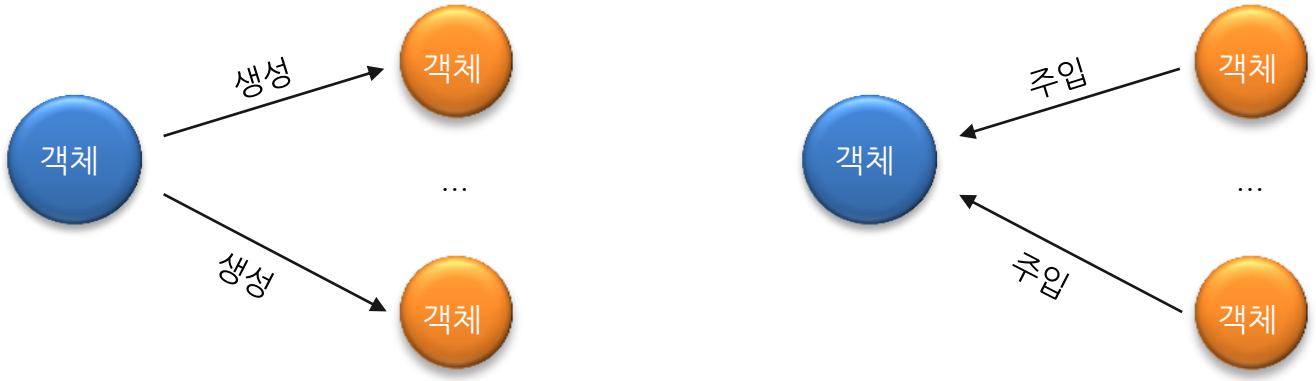
2장 Spring IoC/DI

목차

1. IoC/DI

- IoC Inversion of Control는 객체의 생성과 생명주기를 컨테이너에게 위임하는 형태로 객체의 제어가 바뀜을 의미하는 제어의 역행
- DI Dependency Injection는 의존성 주입으로 컨테이너로 부터 객체를 주입 받는 기법
- IoC/DI를 이용하면 객체 생성과 의존 관계 처리를 컨테이너가 담당하기 때문에 낮은 결합도의 컴포넌트 구현



2. 스프링 컨테이너

- 스프링 컨테이너 Spring Container는 스프링 애플리케이션을 구성하는 빈 Bean을 생성, 관리 및 제공하는 역할
- 스프링 빈 Spring Bean은 스프링 컨테이너로 관리하는 자바 객체로 빈 등록은 xml, Annotaion, 설정 클래스 사용
- ApplicationContext는 빈을 싱글톤으로 관리하는 스프링 컨테이너

주요 어노테이션	설명
@Configuration	설정 클래스 정보를 바탕으로 스프링 컨테이너 생성
@ComponentScan	특정 패키지나 클래스를 기준으로 컴포넌트들을 스캔하여 빈으로 등록
@Bean	설정 클래스에서 사용하는 외부 라이브러리를 Bean으로 등록
@Component	• 사용자 정의 클래스를 Bean으로 등록 • @Service, @Controller, @Repository 등
@Autowired	• 적합한 데이터 타입을 이용해서 의존 객체를 자동으로 주입 • @Resource, @Inject 등

3. 의존성 주입

- 의존성 주입DI, Dependency Injection 객체가 직접 의존하는 객체를 생성하거나 관리하지 않고 외부로부터 주입 받는 디자인 패턴
- 의존성 주입은 객체 지향 프로그래밍에서 코드의 결합도를 낮추고 유연성을 높이는 데 사용
- DI는 객체지향 설계의 중요한 원칙인 SOLID 원칙을 준수하도록 돕는 중요한 메커니즘

의존성 주입 방식	설명
생성자 주입	• 객체를 생성할 때 생성자를 통해 의존성을 주입하는 방식 • 주입할 의존성을 생성자의 매개변수로 전달하여 객체를 생성
세터 주입	• 객체를 생성한 후에 setter 메서드를 통해 의존성을 주입하는 방식 • 주입할 의존성을 객체의 setter 메서드를 호출하여 설정
필드 주입	• 객체의 필드에 직접 의존성을 주입하는 방식 • 필드에 주입할 의존성을 선언