Spring DI - XML

▶ Spring XML 방식

KH정보교육원

Spring 컨테이너 구동 시 spring의 환경 설정 관련된 xml파일을 불러오는데
이 파일에 bean, aop, transaction 등 여러 사항을 다 작성해 구동하는 방식

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
        http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.2.xsd">

    <!-- controllerMapping -->

    <bean class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
        <property name="mappings">
            <props>
                <prop key="/login.do">login</prop>
                <prop key="/board.do">board</prop>
            </props>
        </property>
    </bean>
    <bean id="login" class="com.kh.mvc2.user.controller.LoginController"></bean>
    <bean id="board" class="com.kh.mvc2.board.controller.BoardController"></bean>

    <!-- viewResolver -->

    <bean id="viewResolver"
        class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix" value="/WEB-INF/board/"></property>
        <property name="suffix" value=".jsp"></property>
    </bean>
```

▶ Spring XML 방식

✓ 장점 및 단점

장점	- 생성되는 모든 Bean을 XML에서 확인 가능 - 프로젝트를 운영하는 입장에서 관리의 편의성이 높음
단점	- Bean의 개수가 많아지면 XML파일을 관리하기 어려움 - 여러 개발자가 같은 설정 파일 수정 시 설정에 충돌 발생 가능 - DI에 필요한 적절한 setter메소드 또는 생성자가 코드 내부에 반드시 존재해야 함

▶ Spring XML 기본 설정

KH정보교육원

✓ XML 구조

<beans>태그를 최상위 태그로 하여 <beans>태그 안에
다양한 태그로 값 설정

✓ XML 주요 태그

태그 명	내용	속성
<beans>	xml파일의 최상위 태그로 여러 가지 namespace 설정	namespace, p, c, aop, context, tx, mvc 등
<bean>	스프링에서 사용할 POJO객체를 등록할 때 사용	id, class, scope
<import>	설정 파일을 불러오는 태그	resource

* namespace : 설정을 간편하게 도와주는 기능

▶ Spring XML 태그

✓ <bean>

POJO객체를 컨테이너에 등록해 컨테이너가 사용할 수 있게 하는 태그

<bean id | name="명칭" class="클래스 풀네임" [기타 옵션]/>

✓ 기본 속성

속성 명	내용
id [="String"]	객체 생성 시 사용하는 변수와 비슷 명명 규칙 : 낙타 표기법 사용, 자바 식별자 작성 규칙 적용
name ["String"]	자바 식별자 작성 규칙을 따르지 않으며 특수 기호 포함 시 사용
class ["클래스 풀네임"]	POJO객체를 지정하며 패키지를 포함한 클래스 명 작성

▶ Spring XML 태그

✓ <bean> 기타 속성

속성 명	내용
init-method[="메소드 명"]	객체 생성 후 초기화 하거나 실행되어야 할 기능이 있는 경우
destroy-method[="메소드 명"]	객체 삭제 전에 실행되어야 할 기능이 있는 경우
lazy-init["true | false"]	객체가 즉시 로딩되지 않고 사용 시 로딩(true)
scope["설정 값"]	객체 생성 방식 설정

* scope 설정 값
  - singleton : spring컨테이너 내에 단 하나의 객체 생성(default)
  - prototype : 다수의 객체 존재 가능

▶ <bean> 내부 태그

KH정보교육원

✓ <constructor-arg>

<bean>으로 지정된 객체가 생성될 때 default 생성자가 아닌
매개변수가 있는 생성자로 생성하여 초기 값 대입
(단, 일치하는 매개변수가 있는 생성자가 있어야 하며 객체의 생성자와
매핑 되어 변수 선언 순서대로 대입)

```
<bean id | name="명칭" class="클래스 명 풀네임" [기타 옵션]>
        <constructor-arg value | ref="값 | bean id 명"/>
        <constructor-arg index="숫자" value | ref ="값 | bean id 명"/>
        // index로 설정 가능
</bean>
```

▶ <bean> 내부 태그

KH정보교육원

✓ <property>

<bean>으로 지정된 객체의 필드나 객체의 필드로 있는 collection에 값을 대입할 때 사용

일치하는 setter 메소드가 있을 때 객체의 setter 메소드와 매핑되어 초기 값 대입 (단, 명칭이 일치해야 함)

```
<bean id | name="명칭" class="클래스 명 풀네임" [기타 옵션]>
        <property name="명칭" value="값" /> // 변수가 기본인 경우
        <property name="명칭" ref="bean id 명"/> // 변수가 객체인 경우
</bean>
```