# Struts2的面试题部分

**1、Struts2与Struts1的联系与区别？为什么要用Struts2？**

struts1与struts2都是mvc框架的经典实现模式。Struts2不是从Struts1升级而来,而是由WebWork改名而来,而WebWork只是Xwork加了很多WEB拦截器而已区别：1.核心控制器改成了过滤器（过滤器比Servlet的级别要高，因为程序运行时是先进入过滤器再进入Servlet）2.struts1严重依赖于Servlet(因为太过于依赖于api的HttpServletRequest与HttpServletResponse的两个参数),struts2就则脱离了Servlet的API。3.管理Action时struts1是单例模式，struts2是每个请求产生一个实例。4.在表达式的支持上struts2不仅有jstl，还有功能更加强大的ognl表达式。5.struts1的类型转换是单向的(页面到ActionForm),struts2是双向的(页面到Action再到页面回显)6.校验，struts1没有针对具体方法的校验，struts2提供了指定某个方法进行效验，还有框架校验。7.struts2提供了拦截器，利用拦截器可以在访问Action之前或之后增加如权限拦截等功能。8.struts2提供了全局范围，包范围，Action范围的国际化资源文件管理实现。9.struts2支持多种视图类型

**2、Struts2的核心是什么,体现了什么思想？**

Struts2的核心是拦截器,基本上核心功能都是由拦截器完成,拦截器的实现体现了AOP(面向切面编程)思想

**3、为何继承ActionSupport？**

因为ActionSupport实现了Action接口，提供了国际化，校验的功能。ActionSupport实现了国际化功能：因为它提供了一个getText(String key)方法实现国际化,该方法从资源文件上获取国际化信息。Action接口提供了五个常量(success,error,login,input,none)。(s?k'ses,er?(r) n?n

**4、模型驱动与属性驱动是什么 模型驱动使用时注意什么问题？**

答：模型驱动与属性驱动都是用来封装数据的。1.模型驱动：在实现类中实现ModelDriven<T>接口使用泛型把属性类封装起来，重写getModel()方法，然后在实现类里创建一个属性类的实例，2.属性驱动：在实现类里定义属性，生成get与set方法，通过属性来拿值。

**5、Struts2是怎样进行值封装的？**

struts2的值封装实际上是采用了ognl表达式.6.Struts2如何进行校验1.每个Action类有一个校验文件，命名 Action类名-validation.xml,且与Action类同目录，这是对action里面所有的方法进行校验。2.对Action里面的指定方法做校验使用Action的类名-访问路径\_方法名-validation.xml。在效验文件里又分为两种:字段校验：字段用什么校验器来校验。非字段校验：是用校验器校验什么字段。

**6、谈谈Struts2 的国际化在struts2中是使用了拦截器来实现国际化？**

具体是先配置属性文件,格式是文件名\_语言\_国家名.properties,然后放的位置有Action同包位置,package位置,全局位置(要在常量里面配置), 然后页面通过s:text或者key属性去访问8.OGNL是什么你在项目中如何使用它OGNL是：对象图形导航语言。在struts2中的作用：ognl的实现关系:ActionConetxt。ognl 3个常用的符号 # $ %#1 构造map，list集合。 2 取ognl上下文的值。 3 用来过滤集合。 $1 在校验框架中取资源文件中的值。2 可以在配置文件中传递参数。%使用百分号可以进行转义。

**7、Strust2如何访问Servlet** ？

API1：通过ActionContext可以获得request，application，session三个作用域(struts2实际上是通过适配器来把servlet的api转换为一个map，并把这些map放在ActionContext里面)。 2：通过ActionContext的子类ServletActionContext去获取原滋原味的API。3：可以通过实现ServletRequestAware接口，重写里面的setServletRequest方法可以获得request，实际上是通过set的依赖注入。

**8、什么是拦截器， 说说Struts2用拦截器来干什么， 说出6个拦截器来解释？**

在访问类的某个方法或者属性之前执行，拦截的是Action的请求，进行拦截然后在方法的执行前或者之后加入某些操作。作用：国际化，权限，效验等与源代码无关的操作。拦截器：国际化，表单重复提交，类型转换，自动装配，数据封装,异常处理，模型驱动，请求参数，处理类型转换错误，日志拦截器。

**9、如何实现自定义拦截器？ 需要注意什么？**

实现：1.可以实现Interceptor接口，重写doIntercept方法指定某个方法进行拦截，或者继承AbstractInterceptor类，重写intercept方法。2.在xml配置文件中定义自定义拦截器，然后注入到拦截器栈中，再把拦截器引用到action中。3.可以配置到某个action单独使用，也可以配置到包下面的所有action使用。 注意：要把默认的拦截器栈重新引用，否则会被覆盖。

**10、ActionContext是用来干什么的**？

ActionContext是Action的上下文。

**11、为什么要继承默认的包？**

因为在Strtus2里面默认有很多的常量,拦截器,还有一些bean,如果继承默认的包,这些默认值和常量就会继承过来。

**12、常见的有哪几种结果集类型？**

dispatcher：指得是转发，默认的结果集类型redirect：重定向,redirectAction：重定向到一个Actionstream：是返回一个流,一般用在文件下载上面

**13、开发项目时struts2在页面怎样拿值？**

从值栈中或者是ognl的上下文

**14、怎么样用Struts2进行文件的上传或者下载上传**？

1.在jsp用了s:file标签,把s:from的enctype 属性设置为 multipart/form-data2.在action里面定义三个字段 File file ,String [file]ContentType,String [file] FileName3.用输出流转化成硬盘上面的文件下载：1.在struts.xml中配置result为stream的类型2.在action定义四个属性,默认的有个是InputStream 的类型叫inputStream的,从硬盘上面读取文件到这个流赋值即可.contentType; contentLength; contentDisposition; inputStream;

**15、简单讲下struts里面的标签,说不少于5个**？

s:ifs:forms:urls:propertys:iterater18.默认struts2里面的标签取值都是从那里取值的默认都是从OGNL值栈里面取值的

**16、ValueStack分为那两部分,组成部分是什么,分别怎么访问**？

分为:对象栈和ContextMap访问：1.对象栈前台可以直接访问2.ContextMap访问的时候前面加

**17、标签<s:property>和el表达式有什么区别**？

相同：都可以从值栈里面取值区别：el表达式只能取属性有get set方法的简单数据类型s:property标签：取得数组的一些方法等复杂数据对象类型

**18、说下重复提交的解决思路，Struts2是怎么样解决重复提交的**？

1.在页面表单域加入一个隐藏的s:token字段,然后在session里面也加入同样的值2.当用户第一次请求的时候,把request的值和session对比,相同就通过请求,执行下面拦截器,并且移除Session里面的值3.如果第二次请求时候,Session已经没有这个值,那么就阻断当前请求,定位重复提交 Struts2是通过s:token标签来完成重复提交思路的

# spring的面试题部分

**1、Spring的aop你怎样实现?**

用动态代理和cglib实现,有接口的用动态代理,无接口的用cglib

**2、Spring在SSH起什么作用**？

为大部分框架提供模版,常见的核心类提供初始化,并且整合三层框架

**3、Spring容器内部怎么实现的**？

内部用Map实现,或者说HashMap

**4、怎么样理解IOC与AOP？**

IOC是一种控制反转的思想,降低了对象的耦合度,AOP是面向切面编程,非侵入式编程,实现了非业务性编程(公共功能),譬如日志,权限,事务等等

**5、Spring的事务，事务的作用**？

Spring里面的事务分为编程式事务和声明式事务,一般用声明式事务,用来控制数据操作的完整性,一致性

**6、Spring的IOC和AOP你在项目中是怎么使用的？**

IOC主要来解决对象之间的依赖问题,把所有的bean的依赖关系通过配置文件或者注解关联起来,降低了耦合度,AOP一般用来整合框架时候都可以用得到,事务用的最多,还有个别日志,权限功能也可以用到

**7、Spring主要使用了什么模式？**

工厂模式-->每个Bean的创建通过方法单例模式-->默认的每个Bean的作用域都是单例代理模式-->关于AOP的实现是通过代理,体现代理模式

**8、Spring bean的作用域**？

Scope作用域有4种,常见的有单例或者多例,默认是单例,request,session

**9、Spring的事务是如何配置的？**

1.先配置事务管理器TransactionManager,不同的框架有不同属性

2.再配置事务通知和属性,通过tx:advice

3.配置<aop:config>,设置那些方法或者类需要加入事务

**10、Spring的配置文件最好使用什么文件？**

xml,因为它是最简单,最流行的数据格式

**11、你使用过Spring中的哪些技术？**

bean的管理,AOP技术,IOC技术 ,事务等

**12、为什么要用Spring?**

降低对象耦合度,让代码更加清晰,提供一些常见的模版

**13、说下Spring的注解?**

1.bean的标记注解@Component 通用注解 @Repository 持久层注解 @Service 业务层注解 @Controller:表现层注解2.bean的自动装配注解@AutoWired 默认是按照类型装配,如果有多个类型实现可以用Qualifier来指定名@Resource 默认是按照名称来装配,是JDK里面自带的注解,默认情况下用@AutoWired注解

1.spring类创建对象的

@Controller @Service @Repository @Component

2.spring类的作用域

@Scope(“singleton/ prototype/request/session”)

3.方法参数对应的

@Qualifier()

4.spring mvc 路径

@RequestMapping(value=””,params=””,method=””)

5.spring mvc 参数对应

@PathVariable @RequestParam

6.spring aop 类

@Aspect

7.spring aop方法

@Before(“execution(..)”) @After @AfterReturning @Around @AfterThrowing

8.事务: 事务的底层是aop

@Transactional

**14、写过类似Spring AOP的操作吗？**

简单的写过,譬如前置通知,后置通知的方法,环绕通知,事务就是典型的AOP的实现

**15、Spring中的AOP在你项目中是怎么使用的，用在哪里？**

Struts2和hibernate整合时候都可以用得到, 事务用的最多,还有个别日志,权限功能也可以用到

**16、Spring的事务（传播属性，隔离级别）?**

七大传播属性,四大隔离级别

**17、Spring DI的几种方式?**

setter注入和构造器注入,一般用setter注入

**18、依赖注入的原理?**

就是通过反射机制生成想要的对象注入

**19、说一下整合Spring的核心监听器?**

这个是在SSH整合的时候使用,是整个WEB项目启动的时候初始化Spring的容器. 具体是在web.xml里面配置的ContextLoaderListenerSpring配置文件中的核心是个监听器，是用来初始化Spring的容器

**20、Spring你们为什么用配置文件而不使用注解?**

配置文件耦合度低,容易维护,尤其是在切面或者事务的时候,只配置一次就可以让很多代码拥有事务.

**21、Spring和Hibernate的事务有什么区别？**

Spring的事务提供了统一的事务处理机制,包含了JDBC,Hibernate,IBatis等事务实现,而Hibernate只处理自己事务

**22、Struts2与Spring整合先启动那个容器?**

先启动监听器,因为先要初始化容器,初始化容器了以后Action才能从容器里面获得

**23、让你写Spring的容器，你是怎样实现的？**

我们可以写一个HashMap,如果并发考虑的话要写并发的Map,把bean的名字放在map的key,bean的实现map的value

**24、谈谈Spring的IOC和AOP，如果不用Spring，怎么去实现这两个技术?**

ioc用反射实现 ,AOP用动态代理实现

**25、Spring事务和Hibernate事务的操作上面的区别？**

hibernate的事务只能手动显示代码的方式控制创建事务与提交事务以及回滚。Spring可以通过配置文件设定一类class事务的创建与提交以及回滚，也可以显示代码方式控制。

**26、讲下Spring的七大事务传播有七个**?

常用有两个REQUIERD, REQUIRED\_NEW,REQUIERD表示两个事务的方法调用的时候,前面的时候和后面的合并成一个事务,REQUIRED\_NEW是重启一个事务,各干各的

**27、在同一进程里，有A，B两个方法都对不同的表进行更新数据，假如A方法出异常了，若要B方法执行，怎样配置事务级别，若不要B方法执行，又该怎样配置？**

前者用REQUIRED\_NEW,后者用REQUIRED

**28、事务并发会引起什么问题,怎么解决**?

事务并发会引起脏读,幻读,不可重复读等问题,设定事务的隔离级别就可以解决

**29、事务的隔离级别?**

Spring定义有四种,但是常见的是READ\_COMMIT,Oralce有两种实现,MySQL有四种**30.Spring的IOC容器与工厂类有什么区别？**

IOC(Inversion of Control)对Bean的控制能力更强,能控制对象自动注入,还可以控制生命周期,而工厂类只是简单的创建一个对象,没有什么控制能力

**31、事务的安全问题?**

锁机制的实现原理及在项目中的使用锁有悲观锁和乐观锁,悲观锁一般假设每个人都会修改数据,默认情况下把数据都锁住,影响性能,但安全性高.乐观锁是假设每个人都只读下数据,不会修改数据,性能比较高,但是安全性较低,一般通过增加类似于版本控制里面版本号来解决问题

**32、讲下BeanFactory和ApplicationContext的区别**?

BeanFactory是Spring容器顶级核心接口,比较早,但功能比较少,getBean就是BeanFactory定义的,ApplicationContext是Spring里面的另外一个容器顶级接口,它继承于BeanFactory,但是提供的功能譬如校验,国际化,监听,对Bean的管理功能比较多,一般使用ApplicationContext

**33、什么是Spring？**

Spring是一个开源的Java EE开发框架。Spring框架的核心功能可以应用在任何Java应用程序中，但对Java EE平台上的Web应用程序有更好的扩展性。Spring框架的目标是使得Java EE应用程序的开发更加简捷，通过使用POJO为基础的编程模型促进良好的编程风格。

**34、Spring有哪些优点？**

轻量级：Spring在大小和透明性方面绝对属于轻量级的，基础版本的Spring框架大约只有2MB。

控制反转(IOC)：Spring使用控制反转技术实现了松耦合。依赖被注入到对象，而不是创建或寻找依赖对象。

面向切面编程(AOP)： Spring支持面向切面编程，同时把应用的业务逻辑与系统的服务分离开来。

容器：Spring包含并管理应用程序对象的配置及生命周期。

MVC框架：Spring的web框架是一个设计优良的web MVC框架，很好的取代了一些web框架。

事务管理：Spring对下至本地业务上至全局业务(JAT)提供了统一的事务管理接口。

异常处理：Spring提供一个方便的API将特定技术的异常(由JDBC, Hibernate, 或JDO抛出)转化为一致的、Unchecked异常。

**35、Spring框架有哪些模块？**

Spring框架的基本模块如下所示：

Core module

Bean module

Context module

Expression Language module

JDBC module

ORM module

OXM module

Java Messaging Service(JMS) module

Transaction module

Web module

Web-Servlet module

Web-Struts module

Web-Portlet module

**36、解释核心容器(应用上下文)模块**

这是Spring的基本模块，它提供了Spring框架的基本功能。BeanFactory 是所有Spring应用的核心。Spring框架是建立在这个模块之上的，这也使得Spring成为一个容器。

**37、BeanFactory – BeanFactory 实例**

BeanFactory是工厂模式的一种实现，它使用控制反转将应用的配置和依赖与实际的应用代码分离开来。

最常用的BeanFactory实现是XmlBeanFactory类。

**38、XmlBeanFactory**

最常用的就是org.springframework.beans.factory.xml.XmlBeanFactory，它根据XML文件中定义的内容加载beans。该容器从XML文件中读取配置元数据，并用它来创建一个完备的系统或应用。

**39、解释AOP模块**

AOP模块用来开发Spring应用程序中具有切面性质的部分。该模块的大部分服务由AOP Aliance提供，这就保证了Spring框架和其他AOP框架之间的互操作性。另外，该模块将元数据编程引入到了Spring。

**40、解释抽象JDBC和DAO模块**

通过使用抽象JDBC和DAO模块保证了与数据库连接代码的整洁与简单，同时避免了由于未能关闭数据库资源引起的问题。它在多种数据库服务器的错误信息之上提供了一个很重要的异常层。它还利用Spring的AOP模块为Spring应用程序中的对象提供事务管理服务。

**41、解释对象/关系映射集成模块**

Spring通过提供ORM模块在JDBC的基础上支持对象关系映射工具。这样的支持使得Spring可以集成主流的ORM框架，包括Hibernate, JDO, 及iBATIS SQL Maps。Spring的事务管理可以同时支持以上某种框架和JDBC。

**42、解释web模块**

Spring的web模块建立在应用上下文(application context)模块之上，提供了一个适合基于web应用程序的上下文环境。该模块还支持了几个面向web的任务，如透明的处理多文件上传请求及将请求参数同业务对象绑定起来。

**43、解释Spring MVC模块**

Spring提供MVC框架构建web应用程序。Spring可以很轻松的同其他MVC框架结合，但Spring的MVC是个更好的选择，因为它通过控制反转将控制逻辑和业务对象完全分离开来。

**44、Spring的配置文件**

Spring的配置文件是一个XML文件，文件包含了类信息并描述了这些类是如何配置和互相调用的。

**45、Spring IoC容器是什么？**

Spring IOC负责创建对象、管理对象(通过依赖注入)、整合对象、配置对象以及管理这些对象的生命周期。

**46、IOC有什么优点？**

IOC或依赖注入减少了应用程序的代码量。它使得应用程序的测试很简单，因为在单元测试中不再需要单例或JNDI查找机制。简单的实现以及较少的干扰机制使得松耦合得以实现。IOC容器支持勤性单例及延迟加载服务。

**47、应用上下文是如何实现的？**

FileSystemXmlApplicationContext 容器加载XML文件中beans的定义。XML Bean配置文件的完整路径必须传递给构造器。

FileSystemXmlApplicationContext 容器也加载XML文件中beans的定义。注意，你需要正确的设置CLASSPATH，因为该容器会在CLASSPATH中查看bean的XML配置文件。

WebXmlApplicationContext：该容器加载xml文件，这些文件定义了web应用中所有的beans。

**48、Bean Factory和ApplicationContext有什么区别？**

ApplicationContext提供了一种解决文档信息的方法，一种加载文件资源的方式(如图片)，他们可以向监听他们的beans发送消息。另外，容器或者容器中beans的操作，这些必须以bean工厂的编程方式处理的操作可以在应用上下文中以声明的方式处理。应用上下文实现了MessageSource，该接口用于获取本地消息，实际的实现是可选的。

**49、Spring应用程序看起来像什么？**

一个定义功能的接口

实现包括属性，setter和getter方法，功能等

Spring AOP

Spring的XML配置文件

使用该功能的客户端编程

依赖注入

**50、Spring中的依赖注入是什么？**

依赖注入作为控制反转(IOC)的一个层面，可以有多种解释方式。在这个概念中，你不用创建对象而只需要描述如何创建它们。你不必通过代码直接的将组件和服务连接在一起，而是通过配置文件说明哪些组件需要什么服务。之后IOC容器负责衔接。

**51、有哪些不同类型的IOC(依赖注入)？**

构造器依赖注入：构造器依赖注入在容器触发构造器的时候完成，该构造器有一系列的参数，每个参数代表注入的对象。

Setter方法依赖注入：首先容器会触发一个无参构造函数或无参静态工厂方法实例化对象，之后容器调用bean中的setter方法完成Setter方法依赖注入。

**52、你推荐哪种依赖注入？构造器依赖注入还是Setter方法依赖注入？**

你可以同时使用两种方式的依赖注入，最好的选择是使用构造器参数实现强制依赖注入，使用setter方法实现可选的依赖关系。

**53、什么是Spring Beans？**

Spring Beans是构成Spring应用核心的Java对象。这些对象由Spring IOC容器实例化、组装、管理。这些对象通过容器中配置的元数据创建，例如，使用XML文件中定义的创建。

在Spring中创建的beans都是单例的beans。在bean标签中有一个属性为”singleton”,如果设为true，该bean是单例的，如果设为false，该bean是原型bean。Singleton属性默认设置为true。因此，spring框架中所有的bean都默认为单例bean。

**54、Spring Bean中定义了什么内容？**

Spring Bean中定义了所有的配置元数据，这些配置信息告知容器如何创建它，它的生命周期是什么以及它的依赖关系。

**55、如何向Spring 容器提供配置元数据？**

有三种方式向Spring 容器提供元数据:

XML配置文件

基于注解配置

基于Java的配置

**56、你如何定义bean的作用域？**

在Spring中创建一个bean的时候，我们可以声明它的作用域。只需要在bean定义的时候通过’scope’属性定义即可。例如，当Spring需要产生每次一个新的bean实例时，应该声明bean的scope属性为prototype。如果每次你希望Spring返回一个实例，应该声明bean的scope属性为singleton。

**57、说一下Spring中支持的bean作用域**

Spring框架支持如下五种不同的作用域：

singleton：在Spring IOC容器中仅存在一个Bean实例，Bean以单实例的方式存在。

prototype：一个bean可以定义多个实例。

request：每次HTTP请求都会创建一个新的Bean。该作用域仅适用于WebApplicationContext环境。

session：一个HTTP Session定义一个Bean。该作用域仅适用于WebApplicationContext环境.

globalSession：同一个全局HTTP Session定义一个Bean。该作用域同样仅适用于WebApplicationContext环境.

bean默认的scope属性是’singleton‘。

**58、Spring框架中单例beans是线程安全的吗？**

不是，Spring框架中的单例beans不是线程安全的。

**59、解释Spring框架中bean的生命周期**

Spring容器读取XML文件中bean的定义并实例化bean。

Spring根据bean的定义设置属性值。

如果该Bean实现了BeanNameAware接口，Spring将bean的id传递给setBeanName()方法。

如果该Bean实现了BeanFactoryAware接口，Spring将beanfactory传递给setBeanFactory()方法。

如果任何bean BeanPostProcessors 和该bean相关，Spring调用postProcessBeforeInitialization()方法。

如果该Bean实现了InitializingBean接口，调用Bean中的afterPropertiesSet方法。如果bean有初始化函数声明，调用相应的初始化方法。

如果任何bean BeanPostProcessors 和该bean相关，调用postProcessAfterInitialization()方法。

如果该bean实现了DisposableBean，调用destroy()方法。

**60、哪些是最重要的bean生命周期方法？能重写它们吗？**

有两个重要的bean生命周期方法。第一个是setup方法，该方法在容器加载bean的时候被调用。第二个是teardown方法，该方法在bean从容器中移除的时候调用。

bean标签有两个重要的属性(init-method 和 destroy-method)，你可以通过这两个属性定义自己的初始化方法和析构方法。Spring也有相应的注解：@PostConstruct 和 @PreDestroy。

**61、什么是Spring的内部bean？**

当一个bean被用作另一个bean的属性时，这个bean可以被声明为内部bean。在基于XML的配置元数据中，可以通过把元素定义在 或元素内部实现定义内部bean。内部bean总是匿名的并且它们的scope总是prototype。

**62、如何在Spring中注入Java集合类？**

Spring提供如下几种类型的集合配置元素：

list元素用来注入一系列的值，允许有相同的值。

set元素用来注入一些列的值，不允许有相同的值。

map用来注入一组”键-值”对，键、值可以是任何类型的。

props也可以用来注入一组”键-值”对，这里的键、值都字符串类型。

**63、什么是bean wiring？**

Wiring，或者说bean Wiring是指beans在Spring容器中结合在一起的情况。当装配bean的时候，Spring容器需要知道需要哪些beans以及如何使用依赖注入将它们结合起来。

**64、什么是bean自动装配？**

Spring容器可以自动配置相互协作beans之间的关联关系。这意味着Spring可以自动配置一个bean和其他协作bean之间的关系，通过检查BeanFactory 的内容里没有使用和< property>元素。

**65、解释自动装配的各种模式？**

自动装配提供五种不同的模式供Spring容器用来自动装配beans之间的依赖注入:

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

byName：通过参数名自动装配，Spring容器查找beans的属性，这些beans在XML配置文件中被设置为byName。之后容器试图匹配、装配和该bean的属性具有相同名字的bean。

byType：通过参数的数据类型自动自动装配，Spring容器查找beans的属性，这些beans在XML配置文件中被设置为byType。之后容器试图匹配和装配和该bean的属性类型一样的bean。如果有多个bean符合条件，则抛出错误。

constructor：这个同byType类似，不过是应用于构造函数的参数。如果在BeanFactory中不是恰好有一个bean与构造函数参数相同类型，则抛出一个严重的错误。

autodetect：如果有默认的构造方法，通过 construct的方式自动装配，否则使用 byType的方式自动装配。

**66、自动装配有哪些局限性？**

自动装配有如下局限性：

重写：你仍然需要使用 和< property>设置指明依赖，这意味着总要重写自动装配。

原生数据类型:你不能自动装配简单的属性，如原生类型、字符串和类。

模糊特性：自动装配总是没有自定义装配精确，因此，如果可能尽量使用自定义装配。

**67、你可以在Spring中注入null或空字符串吗？**

完全可以。

**68、什么是Spring基于Java的配置？给出一些注解的例子**

基于Java的配置允许你使用Java的注解进行Spring的大部分配置而非通过传统的XML文件配置。

以注解@Configuration为例，它用来标记类，说明作为beans的定义，可以被Spring IOC容器使用。另一个例子是@Bean注解，它表示该方法定义的Bean要被注册进Spring应用上下文中。

**69、什么是基于注解的容器配置?**

另外一种替代XML配置的方式为基于注解的配置，这种方式通过字节元数据装配组件而非使用尖括号声明。开发人员将直接在类中进行配置，通过注解标记相关的类、方法或字段声明，而不再使用XML描述bean之间的连线关系。

**70、如何开启注解装配？**

注解装配默认情况下在Spring容器中是不开启的。如果想要开启基于注解的装配只需在Spring配置文件中配置元素即可。

**71、@Required 注解**

@Required表明bean的属性必须在配置时设置，可以在bean的定义中明确指定也可通过自动装配设置。如果bean的属性未设置，则抛出BeanInitializationException异常。

**72、@Autowired 注解**

@Autowired 注解提供更加精细的控制，包括自动装配在何处完成以及如何完成。它可以像@Required一样自动装配setter方法、构造器、属性或者具有任意名称和/或多个参数的PN方法。

**73、@Qualifier 注解**

当有多个相同类型的bean而只有其中的一个需要自动装配时，将@Qualifier 注解和@Autowire 注解结合使用消除这种混淆，指明需要装配的bean。

**74、在Spring框架中如何更有效的使用JDBC？**

使用Spring JDBC框架，资源管理以及错误处理的代价都会减轻。开发人员只需通过statements和queries语句从数据库中存取数据。Spring框架中通过使用模板类能更有效的使用JDBC，也就是所谓的JdbcTemplate(例子)。

**75、JdbcTemplate**

JdbcTemplate类提供了许多方法，为我们与数据库的交互提供了便利。例如，它可以将数据库的数据转化为原生类型或对象，执行写好的或可调用的数据库操作语句，提供自定义的数据库错误处理功能。

**76、Spring对DAO的支持**

Spring对数据访问对象(DAO)的支持旨在使它可以与数据访问技术(如 JDBC, Hibernate 及JDO)方便的结合起来工作。这使得我们可以很容易在的不同的持久层技术间切换，编码时也无需担心会抛出特定技术的异常。

**77、使用Spring可以通过什么方式访问Hibernate？**

使用Spring有两种方式访问Hibernate：

使用Hibernate Template的反转控制以及回调方法

继承HibernateDAOSupport，并申请一个AOP拦截器节点

**78、Spring支持的ORM**

Spring支持一下ORM：

Hibernate

iBatis

JPA (Java -Persistence API)

TopLink

JDO (Java Data Objects)

OJB

**79、如何通过HibernateDaoSupport将Spring和Hibernate结合起来？**

使用Spring的SessionFactory 调用LocalSessionFactory。结合过程分为以下三步：

配置Hibernate SessionFactory

继承HibernateDaoSupport实现一个DAO

使用AOP装载事务支持

**80、Spring支持的事务管理类型**

Spring支持如下两种方式的事务管理：

编程式事务管理：这意味着你可以通过编程的方式管理事务，这种方式带来了很大的灵活性，但很难维护。

声明式事务管理：这种方式意味着你可以将事务管理和业务代码分离。你只需要通过注解或者XML配置管理事务。

**81、Spring框架的事务管理有哪些优点？**

它为不同的事务API(如JTA, JDBC, Hibernate, JPA, 和JDO)提供了统一的编程模型。

它为编程式事务管理提供了一个简单的API而非一系列复杂的事务API(如JTA).

它支持声明式事务管理。

它可以和Spring 的多种数据访问技术很好的融合。

**82、你更推荐那种类型的事务管理？**

许多Spring框架的用户选择声明式事务管理，因为这种方式和应用程序的关联较少，因此更加符合轻量级容器的概念。声明式事务管理要优于编程式事务管理，尽管在灵活性方面它弱于编程式事务管理(这种方式允许你通过代码控制业务)。

**83、解释AOP**

面向切面编程,或AOP允许程序员模块化横向业务逻辑，或定义核心部分的功能，例如日志管理和事务管理。

**84、切面(Aspect)**

AOP的核心就是切面，它将多个类的通用行为封装为可重用的模块。该模块含有一组API提供 cross-cutting功能。例如,日志模块称为日志的AOP切面。根据需求的不同，一个应用程序可以有若干切面。在Spring AOP中，切面通过带有@Aspect注解的类实现。

**85、在Spring AOP中concern和 cross-cutting concern的区别是什么？**

Concern(核心逻辑)：表示在应用程序中一个模块的行为。Concern可以定义为我们想要实现的功能。

Cross-cutting concern(横向的通用逻辑)：指的是整个应用程序都会用到的功能，它影响整个应用程序。例如，日志管理（Logging）、安全管理（Security）以及数据交互是应用程序的每个模块都要涉及到的，因此这些都属于Cross-cutting concern。

**86、连接点(Join point)**

连接点代表应用程序中插入AOP切面的地点。它实际上是Spring AOP框架在应用程序中执行动作的地点。

**87、通知(Advice)**

通知表示在方法执行前后需要执行的动作。实际上它是Spring AOP框架在程序执行过程中触发的一些代码。

Spring切面可以执行一下五种类型的通知:

before(前置通知)：在一个方法之前执行的通知。

after(最终通知)：当某连接点退出的时候执行的通知（不论是正常返回还是异常退出）。

after-returning(后置通知)：在某连接点正常完成后执行的通知。

after-throwing(异常通知)：在方法抛出异常退出时执行的通知。

around(环绕通知)：在方法调用前后触发的通知。

**88、切入点(Pointcut)**

切入点是一个或一组连接点，通知将在这些位置执行。可以通过表达式或匹配的方式指明切入点。

**89、什么是引入？**

引入允许我们在已有的类上添加新的方法或属性。

**90、什么是目标对象？**

被一个或者多个切面所通知的对象。它通常是一个代理对象。也被称做被通知（advised）对象。

**91、什么是代理？**

代理是将通知应用到目标对象后创建的对象。从客户端的角度看，代理对象和目标对象是一样的。

**92、有几种不同类型的自动代理？**

BeanNameAutoProxyCreator：bean名称自动代理创建器

DefaultAdvisorAutoProxyCreator：默认通知者自动代理创建器

Metadata autoproxying：元数据自动代理

**93、什么是织入？什么是织入应用的不同点？**

织入是将切面和其他应用类型或对象连接起来创建一个通知对象的过程。织入可以在编译、加载或运行时完成。

**94、解释基于XML Schema方式的切面实现**

在这种情况下，切面由使用XML文件配置的类实现。

**95、解释基于注解方式(基于@AspectJ)的切面实现**

在这种情况下(基于@AspectJ的实现)，指的是切面的对应的类使用Java 5注解的声明方式。

# SpringMVC的面试题部分

**1、讲下SpringMvc和Struts1,Struts2的比较的优势**?

性能上Struts1>SpringMvc>Struts2 开发速度上SpringMvc和Struts2差不多,比Struts1要高

**2、讲下SpringMvc的核心入口类是什么,Struts1,Struts2的分别是什么?**

SpringMvc的是DispatchServlet,Struts1的是ActionServlet,Struts2的是StrutsPrepareAndExecuteFilter

**3、SpringMvc的控制器是不是单例模式,如果是,有什么问题,怎么解决**?

是单例模式,所以在多线程访问的时候有线程安全问题,不要用同步,会影响性能的,解决方案是在控制器里面不能写字段

**4、SpingMvc中的控制器的注解一般用那个,有没有别的注解可以替代**?

一般用@Controller注解,表示是表现层,不能用用别的注解代替.

**5、@RequestMapping注解用在类上面有什么作用?**

用于类上，表示类中的所有响应请求的方法都是以该地址作为父路径。

**6、怎么样把某个请求映射到特定的方法**

上面直接在方法上面加上注解@RequestMapping,并且在这个注解里面写上要拦截的路径

**7、如果在拦截请求中,我想拦截get方式提交的方法,怎么配置**?

可以在@RequestMapping注解里面加上method=RequestMethod.GET

**8、如果在拦截请求中,我想拦截提交参数中包含"type=test"字符串,怎么配置**?

可以在@RequestMapping注解里面加上params="type=test"

**9、我想在拦截的方法里面得到从前台传入的参数,怎么得到**?

直接在形参里面声明这个参数就可以,但必须名字和传过来的参数一样

**10、如果前台有很多个参数传入,并且这些参数都是一个对象的,那么怎么样快速得到这个对象?**

直接在方法中声明这个对象,SpringMvc就自动会把属性赋值到这个对象里面

**11、怎么样在方法里面得到Request,或者Session?**

直接在方法的形参中声明request,SpringMvc就自动把request对象传入

**12、SpringMvc中函数的返回值是什么**?

返回值可以有很多类型,有String, ModelAndView,List,Set等,一般用String比较好,如果是AJAX请求,返回的可以是一个集合

**13、SpringMvc怎么处理返回值的**?

SpringMvc根据配置文件中InternalResourceViewResolver(内部资源视图解析器)的前缀和后缀,用前缀+返回值+后缀组成完整的返回值

**14、SpringMVC怎么样设定重定向和转发的**?

在返回值前面加"forward:"就可以让结果转发,譬如"forward:user.do?name=method4" 在返回值前面加"redirect:"就可以让返回值重定向,"redirect:http://www.uu456.com"

**15、SpringMvc用什么对象从后台向前台传递数据的**?

通过ModelMap对象,可以在这个对象里面用addAttribute()方法,把对象加到里面,前台就可以通过el表达式拿到

**16、SpringMvc中有个类把视图和数据都合并的一起的,叫什么**?

ModelAndView

**17、怎么样把数据放入Session里面**?

可以声明一个request,或者session先拿到session,然后就可以放入数据,或者可以在类上面加上@SessionAttributes注解,里面包含的字符串就是要放入session里面的key

**18、SpringMvc怎么和AJAX相互调用的?**

通过Jackson框架就可以把Java里面的对象直接转化成Js可以识别的Json对象 具体步骤如下 :1.加入Jackson.jar2.在配置文件中配置json的映射3.在接受Ajax方法里面可以直接返回Object,List等,但方法前面要加上@ResponseBody注解

**19、当一个方法向AJAX返回特殊对象,譬如Object,List等,需要做什么处理?**

要加上@ResponseBody注解

**20、SpringMvc里面拦截器是怎么写的**?

有两种写法,一种是实现接口,另外一种是继承适配器类,然后在SpringMvc的配置文件中配置拦截器即可:<!-- 配置SpringMvc的拦截器 --> <mvc:interceptors> <!-- 配置一个拦截器的Bean就可以了 默认是对所有请求都拦截 --><bean id="myInterceptor" class="com.et.action.MyHandlerInterceptor"></bean><!-- 只针对部分请求拦截 --> <mvc:interceptor> <mvc:mapping path="/modelMap.do" /> <bean class="com.et.action.MyHandlerInterceptorAdapter" /> </mvc:interceptor> </mvc:interceptors>

**21、讲下SpringMvc的执行流程系统启动的时候根据配置文件创建spring的容器**?

首先是发送http请求到核心控制器DispatcherServlet，spring容器通过映射器去寻找业务控制器，使用适配器找到相应的业务类，在进业务类时进行数据封装，在封装前可能会涉及到类型转换，执行完业务类后使用ModelAndView进行视图转发，数据放在model中，用map传递数据进行页面显示。

# MyBatis的面试题部分

**1、讲下MyBatis和Hibernate的区别?**

MyBatis是JDBC的轻量级封装,把Sql和java代码独立出来,性能相对比较高,写SQL语句相对于比较灵活,并且容易调试,一般用在大型项目中.Hibernate是JDBC的重量级封装,开发速度比较快,但是性能比较低,调试不方便,一般适合对进度要求的比较高的中小型项目

**2、什么是MyBatis的接口绑定,有什么好处**?

接口映射就是在IBatis中任意定义接口,然后把接口里面的方法和SQL语句绑定,我们直接调用接口方法就可以,这样比起原来了SqlSession提供的方法我们可以有更加灵活的选择和设置.

**3、接口绑定有几种实现方式,分别是怎么实现的?**

接口绑定有两种实现方式,一种是通过注解绑定,就是在接口的方法上面加上@Select@Update等注解里面包含Sql语句来绑定,另外一种就是通过xml里面写SQL来绑定,在这种情况下,要指定xml映射文件里面的namespace必须为接口的全路径名.

**4、什么情况下用注解绑定,什么情况下用xml绑定?**

当Sql语句比较简单时候,用注解绑定,当SQL语句比较复杂时候,用xml绑定,一般用xml绑定的比较多

**5、MyBatis实现一对一有几种方式?具体怎么操作的**?

有联合查询和嵌套查询,联合查询是几个表联合查询,只查询一次,通过在resultMap里面配置association节点配置一对一的类就可以完成;嵌套查询是先查一个表,根据这个表里面的结果的外键id,去再另外一个表里面查询数据,也是通过association配置,但另外一个表的查询通过select属性配置

**6、如果要查询的表名和返回的实体Bean对象不一致,那你是怎么处理的?**

在MyBatis里面最主要最灵活的的一个映射对象的ResultMap,在它里面可以映射键值对, 默认里面有id节点,result节点,它可以映射表里面的列名和对象里面的字段名. 并且在一对一,一对多的情况下结果集也一定要用ResultMap

**7、MyBatis里面的动态Sql是怎么设定的?用什么语法?**

MyBatis里面的动态Sql一般是通过if节点来实现,通过OGNL语法来实现,但是如果要写的完整,必须配合where,trim节点,where节点是判断包含节点有内容就插入where,否则不插入,trim节点是用来判断如果动态语句是以and 或or开始,那么会自动把这个and或者or取掉

**8、MyBatis在核心处理类叫什么?**

MyBatis里面的核心处理类叫做SqlSession

**9、IBatis和MyBatis在细节上的不同有哪些?**

在sql里面变量命名有原来的#变量# 变成了#{变量} 原来的$变量$变成了${变量}, 原来在sql节点里面的class都换名字交type原来的queryForObject queryForList 变成了selectOne selectList原来的别名设置在映射文件里面放在了核心配置文件里

**10、讲下MyBatis的缓存?**

MyBatis的缓存分为一级缓存和二级缓存,一级缓存放在session里面,默认就有,二级缓存放在它的命名空间里,默认是打开的,使用二级缓存属性类需要实现Serializable序列化接口(可用来保存对象的状态),可在它的映射文件中<cache/>

**11、MyBatis(IBatis)的好处是什么?**

ibatis把sql语句从Java源程序中独立出来，放在单独的XML文件中编写，给程序的维护带来了很大便利。ibatis封装了底层JDBC API的调用细节，并能自动将结果集转换成Java Bean对象，大大简化了Java数据库编程的重复工作。因为Ibatis需要程序员自己去编写sql语句，程序员可以结合数据库自身的特点灵活控制sql语句，因此能够实现比hibernate等全自动orm框架更高的查询效率，能够完成复杂查询。