IBATIS - DEBUGGING

It is easy to debug your program while working with iBATIS. iBATIS has built-in logging support and it works with the following logging libraries and searches for them in this order.

- Jakarta Commons Logging *JCL*.
- Log4J
- JDK logging

You can use any of the above listed libraries along with iBATIS.

Debugging with Log4J

Assuming you are going to use Log4J for logging. Before proceeding, you need to cross-check the following points −

- The Log4J JAR file *log4j − version. jar* should be in the CLASSPATH.
- You have log4j.properties available in the CLASSPATH.

Following is the log4j.properties file. Note that some of the lines are commented out. You can uncomment them if you need additional debugging information.

```
# Global logging configuration
log4j.rootLogger = ERROR, stdout

log4j.logger.com.ibatis = DEBUG

# shows SQL of prepared statements
#log4j.logger.java.sql.Connection = DEBUG

# shows parameters inserted into prepared statements
#log4j.logger.java.sql.PreparedStatement = DEBUG

# shows query results
#log4j.logger.java.sql.ResultSet = DEBUG

#log4j.logger.java.sql.Statement = DEBUG

# Console output
log4j.appender.stdout = org.apache.log4j.ConsoleAppender
log4j.appender.stdout.layout = org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern = %5p [%t] - %m%n
```

You can find the complete documentation for Log4J from Apaches site − Log4J Documentation.

iBATIS Debugging Example

The following Java class is a very simple example that initializes and then uses the Log4J logging library for Java applications. We would use the above-mentioned property file which lies in CLASSPATH.

```
import org.apache.log4j.Logger;

import com.ibatis.common.resources.Resources;
import com.ibatis.sqlmap.client.SqlMapClient;
import com.ibatis.sqlmap.client.SqlMapClientBuilder;

import java.io.*;
import java.sql.SQLException;
import java.util.*;
```

```
public class IbatisUpdate{
   static Logger log = Logger.getLogger(IbatisUpdate.class.getName());

   public static void main(String[] args) throws IOException,SQLException{

      Reader rd = Resources.getResourceAsReader("SqlMapConfig.xml");
      SqlMapClient smc = SqlMapClientBuilder.buildSqlMapClient(rd);

      /* This would insert one record in Employee table. */
      log.info("Going to update record.....");
      Employee rec = new Employee();
      rec.setId(1);
      rec.setFirstName( "Roma");
      smc.update("Employee.update", rec );
      log.info("Record updated Successfully ");

      log.debug("Going to read records.....");
      List <Employee> ems = (List<Employee>)
         smc.queryForList("Employee.getAll", null);
      Employee em = null;

      for (Employee e : ems) {
         System.out.print("  " + e.getId());
         System.out.print("  " + e.getFirstName());
         System.out.print("  " + e.getLastName());
         System.out.print("  " + e.getSalary());
         em = e;
         System.out.println("");
      }
      log.debug("Records Read Successfully ");
   }
}
```

Compilation and Run

First of all, make sure you have set PATH and CLASSPATH appropriately before proceeding for compilation and execution.

- Create Employee.xml as shown above.
- Create Employee.java as shown above and compile it.
- Create IbatisUpdate.java as shown above and compile it.
- Create log4j.properties as shown above.
- Execute IbatisUpdate binary to run the program.

You would get the following result. A record would be updated in the EMPLOYEE table and later, the same record would be read from the EMPLOYEE table.

```
DEBUG [main] - Created connection 28405330.
DEBUG [main] - Returned connection 28405330 to pool.
DEBUG [main] - Checked out connection 28405330 from pool.
DEBUG [main] - Returned connection 28405330 to pool.
   1  Roma  Ali  5000
   2  Zara  Ali  5000
   3  Zara  Ali  5000
```

Debug Methods

In the above example, we used only **info** method, however you can use any of the following methods as per your requirements –

```
public void trace(Object message);
public void debug(Object message);
public void info(Object message);
public void warn(Object message);
```

```
public void error(Object message);
public void fatal(Object message);
```