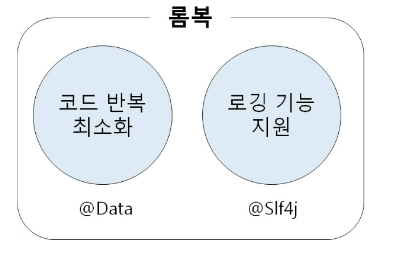
롬복과 리팩터링

Lombok : 코드를 간소화해주는 라이브러리를 의미. 롬복을 사용하면 코드를 간편하게 작성이 가능하다. 또한 로깅 기능을 통해 println()문을 개선할 수 있다. 로깅이란 프로그램의 수행 과정을 기록을 남기는것을 의미

롬복의 기능



리젝터링 : 결과의 변경 없이 코드의 구조 또는 성능을 개선하는 작업을 의미

롬복을 활용하여 리팩터링 진행하기

build.gradle에

compileOnly 'org.projectlombok:lombok'  
annotationProcessor 'org.projectlombok:lombok'

이 두 코드를 작성 (의존성 안에)

DTO 리팩터링 하기

1. ArticleForm() 생성자 간소화과정

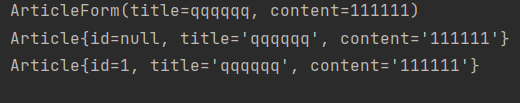
1) ArticleFrom() 생성자 코드 전체를 지운다

2) ArticleForm 클래스 위에 @ AllArgsConstructor 어노테이션을 추가한다. 이러면 클래스 안쪽의 모든 필드, 즉 title과 content를 매개변수로 하는 생성자가 자동으로 만들어진다.

2. toString() 메서드 간소화과정 진행

1) toString()메서드 코드 전체를 지우기

2) @ToString 어노테이션을 추가



toString 메소드가 잘 나오는것을 확인!

엔티티 리펙토링 진행하기

1. Article.java 코드에서 Article() 생성자와 toString() 메소드를 삭제

2. 삭제할 코드를 대체할 어노테이션 작성

1) @AllArgsConstructor

2) @ToString

컨트롤러에 로그를 남기기

println()문으로 데이터를 검증할 시 기록에 남지도 않을뿐더러 서버의 성능에 악영향을 미치기에 이를 개선할 필요성이 존재

이를 위해 로깅 기능을 사용하여 대체하기

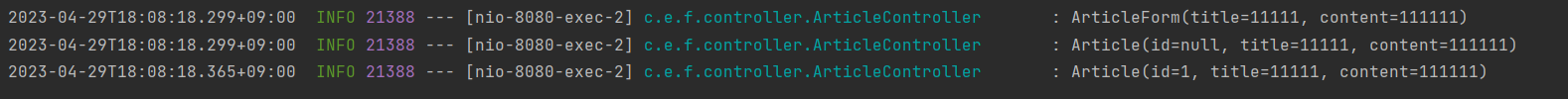
1. ArticleController 클래스 위에 @Slf4j 어노테이션을 추가 Simple Logging Facade for Java 이를 통해 로깅을 사용가능

2. println()문 대체하기

1) System문 주석,

2) log.info(form.toString)문을 작성

3) 나머지 모두 이런식으로 작성



이렇게 시간과 함께 로그 기록이 남는것을 확인할 수 있다. 이전 println()문은 시간이 남지 않았었음

1분퀴즈

- 롬복을 설치하기 위해 값 변경이 필요한 파일 : build.gradle

- 모든 필드를 매개변수로 하는 생성자를 만드는 롬복 어노테이션 : @AllArgsConstructor

- toString()메소드를 대체하는 롬복 어노테이션 : @ToString

- 로깅 기능을 사용하기 위해 필요한 롬복 어노테이션 : @Slf4j

마무리하기

1. 롬복 : 코드를 간소화 해주는 라이브러리이다. 롬복을 사용하면 여러 필수 코드가 반복 되는 것을 최소화할 수 있고, println() 문을 로깅 기능으로 대체할 수 있습니다.

2. 로깅 : 프로그램의 수행 과정을 기록으로 남기는 것을 말합니다. 일종의 자동차 블랙박스

3. 리팩터링 : 코드의 기능에는 변함이 없이 코드의 구조 또는 성능을 개선하는 작업을 말합니다. 리팩터링을 하면 코드의 가독성이 좋아지고, 길이도 짧아져서 개발시간 단축!

4. @AllArgsConstructor : 클래스 안쪽의 모든 필드를 매개변수로 하는 생성자를 만드는 어노테이션으로, @AllArgsConstructor을 사용하면 클래스 내 별도의 생성자를 만들지 않아도 된다!

5. @ToString : toString()메소드를 사용한것과 똑같은 효과를 나타냄

6. @Slf4j : Simple Logging Facade for java의 약자로 로깅을 할 때 사용된다. 로깅 기능으로 로그를 찍으면 나중에라도 그동안 찍힌 로그를 찾아볼 수 있다. 로그를 찍을때는 log.info()문을 사용!