

Advanced Topics in JPA/Spring

Miroslav Blaško

blaskmir@fel.cvut.cz

Winter Term 2018



Contents

1 JPA

2 Spring

3 Task



JPA



Criteria API example

```
SELECT p FROM Product p WHERE p.name LIKE '%p%'
```

Static Metamodel

```
CriteriaBuilder cb = em.  
    getCriteriaBuilder();  
CriteriaQuery<Product> cq = cb.  
    createQuery(Product.class);  
Root<Product> r = cq.from(Product.  
    class);  
cq.where(  
    cb.like(  
        r.get(Product_.name)  
        , "%p%")  
    );  
return em.createQuery(cq).  
    getResultList();
```

Metamodel

```
Metamodel m = em.getMetamodel();  
CriteriaBuilder cb = em.  
    getCriteriaBuilder();  
CriteriaQuery<Product> cq = cb.  
    createQuery(Product.class);  
Root<Product> r = cq.from(Product.  
    class);  
cq.where(  
    cb.like(  
        r.get(  
            m.entity(Product.class)  
                .getSingularAttribute("name",  
                String.class))  
            , "%p%")  
    );  
return em.createQuery(cq).  
    getResultList();
```



Criteria API - main classes

- 1 **CriteriaBuilder used to construct**
 - criteria queries, e.g. `cb.createQuery(User.class)`
 - expressions, e.g. "test that concrete student is member of studygroup"
– `cb.isMember(student, studygroup.get("students"))`
 - predicates, e.g. `cb.and(expression1, expression2)`
- 2 **CriteriaQuery used to define top-level structure of query such as**
 - `Root<User> = cq.from(User.class)`
 - `cq.select(..).where(..)`
- 3 **Root – root type in the from clause that references an entity. It can be used within**
 - expressions e.g. `dog.get(Dog_.color).in("brown", "black")`
 - selections e.g. `q.select(dog.get("color"))`
 - ...



Spring



Spring Data Repositories

Customer repository example

```
interface CustomerRepository extends JpaRepository<Customer, Long> {  
  
    Customer findByEmailAddress(String emailAddress);  
  
    List<Customer> findByLastname(String lastname, Sort sort);  
  
    Page<Customer> findByFirstname(String firstname, Pageable pageable);  
}
```

Example configuration

```
@EnableJpaRepositories("cz.cvut.kbss.ear.eshop.dao")  
public class PersistenceConfig {  
    ...  
}
```



Specifications – reusable predicates

Specification interface

```
interface Specification<T> {
    Predicate toPredicate(
        Root<T> root,
        CriteriaQuery query,
        CriteriaBuilder cb);
}
```

Definition of a repository

```
interface CustomerRepository
extends
    JpaRepository<Customer>,
    JpaSpecificationExecutor
{
    // Your query methods here
}
```

Implementation of specifications

```
public static Specification<Customer>
    customerHasBirthday() {
    return new Specification<Customer> {
        public Predicate toPredicate(Root<T>
            > root, CriteriaQuery query,
            CriteriaBuilder cb) {
            return cb.equal(root.get(
                Customer_.birthday), today);
        }
    };
}
```

Usage of repository

```
customerRepo.findAll(hasBirthday());
customerRepo.findAll(isLongTermCustomer
    ());
```



Task



Task 1 – ProductDao Using Criteria API

Check out branch *b181-seminar-11-task* of the e-shop project (<https://gitlab.fel.cvut.cz/ear/b181-eshop>).

- Uncomment in `pom.xml` relevant part that provides static metamodel generation
- Implement both functions of `ProductDao` using Criteria API
- *Hint:* Use `ProductDaoTest` to debug your implementation
- **Acceptance criteria:** `ProductDao` uses Criteria API with static metamodel and all tests pass.



Task 2 – ProductDao using Spring Data JPA

- Implement ProductDao using JpaRepository or JpaSpecificationExecutor
- **Acceptance criteria:** All tests pass.



Resources

- <https://docs.oracle.com/javaee/6/tutorial/doc/gjrij.html>
- <https://www.baeldung.com/spring-data-criteria-queries>
- <https://spring.io/blog/2011/04/26/advanced-spring-data-jpa-specifications-and-queryd>

