2020-10-14

\*스프링 부트

스프링을 편리하게 사용할 수 있도록 지원한다.

단독으로 실행할 수 있는 스프링 애플리케이션을 쉽게 생성

Tomcat 같은 웹서버를 내장해서 별도의 웹서버를 설치하지 않아도 된다.

손쉬운 빌드 구성을 위한 starter 종속성 제공

스프링과 3rd parth 라이브러리 자동구성

모니터링 제공

\*스프링

스프링 DI컨테이너 기술

스프링 프레임워크

스프링 부트, 스프링 프레임워크 등을 모두 포함한 스프링 생태계

\*스프링을 만든이유?

핵심 개념, 컨셉?

스프링은 **좋은 객체 지향** 애플리케이션을 개발할 수 있게 도와주는 프레임 워크다.

\*객체 지향 프로그래밍

객체들의 모임으로 파악, 각각의 객체는 메시지를 주고받고, 데이터를 처리할 수 있다.

유연하고 변경이 용이하다. -> 내부의 내용을 알지못하고 다른 것으로 대체 가능 ..?!

-다형성

클라이언트는 인터페이스만 알면 된다.

내부 구조를 몰라도 된다.

내부 구조가 변경되어도 영향을 받지 않는다.

대상 자체를 변경해도 영향을 받지 않는다.

오버라이딩(다형성으로 인터페이스를 구현한 객체를 실행 시점에 유연하게 변경할 수 있다.)

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

개발시 인터페이스를 안정적으로 잘 설계하는 것이 중요하다.!!!!

\*좋은 객체 지향 설계의 5가지 원칙 (SOLID)

SRP:단일 책임 원칙(single responsibility principle)

한 클래스는 하나의 책임만 가져야한다.(중요한 기준은 변경이다.)

OCP:개방-폐쇄 원칙(Open/closed principle)

확장에는 열려 있으나 변경에는 닫혀 있어야한다.(다형성)

새로운 클래스를 하나 만들어서 새로운 기능을 구현하는 것 .

구현 객체를 변경하려면 클라이언트 코드를 변경해야한다.

다형성을 사용했지만 OCP 원칙을 지킬 수 없다.

스프링 컨테이너가 객체를 생성하고, 연관관계를 맺어주는 별도의 조립, 설정자가 되어주는 것이다.

LSP:리스코프 치환 원칙(Liskov substitution principle)

프로그램의 객체는 프로그램의 정확성을 깨뜨리지 않으면서 하위 타입의 인스턴스로 바꿀 수 있어야 한다.

ISP:인터페이스 분리 원칙(Interface segregation principle)

특정 클라이언트를 위한 인터페이스 여러 개가 범용 인터페이스 하나보다 낫다

DIP:의존관계 역전 원칙(Dependency inversion principle)

프로그래머는 추상화에 의존해야지 , 구체화에 의존하면 안된다.

정리

다형성만으로는 구현 객체를 변경할 때 클라이언트 코드도 함께 변경된다.

OCP, DIP를 지킬 수 없다. 뭔가 더필요하다!!!

\*스프링 ? 객체지향 ?

DI : 의존관계, 의존성 주입, DI 컨테이너제공

2020-10-15

\*비즈니스 요구사항과 설계

요구사항이 미확정 되어 있어도 인터페이스를 사용해서 미리 구현이 가능하다 .(역할과 구현을 참고)

\*도메인 설계

클라이언트 -> 서비스 -> 저장소( 메모리 저장소, DB 저장소 , 외부 시스템 연동 회원 저장소)

\*어떤 것을 할지 모를 때 메모리 저장소로 구현한다 .

클래스 다이어그램만으로는 확실 하게 알 수 없기 때문에 객체 다이어그램도 그려야 한다.

\*테스트

Junit – 테스트를 하기위해서

//@Test 어노테이션을 사용한후

//given 어떤 것이 주어졌을 때.

//when 무엇을 했을 때

//then 어떻게 되는 가

여기서는 Assertions.assertThat(member).isEqualTo(findMember); 과 같이 사용한다 .

테스트 코드는 필수다.

@BeforeEach 를 사용할 수 있다. Test 를 실행 하기전

\*TIP

인터페이스 구현체가 한 개 일 때 impl를 붙힌다 .

Enum Type은 == 을 쓴다. Member.getGrade() == Grade.VIP

성공테스트 뿐만 아니라 실패 테스트도 만들어 보아야 한다.

**DIP를 지키기 위해서 인터페이스에만 의존하도록 설계와 코드를 변경하면된다. 하지만 NPE가 발생하게된다 .**

**따라서 누군가 구현객체를 생성 시켜줘야한다 .**

**\***관심사의 분리

AppConfig 등장

구현 객체를 생성하고 , 연결하는 책임을 가지는 별도의 설정 클래스

생성자 주입을 한다.

의존관계에 대한 고민은 외부에 맡기고 실행에만 집중하면 된다. (DIP 완성)