面试题：Spring的IOC优点，解决了什么问题(腾讯面试题)

一、控制反转IOC

IoC理论提出的观点大体是这样的：借助于“第三方”实现具有依赖关系的对象之间的解耦：

即把各个对象类封装之后，通过IoC容器来关联这些对象类。这样对象与对象之间就通过IoC容器进行联系，但对象与对象之间并没有什么直接联系。

为什么要把这种方式叫做控制反转呢？

软件系统在没有引入IoC容器之前，对象A依赖对象B，那么A对象在实例化或者运行到某一点的时候，自己必须主动创建对象B或者使用已经创建好的对象B，其中不管是创建还是使用已创建的对象B，控制权都在我们自己手上。

如果软件系统引入了Ioc容器之后，对象A和对象B之间失去了直接联系，所以，当对象A实例化和运行时，如果需要对象B的话，IoC容器会主动创建一个对象B注入到对象A所需要的地方。

通过前面的对比，可以看到对象A获得依赖对象B的过程，由主动行为变成了被动行为，即把创建对象交给了IoC容器处理，控制权颠倒过来了，这就是控制反转的由来！

二、依赖注入DI

所谓依赖注入，就是由IoC容器在运行期间，动态地将某种依赖关系注入到对象之中。

依赖注入（DI）和控制反转（IoC）是从不同的角度描述的同一件事情，就是指通过引入IoC容器，利用依赖关系注入的方式，实现对象之间的解耦。

三、使用IOC的好处

1、IoC生成对象的方式转为外置方式，也就是把对象生成放在配置文件里进行定义，这样，当我们更换一个实现子类将会变得很简单，只要修改配置文件就可以了，完全具有热插拔的特性。

2、可维护性比较好，非常便于进行单元测试，便于调试程序和诊断故障。

四、IOC原理

IOC容器要生成的对象是通过配置文件中给出定义的，用java反射技术，根据配置文件中给出的类名生成相应的对象。

在Spring IoC中经常用到一个设计模式，即工厂模式。

工厂模式是指当应用程序中甲组件需要乙组件协助时，并不是在甲组件中直接实例化乙组件对象，而是通过乙组件的工厂获取，即该工厂可以生成某一类型组件的实例对象。在这种模

式下，甲组件无需与乙组件以硬编码的方式耦合在一起，而只需与乙组件的工厂耦合。

五、使用IOC框架应该注意什么

1、软件系统中由于引入了第三方IOC容器，生成对象的步骤变得有些复杂，本来是两者之间的事情，又凭空多出一道手续，所以，我们在刚开始使用IOC框架的时候，会感觉系统变得不太直观。

2、由于IOC容器生成对象是通过反射方式，在运行效率上有一定的损耗。如果你要追求运行效率的话，就必须对此进行权衡。