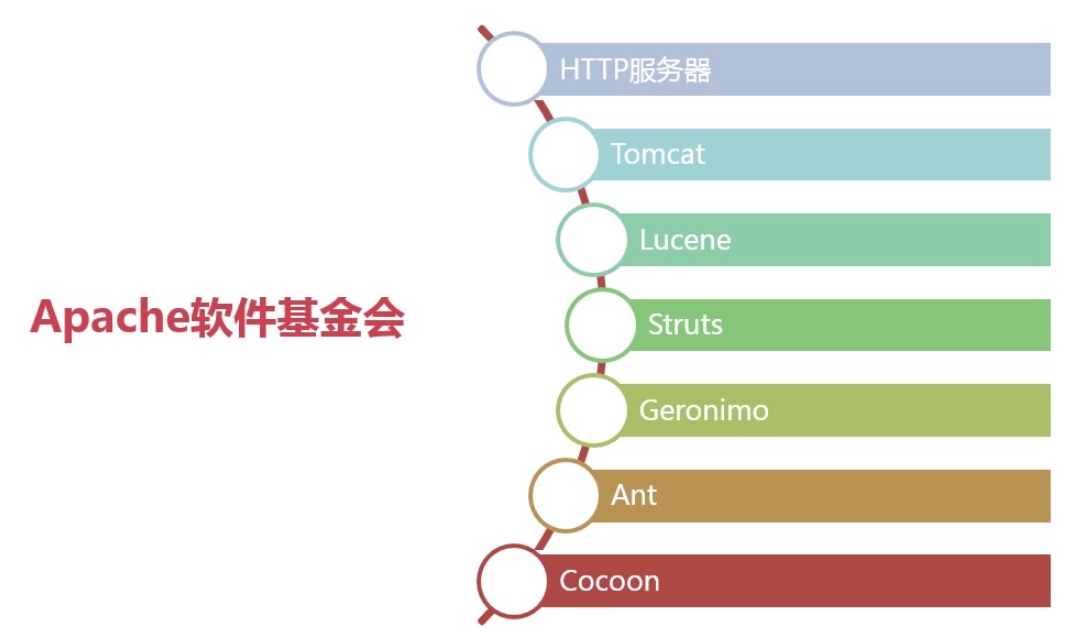
**一．Struts2-BASE**

英文翻译：支撑，支架

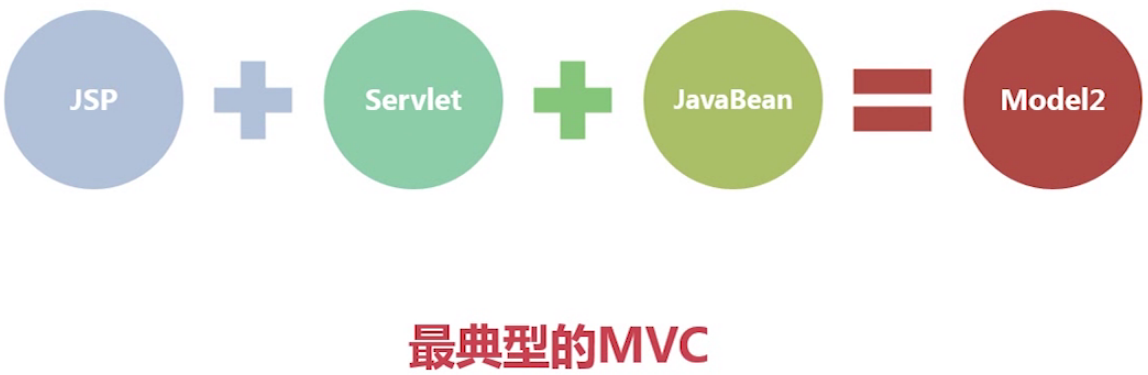
起源：Jakarta的子项目-----Apache软件基金会的开源子项目





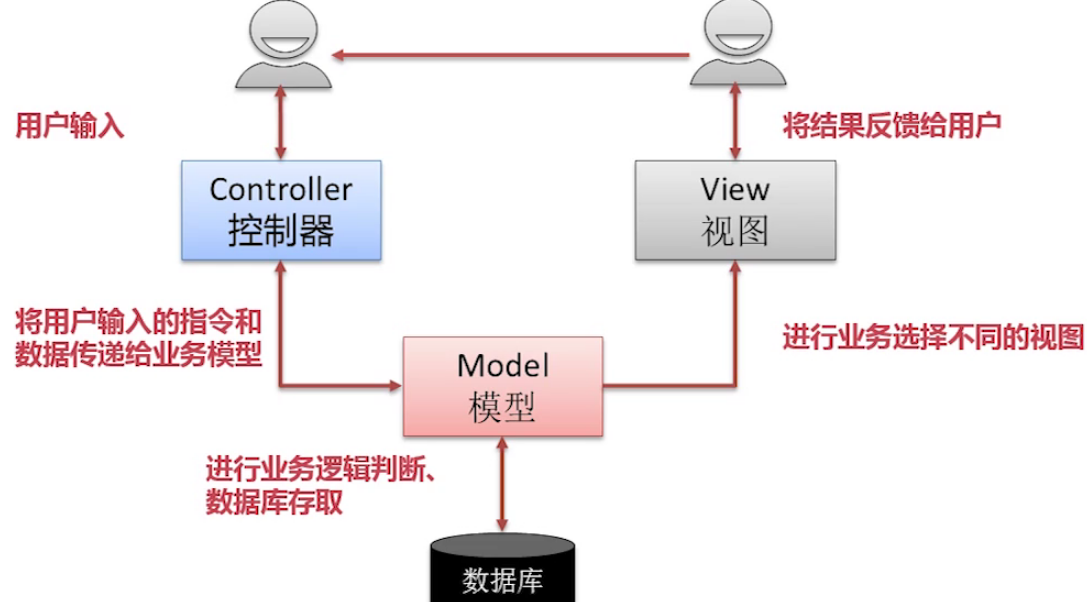






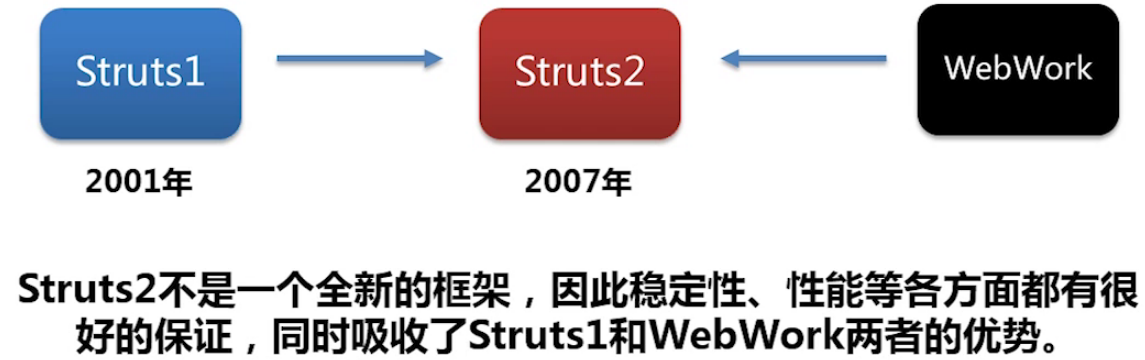
**Model2是最典型的MVC开发模式**





**MVC闭环**

**在Struts中，controller就是action**



对于Struts1有革命性的改革，继承了WebWork血统

注解





Full-jars结构内容：

apps:官网提供5个样例

docs:内部提供Struts2的API

lib:提供运行环境的jar包

基础包：

Commons-fileupload-1.31.jar 上传下载包

Commons-io-2.2.jar io包

Commons-logging-1.1.3.jar 日志包

commons-lang3-3.2.jar 工具包

Freemarker-2.3.22.jar 模板引擎，基于模板生成文本输出的控制工具

Struts2-core-2.3.24.jar 核心包

Xwork-core-2.3.24.jar

Ognl-3.0.6.jar 类似于el表达式

Javassist-3.11.0.GA.jar 解析java类文件的包

**FIRST DEMO：**

1. 创建JavaWeb动态项目---添加lib
2. 配置web.xml

<filter>

<filter-name>struts2</filter-name>

<filter-class> org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter

</filter-class>

</filter>

<filter-mapping>

<filter-name>struts2</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

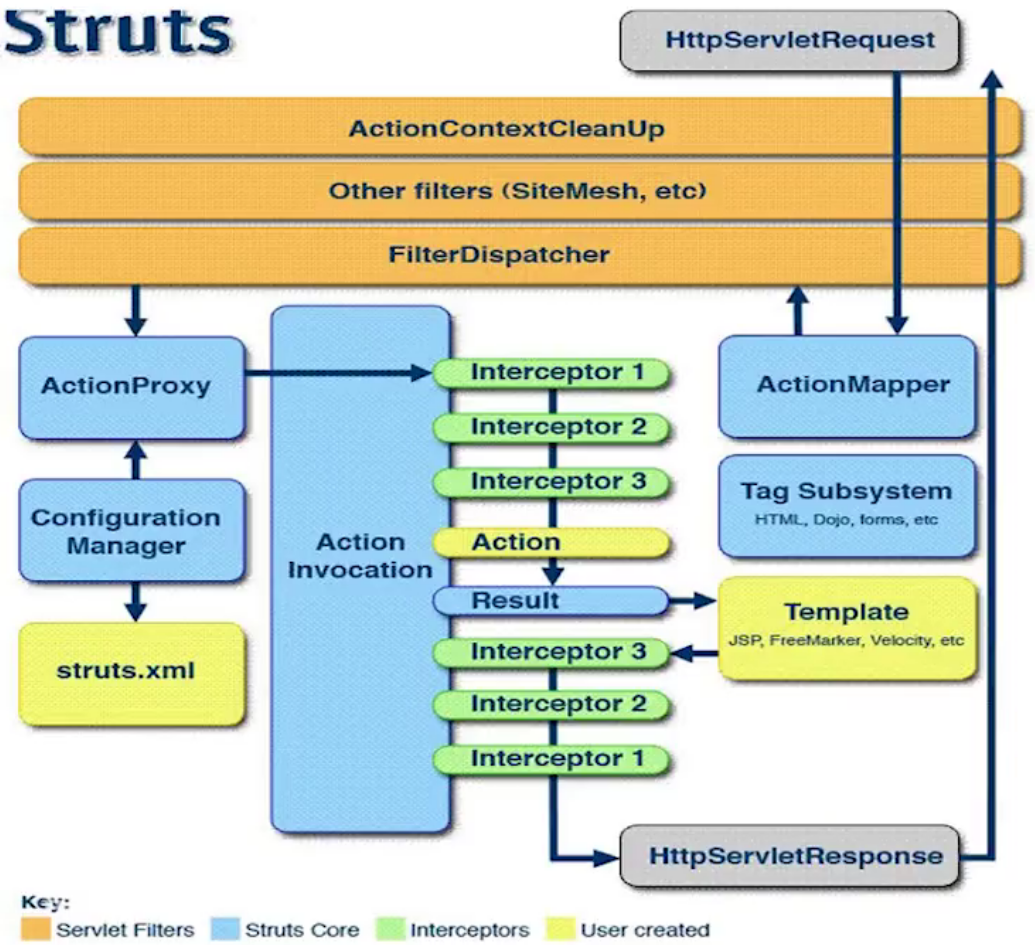
1. 创建action【java类继承ActionSupport】---重写excute方法
2. 添加并配置Struts.xml----创建jsp视图页面接收请求和数据

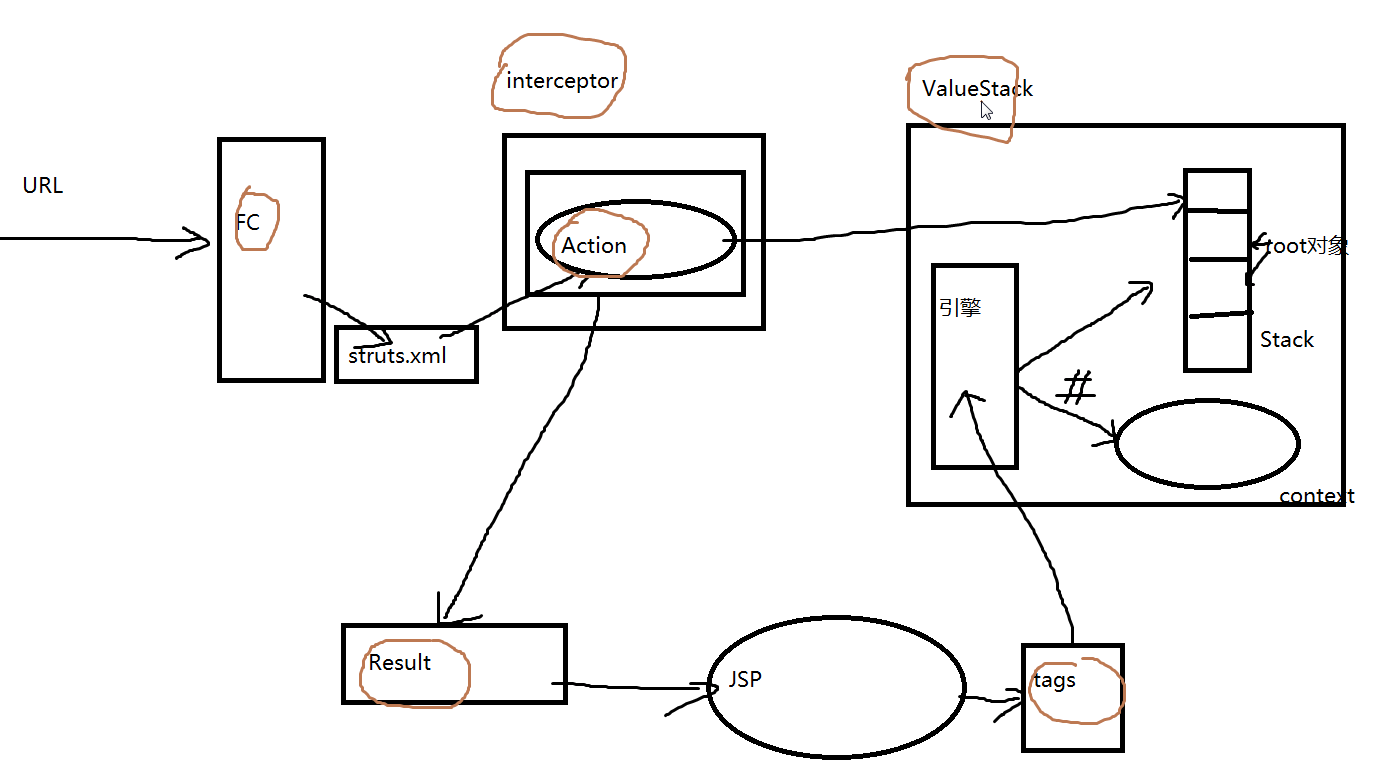
5.浏览器访问通过struts.xml中action后的name，地址栏使用name.action访问，不写action也可以

[struts2](https://www.baidu.com/s?wd=struts2&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YLuhmvn1F-rjRsmyRsnW040ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPH0sPHmLnWc3" \t "https://zhidao.baidu.com/question/_blank)中的action可以继承ActionSupport，也可以不继承ActionSupport。  
不继承ActionSupport的情况只需要有一个方法，返回String，即可，记住，在继承ActionSupport的情况下，必须有无参[构造函数](https://www.baidu.com/s?wd=%E6%9E%84%E9%80%A0%E5%87%BD%E6%95%B0&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YLuhmvn1F-rjRsmyRsnW040ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPH0sPHmLnWc3" \t "https://zhidao.baidu.com/question/_blank)。  
继承ActionSupport的好处在于：  
1、能够使用struts预设的返回字符串，如SUCCESS,INPUT等等。  
2、重写方法，更方便的实现验证、国际化等等功能。  
3、与struts的功能结合紧密，方便开发

提供了很多默认方法，这些默认方法包括国际化信息的方法、默认的处理用户请求的方法等，这样可以大大的简化Acion的开发。

**Struts2工作原理**





**1.**Struts2在web.xml中配置了/\*【代表所有】过滤器，项目启动后过滤器生效，所有的请求都会经过过滤器，

<filter-class> org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter

</filter-class>

但是Struts2会从其中过滤与自己相关【.action】的请求，其他的.jsp和.html会放行。

用户通过HttpServletRequest发送请求，经过一系列核心过滤器：

ActionContextCleanUp【可选】：用于和其他的框架进行集成，例如SiteMesh

FilterDispatcher：Struts2--2.1.2版本之前的核心过滤器

StrutsPrepareAndExecuteFilter：从2.1.3版本替代FilterDispatcher

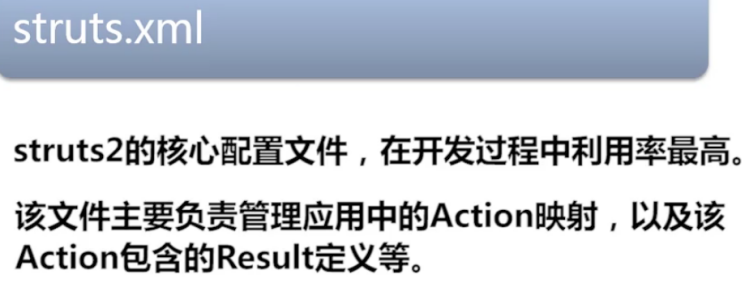
替代原因：一般自定义过滤器写在Struts2的核心过滤器之前，但是如果希望在Struts2拦截之后到action执行之前再写过滤器，FilterDispatcher无法做到。

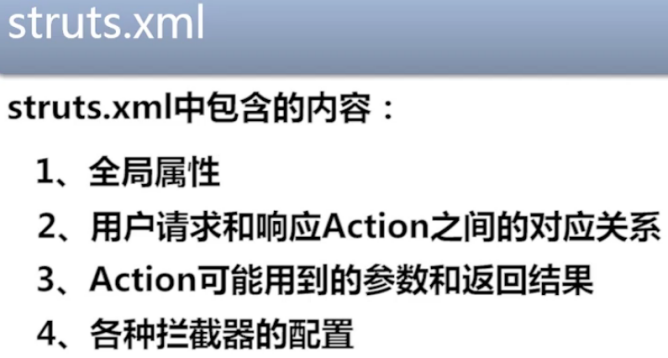
StrutsPrepareAndExecuteFilter见名知意：允许在准备和执行之间添加自定义过滤器

1. 用户请求经过一系列的过滤器之后，到达ActionMapper【只有.action后缀的请求进行匹配】，匹配成功之后通过StrutsPrepareAndExecuteFilter过滤器到ActionProxy【该模块可以通过ConfigurationManager读取Struts.xml文件找到具体的action类】
2. 然后通过ActionProxy代理创建一个action的反向实例ActionInvocation，
3. 再经过一系列的拦截器执行到action，返回一个result字符串对象，该对象对应的是一个视图jsp,然后再经过一些列的拦截器通过HttpServletResponse返回到用户的实例进行显示

**Struts2文件结构**









**二．Struts2-Advanced**



1. **Servlet API**

HttpServletRequest HttpServletResponse ServletContext

JSP request response Application

Struts2不再与Servlet API进行耦合，无需传入request，response参数，方便测试

但底层依旧是使用和访问request，response

Struts2提供三种方式访问Servlet API



ActionContext：一个上下文的类，通过它获取上下文的对象，所有对象以Map形式存储

**2.action的搜索顺序**



Package的namespace命名可以是“/aaa/bbb/ccc”，这是一个包的命名空间

第一步判断：是以整体“/aaa/bbb/ccc”来判断是否有一个包的namespace与之相同

存在：则判断action是否存在，如果action不存在就直接去默认的namespace包里找

不存在：以“/aaa/bbb”来判断，还没有就以“/aaa”来判断，直到默认的namespace

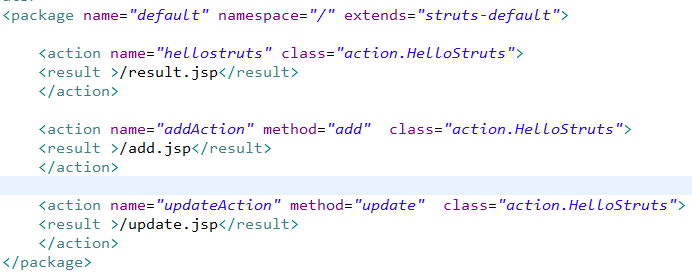
**3.动态方法调用**

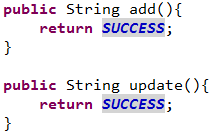


三种方式调用action类定义的方法，感叹号方式官方不建议使用，通配符官方建议



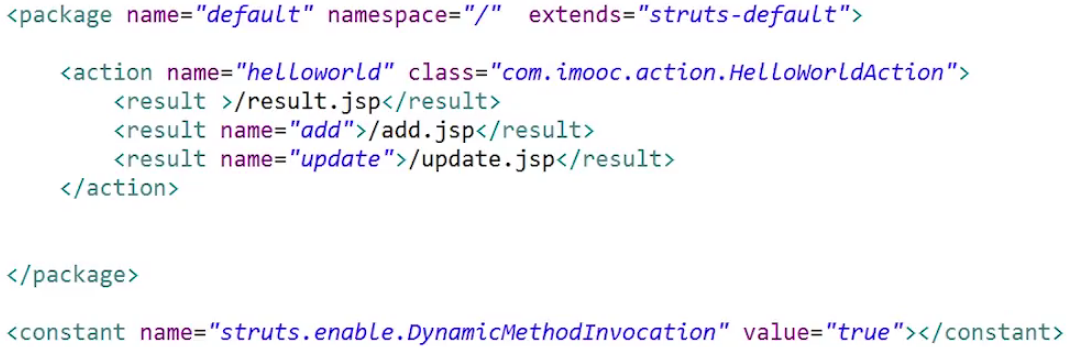
**指定method属性：**

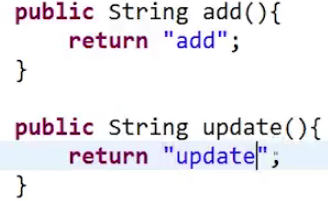






**感叹号方式：**









**4.指定多个配置文件**



**5.默认action**

为了解决404页面不友好，替换错误信息页面

<!-- 默认action,为了解决404页面不友好，替换错误信息页面 ,但无法解决前缀为hellostruts\_xxoo的问题，

因为此种写法会走action匹配xxoo方法，不带前缀hellostruts\_就不会进入action，500错误

-->

<default-action-ref name=*"er"*></default-action-ref>

<action name=*"er"* >

<result >/error.jsp</result>

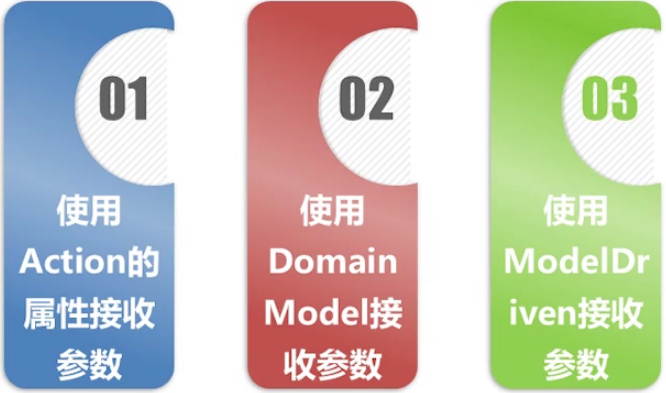
</action>

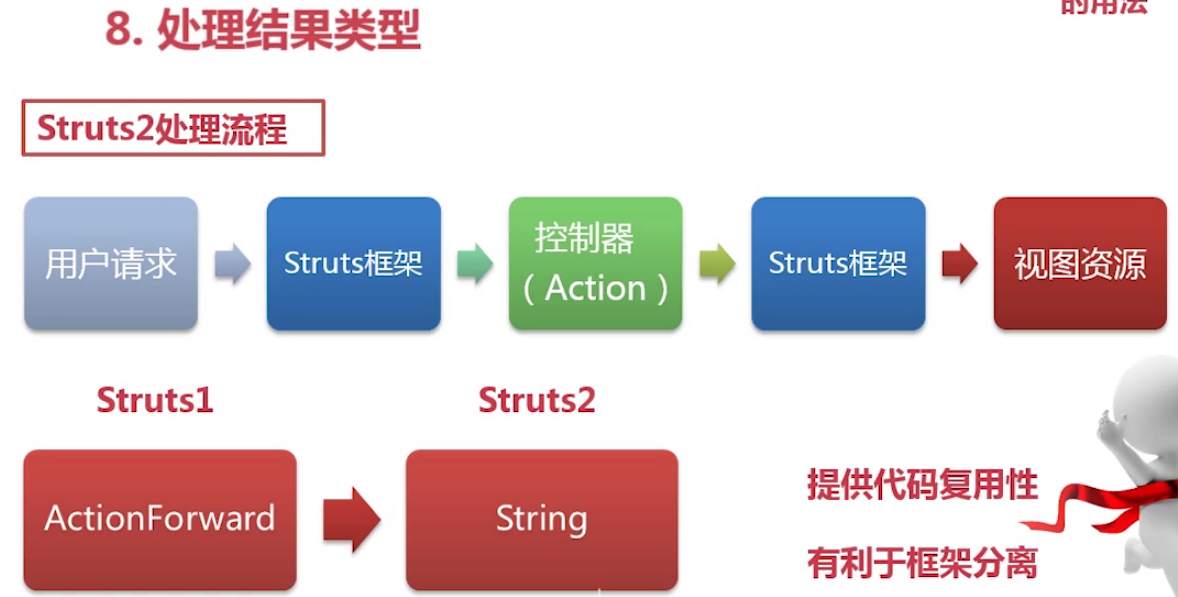
**6.后缀**

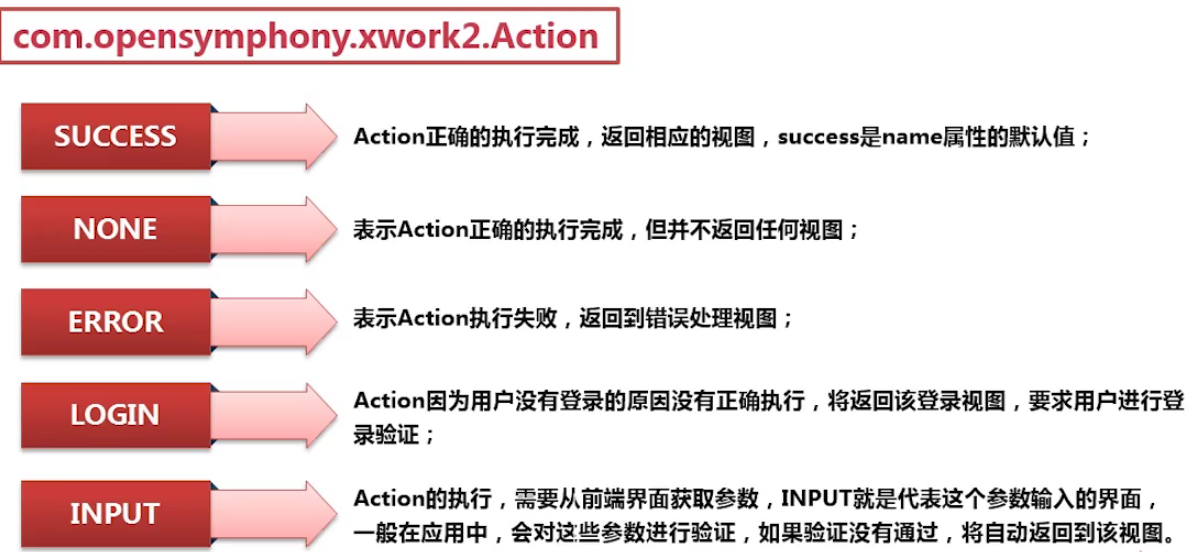
<!-- #指定后缀为html形式的请求可被Struts2处理，可配置多个请求后缀，比如.do .struts, 一旦设定，访问必须遵循value格式，类似于过滤器 -->

<!-- <constant name="struts.action.extension" value="html"></constant> -->

**7.接收参数**

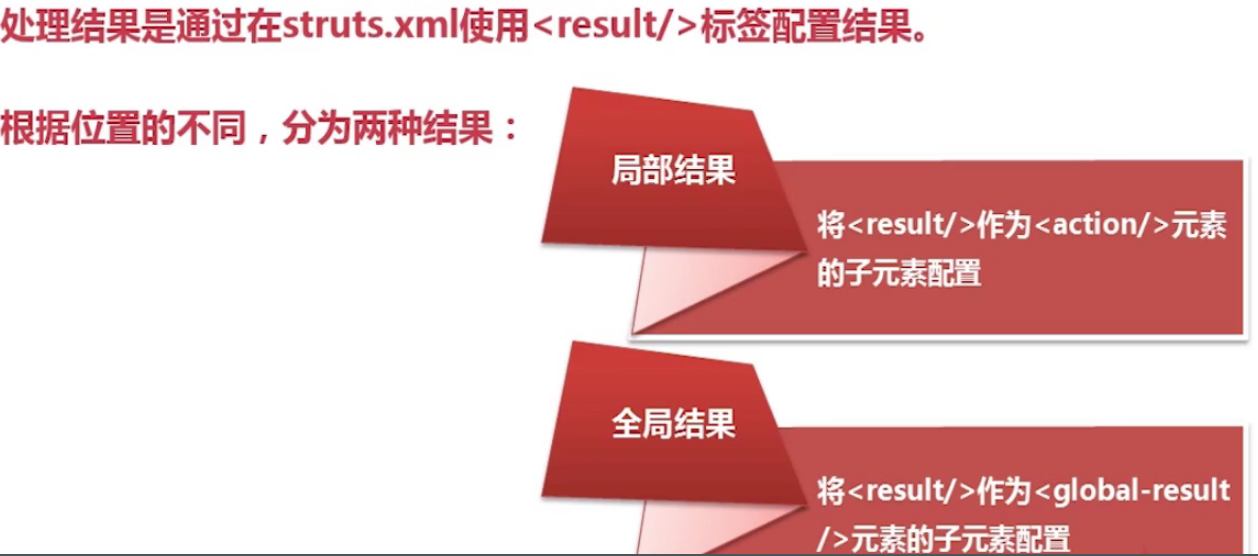


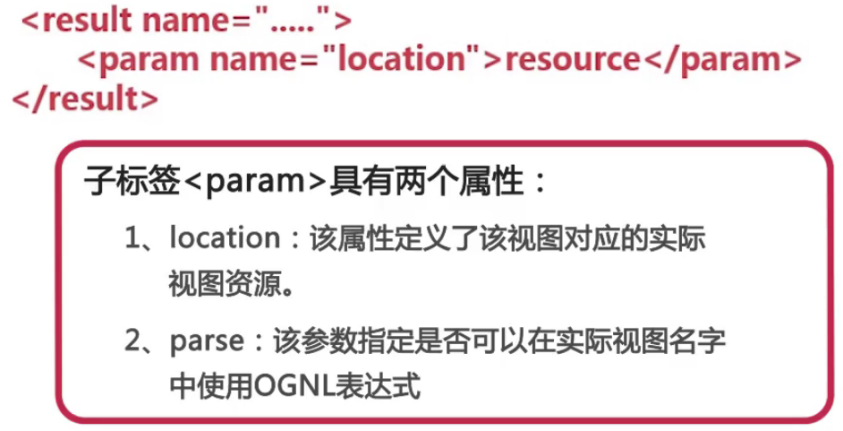




INPUT：自动跳转input界面的方式有两种，第一种是传的参数类型转换错误，第二种是

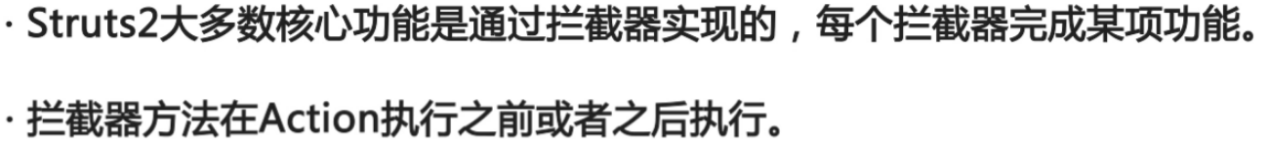
Action的执行方法添加this.addFieldError(fieldName,errorMessage)，无错时无参数值，所以不执行，出错的时候有参数值，执行返回input对应的jsp页面

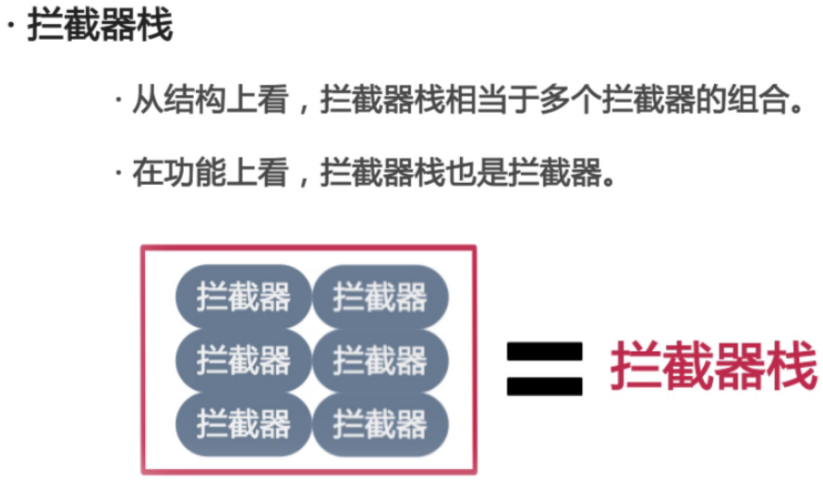


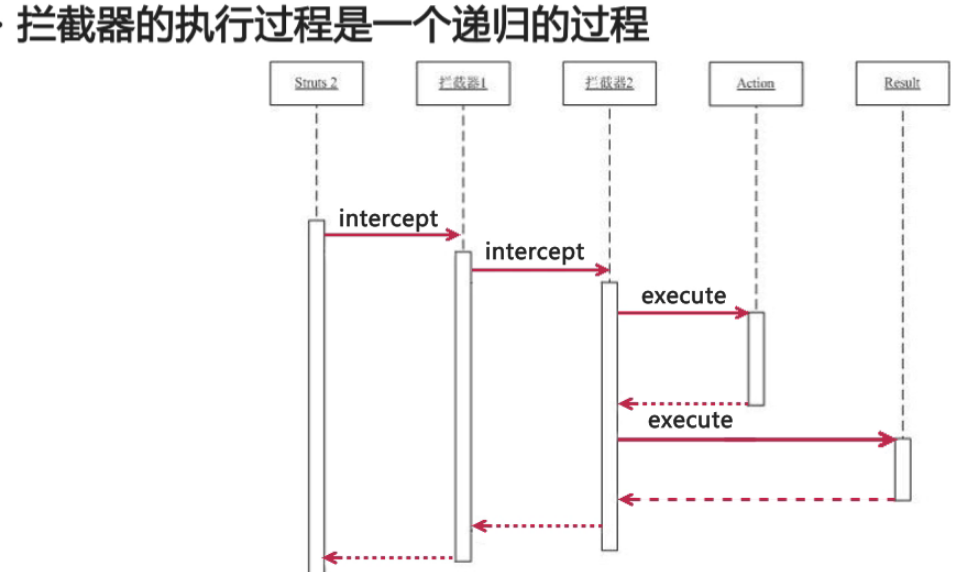




**二．Struts2-拦截器**

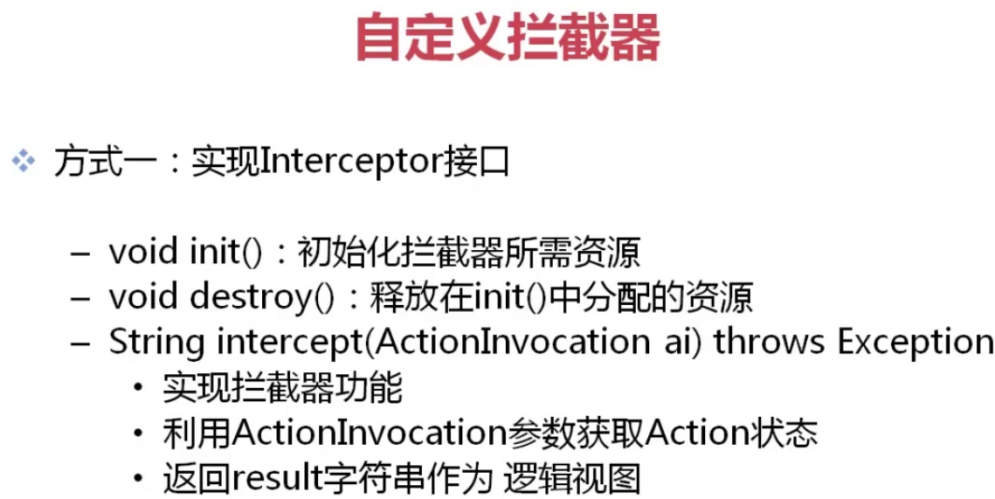


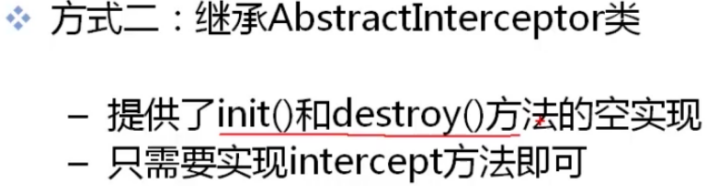




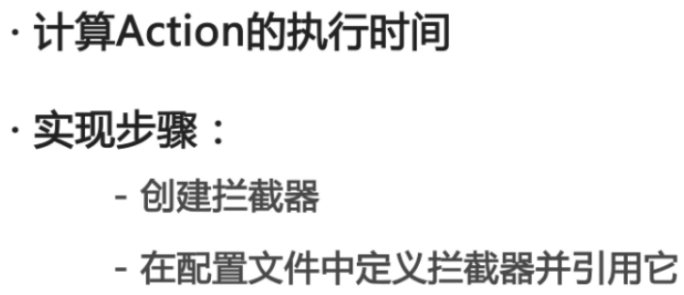
与过滤器Filter非常相似

Action执行完毕之后会返回一个字符串，进行result视图的匹配，此时结果并没有真正的响应到客户端，因为在action执行完以后，它还会需要将我们之前执行的拦截器进行反向的再执行一遍，当把所有的拦截器反向的执行完之后，才会进行真正的响应，实际上是一个递归的过程





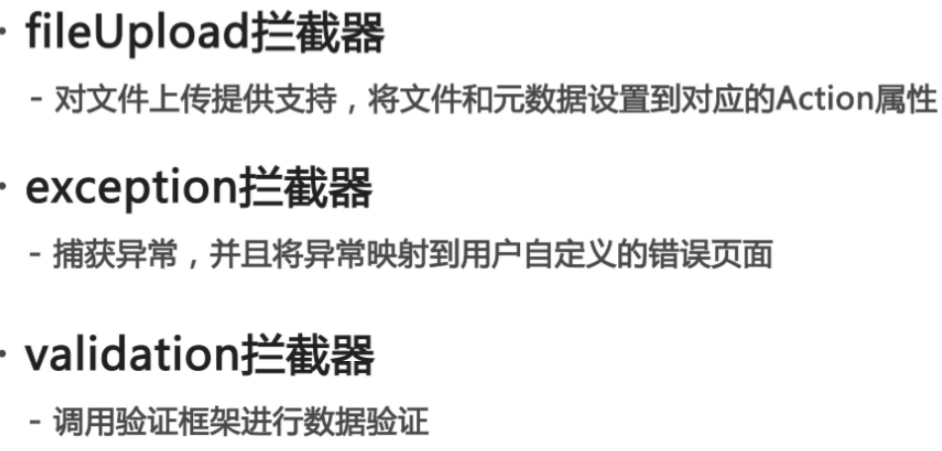
实际开发中，一般都会使用方法二，该抽象类实现了Interceptor接口



实际上定义一个拦截器，分为两大部分：

1. 自定义拦截器：创建类继承AbstractInterceptor，内部实现重写intercept方法，当调用action的时候自动调用该方法
2. 在Struts.xml文件中进行配置，先注册拦截器，在相应的action中引用所定义的拦截器





Struts2中内建了许多拦截器，可以完成相应的功能：

Params拦截器：把客户请求的参数绑定到action的属性中，就是通过该拦截器实现的

staticParams拦截器：通过在Struts.xml文件中为action元素配置子元素param从而实现为action的属性进行赋值

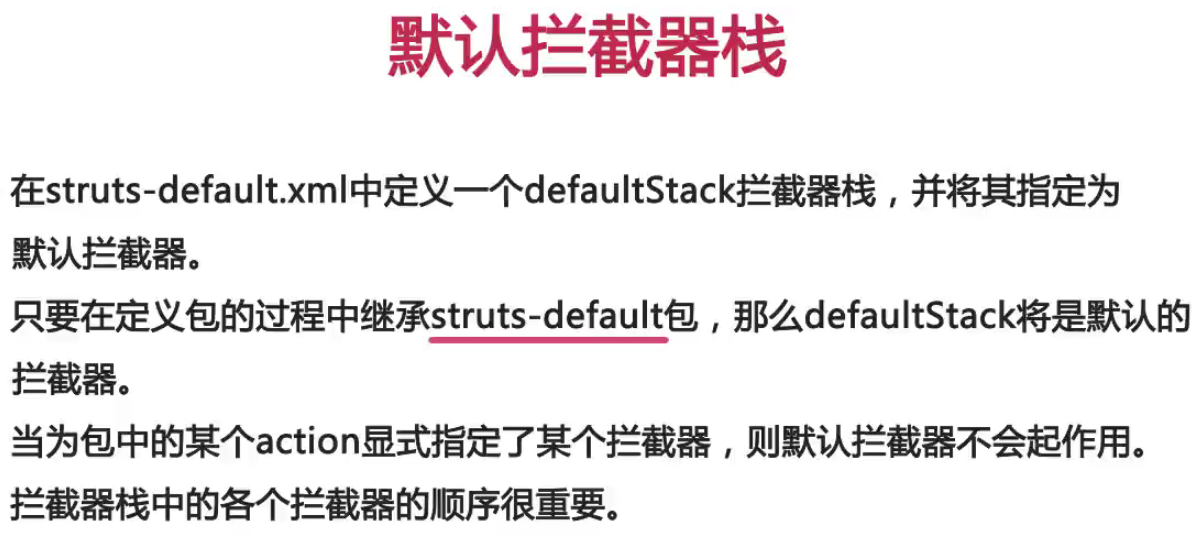
fileUpload拦截器实际上内部使用Commons-FileUpload组件，对该组件进行进一步的封装，使文件的上传下载变得非常简单，会自动将我们上传的文件数据绑定在action之中。

具体信息在Struts-core的jar包中的struts-default.xml文件中查询，Struts在启动时也会自动加载该xml文件

文件内部定义了许多基础的拦截器栈，如basicStack，其引用的拦截器顺序非常重要，决定了每个拦截器应用的顺序

最下方<default-interceptor-ref name=*"defaultStack"*/>

在没有手工指定拦截器的情况下，其默认已经引用了*defaultStack*



一般来说会建议先引用默认的拦截器栈