TP N°1 de Java EE

-

Installation et prise en main de l'environnement de travail.

Vincent GARDEUX

24 août 2009

Table des matières

1 Installation **3**
1.1 Tomcat 3
1.2 Eclipse 4

2 Utilisation de la plateforme **5**
2.1 Création d'un Projet Web 5
2.2 Ma première Servlet 6

3 Intégration de nouvelles library : EL et balises JSTL **10**

1 Installation

L'installation de l'environnement de travail nécessite 2 outils principaux :

- **Apache Tomcat** qui permet de créer un serveur HTTP afin d'héberger nos futures servlets/JSP et de les tester en local avant de les publier éventuellement sur un serveur Web.
- **Eclipse Europa pour Java EE** qui est une version spécifique de l'environnement de développement Eclipse, permettant de créer des projets Web en utilisant par exemple un serveur Apache.

1.1 Tomcat

Commencez par installer Tomcat. Vous pouvez, lors de l'installation, cocher la case "Exemples" qui vous permet d'avoir en + quelques premières Servlet et JSP(recommandé pour débuter). Vous remarquerez que son installation nécessite d'avoir préalablement installé un JRE sur votre machine. Si ce n'est pas le cas, installez le dernier JDK disponible sur le site de sun.

Remarque importante : Le serveur Tomcat se lance par défaut sur le port 8080. Si une autre application (type Oracle) est déjà lancée sur ce port, votre serveur ne pourra pas démarrer. Pour pallier à celà, 2 solutions sont possible : changer le port de Tomcat (ex : 8081) ou bien stopper toutes les applications qui utilisent ce port avant de lancer votre serveur.

Une fois l'installation terminée, lancez Apache Tomcat.
Pour celà, sur Windows il suffit de lancer "Monitor Tomcat" du menu Démarrer, ou bien lancer *tomcat6w.exe* présent dans le répertoire bin.
Sur UNIX il faut lancer le script *startup.sh* pour démarrer le serveur et *shutdown.sh* pour l'arreter.

Pour vérifier qu'il est correctement installé, une fois lancé, ouvrez un navigateur Web et allez à l'url :

`http://127.0.0.1:8080/ ( ou http://localhost:8080 )`

Vous devriez voir apparaître une page de présentation Apache Tomcat.
Si vous les avez installés, des exemples de Servlets et de JSP sont disponibles sur :

`http://127.0.0.1:8080/examples (ou http://localhost:8080/examples)`

Toutes ces pages Web sont stockées dans votre répertoire d'installation de Tomcat dans le dossier webapps. Consultez ce dossier pour explorer son arborescence.

Vous pouvez rajouter à la main de nouvelles applications dans le dossier

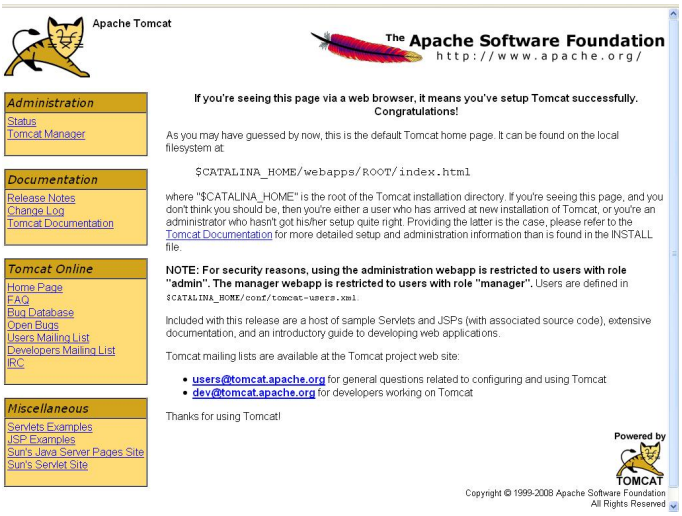

FIGURE 1 – Page de présentation Apache

webapps mais cette opération n'est pas triviale. L'arborescence d'une application Java EE doit contenir un certain nombre de fichier et de dossiers spécifiques tels que le répertoire **WEB-INF**, le fichier de configuration **web.xml**, ...
L'utilisation d'Eclipse nous permet une création bien plus simple de projets WEB, car il créera lui-même les fichiers et dossiers nécessaires. Il y ajoutera une configuration de base qui pourra ensuite être modifiée à votre gré.

1.2 Eclipse

Installez Eclipse pour Java EE. (Il suffit de décompresser l'archive) ou bien mettez à jour votre ancienne version en téléchargeant le plugin "Java EE Developpers". Dans tous les cas, si vous possédiez des fichiers dans votre workspace Eclipse, sauvegardez-les afin de ne pas les perdre.

Puis lancez Eclipse.

2 Utilisation de la plateforme

2.1 Création d'un Projet Web

Sous Eclipse sélectionnez New Project>Web>Dynamic Web Project.

Lorsque vous créez votre projet Web, Eclipse vous demande comment vous voulez l'exécuter, vous devez donc spécifier que vous utilisez un serveur Apache.

Pour celà, dans l'onglet Target Runtime, cliquez sur New...

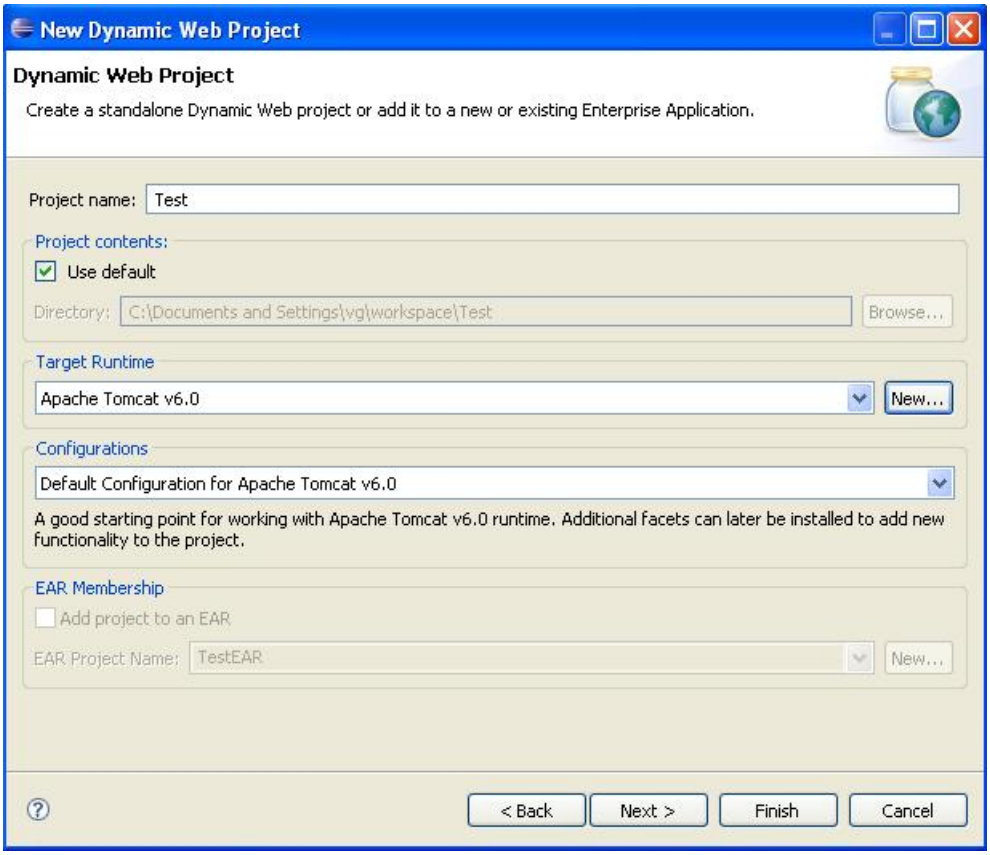

FIGURE 2 – Création d'un nouveau projet Web

Puis sélectionnez votre serveur HTTP : Apache v6.0, ainsi que son emplacement d'installation.

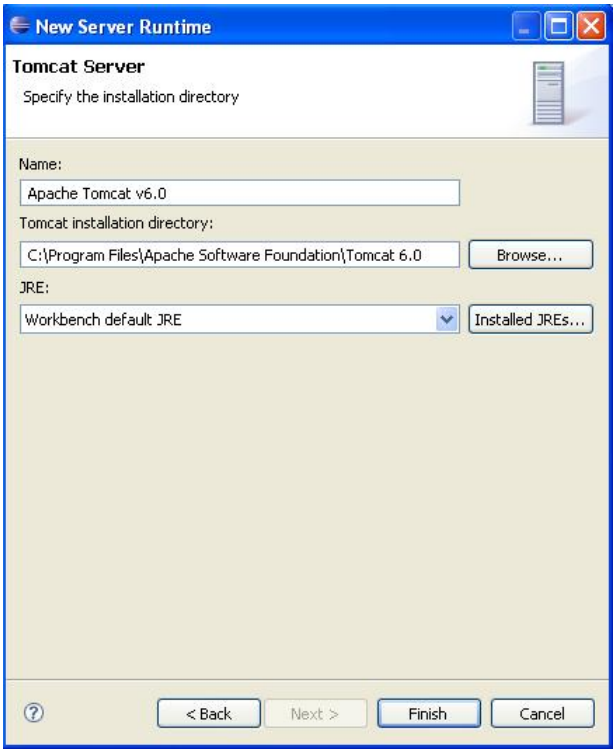

FIGURE 3 – Sélection du serveur Apache

Félicitations ! Vous venez de créer votre premier projet Java EE.

2.2 Ma première Servlet

Pour vérifier que votre installation fonctionne correctement, nous allons créer une petite Servlet de test.

Faites un clic droit sur votre projet puis New>Servlet

Ceci va permettre de générer automatiquement le code nécessaire à la création d'une Servlet. Eclipse nous mâche vraiment le travail !

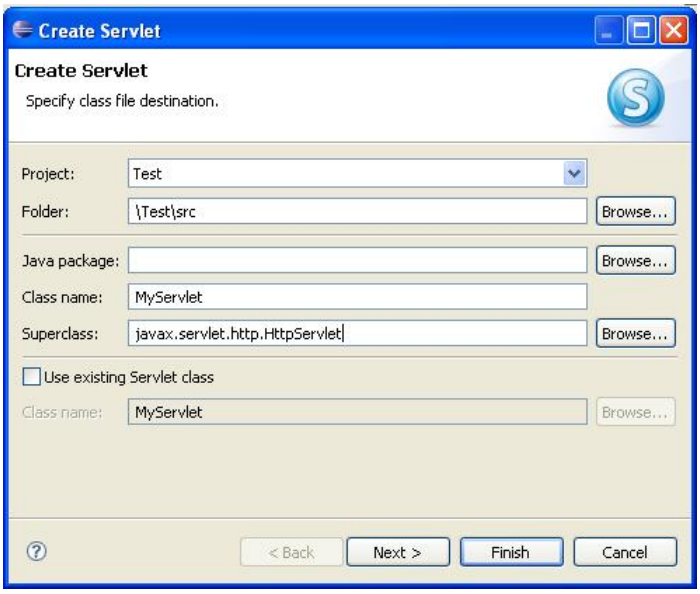

FIGURE 4 – Configuration de la Servlet

Vous n'avez qu'à nommer votre Servlet. Vous remarquez également que votre Servlet hérite de

`javax.servlet.http.HttpServlet`

Ceci montre que votre Servlet est une Servlet Http, qui ne pourra donc échanger des informations que dans ce protocole précis.

Nous pouvons par ailleurs préciser à ce moment un certain nombre de paramètres d'initialisation mais nous ne nous en servirons pas pour ce premier TP.

Lorsque vous cliquez sur Finish, 2 fichiers sont créés :

- **src/MyServlet.java** qui est le fichier java dans lequel vous allez créer le code de la Servlet.
- **build/classes/MyServlet.class** qui est le fichier java compilé (il se compile automatiquement à chaque modification du fichier java)

et un fichier est modifié :

- **WebContent/WEB-INF/web.xml** qui est le fichier de configuration des Servlet et dans lequel les paramètres initiaux de notre nouvelle Servlet viennent d'être ajoutés.

Le dossier **WEB-INF** est obligatoire dans un projet Java EE et doit posséder exactement cette syntaxe. Par ailleurs, ce dossier n'est pas accessible par le WEB. Les usagers de votre site ne pourront donc pas lister son contenu.

Maintenant que tout est initialisé, nous allons créer une Servlet qui générera une page HTML permettant d'afficher un classique "Hello World". Pour celà, il suffit d'écrire votre code dans la méthode doGet().

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    out.println("<HTML>");
    out.println("<HEAD><TITLE> Titre </TITLE></HEAD>");
    out.println("<BODY>");
    out.println("Hello World");
    out.println("</BODY>");
    out.println("</HTML>");
    out.close();
}
```

Pour lancer cette Servlet sur le serveur Apache, il suffit de faire un clic droit sur le fichier java de la Servlet : Run As>Run On Server.

Si tout se passe bien, Eclipse va démarrer le serveur Apache et vous verrez apparaître la page Web dans le navigateur de Eclipse.

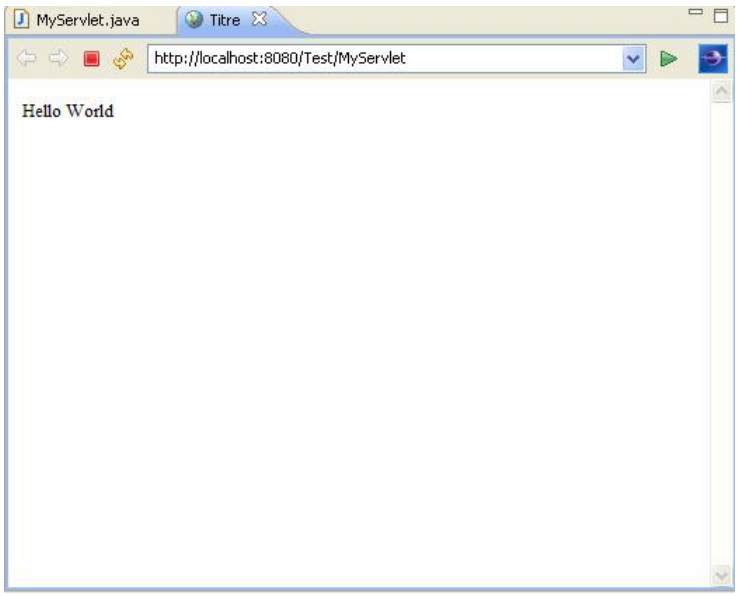

FIGURE 5 – Affichage du HelloWorld

Remarque importante : Si le serveur est déjà démarré par une autre application, Eclipse vous demande de préalablement l'arrêter afin qu'il puisse y

avoir accès. Donc si votre serveur Tomcat est déjà lancé en dehors d'Eclipse, arrêtez-le.

Vous remarquerez que le navigateur d'Eclipse génère automatiquement l'URL correspondant à votre Servlet.
Si vous entrez cette URL dans un navigateur classique, il affichera également la page générée par votre serveur.
L'utilisation d'Eclipse est uniquement une aide au développement de Java EE.

3 Intégration de nouvelles library : EL et balises JSTL

Votre plateforme permet maintenant la création de Servlets et de feuilles JSP. Cependant, nous aurons besoin plus tard d'utiliser les EL et les balises JSTL. Pour terminer l'installation de notre environnement, nous allons intégrer ces fonctionnalités.

Pour celà, si vous avez installé les examples, dans le répertoire d'installation de tomcat, il suffit de copier les 2 fichiers jar situés dans :

`$RépertoireInstallationTomcat$/webapps/examples/WEB-INF/lib/`

à l'emplacement :

`$RépertoireInstallationTomcat$/lib/`

Sinon récupérez les fichers **jstl.jar** et **standard.jar** et copiez les à l'emplacement $RépertoireInstallationTomcat$/lib/

Cette manipulation ajoute les *.jar* directement dans les librairies utilisées par notre serveur tomcat. Ainsi, elles seront disponibles par la suite dans n'importe quel projet.
Si vous souhaitez les utiliser uniquement dans certains projet, plutôt que de les mettre dans le dossier *lib* de tomcat, il suffira de l'ajouter dans le dossier *lib* de ce projet :

`$RépertoireProjetJEE$/WebContent/WEB-INF/lib/`

Pour tester que le serveur reconnaisse bien la syntaxe des EL et des balises JSTL, nous allons créer une petite JSP et la lancer sur le serveur de la même façon que nous avons créer une Servlet mais cette fois-ci en faisant : New>JSP.

```
<%@ page language="java"
    contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions" %>
<html>
<head>
    <title>Test</title>
</head>
<body>
```

```
    <c:set var="chaine">Hello World</c:set>
    ${fn:length(chaine)}
</body>
</html>
```