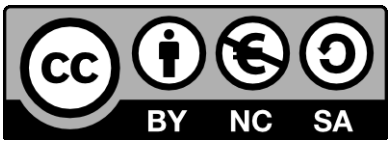
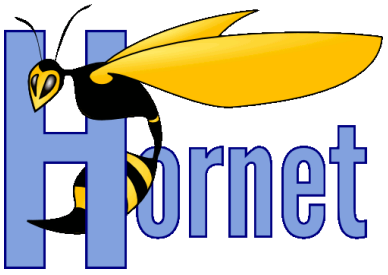


[HORNET]
Migration d'un projet Hornet 2.0 vers Hornet 3.1



Développement Hornet

Version : 1.1 du 22/08/2013 - **Etat** : Validé

SUIVI DES MODIFICATIONS

Version	Auteur	Description	Vérification	Date
1.0	O. Coatanlem F. Bernier-Malcoiffe A. Frigout	Initialisation du document à partir du guide de migration d'un projet Hornet 2.0 vers Hornet 3.0. Mise à jour Hornet 3.1.	S. Percier	10/6/2013
1.1	PY Baloche	Mise à jour Hornet 3.1.2	S. Percier	22/08/13

DOCUMENTS DE REFERENCE

Titre
HORNET_GUI_Création d'un projet Hornet_3.1B
HORNET_GUI_Guide du développeur Hornet 3.1_1.1
HORNET_GUI_Guide de paramétrage_1.1
HORNET_NOR_Normes d'encodage_1.0

SOMMAIRE

TABLEAUX

FIGURES

1 Objectifs du document

Dans le cadre de la mise en place des évolutions au sein du Framework Hornet, ce document spécifie les actions à effectuer au sein d'un projet de type Hornet pour migrer vers la nouvelle version du Framework Hornet 3.1B à partir de la version précédente (2.0).

1.1 Principales évolutions

La version 3 de Hornet comporte des évolutions importantes concernant principalement l'évolution de composants techniques, de l'environnement de développement et de la plateforme cible. Les deux tableaux ci-dessous listent ces principales évolutions.

Composants plateforme cible	Hornet 2.0	Hornet 3.1B
Serveur d'application	Tomcat 5.5.x	Tomcat 6.0.x
Java	JDK 1.5.x	JDK 1.6.x

Tableau 1 : Différences sur serveur d'application et JRE

Composants techniques	Hornet 2.0	Hornet 3.1B
Spring	2.5.1	3.0.7
Spring Security	-	3.1.3
Struts	2.0.11.2	2.3.15.1

Tableau 2 : Différences sur composants majeurs

1.2 Versions cible des composants

Ce document est basé sur les versions de composants suivantes :

- hornetserver 3.1.2
- hornetclient 3.1.1
- hornettemplate 3.1.2

2 Mise à jour technique

2.1 Pré requis

2.1.1 Création d'un projet « modèle » à partir de hornettemplate

La migration vers Hornet 3.1B nécessite de récupérer de nouveaux fichiers et des fichiers mis à jour dans cette nouvelle version du framework. Ainsi, la source de ces fichiers doit être un projet « modèle » créé à partir de hornettemplate (cf. Guide de création d'un projet Hornet).

Dans la suite de ce document, sauf mention d'une autre source, les fichiers mentionnés sont donc ceux du projet créé à partir de hornettemplate **3.1.2.**

2.1.2 Thèmes

Les thèmes doivent être installés sur un serveur de framework Hornet.

2.2 Configuration du projet sous Eclipse

2.2.1 Pré-requis

Avant de commencer cette migration, il faut avoir installé et configuré l'environnement de développement conformément au guide de paramétrage, en particulier pour la configuration de Tomcat 6 et JDK 6 dans Eclipse.

2.2.2 Encodage du projet

Dans les projets éclipse à migrer, vérifier que l'encodage UTF-8 n'est pas hérité de la configuration du workspace mais bien forcée dans les propriétés de chaque projet. De la même manière, pour forcer le format de fin de ligne en mode Unix, sur le même écran, vérifier l'option New text file line delimiter, à positionner sur Unix.

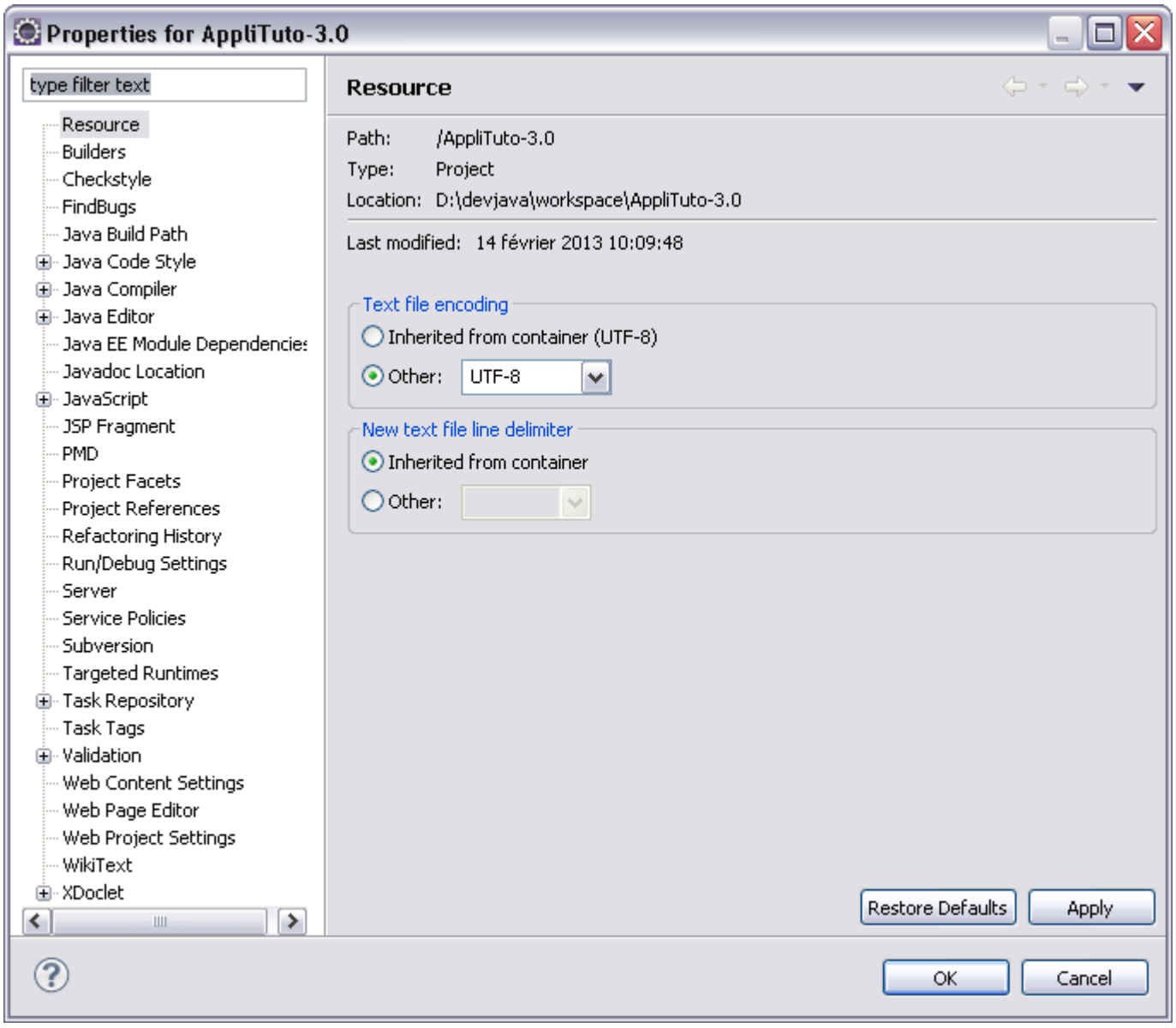


Figure 1 : Propriété d'encodage d'un projet éclipse

Dans le groupe d'options « Text file encoding » :

- Activer « Other »
- Choisir « UTF-8 »
- Appliquer les modifications

> ➲ **Vérifier que les différents fichiers du projet respectent les règles décrites dans le document « Norme d'encodage » (§ « Composants applicatifs concernés par l'encodage »).**

2.2.3 Conversion des fichiers en UTF-8

Pour les fichiers le nécessitant, la conversion de l'encodage pourra être réalisée à l'aide d'un éditeur externe (type Notepad++ ou UltraEdit par exemple) ou d'un outil spécifique.

Quelle que soit la méthode utilisée, la conversion doit être faite vers l'UTF-8 **sans BOM**.

2.2.4 Version de Java

- Ouvrir le menu « Java Build Path » des propriétés du projet

- Supprimer la référence à la librairie de la JRE 1.5
- Cliquer sur « Add Library … »
- Sélectionner « Execution environment » et choisir « JavaSE-1.6 » :

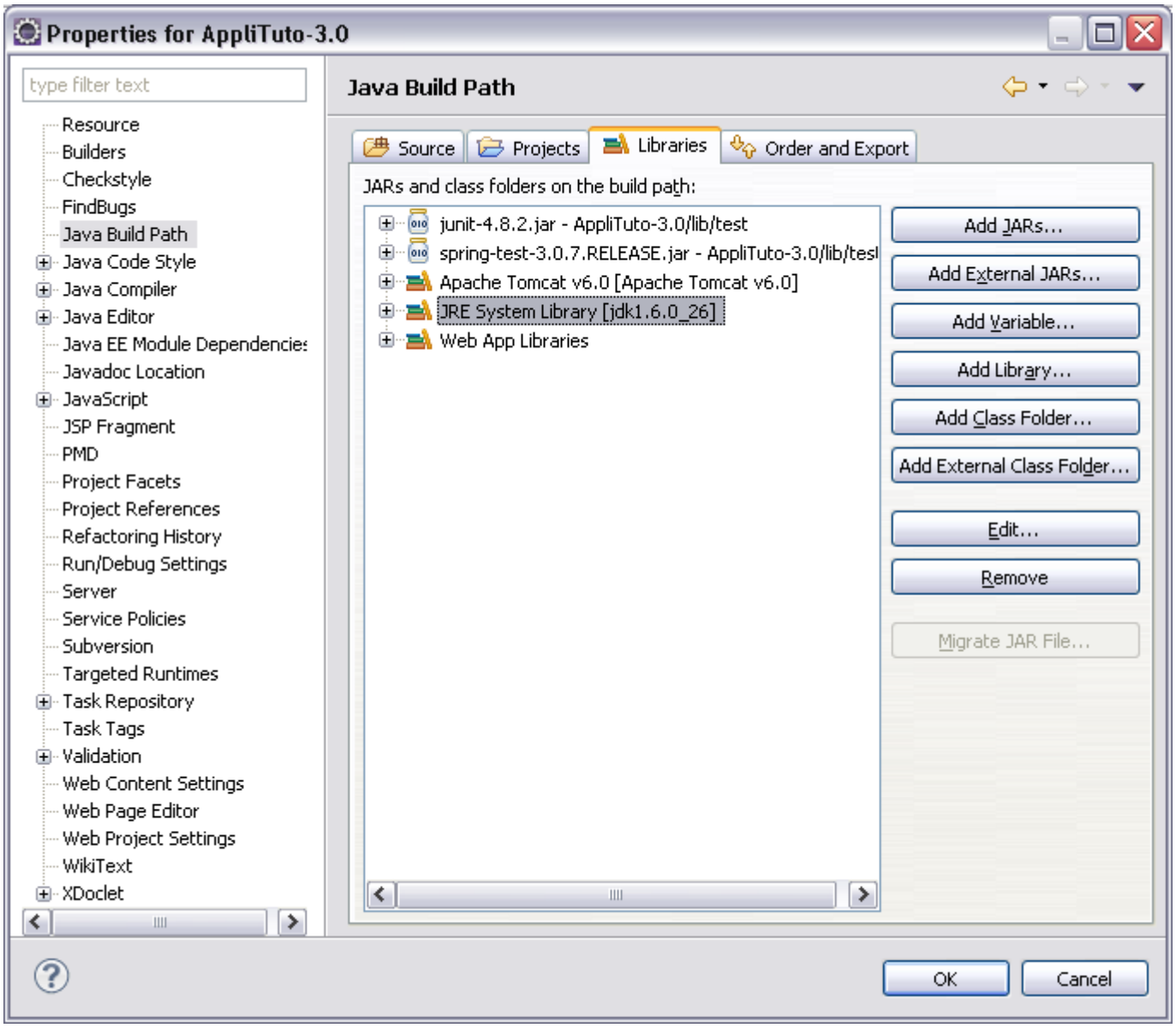


Figure 2 : Propriété de JRE éclipse

- Ouvrir le menu « Java Compiler » des propriétés du projet
- Valider les modifications
- Vérifier que « Enable project specific settings » est coché et choisir « 1.6 » comme valeur pour « Compiler compliance level » :

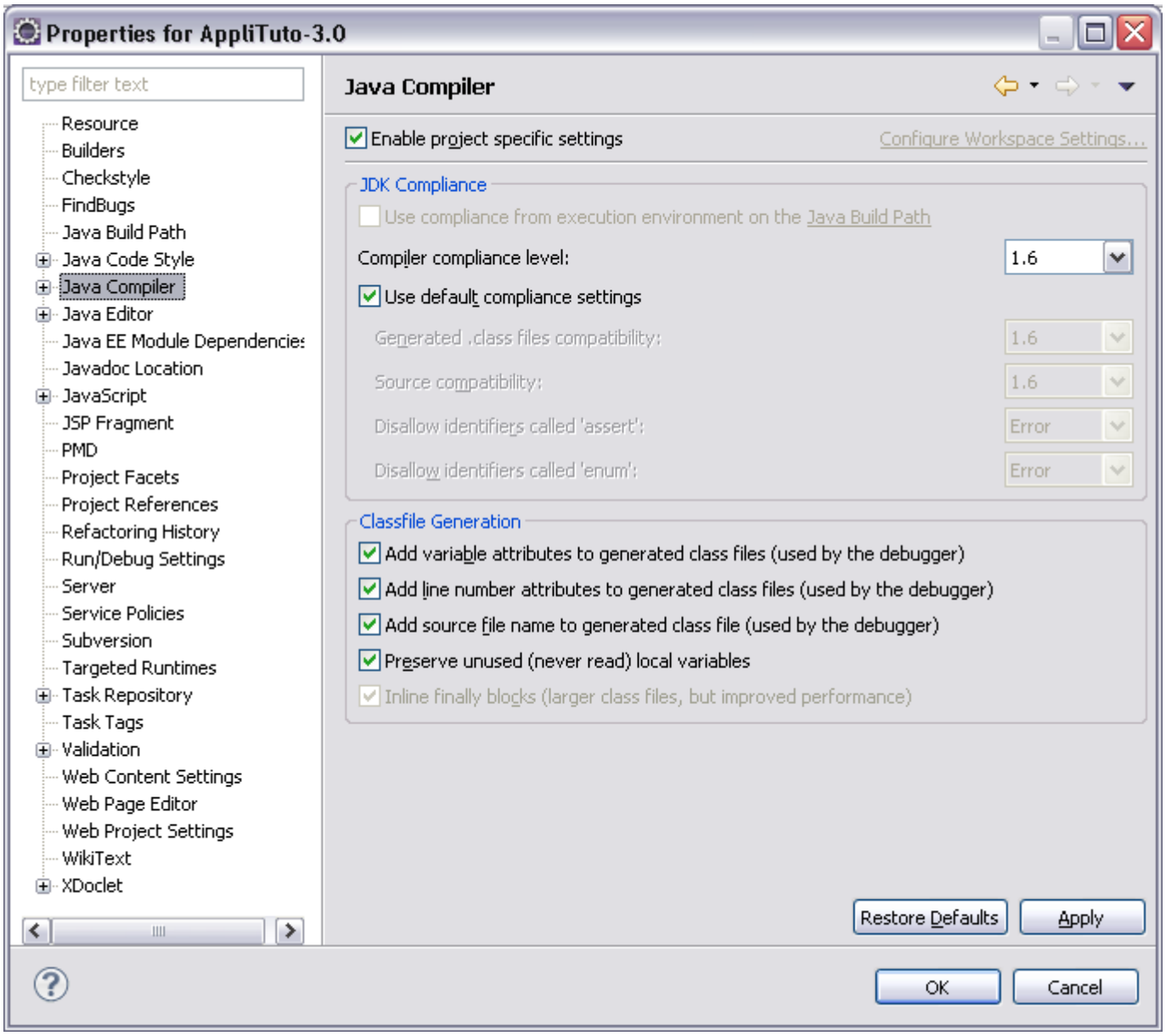


Figure 3 : Propriété de JDK éclipse

- Appliquer les modifications
- Accepter la recompilation du projet

- Ouvrir le menu « Project Facets » des propriétés du projet :

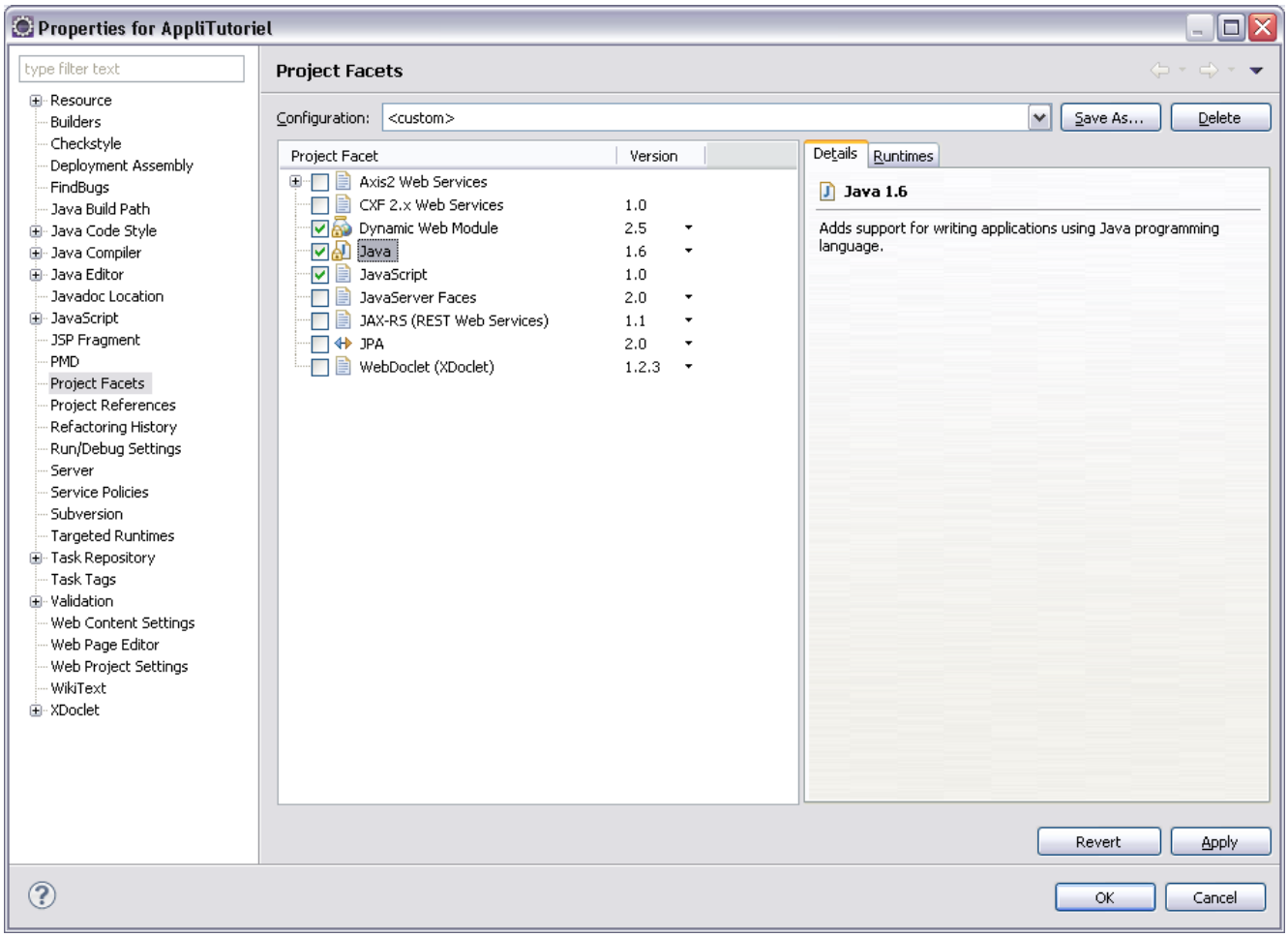


Figure 4 : Propriété de Webapp eclipse

- Pour « Java » choisir « 1.6 »
- Pour « Dynamic Web Module », choisir « 2.5 » puis appliquer les modifications

2.2.5 Librairies serveur

Hornet 2 est conçu pour fonctionner avec la version 5.5 de Tomcat alors qu'Hornet 3 se base sur la version 6.0. Il faut donc remplacer toutes ces références dans les propriétés des différents projets de l'application à migrer.

- Ouvrir le menu « Java Build Path » des propriétés d'un projet web
- Supprimer la référence à la librairie « Apache Tomcat v5.5 » :

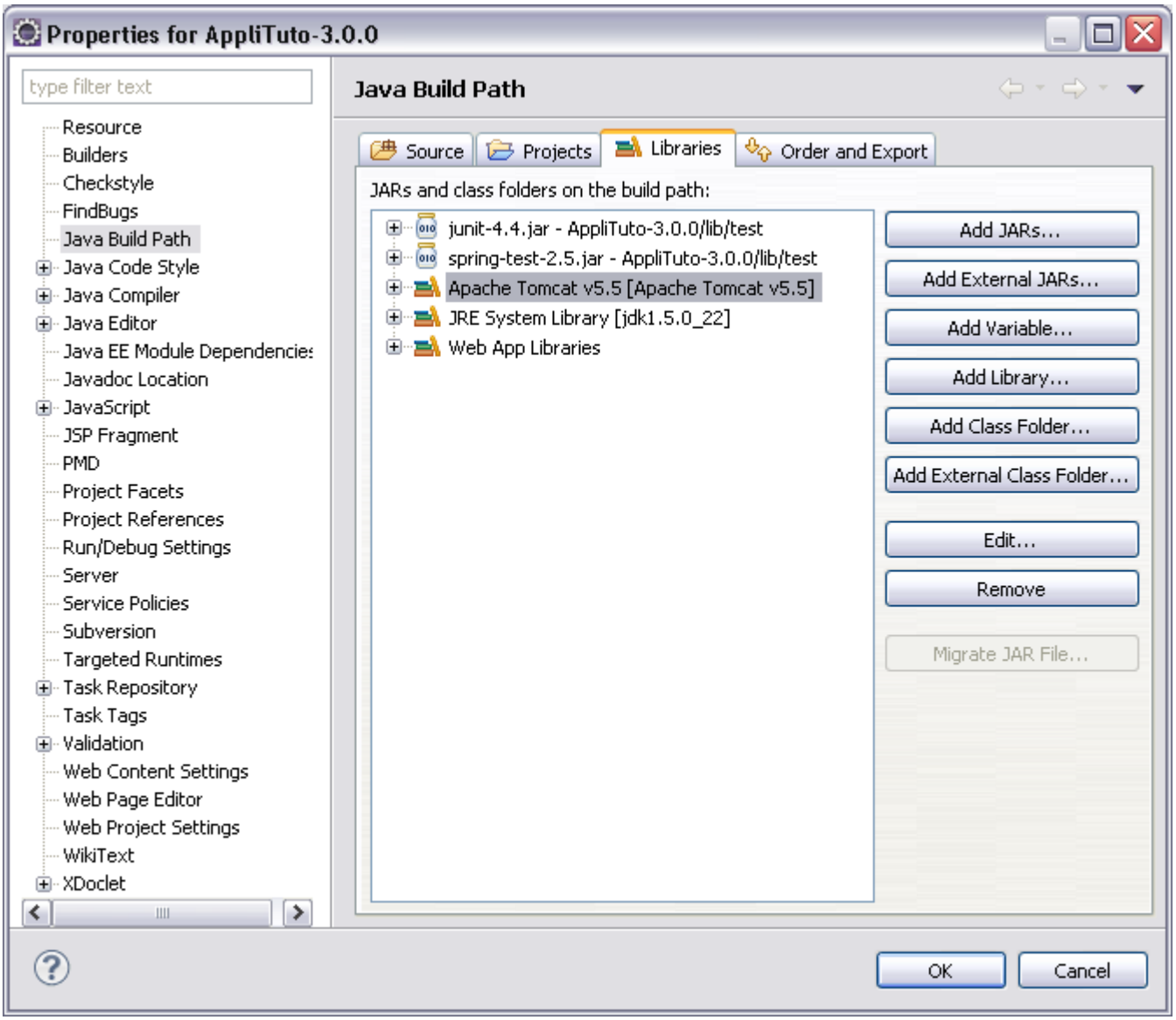


Figure 5 : Propriété de Lbrairies Java eclipse

- Ajouter la librairie correspondant au serveur Tomcat 6.0 :
  - Cliquer sur « Add Library … »
  - Sélectionner « Server Runtime » puis « Next »
  - Choisir « Apache Tomcat 6.0 » puis « Finish »
- Dans le menu « Targetted Runtimes » cocher « Apache Tomcat v6.0 » et décocher « Apache Tomcat v5.5 »

2.2.6 Configuration Ivy du projet

Dans le fichier de configuration Ivy du projet (`ivy.xml` le plus souvent), les tags « exclude » présents dans la configuration des dépendances doivent être mis à jour (cf. Cahier des charges Hornet 3.0 > Annexes > Synthèse des mises à jour de librairies de hornetserver) pour concorder avec la nouvelle configuration de hornetserver.

Supprimer les tags liés à des dépendances obsolètes.

2.2.7 Respect des normes Tomcat 6.0

Modification du web.xml, remplacer :

```
<web-app id="WebApp_ID" version="2.4"
    xmlns="http://java.sun.com/xml/ns/j2ee"
```

```
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
```

par :

```
<web-app id="WebApp ID" version="2.5"
    xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-
app_2_5.xsd">
```

2.3 Gestion des dépendances

2.3.1 Mise à jour des librairies Ivy

Les librairies Ivy à utiliser sont en v2.3.0.

- Remplacer les librairies Ivy par celles du projet créé à partir de hornettemplate 3.1.2

2.3.2 Mise à jour du repository

- Récupérer le repository compatible Hornet **3.1.2.**
- Décompresser l'archive en lieu et place de la version « **2.0** »
  - **D:\devjava\workspace\Repository**

Les développements doivent pointer vers ce nouveau repository.
Le nouveau repository doit être celui par défaut.

Les configurations sont à réaliser dans le fichier ivysettings.properties à la racine du projet.

```
# dépot en ligne pour les fichiers du framework acube et ses dépendances
repository.metier.url=D:/devjava/workspace/Repository/metier
repository.metier.artifact.pattern=${default.artifact.pattern}
repository.metier.ivy.pattern=${default.ivy.pattern}

repository.technique.url=D:/devjava/workspace/Repository/technique
repository.technique.artifact.pattern=${default.artifact.pattern}
repository.technique.ivy.pattern=${default.ivy.pattern}

repository.cache.url=D:/devjava/workspace/Repository/cache
repository.cache.artifact.pattern=${default.artifact.pattern}
repository.cache.ivy.pattern=${default.ivy.pattern}

repository.snapshot.url=D:/devjava/workspace/Repository/snapshot
repository.snapshot.artifact.pattern=${default.artifact.pattern}
repository.snapshot.ivy.pattern=${default.ivy.pattern}
```

2.3.3 Mise à jour du ivysettings.xml

- Ajouter l'en-tête XML suivant au fichier de déclaration des dépendances ivy.xml

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

2.4 Configuration de l'application

2.4.1 Tâche Ant

- **Supprimer**, s'ils existent, les fichiers : « **buildTemplate.xml** » et « **buildTemplate.properties** ».
- **Remplacer** le fichier de construction du projet « **build.xml** ».

Mettre à jour le fichier de propriété : « build.properties » en prenant en compte les anciens paramètres :

- Mettre à jour le numéro de version de la propriété build.jarFwServerVersion avec la version courante du framework Hornet Serveur :

```
build.jarFwServerVersion=3.1.2
```

Les nouvelles propriétés build.projectPackage et build.projectOrganisation sont utilisées pour distinguer le nom de package principal utilisé dans l'application et le nom de l'organisation référencé dans le fichier de description des dépendances ivy.xml ainsi que dans le fichier Manifest du jar ou du war généré.

Remplacer le fichier de construction « **dynamique.xml** »

2.4.2 hornetserver

2.4.2.1 Gestion des dépendances

- Faire pointer le projet vers la version « **hornetserver-all 3.1.2**».

Exemple :

```
<dependency org="fr.gouv.diplomatie.hornet" name="hornetserver-all" rev="3.1.2" conf="compile-
>core,libDependances;runtime->runtime;test->test" transitive="true">
    <exclude name="avalon-framework" />
    <exclude name="batik-all" />
    <exclude name="cglib-nodep" />
    <exclude name="dom4j" />
    <exclude name="fop" />
    <exclude name="itext" />
    <exclude name="jasperreports" />
    <exclude name="jcaptcha-all" />
    <exclude name="jtds" />
    <exclude name="mysql-connector-java" />
    <exclude name="poi" />
    <exclude name="quartz-all" />
    <exclude name="velocity" />
    <exclude name="xmlgraphics-commons" />
    <exclude name="xstream" />
</dependency>
```

La liste des librairies à exclure est à mettre à jour vis-à-vis de la configuration « core » de Hornet.

2.4.2.2 Autres dépendances et update des librairies

- Lancer la tâche Ant « **importHornetLibs** » afin de récupérer les nouvelles librairies dans le répertoire : « **WEB-INF/lib** ».
- Vérifier la présence et la mise à jour des nouvelles dépendances :
  - **hornetserver-core-3.1.2.jar**
  - **lib/test/junit-4.8.2.jar**
  - **lib/test/spring-test-3.0.7.RELEASE.jar**

2.4.3 hornetclient

2.4.3.1 Ressources statiques

Modifier le fichier « **web.xml** », afin de pointer la variable « **fwkRoot** » vers la nouvelle version du framework.

```
<context-param>
        <param-name>fwkRoot</param-name>
        <param-value>http://130.177.222.245/hornet/hornetclient/3.1.1/fwk</param-value>
</context-param>
```

2.4.3.2 Thèmes

- Ajouter dans le fichier « **web.xml** », les variables suivantes afin de définir le thème et sa version :
  - **themeName**
  - **themeVersion**

```
    <context-param>
        <param-name>themeName</param-name>
        <param-value>[nom du theme]</param-value>
    </context-param>

    <!-- Version du theme intranet/internet (vide pour le theme par defaut) -->
    <context-param>
        <param-name>themeVersion</param-name>
```

```
        <param-value>3.1.2</param-value>
    </context-param>
```

2.4.4 YUI

La référence au framework Yahoo UI doit être mise à jour dans le fichier web.xml.
Remplacer :

```
<context-param>
    <param-name>yui3Root</param-name>
    <param-value>http://130.177.222.247/hornet/yui/yui/3.4.1</param-value>
</context-param>
```

Par :

```
<context-param>
    <param-name>yui3Root</param-name>
    <param-value>http://130.177.222.247/hornet/yui/yui/3.8.1</param-value>
</context-param>
```

2.4.5 Struts

2.4.5.1 Remplacement des result-types et interceptors dépréciés

Depuis Struts 2.1, certains noms de result-types et d'interceptors ont été dépréciés et renommés en utilisant la casse « CamelCase ».
Appliquer les modifications suivantes au fichier « **/src/config/struts.xml** » :

- Ajouter l'entête xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE…
```

- Si les result-types suivants sont présents, les renommer avec la casse « camelCase » sans tiret :
  - redirect-action → redirectAction
  - plaintext → plainText
- Si les interceptors suivants sont présents, les renommer avec la casse « camelCase » :
  - external-ref → externalRef
  - model-driven → modelDriven
  - static-params → staticParam
  - scoped-model-driven → scopedModelDriven
  - servlet-config → servletConfig
  - token-session → tokenSession

2.4.5.2 Remplacement du FilterDispatcher

Dans le fichier web.xml, l'utilisation du FilterDispatcher est dépréciée depuis la version 2.1 de Struts.
Remplacer :

```
<filter-class>org.apache.struts2.dispatcher.FilterDispatcher</filter-class>
```

Par :

```
<filter-class>org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter</filter-class>
```

2.4.5.3 Mise à jour des DTD des fichiers XML de validation

L'emplacement de la DTD des fichiers xml pour la validation Struts a changé.
Remplacer :

```
<!DOCTYPE validators PUBLIC "-//OpenSymphony Group//XWork Validator 1.0.2//EN"
"http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd">
```

Par :

```
<!DOCTYPE validators PUBLIC "-//Apache Struts//XWork Validator 1.0.2//EN"
"http://struts.apache.org/dtds/xwork-validator-1.0.2.dtd">
```

2.4.5.4 Mise à jour des tags dans les jsp : s:datetimepicker

Le tag `s:datetimepicker` a été déplacé dans un plugin Dojo pour Struts.
Son équivalent a été recréé dans la taglib Hornet et renommé `calendar`.

Dans les pages JSP concernées :

- Importer la taglib Hornet si elle n'est pas déjà présente :

```
<%@ taglib uri="/hornet-tags" prefix="hornet" %>
```

- Modifier les tags en erreur en les préfixant par `hornet:calendar` au lieu de `s:datetimepicker`
  Par exemple, remplacer :

```
<s:datetimepicker cssClass="yui3-filtre-field" id="filtreDateModif" name="criteres.partenaire.dateModif"
displayFormat="dd/MM/yyyy" iconPath="%{icoAgendaUrl}" />
```

Par :

```
<hornet:calendar cssClass="yui3-filtre-field" id="filtreDateModif" name="criteres.partenaire.dateModif"
displayFormat="dd/MM/yyyy" iconPath="%{icoAgendaUrl}" />
```

2.4.5.5 Mise à jour des tags dans les jsp : attributs « escape »

Dans le tag `s:property` de Struts, l'attribut escape a été remplacé par les quatre attributs suivant :

- escapeCsv
- escapeHtml
- escapeJavaScript
- escapeXml

L'ancien tag `escape` est déprécié et remplacé par `escapeHtml`.
Par défaut, `escapeHtml` est à true. Pour plus de détail sur l'utilisation de ces nouveaux attributs, se référer au Guide du Développeur Hornet

2.4.5.6 Mise à jour des tags dans les jsp : s:form

Depuis le passage à Struts 2.3.15.1, les tags `s:form` doivent avoir une action définie, même vide.

```
<s:form action="" id="idform" >
```

Un bouton `submit` non visible peut être placé au début du formulaire. Il permet de s'assurer que l'action appelée en cliquant sur « enter » dans un champ est bien l'action par défaut. (D'autres boutons `submit` associés à d'autres actions peuvent en effet être présents dans le formulaire).

```
<s:submit id="btnValiderDefault" cssClass="none" value="Valider" name="btnValider" />
```

2.4.5.7 Mise à jour des tags dans les jsp : attributs « required »

Dans le tag `s:label` de Struts, l'attribut `required` a été remplacé par l'attribut `requiredLabel` afin d'éviter tout conflit avec l'attribut réservé par HTML5.

Exemple de création d'un formulaire comportant des champs obligatoires:

```
<s:form action="" id="idform" >

   <s:label for="id1" value="%{getText('champs1.libelle')}" requiredLabel="true"/>
   <s:textfield id="id1" name="champs1" />

   <s:label for="id2" value="%{'champs2'}" requiredLabel="true" />
   <s:datetimepicker id="id2" name="champs2" displayFormat="dd/MM/yyyy" />

   <s:submit value="Rechercher" />
   <s:reset value="Reinitialiser" />
</s:form>
```

2.4.5.8 Mise à jour du paramétrage de field-validator : paramètre « expression »

Dans l'utilisation d'un validateur de champ de type regex, le paramètre `expression` a été remplacé par `regex`.

Exemple :

```
<field-validator type="regex" >
  <param name="regex"><![CDATA[^(\+?[0-9]|\(|\)| |\.|\+\){8,25}$]]></param>
  <message key="reseaux.resTel.invalide" />
</field-validator>
```

2.4.6 Spring

2.4.6.1 Fichiers de configuration

Dans les fichiers de configuration Spring, mettre à jour les références aux XSD dans la balise `beans` pour pointer vers les versions 3.0 :

Modifier les attributs `xsi:schemaLocation` en remplaçant les références à la version 2.5 par 3.0, exemple :

```
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:aop="http://www.springframework.org/schema/jee"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xsi:schemaLocation="
http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-
3.0.xsd
http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.0.xsd
http://www.springframework.org/schema/jee http://www.springframework.org/schema/aop/spring-jee-3.0.xsd">
```

Depuis Spring 3.0, les attributs `dependency-check` et la valeur `autodetect` pour l'`autowire` sont dépréciés.

- Supprimer l'`autowire` et utiliser le tag `constructor-args` pour l'instanciation des beans par Spring.

De plus et afin de limiter au maximum les ambiguïtés lors de l'instanciation par constructeur, il est important de spécifier l'index (basés 0) des paramètres :

```
    <bean id="partenaireService"
        class="fr.gouv.diplomatie.applitutoriel.business.service.PartenaireServiceImpl">
        <constructor-arg ref="partenaireDAO" index="0" />
        <constructor-arg ref="villeService" index="1" />
    </bean>
```

L'utilisation de l'attribut « index » est à privilégier, car l'utilisation de l'attribut « name » pour lever l'ambiguïté ne fonctionne que pour du code Java compilé avec le flag « debug » activé. C'est généralement le cas en développement mais pas forcément lors de la génération du package (war, jar …) à déployer sur la plateforme cible.

2.4.6.2 Classe de test unitaire de la couche présentation

- **Remplacer** le fichier java tst/java/hornet/framework/test/BaseStrutsTestCase.java

2.4.7 Spring Security

2.4.7.1 Présentation

Spring Security permet de gérer la sécurité d'une application Java de manière transverse. Ce composant offre la possibilité d'activer l'authentification et le contrôle d'accès que ce soit au niveau de l'IHM (par le biais d'une taglib) ou de la couche service grâce à des annotations spécifiques.

2.4.7.2 Récupération des classes Javas

- Remplacer/ajouter les fichiers Java suivants :
  - src/java/fr/gouv/diplomatie/hornettemplate/integration/vo/**MenuItemVO**.java
  - src/java/fr/gouv/diplomatie/hornettemplate/business/service/**MenuServiceImpl**.java

- Supprimer (si non utilisé) :
  - src/java/fr/gouv/diplomatie/hornettemplate/web/action/frameset/**Logout**.java

2.4.7.3 Configuration

- Récupérer le fichier src/config/**spring-security.xml** et l'adapter au besoin de l'application à migrer.
- Pour l'authentification CAS, récupérer et adapter le fichier src/config/**cas.properties**.
- Compléter le fichier **menu.xml** pour préciser les rôles associés aux éléments du menu.
- Ajouter éventuellement les annotations @Secured nécessaires au niveau des classes de service.
- Ajouter éventuellement les tags <sec:authorize> dans les JSP pour gérer l'affichage/masquage des éléments de l'IHM en fonction des profils, de même que la balise générale de définition de la taglib :

```
<%@taglib uri="http://www.springframework.org/security/tags" prefix="sec" %>
```

Le Guide du développeur sert de référence pour la réalisation de ces tâches.

Le formulaire de la page de login doit maintenant pointer vers l'action Spring Security :

- Remplacer :

```
<form id="fm1" class="fm-v clearfix" action="<c:url value="/j_security_check" />" method="post">
```

Par :

```
<form id="fm1" class="fm-v clearfix" action="<c:url value="/j_spring_security_check" />" method="post">
```

2.4.8 tagDir

- Mettre à jour le fichier de propriété : « build.properties » :

```
#activation du versionning sur les ressources statiques
build.tagDirVersion=true
```

2.5 Composant IHM

2.5.1 Tableaux

La pagination et le tri des tableaux côté serveur a fait l'objet d'une refonte dans Hornet 3.1 pour permettre l'utilisation de plusieurs tableaux dans une même page (cf Guide du développeur Hornet 3).

- Supprimer :
  - src/java/fr/gouv/diplomatie/<nomprojet>/web/action/**TableServerSort.java**
- Ajouter / modifier :
  - src/java/fr/gouv/diplomatie/<nomprojet>/web/action/**Sort.java**
  - src/java/fr/gouv/diplomatie/<nomprojet>/web/action/**TableState.java**
  - src/java/fr/gouv/diplomatie/<nomprojet>/web/action/**Pagination.java**
  - src/java/fr/gouv/diplomatie/<nomprojet>/web/action/**TablesStatesMap.java**
  - src/java/fr/gouv/diplomatie/<nomprojet>/web/action/**TablesStatesAware.java**

3 Mise à jour structurelle

3.1 Pages dynamiques

3.1.1 Pages JSP

- Remplacer les JSP suivantes dans le répertoire /Webcontent/WEB-INF
  - xml-jsp/commun/error.jsp
  - xml-jsp/commun/success.jsp
  - template/excelExport.jsp
  - tiles-jsp/frameset/filariane.jsp (évolution des styles + dernier élément non cliquable)
- Pour la mise en place de Spring Security :
  - tiles-jsp/plan_site.jsp
  - tiles-jsp/plan-item.jsp
  - tiles-jsp/frameset/menuitem.jsp
  - tiles-jsp/frameset/nav.jsp
  - tiles-jsp/frameset/hd.jsp

3.1.2 Gestion du Menu

- Ajouter le fichier **src/config/menu.xsd** (permet la validation du fichier menu.xml)

> ➲ **Si des développements spécifiques ont été réalisés sur ces pages, ils seront écrasés. Les spécificités seront éventuellement à reporter dans les nouvelles pages.**

4 Mise à jour fonctionnelle

4.1 Sécurité des applications : SpringSecurity

Type : Nouvelle fonctionnalité.

Spring Security permet une sécurisation des applications

Pour en savoir plus, cf §2.4.7 ainsi que « HORNET_Guide-developpeur» , section 3.2.2.4 « Configuration des autorisations ».

4.2 Composant Arborescence

Type : Nouvelle fonctionnalité.

Le composant "Arborescence" permet d'afficher une liste imbriquée d'éléments et de sous-éléments.

Par défaut, l'arborescence est entièrement générée au chargement de la page. Dans le cas où JavaScript est activé, le chargement peut être réalisé de façon dynamique par requêtes AJAX. Cette utilisation est préférable lorsque les données sont trop volumineuses. Cependant aucune alternative n'est mise en place, JavaScript désactivé, dans le cas d'un chargement complet volumineux.

Pour en savoir plus, cf. « HORNET_Guide-developpeur», section 3.3.6.8 « Arborescence ».