Open Source RDBMS for J2ME

jMeSQL version 0.1 connection and usage sample.

06.06.2007

Elizeu Nogueira da Rosa Jr.

http://www.jmesql.net

USING THE DATABASE MANAGER

jMeSQL - Copyright (c) 2007,2007 - All rights reserved.

Once a connection is established to a database in any mode, JDBC methods are used to interact with the database. Use the jdbcConnection, jdbcDriver, jdbcDatabaseMetadata, jdbcResultSet, jdbcStatement and jdbcPreparedStatement.
The database access methods use SQL commands to perform actions on the database and return the results either as a Java primitive type or as an instance of the br.com.dreamsource.mobile.jmesql.jdbcResultSet class.

The SQL dialect used in jMeSQL is as close to the SQL92 standard as it has been possible to achieve so far in a small footprint database engine. The full list of SQL commands is in jMeSQL Syntax documentation.

THE CONNECTION SAMPLE

```
import br.com.dreamsource.mobile.jmesql.exceptions.SQLException;
import br.com.dreamsource.mobile.jmesql.io.File;
import br.com.dreamsource.mobile.jmesql.io.Properties;
import br.com.dreamsource.mobile.jmesql.jdbcConnection;
import br.com.dreamsource.mobile.jmesql.jdbcDriver;
import br.com.dreamsource.mobile.jmesql.jdbcPreparedStatement;
import br.com.dreamsource.mobile.jmesql.jdbcResultSet;
import br.com.dreamsource.mobile.jmesql.jdbcStatement;
import java.util.Enumeration;
import java.util.Vector;

public class DatabaseManager {

    /*
     * the database connection
     */
    private static jdbcConnection connection;

    /*
     * Properties contains ("user", "password", "database")
     */
    private static Properties props;

    public DatabaseManager(String databaseName, String userName, String passwd) throws SQLException {
        //Setting the properties
        props = new Properties();
        props.setProperty("user", userName);
        props.setProperty("password", passwd);
        props.setProperty("database", databaseName);

        if (this.connection == null) {
            jdbcDriver driver = new jdbcDriver();

            /*
             * If the database does not exists,
             * the dbms create a new, default
             * user name is "SA" and
             * password is empty
             */

            this.connection = driver.connect(this.props);
```

jMeSQL - Copyright (c) 2007,2007 - All rights reserved.

```
    }
}

private jdbcConnection getConnection() throws SQLException {
    if (this.connection == null) {
        throw new SQLException("The connection is null!");
    }
    return this.connection;
}

private jdbcPreparedStatement getPreparedStatement(String sqlValue) throws SQLException {
    return this.getConnection().prepareStatement(sqlValue);
}

/*
* Execute de SELECT statement
* Ex: String sqlValue = "SELECT * FROM MYTABLE"
*     jdbcResultSet resultSet = myDatabaseManager.list(sqlValue);
*/
public jdbcResultSet list(String sql) throws SQLException {
    jdbcPreparedStatement prepared = this.getPreparedStatement(sql);
    return prepared.executeQuery();
}

/*
* Execute de SELECT statement based in Vector attributes
* Ex: String sqlValue = "SELECT * FROM MYTABLE WHERE ID = ?"
*     Vector values = new Vector();
*     values.addElement("1");
*     jdbcResultSet resultSet = myDatabaseManager.find(sqlValue, values);
*/
public jdbcResultSet find(String sqlValue, Vector values) throws SQLException {
    jdbcPreparedStatement preparedStatement = this.getPreparedStatement(sqlValue);
    for (int i = 0; i < values.size(); i++) {
        preparedStatement.setString(i + 1, (String)values.elementAt(i));
    }
     return preparedStatement.executeQuery();
}

/*
* Execute de INSERT statement based in Vector attributes
* Ex: String sqlValue = "INSERT INTO MYTABLE (ID, NAME) VALUES (?, ?)"
*     Vector values = new Vector();
*     values.addElement("1");
*     values.addElement("aaa");
*     myDatabaseManager.insert(sqlValue, values);
*/
public boolean insert(String sqlValue, Vector values) throws SQLException {
    return this.executeUpdateSQL(sqlValue, values);
}

/*
* Execute de UPDATE statement based in Vector attributes
```

jMeSQL - Copyright (c) 2007,2007 - All rights reserved.

```
    * Ex: String sqlValue = "UPDATE MYTABLE SET ID = ?, NAME = ? WHERE ID = ?"
    *     Vector values = new Vector();
    *     values.addElement("2");
    *     values.addElement("aaa");
    *     values.addElement("1");
    *     myDatabaseManager.update(sqlValue, values);
    */
    public boolean update(String sqlValue, Vector values) throws SQLException {
        return this.executeUpdateSQL(sqlValue, values);
    }

    /*
    * Execute de UPDATE statement based in Vector attributes
    * Ex: String sqlValue = "DELETE FROM WHERE ID = ?"
    *     Vector values = new Vector();
    *     values.addElement("2");
    *     myDatabaseManager.delete(sqlValue, values);
    */
    public boolean delete(String sqlValue, Vector values) throws SQLException {
        return this.executeUpdateSQL(sqlValue, values);
    }

    private boolean executeUpdateSQL(String sqlValue, Vector values) throws SQLException {
        jdbcPreparedStatement preparedStatement = this.getPreparedStatement(sqlValue);
        for (int i = 0; i < values.size(); i++) {
            preparedStatement.setString(i + 1, (String)values.elementAt(i));
        }
        if (preparedStatement.executeUpdate() > 0) {
            return true;
        } else {
            return false;
        }
    }

}
```

The red text is sample use in your midlet.

jMeSQL - Copyright (c) 2007,2007 - All rights reserved.