TP1 Servlets
23/03/2010 C.Plumejeaud / P.Reignier

1. Configurer tomcat

Tomcat se trouve dans C:/ProgramFiles/ApacheSotfwareFoundation

_ Editez le fichier server.xml du répertoire conf. Ce fichier contient un certain nombre de paramètres de configuration dont :
– le numéro de port utilisé pour envoyer le message d'arrêt au serveur :
<Server port='8005' shutdown=...>
– le numéro de port écouté pour les requêtes http :
<Connector port='8085' ...
_ Modifiez ces deux numéros de ports.
_ Allez dans le répertoire bin. Le serveur est lancé grâce au script startup.sh. Il est arrêté par le script shutdown.sh. Ou bien le service windows Tomcat

Une fois le serveur lancé, vous pouvez vérifier qu'il fonctionne correctement en vous connectant sur sa page d'accueil : http://machine:port/. L'utilisation de la partie manager (liste des applications déployées + upload de fichiers war + . . .) nécessite un login et un mot de passe. Ce login et mot de passe sont à fournir dans le fichier tomcat-users.xml du répertoire conf. Le login doit jouer le rôle manager. Exemple :

```
<?xml version='1.0' encoding='utf-8'?>
<tomcat-users>
<role rolename="manager"/>
<role rolename="standard"/>
<user username="admin" password="admin" roles="standard,manager"/>
</tomcat-users>
```

_ Complétez le fichier tomcat-user.xml
_ Arrêtez et relancez le serveur.
_ Connectez vous sur l'application manager

2. ANT

2.1. Principes

La compilation de vos servlets et l'installation des différents fichiers dans l'arborescence du serveur va être réalisée par un système équivalent au Makefile : ant développé dans le cadre du projet Apache. Le système make - Makefile permet d'automatiser la compilation et l'installation de programmes au travers des règles de dépendance. Ces règles font en particulier appel au compilateur mais également à des ordres systèmes pour la manipulation des fichiers : renommage, copie, suppression . . .
Ces outils systèmes ne portent pas les mêmes noms suivant les plates-formes : cp pour Unix, copy pour MSDOS. Ceci conduit donc à une mauvaise portabilité de l'ensemble et à la réécriture du Makefile selon l'environnement. Le système ant est une réponse à ce problème. Basé sur Java, les incompatibilités entre les systèmes sont directement gérées au niveau de l'outil et plus au niveau du fichier de description (Makefile). Ant prend en charge les règles de dépendances ainsi que toutes les manipulations classiques de fichiers y compris l'envoie par ftp sur un serveur distant.
Le fichier de description est un fichier XML : build.xml.
Pour tout savoir sur ANT, allez sur cette URL: http://ant.apache.org/manual/index.html

Ouvrez ce fichier et recherchez la balise <target name="compile"... Cette tâche dépend de la tâche “prepare”. Ant vérifiera donc d’abord si les fichiers gérés par “prepare” sont à jour où si certaines actions doivent être exécutées. La tâche “compile” consiste en la création d’un répertoire (balise mkdir). Ce répertoire ne sera créé que s’il n’existe pas. Les fichiers sources java sont ensuite compilés à destination de ce répertoire.

2.2 Ajouter des tâches

Ant permet donc de prendre en charge les tâches courantes pour la mise en oeuvre d’un projet : création de répertoires, copies de fichiers, compilation de code Java etc. Ant permet également de rajouter de nouvelles tâches propres à un projet particulier. Cet ajout est réalisé par des fichiers jar. Tomcat propose dans ce cadre de nouvelles tâche permettant :
_ de déployer une nouvelle application.
_ de la recharger (en cas de modification) sans avoir besoin d’arrêter et de redémarrer Tomcat.
_ de la supprimer.

Ces différentes tâches sont celles que l’on retrouve dans l’application manager (ant utilise le manager) et que l’on peut donc réaliser directement depuis ant sans avoir à utiliser l’interface web.
Afin que ant puisse utiliser ces nouvelles tâches, il est nécessaire de lui fournir le fichier jar les implémentant. Ce fichier jar est le fichier catalina-ant.jar présent dans le répertoire /lib de votre installation tomcat. Ce fichier doit être ajouté à votre CLASSPATH.

Sous Unix

Ajouter le fichier catalina-ant.jar à votre variable CLASSPATH :
setenv CLASSPATH ${CLASSPATH}:chemin de homedir:/...../catalina-ant.jar
Remarque : afin de ne pas retaper cette instruction dans chaque nouvelle fenêtre, elle peut être ajoutée au contenu de votre fichier .cshrc

Sous Windows

Vous allez compiler avec ant depuis eclipse. Donc à priori il faut configurer le classpath du projet sous Eclipse pour avoir catalina-ant.jar. Cependant, pour une raison incomprise, ceci ne fonctionne pas. Donc vous devez rajouter dans le build.xml l’instruction suivante :

```
<path id="tomcat.classpath">
        <fileset dir="${catalina.home}/lib" includes="catalina-ant.jar" />
        <fileset dir="${catalina.home}/lib" includes="catalina.jar" />
</path>
```

Et ensuite rédéfinir les tâches de déploiement ainsi :

```
<taskdef name="deploy" classname="org.apache.catalina.ant.DeployTask">
        <classpath refid="tomcat.classpath" />
</taskdef>
<taskdef name="list" classname="org.apache.catalina.ant.ListTask">
        <classpath refid="tomcat.classpath" />
</taskdef>
<taskdef name="reload" classname="org.apache.catalina.ant.ReloadTask">
        <classpath refid="tomcat.classpath" />
</taskdef>
<taskdef name="undeploy" classname="org.apache.catalina.ant.UndeployTask">
        <classpath refid="tomcat.classpath" />
</taskdef>
```

2.3 Property

La balise javac contient le chemin du répertoire source. Ce chemin peut soit être “codé en dur”, soit être mis dans une variable interne au fichier build.xml (cf les variables des Makefile). Dans le cas présent, une variable src.home a été utilisée :

```
srcdir="${src.home}"
```

Les variables dans les fichiers build.xml s’appellent des property. En tête du fichier, vous trouverez par exemple :

```
<property name="manager.url" value="http://localhost:8080/manager"/>
```

Il existe également un mécanisme (l’équivalent de l’include du C ou des Makefiles) permettant de définir ces variables à l’extérieur du fichier build.xml. Ceci permet d’isoler des variables de configuration dans un fichier annexe plus simple.

Le fichier buid.properties du répertoire courant a pour vocation de contenir des informations propres à l’application. Il s’agit en particulier du contexte. Rappel : le contexte est l’URL permettant de désigner votre application. Lors du déploiement, ant donnera au manager de tomcat ce contexte ainsi que le répertoire où se trouve votre arborescence binaire. Ce fichier sera construit à la section première servlet.

Contenu du fichier build.properties :

```
app.path=/monContexte
```

Le fichier build.properties contient des informations plus globales correspondant
à votre environnement de travail. Il s’agit de l’URL, du login et mot de passe du manager ainsi que du chemin d’installation de tomcat. Contenu de ce fichier :

```
# Tomcat 5 installation directory
catalina.home=/usr/java/jakarta-tomcat-5.0.28
# Manager webapp username and password
manager.username=admin
manager.password=admin01
manager.url=http://localhost:<port>/manager/
```

Sous Windows

Vous devez mettre à jour votre fichier **build.properties**, celui qui est placé dans votre projet de développement, notamment pour **catalina.home** et **manager.***.

```
# Last change: $Date: 2010-03-04 17:51:19 +0100 (Jeu, 04 mar 2010) $ by $Author: plumejea $
# $Id$

# src
# directory where java sources are located
#catalina.home=/Applications/apache-tomcat-6.0.20
catalina.home=C:\\Tools\\apache-tomcat-6.0.26

manager.url=http://localhost:1977/manager
manager.username=admin
manager.password=admin01

# application: its name
app.name=TPServlet

# application : its version
app.version=V01-7Mars2010
```

2.4 Build.xml

Le fichier build.xml que vous venez de manipuler est celui proposé par tomcat. Il est suffisamment générique pour pouvoir être utilisé tel quel dans le développement de vos applications WEB. Il sait en particulier retrouver dans vos répertoire les classes java, les fichiers html et le descripteur de déploiement, pour peu que vous ayez respecté l'arborescence source vue en cours. Il sait également recréer une arborescence binaire conforme à la norme. Cette arborescence se trouve après compilation dans le répertoire build
Les tâches proposées par ce build.xml sont :

```
> ant -projecthelp
        Buildfile: build.xml
        Main targets:
                all Clean build and dist directories, then compile
                clean Delete old build and dist directories
                compile Compile Java sources
                dist Create binary distribution (war file)
                install Install application to servlet container
                javadoc Create Javadoc API documentation
                list List installed applications on servlet container
                reload Reload application on servlet container
                remove Remove application on servlet container
        Default target: compile
```

Remarque : ant install permet de déployer une application web lorsque celle-ci n'est pas encore présente. ant reload permet de redéployer une application déjà présente sur le container.

3. Votre développement

Vous allez partir du squelette de projet que vous avez reçu par mail et l'importer sous eclipse.

0. Vérifier que java fonctionne : sous « command » , tapez 'java –version'. Ceci renvoie normalement un message du type suivant :

```
java version "1.6.0_15"
Java(TM) SE Runtime Environment (build 1.6.0_15-b03-226)
Java HotSpot(TM) 64-Bit Server VM (build 14.1-b02-92, mixed mode)
```

Si ce n'est pas le cas, télécharger java (http://java.sun.com/javase/downloads/index.jsp). Sous Windows, et en tant qu'administrateur, vous devez rajouter JAVA_HOME comme variable d'environnement et mettre à jour le Path du Système.

1. Lancer éclipse et choisir un workspace pour éclipse (celui-ci doit être dans une zone « écrivable » de votre machine), donc votre répertoire sur Z:/ peut convenir.

2. Créer un projet par import de source (Cf Figure1) et sélectionner les sources à importer (Cf Figure 2)

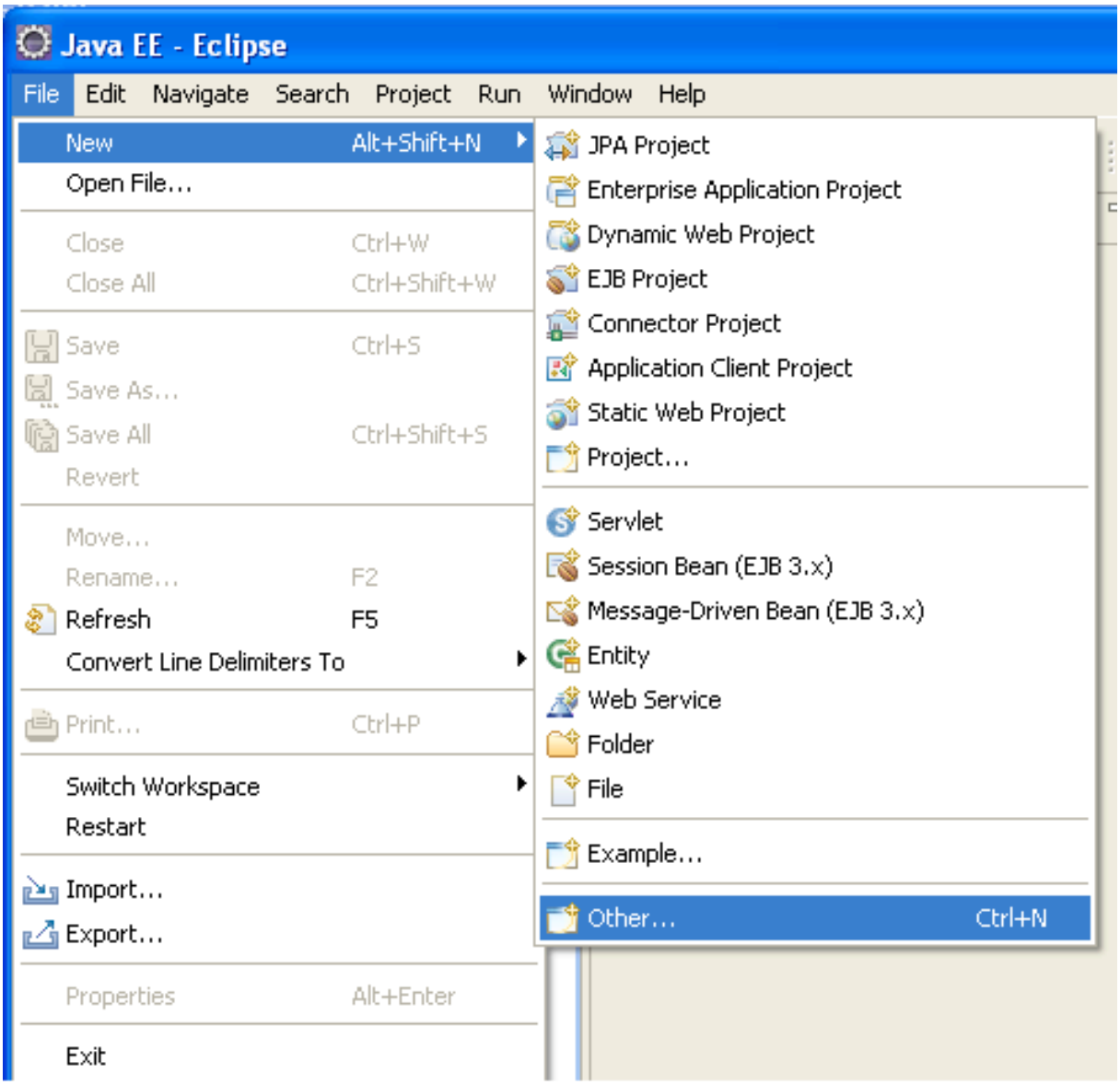

Figure 1. Créer un projet java par import

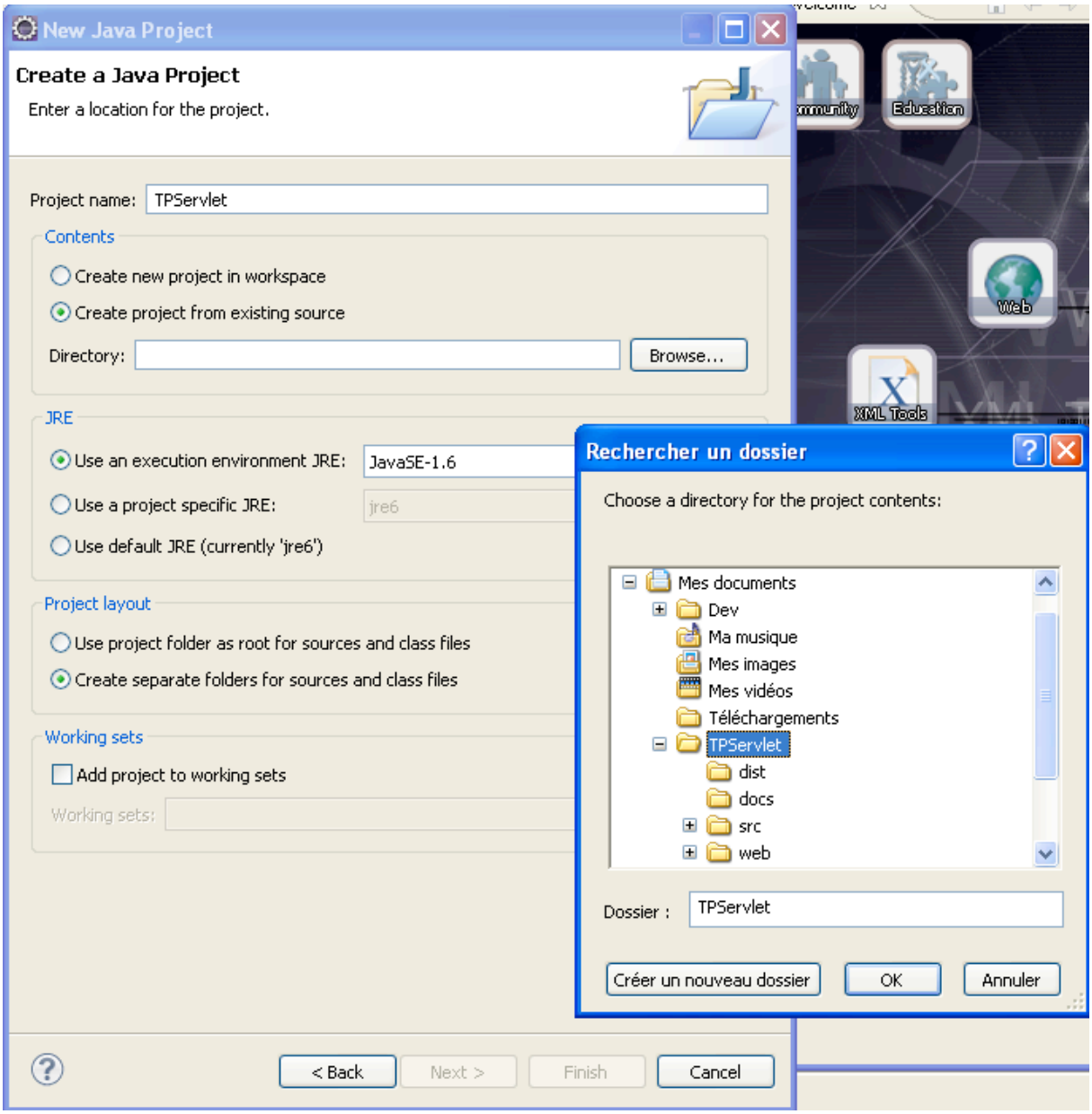

Figure 2. Importer des sources pour créer un projet java

3. Configurer les classpath de votre projet sous eclipse : il faut rajouter les bibliothèques de tomcat en cliquant sur Add External JARs et en choisissant tout ce qui se trouve sous ${tomcat.home}/lib.

Note : cette opération peut se faire ultérieurement en sélectionnant les « properties » du projet « TPServlet », dans le menu contextuel (clic droit) associé au projet.
Lorsque vous choisissez Add JARs, vous pouvez ajouter des bibliothèques (.jar) présent dans le projet que vous venez de créer. Add External JARs vous permet d'aller récupérer des bibliothèques externes au projet. Par exemple, acme.jar et log4J-1.2.14.jar sont des bibliothèques du projet rajoutées par Add JARs.

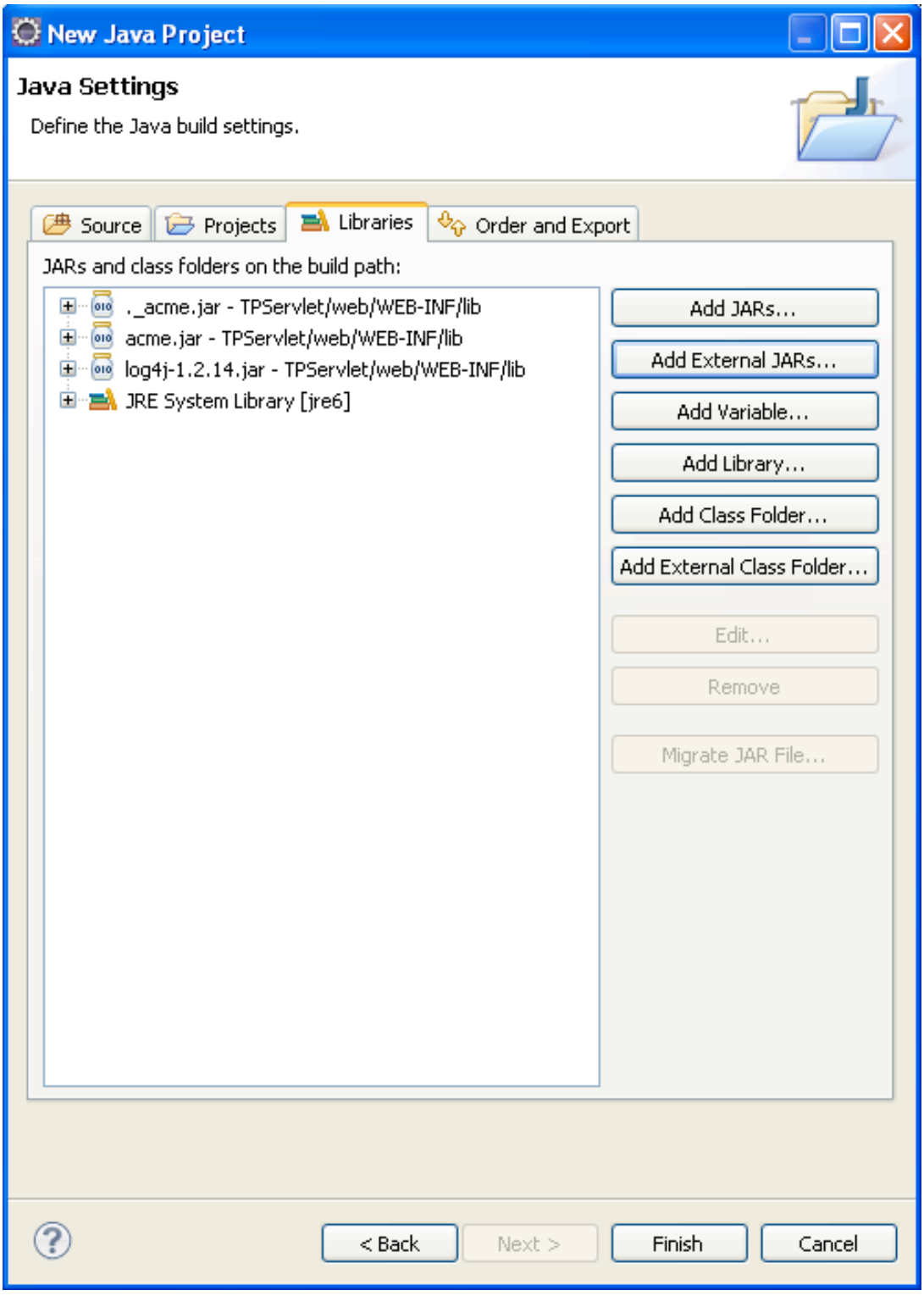

Figure 3. Configurer un classpath de projet

4. Configurer ANT pour votre projet sous eclipse : d'abord ajouter la vue « Ant » au projet (cf Figure 4) puis ajouter votre fichier build.xml comme source à interpréer pour ANT (cf Figure 5).

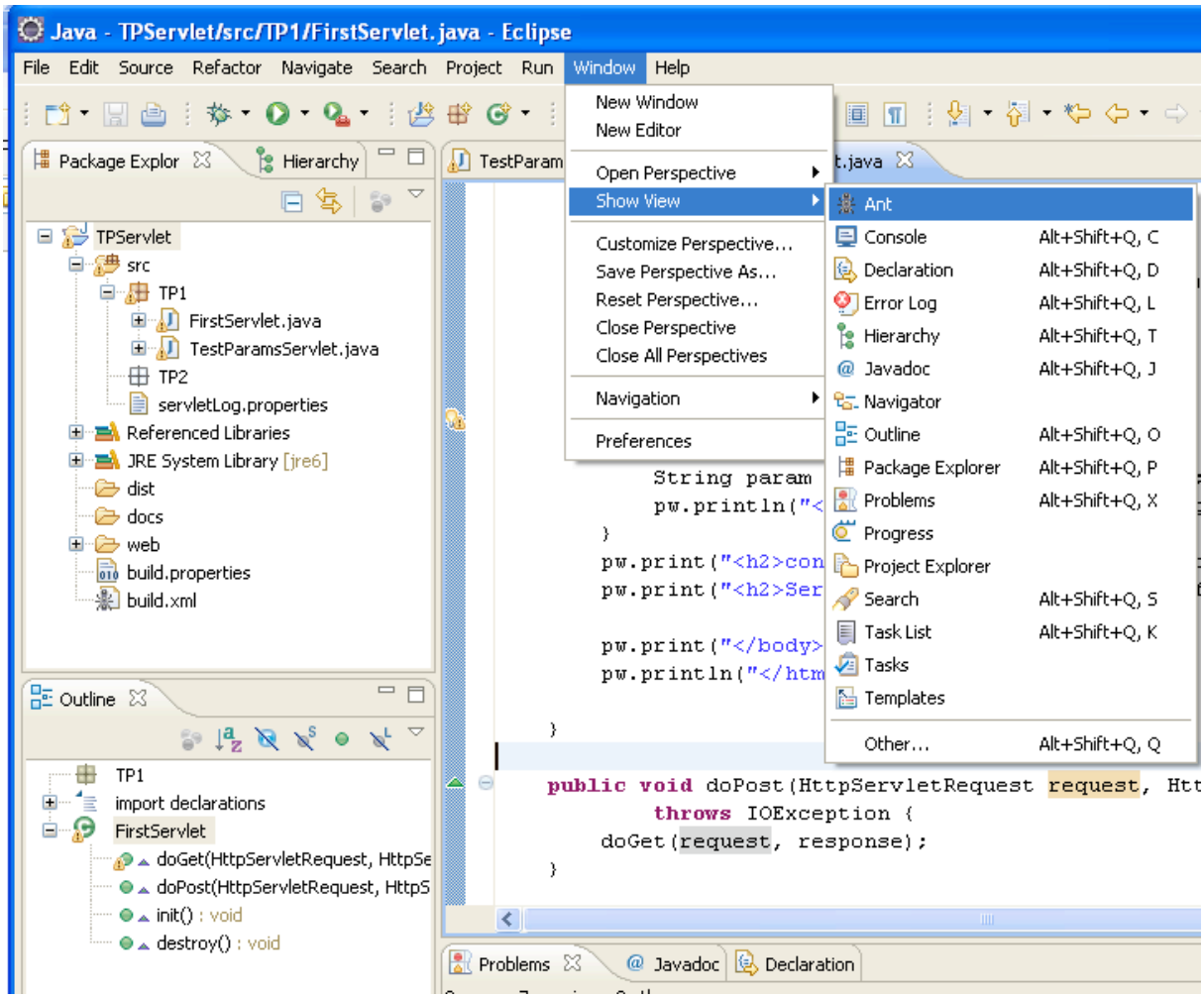

Figure 4. Ajouter une vue ant

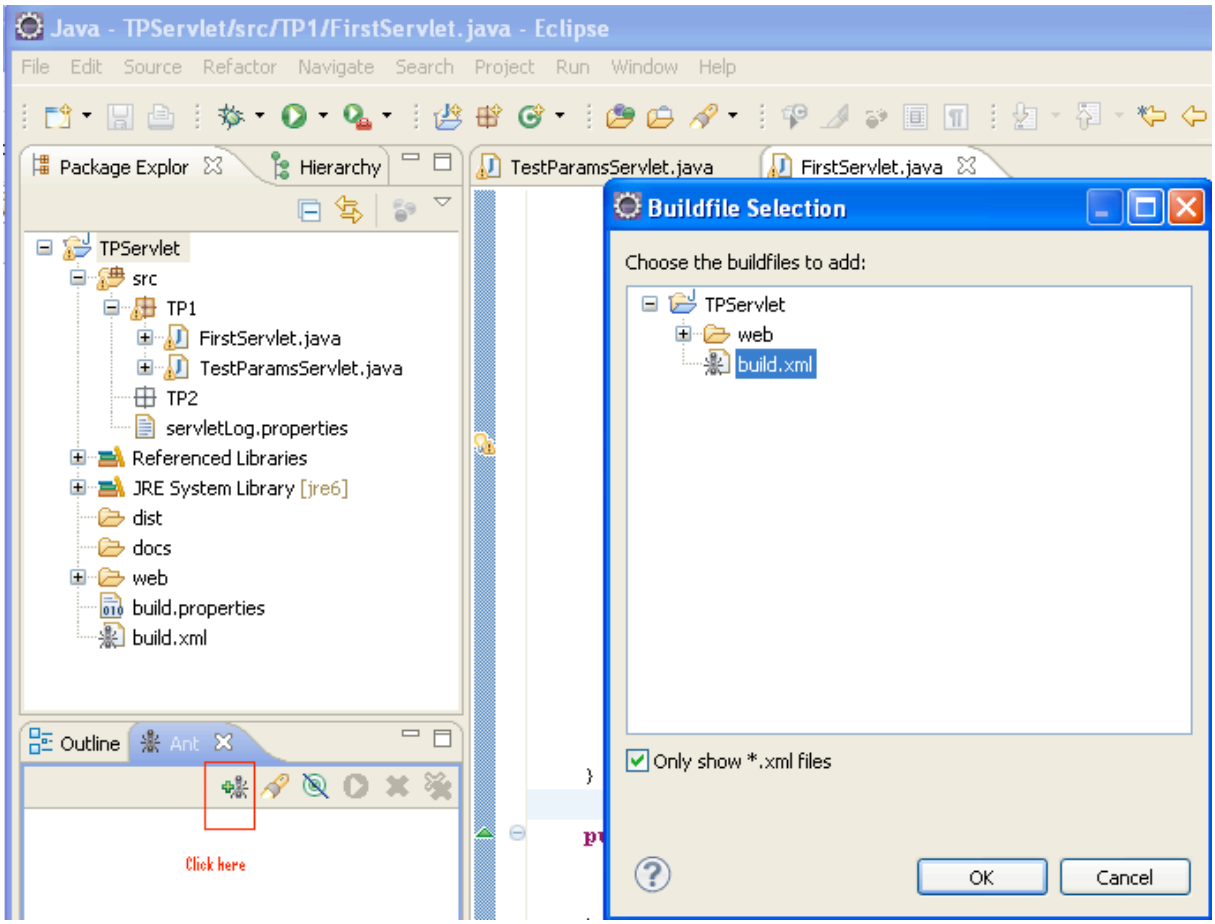

Figure 5. Spécifier le fichier build.xml à interpréter pour ant

5. Normalement, vous pouvez utiliser votre projet en l'état, si vous réalisez la mise à jour du fichier build.xml et build.properties comme recommandé plus haut.

Le projet contient :
_ le répertoire **src/TP[n]** : ce répertoire contiendra les sources Java de vos servlets
_ le répertoire **web/** : ce répertoire contient les pages html, jsp, et les images.
_ le répertoire **web/WEB-INF**. Ce répertoire contient le descripteur de déploiement web.xml
_le répertoire **web/WEB-INF/lib**. Ce répertoire contient les jars additionnels pour l'application. Vous y placerez les jar dont vous aurez besoin au fur et à mesure.

3.1 Factorielle

- Mettre en place le formulaire index.html. Ce formulaire doit envoyer ces données vers l'URL calcule. Ce fichier doit être placé dans le répertoire web.
- Programmez la servlet Factorielle. Cette factorielle doit être associée à l'URL calcule dans le descripteur de déploiement. Le fichier Factorielle.java doit être placé dans le répertoire src/TP1
- Afin de pouvoir compiler, utilisez le fichiers build.xml et build.properties vus précédemment. Rappel : le fichier build.properties situé dans le même répertoire que build.xml contient le contexte de l'application : `app.path=/TPServlet`

3.2 Compteur

Programmez la servlet Compteur.

- Cette servet doit être associée à l'URL visites dans le descripteur de déploiement.

- Le fichier Compteur.java doit être placé dans le répertoire src/TP1

3.3 Lecture d'une base de données

Pre-requis : avoir installé la base de données « insee_cog »

Programmez la servlet JDBCServlet.

- Les paramètres de connection à la base de données (URL, login et password) sont dans des paramètres du context de la webapp.
- Elle exécute la requète suivante (`select * from regions`), et affiche les résultats sous forme tabulaire.
- Cette servet doit être associée à l'URL readDB.form dans le descripteur de déploiement.
- Le fichier JDBCServlet.java doit être placé dans le répertoire src/TP2