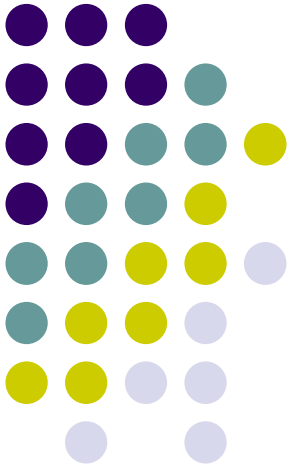
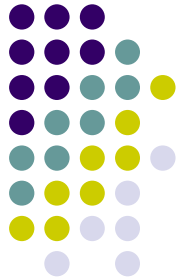
JDBC

Conectando Java al mundo de las Base de Datos



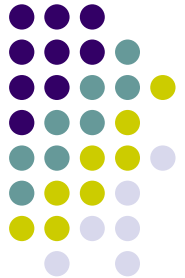
Lic. Claudio Zamoszczyk
claudio@honou.com.ar

Contenidos



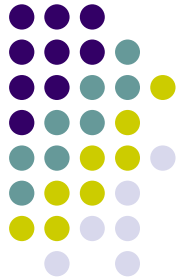
- ¿Qué es JDBC?
- ¿Por qué usar JDBC?
- Estructura del API JDBC
- ¿Cómo usar JDBC?
- Buenas practicas

¿Qué es JDBC?



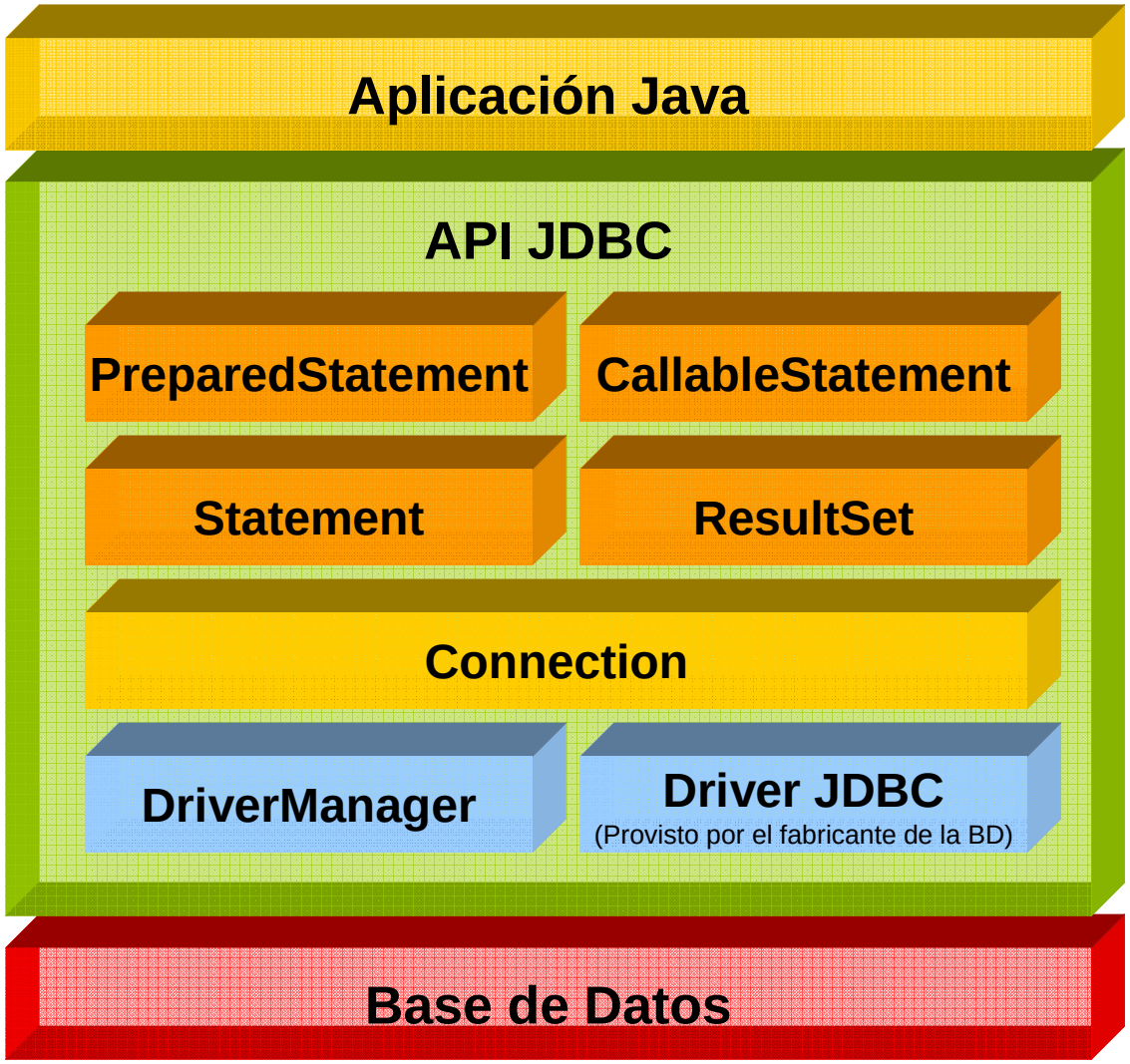
- **JDBC es una API, formada por conjunto de clases e interfaces, que permiten ejecutar sentencias SQL sobre una base de datos (externa) utilizando Java.**
- **JDBC = Java Database Connectivity**

¿Por qué usar JDBC?



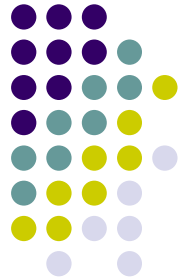
- **Ofrece un estándar de conexión a cualquier base de datos disponible en el mercado (Oracle, MySql, DB2, Sql Server, Informix, etc.).**
- **Permite establecer una conexión a una base de datos de manera simple.**
- **Permite enviar y ejecutar sentencias SQL (Select, Update, Insert, etc).**
- **Permite procesar los resultados de estas sentencias.**
- **Permite ejecutar Store Procedures**

Estructura básica del API JDBC



¿Cómo usar JDBC?

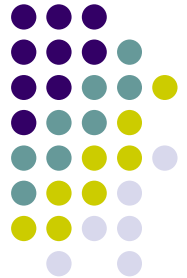
Drivers



- Los Driver JDBC son los componentes que permite conectar a las diferentes BD con Java.
- Cada fabricante es el encargado de proveer el driver correspondiente, y este debe ser compatible con la especificación JDBC.
- Existen 4 tipos de Drivers JDBC.

¿Cómo usar JDBC?

Cargando Drivers (Clas.for(…..))



- **Es necesario primero cargar una clase con el driver de la base de datos (esto lo provee el fabricante de la DB)**
- **Ejemplo:**

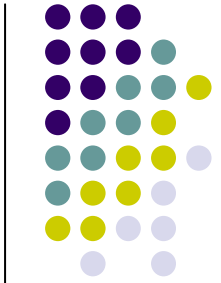
  Class.forName("com.informix.jdbc.IfxDriver");

  Calss.forName("com.novell.sql.LDAPDriver");

  Class.forName("com.mysql.jdbc.Driver");
- **Esto es particular según la base de datos que se usa**

¿Cómo usar JDBC?

Estableciendo la Conexión (Connection)



```
Connection con = DriverManager.getConnection (
    url,“usuario", "password");
```

- **Url de conexión**

jdbc:mysql://localhost/test

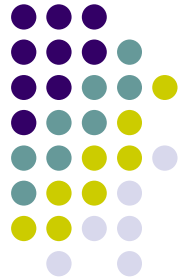
jdbc:oracle://oraserver/db

jdbc:odbc:mydatabase

jdbc:vendor//servidor:puerto/bd…

¿Cómo usar JDBC?

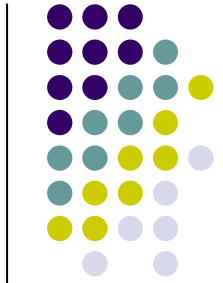
Ejecutando sentencias SQL



- **JDBC permite enviar cualquier tipo de sentencia SQL. Aunque ésta fuera dependiente de la base de datos sólo se correría el riesgo de incompatibilidad al cambiar de base de datos.**

¿Cómo usar JDBC?

Ejecutando sentencias SQL (Statement)



- **Statement stmt = conexion.createStatement();**

- **Métodos de Statement:**

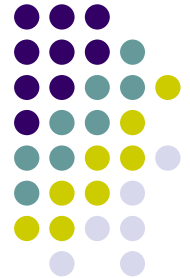
- stmt.execute(String), usado para ejecutar cualquier tipo de comando. Retorna true o false.

- stmt.executeUpdate(String), usado para crear/modificar tablas, típicamente para create, update, delete. Retorna el numero de fila afectadas.

- stmt.executeQuery(String) pusado para hacer consultas, retorna un cojunto de filas representadas en un objeto de la clase ResultSet, típicamente para select.

¿Cómo usar JDBC?

Ejecutando consultas SQL (ResultSet)

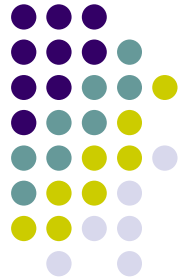


- ResultSet rs = stmt.executeQuery("select * from Personas");

```
while (rs.next()) {
    String s = rs.getString("nombre");
    int y = rs.getInt("dni");
    System.out.println(s + "   "+ y );
}
```

¿Cómo usar JDBC?

Ejecutando sentencias precompiladas SQL (PrepareStatement)



- **Donde se ha usado Satement es generalmente posible usar PreparedStatement para hacer más eficientes las consultas.**
- **Una instrucción con PreparedStatement va a ser, en la mayoría de los casos, traducida a una consulta SQL nativa de la base de datos en tiempo de compilación.**
- **La otra ventaja es que es posible usar parámetros dentro de ella, pudiendo hacer más flexibles las consultas.**

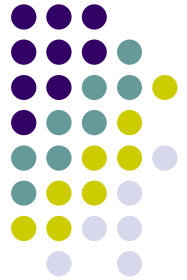
- **PreparedStatement us = con.prepareSatatement(“update alumnos set nombre = ? where direccion like = ?”);**

  us.setString(1,”Juan Carlos”)

  us.setString(2,”Callao 9876541236”);

¿Cómo usar JDBC?

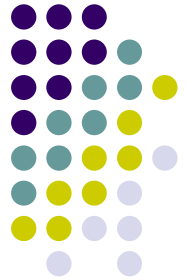
Manejo de transacciones



- **Una transacción consiste en una o más sentencias que han sido ejecutadas y luego confirmadas (commit) o deshechas (rolled back)**
- **Auto-commit está preseteado.**
- **Si Auto-commit está desactivado se debe usar los métodos “commit” o “rollback” explícitamente.**

¿Cómo usar JDBC?

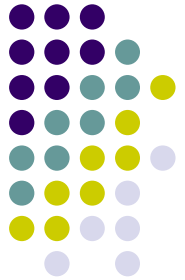
Manejo de transacciones



- **Para hacer uso de transacciones debe primero etearse el auto-commit a false**

```
con.setAutoCommit(false)
PreparedStatement ps = .....
....
ps.executeUpdate()  ....
ps.excecuteUpdate() ...
con.commit();
```

Buenas Practicas



- **Siempre utilizar ANSI Sql para minimizar los problemas que puedan surgir al cambiar de base de datos.**
- **Utilizar siempre una clase que gestione todo el acceso a la BD, separando lo que es lógica del negocio con lógica de acceso a datos.**
- **Crea una conexión consume muchos recursos, siempre que sea posible utilizar un Pool de conexiones.**

¿Preguntas?