# 02\_HIBERNATE

## Mapeo de una Entidad

Una entidad va a ser una simple clase Java que deseamos persistir en la base de datos.

**La clase Java**

La clases Java deberán tener las siguientes características: (POJO)

* Deben tener un constructor público sin ningún tipo de argumentos [1)](http://cursohibernate.es/doku.php?id=unidades:02_hibernate:02_entidad#fn__1).
* Para cada propiedad que queramos persistir debe haber un método get/set asociado.
* Implementar el interfaz [Serializable](http://docs.oracle.com/javase/7/docs/api/java/io/Serializable.html" \t "_blank) [2)](http://cursohibernate.es/doku.php?id=unidades:02_hibernate:02_entidad" \l "fn__2)

# ¿Ya podemos persistir la clase Profesor usando hibernate? Pues ****NO****, debemos indicarle a hibernate toda la metainformación relativa a esta clase.

# Hay que explicarle como se mapeará el objeto en una base de datos relacional , indicando para ello en que tabla de base de datos se debe guardar cuál es la clave primaria de la tabla, las columnas que tiene, etc.

# Se puede realizar la persistencia con:

# Ya hemos visto que se puede configurar usando los ficheros de mapeo o las anotaciones [Mapeo de una Entidad](http://cursohibernate.es/doku.php?id=unidades:02_hibernate:02_entidad)

## Fichero de mapeo ''.hbm.xml''

# Para cada clase que queremos persistir se creará un fichero xml con la información que permitirá mapear la clase a una base de datos relacional. Este fichero estará en el mismo paquete que la clase a persistir.

## Anotaciones

* El uso de Anotaciones Java en el propio código. Estas anotaciones permiten especificar de una forma más compacta y sencilla la información de mapeo de las clases Java.
* Inicialmente Hibernate creó sus propia anotaciones en el paquete [org.hibernate.annotations](http://docs.jboss.org/hibernate/core/4.1/javadocs/org/hibernate/annotations/package-summary.html" \t "_blank)
* pero a partir de la versión 4 de Hibernate la mayoría de dichas anotaciones han sido [java.lang.Deprecated](http://docs.oracle.com/javase/7/docs/api/java/lang/Deprecated.html" \t "_blank) y ya no deben usarse.
* Las anotaciones que deben usarse actualmente son las del estándar de JPA que se encuentran en el paquete **[javax.persistence](http://docs.oracle.com/javaee/5/api/javax/persistence/package-summary.html" \t "_blank).**
* Sin embargo hay características específicas de Hibernate que no posee JPA lo que hace que aun sea necesario usar alguna anotación del paquete **[org.hibernate.annotations](http://docs.jboss.org/hibernate/core/4.1/javadocs/org/hibernate/annotations/package-summary.html" \t "_blank)** pero en ese caso Hibernate 4 no las ha marcado como **[java.lang.Deprecated](http://docs.oracle.com/javase/7/docs/api/java/lang/Deprecated.html" \t "_blank)**.

**ANOTACIONES**

**@Entity**: Se aplica a la clase e indica que esta clase Java es una entidad a persistir. Es una anotación estándar de JPA. En nuestro ejemplo estamos indicando que la clase Profesor es una entidad que se puede persistir.

**@Table(name=“Profesor”)**: Se aplica a la clase e indica el nombre de la tabla de la base de datos donde se persistirá la clase. Es opcional si el nombre de la clase coincide con el de la tabla.

* Es opcional si el nombre de la clase coincide con el de la tabla. Es una anotación estándar de **JPA**. En nuestro ejemplo estamos indicando que la clase **Profesor** se persistirá en la tabla Profesor de la base de datos.

**@Id**: Se aplica a una propiedad Java e indica que este atributo es la clave primaria. Es una anotación estándar de **JPA**. En nuestro ejemplo estamos indicando que la propiedad Java id es la clave primaria.

**@GeneratedValue:** Esta anotación indica que Hibernate deberá generar el valor de la clave primaria.

**NOTA:**

//Cuando en la base de datos el id es autoincrementable osea que se genera se coloca @GenerateValue y no se crea get y set

**NOTA:**

**Generalmente una tabla de llaves foráneas no se mapea a menos de que tengo un campo adicional.**

