JPA with Hibernate 3.0 Transactions



# **Lesson Objectives**

After completing this lesson, participants will be able to understand:

- What is Transaction
- Types of Transactions
- Working with transactions



#### 7.1 Transaction



## What is Transaction?

A transaction is a set of operations that either fail or succeed as a unit.

Transactions are a fundamental part of persistence.

 A database transaction consists of a set of <u>DML</u> (Data Manipulation Language) operations that are committed or rolled back as a single unit.

 An object level transaction is one in which a set of changes made to a set of objects are committed to the database as a single unit.

#### 7.1 Transaction

## What is Transaction?

- •JPA transactions can be managed by:
- the users application
- a framework (such as Spring)
- a J2EE container

#### 7.2: Types of transactions



## Types of Transactions

Transactions can be controller in two ways in JPA

- Java Transaction API (JTA)
  - container-managed entity manager
- EntityTransaction API (tx.begin(), tx.commit(), etc)
  - application-managed entity manager

#### **DEFAULT TYPES:**

Default to JTA in a JavaEE environment and to RESOURCE\_LOCAL in a JavaSE environment.

#### 7.2: Types of transactions



# Types of Transactions

In the **persistence.xml** JPA configuration file, you can have a line like:

```
<persistence-unit name="test-PU" transaction-type="JTA">
(or)
```

<persistence-unit name="test-PU" transaction-type="RESOURCE\_LOCAL">

#### 7.3: Working with transactions



## Application Managed Entity Manager

Applications create EntityManager instances by using directly **Persistence** and **EntityManagerFactory**.

## javax.persistence.Persistence

- Root class for obtaining an EntityManager
- Locates provider service for a named persistence unit
- Invokes on the provider to obtain an EntityManagerFactory

## javax.persistence.EntityManagerFactory

Creates EntityManagers for a named persistence unit or configuration

#### 7.3: Working with transactions



## **Application Managed Entity Manager**

```
public class PersistenceProgram {
 public static void main(String[] args)
  EntityManagerFactory emf =
       Persistence.createEntityManagerFactory("SomePUnit");
  EntityManager em = emf.createEntityManager();
  em.getTransaction().begin();
  // Perform finds, execute queries,
  // update entities, etc.
  em.getTransaction().commit();
  em.close();
  emf.close();
```



## Container Managed Entity Manager

•An EntityManager with a transactional persistence context can be injected by using the **@PersistenceContext** annotation.

```
public class BookmarkSeviceImpl implements BookmarkService {
    @PersistenceContext
    private EntityManager em;

public void save(Bookmark bookmark) {
    if (bookmark.getId() == null) {
        em.persist(bookmark);
    } else {
        em.merge(bookmark);
    }
}
```

# Summary



In this lesson, you have learned about:

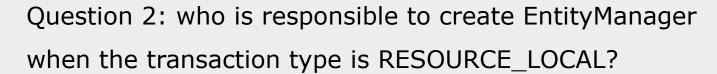
- What is Transaction
- Types of Transactions
- Working with transactions



# **Review Question**

Question 1: what is the default transaction type for J2EE environment?

- JTA
- RESOURCE\_LOCAL



- User
- Container

