

Reference: <http://krams915.blogspot.in/2010/12/spring-mvc-3-tiles-2-integration.html>


What is Tiles:

Consider the following scenario,

Pets Squad Banner

Pet Type

A pet is an animal kept for companionship and enjoyment or a household animal, as opposed to wild animals or to livestock, laboratory animals, working animals or sport animals, which are kept for economic or productive reasons. The most popular pets are noted for their loyal or playful characteristics, for their attractive appearance, or for their song. Pets also generally seem to provide their owners with non-trivial health benefits.[1] keeping pets has been shown to help relieve stress to those who like having animals around. There is now a medically-approved class of "therapy animals," mostly dogs, that are brought to visit confined humans. Walking a dog can provide both the owner and the dog with exercise, fresh air, and social interaction.



Source: Wikipedia


Color theme from "Horses made of sticks" by piahr from Adobe Kuler

Pets Squad Banner

Pet Type

Canines

The dog (*Canis lupus familiaris*[1]) is a domesticated form of the grey/gray wolf, a member of the Canidae family of the order Carnivora. The term is used for both feral and pet varieties. The domestic dog has been the most widely kept working, hunting and companion animal in human history. The word "dog" may also mean the male of a canine species,[2] as opposed to the word "bitch" for the female of the species.[3]



Source: Wikipedia

Color theme from "Horses made of sticks" by piahr from Adobe Kuler

Pets Squad Banner

Pet Type

Felines

The cat (*Felis catus*), also known as the domestic cat or housecat[5] to distinguish it from other felines and felids, is a small furry domesticated carnivorous mammal that is valued by humans for its companionship and for its ability to hunt vermin and household pests. Cats have been associated with humans for at least 9,500 years,[6] and are currently the most popular pet in the world.[7] Owing to their close association with humans, cats are now found almost everywhere on Earth.



Source: Wikipedia

Color theme from "Horses made of sticks" by piahr from Adobe Kuler

In the above three pages of type pets (1st one), dogs(2nd one) and cats(3rd one).

This is the common scenario where we see in all the web sites where the most part of the web page structure remains the same and only the major content changes.

So in order to stop writing the same parts of the page again we use **Tiles framework**.

So the main structure will remain the same. Just like the way we do by using the abstract class in Java where only the abstract methods needs to be implemented and the concrete methods will be the same.

Requirements:

1. Spring
2. Maven
3. Eclipse

To setup Spring-Maven using eclipse please see the previous tutorial. (look **SampleWebApplication**)

Step – 1:

Add **tiles** dependencies in the pom file.

```
<!-- Tiles - dependencies - START -->
<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-core</artifactId>
  <version>2.2.2</version>
  <type>jar</type>
  <scope>compile</scope>
</dependency>

<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-template</artifactId>
  <version>2.2.2</version>
  <type>jar</type>
  <scope>compile</scope>
</dependency>

<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-jsp</artifactId>
  <version>2.2.2</version>
  <type>jar</type>
  <scope>compile</scope>
</dependency>

<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-servlet</artifactId>
  <version>2.2.2</version>
  <type>jar</type>
  <scope>compile</scope>
</dependency>
<!-- Tiles - dependencies - END -->
```

We will also need **slf4j** dependencies. So add them too in the pom.

```
<!-- slf4j - dependency - START -->
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-api</artifactId>
  <version>1.5.6</version>
</dependency>

<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-log4j12</artifactId>
  <version>1.5.6</version>
</dependency>
<!-- slf4j - dependency - END -->
```

Step – 2:

Declare a template definition to manage the pages. So create **tiles-definition.xml** under **WEB-INF**. Please refer **tiles-definition.xml** in TilesWebApplication project.

Explanation:

As per the above images we have three screens.

First page is divided into 4 parts and **Second and Thrid page** are divided into 5 parts each.

So we need two definitions for the two types of pages.

See the tiles-definition.xml file we have template-main definition which we will use for First page and template-detail for second and third page definitions.

Template-main: (Below page is divided into 4 parts.)

```
<definition name="template-main" template="/WEB-INF/pages/main.jsp">
  <put-attribute name="banner-content" value="" />
  <put-attribute name="title-content" value="Pet Type" /> (This part is
constant.)
  <put-attribute name="primary-content" value="" />
  <put-attribute name="footer-content" value="" />
</definition>
```

Template-detail: (Below the page is divided into 5 parts. Below banner.jsp and footer.jsp does not exist)

```
<definition name="template-detail" template="/WEB-INF/jsp/layouts/detail.jsp">
  <put-attribute name="banner-content" value="/WEB-INF/pages/banner.jsp" />
  <put-attribute name="title-content" value="Pet Type" /> (This part of the page is
constant.)
  <put-attribute name="subtitle-content" value="" />
  <put-attribute name="primary-content" value="" />
  <put-attribute name="footer-content" value="/WEB-INF/pages/footer.jsp" />
</definition>
```

Now its like a inheritance in Java. What all are specific to the sub-class will be implemented by the sub-class itself and if any thing same then the sub-class will be using the super-class information.

Consider for Dogs (which is having 5 parts layout structure.)

```
<definition name="dog-tiles" extends="template-detail">
  <put-attribute name="banner-content" value="Dog-Example" />
  <put-attribute name="subtitle-content" value="Canines" />
  <put-attribute name="primary-content" value="/WEB-INF/pages/dogs.jsp" />
</definition>
```

Here we are implementing on the one that is specific to Dogs. The rest of the page will be same as the detail.jsp page specification. Look at detail.jsp page for more details.

In **Dogs.jsp** page previously we implemented everything now we will only implement the required ones. Please refer the Dogs.jsp page and in this the commented section is the one which is common and hence need no to implement again.

The same we will be doing for **Cats.jsp**

Similarly for Pets (which is having a 4 parts layout structure) we will be using **template-main**

definition which is following **main.jsp**

Step – 3:

In the dispatcher **mvc-dispatcher-servlet.xml** include the tiles resource.

```
<import resource="tiles-context.xml"/>
```

So we are importing a resource called **tiles-context.xml**

Step – 4:

Create a file called **tiles-context.xml** under WEB-INF which will point to the **tiles-definition.xml**. Please refer tiles-context.xml file for more details.

Step – 5:

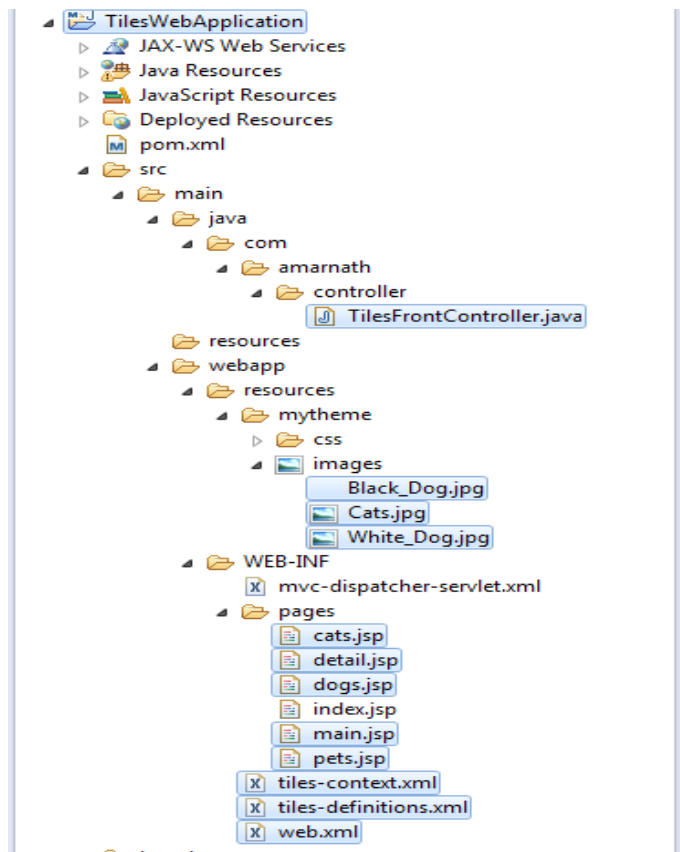
Finally go to the controller class and then you can see the method return was **“index”** but now we are returning **pets-tiles or dogs-tiles or cats-tiles**.

So when **pets-tiles or dogs-tiles or cats-tiles** is returned Spring first checks in the **tiles-definition.xml** and if present then return the corresponding tiles-definition.

Note: All the XML files are under the WEB-INF folder.

Also see the page linking path .. I mean the relative path of the jsp pages .. it is WEB-INF/pages/*.jsp

Finally **Folder Structure** looks like below,



Note – 1: (Error Details)

While I am running the TilesWebApplication I got an error in TOMCAT 6 logger. I think its worth it is worth mentioning here about the error,

SEVERE: Error listenerStart tomcat

The error is due to the logging collision between default **java.util.logger** and the **log4j**. You need to follow the following TOMCAT 6 documentation inorder to resolve this issue.

Link: [TOMCAT6 LOG4J ISSUE RESOLVER](#)

Please do every step here and also please not, \$CATALINA_HOME and \$CATALINA_BASE are nothing but the pointing to the root folder of TOMCAT.

So in our folder structure,

\$CATALINA_HOME = \$CATALINA_BASE = "D:\Work\apache-tomcat-6.0.41"

But in LINUX both of these are pointing to different directories.

\$CATALINA_HOME = "/usr/share/tomcat6"

\$CATALINA_BASE = "/var/lib/tomcat6"

Note – 2: (Error Details)

Also I got an error on Tomcat stating **slf4j** error. Just include that dependencies in pom nad you can eliminate them.