What is Hibernate?

This tutorial explains what Hibernate is and will show some examples why it is useful to use Hibernate.

Generals

Author:
Sebastian Hennebrueder
http://www.laliluna.de/tutorials.html Tutorials for Struts, EJB, xdoclet, JSF, JSP and eclipse.
Date:
February, 25th 2005

PDF Version des Tutorials:

http://www.laliluna.de/assets/tutorials/what-is-hibernate-tutorial-en.pdf

Main features

Hibernate is a solution for object relational mapping and a persistence management solution or persistent layer. This is probably not understandable for anybody learning Hibernate.

What you can imagine is probably that you have your application with some functions (business logic) and you want to save data in a database. When you use Java all the business logic normally works with objects of different class types. Your database tables are not at all objects.

Hibernate provides a solution to map database tables to a class. It copies the database data to a class. In the other direction it supports to save objects to the database. In this process the object is transformed to one or more tables.

Saving data to a storage is called persistence. And the copying of tables to objects and vice versa is called object relational mapping.

Why using Object Relational Mapping?

Better system architecture

When you include all functionality of your application and the access to the database within your dialogs, you will have some severe disadvantages.

It is really difficult to reuse code. You will repeat code at many places. If you change anything it is quite hard to find out all places where you have to add changes.

When you separate your dialogs from your logic and the logic from the persistence mechanism you can more easily apply changes to one part without influencing the other parts.

Reduce time for standard DB actions

Most queries in database development are simple “insert, update, delete” statements. There is no need to develop all these tedious statements. Hibernate helps you save time.

Loading classes from the database looks like

```
Query query = session.createQuery("select b from Bug as b");
      for (Iterator iter = query.iterate(); iter.hasNext();) {
        bugs.add((Bug) iter.next());
      }
    return bugs;
```

saving a class “bug” to the database looks like

```
session.update(bug);
```

Advanced features difficult to develop yourself

Caching solutions, transactions and other features Hibernate provides are not so easy to implement. It is actually non sense to develop something which allready exist.
Using Hibernate everything is there. You just have to use it.

How Hibernate works

What is nice and in my opinion an advantage over entity beans, hibernate starts from simple java classes (Pojo = Plain Old Java Objects).

To use hibernate you will create a simple java class:

```
public class Bug
    implements Serializable
{
    private int hashValue = 0;
    private java.lang.Integer id;
    private java.lang.String title;

    public Bug()
    {
    }
    public Bug(java.lang.Integer fid)
    {
        this.setId(fid);
    }
    public java.lang.Integer getId() {
      return id;
    }
    public void setId(java.lang.Integer id) {
      this.id = id;
    }
    public java.lang.String getTitle() {
      return title;
    }
    public void setTitle(java.lang.String title) {
      this.title = title;
    }
.............
```

Than you create a mapping file. The mapping file explains Hibernate which field in the class is mapped to which field in the database.

```
<hibernate-mapping package="de.laliluna.fehlerbehebung">
    <class name="Bug" table="bug">
        <id name="id" column="fid" type="java.lang.Integer">
            <generator class="sequence">
              <param name="sequence">public.bug_fid_seq</param>
            </generator>
        </id>
        <property name="title" column="ftitle" type="java.lang.String" />
```

```
    </class>
</hibernate-mapping>
```

And you start using your hibernate class.

Example to create a new entry in the database:

```
    try {
      Bug bug = new Bug();
      bug.setTitle("Title");
      bug.setTuserFk(tuser);
          Transaction tx = session.beginTransaction();
      session.save(bug);
      tx.commit();
    } catch (HibernateException e) {
      e.printStackTrace();
    }
```

The creation of the description file can be done with tools like MyEclipse. MyEclipse provides functionality to create classes and description files directly from database tables.

Hibernate includes tools to

- generate Java classes from description files
- description files from existing database tables
- database tables from description files.

Development speed is very high using hibernate.

That' s all.