

# MyMIDP: An JDBC driver for accessing MySQL from mobile devices

Essam Mansour

March 2009

- 1 Introduction
  - Background
  - Related Work
  - Research Problem
  - Research Objective
- 2 MyMIDP
  - Class Overview
  - Usage Scenario
  - Limitations
  - MyMIDP-Client
- 3 Conclusion
- 4 Contact Me

- Mobile devices, such as cell or smart phones, are
- no longer voice communication devices, but also
  - small footprinted information system (IS) clients, and
  - an environment for third party applications.

- A certified collection of Java APIs for the development of software for small devices
- A minimal building blocks and a basic runtime environment for java applications
- Each implementation comes with certain restrictions, such as the size limitation of Jar files.

# Background: Mobile Information Device Profile (MIDP)

DB&IS  
Research  
Group

Introduction

Background

Related Work

Research  
Problem

Research  
Objective

MyMIDP

Class Overview

Usage Scenario

Limitations

MyMIDP-Client

Conclusion

Contact Me

- Is a part of the Java ME framework
- Sits on top of Connected Limited Device Configuration (CLDC)
- Adds APIs for user interface, application model and persistence storage

## \* Database support for mobile devices:

- Lightweight database management system for mobile devices
- Middle-ware approach for synchronizing data between client and server

## \* Limitations:

- handling replicated data but not for client/server data access
- no support for MIDP devices

- There is a lack of standardized software for combining IS and mobile devices
- Most existing approaches concentrate on synchronization aspects or use an additional middle-ware
- Java desktop applications often use an JDBC driver to access servers without additional middle-ware

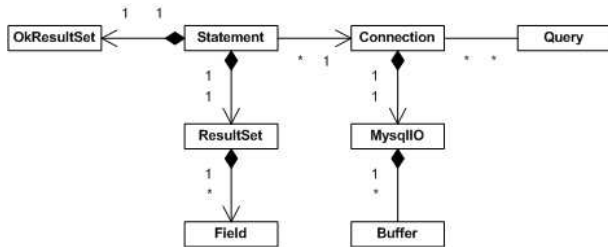
## Problem Statement

Move the method of accessing databases on remote servers from middle-ware over MIPD to MIDP itself using the concept of JDBC.

## Objective Statement

Develop an MIDP-based Java ME driver for MySQL similar to JDBC that allows direct communication of MIDP applications to MySQL servers without a middleware.

- Keep the driver API as near to the JDBC specification as possible
- Keep the Jar file size below 32kB half the popular 64kB limit to leave enough space for the application
- Only implement required features
- Keep the implementation code as simple and performant as possible



Class Overview

Developer familiar with JDBC will not have any problems using our driver, MyMIDP, as follows:

- Create a database connection
- Create and execute a database statement
- Process the result set
- Close the connection

# MyMIDP: Usage Scenario

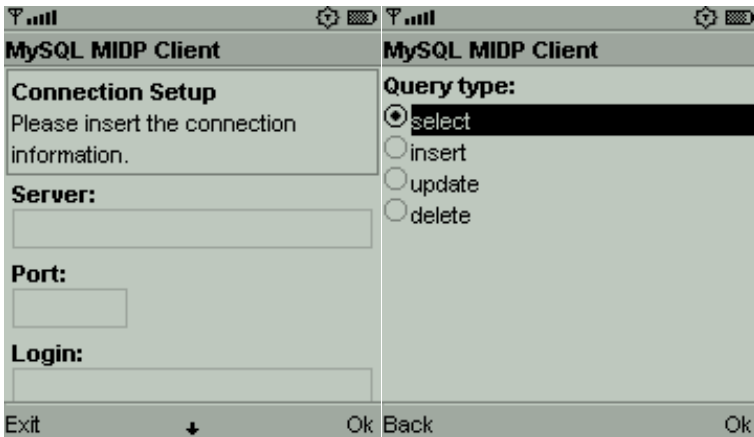
```

import de.iu.db.mysql.mini.Connection;
import de.iu.db.mysql.mini.ResultSet;
import de.iu.db.mysql.mini.Statement;
import de.iu.db.mysql.mini.exceptions.SQLException;
public class Demo {
    public static void main(String[] args) {
        try {
            // connecting to database 'catsanddogs' on server
            // test.somenetwork.net:3006, user 'test', pwd 'run'
            Connection con = new
                Connection("test.somenetwork.net", 3006,
                    "test", "run", "catsanddogs");
            // retrieve some data
            Statement st = con
                .createStatement
                ("SELECT name,age,owner FROM dogs");
            ResultSet rs = st.executeQuery();
            // loop through the result set
            for (; rs.current() < rs.getResultCount(); rs.next()) {
                // access the data using row and column pointer
                System.out
                    .println("The Dog" + rs.getAsInt(0) + " ("
                        + rs.getAsInt(1) + ") is owned by "
                        + rs.getString(2));
            }
            // adding some data
            long count = st
                .executeUpdate
                ("INSERT INTO dogs(name,age,owner)" +
                    "VALUES('Angel',12,'Charlie')");
            System.out.println
                ("Added" + count + " datasets with message:"
                    + st.getMessage());
            // end the session
            con.close();
        }
    }
}

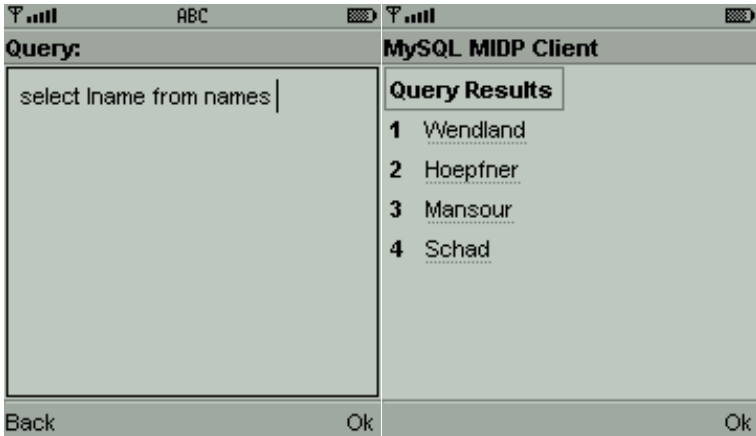
```

Due to the restrictions of the CLDC/MIDP runtime, we had to compromise and drop a number of features available in the normalMySQL JDBC driver, such as:

- Limited character set support
- No support for prepared statements
- Limited data type support
- No transaction handling
- No meta data analysis
- Limited error handling



Screenshots of MyMIDP-client



Screenshots of MyMIDP-client

- Our MyMIDP driver is MIDP-based Java ME driver for MySQL
- Direct communication of MIDP applications to MySQL servers without a middle-ware
- The limitations of MyMIDP mostly result from the restrictions of Java ME and MIDP

*NOTE: The MyMIDP sources and the MyMIDP-client prototype implementation are GPL licensed and available at <http://it.i-u.de/dbis/MyMIDP>.*

- Complete the driver with the communities help
- Using caching strategies that reduces the retransmissions of data
- Our MyMIDP driver is a pre-step in supporting our research in context-aware query processing for mobile devices

## Thank You

Dr. Essam Mansour  
Research Associate  
School of Information Technology  
International University in Germany  
**Email** : [essam.mansour@ieee.org](mailto:essam.mansour@ieee.org)  
**Web page**: <http://it.i-u.de/dbis>