Cómo configurar datasources en el servidor de aplicaciones Jboss, versión 7

La configuración de *datasources* en esta versión del servidor *JBoss* es bastante diferente a la empleada en versiones anteriores[1]. Es por ello que se hace imprescindible conocer mínimamente el nuevo mecanismo de configuración.

Aunque puede emplearse la anotación Java (disponible desde JEE 6) *@DataSourceDefinition*, esta es sólo recomendable para propósitos de test o desarrollo y nunca para producción. Es por ello que nos centraremos en la configuración basada en archivos.

Para aprovechar las capacidades de un pool de conexiones, que se suele asociar a los datasources en los servidores de aplicaciones, deberán realizarse dos tareas:

a) Hacer disponible el controlador JDBC de Java del servidor de base de datos correspondiente en el servidor de aplicaciones JBoss.

b) Configurar el datasource, una vez realizada la tarea anterior.

Completados ambos ya se puede proceder a su uso a través del servicio *JNDI* del servidor de aplicaciones.

Para el ejemplo que vamos a ver, se supone que existe una base de datos *MySQL* denominada *bdtest*, con privilegios de acceso para el usuario *eclipse* con clave *eclipse*.

Instalación del controlador JDBC

En nuestras aplicaciones web emplearemos siempre controladores JDBC de tipo 4[2], y el sistema gestor con el que desarrollaremos tanto los ejemplos del *Fotograma Perdido* como los proyectos finales de módulo será *MySQL 5*, proporcionada por la distribución *Ubuntu 11.04*.

El controlador puede instalarse siguiendo dos posibles vías.

a) Ubicándolo en el área de despliegue, como cualquier archivo empaquetado que pueda procesarse desde el directorio de despliegue.

En este caso se trata de un archivo JAR, pero este tipo de formato también es reconocible por el servidor de aplicaciones. Se copia en *<JBOSS_HOME>/standalone/deployments*. Cualquier controlador compatible con *JDBC 4.0* se reconocerá automáticamente y se cargará en el sistema con su nombre y versión corresponientes. Si el controlador JDBC no fuera compatible con JDBC 4.0 puede serlo aplicando diferentes estrategias. Quizás la más simple de ellas es modificar el archivo JAR del controlador aplicando los pasos siguientes:

- Se crea un directorio temporal (da igual el nombre)
- Se crea dentro de ese directorio temporal un subdirectorio denominado *META-INF*.
- Se crea *META-INF/services*.
- Se crea el archivo *META-INF/services/java.sql.Driver*, que contendrá una sola línea de texto: el nombre de clase completamente cualificada del controlador JDBC.
- Se emplea el comando *jar* (desde el directorio temporal) para actualizar el archivo JAR del controlador con objeto de que incorpore el nuevo archivo en el paquete.
  jar -uf <jdbc-driver>.jar META-INF/services/java.sql.Driver

Todo ello requiere posterior configuración del datasource mediante las etiquetas correspondientes.

b) Instalar el controlador JDBC como un módulo.

Dentro del directorio raíz del servidor JBoss hay un directorio llamado *modules*. Se debe crear entonces un

1 Ya no se emplean archivos -ds.xml, que incluían el nombre del datasource.
2 Más información en:
http://www.gxtechnical.com/gxdlsp/pub/genexus/java/docum/manuals/8.0/mjavab2-2.htm
http://www.javadabbadoo.org/apendice/bds/jdbc/tiposDrivers/index.html

módulo para MySQL con una estructura similar a la que posee la base de datos H2 que se proporciona por defecto en el servidor. La estructura de directorios a crear será la siguiente:

<JBOSS_HOME>/modules/com/mysql/jdbc/main

Dentro del directorio main se precisa crear el archivo *module.xml* que configurará el módulo, además de incorporar también en ese directorio el archivo JAR que contiene el controlador de la base de datos. En nuestro caso, *mysql-connector-java-5.1.17-bin.jar*. Este archivo JAR es compatible con JDBC 4.0, de manera que contiene un archivo *META-INF/services/java.sql.Driver*.

El contenido del archivo *module.xml* es el siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>
<module xmlns="urn:jboss:module:1.0" name="com.mysql.jdbc">
	<resources>
		<resource-root path="mysql-connector-java-5.1.17-bin.jar"/>
	</resources>
	<dependencies>
		<module name="javax.api"/>
		<module name="javax.transaction.api"/>
	</dependencies>
</module>
```

Observar que el nombre del módulo (*com.mysql.jdbc*) coincide con la estructura de directorios creada bajo *modules*.

En la etiqueta *<resource-root>* se establece el nombre del archivo controlador (ruta relativa respecto a *com/mysql/jdbc)*.

Configuración del datasource

Todo esto aún no es suficiente para que JBoss 7 lo arranque como servicio asociado a datasources. Es preciso configurar los datasources haciendo referencia a este módulo. Debe recordarse que JBoss 7 posee dos estrategias diferentes de ejecución: *domain* y *standalone*. La primera orientada a poder lanzar varias instancias del mismo servidor, la segunda para ofrecer sólo una. Nosotros trabajaremos siempre en modo standalone. Sin embargo, la manera en que se configura el datasource es independiente de la configuración de ejecución del servidor, en un caso se modificará el archivo *<JBOSS_HOME>/domain/configuration/domain.xml*, y en el otro el archivo *<JBOSS_HOME>/standalone/configuration/standalone.xml*. Nosotros modificaremos el archivo *standalone.xml*.

La modificación se lleva a cabo en el interior de la etiqueta

<subsystem xmlns="urn:jboss:domain:datasources:1.0">

que contendrá una etiqueta *<datasources>* y en su interior se dispondrá de una etiqueta *<datasource>* por cada datasource que se configure. Dentro de la etiqueta *<datasources>* también se ubicará una etiqueta *<drivers>* que a su vez contendrá tantas etiquetas *<driver>* como controladores estén configurados en el servidor.

El código que debe incluirse es el siguiente:

```
<datasource    jndi-name="java:jboss/datasources/MySqlDS"
               pool-name="MySqlDS"
               enabled="true"
               jta="true"
               use-java-context="true"
               use-ccm="true">
	<connection-url>
		jdbc:mysql://localhost:3306/bdtest
	</connection-url>
	<driver>
		mysql5
```

```
        </driver>
        <transaction-isolation>
                TRANSACTION_READ_COMMITTED
        </transaction-isolation>
        <pool>
                <min-pool-size>
                        10
                </min-pool-size>
                <max-pool-size>
                        20
                </max-pool-size>
                <prefill>
                        true
                </prefill>
        </pool>
        <security>
                <user-name>
                        eclipse
                </user-name>
                <password>
                        eclipse
                </password>
        </security>
</datasource>
<drivers>
        <driver name="mysql5" module="com.mysql.jdbc">
                <xa-datasource-class>
                        com.mysql.jdbc.jdbc2.optional.MysqlXADataSource
                </xa-datasource-class>
        </driver>
        <driver name="h2" module="com.h2database.h2">
                <xa-datasource-class>
                        org.h2.jdbcx.JdbcDataSource
                </xa-datasource-class>
        </driver>
</drivers>
```

En la etiqueta *<driver>* en *<datasource>* hay que proporcionar el nombre único de controlador especificado en la etiqueta *<driver>* dentro de *<drivers>*. En la etiqueta *<drivers>* se especifica también, junto al nombre de controlador, el nombre del módulo que contiene la configuración del controlador, que estará especificado en su correspondiente archivo *module.xml*. La etiqueta *<xa-datasource-class>* hace referencia al nombre de la clase que da soporte de datasources y proporciona funcionalidades especiales en el uso de pool de conexiones y transacciones distribuidas.

Si se hubiera instalado el controlador por la vía de despliegue en lugar de como módulo, la etiqueta *<driver>* de *<datasource>* deberá contener el nombre del archivo controlador desplegado.

Notas sobre algunas etiquetas usadas en la configuración del datasource[3]

etiqueta	descripción
jndi-name	Nombre JNDI del datasource.
pool-name	Nombre del pool de conexiones para el datasource.
enabled	Indica si el datasource debe estar activo o no tras arrancar el servidor.
use-java-context	Si se pone a false, se asocia a un contexto global JNDI.
use-ccm	Activa un gestor de caché de conexiones.
connection-url	La URL de conexión del controlador.
driver (dentro de la etiqueta datasource)	Un nombre de inicio de controlador JDBC especificado en la sección <drivers>, o bien, el nombre del archivo JAR si se ha aplicado el mecanismo de despliegue.
transaction-isolation	

3 Más información en:
http://docs.jboss.org/ironjacamar/userguide/1.0/en-US/html/deployment.html#deployingds_descriptor

etiqueta	descripción
min-pool-size	Número mínimo de conexiones que un pool debe asegurar.
max-pool-size	Número máximo de conexiones de un pool.
prefill	Si rellena el pool de conexiones antes de que se produzca la primera petición sobre él.

Cómo probar un datasource sin escribir código

Basta con ejecutar desde la consola de línea de comandos[5] las dos siguientes instrucciones:

```
connect localhost:9999

/subsystem=datasources/data-source=java\:jboss\/datasources\/MySqlDS:test-connection-in-pool
```

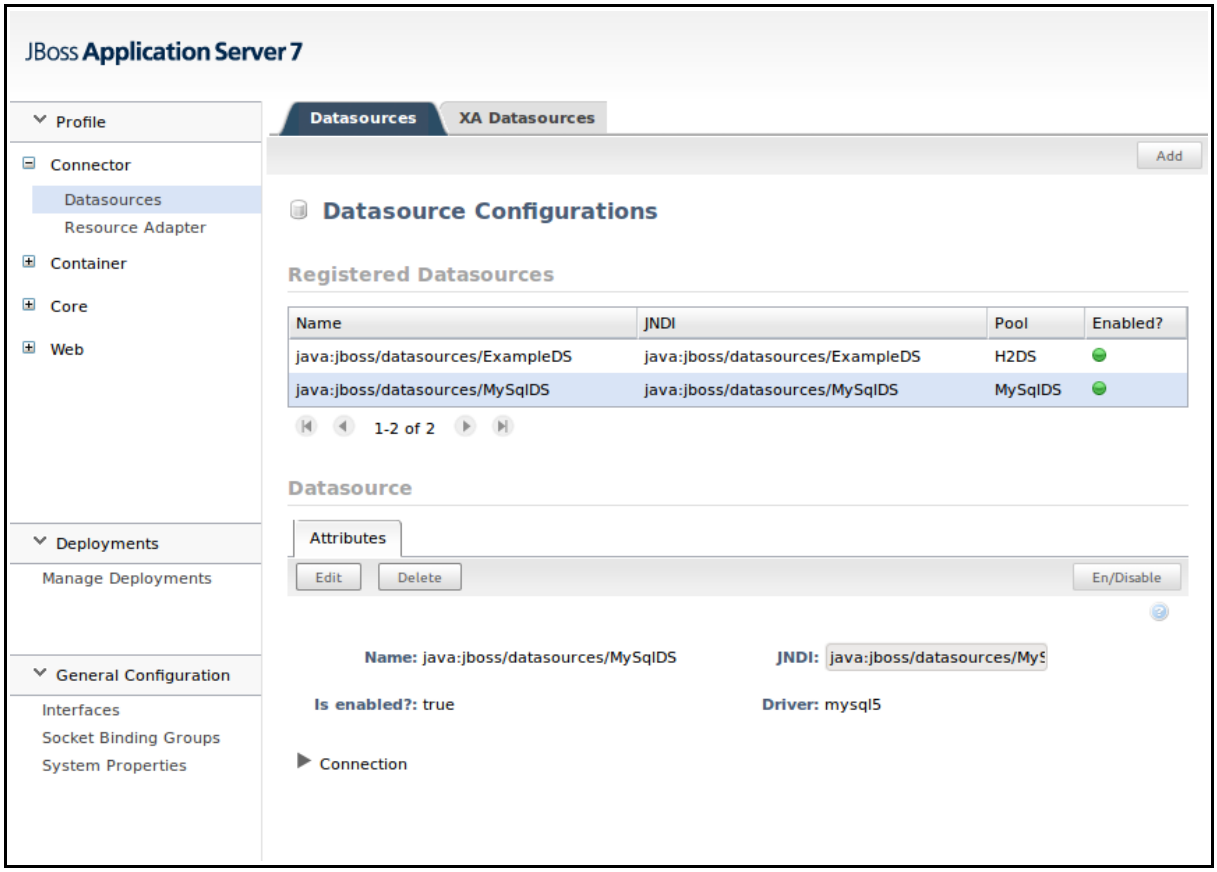
Si todo ha ido bien, se mostrará el siguiente mensaje:

```
{
    "outcome" => "success",
    "result" => [true]
}
```

Comprobar desde la aplicación de administración web de Jboos

Se accede a http://localhost:9090/console

Y desde ahí, a la entrada Datasources de Connector. Si todo ha ido bien, veremos una pantalla como la siguiente:



4 Más información al respecto en:
http://www.guillesql.es/Articulos/SQLServerFAQ_Aislamiento_Transaccion_Isolation_Level.aspx

5 Para acceder a la consola de administración: *jboss-admin.sh* desde <JBOSS_HOME>/bin

Probar el datasource mediante una página JSP

He aquí el cógido de la página JSP que puede emplearse para comprobar que la conexión al datasource se ha llevado a cabo con éxito:

```
<!-- Directivas JSP -->
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@page import="java.sql.*,javax.naming.*,javax.sql.*, java.io.*" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html">
        <title>Ejemplo de conexión a BD con DataSource</title>
    </head>
    <body>
        <%
        Connection conDB = null;
        out.println("Intentando conectar con la base de datos mediante DataSource...");
        try {
                        InitialContext initCtx = new InitialContext();
                        DataSource ds = (DataSource) initCtx.lookup("java:jboss/datasources/MySqlDS");
                        conDB = ds.getConnection();
                        Statement s = conDB.createStatement();
                        out.println("Todo está ¡¡OK!!");
        }
        catch(Exception e)
        {
                out.println("No ha sido posible conectarse.\n"+e.toString());
        }
        finally
        {
                if (conDB!=null)
                        conDB.close();
                conDB = null;
        }

        %>
    </body>
</html>
```