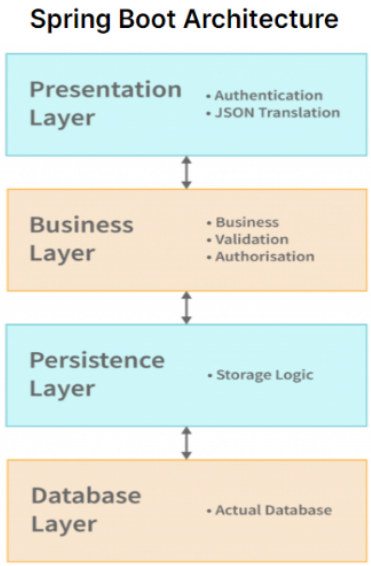
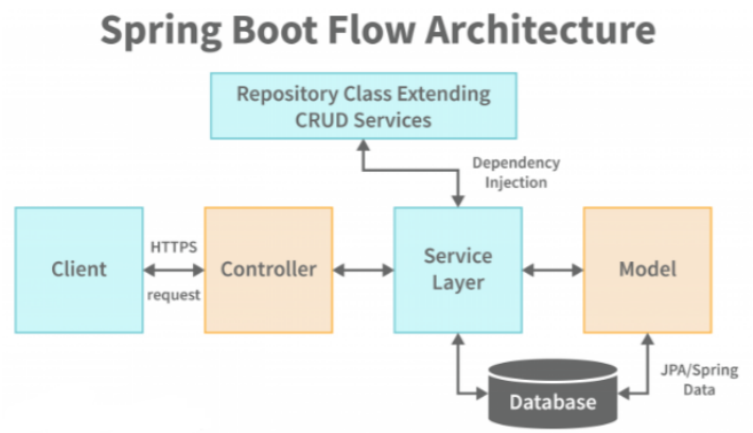
2장 SpringBoot - JPA기초

Spring Boot Acrchitecture





Presentation Layer는 View다 .

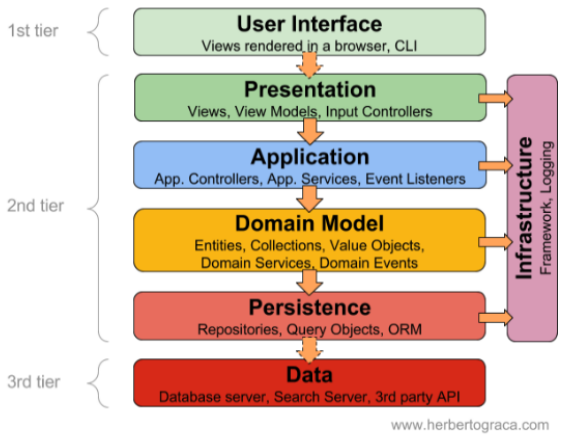
BusinessLayer 는 코드 흐름을 나타낸다.

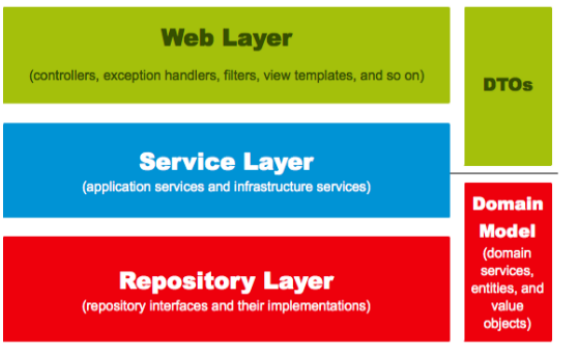
Persistence Layer 는 실제적 데이터를 처리하는 공간

Database Layer 는 DB와 관련된 공간이다.

유저가 무언가를 요청하면 컨트롤러가 받아서 해당 요청을 처리할 서비스를 호출한다 .

호출된 서비스는 DB에 필요한 CRUD중 알맞은 동작을 진행한다.12





https://hudi.blog/layered-architecture/

SQL Mapper VS ORM

SQL Mapper

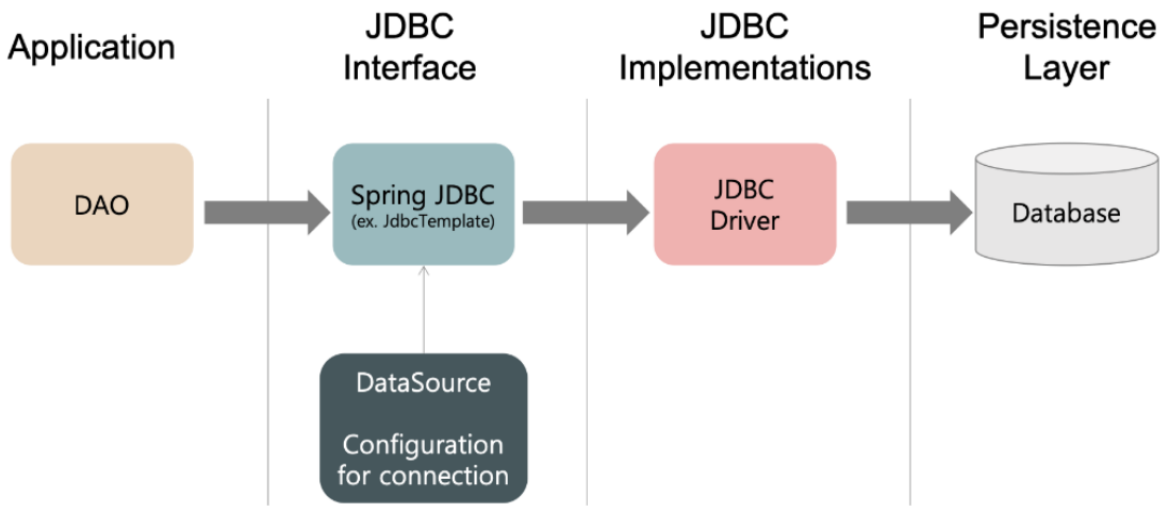
- SQL <—매핑—> Object 필드
- SQL Mapper는 SQL 문장으로 직접 데이터베이스 데이터를 다룬다.
- 즉, SQL Mapper는 SQL을 명시해줘야 한다.
- Ex) Mybatis, JdbcTempletes 등
- 실제로 코드로 SQL문장을 하나하나 다 작성하고 해당 명령어와 Mapper되도록 한다.

ORM(Object-Relational Mapping), 객체-관계 매핑

- 데이터베이스 데이터 <—매핑—> Object 필드
- 객체를 통해 간접적으로 데이터베이스 데이터를 다룬다.
- 객체와 관계형 데이터베이스의 데이터를 자동으로 매핑(연결)해주는 것을 말한다.
- ORM을 이용하면 SQL Query가 아닌 직관적인 코드(메서드)로 데이터를 조작할 수 있다.
- 객체 간의 관계를 바탕으로 SQL을 자동으로 생성한다.
- Persistant API라고도 할 수 있다.
- Ex) JPA, Hibernate 등

JPA를 사용할때 Hibernate를 연결해서 사용해야한다.

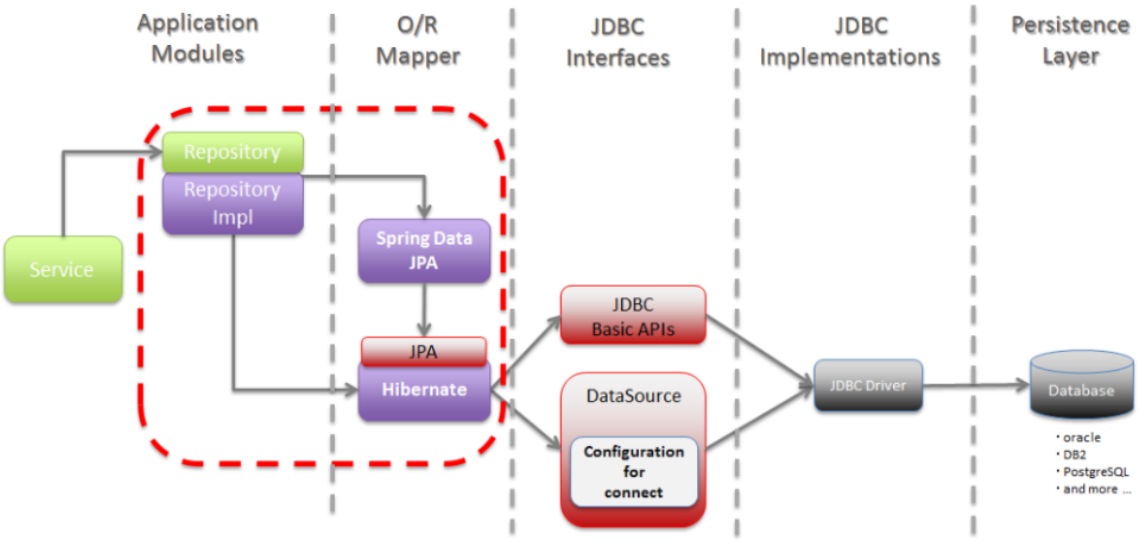
JDBC



https://gmlwjd9405.github.io/2018/12/25/difference-jdbc-jpa-mybatis.html

- JPA를 사용하지않고 스프링에서 DB를 처리하기 위한 방법이다.

JPA(Java Persistent API)



https://gmlwjd9405.github.io/2018/12/25/difference-jdbc-jpa-mybatis.html

- 서비스가 작업을 할때 Repository를 호출한다. Repository는 기본적으로 인터페이스이다.
- 해당 JPA를 인터페이스로 정의해놓으면서 개발자가 직접 SQL명령어를 입력하지 않게된다.
- 사진에 보이는 빨간색 부분을 Interface로 정의함으로써 스프링프레임워크에서 알아서 동작하도록 되어있다. 그러므로 개발자가 신경을 안써도 된다.

Entity

- JPA를 사용해서 테이블을 만들고 SQL 작업을 실행할 수 있다.
- 해당 테이블에 데이터를 추출할때 자바에서 해당 데이터를 받을 수 있는 객체가 필요하다.
- 해당 역할을 하는 객체가 Entity객체이다.
- Entity 객체는 DB에서 데이터에 대한 CRU를 진행할때 사용되는 객체이다.
- Setter 사용을 지향해야한다. 대신 Builder를 사용한다.
- Entity객체는 DB의 테이블과 매핑되는 객체이므로 객체무결성을 중요하게 관리해야한다.

DTO

- DTO에 담기는 데이터는 View에서 클라이언트가 보낸 데이터이거나 혹은 서버에서 사용자에게 넘겨주는 데이터를 담는 역할을 한다.
- 사용자가 서버에게 본인에 대한 정보를 수정을 한다면 수정을 한 내용을 DTO에 담고 Service에서 해당 데이터를 받아서 DTO 객체를 Entity 객체에 담아서 JPA에게 전달한다. 그러면 JPA가 받은 데이터를 통해서 DB의 데이터를 처리한다.
- 추후에는 DTO의 명을 UserCreateForm 등 해당 DTO가 어떤 정보를 담아서 보내는지에 대해 작성하는 객체명으로도 사용할수 있음.

실습 코드

Application.properties

```
spring.datasource.url=jdbc:mariadb://localhost:3306/springdb
spring.datasource.username=scott
spring.datasource.password=tiger

spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.format_sql=true
spring.jpa.show-sql=true
```

MemoBoard.java

```
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

import lombok.AllArgsConstructor;
import lombok.Builder;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.ToString;

@Entity
@Table(name="memoboard")
@ToString
@Getter
@Builder
@AllArgsConstructor
@NoArgsConstructor
public class MemoBoard {
    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY) //DB의 성격에 맞게 알아서 동작하도록 마리아디비는 오토인크리먼트, 오라클은 시퀀스를 만들어서 동작한다.
    private Long mid;

    @Column(length = 200, nullable= false)
    private String content;


}
```

- 해당 코드는 DB에 실제로 만드는 테이블과 Mapping되는 Entity 객체이다.
-

MemoBoardRepository

```
import java.util.List;

import javax.transaction.Transactional;

import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Modifying;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;

import com.kong.king.spring.exam02.entity.MemoBoard;


public interface MemoBoardRepository extends JpaRepository<MemoBoard, Long> {

    List<MemoBoard> findByMidBetweenOrderByMidDesc(Long from, Long to);
    Page<MemoBoard> findByMidBetween(Long from, Long to, Pageable pageable);
    @Query("select m from MemoBoard m order by m.mid desc")
    List<MemoBoard> getMemoBoardListAllDesc();

    @Transactional
    @Modifying
    @Query("update MemoBoard m set m.content = :content where m.mid = :mid")
    int updateMemoBoardContentWithMid(@Param("mid") Long mid, @Param("content")
String content);

    @Transactional
    @Modifying
    @Query("update MemoBoard m set m.content = :#{#param.content} where m.mid = :#
{#param.mid}")
    int updateMemoBoardContentWithObj(@Param("param") MemoBoard memo);

    @Query(value="select m from MemoBoard m where m.mid > :mid",
            countQuery ="select count(m) from MemoBOard m where m.mid > :mid")
    Page<MemoBoard> getListWithQueryandPaging(Long mid, Pageable pageable);

    @Query(value="select m.mid, m.content, CURRENT_DATE from  MemoBoard m where
m.mid > :mid",
            countQuery ="select count(m) from MemoBOard m where m.mid > :mid")
    Page<Object[]> getListWitrhQueryObject(Long mid, Pageable pageable);
```

```
    @Query(value = "select * from memoboard where content like '%7%' ",
nativeQuery=true)
    List<MemoBoard> getNativeResult();

    //insert 작업 : save(엔티티 객체)
    //Select 작업 : findById(키타입), getOne(키 타입)
    //Update 작업 : save(엔티티 객체)
    //Delete 작업 : deleteById(키타입), delete(엔티티 객체)

}
```

- Interface로 어떤 테이블의 레포지토리인지 클래스명으로 작성해주고 JPARepository를 extend를 해주면 되는데. <> 안에 Entity명과 해당 Entity의 Pk의 타입을 적어주면 된다.
- 이런 방식으로 작성을 하면 자동으로 DB안에 테이블이 없으면 자동으로 생성이 된다.

SpringStudyExam02ApplicationTests

```
package com.kong.king.spring.exam02.repository;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.Optional;
import java.util.stream.IntStream;

import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Pageable;
import org.springframework.data.domain.Sort;

import com.kong.king.spring.exam02.entity.MemoBoard;
import com.kong.king.spring.exam02.repository.MemoBoardRepository;

@SpringBootTest
public class MemoBoardRepositoryTest {
    @Autowired
    MemoBoardRepository memoBoardRepository;

    /*
     * @Test void contextLoads() { }
     */
```

```
    //테이블 생성 테스트v 10

    @Test
    public void testClass() {
        System.out.println("객체생성 확인 ----" +
        memoBoardRepository.getClass().getName());
    }


//  테이블 데이터 생성 12번 테스트

     @Test
    public void testInsertDummies() {
        IntStream.rangeClosed(1,100).forEach(i -> {
            MemoBoard memo = MemoBoard.builder()
                    .content("메모...."+i)
                    .build();
        memoBoardRepository.save(memo);
        });
    }

    // 테이블 데이터 생성 13번테스트
    @Test
    public void testInsertDummies2() {
        MemoBoard memo;
        for(int i = 200; i<=250; i++) {
            memo = new MemoBoard();
            memo.setContent("메모x테스트..."+i);
            memoBoardRepository.save(memo);
        }
    }

    //테이블 조회 테스트 14번
    @Test
    public void testSelectById() {
        Long mid = 3L;
        Optional<MemoBoard> result = memoBoardRepository.findById(mid);
        System.out.println("------------");
        if(result.isPresent()) {
            MemoBoard memo = result.get();
            System.out.println(memo);
        }

    }

    //테이블 조회 15번
    @Test
    public void testSelect2ById() {
        Long mid = 30L;
        MemoBoard memo = memoBoardRepository.getOne(mid);
        System.out.println("------------");
```

```
        System.out.println(memo);

    }

    //테이블 조회 16번
    @Test
    public void testSelectAll() {
        List<MemoBoard> list = new ArrayList<MemoBoard>();
        list = memoBoardRepository.findAll();
        System.out.println(list);
    }

    //테이블 업데이트 17번
    @Test
    public void testUpdate() {
        MemoBoard memo = MemoBoard.builder()
                .mid(100L)
                .content("100번 데이터 수정...")
                .build();
        System.out.println(memoBoardRepository.save(memo));
    }
//
    //테이블 삭제 테스트 18번
    @Test
    public void testDelete() {
        Long mid = 100L;
        memoBoardRepository.deleteById(mid);
    }

      //페이징 처리 조회 19번
      @Test
      public void testPageDefault() {
          Pageable pageable =PageRequest.of(0,10);
          Page<MemoBoard> result = memoBoardRepository.findAll(pageable);
          System.out.println(result); System.out.println("-----------------");
          System.out.println("Total Pages : " + result.getTotalPages());
          System.out.println("Total Count : " + result.getTotalElements());
          System.out.println("Page Number : " + result.getSize());
          System.out.println("Page Size : " + result.getSize());
          System.out.println("has next page?" + result.hasNext());
          System.out.println("first page? : " + result.isFirst());
          System.out.println("-----------------"); for(MemoBoard memo :
          result.getContent()) { System.out.println(memo); }
      }

    //테이블 소트테스트 20번
    @Test
    public void testSort() {
        Sort sort1 = Sort.by("mid").descending();

        Pageable pageable = PageRequest.of(0,10,sort1);
        Page<MemoBoard> result = memoBoardRepository.findAll(pageable);
```

```
        result.get().forEach(memo -> {
            System.out.println(memo);
        });
    }

    //테이블 소트테스트 21번
    @Test
    public void testSort2() {
        Sort sort1 = Sort.by("content").ascending();
        Sort sort2 = Sort.by("mid").descending();
        Sort sortAll = sort1.and(sort2);

        Pageable pageable = PageRequest.of(0,10,sortAll);
        Page<MemoBoard> result = memoBoardRepository.findAll(pageable);

        result.get().forEach(memo -> {
            System.out.println(memo);
        });
    }

    /*
     * //Sort 거꾸로하기
     *
     * @Test public void testOsrt() { Sort sort1 = Sort.by("mid").descending();
     * Pageable pageable = PageRequest.of(0,10,sort1); Page<MemoBoard> result =
     * memoBoardRepository.findAll(pageable);
     *
     * result.get().forEach(memo -> { System.out.println(memo); });
     *
     * }
     */

    // 테이블 쿼리메서트 테스트 22번 70~80사이 mid값 거꾸로 정렬해서 나타내기.
    @Test
    public void testQueryMethods() {
        List<MemoBoard> list =
 memoBoardRepository.findByMidBetweenOrderByMidDesc(70L, 80L);
        for(MemoBoard memo : list) {
            System.out.println(memo);
        }
    }

    //테이블 쿼리메서드 테스트 23번
    @Test
    public void testQUertMethodWithPageable() {
        Pageable pageable = PageRequest.of(0,10,Sort.by("mid").descending());
        Page<MemoBoard> result =
memoBoardRepository.findByMidBetween(10L,50L,pageable);
        result.get().forEach(memo -> System.out.println(memo));
    }
```

```
//  //테이블 Query 어노테이션으로 한번실행하기  24번
    @Test
    public void testAtQueryGetAllData() {
        List<MemoBoard> list = memoBoardRepository.getMemoBoardListAllDesc();
        for(MemoBoard memo : list) {
            System.out.println(memo);
        }

    }

//  //Qeury 파라미터 바인딩 1 25번
    @Test
    public void testAtQueryUpdateById() {
        int result = memoBoardRepository.updateMemoBoardContentWithMid(2L, "@Query를
활용한 수정 실습");
        System.out.println("수정 결과 : " + result);
    }


    // Query의 파라미터 바인딩2 26번
    @Test
    public void testAtQueryUpdateByObj() {
        MemoBoard memo = new MemoBoard();

        memo.setMid(2L);
        memo.setContent("@Query를 이용한 수정 - Object전달 방식");

        int result = memoBoardRepository.updateMemoBoardContentWithObj(memo);
        System.out.println("수정 결과 : " + result);
    }

    //Query 어노테이션 연습 27번
  @Test
  public void testAtQueryandPaging() {
     Pageable pageable = PageRequest.of(0,10, Sort.by("mid").descending());
     Page<MemoBoard> result = memoBoardRepository
           .getListWithQueryandPaging(1L, pageable);

     System.out.println("---------");
     System.out.println("result="+result);
     result.get().forEach(memo -> System.out.println(memo));

  }

    //Query 어노테이션 연습 28번
  @Test
  public void test() {
     Pageable pageable = PageRequest.of(0,10, Sort.by("mid").descending());
     Page<Object[]> result = memoBoardRepository
           .getListWitrhQueryObject(1L, pageable);
     System.out.println("----------------");
```

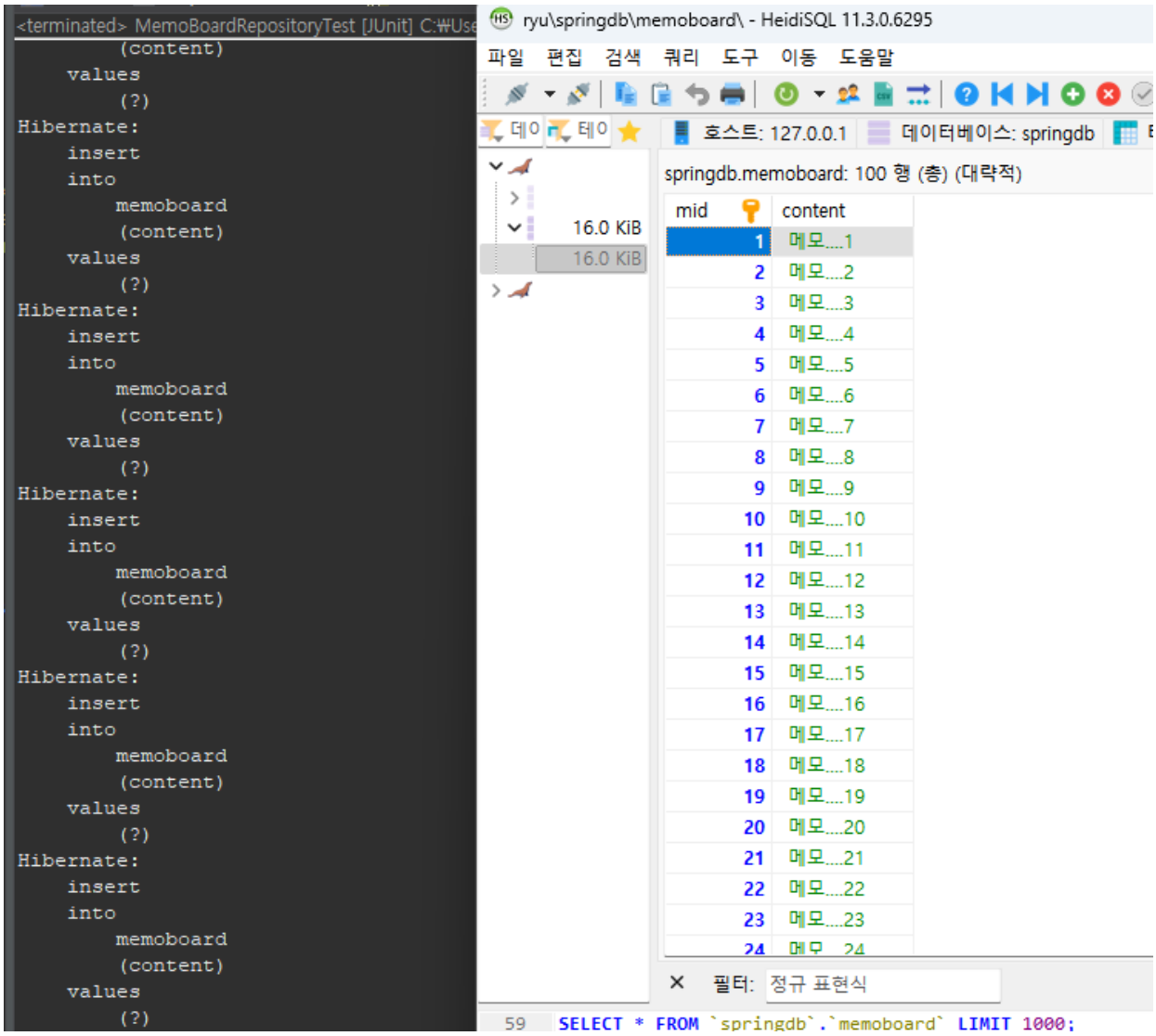
```
      System.out.println("result= "+result);
      result.get().forEach(memo -> System.out.println(Arrays.toString(memo)));
   }

     //Query 어노테이션 연습 29번
   @Test
   public void testNativeQuery() {
      List<MemoBoard> list = memoBoardRepository.getNativeResult();

      for(MemoBoard m : list) {
         System.out.println(m);
      }
   }

}
```

- 해당 클래스는 스프링에서 모듈을 만든후 테스트를 할 수 있다.
- Run as를 클릭하여서 JUnit Test를 클릭하면 실행이된다.
- 이번에 실습할 모든 코드를 다 작성해서 붙였다. 각각 실행 결과에 대해서 서술하겠다.

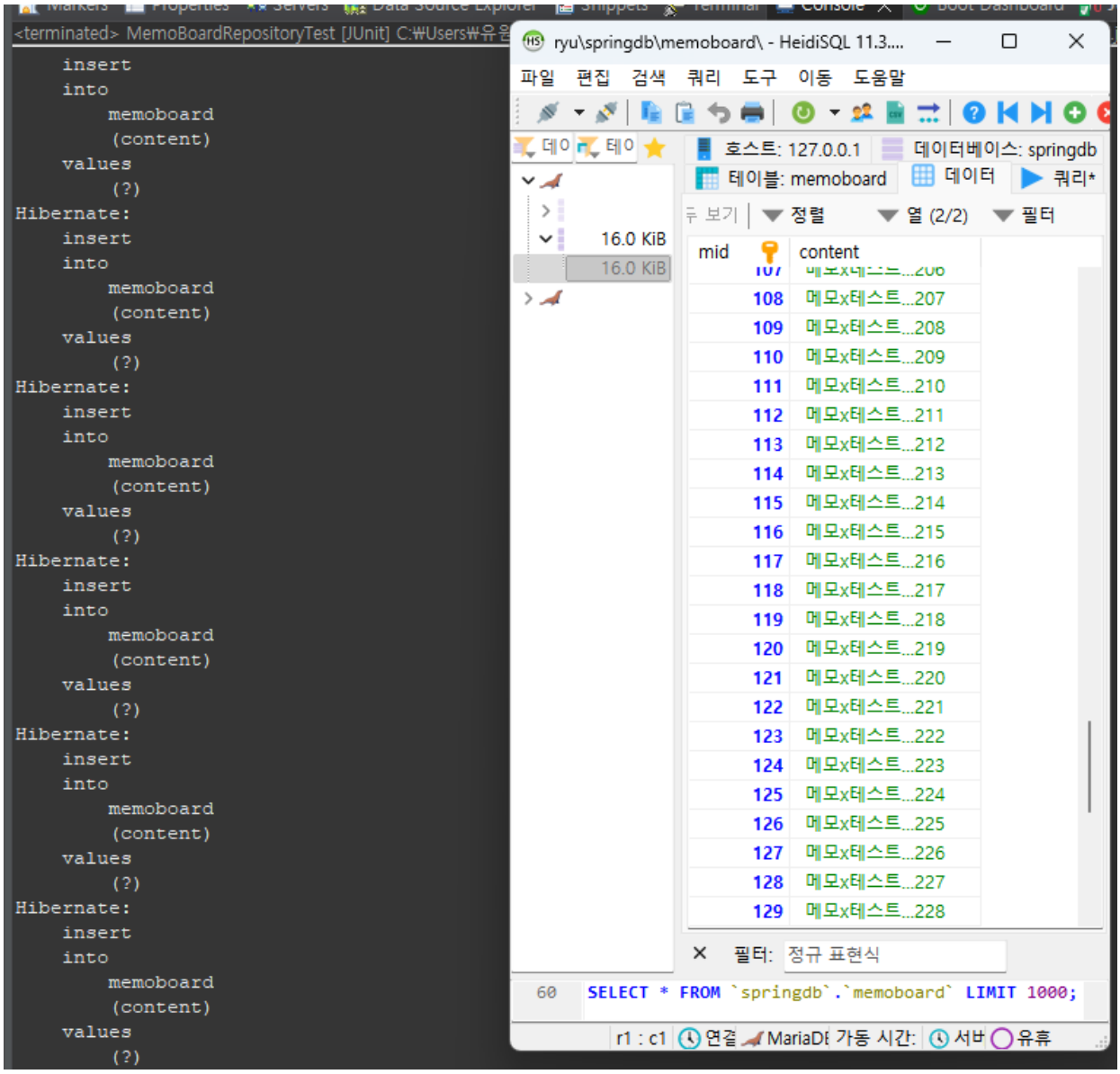
JPA실습 1번

- 실행이되면 DB에서 테이블이 생성된것을 확인할 수 있다.
-
```
2023-04-11 10:16:53.715  WARN 9572 --- [
2023-04-11 10:16:54.116  WARN 9572 --- [
2023-04-11 10:16:54.657  INFO 9572 --- [
객체생성 확인 ----jdk.proxy2.$Proxy114
2023-04-11 10:16:54.879  INFO 9572 --- [i
2023-04-11 10:16:54.881  INFO 9572 --- [i
```

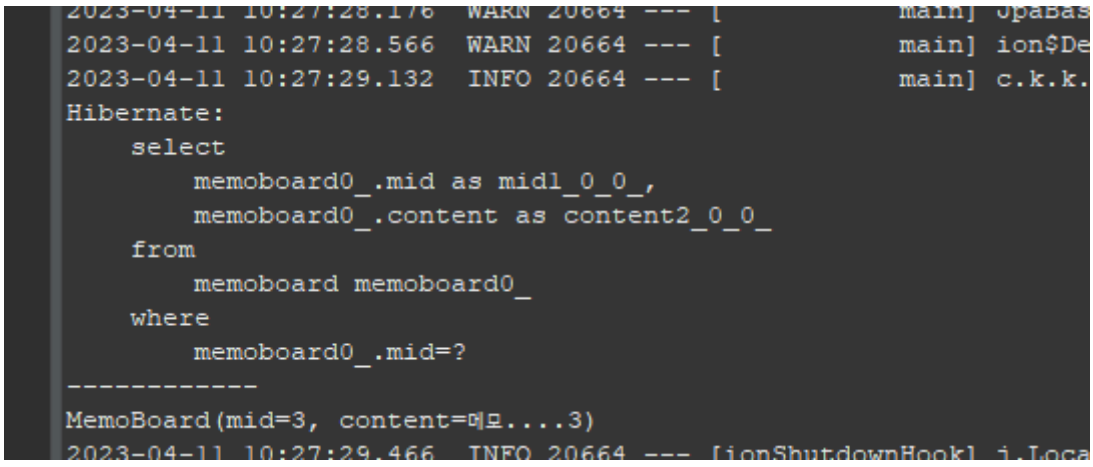
JPA 실습 2번


(content)
values
(?)
Hibernate:
insert
into
memoboard
(content)
values
(?)
Hibernate:
insert
into
memoboard
(content)
values
(?)
Hibernate:
insert
into
memoboard
(content)
values
(?)
Hibernate:
insert
into
memoboard
(content)
values
(?)
Hibernate:
insert
into
memoboard
(content)
values
(?)
ryu\springdb\memoboard\ - HeidiSQL 11.3.0.6295
파일 편집 검색 쿼리 도구 이동 도움말
호스트: 127.0.0.1
데이터베이스: springdb
springdb.memoboard: 100 행 (총) (대략적)
16.0 KiB
16.0 KiB
mid content
1 메모...1
2 메모...2
3 메모...3
4 메모...4
5 메모...5
6 메모...6
7 메모...7
8 메모...8
9 메모...9
10 메모...10
11 메모...11
12 메모...12
13 메모...13
14 메모...14
15 메모...15
16 메모...16
17 메모...17
18 메모...18
19 메모...19
20 메모...20
21 메모...21
22 메모...22
23 메모...23
필터: 정규 표현식
59 SELECT * FROM `springdb`.`memoboard` LIMIT 1000;

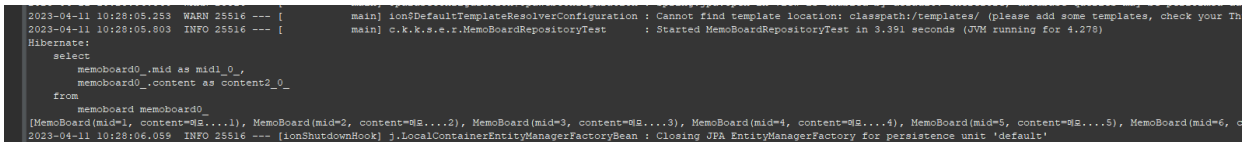
JPA 실습 3번


insert into memoboard (content) values (?)
Hibernate:
ryu\springdb\memoboard\ - HeidiSQL 11.3....
파일 편집 검색 쿼리 도구 이동 도움말
호스트: 127.0.0.1
데이터베이스: springdb
테이블: memoboard
데이터
쿼리*
정렬
열 (2/2)
필터
mid
content
108 메모x테스트...207
109 메모x테스트...208
110 메모x테스트...209
111 메모x테스트...210
112 메모x테스트...211
113 메모x테스트...212
114 메모x테스트...213
115 메모x테스트...214
116 메모x테스트...215
117 메모x테스트...216
118 메모x테스트...217
119 메모x테스트...218
120 메모x테스트...219
121 메모x테스트...220
122 메모x테스트...221
123 메모x테스트...222
124 메모x테스트...223
125 메모x테스트...224
126 메모x테스트...225
127 메모x테스트...226
128 메모x테스트...227
129 메모x테스트...228
필터: 정규 표현식
60 SELECT * FROM `springdb`.`memoboard` LIMIT 1000;

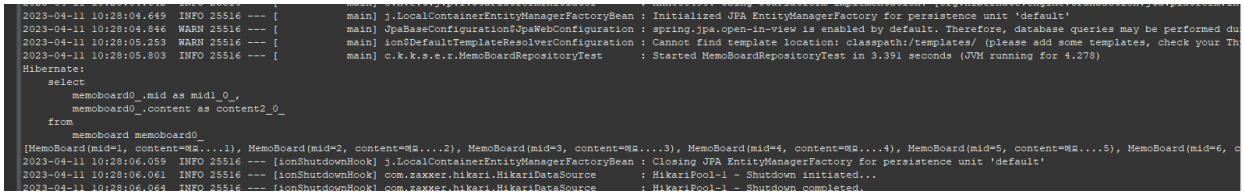
JPA 실습 4번


2023-04-11 10:27:28.566 WARN 20664 --- [main] ion$De
2023-04-11 10:27:29.132 INFO 20664 --- [main] c.k.k.
Hibernate:
select
memoboard0_.mid as mid1_0_0_,
memoboard0_.content as content2_0_0_
from
memoboard memoboard0_
where
memoboard0_.mid=?
MemoBoard(mid=3, content=메모....3)

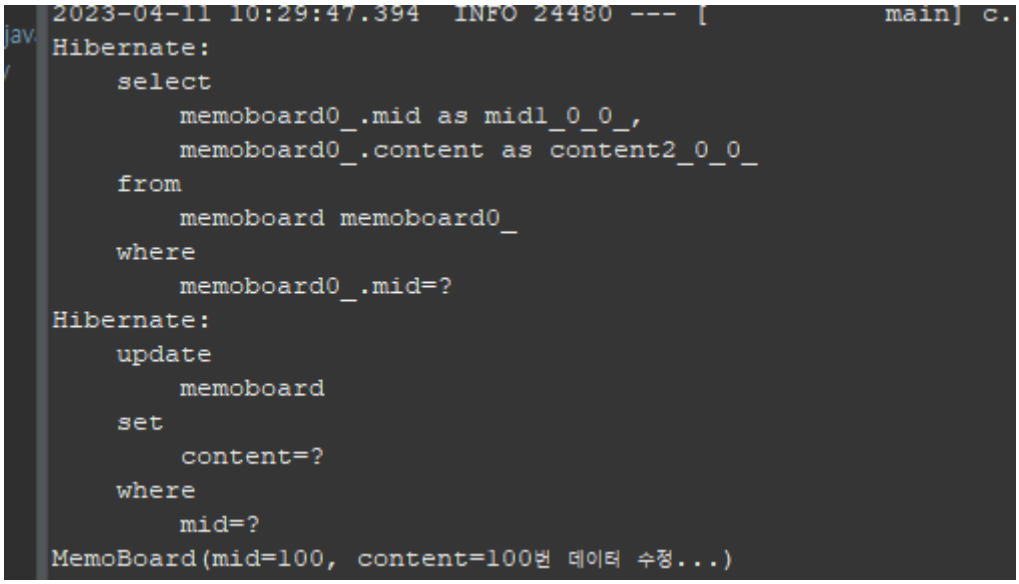
JPA 실습 5번



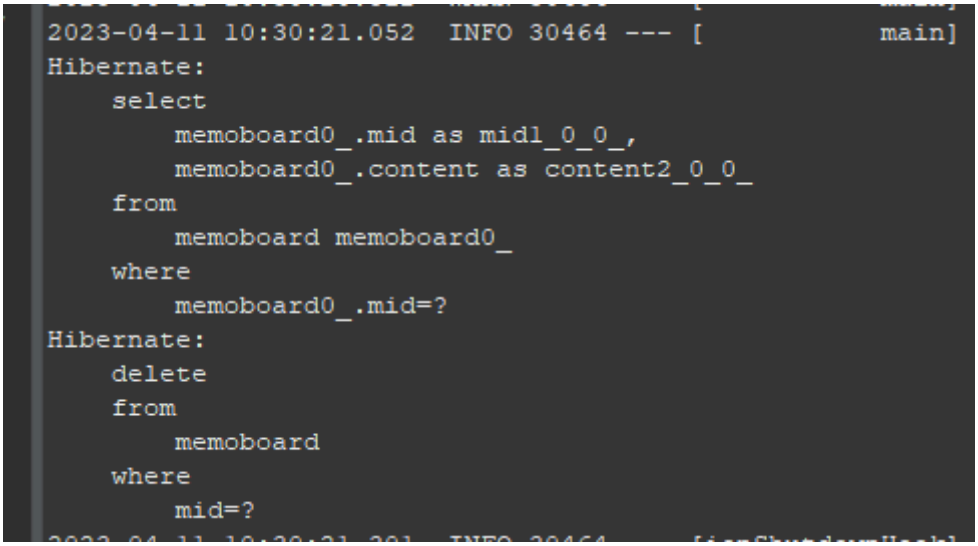
JPA 실습 6번



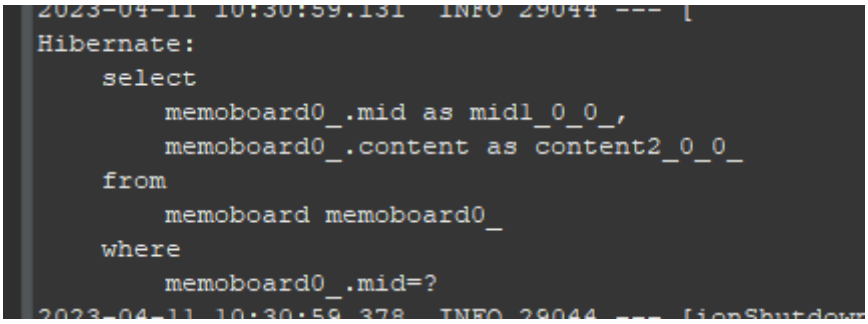
JPA 실습 7번



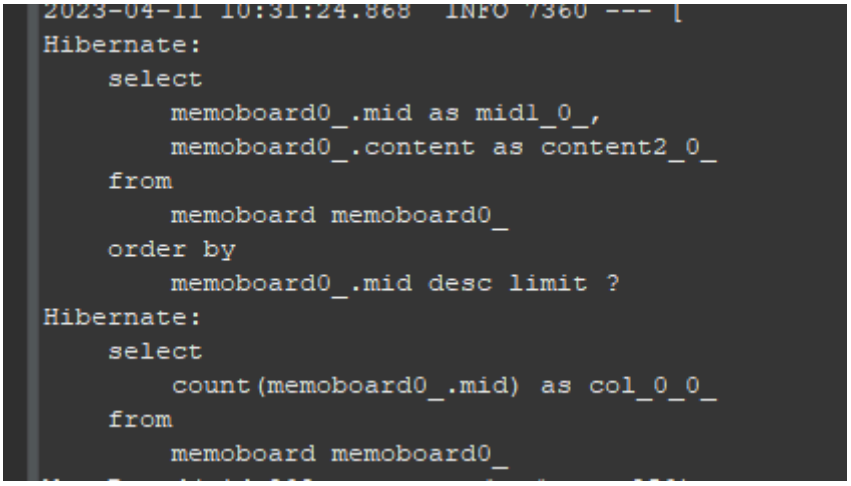
JPA 실습 8번



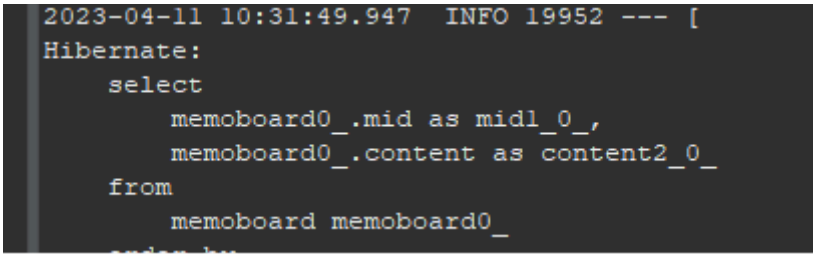
JPA 실습 9번



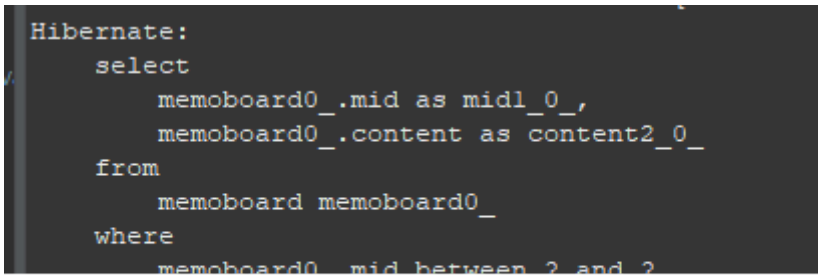
JPA 실습 10번



JPA 실습 11번

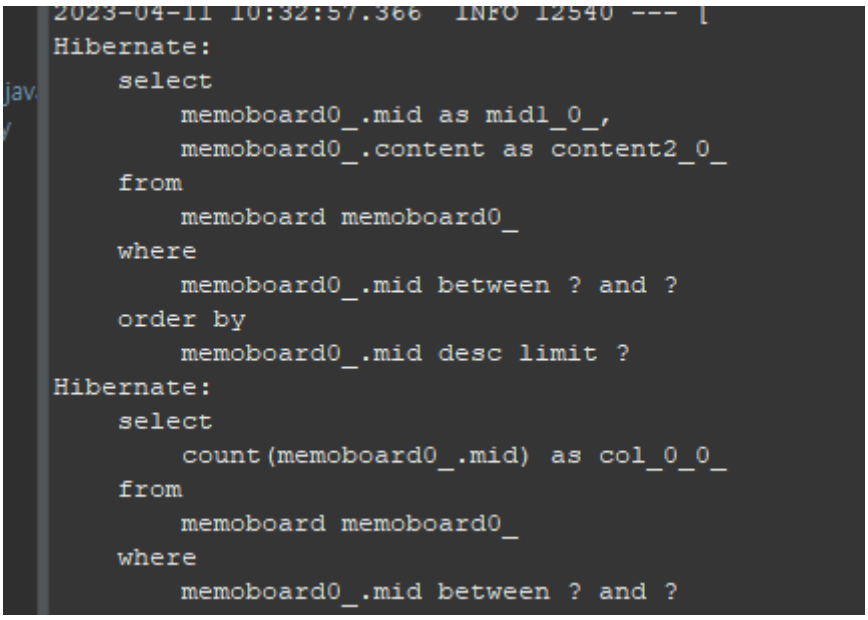


JPA 실습 12번



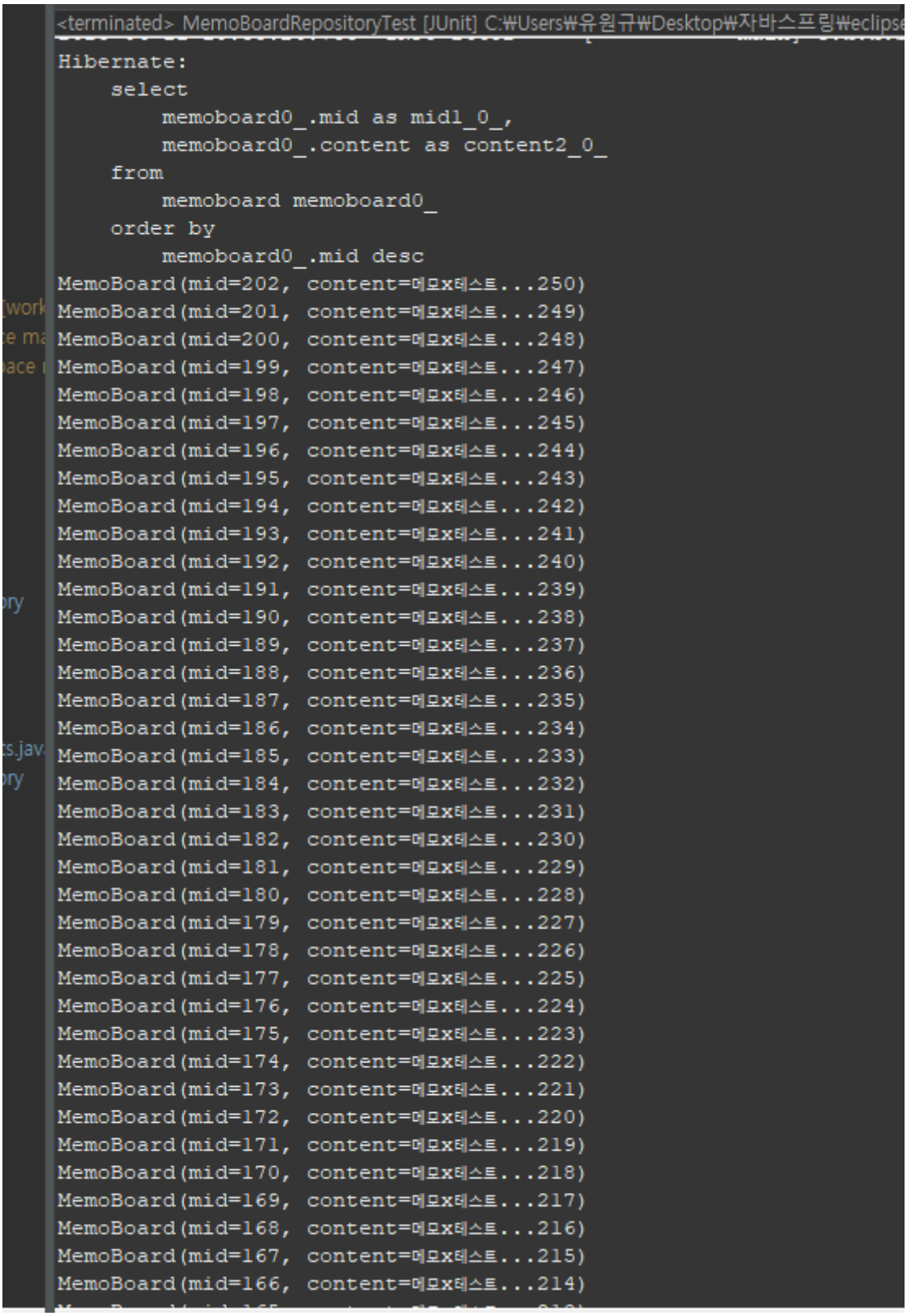
```
Hibernate:
    select
        memoboard0_.mid as mid1_0_,
        memoboard0_.content as content2_0_
    from
        memoboard memoboard0_
    where
        memoboard0_.mid between ? and ?
```

JPA 실습 13번



```
2023-04-11 10:32:57.366  INFO 12540 --- [
Hibernate:
    select
        memoboard0_.mid as mid1_0_,
        memoboard0_.content as content2_0_
    from
        memoboard memoboard0_
    where
        memoboard0_.mid between ? and ?
    order by
        memoboard0_.mid desc limit ?
Hibernate:
    select
        count(memoboard0_.mid) as col_0_0_
    from
        memoboard memoboard0_
    where
        memoboard0_.mid between ? and ?
```

JPA 실습 14번


<terminated> MemoBoardRepositoryTest [JUnit] C:₩Users₩유원규₩Desktop₩자바스프링₩eclipse
Hibernate:
select
memoboard0_.mid as mid1_0_,
memoboard0_.content as content2_0_
from
memoboard memoboard0_
order by
memoboard0_.mid desc
MemoBoard(mid=202, content=메모x테스트...250)
MemoBoard(mid=201, content=메모x테스트...249)
MemoBoard(mid=200, content=메모x테스트...248)
MemoBoard(mid=199, content=메모x테스트...247)
MemoBoard(mid=198, content=메모x테스트...246)
MemoBoard(mid=197, content=메모x테스트...245)
MemoBoard(mid=196, content=메모x테스트...244)
MemoBoard(mid=195, content=메모x테스트...243)
MemoBoard(mid=194, content=메모x테스트...242)
MemoBoard(mid=193, content=메모x테스트...241)
MemoBoard(mid=192, content=메모x테스트...240)
MemoBoard(mid=191, content=메모x테스트...239)
MemoBoard(mid=190, content=메모x테스트...238)
MemoBoard(mid=189, content=메모x테스트...237)
MemoBoard(mid=188, content=메모x테스트...236)
MemoBoard(mid=187, content=메모x테스트...235)
MemoBoard(mid=186, content=메모x테스트...234)
MemoBoard(mid=185, content=메모x테스트...233)
MemoBoard(mid=184, content=메모x테스트...232)
MemoBoard(mid=183, content=메모x테스트...231)
MemoBoard(mid=182, content=메모x테스트...230)
MemoBoard(mid=181, content=메모x테스트...229)
MemoBoard(mid=180, content=메모x테스트...228)
MemoBoard(mid=179, content=메모x테스트...227)
MemoBoard(mid=178, content=메모x테스트...226)
MemoBoard(mid=177, content=메모x테스트...225)
MemoBoard(mid=176, content=메모x테스트...224)
MemoBoard(mid=175, content=메모x테스트...223)
MemoBoard(mid=174, content=메모x테스트...222)
MemoBoard(mid=173, content=메모x테스트...221)
MemoBoard(mid=172, content=메모x테스트...220)
MemoBoard(mid=171, content=메모x테스트...219)
MemoBoard(mid=170, content=메모x테스트...218)
MemoBoard(mid=169, content=메모x테스트...217)
MemoBoard(mid=168, content=메모x테스트...216)
MemoBoard(mid=167, content=메모x테스트...215)
MemoBoard(mid=166, content=메모x테스트...214)

JPA 실습 15번

```
2023-04-11 10:34:03.985  INFO 1674
Hibernate:
    update
        memoboard
    set
        content=?
    where
        mid=?
수정 결과 : 1
```

JPA 실습 16번

```
2023-04-11 10:35:25.198  WARN 31148 --
2023-04-11 10:35:25.932  INFO 31148 --
Hibernate:
    update
        memoboard
    set
        content=?
    where
        mid=?
수정 결과 : 1
2023-04-11 10:35:26.288  INFO 31148 --
```

JPA 실습 17번

```
2023-04-11 10:36:17.368  INFO 10968 --- [           main
Hibernate:
    select
        memoboard0_.mid as mid1_0_,
        memoboard0_.content as content2_0_
    from
        memoboard memoboard0_
    where
        memoboard0_.mid>?
    order by
        memoboard0_.mid desc limit ?
Hibernate:
    select
        count(memoboard0_.mid) as col_0_0_
    from
```

JPA 실습 18번

```
Hibernate:
    select
        memoboard0_.mid as col_0_0_,
        memoboard0_.content as col_1_0_,
        CURRENT_DATE as col_2_0_
    from
        memoboard memoboard0_
    where
        memoboard0_.mid>?
    order by
        memoboard0_.mid desc limit ?
Hibernate:
    select
        count(memoboard0_.mid) as col_0_0_
    from
        memoboard memoboard0_
    where
        memoboard0_.mid>?
```

JPA 실습 19번

```
Hibernate:
    select
        *
    from
        memoboard
    where
        content like '%7%'
MemoBoard(mid=7, content=메모....7)
MemoBoard(mid=17, content=메모....17)
MemoBoard(mid=27, content=메모....27)
MemoBoard(mid=37, content=메모....37)
MemoBoard(mid=47, content=메모....47)
MemoBoard(mid=57, content=메모....57)
MemoBoard(mid=67, content=메모....67)
MemoBoard(mid=70, content=메모....70)
MemoBoard(mid=71, content=메모....71)
MemoBoard(mid=72, content=메모....72)
MemoBoard(mid=73, content=메모....73)
MemoBoard(mid=74, content=메모....74)
```

실습 후기

여태까지 DAO를 사용해서 DBCP 혹은 JDBC API를 사용해서 DB와 연결하고 SQL문을 직접 작성을 하였습니다.

하지만 이번 기회에 JPA를 사용하고 JDBC API , JPA를 사용해서 DB와 연결을 하는 방법을 사용해보았습니다.

실제로 코드를 최소 몇백줄을 적었던 작업들이 JPA를 사용함으로 백줄 안으로 작성이 되는 모습을 살필 수 있었습니다.

아직 자세하게 JPA를 사용하기 위한 메소드 정의 방법이나 구현 방법에 대해 완벽하게 터득했다고는 할 수는 없겠지만 이번 실습을 통해서 대략적으로 JPA가 무엇인지 JPA를 활용해서 어떻게 SQL문을 작성해야하는지에 대해서 공부를 할 수 있는 기회였던거 같습니다.

캡스톤을 진행할때 몇백줄을 적는것보다 JPA에 대해 공부를 해서 작업량을 줄이는게 더 효율적일거 같다는 생각이 들었고 추후 부족한 부분에 대해서는 공부를 병행하면서 캡스톤에 해당 기술을 사용하면 좋을거 같다는 생각이 들었습니다.

그리고 실습을 하는과정에서 Lombok을 추가하는 과정에서 에러 사항이 존재했습니다.

롬복을 cmd창에서 명령어를 쳐서 롬복을 실행시킨후 exe파일에 적용을 하면 이클립스init 메모장에 lombok 경로가 추가가 되는데

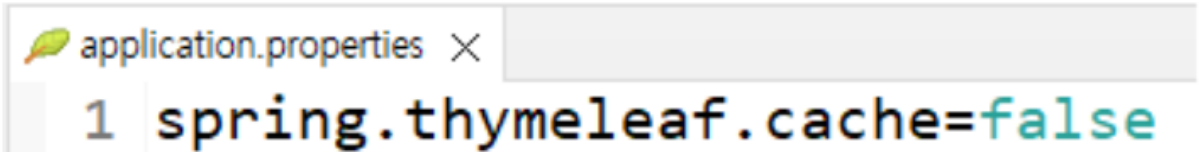
롬복 경로에 한글이 추가가 되면 안되는 문제가 존재했습니다. 그래서 이런 문제때문에 이클립스를 끈 다음에 롬복 적용후 재실행을 할려고하면 실행이 안되는 문제때문에 2시간정도 이것저것 에러사항을 찾을려고 시도를 했습니다.

그래서 메모장에 경로에 한글이 포함되지않도록 바꿔준후 해당경로에 lombok.jar파일을 이동하여서 인식이 되도록 설정을하여서 에러를 고치게되었습니다.

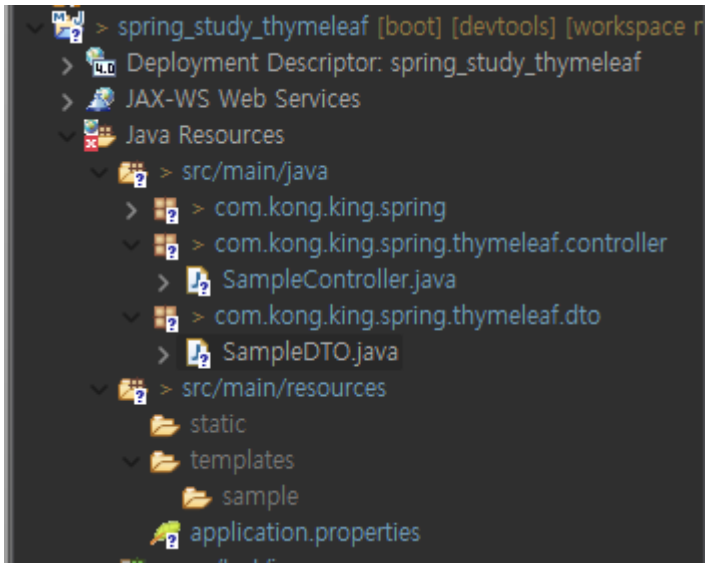
3장 Thymeleaf 기초

html 파일안에다가 Thymealf를 사용한다.

변경후에 만들어진 결과를 캐싱(보관) 하지 않도록 설정



- application.properties에 해당 설정을 해주어야한다.



- 기존에는 WEB-INF에 폴더 생성해서 관리를 했는데 thymeleaf를 사용을 한다면 src/main/resources안에 템플릿 폴더안에 타임리프 파일을 관리를 하면된다.

html 파일 생성

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org" lang="ko">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
<h1 th:text="${'Hello World'}"></h1>

</body>
</html>
```

- 이렇게 작성을하면 html xmlns 문구를 사용해서 타임리프 org를 땡겨오게되면 해당페이지를 타임리프를 사용할 수 있도록 지원이 된다.

DTO 파일 생성

```
package com.kong.king.spring.thymeleaf.dto;

import java.time.LocalDateTime;

import lombok.Builder;
import lombok.Data;

@Data
@Builder(toBuilder = true)
public class SampleDTO {
```

```
    private Long sno;
    private String name;
    private String dept;
    private LocalDateTime regTime;
}
```

- 여기서 @Data 어노테이션은 Setter, Getter, toString등 어노테이션이 다포함되어있다.

컨트롤러 파일 생성

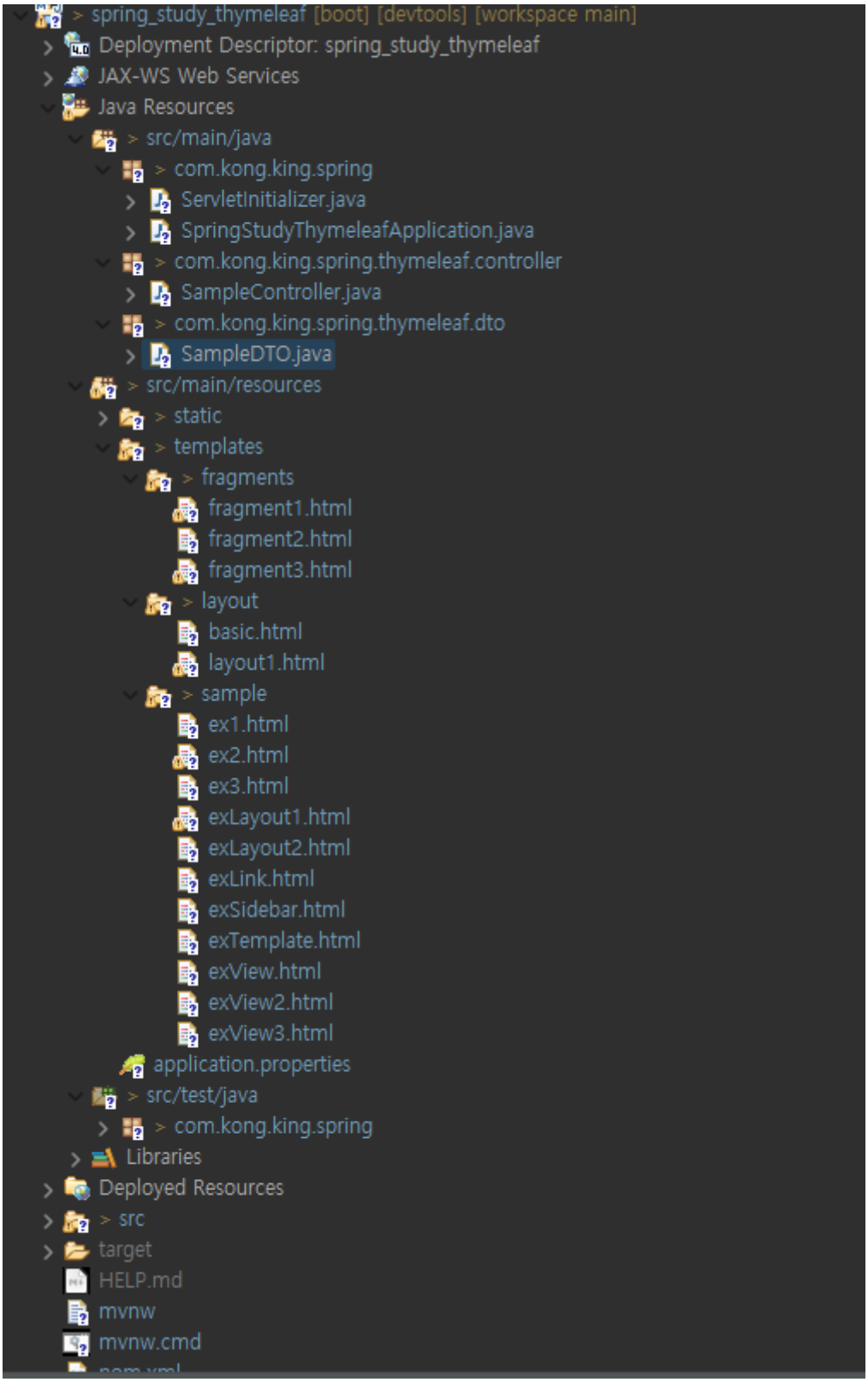
```
package com.kong.king.spring.thymeleaf.controller;

import java.time.LocalDateTime;
import java.util.List;
import java.util.stream.Collectors;
import java.util.stream.IntStream;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import com.kong.king.spring.thymeleaf.dto.SampleDTO;

import lombok.extern.log4j.Log4j2;

@Controller
@RequestMapping("/sample")
@Log4j2
public class SampleController {

    @GetMapping("/ex1")
    public void ex1() {
        log.info("ex1.....");
    }

    @GetMapping({"/ex2", "/exLink"})
    public void exModel(Model model) {
        List<SampleDTO> list = IntStream.rangeClosed(1, 20).asLongStream()
        .mapToObj(i -> {
            SampleDTO dto = SampleDTO.builder()
                    .sno(i)
                    .name("홍길동.." + i % 3)
                    .dept("컴퓨터.." + i % 6)
                    .regTime(LocalDateTime.now())
                    .build();
            return dto;
```

```
        }).collect(Collectors.toList());

        model.addAttribute("list", list);
        log.info("ex2.....");
    }

    @GetMapping({"/exInline"})
    public String exInline(RedirectAttributes redirectAttributes) {

        log.info("exInline.........");

        SampleDTO dto = SampleDTO.builder()
                .sno(100L)
                .name("박철수.." + 100)
                .dept("인공지능.." + 100)
                .regTime(LocalDateTime.now())
                .build();
        redirectAttributes.addFlashAttribute("result","success");
        redirectAttributes.addFlashAttribute("dto",dto);

        return "redirect:/sample/ex3";
    }

    @GetMapping("/ex3")
    public void ex3() {
        log.info("ex3.........");
    }

    @GetMapping("/exView")
    public void exView() {
        log.info("exView.........");
    }

    @GetMapping("/exView2")
    public void exView2(@RequestParam(value="sno", required=false) String sno, Model
m) {

        m.addAttribute("sno", sno);
        log.info("exView2 with RequestParam.........");
    }

    @GetMapping("/exView3/{sno}")
    public String exView3(@PathVariable String sno, Model m) {

        m.addAttribute("sno", sno);
        log.info("exView3 with PathVariable.........");
        return "sample/exView3";
    }

    @GetMapping({"/exLayout1", "/exLayout2", "/exTemplate", "/exSidebar"})
    public void exLayout1() {
        log.info("exLayout1.........");
```

```
    }
}
```

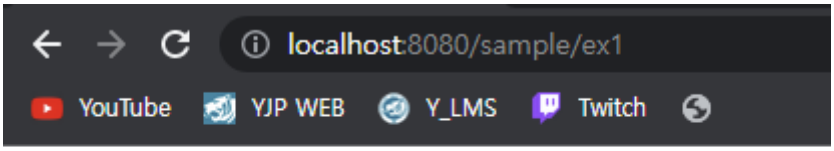
- 컨트롤러 전체 코드입니다.

파일 구조



ex1.html실습

```
<!DOCTYPE html>
<html xmlns:th = "http://www.thymeleaf.org" lang="ko">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <h1 th:text="${'Hello World'}"></h1>
</body>
</html>
```



Hello World

ex2.html 실습

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <h1>Thymeleaf each 반복문 제어구조</h1>
    <p>
    <ul>
        <li th:each="dto : ${list}">
            [[${dto}]]
        </li>
    </ul>
    <hr>

    <h1>Thymeleaf each 반복문 상태객체 활용</h1>
    <p>
    <h2>순번, 인덱스 번호, 홀수/짝수 등 지정</h2>
    <p>
    <ul>
```

```
        <li th:each="dto, state : ${list}">
            [[${state.index}]] --- [[${dto}]]
        </li>
    </ul>
    <hr>

    <h1>Thymeleaf if 제어문</h1>
    <h2>sno가 5의 배수만 출력</h2>
    <p>
    <ul>
        <li th:each="dto, state : ${list}" th:if="${dto.sno % 5 == 0}">
            [[${dto}]]
        </li>
    </ul>
    <hr>

    <h1>Thymeleaf if, unless제어구조</h1>
    <h2>sno가 5로 나눈 나머지가 0일 경우 sno만 출력, 그렇지 않다면 SampleDTO의 name 출력
</h2>
    <p>
    <ul>
        <li th:each="dto, state : ${list}">
        <span th:if="${dto.sno % 5 == 0}" th:text="${dto.sno + '----------' +
dto.sno}"></span>
        <span th:unless="${dto.sno % 5 == 0}" th:text="${dto.sno + ' : ' +
dto.name}"></span>
        </li>
    </ul>
    <hr>

    <h1>Thymeleaf 삼항 연산자 제어구조1</h1>
    <p>
    <ul>
        <li th:each="dto, state : ${list}" th:text="${dto.sno % 3 == 0} ?
${dto.sno}">
        </li>
    </ul>
    <hr>

    <h1>Thymeleaf 삼항 연산자 제어구조2</h1>
    <p>
    <ul>
        <li th:each="dto, state : ${list}" th:text="${dto.sno % 3 == 0} ? ${dto.sno}
: ${dto.name}">
        </li>
    </ul>
    <hr>

    <h1>Thymeleaf 삼항 연산자 제어구조3 - 특정상황 CSS반영</h1>
    <p>
    <style>
        .target{
```

```
            background-color: red;
        }
    </style>
    <ul>
        <li th:each="dto, state : ${list}" th:class="${dto.sno % 5 == 0} ? 'target'"
th:text="${dto}">
        </li>
    </ul>
    <hr>

    <h1>Thymeleaf th:block 활용</h1>
    <p>
    <ul>
        <th:block th:each="dto: ${list}">
            <li th:text="${dto.sno % 5 == 0} ? ${dto.sno} : ${dto.name}">
            </li>
        </th:block>
    </ul>
    <hr>

    <h3>Thymeleaf 기본객체 #numbers 활용</h3>
    <p>
    <ul>
        <li th:each="dto : ${list}">
            [[${#numbers.formatInteger(dto.sno,5)}]]
        </li>
    </ul>
    <hr>

    <h3>Thymeleaf #temporals 활용, 날짜 포맷 설정</h3>
    <p>
    <ul>
        <li th:each="dto : ${list}">
            [[${dto.sno}]] --- [[${#temporals.format(dto.regTime,'yyyy/MM/dd')}]]
        </li>
    </ul>
    <hr>

</body>
</html>
```

Thymeleaf each 반복문 제어구조

- SampleDTO(sno=1, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=2, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=3, name=홍길동..0, dept=컴퓨터..3, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=4, name=홍길동..1, dept=컴퓨터..4, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=5, name=홍길동..2, dept=컴퓨터..5, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=6, name=홍길동..0, dept=컴퓨터..0, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=7, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=8, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=9, name=홍길동..0, dept=컴퓨터..3, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=10, name=홍길동..1, dept=컴퓨터..4, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=11, name=홍길동..2, dept=컴퓨터..5, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=12, name=홍길동..0, dept=컴퓨터..0, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=13, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=14, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=15, name=홍길동..0, dept=컴퓨터..3, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=16, name=홍길동..1, dept=컴퓨터..4, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=17, name=홍길동..2, dept=컴퓨터..5, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=18, name=홍길동..0, dept=컴퓨터..0, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=19, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=20, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T16:40:04.554580500)

Thymeleaf each 반복문 상태객체 활용

순번, 인덱스 번호, 홀수/짝수 등 지정

- 0 --- SampleDTO(sno=1, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T16:40:04.554580500)
- 1 --- SampleDTO(sno=2, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T16:40:04.554580500)
- 2 --- SampleDTO(sno=3, name=홍길동..0, dept=컴퓨터..3, regTime=2023-04-14T16:40:04.554580500)
- 3 --- SampleDTO(sno=4, name=홍길동..1, dept=컴퓨터..4, regTime=2023-04-14T16:40:04.554580500)
- 4 --- SampleDTO(sno=5, name=홍길동..2, dept=컴퓨터..5, regTime=2023-04-14T16:40:04.554580500)
- 5 --- SampleDTO(sno=6, name=홍길동..0, dept=컴퓨터..0, regTime=2023-04-14T16:40:04.554580500)
- 6 --- SampleDTO(sno=7, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T16:40:04.554580500)
- 7 --- SampleDTO(sno=8, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T16:40:04.554580500)
- 8 --- SampleDTO(sno=9, name=홍길동..0, dept=컴퓨터..3, regTime=2023-04-14T16:40:04.554580500)
- 9 --- SampleDTO(sno=10, name=홍길동..1, dept=컴퓨터..4, regTime=2023-04-14T16:40:04.554580500)
- 10 --- SampleDTO(sno=11, name=홍길동..2, dept=컴퓨터..5, regTime=2023-04-14T16:40:04.554580500)
- 11 --- SampleDTO(sno=12, name=홍길동..0, dept=컴퓨터..0, regTime=2023-04-14T16:40:04.554580500)
- 12 --- SampleDTO(sno=13, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T16:40:04.554580500)
- 13 --- SampleDTO(sno=14, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T16:40:04.554580500)
- 14 --- SampleDTO(sno=15, name=홍길동..0, dept=컴퓨터..3, regTime=2023-04-14T16:40:04.554580500)
- 15 --- SampleDTO(sno=16, name=홍길동..1, dept=컴퓨터..4, regTime=2023-04-14T16:40:04.554580500)
- 16 --- SampleDTO(sno=17, name=홍길동..2, dept=컴퓨터..5, regTime=2023-04-14T16:40:04.554580500)
- 17 --- SampleDTO(sno=18, name=홍길동..0, dept=컴퓨터..0, regTime=2023-04-14T16:40:04.554580500)
- 18 --- SampleDTO(sno=19, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T16:40:04.554580500)
- 19 --- SampleDTO(sno=20, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T16:40:04.554580500)

Thymeleaf if 제어문

sno가 5의 배수만 출력

- SampleDTO(sno=5, name=홍길동..2, dept=컴퓨터..5, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=10, name=홍길동..1, dept=컴퓨터..4, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=15, name=홍길동..0, dept=컴퓨터..3, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=20, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T16:40:04.554580500)

Thymeleaf if, unless제어구조

sno가 5로 나눈 나머지가 0일 경우 sno만 출력, 그렇지 않다면 SampleDTO의 name 출력

- 1 : 홍길동..1
- 2 : 홍길동..2
- 3 : 홍길동..0
- 4 : 홍길동..1
- 5----------5
- 6 : 홍길동..0
- 7 : 홍길동..1
- 8 : 홍길동..2
- 9 : 홍길동..0
- 10----------10
- 11 : 홍길동..2
- 12 : 홍길동..0
- 13 : 홍길동..1
- 14 : 홍길동..2
- 15----------15
- 16 : 홍길동..1
- 17 : 홍길동..2
- 18 : 홍길동..0
- 19 : 홍길동..1
- 20----------20

Thymeleaf 삼항 연산자 제어구조1

-
-
- 3
-
-
- 6
-
-
- 9
-
-
- 12
-
-
- 15
-
-
- 18
-
-

Thymeleaf 삼항 연산자 제어구조2

- 홍길동..1
- 홍길동..2
- 3
- 홍길동..1
- 홍길동..2
- 6
- 홍길동..1
- 홍길동..2
- 9
- 홍길동..1
- 홍길동..2
- 12
- 홍길동..1
- 홍길동..2
- 15
- 홍길동..1
- 홍길동..2
- 18
- 홍길동..1
- 홍길동..2

Thymeleaf 삼항 연산자 제어구조3 - 특정상황 CSS반영

- SampleDTO(sno=1, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=2, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=3, name=홍길동..0, dept=컴퓨터..3, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=4, name=홍길동..1, dept=컴퓨터..4, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=5, name=홍길동..2, dept=컴퓨터..5, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=6, name=홍길동..0, dept=컴퓨터..0, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=7, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=8, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=9, name=홍길동..0, dept=컴퓨터..3, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=10, name=홍길동..1, dept=컴퓨터..4, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=11, name=홍길동..2, dept=컴퓨터..5, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=12, name=홍길동..0, dept=컴퓨터..0, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=13, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=14, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=15, name=홍길동..0, dept=컴퓨터..3, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=16, name=홍길동..1, dept=컴퓨터..4, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=17, name=홍길동..2, dept=컴퓨터..5, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=18, name=홍길동..0, dept=컴퓨터..0, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=19, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T16:40:04.554580500)
- SampleDTO(sno=20, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T16:40:04.554580500)

Thymeleaf th:block 활용

- 홍길동..1
- 홍길동..2
- 홍길동..0
- 홍길동..1
- 5
- 홍길동..0
- 홍길동..1
- 홍길동..2
- 홍길동..0
- 10
- 홍길동..2
- 홍길동..0
- 홍길동..1
- 홍길동..2
- 15
- 홍길동..1
- 홍길동..2
- 홍길동..0
- 홍길동..1
- 20

Thymeleaf 기본객체 #numbers 활용

- 00001
- 00002
- 00003
- 00004
- 00005
- 00006
- 00007
- 00008
- 00009
- 00010
- 00011
- 00012
- 00013
- 00014
- 00015
- 00016
- 00017
- 00018
- 00019
- 00020

Thymeleaf #temporals 활용, 날짜 포맷 설정

- 1 --- 2023/04/14
- 2 --- 2023/04/14
- 3 --- 2023/04/14
- 4 --- 2023/04/14
- 5 --- 2023/04/14
- 6 --- 2023/04/14
- 7 --- 2023/04/14
- 8 --- 2023/04/14
- 9 --- 2023/04/14
- 10 --- 2023/04/14
- 11 --- 2023/04/14
- 12 --- 2023/04/14
- 13 --- 2023/04/14
- 14 --- 2023/04/14
- 15 --- 2023/04/14
- 16 --- 2023/04/14
- 17 --- 2023/04/14
- 18 --- 2023/04/14
- 19 --- 2023/04/14
- 20 --- 2023/04/14

---

ex3.html 실습

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org" lang="en">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <h1 th:text="${result}"></h1>
    <h1 th:text="${dto}"></h1>
```

```
    <script th:inline="javascript">

        var msg = [[${result}]];
        var dto = [[${dto}]];
    </script>

</body>
</html>
```

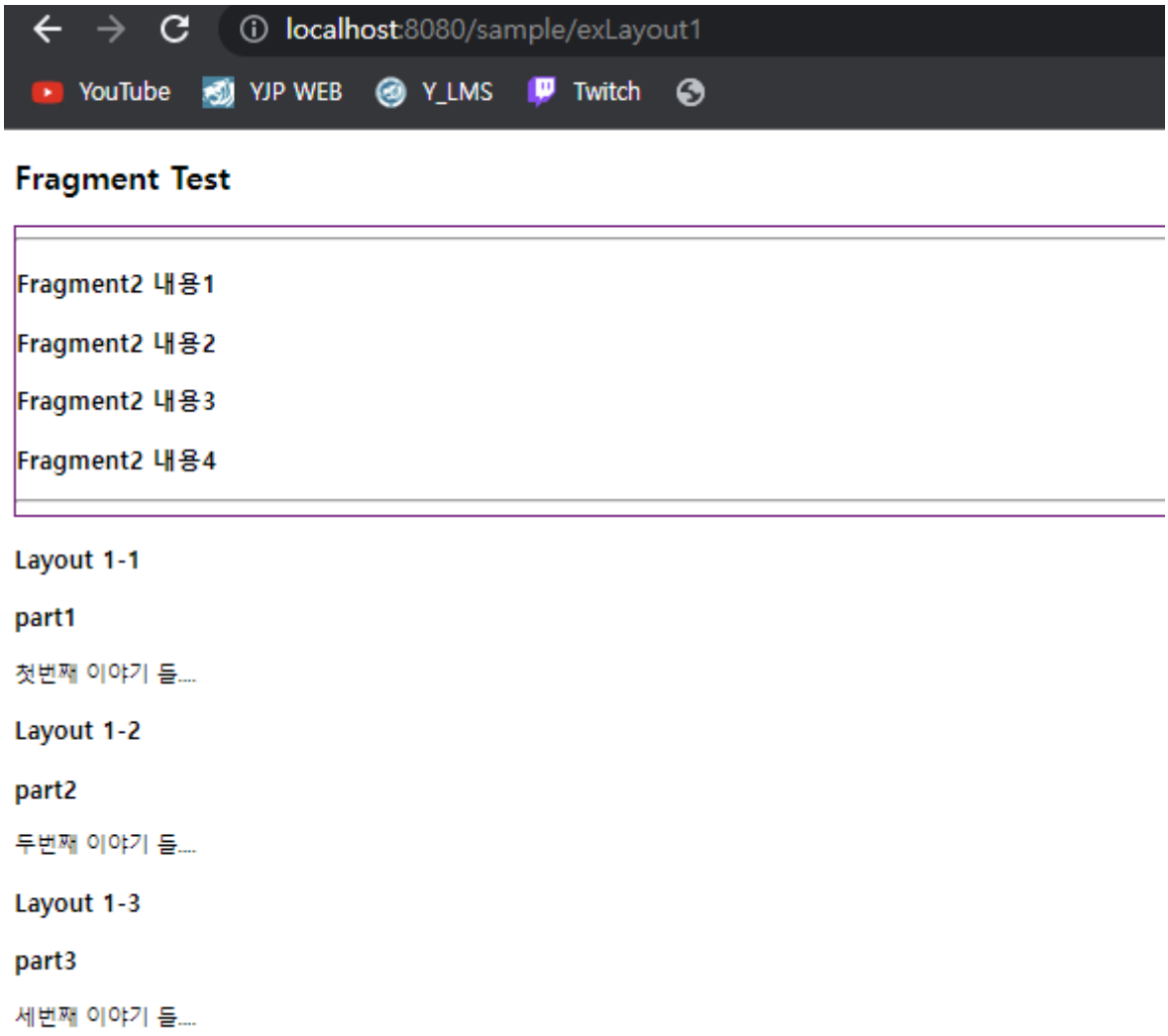
---

success

SampleDTO(sno=100, name=박철수..100, dept=인공지능..100, regTime=2023-04-14T16:41:39.948414400)

exLayout1.html 실습

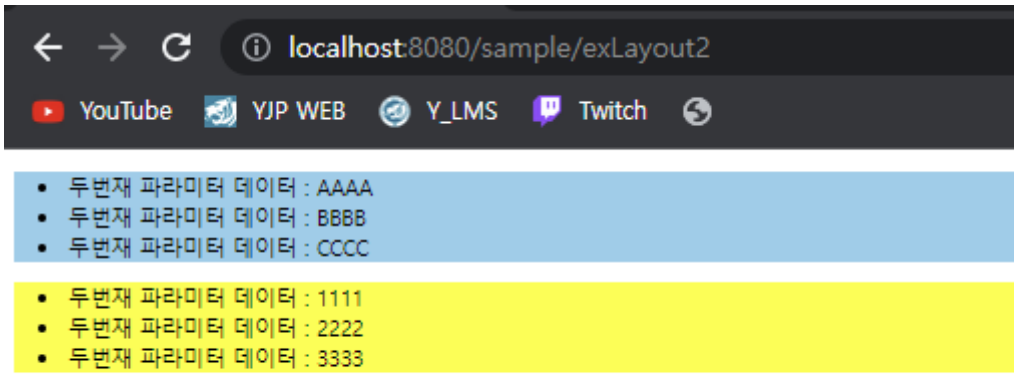
```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <h2>Fragment Test</h2>

    <div style="border : 1px solid purple">
        <th:block th:replace="~{/fragments/fragment2}"></th:block>
    </div>

    <h3>Layout 1-1</h3>
    <div th:replace="~{/fragments/fragment1 :: part1}"></div>

    <h3>Layout 1-2</h3>
    <div th:replace="~{/fragments/fragment1 :: part2}"></div>

    <h3>Layout 1-3</h3>
    <th:block th:replace="~{/fragments/fragment1 :: part3}"></th:block>
</body>
</html>
```



exLayout2.html 실습

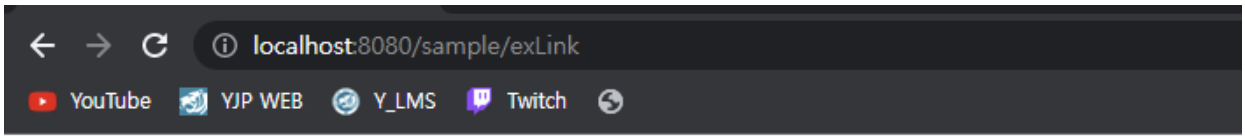
```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org" lang="ko">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <th:block th:replace="~{fragments/fragment3:: target(~{this:: #ulFirst}, ~
{this::#ulSecond})}">
        <ul id="ulFirst">
            <li>두번재 파라미터 데이터 : AAAA</li>
            <li>두번재 파라미터 데이터 : BBBB</li>
            <li>두번재 파라미터 데이터 : CCCC</li>
        </ul>

        <ul id="ulSecond">
            <li>두번재 파라미터 데이터 : 1111</li>
```

```
            <li>두번재 파라미터 데이터 : 2222</li>
            <li>두번재 파라미터 데이터 : 3333</li>
        </ul>
    </th:block>
</body>
</html>
```



exLink.html 실습

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org" lang="ko">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <h1>Thymeleaf 링크 처리 : th:href 활용 예제</h1>
    <p>
    <ul>
        <li th:each="dto : ${list}">
            <a th:href="@{/sample/exView}">[[${dto}]]</a>
        </li>
    </ul>
    <hr>

    <h1>Thymeleaf 링크 처리 : th:href 활용 예제 - path이용 파라미터 추가</h1>
    <p>
    <ul>
        <li th:each="dto : ${list}">
            <a th:href="@{/sample/exView3/{sno}(sno=${dto.sno})}">[[${dto}]]</a>
        </li>
    </ul>
</body>
</html>
```



Thymeleaf 링크 처리 : th:href 활용 예제

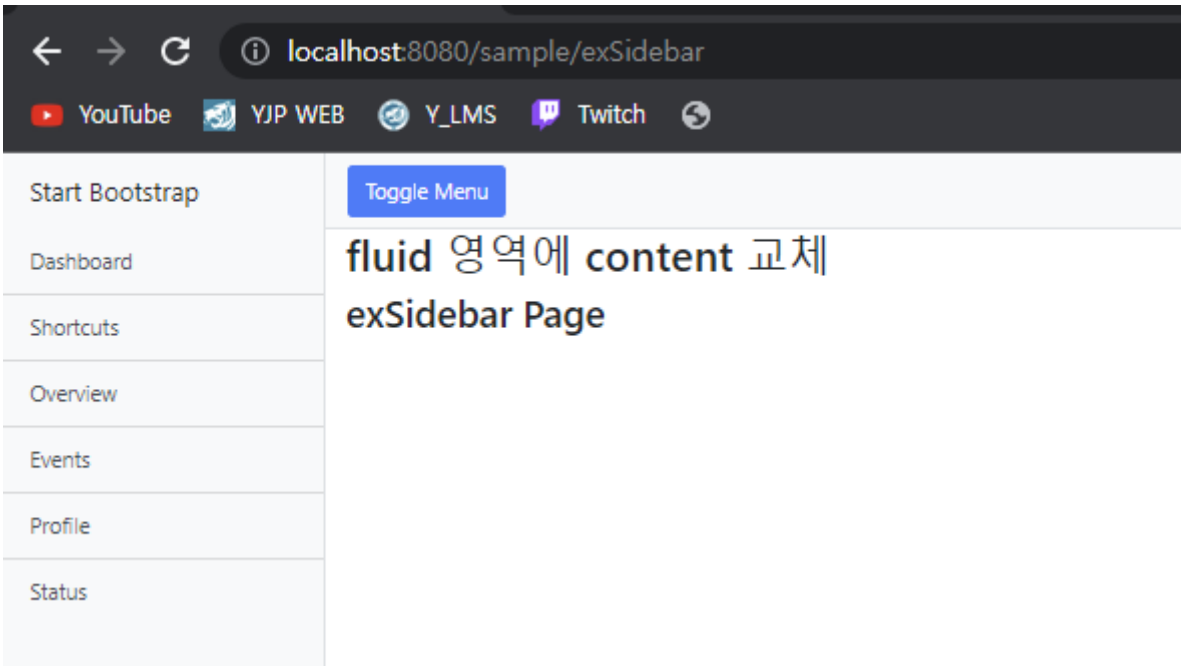
- SampleDTO(sno=1, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=2, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=3, name=홍길동..0, dept=컴퓨터..3, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=4, name=홍길동..1, dept=컴퓨터..4, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=5, name=홍길동..2, dept=컴퓨터..5, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=6, name=홍길동..0, dept=컴퓨터..0, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=7, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=8, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=9, name=홍길동..0, dept=컴퓨터..3, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=10, name=홍길동..1, dept=컴퓨터..4, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=11, name=홍길동..2, dept=컴퓨터..5, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=12, name=홍길동..0, dept=컴퓨터..0, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=13, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=14, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=15, name=홍길동..0, dept=컴퓨터..3, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=16, name=홍길동..1, dept=컴퓨터..4, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=17, name=홍길동..2, dept=컴퓨터..5, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=18, name=홍길동..0, dept=컴퓨터..0, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=19, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=20, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T17:14:27.112584900)

Thymeleaf 링크 처리 : th:href 활용 예제 - path이용 파라미터 추가

- SampleDTO(sno=1, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=2, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=3, name=홍길동..0, dept=컴퓨터..3, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=4, name=홍길동..1, dept=컴퓨터..4, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=5, name=홍길동..2, dept=컴퓨터..5, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=6, name=홍길동..0, dept=컴퓨터..0, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=7, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=8, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=9, name=홍길동..0, dept=컴퓨터..3, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=10, name=홍길동..1, dept=컴퓨터..4, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=11, name=홍길동..2, dept=컴퓨터..5, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=12, name=홍길동..0, dept=컴퓨터..0, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=13, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=14, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=15, name=홍길동..0, dept=컴퓨터..3, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=16, name=홍길동..1, dept=컴퓨터..4, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=17, name=홍길동..2, dept=컴퓨터..5, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=18, name=홍길동..0, dept=컴퓨터..0, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=19, name=홍길동..1, dept=컴퓨터..1, regTime=2023-04-14T17:14:27.112584900)
- SampleDTO(sno=20, name=홍길동..2, dept=컴퓨터..2, regTime=2023-04-14T17:14:27.112584900)

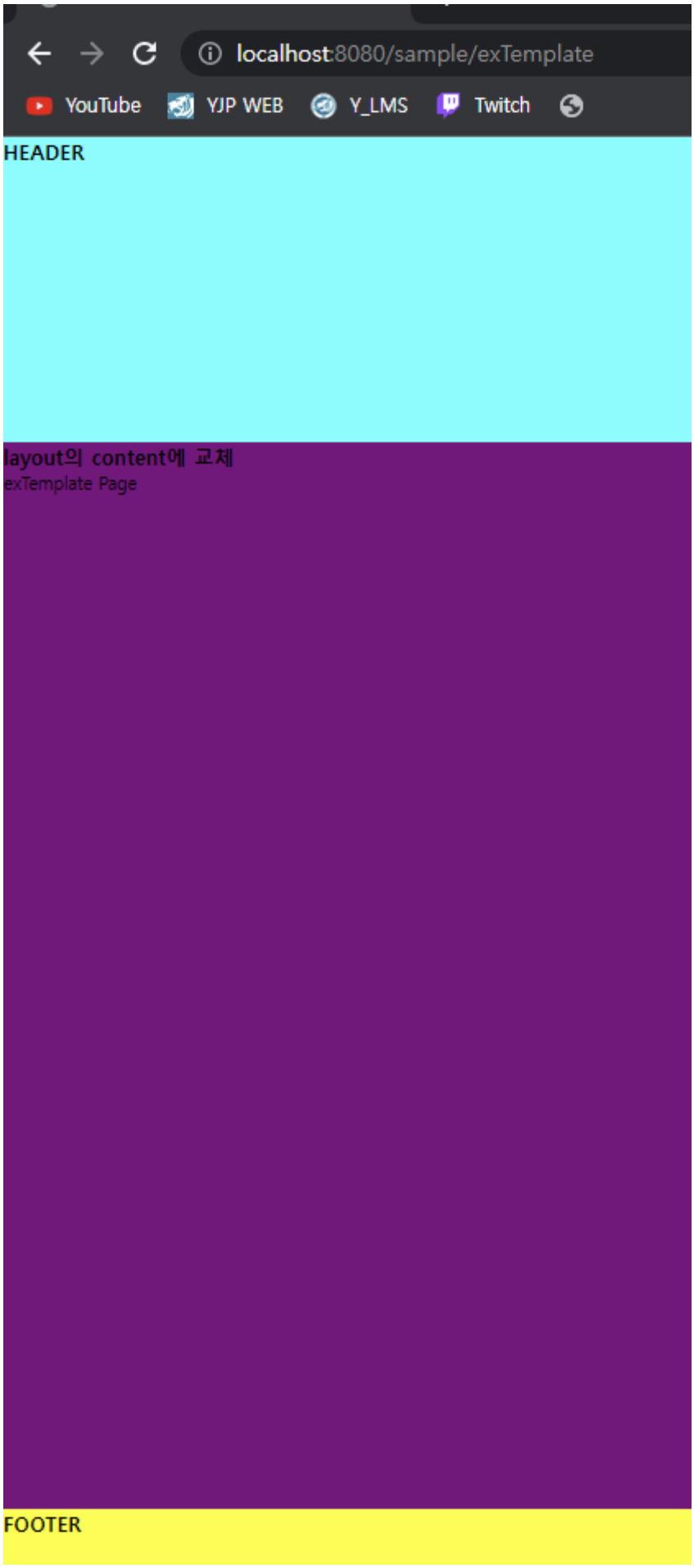
exSidebar.html실습

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org" lang="ko">

<th:block th:replace="~{/layout/basic :: setContent(~{this::content})}">
    <th:block th:fragment="content">
        <h2>fluid 영역에 content 교체</h2>
        <h3>exSidebar Page</h3>
    </th:block>
</th:block>
```



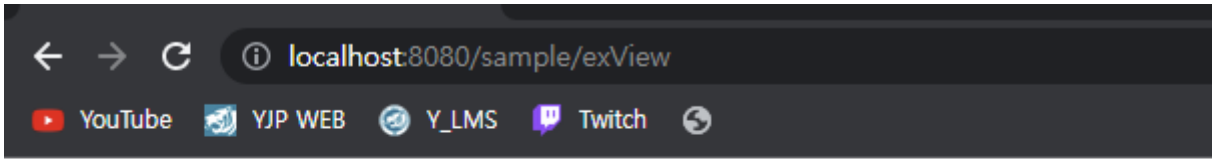
exTemplate.html 실습

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org" lang="ko">

<th:block th:replace="~{/layout/basic :: setContent(~{this::content})}">
    <th:block th:fragment="content">
        <h2>fluid 영역에 content 교체</h2>
        <h3>exSidebar Page</h3>
    </th:block>
</th:block>
```


localhost:8080/sample/exTemplate
YouTube
YJP WEB
Y_LMS
Twitch
HEADER
layout의 content에 교체
exTemplate Page
FOOTER

exView.html 실습

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org" lang="ko">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <h1>Thymeleaf : th:href 링크 테스트 : 링크 클릭 후 실행 화면</h1>
    <p>
    <a th:href="@{/sample/exLink}">링크실습예제(exLink)로 돌아가기</a>
</body>
</html>
```

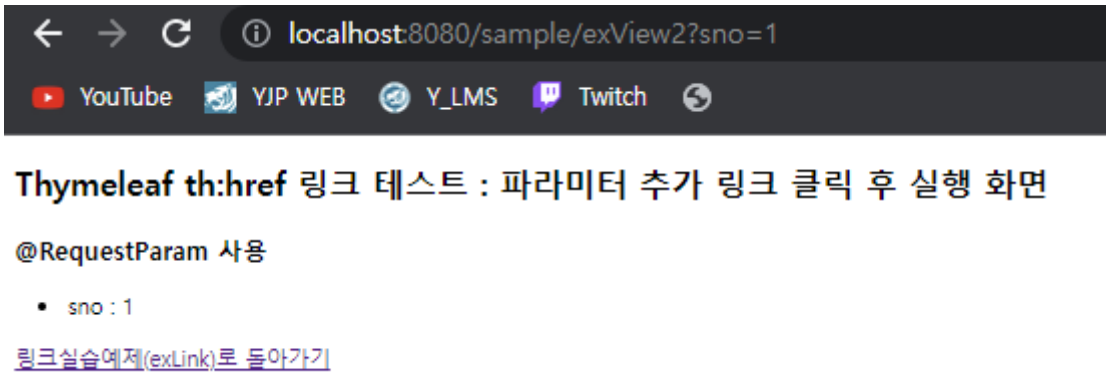


Thymeleaf : th:href 링크 테스트 : 링크 클릭 후 실행 화면

링크실습예제(exLink)로 돌아가기

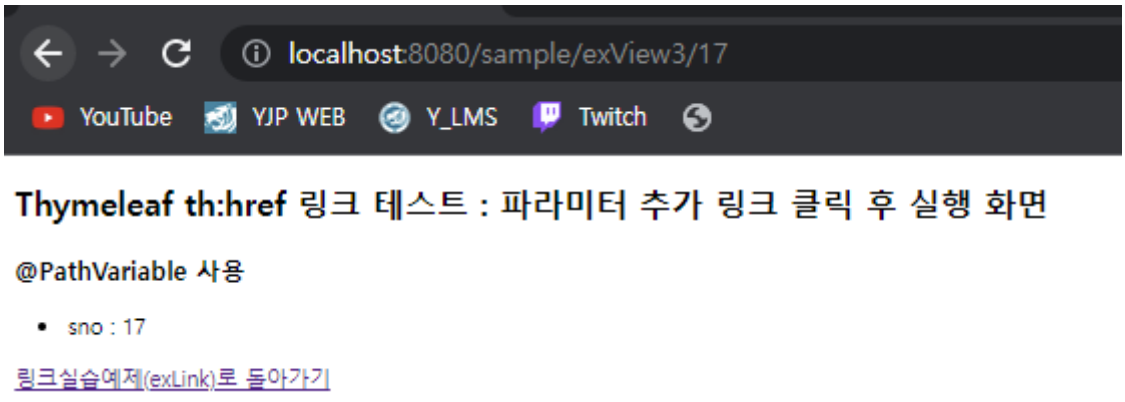
exView2.html 실습

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org" lang="ko">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <h2>Thymeleaf th:href 링크 테스트 : 파라미터 추가 링크 클릭 후 실행 화면</h2>
    <p>
    <h3>@RequestParam 사용</h3>
    <p>
    <ul>
        <li th:text="'sno : ' + ${sno}"></li>
    </ul>
    <p>
    <a th:href="@{/sample/exLink}">링크실습예제(exLink)로 돌아가기</a>
</body>
</html>
```



exView3.html 실습

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org" lang="ko">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <h2>Thymeleaf th:href 링크 테스트 : 파라미터 추가 링크 클릭 후 실행 화면</h2>
    <p>
    <h3>@PathVariable 사용</h3>
    <p>
    <ul>
        <li th:text="'sno : ' + ${sno}"></li>
    </ul>
    <p>
    <a th:href="@{/sample/exLink}">링크실습예제(exLink)로 돌아가기</a>
</body>
</html>
```



실습후기

일단 실습을 할때 ppt를 참고하여 코드를 작성하는 부분에 있어서 많은 오타가 존재했습니다.

그래서 오타를 찾아서 재 작성하는데 시간이 되게 많이 걸린 거같습니다.

이번기회에 Spring의 템플릿 엔진인 타임리프에 대해 공부를 하게 되면서 Jsp , jstl/el , thymeleaf 까지 자바의 view 역할이 가능한 아이들을 모두 써본거같습니다.

제일 편안했던건 타임리프 인거같고 제일 불편한건 jsp 형식이었던거같습니다.

기존에 코드를 하나하나 엄청나게 길게 작성을하였고 꺽세표같은것도 세세하게 신경을써서 작성을 하는등

if나 for문같은거작성할때도 대괄호를 잘 닫았나 안닫았나 체크하는등 여러 행동들이 필요했는데 jstl이나 thymeleaf같은 애들은 해당 태그안에 태그형식으로 집어넣으면서 해당 문제에 대해 덜 신경쓰게 되어서 좋았던거같습니다.

추후에 개인프로젝트를 진행할 생각인데 타임리프를 사용하거나 js중 vue나 react 를 사용해서 연동해서 사용하는 방법을 따로 개인프젝을 진행하는걸 염두에 두고있는데추후에 해당사항에 대해 다시공부를 해서 작성을하거나 버전관리 형식으로 개인프로젝트를 진행하는것에 대해 염두에 두고있습니다.