1、描述Struts的工作流程

（1）客户端发出一个请求，然后核心过滤器PrepareAndExecuteFilter进行拦截。

（2）在web容器启动的时候，Struts2中有三个核心的类会工作:ConfigurationManager、ActionMapper、ObjectFactory.

ConfigurationManager用来存放配置文件中的一些基础配置信息；

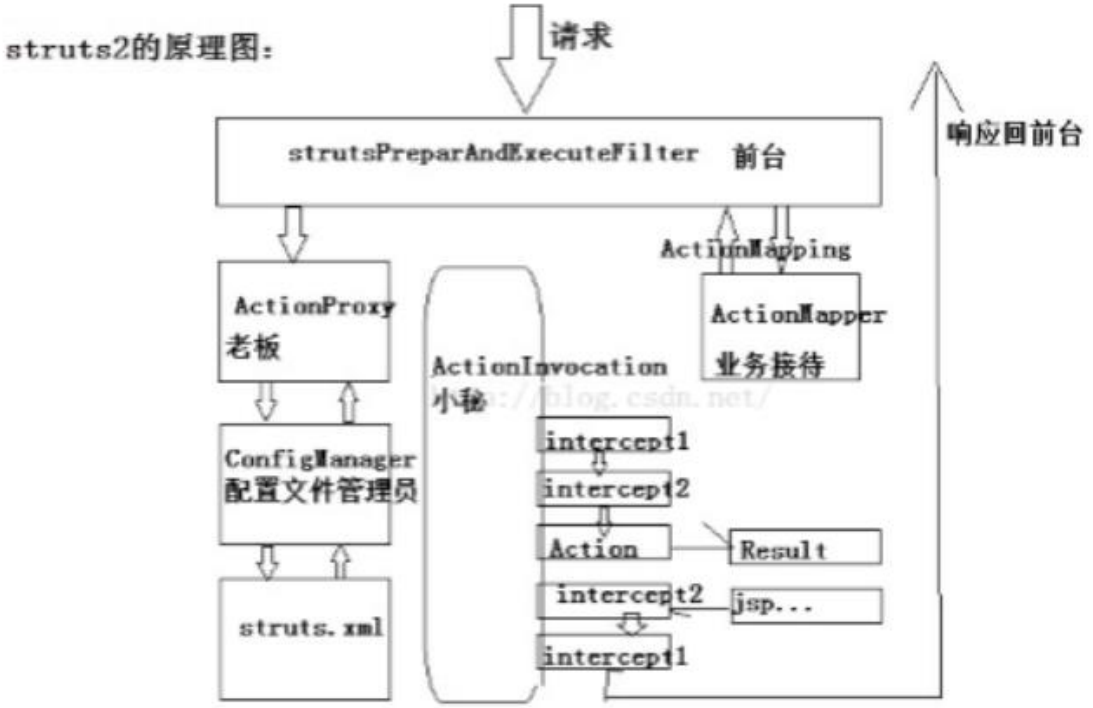
ActionMapper用来存放请求URL和Action的对应关系信息；

ObjectFactory主要用来创建Struts2中的对象，例如：Action、Result、Interceptor等。

（3）在第一步发出请求后，PrepareAndExecuteFilter会根据URL向ActionManager匹配对应的Action实例。匹配到之后，PrepareAndExecuteFilter会把请求处理交给ActionProxy,ActionProxy是Action的代理对象。ActionProxy会创建一个ActionInvocation实例，表示Action的执行状态。

（4）ActionInvocation加载Action相关的所有Interceptor.ActionInvocation通过Invoke方法调用Action中的方法

（5）当Action执行完毕后。返回String类型的视图名称。视图名称对应一个Result对象，然后Result会进行解析，得到真正的视图资源，再转发或跳转给客户端



jsp请求：

```
<div class="back_link"><a href="index.jsp">返回</a></div>
    <form action="addFruit_FruitAction" method="post">
        <table id="tbl" border="1">
            <caption>添加水果信息</caption>
            <tr>
                <th>名称：</th>
                <td><input type="text" name="fruit.fname"/></td>
            </tr>
```

web.xml中：

```
<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>
```

根据ActionInvocation找到对应的Action的方法，通过Invoke方法调用action的方法并且返回一个String类型的视图名称：

```
public String addFruit(){
    boolean flag = fruitBiz.addFruit(fruit);
    return "preIndex";
}
```

根据返回的视图名称找到对应的Result(在struts.xml配置文件中)：

```
<constant name="struts.devMode" value="true" />
<package name="default" namespace="/" extends="struts-default">
    <action name="*_*" class="com.gem.fruit.actions.{2}" method="{1}">
        <result type="redirect">/index.jsp</result>
        <result name="preIndex" type="redirectAction">
            <param name="actionName">preIndex_FruitAction</param>
            <param name="namespace">/</param>
        </result>
        <result name="detail">/detail.jsp</result>
        <result name="login" type="redirect">/login.jsp</result>
    </action>
</package>
```

跳转到客户端界面进行信息的展示

54. struts2 和 springMVC 的区别？

1、Struts2 是类级别的拦截， 一个类对应一个 request 上下文，SpringMVC 是方法级别的拦截，一个方法对应一个 request 上下文，而方法同时又跟一个 url 对应,所以说从架构本身上 SpringMVC 就容易实现 restful url,而 struts2 的架构实现起来要费劲，因为 Struts2 中 Action 的一个方法可以对应一个 url，而其类属性却被所有方法共享，这也就无法用注解或其他方式标识其所属方法了。

2、由上边原因，SpringMVC 的方法之间基本上独立的，独享 request response 数据，请求数据通过参数获取,处理结果通过 ModelMap 交回给框架,方法之间不共享变量,而 Struts2 搞的就比较乱，虽然方法之间也是独立的，但其所有 Action 变量是共享的，这不会影响程序运行，却给我们编码 读程序时带来麻烦，每次来了请求就创建一个 Action，一个 Action 对象对应一个 request 上下文。

3、由于 Struts2 需要针对每个 request 进行封装，把 request，session 等 servlet 生命周期的变量封装成一个一个 Map，供给每个 Action 使用，并保证线程安全，所以在原则上，是比较耗费内存的。

4、 拦截器实现机制上，Struts2 有以自己的 interceptor 机制，SpringMVC 用的是独立的 AOP 方式，这样导致 Struts2 的配置文件量还是比 SpringMVC 大。

5、SpringMVC 的入口是 servlet，而 Struts2 是 filter（这里要指出，filter 和 servlet 是不同的。以前认为 filter 是 servlet 的一种特殊），这就导致了二者的机制不同，这里就牵涉到 servlet 和 filter 的区别了。

6、SpringMVC 集成了 Ajax，使用非常方便，只需一个注解@ResponseBody 就可以实现，然后直接返回响应文本即可，而 Struts2 拦截器集成了 Ajax，在 Action 中处理时一般必须安装插件或者自己写代码集成进去，使用起来也相对不方便。

7、SpringMVC 验证支持 JSR303，处理起来相对更加灵活方便，而 Struts2 验证比较繁琐，感觉太烦乱。

8、Spring MVC 和 Spring 是无缝的。从这个项目的管理和安全上也比 Struts2 高(当然 Struts2 也可以通过不同的目录结构和相关配置做到 SpringMVC 一样的效果，但是需要 xml 配置的地方不少)。

9、 设计思想上，Struts2 更加符合 OOP 的编程思想， SpringMVC 就比较谨慎，在 servlet 上扩展。

10、SpringMVC 开发效率和性能高于 Struts2。

11、SpringMVC 可以认为已经 100%零配置。