

一、实验步骤

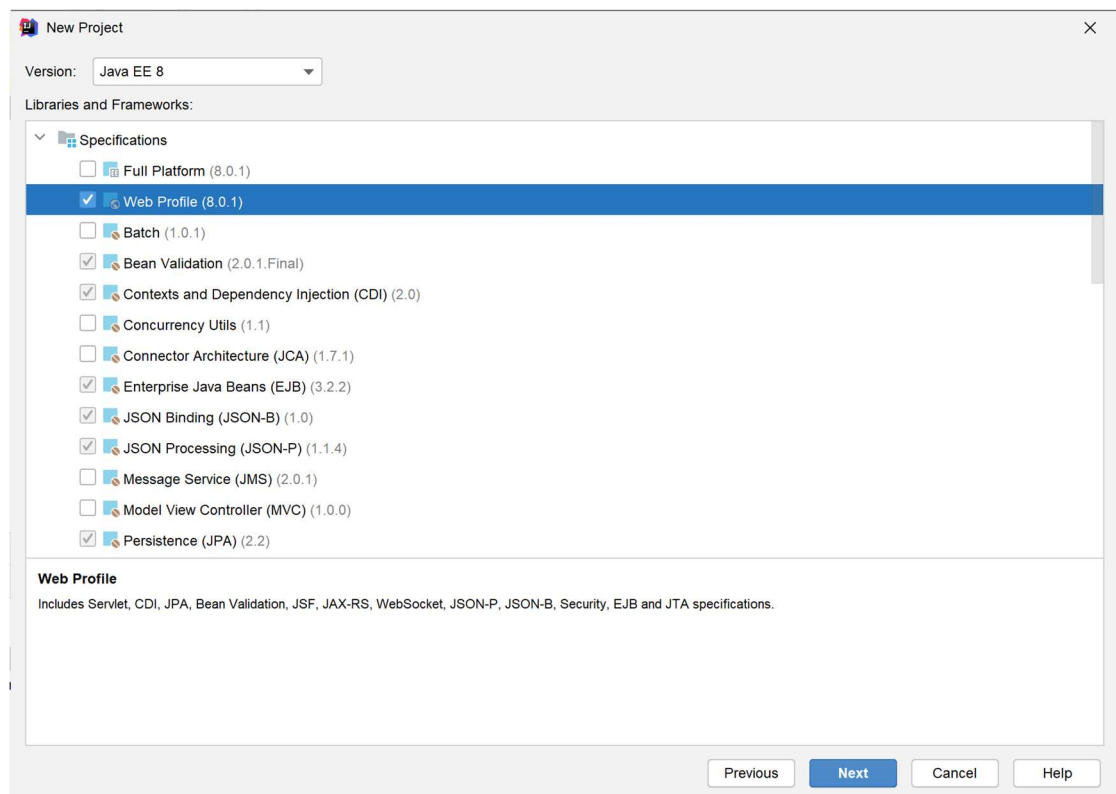
1.1、创建项目以及配置环境

由于此次使用到了 struts 框架，而通过初步验证，下载的 struts 包并不足以支撑起配置 struts 框架。

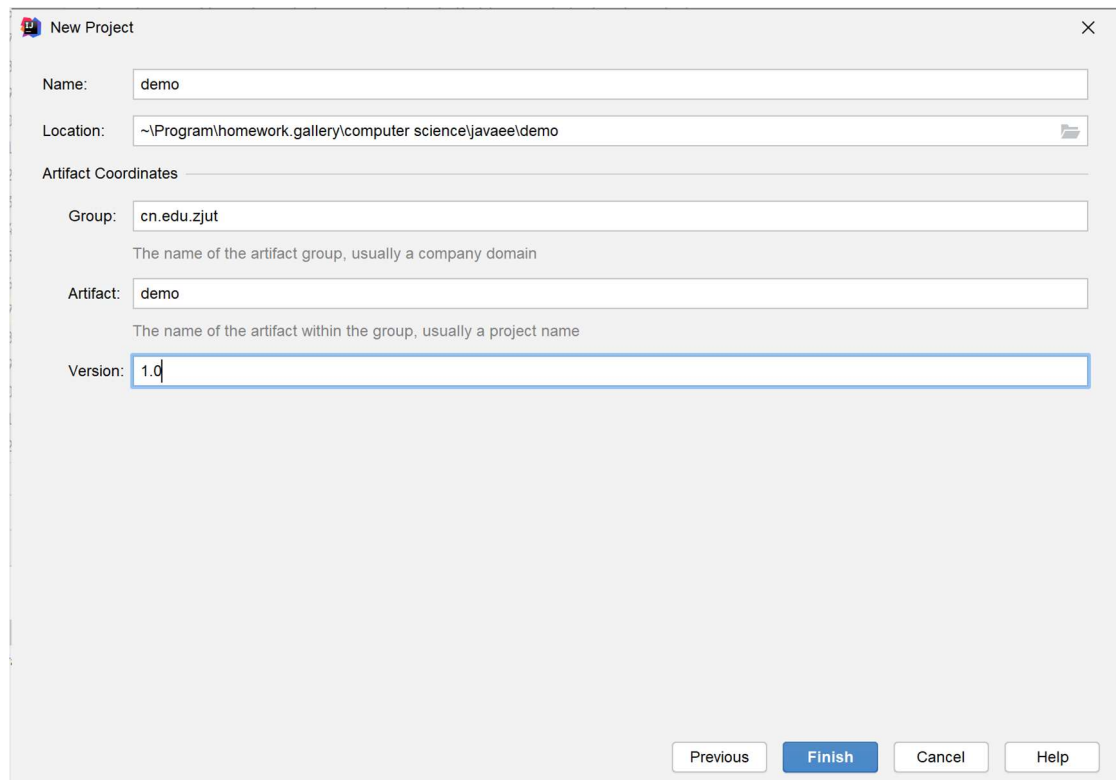
此实验用到的 IDE 是 IntelliJ IDEA，使用到的包管理器是 Gradle。

第一步：创建一个新的项目（New>Project...），选择 Java Enterprise，Build Tool 选择 Gradle。点击 Next。

然后选择 Version 为 Java EE 8，架构选择 Web Profile



根据需求选择 Name、Location、Group



然后添加 struts2 的插件。

删除 resources\META-INF 文件夹和 webapp\WEB-INF 下的 faces-config.xml

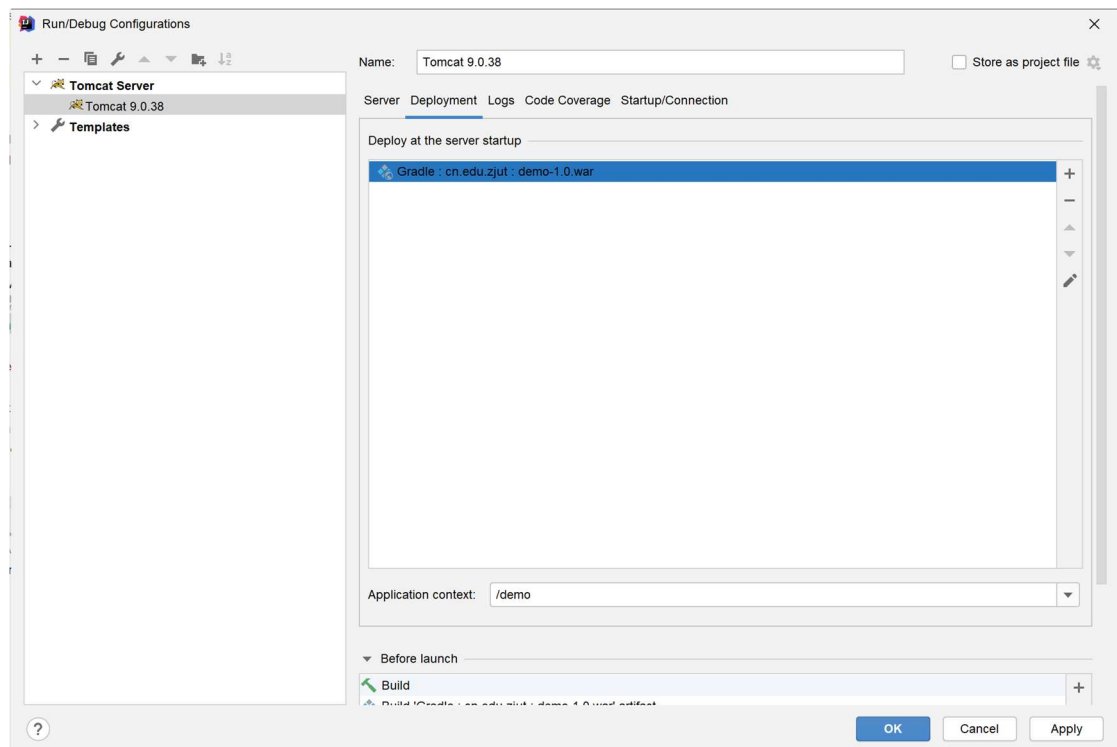
第二步：进行 Gradle 的配置

在 build.gradle 的 repositories 内加上 maven { url

"https://maven.aliyun.com/repository/public" } dependencies 内加上 compile group: 'org.apache.struts', name: 'struts2-core', version: '2.5.25'。表示启用 aliyun 的镜像站，并导入 struts 需要的依赖。

第三步：添加启动项配置

启动项，选了 Tomcat Server > Local，然后选择运行的 war 包，修改部署的 Application Context



第四步：添加必要的文件

在 `src\main\java\resources` 下添加 `struct.xml` 文件

1.2、实验一

第一步：新建 `UserBean`，`UserService` 和 `UserAction` 类，编写相应的程序代码

```
ml x login.jsp x UserBean.java x UserService.java x UserAction.java x
2
3 import bean.UserBean;
4 import service.UserService;
5
6 public class UserAction {
7     private UserBean loginUser;
8
9     public UserBean getLoginUser() {
10         return loginUser;
11     }
12
13     public void setLoginUser(UserBean loginUser) {
14         this.loginUser = loginUser;
15     }
16
17     public String login(){
18         UserService userService = new UserService();
19         if (userService.login(loginUser)){
20             return "success";
21         }
22         return "fail";
23     }
24 }
```

第二步：配置 struts.xml 文件

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <!DOCTYPE struts PUBLIC
4     "-//Apache Software Foundation//DTD Struts Configuration 2.5//EN"
5     "http://struts.apache.org/dtds/struts-2.5.dtd">
6 <struts>
7     <package name="strutsBean" extends="struts-default" namespace="/">
8         <action name="login" class="action.UserAction" method="login">
9             <result name="success">/loginSuccess.jsp</result>
10            <result name="fail">/loginFail.jsp</result>
11        </action>
12    </package>
13 </struts>
```

第三步：配置 web.xml 文件

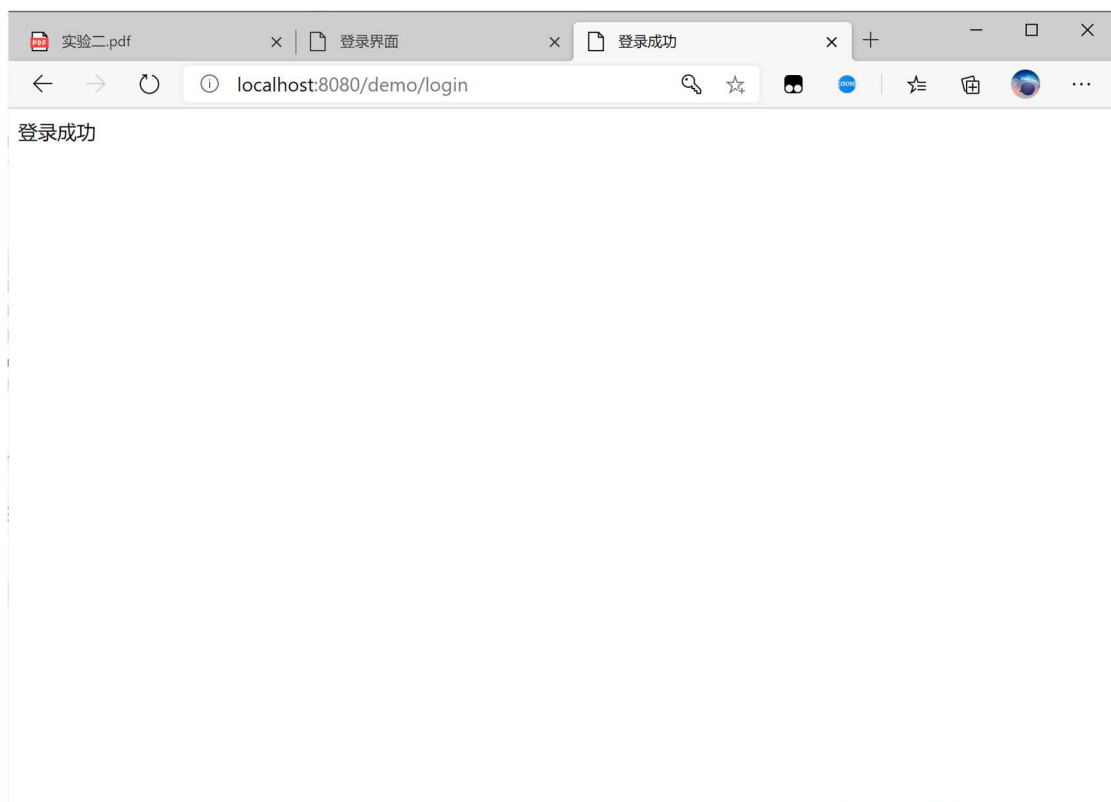
第四步：创建 login.jsp, loginSuccess.jsp, loginFailed.jsp, 然后编写相应的代码

```

8      <%@ page contentType="text/html;charset=UTF-8" language="java" %>
9      <%@ taglib prefix="s" uri="/struts-tags" %>
10     <html>
11     <head>
12         <title>登录界面</title>
13     </head>
14     <body>
15         <s:form action="login" method="POST">
16             <div>
17                 <label>请输入用户名: <s:textfield name="loginUser.account"/></label>
18             </div>
19             <div>
20                 <label>请输入密码: <s:password name="loginUser.password"/></label>
21             </div>
22             <s:submit/>
23         </s:form>
24     </body>
25 </html>
26

```

登录成功界面



第五步：尝试修改 struts2 的配置，跟踪对应的错误。

1. 修改 login.jsp 和 LoginAction，使得 parameter 与 LoginAction 不一致。

```

<s:form action="login" method="POST">
    <div>
        <label>请输入用户名: <s:textfield name="loginUser.ac"/></label>
    </div>
    <div>
        <label>请输入密码: <s:password name="loginUser.password"/></label>
    </div>
    <s:submit/>
</s:form>

```

结果：访问 login.jsp 没有发生错误，点击提交时发生 500 错误

HTTP状态 500 - 内部服务器错误

类型 异常报告

描述 服务器遇到一个意外的情况，阻止它完成请求。

例外情况

```

java.lang.NullPointerException
    service.UserService.login(UserService.java:7)
    action.UserAction.login(UserAction.java:19)
    java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
    java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    java.base/java.lang.reflect.Method.invoke(Method.java:566)
    ognl.OgnlRuntime.invokeMethodInsideSandbox(OgnlRuntime.java:1266)
    ognl.OgnlRuntime.invokeMethod(OgnlRuntime.java:1251)
    ognl.OgnlRuntime.callAppropriateMethod(OgnlRuntime.java:1969)
    ognl.ObjectMethodAccessor.callMethod(ObjectMethodAccessor.java:68)
    com.opensymphony.xwork2.ognl.accessor.XWorkMethodAccessor.callMethodWithDebugInfo(XWorkMethodAccessor.java:98)
    com.opensymphony.xwork2.ognl.accessor.XWorkMethodAccessor.callMethod(XWorkMethodAccessor.java:90)
    ognl.OgnlRuntime.callMethod(OgnlRuntime.java:2045)
    ognl.ASTMethod.getValueBody(ASTMethod.java:97)
    ognl.SimpleNode.evaluateGetValueBody(SimpleNode.java:212)
    ognl.SimpleNode.getValue(SimpleNode.java:258)
    ognl.Ognl.getValue(Ognl.java:537)
    ognl.Ognl.getValue(Ognl.java:501)
    com.opensymphony.xwork2.ognl.OgnlUtil$3.execute(OgnlUtil.java:492)
    com.opensymphony.xwork2.ognl.OgnlUtil.compileAndExecuteMethod(OgnlUtil.java:544)
    com.opensymphony.xwork2.ognl.OgnlUtil.callMethod(OgnlUtil.java:490)
    com.opensymphony.xwork2.DefaultActionInvocation.invokeAction(DefaultActionInvocation.java:438)
    com.opensymphony.xwork2.DefaultActionInvocation.invokeActionOnly(DefaultActionInvocation.java:293)
    com.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:254)

```

```

public class UserBean {
    private String account;
    private String password;
}

```

由于报的错误是在 login 方法内的 NullPointerException，因此我们得到的结果是表单进行填充时，首先创建一个 UserBean 对象，然后对表单内出现的字段进行填充，因此调用 login 方法时，account=null，因此报错。

我们进行进一步修改，将 account 修改为 account="123"

我们发现输入"";123 时，仍然能够登录，因此"123"是 account 的默认字段，这时容易出现奇怪的问题。

2. 修改 loginAction 和 struts.xml, 是返回的结果和 result 不一致

```
public class UserAction {
    private UserBean loginUser;

    public UserBean getLoginUser() { return loginUser; }

    public void setLoginUser(UserBean loginUser) { this.loginUser = loginUser; }

    public String login(){
        UserService userService = new UserService();
        if (userService.login(loginUser)){
            return "success";
        }
        return "t";
    }
}
```



1.3、实验二

步骤一：修改 UserBean, 在 UserService 和 UserAction 中分别添加相关的业务逻辑。

步骤二：修改 struts.xml 文件, 添加注册的相关页面调用逻辑。

```
<action name="register" class="action.UserAction" method="register">
    <result name="success">/registerSuccess.jsp</result>
    <result name="fail">/registerFail.jsp</result>
</action>
```

步骤三：创建 register.jsp、registerSuccess.jsp、registerFail.jsp, 并修改相关的页面, 使用 struts 和 struts-dojo 标签库。


步骤四：启动调试

请输入用户名:

请输入密码:

重复输入密码:

真实姓名:

请输入生日: 

联系地址:

联系电话:

电子邮箱:

cht 您注册成功了!

1.4、实验三

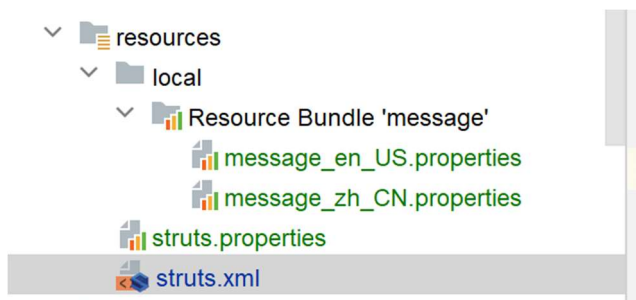
Please input your account:

Please input your password:

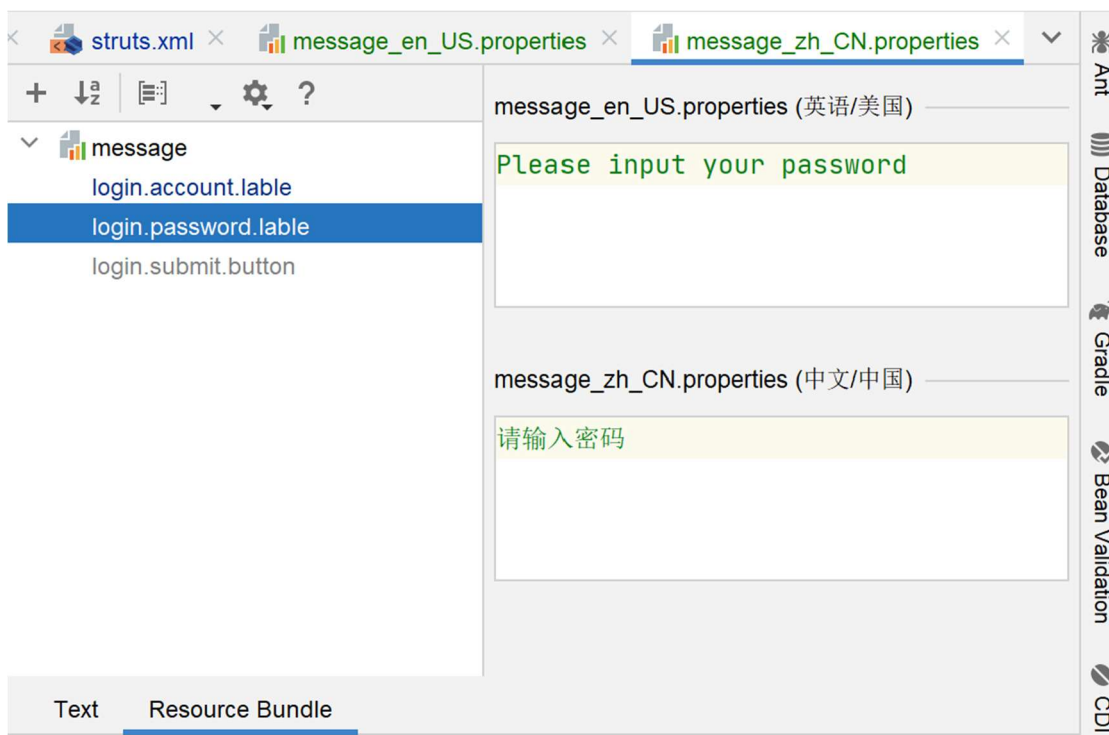
请输入用户名:

请输入密码:

应用国际化的关键代码:



需要一个 struts.properties 文件，由于配置 i18n，然后两个对应的资源字典，组合成一个 Bundle，然后在对应的 resource bundle 中编辑。



二、实验结果及分析

2.1、实验一

(2)

struts 框架是一个解决 MVC 问题的非侵入式框架，一般用于解决网页的 MVC 构建问题。首先，Filter 和 struts.xml 属于 struts 框架的配置部分，相当于 MVC 框架中的 Controller，而 Service 类属于服务模型，JavaBean 属于业务模型，同属于 MVC 框架中的 Model 部分。而 jsp 页面属于 MVC 框架中的 View 部分。

首先浏览器发送一个请求，由 Filter 接收，进入 struts2 的逻辑回路。框架查

看 struts.xml 文件中的配置，找到对应的 action，创建 bean 数据模型，然后创建对应的 action 实例，注入 bean，然后调用对应的方法。等待 action 返回结果后，struts 框架再次根据结果找到对应的返回页面。

(3)

表单参数与 Action 属性必须一一对应，表单参数中，使用.来表示层级的观念，例如 loginUser.name.first 表示三级属性。其中二级属性可以是一个 JavaBean 对象。

(4)

Action 中的 execute()方法用来执行对应的业务逻辑，并返回不同的结果，特点是返回的值一定是 String，用来和 struts.xml 对应来确定将要返回的页面。

(5)

struts 是根节点，package 表示一类 action，其具有相似的特征，但是一般小的项目只需要一个 package。

package 的子级是 action，表示一个 action 动作，其具有 name, class, method:"execute"三个 attribute，其中 name 表示 action 的名字，其对于 jsp 页面中的 action，class 表示执行 action 的类，是类的全名，而 method 表示执行的方法，默认为 execute。

action 的子级是 result，其有 name 一个 attribute，表示一个结果分支，应当与 action 的返回结果保持一致，而其内容应当是 forward 的页面。

(6)

问题 1：执行 action 后导致 500 错误，原因，没有在 web.xml 中配置对应的 Filter。

2.2、实验二

struts2 中标签有很多种，从类型上大致分为四类，控制标签、数据标签、表单标签和 Ajax 标签，各个功能的概述如下

控制标签：

标签名	作用
s:if, s:elseif, s:else	一组分支选择语句，用于进行条件判断。

s:iterator	迭代输出其内容。
s:merge	以两个或多个列表为参数，用来合并列表，每个元素为一个多元组。
s:append	以两个或多个列表为参数，用来附加列表，每个列表的元素类型保持一致。
s:generator	创建一个迭代器，用来辅助 iterator 迭代。

数据标签：

s:action	通过指定的 action 名称和可选的命名空间直接从 jsp 页面调用 action。
s:include	用于在一个 jsp 页面中包含一个 jsp 页面。
s:bean	用于实例化一个符合 JavaBeans 规范的类，
s:date	以快速简单的方式格式化日期。
s:param	子属性，用于表示父级标签的参数。
s:property	用于获取一个值的属性
s:push	用于推送堆栈的值
s:set	用于为指定范围内的变量赋值
s:text	用于呈现 <code><n></code> 文本消息
s:url	用于创建 url 标签。

表单标签和 Ajax 标签：

struts 标签使用 dojo 框架来实现 ajax 标签。

2.3、实验三

struts.properties 的作用：struts.properties 文件用来定义大量的 struts 属性，其是一个规范的 key=value 键值对文件，常见的属性如下。

struts.configuration	用来指定加载 struts2 配置文件的配置文件管理器。
struts.locale	指定 Web 应用的默认 locale

struts.i18n.encoding	指定 Web 应用的默认编码集
struts.objectFactory	指定 struts2 默认的 ObjectFactory Bean
struts.multipart.parser	该属性指定处理 multipart/form-data 的 MIME 类型（文件上传）请求的框架，该属性支持 cos、pell 和 jakarta 等属性值
struts.multipart.saveDir	该属性指定上传文件的临时保存路径
struts.multipart.maxSize	该属性指定 struts2 文件上传中整个请求允许的最大字节数。
struts.custom.properties	该属性指定 struts2 应用加载用户自定义的属性文件
struts.custion.i18n.resources	该属性指定 struts2 应用所需要的国际化资源文件
struts.configuration	该属性指定加载 struts2 配置文件的配置文件管理器。