



SPRING Framework Overview

강 사 : 임 정 훈



목 차

1

Spring Framework 특징

2

Spring Framework 구성요소

3

IoC/DI/AOP 개념

4

Spring MVC

5

Spring 수업계획



Framework 란?

- 정보시스템을 개발 또는 운영할 때 필요한 기본 기능을 미리 구현해 놓은 것.
- 추가 기능을 개발하여 조립함으로써 전체 정보시스템을 완성할 수 있음.



<http://www.egovframe.go.kr> 참고



SPRING 허와 실

실(truth)

- 객체지향 원칙 적용
- 단순하고 깔끔한 코드 작성
- 생산성과 품질 향상
- **유연성과 확장성 증대**

허(falsity)

- 코드가 짧아진다.
- 시간/노력이 감소된다.
- 웹 애플리케이션만 적용된다.



SPRING 등장배경

- 등장배경

- 자바 기반 엔터프라이즈 애플리케이션 개발의 복잡함을 간소화하기 위해서 개발된 프레임워크
- POJO 개념 도입 ➔ 컴포넌트간 낮은 결합도 ➔ 테스트, 재사용성, 유지보수성 향상



SPRING 등장배경

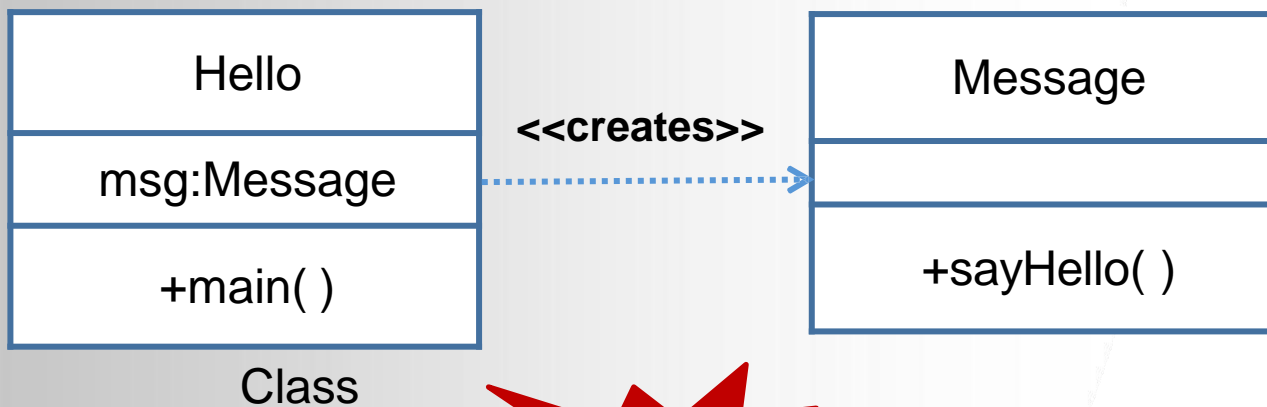
- **POJO : 평범한 자바객체**

- 구현(implements) 또는 확장(extends)하지 않은 순수한 자바객체 의미
 - ✓ 특정 interface나 class로 부터 상속받지 않음
 - ✓ 어떠한 자바 API도 импорт하지 않음
- 다른 객체 접근을 위해서 스프링이 객체 주입(DI)



SPRING 등장배경

메인 클래스와 구현 클래스 간의 **강한 결합 관계**

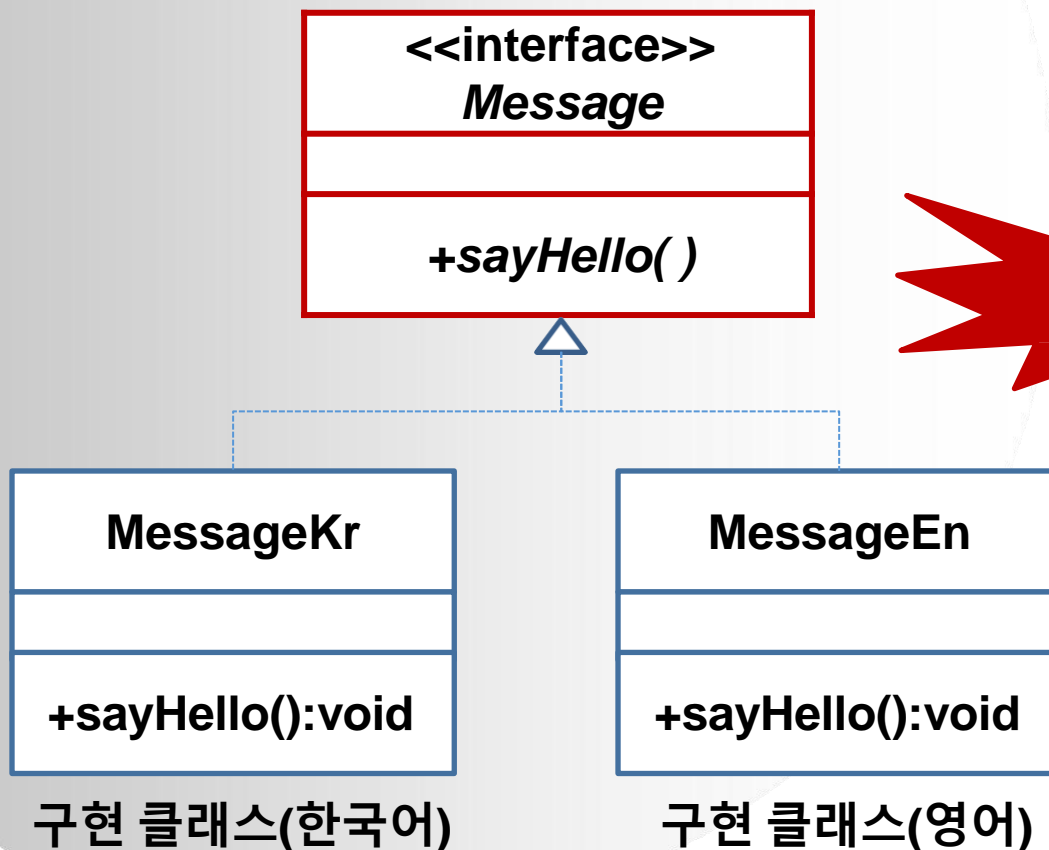


유지보수
문제점



클래스 간의 강 결합 예제

인터페이스의 구현 클래스 상속관계



메서드 강제
구현 요구



SPRING 등장배경

- 의존성 주입(Dependency Injection)

- POJO에서 외부 객체 접근을 위해 필요한 객체를 스프링에서 주입하는 의미
- 제3자가 객체 주입 ➔ '제어의 역전(IoC)' 라고 표현



SPRING 전략 및 이점

● SPRING 주요 전략

- 자바의 복잡함을 간소화하기 위한 전략.
 1. POJO를 이용한 가볍고, 비 침투적인 개발.
(interface, class, API 침투 차단)
 2. DI와 인터페이스 기반 느슨한 결합도(클래스 약결합).
 3. 애스펙트 지향 프로그래밍을 통한 공통 관심사 재사용.



SPRING 전략 및 이점

- **SPRING 이점**

1. 코드의 간소화로 이해하기 쉽다.
2. 낮은 결합도 ➔ 테스트 용이, 코드 수정 범위 축소
3. 클래스 재사용성 및 유지보수성 향상



Spring Framework 특징(1)

① 스프링은 경량 컨테이너

- 스프링은 자바 기반 프레임워크(JavaEE에 비해서 경량)
- 자바 객체 생성, 소멸 등의 라이프 사이클 관리

② 스프링은 DI(Dependency Injection) 지원

- 설정 파일(xml)이나 annotation을 통해서 객체 간 의존관계 설정
- 확장성과 유지보수성 (▲)



Spring Framework 특징(2)

③ 스프링은 AOP(Asspect Oriented Programming) 지원

- 모듈에서 공통으로 필요한 부분(트랜잭션, 로깅, 보안)을 관심(Aspect)으로 분리해 놓고 필요한 경우 적용

④ 스프링은 POJO(Plain Old Java Object) 지원

- 스프링 컨테이너에 저장되는 자바 객체는 특정한 인터페이스를 구현하거나 클래스를 상속받지 않음
- 평범한 **JAVA 객체**



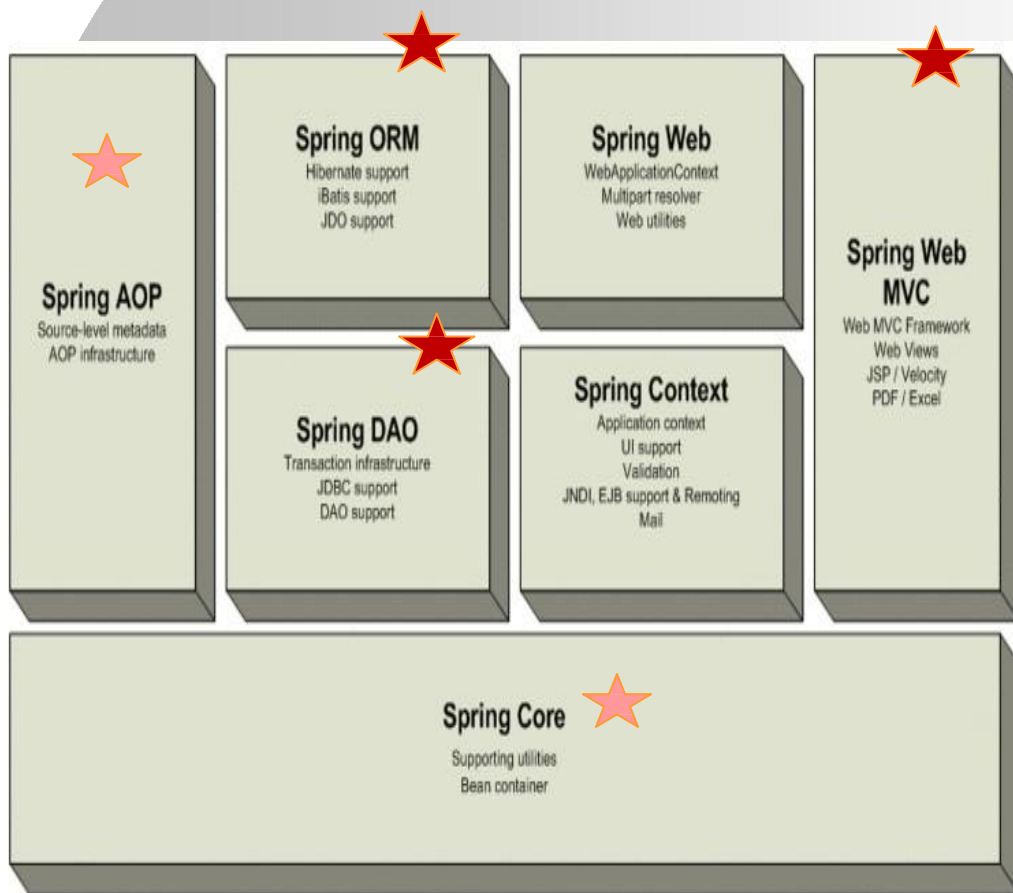
Spring Framework 특징(3)

⑤ 스프링은 DB처리와 관련된 다양한 API 지원

- Spring-JDBC, iBatis, myBatis, Hibernate, JPA 등 ORM(Object Relational Mapping) 처리를 위해 널리 사용되는 라이브러리의 연동 지원
- 엔터프라이즈 어플리케이션(JMS, 메일, 스케줄링 등) 개발에 필요한 다양한 API를 설정 파일과 annotation 을 통해서 손쉽게 사용 (Struts2, JSF와 같은 프레임워크와의 연동 지원)



SPRING Framework 구성요소

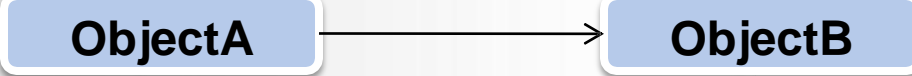



1. **Spring Core**
IoC/DI Spring
2. **Spring AOP**
3. **Spring ORM**
(iBatis, myBatis, Hibernate)
4. **Spring DAO**
DAO , JDBC DB
5. **Spring WEB**
6. **Spring Context**
UI, ,
7. **Spring MVC**
MVC

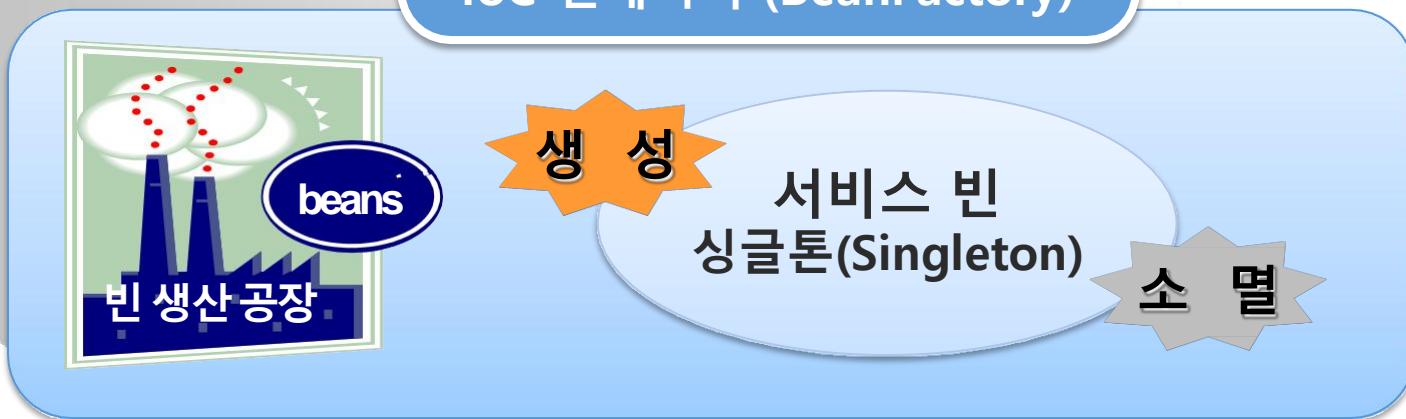


IoC 소개

■ Spring IoC (Inversion of Control : 제어의 역전)

- 기존 제어 방식 : 
- Spring IoC 방식 : 
- Bean : Spring이 만들고 관계를 설정하는 객체(POJO)
- 개발에 있어서 확장성과 재 사용성 극대화 추구

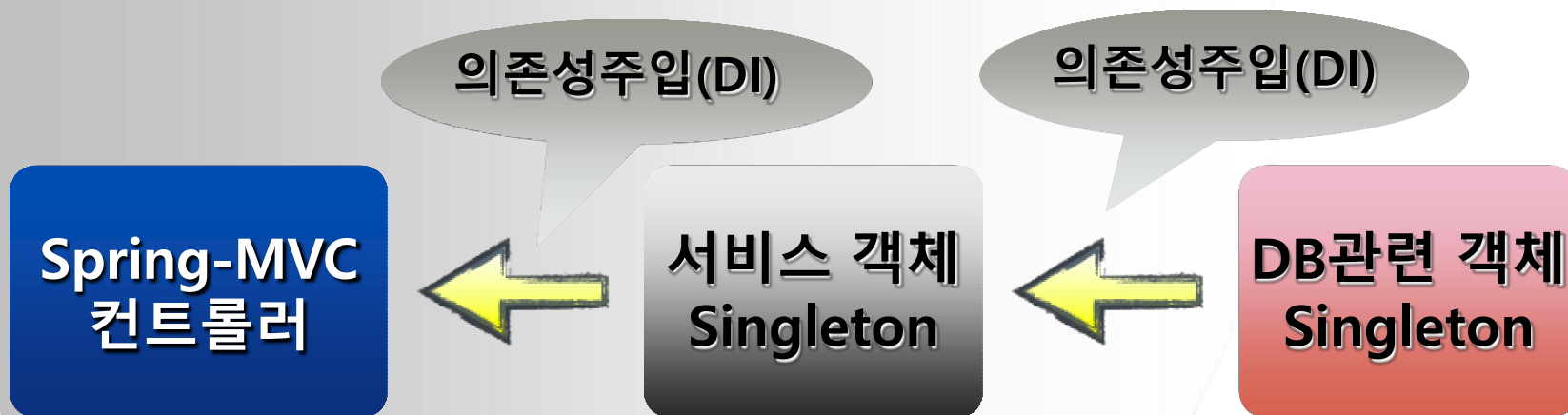
IoC 컨테이너 (BeanFactory)





DI(Dependency Injection)

- 의존성 주입(Dependency Injection)
- IoC 컨테이너에 의해서 생성된 Bean 객체 주입

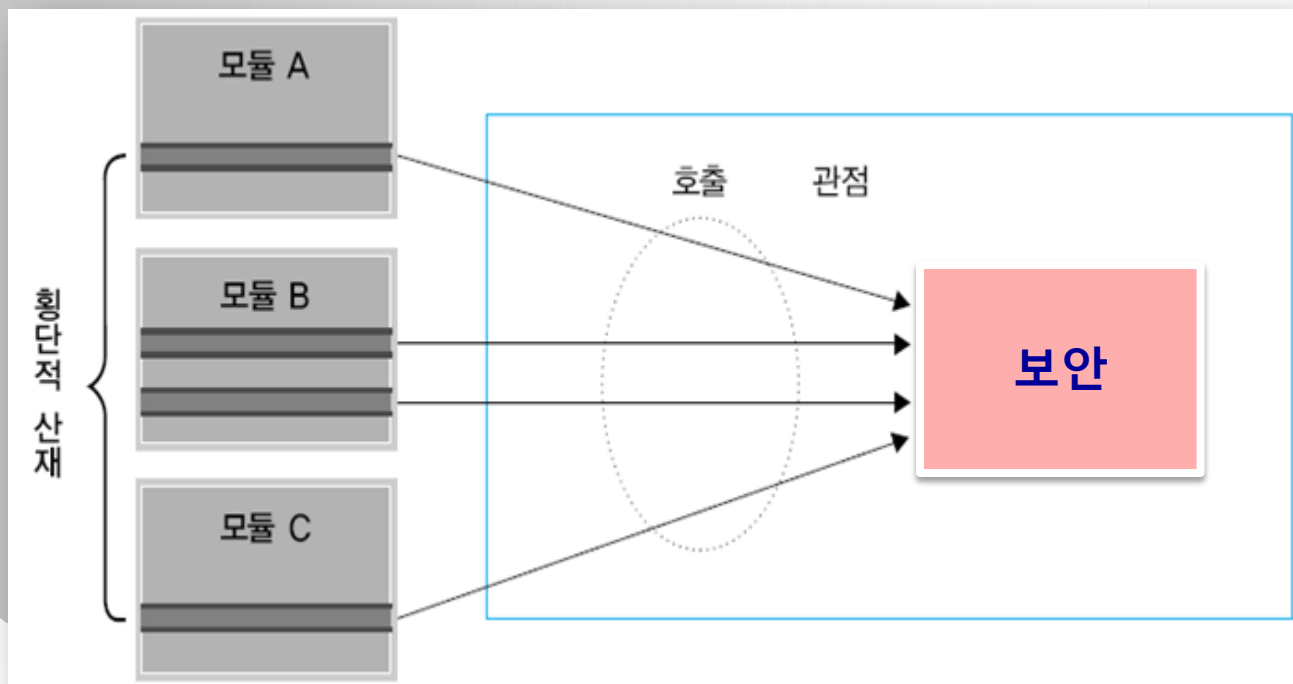




AOP(Aspect Oriented Programming)

- 관점 지향 프로그래밍

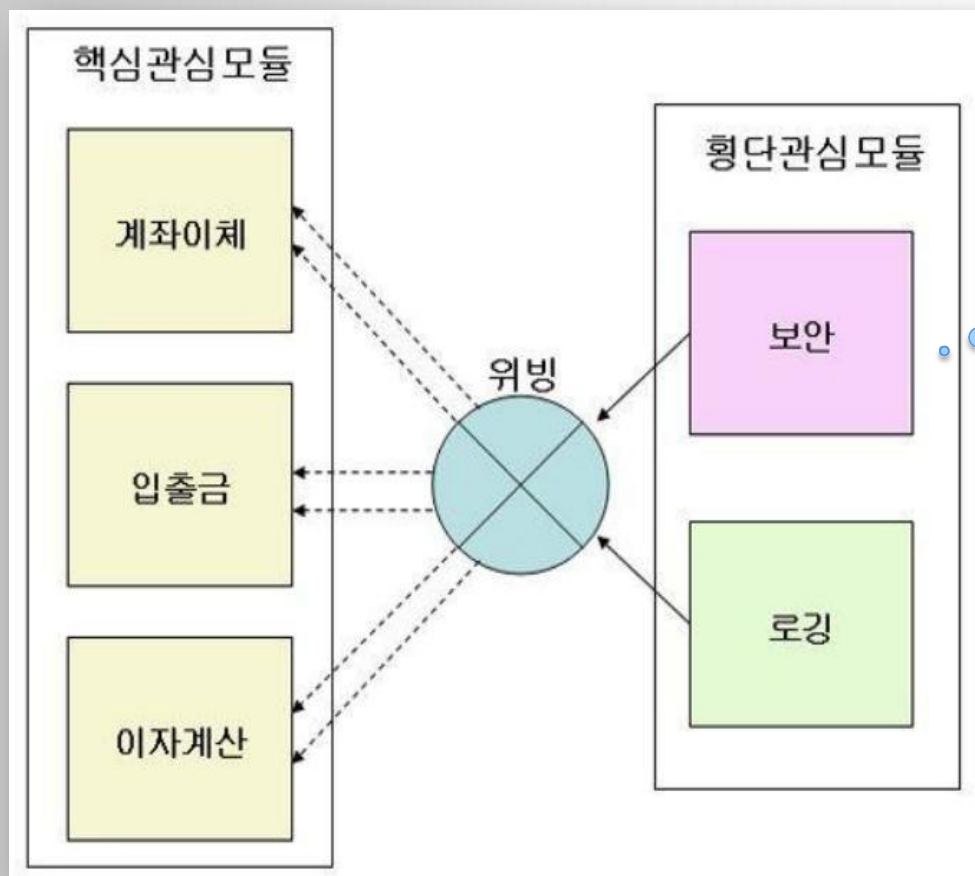
- 액스펙트 : 기존 객체지향 프로그래밍에서 모듈화 기법으로 추가된 중복 코드(공통 처리 부분)를 별도의 독립된 클래스(Asspect)로 만들어 놓은 것
- 모듈이 실행될 때 애스펙트(ASPECT)를 위임해서 실행되게 하는 프로그래밍 방식





AOP(Aspect Oriented Programming)

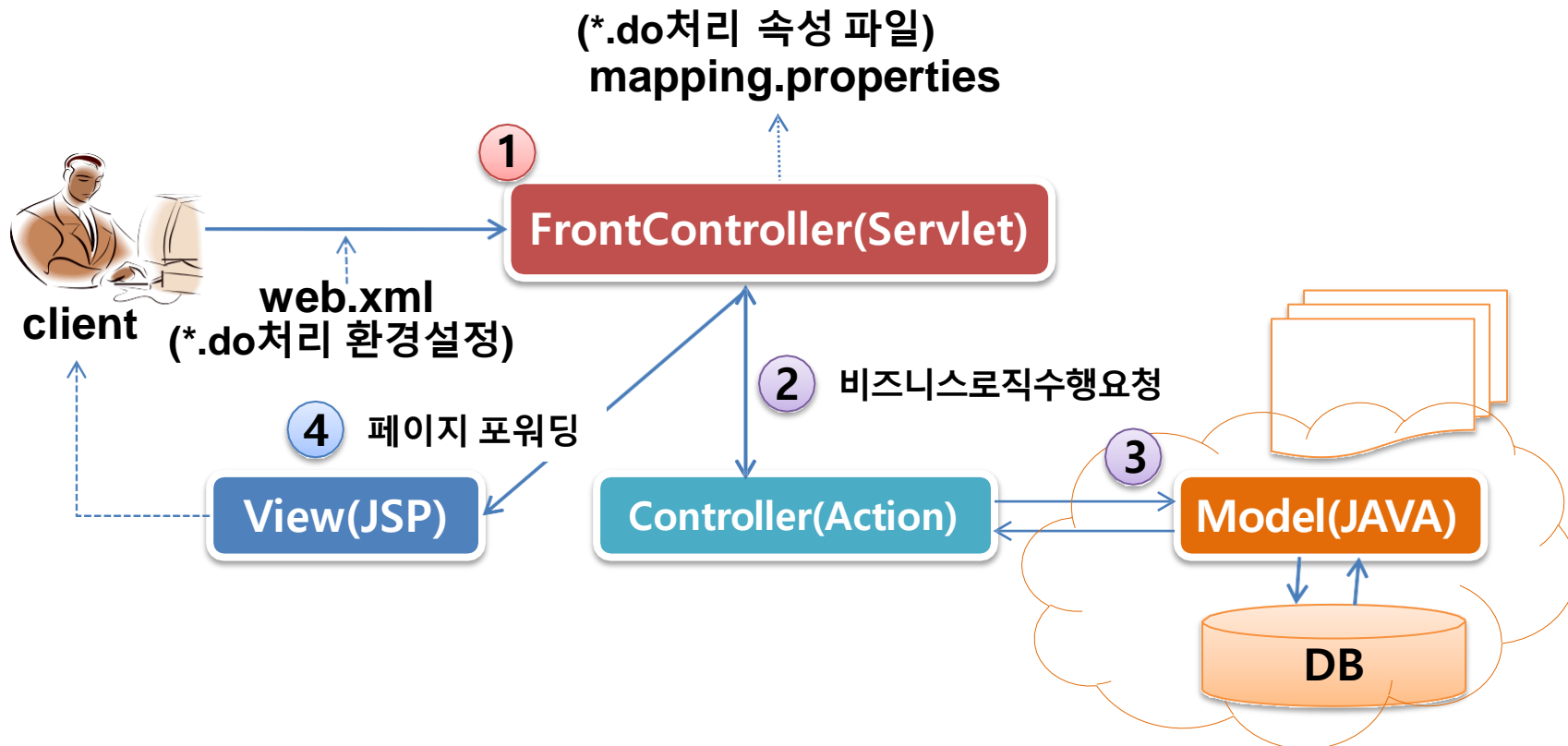
- 핵심관심모듈 vs 횡단관심모듈(엑스펙트)



엑스펙트
(Aspect)

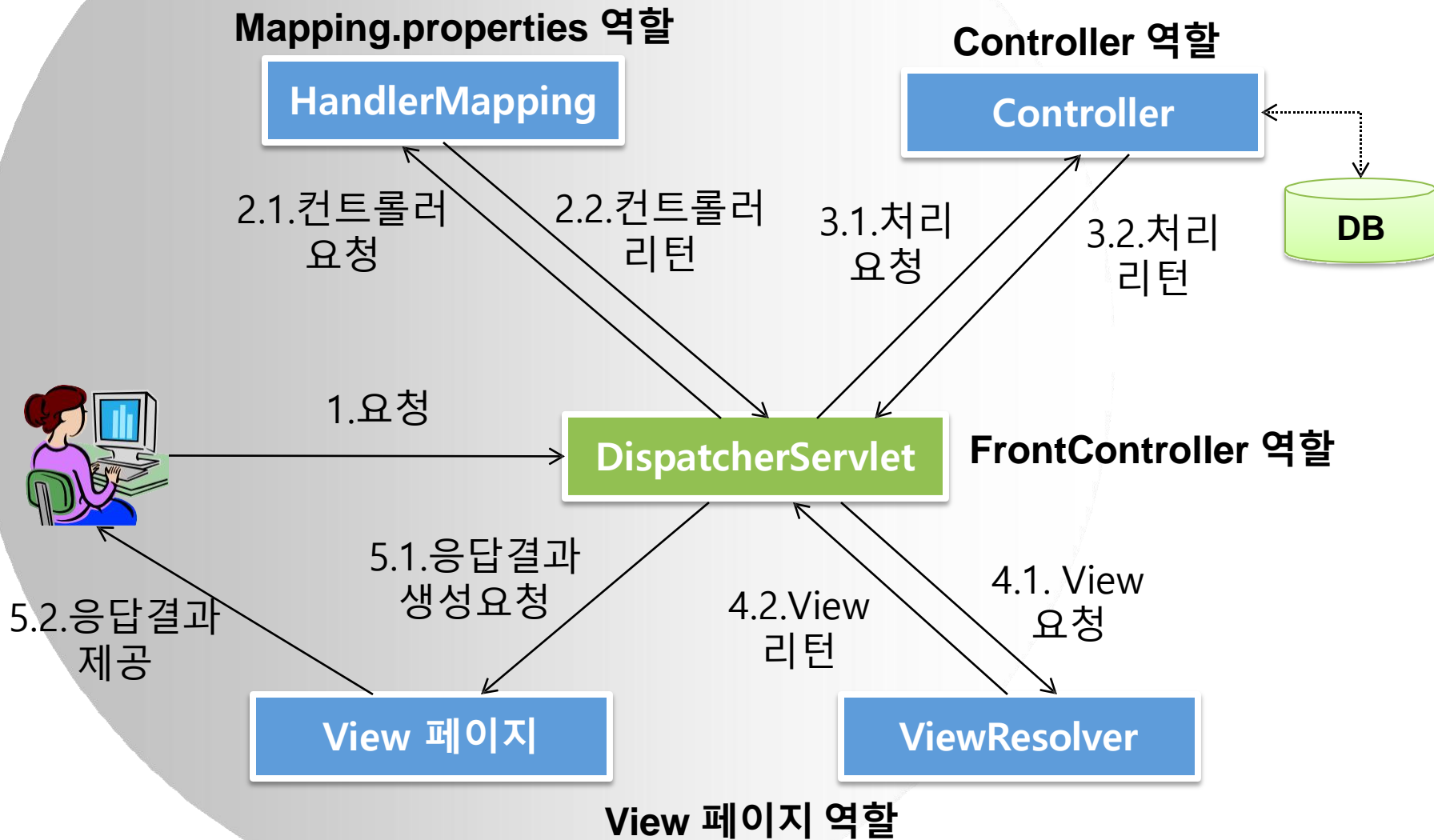


JSP 에서의 MVC 디자인 패턴(모델2방식)



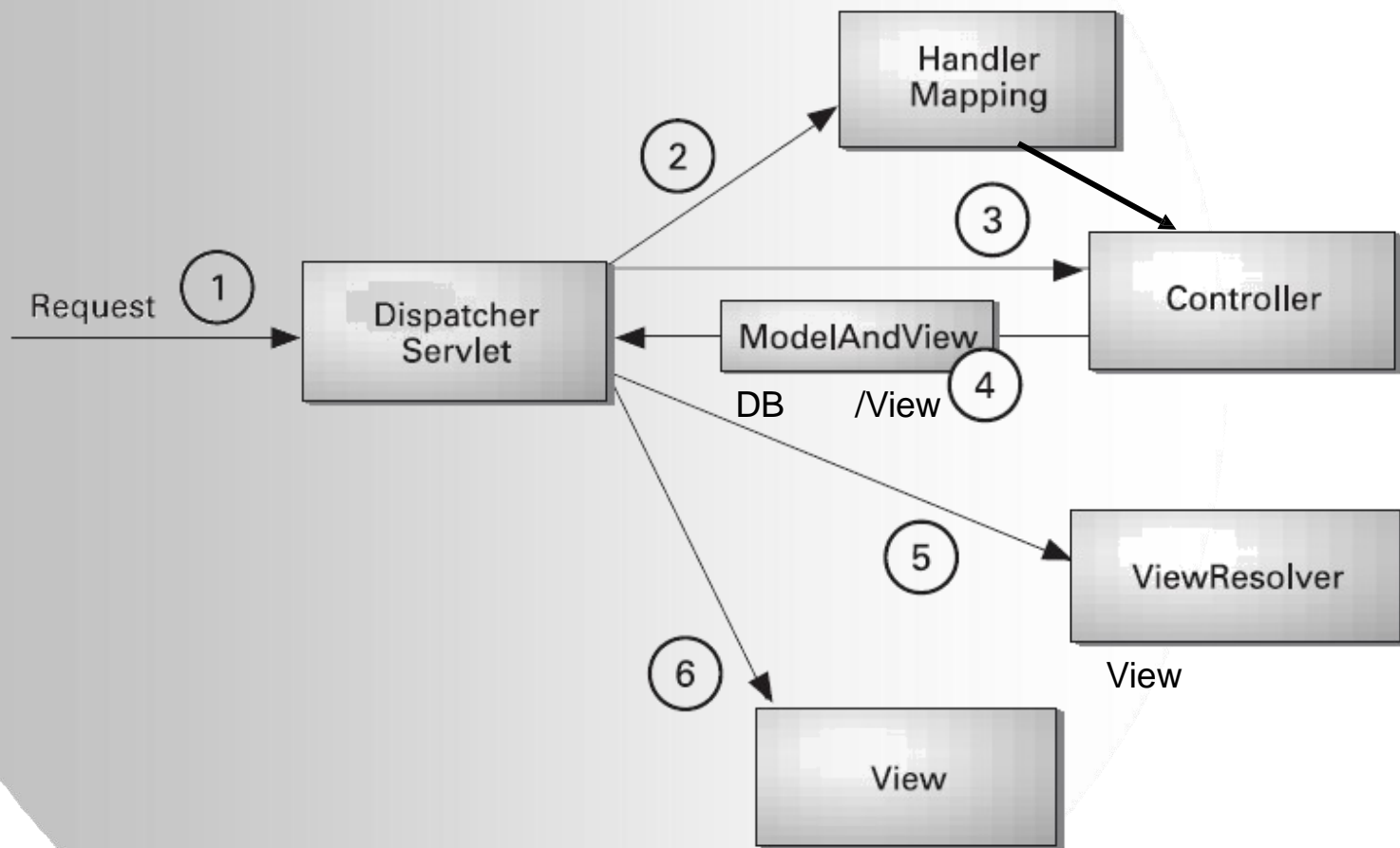


Spring MVC Flow(1)





Spring MVC Flow(2)



스프링 MVC 프레임 워크 Flow



수업계획

SPRING Framework

순서	주 제	수업내용 & 목표	비고
0	Spring Overview	• 스프링 개요 및 환경설정, STS	PPT
1	IoC(Inversion of Control)	• Interface, 설정파일, IoC 개요, POJO	Java APP 콘솔
2	DI(Dependency Injection)	• Has a 관계, DI 개요, 디자인패턴 • ApplicationContext, Autowiring	
3	Annotation	• Spring 주요 어노테이션 종류	
4	AOP(Aspect Oriented Programming)	• AOP개념 및 실습	
5	Spring-JDBC	• spring-JDBC 개요 및 실습	Web APP 브라우저
6	Spring-MVC	• spring-web-mvc 설정 및 실습	
7	SpringMVC_JDBC	• spring-web-mvc와 jdbc 설정 및 실습	
8	iBatis_Exam	• iBatis 개요 및 실습	
9	myBaties_Exam	• myBaties 개요 및 실습	



실습 환경

Library

- org.springframework.xxx- 3.2.3.RELEASE.jar
- ibatis-2.3.4.726.jar
- mybatis-3.0.5.jar
- jstl-api-1.2.jar
- ojdbc14.jar-Oracle 10g

Tools

- Java SE 7.0 (jdk1.7)
- eclipse-jee-juno-SR1-win32
- apache-tomcat-7.0.33
- Apple Safari/Firefox



SPRING 용어

용어	의 미
1. IoC/POJO/Bean	
2. DI(Dependency Injection)	
3. ApplicationContext	
4. Autowiring	
5. Spring 주요 Annotation	
6. AOP	
7. Spring JDBC	
8. iBatis/myBatis	
9. Spring DAO	
10. Spring MVC	