<http://www.iteye.com/problems/96524>

SPRING什么依赖注入，配置文件，装配类，搞来搞去，现在又用回注解了，请问这跟程序里写new有什么不同呢？

1.

使用new，前提是使用者必然和被使用者发生耦合，至少需要知道构造方法。   
如果一个类在项目中到处被new，这个类就被深度耦合了，任何对其构造方法的修改，都会造成不可预计的后果，有十个类私用它，你就需要修改十个类，有一百个类使用它，你就需要改一百个。   
  
从代码的角度来说，无数个new + 构造方法，就是最大的重复代码。   
  
  
使用IOC容器，等于把创建对象的工作统一交给了IOC容器，被使用者的构造方法发生修改，你调整下容器配置就行了，所有使用者根本不关心你是怎么创建的，拿来用就行了。   
  
更何况spring还提供了非常多的附加功能，单例非单例，是否延迟加载，是否需要使用代理类扩展功能，这些IOC容器在对象创建时统统都能帮你解决。

2.

new是小事，更关键在于IoC。无论是配置文件还是注解都是一种实现方式。注解能提高编程速度，而配置文件更便于管理，而且现在的IDE工具对Spring支持都很好，哪怕是注解也能第一时间找到来源类。   
  
实例化一个对象怎么弄效率影响都不大，而IoC的注入机制对开发效率的影响就太大了。如果还用上了AOP，那用不用Spring直接对整个项目的代码结构产生了巨大影响。   
  
需要一个类的时候直接向Spring去要和自己去new一个对象或者是getInstance的体验是完全不同的。

3.

spring为了管理对象的生命周期的一个工具   
  
假设，你的系统中有很多个类，有的需要单例，有的不需要，那么需要单例的就需要你自己来实现单例了。   
  
而spring就提供了一个生成单例的工具，免得我们重复去开发了。   
  
spring在生成对象实例的时候是用的反射机制，应该是有两种jdk自带的和cglib（以前看到的，记得不是很清楚了），你说的new这种方式就是我们平常用到了。他们都是用来生成对象实例的，只是实现的机制上不同而已。如果为了提供一个类似工具类这样的，直接用new的方式就满足不了大部分的场景，而反射就没有问题了。   
  
至于spring中的配置文件和注解的，这个就不用多说了。看你的理解了，各有优缺点。也争论了好久了。