ORM WITH JPA

krissadewo@greenlabs.or.id

ORM

Sebuah cara untuk menempatkan objek kedalam sebuah bentuk database relasional menggunakan metadata yang telah didefinisikan di dalam kelas dari objek tersebut.

JPA

Java Persistence API.
Standard yang mengatur bagaimana ORM bekerja di dalam Teknologi Java

SPESIFIKASI JPA

1. Metadata Mapping dengan menggunakan annotasi
2. API untuk melakukan perintah CRUD
3. Standar perintah dengan menggunakan JPQL, Criteria Query.
4. Monitoring, Dirty Checking, Association fetching, Basic Caching.

Keuntungan

1. Productivity
2. Maintainability
3. Performance
4. Vendor Independence

IMPLEMENTASI JPA

1. Hibernate
2. OPen JPA
3. Toplink
4. Eclipselink
5. Data Nucleus
6. ObjectDB

HIBERNATE

Di buat oleh Gavin King

Di akuisisi oleh JBoss

2010 Versi 3.5 menjadi standar.(Implement JSR 317/JPA 2.0)

2013 Versi 4.3.0 standard.(Implement JSR 338/JPA 2.1)

OBJECT LIFECYCLE

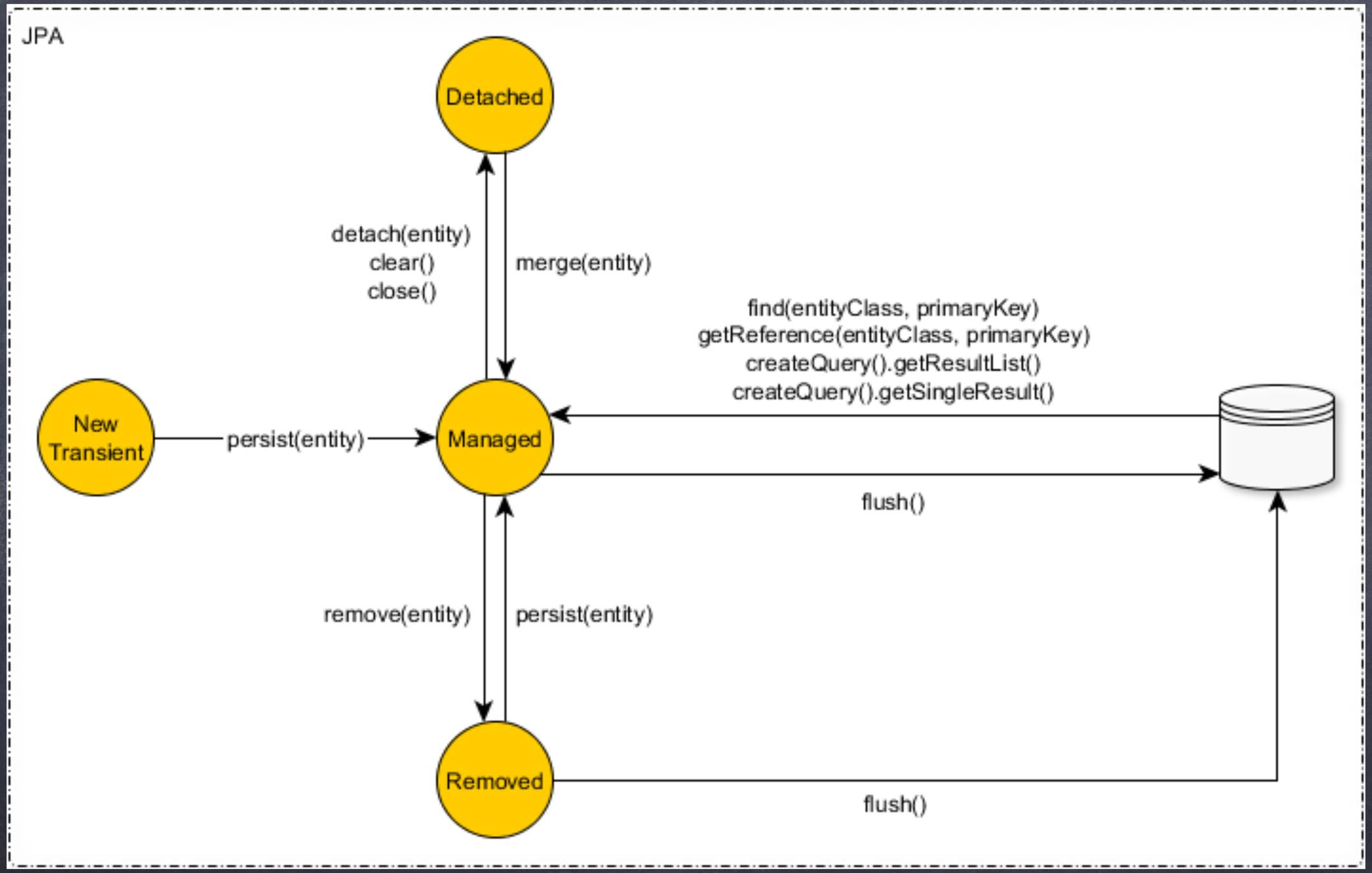
Persistent context bertanggung jawab terhadap state dari objek

Transient
entity has just been instantiated and is not associated with a persistence context and is not mapped to any database table row

Persistence / Managed
associated with a database table row and it's being managed by the current running Persistence Context.

Detached
Once the current running Persistence Context is closed all the previously managed entities become detached.

Removed
the entity has an associated identifier and is associated with a persistence context, however it is scheduled for removal from the database.


JPA
Detached
detach(entity)
clear()
close()
merge(entity)
find(entityClass, primaryKey)
getReference(entityClass, primaryKey)
createQuery().getResultList()
createQuery().getSingleResult()
New Transient
persist(entity)
Managed
flush()
remove(entity)
persist(entity)
Removed
flush()

Untuk mengelola objek maka jpa menggunakan beberapa kelas seperti berikut :

★Persistence unit

Untuk mendefinisikan pengaturan pengaturan yang diperlukan

★Entity Manager Factory

Untuk membuat Entity Manager.

★Persistence context

Berisi kumpulan entity instance yang bersifat unik. Entity instance dan lifecycle dari objek akan dikelola disini.

★Entity Manager

Melakukan interaksi dengan Persistence Context.

★Entity

Sebuah kelas yang mempresentasikan table yang ada didatabase

Making Entity Change State

```
Person person = new Person();
person.setId( 1L );
person.setName("John Doe");
```

Persistence

```
entityManager.persist( person );
```

Remove

```
entityManager.remove( person );
```

```
Person person = entityManager.find( Person.class, personId );
person.setName("John Doe");
entityManager.flush();
```

More....

Object Relations Support

One To One
Many To One

One To Many
Many To Many

Query

JPQL Query

```
Query query = entityManager.createQuery(
    "select p " +
    "from Person p " +
    "where p.name like :name"
);
```

Type Query
Named Query

HQL Query

```
org.hibernate.query.Query query = session.createQuery(
    "select p " +
    "from Person p " +
    "where p.name like :name"
);
```

Criteria Query

```
CriteriaBuilder builder = entityManager.getCriteriaBuilder();

CriteriaQuery<Person> criteria =
builder.createQuery( Person.class );
Root<Person> root = criteria.from( Person.class );
criteria.select( root );
criteria.where( builder.equal( root.get( Person_.name ), "John
Doe" ) );

List<Person> persons =
entityManager.createQuery( criteria ).getResultList();
```

Advance Topics

Managing Detached Entity
Lock Management
Caching
Cascading entity state transitions
Transactions and concurrency control
Performance Tuning

Ref :

http://docs.jboss.org/hibernate/orm/5.2/userguide/