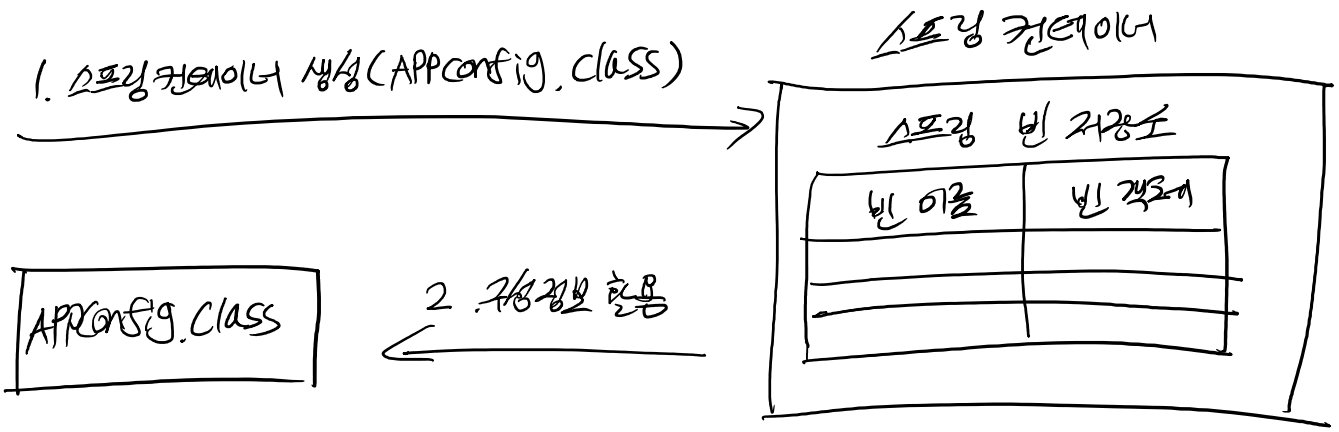
스프링핵심원리-스프링컨테이너와 스프링빈

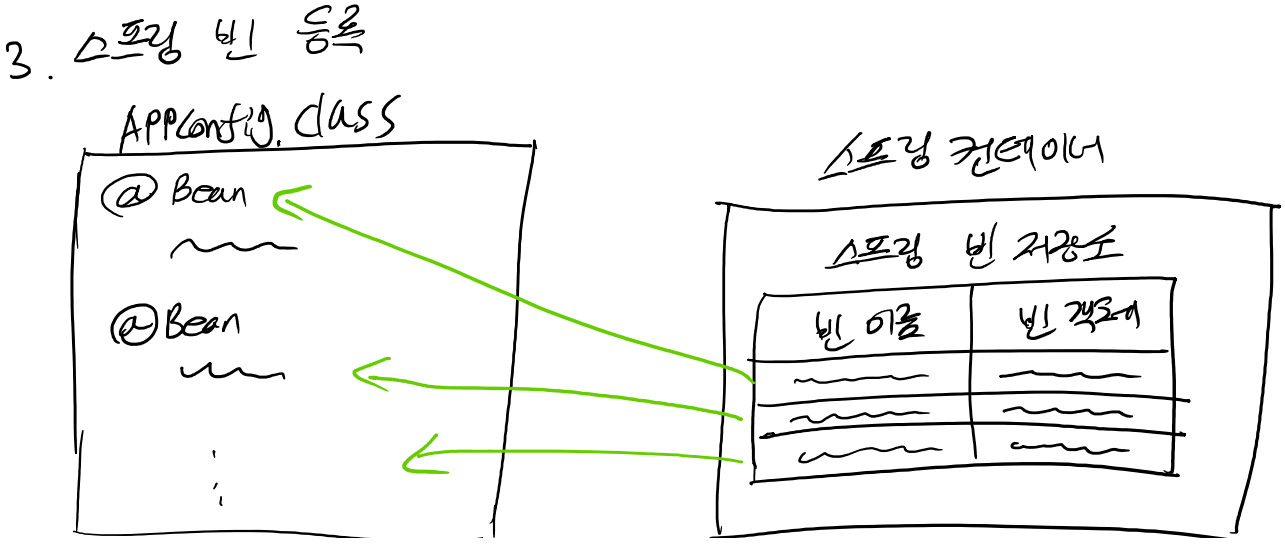
2021년 8월 5일 목요일 오후 4:57

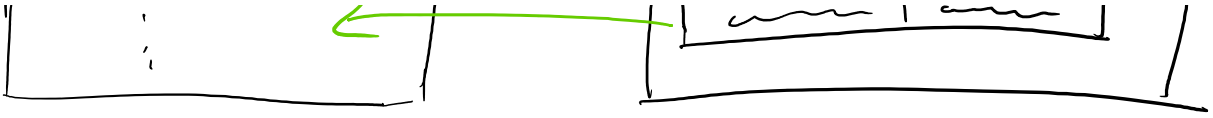
ApplicationContext 를 스프링 컨테이너라고 함
AppConfig 사용해서 직접 객체 생성하고 DI 했지만, 이제는 스프링 컨테이너를 통해서 사용
스프링 컨테이너는 @Configuration 이 붙은 AppConfig 를 설정 정보로 사용
여기서 @Bean 이 붙은 메소드를 모두 호출해서 반환된 객체를 스프링 컨테이너에 등록
등록된 객체를 스프링 빈이라 함
즉, 이전에는 AppConfig 통해서 직접 조회 이제는 스프링 컨테이너 통해서 조회

이러한 중간과정이 붙었는데 이를 통해 얻는 이점은??

ApplicationContext 는 인터페이스
ex) ApplicationContext applicationContext = new AnnotationConfigApplicationContext(AppConfig.class);
스프링 컨테이너는 XML을 기반으로 만들 수 있고, 어노테이션 기반의 자바 설정 클래스로 만들 수 있음 (요즘에 다 이러한 방식을 사용, AppConfig 사용 했던 방식)
스프링 컨테이너 생성시에 구성정보 지정해주어야 함 (여기서는 AppConfig.class))







스프링 컨테이너는 매개변수로 넘어온 설정 클래스 정보를 사용해서 스프링 빈 등록
빈 이름은 항상 다른 이름을 부여해야 한다 (같은 이름 부여시, 겹치거나 덮어지거나 오류발생)

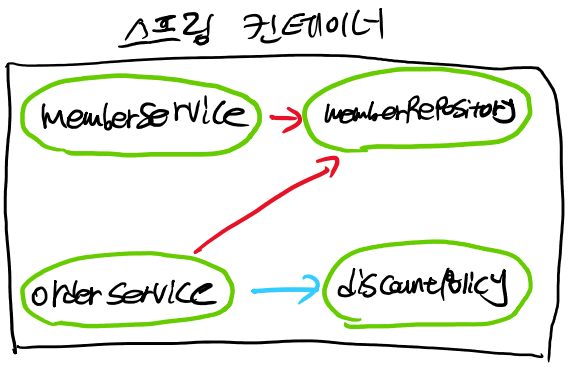
4. 스프링 빈 의존관계 설정

```java
@Bean
public MemberService memberService() { //역할
    return new MemberServiceImpl(memberRepository()); //구현
}

@Bean
public MemberRepository memberRepository() { //역할
    return new MemoryMemberRepository(); //구현
}

@Bean
public DiscountPolicy discountPolicy() { //역할
    //return new FixDiscountPolicy(); //구현
    return new RateDiscountPolicy(); //구현체 갈아끼우기
}

@Bean
public OrderService orderService() {
    return new OrderServiceImpl(memberRepository(), discountPolicy());
}
```



스프링 컨테이너는 설정 정보를 참고해서 의존관계를 주입(DI) 한다.
단순히 자바 코드를 호출하는 것과 다르다. 차이??
=>?

스프링은 빈을 생성하고 의존관계를 주입하는 단계가 나누어져 있음
(실제로는 생성자를 호출하면서 의존관계도 한번에 처리해버림)

//RoleROLE_APPLICATION:직접등록한어플리케이션빈
//RoleROLE_INFRASTRUCTURE:스프링이내부에서사용하는빈

부모타입으로 조회하면 자식타입도 함께 조회한다
고로 모든 자바 객체의 최고 부모인 Object 로 조회하면 전체 조회됨