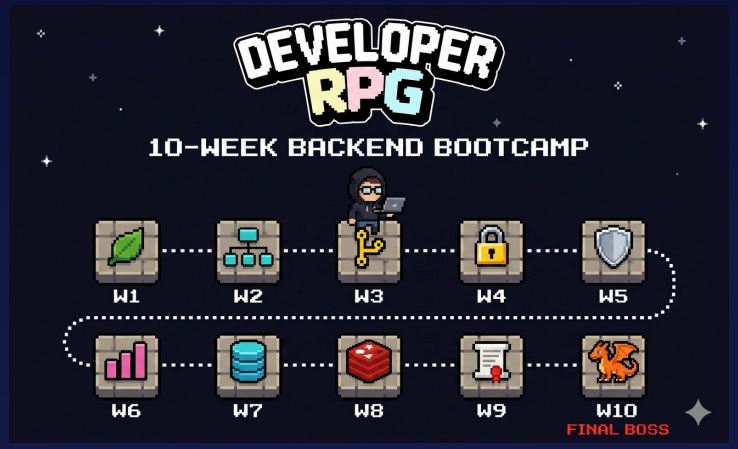

Week 2

JPA 연관관계 / 영속성 컨텍스트

2026-06-13 (토) · 미션 공개 + 주간 방향

JPA? 그게 뭔데 🤔

지난 주 돌아보기 — Week 1

항목	결과
제출률	{{N}}/{{M}} 명 ({{X%}})
5축 평균	{{a}}점 / 5점
mission-guard 통과율	{{Y%}}
잘된 점	3계층 분리 근거를 말로 푼 학생이 많음
아쉬운 점	`@Transactional` 위치 근거가 빈약 (그냥 Service 위)
이번 주 가져갈 것	왜 그 위치냐를 evidence 에 박는 습관

⚔ QUEST 03-week2-jpa

- **type**: `code` · 마감: 2026-06-19 (금) `23:59`
- **통과 조건**: 리팩토링 PR 머지 + 피어리뷰 완료

> *"Week 1 코드를 그대로 이어 받는다."*

이번 쭈 목표 🎯

1. **연관관계 주인** 본인이 결정 — 양방향 매핑에서 어느 쪽 이 FK 를 가지는지
2. **모든 연관관계 LAZY** + N+1 발생 현장 쿼리 로그 로 확인
3. **변경 감지** 활용 1건 — `setter` 만 호출하고 `save()` 안 부르기

> *게이트 완화: N+1 못 잡아도 fetch join / `@EntityGraph` 차이 를 SQL 로그로 설명하면 통과.*

시작하는 법 — Week 1 코드 그대로

```
# Week 1 폴더의 project/ 를 Week 2 로 복사
cp -r 02-week1-spring-boot/project 03-week2-jpa/project

# JPA 의존성 + H2/MySQL 추가, 메모리 저장소 → JpaRepository
```

> *이게 Week 4~7 starter 운영 원칙과 같은 흐름.* 환경 세팅 반복 금지.

핵심 개념 1 — 영속성 컨텍스트 4 기능

EntityManager 가 캐시 처럼 동작.

기능	효과
1차 캐시	동일 PK 재조회 시 SELECT 안 감
변경 감지	flush 때 바뀐 Entity 만 UPDATE
쓰기 지연	INSERT/UPDATE 모아서 flush 시
동일성 보장	`==` 비교가 같은 객체 보장

각 기능을 evidence/n1-detection-guide.md 에서 본인 말로.

핵심 개념 2 — 연관관계 주인

`Post` 1 : N `Comment` — 누가 FK?

```
// N 쪽이 주인 (FK 가짐)
@ManyToOne(fetch = LAZY)
@JoinColumn(name = "post_id")
private Post post;

// 거울 쪽 (mappedBy)
@OneToMany(mappedBy = "post")
private List<Comment> comments;
```

핵심 개념 3 — N+1 와 fetch join

```
-- N+1 : 1 + N 번
SELECT * FROM post;                    -- 1
SELECT * FROM comment WHERE post_id=?; -- N

-- fetch join : 1 번
SELECT p, c FROM Post p
LEFT JOIN FETCH p.comments c;
```

`spring.jpa.show-sql=true` 로 직접 확인하고 캡처.

핵심 개념 4 — 변경 감지

`save()` 호출 안 해도 트랜잭션 종료 시 UPDATE.

```
@Transactional
public void updateTitle(Long id, String title) {
  Post post = postRepo.findById(id).orElseThrow();
  post.changeTitle(title);  // setter 호출만
  // save() 호출 X — 변경 감지가 처리
}
```

evidence/dirty-checking-snapshot.md 에 flush 전후 SQL 로그 캡처.

이거 하면 망함 ⚠️

- ❌ 양방향 매핑에서 양쪽 다 setter → ✅ 편의 메서드 1개로 양쪽 동기화
- ❌ EAGER 로 처음부터 다 가져옴 → ✅ LAZY 기본, fetch join 으로 케이스별 해결
- ❌ N+1 보고 그냥 EAGER 로 도망 → ✅ fetch join 또는 `@EntityGraph`
- ❌ `setter` 안 부르고 `save(new Post(...))` 로 전체 교체 → ✅ 변경 감지 활용

이번 주 제출할 것 📦

```
03-week2-jpa/
├── report.md
├── project/                              # Week 1 코드 이어 받기
└── evidence/
    ├── entity-design-notes.md            # Entity 설계 결정
    ├── association-owner-decision.md     # 연관관계 주인 근거
    ├── n-plus-one-before.md              # N+1 SQL 로그
    ├── n-plus-one-after.md               # fetch join 후 로그
    ├── dirty-checking-snapshot.md        # 변경 감지 로그
    └── n1-detection-guide.md             # 학습 보조 (main)
```

평가는 어떻게 (5축) 

축	가중	핵심
요구사항 충족	★★	연관관계 / LAZY / 쿼리 로그
구조	★★	Entity 설계 / 편의 메서드
기술 적용	★★★	연관관계 주인 결정 근거
검증 근거	★★★	쿼리 로그 before/after
설명력	★★	N+1 또는 fetch join 차이 설명

> *Week 2 는 기술 포인트 적용 가중. SQL 로그 한 장이 핵심 evidence.*

피어리뷰 (1차) — 이번 주부터

- 같은 코호트 학생 2명 PR 을 읽고 1개씩 코멘트
- 코멘트 양식: `잘된 점 1줄 / 개선 1줄` (질문도 OK)
- 톤: 비판적이지만 격려 — 5점 만점 시 5점, 비웃기 시 0점
- 점수 룰: 참여 OK `+5`, 미참여 `0` (`submissions.md` 260–264)
- 본인 PR 에 받은 코멘트 1개 이상 반영 → `evidence/review-response.md`

> *피어리뷰는 본인 코드 보는 눈 이 길러지는 가장 빠른 방법.*

이번 주 일정 

- **제출 마감**: 2026-06-19 (금) `23:59`
- **토 15:00–16:30**: 격주 **강의** — JPA 연관관계 / 영속성 (직후 슬롯)
- **오피스아워**: 화·목 `21:00` `{cohort}-질문` 채널 스레드
- **PR / AI 리뷰 알림**: `{cohort}-리뷰` 채널 자동 알림
- **TIL**: 매일 한 줄 `{cohort}-til` 채널 (학습 로그 누적)

질문 ㄱㄱ ?

오늘 15:00 — 도메인 모델링 강의 (게시판 5초 vs 5시간)
다음 주 토 14:00 — Week 3 (이력서 스토리라인) + 첫 학생 발표

잠깐 쉬고 15:00 에 다시 뵐게요.

*다음 주: **Week 3 — 백엔드 이력서 차별화** + 학생 내부 발표 시작.*