# 0110510-jsf-logo.tiff

**Ejercicios Intermedios y acceso a base de datos**

## Ejercicio Integrador

**Objetivo**: Crear una pequeña aplicación WEB que realice 4 operaciones básicas (CRUD) integrando las tecnologías vistas en el curso (Acceso a BD, Hibernate, JSF y Primefaces).

Como componente de JSF se usará Primefaces que facilita la creación de nuestras aplicaciones.

El resultado será similar al siguiente:

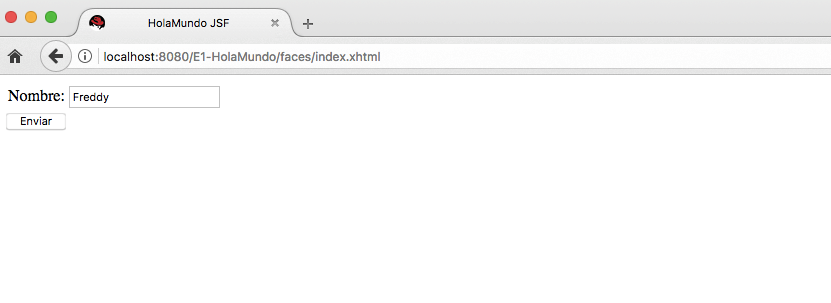


Ilustración 1 - Resultado esperado del ejercicio 1

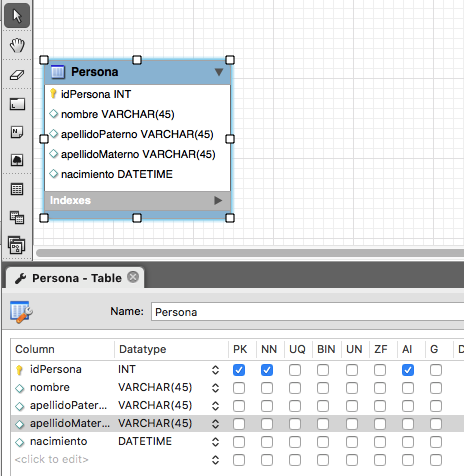
**Sección I**

## Creación de la base de datos

Requisitos:

* NetBeans v8.x
* MySQL Server
* MySQL Workbench

Primero empezaremos con la base de datos, abriremos MySQL Workbench, que viene en la suite que se instaló de MySQL community, nos dirigimos a File -> New Model y crear el siguiente modelo con una tabla llamada Persona:



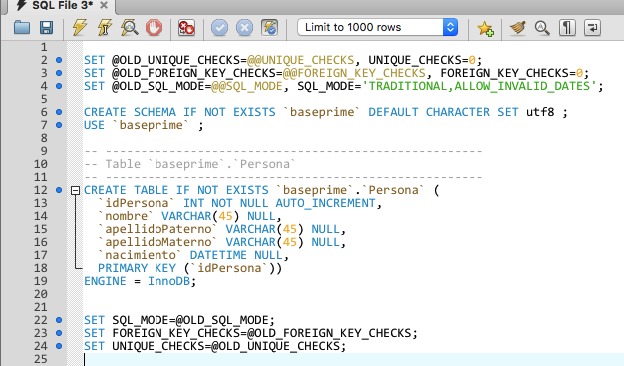
Guardar el modelo y exportarlo como Script SQL.

* Crear una base de datos con las siguientes características:

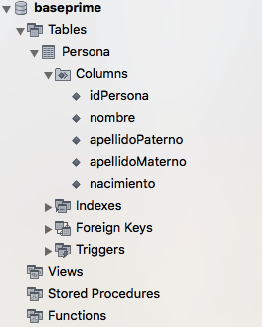
create database baseprime;

use baseprime;

Insertar el script a la BD.



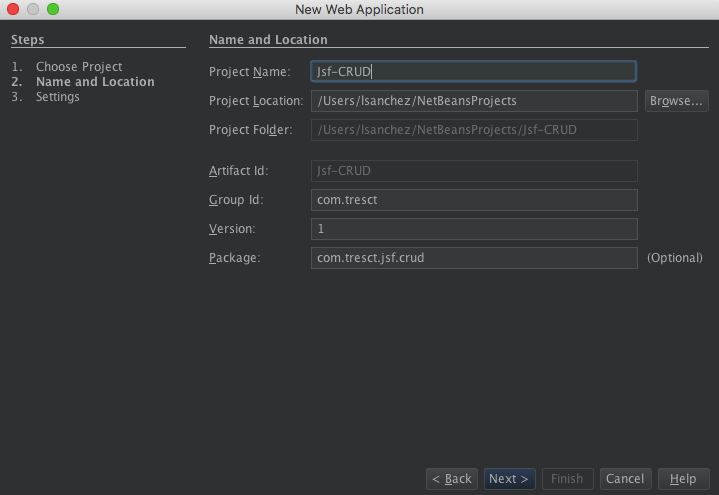
Validar que la base se encuentre activa:



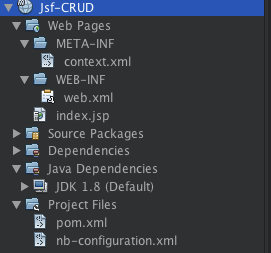
**Sección II**

## Creación del proyecto WEB.

Abriremos NetBeans IDE y crearemos un proyecto Web apoyados de la herramienta Maven con la configuración que se muestra a continuación:



Generar la siguiente estructura:



Agregar las siguientes dependencias en el POM.xml:

***<dependencies>***

***<dependency>***

***<groupId>org.primefaces</groupId>***

***<artifactId>primefaces</artifactId>***

***<version>5.3</version>***

***</dependency>***

***<dependency>***

***<groupId>org.hibernate</groupId>***

***<artifactId>hibernate-core</artifactId>***

***<version>5.2.1.Final</version>***

***</dependency>***

***<dependency>***

***<groupId>org.hibernate</groupId>***

***<artifactId>hibernate-validator</artifactId>***

***<version>5.2.1.Final</version>***

***</dependency>***

***<dependency>***

***<groupId>org.hibernate</groupId>***

***<artifactId>hibernate-entitymanager</artifactId>***

***<version>5.2.1.Final</version>***

***</dependency>***

***<dependency>***

***<groupId>mysql</groupId>***

***<artifactId>mysql-connector-java</artifactId>***

***<version>5.1.43</version>***

***</dependency>***

***<dependency>***

***<groupId>javax.servlet</groupId>***

***<artifactId>servlet-api</artifactId>***

***<version>2.5</version>***

***<scope>provided</scope>***

***</dependency>***

***<dependency>***

***<groupId>javax.servlet.jsp</groupId>***

***<artifactId>jsp-api</artifactId>***

***<version>2.1</version>***

***<scope>provided</scope>***

***</dependency>***

***<dependency>***

***<groupId>com.sun.faces</groupId>***

***<artifactId>jsf-api</artifactId>***

***<version>2.2.14</version>***

***</dependency>***

***<dependency>***

***<groupId>com.sun.faces</groupId>***

***<artifactId>jsf-impl</artifactId>***

***<version>2.2.14</version>***

***</dependency>***

***<dependency>***

***<groupId>javax.faces</groupId>***

***<artifactId>jsf-api</artifactId>***

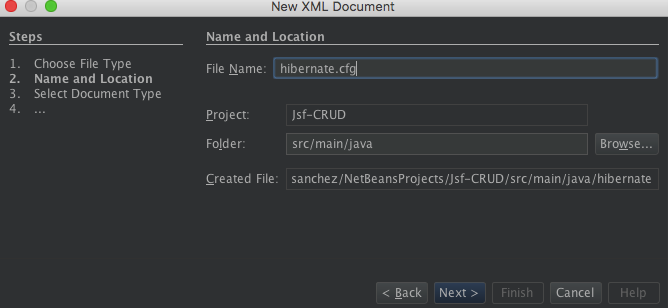
***<version>2.0</version>***

***<scope>provided</scope>***

***</dependency>***

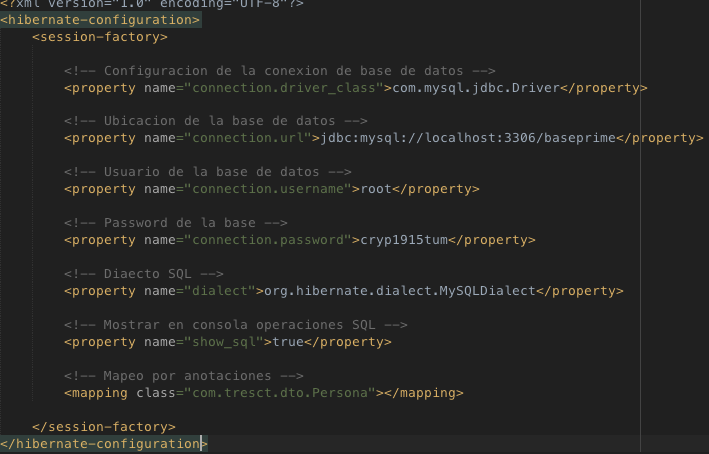
***</dependencies>***

Generar los archivos básicos de configuración en Hibernate:



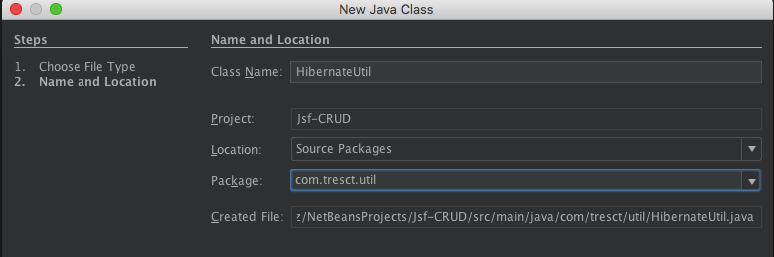
Dentro del archivo de configuración agregaremos lo siguiente:

Este archivo contiene la configuración para entablar la conexión con la base de datos y los mapeos de las clases Java que representan las tablas en BD.

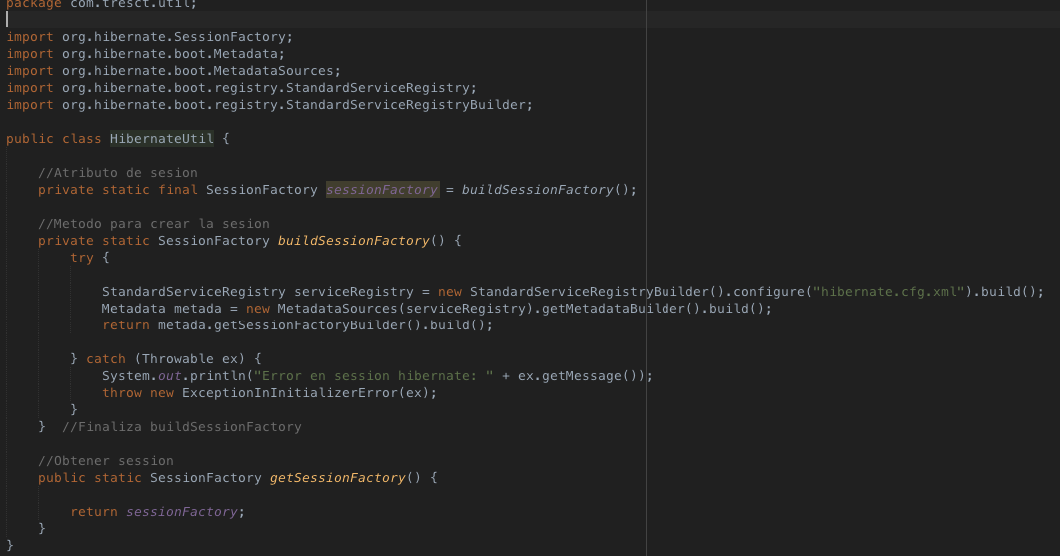


Posteriormente escribiremos la clase Java para HibernateUtil:

Recordar que es necesario escribir un archivo Util para utilizar el SessionFactory de Hibernate. Sin este archivo no podemos realizar prácticamente nada con Hibernate.

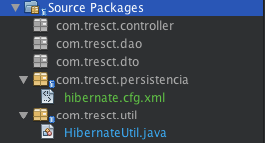


Dentro del archivo escribir lo siguiente:



Ahora es tiempo de construir la arquitectura de nuestro proyecto:

Para ello necesitamos crear los siguientes paquetes:

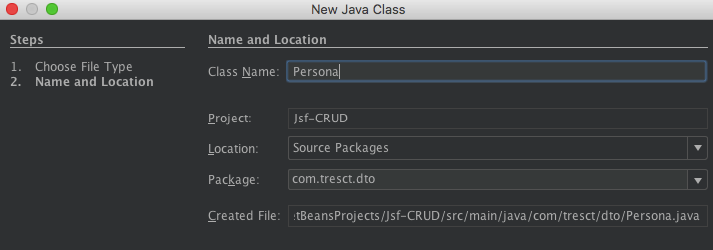


**Sección III**

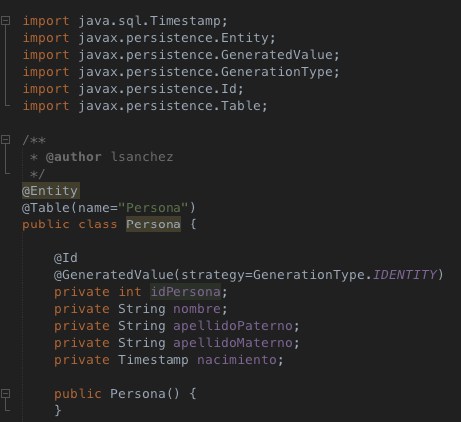
## Creación del modelo.

Construiremos el modelo de nuestra aplicación por medio de los patrones de diseño DTO y DAO utilizando los paquetes antes creados.

1. Construir la clase DTO correspondiente a la tabla Persona de la BD.

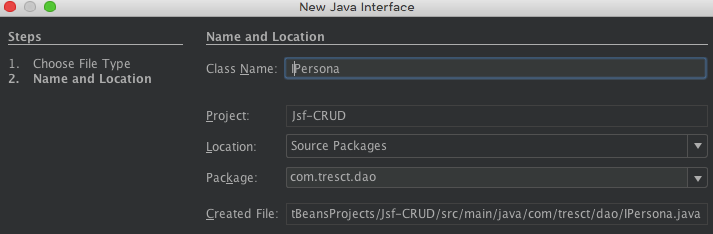


No olvide utilizar las anotaciones de Hibernate para realizar los mapeos objeto – relacional

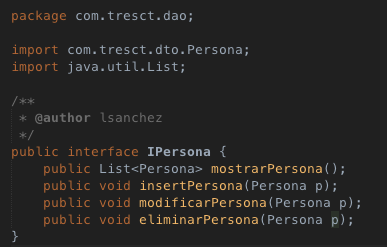


Esta imagen muestra únicamente el constructor vacío. Se deberá complementar con constructor con todos los parámetros, metodos set, get y por ultimo con el método toString().

1. Construcciones de las interfaces donde se definen reglas de implementación:

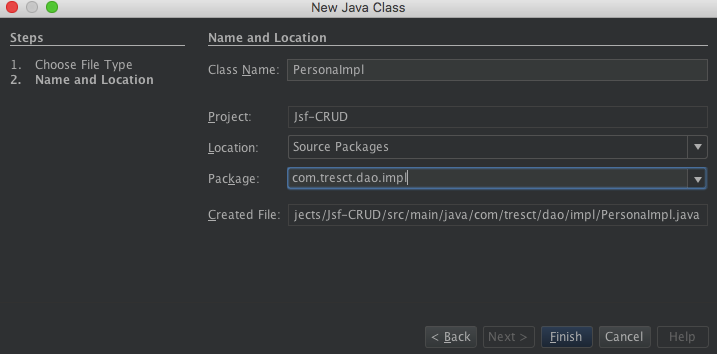


Se le dará el nombre de ***IPersona***.



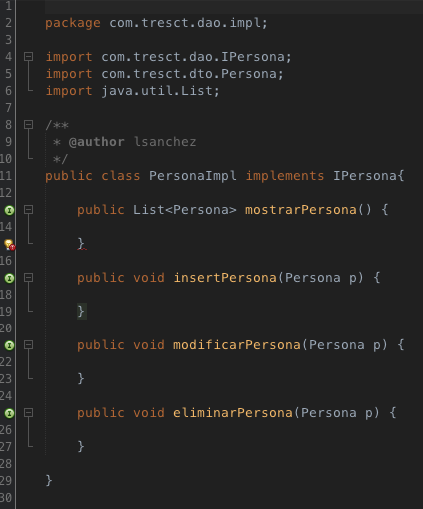
1. Creación de la implementación de la interfaz. Aquí se define el como se van a implementar los metodos CRUD de la aplicación.

Para ello vamos a crear una nueva clase Java con la siguiente configuración:



No olvidar que esta clase debe implementar la interfaz.

Generando la siguiente estructura de clase:

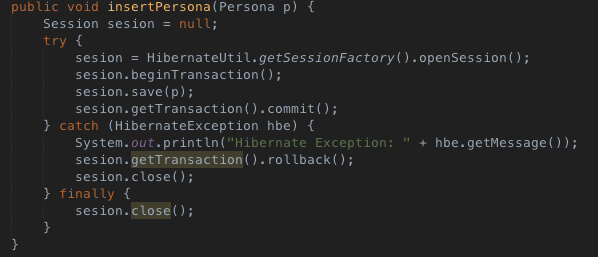


E iremos llenado los metodos de la siguiente forma:

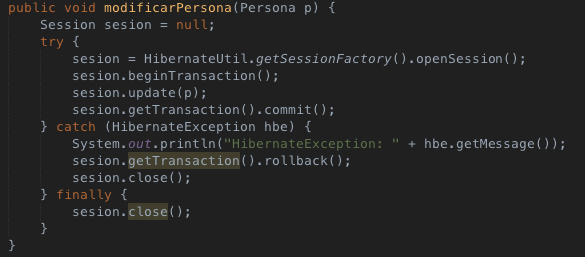
mostrarPersonas()



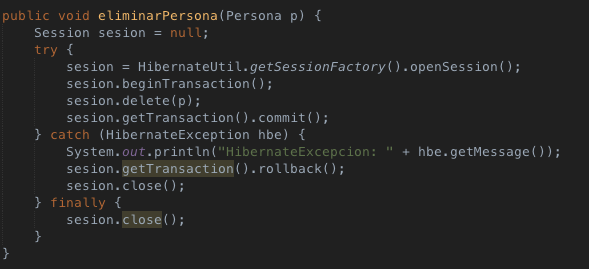
insertarPersona()



modificarPersona()



eliminarPersona()



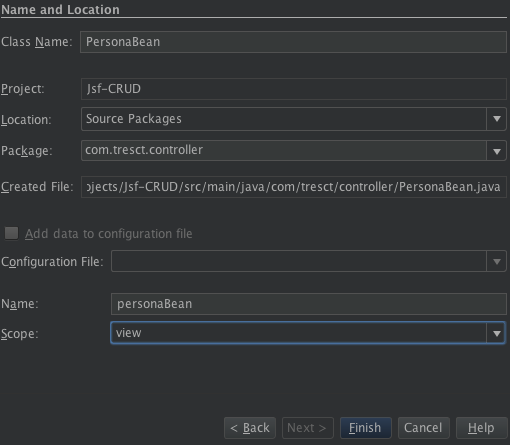
Con estos metodos finalizamos la construcción del modelo de nuestra aplicación.

**Sección IV**

## Creación del controlador

Para la construcción del controlador de nuestra aplicación crearemos un JSF ManagedBean el cual manejara la vista donde incluiremos nuestro componentes.

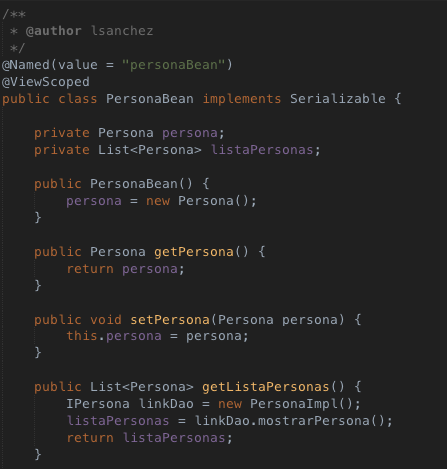
1. Creación del Bean Administrado PersonaBean con la siguiente configuración:



Ahora modificaremos el Bean de la siguiente manera:

* Crearemos un objeto de tipo persona
* Crearemos una lista de objetos persona
* Crearemos getters y setters para que puedan ser accedidos esde la vista
* Se añadirán los metodos insertar, modificar, eliminar y mostrar, los cuales accederán a nuestros metodos del DAO.

Nota: Dentro del constructor personaBean() definir lo siguiente: persona = new Persona() para que deje de ser un objeto nulo.



**Sección V**

## Creación de la vista

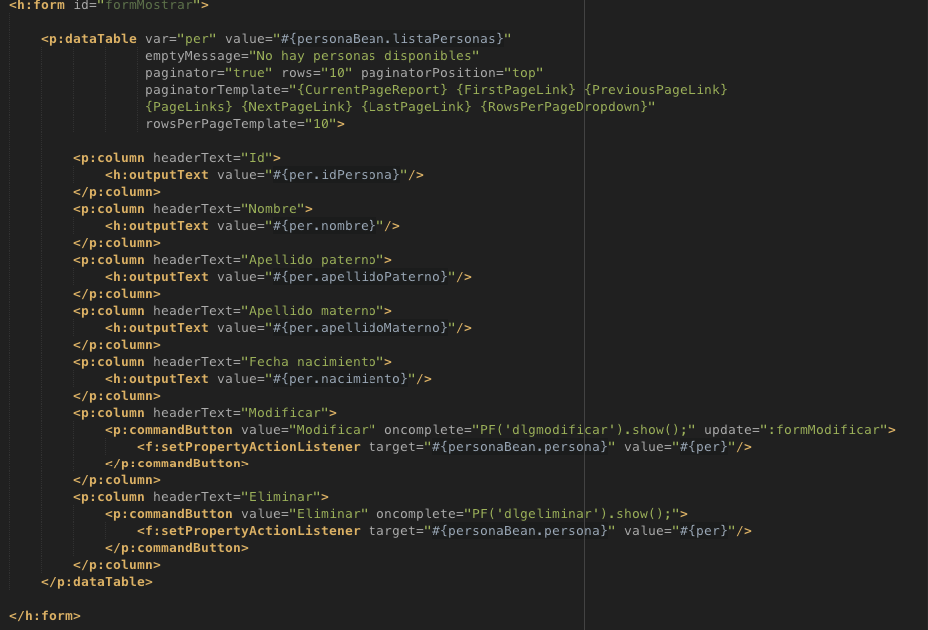
Para la construcción de la vista utilizaremos JSF Facelets y componentes de primefaces.

El objetivo es comprender el uso y la practicidad de utilizar una de las extensiones mas populares para JSF.

1. Operación Insert



1. Operación mostrar



1. Modificar y eliminar
2. 