spring 题库

1.谈谈你对 MVC 的理解：

解析：MVC 是一种设计模式,是将输入，输出，控制进行了分离。

M：(mode)即模型层：用于对数据的持久化操作，也可以说是表示业务数据和业务处理的，一个模型层可以为多个视图层提供数据，提高了重用性。

V:(view)即：视图层，是用于展示的，也就是说想用户显示相关数据和接受用户输入数据。

C：control 即控制层，是用于连接模型层和视图层的中间层，可以很好的解决耦合问题。控制层将视图层接受到的数据交给控制层，有控制层调用模型层来完成相关处理，模型层处理完后，然后通过控制层想展示层显示结果。

2.Struts1.x 和 Struts2.x 的区别？

解析：

1）Struts1.x 是由 W3C 组织提供的框架，Struts2.x 原名是 webwork，后有 W3C 收购而该名

2）Struts1.x 的提交路径是*.do ;Struts2.x 提交路径是*.action

3）Struts1.x 中统一管理提交路径的核心类是 ActionServlet,Struts2.x 中统一管理提交路径的是核心是 Filter

4）Struts1.x 中有 ActionForm Struts2.x 中没有 ActionForm 只有 Action

5）Struts1.x 要依赖 Servlet API,每个 excute 方法都会带上 HttpSerrvletRequest 和 response 将其进行了暴露，Struts2.x 中不依赖于 Servlet,这样可以不必要的去放访问这连个，当然也提供了方法来得到 request 和 response，提供的类为 ServletActionContext 可以去获取 request 和 response

3.项目中为什么要使用 SSH?

解析：SSH 分别代表 Struts Spring Hibernate，

首先如果使用了 Struts，那么 Struts 是 MVC 设计模式的一种很好的体现，会将这个应用程序的输入，输出，控制进行很好的分离，使得开发者 更多的关注业务逻辑的实现，第二点在于使用 Struts 后，Struts 提供了大量的 taglib，如果很好的使用这些标签库可以很大的提高开发效率。

其次 Hibernate 是一个很好 ORMaping 框架，可以通过操作类来改变数据库中表，并且其完全自动化的操作很好的节省了开发的时间，不像 JDBC 方式中采用 PreparedStament 那样一个一个 set get 那样处理过于冗余，所以如果不是对性能很高的要求，使用 hibernate 是个不错的选择。

最后 Spring,主要体现在 IOC 和 AOP；即控制反转和面向切面的编程，利用 IOC 和很容易为实现类中注入 bean 对象；利用 AOP 可以很容易的实现事务处理方面的操作。

综合而言:SSH 使用与否取决于项目和公司的进度而定。

4.Struts 和 Spring 是如何整合的？

解析：Struts 有 Strust1.x 版本和 Struts2.x 版本，两个结合 Spring 的方式有点不同；

如果是 Struts1.3 结合 Spring3.0，那么就需要在 Struts 的核心配置文件 struts-config.xml 中进行一些配置：

（1）需要加载 IOC 容器，也就说需要一个 ContextLoadPlugIn,即需要下面的配置：

```
<plug-inclassName="org.springframework.web.struts.ContextLoaderPlugIn"><set-property  properrty="ContextConfigLocation" value="/WEB-INF/classes/application-context.xml">
</set-property>
</plug-in>
```

（2）需要替换委托处理器，在没有用 spring 的情况下，action 的创建是由 RequestProcessor 来处理完成的，而如果用了 Spring 后，应该叫 Spring 来管理这个 action 的创建，所以需要一个处理器替换即配置：

```
<controllerprocessName="org.springframework.web.struts.DelegatingRequestProcessor"></controller>
```

如果是struts2.x去结合spring,也要完成类似的操作,只不过是写在了web.xml而不是struts2的核心文件struts.xml中，需要的做的工作仍然是：

1.加载IOC容器：只不过不是plug-in 插件而是：

```
<context-param>

     <param-name>ContextConfigLocation</param-name>

   <param-value>/WEB-INF/classes/appliactionContext.xml</param-value>

</context-param>
```

2.替换委托处理器，变成了监听器

```
<listener>

   <listener-class>org.springframework.web.context.ContextLoaderListe
rner</listener-class>

</listener>
```

5.Struts如何实现国际化？

解析：Struts1.x中实现国际化靠的是bean标签中的<bean-message key="" />标签来实现的，

其中key的内容对应国际化中不同的资源文件中的key.

Struts2.x实现国际化 同样首先需要定义多个同文件名但不同的资源文件。然后依靠的是Struts标签库中的:<s:i18n name=""></s:i18n> 和<s:text name=""></s:text>实现。

i18n 中name对应的资源文件名　　text中的name对应key值。

6.logic的标签有那几个？

分为条件 循环 跳转

条件如：logic:present　循环如：<logic:iterate name="" id="" ></logic:iterate>　跳转：logic:forward

7.Struts 中的 action 是单实例还是多实例的？

解析：单实例的，当多个用户去访问一个请求的时候，在服务器内存中只有一个与之对应的 action 对象。

因为当服务器加载 strtus 配置的时候就会创建一个 action，那么以后的没发送一个请求过来，服务器都会去检索是否存在一个 action 与此请求对应，如果有就会直接使用这个 action，而不会再去创建新的 action，所以是单实例的。

8.DispatchAction 是用什么技术实现的？

解析：DispatchAction 是 Action 的一个子类，主要是用来实现分发的，即一个 action 可以处理多个功能。普通的 Action，在处理业务的时候只能写在 execute 方 法中，但是有了分发的 DispatchAction，相同的业务就可以放到同一个 action 中了，便于管理，实现的技术仍然是 action 的技术，但多 传递了一个参数即 status 状态参数。

10Sturts2 和 Struts1. 的不同点主要在？

解析：不同主要在处理的核心类：Struts1.x 和核心处理类是 ActionServlet Struts2.x 的核心类主要是 Filter;其它的不同点，找前面的回答即可。

11.Struts 的工作流程？

解析：

Struts1.x 的工作流程：

.jsp 提交---》根据路径.do 找打对应的 ActionServlet,ActionServlet 在初始化的时候会去加载核心配置文件 struts- config.xml，然后反射创建 ActionForm 和 Action ，首先进入的是 ActionForm,ActionForm 负责接收参数，

和参数的基本验证，当参数验证未通过的时候，会根据 return errors,找到配置中 input,对应的路径，跳转到错误页。当验证通过会进入 action，执行逻辑判断和页面的跳转，依靠的是 mapping.findForward(),对应于配置中的<forward>路径。

Struts2.x 的工作流程：

.jsp 页面提交--》提交的路径是/，但路径的后缀为*.action-->找到对应的 Filter-->过滤器会进行路径分 配，符合的进入 action 中，在 action 中主要有 validate 来进行参数的验证,这个 action 必须去继承一个类 ActionSupport 类，验证不通过同样找到 input 来进行错误页跳转,通过进入 execute 方法继续操作，跳转路径通过<result>来配置。

12.什么是 Hibernate 的并发机制？怎么去处理并发问题?

解析：hibernate 的并发机制主要体现在两个方面：

1）hibernate 的 session 是非线程安全的，对于单个请求，单个回话，单元的操作完成之后，这个 session 就丢弃了，如果将 Hibernate 的 session 进行共享，那么就可以实行并发了。

2）多个事务对同一资源的访问也可以实现并发，但这样会带来资源的错乱，因此要解决资源错乱，即处理并发的问题：

1）设置事务的隔离级别：

serializabled 串行化 最高

Repeatedable 可重复读

Commit reader 已提交的读

Uncommit reader 未提交的读

2）加锁 有 乐观锁 和悲观锁

乐观硕的性能要比悲观锁的并发并能高

13.Hibernate 与 JDBC 的联系？

解析：简单的来讲：hibernate = jdbc 的封装+xml 文件。因为 hibernate 实质是将 jdbc 进行了轻量级的封装,包括 jdbc 的数据库连接已经 sql 语句 CRUD 封装成了 HQL 语句，对数据库表的查询该成了对类的查询。

14. Hibernate 与 Spring 的联系？

解析：hibernate 于 spring 的联系反映在 hibernate 的一些对象交给了 spring 来管理创建，比如 hibernate 的数据库连接 类，不需要单独去建立，有 spring 来注入完成，以及想 dao 层的 hibernateTemplate ，sessionFactory 都有 spring 来完成管理，即依赖注入的关系。

15.Hibernate 自带的分页机制是什么？如果不使用 Hibernate 自带的分页，则采用什么方式分页？

解析：hibernate 的自带分页机制的实现依靠的是在获取到 session 对象以后，创建出 query 对象，然后依靠 query.setFirstResult() 来设置显示的第一条 query.setMaxResults() 来最后一条数据。如果不依赖这种分页，还可以采用 sql 的方式实现分页，即执行 createSqlQuery 也可以实现分页。

16.update（）和 saveOrUpdate（）的区别？

解析：update 和 savaOrUpdate 都是 session 对 pojo 的操作，但不同点在于，update 修改的数据必须在数据库存在，savaOrupdate 可以在数据库没有这条数据，没有的是候会执行保存操作。

17.hibernate 的三种状态之间如何转换 ？

解析：hibernate 有三种状态分别是：瞬时态 持久态 游离态

之间的转换：

瞬时态 到 持久态：执行 save（）操作

持久态 到 瞬时态： 执行 delete()操作

持久态 到 游离态： session 关闭

游离态 到 持久态： upate() 方法

18.hibernate 拒绝连接、服务器崩溃的原因？最少写 5 个

解析：1）数据库实例没有开

2）服务器没有开

3）网络连接错误

4）驱动包没有加

5）url 或其它配置出错

19.简要介绍 Hibernate?

解析：hibernate 是一个非常优秀的 ORMapping 的映射框架，对 jdbc 进行了轻量级的封装，是开发者可以通过操作对象来操作数据库，在 hibernate 中主要的接口有 5 个分别是

Seesion SessionFactory Configruation Transaction Query

Seesion:主要是用来完成对数据库中的数据进行 CRUD 操作的。

SeesionFactory: 是储存数据库数据源，是一个重量级对象，一般在一个数据库就只会创建一个。

Configruation:是用于读取配置文件和映射文件的类，用于创建 SeesionFactory

Query：是用于执行 hql 语句的核心对象

20Hibernate 中主键的生成方式有？

1）assign ： 手工来配置主键

2）native： 自动生成主键，使用于 mysql 类似的数据库

3）sequence: 自动生成主键，使用于 Oracle 类似的数据库

4）increment: 自动通过程序来生成主键，适用于所有的数据库

5） uuid : 随机生成 32 位 16 进制的主键

21.hibernate 的缓存机制？

解析：hibernate 的缓存分为一级缓存和二级缓存；一级缓存也就是我们常说的 seession 缓存，它的生命周期伴随着 session 的消失 而消失，而在程序中 session 肯定是要关闭的，所以这种缓存很少使用。二级缓存也称为 seesionFactory 缓存，也就是说伴随着 sessionFactory 来消失，而一般在 sessionFactory 只有一个所以不会随意关闭，所以有用，但默认情况下二级缓存是关闭的需要手工 打开，并且还需要加上一些配置，常用的二级缓存包有两种一种是 Ehcacahe,

另一种是 oscache,我用过的是 ehcache,要加入二级缓存需要做如下的配置：

1）在添加 show_sql 的位置加入三个东西，这里我就不写了

2）在需要二级缓存的对象的映射文件中加入<cache value="">因为并不是所有的 pojo 都要二级缓存，否则服务器压力太大，所以需要的地方才加。这里就涉及到了缓存策略：

主要有 read-only:只读，即只能读取数据，而不能对数据进行修改，一般用在不经常改动的地方。

read-write:可读写，这种灵活性虽然很大，但效率性能上其实不是很高

nostritct-read-wirte:不严格的读写,这种是只你修改写入数据后，并不会立即更新，需要过一段时间，使用类似新闻这种，效率性能得到了提高，但是不精确

traction：事务处理的，已经没用，被前面的替代了

3）在加入了缓存策略以后，然后拷贝一个 ehcache 的 xml 文件放到 src 下面

4）需要用到 query.setCacheable(true)　开启二级缓存。

22.Spring 主要用来做什么？

解析：spring 的主要功能就是：IOC 和 AOP

IOC：即控制反转，也就是通过一个配置文件在该配置文件中描述了对象和对象的属性信息，通过反射的方法来产生这个对象，并且通过 IOC 方式，如果 配置文件中对象的属性带有值，也会一块为对象附上值，而这些值可以是基本数据类型也可以是引用数据类型，因此 IOC 也称为依赖注入。可以替代工厂类

AOP：面向切面的编程，在 SSH 中主要用来是实现动态的代理设计，也就是说将一系列相同的操作封装到一个规则类，在实现这个实现类的时候，先执行这个规则类的初始化操作，在回收进行相应的收尾操作，可以简化 ServiceImpl 操作。

23.spring 中的哪个类的哪个方法可用于获取 bean

解析：采用 ApplicationContext ac = ClassPathXmlApplicationContext("applicationContext.xml");

ac.getBean()的方法可以获取 bean 对象。

24.spring 中可以用注入获得属性值，还有其他的方式吗？

解析：也可以采用读取配置文件的方式获取属性值

25.spring 在项目中如何充当粘合剂 ？

解析：比如在 web 层中我们只需要定义业务层的接口，但具体的实现类我交 spring 来依赖注入，这样可以很好的解决耦合的问题，另外 spring 的事务管理可以很好的对 hibernate 进行事务配置。这些都可以看出 spring 在项目中起着一种管理作用。

26.spring 的事务如何配置 ？

解析：这个实际就是 AOP 怎么写的，这个太难了，等面试的前一天路上看一下就可以了。

27.在 Spring 中，transaction 事务处理的实现有那两种方式？

解析：1.代码控制事务：即在程序中引入一个新的模板类，在这类中封装事务管理功能。

2.参数配置控制事务：在 applicationContext.xml 文件中增加一个事务代理配置

28.在 Spring 框架中，如何解决 Web 页面乱码问题？

解析：如果在在 spring 框架中解决乱码的问题，可以采用使用 spring 自带的一个 CharacterEncodingFilter 这个类可以实现，需要在 web.xml 的 Filter 处进行配置。

29.在 Struts 中，如何实现防止表单的重复提交操作？

解析：采用的 Token(指令牌)的方式，可以很好的解决重复提交的问题，它的原理是在你提交之前会在你 Pre 页面设置一个随机数的值，然后在你的 jsp 页面上也会有一个随机值，然后在提交的时候会取得去对比这两个值，如果相等就进行提交，提交后会改变 Pre 之中的那个随机值，但页面中的随机值没有 变，这样当你下次再重复提交的时候就提交不成功了。

30.Struts 的入口类？

解析：Struts 的入口类就是 ActionServlet,由这个核心类来控制所有路径分发。

31.写出你熟悉的开源框架以及各自的作用。

解析：常用的开源框架有：

Struts：统一管理提交的路径

Spring：实现 IOC 和起到解耦合的作用

Hibernate：实现对数据的持久化操作。

应用服务器有：Tomcat Jboss 等

32.请写出 spring 中 IOC 的三种实现机制

解析：1.setter getter 方法注入　2.构造方法注入　3 接口注入

33.Spring 框架的优点都有什么？

解析：1.Spring 是分层架构的，你可以选择你需要的层进行管理，而不需要的层可以不关心。

2.Spring 的 IOC 和 AOP 可以很大程度上简化代码。

3.Spring 可以降低组件之间的耦合性。

4.Spring 是开源免费的。

34. 描述一下 Spring 中实现 DI（Dependency Injection）的几种方式

解析：1.set get 注入，即在实现类中定义出该属性，并实现 set 方法，然后在 spring 配置中进行 set 的注入

2.构造方法的注入　　3.接口的注入

35. 简述你对 IoC（Inversion of Control）的理解

解析：所谓的 IOC，即控制反转，它的意思个人理解就是说，假设有类 A 和接口 B，现在在类 A 中我需要调用接口 B 中的方法，那么最简单的方法就是在类 A 中对接口 B 进行实例化出对象来，接口不能自己独立实例化，所以现在假设有子类 C，然后直接用子类帮父类进行实例化，然后就可以调用接口 B 中的方法了，但 是这样耦合性过大，依靠 IOC 的话，就在类 A 中只定义出接口 B，但具体的实现类，是在在 spring 的配置中，通过注入的方法来为接口 B 将 C 注入成 B 的实 例。这就是 IOC 依赖注入

36. Spring 对多种 ORM 框架提供了很好的支持，简单描述在 Spring 中使用 Hibernate 的方法。

解析：比如 SeesionFactory 的实现就是依靠的是注入方式和 spring 提供的方式，还有在 DAO 的实现类的时候会继承 HibernateDAOSupport，然后获得 super.getHiberanteTempalte() 这个核心对象，都体现了 spring 对 Hibernate 提供了很好的支持。

37. 如何在 Spring 的 applicationContext.xml 里面使用 JNDI 而不是 datasource?

解析：可以采用 spring 中提供的一个类：如下：

```
<bean id="dataSource"
class="org.springframework.jndi.JndiObjectFactoryBean">

    <property name="jndiName">

      <value>XXX</value>    具体的值记不住就算了
```

```
    </property>

</bean>
```

38. Spring 里面如何配置数据库驱动？

解析：

```
    <bean id="dataSource"
class="org.apache.commons.jdbc.BaseDataSoruce">

            <property name="driveClassName"
vlaue="org.gjt.mm.mysql.Driver">

            <property
name="url"   value="jdbc:mysql://localhost:3306/ajax">

          <prorperty name="username" value="root" >

            <property name="password" value="tiger">

  </bean>
```

39. Spring 里面 applicationContext.xml 文件能不能改成其他文件名？

解析：可以的，

对于 struts1.x 而言它的 DelegtaingRequestProcessor 默认情况下是会去加载 applicationContext.xml,但是我们可以通过设置来改变设置 ContextConfigLoaction 来改变其位置。

如下：

```
<plug-in
className="org.springframework.web.struts.ContextLoaderPlugIn">

    <set-property property="contextConfigLoacation"
value="/WEB-INF/classes/applicationContext-*.xml">

</plug-in>
```

对于Sruts2.而言默认情况下它的ContextLoaderListener,是去WEB-INF下面找，但可以通过

<context-param>来改变

<cotext-param>

<param-name>contextConfigLocation</param-name>

<param-value>/WEB-INF/classes/applicationContext-*.xml</param-value>

</context-param>

<listener>

<listener-class>org.springframework.web.context.ContextLoaderListenter</listener-class>

</listener>

40. 如何在 web 应用里面配置 spring?

解析：在你 web.xml 中加上 ContxtLoaderListener 即可，如下

<listener>

<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>

</listener>

41.Spring 里面如何定义 hibernate mapping？

解析，要在 spring 中定义映射文件，则需要在 spring 的核心配置文件中 applicationContext.xml

中设置：

<property name="mappingResources">

<list>

```
            <value>cn/mdj/vo/User.hbm.xml </value>
        <list>
    </property>
```

42. 解释一下 Dependency injection(DI,依赖注入)和 IOC(Inversion of control,控制反转)?

解析：DI 依赖注入应该是 IOC 控制反转的一个特例，因为控制反转的时候设置的值除基本数据类型外也可以是引用数据类型，而引用数据类型的时候，通常喜欢称为 DI。

43. spring 中的 BeanFactory 与 ApplicationContext 的作用和区别？

解析：BeanFactory 应该属于 ApplicationContext

功能上来讲：BeanFactory 主要是完成：对 bean 配置文件的读取，加载，管理，实例化，维护等操作。

ApplicationContext:除了可以完成 BeanFactory 能完成的操作外还可以完成其他的操作：

如果国际化，事务，资源访问等。

注：

常用的获取 ApplicationContext 的方法：
FileSystemXmlApplicationContext：从文件系统或者 url 指定的 xml 配置文件创建，参数为配置文件名或文件名数组
ClassPathXmlApplicationContext：从 classpath 的 xml 配置文件创建，可以从 jar 包中读取配置文件

44. 如何配置 spring+struts?

解析：前面有这里就不再说了

45. spring 的 jdbc 与传统的 jdbc 有什么区别，其核心类有那些?

解析：spring 的 jdbc,相对于传统 jdbc 而言，不需要在写 DatabaseConnection 数据库连接类，而且在 DAO 实现类的操作 上，传统的 JDBC 依靠的是 PreparedStatement 和 Connection 类在进行对数据库的操作,而在 Spring+jdbc 方式上操作的核心类是继承了 JdbcDAOSupport 的类,用的操作数据 库的核心对象是 super.getJdbcTemplate()

46. spring 的配置的主要标签是什么?有什么作用?

```
<beans>
<bean id=” ” class=” ” init=” ” destroy=” ” singleton=” ” >
<property name=” ” >
<value></value>
</property>
<property name=” ” >
<ref local></ref>
</property>
</bean>
</beans>
```

47. Spring 如何实现资源管理?

applicationContext.getResource(“classpath:文件名”):在 src 根目录下，在类路径下
applicationContext.getResource(“classpath:/chap01/文件名”): 以 src 根目录下的基准往下走。
applicationContext.getResource(“file:c:/a.properties”):在系统文件目录下。

48. Spring 中 ApplicationContext 的作用是什么？

解析：

beanFactory
国际化(getMesage)

资源管理:可以直接读取一个文件的内容(getResource)
加入 web 框架中(加入一个 servlet 或监听器)
事件处理

49. spring 中的核心类有那些，各有什么作用?

BeanFactory：产生一个新的实例，可以实现单例模式

ApplicationContext:提供框架的实现，包括 BeanFactory 的所有功能

50. 请介绍一下 Spring 框架中 Bean 的生命周期

解析：

1）Spring 通常通过配置文件定义 Bean

2）初始化 Bean。 通过方法或则通过接口

3）Bean 的调用 。通过 BeanFactoy 或则 ApplicationContext

4）Bean 的销毁 。通过方法或则接口来实现

51.spring 工作原理

解析：

1）springmvc 将所有请求都提交给 DispatchServlet,它委托其它模块来完成对请求的处理过程。

2）DispatchServlet 会去寻找一个或多个的 HandlerMapping,然后找到 Controller

3）DispatchServlet 请请求提交到目标 Controller

4）Controller 在处理完业务后会返回一个 ModerAndView

5）DispatchServlet 会去查找一个或多个 ViewResover 视图解析器，去找到 ModerAndView 的视图对象。

6）然后有这个视图对象来进行渲染返回客户端。

52.介绍一下 Spring 的事务管理：

解析：

事务就是对一系列的数据库操作（比如插入多条数据）进行统一的提交或回滚操作，如果插入成功，那么一起成功，如果中间有一条出现异常，那么回滚之前的所有操作。

spring 提供了几个关于事务处理的类：

TransactionDefinition //事务属性定义

TranscationStatus //代表了当前的事务，可以提交，回滚。

53.

综合：

一、Spring 基础知识及 IOC_选择题

1. 下面关于 spring 描述错误的是：（ ）

A　Spring 支持可插入的事务管理器，使事务划分更轻松，同时无需处理底层的问题。

B　Spring 事务管理的通用抽象层还包括 JTA 策略和一个 JDBC DataSource。

C　与 JTA 或 EJB CMT 一样，Spring 的事务支持依赖于 Java EE 环境。

D　Spring 事务语义通过 AOP 应用于 POJO，通过 XML 或 Java SE 5 注释进行配置。

参考答案：C

2. 下面选项哪个不是 Spring 中接口注入的方式？（ ）

A 接口注入

B 构造子注入

C 设值注入

D 指针注入

参考答案：D

3. 下列关于 Spring 特性中 IoC 描述错误的是：（）

A IoC 就是指程序之间的关系由程序代码直接操控。

B 所谓“控制反转”,是指控制权由应用代码转到外部容器，控制权的转移，

C IoC 将控制创建的职责搬进了框架中；并把它从应用代码脱离开来

D 当使用 Spring 的 IoC 容器时只需指出组件需要的对象，在运行时 Spring 的 IoC 容器会根据 XML 配置数据提供给它。

参考答案：A

二、Spring 基础知识及 IOC_简答题

1. 请简述 Spring 的工作机制？

参考答案：Spring 的工作机制可从以下几点来描述：（1）spring mvc 请所有的请求都提交给 DispatcherServlet,它会委托应用系统的其他模块负责负责对请求进行真正的处理工作。 （2）DispatcherServlet 查询一个或多个 HandlerMapping,找到处理请求的 Controller。 （3）DispatcherServlet 请请求提交到目标 Controller （4）Controller 进行业务逻辑处理后，会返回一个 ModelAndView （5）Dispathcher 查询一个或多个 ViewResolver 视图解析器,找到 ModelAndView 对象指定的视图对象（6）视图对象负责渲 染返回给客户端。

2. 请回答你为什么用 Spring 的工作机制？

参考答案：Spring 能有效的组织你的中间层对象，不管你是否选择使用了 EJB。如果你仅仅使用了 strus 或其他为 j2ee 的 API 特性的 framework，spring 致力于解决剩下的问题。Spring 能够消除在许多工程中常见的对 singleton 的过多的使 用。这个是一个很大的问题，它降低了系统的可测试性和面向对象的程度。

通过把对接口编程而不是对类编程的代价几乎减少到没有，spring 能够促进良好的 变成习惯的养成。

3. 请简述 Spring 是什么？

参考答案：Spring 是一个轻型的容器，是 J2EE 规范的轻量级实现，是企业应用的“一站式”解决方案。其中的核心就是 bean 工厂，用以构造我们需要的 Model，spring 是非侵入式的，Spring 的应用中的对象不依赖于 Spring 的特定类。

4. 简述 spring 的组成？

参考答案：Spring 主要由以下六个部分组成① Spring Core：核心容器，BeanFactory 提供了组件生命周期的管理，组件的创建，装配，销毁等功能。SpringContext：ApplicationContext，扩展核心容器，提供事件处理、国际化等功能。它提供了一些企业级服务的功能，提供了 JNDI，EJB，RMI 的支持。② Spring AOP：提供切面支持。③ Spring DAO：提供事务支持，JDBC，DAO 支持。④ Spring ORM：对流行的 O/R Mapping 封装或支持。⑤Spring Web：提供 Web 应用上下文，对 Web 开发提供功能上的支持，如请求，表单，异常等。⑥ Spring Web MVC：全功能 MVC 框架，作用等同于 Struts。

5.简述 Spring 容器提供了哪些功能？

参考答案：Spring 容器提供了对对象的管理，如 Spring 容器负责生成、组装、销毁组件，，另外 Spring 容器还提供了对持久化的支持，对事务的支持。另外 Spring 容器提供了国际化等功能。

6. 在 Spring 中，bean 的注入有几种方式，各是什么？

参考答案：Spring 中，Bean 的注入有两中方式，分别是 Setter 注入和构造器注入。

7. 请简述：Spring bean 的作用域？

参考答案：在 spring2.0 之前 bean 只有 2 种作用域即：singleton(单例)、non-singleton（也称 prototype），Spring2.0 以后，增加了 session、request、global session 三种专用于 Web 应用程序上下文的 Bean。因此，默认情况下 Spring2.0 现在有五种类型的 Bean。当然，Spring2.0 对 Bean 的类型的设计进行了重构，并设计出灵活的 Bean 类型支持，理论上可以有无数多种类型的 Bean，用户可以根据自己的需要，增加新的 Bean 类 型，满足实际应用需求。

8. 请叙述设值注入的优点？

参考答案：设置注入的优点：（1）与传统的 JavaBean 的写法更相似，程序开发人员更容易了解和接受。通过 setter 方法设定依赖关系显得更加直观、自然。（2）对于有复杂的依赖关系，如果采用构造注入，会导致构造器过于臃肿，难以阅读。Spring 在创建 bean 实例时，需要同时实例化其依赖的全部实例，因而导致性能的下降，而使用设值注入能够解决这些问题。（3）尤其是某些属性可选的情况下，多参数的构 造器更加笨重。

9. 请叙述构造注入的优点？

参考答案：构造注入的优点：（1）可以在构造器中决定依赖关系的注入顺序，优先依赖的优先注入。（2）对于依赖关系无须 变化的 bean，构造注入更加有用处。因为没有 setter 方法，所有的依赖关系全部在构造器内设定，因此，无须担心后续的代码对依赖关系产生破坏。 （3）依赖关系只能在构造器中设定，则只有组建的创建者才能改变组建的依赖关系。对组建的调用者而言，组建内部的依赖关系完全透明，更符合高内聚的原则。

10. 说出 bean 工厂创建 bean 的三种方式？

参考答案:Bean 工厂创建 bean 的三种方式分别是：

（1）直接构造，需要一个 default constructor 和相应的 setters/getters 方法。

要注意的是,我们可以为 getter/setter 方法传递参数(用 preperties 标签),也可以为构造函数传递参数

（2）采用静态工厂方法，如:

<bean id="exampleBean"class="examples.ExampleBean2"

factory-method="createInstance"/>

这里要求 examples.ExampleBean2 类有个静态方法 createInstance

（3）非静态工厂方法

<bean id="myFactoryBean" class="..."> </bean>

<bean id="exampleBean"

  factory-bean="myFactoryBean" factory-method="createInstance"/>

在这里,必须没有 class 标签,factory-bean 是 BeanFactory,factory-method 是它的非静态方法，myFactoryBean 可以通过容器来管理和配置。

11. 请写出 bean 的生命周期的方法？

参考答案：（1）通过设置 bean 的 init-method 属性指定初始化的方法，他的限制是方法无法接受任何参数，方 法可以为 static。（2）实现 InitializingBean 接口的 afterPrioertiesSet（）方法。（3）销毁对象可以通过 disposableBean 的 destroy 的实现。

12. 请简述你对 IOC 的理解？

参考答案：IOC 即 Inversion of Control，就是反转控制，Ioc 将控制创建的职责搬进了框架之中，并把它从应用代码中分隔开来，使用 Ioc 容器则需要指出组件需要什么对象在运行时 容器会提供给它，容器是通过查看对象的参数表做到的，也可能根据配置数据如 xml。

13. 请回答：IoC 最大的好处是什么？

参考答案：IoC 最大的好处是降低了对象的耦合性，实现了应用的松散耦合。因为把对象生成放在了 XML 里定义,所以当我们需要换一个实现子类将会变成很简单（一般这样的对象都是现实于某种接口的），只要修改 XML 就可以了。

14. 简述 IoC 的类型？

参考答案：IOC 可以分为三种注入类型，分别是构造函数注入、属性注入和接口注入。Spring 主要支持构造函数注入和属性注入。

15. Spring 中依赖注入与传统编程之间的差别是什么？

参考答案：在传统的程序设计过程中，通常由调用者来创建被调用者的实例。但在 Spring 里，创建被调用者的工作不再由 调用者来完成，因此称为控制反转；创建被调用者实例的工作通常由 Spring 容器来完成，然后注入调用者，因此也称为依赖注入。依赖注入的主要作用是起到 解耦合的作用。

三、.AOP_简答题

1. 说出 Spring 的通知类型有哪些？

参考答案：Spring 的通知类型有(1) MethodBeforeAdvice　(2) AfterReturningAdvice　(3) MethodInterceptor (4)ThrowsAdvice

2. 谈谈目标对象实现接口与目标对象不实现接口有什么区别？

参考答案：目标对象实现接口与目标对象不实现接口主要有以下几点区别：

（1）如果目标对象实现了接口,默认采用 JDK 的动态代理机制实现 AOP

（2）如果目标对象实现了接口,可以强制 spring 采用 CGLIB 实现代理

（3） 如果目标对象没有实现接口,必须采用 CGLIB 实现代理,spring 会自动的在 CGLIB 和 JDK 动态代理之间切换

3. 请描述 JDK 动态代理和 CGLI 代理的区别？

参考答案：JDK 的动态代理只能对实现了接口的目标类进行代理，而不实现接口的类就不能使用 JDK 的动态代理 CGLIB 是针对类来实现代理，当没有实现接口的类需要代理时就需要通过 CGLIB 来实现代理了，他的原理是对指定的目标类生成一个子类，并覆盖其中方法实现增强，但是因为采用的是继承，所以不能对 finall 类进行继承。二者在某些特殊场合需混合使用

4. 简述 ProxyFactoryBean 的作用是什么？

参考答案：ProxyFactoryBean 的作用是依照配置信息，将切面应用到目标对象，生成动态代理对象。

5. 叙述 Spring 中的自动代理的原理？

参考答案：Spring 在生成代理对象的时候，默认情况下，会使用被代理对象的接口来生成代理对象。如果被代理对象没有实现接口，此时，Spring 会使用 CGLIB 生成代理对象，此时该代理对象是被代理对象的子类。

5. 写出创建代理对象需指定的三要素是什么？

参考答案：创建代理对象需要指定的三要素是：target：设定目标对象（只能是一个）；proxyInterfaces：设定代理接口（目标对象所实现的接口）；interceptorNames：设定拦截器的名字（各个 advice 或 advisor bean 的列表）

6. 写出代理的两种方式分别是什么？

参考答案：代理的两种方式是：静态代理和动态代理，其中静态代理针对每个具体类分别编写代理类；针对一个接口编写一个代理类。而动态代理针对一个方面编写一个 InvocationHandler，然后借用 JDK 反射包中的 Proxy 类为各种接口动态生成相应的代理

7. 请简述：什么是 AOP？

参考答案：将程序中的交叉业务逻辑提取出来，称之为切面。将这些切面动态织入到目标对象，然后生成一个代理对象的过程。

8. 简述 AOP 核心？

参考答案：AOP 核心主要包括以下内容：（1）Aspect（切面），（2）Joinpoint（连接点），（3）Advice（通知），（4）Pointcut（切入点），（5）Introduction（引入），（6）Weaving（织入），（7）Target（目标对象），（8）Proxy（代理对象）

9. 请叙述 AOP 事务的含义？

参考答案：Spring 中进行事务管理的通常方式是利用 AOP（面向切片编程）的方式，为普通 java 类封装事务控制，它是通过动态代理实现的，由于接口是延迟实例化的，spring 在这段时间内通过拦截器，加载事务切片。

四、Spring 对持久化的支持_简答题

1. 请叙述 Spring 对持久层支持所采用的策略？

参考答案：Spring 对持久层采取了很好的支持，这些支持策略主要有：(1)Spring 对持久层“不发明重复的轮 子”，即没有重新实现新的持久层方案，对现有持

久层方案做封装，更利于使用。（2）采用 DAO 模式。（3）提供了大量的模板类来简化编程 （HibernateDaoSupport，JdbcTemplate 等）（4）重新设计了一套完善的异常体系结构：① 类型丰富，细化异常类型。② 全都是运行时异常（RuntimeException）。

2. 请问 Spring 如何简化事务配置？

参考答案：pring 简化事务配置有两种方式：第一种方式就是使用 TransactionProxyFactoryBean 创建事务代理（通常事务代理以 Service 层为目标 bean）配置 hibernate 的事务管 理器，使用 HibernateTransactionManager 类，该类实现了 PlatformTransactionManager 接口,针对 hibernate 持久化连接的特定实现。第二种方式使用自动创建代理简化事务配置使用 BeanNameAutoProxyCreator 和 DefaultAdvisorAutoProxyCreator 创建代理时，并不一定是创建事务代理，关键在于传入的拦截器，如果传入事务拦截器，将可自 动生成事务代理

3. 请简述 Spring 的事务机制？

参考答案：Spring 对事务的支持很丰富，除了编程式的处理事务，Spring 还支持声明式事务。其次 Spring 使 用事务服务代理和事务管理器（如 HibernateTransactionManager）来支持事务服务。另外 Spring 对事务的边界多了一种嵌套事 务。

4．请回答：Spring API 中的 getCurrentSession()和 openSession()两个方法的区别？

参考答案：getCurrentSession()和 openSession()两个方法主要有两点的区别：（1）采用 getCurrentSession（）创建的 session 会绑定到当前线程中，而采用 openSession（）创建的 session 则不会。（2） 采用 getCurrentSession（）创建的 session 在 commit 或者 rollback 后会自动关闭，而采用 openSession 的方式 需要手动进行关闭。

5. 请叙述 Spring 中使用 Hibernate 事务的步骤？

参考答案：Spring 中使用 Hibernate 事务的步骤为：（1）配置数据源 （2）配置 sessionfactory （3）配置事务管理器 （4）创建事务服务代理

6. 请叙述关于 Spring 的声明式事务处理？

参考答案：Spring 声明式事务让我们从复杂的事务处理中得到解脱。使得我们再也无需要去处理获得连接、关闭连接、事 务提交和回滚等这些操作。再也无需要我们在与事务相关的方法中处理大量的 try…catch…finally 代码。我们在使用 Spring 声明式事务时， 有一个非常重要的概念就是事务属性。事务属性通常由事务的传播行为，事务的隔离级别，事务的超时值和事务只读标志组成。我们在进行事务划分时，需要进行事 务定义，也就是配置事务的属性。

7. 请叙述 Spring 的事务传播属性与隔离级别？

参考答案：在使用 Spring 时，大部分会用到他的声明式事务，简单的在配置文件中 进行一些规则配置，利用 Spring 的 AOP 功能就能轻松搞定事务问题；这里面就涉及到一个事务的传播属性问题 Propagation，它在 TransactionDefinition 接口中定义，以供 PlatfromTransactionManager 使用,PlatfromTransactionManager 是 spring 事务管理的核心接口。

在 TransactionDefinition 接口中定义了五个不同的事务隔离级 别，ISOLATION_DEFAULT 这是一个 PlatfromTransactionManager 默认的隔离级别，使用数据库默认的事务隔离级别.另外四个与 JDBC 的隔离级别相对 应，ISOLATION_READ_UNCOMMITTED 这是事务最低的隔离级别，它充许别外一个事务可以看到这个事务未提交的数据。这种隔离级别会产生脏读，不可重复读和幻像读。

五、Spring+Struts+Hibernate

1. Spring 对多种 ORM 框架提供了很好的支持，简单描述在 Spring 中使用 Hibernate 的方法，并结合事务管理？

参考答案：Hibernate 是最优秀的 ORM 框架，Spring 对其提供了很好的支持，那么在 Spring 中使用 Hibernate 时要：（1）为每一个 bean 写 hibernate 映射文件，配置 datasourse，hibernateDaoTemplate，sessionFactory，把 datasourse 和映射文件注入到 sessionFactory （2）每个 dao 都继承 spring 容器中提供的一个类 HibernateDaoSupport，为每个 dao 注入 hibernateDaoTemplate（3）在 dao 中使用 getHibernateDaoTemplate（）的方法。Spring 中可以把需要进 行事务控制的 Biz 注入到 transactionProxy，为 biz 方法配置 transactionAttribute

2．请比较一下 Spring framework 与 Struts？

参考答案：Struts 只是一个 MVC 框架（Framework），用于快速的开发 Java Web 应用。Struts 实现的重点在 C（Controller），包括 Actionservlet/RequestProcessor 和我们定制的 Action，也为 V（View）提供了一系列的标签（Custom Tag）。而 Spring 是一个轻型容器，其中核心是 bean 工厂（beanfactory），用以构造我们所需要的 Model。在此基础 上，spring 提供了 AOP（面向切面编程）的实现，用它来提供非管理环境下申明方式的事务、安全等服务；对 Bean 工厂的扩展 ApplicationContext 更加方便我们实现 j2ee 的应用；Dao/ORM 的实现方便我们进行数据库的开发；Web MVC 和 Spring Web 提供了 java Web 应用的框架或与其他流行的 Web 框架进行集成。两者进行结合在一起使用是最好的方式。

3. 请叙述编写业务逻辑的方法？

参考答案：继承 HibernateDaoSupport 类，使用 HibernateTemplate 来持久 化，HibernateTemplate 是 Hibernate Session 的轻量级封装。默认情况下运行期异常才会回滚（包括继承了 RuntimeException 子类），普通异常是不会滚的。编写业务逻辑方法 时，最好将异常一直向上抛出，在表示层（struts）处理。关于事务边界的设置，通常设置到业务层，不要添加到 Dao 上。

4. 在 Web 分层架构中业务层为什么都选择 Spring？

参考答案：因为 Service 层需要处理业务逻辑和交叉业务逻辑，处理事务，日志，安全等，而这些与 Spring 的 IoC 特性，AOP 等不谋而合。