JDBC Datasource

4.0 DataSource

DataSource object is a factory for connection for the particular database it represents.An object that implements the DataSource interface will be registered with a naming service based on the Java Naming and Directory (JNDI) API. A data source can reside on a remote server, or it can be on a local desktop machine.

The DataSource interface is implemented by a driver vendor. There are three types of implementations,

4.0.1 Basic implementation

Produces a standard Connection object.

4.0.2 Connection pooling implementation

Produces a Connection object that will automatically participate in connection pooling. This implementation works with a middle-tier connection pooling manager.

4.0.3 Distributed transaction implementation

Produces a Connection object that may be used for distributed transactions and almost always participates in connection pooling. This implementation works with a middle-tier transaction manager and almost always with a connection pooling manager. Since the Datasource object is registered with JNDI and its properties can be modified when necessary. The benefit is that this change in property will not change in the code which is accessing this Datasource.

Datasource object can be retrieved through JNDI lookup and then used to create a Connection Object. With the above Basic Implementation in Datasource, a connection object retrieved from datasource is identical to a connection object retrieved from DriverManager.

4.1 Advantages over the DriverManger

Datasource object register the logical name with JNDI. The application uses only the logical name to retrieve the Datasource object associated with that name.

No need to hardcode the driver informaiton with the Datasource object via the application code.

Datasource provides the connection pooling feature. Connection pooling can increase the performance by reusing the connections instead of getting physical connection each time when requested. Creating a new physical connection is costlier.

DataSource object provide Distributed transaction feature. Distributed transaction allows an application to do the heavy duty database work of large enterprises.

4.2 References

http://download.oracle.com/javase/1.4.2/docs/guide/jdbc/getstart/datasource.html

http://java.dzone.com/articles/jdbc-faq-getting-started