## 13장 완성 코드

### test/…/service/ArticleServiceTest.java

**package** com.example.firstproject.service;  
  
**import** com.example.firstproject.dto.ArticleForm;  
**import** com.example.firstproject.entity.Article;  
**import** org.junit.jupiter.api.Test;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.boot.test.context.SpringBootTest;  
**import** org.springframework.transaction.annotation.Transactional;  
  
**import** java.util.ArrayList;  
**import** java.util.Arrays;  
**import** java.util.List;  
  
**import static** org.junit.jupiter.api.Assertions.\*;  
  
@SpringBootTest  
**class** ArticleServiceTest {  
 @Autowired  
 ArticleService **articleService**;  
  
 @Test  
 **void** index() {  
 *// 1. 예상 데이터* Article a = **new** Article(1L, **"가가가가"**, **"1111"**);  
 Article b = **new** Article(2L, **"나나나나"**, **"2222"**);  
 Article c = **new** Article(3L, **"다다다다"**, **"3333"**);  
 List<Article> expected = **new** ArrayList<Article>(Arrays.*asList*(a, b, c));  
  
 *// 2. 실제 데이터* List<Article> articles = **articleService**.index();  
  
 *// 3. 비교 및 검증  
 assertEquals*(expected.toString(), articles.toString());  
 }  
  
 @Test  
 **void** show\_성공\_존재하는\_id\_입력() {  
 *// 1. 예상 데이터* Long id = 1L;  
 Article expected = **new** Article(id, **"가가가가"**, **"1111"**);  
  
 *// 2. 실제 데이터* Article article = **articleService**.show(id);  
  
 *// 3. 비교 및 검증  
 assertEquals*(expected.toString(), article.toString());  
 }  
  
 @Test  
 **void** show\_실패\_존재하지\_않는\_id\_입력() {  
 *// 1. 예상 데이터* Long id = -1L;  
 Article expected = **null**;  
  
 *// 2. 실제 데이터* Article article = **articleService**.show(id);  
  
 *// 3. 비교 및 검증  
 assertEquals*(expected, article);  
  
 }  
  
 @Transactional  
 @Test  
 **void** create\_성공\_title과\_content만\_있는\_dto\_입력() {  
 *// 1. 예상 데이터* String title = **"라라라라"**;  
 String content = **"4444"**;  
 ArticleForm dto = **new** ArticleForm(**null**, title, content);  
 Article expected = **new** Article(4L, title, content);  
  
 *// 2. 실제 데이터* Article article = **articleService**.create(dto);  
  
 *// 3. 비교 및 검증  
 assertEquals*(expected.toString(), article.toString());  
 }  
  
 @Transactional  
 @Test  
 **void** create\_실패\_id가\_포함된\_dto\_입력() {  
 *// 1. 예상 데이터* Long id = 4L;  
 String title = **"라라라라"**;  
 String content = **"4444"**;  
 ArticleForm dto = **new** ArticleForm(id, title, content);  
 Article expected = **null**;  
  
 *// 2. 실제 데이터* Article article = **articleService**.create(dto);  
  
 *// 3. 비교 및 검증  
 assertEquals*(expected, article);  
 }  
}

### 셀프체크 정답

#### test/…/service/ArticeServiceTest.java

**package** com.example.firstproject.service;  
  
**import** com.example.firstproject.dto.ArticleForm;  
**import** com.example.firstproject.entity.Article;  
**import** org.junit.jupiter.api.Test;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.boot.test.context.SpringBootTest;  
**import** org.springframework.transaction.annotation.Transactional;  
  
**import** java.util.ArrayList;  
**import** java.util.Arrays;  
**import** java.util.List;  
  
**import static** org.junit.jupiter.api.Assertions.\*;  
  
@SpringBootTest  
**class** ArticleServiceTest {  
 @Autowired  
 ArticleService **articleService**;  
  
 @Test  
 **void** index() {  
 *// 1. 예상 데이터* Article a = **new** Article(1L, **"가가가가"**, **"1111"**);  
 Article b = **new** Article(2L, **"나나나나"**, **"2222"**);  
 Article c = **new** Article(3L, **"다다다다"**, **"3333"**);  
 List<Article> expected = **new** ArrayList<Article>(Arrays.*asList*(a, b, c));  
  
 *// 2. 실제 데이터* List<Article> articles = **articleService**.index();  
  
 *// 3. 비교 및 검증  
 assertEquals*(expected.toString(), articles.toString());  
 }  
  
 @Test  
 **void** show\_성공\_존재하는\_id\_입력() {  
 *// 1. 예상 데이터* Long id = 1L;  
 Article expected = **new** Article(id, **"가가가가"**, **"1111"**);  
  
 *// 2. 실제 데이터* Article article = **articleService**.show(id);  
  
 *// 3. 비교 및 검증  
 assertEquals*(expected.toString(), article.toString());  
 }  
  
 @Test  
 **void** show\_실패\_존재하지\_않는\_id\_입력() {  
 *// 1. 예상 데이터* Long id = -1L;  
 Article expected = **null**;  
  
 *// 2. 실제 데이터* Article article = **articleService**.show(id);  
  
 *// 3. 비교 및 검증  
 assertEquals*(expected, article);  
  
 }  
  
 @Transactional  
 @Test  
 **void** create\_성공\_title과\_content만\_있는\_dto\_입력() {  
 *// 1. 예상 데이터* String title = **"라라라라"**;  
 String content = **"4444"**;  
 ArticleForm dto = **new** ArticleForm(**null**, title, content);  
 Article expected = **new** Article(4L, title, content);  
  
 *// 2. 실제 데이터* Article article = **articleService**.create(dto);  
  
 *// 3. 비교 및 검증  
 assertEquals*(expected.toString(), article.toString());  
 }  
  
 @Transactional  
 @Test  
 **void** create\_실패\_id가\_포함된\_dto\_입력() {  
 *// 1. 예상 데이터* Long id = 4L;  
 String title = **"라라라라"**;  
 String content = **"4444"**;  
 ArticleForm dto = **new** ArticleForm(id, title, content);  
 Article expected = **null**;  
  
 *// 2. 실제 데이터* Article article = **articleService**.create(dto);  
  
 *// 3. 비교 및 검증  
 assertEquals*(expected, article);  
 }  
  
 @Test  
 @Transactional  
 **void** update\_성공\_존재하는\_id와\_title\_content가\_있는\_dto\_입력() {  
 *// 예상 데이터* Long id = 1L;  
 String title = **"가나다라"**;  
 String content = **"1234"**;  
 ArticleForm dto = **new** ArticleForm(id, title, content);  
 Article expected = **new** Article(id, title,content);  
  
 *// 실제 데이터* Article article = **articleService**.update(id, dto);  
  
 *// 비교 및 검증  
 assertEquals*(expected.toString(), article.toString());  
 }  
  
 @Test  
 @Transactional  
 **void** update\_성공\_존재하는\_id와\_title만\_있는\_dto\_입력() {  
 *// 예상 데이터* Long id = 1L;  
 String title = **"AAAA"**;  
 String content = **null**;  
 ArticleForm dto = **new** ArticleForm(id, title, content);  
 Article expected = **new** Article(1L, **"AAAA"** , **"1111"**);  
  
 *// 실제 데이터* Article article = **articleService**.update(id, dto);  
  
 *// 비교 및 검증  
 assertEquals*(expected.toString(), article.toString());  
 }  
  
 @Test  
 @Transactional  
 **void** update\_실패\_존재하지\_않는\_id의\_dto\_입력() {  
 *// 예상 데이터* Long id = -1L;  
 String title = **"가나다라"**;  
 String content = **"1234"**;  
 ArticleForm dto = **new** ArticleForm(id, title, content);  
 Article expected = **null**;  
  
 *// 실제 데이터* Article article = **articleService**.update(id, dto);  
  
 *// 비교 및 검증  
 assertEquals*(expected, article);  
 }  
  
 @Test  
 @Transactional  
 **void** delete\_성공\_존재하는\_id\_입력() {  
 *// 예상 데이터* Long id = 1L;  
 Article expected = **new** Article(id, **"가가가가"**, **"1111"**);  
  
 *// 실제 데이터* Article article = **articleService**.delete(id);  
  
 *// 비교 및 검증  
 assertEquals*(expected.toString(), article.toString());  
 }  
  
 @Test  
 @Transactional  
 **void** delete\_실패\_존재하지\_않는\_id\_입력() {  
 *// 예상 데이터* Long id = -1L;  
 Article expected = **null**;  
  
 *// 실제 데이터* Article article = **articleService**.delete(id);  
  
 *// 비교 및 검증  
 assertEquals*(expected, article);  
 }  
}