Lecture 12: Database Connectivity and Exam Preparation

Instructor: Musard Balliu, musard@chalmers.se

http://www.cse.chalmers.se/~musard

QUESTIONS?

Java and Databases

Integration between programming languages and database management systems isn't easy.

In this lecture we will learn how to integrate graphical user interfaces, Java programms and SQLite databases altogether.

Different libraries allow to put ordinary SQL statements in a JAVA source code that interacts with an underlying database.

JDBC

Java Database Connectivity (JDBC) defines a number of interfaces which are implemented by the different database vendors

JDBC allows us to access a database from a Java program, using all the usual SQL statements: CREATE, INSERT, UPDATE, DELETE, SELECT.

The operations are largely independent of the specific database. Changing the kind of database used (e.g., from SQLite to MySQL) should only require minor changes in the program. You can access several kinds of database at the same time.

Connecting to a database

Java needs a JDBC driver that takes care of the communication between the program and the database. These drivers are unique for the database we want to use, i.e. the driver for SQLite is different from the driver for MySQL.

The SQLite JDBC driver (version 3.8.11.2) can be downloaded from https://bitbucket.org/xerial/sqlitejdbc/downloads

The JDBC jar file can be added to the Eclipse ennvironment as done for the Becker's Robot library.

Loading the driver and Connecting to SQLite database

```
import java.sql.*;
public class SQLiteJDBC {
 public static void main( String args[] ) {
   try {
   //load the driver
   Class.forName("org.sqlite.JDBC");
   //connect to SQLite database
   c = DriverManager.getConnection("jdbc:sqlite:test.db");
   System.out.println("Connection successfully opened");
    ...
   c.close();
   } catch ( Exception e ) {
   System.err.println(e.getClass().getName()+" : "
    +  e.getMessage());
   System.exit(0); }
   System.out.println("Driver loaded successfully");
 }
```

Operations on SQLite databases

We will now create a table COMPANY containing NAME, AGE, ADDRESS and SALARY of employees working for the company. Each employee is associated with a unique ID.

Creating a table: Code

Populating the table: Code

Visualizing table contents: Code

Updating the table: Code

Deleting an item: Code

Adding a GUI

We now build a GUI to visualize, insert and delete data from table COMPANY.

To separate the different concerns, we create a class to establish connection with the database, a class to parse the query results and a class to implement the graphical user interface.

Connect to database: Code

Parse query results: Code

Graphical user interface: Code

Putting it all together: Code

Course objectives

- the basics of (imperative) programming: lectures 2–5
- the basics of object-oriented programming: lectures 6–8
- using and understanding APIs for, e.g. graphical user interfaces or database connectivity: lectures 9–12

Testing course objectives

- the basics of (imperative) programming:
  two problems (10 + 20 pts): repairing a simple program ("your boss wrote this"), and a problem from CodingBat (maybe with some changes).

- the basics of object-oriented programming:
  two questions (10 + 10 pts), involving scope, "is a", constructors, `super`, `this`, etc.

- using and understanding APIs for, e.g. graphical user interfaces or database connectivity:
  two problems (30 + 20 pts): along the lines of assignment 2, filling in some implementation for a given set of classes, and using the classes in a small program.

Materials to help prepare exam

- The slides and code from the lectures (I go through them when I prepare the exams).
- Original exam solutions, November 17, 2014
- First re-exam solutions, December 12, 2014
- Second re-exam solutions, August 27, 2015

Important points

- READ THE QUESTIONS!
- Advice from a student: "Go through the previous exams repeatedly! Don't get stuck on a hard question, skip it, and try again tomorrow."
- That also holds for the actual exam: don't spend to much time on hard questions! Try to answer those questions you can, and come back to the more difficult ones later: you have time (four hours!).
- READ THE QUESTIONS!

Course evaluation

After the exam, the evaluation is open in the GUL. The evaluation has to be completed in *three weeks*.

PLEASE PARTICIPATE!

Homework

- Read the solutions to previous exam questions, but try to solve them on your own first!

- Do the coding bat exercises.