实验三 Struts2 基础应用——基于 Struts2 框架的用户登录与注册模块

一、基础实验——Struts2 框架搭建

（一）实验目的

1、掌握 Struts2 应用的基本开发步骤和常规配置；
2、观察表单参数与 Action 属性的赋值关系，观察 Action 的 execute()方法及其返回值，并能够正确应用；
3、观察配置文件 struts.xml 中的主要元素及属性，并能够正确应用；
4、理解 Struts2 框架中 MVC 设计模式的体现，理解 Action，FilterDispatcher，struts.xml 的主要作用，并能够正确应用。

（二）基本知识与原理

1、Struts2 是从 WebWork 框架上发展起来的 MVC 框架；
2、FilterDispatcher 是 Struts2 中的核心控制器，客户端对服务器端的请求将被 FilterDispatcher 过滤；若请求需要调用某个 Action，则框架将根据配置文件 struts.xml，找到需要调用的 Action 类；
3、Action 类是一个符合一定命名规范的 JavaSE 类，作为业务控制器使用；Action 中的 execute()方法用于调用 Model 层的业务逻辑类，并根据返回结果决定页面导航；
4、若 Action 类中需要使用表单提交的请求参数，那么必须在 Action 类中声明与表单域的名字对应的变量，并为变量提供 getters/setters 方法；
5、Action 类需要在 struts.xml 中进行配置才能使用；
6、编译运行基于 Struts2 框架的 Web 工程，需要导入 struts2 的 8 个核心 jar 包：

表 2-1 strut2 的 8 个核心 jar 包

文件名	说明
struts2-core-6.0.3.jar	Struts 2 框架的核心类库
ognl-3.3.3.jar	Struts 2 使用的一种表达式语言类库
log4j-api-2.18.0.jar	日志输出组件

freemarker-2.3.31.jar	Struts 2 的标签模板使用类库
javassist-3.29.0.GA.jar	代码生成工具包
commons-lang3-3.10.jar	Apache 语言包，是 java.lang 包的扩展
commons-text-1.8.jar	Apache 语言包，用于字符串处理
commons-io-2.9.1.jar	Apache IO 包
commons-fileupload-1.4.jar	Struts 2 文件上传依赖包

（三）实验内容及步骤

1、登录 http://struts.apache.org/download.cgi 站点，下载 Struts2 的最新版（Full Distribution）；

2、在 Eclipse 中新建 Web 工程 struts-prj1；

3、将 Struts2 中的 8 个核心包增加到 Web 应用中，即复制到“%workspace% struts-prj1\WebContent\WEB-INF\lib”路径下，如下图所示；

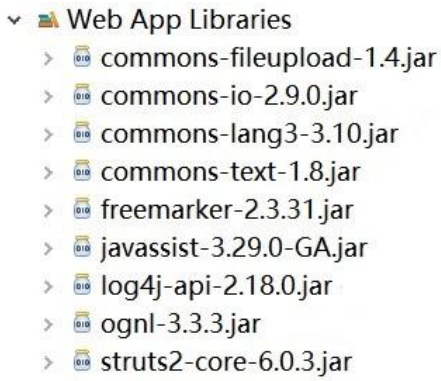

图 3-1 Struts2 核心包

4、在 struts-prj1 中新建 login.jsp 页面，作为用户登录的视图（部分代码如下），注意表单（form）中 action 属性的值，注意用户名与密码输入框中 name 属性的值；

```
<form action="login" method="post">
    请输入用户名：<input name="loginUser.username" type="text"><BR>
    请输入密码：<input name="loginUser.password" type="password">
    <input type="submit" value="登录">
</form>
```

5、在 struts-prj1 中新建 loginSuccess.jsp 和 loginFail.jsp 页面，分别作为登录成功或登录失败的视图，在页面中显示“登录成功”或“登录失败”；

6、在 struts-prj1 中新建 cn.edu.zjut.bean 包，并在其中创建 UserBean.java，用于记录登录用户信息（代码如下），注意该 JavaBean 中属性名的写法；

```
package cn.edu.zjut.bean;
public class UserBean {
```

```
    private String username="";
    private String password="";

    public String getUsername() {
        return username;
    }
    public void setUsername(String username) {
        this.username= username;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
}
```

7、在 struts-prj1 中新建 cn.edu.zjut.service 包，并在其中创建 UserService.java，用于实现登录逻辑，为简化登录逻辑，将登录成功的条件设置为：用户名和密码相同（代码如下）；

```
package cn.edu.zjut.service;
import cn.edu.zjut.bean.UserBean;
public class UserService {
    public boolean login(UserBean loginUser) {
        if (loginUser.getUsername().equals(loginUser.getPassword())){
            return true;
        }
        return false;
    }
}
```

8、在 struts-prj1 中新建 cn.edu.zjut.action 包，并在其中创建 UserAction.java，调用登录逻辑，并根据登录结果不同而返回不同的内容（代码如下），注意该 Action 中的属性名及相应的 getters 和 setters 方法、execute()方法及返回值；

```
package cn.edu.zjut.action;
import cn.edu.zjut.bean.UserBean;
import cn.edu.zjut.service.UserService;
public class UserAction {
    private UserBean loginUser;
    public UserBean getLoginUser() {
            return loginUser;
    }
    public void setLoginUser(UserBean loginUser) {
            this.loginUser = loginUser;
    }
```

```
    public String execute() {
        UserService userServ = new UserService();
        if (userServ.login(loginUser)) {
            return "success";
        }
        return "fail";
    }
}
```

9、在工程 struts-prj1 的 src 目录中创建 struts.xml 文件，用于配置 Action 并设置页面导航（部分代码如下），注意 action 标签中 name 属性和 class 属性的值，以及 result 子标签的属性；

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 6.0//EN"
    "http://struts.apache.org/dtds/struts-6.0.dtd">
<struts>
  <package name="strutsBean" extends="struts-default" namespace="/">
        <action name="login" class="cn.edu.zjut.action.UserAction">
            <result name="success">/loginSuccess.jsp</result>
            <result name="fail">/loginFail.jsp</result>
        </action>
  </package>
</struts>
```

10、编辑 Web 应用的 web.xml 文件，增加 Struts2 核心 Filter 的配置（部分代码如下）；

```
<!-- 定义 Struts2 的核心 Filter -->
<filter>
    <filter-name>struts2</filter-name>
      <filter-class>
           org.apache.struts2.dispatcher.filter.StrutsPrepareAndExec
      uteFilter
    </filter-class>
</filter>
<!-- 让 Struts2 的核心 Filter 拦截所有请求 -->
<filter-mapping>
    <filter-name>struts2</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
```

11、将 struts-prj1 部署在 Tomcat 服务器上；

12、通过浏览器访问 login.jsp 页面，并记录运行结果；

13、尝试对以上第 4、6、8、9 中的关键代码（粗体）进行修改，观察修改后的运行结果。

（四）实验要求

1、填写并上交实验报告，报告中应包括：
（1）运行结果截图；修改后的关键代码，及相应的运行结果或报错信息；
（2）根据实验过程，总结 jsp 页面、Action 类、Service 类、JavaBean、Filter 和 struts.xml 文件的作用，整理 Struts2 应用中从请求到响应的完整流程，思考并总结 Struts2 框架中 MVC 的体现；
（3）根据实验过程，总结表单参数与 Action 属性的赋值关系，并记录下来；
（4）根据实验过程，总结 Action 的 execute()方法的作用和特点，并记录下来；
（5）根据实验过程，查找相关资料，写出本实验中配置文件 struts.xml 里各元素及其属性的作用；
（6）碰到的问题及解决方案或思考；
（7）实验收获及总结。
2、上交程序源代码，代码中应有相关注释。

二、提高实验——Spring 和 Struts2 的整合

（一）实验目的

1、掌握 Spring 框架与 Struts2 框架整合的基本步骤，理解 Spring 容器对 Bean 实例的管理；
2、理解 Spring 容器对 Struts2 核心控制器 Action 的管理，并理解 Struts 配置文件和 web.xml 中相应产生的变化；
3、进一步理解 Spring 中控制反转 IoC 的核心机制；
4、掌握 Struts2 常用标签的基本使用方法；
5、能参考 Struts2 标签的使用说明文档，对各类标签进行灵活应用。

（二）基本知识与原理

1、使用 Spring 整合 Struts2 框架，其核心思想是将 Struts2 的 Action 实例交给 Spring 框架的 IoC 容器装配管理，因此在 Action 类中应提供必要的 setters 方法以注入所需的属性，同时 struts.xml 文件中的<action>元素的 class 属性将不再是该 Action 对应的实际类型，而是与 applicationContext.xml 中 Action 的 bean 的 id 对应；
2、使用 Spring 整合 Struts2 框架，还需要在 web.xml 文件中配置一个 listener 来

完成加载 Spring 配置文件的功能；

1、使用 Struts2 标签的形式来表达页面逻辑，可以尽量避免在视图中使用 Java 代码，让逻辑与显示分离，提高视图的可维护性；

2、Struts2 标签库的主要 tld 文件为 struts-tags.tld，在 struts2-core-*.*.*.jar 包中；

3、Struts2 标签的使用步骤和使用 JSTL 相同，只需在 JSP 页面中使用 taglib 指令引入标签库中 tld 文件的 uri，并指定前缀即可，例如：
<%@ taglib prefix="s" uri="/struts-tags"%>；

4、根据 Struts2 标签的主要作用，可以将其分为：用于生成页面元素的 UI 标签、用于实现流程控制的控制类标签、用于控制数据的数据标签和用于支持 Ajax 的标签。

（三）实验内容及步骤

1、将 common-logging-1.2.jar 和 Spring 的 4 个基础 JAR 包添加到工程中；

2、将 Struts2 对 Spring 进行支持的 JAR 包 struts2-spring-plugin-*.*.*.jar 添加到工程中；

3、将 Spring 支持 web 开发的 JAR 包 spring-web-*.*.*.RELEASE.jar 添加到工程中；

4、在 cn.edu.zjut.service 包中，提取 UserService 类的接口 IUserService，代码略；

5、在 cn.edu.zjut.action 包，修改 UserAction.java，提供 setters 方法以注入所需的 service 实例，代码片段如下：

```
package cn.edu.zjut.action;
import cn.edu.zjut.service.IUserService;

public class UserAction {
    private Customer loginUser;
    private IUserService userService = null;

    //省略 loginUser 的 getters/setters 方法

    public void setUserService(IUserService userService) {
        this.userService = userService;
    }

    public String execute() {
        if (userService.login(loginUser)) {
            return "success";
        }
        return "fail";
```

```
    }
}
```

6、在项目的 WebRoot/WEB-INF/路径下创建 Spring 配置文件 applicationContext.xml，并增加对 UserAction、UserService 实例的配置，代码片段如下：

```
<bean id="userAction" class="cn.edu.zjut.action.UserAction"
      scope="prototype">
   <property name="userService" ref="userService" />
</bean>
<bean id="userService" class="cn.edu.zjut.service.UserService"/>
```

7、修改 Struts2 配置文件 struts.xml，使得<action>元素的 class 属性与 Spring 配置文件 applicationContext.xml 中 Action 的 bean id 对应，代码片段如下：

```
<struts>
   <package name="strutsBean" extends="struts-default"
            namespace="/">
      <action name="login" class="userAction">
         <result name="success">/loginSuccess.jsp</result>
         <result name="fail">/loginFail.jsp</result>
      </action>
   </package>
</struts>
```

8、修改 Web 应用的 web.xml 文件，添加对 Spring 监听器的配置，代码片段如下：

```
<web-app>
  <listener>
     <listener-class>
           org.springframework.web.context.ContextLoaderListener
     </listener-class>
  </listener>
……
<web-app>
```

9、将 struts-prj1 重新布署在 Tomcat 服务器上，通过浏览器访问 login.jsp 页面，并记录运行结果；

10、在 struts-prj1 工程中增加用户注册功能，新建 register.jsp 页面作为用户注册的视图，页面使用 Struts2 的 UI 标签来生成表单元素，包括用户名、密码、确认密码、真实姓名、性别、生日、联系地址、联系电话和电子邮箱等，部分代码如下：

```
<%@ taglib prefix="s" uri="/struts-tags"%>
<html>
<head>
   <s:head theme="xhtml"/>
</head>
<body>
```

```
<s:form action="register" method="post">
    <s:textfield name="loginUser.username" label="请输入用户名"/>
    <s:password name="loginUser.password" label="请输入密码"/>
    ……
    <s:radio name="loginUser.sex" list="#{1 : '男', 0 : '女'}" label
        ="请输入性别"/>
    <s:textfield name="loginUser.birthday" label="请输入生日
(yyyy-MM-dd)">
        <s:param name="value">
            <s:date name="loginUser.birthday" format="yyyy-MM-dd"/>
        </s:param>
    </s:textfield>
    ……
    <s:submit value="注册"/>
    <s:reset value="重置"/>
</s:form>
</body>
</html>
```

11、在 struts-prj1 中新建 regFail.jsp 页面，作为注册失败的视图，在页面中显示“注册失败”，代码略；

12、在 struts-prj1 中新建 regSuccess.jsp 页面，作为注册成功的视图，使用 Struts2 的数据标签和控制标签来生成注册成功的信息，并将登录用户信息保存在会话范围内，部分代码如下：

```
<!-- 数据标签 property -->
<s:property value="loginUser.username"/>
<!-- 控制标签 if/else -->
<s:if test='loginUser.sex=="1"'>
    <s:text name="先生，"/>
</s:if>
<s:else>
    <s:text name="女士，"/>
</s:else>
您注册成功了!
<!-- 数据标签 set -->
<s:set var="user" value="loginUser" scope="session"/>
```

13、修改 UserBean.java，增加属性用于记录注册用户信息，部分代码如下：

```
public class UserBean {
    private String username="";
    private String password="";
    private String repassword="";
    private String name="";
    private String sex="";
    private String birthday="";
```

```
    private String address="";
    private String phone="";
    private String email="";
    //省略 getters/setters 方法
}
```

14、修改 UserService.java，增加用户注册逻辑，为简化注册逻辑，将注册成功的条件设置为：用户名、密码和确认密码相同，而且不为空字串，代码略；

15、修改 UserAction.java 中的 execute()方法，注释登录逻辑，调用注册逻辑，并根据注册成功与否而返回不同的内容，代码略；

16、修改 struts.xml 文件，对用户注册进行配置并设置页面导航，代码略；

17、将 struts-prj1 重新布署在 Tomcat 服务器上，通过浏览器访问 register.jsp 页面，并记录运行结果。

（四）实验要求

1、填写并上交实验报告，报告中应包括：

（1）运行结果截图；

（2）结合实验过程，总结 Spring 整合 Struts2 框架的关键步骤，并记录下来；

（3）根据实验步骤 5，总结本实验中的 UserAction 与基础实验中的写法关键区别，并记录下来；

（4）根据实验步骤 6，查找相关资料，总结配置文件 applicationContext.xml 中 bean 元素的 prototype 属性及其取值的含义，并记录下来；

（5）根据实验步骤 8，查找相关资料，总结 web.xml 文件中添加监听器的目的，并记录下来；

（6）应用各种 Struts2 标签的关键代码，及相应的运行结果或报错信息；

（7）根据实验过程，查找相关资料，总结 Struts2 中标签及其属性的作用和用法并记录下来；

（8）碰到的问题及解决方案或思考；

（9）实验收获及总结。

2、上交程序源代码，代码中应有相关注释。

三、扩展实验——Action 与 ActionSupport

（一）实验目的

1、掌握 Struts2 的 Action 类中自定义方法的使用；

2、掌握 Struts2 中 Action 类的不同调用方式和相应的配置方法；
3、了解 Action 接口的作用，理解 ActionSupport 类的作用；
4、掌握在 Struts2 中使用校验器或手工编码的方式，对请求参数进行数据校验的方法，掌握在 JSP 页面中显示错误信息和提示信息的方法；
5、掌握在 Action 中使用国际化资源文件的方法；
6、掌握 Struts2 内置类型转换器的作用和使用方法。

（二）基本知识与原理

1、Action 类中的默认方法名是 execute()方法，可以被自动调用；
2、在 Action 中也允许定义其它方法名，可以同时定义多个方法，分别处理不同的逻辑；
3、当 Action 中使用了自定义方法，则该 Action 就需要特定的配置，一般有四种调用方式：
（1）在 struts.xml 文件中通过 method 属性指定方法名；
（2）使用动态方法调用方式（DMI）；
（3）使用提交按钮的 method 属性；
（4）使用通配符配置 Action；
4、为了让用户开发的 Action 类更规范，Struts2 提供了一个 Action 接口，该接口定义了 Struts2 的 Action 处理类应该实现的规范；
5、Struts2 还为 Action 接口提供了一个实现类：ActionSupport，该类提供了若干默认方法，包括：默认的处理用户请求的方法（excute()方法）、数据校验的方法、添加校验错误信息的方法、获取国际化信息的方法等，部分重要方法列表如下：

表 3-1 ActionSupport 类的部分重要方法

方法名	说明
public String execute()	默认的处理用户请求的方法，直接返回 SUCCESS 字符串。
public void validate()	空的输入校验方法，常被 Action 类覆盖，实现对输入参数的校验。
public void addActionError(String anErrorMessage)	将 Action 级别的错误信息添加到 Action 中。
public void addActionMessage(String aMessage)	将 Action 级别的消息添加到 Action 中。
public void addFieldError(String	将域级错误信息添加到特定的域中。

fieldName, String errorMessage)	
public String getText(String aTextName)	从国际化资源文件中获取属性值，其中参数是属性文件的 key 值

6、Struts2 框架提供了校验器和手工编码两种方式对请求参数进行数据校验，当 Action 类继承了 ActionSupport 类，就可以通过定义名为“<ActionClassName>-<ActionAliasName>-validation.xml”的校验规则文件的方法进行校验器校验，也可以通过重写 ActionSupport 类的 validate()方法或 validateXxx()方法进行手动校验；

7、在 JSP 页面中使用 Struts2 标签生成的表单，能将域级别的错误信息将自动显示到表单元素处；

8、在 JSP 页面中使用 fielderror 标签，可以集中显示所有的域级错误信息；使用 actionerror 标签，可以显示所有的 Action 级别错误信息；使用 actionmessage 标签，可以显示 Action 消息；

9、Struts2 框架中提供了部分内置的类型转换器，可以将请求参数的 String 类型转换成基本数据类型及对应的包装器类型、日期类型、数组类型、集合类型等，当 Action 类继承了 ActionSupport 类，则内置的类型转换器将默认生效，可以直接使用；

10、如需修改默认的类型转换校验信息，则要在 Action 类的包中声明名为“Action 类名.properties”的局部属性文件；

11、Struts2 框架同时支持自定义类型转换器，将请求参数转换成任意一种类型。

（三）实验内容及步骤

1、在 cn.edu.zjut.action 包，修改 UserAction.java，定义 login()方法和 register()方法，分别用于调用登录逻辑和注册逻辑，部分代码如下：

```
package cn.edu.zjut.action;
……
public class UserAction {
    ……
    public String login() {
        if (userService.login(loginUser)) {
            return "success";  }
        return "fail";
    }
    public String register() {
        if (userService.register(loginUser)) {
            return "regsuccess";  }
        return "regfail";
```

```
    }
}
```

2、修改 struts.xml 文件，通过 action 标签中 method 属性指定方法名，部分代码如下：

```
<struts>
  <package name="strutsBean" extends="struts-default" namespace="/">
    <action name="login" class="cn.edu.zjut.action.UserAction"
            method="login">
        <result name="success">/loginSuccess.jsp</result>
        <result name="fail">/loginFail.jsp</result>
    </action>
    <action name="register" class="cn.edu.zjut.action.UserAction"
            method="register">
        <result name="regsuccess">/regSuccess.jsp</result>
        <result name="regfail">/regFail.jsp</result>
    </action>
  </package>
</struts>
```

3、将 struts-prj1 重新部署，通过浏览器访问 login.jsp 与 register.jsp 页面，并记录运行结果；

4、查找相关资料，尝试使用 Action 自定义方法的其它三种调用和配置方式：动态方法调用方式（DMI）、提交按钮的 method 属性、通配符配置 Action，并记录关键配置和运行结果；

5、修改 UserAction 类，使其继承 ActionSupport 类，并在 UserAction 类中覆盖 ActionSupport 类的 validateXxx()方法，用于对用户登录的请求参数 username 和 password 进行校验：若用户名或密码为空，则使用 addFieldError（域级）添加错误信息，部分代码如下：

```
import com.opensymphony.xwork2.ActionSupport; ……
public class UserAction extends ActionSupport {
    ……
  public void validateLogin() {
    String username = loginUser.getUsername();
    String pwd = loginUser.getPassword();
    if (username == null || username .equals("")) {
        this.addFieldError("loginUser.username", "请输入您的用户名！");
    }
    if (pwd == null || pwd.equals("")) {
        this.addFieldError("loginUser.password", "请输入您的密码！");
    }
  }
}
```

6、修改 struts.xml 文件，在 login action 的配置中增加 validateXxx()方法校验出错时的页面导航（<result name="input">），部分代码如下：

```
<struts>
  <package name="strutsBean" extends="struts-default" namespace="/">
    <action name="login" class="cn.edu.zjut.action.UserAction"
            method="login">
        <result name="success">/loginSuccess.jsp</result>
        <result name="fail">/loginFail.jsp</result>
        <result name="input">/login.jsp</result>
    </action>
    ……
  </package>
</struts>
```

7、重新将 struts-prj1 部署在 Tomcat 服务器上，通过浏览器访问 login.jsp 页面，观察并记录运行结果；

8、修改 login.jsp 页面，在表单前增加 fielderror 标签：<s:fielderror/>，再通过浏览器访问 login.jsp 页面，观察并记录运行结果；

9、修改 UserAction.java，在调用登录逻辑的 login()方法中，对登录情况进行校验：若登录成功，使用 addActionMessage()方法添加“登录成功！”的 Action 提示消息，若登录失败，使用 addActionError()方法添加 Action 级别的错误信息，部分代码如下：

```
public class UserAction extends ActionSupport {
    ……
    public String login() {
        if (userServ.login(loginUser)) {
            this.addActionMessage("登录成功！");
            return "success";
        } else {
            this.addActionError("用户名或密码错误，请重新输入！");
            return "fail";
        }
}
```

10、修改 login.jsp 页面，增加 actionerror 标签（<s:actionerror/>）Action 级别的错误信息；修改 loginSuccess.jsp，使用 actionmessage 标签（<s:actionmessage/>）显示 Action 提示消息；

11、修改 struts.xml 文件中用户登录的页面导航设置，将登录失败时转向的页面从 loginFail.jsp 修改为 login.jsp；

12、重新将 struts-prj1 部署在 Tomcat 服务器上，通过浏览器访问 login.jsp 页面，观察并记录运行结果；

13、在工程 struts-prj1 中创建“UserAction-login-validation.xml”校验规则文件，

使其与 UserAction 类位于同一目录下，配置校验信息，使用校验器对请求参数进行校验，具体代码如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE validators PUBLIC
    "-//Apache Struts//XWork Validator 1.0.2//EN"
    "http://struts.apache.org/dtds/xwork-validator-1.0.2.dtd">

<validators>
    <field name="loginUser.account">
        <field-validator type="requiredstring">
            <param name="trim">true</param>
            <message>用户名不能为空</message>
        </field-validator>
    </field>
    <field name="loginUser.password">
        <field-validator type="requiredstring">
            <param name="trim">true</param>
            <message>密码不能为空</message>
        </field-validator>
    </field>
</validators>
```

14、重新将 struts-prj1 部署在 Tomcat 服务器上，通过浏览器访问 login.jsp 页面，观察并记录运行结果；

15、修改 UserBean.java，将用于保存注册用户生日的变量类型改为 Date 类型，使用 Struts2 内置的类型转换器对请求参数进行校验；

16、修改 struts.xml 文件，在 register action 的配置中增加类型转换出错时的页面导航（<result name="input">），将其返回至注册页面；

17、重新将 struts-prj1 部署在 Tomcat 服务器上，通过浏览器访问 register.jsp 页面，当用户输入的生日不合法时，观察并记录运行结果；

18、在 cn.edu.zjut.action 包中创建局部属性文件“UserAction.properties”，修改类型转换的校验信息，具体代码如下：

```
#其中 invalid.fieldvalue 不能随意修改，loginUser.birthday 是请求参数域名，
#应根据实际需要进行修改
invalid.fieldvalue.loginUser.birthday=生日必须是日期，并符合
“yyyy-mm-dd”格式
```

19、重新将 struts-prj1 部署在 Tomcat 服务器上，通过浏览器访问 register.jsp 页面，当用户输入的生日不合法时，观察并记录运行结果；

20、在 cn.edu.zjut.action 包中创建 UserAction -register-validation.xml 文件，增加校验信息的配置，使用校验器对用户注册的请求参数进行校验，要求注册时两次密码输入相同、email 地址格式符合要求等（参考实验步骤 13）；

21、重新将 struts-prj1 部署在 Tomcat 服务器上，通过浏览器访问 register.jsp 页面，观察并记录运行结果。

（四）实验要求

1、填写并上交实验报告，报告中应包括：

（1）运行结果截图；

（2）查找相关资料，根据实验过程，总结 Action 自定义方法的四种调用和配置方式；

（3）根据实验过程，查找相关资料，总结 validate()方法或 validateXxx()方法的作用、使用时的要点或注意事项，总结在 JSP 页面中显示错误信息和提示信息的方法，并记录实验中相应的关键代码；

（4）根据实验过程，总结校验器校验的使用和配置方法，并结合相应案例将其记录下来；

（5）根据实验过程，总结 Struts2 中常用的内置类型转换器及其使用方法；

（6）碰到的问题及解决方案或思考；

（7）实验收获及总结。

2、上交程序源代码，代码中应有相关注释。