JPA Konfigürasyonu

harezmi bilişim çözümleri

JPA Konfigürasyonu

- JPA konfigürasyonu **spesifikasyona göre iki farklı biçim**de gerçekleştirilebilir
  - Java EE Mode
  - Java SE Mode
- Java EE mode'da JPA provider (Hibernate) **container tarafından** başlatılmaktadır
- **Java EE modu**nda EntityManager, Transaction oluşturma ve yönetme genellikle **container tarafından** gerçekleştirilir

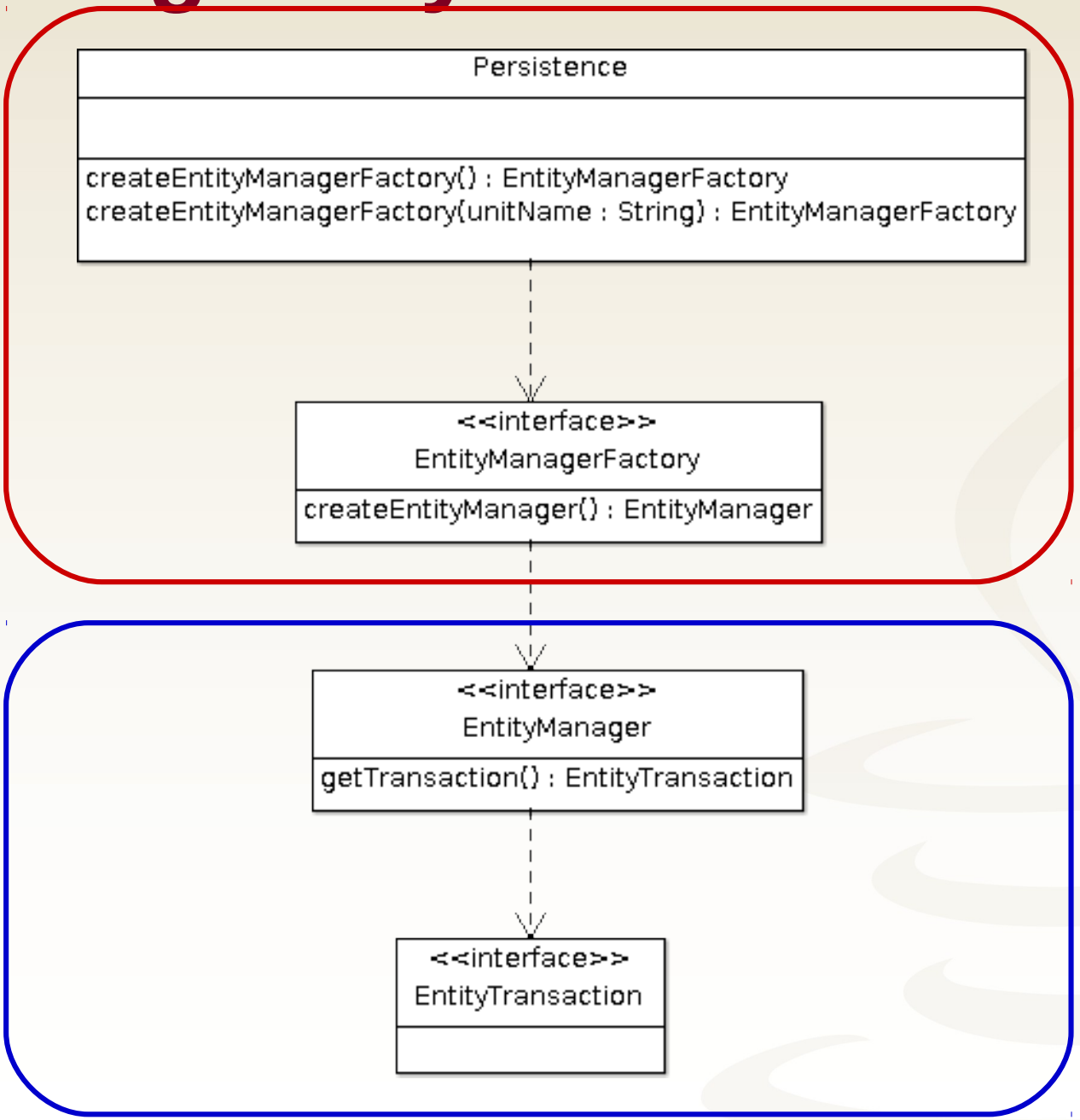
JPA Konfigürasyonu

- Container, **EJB Container veya Spring Container** olabilir, mutlaka Java EE sunucu kullanmak **şart değil**dir
- Java SE mode'da ise JPA provider **uygulama içerisinden** başlatılmaktadır
- **Java SE modu**nda EntityManager, Transaction oluşturma ve yönetme **uygulama kodu içerisinde, uygulamanın kendisi** tarafından yapılmaktadır

JPA Konfigürasyonu

Uygulama genelinde tek seferlik EMF yaratılır

Persistence işlemleri EM ve TX üzerinden gerçekleştirilir



JPA Konfigürasyon Dosyası

- classpath:/META-INF/**persistence.xml** default **konfigürasyon dosyası**dır
- JPA konfigürasyonları **persistence unit**'ler şeklinde yapılır
- Her bir persistence unit içerisinde JPA provider'ın çalışması ile ilgili **mapping metadata** ve **ayarlar** yer alır
- Her bir "**persistence unit**" ayrı bir isme sahiptir
- Bootstrap sırasında bu isimle **EntityManagerFactory** nesnesi elde edilir

JPA Konfigürasyon Örneği

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.1" ...>
   <persistence-unit name="testjpa"
           transaction-type="RESOURCE_LOCAL">

      <jar-file>packagedEntities.jar</jar-file>
      <class>com.example.Person</class>

      <properties>
         <property name="javax.persistence.jdbc.driver"
                   value="org.h2.Driver" />
         <property name="javax.persistence.jdbc.url"
                   value="jdbc:h2:tcp://localhost/~/test" />
         <property name="javax.persistence.jdbc.user" value="sa" />
         <property name="javax.persistence.jdbc.password" value="" />

         <property name="hibernate.dialect"
                   value="org.hibernate.dialect.H2Dialect" />
         <property name="hibernate.hmb2ddl.auto" value="create" />

         <property name="hibernate.ejb.cfgfile" value="/hibernate.cfg.xml" />
      </properties>

   </persistence-unit>
</persistence>
```

Belirtilmez ise JavaEE modu için default JTA, JavaSE modu için ise default RESOURCE_LOCAL'dir

hibernate.ejb.cfgfile property tanımı ile hibernate.cfg.xml deki tanımlar doğrudan da yüklenebilir

JPA Konfigürasyon Dosyası

- JPA spesifikasyonunda **mapping metadata**'nın **otomatik tespiti** sadece Java EE ortamında şart koşulmuştur
- Java SE ortamında mapping metadata teker teker **<class>** veya **<jar-file>** elemanları ile belirtilmelidir
- Hibernate, JPA provider olarak her iki ortamda da annotated sınıfları **otomatik tespit** eder
- **archive.autodetection** property ile metadata dosyalarının yüklenme sırası değiştirilebilir
  - Default sıra: class, hbm

JPA Konfigürasyon Dosyası

```
<persistence-unit ...>
        <provider>
            org.hibernate.jpa.HibernatePersistenceProvider
        </provider>
</persistence-unit>
```

- Uygulama sunucularının classpath'inde aynı anda **birden fazla JPA vendor** yer alabilir
- persistence.xml'de **<provider>** tanımı sadece **classpath'de birden fazla JPA vendor**'un söz konusu olduğu durumlarda gereklidir

EntityManagerFactory'nin Yaratılması

Uygulama içerisinde Persistence sınıfından EMF yaratılması genellikle JavaSE modunda olur. JavaEE modunda ise EMF container tarafından yaratılır

```
EntityManagerFactory entityManagerFactory =
   Persistence.createEntityManagerFactory("testjpa");
```

İletişim

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