Introduction to struts and tiles with a working example

Step by Step example using struts tiles. Example application shows how to list, create, edit and save data. Source code is provide.

As development environment we used eclipse with the plugin myeclipse.

Generals

Author: Sascha Wolski

http://www.laliluna.de/tutorial.html – Tutorials for Struts, EJB, xdoclet and eclipse.

Date: November, 8st 2004 Software: MyEclipse 3.8 Jboss 3.2.5 Source code: Source code for this tutorial can be downloaded here. The PDF version is located here.

Introduction

This Tutorial explain how to use struts tiles with a simple example.

What is struts

Consider a example library application whose web page layout has a header, body and footer. On this example i will explain what is tiles and when you will be use it.

Header
Body
Footer

The developer have two alternativ ways to create the layout. He add the header and footer to each page of the site or he use the command <jsp:include>, to include the header and footer into the pages. In the first way all pages contains the header and footer source code. When the header or footer will changed, the developer have to change all pages. In the second way the header and footer are placed in seperated files, so the files can be included into the pages where they will be needed.

When he change the layout and add a menu to the library application, shown by the picture below, the developer have to change all pages, because he have to add the inlude command of the menu in each page. On this situation tiles is the best way to develope the page layout, thus the developer don't have to change each page.

	Header				
Menu	Body				
Footer					

Tiles use a seperate layout file, that contains the container of the layout. When the layout will be changed only the layout file and the tiles configuration files have to change by the developer. That will save many time on large applications.

Your first tiles application

Create a new struts project with File > New > Project or use the shortcut Strg + n. Select the Wizard in J2EE Web Project.

A New	×
Select a wizard Create J2EE Web Project	
<u>W</u> izards:	
Class Extension Point Schema Interface Java Project Plug-in Project CVS Database Explorer J2EE EIB Project EIB Project EXEPTION Application Project Web Project EXEPTION From EXEPTION From EXERCT EXERCT Struts 1.0 EXERCT Struts 1.1	
< <u>B</u> ack. <u>N</u> ext > Einish	Cancel

	14 mart	-
Project Name	Libraryweb	
Ocation Directory	C:\Programme\eclipse\eclipse\workspace\LibraryWeb	Browse
Source folder	src	
Neb root folder	WebRoot	
Context root URL	/LibraryWeb	ĺ
J2EE Specification	n Level	
C J2EE 1.3 •	J2EE 1.4 [default]	
ISTL Support	an and a second second	
Add JSTL 1.0	libraries to WEB-INF/lib folder	

After creating the project, your Package Explorer looks like the picture below.



For now your project is a normal J2EE project, so we need to add the struts capabilityies. Right click on the project and add the capabilityies for struts with Add Struts Capabilityies.

e M	lyEclip	ose -	Eclip	se Plat	form														
File	Edit	Sour	ce l	Refactor	Na	wigate	Sea	rch	Project	MyB	clipse	Run	Wind	low	Help				
	- 6	1 🖻];	🗗 🕵	•]	参•	0	- 9	-	<u>0</u> +	₿ 6	•]	ر 😕	P	f ⁰	•]	*>	¢	• =>
Hier	archy	# P	acka	ge Explo	rer 🕽	3													
	-	_	_						Q [⊒ ∉									
	E	orary	\supset	lew						•									
		JRE	(Go Into															
	÷	J2E	(Open in I	New V	Window													
	ė <u>e</u>	> We	(Open Ty	pe Hi	erarchy		F4	F										
		····		Тору				Ct	rl+⊂										
	Ē	🤁	B F	Paste				Ct	rl+∀										
			×	Delete				De	elete										
			9	Source				Al	t+Shift+	-s 🕨									
			F	Refactor				Al	t+Shift+	-т ▶									
			کے I	mport															
			۵ کے	Export	•														
			ۍ ۴ کې	Refresh				FS	5										
			(Ilose Pro	oject														
			F	Run						•									
			0	Debug						•									
			٦	ſeam						•									
			(Compare	: With) 				•									
			H N	kestore i AvEclinsi	rrom =	Local Hi	story			•	I Add	and F	lomovi	- Pro	iect D	eplov	ment	s	1
									L. Cabo	-	0.dd	Shout	c Capa	bilitie					
			ŀ	ropertie	:5			н	t+Enter		Add	JSE C	apabili	itiec					
											Add	JSTL	1.0 Lib	IS					
											Add	Hiber	nate C	apab	oilities.				_
											Run	XDoc	let						
											Run	Valida	ation						

Change the properties Base package for new classes and Default application resource

ew 🖉				×
Struts Support for MyEclip	se Web Project			
Enable project for Struts dev	elopment			الو 💽
Web project:	LibraryWeb			
Web-root folder:	/WebRoot			
Servlet specification:	2.4			
Struts config path:	/WEB-INF/struts-conl	fig.xml		Browse
Struts specification:	C Struts 1.0	Struts 1.1		
ActionServlet name:	action			
URL pattern	● *.do	○ /do/*		
Base package for new classes	;; de.laliluna.tutorial.libi	rary struts		Browse
Default application resource:	de.laliluna.tutorial.libi	ary.struts.ApplicationResources		
	🔽 Install Struts jars	🔽 Install Struts TLDs		
			<u>E</u> inish	Cancel

Create a default page

Ok, now we want to create a default page. Right click (yes again) on the Folder MebRoot in the Project and choose New > JSP.



Set the name to index.jsp and choose on template to use > Standard JSP using Struts 1.1 MyEcplise will use the template to create the JSP File.

🧲 Create a new	JSP page.		×
JSP Wizard			<j></j>
File <u>P</u> ath: File <u>N</u> ame:	/LibraryWeb/WebRoot index.jsp		Browse
Template to use	2] Standard JSP using Struts 1.1		
	[Einish	Cancel

You will find the file index.jsp in the folder WebRoot of the project. On the top of the file you will find the struts tag libraries. These includes will be use to access the tags of struts. In your case we only need the logic tag library.



Insert the following line below the included logic tag.

<logic:forward name="welcome" />

This line will arranges struts to find a forward with the name welcome. If the application don't find this forward it will leads an error.

Add the tiles functionality

Open the file struts-config.xml in the folder WEB-INF and replace the following line

```
<controller bufferSize="4096" debug="0" />
with
<controller processorClass="org.apache.struts.tiles.TilesRequestProcessor"
bufferSize="4096" debug="0" />
```

Below the line

```
<message-resources
parameter="de.laliluna.tutorials.library.struts.ApplicationResources" />
add
<plug-in className="org.apache.struts.tiles.TilesPlugin">
    <set-property property="definitions-config" value="/WEB-INF/tiles-defs.xml" />
    <set-property property="moduleAware" value="true" />
    <set-property property="definitions-parser-validate" value="true" />
    </plug-in>
```

Struts will load the tiles - controller and the tiles - plugin

Create the layout file

Create a new jsp file in <code>WebRoot/jsp/</code> and set the name to <code>siteLayout.jsp</code> Add the following source code.

```
<%@ page language="java"%>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-html" prefix="html" %>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-tiles" prefix="tiles" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html:html locale="true">
 <head>
  <html:base />
  <title><tiles:getAsString name="title" /></title>
 </head>
 <body>
 <tiles:insert attribute="header" />
         <tiles:insert attribute="navigation" />
             <tiles:insert attribute="body" />
         \langle / \pm r \rangle
         <tiles:insert attribute="footer" />
    </body>
</html:html>
```

The black marked tags are the placeholder for title, header, navigation, body and footer. The first tag in the listing above is <tiles:getAsString ...>. This tag retrieves the title as string and insert it there. The following tags <tiles:insert ..> will be used to insert different jsp files, strings and action mappings into the siteLayout.jsp.

Create the tiles definitions

Create the file tiles-defs.xml in the folder WEB-INF, if it not exists.



Open the file and add the following source code.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE tiles-definitions PUBLIC
       "-//Apache Software Foundation//DTD Tiles Configuration 1.1//EN"
      "http://jakarta.apache.org/struts/dtds/tiles-config 1 1.dtd">
<tiles-definitions>
     <!-- Base Tiles Definition -->
     <definition name="base.definition" path="/jsp/siteLayout.jsp">
           <put name="header" value="/jsp/header.jsp" />
           <put name="navigation" value="/navigation.do" />
           <put name="footer" value="/jsp/footer.jsp" />
     </definition>
     <!-- Tiles Definition of welcome page -->
     <definition name="page.welcome" extends="base.definition">
           <put name="title" value="Welcome page" />
           <put name="body" value="/jsp/index.jsp" />
     </definition>
     <!-- Tiles Definition of BookList -->
     <definition name="page.booklist" extends="base.definition">
           <put name="title" value="Book list page" />
           <put name="body" value="/jsp/bookList.jsp" />
     </definition>
</tiles-definitions>
```

Within the tag block <tiles-definitions> you add the different tiles definitions. With the parameter path, of the base definition, you can set a jsp file, that contains the layout of the web site. Within the tag of the base definition, with put you set jsp files, strings and action mapping, so you can call them in the layout file. Each of the next definitions extends from the base.definition with the parameter extends="base.definition". They will however reuse the header, navigation and footer from the base.definition. Within the extends tiles definitions you only add the jsp files, strings and action mappings that are different from the base.definition.

Create the jsp files

Now create the needed jsp files in the folder WebRoot/jsp/. Right click on the project > New > JSP.

header.jsp

navigation.jsp

footer.jsp

index.jsp

bookList.jsp

We only need the struts tag libraries on the top of the file, because the container of the web site layout is placed in the siteLayout.jsp.

```
<%@ page language="java"%>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-bean" prefix="bean" %>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-html" prefix="html" %>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-logic" prefix="logic" %>
```

Open the file Datei header.jsp in the folder WebRoot/jsp/. Change the content of the file to.

```
<%@ page language="java"%>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-bean" prefix="bean" %>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-html" prefix="html" %>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-logic" prefix="logic" %>
```

That is the header page.

Open the file footer.jsp and change the content to.

```
<%@ page language="java"%>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-bean" prefix="bean" %>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-html" prefix="html" %>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-logic" prefix="logic" %>
```

That is the footer page.

Repeat it with the file index.jsp.

```
<%@ page language="java"%>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-bean" prefix="bean" %>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-html" prefix="html" %>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-logic" prefix="logic" %>
```

Welcome page.

and the file bookList.jsp

```
<%@ page language="java"%>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-bean" prefix="bean" %>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-html" prefix="html" %>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-logic" prefix="logic" %>
```

That is the book list page.

Open the last file navigation.jsp and add two links refers to the action mappings, we will create in the next step. action="welcome" call the action mapping with the name welcome.

```
<%@ page language="java"%>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-bean" prefix="bean" %>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-html" prefix="html" %>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-logic" prefix="logic" %>
Navigation page
<br />
<html:link action="welcome"> Goto welcome page </html:link>
<br />
<html:link action="bookList"> Goto book list page </html:link>
<br />
<br />
</br /
```

The action mapping

What is a action mapping?

The action mapping is the heart of struts. It managed all actions between the application and the user. You can define which action will be executed by creating a action mapping.

Open the file struts-config.xml. Right click on Action-Mapping > New Action in the outline window.

🗄 Outline 🛛 🔻 🗖 🗖	🗆 🗄 Outline 🗙 🗸 🖵 🗖
data-sources	Configuration data-sources
🛁 🧓 form-beans	form-beans
	global-exceptions
	global-forwards
	New Action
controller	New Form, Action and JSP
message-resources	Bssagenesources

Create three action mappings. The first two mappings refers to the tiles definitions page.welcome and page.booklist. The last one refers to the navigation.jsp that will display the navigation. We use this mapping in the tiles definition base.definition. (see Create the tiles definitions)

Set the Use case: welcome of the action mapping, that will display the welcome page. Choose Create new Action class, so MyEclipse will create a action class. On Forward add a new Forward to the tiles definition (page.welcome) that will display the welcome page with the header, navigation and footer.

New Action	1	×
Struts Action Create Struts	Declaration 1.1 Action	
Config/Module Use case: 🤇	: /LibraryWeb/WebRoot/WEB-INF/struts-config.xml welcome	Browse
Path: Action Type:	/welcome Type C Forward C Include	
Action Impl Superclass: Type:	Create new Action class Use existing Action class org.apache.struts.action.Action de.laliluna.tutorials.library.struts.action.WelcomeAction	
Optional Deta Form Par- Forward	ils ameter Methods Forwards Exceptions > New Forward	
	Forward name: showwelcome Brows	e
	Add Close	

Repeat this for the bookList Action Mapping and the Forward page.booklist

➡ New Action		×
Struts Action	Declaration	
Create Struts	1.1 Action	
Config/Module:	/LibraryWeb/WebRoot/WEB-INF/struts-config.xml	Browse
Use case	bookList	
Path:	/bookList	
Action Type:	Type C Forward C Include	
Action Impl	Create new Action class Use existing Action class org.apache.struts.action.Action	
Optional Detai	s meter Methods Forwards Exceptions	
Forwards	+ showBookList - [page.booklist]	Add Edit Remove

Create the action mapping of the navigation. On Forward we not refer to a tiles definition, we refer directly to the file navigation.jsp.

Config/Module:	/LibraryWeb/WebRoot/WEB-INF/struts-config.xml	Browse
Use case: 🤇	navigation	
Path:	Inavigation	
Action Type:	Type C Forward C Include	
Action Impl:	Create new Action class	
Superclass:	org.apache.struts.action.Action	
Туре:	de.laliluna.tutorials.library.struts.action.NavigationAction	
Optional Detail		
Form Para	neter Methods Forwards Exceptions	
Forwar(New Forward	×
Fo	rward name: showNavigation	
Fo	rward path: /jsp/navigation.jsp Browse	
	Redirect Context relative	
	Add Close	

The action forward

What is an action forward?

A action forward can be used to forward to a jsp or action mapping. There are two different action forwards. The global action forward and the local action forward. You can access a global action forward on each jsp or action class. A local action forward can only be accessed by the assigned action class.

Remain in the struts-config.xml and the outline window. Create a new forward on global forward with right click.



Choose the Forward Scope "Global Forward" and use the name, we have set in the default page. The Forward path refers to the action mapping, that will diplay the welcome page.

➡ New Forward	d	×
Struts 1.1 Forw	vard Declaration	
Create Struts 1	.1 Forward	
Config/Module:	/LibraryWeb/WebRoot/WEB-INF/struts-config.xml	Browse
Forward Scope	Global Forward C Local Action Forward	
Action Path:		Browse,.,
Forward pame:	welcome	
Forward path:	/welcome.do	Browse
	Redirect Context relative	
	Einish	Cancel

You find the following in your struts-config.xml

```
<global-forwards >
  <forward name="welcome"
          path="/welcome.do"
           redirect="true" />
</global-forwards>
<action-mappings >
 <action path="/welcome"</pre>
          type="de.laliluna.tutorials.library.struts.action.WelcomeAction">
          <forward name="showWelcome" path="page.welcome" />
 </action>
 <action path="/bookList"</pre>
          type="de.laliluna.tutorials.library.struts.action.BookListAction">
          <forward name="showBookList" path="page.booklist" />
 </action>
 <action path="/navigation"</pre>
          type="de.laliluna.tutorials.library.struts.action.NavigationAction">
          <forward name="showNavigation" path="/jsp/navigation.jsp" />
 </action>
</action-mappings>
```

Edit the action classes

Open the action class NavigationAction.java in the package de.laliluna.tutorial.library.action. Change the method execute.

```
/**
 * Method execute
 * @param mapping
 * @param form
 * @param request
 * @param response
 * @return ActionForward
 */
public ActionForward execute(
 ActionMapping mapping,
 ActionForm form,
 HttpServletRequest request,
 HttpServletResponse response) {
 return mapping.findForward("showNavigation");
}
```

In the next step open the class WelcomeAction.java and change the method execute.

```
/**
 * Method execute
 * @param mapping
 * @param form
 * @param request
 * @param response
 * @return ActionForward
 */
public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response) {
        return mapping.findForward("showWelcome");
}
```

Repeat this on the last class BookListAction.java.

```
/**
 * Method execute
 * @param mapping
 * @param form
 * @param request
 * @param response
 * @return ActionForward
 */
public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response) {
        return mapping.findForward("showBookList");
}
```

The command return mapping.findForward (<forward>); search for a forward with namen within the clasp and refres to it. We dont add any other functionality to the class, In den Action Klassen werden wir keine weitere Funktionalität hinzufügen, that should not be a component of this tutorials.

Congratulation, thats all.

Test your application

Start the jboss and deploy the project as Package Archive

(
New Deploymen	t o
Create new proje	ect deployment for LibraryWeb
IIIah Dusisahi	1 (kuram 41) - k
web Project:	Libraryweb
Server:	JBoss 3
Deploy type:	C Exploded Archive C Packaged Archive
Deploy Location:	rogramme\jboss-3.2.5\server\default\deploy\LibraryWeb.war

Call the project in your browser http://localhost:8080/LibraryWeb/

That is the header page.		
Navigation page Goto welcome page Goto book list page	Welcome page.	