

# HIBERNATE - MAPPING FILES

[http://www.tutorialspoint.com/hibernate/hibernate\\_mapping\\_files.htm](http://www.tutorialspoint.com/hibernate/hibernate_mapping_files.htm)

Copyright © tutorialspoint.com

An Object/relational mappings are usually defined in an XML document. This mapping file instructs Hibernate how to map the defined class or classes to the database tables.

Though many Hibernate users choose to write the XML by hand, a number of tools exist to generate the mapping document. These include **XDoclet**, **Middlegen** and **AndroMDA** for advanced Hibernate users.

Let us consider our previously defined POJO class whose objects will persist in the table defined in next section.

```
public class Employee {
    private int id;
    private String firstName;
    private String lastName;
    private int salary;

    public Employee() {}
    public Employee(String fname, String lname, int salary) {
        this.firstName = fname;
        this.lastName = lname;
        this.salary = salary;
    }
    public int getId() {
        return id;
    }
    public void setId( int id ) {
        this.id = id;
    }
    public String getFirstName() {
        return firstName;
    }
    public void setFirstName( String first_name ) {
        this.firstName = first_name;
    }
    public String getLastName() {
        return lastName;
    }
    public void setLastName( String last_name ) {
        this.lastName = last_name;
    }
    public int getSalary() {
        return salary;
    }
    public void setSalary( int salary ) {
        this.salary = salary;
    }
}
```

There would be one table corresponding to each object you are willing to provide persistence. Consider above objects need to be stored and retrieved into the following RDBMS table:

```
create table EMPLOYEE (
    id INT NOT NULL auto_increment,
    first_name VARCHAR(20) default NULL,
    last_name VARCHAR(20) default NULL,
    salary INT default NULL,
    PRIMARY KEY (id)
);
```

Based on the two above entities we can define following mapping file which instructs Hibernate how to map the defined class or classes to the database tables.

```

<?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="Employee" table="EMPLOYEE">
    <meta attribute="class-description">
      This class contains the employee detail.
    </meta>
    <id name="id" type="int" column="id">
      <generator />
    </id>
    <property name="firstName" column="first_name" type="string"/>
    <property name="lastName" column="last_name" type="string"/>
    <property name="salary" column="salary" type="int"/>
  </class>
</hibernate-mapping>

```

You should save the mapping document in a file with the format <classname>.hbm.xml. We saved our mapping document in the file Employee.hbm.xml. Let us see little detail about the mapping elements used in the mapping file:

- The mapping document is an XML document having **<hibernate-mapping>** as the root element which contains all the <class> elements.
- The **<class>** elements are used to define specific mappings from a Java classes to the database tables. The Java class name is specified using the **name** attribute of the class element and the database table name is specified using the **table** attribute.
- The **<meta>** element is optional element and can be used to create the class description.
- The **<id>** element maps the unique ID attribute in class to the primary key of the database table. The **name** attribute of the id element refers to the property in the class and the **column** attribute refers to the column in the database table. The **type** attribute holds the hibernate mapping type, this mapping types will convert from Java to SQL data type.
- The **<generator>** element within the id element is used to automatically generate the primary key values. Set the **class** attribute of the generator element is set to **native** to let hibernate pick up either **identity**, **sequence** or **hilo** algorithm to create primary key depending upon the capabilities of the underlying database.
- The **<property>** element is used to map a Java class property to a column in the database table. The **name** attribute of the element refers to the property in the class and the **column** attribute refers to the column in the database table. The **type** attribute holds the hibernate mapping type, this mapping types will convert from Java to SQL data type.

There are other attributes and elements available which will be used in a mapping document and I would try to cover as many as possible while discussing other Hibernate related topics.