# 依赖注入

## 理**解（又称控制反转）**

当一个类(Person)对另一个类(Mobile)有依赖时，不再该类(Person)内部对依赖的类(Moblile)进行实例化，而是之前配置一个beans.xml/注解方式,告诉spring/bean容器所依赖的类(Mobile)，在实例化该类(Person)时，容器自动注入一个所依赖的类(Mobile)的实例。



类如@component的注解@service/@controller/@repository/@[component](https://so.csdn.net/so/search?q=component&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/m0_57549982/article/details/_blank)

类如autowared的注解@resource/@RequiredArgsConstructor

## 依赖注入三种方式

### 1、种类

1.通过构造器来注入；

2.通过setter方法来注入；

3.通过filed变量来注入

## **依赖注入好处**

**依赖注入：是指程序运行过程中，如果需要创建一个对象，无须再代码中new创建，而是依赖外部的注入。**

**spring的依赖注入对调用者和被调用者几乎没有任何要求，完全支持对pojo之间依赖关系的管理**

**new对象：类的头部进行实例化对象和依赖注入一个效果，这个时候该对象不管是否使用都贯穿该类的始终。该类对象不被回收，这个实例化对象也不会被回收。**

**如果要使用多例对象则最好使用new创建对象而不是依赖注入，即使依赖注入有多例模式也不推荐。**

**spring实现了对象池，一些对象创建和使用完毕之后不会被销毁，放进对象池（某种集合）以备下次使用，下次再需要这个对象，不new，直接从池里出去来用。节省时间，节省cpu。做到内存节省并且代码的耦合度也降低。**

**面向接口编程中，让依赖注入只需要找到符合规范的接口注入即可实现调用者和被调用者解耦。对象的调用关系由spring管理。**

**注：1）理解为javabean（一致性设计模式的公共方法将内部域暴露成员属性），java对象（pojo），spring对象（spring容器）**

**2） BeanFactory 与 ApplicationContext 的区别（bean容器=spring容器）BeanFactory是Spring 的核心容器, 主要的 ApplicationContext 实现都【组合】了它的功能**