스프링 빈

* 스프링 Ioc컨테이너가 관리하는 객체
* IOC 컨테이너 : inversion of control (제어 반전)
* ApplicationContext 인터페이스 : spring ioc 컨테이너를 대표하며, 빈을 인스턴스화 하고 모은다.

빈이 필요한 이유

* 객체를 빈으로 등록하지 않고, 의존성 주입을 해준다면, new라는 키워드를 이용해서 객체를 생성해야 한다.
* 만약 생성자의 구현체가 바뀌게 된다면, 생성자에게 구현체를 선택해야하는 책임이 있기에 변경을 해줘야만 한다.
* 만약 의존성을 주입할 때, 여러 개의 의존성을 주입하는 것에 대한 순서를 알아야 한다.
* 만약 하나의 어플을 만들 때, 이 범위가 늘어나게 된다면, 의존성을 주입해야 할 객체가 계속 늘어나야 하므로, 이를 일일이 지정해주면 매우 복잡한 식이 만들어진다.
* 따라서 이를 빈에 등록하여 IOC컨테이너가 관리하도록 한다. 이를 통해, 주입된 의존성을 사용하는 것에만 집중하면 된다.

빈 = 싱글톤

* 일반적인 객체에 싱글톤을 주입한다면, 접근 지정자 Private를 이용해야 하므로, 결국 다형성을 이용할 수 없다.
* 따라서, 상속을 할 수 없다
* 싱글턴으로 구현하면, 객체= 공유객체가 되므로, 단위 테스트를 할 수 없다.

Ioc컨테이너의 빈 관리 방법

* 빈에서 싱글톤 객체를 생성
* 이 때, 빈으로 등록하기 위하여 어노테이션, java설정 클래스, xml을 기반으로한 configuration 메타데이터를 이용한 bean definition생성
* 빈으로 등록할 pojo(오래된 자바 객체)와, 빈 def정보를 기반으로 빈을 생성한다.
* 이 때, 싱글톤 패턴이 아닌, 일반 클래스를 이용하여 객체를 생성한다.
* 이를 레지스트리에 등록하면, 의존성 자동주입을 하게 된다.
* 커넥션 풀과 같이 초기화가 필요한 객체들은 초기화를 하게 되고, 이 과정이 완료되면 비로소 빈을 사용할 수 있게 된다.
* 스프링 컨테이너가 종료될 때, 빈 스코프가 싱글톤인 객체들도 소멸하게 된다.

주의점

* 빈이 어떤 value 를 가질때, 이를 변화시키는 요소(스레드1)가 있다고 해보자, 그럼 이를 가져와서 사용하는 스레드2번은 계속하여 값이 변화하게 된다. 따라서 빈의 상태를 예측할 수 없기에, 우리가 필요한 결과를 도출할 수 없다. 따라서 빈스코프를 싱글톤으로 설정할 경우 상태를 가지면 안댐!!!!!!!!
* 상태를 가지는 빈은 존재할수 없는가? : 존재할 수 있다. (이를 빈 스코프라고 한다). 빈은 생성되고, 존재하고 적용되는 범위를 지정할 수 있다.
* 이를 스코프 어노테이션을 사용하여 지정할 수 있다.
* 이때, 스코프 어노테이션을 프로토 타입으로 설정한다면, ioc컨테이너와 함께 생성되고 소멸하는 것이 아니라, 요청이 올 때마다 객체가 생성되게 된다.
* 이는 모든 스레드에서 공유하지 않으므로, 상태를 가져도 무방하다.
* 의존성을 자동으로 주입해야 할 인터페이스에 구현체가 2개 이상이면, 이를 주입하는 과정에서 충돌이 일어나 결국 에러가 발생함.
* 이를 해결하기 위해서 어노테이션을 이용하게 된다!
* @primary, @qualifire(“빈의 이름”)을 이용하여 특정 상황에서 어떤 빈이 우선순위인지, 어떤 빈을 사용해야하는지 특정지어 사용할 수 있다.

어노테이션

* 자바를 이용하여 어플을 만들 때, 우리가 해야할 초기화 방법은 2가지가 있다.
* 1. Xml을 이용하여 외부파일에 설정 정보를 저장함.
* 2. 코드파일을 직접 심어서 저장함(이때, annotation이 사용됨)
* Xml을 이용하면 일일이 파일을 열어서 직접 바꿔야했으나,

어노테이션을 설정하면, 객체를 변화했을 시에, 이 어노테이션을 가지고 있는 객체가 하나밖에 없으므로 자동적으로 코드를 연결해준다.

\*모듈을 변경하는 방법이 매우 쉬워짐

* Xml파일 내의 프로퍼티를 따로 설정하지 않아도 @autowired 를 이용하면 이를 설정해줌.
* 객체 생성 방법을 더욱더 간단하게 만들어줌

결론적으로 이는 의존성 주입과 관련된 어노테이션, 개념들이다.

DI(dependency injection) : 의존성 주입