Struts 1.1
Tiles Framework

Sang Shin

sang.shin@sun.com
www.javapassion.com
Java™ Technology Evangelist
Sun Microsystems, Inc.

Disclaimer & Acknowledgments

- Even though Sang Shin is a full-time employee of Sun Microsystems, the content in this presentation is created as his own personal endeavor and thus does not reflect any official stance of Sun Microsystems.
- Sun Microsystems is not responsible for any inaccuracies in the contents.
- Acknowledgments:
  - The sample codes are from Struts 1.1 package
  - Some slides are created from JavaDoc of Tiles package

Revision History

- 12/01/2003: version 1: created by Sang Shin
- 04/09/2004: version 2: speaker notes are polished a bit
- 05/04/2004: version 3: Tiles slides are separated out from advanced struts
- 07/10/2005: version 4: More slides are added
- Things to do
  - speaker notes need to be added to the later part of the presentation
  - more example codes need to be added

Topics

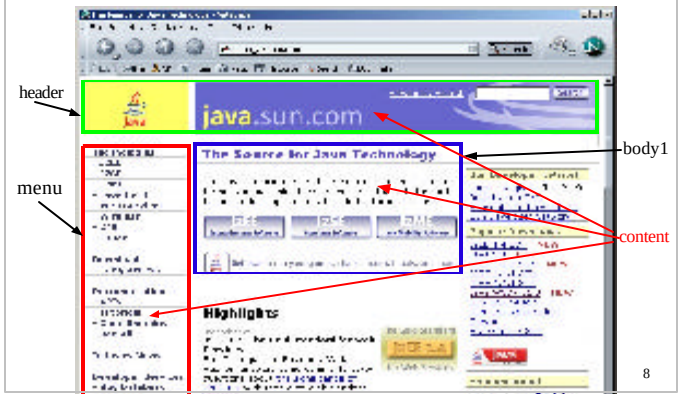
- Evolution of Web page layout technologies
  - 4 different generations
- Tiles framework
  - Layout (template)
  - Tiles tag library
  - Screen definitions (definitions)
  - Internationalization
  - Multi-channels
  - Configuration

Evolution of Web Page Layout Technology (Before we talk about Tiles Framework)

Presentation Needs of Web Application

- Common look and feel among all pages
  - Common layout with common header, footer, menus, forms, copyright, promotions, etc.
- Easy maintenance
  - You don't want hard-code layout in each page
  - Single place to change layout is desirable
- Separation of layout from content
  - Layout and contents should be able to change without affecting each other

Layout versus Content



Layout versus Contents

- Layout
  - How a page is logically structured
    - example: header is located on the top of the page
  - Deals with consistent Look and feel
  - Most pages in a single application share the same layout
    - header, footer, menu-bar, etc.
- Contents
  - What gets displayed
    - example: Login page has login content while logout page has logout content while using the same layout

Evolution of Web Page Layout Technology (1)

- 1st generation: JSP pages with embedded HTML layout tags
  - No separation of layout from contents
- 2nd generation: JSP pages with JSP include directive's (static) or JSP include action's (dynamic)
  - Some separation of layout from contents
  - JSP pages are reusable
  - Layouts are not reusable, however, because every JSP page contains layout information – if you want to change layout, you have to change every JSP page

Evolution of Web Page Layout Technology (2)

- 3rd generation: JSP templates (one defined by David Geary)
  - Separation of layout from contents
  - Both JSP pages and Layouts are reusable
- 4th generation: Tiles
  - Separation of layout from contents
  - Both JSP pages and Layouts are reusable
  - Superset of JSP templates with more features
  - Extends concept of JSP templates with "parameterized components" or "Tiles"

The 2nd Generation: Display Page with JSP Include Directives

Example: Every display page has hard-coded layout

```
<html><head><title>Templates</title></head>
<body background='graphics/blueAndWhiteBackground.gif'>

<table width='610'>
<tr valign='top'><td><jsp:include page='sidebar.jsp'/></td>
<td><table>
<tr><td><jsp:include page='header.html'/></td></tr>
<tr><td><jsp:include page='chapter.jsp'/></td></tr>
<tr><td><jsp:include page='footer.jsp'/></td></tr>
</table>
</td>
</tr>
</table>
</body></html>
```


Issues with the 2nd Gen.: JSP pages with JSP include directives

- The layout is still mixed with contents
  - Each display page explicitly specifies where header.jsp goes and where footer.jsp goes (hard-coded layout)
- Every display page has to have the same statements in the same order
  - If layout scheme needs to be changed, every display page has to be changed
- Reason for 3rd generation approach: Template

The 3rd Generation: Template

What is a Template?

- Template is a JSP page that uses JSP custom tag library to describe the layout of a page without specifying contents
- The template acts as a definition for what the pages of an application will look like, without specifying the content
- Content is inserted into the template page during runtime
- Several display pages use the same template

Why Template?

- Separation of Layout from content
  - Content and layout can change without interfering each other
- A common template is shared by many display pages
  - A single place to change when layout change is required
- Template provides consistent look and feel without having to hard-code it in every page

Example Display Page Using a Template

```
<%@ taglib URI='/WEB-INF/struts-template.tld' prefix='template' %>

<template:insert template='/defaultTemplate.jsp'>
  <template:put name='title' content='Java Passion' direct='true'/>
  <template:put name='header' content='/header.html'/>
  <template:put name='sidebar' content='/sidebar.jsp'/>
  <template:put name='content' content='/introduction.html'/>
  <template:put name='footer' content='/footer.html'/>
</template:insert>
```


Tiles Framework

What is Tiles Framework?

- Tiles framework allows building pages by assembling reusable Tiles
- A display page can be built by assembling a header, a footer, a menu and a body Tiles

What is a Tile?

- Each Tile (header, menu, body, ...) is a JSP page and can itself be built by assembling other Tiles
- Using Tiles can be compared as using Java methods: You need to define the Tiles (the method body), and then you can "call" this body anywhere you want, passing it some parameters. In Tiles, parameters are called "attributes" in order to avoid confusion with the request parameters

Inserting a Tile

- Inserting the body, or calling it, is done with the tag <tiles:insert ...> anywhere in a JSP page
- Insertion can also be done by specifying a definition name as the path of a Struts forward or as input, forward or include attributes of a Struts action

Example: Inserting a JSP Page

- This example inserts the specified page in place of the tag

```
<tiles:insert page="/layouts/commonLayout.jsp" flush="true" />
```

- The page attribute is any valid URL pointing to a resource inside the current site

Example: Inserting a Tile passing some attributes

- This example inserts the specified page, passing it the attributes.

```
<tiles:insert page="/layouts/classicLayout.jsp" flush=&quot;true">
  <tiles:put name="title"  value="Page Title" />
  <tiles:put name="header" value="/common/header.jsp" />
  <tiles:put name="footer" value="/common/footer.jsp" />
  <tiles:put name="menu"   value="/common/menu.jsp" />
  <tiles:put name="body"   value="/tiles/mainBody.jsp" />
</tiles:insert>
```

- Attributes are stored in a Tiles context which is passed to the inserted page and can then be accessed by their names

Example: Inserting a Tile by an attribute

- This inserts the Tiles referenced by the attribute "menu" value.

```
<tiles:insert attribute='menu' />
```

- The specified attribute value is first retrieved from current Tile's context, and then the value is used as a page target to insert.

Tiles Framework

- Superset of Template
  - Tiles framework supports template functionality
    - Tiles framework use a term “Layout” for a template
  - Support parameter passing
- Extra features over Template
  - Screen definitions
  - Dynamic page building
  - Reuse of Tiles
  - Internationalization
  - Multi-channels

Dynamic Page Building

- Tiles are gathered dynamically during page reload – Dynamic page building
- It is possible to change any attributes: layout, list of Tiles in portal, list of menu items, ...

Reuse of Tiles

- If well defined, a Tile can be reused in different locations
- Dynamic attributes are used to parameterize Tiles
- It is possible to define libraries of reusable Tiles
- Build a page by assembling predefined components, give them appropriate parameters

Tiles Framework: Layout (Template)

Layout

- Serves same purpose as Template
- Define common layouts and reuse them across many different projects
  - Define menu layouts and pass lists of items and links
  - Define a portal layout, use it by passing list of Tiles (pages) to show
- Reuse or customize existing layouts, or define your own ones
- Layout itself is also considered as a tile in Tiles framework
  - Called as “Layout tiles” (as opposed to “Non-layout tiles”

Pre-built Layouts from Tiles Framework

- Classic layout
  - header, left menu, body, footer
- Menu layout
  - menu with links
- Vertical box layout
  - a list of tiles in a vertical column
- Columns layout
- Center layout
- Tabs layout

Classic Layout (classicLayout.jsp)

```
<%@ taglib uri="/WEB-INF/struts-tiles.tld" prefix="tiles" %>

<%-- Layout Tiles
 This layout create a html page with <header> and <body> tags. It render
  a header, left menu, body and footer tile.
 @param title String use in page title
 @param header Header tile (jsp url or definition name)
 @param menu Menu
 @param body Body
 @param footer Footer
--%>
<HTML>
  <HEAD>
<%-- <link rel=stylesheet href="<%=request.getContextPath()%
     >/layouts/stylesheet.css" type="text/css"> --%>
    <title><tiles:getAsString name="title"/></title>

  </HEAD>
```

Classic Layout (classicLayout.jsp)

```
<body bgcolor="#ffffff" text="#000000" link="#023264" alink="#023264"
    vlink="#023264">
<table border="0" width="100%" cellspacing="5">
<tr>
  <td colspan="2"><tiles:insert attribute="header" /></td>
</tr>
<tr>
  <td width="140" valign="top">
    <tiles:insert attribute='menu'/>
  </td>
  <td valign="top"  align="left">
    <tiles:insert attribute='body' />
  </td>
</tr>
<tr>
  <td colspan="2">
    <tiles:insert attribute="footer" />
  </td>
</tr>
</table>
</body>
</html>
```

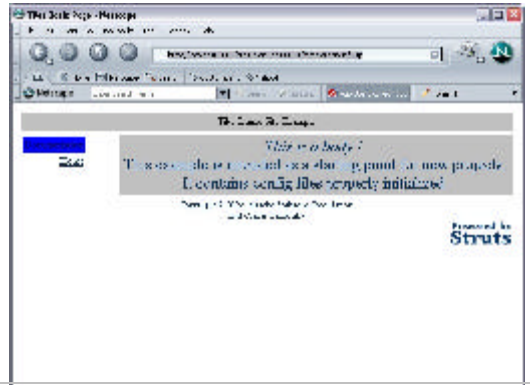

Passing Parameters to Layout

- Note that classicLayout.jsp does not know anything about content
  - this is the reason why this layout can be reused
- So content has to be supplied or passed as parameters to the layout page at runtime from your display page
  - let's see an example of templateNoDef.jsp in the following page

templateNoDef.jsp

```
<%@ taglib uri="/WEB-INF/struts-tiles.tld" prefix="tiles" %>

<%-- Insert a layout rendering requested tiles. --%>
<tiles:insert page="/layouts/classicLayout.jsp"  flush="true">
  <tiles:put name="title"  value="Tiles Basic Page" />
  <tiles:put name="header" value="/tiles/common/header.jsp" />
  <tiles:put name="footer" value="/tiles/common/footer.jsp" />
  <tiles:put name="menu"    value="/tiles/simpleMenu.jsp" />
  <tiles:put name="body"   value="/tiles/body.jsp" />
</tiles:insert>
```

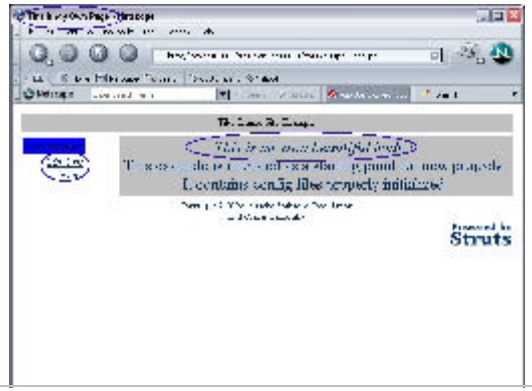

templateNoDef.jsp



myOwnDisplayPage.jsp

```
<%@ taglib uri="/WEB-INF/struts-tiles.tld" prefix="tiles" %>

<%-- Insert a layout rendering requested tiles. --%>
<tiles:insert page="/layouts/classicLayout.jsp"  flush="true">
  <tiles:put name="title"  value="This Is My Own Page" />
  <tiles:put name="header" value="/tiles/common/header.jsp" />
  <tiles:put name="footer" value="/tiles/common/footer.jsp" />
  <tiles:put name="menu"   value="/tiles/myOwnSimpleMenu.jsp" />
  <tiles:put name="body"   value="/tiles/myOwnBody.jsp" />
</tiles:insert>
```


myOwnDisplayPage.jsp



Tiles Framework:
Tiles Tag Library

Tags in Tiles Tag Library

- insert
- put
- definition
- getAsString
- get
- add
- importAttribute
- useAttribute
- putList
- initComponentDefinitions

<tiles:insert> tag

- In a Layout tile
  - Prescribes where the content will go using attribute attribute
  - example: <tiles:insert attribute='menu'/>
- In a non-layout tile
  - Retrieves a layout using page attriute
  - example: <tiles:insert page="/layouts/classicLayout.jsp" flush="true">

classicLayout.jsp as Layout tile

```
<body bgcolor="#ffffff" text="#000000" link="#023264" alink="#023264"
    vlink="#023264">
<table border="0" width="100%" cellspacing="5">
<tr>
  <td colspan="2"><tiles:insert attribute="header" /></td>
</tr>
<tr>
  <td width="140" valign="top">
    <tiles:insert attribute='menu'/>
  </td>
  <td valign="top"  align="left">
    <tiles:insert attribute='body' />
  </td>
</tr>
<tr>
  <td colspan="2">
    <tiles:insert attribute="footer" />
  </td>
</tr>
</table>
</body>
</html>
```


templateNoDef.jsp as non-Layout tile

```
<%@ taglib uri="/WEB-INF/struts-tiles.tld" prefix="tiles" %>

<%-- Insert a layout rendering requested tiles. --%>
<tiles:insert page="/layouts/classicLayout.jsp"  flush="true">
  <tiles:put name="title"  value="Tiles Basic Page" />
  <tiles:put name="header" value="/tiles/common/header.jsp" />
  <tiles:put name="footer" value="/tiles/common/footer.jsp" />
  <tiles:put name="menu"   value="/tiles/simpleMenu.jsp" />
  <tiles:put name="body"   value="/tiles/body.jsp" />
</tiles:insert>
```


<tiles:put> tag

- Define an attribute to pass to tile/component/template
- Can only be used inside 'insert' or 'definition' tag
- Value (or content) is specified using attribute 'value' (or 'content'), or using the tag body

Tiles Framework:
Screen Definitions

What is a Screen Definition?

- Create a screen by assembling Tiles : header, footer, menu, body
- Definitions can take place :
  - in a centralized xml file
  - directly in jsp page
  - in struts action
- Definitions provide an inheritance mechanism : a definition can extends another one, and override parameters.

Why Definitions?

- Without definitions (in scheme shown previously), in each of non-layout tiles, there is redundant code that specifies commonly used contents
  - In the following two slides, there are two non-layout tiles (templateNoDef.jsp and templateNoDef2.jsp)
  - There are redundant code (header, footer)
  - With large number of tiles, replacing, for example, a header with a different one could be a chore

templateNoDef.jsp as non-Layout tile #1

```
<%@ taglib uri="/WEB-INF/struts-tiles.tld" prefix="tiles" %>

<%-- Insert a layout rendering requested tiles. --%>
<tiles:insert page="/layouts/classicLayout.jsp" flush="true">
  <tiles:put name="title"  value="Tiles Basic Page" />
  <tiles:put name="header" value="/tiles/common/header.jsp" />
  <tiles:put name="footer" value="/tiles/common/footer.jsp" />
  <tiles:put name="menu"   value="/tiles/simpleMenu.jsp" />
  <tiles:put name="body"   value="/tiles/body.jsp" />
</tiles:insert>
```

templateNoDef2.jsp as non-Layout tile #2

```
<%@ taglib uri="/WEB-INF/struts-tiles.tld" prefix="tiles" %>

<%-- Insert a layout rendering requested tiles. --%>
<tiles:insert page="/layouts/classicLayout.jsp" flush="true">
  <tiles:put name="title"  value="Tiles Basic Page 2" />
  <tiles:put name="header" value="/tiles/common/header.jsp" />
  <tiles:put name="footer" value="/tiles/common/footer.jsp" />
  <tiles:put name="menu"   value="/tiles/simpleMenu2.jsp" />
  <tiles:put name="body"   value="/tiles/body2.jsp" />
</tiles:insert>
```


Why Definitions?

- Definitions allow you to specify common contents that are used by non-layout tiles
  - Allows you specify only page-specific contents in your non-layout tiles
- Provides centralized location for page descriptions
- Supports inheritance and overloading mechanism
  - A definition can extend another one and override some (or all) of its parameters

How Definitions are Used

- Insertion of a Tiles body can be associated to a logical name in what Tiles calls a "definition"
- A definition contains a logical name, a page used as body and some attribute values.
- The definition declaration doesn't insert the associated Tiles body. It just associates it with the name.
- A definition name can be used anywhere insertion of a Tiles body can occur.
- The associated Tiles body is then inserted with associated attributes.

How Definitions are Used

- The definition declarations can be done in JSP pages or in one or more centralized files
- A definition can extend another one, overload some attributes, add new attributes
  - This allows the declaration of a "master" definition declaring the common layout, header, menu and footer
  - All other definitions extend this master layout thereby making it possible to change the entire site look & feel simply by changing the master definition.

Tiles Framework:
Tiles Definitions File

Where to Create Definitions?

- In a JSP page or an XML configuration file
- If you create an XML configuration file
  - Tiles Definition File
  - Usually as /WEB-INF/tiles-def.xml
    - specified in Plugin configuration
  - Contains all the definitions for the entire application

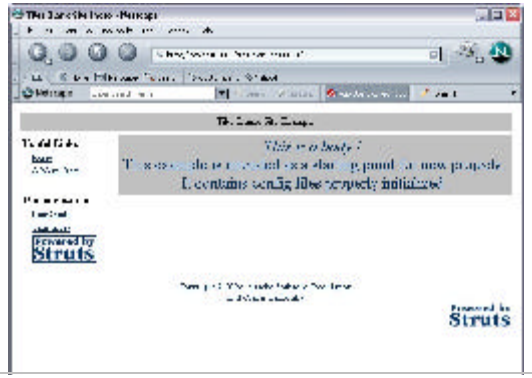
Tiles Definition File (WEB-INF/tiles-def.xml) From tiles-blank-struts1-1 Sample Code

```
<tiles-definitions>

  <!-- Main page layout used as a root for other pages defintion. -->
  <definition name="site.mainLayout" path="/layouts/classicLayout.jsp">
    <put name="title"  value="Tiles Blank Site" />
    <put name="header" value="/tiles/common/header.jsp" />
    <put name="menu"   value="site.menu.bar" />
    <put name="footer" value="/tiles/common/footer.jsp" />
    <put name="body"   value="/tiles/body.jsp" />
  </definition>

  <!-- This definition inherits from the main definition.
  It overload the page title, and the body used.
  Use the same mechanism to define new pages sharing common
  properties (here header, menu, footer, layout) -->
  <definition name="site.index.page" extends="site.mainLayout" >
    <put name="title"  value="Tiles Blank Site Index" />
    <put name="body"   value="/tiles/body.jsp" />
  </definition>
```


How Tiles Definition is Used: index.html of struts-tiles-blank Sample

```
<%@ page language="java" %>
<%@ taglib uri="/WEB-INF/struts-tiles.tld" prefix="tiles" %>

  <%-- Insert a definition described in tiles configuration file
    Change the definition name to insert another definition.
    It is possible to overload some definition attribute by adding
    some put tags with appropriate name.
  --%>
<tiles:insert definition="site.index.page" flush="true" />
```

How Tiles Definition is Used: index.html of struts-tiles-blank Sample code



Tiles Definition File (WEB-INF/tiles-def.xml) From tiles-blank-struts1-1 Sample Code

```
<!-- Menu bar definition
   This definition describe a "bar" of menu stacked verticaly.
   Each menu is describe elsewhere.
   Add new entry in the list to add new menu. -->

<definition name="site.menu.bar" path="/layouts/vboxLayout.jsp" >
  <putList name="list" >
    <add value="site.menu.links" />
    <add value="site.menu.documentation" />
  </putList>
</definition>
  ...
```

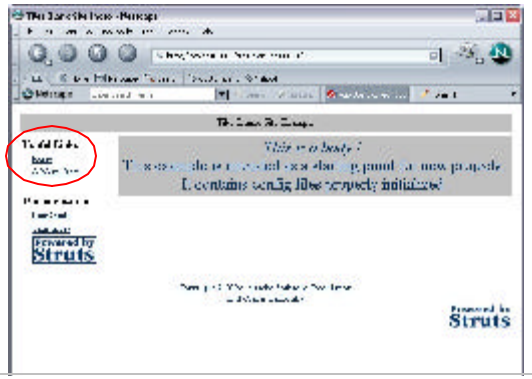

<tiles:putList> tag

- Declare a list that will be pass as attribute to tile
- List elements are added using the tag 'add'
- Can only be used inside 'insert' or 'definition' tag

Tiles Definition File (WEB-INF/tiles-def.xml) From tiles-blank-struts1-1 Sample Code

```
<!-- Menu description
   A menu has a title and a set of entries rendered as links.
   Add new entry to add new links in menu.-->

<definition name="site.menu.links" path="/layouts/menuNoStruts.jsp" >
  <put name="title" value="Useful Links" />
    <putList name="items" >
      <item value="Home"
            link="/index.jsp"
            classtype="org.apache.struts.tiles.beans.SimpleMenuItem" />
      <item value="A Menu Item"
            link="/templateNoDef.jsp"
            classtype="org.apache.struts.tiles.beans.SimpleMenuItem" />
    </putList>
</definition>
  ...
```

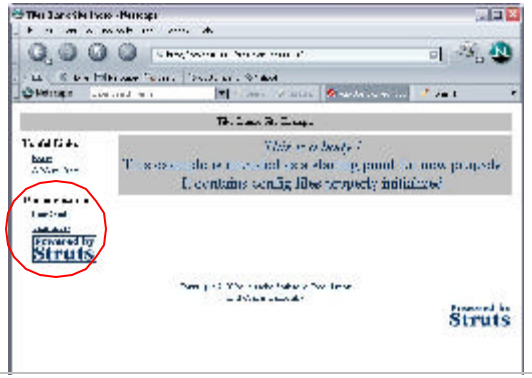
How Tiles Definition is Used: index.html of struts-tiles-blank Sample code



Tiles Definition File (WEB-INF/tiles-def.xml) From tiles-blank-struts1-1 Sample Code

```
<!-- Another Menu description
   A menu has a title and a set of entries rendered as links.
   Add new entry to add new links in menu.-->

<definition name="site.menu.documentation" path="/layouts/menuNoStruts.jsp" >
  <put name="title" value="Documentation" />
    <putList name="items" >
      <item value="User Guide"
              link="/index.jsp"
              classtype="org.apache.struts.tiles.beans.SimpleMenuItem" />
      <item value="Tags Index"
              link="/index.jsp"
              classtype="org.apache.struts.tiles.beans.SimpleMenuItem" />
      <item value="Struts Home"
              icon="/images/struts-power.gif"
              link="http://www.apache.org"
              classtype="org.apache.struts.tiles.beans.SimpleMenuItem" />
    </putList>
</definition>
```


How Tiles Definition is Used: index.html of struts-tiles-blank Sample code



Tiles Framework: Tiles & Struts

Using Tiles Framework with Struts

- Use the Tiles plug-in to enable Tiles definitions
- This plug-in creates the definition factory and passes it a configuration object populated with parameters
- Parameters can be specified in the web.xml file or as plug-in parameters
- The plug-in first reads parameters from web.xml, and then overloads them with the ones found in the plug-in
- All parameters are optional and can be omitted

Definition of a Tiles Plug-in in the struts-config.xml

```
<!-- ========== Tiles plug-in setting settings ==================== -->
 <!-- Here we specified the tiles plug-in.
      This plug-in register appropriate Request Processor -->
 <!--  <controller
    processorClass="org.apache.struts.tiles.TilesRequestProcessor"  />  -->

 <!-- === Associated Messages Ressource settings ============ -->
 <!-- Not used by tiles or this website, but needed due to a bug in actual Struts
    version -->
<message-resources parameter="org.apache.struts.webapp.tiles.dev1-
   1.ApplicationResources" null="false" />


<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" />
</plug-in>
```


How a Definition can be used as Forward in Struts

- Tile definition can be used as Struts forward (instead of URLs)
- In struts-config.xml file

```
<global-forwards>
  <forward name=”success”
           path=”site.mainLayout” />
</global-forwards>
```


Tiles Framework: Internationalization

Internationalization

- It is possible to load different Tiles according to the user's Locale
- A mechanism similar to Java properties files is used for definition files: you can have one definition file per Locale, the appropriate definition is loaded according to the current Locale
  - tiles-definitions-en.xml
  - tiles-definitions-de.xml

69

Tiles Framework: Multi-channels

70

Multi-Channels

- It is possible to load different Tiles according to a key stored e.g. in session context
- The key could hold e.g. user privileges, browser type, ...
- A mechanism similar to Java properties files is used for definition files: you can have one definition file per key, the appropriate definition is loaded according to the key

71

Tiles Framework: How to configure Tiles framework

72

Things to Configure

- Configure WEB-INF/web.xml file

```
<taglib>
    <taglib-uri>/WEB-INF/struts-tiles.tld</taglib-uri>
    <taglib-location>/WEB-INF/struts-tiles.tld</taglib-location>
</taglib>
```

- Configure Tiles Plugin in struts-config.xml file

```
<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>
```


Passion!