

EJB ve Hibernate

Burak Bayramlı

Bu belge, \LaTeX ile üretilmiştir

PersistenceContext Enjekte Etmek

```
@Stateful
public class XYZBean implements XYZ {

    @PersistenceContext
    EntityManager entityManager;
    ...
}
```

persistence.xml

```
<persistence>
  <persistence-unit name="kitapDemoDatabase">
    <provider>org.hibernate.ejb.HibernatePersistence</provider>
    <jta-data-source>java:[DATA SOURCE ISMI]</jta-data-source>
    <properties>
      <property name="hibernate.hbm2ddl.auto" value="create-drop"/>
      <property name="hibernate.show_sql" value="true"/>
      <property name="hibernate.cache.provider_class"
        value="org.hibernate.cache.HashtableCacheProvider"/>
      <property name="hibernate.transaction.manager_lookup_class"
        value="org.hibernate.transaction.JBossTransactionManagerLookup"/>
      <property name="dialect">org.hibernate.dialect.MySQLDialect</property>
    </properties>
  </persistence-unit>
</persistence>
```

kitapdemo-ds.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<datasources>
  <local-tx-datasource>
    <jndi-name>[__DATA_SOURCE_ISMI__]</jndi-name>
    <connection-url>jdbc:mysql://localhost:3306/[___DB_ISMI___]</connection-url>
    <driver-class>org.gjt.mm.mysql.Driver</driver-class>
    <user-name>___USER___</user-name>
    <password>___PASSWORD___</password>
    <exception-sorter-class-name>
      org.jboss.resource.adapter.jdbc.vendor.MySQLExceptionSorter
    </exception-sorter-class-name>
    <metadata>
      <type-mapping>mySQL</type-mapping>
    </metadata>
  </local-tx-datasource>

```

</datasources>

Hibernate ve Transaction

- `persistence.xml` üzerinde yaptığımız hibernate. → `transaction.manager.lookup_class` sayesinde JBoss Hibernate üzerinden yaptığımız tüm veri erişimleri, o anda işlemekte olan transaction'a dahil olacaktır.
- Hibernate ise bu transaction'ı alıp JDBC'ye aktarabilecektir.
- JDBC zaten veri tabanı için bir önyüzdür. Böylece DB

transaction'ı da bizim kontrolümüzde olacak.

Session Bean ve Transaction

```
@Stateful
public class ServiceXYZ implements XYZ, Serializable
{
    @TransactionAttribute
    (TransactionAttributeType.REQUIRED)
    public void metot1() {
        ...
    }
    ...
}
```

Görevler

- JBoss'u kurmadıysak kuralım.
- CounterStateful projesini yeni bir isime kopyalayalım, mesela EjbHibernate. Bu projede resources altında kitapdemo-ds.xml, resources/META-INF altında bir persistence.xml dosyaları ekleyelim. Derleme script'i bu dosyaları gerekli yerlere koyacak şekilde kodlandı zaten.

- Şimdi daha önce geliştirdiğimiz Car ve Garage POJO'larını bu yeni projeye ekleyelim.
- CarService adında bir EJB yazalım. Bu EJB'de getCar ve addCar adında iki metot yazalım, bu metotlar parametre olarak Car POJO'sunu alıp geri döndürecekler ve isimlerinin belirttiği işi yapacaklar.
- Bir mainline programdan bu metotları test edelim.

Son