JDBC

Dr. Günter Kolousek

2012-09-09

Übersicht

Zweck und Aufbau

- Zweck
  - (“low level”) Zugriff auf relationale DB
  - 3 Tätigkeiten
    - Mit DB verbinden
    - Anweisungen über Verbindung senden
    - Ergebnisse lesen
- Aufbau: 4 Teile
  - JDBC API
  - JDBC Driver Manager
  - JDBC Test Suite
  - JDBC-ODBC Bridge

Arten von JDBC Treibern

- Typ 1: Mapping zu anderem DB-API (z.B. ODBC)
- Typ 2: Teilweise in Java, teilweise native
- Typ 3: In Java, via Middleware Server zum DBMS
- Typ 4: In Java, mittels Netzwerk zum DBMS

Sqlite

- http://sqlite.org
- inprocess DB (vs. Client/Server)
- DB in einer Datei
- plattformübergreifend
  - 32 vs. 64 Bit
  - big-endian vs. little-endian
- verschiedene Programmiersprachen
- “small”
- Alternativen: Oracle, MS-SQL, PostgreSQL, MySQL, Java DB (Apache Derby)

Verbindung herstellen

- DriverManager laden

```
1  Class.forName("org.sqlite.JDBC");
```

- Verbindung herstellen

```
1  Connection conn =
2    DriverManager.getConnection("jdbc:sqlite:guestbook.db");
```

- Connection String von Driver abhängig
  - Meist: `jdbc:dbms:dbname`
- überladene Methoden
  - mit Benutzername und Passwort
  - mit `Properties` Objekt (auch Benutzer & Passwort)
- Alles im `java.sql` Paket!

Einfache Anweisungen 1

- Anweisung anlegen

```
1  Statement stat = conn.createStatement();
2  // ...
3  stat.close();
```

- Update

```
1  stat.executeUpdate("drop table if exists guestbook");
2  int count = stat.executeUpdate(
3    "update guestbook set text='x' where email='x@y.z'");
```

Einfache Anweisungen 2

- Query

```
1  ResultSet rs =
2      stat.executeQuery("select * from guestbook");
```

- Allgemein

```
1  boolean type = stat.execute("select * from guestbook");
2  // true -> ResultSet
3  // false -> update count oder kein Ergebnis
```

`ResultSet` 1

- Ergebnismenge einer Abfrage
- verwaltet Cursor zur aktuellen Row (kein DB Cursor!)
- iterierbar mit Methode `next`
  - `true` weitere Row vorhanden
  - `false` keine weitere Row
- Defaultmäßig
  - nur eine Richtung iterierbar
  - nur ein Mal iterierbar
  - nicht veränderbar
- Verschiedene Methoden für Spaltenzugriff
  - mittels Name, z.B.: `rs.getString("email");`
  - mittels Spaltenindex (beginnt mit 1!), z.B. `rs.getInt(1);`
  - verschiedene Typen, z.B.: `getBoolean`, `getDate`, `getDouble`

ResultSet 2

```
1  ResultSet rs = stmt.executeQuery("SELECT a, b, c FROM tab1");
2
3  while (rs.next()) {
4      int a = rs.getInt("a");
5      String b = rs.getString(2);
6      float c = rs.getFloat("c");
7  }
```

ResultSet 3

- Verändern:

```
1  rs.updateString("email", "x.y@example.com");
2  rs.updateRow();
```

- Einfügen:

```
1  rs.moveToInsertRow();
2  rs.updateInt("age", 46);
3  rs.insertRow();
4  rs.moveToCurrentRow();
```

- Löschen: `rs.deleteRow();`

Transaktionen

- Default: autocommit
  - jede SQL-Anweisung = eigene Transaktion
- `conn.setAutoCommit(false);`
- `conn.commit();`
- `conn.rollback();`
- Weiters, z.B.:
  - `Savepoint s1 = conn.setSavepoint(); ...`
    `conn.rollback(s1);`
  - `conn.releaseSavepoint(s1);`
  - Isolation Levels:
    - `out.println(conn.getTransactionIsolation())`
    - `int level =`
      `Connection.TRANSACTION_READ_COMMITTED;`
      `conn.setTransactionIsolation(level);`

PreparedStatement

- beliebig oft ausführbar (mit verschiedene Werten)
- mit SQL Statement angelegt
- zum DBMS gesendet, dort übersetzt
- Beispiel:

```
1  String update = "update users " +
2    "set email= ? where name= ?";
3  PreparedStatement prep = conn.prepareStatement(update);
4  while (evenMoreUpdates) {
5      prep.setString(1, "a@b.c");
6      prep.setString(2, "a");
7      int count = prep.executeUpdate();
8      // auch: executeQuery bzw. execute
9  }
```

Alternativen

- Object-Relational Mapping
  - JEE: Java Persistence API (JPA)
  - Hibernate (auch mit JPA)
  - Apache JDO: Übermenge von JPA, “beliebige Datenquellen”
    - DataNucleus: RDBMS, Excel, XML, JSON, db4o, LDAP
- OO Datenbank
  - db4o für Java **und** .Net
- NoSQL
  - Dokumentenorientiert: MongoDB
  - Graphen DB: Neo4j
  - Key-value: Berkley DB
    - Datenstrukturorientiert: Redis
  - …