Apache Derby for Eclipse

Written by Andrew Fierro

Requirements

The following system was used in writing this document

- Windows or Mac OS X
- Eclipse IDE for Java Developers (available at www.eclipse.org)
- Java 5 or Java 6

What is Apache Derby?

Derby is a relational database implemented in Java. It complies with Java, JDBC, and SQL standards. Furthermore, it provides at an embed driver that allows the database to run in the same JVM as the application without the need for separate processes to start or stop. Also supported is a network server for a client/server application.

Installation

After you have installed Eclipse, you need to download Apache Derby. This is actually a plug in for eclipse which makes the installation really simple.

On the downloads page, download

- derby_core_plugin_VERSION.zip
- derby_ui_plugin_VERSION.zip

The downloads page can be found at http://db.apache.org/derby/derby_downloads.html. Go ahead and download the latest release. Also, [3] and [4] are the direct links at the time of this document's writing.

After downloading, unzip both of these files into your eclipse/ directory (inside the zip is already a /plugins folder). After doing this, you should have 2 folders named

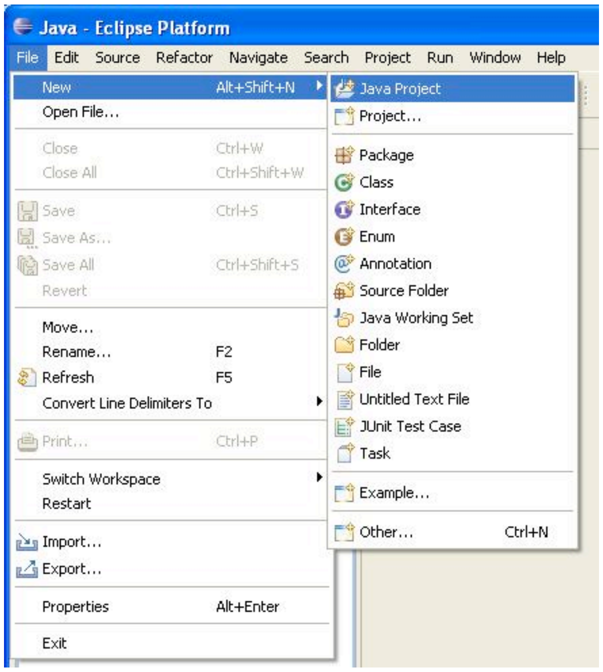
- org.apache.derby.core_10.4.2
- org.apache.derby.ui_1.1.2

in your eclipse/plugins directory. Apache Derby should now be installed!

Configuring your project with Apache Derby in Eclipse

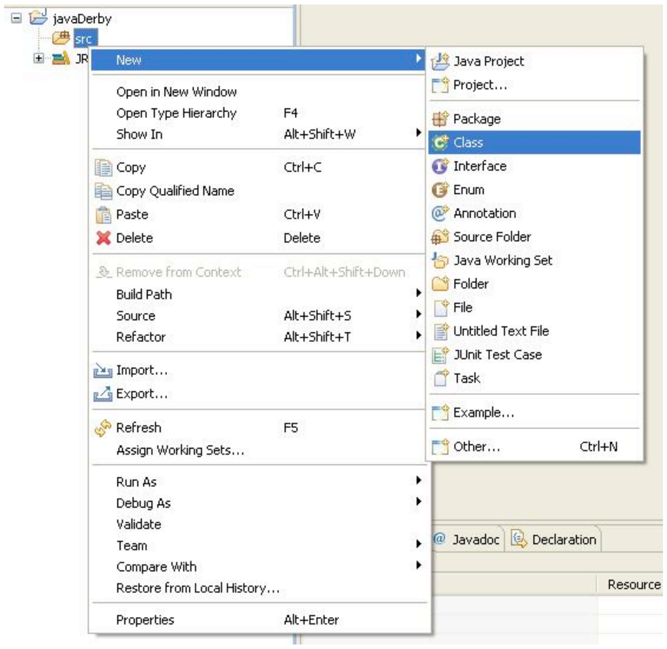
Open up Eclipse now. When asked for a workspace, the default usually works well.

Select New->Java Project



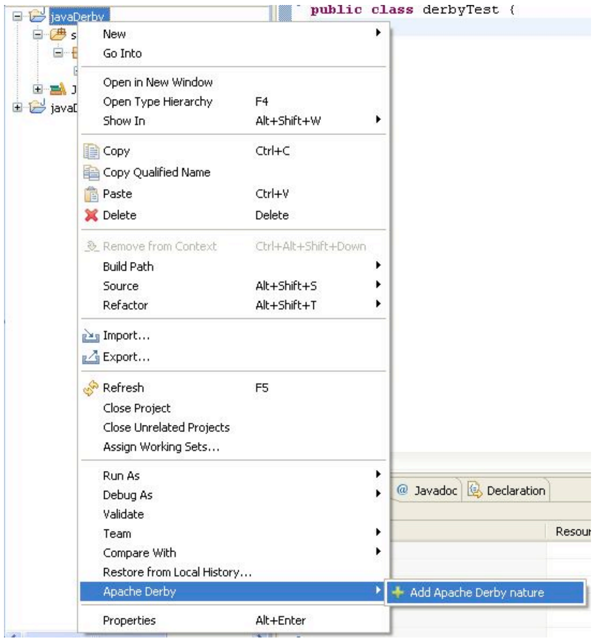
Name the project and click 'Finish'.

Create a new class source file



Name your class and click 'Finish'.

Now, right-click your project and go to the menu Apache Derby and click ‘Add Apache Derby nature’



This will add the correct .jar files to your project and to your build path. Now you will be able to access the derby database.

Coding with Apache Derby

Most of this was taken from Adam Turner’s write up. There are some try and catch statements that need to be inserted that you will see in the section, ‘Complete Example’.

Load the JDBC Driver

```
// JDBC Driver for Derby
String driver = “org.apache.derby.jdbc.EmbeddedDriver”;
// Load the driver
Class.forName(driver).newInstance();
```

Connect to the database via database URL

```
// Protocol
String protocol = “jdbc:derby:”;
// Path for database
String databaseDir = “derbyDB”;
// Username and password for the database
String user = “smith”;
String pass = “pass”;
// Create a connection to the database.  Create=true causes a new database to be
created if one does not exist
Connection connection = DriverManager.getConnection(protocol + databaseDir +
“;create=true”, user, pass);
```

Statement format

```
Statement statement = connection.createStatement();
statement.execute(“some SQL statement”);
```

Shut down the database

```
DriverManager.getConnection(protocol + databaseDir + “;shutdown=true”);
```

All of this is shown in the complete example with comments.

More Features

Apache Derby plug in for eclipse also comes with a network server for client/server applications and ij, a tool to run SQL commands against the network server. For more information on these utilities, see [1].

Complete Example

This program checks to see if a table exists, creates one if not, and then pulls values from the table. The first time you run the program the table will be created, the second time the values are returned.

```
// Must be imported for SQL
import java.sql.*;

public class DerbyTest {

        public static void main(String[] args){

        // Declare variables
        String databaseDir = "derbyDB";
        String driver = "org.apache.derby.jdbc.EmbeddedDriver";
        String protocol = "jdbc:derby:";
        String user = "smith";
        String pass = "pass";
        int defaultRadius;
        int defaultWidth;

        // Load the Driver
         try {
             Class.forName(driver).newInstance();
             System.out.println("Loaded the appropriate driver");
         } catch (ClassNotFoundException cnfe) {
             System.err.println("\nUnable to load the JDBC driver " + driver);
             System.err.println("Please check your CLASSPATH.");
             cnfe.printStackTrace(System.err);
         } catch (InstantiationException ie) {
             System.err.println(
                          "\nUnable to instantiate the JDBC driver " + driver);
             ie.printStackTrace(System.err);
         } catch (IllegalAccessException iae) {
             System.err.println(
                          "\nNot allowed to access the JDBC driver " + driver);
             iae.printStackTrace(System.err);
         }

        // Create the connection
         try {
          Connection connection = DriverManager.getConnection(protocol +
databaseDir + ";create=true", user, pass);

          Statement s = connection.createStatement();



          DatabaseMetaData dbmd = connection.getMetaData();
          ResultSet results = dbmd.getTables(null, null, "CIRCLE_INFO", null);
         // See if table exists, if not create it
        if(results.next() && results.getString(3).equals("CIRCLE_INFO")) {
                  System.out.println("Table already exists... moving on.");
          }
          else {
                  // Create table called CIRCLE_INFO with 2 columns
                  s.execute("CREATE TABLE CIRCLE_INFO (" +
                                  "RADIUS INT, " +
                                  "BORDER_WIDTH INT)");
          }
        // Get result set from CIRCLE_INFO
          ResultSet rs = s.executeQuery("select * from CIRCLE_INFO");
        // If the table is empty, we will insert values into it
          if(!rs.next()) {
                  s.execute("insert into CIRCLE_INFO values(50, 4)");
                  defaultWidth = 4;
                  defaultRadius = 50;
                  System.out.println("inserting... width=4 and radius=50");
          }
          else{
                    // Table is not empty, so get the values and output them
                  defaultRadius= rs.getInt(1);
                  defaultWidth = rs.getInt(2);
                  System.out.println("Radius=" + defaultRadius);
                  System.out.println("Width=" + defaultWidth);
          }
            // Shutdown derby database
             try {
                 // the shutdown=true attribute shuts down Derby
                 DriverManager.getConnection("jdbc:derby:;shutdown=true");
             }
             catch (SQLException se) {
                 if (( (se.getErrorCode() == 50000)
                          && ("XJ015".equals(se.getSQLState()) ))) {
                      // we got the expected exception
                      System.out.println("Derby shut down normally");
                      // Note that for single database shutdown, the expected
                      // SQL state is "08006", and the error code is 45000.
                 } else {
                      // if the error code or SQLState is different, we have
                      // an unexpected exception (shutdown failed)
                      System.err.println("Derby did not shut down normally");
                      printSQLException(se);
                 }
             }
         }
         catch (SQLException sqle) {
          printSQLException(sqle);

         }
        }
        // This function just prints the errors, if any, caused in the SQL
        public static void printSQLException(SQLException e)
            {

                while (e != null)
                {
                    System.err.println("\n----- SQLException -----");
                    System.err.println("  SQL State:  " + e.getSQLState());
                    System.err.println("  Error Code: " + e.getErrorCode());
                    System.err.println("  Message:    " + e.getMessage());
                    // for stack traces, refer to derby.log or uncomment this:
                    //e.printStackTrace(System.err);
                    e = e.getNextException();
                }
            }
}
```

References

[1] Apache Derby DB Project, 2009. http://db.apache.org/derby/

[2] Adam Turner. Working with Apache Derby.

[3] Apache Derby Plugin, 2009. http://apache.mirror99.com/db/derby/db-derby-10.4.2.0/derby_core_plugin_10.4.2.zip. At time of release of this document.

[4] Apache Derby UI, 2009. http://apache.mirror99.com/db/derby/db-derby-10.4.2.0/derby_ui_plugin_1.1.2.zip. At time of release of this document.