection και Statement pooling.

1.7 Δημιουργία σύνδεσης με το JDBC

Το JDBC αναπαριστά μια σύνδεση με μια βάση δεδομένων μέσω της διεπαφής Connection. Έτσι, για την σύνδεση σε μια βάση δεδομένων απαιτείται η απόκτηση ενός στιγμιότυπου της διεπαφής Connection από τον οδηγό JDBC. Το JDBC υποστηρίζει δύο τρόπους για την απόκτηση πρόσβασης σε μια σύνδεση βάσης δεδομένων:

- Μέσω του JDBC DataSource
- Με την χρήση του JDBC DriverManager

1.7.1 Δημιουργία σύνδεσης με το DataSource

Η δημιουργία μιας σύνδεσης σε μια βάση δεδομένων χρησιμοποιώντας το DataSource επιτυγχάνεται με τον παρακάτω κώδικα:

```
Context ctx = new InitialContext( );
DataSource ds = (DataSource)ctx.lookup("jdbc/dsn");
Connection conn = ds.getConnection( );
```

Η μόνη απαίτηση είναι να υπάρχει μια JNDI (Java Naming and Directory Interface) υπηρεσία καταλόγου που να περιέχει ένα DataSource με το όνομα jdbc/dsn.

Η πρώτη γραμμή παίρνει ένα αντικείμενο InitialContext σε συμφωνία με τις προδιαγραφές του JNDI. Το InitialContext επιτρέπει την πλοήγηση σε μια υπηρεσία καταλόγου. Σε αυτή την περίπτωση χρησιμοποιείται για την εύρεση ενός στιγμιότυπου DataSource με το οποίο αμέσως μετά δημιουργείται η σύνδεση. [4]

1.7.2 Δημιουργία σύνδεσης με τον DriverManager

Η τάξη DriverManager ανήκει στο πακέτο java.sql, διατηρεί μια λίστα υλοποιήσεων της διεπαφής java.sql.Driver και παρέχει συνδέσεις βασισμένες στο JDBC URL που θα δωθεί. Το JDBC URL είναι της μορφής jdbc:πρωτόκολλο:υπό-πρωτόκολλο. Αυτό το URL λέει στον DriverManager με ποια μηχανή βάσης δεδομένων θα γίνει η σύνδεση και του παρέχει τις πληροφορίες που χρειάζεται για να την δημιουργήσει.

Το τμήμα του πρωτοκόλλου στο URL αναφέρεται σε κάποιον οδηγό JDBC. Το

υπό-πρωτόκολλο παρέχει τις απαραίτητες πληροφορίες σύνδεσης για την συγκεκριμένη υλοποίηση. Οι περισσότεροι οδηγοί απαιτούν τουλάχιστον το όνομα μιας βάσης με την οποία θα γίνει η σύνδεση. Συχνά καθορίζεται και ένας εξυπηρετητής καθώς και ένας αριθμός θύρας στο υπό-πρωτόκολλο. Το JDBC URL του κάθε οδηγού είναι διαφορετικό και συνήθως παρέχεται στο documentation του οδηγού. Άσχετα από την μορφή του URL, κύρια λειτουργία του είναι να καθορίσει τον οδηγό που χρειάζεται η εφαρμογή και να παρέχει στον οδηγό αυτό όσες πληροφορίες του είναι απαραίτητες για να δημιουργηθεί μια σύνδεση σε μια βάση δεδομένων. [4]

Προτού να είναι δυνατή η χρήση ενός URL για την δημιουργία μιας σύνδεσης με τον DriverManager, θα πρέπει προηγουμένως να δηλωθεί η υλοποίηση του Driver στον DriverManager. Η δήλωση του Driver μπορεί να γίνει με δύο τρόπους: [4]

- Καθορίζοντας τα ονόματα των τάξεων υλοποιήσεων του Driver που πρέπει να δηλωθούν, στην γραμμή εντολών με την χρήση της ιδιότητας jdbc.drivers:

```
Java -Djdbc.drivers=com.mysql.jdbc.Driver MyAppClass
```

- Δηλώνοντας κάποιον συγκεκριμένο οδηγό μέσα από το πρόγραμμα χρησιμοποιώντας την Class.forName():

```
Class.forName("com.mysql.jdbc.Driver");
```

Σημειώνεται πως από την έκδοση JDBC 4.0 ο DriverManager θα φορτώσει αυτόματα οποιουσδήποτε οδηγούς JDBC 4.0 που βρίσκονται μέσα στο CLASSPATH, όταν μια εφαρμογή επιχειρήσει για πρώτη φορά να συνδεθεί σε κάποια πηγή δεδομένων καθορίζοντας ένα URL. Οδηγοί που είναι προ-JDBC 4.0 πρέπει να φορτωθούν μη-αυτόματα από την εφαρμογή χρησιμοποιώντας κάποιον από τους δύο τρόπους που αναφέρθηκαν προηγουμένως. [7]

Αφού δηλωθεί ο οδηγός, μπορεί να ζητηθεί ένα Connection από τον DriverManager καλώντας την μέθοδο getConnection() του οδηγού. Στην κλήση της μεθόδου πρέπει να συμπεριλαμβάνονται τουλάχιστον το JDBC

URL, το user ID και το password:

```
Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost/myDB", "username",
"password");
```

Αυτό το κομμάτι κώδικα επιστρέφει μια σύνδεση με μία βάση MySQL, που έχει το όνομα MyDB, χρησιμοποιώντας τον οδηγό MySQL. Η σύνδεση αυτή θα δημιουργηθεί με την άδεια του χρήστη “username” ο οποίος θα αναγνωριστεί από τον κωδικό “password”.
Ο παρακάτω κώδικας δείχνει ένα βασικό πρόγραμμα Java που χρησιμοποιεί το JDBC για να συνδεθεί σε μια βάση δεδομένων MySQL:

```
import java.sql.*;
public class BasicJDBCDemo {
        Connection conn;
        String url = "jdbc:mysql://localhost/myDB";
        String username = “username”;
        String password = “password”;

        public static void main(String[] args) {
                try {
                        Class.forName("com.mysql.jdbc.Driver");
                }
                catch (Exception e) {
                        System.out.println ("Could not load the driver: \n" +
e.getMessage()) ;
                        System.exit(1);
                }
                try {
                        conn = DriverManager.getConnection(url, username,
password);
                }
                catch (Exception e) {
```

```
                    System.out.println ("Could not connect to " + URL + ”\n” +
e.getMessage()); System.exit(1);
                }
        }
}
```

1.8 Εκτέλεση εντολών SQL

Το πιο βασικό στοιχείο επικοινωνίας μέσω ενός Connection είναι το Statement. Η εφαρμογή ενθυλακώνει εντολές SQL σε ένα Statement ή σε κάποια από τις υπό-τάξεις του και επεξεργάζεται τα αποτελέσματα.
Η τάξη Connection επιτρέπει την δημιουργία στιγμιότυπων Statement μέσω της μεθόδου createStatement():

```
Statement stmt = conn.createStatement( );
```

Τώρα το στιγμιότυπο Statement μπορεί να χρησιμοποιηθεί για την αποστολή εντολών SQL στην βάση δεδομένων:

```
stmt.executeUpdate("UPDATE drinks SET val = 'coffee' WHERE id = 3");
```

1.8.1 Result sets

Στο προηγούμενο παράδειγμα, η εντολή SQL που θα εκτελεστεί τροποποιεί την βάση δεδομένων αλλά δεν επιστρέφει γραμμές. Διαφορετικά, εάν οι εντολές SQL πρόκειται να επιστρέψουν γραμμές, μπορεί να χρησιμοποιηθεί η μέθοδος executeQuery() σε συνδυασμό με ένα αντικείμενο ResultSet:

```
ResultSet rs = stmt.executeQuery("SELECT id, val FROM test");
```

Για την προσπέλαση των γραμμών που έχουν επιστραφεί από το ερώτημα SQL που έχει εκτελεστεί και την εκτύπωση τους μπορεί να χρησιμοποιηθεί η μέθοδος next() του στιγμιότυπου ResultSet:

```
while(rs.next()) {
        System.out.println("ID: " + rs.getInt(1) + ", rs.getString(2));
```

```
}
```

Πριν την πρώτη κλήση της next(), το ResultSet δεν δείχνει σε καμία από τις γραμμές που μπορεί να επιστράφηκαν από το ερώτημα. Με την πρώτη κλήση της next(), το ResultSet θα δείχνει στην πρώτη γραμμή των αποτελεσμάτων. Έως την επόμενη κλήση της next(), οποιεσδήποτε ενέργειες εκτελεστούν στο ResultSet θα επιδράσουν στην γραμμή αυτή. Κάθε κλήση της next() μετακινεί το ResultSet στην επόμενη γραμμή των αποτελεσμάτων. Εάν δεν υπάρχει επόμενη γραμμή, η κλήση της next()θα επιστρέψει false.
Για την ανάκτηση των τιμών των στηλών μιας γραμμής μπορούν να χρησιμοποιηθούν οι μέθοδοι get της διεπαφής ResultSet. Στο προηγούμενο παράδειγμα, η getInt() χρησιμοποιείται για την ανάκτηση της στήλης id, ενώ η getString() για την ανάκτηση της στήλης val. Αυτές οι μέθοδοι get μπορούν να δεχθούν τον αριθμό μιας στήλης (ξεκινώντας από το 1), ή το όνομα μιας στήλης. [4]
Τέλος, σε περίπτωση που δεν είναι γνωστό εάν κάποια εντολή SQL επιστρέφει γραμμές ή όχι, θα πρέπει να χρησιμοποιηθεί η μέθοδος execute(). [7]

1.8.2 Δοσοληψίες

Οι δοσοληψίες χρησιμοποιούνται για να παρέχουν ακεραιότητα δεδομένων, σωστή σημασιολογία εφαρμογών και μια συνεπή όψη των δεδομένων κατά την ταυτόχρονη προσπέλαση. Η διαχείριση δοσοληψιών στο JDBC API ακολουθεί την προδιαγραφή SQL:2003 και συμπεριλαμβάνει τις ακόλουθες έννοιες: [7]

- Auto-commit mode
- Transaction isolation levels
- Savepoints

1.8.3 Όρια δοσοληψιών και Auto-commit

Το πότε θα ξεκινήσει μια καινούργια δοσοληψία αποφασίζεται από τον οδηγό JDBC ή την υποκείμενη πηγή δεδομένων. Συνήθως μια νέα δοσοληψία ξεκινά όταν η τρέχουσα εντολή SQL απαιτεί μια και δεν υπάρχει κάποια άλλη ενεργή δοσοληψία. Το εάν κάποια εντολή SQL απαιτεί μια δοσοληψία καθορίζεται επίσης από την προδιαγραφή SQL:2003.

Το χαρακτηριστικό auto-commit καθορίζει το πότε λήγει μια δοσοληψία. Η ενεργοποίηση του auto-commit προκαλεί την εκτέλεση μιας δοσοληψίας μετά από την ολοκλήρωση κάθε εντολής SQL. Το σημείο στο οποίο μια εντολή θεωρείται πως έχει ολοκληρώσει εξαρτάται από τον τύπο της εντολής SQL καθώς και από το τι κάνει η εφαρμογή μετά την εκτέλεσή της:

- Για εντολές γλώσσας τροποποίησης δεδομένων (DML – Data Manipulation Language) όπως οι Insert, Update, Delete καθώς και για εντολές γλώσσας ορισμού δεδομένων (DDL – Data Definition Language), η εντολή ολοκληρώνει μόλις τελειώσει η εκτέλεσή της.
- Για εντολές Select, η εντολή ολοκληρώνει όταν το σχετιζόμενο result set κλείσει.
- Για αντικείμενα CallableStatement ή για εντολές που επιστρέφουν πολλαπλά αποτελέσματα, η εντολή ολοκληρώνει όταν όλα τα σχετιζόμενα result set έχουν κλείσει και όλα τα update counts και οι παράμετροι εξόδου έχουν ληφθεί.

1.8.4 Απενεργοποίηση του Auto-commit

Θεωρώντας πως υπάρχει ένα αντικείμενο conn του τύπου Connection, το auto-commit μπορεί να απενεργοποιηθεί καλώντας την μέθοδο setAutoCommit() του αντικειμένου conn:

```
conn.setAutoCommit(false);
```

Όταν το auto-commit έχει απενεργοποιηθεί, κάθε δοσοληψία θα πρέπει να εκτελεστεί ή να αναιρεθεί μη-αυτόματα καλώντας τις μεθόδους commit και rollback της Connection αντίστοιχα. Αυτό είναι χρήσιμο σε περιπτώσεις όπου η διαχείριση των δοσοληψιών γίνεται ένα επίπεδο επάνω από τον οδηγό,

όπως:

- Όταν η εφαρμογή χρειάζεται να ομαδοποιήσει πολλαπλές εντολές SQL σε μια μόνο δοσοληψία
- Όταν οι δοσοληψίες διαχειρίζονται από τον εξυπηρετητή εφαρμογών

Η προεπιλογή είναι το auto-commit να είναι ενεργοποιημένο όταν δημιουργείται ένα αντικείμενο Connection. Εάν η τιμή του auto-commit αλλάξει κατά την διάρκεια μιας δοσοληψίας, η τρέχουσα δοσοληψία εκτελείται. [7]

1.8.5 Transaction isolation levels

Τα επίπεδα απομόνωσης δοσοληψιών καθορίζουν ποια δεδομένα είναι ορατά στις εντολές μιας δοσοληψίας. Επηρεάζουν άμεσα το επίπεδο της ταυτόχρονης πρόσβασης ορίζοντας ποια αλληλεπίδραση επιτρέπεται μεταξύ δοσοληψιών που ενεργούν στην ίδια πηγή δεδομένων. Η αλληλεπίδραση μεταξύ ταυτόχρονων δοσοληψιών χωρίζεται στις εξής κατηγορίες:

- Dirty reads: Συμβαίνουν όταν επιτρέπεται στις δοσοληψίες να βλέπουν ανεκτέλεστες αλλαγές στα δεδομένα. Αυτό σημαίνει πως αλλαγές που συμβαίνουν σε μια δοσοληψία είναι ορατές εκτός της δοσοληψίας πριν αυτή εκτελεστεί. Εάν οι αλλαγές αναιρεθούν αντί να εκτελεστούν, είναι πιθανό άλλες δοσοληψίες να έχουν εργαστεί βασισμένες σε λάθος, μεταβατικά δεδομένα.
- Nonrepeatable reads συμβαίνουν όταν:
  1. Η δοσοληψία A διαβάζει μια γραμμή
  2. Η δοσοληψία B αλλάζει την γραμμή αυτή
  3. Η δοσοληψία A διαβάζει την ίδια γραμμή για δεύτερη φορά και παίρνει διαφορετικά αποτελέσματα
- Phantom reads συμβαίνουν όταν:
  1. Η δοσοληψία A διαβάζει όλες τις γραμμές που ικανοποιούν μια συνθήκη WHERE
  2. Η δοσοληψία B εισάγει μια επιπλέον γραμμή που ικανοποιεί την συνθήκη αυτή
  3. Η δοσοληψία A επανεκτιμεί την συνθήκη WHERE και λαμβάνει την καινούργια γραμμή «φάντασμα».

Το JDBC αυξάνει τα τέσσερα επίπεδα απομόνωσης δοσοληψιών που ορίζει η SQL:2003 προσθέτοντας το TRANSACTION_NONE. Από το λιγότερο περιοριστικό προς το περισσότερο περιοριστικό, τα επίπεδα απομόνωσης δοσοληψιών είναι:

1. TRANSACTION_NONE: υποδεικνύει πως ο οδηγός δεν υποστηρίζει δοσοληψίες, που σημαίνει πως δεν είναι συμμορφωμένος στο JDBC.
2. TRANSACTION_READ_UNCOMMITED: επιτρέπει σε δοσοληψίες να βλέπουν ανεκτέλεστες αλλαγές στα δεδομένα. Αυτό σημαίνει πως είναι πιθανό να συμβούν dirty reads, nonrepeatable reads και phantom reads.
3. TRANSACTION_READ_COMMITED: σημαίνει πως οποιεσδήποτε αλλαγές συμβαίνουν εντός μιας δοσοληψίας δεν είναι ορατές εκτός της δοσοληψίας έως αυτή να εκτελεστεί. Αυτό αποτρέπει τα dirty reads, αλλά είναι πιθανό να συμβούν nonrepeatable reads και phantom reads.
4. TRANSACTION_REPEATABLE_READ: απαγορεύει τα dirty reads και τα nonrepeatable reads, αλλά είναι πιθανό να συμβούν phantom reads.
5. TRANSACTION_SERIALIZABLE: αποτρέπει τα dirty reads, τα nonrepeatable reads και τα phantom reads.

Το προεπιλεγμένο επίπεδο απομόνωσης για ένα αντικείμενο Connection καθορίζεται από τον οδηγό που παρέχει την σύνδεση. Συνήθως είναι το προεπιλεγμένο επίπεδο απομόνωσης που υποστηρίζεται από την υποκείμενη πηγή δεδομένων. Η μέθοδος setTransactionIsolation της Connection παρέχεται για να επιτρέπει σε πελάτες JDBC να αλλάξουν το επίπεδο απομόνωσης δοσοληψιών ενός αντικειμένου Connection. Το καινούργιο επίπεδο απομόνωσης παραμένει ενεργό για το υπόλοιπο της συνεδρίας ή έως την επόμενη κλήση της μεθόδου setTransactionIsolation. Το αποτέλεσμα της κλήσης της μεθόδου setTransactionIsolation κατά την διάρκεια μιας δοσοληψίας ορίζεται από την υλοποίηση. Συνιστάται οι οδηγοί να υλοποιούν την μέθοδο setTransactionIsolation να αλλάζει το επίπεδο απομόνωσης με την επόμενη δοσοληψία. Μια ακόμη έγκυρη υλοποίηση είναι να εκτελείται η τρέχουσα δοσοληψία ώστε η αλλαγή να επιδρά άμεσα.
Είναι πιθανό για έναν οδηγό JDBC να μην υποστηρίζει κανένα από τα

τέσσερα επίπεδα απομόνωσης (χωρίς το TRANSACTION_NONE). Εάν ένας οδηγός δεν υποστηρίζει το επίπεδο απομόνωσης που καθορίζεται από μια κλήση της setTransactionIsolation, επιτρέπεται να το αντικαταστήσει με κάποιο υψηλότερο και πιο περιοριστικό επίπεδο απομόνωσης δοσοληψιών. Η μέθοδος supportsTransactionIsolationLevel της DataBaseMetadata μπορεί να χρησιμοποιηθεί για να διαπιστωθεί εάν ο οδηγός υποστηρίζει κάποιο επίπεδο ή όχι.
Καθώς το επίπεδο απομόνωσης των δοσοληψιών αυξάνεται, επιβαρύνεται το σύστημα διαχείρισης βάσεων δεδομένων. Αυτό έχει ως αποτέλεσμα την πτώση του βαθμού ταυτόχρονης προσπέλασης που υποστηρίζεται, μειώνοντας έτσι την απόδοση της εφαρμογής.

1.8.6 Savepoints

Η μέθοδος Connection.setSavepoint μπορεί να χρησιμοποιηθεί για την δημιουργία ενός savepoint στην τρέχουσα δοσοληψία. Η κλήση της setSavepoint προκαλεί την έναρξη μιας δοσοληψίας εάν δεν υπάρχει κάποια ενεργή δοσοληψία. Η μέθοδος Connection.rollback έχει υπερφορτωθεί ώστε να δέχεται ένα savepoint ως παράμετρο.

```
conn.createStatement();
stmt.executeUpdate("INSERT INTO drinks VALUES(1,“coffee”)”);

Savepoint savepnt1 = conn.setSavepoint("SAVEPOINT_1");

stmt.executeUpdate("INSERT INTO drinks VALUES(2,“tea”)");
conn.rollback(savepnt1);
conn.commit();
```

Στο παραπάνω παράδειγμα εισάγεται μια γραμμή σε έναν πίνακα, δημιουργείται το savepoint savepnt1 και εισάγεται μια ακόμα γραμμή στον πίνακα. Όταν αργότερα η δοσοληψία επιστρέψει στο σημείο savepnt1, η δεύτερη εισαγωγή αναιρείται, αλλά η πρώτη παραμένει ανέπαφη. Όταν τελικά εκτελεστεί η δοσοληψία θα έχει εισαχθεί μόνο μια γραμμή στον πίνακα. [7]

1.8.6 Αποδέσμευση ενός savepoint

Η μέθοδος Connection.releaseSavepoint δέχεται ένα savepoint ως παράμετρο και το διαγράφει μαζί με οποιαδήποτε ακόλουθα savepoints της τρέχουσας δοσοληψίας. Μετά την διαγραφή ενός savepoint, η χρήση του σε κάποια λειτουργία οπισθοδρόμησης θα προκαλέσει ένα SQLException.

Οποιαδήποτε savepoints δημιουργήθηκαν κατά την διάρκεια μιας δοσοληψίας αποδεσμεύονται αυτόματα όταν η δοσοληψία λήξει ή όταν η δοσοληψία αναιρεθεί πλήρως. Η οπισθοδρόμηση μιας δοσοληψίας σε ένα savepoint θα αποδεσμεύσει οποιαδήποτε savepoints δημιουργήθηκαν μετά το συγκεκριμένο savepoint.

Κεφάλαιο 2: Βάσεις Δεδομένων και MySQL

Εισαγωγή

Η MySQL είναι το πιο ευρέως χρησιμοποιούμενο σύστημα διαχείρισης βάσεων δεδομένων ανοιχτού κώδικα. Αυτό συμβαίνει λόγω του ότι η MySQL είναι γρήγορη, σταθερή, εύκολη στην εκμάθηση, μπορεί να εκτελεστεί σε όλα τα δημοφιλή λειτουργικά συστήματα, οι εφαρμογές που την χρησιμοποιούν μπορούν να γραφούν σε διάφορες γλώσσες προγραμματισμού και υπάρχει πληθώρα εγγράφων στο Διαδίκτυο αλλά και πολλά βιβλία για αυτήν. Η MySQL είναι διαθέσιμη δωρεάν για πολλές εφαρμογές (άδεια GPL), αλλά επειδή οι περιορισμοί του GPL δεν είναι αποδεκτοί για πολλές εμπορικές εφαρμογές, υπάρχουν εμπορικές άδειες και προαιρετικά συμβόλαια υποστήριξης σε λογικές τιμές. Η MySQL είναι τόσο δημοφιλής στην αγορά των βάσεων δεδομένων όσο το Linux στην αγορά των λειτουργικών συστημάτων. Επιλέγεται όλο και συχνότερα ως το σύστημα διαχείρισης βάσεων δεδομένων για ιστότοπους, και όχι μόνο για μικρούς. Χρησιμοποιείται ήδη από μεγάλες εταιρίες με τεράστιες ποσότητες δεδομένων, όπως η Yahoo! και η NASA. [9]

2.1 Τι είναι μια βάση δεδομένων;

Ο όρος βάση δεδομένων είναι ίσως αρκετά ασαφής, καθώς μια βάση δεδομένων μπορεί να είναι μια λίστα διευθύνσεων σε κάποιο πρόγραμμα λογιστικού φύλλου (όπως το Excel), ή μπορεί να είναι τα αρχεία διαχείρισης μιας τηλεπικοινωνιακής εταιρίας στην οποία καταγράφονται μερικές εκατομμύρια κλήσεις ημερησίως, με ακριβής υπολογισμούς των χρεώσεων τους και παραγωγή των μηνιαίων λογαριασμών των πελατών τους. Μια απλή βάση δεδομένων μπορεί να λειτουργεί απομονωμένη τοπικά σε έναν υπολογιστή και για έναν χρήστη, ενώ άλλες μπορεί να προσπελαύνονται συγχρόνως από χιλιάδες χρήστες. Το μέγεθος μιας βάσης δεδομένων μπορεί να ποικίλλει από μερικά kilobyte έως μερικά terabyte. [9]

2.2 Σχέσεις, Συστήματα Διαχείρισης Βάσεων Δεδομένων, Διακομιστές και Πελάτες

Μια βάση δεδομένων είναι μια ταξινομημένη συλλογή δεδομένων, η οποία αποθηκεύεται κανονικά σε ένα η περισσότερα συσχετιζόμενα αρχεία. Τα δεδομένα είναι δομημένα σε πίνακες, μεταξύ των οποίων είναι δυνατόν να υπάρχουν αναφορές. Η ύπαρξη αυτών των σχέσεων μεταξύ των πινάκων οδήγησε στο να ονομαστούν οι βάσεις δεδομένων αυτού του είδους σχεσιακές βάσεις δεδομένων.

Η MySQL, η Oracle, ο Microsoft SQL server και η IBM DB2 είναι παραδείγματα συστημάτων διαχείρισης σχεσιακών βάσεων δεδομένων, τα οποία και παρέχουν προγράμματα για την διαχείριση σχεσιακών βάσεων δεδομένων. Μεταξύ των καθηκόντων αυτών των συστημάτων βρίσκεται, εκτός της ασφαλούς αποθήκευσης των δεδομένων, η επεξεργασία εντολών για την αναζήτηση, ανάλυση και ταξινόμηση των υπαρχόντων δεδομένων καθώς και η αποθήκευση νέων δεδομένων. Όλα αυτά θα πρέπει να μπορούν να λάβουν μέρος όχι μόνο σε έναν μεμονωμένο υπολογιστή, αλλά και εντός ενός δικτύου. Για τον λόγο αυτό ο υπολογιστής στον οποίο υπάρχει η βάση δεδομένων λέγεται και διακομιστής βάσης δεδομένων.

Κάθε πρόγραμμα που συνδέεται στην βάση δεδομένων λέγεται πελάτης βάσης δεδομένων. Οι πελάτες βάσεων δεδομένων έχουν στόχο την απλούστευση της χρήσης της βάσης δεδομένων για τον τελικό χρήστη, καθώς η απευθείας επικοινωνία με την βάση δεδομένων χρησιμοποιώντας εντολές είναι άβολη. Αντί αυτού, ο χρήστης έχει στην διάθεσή του πίνακες και άλλα στοιχεία ενός γραφικού περιβάλλοντος για την εύρεση ή προσθήκη πληροφοριών. Ένα πρόγραμμα πελάτης βάσης δεδομένων μπορεί να έχει υλοποιηθεί με διάφορους τρόπους, όπως για παράδειγμα με σελίδες HTML, παραδοσιακά παραθυρικά προγράμματα κτλ.

2.3 Πίνακες, Εγγραφές, Πεδία, Ερωτήματα, SQL, Ευρετήρια και Κλειδιά

Στους πίνακες όπου είναι δομημένα τα δεδομένα, κάθε γραμμή ενός πίνακα λέγεται εγγραφή. Η δομή μιας εγγραφής καθορίζεται από τον ορισμό του πίνακα. Κάθε στήλη του πίνακα αντιστοιχεί σε ένα πεδίο της εγγραφής. Για παράδειγμα, μια εγγραφή σε έναν πίνακα με διευθύνσεις θα μπορούσε να έχει

τα πεδία όνομα, επώνυμο, οδός κτλ. Κάθε πεδίο είναι κάποιου συγκεκριμένου τύπου ο οποίος και καθορίζει το είδος της πληροφορίας που μπορεί να αποθηκευτεί στο πεδίο αυτό, όπως ένας αριθμός συγκεκριμένης μορφής ή ένα string χαρακτήρων με προκαθορισμένο μέγιστο αριθμό χαρακτήρων.

Η περιγραφή μιας βάσης δεδομένων που αποτελείται από διάφορους πίνακες, με πεδία, σχέσεις και ευρετήρια λέγεται μοντέλο βάσης δεδομένων. Το μοντέλο αυτό καθορίζει την κατασκευή των δομών δεδομένων καθώς και τον τύπο των δεδομένων που θα αποθηκευτούν.

Οι πίνακες συνήθως δεν διατηρούν τις εγγραφές με κάποια συγκεκριμένη σειρά, αλλά με την σειρά που αποθηκευτήκαν. Παρόλα αυτά για την αποδοτική χρήση των δεδομένων είναι απαραίτητο αυτά τα αταξινόμητα δεδομένα να ταξινομηθούν σύμφωνα με ένα ή και περισσότερα κριτήρια. Για παράδειγμα μπορεί να θέλουμε μια λίστα των εγγραφών των πελατών μιας συγκεκριμένης πόλης, ταξινομημένη αλφαβητικά κατά επώνυμο, οι οποίοι να έχουν κάνει κάποια αγορά τον τελευταίο χρόνο. Για την δημιουργία τέτοιων λιστών χρησιμοποιούνται τα ερωτήματα. Το αποτέλεσμα ενός ερωτήματος είναι πάλι ένας πίνακας, ο οποίος όμως βρίσκεται στην RAM και όχι στον σκληρό δίσκο.

Για τον σχηματισμό των ερωτημάτων χρησιμοποιούνται εντολές SQL (Structured Query Language). Η γλώσσα αυτή έχει καθιερωθεί ως πρότυπο για τον σχηματισμό ερωτημάτων. Παρόλα αυτά οι κατασκευαστές των διάφορων συστημάτων διαχείρισης βάσεων δεδομένων επεκτείνουν την γλώσσα με διαφορετικούς τρόπους, μειώνοντας την συμβατότητα μεταξύ των διαφορετικών αυτών συστημάτων.

Όταν οι πίνακες γίνουν μεγάλοι, η ταχύτητα με την οποία μπορεί να απαντηθεί ένα ερώτημα, εξαρτάται σημαντικά από την ύπαρξη κάποιου κατάλληλου ευρετηρίου που να καθορίζει την σειρά των εγγραφών. Ένα ευρετήριο είναι ένας βοηθητικός πίνακας ο οποίος περιέχει πληροφορίες σχετικές μόνο με την σειρά των εγγραφών. Ένα ευρετήριο λέγεται αλλιώς και κλειδί.

Ένα ευρετήριο επιταχύνει την πρόσβαση στα δεδομένα, αλλά έχει και μειονεκτήματα. Κάθε ευρετήριο αυξάνει τον χώρο που είναι απαραίτητος στον σκληρό δίσκο για την αποθήκευση του αρχείου της βάσης δεδομένων. Επίσης το ευρετήριο πρέπει να ενημερώνεται κάθε φορά που τα δεδομένα αλλάζουν, κάτι που απαιτεί χρόνο. Έτσι ένα ευρετήριο μπορεί να επιταχύνει την

ανάγνωση των δεδομένων, αλλά απαιτεί περισσότερο χρόνο για την εισαγωγή και τροποποίηση των δεδομένων. Η συμβολή ενός ευρετηρίου στην απόδοση εξαρτάται από τον τρόπο χρήσης των δεδομένων της βάσης δεδομένων.

Μια ειδική περίπτωση ευρετηρίου είναι το κύριο ευρετήριο, ή κύριο κλειδί. Ένα κύριο κλειδί πρέπει να εξασφαλίζει την μοναδικότητα κάθε εγγραφής. Συχνά, χρησιμοποιείται για τον σκοπό αυτό ένας αύξων αριθμός. Τα κύρια κλειδιά παίζουν σημαντικό ρόλο στις σχεσιακές βάσεις δεδομένων και μπορούν να αυξήσουν σημαντικά την ταχύτητα πρόσβασης στα δεδομένα.

2.4 MySQL

Η MySQL είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων. Σύμφωνα με τους οπαδούς της, η MySQL είναι πιο γρήγορη, αξιόπιστη, φτηνή και γενικότερα καλύτερη από οποιοδήποτε άλλο σύστημα διαχείρισης βάσεων δεδομένων (συμπεριλαμβανόμενων και εμπορικών συστημάτων, όπως η Oracle και η DB2). Πολλοί αντίπαλοι της MySQL δεν είναι σύμφωνοι με αυτήν την άποψη και σε μερικές περιπτώσεις ισχυρίζονται ακόμα πως η MySQL δεν είναι καν σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων.

Όμως ο αριθμός των χρηστών της MySQL αυξάνεται συνεχώς και η πλειοψηφία αυτών είναι αρκετά ευχαριστημένοι με την MySQL. Παρόλα αυτά η MySQL δεν έχει κάποια από τα χαρακτηριστικά που θεωρούνται δεδομένα σε άλλα συστήματα διαχείρισης βάσεων δεδομένων.

2.4.1 Χαρακτηριστικά της MySQL

Μερικά από τα πιο σημαντικά χαρακτηριστικά της MySQL είναι:

- **Σύστημα Διαχείρισης Σχεσιακών Βάσεων Δεδομένων:** Όπως όλα σχεδόν τα υπόλοιπα συστήματα διαχείρισης βάσεων δεδομένων στην αγορά, η MySQL είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων.
- **Αρχιτεκτονική Πελάτη/Εξυπηρετητή (Client/Server):** Η MySQL είναι ένα σύστημα πελάτη/εξυπηρετητή. Υπάρχει ένας εξυπηρετητής βάσης δεδομένων (MySQL) και πιθανώς πολλοί πελάτες (προγράμματα

εφαρμογών), οι οποίοι επικοινωνούν με τον εξυπηρετητή, στέλνοντας ερωτήματα, αποθηκεύοντας αλλαγές κτλ. Οι πελάτες μπορεί να εκτελούνται στον ίδιο υπολογιστή, σε άλλο υπολογιστή ενός τοπικού δικτύου ή ακόμα και του Internet. Σχεδόν όλα τα δημοφιλή συστήματα διαχείρισης βάσεων δεδομένων (Oracle, Microsoft SQL Server, κτλ) είναι συστήματα πελάτη/εξυπηρετητή. Αντιθέτως, στα συστήματα εξυπηρετητών αρχείων (όπως η Microsoft Access, η dBase κτλ), η απόδοση παρουσιάζει σημαντική πτώση καθώς αυξάνει ο αριθμός των χρηστών.

- **Συμβατότητα SQL:** Η MySQL χρησιμοποιεί την γλώσσα βάσεων δεδομένων SQL (Structured Query Language). Η SQL είναι μια τυποποιημένη γλώσσα που χρησιμοποιείται για τον σχηματισμό ερωτημάτων, την ενημέρωση των δεδομένων και την διαχείριση της βάσης δεδομένων. Υπάρχουν πολλές διάλεκτοι της SQL, σχεδόν τόσες όσα και τα συστήματα διαχείρισης βάσεων δεδομένων που υπάρχουν. Η MySQL συμμορφώνεται στο παρών πρότυπο SQL (SQL:2003), παρόλα αυτά με σημαντικούς περιορισμούς και μεγάλο αριθμό επεκτάσεων. Μέσω της ρύθμισης συστήματος sql-mode, ο εξυπηρετητής μπορεί να συμπεριφερθεί σε μεγάλο βαθμό συμβατά με διάφορα συστήματα διαχείρισης βάσεων δεδομένων, όπως η IBM DB2 και η Oracle. Η ρύθμιση αυτή αλλάζει μόνο μερικές από τις συμβάσεις σύνταξης και δεν εξομοιώνει πλήρως τα άλλα συστήματα.
- **SubSELECTs:** Από την έκδοση 4.1 και μετά, η MySQL είναι σε θέση να επεξεργαστεί ένα ερώτημα της μορφής (αλλά και πολυάριθμων παραλλαγών της): SELECT * FROM pinakas1 WHERE x IN (SELECT y FROM pinakas2);
- **Όψεις (Views):** Οι όψεις σχετίζονται με κάποιο ερώτημα SQL το οποίο αντιμετωπίζεται ως ένα διακριτό στοιχείο της βάσης δεδομένων και προσφέρει μια συγκεκριμένη όψη της βάσης δεδομένων. Η MySQL υποστηρίζει τις όψεις από την έκδοση 5.0 και μετά.
- **Αποθηκευμένες διαδικασίες (Stored procedures):** Οι αποθηκευμένες διαδικασίες είναι κώδικας SQL που έχει αποθηκευτεί στο σύστημα διαχείρισης βάσεων δεδομένων. Γενικά χρησιμοποιούνται

για να απλοποιήσουν μερικά βήματα, όπως πχ κατά την εισαγωγή ή διαγραφή μιας εγγραφής. Για προγραμματιστές πελατών αυτό έχει το πλεονέκτημα του ότι δεν χρειάζεται να επεξεργαστούν τους πίνακες απευθείας, αλλά βασίζονται στις αποθηκευμένες διαδικασίες. Όπως συμβαίνει και με τις όψεις, οι αποθηκευμένες διαδικασίες βοηθούν στην διαχείριση μεγάλων συστημάτων διαχείρισης βάσεων δεδομένων, αλλά μπορούν επίσης να αυξήσουν την απόδοση του συστήματος Η MySQL υποστηρίζει τις αποθηκευμένες διαδικασίες από την έκδοση 5.0 και μετά.

- **Triggers:** Τα triggers είναι εντολές SQL που εκτελούνται αυτόματα από τον εξυπηρετητή, όταν λαμβάνουν μέρος διάφορες λειτουργίες της βάσης δεδομένων (όπως INSERT, UPDATE και DELETE). Η MySQL υποστηρίζει μερικώς τα triggers από την έκδοση 5.0 και μετά.
- **Unicode:** Η MySQL υποστηρίζει μια πληθώρα σετ χαρακτήρων από την έκδοση 4.1 και μετά, συμπεριλαμβανόμενων και των Latin-1, Latin-2 και Unicode (με την παραλλαγή UTF8 ή UCS2).
- **Διεπιφάνεια χρήστη:** Παρέχονται διάφορες βολικές διεπιφάνειες χρήστη για την διαχείριση ενός εξυπηρετητή MySQL.
- **Αναζήτηση πλήρους κειμένου (Full-text search):** Η αναζήτηση αυτού του τύπου απλουστεύει και επιταχύνει την αναζήτηση λέξεων που βρίσκονται μέσα σε ένα πεδίο κειμένου. Εάν ανατεθεί στην MySQL η αποθήκευση κειμένου (όπως πχ σε ομάδα συζήτησης στο Internet), μπορεί να χρησιμοποιηθεί η αναζήτηση πλήρους κειμένου για την υλοποίηση μίας αποδοτικής λειτουργίας αναζήτησης.
- **Αναπαραγωγή (Replication):** Η αναπαραγωγή καθιστά δυνατή την αντιγραφή των περιεχομένων μιας βάσης δεδομένων σε κάποιον αριθμό υπολογιστών. Η δημιουργία αντιγράφων των δεδομένων σε περισσότερους υπολογιστές αυξάνει την προστασία του συστήματος από πιθανές βλάβες, έτσι ώστε σε περίπτωση που κάποιος υπολογιστής σταματήσει να λειτουργεί ένας άλλος θα είναι σε θέση να τον αντικαταστήσει. Επίσης βελτιώνουν την ταχύτητα εκτέλεσης των ερωτημάτων βάσης δεδομένων.
- **Δοσοληψίες (Transactions):** Στα πλαίσια ενός συστήματος

διαχείρισης βάσεων δεδομένων, μια δοσοληψία σημαίνει την εκτέλεση περισσότερων εντολών ως ένα μπλοκ. Το σύστημα διαχείρισης βάσεων δεδομένων εξασφαλίζει πως θα εκτελεστούν επιτυχώς όλες οι εντολές ή αλλιώς καμία από αυτές. Αυτό ισχύει ακόμα και στην περίπτωση διακοπής ρεύματος, αποτυχίας του υπολογιστή ή οποιουδήποτε άλλου καταστροφικού γεγονότος που μπορεί να συμβεί κατά την διάρκεια μιας δοσοληψίας. Έτσι, για παράδειγμα, δεν μπορεί να συμβεί να αναληφθεί ένα ποσό χρημάτων από τον λογαριασμό Α χωρίς να κατατεθεί στον λογαριασμό Β λόγω κάποιου σφάλματος του συστήματος. Οι δοσοληψίες δίνουν επίσης στους προγραμματιστές την δυνατότητα της διακοπής μιας ακολουθίας εντολών οι οποίες έχουν ήδη εκτελεστεί (ένα είδος ανάκλησης). Σε πολλές περιπτώσεις αυτό οδηγεί σε σημαντική απλοποίηση της προγραμματιστικής διαδικασίας. Η MySQL υποστηρίζει τις δοσοληψίες εδώ και πολύ καιρό. Πρέπει να σημειωθεί πως η MySQL μπορεί να αποθηκεύσει τους πίνακες σε διάφορες διατάξεις. Ο προεπιλεγμένος τύπος πίνακα λέγεται MyISAM και αυτός ο τύπος δεν υποστηρίζει δοσοληψίες. Αλλά υπάρχουν αρκετοί επιπλέον τύποι πινάκων οι οποίοι υποστηρίζουν τις δοσοληψίες. Ο πιο διάσημος από αυτούς είναι ο τύπος InnoDB.

- **Περιορισμοί Ξένου Κλειδιού:** Αυτοί είναι κανόνες που εξασφαλίζουν πως δεν υπάρχουν αναφορές μεταξύ συνδεδεμένων πινάκων οι οποίες να μην οδηγούν πουθενά.
- **Λειτουργίες GIS:** Από την έκδοση 4.1 και μετά, η MySQL υποστηρίζει την αποθήκευση και επεξεργασία δυσδιάστατων γεωγραφικών δεδομένων. Έτσι η MySQL είναι κατάλληλη για εφαρμογές GIS (Geographical Information Systems).
- **Γλώσσες Προγραμματισμού:** Υπάρχει ένας μεγάλος αριθμός APIs (Application Programming Interfaces – Διασυνδέσεις Προγραμματισμού Εφαρμογών) και βιβλιοθηκών για την ανάπτυξη εφαρμογών MySQL. Για τον προγραμματισμό πελατών μπορούν να χρησιμοποιηθούν, μεταξύ άλλων, οι γλώσσες C, C++, Java, Perl, PHP, Python κτλ.
- **ODBC:** Η MySQL υποστηρίζει την Connector/ODBC διασύνδεση ODBC. Αυτό επιτρέπει σε όλες τις γλώσσες προγραμματισμού που

εκτελούνται στα λειτουργικά συστήματα Microsoft Windows (Delphi, Visual Basic κτλ) την πρόσβαση στην MySQL. Η διασύνδεση ODBC μπορεί να υλοποιηθεί και για το λειτουργικό σύστημα UNIX. Υποστηρίζεται επίσης και η πλατφόρμα .NET της Microsoft, χρησιμοποιώντας τον πάροχο ODBC ή την .NET διασύνδεση Connector/NET.

- **Ανεξαρτησία πλατφόρμας:** Δεν έχουν μόνο οι εφαρμογές πελάτες την δυνατότητα να εκτελούνται σε διάφορα λειτουργικά συστήματα. Η ίδια η MySQL μπορεί να εκτελεστεί σε έναν αριθμό λειτουργικών συστημάτων. Τα πιο σημαντικά είναι τα Apple Macintosh OS X, Linux, Microsoft Windows και UNIX (μαζί με τις διάφορες εκδόσεις του όπως τα FreeBSD, OpenBSD, Sun Solaris κτλ).
- **Ταχύτητα:** Η MySQL θεωρείται πως είναι ένα πολύ γρήγορο πρόγραμμα βάσης δεδομένων. Η ταχύτητα αυτή επιβεβαιώνεται από έναν μεγάλο αριθμό τεστ αποδόσεων (benchmark tests), παρόλο που τέτοιου είδους τεστ πρέπει να αντιμετωπίζονται με μεγάλη προσοχή και σκέψη, ανεξαρτήτως της πηγής από την οποία προέρχονται.

2.4.2 Περιορισμοί της MySQL

- Όταν η MySQL χρησιμοποιείται με τον προεπιλεγμένο τύπο πινάκων MyISAM, τότε το κλείδωμα (locking), δηλαδή το προσωρινό μπλοκάρισμα της πρόσβασης και της αλλαγής των δεδομένων της βάσης δεδομένων, βρίσκεται εντός λειτουργίας μόνο για ολόκληρους πίνακες (table locking – κλείδωμα πινάκων). Το πρόβλημα του κλειδώματος των πινάκων μπορεί να παρακαμφθεί υλοποιώντας πίνακες που μπορούν να χρησιμοποιηθούν σε δοσοληψίες, όπως πίνακες InnoDB, οι οποίοι υποστηρίζουν το κλείδωμα γραμμών (row locking).
- Όταν χρησιμοποιούνται πίνακες MyISAM, η MySQL δεν μπορεί να εκτελέσει hot backups, τα οποία είναι αντίγραφα ασφαλείας που δημιουργούνται κατά την διάρκεια της λειτουργίας χωρίς να μπλοκάρονται οι πίνακες με κλειδώματα. Η λύση και πάλι είναι οι πίνακες InnoDB, μόνο που σε αυτή τη περίπτωση η λειτουργία hot

backup είναι διαθέσιμη μόνο στη μορφή εμπορικού συμπληρώματος.

- Πολλά συστήματα διαχείρισης βάσεων δεδομένων προσφέρουν την δυνατότητα ορισμού προσαρμοσμένων τύπων δεδομένων. Η MySQL δεν υποστηρίζει αυτήν την δυνατότητα.
- Η MySQL δεν υποστηρίζει την απευθείας επεξεργασία δεδομένων XML. Διάφορα εμπορικά συστήματα διαχείρισης βάσεων δεδομένων προσφέρουν σημαντικά περισσότερη λειτουργικότητα σε αυτόν τον τομέα.
- Η MySQL είναι ένα πολύ γρήγορο σύστημα διαχείρισης βάσεων δεδομένων αλλά είναι πολύ περιορισμένο όσον αφορά την χρησιμότητα του σε εφαρμογές πραγματικού χρόνου και δεν υποστηρίζει συναρτήσεις OLAP (Online Analytical Processing). Το OLAP αναφέρεται σε ειδικές μεθόδους για την διαχείριση και ανάλυση πολυδιάστατων δεδομένων. Τα συστήματα διαχείρισης βάσεων δεδομένων που έχουν την ικανότητα OLAP λέγονται συχνά αποθήκες δεδομένων (data warehouses).
- Η MySQL υποστηρίζει, από την έκδοση 5.0 και μετά, τις αποθηκευμένες διαδικασίες και τα triggers, αλλά οι λειτουργίες αυτές δεν έχουν ωριμάσει πλήρως και δεν παρουσιάζουν την ίδια σταθερότητα και ποικιλία λειτουργιών που παρέχουν τα εμπορικά συστήματα διαχείρισης βάσεων δεδομένων.
- Παρόμοιοι περιορισμοί ισχύουν και για τις λειτουργίες GIS που εισήχθησαν στην έκδοση 4.1. Τα εμπορικά συστήματα διαχείρισης βάσεων δεδομένων προσφέρουν σε μερικές περιπτώσεις σημαντικά μεγαλύτερη λειτουργικότητα.

2.4.3 Δικαιώματα Άδειας Χρήσης της MySQL

Ένα ακόμα από τα χαρακτηριστικά της MySQL που έχει ενδιαφέρον είναι η άδεια χρήσης. Η MySQL είναι ένα πρόγραμμα ανοιχτού λογισμικού. Αυτό σημαίνει πως όλος ο πηγαίος κώδικας της MySQL διατίθεται ελεύθερα. Από τον Ιούνιο του 2000 και μετά (δηλαδή από την έκδοση 3.23.19 και μετά) η MySQL διατίθεται υπό την άδεια GNU Public License (GPL). Έτσι εξασφαλίζεται πως η MySQL θα συνεχίσει να διατίθεται ελεύθερα υπό την έννοια του ανοιχτού λογισμικού. Για εμπορικές εφαρμογές της MySQL

διατίθεται μία εμπορική άδεια.

Χρήση της MySQL με Άδεια Χρήσης Ανοιχτού Λογισμικού

Η παρακάτω λίστα περιγράφει τις διαφορετικές περιπτώσεις στις οποίες κάποιος μπορεί να χρησιμοποιήσει την MySQL ελεύθερα, υπό την έννοια της άδειας GPL:

- Η MySQL μπορεί να χρησιμοποιηθεί δωρεάν, εάν μια εφαρμογή που έχει αναπτυχθεί δεν πρόκειται να χρησιμοποιηθεί εμπορικά. Μόνο όταν πρόκειται να πουληθεί η εφαρμογή σε πελάτες γίνεται απαραίτητη η απόκτηση άδειας. Ο κανόνας αυτός διατυπώνεται στην ιστοσελίδα της MySQL ως: «Ελεύθερη χρήση για όσους δεν αντιγράφουν, παραλλάσσουν ή διανέμουν ποτέ.»
- Η MySQL μπορεί να χρησιμοποιηθεί δωρεάν εντός ενός ιστότοπου. Επίσης μια εφαρμογή που έχει αναπτυχθεί σε PHP και εγκατασταθεί στον πάροχο υπηρεσίας διαδικτύου, δεν είναι απαραίτητο να διατεθεί ελεύθερα ο κώδικας PHP υπό την έννοια της άδειας GPL.
- Παρομοίως, ένας πάροχος υπηρεσίας διαδικτύου μπορεί να διαθέσει την MySQL στους πελάτες του χωρίς να χρειάζεται να πληρώσει τέλη για μια άδεια MySQL. Αυτό συμβαίνει διότι από την στιγμή που η MySQL εκτελείται αποκλειστικά στον υπολογιστή του παρόχου, η εφαρμογή θεωρείται εσωτερική.
- Τέλος, η MySQL μπορεί να χρησιμοποιηθεί δωρεάν σε όλα τα προγράμματα τα οποία εκτελούνται υπό την άδεια GPL ή συγκρίσιμη ελεύθερη άδεια.

Χρήση της MySQL με Εμπορική Άδεια Χρήσης

Υπό την έννοια της άδειας GPL η ακόλουθες χρήσεις απαγορεύονται:

- Απαγορεύεται η αλλαγή ή επέκταση της MySQL ή η πώληση της καινούργιας έκδοσης ή προϊόντος χωρίς παράλληλα να διατίθεται ελεύθερα ο πηγαίος κώδικας των αλλαγών. Αυτό σημαίνει πως απαγορεύεται η ανάπτυξη ενός νέου συστήματος διαχείρισης βάσεων δεδομένων βασιζόμενου στην MySQL εάν οι επεκτάσεις δεν διατεθούν ελεύθερα στο κοινό της MySQL υπό την έννοια της άδειας GPL.

- Απαγορεύεται η ανάπτυξη ενός εμπορικού προγράμματος το οποίο χρησιμοποιεί την MySQL ως βάση δεδομένων, εάν ο κώδικας δεν είναι διαθέσιμος υπό την έννοια του ανοιχτού λογισμικού.

Εάν οι περιορισμοί της άδειας GPL δεν είναι αποδεκτοί για κάποιον που θέλει να παράγει εμπορικό λογισμικό, τότε μπορεί να πουλήσει το προϊόν (πρόγραμμα) με μία εμπορική άδεια MySQL. Με τον τρόπο αυτό η MySQL επιτρέπεται να χρησιμοποιηθεί ακόμα και όταν ο κώδικας δεν είναι διαθέσιμος υπό την έννοια της άδειας GPL.

2.4.4 Τρόπος λειτουργίας του συστήματος πρόσβασης

Έλεγχος πρόσβασης δύο επιπέδων (Two-Tiered)

Ο έλεγχος της πρόσβασης στην MySQL χωρίζεται σε δύο επίπεδα. Στο πρώτο επίπεδο γίνεται έλεγχος για το αν ο χρήστης έχει το δικαίωμα να συνδεθεί στην MySQL ή όχι. Αυτό επιτυγχάνεται ελέγχοντας το όνομα χρήστη (user name), το όνομα (ή διεύθυνση IP) του υπολογιστή του χρήστη (host name) και το συνθηματικό του χρήστη (password).

Μόνο εάν μπορέσει να δημιουργηθεί μια σύνδεση περνάει ο έλεγχος στο δεύτερο επίπεδο, το οποίο αφορά κάθε εντολή βάσης δεδομένων. Για παράδειγμα, όταν εκτελείται μια SELECT, η MySQL ελέγχει εάν ο χρήστης έχει δικαίωμα πρόσβασης στην βάση, τον πίνακα και την στήλη. Εάν εκτελεστεί μια INSERT, τότε η MySQL ελέγχει εάν ο χρήστης έχει το δικαίωμα να τροποποιήσει την βάση, τον πίνακα και τελικά την στήλη.

Δικαιώματα (Privileges)

Η MySQL χρησιμοποιεί πίνακες στους οποίους αποθηκεύονται τα δικαιώματα. Εάν ένας χρήστης έχει το δικαίωμα SELECT για μια συγκεκριμένη βάση, τότε μπορεί να προσπελάσει τα δεδομένα της βάσης (αλλά όχι να τα τροποποιήσει). Εάν ο χρήστης έχει ένα global δικαίωμα SELECT, τότε έχει το δικαίωμα να προσπελάσει τα δεδομένα όλων των βάσεων δεδομένων που είναι αποθηκευμένων στην MySQL.

Τα δικαιώματα που αναγνωρίζονται φαίνονται στους πίνακες 2-1,2-2 και 2-3. Αξίζει να σημειωθεί πως οι αντίστοιχες στήλες των πινάκων mysql τελειώνουν

πάντα με το επίθεμα _priv. Για παράδειγμα το δικαίωμα Select είναι αποθηκευμένο στην στήλη Select_priv. Μερικές φορές τα ονόματα των στηλών γράφονται με συντομογραφίες, όπως π.χ. Create_tmp_table για το δικαίωμα Create Temporary Table.

Πίνακας 2-1: Δικαιώματα MySQL

Δικαίωμα MySQL	**Περιγραφή**
	Για την πρόσβαση σε πίνακες
Select	Μπορούν να προσπελαστούν τα δεδομένα (εντολή SELECT).
Insert	Μπορούν να αποθηκευτούν νέες εγγραφές (INSERT).
Update	Μπορούν να τροποποιηθούν υπάρχουσες εγγραφές (UPDATE).
Delete	Μπορούν να διαγραφούν υπάρχουσες εγγραφές (DELETE).
Lock Tables	Μπορούν να κλειδωθούν πίνακες (LOCK).

Πίνακας 2-2: Δικαιώματα MySQL

Δικαίωμα MySQL	**Περιγραφή**
Για την τροποποίηση Βάσεων Δεδομένων, Πινάκων και Όψεων	
Create	Μπορεί να δημιουργήσει καινούργιες βάσεις δεδομένων και πίνακες.
Create Temporary Table	Μπορεί να δημιουργήσει προσωρινούς πίνακες.
Alter	Μπορεί να μετονομάσει πίνακες και να αλλάξει την δομή τους.
Index	Μπορεί να δημιουργήσει και να διαγράψει ευρετήρια πινάκων.
References	Ατεκμηρίωτο. Ίσως μελλοντικά να επιτρέπει συνδέσμους μεταξύ πινάκων.
Drop	Μπορεί να διαγράψει υπάρχουσες βάσεις δεδομένων και πίνακες.
Create View	Μπορεί να ορίσει όψεις (από την MySQL 5.0 και μετά).
Show View	Μπορεί να επιθεωρήσει τον ορισμό των όψεςν με την εντολή SHOW CREATE VIEW (από την MySQL 5.0 και μετά).
Για Αποθηκευμένες Διαδικασίες (από την MySQL 5.0 και μετά)	
Alter Routine	Μπορεί να τροποποιήσει υπάρχουσες αποθηκευμένες διαδικασίες.
Create Routine	Μπορεί να ορίσει καινούργιες αποθηκευμένες διαδικασίες.
Execute	Μπορεί να εκτελέσει αποθηκευμένες διαδικασίες.
Για την Πρόσβαση Δεδομένων	
File	Μπορεί να προσπελάσει και να τροποποιήσει αρχεία του τοπικού συστήματος αρχείων.
Create User	Μπορεί να δημιουργήσει καινούργιους χρήστες MySQL (από την MySQL 5.0.3 και μετά).

Πίνακας 2-3: Δικαιώματα MySQL

Δικαίωμα MySQL	**Περιγραφή**
	Για την Διαχείριση της MySQL
Grant Option	Μπορεί να δώσει σε χρήστες τα δικαιώματα που έχει.
Show Databases	Μπορεί να δει μια λίστα όλων των βάσεων δεδομένων (SHOW DATABASES).
Process	Μπορεί να δει τις διεργασίες MySQL άλλων χρηστών (SHOW PROCESSLIST).
Super	Μπορεί να τερματίσει τις διεργασίες MySQL άλλων χρηστών (KILL), και να τροποποιήσει και να εκτελέσει μερικές εντολές διαχείρισης (CHANGE/PURGE MASTER, SET GLOBAL).
Reload	Μπορεί να εκτελέσει διάφορες εντολές (reload, refresh, flush-xxx).
Replication Client	Μπορεί να εξακριβώσει πληροφορίες για τους συμμετέχοντες σε ένα σύστημα αναπαραγωγής.
Replication Slave	Μπορεί να προσπελάσει τα δεδομένα του εξυπηρετητή MySQL μέσω αναπαραγωγής.
Shutdown	Μπορεί να τερματίσει την MySQL.

Το δικαίωμα Grant Option

Το δικαίωμα Grant Option σημαίνει πως ένας χρήστης MySQL μπορεί να δώσει δικαιώματα σε άλλους χρήστες. Όμως η ικανότητα αυτή περιορίζεται στα δικαιώματα τα οποία ο ίδιος έχει. Έτσι κανένας χρήστης δεν μπορεί να δώσει δικαιώματα σε κάποιον άλλον, τα οποία να μην έχει ήδη ο ίδιος.

Το δικαίωμα File

Οι χρήστες MySQL με το δικαίωμα File μπορούν να χρησιμοποιήσουν εντολές SQL για απευθείας πρόσβαση στο σύστημα αρχείων του υπολογιστή στον οποίο εκτελείται ο εξυπηρετητής MySQL, για παράδειγμα, με την εντολή “SELECT ... INTO OUTFILE όνομα_αρχείου“ με την εντολή LOAD DATA ή την συνάρτηση LOAD_FILE.

Στην περίπτωση της πρόσβασης σε αρχεία, είναι απαραίτητο να δοθεί προσοχή στα δικαιώματα πρόσβασης στο αρχείο συστήματος. Σε συστήματα Unix/Linux ο εξυπηρετητής MySQL εκτελείται από τον λογαριασμό mysql. Για τον λόγο αυτό μόνο τα αρχεία που είναι προσπελάσιμα από τον Unix/Linux χρήστη mysql μπορούν να προσπελαστούν. Παρόλα αυτά το δικαίωμα File είναι συνήθως ένα αξιόλογο ρίσκο για την ασφάλεια.

Global δικαιώματα εναντίων δικαιωμάτων Object

Στην MySQL τα δικαιώματα μπορούν να είναι global ή συσχετισμένα με κάποιο συγκεκριμένο αντικείμενο. Global σημαίνει πως το δικαίωμα ισχύει για όλα τα αντικείμενα MySQL (βάσεις δεδομένων, πίνακες και στήλες πινάκων).
Η διαχείριση δικαιωμάτων συσχετισμένων με αντικείμενα είναι πιο δύσκολη, αλλά και πιο ασφαλής. Μόνο με τον τρόπο αυτό μπορεί να επιτευχθεί ένας χρήστης MySQL να μπορεί να τροποποιήσει έναν συγκεκριμένο πίνακα και όχι όλους τους πίνακες της MySQL. Μια προϋπόθεση κατά την χρήση των δικαιωμάτων Object πως τα αντίστοιχα δικαιώματα Global δεν έχουν τεθεί. Κάτι που απαγορεύεται καθολικά δεν μπορεί να επιτρέπεται σε επίπεδο αντικειμένων.

Η βάση δεδομένων mysql

Για την διαχείριση των δικαιωμάτων πρόσβασης η MySQL χρησιμοποιεί μια βάση δεδομένων με το όνομα mysql. Η βάση αυτή αποτελείται από διάφορους πίνακες, υπεύθυνους για διάφορες πτυχές των δικαιωμάτων πρόσβασης.

Οι πίνακες της βάσης δεδομένων MySQL

Η βάση δεδομένων mysql περιέχει έναν μεγάλο αριθμό πινάκων για διάφορες εργασίες διαχείρισης. Στον πίνακα 2-4 φαίνονται οι έξι πίνακες που χρησιμοποιούνται για την διαχείριση των δικαιωμάτων πρόσβασης.

Πίνακας 2-4: Πίνακες της mysql για την διαχείριση δικαιωμάτων πρόσβασης

Όνομα	**Περιγραφή**
user	Ελέγχει ποιος (όνομα χρήστη) έχει πρόσβαση στην MySQL και από ποιόν υπολογιστή (όνομα υπολογιστή). Αυτός ο πίνακας περιέχει επίσης δικαιώματα Global.
db	Καθορίζει ποιοι χρήστες έχουν πρόσβαση σε ποιες βάσεις δεδομένων.
host	Επεκτείνει τον πίνακα db με πληροφορίες για τα ονόματα υπολογιστών που είναι επιτρεπτά (αυτά που δεν υπάρχουν στον πίνακα db).
tables_priv	Καθορίζει ποιος έχει πρόσβαση σε ποιους πίνακες μιας βάσης δεδομένων.
columns_priv	Καθορίζει ποιος έχει πρόσβαση σε ποιες στήλες ενός πίνακα.
func	Ενεργοποιεί την διαχείριση UDFs (User Defined Functions). Αυτό είναι ακόμα ατεκμηρίωτο.
procs_priv	Καθορίζει ποιός επιτρέπεται να εκτελέσει αποθηκευμένες διαδικασίες.

Στο σύστημα πρόσβασης δύο επιπέδων της MySQL, ο πίνακας user είναι υπεύθυνος για το πρώτο επίπεδο (για την σύνδεση με την MySQL). Ο πίνακας user περιέχει όλα τα global δικαιώματα.

Για το δεύτερο επίπεδο (πρόσβαση σε συγκεκριμένα αντικείμενα: βάσεις δεδομένων, πίνακες και στήλες) υπεύθυνοι είναι οι πίνακες db, host, tables_priv και columns_priv μαζί με τον πίνακα user.

Οι πίνακες user, db, host, tables_priv και columns_priv ελέγχονται με αυτήν την σειρά. Αυτό σημαίνει πως εάν δοθεί το δικαίωμα Select σε έναν χρήστη στον πίνακα user, τότε δεν θα ληφθούν υπόψη οι υπόλοιποι τέσσερις πίνακες στον έλεγχο της πρόσβασης μίας εντολής SELECT που θα εκτελέσει ο συγκεκριμένος χρήστης. Το αποτέλεσμα είναι πως εάν πρόκειται να γίνουν λεπτομερείς εξαιρέσεις στα δικαιώματα χρήσης, τα global δικαιώματα του πίνακα user πρέπει να έχουν τεθεί στο N.

2.4.5 Java JDBC και Connector/J

Ο Connector/J είναι ένας οδηγός JDBC για την MySQL. Το JDBC είναι μια συλλογή τάξεων που βοηθούν στον προγραμματισμό εφαρμογών βάσεων δεδομένων Java. Το JDBC είναι ανεξάρτητο των συστημάτων διαχείρισης βάσεων δεδομένων. Για τον λόγο αυτό, για να δημιουργηθεί μια σύνδεση με ένα σύστημα διαχείρισης βάσεων δεδομένων μέσω του JDBC, θα πρέπει να χρησιμοποιηθεί ένας οδηγός για την συγκεκριμένη βάση δεδομένων. Για το σύστημα διαχείρισης βάσεων δεδομένων ο οδηγός αυτός είναι ο Connector/J.
Ο Connector/J είναι ένας οδηγός τύπου 4. Αυτό σημαίνει πως έχει υλοποιηθεί το πρωτόκολλο MySQL πλήρως σε Java και δεν βασίζεται στις βιβλιοθήκες πελάτη της MySQL. Διατίθενται διαφορετικές εκδόσεις οι οποίες είναι συμβατές με τις προδιαγραφές JDBC 3.0 και JDBC 4.0.

Εγκατάσταση του Connector/J

Το πακέτο Connector/J, μπορεί να εγκατασταθεί χρησιμοποιώντας την δυαδική (binary) έκδοση ή την έκδοση πηγαίου κώδικα. Η έκδοση πηγαίου κώδικα επιτρέπει την περεταίρω παραμετροποίηση της εγκατάστασης. Και στις δύο περιπτώσεις θα πρέπει να προστεθεί χειροκίνητα η τοποθεσία του Connector/J στο Java CLASSPATH.

Εγκατάσταση του Connector/J χρησιμοποιώντας την Binary έκδοση

Ο πιο εύκολος τρόπος εγκατάστασης είναι χρησιμοποιώντας την Binary έκδοση του πακέτου Connector/J. Η Binary έκδοση είναι διαθέσιμη σε αρχείο Tar/Gzip ή Zip το οποίο πρέπει να αποσυμπιεστεί σε κάποια κατάλληλη τοποθεσία και έπειτα προαιρετικά να δηλωθεί η τοποθεσία αυτή στο CLASSPATH.
Στο αρχείο tar.gz ή zip περιέχονται τα αρχεία με τον πηγαίο κώδικα, τα αρχεία τάξεων και το αρχείο JAR με το όνομα mysql-connector-java-[έκδοση]-bin.jar.
Αφού αποσυμπιεστούν τα περιεχόμενα μπορεί να εγκατασταθεί τοποθετώντας το αρχείο mysql-connector-java-[έκδοση]-bin.jar στο classpath, είτε προσθέτοντας το πλήρες μονοπάτι (path) στην μεταβλητή περιβάλλοντος CLASSPATH, ή προσδιορίζοντας την θέση του κατευθείαν στην γραμμή εντολών χρησιμοποιώντας τον διακόπτη –cp κατά την εκκίνηση της JVM.

Εάν πρόκειται να χρησιμοποιηθεί ο οδηγός με τον JDBC DriverManager, τότε η τάξη java.sql.Driver υλοποιείται από την τάξη com.mysql.jdbc.Driver.

Java, JDBC και τύποι MySQL

Ο MySQL Connector/J είναι ελαστικός στον τρόπο με τον οποίο διαχειρίζεται μετατροπές μεταξύ τύπων δεδομένων MySQL και τύπων δεδομένων Java.
Γενικά, κάθε τύπος δεδομένων MySQL μπορεί να μετατραπεί σε ένα String της Java και κάθε αριθμητικός τύπος μπορεί να μετατραπεί σε οποιονδήποτε αριθμητικό τύπο της Java, αν και είναι πιθανό να συμβεί στρογγυλοποίηση ή overflow και να οδηγήσει σε έλλειψη ακρίβειας.
Από την έκδοση Connector/J 3.10 και μετά, ο οδηγός JDBC προειδοποιεί ή προκαλεί εξαιρέσεις DataTruncation όπως απαιτείται από την προδιαγραφή JDBC. Στον πίνακα 2-5 φαίνονται οι μετατροπές που είναι εγγυημένο πως θα λειτουργήσουν.

Πίνακας 2-5: Μετατροπές μεταξύ τύπων δεδομένων MySQL και Java [11]

Τύποι δεδομένων MySQL	**Μπορούν να μετατραπούν πάντα στους εξής τύπους δεδομένων Java**
CHAR, VARCHAR, BLOB, TEXT, ENUM, SET	java.lang.String, java.io.InputStream, java.io.Reader, java.sql.Blob, java.sql.Clob
FLOAT, REAL, DOUBLE PRECISION, NUMERIC, DECIMAL, TINYINT, SMALLINT, MEDIUMINT, INTEGER, BIGINT	java.lang.String, java.lang.Short, java.lang.Integer, java.lang.Long, java.lang.Double, java.math.BigDecimal
DATE, TIME, DATETIME, TIMESTAMP	java.lang.String, java.sql.Date, java.sql.Timestamp

Η μέθοδος ResultSet.getObject() χρησιμοποιεί μετατροπές τύπων, μεταξύ τύπων δεδομένων MySQL και Java, που ακολουθούν την προδιαγραφή JDBC. Στούς πίνακες 2-6 και 2-7 φαίνονται οι τάξεις Java με τις οποίες

επιστρέφονται οι τύποι MySQL καθώς και η τιμή που επιστρέφει η μέθοδος ResultSetMetaData.GetColumnClassName.

Πίνακας 2-6: Μετατροπή τύπων MySQL σε τύπους Java για την μέθοδο ResultSet.getObject()

Όνομα τύπου MySQL	**Τιμή επιστροφής μεθόδου GetColumnClassName**	**Τάξη Java με την οποία επιστρέφει**
BIT(1) (νέο στην MySQL-5.0)	BIT	java.lang.Boolean
BIT(> 1) (νέο στην MySQL-5.0)	BIT	byte[]
TINYINT	TINYINT	java.lang.Boolean εάν η ιδιότητα tinyInt1isBit είναι true (είναι η προεπιλογή) και το μέγεθος 1 ή αλλιώς java.lang.Integer.
BOOL, BOOLEAN	TINYINT	Όπως στο TINYINT, προς το παρών αυτά είναι ψευδώνυμα του TINYINT(1).
SMALLINT[(M)] [UNSIGNED]	SMALLINT [UNSIGNED]	java.lang.Integer (ανεξάρτητα από το αν είναι UNSIGNED η όχι)
MEDIUMINT[(M)] [UNSIGNED]	MEDIUMINT [UNSIGNED]	java.lang.Integer, εάν είναι UNSIGNED τότε java.lang.Long (C/J 3.1 και παλιότερα), ή java.lang.Integer για C/J 5.0 και μετά.
INT,INTEGER[(M)] [UNSIGNED]	INTEGER [UNSIGNED]	java.lang.Integer, εάν είναι UNSIGNED τότε java.lang.Long
BIGINT[(M)] [UNSIGNED]	BIGINT [UNSIGNED]	java.lang.Long, εάν είναι UNSIGNED τότε java.math.BigInteger
FLOAT[(M,D)]	FLOAT	java.lang.Float
DOUBLE[(M,B)]	DOUBLE	java.lang.Double

Πίνακας 2-7: Μετατροπή τύπων MySQL σε τύπους Java

DECIMAL[(M[,D])]	DECIMAL	java.math.BigDecimal
DATE	DATE	java.sql.Date
DATETIME	DATETIME	java.sql.Timestamp
TIMESTAMP[(M)]	TIMESTAMP	java.sql.Timestamp
TIME	TIME	java.sql.Time
YEAR[(2|4)]	YEAR	Εάν η ιδιότητα yearIsDateType είναι false, τότε ο επιστρεφόμενος τύπος αντικειμένου είναι java.sql.Short. Εάν είναι true (είναι η προεπιλογή) τότε επιστρέφεται ένα αντικείμενο του τύπου java.sql.Date (με την ημερομηνία ρυθμισμένη στις 1 Ιανουαρίου και μεσάνυχτα).
CHAR(M)	CHAR	java.lang.String (εάν το σετ χαρακτήρων της στήλης είναι BINARY, τότε επιστρέφεται byte[].)
VARCHAR(M) [BINARY]	VARCHAR	java.lang.String (εάν το σετ χαρακτήρων της στήλης είναι BINARY, τότε επιστρέφεται byte[].)
BINARY(M)	BINARY	byte[]
VARBINARY(M)	VARBINARY	byte[]
TINYBLOB	TINYBLOB	byte[]
TINYTEXT	VARCHAR	java.lang.String
BLOB	BLOB	byte[]
TEXT	VARCHAR	java.lang.String
MEDIUMBLOB	MEDIUMBLOB	byte[]
MEDIUMTEXT	VARCHAR	java.lang.String
LONGBLOB	LONGBLOB	byte[]
LONGTEXT	VARCHAR	java.lang.String
ENUM('val1','val2',...)	CHAR	java.lang.String
SET('val1','val2',...)	CHAR	java.lang.String

Κεφάλαιο 3: Netbeans

Εισαγωγή

Αρχικά, περίπου κατά την αρχή/μέση του 1990 όπου η Java άρχισε να γίνεται δημοφιλής, ο κώδικας γραφόταν χρησιμοποιώντας απλά εργαλεία επεξεργασίας κειμένου όπως το Notepad. Συνδυάζοντας έναν επεξεργαστή κειμένου και την γραμμή εντολών, οι χρήστες μπορούσαν να γράψουν και να μεταγλωττίσουν τον κώδικα.

Σύντομα διαπιστώθηκε πως η προσέγγιση αυτή δεν παρείχε το καλύτερο περιβάλλον ανάπτυξης. Για παράδειγμα εάν κάποιος έκανε ένα συντακτικό λάθος στον επεξεργαστή κειμένου, δεν υπήρχε τρόπος αυτό να αναγνωριστεί μέχρι να αποθηκευτεί και μεταγλωττιστεί το αρχείο. Τα επόμενα βήματα για κάποιον που ακολουθεί αυτή τη μέθοδο, ήταν η επιθεώρηση των σφαλμάτων της μεταγλώττισης, η εύρεση της γραμμής που προκαλεί το σφάλμα και ο προσδιορισμός της αιτίας. Τα σφάλματα της μεταγλώττισης δεν εξυπηρετούν πάντα στην διάγνωση του προβλήματος του κώδικα.

Πολλοί αρχάριοι προγραμματιστές ξεκινούν χρησιμοποιώντας έναν επεξεργαστή κειμένου και κάποιο περιβάλλον γραμμής εντολών. Δεν υπάρχει κάποιο λάθος στην προσέγγιση αυτή, καθώς ακόμα και μερικοί επαγγελματίες κάνουν το ίδιο πράγμα. Για κάποιον αρχάριο που μαθαίνει την Java, η χρήση ενός απλού επεξεργαστή κειμένου μπορεί μερικές φορές να είναι η ευκολότερη και γρηγορότερη προσέγγιση. Παρόλα αυτά οι επεξεργαστές κειμένου δεν προσφέρουν βοήθεια για την σύνταξη της γλώσσας, την ολοκλήρωση του Compliler, την υποστήριξη ευφυούς αναδόμησης ή άλλες δυνατότητες εγγραφής κώδικα.

Ένα από τα χρήσιμα χαρακτηριστικά που έχουν οι επεξεργαστές κειμένου είναι η Εύρεση και Αντικατάσταση (Find and Replace). Με αυτήν την απλή ικανότητα, οι προγραμματιστές μπορούσαν να αντικαταστήσουν μια λέξη η φράση (όσες φορές εμφανίζεται στο κείμενο) με κάποια άλλη. Αυτό λειτουργούσε σε κάποιες περιπτώσεις, αλλά μπορούσε να δημιουργήσει και προβλήματα.

Μερικοί επεξεργαστές κειμένου προσφέρουν πιο προηγμένη υποστήριξη για γλώσσες προγραμματισμού. Το δημοφιλές εργαλείο Emacs, που βασίζεται

στο Unix, προσφέρει πολλά ενδιαφέροντα χαρακτηριστικά, όπως προηγμένη ταύτιση και αντικατάσταση κειμένου. Μέσω επεκτάσεων (plugins) μπορεί ακόμα να προσφέρει επισήμανση της σύνταξης της Java, δόμηση του κώδικα, βασική αποσφαλμάτωση (debugging) και υποστήριξη μεταγλώττισης. Όλα αυτά είναι σημαντικά χαρακτηριστικά αλλά και πάλι δεν προσφέρουν το πιο ελαστικό και παραγωγικό περιβάλλον.

Μια ερώτηση που μπορεί να έχει κάποιος ο οποίος χρησιμοποιεί έναν επεξεργαστή κειμένου, είναι γιατί να χρησιμοποιήσει ένα IDE. Μερικοί προγραμματιστές τείνουν να δεσμεύονται με ένα συγκεκριμένο σετ εργαλείων ή γλώσσα προγραμματισμού και αντιστέκονται στις αλλαγές. Είναι όμως σημαντική, στον σημερινό κόσμο όπου τα πράγματα αλλάζουν συνεχώς, η ικανότητα της προσαρμογής σε καινούργιες τεχνολογίες.

Καινούργια σετ εργαλείων μπορούν να βοηθήσουν τους προγραμματιστές με πολλούς τρόπους. Ο χρόνος ενός προγραμματιστή θα πρέπει να καταναλώνεται στην παραγωγή και την δοκιμή του κώδικα. Δεν θα έπρεπε να χάνεται χρόνος για την μετονομασία μεθόδων στον κώδικα, την παραγωγή τεκμηρίωσης (documentation) του project, ή την μεταγλώττιση όλων των τάξεων εντός ενός πακέτου. Από την στιγμή που θα προσδιοριστεί η δράση που πρέπει να εκτελεστεί, το εργαλείο θα πρέπει να μπορεί να το κάνει εύκολα.

Τα ολοκληρωμένα περιβάλλοντα ανάπτυξης λογισμικού (IDE – Integrated Development Environments) κυριολεκτικά παρέχουν ένα ολόκληρο περιβάλλον εργασίας. Συνδυάζουν πολλά διαφορετικά εργαλεία με συνεκτικό τρόπο, έτσι ώστε υπηρεσίες και λειτουργίες που χρειάζονται να ενσωματώνονται απρόσκοπτα. [10]

Μερικά από τα τεχνικά πλεονεκτήματα των IDEs είναι τα παρακάτω:

- Γραφικό περιβάλλον χρήσης (GUI) για την εκτέλεση λειτουργιών
- Ομαδοποίηση του πηγαίου κώδικα και των αρχείων ρυθμίσεων υπό την έννοια ενός project
- Στενή ενσωμάτωση με τον μεταγλωττιστή
- Σύνδεση με αποθήκη πηγαίου κώδικα
- Δυνατότητα συντονισμού επίδοσης, ανάλυσης και φόρτωσης του κώδικα

- Δυνατότητα χρήσης εργαλείων και επεκτάσεων τρίτων κατασκευαστών
- Ικανότητα της αποσφαλμάτωσης του κώδικα εκτελώντας μια γραμμή την φορά
- Γρήγορη πρόσβαση και εύκολη παραγωγή τεκμηρίωσης του project

Μερικά από τα πλεονεκτήματα της χρήσης ενός IDE σε ένα εμπορικό περιβάλλον είναι τα παρακάτω:

- Μειώνει τον χρόνο της ανάπτυξης
- Αυξάνει την ποιότητα και την αξιοπιστία του κώδικα
- Τυποποιεί την διαδικασία ανάπτυξης του λογισμικού
- Παρέχει μια κοινή πλατφόρμα για τους προγραμματιστές του προσωπικού μειώνοντας τον χρόνο εκπαίδευσης

Μερικά από αυτά τα πλεονεκτήματα είναι σίγουρα συζητήσιμα και μπορούν να πραγματοποιηθούν μόνο μετά από προσεκτική ανάλυση, υλοποίηση και εκτέλεση. Εμπλέκονται πολλοί παράγοντες, αλλά ένα πραγματικά καλό εργαλείο IDE μπορεί να αποτελέσει τα θεμέλια για την επίτευξη σημαντικών στόχων, όπως αυτών που αναφέρθηκαν παραπάνω.
Μερικοί από τους πιο σημαντικούς λόγους για τους οποίους οι προγραμματιστές θα έπρεπε να χρησιμοποιούν το NetBeans είναι:

- Διαισθητικός και εύκολος στην χρήση σχεδιαστής GUI Matisse για την ανάπτυξη Swing: Με λίγη ή καθόλου γνώση Swing, οι χρήστες, μπορούν ήδη να εργαστούν, μεταφέροντας και αποθέτοντας στοιχεία σε ένα WYSIWYG (What You See Is What You Get) παράθυρο σχεδίασης. Ο σχεδιαστής GUI Matisse παράγει πραγματικό κώδικα Swing.
- Ισχυρή υποστήριξη αναδόμησης: Αυτό επιτυγχάνεται με την μηχανή Jackpot, η οποία καθιστά δυνατή την αναδόμηση κώδικα Java χρησιμοποιώντας μια γλώσσα ερωτημάτων που θυμίζει κανονικές εκφράσεις. Σχεδιασμένη από τον James Gosling, η γλώσσα ερωτημάτων είναι αρκετά εύκολη στην χρήση και επιτρέπει την ταύτιση και αντικατάσταση μοτίβων. Τα ερωτήματα μπορούν να ταυτίσουν συγκεκριμένους τύπους Java ή στιγμιότυπα αντικειμένων.

- Υποστήριξη UML project: Οι προγραμματιστές μπορούν να δημιουργήσουν ένα UML (Unified Modeling Language) project, για την μοντελοποίηση του κώδικα, βημάτων διεργασιών ή σχεδιαστικών μοτίβων. Τα UML projects μπορούν να συνδεθούν άμεσα με Java projects. Καθώς ένας χρήστης δημιουργεί και τροποποιεί τα αντικείμενα και διαγράμματα UML, ο αντίστοιχος κώδικας Java δημιουργείται αυτομάτως. Εάν ο πηγαίος κώδικας του συνδεμένου Java project αλλάξει, το διάγραμμα ενημερώνεται επίσης αυτομάτως. Δίνεται η δυνατότητα της εξαγωγής των διαγραμμάτων, της παραγωγής κώδικα και της δημιουργίας αναφορών του project.
- Υποστήριξη φορητών εφαρμογών J2ME: Το NetBeans IDE υποστηρίζει την Java 2 Micro Edition (J2ME) Mobile Information Device Profile (MIDP) 2.0 και παρέχει ένα GUI σχεδίασης, έναν οδηγό ασύρματης σύνδεσης, ένα εισαγωγικό διδακτικό βοήθημα καθώς και πολλά δείγματα εφαμογών.
- Εργαλεία συνεργασίας κατασκευαστών λογισμικού: Οι κατασκευαστές λογισμικού μπορούν να συνδεθούν σε ένα δημόσιο ή ιδιωτικό περιβάλλον και να μοιραστούν κώδικα. Μπορούν να λάβουν μέρος σε δημόσιες ή ιδιωτικές συζητήσεις. Επίσης είναι δυνατή η μεταφορά και η απόθεση κώδικα ή και ολόκληρων project στο παράθυρο της συνομιλίας για τον διαμοιρασμό κώδικα με έναν ή περισσότερους προγραμματιστές. Το NetBeans υποστηρίζει την κωδικοποίηση ομάδων πολλών χρηστών. Καθώς ένας χρήστης ξεκινά να αλλάζει ένα μπλοκ κώδικα, αυτό επισημαίνεται και κλειδώνεται για τους άλλους χρήστες που το διαμοιράζονται. Στις περιπτώσεις στις οποίες τα μέλη μιας ομάδας ανάπτυξης λογισμικού βρίσκονται σε διαφορετικές τοποθεσίες, αυτό το εργαλείο μπορεί να αποδειχθεί χρήσιμο.
- Εύκολο στην χρήση κέντρο ενημερώσεων: Το κέντρο ενημερώσεων του NetBeans επιτρέπει την επιλογή των δικτυακών τόπων στους οποίους θα γίνεται έλεγχος για αλλαγές, ενημερώσεις και καινούργια δομικά στοιχεία.
- Υποστήριξη JSP και Tomcat: Το NetBeans συμπεριλαμβάνει τον Apache Tomcat. Αφού χρησιμοποιηθεί ο οδηγός νέου project για την

δημιουργία ενός project εφαρμογής web, μπορούν να δημιουργηθούν αρχεία JSP (JavaServer Pages). Ο χρήστης μπορεί κάνοντας δεξί κλικ πάνω σε οποιοδήποτε αρχείο JSP και μετά επιλέγοντας από το μενού που εμφανίζεται την επιλογή Run File, να το εκτελέσει. Ο εξυπηρετητής Tomcat ξεκινά αμέσως, ανοίγει ο προεπιλεγμένος πλοηγός και εμφανίζει το αρχείο που εκτελείται στον Tomcat. Επίσης, το NetBeans ενεργοποιεί το HTTP Monitor.

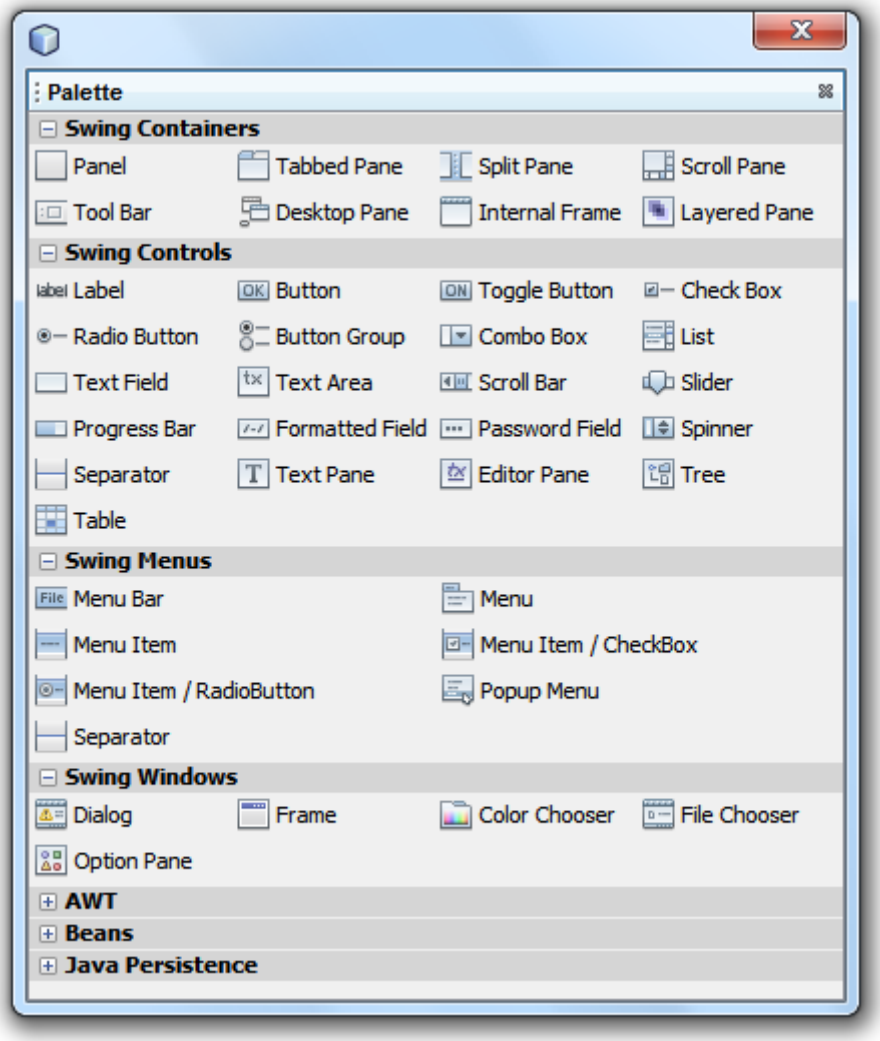
- NetBeans HTTP Monitor: Το NetBeans Monitor μπορεί να ενεργοποιηθεί κατά την αποσφαλμάτωση ή εκτέλεση της εφραμογής web. Επιτρέπει την παρακολούθηση των αιτήσεων, των κεφαλίδων HTTP, των cookies, της συνεδρίας (session) και των παραμέτρων πελάτη/εξυπηρετητή. Έτσι δεν είναι πλέον αναγκαίο να γραφεί κώδικας από την πλευρά του εξυπηρετητή (Server Side) για την παρακολούθηση των περιεχομένων αυτών των μεταβλητών. Μέσα από το NetBeans μπορεί να αποσφαλματωθεί ο κώδικας, να εκτελεστεί γραμμή προς γραμμή και παράλληλα να παρακολουθούνται οι επιθυμητές ιδιότητες.

Τα χαρακτηριστικά αυτά είναι μόνο ένα δείγμα αυτών που μπορεί να προσφέρει το NetBeans. Άλλα Java IDEs μπορεί να παρέχουν μερικές από τις δυνατότητες που αναφέρθηκαν, αλλά το NetBeans έχει το πιο διαισθητικό περιβάλλον εργασίας και το πιο ολοκληρωμένο ενσωματωμένο σετ εργαλείων.

3.1 Ανάπτυξη εφαρμογών GUI - Δημιουργία μιας εφαρμογής GUI

Το NetBeans έχει μερικές εντυπωσιακές δυνατότητες ανάπτυξης εφαρμογών γραφικού περιβάλλοντος χρήστη. Παρέχει τον κατασκευαστή GUI Matisse με τον οποίο είναι δυνατός ο WYSIWYG (What You See Is What You Get) σχεδιασμός των εφαρμογών. Ο κατασκευαστής GUI συμπεριλαμβάνει διαισθητικά και εύκολα στην χρήση χαρακτηριστικά, όπως οδηγούς, άγκυρες κτλ. Συνδυαζόμενος με τα υπόλοιπα χαρακτηριστικά του NetBeans, οι κατασκευαστές λογισμικού έχουν στην διάθεσή τους ένα επαγγελματικό περιβάλλον κωδικοποίησης για την ανάπτυξη εφαρμογών.

Ένα από τα πρώτα εργαλεία που πρέπει να κατανοήσει κάποιος που δουλεύει με εφαρμογές GUI είναι η λίστα των διαθέσιμων συστατικών στο παράθυρο της παλέτας. Στο σχήμα 3.1 φαίνεται η παλέτα με τα διαθέσιμα συστατικά.



Σχήμα 3-1: Η παλέτα συστατικών GUI του NetBeans

Όπως φαίνεται στο σχήμα 3-1, το παράθυρο της παλέτας παρέχει συστατικά για Swing Containers, Swing Controls, Swing Menus, και Swing Windows που αποτελούν μέρος του πακέτου javax.swing όπως φαίνεται και στους πίνακες 3-1, 3-2 και 3-3.

Πίνακας 3-1: Συστατικά Swing Container που είναι διαθέσιμα στην παλέτα

Component Name	**Representative Class**
Panel	javax.swing.JPanel
Tabbed Pane	javax.swing.JTabbedPane
Split Pane	javax.swing.JSplitPane
Scroll Pane	javax.swing.JScrollPane
Tool Bar	javax.swing.JToolBar
Desktop Pane	javax.swing.JDesktopPane
Internal Frame	javax.swing.JInternalFrame
Layered Pane	javax.swing.JLayeredPane
Label	javax.swing.JLabel
Button	javax.swing.JButton
Toggled Button	javax.swing.JToggleButton
Check Box	javax.swing.JCheckBox
Radio Button	javax.swing.JRadioButton
Button Group	javax.swing.ButtonGroup
Combo Box	javax.swing.JComboBox
List	javax.swing.JList
Text Field	javax.swing.JTextField
Text Area	javax.swing.JTextArea
Scroll Bar	javax.swing.JScrollBar
Slider	javax.swing.JSlider
Progress Bar	javax.swing.JProgressBar
Formatted Field	javax.swing.JFormattedTextField
Password Field	javax.swing.JPasswordField
Spinner	javax.swing.JSpinner
Separator	javax.swing.JSeparator
Text Pane	javax.swing.JTextPane
Editor Pane	javax.swing.JEditorPane
Tree	javax.swing.JTree
Table	javax.swing.JTable

Πίνακας 3-2: Συστατικά Swing Menu που είναι διαθέσιμα στην παλέτα

Component Name	**Representative Class**
Menu Bar	javax.swing.JMenuBar
Menu	javax.swing.JMenu
Menu Item	javax.swing.JMenuItem
Menu Item/CheckBox	javax.swing.JCheckBoxMenuItem
Menu Item/RadioButton	javax.swing.JRadioButtonMenuItem
Popup Menu	javax.swing.JPopupMenu
Separator	javax.swing.JSeparator

Πίνακας 3-3: Συστατικά Swing Window που είναι διαθέσιμα στην παλέτα

Component Name	**Representative Class**
Dialog	javax.swing.JDialog
Frame	javax.swing.JFrame
Color Chooser	javax.swing.JColorChooser
File Chooser	javax.swing.JFileChooser
Option Pane	javax.swing.JOptionPane

Το παράθυρο της παλέτας παρέχει επίσης συστατικά AWT που είναι μέρος του πακέτου java.awt, όπως φαίνεται και στους πίνακες 3-4 και 3-5. Οι τομείς Beans και Java Persistance που φαίνονται στο σχήμα 3-1 είναι διεπαφές του πακέτου javax.persistence για την χρήση persistance units.

Πίνακας 3-4: Συστατικά AWT που είναι διαθέσιμα στην παλέτα

Component Name	**Representative Class**
Label	java.awt.Label
Button	java.awt.Button
Text Field	java.awt.TextField
Text Area	java.awt.TextArea
Checkbox	java.awt.Checkbox
Choice	java.awt.Choice
List	java.awt.List
Scrollbar	java.awt.Scrollbar

Πίνακας 3-5: Συστατικά AWT που είναι διαθέσιμα στην παλέτα

Scroll Pane	java.awt.ScrollPane
Panel	java.awt.Panel
Canvas	java.awt.Canvas
Menu Bar	java.awt.MenuBar
Popup Menu	java.awt.PopupMenu

Δημιουργία του Project

Οι Java εφαρμογές GUI μπορούν να δημιουργηθούν μέσα σε ένα τυπικό NetBeans Java Application project. Για την δημιουργία ενός νέου project είναι απαραίτητα τα παρακάτω βήματα:

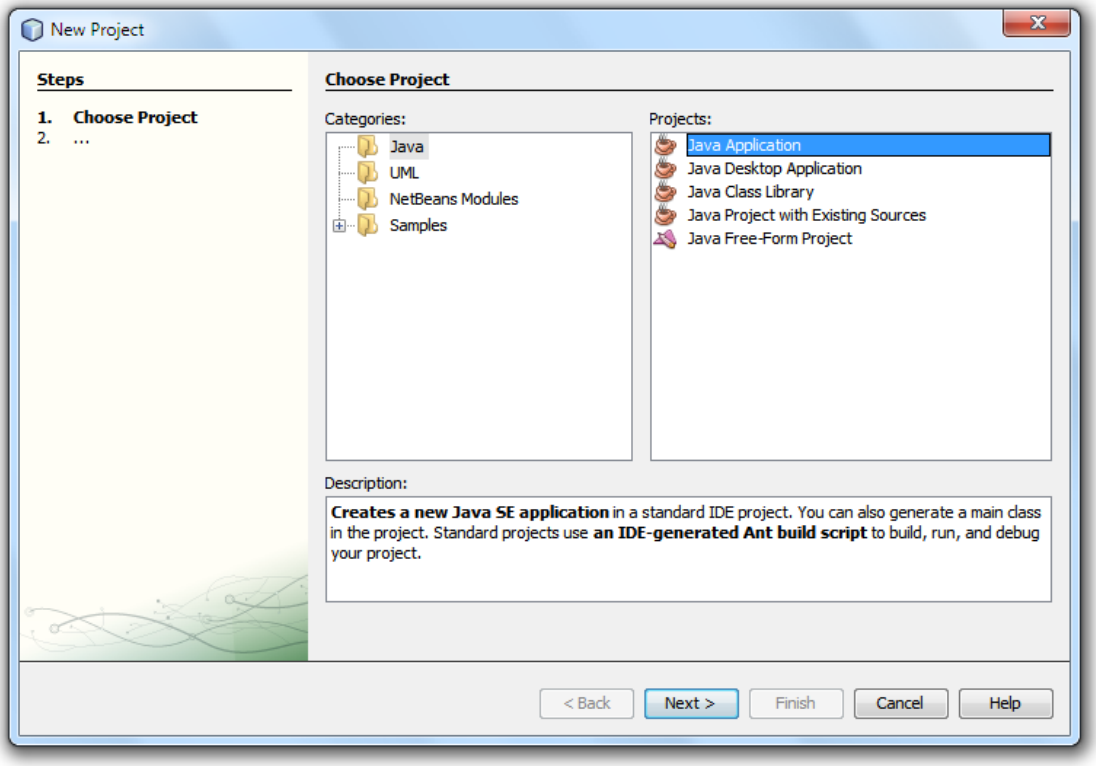
1. Επιλογή File->New Project από το κυρίως μενού.
2. Στο παράθυρο του νέου Project (Σχήμα 3-2) πρέπει να επιλεχθεί η Java στην λίστα των κατηγοριών και Java Application στην λίστα των Projects. Έπειτα κλικ στο Next για συνέχεια.
3. Στο επόμενο παράθυρο (Σχήμα 3-3) πρέπει να δοθούν τιμές για τα πεδία Project Name και Project Location.
4. Τέλος μπορεί να επιλεχθεί εάν το νέο Project θα είναι το κυρίως project καθώς και το αν θα δημιουργηθεί μια κενή Main Class. Πατώντας το κουμπί Finish ολοκληρώνεται η δημιουργία του project.

Όταν το project δημιουργηθεί, θα ανοιχθεί στο παράθυρο των Projects. Στο σημείο αυτό μπορούν να δημιουργηθούν ή να εισαχθούν τάξεις στο project.

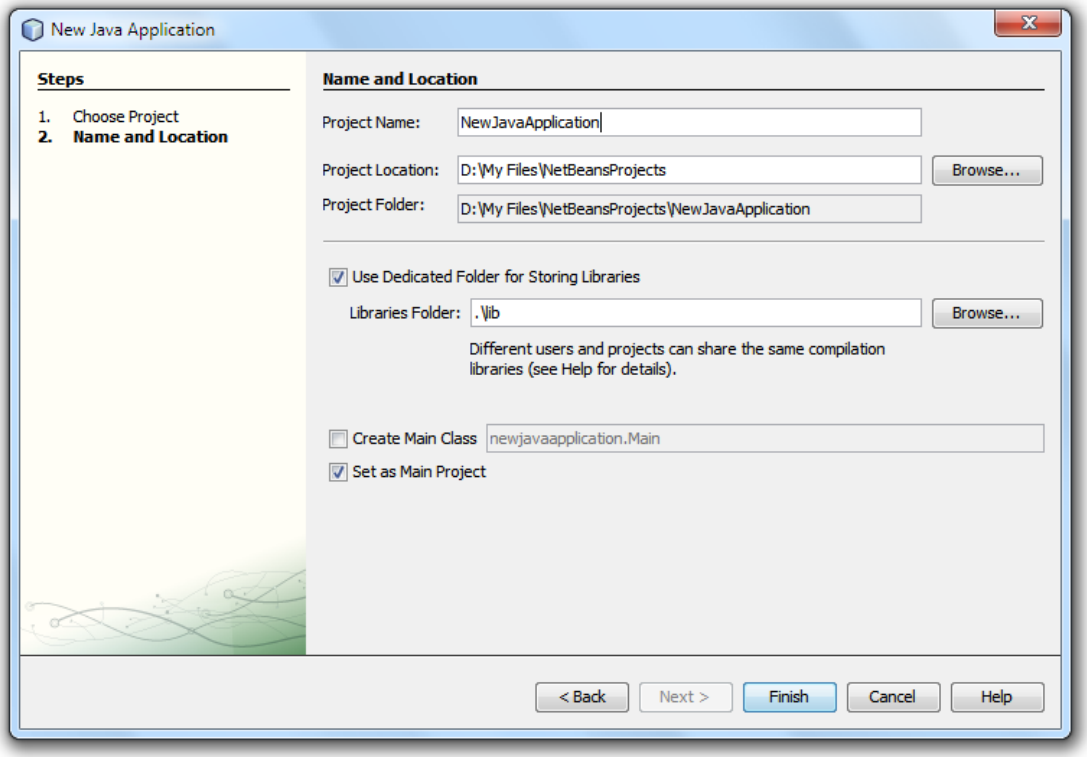
Δημιουργία της αρχικής JFrame Τάξης

Για την εισαγωγή μιας τάξης JFrame αρκούν τα παρακάτω βήματα:

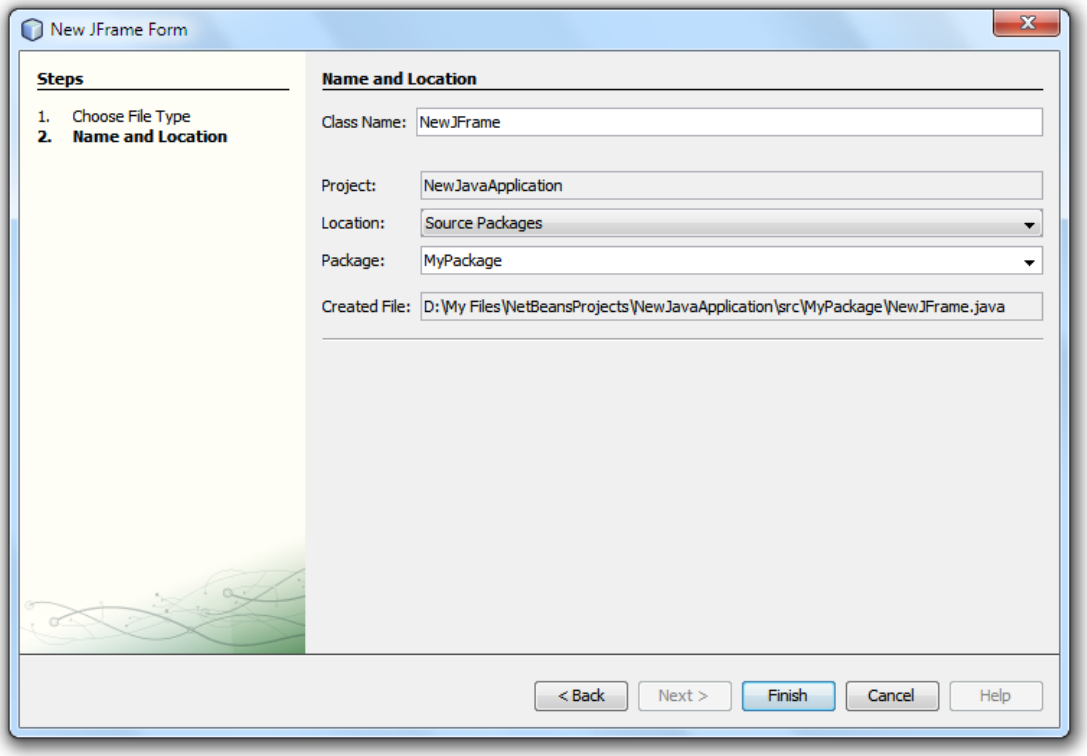
1. Στο παράθυρο των Projects, δεξί κλικ στο Source Packages του νέου project και μετά New->JFrame Form από το μενού που εμφανίζεται.
2. Στο επόμενο παράθυρο πρέπει να εισαχθούν τιμές για τα πεδία Class Name και Package.
3. Τέλος, πατώντας το κουμπί τέλος δημιουργείται η τάξη η οποία ανοίγει στον επεξεργαστή φόρμας.



Σχήμα 3-2: Παράθυρο νέου project.



Σχήμα 3-3: Παράθυρο ονομασίας και τοποθεσίας του νέου project.

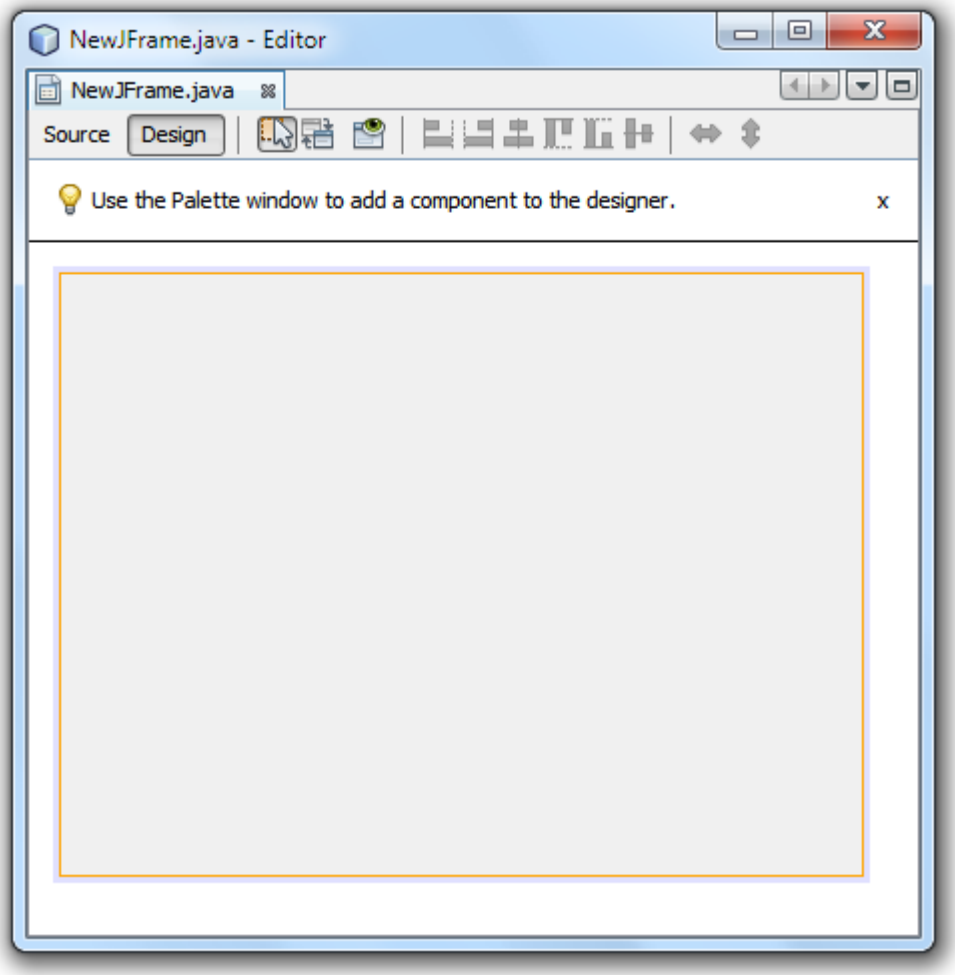


Σχήμα 3-4: Παράθυρο ονομασίας και τοποθεσίας της νέας φόρμας JFrame.

Δουλεύοντας με τον Επεξεργαστή Φόρμας

Αφού δημιουργηθεί η τάξη Java που αντιπροσωπεύει το GUI, μπορεί να ξεκινήσει η επεξεργασία της στον επεξεργαστή φόρμας του NetBeans. Ο επεξεργαστής φόρμας περιλαμβάνει δύο όψεις της τάξης Java, την προβολή κώδικα στην οποία εμφανίζεται ο κώδικας της και την προβολή σχεδίασης στην οποία εμφανίζεται μια WYSIWYG όψη της GUI φόρμας της.

Την πρώτη φορά που θα ανοίξει στον επεξεργαστή φόρμας η τάξη Java που δημιουργήθηκε, θα είναι επιλεγμένη από προεπιλογή η προβολή σχεδίασης. Θα εμφανίζει μια γκρι τετράγωνη περιοχή, όπως φαίνεται στο σχήμα 3-5, στην οποία μπορούν να εισαχθούν συστατικά από την παλέτα. Με τον τρόπο αυτό μπορεί να σχεδιαστεί οπτικά η φόρμα GUI που πρόκειται να δημιουργηθεί.

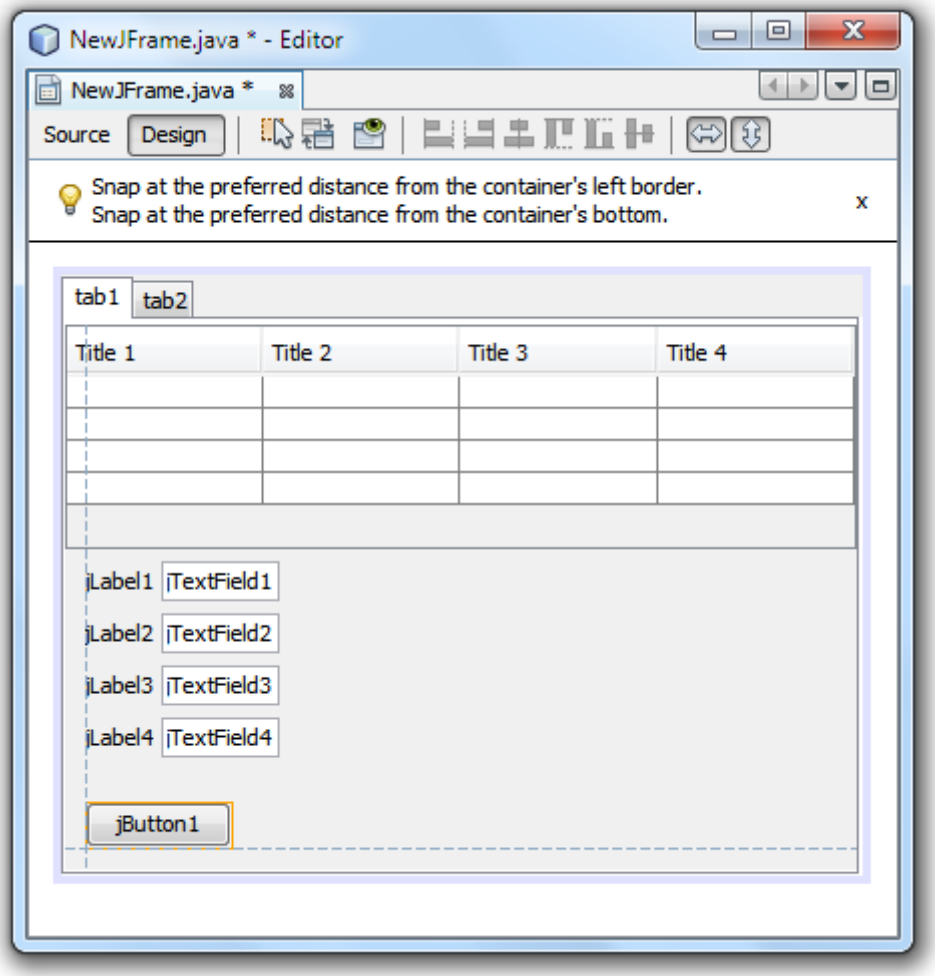


Σχήμα 3-5: Ένα κενό JFrame όπως εμφανίζεται στην προβολή σχεδίασης του επεξεργαστή φόρμας.

Εισαγωγή συστατικών στην φόρμα

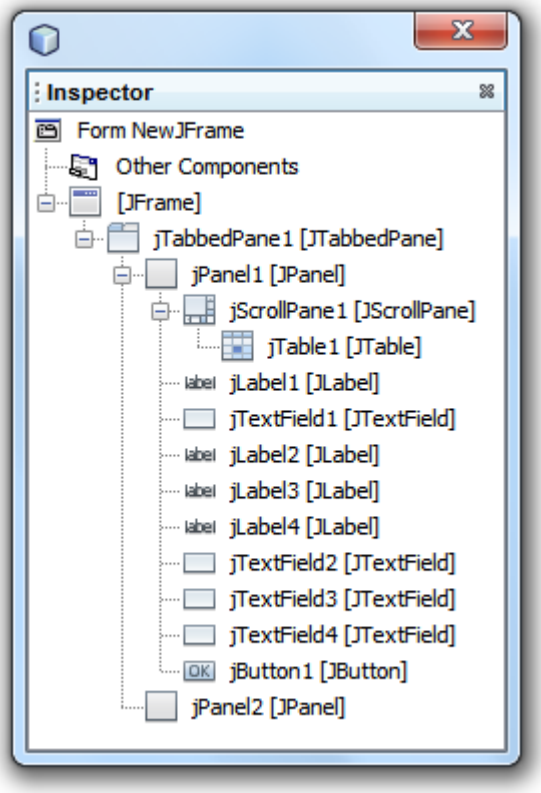
Αφού δημιουργηθεί η φόρμα, μπορούν να εισαχθούν σε αυτήν συστατικά του GUI. Υπάρχουν πολλοί τρόποι για να γίνει αυτό, αλλά πάντα θα πρέπει πρώτα να επιλεχθεί το επιθυμητό συστατικό και μετά να τοποθετηθεί στο επιθυμητό σημείο. Μπορεί να επιλεχθεί κάποιο συστατικό χρησιμοποιώντας την παλέτα, ή κάνοντας δεξί κλικ πάνω στην φόρμα και επιλέγοντας από το μενού που εμφανίζεται την επιλογή Add From Palette. Ένας άλλος τρόπος που μπορεί να φανεί πιο χρήσιμος, ειδικά στην περίπτωση που συστατικά της φόρμας εμπεριέχουν ή επικαλύπτουν το ένα το άλλο, είναι να επιλεχθεί το επιθυμητό συστατικό κάνοντας δεξί κλικ (από τον Inspector του NetBeans) πάνω στον container στον οποίο πρόκειται να εισαχθεί και επιλέγοντας και πάλι από το μενού που εμφανίζεται την επιλογή Add From Palette. Επίσης,

εάν κάποιο συστατικό βρίσκεται κατά λάθος σε λάθος container, χρησιμοποιώντας τον Inspector μπορεί να μεταφερθεί στον σωστό.



Σχήμα 3-6: Εισαγωγή ενός JButton στην προβολή σχεδίασης του επεξεργαστή φόρμας.

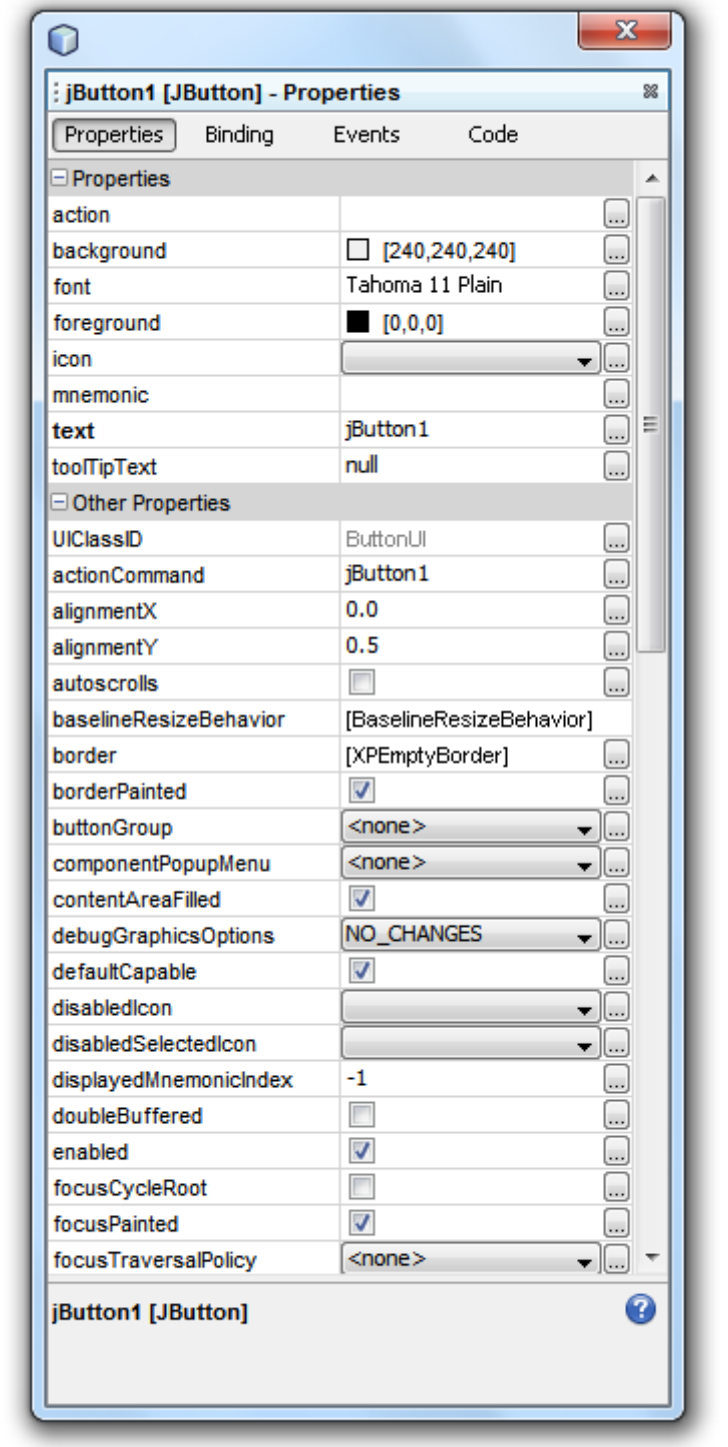
Στο σχήμα 3-6 φαίνεται η φόρμα με κάποια συστατικά τα οποία έχουν εισαχθεί. Επίσης φαίνεται και η τοποθέτηση ενός JButton στην φόρμα. Καθώς μετακινείται το συστατικό JButton στην φόρμα, εμφανίζονται διακεκομμένες γραμμές σε διάφορα σημεία, ανάλογα με την ευθυγράμμιση του συστατικού. Οι γραμμές αυτές δείχνουν με ποιο τρόπο θα καθορίζεται η σχετική θέση του συστατικού, σε περίπτωση που το μέγεθος της φόρμας, ή κάποιου άλλου συστατικού με το οποίο έχει ευθυγραμμιστεί, αλλάξει. Στο σχήμα 3-7 φαίνεται ο Inspector του NetBeans.



Σχήμα 3-7: Ο Inspector του NetBeans.

Τροποποίηση των χαρακτηριστικών των συστατικών

Κάθε συστατικό, συμπεριλαμβανόμενου και του JFrame, έχει έναν μεγάλο αριθμό χαρακτηριστικών που μπορούν να τροποποιηθούν χρησιμοποιώντας το παράθυρο Properties του NetBeans. Έτσι, αφού εισαχθούν συστατικά στην φόρμα, μπορούν να τροποποιηθούν τα χαρακτηριστικά τους. Στο σχήμα 3-8 φαίνεται το παράθυρο Properties του NetBeans.



Σχήμα 3-8: Το παράθυρο Properties του NetBeans με τα χαρακτηριστικά ενός JButton.

Το αρχείο Java που δημιουργήθηκε περιέχει την ακόλουθη μέθοδο main η οποία παράχθηκε αυτομάτως:

```
public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new NewJFrame().setVisible(true);
            }
        });
    }
```

Αυτή καλεί τον δομητή, ο οποίος είναι ορισμένος ως:

```
public NewJFrame() {
        initComponents();
 }
```

Ο δομητής καλεί την μέθοδο initComponents που είναι private, και αρχικοποιεί τα συστατικά που εμφανίζονται στην φόρμα:

```
// <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jTabbedPane1 = new javax.swing.JTabbedPane();
        jPanel1 = new javax.swing.JPanel();
        jScrollPane1 = new javax.swing.JScrollPane();
        jTable1 = new javax.swing.JTable();
        jLabel1 = new javax.swing.JLabel();
        jTextField1 = new javax.swing.JTextField();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jTextField2 = new javax.swing.JTextField();
        jTextField3 = new javax.swing.JTextField();
```

```
        jTextField4 = new javax.swing.JTextField();
        jButton1 = new javax.swing.JButton();
        jPanel2 = new javax.swing.JPanel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jTable1.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {
                {null, null, null, null},
                {null, null, null, null},
                {null, null, null, null},
                {null, null, null, null}
            },
            new String [] {
                "Title 1", "Title 2", "Title 3", "Title 4"
            }
        ));
        jScrollPane1.setViewportView(jTable1);

        jLabel1.setText("jLabel1");

        jTextField1.setText("jTextField1");

        jLabel2.setText("jLabel2");

        jLabel3.setText("jLabel3");

        jLabel4.setText("jLabel4");

        jTextField2.setText("jTextField2");

        jTextField3.setText("jTextField3");

        jTextField4.setText("jTextField4");
```

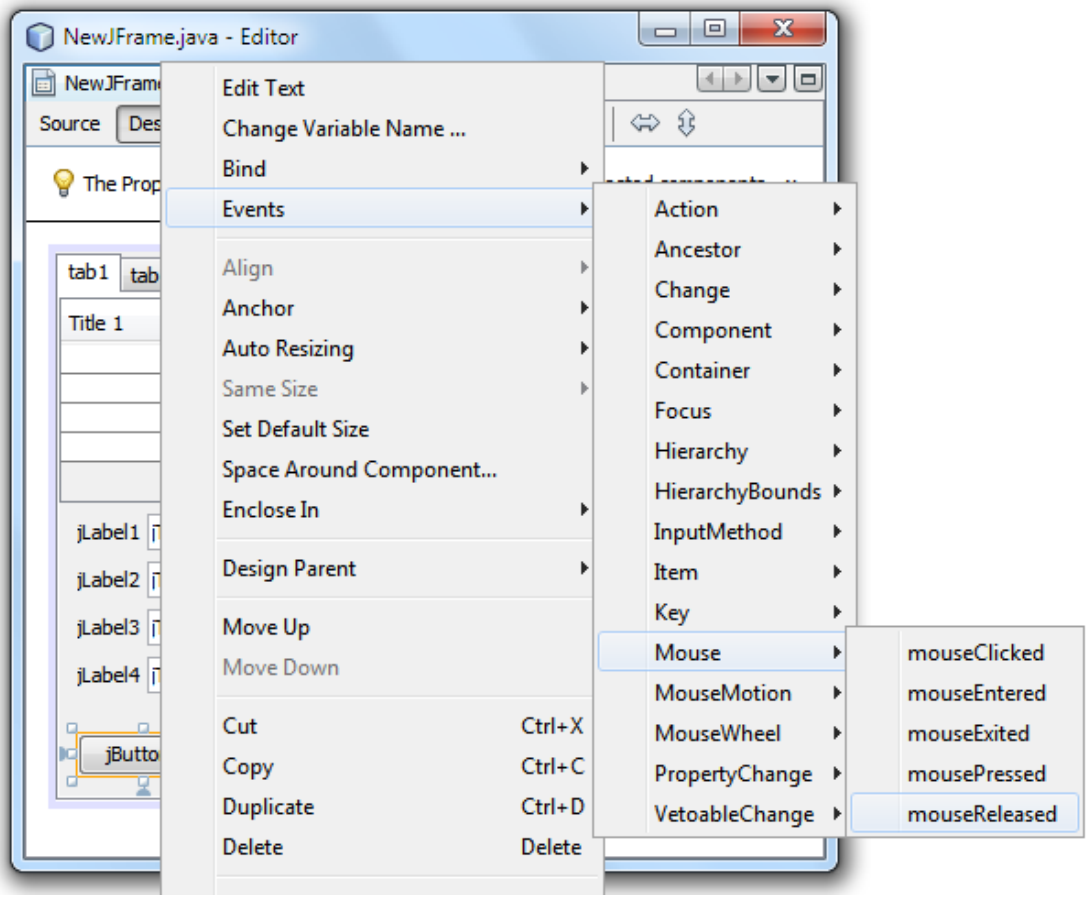
```
        jButton1.setText("jButton1");

        jTabbedPane1.addTab("tab1", jPanel1);

        jTabbedPane1.addTab("tab2", jPanel2);

//Ακολουθεί κώδικας αρχικοποίησης για το GroupLayout ο οποίος αφαιρέθηκε
…
…
…
    }// </editor-fold>
```

Ένα από τα πλεονεκτήματα του κατασκευαστή GUI Matisse του NetBeans είναι πως παρέχει ακριβή παραγωγή κώδικα και την δυνατότητα να διατηρείται ο κώδικας ενήμερος, ειδικά όταν τα συστατικά αλλάζουν. Όποτε γίνονται αλλαγές στα χαρακτηριστικά των συστατικών στο παράθυρο Properties, ο Matisse κάνει τις απαραίτητες αλλαγές στον κώδικα.

Εισαγωγή Events

Μία σημαντική έννοια κατά την εργασία με εφαρμογές GUI είναι η διαχείριση των γεγονότων (events). Είτε πρόκειται για το πάτημα ενός πλήκτρου του πληκτρολογίου, το κλικ ενός ποντικιού ή κάποιο άλλο συμβάν, τα events πρέπει να αντιμετωπιστούν κατάλληλα. Τα events είναι ο τρόπος με τον οποίο οι χρήστες αλληλεπιδρούν με μια εφαρμογή.

Κάνοντας δεξί κλικ πάνω σε κάποιο συστατικό της φόρμας και επιλέγοντας από το μενού που εμφανίζεται την επιλογή events, μπορεί να γραφεί ο κώδικας για τον χειρισμό των events του. Στο σχήμα 3-9 φαίνεται το μενού Events ενός JButton.



Σχήμα 3-9: Το μενού Events ενός JButton.

Ο επεξεργαστής φόρμας θα αλλάξει σε προβολή κώδικα και θα εμφανίσει την νέα κενή μέθοδο χειρισμού γεγονότος που δημιουργήθηκε, πχ την jButton1MouseReleased η οποία σε κώδικα θα είναι κάπως έτσι:

```
private void jButton1MouseReleased(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:
    }
```

Ο επεξεργαστής φόρμας παράγει αυτόματα τον κώδικα για την καταχώρηση του χειριστή γεγονότος και τον τοποθετεί στην μέθοδο initComponents. Ένας νέος MouseListener προστίθεται στο κουμπί jButton1, ο οποίος καλεί την μέθοδο jButton1MouseReleased:

```
jButton1.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseReleased(java.awt.event.MouseEvent evt) {
```

```
            jButton1MouseReleased(evt);
        }
    });
```

3.2 Ο επεξεργαστής κώδικα

Ο επεξεργαστής κώδικα είναι ένα από τα πιο σημαντικά σημεία στα οποία επικεντρώνεται το NetBeans. Με αυτόν οι κατασκευαστές λογισμικού γράφουν νέο κώδικα, επεξεργάζονται υπάρχων κώδικα, παράγουν την τεκμηρίωση του λογισμικού και εκτελούν πολλές σημαντικές εργασίες.

Ο επεξεργαστής κώδικα του NetBeans δεν είναι μόνο ένας μηχανισμός για την σύνταξη κειμένου, αλλά ένα πλήρες εξοπλισμένο περιβάλλον σχεδιασμένο να υποστηρίξει τον χρήστη. Είτε πρόκειται για συντομογραφίες για γρηγορότερη κωδικοποίηση, αυτόματη συμπλήρωση κώδικα ή βοηθήματα πλοήγησης και τεκμηρίωσης, ο επεξεργαστής κειμένου στοχεύει να προσφέρει κάθε δυνατή άνεση.

Δουλεύοντας στο παράθυρο Projects

Το παράθυρο Projects είναι η κύρια τοποθεσία στην οποία εμφανίζονται τα αρχεία που συσχετίζονται με την εφαρμογή. Έχει την δομή δέντρου γονέα-παιδιού, όπου ο κόμβος γονέας είναι το project και οι κόμβοι παιδιά οι κατηγορίες στις οποίες οργανώνει τα αρχεία το NetBeans. Για τους περισσότερους τύπους project Java, τα αρχεία ταξινομούνται σε τέσσερις ομάδες:

- Source Packages
- Test Packages
- Libraries
- Test Libraries

Source Packages

Η κατηγορία Source Packages είναι το μέρος που ορίζεται ο πηγαίος κώδικας που χρησιμοποιείται στην εφαρμογή. Η προσθήκη πακέτων επιτυγχάνεται εύκολα. Κάνοντας δεξί κλικ στο Source Packages και επιλέγοντας από το μενού που εμφανίζεται την επιλογή New->Java Package. Στο παράθυρο New

Package που θα εμφανιστεί, μπορεί να καθοριστεί το όνομα του καινούργιου πακέτου. Αφού δοθεί ένα όνομα στο πακέτο, πατώντας το κουμπί Finish προστίθεται στην κατηγορία Source Packages του παραθύρου των projects. Κάνοντας δεξί κλικ σε σχεδόν οποιονδήποτε κόμβο που υπάρχει στο παράθυρο των Projects, όπως σε ονόματα project, πακέτα και αρχεία Java, παρουσιάζονται διάφορες επιλογές στο μενού που εμφανίζεται.

Test Packages

Ο κόμβος Test Packages του παραθύρου των Projects είναι σχεδόν πανομοιότυπος με τον κόμβο Source Packages. Παρόλα αυτά, ο κόμβος Test Packages ορίζει την δομή των πακέτων για τις τεστ τάξεις και τα Junit τεστ της εφαρμογής. Εάν εκτελούταν τα τεστ του project πηγαίνοντας στο μενού Run και επιλέγοντας το Test ΌνομαProject, θα εκτελούταν οι τάξεις που βρίσκονται στον κόμβο Test Packages. Ο πηγαίος κώδικας των Test Packages είναι χωρισμένος από τον κανονικό πηγαίο κώδικα των Source Packages.

Libraries

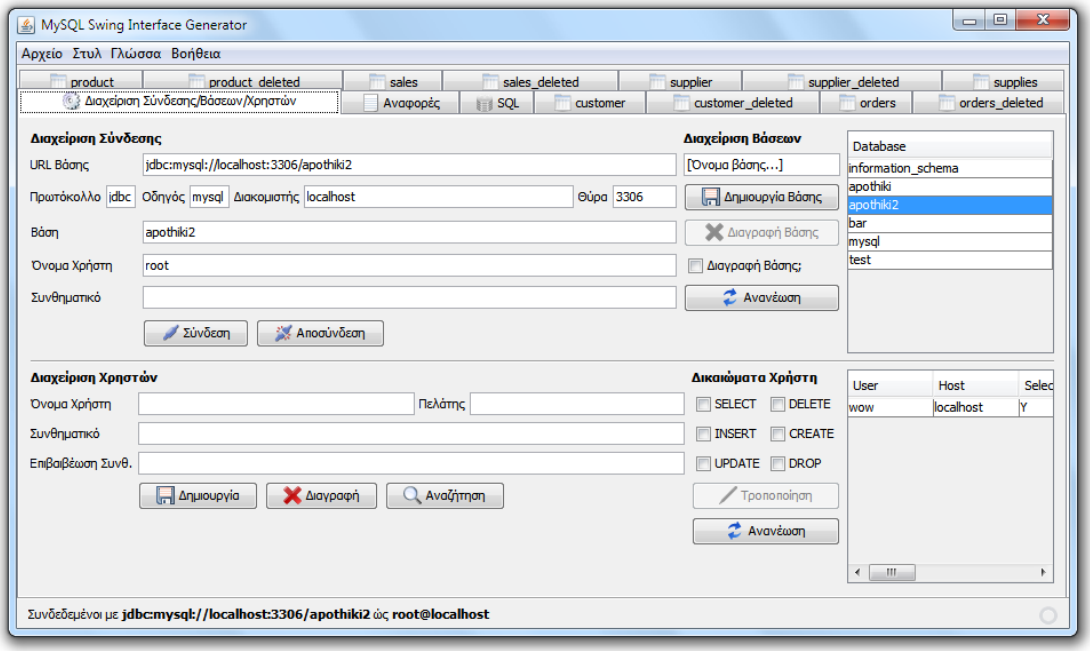
Ο κόμβος Libraries του παραθύρου των Projects περιλαμβάνει τις τάξεις βιβλιοθηκών που θα χρησιμοποιεί η εφαρμογή. Εάν χρειάζεται να χρησιμοποιηθούν μη τυπικές βιβλιοθήκες (όπως οδηγοί JDBC) ή τάξεις ενός εξωτερικού project, αυτές μπορούν να οριστούν στον κόμβο Libraries. Η προσθήκη ενός αρχείου JAR στις βιβλιοθήκες του project μπορεί να γίνει κάνοντας δεξί κλικ στον κόμβο Libraries και επιλέγοντας Add JAR/Folder.

Test Libraries

Παρόμοια με τον κόμβο Libraries, ο κόμβος Test Libraries περιλαμβάνει αρχεία τάξεων ή αρχεία JAR στα οποία αναφέρονται οι τάξεις τεστ του project. Μπορούν να εισαχθούν αρχεία στον κόμβο Test Libraries κάνοντας δεξί κλικ επάνω του και και επιλέγοντας Add JAR/Folder. Το αρχείο JAR του Junit υπάρχει κατά προεπιλογή στον κόμβο Test Libraries.

Κεφάλαιο 4: Οδηγίες Χρήσης MySQL Swing Interface Generator

Διαχείριση Σύνδεσης/Βάσεων/Χρηστών



Διαχείριση Σύνδεσης

URL Βάσης: Εδώ σχηματίζεται αυτόματα το URL καθώς συμπληρώνονται τα πεδία που ακολουθούν. Φυσικά είναι δυνατή και η άμεση πληκτρολόγηση ενός URL.

Πρωτόκολλο: Το πρωτόκολλο που θα χρησιμοποιηθεί για την σύνδεση. Η εφαρμογή υποστηρίζει το πρωτόκολλο JDBC.

Οδηγός: Ο οδηγός βάσης δεδομένων που θα χρησιμοποιηθεί για την σύνδεση. Η εφαρμογή υποστηρίζει την βάση δεδομένων MySQL.

Διακομιστής: Η διεύθυνση δικτύου ή το όνομα του υπολογιστή στον οποίο έχει εγκατασταθεί και εκτελείται η υπηρεσία της βάσης δεδομένων.

Θύρα: Η θύρα του διακομιστή στην οποία δέχεται συνδέσεις η υπηρεσία της βάσης δεδομένων.

Βάση: Το όνομα της βάσης δεδομένων με την οποία θα γίνει η σύνδεση.

Όνομα Χρήστη: Το όνομα του χρήστη που θα συνδεθεί.

Συνθηματικό: Το συνθηματικό του χρήστη που θα συνδεθεί.

Σύνδεση: Συνδέει την εφαρμογή με την βάση δεδομένων, χρησιμοποιώντας

τα παραπάνω στοιχεία.
Αποσύνδεση: Καταργεί την σύνδεση, εάν αυτή υπάρχει.

Διαχείριση Βάσεων

[Όνομα Βάσης]: Το όνομα της βάσης δεδομένων που θα δημιουργηθεί/διαγραφεί.
Δημιουργία Βάσης: Δημιουργεί μια βάση δεδομένων με το όνομα που δόθηκε.
Διαγραφή Βάσης: Για να ενεργοποιηθεί πρέπει πρώτα να τσεκαριστεί το checkbox που βρίσκεται από κάτω. Διαγράφει την βάση δεδομένων που έχει επιλεγεί στον πίνακα δεξιά.
Ανανέωση: Ανανεώνει τον πίνακα δεξιά, όπου φαίνονται οι διαθέσιμες βάσεις δεδομένων.
Πίνακας: Περιέχει τις διαθέσιμες βάσεις δεδομένων.
Διαχείριση Χρηστών
Όνομα Χρήστη: Το όνομα του χρήστη που θα δημιουργηθεί/διαγραφεί/αναζητηθεί.
Πελάτης: Η διεύθυνση δικτύου ή το όνομα του υπολογιστή από τον οποίο συνδέεται ο χρήστης.
Συνθηματικό: Το συνθηματικό του χρήστη.
Επιβεβαίωση Συνθηματικού: Επιβεβαίωση του συνθηματικού του χρήστη.
Δημιουργία: Δημιουργεί τον χρήστη με τα στοιχεία που δόθηκαν παραπάνω και με τα δικαιώματα που έχουν επιλεγεί από δεξιά.
Διαγραφή: Διαγράφει τον χρήστη με τα στοιχεία που δόθηκαν παραπάνω.
Αναζήτηση: Αναζητεί τον χρήστη/ες με τα στοιχεία που δόθηκαν παραπάνω και τα δικαιώματα που έχουν επιλεγεί από δεξιά.

Δικαιώματα Χρήστη

SELECT: Το δικαίωμα του χρήστη να ανακαλεί εγγραφές από την βάση δεδομένων.
INSERT: Το δικαίωμα του χρήστη να προσθέτει εγγραφές στην βάση δεδομένων.

UPDATE: Το δικαίωμα του χρήστη να τροποποιεί εγγραφές της βάσης δεδομένων.

DELETE: Το δικαίωμα του χρήστη να διαγράφει εγγραφές της βάσης δεδομένων.

CREATE: Το δικαίωμα του χρήστη να δημιουργεί πίνακες στην βάση δεδομένων.

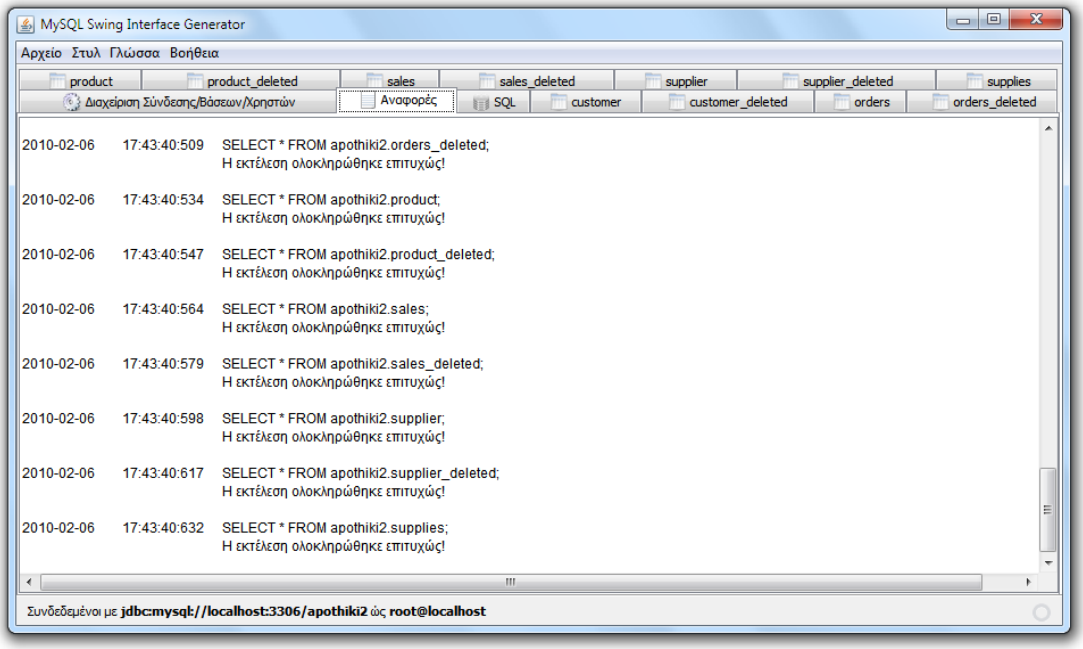
DROP: Το δικαίωμα του χρήστη να διαγράφει πίνακες της βάσης δεδομένων.

Τροποποίηση: Τροποποιεί τα δικαιώματα ενός υπάρχοντος χρήστη.

Ανανέωση: Ανανεώνει τον πίνακα όπου φαίνονται οι χρήστες της βάσης δεδομένων.

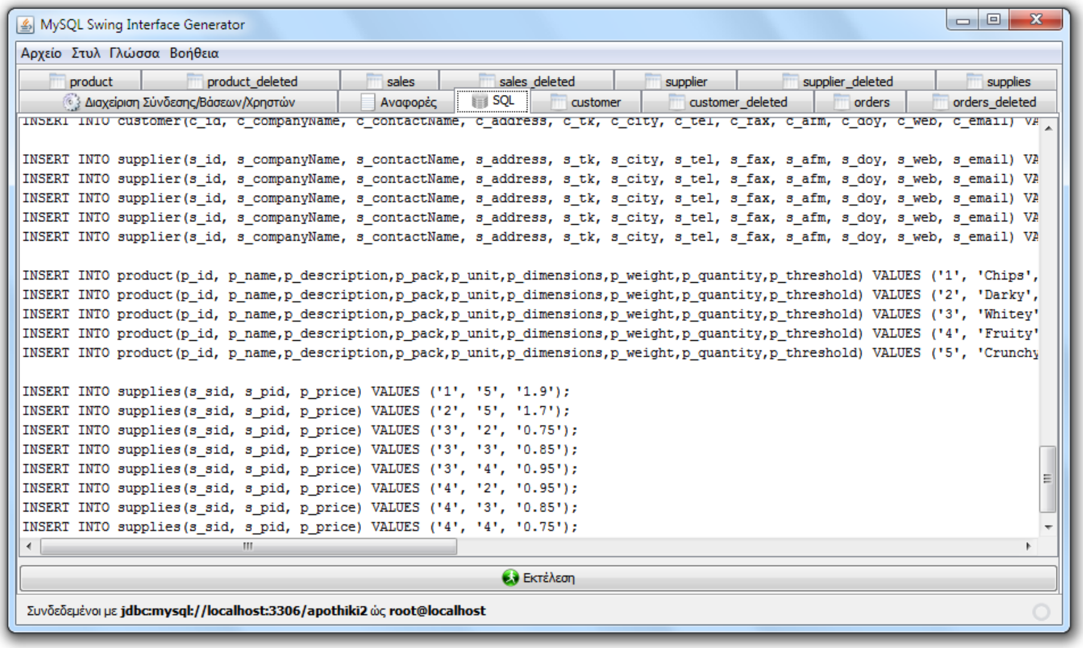
Πίνακας: Περιέχει τους χρήστες της βάσης δεδομένων.

Αναφορές

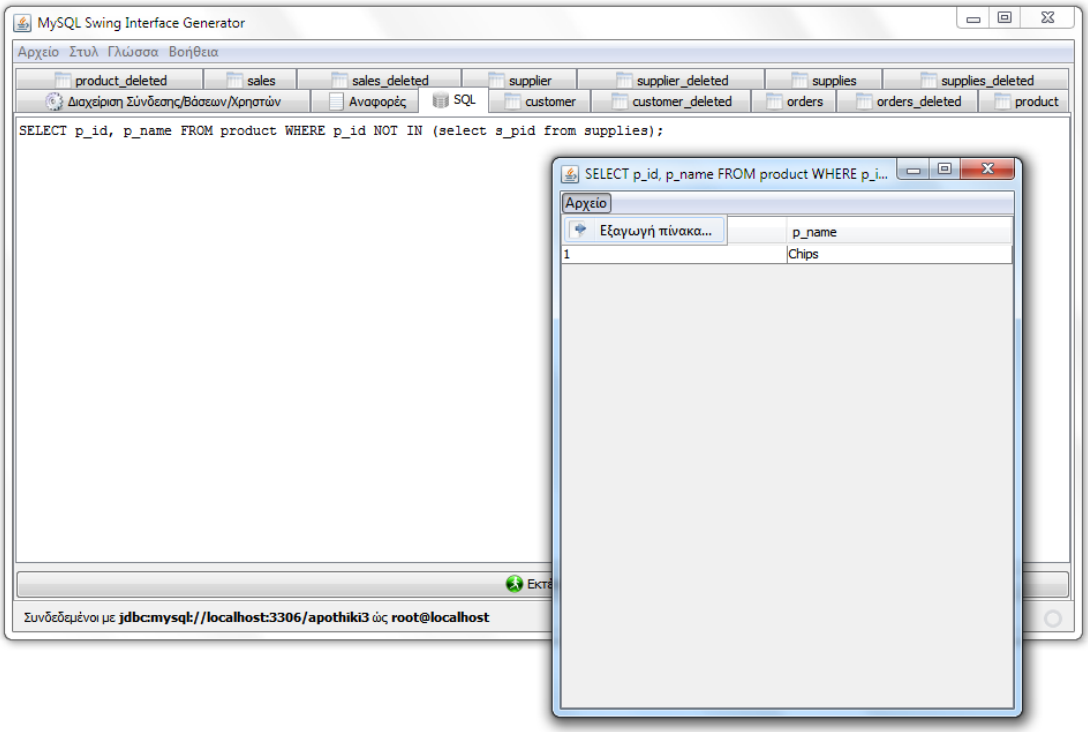


Στην καρτέλα Αναφορές προβάλλονται όλες οι σημαντικές πληροφορίες που αφορούν τις δραστηριότητες της εφαρμογής, όπως πχ όλες οι εντολές SQL που αποστέλλονται στην βάση και το αποτέλεσμά τους κτλ.

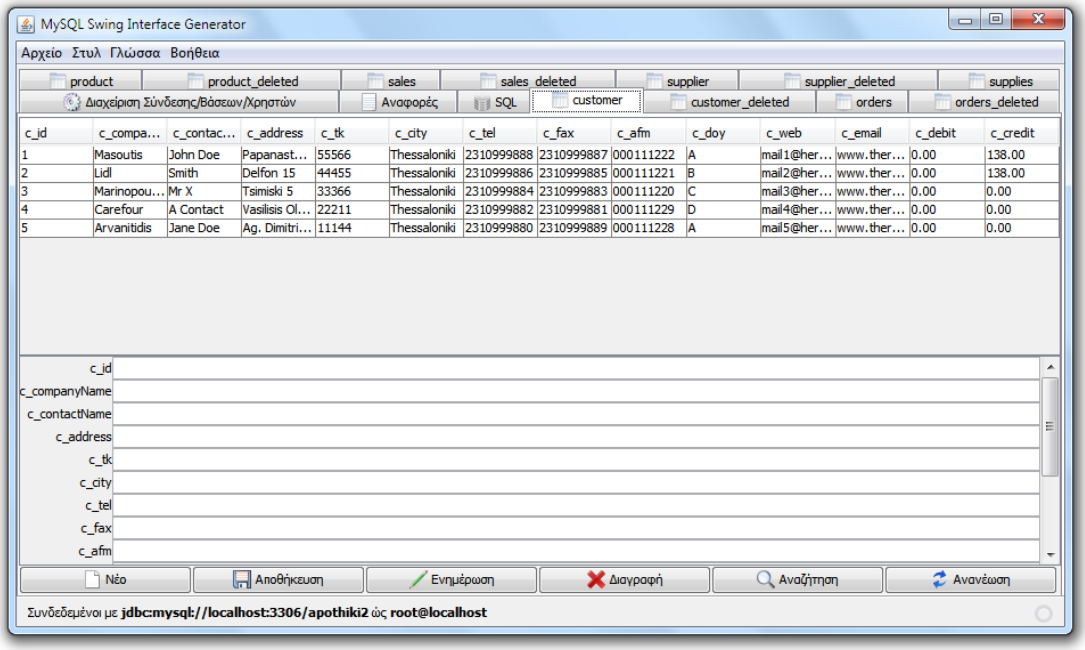
SQL



Στην καρτέλα SQL μπορεί να εισαχθεί κώδικας SQL και έπειτα πατώντας στο κουμπί Εκτέλεση να εκτελεστεί. Υποστηρίζονται όλες οι εντολές SQL. Στην περίπτωση ερωτήματος SELECT δημιουργείται ένα νέο παράθυρο με ένα πίνακα ο οποίος περιέχει τα αποτελέσματα του ερωτήματος. Δίνεται και σε αυτό το παράθυρο η δυνατότητα εξαγωγής του πίνακα σε αρχείο. Προσοχή: για να αναγνωριστεί η εντολή SELECT, θα πρέπει να γραφεί το πρώτο SELECT και το πρώτο FROM του ερωτήματος στα κεφαλαία.



Παραγόμενες Καρτέλες



Αφού η εφαρμογή συνδεθεί επιτυχώς με την βάση, δημιουργεί για κάθε πίνακα που βρίσκει μια καρτέλα που περιλαμβάνει έναν πίνακα για την προβολή των εγγραφών, μια περιοχή στην οποία εισάγεται ένα label και ένα text field για κάθε πεδίο του πίνακα καθώς και έξι κουμπιά:

Νέο: Καθαρίζει όλα τα πεδία της καρτέλας.

Αποθήκευση: Αποθηκεύει τα περιεχόμενα των πεδίων σε μια νέα εγγραφή.

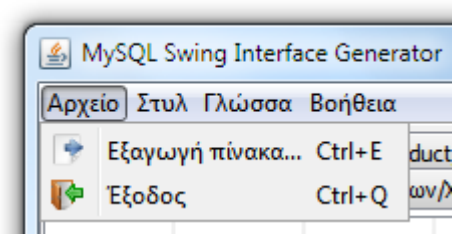
Ενημέρωση: Ενημερώνει τα δεδομένα μιας υπάρχουσας εγγραφής με τα περιεχόμενα των πεδίων.

Διαγραφή: Η εφαρμογή ρωτάει τον χρήστη εάν θέλει να κρατήσει αντίγραφο της διαγραφόμενης εγγραφής και διαγράφει την επιλεγμένη εγγραφή. Εάν ο χρήστης επέλεξε να κρατήσει αντίγραφο της εγγραφής και η διαγραφή ολοκληρώθηκε επιτυχώς, δημιουργεί έναν πίνακα (εάν δεν υπάρχει ήδη), με πρόθεμα το όνομα του πίνακα στον οποίο εκτελείται η διαγραφή και επίθεμα το „_deleted“ και εισάγει σε αυτόν την διαγραμμένη εγγραφή.

Αναζήτηση: Αναζητεί την εγγραφή/ες που ταιριάζουν με τα περιεχόμενα των πεδίων.

Ανανέωση: Ανανεώνει τα περιεχόμενα του πίνακα, εμφανίζοντας πάλι όλες τις εγγραφές του πίνακα.

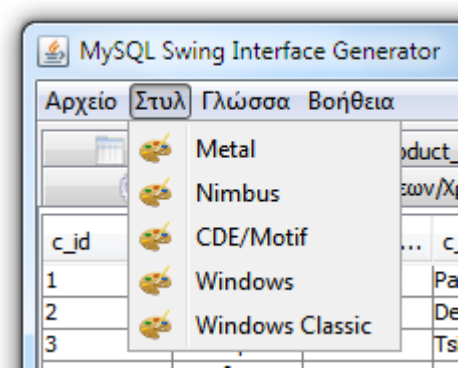
Αρχείο



Εξαγωγή πίνακα: Εξάγει τα περιεχόμενα του πίνακα της επιλεγμένης καρτέλας σε ένα αρχείο. Δίνεται επιλογή ανάμεσα σε τρεις τύπους αρχείων: Αρχεία κειμένου, Αρχεία Excel και Αρχεία HTML.

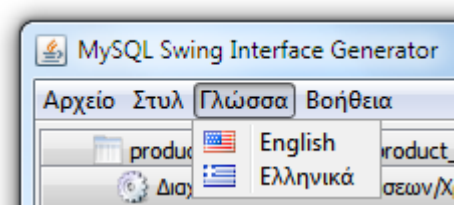
Έξοδος: Τερματίζει την εφαρμογή.

Στυλ



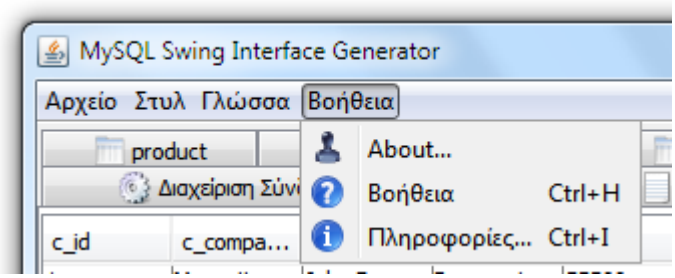
Η εφαρμογή αναζητεί κατά την εκκίνηση τα διαθέσιμα στυλ του συστήματος και τοποθετεί όσα βρεθούν σε αυτό το μενού.

Γλώσσα



Σε αυτό το μενού μπορεί να επιλεγεί η γλώσσα της εφαρμογής, ανάμεσα σε Αγγλικά και Ελληνικά. Προσοχή, η αλλαγή της γλώσσας απαιτεί την επανεκκίνηση της εφαρμογής, σε περίπτωση που υπάρχει ενεργή σύνδεση, αυτή θα τερματιστεί.

Βοήθεια



About: Πληροφορίες σχετικές με την ανάπτυξη της εφαρμογής.
Βοήθεια: Παραπέμπει σε αυτό το αρχείο.

Πληροφορίες: **Πληροφορίες για τις εκδόσεις του λογισμικού και την σύνδεση.Βιβλιογραφία**

[1] Bernard Van Haecke, “JDBC 3.0 - Java Database Connectivity”, M&T Books An imprint of Hungry Minds, Inc, 2002.

[2] Doug Lowe, Barry Burd, “Java All-In-One Desk Reference For Dummies, 2nd Edition”, John Wiley & Sons, 2007.

[3] Mahmoud Parsian, “JDBC Metadata, MySQL, and Oracle Recipes: A Problem-Solution Approach”, Apress, 2006.

[4] George Reese, “Java Database Best Practices”, O'Reilly & Associates, Inc, 2003

[5] Sun Microsystems Inc., Sun Developer Network (SDN), http://java.sun.com

[6] Maydene Fisher, Jon Ellis, Jonathan Bruce, “JDBC™ API Tutorial and Reference, Third Edition”, Addison Wesley, 2003

[7] Lance Andersen, “JDBC™ 4.0 Specification”, Sun Microsystems Inc., November 7 2006

[8] SDN – Sun Developer Network, “Core J2EE Patterns - Data Access Object”, Core J2EE Pattern Catalog, Sun Microsystems, 2001-2002

[9] Michael Kofler, “The Definitive Guide to MySQL5 – Third Edition”, Apress, 2005

[10] Adam Myatt with Brian Leonard and Geertjan Wielenga of Netbeans.org, “Pro NetBeans IDE 6 Rich Client Platform Edition”, Apress, 2008

[11] Stefan Hinz, Team Lead, Paul DuBois, Jonathan Stephens, Martin 'MC' Brown, Anthony Bedford, “MySQL 5.5 Reference Manual”, 2010