


SeSAC 4기, 

Spring 수업

WITH 팀 리처드



Spring이란?

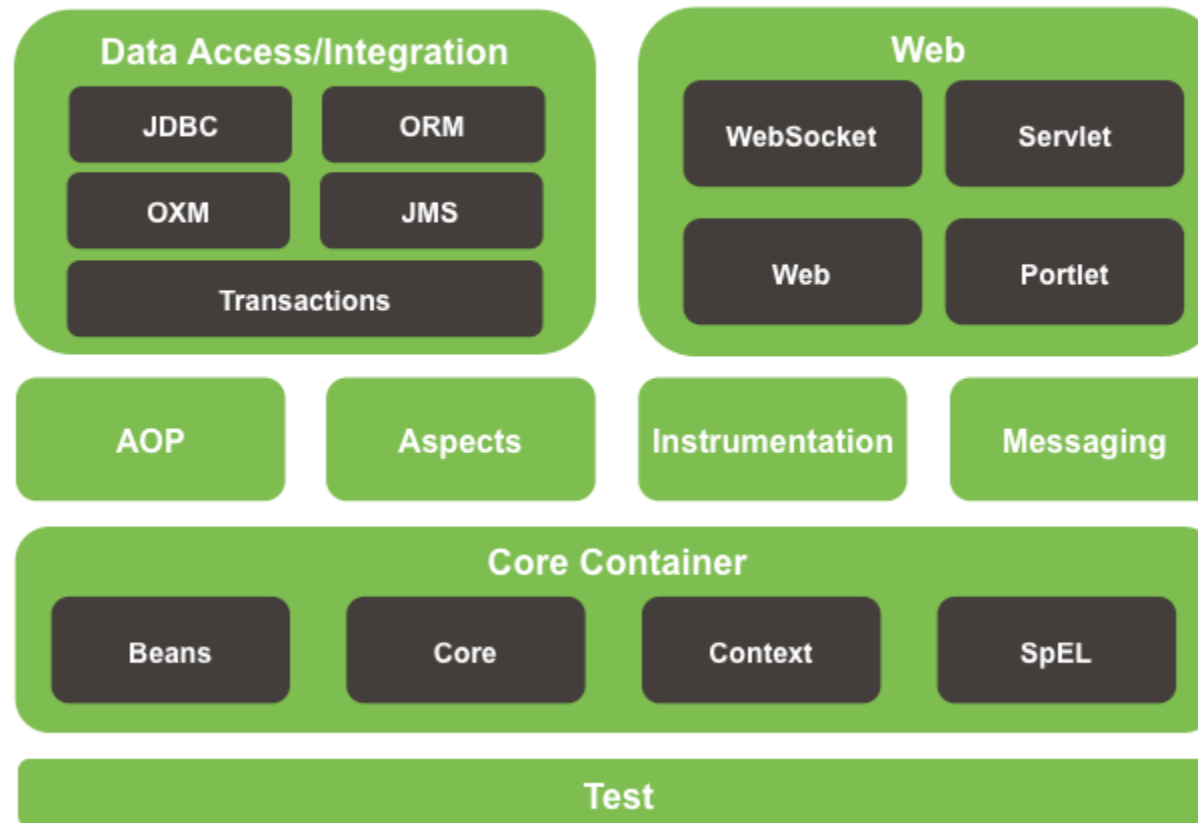
- Java 기반의 웹 어플리케이션을 만들 수 있는 **백엔드 프레임워크**
- Java로 다양한 어플리케이션을 만들기 위한 프로그래밍 툴
- 수많은 국내 기업과 해외 기업에서 많이 사용하는 프레임워크



Spring Architecture



Spring Framework Runtime



스프링 프레임워크 특징

1. IoC (Inversion of Control, 제어 반전)
2. DI (Dependency Injection, 의존성 주입)
3. AOP (Aspect Object Programming, 관점 지향 프로그래밍)
4. POJO (Plain Old Java Object 방식)

특징 1) IoC

- Inversion of Control, 제어 반전
- 객체의 생성부터 소멸까지 개발자가 직접 하는 것이 아닌 Spring Container 가 대신해주는 것
- 제어권이 개발자가 아닌 IoC 에 있으며, IoC 가 개발자의 코드를 호출해 필요한 객체를 생성, 소멸해 생명주기를 관리한다.

특징 1) IoC

1. 일반 객체 생성

```
class Sample {  
    private Apple apple = new Apple();  
}
```

2. IoC 객체 : 제어권이 외부에 있는 객체

```
class Sample {  
    private Apple apple;  
  
    public Sample(Apple apple){  
        this.apple = apple;  
    }  
}  
  
class SampleTest {  
    Apple apple = new Apple();  
    Sample sample = new Sample(apple);  
}
```

→ Apple 객체의 제어권이 Sample 에 있는 것이 아닌, SampleTest에게 있다.

→ 의존성을 역전시켜 제어권을 직접 갖지 않는 것이 **IoC**

특징 2) DI

- Dependency Injection, 의존성 주입
- 구성 요소의 의존 관계가 소스코드 내부가 아닌 외부의 설정 파일을 통해 정의되는 것.
- DI를 통해서 IoC를 이룬다.
- 이를 통해 코드 간의 **재사용률**을 높이고, **모듈 간의 결합도**를 낮출 수 있다.

특징 2) DI

방법 1. Field Injection (필드 주입)

```
//Field Injection  
@Autowired  
private FieldService fieldService;
```

방법 2. Setter Injection (수정자 주입)

```
//Setter Injection  
private SetterService setterService;  
  
@Autowired  
public void setSetterService(SetterService setterService) {  
    this.setterService = setterService;  
}
```

방법 3. Constructor Injection (생성자 주입)

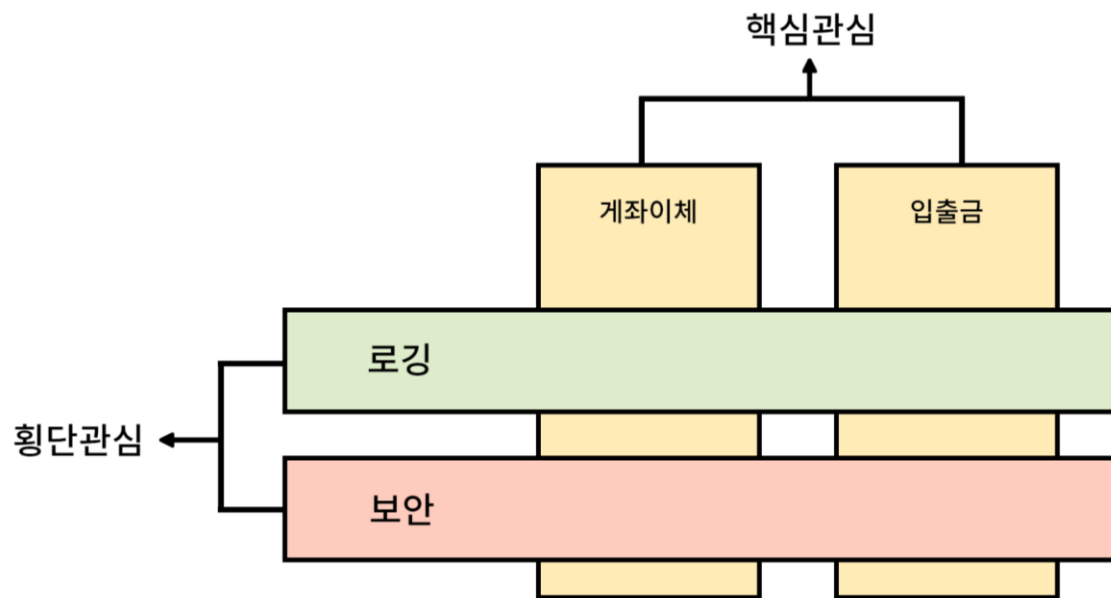
```
//Constructor Injection  
private final ConstructorService constructorService;  
  
@Autowired  
public ExampleComponent(ConstructorService constructorService) {  
    this.constructorService = constructorService;  
}
```


특징 3) AOP

- Aspect Object Programming, 관점 지향 프로그래밍
- 어떤 로직을 기준으로 핵심적인 관점, 부가적인 관점으로 나누어서 보고 그 관점을 기준으로 각각 모듈화하겠다는 것
- 기능을 **비즈니스 로직**과 **공통 모듈**로 구분한 후, 개발자의 코드 밖에서 필요한 시점에 비즈니스 로직을 삽입하여 실행되도록 한다.

특징 3) AOP

- 계좌이체와 입출금 로직을 처리할 때 공통적으로 로깅과 보안 작업을 수행해야 한다.
- 일반적으로 이 경우, 공통의 코드를 두 개의 로직에 모두 넣고 사용한다.
- AOP는 공통관심(로깅, 보안)을 따로 빼내어 객체별로 처리하는 것이 아닌 관점별로 외부에서 접근해 사용하도록 만든다.



특징 4) POJO

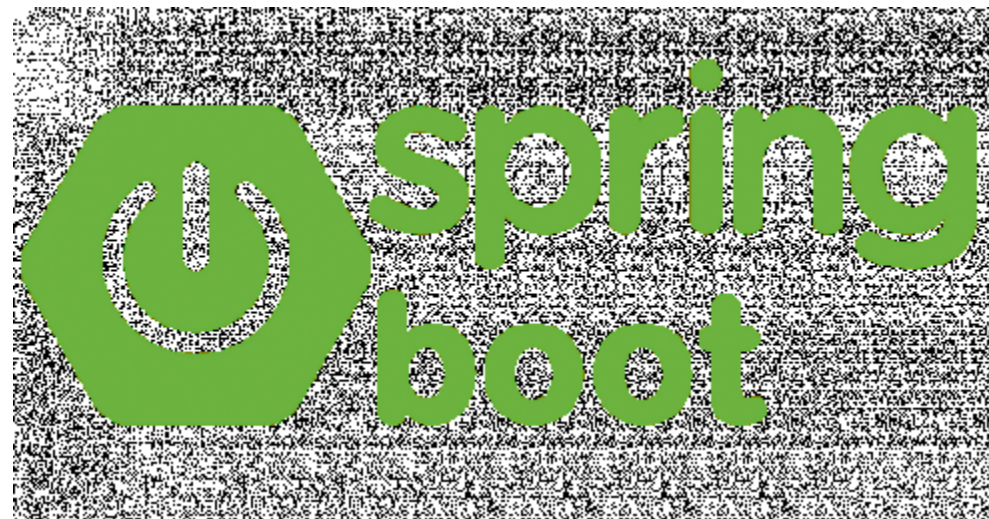
- Plain Old Java Object, 단순한 자바 오브젝트
- 객체 지향적인 원리에 충실하면서 환경과 기술에 종속되지 않고 필요에 따라 재활용될 수 있는 방식으로 설계된 오브젝트
- getter / setter 같이 기본적인 기능만 가진 Java 오브젝트

```
class Person{  
    private String name;  
    public String getName(){  
        return name;  
    };  
    public void setName(String name) {  
        this.name = name;  
    }  
}
```

Spring Boot란?

Spring Boot란?

- Spring을 더 쉽게 이용하기 위한 도구
- Spring을 이용할 시, 필요한 여러 가지 세팅 작업들 (ex) 톰캣 서버 설정, XML 설정 등) 없이 쉽고 빠르게 프레임워크를 사용할 수 있도록 만들어진 것



IntelliJ 설치하기

제공된 IntelliJ 설치 PDF를 참고해 IntelliJ 설치 완료하기