

# LESSON 7

SPRING BOOT: JPA AND DATABASES

MENTOR: ANDREY POMELOV

## SPRING BOOT: JPA AND DATABASES

---

- ORM
- Spring Data JPA
- Entities
- JPA's Persistence Context
- JPA's 4 Lifecycle States

# ORM

---

**ORM** — Object-Relational Mapping или в переводе на русский объектно-реляционное отображение. Это технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования. Если упростить, то ORM это связь Java объектов и записей в БД

## SPRING DATA JPA

---

Репозитории **Spring Data JPA** - это интерфейсы, которые вы можете определить для доступа к данным. Запросы JPA создаются автоматически из имен ваших методов. Например, интерфейс `CityRepository` может объявить метод `findAllByState(String state)`, чтобы найти все города (`city`) в данном штате (`state`).

## ENTITIES

---

Сущности в JPA — это не что иное, как POJO, представляющие данные, которые могут быть сохранены в базе данных. Сущность представляет собой таблицу, хранящуюся в базе данных. Каждый экземпляр сущности представляет собой строку в таблице.

## JPA'S PERSISTENCE CONTEXT

---

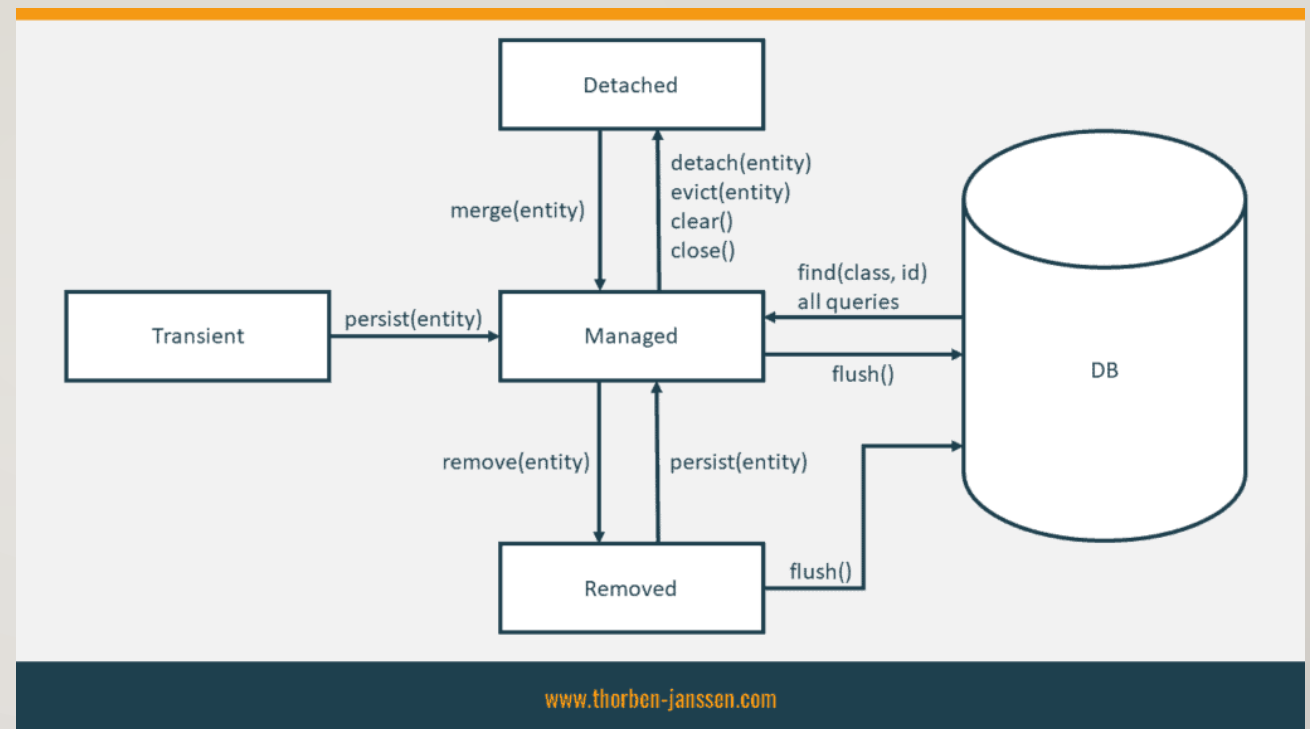
**Persistence Context** — одна из основных концепций JPA. Вы можете думать об этом как о наборе всех объектов сущностей, которые вы использовали в своем Use Case (пользовательском сценарии). Каждый из них представляет собой запись в базе данных.

Но напрямую мы не работаем с Persistence Context. Для этого мы используем Entity Manager или "менеджер сущностей". Именно он знает про контекст и про то, какие там живут сущности. Мы же взаимодействуем с Entity Manager'ом.



## JPA'S 4 LIFECYCLE STATES

Модель жизненного цикла состоит из 4 состояний: transient(переходный), managed(управляемый), removed(удаленный), and detached(отсоединенный)



# JPA'S 4 LIFECYCLE STATES

---

## 1. Transient

Состояние жизненного цикла вновь созданного объекта сущности называется переходным . Сущность еще не была сохранена, поэтому она не представляет никакой записи базы данных.

```
Student student = new Student();  
student.setFirstName("Andrew");  
student.setLastName("Brown");
```



## JPA'S 4 LIFECYCLE STATES

---

### 2. Managed

Все сущностные объекты, присоединенные к текущему контексту постоянства, находятся в управляемом состоянии жизненного цикла . Это означает, что ваш поставщик постоянства, например Hibernate, обнаружит любые изменения в объектах и сгенерирует необходимые операторы SQL INSERT или UPDATE, когда он сбрасывает persistence context.

```
Student student = new Student();  
student.setFirstName("Andrew");  
student.setLastName("Brown");  
em.persist(student);
```

## JPA'S 4 LIFECYCLE STATES

---

### 3. Detached

Сущность, которая ранее управлялась, но больше не привязана к текущему persistence context, находится в состоянии жизненного цикла detached .

Сущность отсоединяется, когда вы закрываете persistence context. Обычно это происходит после обработки запроса. Затем транзакция базы данных фиксируется, persistence context закрывается, и объект сущности возвращается вызывающей стороне. Затем вызывающая сторона извлекает объект сущности в состоянии жизненного цикла detached .

Вы также можете программно отсоединить сущность, вызвав метод detach в EntityManager .

```
em.detach(student);
```

## JPA'S 4 LIFECYCLE STATES

---

### 4. Removed

Когда вы вызываете метод удаления в вашем EntityManager , сопоставленная запись базы данных не удаляется немедленно. Сущностный объект только меняет свое состояние жизненного цикла на remove .

Во время следующей операции сброса Hibernate сгенерирует оператор SQL DELETE для удаления записи из таблицы базы данных.

```
em.remove(student);
```