**Java Persistence API (JPA)**

JPA es la propuesta estándar que ofrece Java para implementar un Framework Object Relational Mapping (ORM), que permite interactuar con la base de datos por medio de objetos, de esta forma, JPA es el encargado de convertir los objetos Java en instrucciones para el Manejador de Base de Datos (MDB).

Cuando em pezamos a trabajamos con bases de datos en Java lo primero que nos enseñan es a utiliza el API de JDBC el cual nos permite realizar consultas directas a la base de datos a través de consultas SQL nativas. JDBC por mucho tiempo fue la única forma de interactuar con las bases de datos, pero representaba un gran problema y es que Java es un lenguaje orientado a objetos y se tenía que convertir los atributos de las clases en una consulta SQL como SELECT, INSERT, UPDATE, DELTE, etc. lo que ocasionaba un gran esfuerzo de trabajo y un provocaba muchos errores en tiempo de ejecución, debido principalmente a que las consultas SQL se tenían que generar frecuentemente al vuelo.

Puedes ver en este post la [diferencia entre JPA y JDBC](http://www.oscarblancarteblog.com/2014/07/15/jpa-hibernate-vs-jdbc/).

Una de las cosas más importantes para comprender que es JPA es entender que JPA es una especificación y no un Framework como tal, ¿pero quiere decir esto exactamente?, pues bien, una especificación no es más que un documento en el cual se plasman las reglas que debe de cumplir cualquier proveedor que dese desarrollar una implementación de JPA, de tal forma que cualquier persona puede  tomar la especificación y desarrollar su propia implementación de JPA, ¿Esto quiere decir que pueden existir muchas implementaciones de JPA? la respuesta es sí, de echo en la actualidad existen varios proveedor como lo son los siguientes:

* [Hibernate](http://hibernate.org/)
* [ObjectDB](http://www.objectdb.com/)
* [TopLink](http://www.oracle.com/technetwork/middleware/toplink/overview/index-089172.html)
* [EclipseLink](http://www.eclipse.org/eclipselink/)
* [OpenJPA](http://openjpa.apache.org/)

Dentro de las implementaciones más utilizadas están Hibernate, EclipseLink & TopLink, las dos primeras son las más utilizadas en el mundo open source y TopLink es muy utilizada en desarrollos y productos relacionados con Oracle. Antes de preguntarnos cuál es la diferencia entre todas estas implementaciones tenemos que comprender que en teoría todas debería de ofrecer la misma funcionalidad y el mismo comportamiento, lo que nos permitiría migrar entre una implementación a otra sin afectar en nada nuestra aplicación. Desde luego esto es solo teoría, ya que en la actualidad no todas las implementaciones implementan al 100% la especificación de JPA, además en escenario muy concretos puede que se comporten ligeramente diferente, por lo que puede requerir realizar algunos ajustes antes de migrar correctamente de proveedor.

**Ciclo de vida de las Entidades**

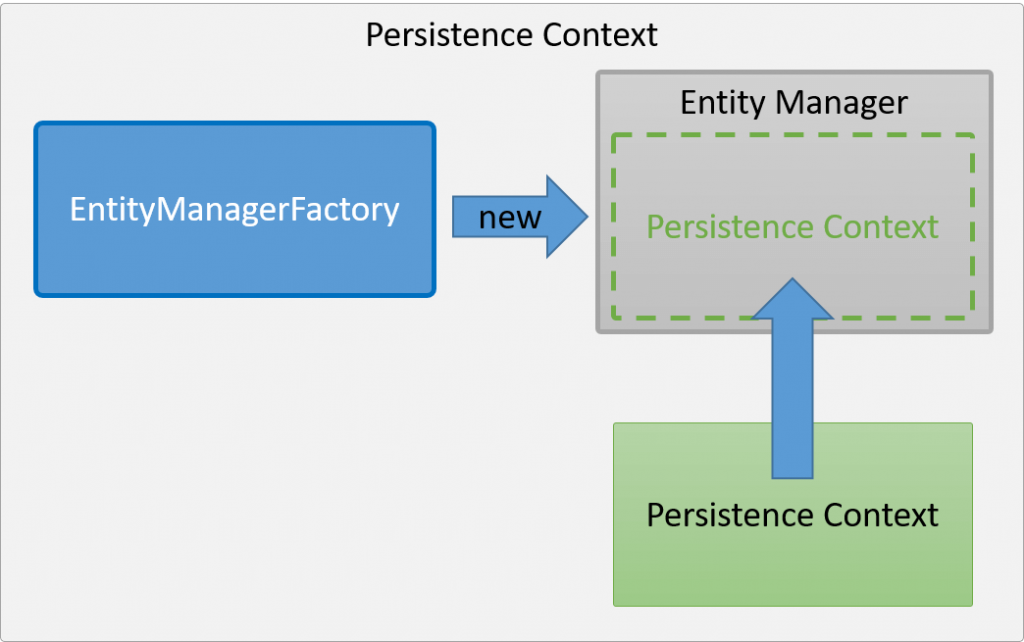


Entender el ciclo de vida de las Entidades es sin duda uno de los puntos cruciales de JPA, pues entender cómo es que una Entidad es gestionada por EntityManager nos permitirá entender mejor como es que JPA funciona y prevenir muchos errores en tiempo de ejecución.

Lo primero que debemos de entender, es que todas entidades que utilicemos con JPA, serán administradas por el EntityManager, es por este motivo que hemos agregado este esta sección al capítulo de EntityManager.

**Persistence Context:**

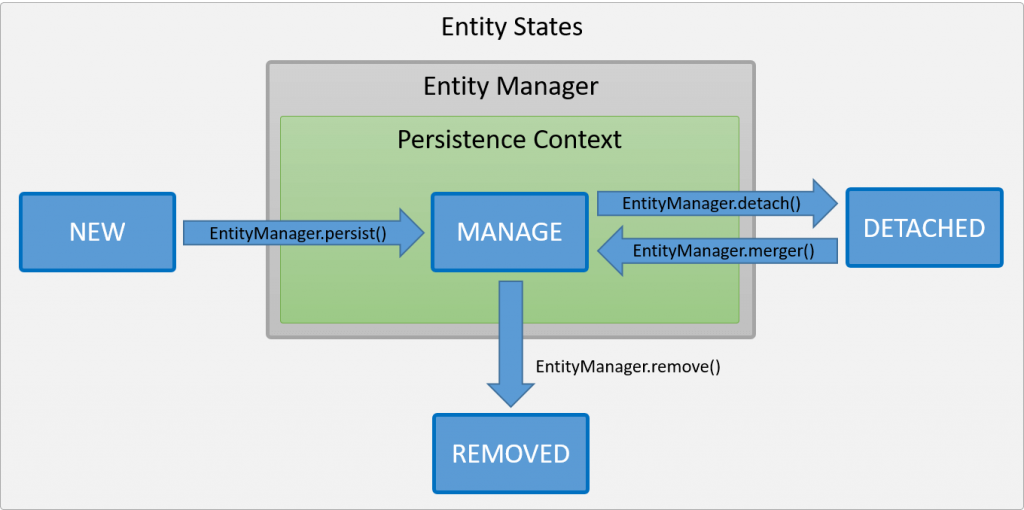
Antes de entrar a los estados de las Entidades es importante entender un nuevo concepto que no hemos analizados en esta guía, se trata del Contexto de persistencia (Persistence Context), este lo podemos ver como contenedor en donde se encuentra todas las Entidades administradas por el EntityManager. Cuando un nuevo EntityManager es creado a través del EntityManagerFactory este le asigna un Unidad de persistencia.

Ciclo de vida de las Entidades

Cabe mencionar que el Persistence Context no es un objeto con el cual nosotros interactuaremos, ya que este esta encapsulado dentro de Entity Manager y solo a través de este es posible afectarlo. En la siguiente sección hablaremos de las operaciones que ofrece el Entity Manager y con las cuales es posible afectar a Unidad de persistencia y las entidades.

**Estados de las entidades:**

A medida que trabajamos con las entidades, estas van cambiando de estado, y el estado de estas será utilizado para realizar operaciones en la base de datos. Un aspecto importante cuando utilizamos JPA es que ya no estamos trabajando con sentencias SQL y en su lugar trabajamos con las Entidades y sus estados. Las entidades en JPA puedan pasar por cuatro estados distintos que se pueden ver en la siguiente imagen:

Estados de las Entidades

Los cuatro estados son:

* **New:** Estado que tiene una entidad cuando es creada con el operador new, por lo tanto, no existe en la base de datos y no está asociada a un contexto de persistencia. Todas las entidades en estado new no serán afectadas en la base de datos cuando la transacción finalice, pues no está asociado a al Context Persistence.
* **Manage:** Todas las entidades que están siendo administradas por JPA están en estado manage, lo que indica que son parte de un contexto de persistencia. Aplica para entidades que serán insertadas, actualizadas o fueron obtenidas por el EntityManager como parte de alguna operación SELECT.
* **Detached:** Las entidades en estado detached, son entidades que si existen en la base de datos, pero por algún motivo no son parte de un contexto de persistencia. Todas las entidades en este estatus no serán afectadas en la base de datos, pues no pertenecen a un Persistence Context
* **Removed:** Cuando una Entidad es marcada para ser eliminada, pasa automáticamente al estado de Removed. Cando la transacción finaliza, todas las Entidades en este status serán eliminadas de la base de datos y ya no existirán más en el Persistence Context.

En la siguiente sección hablaremos de las operaciones básicas que nos permite utilizar el EntityManager y con las cueles es posible mover de estados las entidades. Por lo pronto es importante entender los estados por lo que pueden pasar las entidades para que nos facilite entender las operaciones para buscar, persistir, actualizar y borrar de JPA.