élogos

TECNOLOGIA JDBC

élogos

ÍNDICE

TECNOLOGÍA JDBC

1. Drivers de bases de datos

Un driver es un software intermediario que utilizan las aplicaciones para acceder a las bases de datos.

Las instrucciones JDBC de acceso a datos se ejecutan sobre el driver, encargándose este de traducirlas a las instrucciones nativas de la base de datos con la que se va a trabajar.

Gracias al uso de los driver, las aplicaciones pueden abstraerse del tipo de base de datos con el que se va a trabajar. El API utilizado (JDBC) es universal.

Tipos de drivers

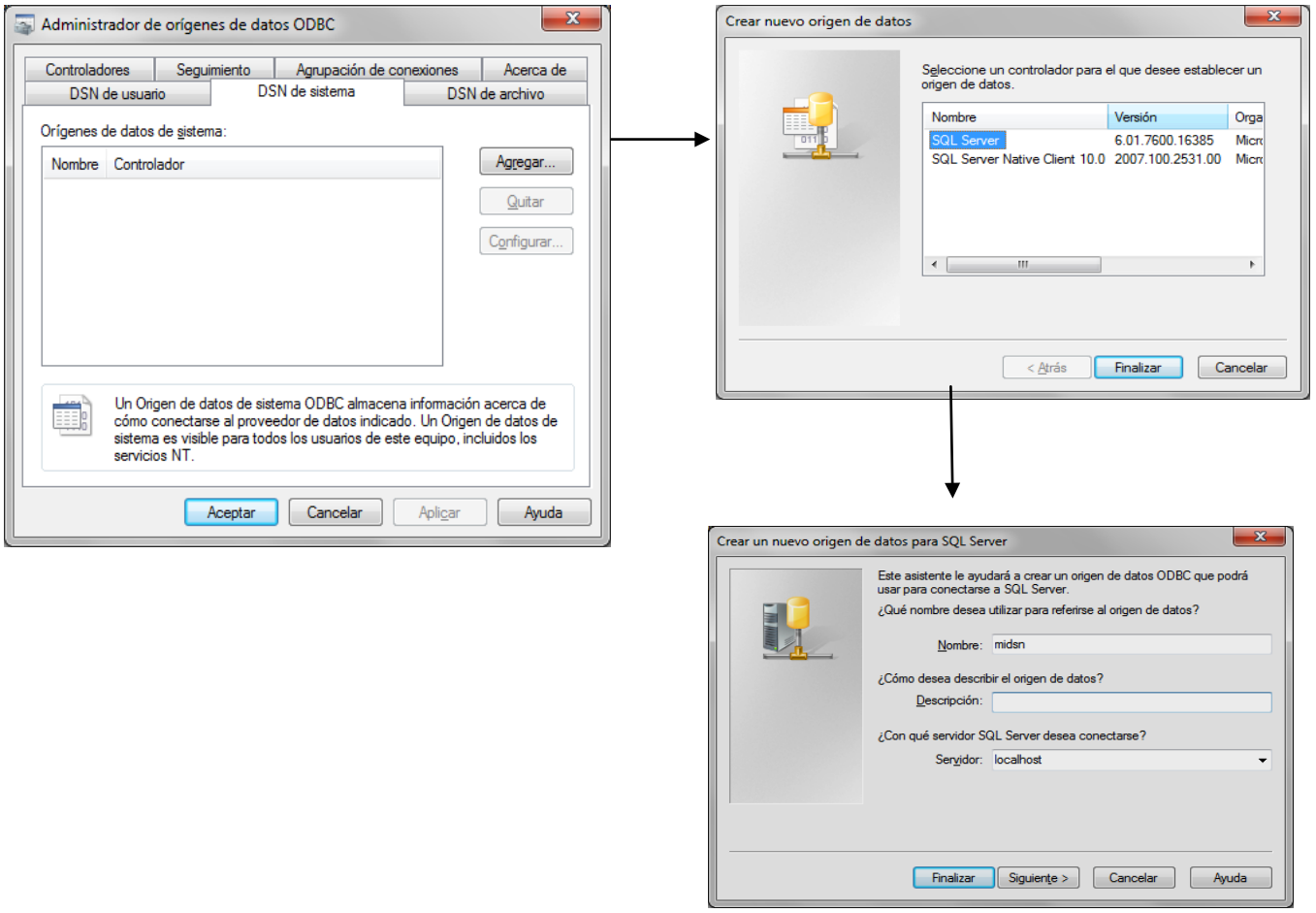
El concepto de driver JDBC no es exclusivo de Java, otras tecnologías utilizan también drivers para acceder a base de datos, es el caso de los drivers ODBC de Microsoft.

En el caso de Java, tenemos cuatro tipos de drivers

1. JDBC – ODBC bridge driver
2. Native API Partly Java driver
3. Native Protocol pure Java driver
4. JDBC- Net pure Java driver

Configuración de driver ODBC

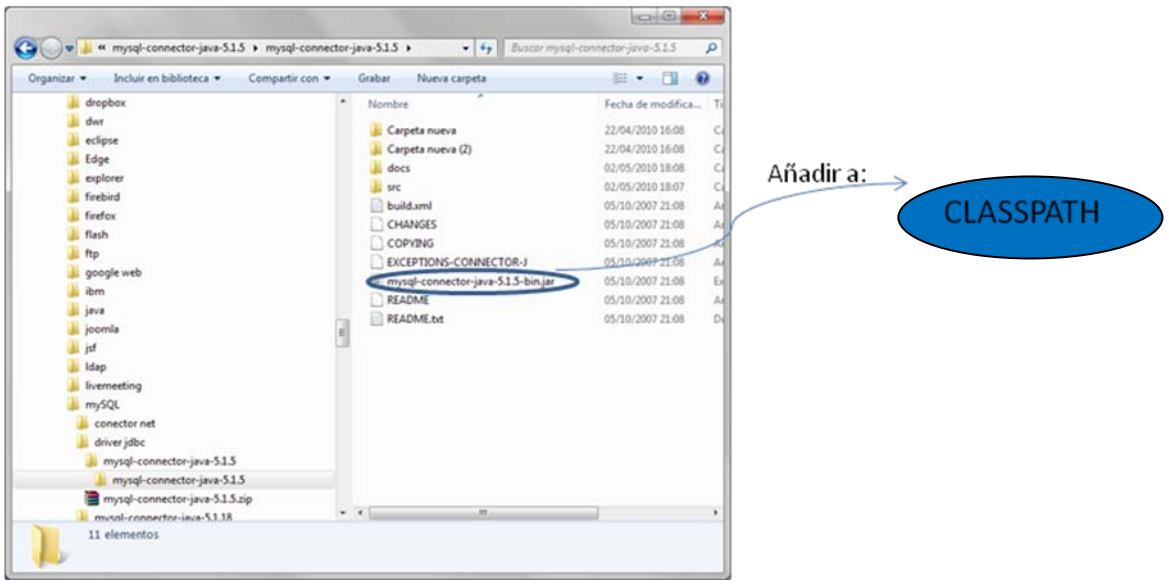
Si queremos utilizar el driver puente JDBC-ODBC, debemos configurar en nuestra máquina el driver ODBC. Esto requiere crear un DSN (Data Source Name).
Un DSN representa un nombre asociado a una base de datos y se crea a través del panel de control e Windows:



Utilización de driver nativo

Para utilizar un driver nativo no se requiere realizar ninguna configuración en la máquina donde se va a ejecutar la aplicación.

Los driver nativos se distribuyen en archivos de librería Java (jar). Lo único que tendremos que hacer será añadir la ruta del .jar a la dirección classpath de nuestra máquina. Esta operación no será necesaria si estamos utilizando un entorno de desarrollo y queremos probar la aplicación; bastará con añadir el archivo .jar a las referencias del proyecto.



2. El API JDBC

Interfaces y clases de JDBC

El API JDBC está formado por una serie de clases e interfaces dentro del paquete java.sql.
A continuación presentamos las clases e interfaces más importantes de este paquete y que serán objeto de estudio en esta lección:

- **DriverManager**. Establece conexiones con la base de datos a través del driver.
- **Connection.** Representa una conexión con la BD.
- **Statement.** Permite ejecutar consultas SQL sobre la base de datos.
- **PreparedStatement.** Utilización de consultas preparadas.
- **CallableStatement.** Ejecuta procedimientos almacenados.
- **ResultSet.** Manipula conjunto de registros.
- **ResultSetMetadata.** Proporciona información sobre la estructura de los datos (metadatos).

"Todos los métodos de las clases e interfaces JDBC pueden lanzar la excepción SQLException que habrá que capturar"

Clases para establecer la conexión con la base de datos

Antes de comenzar a utilizar las clases/interfaces de JDBC, lo primero será cargar en memoria el driver JDBC que vamos a utilizar.

Para ello, utilizaremos el método estático forName de la clase Class:

```
static Class <Class>.forName (String className)
        throws ClassNotFoundException

Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");
```

Lo que hace esta función es registrar el driver en un subsistema de la JVM denominado DriverManager.

Interface Connection

Obtener una conexión a la base de datos. Están representadas por la interface Connection:

Connection

Access Connection **Oracle Connection**

Normalmente se tendrá una referencia del interface Connection para acceder a los objetos que deriven. Para obtener el método Connection se utiliza el siguiente método:

```
static Connection <DriverManager> getConnection
        (String url) throws SQLException
```

El parámetro url determina la cadena de conexión que indica qué driver vamos a utilizar y qué parámetros se les va a pasar. La cadena de conexión es un texto con tres partes:

```
"jdbc:<driver>:<parametros>"

Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");
Connection conexion = DriverManager.getConnection
        ("jdbc:odbc:DSNBanco");
```

En el caso de acceder con un driver nativo a una base de datos de MySQL será:

```
Class.forName ("com.mysql.jdbc.Driver");

Connection cn=DriverManager.getConnection (“jdbc: mysql: //localhost:3306/libros","root","root");
```