A Gentle Introduction to the JPA

Java User Group
Mark de Reeper
Sun Microsystems

Agenda

- Background
- What is Object-relational Mapping?
- Primary Features
- The Basics
- Advanced Topics
- Still only 1.0
- Futures
- References

Background

- Java Persistence API introduced in JSR-220 (Enterprise JavaBeans™ 3.0)
- Unifies existing object-relational mapping technologies into a standard enterprise API
- Works with either Java EE 5 or Java SE 5
  - Superior ease of use within a Java EE 5 container
  - Client API with local transactions in Java SE platform
- Leverages Java SE 5 Annotations
  - Can still be (XML) mapping file driven
- Service Provider Interface (SPI) for container/persistence provider plugability

What is Object-relational Mapping?

- From http://en.wikipedia.org/wiki/Object-relational_mapping

  Object-Relational mapping (aka O/RM, ORM, and O/R mapping) is a programming technique for converting data between incompatible type systems in databases and Object-oriented programming languages. This creates, in effect, a "virtual object database" which can be used from within the programming language.

  There are both free and commercial packages available that perform object-relational mapping, although some programmers opt to create their own ORM tools.

Primary Features

- POJO-based persistence model
  - Plain Old Java Objects
- Supports traditional O-O modelling concepts
  - Inheritance, polymorphism, encapsulation, etc.
- Standard abstract relational query language
  - Java Persistence Query Language
- Standard O/R mapping metadata
  - Using Java SE 5 Annotations and/or XML
- Portability across providers (implementations)
  - Toplink, Hibernate, Kodo, OpenJPA, EclipseLink

The Basics – Getting Started

- Minimum of 2 files
  - Entity definition (POJO)
  - Persistence Unit Definition (XML)
- JPA Library(s)
- Database
- Example Runner (Main)

The Basics – Entity Definition

- Tag the ID (Primary Key) field
- Defaults cover Table and Column names
- Optionally implement Serializable

```
@Entity
public class ExampleEntity implements Serializable {

    @Id
    private long id;
    private int fieldInt;
    private long fieldLong;
    private String fieldString;
.....
```

The Basics – Primary Keys

- The primary key, or the property or field of a composite primary key, must be one of the following Java language types:
  - Java primitive types
  - Java primitive wrapper types
  - java.lang.String
  - java.util.Date (the temporal type should be DATE)
  - java.sql.Date
  - Primary Key Classes
- Annotations

```
@Id
@GeneratedValue(strategy=GenerationType.AUTO)
```

The Basics – Relationships

- One-to-one
  - Each entity instance is related to a single instance of another entity.
- One-to-many
  - An entity instance can be related to multiple instances of the other entities.
- Many-to-one
  - Multiple instances of an entity can be related to a single instance of the other entity.
- Many-to-many
  - The entity instances can be related to multiple instances of each other.

The Basics – Query Language

- Very SQL like, derived from EJBQL
- Named or Positional Parameters
  - :named or ?1
- Functions and Expressions
  - DISTINCT, IS NULL, IS EMPTY, BETWEEN, LIKE, IN, MAX, MIN, SUM, COUNT, LENGTH, TRIM, ABS, SQRT, CONCAT
- GROUP and ORDER BY
- Bulk Updates and Deletes
- Simple Example
  - SELECT v FROM Visitor v WHERE v.attending = 'JUG'

The Basics – EntityManager

- javax.persistence.EntityManager
  - Key interface into the Persistence Unit
    - Think UnitOfWork in Toplink or Session in Hibernate
- Provides CRUD interface
  - persist - Create
  - find – Retrieve
  - merge - Update
  - remove – Delete
- Provides access to Query engine
  - createQuery
  - createNameQuery

The Basics – Persistence Unit

- META-INF/persistence.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<persistence version="1.0" xmlns="http://java.sun.com/xml/ns/persistence"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
   http://java.sun.com/xml/ns/persistence/persistence_1_0.xsd">

<persistence-unit name="examplePU toplink" transaction-type="RESOURCE_LOCAL">
   <provider>oracle.toplink.essentials.PersistenceProvider</provider>
   <class>gentlejpabasic.ExampleEntity</class>
   <class>gentlejpabasic.ExampleRelationshipEntity</class>
       <properties>
         <property name="toplink.jdbc.user" value="root"/>
         <property name="toplink.jdbc.password" value="*****"/>
         <property name="toplink.jdbc.url" value="jdbc:mysql://localhost/examplejpa"/>
         <property name="toplink.jdbc.driver" value="com.mysql.jdbc.Driver"/>
         <property name="toplink.ddl-generation" value="create-tables"/>
         <property name="toplink.logging.level" value="OFF"/>
       </properties>

  </persistence-unit>

</persistence>
```

The Basics – Putting it all Together

- Time to look at some code!

Advanced Topics

- Transactions
  - Java SE World

```
entityManager.getTransation().begin();
entityManager.getTransaction().commit();
```

  - Java EE World
    - Container Managed
- Injection

```
   @PersistenceContext
   private EntityManager em;
or
   @PersistenceUnit
   private EntityManagerFactory emf;

   @Resource
   private UserTransaction utx;
```

Advanced Topics - cont

- Optimistic Locking

```
@Version
private long versionId;
```

- Mapped Superclasses

```
@MappedSuperclass
public abstract class BaseEntity {
    @Id
    private long id;
    @Version
    private long version;
....
```

- Spring Integration
  - > JpaDaoSupport
  - > JpaTemplate

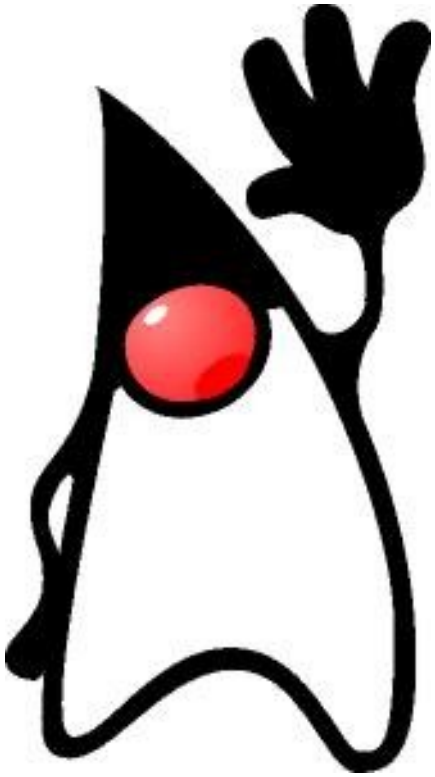
Futures

- Some possible new features of JPA 2.0:
  - Better validation support, e.g. @Length(5), @Max(2)
  - Improved JPQL support
  - Criteria API support
    - Toplink and Hiberate already have this.
  - The HashMap like hints (e.g. logging, caching) to be standardised
    - toplink.logging.level
    - toplink.cache.type.default

References

- Java EE Tutorial
  - http://java.sun.com/javaee/5/docs/tutorial/doc/
- Java EE Platform Introduction
  - http://java.sun.com/javaee/5/docs/firstcup/doc/toc.html
- The SpringFramework
  - http://springframework.org/

Questions?



A Gentle Introduction to the JPA

Mark de Reeper
mark.dereeper@sun.com