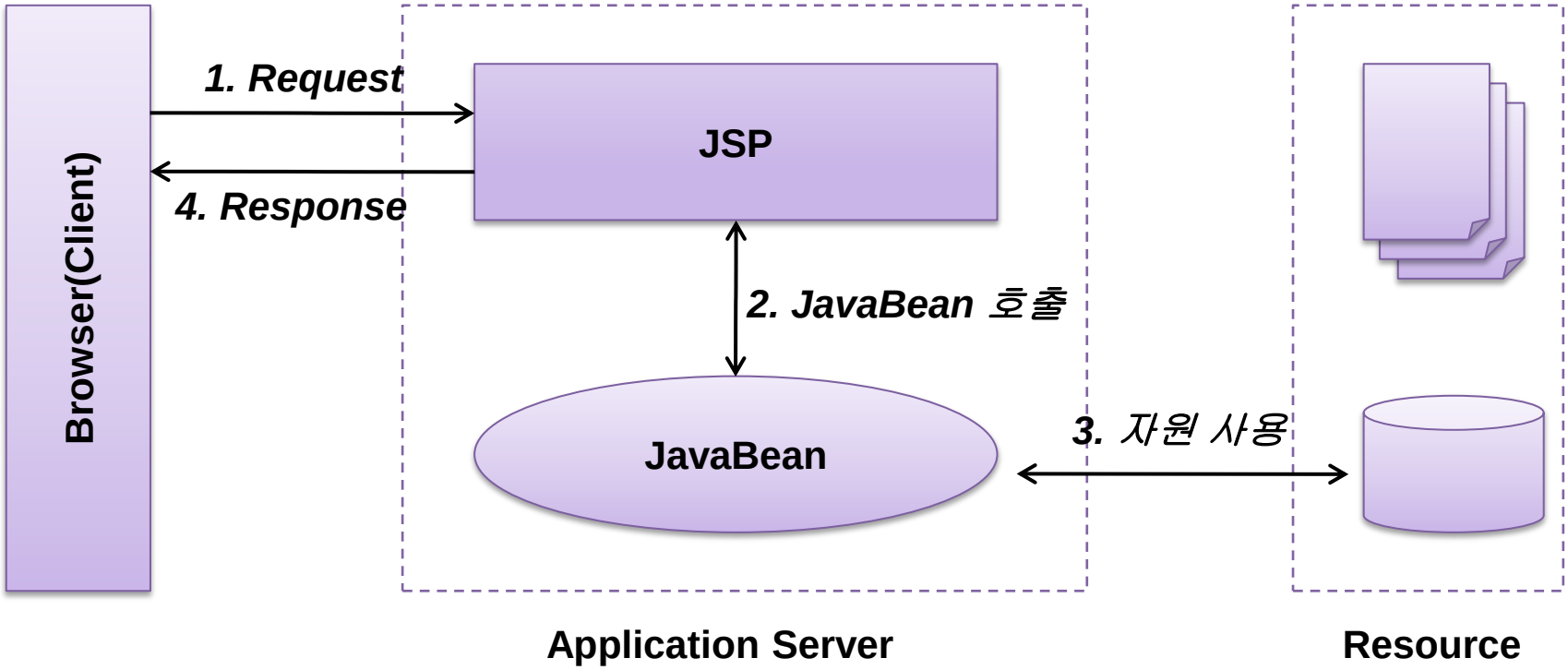
Spring Framework
- 01. 스프링 프레임워크

CONTENTS

- Web Application 설계방식
- 프레임워크란
- 스프링 프레임워크란
- 스프링프레임워크 설치와 모듈 구성
- DepenDency Injection과 스프링프레임워크
- AOP와 스프링

Web Application 설계방식

- 모델1 개요
  - JSP 만 이용하여 개발하는 경우
  - JSP + Java Bean을 이용하여 개발하는 경우



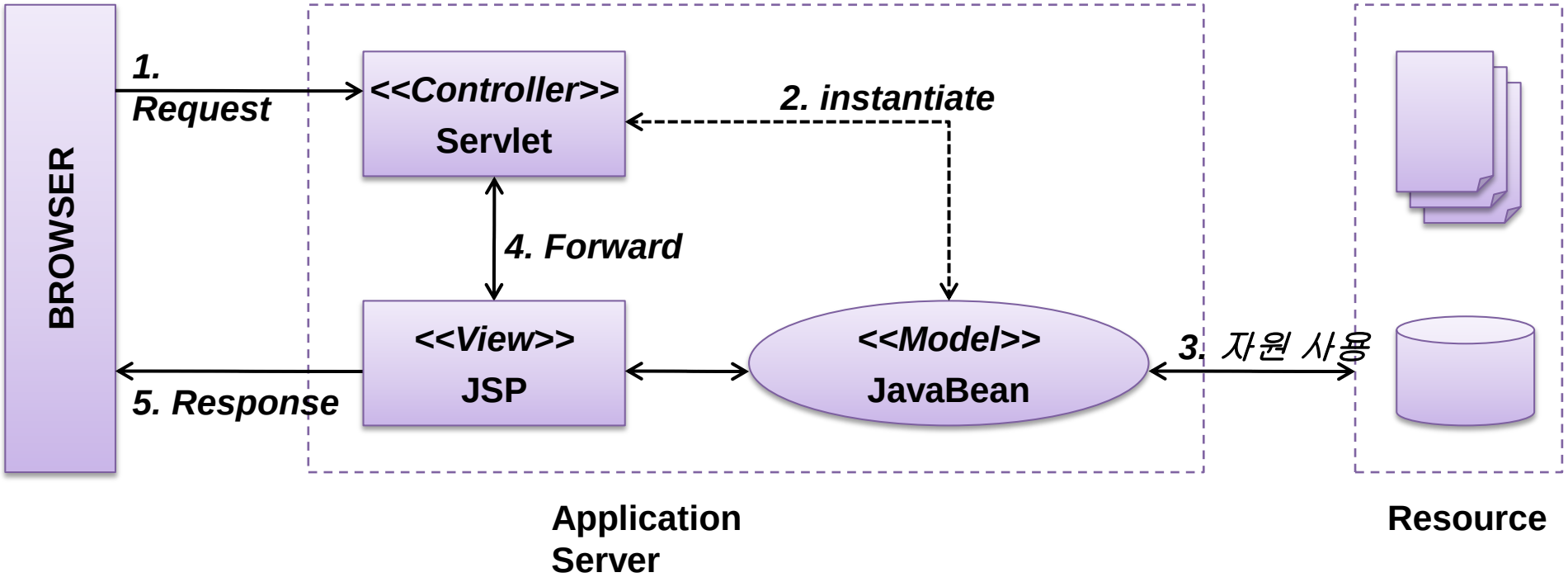
Web Application 설계방식

- 모델1의 장단점
  - 장점
    - 개발속도가 빠름
    - 개발자의 기술적인 숙련도가 낮아도 배우기 쉬워 빠르게 적용 가능
  - 단점
    - JSP 페이지에서 프레젠테이션 로직과 비즈니스 로직이 혼재되어 복잡
    - 로직의 혼재로 인해 개발자와 디자이너의 작업 분리가 어려움
    - JSP 코드의 복잡도로 인해 유지보수가 어려워짐
- 웹 애플리케이션이 복잡해지고 사용자 요구가 증가함에 따라 새로운 개발 방식을 요구

Web Application 설계방식

- 모델2 개요
  - GUI 개발모델인 MVC를 웹 애플리케이션에 적용한 방식
  - Application의 역할을 Model – View – Controller로 분리
  - Model: Business Logic 담당
    - Java Bean으로 구현
    - Business Service(Manager)
    - Business Logic의 workflow 관리
    - DAO (Data Access Object) : DB 연동해 Business Logic 처리
  - View: Client에게 응답을 처리한다.
    - JSP로 구현
  - Controller: 클라이언트의 요청을 받아 Model과 View사이에서 이벤트 흐름 제어
    - Servlet으로 구현
    - Client의 요청을 받아 Client가 보낸 Data를 읽고 검사
    - Model에게 Business Logic을 요청
    - Model의 처리 결과에 맞는 View에게 응답 요청

Web Application 설계방식



Web Application 설계방식

- 모델2의 장단점
  - 장점
    - 비즈니스 로직과 프리젠테이션의 분리로 인해 어플리케이션이 명료해지며 유지보수와 확장이 용이함
    - 디자이너와 개발자의 작업을 분리해 줌
  - 단점
    - 개발 초기에 아키텍처 디자인을 위한 시간의 소요로 개발 기간이 늘어남
    - MVC 구조에 대한 개발자들의 이해가 필요함

프레임워크

- **프레임워크**
  - 뼈대 혹은 틀
  - 소프트웨어 관점 : 아키텍처에 해당하는 골격 코드

- **'아키텍쳐' , '골격'**
  - 애플리케이션을 개발할 때 가장 중요한 것이 애플리케이션의 구조를 결정하는 것이 아키텍처인데 이 아키텍처에 해당하는 골격 코드를 프레임워크가 제공한다.

    예) 컵을 만든다.

    A와 B는 자유롭게 컵을 만든다.

    A의 퇴사, A가 만든 컵을 고쳐야 한다....

프레임워크

- **기존 애플리케이션 개발 과정의 문제점**
  - 시스템 개발과정에서 대부분의 개발자들은 산출물에 입각해서 개발을 하므로 아키텍처의 일관성이 잘 유지 되지만 유지보수 과정에서는 개발자의 경험에 의존하는 경우가 많다.

- **프레임워크는 이러한 문제를 근본적으로 해결**

  - 애플리케이션 개발에 기본이 되는 뼈대나 틀을 제공

  : 개발자에게 모든 것을 위임하는 것이 아니라 애플리케이션의 기본 아키텍처는 프레임워크가 제공하고, 그 뼈대에 살을 붙이는 작업만 개발자가 하는 것이다.

프레임워크

- **프레임워크의 장점**

1. 빠른 구현 시간

개발자는 비즈니스 영역만 구현하면 됨. 제한된 시간에 많은 기능을 구현할 수 있다.

2. 쉬운 관리

유지보수에 들어가는 인력과 기간을 줄일 수 있다.

3. 개발자의 역량 획일화

숙련된 개발자와 초급개발자의 코드가 비슷해진다.

관리자 입장에서 개발 인력을 더 효율적으로 구성할 수 있다.

4. 검증된 아키텍처의 재사용과 일관성 유지

아키텍처에 관한 고민이나 검증 없이 애플리케이션을 개발한다.

유지보수 과정에서 아키텍처가 왜곡되거나 변형되지 않는다.

스프링 프레임워크란

- **로드존슨이 2004년에 만든 오픈 소스 프레임워크.**
- **스프링 프레임워크가 등장하기 전까지는 자바 기반의 엔터프라이즈 애플리케이션은 대부분 EJB(Enterprise Java Beans)로 개발되었다.**
- **EJB의 문제점**

  - 복잡하고, 고가의 비용발생, 많은 시간과 노력이 필요.
    스팩의 복잡함, 학습에 많은 시간이 필요, 유지보수 역시 복잡함,
    설치를 위해 WAS(Web Application Sever)가 필요
    : JEUS, Weblogic, WebSpere 등, 수천만원의 고가 장비.
  - 다양한 디자인 패턴을 이해하고 적용해야 함.

- **스프링 프레임워크는 이미 많은 디자인 패턴이 적용되어 배포되는 프레임워크이기 때문에 많은 디자인 패턴을 사용하는 것과 같다.**

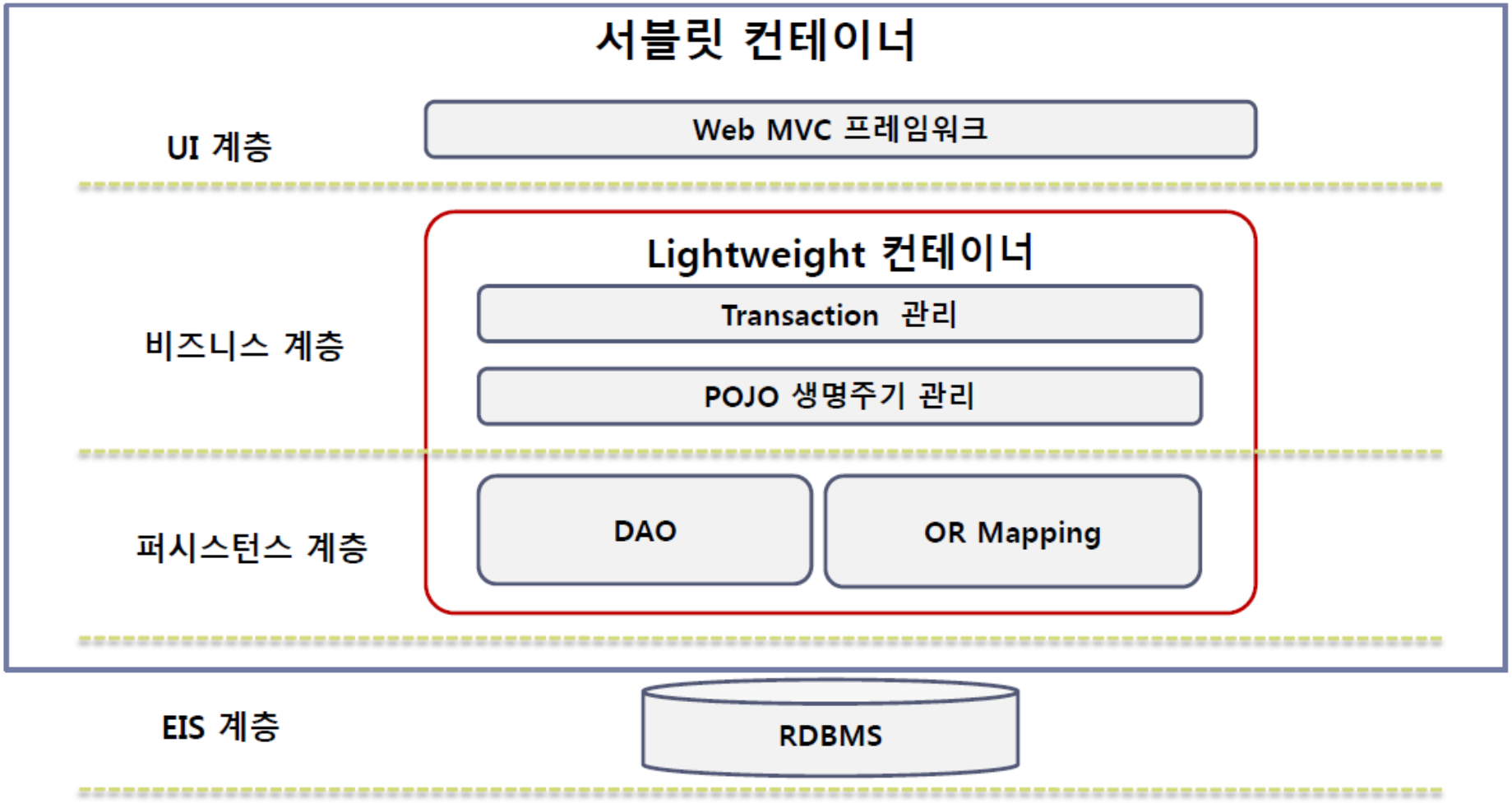
스프링 프레임워크의 주요 특징

- **스프링은 경량의 프레임워크이다.**
  - 자바 객체를 담고 있는 컨테이너다
  - 자바객체의 생성, 소멸과 같은 라이프 사이클을 관리한다.
- **스프링은 DI(dependency Injection)패턴을 지원한다.**
  - 설정파일을 통해서 의존관계를 설정할 수 있다.
- **스프링은 AOP(Aspect Oriented Programming)를 지원한다.**
  - 트랙잭션이나 로깅, 보안과 같은 공통으로 필요로 하는 기능을 분리해서 각각의 모듈에 적용할 수 있다.
- **스프링은 POJO(Plain Old Java Object)를 지원한다.**
  - 특정 인터페이스나 클래스를 상속받지 않은 자바 객체를 스프링 컨테이너가 저장하고 있다.

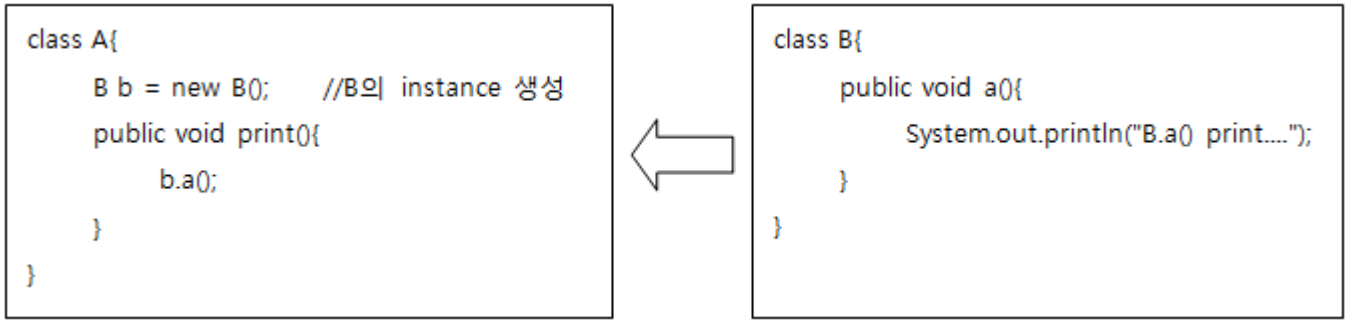
스프링 프레임워크의 주요 특징

- **트랜잭션 처리를 위한 일관된 방법을 제공한다.**
- **영속성과 관련된 다양한 API를 제공한다.**
  - **JDBC, iBatis(myBatis),** Hibernate, JPA등과 같은 프레임워크와의 연동을 지원한다.
- **자체적으로 MVC 프레임워크를 제공**
  - 스프링만 가지고 MVC 기반의 웹 어플리케이션을 개발

스프링 프레임워크의 주요 특징

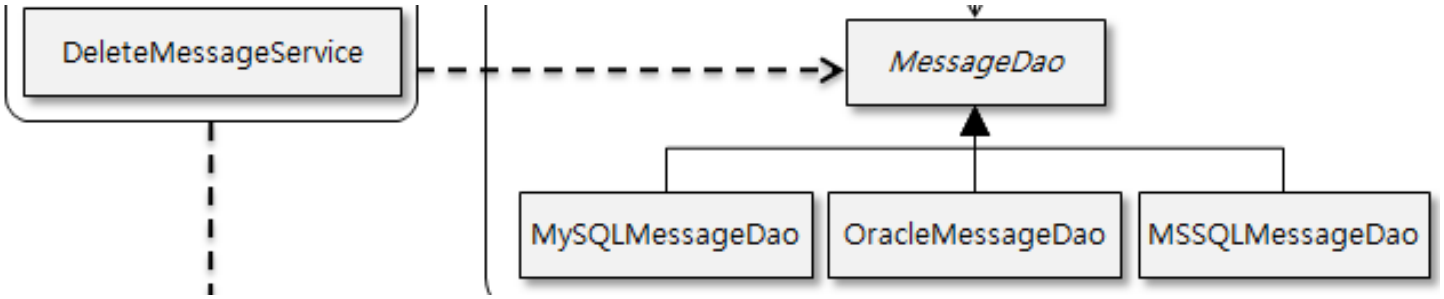


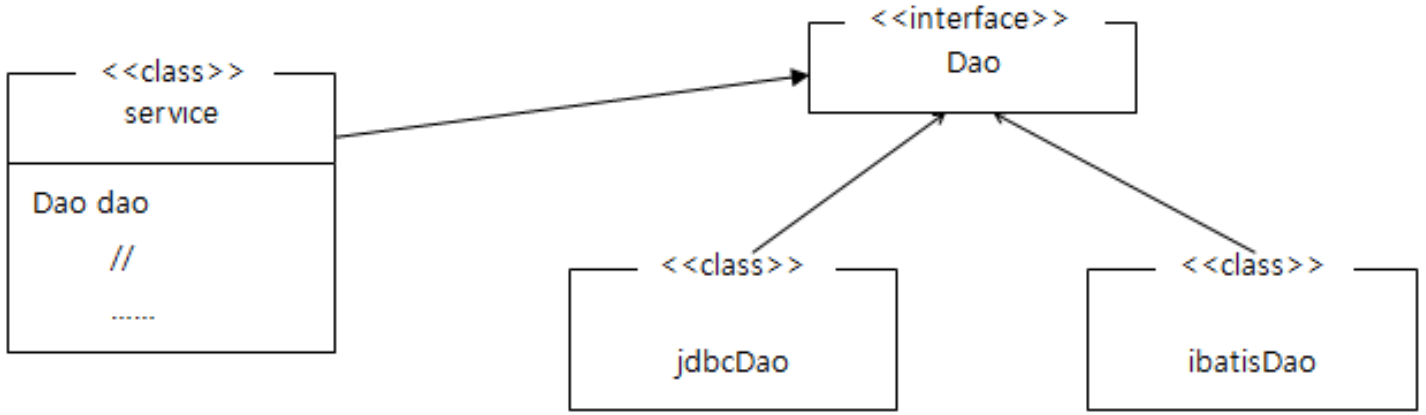
DepenDency Injection과 스프링프레임워크



- **A has a B. (A는 B에 의존한다.)**
  - 의존하는 객체를 취할 때 사용하는 클래스(A)측에서 호출. 이를 제어순행이라고 함
- **DI는 의존하는 객체에 대한 취득(획득)의 책임이 사용하는 class에 있지 않고 스프링 컨테이너가 제공(주입)해 준다.**
- 즉, A class에서 B class의 메소드를 사용하기 위해서는 B b = new B();를 통해 B class의 instance인 b를 생성해야 하는데, 스프링의 DI는 A class를 생성할 때 B class의 instance(객체)를 생성하여 주입까지 해주는 기능을 제공.
  그러므로 개발자는 A class에서 B b = new B();라는 식의 코드를 작성할 필요가 없이 스프링 컨테이너가 만들어 놓은 instance(bean)를 가지고 와서 사용만 하면 됨. (xml 설정을 통해)

DepenDency Injection과 스프링프레임워크





DepenDency Injection과 스프링프레임워크

- **DAO Interface를 통한 의존성 낮춤**

  B class의 메소드 이름이 바뀌면 A class의 메소드 이름도 바뀌어야 한다.(의존성이 높음).
  이는 내가 의존하는 class가 바뀌면 그것을 사용하는 모든 class가 전부 수정해야 한다는 문제점을 가지고 있다.
  이런 결합도가 높은 코드를 위해 interface 기반으로 만들어 결합도를 낮추어야 한다.

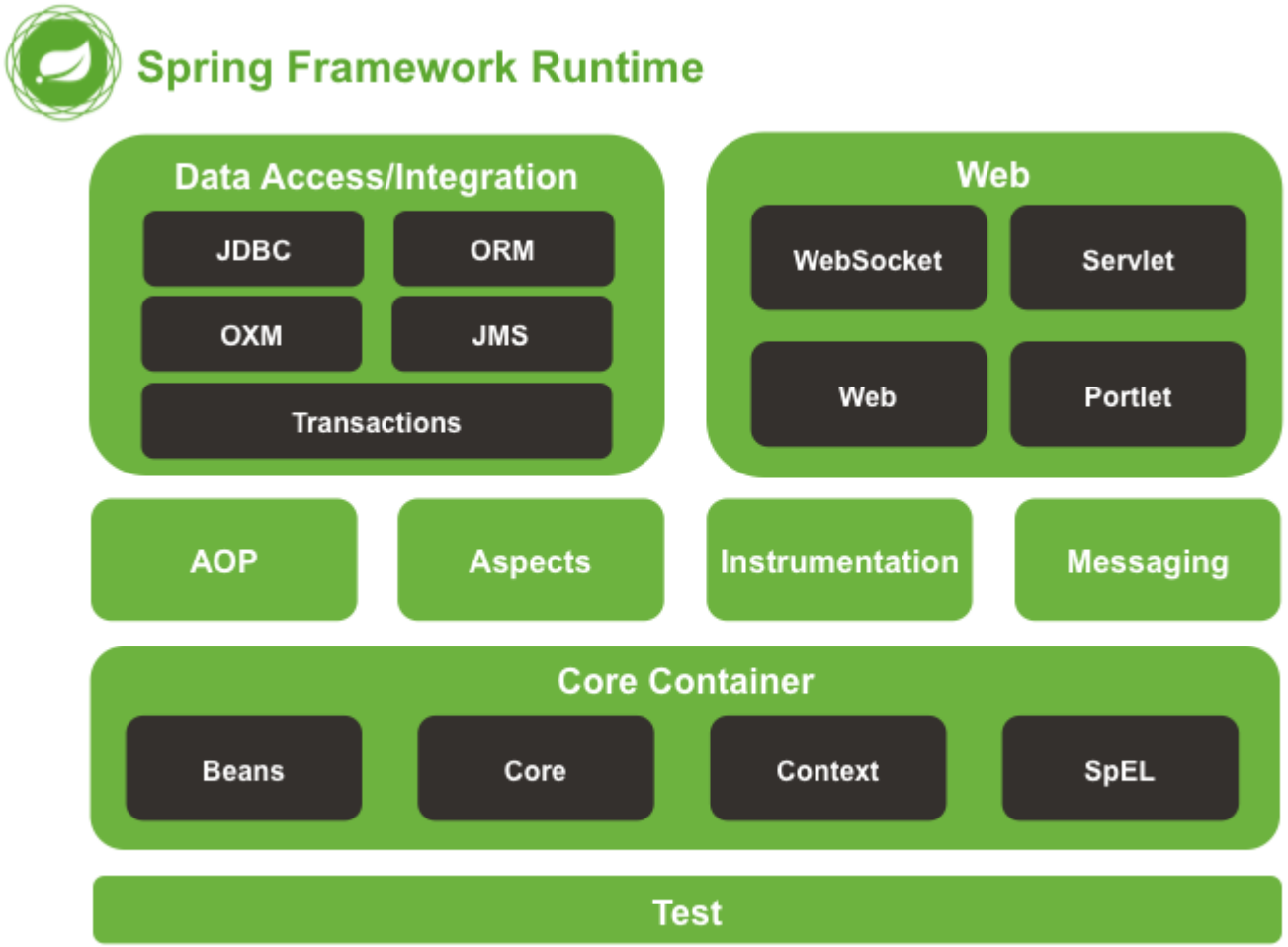
  dao = new jdbcDao();

  dao = new ibatisDao();

  이런 식으로 사용하면 의존성이 낮아짐. 스프링의 경우 컨테이너가 빈 팩토리 역할로 설정 파일에서만 필요한 설정을 해주면 기존코드 변경하지 않고도 dao의 값(jdbc인지, ibatis인지)을 가져와 사용할 수 있음. 왜냐 객체의 의존성을 xml에서 설정가능 하기 때문이다.
  즉, **코드의 변경 없이 xml의 설정만으로 개발자가 원하는 객체의 주입으로 바꿀 수 있음.**

스프링 프레임워크의 모듈

- Spring은 20여개의 모듈로 구성



스프링 프레임워크의 모듈

모듈 명	설 명
Beans	BeanFactory 인터페이스를 통해 구현됨
Core	프레임워크의 가장 기본적인 부분 컨테이너 기능을 수행하기 위해 의존성 주입[DI] 기능을 제공
Context	spring-core, spring-beans 모듈을 확장해서 국제화, 이벤트 처리, 리소스 로딩, 서블릿 컨테이너를 위한 컨텍스트 생성 등의 기능을 추가로 제공. ApplicationContext 인터페이스를 통해 구현됨
Expression Language	객체에 접근하고 객체를 조작하기 위한 표현 언어를 제공
AOP	AOP Alliance에 호환되는 AOP 구현을 제공
Aspects	AspectJ와의 통합을 제공
Web(MVC/Remoting)	Spring MVC를 제공하며 struts와도 연동 기능 제공등 웹 관련 기능 지원
Data Access/Integration	JDBC를 위한 템플릿 제공. 따라서 간결한 코드로 JDBC 프로그램 가능, iBatis 및 하이버네이트 등의 ORM api를 위한 통합 레이어 제공. Spring이 제공하는 트랜젝션과의 연동 지원

스프링 프레임워크의 설치와 모듈구성

- **MAVEN 이용하기**
  - 의존 관계의 library들도 자동 다운로드

http://repo.springsource.org/release

https://mvnrepository.com/

스프링 프레임워크의 설치와 모듈구성

- **Spring Project 를 위한 구성**

  - **Spring Framework library**

  - **설정 메타정보 파일**
    - **Spring은 설정 메타데이터 정보를 필요로 함**
    - 이 설정 메타데이터는 **Spring 컨테이너에 객체 생성 및 관계 설정 내용을 XML 또는 properties[프로퍼티] 파일, 소스코드 애노테이션과 같은 외부 리소스로 작성**

  - **자바 소스들**
    - Spring 빈
    - Spring 빈 사용 클래스들
    - 이외의 자바 클래스들

STS 다운로드

- **http://spring.io/tools**
  - Spring Tool Suite는 Spring 응용 프로그램 개발을 위해 사용자 정의 된 Eclipse 기반 개발 환경입니다.

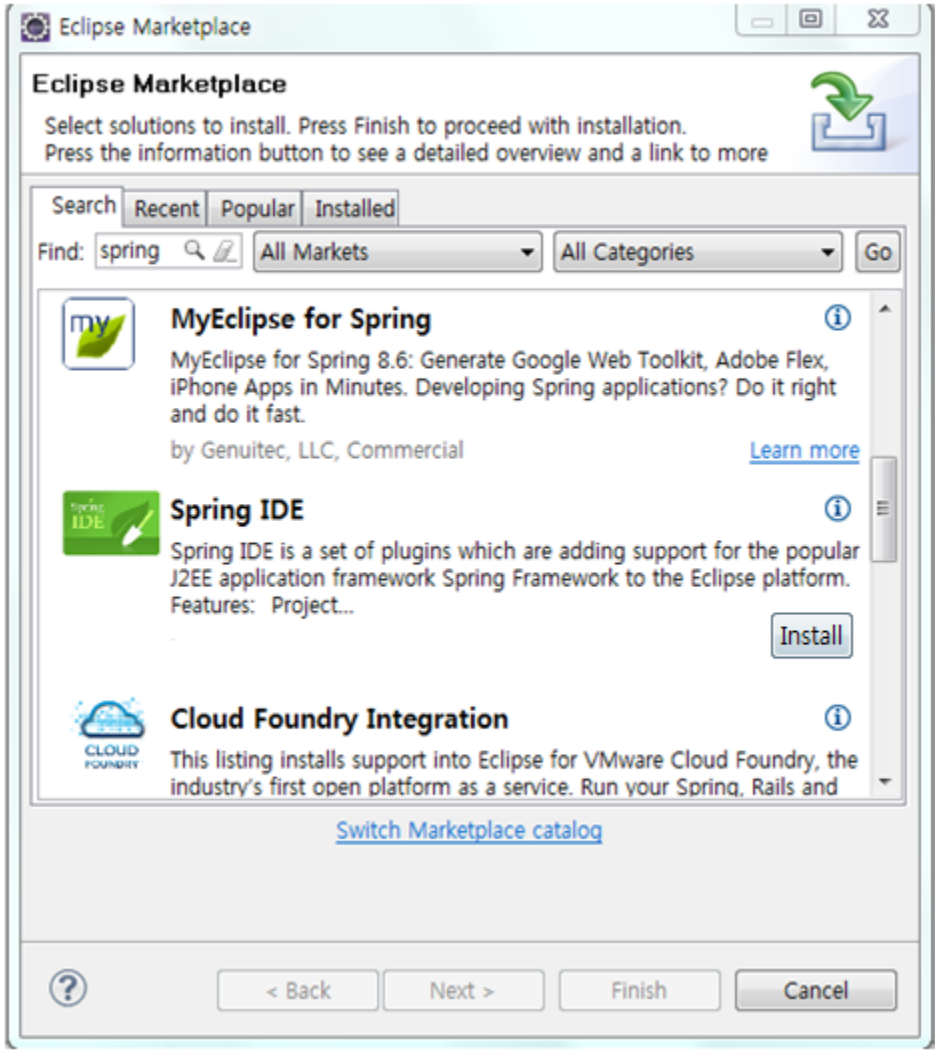
    Pivotal tc Server, Pivotal Cloud Foundry, Git, Maven, AspectJ 등의 통합을 포함하여 Spring 애플리케이션을 구현, 디버깅, 실행 및 배포 할 수 있는 환경을 제공합니다.

  - 이클립스 4.17.부터 java11을 요구하기 때문에 JAVA8 사용시
  - Spring Tool Suite 3.9.13 (New and Noteworthy)

    full distribution on Eclipse 4.16 을 선택해야 함.

스프링 프레임워크의 설치와 모듈구성

- **Spring IDE 설치**



스프링 프레임워크의 설치와 모듈구성

- **Spring IDE 설치**