UNIVERSITÀ DELLA CALABRIA

DIPARTIMENTO DI **MATEMATICA E INFORMATICA**

Interfacce Grafiche e Programmazione ad Eventi

Carmine Dodaro

Anno Accademico 2019/2020

Database in Java

JDBC

- È l'acronimo di Java Database Connectivity
- API che permette di accedere a tutti i database relazionali allo stesso modo
- Permette ai programmatori di non doversi confrontare con i dettagli del database
- Per usare JDBC è sufficiente conoscere SQL

Funzionamento

- Per ogni DBMS esiste un driver JDBC, messo a disposizione dal produttore del DBMS, al cui interno è gestita la comunicazione con il DBMS specifico.
- JDBC Driver:
  - Contiene un insieme di classi Java che seguono gli standard JDBC
  - Implementano varie funzionalità tra cui connessione e accesso al database, ecc.
- Cosa deve fare un programmatore per usare un database:
  - Caricare il driver JDBC corrispondente al database che si vuole utilizzare
  - Utilizzare le API JDBC per eseguire le varie operazioni sul database

Classi da ricordare

- DriverManager è la classe che permette di gestire i driver JDBC. Fornisce il metodo statico getConnection, il quale riceve come parametro un URL dove si trova il database e restituisce un oggetto Connection
- Connection è la classe che gestitsce una connessione con il database specifico. Tutte le operazioni effettuate (inserimenti o cancellazioni di tuple nel database, query ecc.) sono fatte all'interno della sessione creata da questa connessione.
- Statement è la classe che può essere utilizzata per effettuare un'operazione sul database ed analizzare il risultato ottenuto.
- PreparedStatement è la classe che rappresenta un'operazione SQL precompilata. Questo oggetto può essere utilizzato per effettuare in modo efficiente una determinata operazione più volte.
- ResultSet è una tabella di dati che rappresentano un risultato ottenuto dal database, di solito è generato dopo aver effettuato una query sul database.

JDBC: Esempio di connessione al database

SQLite

1. Scaricare il driver JDBC da https://bitbucket.org/xerial/sqlite-jdbc/downloads/
2. Creare una cartella lib all'interno del vostro progetto (allo stesso livello di src)
3. Copiare il file jar scaricato all'interno della cartella lib
4. Da eclipse, cliccare con il tasto destro sul nome del progetto, scegliere Properties, poi Java Build Path, poi Libraries, poi Add Jars..., poi selezionare il progetto, cartella lib e infine il file jar.

O in alternativa si può usare un tool che ci gestisce in automatico le dipendenze, ad esempio maven.

Connessione al db

```
String url = "jdbc:sqlite:db_name.db";
Connection con = DriverManager.getConnection(url);
if (con != null && !con.isClosed())
    System.out.println("Connected!");
```

JDBC: gestione del database

Creare una tabella

```
String query = "CREATE TABLE IF NOT EXISTS users(id int,
    first_name varchar(50), last_name varchar(50), username
    varchar(50));";
Statement stmt = con.createStatement();
stmt.executeUpdate(query);
stmt.close();
```

Inserire tuple all'interno della tabella

```
Statement stmt = con.createStatement();
stmt.executeUpdate("INSERT INTO users VALUES(1,'mario', 'rossi',
     'ciao1');");
stmt.executeUpdate("INSERT INTO users VALUES(2,'mario', 'bianchi
    ','ciao2');");
stmt.executeUpdate("INSERT INTO users VALUES(3,'francesco', '
    rossi','ciao3');");
stmt.close();
```

JDBC: gestione del database

Rimuovere i dati dall'interno della tabella

```
Statement stmt = con.createStatement();
stmt.executeUpdate("DELETE from users;");
stmt.close();
```

Interrogare il database

```
String query = "select * from users where first_name=?;";
PreparedStatement stmt = con.prepareStatement(query);
stmt.setString(1, "mario");
ResultSet rs = stmt.executeQuery();
System.out.println("Results:");
while(rs.next()) {
  System.out.println("Id: " + rs.getInt(1) + " last name: " + rs
      .getString("last_name"));
}
stmt.close();
```

Qualche osservazione

Uso delle query parametriche

Quando si effettuano delle query, degli inserimenti o delle rimozioni e si riceve come parametro qualcosa da input, è buona norma usare PreparedStatement e settare i parametri segnati con il ?. Ad esempio:

```
//OK:
public void print(String name) {
  String query = "select * from users where first_name=?;";
  PreparedStatement stmt = con.prepareStatement(query);
  stmt.setString(1, "mario");
  ...
}

//NO:
public void print(String name) {
  String query = "select * from users where first_name='"+name+"
      ';";
  ...
}
```

Osservazione sulle password

Password

Quando si utilizza il database per memorizzare delle password, è opportuno non memorizzarle mai in chiaro! Ci sono diverse possibili soluzioni che si possono adottare:

- Bcrypt
- PBKDF2
- Scrypt.

Esempio con spring security

```
//Encrypt
String originalPassword = "password";
String generatedSecuredPasswordHash = BCrypt.hashpw(
    originalPassword, BCrypt.gensalt(12));
System.out.println(generatedSecuredPasswordHash);

//Check
boolean matched = BCrypt.checkpw(originalPassword,
    generatedSecuredPasswordHash);
System.out.println(matched);
```