

# 4장 롬복과 리팩터링

출처: 코딩 자율학습 스프링부트3 자바 백엔드 개발 입문, 홍팍, 길벗, 2023

# 롬복(Lombok)이란

- 자바 개발을 하다 보면, getter, setter, 생성자, `toString()` 등과 같은 필수 코드들을 만들기 위해 반복 작업을 해야 할 때가 많음
- 룸복은 이러한 반복 작업을 없애 주고, 코드를 간소하게 해주는 라이브러리
- 자바 어노테이션 기반
  - `@Getter`, `@Setter`, `@ToString`
  - `@NoArgsConstructor`, `@AllArgsConstructor`, `@Slf4j`, ...
- 룸복의 기능
  - 반복적인 코드 자동 생성
    - ✓ 코드 반복 최소화, 코드 간결화
    - ✓ 필수 코드들을 간편하게 작성
    - ✓ 가독성 증가
    - ✓ 생산성 향상
  - 로깅(logging) 기능 지원
    - ✓ 로깅: 프로그램의 수행 과정을 기록으로 남기는 것

# 리팩토링(Refactoring)

- 코드의 기능은 변경 없이 코드의 구조 또는 성능을 개선하는 작업
  - 버그를 없애거나 새로운 기능의 추가하는 것이 아님
- 코드의 품질 향상과 생산성 향상이 목적
- 리팩토링의 효과
  - 가독성 향상
    - ✓ 코드가 이해하기 쉬워짐
  - 유지보수성 향상
    - ✓ 코드 변경시 영향 범위의 최소화
  - 테스트 용이
    - ✓ 단위 테스트 작성이 용이
  - 재사용성 향상
    - ✓ 모듈화된 구조로 재사용성 향상
  - 버그 예방
    - ✓ 구조적 개선으로 오류 감소

# 롬복을 활용해 리팩터링하기

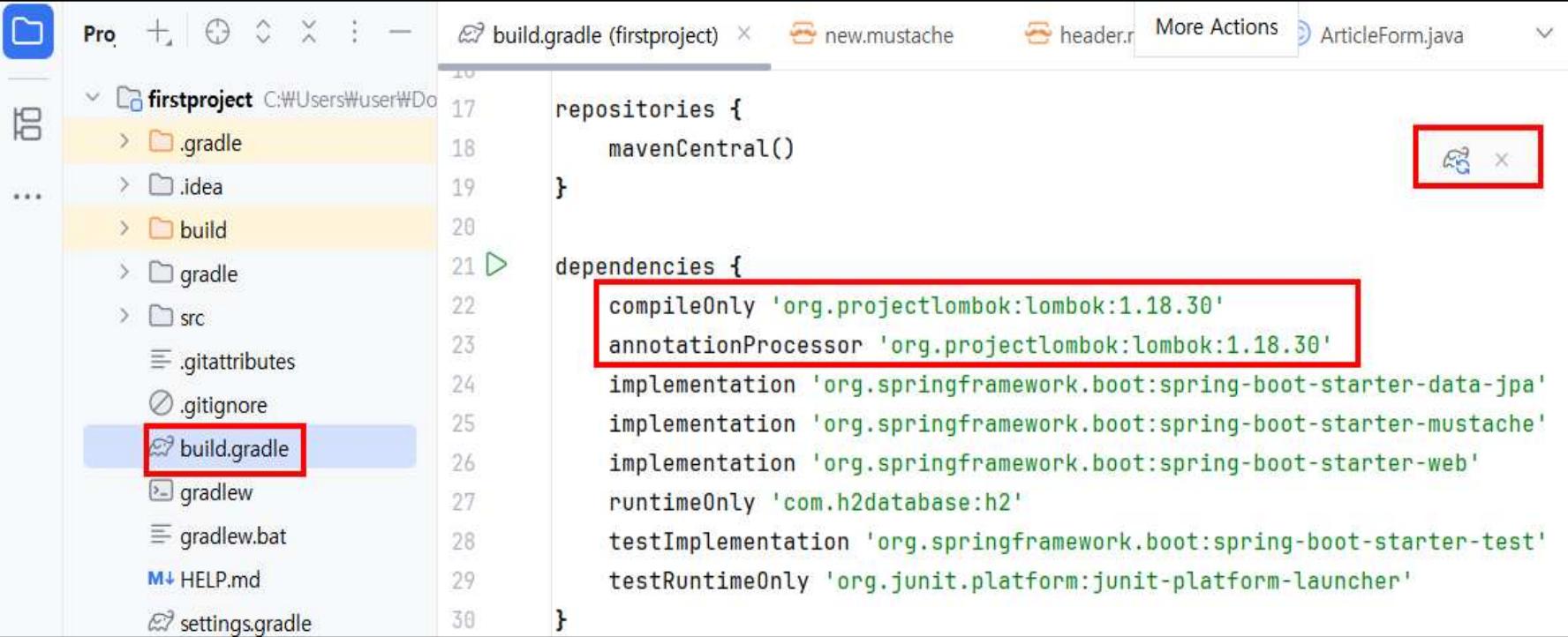
- 룸복을 사용해 DTO, 엔티티를 리팩터링하기
- 로깅을 사용해 컨트롤러의 `println()`문을 로깅 기능으로 대체

# 롬복을 활용해 리팩터링하기

- **롬복 설치하기**

- 프로젝트 탐색기에서 [firstproject – src] 디렉토리에서 build.gradle 파일을 열기
- build.gradle 파일 수정 (교재 140페이지 수정)
  - ✓ compileOnly 'org.projectlombok:lombok:1.18.30'  
annotationProcessor 'org.projectlombok:lombok:1.18.30'
-  (Sync Gradle Changes) 클릭, 상태 표시줄에 다운로드 현황 및 완료

교재 140페이지  
수정



```
build.gradle (firstproject)
repositories {
    mavenCentral()
}

dependencies {
    compileOnly 'org.projectlombok:lombok:1.18.30'
    annotationProcessor 'org.projectlombok:lombok:1.18.30'

    implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
    implementation 'org.springframework.boot:spring-boot-starter-mustache'
    implementation 'org.springframework.boot:spring-boot-starter-web'
    runtimeOnly 'com.h2database:h2'
    testImplementation 'org.springframework.boot:spring-boot-starter-test'
    testRuntimeOnly 'org.junit.platform:junit-platform-launcher'
}
```

# DTO 리팩터링

- 프로젝트 탐색기에서 [com.example.firstproject – dto – ArticleForm] 열기
- ArticleForm() 생성자 간소화
  - ArticleForm() 생성자 코드 전체 삭제
  - ArticleForm 클래스 선언문 윗줄에 @AllArgsConstructor 어노테이션 추가
    - ✓ 클래스에서 선언된 모든 필드들(title, content)를 매개변수로 하는 생성자가 자동으로 생성됨
- toString() 메소드 간소화
  - toString() 메소드 전체 삭제
  - ArticleForm 클래스 선언문 윗줄에 @ToString 어노테이션 추가
    - ✓ toString() 메서드가 자동으로 생성됨

→ ArticleForm 클래스의 코드가 간단해지고, 개발 시간 단축

# DTO 리팩터링

- 리팩터링된 ArticleForm 클래스

```
package com.example.firstproject.dto;

import com.example.firstproject.entity.Article;
import lombok.AllArgsConstructor;
import lombok.ToString;

@AllArgsConstructor
@ToString
public class ArticleForm {
    private String title;
    private String content;

    public Article toEntity() {
        return new Article(id: null, title, content);
    }
}
```

자동으로 import

새 어노테이션 추가

# DTO 리팩터링

- 서버 재실행
- localhost:8080/articles/new 접속
  - 새 글을 입력하고 [submit]

The screenshot illustrates a development environment where a DTO (Data Transfer Object) has been refactored. On the left, a browser window shows a form at `localhost:8080/articles/new`. The form has two input fields: '제목' (Title) containing 'aaaa' and '내용' (Content) containing '1111'. A red box highlights the 'Submit' button. An arrow points from this browser window to the right side of the screenshot.

The right side shows a 'Whitelabel Error Page' at `localhost:8080/articles/create`. The page states: "This application has no explicit mapping for /error, so you are seeing this as a fallback." It also shows the timestamp "Sun Sep 21 18:35:32 KST 2025" and the message "There was an unexpected error (type=Not Found, status=404)."

Below the error page is a terminal window titled "Run" showing the output of the application. The log entries are:

```
2025-09-21T18:35:23.173+09:00 INFO 15980 --- [nio-8080-exec-1] com.example.firstproject.FirstprojectApplication.main... : Started FirstprojectApplication in 1 min, 42 sec
2025-09-21T18:35:23.175+09:00 INFO 15980 --- [nio-8080-exec-1] com.example.firstproject.FirstprojectApplication.main... : Started FirstprojectApplication in 1 min, 31 sec
2025-09-21T18:35:23.179+09:00 INFO 15980 --- [nio-8080-exec-1] com.example.firstproject.FirstprojectApplication.main... : ArticleForm(title='aaaa', content='1111')
2025-09-21T18:35:23.179+09:00 INFO 15980 --- [nio-8080-exec-1] com.example.firstproject.FirstprojectApplication.main... : Article{id=null, title='aaaa', content='1111'}
2025-09-21T18:35:23.179+09:00 INFO 15980 --- [nio-8080-exec-1] com.example.firstproject.FirstprojectApplication.main... : Article{id=1, title='aaaa', content='1111'}
```

A red box highlights the last three log entries, which represent the creation of a new Article object with id=1, title='aaaa', and content='1111'.

# DTO 리팩터링

- H2 DB 웹 콘솔 로그인

```
firs 34 min, 39 sec > Q jdbc < Cc W .* 1/2 ↑ ↓ ⌂ :  
34 min, 29 sec : HikariPool-1 - Starting...  
: HikariPool-1 - Added connection conn0: url= jdbc:h2:mem:33f8764b-8ace-479b-a8e9-8ef41ecdaa19 user=SA  
: HikariPool-1 - Start completed.  
: H2 console available at '/h2-console'. Database available at 'jdbc:h2:mem:33f8764b-8ace-479b-a8e9-8ef41ecdaa19'  
: HHH000204: Processing PersistenceUnitInfo [name: default] CTRL + C  
: HHH000412: Hibernate ORM core version 6.2.2.Final
```

- localhost:8080/h2-console 접속

localhost:8080/h2-console/login.jsp?jsessionid=ed0ee2916e

English Preferences Tools Help

Login

Saved Settings: Generic H2 (Embedded)

Setting Name: Generic H2 (Embedded) Save Remove

Driver Class: org.h2.Driver

JDBC URL: **jdbc:h2:mem:33f8764b-8ace-479b-a8e9-8ef41ecdaa19**

User Name: sa **CTRL + V**

Password:

Connect Test Connection

Article 테이블에 데이터 삽입 확인

Auto commit Max rows: 1000 Auto complete

jdbc:h2:mem:33f8764b-8ace-479b-a8e9-8ef41ecdaa19

ARTICLE (1)

INFORMATION\_SCHEMA

Sequences

Users

H2 2.1.214 (2022-06-13)

Run Run Selected Auto complete Clear SQL statement:

SELECT \* FROM ARTICLE;

SELECT \* FROM ARTICLE;

ID	CONTENT	TITLE
1	1111	aaaa

(1 row, 3 ms)

# 엔티티 리팩터링하기

- 프로젝트 탐색기에서 [com.example.firstproject – entity – Article] 열기
- Article 클래스에서 생성자와 `toString()`을 삭제 후, `@AllArgsConstructor`, `@ToString` 어노테이션 추가

```
package com.example.firstproject.entity;

import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.Id;
import lombok.AllArgsConstructor;
import lombok.ToString;

@Entity
public class Article {
    @Id
    @GeneratedValue
    private Long id;
    @Column
    private String title;
    @Column
    private String content;
}
```

# 컨트롤러에 로그 남기기

- 프로젝트 탐색기에서 [com.example.firstproject-controller-ArticleController] 열기
- 서버의 `println()`문으로 데이터를 검증하면 기록에 남지 않아 나중에 다시 찾아 볼 수 없고, 서버의 성능에도 악영향 → 로깅 기능 사용
  - 로깅 기능을 사용하여 서버에서 일어나는 일을 모두 기록
- 룸복의 `@Slf4j` 사용하여 로깅
  - Slf4j(simple Logging Façade for java)
  - ArticleController.java에 `@Slf4j` 추가, `log.info("문자열")`추가

# 컨트롤러에 로그 남기기

ArticleController.java

```
import lombok.extern.slf4j.Slf4j;
(중략)

@Slf4j
@Controller //컨트롤러 선언
public class ArticleController {
    (중략)

    @PostMapping("/articles/create")
    public String createArticle(ArticleForm form){
        log.info(form.toString());
        //System.out.println(form.toString());
        // 1. DTO를 엔티티로 변환
        Article article = form.toEntity();
        log.info(article.toString());
        //System.out.println(article.toString());
        // 2. 리파지터리로 엔티티를 DB에 저장
        Article saved = articleRepository.save(article);
        log.info(saved.toString());
        //System.out.println(saved.toString());
        return "";
    }
}
```

# 컨트롤러에 로그 남기기

- 서버 재실행
- localhost:8080/articles/new 접속
  - 새 글을 입력하고 [submit]

The screenshot shows two windows side-by-side. On the left is a 'localhost:8080/articles/new' page with a form containing 'aaaa' in the title field and '1111' in the content field. A red box highlights the 'Submit' button. On the right is a 'Whitelabel Error Page' from 'localhost:8080/articles/create'. The page displays the message: 'This application has no explicit mapping for /error, so you are seeing this as a fallback.' Below it shows the timestamp 'Sun Sep 21 18:35:32 KST 2025' and the error message 'There was an unexpected error (type=Not Found, status=404)'. A red arrow points from the 'Submit' button on the left to the error page on the right.

- 로그로 찍은 결과 확인

```
2025-09-22T15:52:42.389+09:00 INFO 25744 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2025-09-22T15:52:42.435+09:00 INFO 25744 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 46 ms
2025-09-22T15:52:51.206+09:00 INFO 25744 --- [nio-8080-exec-2] c.e.f.controller.ArticleController : ArticleForm(title=aaaa, content=1111)
2025-09-22T15:52:51.207+09:00 INFO 25744 --- [nio-8080-exec-2] c.e.f.controller.ArticleController : Article(id=null, title=aaaa, content=1111)
2025-09-22T15:52:51.406+09:00 INFO 25744 --- [nio-8080-exec-2] c.e.f.controller.ArticleController : Article(id=1, title=aaaa, content=1111)
```