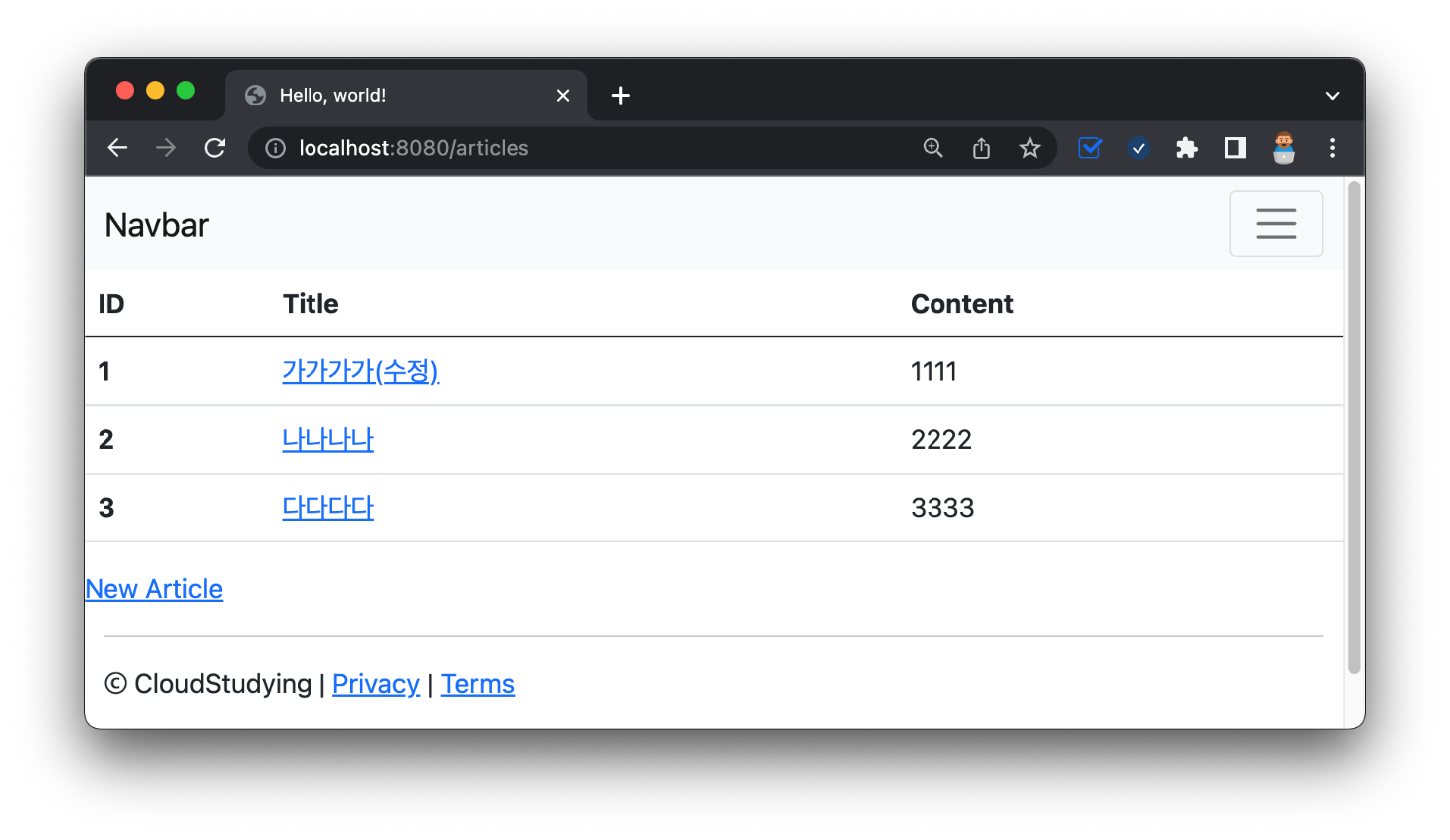
15 데이터 수정(update)하기

**데이터 수정하기**

**미션**

데이터를 수정하고, 이를 확인하시오.  


**03:33 더미 데이터 작성 - data.sql, insert into**

../resources/data.sql

1. -- 15강: article 더미 데이터
2. INSERT INTO article(id, title, content) VALUES(1, '가가가가', '1111');
3. INSERT INTO article(id, title, content) VALUES(2, '나나나나', '2222');
4. INSERT INTO article(id, title, content) VALUES(3, '다다다다', '3333');

**⚠️  failed to execute sql script statement 에러 발생 시**

../resources/application.properties

1. # 09강: h2 DB, 웹 콘솔 설정
2. spring.h2.console.enabled=true
3. # 15강: data.sql 적용을 위한 설정(스프링부트 2.5 이상 필수)
4. spring.jpa.defer-datasource-initialization=true
5. # spring boot 2.5.
6. spring.sql.init.mode=always

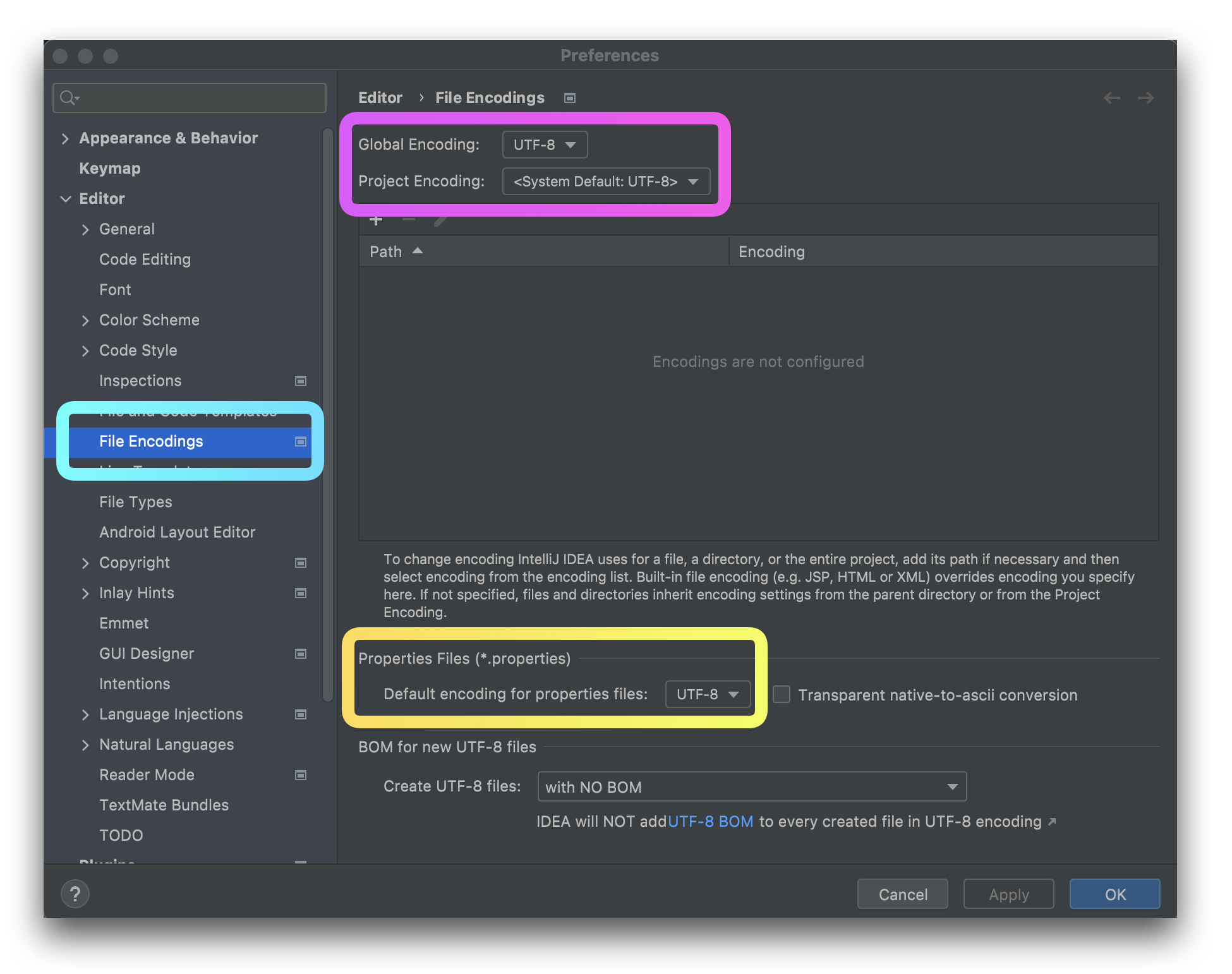
**⚠️  한글 깨짐 발생 시, “인텔리제이 한글 깨짐”으로 구글링!**

메뉴바 > Help > VM 옵션 수정

다음 설정을 끝부분에 추가한다.

1. -Dfile.encoding=UTF-8

메뉴바 > File > Settings(or Preferences) > … > File Encodings 설정

Global Encoding, Project Encoding, Properties Files 모두를 UTF-8로 바꾼다.  


**05:16 수정 페이지 변경 - form, action, method, post, patch**

../articles/edit.mustache

1. {{>layouts/header}}
2. {{#article}}
3. <form class="container" action="/articles/update" method="post">
4. <input name="id" type="hidden" value="{{id}}" />
5. <div class="mb-3">
6. <label class="form-label">제목</label>
7. <input type="text" class="form-control" name="title" value="{{title}}">
8. </div>
9. <div class="mb-3">
10. <label class="form-label">내용</label>
11. <textarea class="form-control" rows="3" name="content">{{content}}</textarea>
12. </div>
13. <button type="submit" class="btn btn-primary">Submit</button>
14. <a href="/articles/{{id}}">Back</a>
15. </form>
16. {{/article}}
17. {{>layouts/footer}}

**07:28 수정 폼 받기 - Controller, PatchMapping, PostMapping, DTO**

../controller/ArticleController

1. ...
2. @Controller
3. @Slf4j
4. public class ArticleController {
5. ...
6. @PostMapping("/articles/update")
7. public String update(ArticleForm form) {
8. log.info(form.toString());
9. return "";
10. }
11. }

**08:36 DTO 변경 - id 필드 추가 및 엔티티 변환 메소드 변경**

../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 Long id;
9. private String title;
10. private String content;
11. public Article toEntity() {
12. return new Article(id, title, content);
13. }
14. }

**09:59 수정 폼 처리하기**

../controller/ArticleController

1. ...
2. @Controller
3. @Slf4j
4. public class ArticleController {
5. ...
6. @PostMapping("/articles/update")
7. public String update(ArticleForm form) {
8. log.info(form.toString());
9. // 1: DTO를 엔티티로 변환
10. Article articleEntity = form.toEntity();
11. log.info(articleEntity.toString());
12. // 2: 엔티티를 DB로 저장
13. // 2-1: DB에서 기존 데이터를 가져옴
14. Article target = articleRepository.findById(articleEntity.getId())
15. .orElse(null);
16. // 2-2: 기존 데이터가 있다면, 값을 갱신
17. if (target != null) {
18. articleRepository.save(articleEntity);
19. }
20. // 3: 수정 결과 페이지로 리다이렉트
21. return "redirect:/articles/" + articleEntity.getId();
22. }
23. }

**🔥 구글링 훈련하기**

* jpa data.sql 사용법
* jpa defer-datasource-initialization
* 인텔리제이 한글 깨짐