Licenciatura en Sistemas - Orientación a Objetos II – 2017

Prof. Titular: Lic. María Alejandra Vranić alejandravranic@gmail.com
facebook: alejandravranic

Prof. Ayudantes: APU Leandro Ríos leandro.rios.unla@gmail.com
facebook: Leandro Ríos

APU Gustavo Siciliano gussiciliano@gmail.com
facebook: Gus Siciliano

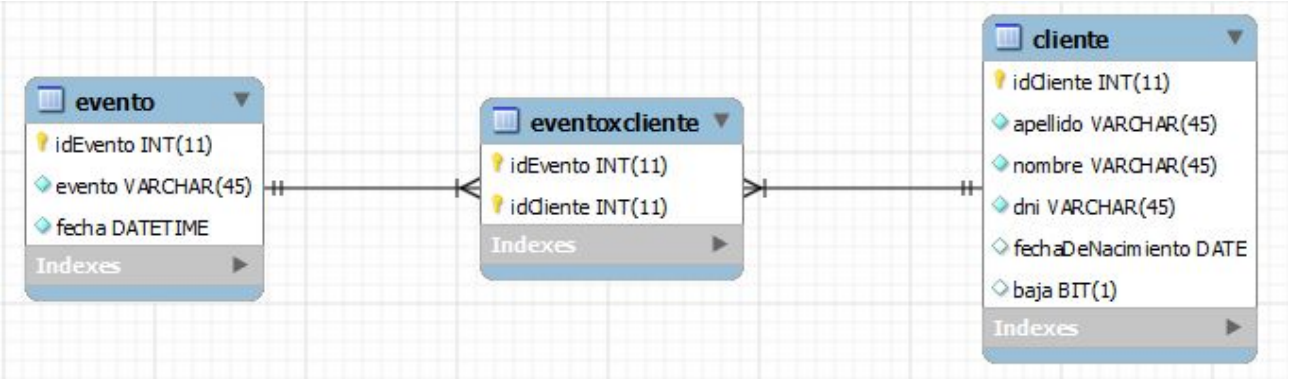
Grupo en facebook: Unla2017OO2-1

IDE: Eclipse
Persistencia de datos: MySQL
Bibliografía: ver programa Hibernate

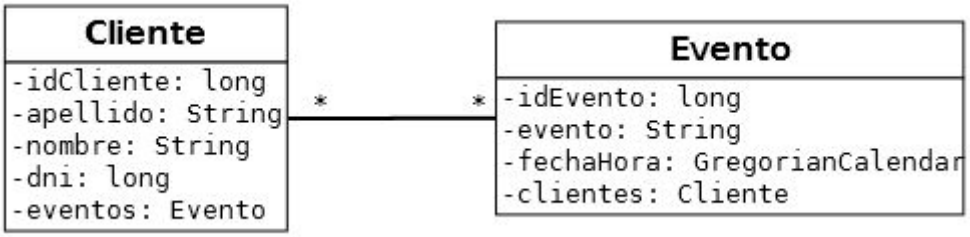
Framework Hibernate Relación Muchos-a-Muchos (bidirecional)

Importar bd-hibernate-muchos-a-muchos.sql

Modelo entidad relación



UML Diagrama de Clases - Capa de Datos



Archivos de mapeo

Cliente.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping>
<class name="datos.Cliente" table="cliente">
    <id column="idCliente" name="idCliente">
```

```
      <generator class="identity"/>
    </id>
    <property column="apellido" name="apellido" type="string"/>
    <property column="nombre" name="nombre" type="string"/>
    <property column="dni" name="dni" type="int"/>
    <property column="fechaDeNacimiento" name="fechaDeNacimiento" type="calendar"/>
    <property column="baja" name="baja" type="boolean"/>
    <set table="eventoxcliente" name="eventos" outer-join="true">
       <key column="idCliente"/>
       <many-to-many column="idEvento" class="datos.Evento"/>
       </set>
 </class>
</hibernate-mapping>
```

Evento.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
<class name="datos.Evento" table="evento">
    <id column="idEvento" name="idEvento">
      <generator class="identity"/>
    </id>
    <property column="evento" name="evento" type="string"/>
    <property column="fecha" name="fecha" type="calendar"/>
    <set table="eventoxcliente" name="clientes" outer-join="true">
       <key column="idEvento"/>
    <many-to-many column="idCliente" class="datos.Cliente"/>
    </set>
</class>
</hibernate-mapping>
```

En hibernate.cfg.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD
3.0//EN" "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
  <session-factory>
    <property name="connection.driver_class">com.mysql.jdbc.Driver</property>
    <property
name="connection.url">jdbc:mysql://localhost/bd-hibernate-muchos-a-muchos</property>
    <property name="connection.username">root</property>
    <property name="connection.password">root</property>
    <property name="connection.pool_size">1</property>
    <property name="dialect">org.hibernate.dialect.MySQLDialect</property>
    <property name="show_sql">true</property>    <!-- en true muestra hql en consola-->

    <!--Mapeo Entidades    -->
       <mapping resource="mapeos/Cliente.hbm.xml"/>
       <mapping resource="mapeos/Evento.hbm.xml"/>
    </session-factory>
</hibernate-configuration>
```

En la capa de datos en Cliente. Implementar equals, agregar y eliminar evento

```java
package datos;

import java.util.GregorianCalendar;
import java.util.Iterator;
import java.util.Set;
import negocio.Funciones;


public class Cliente {
	private long idCliente;
	private String apellido;
	private String nombre;
	private int dni;
	private GregorianCalendar fechaDeNacimiento;
	private boolean baja;
	private Set<Evento> eventos;

	public Cliente(){}

	public Cliente(String apellido, String nombre, int dni,
			GregorianCalendar fechaDeNacimiento) {
		super();
		this.apellido = apellido;
		this.nombre = nombre;
		this.dni= dni;
		this.fechaDeNacimiento = fechaDeNacimiento;
		this.baja=false;
	}


	public long getIdCliente() {
		return idCliente;
	}

	protected void setIdCliente(long idCliente) {
		this.idCliente = idCliente;
	}

	public String getApellido() {
		return apellido;
	}

	public void setApellido(String apellido) {
		this.apellido = apellido;
	}

	public String getNombre() {
		return nombre;
	}

	public void setNombre(String nombre) {
		this.nombre = nombre;
	}


	public int getDni() {
		return dni;
	}
```

```
	public void setDni(int dni) {
		this.dni = dni;
	}


	public GregorianCalendar getFechaDeNacimiento() {
		return fechaDeNacimiento;
	}

	public void setFechaDeNacimiento(GregorianCalendar fechaDeNacimiento) {
		this.fechaDeNacimiento = fechaDeNacimiento;
	}

	public boolean isBaja() {
		return baja;
	}

	public void setBaja(boolean baja) {
		this.baja = baja;
	}


	public Set<Evento> getEventos() {
		return eventos;
	}

	protected void setEventos(Set<Evento> eventos) {
		this.eventos = eventos;
	}

	public boolean equals(Cliente c){
	}

	public boolean agregar(Evento evento){
	}

	 public boolean eliminar(Evento evento){
	}

	public String toString(){
		return (idCliente+" "+apellido+" "+nombre+" DNI: "+dni+" F.de Nacimiento:
"+Funciones.traerFechaCorta(fechaDeNacimiento)+" "+baja);
	}
}
```

Crear el dato Evento. Implementar equals, agregar y eliminar cliente

```
package datos;

import java.util.GregorianCalendar;
import java.util.Iterator;
import java.util.Set;

import negocio.Funciones;
public class Evento {
	private long idEvento;
```

```
	private String evento;
	private GregorianCalendar fecha;
	private Set<Cliente> clientes;

	public Evento(){}

	public Evento(String evento, GregorianCalendar fecha) {
		super();
		this.evento = evento;
		this.fecha = fecha;
	}

	public long getIdEvento() {
		return idEvento;
	}

	protected void setIdEvento(long idEvento) {
		this.idEvento = idEvento;
	}

	public String getEvento() {
		return evento;
	}

	public void setEvento(String evento) {
		this.evento = evento;
	}

	public GregorianCalendar getFecha() {
		return fecha;
	}

	public void setFecha(GregorianCalendar fecha) {
		this.fecha = fecha;
	}

	public Set<Cliente> getClientes() {
		return clientes;
	}

	public void setClientes(Set<Cliente> clientes) {
		this.clientes = clientes;
	}

	public boolean equals(Evento evento){
	 }

	public boolean agregar(Cliente cliente){
	}

	public boolean eliminar(Cliente cliente){
	}

	public String toString(){
		return idEvento+" "+evento+" "+Funciones.traerFechaCorta(fecha)+"
"+Funciones.traerHora(fecha);
	}
```

```
}
```

En ClienteDao agregar el método:

```
public Cliente traerClienteYEventos(long idCliente) throws HibernateException {
    Cliente objeto = null;
    try {
        iniciaOperacion();
        String hql="from Cliente c where c.idCliente =" + idCliente;
        objeto = (Cliente) session.createQuery(hql).uniqueResult();
        Hibernate.initialize(objeto.getEventos());
    } finally {
        session.close();
    }
    return objeto;
}
```

Desarrollo pendiente:

Crear la clase EventoDao

En EventoABM agregar los casos de uso:

Agregar un cliente a un evento

A partir de los parámetro idCliente y idEvento:

- traer el cliente
- traer evento y sus clientes
- agregar el cliente a un evento y actualizar.

Eliminar un cliente a un evento y actualizar

En EventoABM agregar los casos de uso:

Traer un cliente y los eventos

Agregar un evento a un cliente y actualizar

Eliminar un evento a un cliente

Realizar los test de los casos de uso