Chapter01 프레임워크의 개요

1. 프레임워크

프레임 워크

프레임워크(Framework)

- **소프트웨어 관점에서 프레임워크는 아키텍처(architecture)에 해당하는 골격 코드를 제공함**
  - 아키텍처 : 어플리케이션을 개발할 때 가장 중요한 요소 중 하나인 어플리케이션의 기본 구조를 의미
  - 골격 코드 : 아키텍처의 골격 코드를 프레임워크가 제공함

- **프레임워크를 사용하지 않은 어플리케이션의 개발은 개발자의 능력에 의존함**
  - 개발자의 능력에 따라 프로젝트의 산출물의 품질이 달라질 수 있음
  - 유지보수에 문제가 발생할 수 있음
    개발자의 경험에 의한 유지보수가 진행됨
    개발자가 유지보수를 수행할 수 없는 상황이 발생할 경우 많은 문제가 발생할 수 있음

- **프레임워크는 어플리케이션의 개발에 공통으로 필요한 하부 구조를 제공함**
  - 비기능적인 요소(성능, 보안, 확장성, 안정성)등을 구현하는 라이브러리를 제공함
    개발자는 하부구조 구현이 신경 쓰지 않고 비즈니스로직만 개발 할 수 있음

프레임워크

프레임워크를 사용하는 이유

- **비기능적인 요소를 초기 개발 단계마다 개발자가 구현하지 않아도 됨**
  - 기본적으로 프레임워크 라이브러리 형식으로 제공
    프레임워크는 아키텍처에 해당하는 골격 코드를 제공하므로 개발자는 비즈니스로직 개발에만 집중할 수 있음
  - 빠른 시간에 구현이 가능
- **반복적으로 발생될 수 있는 문제를 해결하기 위한 solution 들을 제공**
  - 예를 들어, 한글 처리와 같은 문제를 해결하는 솔루션을 제공
- **쉬운 프로젝트와 산출물 관리**
  - 같은 프레임워크가 적용된 어플리케이션은 아키텍처가 같으므로 유지보수가 쉬움
- **개발자들의 역량 보완**
  - 프레임워크를 사용하면 개발자들 간의 능력 간격을 좁힐 수 있음

프레임워크의 주요 구성 요소

- **IoC(Inversion of Control) : 제어의 역전**
  - 객체의 생성부터 소멸까지의 라이프 사이클을 개발자가 아닌 프레임워크가 수행
  - 컨테이너 역할을 수행하는 프레임워크에 제어의 권한을 위임하여 개발자의 코드 구현을 줄임
  - 스프링 프레임워크의 주요 개념으로 의존성 주입(Dependency Injection)을 통해 IoC를 관리함

- **라이브러리 (Library)**
  - 프레임워크는 특정 부분의 기술적인 구현을 라이브러리 형태로 제공함

- **디자인 패턴 (Design Pattern)**
  - 소프트웨어 개발 과정 중 발견된 노하우를 축척해 이름을 붙여 재사용하기 좋은 형태로 제공하는 것을 의미
    대표적인 예는 GoF(Gang of Four) 디자인 패턴으로 23가지 디자인 패턴을 제공함
  - 디자인 패턴을 따른다면 팀 프로젝트를 진행해도 범용적인 코딩 스타일이 가능하며, 유지보수도 용이함
  - 대부분의 프레임워크는 디자인패턴을 적용하고 있음

주요 프레임워크

기능	프레임워크
웹(MVC)	Spring MVC, Struts2, WebWork, PlayFramework
OR(Object-Relation) 매핑	MyBatis, Hibernate, JPA, Spring JDBC
AOP(Aspect Oriented Programming)	Spring AOP, AspectJ, JBoss AOP
DI(Dependency Injection)	Spring DI, Google Guice
Build와 Library 관리	Ant, Maven, Gradle
단위 테스트	jUnit, TestNG, Catus
Javascript	jQuery, AngularJS, Node.js

2. 스프링 프레임워크

스프링 프레임워크의 개요

스프링 프레임워크(Spring Framework)

- 2004년 로드 존슨(Rod Johnson)에 의해 개발된 오픈소스 프레임워크
- EJB(Enterprise Java Beans)의 문제점 대두
  - 스프링 프레임워크가 등장하기 전 대부분의 자바 기반 엔터프라이즈 어플리케이션은 EJB로 개발됨
  - EJB 기술은 개발자들에게 많은 부담을 주고 있었음
    스펙이 너무 복잡해 학습에 많은 시간이 필요하며 유지보수도 어려움
    EJB로 개발된 컴포넌트를 실행하기 위해서는 고가의 WAS가 필요 했었음
  - EJB는 스프링 프레임워크가 등장할 때까지 대부분의 엔터프라이즈 어플리케이션 개발에 사용된 주요 기술로 사용됨
- 스프링 프레임워크는 경량 컨테이너 프레임워크임
  - POJO(Plain Old Java Object)를 사용해 구현
  - 개발에 있어서 EJB보다 간단함
  - 별도의 디자인 패턴을 신경 쓰지 않아도 됨
    스프링 프레임워크는 디자인 패턴이 적용되어 사용되므로 프레임워크 사용 자체가 디자인 패턴의 사용을 의미

스프링 프레임워크의 개요

스프링 프레임워크의 특징

- **Java Enterprise Application의 개발을 편하게 해주는 오픈소스 경량 어플리케이션 프레임워크**
  - 어플리케이션 프레임워크(Application Framework)
    특정 기술이나 업무 분야에 국한되지 않고 어플리케이션 전 영역을 포괄하는 범용적인 프레임워크
  - 경량 프레임워크(Lightweight Framework)
    단순한 컨테이너를 사용해 엔터프라이즈 어플리케이션 개발을 위한 고급 기술을 사용할 수 있는 프레임워크

- **EJB에 비해 크기가 작음**
  - 스프링은 여러 개의 모듈로 구성되어 있으며, 각 모듈은 하나 이상의 JAR 파일로 구성됨
    소수의 JAR만 사용해도 개발과 실행이 가능함
  - POJO 형태의 객체를 사용하고 관리함
    POJO는 어떤 프레임워크에도 종속되지 않은 가벼운 객체임
  - 고가의 WAS 없이 tomcat 만으로 수행 가능

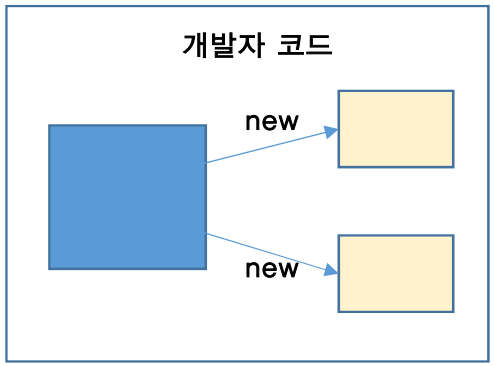
- **Enterprise 어플리케이션 개발에 용이**
  - 개발자는 비기능적인 하위 레벨(보인, 로깅 등)에 신경 쓰지 않고 비즈니스로직 개발에 전념할 수 있음

스프링 프레임워크의 구성요소

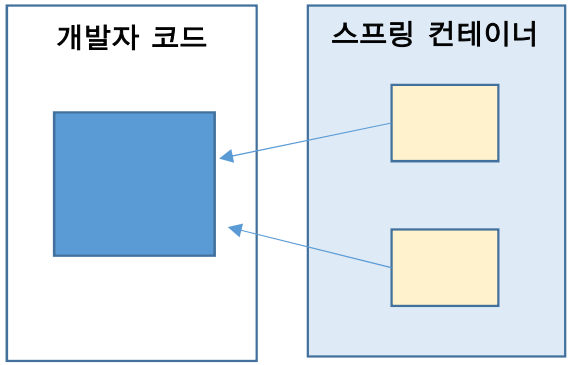
의존 주입 (Dependency Injction)

- **스프링은 의존 주입을 사용해 어플리케이션을 구성하는 객체간 낮은 결합도를 유지할 수 있음**
  - 비즈니스 컴포넌트를 개발할 때 가장 중요한 것은 낮은 결합도와 높은 응집도임

- **의존 주입은 확장 가능한 객체를 외부에서 생성하고, 객체 간의 의존을 동적으로 연결해 주는 개념임**
  - 객체는 개발자가 자바 코드로 직접 생성하는 것이 아니라 컨테이너가 생성해 개발자 코드에 주입하는 구조임
  - 이를 제어의 역행(IoC : inversion of Control)이라고 함



DI를 사용하지 않는 경우

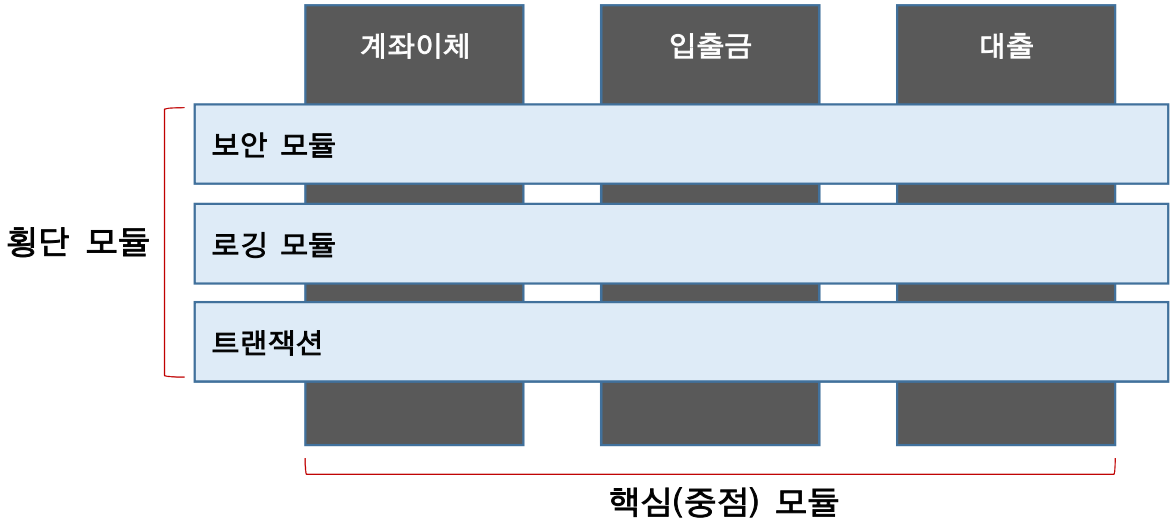


DI를 사용하는 경우

스프링 프레임워크의 구성요소

관점 지향 프로그래밍 (AOP : Aspect Oriented Programming)

- **어플리케이션의 비즈니스로직에서 공통적으로 사용되는 로직을 별도의 클래스로 분리**
  - 분리를 통해 공통된 로직이 비즈니스로직들에 직접 명시되지 않도록 하고
  - 분리된 기능의 수행 여부를 비즈니스로직에서 별도로 명시하지 않고 선언적으로 처리되도록 함
  - 개발자가 응집도가 높게 개발할 수 있도록 지원함



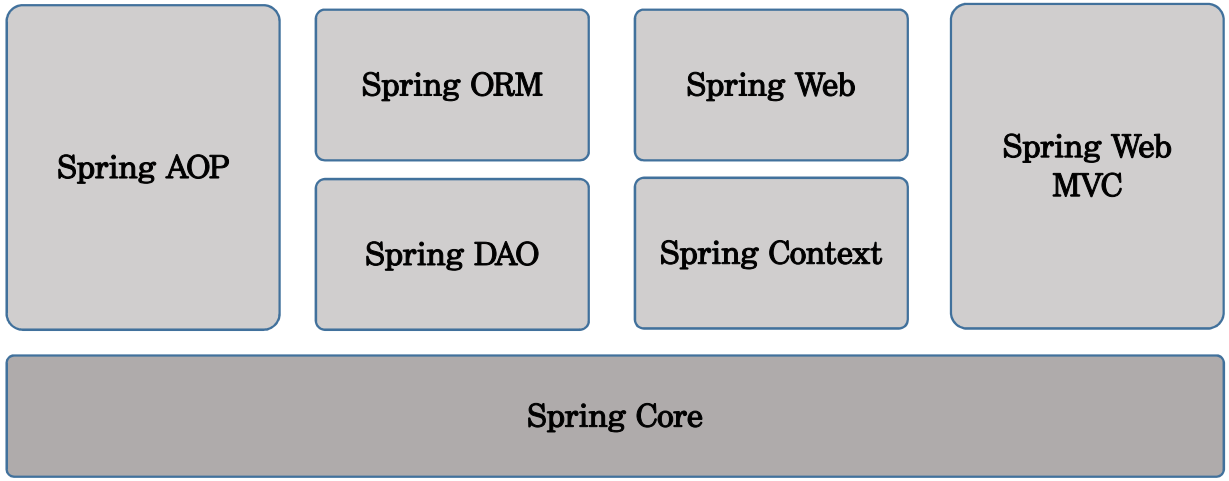
스프링 프레임워크의 구성요소

컨테이너

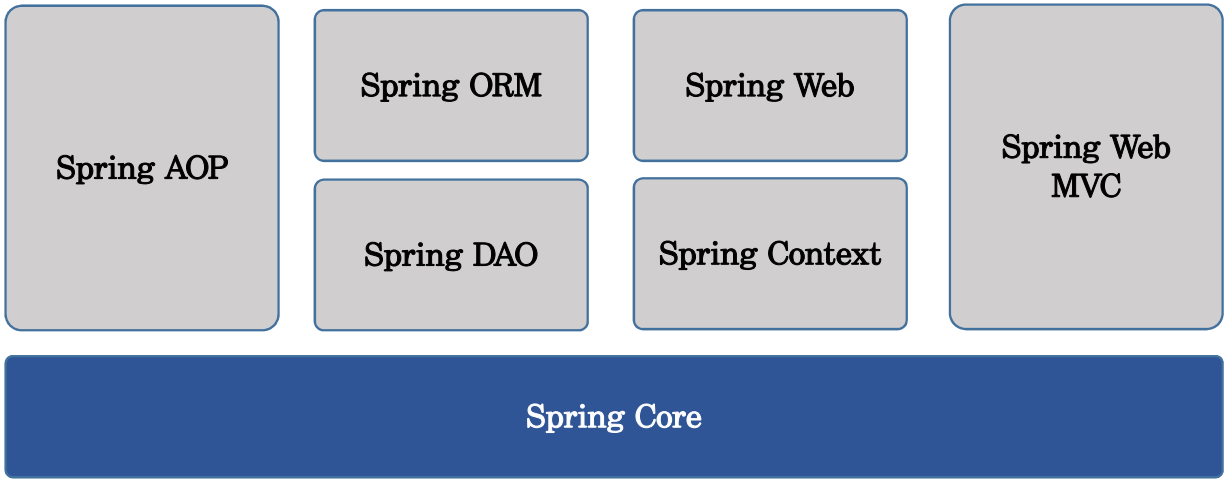
- **특정 객체의 생성과 관리를 담당하고 객체를 운영하기 위한 다양한 기능을 제공**
  - 일반적으로 서버 안에 포함되어 구동됨
  - 컨테이너의 예로 Servlet Container와 EJB Container가 있음
- **스프링 컨테이너는 프레임워크에서 객체를 생성하고 관리하는 역할을 수행**
  - 자바 객체의 Life Cycle을 관리함
  - 의존 주입, AOP 등의 기능으로 객체 사이의 의존 관계를 관리함
  - IoC 컨테이너라고도 함

스프링 프레임워크의 기능적 구성요소

스프링 프레임워크의 기능적 구성요소

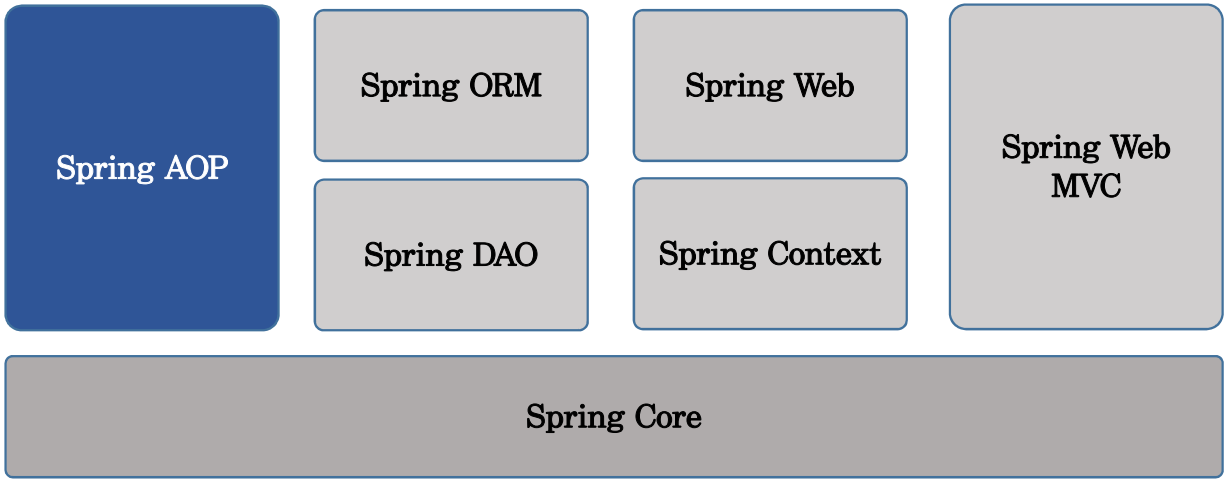


스프링 프레임워크의 기능적 구성요소



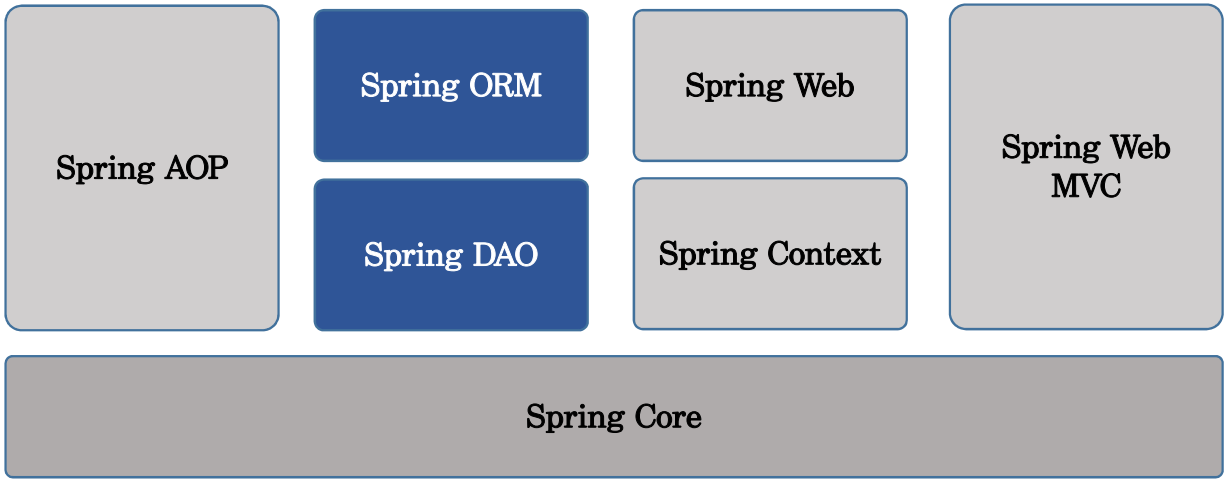
기능	설명
Spring Core	• Spring Framework의 기본 기능(컨테이너의 기능)을 제공 • 컨테이너의 역할을 하는 클래스들로 구성 • Core 내의 BeanFactory는 스프링의 기본 컨테이너이면서 DI의 기반이 됨

스프링 프레임워크의 기능적 구성요소



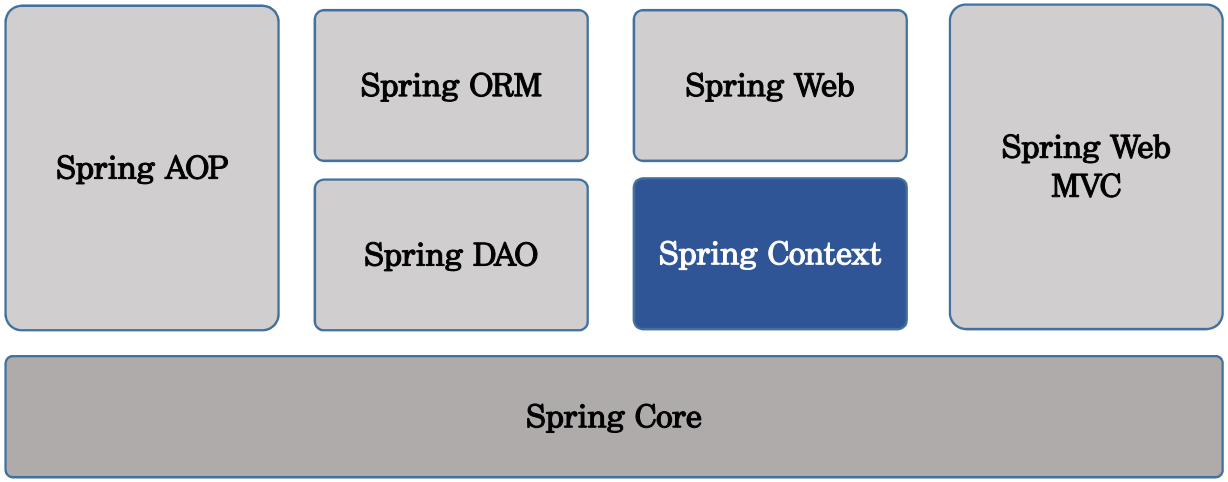
기능	설명
Spring AOP	• AOP 모듈을 통해 관점 지향 프로그래밍을 지원

스프링 프레임워크의 기능적 구성요소



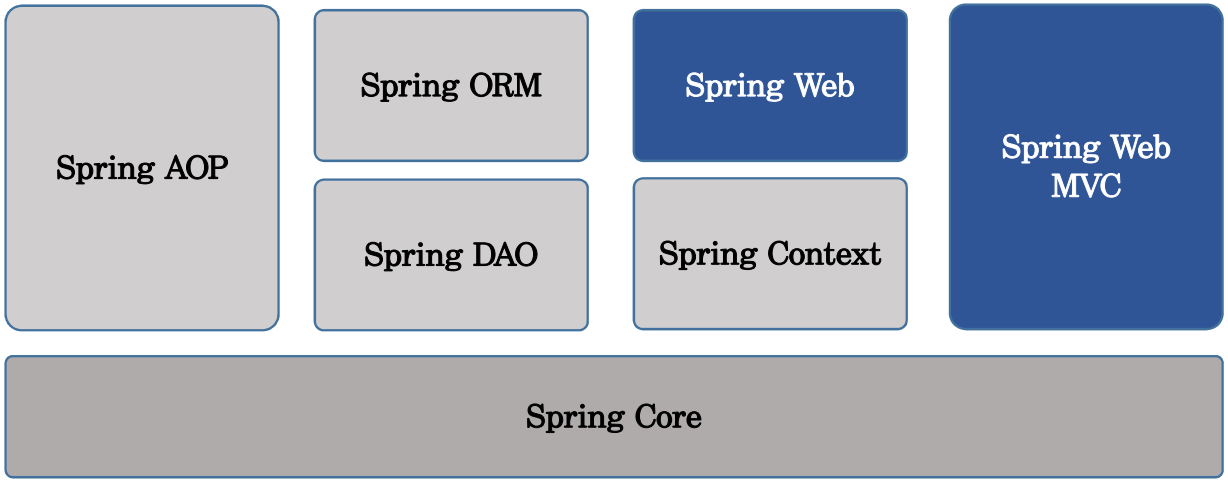
기능	설명
Spring ORM	• MyBatis, Hibernate, JPA 등 주로 사용되는 ORM 프레임워크와 연결 지원 • ORM 관련 제품들을 스프링의 기능을 조합하는 기능을 지원
Spring DAO	• JDBC에 대한 추상화 계층으로 JDBC 코딩이나 예외 처리 부분은 간소화 함 • AOP 모듈을 사용해 트랜잭션 기능도 제공

스프링 프레임워크의 기능적 구성요소



기능	설명
Spring Context	• Spring Core 부분을 확장한 개념 • BeanFactory 개념을 확장해 국제화, 애플리케이션 생명주기, 유효성 검증 지원

스프링 프레임워크의 기능적 구성요소



기능	설명
Spring Web	• 브라우저에서 동작하는 웹 어플리케이션 개발에 필요한 기본 기능을 제공 • Webwork나 struts와 같은 다른 어플리케이션 프레임워크와의 통합을 지원
Spring Web MVC	• 프리젠테이션 로직과 비즈니스 로직이 분리된 웹 어플리케이션 개발 지원