1. 스프링 MVC

1) 주요 구성요소

◆DispatcherServlet - 클라이언트의 요청을 받는다. 요청을 처리하는 중심자 역할 수행
[이미 구현되어져 있음. 기존의 Controller.java(servlet) 역할]
◆HandlerMapping - 클라이언트의 요청 URL을 맵핑
◆Controller - 클라이언트의 요청을 실제적으로 처리하는 클래스.
[@Controller를 붙여 생성. 각 Command와 내용]
◆ModelAndView - (커맨드객체) 비즈니스 로직에서 처리한 결과 데이터. 뷰로 전달
◆ViewResolver - 결과 페이지 생성
◆View - 결과 페이지

2) 처리흐름

A. DispatcherServlet가 클라이언트의 요청을 받는다.
B. 요청 URL을 Controller에서 검색 - HandlerMapping
C. DispatcherServlet가 Controller에 처리를 요청
D. 요청을 처리한 결과인 ModelAndView 리턴
E. DispatcherServlet가 결과 페이지를 ViewResolver에 생성 요청
F. 결과 페이지 출력

3) controller

일반 자바파일이다. 단, 클래스를 @Controller으로 지정해야 함.
클라이언트의 요청 종류에 따라 실행해야하는 메서드를 등록한다.

4) jsp 파일에 입력된 값을 받아오는 방법

1. 커맨드 객체(dto)를 사용해 파라미터를 통해 받아온다. (전체)

```
@PostMapping("/member/join")
public String join(Member m){
```

   - 시스템에서 Member 클래스로 이동
   - 해당 클래스 타입의 디폴트 생성자로 객체 생성
   - jsp파일에서 각 tag의 name을 보고, setName()을 호출하여, textbox에 입력한 값을 담아 Member 타입으로 생성한 객체에 값을 세팅해준다.

2. @RequestParam을 사용해 파라미터를 통해 받아온다. (일부)
3. HttpServletRequest req를 파라미터로 받아와서 각 값을 참조한다.

5) 커맨드 객체

◆Map, Model, ModelMap
자동으로 view로 전달
new로 생성하는 것x
파라메터 안에 넣으면 자동으로 생성해서 사용할 수 있음

```
@GetMapping("/test/map")
    public void testMap(Map<String,String> map){
        //map(command 객체)을 통해 자동으로 view page로 전달한다.
        //주의: command 객체는 반드시 파라미터에 넣어야 함(시스템에서 생성)
        map.put("test1", "aaa");
        map.put("test2", "bbb");
        map.put("test3", "ccc");
    }
```

2. 실습

1) 로그인/회원가입/수정/탈퇴

- login 구현하기

- MemberController: loginForm(), login()
- login.jsp

- edit
- logout
- out

2) 게시판

a. com.example.springApp1.model.board 패키지

- Board(dto) 클래스[dto]
- BoardMapper(interface)[dao]

b. static/mappers 폴더

- BoardMapper.xml[daoimpl]

cf.resultMap?

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="com.example.springApp1.model.board.BoardMapper">
    <resultMap type="com.example.springApp1.model.board.Board" id="board"> #id: resultmap이름. resultMap 다수여도 ok
        <result property="num" column="num" />
        <result property="writer" column="writer" />
        <result property="title" column="title" />
        <result property="w_date" column="w_date" />
        <result property="content" column="content" />
    </resultMap>

    # resultMap 사용예시
    <select id="selectById" resultMap="Board" parameterType="int"> # resultMap 설정시 위에서 지정한 id를 넣어준다.
        select * from member where id=#{id}
    </select>
```

*ArrayList의 경우 generic type을 resultMap에 써줌.

c. BoardService.java [serviceimpl]