JDBC application using Java for MS Access

A JDBC driver is a software component enabling a Java application to interact with a database. To connect with individual databases, it requires drivers for each database. The JDBC driver gives out the connection to the database and implements the protocol for transferring the query and result between client and database.

Before Java 8: It uses JDBC-ODBC bridge (in particular for MS Access) to converts JDBC method calls into ODBC function calls. The use of this driver leads to other installation dependencies; for example, ODBC must be installed on the computer having the driver and the database must support an ODBC driver. The use of this driver is discouraged if the alternative of a pure-Java driver is available.

Please consult the respective handouts at our drive, as an example.

Using Java 8 or later: The JDBC-ODBC Bridge has been removed from Java SE 8 and is not supported now. The ODBC driver from Microsoft only works in Windows, and there are separate 32-bit and 64-bit versions of the Access Database Engine (and ODBC driver) which can be a nuisance for deployment.

Now, the solution is *UCanAccess,* which is a pure Java JDBC driver that allows us to read from and write to Access databases without using ODBC. Since it is a pure Java implementation, it runs on both Windows and non-Windows operating systems (e.g., Linux/unix). It uses two other packages, *Jackcess* (MS Access input/output library http://jackcess.sourceforge.net/) and *HSQLDB* (HSQLDB as synchronized DBMS http://hsqldb.org/), to perform these tasks. Hence, it has its own dependencies. In short, it require to include the following components to use *UCanAccess*:

UCanAccess (ucanaccess-x.x.x.jar)
HSQLDB (hsqldb.jar, version 2.2.5 or newer)
Jackcess (jackcess-2.x.x.jar)
commons-lang (commons-lang-2.6.jar, or newer 2.x version)
commons-logging (commons-logging-1.1.1.jar, or newer 1.x version)

Difference between Java 1.7 and Java 1.8 versions?

- Until **Java 1.7** version, we are using Jdbc-Odbc bridge to connect MS Access database using the JDBC driver class **sun.jdbc.odbc.JdbcOdbcDriver** (Please see handouts for this purpose)
- Whereas in **Java 1.8** version, *UCanAccess* driver should be used to connect to MS Access database using driver class **net.ucanaccess.jdbc.UcanaccessDriver**

How to use *UCanAccess*?

Fortunately, ***UCanAccess* includes all of the required JAR files in its distribution file** (http://ucanaccess.sourceforge.net/site.html). Upon unzipping, you will see something like:

```
ucanaccess-4.0.1.jar
  /lib/
    commons-lang-2.6.jar
    commons-logging-1.1.1.jar
    hsqldb.jar
    jackcess-2.1.6.jar
```

All you need to do is add all these **five (5)** JARs to your project using one of the following way:

1. **CLASSPATH:** In simple set CLASSPATH variable using either your command prompt or environmental variables.
2. **Eclipse:** Right-click the project in Package Explorer and choose Build Path > Configure Build Path.... Click the "Add External JARs..." button to add each of the five (5) JARs
3. **NetBeans:** Expand the tree view for your project, right-click the "Libraries" folder and choose "Add JAR/Folder...", then browse to the JAR file.

CLASSPATH Example:
Suppose the required libraries are available at the following path:

For MS Access: C:\UCanAccess-4.0.4-bin\ucanaccess-4.0.4.jar;
C:\UCanAccess-4.0.4-bin\lib\commons-lang-2.6.jar;
C:\UCanAccess-4.0.4-bin\lib\commons-logging-1.1.3.jar;
C:\UCanAccess-4.0.4-bin\lib\hsqldb.jar;
C:\UCanAccess-4.0.4-bin\lib\jackcess-2.1.11.jar;

For MySQL: C:\MySQL\mysql-connector-java-5.0.8-bin.jar;

Concatenate them and set the CLASSPATH variable:

```
set CLASSPATH=.;C:\UCanAccess-4.0.4-bin\ucanaccess-4.0.4.jar;C:\UCanAccess-4.0.4-bin\lib\commons-lang-2.6.jar;C:\UCanAccess-4.0.4-bin\lib\commons-logging-1.1.3.jar;C:\UCanAccess-4.0.4-bin\lib\hsqldb.jar;C:\UCanAccess-4.0.4-bin\lib\jackcess-2.1.11.jar;C:\MySQL\mysql-connector-java-5.0.8-bin.jar;
```

Source Code Examples:
Please consult our shared drive.