🏷️

라이브 세션-2022.09.08(수): 슬라이스 테스트 실습

✅ 아바타 실습(10분)

Spring Data JPA 실습 리뷰

1️⃣ 주문한 커피가 Response에 포함되는 기능

- 코드로 리뷰

2️⃣ Auditing

- Auditing(감사)이란?
  - 어떤 대상을 조사하는것
- Spring Data JPA에서는?
  - 엔티티의 생성일, 수정일, 작성자 등을 투명하게 추적할 수 있는 기능
  - 생성일, 수정일, 작성자 등을 각 엔티티에서 제거함으로써 공통화 해준다.
  - 개발자가 직접 필드 데이터를 입력할 필요 없음.
- 사용하는 애너테이션
  - `@EnableJpaAuditing`
    - JPA Auditing 기능을 활성화 해준다.
    - `@Configuration` 설정이 된 클래스에 추가해주면 됨.
  - `@MappedSuperclass`

        - 상위 클래스의 필드를 엔티티 필드로 매핑해 준다.
        - 즉 상위 클래스에 정의된 필드들이 엔티티 클래스의 컬럼 매핑 대상 필드가 된다.
    - `@EntityListeners(AuditingEntityListener.class)`
        - 엔티티에서 등록, 수정 등의 이벤트를 리스닝한다.
    - `@CreatedDate`
        - 엔티티를 DB에 INSERT시, 생성일시 자동 저장
    - `@LastModifiedDate`
        - 엔티티를 DB에 UPDATE시, 수정일시 자동 저장
    - `@CreatedBy`
        - 엔티티를 DB에 생성한 사람을 저장
        - `AuditorAware` 인터페이스 구현 클래스가 Bean에 등록되어야 한다.

3️⃣ CASCADE 옵션

- 연관 관계가 맺어진 엔티티에 대한 연쇄적인 작업
    - `CascadeType.PERSIST`
        - 엔티티 저장 시, 주 테이블 데이터 저장 → 자식 테이블 데이터 저장
    - `CascadeType.REMOVE`
        - 엔티티 삭제 시, 자식 테이블 데이터 삭제 → 주 테이블 데이터 삭제
    - `CascadeType.MERGE`
        - 연관된 엔티티 변경 시, 연관된 테이블까지 업데이트

4️⃣ FETCH 옵션

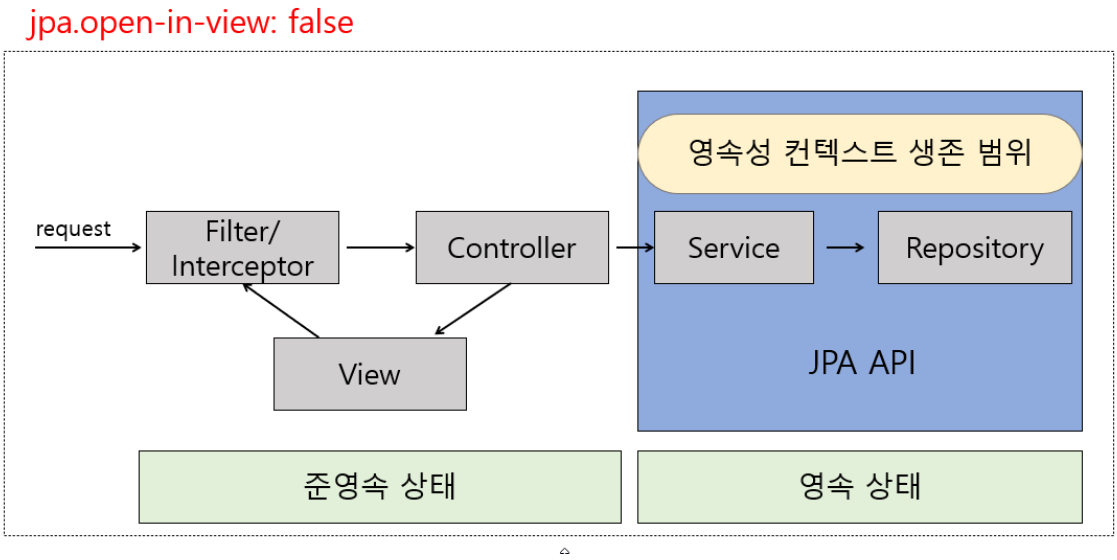
- 연관 관계 매핑이 되어 있는 엔티티의 데이터를 가져오는 시점
    - `FetchType.LAZY`
        - 실제 호출 시점에 DB에서 조회
    - `FetchType.EAGER`

- 연관된 엔티티를 JOIN을 통해 한꺼번에 조회

5️⃣ Spring Data JPA에서의 영속성 컨텍스트(Persistence Context) 종료 시점

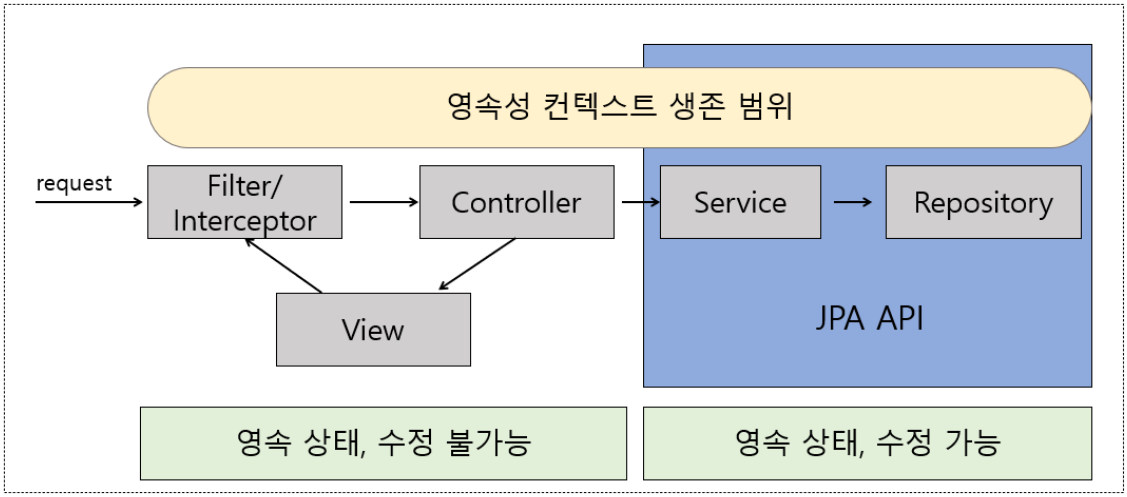
- Lazy 로딩을 JPA API 사용 바깥까지 허용하는 것.(default: true)
- false로 하면 서비스 바깥 쪽 영역까지 Lazy 로딩을 허용하지 않는다.

- `jpa.open-in-view: false`



- `jpa.open-in-view: true`

jpa.open-in-view: true



6️⃣ DTO 리팩토링

- DTO 클래스가 많아지는 것을 최소화 할 수 있는 방법
- DTO 클래스 자체를 하나의 클래스 안에서 static class 멤버로 취급
- 코드로 설명

7️⃣ CustomBeanUtils를 이용한 updateMember() 메서드 리팩토링

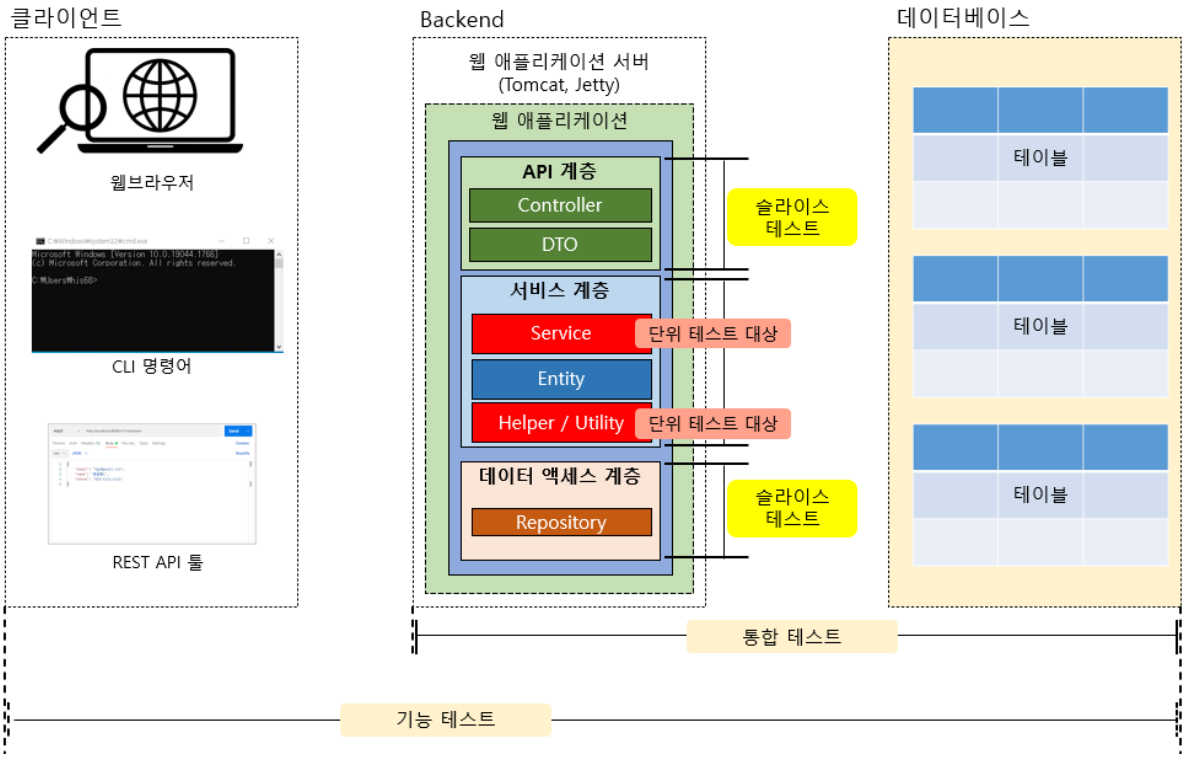
- Spring에서 제공하는 `BeanWrapper`를 이용하면 객체 간에 필드를 복사할 수 있다.
- 코드로 설명

✅ 단위 테스트 리뷰

1️⃣ 테스트를 해야 되는 이유

- 잘 알고 계실거라 생각합니다. ^^

2️⃣ 애플리케이션의 테스트 분류



- DB를 사용하면 단위 테스트이다 아니다?

3️⃣ 테스트 케이스(Test Case)란?

- 테스트를 위한 입력 데이터, 실행 조건, 기대 결과를 표현하기 위한 명세
- 메서드 등 하나의 단위를 테스트하기 위해 작성하는 테스트 코드

4️⃣ Assertion(어써션)이란?

- **예상하는 결과 값이 참(true)이길 바라는 것**

5️⃣ 단위 테스트를 위한 F.I.R.S.T 원칙

- **Fast(빠르게)**
- **Independent(독립적으로)**
- **Repeatable(어떤 환경에서도 반복 가능하도록)**
  - 환경이 달라도 반복해서 동일하게 동작
- **Self-validating(셀프 검증이 되도록)**
  - Assertion
- **Timely(시기 적절하게)**
  - 기능 구현을 하기 직 전에 작성(TDD에서는 그렇게 한다.)
  - 기능 구현 후 작성하더라도 일부만 구현 되면 부분적으로 단위 테스트 하는게 좋다

6️⃣ 단위 테스트 실습 리뷰

- `StampCalculator`
- `RandomPasswordGenerator`
- 코드로 확인

✅ Hamcrest

- JUnit 기반의 단위 테스트에서 사용할 수 있는 Assertion Framework

- **매쳐(Matcher)**가 자연스러운 문장으로 이어진다.
- 테스트 실패 메시지를 이해하기 쉽다

```
assertThat(actual, is(equalTo(expected)))
```

- 예외에 대한 테스트가 조금 애매.
    - JUnit Assertion 메서드와 함께 사용

✅ 슬라이스 테스트

1️⃣ API 계층 테스트 리뷰

- Controller 테스트
- MockMvc 이용
- `@SpringBootTest`
    - 전체 빈 Application Context에 등록
    - 가장 손쉽게 테스트 가능
    - 필요 없는 부분까지 다 불러온다
    - 상대적으로 무거움
    - `@AutoConfigureMockMvc` 와 함께 사용
- `@WebMvcTest`
    - 테스트에 필요한 빈만 Application Context에 등록
    - 테스트에 필요한 빈들을 수동으로 등록 필요.
    - 상대적으로 가벼움
    - 자동 구성을 안해주므로 추가적인 설정이 필요할 수 있다.

2️⃣ Controller 테스트 클래스 기본 구조 리뷰

- 코드로 설명

3️⃣ MemberController postMember() 테스트 케이스 리뷰

- 코드로 설명

4️⃣ 데이터 액세스 계층 테스트 리뷰

- 데이터 액세스 계층을 테스트 규칙
  - **DB의 상태를 테스트 케이스 실행 이전으로 되돌려서 깨끗하게 만든다.**

5️⃣ Repository 테스트 리뷰

- `@DataJpaTest` 애너테이션 사용
  - 데이터 액세스 계층과 관련된 의존성을 자동 구성 해준다.
  - `@Transactional` 을 포함하고 있음
  - 테스트 케이스 실행 끝나면 rollback 됨
- MemberRepositoryTest
  - 코드로 리뷰

✅ 슬라이스 테스트 실습 리뷰

- `MemberControllerHomeworkTest`
  - 코드로 확인

추석 연휴 잘 보내세요!! 아프지 마시구요!!