# spring

1. 如何实现一个IOC容器
2. spring是什么？
   1. 是一个控制反转IOC和面向切面AOP的容器框架
   2. 简化开发，提高效率
3. 谈谈你对AOP的理解
   1. 将程序中的一些交叉业务逻辑（安全，日志，事务）。封装成一个切面。然后注入到目标对象（具体业务逻辑）中
   2. AOP可以对功能进行增强，例如可以在执行某个方法之前（之后）做一些事情
4. 谈谈你对IOC的理解
5. BeanFactory和ApplicationContext有什么区别？
   1. ApplicationContext是BeanFactory的子接口
   2. BeanFactroy只有在使用到某个Bean时，才会对Bean进行加载实例化（延迟加载）
      1. 弊端
         1. 如果Bean的某一个属性没有注入，要到第一次使用调用getBean方法才能抛出异常
   3. ApplicationContext在启动时，一次性创建出所有的Bean
      1. 好处
         1. 有利于检查所依赖属性是否注入
      2. 弊端
         1. 占用内存空间，当应用程序配置Bean多时，启动就慢
   4. ApplicationContext支持更多功能
      1. 可同时加载多个配置文件
      2. 支持国际化
      3. 统一的资源文件的访问方式
      4. 提供在监听器中注册bean的事件
   5. 两者都支持BeanPostProcessor，BeanFactoryPostProcessor的使用
      1. BeanFactory需要手动注册
      2. ApplicationContext是自动注册
6. 描述一下Spring Bean的生命周期？
   1. 1. 通过构造器创建bean实例
   2. 2. 为bean的属性设置值和对其他bean的引用
   3. 3. 把bean实例传递给bean后置处理器的方法
   4. 4. 调用bean的初始化方法
   5. 5. 把bean实例传递给bean后置处理器的方法
   6. 6. 使用bean
   7. 7. 容器关闭时，调用bean的销毁方法
7. AOP执行原理简单说下
   1. AOP使用的是动态代理，它不会去修改字节码，会在内存中临时为方法生成一个AOP对象，此对象包含了目标对象的全部方法，并且在切点做了增强处理
   2. JDK动态代理提供反射来接收被代理的类，要求代理的类必须实现一个接口
   3. 而jdk动态代理的核心是InvocationHandler接口和Proxy类
   4. 如果目标类没有实现接口，AOP则会选择CGLIB来实现
   5. 它是通过继承的方式做动态代理的，如果某个类被标记为final，就没办法了