Labo Spring 2
- *Template* -

Objectif

Utiliser un template pour standardiser l'apparence de toutes les pages jsp.

N.B. Vous devez implémenter les étapes 1 à 6 avant de pouvoir déployer et tester votre application Web.

Étape 1 : pom.xml

Une dépendance est présente dans le pom.xml pour pouvoir utiliser la librairie Tiles permettant d'appliquer des templates aux pages jsp.

```
<dependency>
    <groupId>org.apache.tiles</groupId>
    <artifactId>tiles-jsp</artifactId>
    <version>3.0.8</version>
</dependency>
```

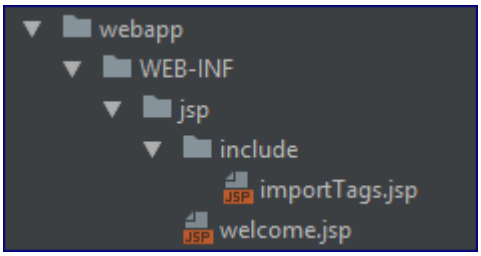
Etape 2 – Import des librairies de tags

Plusieurs librairies de tags devront être importées dans toutes les pages jsp. Ces imports vont être placés une fois pour toutes dans le fichier *importTags.jsp* qui sera lui-même importé dans toutes les pages jsp.

Créez un répertoire *include* dans WEB-INF/jsp.

Créez-y le fichier *importTags.jsp* qui contient les différents imports de librairies de tags.

New ⇨ File ⇨ Name : *importTags.jsp*



Librairie de tags

Des librairies de tags peuvent être utilisées dans les pages jsp.

A chaque librairie de tags va être associé un préfixe.

Par exemple, une librairie propose des balises pour la gestion des formulaires ; on va lui associer le préfixe "*form*" :
<%@ taglib ***prefix="form"*** *uri="http://www.springframework.org/tags/form" %>*

Chaque fois que le programmeur souhaitera utiliser une balise de cette librairie, il devra la préfixer par "*form*". Par exemple pour appeler la balise "*button*" de cette librairie pour créer un bouton dont l'intitulé est "*Submit*", il faut écrire :
*<**form**:button>Submit</**form**:button>*

Déclarez les librairies suivantes dans le fichier *importTags.jsp* :

importTags.jsp

```
<%@ taglib prefix="c"       uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="spring"  uri="http://www.springframework.org/tags"%>
<%@ taglib prefix="form"    uri="http://www.springframework.org/tags/form" %>
<%@ taglib prefix="fmt"     uri="http://java.sun.com/jsp/jstl/fmt" %>
```

Adaptez la page *welcome.jsp* (et toutes les futures pages) : ajoutez-y l'include du fichier *importTags.jsp.*

welcome.jsp

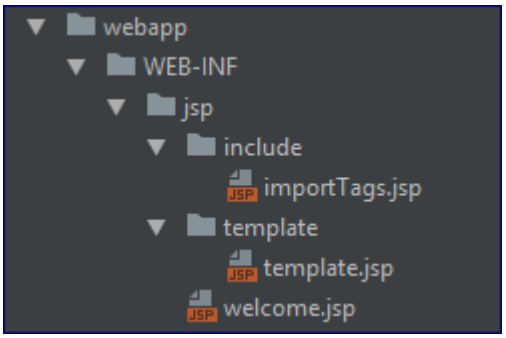
```
<%@ page pageEncoding="UTF-8"
        contentType="text/html; charset=UTF-8"%>
<%@ include file="include/importTags.jsp"%>
<html>
<head>
   <title>Welcome</title>
</head>
<body>
Welcome our wonderful world !
</body>
</html>
```

Etape 3 - Création du template

Template

Le template est une page jsp qui contient la structure commune à toutes les pages et qui servira donc de base à la création de toutes les autres pages.

Créez un répertoire *template* dans *WEB-INF/jsp*. Créez-y le fichier *template.jsp*.



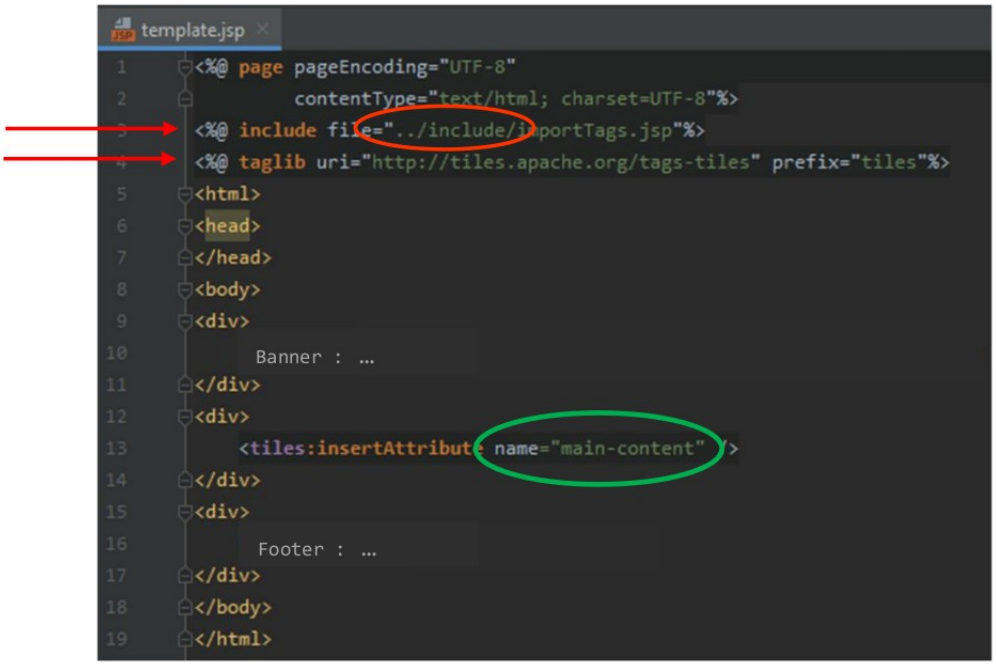
Implémentez le format de template de votre choix. Prévoyez un bandeau dans la partie supérieure de chaque page (ex : nom du site, messages, liens et infos diverses…) et un pied de page dans la partie inférieure de chaque page (ex : copyright, autres liens utiles…). Des images (logo…) pourront être ultérieurement ajoutées dans le template (voir étape 8).

Nous allons utiliser ***tiles*** comme gestionnaire de template. Ajoutez pour ce faire la librairie correspondante dans le *template.jsp* :

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

N'oubliez pas d'inclure le fichier *importTags.jsp* en spécifiant le chemin correct.

Ajoutez ensuite dans *template.jsp* un tag *<tiles:insertAttribute>* afin de définir la zone éditable, c'est-à-dire la zone qui sera remplacée par le contenu de la page client. Donnez un nom à cette zone : "*main-content*". La zone *main_content* du template sera donc remplacée par le contenu de chaque page jsp.



Etape 4 – tiles.xml

Fichier de configuration du template

Le fichier *tiles.xml* contient la définition du template à utiliser et des informations sur l'appel des pages jsp sur base du template.

Le répertoire *resources* dans *WEB-INF* contient le fichier de configuration *tiles.xml*.

tiles.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE tiles-definitions PUBLIC
        "-//Apache Software Foundation//DTD Tiles Configuration 3.0//EN"
        "http://tiles.apache.org/dtds/tiles-config_3_0.dtd">
<tiles-definitions>
    <!--
    We declare a new template named template-main.
    This template is used for displaying the main page.
    It has 4 attributes. These attributes are placeholder for our contents
    For each attribute, we have assigned a corresponding JSP
    -->
    <definition name="template-main" template="/WEB-INF/jsp/template/template.jsp">
        <put-attribute name="main-content" value="" />
    </definition>

    <!-- "ajax:" renders the page as-is, without the template -->
    <definition name="ajax:**" template="/WEB-INF/jsp/ajax/{1}.jsp" />

    <!-- "tiles:" renders the specified page within the template-main -->

    <definition name="integrated:**" extends="template-main">

        <put-attribute name="main-content" value="/WEB-INF/jsp/{1}.jsp" />
    </definition>

    <definition name="error" extends="template-main">
        <put-attribute name="main-content" value="/WEB-INF/jsp/error.jsp" />
    </definition>
</tiles-definitions>
```

Ce fichier de configuration précise entre autres :

- Le nom donné à la zone du template qui devra être remplacée par le contenu de chaque page (ici : "***main-content***") ; la zone *main_content* du template sera donc remplacée par le contenu de chaque page jsp.

- La chaîne de caractères qu'on utilisera pour préfixer le nom des pages jsp auxquelles on veut appliquer le template (ici : "***integrated:***"*).* L'appel des pages jsp devra désormais être préfixé de "*integrated :*" pour que le template soit appliqué. Notez que "*integrated*" n'est pas un mot réservé. On peut choisir n'importe quelle chaîne de caractères. Pour rappel, le nom des pages jsp à afficher est précisé dans le résultat (return) des méthodes *get* des controllers. C'est donc dans le return de ces méthodes que le nom des pages jsp sera préfixé de "*integrated*:" (cf. étape 6).

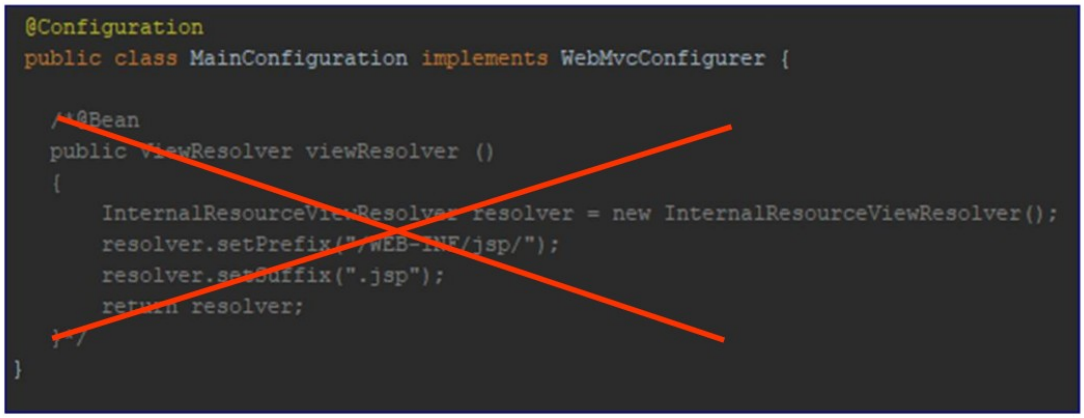
Etape 5 – TilesConfigurer et ViewResolver

Rôle du ViewResolver

Pour rappel, le rôle d'un *ViewResolver* est de localiser les pages JSP à partir entre autres du répertoire où elles sont stockées et de leur suffix jsp.

Il faut adapter le *ViewResolver* de la série 1 afin que le template soit pris en compte.

Adaptez la classe *MainConfiguration* : supprimez le bean *ViewResolver* (ou mettez-le en commentaires).



La classe *TilesConfiguration* dans le package *configuration* contient 2 beans : *TilesConfigurer* et *ViewResolver* (injection de dépendance). Décommentez les méthodes de cette classe.

Le bean *tilesConfigurer* permet entre autres de préciser où se trouve le fichier de configuration de Tiles (*tiles.xml*).

TilesConfiguration.java

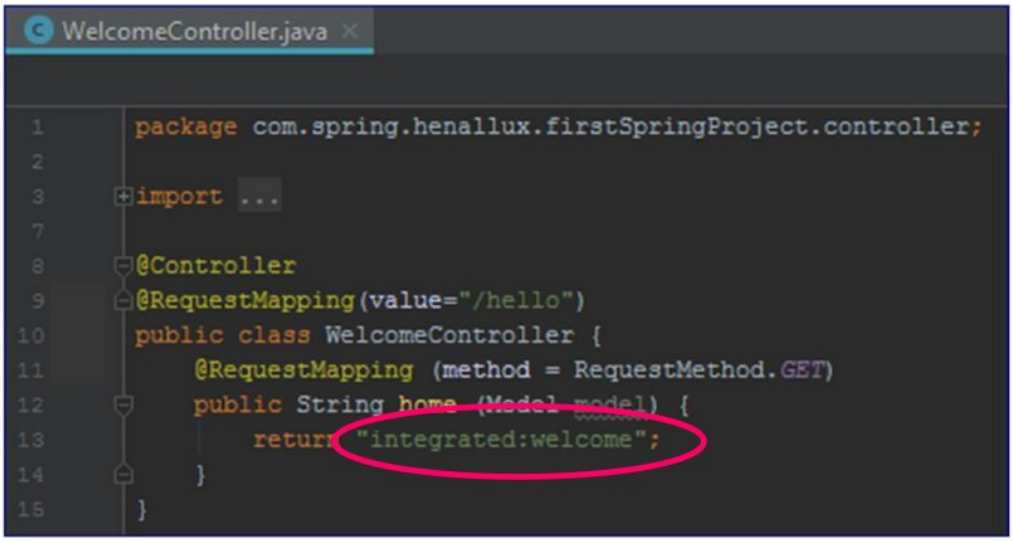
```java
package com.spring.henallux.firstSpringProject.configuration;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.servlet.ViewResolver;
import org.springframework.web.servlet.view.tiles3.TilesViewResolver;
import org.springframework.web.servlet.view.tiles3.TilesConfigurer;
import org.springframework.web.servlet.view.tiles3.TilesView;

@Configuration
public class TilesConfiguration {
    @Bean
    public TilesConfigurer tilesConfigurer()
    {
        final TilesConfigurer configurer = new TilesConfigurer();
        configurer.setDefinitions("WEB-INF/resources/tiles.xml");
        configurer.setCheckRefresh(true);
        return configurer;
    }

    @Bean
    public ViewResolver tilesViewResolver ()
    {
        final TilesViewResolver resolver = new TilesViewResolver();
        resolver.setViewClass(TilesView.class);
        return resolver;
    }
}
```

Etape 6 - Appel des pages jsp dans les controllers

L’appel de toute page jsp dans un controller doit désormais être préfixé de "*integrated* :", comme renseigné dans le fichier *tiles.xml,* pour que le template soit appliqué sur ces pages.

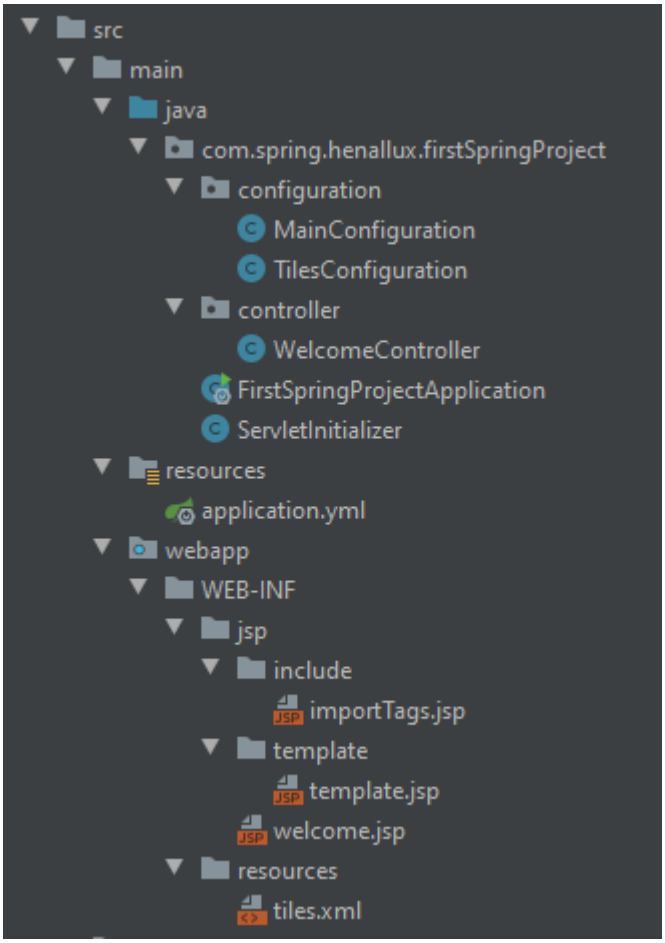


Déployez l’application et testez-la.

En cas d'erreur…

Comme précisé dans le labo 1, vérifiez que le path de votre application ne contient aucun nom de répertoire avec des espaces ou des caractères spéciaux.

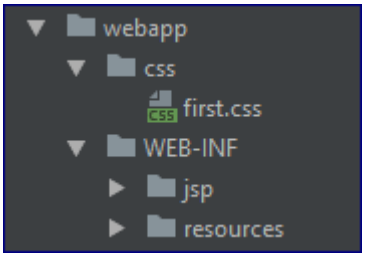
Les erreurs les plus fréquentes viennent aussi souvent d'une mauvaise structure des répertoires. Vérifiez la structure de votre application.



Etape 7 – Utilisation de CSS

Créez un répertoire css dans src/main/webapp.

Placez-y un fichier CSS de votre choix.



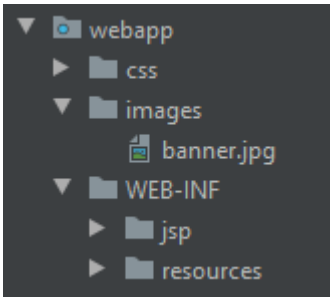
Utilisez ce fichier CSS dans la partie *head* du template en utilisant *<spring :url>* pour définir le chemin du fichier CSS.

```
template.jsp
3   <%@ include file="../include/importTags.jsp"%>
4   <%@ taglib uri="http://tiles.apache.org/tags-tiles" prefix="tiles"%>
5   <html>
6   <head>
7       <link type="text/css" href="<spring:url value='/css/first.css' />"
8             rel="Stylesheet">
```

Etape 8 – Images

Créez un répertoire *images* dans *src/main/webapp*.

Placez-y des fichiers image.

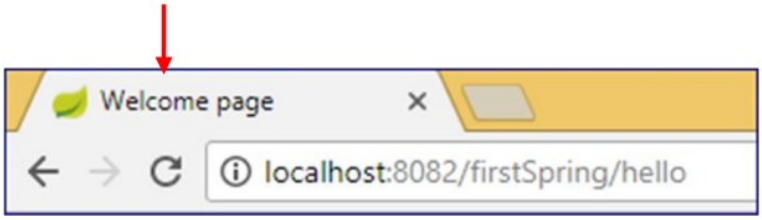


Affichez ces images dans le template en utilisant *<spring :url>* pour définir le chemin de l'image (placez par exemple le logo du site dans le bandeau du template).

```
<img src ='<spring:url value="/images/banner.jpg"/>' />
```

Etape 9 – Balise <title> des pages

Modifiez l'intitulé de l'onglet correspondant à la page.



Les valeurs des intitulés de page devront être fournies par les controllers via un attribut du dictionnaire *Model* appelé par exemple *title*. Pour associer un attribut au model, référez-vous à l'étape 11 du labo 1.

Récupérez la valeur de cet attribut du *Model* et affichez-la dans la balise <*title*> dans <*head*> du template.

```
<title>${title}</title>
```