스프링 컨테이너가 생성 되어서 스프링 컨테이너가 종료될 때까지 스프링 빈이 유지된다고 학습했는데, 이는 스프링 빈이 기본적으로 싱글톤 스코프로 생성되기 때문이다.

하지만, 스프링 스코프에는 다양한 스포크가 존재함.

1. 싱글톤 : 기본적인 스코프, 스프링 컨테이너의 시작과 종료까지 유지되고, 관리하는 가장 넓은 범위의 스코프

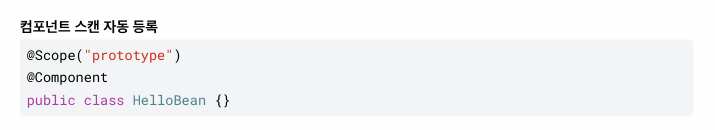
2. 프로토 타입 : 객체의 생성, 의존관계 주입, 초기화 메소드까지 실행되지만, 종료메소드는 실행안 해줌.

3. 웹 관련 스코프 (스프링 웹과 관련된 기능이 들어가야 쓸 수 있음.)

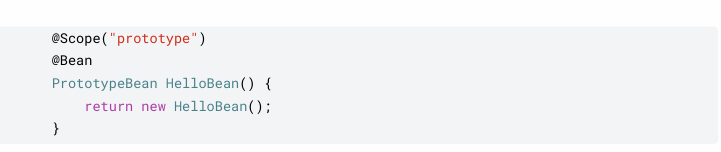
- request : 웹 요청이 들어오고 나갈 때까지 유지되는 스코프

- sessions : 웹 세션이 생성되고 종료될 때 까지 유지되는 스코프

- application : 웹의 서블릿 컨텍스트와 같은 범위로 유지되는 스코프



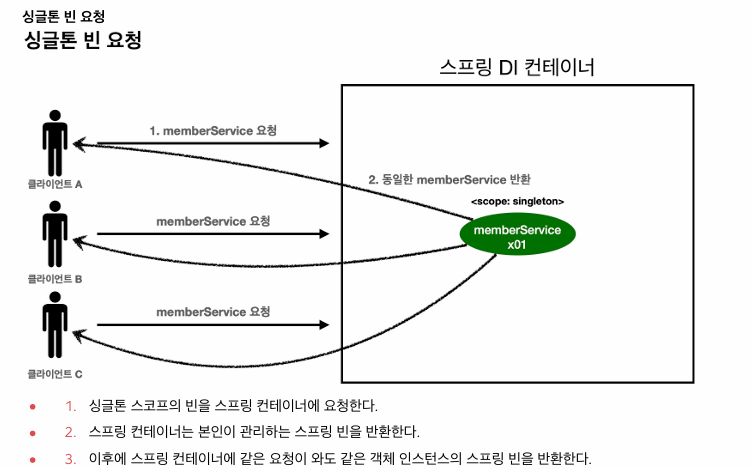
빈스코프 수동 등록

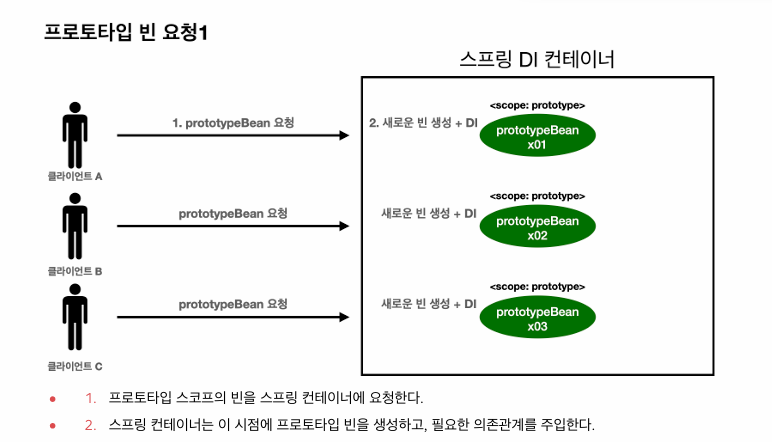


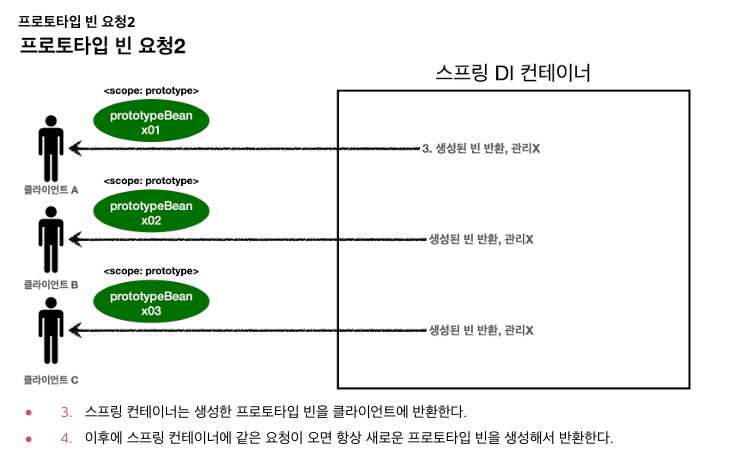
**프로토타입 스코프**

싱글톤 스코프의 빈 조회시 : 항상 같은 인스턴스의 빈을 반환한다.

프로토타입 스코프의 빈 조회시 : 항상 새로운 인스턴스를 생성해서 반환한다.



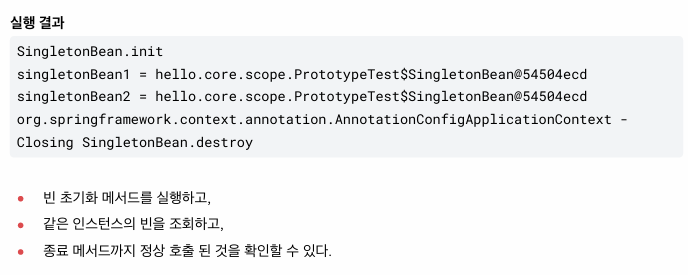




정리

프로토타입 스코프의 스프링 컨테이너는 빈을 생성하고, 의존관계 주입, 초기화까지만 처리한다. 매번 새로운 빈을 계속 생성하고, 반환한다음 프로토타입 빈을 관리하지 않는다. 빈의 종료 메소드는 클라이언트가 호출해야한다. @PreDestroy 같은 종료 메소드는 호출되지 않음.



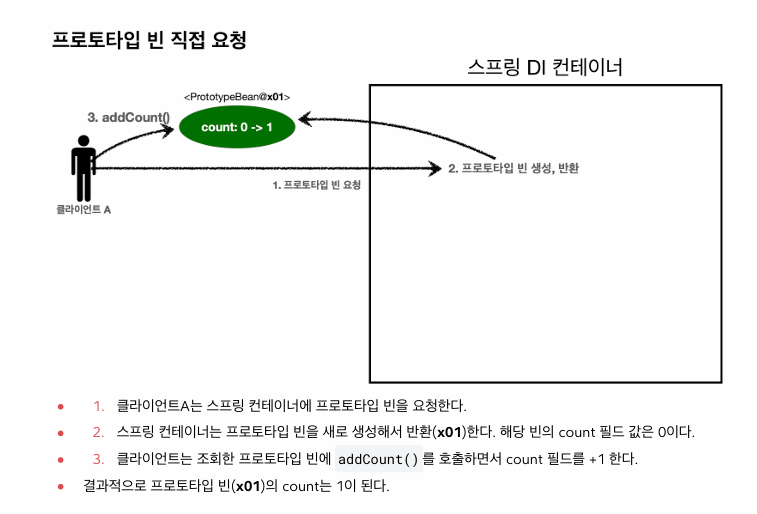


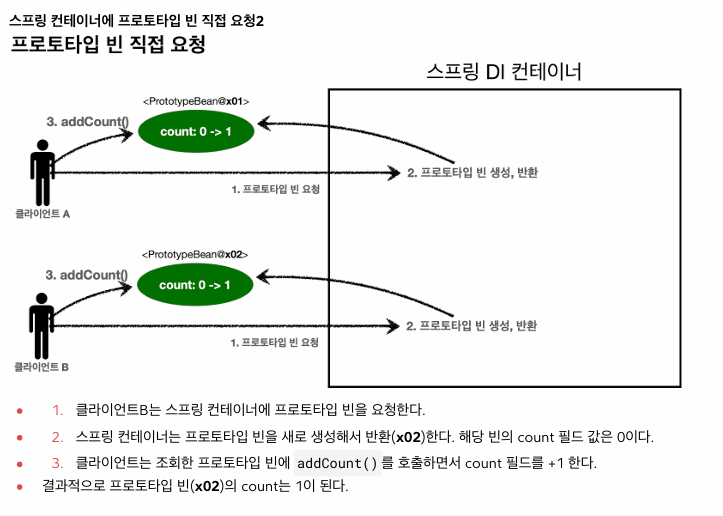




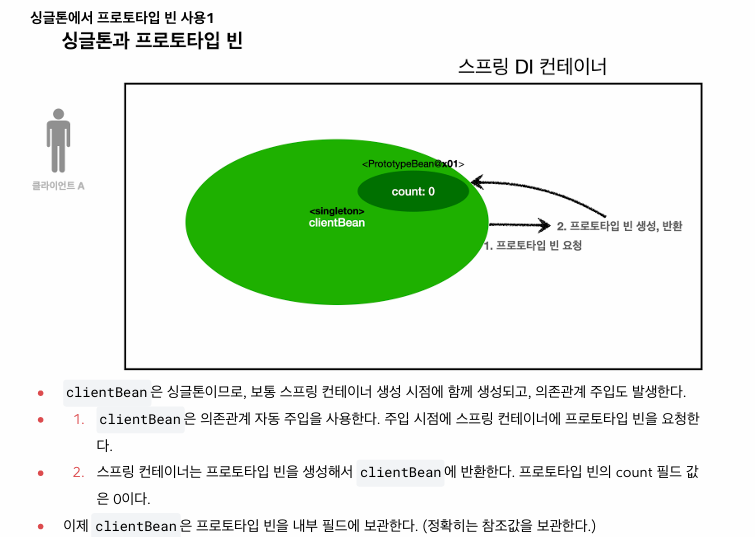
**프로토타입 스코프와 싱글톤 빈을 함꼐 사용할 때 문제점.**

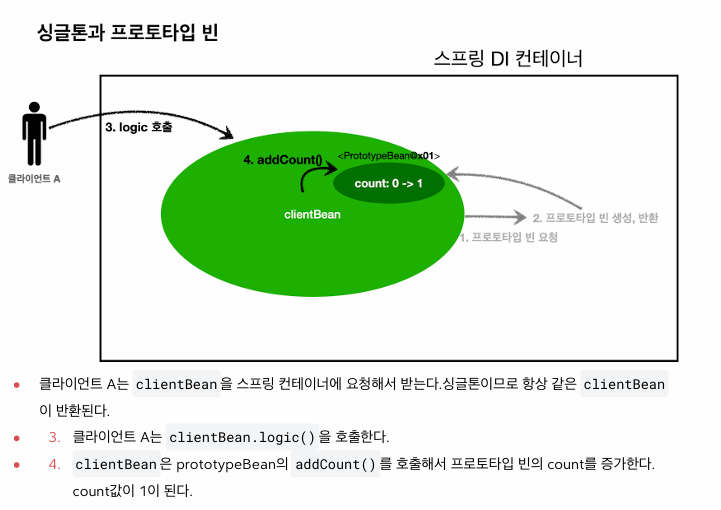
프로토타입은 요청할 때 마다 항상 새로운 객체 인스턴스를 생성해서 반환하지만, 싱글톤 스코프와 같이 사용하면 의도대로 동작하지 않을 수 있다.

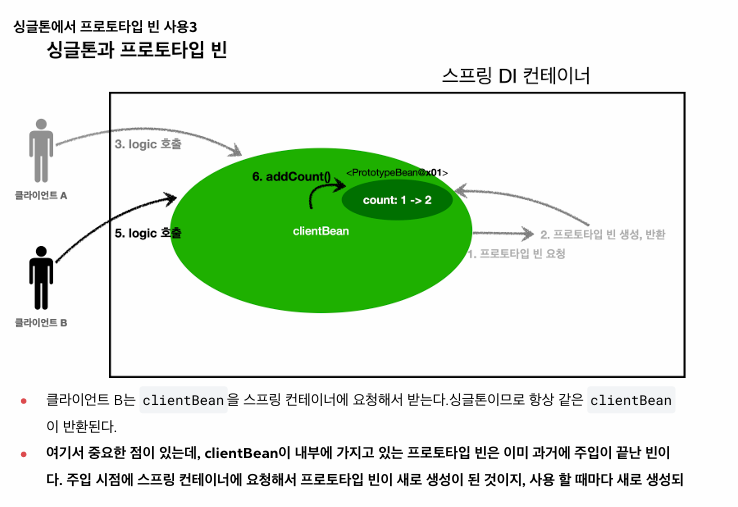


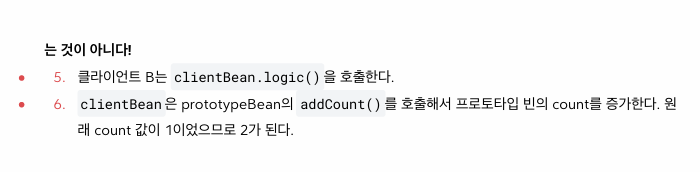


싱글톤 스코프의 빈에서 프로토타입 빈을 주입해서 사용하는 경우 살펴봄.



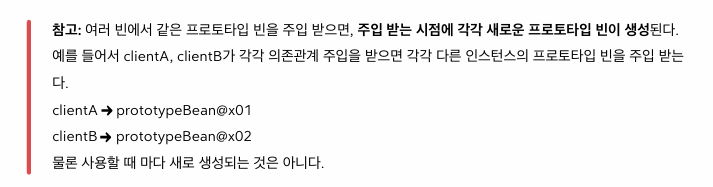






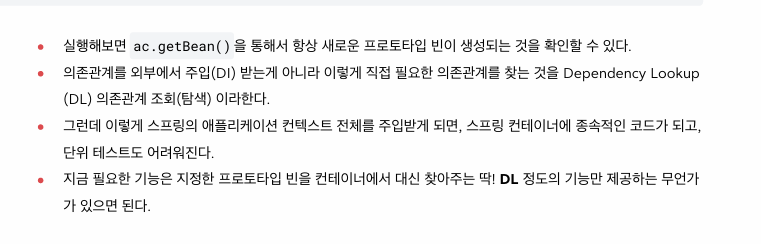
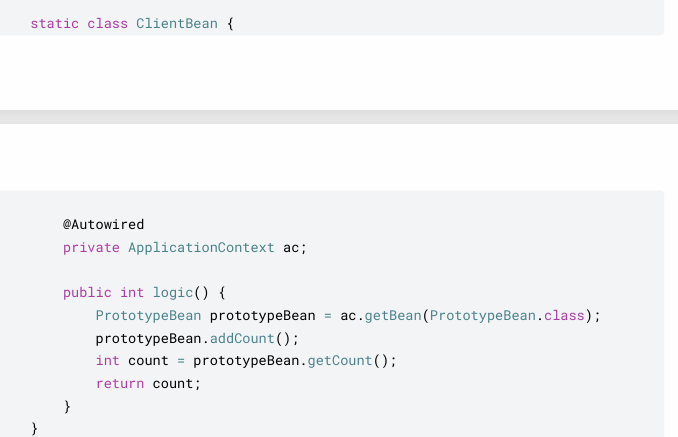
싱글톤 스코프 빈 안에 주입되어있는 프로토타입 빈은 처음 주입될 때만 생성되고, 그 이후에 logic을 이용해서 빈이 사용될 때마다 새로 생성되는 것이 아니다.

만약 의도가, 프로토타입을 사용할 때마다 새로 생성되게 하고자 하면, 다른 처리 방식이 필요하다.



**프로토타입 스코프 - 싱글톤 빈과 함꼐 사용 시 Provider로 문제 해결**

가장 간단한 방법 : 싱글톤 빈이 프로토타입을 사용할 때마다 스프링 컨테이너에 새로 요청하는 것. -> 무식한 방법이다.



Applicationcontext 는 별의 별 기능을 다 제공해주는데, 싱글톤에서 프로토타입 빈 사용 시 필요한 기능은 컨테이너에 등록되어있는 프로토타입 빈만 찿아주는 기능만 있으면 된다.

이를 해주는 것이 ObectProvider이다.

ObjectProvider<PrototypeBean> prototypeBeanBrovider;

PrototypeBeanProvider.getObject(); 를 통해 항상 새로운 프로토타입 빈이 생성된다.

이전에는 ObjectFactory가 있었는데, ObjectFactory에 기능을 추가해서 ObjectProvider를 제공해줌.

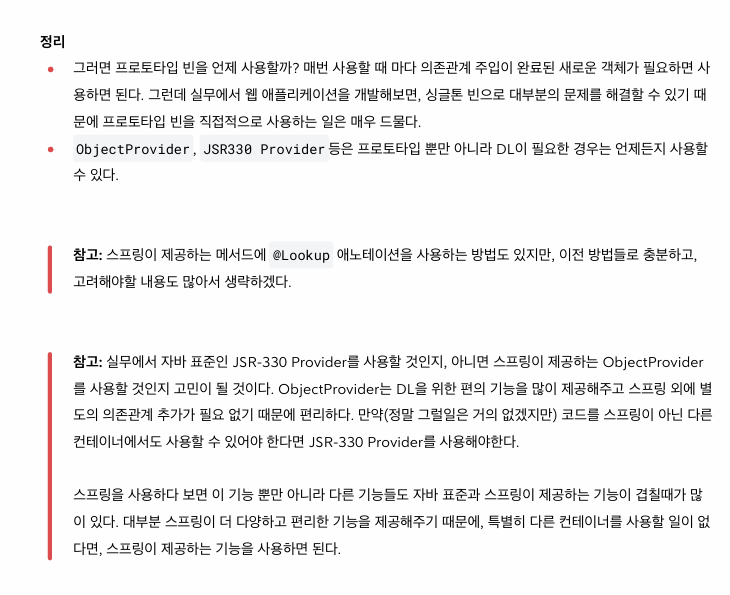
ObjectProvider는 스프링이 제공하는 기능이지만. 기능이 단순하므로 단위테스트를 만들거나 mock 코드를 만들기는 훨씬 쉽다.

ObjectProvider는 스프링에 의존적이다.

스프링에 의존적이지 않은 순수 자바 코드를 사용하기 위해서는 JSR-330 Provider를 사용하면 된다. JSR(자바 표준임)

Provider도 ObjectProvider와 마찬가지로 스프링컨테이너를 통해 해당 빈을 찾아 반환한다. 메소드는 get을 사용한다. 자바 표준이라 스프링에 종속적이지 않다.

get() 메소드 하나로 기능이 매우 단순하지만, 별도의 라이브러리를 땡겨서 사용해야한다. 자바 표준이므로 다른 컨테이너에서도 사용할 수 있다.



컨테이너 관련 기술은 자바 표준에서도 만들었지만, 스프링이 워낙 잘되어있고, 자바 표준은 불편하기 때문에 사람들이 잘 안쓴다. 기능이 편리하면, 스프링 것을 씀(Autowired). 기능이 비슷하거나 스프링에서 자바 표준을 쓰라고 권장하면 자바 표준을 씀.(PostConstructor, preDestory)

**웹 스코프**

웹스코프 :

웹환경에서만 동작한다.

프로토 타입과 다르게 스프링이 해당 스코프의 종료시점 까지 관리해서 종료메소드가 호출된다.

웹 스코프의 종류

1. Request : Http 요청 하나가 들어오고 나갈 때까지 유지되는 스코프. 각각의 http 요청마다 별도의 빈 인스턴스가 생성되고 관리된다.

2. Session : Http Session과 동일한 생명주기를 가짐.

3. Application : 서블릿 컨텍스트와 동일한 생명주기를 가짐.

4. Websocket : 웹소켓과 동일한 생명주기를 가짐.

