SPRING DI와 객체관리

백성애

1. BEANFACTORY 인터페이스

- 스프링 컨테이너는 빈 객체를 저장하고 있으며, 각 객체간의 의존 관계를 관리해준다.
- BeanFactory와 ApplicationContext가 스프링 컨테이너 역할을 수행하는 인터페이스.

- 1. BEANFACTORY 인터페이스
- 기본적인 의존성 주입을 지원하는 가장 간단한 형태의 Container
- 구현 클래스:org.springframework.beans.factory.xml.

XmlBeanFactory클래스

1. BEANFACTORY 인터페이스

[예제]

```
Resources res=new FileSystemResource("beans.xml");
XmlBeanFactory factory=new XmlBeanFactory(res);
MessageService
mservice=(MessageService)factory.getBean("msgService");
```

Resource 구현 클래스를 통해 다양한 종류의 자원을 동일한 방식으로 표현하도록 하고 있다.
이 리소스를 이용해XmlBeanFactory에 설정정보를 전달 한다.
그런 뒤 getBean()메소드를 이용해 빈을 가져온다.

1_2. RESOURCE구현 클래스

클래스	설명
org.springframework.core.io. **FileSystemResource**	파일 시스템의 특정 파일로부터 정보를 읽어옴
org.springframework.core.io. **InputStreamResource**	InputStream으로부터 정보를 읽어옴
org.springframework.core.io. **ClasspathResource**	클래스패스에 있는 자원으로부터 정보를 읽어옴
org.springframework.core.io. **UrlResource**	특정 URL로부터 정보를 읽어옴
org.springframework.web,context.support. **ServiceContextResource**	웹 애플리케이션의 루트 디렉터리를 기준으로 지정한 경로에 위치한 자원으로부터 정보를 읽어옴.

2. APPLICATIONCONTEXT 인터페이스

- org.springframework.context.ApplicationContext 인터페이스는 BeanFactory 를 상속받은 하위 인터페이스.
- 스프링은 이를 사용해 의존 관계를 주입한다.
- ApplicationContext는 컨텍스트에 등록된 빈을 관리하는 역할을 한다.
- 또한 AOP를 적용하는 정교한 작업도 처리

APPLICATIONCONTEXT의 주요 역할

- 빈 관리 기능,
- 빈 객체 라이프사이클,
- 파일 같은 자원 처리 추상화,
- 메시지 , 국제화, 이벤트 지원,
- xml스키마 확장을 통한 편리한 설정

  등 추가적인 기능을 제공.

APPLICATIONCONTEXT 구현 클래스

[1] org.springframework.context.support.
ClasspathXmlApplicationContext
: 클래스패스에 위치한 xml파일로 부터 설정 정보를 로드

[2] ...FileSystemXmlApplicationContext
: 파일시스템에 위치한 xml파일로 부터 설정 정보 로드

[3]...AnnotationConfigApplicationContext:
클래스패스에 위치한 자바로부터 설정 정보를 로드

3. WEBAPPLICATIONCONTEXT 인터페이스

- org.springframework.web.context.WebApplicationContext인터페이스는 웹 어플리케이션을 위한 ApplicationContext다.
- 하나의 웹어플(즉, 하나의ServletContext) 마다 한 개 이상의 WebApplicationContext를 가질 수 있다.

WEBAPPLICATIONCONTEX 구현 클래스

[1] org.springframework.context.support.

XmlWebApplicationContext:

: 웹어플리케이션이 위치한 xml파일로 부터 설정 정보를 로드

- [2].. AnnotaionConfigWebApplicationContext:
- 어노테이션을 이용한 설정을 사용할 때

APPLICATIONCONTEXT계층관계

- ApplicationContext는 계층관계를 가질 수 있다.
- 이 때 부모 컨텍스트는 하나만 가질 수 있지만, 자식 컨텍스트의 수에는 제한이 없음

- 자식 컨텍스트는 부모에 등록된 빈에 접근할 수 있으나, 부모는 자식 컨텍스트에 등록된 빈에접근 불가

- 컨텍스트 계층기능을 사용하면 애플리케이션 빈(서비스,리포지터리,인프라)과 웹에서 사용하는 빈(핸들러와 뷰)을 따로 분리할 수 있다.
- (예를 들어 빈이 여러 서블릿에서 쓰이는 경우 각 서블릿 마다 이 빈을 만들지 않고 이미 등록된 빈을 가져다 쓰면됨)

리소스 접근시 사용하는 PREFIX

- ApplicationContext의 기본적인 리소스 로딩방식만 사용해 리소스를 가져올 필요는 없다.
- 리소스 로더가 인식하는 접두사를 사용하면 특정 위치에 있는 리소스를 가져올 수 있다.

prefix	Resource를 찾는 곳
classpath	루트 클래스패스
file	파일시스템
http	웹애플리케이션 루트

리소스 접근시 사용하는 정규식

- 접두사를 사용해 리소스를 접근할 때 ant스타일의 정규 표현식을 사용해 어떤 파일을 가져올 지 지정할 수도 있다.

- * : '현재 레벨'이나 '단일 레벨'을 가리킴
- ** : '현재 레벨과 그 하위 모든 레벨'을 가리킴

- ex) **classpath:/META-INF/spring/*.xml**
- META-INF/spring 디렉토리가 클래스패스가 되고 이 클래스패스에서 xml확장자를 가진 모든 파일을 로딩함
- **file:/conf/**/*.properties**
- /conf 디렉토리와 이 디렉토리의 모든 하위 디렉토리에서
- Properties 확장자를 갖는 모든 파일을 로딩함

스프링의 설정유형

설정유형	쓰임새
[1] XML설정	서드파티 라이브러리 또는 서로 다른 개발환경과 함께 사용할 수 있다. 쉽지만 설정 파일 수가 많아질 수 있고 여러 파일로 나뉘어져 각각을 추적해야 함
[2] 어노테이션 설정	애플리케이션에 스프링 컨텍스트를 붙이는 방식. 도메인 기반 어노테이션을 사용함으로써 편리함.
[3] 자바빈 설정	자바빈에 설정정보를 두는 방식으로 자주 변하지 않는 빈 또는 컴포넌트에 사용할 수 있다.

BEAN 태그 속성

속성	
id	빈의 식별자.
class	빈의 클래스 지정. 전체 자바 패키지로 지정
scope	빈을 생성할 방법을 지정. 디폴트값은 singleton(단일 객체) 그 외 prototye(빈이 필요할 때 마다 객체 생성) request(각 http요청에서 단일 객체 생성) session (http세션이 존재하는 등안 빈 생성,유지)
init-method	빈이 생성된 후 호출될 메소드 이름을 지정.
destroy-method	빈을 사용 완료한 후 호출될 메소드 이름을 지정
factory-method	빈을 생성하는데 사용될 메소드 이름 지정. 팩토리 패턴으로 객체를 제공할 때 사용. 즉 객체의 인스턴스를 생성할 메소드를 지정한다.