UsingHibernateToMakeYourJavaBeansPersistent

Hibernate and Cocoon

This is a "howto" about using Hibernate in Cocoon. It uses XSP to create a Java Bean, populate its fields, and write it to disk using Hibernate.

Hibernate is a great framework if you want to make your Java beans persistent. But the documentation that comes with Hibernate is sparse and intimidating. So Glen Smith wrote a Hibernate kickstart:

I am using his example and I am trying to provide a) the complete source code, b) port it to Hibernate 2.0, and c) make it work with Cocoon.

This example is as simple as possible (no style sheets, only one Hibernate function, etc.). Once you have this up and running, go to the tutorial mentioned above and the Hibernate reference documentation.

I had some problems with versions of Xalan etc. ("jar hell"), so I am trying to provide details on this topic.

At the moment I am working on using Hibernate in combination with XMLForms. If someone is interested (or already did the job), let me know...

The example below uses a MySQL database server. Porting to another database should be straightforward once you have the Cocoon SQL samples up and running for your favourite database 🙂.

Apart from the Hibernate distribution, all files mentioned below are in the zip file attached to this page.

Versions:BR

```
Cocoon: 2.1 dev (CVS HEAD april 5 2003)\\
Hibernate: 2.0 beta 4\\
Tomcat: 4.18\\
```

1. Make sure Cocoon is running OK. I needed to copy the following files from the Cocoon build to "TOMCAT_HOME/common/endorsed"
   `xalan-2.4.1.jar`
   `xercesImpl-2.1.0.jar`
   `xml-apis.jar`
2. Download Hibernate 2.0 beta4 from Sourceforge and extract. Please note that the tutorial by Glen Smith mentioned above deals with version 1.2.x
3. Create the "users" table in your "test" database. See "mysql.sql" for MySQL.
4. Compile the User.java bean. See "User.java". Some classes you imported in step 2) are needed. So make sure they are in your classpath. Put the output class into "TOMCAT_HOME/webapps/cocoon/WEB-INF/classes/test"
5. Create a file "User.hbm.xml" in "TOMCAT_HOME/webapps/cocoon/WEB-INF/classes/test". This file contains the mapping between bean and database columns
6. Create a file "hibernate.properties" in "TOMCAT_HOME/webapps/cocoon/WEB-INF/classes". This file contains info about your DB and credentials (jdbc driver, username and password).
7. Copy the following files from the hibernate 2.0 distribution to "TOMCAT_HOME/webapps/cocoon/WEB-INF/lib":

   ```
    hibernate2.jar\\odmg.jar\\j2ee.jar\\jcs.jar\\jdom.jar\\junit.jar\\connector.jar\\commons-pool.
   jar\\commons-dbcp.jar\\ commons-beanutils.jar\\dom4j.jar\\cglib.jar\\c3p0.jar
   ```

   This is the most tricky part. I did not copy the Ant libs (ant.jar and optional.jar) because Ant is already in my classpath. I did not copy the xerces, xalan and xml-apis.jar because the Cocoon versions are more recent and are already in the endorsed lib. The commons- libs in Cocoon seem to be in a state of flux. You need to copy the ones that are not provided by Cocoon.
8. Copy test.xml and sitemap.xmap into "TOMCAT_HOME/webapps/cocoon/userhomes"
9. Try to open the url http://localhost:8080/cocoon/userhomes/test. If all goes well, a record is added to your database. If you press the refresh button in your browser you will get an error because you are trying to insert a new record with the same primary key. (I told you this is a minimal example).

Other Resources

- WoodyHibernateAndFlow
- CformsHibernateAndFlow

Cocoon-Hibernate.zip