# springIOC解决循环依赖原理

## 背景：

有类BeanA、BeanB，类BeanA中有属性依赖BeanB的实例，BeanB中有属性依赖BeanA的实例，在我们手写代码解决这种依赖怎么写呢？如下：

BeanA a = new BeanA();

BeanB b = new BeanB();

a.setBeanB(b);

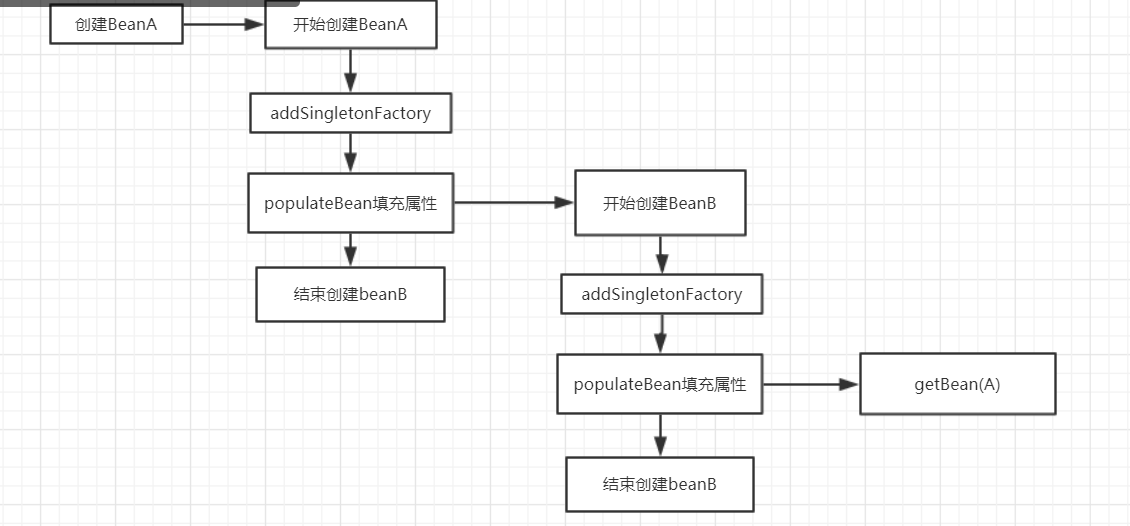
b.setBeanA(a);

## **SpringIOC概念解析**

1. 我们都知道spring是一个IOC容器，其功能就是生产装配一个我们需要的Bean对象，上例中 new BeanA()，只是做了对象的实例化，还没有注入其相应的属性，并不是spring生产的最终Bean，而是将BeanA的属性填充完成了之后的对象，才是spring的最终产物
2. Spring的能解决的循环依赖

|  |  |
| --- | --- |
| 单例的属性的循环依赖 | 能解决 |
| 单例的构造方法循环依赖 | 不能解决 |
| 非单例的循环依赖 | 不能解决 |

## **spring Bean加载过程**



Spring中解决循环依赖的关键map对象

|  |  |
| --- | --- |
| singletonObjects | 用于所有已经创建完成的单例bean存放 |
| singletonsCurrentlyInCreation | 存放正在创建Bean的，一个Bean如果还在实例化、填充属性的过程中，那么这个对象的名字就会放在这个map中 |
| singletonFactories | Spring在创建的时候，会为每一个bean生产一个简单工厂对象，放在这个中 |
| earlySingletonObjects | 调用bean的工厂对象产生的bean放在这个中，这时的bean还没有填充属性 |

1. Spring在getBean的时候，先从singletonObjects中获取，如果获取到了直接返回，没有才执行创建bean的过程
2. 在创建bean的过程中，先实例化bean，并为这个bean添加一个简单工厂ObjectFactory，放入singletonFactories中，这时的bean还没有填充属性的
3. 填充属性，在这步中如果有依赖其他的bean B，会调用其它bean的获取方法，获取完成之后进行属性填充。
4. 在属性填充时，有一个Bean A填充属性bean B，beanB 在创建并填充属性Bean A时，那么他会从singletonFactories中获取一个简单工厂，然后调用这个简单工厂的创建方法，获取一个没有进行属性填充的对象，并放入earlySingletonObjects中；其它bean B获取了一个没有填充属性的对象并对自己进行属性填充，完成自己的创建过程。
5. 创建完成，将创建完成的bean加入到singletonObjects中，并从singletonsCurrentlyInCreation、singletonFactories、earlySingletonObjects中移除

## **疑问**

1. 上述过程为什么需要一个singletonFactories，而不是直接把刚实例化了的对象放入提前曝光的容器earlySingletonObjects中呢？这样也没问题。

答： 其实这里涉及到了aop的一些知识，假如BeanB需要填充的对象BeanA是一个需要代理过的对象，我们直接将BeanA的刚实例化了对象放入BeanB显然不对，而通过简单工厂Object Factory创建的对象却是一个代理过后的对象