IoC

OOP에선 프로그램이 시작되는 지점에서 다음에 사용할 오브젝트를 결정하고,

결정한 오브젝트를 생성 및 그 안의 메서드를 호출하는 방식의 작업이 반복.

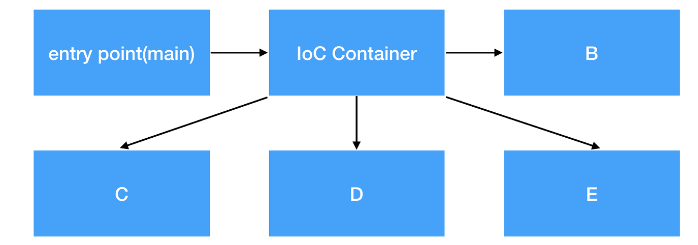
제어의 역전은 main()과 같은 프로그램의 시작을 담당하는 엔트리 포인트를

제외하면 모든 오브젝트는 제어권을 가진 Container에 의해 관리되고

결정됨.

이를 통해 객체간의 결합도를 줄이고(컴포넌트) 유연한 코드를 작성, 가독성,

코드중복, 유지 보수와 같은 문제를 편하게 다룰 수 있음



IoC Container에 모든 관계 설정의 책임을 맡김

Ioc의 분류

DL(Dependency LookUp) : 의존성 검색

Bean에 접근하기 위해 컨테이너가 제공하는 API를 이용하여

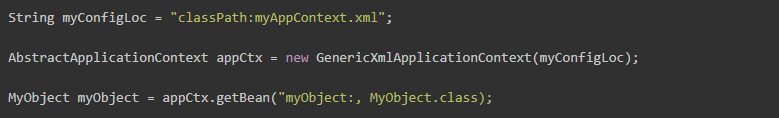
Bean을 LookUp

ID값을 검색하여 bean객체를 불러옴

ex)



xml에 생성된 bean을



클래스에서 선택하여 가져옴

DI(Dependency Injection) : 의존성 주입

bean객체 안에 또 다른 bean객체가 속할 경우 사용

Dependency

코드에서 두 모듈간의 연결

OOP에선 두 클래스 간의 관계

일반적으로 둘 중 하나가 다른 하나를 어떤 용도를 위해 사용함

하나의 모듈이 바뀌면 이 모듈을 Dependency 하는 다른 모듈에 영향을 미침

Dependency없이 독립된 모듈을 컴포넌트(Component)라고 함

DI(Dependency Injection) : 의존성 주입

객체 레퍼런스를 컨테이너로부터 주입 받아서, 실행시에 동적으로 의존관계 생성

Setter Injection (Setter 메서드를 이용한 의존성 삽입)

Constructor Injection (생성자를 이용한 의존성 삽입)

Method Injection (일반 메서드를 이용한 의존성 삽입)

Spring DI 컨테이너( =Ioc컨테이너 = BeanFactory)

DI 컨테이너가 관리하는 객체를 빈이라 하고 컨테이너를 BeanFactory라 함

Bean을 등록, 초기화, 조회, 관리함

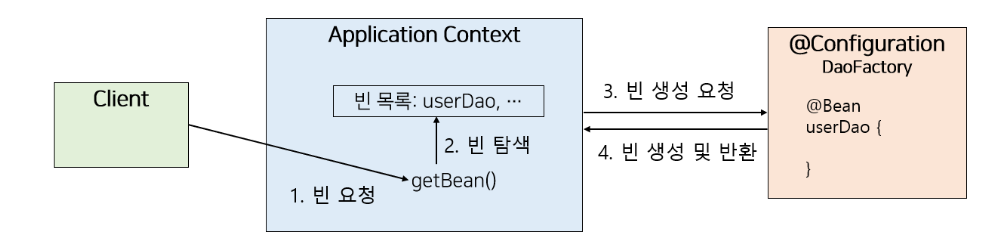
getBean() 을 통해 컨테이너의 빈의 값을 가져옴



빈 이름과 타입으로 property 가져옴

애플리케이션 컨텍스트(Application Context)

beanFactory + 생성, 관계설정등의 제어작업



애플리케이션 컨텍스트의 장점

클라이언트는 구체적인 팩토리 클래스를 알 필요가 없다

애플리케이션 컨텍스트는 종합 IoC 서비스를 제공해준다.

애플리케이션 컨텍스트는 빈을 검색하는 다양한 방법을 제공한다.

bean의 scope

스프링 빈이 생성되고 존재하고 적용되는 범위

default는 singleton

singleton은 스프링 컨테이너가 존재하는 한 계속 유지

prototype은 getBean() 마다 새로운 오브젝트를 만들어줌

injection 방법

Setter Injection : <property>태그

매개변수가 없는 생성자를 호출하여 추기화 후

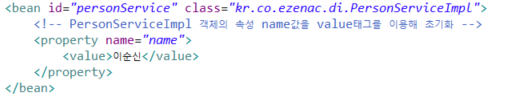
setter 메서드를 통해 Bean을 주입할 때 사용

<value>를 통해 값/객체 주입

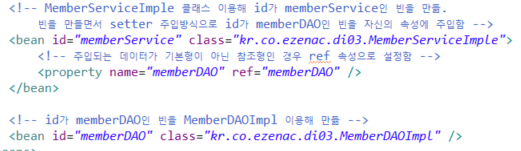
<ref> 통해 bean 주입

컬렉션 타입의 값 주입할 경우

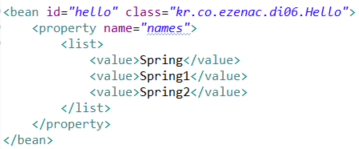
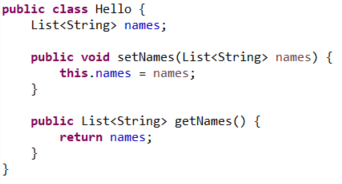
<list>,<set> 등의 태그 이용



<value> 통한 값 주입



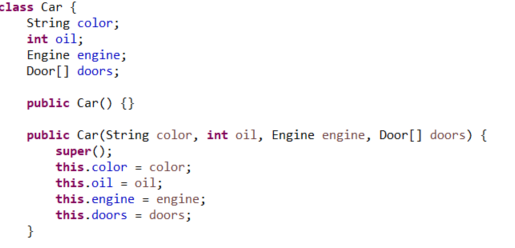
<ref> 통한 bean 주입



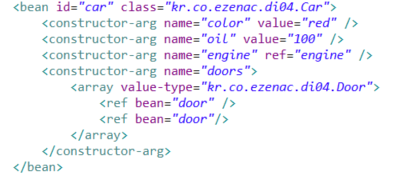
list형태의 객체와 <list>태그를 통한 같은 타입의 bean property주입

Constructor Injection : <constructor-arg>태그

매개변수가 있는 생성자를 호출하여 초기화



매개변수가 있는 생성자를 가지고 있음



생성자를 이용한 초기화

AOP

로직을 기준으로 핵심 기능(Core Concerns)과 부가기능(Cross-cutting Concerns)을

분리하여 그 관점을 기준으로 각각 모듈화

흔어진 관심사를 Aspect로 모듈화하고 핵심적인 비즈니스 로직에서 분리하여 재사용

하기 위함

AOP용어

Aspect : Advice + PointCut

target : 핵심 기능을 담고 있는 모듈 Advice의 부여 대상

Advice: 부가기능을 담고 있는 모듈

JointPoint: Advice가 적용될 위치, 시점,

PointCut: JointPoint의 상세 스펙,

Advice를 적용할 target 메서드를 선별하는 정규 표현식

표현식은 execution으로 시작함