11 장 완성 코드

api/FirstApiController.java

```
package com.example.firstproject.api;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class FirstApiController {

    @GetMapping("/api/hello")
    public String hello() {
        return "hello world!";
    }

}
```

api/ArticleApiController.java

```
package com.example.firstproject.api;

import com.example.firstproject.dto.ArticleForm;
import com.example.firstproject.entity.Article;
import com.example.firstproject.repository.ArticleRepository;
import lombok.extern.slf4j.Slf4j;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@Slf4j
@RestController
public class ArticleApiController {
    @Autowired
    private ArticleRepository articleRepository;

    // GET
    @GetMapping("/api/articles")
    public List<Article> index() {
```

```
        return articleRepository.findAll();
    }

    @GetMapping("/api/articles/{id}")
    public Article show(@PathVariable Long id) {
        return articleRepository.findById(id).orElse(null);
    }

    // POST
    @PostMapping("/api/articles")
    public Article create(@RequestBody ArticleForm dto) {
        Article article = dto.toEntity();
        return articleRepository.save(article);
    }

    // PATCH
    @PatchMapping("/api/articles/{id}")
    public ResponseEntity<Article> update(@PathVariable Long id,
                                          @RequestBody ArticleForm dto) {
        // 1. DTO -> 엔티티 변환하기
        Article article = dto.toEntity();
        log.info("id: {}, article: {}", id, article.toString());

        // 2. 타깃 조회하기
        Article target = articleRepository.findById(id).orElse(null);

        // 3. 잘못된 요청 처리하기
        if (target == null || id != article.getId()) {
            // 400, 잘못된 요청 응답!
            log.info("잘못된 요청! id: {}, article: {}", id,
article.toString());
            return ResponseEntity.status(HttpStatus.BAD_REQUEST).body(null);
        }

        // 4. 업데이트 및 정상 응답(200)하기
        target.patch(article);
        Article updated = articleRepository.save(target);
        return ResponseEntity.status(HttpStatus.OK).body(updated);
    }

    // DELETE
    @DeleteMapping("/api/articles/{id}")
    public ResponseEntity<Article> delete(@PathVariable Long id) {
        // 1. 대상 찾기
        Article target = articleRepository.findById(id).orElse(null);

        // 2. 잘못된 요청 처리하기
        if (target == null) {
            return ResponseEntity.status(HttpStatus.BAD_REQUEST).body(null);
        }
```

```
        // 3. 대상 삭제하기
        articleRepository.delete(target);
        return ResponseEntity.status(HttpStatus.OK).build();
    }
}
```

entity/Article.java

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

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.ToString;

@AllArgsConstructor
@NoArgsConstructor
@ToString
@Entity
@Getter
public class Article {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY) // DB가 id 자동 생성
    private Long id;

    @Column
    private String title;

    @Column
    private String content;

    public void patch(Article article) {
        if (article.title != null)
            this.title = article.title;
        if (article.content != null)
            this.content = article.content;
    }
}
```

셀프체크 정답

api/CoffeeApiController.java

```java
package com.example.firstproject.api;

import com.example.firstproject.dto.CoffeeDto;
import com.example.firstproject.entity.Coffee;
import com.example.firstproject.repository.CoffeeRepository;
import lombok.extern.slf4j.Slf4j;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

@Slf4j
@RestController
public class CoffeeApiController {
    @Autowired
    private CoffeeRepository coffeeRepository;

    // GET
    @GetMapping("/api/coffees")
    public Iterable<Coffee> index() {
        return coffeeRepository.findAll();
    }

    @GetMapping("/api/coffees/{id}")
    public ResponseEntity<Coffee> show(@PathVariable Long id) {
        Coffee coffee = coffeeRepository.findById(id).orElse(null);
        return (coffee != null) ?
                ResponseEntity.status(HttpStatus.OK).body(coffee) :
                ResponseEntity.status(HttpStatus.BAD_REQUEST).body(null);
    }

    // POST
    @PostMapping("/api/coffees")
    public ResponseEntity<Coffee> create(@RequestBody CoffeeDto coffeeDto) {
        Coffee coffee = coffeeDto.toEntity();
        if (coffee.getId() != null) {
            return ResponseEntity
                    .status(HttpStatus.BAD_REQUEST)
                    .body(null);
        }
        Coffee created = coffeeRepository.save(coffee);
        return ResponseEntity
                .status(HttpStatus.CREATED)
                .body(created);
```

```
    }

    // PATCH
    @PatchMapping("/api/coffees/{id}")
    public ResponseEntity<Coffee> update(@PathVariable Long id,
                                         @RequestBody CoffeeDto coffeeDto) {
        Coffee coffee = coffeeDto.toEntity();
        log.info("id: {}, coffee: {}", id, coffee.toString());

        Coffee target = coffeeRepository
                .findById(id)
                .orElse(null);

        if (target == null || id != coffee.getId()) {
            log.info("잘못된 요청! id: {}, coffee: {}", id, coffee.toString());
            return ResponseEntity
                    .status(HttpStatus.BAD_REQUEST)
                    .body(null);
        }

        target.patch(coffee);
        Coffee updated = coffeeRepository.save(target);
        return ResponseEntity
                .status(HttpStatus.OK)
                .body(updated);
    }

    // DELETE
    @DeleteMapping("/api/coffees/{id}")
    public ResponseEntity<Coffee> delete(@PathVariable Long id) {
        Coffee target = coffeeRepository
                .findById(id)
                .orElse(null);

        if (target == null) {
            return ResponseEntity
                    .status(HttpStatus.BAD_REQUEST)
                    .body(null);
        }
        coffeeRepository.delete(target);

        return ResponseEntity
                .status(HttpStatus.OK)
                .body(null);
    }
}
```

dto/CoffeeDto.java

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

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

@AllArgsConstructor
@ToString
public class CoffeeDto {
    private Long id;
    private String name;
    private String price;

    public Coffee toEntity() {
        return new Coffee(id, name, price);
    }
}
```

entity/Coffee.java

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

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.ToString;

@Entity
@AllArgsConstructor
@NoArgsConstructor
@ToString
@Getter
public class Coffee {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY) // 자동 생성 전략
    private Long id;
    @Column
    private String name;
    @Column
    private String price;

    public void patch(Coffee coffee) {
        if (coffee.name != null)
            this.name = coffee.name;
        if (coffee.price != null)
            this.price = coffee.price;
```

```
    }
}
```

repository/CoffeeRepository.java

```
package com.example.firstproject.repository;

import com.example.firstproject.entity.Coffee;
import org.springframework.data.repository.CrudRepository;

public interface CoffeeRepository extends CrudRepository<Coffee, Long> {
}
```