다시 스프링으로…

스프링 이야기에 왜 객체 지향 이야기가 나오는가?

* 스프링은 다음 기술로 **다형성 + OCP, DIP** 를 가능하게 지원
  + DI(Dependency Injection): 의존관계, 의존성 주입
  + DI 컨테이너 제공
* 클라이언트 코드의 변경 없이 기능 확장
* 쉽게 부픔을 교체하듯 개발

스프링이 없던 시절로

* 옛날 어떤 개발자가 좋은 객체 지향을 개발하려고 OCP, DIP 원칙을 지키면서 개발을 해보니 너무 할일이 많았다. 배보다 배꼽이 더크다. 그래서 프레임워크로 만들어버림
* 순수하게 자바로 OCP, DIP 원칙들을 지키면서 개발을 해보면, 결국 스프링 프레임워크를 만들게 된다. (더 정확히는 DI 컨테이너)
* DI 개념은 말로 설명해도 이해가 잘 안된다. 코드로 짜봐야 필요성을 알게 된다!
* 그러면서 이제 스프링이 왜? 만들어졌는지 코드로 보자

정리

* 모든 설계에 **역할**과 **구현**을 분리하자
* 자동차, 공연의 예를 떠올려보자.
* 애플리케이션 설계도 공연을 설계 하듯이 배역만 만들어두고, 배우는 언제든지 **유연**하게 **변경**할 수 있도록 만드는 것이 좋은 객체 지향 설계이다.
* 이상적으로는 모든 설계에 인터페이스를 부여하자

실무적인 고민..

* 하지만 인터페이스를 도입하면 추상화라는 비용이 발생한다.
* 기능을 확장할 가능성이 없다면, 구현 클래스를 직접 사용하고 향후 꼭 필요할 때 리팩토링 해서 인터페이스를 도입하는 것도 방법이다.