



使用 Struts 创建 Web 应用程序

目录

使用 Struts 创建 Web 应用程序 1

简介: 使用 Struts 创建 Web 应用程序 1

课程 1: 创建 Struts 项目 2

 课程要点 3

课程 2: 创建 Web 图节点 3

 课程要点 5

课程 3: 创建表单 bean 5

 课程要点 8

课程 4: 连接 Web 图中的节点 8

 课程要点 10

课程 5: 编辑表单 bean 10

 课程要点 11

课程 6: 编辑操作映射 11

 课程要点 13

课程 7: 对 JavaServerPages (JSP) 页添加元素 . . . 13

 输入页 13

 输出页 15

 课程要点 15

课程 8: 显示错误消息 15

 课程要点 17

课程 9: 测试 Struts 应用程序 17

 课程要点 18

总结: 使用 Struts 创建 Web 应用程序 18

使用 Struts 创建 Web 应用程序

在本教程中，您将学习如何创建具有两个 Web 页面、一个操作映射和一个表单 bean 的简单 Struts 应用程序。将使用 Web 图编辑器来帮助您使应用程序的逻辑流可视化。在创建应用程序之后，您将了解如何设置运行时测试环境和测试应用程序。

学习目标

您将学习如何完成下列任务：

- 创建 Struts 项目
- 编辑 Web 图
- 创建并编辑表单 bean
- 创建输入和输出 JavaServer Pages (JSP) 文件
- 创建操作和操作映射
- 将 `ActionError` 标记添加至输入 JSP 页
- 测试 Struts 应用程序

60 分钟

相关信息



[查看 PDF 版本](#)

简介：使用 Struts 创建 Web 应用程序

在本教程中，您将学习如何创建具有两个 Web 页面、一个操作映射和一个表单 bean 的简单 Struts 应用程序。将使用 Web 图编辑器来帮助您将应用程序的逻辑流可视化。最终应用程序显示一个输入 Web 页面，用户可以在该页面输入年、月、日。如果输入内容有效，就会计算出当天是星期几并显示在一个输出 Web 页面上。如果输入内容无效，会在输入 Web 页面上显示错误消息。您将了解如何设置运行时测试环境和测试应用程序。

学习目标

您将学习如何完成下列任务：

- 创建 Struts 项目
- 编辑 Web 图
- 创建并编辑表单 bean
- 创建输入和输出 JavaServer Pages (JSP) 文件
- 创建操作和操作映射
- 将 `ActionError` 标记添加至输入 JSP 页
- 测试 Struts 应用程序

所需时间

完成本教程大约要花 **60 分钟**。如果要学习与本教程有关的其他概念，也许要花更长的时间才能完成。

技能级别

高级

适用对象

Web 应用程序开发者和 Web 用户界面设计人员

系统要求

为了完成本教程，在本地机器上需要配置下列服务器之一：

- WebSphere® Application Server 5.1 测试环境或远程服务器
- WebSphere Application Server 6.x 运行时服务器
- Apache Tomcat 5.x 运行时服务器

先决条件

要完成本教程，您必须熟悉下列方面的知识：

- 基本的 Web 设计概念，如站点、页面、浏览器和服务
- Web 页面元素，如表、超链接、表单和图像
- 编辑用于 Web 页面的 HTML 代码
- 基于 Eclipse 的产品中的工作台透视图和视图

课程 1: 创建 Struts 项目

创建支持 Struts 的动态 Web 项目。

Struts 是一个用于构建 Java™ Web 应用程序的开放式源代码软件框架。Struts 使用模型 - 视图 - 控制器 (MVC) 应用程序开发设计模式。MVC 设计模式通过将应用程序分为下列三个部分来帮助您创建和维护复杂的 Web 应用程序：

- 模型用于维护应用程序中的数据和业务逻辑。由于本教程不使用数据库，仅有的数据都是由用户输入的，所以本教程中没有模型。
- 视图是用来向用户显示信息的界面。在本教程中，视图由两个 Web 页面组成，它们用于收集用户输入的数据并对用户显示结果。
- 控制器用于处理应用程序中的事件。在本教程中，控制器由操作映射和表单 bean 组成，它们协同工作以处理用户输入的信息。

提示：要了解关于 Struts 技术的更多信息，请访问 <http://struts.apache.org/>。

要创建新的基于 Struts 的 Web 项目：

1. 在菜单栏中选择 **文件** → **新建** → **项目** 以打开新建项目窗口。
2. 选择 **Web** → **动态 Web 项目** 并单击 **下一步** 以打开新建动态 Web 项目向导。
3. 在 **项目名称** 字段中输入 **DayOfWeek**。
4. 在 **目标运行时** 字段中选择已配置的运行时。单击 **下一步**。

注：如果尚未配置运行时环境，则单击 **新建** 以打开新建服务器运行时向导。对于本教程，可以使用下列服务器之一：

- WebSphere Application Server 5.1 测试环境或远程服务器

2 使用 Struts 创建 Web 应用程序

- WebSphere Application Server 6.x 运行时服务器
- Apache Tomcat 5.x 运行时服务器

(有关配置 Apache Tomcat 服务器的更多信息, 请参阅 <http://tomcat.apache.org/download-55.cgi>。)

5. 在“项目构面”页面上, 单击 **Struts**, 然后单击下一步。
6. 在“Web 模块”页面上, 单击下一步。
7. 在“Struts 设置”页面上, 单击完成。
8. 如果您不在 J2EE 透视图中, 则单击是以打开此透视图。
9. 如果这是您第一次创建 Web 项目, 则单击确定以启用必需的产品功能。

可以在“项目资源管理器”中查看新的 Struts Web 项目。将打开相应的 Web 图。

提示: 在 Web 图上选择“欢迎使用 Web 图编辑器”对象, 然后按 Delete 键以删除该对象。



课程要点

您学习了如何创建 Struts Web 项目、选择运行时环境以及打开新的 Web 图。

课程 2: 创建 Web 图节点

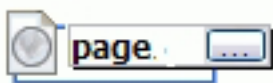
通过本课程学习如何向新项目的 Web 图中添加节点。

Web 图可帮助您使 Web 应用程序的流可视化和更改这些流。在使用 Web 图编辑器之前, 您应当了解下列术语:

- 节点表示资源, 例如, Web 页面、Java bean、Struts 操作或 Web 应用程序。
- 连接表示两个资源之间的逻辑流。

要在 Web 图上创建用来表示两个 Web 页面和一个 Struts 操作的节点:

1. 打开选用板的 Web 部件抽屉。将一个 Web 页面节点拖到您的 Web 图中。该图将显示新的 Web 页面节点和缺省名称 page.jsp。



提示: 在 Web 图中创建新节点时, 将自动选择该节点的名称然后进行编辑。如果在输入新名称之前单击图中的其他位置, 则新的 Web 页面名称将保持不变。要将某个节点重命名, 单击以突出显示该名称, 然后输入新名称。

2. 输入 `input.jsp` 作为该 Web 页面节点的新名称。

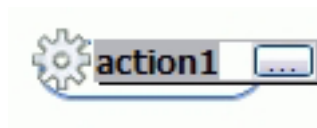


注: 一旦命名 Web 页面节点, 就会在 `DayOfWeek\WebContent` Web 项目文件夹中创建此节点所表示的 `JavaServer Pages (JSP)` 文件。

3. 将另一个 Web 页面节点拖到该 Web 图中。
4. 输入 `output.jsp` 作为第二个 Web 页面的名称。



5. 将一项 Struts 操作拖到该图中以创建一个称为 `action1` 的节点。

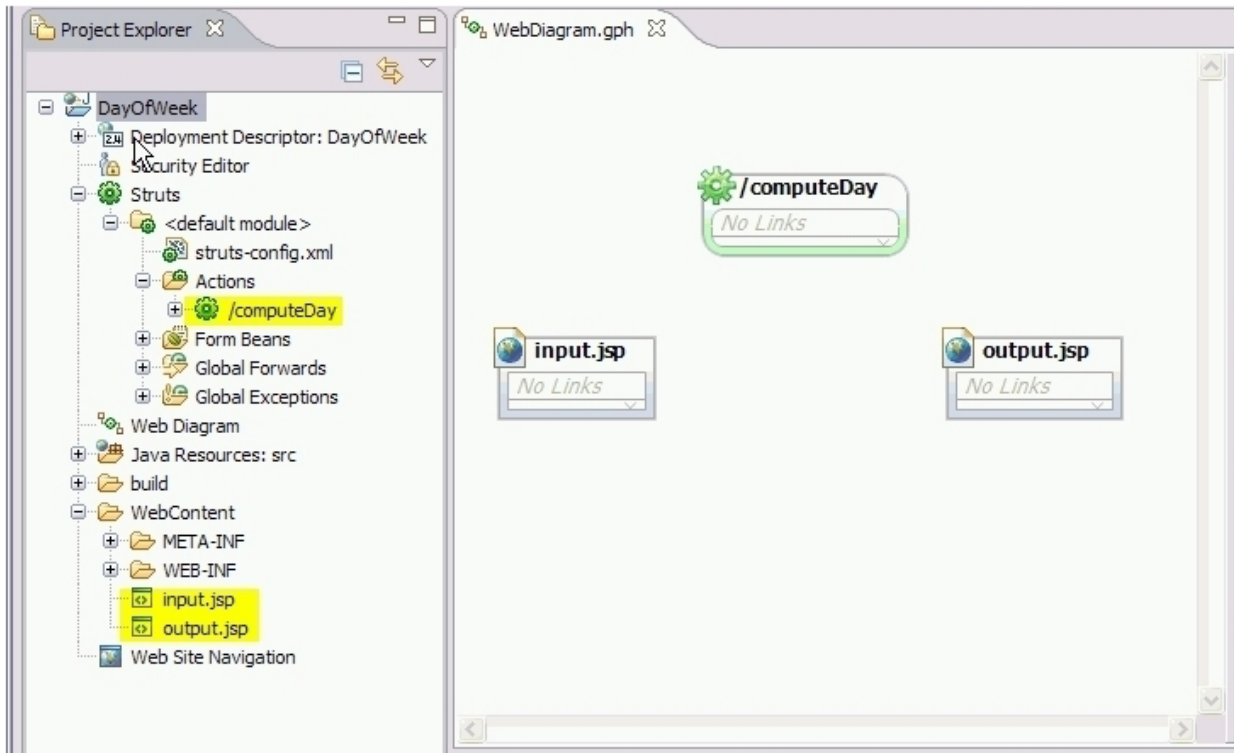


6. 输入 `computeDay` 作为该操作的名称, 然后按 **Enter** 键。Web 图中将显示名为 `/computeDay` 的操作节点。
要在“项目资源管理器”中查看 `/computeDay` 操作, 浏览至 Web 项目文件夹, 然后浏览至 `DayOfWeek\Struts\<default module>\Actions`。



7. 保存 Web 图。

您的 Web 项目中应包含下列各项: `/computeDay`、`input.jsp` 和 `output.jsp`。



课程要点

在本课中，您添加了输入 JSP 文件、输出 JSP 文件和 /computeDay 操作。

您学习了如何完成下列任务：

- 将 JSP 文件和 Struts 操作添加至 Web 图。
- 在“项目资源管理器”中查找 JSP 文件和 Struts 操作。

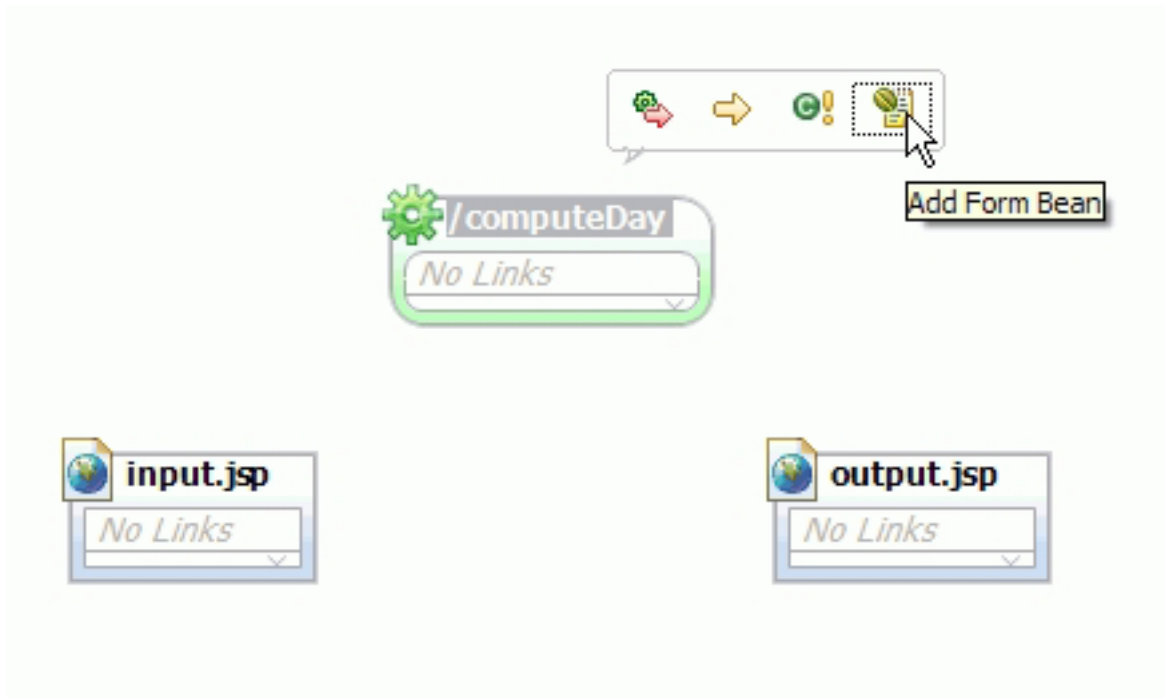
课程 3: 创建表单 bean

学习如何创建表单 bean。

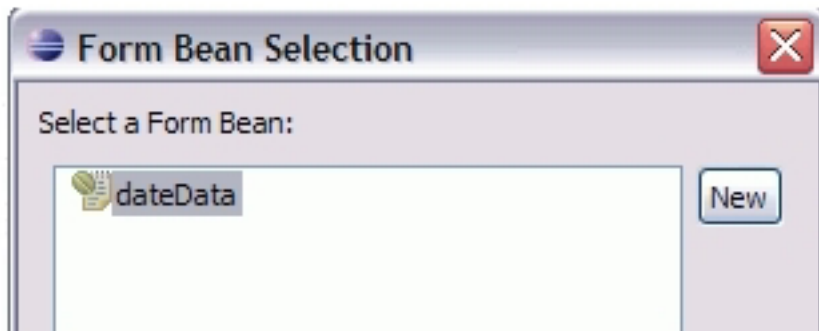
表单 *bean* 是一种 Java bean。表单 bean 存储来自所提交客户端请求的 HTML 表单数据，或者存储来自用户所单击 Struts 操作链接的输入数据。

要创建表单 bean：

1. 在图中，将鼠标指针置于 /computeDay 操作上并单击**添加表单 Bean**。



2. 在选择表单 Bean 窗口中，单击**新建**以打开新建表单 Bean 向导。
3. 在表单 **Bean** 名称字段中输入 dateData。单击下一步。



4. 单击**下一步**以跳过下一个向导页。由于是新的应用程序，因此项目中的页面尚不包含任何表单信息。如果它们包含任何表单信息，则可以在此处为此信息填充字段。
5. 在用于为 ActionForm 类创建新字段的页面上，使用**添加**按钮添加下列字段并单击“下一步”：

名称	类型
year	整数
month	整数
day	整数
dayOfWeek	字符串

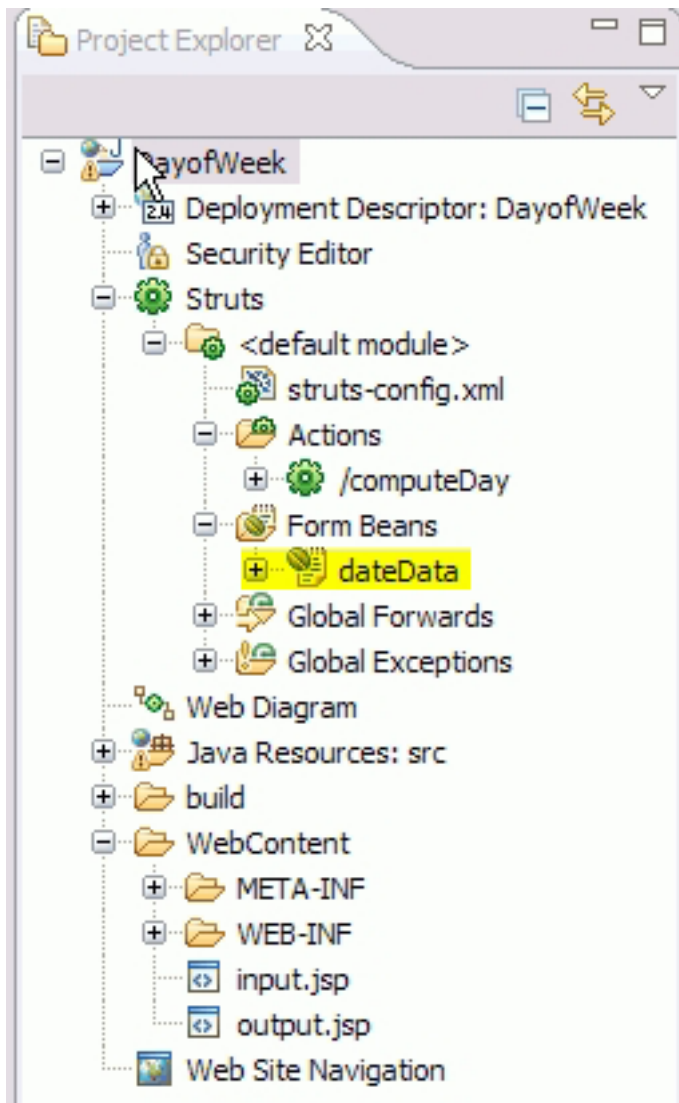
6. 在用来为 ActionForm 类创建映射的页面上，在 **Java 包** 字段中输入 com.ibm.dayofweek。单击**完成**。将显示已更新的选择表单 Bean 窗口。

7. 在选择表单 Bean 窗口中，单击刚刚创建的 dateData 表单 bean。单击**确定**。
8. 在 /computeDay 节点中，双击位于该节点右下角附近的小三角形。“数据”部分将展开，以显示与该节点相关联的 dateData 表单 bean。



已将 dateData 表单 bean 添加至 /computeDay 节点，并且在 Web 项目文件夹中创建了 dateData.java 文件。在“项目资源管理器”中，浏览至 DayOfWeek\Struts\<default module>\Form Beans 文件夹以查看该文件。

在本课程结束时，“项目资源管理器”类似于下图：



课程要点

在本课程中，创建了 `dateData` 表单 bean 并且定义了 `ActionForm` 类。

您学习了如何完成下列任务：

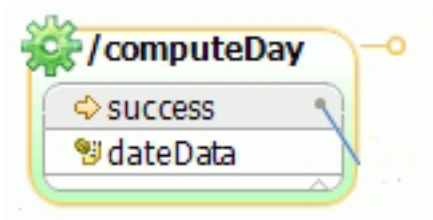
- 启用悬浮菜单。
- 定义 `ActionForm` 类的参数。
- 在“项目资源管理器”的 Web 项目文件夹中查找 Java 文件。

课程 4：连接 Web 图中的节点

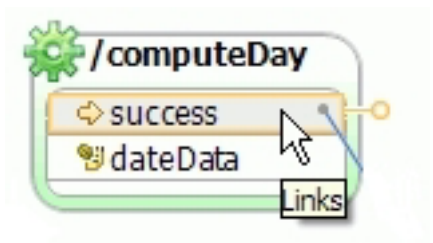
学习如何使用连接柄来连接 Web 图中的节点。

如果已经将节点放入 Web 图中，下一步的任务就是连接这些节点了。您将使用连接柄来完成此任务。有两种类型的连接柄：

- 可以使用顶级连接柄来绘制源自节点的连接。



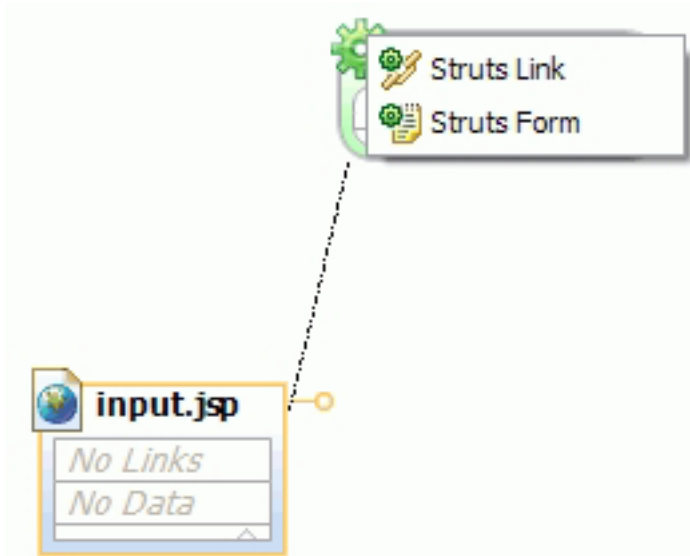
- 可以使用嵌套连接柄来绘制源自节点部分中某项的连接。



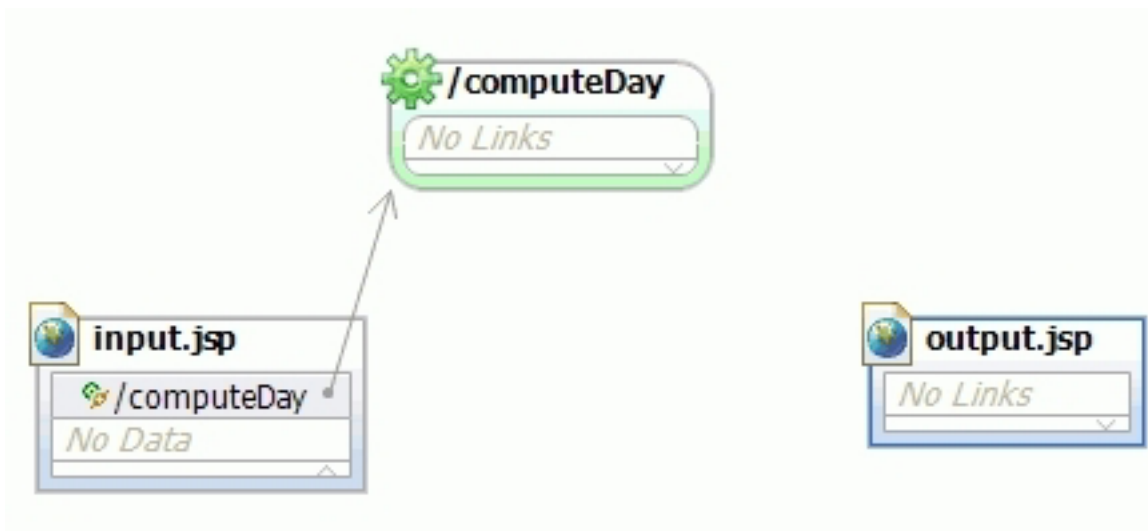
1. 在图中，将鼠标指针置于 `input.jsp` 节点上。将突出显示该节点，并且显示了一个连接柄，还显示了一个悬浮操作菜单。



2. 单击连接柄并将它从输入 JSP 节点拖至 `/computeDay` 操作。将显示一个弹出菜单。



3. 选择 **Struts 表单** 以在处理表单时将输入提交给操作。这就会将 Struts 表单添加至输入 JSP 节点，并且有一条线从该节点连接至 /computeDay 操作。



提示： 还可以使用悬浮菜单对输入 JSP 节点添加 Struts 链接，然后将链接的连接柄拖至 /computeDay 操作。

4. 使用本地转发将 /computeDay 操作连接至 output.jsp 节点：
 - a. 将鼠标指针置于 /computeDay 节点上以显示悬浮菜单。
 - b. 单击**添加本地转发**图标。这就会将名为“成功”的本地转发添加至该节点。缺省情况下，在每个节点中创建的第一个本地转发的名称为“成功”。
 - c. 将连接柄从“成功”链接拖至输出 JSP 节点。这就在此链接与节点之间绘制了一条线来显示连接。

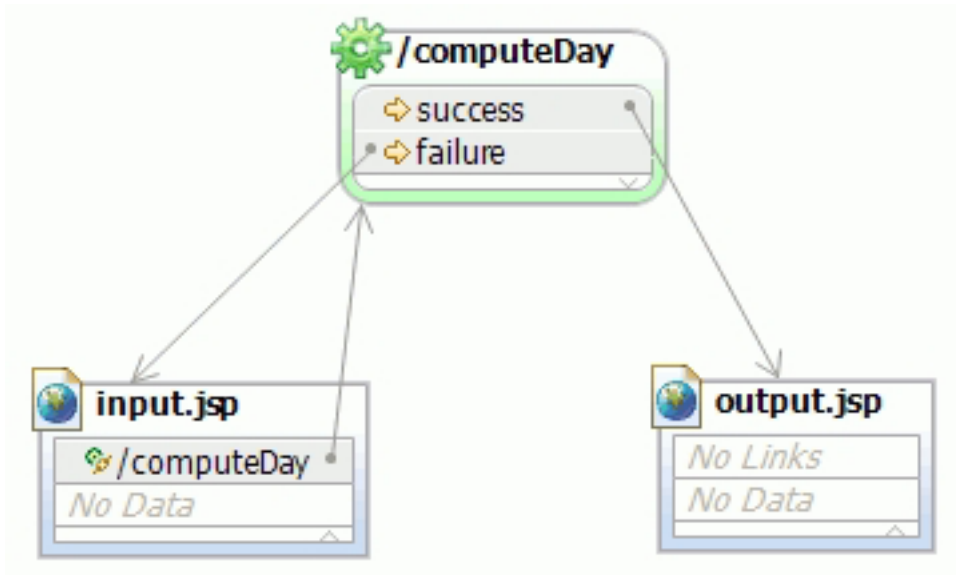
提示： 节点的“链接”部分中的每一项都还有一个连接柄。当单击某个节点以抓取其连接柄时，应确保您拖动的是节点的连接柄而不是链接的连接柄。

5. 使用本地转发将 /computeDay 节点连接至输入 JSP 节点：
 - a. 将鼠标指针置于 /computeDay 节点上，并将它的连接柄拖至输入 JSP 节点。

- b. 在悬浮菜单上选择**本地转发**。这就会将名为“失败”的本地转发添加至该节点，并且有一条线从此本地转发连接至输入 JSP 节点。缺省情况下，在每个节点中创建的第二个本地转发的名称为“失败”。

6. 保存 Web 图。

Web 图编辑器中应显示一个 Struts 链接和两个本地转发。



课程要点

很好！您已经将 Web 图中的节点连接起来了。

您学习了如何完成下列任务：

- 选择顶级连接柄或嵌套连接柄以绘制连接。
- 添加从输入 JSP 节点到操作节点的 Struts 链接。
- 添加从操作节点到输出 JSP 节点的本地转发。

课程 5: 编辑表单 bean

编辑表单 bean 源文件和 Java 资源文件以使它们在 Struts 应用程序中正常工作。

1. 在“项目资源管理器”中，双击 /computeDay 节点中的 dateData 表单 bean，以在 Java 编辑器中打开。
2. 在该编辑器中，在文件底部附近查找以下代码行：

```
ActionErrors errors = new ActionErrors();
```

3. 紧接着此代码行下面插入以下代码：

```
if (year < 1582)
{
    errors.add("year",new org.apache.struts.action.ActionError("pre_gregorian"));
}
```

4. 保存并关闭 DateData.java 文件。
5. 在“项目资源管理器”中，浏览至 DayOfWeekJava\Resources: src\com.ibm.dayofweek.resources 文件夹。
6. 双击 ApplicationResources.properties 文件。
7. 在 ApplicationResources.properties 文件中，除去以 errors.header 和 errors.footer 开头的那些行中的 # 字符。

8. 在该文件末尾添加以下代码:

```
pre-gregorian=<li>Date is before 1582, the year the Gregorian calendar began</li>
```

```
# Optional header and footer for <errors/> tag.
errors.header=<ul>
errors.footer=</ul>
pre-gregorian=<li>Date is before 1582, the year the Gregorian calendar began</li>
```

9. 保存并关闭该文件。

课程要点

在本课程中, 您编辑了表单 bean 的源文件。

您学习了如何完成下列任务:

- 查找表单 bean 的源文件。
- 编辑源代码。

课程 6: 编辑操作映射

操作映射表示一项在运行的操作。当它正在运行时, 它将确定应用程序的流。

操作映射是一个配置文件条目, 通常包含对操作类的引用。此条目可以包含对某操作可使用的表单 bean 的引用, 此外, 还可以定义仅对此操作可视的本地转发和异常。为 JavaServer Pages (JSP) 页编写代码时, 可以引用操作映射以触发某项操作。

操作类中包含在提交表单时就会调用的 `execute` 方法。在此方法中, 可以提供逻辑以确定计算结果是“成功”(由指向输出 JSP 页的转发来定义)还是“失败”(由指向输入 JSP 页的转发来定义)。

要编辑操作映射:

1. 在“项目资源管理器”中, 浏览至 `DayOfWeek\Java Resources: src\dayofweek.actions` 文件夹。
2. 双击 `ComputeDayAction.java` 文件以将它打开。
3. 在该文件中, 选择所有文本并按 **Delete** 键以删除这些文本。
4. 复制以下代码并将它粘贴到该文件中:

```
package dayofweek.actions;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionError;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;

import com.ibm.dayofweek.DateData;

/**
 * @version 1.0
 * @author
 */

public class ComputeDayAction extends Action {
```

```

/**
 * Constructor
 */
public ComputeDayAction() {

    super();

}

public ActionForward execute(
    ActionMapping mapping,
    ActionForm form,
    HttpServletRequest request,
    HttpServletResponse response)
    throws Exception {

    ActionErrors errors = new ActionErrors();
    ActionForward forward = new ActionForward();
    // return value
    DateData dateData = (DateData) form;
    int year;
    int month;
    int day;
    int dayOfWeek;
    int valcen;
    int valleap;
    int valyear;
    int valmon;
    int valday;
    int[] centuries = new int[4];
    int[] months = new int[13];
    String[] daysOfWeek = new String[7];
    centuries[0] = 2;
    centuries[1] = 0;
    centuries[2] = 5;
    centuries[3] = 3;
    months[1] = 5;
    months[2] = 1;
    months[3] = 0;
    months[4] = 3;
    months[5] = 5;
    months[6] = 1;
    months[7] = 3;
    months[8] = 6;
    months[9] = 2;
    months[10] = 4;
    months[11] = 0;
    months[12] = 2;
    daysOfWeek[0] = "Sunday";
    daysOfWeek[1] = "Monday";
    daysOfWeek[2] = "Tuesday";
    daysOfWeek[3] = "Wednesday";
    daysOfWeek[4] = "Thursday";
    daysOfWeek[5] = "Friday";
    daysOfWeek[6] = "Saturday";

    try {
        day = dateData.getDay();
        month = dateData.getMonth();
        year = dateData.getYear();
        if (month < 3) {
            year--; // Subtract 1 from year
        }
        valcen = centuries[year / 100 % 4];
        valleap = year % 100 / 4;
        valyear = year % 100 % 7;
    }
}

```

```

        valmon = months[month];
        valday = day % 7;
        dayOfWeek = valcen + valleap + valyear + valmon + valday;
        dayOfWeek = dayOfWeek % 7;
        dateData.setDayOfWeek(daysOfWeek[dayOfWeek]);
        request.setAttribute("dateData", dateData);

    } catch (Exception e) {

        // Report the error using the appropriate name and ID.
        errors.add("name", new ActionError("id"));
        e.printStackTrace();
    }

    // If a message is required, save the specified key(s)
    // into the request for use by the tag.

    if (!errors.isEmpty()) {
        saveErrors(request, errors);
        forward = mapping.findForward("failure");
    } else {
        forward = mapping.findForward("success");
    }

    // Finish with
    return (forward);
}
}

```

提示： 您的项目中的包名可能不是 `dayofweek.actions`。在这种情况下，应将上述第一个代码行中的 `dayofweek.actions` 替换为您的包名。

5. 保存并关闭该文件。

课程要点

现在，操作类中包含在提交 Web 表单时就会调用的 `execute` 方法。您已经创建了逻辑以确定计算结果是“成功”（由指向输出 JSP 页的转发来定义）还是“失败”（由指向输入 JSP 页的转发来定义）。

您学习了如何完成下列任务：

- 向操作类添加 `execute` 方法。
- 创建逻辑以确定操作是成功还是失败。

课程 7：对 JavaServerPages (JSP) 页添加元素

对 JSP 页添加输入字段、按钮和输出字段，以创建用于输入和输出的动态用户界面。

在输入字段中，最终应用程序的用户将输入感兴趣的年月日。输入页中包含“提交”和“刷新”按钮。在输出字段中，将显示所输入的年月日是星期几。要创建这些页：

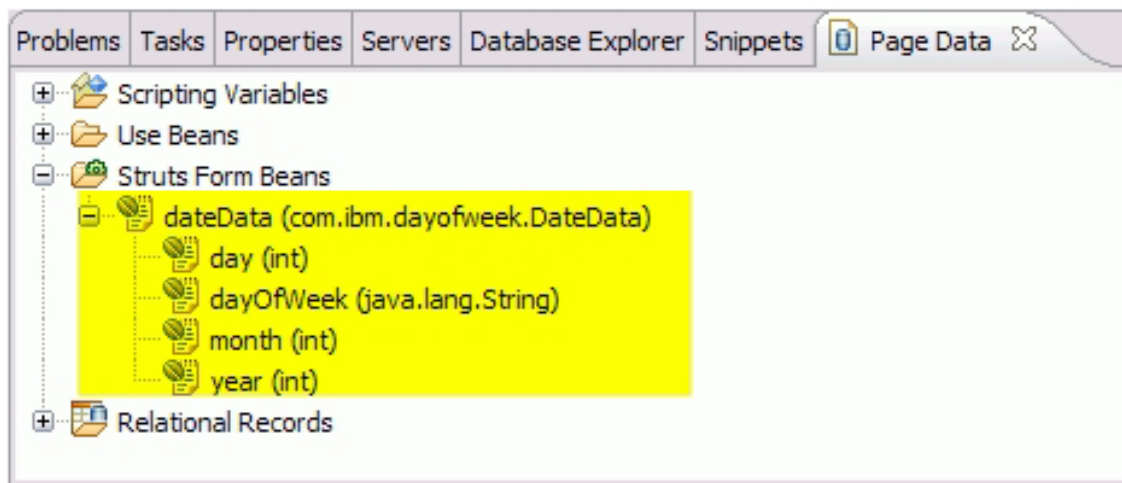
1. 对输入 JSP 页添加元素。
2. 对输出 JSP 页添加元素。

输入页

要对输入 JSP 页添加字段和按钮：

1. 在 Web 图中，双击 `input.jsp` 页以在 Page Designer 中打开。

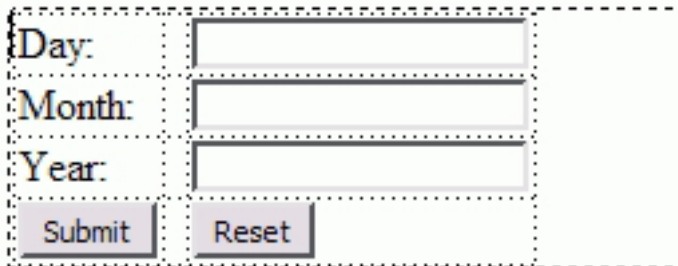
2. 在 Page Designer 中，单击“设计”选项卡。
3. 在主菜单栏中，选择窗口 → 显示视图 → 其他。
4. 在此窗口中，选择 **Web** → 页数据，然后单击确定以打开“页数据”视图。
5. 在该“页数据”视图中展开 Struts 表单 Bean 文件夹。查找 dateData 表单 bean 和已定义的字段。



6. 将“日”、“月份”和“年份”字段从“页数据”视图拖至 Page Designer 中的输入 JSP 页。 将打开插入数据对象窗口。



7. 选择位于顶部的**更新现有记录**。
8. 单击**选项**，并确保选中了“**提交**”按钮和“**删除**”按钮复选框。然后，单击**确定**。
9. 单击**完成**以在表单中创建这些字段和按钮。



10. 保存并关闭该输入 JSP 文件。

输出页

要对输出 JSP 页添加字段：

1. 在 Web 图中，双击 output.jsp 页以在 Page Designer 中打开。
2. 在 Page Designer 中，单击“设计”选项卡。
3. 在该“页数据”视图中展开 Struts 表单 Bean 文件夹。查找 dateData 表单 bean 和已定义的字段。
4. 将“星期”字段从“页数据”视图拖至 Page Designer 中的输出 JSP 页。将显示插入数据对象窗口。
5. 选择位于顶部的**显示现有记录**并单击**完成**，以在该页中创建字段。
6. 保存并关闭该输出 JSP 文件。

课程要点

在此课程中，对输入 JSP 页添加了输入字段和按钮。还对输出 JSP 页添加了输出字段。

您学习了如何完成下列任务：

- 打开并修改 Faces JSP 页。
- 对 JSP 页添加输入字段、输出字段和按钮。

课程 8：显示错误消息

当操作将错误消息返回到输入 JavaServer Pages (JSP) 页时显示这些错误消息。

要使用 ActionErrors，必须添加 Struts HTML 标记以在输入 JSP 页上显示错误消息。还必须对操作映射添加输入字段。

要修改应用程序以将任何 Bean 验证错误返回到输入 JSP 页：

1. 在 DateData 类的 validate 方法中，验证输入年份是否小于 1582，此方法将返回带有“pre-gregorian”消息的错误。

```
public ActionErrors validate(ActionMapping mapping, HttpServletRequest request) {

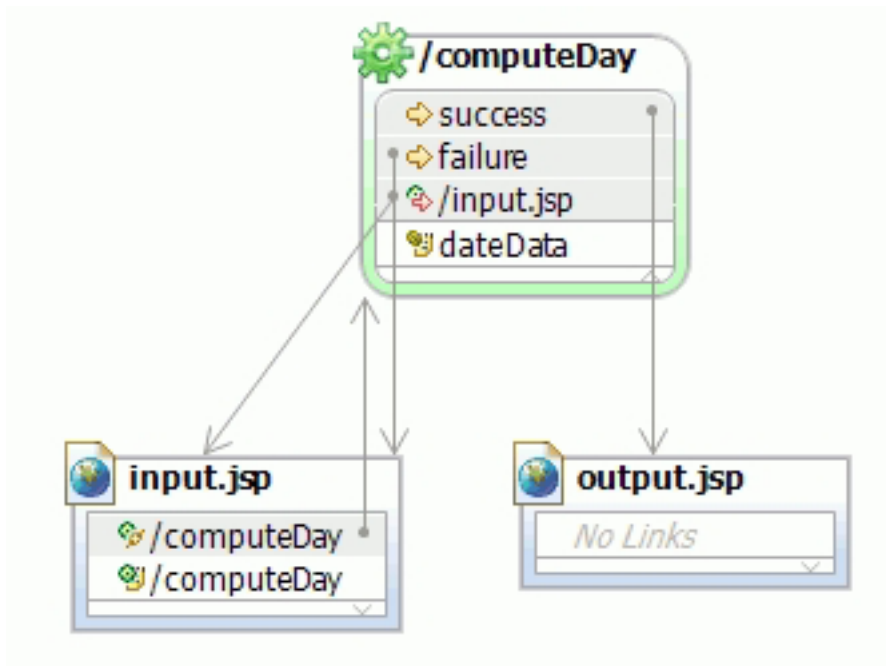
    ActionErrors errors = new ActionErrors();
    if (year < 1582)
    {
        errors.add("year", new org.apache.struts.action.ActionError("pre_gregorian"));
    }
}
```

2. 在 `ApplicationResources.properties` 文件中找到“pre-gregorian”错误消息，并验证该消息是否与以下定义相匹配：

pre-gregorian=Date is before 1582, the year the Gregorian calendar began

3. 要为该错误消息指定目标 JSP 页，为使用表单 bean 的操作选择输入：
 - a. 打开 Web 图。
 - b. 在图中，将鼠标指针置于 `/computeDay` 节点上，并将节点连接柄拖至 `input.jsp` 节点。
 - c. 在弹出菜单中，选择**操作输入**。

获得的 Web 图应类似于下图：



4. 更新输入 JSP 页以显示错误消息：
 - a. 双击 `input.jsp` 页以在 Page Designer 中打开。
 - b. 从选用板的“Struts HTML 标记”抽屉中将“错误”标记拖到该页中您想显示错误消息的位置。

输入 JSP 页现在应类似于下图：

- * First error message
- * Second error message
- * Third error message

Day:	
Month:	
Year:	
Submit	Reset

5. 保存并关闭输入 JSP 页和 Web 图。

课程要点

在本课程中，在输入 JSP 页中创建了错误消息的显示。

您学习了如何完成下列任务：

- 添加 HTML 标记以在输入 JSP 页中显示错误消息。
- 修改操作映射以包括输入字段。

课程 9：测试 Struts 应用程序

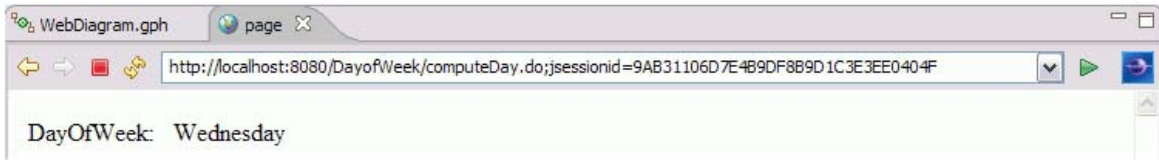
在 IBM® WebSphere 或 Apache Tomcat 运行时环境中测试 Struts 应用程序。在此环境中，您应当能够看到先前添加至 JSP 页的输入和输出控件。

要测试 Struts 应用程序：

1. 在“项目资源管理器”中，右键单击 input.jsp 文件并选择**运行方式** → **在服务器上运行**。
2. 在选择服务器窗口中，选择 **WebSphere Application Server 6.x** 或 **Apache Tomcat Server**，然后单击**完成**。
3. 如果出现服务器操作窗口，则单击**是**以启动该服务器。
4. 单击**完成**。输入 JSP 页将在 Web 浏览器视图中打开。



5. 在该 JSP 页中，在各个字段中输入以下数据：
 - a. 在“日”字段中输入 10。
 - b. 在“月份”字段中输入 12。
 - c. 在“年份”字段中输入 1962。
6. 单击“提交”按钮。输出 JSP 页显示在 Web 浏览器视图中。“星期”字段中的值应为星期三。



可以在输入 JSP 页的各个字段中输入其他日期以计算当天是星期几。

课程要点

祝贺您！您已经成功测试了 Struts 应用程序并在模拟 Web 浏览器中查看了输出。

您学习了如何完成下列任务：

- 选择测试环境。
- 对输入 JSP 页的各个字段输入数据。
- 在输出 JSP 页中以字符串格式显示结果。

总结：使用 Struts 创建 Web 应用程序

在本教程中，您学习了如何创建简单的 Struts 应用程序。

已学习的课程

完成这些课程之后，您应该能够完成下列任务：

- 创建 Struts 项目
- 编辑 Web 图
- 创建并编辑表单 bean
- 创建两个 JavaServer Pages (JSP) 文件
- 创建操作和操作映射
- 将 ActionError 标记添加至输入 JSP 页
- 测试 Struts 应用程序

其他资源

如果您想要更多地了解本教程中讨论的主题，可参阅下列信息源：

- Apache Struts Web 站点 (<http://struts.apache.org>) 中包含有关 Struts 框架的信息。
- 联机帮助 (帮助 → 帮助内容) 包含有关 Struts 应用程序和 Web 图的文档。

相关信息



