

ORM Metadata



Hibernate XML Tabanlı Metadata

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD
3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
  <class name="com.javaegitimleri.petclinic.model.Pet" table="PETS">
    <id name="id" type="long">
      <column name="ID" />
      <generator class="assigned" />
    </id>
    <property name="name" type="string">
      <column name="NAME" />
    </property>
  </class>
</hibernate-mapping>
```

Hibernate'e özel XML metadata tanımıdır (hbm.xml)

Burada bir sınıfın veritabanındaki bir tabloya mapping'i gösterilmektedir

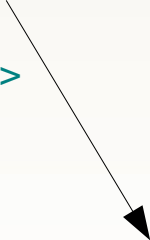
Hibernate XML Tabanlı Metadata

- XML metadata dosyaları **hbm.xml** uzantısı ile oluşturulur
- Tek bir **hbm.xml** dosyasında **birden fazla entity** için de tanım yapmak mümkündür
- Ancak genellikle **her entity sınıf için ayrı ayrı** yazılır
- **hbm.xml** dosyaların path'leri **hibernate.cfg.xml** içerisinde tanımlanmalıdır

Hibernate XML Tabanlı Metadata

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

<hibernate-configuration>
  <session-factory>
    <mapping resource="com/javaegitimleri/petclinic/model/Person.hbm.xml" />
    <mapping resource="com/javaegitimleri/petclinic/model/Vet.hbm.xml" />
    <mapping resource="com/javaegitimleri/petclinic/model/Owner.hbm.xml" />
    <mapping resource="com/javaegitimleri/petclinic/model/Pet.hbm.xml" />
    <mapping resource="com/javaegitimleri/petclinic/model/PetType.hbm.xml" />
    <mapping resource="com/javaegitimleri/petclinic/model/Visit.hbm.xml" />
    <mapping resource="com/javaegitimleri/petclinic/model/Specialty.hbm.xml" />
  </session-factory>
</hibernate-configuration>
```



hbm.xml dosyaları projenin classpath'inde aranır

JPA ve XML Metadata

- JPA anotasyonları yerine **XML mapping**'ler kullanılabilir
- **META-INF/orm.xml** dosyası, JPA tarafından otomatik tespit edilir ve yüklenir
- JPA anotasyonlarının her birisi için bir XML karşılığı vardır
- JPA anotasyonları XML ile **override** edilebilir

JPA ve XML Metadata

- **<xml-mapping-metadata-complete>** elemanı mevcut ise sınıflardaki anotasyonlar tamamen göz ardı edilir
- Bu özellik sınıf düzeyinde **metadata-complete="true"** ile de sağlanabilir
- Hibernate ve JPA XML metadata'ları birbirleri ile **uyumsuzdur**, birlikte kullanılamazlar

JPA ve XML Metadata Örneği

```
<entity-mappings>
  <persistence-unit-metadata>
    <xml-mapping-metadata-complete/>
  </persistence-unit-metadata>
  <entity class="com.javaegitimleri.petclinic.model.Pet"
access="FIELD" metadata-complete="true">
    <attributes>
      <id name="id">
        <generated-value strategy="AUTO"/>
      </id>
      <basic name="name">
        <column name="PET_NAME"/>
      </basic>
      <one-to-many name="visits" target-
entity="com.javaegitimleri.petclinic.model.Visit"/>
    </attributes>
  </entity>
</entity-mappings>
```

Farklı Metadata Formatlarını Bir Arada Kullanmak

- Farklı sınıflar **farklı metadata formatlarına** sahip olabilirler
- Ancak **Hibernate aynı sınıf için** bu formatlardan **sadece birini** kullanmaya izin verir
- Hibernate **XML tanımları, anotasyon tanımlarını override edemez**
- Her sınıf için **iki yöntemden birisi** tercih edilmelidir

Farklı Metadata Formatlarını Bir Arada Kullanmak

- Hibernate için **hbm.xml** dosyaları, anotasyonlardan **önceliklidir**
- İstenirse **hibernate.mapping.precedence** property tanımı ile bu sıralama **class,hbm** şeklinde değiştirilebilir
- **JPA**'da ise XML ve anotasyonlar **sınıf düzeyinde de birlikte** kullanılabilir, sınıf düzeyinde anotasyon tanımları XML tanımları ile override edilebilir

Global Metadata

- **Birden fazla sınıfa etki edecek, uygulama genelinde kullanılacak metadata tanımlarına** da ihtiyaç duyulmaktadır
- Örneğin,
 - İsimlendirilmiş sorgular (**named query**)
 - Uygulamaya özel eşleştirme tipleri (**UDT**)
 - veri filtre tanımları (**filter**) global metadata'dır

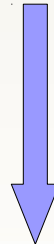
Global Metadata

- Global metadata, sınıflarda da tanımlanabilir
- Ancak tek bir yerde toplanmaları daha iyi bir pratiktir
- Bu toplama işlemi herhangi bir paket altına konan **package-info.java** ile yapılabilir

package-info.java

```
@TypeDefs(value={
    @TypeDef(name="money-simple",typeClass=MoneyUserType.class),
    @TypeDef(name="money-composite",typeClass=MoneyCompositeUserType.class),
    @TypeDef(name="money-
parameterized",typeClass=MoneyParameterizedUserType.class,parameters={@Par
ameter(name="dbCurrencyCode",value="TRL")}))
package com.javaegitimleri.petclinic.model;

import org.hibernate.annotations.Parameter;
import org.hibernate.annotations.TypeDef;
import org.hibernate.annotations.TypeDefs;
```



Global metadata'nın aktivasyonu için hibernate.cfg.xml'e package-info.java dosyasının bulunduğu paket de belirtilmelidir

```
<hibernate-configuration>
    <session-factory>
        ...
        <mapping package="com.javaegitimleri.petclinic.model"/>
    </session-factory>
</hibernate-configuration>
```

package-info.java ve JPA

- **JPA spesifik anotasyonlar package-info.java içerisinde kullanılamaz**
- Dolayısı ile isimlendirilmiş sorguları package-info.java içerisinde tanımlamak için **Hibernate anotasyonlarına** ihtiyaç duyulur

İletişim

- Harezmi Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- <http://www.java-egitimleri.com>
- info@java-egitimleri.com

