**尚硅谷——SSM阶段考套题（一）及答案**

#### 请写出 spring 中常用的依赖注入方式。

通过 setter 方法注入

通过构造方法注入

#### 简述Spring中如何基于注解配置Bean和装配Bean,

(1). 首先要在Spring中配置开启注解扫描

|  |
| --- |
| <context:component-scan base-package=” ”></ context:component-scan> |

(2). 在具体的类上加上具体的注解.

(3). Spring 中通常使用@Autowired 或者是@Resource 等注解进行bean的装配.

#### 三、说出Spring 或者 Springmvc中常用的5个注解 ，并解释含义

[1]. @Component 基本注解，标识一个受Spring管理的组件

[2][. @Controller](mailto:.@Controller) 标识为一个表示层的组件

[3]. @Service 标识为一个业务层的组件

[4]. @Repository 标识为一个持久层的组件

[5]. @Autowired 自动装配

[6]. @Qualifier(“”) 具体指定要装配的组件的id值

[7]. @RequestMapping() 完成请求映射

[8]. @PathVariable 映射请求URL中占位符到请求处理方法的形参

只要说出5个注解并解释含义即可，如上答案只做参考

#### 四、请解释Spring Bean的生命周期？

1.默认情况下，IOC容器中bean的生命周期分为五个阶段:

1. 调用构造器 或者是通过工厂的方式创建Bean对象
2. 给bean对象的属性注入值
3. 调用初始化方法，进行初始化， 初始化方法是通过init-method来指定的.
4. 使用
5. IOC容器关闭时， 销毁Bean对象.

2.当加入了Bean的后置处理器后，IOC容器中bean的生命周期分为七个阶段:

① 调用构造器 或者是通过工厂的方式创建Bean对象

② 给bean对象的属性注入值

③ 执行Bean后置处理器中的 postProcessBeforeInitialization

1. 调用初始化方法，进行初始化， 初始化方法是通过init-method来指定的.
2. 执行Bean的后置处理器中 postProcessAfterInitialization
3. 使用
4. IOC容器关闭时， 销毁Bean对象

只需要回答出第一点即可。 第二点也回答可适当加分.

#### 五、Springmvc中如何解决POST请求中文乱码问题

Springmvc中通过CharacterEncodingFilter解决中文乱码问题.

在web.xml中加入：

<filter>

<filter-name>CharacterEncodingFilter</filter-name>

<filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>

<init-param>

<param-name>encoding</param-name>

<param-value>utf-8</param-value>

</init-param>

</filter>

<filter-mapping>

<filter-name>CharacterEncodingFilter</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

#### 六、MyBatis中 #{}和${}的区别是什么？

#{}是预编译处理，${}是字符串替换。

Mybatis在处理#{}时，会将sql中的#{}替换为?号，调用PreparedStatement的set方法来赋值；

Mybatis在处理${}时，就是把${}替换成变量的值。

使用#{}可以有效的防止SQL注入，提高系统安全性。

#### 七、Mybatis 结果集 的 映射方式有几种，并分别解释每种映射方式如何使用。

自动映射 ，通过resultType来指定要映射的类型即可。

自定义映射 通过resultMap来完成具体的映射规则，指定将结果集中的哪个列映射到对象的哪个属性。

#### 八、简述MyBatis的单个参数、多个参数如何传递及如何取值。

MyBatis传递单个参数，如果是普通类型(String+8个基本)的，取值时在#{}中可以任意指定，如果是对象类型的，则在#{}中使用对象的属性名来取值

MyBatis传递多个参数，默认情况下，MyBatis会对多个参数进行封装Map. 取值时

在#{}可以使用0 1 2 .. 或者是param1 param2..

MyBatis传递多个参数，建议使用命名参数，在Mapper接口的方法的形参前面使用

@Param() 来指定封装Map时用的key. 取值时在#{}中使用@Param指定的key.

#### 九、简述Mybatis的动态SQL，列出常用的6个标签及作用

动态SQL是MyBatis的强大特性之一 基于功能强大的OGNL表达式。

动态SQL主要是来解决查询条件不确定的情况，在程序运行期间，根据提交的条件动态的完成查询

常用的标签:

<if> : 进行条件的判断

<where>：在<if>判断后的SQL语句前面添加WHERE关键字，并处理SQL语句开始位置的AND 或者OR的问题

<trim>：可以在SQL语句前后进行添加指定字符 或者去掉指定字符.

<set>: 主要用于修改操作时出现的逗号问题

<choose> <when> <otherwise>：类似于java中的switch语句.在所有的条件中选择其一

<foreach>：迭代操作

#### 十、简述Spring中自动装配常用的两种装配模式

byName: 根据bean对象的属性名 进行装配

byType： 根据bean对象的属性的类型进行装配,需要注意匹配到多个兼容类型的bean对象时，会抛出异常.

#### 十一、 简述动态代理的原理， 常用的动态代理的实现方式

动态代理的原理: **使用一个代理将对象包装起来**，然后用该代理对象取代原始对象。任何对原始对象的调用都要通过代理。

代理对象决定是否以及何时将方法调 用转到原始对象上

动态代理的方式

基于接口实现动态代理： JDK动态代理

基于继承实现动态代理： Cglib、Javassist动态代理

#### 十二、请解释@Autowired注解的工作机制及required属性的作用

1. 首先会使用byType的方式进行自动装配，如果能唯一匹配，则装配成功，

如果匹配到多个兼容类型的bean, 还会尝试使用byName的方式进行唯一确定.

如果能唯一确定，则装配成功，如果不能唯一确定，则装配失败，抛出异常.

1. 默认情况下， 使用@Autowired标注的属性必须被装配，如果装配不了，也会抛出异常.

可以使用required=false来设置不是必须要被装配.

#### 十三、简述Springmvc中ContextLoaderListener的作用以及实现原理

1. 作用:ContextLoaderListener的作用是通过监听的方式在WEB应用服务器启动时将Spring的容器对象进行初始化.
2. 原理: ContextLoaderListener 实现了ServletContextListener接口，用于监听

ServletContext的创建，当监听到ServletContext创建时，在对应contextInitialized

方法中，将Spring的容器对象进行创建，并将创建好的容器对象设置到ServletContext域对象中，

目的是让各个组件可以通过ServletContext共享到Spring的容器对象

#### 十四、简述Mybatis提供的两级缓存，以及缓存的查找顺序

MyBatis的缓存分为一级缓存和 二级缓存。

一级缓存是SqlSession级别的缓存，默认开启。

二级缓存是NameSpace级别(Mapper)的缓存，多个SqlSession可以共享，使用时需要进行配置开启。

缓存的查找顺序: 二级缓存 一级缓存 数据库

#### 十五、简述Spring与Springmvc整合时，如何解决bean被创建两次的问题

Bean被创建两次的问题是在组建扫描的配置中指定Springmvc只负责扫描WEB相关的组件，Spring扫描除了Springmvc之外的组件。

#### 十六、简述Spring声明式事务中@Transaction中常用的两种事务传播行为

通过propagation来执行事务的传播行为

REQUIRED: 使用调用者的事务，如果调用者没有事务，则启动新的事务运行

REQUIRES\_NEW: 将调用者的事务挂起，开启新的事务运行。

#### 十七、简述Springmvc中处理模型数据的两种方式

1. 使用ModelAndView 作为方法的返回值，将 模型数据 和 视图信息封装到ModelAndView中
2. 使用Map 或者是Model 作为方法的形参. 将模型数据添加到Map或者是Model中.

#### 十八、简述REST中 HiddenHttpMethodFilter过滤器的作用

该过滤器主要负责转换客户端请求的方式，当浏览器的请求方式为POST,并且在请求中能通过 \_method获取到请求参数值,

该过滤器就会进行请求方式的转换.

一般在REST中，都是将POST请求转换为对应的DELETE 或者是PUT

#### 十九、简述Springmvc中如何返回JSON数据

1. 在工程最终加入jackson的jar包
2. 在请求处理方法中 将返回值改为具体返回的数据的类型， 例如 数据的集合类型 List<Employee>等
3. 在请求处理方法上使用@ResponseBody注解

#### 二十、 Springmvc中如何完成重定向

在请求处理方法的返回值前面加 redirect: 前缀, 最终会解析得到RedirectView，RedirectView会完成重定向的操作.

#### 二十一、SpringMVC 的工作原理

（1）用户向服务器发送请求，请求被springMVC 前端控制器 DispatchServlet 捕获；

（2）DispatcherServle 对请求 URL 进行解析，得到请求资源标识符（URL），然后根据该 URL 调用 HandlerMapping将请求映射到处理器 HandlerExcutionChain；

（3）DispatchServlet 根据获得 Handler 选择一个合适的HandlerAdapter 适配器处理；

（4）Handler 对数据处理完成以后将返回一个 ModelAndView（）对象给 DisPatchServlet;

（5）Handler 返回的 ModelAndView() 只是一个逻辑视图并不是一个正式的视图， DispatcherSevlet 通过ViewResolver 试图解析器将逻辑视图转化为真正的视图View;

（6）DispatcherServle 通过 model 解析出 ModelAndView()中的参数进行解析最终展现出完整的 view 并返回给客户端;

#### 二十二、 Spring中常用的设计模式

1. 代理模式——spring 中两种代理方式，若目标对象实现了若干接口，spring 使用jdk 的java.lang.reflect.Proxy类代理。若目标兑现没有实现任何接口，spring 使用 CGLIB 库生成目标类的子类。
2. 单例模式——在 spring 的配置文件中设置 bean 默认为单例模式。
3. 模板方式模式——用来解决代码重复的问题。

比如：RestTemplate、JmsTemplate、JpaTemplate

1. 工厂模式——在工厂模式中，我们在创建对象时不会对客户端暴露创建逻辑，并且是通过使用同一个接口来指向新创建的对象。Spring 中使用 beanFactory 来创建对象的实例。