Spring DI - XML

▶ Spring XML 방식

✓ XML 방식

Spring 컨테이너 구동 시 **spring의 환경 설정 관련된 xml 파일**을 불러오는데,
이 파일에 **bean, aop, transaction 등 여러 사항을 다 작성하여 구동하는 방식**이다.

```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 기본 설정

✓ XML 구조

<beans> 태그를 최상위 태그로 하여 <beans> 태그 안에 다양한 태그로 값을 설정할 수 있다.

✓ XML 주요 태그

태그명	내 용	속 성
<beans>	xml 파일의 최상위 태그로 여러가지 namespace를 설정	namespace : p, c, aop, context, tx, mvc 등
<bean>	스프링에서 사용할 POJO 객체를 등록할 때 사용	Id, class, scope
<import>	설정 파일을 불러오는 태그	resource

* namespace : 설정을 간편하게 도와주는 기능

▶ Spring XML 태그 - <bean>

✓ <bean>

POJO 객체를 컨테이너에 등록하여 컨테이너가 사용할 수 있게 만드는 태그

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

✓ 기본 속성

속성명	내 용
id [="String"]	객체 생성 시 사용하는 변수라고 보면된다. 명명 규칙 : 낙타 표기법 사용 / 자바 식별자 작성 규칙 적용
name [="String"]	자바 식별자 작성 규칙을 따르지 않음 / 특수 기호 포함 시 사용
class [="클래스 풀네임"]	POJO객체를 지정 / 패키지를 포함한 클래스명 작성

▶ Spring XML 태그 - <bean>

✓ <bean> 기타 속성

속성명	내 용
init-method [="메소드명"]	객체 생성 후 초기화 하거나 실행되어야 할 기능이 있는 경우
destroy-method [="메소드명"]	객체 삭제되기 전에 실행되어야 할 기능이 있는 경우
lazy-init [="true | false"]	객체가 즉시 로딩되지 않고 사용시 로딩 선택(true)
scope [="설정값"]	객체 생성 방식을 지정

*** scope 설정 값**

- singleton : spring 컨테이너 내에 단 하나의 객체만 생성 (default)
- prototype : 다수의 객체가 존재할 수 있음

▶ <bean> 내부 태그

✓ <constructor-arg>

<bean>으로 지정된 객체가 생성 될 때 default 생성자가 아닌 매개변수가 있는 생성자로 생성하여 초기값을 대입 **(단, 일치하는 매개변수가 있는 생성자가 있어야 된다.)**

 # 객체의 생성자와 매핑된다. 변수 선언 순서대로 대입

작성 :

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

▶ <bean> 내부 태그

✓ <property>

<bean>으로 지정된 객체의 필드에 값을 대입할 때 사용

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

초기값을 대입 **(단, 일치하는 setter 메소드가 있어야 된다.)**

 # 객체의 setter 메소드와 매핑된다. (명칭이 일치 해야된다.)

작성 :

```
<bean id | name="명칭" class="클래스명 풀네임 "  [기타옵션]>
        <property name="명칭" value="값" />          // 변수가 기본인 경우
        <property name="명칭" ref="bean id명" />     // 변수가 객체인 경우
</bean>
```

▶ Spring XML 방식 장/단점

✓ XML 단독 사용

모든 Bean을 명시적으로 XML에 등록하는 방법

장점	- 생성되는 모든 Bean을 XML에서 확인 할 수 있다. - 프로젝트를 운영하는 입장에서 관리의 편의성이 높다.
단점	- Bean의 개수가 많아지면 XML 파일을 관리하기 어렵다. - 여러 개발자가 같은 설정 파일을 수정하면 설정에 충돌이 발생 할 수 있다. - DI에 필요한 적절한 setter 메소드 또는 생성자가 코드 내부에 반드시 존재해야 된다.