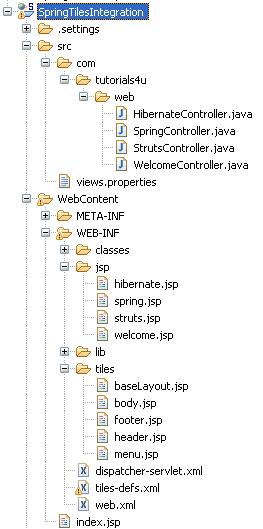
**Spring Tiles Integration**

In this example you will learn how to integrate Spring with Tiles 2. The directory structure of the example is shown below.



Add the following library files to the lib directory.

01.antlr-runtime-3.0

02.commons-logging-1.0.4

03.org.springframework.asm-3.0.0.M3

04.org.springframework.beans-3.0.0.M3

05.org.springframework.context-3.0.0.M3

06.org.springframework.context.support-3.0.0.M3

07.org.springframework.core-3.0.0.M3

08.org.springframework.expression-3.0.0.M3

09.org.springframework.web-3.0.0.M3

10.org.springframework.web.servlet-3.0.0.M3

11.

12.commons-beanutils-1.7.0

13.commons-digester-1.8

14.commons-logging-api-1.1

15.jstl

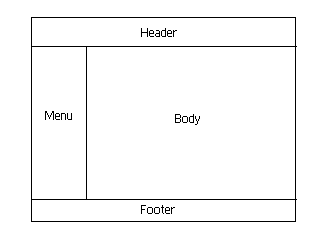
16.standard

17.tiles-api-2.0.4

18.tiles-core-2.0.4

19.tiles-jsp-2.0.4

You will see how to create a simple classic tile layout with a header, menu, body and footer regions.



In Spring to use Tiles, configure the following Tile definition in the Spring configuration file.

01.<?xml version="1.0" encoding="UTF-8"?>

02.<beans xmlns="http://www.springframework.org/schema/beans"

03.xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

04.xmlns:p="http://www.springframework.org/schema/p"

05.xmlns:context="http://www.springframework.org/schema/context"

06.xsi:schemaLocation="

07.http://www.springframework.org/schema/beans

08.http://www.springframework.org/schema/beans/spring-beans.xsd

09.http://www.springframework.org/schema/context

10.http://www.springframework.org/schema/context/spring-context.xsd">

11.

12.<bean id="viewResolver" class="org.springframework.web.servlet.view. ResourceBundleViewResolver" p:basename="views" />

13.

14.<context:component-scan base-package="com.tutorials4u.web" />

15.

16.

17.<bean id="tilesConfigurer" class="org.springframework.web.servlet.view.tiles2. TilesConfigurer" p:definitions="/WEB-INF/tiles-defs.xml" />

18.

19.</beans>

Using the definitions attribute specify the location of the tiles definition file. Here the location is "/WEB-INF/tiles-defs.xml". The tiles definition file is shown below.

01.<?xml version="1.0" encoding="UTF-8" ?>

02.

03.<!DOCTYPE tiles-definitions PUBLIC

04."-//Apache Software Foundation//DTD Tiles Configuration 2.0//EN"

05."http://tiles.apache.org/dtds/tiles-config\_2\_0.dtd">

06.

07.<tiles-definitions>

08.

09.<definition name="baseLayout" template="/WEB-INF/tiles/baseLayout.jsp">

10.<put-attribute name="title"  value="Template"/>

11.<put-attribute name="header" value="/WEB-INF/tiles/header.jsp"/>

12.<put-attribute name="menu"   value="/WEB-INF/tiles/menu.jsp"/>

13.<put-attribute name="body"   value="/WEB-INF/tiles/body.jsp"/>

14.<put-attribute name="footer"   value="/WEB-INF/tiles/footer.jsp"/>

15.</definition>

16.

17.<definition name="welcome" extends="baseLayout">

18.<put-attribute name="title"  value="Welcome"/>

19.<put-attribute name="body"   value="/WEB-INF/jsp/welcome.jsp"/>

20.</definition>

21.

22.<definition name="struts" extends="baseLayout">

23.<put-attribute name="title"  value="Struts"/>

24.<put-attribute name="body"   value="/WEB-INF/jsp/struts.jsp"/>

25.</definition>

26.

27.<definition name="hibernate" extends="baseLayout">

28.<put-attribute name="title"  value="Hibernate"/>

29.<put-attribute name="body"   value="/WEB-INF/jsp/hibernate.jsp"/>

30.</definition>

31.

32.<definition name="spring" extends="baseLayout">

33.<put-attribute name="title"  value="Spring"/>

34.<put-attribute name="body"   value="/WEB-INF/jsp/spring.jsp"/>

35.</definition>

36.</tiles-definitions>

Here we first define a base layout, later we extend the base layout and create three more tile definitions by changing only the title and the body regions.

To dispaly views we use the ResourceBundleViewResolver. By default the views.properties file will be used to store the key value pairs, we specify this using the basename attribute.

01.welcome.(class)=org.springframework.web.servlet.view.tiles2.TilesView

02.welcome.url=welcome

03.

04.struts.(class)=org.springframework.web.servlet.view.tiles2.TilesView

05.struts.url=struts

06.

07.hibernate.(class)=org.springframework.web.servlet.view.tiles2.TilesView

08.hibernate.url=hibernate

09.

10.spring.(class)=org.springframework.web.servlet.view.JstlView

11.spring.url=/WEB-INF/jsp/spring.jsp

The welcome, friends and office refers to the tile definition names (the one to right side of the = sign). For tile views we use "org.springframework.web.servlet.view.tiles2. TilesView" class. You can also have other views together with the tile views. The about url is mapped to the about.jsp page with is a org.springframework.web.servlet.view.JstlView.

The baseLayout.jsp file contains the table structure to hold the different regions.

01.<%@ taglib uri="http://tiles.apache.org/tags-tiles" prefix="tiles" %>

02.<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"

03."http://www.w3.org/TR/html4/loose.dtd">

04.

05.<html>

06.<head>

07.<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

08.

09.<title><tiles:insertAttribute name="title" ignore="true" /></title>

10.</head>

11.<body>

12.<table border="1" cellpadding="2" cellspacing="2" align="center">

13.<tr>

14.

15.<td height="30" colspan="2">

16.<tiles:insertAttribute name="header" />

17.</td>

18.</tr>

19.<tr>

20.<td height="250">

21.

22.<tiles:insertAttribute name="menu" />

23.</td>

24.<td width="350">

25.<tiles:insertAttribute name="body" />

26.</td>

27.</tr>

28.

29.<tr>

30.<td height="30" colspan="2">

31.<tiles:insertAttribute name="footer" />

32.</td>

33.</tr>

34.</table>

35.

36.</body>

37.</html>

Here we use annotated controller handler mapping to handle the request. In the redirect.jsp page we forward the request to the welcome.htm url.

1.<% response.sendRedirect("welcome.htm"); %>

The welcome.htm url will be handled by the WelcomeController class, where we forward it to the welcome tile page.

01.package com.tutorials4u.web;

02.

03.import org.springframework.stereotype.Controller;

04.import org.springframework.web.bind.annotation.RequestMapping;

05.

06.@Controller

07.public class WelcomeController {

08.

09.@RequestMapping("/welcome.htm")

10.public String redirect()

11.{

12.return "welcome";

13.}

14.}

