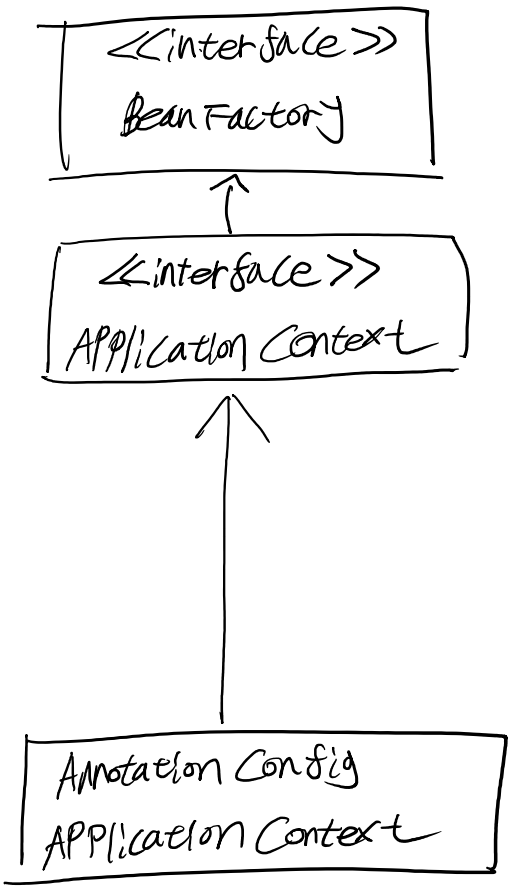
스프링핵심원리-스프링컨테이너와 스프링빈2

2021년 8월 6일 금요일 오후 4:37

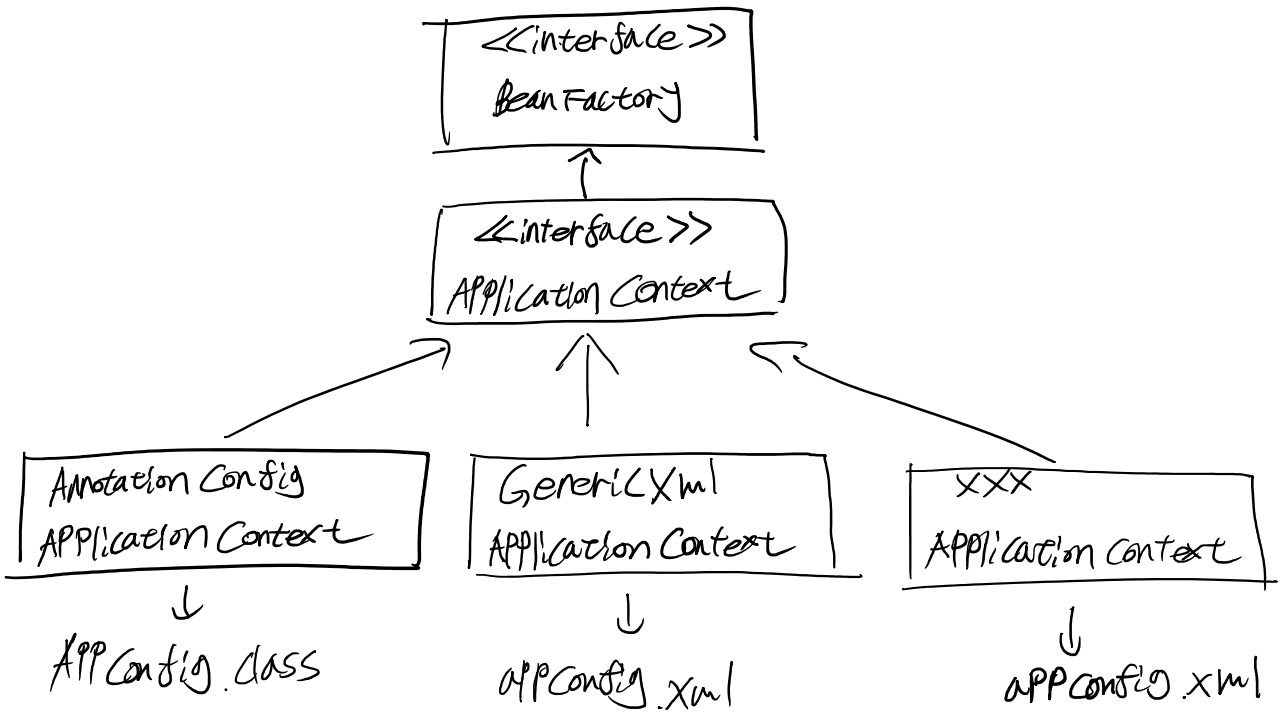


BeanFactory : 스프링 컨테이너의 최상위 인터페이스
스프링빈 관리 조회 역할
getBean() 제공 등 우리가 사용했던 대부분의 기능 제공

ApplicationContext : BeanFactory 기능 모두 상속받아서 제공
이놈의 존재의의는? 수많은 부가기능 제공 => 주로 사용함
<<interface>> MessageSource : 메시지소스를 활용한 국제화 기능
<<interface>> EnvironmentCapable : 환경변수
- 로컬,개발,운영 구분해서 처리

<<interface>> ApplicationEventPublisher : 어플리케이션 이벤트
- 이벤트 발행하고 구독하는 모델 편리하게 지원

<<interface>> ResourceLoader : 편리한 리소스 조회
- 파일,클래스패스,외부 등에서 리소스 편리하게 조회



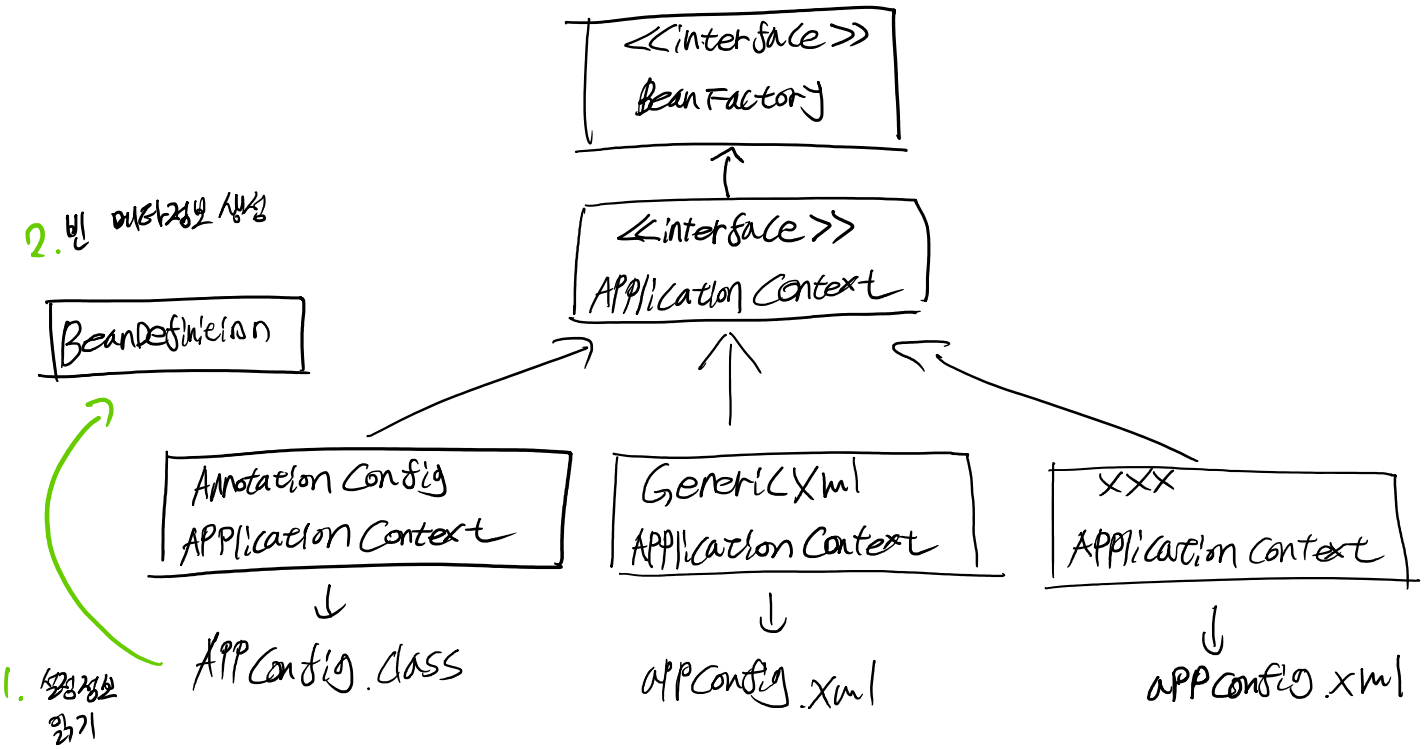
어노테이션 기반 자바코드 사용
지금까지 했던 것

new AnnotationConfigApplicationContext(AppConfig.class)

XML 설정 사용
최근 트랜드상 스프링 부트를 많이 사용하면서 잘 사용하지 않지만,
XML을 사용하면 컴파일 없이 빈 설정 정보를 변경할 수 있는 장점도 존재
GenericApplicationContext 를 사용하면서 xml 설정 파일을 넘기면 된다

스프링 빈 설정 메타 정보 - BeanDefinition

BeanDefinition 이라는 추상화를 통해서 다양한 설정 형식 지원
역활과 구현 개념적으로 나눈 것
위에서 한 방법들(자바코드,XML 등)을 통해 BeanDefinition 만들고
스프링 컨테이너가 BeanDefinition (빈 설정 메타정보)을 기반으로 스프링 빈 생성



BeanDefinition 정보

- BeanClassName : 생성할 빈의 클래스명
- factoryBeanName : 팩토리 역할의 빈을 사용할 경우 이름
- factoryMethodName : 빈을 생성할 팩토리 메소드 지정
- Scope : 싱글톤
- lazyInit : 스프링 컨테이너를 생성할 때 빈을 생성하는 것이 아니라 실제 빈을 사용할 때까지 최대한 생성을 지연 처리하는지 여부
- InitMethodName : 빈을 생성하고, 의존관계를 적용한 뒤에 호출되는 초기화 메소드 명

- DestroyMethodName : 빈의 생명주기가 끝나서 제거하기 직전에 호출되는 메서드 명
- Constructor argument, Properties : 의존관계 주입에서 사용한다.