XAMPP – JDBC – SERVLETS

ΗΥ 360 – Αρχεία και Βάσεις Δεδομένων

XAMPP

XAMPP: Προτεινόμενο πρόγραμμα δημιουργίας & διαχείρισης σχεσιακών βάσεων δεδομένων τύπου MySQL.

Σύνδεσμος για να κατέβει η έκδοση **8.0.0**:

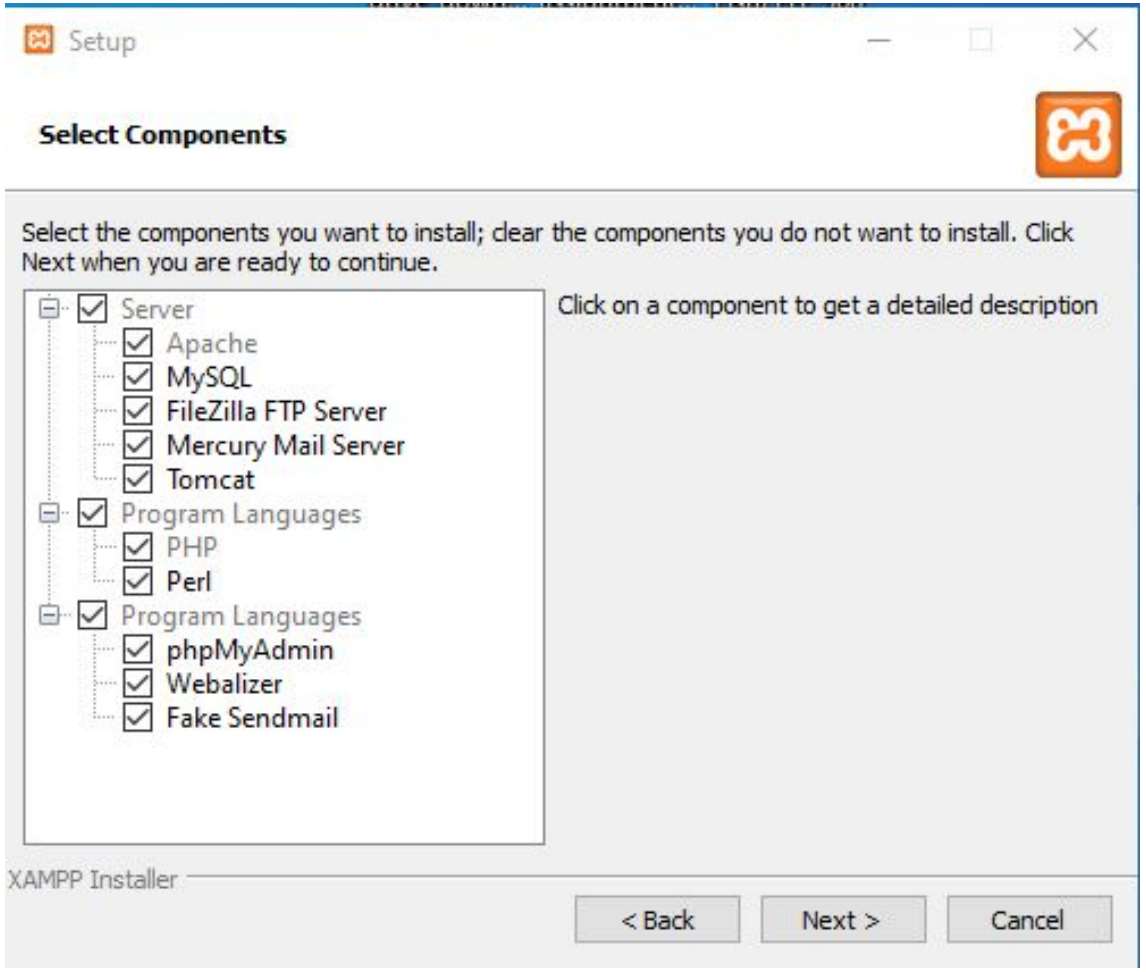
https://www.apachefriends.org/download.html

Εγκατάσταση στο XAMPP τα Components:

- Σύμφωνα με την παρακάτω εικόνα.

XAMPP

XAMPP Components



XAMPP

Μετά την εγκατάσταση τρέχουμε το XAMPP Control Panel.

Ξεκινάμε τα Modules:

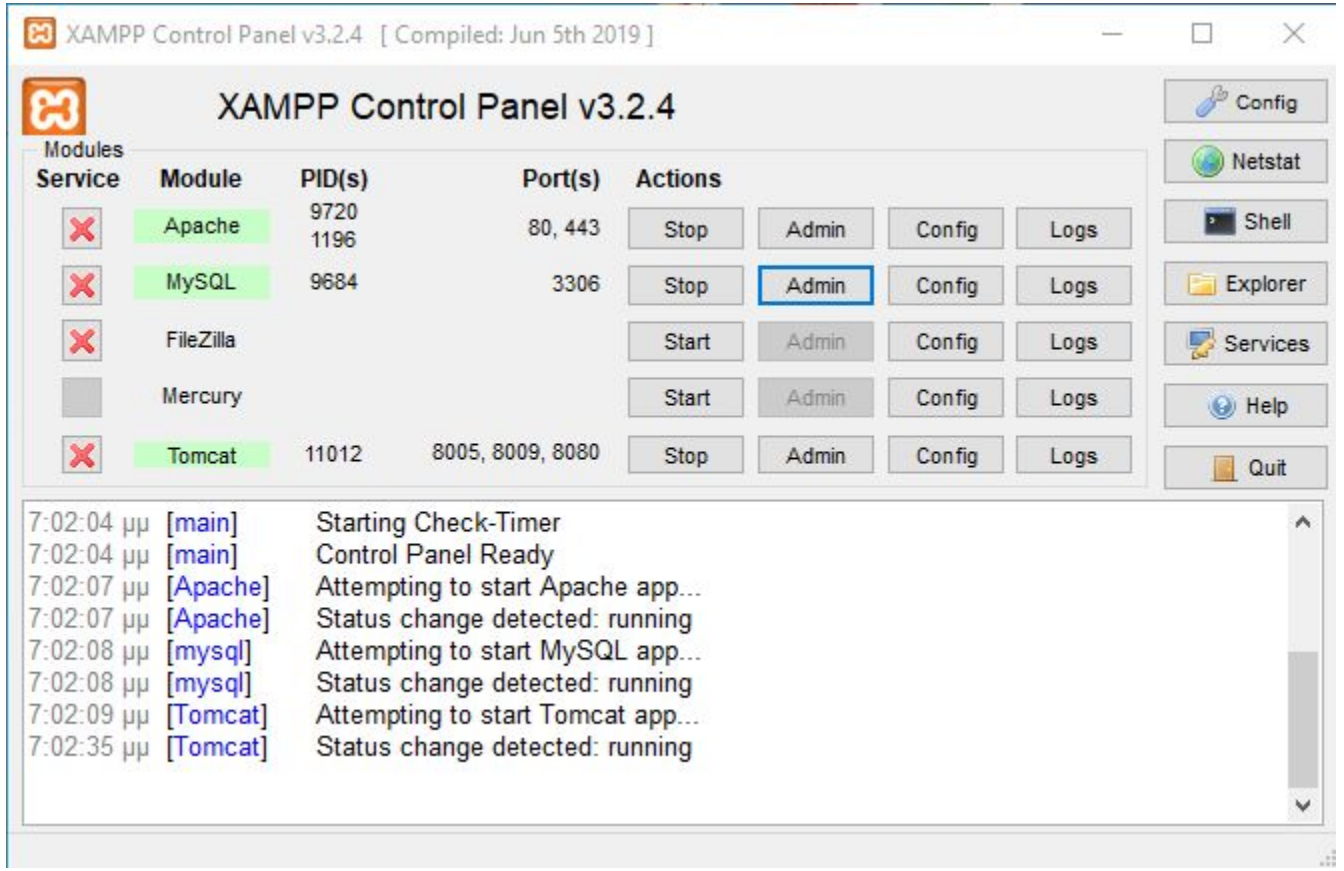
1. Apache
2. MySQL
3. Tomcat

Για τη διαχείρισή τους όπως για παράδειγμα η αλλαγή Port που τρέχουν γίνεται μέσω του config.

Το Control Panel του XAMPP όπως και τα Modules φαίνονται στην παρακάτω εικόνα.

XAMPP

XAMPP Control Panel

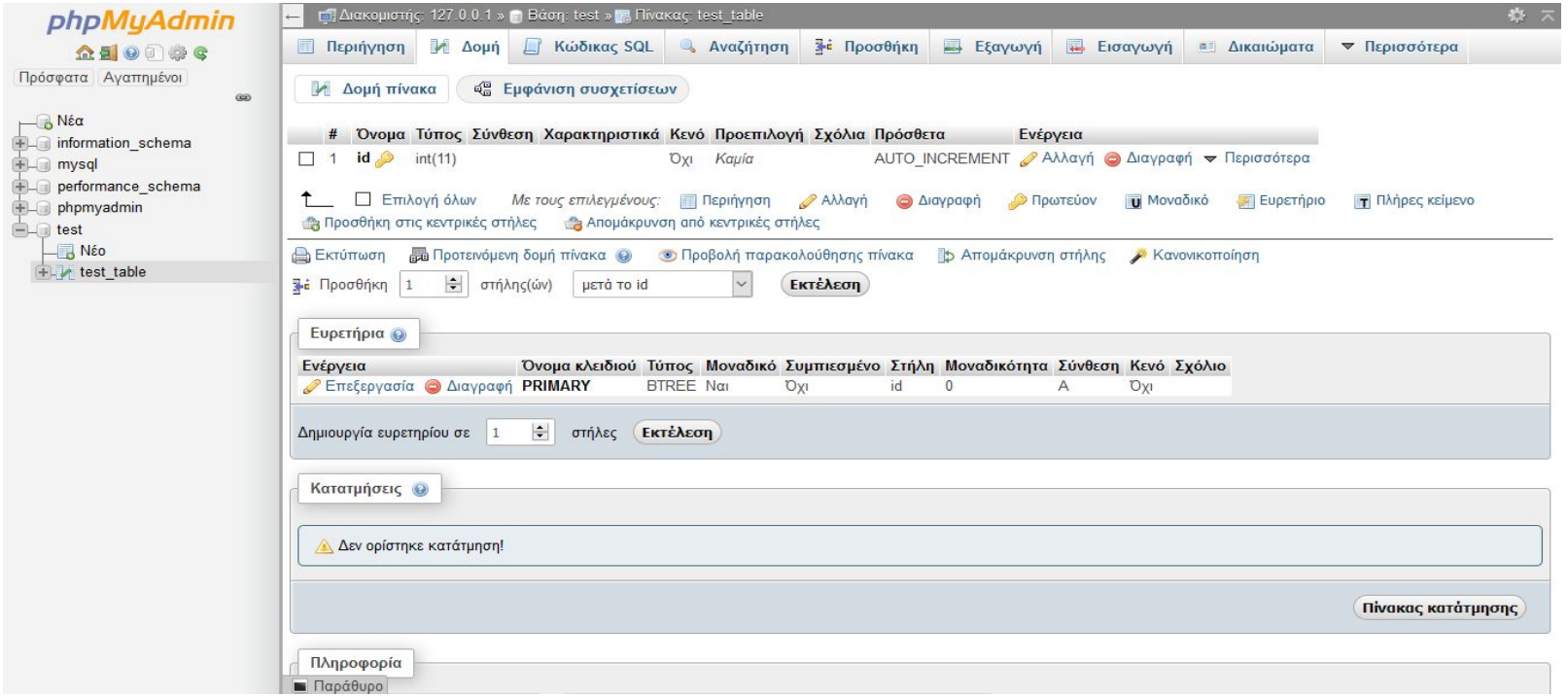


XAMPP

Για τη δημιουργία αλλά και διαχείριση της βάσεως δεδομένων πηγαίνουμε στο Admin του MySQL. Μέσω browser που μας ανοίγει γίνεται η όποια εργασία επιθυμούμε.

**Σε περίπτωση σφάλματος: error processing request 200… , δοκιμάστε να το τρέξετε με διαφορετικό browser.*

Παράδειγμα



JDBC

JDBC: Driver της Java με τον οποίο γίνεται η σύνδεση στη MySQL βάση δεδομένων.

Κάνουμε Import:

```
import java.sql.*;
```

Στις βιβλιοθήκες του IDE μας βάζουμε τη βιβλιοθήκη που πραγματοποιεί τη σύνδεση με τη βάση δεδομένων με το όνομα mysql-connector-java-8.0.22.jar

Το mysql-connector-java-8.0.22.jar το κατεβάζουμε από το παρακάτω σύνδεσμο διαλέγοντας για Operation System το Platform Independent:

https://dev.mysql.com/downloads/connector/j/?os=26

Καθορίζουμε τον Driver που θα χρησιμοποιήσει η Java για τη σύνδεση με τη βάση

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

JDBC

Για να γίνει η σύνδεση δηλώνουμε τη βάση δεδομένων που θέλουμε να συνδεθούμε μαζί με όλα τα Credentials

```
String url = new String("jdbc:mysql://localhost“);
String databaseName = new String(“test“);
int port = 3306;
String username = new String("root“);
String password = new String(""“);                    //Για σύνδεση στη βάση χωρίς κωδικό

Connection con = DriverManager.getConnection(
url + ":" + port + "/" + databaseName + "?characterEncoding=UTF-8", username, password);
```

JDBC: Ενημερώσεις στην βάση

Βήματα για σύνδεση στη βάση δεδομένων και αποστολή επερωτήσεων σε αυτή.

1. Ορίζουμε object τύπου Connection (όπως είδαμε στην προηγούμενη διαφάνεια)
2. Ορίζουμε όσα object τύπου Statement χρειαζόμαστε για επερωτήσεις
3. Η Statement παρέχει μεθόδους αποστολής queries στη βάση δεδομένων
4. Τελειώνοντας όσα Statements θέλουμε πρέπει πρώτα να δώσουμε τέλος στο Statement και έπειτα τέλος στο Connection. Αυτό γίνεται με την μέθοδο close() και για το Statement και για το Connection;

Π.χ για κλείσιμο του Connection:

```
con.close();
```

Παραδείγματα θα δούμε παρακάτω.

JDBC: Ενημερώσεις στην βάση

Παράδειγμα 1

```
String   createEmployee = new String(
" CREATE EMPLOYEE(FNAME varchar(10),"+
"LNAME varchar(10),  SSN varchar(10),
DNO NUMBER, SALARY NUMBER,"+
"PRIMARY KEY(SSN)," +
"FOREIGN KEY(DNO) REFFERENCES DEPARTMENT)");

Statement stmt = con.createStatement();          // δημιουργούμε ένα αντικείμενο το οποίο
                                                 // θα κάνει τις ενημερώσεις / επερωτήσεις
                                                 // στην βάση με την οποία έχουμε συνδεθεί

stmt.executeUpdate(createEmployee);              // η μέθοδος executeUpdate χρησιμοποιείται
                                                 // για να εκτελέσομε τις ενημερώσεις στην
                                                 // βάση
```

JDBC: Ενημερώσεις στην βάση

Παράδειγμα 2

```
String   insertEmployee = new String(
"INSERT INTO EMPLOYEE"+
"VALUES ('george', 'papadopoulos',  '98877', 8,  450000)";

stmt.executeUpdate(insertEmployee));
```

Παράδειγμα 3

```
String   deleteEmployee = new String(
    "DELETE FROM EMPLOYEE"+
     "WHERE LNAME='papadopoulos' ");

stmt.executeUpdate(deleteEmployee);
```

JDBC: Επερωτήσεις στην βάση

ResultSet : object το οποίο μας επιτρέπει να κρατάμε τα αποτελέσματα μιας επερώτησης.

Επομένως η συνάρτηση executeQuery «Executes a SQL statement that returns a single ResultSet».

Το αντικείμενο τύπου ResultSet περιέχει πλειάδες.

Παράδειγμα

```
String   queryEmployee = new String(
        " SELECT  FNAME, LNAME, SALARY" +
        " FROM EMPLOYEE"+
        " WHERE DNO=5");

ResultSet rs = stmt.executeQuery(queryEmployee);          // η μέθοδος executeQuery
                                                          // χρησιμοποιείται για να
                                                          //  εκτελέσομε τις επερωτήσεις
                                                          // στην βάση
```

JDBC: Επερωτήσεις στην βάση

Η μέθοδος next μας δίνει το επόμενο αποτέλεσμα που μας επέστρεψε μια επερώτηση.

Παράδειγμα

```
ResultSet rs = stmt.executeQuery(queryEmployee);

while (rs.next()){
      string  fname = rs.getString(“FNAME”);
      string lname = rs.getString(“LNAME”);
      int sal = rs.gerInt(“SALARY”);
      System.out.println(fname + ”  “ + lname + “ “ + sal);
  }
```

JDBC: Prepared Statements

Το αντικείμενο PreparedStament μας επιτρέπει να εκτελούμε prepared statements (https://www.javatpoint.com/PreparedStatement-interface).

Παράδειγμα

```
String   preparedStament = new String(
        “UPADATE EMPLOYEE” +
        “ SET SALARY= ?”+
        “ WHERE  LNAME= ?”);

PreparedStament updateEmployee = con.preparedStament(preparedStament);

updateEmployee.setInt(1,300000);                    // δίνουμε τιμή στην πρώτη άγνωστη μεταβλητή
updateEmployee.setString(2, ‘papadopoulos’);        // δίνουμε τιμή στην δεύτερη άγνωστη μεταβλητή

updateEmployee.executeUpadate();
```

JDBC: Exceptions

- Η μέθοδος getMessage επιστρέφει ένα string που περιγράφει το λάθος.
- Η μέθοδος getSQLState επιστρέφει ένα string που είναι το αναγνωριστικό του λάθους σύμφωνα με την σύμβαση της SQL.
- Η μέθοδος getErrorCode επιστρέφει ένα ακέραιος ο οποίος είναι ο κωδικός του λάθος

Παράδειγμα

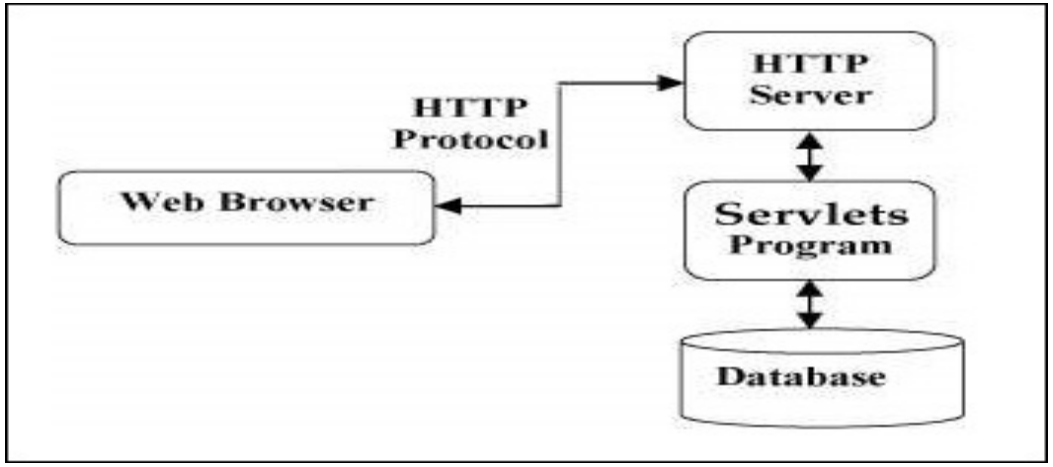
```
try {    // κώδικα που περιέχει τις sql εντολές
        // π.χ ResultSet rs = stmt.executeQuery(queryEmployee);
    }
catch(SQLException ex) {
    System.out.println("\n Exception \n");
      while (ex!=null) {
            System.out.println("Message: " + ex.getMessage());
            System.out.println("SQLState:  "+ ex.getSQLState());
            System.out.println("ErrorCode:  "+getErrorCode());
             ex=ex.getNextException();   }
```

Servlets

- Βήμα 1ο: Εγκατάσταση ενός servlet engine. (tomcat)
- Βήμα 2ο: Χρήση του HttpServlet Class για να μπορείτε να στέλνετε και να λαμβάνετε δεδομένα από το Web.

HttpServlet Class

- Είναι μια κλάση που μας επιτρέπει να επικοινωνούμε με τον web browser μέσω java.
- Δύο κύριες λειτουργίες η Request και Response.
  - Get: Παίρνει πληροφορίες από τον server (HttpServletRequest).
  - Post: Δίνει πληροφορίες στον server (HttpServletResponse). Με την κλάση αυτή μπορούμε να στείλουμε html που θα είναι τα αποτελέσματα μιας επερώτησης.



Servlets

```
import java.io.*;
import java.net.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class MemberAction extends HttpServlet{
public void doGet(HttpServletRequest request, HttpServletResponse response){
int result;
try{
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();          //Ορισμός της μεταβλητής
                                                         //για να γράψουμε στο web
        String login = request.getParameter("login");    //Λαμβάνουμε δεδομένα από το
                                                         //από το Web
        String passwd = request.getParameter("passwd");
        NewMember nmember = new NewMember(login,passwd);
        result = nmember.NewMemberSign();
        if (result == 0){                                //Κάνουμε κάποια ενέργεια
```

Servlets

```
if (result==1){
        String heading=new String("New Member");
        String heading1= new String("Welcome " + login);
        out.println("<HTML>");
        out.println("<HEAD>");          out.println("<TITLE>" + heading + "</TITLE>");     out.println("</HEAD>");
        out.println("<BODY BGCOLOR=\"#000000\">");
        out.println("<TABLE WIDTH=\"100%\">");
        out.println("<TR><TD>");
        out.println("<FONT COLOR=\"YELLOW\">");
        out.println("<H1 ALIGN=\"CENTER\">" + heading1 + "</H1>");
        out.println("<BR>");
        out.println("<H2 ALIGN=\"CENTER\">" + "You become a Member" + "</H2>");
        out.println("<H3 ALIGN=\"CENTER\">" +          "Please go" + "<u><a
        href=\"http://paradeigma:40044/examples/servlets/index.html\" color=\"white\">" + " back " + "</a></u>" +          "and retype
        your login and password" + "</H3>");
        out.println("</FONT></TD></TR></TABLE>");
        out.println("</BODY></HTML>");
}
```

Servlets

```
out.close();
      }
            catch(Exception ex){
            ex.printStackTrace();
             }
 }

 private boolean insertIntoBoat( String StringInsert){
  try {
        statement = con.createStatement();
        statement.execute( “INSERT INTO Boat values (“+
        StringInsert+”)”);
        }
    catch ….
}
```

Servlets

HTML κείμενο.

```
<HTML>
<HEAD> …… </HEAD>
<BODY>
<FORM  ACTION=http://localhost:8080/servlet/ProjectServlet  METHOD=POST>
<PRE>
FID:              <INPUT  TYPE=text  NAME=FID>
Arrival            <INPUT TYPE=text   NAME=ARRIVAL>
Departure        <INPUT TYPE=text   NAME=DEPARTURE>
</PRE>
<p>
<INPUT TYPE=SUBMIT  Value="Submit">
</P>
</FORM>
</BODY>
</HTML>
```

Servlets

Επιστροφή αποτελεσμάτων μιας επερώτησης.

Θα πρέπει να χρησιμοποιήσουμε τον τύπο HttpServletResponse για να ορίσουμε ένα PrintWriter.

Στην συνέχεια πρέπει να ορίσουμε τον κατάλληλο html πίνακα (με χρήση της while που χρησιμοποιούμε για να πάρουμε τα αποτελέσματα μια query) και στην συνέχεια να χρησιμοποιήσουμε τον PrintWriter για να το στείλουμε στην web browser.

Μπορείτε να υλοποιήσετε είτε την doGet είτε την doPost μέθοδο.

ΧΡΗΣΙΜΑ LINKS

Για JDBC:

https://www.tutorialspoint.com/jdbc/index.htm

https://dev.mysql.com/doc/connector-j/5.1/en/connector-j-usagenotes-connect-drivermanager.html

https://www.javatpoint.com/PreparedStatement-interface

Για Servlets:

https://www.tutorialspoint.com/servlets/index.htm

https://www.javatpoint.com/servlet-tutorial

Για πληροφορίες σχετικά με τις εντολές στην Java δείτε το JAVA API.