PRACTICA 3: MANEJO DE FECHAS

TIPO FECHA: DATE

Si en Java tienes definido una variable, atributo de clase tipo fecha se pueden utilizar distintos tipos según las necesidades del programador:

```
Java.util.Date

 Java.util.Date (Fecha y Hora)

Clases con las que trabaja el API de jdbc

Java.sql.Date (fecha)
Java.sql.Time (hora)
Java.sql.TimeStamp (fecha + hora)
```

Pojo

Declarar los atributos del pojo como Date (**java.util.Date**)

```
public class Salida {
	private Integer idSalida;
	private Integer usuario;
	private Integer articulo;
	private  Date fechaSalida;
	private Date fechaDevolucion;
...
```

Capa Interfaz

Leer fecha de interfaz

Posiblemente la interfaz recojas la fecha como un String, para manejarlo como fecha

```
String fecha= “20/10/2008”
```

Se puede crear el Objeto format que transforma el String en un format que se puede transformar a java.util.Date. El formato de fecha que debes utilizar “dd/MM/yyyy”, ojo con las mayúsculas y minúsculas porque sino la traducción a MySQL se hará incorrecta.

```
SimpleDateFormat  sf= new SimpleDateFormat("dd/MM/yyyy");

Date date = sf.parse(sfecha);
```

O bien se pueden realizar en una sola instrucción.

```
Java.util.Date f= new SimpleDateFormat(”dd/MM/yyyy”).parse(fecha);
```

Almacenar fecha en el pojo

En nuestro caso particular en el que trabajamos con capas DAO, de trabajo con clases DAO, leemos una fecha de la interfaz de usuario y la debemos de almacenar en el atributo del Pojo. Por ejemplo

```
Salida s= new Salida();

s.setfechaSalida(date);
```

Leer fecha de un pojo

Por otra parte para mostrar en la interfaz la información recuperada de la base de datos se ha de recuperar también a través de la información que en la capa DAO se ha almacenado en el pojo.

```
Date date=s.getfechaSalida();
```

Clases DAO

Java.sql.Date

Desde las Clases DAO se pueden realizar dos tipos de operaciones con la base de datos. Operaciones que implican escribir información en la base de datos o leer datos de la base de datos a través de una conexión JDBC

Escribir en la base de datos (pasar el valor java.util.Date que es el tipo que almacena el pojo a java.sql.Date a un Statement)

//Pasamos Pojo con atributos fecha tipo java.util.Date

El valor recogido en la interfaz que hemos guardado en el Pojo lo pasamos a la capa DAO

```
public void grabar(Salida s) throws BusinessException {

…

//operación sql asociada al PreparedStatemen

Prepared Satatemen stm= …

Stm.setDate(1,new java.sql.Date(s.getfechasalida().getTime());

…}
```

Leer una fecha de la base de datos del resultSet (leer el valor java.sql.Date a un java.util.Date de un ResultSet)

Recuperar información del ResultSet.

```
java.util.Date fecha = result.getDate("fechasalida");
```

Guardar la información en el Pojo.

```
s.setfechaSalida(fecha);
```

TIPO FECHA: LOCALDATETIME

```
java.util.localDateTime
```

Pojo

Declarar los atributos del pojo como Date (**java.util.Date**)

```
public class Salida {
	private Integer idSalida;
	private Integer usuario;
	private Integer articulo;
	private  LocalDateTime fechaSalida;
	private LocalDateTime fechaDevolucion;
...
```

Interfaz

Leer fecha del interfaz como String y pasarla a java.util. Formato de la fecha `yyyy-MM-dd HH:mm:ss` ojo con las mayúsculas y minúsculas sino la traducción será incorrecta.

```
String str = "2016-03-04 11:30:40";

DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss");

LocalDateTime dateTime = LocalDateTime.parse(str, formatter);
```

dataTime será el valor que se guarde en **el atributo fecha del Pojo**

Salida s;

```
s.setFechaSalida(dateTime);
```

El tipo del atributo del Pojo será del la clase LocalDateTime

Clases DAO

```
Java.sql.Date
```

Desde las Clases DAO se pueden realizar dos tipos de operaciones con la base de datos. Operaciones que implican escribir información en la base de datos o leer datos de la base de datos a través de una conexión JDBC.

Escribir en la base de datos PreparedStatement(pasar el valor java.util.LocalDateTime a java.sql.Timestamp a un Statement)

```
public void grabar(Salida s) throws BusinessException {
```

```
pstm.setTimestamp(3,java.sql.Timestamp.valueOf(s.getFechaSalida()));
```

Leer una fecha de la base de datos del resultSet (leer el valor java.util.LocalDateTime a java.sql.Timestamp de un ResultSet)

```
LocalDateTime fechaSal = rsSelect.getTimestamp("fechaSalida")).toLocalDateTime();

salida.setFechaSalida(fechaSal);
```

PRACTICA MANEJO DE FECHAS

1) Implementar DAO Salida, método grabar, buscarporId, buscarTodos.
2) Realizar un método que el usuario introduzca dos fechas y obtengas un listado de los artículos y usuarios (articulo, usuario) de los que se han registrado salidas entre esas dos fechas