　struts2其实就是为我们封装了servlet，简化了jsp跳转的复杂操作，并且提供了易于编写的标签，可以快速开发view层的代码。

过去，我们用jsp和servlet搭配，实现展现时，大体的过程是：

　　1 jsp触发action

　　2 servlet接受action,交给后台class处理

　　3 后台class跳转到其他的jsp，实现数据展现

　　+++++++++现在有了struts2，实现过程变为++++++++++++

　　1 jsp出发action

　　2 struts2拦截请求，调用后台action

3 action返回结果，由不同的jsp展现数据

（1）视图（view）

在Struts中，视图层包含两个部分，JSP页面和ActionForm。

ActionForm封装了用户提交的表单信息，其实ActonForm本质上就是JavaBean，这些JavaBean中没有业务逻辑，只提供了所有属性的getter和setter方法，这些属性和用户表单中的输入项是一一对应的。在Struts中就是通过ActionForm把用户表单信息提交给控制器。

JSP页面是经典MVC中主要的视图组件，主要是信息显示和控制器处理结果显示的功能。

除了以上，struts还提供了一个强大的struts标签库，来帮助用户解决显示逻辑，并且利用ActonForm组件将信息传递到控制层。

（2）控制器（Controller）

在控制层，struts提供了一个控制器组件ActionServlet，它继承自HttpServlet，并重载了HttpServlet的doGet（），doPost（）方法，可以接受HTTP的响应，并进行转发，同时还提供了使用XML进行转发Mapping（映射）的功能。

（3）模型（Model）

模型表示状态和业务逻辑的处理，在一般的web应用程序中，用JavaBean或者EJB来实现系统的业务逻辑。在Struts中，struts提供Action对象，来管理业务逻辑的**调用**，帮助用户分离业务逻辑，也就是说struts本身不实现业务逻辑，但可以调用已完成的业务逻辑。

什么是spring框架

spring是J2EE应用程序框架，是轻量级的IoC和AOP的容器框架，主要是针对javaBean的生命周期进行管理的轻量级容器，可以单独使用，也可以和Struts框架，ibatis框架等组合使用。

Spring优点

轻量级的容器框架没有侵入性

使用IoC容器更加容易组合对象直接间关系，面向接口编程，降低耦合

Aop可以更加容易的进行功能扩展，遵循ocp开发原则

创建对象默认是单例的，不需要再使用单例模式进行处理

**什么是Hibernate？**

      Hibernate，翻译过来是冬眠的意思，正好现在已经进入秋季，世间万物开始准备冬眠了。其实对于对象来说就是持久化。

      持久化（Persistence），即把数据（如内存中的对象）保存到可永久保存的存储设备中（如磁盘）。持久化的主要应用是将内存中的对象存储在关系型的数据库中，当然也可以存储在磁盘文件中、XML数据文件中等等。

持久化是将程序数据在持久状态和瞬时状态间转换的机制。

JDBC就是一种持久化机制。文件IO也是一种持久化机制。

一、Hibernate是对JDBC进一步封装

原来没有使用Hiberante做持久层开发时，存在很多冗余，如：各种JDBC语句，connection的管理，所以出现了Hibernate把JDBC封装了一下，我们不用操作数据，直接操作它就行了。

二、我们再从分层的角度来看

我们知道非常典型的三层架构：表示层，业务层，还有持久层。Hiberante也是持久层的框架，而且持久层的框架还有很多，比如：IBatis，Nhibernate，JDO，OJB，EJB等等。

三、Hibernate是开源的一个ORM（对象关系映射）框架。

ORM，即Object-Relational Mapping，它的作用就是在关系型数据库和对象之间做了一个映射。从对象（Object）映射到关系（Relation），再从关系映射到对象。这样，我们在操作数据库的时候，不需要再去和复杂SQL打交道，只要像操作对象一样操作它就可以了（把关系数据库的字段在内存中映射成对象的属性）。

**context:component-scan**

----- 在xml配置了这个标签后，spring可以自动去扫描base-pack下面或者子包下面的java文件，如果扫描到有@Component @Controller@Service等这些注解的类，则把这些类注册为bean。

**@Autowired与@Resource**

------ 1、@Autowired与@Resource都可以用来装配bean. 都可以写在字段上,或写在setter方法上。

2、@Autowired默认按类型装配（这个注解是属业spring的），默认情况下必须要求依赖对象必须存在，如果要允许null 值，可以设置它的required属性为false，如：@Autowired(required=false) ，如果我们想使用名称装配可以结合@Qualifier注解进行使用，如下：

Java代码 收藏代码

@Autowired() @Qualifier("baseDao")

private BaseDao baseDao;

3、@Resource（这个注解属于J2EE的），默认安照名称进行装配，名称可以通过name属性进行指定，

如果没有指定name属性，当注解写在字段上时，默认取字段名进行按照名称查找，如果注解写在setter方法上默认取属性名进行装配。 当找不到与名称匹配的bean时才按照类型进行装配。但是需要注意的是，如果name属性一旦指定，就只会按照名称进行装配。

**@RequestMapping**

-------- RequestMapping是一个用来处理请求地址映射的注解，可用于类或方法上。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。

RequestMapping注解有六个属性，下面我们把她分成三类进行说明。

**1、 value， method；**

value：     指定请求的实际地址，指定的地址可以是URI Template 模式（后面将会说明）；

method：  指定请求的method类型， GET、POST、PUT、DELETE等

**2、 consumes，produces；**

consumes： 指定处理请求的提交内容类型（Content-Type），例如application/json, text/html;

produces:    指定返回的内容类型，仅当request请求头中的(Accept)类型中包含该指定类型才返回；

**3、 params，headers；**

params： 指定request中必须包含某些参数值是，才让该方法处理。

headers： 指定request中必须包含某些指定的header值，才能让该方法处理请求。

**@SuppressWarnings("")**

**----** 简介： java.lang.SuppressWarnings是J2SE 5.0中标准的Annotation之一。可以标注在类、字段、方法、参数、构造方法，以及局部变量上。  
作用：告诉编译器忽略指定的警告，不用在编译完成后出现警告信息。

@SuppressWarnings("unchecked")

告诉编译器忽略 unchecked 搜索警告信息，如使用List，ArrayList等未进行参数化产生的警告信息。

**@Service**

@Service是一个注解啊，告诉spring创建一个实现类的实例

括号里的参数理论上可以随便设置，因为这是你要实现的类的一个对象名而已，比如在一个Student实现类上加上@Service("student")，括号里面就是给实例化后的Student对象取个名字。。。  
这是在你一个接口有多个实现类的情况下区分实现类的方式。。。。  
比如Student实现了Person接口，在你的controller里面@Autowired Person时，假如这时Person还有另一个实现类User，为了确定实例化Student还是User， @Service括号里面的东西就有用了，  
@Autowired  
@Qualifier("student") 或 @Resource("student")  
private Person person;  
这样就确定实例化了一个Person指向一个Student对象。。。

**Struts2注解（常用）**

Namespace：指定命名空间。

ParentPackage：指定父包。

Result：提供了Action结果的映射。（一个结果的映射）

Results：“Result”注解列表

ResultPath：指定结果页面的基路径。

Action：指定Action的访问URL。

Actions：“Action”注解列表。

@Action(value="testAction",results={@Result(name="success",location="/success.jsp")})

public class testAction extends ActionSupport {

@Override

public String execute() throws Exception {

return SUCCESS;

}

}

这就相当于如下的xml配置

<result name="success">/success.jsp</result>

</action>