Spring

라이브러리와 프레임워크 차이

- 공통점
  - 특정 문제를 일반적인 방법(모델 뷰 컨트롤러)으로 해결하기 위한 코드를 제공한다.
  - 재활용할 수 있다.
- 차이점
  - Framework 는 원하는 기능을 구현하기 위하여 일정한 형태 (골격)를 제공한다. e.g. WebApplication, 레디스 영속 프레임워크 등등
  - 기능(함수)의 집합. e.g. 레디스 클라이언트 라이브러리
  - 주도권
  - 레디스 영속 프레임워크 vs 레디스 클라이언트 라이브러리

프레임워크 사용해아하는 이유

- 기능적 요구 사항과 비 기능적 요구사항(비기능 요구사항 : 프레임워크 쓰면 구현되있음)
- 반복되는 기능
- 비즈니스에 집중
- 빠른 개발

스프링 프레임워크

- 자바 엔터프라이즈 개발을 편하게 해주는 오픈소스 경량급 애프리케이션 프레임워크
- 스프링(Spring) 프레임워크는 가볍고 엔터프라이즈 애플리케이션을 구축하는 하나의 대안(Potential one-stop-shop)입니다.
- 스프링은 모듈화되어 있어 필요한 부분만 사용할 수 있으며, 나머지는 가져올 필요가 없습니다.
- 스프링은 비침투적(Non-intrusive)으로 설계되어 있어 도메인 로직 코드는 일반적으로 프레임워크에 대한 의존성이 없습니다.
- 스프링 프레임워크는 Java 애플리케이션 개발에 대한 종합 인프라 지원을 제공합니다.
- 스프링은 인프라를 처리하므로 애플리케이션에 집중할 수 있게 해줍니다.
- 스프링은 "Plain Old Java Objects" (POJOs)에서 애플리케이션을 구축하고 엔터프라이즈 서비스를 POJO에 비침투적으로 적용할 수 있도록 해 줍니다.

```
GPT

스프링(Spring) 프레임워크는 자바 엔터프라이즈 개발을 편하게 해주는 오픈소스 경량급 애프리케이션 프레임워크입니다. 스프링은 IoC(Inversion of Control)와 DI(Dependency Injection)를 지원하여 객체 간의 결합도를 낮추고 유지보수성을 높일 수 있습니다. 또한 스프링은 AOP (Aspect Oriented Programming)를 지원해서, 핵심 로직과 부가적인 기능을 분리시켜서 개발할 수 있게 해줍니다. 따라서 스프링 프레임워크를 사용하면 비즈니스 로직에 집중하여 빠른 속도로 애플리케이션을 구현할 수 있습니다.
```

경량 컨테이너로서, Spring Bean 을 직접 관리한다.

- Spring Bean 객체의 **라이프 사이클**을 관리한다.
    - Spring Bean : **Spring Container** 가 관리하는 중요 객체
- Container - Spring Bean **객체의 생성, 보관, 제거**(라이브 사이클)에 관한 모든일을 처리한다.

POJO(Plain Old Java Object) 기반의 프레임워크.

- 일반적인 J2EE 프레임워크와 비교하여, 특정한 인터페이스를 구현하거나 상속을 받을 필요가 없다.
- 기존에 존재하는 라이브러리를 사용하기 편리하다.

제어 역전(IoC : Inversion of Control)

- 컨트롤의 제어권이 사용자(개발자)가 아니라 프레임워크따라 Spring에서 사용자의 코드를 호출한다.
- 의존성 주입(DI : Dependency Injection)
- DDD, TDD와 같은 프로그래밍 개발론에도 적합한 프레임워크이다.

관점 지향 프로그래밍(AOP : Aspect-Oriented Programming)을 지원.

- 성능측정 어플리케이션 관점 : 스레드 시작, 종료 등 처리시간이 궁금,
- 보안의 관점 : rest api 등을 시작하기전에 인증인가가 되어있어야함.
- 복잡한 비지니스 영역의 문제와 공통된 지원 영역의 문제를 분리할 수 있음.
- 문제 해결을 위한 집중.
- e.g. Transaction, Logging, Security and etc.

영속성과 관련된 다양한 서비스 지원.

- e.g. MyBatis, Hibernate, JdbcTemplate 등등
- 영속성 : 영원히 저장하고, 유지시킨다.
    - 기능 : DB 읽기, 쓰기, 저장, 수정,삭제
    - 메모리는 휘발성, DB는 비휘발성

높은 확장성 및 범용성 그리고 Eco System

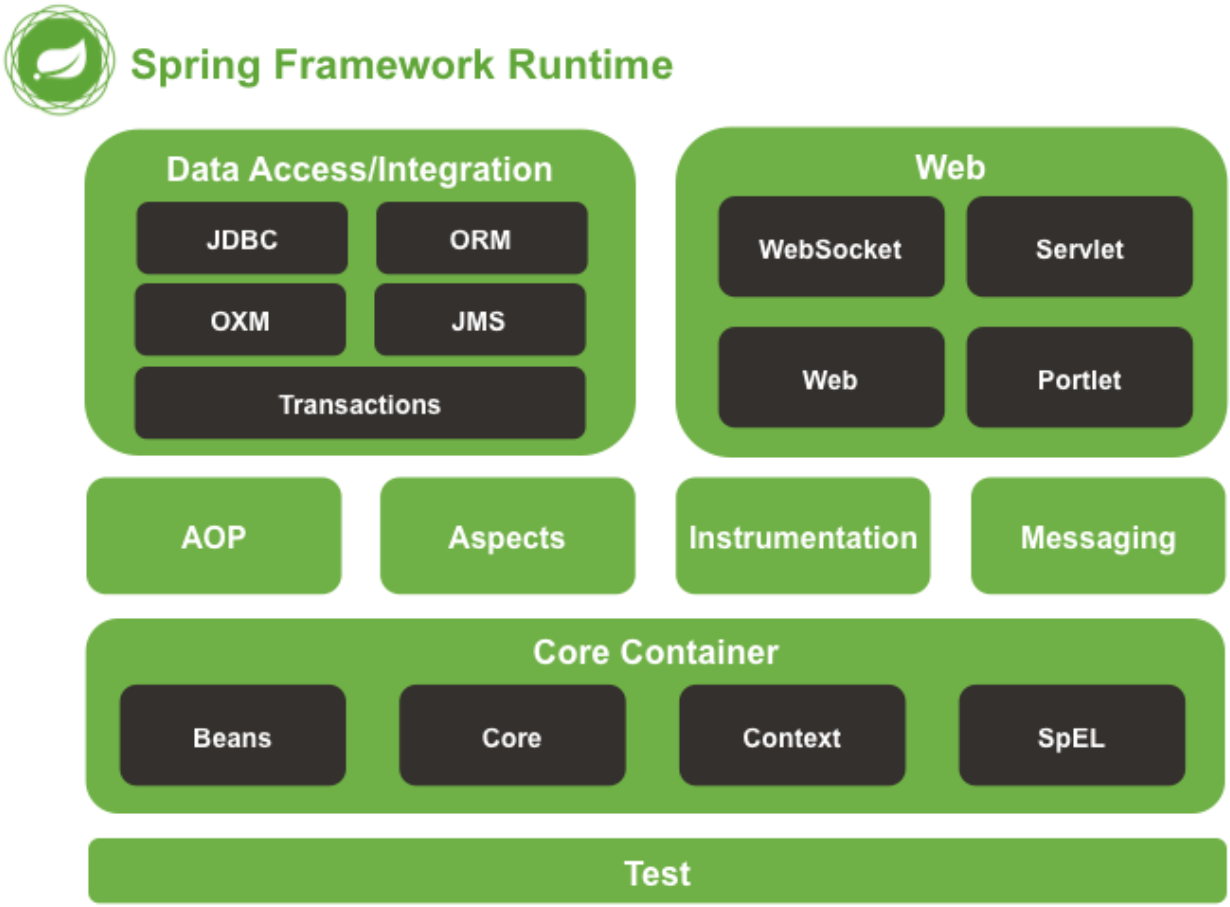
한국의 Spring Framework

- 전자정부 표준 프레임워크
    - Spring Framework 기반
    - https://www.egovframe.go.kr/
    - 대기업 SI 3사 (Samsung SDS, SK C&C, LG CNS) 참여
    - 정부 시스템 프레임워크 표준화
- 대부분의 Java 기반 Back End 서비스에서 사용

Spring Framework State

- https://tanzu.vmware.com/content/ebooks/the-state-of-spring-2021

Spring Framework Modules



Core Container

모듈	설명
spring-core	Spring의 핵심 유틸리티가 포함된 모듈
spring-context	Spring의 ApplicationContext 클래스들, 스케줄링 클래스들, AOP 관련 클래스들, Cache 관련 클래스들을 제공한다
spring-context-support	Third-part 라이브러리를 통합하여, Spring ApplicationContext
spring-beans	Spring Bean 과 관련된 클래스와 어노테이션들을 제공한다.
spring-expression	Spring Expression Language (SpEL) 관련 기능을 제공한다.

AOP

모듈	설명
spring-aop	Proxy-based AOP support
spring-aspects	AspectJ based aspects

Data Access/Integration, Web, Test

모듈	설명
spring-jdbc	DataSource 설정 및 JDBC 접근을 포함한 JDBC 지원 패키지
spring-jms	JMS 메시지 전송 및 수신을 위한 헬퍼 클래스를 포함한 JMS 지원 패키지
spring-messaging	메시징 아키텍처와 프로토콜 지원 패키지
spring-orm	ORM(Object/Relational Mapping)으로, JPA와 Hibernate 지원을 포함한다.
spring-oxm	Object/XML 매핑을 위한 패키지
spring-tx	DAO 지원과 JCA(Java EE Connector Architecture) 통합을 포함한 트랜잭션 인프라스트럭처 패키지
spring-web	클라이언트 및 웹 리모팅을 포함한 웹 지원 패키지
spring-webmvc	Web 애플리케이션용 REST 웹 서비스 및 MVC 구현을 위한 패키지
spring-webmvc-portlet	포틀릿(Portlet) 환경에서 사용되는 MVC 구현을 위한 패키지
spring-websocket	WebSocket 및 SockJS 구현을 포함하며, STOMP를 지원하는 패키지

Spring Projects

Spring Projects	설명
Spring Boot	스프링 어플리케이션을 쉽게 만들고 실행할 수 있도록 자동구성을 지원해주는 프로젝트
Spring Framework	스프링의 핵심 기능인 DI, AOP 등을 제공하는 프레임워크
Spring Cloud DataFlow	메시징 교환을 위한 SEDA(Staged, Event-Driven Architecture) 구축을 지원하는 프레임워크
Spring Cloud	클라우드 네이티브 애플리케이션 개발에 필요한 인프라스트럭처를 제공하는 프로젝트
Spring Data	JPA, MongoDB, Redis, Elasticsearch 등 다양한 데이터베이스와 연동하여 데이터 액세스를 위한 라이브러리를 제공하는 프로젝트
Spring Integration	Enterprise Integration Pattern을 구현하기 위한 라이브러리를 제공하는 프로젝트
Spring Batch	대용량 데이터 처리를 위한 일괄처리(Batch)를 구현하기 위한 라이브러리를 제공하는 프로젝트
Spring Security	인증과 권한 부여를 위한 라이브러리를 제공하는 프로젝트
Spring HATEOAS	Hypermedia As The Engine Of Application State를 구현하기 위한 라이브러리를 제공하는 프로젝트
Spring AMQP	메시지 큐(Message Queue)를 구현하기 위한 라이브러리를 제공하는 프로젝트

- https://spring.io/projects

Spring Boot vs Spring Core?

- 대세는 Spring Boot Project // 미리 준비해져 개발을 시작만하면 됨
    - 대용량 서비스
    - Cloud
    - MSA
    - Pivotal
- 하지만 기본은 Spring Core

스프링코어

- 일일이 하나하나 올려야함

스프링 부트

- 이미 만들어진 걸 가이드에 맞추면 동작하는 애플리케이션

스프링 관련

- 스프링 코어를 기본으로 함

실습1

```
1. IntelliJ 에서 maven 프로젝트를 생성합니다.
- NEW PROJECT , 아키텍트 사용 x

2. maven 속성을 지정합니다.
- Name: springframework-core-message-sender
- GroupId: com.nhnacademy.edu.springframework
- ArtifactId: springframework-core-message-sender
- Version: 1.0-SNAPSHOT
3. pom.xml 파일에 spring-context 라이브러리 의존성을 추가 합니다.
<dependencies>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>5.3.17</version>
    </dependency>
</dependencies>
```