NuoDB and Hibernate: Sample Program

Edit Share Add Tools

Added by Elisabete Vail, last edited by Elisabete Vail on Aug 02, 2013

Build your own Hibernate/Java/NuoDB sample

The purpose of this Wiki page is to help new members of the support team...

- Achieve a basic understanding of how to configure Hibernate with NuoDB
- Identify key components needed to connect Hibernate to NuoDB

Programs you'll need to install:

- Download/Install either Hibernate 3.6.6 or Hibernate 4.1.x
- Download/Install Java JDK 6 (or later)
- Download/Install NuoDB

Prep your environment:

- Run the NuoDB Quickstart script to build the "test" database
- Ensure that your $PATH and $JAVA_HOME paths are set correctly
- Create a script to add the NuoDB JDBC Driver, NuoDB Hibernate Dialect and any required Hibernate jar files to your $CLASSPATH

Linux / Mac:

NuoDB Hibernate Dialect Jar:	/opt/nuodb/jar/nuodb-hibernate-1.0.jar
NuoDB JDBC Driver Jar:	/opt/nuodb/jar/nuodbjdbc.jar

Windows:

NuoDB Hibernate Dialect Jar:	C:\Program Files\NuoDB\jar\nuodb-hibernate-1.0.jar
NuoDB JDBC Driver Jar:	C:\Program Files\NuoDB\jar\nuodbjdbc.jar

Ok, now we're ready to get started…

Use the NuoSQL client to create a "Hockey Player" table:

```
SQL> USE USER;
SQL> CREATE TABLE hockey_player (id INT, firstName STRING, lastName STRING);
```

First, we'll write a "Hockey_player.java" class to define our Hockey Player object:

```
public class Hockey_player {
private int id;
private String firstName;
private String lastName;
    public Hockey_player(){};
    public int getId(){
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getFirstName() {
        return firstName;
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
    public String getLastName() {
        return lastName;
    }
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }
} //class
```

Then we'll write a "Create_HockeyPlayer.java" class to create the Hibernate SessionFactory, open Session and insert a Hockey Player object into our NuoSQL table:

Note: The following example was written for the Hibernat 4.1.12 version

```java
import org.hibernate.cfg.Configuration;
import org.hibernate.HibernateException;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.service.ServiceRegistry;
import org.hibernate.service.ServiceRegistryBuilder;

public class Create_HockeyPlayer{
    private static SessionFactory sessionFactory;
    private static ServiceRegistry serviceRegistry;

//Pull the related configuration information
    static {
        Configuration configuration = new Configuration();
        configuration.configure();
        serviceRegistry =
        new ServiceRegistryBuilder().applySettings(configuration.getProperties()).buildServiceRegistry();

//Create session Factory Object
        try {
            sessionFactory = configuration.buildSessionFactory(serviceRegistry);
        }       catch (HibernateException he){
                    System.err.println("Creation of SessionFactory has failed: " + he);
                    throw new ExceptionInInitializerError(he);
                }
    }

//Return the SessionFactory
    public static SessionFactory getSessionFactory(){
        return sessionFactory;
    }

    public static void main(String[] args){

    //Use the SessionFactory to build new session (specific to Hibernate 4)
        SessionFactory sessionFactory = Create_HockeyPlayer.getSessionFactory();
        Session session = sessionFactory.openSession();
        session.beginTransaction();

 // Call the set methods in my Hockey_Player class to create new player
        Hockey_player player = new Hockey_player();
        player.setId(123);
        player.setFirstName("Betty");
        player.setLastName("White");

    // Save the player's information, commit transaction and close session
        session.save(player);
        session.getTransaction().commit();
        session.close();

    }//main
}//class
```

Next we'll create an “hockey_player.hbm.xml” mapping file to define the data types associated with our “Hockey_player.java” class:

```xml
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD//EN"
"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<hibernate-mapping>
    <class name="Hockey_player" table="HOCKEY_PLAYER">
        <meta attribute="class-description">Class with hockey player info</meta>
        <id name="id" type="int" column="id">
            <generator class="assigned"/>
        </id>
        <property name="firstName" column="firstName" type="string"/>
        <property name="lastName" column="lastName" type="string"/>
    </class>
</hibernate-mapping>
```

Finally, we'll add a “hibernate.cfg.xml” configuration file to provide Hibernate with our NuoDB connection info and the path to our mapping file:

```xml
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-configuration SYSTEM
```

```
"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
    <session-factory>
        <property name="hibernate.dialect">com.nuodb.hibernate.NuoDBDialect</property>
        <property name="hibernate.connection.driver_class">com.nuodb.jdbc.Driver</property>

        <!-- Assume test is the database name -->
        <property name="hibernate.connection.url">jdbc:com.nuodb://localhost/test</property>
        <property name="hibernate.connection.schema">user</property>
        <property name="hibernate.connection.username">dba</property>
        <property name="hibernate.connection.password">goalie</property>
        <property name="hibernate.temp.use_jdbc_metadata_defaults">false</property>

        <!-- List of XML mapping files -->
        <mapping resource="hockey_player.hbm.xml"/>

    </session-factory>
</hibernate-configuration>
```

Now let's see if it works!

1. Create a new directory and place your Java and XML files within this directory
2. Add the path of your "hibernate.cfg.xml" file to your $CLASSPATH script
3. Then cd into your newly created "Hibernate" directory via your Terminal and run the following commands:

```
# Run the classpath script…
$ source classpath_script.sh

# Compile your java classes
$ javac Hockey_player.java
$ javac Create_HockeyPlayer.java

# Run your java program to create a Hockey Player object and add it to your table
$ java Create_HockeyPlayer
```

4. Log back into NuoSQL and run a SELECT query against your table to see if our hockey player ("Betty White") has been entered

```
SQL > USE USER;
SQL > SELECT * FROM hockey_player;

 ID  FIRSTNAME  LASTNAME
 --- ---------- ---------
 123   Betty     White
```

Like   Be the first to like this   Labels None

1 Child Page

NuoDB Hibernate Dialect