스프링 핵심 원리

!!! 객체 지향 프로그래밍

JPA : 자바 어플리케이션에서 관계형 데이터베이스를 사용하는 방식을 정의한 인터페이스이다. ( 자바에서 제공하는 api이다) ORM 기술 사용(객체와 디비의 데이터를 자동으로 매핑 해준다)

JDBC : DB에 접근 할 수 있도록 도와주는 자바 api 이다

Hibernate : JPA(인터페이스)의 구현체 이다

스프링 프레임워크, 스프링 부트

스프링 데이터, 스프링 세션, 시큐리티, rest docs, 배치, 클라우드

스프링 프레임워크 :

스프링 부트 : 스프링을 편리하게 사용할 수 있도록 지원,

좋은 객체 지향 특징

1. 추상화
2. 캡슐화
3. 상속
4. 다형성

!!!! 다형성 : 역할과 구현으로 구분하면 된다!!

역할 = 인터페이스, 구현 = 인터페이스를 구현한 클래스, 구현 객체

객체를 설계할 때 역할과 구현을 명확히 분리

클라이언트를 변경하지 않고, 서버의 구현 기능을 유연하게 변경할 수 있다.

객체의 협력이라는 관계

* 혼자 있는 객체는 없다
* 클라이언트 : 요청 서버 : 응답
* 수 많은 객체 클라이언트와 객체 서버는 서로 협력 관계를 갖는다

인터페이스를 안정적으로 잘 설계하는 것이 중요!!!! ( 역할이 변경되면 클라이언트 전부다 변경됨)

객체지향은 다형성이 제일 중요!!

IOC(제어의 역전) DI(의존관계 주입)

Spring 은 다형성을 자유롭게 사용하여 구현 할 수 있도록 도와주는 거라고 봐도 된다.

Solid

SRP : 단일 책임 원칙

* 한 클래스는 하나의 책임만 가져야 한다.
* 변경이 있을 때 파급 효과가 적으면 된다.

OCP : 개방 폐쇄 원칙

* 소프트웨어는 열려 있으나 변경에는 닫혀 있어야한다.
* 다형성을 활용 하면 된다.-

LSP: 리스코프 치환 원칙

* 다형성에서 인터페이스의 규약을 지켜야 한다.

ISP: 인터페이스 분리 원칙

* 인터페이스가 명확해지고, 대체 가능성이 높아진다.

DIP: 의존관계 역전 원칙

* 구현체가 아닌 역할에 의존해야 한다.

스프링은 다음 기술로 다형성 + OCP, DIP를 가능하게 지원

* DI: 의존관계, 의존성 주입
* DI 컨테이너 제공
* 클라이언트 코드의 변경없이 확장 가능