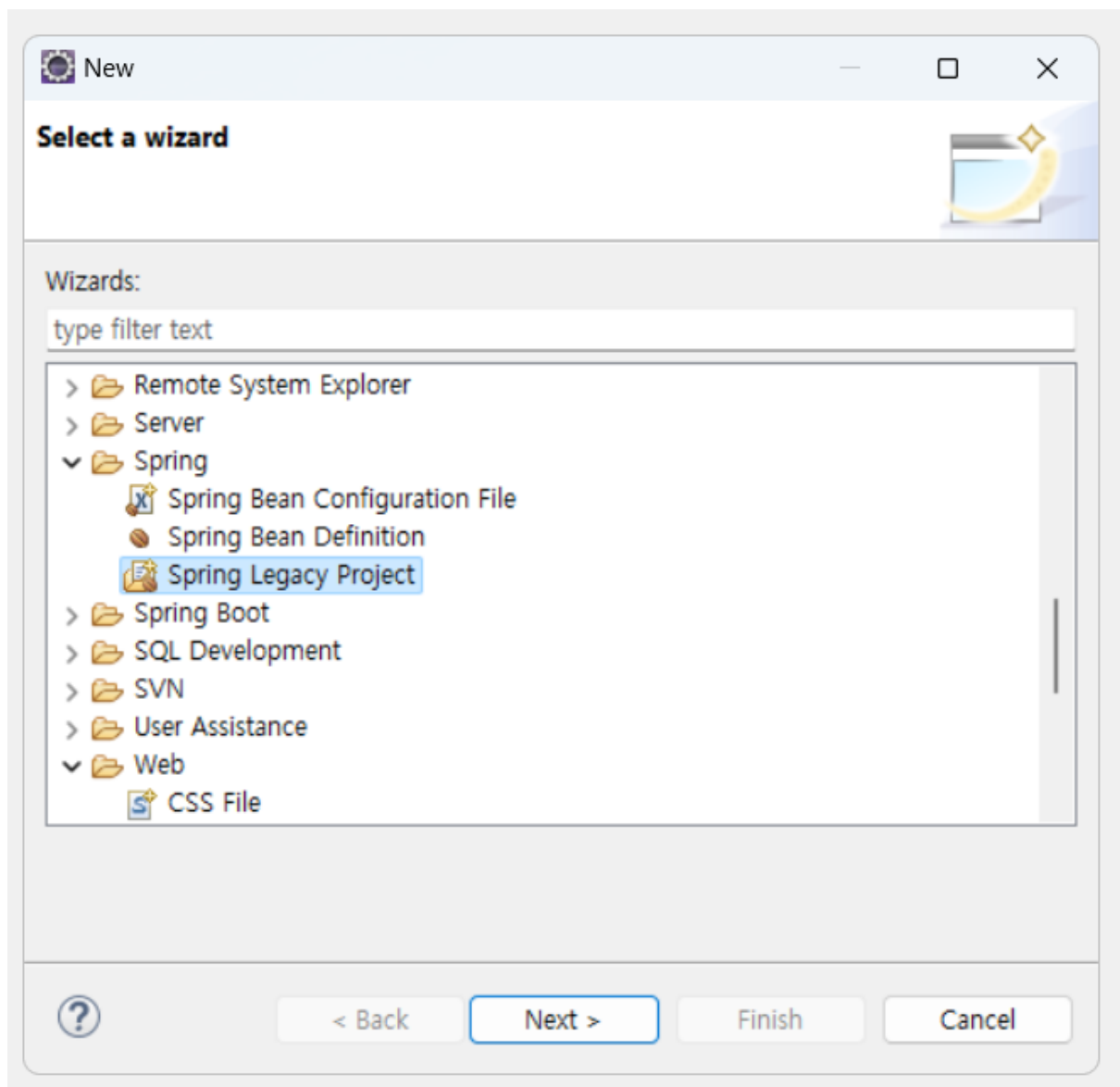


# 230227 Spring Day 1

날짜	@2023년 2월 27일
태그	spring

## SPRING Basic



New Spring Legacy Project

### Spring Legacy Project

Click 'Next' to load the template contents.

Project name:


☒ Use default location

Location:

Select Spring version:

Templates:

- Simple Projects
  - Simple Java
  - Simple Spring Maven
  - Simple Spring Web Maven
- Batch
- GemFire
- Integration
- Persistence
- Simple Spring Utility Project
- Spring MVC Project**

 requires downloading [Configure templates...](#)

Description:  
A new Spring MVC web application development project

URL: <https://dist.springsource.com/release/STS/help/org.springframework.templates.mvc-3.2.2.zip>

Working sets

☐ Add project to working sets

Working sets:

New Spring Legacy Project

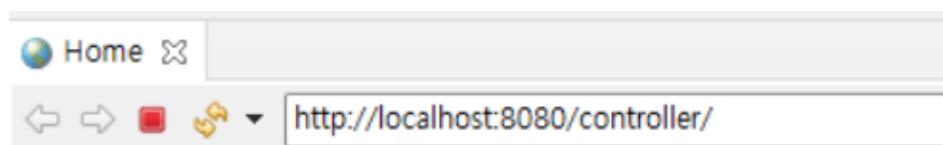
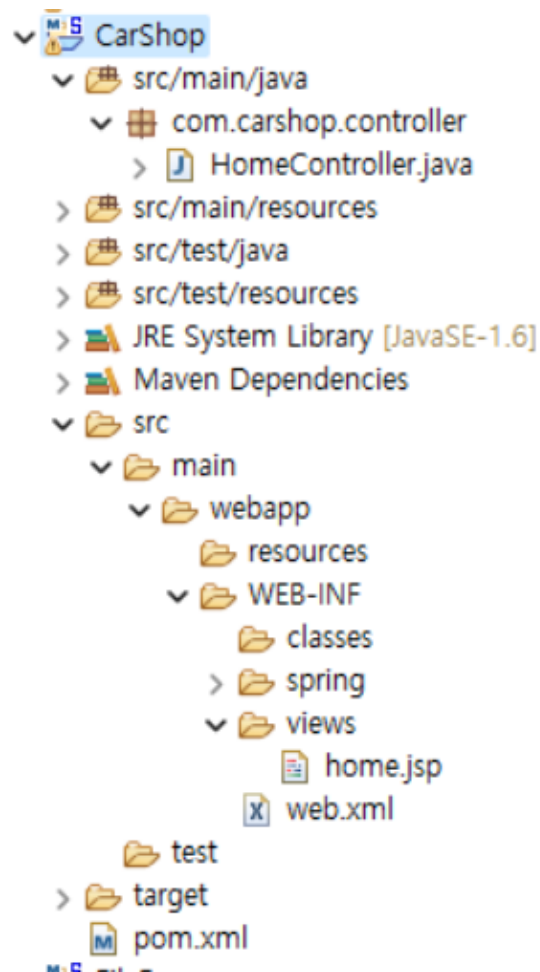
**Project Settings - Spring MVC Project**

Define project specific settings. Required settings are denoted by "\*".

Please specify the top-level package e.g. com.mycompany.myapp\*

com.carshop.controller

? < Back Next > Finish Cancel



계층적 구조

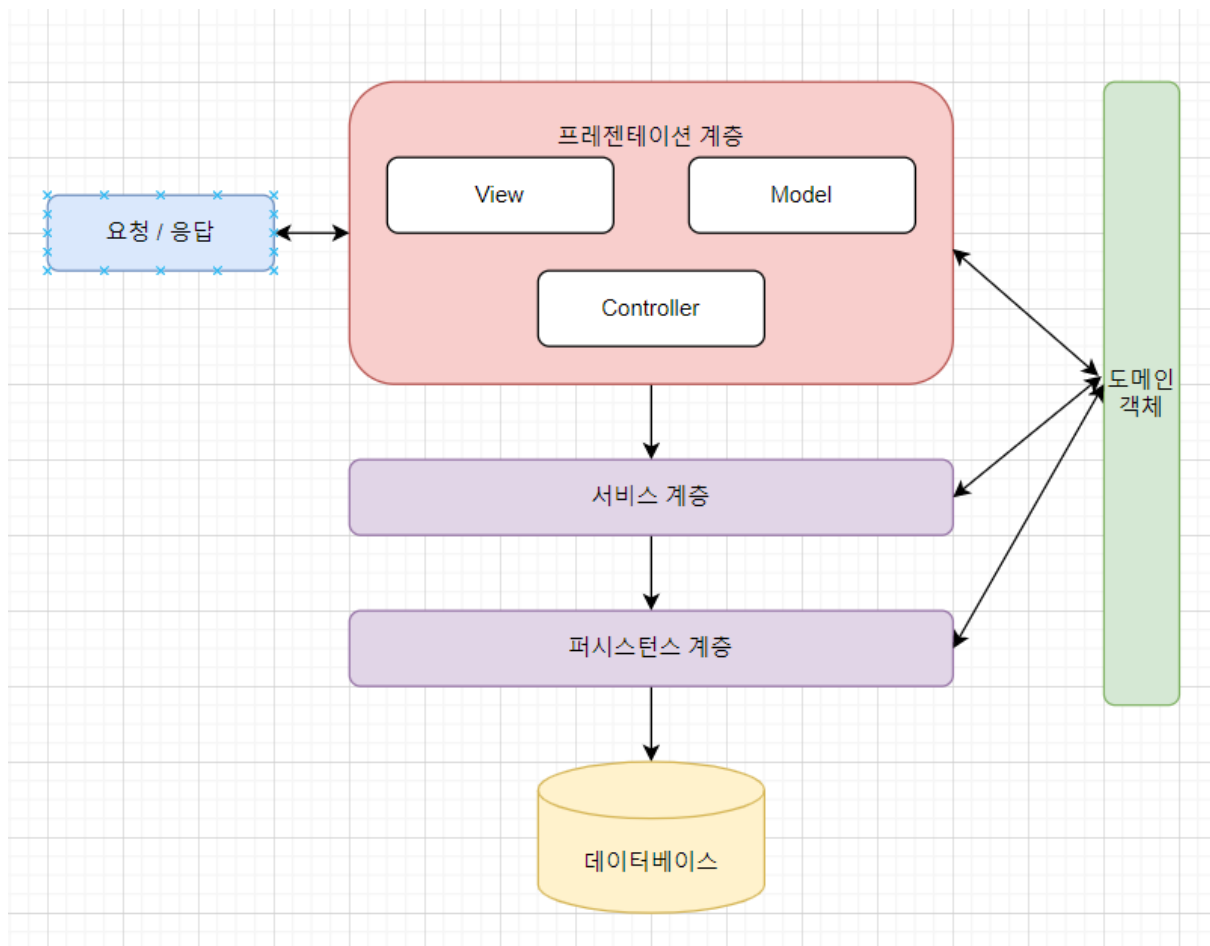
계층적 구조 없이 한곳에서 모든 작업을 한번에 처리하면

- 코드의 복잡성 증가

- 유지 보수의 어려움
- 유연성 부족
- 중복 코드 증가
- 낮은 확장성

등의 문제가 발생할수 있다.

계층적 구조는 퍼시스턴스persistence 계층 서비스 service계층 프레젠테이션 presentation계층 으로 분리하여 사이트를 구성한다. mvc는 이중에서 프레젠테이션 계층에 속하게 된다.

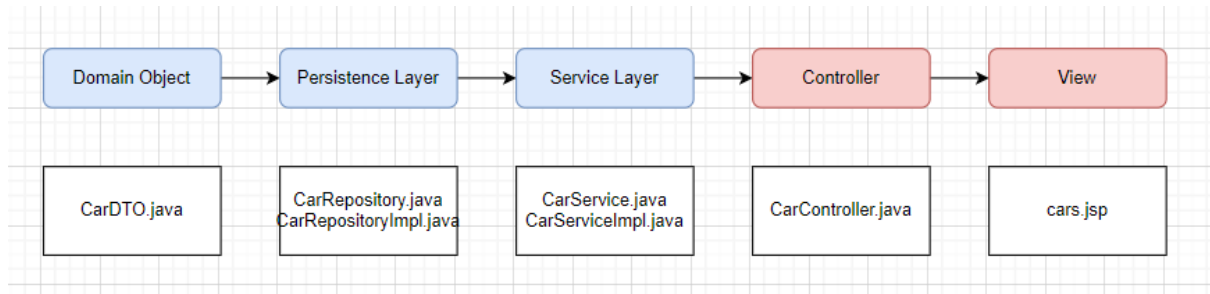


도메인 객체 : domain object 데이터 모델, 객체 정보를 저장한다. jsp 에서의 DTO와 같다.

퍼시스턴스 계층 : persistence layer 데이터 액세스 계층, 데이터베이스에 접근하여 데이터를 처리한다. jsp DAO 와 같다.

서비스 계층 : service layer 비즈니스 계층이라고도 하고 애플리케이션에서 제공하는 포괄적인 서비스를 표현한다. 프레젠테이션 계층과 퍼시스턴스 계층을 연결하는 역할을 한다.

프레젠테이션 계층 presentation layer 사용자에게 데이터를 입력 받거나 결과를 서버에 전달하고 사용자에게 보여주는 계층이다.



## DTO (도메인 객체)

### CarDTO.java

```
package com.carshop.controller;

public class CarDTO {

    private String cid, cname, cprice, ccate, cdesc;

    public String getCid() {
        return cid;
    }

    public void setCid(String cid) {
        this.cid = cid;
    }

    public String getCname() {
        return cname;
    }

    public void setCname(String cname) {
        this.cname = cname;
    }

    public String getCprice() {
        return cprice;
    }

    public void setCprice(String cprice) {
        this.cprice = cprice;
    }
}
```

```

public String getCcate() {
    return ccate;
}

public void setCcate(String ccate) {
    this.ccate = ccate;
}

public String getCdesc() {
    return cdesc;
}

public void setCdesc(String cdesc) {
    this.cdesc = cdesc;
}

public CarDTO(String cid, String cname, String cprice, String ccate, String cdesc) {

    this.cid = cid;
    this.cname = cname;
    this.cprice = cprice;
    this.ccate = ccate;
    this.cdesc = cdesc;
}

public CarDTO() {

}

}

```

## interface 복습...

- 1 인터페이스는 일종의 명령서이다.
- 2 인터페이스는 다형성을 표현/제어한다.

## CarRepository.java 인터페이스

```

package com.carshop.controller;

import java.util.List;

public interface CarRepository {

    List<CarDTO> getAllCarList();
}

```

```
}
```

## CarRepositoryImpl.java 구현 클래스

```
package com.carshop.controller;

import java.util.ArrayList;
import java.util.List;

import org.springframework.stereotype.Repository;

@Repository
public class CarRepositoryImpl implements CarRepository {

    private List<CarDTO> listOfCars = new ArrayList<CarDTO>();

    public CarRepositoryImpl() {

        CarDTO car1 = new CarDTO("c0001", "소나타", "2500", "승용차", "거의 새거");
        CarDTO car2 = new CarDTO("c0002", "그랜저", "3500", "승용차", "아주 새거");
        CarDTO car3 = new CarDTO("c0003", "아반테", "2000", "승용차", "극히 새거");

        listOfCars.add(car1);
        listOfCars.add(car2);
        listOfCars.add(car3);

    }

    @Override
    public List<CarDTO> getAllCarList() {

        return listOfCars;

    }

}
```

## CarService.java 인터페이스

```
package com.carshop.controller;

import java.util.List;

public interface CarService {

    List<CarDTO> getAllCarList();

}
```



## CarServiceImpl.java 구현 클래스

```
package com.carshop.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class CarServiceImpl implements CarService {

    @Autowired
    private CarRepository carRepository;

    @Override
    public List<CarDTO> getAllCarList() {
        // TODO Auto-generated method stub
        return carRepository.getAllCarList();
    }

}
```

## CarController.java

```
package com.carshop.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;

@Controller
public class CarController {

    @Autowired
    private CarService carService;

    @RequestMapping("/cars")
    public String CarList(Model model) {
        List<CarDTO> list = carService.getAllCarList();
        model.addAttribute("carList", list);
        return "cars";
    }

}
```

# Cars.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Car List</title>
</head>
<body>
    <link
        href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css"
        rel="stylesheet"
        integrity="sha384-GLhLTQ8iRABdZLL603oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD"
        crossorigin="anonymous">

    <nav class="navbar navbar-expand navbar-dark bg-dark"
        aria-label="Second navbar example">
        <div class="container-fluid">
            <a class="navbar-brand" href="#">CarShop</a>
            <button class="navbar-toggler" type="button"
                data-bs-toggle="collapse" data-bs-target="#navbarsExample02"
                aria-controls="navbarsExample02" aria-expanded="false"
                aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>

            <div class="collapse navbar-collapse" id="navbarsExample02">
                <ul class="navbar-nav me-auto">
                    <li class="nav-item"><a class="nav-link active"
                        aria-current="page" href="/">홈</a></li>
                    <li class="nav-item"><a class="nav-link" href="/cars">차량
                        보기</a></li>
                    <li class="nav-item"><a class="nav-link" href="/boards">게시판</a>
                    </li>
                </ul>
                <form role="search">
                    <input class="form-control" type="search" placeholder="Search"
                        aria-label="Search">
                </form>
            </div>
        </div>
    </nav>

    <div class="alert alert-dark">
        <div class="container">
            <h1>차량 보기</h1>
        </div>
    </div>

    <div class="container">
        <div class="row" align="center">
```

```
<c:forEach items="${carList}" var="car">
  <div class = "col-md-4">
    <h3>${car.cid}</h3>
    <p>${car.cname}</p>
    <p>${car.cprice} 만원</p>
  </div>

</c:forEach>

</div>
</div>

<script
  src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.bundle.min.js"
  integrity="sha384-w76AqPfdkMBDXo30jS1Sgez6pr3x5MlQ1ZAGC+nuZB+EYdgRZgiwxhTBTkF7CXvN"
  crossorigin="anonymous"></script>
</body>
</html>
```