**6、请解释 Spring Bean 的生命周期？**

Servlet生命周期：实例化，初始化init，接收请求，销毁destory;

Spring 上下文的bean生命周期也是类似：

1. 实例化bean；
2. 设置对象属性（依赖注入）
3. 处理Aware接口
4. BeanPostProcessor
5. InitialzingBean与init-method
6. DisposableBean
7. Destory-method

**7、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：单例（默认）
2. Prototype：为每一个bean请求提供一个实例
3. Request：请求
4. Session：与request差不多，每个session都有一个bean，session过期bean失效
5. Global-session：全局作用域

**8、使用注解之前要开启自动扫描功能**

1. @configuration:把一个类作为一个ioc容器，方法上注册了@bean就会作spring容器bean。
2. @scope：作用域
3. @Lazy(true):延迟初始化
4. @Service：标注业务层组件
5. @Controller：标注控制层注解
6. @Repository：标注数据访问组件
7. @Component:泛指组件，组件不好归类时，可以用这个标注
8. @scope用于指定scope作用域
9. @postConstruct：指定初始化方法
10. @preDestory:在方法上销毁
11. @DependsOn:定义bean初始化和销毁顺序
12. @Primay:@Primay作为bean首选，不然抛出异常

**9、Spring 框架中的单例 Beans 是线程安全的么？**

Spring框架没有对单例bean进行线程封装，需要自己搞定，springbean没有可变状态，所以是线程安全的

**10、Spring 如何处理线程并发问题？**

采用threadLocal进行处理，解决线程安全

**11、Spring 基于 xml 注入 bean 的几种方式：**

1. set方法注入
2. 构造器注入：通过index设置参数位置；通过type设置参数类型
3. 静态工厂注入
4. 实例工厂

**12、Spring 的自动装配：**

使用autowire来配置自动装载模式

Spring框架xml配置有5种自动装配：

1. No：通过手工设置ref属性进行装配
2. byName：通过bean名称自动装配
3. byType：通过参数数据类型进行装配
4. Constructor:利用构造函数装配，并且通过byType装配
5. Autoetect:自动探测，有构造方法用constructor装配否侧使用byType

**13、@Autowired 和@Resource 之间的区别**

@autowired默认按照类型装配注入，默认下依赖对象必须存在

@resource默认按照名称装配注入，当找不到按类型注入

**14、Spring 框架中都用到了哪些设计模式？**

1. 工厂模式：创建对象的实例
2. 单例模式：bean默认单例
3. 代理模式：aop用到jdk动态代理和cglib字节码生成技术
4. 模板方法：解决代码重复
5. 观察者模式：当对象状态改变，所有依赖他对象自动更新

**15、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事务种类：(2种)

编程试事务管理使用transaction Template

方法式事务管理建立在AOP之上

1. spring事务传播行为
2. PROPAGATION\_REQUIRED:没事事务就创建一个，有就加入该事务
3. PROPAGATION\_SUPPORTS:支持当前事务，存在就加入，不存在以非事务执 行
4. PROPAGATION\_MANDATORY:创建新事物，存在就加入，不存在抛出异常
5. PROPAGATION\_NOT\_SUPPORTED:以非事务方式执行，存在就挂起
6. PROPAGATION\_NEVER：以非事务方式执行，存在就抛出异常

（3）spring中隔离级别：

1. ISOLATION\_DEFAULT：这是个 PlatfromTransactionManager 默认的隔离级别。使用数据库默认的事务隔离级别。
2. ISOLATION\_READ\_UNCOMMITTED：读未提交，允许另外一个事务可以看到这个事务未提交的数据。
3. ISOLATION\_READ\_COMMITTED：读已提交，保证一个事务修改的数据提交后 才能被另一事务读取，而且能看到该事务对已有记录的更新。
4. ISOLATION\_REPEATABLE\_READ：可重复读，保证一个事务修改的数据提交后才能被另一事务读取，但是不能看到该事务对已有记录的更新。
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。