**一、SSH框架**

SSH作为struts2+spring+hibernate的一个集成框架，是目前较流行的一种Web应用程序开源框架。

集成SSH框架的系统从职责上分为四层：表示层、业务逻辑层、数据持久层和域模块层，帮助开发人员在短期内搭建Web应用程序。其中使用Struts作为系统的整体基础架构，负责MVC的分离，控制业务跳转，利用Hibernate框架对持久层提供支持，Spring做管理，管理struts和hibernate。

a、Struts   
Struts是个表示层框架，负责界面展示，接收请求，分发请求。  
b、Hibernate   
Hibernate是个持久层框架，负责与数据库的交互。  
c、Spring   
Spring是一个业务层框架，也是一个整合框架，管理struts和hibernate。

**二、为什么要用Spring**

1.方便解耦，简化开发  
通过Spring提供的IoC容器，可以将对象之间的依赖关系交由Spring进行控制，避免硬编码所造成的过度程序耦合。

2.AOP编程的支持  
通过Spring提供的AOP功能，方便进行面向切面的编程，如性能监测、事务管理、日志记录等。

3.声明式事务的支持  
4.方便集成各种优秀框架

5.降低Java EE API的使用难度  
如对JDBC，JavaMail，远程调用等提供了简便封装

1.使用Spring的IOC容器，将对象之间的依赖关系交给Spring，降低组件之间的耦合性，让我们更专注于应用逻辑

2.可以提供众多服务，事务管理，WS等。

3.AOP的很好支持，方便面向切面编程。

4.对主流的框架提供了很好的集成支持，如Hibernate,Struts2,JPA等

5.Spring DI机制降低了业务对象替换的复杂性。

6.Spring属于低侵入，代码污染极低。

7.Spring的高度可开放性，并不强制依赖于Spring，开发者可以自由选择Spring部分或全部

**三、**

**四、介绍一下Spring的事务管理**   
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。这样可以防止出现脏数据，防止数据库数据出现问题。

开发中为了避免这种情况一般都会进行事务管理。Spring中也有自己的事务管理机制，一般是使用TransactionMananger进行管理，可以通过Spring的注入来完成此功能。

**五、什么是依赖注入，依赖注入的作用是什么?**   
IOC是一种思想，它能指导我们如何设计出松耦合、更优良的程序。传统应用程序都是由我们在类内部主动创建依赖对象，从而导致类与类之间高耦合，难于测试；有了IOC容器后，把创建和查找依赖对象的控制权交给了容器，由容器进行注入组合对象，所以对象与对象之间是松散耦合，这样也方便测试，利于功能复用，更重要的是使得程序的整个体系结构变得非常灵活。

依赖注入的作用：减少类间耦合度，避免用new来创建对象

**六、什么是AOP，AOP的作用是什么?**   
AOP，面向切面编程，就是把可重用的功能提取出来，然后将这些通用功能在合适的时候织入到应用程序中，比如事务管理、权限控制、日志记录、性能统计等。

AOP的作用  
AOP并没有帮助我们解决任何新的问题，它只是提供了一种更好的办法，能够用更少的工作量来解决现有的一些问题，使得系统更加健壮，可维护性更好。

**七、Spring中的BeanFactory与ApplicationContext的作用有哪些?**  
1、BeanFactory负责读取bean的配置文件，管理bean的加载、实例化，维护bean之间的依赖关系，负责bean的生命周期。   
2、ApplicationContext除了提供上述BeanFactory所能提供的功能之外，还提供了更完整的框架功能：   
a. 国际化支持  
b. 资源访问  
c. 事件传递

**八、连接池**  
c3p0、dbcp、proxool、jndi数据源、spring自带的数据源(DriverManagerDataSource)

<bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource"          
        destroy-method="close">   ….

<bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource"          
        destroy-method="close">  …

\*ComboPooledDataSource和BasicDataSource一样提供了一个用于关闭数据源的close()方法，这样我们就可以保证Spring容器关闭时数据源能够成功释放。

<bean id=*"dataSourceData"* class=*"org.springframework.jndi.JndiObjectFactoryBean"*>

<property name=*"jndiName"* value=*"java:comp/env/q114db"* />

</bean>

<bean id="dataSource"

      class="org.springframework.jdbc.datasource.DriverManagerDataSource">…

网上对三种连接池的评价如下：

C3P0比较耗费资源，效率方面可能要低一点。  
DBCP在实践中存在BUG，在某些种情会产生很多空连接不能释放，Hibernate3.0已经放弃了对其的支持。  
Proxool的负面评价较少，现在比较推荐它，而且它还提供即时监控连接池状态的功能，便于发现连接泄漏的情况。

**九、跨库事务**  
Atomikos、jotm