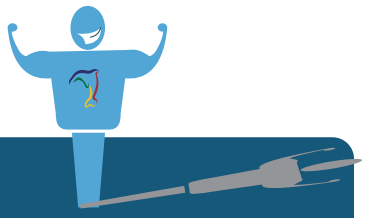
autentia

Avenida de Castilla,1 - Edificio Best Point - Oficina 21B
28830 San Fernando de Henares (Madrid)
tel./fax: +34 91 675 33 06
info@autentia.com - **www.autentia.com**

¿**Qué ofrece** Autentia Real Business Solutions S.L?

Somos su empresa de **Soporte a Desarrollo Informático**.
Ese apoyo que siempre quiso tener...

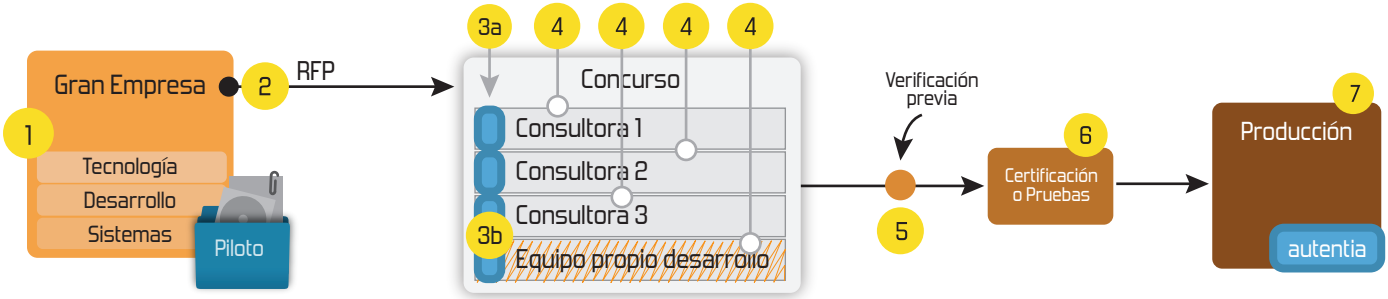
1. Desarrollo de componentes y proyectos a medida



2. Auditoría de código y recomendaciones de mejora

3. Arranque de proyectos basados en nuevas tecnologías

1. Definición de frameworks corporativos.
2. Transferencia de conocimiento de nuevas arquitecturas.
3. Soporte al arranque de proyectos.
4. Auditoría preventiva periódica de calidad.
5. Revisión previa a la certificación de proyectos.
6. Extensión de capacidad de equipos de calidad.
7. Identificación de problemas en producción.



4. Cursos de formación (impartidos por desarrolladores en activo)

Spring MVC, JSF-PrimeFaces /RichFaces, HTML5, CSS3, JavaScript-jQuery

Gestor portales (Liferay)
Gestor de contenidos (Alfresco)
Aplicaciones híbridas

Control de autenticación y acceso (Spring Security)
UDDI
Web Services
Rest Services
Social SSO
SSO (Cas)

JPA-Hibernate, MyBatis
Motor de búsqueda empresarial (Solr)
ETL (Talend)

Dirección de Proyectos Informáticos.
Metodologías ágiles
Patrones de diseño
TDD

Tareas programadas (Quartz)
Gestor documental (Alfresco)
Inversión de control (Spring)

BPM (jBPM o Bonita)
Generación de informes (JasperReport)
ESB (Open ESB)

Compartimos nuestro conocimiento en:
www.adictosaltrabajo.com

Para más información visítenos en:
www.autentia.com

Adictos Al Trabajo — En el verano, unos tiran la paja y otros recogen el grano
Powered by autentia
Hosting patrocinado por enredados

Inicio Quienes somos Tutoriales Formación Comparador de salarios Comic Charlas Más

Estas en: **Inicio** **Tutoriales** EJB 3.0 y pruebas de persistencia con Maven, JUnit 4 y Embedded JBoss sobre...

Ultimas Noticias

» Disponible la primera versión de los plugins para integrar Maven y Bugzilla.
» Historia de la Informática. Capítulo 74. 1997
» Comentando el libro: Guía de Estilo, Protocolo y Etiqueta en la empresa.
» Comentando el libro: "El viaje a la felicidad. Nuevas claves científicas" de Eduardo Punset
» Lanzamiento del nuevo Web de Autentia
» Si se pregunta ¿Qué ofrece este Web?
» Historia de la Informática. Capítulo 73. 1996 (2ª Parte)
» Autentia cumple 6 años
» Autentia colabora en la difusión de las metodologías ágiles en español

+Noticias Destacadas

» Lanzamiento del nuevo Web de Autentia
» Contratos ágiles: Vendiendo Scrum a tus clientes.
» Quinta charla Autentia + Proyectalis + Agile Spain: Contratos ágiles: Vendiendo Scrum a tus clientes
» Lo mejor de esta semana: Curso de Scrum con Ángel Medinilla

+Comentarios Cómic

+Enlaces

Tutorial desarrollado por



Jose Manuel Sánchez Suárez

Consultor tecnológico de desarrollo de proyectos informáticos. Diseñador de Adictos Al Trabajo 2.0

Puedes encontrarme en Autentia

Somos expertos en Java/J2EE

Catálogo de servicios de Autentia

Descargar (6,2 MB)

Descargar en versión comic (17 MB)

AdictosAlTrabajo.com es el Web de difusión de conocimiento de Autentia.

autentia real business solutions

Catálogo de cursos

Descargar este documento en formato PDF: embeddedJBossPersistenceTests.pdf

Fecha de creación del tutorial: 2009-09-14

EJB 3.0 y pruebas de persistencia con Maven, JUnit 4 y Embedded JBoss sobre Java 6.

0. Índice de contenidos.

- 1. Introducción.
- 2. Entorno.
- 3. La capa de persistencia.
- 4. Configuración de la unidad de persistencia.
- 5. Ejecución del test desde maven.
- 6. Conclusiones.

1. Introducción.

El hecho de que haya casi plagiado el título de este tutorial del de mi compiCarlos García sobre EJB 3.0 y pruebas unitarias con Maven, JUnit 4 y Embedded JBoss sobre Java 6 no es casual, puesto que lo que vamos a hacer es darle continuidad, probando ahora cómo llevar a cabo un test de persistencia de un EJB de entidad, bajo el soporte de un EJB de servicio que implementa el patrón dao, en el mismo entorno.

La configuración en el entorno de Embedded Jboss 2 ya la vimos de la mano de Alejandro, ahora se trata de comprobar si la misma funciona en el entorno de la versión 3.

Vamos a ir, sobre la base del tutorial de Carlos, paso a paso añadiendo la configuración necesaria para JPA, comentando los errores que se pueden ir produciendo y cómo darles una solución.

2. Entorno.

El tutorial está escrito usando el siguiente entorno:

- Hardware: Portátil Asus G1 (Core 2 Duo a 2.1 GHz, 2048 MB RAM, 120 GB HD).
- Sistema operativo: Windows Vista Ultimate.
- JDK 1.6.0_14
- Maven 2.1.
- Embedded Jboss beta3.SP10

3. La capa de persistencia.

La capa de persistencia se va a componer de un EJB de entidad y uno de servicio sin estado que expone una interfaz local.

User.java: el EJB de entidad.

Catálogo de servicios Autentia (PDF 6,2MB)



En formato comic...

Google

Web
www.adictosaltrabajo.com
Buscar

Últimos tutoriales

2009-09-14
EJB 3.0 y pruebas de persistencia con Maven, JUnit 4 y Embedded JBoss sobre Java 6.

2009-09-12
Instalación de Liferay en Tomcat existente

2009-09-11
Release Bugzilla Maven Plugin

2009-09-11
Enlazar Bugzilla con MavenChangesPlugin

2009-09-08
Sobre las reglas de codificación o... ¿de dónde salen esos caracteres "raros"?.

2009-08-28
Cómo hacer deploy del site de Maven en SourceForge

2009-08-26
Ordenación por cantidades en informe cruzado

2009-08-20
Selenium IDE-Incorporando while en los test

2009-08-14
Blender y JMonkeyEngine. Exportación de archivos Blender y uso de los mismos en JMonkeyEngine

2009-08-14
5º tutorial TNT Concept Versión 0.16.1 Gestión de informes, vacaciones y utilidades

2009-08-14
Joomla 1.5. Instalación y configuración

2009-08-13
Introducción a los diagramas EPC (Event-Driven Process Chain)

2009-08-10
Blender. Animaciones avanzadas y renderización

2009-08-10
Gestión de Calidad, tablón y seguimiento en TNT Concept Versión 0.16.1

2009-08-10
Cómo hacer una página web

2009-08-06
Tips And Tricks JUnit Spring

2009-08-03
Instalación de VirtualBox PUEL

2009-08-03
Gestión de contactos y pedidos en TNT Concept versión 0.16.1

2009-08-03
Comentando el libro: La estrategia del océano azul

2009-07-30
Funciones esenciales para crear un juego.

2009-07-30
2º tutorial TNT Concept versión 1.16.1

Ads by Google Tomcat Cluster Java Programs Video Tutorial Java Training Course Sun Java Class

Copyright 2003-2009 © All Rights Reserved | Texto legal y condiciones de uso | Banners | Powered by Autentia | Contacto

view plain print ?

```
01. package com.autentia.training.ejb3.entities;
02.
03. import java.io.Serializable;
04.
05. import javax.persistence.Column;
06. import javax.persistence.Entity;
07. import javax.persistence.GeneratedValue;
08. import javax.persistence.GenerationType;
09. import javax.persistence.Id;
10.
11. import org.apache.commons.lang.builder.EqualsBuilder;
12. import org.apache.commons.lang.builder.HashCodeBuilder;
13.
14.
15. /**
16.  * Entity linked to the table "User" of the data model.
17.  *
18.  * @author Autentia Real Business Solutions S.L.
19.  * @see http://www.autentia.com
20.  *
21.  */
22. @Entity
23. public class User implements Serializable{
24.
25.     private static final long serialVersionUID = 5729361595476015876L;
26.
27.     /** Id*/
28.     private Integer id;
29.
30.     /** The name */
31.     private String name;
32.
33.     /** The password */
34.     private String password;
35.
36.     /**
37.       * Gets the identifier of the user
38.       * Strategy : Auto increment the value of the identifier
39.       * @return Id
40.       */
41.     @Id
42.     @GeneratedValue(strategy = GenerationType.AUTO)
43.     public Integer getId() {
44.         return (Integer) id;
45.     }
46.
47.     protected void setId(Integer id){
48.         this.id = id;
49.     }
50.
51.     /**
52.       * Gets the name
53.       * @return name
54.       */
55.     @Column(nullable=false)
56.     public String getName() {
57.         return name;
58.     }
59.
60.     /**
61.       * Sets the name
62.       * @param name
63.       */
64.     public void setName(String name) {
65.         this.name = name;
66.     }
67.
68.     /**
69.       * Gets the password
70.       * @return password
71.       */
72.     @Column(nullable=false)
73.     public String getPassword() {
74.         return password;
75.     }
76.
77.     /**
78.       * Sets the password
79.       * @param password
80.       */
81.     public void setPassword(String password) {
82.         this.password = password;
83.     }
84.
85.     /**
86.       * Get the hashCode of the user.
87.       * <p>
88.       * Uses all fields except those that are the subject of an inverse relationship
89.       * </p>
90.       * @return hashCode
91.       */
92.     @Override
93.     public int hashCode() {
94.         final HashCodeBuilder hcb = new HashCodeBuilder();
95.         hcb.append(super.hashCode());
96.         hcb.append(getName());
97.         hcb.append(getPassword());
98.         return hcb.toHashCode();
99.     }
100.
101.    /**
102.      * Verifies that the object passed as parameter is equal to the current.
103.      * <p>
104.      * Uses all fields except those that are the subject of an inverse relationship
105.      * </p>
106.      * @param an user object
107.      * @return the result of the equals method
108.      */
109.    @Override
110.    public boolean equals(Object obj) {
111.        if (this == obj) {
112.            return true;
113.        }
114.        try {
115.            final User other = (User) obj;
116.            final EqualsBuilder eqb = new EqualsBuilder();
117.            eqb.append(getName(), other.getName());
118.            eqb.append(getPassword(), other.getPassword());
119.            return eqb.isEquals();
120.        } catch (Exception e) {
121.            // if nay error returns false
122.        }
123.        return false;
124.    }
125.
126. }
```

Dao.java: el interfaz de EJB Local.

2009-07-29
Hibernate Search, Bridges, Analizadores y más

2009-07-24
Migración de EJB3 a JPA y Spring.

2009-07-20
Directorio de ejemplos de jMonkey Engine

2009-07-19
JSR-179 Location API para J2ME: Posicionamiento geográfico en nuestras aplicaciones.

2009-07-16
Gestión de Usuarios en TNT Concept versión 0.16.1

2009-07-16
Continuación del Tutorial: JMonkeyEngine, Creación de nuestro primer juego.

2009-07-16
Como implementar el Scene Monitor para analizar las escenas en jMonkeyEngine

2009-02-26
Transformaciones de escena en JMonkeyEngine

2009-07-15
Detalles del juego de la moto en jMonekyEngine.

2009-07-14
JMonkeyEngine, Creación de nuestro primer juego.

2009-07-13
Ajax tests con Selenium: prototype.js e ICEfaces.

2009-07-08
AOP con AspectJ y Maven

2009-07-07
Instalación y configuración de Eclipse Galileo

2009-07-07
Iniciarse en el manejo de JME, Creación de un Cloth.

2009-07-06
Primeros pasos con Blender: Pintando nuestra mascota en 3D

2009-07-06
DBUnit-Exportar e Importar BBDD

2009-07-05
JMeter, Pruebas de stress sobre aplicaciones web: Grabando y reproduciendo navegaciones

2009-07-02
Axis2: Invocación de Servicios Web usando distintos MEP

2009-07-02
Instalación OpenOffice

2009-07-02
Juegos 3D en Java: Blender y JMonkeyEngine

2009-06-20
StAX (Xml Pull Parser): Streaming API para XML

2009-06-15
Configuración de la desconexión de usuarios con ICEFaces

2009-06-10
LWUIT: Una librería gráfica tipo AWT o Swing para J2ME

2009-06-10
Mapas mentales con XMind

2009-02-26
Redimensionar Imagenes en Windows Vista

2009-06-08
UploadFile con Icefaces + Hibernate + Anotaciones

2009-06-05
Habilitar exportación en Liferay

2009-06-01
Registrar Liferay en Eclipse

```
view plain print ?
01. package com.autentia.training.ejb3.persistence;
02.
03. import java.io.Serializable;
04.
05. import javax.ejb.Local;
06.
07. /**
08.  * Base interface for all DAO classes in the application.
09.  *
10.  * <p>
11.  * For more information on the DAO pattern, visit: <a href=
12.  * "http://java.sun.com/blueprints/corej2eepatterns/Patterns/DataAccessObject.html"
13.  * target=" blank">http://java.sun.com/blueprints/corej2eepatterns/Patterns/
14.  * DataAccessObject.html</a>.
15.  *
16.  * @author Autentia Real Business Solutions S.L.
17.  * @see http://www.autentia.com
18.  *
19.  */
20. @Local
21. public interface Dao {
22.
23.     /** Persists the entity object into database
24.       * @param entity entity to persist
25.       */
26.     public void persist(Object entity);
27.
28.     /** Removes an object from persistent storage in the database
29.       * @param entity entity to remove
30.       */
31.     public void remove(Object entity);
32.
33.
34.     /** Retrieves an object of the Class indicated, that was previously persisted to the database,
35.       *  using the indicated id as primary key
36.       *  @param entityClass
37.       *  @param id
38.       *  @return an entity
39.       */
40.     public <T> T get(Class<T> entityClass, Serializable id);
41.
42.    ...
43.
44. }
```

GenericDao.java: el EJB de servicio que expone la interfaz Local.

```
view plain print ?
01. package com.autentia.training.ejb3.persistence;
02.
03. import java.io.Serializable;
04.
05. import javax.ejb.Stateless;
06. import javax.persistence.EntityManager;
07. import javax.persistence.PersistenceContext;
08.
09. /**
10.  *  Generic DAO, provides basic CRUD operations, based in JPA EntityManager.
11.  *
12.  * @author Autentia Real Business Solutions S.L.
13.  * @see http://www.autentia.com
14.  *
15.  */
16. @Stateless
17. public class GenericDao implements Dao {
18.
19.     @PersistenceContext
20.     private EntityManager em;
21.
22.     /** @see Dao#persist(TransferObject) */
23.     public void persist(Object entity) {
24.         em.persist(entity);
25.     }
26.
27.     /** @see Dao#remove(TransferObject) */
28.     public void remove(Object entity) {
29.         em.remove(em.merge(entity));
30.     }
31.
32.     /** @see Dao#get(Class, Serializable) */
33.     public <T> T get(Class<T> entityClass, Serializable id) {
34.         return (T) em.find(entityClass, id);
35.     }
36.
37.     ...
38.
39. }
```

Estas clases se ubicarán bajo la carpeta src/main/java, puesto que son clases de aplicación y para que compilen necesitaremos añadir la dependencia al siguiente artefacto en el pom.xml:

```
view plain print ?
01. <dependency>
02.     <groupId>javax.persistence</groupId>
03.     <artifactId>persistence-api</artifactId>
04.     <version>1.0</version>
05. </dependency>
```

El test es el siguiente. Muestro sólo el método, puesto que la clase que tiene el soporte para levantar el Embedded Jboss, la podéis encontrar en eltutorial de Carlos. Estará ubicado bajo la carpeta src/test/java:

```
view plain print ?
01. @Test
02. public void userCrudOperations() throws javax.naming.NamingException  {
03.     InitialContext ctx = new InitialContext();
04.     Dao dao = (GenericDao)ctx.lookup("GenericDao/local");
05.
06.     User user = new User();
07.     user.setName("Jose Manuel Sánchez");
08.     user.setPassword("password");
09.
10.     dao.persist(user);
11.
12.     Assert.assertNotNull("User not found by id:' + user.getId(), dao.get(User.class, user.getId()));
13. }
```

Si ejecutamos el test sin más, se producirá la siguiente excepción:

```
view plain print ?
01. javax.naming.NameNotFoundException: GenericDao not bound
```

puesto que es necesario añadir, al jar que se despliega, las clases con las que estamos trabajando:

```
view plain print ?
01. private static void deploy() throws DeploymentException {
02.     jar = AssembledContextFactory.getInstance().create("ejbTestCase.jar");
03.     jar.addClass(User.class);
04.     jar.addClass(Dao.class);
05.     jar.addClass(GenericDao.class);
06.
07.    ...
08.
```

Si os parece tedioso añadir una a una las clases necesarias, existe, entre otros, un método como el que sigue, que añade todas las clases de un paquete raíz a partir del package de la

2009-05-29
Liferay Social Office

Últimas ofertas de empleo

2009-07-31
T. Información - Operador (dia / noche) - BARCELONA.

2009-06-25
Atención a cliente - Call Center - BARCELONA.

2009-06-19
Otras - Ingenieria (minas, puentes y puertos) - VALENCIA.

2009-06-17
Comercial - Ventas - ALICANTE.

2009-06-03
Comercial - Ventas - VIZCAYA.

Ads by Google

clase que se pasa como primer parámetro:

```
view plain print ?
01. jar.addResources(User.class, new String[]{"**/entities/*.class"}, null);
```

4. Configuración de la unidad de persistencia.

En este punto el test aún no pasa, se debería producir la siguiente excepción:

```
view plain print ?
01. java.lang.RuntimeException: Illegal @PersistenceUnit on private javax.persistence.EntityManager
02. com.autentia.training.ejb3.persistence.GenericDao.em :There is nodefault persistence unit in this deployment.
```

Nos falta un fichero **persistence.xml** en el que definiremos la unidad de persistencia:

- **src/main/resources/META-INF/persistence.xml**: con la configuración para el entorno de producción, esto es, la de la aplicación en un entorno no de tests,
- **src/test/resources/persistence-test.xml**: con la configuración para el entorno de tests. El cambio de nomenclatura del fichero respecto al de producción, al anterior, viene dado porque ambos estarán en el classpath de la ejecución de los tests, de ahí que no convenga tampoco incluirlo dentro de una carpeta META-INF dentro de resources, para que no colisionen.

El contenido del fichero persistence-test.xml, podría ser como sigue:

```
view plain print ?
01. <?xml version="1.0" encoding="UTF-8"?>
02.   <persistence version="1.0"
03.                  xmlns="http://java.sun.com/xml/ns/persistence"
04.                  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
05.                  xsi:schemaLocation="http://java.sun.com/xml/ns/persistence http://java.sun.com/xml/ns/persistence/persistence_1_0.xsd">
06.
07.     <persistence-unit name="testPersistenceUnit" transaction-type="JTA">
08.
09.       <jta-data-source>java:DefaultDS</jta-data-source>
10.
11.       <class>com.autentia.training.ejb3.entities.User</class>
12.
13.       <properties>
14.             <property name="hibernate.dialect" value="org.hibernate.dialect.HSQLDialect" />
15.             <property name="hibernate.hbm2ddl.auto" value="create"/>
16.             <property name="hibernate.show_sql" value="true" />
17.       </properties>
18.
19.     </persistence-unit>
20.
21.   </persistence>
```

La fuente de datos DefaultDS se incluye por defecto en el entorno de Embedded Jboss, la podéis encontrar en \src\test\resources\deploy\hsqldb-ds.xml, y configura una base de datos hsqldb.

Simplemente con incluir el fichero persistence-test.xml al directorio de resources no funciona puesto que tenemos que añadirlo al jar que se despliega. Así creammos un directorio de ensamblado que se llame META-INF dentro del jar y añadimos el fichero sin el path:

```
view plain print ?
01. private static void deploy() throws DeploymentException {
02.     jar = AssembledContextFactory.getInstance().create"ejbTestCase.jar");
03.
04.     ...
05.
06.   AssembledDirectory metainf = jar.mkdir"META-INF");
07.     metainf.addResource("persistence-test.xml", "persistence.xml");
08.
09.     Bootstrap.getInstance().deploy(jar);
10.
11.     ...
12.  }
```

En este punto, ejecutando el test desde Eclipse con la propiedad "-Dsun.lang.ClassLoader.allowArraySyntax=true" asignada en el arranque de la VM, debería funcionar correctamente, aunque suelta la siguiente traza a nivel de warning:

```
view plain print ?
01. WARN  [org.hibernate.ejb.packaging.InputStreamZippedJarVisitor] Unable to find file (ignored): vf//12526767712961/
02. java.lang.RuntimeException: Cannot open stream
03.     at org.jboss.virtual.plugins.context.vfs.AssembledDirectoryHandler.openStream(AssembledDirectoryHandler.jav117)
```

Si bien, con maven desde línea de comandos, deberíamos tener esta maravillosa excepción:

```
view plain print ?
01. ERROR [org.jboss.kernel.plugins.dependency.AbstractKernelController] Error installing to Parse: name=vf//12526314481830/
02. state=Not Installed mode=Manual requiredState=Parse
03. java.lang.AssertionError: expected a jar or file url, but was vfs//12526297739780/
```

vfs o Virtual File System es una abstracción de un sistema de ficheros, precisamente el que se crea para ensamblar el jar y el número es el nombre del mismo.

Pero... ¿cuál es el problema?

- parece que el encargado de desplegar la unidad de persistencia ahora no soporta este sistema de ficheros, puesto que en la versión 2 de Embedded Jboss no teníamos este problema,
- pero ¿será una problema de classpath?, puesto que desde el entorno de Eclipse no se reproduce.
- ¿habrá sido al subir de versión la JDK?, no, porque es la misma que la del Eclipse, la 1.6, y ahí sí funciona,
- ...

Como hemos comentado en el resto de tutoriales sobre Embedded Jboss, está aún en versión beta, la documentación es escasa y las respuestas en los foros a las cuestiones que se plantean no son de mucha calidad.

La casuística de nuestro error es la misma que se describe en este post del foro de Jboss, y la solución que le dan pasa por dejar de usar el directorio virtual y realizar un scan de un directorio físico para que monte todo lo que se encuentre dentro de él.

```
view plain print ?
01. private static void deploy() throws DeploymentException {
02.  Bootstrap.getInstance().scanClasspath"embeddedJBoss3Java6\\target\\classes"); // donde embeddedJBoss3Java6 es el nombre del proyecto
03. }
```

También se podría incluir algo como lo que sigue, que permite desplegar las librerías que contegan el recurso que recibe como parámetro.

```
view plain print ?
01. private static void deploy() throws DeploymentException {
02. Bootstrap.getInstance().deployResourceBases"META-INF/persistence.xml");
03. }
```

Y el caso es que ambas soluciones funcionan, pero no es lo que queremos puesto que está levantando la unidad de persistencia que se encuentra en el directorio src/main/resources/META-INF, se podría parchear en base a mover ficheros de un sitio a otro con la ayuda del plugin de resources de maven, pero estaríamos complicandolo más aún. Hay que poner especial cuidado en revisar la siguiente traza para comprobar qué unidad de persistencia está levantando:

```
view plain print ?
01. INFO  [org.hibernate.ejb.Ejb3Configuration] Processing PersistenceUnitInfo [
02. name: testPersistenceUnit
03. ...]
```

5. Ejecución del test desde maven.

Entonces... ¿qué diferencia existe entre la ejecución desde Eclipse y con maven desde línea de comandos?, la respuesta es sencilla, aunque ha costado algo más de lo deseado encontrarla.

El plugin de surefire es el encargado de ejecutar los tests en el entorno de maven y por defecto tiene activa la propiedad**enableAssertions** que habilita las aserciones de la JVM en la ejecución de los tests, modificando el comportamiento por defecto de la JVM, que lo tiene desactivado y que es como se ecuentra en el entorno de ejecución desde Eclipse.

Las asercciones o asertos son instrucciones del propio lenguaje Java, que permiten poner a prueba suposiciones dentro de la lógica de un programa. La lógica es la misma de los Assert.assert de los tests de Junit, pero forman parte del propio lenguaje, con lo que los evalúa la propia JVM en tiempo de ejecución.

```
view plain print ?
01. void foo() {
02.     for (...) {
03.         if (...)
04.             return;
05.     }
06.     assert false; // Execution should never reach this point!
07. }
08.
09. // Confirm adherence to precondition in nonpublic methoc
10. assert interval > 0 && interval <= 1000/MAX_REFRESH_RATE : interval;
11.
12. // org.jboss.ejb3.deployers.PersistenceUnitParsingDeployer.java:11(
13. assert persistenceUnitRootUrl.getProtocol().equals("jar") || persistenceUnitRootUrl.getProtocol().equals("file") :
14. "expected a jar or file url, but was " + persistenceUnitRootUrl;
```

El último de los tres ejemplos anteriores es precisamente el código que está lanzando la excepción en la ejecución del test que nos ocupa, ejecutando maven por línea de comandos, y el caso es que la aserción no debe ser tal, o no tan importante, puesto que desde Eclipse el test funciona y deshabilitando los assert en el plugin de surefire también.

```
view plain print ?
01. <plugin>
02.     <groupId>org.apache.maven.plugins</groupId>
03.     <artifactId>maven-surefire-plugir</artifactId>
04.     <version>2.4</version>
05.     <configuration>
06.         <enableAssertions>false</enableAssertions>
07.         <argLine>-Dsun.lang.ClassLoader.allowArraySynta=true</argLine>
08.     </configuration>
09. </plugin>
```

Deshabilitando la propiedad enableAssertions, el test funciona, aunque seguimos teniendo la traza a nivel de warning que comentábamos más arriba. Existe unhilo en el foro de Jboss donde tratan el tema, pero no parece que sea demasiado relevante puesto que se trata de un error al abrir el directorio virtual, seguramente producido por no estar activo el assert. El caso es que se realiza un scan en busca de clases anotadas, y se produce esa excepción. Como nosotros las incluimos en el persistence.xml no pasa de ser un warning.

Por último, hemos comprobado que bajo una versión 1.5 de Java los tests también pasan, con el mismo warning.

9. Conclusiones.

Recordar que Embedded Jboss sigue en versión beta y ya lleva unos años así, y este tipo de problemas con la subida de versiones los tenemos que ir asumiendo. Al subir de versión una aplicación, tendremos que adaptar el entorno de pruebas si es necesario, pero puede ocurrirnos lo contrario, que no podamos subir de versión la tecnología de una aplicación porque no está soportada en el entorno de pruebas.

Por otro lado, si cuando comenzamos con Embeded Jboss era la única solución disponible para hacer uso de un micro-contenedor de EJBs en el entorno de nuestros tests, hoy día existen más y alguna bastante interesante como Apache Open EJB, que iremos evaluando, en breve.

Un saludo.

Jose

mailto:jmsanchez@autentia.com

¿Qué te ha parecido el tutorial? Déjanos saber tu opinión y ivota!

Muy malo Malo Regular Bueno Muy bueno

Votar

Anímate y coméntanos lo que pienses sobre este tutorial

Puedes opinar o comentar cualquier sugerencia que quieras comunicarnos sobre este tutorial; con tu ayuda, podemos ofrecerte un mejor servicio.

Nombre: E-Mail:

Comentario:

Enviar comentario **Texto Legal y condiciones de uso**

- Puedes inscribirte en nuestro servicio de notificaciones haciendo clic aquí.
- Puedes firmar en nuestro libro de visitas haciendo clic aquí.
- Puedes asociarte al grupo AdictosAlTrabajo en XING haciendo clic aquí.
- Añadir a favoritos Technorati. ADD THIS BLOG TO MY Technorati FAVORITES

SOME RIGHTS RESERVED Esta obra está licenciada bajo licencia Creative Commons de Reconocimiento-No comercial-Sin obras derivadas 2.5

Recuerda

Autentia te regala la mayoría del conocimiento aquí compartido (Ver todos los tutoriales). Somos expertos en: J2EE, Struts, JSF, C++, OOP, UML, UP, Patrones de diseño ... y muchas otras cosas.

¿Nos vas a tener en cuenta cuando necesites consultoría o formación en tu empresa?, ¿Vas a ser tan generoso con nosotros como lo tratamos de ser con vosotros?

Somos pocos, somos buenos, estamos motivados y nos gusta lo que hacemos ...

Autentia = Soporte a Desarrollo & Formación.

info@autentia.com

soluciones reales para SU

Tutoriales recomendados

Nombre	Resumen	Fecha	Visitas	Valoración	Votos	Pdf
Instalación de Liferay en Tomcat existente	En este tutorial vamos a solventar el problema de instalar Liferay cuando ya contamos con un Tomcat existente y necesitamos que Liferay conviva con el resto de aplicaciones que corren en el servidor.	2009-09-12	68	-	-	
Enlazar Bugzilla con MavenChangesPlugin	En este tutorial veremos como enlazar Bugzilla con MavenChangesPlugin	2009-09-11	45	-	-	
Release Bugzilla Maven Plugin	En este tutorial vamos a mostrar como automatizar un conjunto de acciones que hay que hacer siempre en los sistemas de gestión de incidencias, tales como dar de alta una nueva versión del producto, cerrar las incidencias que soluciona la nueva versión, et	2009-09-11	52	-	-	
Sobre las reglas de codificación o... ¿de dónde salen esos caracteres "raros"?.	En este tutorial vamos a tratar de dar algo de luz a los errores que tenemos habitualmente con la codificación de caracteres en aplicaciones en las que se ven implicados varios sistemas que intercambian o almacenan información.	2009-09-08	270	Muy bueno	4	
Cómo hacer deploy del site de Maven en SourceForge	Este tutorial nos enseña un poco mas sobre Maven	2009-08-28	332	Muy bueno	1	
Selenium IDE-Incorporando while en los test	En este tutorial veremos como ampliar los comandos disponibles en Selenium IDE adaptandolos a nuestras necesidades.	2009-08-20	397	Bueno	5	
Joomla 1.5. Instalación y configuración	Veamos en detalle cómo instalar Joomla 1.5 y aplicar algunas configuraciones posteriores de las disponibles en este CMS	2009-08-14	1824	Muy bueno	11	
Instalación de VirtualBox PUEL	En este tutorial os enseñamos a instalar VirtualBox como alternativa a VMWare para la virtualización de sistemas operativos.	2009-08-03	1093	Bueno	8	
Migración de EJB3 a JPA y Spring.	Este tutorial de como emigrar una aplicación montada con EJB3 y JSF al soporte que proporciona Hibernate para JPA y a Spring, con el mismo soporte de JSF.	2009-07-24	1192	Muy bueno	4	
AOP con AspectJ y Maven	Programacion orientada a aspectos con AspectJ y Maven	2009-07-08	1031	Bueno	1	

Nota:

Los tutoriales mostrados en este Web tienen como objetivo la difusión del conocimiento. Los contenidos y comentarios de los tutoriales son responsabilidad de sus respectivos autores. En algún caso se puede hacer referencia a marcas o nombres cuya propiedad y derechos es de sus respectivos dueños. Si algún afectado desea que incorporemos alguna reseña específica, no tiene más que solicitarlo. Si alguien encuentra algún problema con la información publicada en este Web, rogamos que informe al administrador rcanales@adictosaltrabajo.com para su resolución.