1.**请解释 Spring Bean 的生命周期？**

Servlet 的生命周期：实例化，初始init，接收请求service，销毁destroy；

Spring 上下文中的 Bean 生命周期也类似：

（1）实列化Bean

对于 BeanFactory 容器，当客户向容器请求一个尚未初始化的bean时，或初始化bean的时候需要注入另一个尚未初始化的依赖时，容器就会调用 createBean进行实例化。

1. 设置对象属性

实例化后的对象被封装在BeanWrapper对象中，紧接着，Spring根据

BeanDefinition中的信息以及通过BeanWrapper提供的设置属性的接口完 成依赖注入。

1. 处理Aware接口

接着，Spring 会检测该对象是否实现了 xxxAware 接口，并将相关的 xxxAware 实例注入给 Bean。

(4）BeanPostProcessor

如果想对 Bean 进行一些自定义的处理，那么可以让Bean实现了BeanPostProcessor接口,那将会调用postProcessBeforeInitialization方法。

（5）lnitializingBean与init-method

如果 Bean 在 Spring 配置文件中配置了 init-method 属性，则会自动调用其配置的初始化方法。

1. 如果这个Bean实现了BeanPostProcessor接口，将会调 postProcessAfterInitialization方法。
2. DisposableBean

当 Bean不再需要时，会经过清理阶段，如果Bean实现了 DisposableBean 这个接口，会调用其实现的destroy()方法。

1. destroy-method

最后，如果这个Bean的Spring配置中配置了destroy-method属性，会自动调用其配置的销毁方法。

**2.解释 Spring 支持的几种 bean 的作用域？**

Spring容器中的bean可以分为5个范围：

1. singleton

默认，每个容器中只有一个bean的实例，单例的模式由BeanFactory自身来维护。

1. Prototype

为每一个 bean 请求提供一个实例。

1. request

为每一个网络请求创建一个实例，在请求完成以后，bean会失效并被垃圾回收器回收。

1. Session

与request范围类似，确保每个session中有一个bean的实例，在 session 过期后，bean 会随之失效。

1. global-session

全局作用域，global-session和Portlet应用相关。

3.**使用注解之前要开启自动扫描功能？**

@Scope 注解 作用域

@Lazy(true) 表示延迟初始化

@Service 用于标注业务层组件

@Controller 用于标注控制层组件（如 struts 中的 action）

@Repository 用于标注数据访问组件，即 DAO 组件。

@Component 泛指组件，当组件不好归类的时候，我们可以使用这个注解进行标注。

@Scope 用于指定 scope 作用域的

@PostConstruct 用于指定初始化方法（用在方法上）

@PreDestory 用于指定销毁方法（用在方法上）

@DependsOn:定义Bean初始化销毁时的顺序

@Autowired 默认按类型装配

@Resource 默认按名称装配，当找不到与名称匹配的bean才会按类型装配。

**4.Spring 框架中的单例 Beans 是线程安全的么？**

Spring 框架并没有对单例bean进行任何多线程的封装处理。关于单例 bean的线程安全和并发问题需要开发者自行去搞定。但实际上，大部分的 Spring bean 并没有可变的状态，所以在某种程度上说Spring的单例bean是线程安全的。

**5.Spring 如何处理线程并发问题？**

在一般情况下，只有无状态的Bean才可以在多线程环境下共享，在Spring 中，绝大部分 Bean 都可以声明为 singleton 作用域，因为Spring对一些 Bean中非线程安全状态采用ThreadLocal进行处理，解决线程安全问题.

ThreadLocal和线程同步机制都是为了解决多线程中相同变量的访问冲突问题,ThreadLocal 会为每一个线程提供一个独立的变量副本，从而隔离了多个线程对数据的访问冲突。

**6.Spring 基于xml注入bean的几种方式?**

（1）Set方法注入

（2）构造器注入

（3）静态工厂注入

（4）实例工厂

7.**Spring 的自动装配？**

在Spring框架xml 配置中共有5种自动装配：

（1）no：默认的方式是不进行自动装配的，通过手工设置ref属性来进行 装配bean。

（2）byName：通过bean的名称进行自动装配，如果一个bean的 property 与另一 beanname 相同，就进行自动装配。

（3）byType：通过参数的数据类型进行自动装配。

（4）constructor：利用构造函数进行装配，并且构造函数的参数通过 byType进行装配。

（5）autodetect：自动探测，如果有构造方法，通过construct的方式自 动装配，否则使用 byType 的方式自动装配。

**8.@Autowired 和@Resource 之间的区别？**

(1) @Autowired 默认是按照类型装配注入的，默认情况下它要求依赖对象 必须存在。

(2) @Resource 默认是按照名称来装配注入的，只有当找不到与名称匹配的 bean才会按照类型来装配注入。

**9.Spring 框架中都用到了哪些设计模式？**

（1）工厂模式：BeanFactory就是简单工厂模式的体现，用来创建对象的 实例。

（2）单例模式：Bean 默认为单例模式。

（3）代理模式：Spring的AOP功能用到了JDK的动态代理和CGLIB字节码 生成技术。

（4）模板方法：用来解决代码重复的问题。

（5）观察者模式：定义对象键一种一对多的依赖关系，当一个对象的状态 发生改变时，所有依赖于它的对象都会得到通知被制动更新。

**10.Spring 事务的实现方式和实现原理？**

Spring事务的本质其实就是数据库对事务的支持，没有数据库的事务支持，spring是无法提供事务功能的。真正的数据库层的事务提交和回滚是通过binlog或者redo log实现的。

（1）Spring事务的种类

1）编程式事务管理使用 TransactionTemplate。

2）声明式事务管理建立在 AOP 之上的。

（2)spring的事务传播行为

1)PROPAGATION\_REQUIRED：如果当前没有事务，就创建一个新事务，如果当前存在事务，就加入该事务，该设置是最常用的设置。

2) PROPAGATION\_SUPPORTS：支持当前事务，如果当前存在事务，就加入该事务，如果当前不存在事务，就以非事务执行。

3）PROPAGATION\_MANDATORY：支持当前事务，如果当前存在事务，就加入该事务，如果当前不存在事务，就抛出异常。

4）PROPAGATION\_REQUIRES\_NEW：创建新事务，无论当前存不存在事务，都创建新事务。

5）PROPAGATION\_NOT\_SUPPORTED：以非事务方式执行操作，如果当前存在事务，就把当前事务挂起

6）PROPAGATION\_NEVER：以非事务方式执行，如果当前存在事务，则抛出异常。

7） PROPAGATION\_NESTED：如果当前存在事务，则在嵌套事务内执行。如果当前没有事务，则按 REQUIRED 属性执行。

（3）Spring中的隔离级别

1）ISOLATION\_DEFAULT：使用数据库默认的事务隔离级别。

2）ISOLATION\_READ\_UNCOMMITTED：读未提交

3）ISOLATION\_READ\_COMMITTED：读已提交

4）ISOLATION\_REPEATABLE\_READ：可重复读

5）ISOLATION\_SERIALIZABLE：一个事务在执行的过程中完全看不到其他事务对数据库所做的更新。