Spring EJB and JPA

OpenEJB 3.1 or later required

This example shows how to combine Spring, OpenEJB and Hibernate using the integration code provided by OpenEJB. Here, OpenEJB is used as an embeddable EJB container inside of Spring. See the Spring page for details.

We use the basic Movie example and expand it to include more objects to demonstrate both Spring beans, EJB Session beans, and JPA persistent objects in one application. The premise of the example is a Cineplex that has a number of Theaters (viewing screens), each playing a number of Movies. The basic object layout is as follows:

Object	Type	Description
CineplexImpl	@Stateless	Shows the use of @Resource to have Spring beans injected. Specifically, the *Theaters* Spring bean
Theaters	Spring bean	Simple wrapper object injected into *CineplexImpl*
Theater	Spring bean	Shows that EJBs can be injected into Spring beans. Uses both the *Movies* EJB and the *Movie* JPA objects
MoviesImpl	@Stateful	Wraps a JPA EntityManager and provides transactional access to the persistent *Movie* objects
Movie	@Entity	Basic JPA bean that is used both by Spring beans and EJBs. The same *Movie* object as in all the other persistence related examples.
AvailableMovies	Spring bean	Simple object used as a clever way to seed the EntityManager (and really, the database) with persistent *Movie* objects

Required jars

To setup the integration you'll need:

1. The standard OpenEJB 3.1 libraries
2. The openejb-spring-3.1.jar or later
3. Spring 2.5 or other (any version should work)

In Maven2 this can be done by adding the following dependencies to your pom.xml

{snippet:id=required|url=openejb3/examples/spring-integration/pom.xml|lang=xml}

For other environments, you can simply download an openejb-3.1.zip or later and include all the jars under the lib/ directory in your classpath. Then download and add the openejb-spring-3.1.jar along with your Spring jars.

The Spring xml

Bootstrapping and Configuring OpenEJB is fairly simple.

{snippet:id=bootstrapping|url=openejb3/examples/spring-integration/src/main/resources/movies.xml|lang=xml}

As well, you can optionally declare any resources or containers. Anything declarable as a <Resource> or <Container> in the openejb.xml can instead be declared in the Spring xml file as shown here.

{snippet:id=resources|url=openejb3/examples/spring-integration/src/main/resources/movies.xml|lang=xml}

And finally our Spring beans.

{snippet:id=pojos|url=openejb3/examples/spring-integration/src/main/resources/movies.xml|lang=xml} Don't forget {snippet:id=annotations|url=openejb3/examples/spring-integration/src/main/resources/movies.xml|lang=xml}

It allows various annotations to be detected in bean classes: Spring's @Required and @Autowired, as well as JSR 250's @PostConstruct, @PreDestroy and @Resource (if available), JAX-WS's @WebServiceRef (if available), EJB3's @EJB (if available), and JPA's @PersistenceContext and @PersistenceUnit (if available). Alternatively, you may choose to activate the individual BeanPostProcessors for those annotations.

The Code

In efforts to keep the example page somewhat short, we'll show just three beans, each demonstrating a particular relationship.

The first is the CineplexImpl EJB which shows EJB -> Spring.

{snippet:id=code|url=openejb3/examples/spring-integration/src/main/java/org/superbiz/spring/CineplexImpl.java|lang=java}

The second is the Theater Spring bean which shows Spring -> EJB.

{snippet:id=code|url=openejb3/examples/spring-integration/src/main/java/org/superbiz/spring/Theater.java|lang=java}

The last is the AvailableMovies Spring bean which Shows Spring -> EJB -> JPA

{snippet:id=code|url=openejb3/examples/spring-integration/src/main/java/org/superbiz/spring/AvailableMovies.java|lang=java}

The TestCase

The JUnit TestCase uses a ClassPathXmlApplicationContext to load the Spring ApplicationContext. Anything that loads your Spring xml file should work fine. The following code would work a plain java app as well.

{snippet:id=code|url=openejb3/examples/spring-integration/src/test/java/org/superbiz/spring/MoviesTest.java|lang=java}

Running

The source for this example can be downloaded from svn via:

$ svn co http://svn.apache.org/repos/asf/openejb/trunk/openejb3/examples/spring-integration

Then, in the "spring-integration" directory, run:

$ mvn clean install

Which should create output like the following.

--- T E S T S --- Running org.superbiz.spring.MoviesTest log4j:WARN No appenders could be found for logger (org.springframework.context.support.ClassPathXmlApplicationContext). log4j:WARN Please initialize the log4j system properly. Apache OpenEJB 3.1 build: 20081009-03:31 http://openejb.apache.org/ INFO - openejb.home = /Users/dblevins/work/openejb3/examples/spring-integration INFO - openejb.base = /Users/dblevins/work/openejb3/examples/spring-integration INFO - Configuring Service(id=Default JDK 1.3 ProxyFactory, type=ProxyFactory, provider-id=Default JDK 1.3 ProxyFactory) INFO - Configuring Service(id=MovieDatabase, type=Resource, provider-id=Default JDBC Database) INFO - Configuring Service(id=MovieDatabaseUnmanaged, type=Resource, provider-id=Default JDBC Database) INFO - Found EjbModule in classpath: /Users/dblevins/work/openejb3/examples/spring-integration/target/classes INFO - Beginning load: /Users/dblevins/work/openejb3/examples/spring-integration/target/classes INFO - Configuring enterprise application: classpath.ear INFO - Configuring Service(id=Default Stateless Container, type=Container, provider-id=Default Stateless Container) INFO - Auto-creating a container for bean CineplexImpl: Container(type=STATELESS, id=Default Stateless Container) INFO - Auto-linking resource-ref 'org.superbiz.spring.CineplexImpl/theaters' in bean CineplexImpl to Resource(id=theaters) 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 PersistenceUnit(name=movie-unit, provider=org.hibernate.ejb.HibernatePersistence) INFO - Enterprise application "classpath.ear" loaded. INFO - Assembling app: classpath.ear INFO - PersistenceUnit(name=movie-unit, provider=org.hibernate.ejb.HibernatePersistence) INFO - Jndi(name=CineplexImplLocal) --> Ejb(deployment-id=CineplexImpl) INFO - Jndi(name=MoviesLocal) --> Ejb(deployment-id=Movies) INFO - Created Ejb(deployment-id=Movies, ejb-name=Movies, container=Default Stateful Container) INFO - Created Ejb(deployment-id=CineplexImpl, ejb-name=CineplexImpl, container=Default Stateless Container) INFO - Deployed Application(path=classpath.ear) INFO - Exported EJB Movies with interface org.superbiz.spring.Movies to Spring bean MoviesLocal INFO - Exported EJB CineplexImpl with interface org.superbiz.spring.Cineplex to Spring bean CineplexImplLocal Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 3.141 sec Results : Tests run: 1, Failures: 0, Errors: 0, Skipped: 0