FUENTES DE DATOS JNDI EN TOMCAT 10

Dpto. Informática y comunicaciones.
Curso. 2024 / 2025
José Luís Casado Valero

Virrey Morcillo
Instituto de Educación Secundaria

1. Fuente de datos JNDI

1.1 Introducción

Para dar soporte de acceso a datos de diferentes fuentes de datos debemos dotar a Tomcat de los drivers de cada uno de los tipos de conexiones que vamos a utilizar, para ello, Apache Tomcat activando el escaneo del controlador durante el inicio de Tomcat. Esto está habilitado por defecto.

Significa que solo las bibliotecas visibles para el cargador de clases común y sus padres serán escaneadas en busca de controladores de base de datos. Esto incluye controladores en $CATALINA_HOME/lib, $CATALINA_BASE/lib.

Los controladores empaquetados en aplicaciones web (en WEB-INF/lib) no serán visibles y no se cargarán automáticamente.

En nuestro caso, deberemos copiar la librería del conector correspondiente en el directorio lib del servidor de aplicaciones Apache Tomcat ($CATALINA_BASE/lib).

Por ejemplo para mysql : **mysql-connector-java-8.0.17.jar**

1.2 Configuración del contexto

Una vez cardados los controladores necesarios, necesitamos indicar a la aplicación la configuración de la conexión a datos que puede utilizar, para ello, agregaremos una declaración al contexto de la aplicación. Fichero context.xml situado en META-INF.

Por ejemplo:

```
<Context>
    <Resource url="jdbc:mysql://localhost:3306/incidencias?autoReconnect=true" name="jdbc/incidencias"
      username="root" validationQuery="SELECT 1" auth="Container" driverClassName="com.mysql.jdbc.Driver"
      maxTotal="100" maxIdle="30" maxWaitMillis="10000" password="12345678" allowPublicKeyRetrieval="true"
      useSSL="false" serverTimezone="UTC"
      removeAbandonedOnBorrow="true" testOnBorrow="true" type="javax.sql.DataSource" />
</Context>
```

Donde:

- url es la url donde se encuentra la base de datos.
- Name es el nombre de la fuente de datos, el que se utilizará durante la aplicación
- username es el usuario de conexión
- validationQuery es la consulta utilizada para comprobar que la conexión esta viva.
- Auth indica cuando se realiza la conexión, si a nivel de aplicación o contenedor.
- DriverClassName es el nombre de la clase del driver.

- MaxTotal es el numero máximo de conexión en el pool (cola) de conexiones.
- MaxIdle es el numero máximo de conexiones en desuso en la cola.
- MaxWaitMillis es el tiempo máximo de espera a la conexión.
- Password es la contraseña
- allowPublicKeyRetrieval define si esta permitida la clave publica en la conexión.
- UseSSL indica si se usa comunicación cifrada.
- ServerTimezone es la zona horaria utilizada.
- RemoveAbandonedOnBorrow indica si son eliminadas y recicladas las conexiones abandonadas.
- TestOnBorrow indica si comprueba el estado de la conexión antes de utilizarla
- type es el nombre de la clase Java que espera la aplicación cuando se requiere el recurso.

1.3 configuración de web.xml

Por último creamos una referencia al la fuente de datos en nuestro fichero de configuración WEB-INF/web.xml:

```
<web-app version="6.0" xmlns="https://jakarta.ee/xml/ns/jakartaee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="https://jakarta.ee/xml/ns/jakartaee https://jakarta.ee/xml/ns/jakartaee/web-app_6_0.xsd">

    <resource-ref>
        <description>DB Connection</description>
        <res-ref-name>jdbc/incidencias</res-ref-name>
        <res-type>javax.sql.DataSource</res-type>
        <res-auth>Container</res-auth>
    </resource-ref>

</web-app>
```

1.4 Código de ejemplo

Para acceder dentro de nuestra aplicación a la fuente de datos:

```
Connection con = null; // Variable para la conexión.
PreparedStatement st = null; // Preparacion de la consulta
ResultSet rs = null; // Gestor del resultado de la consulta.

try {
        Context ctx = new InitialContext(); // Obtencion del contexto.
        DataSource    datasource = (DataSource) ctx.lookup("java:comp/env/jdbc/incidencias"); // Recuperación de la fuente de datos.
        con = datasource.getConnection(); // Apertura de la conexión

        st = (PreparedStatement) con.prepareStatement("CONSULTA A EJECUTAR"); // Indicamos la consulta a ejecutar.
        st.setString(x, "valor"); // Rellenamos los valores de la consulta.
        …….

        rs = st.executeQuery();

        while (rs.next()) {
                // recorremos cada uno de los resultados obtenidos.
        }
 } catch (Exception ex) {
        Logger.getLogger(DaoCamioneros.class.getName()).log(Level.SEVERE, null, ex);
```

```
} finally {
        if (rs != null) {
        try {
            rs.close();
        } catch (SQLException ex) {
            Logger.getLogger(UtilDb.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
    rs = null;

    if (st != null) {
        try {
            st.close();
        } catch (SQLException ex) {
            Logger.getLogger(UtilDb.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
    st = null;
        if (con != null) {
            try {
                con.close();
            } catch (SQLException ex) {
                Logger.getLogger(Articulos.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    }
```