JPA Eclipselink

Example jpa-eclipselink can be browsed at https://github.com/apache/tomee/tree/master/examples/jpa-eclipselink

This example shows how to configure `persistence.xml` to work with Eclipselink. It uses an `@Entity` class and a `@Stateful` bean to add and delete entities from a database.

Creating the JPA Entity

The entity itself is simply a pojo annotated with `@Entity`. We create one pojo called `Movie` which we can use to hold movie records.

```
package org.superbiz.eclipselink;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

@Entity
public class Movie {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private long id;

    private String director;
    private String title;
    private int year;

    public Movie() {
    }

    public Movie(String director, String title, int year) {
        this.director = director;
        this.title = title;
        this.year = year;
    }

    public String getDirector() {
        return director;
    }

    public void setDirector(String director) {
        this.director = director;
    }

    public String getTitle() {
```

```
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public int getYear() {
        return year;
    }

    public void setYear(int year) {
        this.year = year;
    }

}
```

Database Operations

This is the bean responsible for database operations; it allows us to persist or delete entities. For more information we recommend you to see injection-of-entitymanager

```
package org.superbiz.eclipselink;

import javax.ejb.Stateful;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import javax.persistence.PersistenceContextType;
import javax.persistence.Query;
import java.util.List;

@Stateful
public class Movies {

    @PersistenceContext(unitName = "movie-unit", type = PersistenceContextType
.EXTENDED)
    private EntityManager entityManager;

    public void addMovie(Movie movie) throws Exception {
        entityManager.persist(movie);
    }

    public void deleteMovie(Movie movie) throws Exception {
        entityManager.remove(movie);
    }

    public List<Movie> getMovies() throws Exception {
        Query query = entityManager.createQuery("SELECT m from Movie as m");
        return query.getResultList();
    }
}
```

Persistence.xml with EclipseLink configuration

This operation is too easy, just set the `provider` to `org.eclipse.persistence.jpa.PersistenceProvider` and add additional properties to the persistence unit. The example has followed a strategy that allows the creation of tables in a HSQL database. For a complete list of persistence unit properties see here

```
<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="movie-unit">
    <provider>org.eclipse.persistence.jpa.PersistenceProvider</provider>
    <jta-data-source>movieDatabase</jta-data-source>
    <non-jta-data-source>movieDatabaseUnmanaged</non-jta-data-source>
    <properties>
      <property name="eclipselink.target-database" value=
"org.eclipse.persistence.platform.database.HSQLPlatform"/>
      <property name="eclipselink.ddl-generation" value="create-tables"/>
      <property name="eclipselink.ddl-generation.output-mode" value="database"/>
    </properties>
  </persistence-unit>
</persistence>
```

MoviesTest

Testing JPA is quite easy, we can simply use the `EJBContainer` API to create a container in our test case.

```
package org.superbiz.eclipselink;

import junit.framework.TestCase;

import javax.ejb.embeddable.EJBContainer;
import javax.naming.Context;
import java.util.List;
import java.util.Properties;

/**
 * @version $Revision: 607077 $ $Date: 2007-12-27 06:55:23 -0800 (Thu, 27 Dec 2007) $
 */
public class MoviesTest extends TestCase {

    public void test() throws Exception {
        Properties p = new Properties();
        p.put("movieDatabase", "new://Resource?type=DataSource");
        p.put("movieDatabase.JdbcDriver", "org.hsqldb.jdbcDriver");
        p.put("movieDatabase.JdbcUrl", "jdbc:hsqldb:mem:moviedb");

        final Context context = EJBContainer.createEJBContainer(p).getContext();

        Movies movies = (Movies) context.lookup("java:global/jpa-eclipselink/Movies");

        movies.addMovie(new Movie("Quentin Tarantino", "Reservoir Dogs", 1992));
        movies.addMovie(new Movie("Joel Coen", "Fargo", 1996));
        movies.addMovie(new Movie("Joel Coen", "The Big Lebowski", 1998));

        List<Movie> list = movies.getMovies();
        assertEquals("List.size()", 3, list.size());

        for (Movie movie : list) {
            movies.deleteMovie(movie);
        }

        assertEquals("Movies.getMovies()", 0, movies.getMovies().size());
    }
}
```

Running

When we run our test case we should see output similar to the following.

```
-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running org.superbiz.eclipselink.MoviesTest
Apache OpenEJB 4.0.0-beta-1    build: 20111002-04:06
```

```
http://tomee.apache.org/
INFO - openejb.home = /Users/dblevins/examples/jpa-eclipselink
INFO - openejb.base = /Users/dblevins/examples/jpa-eclipselink
INFO - Using 'javax.ejb.embeddable.EJBContainer=true'
INFO - Configuring Service(id=Default Security Service, type=SecurityService,
provider-id=Default Security Service)
INFO - Configuring Service(id=Default Transaction Manager, type=TransactionManager,
provider-id=Default Transaction Manager)
INFO - Configuring Service(id=movieDatabase, type=Resource, provider-id=Default JDBC
Database)
INFO - Found EjbModule in classpath: /Users/dblevins/examples/jpa-
eclipselink/target/classes
INFO - Beginning load: /Users/dblevins/examples/jpa-eclipselink/target/classes
INFO - Configuring enterprise application: /Users/dblevins/examples/jpa-eclipselink
INFO - Configuring Service(id=Default Stateful Container, type=Container, provider-
id=Default Stateful Container)
INFO - Auto-creating a container for bean Movies: Container(type=STATEFUL, id=Default
Stateful Container)
INFO - Configuring Service(id=Default Managed Container, type=Container, provider-
id=Default Managed Container)
INFO - Auto-creating a container for bean org.superbiz.eclipselink.MoviesTest:
Container(type=MANAGED, id=Default Managed Container)
INFO - Configuring PersistenceUnit(name=movie-unit,
provider=org.eclipse.persistence.jpa.PersistenceProvider)
INFO - Auto-creating a Resource with id 'movieDatabaseNonJta' of type 'DataSource for
'movie-unit'.
INFO - Configuring Service(id=movieDatabaseNonJta, type=Resource, provider-
id=movieDatabase)
INFO - Adjusting PersistenceUnit movie-unit <non-jta-data-source> to Resource ID
'movieDatabaseNonJta' from 'movieDatabaseUnmanaged'
INFO - Enterprise application "/Users/dblevins/examples/jpa-eclipselink" loaded.
INFO - Assembling app: /Users/dblevins/examples/jpa-eclipselink
INFO - PersistenceUnit(name=movie-unit,
provider=org.eclipse.persistence.jpa.PersistenceProvider) - provider time 511ms
INFO - Jndi(name="java:global/jpa-eclipselink/Movies!org.superbiz.eclipselink.Movies")
INFO - Jndi(name="java:global/jpa-eclipselink/Movies")
INFO -
Jndi(name="java:global/EjbModule225280863/org.superbiz.eclipselink.MoviesTest!org.supe
rbiz.eclipselink.MoviesTest")
INFO - Jndi(name="java:global/EjbModule225280863/org.superbiz.eclipselink.MoviesTest")
INFO - Created Ejb(deployment-id=Movies, ejb-name=Movies, container=Default Stateful
Container)
INFO - Created Ejb(deployment-id=org.superbiz.eclipselink.MoviesTest, ejb-
name=org.superbiz.eclipselink.MoviesTest, container=Default Managed Container)
INFO - Started Ejb(deployment-id=Movies, ejb-name=Movies, container=Default Stateful
Container)
INFO - Started Ejb(deployment-id=org.superbiz.eclipselink.MoviesTest, ejb-
name=org.superbiz.eclipselink.MoviesTest, container=Default Managed Container)
INFO - Deployed Application(path=/Users/dblevins/examples/jpa-eclipselink)
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 3.188 sec
```

```
Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
```