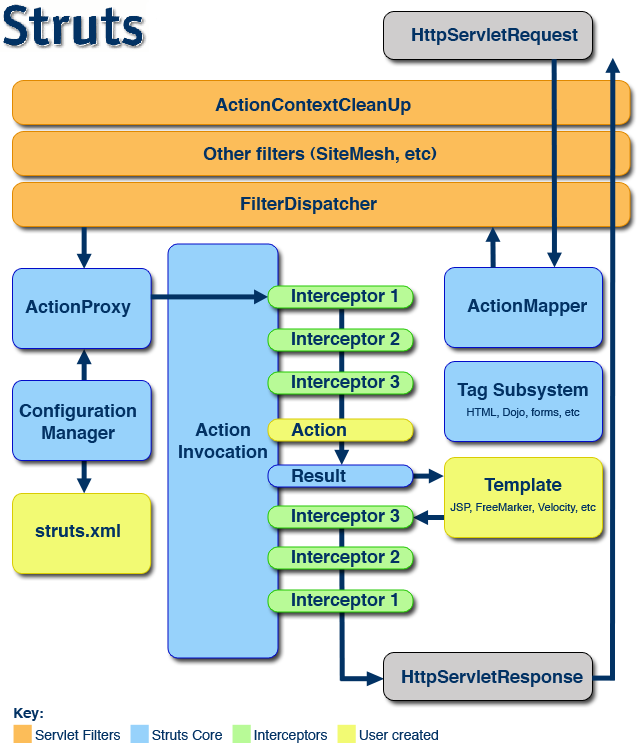
# Struts2面试题

## 1、struts2工作流程



Struts 2框架本身大致可以分为3个部分：

核心控制器FilterDispatcher、业务控制器Action和用户实现的企业业务逻辑组件。

核心控制器FilterDispatcher是Struts 2框架的基础，

包含了框架内部的控制流程和处理机制。

业务控制器Action和业务逻辑组件是需要用户来自己实现的。

用户在开发Action和业务逻辑组件的同时，还需要编写相关的配置文件，

供核心控制器FilterDispatcher来使用。

Struts 2的工作流程相对于Struts 1要简单，与WebWork框架基本相同，

所以说Struts 2是WebWork的升级版本。基本简要流程如下：

1 、客户端初始化一个指向Servlet容器的请求；

2、 这个请求经过一系列的过滤器（Filter）

（这些过滤器中有一个叫做ActionContextCleanUp的可选过滤器，

这个过滤器对于Struts2和其他框架的集成很有帮助，例如：SiteMesh Plugin）

3 、接着FilterDispatcher被调用，

FilterDispatcher询问ActionMapper来决定这个请是否需要调用某个Action

4、如果ActionMapper决定需要调用某个Action，

FilterDispatcher把请求的处理交给ActionProxy

5、ActionProxy通过Configuration Manager询问框架的配置文件，

找到需要调用的Action类

6、ActionProxy创建一个ActionInvocation的实例。

7、ActionInvocation实例使用命名模式来调用，

在调用Action的过程前后，涉及到相关拦截器（Intercepter）的调用。

8、一旦Action执行完毕，ActionInvocation负责根据struts.xml中的配置找到对应的返回结果 。返回结果通常是（但不总是，也可 能是另外的一个Action链）一个需要被表示的JSP或者FreeMarker的模版。 在表示的过程中可以使用Struts2 框架中继承的标签。 在这个过程中需要涉及到ActionMapper

9、响应的返回是通过我们在web.xml中配置的过滤器   
10、如果ActionContextCleanUp是当前使用的，则FilterDispatecher将不会清理sreadlocal ActionContext;如果ActionContextCleanUp不使用，则将会去清理sreadlocals。

## 2、说下Struts的设计模式

MVC模式: web应用程序启动时就会加载并初始化ActionServler。用户提交表单时，一个配置好的ActionForm对象被创建，并被填入表单相应的数据，ActionServler根据Struts-config.xml文件配置好的设置决定是否需要表单验证，如果需要就调用ActionForm的Validate（）验证后选择将请求发送到哪个Action，如果Action不存在，ActionServlet会先创建这个对象，然后调用Action的execute（）方法。Execute（）从ActionForm对象中获取数据，完成业务逻辑，返回一个ActionForward对象，ActionServlet再把客户请求转发给ActionForward对象指定的jsp组件，ActionForward对象指定的jsp生

成动态的网页，返回给客户。

## 3、拦截器和过滤器的区别

1、拦截器是基于java反射机制的，而过滤器是基于函数回调的。  
2、过滤器依赖于servlet容器，而拦截器不依赖于servlet容器。  
3、拦截器只能对Action请求起作用，而过滤器则可以对几乎所有请求起作用。  
4、拦截器可以访问Action上下文、值栈里的对象，而过滤器不能。  
5、在Action的生命周期中，拦截器可以多次调用，而过滤器只能在容器初始化时被调用一次。

## 4、struts1于struts2的比较

1、Action   类:   
    Struts1要求Action类继承一个抽象基类。Struts1的一个普遍问题是使用抽象类编程而不是接口。   
  Struts   2   Action类可以实现一个Action接口，也可实现其他接口，使可选和定制的服务成为可能。Struts2提供一个ActionSupport基类去 实现常用的接口。Action接口不是必须的，任何有execute标识的POJO对象都可以用作Struts2的Action对象。   
  
2、线程模式:   
    Struts1   Action是单例模式并且必须是线程安全的，因为仅有Action的一个实例来处理所有的请求。单例策略限制了Struts1   Action能作的事，并且要在开发时特别小心。Action资源必须是线程安全的或同步的。   
Struts2   Action对象为每一个请求产生一个实例，因此没有线程安全问题。（实际上，servlet容器给每个请求产生许多可丢弃的对象，并且不会导致性能和垃圾回收问题）   
  
3、Servlet   依赖:   
    Struts1   Action   依赖于Servlet   API   ,因为当一个Action被调用时HttpServletRequest   和   HttpServletResponse   被传递给execute方法。   
Struts   2   Action不依赖于容器，允许Action脱离容器单独被测试。如果需要，Struts2   Action仍然可以访问初始的request和response。但是，其他的元素减少或者消除了直接访问HttpServetRequest   和   HttpServletResponse的必要性。   
  
4、可测性:   
   测试Struts1   Action的一个主要问题是execute方法暴露了servlet   API（这使得测试要依赖于容器）。一个第三方扩展－－Struts   TestCase－－提供了一套Struts1的模拟对象（来进行测试）。   
Struts   2   Action可以通过初始化、设置属性、调用方法来测试，“依赖注入”支持也使测试更容易。   
  
5、捕获输入:   
   Struts1   使用ActionForm对象捕获输入。所有的ActionForm必须继承一个基类。因为其他JavaBean不能用作ActionForm，开发者经 常创建多余的类捕获输入。动态Bean（DynaBeans）可以作为创建传统ActionForm的选择，但是，开发者可能是在重新描述(创建)已经存 在的JavaBean（仍然会导致有冗余的javabean）。   
Struts   2直接使用Action属性作为输入属性，消除了对第二个输入对象的需求。输入属性可能是有自己(子)属性的rich对象类型。Action属性能够通过   web页面上的taglibs访问。Struts2也支持ActionForm模式。rich对象类型，包括业务对象，能够用作输入/输出对象。这种   ModelDriven   特性简化了taglib对POJO输入对象的引用。   
  
6、表达式语言：   
   Struts1   整合了JSTL，因此使用JSTL   EL。这种EL有基本对象图遍历，但是对集合和索引属性的支持很弱。   
Struts2可以使用JSTL，但是也支持一个更强大和灵活的表达式语言－－ "Object   Graph   Notation   Language "   (OGNL).   
  
7、绑定值到页面（view）:   
   Struts   1使用标准JSP机制把对象绑定到页面中来访问。   
   Struts   2   使用   "ValueStack "技术，使taglib能够访问值而不需要把你的页面（view）和对象绑定起来。ValueStack策略允许通过一系列名称相同但类型不同的属性重用页面（view）。   
  
8、类型转换：   
   Struts   1   ActionForm   属性通常都是String类型。Struts1使用Commons-Beanutils进行类型转换。每个类一个转换器，对每一个实例来说是不可配置的。   
  Struts2   使用OGNL进行类型转换。提供基本和常用对象的转换器。   
  
9、校验：   
   Struts   1支持在ActionForm的validate方法中手动校验，或者通过Commons   Validator的扩展来校验。同一个类可以有不同的校验内容，但不能校验子对象。   
   Struts2支持通过validate方法和XWork校验框架来进行校验。XWork校验框架使用为属性类类型定义的校验和内容校验，来支持chain校验子属性   
  
10、Action执行的控制：   
   Struts1支持每一个模块有单独的Request   Processors（生命周期），但是模块中的所有Action必须共享相同的生命周期。   
   Struts2支持通过拦截器堆栈（Interceptor   Stacks）为每一个Action创建不同的生命周期。堆栈能够根据需要和不同的Action一起使用。

## 为什么要使用Struts2

Struts2 是一个相当强大的Java Web开源框架，是一个基于POJO的Action的MVC Web框架。它基于当年的Webwork和XWork框架，继承其优点，同时做了相当的改进。  
1.Struts2基于MVC架构，框架结构清晰，开发流程一目了然，开发人员可以很好的掌控开发的过程。  
2使用OGNL进行参数传递。  
OGNL提供了在Struts2里访问各种作用域中的数据的简单方式，你可以方便的获取Request，Attribute，Application，Session，Parameters中的数据。大大简化了开发人员在获取这些数据时的代码量。  
3强大的拦截器  
Struts2 的拦截器是一个Action级别的AOP，Struts2中的许多特性都是通过拦截器来实现的，例如异常处理，文件上传，验证等。拦截器是可配置与重用的，可以将一些通用的功能如：登录验证，权限验证等置于拦截器中以完成一些Java Web项目中比较通用的功能。在我实现的的一Web项目中，就是使用Struts2的拦截器来完成了系统中的权限验证功能。  
4易于测试  
Struts2的Action都是简单的POJO，这样可以方便的对Struts2的Action编写测试用例，大大方便了5Java Web项目的测试。  
易于扩展的插件机制在Struts2添加扩展是一件愉快而轻松的事情，只需要将所需要的Jar包放到WEB-INF/lib文件夹中，在struts.xml中作一些简单的设置就可以实现扩展。  
6模块化管理  
Struts2已经把模块化作为了体系架构中的基本思想，可以通过三种方法来将应用程序模块化：将配置信息拆分成多个文件把自包含的应用模块创建为插件创建新的框架特性，即将与特定应用无关的新功能组织成插件，以添加到多个应用中去。  
7全局结果与声明式异常  
为应用程序添加全局的Result，和在配置文件中对异常进行处理，这样当处理过程中出现指定异常时，可以跳转到特定页面。  
他的如此之多的优点，是很多人比较的青睐，与spring ,Hibernate进行结合，组成了现在比较流行的ssh框架，当然每个公司都要自己的框架，也是ssh变异的产品。

## struts2有哪些优点？

1）在软件设计上Struts2的应用可以不依赖于Servlet API和struts API。 Struts2的这种设计属于无侵入式设计；

2）拦截器，实现如参数拦截注入等功能；

3）类型转换器，可以把特殊的请求参数转换成需要的类型；

4）多种表现层技术，如：JSP、freeMarker、Velocity等；

5）Struts2的输入校验可以对指定某个方法进行校验；

6）提供了全局范围、包范围和Action范围的国际化资源文件管理实现

## struts2是如何启动的？

struts2框架是通过Filter启动的，即StrutsPrepareAndExecuteFilter，此过滤器为struts2的核心过滤器；

StrutsPrepareAndExecuteFilter的init()方法中将会读取类路径下默认的配置文件struts.xml完成初始化操作。struts2读取到struts.xml的内容后，是将内容封装进javabean对象然后存放在内存中，以后用户的每次请求处理将使用内存中的数据，而不是每次请求都读取struts.xml文件。

## struts2框架的核心控制器是什么？它有什么作用？

1）Struts2框架的核心控制器是StrutsPrepareAndExecuteFilter。

2）作用：

 负责拦截由<url-pattern>/\*</url-pattern>指定的所有用户请求，当用户请求到达时，该Filter会过滤用户的请求。默认情况下，如果用户请求的路径

不带后缀或者后缀以.action结尾，这时请求将被转入struts2框架处理，否则struts2框架将略过该请求的处理。

可以通过常量"struts.action.extension"修改action的后缀，如：

<constant name="struts.action.extension" value="do"/>

如果用户需要指定多个请求后缀，则多个后缀之间以英文逗号（,）隔开。

<constant name="struts.action.extension" value="do,go"/>

## struts2配置文件的加载顺序？

struts.xml ——> struts.properties

常量可以在struts.xml或struts.properties中配置，如果在多个文件中配置了同一个常量，则后一个文件中配置的常量值会覆盖前面文件中配置的常量值.

struts.xml文件的作用：通知Struts2框架加载对应的Action资源

## struts2常量的修改方式？

常量可以在struts.xml或struts.properties中配置，两种配置方式如下：

1）在struts.xml文件中配置常量

<constant name="struts.action.extension" value="do"/>

2）在struts.properties中配置常量（struts.properties文件放置在src下）：

struts.action.extension=do

## struts2如何访问HttpServletRequest、HttpSession、ServletContext三个域对象？

方案一：

HttpServletRequest request =ServletActionContext.getRequest();

HttpServletResponse response =ServletActionContext.getResponse();

HttpSession  session=   request.getSession();

ServletContext servletContext=ServletActionContext.getServletContext();

方案二：

类 implements ServletRequestAware,ServletResponseAware，SessionAware，ServletContextAware

注意：框架自动传入对应的域对象

## struts2是如何管理action的？这种管理方式有什么好处？

struts2框架中使用包来管理Action，包的作用和java中的类包是非常类似的。

主要用于管理一组业务功能相关的action。在实际应用中，我们应该把一组业务功能相关的Action放在同一个包下。

## struts2中的默认包struts-default有什么作用？

1）struts-default包是由struts内置的，它定义了struts2内部的众多拦截器和Result类型，而Struts2很多核心的功能都是通过这些内置的拦截器实现，如：从请求中

把请求参数封装到action、文件上传和数据验证等等都是通过拦截器实现的。当包继承了struts-default包才能使用struts2为我们提供的这些功能。

2）struts-default包是在struts-default.xml中定义，struts-default.xml也是Struts2默认配置文件。 Struts2每次都会自动加载 struts-default.xml文件。

3）通常每个包都应该继承struts-default包。

## struts2如何对指定的方法进行验证？

1）validate()方法会校验action中所有与execute方法签名相同的方法；

2）要校验指定的方法通过重写validateXxx()方法实现， validateXxx()只会校验action中方法名为Xxx的方法。其中Xxx的第一个字母要大写；

3）当某个数据校验失败时，调用addFieldError()方法往系统的fieldErrors添加校验失败信息（为了使用addFieldError()方法，action可以继承ActionSupport）， 如果系统 的fieldErrors包含失败信息，struts2会将请求转发到名为input的result；

4）在input视图中可以通过<s:fielderror/>显示失败信息。

5）先执行validateXxxx()->validate()->如果出错了，会转发<result name="input"/>所指定的页面，如果不出错，会直接进行Action::execute()方法

## struts2默认能解决get和post提交方式的乱码问题吗？

不能。struts.i18n.encoding=UTF-8属性值只能解析POST提交下的乱码问题。

## 请你写出struts2中至少5个的默认拦截器？

fileUpload      提供文件上传功能

i18n            记录用户选择的locale

cookies         使用配置的name,value来是指cookies

checkbox        添加了checkbox自动处理代码，将没有选中的checkbox的内容设定为false，而html默认情况下不提交没有选中的checkbox。

chain           让前一个Action的属性可以被后一个Action访问，现在和chain类型的result（）结合使用。

alias           在不同请求之间将请求参数在不同名字件转换，请求内容不变

## 值栈ValueStack的原理与生命周期？

1）ValueStack贯穿整个 Action 的生命周期，保存在request域中，所以ValueStack和request的生命周期一样。当Struts2接受一个请求时，会迅速创建ActionContext，

ValueStack，action。然后把action存放进ValueStack，所以action的实例变量可以被OGNL访问。 请求来的时候，action、ValueStack的生命开始，请求结束，action、    ValueStack的生命结束；

2）action是多例的，和Servlet不一样，Servelt是单例的；

3）每个action的都有一个对应的值栈，值栈存放的数据类型是该action的实例，以及该action中的实例变量，Action对象默认保存在栈顶；

4）ValueStack本质上就是一个ArrayList；

5）关于ContextMap，Struts 会把下面这些映射压入 ContextMap 中：

parameters  :   该 Map 中包含当前请求的请求参数

request     :   该 Map 中包含当前 request 对象中的所有属性  session :该 Map 中包含当前 session 对象中的所有属性

application :该 Map 中包含当前 application 对象中的所有属性

attr:该 Map 按如下顺序来检索某个属性: request, session, application

6）使用OGNL访问值栈的内容时，不需要#号，而访问request、session、application、attr时，需要加#号；

7）注意： Struts2中，OGNL表达式需要配合Struts标签才可以使用。如：<s:property value="name"/>

8）在struts2配置文件中引用ognl表达式 ,引用值栈的值 ，此时使用的"$"，而不是#或者%;

## ActionContext、ServletContext、pageContext的区别？

1）ActionContext是当前的Action的上下文环境，通过ActionContext可以获取到request、session、ServletContext等与Action有关的对象的引用；

2）ServletContext是域对象，一个web应用中只有一个ServletContext，生命周期伴随整个web应用；

3）pageContext是JSP中的最重要的一个内置对象，可以通过pageContext获取其他域对象的应用，同时它是一个域对象，作用范围只针对当前页面，当前页面结束时，pageContext销毁，

生命周期是JSP四个域对象中最小的。

### result的type属性中有哪几种结果类型？

一共10种：

dispatcher

struts默认的结果类型，把控制权转发给应用程序里的某个资源不能把控制权转发给一个外部资源，若需要把控制权重定向到一个外部资源, 应该使用

redirect结果类型

redirect    把响应重定向到另一个资源（包括一个外部资源）

redirectAction      把响应重定向到另一个 Action

freemarker、velocity、chain、httpheader、xslt、plainText、stream

## 拦截器的生命周期与工作过程？

1）每个拦截器都是实现了Interceptor接口的 Java 类；

2）init(): 该方法将在拦截器被创建后立即被调用, 它在拦截器的生命周期内只被调用一次. 可以在该方法中对相关资源进行必要的初始化；

3）intercept(ActionInvocation invocation): 每拦截一个动作请求, 该方法就会被调用一次；

4）destroy: 该方法将在拦截器被销毁之前被调用, 它在拦截器的生命周期内也只被调用一次；

5）struts2中有内置了18个拦截器。

## struts2如何完成文件的上传？

1、JSP页面：

1）JSP页面的上传文件的组件：<s: file name=”upload” />，如果需要一次上传多个文件, 就必须使用多个 file 标签, 但它们的名字必须是相同的，即：

 name=“xxx”的值必须一样；

2）必须把表单的enctype属性设置为：multipart/form-data；

 3）表单的方法必须为post，因为post提交的数据在消息体中，而无大小限制。

2、对应的action：

 1）在 Action 中新添加 3 个和文件上传相关的属性；

2）如果是上传单个文件, uploadImage属性的类型就是 java.io.File, 它代表被上传的文件, 第二个和第三个属性的类型是 String, 它们分别代表上传文

件的文件名和文件类型，定义方式是分别是：

jsp页面file组件的名称+ContentType,  jsp页面file组件的名称+FileName

3）如果上上传多个文件, 可以使用数组或 List