**Spring 中相关面试题**

[Spring](http://localhost:9093/itour/work/detail?id=135) [面试](http://localhost:9093/itour/work/detail?id=135)

阅读  最后发布于 2022-08-05 21:31

**基础部分**

**1.什么Spring？**

从微观上来说，Spring即spring-framework，是开源的轻量级的一站式JAVAEE框架。Spring在JAVAEE三层结构中分别提供了不同的解决技术，web提供了SpringMvc，Service提供了IOC, 数据访问层提供了JdbcTemplate；

从宏观上来说，Spring指的是Spring构建的整个Spring生态圈。Spring的整个生态圈覆盖了web开发、数据访问、安全控制、分布式、消息服务、移动开发、批处理...

**2.Spring的核心模块？**

控制反转(IOC)：对象创建控制权的反转，将创建对象的控制权交给Spring，让Spring来管理对象；

面向切面(AOP)：采用横向抽取机制，通过动态代理实现增强。即在不修改代码的情况下对对功能进行增强,比如日志、事务管理功能;

**Spring IOC部分**

**1.什么是Spring IOC？**

**2.Spring IOC的底层原理**

Spring IOC底层采用了XML解析、工厂模式、反射技术来实现；

**3.Spring IOC实现有哪几种方式？**

1.Spring提供了Spring IOC实现的两种方式，BeanFactory和ApplicationContext。

2.BeanFactory是IOC容器基本实现，是Spring内部的使用接口，不提供给开发人员使用；

3.ApplicationContext是BeanFactory接口的子接口，提供了更多更强大的功能，一般是由开发人员使用的；

**4.BeanFactory和ApplicationContext接口区别？；**

1.BeanFactory加载配置文件的时候不会创建对象，在获取对象（使用）的时候才会去创建对象；

2.ApplicationContext加载配置文件的时候就会把配置文件中的对象进行创建；

3.当Spring与web项目集成时，一般把耗时、好资源的操作都在项目启动的时候进行处理更合适。

**5.Spring IOC管理Bean的方式**

Spring IOC管理Bean是指Spring创建对象和Spring注入属性操作的方式，Spring提供了两种方式来管理Bean：

1.基于XML配置文件实现；

2.基于注解方式实现

**6.什么是依赖注入?依赖注入和IOC的关系？**

依赖注入（DI），就是注入属性，DI是IOC中的一种具体实现，表示依赖注入，需要在创建对象的基础上完成；

**7.Spring中依赖注入支持的方式？**

1）使用Set方法注入，使用property属性完成属性注入；

2）使用构造函数注入，使用constructor-arg标签完成属性注入；

3）使用p命名空间注入，在配置文件中添加P名称空间，在Bean上进行操作；

**8.Spring中依赖注入如何null值、特殊字符的值、集合？**

1）使用null标签注入null值；

<bean id="book" class="com.itour.Book">

<property name="bname"><null/></property>

</bean>

2）使用CDATA将特殊字符包裹实现特殊字符的值的注入；

<bean id="book" class="com.itour.Book">

<property name="bname">

<!--使用<![CDATA[特殊字符值]]>注入含特殊字符的值-->

<value><![CDATA[<<朱自清散文集>>]]></value>

</property>

</bean>

3）使用ref属性注入JavaBean

<bean id="bookService" class="com.itour.BookService">

<!--注入bookDao对象

name:类里面属性名称；

ref：创建bookDao对象bean标签的id值

-->

<property name="bookDao" ref="bookDaoImpl"> </property>

</bean>

<bean id="bookDaoImpl" class="com.itour.BookDao"></bean>

4)使用Array标签注入数组，使用List，set，map标签来注入对应的集合？

<bean id="book" class="com.itour.Book">

<!--注入数组-->

<property name="arr">

<array>

<value>游泳</value>

<value>打球</value>

</array>

</property>

<!--注入List-->

<property name="list">

<list>

<value>游泳</value>

<value>打球</value>

</list>

</property>

<!--注入Set-->

<property name="sets">

<set>

<value>游泳</value>

<value>打球</value>

</set>

</property>

<!--注入Map-->

<property name="maps">

<map>

<entry key="travel" value="旅行"></entry>

<entry key="food" value="美食"></entry>

</map>

</property>

<!--注入List<JavaBena>-->

<property name="commentList">

<list>

<ref bean="commnet1"></ref>

<ref bean="commnet2"></ref>

</list>

</property>

</bean>

<bean id="commnet1" class="com.itour.Comment"></bean>

<bean id="commnet2" class="com.itour.Comment"></bean>

**9.Spring中 FactoryBean和普通Bean的区别？**

1、Spring 有两种类型bean,一种普通 bean，另外一种工厂 bean (FactoryBean) 。

2、普通 bean：在配置文件中定义 bean 类型就是返回类型。

3.工厂bean：在配置文件定义bean 类型可以和返回类型不一样。通过实现FactoryBean实现对应的方法。

**10.Spring 中Bean作用域是指什么？如何设置Bean是单例的还是示例的？singleton和prototype有什么区别？**

在Spring里面，Bean的作用域指设置创建Bean实例是单实例还是多实例；单实例在整个程序中bean实例只有一个，多实例每次创建对象都会创建一个新的对象。

在Spring配置文件中Bean标签里面有属性(scope)用于设置是单实例还是多实例。scope的属性值singleton表示单实例对象，是默认值，prototype表示多实例对象。

singleton是单实例，prototype是多实例；

设置scope的值为singleton的时候，加载Spring配置文件的时候就会创建单例对象。 设置socpe的值为prototype的时候，在加载spring配置文件的时候不会创建对象，在调用getBean方法的时候会创建多实例对象；

**11.Bean的生命周期**

生命周期是指从对象创建到对象销毁的过程；Bean的声明周期包含了：

1.通过构造器创建 bean 实例（无参数构造）

2.把Bean实例传递Bean后置处理器的方法postProcessBeforeInitialization中；

3.为bean的属性设置值和对其他bean引用（调用 set 方法）

4.把Bean实例传递Bean后置处理器的方法postProcessAfterInitialization中；

5.调用bean的初始化的方法（可以配置init-method）

<bean id="person" class="com.itour.Person" init-method="initMethod" destroy-method="destoryMethod"></bean>

6.bean的使用（对象获取）

7.当容器关闭时候，调用bean 的销毁的方法（使用destroy-method方法配置销毁的方法）

**12.Bean的自动装配**

根据指定装配规则（属性名或者属性类型），Spring自动将匹配的属性值进行注入； 实现自动装配可通过bean标签中的autowire，配置自动装配。autowire有两个值： byName：根据属性名称注入，注入的bean的id必须和类属性名词一致； byType:根据属性类型注入

根据属性名称注入，要求实体类中的属性名称，和要注入的id的名称必须一致；

<bean id="emp" class="com.itour.Emp" autowire="byName"></bean>

<bean id="dept" class="com.itour.Dept"></bean>

根据类型注入，如果配置了多个类型相同的Bean，会报错；

<bean id="emp" class="com.itour.Emp" autowire="byType"></bean>

<bean id="dept" class="com.itour.Dept"></bean>

**13.Spring针对Bean管理提供了哪些注解？**

Spring针对管理Bean提供了四个创建对象的注解，四个注解功能一样，都可以创建Bean的实例。

1.@Component:一般用于实体层对象创建

2.@Service:一般用于Service层对象创建；

3.@Controller:一帮用于controller层对象创建

4.@Repository:一般拥有dao层对象创建；

**14.如何通过注解方式实现属性注入**

实现属性注入spring提供了@Autowired，@Qualifier，@Value注解。java提供了@resource注解；

1.@Autowired：根据属性类型进行自动装配。

2.@Qualifier：根据属性名称进行注入。@Qualifier注解的使用，需要和@Autowired一起使用； 当有多个实现类时，可以指定具体需要注入的实现类。

3.@Value:注入普通类型属性；

4.@Resource:可以根据类型注入，也可以根据名称注入；@Resource(name="userDao") 默认根据类型注入，指定name属性是根据名称注入； @Resource属于javax.annotation.Resource包下，是java提供的注解，不属于spring提供的注解，spring建议使用@Autowired和@Qualifier；

**Spring AOP部分**

**1.什么Spring AOP？**

AOP:采用横向抽取机制，对业务逻辑的各个部分进行隔离，实现对功能的拓展；

**2.Spring AOP底层原理？**

Spring AOP底层使用了横向抽取机制，使用动态代理方式实现；

**3.JDK动态代理和CGlib动态代理的实现？**

1.Spring AOP 底层使用了动态代理来增强类中某个方法的功能。

2.在有接口的情况下，使用JDK的动态代理，通过创建接口实现类代理对象来增强类的方法；

3.没有接口的情况下，使用CGLIB动态代理，通过创建子类的代理对象来增强类的方法；

4.使用JDK动态代理，利用Proxy类里面的newProxyInstance方法创建代理对象；

5.使用CGLIB动态代理，通过字节码技术为一个类创建子类，并在子类中采用方法拦截的技术拦截所有父类方法的调用，顺势织入横切逻辑。

**4.AOP中的术语**

1.)通知/增强(Advice):增强的逻辑称为增强;比如拓展日志功能，这个日志功能被称为增强;分为五种类型：前置通知（在方法之前执行）、后置通知（在方法之后执行）、异常通知:(在方法出现异常时执行)、最终通知:（在后置之后执行）、环绕通知（在方法之前和之后执行）。

2.)切入点(Pointcut)：在类里面可以有许多方法被增强，比如实际的操作中我们只增强了add() 和update(),实际增强的方法就叫切入点;

3.)连接点(Joinpoint)：类里面那些方法可以被增强,这些方法被称为连接点;

4.)切面(Aspect):把增强应用到具体的方法上面的过程称为切面（把增强应用到切入点的过程，例如把权限认证用到登录功能上的过程）

**5.如何实现Spring AOP？**

1.在Spring里面进行AOP的操作，基于aspectj实现提供了两种实现方式：基于aspectj的xml和注解方式。

2.基于xml主要是通过在Spring中配置事务切入点和切面来实现。

3.基于注解方式注意是使用@Aspect标识增强类，在作为通知方法上面添加通知类型注解，使用切入点表达式配置切入点完成方法的增强。

**6.Spring中如何实现事务操作？**

Spring 提供了基于注解和xml方式实现注解，使用注解方式主要是使用@Transactional来实现的对事务的操作；基于xml注意是通过配置通知、切入点和切面来实现事务的操作；

**7.Spring中的传播机制？**

**其他部分**

1.Spring中使用了哪些设计模式？

1.工厂模式：Spring IOC Bean管理

2.单例模式：Spring Bean默认是单例的；

3.代理模式：Spring AOP，Spring事物管理；