스프링 컨테이너와 스프링 빈

타입으로 조회 시 같은 타입의 스프링 빈이 둘 이상이면 오류 발생.

맞아요! **(4)**에서 ac.getBeansOfType(MemberRepository.class)를 사용하면, 같은 타입의 모든 빈을 Map 형태로 반환하기 때문에 오류가 발생하지 않습니다. 이 방식은 이름과 객체를 모두 포함해서 보여주니까요.

그러나 **(3)**처럼 ac.getBean(MemberRepository.class)로 특정 객체를 찾으려는 경우에는 오류가 발생할 수 있습니다. 왜냐하면:

타입만 보고 찾는 경우:

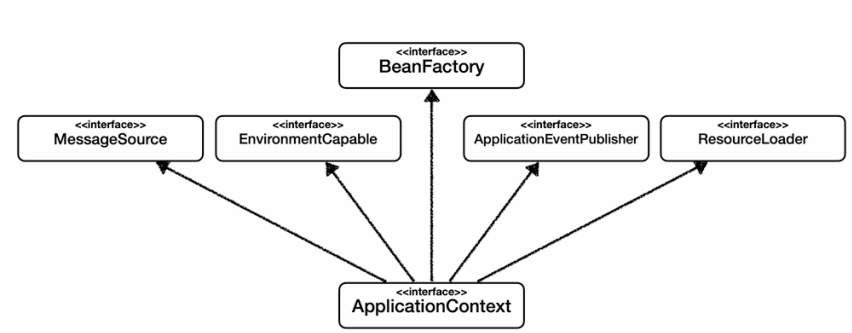
MemberRepository라는 타입의 빈이 2개 (memberRepository1, memberRepository2) 있으므로, 스프링이 어떤 것을 반환해야 할지 알 수 없습니다.
그래서 NoUniqueBeanDefinitionException 오류가 발생합니다.
특정 객체를 찾으려면 이름 지정 필수:

ac.getBean("memberRepository1", MemberRepository.class)처럼 이름과 타입을 동시에 지정하면 정확히 그 객체를 찾을 수 있습니다.
정리하면:
(4) Map 조회 방식: 이름을 포함해서 반환하므로 중복된 타입이 있어도 문제가 없음.
(3) 단일 조회 방식: 이름 없이 타입만으로 조회하면, 중복된 타입 때문에 오류 발생.
즉, (4)에서 얻은 Map의 키(key)를 이용해서 필요한 객체를 지정하면 오류 없이 사용할 수 있습니다. 😄

스프링 빈 조회 - 상속 관계

부모 타입으로 조회하면, 자식 타입도 함께 조회됨

BeanFactory와 ApplicationContext



- **메시지소스를 활용한 국제화 기능**
  - 예를 들어서 한국에서 들어오면 한국어로, 영어권에서 들어오면 영어로 출력
- **환경변수**
  - 로컬, 개발, 운영등을 구분해서 처리
- **애플리케이션 이벤트**
  - 이벤트를 발행하고 구독하는 모델을 편리하게 지원
- **편리한 리소스 조회**
  - 파일, 클래스패스, 외부 등에서 리소스를 편리하게 조회

스프링 빈 설정 메타 정보 - BeanDefinition

- 스프링 컨테이너는 자바코드인지, XML인지 몰라도 됨. 오직 BeanDefinition만 알면 됨.
- BeanDefinition = 빈 설정 메타 정보
- @Bean, <bean> 당 각각 하나씩 메타 정보가 생성됨.
- 스프링 컨테이너는 이 메타 정보를 기반으로 스프링 빈을 생성.

ApplicationContext 는 getBeanDefinition을 못함

AppConfig - factory method를 통해 등록하는 방법