Programación 2

Java Database Connectivity (*JDBC*)

Eduardo Godoy
Ingeniero en Informática
eduardo.gl@gmail.com

1^{er} Semestre de 2017

Contenido

Contenido

Contenido

JDBC

Introducción

Introducción

- Aplicación que procese información **base de datos**.
- Normalmente se utilizan bases de datos relacionales.
- SQL: Lenguaje estándar para acceder a una base datos.

JDBC

Introducción

Introducción

- Aplicación que procese información **base de datos**.
- Normalmente se utilizan bases de datos relacionales.
- SQL: Lenguaje estándar para acceder a una base datos.

JDBC

Introducción

Introducción

- Aplicación que procese información **base de datos**.
- Normalmente se utilizan bases de datos relacionales.
- SQL: Lenguaje estándar para acceder a una base datos.

JDBC

Introducción

Introducción

- Aplicación que procese información **base de datos**.
- Normalmente se utilizan bases de datos relacionales.
- SQL: Lenguaje estándar para acceder a una base datos.

Contenido

JDBC

Características de JDBC

Características de JDBC

- Es un interfaz orientado a objetos de Java para SQL.
- Se utiliza para enviar sentencias SQL a un sistema gestor de BD (DBMS).
- Con JDBC tenemos que continuar escribiendo las sentencias SQL.

JDBC

Características de JDBC

Características de JDBC

- Es un interfaz orientado a objetos de Java para SQL.
- Se utiliza para enviar sentencias SQL a un sistema gestor de BD (DBMS).
- Con JDBC tenemos que continuar escribiendo las sentencias SQL.

JDBC

Características de JDBC

Características de JDBC

- Es un interfaz orientado a objetos de Java para SQL.
- Se utiliza para enviar sentencias SQL a un sistema gestor de BD (DBMS).
- Con JDBC tenemos que continuar escribiendo las sentencias SQL.

JDBC

Características de JDBC

Características de JDBC

- Es un interfaz orientado a objetos de Java para SQL.
- Se utiliza para enviar sentencias SQL a un sistema gestor de BD (DBMS).
- Con JDBC tenemos que continuar escribiendo las sentencias SQL.

JDBC

Características de JDBC

Características de JDBC

- La filosofía de JDBC es proporcionar transparencia al desarrollador frente al gestor de BD.
- JDBC utiliza un Gestor de Controladores o **DriverManager** que hace de interfaz con el controlador específico de la BD.

JDBC

Características de JDBC

Características de JDBC

- La filosofía de JDBC es proporcionar transparencia al desarrollador frente al gestor de BD.
- JDBC utiliza un Gestor de Controladores o **DriverManager** que hace de interfaz con el controlador específico de la BD.

JDBC

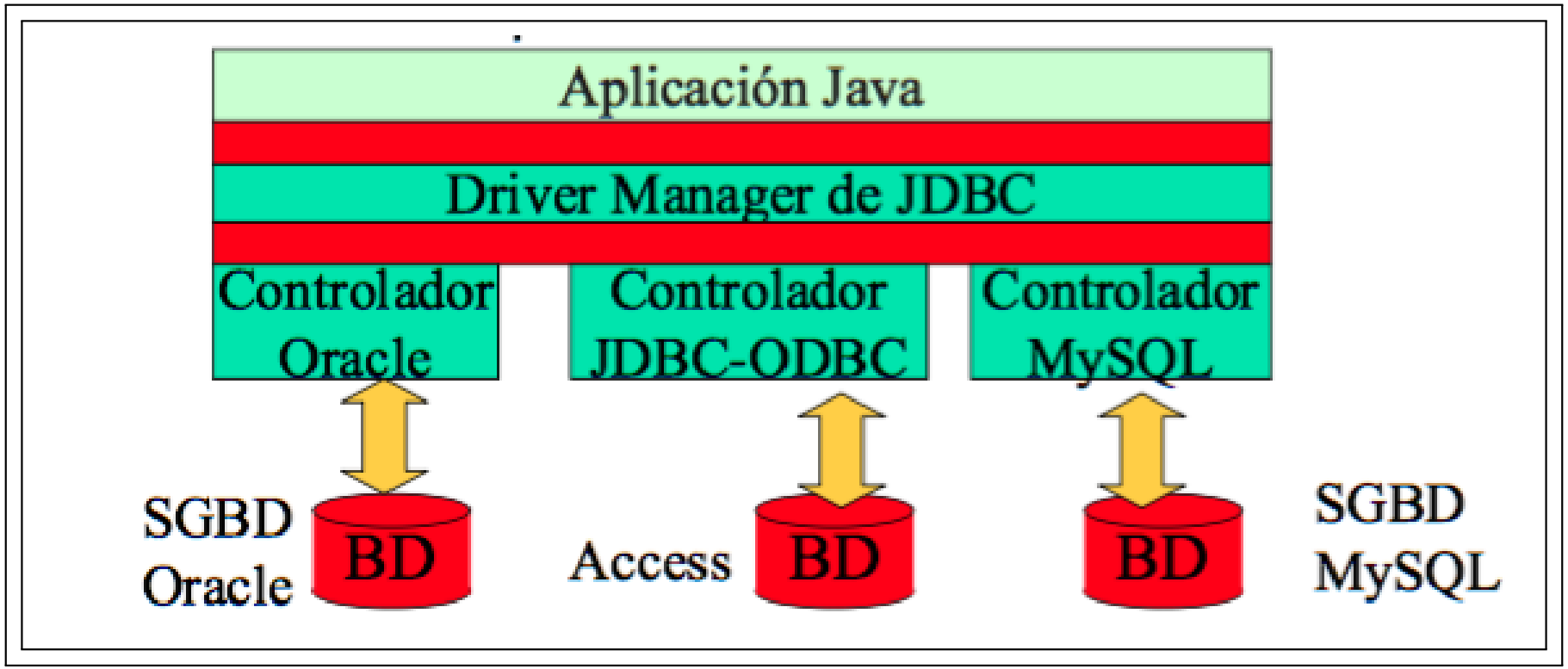
Características de JDBC

Características de JDBC

- La filosofía de JDBC es proporcionar transparencia al desarrollador frente al gestor de BD.
- JDBC utiliza un Gestor de Controladores o **DriverManager** que hace de interfaz con el controlador específico de la BD.

JDBC

Características de JDBC



JDBC

Características de JDBC

Características de JDBC

- La especificación JDBC incluye 8 interfaces y 10 clases, en el paquete estándar **java.sql**.
- Podemos dividirlos en los siguientes grupos: Núcleo de JDBC, interfaces y clases que todos los controladores deben implementar.
  - Extensiones al paquete **java.lang**, extensiones para SQL.
  - Extensiones al paquete **java.util**, son extensiones a **java.util.Date**.
  - Metadatos para SQL, permiten examinar dinámicamente las propiedades de BD y controladores.

JDBC

Características de JDBC

Características de JDBC

- La especificación JDBC incluye 8 interfaces y 10 clases, en el paquete estándar **java.sql**.
- Podemos dividirlos en los siguientes grupos: Núcleo de JDBC, interfaces y clases que todos los controladores deben implementar.
  - Extensiones al paquete **java.lang**, extensiones para SQL.
  - Extensiones al paquete **java.util**, son extensiones a **java.util.Date**.
  - Metadatos para SQL, permiten examinar dinámicamente las propiedades de BD y controladores.

JDBC

Características de JDBC

Características de JDBC

- La especificación JDBC incluye 8 interfaces y 10 clases, en el paquete estándar **java.sql**.
- Podemos dividirlos en los siguientes grupos: Núcleo de JDBC, interfaces y clases que todos los controladores deben implementar.
  - Extensiones al paquete **java.lang**, extensiones para SQL.
  - Extensiones al paquete **java.util**, son extensiones a **java.util.Date**.
  - Metadatos para SQL, permiten examinar dinámicamente las propiedades de BD y controladores.

JDBC

Características de JDBC

Características de JDBC

- La especificación JDBC incluye 8 interfaces y 10 clases, en el paquete estándar **java.sql**.
- Podemos dividirlos en los siguientes grupos: Núcleo de JDBC, interfaces y clases que todos los controladores deben implementar.
  - Extensiones al paquete **java.lang**, extensiones para SQL.
  - Extensiones al paquete **java.util**, son extensiones a **java.util.Date**.
  - Metadatos para SQL, permiten examinar dinámicamente las propiedades de BD y controladores.

JDBC

Características de JDBC

Características de JDBC

- La especificación JDBC incluye 8 interfaces y 10 clases, en el paquete estándar **java.sql**.
- Podemos dividirlos en los siguientes grupos: Núcleo de JDBC, interfaces y clases que todos los controladores deben implementar.
  - Extensiones al paquete **java.lang**, extensiones para SQL.
  - Extensiones al paquete **java.util**, son extensiones a **java.util.Date**.
  - Metadatos para SQL, permiten examinar dinámicamente las propiedades de BD y controladores.

JDBC

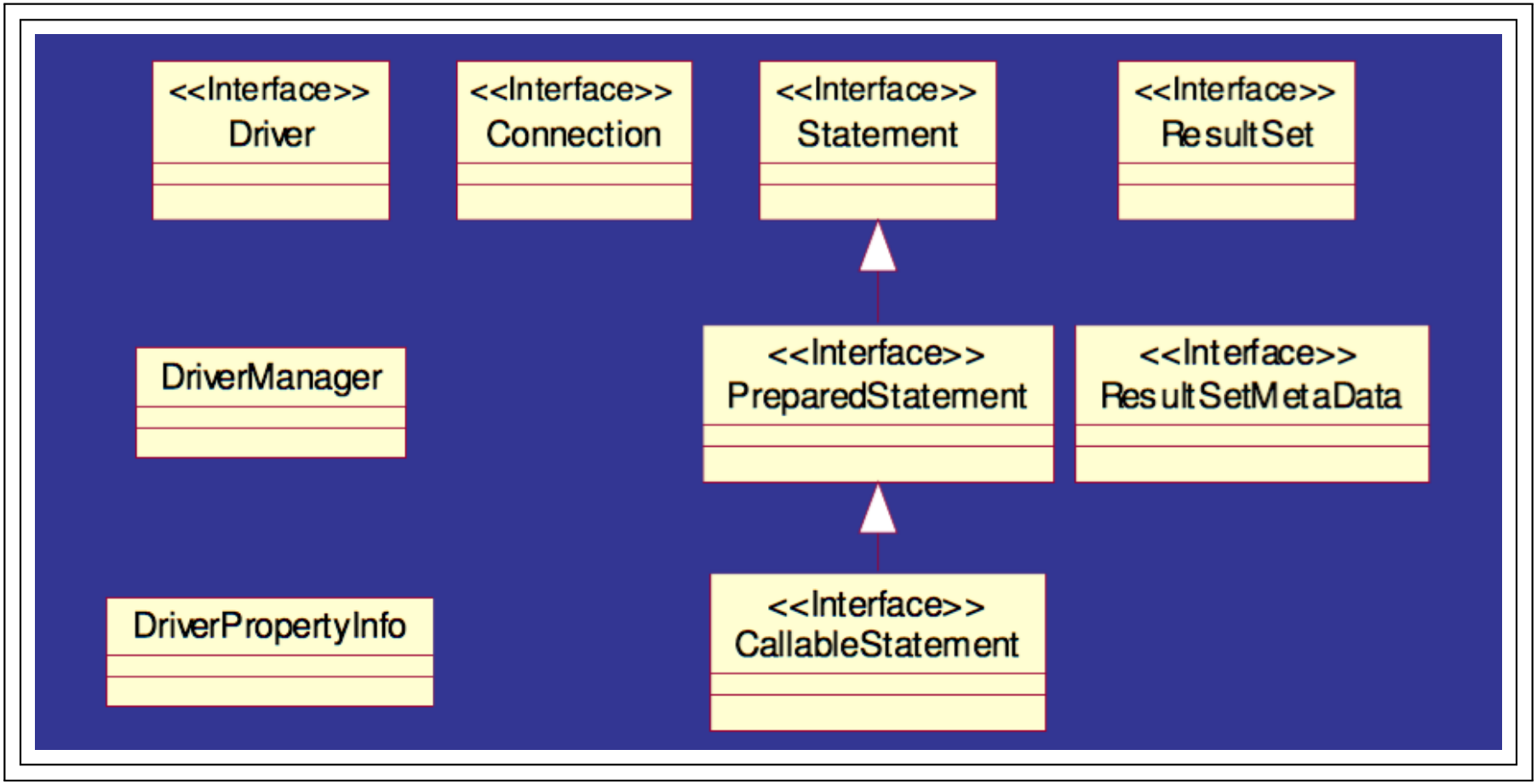
Características de JDBC

Características de JDBC

- La especificación JDBC incluye 8 interfaces y 10 clases, en el paquete estándar **java.sql**.
- Podemos dividirlos en los siguientes grupos: Núcleo de JDBC, interfaces y clases que todos los controladores deben implementar.
  - Extensiones al paquete **java.lang**, extensiones para SQL.
  - Extensiones al paquete **java.util**, son extensiones a **java.util.Date**.
  - Metadatos para SQL, permiten examinar dinámicamente las propiedades de BD y controladores.

JDBC

Características de JDBC



Contenido

JDBC

Pasos para una conexión JDBC - Primer paso

Pasos para una conexión JDBC - Primer paso

- Antes de conectar con la base de datos, debemos considerar dos aspectos:
  - Registrar un controlador.
  - Convenciones de nombres para la base de datos.

JDBC

Pasos para una conexión JDBC - Primer paso

Pasos para una conexión JDBC - Primer paso

- Antes de conectar con la base de datos, debemos considerar dos aspectos:
  - Registrar un controlador.
  - Convenciones de nombres para la base de datos.

JDBC

Pasos para una conexión JDBC - Primer paso

Pasos para una conexión JDBC - Primer paso

- Antes de conectar con la base de datos, debemos considerar dos aspectos:
  - Registrar un controlador.
  - Convenciones de nombres para la base de datos.

JDBC

Pasos para una conexión JDBC - Primer paso

Pasos para una conexión JDBC - Primer paso

- Antes de conectar con la base de datos, debemos considerar dos aspectos:
  - Registrar un controlador.
  - Convenciones de nombres para la base de datos.

JDBC

Pasos para una conexión JDBC - Primer paso

Registrar un controlador

- Determinados controladores requerirán la instalación y configuración de software específico en el cliente. Ejemplo: el origen ODBC o la fuente de datos nativa.
- Otros controladores son simplemente una clase Java y bastará que la **Java Virtual Machine** las pueda localizar mediante el **classpath**.

JDBC

Pasos para una conexión JDBC - Primer paso

Registrar un controlador

- Determinados controladores requerirán la instalación y configuración de software específico en el cliente. Ejemplo: el origen ODBC o la fuente de datos nativa.
- Otros controladores son simplemente una clase Java y bastará que la **Java Virtual Machine** las pueda localizar mediante el **classpath**.

JDBC

Pasos para una conexión JDBC - Primer paso

Registrar un controlador

- Determinados controladores requerirán la instalación y configuración de software específico en el cliente. Ejemplo: el origen ODBC o la fuente de datos nativa.
- Otros controladores son simplemente una clase Java y bastará que la **Java Virtual Machine** las pueda localizar mediante el **classpath**.

JDBC

Pasos para una conexión JDBC - Primer paso

Registrar un controlador

- Para que la JVM pueda localizar una clase de driver, se debe:
  - Situar la ruta a la clase en el *CLASSPATH*.
  - **Añadir un JAR externo en el proyecto donde está nuestra aplicación.**
  - Configurar una asociación o pool de conexiones (ambientes web).

JDBC

Pasos para una conexión JDBC - Primer paso

Registrar un controlador

- Para que la JVM pueda localizar una clase de driver, se debe:
  - Situar la ruta a la clase en el *CLASSPATH*.
  - **Añadir un JAR externo en el proyecto donde está nuestra aplicación.**
  - Configurar una asociación o pool de conexiones (ambientes web).

JDBC

Pasos para una conexión JDBC - Primer paso

Registrar un controlador

- Para que la JVM pueda localizar una clase de driver, se debe:
  - Situar la ruta a la clase en el *CLASSPATH*.
  - **Añadir un JAR externo en el proyecto donde está nuestra aplicación.**
  - Configurar una asociación o pool de conexiones (ambientes web).

JDBC

Pasos para una conexión JDBC - Primer paso

Registrar un controlador

- Para que la JVM pueda localizar una clase de driver, se debe:
  - Situar la ruta a la clase en el *CLASSPATH*.
  - **Añadir un JAR externo en el proyecto donde está nuestra aplicación.**
  - Configurar una asociación o pool de conexiones (ambientes web).

JDBC

Pasos para una conexión JDBC - Primer paso

Registrar un controlador

- Para que la JVM pueda localizar una clase de driver, se debe:
  - Situar la ruta a la clase en el *CLASSPATH*.
  - **Añadir un JAR externo en el proyecto donde está nuestra aplicación.**
  - Configurar una asociación o pool de conexiones (ambientes web).

JDBC

Pasos para una conexión JDBC - Primer paso

Registrar un controlador

- Averiguar el nombre de la clase provista por el paquete a ser usado.
- Localizar la clase en el directorio apropiado para que pueda ser cargado.
- Alternativamente se puede cargar la clase en el programa usando el comando:
  - Class.forName("sun.jdbcJdbcOdbcDriver")
  - DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver()).

JDBC

Pasos para una conexión JDBC - Primer paso

Registrar un controlador

- Averiguar el nombre de la clase provista por el paquete a ser usado.
- Localizar la clase en el directorio apropiado para que pueda ser cargado.
- Alternativamente se puede cargar la clase en el programa usando el comando:
  - Class.forName("sun.jdbcJdbcOdbcDriver")
  - DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver()).

JDBC

Pasos para una conexión JDBC - Primer paso

Registrar un controlador

- Averiguar el nombre de la clase provista por el paquete a ser usado.
- Localizar la clase en el directorio apropiado para que pueda ser cargado.
- Alternativamente se puede cargar la clase en el programa usando el comando:
  - Class.forName("sun.jdbcJdbcOdbcDriver")
  - DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver()).

JDBC

Pasos para una conexión JDBC - Primer paso

Registrar un controlador

- Averiguar el nombre de la clase provista por el paquete a ser usado.
- Localizar la clase en el directorio apropiado para que pueda ser cargado.
- Alternativamente se puede cargar la clase en el programa usando el comando:
  - Class.forName("sun.jdbcJdbcOdbcDriver")
  - DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver()).

JDBC

Pasos para una conexión JDBC - Primer paso

Registrar un controlador

- Averiguar el nombre de la clase provista por el paquete a ser usado.
- Localizar la clase en el directorio apropiado para que pueda ser cargado.
- Alternativamente se puede cargar la clase en el programa usando el comando:
  - **Class.forName("sun.jdbcJdbcOdbcDriver")**
  - DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver()).

JDBC

Pasos para una conexión JDBC - Primer paso

Registrar un controlador

- Averiguar el nombre de la clase provista por el paquete a ser usado.
- Localizar la clase en el directorio apropiado para que pueda ser cargado.
- Alternativamente se puede cargar la clase en el programa usando el comando:
  - **Class.forName("sun.jdbcJdbcOdbcDriver")**
  - **DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver())**.

JDBC

Pasos para una conexión JDBC - Primer paso

```
try {
    //Ejemplo para Oracle
    DriverManager.registerDriver(new oracle.jdbc.driver.
    Oracledriver());
    //Ejemplo para PostgreSQL
    Class.forName("org.postgresql.Driver");
    //Ejemplo para MySQL
    Class.forName("com.mysql.jdbc.Driver");
} catch (ClassNotFoundException ex) {
    System.out.println("Error en la carga del driver JDBC");
}
```

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- La clase **DriverManager** es la responsable de:
  - Seleccionar el driver.
  - Crear una nueva conexión a la base de datos.
- Previamente habremos registrado el controlador en el **DriverManager**.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- La clase **DriverManager** es la responsable de:
  - Seleccionar el driver.
  - Crear una nueva conexión a la base de datos.
- Previamente habremos registrado el controlador en el **DriverManager**.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- La clase **DriverManager** es la responsable de:
  - Seleccionar el driver.
  - Crear una nueva conexión a la base de datos.
- Previamente habremos registrado el controlador en el **DriverManager**.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- La clase **DriverManager** es la responsable de:
  - Seleccionar el driver.
  - Crear una nueva conexión a la base de datos.
- Previamente habremos registrado el controlador en el **DriverManager**.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- La clase **DriverManager** es la responsable de:
  - Seleccionar el driver.
  - Crear una nueva conexión a la base de datos.
- Previamente habremos registrado el controlador en el **DriverManager**.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- Crear una conexión:
  - Connection con = DriverManager.getConnection(url,user,pass)
- El parámetro imprescindible es la url de la base de datos; para especificar la base de datos que queremos utilizar.
- También se puede especificar el usuario y la clave con los que nos queremos conectar a la base de datos.
- El **DriverManager** buscará un driver, entre los registrados que pueda usar el protocolo especificado en la url.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- Crear una conexión:
  - **Connection con = DriverManager.getConnection(url,user,pass)**
- El parámetro imprescindible es la url de la base de datos; para especificar la base de datos que queremos utilizar.
- También se puede especificar el usuario y la clave con los que nos queremos conectar a la base de datos.
- El **DriverManager** buscará un driver, entre los registrados que pueda usar el protocolo especificado en la url.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- Crear una conexión:
  - **Connection con = DriverManager.getConnection(url,user,pass)**
- El parámetro imprescindible es la url de la base de datos; para especificar la base de datos que queremos utilizar.
- También se puede especificar el usuario y la clave con los que nos queremos conectar a la base de datos.
- El **DriverManager** buscará un driver, entre los registrados que pueda usar el protocolo especificado en la url.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- Crear una conexión:
  - **Connection con = DriverManager.getConnection(url,user,pass)**
- El parámetro imprescindible es la url de la base de datos; para especificar la base de datos que queremos utilizar.
- También se puede especificar el usuario y la clave con los que nos queremos conectar a la base de datos.
- El **DriverManager** buscará un driver, entre los registrados que pueda usar el protocolo especificado en la url.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- Crear una conexión:
  - **Connection con = DriverManager.getConnection(url,user,pass)**
- El parámetro imprescindible es la url de la base de datos; para especificar la base de datos que queremos utilizar.
- También se puede especificar el usuario y la clave con los que nos queremos conectar a la base de datos.
- El **DriverManager** buscará un driver, entre los registrados que pueda usar el protocolo especificado en la url.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- Crear una conexión:
  - **Connection con = DriverManager.getConnection(url,user,pass)**
- El parámetro imprescindible es la url de la base de datos; para especificar la base de datos que queremos utilizar.
- También se puede especificar el usuario y la clave con los que nos queremos conectar a la base de datos.
- El **DriverManager** buscará un driver, entre los registrados que pueda usar el protocolo especificado en la url.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- En JDBC las bases de datos se designan mediante una URL.
- La forma de la url es la siguiente: **jdbc:¡subprotocolo¿:¡dominio¿**
- **Subprotocolo**: identifica el mecanismo de conexión o el controlador JDBC concreto.
- **Dominio**: Depende del subprotocolo, puede ser simplemente un nombre simple para la base de datos o una serie de parámetros que permiten encontrar la BD.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- En JDBC las bases de datos se designan mediante una URL.
- La forma de la url es la siguiente: **jdbc:¡subprotocolo¿:¡dominio¿**
- **Subprotocolo**: identifica el mecanismo de conexión o el controlador JDBC concreto.
- **Dominio**: Depende del subprotocolo, puede ser simplemente un nombre simple para la base de datos o una serie de parámetros que permiten encontrar la BD.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- En JDBC las bases de datos se designan mediante una URL.
- La forma de la url es la siguiente: **jdbc:¡subprotocolo¿:¡dominio¿**
- **Subprotocolo**: identifica el mecanismo de conexión o el controlador JDBC concreto.
- **Dominio**: Depende del subprotocolo, puede ser simplemente un nombre simple para la base de datos o una serie de parámetros que permiten encontrar la BD.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- En JDBC las bases de datos se designan mediante una URL.
- La forma de la url es la siguiente: **jdbc:¡subprotocolo¿:¡dominio¿**
- **Subprotocolo**: identifica el mecanismo de conexión o el controlador JDBC concreto.
- **Dominio**: Depende del subprotocolo, puede ser simplemente un nombre simple para la base de datos o una serie de parámetros que permiten encontrar la BD.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- En JDBC las bases de datos se designan mediante una URL.
- La forma de la url es la siguiente: **jdbc:¡subprotocolo¿:¡dominio¿**
- **Subprotocolo**: identifica el mecanismo de conexión o el controlador JDBC concreto.
- **Dominio**: Depende del subprotocolo, puede ser simplemente un nombre simple para la base de datos o una serie de parámetros que permiten encontrar la BD.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- Si no se producen excepciones el método **getConnection** nos devuelve un objeto que implementa la interfaz **Connection**.
- Podemos crear varias conexiones con distintas bases de datos o incluso con la misma.
- Cada conexión representa una sesión con la BD.
- El objeto **Connection** nos permitirá acceder a la base de datos para realizar operaciones sobre ella y obtener resultados.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- Si no se producen excepciones el método **getConnection** nos devuelve un objeto que implementa la interfaz **Connection**.
- Podemos crear varias conexiones con distintas bases de datos o incluso con la misma.
- Cada conexión representa una sesión con la BD.
- El objeto **Connection** nos permitirá acceder a la base de datos para realizar operaciones sobre ella y obtener resultados.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- Si no se producen excepciones el método **getConnection** nos devuelve un objeto que implementa la interfaz **Connection**.
- Podemos crear varias conexiones con distintas bases de datos o incluso con la misma.
- Cada conexión representa una sesión con la BD.
- El objeto **Connection** nos permitirá acceder a la base de datos para realizar operaciones sobre ella y obtener resultados.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- Si no se producen excepciones el método **getConnection** nos devuelve un objeto que implementa la interfaz **Connection**.
- Podemos crear varias conexiones con distintas bases de datos o incluso con la misma.
- Cada conexión representa una sesión con la BD.
- El objeto **Connection** nos permitirá acceder a la base de datos para realizar operaciones sobre ella y obtener resultados.

JDBC

Pasos para una conexión JDBC - Segundo paso

Pasos para una conexión JDBC - Segundo paso

- Si no se producen excepciones el método **getConnection** nos devuelve un objeto que implementa la interfaz **Connection**.
- Podemos crear varias conexiones con distintas bases de datos o incluso con la misma.
- Cada conexión representa una sesión con la BD.
- El objeto **Connection** nos permitirá acceder a la base de datos para realizar operaciones sobre ella y obtener resultados.

JDBC

Pasos para una conexión JDBC - Segundo paso

```
try {
   //Oracle
   Connection con = DriverManager.getConnection("jdbc:oracle:
     thin:@localhost:1521:base_datos","user","password");
   //PostgreSQL
   Connection con = DriverManager.getConnection("jdbc:
     postgresql://localhost:5432/base_datos","user","password")
     ;
   //MySQL
   Connection con = DriverManager.getConnection("jdbc:mysql://
     localhost:3306/base_datos","user","password");
} catch (SQLException e) {
    System.out.println(e);
}
```

JDBC

Pasos para una conexión JDBC - Tercer paso

Pasos para una conexión JDBC - Tercer paso

- Utilizamos la conexión con la base de datos creada anteriormente para enviar comandos y sentencias SQL.
- El objeto conexión funciona como un enlace directo con el controlador de la BD.
- Creamos un objeto de la clase **Statement** que servirá de *envoltorio* para las sentencias SQL.
- Cuando pasamos la SQL a la conexión este lo envía al controlador que a su vez lo redirecciona a la BD, que nos devolverá los resultados.

JDBC

Pasos para una conexión JDBC - Tercer paso

Pasos para una conexión JDBC - Tercer paso

- Utilizamos la conexión con la base de datos creada anteriormente para enviar comandos y sentencias SQL.
- El objeto conexión funciona como un enlace directo con el controlador de la BD.
- Creamos un objeto de la clase **Statement** que servirá de *envoltorio* para las sentencias SQL.
- Cuando pasamos la SQL a la conexión este lo envía al controlador que a su vez lo redirecciona a la BD, que nos devolverá los resultados.

JDBC

Pasos para una conexión JDBC - Tercer paso

Pasos para una conexión JDBC - Tercer paso

- Utilizamos la conexión con la base de datos creada anteriormente para enviar comandos y sentencias SQL.
- El objeto conexión funciona como un enlace directo con el controlador de la BD.
- Creamos un objeto de la clase **Statement** que servirá de *envoltorio* para las sentencias SQL.
- Cuando pasamos la SQL a la conexión este lo envía al controlador que a su vez lo redirecciona a la BD, que nos devolverá los resultados.

JDBC

Pasos para una conexión JDBC - Tercer paso

Pasos para una conexión JDBC - Tercer paso

- Utilizamos la conexión con la base de datos creada anteriormente para enviar comandos y sentencias SQL.
- El objeto conexión funciona como un enlace directo con el controlador de la BD.
- Creamos un objeto de la clase **Statement** que servirá de *envoltorio* para las sentencias SQL.
- Cuando pasamos la SQL a la conexión este lo envía al controlador que a su vez lo redirecciona a la BD, que nos devolverá los resultados.

JDBC

Pasos para una conexión JDBC - Tercer paso

Pasos para una conexión JDBC - Tercer paso

- Utilizamos la conexión con la base de datos creada anteriormente para enviar comandos y sentencias SQL.
- El objeto conexión funciona como un enlace directo con el controlador de la BD.
- Creamos un objeto de la clase **Statement** que servirá de *envoltorio* para las sentencias SQL.
- Cuando pasamos la SQL a la conexión este lo envía al controlador que a su vez lo redirecciona a la BD, que nos devolverá los resultados.

JDBC

Pasos para una conexión JDBC - Tercer paso

Pasos para una conexión JDBC - Tercer paso

- La clase **Connection** tiene varios métodos que permiten crear un objeto **Statement** o una de sus variantes.
- **createStatement**, para crear sentencias simples.
- prepareStatement, para sentencias que pueden contener parámetros, optimiza la utilización repetida de estas sentencias.
- prepareCall, para funciones o procedimientos almancenados.

JDBC

Pasos para una conexión JDBC - Tercer paso

Pasos para una conexión JDBC - Tercer paso

- La clase **Connection** tiene varios métodos que permiten crear un objeto **Statement** o una de sus variantes.
- **createStatement**, para crear sentencias simples.
- prepareStatement, para sentencias que pueden contener parámetros, optimiza la utilización repetida de estas sentencias.
- prepareCall, para funciones o procedimientos almancenados.

JDBC

Pasos para una conexión JDBC - Tercer paso

Pasos para una conexión JDBC - Tercer paso

- La clase **Connection** tiene varios métodos que permiten crear un objeto **Statement** o una de sus variantes.
- **createStatement**, para crear sentencias simples.
- prepareStatement, para sentencias que pueden contener parámetros, optimiza la utilización repetida de estas sentencias.
- prepareCall, para funciones o procedimientos almancenados.

JDBC

Pasos para una conexión JDBC - Tercer paso

Pasos para una conexión JDBC - Tercer paso

- La clase **Connection** tiene varios métodos que permiten crear un objeto **Statement** o una de sus variantes.
- **createStatement**, para crear sentencias simples.
- prepareStatement, para sentencias que pueden contener parámetros, optimiza la utilización repetida de estas sentencias.
- prepareCall, para funciones o procedimientos almancenados.

JDBC

Pasos para una conexión JDBC - Tercer paso

Pasos para una conexión JDBC - Tercer paso

- La clase **Connection** tiene varios métodos que permiten crear un objeto **Statement** o una de sus variantes.
- **createStatement**, para crear sentencias simples.
- prepareStatement, para sentencias que pueden contener parámetros, optimiza la utilización repetida de estas sentencias.
- prepareCall, para funciones o procedimientos almancenados.

JDBC

Pasos para una conexión JDBC - Tercer paso

```
try {
   // Creamos el objeto sentencia
   Statement stmt = con.createStatement();
   // ...
} catch (Exception e) {
   System.out.println(e);
}
```

JDBC

Pasos para una conexión JDBC - Cuarto paso

Pasos para una conexión JDBC - Cuarto paso

- Ejecución
  - **executeQuery**(), ejecución de consultas, sentencia SELECT.
  - **executeUpdate**(), actualizaciones de valores en al base de datos. **INSERT**, **UPDATE**, **DELETE**. Sólo devuelve la cuenta de las columnas afectadas.
  - **execute**(), se usa para ejecutar sentencias que no se conocen a priori o que devuelven resultados no homogéneos.

JDBC

Pasos para una conexión JDBC - Cuarto paso

Pasos para una conexión JDBC - Cuarto paso

- Ejecución
  - **executeQuery**(), ejecución de consultas, sentencia SELECT.
  - **executeUpdate**(), actualizaciones de valores en al base de datos. **INSERT, UPDATE, DELETE**. Sólo devuelve la cuenta de las columnas afectadas.
  - **execute**(), se usa para ejecutar sentencias que no se conocen a priori o que devuelven resultados no homogéneos.

JDBC

Pasos para una conexión JDBC - Cuarto paso

Pasos para una conexión JDBC - Cuarto paso

- Ejecución
  - **executeQuery**(), ejecución de consultas, sentencia SELECT.
  - **executeUpdate**(), actualizaciones de valores en al base de datos. **INSERT, UPDATE, DELETE**. Sólo devuelve la cuenta de las columnas afectadas.
  - **execute**(), se usa para ejecutar sentencias que no se conocen a priori o que devuelven resultados no homogéneos.

JDBC

Pasos para una conexión JDBC - Cuarto paso

Pasos para una conexión JDBC - Cuarto paso

- Ejecución
  - **executeQuery**(), ejecución de consultas, sentencia SELECT.
  - **executeUpdate**(), actualizaciones de valores en al base de datos. **INSERT, UPDATE, DELETE**. Sólo devuelve la cuenta de las columnas afectadas.
  - **execute**(), se usa para ejecutar sentencias que no se conocen a priori o que devuelven resultados no homogéneos.

JDBC

Pasos para una conexión JDBC - Cuarto paso

Pasos para una conexión JDBC - Cuarto paso

- Ejecución
  - **executeQuery**(), ejecución de consultas, sentencia SELECT.
  - **executeUpdate**(), actualizaciones de valores en al base de datos. **INSERT, UPDATE, DELETE**. Sólo devuelve la cuenta de las columnas afectadas.
  - **execute**(), se usa para ejecutar sentencias que no se conocen a priori o que devuelven resultados no homogéneos.

JDBC

Pasos para una conexión JDBC - Cuarto paso

```
try {
   // ...
   // Ejecutamos una sentencia SQL
   ResultSet rs = stmt.executeQuery("SELECT id, nombre, email,
      telefono FROM Contactos WHERE nombre LIKE '%pepe%'");
} catch (SQLException e) {
   System.out.prinln(e);
}
```

JDBC

Pasos para una conexión JDBC - Quinto paso

Pasos para una conexión JDBC - Quinto paso

- Recuperación
  - Cuando ejecutamos una consulta debemos emplear el método **executeQuery()**, que devuelve un objeto **ResultSet**, que nos permitirá acceder a los resultados.
  - El **ResultSet** utiliza el concepto de cursor de base de datos para ir moviéndose por las filas de datos recuperadas.
  - Las columnas pueden accederse en cualquier orden, utilizando su posición o su nombre.
  - El objeto **ResultSet** incluye métodos getXXX que permiten recuperar valores de distintos tipos.

JDBC

Pasos para una conexión JDBC - Quinto paso

Pasos para una conexión JDBC - Quinto paso

- Recuperación
  - Cuando ejecutamos una consulta debemos emplear el método **executeQuery**(), que devuelve un objeto **ResultSet**, que nos permitirá acceder a los resultados.
  - El **ResultSet** utiliza el concepto de cursor de base de datos para ir moviéndose por las filas de datos recuperadas.
  - Las columnas pueden accederse en cualquier orden, utilizando su posición o su nombre.
  - El objeto **ResultSet** incluye métodos getXXX que permiten recuperar valores de distintos tipos.

JDBC

Pasos para una conexión JDBC - Quinto paso

Pasos para una conexión JDBC - Quinto paso

- Recuperación
  - Cuando ejecutamos una consulta debemos emplear el método **executeQuery**(), que devuelve un objeto **ResultSet**, que nos permitirá acceder a los resultados.
  - El **ResultSet** utiliza el concepto de cursor de base de datos para ir moviéndose por las filas de datos recuperadas.
  - Las columnas pueden accederse en cualquier orden, utilizando su posición o su nombre.
  - El objeto **ResultSet** incluye métodos getXXX que permiten recuperar valores de distintos tipos.

JDBC

Pasos para una conexión JDBC - Quinto paso

Pasos para una conexión JDBC - Quinto paso

- Recuperación
  - Cuando ejecutamos una consulta debemos emplear el método **executeQuery**(), que devuelve un objeto **ResultSet**, que nos permitirá acceder a los resultados.
  - El **ResultSet** utiliza el concepto de cursor de base de datos para ir moviéndose por las filas de datos recuperadas.
  - Las columnas pueden accederse en cualquier orden, utilizando su posición o su nombre.
  - El objeto **ResultSet** incluye métodos getXXX que permiten recuperar valores de distintos tipos.

JDBC

Pasos para una conexión JDBC - Quinto paso

Pasos para una conexión JDBC - Quinto paso

- Recuperación
  - Cuando ejecutamos una consulta debemos emplear el método **executeQuery**(), que devuelve un objeto **ResultSet**, que nos permitirá acceder a los resultados.
  - El **ResultSet** utiliza el concepto de cursor de base de datos para ir moviéndose por las filas de datos recuperadas.
  - Las columnas pueden accederse en cualquier orden, utilizando su posición o su nombre.
  - El objeto **ResultSet** incluye métodos getXXX que permiten recuperar valores de distintos tipos.

JDBC

Pasos para una conexión JDBC - Quinto paso

Pasos para una conexión JDBC - Quinto paso

- Recuperación
  - Cuando ejecutamos una consulta debemos emplear el método **executeQuery**(), que devuelve un objeto **ResultSet**, que nos permitirá acceder a los resultados.
  - El **ResultSet** utiliza el concepto de cursor de base de datos para ir moviéndose por las filas de datos recuperadas.
  - Las columnas pueden accederse en cualquier orden, utilizando su posición o su nombre.
  - El objeto **ResultSet** incluye métodos getXXX que permiten recuperar valores de distintos tipos.

JDBC

Pasos para una conexión JDBC - Quinto paso

```
// ...
String id, nombre;
Date fecha;
int estado;
double saldo;
while (rs.next()) {
  // Se recupera cada columna por separado
  id = rs.getString("id");
  nombre = rs.getString("nombre");
  fecha = rs.getDate("fecha");
  estado = rs.getInt("estado");
  saldo = rs.getDouble("saldo");
  // Operacion que realizamos con cada fila
  System.out.println("ID: " + id + ", Nombre: " + nombre + ",
     Fecha: " + fecha);
}
// ...
```

Preguntas

Preguntas ?