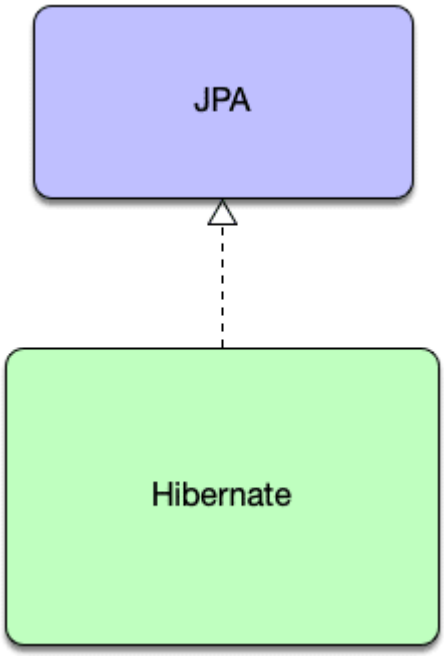
JPA vs Hibernate . ¿Qué diferencia hay entre ellos y cual debemos elegir? . Esta es una de las preguntas más habituales cuando uno empieza a trabajar con un framework de Persistencia.

¿Que es JPA?

JPA o Java Persistence API es una especificación concretamente la JSR 338 . Una especificación no es más que un documento en el que se define como se debe gestionar una funcionalidad X. En este caso una capa de persistencia con objetos Java . Por ejemplo que anotaciones han de usarse, como han de persistirse los objetos como han de buscarse , cuál es su ciclo de vida etc .

Al tratarse de un documento evidentemente no implementa nada . Para poder trabajar con ella necesitaremos tener un Framework que implemente la especificación, uno de los más conocidos es Hibernate



¿Existen más implementaciones de JPA que al fin y al cabo es únicamente una especificación? . Si aquí podemos ver algunas:

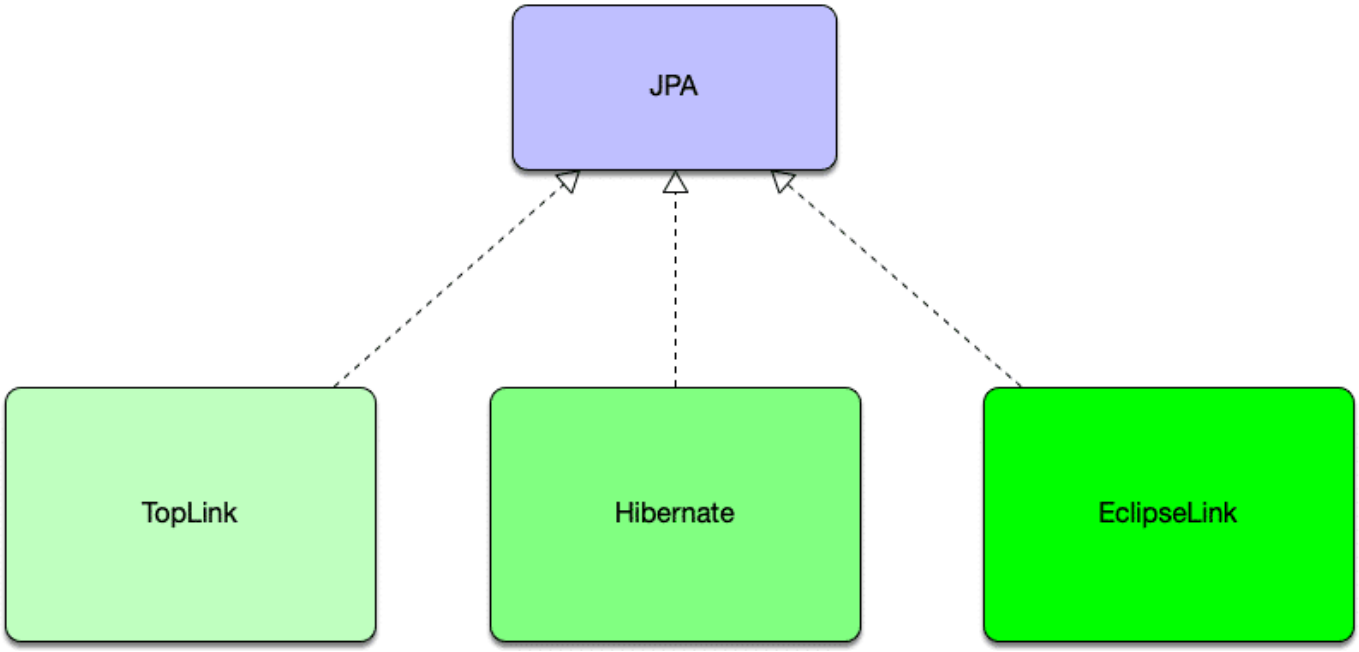
EclipseLink : Otra implementación muy utilizada ya que esta apoyada por la fundación Eclipse y da soporte a JPA.

DataNucleus : Otro framework de persistencia que soporta JPA pero incluye muchas más opciones y tipos de persistencia.

TopLink : La implementación de Oracle clásica para sus productos hoy por hoy esta basada en EclipseLink

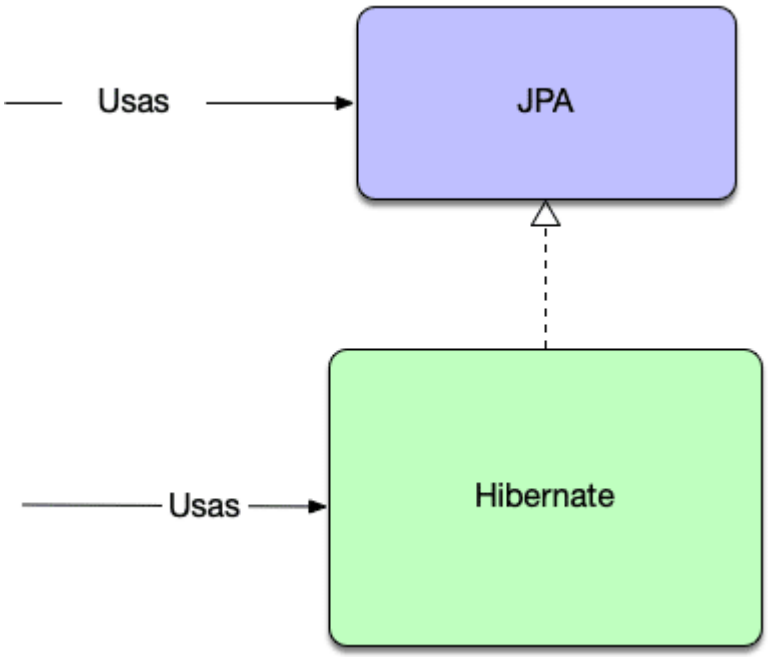
ObjectDB Otra implementación de JPA que promete un rendimiento alto comparado con sus competidores.

Por lo tanto no solo existe Hibernate



¿JPA vs Hibernate?

Normalmente la pregunta de mucha gente es que usar si JPA o Hibernate , es decir si usar directamente la especificación o usar directamente el framework.



¿Usar Hibernate?

Muchas veces me encuentro con respuestas bastante razonables en las que la gente prefiere usar el framework (Hibernate) de forma directa . ¿Porque? Porque el framework tiene capacidades adicionales que JPA como standard y especificación no soporta.

Hasta aquí parece todo muy razonable. Sin embargo yo soy partidario de usar siempre JPA . ¿Porque? . Porque en la mayoría de las situaciones el usar directamente el framework aunque aporte alguna cosa adicional esos aportes no son críticos y las pocas ventajas que aporta no son suficientes comparadas con las ventajas de usar directamente la especificación.

¿Usar la especificación?

Vamos a ver un poco sus ventajas:

1. La especificación esta muy trabajada y a nivel cultural muy extendida entre todos los desarrolladores . El uso concreto de un framework no lo esta tanto.
2. La especificación nos permitirá cambiar de implementación de forma transparente si tenemos la necesidad . Es raro tener que hacerlo pero existe la posibilidad y permitirá portar de forma más sencilla nuestra aplicación entre diferentes servidores Java EE que usen diferentes especificaciones.
3. La especificación es mucho más atemporal ya que es un documento y todo el mundo se ciñe a él. Por lo tanto lo aprendido dura más en el tiempo y es más sólido al paso de las décadas.
4. Al ser algo tan inmutable es habitual que aparezcan nuevos frameworks o nuevas especificaciones que lo complementen. Por ejemplo un caso claro de JPA es el uso de Spring Data con JPA que permite un trabajo mucho más cómodo y rápido con Repositorios.

JPA vs Hibernate Conclusión

Mi recomendación es usar siempre JPA como especificación y programar contra la especificación usando EntityManager ,EntityManagerFactory etc y no usar clases concretas

de una implementación como puedan ser en Hibernate Session y SessionFactory. ¿Cuál de todas las implementaciones usar? . Pues para mí esta claro que Hibernate es la implementación más conocida y por lo tanto la que debemos instalar . Eso sí recordemos que debemos usar solo anotaciones del standard de JPA y evitar en lo posible hacer uso directo de las funciones de Hibernate esto nos permitirá mantener la portabilidad del código y las ventajas de que soluciones que salgan a futuro y se apoyen en JPA , las podamos integrar de forma natural. Estas son mis conclusiones sobre JPA vs Hibernate

Otros artículos relacionados

1. JPA Polymorphic Query
2. JPA Single Table Inheritance
3. JPA Orphan Removal y como usarlo