Labo Spring 2

Template

Objectif : utiliser un template pour toutes les pages jsp.

Etape 1 – pom.xml

Ajoutez une dépendance

```
<dependency>
    <groupId>org.apache.tiles</groupId>
    <artifactId>tiles-jsp</artifactId>
    <version>3.0.4</version>
</dependency>
```

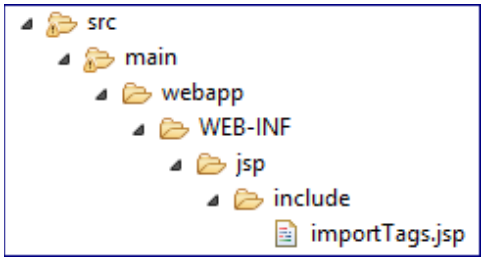
Etape 2 – Import des librairies de tags

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 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.



importTags.jsp

```
1 <%@ taglib prefix="c"       uri="http://java.sun.com/jsp/jstl/core" %>
2 <%@ taglib prefix="spring"  uri="http://www.springframework.org/tags"%>
3 <%@ taglib prefix="form"    uri="http://www.springframework.org/tags/form" %>
4 <%@ 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

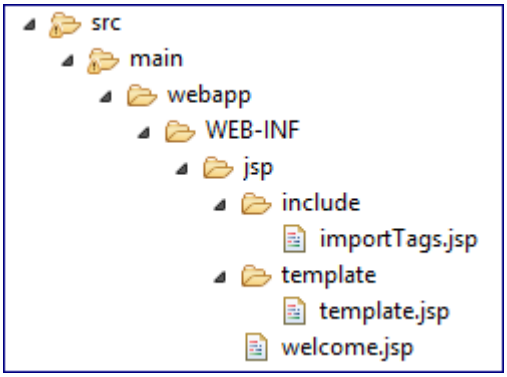
```
1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2      pageEncoding="UTF-8"%>
3  <%@ include file="include/importTags.jsp"%>
4  <html>
5  <head>
6  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
7  <title>Welcome</title>
8  </head>
9  <body>
10 Welcome in our magic world!
11 </body>
12 </html>
```

Etape 3 - Création du template

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

Créez-y le fichier *template.jsp*.

Implémentez le format de template de votre choix.

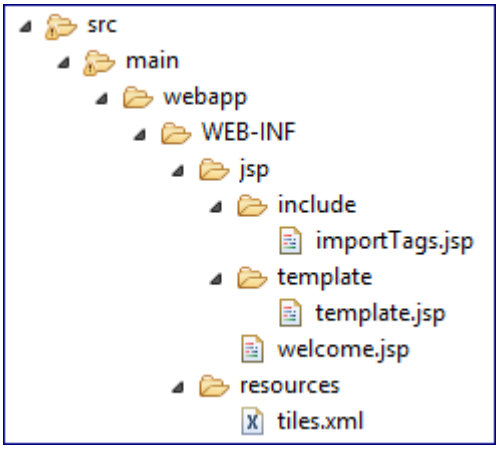



```
<%@ include file="../include/importTags.jsp"%>
<%@ taglib uri="http://tiles.apache.org/tags-tiles" prefix="tiles"%>

<!DOCTYPE HTML>
<html>

        <head>
        ...
        </head>

        <body>
        ...
                <div>
                        <tiles:insertAttribute name="main-content" />
                </div>
        ...
        </body>
</html>
```

Etape 4 – tiles.xml

Créez un répertoire *resources* dans *WEB-INF*.

Créez-y le fichier de configuration *tiles.xml* contenant la définition du template à utiliser et des informations sur l'appel des pages jsp sur base du template (*vous pouvez vous inspirer du fichier exemple_tiles.xml disponible sur Claroline).*




```
<?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>
```

L'appel des pages jsp devra désormais être préfixé de ***"integrated :"***, comme précisé dans le fichier *tiles.xml*, pour que le template soit appliqué.

La zone ***main_content*** du template sera remplacée par le contenu de chaque page jsp.

Etape 5 – Tiles Configurer et View Resolver

Adaptez la classe *MainConfig* : supprimez le bean *ViewResolver*.

```
@Configuration
public class MainConfig extends WebMvcConfigurerAdapter {

    @Bean
    public ViewResolver viewResolver ()
    {
        InternalResourceViewResolver resolver = new InternalResourceViewResolver();
        resolver.setPrefix("/WEB-INF/jsp/");
        resolver.setSuffix(".jsp");
        return resolver;
    }
}
```

Créez la classe *TilesConfig* dans le package *config* de *src/main/java/com.spring.henallux*. Cette classe contient 2 beans : *TilesConfigurer* et *ViewResolver* (injection de dépendance*) (vous pouvez vous inspirer du fichier exemple_TilesConfig disponible sur Claroline).*

```
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 TilesConfig {
    @Bean
    public TilesConfigurer tilesConfirgurer()
    {
        final TilesConfigurer configurer = new TilesConfigurer();
        configurer.setDefinitions(new String[] {"WEB-INF/resources/tiles.xml"});
        configurer.setCheckRefresh(true);
        return configurer;
    }

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

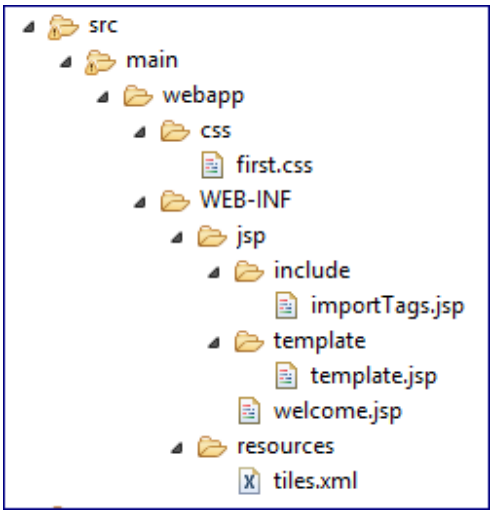
Etape 6 - Appel des pages jsp dans les controllers

L'appel de toute page jsp dans un controller doit désormais être préfixé de *"integrated :"*.

```
@Controller
@RequestMapping(value="/welcome")
public class WelcomeController {

    @RequestMapping(method=RequestMethod.GET)
    public String home(Model model) {
        return "integrated:welcome";
    }
}
```

Etape 7 – Utilisation de CSS

Créez un répertoire *css* dans *src/main/webapp*.
Placez-y un fichier CSS de votre choix.



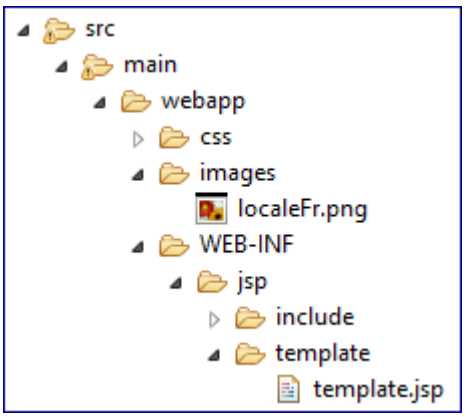
Utilisez ce fichier CSS dans le template en utilisant *<spring :url>* pour définir le chemin du fichier CSS.

template.jsp

```
1  <%@ include file="../include/importTags.jsp"%>
2  <%@ taglib uri="http://tiles.apache.org/tags-tiles" prefix="tiles"%>
3
4  <!DOCTYPE HTML>
5  <html>
6
7  <head>
8  <meta http-equiv="Content-Type" content="type=text/html; charset=utf-8">
9
10 <link type="text/css" href="<spring:url value='/css/first.css' />"
11       rel="Stylesheet">
```

Etape 8 – Images

Créez un répertoire *images* dans *src/main/webapp*.
Placez-y des fichiers image.

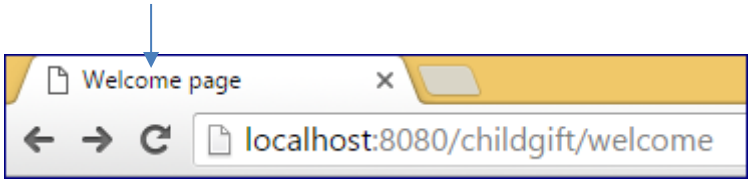


Affichez ces images dans le template en utilisant *<spring :url>* pour définir le chemin de l'image.

```
<img src='<spring:url value="/images/localeFr.png"/>'/>
```

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 *Model*.

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

Etape 10 – Header et Footer

Adaptez le template et éventuellement les fichiers CSS en conséquence pour prévoir un header (ex : Un logo et/ou nom du site) et un footer (ex : un copyright et une année) dans le template.