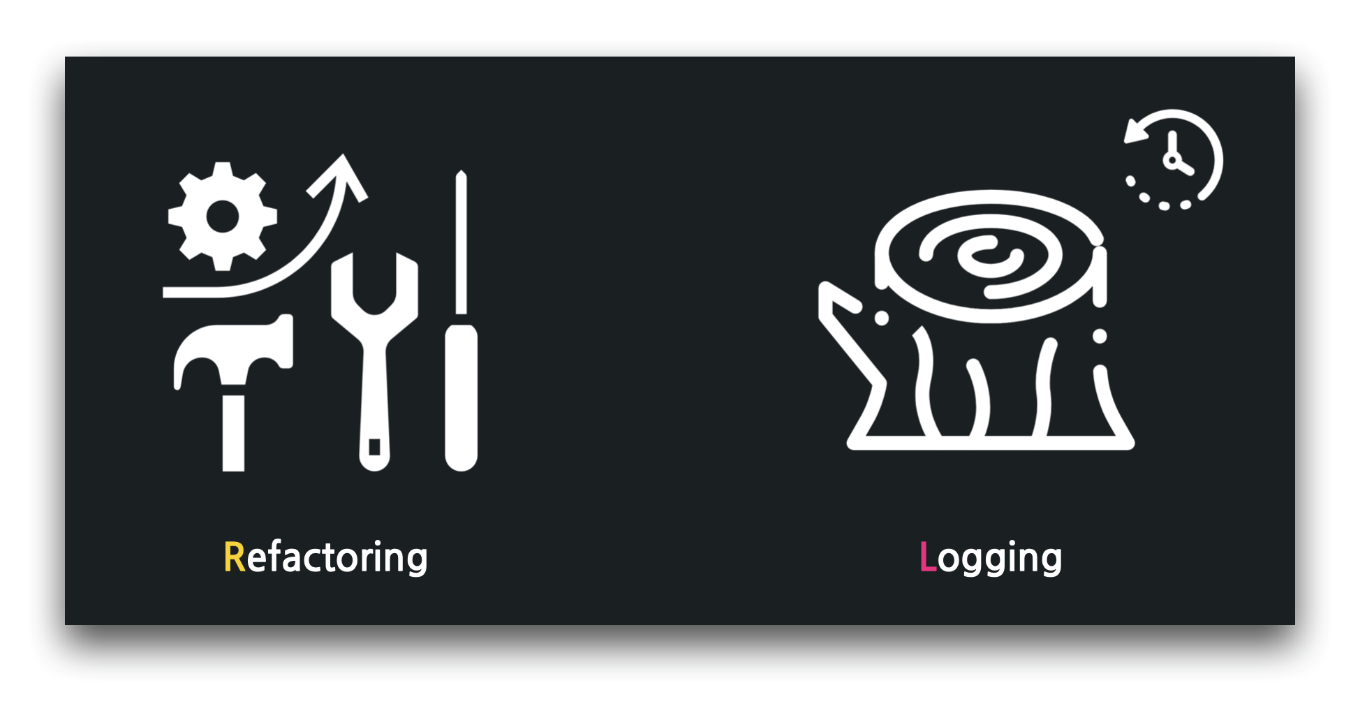
10 롬복(lombok)과 리팩터링(refactoring)

**롬복과 리팩터링**

**미션**

롬복을 활용하여 기존 코드를 리팩터링 하시오.  


**롬복 라이브러리 추가**

firstproject/build.gradle

1. plugins {
2. id 'org.springframework.boot' version '2.6.6' // 스프링 부트 버전
3. id 'io.spring.dependency-management' version '1.0.11.RELEASE'
4. id 'java'
5. }
6. group = 'com.example'
7. version = '0.0.1-SNAPSHOT'
8. sourceCompatibility = '17' // JDK 버전
9. repositories {
10. mavenCentral()
11. }
12. dependencies {
13. // 10강: 롬복 추가
14. compileOnly 'org.projectlombok:lombok'
15. annotationProcessor 'org.projectlombok:lombok'
16. implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
17. implementation 'org.springframework.boot:spring-boot-starter-mustache'
18. implementation 'org.springframework.boot:spring-boot-starter-web'
19. runtimeOnly 'com.h2database:h2'
20. testImplementation 'org.springframework.boot:spring-boot-starter-test'
21. }
22. test {
23. useJUnitPlatform()
24. }

⚠️ 최신 IntelliJ는 롬복 플러그인을 자체적으로 포함합니다.

**리팩터링, 코드 개선하기**

../dto/ArticleForm

1. package com.example.firstproject.dto;
2. import com.example.firstproject.entity.Article;
3. import lombok.AllArgsConstructor;
4. import lombok.ToString;
5. @AllArgsConstructor
6. @ToString
7. public class ArticleForm {
8. private String title;
9. private String content;
10. public Article toEntity() {
11. return new Article(null, title, content);
12. }
13. }

../entity/Article

1. package com.example.firstproject.entity;
2. import lombok.AllArgsConstructor;
3. import lombok.ToString;
4. import javax.persistence.Column;
5. import javax.persistence.Entity;
6. import javax.persistence.GeneratedValue;
7. import javax.persistence.Id;
8. @Entity
9. @AllArgsConstructor
10. @ToString
11. public class Article {
12. @Id
13. @GeneratedValue
14. private Long id;
15. @Column
16. private String title;
17. @Column
18. private String content;
19. }

**로그 남기기**

../controller/ArticleController

1. package com.example.firstproject.controller;
2. import com.example.firstproject.dto.ArticleForm;
3. import com.example.firstproject.entity.Article;
4. import com.example.firstproject.repository.ArticleRepository;
5. import lombok.extern.slf4j.Slf4j;
6. import org.springframework.beans.factory.annotation.Autowired;
7. import org.springframework.stereotype.Controller;
8. import org.springframework.web.bind.annotation.GetMapping;
9. import org.springframework.web.bind.annotation.PostMapping;
10. @Controller
11. @Slf4j // 로깅을 위한 롬복 어노테이션
12. public class ArticleController {
13. @Autowired
14. private ArticleRepository articleRepository;
15. @GetMapping("/articles/new")
16. public String newArticleForm() {
17. return "articles/new";
18. }
19. @PostMapping("/articles/create")
20. public String createArticle(ArticleForm form) {
21. log.info(form.toString()); // println() 을 로깅으로 대체!
22. Article article = form.toEntity();
23. log.info(article.toString()); // println() 을 로깅으로 대체!
24. Article saved = articleRepository.save(article);
25. log.info(saved.toString()); // println() 을 로깅으로 대체!
26. return "";
27. }
28. }

**🔥 구글링 훈련하기**

* java lombok 사용법
* 리팩터링이란
* 로깅이란
* @AllArgsConstructor
* @ToString
* @Slf4j