Développons en Java avec Eclipse

Jean Michel DOUDOUX

Table des matières

Table des matières

Table des matières

Table des matières

Table des matières

Table des matières

Table des matières

Développons en Java avec Eclipse

Version 0.60

du 26/06/2005

par Jean−Michel DOUDOUX

Préambule

A propos de ce document

Ce document fait suite à mon premier didacticiel "Développons en Java". C'est un didacticiel qui se propose de fournir des informations pratiques sur la mise en oeuvre et l'utilisation d'Eclipse et de quelques un de ces nombreux plug−ins.

Celui−ci est composé de six grandes parties :

1. les bases pour l'utilisation d'Eclipse
2. le développement en Java
3. les fonctions avancées d'Eclipse
4. le développement avancé avec Java
5. d'autres plug−ins

Chacune de ces parties est composée de plusieurs chapitres dont voici la liste complète :

- Préambule
- Introduction
- Installation et exécution d'Eclipse
- Le plan de travail (Workbench)
- L'espace de travail (Workspace)
- Les fonctions pratiques du workbench d'Eclipse
- Le Java development tooling (JDT) d'Eclipse
- Deboguer du code java
- Le refactoring
- Ant et Eclipse
- JUnit et Eclipse
- L'aide dans Eclipse
- CVS 2.0 et Eclipse 2.1
- CVS 2.5 et Eclipse 3.0
- La gestion de la plate−forme
- Des plug−ins pour le développement avec Java
- Le développement J2EE
- Java Server Faces et Eclipse
- JAXB et Eclipse
- Struts et Eclipse
- Le développement d'interfaces graphiques
- Eclipse WebTools Platform
- Le développement sans Java
- Eclipse et UML
- Eclipse et les bases de données
- Eclipse et Hibernate
- Eclipse et J2ME
- Annexes

Les sections qui concernent des plug−ins n'ont pas pour vocation d'être une documentation complète sur l'utilisation de ces plug−ins mais simplement de fournir les bases pour les installer et mettre en oeuvre un minimum de fonctionnalités qu'ils proposent.

Je suis ouvert à toutes réactions ou suggestions concernant ce document notamment le signalement des erreurs, les points à éclaircir, les sujets à ajouter, etc. ... N'hésitez pas à me contacter : jean−michel.doudoux@wanadoo.fr

Ce document est disponible aux formats HTML et PDF à l'adresse suivante : http://perso.wanadoo.fr/jm.doudoux/java/

Ce manuel est fourni en l'état, sans aucune garantie. L'auteur ne peut être tenu pour responsable des éventuels dommages causés par l'utilisation des informations fournies dans ce document.

La version pdf de ce document est réalisée grâce à l'outil HTMLDOC 1.8.23 de la société Easy Software Products. Cet excellent outil freeware peut être téléchargé à l'adresse : http://www.easysw.com

Note de licence

Marques déposées

Historique des versions

Version	Date	Evolutions
0.10 béta	08/04/2003	1ere version diffusée sur le web.
0.20	13/07/2003	Ajout des chapitres Junit, Ant, Aide, Déboguer du code Java Ajout des sections : importation, exportation, génération javadoc, informations sur les plug−ins, le plug−in Jalopy Compléments ajoutés au chapitre JDT Application d'une feuille de styles CSS pour la version HTML Corrections et ajouts divers
0.30	04/01/2004	Ajouts dans les chapitres "JDT", "deboguer du code Java" et "La gestion de la plate−forme" Ajout du chapitre "le refactoring" , "le développement sans java" et "Le développement d'interfaces graphiques"

		Réduction de la taille des images : réduction de la taille et passage en niveau de gris pour la version PDF Corrections et ajouts divers
0.40	24/05/2004	Ajout des chapitres "Eclipse et les bases de données", "le développement J2EE" et "JAXB et Eclipse" Ajouts dans le chapitre "Refactoring" et dans la section "Eclipse VE" Corrections et ajouts divers
0.50 0.50.1	07/12/2004 11/12/2004	Prise en compte d'Eclipse 3.0 dans différents chapitres Ajout des chapitres : "Eclipse et UML", "Java Server Faces et Eclipse" et "Struts et Eclipse" Ajout d'une section concacrée au plug−in "Log4E" Remise en page complète pour éviter autant que possible les blancs en bas de pages. Découpage en 5 parties Corrections et ajouts divers
0.60	26/06/2005	Ajout des chapitres "Eclipse Webtools Platform", "Eclipse et Hibernate", "Eclipse et J2ME" et "CVS 2.5 et Eclipse 3.0" Corrections et ajouts divers

Partie 1 : les bases pour l'utilisation d'Eclipse

Cette première partie est chargée de présenter les bases de l'utilisation d'Eclipse.

Elle comporte les chapitres suivants :

- Introduction : présentation générale d'Eclipse
- Installation et exécution : détaille l'installation et l'exécution des trois versions majeurs d'Eclipse sous Windows et Linux
- Le plan de travail (Workbench) : présente le plan de travail qui fournit les élements de l'ergonomie notamment au travers des perspectives, des vues et des éditeurs
- L'espace de travail (Workspace) : présente l'espace de travail qui stocke les projets et leur contenu et détaille des opérations de base sur les élément de l'espace de travail
- Les fonctions pratiques du plan de travail : détaille certaines fonctions avancées du plan de travail : la recherche, la gestion des tâches et des signets et la comparaison d'éléments

1. Introduction

Eclipse est un environnement de développement intégré (Integrated Development Environment) dont le but est de fournir une plate−forme modulaire pour permettre de réaliser des développements informatiques.

I.B.M. est à l'origine du développement d'Eclipse qui est d'ailleurs toujours le coeur de son outil Websphere Studio Workbench (WSW), lui même à la base de la famille des derniers outils de développement en Java d'I.B.M. Tout le code d'Eclipse a été donné à la communauté par I.B.M afin de poursuivre son développement.

Eclipse utilise énormément le concept de modules nommés "plug−ins" dans son architecture. D'ailleurs, hormis le noyau de la plate−forme nommé "Runtime", tout le reste de la plate−forme est développé sous la forme de plug−ins. Ce concept permet de fournir un mécanisme pour l'extension de la plate−forme et ainsi fournir la possiblité à des tiers de développer des fonctionnalités qui ne sont pas fournies en standard par Eclipse.

Les principaux modules fournis en standard avec Eclipse concernent Java mais des modules sont en cours de développement pour d'autres langages notamment C++, Cobol, mais aussi pour d'autres aspects du développement (base de données, conception avec UML, ...). Ils sont tous développés en Java soit par le projet Eclipse soit par des tiers commerciaux ou en open source.

Les modules agissent sur des fichiers qui sont inclus dans l'espace de travail (Workspace). L'espace de travail regroupe les projets qui contiennent une arborescence de fichiers.

Bien que développé en Java, les performances à l'exécution d'Eclipse sont très bonnes car il n'utilise pas Swing pour l'interface homme−machine mais un toolkit particulier nommé SWT associé à la bibliothèque JFace. SWT (Standard Widget Toolkit) est développé en Java par IBM en utilisant au maximum les composants natifs fournis par le système d'exploitation sous jacent. JFace utilise SWT et propose une API pour faciliter le développement d'interfaces graphiques.

Eclipse ne peut donc fonctionner que sur les plate−formes pour lesquelles SWT a été porté. Ainsi, Eclipse 1.0 fonctionne uniquement sur les plate−formes Windows 98/NT/2000/XP et Linux.

SWT et JFace sont utilisés par Eclipse pour développer le plan de travail (Workbench) qui organise la structure de la plate−forme et les intéractions entre les outils et l'utilisateur. Cette structure repose sur trois concepts : la perspective, la vue et l'éditeur. La perspective regoupe des vues et des éditeurs pour offrir une vision particulière des développements. En standard, Eclipse propose huit perspectives.

Les vues permettent de visualiser et de sélectionner des éléments. Les éditeurs permettent de visualiser et de modifier le contenu d'un élement de l'espace de travail.

1.1. Les points forts d'Eclipse

Eclipse possède de nombreux points forts qui sont à l'origine de son énorme succès dont les principaux sont :

- Une plate−forme ouverte pour le développement d'applications et extensible grâce à un mécanisme de plug−ins
- Plusieurs versions d'un même plug−in peuvent cohabiter sur une même plate−forme.
- Un support multi langage grâce à des plug−ins dédiés : Cobol, C, PHP, C# , ...
- Support de plusieurs plate−formes d'exécution : Windows, Linux, Mac OS X, ...
- Malgré son écriture en Java, Eclipse est très rapide à l'exécution grâce à l'utilisation de la bibliothèque SWT
- Les nombreuses fonctionnalités de développement proposées par le JDT (refactoring très puissant, complétion de code, nombreux assistants, ...)
- Une ergonomie entièrement configurable qui propose selon les activités à réaliser différentes « perspectives »
- Un historique local des dernière modifications
- La construction incrémentale des projets Java grâce à son propre compilateur qui permet en plus de compiler le code même avec des erreurs, de générer des messages d'erreurs personnalisés, de sélectionner la cible (java 1.3 ou 1.4) et de mettre en oeuvre le scrapbook (permet des tests de code à la volée)
- Une exécution des applications dans une JVM dédiée sélectionnable avec possibilité d'utiliser un débogueur complet (points d'arrêts conditionnels, visualiser et modifier des variables, évaluation d'expression dans le contexte d'exécution, changement du code à chaud avec l'utilisation d'une JVM 1.4, ...)
- Propose le nécessaire pour développer de nouveaux plug−ins
- Possibilité d'utiliser des outils open source : CVS, Ant, Junit
- La plate−forme est entièrement internationalisée dans une dizaine de langue sous la forme d'un plug−in téléchargeable séparément
- Le gestionnaire de mise à jour permet de télécharger de nouveaux plug−ins ou nouvelles versions d'un plug−in déjà installées à partir de sites web dédiés (Eclipse 2.0).
- ...

1.2. Les différentes versions d'Eclipse

Pour chaque version d'Eclipse possède un nombre plus ou moins important de types nommés "build" :

- "Nightly Builds" : version en cours de développement créée de façon journalière contenant les modifications de la journée.
- "Integration Builds" : assemblage des sous−projets pour la réalisation de tests
- "Stable Builds" : version testée
- "Releases" : version diffusée et "fiable"

Il existe plusieurs versions de type "Release" d'Eclipse :

Date	Version
Mars 2005	3.0.2
Septembre 2004	3.0.1
Juin 2004	3.0
Mars 2004	2.1.3
Novembre 2003	2.1.2
Juillet 2003	2.1.1
Avril 2003	2.1
Novembre 2002	2.0.2
Septembre 2002	2.0.1

Juillet 2002	2.0
Novembre 2001	1.0

La version 3.1 est en cours de développement : cette version devrait permettre l'utilisation d'Eclipse avec Java 5.0.

Ce document couvre essentiellement les versions 2.x et 3.0.x d'Eclipse. Différents pictogrammes sont utilisés pour signaler des fonctionnalités apportées par une version particulière d'Eclipse :

: pour la version 2.0.x d'Eclipse

: pour la version 2.1.X d'Eclipse

: pour la version 3.0.X d'Eclipse

Par défaut, les fonctionnalités décrites le sont pour la version 2.x d'Eclipse.

1.3. Les différents sous projets d'Eclipse

Le projet Eclipse est divisé en plusieurs sous projets :

- le projet "Eclipse" : ce projet développe l'architecture et la structure de la plate−forme Eclipse.
- le projet "Eclipse Tools" : ce projet développe ou intègre des outils à la plate−forme pour permettre à des tiers d'enrichir la plate−forme. Il possède plusieurs sous projets tel que CDT (plug−in pour le developpeemnt en C/C++), GEF (Graphical Editing Framework), EMF (Eclipse Modeling Framework), Cobol (plug−in pour le développement en Cobol), VE (Visual Editor) pour la création d'IHM, UML2 pour une implémentation d'UML reposant sur EMF, ...
- le projet "Eclipse Technology" : ce projet, divisé en trois catégories, propose d'effectuer des recherches sur des évolutions de la plate−forme et des technologies qu'elle met en oeuvre.
- le projet "Eclipse Web Tools Platform" : ce projet a pour but d'enrichir la plate−forme enfin de proposer un framework et des services pour la création d'outils de développement d'applications web. Il est composé de deux sous projets : WST (Web Standard Tools) et JST (J2EE Standard Tools)
- le projet "Eclipse Test and Performance Tools Platform Project" : ce projet a pour but de développer une plate−forme servant de support à la création d'outils de tests et d'analyses
- le projet "Business Intelligence and Reporting Tools (BIRT) Project : ce projet a pour but de développer une plate−forme facilitant l'intégration de générateur d'états. Il est composé de 4 sous projets : ERD (Eclipse Report Designer), WRD (Web based Report Designer), ERE (Eclipse Report Engine) et ECE (Eclipse Charting Engine).

2. Installation et exécution

Eclipse 1.0 peut être installé sur les plate−formes Windows (98ME et SE / NT / 2000 / XP) et Linux.

Eclipse 2.0 peut être installé sur les plate−formes Windows (98ME et SE / NT / 2000 / XP), Linux (Motif et GTK), Solaris 8, QNX, AIX 5.1, HP−UX 11.

Eclipse 2.1 peut être installé sur toutes les plate−formes sitées précédemment mais aussi sous MAC OS X.

Quel que soit la plate−forme, il faut obligatoirement qu'un JDK 1.3 minimum y soit installé. La version 1.4 est fortement recommandée pour améliorer les performances et pouvoir utiliser le remplacement de code lors du débogage (technologie JPDA).

Lors de son premier lancement, Eclipse crée par défaut un répertoire nommé Workspace qui va contenir les projets et les éléments qui les composent.

Le principe pour l'installation de toutes les versions d'Eclipse restent le même et d'une grande simplicité puisqu'il suffit de décompresser le contenu de l'archive d'Eclipse dans un répertoire du système.

2.1. Installation d'Eclipse sous Windows

2.1.1. Installation d'Eclipse 1.0

Il faut télécharger le fichier eclipse−SDK−R1.0−win32.zip (36,5 Mo) et le dézipper.

Pour lancer Eclipse sous Windows, il suffit de double cliquer sur le fichier eclipse.exe.

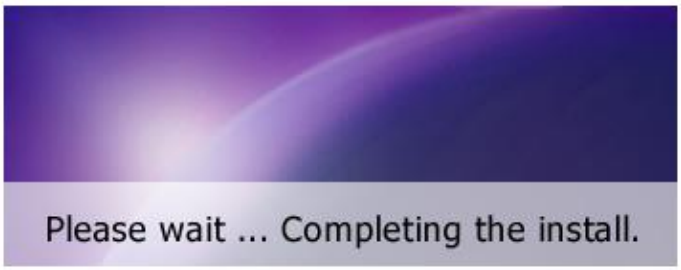
Si la splash screen reste affichée et que l'application ne se lance pas, c'est qu'elle n'arrive pas à trouver le JDK requis.

2.1.2. Installation d'Eclipse 2.0

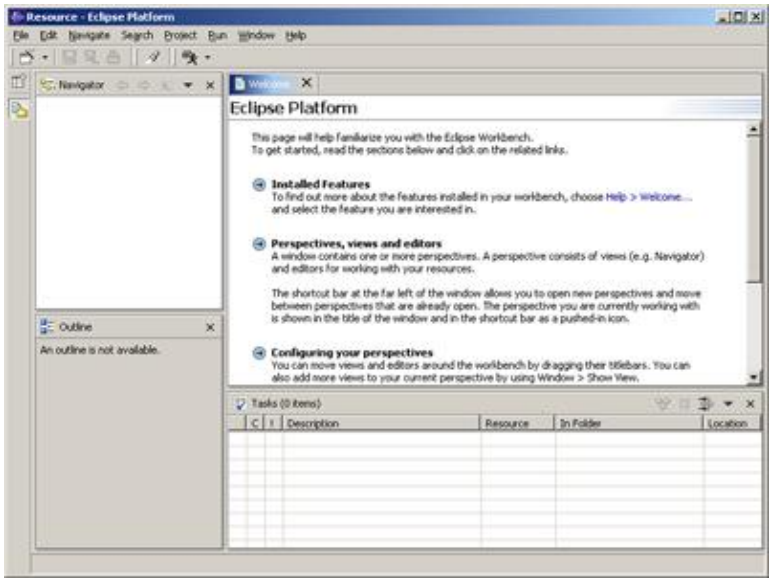
Il faut télécharger le fichier eclipse−SDK−2.0.2−win32.zip sur le site :

http://www.eclipse.org/downloads/index.php

Il suffit ensuite d' extraire l'archive dans un répertoire par exemple c:\java et d'éxécuter le programme eclipse.exe situé dans le répertoire eclipse créé lors de la décompression.



L'application termine l'installation, puis s'exécute.



2.1.3. Installation des traductions de la version 2.x

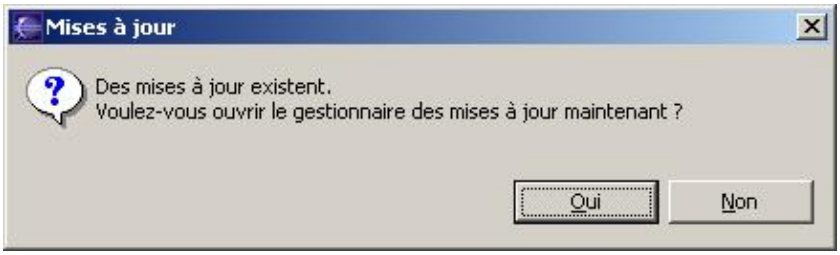
Par défaut, la langue utilisée dans Eclipse est l'anglais. I.B.M. propose des traductions pour la version 2.0.2 d'Eclipse dans différentes langues.

Il faut télécharger le fichier eclipse−nls−SDK−2.0.x.zip sur la page

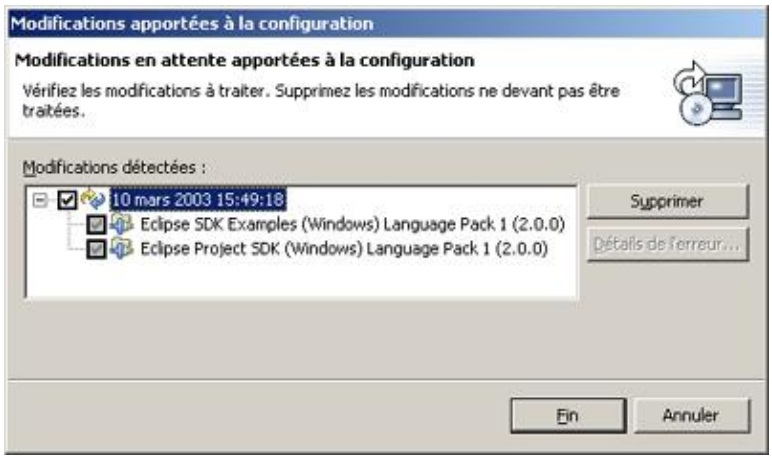
http://download.eclipse.org/downloads/drops/L−2.0.x_Translations−200301071000/index.php

Il suffit ensuite le décompresser dans le répertoire qui contient le répertoire Eclipse (par exemple : c:\java).

Avec une connexion internet, lors de l'éxecution suivante, l'application vérifie si une mise à jour des traductions n'est pas disponible.

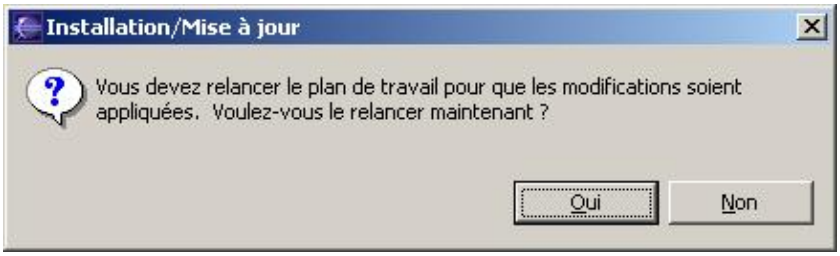


Un clic sur le bouton "Oui" ouvre une boîte de dialogue qui permet de sélectionner la mise à jour à installer.

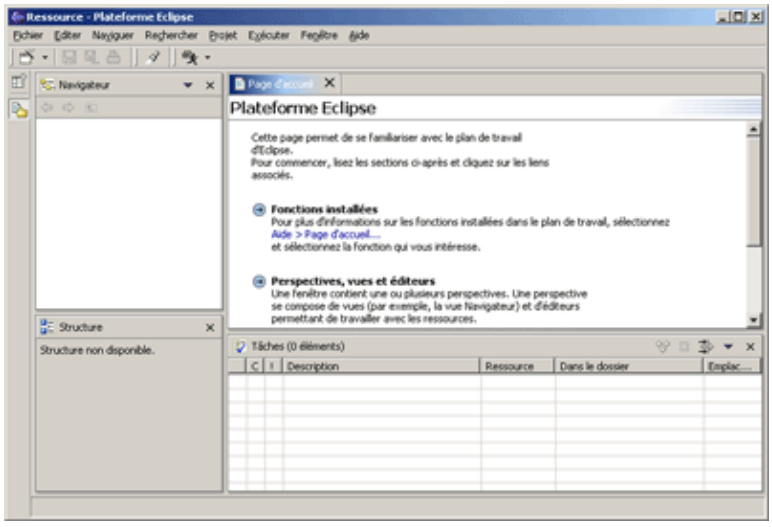


Il suffit de cocher la mise à jour souhaitée et de cliquer sur le bouton "Fin".

Les mises à jour sont téléchargées et installées. L'application doit être redémarrée.



L'application redémarre automatiquement après un clic sur le bouton "Oui", localisée dans la langue du poste.



Pour la version 2.1., il faut télécharger le fichier correspondant au système d'exploitation utilisé sur la page :

http://download.eclipse.org/downloads/drops/L-2.1.1_Translations-200307021300/index.php

La procédure d'installation est identique à celle de la version 2.0.

Pour la version 2.1.1., il existe aussi un patch pour les traductions téléchargeable à la même url nommé eclipse2.1.1.1-SDK-LanguagePackFeature-patch.zip

Ce patch doit être installé après avoir installé les traductions initiales de la version 2.1.1 en dézippant le contenu du fichier dans le répertoire qui contient le répertoire Eclipse.

2.1.4. Installation d'Eclipse 3.0.1

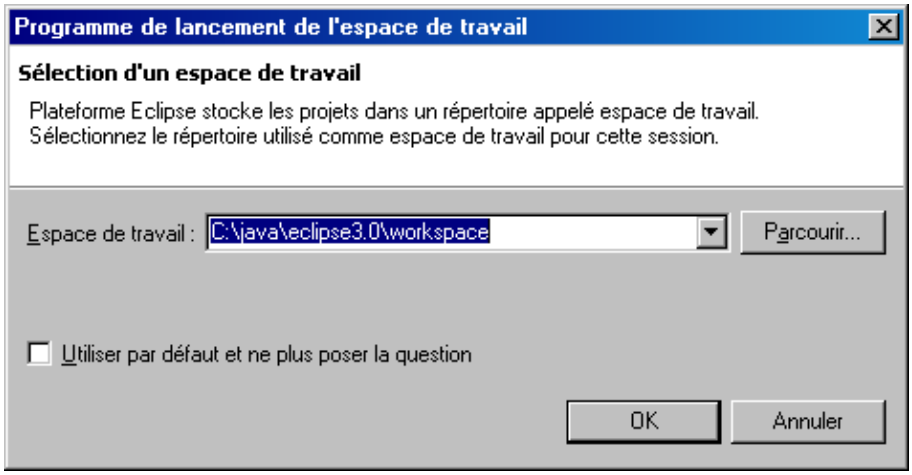
Il faut télécharger le fichier eclipse−SDK−3.0.1−win32.zip sur le site :

http://www.eclipse.org/downloads/index.php

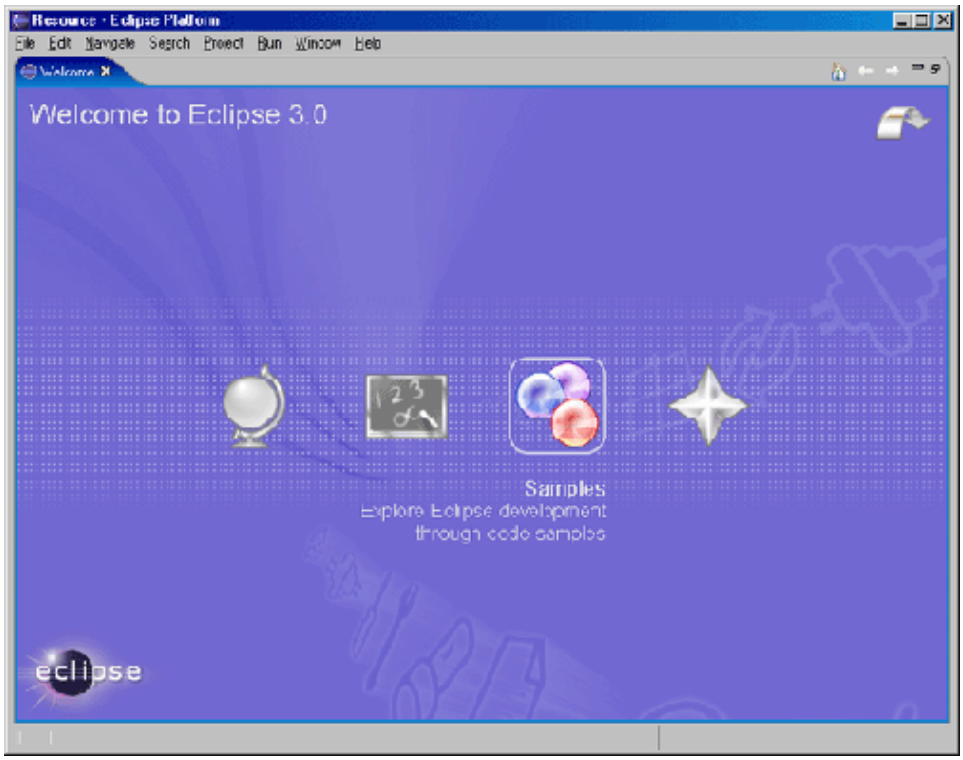
L'installation d'Eclipse 3.0 se fait de la même façon que pour les versions antérieures : ll suffit d' extraire l'archive dans un répertoire par exemple c:\java et d'éxécuter le programme eclipse.exe situé dans le répertoire eclipse décompressé.

Pour lancer Eclipse, il suffit de lancer le fichier eclipse.exe.

Durant la phase de lancement, Eclipse 3 propose de sélectionner l'espace de travail à utiliser. Par défaut, c'est celui présent dans le répertoire workspace du répertoire d'Eclipse qui est proposé.



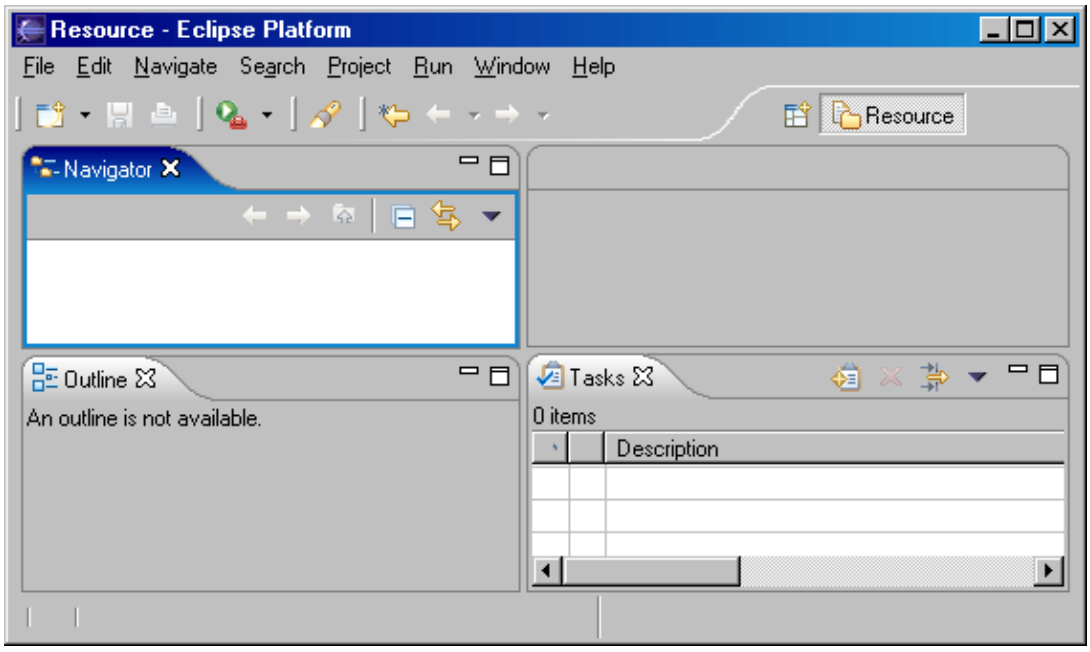
Cette demande est particulièrement utile lors de l'utilisation de plusieurs espaces de travail. Si un seul et unique workspace est utilisé, le plus simple est de cocher la case « Utiliser par défaut et ne plus poser la question » avant de cliquer sur le bouton « OK ». L'espace précisé deviendra alors celui par défaut qui sera toujours utilisé par Eclipse.



Eclipse 3.0 possède une page d'accueil permettant d'obtenir des informations sur l'outil réparties en quatre thèmes :

- overview : permet d'accéder rapidement à la partie de l'aide en ligne correspondant au thème sélectionné
- tutorials : permet d'accéder à des assistants qui permettent sous la forme de didacticiel de réaliser de simples applications ou plug−ins
- samples : permet de lancer des exemples d'applications avec SWT et JFace qu'il faut télécharger sur internet
- whats's new : permet d'accéder rapidement à la partie de l'aide en ligne concernant les nouveautés d'Eclipse 3.0

Eclipse 3.0 possède une nouvelle interface liée à une nouvelle version de SWT.



2.1.5. Installation des traductions de la version 3.0.x

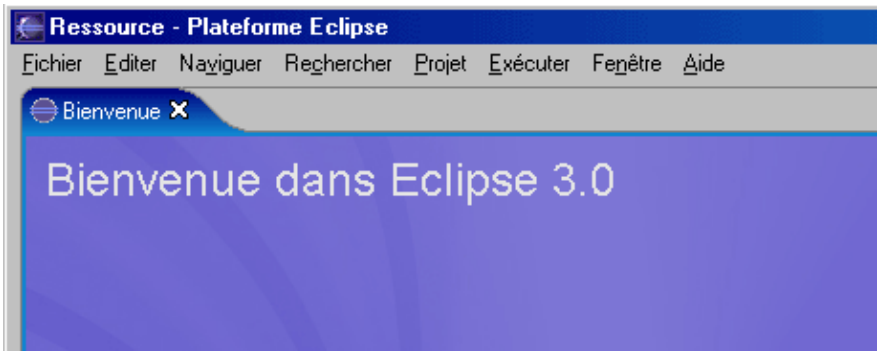
Par défaut, la langue utilisée dans Eclipse est l'anglais. I.B.M. propose des traductions pour les versions 3.0.x d'Eclipse dans différentes langues.

Il faut télécharger le fichier NLpack−eclipse−SDK−3.0.x−win32.zip sur la page

http://download.eclipse.org/downloads/drops/L−3.0.1_Translations−200409161125/index.php

Il suffit ensuite de le décompresser dans le répertoire qui contient le répertoire Eclipse (par exemple : c:\java).

Exécuter la commande eclipse –clean pour l'exécution suivante d'Eclipse.

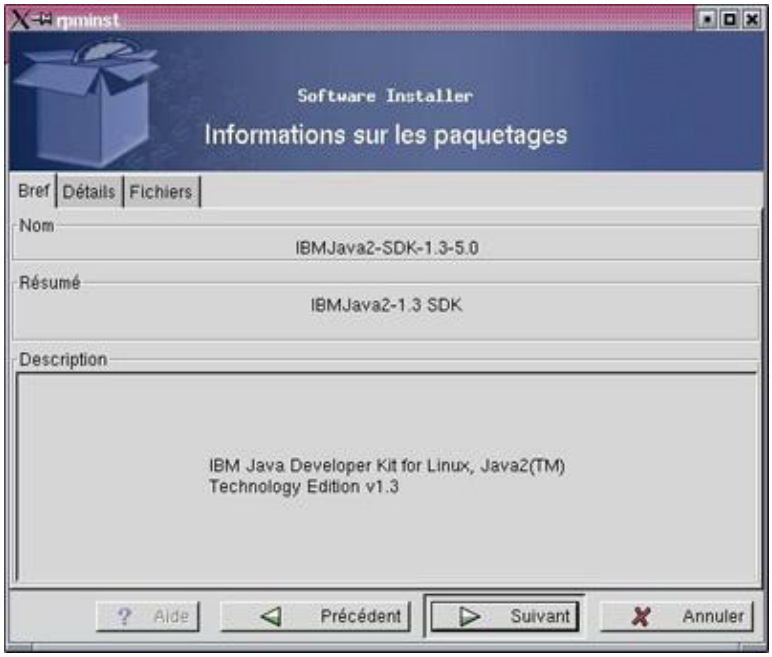


2.2. Installation d'Eclipse sous Linux

Toutes les versions d'Eclipse peuvent être installées sous Linux.

2.2.1. Installation d'Eclipse 1.0 sous Mandrake 8.1

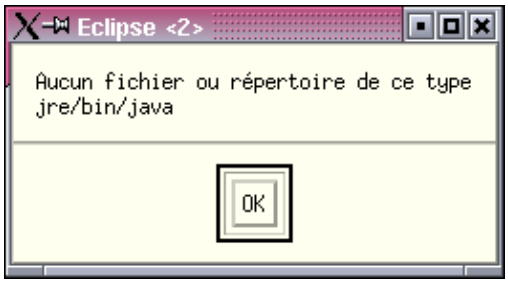
Il faut installer un JDK 1.3 minimum, par exemple celui fourni par IBM qu'il est possible d'installer grâce au gestionnaire de paquet.



Il faut ajouter le répertoire bin du JDK à la variable système PATH pour permettre au système de trouver les exécutables nécessaires.

PATH=$PATH:/opt/IBMJava2−13/bin

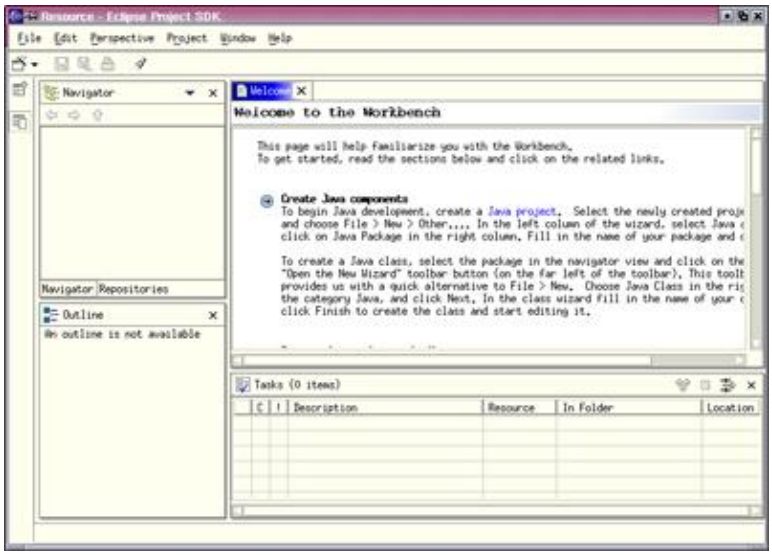
Si les exécutables ne sont pas trouvés, une boîte de dialogue affiche le message suivant :



Pour exécuter Eclipse, il suffit de lancer eclipse dans un shell.

Exemple :

[jumbo@charlemagne eclipse]$ ./eclipse −data workspace



2.2.2. Installation Eclipse 2.1 sous Mandrake 8.0

Il faut installer un JDK 1.3 minimum, par exemple celui fourni par IBM qu'il est possible d'installer grâce au gestionnaire de paquet. Il faut aussi que la variable JAVA_HOME contienne le chemin vers le JDK.

Exemple :

```
[java@localhost eclipse]$ echo $JAVA_HOME
/usr/java/jdk1.3.1
[java@localhost eclipse]$ java −version
java version "1.3.1"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.3.1−b24)
Java HotSpot(TM) Client VM (build 1.3.1−b24, mixed mode)
[java@localhost eclipse]$ which java
/usr/java/jdk1.3.1/bin/java
```

Il existe deux versions pour Linux en fonction de la bibliothèque graphique utilisée : une utilisant Motif et une autre utilisation GTK 2.

Pour la version Motif, il faut télécharger le fichier eclipse−SDK−2.1.1−linux−motif.zip.

Il faut décompresser le fichier dans un répertoire, par exemple /opt avec l'utilisateur root.

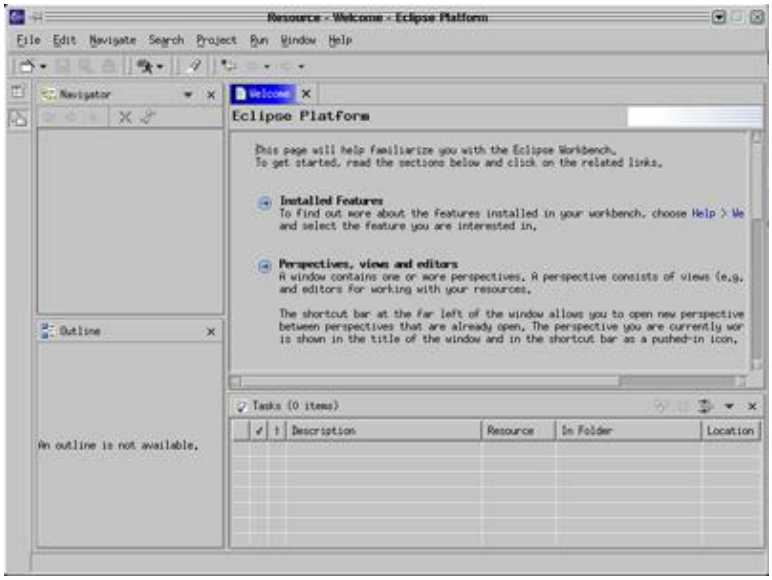
Exemple :

```
[root@localhost opt]# unzip eclipse−SDK−2.1−linux−motif.zip
Archive:  eclipse−SDK−2.1−linux−motif.zip
   creating: eclipse/
  inflating: eclipse/libXm.so.2.1
    linking: eclipse/libXm.so        −> libXm.so.2.1
    linking: eclipse/libXm.so.2      −> libXm.so.2.1
   creating: eclipse/plugins/
        ...
[java@localhost eclipse]$ ll
total 2004
drwxrwxr−x    5 root     root         4096 Mar 27 22:11 ./
drwxr−xr−x   11 root     root         4096 Jul 10 00:12 ../
−rw−rw−r−−    1 root     root           59 Mar 27 22:11 .eclipseproduct
−rw−rw−r−−    1 root     root        15048 Mar 27 22:11 cpl−v10.html
−rwxr−xr−x    1 root     root        41003 Mar 27 22:11 eclipse*
drwxrwxr−x   10 root     root         4096 Mar 27 22:11 features/
−rw−rw−r−−    1 root     root        10489 Mar 27 22:11 icon.xpm
−rw−rw−r−−    1 root     root          619 Mar 27 22:11 install.ini
lrwxrwxrwx    1 root     root           12 Jul 10 00:11 libXm.so −> libXm.so.2.1*
lrwxrwxrwx    1 root     root           12 Jul 10 00:11 libXm.so.2 −> libXm.so.2.1*
−rwxr−xr−x    1 root     root      1915756 Mar 27 22:11 libXm.so.2.1*
−rw−rw−r−−    1 root     root         4743 Mar 27 22:11 notice.html
drwxrwxr−x   64 root     root         4096 Mar 27 22:11 plugins/
drwxrwxr−x    2 root     root         4096 Mar 27 22:11 readme/
−rw−rw−r−−    1 root     root        16549 Mar 27 22:11 startup.jar
```

Pour utiliser Eclipse avec un utilisateur sans privilège particulier, il faut définir la variable LD_LIBRAY_PATH et exécuter Eclipse

Exemple :

```
[java@localhost java]$ cd /opt/eclipse
[java@localhost eclipse]$ LD_LIBRAY_PATH=/opt/eclipse:$LD_LIBRARY_PATH
[java@localhost eclipse]$ /opt/eclipse/eclipse −data $HOME/workspace
```



Si la variable LD_LIBRAY_PATH n'est pas correctement valorisée, le message d'erreur suivant est affiché et Eclipse ne peut pas se lancer.

Exemple :

```
[java@localhost java]$ /opt/eclipse/eclipse -data $HOME/workspace
/opt/eclipse/eclipse: error while loading shared libraries: libXm.so.2: cannot l
oad shared object file: No such file or directory
```

2.2.3. Installation Eclipse 3.0.x sous Mandrake 10.1

Il faut obligatoirement qu'un JDK 1.3 minimum soit installé sur le système. Dans cette section le JDK utilisé est le 1.4.2 de Sun.

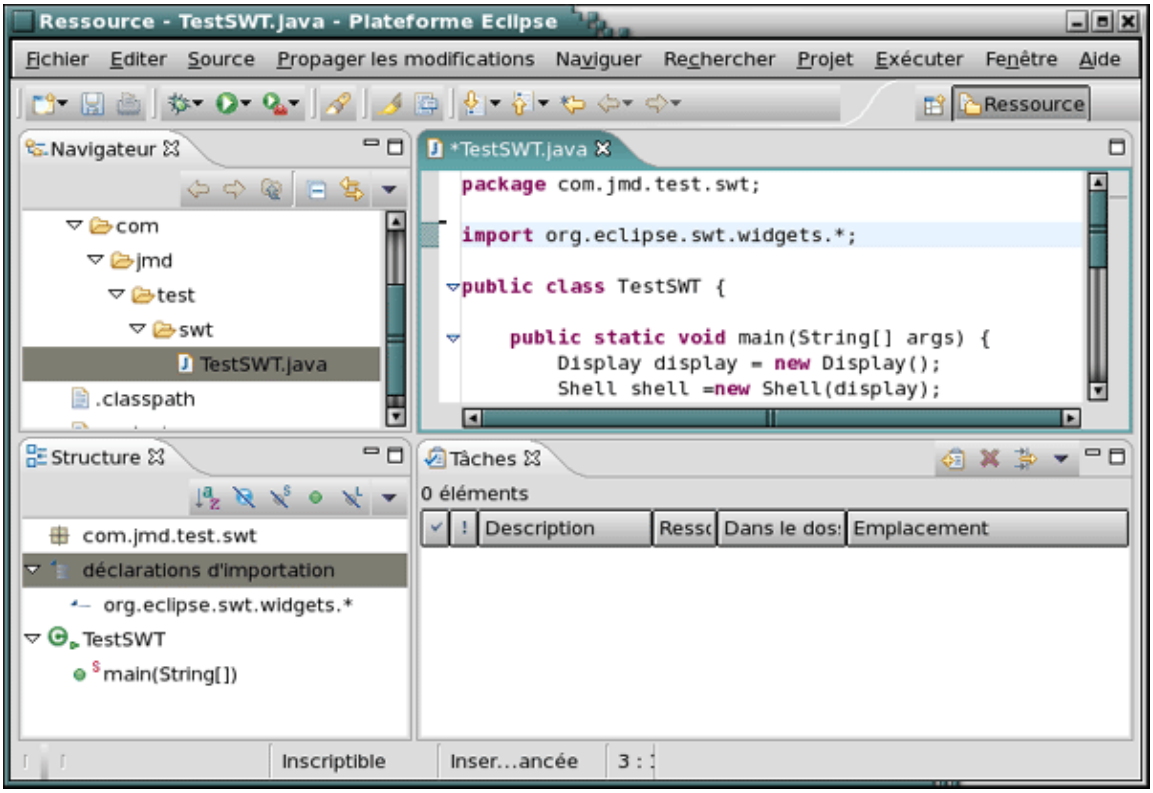
Exemple :

```
[java@localhost eclipse]$ java -version
java version "1.4.2_06"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.4.2_06-b03)
Java HotSpot(TM) Client VM (build 1.4.2_06-b03, mixed mode)
```

Deux versions existent pour Linux selon la bibliothèque graphique utilisée :

- une utilisant Motif
- une utilisant Gtk : eclipse-SDK-3.0-linux-gtk.zip

La version Gtk sera utilisée dans cette section.



2.2.3.1. Installation par et pour un seul utilisateur

Le plus simple est de décompresser le fichier eclipse−SDK−3.0−linux−gtk.zip dans le répertoire home de l'utilisateur.

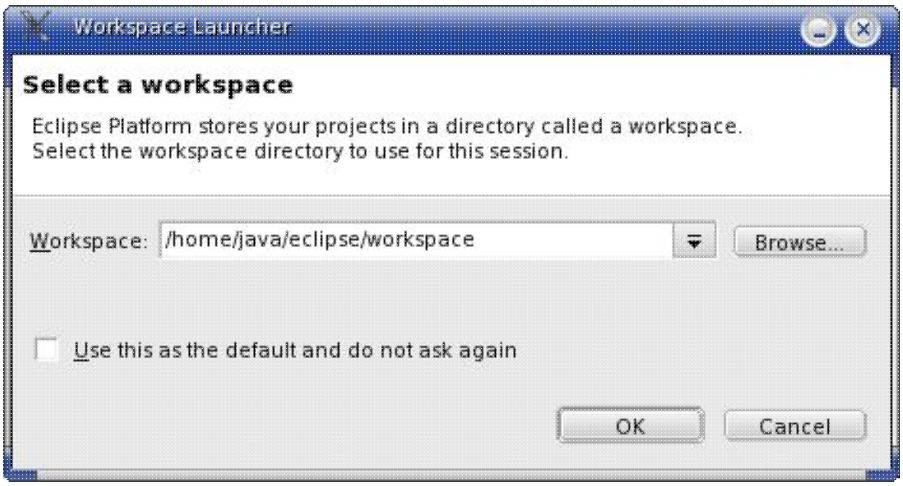
L'inconvénient de cette méthode est que par défaut seul cet utilisateur pourra utiliser Eclipse.

Exemple :

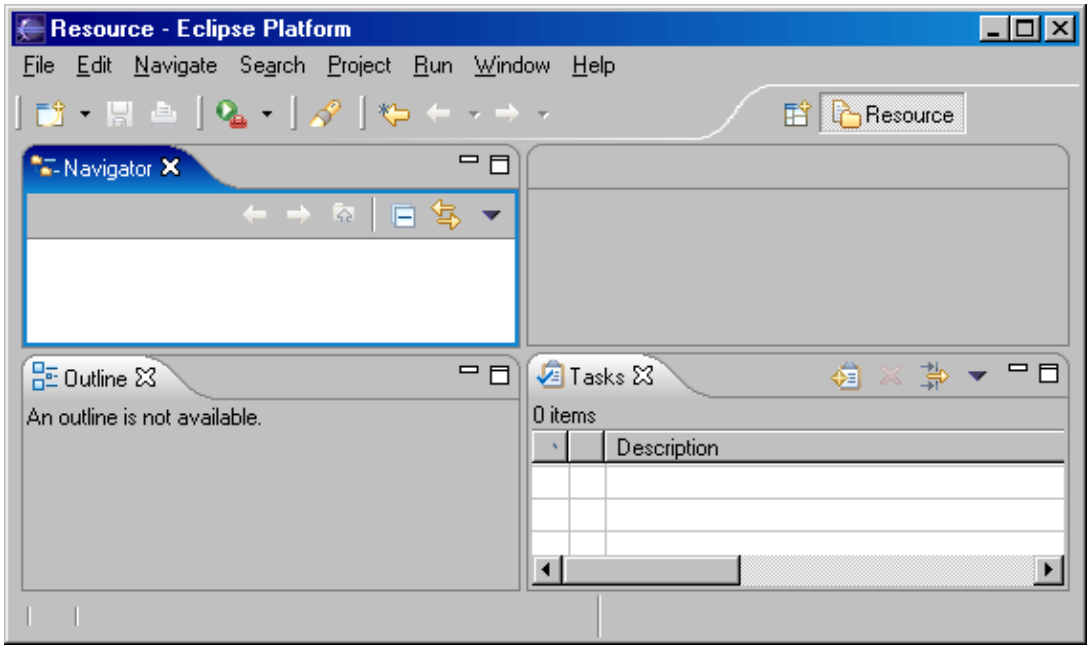
```
[java@localhost eclipse]$ cp eclipse−SDK−3.0−linux−gtk.zip ~
[java@localhost eclipse]$ cd ~
[java@localhost java]$ unzip eclipse−SDK−3.0−linux−gtk.zip
   creating: eclipse/plugins/org.eclipse.pde.build_3.0.0/
  inflating: eclipse/plugins/org.eclipse.pde.build_3.0.0/pdebuild.jar
   creating: eclipse/plugins/org.eclipse.pde.build_3.0.0/lib/
  inflating: eclipse/plugins/org.eclipse.pde.build_3.0.0/lib/pdebuild−ant.jar
  inflating: eclipse/plugins/org.eclipse.pde.build_3.0.0/.options
   creating: eclipse/plugins/org.eclipse.pde.build_3.0.0/feature/
...
 inflating: eclipse/startup.jar
  inflating: eclipse/icon.xpm
  inflating: eclipse/eclipse
[java@localhost java]$ ll
total 87140
drwxr−xr−x  3 java java     4096 oct 16 21:24 Desktop/
drwxr−xr−x  2 java java     4096 oct 16 22:34 Documents/
drwxr−xr−x  6 java java     4096 oct 18 23:23 eclipse/
−rwxr−−r−−  1 java java 89113015 oct 18 23:23 eclipse−SDK−3.0−linux−gtk.zip*
−rw−rw−r−−  1 java java        2 oct 16 22:23 java.sh~
drwx−−−−−−  3 java java     4096 oct 18 23:17 tmp/
[java@localhost java]$ cd eclipse
[java@localhost eclipse]$ ll
total 100
drwxrwxr−x   2 java java  4096 jun 25 18:43 configuration/
−rw−rw−r−−   1 java java 15049 jun 25 18:43 cpl−v10.html
−rwxr−xr−x   1 java java 27119 jun 25 18:43 eclipse*
drwxr−xr−x   9 java java  4096 oct 18 23:23 features/
−rw−rw−r−−   1 java java 10489 jun 25 18:43 icon.xpm
−rw−rw−r−−   1 java java  5810 jun 25 18:43 notice.html
drwxr−xr−x  85 java java  4096 oct 18 23:23 plugins/
drwxrwxr−x   2 java java  4096 jun 25 18:43 readme/
−rw−rw−r−−   1 java java 17663 jun 25 18:43 startup.jar
[java@localhost eclipse]$ ./eclipse&
[1] 3093
[java@localhost eclipse]$
```

Le fichier eclipse−SDK−3.0−linux−gtk.zip peut être supprimé.

Le workspace a utiliser peut être celui proposé par défaut (celui dans le répertoire d'installation d'Eclipse)



L'apparence d'Eclipse depend du thème et de l'environnement graphique utilisé, ici sous KDE :



2.2.3.2. Installation par root pour plusieurs utilisateurs

Eclipse peut être décompresser dans un répertoire accessible aux utilisateurs, par exemple /usr/local.

Exemple :

```
[root@localhost eclipse3.0.x]# cp eclipse-SDK-3.0-linux-gtk.zip /usr/local
[root@localhost eclipse3.0.x]# cd /usr/local
[root@localhost local]# ll
total 87164
drwxr-xr-x  2 root root     4096 jan  5  2004 bin/
drwxr-xr-x  2 root root     4096 jan  5  2004 doc/
-rwxr--r--  1 root root 89113015 oct 18 23:55 eclipse-SDK-3.0-linux-gtk.zip*
drwxr-xr-x  2 root root     4096 jan  5  2004 etc/
drwxr-xr-x  2 root root     4096 jan  5  2004 games/
drwxr-xr-x  2 root root     4096 jan  5  2004 include/
drwxr-xr-x  2 root root     4096 jan  5  2004 lib/
drwxr-xr-x  2 root root     4096 jan  5  2004 libexec/
drwxr-xr-x  3 root root     4096 oct 16 22:19 man/
drwxr-xr-x  2 root root     4096 jan  5  2004 sbin/
drwxr-xr-x  5 root root     4096 oct 14 01:57 share/
drwxr-xr-x  2 root root     4096 jan  5  2004 src/
[root@localhost local]# unzip -q eclipse-SDK-3.0-linux-gtk.zip
```

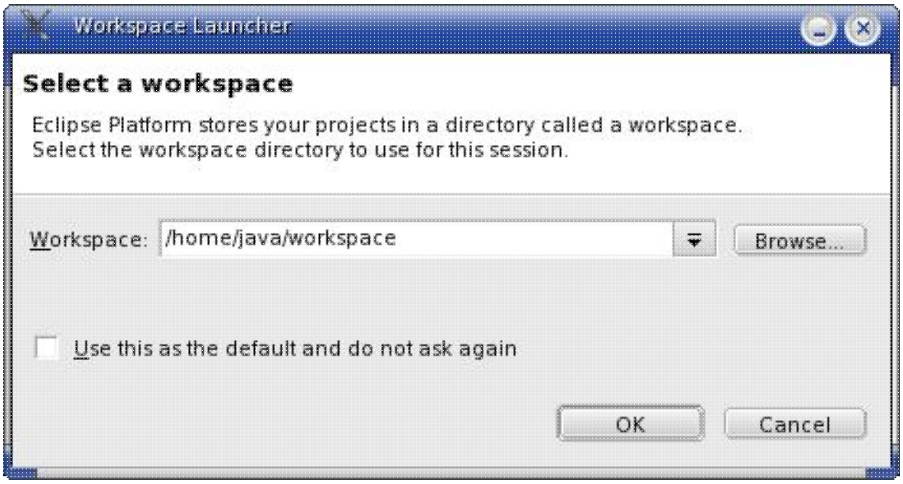
```
[root@localhost local]# ll
total 87168
drwxr-xr-x  2 root root     4096 jan  5  2004 bin/
drwxr-xr-x  2 root root     4096 jan  5  2004 doc/
drwxr-xr-x  6 root root     4096 oct 18 23:56 eclipse/
-rwxr--r--  1 root root 89113015 oct 18 23:55 eclipse-SDK-3.0-linux-gtk.zip*
drwxr-xr-x  2 root root     4096 jan  5  2004 etc/
drwxr-xr-x  2 root root     4096 jan  5  2004 games/
drwxr-xr-x  2 root root     4096 jan  5  2004 include/
drwxr-xr-x  2 root root     4096 jan  5  2004 lib/
drwxr-xr-x  2 root root     4096 jan  5  2004 libexec/
drwxr-xr-x  3 root root     4096 oct 16 22:19 man/
drwxr-xr-x  2 root root     4096 jan  5  2004 sbin/
drwxr-xr-x  5 root root     4096 oct 14 01:57 share/
drwxr-xr-x  2 root root     4096 jan  5  2004 src/
[root@localhost local]# rm eclipse-SDK-3.0-linux-gtk.zip
rm: détruire fichier régulier `eclipse-SDK-3.0-linux-gtk.zip'? o
[root@localhost local]#
```

Il suffit alors à un utilisateur d'exécuter Eclipse

Exemple :

```
[java@localhost java]$ /usr/local/eclipse/eclipse&
[1] 3676
[java@localhost java]$
```

Attention dans ce cas, il sera nécessaire de préciser le chemin d'un workspace utilisable en écriture par l'utilisateur.



2.2.3.3. Installation des traductions

L'installation des traductions se fait en décompresssant le fichier NLpack-eclipse-SDK-3.0.x-linux-gtk.zip, téléchargé sur le site d'Eclipse, dans le répertoire contenant le répertoire d'Eclipse.

Exemple :

```
[java@localhost java]$ ll
total 23664
drwxr-xr-x  3 java java     4096 nov  6 00:42 Desktop/
drwxr-xr-x  2 java java     4096 oct 19 00:00 Documents/
drwxr-xr-x  7 java java     4096 nov  6 11:03 eclipse/
-rw-rw-r--  1 java java        2 oct 16 22:23 java.sh~
```

```
-rwxr--r--  1 root root 24175833 nov  6 11:02 NLpack-eclipse-SDK-3.0.x-linux-gtk.zip*
drwx------  4 java java     4096 nov  6 11:00 tmp/
drwxr-xr-x  3 java java     4096 oct 18 23:59 workspace/
[java@localhost java]$ unzip Nlpack-eclipse-SDK-3.0.x-linux-gtk.zip
[java@localhost eclipse]$ ./eclipse -clean&
```

Comme sous Windows, l'exécution suivante d'Eclipse doit se faire avec l'option −clean.

2.3. Les paramètres

Eclipse 2.x accepte plusieurs paramètres sur la ligne de commande qui lance l'application.

Le paramètre –vm permet de préciser la machine virtuelle qui va exécuter Eclipse. Ce paramètre est utile lorsque la machine possède plusieurs JRE installés.

Exemple sous Windows :

Eclipse.exe −vm C:\java\jre1.4.2\bin\javaw.exe

Le paramètre –data permet de préciser l'emplacement du workspace. Ce paramètre est utile pour pouvoir utiliser plusieurs workspaces.

Exemple sous Windows :

Eclipse −data C:\Test\workspace

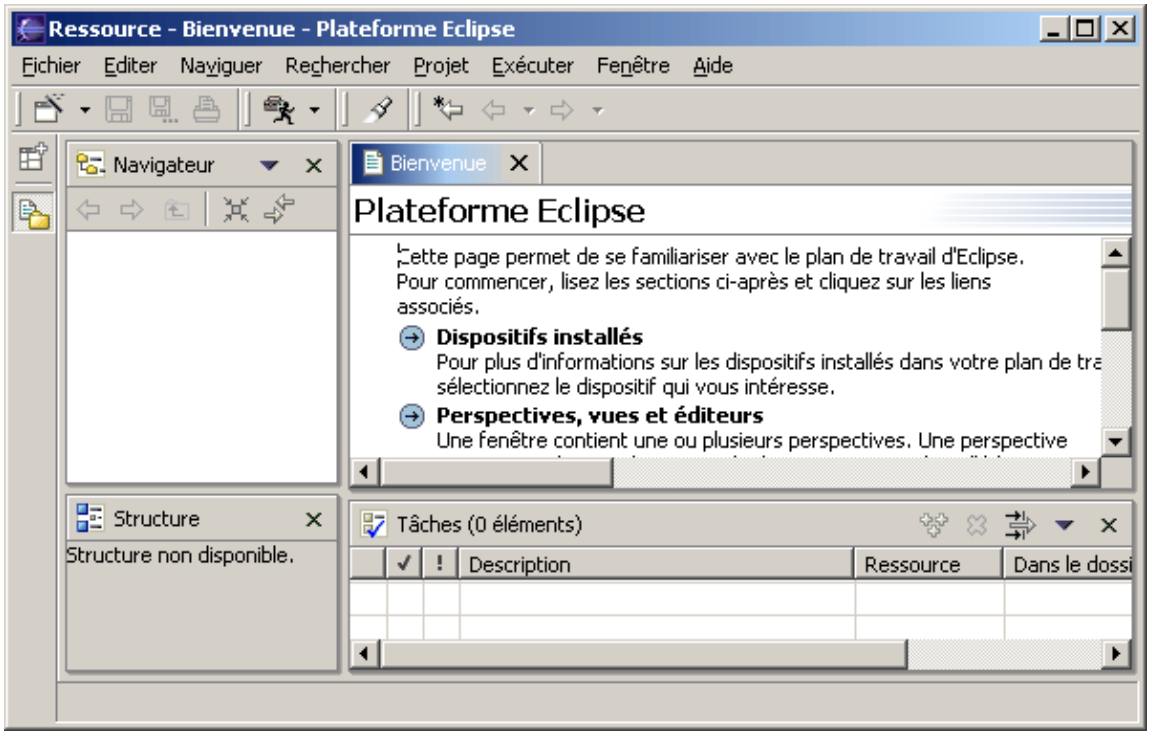
Le paramètre –ws permet sous les environnements de type Unix de préciser la bibliothèque graphique utilisée. Les valeurs possibles sont "motif" et "gtk".

La paramètre –arch permet de préciser l'architecture d'exécution.

La paramètre –vmargs permet de préciser des arguments à la machine virtuelle qui exécute Eclipse.

3. Le plan de travail (Workbench)

Au lancement d'Eclipse, une seule fenêtre s'ouvre contenant le plan de travail (Workbench). Le plan de travail est composé de perspectives dont plusieurs peuvent être ouvertes mais une seule est affichée en même temps. A l'ouverture, c'est la perspective "Ressource" qui est affichée par défaut.



Une perspective contient des sous fenêtres qui peuvent contenir des vues (views) et des éditeurs (editors)

La partie gauche du plan de travail est composée d'une barre qui contient une icône pour chaque perspective ouverte et une icône pour ouvrir une nouvelle perspective. L'icône enfoncée est celle de la perspective actuellement affichée. Le titre de la fenêtre du plan de travail contient le nom de la perspective courante.

Eclipse possède dans le plan de travail une barre de menu et une barre de tâches. Elles sont toutes les deux dynamiques en fonction du type de la sous fenêtre active de la perspective courante.

Eclipse propose de nombreux assistants pour faciliter la réalisation de certaines taches comme la création d'entités.

3.1. Les perspectives

Une perspective présente une partie du projet de développement selon un certain angle de vue.

Chaque perspective possède une icône qui permet de l'identifier plus rapidement. La version 1.00 d'Eclipse possède les perspectives suivantes :

Perspective	Icône	Rôle
Debug		Débogueur
Help		Aide en ligne
Java		Ecriture de code Java
Java Type Hierarchy		Navigation dans la hiérarchie et les éléments des classes
Plug−in Development		Création de plug−in
Resource		Gestion du contenu de l'espace de travail
Scripts		
Team		Gestion du travail collaboratif

Perspective	Icône	Rôle
Débogage		Débogueur
Java		Ecriture de code Java
Navigation Java		Navigation dans la hiérarchie et les éléments des classes
Hiérarchies des types Java		
Développement de plug−in		Création de plug−in
Ressource		Gestion du contenu de l'espace de travail
Installation/Mise à jour		Installation et mise à jour de plug−ins via le web
Exploration du référentiel CVS		Gestion du travail collaboratif avec CVS

Perspective	Icône	Rôle
Débogage		Débogueur
Java		Ecriture de code Java
Navigation Java		Navigation dans la hiérarchie et les éléments des classes
Hierarchie de type Java		
Développement de plug−in		Création de plug−ins
Ressource		Gestion du contenu de l'espace de travail
Synchronisation de l'équipe		
Exploration du référentiel CVS		Gestion du travail collaboratif avec CVS

Pour ouvrir une nouvelle perspective, il y a deux manières de procéder :

- Cliquer sur l'icône dans la barre des perspectives

- Utiliser l'option "Ouvrir" du menu "Perspective" (Eclipse 1.0) ou l'option "Ouvrir la perspective" du menu "Fenêtre" (Eclipse 2.0)

Lors d'un clic sur l'icône  , un menu flottant s'ouvre pour permettre la sélection de la perspective à ouvrir. Si elle n'appartient pas à la liste, elle est accessible en cliquant sur l'option « Autre ».

Eclipse 1.0

Eclipse 2.0

Un clic sur l'option « Autre... » ouvre une boite de dialogue dans laquelle il est possible de sélectionner la nouvelle perspective à afficher.

Eclipse 1.0

Eclipse 2.0

La perspective par défaut (celle qui est affichée à l'ouverture de l'application) est indiquée.

Il est possible d'ouvrir plusieurs perspectives d'un même type en même temps. Cependant une seule perspective, quelque soit son type est affichée à un moment donné.

Toutes les perspectives ouvertes possèdent une icône dans la barre des perspectives. Pour en afficher une, il suffit de cliquer sur son icône. La perspective courante est celle dont l'icône est enfoncée.

Dans Eclipse 3, la position de la barre des perspectives peut être modifiée. Par défaut, elle se situe en haut à droite. Le menu contextuel « Verrouiller » permet de modifier son positionnement.

L'option « Afficher le texte », cochée par défaut, permet d'avoir à coté de l'icône un nom court facilitant l'identification de chaque perspective.

3.2. Les vues et les éditeurs

Une perspective est composée de sous fenêtres qui peuvent être de deux types :

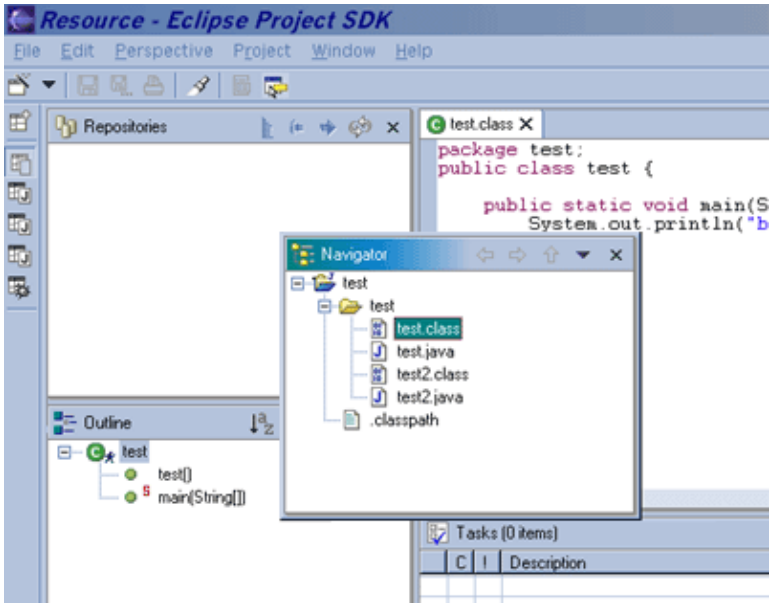
- les vues
- les éditeurs

Une vue permet de visualiser et de sélectionner des éléments. Il ne peut y avoir qu'une seule vue particulière dans une même perspective (il n'est pas possible d'afficher plusieurs fois la même vue dans une même perspective). Plusieurs vues différentes peuvent être rassemblées dans une même sous fenêtre un effectuant un cliquer/glisser de la barre de titre d'une vue sur une autre. L'accès à chaque vue se fait alors grâce à un onglet.

Un éditeur permet de visualiser mais aussi de modifier le contenu d'un élément. Un éditeur peut contenir plusieurs éléments, chacun étant identifié par un onglet.

Dans une perspective, il ne peut y avoir qu'une seule sous fenêtre active contenant soit un éditeur soit une vue. La barre de titre de cette sous fenêtre est colorée. Pour activer une sous fenêtre, il suffit de cliquer dessus.

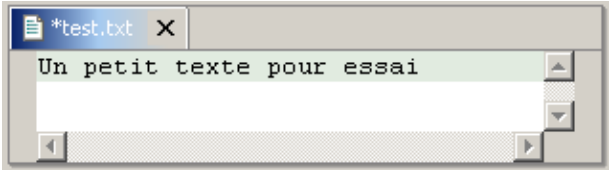
Avec Eclipse 1.0 sous Windows, les vues peuvent être extraites du workbench pour devenir des fenêtres indépendantes. Pour cela, il faut cliquer sur la barre de titre de la vue, en maintenant le bouton de la souris enfoncé, effectuer un glissement avec la souris vers une zone non empilable (sur un éditeur, les bords de l'écran, ...) et de relacher le bouton de la souris (le curseur de la souris prend la forme d'une petite fenêtre aux endroits adéquats).



Pour réaliser l'opération inverse, il faut faire glisser la fenêtre au dessus d'une vue existante : elles seront alors empilées.

3.2.1. Les éditeurs

Il existe plusieurs éditeurs en fonction du type de l'élément qui est édité.



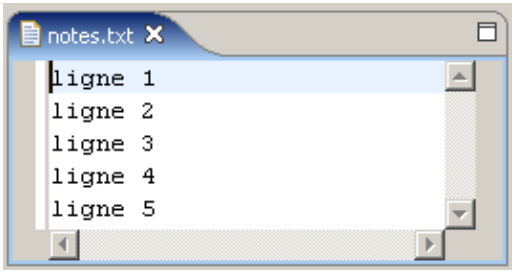
L'onglet de l'éditeur contient le libellé de l'élément traité. Une petite étoile apparaît à droite de ce libellé si l'élément a été modifié sans être sauvegardé.

Pour fermer l'éditeur contenant l'élément en cours, il suffit de cliquer sur l'icône en forme de croix de l'onglet.

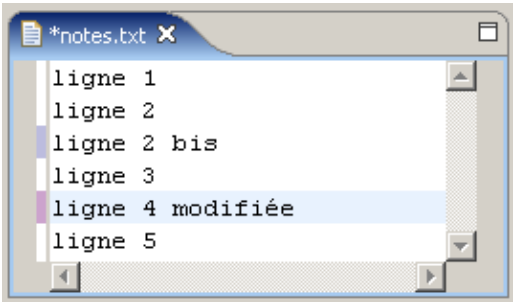
Une confirmation sera demandée en cas de fermeture alors que l'élément a été modifié sans sauvegarde. Il est aussi possible d'utiliser l'option "Fermer" et "Fermer tout" du menu "Fichier" du plande travail pour fermer respectivement le fichier en cours ou tous les fichiers.

Si l'élément édité ne possède pas d'éditeur dédié dans Eclipse, il tente d'ouvrir un outil associé au type de la ressource dans le système d'exploitation.

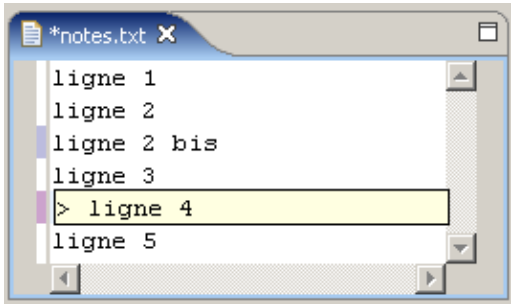
Quick Diff est une fonctionnalité qui permet de visualiser rapidement les modifications apportées dans un éditeur par rapport à une source (la version sur disque dans l'espace de travail par défaut).



Ainsi les lignes ajoutées et modifiées apparaissent avec une couleur différente dans la barre à gauche de la zone d'édition.

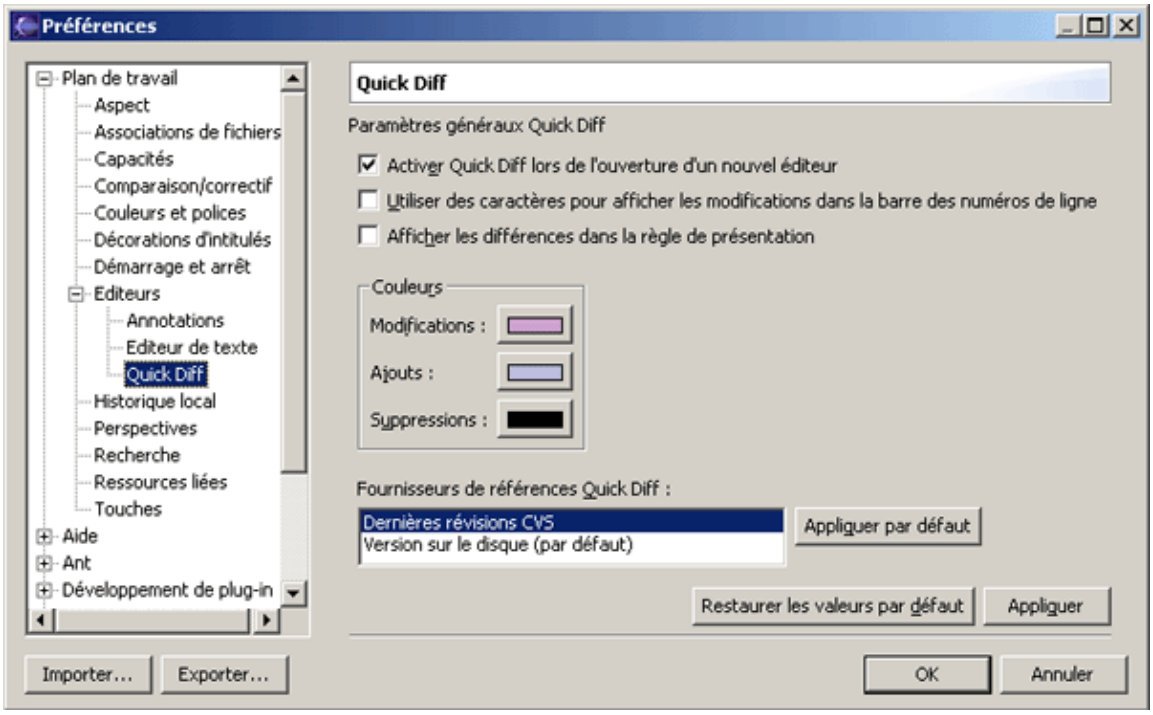


En laissant le curseur de la souris sur la zone colorée, un bulle d'aide affiche le contenu de la zone originale.

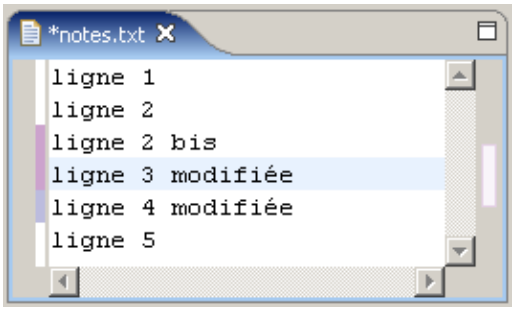


Pour activer ou désactiver Quick Diff, il est possible d'utiliser la combinaison de touche Ctrl+Maj+Q ou d'utiliser l'option « Activer/Désactiver Quick Diff » du menu contextuel de la barre de gauche.

Les paramètres de Quick Diff peuvent être configurés dans les préférences sous l'arborescence « Plan de travail / Editeurs/ Quick Diff ».



L'option "Afficher les différences dans la règle de présentation" permet de marquer les lignes modifiées par une zone blanche dans la barre à droite de l'éditeur.

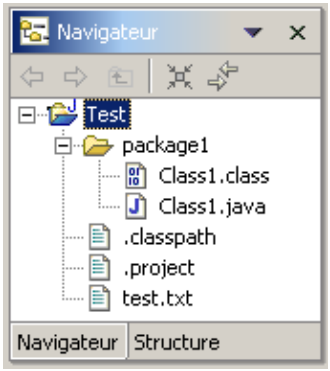


Plusieurs raccourcis ont été ajoutés dans les éditeurs :

Raccourcis clavier	Rôle
Alt+flèche vers le haut / bas	Déplacement d'un ensemble de lignes sélectionnées
Ctrl +Alt+flèche vers le haut	Copie d'un ensemble de lignes sélectionnées
Ctrl+Maj+Entrée	Insérer une ligne au dessus de la ligne courante
Maj+Entrée	Insérer une ligne en dessous de la ligne courante
Ctrl+Maj+Y	Conversion du texte sélectionné en minuscule
Ctrl+Maj+X	Conversion du texte sélectionné en majuscule

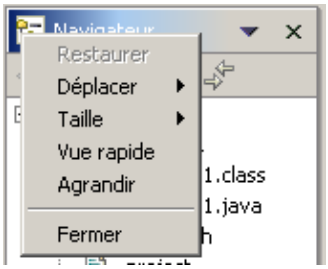
3.2.2. Les vues

Les vues permettent de présenter des informations et de naviguer dans les ressources. Plusieurs vues peuvent être réunies dans une même sous fenêtre : dans ce cas, le passage d'une vue à l'autre se fait via un clic sur l'onglet concerné.

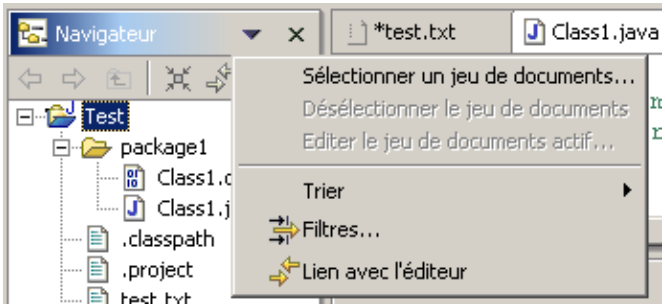


Les vues possèdent deux menus :

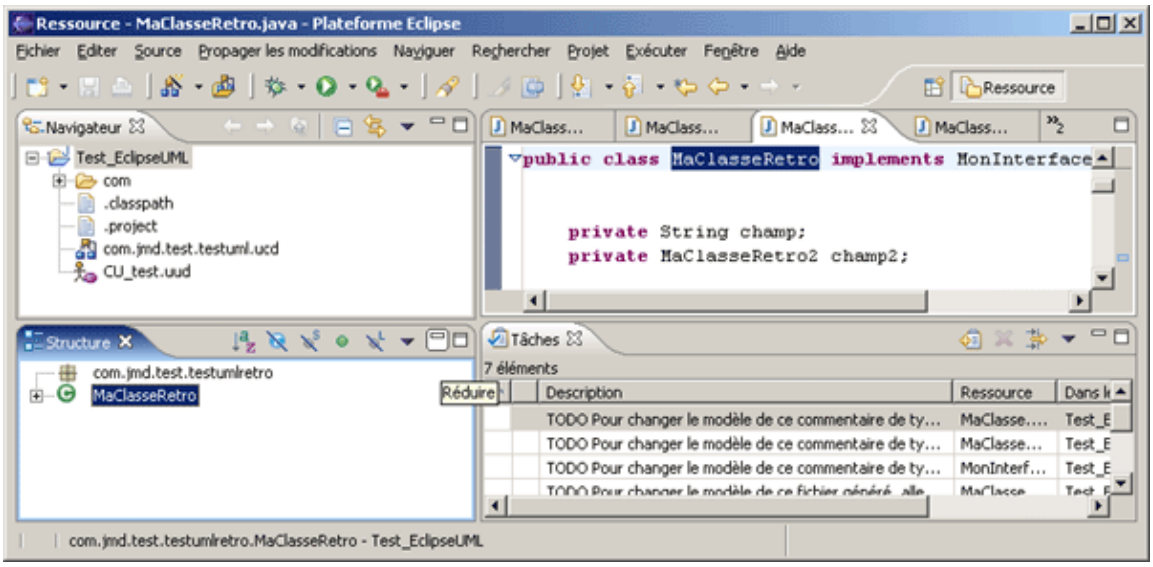
- un menu associé à la sous fenêtre activable en cliquant sur la petite icône en haut à gauche. Les options de ce menu concerne la sous fenêtre elle même.



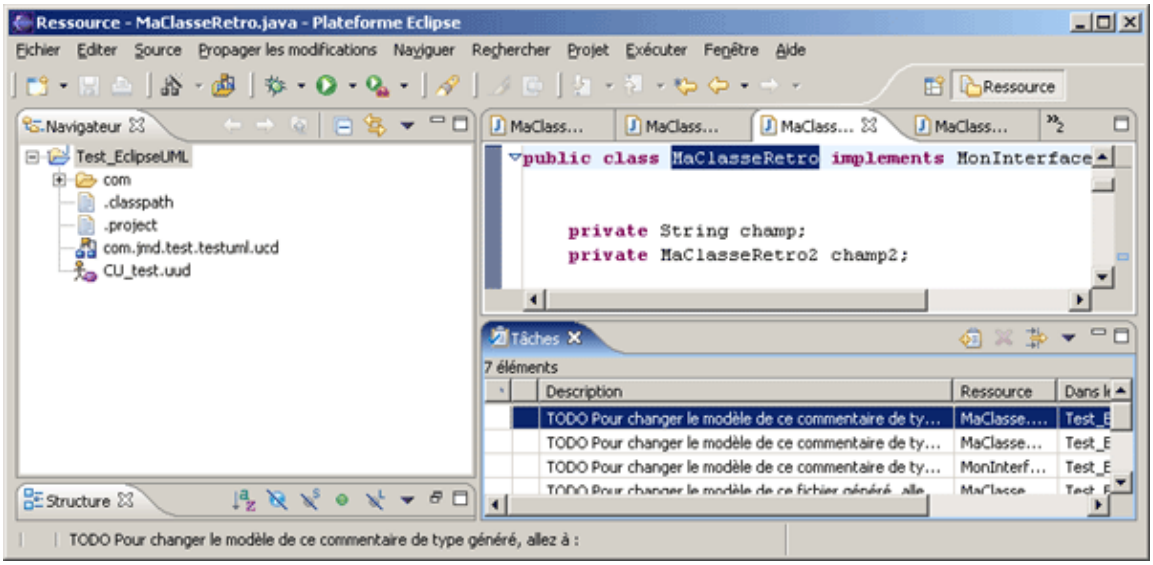
- le second menu est activable en cliquant sur l'icône en forme de triangle dont la base est en haut  . Les options de ce menu concerne le contenu de la vue notamment le tri ou le filtre.



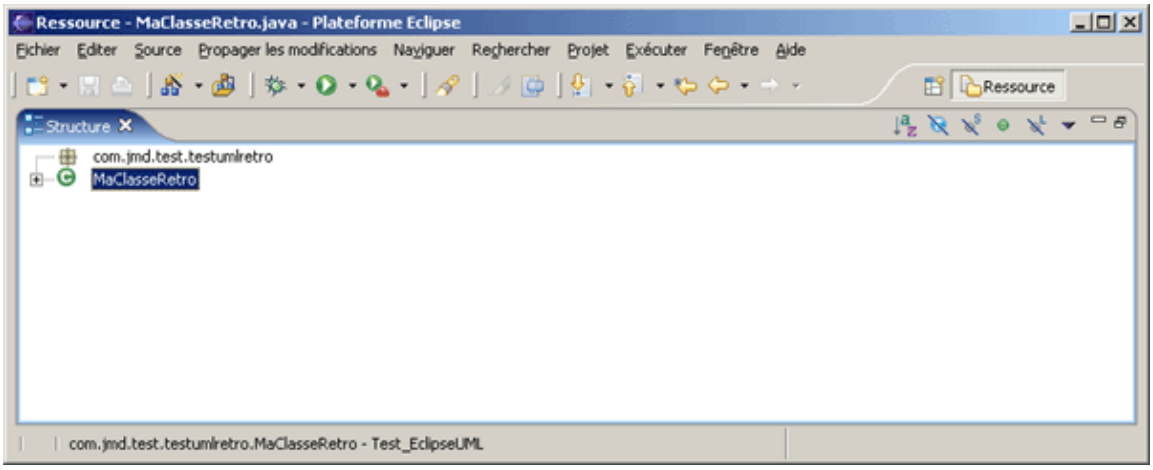
Eclipse 3.0 permet de réduire une vue ou d'agrandir une vue ou un éditeur.



Les vues peuvent être réduites en cliquant sur le bouton  .



Inversement un clic sur le bouton permet de restaurer la taille de la vue ou de l'éditeur. Un clic sur le bouton permet d'agrandir la vue ou l'éditeur.



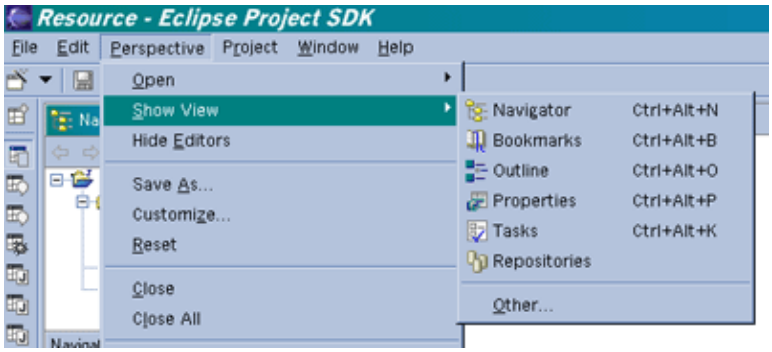
Cette option est particulièrement intéressante car elle évite d'avoir à modifier l'agencement des vues et des éditeurs pour maximiser la taille de l'un d'entre eux.

L'agrandissement et la restauration peuvent aussi être obtenues en double cliquant dans la barre de titre de la vue ou de l'éditeur.

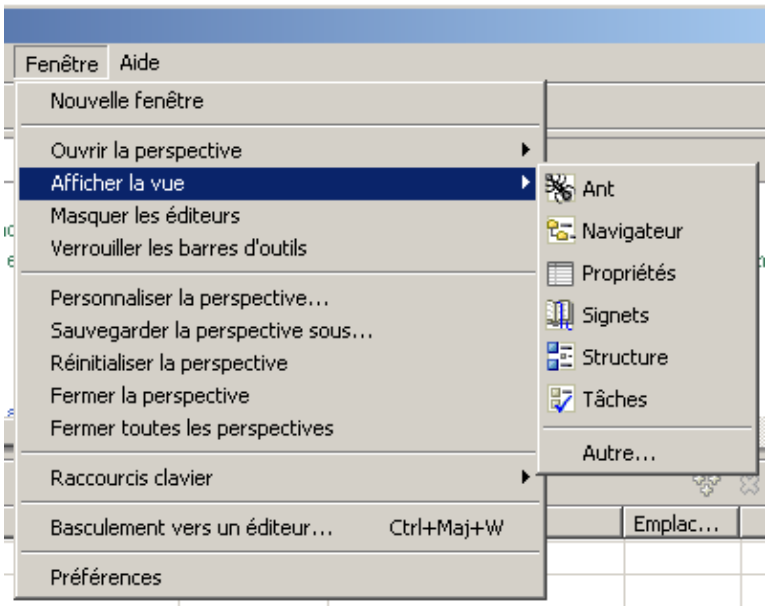
3.3. Organiser les composants de la perspective

Chaque perspective possède une organisation par défaut de ses sous fenêtres. Pour revenir à cette organisation par défaut, il faut utiliser l'option "Reset" du menu "Perspective" avec Eclipse 1.0 ou l'option "Réinitialiser la perspective" du menu "Fenêtre" dans Eclipse 2.0.

Avec Eclipse 1.0, l'option "Show View" du menu "Perspective" permet de visualiser une vue particulière qu'il suffit de sélectionner dans le sous menu.



Avec Eclipse 2.0, l'opération équivalente est effectuée en sélectionnant l'option "Afficher la vue" du menu "Fenêtre".

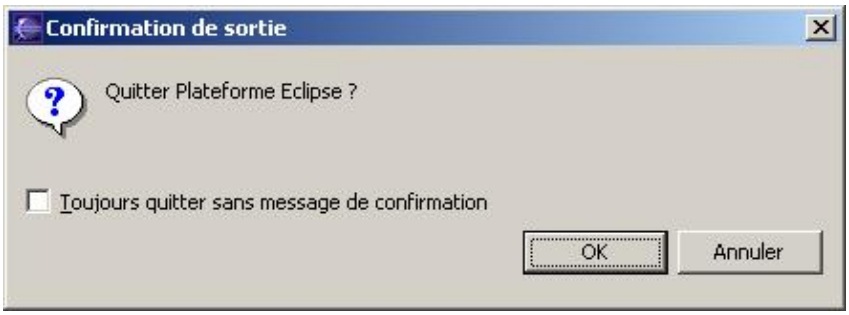


3.4. Fermer le plan de travail

Pour fermer le plan de travail et donc quitter Eclipse, il y a deux possibilités :

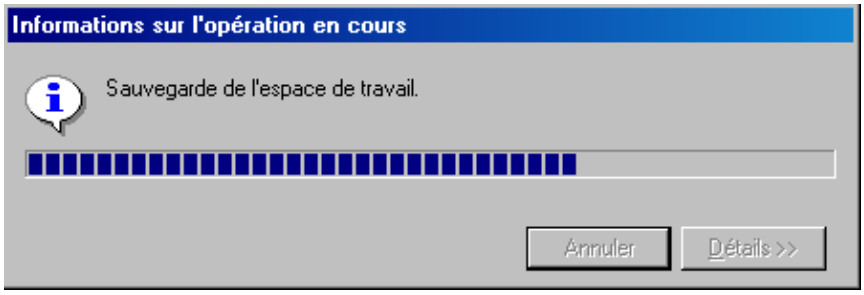
- Fermer la fenêtre du plande travail
- Sélectionner l'option "Quitter" du menu "Fichier"

Une boîte de dialogue permet de confirmation la fermeture de l'application.



Si des ressources doivent être sauvegardées, une boîte de dialogue apparaît contenant ces ressources. Il faut indiquer celles qui doivent être enregistrées et cliquer sur le bouton "OK".

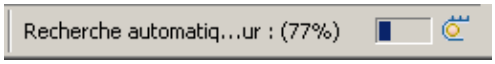
A sa fermeture, Eclipse enregistre certaines données dans l'espace de travail.



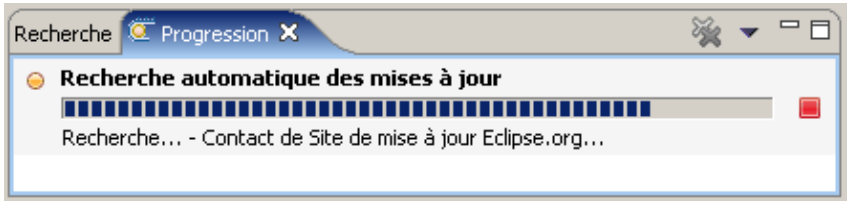
3.5. Exécution de traitements en arrière plan

Certains fonctionnalités longues en temps de traitement sont réalisées en tache de fond.

L'exécution d'un traitement en tâche de fond est signalée par un message dans la barre de statut



La vue « Progression » affiche des informations sur l'état d'avancement des traitements



4. L'espace de travail (Workspace)

L'espace de travail est l'entité qui permet de conserver les projets et leur contenu. Physiquement c'est un répertoire du système d'exploitation qui contient une hiérarchie de fichiers et de répertoires. Il y a d'ailleurs un répertoire pour chaque projet à la racine de l'espace de travail.

Il est possible de parcourir cette arborescence et d'en modifier les fichiers avec des outils externes à Eclipse.

L'espace de travail contient tous les éléments développés pour le projet : il est possible de créer, de dupliquer, de renommer ou de supprimer des éléments. Ces opérations de gestion sont réalisées dans la vue "Navigateur" de la perspective "Ressource".

4.1. La perpective "Ressource"

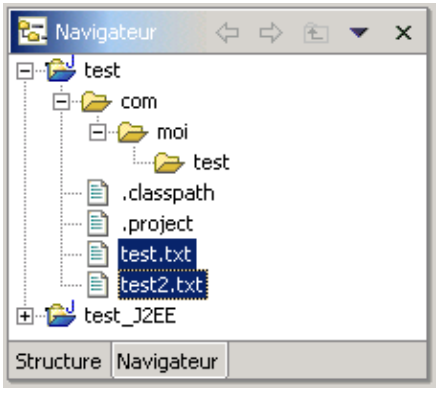
La perspective "Ressource" est la perspective qui s'ouvre par défaut au lancement d'Eclipse. Elle a pour but de gérer les différents éléments qui composent l'espace de travail : projets, dossiers et fichiers.

Par défaut, cette perspective contient les fenêtres suivantes :

- la vue "Navigateur" qui affiche les ressources (arborescence des fichiers) de l'espace de travail
- un éditeur qui permet d'éditer une ressource sélectionnée dans la vue "Navigateur"
- la vue "Structure" qui permet d'obtenir une arborescence présentant les grandes lignes de certaines ressources en cours de traitement
- la vue "Tâches" qui affiche une liste de tâche à effectuer

4.1.1. La vue "Navigateur"

Dans l'espace de travail, chaque projet contient une hiérarchie composée de dossiers et de fichiers. La vue "Navigateur" permet de présenter, de naviguer dans l'arborescence et de sélectionner une ressource.

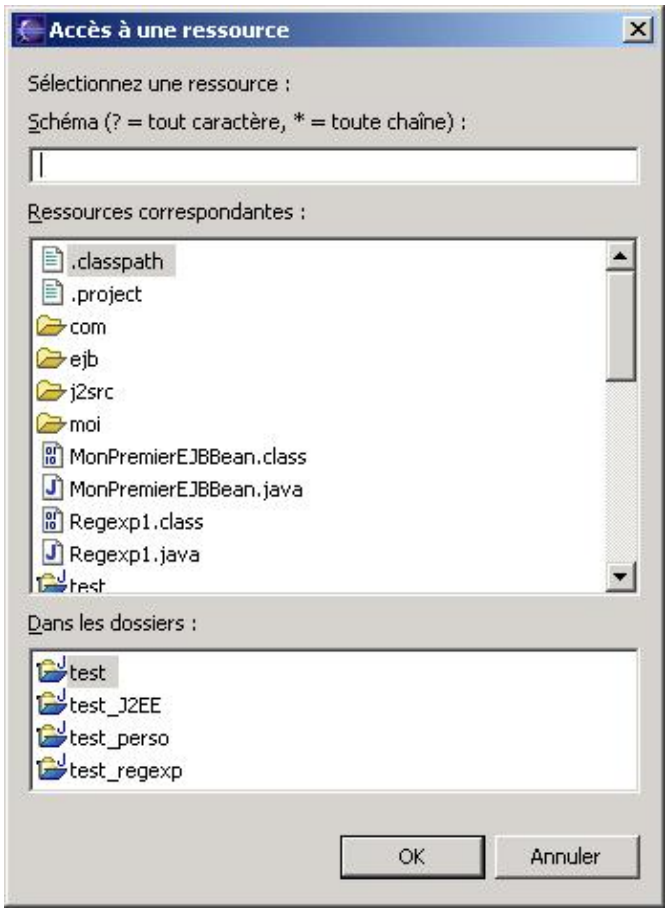


A partir de la vue "Navigateur", il est possible d'ouvrir le fichier sélectionné dans un éditeur :

- avec l'éditeur par défaut associé au type du fichier, il suffit de double cliquer sur le fichier dans le navigateur ou d'utiliser l'option "Ouvrir" du menu contextuel.
- avec un autre éditeur en utilisant l'option "Ouvrir Avec" du menu contextuel

L'association d'un type de fichier avec un éditeur peut être faite dans les préférences.

La vue "Navigateur" contient une option particulièrement pratique pour retrouver une ressource : l'outil "Accéder à". Cet outil permet à partir d'un motif (Pattern) de retrouver les ressources qui respectent le motif dans leur nom. L'option "Accéder à / Ressource" du menu contextuel de la vue "Navigateur" permet d'ouvrir une boîte de dialogue contenant l'outil.



Au fur et à mesure de la saisie du motif, la liste des fichiers correspondant s'affiche.

Il suffit de choisir le fichier et de cliquer sur le bouton "OK" pour fermer la boîte de dialogue et sélectionner le fichier dans la vue "Navigateur".

Par défaut, la vue "Navigateur" affiche tous les projets contenus dans l'espace de travail. Il est possible de limiter la vue à la hiérarchie d'un projet ou d'un dossier en le sélectionnant et en utilisant l'option "Suivant" du menu contextuel.

Les boutons  permettent de passer d'un mode à l'autre.

Le menu contextuel propose aussi des options pour copier, déplacer, renommer et supprimer une ressource.

4.2. La création de nouvelles entités

Dans Eclipse, on peut créer différents types d'entités qui seront stockées dans l'espace de travail :

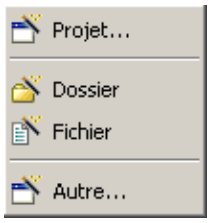
- des projets
- des répertoires pour organiser les projets
- des ressources de différents types qui sont des fichiers

Il existe plusieurs façons de créer ces entités :

- l'option « Nouveau » du menu « Fichier»
- l'option « Nouveau » du menu contextuel de la vue "Navigateur"
- le bouton « Assistant nouveau » dans la barre d'outils

La création est réalisée grâce à un assistant dont le contenu est dynamique en fonction de l'élément à créer.

L'option "Nouveau" du menu "Fichier" ou du menu contextuel de la vue "Navigateur" propose le même sous menu :

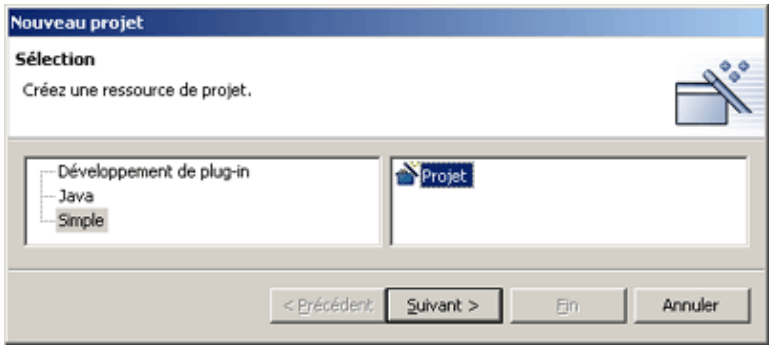


Il est ainsi possible de créer rapidement un projet, un répertoire ou un fichier. Si le fichier est d'un type particulier, un clic sur l'option "Autre" ouvre un assistant qui permet sur sa première page de sélectionner le type de l'entité à créer.

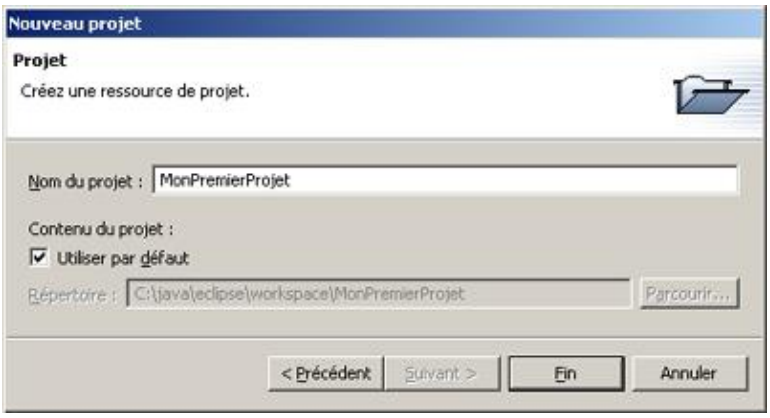
4.2.1. La création d'un projet

Le projet est l'unité de base de l'espace de travail. Chaque ressource doit être incluse directement dans un projet ou indirectement dans un répertoire appartenant à un projet.

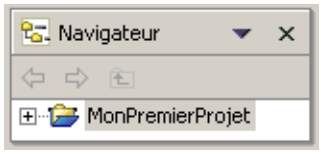
L'assistant permet de sélectionner le type du projet à créer. Il suffit alors de sélectionner la famille, le type du projet et de cliquer sur le bouton "Suivant".



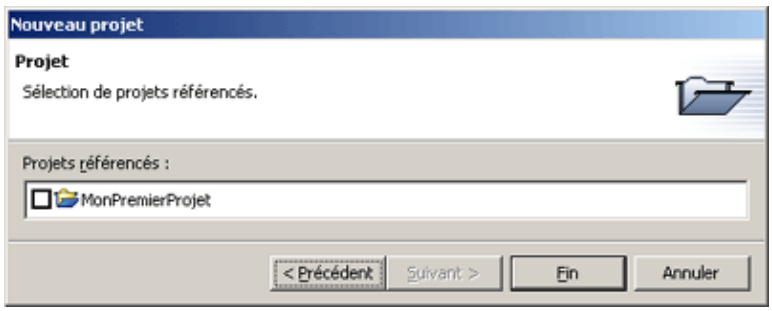
La création se fait grâce à un assistant qui demande le nom du nouveau projet. Ce nom ne doit pas contenir de blanc ou des caractères non alphabétiques.



Un clic sur le bouton "Fin" déclenche la création du projet. Le projet apparait dans la vue Navigateur.



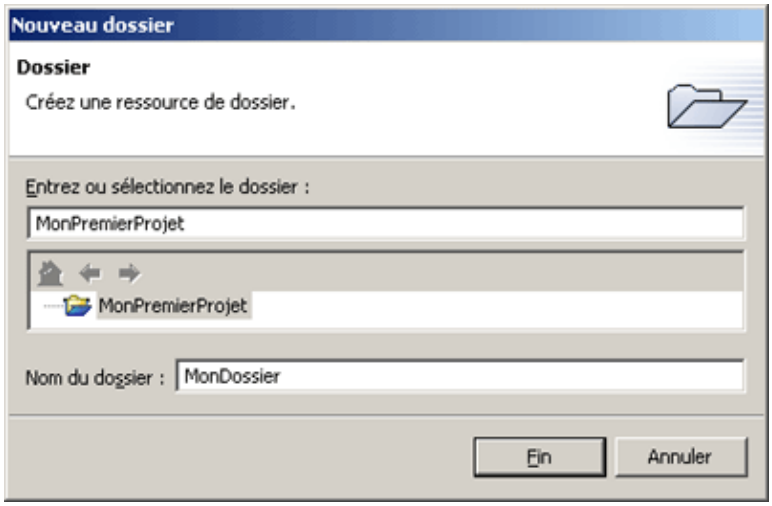
Si l'espace d travail contient déjà plusieurs projets, il est possible d'associer un ou plusieurs de ceux ci avec le nouveau en cours de création. Pour réaliser cette association, il suffit de cliquer sur le bouton "Suivant" pour afficher le second volet de l'assistant. Il suffit de cocher les projets concernés.



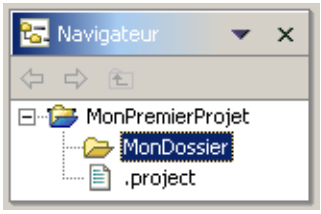
Il est possible de créer des projets particuliers selon le type d'applications à développer.

4.2.2. La création d'un répertoire

L'assistant de création de répertoire permet de créer un répertoire dans un projet. Par défaut, c'est le projet courant qui est sélectionné.



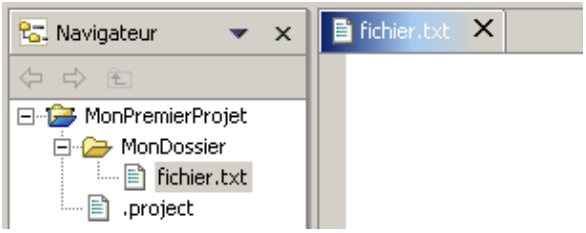
Il suffit ensuite de saisir le nom sans espace ni caractère non alphabétique et de cliquer sur le bouton "Fin". Le nouveau répertoire apparaît dans la vue "Navigateur".



4.2.3. La création d'un fichier

L'assistant de création de fichiers permet de choisir le projet et le répertoire dans lequel le fichier sera créé. Une fois cette localisation choisie il suffit de saisir le nom du fichier, son extension et de cliquer sur le bouton "Fin". Ce nom ne doit pas contenir de blanc ou des caractères non alphabétiques.

Si un éditeur est associé au type du nouveau fichier, l'éditeur est ouvert avec le nouveau fichier.



4.3. La duplication d'un élément

Dans la vue "Navigateur", pour dupliquer un élément, le plus simple est de faire un cliquer/glisser de l'élément en maintenant la touche Ctrl enfoncée vers son répertoire de destination dans le navigateur.

Il est aussi possible de sélectionner l'élément dans la vue "Navigateur" et d' effectuer une des opérations suivantes :

- appuyer sur la combinaison de touches Ctrl + C
- sélectionner l'option "Copier" du menu contextuel

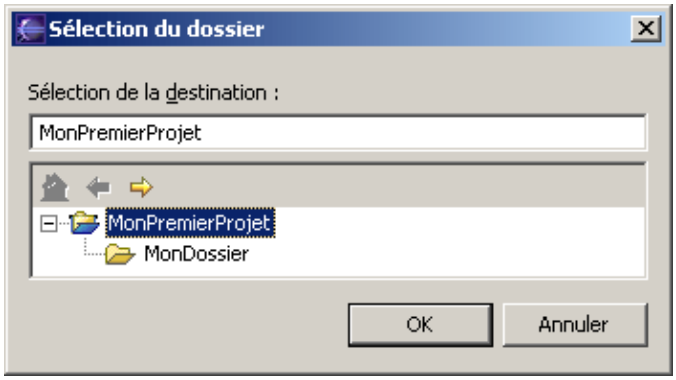
Il suffit alors de sélectionner dans la vue "Navigateur" le répertoire de destination et d'effectuer une des opérations suivantes :

- appuyer sur la combinaison de touches Ctrl + V
- sélectionner l'option "Coller" du menu contextuel

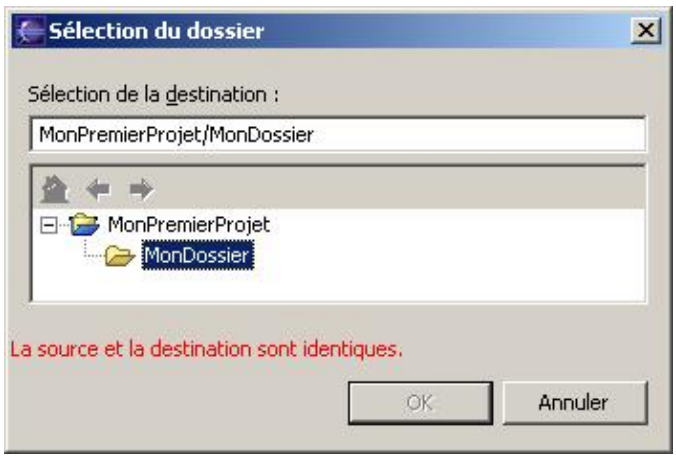
4.4. Le déplacement d'un élément

Dans la vue "Navigateur", pour déplacer un élément, le plus simple est de faire un cliquer/glisser de l'élément vers son répertoire de destination dans la vue "Navigateur".

Il est aussi possible de sélectionner l'option "Déplacer" du menu contextuel associé à cet élément et de sélectionner le répertoire de destination dans la boîte de dialogue.

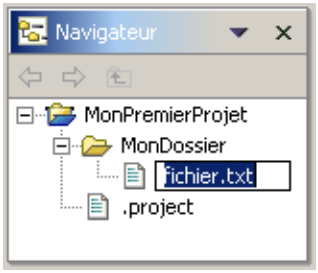


Si le répertoire de destination sélectionné est identique au répertoire d'origine du fichier, une message est affiché.



4.5. Renommer un élément

Pour nommer un élément, il suffit de sélectionner l'élément dans la vue "Navigateur", de sélectionner l'option "Renommer" du menu contextuel, saisir le nouveau nom et appuyer sur la touche "entrée".



4.6. La suppression d'un élément

Pour supprimer un élément, il suffit de le sélectionner dans la vue "Navigateur" et d'effectuer une des actions suivantes :

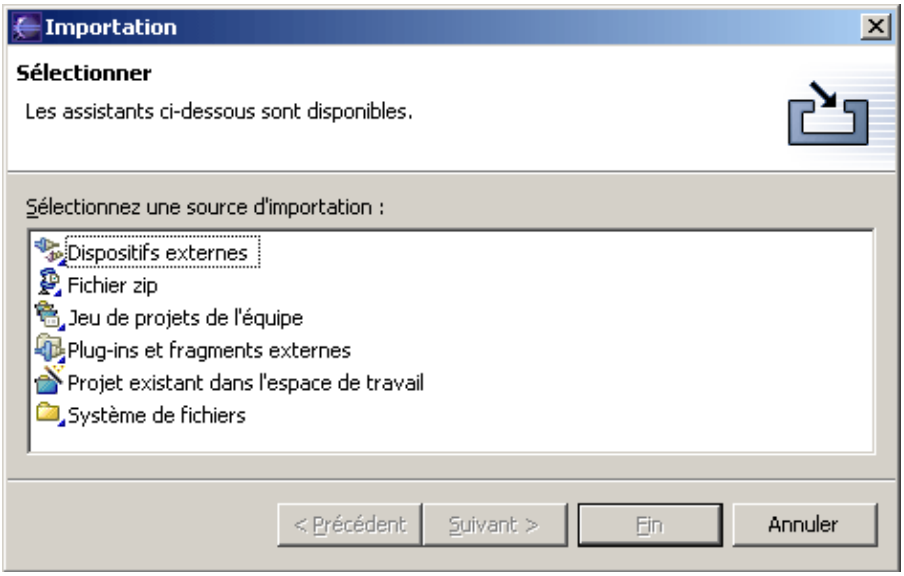
- choisir l'option "Supprimer" du menu contextuel de l'élément
- appuyer sur la touche Suppr
- choisir l'option "Supprimer" du menu "Editer"

Une boîte de dialogue demande de confirmer la suppression.

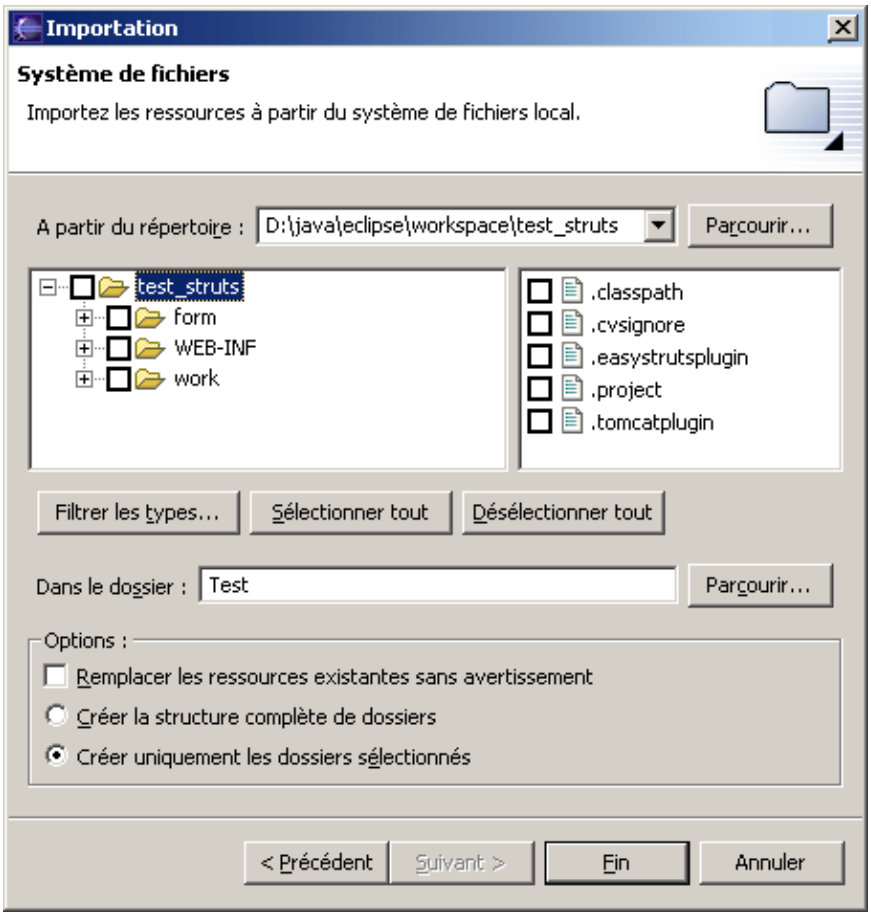
4.7. Importation

L'importation permet d'inclure dans l'espace de travail un certains nombre de fichiers externes. Attention, l'importation ne peut se faire que dans un projet existant.

Il faut utiliser le menu "Fichier/Importer"



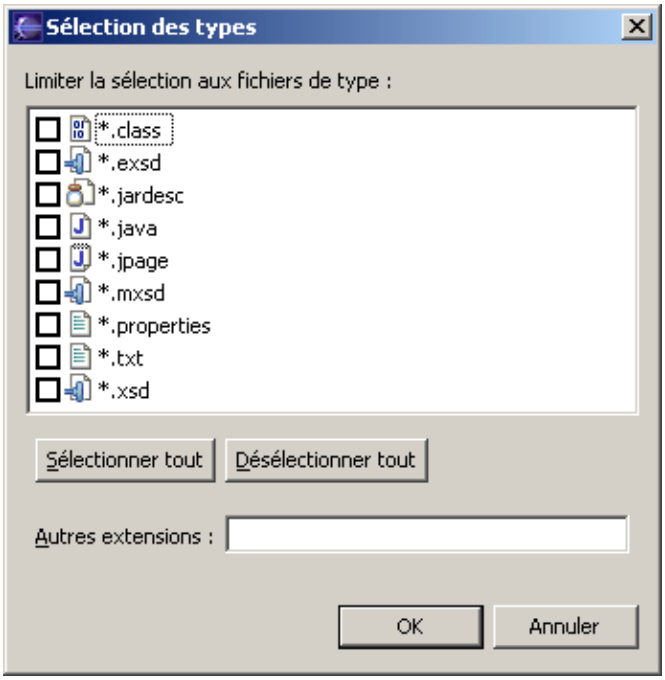
Sélectionnez le type de la source d'importation et cliquez sur le bouton "Suivant"



Sélectionnez le répertoire, puis cochez chacun des éléments concernés.

Il est très important de vérifier et de modifier si nécessaire le répertoire de destination qui doit être l'un des projets de l'espace de travail.

En cliquant sur le bouton "Filtrer les types ...", une boîte de dialogue permet de sélectionner les fichiers concernés à partir de leurs extensions.



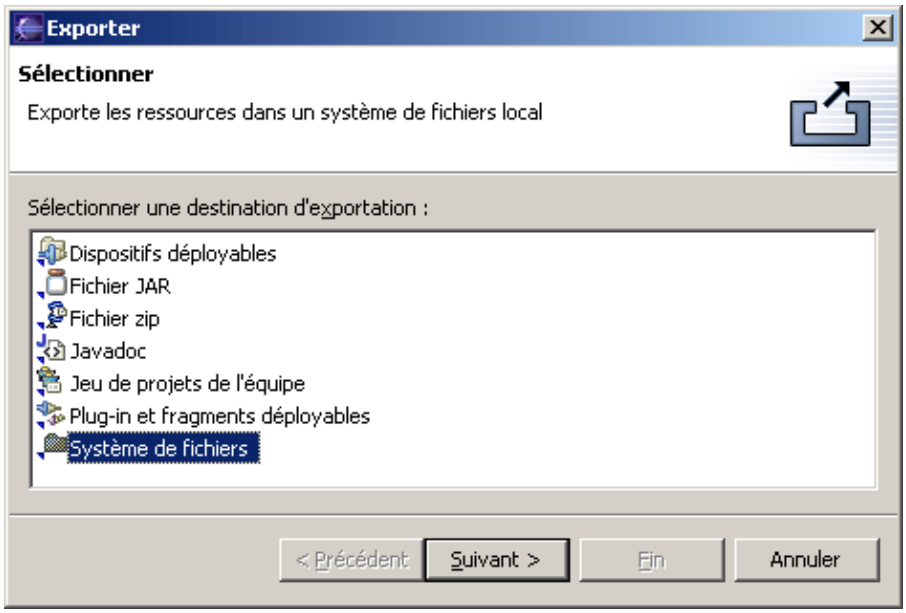
Il suffit de sélectionner les extensions parmi celles proposées, de saisir éventuellement d'autres extensions et de cliquer sur le bouton "OK". Le filtre est alors directement appliqué sur la sélection.

Une fois la sélection terminée, il suffit de cliquer sur le bouton "Fin" pour lancer l'importation.

Au cas ou une ressource existerait déjà dans la destination, un message demande la confirmation du remplacement.

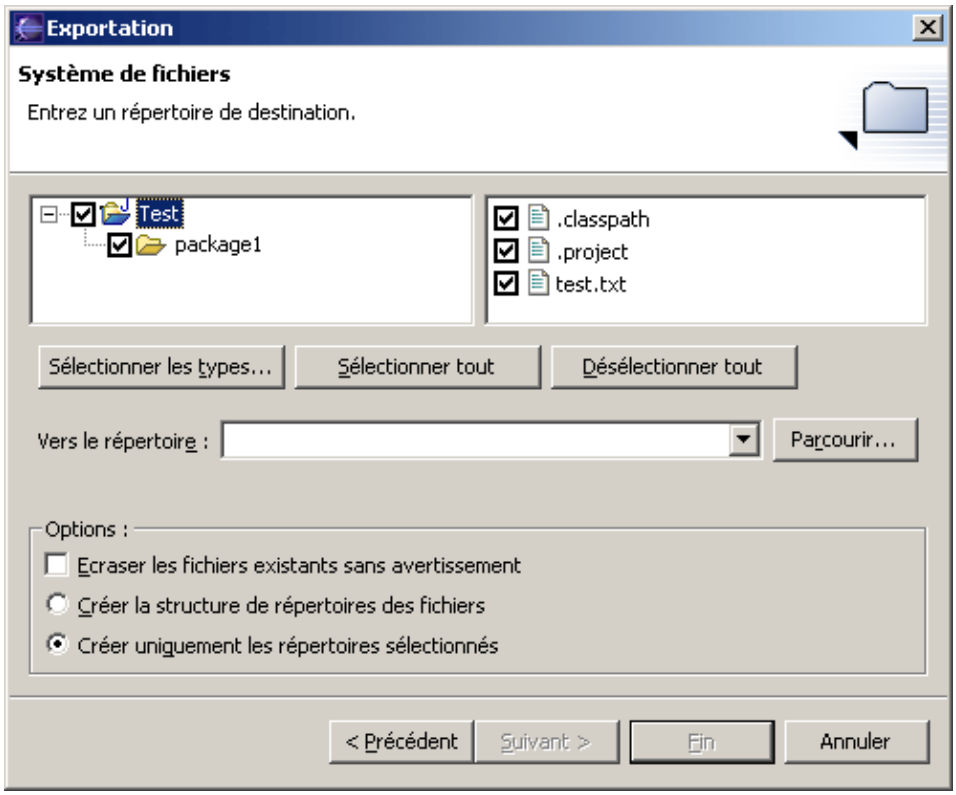
4.8. Exportation

Pour exporter tout ou partie du workspace, il faut utiliser le menu "Fichier/Exporter".



L'assistant demande de sélectionner le format d'exportation.

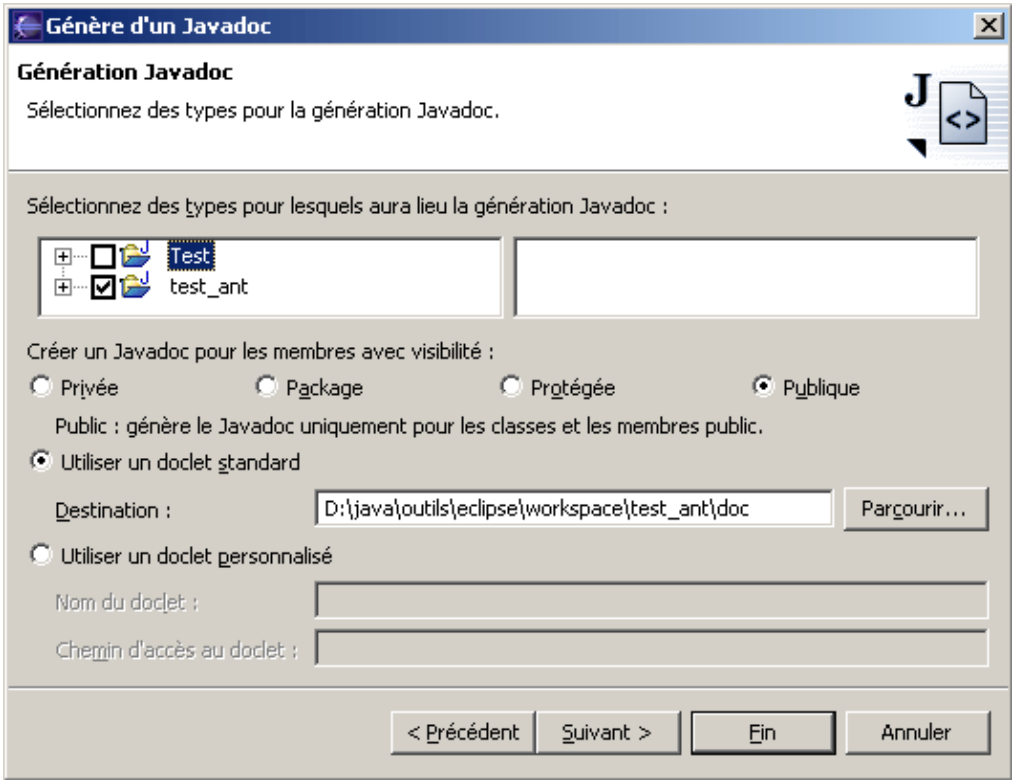
Si le format choisi est "Système de fichiers", l'assistant demande les informations nécessaires : les fichiers à exporter etle répertoire de destination



Attention : pour que la structure des répertoires soient conservées dans la cible, il faut obligatoirement que les

répertoires soient sélectionnés.

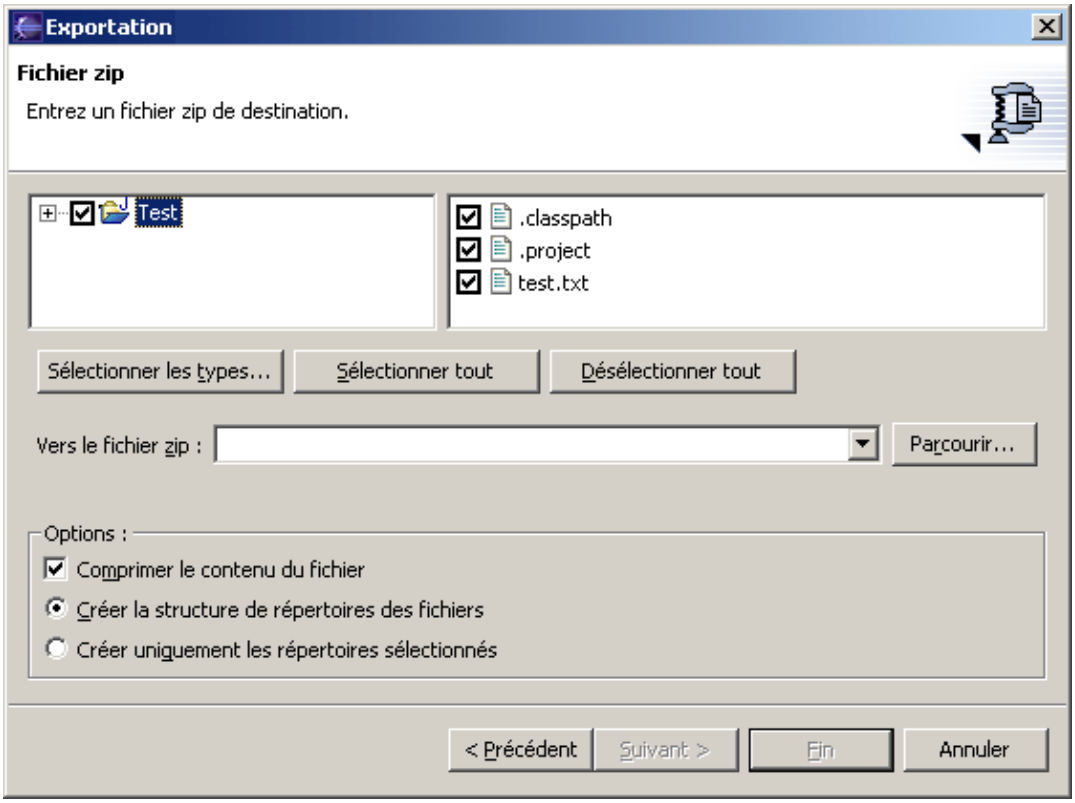
Si le format choisi est "Javadoc", l'assistant demande les informations nécessaires : les fichiers à exporter, le répertoire de destination et les options à utiliser lors de la génération



Les deux pages suivantes permettent de préciser des options pour l'outil Javadoc.

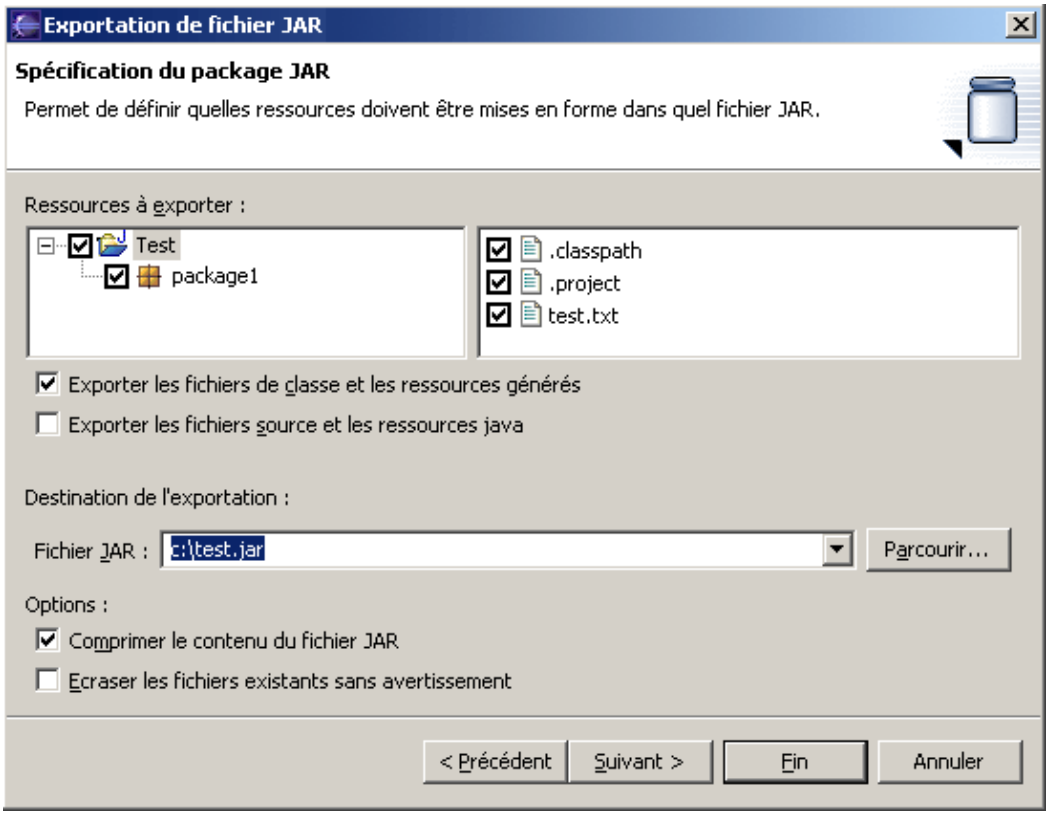
Un clic sur le bouton "Fin" permet de générer la documentation.

Si le format choisi est "Fichier Zip", l'assistant demande les informations nécessaires : les fichiers à exporter et le nom du fichier zip à générer.

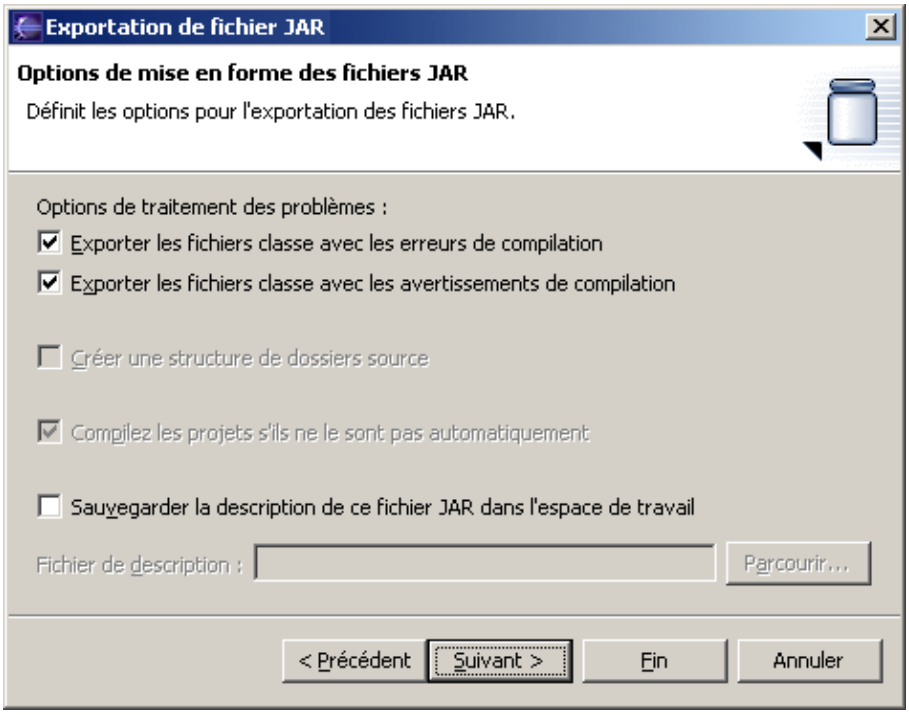


Un clic sur le bouton "Fin" permet de créer le fichier zip contenant tous les éléments sélectionnés.

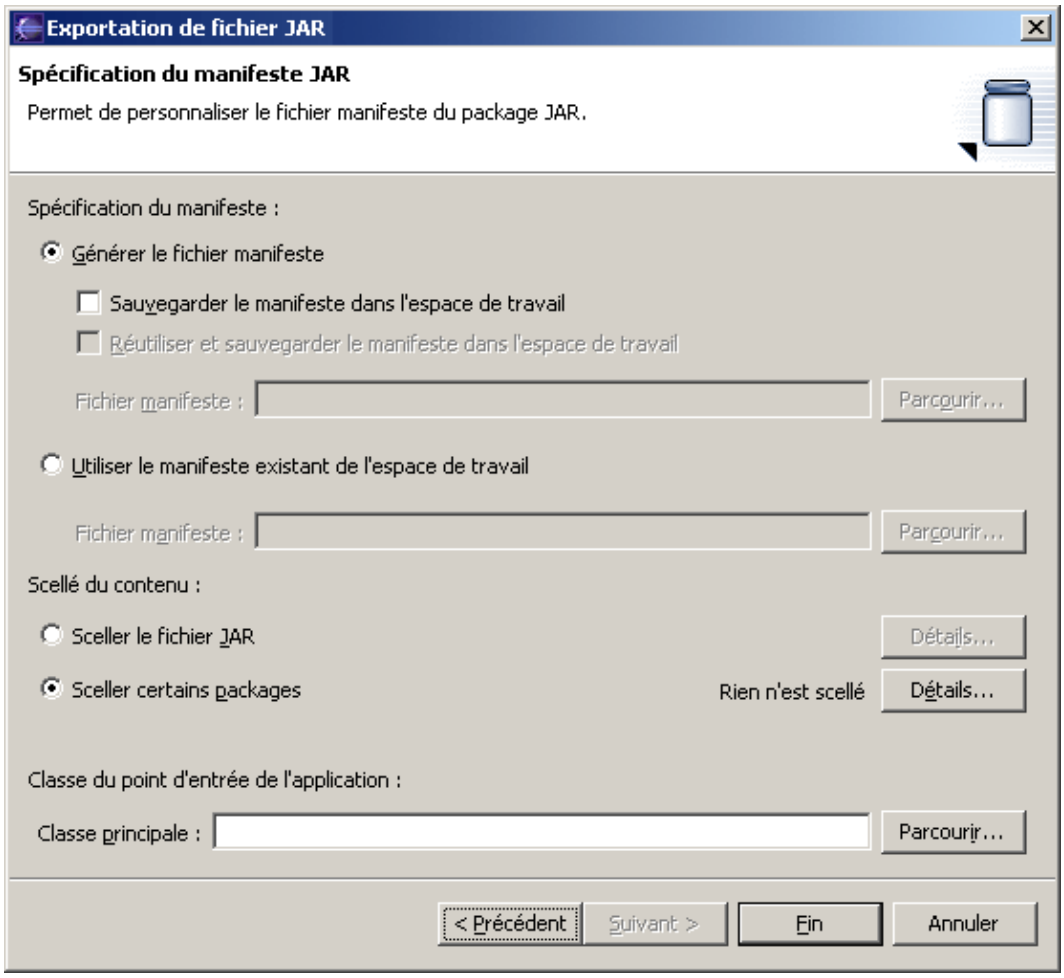
Si le format est "Fichier Jar", l'assistant demande les informations nécessaires : le ou les projets à exporter, le fichier jar à créer et les options à utiliser.



Un clic sur le bouton "Suivant" affiche une nouvelle page de l'assistant pour préciser certaines options.



Un clic sur le bouton "Suivant" affiche une nouvelle page de l'assistant pour préciser le fichier manifest.


Exportation de fichier JAR
Spécification du manifeste JAR
Permet de personnaliser le fichier manifeste du package JAR.
Spécification du manifeste :
Générer le fichier manifeste
Sauvegarder le manifeste dans l'espace de travail
Réutiliser et sauvegarder le manifeste dans l'espace de travail
Fichier manifeste :
Parcourir...
Utiliser le manifeste existant de l'espace de travail
Fichier manifeste :
Parcourir...
Scellé du contenu :
Sceller le fichier JAR
Détails...
Sceller certains packages
Rien n'est scellé
Détails...
Classe du point d'entrée de l'application :
Classe principale :
Parcourir...
< Précédent
Suivant >
Fin
Annuler

5. Les fonctions pratiques du plan de travail

Eclipse fournit dans le plan de travail plusieurs outils très pratiques qui permettent :

- d'effectuer des recherches
- de gérer une liste de tâches à faire
- de gérer une liste de signets d'éléments
- de comparer des éléments

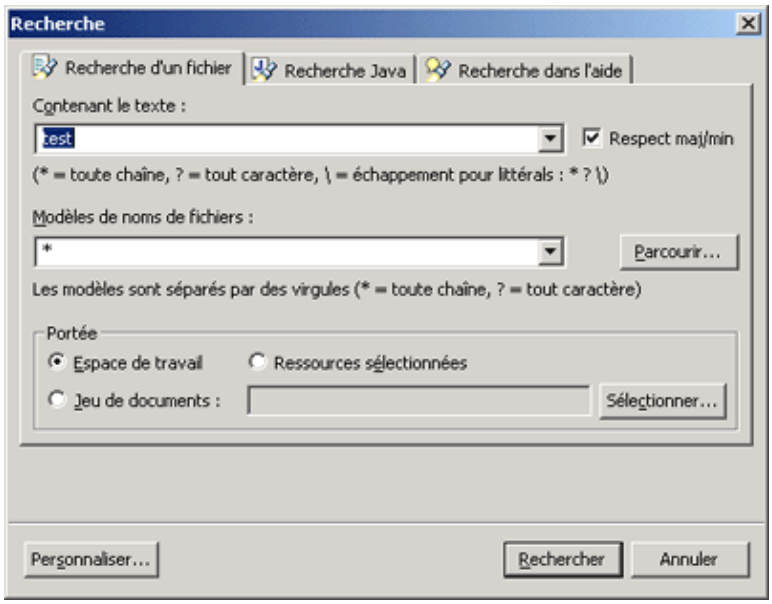
5.1. La fonction de recherche

Cette fonction de recherche permet d'obtenir une liste d'éléments qui contiennent une chaîne désignée par un motif.

Elle peut se faire dans tous les fichiers, dans les fichiers source Java ou dans l'aide en ligne.

5.1.1. La recherche dans les fichiers

Pour effectuer une recherche, il faut cliquer sur l'îcone  de la barre d'outils du plan de travail. Une boîte de dialogue permet de saisir les critères de recherche.



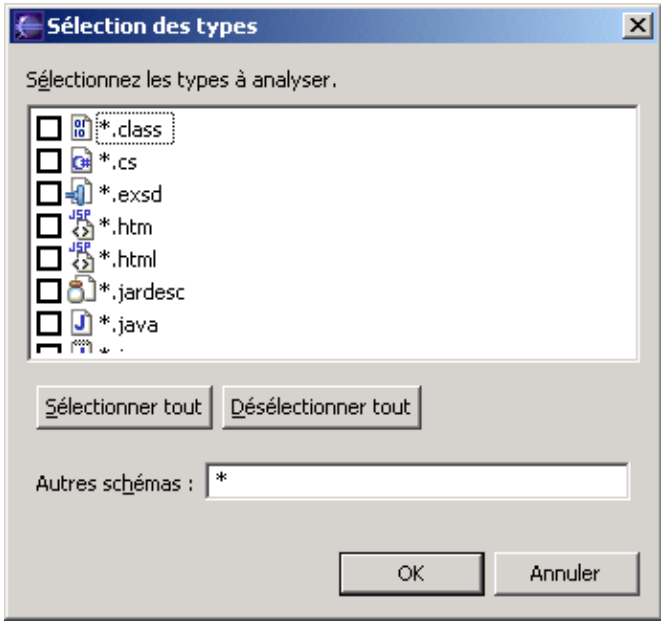
L'onglet "Recherche d'un fichier" permet de faire une recherche de fichiers contenant un texte respectant un motif. Ce motif peut être saisi ou sélectionné dans la liste déroulante à partir des précédents motifs recherchés.

Il est possible de saisir les caractères recherchés et d'utiliser trois caractères dont la signification est particulière :

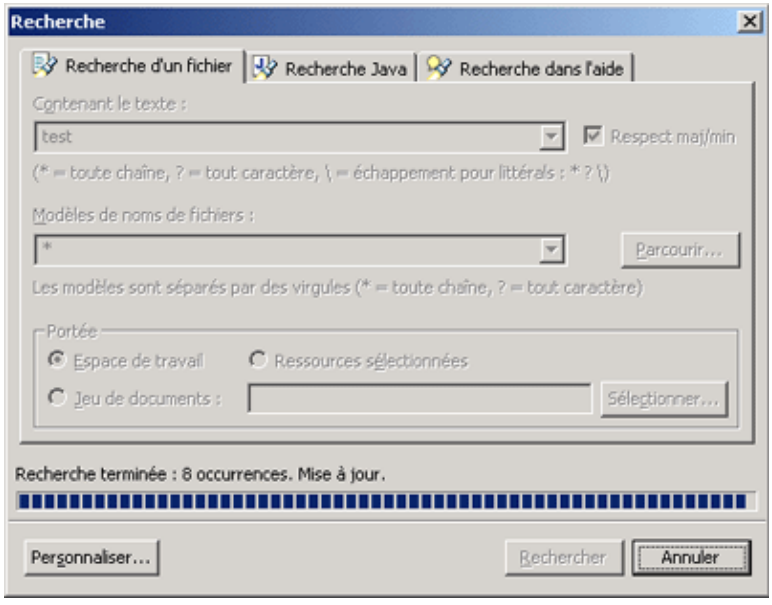
- * : représente zéro ou plusieurs caractères quelconques
- ? : représente un caractère quelconque
- \ : permet de déspécialiser le caractère *, ? et \

Il est possible de vouloir tenir compte de la casse en cochant la case "Respect maj/min".

Il est aussi possible de restreindre la recherche à certains fichiers en précisant un motif particulier. Un clic sur le bouton "Parcourir" ouvre une boite de dialogue qui permet de sélectionner un ou plusieurs types prédéfinis.

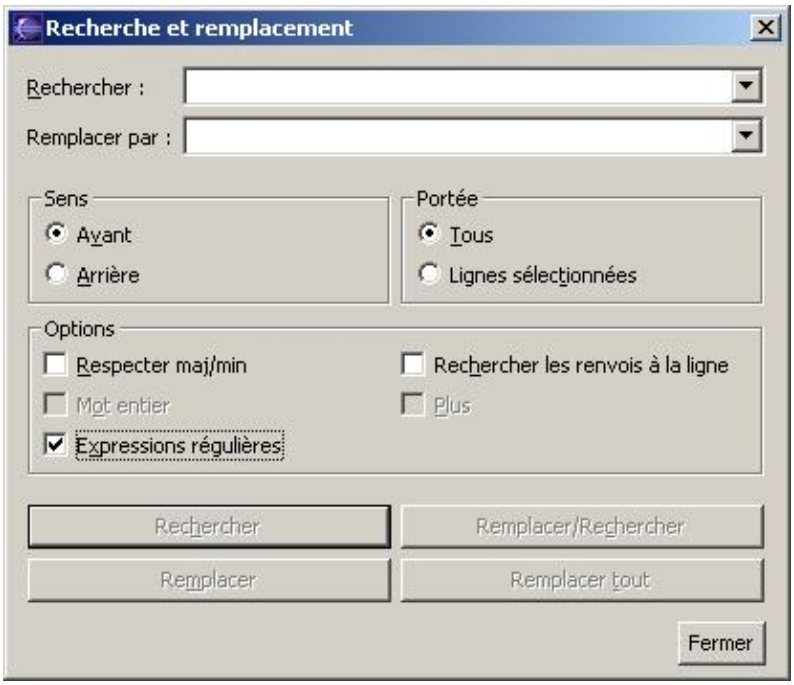


Pour lancer la recherche, il suffit de cliquer sur le bouton "Rechercher".



Une barre de progression indique l'évolution de la recherche et le nombre de fois ou le motif est trouvé. Un clic sur le bouton "Annuler" permet d'interrompre la recherche.

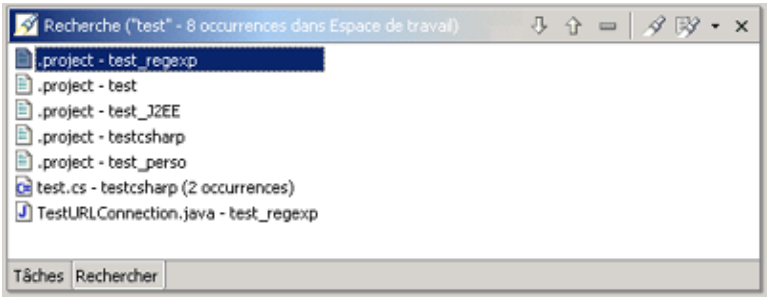
Il est possible d'utiliser les expressions régulières pour effectuer une recherche. Pour cela, il faut cocher la caser correspondante dans la boîte de dialogue.



Dans la zone de saisie du mot à rechercher, l'appui sur la combinaison de touches Ctrl+espace ouvre un assistant qui facilite la saisie d'une expression régulière.

5.1.2. L'exploitation des résultats de recherche

Une fois la recherche terminée, la vue "Recherche" affiche les éléments contenant le motif et le nombre de fois ou le motif a été trouvé dans chaque élément.

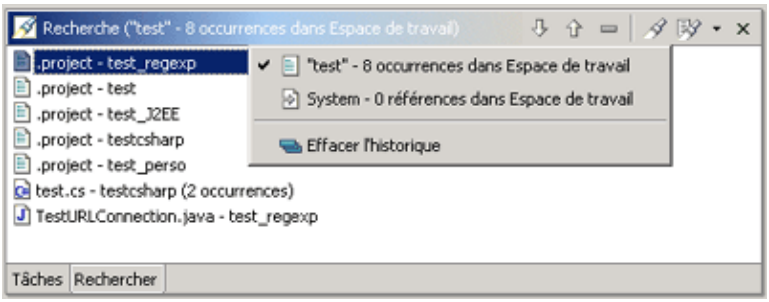


Le bouton  permet de passer à l'occurence suivante quelque soit l'élément qui la contienne. Lors du changement de l'élément qui contient l'occurence, celui ci est ouvert dans l'éditeur.

Il est possible de supprimer une ou plusieurs occurences dans la vue "Recherche". Le menu contextuel propose plusieurs options en fonction de la situation actuelle :

- "Supprimer l'occurence sélectionnée" : cette option permet de supprimer l'occurence courante de l'élément en cours
- "Supprimer les occurences en cours" : permet de supprimer toute les occurences de l'élément et l'élement de la liste
- "Supprimer toutes les occurences" : permet de supprimer tous les éléments

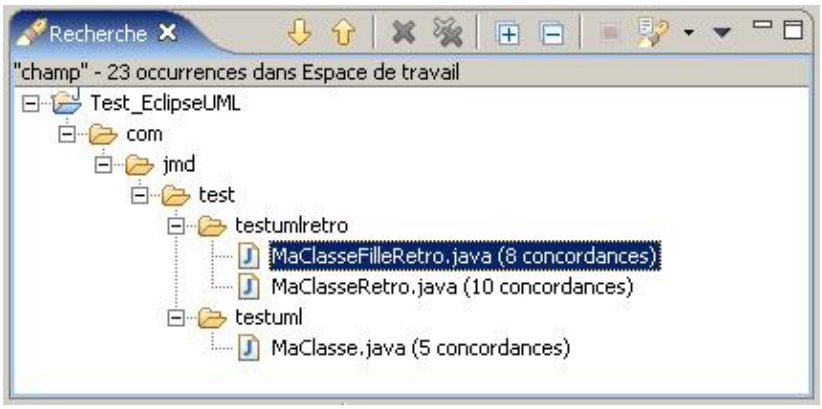
La vue "Recherche" affiche le résultat de la recherche courante mais elle mémorise aussi les précédentes recherches. Pour afficher les résultats des précédentes recherches, il suffit de sélectionner la recherche en utilisant le bouton  . Un menu affiche la liste des précédents motifs de recherche et le nombre d'occurences trouvées. La recherche courante est cochée.



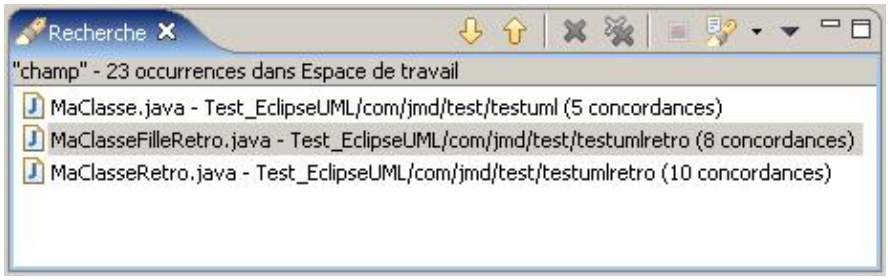
Il est toujours possible de réitérer la recherche en utilisant l'option "Nouvelle recherche" du menu contextuel de la vue.

La vue « Recherche » propose deux façons d'afficher les résultats d'une recherche :

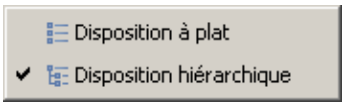
- arborescence (par défaut)



- tableau



Le bouton  ouvre un menu déroulant qui permet de sélectionner le mode d'affichage



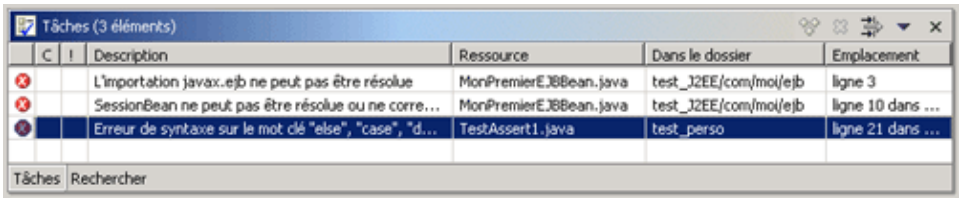
5.2. La liste des tâches

La vue "Tâches" affiche et permet de gérer une liste de tâches à faire. Ces tâches peuvent être de plusieurs types :

- des actions à réaliser
- des erreurs de compilation à corriger
- des points d'arrêt pour le débogage

Ces tâches peuvent être ou non associées à un élément de l'espace de travail. Par exemple, une erreur de compilation est associée à un fichier source.

Lorsqu'une tâche est associée à un élement, le nom de cet élément apparait dans la colonne "Ressource" et sa localisation dans l'espace de travai dans la colonne "Dans le dossier".

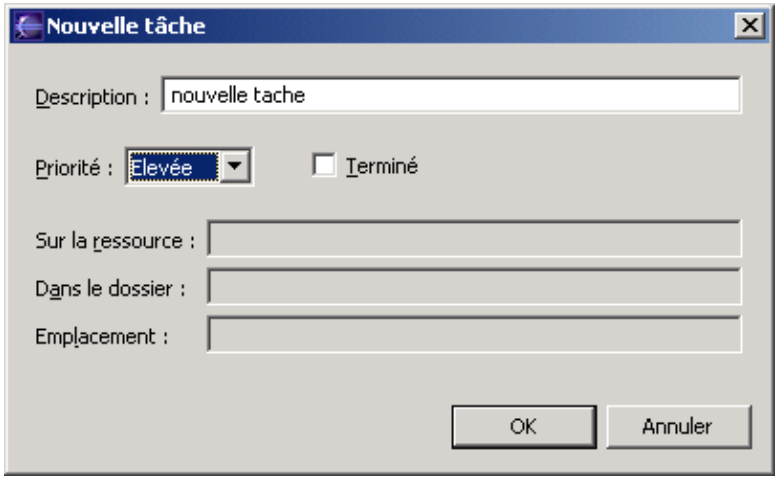


Il est possible d'accéder à l'élément associé à la tâche en double cliquant sur la tâche ou en sélectionnant l'option "Accéder à" du menu contextuel de la tâche. L'élément est ouvert dans l'éditeur avec le curseur positionné sur la ligne associée à la tâche.

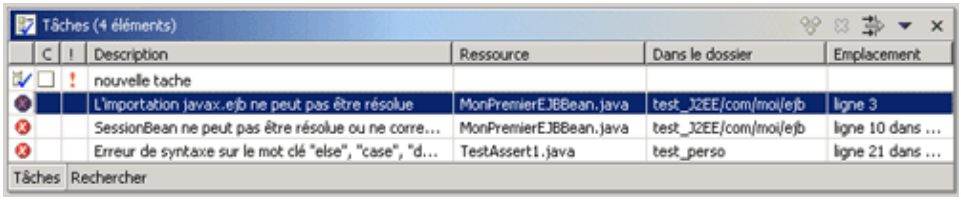
5.2.1. La création d'une tâche

Pour créer une tâche qui ne soit pas associée à un élément, il suffit de cliquer sur le bouton  de la vue.

Une boîte de dialogue permet de saisir la description et la priorité de la nouvelle tâche.



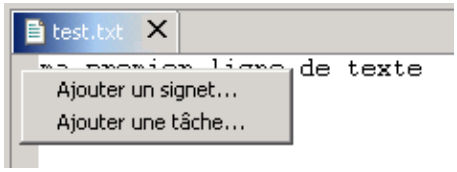
Un clic sur le bouton "OK" crée la nouvelle tâche.



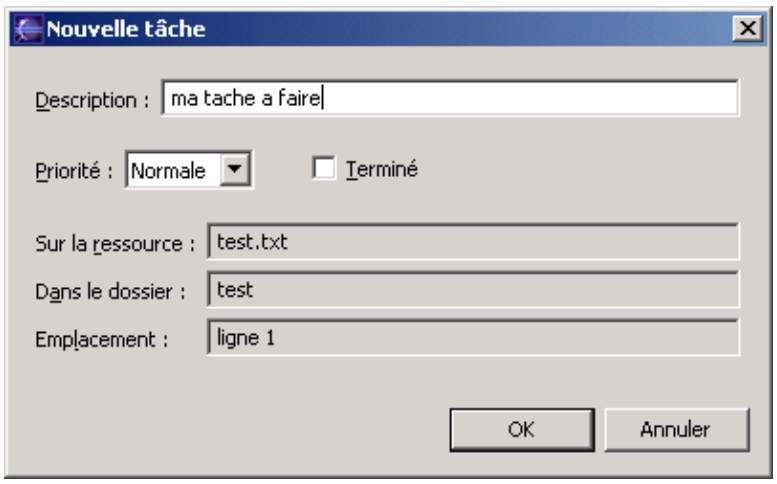
5.2.2. La création d'une tâche associée à un élément

La création d'une tâche associée à un élement ne se fait pas dans la vue "Tâches" mais directement dans un éditeur qui contient l'élément. La tâche est associée à une ligne de l'élément.

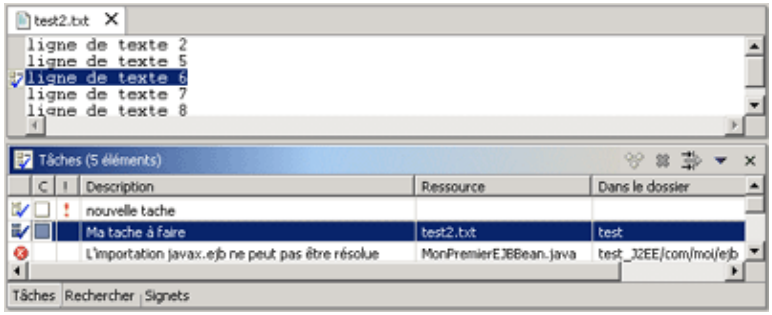
Dans la barre à gauche de l'éditeur, le menu contextuel contient l'option "Ajouter une tâche ..."



Une boite de dialogue demande de saisir la description de la tâche et de sélectionner la priorité.



Un clic sur le bouton "OK" crée la tâche et une marque particulière apparait sur la ligne concernée dans la barre de gauche de l'éditeur.



Cette marque reste associée à la ligne même si la position de la ligne dans le fichier change (par ajout ou suppression de lignes dans la ressource).

5.2.3. La suppression d'une tache associée à un élément

Il suffit de sélectionner l'option "Supprimer une tâche" du menu contextuel associé à la marque de la tâche. La marque disparait et la tâche est supprimée de la liste.

5.3. La liste des signets

Les signets (bookmarks) permettent de maintenir une liste d'éléments particuliers dans le but est de permettre d'y accéder rapidement. Pour afficher la vue "Signets", il faut sélectionner l'option "Afficher la vue / Signets" du menu "Fenêtre" du plan de travail.

A partir de la vue "Signets", pour ouvrir un élément dans l'éditeur, il y a trois possibilités :

- double cliquer sur le signet
- sélectionner l'option "Accéder à" du menu contextuel associé au signet
- cliquer sur le bouton  une fois le signet sélectionné

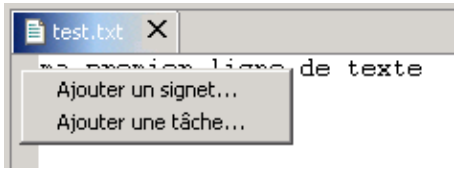
Il est aussi possible à partir d'un signet de sélectionner l'élément dans la vue "Navigateur" en utilisant l'option "Afficher dans le navigateur" du menu contextuel.

5.3.1. La création d'un signet

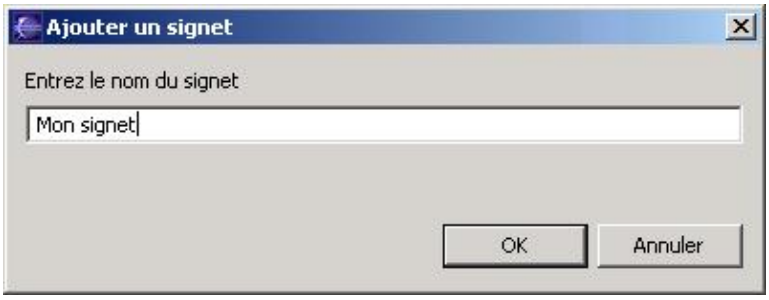
Un signet peut concerner un élément (un fichier) ou plus précisement une composante de cet élément (une position particulière dans le fichier).

Pour créer un signet sur un élement, il suffit de le sélectionner dans la vue "Navigateur" et de sélectionner l'option "Ajouter un signet" du menu contextuel.

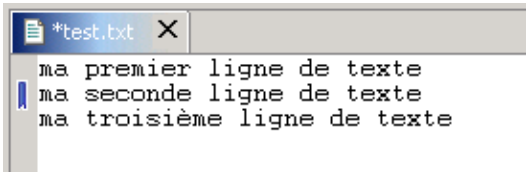
Pour créer un signet sur une ligne de l'élément, il suffit de positionner le curseur sur la ligne désirée dans l'éditeur et de sélectionner l'option "Ajouter un signet" du menu contextuel de la barre de gauche de l'éditeur.



Une boite de dialogue demande de saisir la description.



Le signet est ajouté dans la liste des signets et une marque est affichée dans la barre de gauche de l'éditeur sur la ligne concernée.



5.3.2. La suppression d'un signet

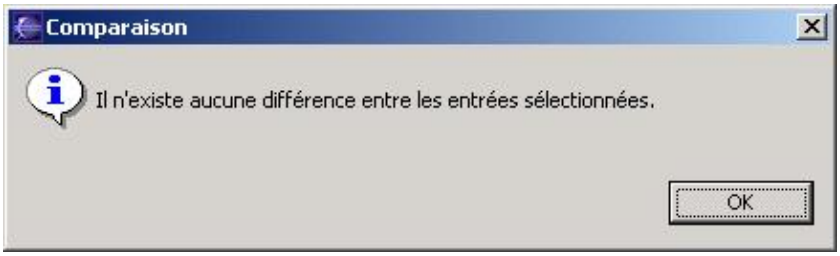
Pour supprimer un signet, il y a trois possibilités :

- dans la vue "Signets", sélectionner le signet et cliquer sur le bouton 
- dans la vue "Signets", sélectionner le signet et l'option "Supprimer" du menu contextuel associé au signet
- dans l'éditeur, sélectionner l'option "Supprimer un signet" du menu contextuel associé à l'icône du signet dans la barre de gauche

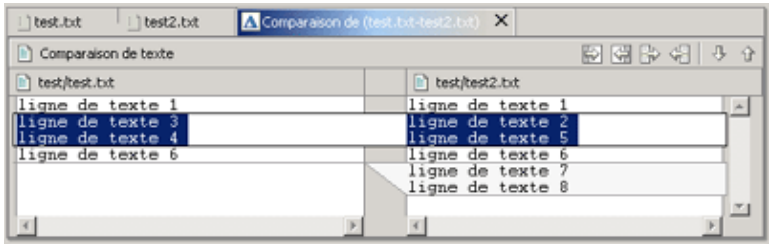
5.4. La comparaison d'éléments

Le plan de travail dispose d'un outil pratique pour comparer le contenu de deux éléments. Pour réaliser cette comparaison, il faut sélectionner ces deux éléments en maintenant la touche Ctrl enfoncée dans la vue "Navigateur" et sélectionner l'option "Comparer / Réciproquement" du menu contextuel.

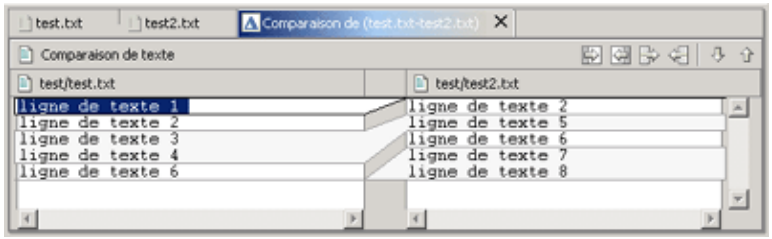
Si les deux fichiers sont identiques, une boîte de dialogue s'affiche :



Si les deux fichiers possèdent des différences, un éditeur particulier s'ouvre. Cet éditeur spécial pour les comparaisons, affiche chaque ligne des deux fichiers dans deux colonnes. Une colonne centrale permet de voir de façon graphique les différences grâce à des lignes.



Dans la barre centrale, les lignes en gris foncé sont identiques, les lignes en blanc sont des différences entre les deux fichiers.



La vue de comparaison de fichier contient une barre d'outils qui permet de naviguer dans les différences et de les reporter pour effectuer une synchronisation sélective.

La flèche vers le haut et le bas permet de naviguer dans les différences respectivement la suivante et la précédente.

Les quatre premiers boutons permettent respectivement :

- de copier tout le document de gauche dans le document de droite
- de copier tout le document de droite dans le document de gauche
- de reporter la différence courante de gauche dans le document de droite
- de reporter la différence courante de droite dans le document de gauche

Partie 2 : le développement en Java

Cette seconde partie est chargée de présenter les bases de l'utilisation d'Eclipse pour le développement avec Java.

Elle comporte les chapitres suivants :

- Le Java Development Tooling (JDT) : Détaille le JDT qui fournit des outils pour permettre le développement avec Java
- Déboguer du code Java : détaille la perspective Débogage dédiée à la recherche et à la correction des bugs.
- Le refactoring : détaille les puissantes fonctionnalités de refactoring proposées par Eclipse
- Ant et Eclipse : présente l'utilisation de l'outil Ant avec Eclipse
- JUnit et Eclipse : présente l'utilisation de JUnit avec Eclipse pour réaliser l'automatisation des tests unitaires.

6. Le Java Development Tooling (JDT)

Le Java Development Tooling (JDT) est inclus dans Eclipse pour fournir des outils de développement en Java. Il inclus plusieurs plug−ins et apporte :

- les perspectives "Java" et "Navigation Java"
- les vues "Packages" et "Hierarchie"
- les éditeurs "Java" et "Scrapbook"
- les assistants : pour créer de nouveaux projets, packages, classes, interfaces, ...

Dans l'espace de travail, il définit un projet de type particulier pour les projets Java. L'arborescence de ces projets contient un fichier particulier nommé .classpath qui contient la localisation des bibliothèques utiles à la compilation et à l'exécution du code.

6.1. Les projets de type Java

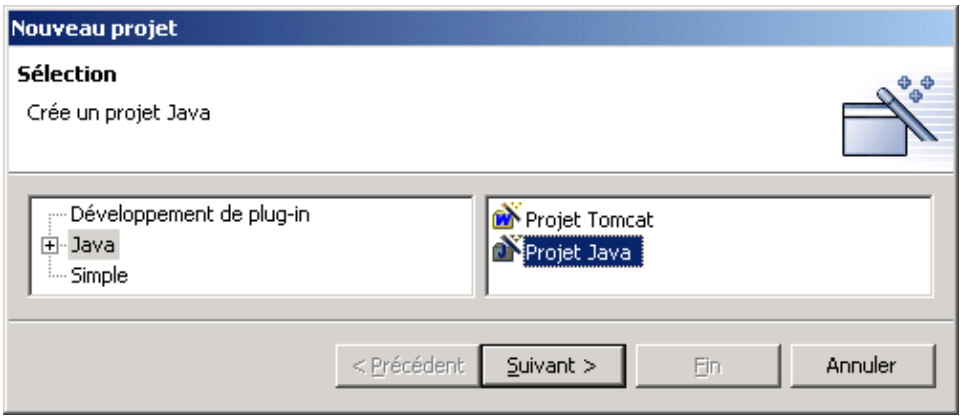
Pour pouvoir développer des entités en Java, il faut les regrouper dans un projet de type Java.

6.1.1. La création d'un nouveau projet Java

Dans la perspective "Java", il y a plusieurs possibilités pour lancer l'assistant de création d'un nouveau projet :

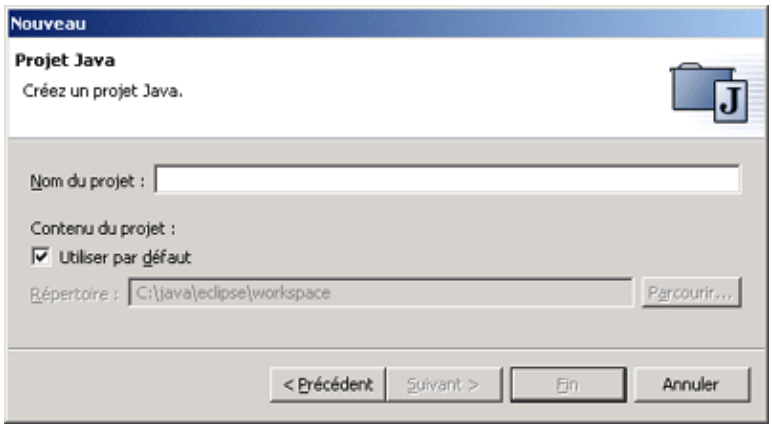
- sélectionner l'option "Projet" du menu "Fichier/Nouveau"
- sélectionner l'option "Nouveau/Projet" du menu contextuel de la vue "Packages"

L'assistant demande le type de projet à créer.

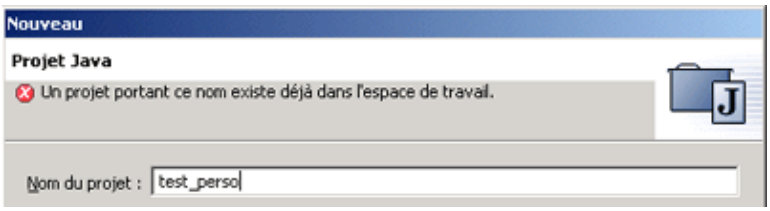


Pour demander directement la création d'un projet "Java", il suffit de cliquer sur l'îcone  de la barre d'outils.

L'assistant demande le nom du projet.

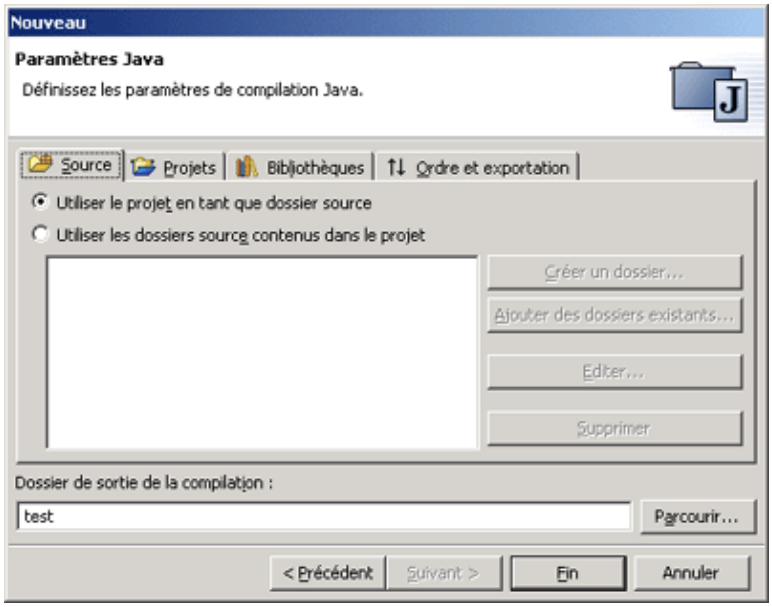


Ce nom de projet ne doit pas déjà être utilisé dans l'espace de travail courant sinon un message d'erreur est affiché.



En cliquant sur le bouton "Fin", le projet est créé avec des paramètres par défaut.

Pour modifier certains paramètres avant la création du projet, suffit de cliquer sur le bouton "Suivant" :

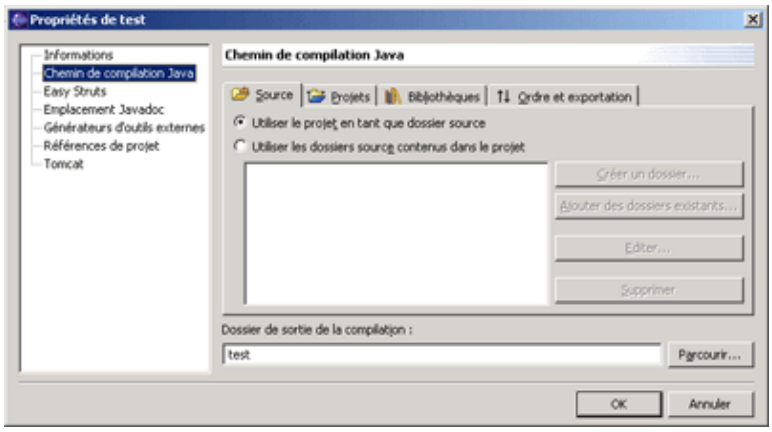


La modification de ces paramètres sera détaillée dans la section suivante. Une fois les paramètres modifiées, cliquer sur le bouton "Fin". Le projet apparait dans la vue "Packages" de la perspective.

6.1.2. Les paramètres d'un projet Java

Les principaux paramètres d'un projet peuvent être modifiés :

- lors de l'utilisation de l'assistant à la création du projet
- en sélectionnant le menu contextuel "Propriétés" sur le projet sélectionné dans la vue "Packages" et de choisir "Chemin de compilation Java"



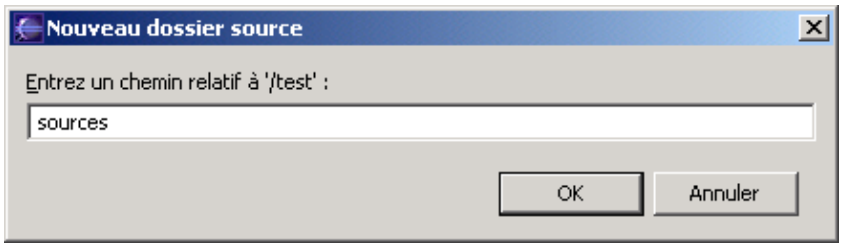
Les propriétés "Chemin de compilation Java" sont regroupées dans quatres onglets :

Onglet	Rôle
Source	Permet de préciser le répertoire qui va contenir les sources et celui qui va contenir le résultat des compilations
Projets	Permet d'utiliser d'autre projet avec le projet courant
Bibliothèques	Permet d'ajouter des bibliothèques au projet
Ordre et exportation	Permet de préciser l'ordre des ressources dans la classpath

L'onglet "Source" permet de préciser le répertoire qui va contenir les sources : par défaut, c'est le répertoire du projet lui même (l'option "utiliser le dossier projet en tant que dossier source" est sélectionné).

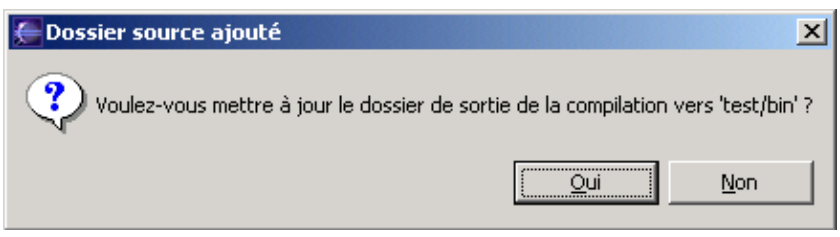
Pour stocker les ressources dans un répertoire dédié, il faut sélectionner l'option "Utiliser les dossiers sources contenus dans le projet". La liste permet de sélectionner le ou les répertoires.

Le bouton "Créer un dossier" ouvre une boîte de dialogue qui demande le nom du répertoire.



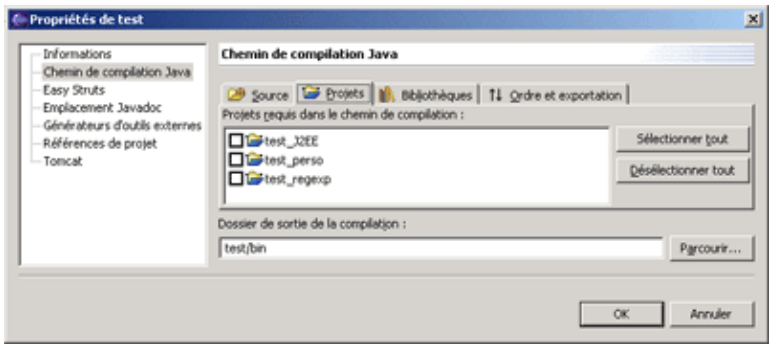
Il suffit de saisir le nom, par exemple "sources" et cliquer sur le bouton "OK"

Par défaut, dès qu'un premier répertoire contenant les sources est sélectionné, Eclipse propose de créer un répertoire bin qui va contenir le résultat des différentes compilations.



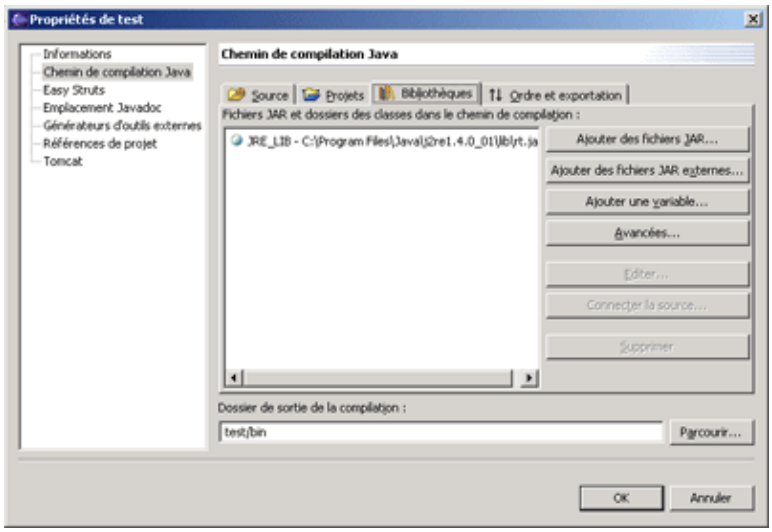
La réponse à la question est libre mais il est préférable de répondre "Oui".

L'onglet "Projets" permet d'ajouter des projets contenus dans l'espace de travail au classpath.



Il suffit de cocher les projets à inclure dans le classpath.

L'onglet "Bibliothèques" permet d'ajouter des bibliothèques externes au projet notamment des fichiers .jar.

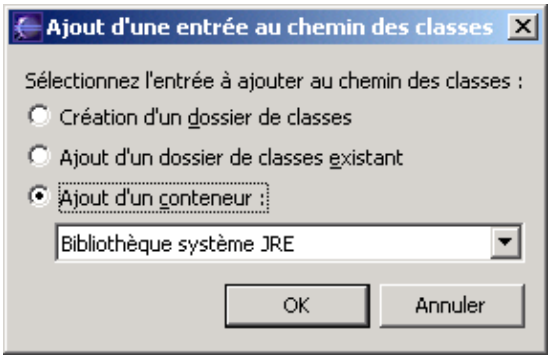


Les bibliothèques incluses dans le classpath du projet courant sont affichées dans la liste.

Pour ajouter une nouvelle bibliothèque contenue dans l'espace de travail, il suffit de cliquer sur "Ajouter des fichiers jar". Pour ajouter des fichiers jar qui ne sont pas contenus dans l'epsace de travail, il suffit de cliquer sur le bouton "Ajouter des fichiers jar externes".

Une boite de dialogue permet de sélectionner le fichier jar. En cliquant sur le bouton "Ouvrir", le fichier jar est ajouté dans la liste.

Le bouton "Avancées ... " permet d'ajouter d'autres entités au classpath notamment des répertoires qui contiennent des fichiers compilés.



Le bouton "Editer" permet de modifier les caractéristiques de la bibliothèque (son chemin d'accès dans le cas d'un fichier jar).

Le bouton "Supprimer" permet de supprimer une bibliothèque du classpath.

L'onglet "Ordre et exportation" permet de modifier l'ordre des bibliothèques dans le classpath, de préciser la cible des éléments générés (le répertoire qui va les contenir) et de définir les ressources qui seront utilisables par les autres projets de l'espace de travail lorsque le projet sera lié avec eux.

6.2. La création d'entité

Dans un projet Java, il est possible de créer différentes entités qui entrent dans sa composition : les packages, les classes et les interfaces.

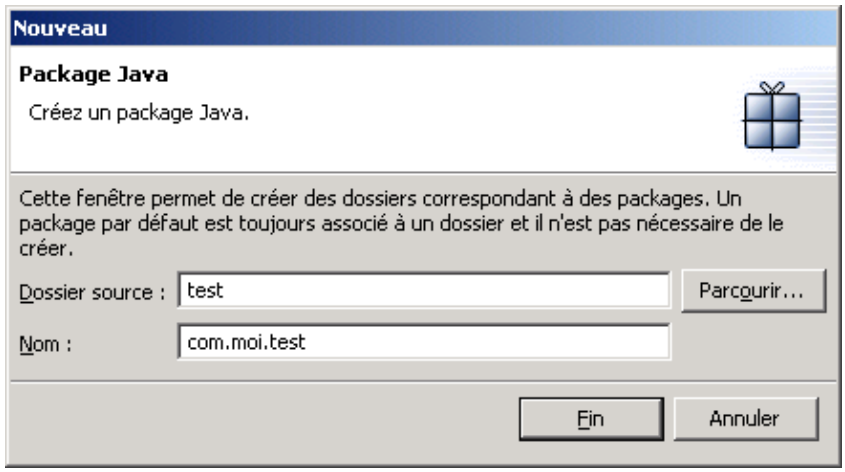
6.2.1. Les packages

Il est possible de créer les packages à l'avance même si ceux ci peuvent être créer automatiquement en même temps qu'une classe qui la contient.

Pour créer un nouveau package, il y a plusieurs possibilités :

- cliquer sur la flèche de l'icône  de la barre d'outils de la perspective "Java" et sélectionner "Package"
- cliquer sur l'icône  de la barre d'outils de la perspective "Java"
- sélectionner l'option "Package" du menu "Fichier / Nouveau"

L'assistant demande le nom du package.



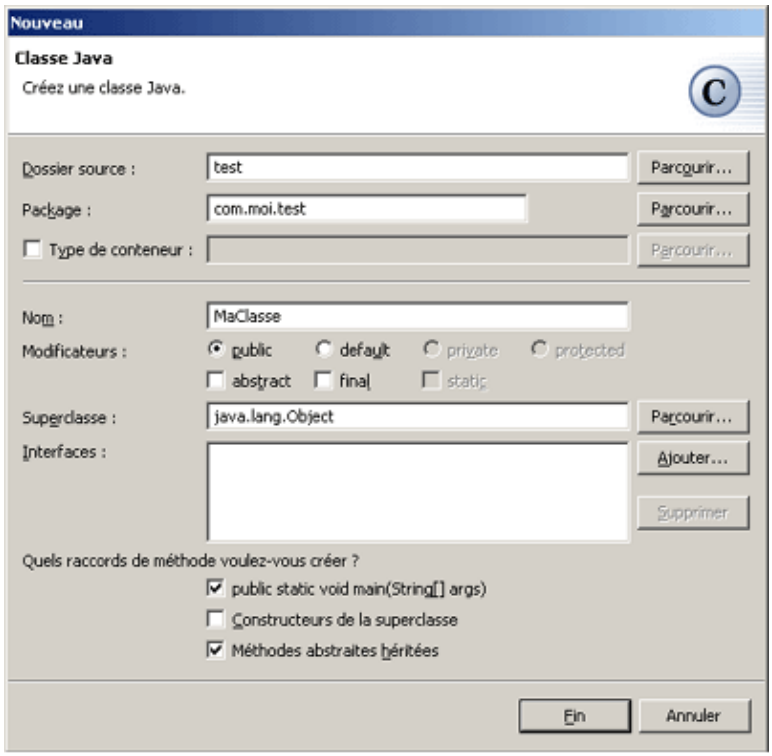
Cliquez sur le bouton "Fin", pour créer le nouveau package. Le package apparait dans la vue "Packages".

6.2.2. Les classes

La création d'une classe peut se faire :

- soit en cliquant sur l'îcône  dans la barre d'outils
- soit en sélectionnant l'option Classe du menu " Fichier/Nouveau "

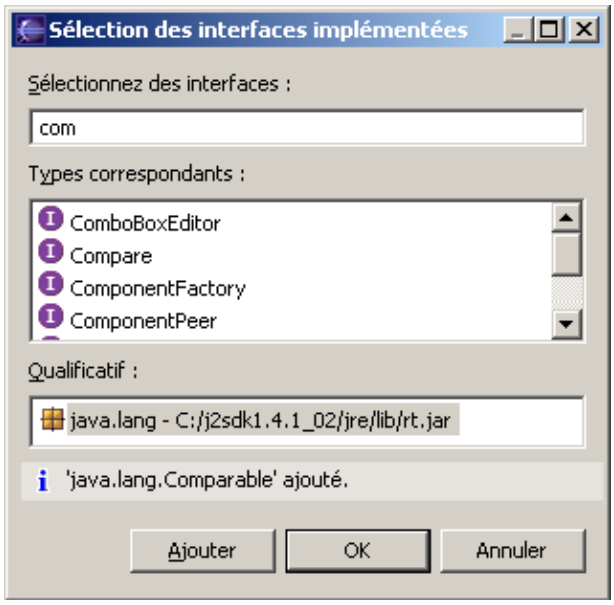
Un assistant facilite la création de la classe.



L'assistant demande de renseigner les différentes caractéristiques de la nouvelle classe : le projet et le package d'appartenance, le nom, les modificateurs, la classe mère, les interfaces implémentées. Enfin, il est possible de demander à l'assistant de générer certaines méthodes.

Si un projet ou un package est sélectionné dans la vue "Packages", celui ci est automatiquement repris par l'assistant.

L'ajout d'une interface implémentée se fait en la sélectionnant dans une liste.



Pour ajouter une interface, il suffit de double cliquer dessus ou de la sélectionner et d'appuyer sur le bouton "Ajouter". Une fois toutes les interfaces ajoutées, il suffit de cliquer sur le bouton "OK".

Toutes les méthodes définies dans la ou les interfaces sélectionnées seront présentes dans le code source de la classe générée. Si le package saisi n'existe pas dans le projet, celui ci sera créé en même temps que la classe.

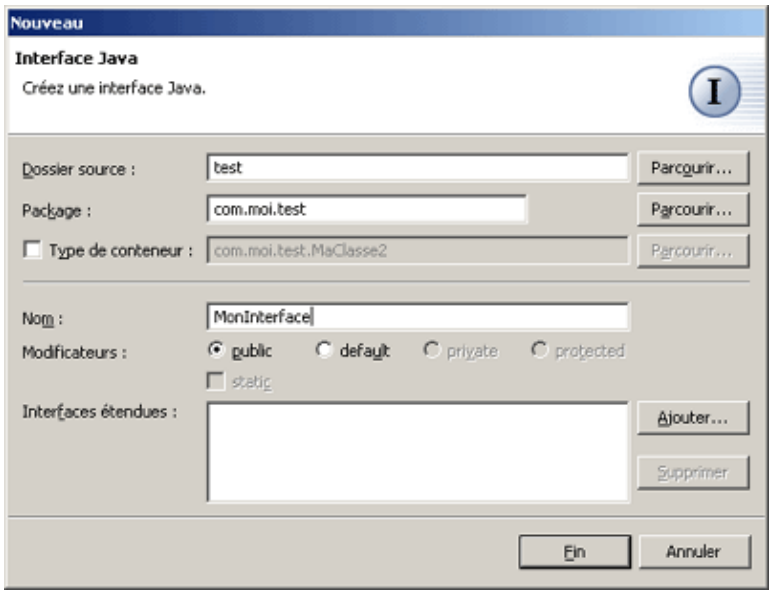
Une fois toutes les données utiles renseignées, il suffit de cliquer sur le bouton " Fin " pour que la classe soit générée et que l'éditeur s'ouvre avec le contenu de son code source.

6.2.3. Les interfaces

La création d'une interface peut se faire :

- en cliquant sur l'icône  dans la barre d'outils
- en sélectionnant l'option Interface du menu " Fichier/Nouveau "

Un assistant facilite la création de l'interface.



L'assistant demande de renseigner les différentes caractéristiques de la nouvelle interface : le projet et le package d'appartenance, le nom, les modificateurs ainsi que les éventuelles interfaces héritées.

Une fois toutes les données utiles renseignées, il suffit de cliquer sur le bouton " Fin " pour que l'interface soit générée et que l'éditeur s'ouvre avec le contenu de son code source.

6.3. Les vues du JDT

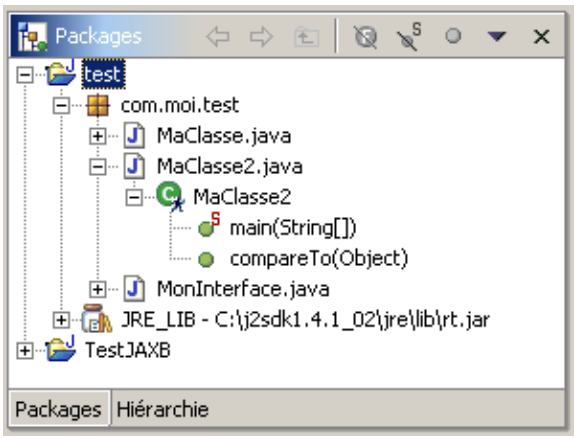
Le JDT contient les vues "Packages" et "Hiérarchie"

6.3.1. La vue "Packages"

Cette vue permet d'afficher de façon arborescente le contenu des différents packages définis et utilisés dans chaque projet ainsi que les bibliothèques utilisées par le projet.

Pour les éléments contenant du code source, l'arborescence sous jacente permet de voir les différents membres qui composent l'élément.

Un double clic sur des éléments de l'arborescence, permet d'ouvrir l'éditeur directement sur l'élément sélectionné.



Chaque élément de l'arborescence possède une petite icône en fonction en son type :

Icône	Type de l'élément
	Projet de type Java
	Package
	Elément Java : classe ou interface
	Interface Java
	Classe public Java
	Classe Java pouvant être exécutée (possédant une méthode main())
	Classe protected
	Classe package friendly
	Classe private
	Champ public
	Champ private

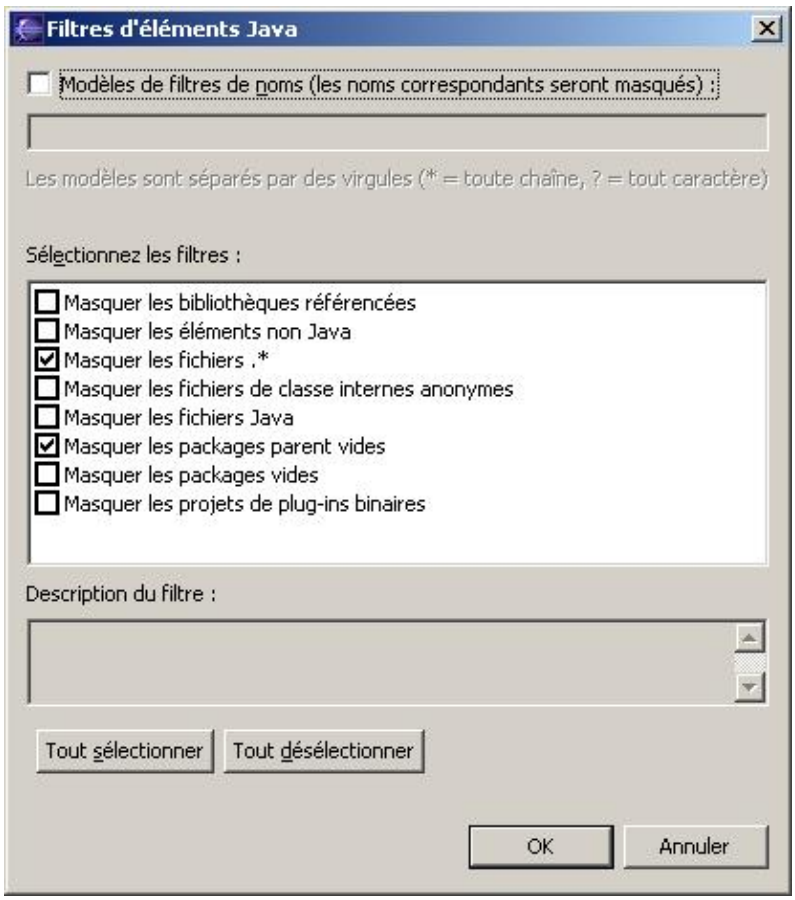
	Champ protected
	Champ package friendly
	Méthode public
	Méthode private
	Méthode protected
	Méthode package friendly

Le bouton  permet de masquer les champs définis dans les éléments Java.

Le bouton  permet de masquer les membres statiques.

Le bouton  permet de masquer tous les membres qui ne sont pas publics.

Il est possible de restreindre les entités affichées dans la vue package. Il suffit de cliquer sur bouton  et de sélectionner l'option " Filtres ".



Il suffit de cocher les filtres qui doivent être appliqués.

A partir de l'éditeur, il est possible de sélectionner dans la vue "Package", l'élément en cours d'édition en utilisant l'option "Afficher dans la vue package" du menu contextuel.

6.3.2. La vue "Hiérarchie"

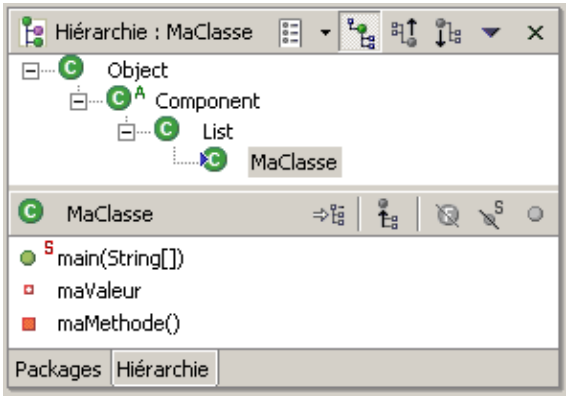
Cette vue affiche la hiérarchie d'un élément.

Pour afficher la hiérarchie d'un élément, il y a plusieurs possibilités :

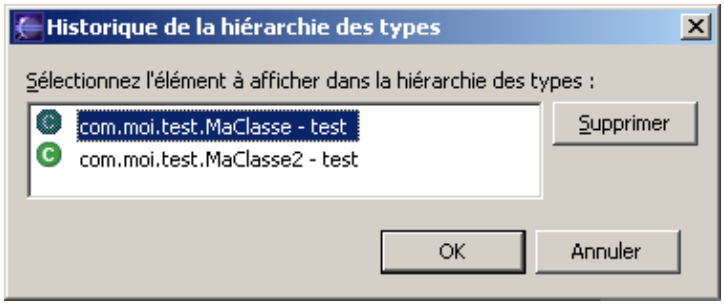
- dans la vue Package sélectionner l'élément et utiliser l'option "Ouvrir la hiérarchie des types" du menu contextuel.
- dans l'éditeur, utiliser le menu contextuel "Ouvrir la hiérarchie des types"

Elle se compose de deux parties :

- une partie supérieure qui affiche la hiérarchie de l'élément.
- une partie inférieure qui affiche la liste des membres de l'élément

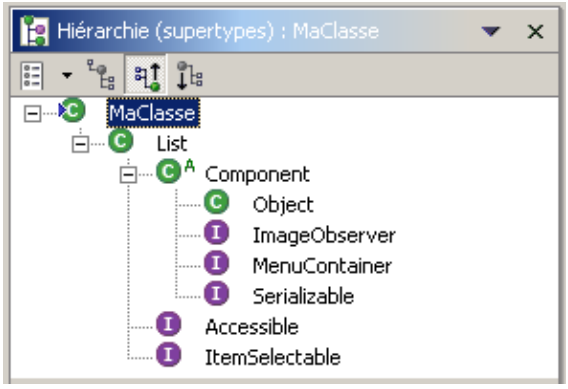


Le bouton  permet de sélectionner dans un historique un élément qui a déjà été affiché dans la vue.



Il suffit de sélectionner l'élément concerné et de cliquer sur le bouton "OK".

Le bouton  permet d'afficher la hiérarchie des classes mères et des interfaces qu'elles implémentent de l'élément courant.



Le bouton menu permet de changer la présentation de la vue :

-  : les deux parties sont affichées horizontalement
- : les deux parties sont affichées verticalement
-  : n'affiche que la partie qui présente la hiérarchie

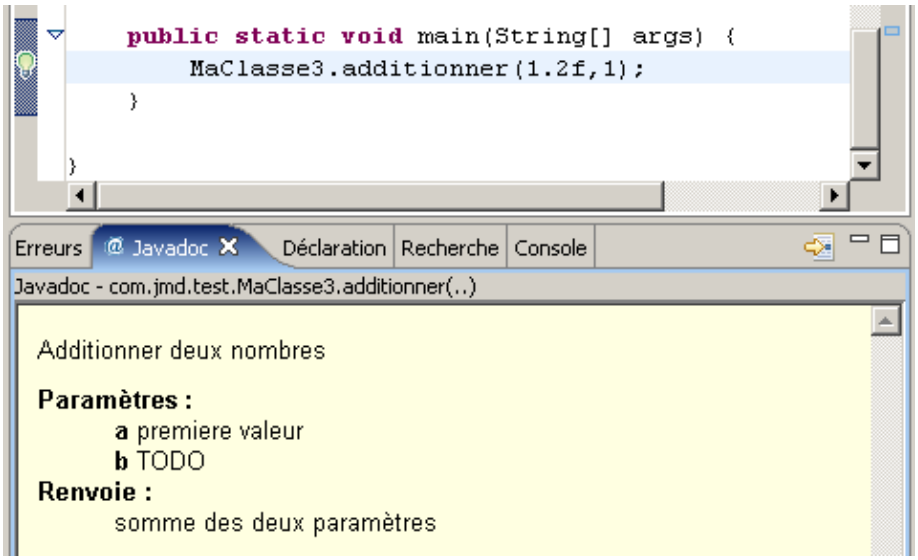
Le bouton  permet de masquer les champs

Le bouton  permet de masquer les membres statiques.

Le bouton  permet de masquer tous les membres qui ne sont pas publics.

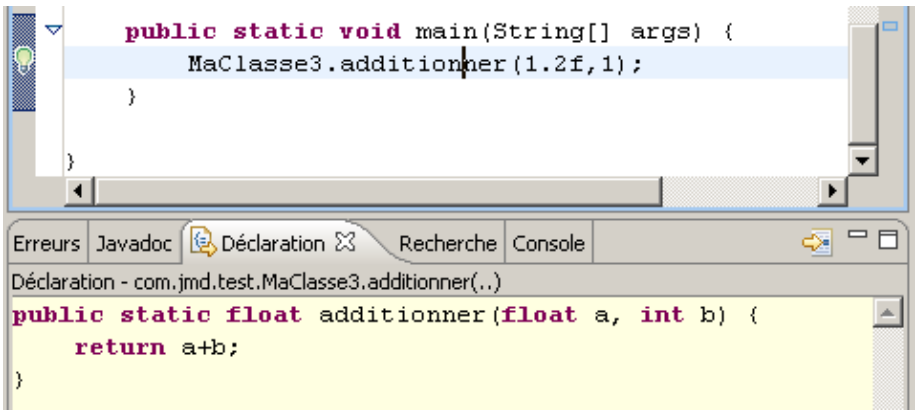
6.3.3. La vue "Javadoc"

La vue "Javadoc" permet d'afficher la documentation de l'élément sélectionné dans la perspective "Java".



6.3.4. La vue "Déclaration"

La vue "Déclaration" permet d'afficher le code de l'élément sélectionné dans la perspective "Java".



6.4. L'éditeur de code

Le JDT propose un éditeur dédié au fichier contenant du code Java. Il propose des fonctionnalités particulièrement pratiques pour le développement de code Java notamment :

- la coloration syntaxique
- la complétion de code
- le formatage du code source

- l'importation et l'exportation de code via un assistant
- une forte synergie avec le débogueur

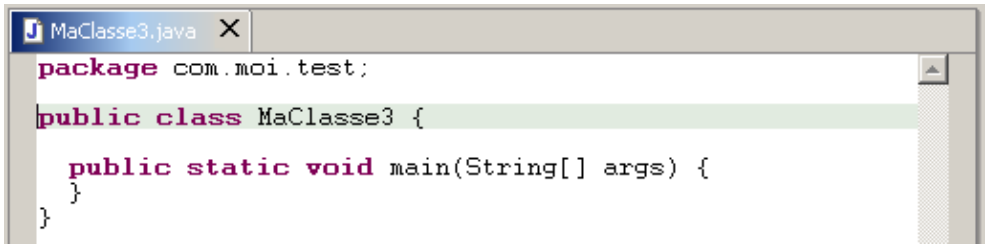
Pour ouvrir un élément dans l'éditeur, il y a deux façons principales :

- double cliquer sur un élément dans la vue "Navigateur"
- double cliquer sur un élément dans la vue "Packages"

L'éditeur peut être invoqué sur un fichier .java ou un fichier .class. Dans le cas d'un fichier .class, si le fichier source est disponible dans l'IDE, alors l'éditeur affiche le contenu du code source.

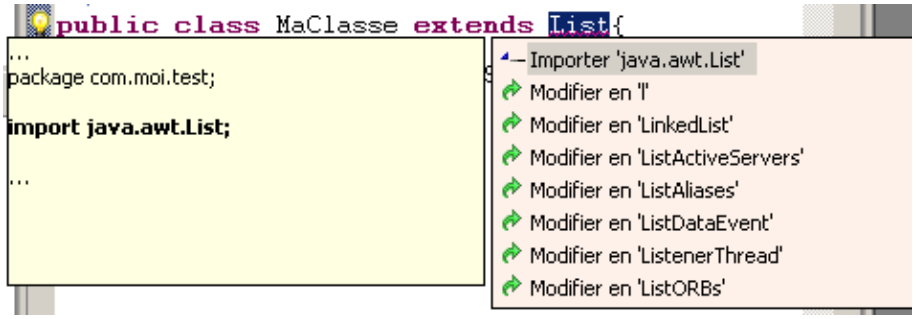
6.4.1. Utilisation de l'éditeur de code

La ligne ou se situe le curseur apparaît en gris.



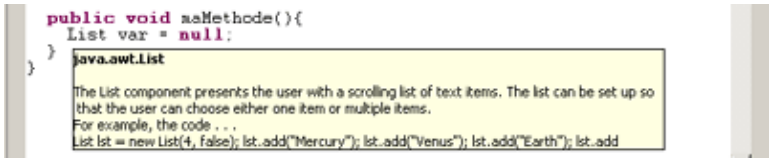
De chaque côté de la zone d'édition, il y a une colonne qui peut contenir de petites icônes pour fournir des informations à l'utilisateur.

Par exemple, si l'on fait hériter une classe d'une classe dont le package n'est pas importé, un clic sur la petite ampoule jaune permet d'obtenir des propositions de corrections.



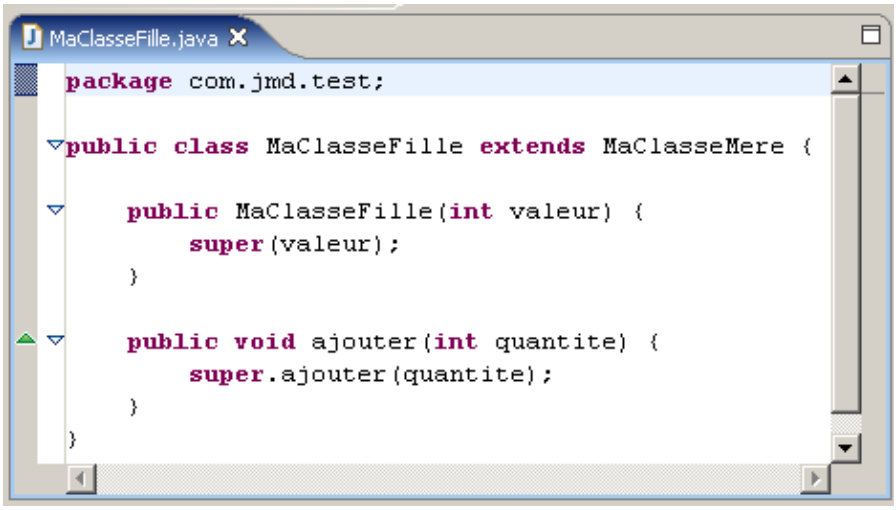
Il suffit de sélectionner une des actions proposées dans la liste pour que celle ci soit automatiquement mise en oeuvre. Un aperçu des modifications impliquées par l'action sélectionnée est affiché dans une bulle d'aide.

Le bouton  de la barre d'outils permet, s'il est sélectionné, d'afficher une bulle d'aide contenant des informations sur l'élément sous lequel est le curseur.

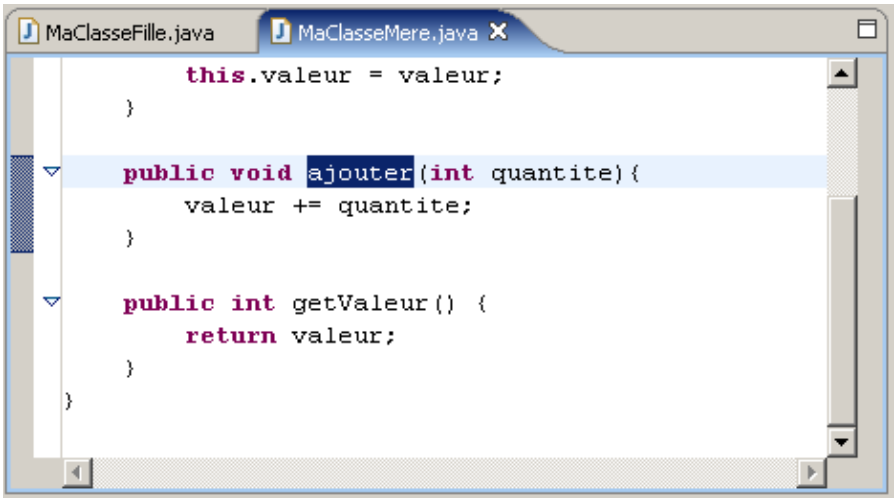


Une description plus détaillée peut être obtenue en positionnant le curseur sur l'élément et en appuyant sur la touche F2 ou en sélectionnant l'option " Afficher une description de type infobulles" du menu " Editer ".

Les méthodes héritées qui sont réécrites sont signalées par une petite icône.

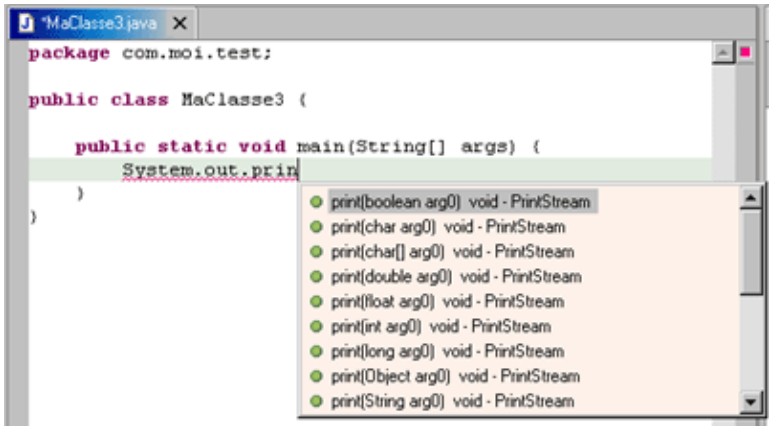


Un clic sur cette petite icône permet d'ouvrir l'éditeur sur la méthode de la classe mère.

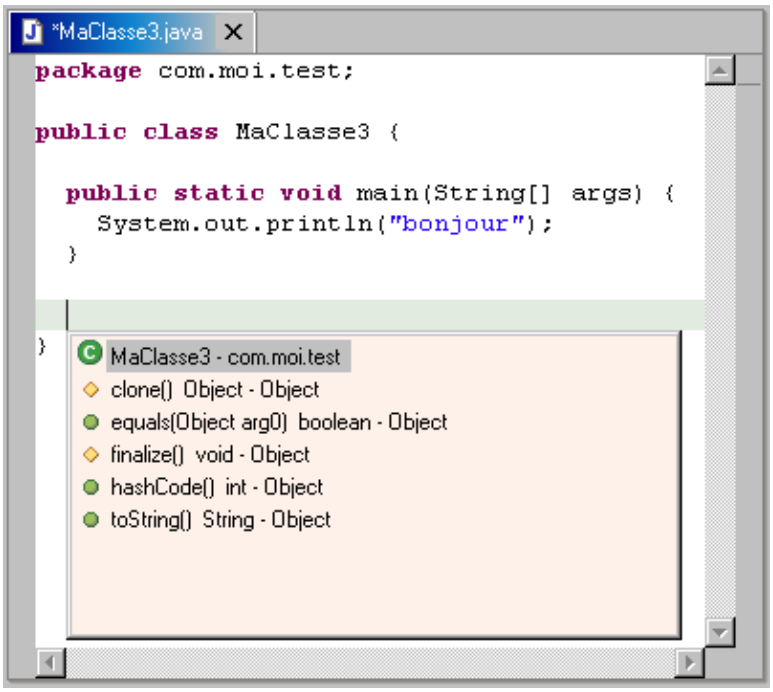


6.4.2. Complétion de code

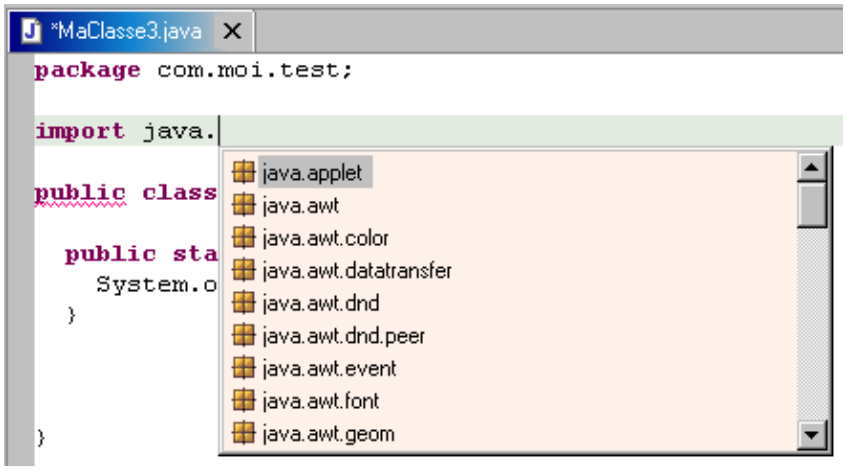
La complétion de code permet de demander à l'IDE de proposer des suggestions pour terminer le morceau de code en cours d'écriture. Dans l'éditeur Eclipse, pour l'activer, il suffit d'appuyer sur les touches Ctrl et espace en même temps.



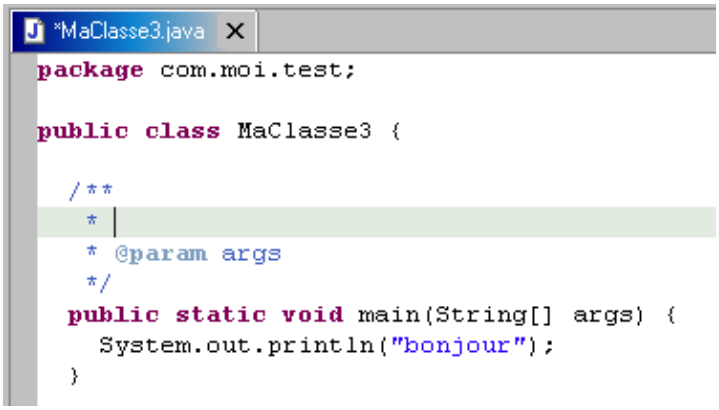
Cette fonction peut être appelée alors qu'aucune ou une partie du code à compléter est saisie.



La complétion de code s'adapte au contexte dans lequel elle est appelée. Par exemple, elle peut être appelée pour compléter une clause d'importation. Dans ce cas, elle propose une liste de packages en tenant compte du code déjà saisi.

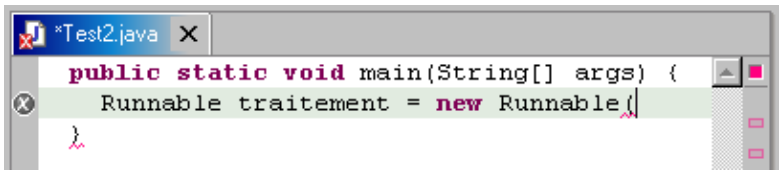


L'éditeur peut générer la structure d'un commentaire Javadoc dans le source. Par exemple, avant une méthode, il suffit de saisir /** puis d'appuyer sur la touche " Entrée ".

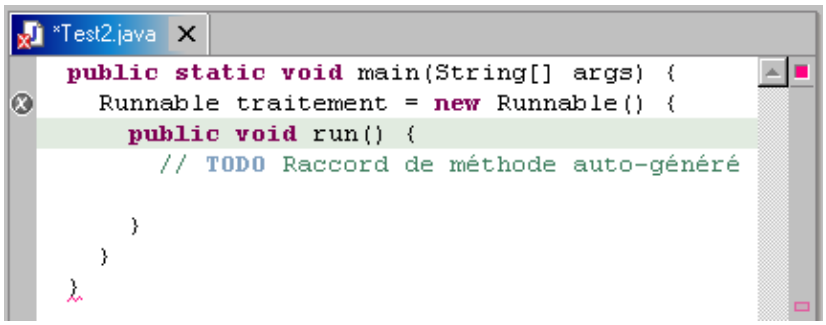


L'appel de la complétion de code en appuyant sur les touches Ctrl + Espace permet aussi de faciliter la saisie des commentaires de type Javadoc.

La complétion de code permet de générer une classe interne à partir d'une interface.

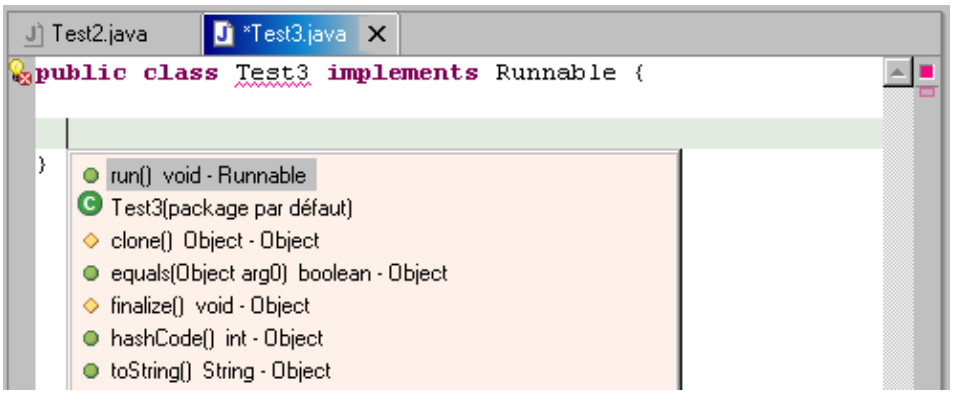


Après la saisie de la parenthèse ouvrante, il suffit d'appuyer sur les touches Ctrl + Espace. L'éditeur va générer la classe interne qui implémente l'interface.



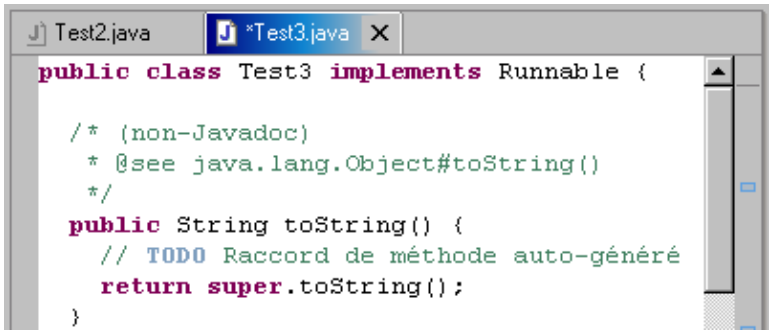
Il suffit de rajouter le ; à la fin de la déclaration de la classe interne et mettre le code nécessaire à la place du commentaire représentant une tâche à faire.

La complétion de code permet aussi de définir des méthodes définies dans une interface implémentée dans la classe ou de redéfinir une méthode héritée.



Dans le code du corps de la classe, il suffit d'appuyer directement sur les touches Ctrl + Espace. L'éditeur propose la liste des méthodes à implémenter ou à redéfinir en fonction de la déclaration de la classe.

La signature de la méthode sélectionnée est généré dans le code.

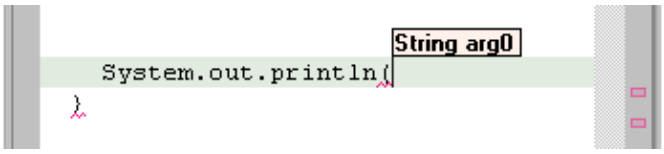


Par défaut, dans le cas de la redéfinition d'une méthode, l'appel à la méthode correspondante de la classe mère est appelée.

6.4.3. Affichage des paramètres sous la forme d'une bulle d'aide

Il est quasiment impossible de retenir les arguments nécessaires à toutes les méthodes de toutes les classes utilisées. L'éditeur d'Eclipse propose d'afficher sous la forme d'une bulle d'aide, les paramètres avec leur type pour la méthode en cours de rédaction dans le code.

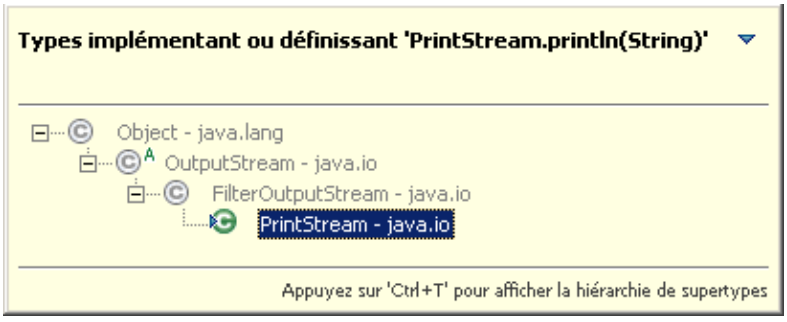
Pour utiliser cette fonction, il suffit d'appuyer sur les touches Ctrl + Maj + Espace en même temps dans l'éditeur pendant la saisie d'un ou des paramètres d'une méthode.



Si la méthode est surchargée, alors Eclipse demande de choisir la méthode à utiliser pour ainsi déterminer avec précision les paramètres à afficher dans la bulle d'aide.

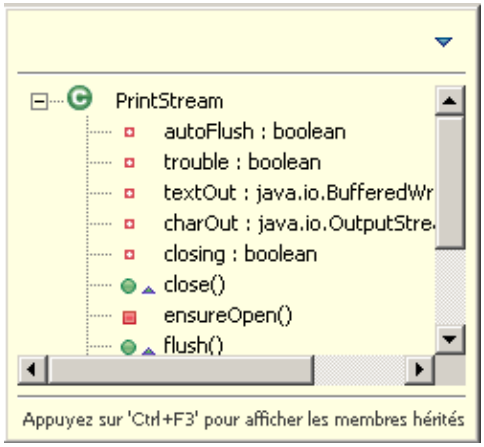
6.4.4. Hiérarchie de type dans une bulle d'aide

Il est possible demander dans l'éditeur de code d'afficher une bulle d'aide contenant la hiérarchie d'un type. Pour cela, il suffit de mettre le curseur sur un type, une méthode ou un package et d'appuyer sur la combinaison de touches Ctrl+T.

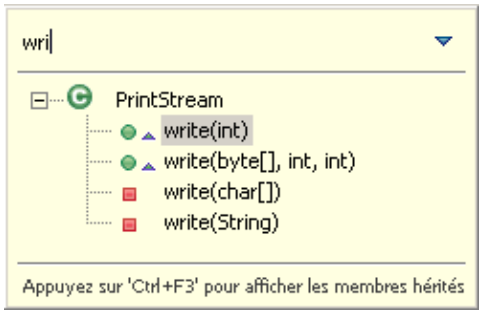


6.4.5. Affichage des membres dans une bulle d'aide

Il est possible demander dans l'éditeur de code d'afficher une bulle d'aide contenant la liste des membres d'un type. Pour cela, il suffit de mettre le curseur sur un type ou une méthode et d'appuyer sur la combinaison de touches Ctrl+F3.

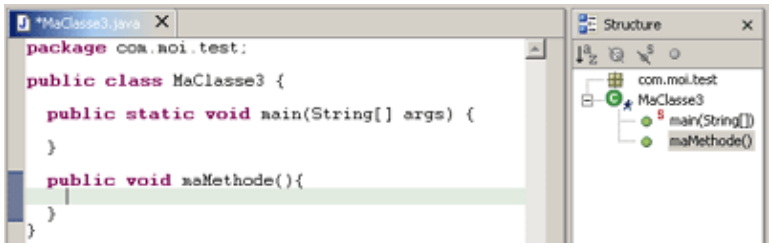


Il est possible de filtrer les éléments en saisissant les premiers caractères des membres à afficher :



6.4.6. L'éditeur et la vue Structure

Il existe un lien entre l'éditeur et la vue "Structure". Si cette vue est visible dans la perspective, dès que le curseur se déplace sur un membre de la classe en cours d'édition, le membre concerné est automatiquement sélectionné dans la vue "Structure".



Les lignes concernant le membre sélectionné sont marqués par une partie grisée dans la colonne de gauche de l'éditeur.

Les modifications apportées dans le code source (ajout, modification ou suppression de membres) sont automatiquement répercutées dans la vue "Structure".

Le bouton  de la barre d'outils permet de limiter l'affichage dans l'éditeur du membre sélectionné dans la vue "Structure".

Pour réafficher le source complet, il suffit de cliquer de nouveau sur le bouton.

6.4.7. La coloration syntaxique

L'éditeur possède une fonction de coloration syntaxique. Par défaut, les éléments sont colorés de la façon suivante :

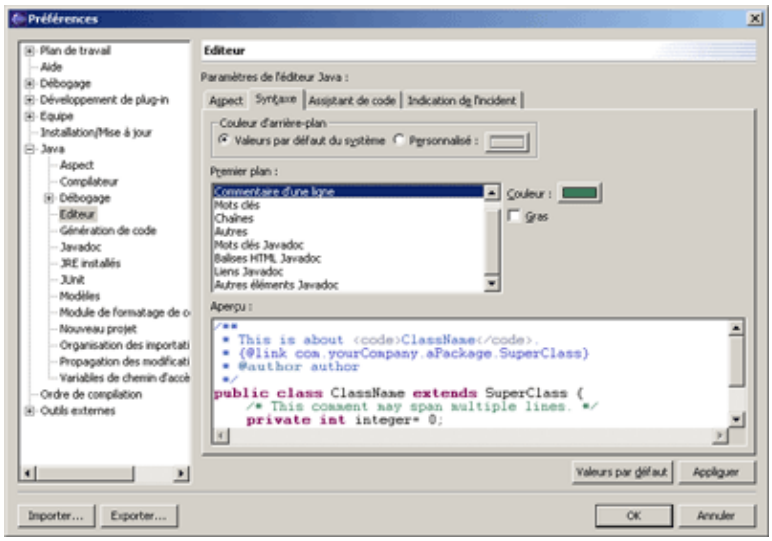
- les mots clés mots du langage sont colorés en violet gras
- les chaînes de caractères sont en bleu
- les commentaires sont en vert
- les commentaires Javadoc sont en bleu plus clair
- les balises Javadoc sont en gris
- les autres éléments sont en noir

Exemple :

```
package com.moi.test;

import java.awt.List;

/**
 * @author user
 *
 * To change this generated comment edit the t
 */
public class MaClasse extends List{

  private int maValeur ;

  public static void main(String[] args) {
  }

  private void maMethode() {}
}
```

Il est possible de modifier ces couleurs par défaut dans les préférences (menu Fenêtres/Préférence)

Il faut sélectionner l'élément Java/Editeur dans l'arborescence. Cet élément possède quatre onglets. L'onglet syntaxe permet de modifier les couleurs.

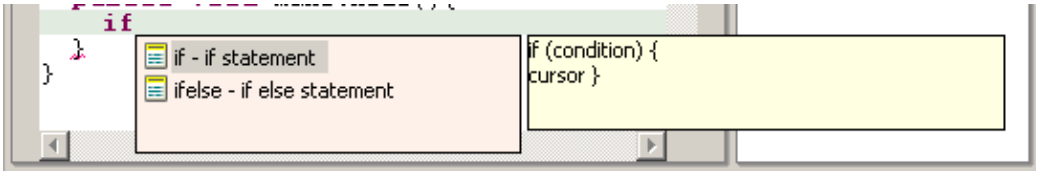


Il suffit de sélectionner l'élément concerné dans la liste déroulante et de sélectionner la couleur qui lui est associée en cliquant sur le bouton couleur. Une boîte de dialogue permet de sélectionner la nouvelle couleur à utiliser.

6.4.8. Utilisation des modèles

Il suffit de saisir le nom du modèle et d'appuyer sur les touches Ctrl + espace en même temps

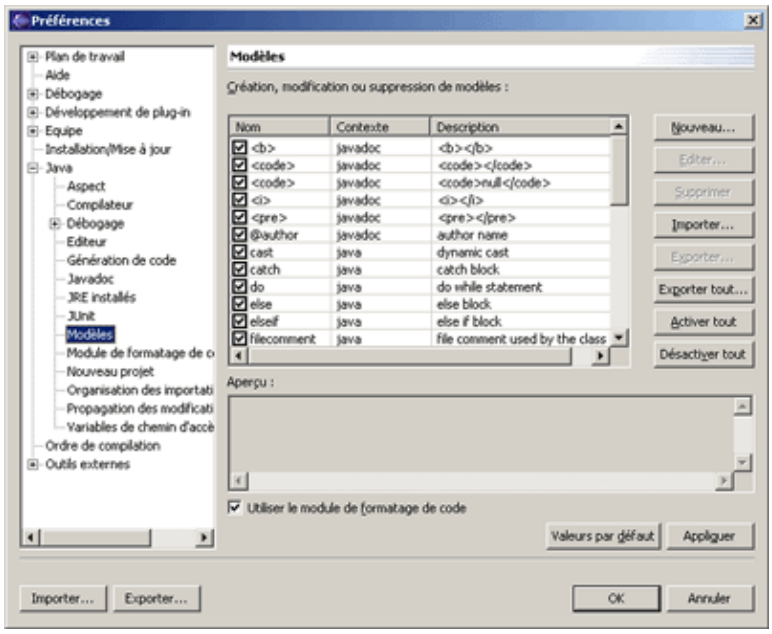
Exemple : avec le modèle if



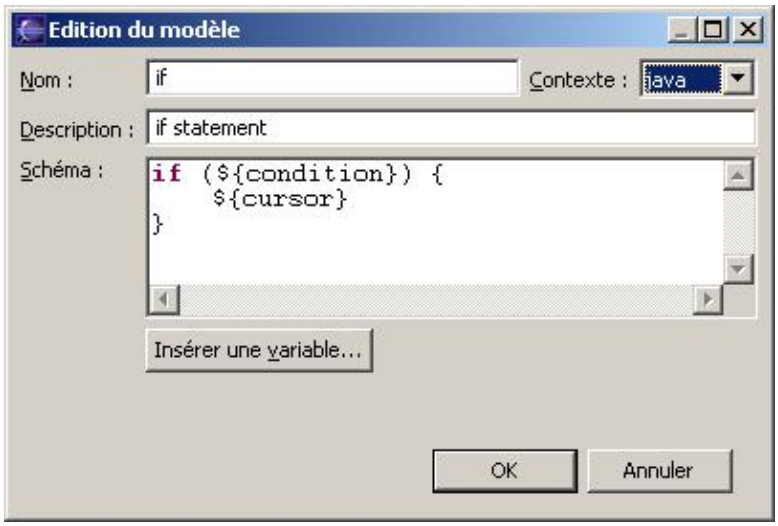
Il suffit de sélectionner le modèle à insérer pour qu'il soit immédiatement insérer dans le code.

Il est possible d'ajouter, de modifier ou de supprimer un modèle en utilisant les préférences (menu Fenêtre/Préférences).

Il suffit de sélectionner dans l'arborescence "Java/Modèles"



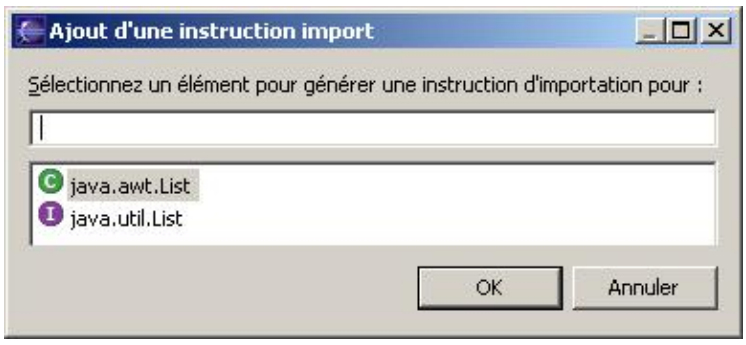
Pour modifier un modèle, il suffit de cliquer sur le bouton "Editer"



6.4.9. La gestion des importations

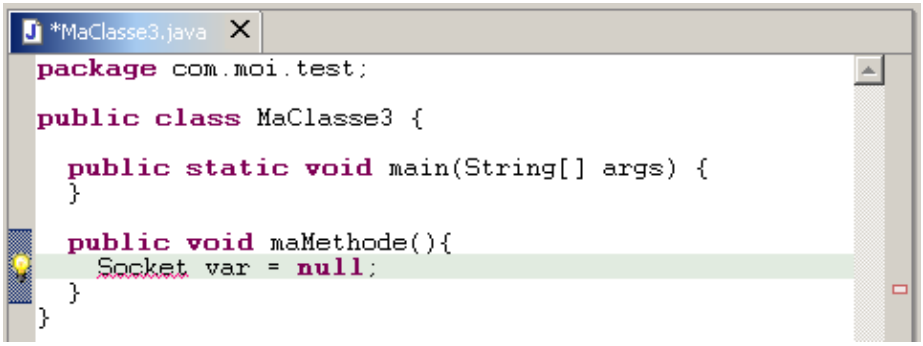
Il est possible de faire insérer la clause import pour un élément utilisé dans le code. Pour cela, il suffit de mettre le curseur sur l'élément concerné et de sélectionner l'option " Source/Ajout d'une instruction d'import " ou d'appuyer sur les touches Ctrl + Maj + M.

Si l'élément est déclaré dans un package unique, la clause import est automatiquement générée. Sinon une boîte de dialogue demande de sélectionner l'élément pour lequel la clause import sera générée.

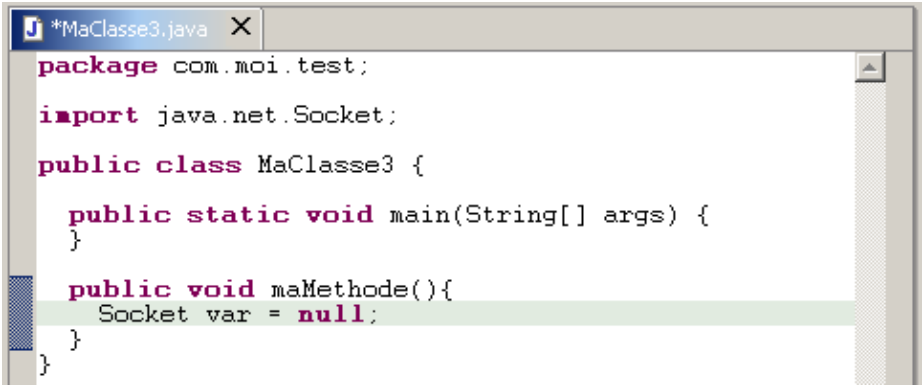


La fonctionnalité "Organisation des importations" est très pratique car elle permet d'insérer automatiquement les clauses imports avec les package requis par le code.

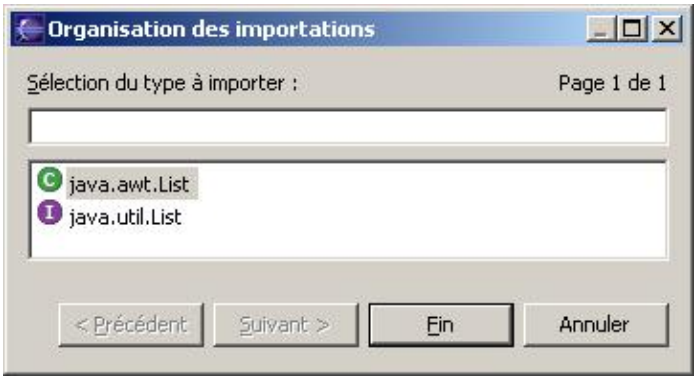
Par exemple : une variable de type Socket est déclarée, sans que le package java.net ne soit importé



Pour ajouter automatiquement la clause d'importation, il suffit d'utiliser l'option "Source/Organiser les importations" du menu contextuel.

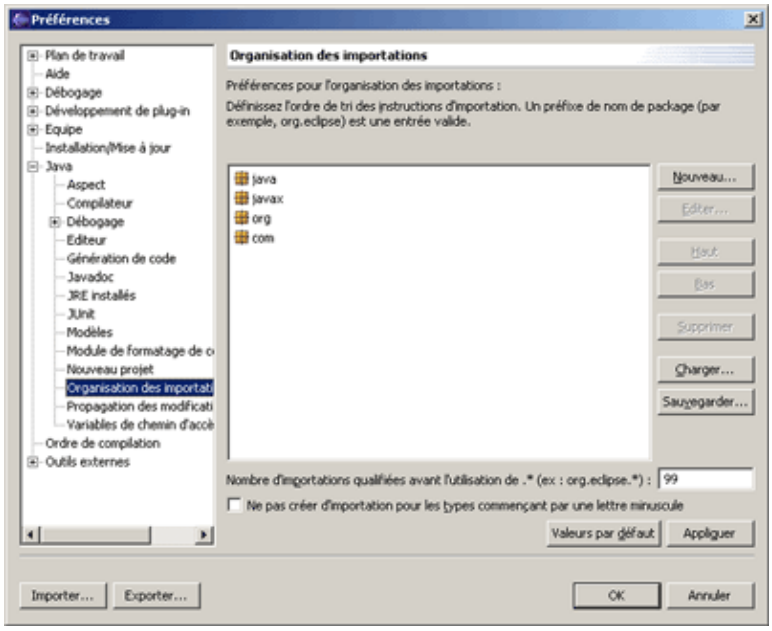


La clause import est insérée dans le code. Si un élément est contenu dans plusieurs packages, une boîte de dialogue demande la sélection du type à importer.



Cette fonctionnalité supprime aussi automatiquement les importations qui sont inutiles car aucune classe incluse dans ces packages n'est utilisée dans le code.

Certains réglages de cette fonctionnalité peuvent être effectués dans les préférences (menu "Fenêtre/Préférences"). Il suffit de sélectionner " Java/Organisation des importations " dans l'arborescence.



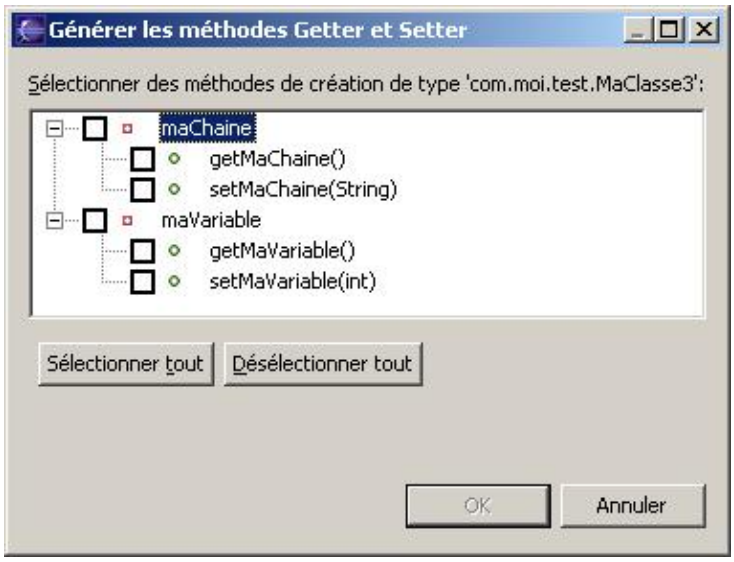
Les importations sont mises à jour lors d'un copier/coller d'une portion de code source. Cette nouvelle fonctionnalité très pratique permet lors d'un copier/coller d'un morceau de code d'une classe dans une autre de mettre à jour les importations requises.

Attention : cette fonctionnalité ne s'applique que si le copier et le coller est fait dans Eclipse.

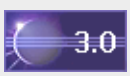
6.4.10. La génération de getter et setter

Il suffit d'utiliser l'option "Source/Générer les méthodes getter et setter" du menu contextuel.

Une boîte de dialogue permet de sélectionner, pour chaque attribut de la classe en cours d'édition, les getters et setters à générer

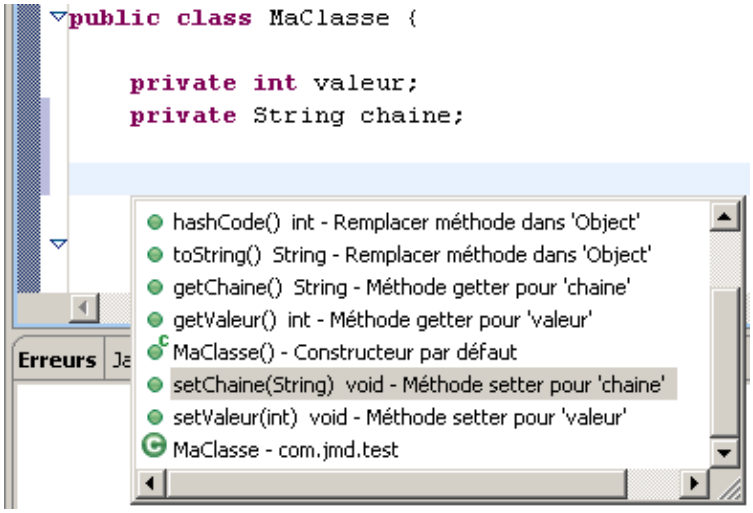


Il suffit de sélectionner les méthodes nécessaires pour qu'une génération par défaut soit effectuée dans le code.



Il est possible de créer directement des méthodes de type getter/setter ou le constructeur par défaut en utilisant la combinaison Ctrl+espace dans le corps d'une classe

Exemple : demander la création d'un setter pour le champ chaîne.

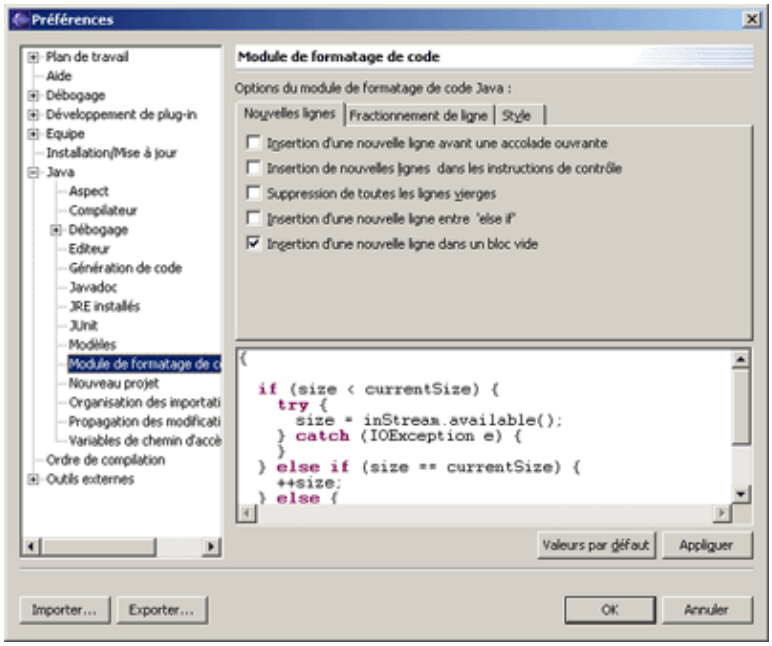


6.4.11. Formater le code

L'éditeur peut formater le code selon des règles définies. Pour utiliser cette fonctionnalité, il suffit d'utiliser l'option " Formater " du menu contextuel.

Les règles utilisées pour formater sont définies dans les préférences (menu "Fenêtre/Préférences").

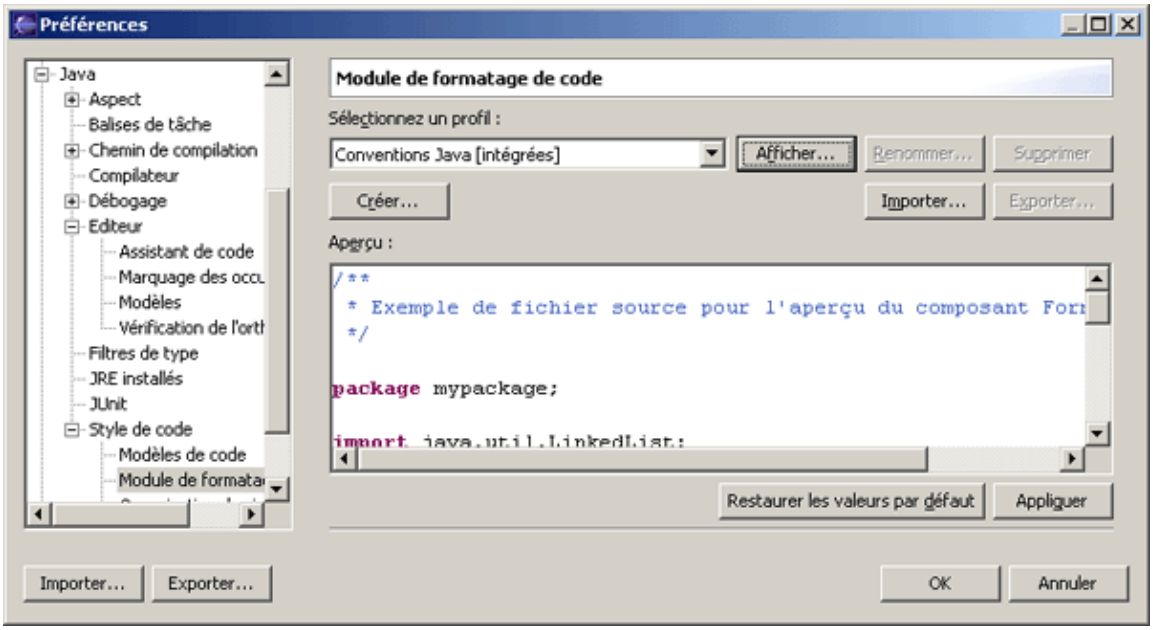
Il faut sélectionner dans l'arborescence, "Java/Module de formatage de code". Les règles de formatage peuvent être définies grâce à trois onglets.



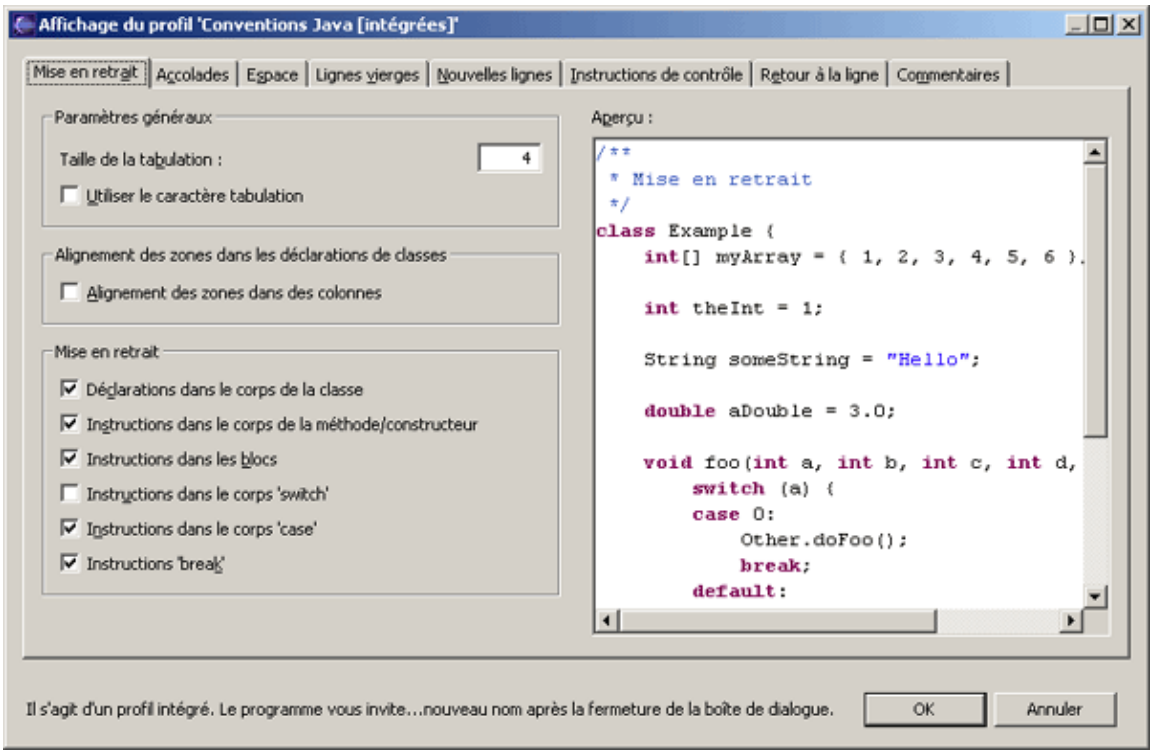
Le module de formatage de code a été entière réécrit et propose plusieurs fonctionnalités intéressantes :

- il propose deux profils prédéfinis (convention java et Eclipse 2.1) non modifiable sans duplication
- il permet de créer son propre profil
- il propose un aperçu du style sélectionné
- échanger des profiles entre plusieurs installations d'Eclipse 3 par import/export

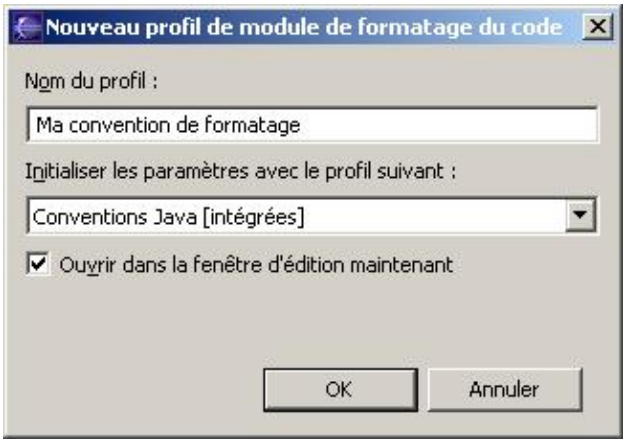
Les préférences permettent de mettre en oeuvre ces nouvelles fonctionnalités.



Le bouton « Afficher » permet de voir les différents options qui composent le profil.

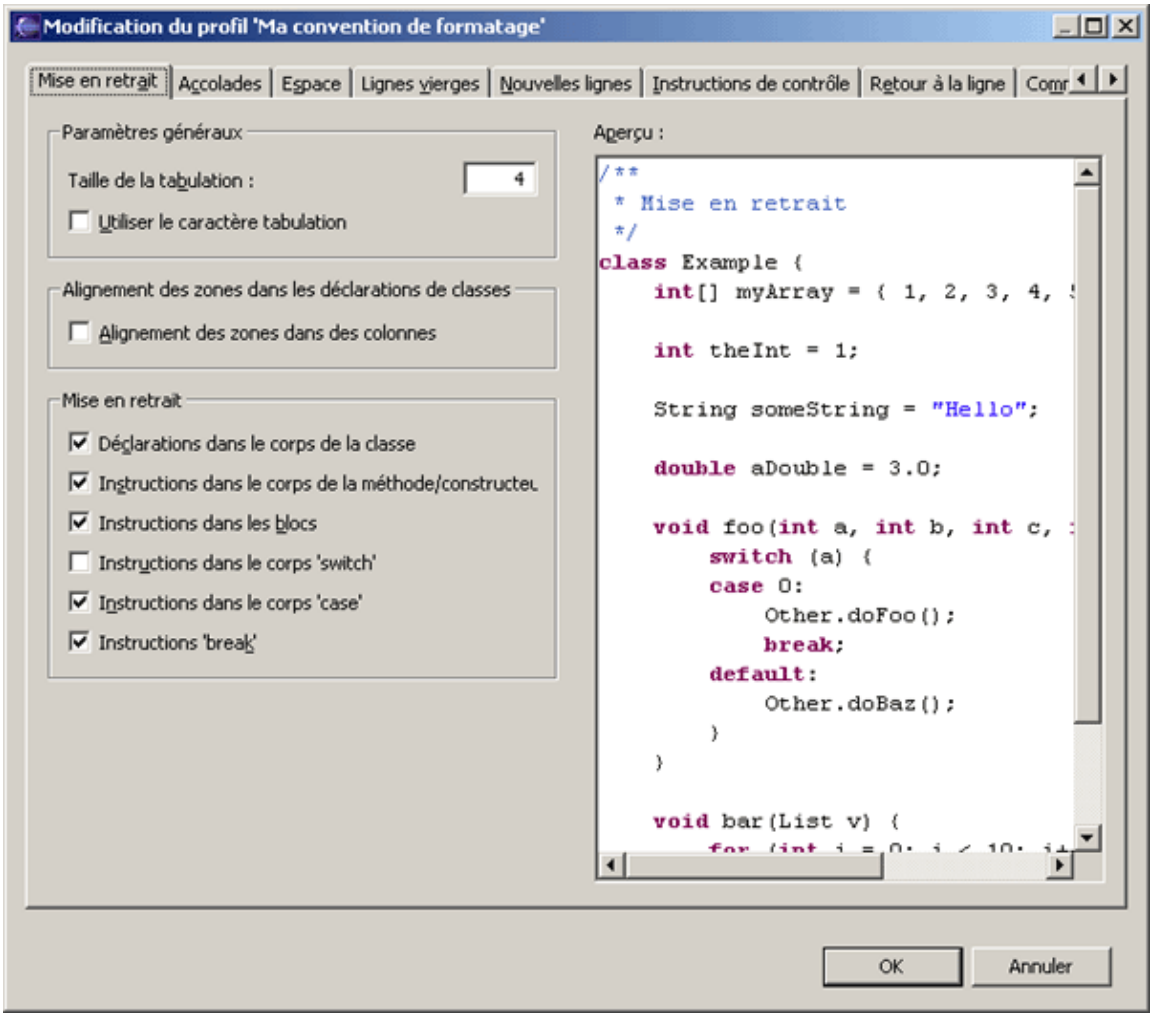


Le bouton « Créer » permet de créer son propre style à partir d'un style existant.



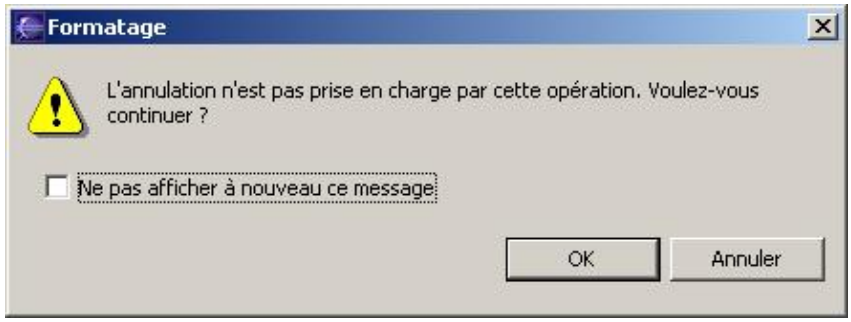
Il faut saisir le nom, sélectionner le profil d'origine et cliquer sur le bouton « OK »

La configuration des paramètres du profil est particulièrement riche et permet un niveau de détails très important.



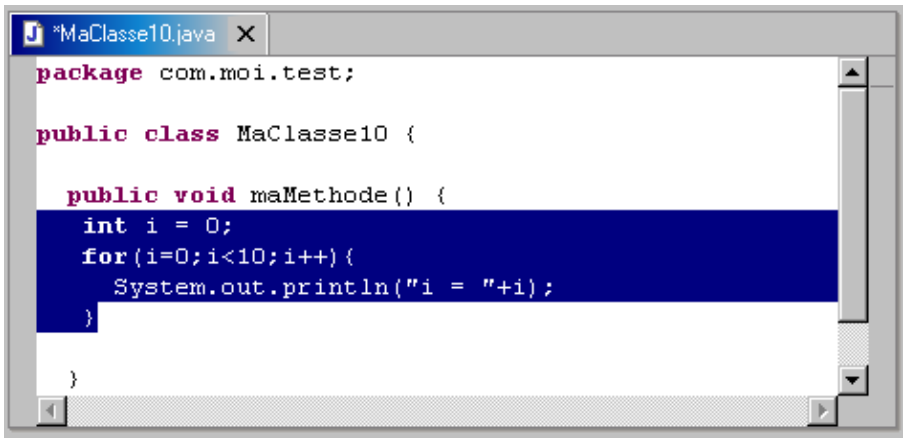
Dans la perspective Java, il est possible de demander le formatage du code source de tous les fichiers d'un package ou d'un projet en utilisant l'option « Source / Formater » du menu contextuel des éléments.

Une boite de dialogue demande la confirmation car l'opération ne peut être annulée.

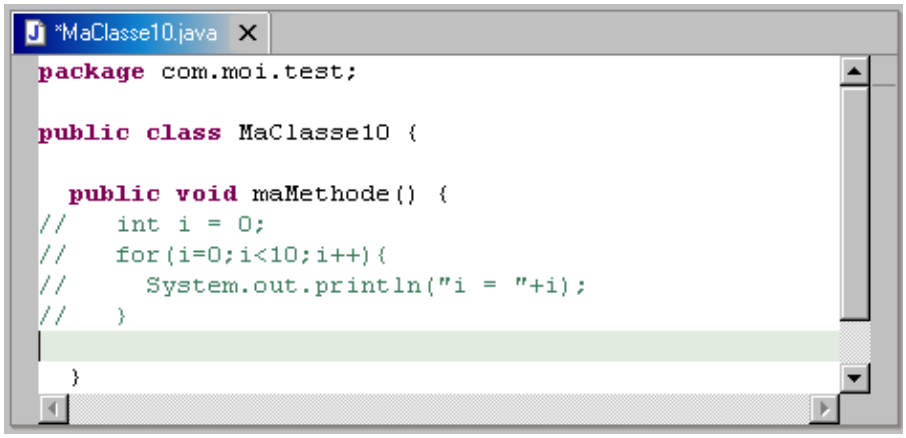


6.4.12. Mise en commentaire d'une portion de code

L'éditeur permet de mettre en commentaire une portion de code. Il suffit de sélectionner la portion de code concernée.



Puis, il faut utiliser l'option " Source / Mettre en commentaires " du menu contextuel de l'éditeur.

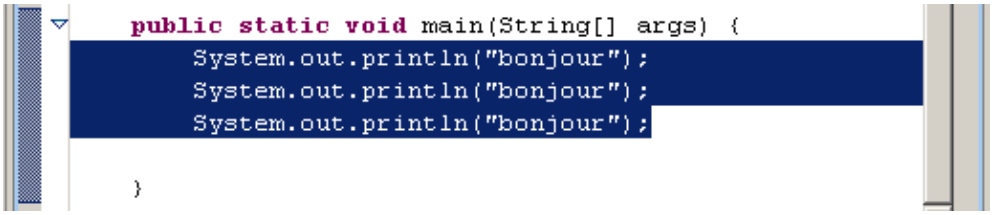


Pour supprimer les commentaires sur une portion de code, il suffit de sélectionner la portion de code et d'utiliser l'option " Source / Supprimer la mise en commentaire " du menu contextuel.

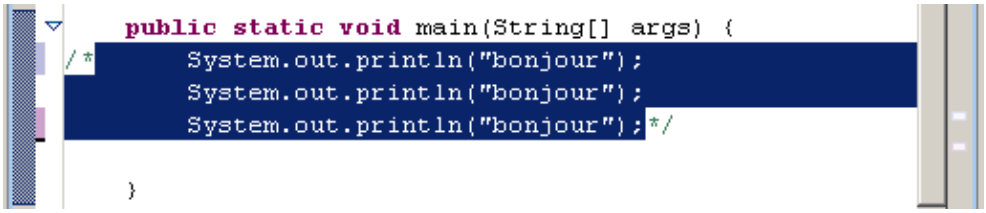
Les options de menu « Source / Mettre en commentaire » et « Source / Supprimer les commentaires » sont remplacées par une unique option « Source / Ajout/suppression de la mise en commentaires » qui effectue l'opération en fonction du contexte.

L'option « Source / Mettre en commentaire un bloc » du menu contextuel permet de mettre en commentaire la portion de code sélectionner grâce à un commentaire de type multi−ligne.

Exemple



Résultat :



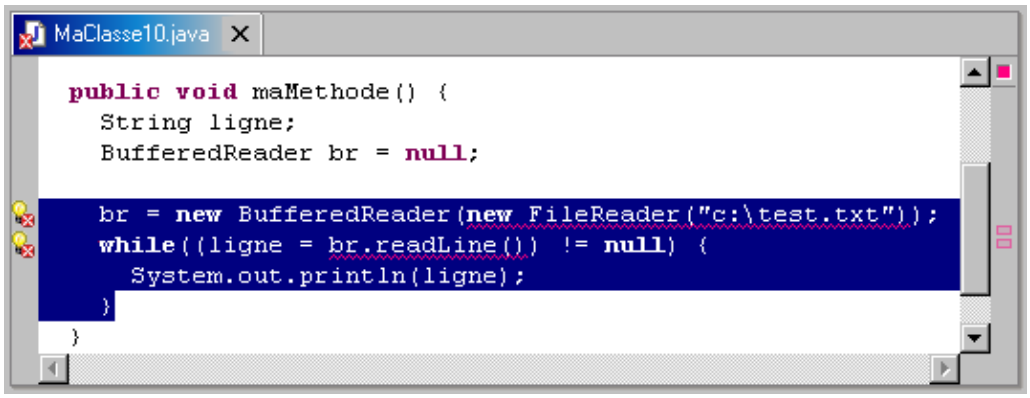
Pour utiliser cette fonctionnalité, il est aussi possible d'utiliser la combinaison de touches Ctrl+Maj+/

L'option « Source / Supprimer ma mise en commentaire un bloc » permet l'opération inverse.

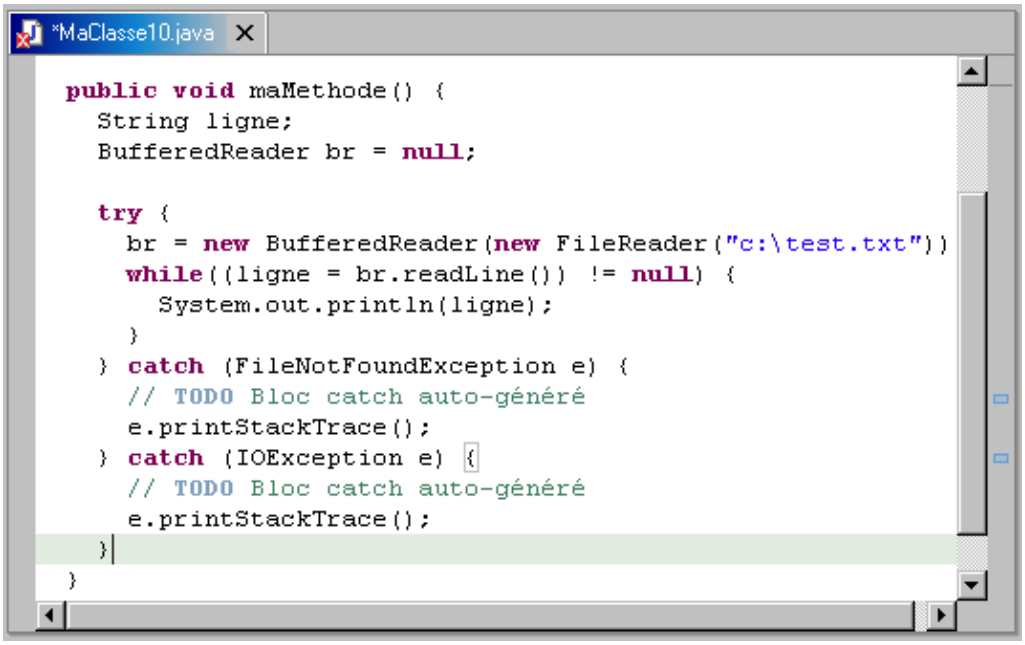
6.4.13. Protéger une portion de code avec un bloc try/catch

L'éditeur propose une option qui permet automatiquement de rechercher, dans un bloc de code sélectionné, une ou plusieurs exceptions pouvant être levées et de protéger le bloc de code avec une instruction de type try / catch.

Il faut sélectionner le bloc de code à protéger.

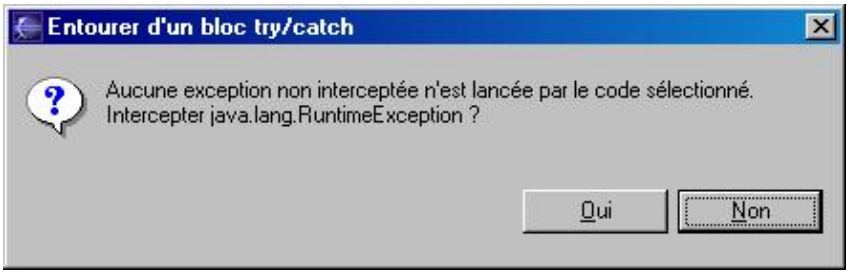


Puis utiliser l'option " Source / Entourer d'un bloc try / catch ". L'éditeur analyse le code et génère la cause try / catch qui va capturer toutes les exceptions qui peuvent être levées dans le bloc de code sélectionné.



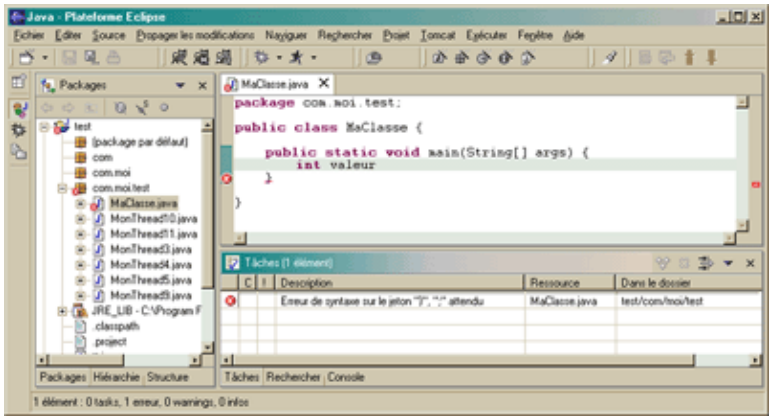
Chacune des instructions catch est marquée avec une tache " bloc catch auto−généré " pour indiquer au développeur d'ajouter le code nécessaire au traitement de l'exception.

Si le bloc de code ne contient aucun appel de méthode susceptible de lever une exception, une boîte de dialogue demande si l'instruction catch doit capturer une exception de type RuntimeException.



6.4.14. Les erreurs

Lors de la sauvegarde du fichier, celui ci est compilé et les erreurs trouvées sont signalées grâce à plusieurs indicateurs dans différentes vues.

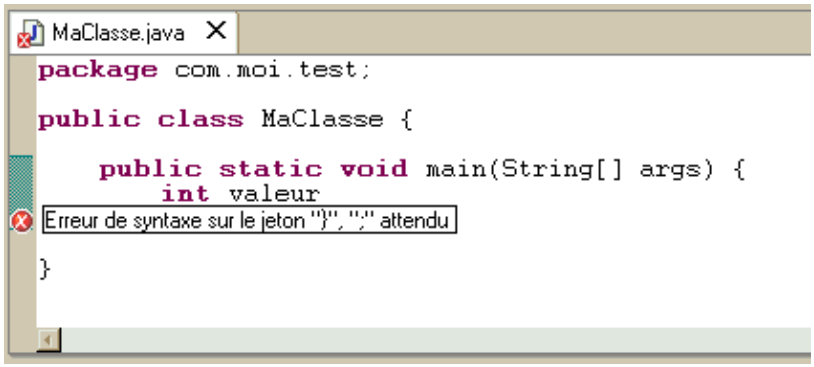


Les erreurs sont signalées par une îcone ronde rouge contenant une croix blanche dans les vues suivantes :

- dans l'éditeur, la ou les lignes contenant une ou plusieurs erreurs sont marquées avec cette icône

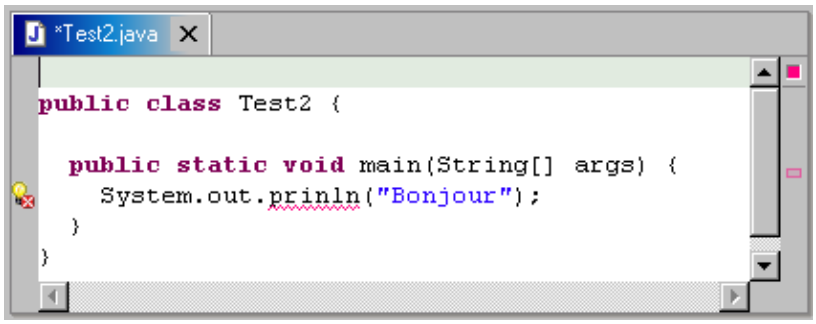
- dans la vue "Tâches", une entrée pour chaque erreur est créée pour faciliter leur recencement et leur accès
- dans la vue "Packages", tous les éléments allant du fichier source au projet, sont marqués de la petite icône

Dans l'éditeur, le simple fait de laisser le pointeur de la souris sur la petite icône permet d'afficher une bulle d'aide qui précise l'erreur.

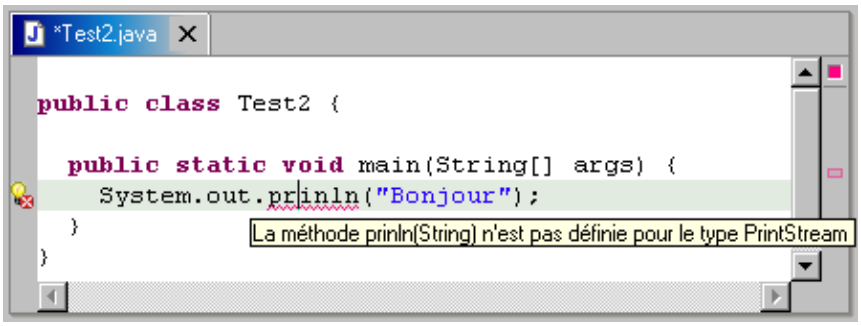


Les boutons  et  permettent respectivement de se positionner sur l'erreur suivante et sur l'erreur précédente.

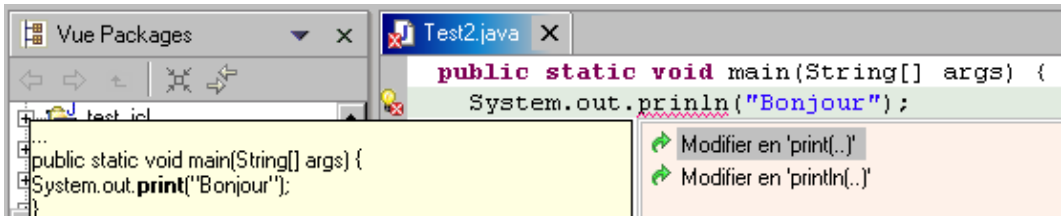
Ces mécanismes sont aussi utilisables au cours de la saisie du code Java dans l'éditeur. Eclipse est capable de détecter un certain nombre d'erreurs et de les signaler.



En positionnant le curseur sur la partie de code soulignée en rouge, une bulle d'aide fournie des informations sur l'erreur.



Un clic sur la petite icône en forme d'ampoule jaune, permet d'ouvrir une fenêtre qui propose des solutions de corrections



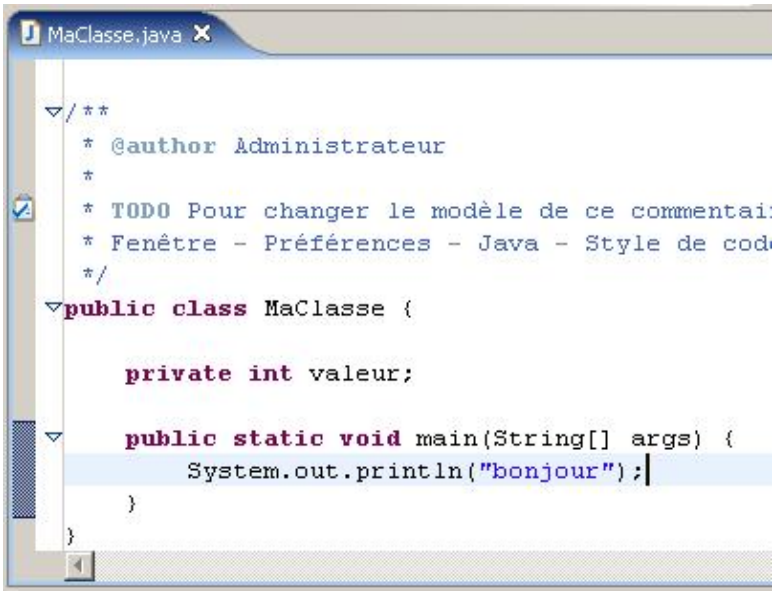
La seconde fenêtre permet de pré−visualiser les modifications qui seront apportées par la proposition choisie.

6.4.15. Masquer certaines portions de code

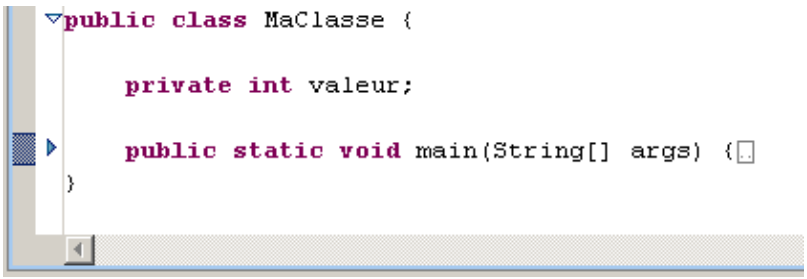
L'éditeur de code Java propose de masquer certaines portions de code correspondant :

- aux instructions d'importation
- aux commentaires
- aux corps des méthodes

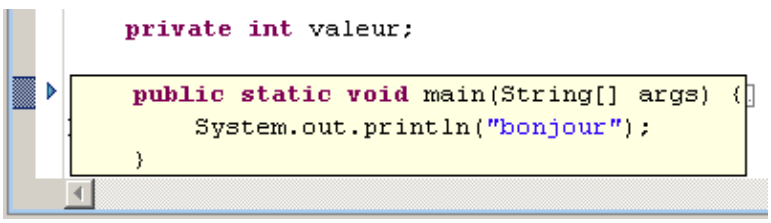
La définition de ces zones est automatique. Le début d'une zone non masquée est signalée par une petit icône dans la barre de gauche de l'éditeur ▽



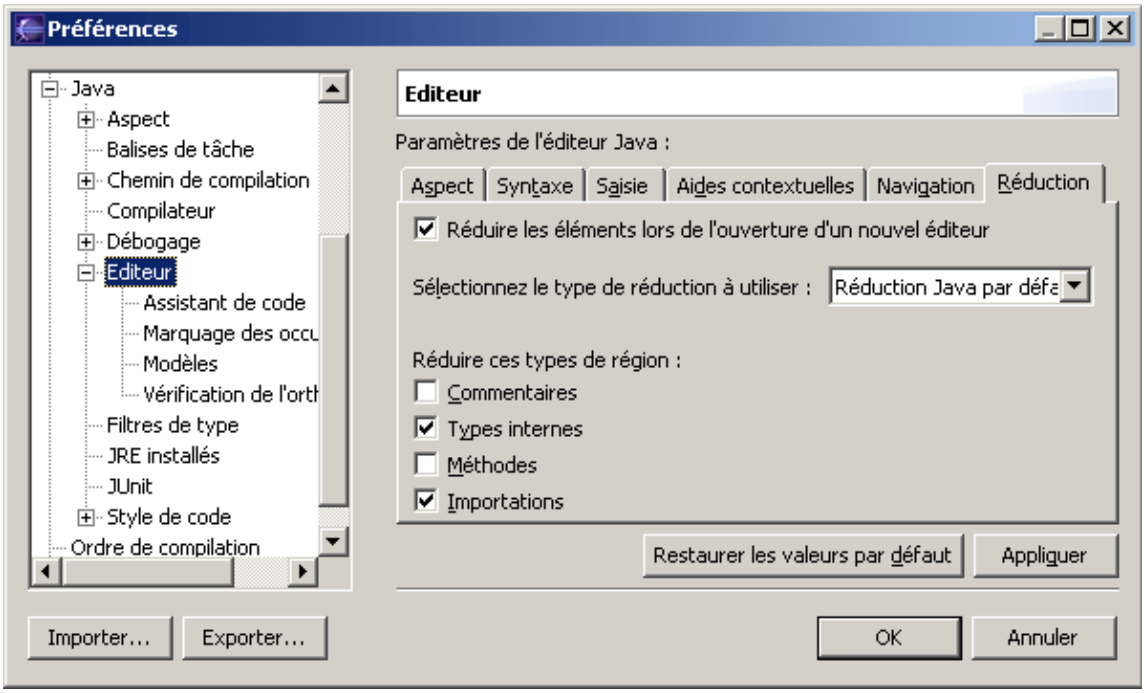
Pour masquer la zone, il suffit de cliquer sur le petit icône ▽. La zone est masquée et l'icône change d'apparence : ▷



En laissant le curseur de la souris sur l'icône ▷ , une bulle d'aide affiche le contenu de la zone masquée.



Les paramètres de cette fonctionnalité peuvent être modifiés dans les préférences.



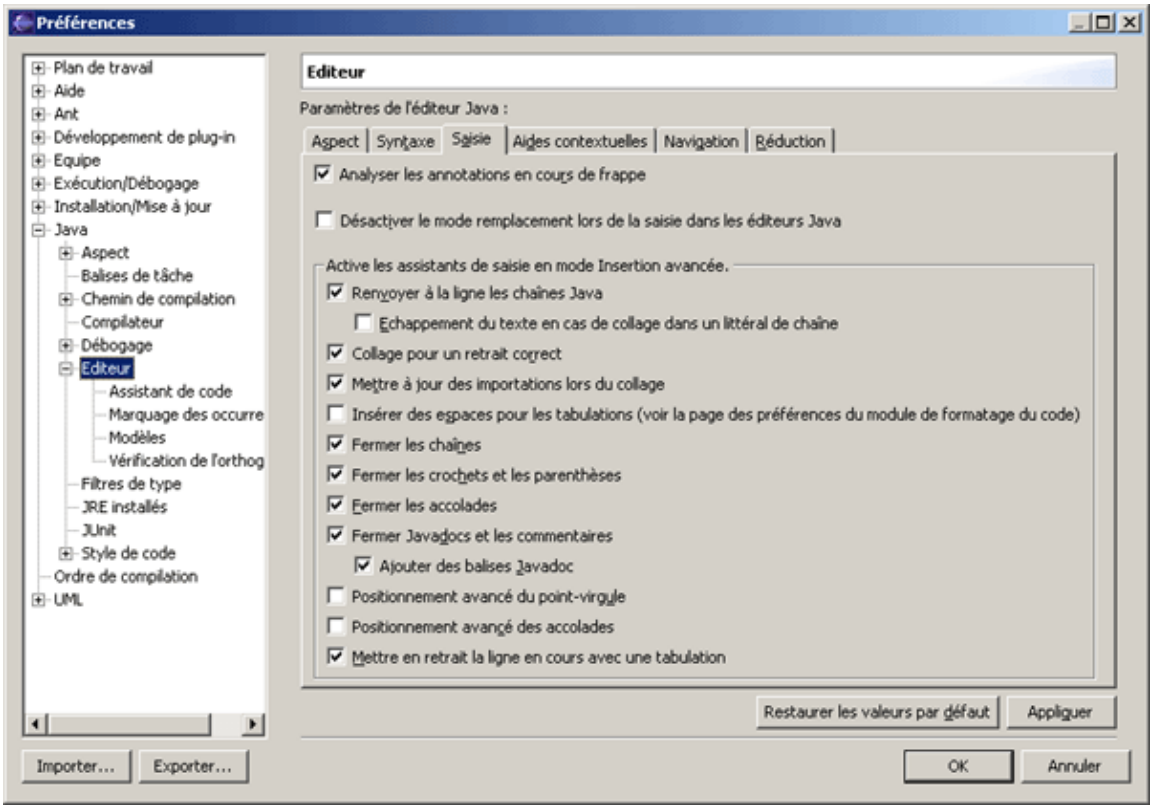
6.4.16. Le mode « Insertion Avancée »

Ce mode activé par défaut permet de simplifier la tâche du développeur lors de l'insertion d'une portion de texte dans l'éditeur de code Java. Il permet notamment de fermer automatiquement les parenthèses, les accolades, les guillemets, ...

Le mode de fonctionnement est signalé dans la barre d'état : 

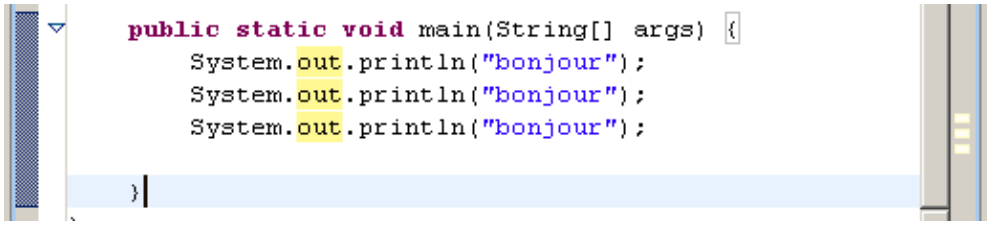
. Pour basculer du mode normal au mode avancé, il suffit d'utiliser la combinaison de touche Ctrl+Maj+Ins

Le paramétrage de ce mode ce fait dans les préférences.



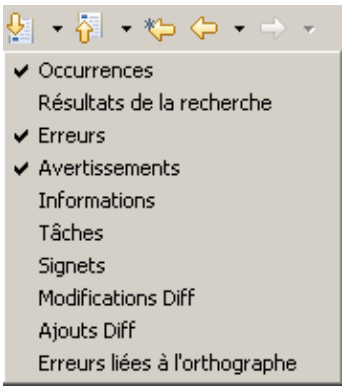
6.4.17. Marquer les occurrences trouvées

Suite à une recherche, il est possible de demander à l'éditeur de marquer les occurrences trouvées dans le code source en activant le bouton  de la barre d'outils ou en utilisant la combinaison de touche Alt+Maj+O.



Les occurrences sont marquées sous la forme d'annotations.

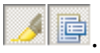
Il est possible d'utiliser les boutons  et  pour se positionner sur l'occurrence suivante ou précédente. Il faut cependant que les annotations de type occurrences soient cochées dans le menu déroulant associé aux boutons.

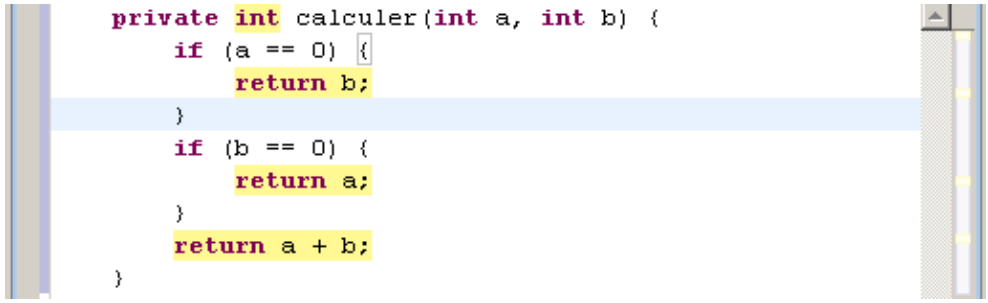


6.4.18. Marquer les points de sortie d'une méthode

Même si cela n'est pas recommandé, il est possible de mettre plusieurs instruction return dans une méthode.

L'éditeur propose une fonctionnalité pour marquer sous forme d'annotation les points de sortie d'une méthode.

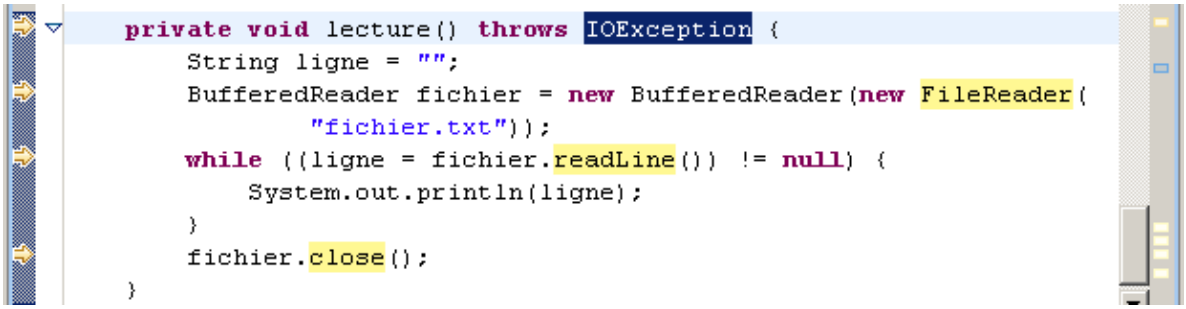
Pour cela, il faut positionner le curseur sur le type de retour de la méthode et activer les boutons  .



6.4.19. Marquer les emplacements ou une exception est levée

L'éditeur propose une fonctionnalité qui permet de marquer les méthodes qui lèvent une exception d'un type déclaré dans la signature de la méthode.

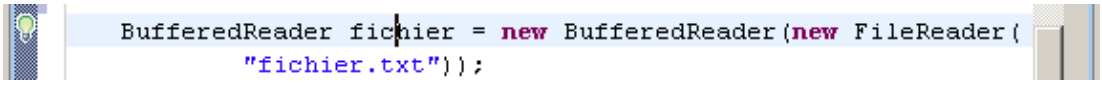
Pour cela, il faut placer le curseur sur une des exceptions utilisées dans la clause throws de la déclaration de la méthode et d'utiliser l'option "Rechercher / Occurrences d'exceptions" du menu principal.



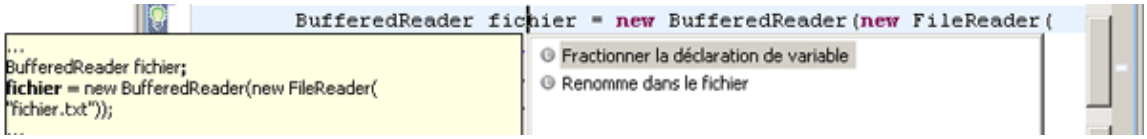
6.4.20. L'assistant de correction rapide

L'éditeur peut faciliter la rédaction du code en proposant un certains nombre d'action selon le contexte.

Pour savoir si une ou plusieurs correction rapide peuvent être proposée, il faut activer l'option dans les préférences, sous l'arborescence « Java / Editeur », sur l'onglet « Aspect », il faut cocher l'option « Activer l'ampoule pour l'assistant rapide » qui par défaut est décochée.



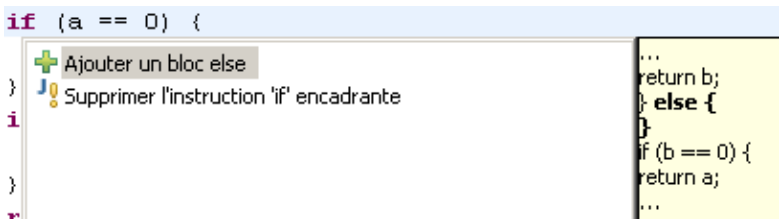
Il suffit alors de cliquer sur l'ampoule ou d'utiliser la combinaison de touche Ctrl+1 pour activer l'assistant.



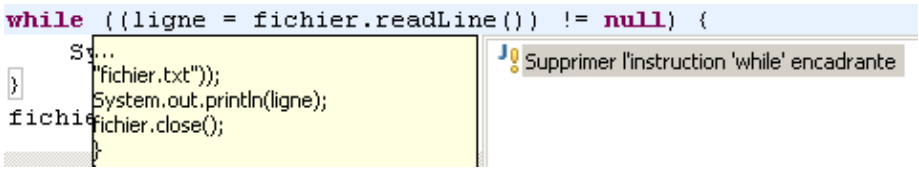
Les opérations proposées par l'assistant le sont fonction du contexte.

Les différentes opérations proposées sont nombreuses et facilitent souvent la réalisation de petites tâches fastidieuses à réaliser manuellement, par exemple :

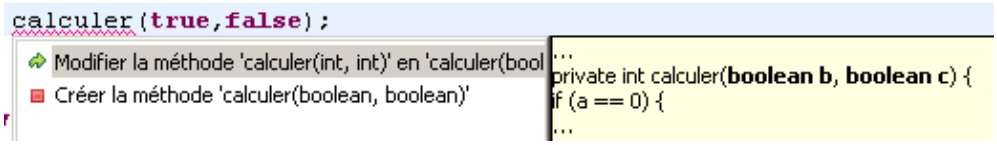
- ajouter un bloc else à une instruction if



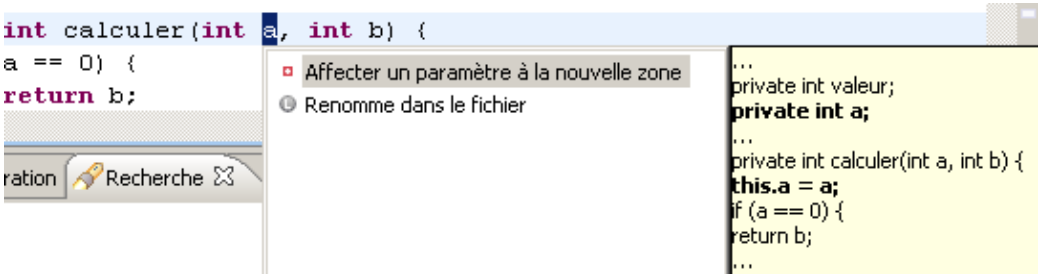
- supprimer une instruction englobante



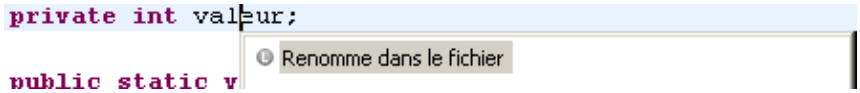
- modifier l'appel ou la méthode en cas de non concordance de paramètres



- créer un nouveau champ qui sera initialisé avec un paramètre de la méthode



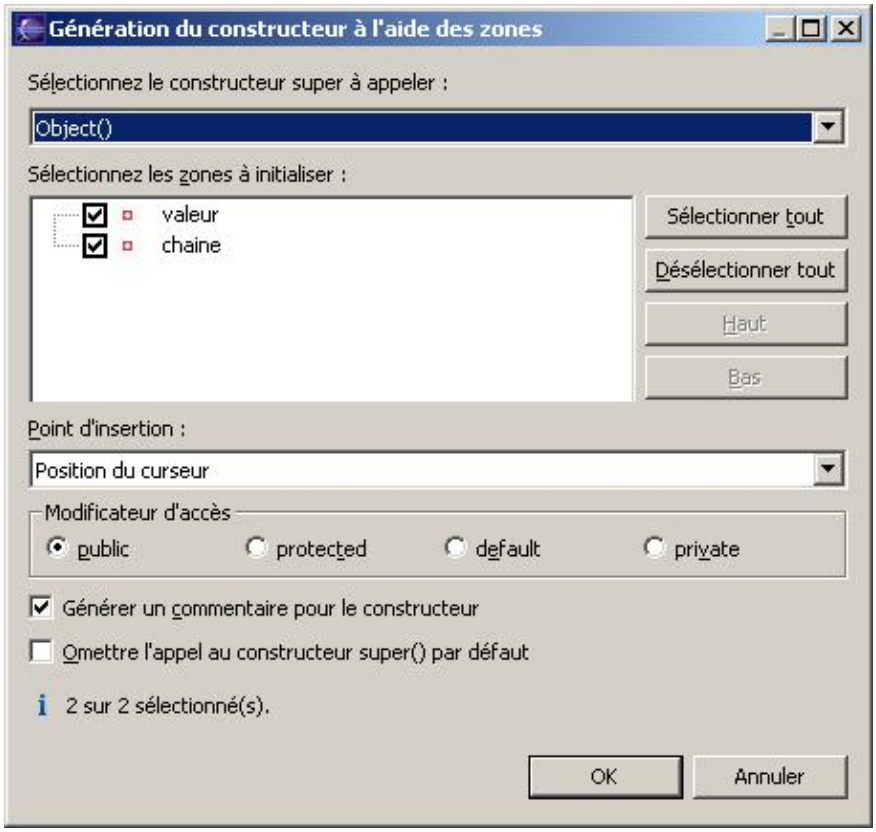
- renommer localement un membre



6.4.21. La génération de constructeur

3.0

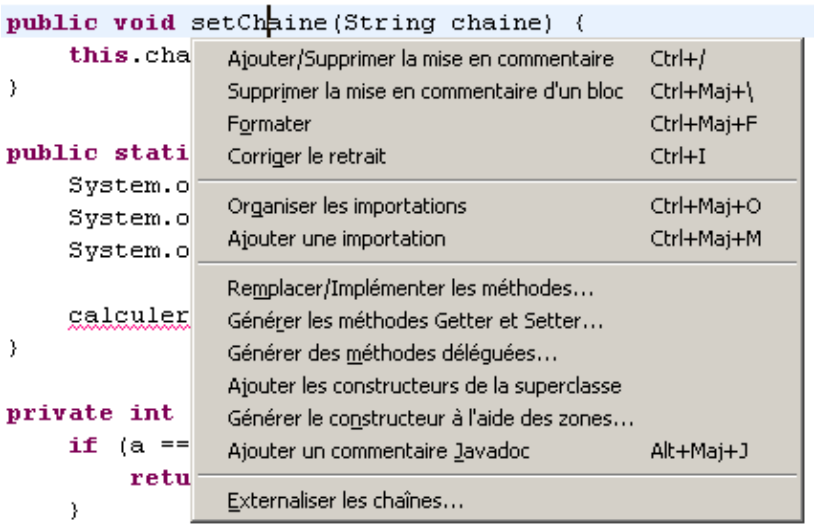
L'option « Source / générer le constructeur à partir des zones ... » du menu contextuel permet de faciliter la création d'un constructeur.



Il est alors possible de sélectionner les paramètres du nouveau constructeur et les différentes options pour que le code du constructeur soit généré.

6.4.22. Raccourci clavier pour avoir accès aux fonctions de modification du code source

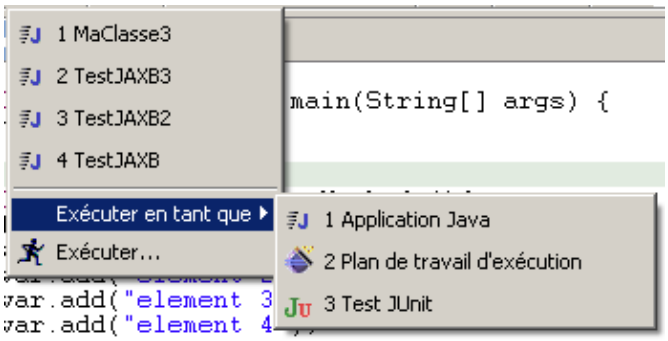
L'éditeur propose un menu contextuel pour accéder rapidement aux fonctions de modification du code source en utilisant la combinaison de touche Alt+Maj+S



6.5. Exécution d'une application

Dans la perspective "Java", il est possible d'exécuter une application de plusieurs façons.

Pour exécuter l'application en cours d'édition, il suffit de cliquer sur la flèche du bouton  de la barre d'outils.

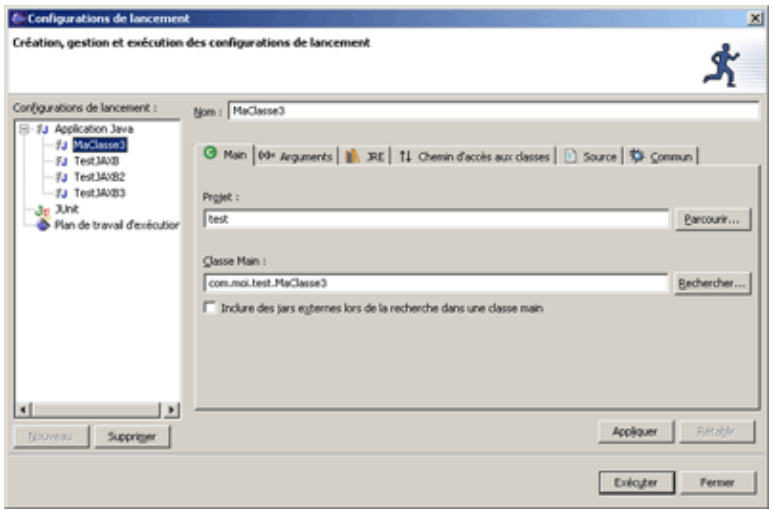


Ce menu déroulant propose différentes options :

- relancer les précédentes exécutions listées dans la première partie du menu
- l'option "Exécuter en tant que" permet de lancer l'application dans trois modes différents (Application java, Test JUnit et plan de travail d'exécution)
- l'option "Exécuter" ouvre une boîte de dialogue pour paramétrer précisément les options d'exécution

L'option "Exécuter en tant que / Application Java" lance la méthode main() d'une application.

L'option "Exécuter" ouvre une boîte de dialogue qui permet de saisir tous les paramètres d'exécution.



La boite de dialogue se compose de six onglets.

L'onglet "Main" permet de sélectionner le projet et la classe de ce projet qui contient la méthode main() à exécuter.

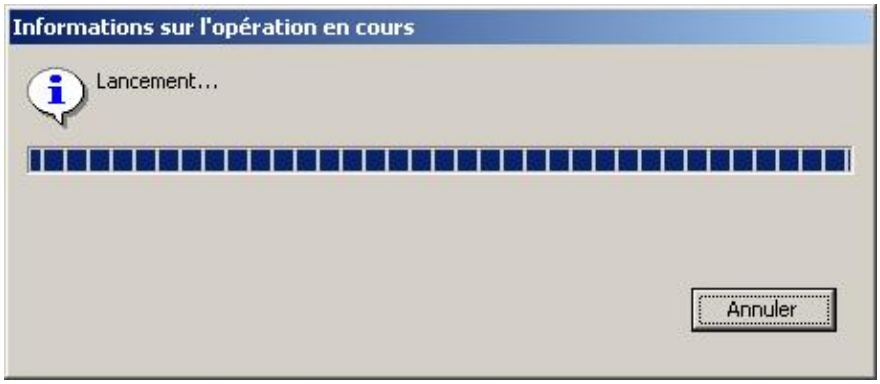
L'onglet "Arguments" permet de préciser les arguments passés à l'application et à la machine virtuelle.

L'onglet "JRE" permet de sélectionner le JRE à utiliser lors de l'exécution.

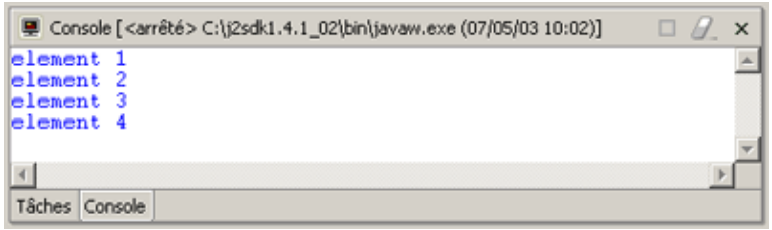
L'onglet "Chemin d'accès aux classes" permet de modifier la liste et l'ordre des bibliothèques utilisées lors de l'exécution. Cet onglet permet de modifier la liste définie dans le projet qui est celle utilisée par défaut.

L'onglet "Commun" permet de préciser le type de lancement et le mode d'exécution et de débogage.

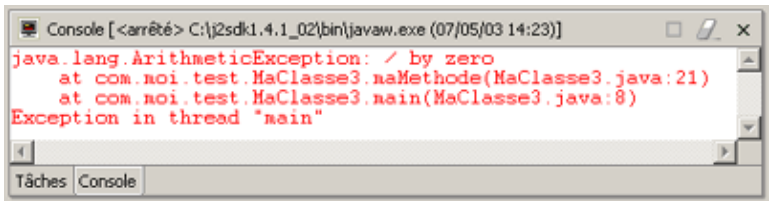
Une fois tous les paramètres voulus renseignés, il suffit de cliquer sur le bouton " Exécuter " pour lancer l'application.



La vue " Console " permet de voir les données qui sont envoyées dans le flux standard de sortie et d'erreurs.



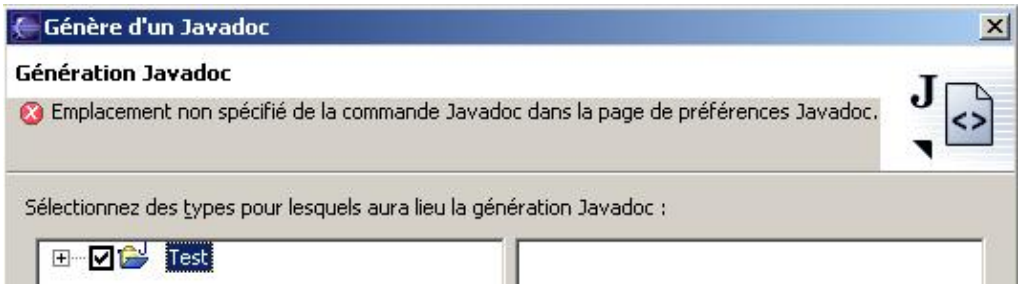
Les messages ayant pour origine une exception sont aussi envoyés dans cette vue.



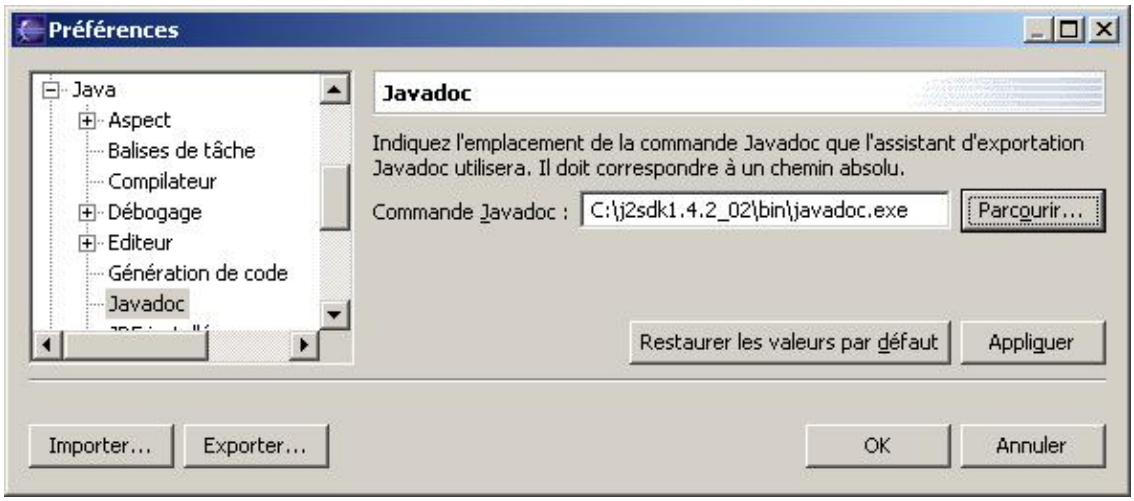
6.6. Génération de la documentation Javadoc

Pour demander la génération de la documentation Javadoc, il faut utiliser le menu "Projet/Générer/Le javadoc".

Pour utiliser cet option, il faut obligatoirement que les préférences concernant Javadoc soit renseignées, sinon un message d'erreur empêche l'utilisation de l'assistant

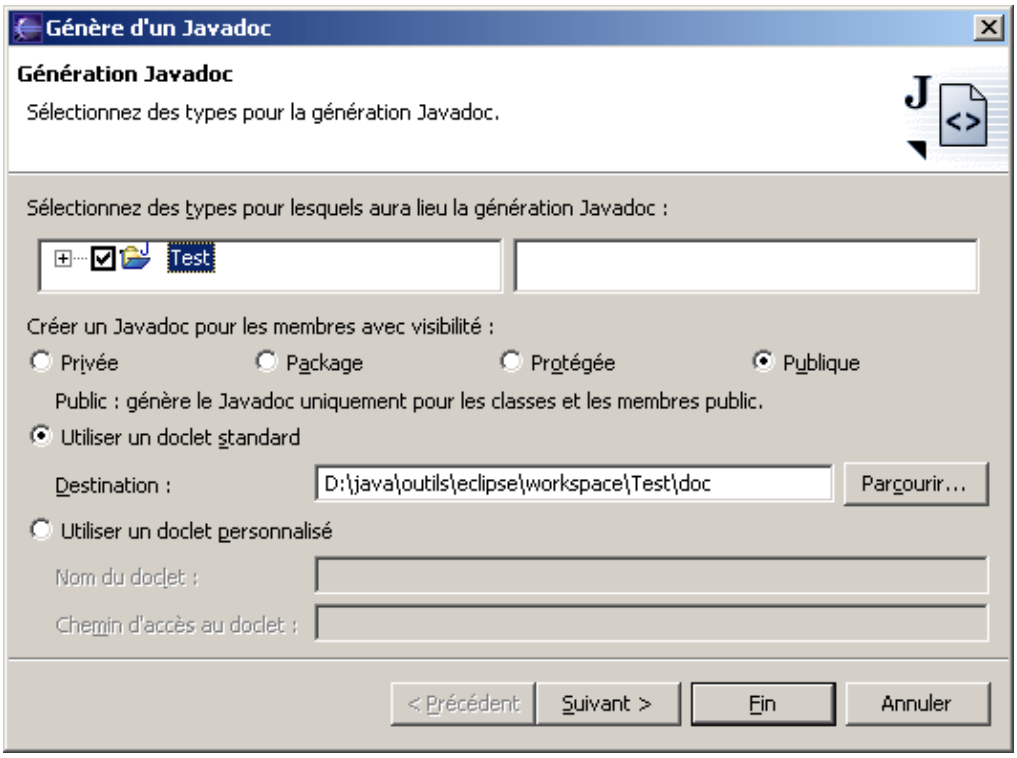


Pour résoudre le problème, il faut aller dans l'arborescence "Java/Javadoc" des préférences, cliquer sur le bouton "Parcourir" et sélectionner le fichier javadoc.exe du JDK à utiliser.



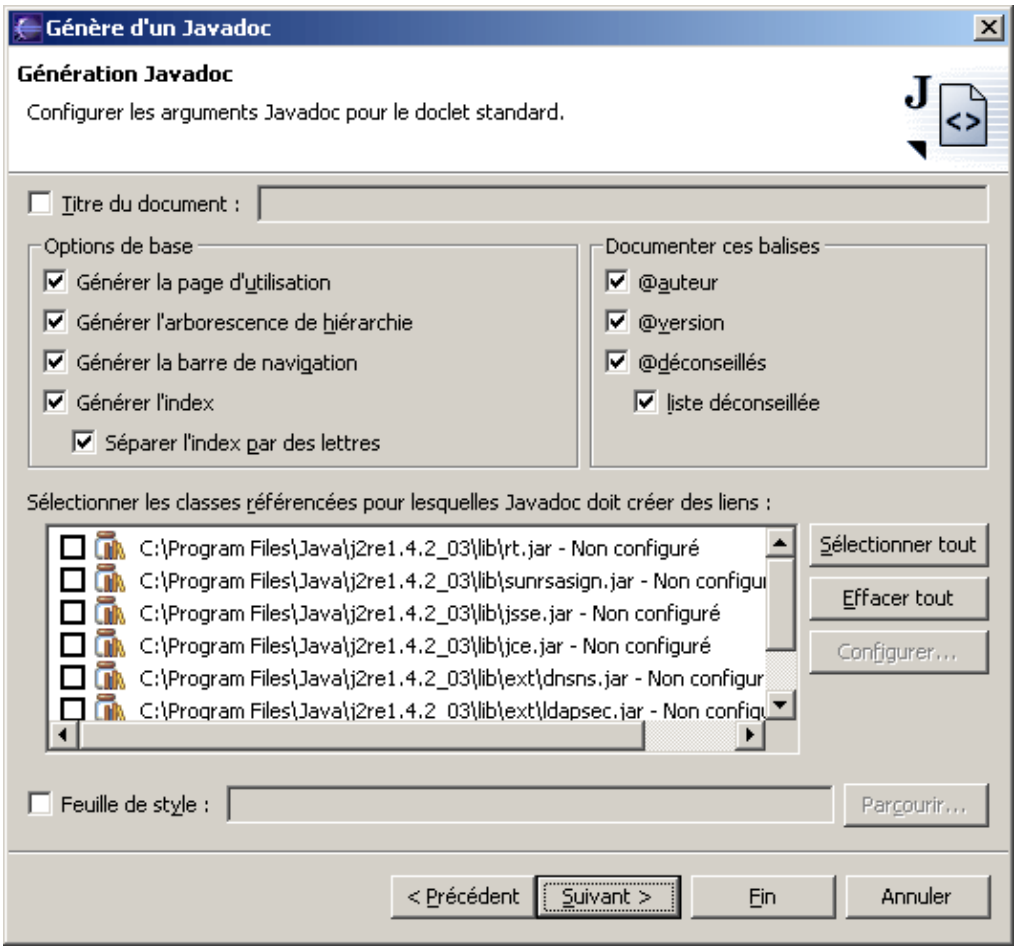
Cliquez sur le bouton "OK" pour valider les modifications.

La génération de la documentation au format Javadoc se fait avec un assistant. Il faut lui indiquer : le ou les projets concernés, la visibilité des membres à inclure, le doclet à utiliser et le répertoire de destination.



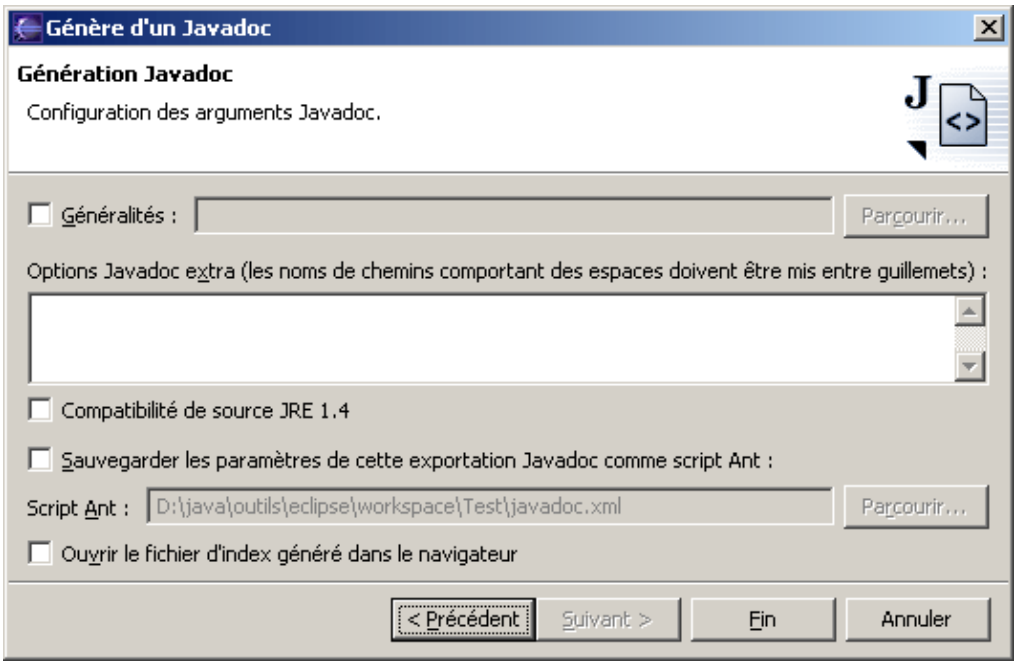
Cliquez sur le bouton "Suivant"

La page suivante de l'assistant permet de préciser des options pour l'outil Javadoc.

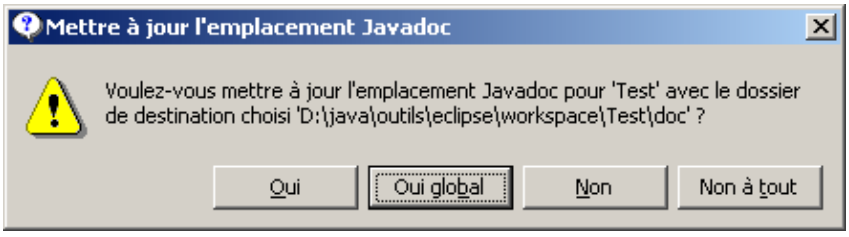


Une fois les options configurées, cliquez sur le bouton "Suivant".

La page suivante de l'assistant permet de préciser d'autres options.

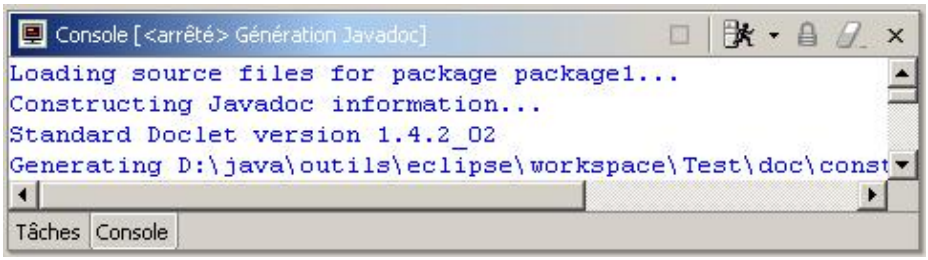


En cliquant sur "Fin", la génération de la documentation est effectuée. L'assistant demande si l'emplacement javadoc du projet doit être mis à jour.

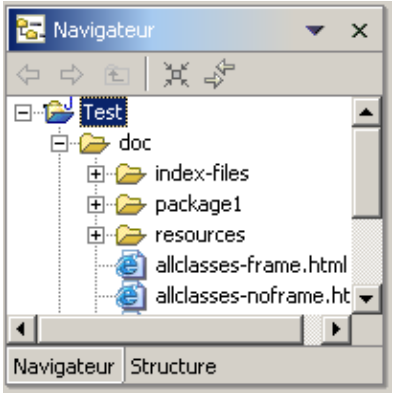


Il est conseillé de répondre "Oui" pour permettre d'avoir accès à cette documentation à partir de l'éditeur en appuyant sur les touches "Shift" + "F2".

Le détail de la génération est affiché dans le vue "Console".

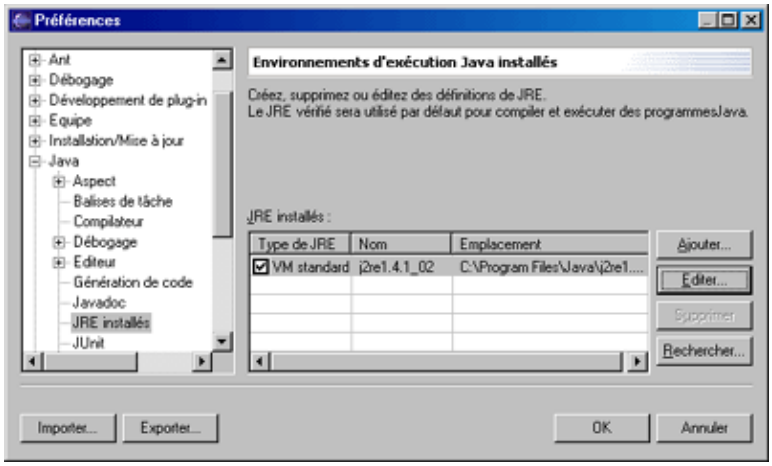


Les fichiers apparaissent dans la vue "Navigateur".

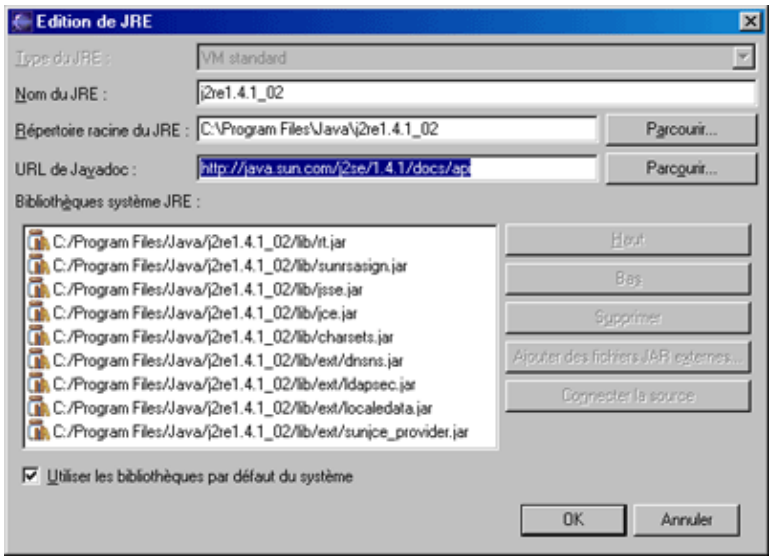


6.7. Définition du JRE à utiliser

Eclipse est capable de travailler avec plusieurs JRE. Dans l'arborescence " Java/JRE installé " des préférences, il est possible de définir plusieurs JRE installés sur la machine.



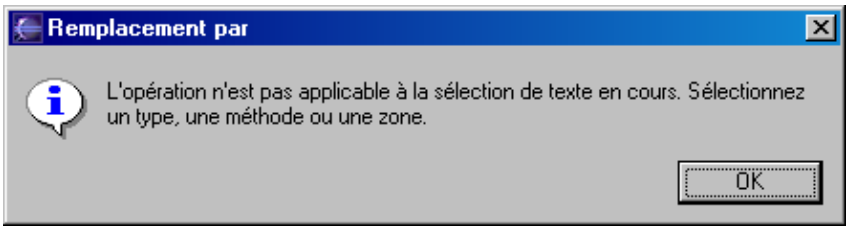
Un clic sur le bouton "Editer" permet de modifier les données du JRE défini dans Eclipse.



6.8. Utilisation de l'historique local

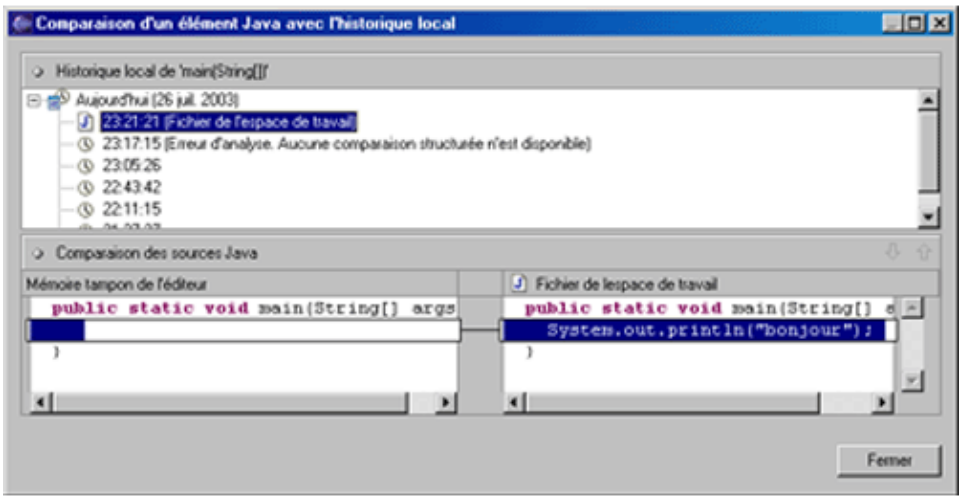
L'historique local est une fonctionnalité proposée par Eclipse pour conserver un certain nombre de versions du code pour chaque élément contenu dans l'espace de travail.

Pour pouvoir utiliser l'historique local, il faut placer le curseur sur un élément de code sinon un message d'erreur est affiché :



L'option "Historique local" du menu contextuel propose 4 options :

- " Comparer à ... "

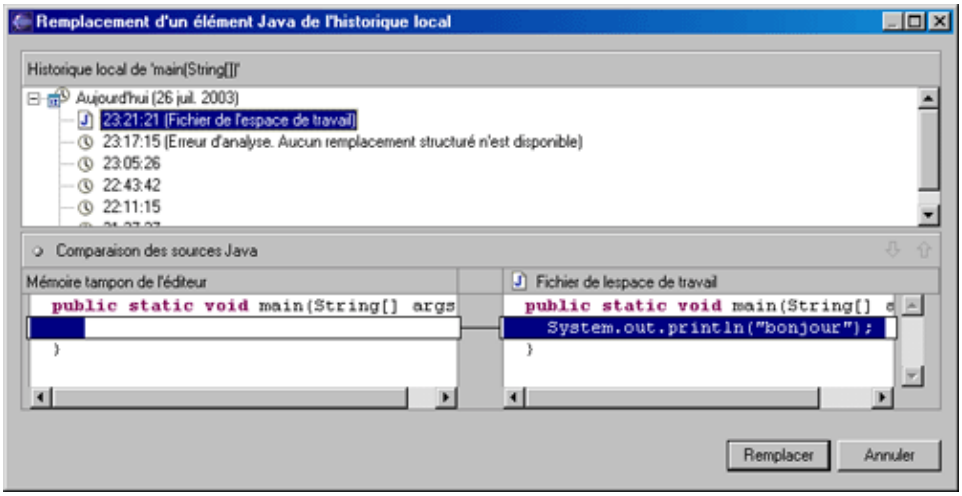


Cette option permet de comparer la version contenue dans l'éditeur avec celles contenues dans l'historique. Il n'est pas possible de reporter les modifications avec cette option.

- " Remplacer par l'élément précédent "

Cette option permet de remplacer la version de l'éditeur par la dernière contenue dans l'historique : elle correspond à la dernière sauvegarde.

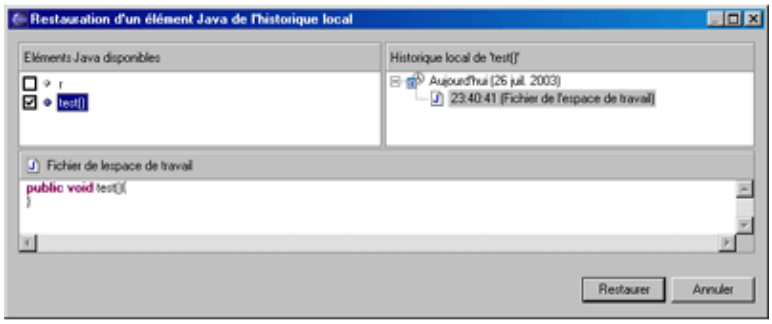
- " Remplacer par ... "



Il suffit de sélectionner la version désirée et de cliquer sur le bouton "Remplacer".

- " Restaurer à partir de ... "

Cette option permet de restaurer des éléments contenus dans l'historique mais qui ne sont plus présents dans l'éditeur de code.

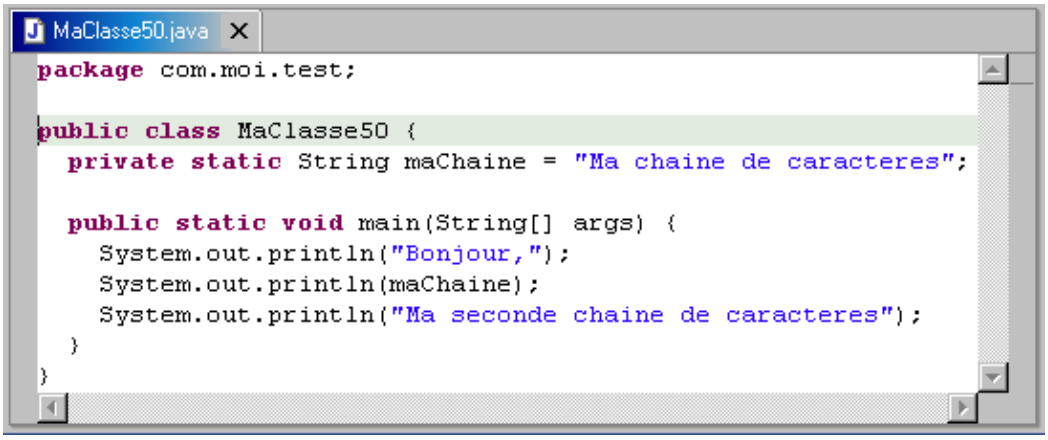


Il suffit de cocher le ou les éléments et de sélectionner la version de l'historique à restaurer et de cliquer sur le bouton "Restaurer".

6.9. Externaliser les chaînes

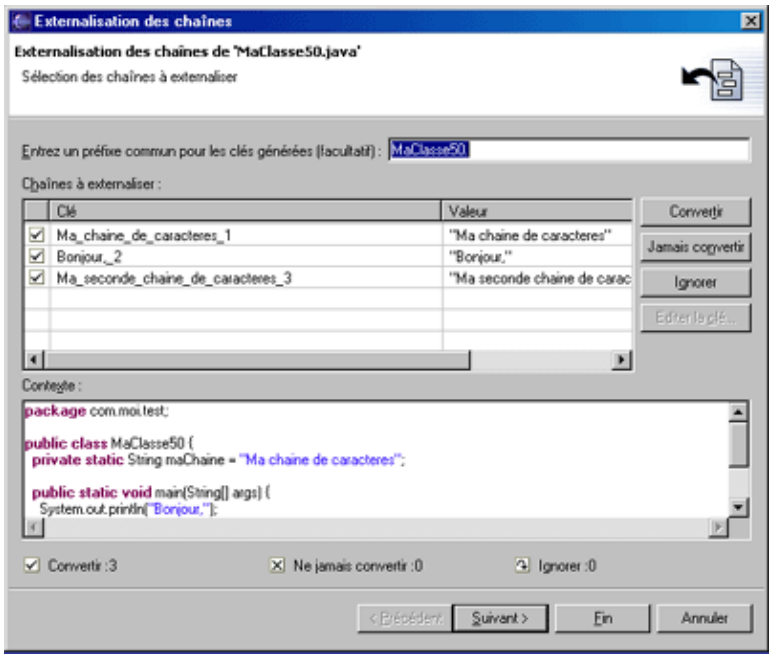
Eclipse possède une fonction permettant d'automatiser l'externalisation des chaînes de caractères dans un fichier de ressource afin de faciliter l'internationalisation de la classe traitée.

L'exemple de cette section utilise le code source suivant :



Dans la vue "Package", il faut sélectionner le fichier source puis utiliser l'option " Source / Externaliser les chaînes " du menu contextuel.

Eclipse analyse le code source à la cherche de chaînes de caractères et affiche l'assistant " Externalisation des chaînes ".

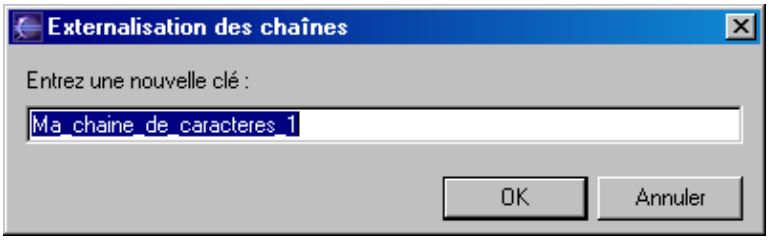


La première page de l'assistant permet de sélectionner les chaînes de caractères à traiter détectées par Eclipse.

Pour chaque chaîne, il est possible de changer le nom de la clé associée à la chaîne de caractères et de préciser si la chaîne doit être traitée ou non.

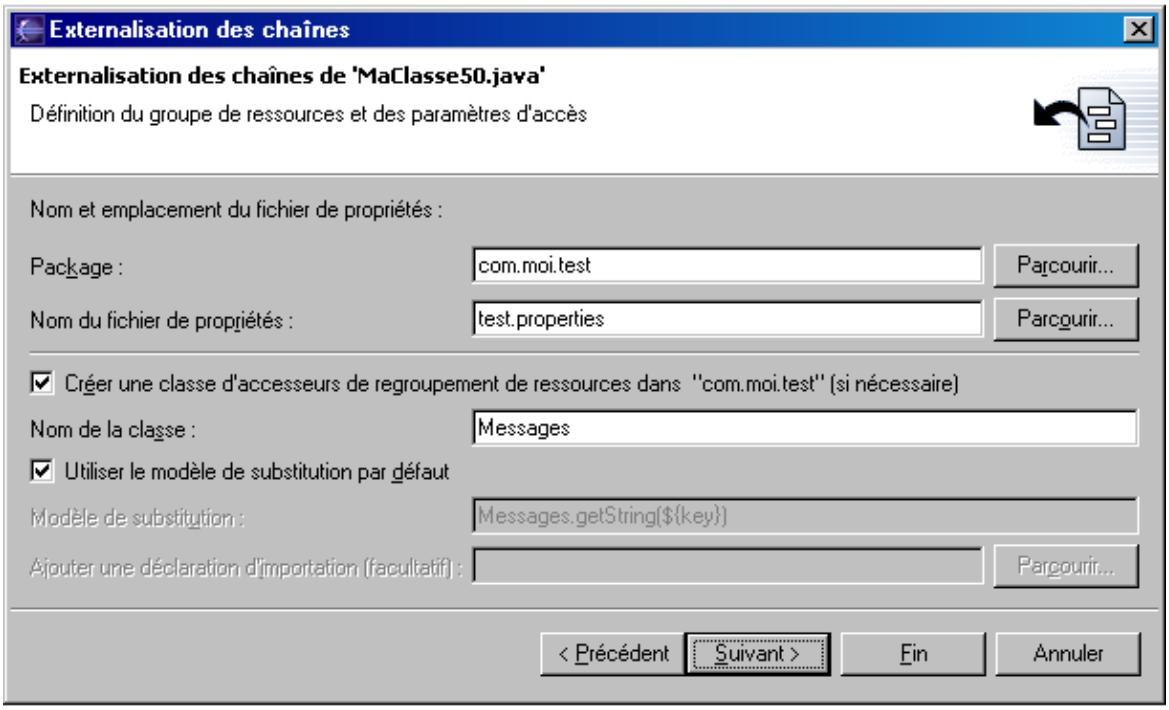
Pour modifier la clé, il est possible de cliquer sur la clé et de saisir le nouveau nom ou de sélectionner la ligne

de la chaîne et de cliquer sur le bouton " Editer la clé "

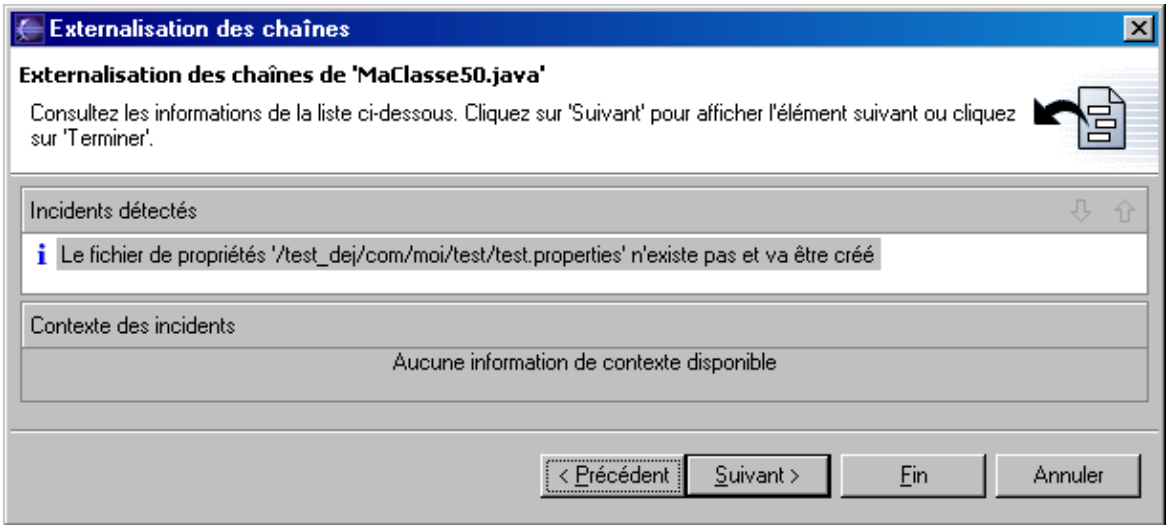


Pour indiquer si la chaîne doit être traitée, il est possible de cliquer plusieurs fois sur la case à cocher de la ligne correspondante pour obtenir le symbole correspondant à la valeur voulue ou de cliquer sur le bouton " Convertir ", " Jamais convertir " ou " Ignorer " après avoir sélectionné la ligne désirée.

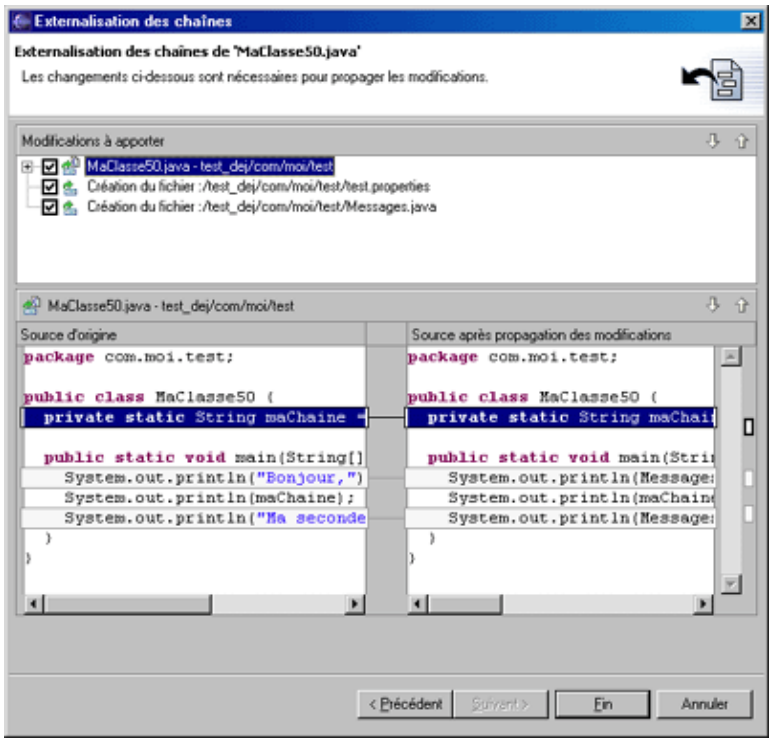
Un clic sur le bouton " Suivant " permet de préciser des informations sur le fichier de ressource qui sera généré.



Un clic sur le bouton " Suivant " affiche une liste des problèmes détectés.



Un clic sur le bouton " Suivant " permet d'afficher une page qui prévisualise les modifications qui vont être apportées.

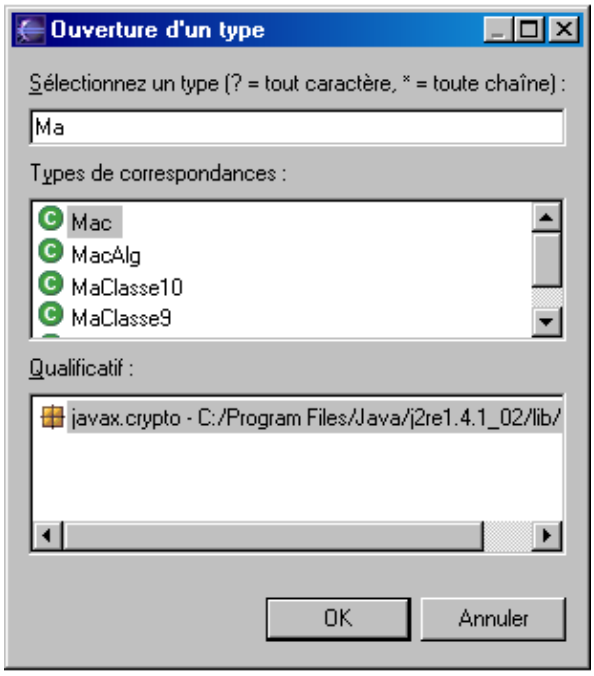


Il est possible de sélectionner tout ou partie des modifications en les cochant.

Un clic sur le bouton "Fin" met en oeuvre les modifications.

6.10. Ouverture d'un type

Le bouton  de la barre d'outils permet de lancer l'outil " Ouverture d'un type ". Cet outil est particulièrement pratique pour rechercher et ouvrir dans l'éditeur le code d'une classe dont on connaît tout ou partie du nom.



Il suffit de saisir le début du nom de la classe ou de l'interface pour que la liste des entités répondant au critère se construise de façon incrémentale.

Il est aussi possible de saisir un motif à l'aide des caractères ? pour représenter un caractère quelconque unique et * pour représenter aucun ou plusieurs caractères quelconques.

La zone qualificatif affiche le ou les packages ou l'entité est définie. Ce nom de package est suivi du nom du projet si l'entité est définie dans l'espace de travail ou du nom du fichier qui contient la version compilée pour une entité externe.

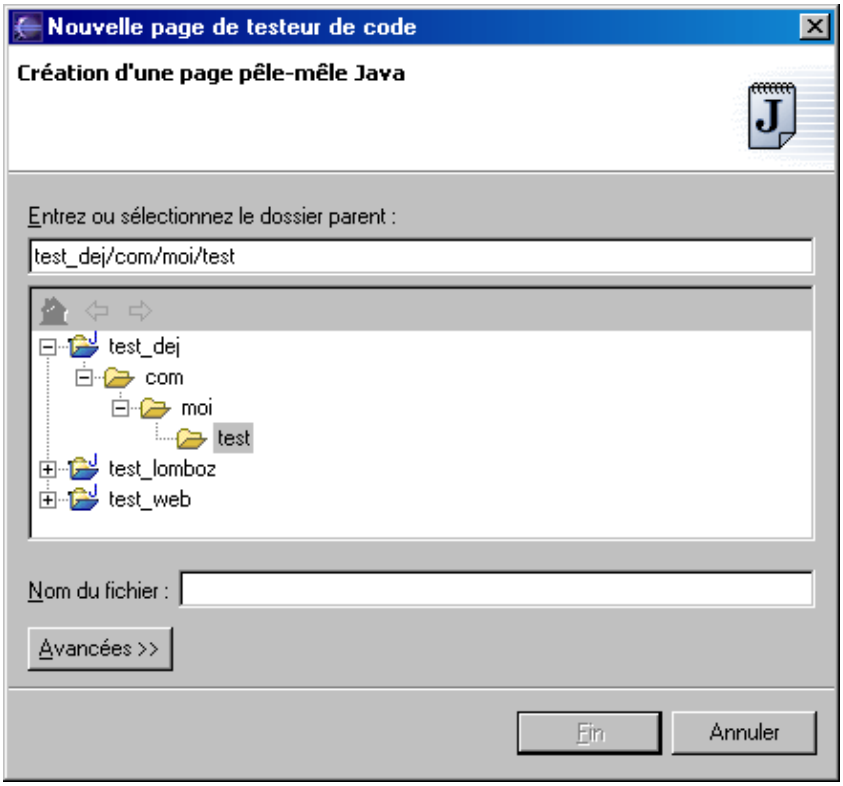
Une fois l'élément voulu sélectionné, un clic sur le bouton " OK " ouvre l'éditeur avec cette entité.

6.11. Utilisation du scrapbook

Le scrapbook, traduit par " page de testeur de code ", est une fonctionnalité qui permet de tester des morceaux de code dans une machine virtuelle. Cette fonctionnalité très intéressante était déjà présente dans l'outil Visual Age d'IBM.

Pour pouvoir l'utiliser, il faut créer une nouvelle "page de testeur de code", en utilisant une des possibilités d'Eclipse pour créer une nouvelle entité.

Comme pour la création de toute nouvelle entité, un assistant permet de recueillir les informations nécessaires à la création du scrapbook.



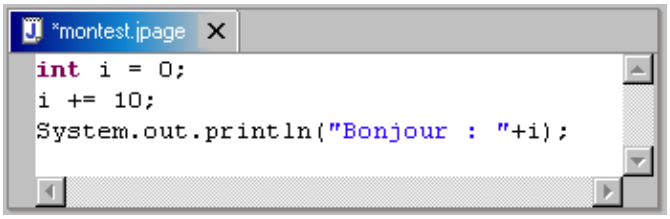
Une "page de testeur de code" est physiquement un fichier contenant du code java et ayant une extension .jpage

La seule page de l'assistant permet de sélectionner le répertoire qui va contenir le fichier ainsi que son nom. Par défaut, l'extension .jpage est ajoutée.

Un clic sur le bouton "Fin" permet de générer le fichier et d'ouvrir l'éditeur avec son contenu.

Le grand avantage est de pouvoir tester des morceaux de code sans avoir à créer une classe et une méthode main() et de bénéficier de fonctionnalités particulières pour tester ce code.

Exemple :



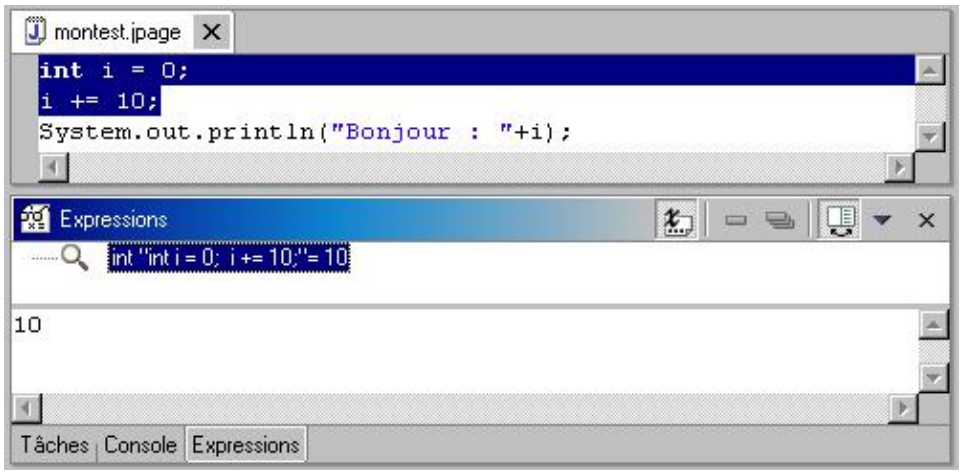
Lors de l'édition du code contenu dans le scrapbook, la barre d'outils est enrichie de plusieurs boutons qui permettent d'utiliser les principales fonctionnalités du scrapbook.

Bouton	Rôle
	Permet d'exécuter un morceau de code et d'évaluer le résultat de l'exécution
	Permet d'afficher dans l'éditeur de code, le résultat de l'exécution d'un morceau de code
	Permet d'exécuter un morceau de code et d'afficher le résultat dans la console
	Permet d'arrêter l'exécution dans la machine virtuelle
	Permet de définir les importations nécessaires

Les trois premiers boutons ne sont utilisables que si un morceau de code est sélectionné dans le scrapbook. Le quatrième bouton n'est utilisable que si une machine virtuelle exécute du code du scrapbook.

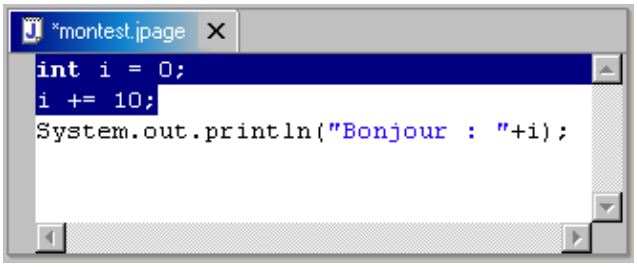
La fonction " Inspecter " permet de visualiser, dans la vue " Expressions ", les valeurs des objets contenus dans le code sélectionné.

Pour la mettre en oeuvre, il suffit de sélectionner un morceau de code dans l'éditeur du scrapbook et de cliquer sur le bouton  ou d'utiliser l'option " Inspecter " du menu contextuel.

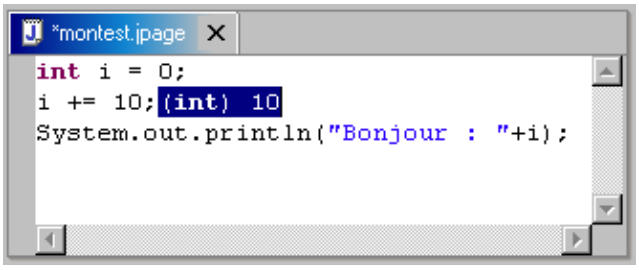


La fonction " Affichage du résultat de l'exécution " permet d'exécuter un morceau de code et d'afficher le résultat de l'exécution dans l'éditeur juste après la fin de la sélection du morceau de code.

Pour mettre en oeuvre cette fonctionnalité, il faut sélectionner un morceaux de code dans l'éditeur du scrapbook.

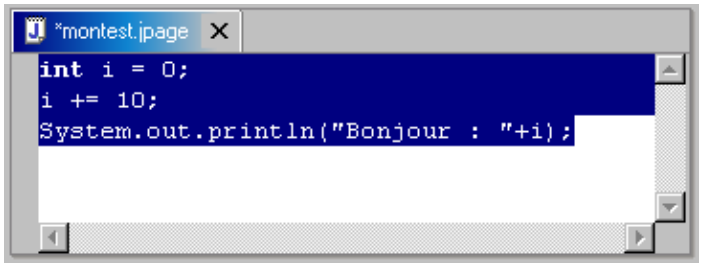


Il faut ensuite cliquer sur le bouton  ou d'utiliser l'option " Afficher " du menu contextuel.

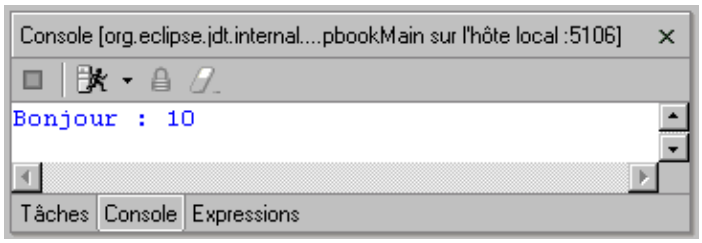


L'affichage insère le type entre parenthèse et la valeur du résultat dans l'éditeur.

La fonction " Exécuter " permet d'exécuter d'un morceau de code et d'afficher le résultat dans la vue " Console ". Pour mettre en oeuvre cette fonctionnalité, il faut sélectionner un morceau de code dans l'éditeur du scrapbook.



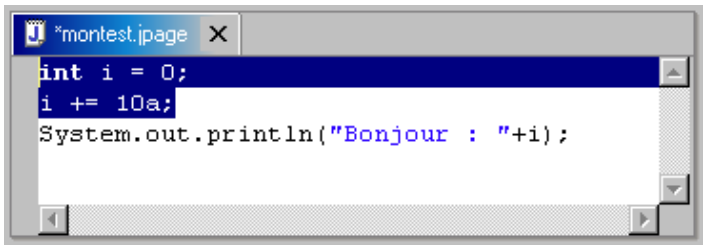
Il faut ensuite cliquer sur le bouton  ou utiliser l'option " Exécuter " du menu contextuel.



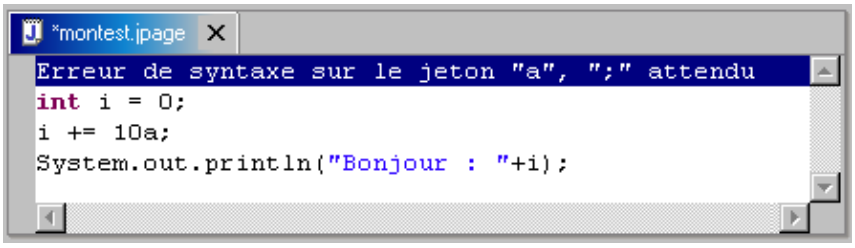
Lors de l'exécution de code dans le scrapbook, une machine virtuelle dédiée est lancée pour cette exécution. Pour pouvoir relancer une exécution, il faut arrêter le machine virtuelle précédemment lancée. L'arrêt de cette machine virtuelle peut se faire en cliquant sur le bouton  .

Lors de l'exécution de code dans le scrapbook, si une erreur de syntaxe est détectée, celle ci est signalée directement dans le code de l'éditeur

Exemple :

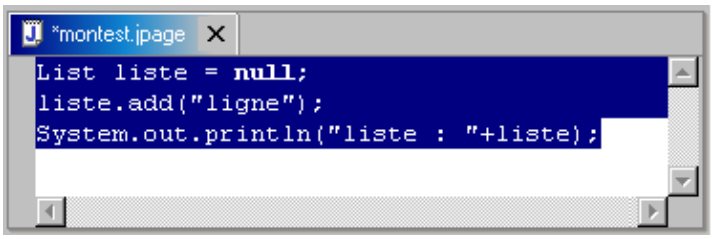


Lors de l'exécution de ce morceau de code, l'erreur suivante est affichée dans l'éditeur

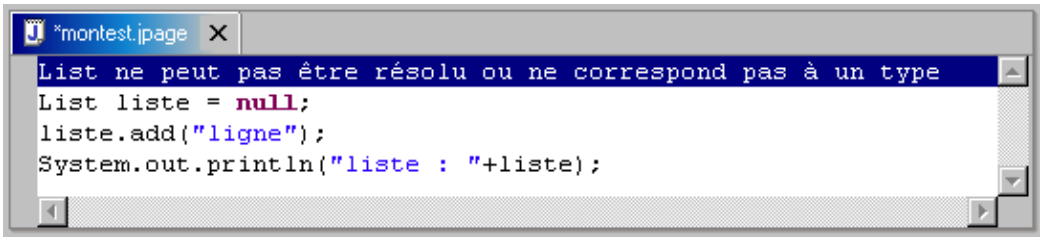


La structure d'un fichier java classique n'étant pas respecter dans le scrapbook, la gestion des clauses d'import est gérée de façon particulière.

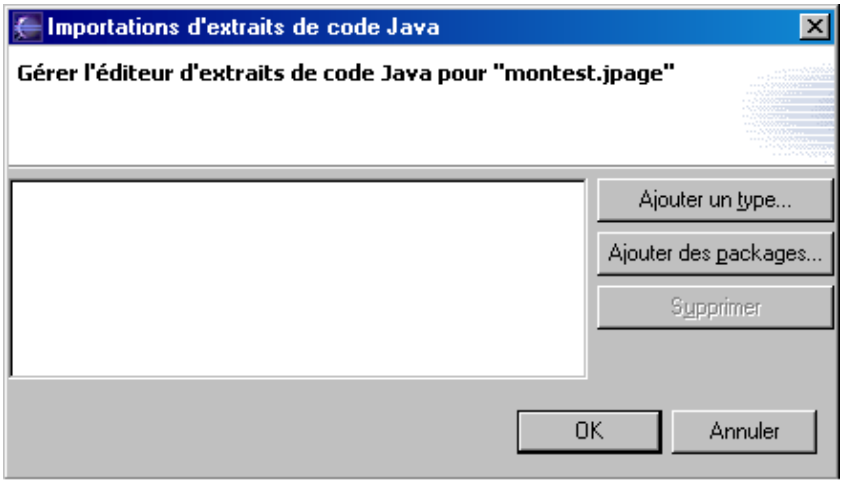
Exemple :



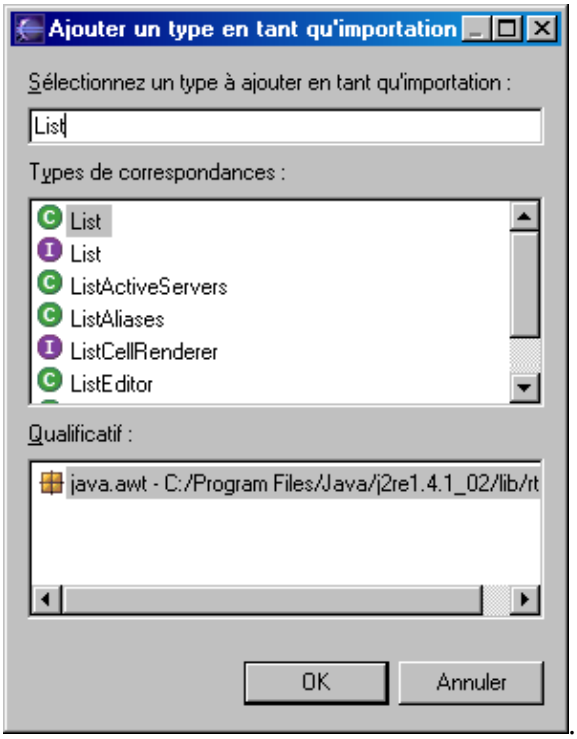
L'exécution de ce morceau de code génère l'erreur suivante :



Pour ajouter une clause import, il cliquer sur le bouton  ou utiliser l'option " Définit les importations " du menu contextuel.

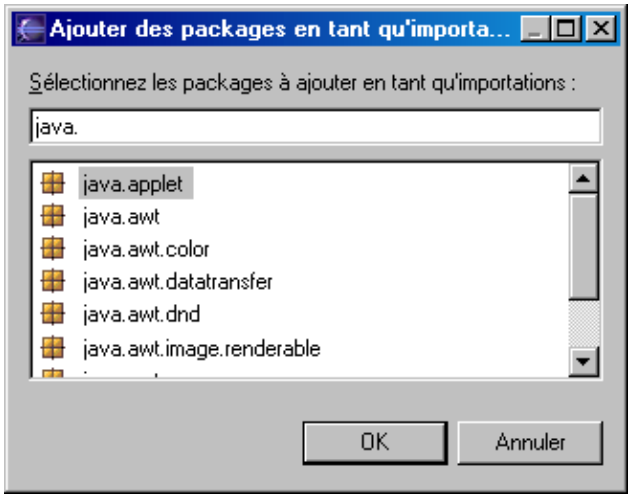


Un clic sur le bouton " Ajouter un type " permet d'importer une classe ou une interface dans le scrapbook.



La zone de saisie du type permet une recherche incrémentale dans toutes les entités définies dans le chemin de compilation du projet.

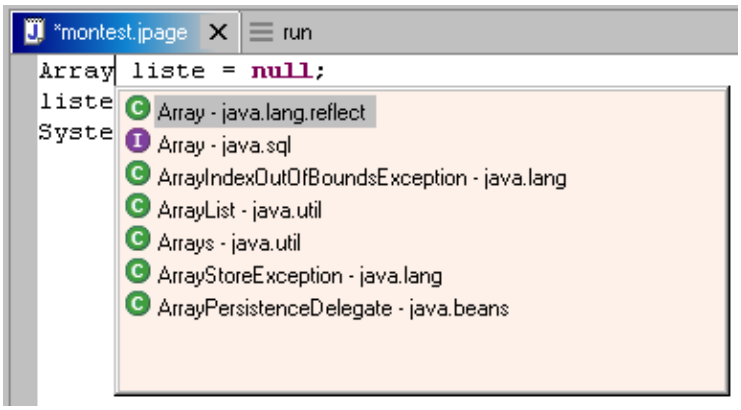
Un clic sur le bouton " Ajouter un Package " permet d'importer un package



La zone de saisie du package permet une recherche incrémentale du package désiré parmi tous ceux définis dans le chemin de compilation du projet.

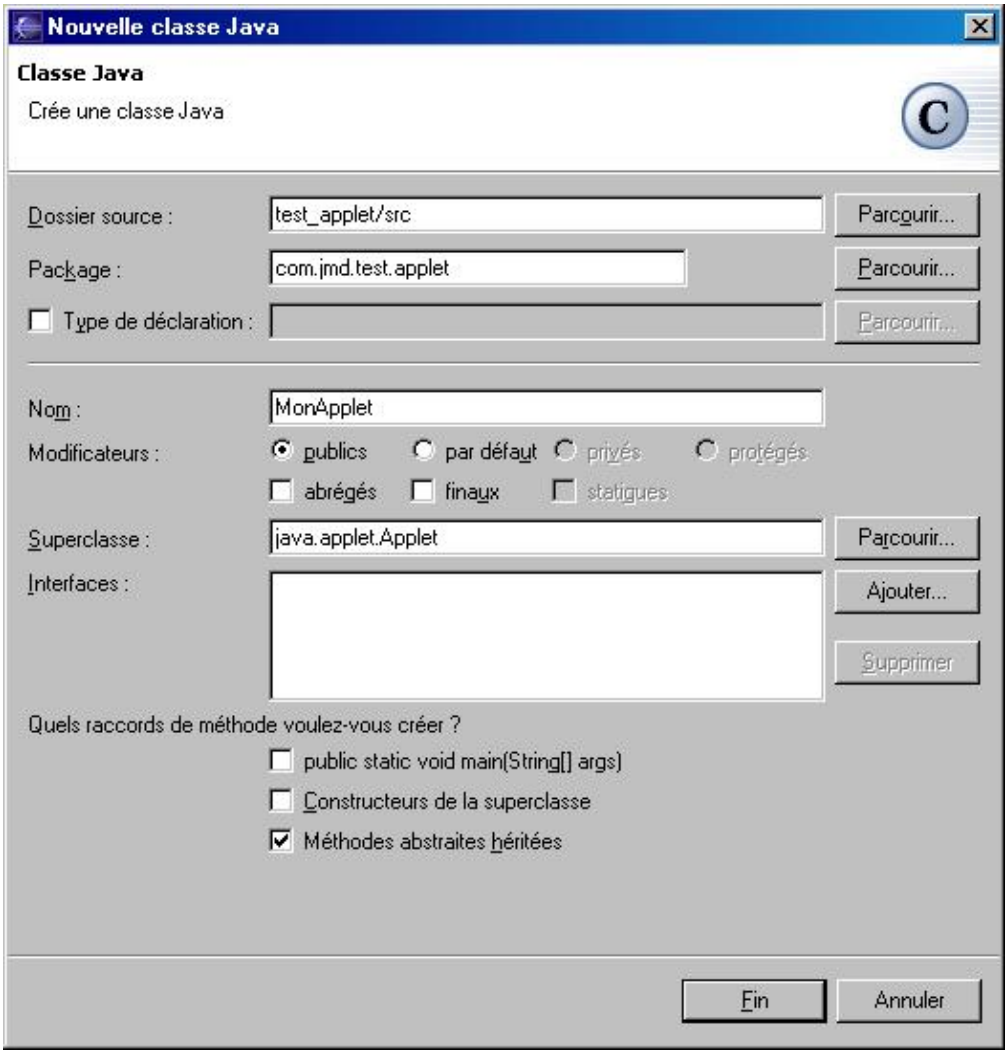
Enfin pour supprimer l'importation d'une entité, il suffit de la sélectionner et de cliquer sur le bouton " Supprimer ".

L'assistant de code, est bien sûre disponible dans l'éditeur du scrapbook toujours en utilisant la combinaison de touches "Ctrl" + "Espace".



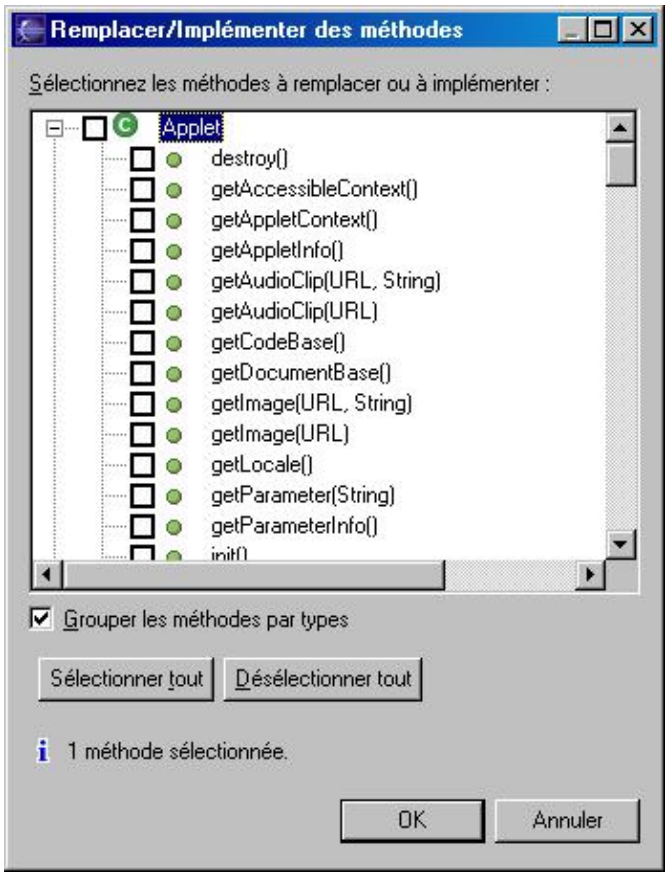
6.12. Le développement d'applets

Il faut créer un nouvelle classe qui hérite de la classe java.applet.Applet



La nouvelle classe est créée et l'éditeur de code ouvre son source.

Pour faciliter la réécriture des méthodes utiles, il est possible d'utiliser l'option « source / Remplacer/Implémenter les méthodes ... » du menu contextuel de l'éditeur de code.



Il suffit de cocher chaque méthode désirée et de cliquer sur le bouton « Ok ».

Le code source est enrichi avec la ou les méthodes sélectionnées :

Exemple :

```
/* (non-Javadoc)
 * @see java.awt.Component#paint(java.awt.Graphics)
 */
public void paint(Graphics arg0) {
  // TODO Raccord de méthode auto-généré
  super.paint(arg0);
}
```

Il suffit d'insérer le code dans ces méthodes.

Par exemple :

Exemple :

```
/* (non-Javadoc)
 * @see java.awt.Component#paint(java.awt.Graphics)
 */
public void paint(Graphics arg0){
  super.paint(arg0);
  arg0.drawString("Bonjour",50,50);
}
```

Pour exécuter une applet, il faut utiliser l'option « Exécuter en tant que / Applet » du bouton « Exécuter » de la barre d'outils.

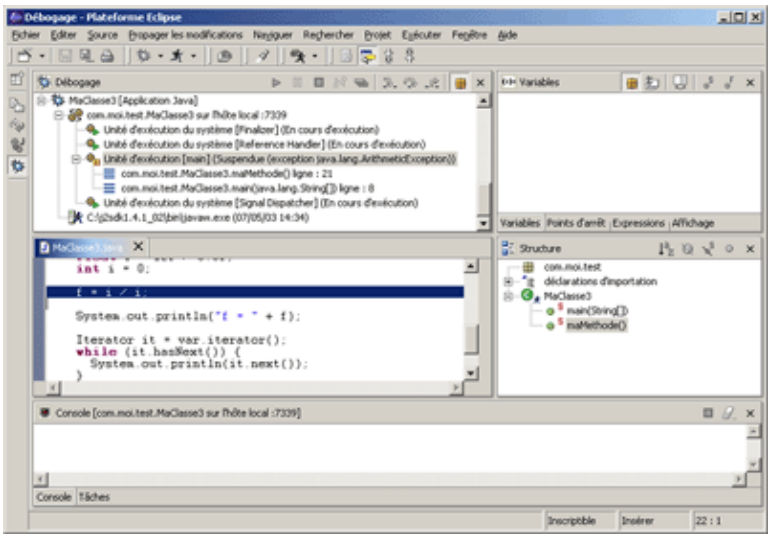

AppletViewer : com....
Applet
Bonjour
Applet démarré.

7. Déboguer du code Java

7.1. La perspective "Debogage"

Pour déboguer du code Java, Eclipse propose une perspective dédiée : la perspective "Débogage".

Celle ci est automatiquement affichée lorsqu'une application est lancée sous le contrôle du débogueur en utilisant le bouton  de la barre d'outils. Son principe de fonctionnement est identique au bouton d'exécution situé juste à côté de lui.



Par défaut, la perspective "Débogage" affiche quelques vues aussi présentes dans la perspective Java (les vues "Structure" et "Console") ainsi que l'éditeur de code Java.

Elle affiche aussi plusieurs vues particulièrement dédiées au débogage.

7.2. Les vues spécifiques au débogage

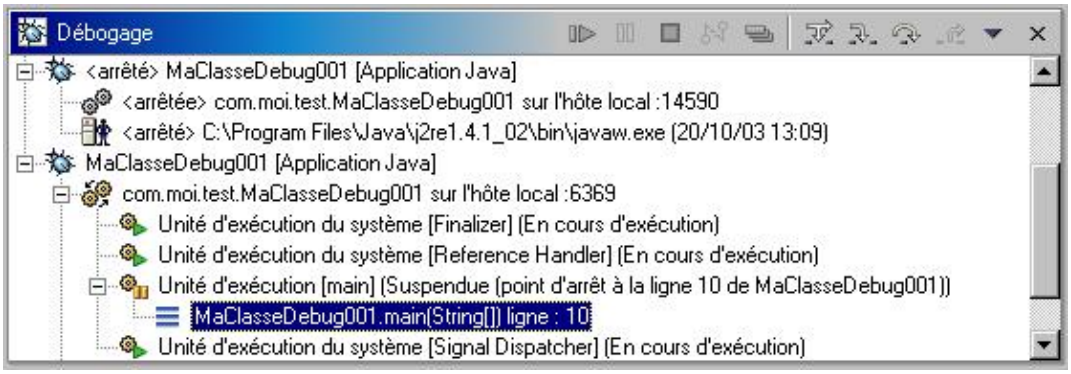
Les vues spécifiques au débogage sont :

- la vue "Débogage" : affiche sous la forme d'une arborescence, les différents processus en cours d'exécution ou terminés
- la vue "Variables" : affiche les variables utilisées dans les traitements en cours de débogage
- la vue "Points d'arrêts" : affiche la liste des points d'arrêts définis dans l'espace de travail
- la vue "Expressions" : permet d'inspecter une expression en fonction du contexte des données en cours d'exécution

- la vue "Affichage" : permet d'afficher le résultat de l'évaluation d'une expression

7.2.1. La vue "Débogage"

Cette vue affiche sous la forme d'une arborescence, les différents processus en cours d'exécution ou terminés.



Cette vue possède plusieurs icônes qui permettent d'agir sur les éléments affichés.

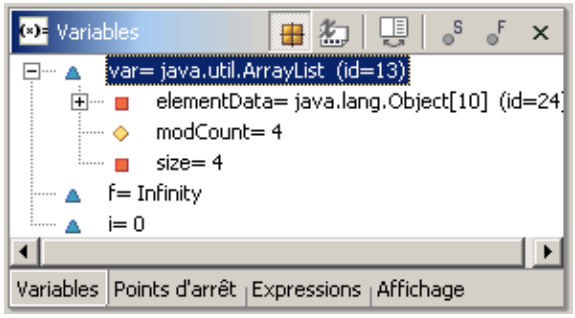
Icône	Rôle
	Reprendre l'exécution précédemment interrompue
	Interrompre l'exécution du processus
	Demande de mettre fin au processus
	Enlève de la liste tous les processus qui sont terminés
	
	Exécute la ligne courante et arrête l'exécution sur la première ligne de code incluse dans la première méthode de la ligne courante (raccourci : F5)
	Exécute la ligne courante et arrête l'exécution avant la ligne suivante (raccourci : F6)
	Exécute le code de la ligne courante jusqu'à la prochaine instruction return de la méthode (raccourci : F7)
	Affiche ou non le nom pleinement qualifiés des objets

L'exécution d'un processus est automatiquement suspendu dans les cas suivants :

- un point d'arrêt est rencontré
- une exception non capturée est propagée jusqu'au sommet de la pile d'appel

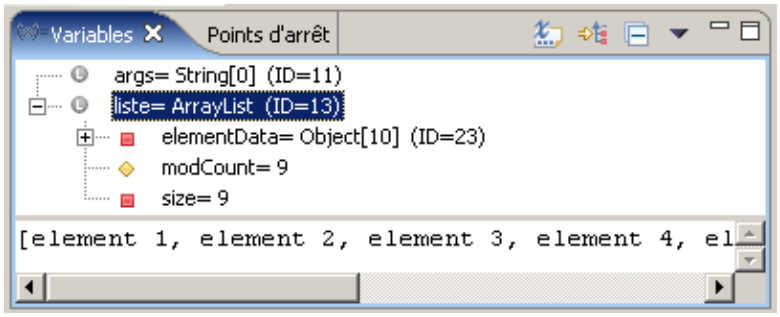
7.2.2. La vue "Variables"

Cette vue permet de visualiser les valeurs des variables utilisées dans les traitements en cours de débogage. Ces valeurs peuvent être unique si la variable est de type primitive ou former une arborescense contenant chacun des champs si la variable est un objet.

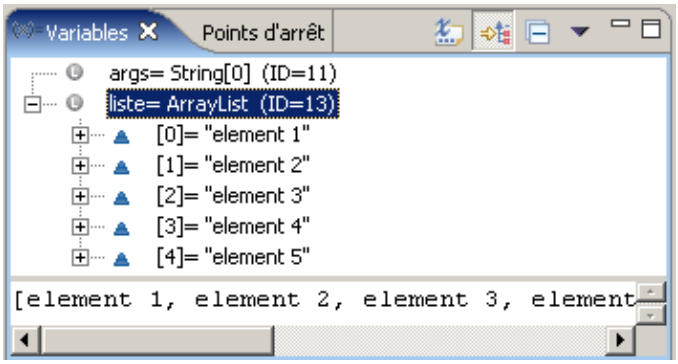


3.0

La liste des éléments d'une collection est affichée dans la vue « Variable » si la variable inspectée est une collection.

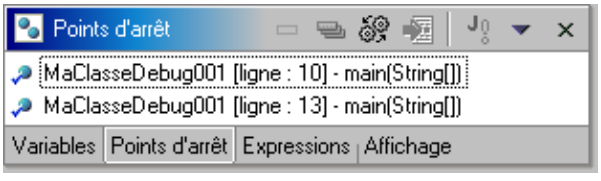


Le bouton  permet d'afficher le contenu des éléments de la collections.



7.2.3. La vue "Points d'arrêts"

Cette vue permet de recenser tous les points d'arrêts définis dans l'espace de travail.

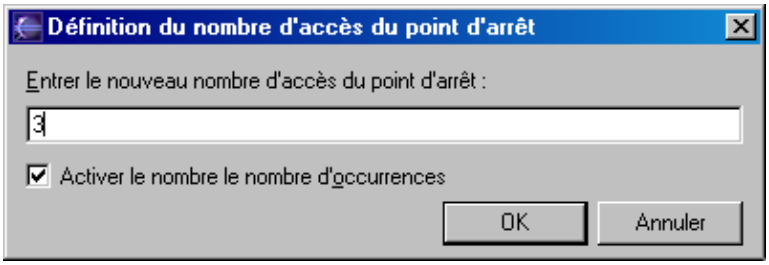


Un double clic sur un des points d'arrêts permet d'ouvrir l'éditeur de code directement sur la ligne ou le point d'arrêt est défini. Cette action est identique à l'utilisation de l'option "Accéder au fichier" du menu contextuel ou à un clic sur le bouton  de la barre de titre de la vue.

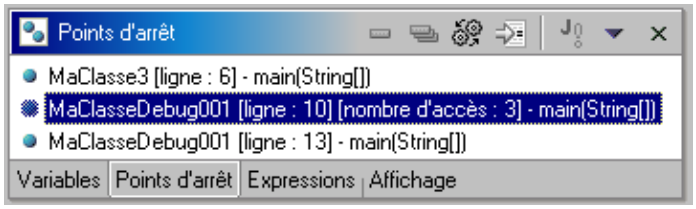
Il est possible grâce au menu contextuel de contrôler les points d'arrêts.

L'option "Nombres d'occurences" permet de préciser le nombre fois ou le point d'arrêt exécuté sera ignoré avant qu'il n'intérrompe l'exécution. Ceci est particulièrement pratique lorsque l'on débogue du code contenu dans une boucle et que l'on connait l'itération qui pose problème.

Lors du clic sur l'option "Nombres d'occurences", une boîte de dialogue permet de préciser le nombre de passage sur le point d'arrêt à ignorer.



Ce nombre est indiqué dans la vue "Points d'arrêts", précédé du libellé "nombre d'accès".

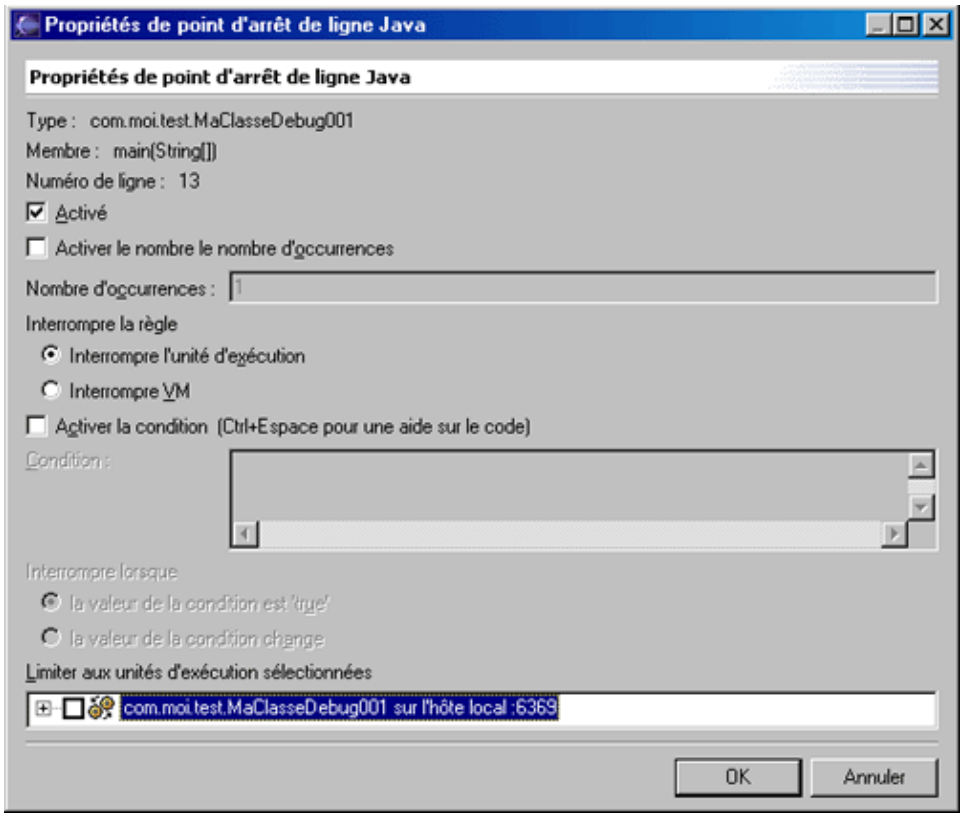


Il est possible d'activer ou de désactiver le point d'arrêt sélectionné respectivement grace à l'option "Activer" ou "Désactiver".

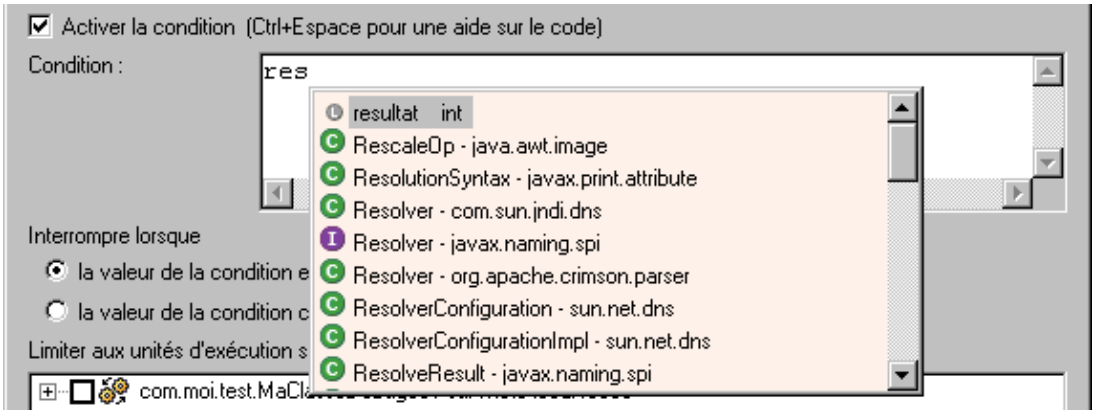
L'option "Supprimer" permet de supprimer le point d'arrêt sélectionné. Il est aussi possible d'utiliser le bouton de la barre de titre de la vue.

L'option "Supprimer tout" permet de supprimer tous les points d'arrêts définis. Il est aussi possible d'utiliser le bouton de la barre de titre de la vue.

L'option "Propriétés ..." permet d'ouvrir une boîte de dialogue pour régler les différents paramètres du point d'arrêt sélectionné.

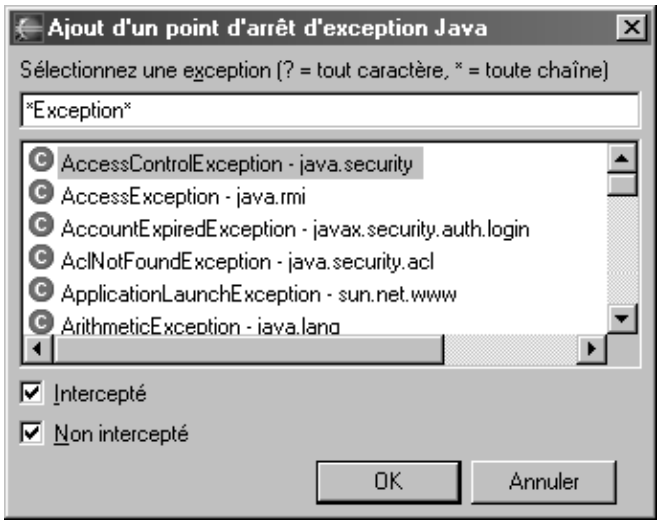


Un de ces paramètres les plus intéressants est la possibilité de mettre une condition d'activation du point d'arrêt. Il suffit pour cela de cocher la case "Activer la condition" et de la saisir dans la zone de texte prévue à cet effet. Dans cette zone de texte, l'assistant de complétion de code est utilisable.



Dans cette vue, il est aussi possible de demander l'arrêt de l'exécution non pas à l'exécution d'une ligne de code mais à la lever d'une exception particulière. Pour cela, il suffit de cliquer sur le bouton  .

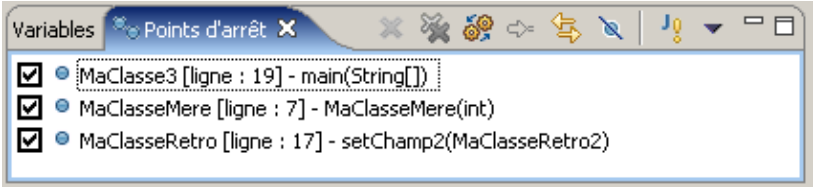
Une boîte de dialogue permet de sélectionner l'exception concernée.



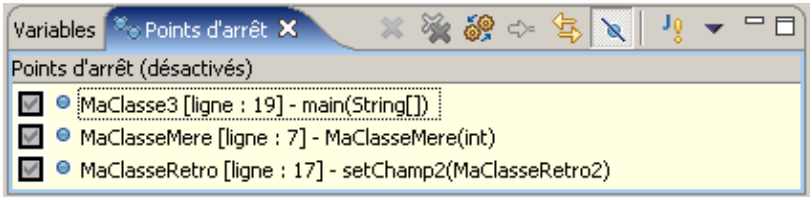
Le nouveau point d'arrêt est affiché dans la vue "Point d'arrêts".

Il est possible de préciser si l'arrêt se fait sur une exception interceptée, non intercepté ou dans les deux cas. Ce type de point d'arrêt possède les mêmes propriétés que les points d'arrêts liés à une ligne de code.

Une case à cocher devant chaque point d'arrêt permet d'activer ou non un point d'arrêt.



Le bouton  permet de demander la désactivation de tous les points d'arrêts enregistrés dans l'espace de travail.

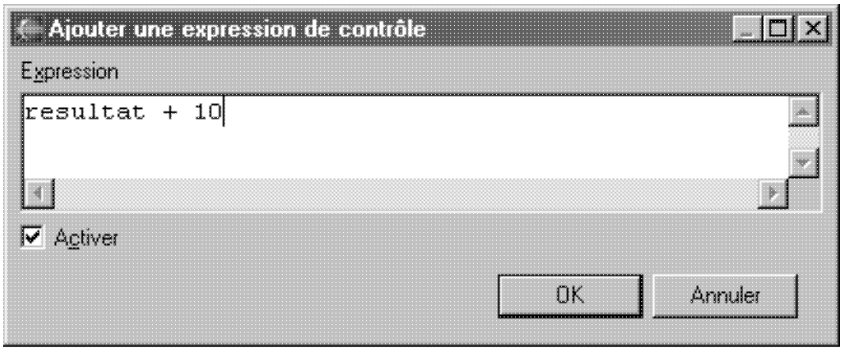


7.2.4. La vue "Expressions"

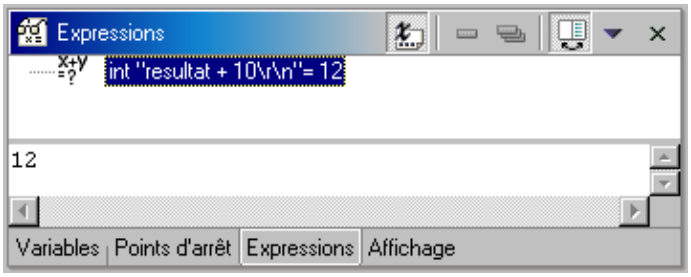
La vue "Expressions" permet d'inspecter la valeur d'une expression selon les valeurs des variables dans les traitements en cours de débogage.

Pour ajouter une nouvelle expression, il suffit d'utiliser l'option  "Ajouter une expression de contrôle Java" du menu contextuel.

Une boite de dialogue permet de saisir l'expression.



La vue "Expressions" affiche le résultat de l'évaluation en tenant compte du contexte d'exécution.



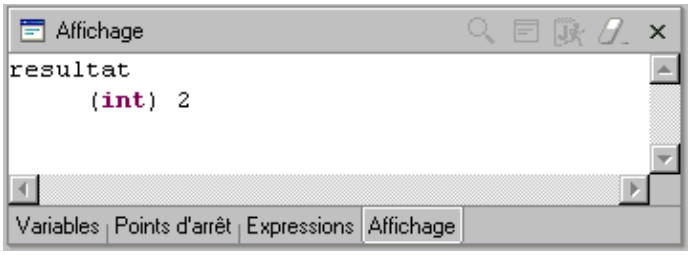
L'option "Réévaluer l'expression de contrôle" permet de recalculer la valeur de l'expression.

L'option "Editer l'expression de contrôle" permet de modifier l'expression.

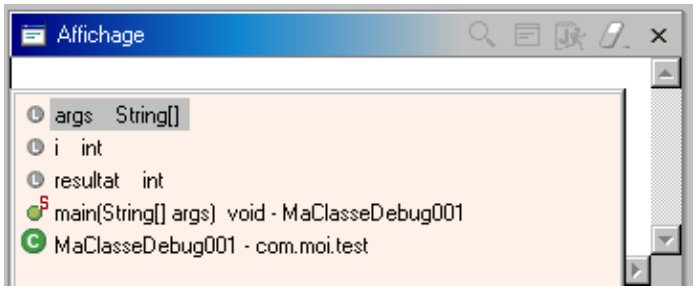
7.2.5. La vue "Affichage"

La vue "Affichage" permet d'afficher le résultat de l'évaluation d'une expression.

La valeur de l'expression à afficher peut avoir plusieurs origines. Le plus pratique est de sélectionner un morceau de code dans l'éditeur et d'utiliser l'option "Afficher" du menu contextuel. Le résultat de l'évaluation est affiché dans la vue "Affichage".



Il est aussi possible d'ajouter directement une expression dans la vue en utilisant l'option "Assistant de contenu" du menu contextuel.



Il faut sélectionner le membre ou la variable concerné.

Pour obtenir des informations sur sa valeur, il suffit de sélectionner l'expression dans la vue. Diverses actions sont alors disponibles avec les boutons dans la barre de titre de la vue ou dans les options du menu contextuel :

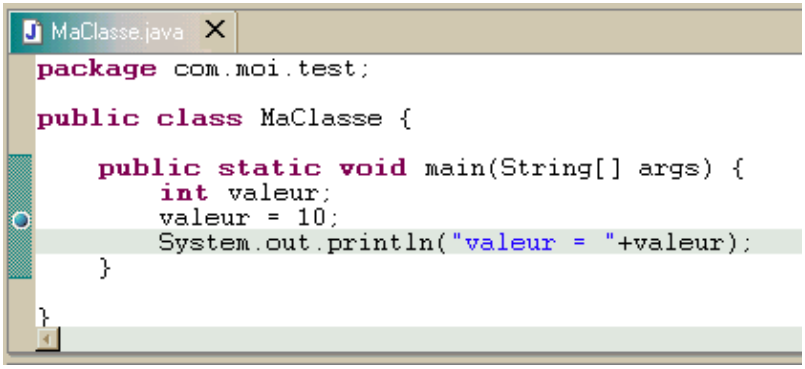
- inspecter : exécution et évaluation dans la vue "Expressions"
- afficher : afficher le résultat dans la vue "Affichage"
- exécuter : exécuter l'expression sélectionnée. Il est ainsi possible de modifier la valeur d'une variable

7.3. Mise en oeuvre du débogueur

La mise en oeuvre du débogueur reste comparable à celle proposée par d'autres IDE : mise en place d'un point d'arrêt, exécution plus ou moins rapide dans le débogueur pour arriver à cibler le problème.

7.3.1. Mettre en place un point d'arêt

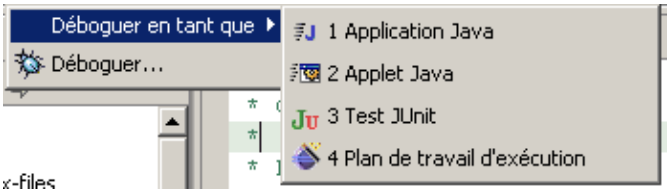
Pour placer un point d'arrêt, il suffit dans l'éditeur de double cliquer dans la barre de gauche pour faire apparaître une icône ronde bleue.



7.3.2. Exécution dans le débogueur

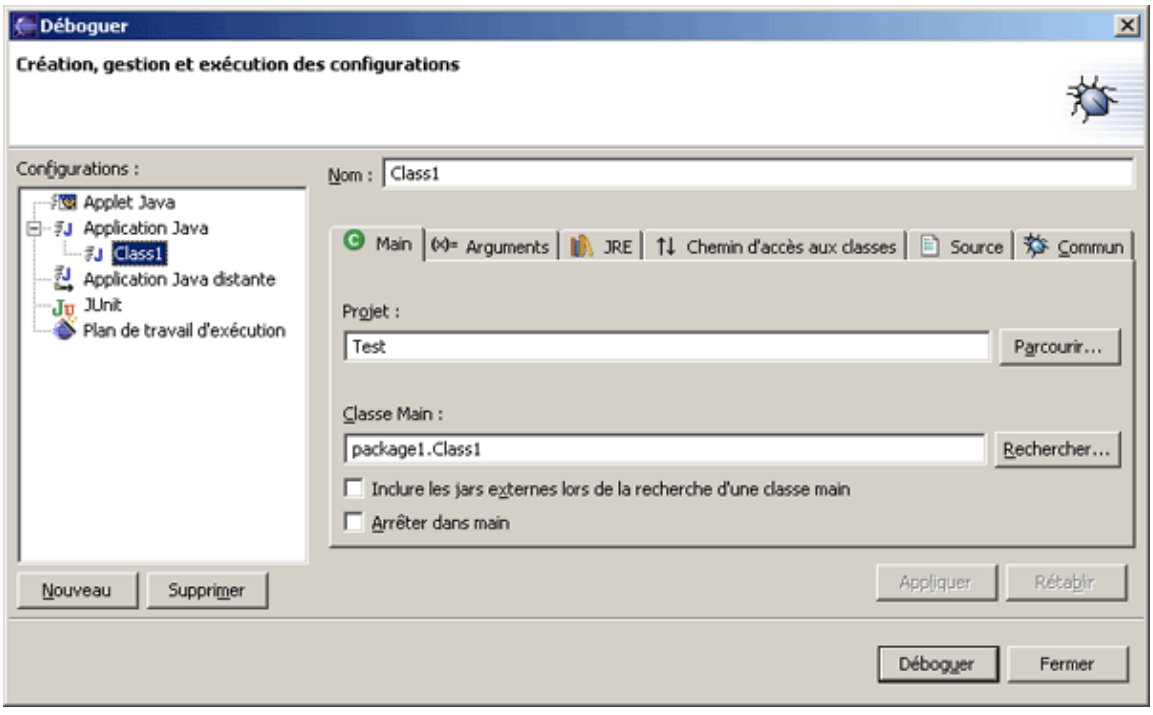
Pour lancer l'exécution dans le débogueur, il faut ouvrir le menu en cliquant sur flèche du bouton 

Un menu déroulant propose de déboguer les dernières applications exécutées ou de lancer le débogage de la source courante en proposant deux options de menu :



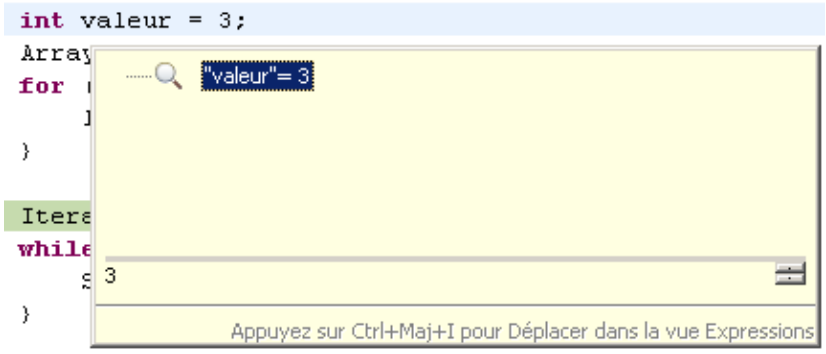
Le plus simple pour lancer le débogage d'une application est de sélectionner l'option « Déboguer en tant que / Application Java »

L'option "Déboguer ... " permet de fournir des paramètres précis en vue de l'exécution d'une application sous le débogueur. Un assistant permet de sélectionner la classe et de préciser les paramètres.



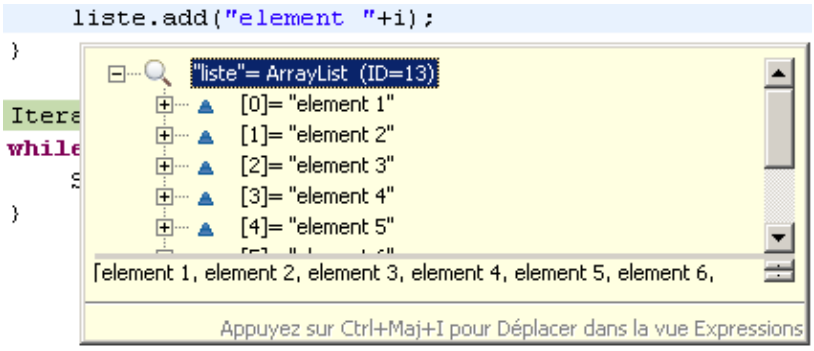
Il ne reste plus qu'à mettre en oeuvre les différentes fonctionnalités proposées par les vues de la perpective pour déboguer le code.

Lors d'une demande d'inspection d'une variable dans l'éditeur affichant le code, une bulle d'aide affiche les valeurs de la variable.



Pour accéder à la vue expression, il suffit de réutiliser la combinaison de touche Ctrl+Maj+I.

Dans la fenêtre affichant le code, l'inspection d'une variable ouvre une bulle d'aide qui affiche en popup les informations sur la collection



8. Le refactoring

Eclipse intègre de puissantes fonctionnalités pour faciliter le refactoring. Le refactoring consiste à modifier la structure d'un fichier source et si nécessaire à propager ces modifications dans les autres fichiers sources pour maintenir autant que possible l'intégrité du code existant.

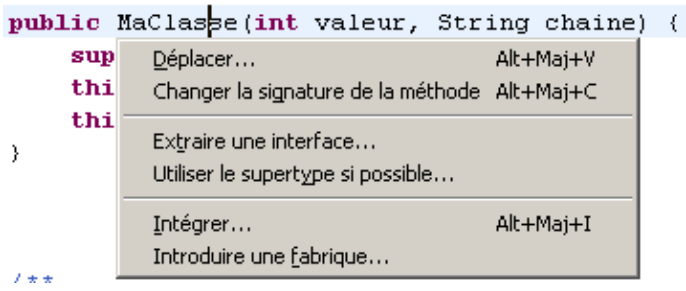
Un menu nommé « Propager les modifications » permet d'utiliser certaines de ces fonctionnalités : il est présent dans le menu principal et dans de nombreux menus contextuels. Chacune de ces fonctionnalités est applicable sur un ou plusieurs types d'entités selon le cas : projets, classes et membres d'une classe. Certaines de ces fonctionnalités possèdent un raccourci clavier.

Type de modification	Fonctionnalités	Entité(s) concernée(s)	Raccourci clavier
Structure du code	Renommer	projets, packages, classes, champs, méthodes, variables locales, paramètres	Alt+Maj+R
	Déplacer	projets, packages, classes, méthodes et champs statiques	Alt+Maj+V
	Changer la signature de la méthode	méthodes	
	Convertir une classe anonyme en classe imbriquée	classes anonymes	
	Convertir un type imbriqué au niveau supérieur	classes imbriquées	
Structure au niveau de la classe	Transférer	méthodes ou champs	
	Extraire	méthodes ou champs	
	Extraire une interface	classes	
	Utiliser le supertype si possible	classes	
Structure à l'intérieure d'une classe	Intégrer	méthodes, constantes et variable locales	Alt+Maj+I
	Extraire la méthode	morceau de code sélectionné	Alt+Maj+M
	Extraire la variable locale	morceau de code sélectionné pouvant être transformé en variable	Alt+Maj+L
	Extraire une constante		

		morceau de code sélectionné pouvant être transformé en constante	
	Convertir la variable locale en zone	variables locales	
	Encapsuler la zone	champs	

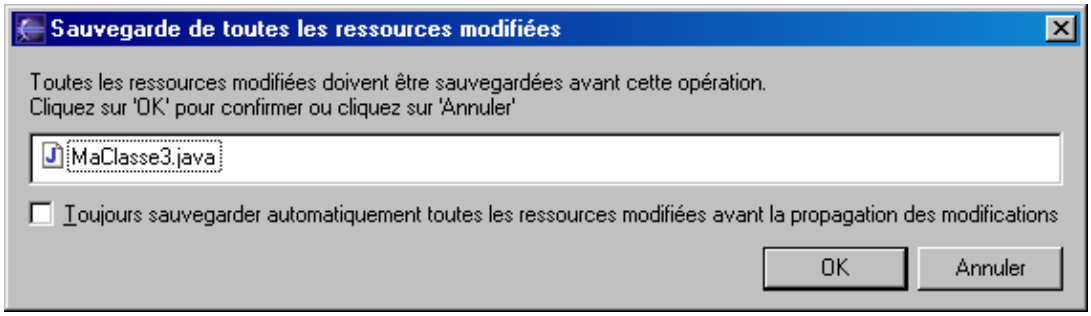
L'accès aux fonctions de refactoring est contextuel : seules les fonctions applicables dans le contexte courant peuvent être activées.

L'éditeur propose un menu contextuel pour accéder rapidement aux fonctions de refactoring en utilisant la combinaison de touche Alt+Maj+T



Les fonctions proposées le sont en fonction du contexte (le type d'entité sur lequel le curseur est placé).

Avant de pouvoir utiliser ces fonctionnalités, il faut sauvegarder l'élément qui sera modifié sinon un message d'erreur est affiché.

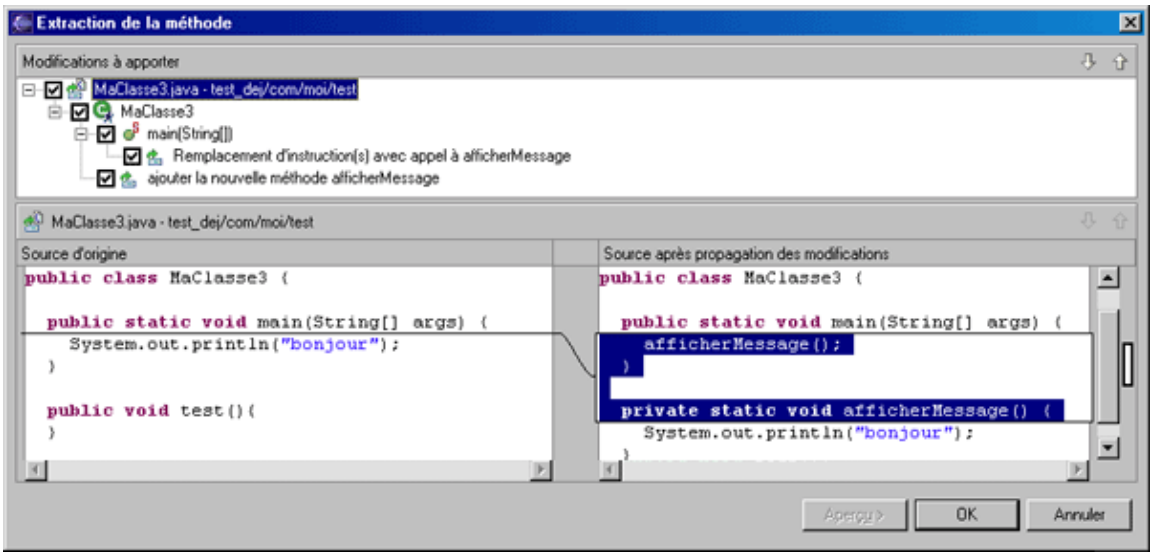


L'usage de ces fonctions utilise toujours le même principe de fonctionnement :

1. sélection de l'élément concerné par la fonction de refactoring
2. appel de la fonction en utilisant l'option du menu « Propager les modifications »
3. renseigner les informations utiles à la fonction dans la boîte de dialogue
4. pré−visualiser et valider individuellement les modifications qui seront apportées
5. demander la mise en oeuvre des modifications en cliquant sur « Ok » (ceci peut être fait sans demander la prévisualisation)

Les fonctionnalités du refactoring sont accessibles à partir des vues "Packages" et "Structure" ou de l'éditeur de code Java selon l'entité concernée par la modification. Dans les vues, il suffit de sélectionner le ou les éléments concernés. Dans l'éditeur de code, il faut sélectionner le nom de l'élement ou de positionner le curseur dessus avant d'appeler la fonctionnalité.

Toutes les fonctionnalités utilisent un assistant qui propose toujours à la fin un bouton « Apercu > » permettant d'ouvrir une boîte de dialogue qui permet de pré−visualiser chacune des modifications résultant de l'usage de la fonction.



Une arborescence permet d'afficher chacun des éléments qui sera modifié ainsi que toutes les modifications pour ces éléments. Il est possible de cocher ou non tout ou partie des modifications pour qu'elles soient appliquées ou non.

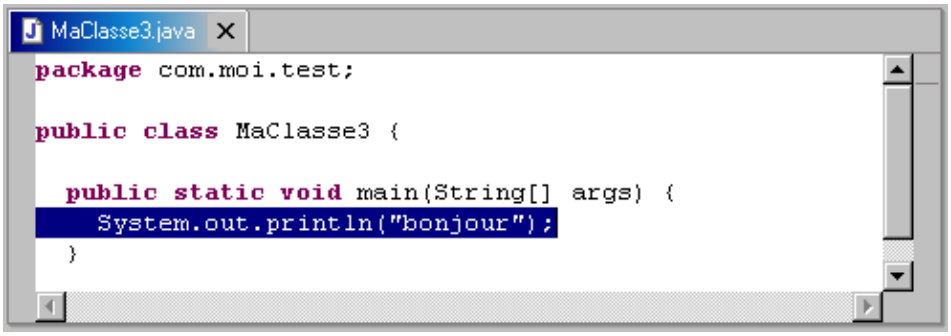
La partie inférieure présente, pour l'élément sélectionné, le code source actuel à gauche et le code source tel qu'il sera après l'application de la fonction à droite.

Un clic sur le bouton « OK » permet de mettre en oeuvre toutes les modifications qui sont cochées.

8.1. Extraction d'une méthode

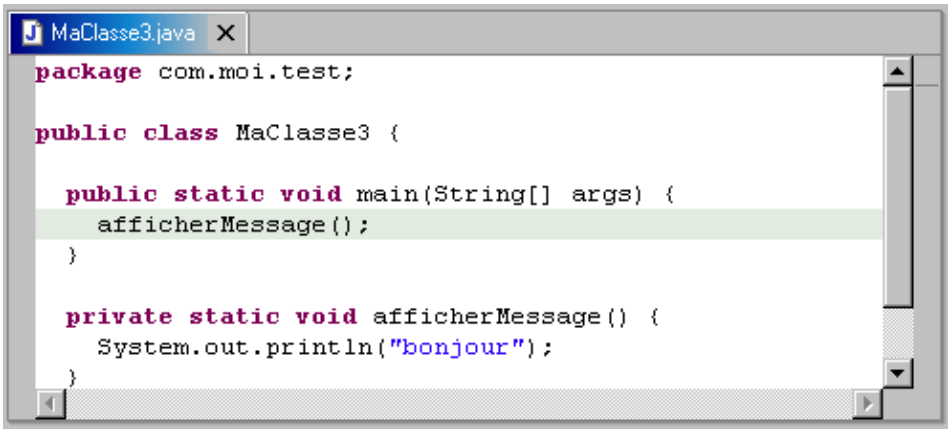
Cette fonction permet de transférer un morceau de code dans une méthode qui sera créée pour l'occasion.

Pour l'utiliser, il faut sélectionner le morceau de code.

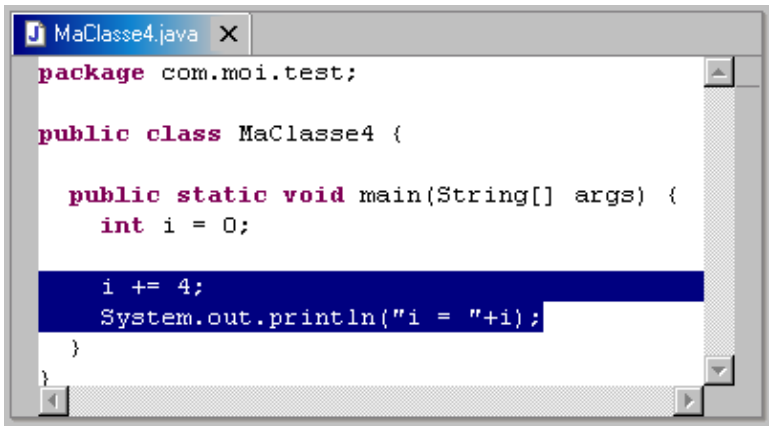


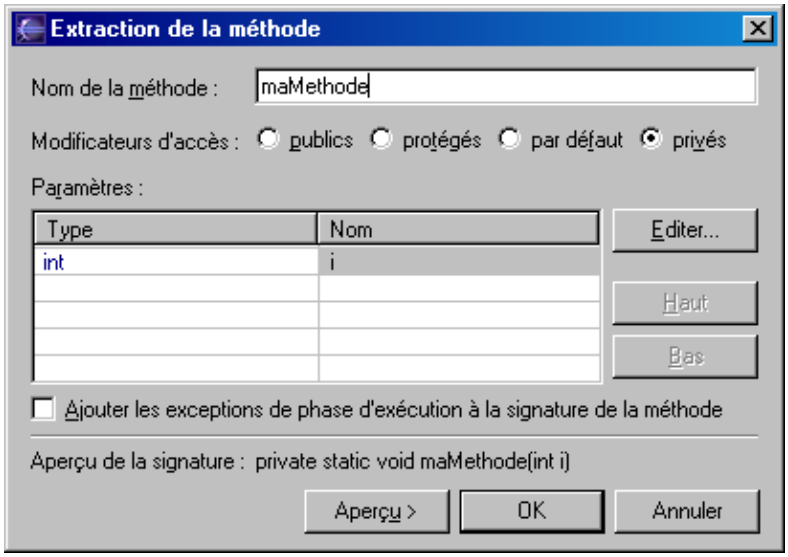
Il faut sélectionner l'option « Extraire la méthode ... » du menu principal ou du menu contextuel « Propager les modifications ». Une boite de dialogue permet de saisir le nom de la méthode et de sélectionner le modificateur d'accès.

Un clic sur le bouton « OK » permet de mettre en oeuvre automatiquement les modifications.



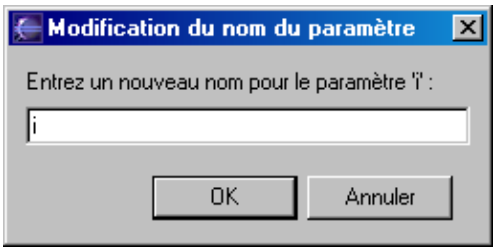
Dans le cas ou le code sélectionné contient une ou plusieurs variables, la boîte de dialogue contient en plus la liste des variables détectées dans le code.



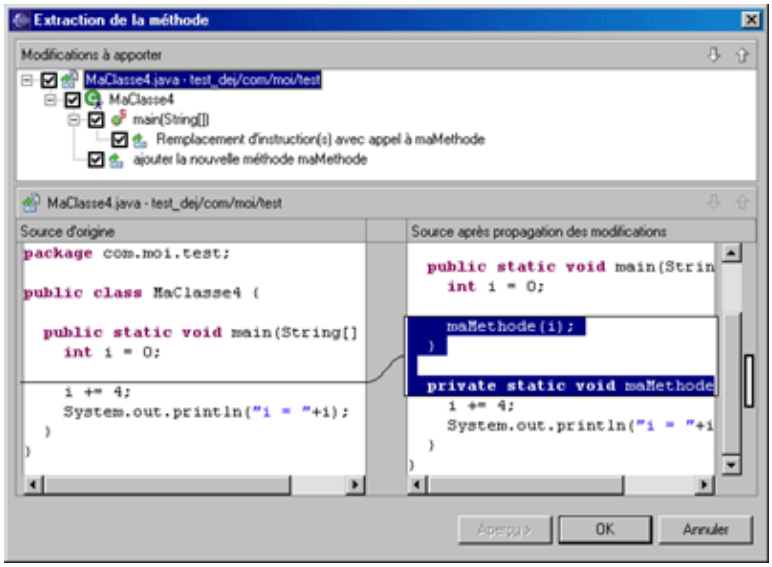


Il faut saisir le nom de la méthode, sélectionner son modificateur d'accès et éventuellement modifier les paramètres puis cliquer sur le bouton "OK".

La liste des variables détectées dans le code est affichée dans la liste des paramètres. Pour modifier le nom d'un des paramètres, il suffit de le sélectionner et de cliquer sur le bouton « Editer... ».

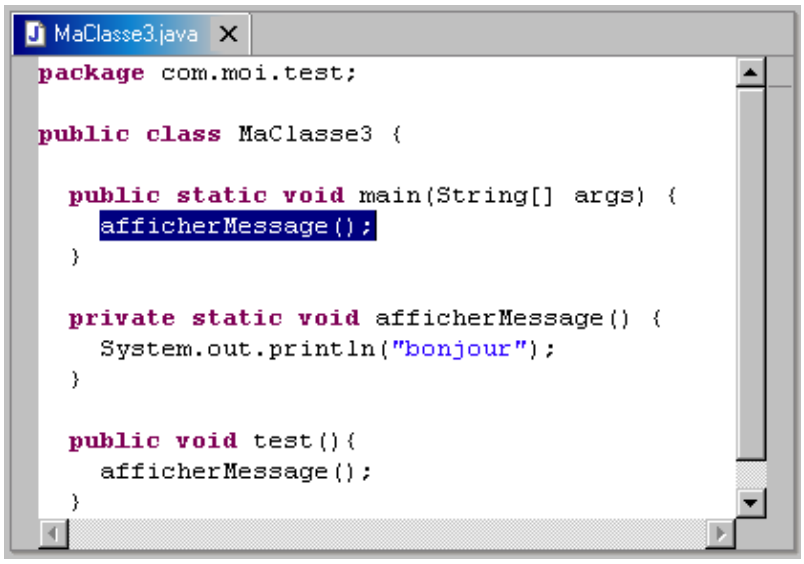


Le bouton « Aperçu » permet de voir et de valider les modifications qui seront apportées.

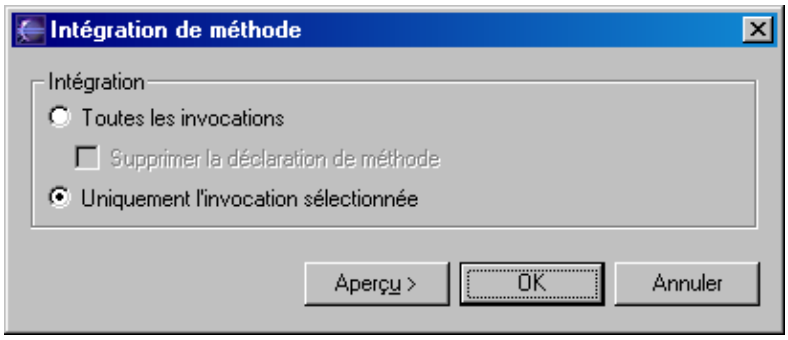


8.2. Intégrer

L'intégration permet de remplacer l'appel d'une méthode par le code contenu dans cette méthode.

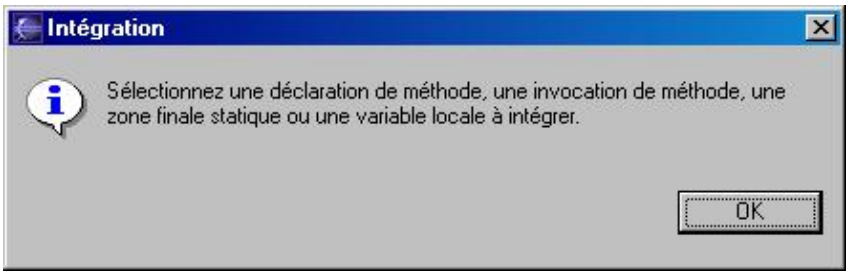


Puis il faut utiliser l'option « Propager les modifications/Intégrer... » du menu principal ou contextuel.



En fonction du contexte d'appel, une boîte de dialogue permet de sélectionner la portée des modifications.

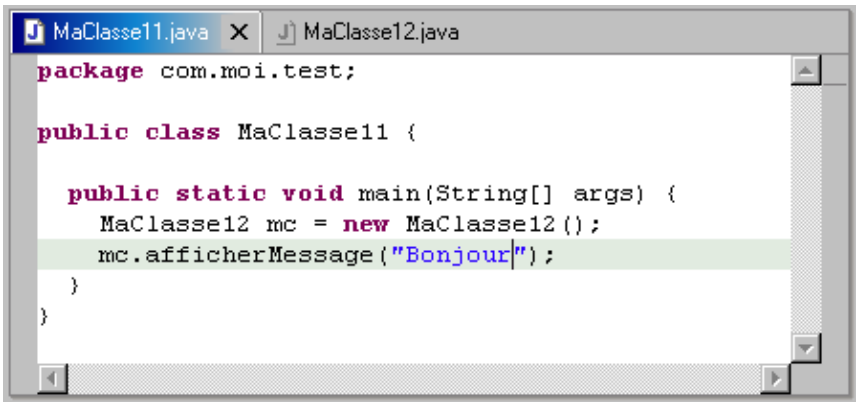
Si le code sélectionné ne correspond pas à une méthode, un message d'erreur est affiché.

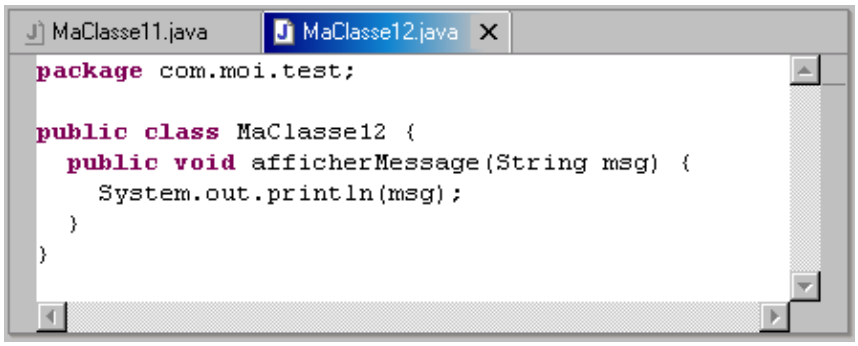


8.3. Renommer

La fonction "Renommer" permet d'attribuer un nouveau nom à une entité présente dans l'espace de travail. La grand intérêt de réaliser une telle opération via cette fonctionnalité plutôt que de le faire à la "main" et qu'elle automatise la recherche et la mise à jour de toute les références à l'élément dans l'espace de travail.

Les exemples de cette section utilisent les deux classes suivantes :

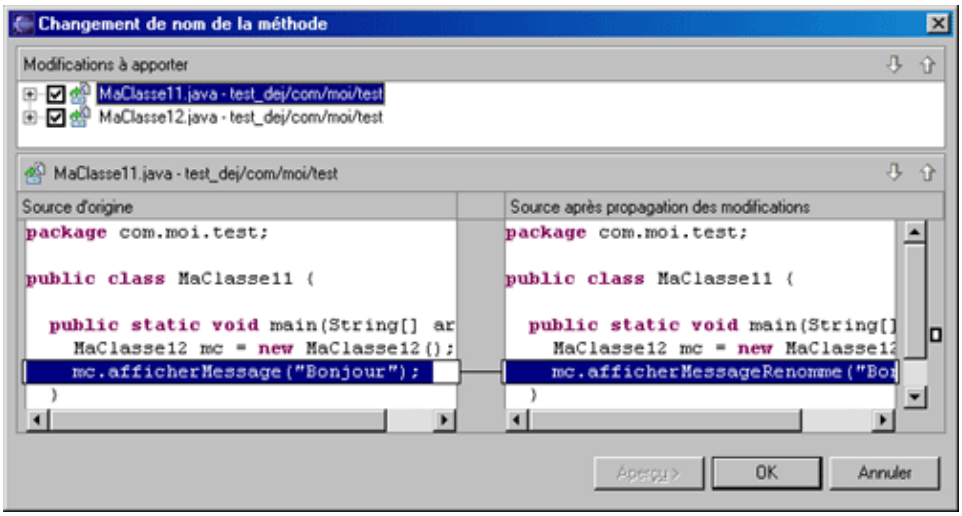




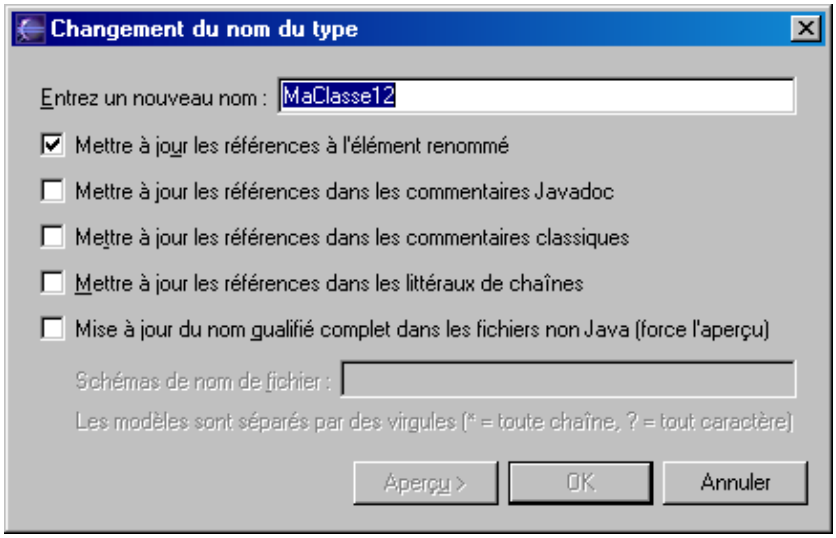
Le renommage concerne l'entité sélectionnée dans la vue "Package" ou "Structure" ou sous le curseur dans l'éditeur de code Java.

Par exemple, pour renommer une méthode, il suffit de positioner le curseur sur le nom de la déclaration d'une méthode dans le code source et sélectionner l'option « Propager les modifications / Renommer » du menu contextuel.

Il suffit alors de saisir le nouveau nom. Eclipse permet de renommer la méthode dans sa classe de définition mais remplace aussi tous les appels de la méthode contenue dans l'espace de travail.

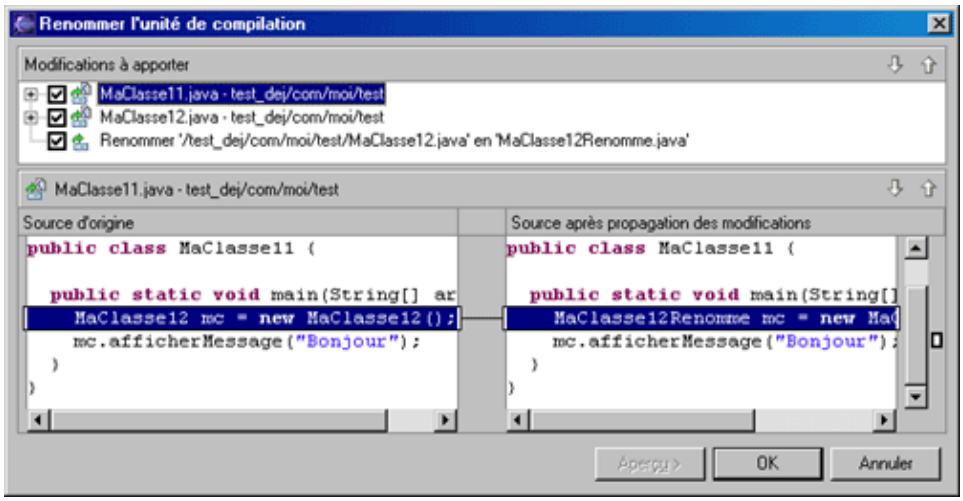


Pour renommer une classe, il y a deux possibilités : selectionner la classe dans la vue « Packages » ou positionner le curseur sur la définition du nom de la classe dans l'éditeur. Puis il faut utiliser l'option « Renommer » du menu contextuel « Propager les modifications ».

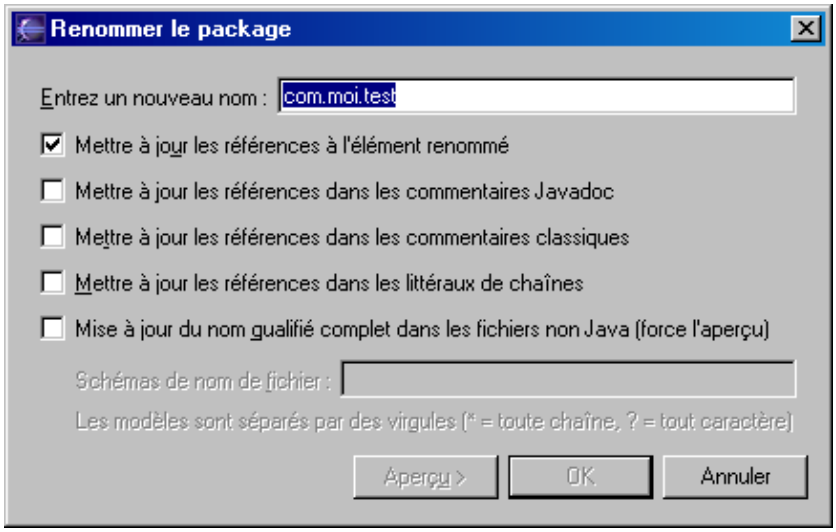


Il est possible de préciser les types d'entités qui doivent être mise à jour.

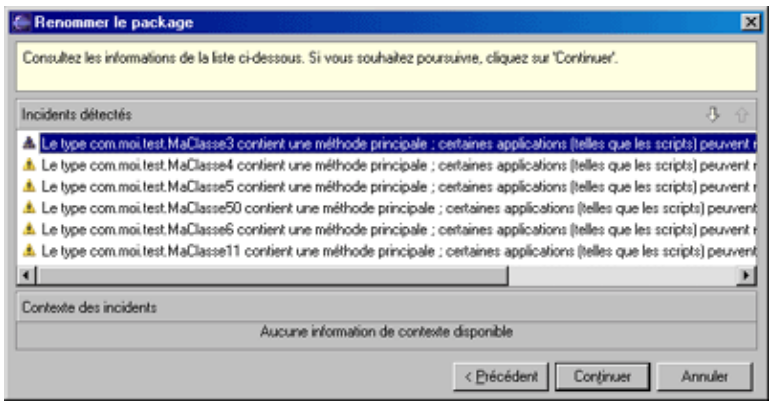
A la validation, Eclipse effectue toutes les modifications nécessaires suite au renommage de la classe, notamment le remplacement à tous les endroits dans l'espace de travail ou la classe est utilisée.



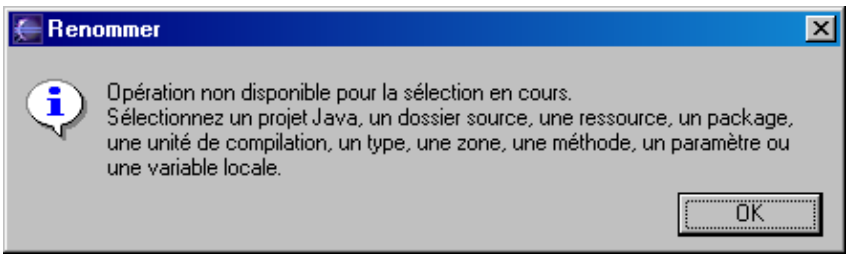
Le principe est le même pour renommer un package.



Si Eclipse détecte des problèmes potenciels lors de l'analyse de la demande, il affiche dans une boite de dialogue la liste de ces problèmes et demande un confirmation avant de poursuivre les traitements.



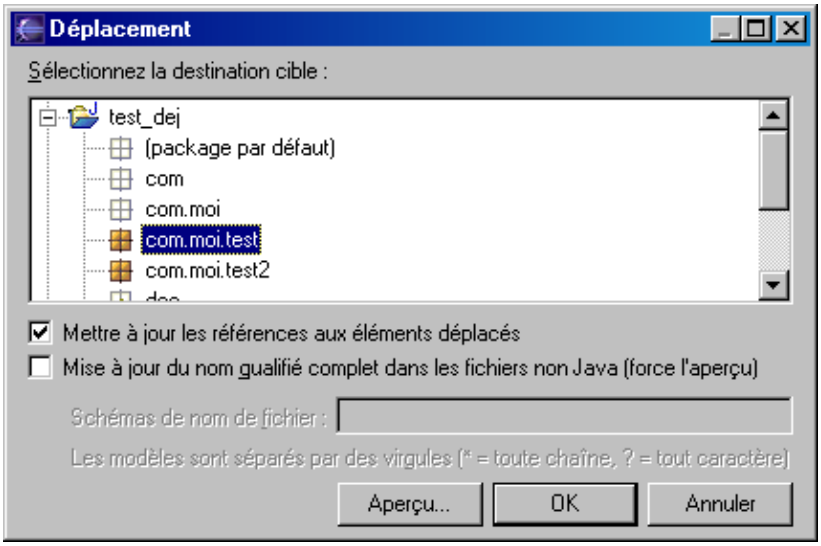
Si la sélection ne correspond à aucun élément renommable, un message d'erreur est affiché.



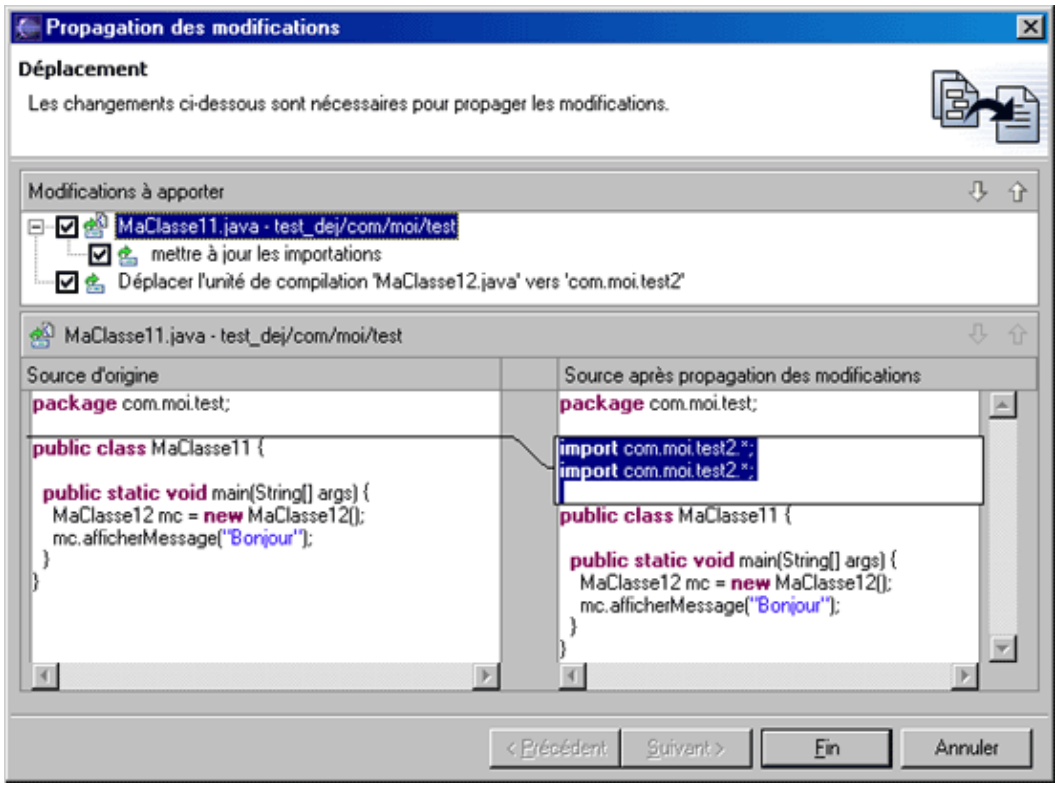
8.4. Déplacer

La fonction "Déplacer" permet de déplacer certains éléments précis notamment une classe dans l'espace de travail pour la déplacer dans un autre package. La grand intérêt de réaliser une telle opération via cette fonctionnalité est d'automatiser la recherche et la mise à jour de toute les références à l'élement dans l'espace de travail.

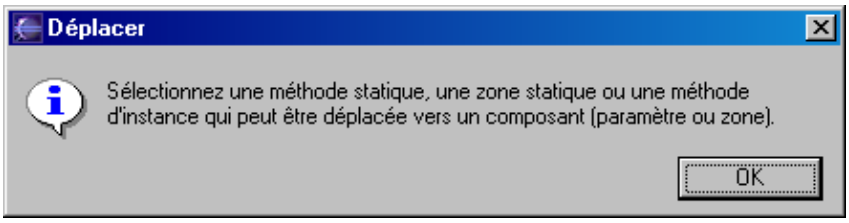
Par exemple, pour déplacer une classe dans un autre package, il faut sélectionner la classe dans la vue "Packages" et de sélectionner l'option "déplacer" du menu "Propager les modifications"



En plus du déplacement de la classe, les clauses import correspondantes sont modifiées dans les classes qui utilisent la classe déplacée.

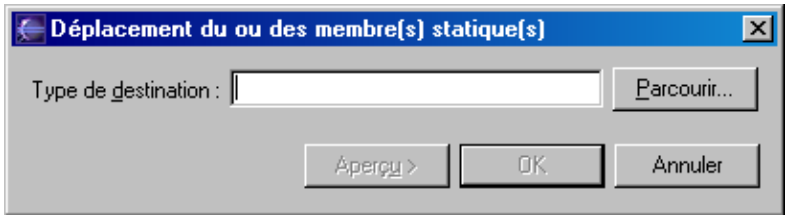


Si le code sélectionné dans l'éditeur de code Java ne correspond pas à une entité concernée par cet action, un message d'erreur est affiché.



Dans l'éditeur de code il faut sélectionner ou mettre le curseur sur un élément statique, avant d'utiliser la fonctionnalité.

Une boite de dialogue permet de sélectionner la classe dans laquelle la méthode va être transférée.

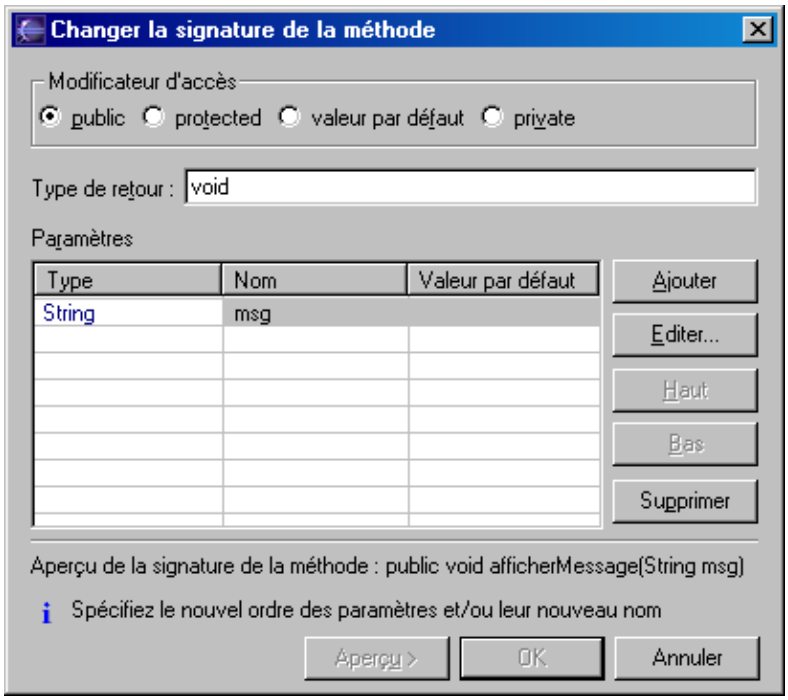


Le bouton "Parcourir" permet d'ouvrir une boîte de dialogue pour sélectionner la classe de destination.

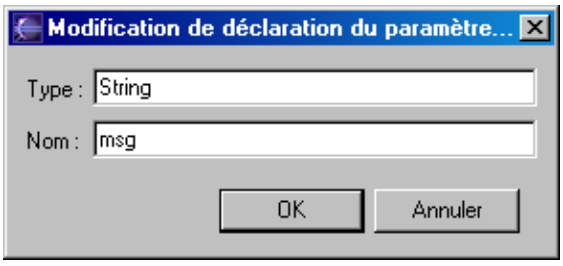
8.5. Changer la signature de la méthode

Cette fonction permet de modifier la signature d'une méthode : changer les attributs de visibilités, les paramètres (ajouter/supprimer un paramètre ou modifier le nom, le type ou l'ordre des paramètres) ou le type de retour de la méthode.

Pour utiliser cette fonction, il faut sélectionner dans l'éditeur le nom d'une méthode et appeler la fonction par le menu contextuel. Une boîte de dialogue permet de modifier les éléments qui constituent la signature de la méthode.



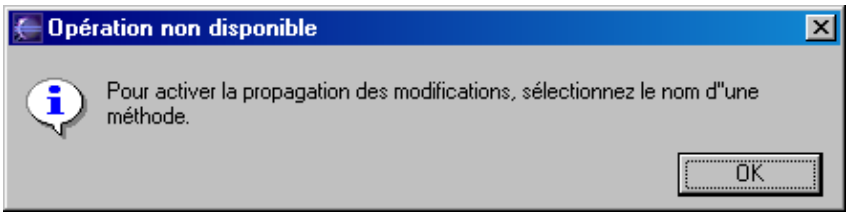
Les actions réalisées par les boutons "Ajouter" et "Supprimer" sont assez explicite. Le bouton "Editer" permet de modifier le paramètre sélectionné.



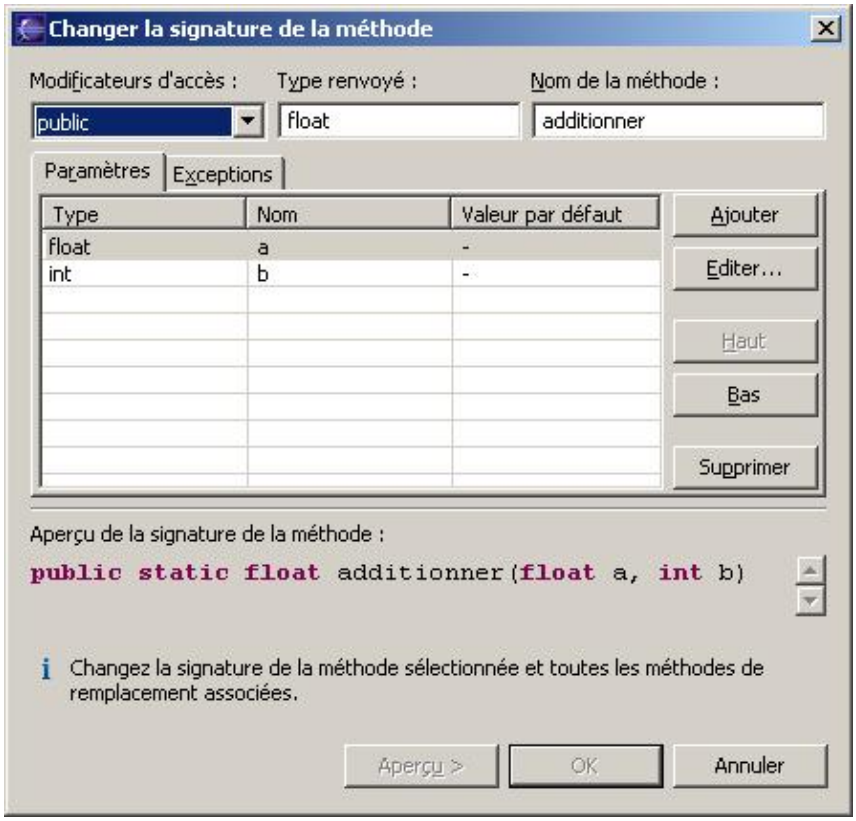
Les boutons "Haut" et "Bas" permettent de modifier l'ordre du paramètre sélectionné.

La valeur par défaut est importante car elle sera utilisée lors de la modification de l'utilisation de la méthode, pour éviter les erreurs de compilation.

Si le code sélectionné dans l'éditeur de code lors de l'appel de la fonction ne correspond pas à une entité concernée par cet action, un message d'erreur est affiché.

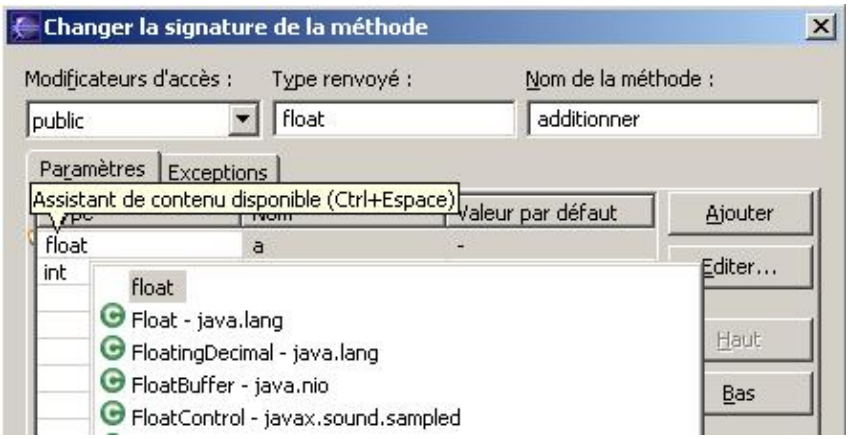


Cette fonction a été améliorée avec plusieurs fonctionnalités.



La boîte de dialogue permet maintenant aussi de renommer le nom de la méthode.

Le type peut être recherché grâce à un assistant en utilisant la combinaison de touche Ctrl+Espace.



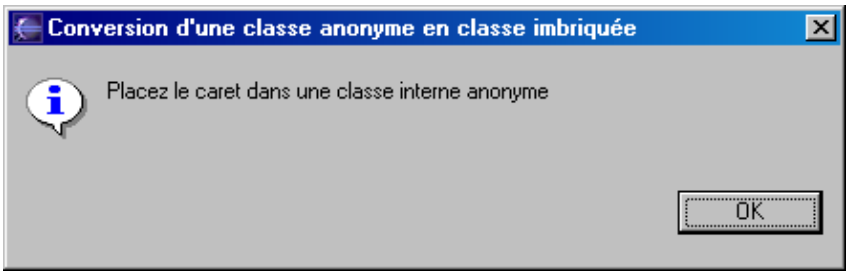
L'onglet "Exceptions" permet de modifier les exceptions propager par la méthode

Lors des mises à jours liées à cette fonctionnalité, la documentation Javadoc de la méthode sera mise à jour.

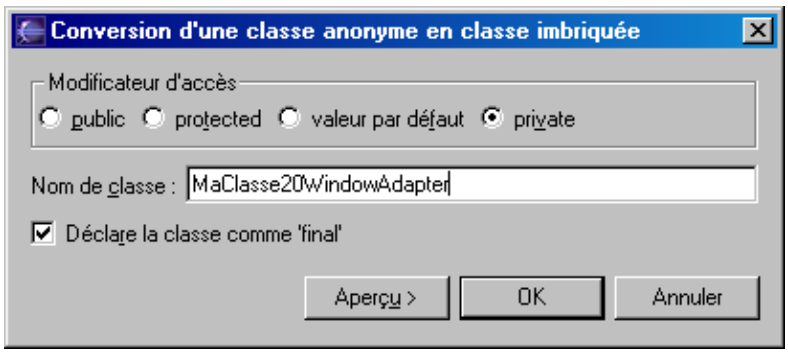
8.6. Convertir une classe anonyme en classe imbriquée

Cette fonction permet de transformer une classe anonyme souvent utilisée pour créer des listeners en une classe imbriquée.

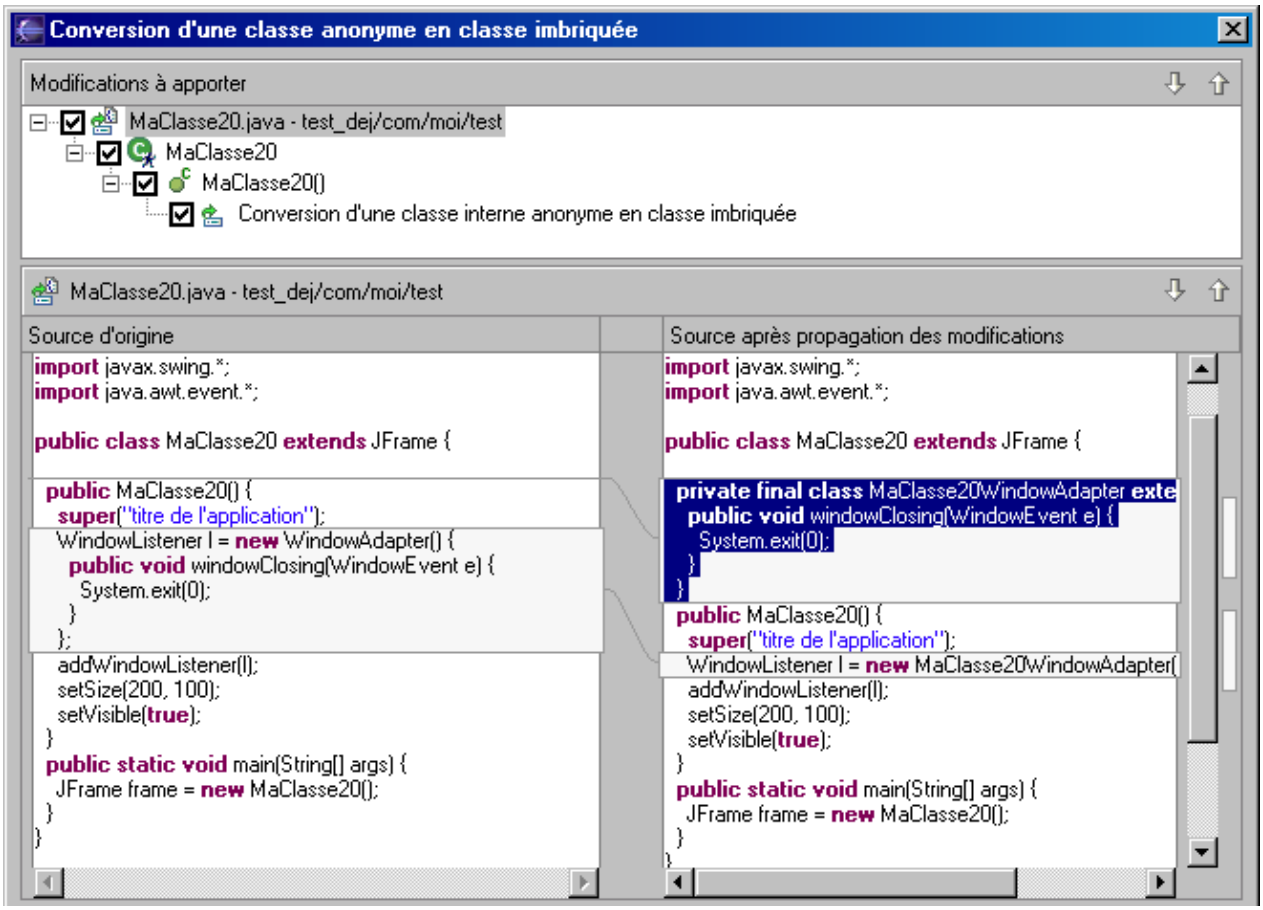
Pour utiliser cette fonction, il faut placer le curseur dans le code de la classe anonyme. Si tel n'est pas le cas lors de l'appel de la fonction, une message d'erreur est affiché.



Une boîte de dialogue permet de saisir les informations concernant la classe qui sera générée.



La nouvelle classe imbriquée est crée en tenant compte des informations saisies.

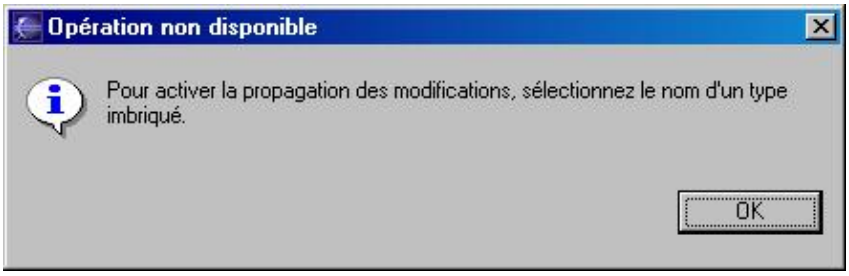


8.7. Convertir un type imbriqué au niveau supérieur

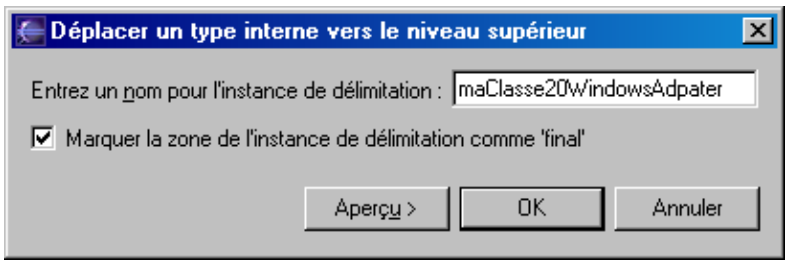
Cette fonction permet de convertir une classe imbriquée en une classe définie dans un fichier particulier.

Pour utiliser cette fonction, il faut sélectionner une classe imbriquée dans la vue "Packages" ou "Structure" ou sélectionner dans l'éditeur de code le nom d'une classe imbriquée.

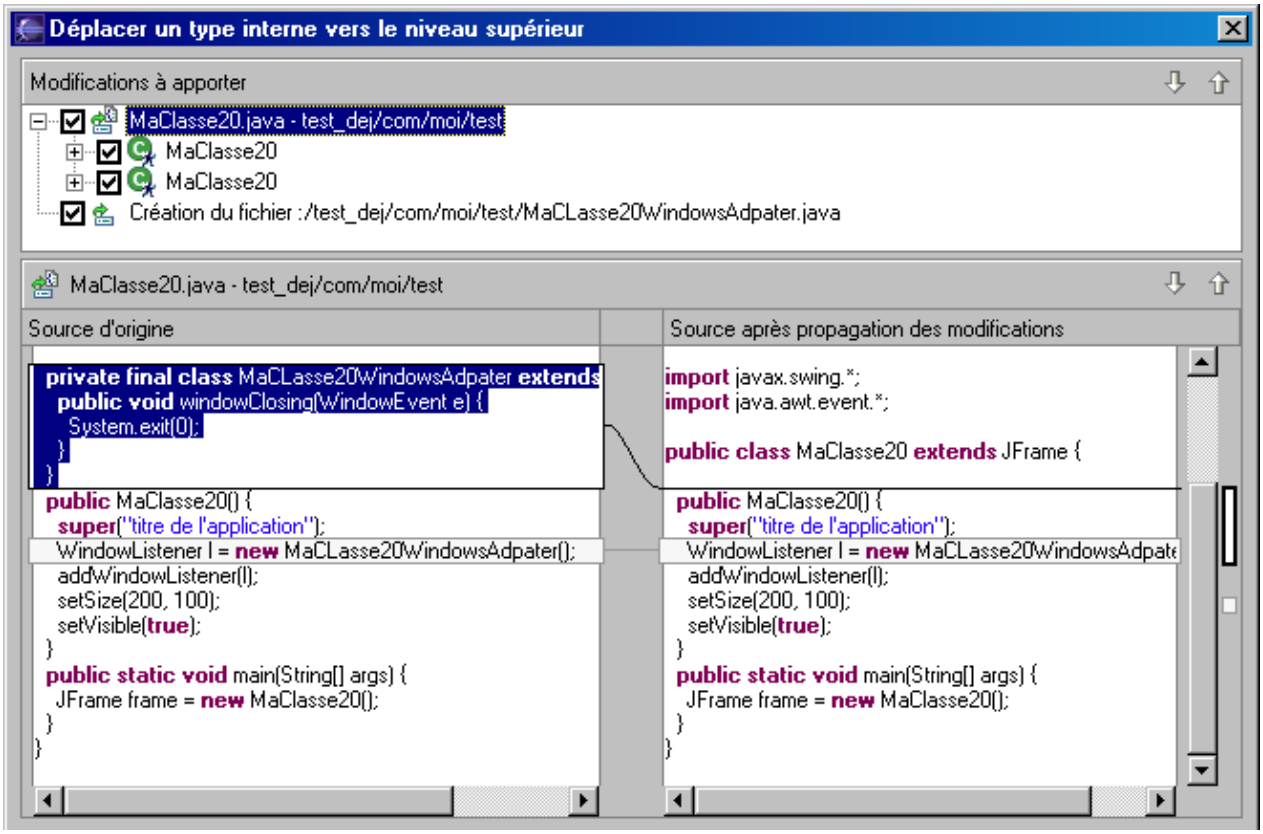
Si le code sélectionné ne correspond pas à une entité concernée par cet action, un message d'erreur est affiché.



Lors de la l'appel de la fonction, une boite de dialogue permet de préciser des informations sur la classe qui va être déplacée.



Voici un exemple de l'aperçu de l'utilisation de cet fonction

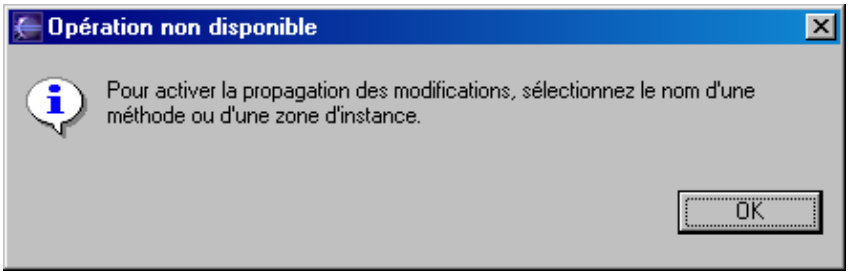


Le nom pour l'instance de délimitation permet de stocker une instance de la classe englobante si la classe avait besoin de l'accès à des membres de l'ancienne classe englobante.

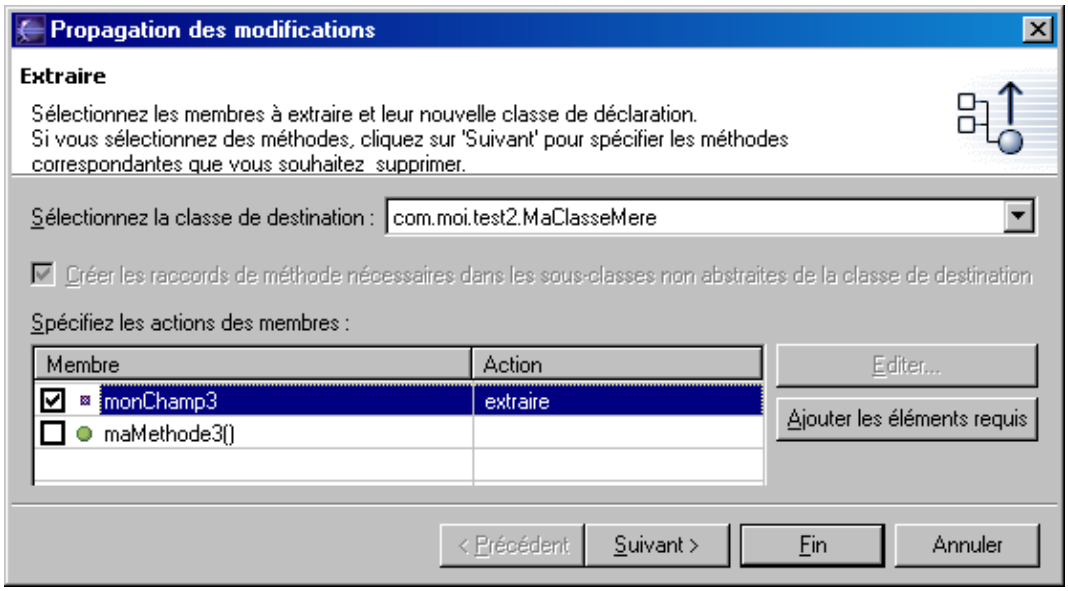
8.8. Extraire

Cette fonction permet de déplacer la déclaration d'un membre d'une classe fille dans une de ces classes meres.

Pour utiliser cette fonction, il faut sélectionner un membre d'une classe fille ou positionner le curseur sur un de ces membres sinon un message d'erreur est affiché.

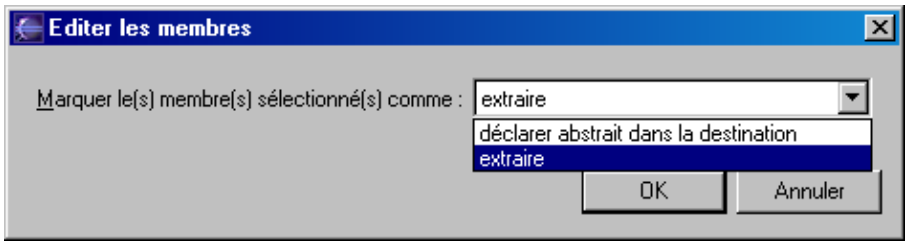


Une boîte de dialogue permet de sélectionner le ou les membres concernés dont celui à partir duquel la fonction a été appelée.

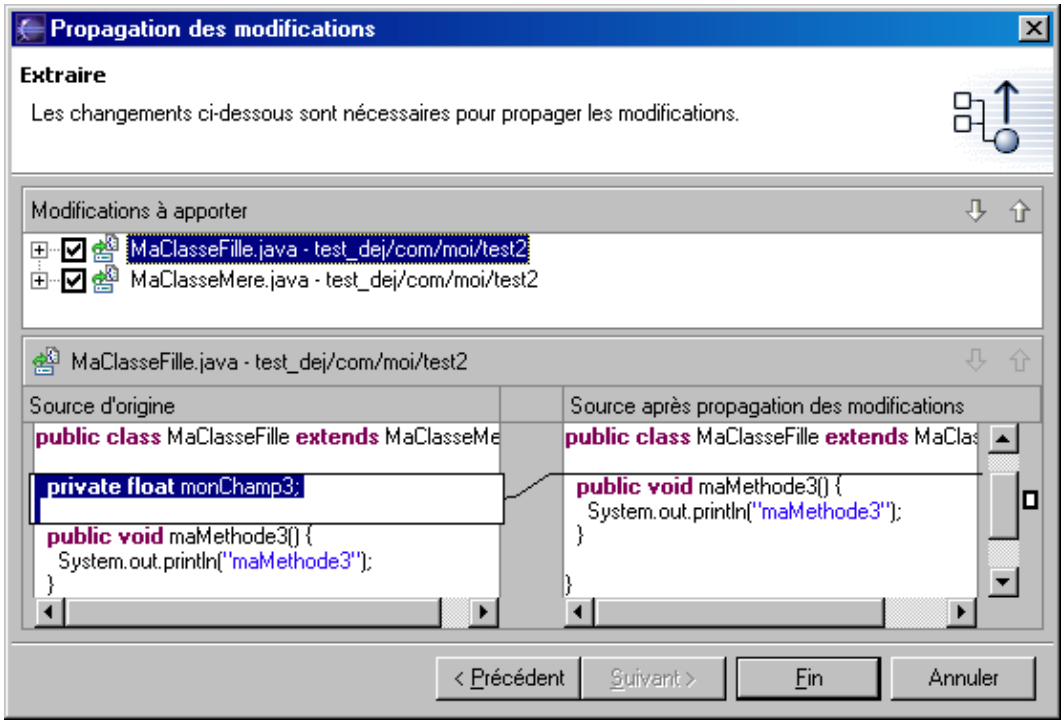


Il suffit de cocher le ou les membres concernées et de sélectionner la classe mère dans la liste déroulante.

Pour les méthodes, le bouton "Editer" permet de sélectionner l'action qui sera réalisée.



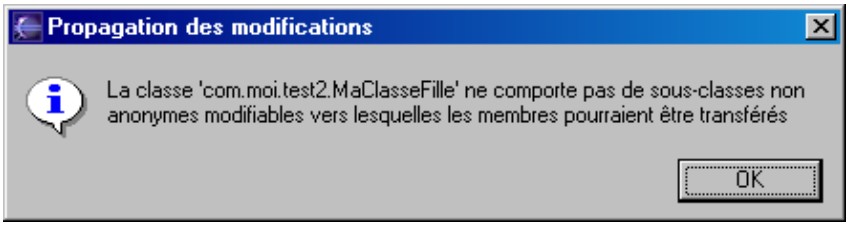
L'action "extraire" permet de déplacer la déclaration et le code de la méthode dans la classe mère sélectionnée. L'action "déclarer abstrait dans la déclaration" permet de déclarer la classe choisie abstraite avec une déclaration de la méthode elle aussi abstraite. Dans ce dernier cas, le code de la méthode est laissé dans la classe fille en tant que rédéfinition.



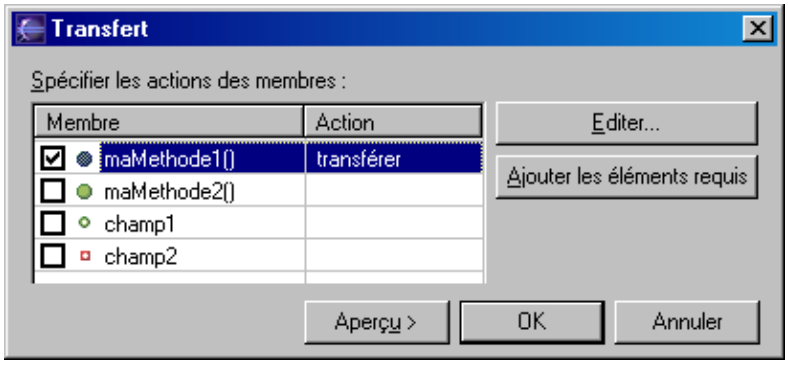
8.9. Transferer

Cette fonction permet de transferer une méthode dans les classes filles.

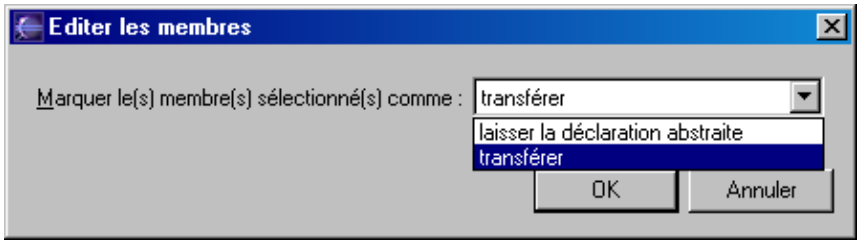
Pour utiliser cette fonction il faut sélectionner un membre d'une classe possèdant au moins une classe fille ou mettre le curseur sur un de ces membres dans l'éditeur de code. Si la fonction est appelée sur une classe qui ne possède pas de classe fille, un message d'erreur est affiché.



Une boîte de dialogue permet de sélectionner le ou les membres concernés par le transfert.



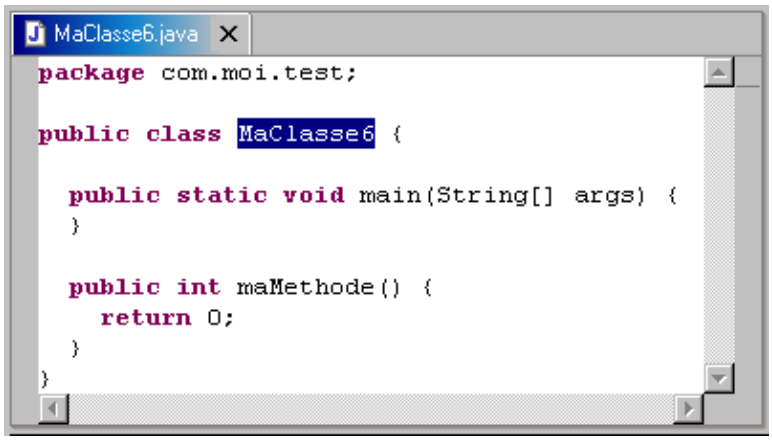
Pour une méthode, le bouton "Editer" permet de sélectionner l'action qui sera réaliser.



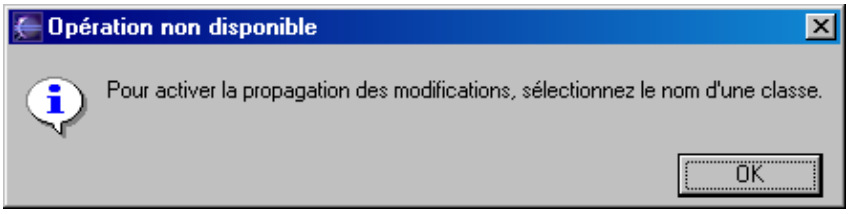
L'action "transfert" permet de déplacer la déclaration et le code de la méthode dans la ou les classes filles. L'action "laisser la déclaration abstraite" permet de laisser la déclaration abstraite de la méthode dans la classe mère.

8.10. Extraire une interface

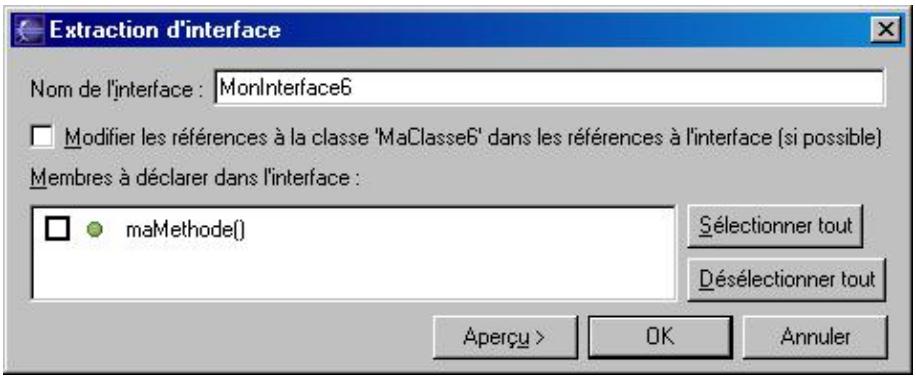
Cette fonction permet de définir à postériori une interface à partir d'une ou plusieurs méthodes définies dans une classe. Il faut sélectionner dans le code le nom d'une classe.



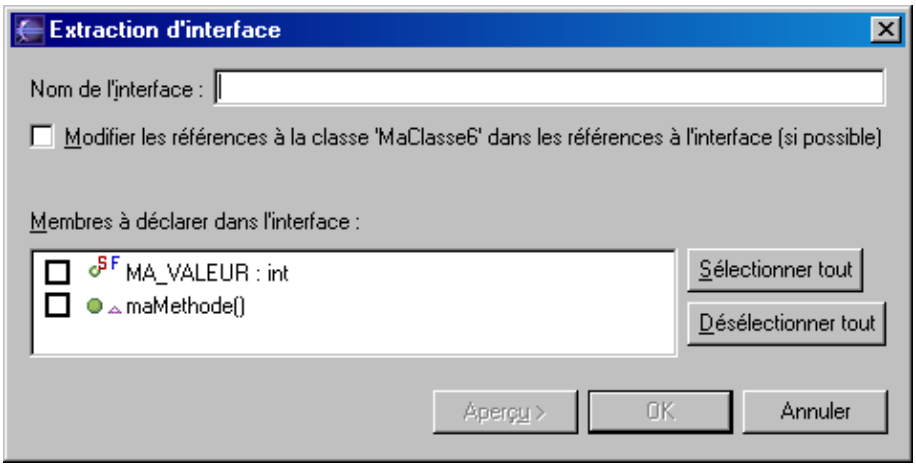
Si la partie sélectionnée ne correspond pas à un nom de classe, un message d'erreur est affiché.



Il suffit ensuite d'utiliser l'option « Extraire une interface » du menu « Propager les modifications ». Une boite de dialogue permet de préciser le nom de l'interface et de sélectionner les membres à déclarer dans l'interface.



Pour respecter les spécifications du langage Java, les membres qu'il est possible d'intégrer dans l'interface sont des méthodes et des constantes publiques, par exemple :



Le résultat est la création de l'interface et son implémentation par la classe sélectionnée.

MaClasse6.java | MonInterface6.java

```
package com.moi.test;

public interface MonInterface6 {
  public abstract int maMethode();
}
```

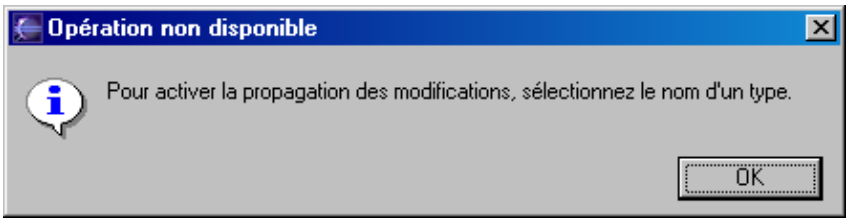
MaClasse6.java | MonInterface6.java

```
package com.moi.test;

public class MaClasse6 implements MonInterface6 {

  public static void main(String[] args) {
  }

  public int maMethode() {
    return 0;
  }
}
```

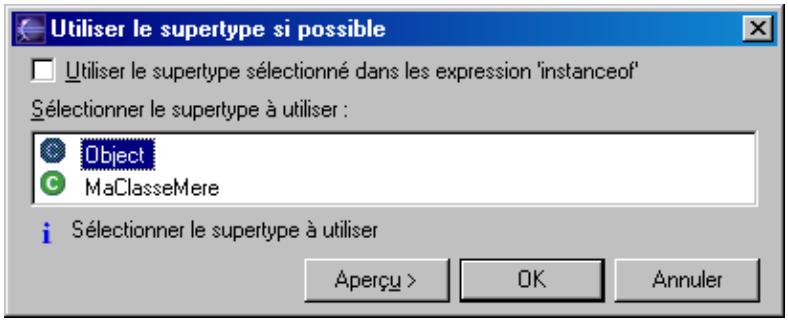
8.11. Utiliser le supertype si possible

Cette fonction permet de remplacer l'utilisation d'un type par un de ces super types si celui ci en possède.

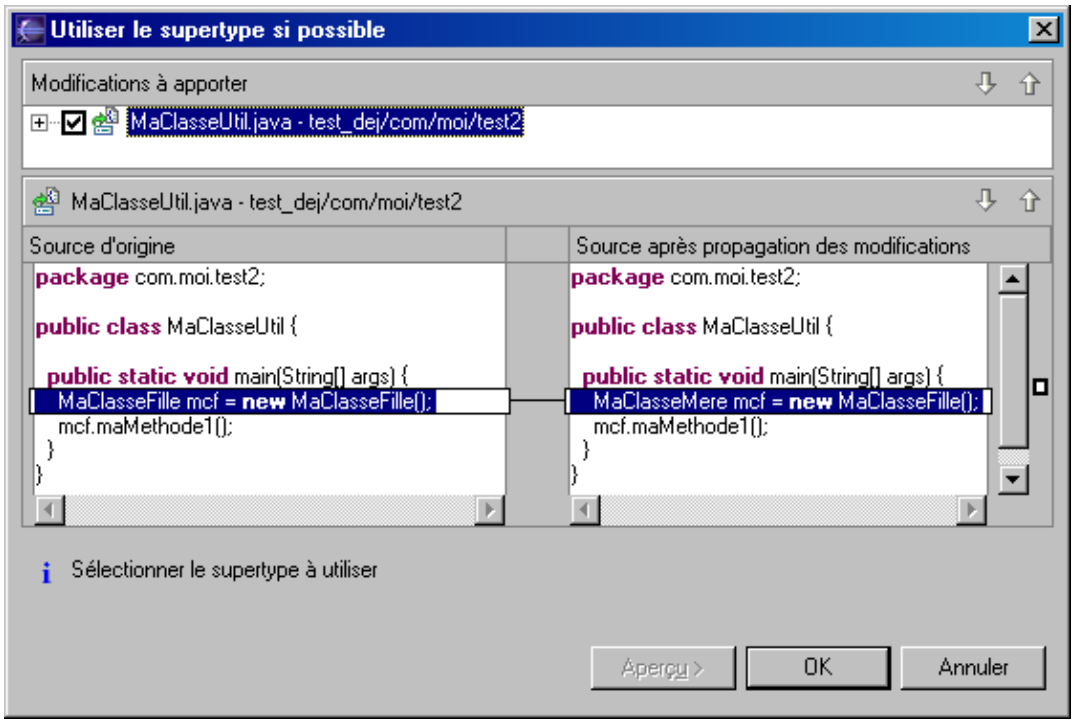
Pour utiliser cette fonction, il faut sélectionner un type dans le code de l'éditeur ou mettre le curseur sur ce dernier sinon un message d'erreur est affiché :



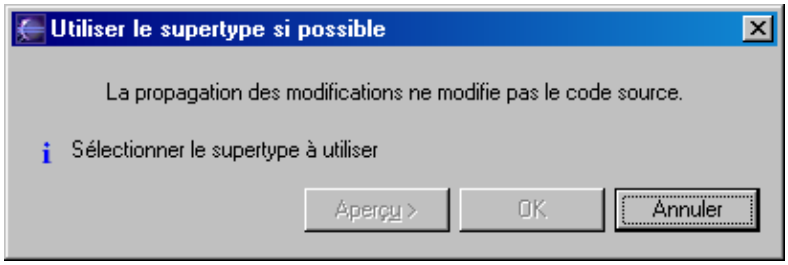
Lors de l'appel de cette fonction, une boîte de dialogue permet de sélectionner une des classes mère du type sélectionné.



La pré−visualisation permet de voir les modifications qui seront apportées dans le code.



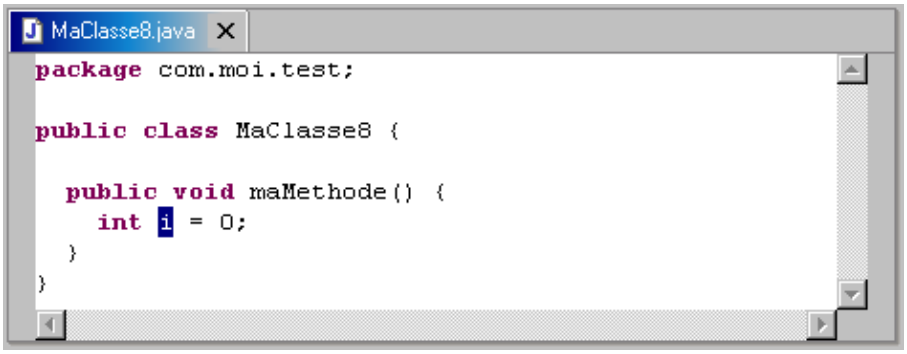
Si l'utilisation du super type sélectionné n'est pas possible (par exemple car une méthode utilisée n'est pas définie dans le type sélectionné), aucune modification n'est faite dans le code et l'aperçu affiche le message suivant :



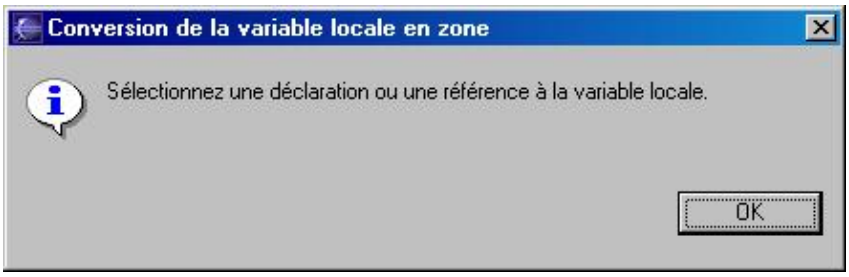
8.12. Convertir la variable locale en zone

La fonction « Convertir la variable locale en zone » permet de transformer une variable locale à une méthode en un champ de la classe.

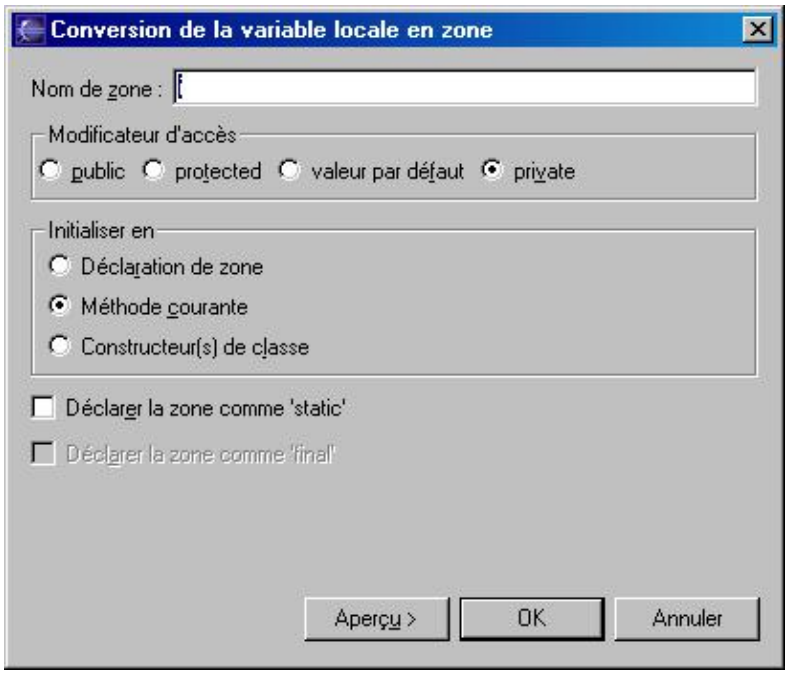
Pour utiliser cette fonction, il faut sélectionner dans le code une variable locale, puis utiliser l'option « Convertir la variable locale en zone » du menu « Propager les modifications ».



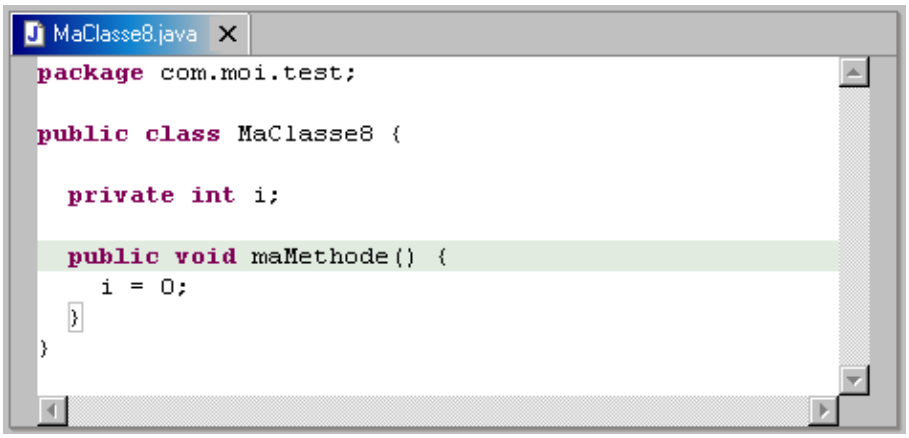
Si la sélection ne correspond pas à une variable locale, un message d'erreur est affiché.



Une boîte de dialogue permet de préciser le nom du champ, le modificateur d'accès et l'endroit dans le code où le champ sera initialisé.



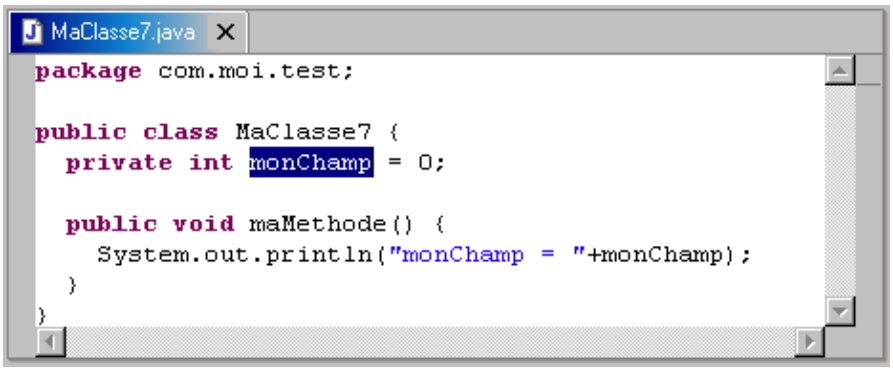
Le résultat de l'exécution de la fonction est le suivant :



8.13. Encapsuler la zone

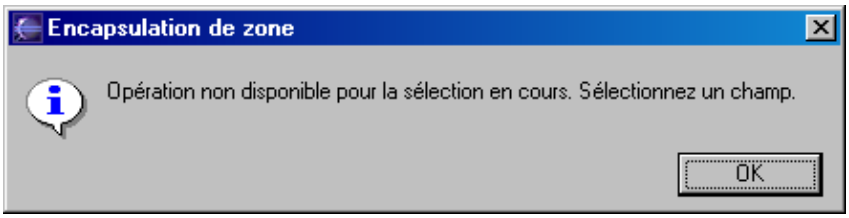
Cette fonction permet d'encapsuler un attribut en générant un getter et éventuellement un setter.

Il faut sélectionner dans le code un champ de la classe.

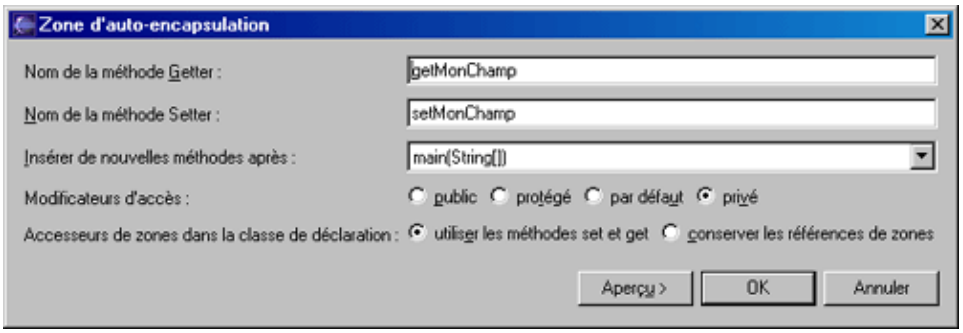


Il faut ensuite utiliser l'option « Encapsuler la zone » du menu « Propager les modifications ».

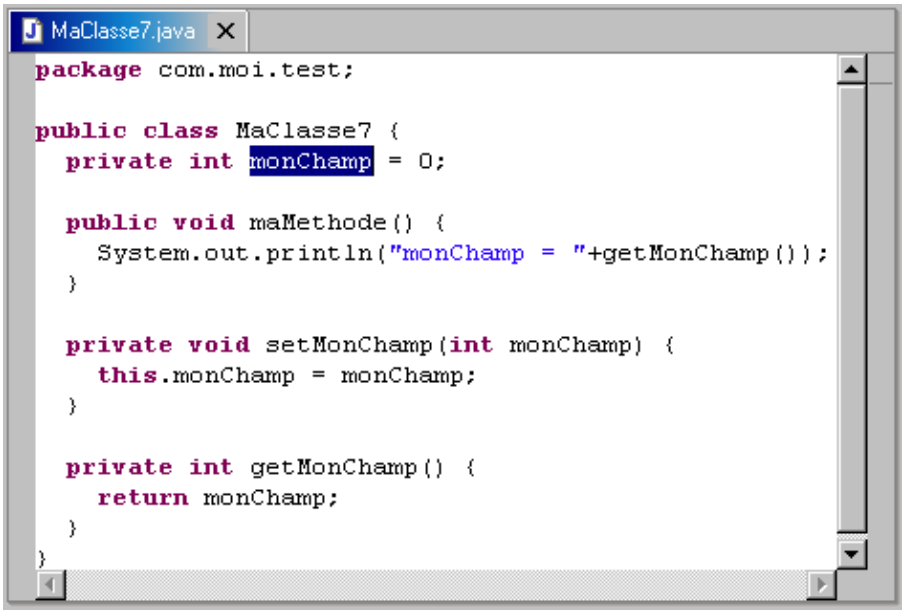
Si la sélection dans le code ne correspond pas à un champ de la classe, un message d'erreur est affiché.



Une boite de dialogue permet de préciser les options pour la génération du getter et du setter



Le résultat de l'exécution génère le getter et le setter et remplace l'utilisation du champ par ces méthodes.

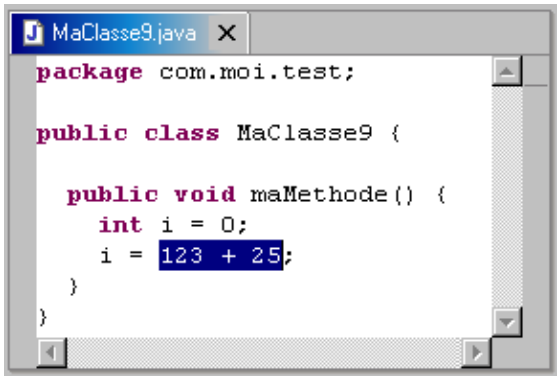


En sélectionnant l'option « conserver les références de zones », ces méthodes ne sont pas utilisées dans la classe où le champ est utilisé.

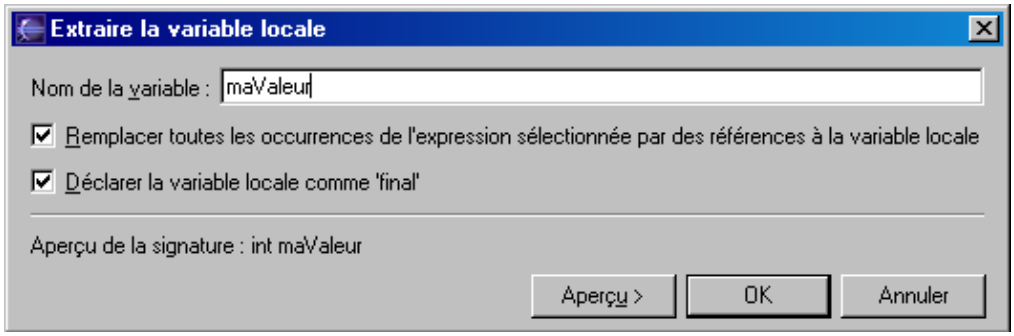
8.14. Extraire la variable locale

Cette fonction permet de définir une constante à partir d'une expression.

Il faut sélectionner une expression dans le code source.

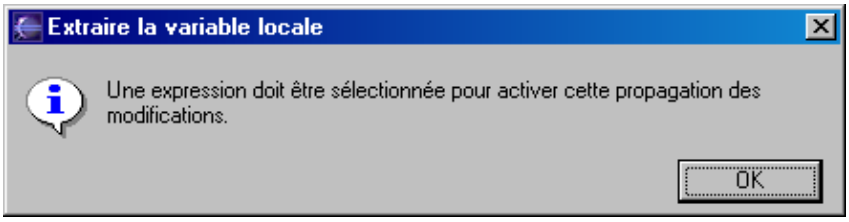


Puis utiliser l'option « Extraire la variable locale » du menu « Propager les modifications ».

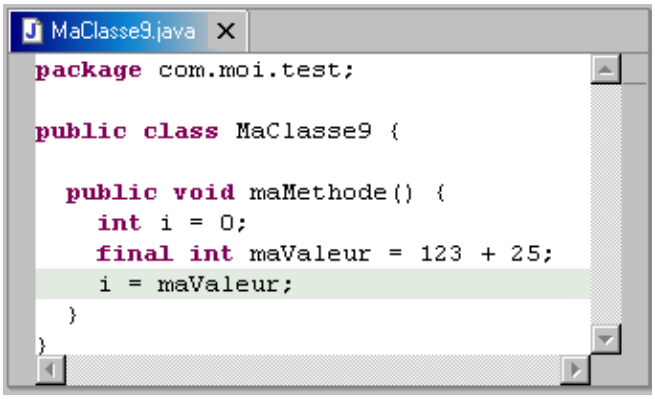


Une boîte de dialogue permet de saisir le nom de la variable, de préciser si toutes les occurrences doivent être modifiées et si la variable doit être une constante.

Si la sélection dans le code source est incorecte, un message d'erreur est affiché.

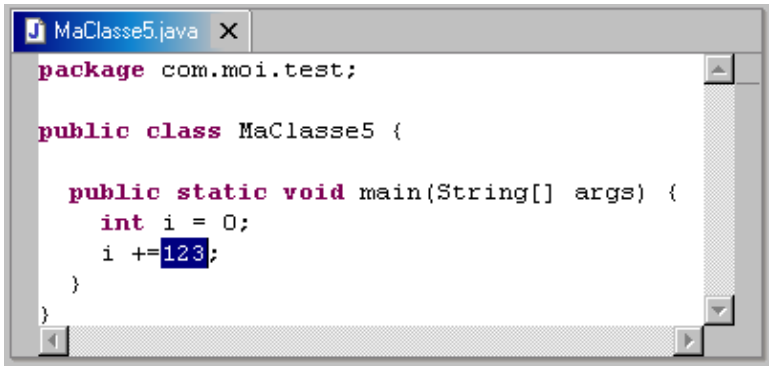


Le résultat de l'exécution de cette fonction est le suivant :

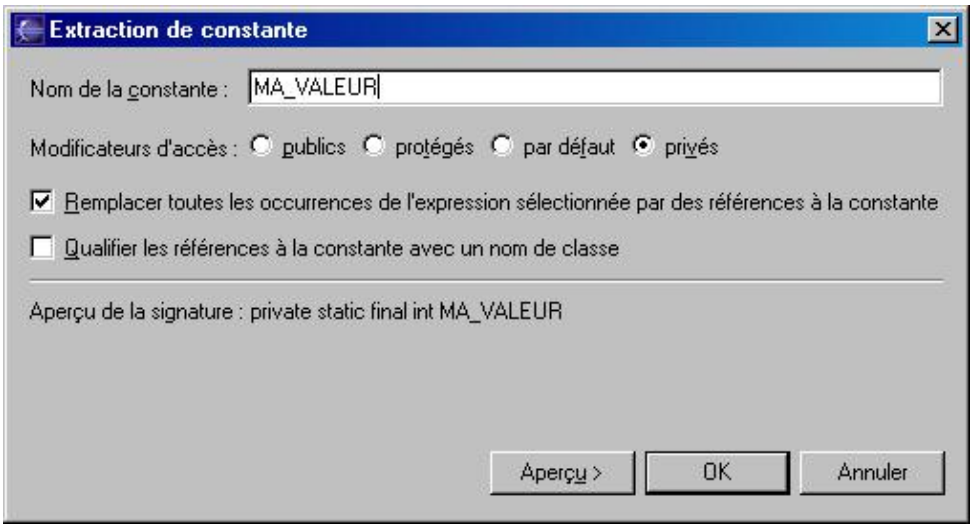


8.15. Extraire une constante

Cette fonction permet de définir une constante à partir d'une valeur en dur utilisée dans le code. Il suffit de sélectionner cette valeur dans le code source.

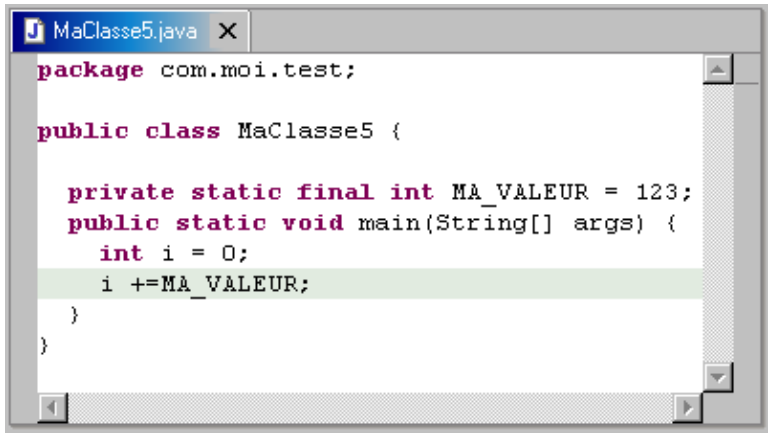


Puis d'utiliser l'option « Extraire une constante ... » du menu « Propager les modifications ». Une boite de dialogue apparaît pour préciser le nom de la constante à créer et son modificateur d'accès.



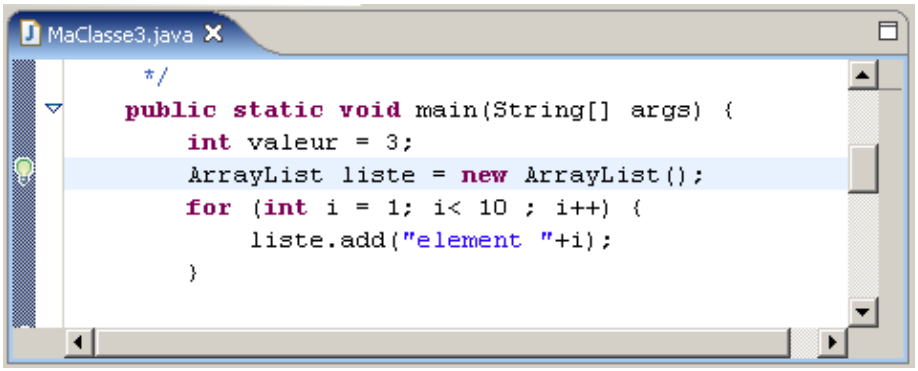
Il est possible de préciser si toutes les occurrences de la valeur doivent être remplacées dans le code source.

Il est aussi possible de préciser l'utilisation de la qualification de la constante avec le nom de la classe dans laquelle elle est définie.

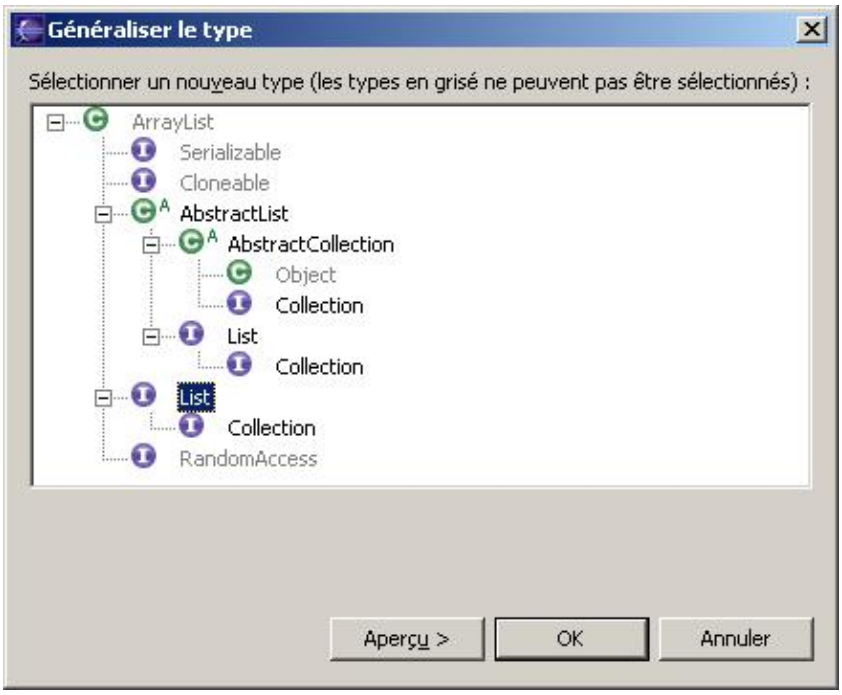


8.16. Généraliser le type

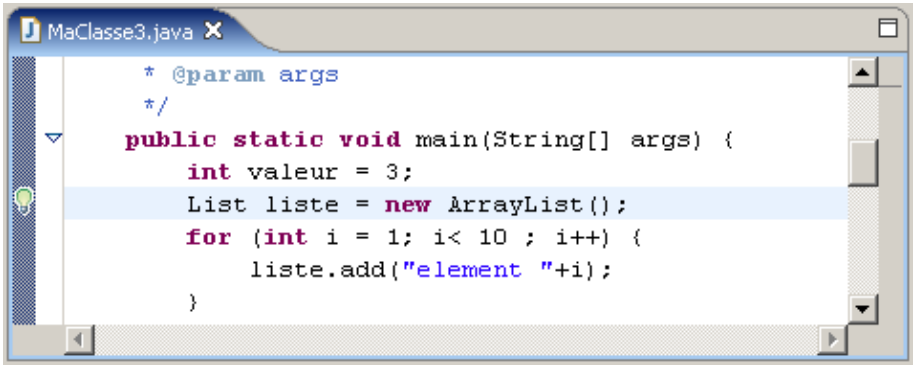
Cette fonction permet de remplacer le type d'une variable par un de ces super types. Il faut positionner le curseur sur la déclaration d'une variable, d'un champ, d'un paramètre ou d'une valeur de retour.



Une boite de dialogue affiche la liste des super types.



Il suffit de sélectionner le type désiré et de cliquer sur le bouton « Ok » pour que le code soit mise à jour.

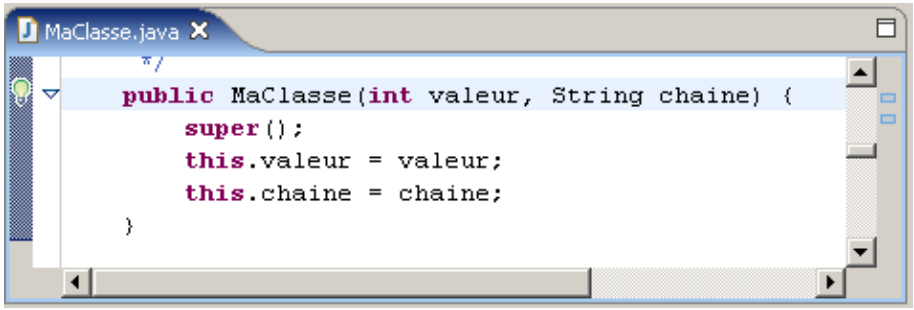


Cette fonctionnalité est très utile car elle facilite la mise en oeuvre d'une bonne pratique de programmation consistant à utiliser un super type comme type d'une variable lorsque cela est

possible.

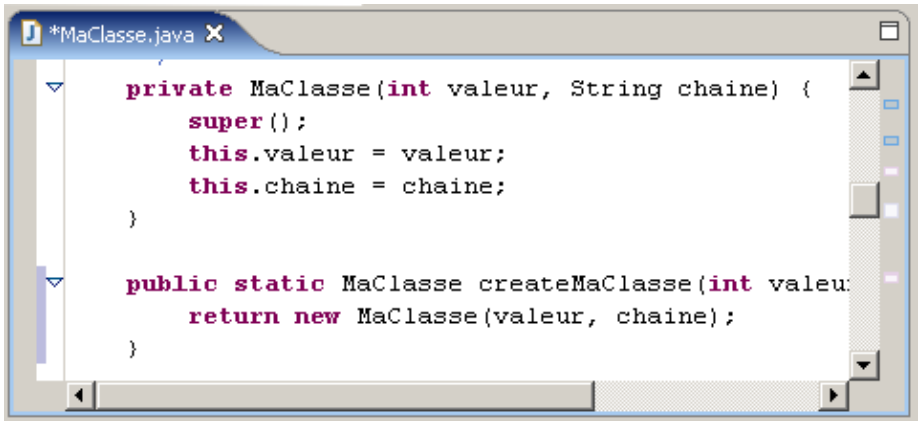
8.17. Introduire une fabrique

Cette fonction permet d'obliger l'utilisation d'une fabrique pour créer une occurrence de la classe.



Il faut positionner le curseur sur un constructeur puis appeler la fonction.

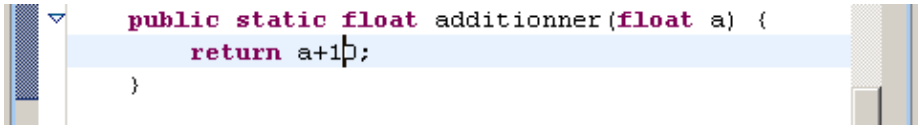
Le code est modifié en créant la méthode de fabrication et en rendant le constructeur privé.



8.18. Introduire un paramètre

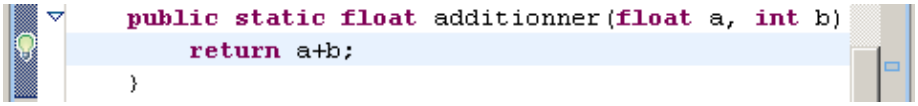
Cette fonction permet de remplacer une expression utilisée dans une méthode par un paramètre fourni à cette méthode.

Il faut positionner le curseur sur une expression littérale et appeler la fonction.



Une boîte de dialogue permet de saisie le nom du paramètres

En cliquant sur le bouton « Ok », le paramètre est ajouté dans la signature de la méthode et la valeur littérale est remplacée par le paramètre.



8.19. Annuler ou refaire une opération de refactoring

Le menu "Propager les modifications" possède deux options "Annuler" et "Rétablir" qui ont un rôle similaire aux deux options du même nom dans le menu "Edition". L'utilisation de ces deux options est cependant dédiées à l'annulation ou le rétablissement des modifications opérées par la dernière opération de refactoring.

Ces deux options ne sont plus disponibles dès que des modifications supplémentaires sont effectués et sauvegardées dans au moins un des fichiers modifiés par l'opération de refactoring.

Si un des fichiers modifiés par l'opération de refactoring est modifié sans être sauvegardé avant l'utilisation d'une de ces deux options, un message d'erreur est affiché.

9. Ant et Eclipse

Ant est un projet du groupe Apache−Jakarta. Son but est de fournir un outil écrit en Java pour permettre la construction d'applications (compilation, exécution de taches post et pré compilation ...). Ces processus de construction sont très importants car ils permettent d'automatiser des opérations répétitives tout au long du cycle de vie de l'application (développement, tests, recette, mise en production ...).

Ant pourrait être comparé au célèbre outil make sous Unix. Il a été développé pour fournir un outil de construction indépendant de toute plate−forme car écrit avec et pour Java. Il repose sur un fichier de configuration XML qui décrit les différentes tâches qui devront être exécutées par l'outil. Ant fournit un certain nombre de tâches courantes qui sont codées sous forme d'objets développés en Java.

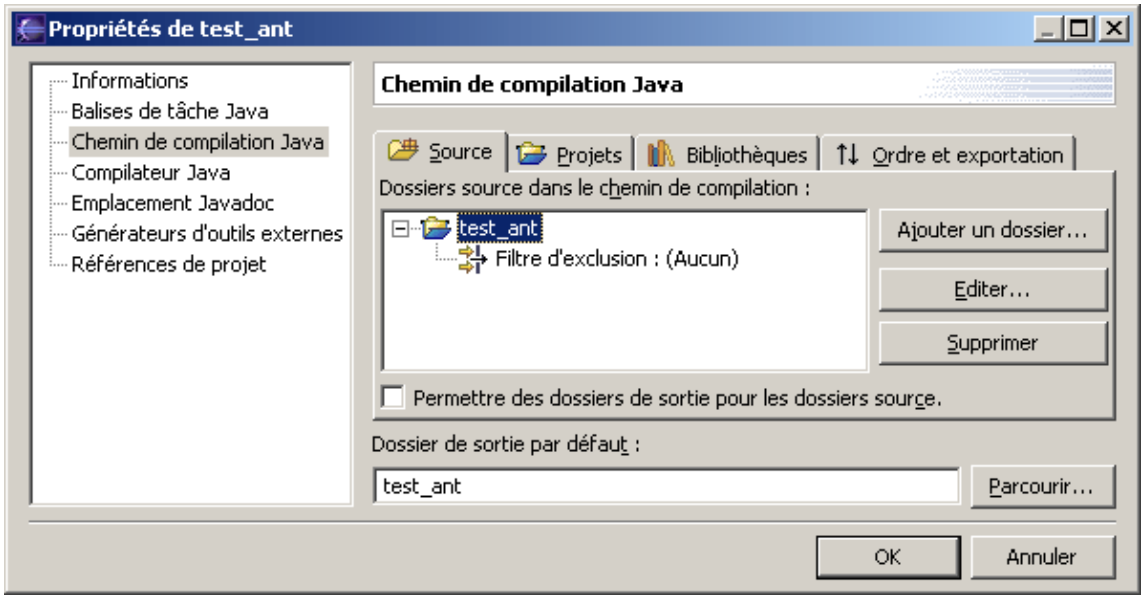
Le fichier de configuration contient un ensemble de cibles (target). Chaque cible contient une ou plusieurs tâches. Chaque cible peut avoir une dépendance envers une ou plusieurs autres cibles pour pouvoir être exécutée.

Pour obtenir plus de détails sur l'utilisation de Ant, il est possible de consulter la documentation de la version courante à l'url suivante : http://jakarta.apache.org/ant/manual/index.html

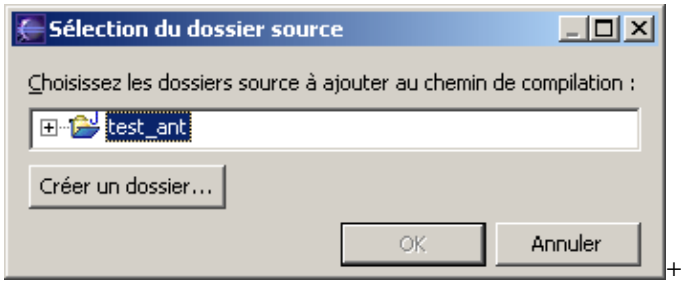
9.1. Structure du projet

Pour utiliser Ant, il faut organiser différemment la structure du projet si celui−ci utilise la structure par défaut d'un projet Java. Par défaut, les fichiers sources .java et leurs homologues compilés .class sont dans le même répertoire à la racine du projet.

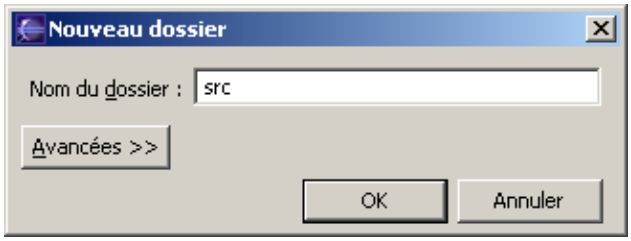
Il faut mettre les sources dans un répertoire et les fichiers .class dans un autre. Ce changement peut être fait dans l'onglet "Source" des propriétés du projet.



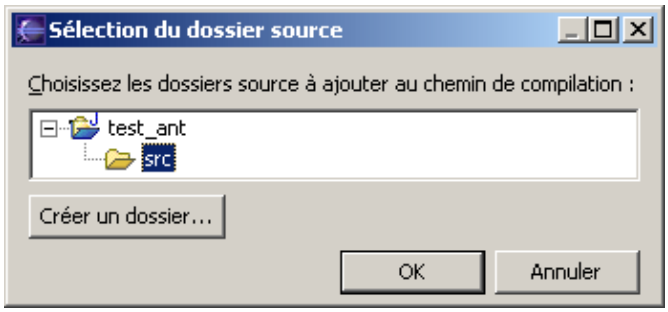
Cliquez sur le bouton "Ajouter un dossier ".



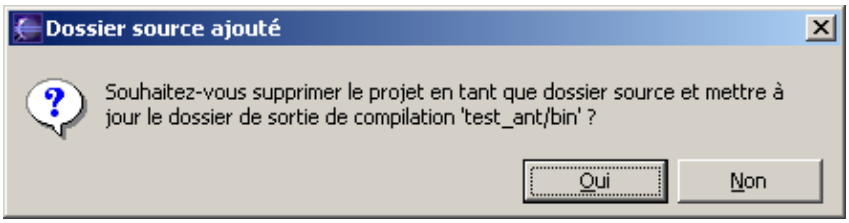
Cliquez sur le bouton "Créer un dossier ".



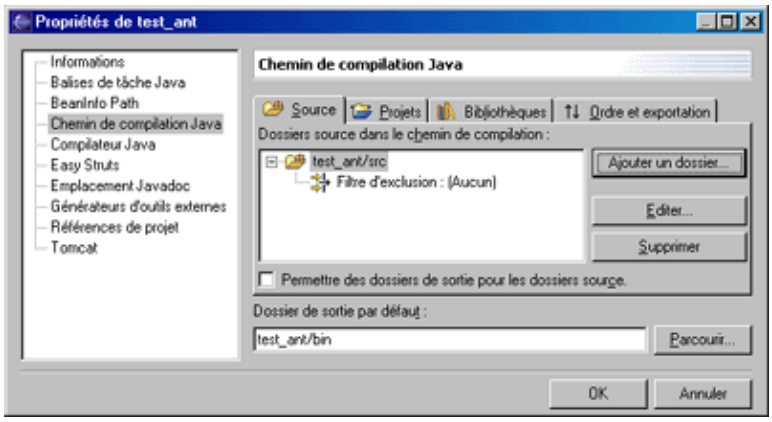
Il faut saisir le nom du répertoire qui va contenir les sources (par exemple src) et cliquer sur le bouton "OK".



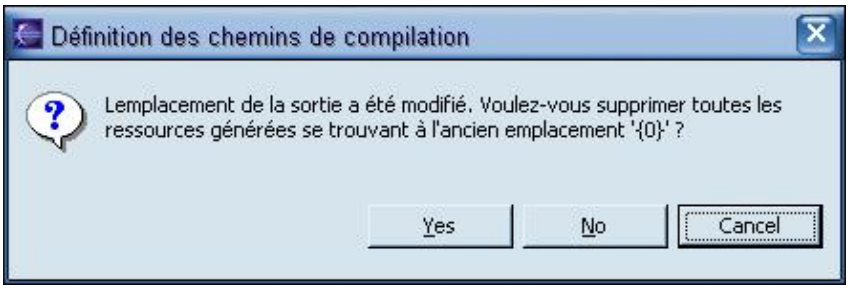
Cliquez sur le bouton "OK".



En cliquant sur "Oui", Eclipse va automatiquement créer un répertoire bin qui va contenir le résultat des compilations des sources.



Cliquez sur le bouton "OK".



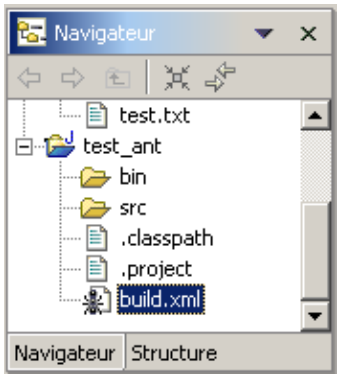
Cliquez sur le bouton "Yes".

Il faut ensuite déplacer les fichiers .java existant dans le répertoire src en effectuant un copier/coller dans la vue "Navigateur".

Il faut ensuite créer un répertoire build contenant deux sous dossiers : lib et doc. Ces dossiers vont contenir respectivement les fichiers de distribution générés (.jar, .war, .ear selon le type d'application) et la documentation des classes au format Javadoc.

9.2. Création du fichier build.xml

Les ordres de générations sont fournis à Ant sous la forme d'un fichier au format xml nommé build.xml. Il faut créer ce nouveau fichier à la racine du projet.



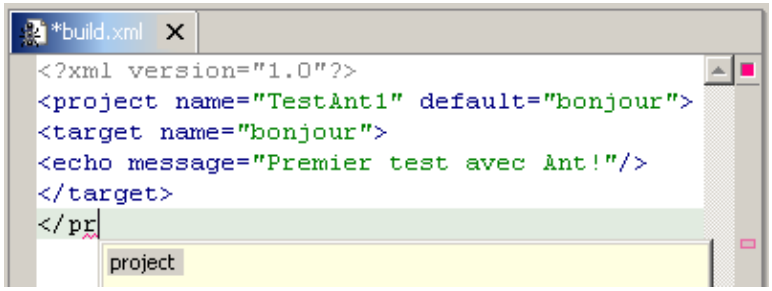
Le fichier est automatiquement reconnu comme étant un fichier de configuration pour Ant : une icône particulière contenant une fourmi est associée au fichier.

Il suffit ensuite d'éditer le fichier pour insérer les paramètres d'exécution.

Exemple : afficher un message

```
<?xml version="1.0"?>
<project name="TestAnt1" default="bonjour">
  <target name="bonjour">
    <echo message="Premier test avec Ant!"/>
  </target>
</project>
```

Un éditeur particulier est dédié à l'édition du fichier build.xml de Ant. Il propose notamment un achèvement du code pour les tags en utilisant la combinaison de touches Ctrl + espace.

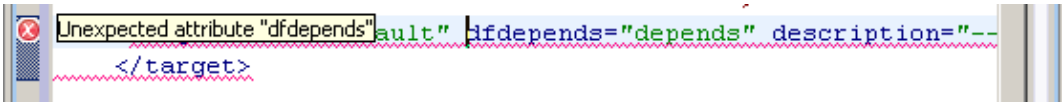


La vue structure affiche l'arborescence du fichier.

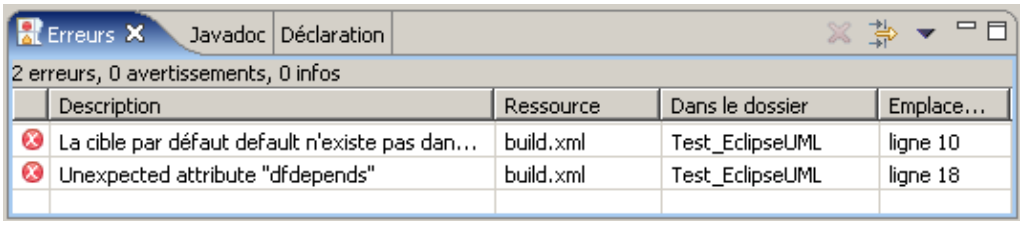
Une fois le contenu du fichier saisi, il suffit de l'enregistrer.

3.0

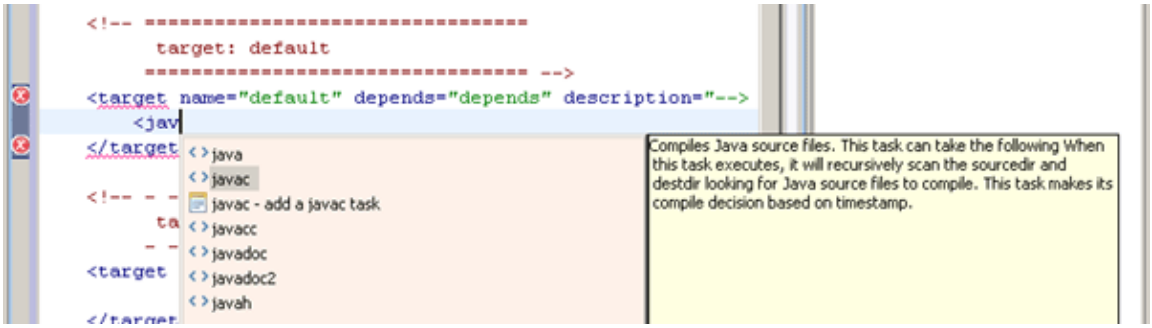
Les erreurs dans le fichier de configuration sont signalées dans l'éditeur avec possibilité d'obtenir une bulle d'aide précisant le problème en laissant le curseur de la souris sur la petite icônes rouge avec une croix blanche.



A la sauvegarde du fichier de configuration, les erreurs persistantes sont signalées dans la vue « Erreurs »

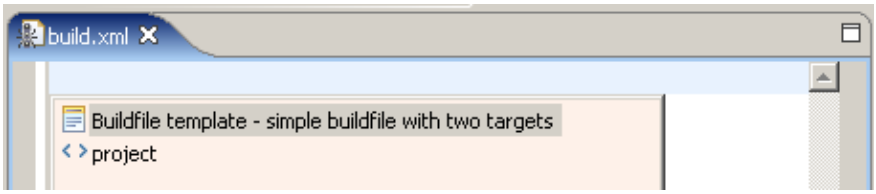


L'éditeur de fichier de configuration de Ant propose une complétion de code avec une bulle d'aide qui permet de fournir des informations sur l'entité sélectionnée.



L'éditeur propose des modèles de code utilisable via la combinaison de touches Ctrl+espace.

Par exemple, à la création d'un fichier build.xml vide, il est possible de demander l'insertion d'un modèle contenant deux cibles. Une bulle d'aide fournit un aperçu du code qui sera inséré.



Il existe aussi des modèles pour de nombreuses tâches Ant.

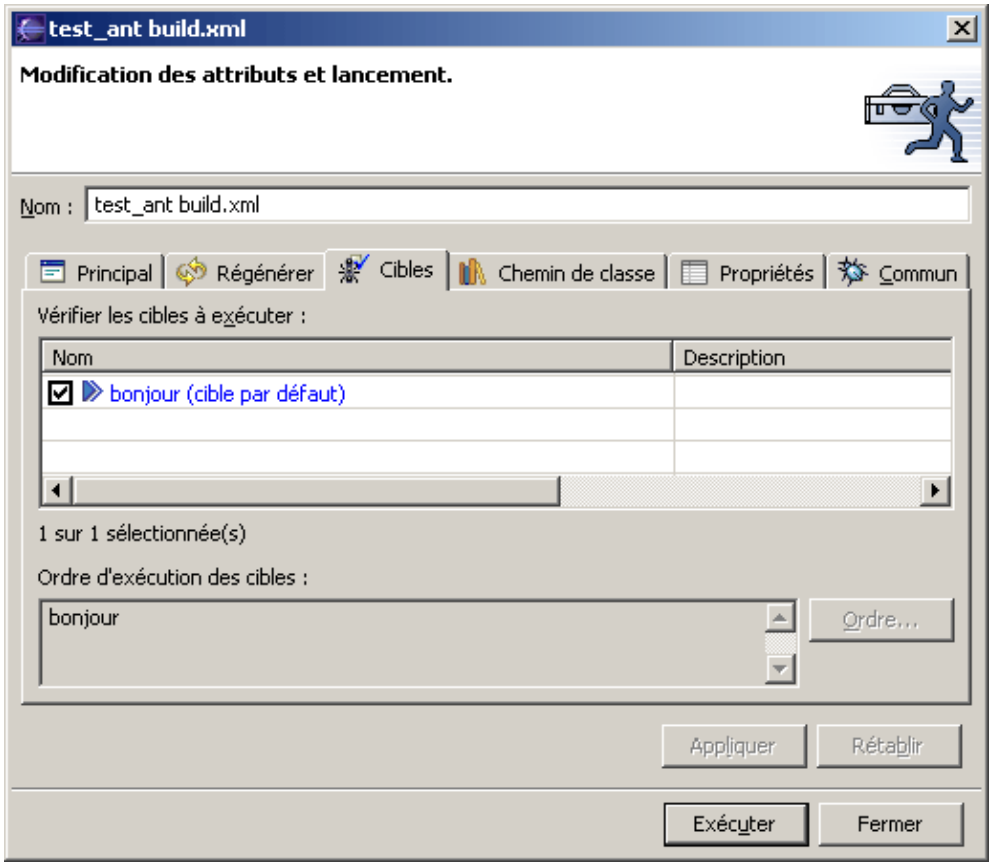
L'éditeur de fichier de configuration d'Ant peut formater le code source XML du fichier de configuration en sélectionnant l'option « Formater » du menu contextuel ou en utilisant la combinaison de touches Ctrl+Maj+F.

L'exécution de Ant se fait par défaut dans une machine virtuelle dédiée et non plus dans la machine virtuelle dans laquelle Eclipse s'exécute.

9.3. Exécuter Ant

Pour exécuter Ant avec un fichier build.xml, il suffit dans la vue "Navigateur" ou "Packages" de sélectionner ce fichier build.xml et d'utiliser l'option "Exécuter Ant" du menu contextuel.

Une boite de dialogue s'ouvre. Elle permet de modifier quelques paramètres externes à Ant et de lancer l'exécution.



Par défaut, la tâche définie par défaut dans le fichier build.xml est sélectionnée.

Pour lancer l'exécution, il suffit de cliquer sur le bouton "Exécuter".

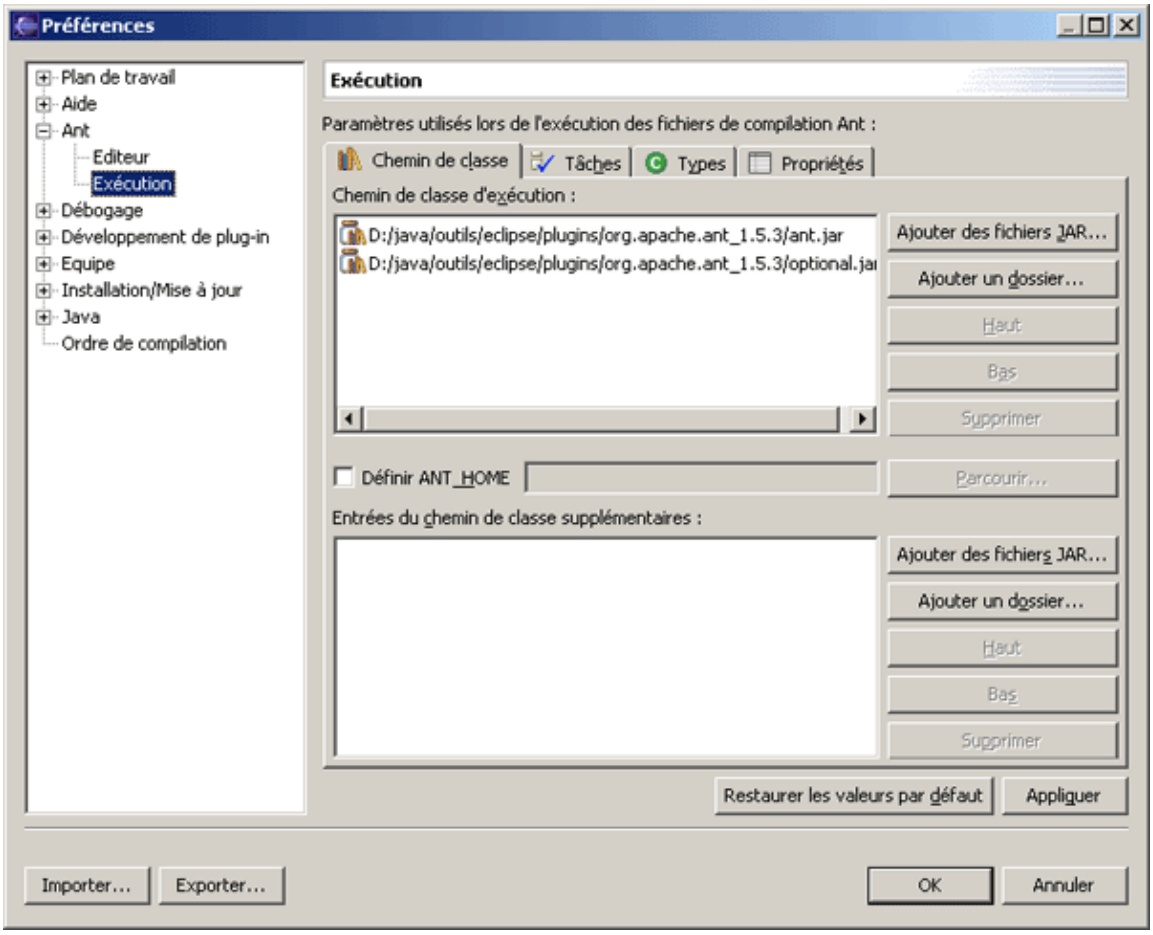
Le résultat de l'exécution s'affiche dans la vue "Console"

Exemple :

```
Buildfile: I:\eclipse\workspace\test_junit\build.xml

bonjour:
        [echo] Premier test avec Ant!
BUILD SUCCESSFUL
Total time: 401 milliseconds
```

9.4. Les paramètres

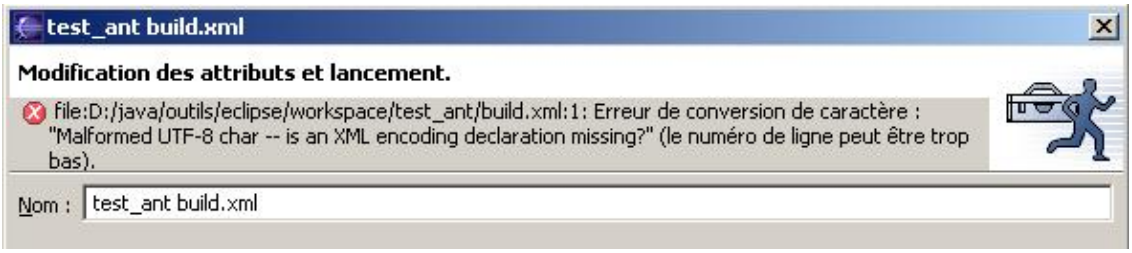
Dans les préférences, il est possible de préciser des paramètres par défaut utilisés lors de l'édition d'un fichier ant ou de son exécution.



9.5. Résolution des problèmes

Plusieurs problèmes peuvent survenir lors de l'utilisation de Ant. Voici une solution pour quelques uns d'entre eux.

9.5.1. Utilisation de caractères accentués



Il faut ajouter l'attribut encoding avec le jeux de caractères utilisés dans le prologue du fichier build.xml.

Exemple :

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
```

9.5.2. Impossible de lancer la tache javadoc

Exemple :

```
Buildfile: I:\eclipse\workspace\test_junit\build.xml
init:
        [echo] Generation numero : 7 du July 2 2003
compil:
doc:
     [javadoc] Generating Javadoc
     [javadoc] Javadoc execution
     [javadoc] BUILD FAILED: file:I:/eclipse/workspace/test_junit/build.xml:37:
     Javadoc failed: java.io.IOException: CreateProcess: javadoc.exe -d "I:\eclipse
    \workspace\test_junit\build\doc" -use -package -classpath "I:\eclipse\s
     tartup.jar;I:\eclipse\workspace\test_junit\junit.jar" -version
     -author "I:\eclipse\workspace\test_junit\src\MaClasse.java" "I:\eclipse
    \workspace\test_junit\src\MaClasse2.java" error=2
Total time: 681 milliseconds
```

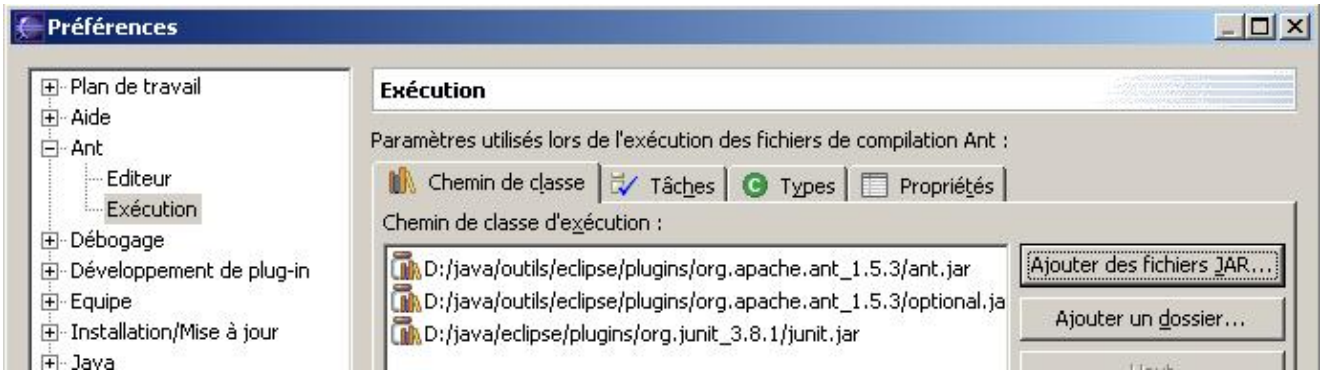
Il faut vérifier la présence de l'outil dans les répertoires désignés par la variable d'environnement PATH du système d'exploitation. Dans le cas de Javadoc sous Windows, il faut s'assurer que le répertoire %JAVA_HOME%\bin soit inséré dans la variable PATH. Si cette dernière doit être modifiée, il faut arrêter et relancer Eclipse après la modification pour que celle ci soit prise en compte.

9.5.3. Impossible d'utiliser la tâche JUnit

Exemple :

```
Buildfile: I:\eclipse\workspace\test_junit\build.xml
init:
        [echo] Generation numero : 13 du July 2 2003
compil:
test:
       [junit] BUILD FAILED: file:I:/eclipse/workspace/test_junit/build.xml:62:
 Could not create task or type of type: junit.
Ant could not find the task or a class this task relies upon.
```

Dans les préférences, il faut rajouter le fichier junit.jar dans l'onglet "Classpath" de l'arborescence "Ant/Runtime"



9.6. Un exemple complet

Exemple :

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<project name="TestAnt1" default="all">
  <description>
    Génération de l'application
  </description>

  <property name="bin" location="bin"/>
  <property name="src" location="src"/>
  <property name="build" location="build"/>
  <property name="doc" location="${build}/doc"/>
  <property name="lib" location="${build}/lib"/>
  <property name="junit_path" value="junit.jar"/>

  <target name="init" description="Initialisation">
    <tstamp/>
    <buildnumber file="numerobuild.txt" />
    <echo message="Generation numero : ${build.number} du ${TODAY}"/>
  </target>

  <target name="compil" depends="init" description="Compilation">
    <javac srcdir="${src}" destdir="${bin}">
          <classpath>
        <pathelement path="${java.class.path}"/>
        <pathelement location="${junit_path}"/>
      </classpath>
    </javac>
  </target>

  <target name="all" depends="init, compil, test, doc"
    description="Generation complete">
    <echo message="Generation complete."/>
  </target>
  <target name="doc" depends="compil" description="Generation de la documentation">
    <javadoc destdir="${doc}" author="true" version="true" use="true" package="true">
      <fileset dir = "${src}">
        <include name="**/*.java"/>
        <exclude name="**/*Test*"/>
      </fileset>
      <classpath>
        <pathelement path="${java.class.path}"/>
        <pathelement location="${junit_path}"/>
```

```
      </classpath>
    </javadoc>
  </target>

  <target name="test" depends="compil" description="Executer les tests avec JUnit">
    <junit fork="yes" haltonerror="true" printsummary="on">
      <formatter type="plain" usefile="false" />
      <test name="ExecuterLesTests"/>
      <classpath>
        <pathelement location="${bin}"/>
        <pathelement location="${junit_path}"/>
      </classpath>
    </junit>
  </target>
</project>
```

Résultat de l'exécution :

```
Buildfile: I:\eclipse\workspace\test_junit\build.xml
init:
        [echo] Generation numero : 16 du July 2 2003
compil:
test:
       [junit] Running ExecuterLesTests
       [junit] Tests run: 3, Failures: 0, Errors: 0, Time elapsed: 0,02 sec
       [junit] Testsuite: ExecuterLesTests
       [junit] Tests run: 3, Failures: 0, Errors: 0, Time elapsed: 0,02 sec
       [junit]
       [junit] Testcase: testCalculer took 0,01 sec
       [junit] Testcase: testCalculer took 0 sec
       [junit] Testcase: testSommer took 0 sec
doc:
     [javadoc] Generating Javadoc
     [javadoc] Javadoc execution
     [javadoc] Loading source file I:\eclipse\workspace\test_junit\src\MaClasse.java...
     [javadoc] Loading source file I:\eclipse\workspace\test_junit\src\MaClasse2.java...
     [javadoc] Constructing Javadoc information...
     [javadoc] Standard Doclet version 1.4.1
     [javadoc]
     [javadoc] Building tree for all the packages and classes...
     [javadoc] Building index for all the packages and classes...
     [javadoc] Building index for all classes...
all:
        [echo] Generation complete.
BUILD SUCCESSFUL
Total time: 4 seconds
```

10. JUnit et Eclipse

La version 2.1 d'Eclipse intègre la possibilité d'utiliser JUnit directement dans l'IDE.

JUnit est un framework open source pour réaliser des tests unitaires sur du code Java. Le principal intérêt est de s'assurer que le code répond toujours au besoin même après d'éventuelles modifications.

Le but est d'automatiser les tests. Ceux ci sont exprimés dans des classes sous la forme de cas de tests avec leurs résultats attendus. JUnit exécute ces tests et les comparent avec ces résultats.

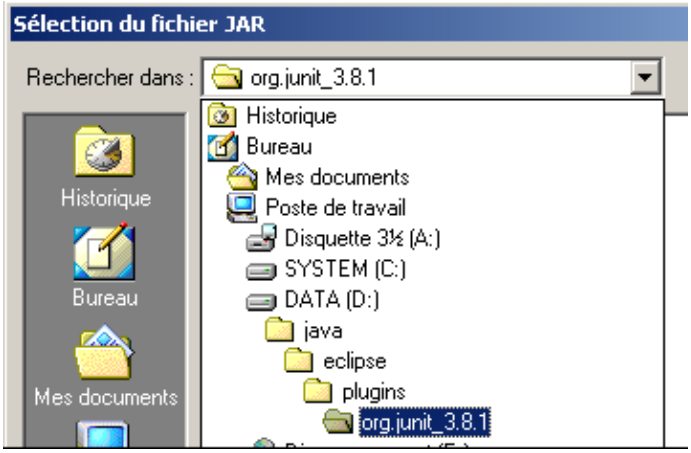
Avec Junit, l'unité de tests est une classe dédiée qui regroupe des cas de tests. Ces cas de tests exécutent les tâches suivantes :

- création d'une instance de la classe et de tout autre objet nécessaire aux tests
- appel de la méthode à tester avec les paramètres du cas de test
- comparaison du résultat obtenu avec le résultat attendu : en cas d'échec, une exception est levée

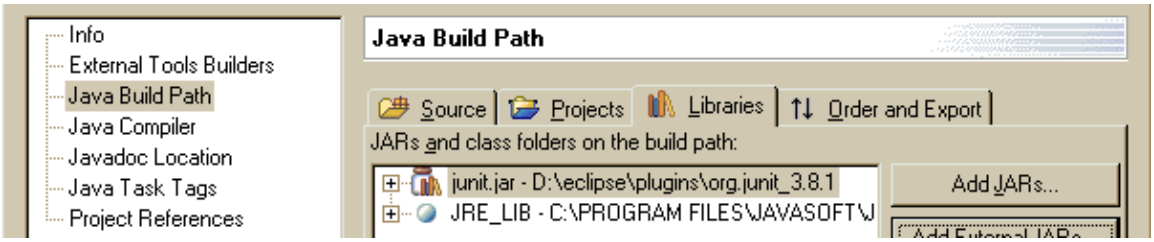
10.1. Paramétrage de l'environnement

Pour pouvoir utiliser JUnit dans un projet, il faut ajouter le fichier junit.jar dans le classpath du projet. Il faut pour cela :

- sélectionner les propriétés du projet
- dans l'onglet librairies, cliquer sur le bouton « Add External Jar »
- sélectionner le fichier junit.jar dans le répertoire plugins/org.junit_3.8.1 du répertoire d'Eclipse



Eclipse ajoute le fichier junit.jar au classpath



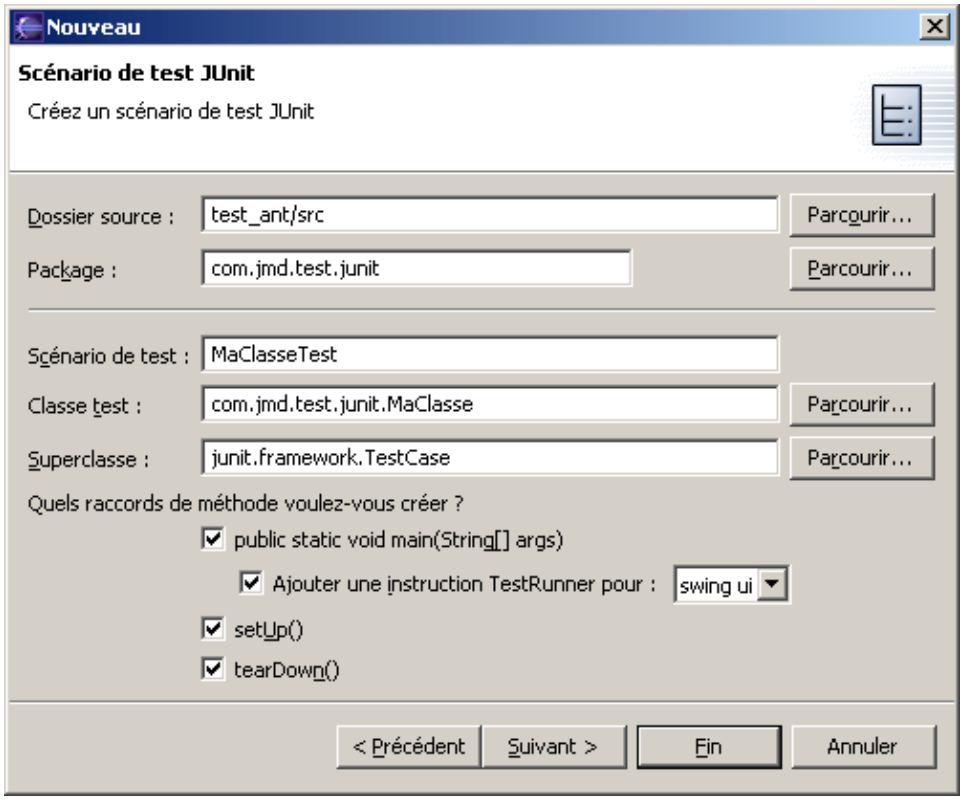
Dans ce chapitre, la classe suivante est utilisée comme classe à tester avec JUnit.

Exemple :

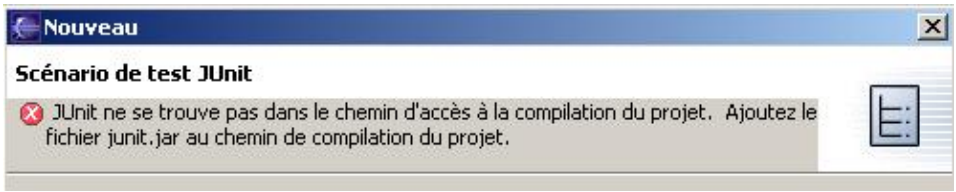
```
package com.moi.test.junit;
public class MaClasse {
  public int additioner(int a, int b) {
    return a + b;
  }
}
```

10.2. Ecriture des cas de tests

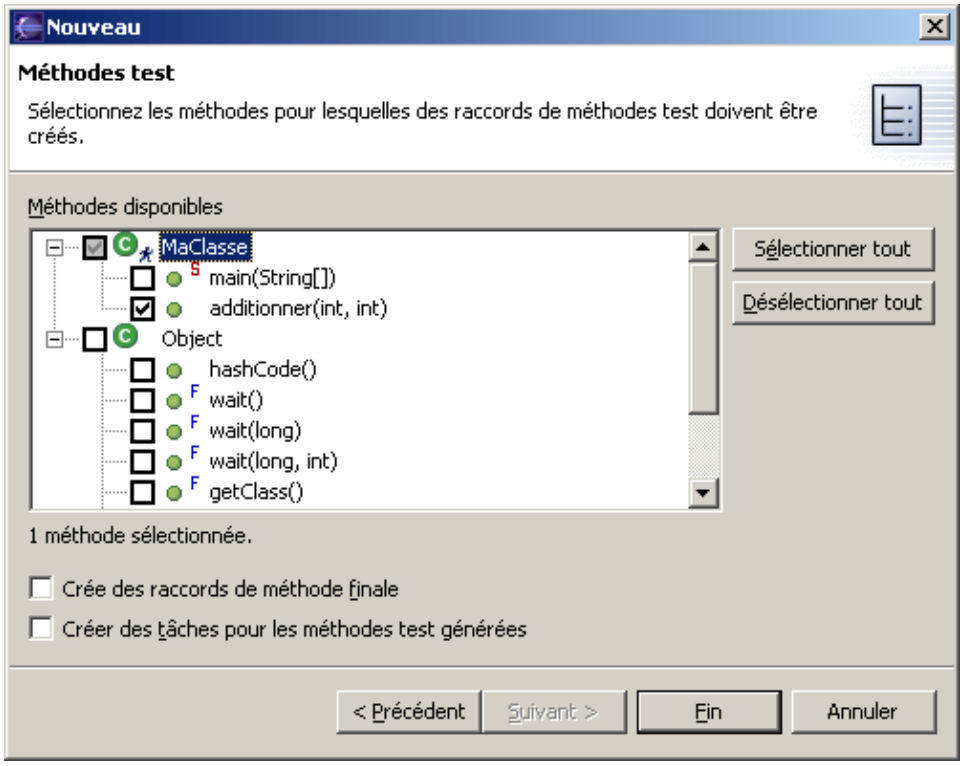
Pour utiliser JUnit, il faut créer une classe qui va contenir les cas de test. Il faut créer une nouvelle entité de type " Java / JUnit / Scénario de test ".



Si le fichier junit.jar n'est pas inclus dans le classpath du projet, un message d'erreur est affiché et il est impossible de poursuivre l'exécution de l'assistant.



Cliquez sur le bouton "Suivant".



Il faut compléter la classe générée selon les besoins : par exemple, ajouter un attribut qui va contenir une instance de la classe à tester, ajouter l'instanciation de cette classe dans la méthode setUp() et libérer cette instance dans la méthode tearDown().

Il faut ajouter les traitements nécessaires dans les méthodes testXXX() en utilisant l'API de JUnit.

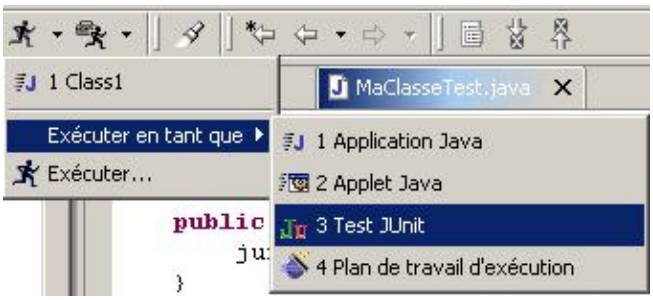
Exemple :

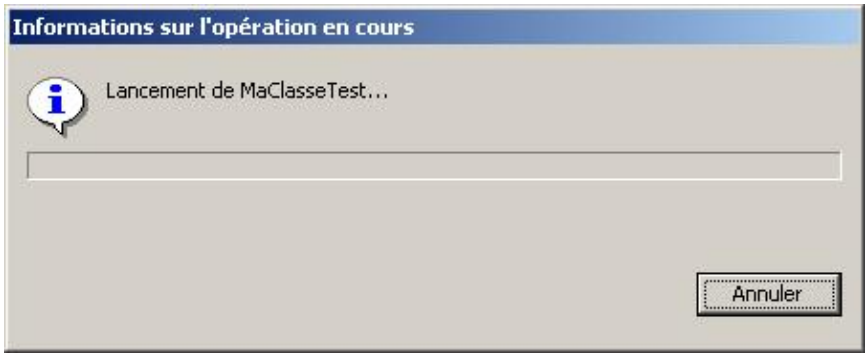
```
package com.moi.test.junit;

import junit.framework.TestCase;

public class MaClasseTest extends TestCase {
   private MaClasse maClasse = null;

   public MaClasseTest(String arg0) {
         super(arg0);
   }

   public static void main(String[] args) {
         junit.swingui.TestRunner.run(MaClasseTest.class);
   }

   protected void setUp() throws Exception {
         super.setUp();
         maClasse = new MaClasse();
   }

   protected void tearDown() throws Exception {
         super.tearDown();
         maClasse = null;
   }

   public void testAdditioner() {
         assertTrue(maClasse.additioner(2,2) == 4);
   }
}
```

JUnit utilise l'instrospection pour exécuter les méthodes commençant par test.

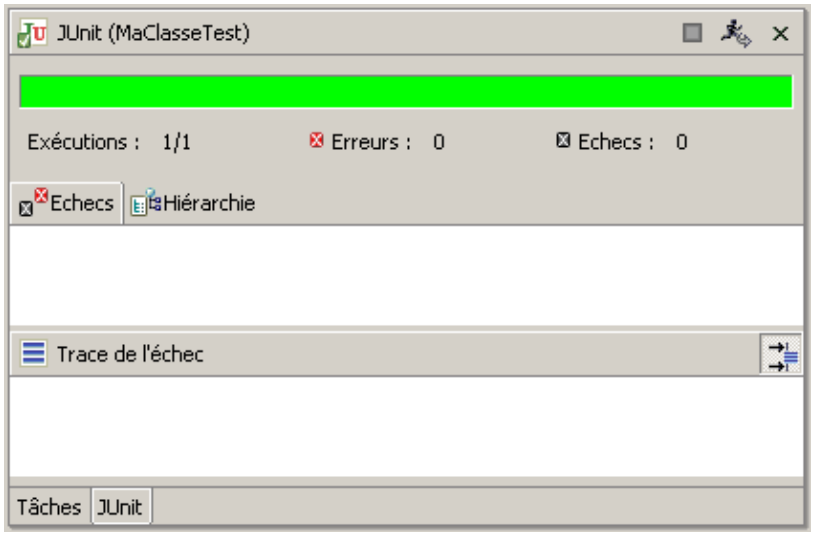
10.3. Exécution des cas de tests

Pour exécuter les tests, il faut exécuter la classe en tant que « Test JUnit ».





Eclipse exécute les tests et affiche le résultat dans une vue dédiée.



Si tous les cas de tests ont été exécutés avec succès, une ligne verte est affichée.

Cette vue contient deux onglets :

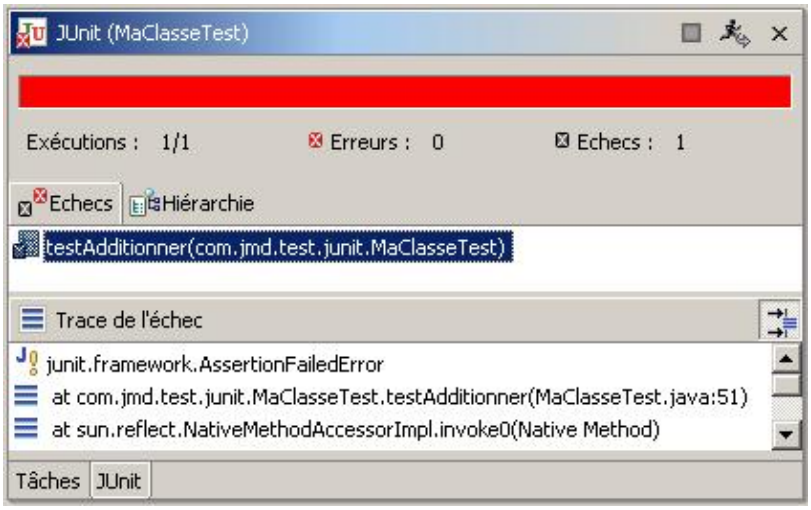
- « Failures » : contient la liste des cas de tests qui ont échoués
- « Hierarchy » : contient une arborescence des cas de tests

Dans le cas où un ou plusieurs tests échouent, la ligne est rouge.

Exemple : si le cas test suivant est ajouté :

```
public void testAdditioner() {
      assertTrue(maClasse.additioner(2,2) == 4);
      assertTrue(maClasse.additioner(2,3) == 4);
}
```

Une exécution de l'exemple précédent permet d'avoir le résultat suivant :



Partie 3 : les fonctions avancées d'Eclipse

Cette troisième partie présente des fonctionnalités avancées d'Eclipse.

Elle comporte les chapitres suivants :

- L'aide dans Eclipse : présente comment obtenir de l'aide lors de l'utilisation d'Eclipse.
- CVS 2.0 et Eclipse 2.1 : détaille l'utilisation de CVS 2.0 avec Eclipse 2.1.
- CVS 2.5 et Eclipse 3.0 : détaille l'utilisation de CVS 2.5 avec Eclipse 3.0.
- La gestion de la plate−forme : détaille les fonctionnalités proposées pour gérer les plug−ins et la plate−forme Eclipse.

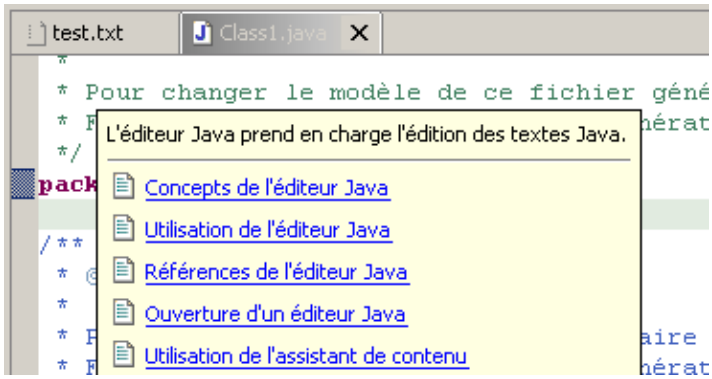
11. L'aide dans Eclipse

11.1. L'aide en ligne

L'aide en ligne est disponible dans toute l'interface de l'IDE au moyen de la touche F1. Cette aide est contextuelle en fonction de l'élément sélectionné dans l'interface au moment de l'appui sur la touche.

Le choix des sujets se rapportant à l'élément courant est affiché dans une info bulle. Il suffit de cliquer sur l'élément sélectionné pour que l'aide en ligne correspondante s'affiche.

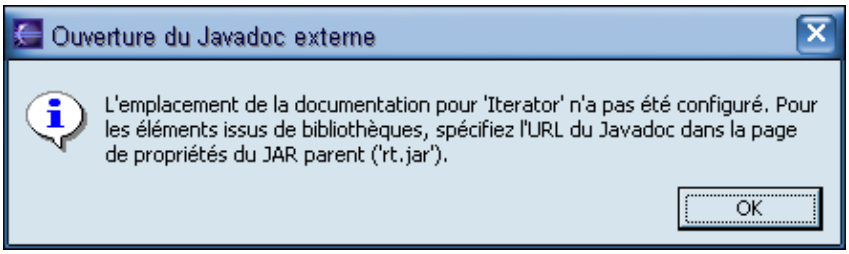
Exemple avec l'appui sur F1 dans l'éditeur de code Java



11.2. L'aide Javadoc

Si la configuration de l'IDE est correcte, il est possible d'accéder directement à la page Javadoc d'un élément java en plaçant le curseur sur cet élément et en appuyant sur F2.

Si l'accès à la documentation Javadoc pour l'élément n'est pas configuré un message d'erreur s'affiche.



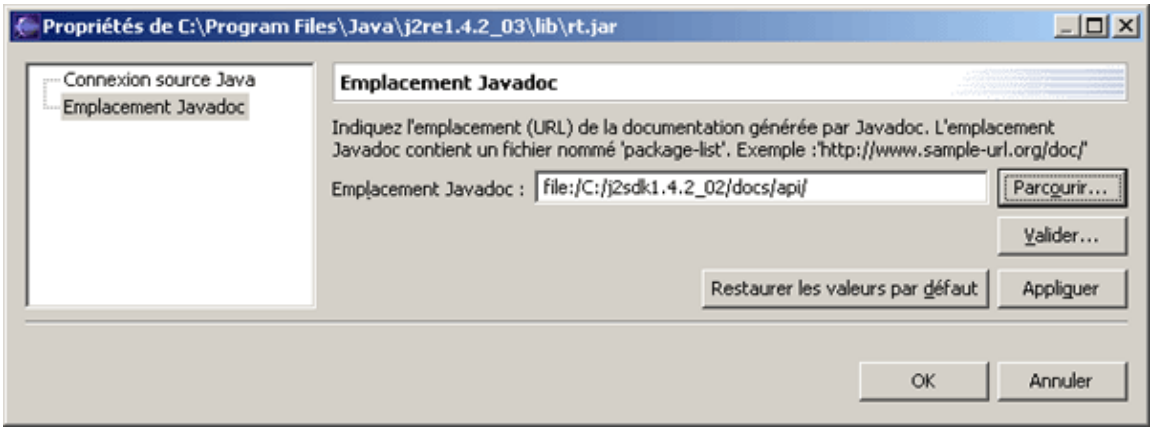
Pour configurer l'IDE, il faut sélectionner la ressource qui contient l'élément dans la vue "Packages" :

- un projet
- un fichier .jar

L'exemple ci dessous permet d'associer la doc du JDK avec le fichier rt.jar

Il faut sélectionner le fichier rt.jar du projet dans la vue "Packages" et sélectionner le menu contextuel "Propriétés".

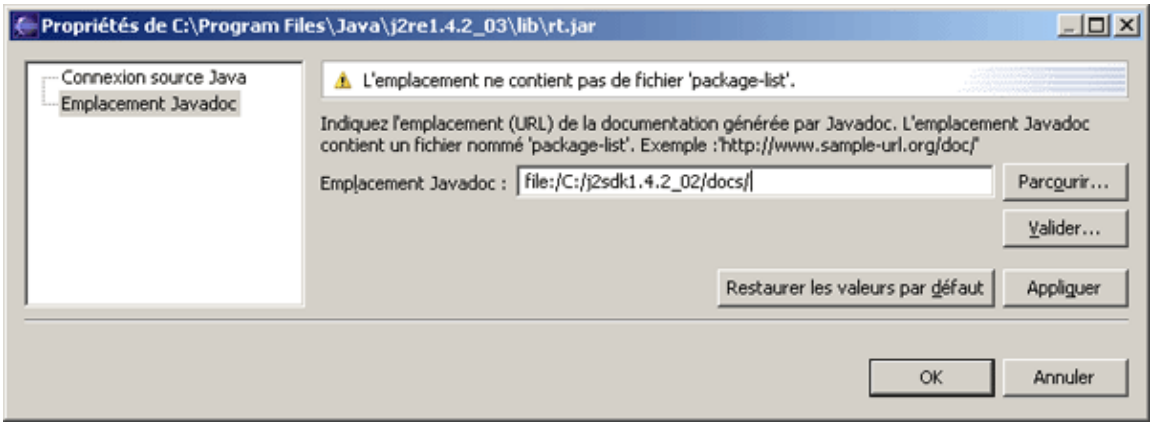
Il faut sélectionner "Emplacement Javadoc" puis cliquer sur parcourir pour sélectionner le répertoire qui contient le fichier package−list de la documentation.



Le bouton "Valider" permet de vérifier si l'emplacement sélectionné contient les éléments nécessaires.

Enfin, il faut cliquer sur le bouton "OK" pour valider les changements.

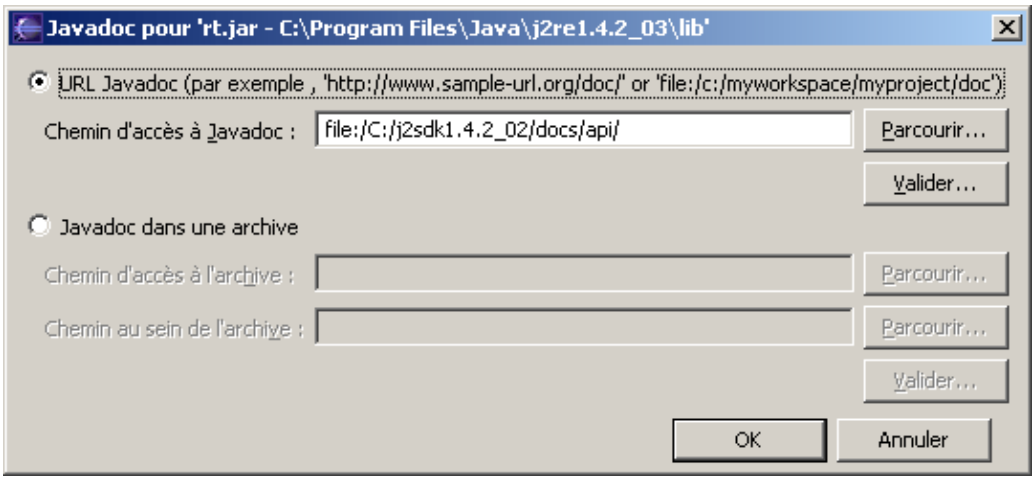
Si le chemin n'est pas correct, un message d'erreur est affiché.



Il est alors possible dans l'éditeur de code Java de positionner le curseur sur un élément contenu dans un des emplacements Javadoc pour que l'aide en ligne affiche la page concernée par l'élément dans la documentation Javadoc.

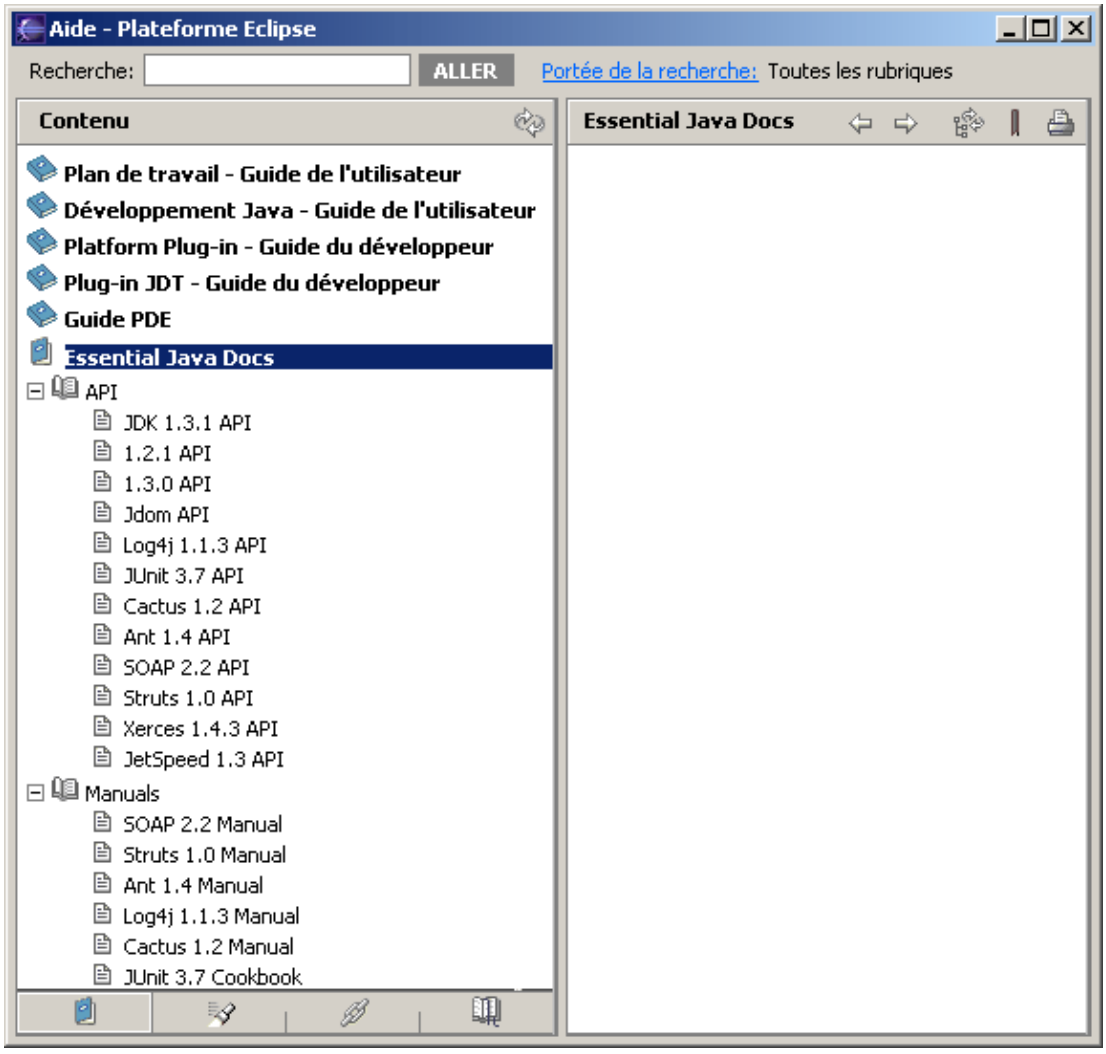
Ce processus est applicable à toutes les API dont la documentation Javadoc est disponible.

Il n'est plus utile de décompresser le contenu d'une documentation Javadoc pour pouvoir l'associer à une bibliothèque.



11.3. Le plug−in Java docs de Crionics

La société Crionics propose un plug−in qui s'intègre dans l'aide en ligne d'Eclipse et qui contient la documentation du JDK 1.3 et de quelques API open source.



Ce plug−in crée une entrée nommée "Essentials Java Docs" dans la table des matières de l'aide en ligne. Cette documentation regroupe la documentation du JDK 1.3.1, de quelques API open source fréquement utilisée

ainsi que quelques manuels pour ces API.

Ce plug−in est téléchargeable sur le site de Crionics (environ 30 Mo):
http://www.crionics.com/products/opensource/eclipse/com.crionics.java.doc.zip

Pour l'installer, il suffit de décompresser son contenu dans le répertoire plugins d'intallation d'Eclipse.

12. CVS 2.0 et Eclipse 2.1

CVS (Concurrent Versions System) est un outil libre de gestion de versions. Initialement développé pour Unix, une version pour windows NT/2000 de CVS peut être téléchargée à l'url http://www.cvsnt.org/

Toutes les données sont stockées dans un référentiel (repository). Chaque modification d'une ressource gérée par CVS associe à cette entité un numéro de révision unique.

Une version contient un ensemble de ressource, chacune ayant une révision particulière pour la version correspondante.

CVS ne verrouille pas ces ressources. Deux développeurs peuvent créer chacun une révision d'une même ressource. La fusion des deux versions est à la charge d'un des développeurs.

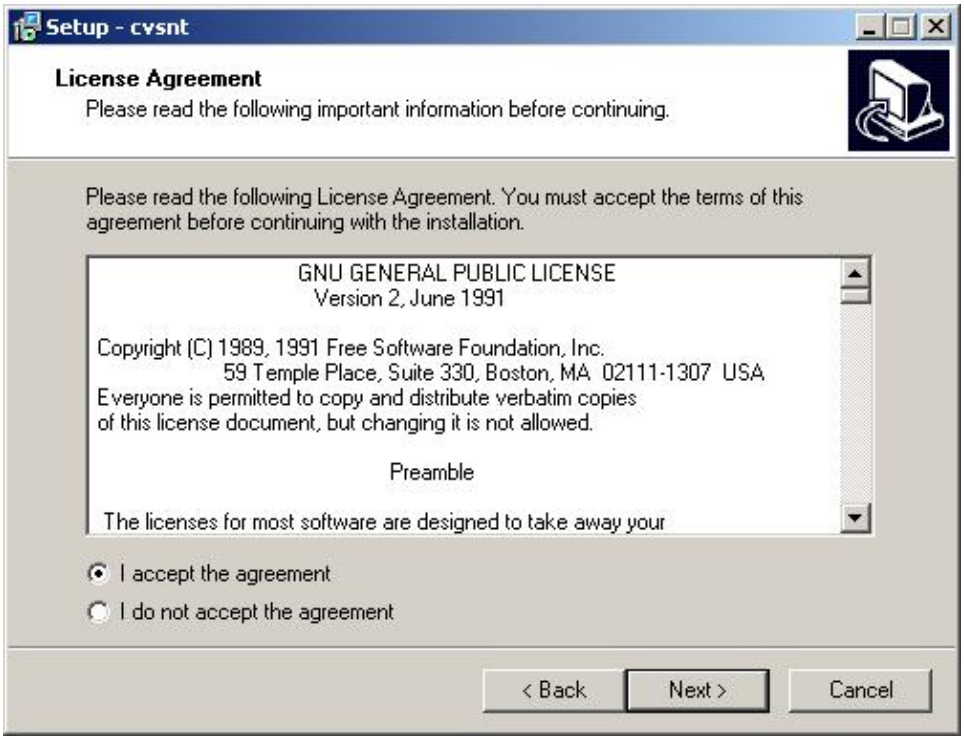
Eclipse propose une perspective pour utiliser CVS dans un projet.

12.1. Installation de cvsnt

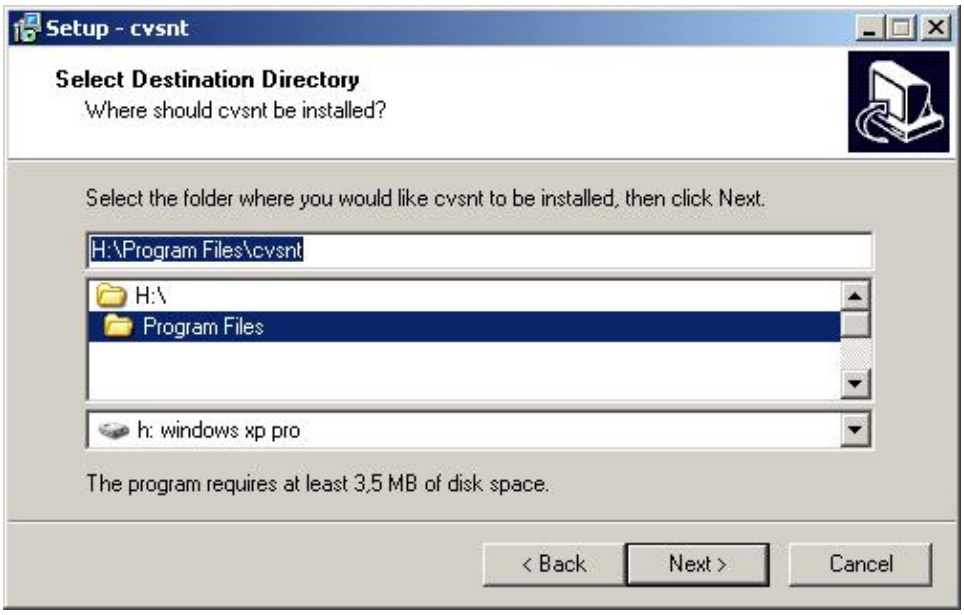
Il faut créer deux répertoires, par exemple c:\cvs\cvsrepo et c:\cvs\cvstemp

Exécuter le programme d'installation cvsnt−2.0.0.exe

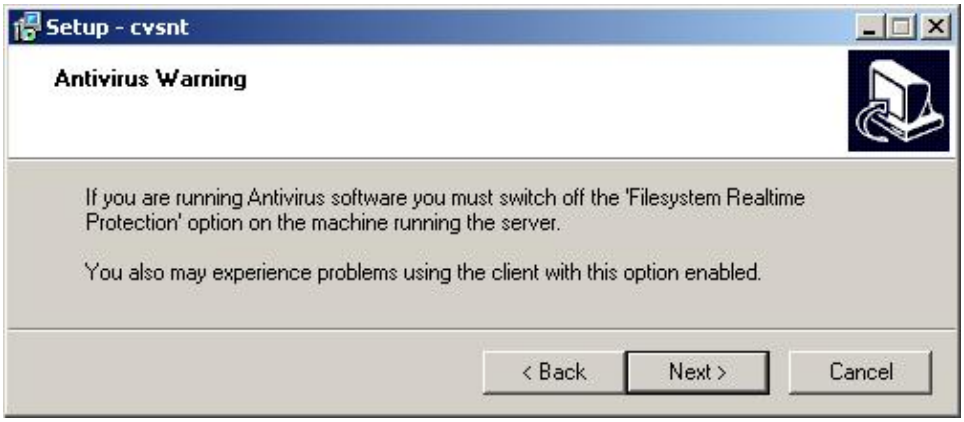
Cliquez sur le bouton «Next»

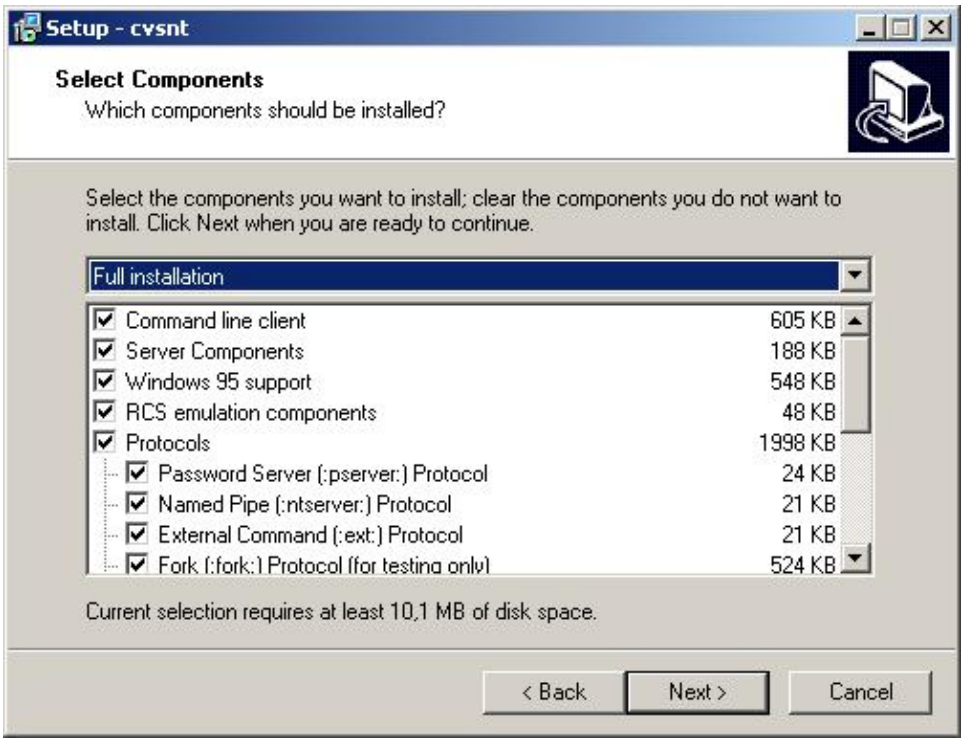


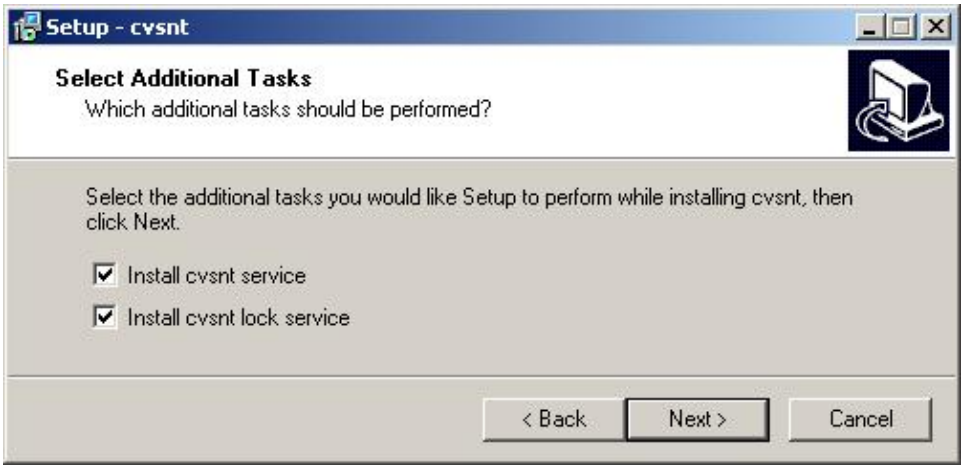
Lisez la licence et si vous l'acceptez, cliquez sur le bouton «Next»



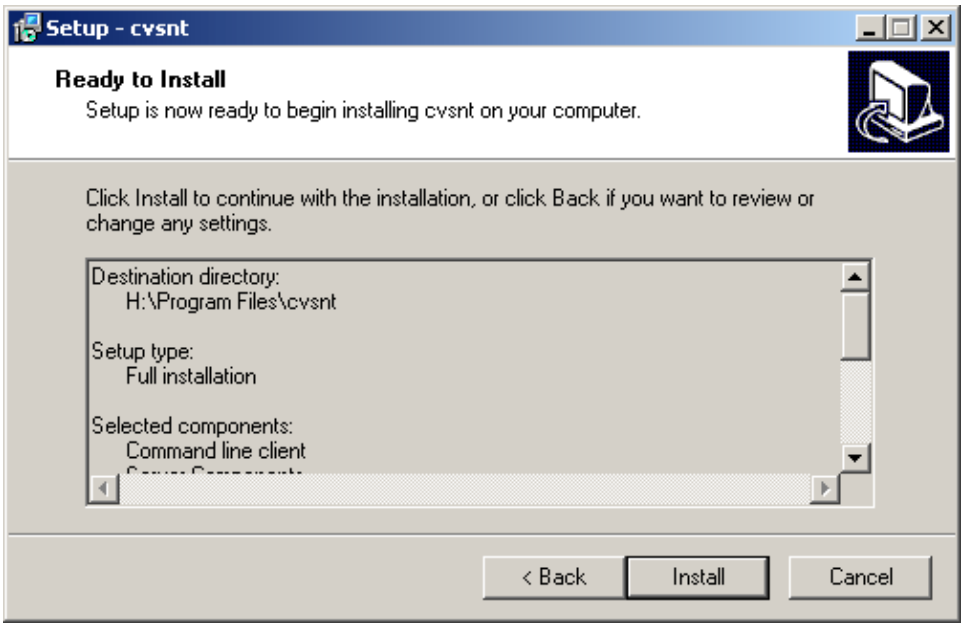
Sélectionnez le répertoire d'installation et cliquez sur le bouton «Next»



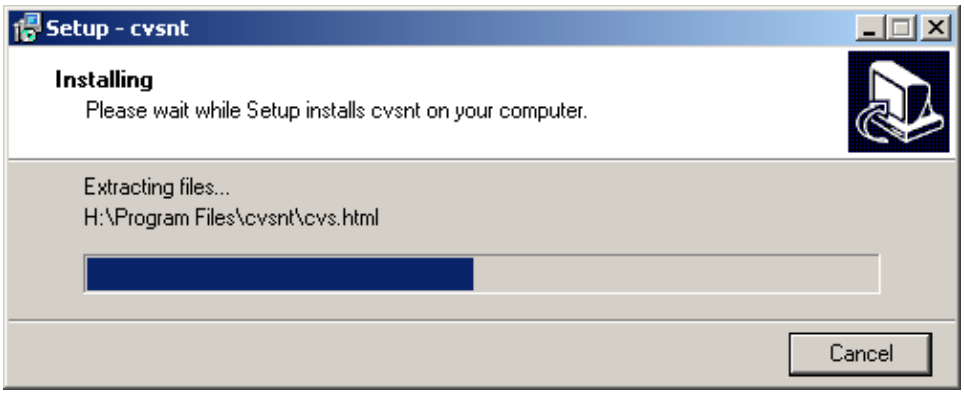




Cliquez sur le bouton «Next»



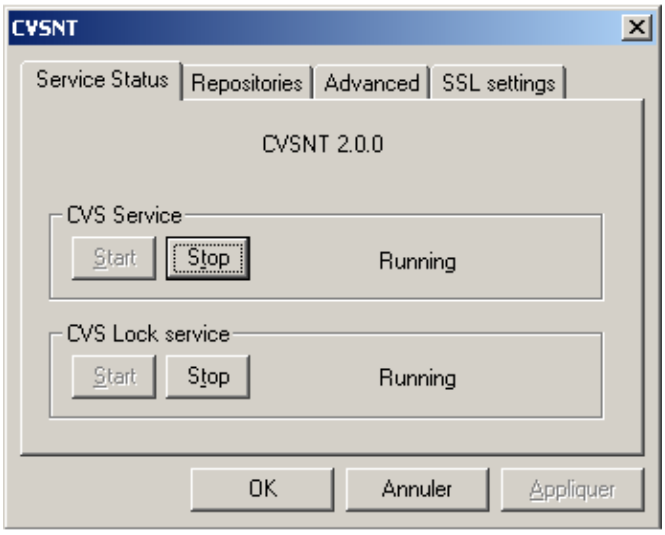
Cliquez sur le bouton «Install»



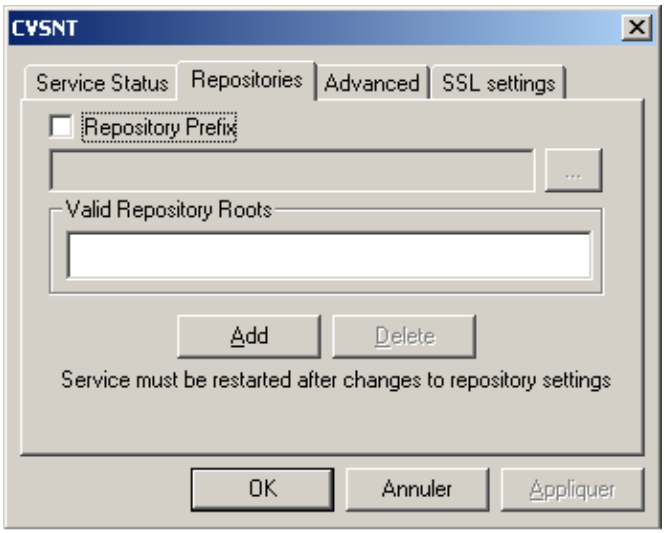
Une fois l'installation terminée, cliquez sur le bouton «Finish».

Exécutez «le service control panel» en cliquant sur l'icône 

dans le menu "Démarrer/Programmes/CVTNT" ou dans le panneau de configuration

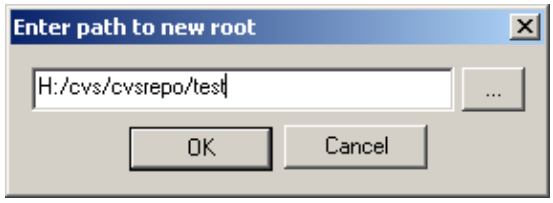


Sélectionnez l'onglet «Repositories»

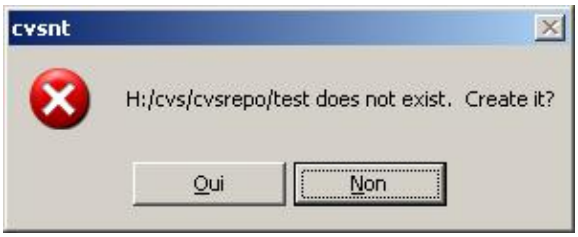


Cliquez sur «Repository prefix» et sélectionnez le répertoire cvsrepo précédemment créé.

Cliquez sur le bouton «Add»

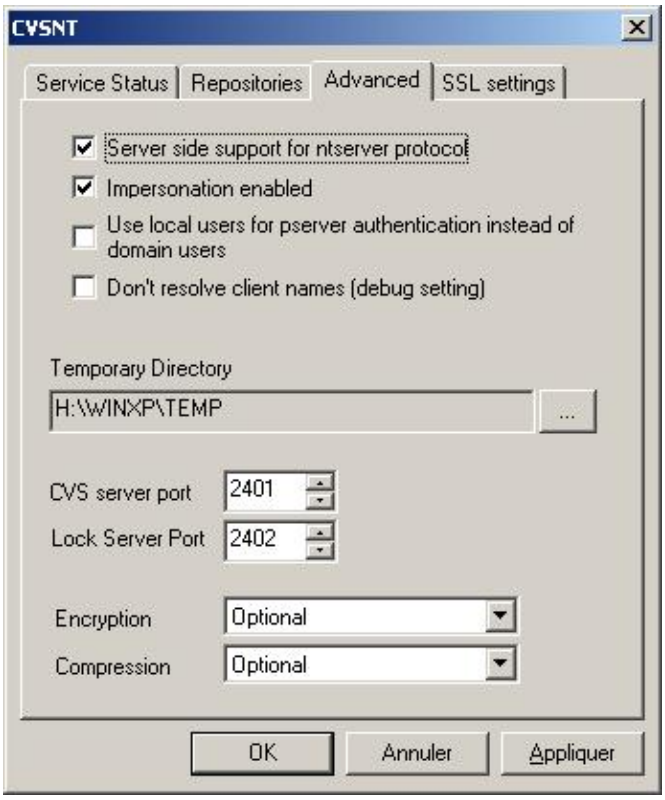


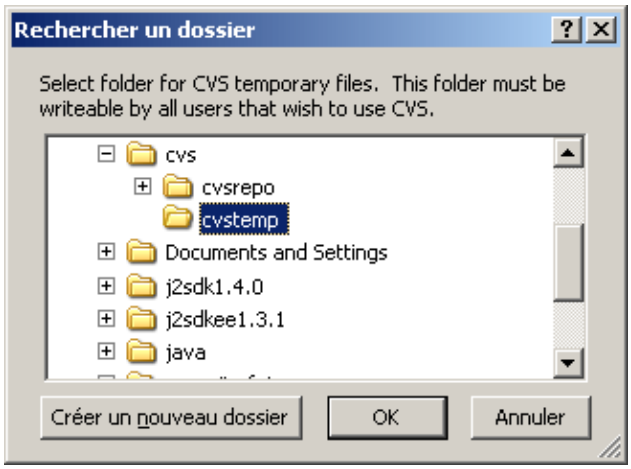
Saisissez le nom du répertoire et cliquez sur le bouton «OK»



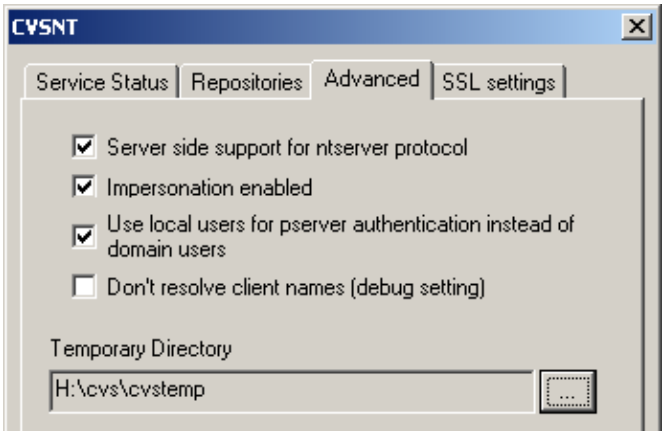
Cliquez sur le bouton «Oui»

Sélectionnez l'onglet «Advanced» et cliquez sur le bouton "..."

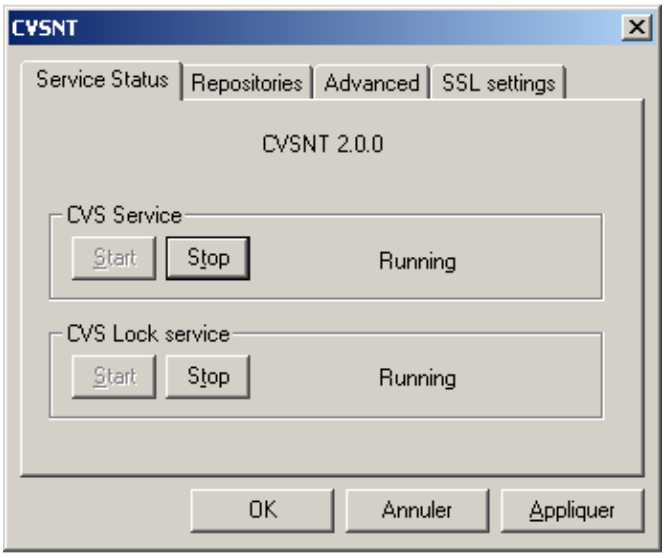




Sélectionner le répertoire temporaire précédemment créé



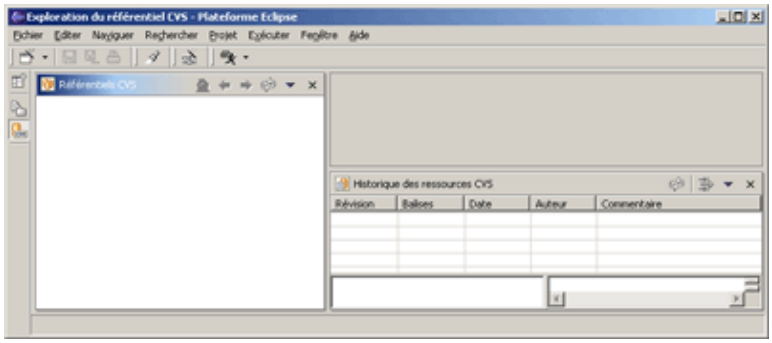
Sur l'onglet «Service Status», cliquez sur le bouton «Start»



Cliquez sur le bouton “Ok”

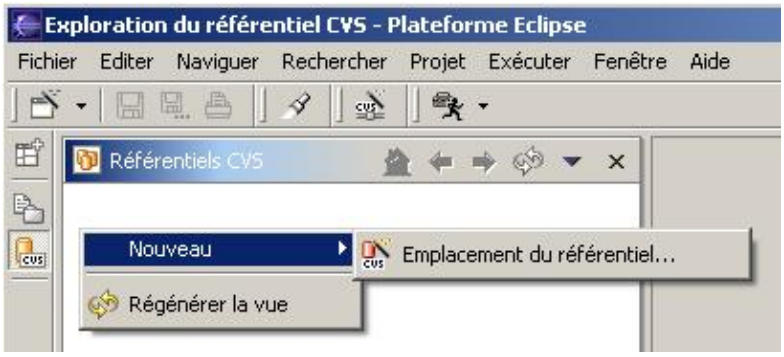
12.2. La perspective CVS

La perspective «Exploration du référentiel CVS» permet de gérer les échanges et le contenu des projets stockés sous CVS.

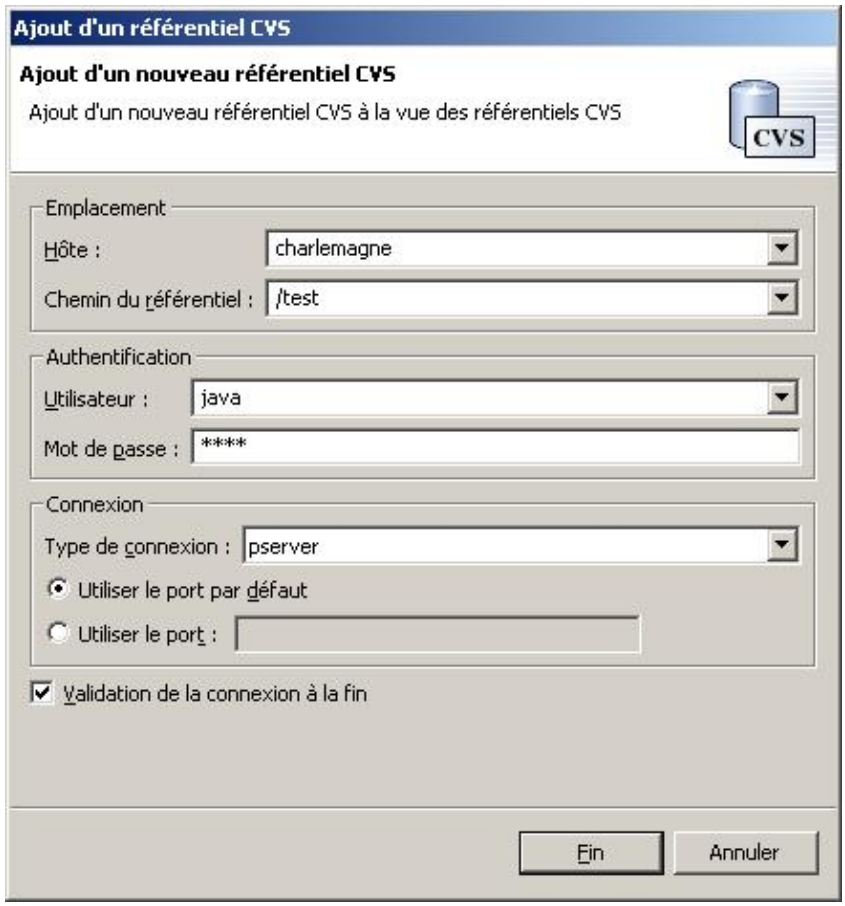


12.2.1. La création d'un emplacement vers un référentiel

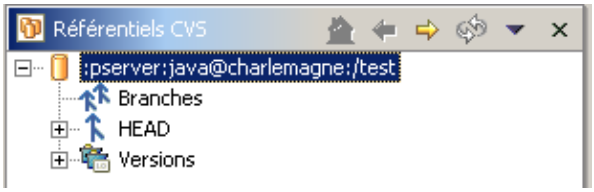
Dans la vue "Référentiels CVS", sélectionner l'option "Nouveau/Emplacement du référentiel" du menu contextuel.



Une boite de dialogue s'ouvre pour définir un nouvel emplacement. Un emplacement contient uniquement les informations sur une connexion.



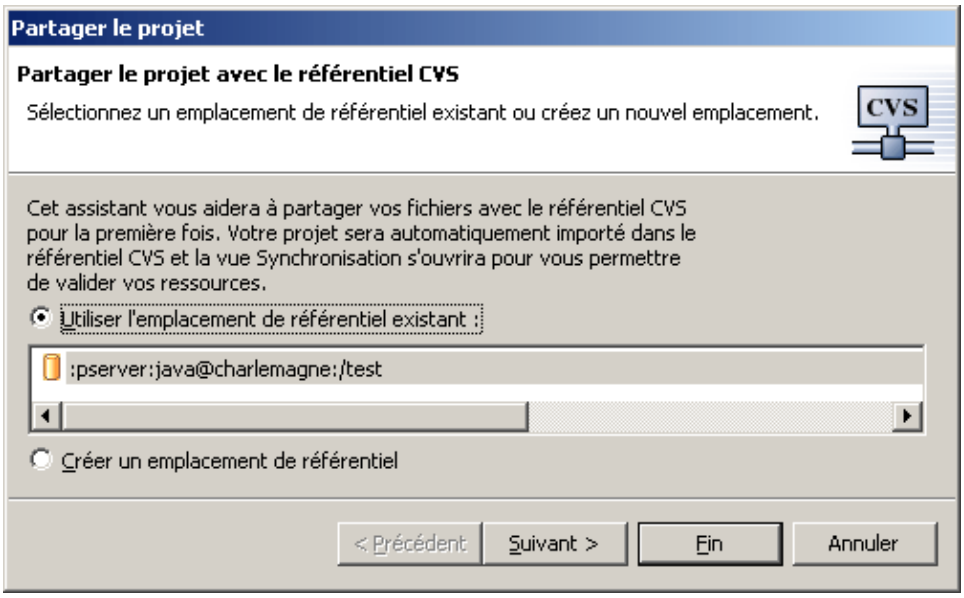
Il faut renseigner le nom de la machine, le chemin du référentiel, le nom de l'utilisateur, son mot de passe (celui de windows) et le type de connexion (utilisez pserver) puis cliquer sur le bouton "Fin"



Si la connexion ne peut être établie, un message d'erreur est affiché

12.2.2. Partager un projet

Il faut sélectionner un projet dans une vue et sélectionner l'option "Equipe/Partager le projet" du menu contextuel.

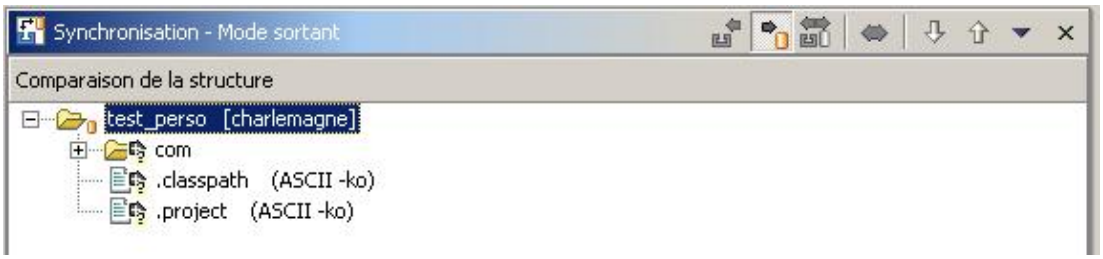


Cliquez sur le bouton "Suivant".

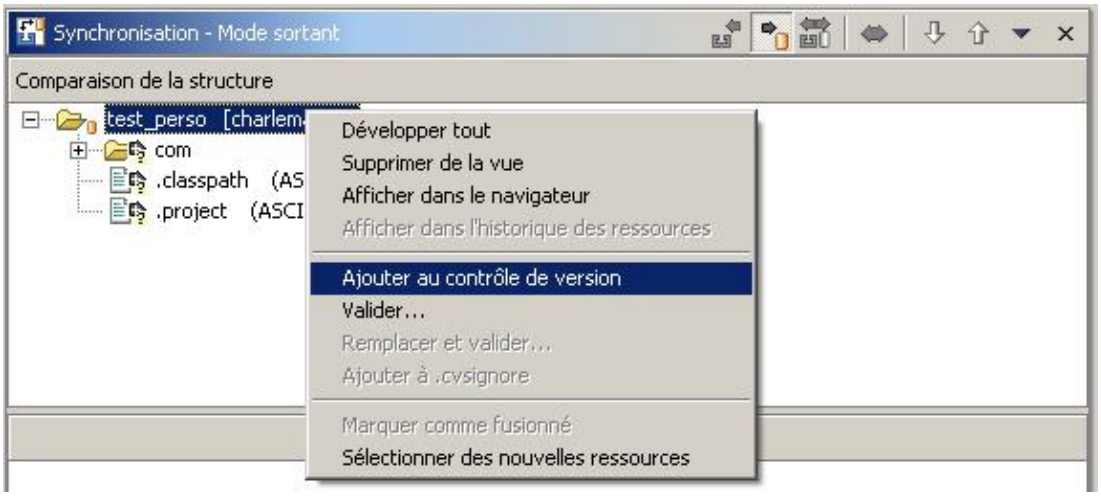
Cette étape permet de donner un nom au module : soit celui du projet Eclipse soit un nom spécifique à préciser. Cliquez sur le bouton "Suivant".

Cliquez sur le bouton "Fin".

La vue "Synchronisation − Mode sortant" affiche les fichiers qui ont été modifiés

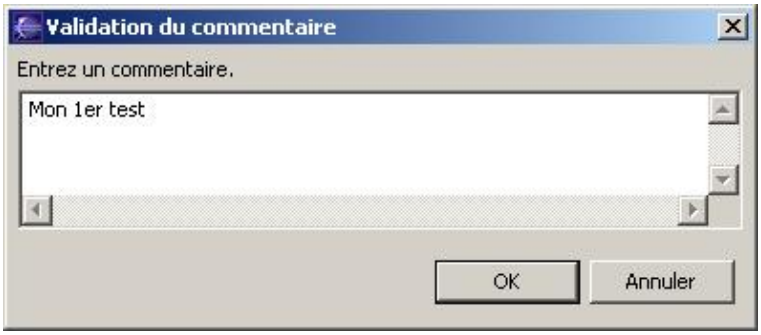


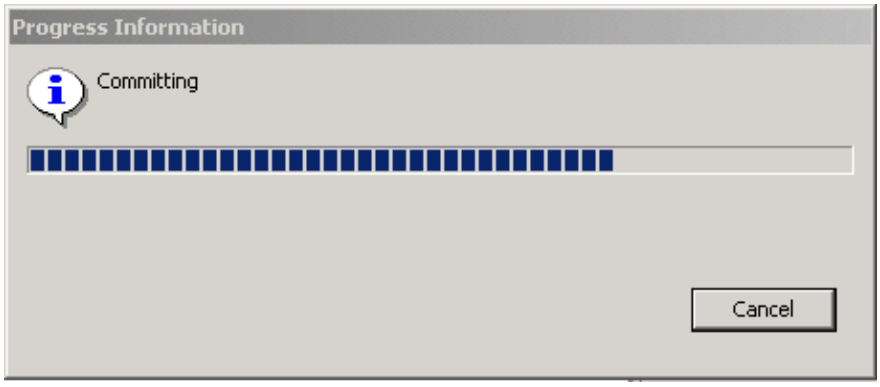
Dans cette vue, il faut sélectionner le projet et activer l'option "Ajouter au contrôle de version" du menu contextuel.



Une fois le traitement effectué, activer l'option "Valider" du menu contextuel

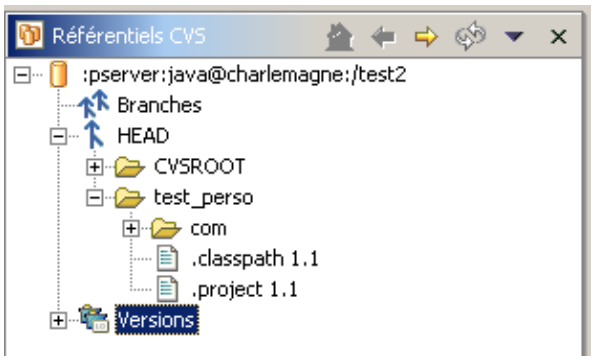
Une boite de dialogue demande la saisie d'un commentaire





12.2.3. Voir le projet dans la perspective CVS

Pour voir le projet, il faut rafraîchir la vue «Referentiel CVS» en cliquant sur le bouton  ou en activant l'option "Régénérer la vue" dans le menu contextuel.



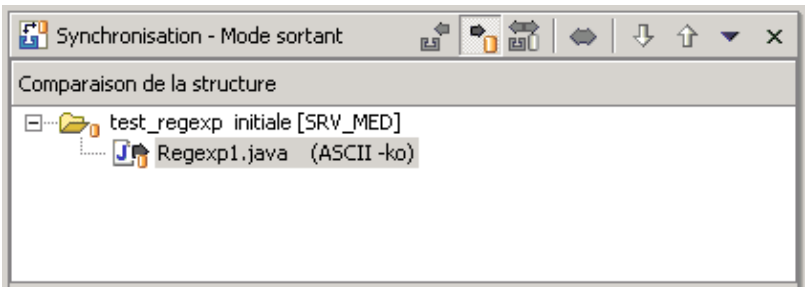
Remarque : si l'arborescence du projet n'est pas affichée, il suffit de cliquer sur le bouton et de cocher l'option "Afficher les dossiers".

12.3. L'utilisation des révisions

12.3.1. Créer une révision

Lorsqu'une ressource est modifiée et sauvegardée localement dans l"espace de travail, il est possible d'enregistrer ces modifications dans CVS sous la forme d'une révision.

Il suffit de sélectionner la ressource et d'activer l'option "Equipe/Synchroniser avec le référentiel". La vue "Synchronisation" s'affiche avec la ressource modifiée.



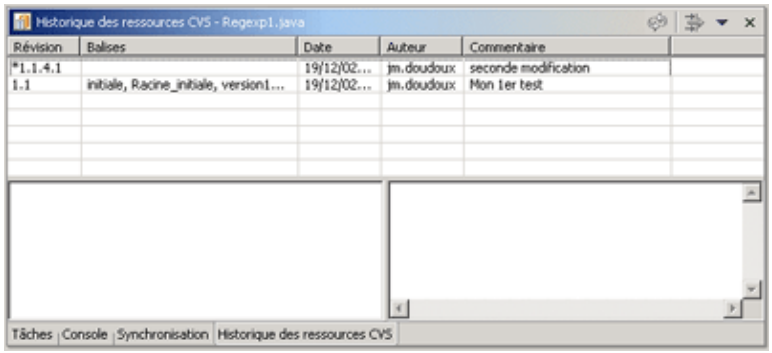
Dans cette vue, il suffit de sélectionner la ressource et d'activer l'option "Valider". Il est possible de saisir un commentaire.

La vue "Synchronisation" permet la synchronisation entre les ressources locales (dans l'espace de travail) et celles contenues dans CVS. Les trois premiers boutons permettent de préciser le sens de la synchronisation :

-  mode entrant : modifications contenues dans le référentiel à intégrer dans l'espace de travail
-  mode sortant : modifications contenues dans l'espace de travail à intégrer dans le référentiel
-  mode entrant/sortant : modifications dans l'espace de travail et le référentiel à intégrer dans l'un et l'autre

12.3.2. Gestion des révisions

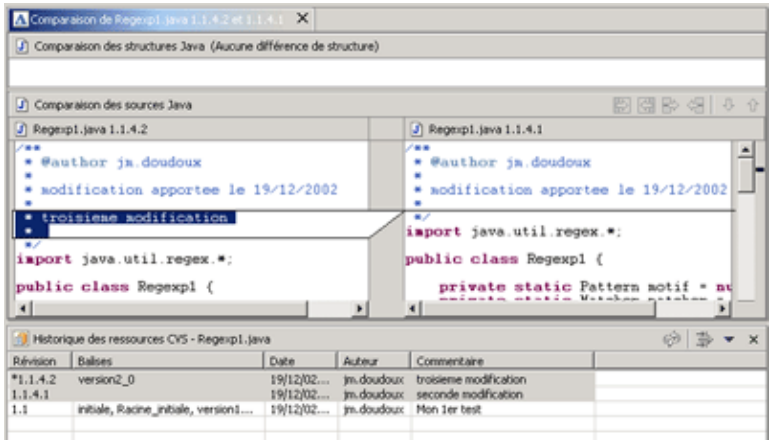
Il faut sélectionner une ressource et activer l'option "Equipe/Afficher dans l'historique des versions". La vue "Historique des ressources CVS" s'affiche en contenant les différentes révision de la ressource.



La révision courante est précédée d'une petite étoile.

Pour remplacer la ressource par celle correspondant à une autre révision, il suffit de sélectionner la révision et d'activer l'option "Obtenir une révision avec des marqueurs" du menu contextuel. Il faut ensuite cliquer sur le bouton "Ok" lors du message d'avertissement et la révision courante est modifiée.

Il est possible de comparer le contenu de deux révisions. Pour cela, il suffit de sélectionner les deux révisions en maintenant la touche Ctrl enfoncée et d'activer l'option "Comparer" du menu contextuel.

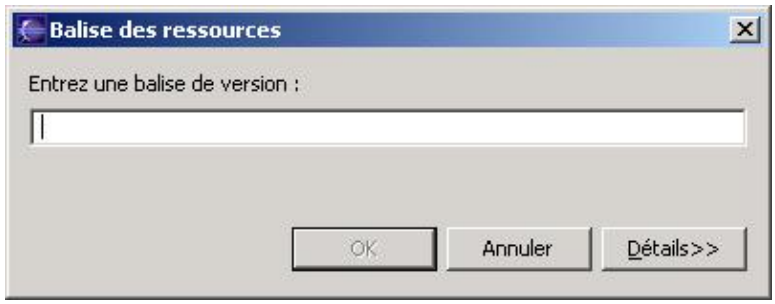


12.4. La gestion des versions d'un projet

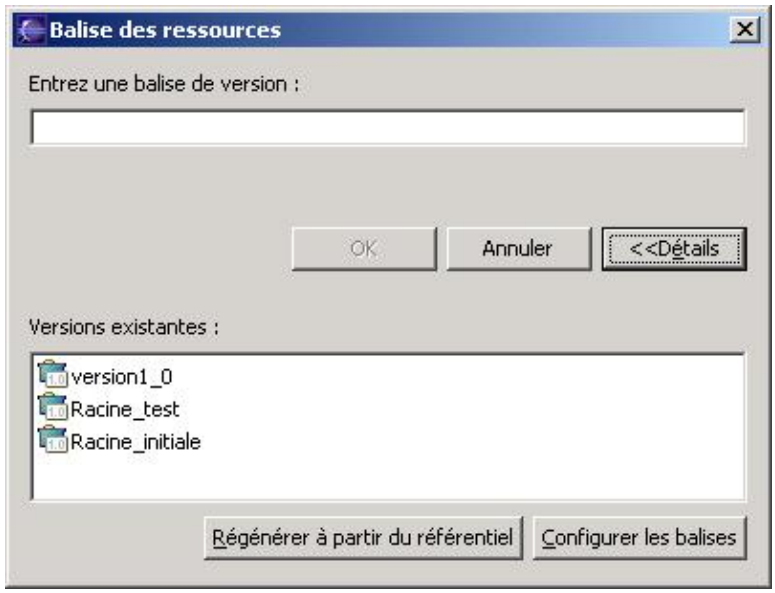
12.4.1. La création d'une version d'un projet

Il faut sélectionner le projet et activer l'option "Equipe/Baliser en tant que version" du menu contextuel.

Une boite de dialogue demande le nom de la balise



Un clic sur le bouton "Détails" permet de voir les versions existantes.



Il suffit de saisir le nom et de cliquer sur le bouton "Ok"

La version apparaît dans la vue "Référentiels CVS"

12.5. Obtenir une version dans le worskspace

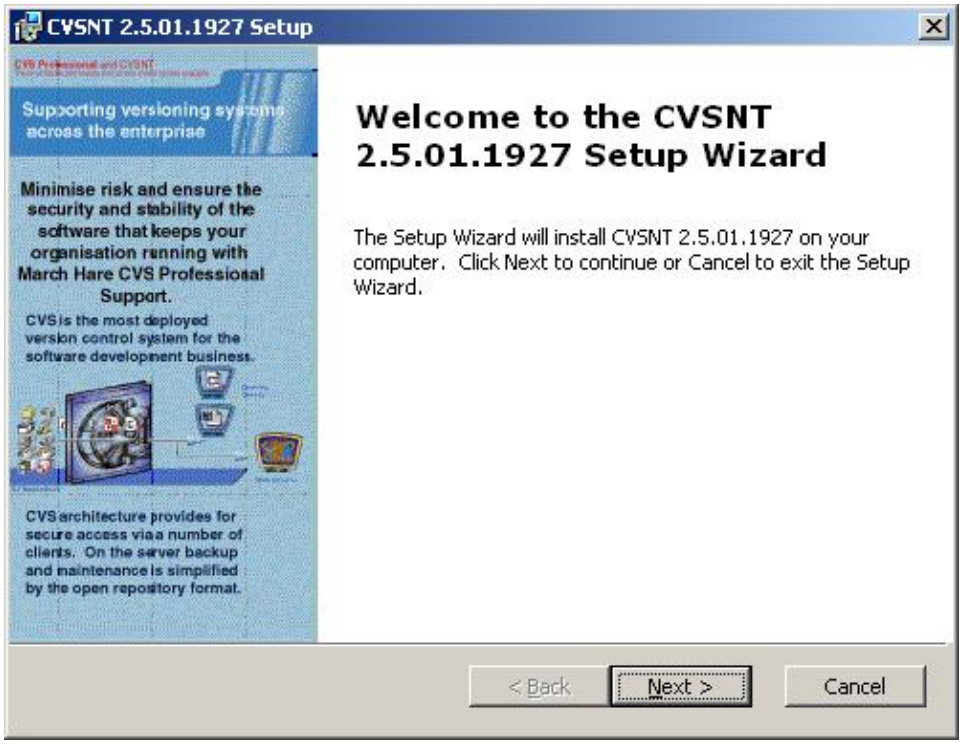
Dans la vue "Référentiels CVS", il suffit de cliquer sur la version concernée et d'activer l'option "Réserver en tant que projet" du menu contextuel.

Si le projet est déjà présent dans le workspace, un message demande la confirmation pour le remplacement.

13. CVS 2.5 et Eclipse 3.0

13.1. Installation et configuration de CVS 2.5

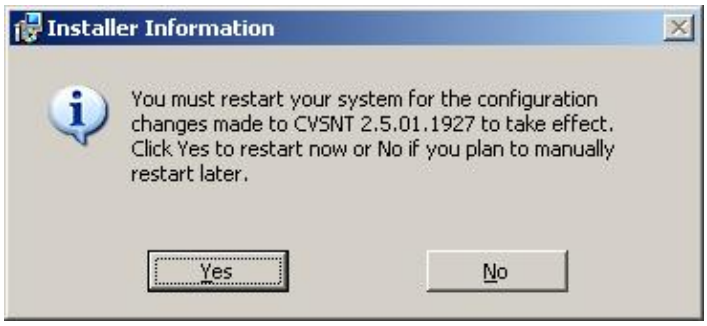
Il faut télécharger le fichier cvsnt−2.5.01.1927.msi sur le site http://www.cvsnt.com/cvspro/ et l'exécuter



Il suffit de suivre les différentes étapes de l'assistant pour réaliser l'installation

- Sur la page d'accueil, cliquez sur le bouton « Next »
- Sur la page « End−User License Agreement », sélectionnez « I accept the terms in the Licence Agreement » et cliquez sur le bouton « Next »
- Sur la page « Choose Setup Type », cliquez sur le bouton « Complete »
- Sur la page « Ready to Install », cliquez sur le bouton « Install »
- Les fichiers sont copiés sur le système
- Cliquez sur le bouton « Finish »

Une boîte de dialogue informe de la nécessite de redémarrer le système.

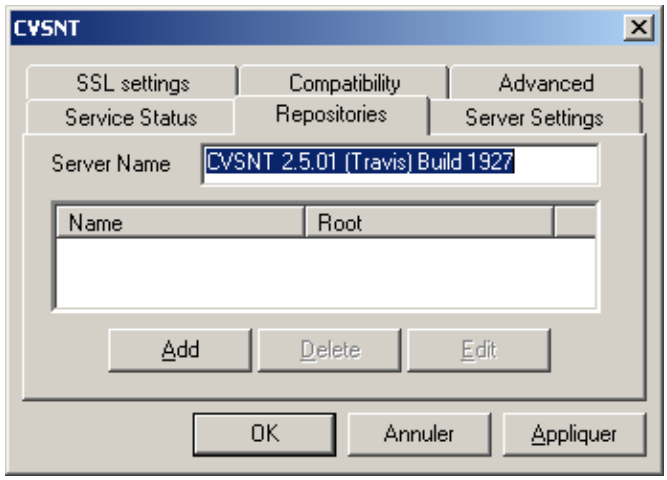


Cliquez sur le bouton « Yes » pour redémarrer le système.

Une petite icône est ajoutée dans le panneau de configuration

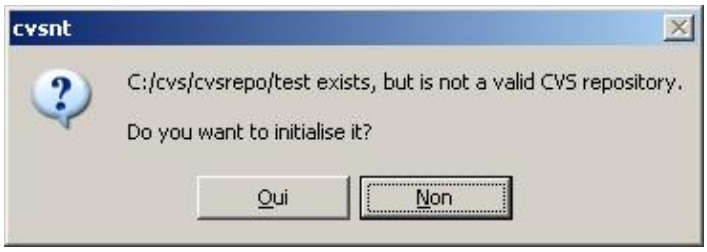
Un double clic sur cette icône permet d'accéder à la configuration de CVSNT.

Cliquez sur l'onglet « Repositories »

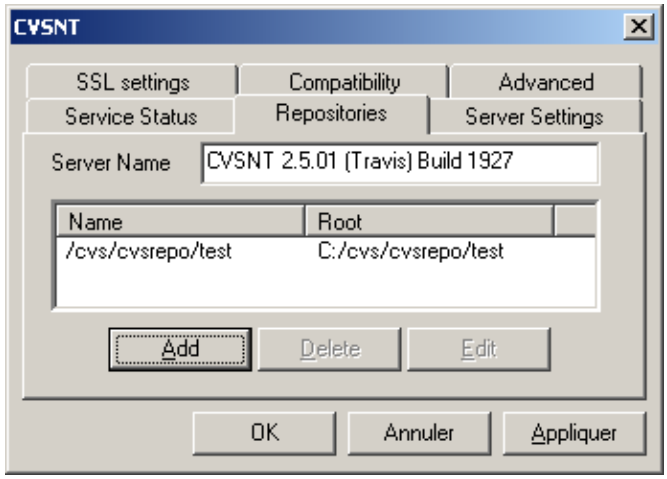


Cliquez sur le bouton « Add »

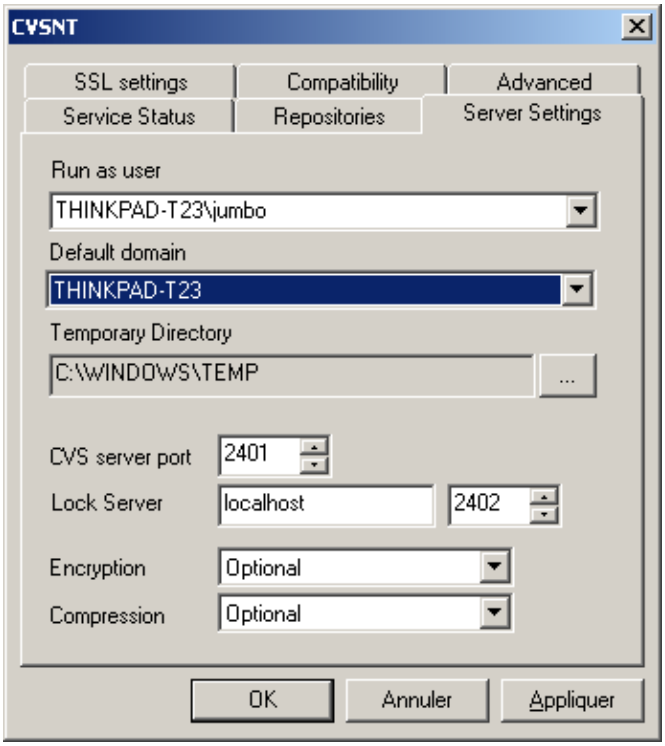
Sélectionner l'emplacement de stockage des fichiers, saisissez le nom du référentiel et la description puis cliquez sur le bouton « Ok ».



Cliquez sur le bouton « Oui » pour procéder à l'initialisation du référentiel



Sélectionnez l'onglet « Server Settings »

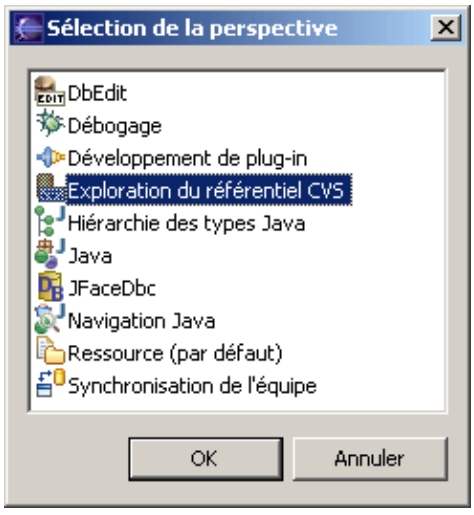


Sélectionnez les informations utiles et cliquez sur le bouton « Appliquer »

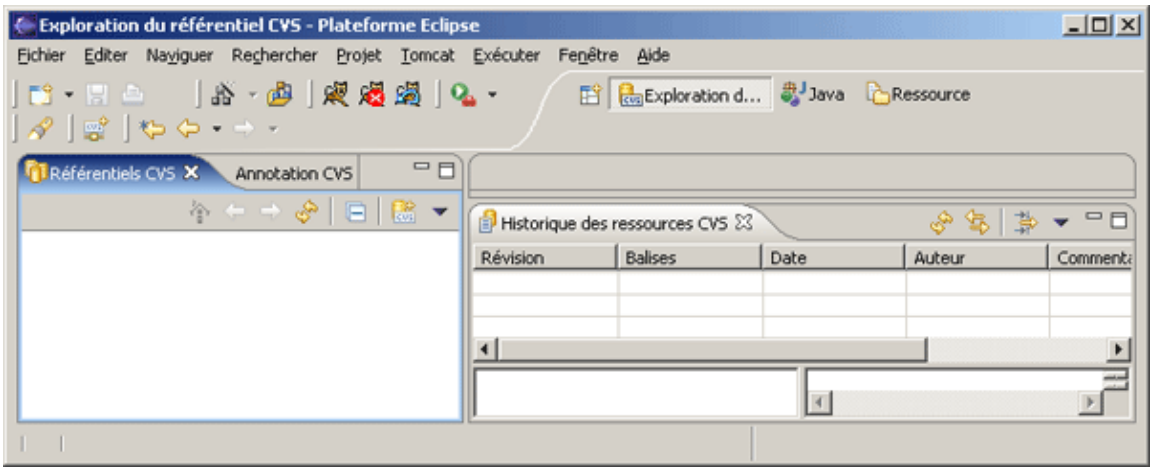
Sélectionnez l'onglet « Service Status » et démarrez les deux services proposés ci ceux−ci sont arrêtés.

13.2. La perspective « Exploration du référentiel CVS »

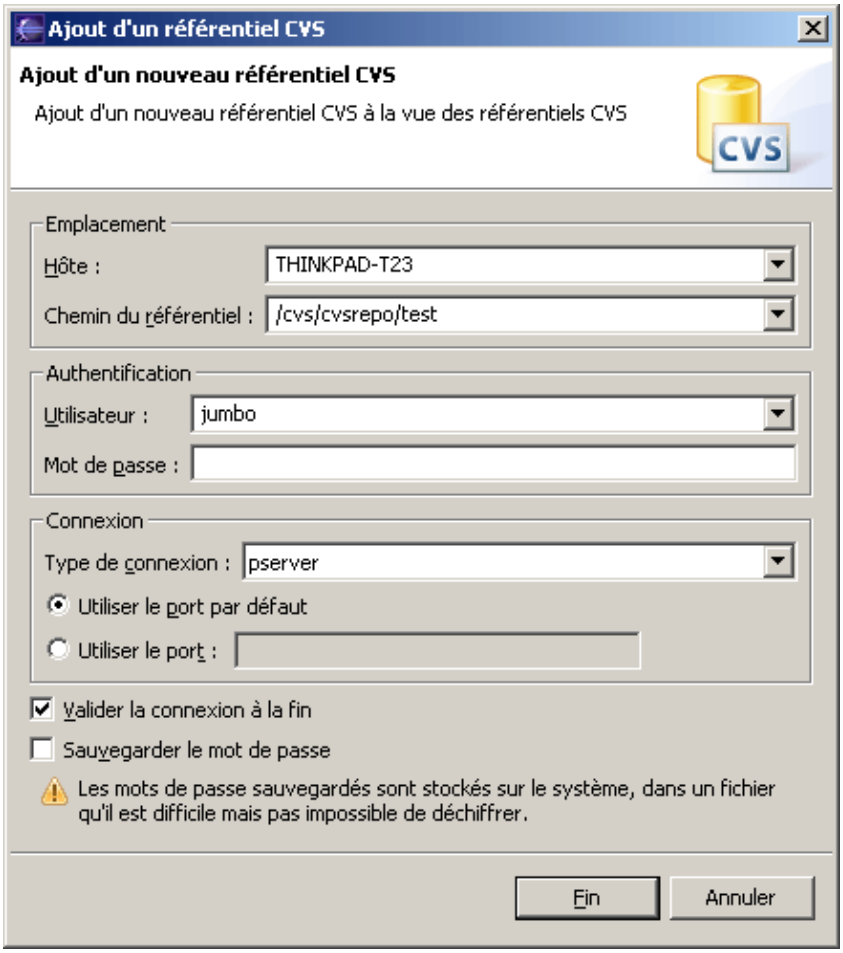
Pour ouvrir une perspective, il faut sélectionnez le menu « Fenêtre/Ouvrir la perspective/Autre ».



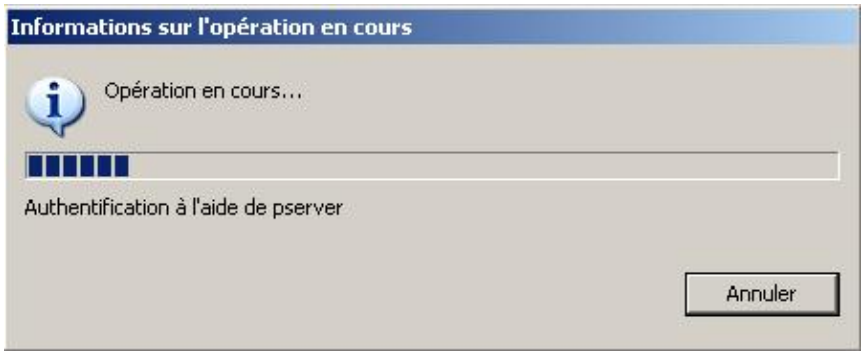
Sélectionnez l'option « Exploration du référentiel CVS » et cliquez sur le bouton « OK » pour ouvrir la perspective.



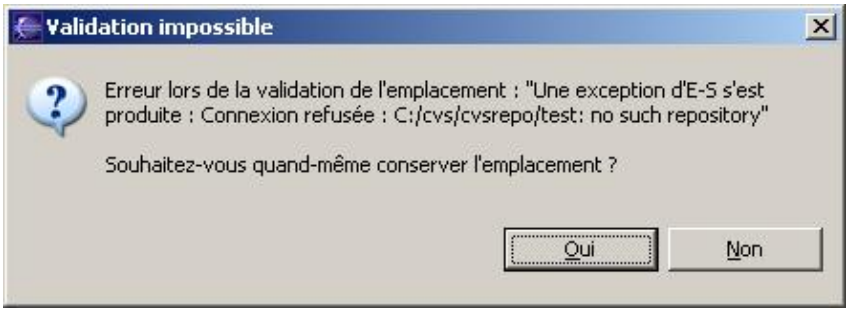
Dans la vue « Référentiels CVS », sélectionnez l'option « Créer/Emplacement du référentiel » du menu contextuel.



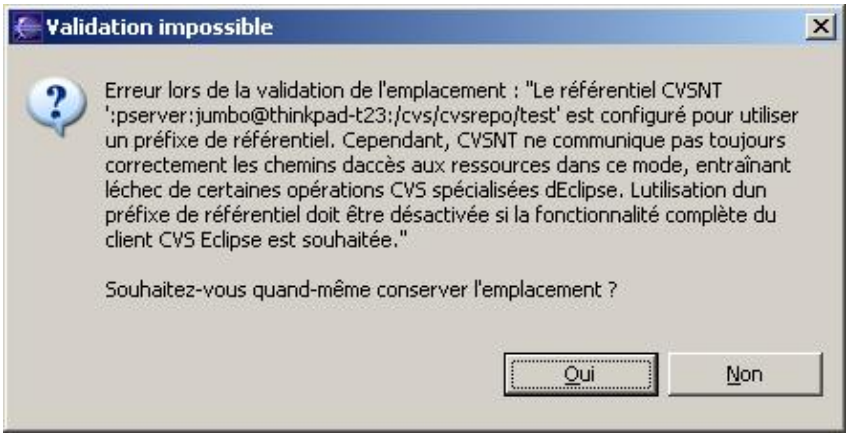
Saisissez le nom de l'hôte, le chemin du référentiel, nom de l'utilisateur et son mot de passe pour se connecter à CVS, le type de connexion. Cliquez sur le bouton « Fin » pour demander la connexion



Un message d'erreur informe de l'inexistante du référentiel si ce dernier n'est pas trouvé.



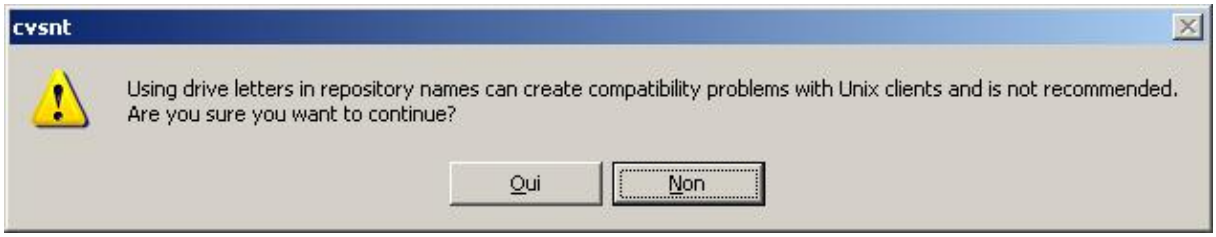
Comme CVS est configuré pour utiliser un préfixe, un message d'erreur est affiché.



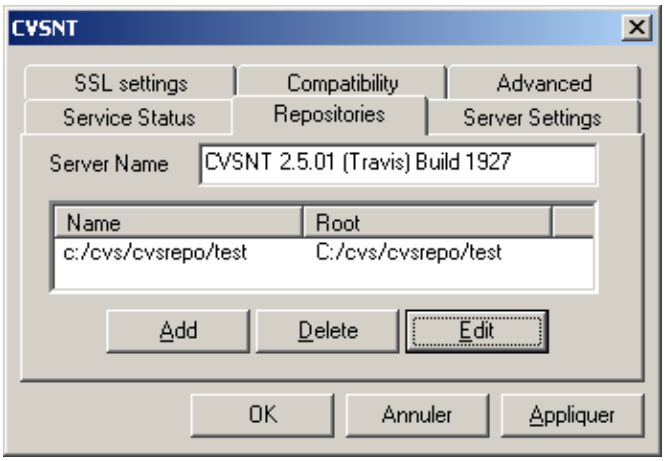
Cliquez sur le bouton « Non »

Il est alors nécessaire de modifier le nom du référentiel dans la configuration de CVSNT en précisant le chemin complet du référentiel dans son nom.

Cliquez sur le bouton « OK »

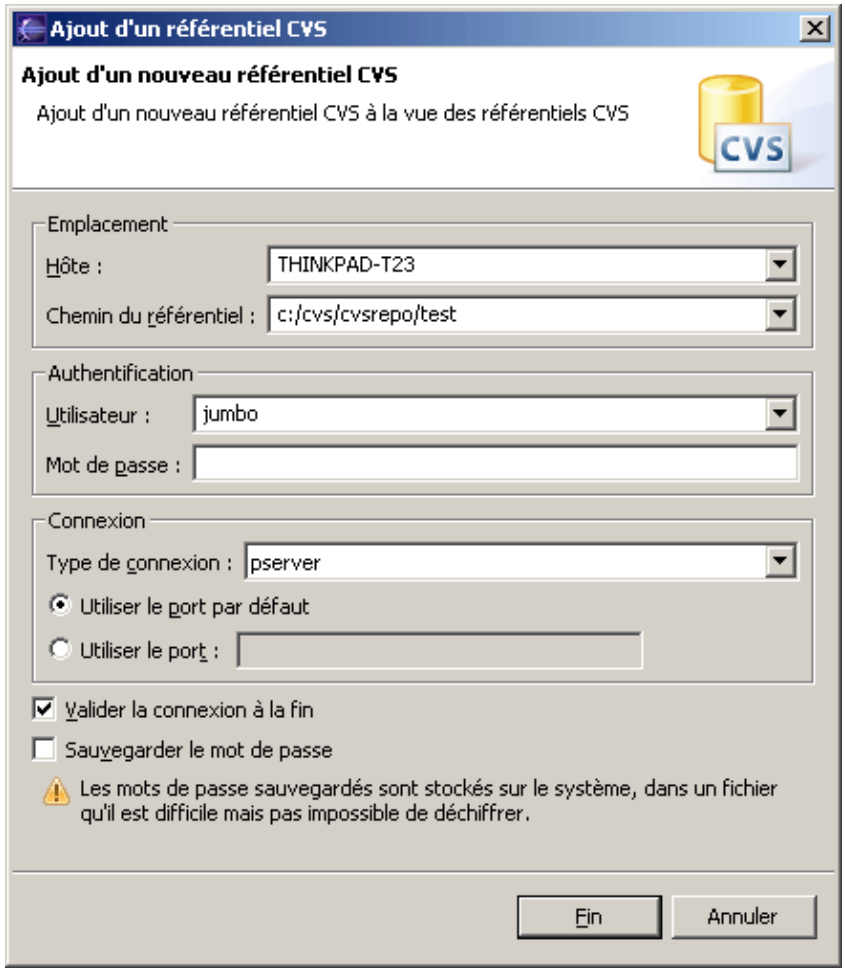


Cliquez sur le bouton « Oui »



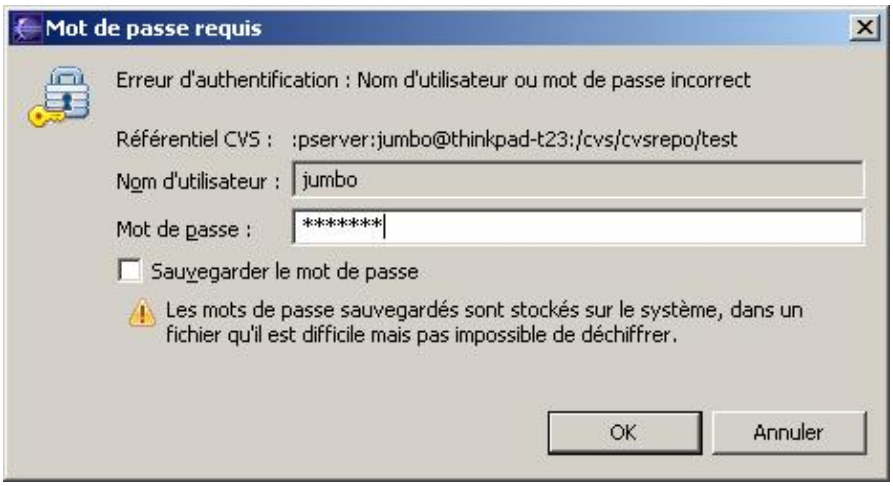
Cliquez sur le bouton « Appliquer » puis sur le bouton « OK »

Il suffit alors de modifier les paramètres du référentiel sous Eclipse

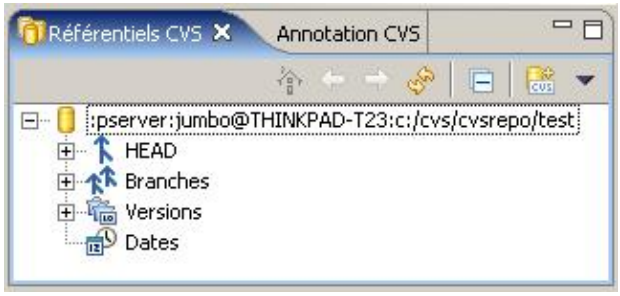


Le chemin du référentiel doit contenir le chemin complet. Cliquez sur le bouton « Fin »

Si le mot de passe saisi est incorrect, une boîte de dialogue permet de le ressaisir.



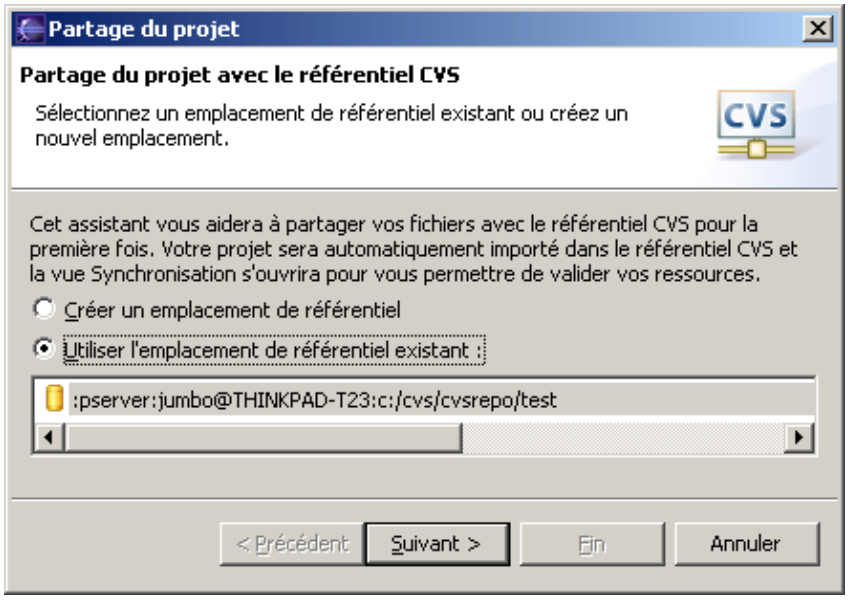
Le nouveau chemin de référentiel est ajouté dans la vue « Référentiels CVS ».



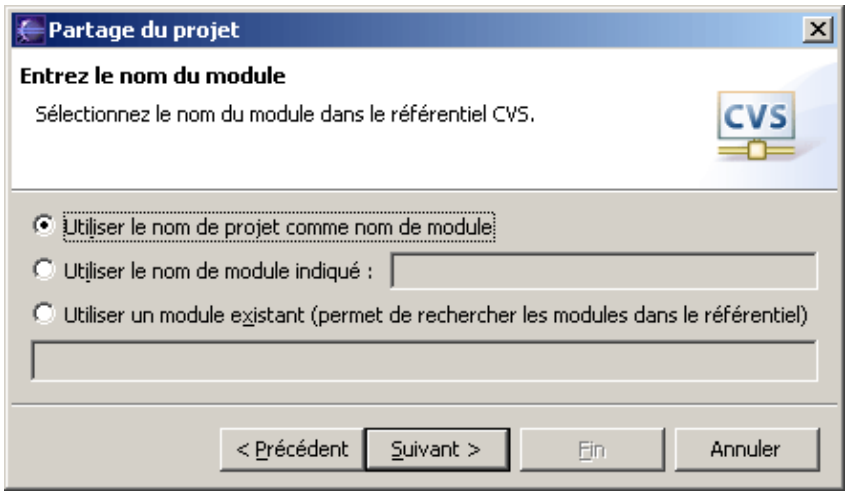
Un chemin de référentiel n'est pas une connexion mais des informations nécessaires à cette connexion réalisée lorsque que les traitements en ont besoin.

13.3. Ajouter un projet au référentiel

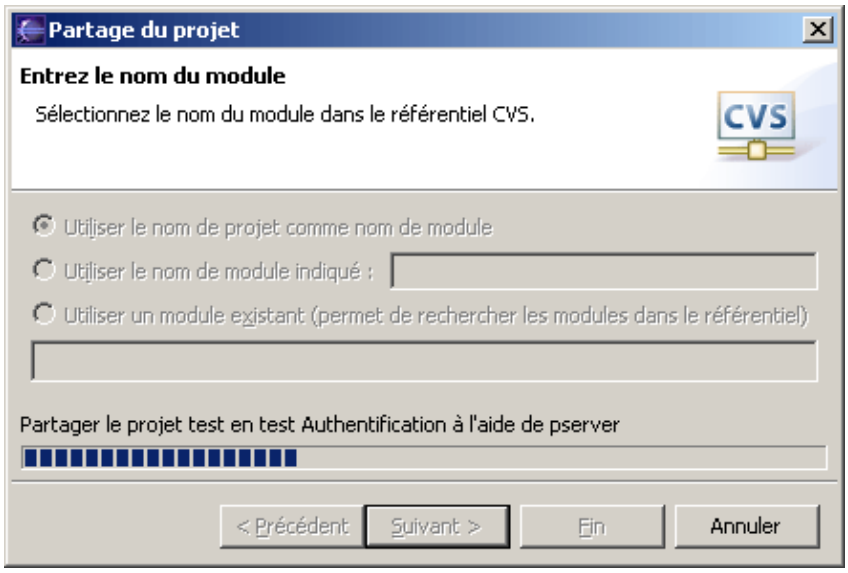
Pour ajouter un projet au référentiel, il faut utiliser l'option « Equipe/Partager le projet ... » du menu contextuel du projet



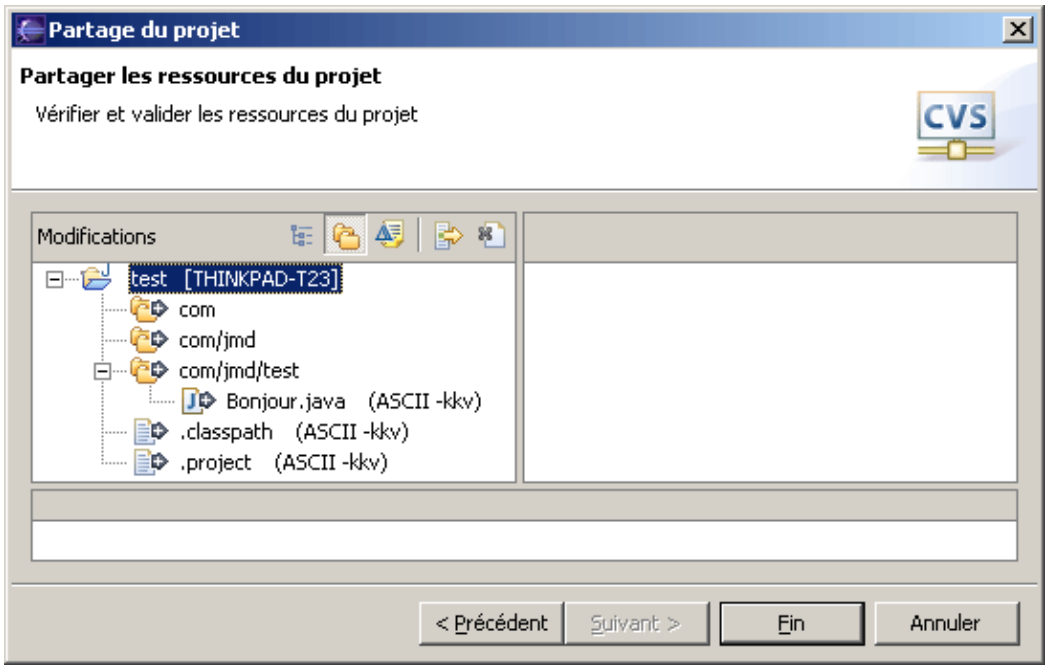
Sélectionnez « Utiliser l'emplacement de référentiel existant » et cliquez sur le bouton « Suivant »



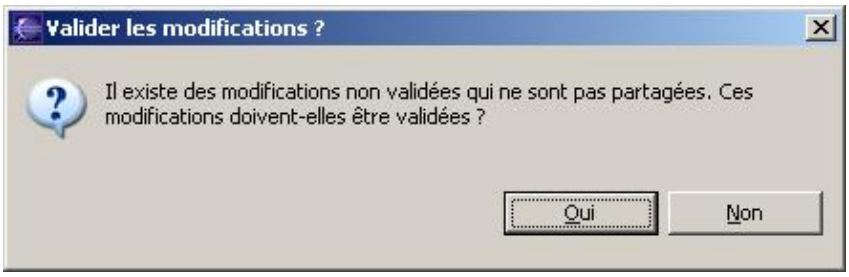
Sélectionnez « Utiliser le nom de projet comme nom de module » et cliquez sur le bouton « Suivant »



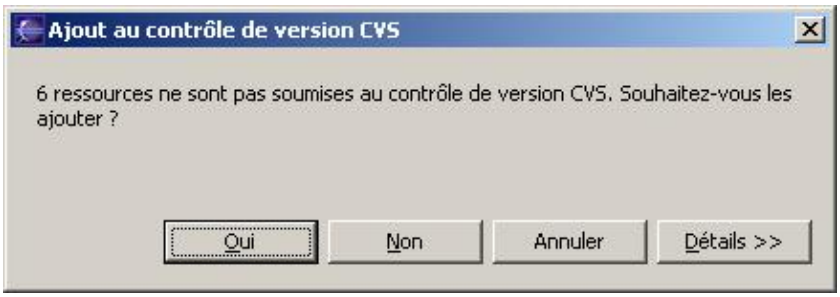
La page suivante de l'assistant permet de voir les fichiers qui seront ajoutés au référentiel.



Cliquez sur le bouton « Fin »

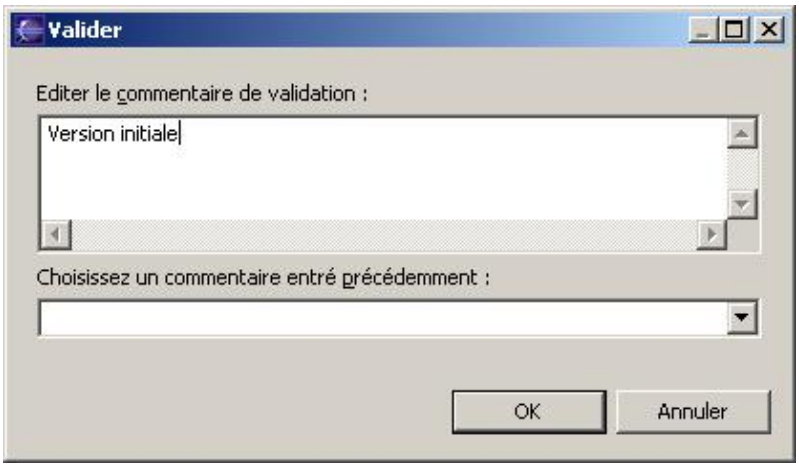


Cliquez sur le bouton « Oui »

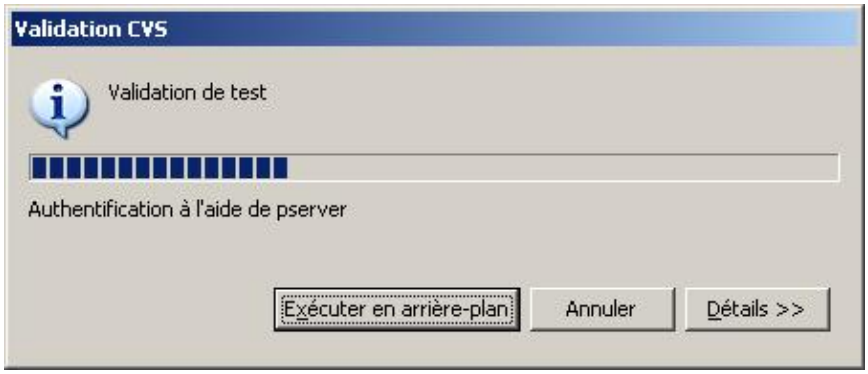


Cliquez sur le bouton « Détails » pour obtenir la liste des ressources à ajouter dans le référentiel.

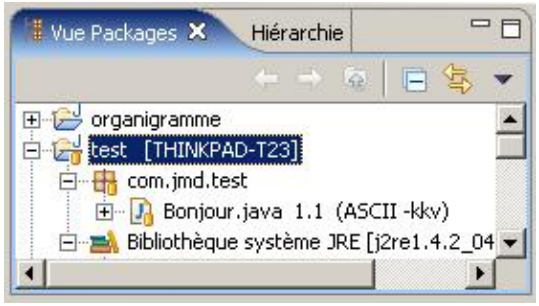
Cliquez sur le bouton « Oui »



Saisissez le commentaire et cliquez sur le bouton « OK »



Une boîte de dialogue affiche la progression des traitements.

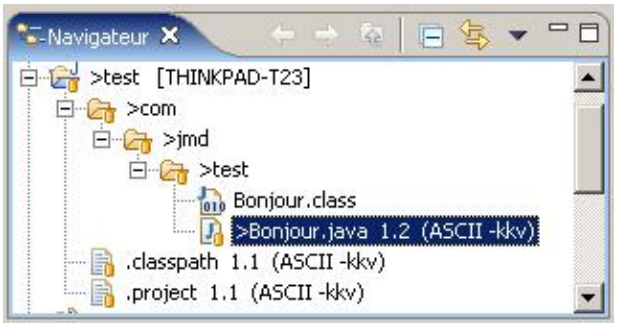


Les projets qui sont connectés à un référentiel sont identifiables grâce à plusieurs informations :

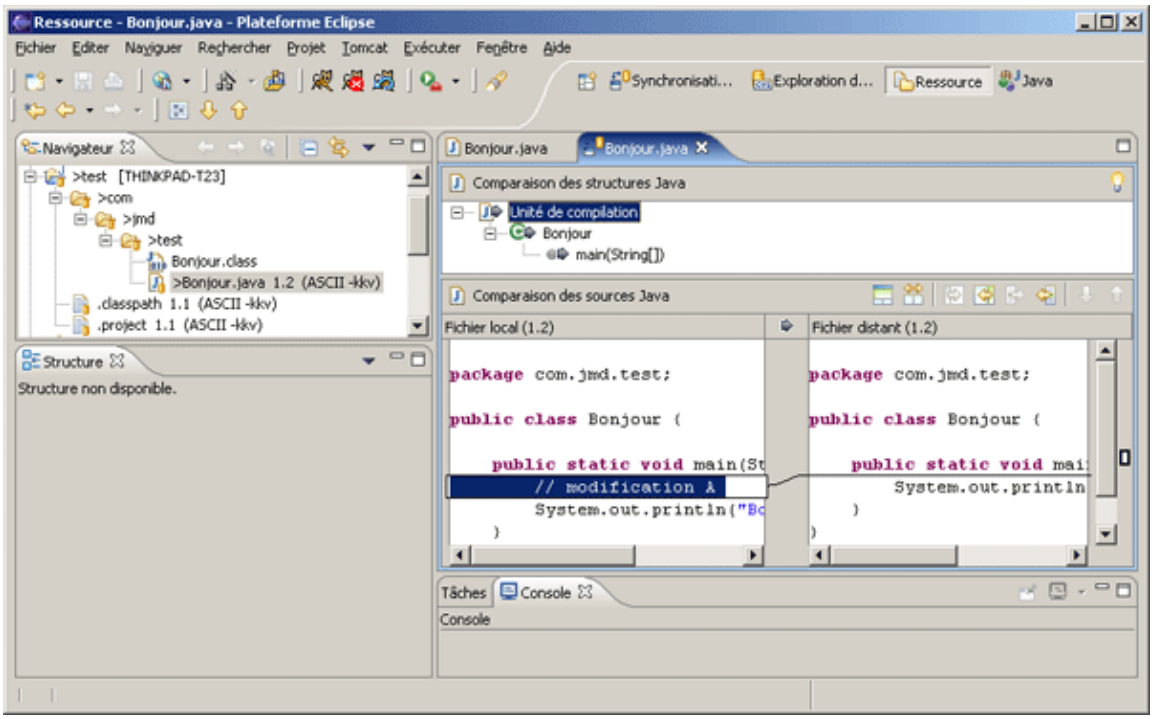
- le projet et les éléments inclus dans le référentiel possèdent une icône avec un petit cylindre jaune
- le nom du projet est suivi du nom du référentiel entre crochets
- le nom des éléments du projet sont suivis de la version et du type de fichiers entre parenthèses

13.4. Reporter des modifications dans le référentiel

Dès qu'un élément inclus dans le référentiel est modifié, ce dernier et le nom de l'ensemble de ces éléments pères sont précédés d'un caractère « > » permettant de rapidement visualiser les fichiers modifiés.

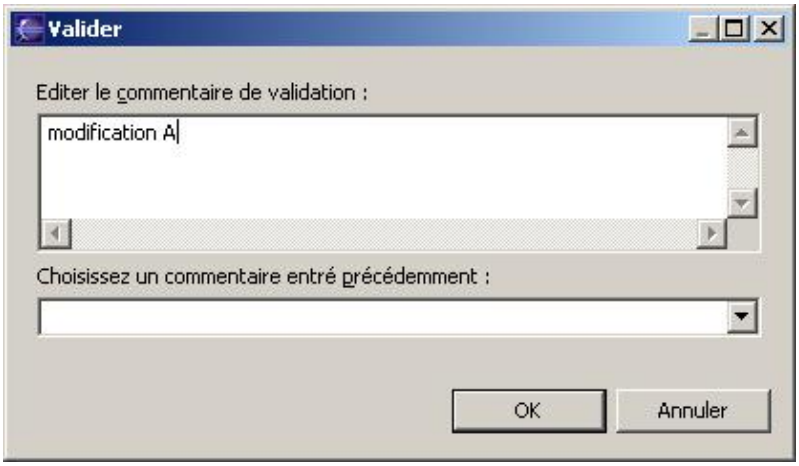


Pour reporter les modifications, il est possible de le faire au niveau de chaque éléments ou au niveau du projet en utilisant l'option « Equipe/Synchroniser avec le référentiel » du menu contextuel.



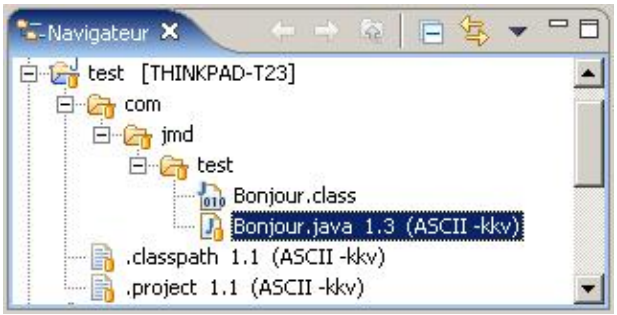
La vue affiche les différences entre la version locale et la version contenue dans le référentiel.

Pour reporter les modifications dans le référentiel, il faut utiliser l'option « Equipe/Valider » de l'élément comparé.



Une boîte de dialogue permet de saisir le commentaire de validation. Cliquez sur le bouton « OK ».

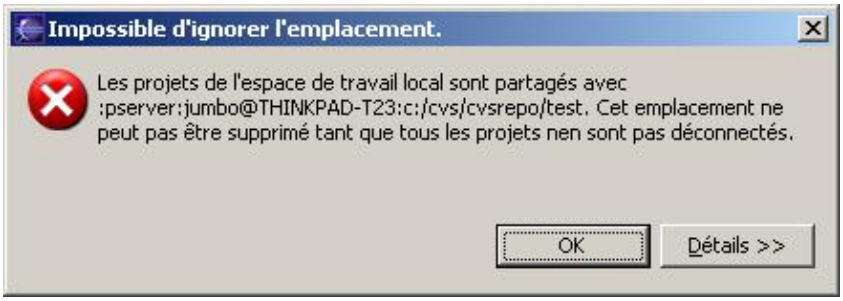
Comme la version locale et la version dans le référentiel sont identiques, les caractères « > » disparaissent.



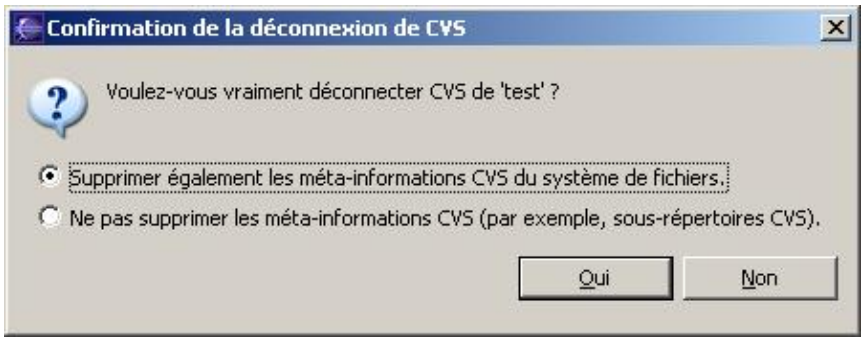
13.5. Déconnecter un projet du référentiel

Pour supprimer un référentiel, il faut sélectionner l'option « Ignorer l'emplacement » du menu contextuel

Il est obligatoire qu'aucun projet ne soit connecté sur ce dernier sinon un message d'erreur empêche l'opération.



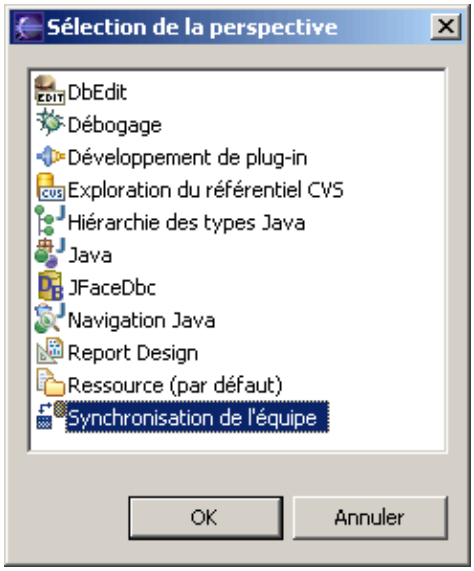
Pour déconnecter un projet, il faut utiliser l'option « Equipe/Déconnecter ... » du menu contextuel du projet.



Cliquez sur le bouton « Oui »

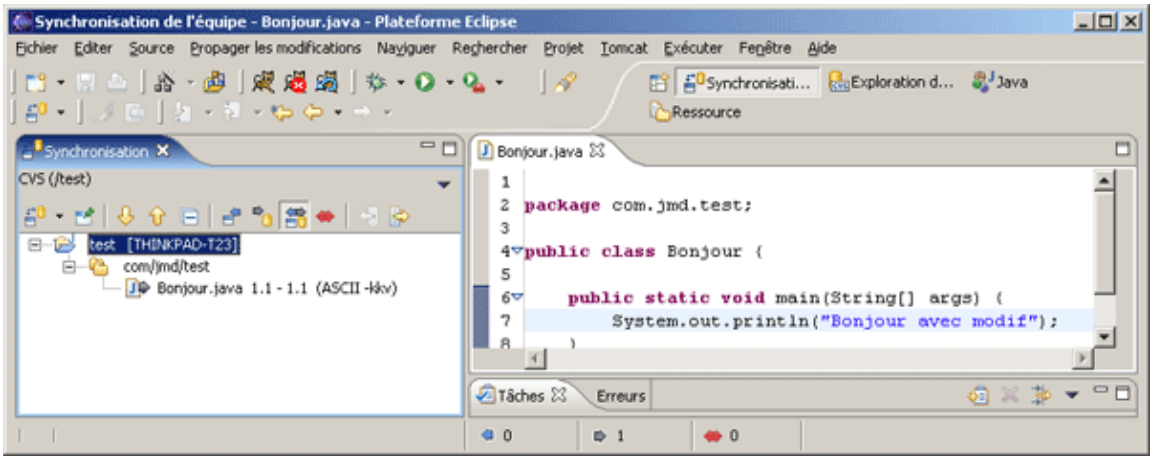
13.6. La perspective Synchronisation de l'équipe

Pour ouvrir cette perspective, il faut utiliser l'option « Fenêtre/Ouvrir la perspective/Autre » et sélectionner l'option « Synchronisation de l'équipe »



Cette perspective est aussi utilisée lors de l'utilisation de l'option « Equipe/Synchroniser avec le référentiel ... »

La vue « Synchronisation » permet de synchroniser les sources contenues dans l'espace de travail et le référentiel de CVS.



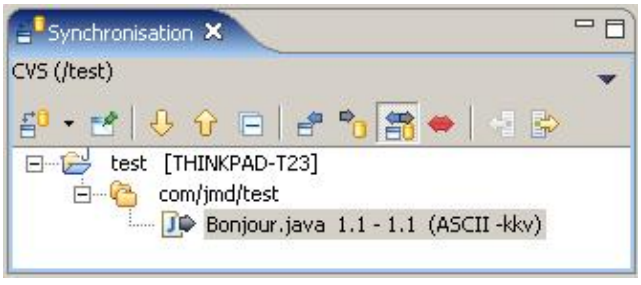
Les modifications sont signalées avec de petites icônes :

	Pour les modifications contenues dans le référentiel à intégrer dans l'espace de travail
	Pour les modifications contenues dans l'espace de travail à intégrer dans le référentiel
	Pour signaler des modifications effectuées dans l'espace de travail et dans le référentiel

La vue synchronisation propose plusieurs modes pour faciliter la synchronisation :

	Mode entrant : des modifications sont présentes dans le référentiel mais pas encore en local
	Mode sortant : modifications locales qui ne sont pas encore enregistrées dans le référentiel
	Mode entrant/sortant : utilisation des deux modes précédents simultanément
	Mode « en conflit » : des modifications locales existent et des modifications différentes sont présentes dans le référentiel

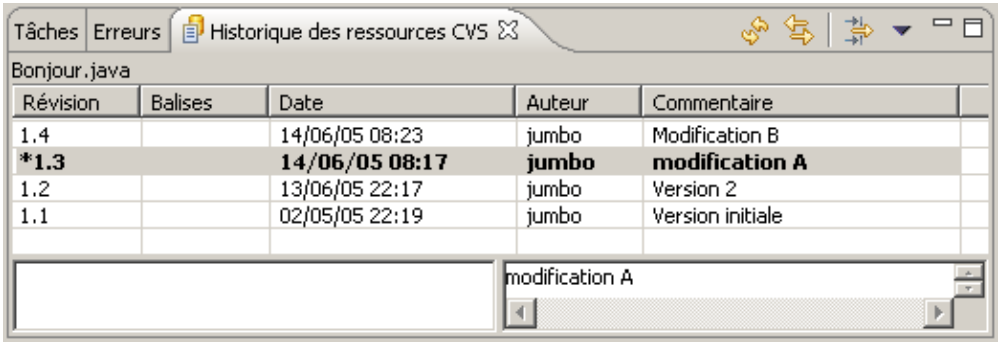
Les modifications selon le mode courant sont affichées :



Si le mode courant ne contient aucune modification, il propose de passer dans un autre mode.

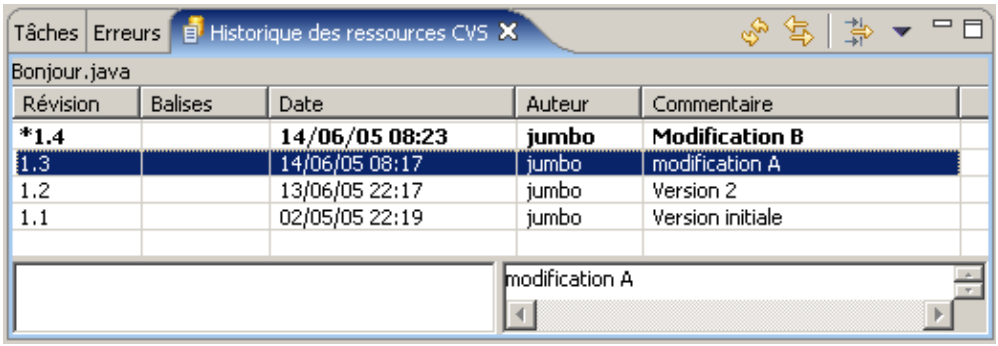
Pour connaître l'historique des versions, il faut utiliser dans la vue « Synchronisation » l'option « Afficher dans l'historique des versions » du menu contextuel d'un élément.

La vue « Historique des ressources CVS » s'affiche.



La version correspondant à celle dans l'espace de travail est signalée avec une petite étoile.

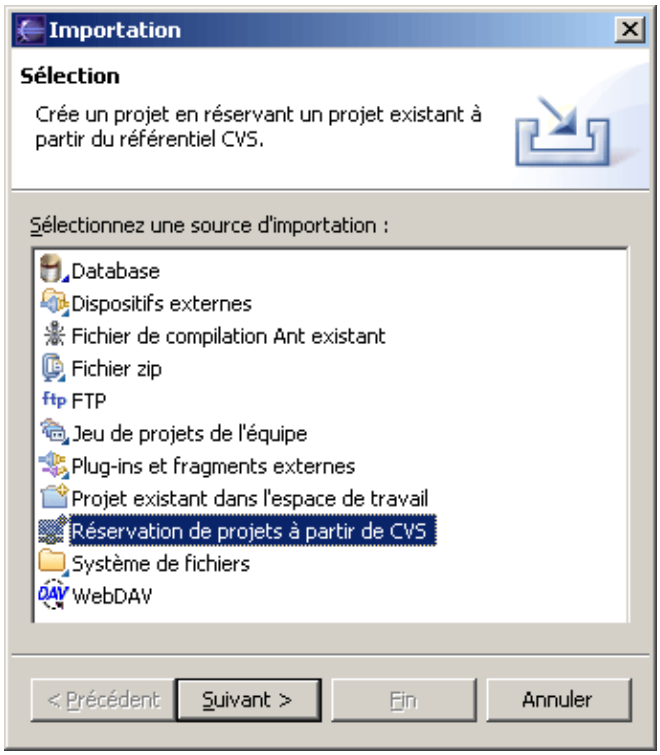
A partir de cette vue, il est possible de récupérer dans l'espace de travail n'importe quelle version en la sélectionnant et en utilisant l'option « Obtenir le contenu » du menu contextuel.



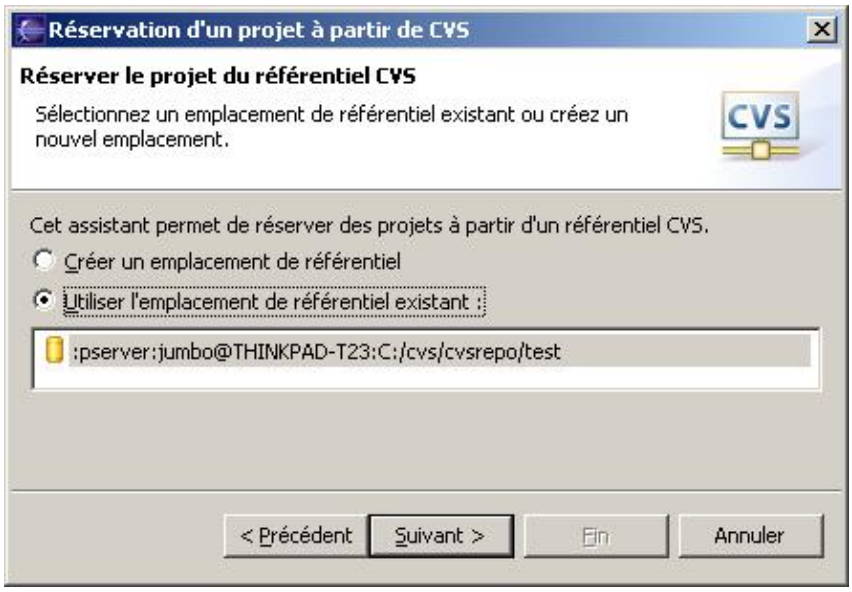
13.7. Importation d'un projet à partir de CVS

Il est possible à n'importe quel utilisateur pouvant se connecter au référentiel d'importer un projet dans son espace de travail pour le modifier.

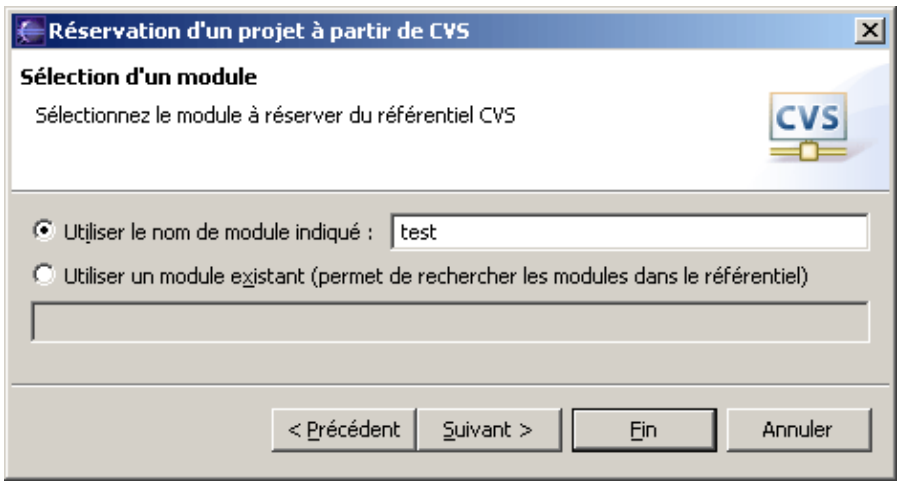
Il faut utiliser l'option « Fichier/Importer ...» du menu principal.



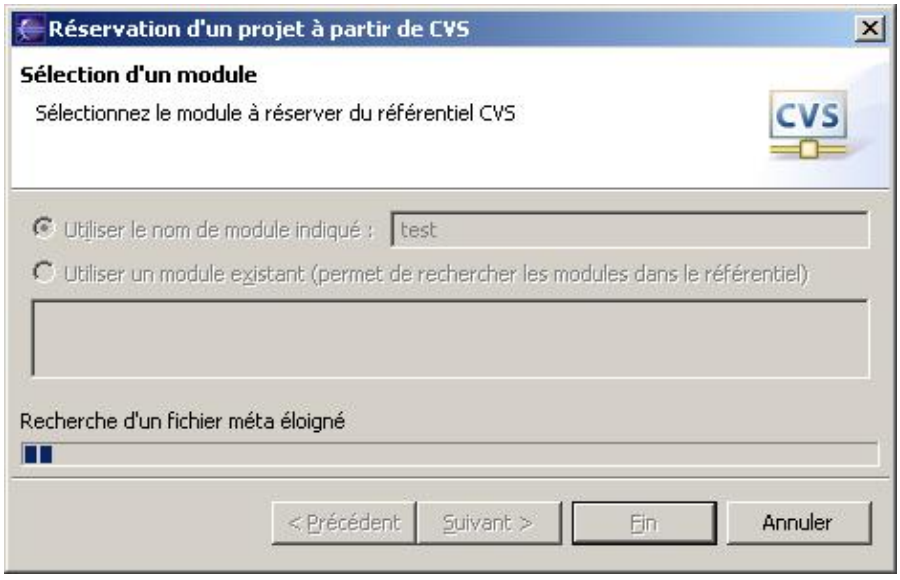
Sélectionnez « Réservation de projets à partir de CVS » puis cliquez sur le bouton « Suivant ».



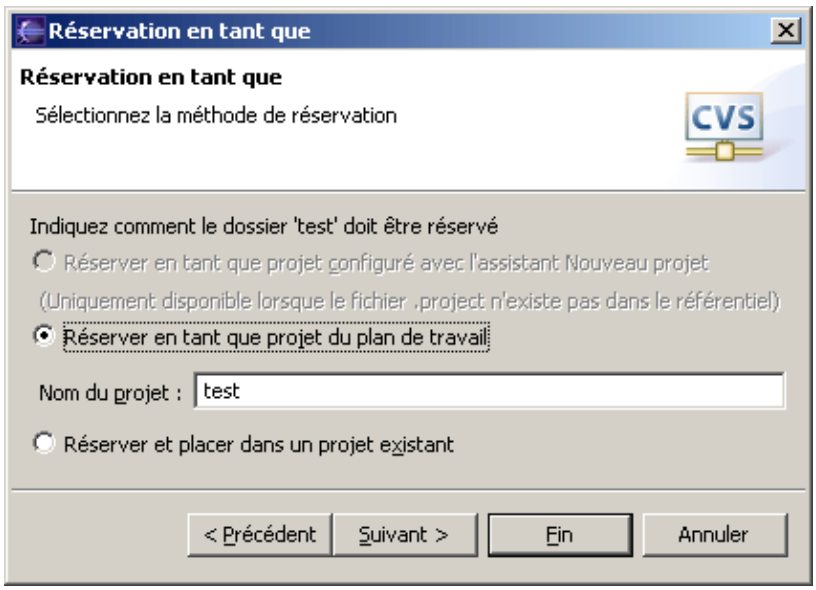
Le référentiel est sélectionné par défaut, cliquez sur le bouton « Suivant ».



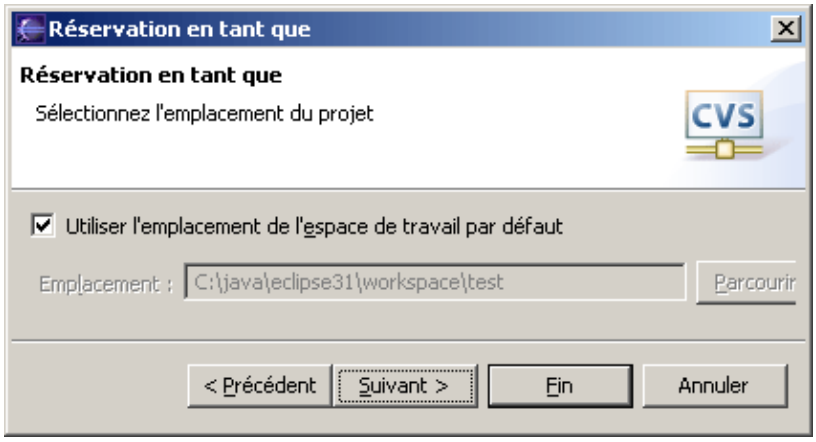
Saisir le nom du module et cliquez sur le bouton « Suivant ». Il est aussi possible de cocher la case « Utiliser un module existant » pour demander à l'assistant de rechercher les modules inclus dans le référentiel et de permettre de sélectionner ceux concernés.



L'assistant se connecte au référentiel



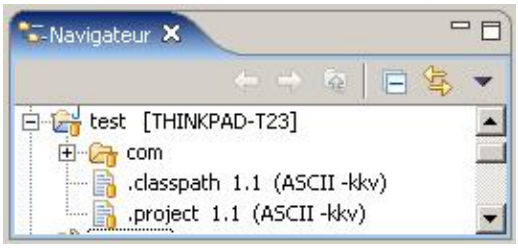
Cliquez sur le bouton « Suivant »



Cliquez sur le bouton « Suivant »

Sélectionnez la balise HEAD et cliquez sur le bouton « Fin »

Le projet est importé dans l'espace de travail.



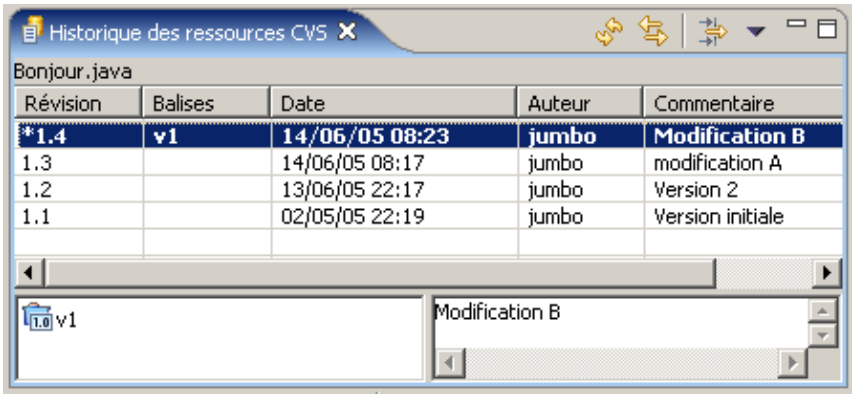
13.8. Versionner un projet

Il faut utiliser l'option « Equipe/Baliser en tant que version... » dans la vue « Navigateur » ou « Packages »

Si toutes les modifications ne sont pas validées, un message de confirmation est demandé à l'utilisateur.

Une boîte de dialogue permet de saisir le nom de la balise de version (ce nom ne doit pas contenir d'espaces)

Il suffit de saisir le nom de la balise et de cliquer sur le bouton « OK »



14. La gestion de la plate−forme

Eclipse est conçu pour être un outil modulaire. De nombreux modules (plug−ins) sont fournis avec Eclipse mais il est aussi très facile d'en ajouter d'autres développés par la communauté ou des sociétés commerciales.

L'installation d'un plug−in est souvent très simple car elle consiste essentiellement à décompresser l'archive qui contient le plug−in pour que le contenu soit insérer dans le répertoire "plugins" où est installé Eclipse.

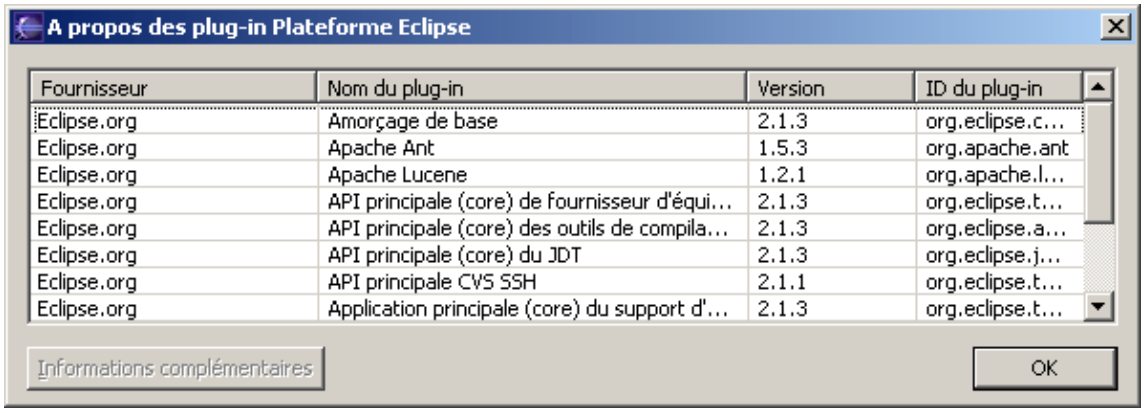
Eclipse 2.0 propose une fonctionnalité qui permet d'installer ou de mettre à jour des plug−ins via le web.

14.1. Informations sur les plug−ins installés

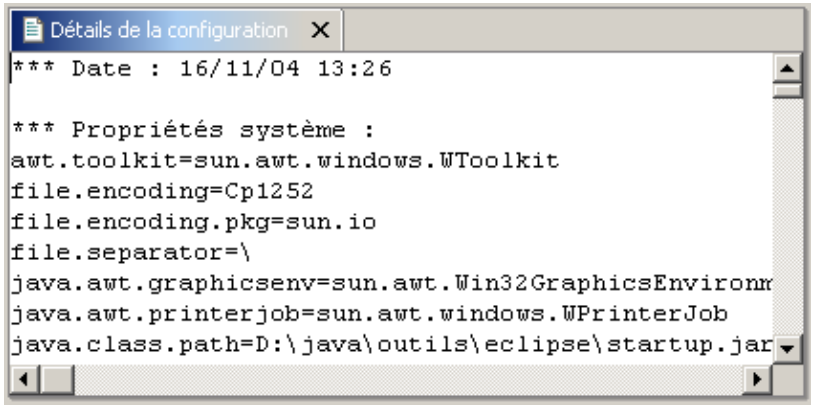
Pour obtenir des informations sur la plate−forme, il faut utiliser le menu "Aide / A propos de plateforme Eclipse".

Cette boîte de dialogue affiche des informations générales sur Eclipse et permet de sélectionner le type d'informations supplémentaires souhaitées.

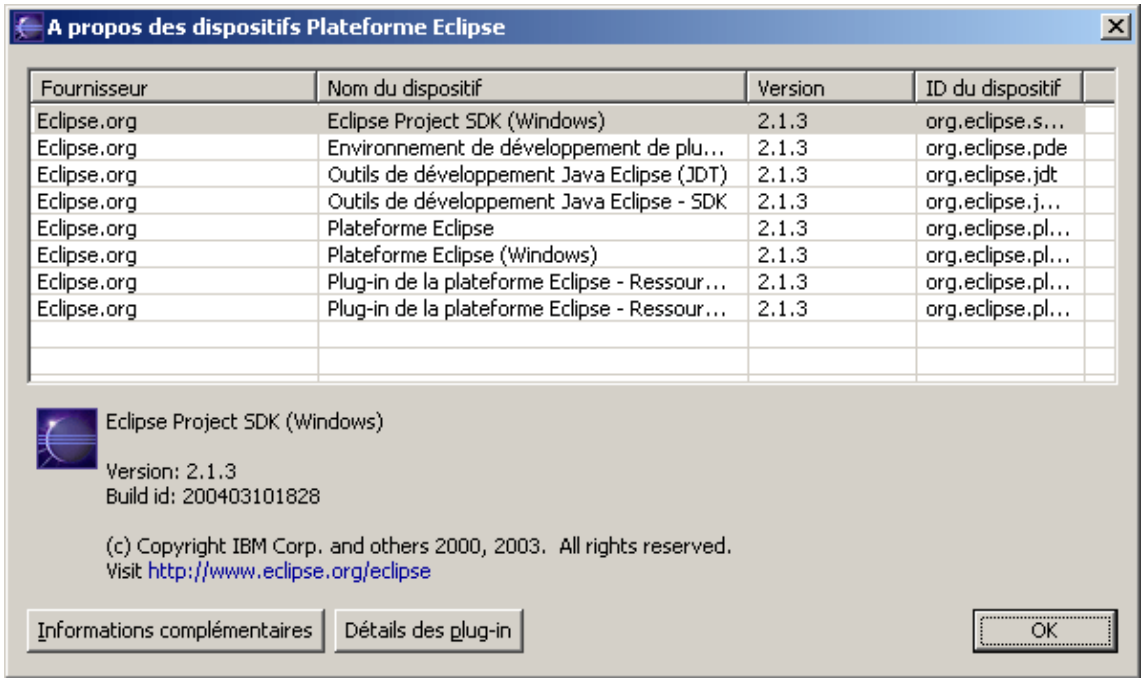
Cliquez sur le bouton "Détails des plug−ins" pour obtenir une liste de tous les plug−ins installés.



Un clic sur le bouton "Détails de la configuration" affiche un fichier dans la vue éditeur contenant des informations détaillées sur la configuration en cours d'utilisation.



Un clic sur le bouton "Détails des dispositifs", permet d'avoir des informations sur les éléments de base d'Eclipse.



14.2. Installation du plug−in JadClipse sous Eclipse 1.0

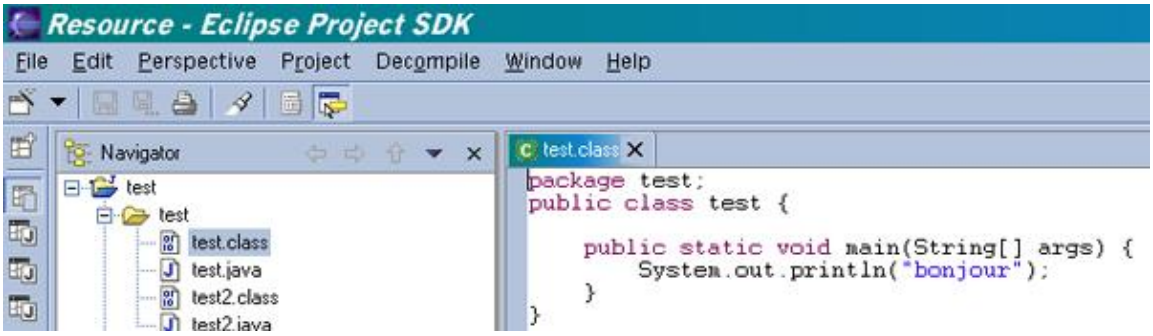
Jadclipse est un plug−in qui permet d'utiliser le décompilateur Jad dans Eclipse.

Il suffit de télécharger le fichier zip contenant le module à l'url http://sourceforge.net/projects/jadclipse/. Il faut aussi avoir installer l'outil Jad.

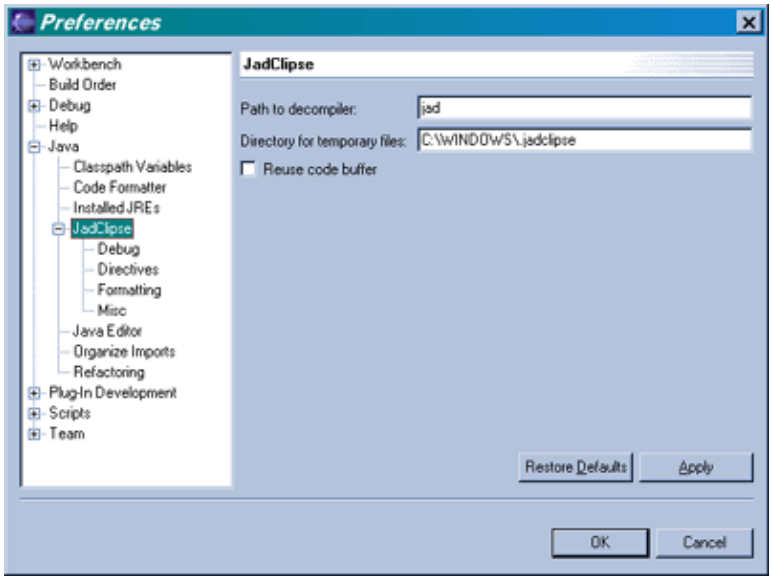
Pour installer le plug−in, il suffit simplement de décompresser le fichier dans le répertoire "plugins" du répertoire principal d'Eclipse.

Pour que le module soit pris en compte et configurer automatiquement, il suffit de lancer ou relancer Eclipse.

A l'ouverture d'un fichier .class, le menu « Decompile » apparaît dans le menu principal. Il faut cocher l'option « Engage Jadclipse ». Chaque fichier .class ouvert est automatiquement décompilé et le code source est affiché dans l'éditeur.



Le plug−in est parfaitement intégré dans Eclipse : les options du plug−in sont modifiables dans l'arborescence "Java / Jadclipse" des préférences (menu "Window / Preferences").



14.3. La mise à jour des plug−ins avec Eclipse 2.0

Eclipse 2.0 propose des fonctionnalités pour permettre la mise à jour ou l'installation de plug−ins en utilisant le web.

14.3.1. La perspective « Installation / Mise à jour »

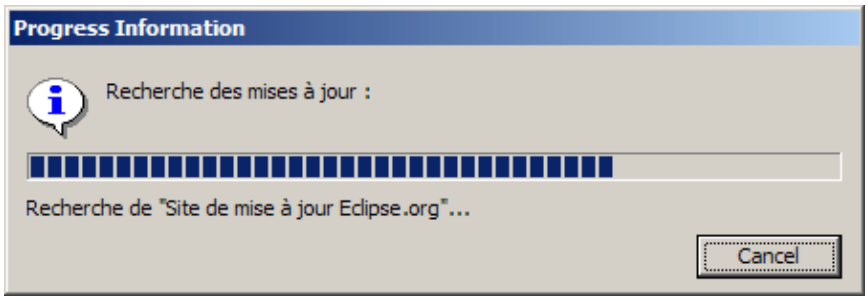
La perspective « Installation / Mise à jour » permet la gestion des mises à jour. Pour l'afficher, il suffit de sélectionner cette perspective ou de sélectionner l'option « Mise à jour des logiciels / Gestionnaire des mises à jour » du menu "Aide".

La vue "Mises à jour des dispositifs" recence la liste des sources d'où peuvent être téléchargées les mises à jour.

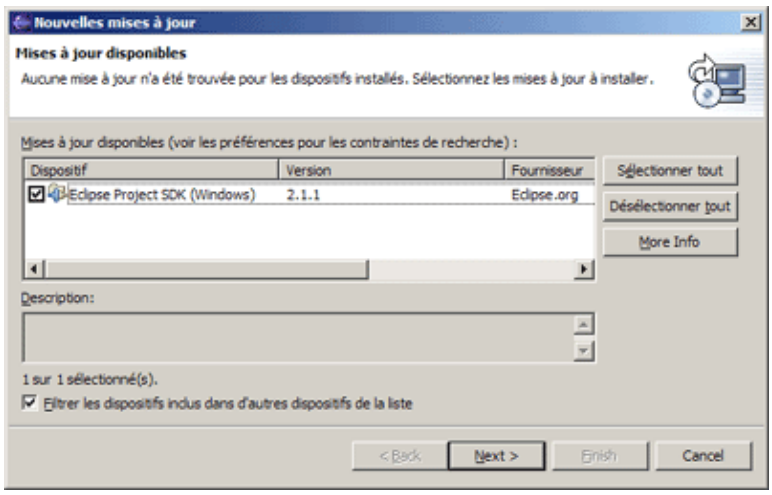
14.3.2. Recherche et installation des mises à jour

Il faut utiliser l'option "Mise à jour des logiciels" du menu "Aide".

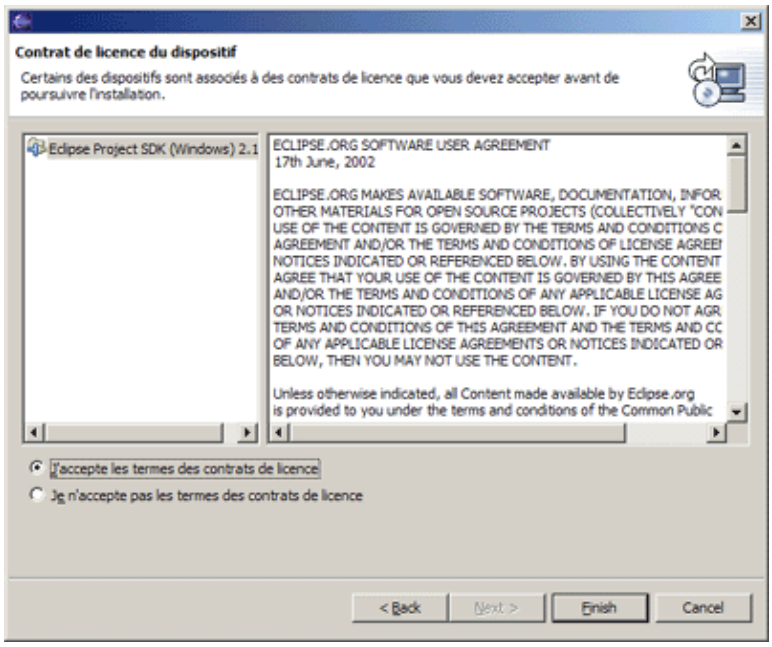
Un assistant recherche les mises à jour pour les plug−ins dont le site est enregistré dans Eclipse.



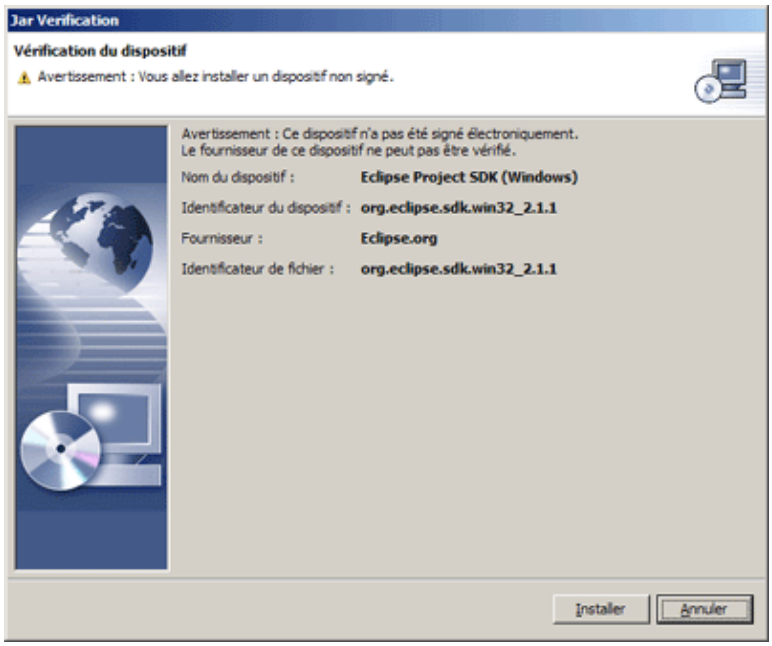
L'exemple ci dessous illustre la mise à jour d'une version 2.1 d'Eclipse avec la version 2.1.1.



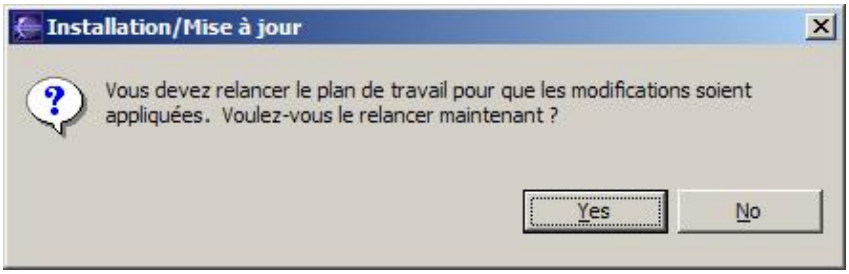
Cliquez sur le bouton "Next"



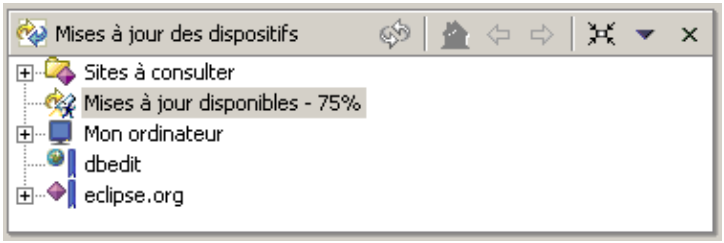
Lisez la licence et si vous l'acceptez, Cliquez sur le bouton "J'accepte les termes des contrats de licence" puis sur le bouton "Finish"



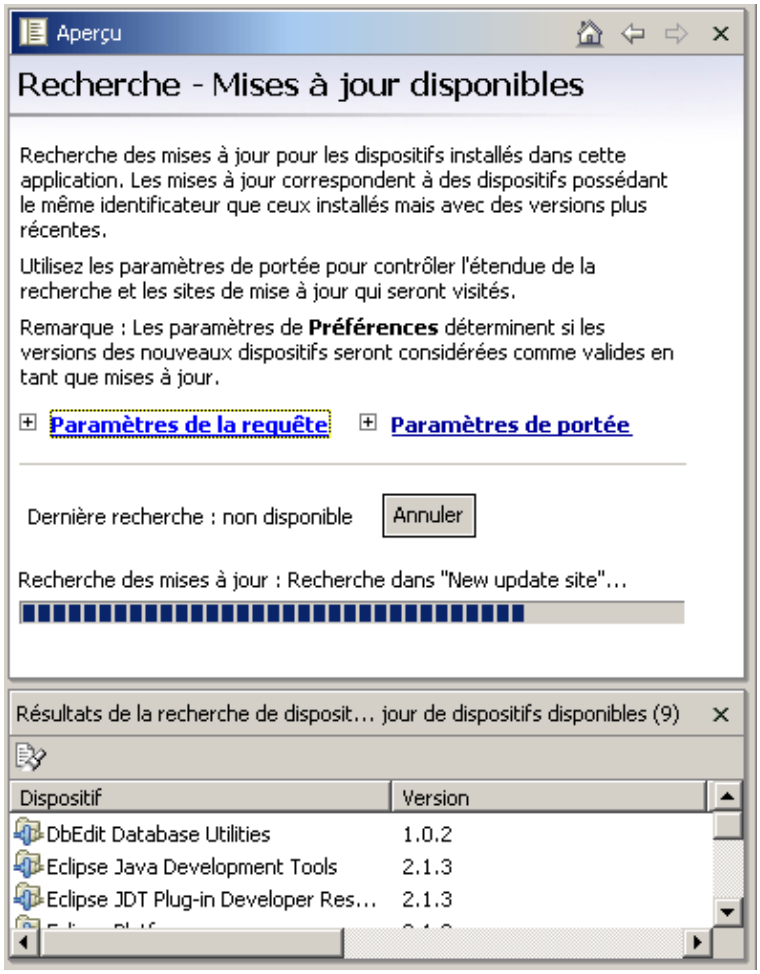
Cliquez sur le bouton "Installer". Une fois l'installation terminée, il faut relancer Eclipse.



Pour rechercher les mises à jour, il est aussi possible dans la perspective "Installation/mise à jour" de cliquer sur "Mise à jour disponible" dans la vue "Mises à jour des dispositifs".



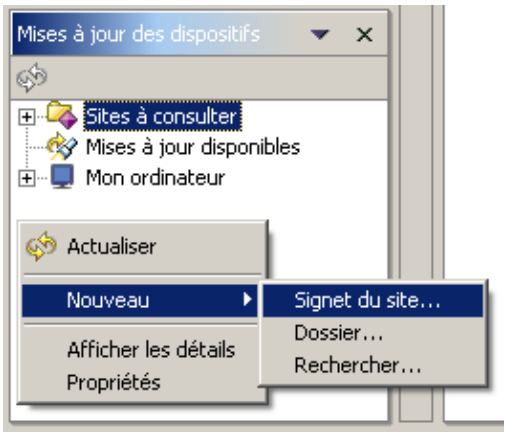
La progression et le détail des mises à jour trouvées sont affichés dans la vue "Aperçu"



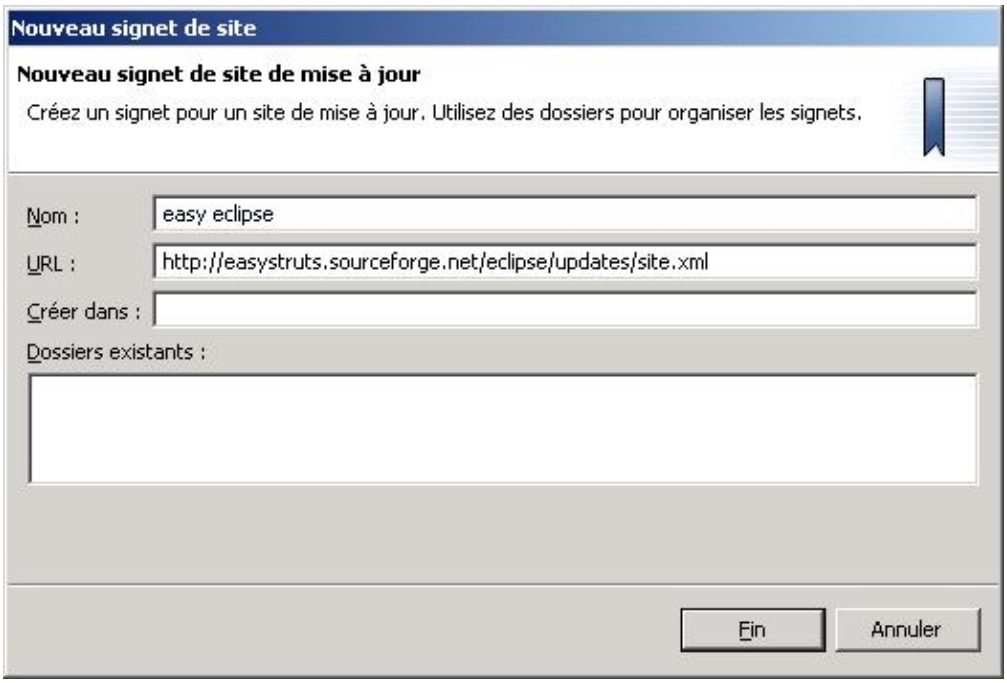
14.3.3. Installation d'un nouveau plug−in

L'exemple ci dessous met en oeuvre l'installation du plug−in Easy Struts. Pour d'autres plug−in permettant leur mise à jour par le réseau, la procédure est similaire.

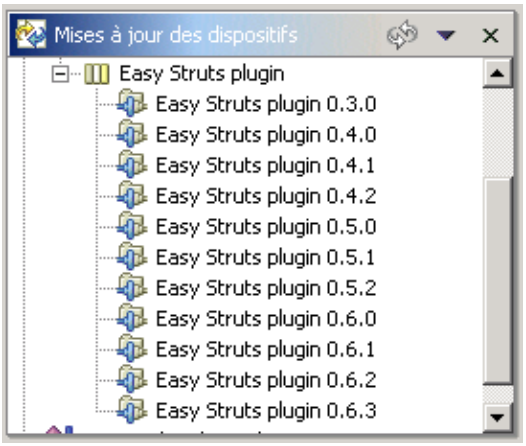
Pour ajouter une nouvelle source, il suffit de sélectionner l'option « Nouveau / Signet du site ... » du menu contextuel de la vue « Mises à jour des dispositifs ».



Une boîte de dialogue permet de saisir un nom et l'URL du signet.

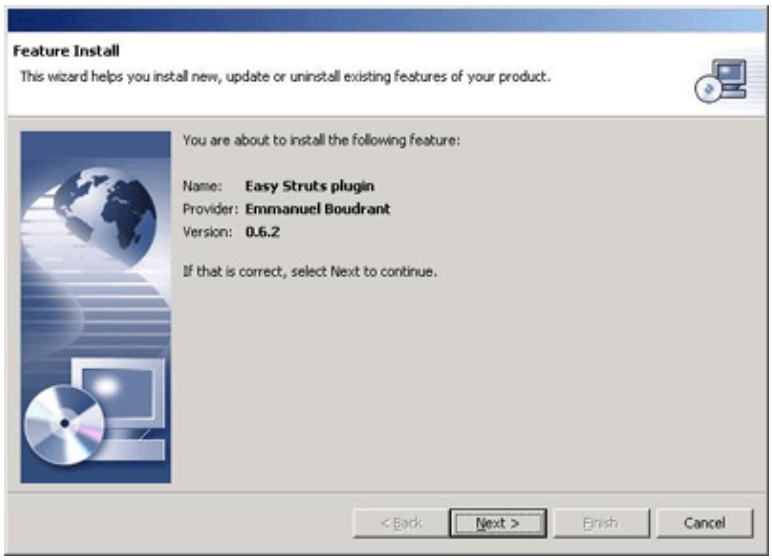


Il faut saisir le nom et l'url du fichier site.xml désiré puis cliquer sur le bouton "Fin".

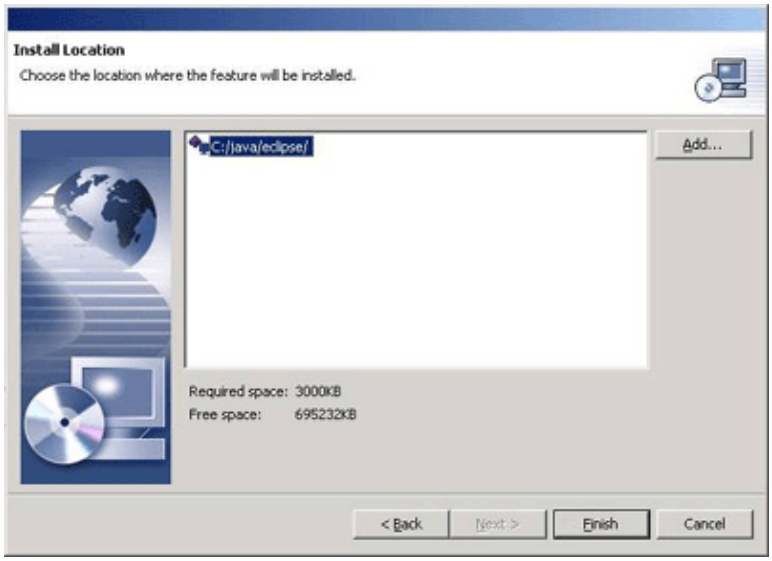


L'application va lire le fichier et affiche les plug−ins disponibles dans l'arbrescence. Il suffit de sélectionner la version désirée. La vue "Aperçu" affiche le details du plug−in sélectionné.

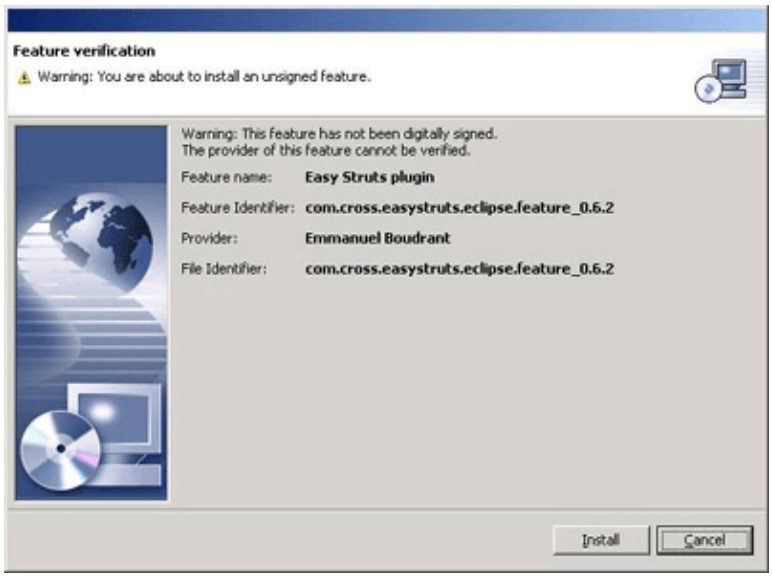
Pour installer la version sélectionnée, cliquer sur le bouton "Install" dans la vue « Aperçu ».



Cliquer sur le bouton "Next". Lire et accepter la licence, puis cliquer sur le bouton “Next”



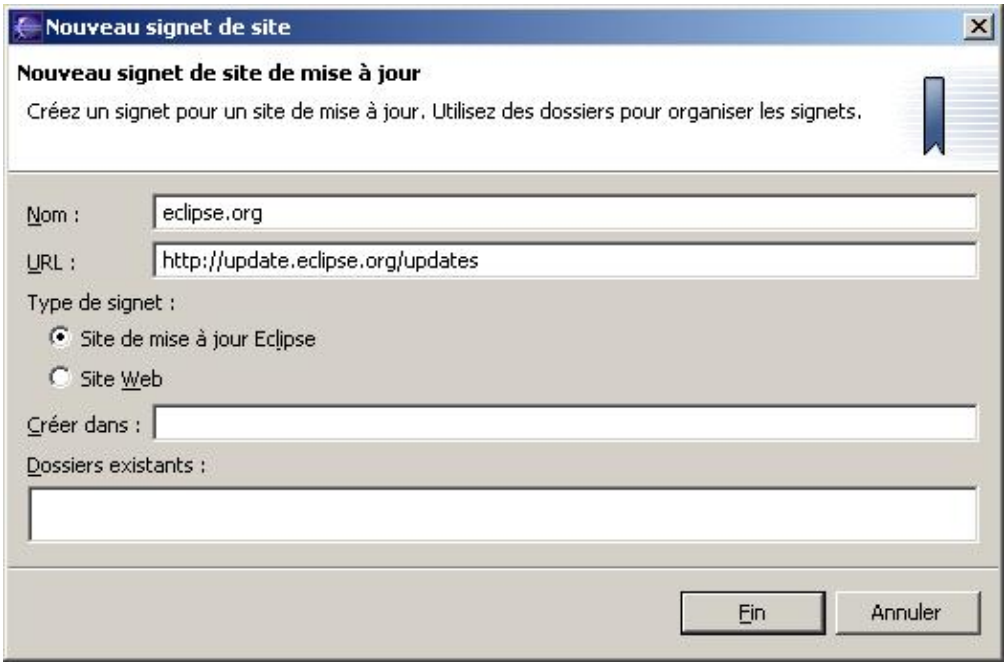
Sélectionner la cible d'installation et cliquer sur le bouton «Finish».



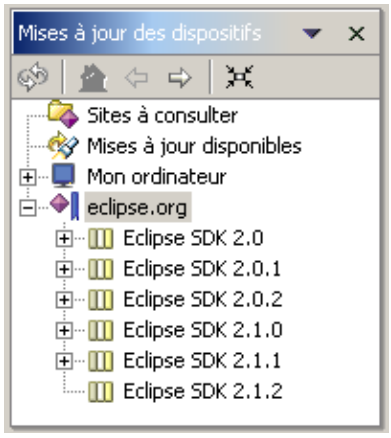
Il faut confirmer l'installation, car le package n'est pas signé, en cliquant sur le bouton « Install »

Le plug−in est téléchargé et installé. L'assistant propose de redémarrer le plan de travail pour que les modifications soient prises en compte. Cliquez sur le bouton "Yes".

Il peut être intéressant d'ajouter le site officiel d'Eclipse pour ajouter des plug−ins qui ne sont pas fournis en standard.



Un fois le nouveau signet configuré, il suffit de naviguer dans l'arborescence en fonction de la version de la plate−forme pour pouvoir installer un ou plusieurs plug−ins parmis ceux proposés.

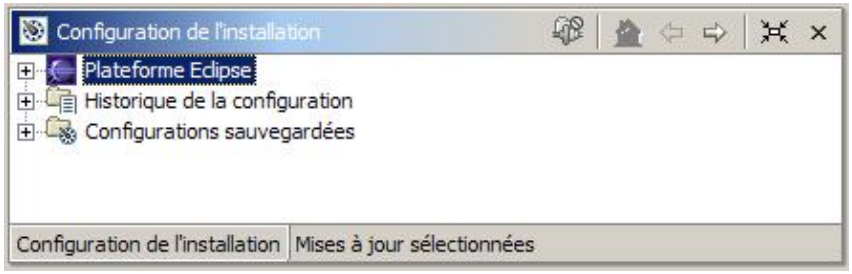


14.3.4. Sauvegarde et restauration d'une configuration

A chaque mise à jour, Eclipse enregistre la configuration.

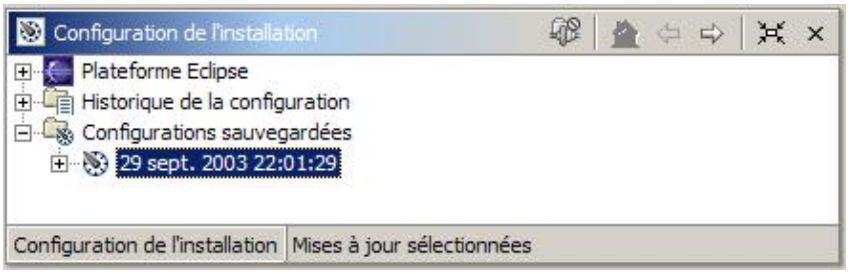
Il est aussi possible de demander la sauvegarde explicite de la configuration : il faut, dans la perspective

« Installation / Mise à jour », sélectionner « Plate−forme Eclipse » dans la vue « Configuration de l'installation »



Il suffit de cliquer sur le bouton « Sauvegarder » dans la vue « Aperçu »

Pour restaurer une configuration, il suffit de sélectionner, dans la vue « Configuration de l'installation », la sauvegarde automatique (dans « historique de la configuration) ou la sauvegarde manuelle (dans « Configuration sauvegardées).



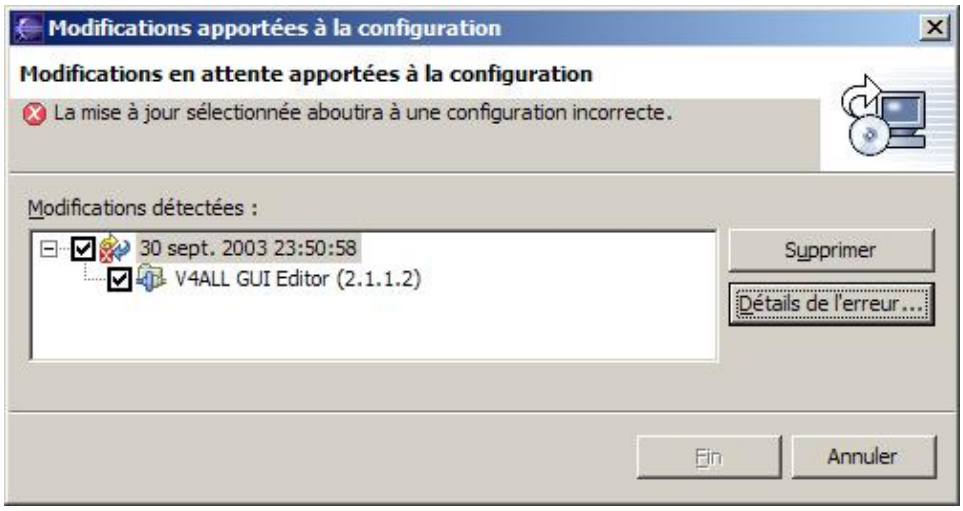
Il suffit ensuite de cliquer sur le bouton « Restaurer » de la vue « Aperçu ».

Cette opération nécessite un redémarrage de l'application.

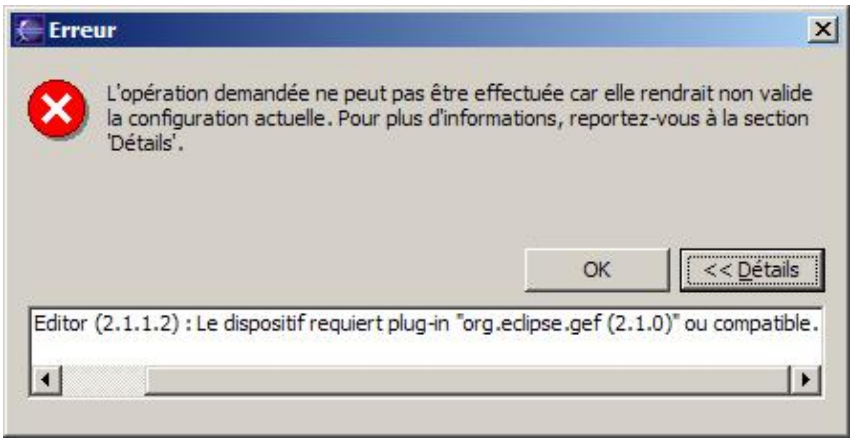
14.3.5. Résolution des problèmes de dépendances

Il est possible que la mise à jour ou l'installation d'un plug−in échoue notamment pour des questions de dépendances vis−à−vis d'un autre plug−in.

Dans ce cas, la mise à jour est signalée avec une croix blanche dans un rond rouge.

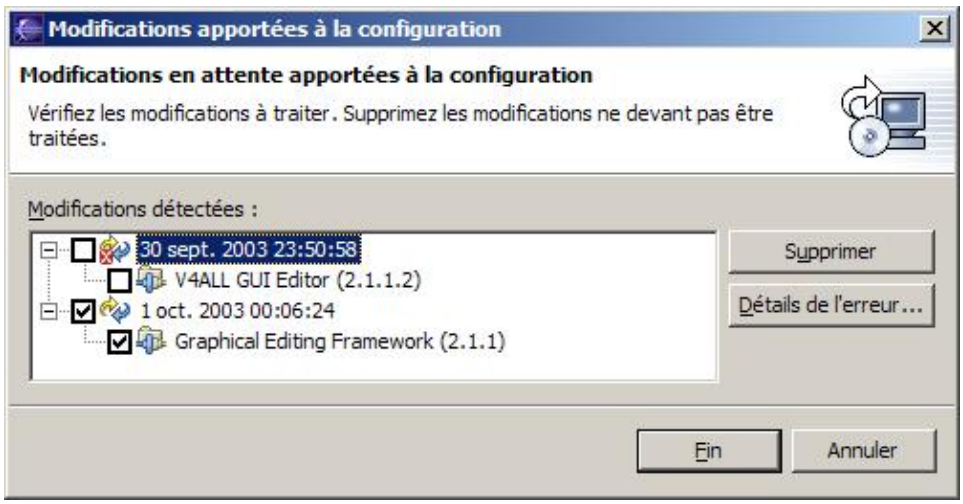


Pour avoir plus d'informations, il suffit de sélectionner la mise à jour et de cliquer sur le bouton « Détails de l'erreur ».

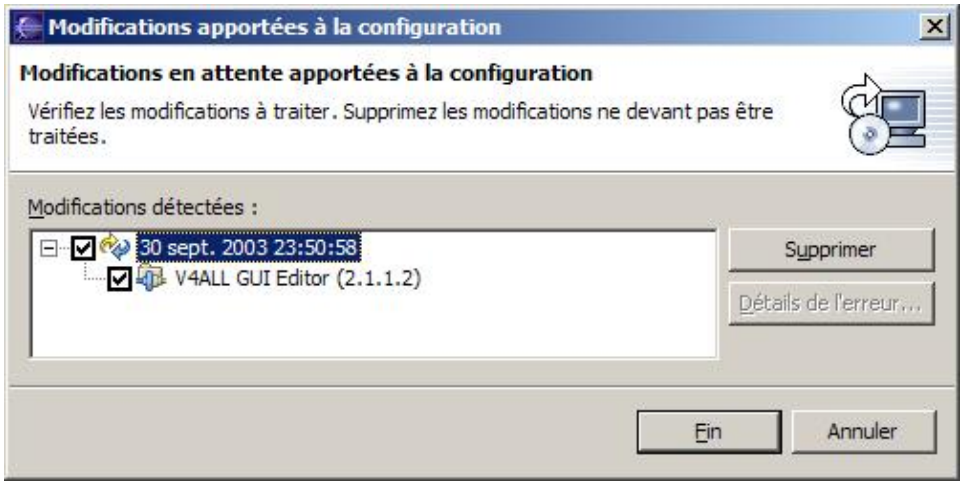


Dans l'exemple ci−dessus, il manque un plug−in ou la version installée n'est pas correcte.

Pour résoudre le problème, il suffit de quitter Eclipse, d'installer le plug−in (en le décompressant dans le bon répertoire) et de relancer Eclipse.



Il suffit alors de décocher la mise à jour posant problème et de cliquer sur le bouton « Fin » afin de mettre à jour le plug−in manquant. Cette opération nécessite le redémarrage d'Eclipse.



Suite à ce redémarrage, le plug−in erroné n'apparaît plus en erreur et peut être installé en cliquant sur le bouton « Fin ».

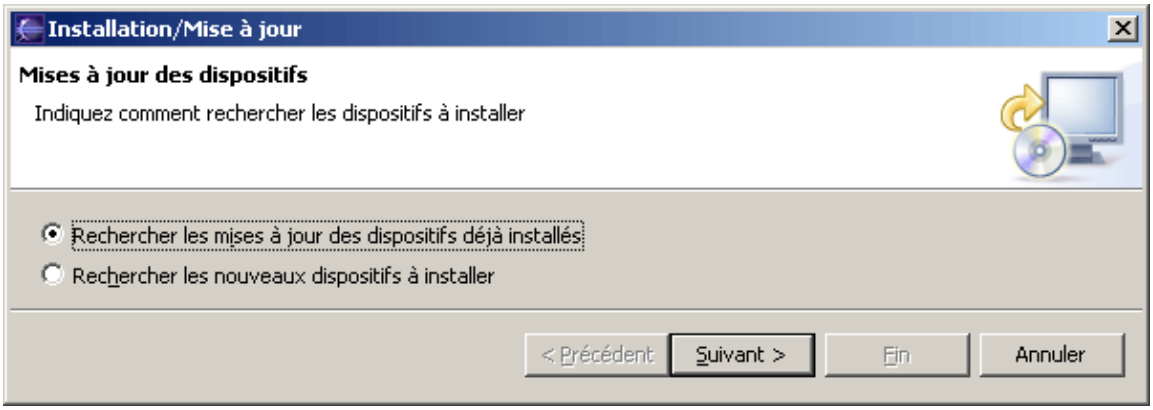
14.4. La mise à jour des plug−ins avec Eclipse 3.0

La perspective "Installation et mise à jour" est supprimée et l'option "Mise à jour de logiciels" du menu "Aide" propose maintenant deux options :

- "Rechercher et installer ..."
- "Configuration du produit ..."

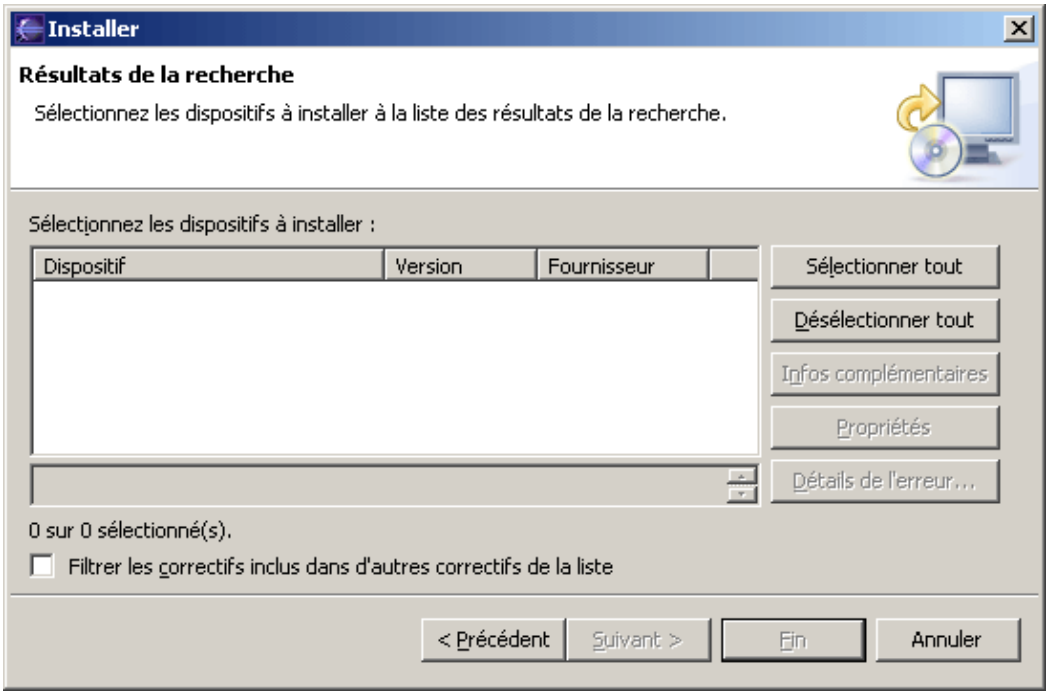
14.4.1. Recherche et installation de plug−ins

L'option "Rechercher et installer ... " permet de rechercher et d'installation un plug−in ou une nouvelle version d'un plug−in installé grâce à un assistant.

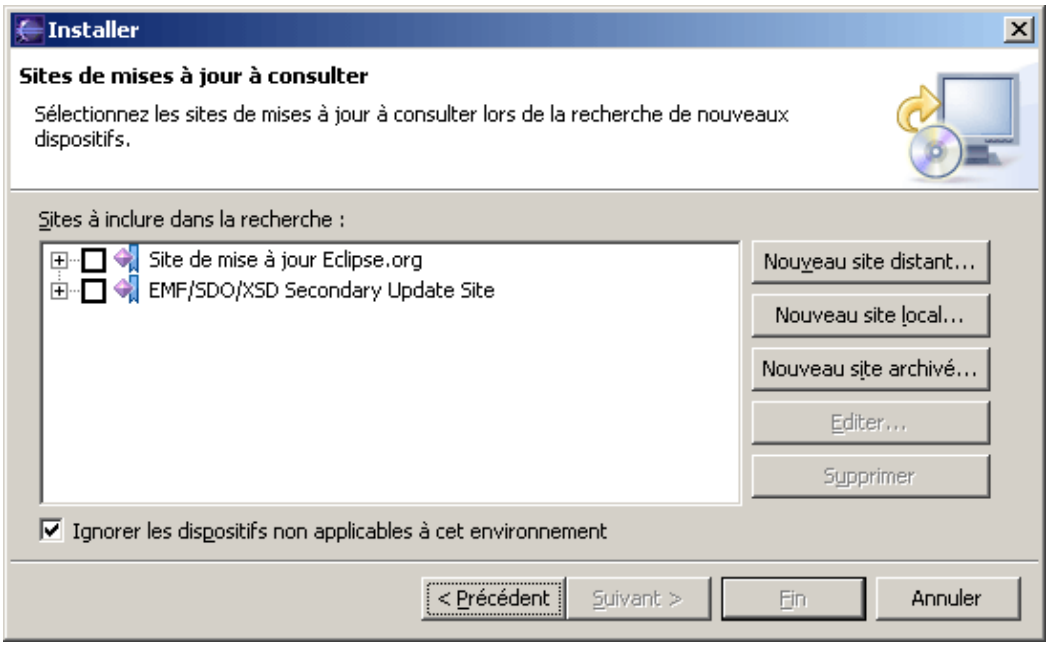


La première page permet de sélectionner le type d'élément à rechercher.

L'option "Rechercher les mises à jour des dispositifs déjà installés" permet de rechercher dans les sites enregistrés, des nouvelles versions des plug−ins installés.

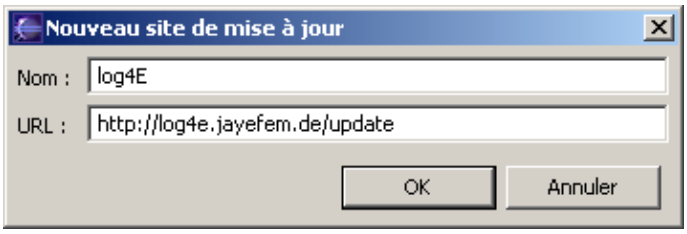


L'option "Rechercher les nouveaux dispositifs à installer" permet de gérer la liste des sites proposant des plug−ins et ainsi d'obtenir de nouveaux plug−ins.



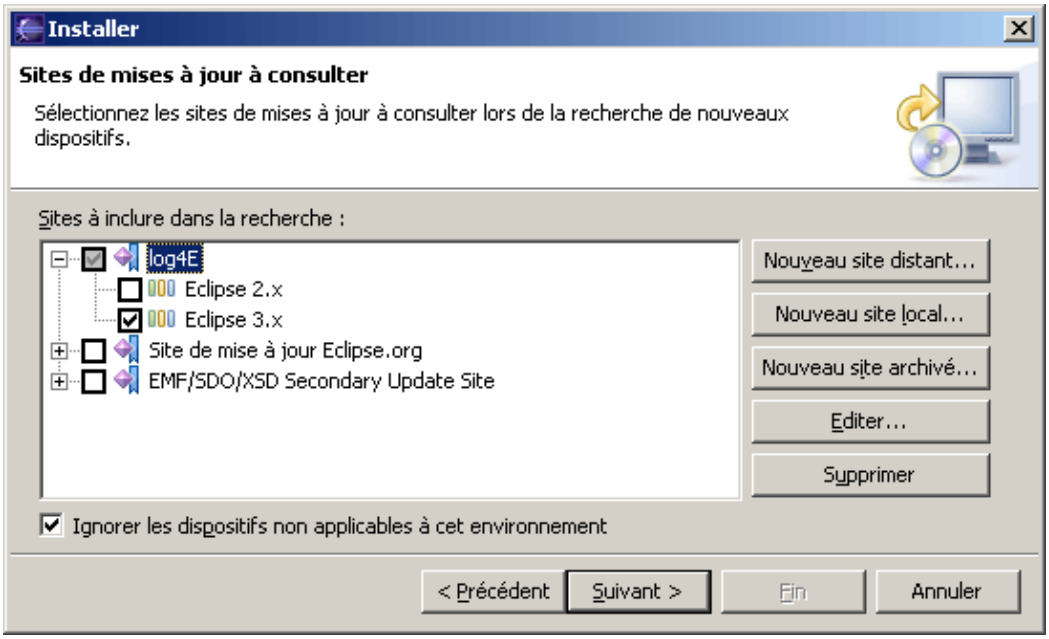
Cette page permet de sélectionner les sites concernés par la recherche. Il est possible d'ajouter de nouveaux sites grâce aux boutons de droite.

Un clic sur le bouton "Nouveau site distant ..." ouvre une boîte de dialogue qui permet de saisir les informations le nouveau site. Il suffit de saisir un nom et l'url du site.

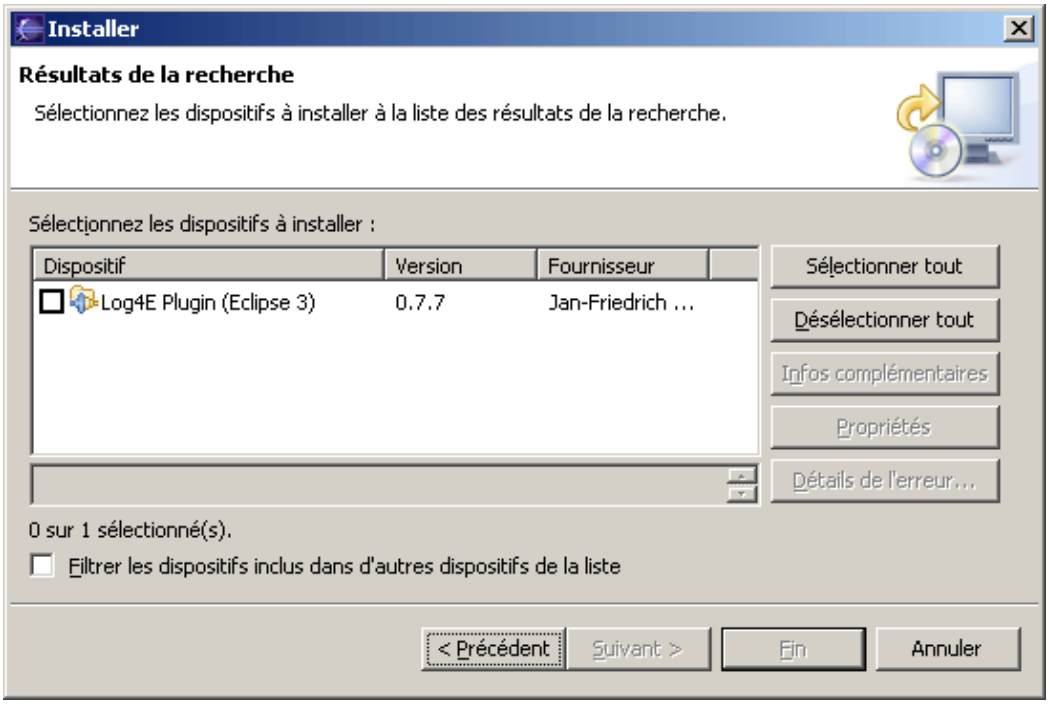


Cliquez sur le bouton "OK"

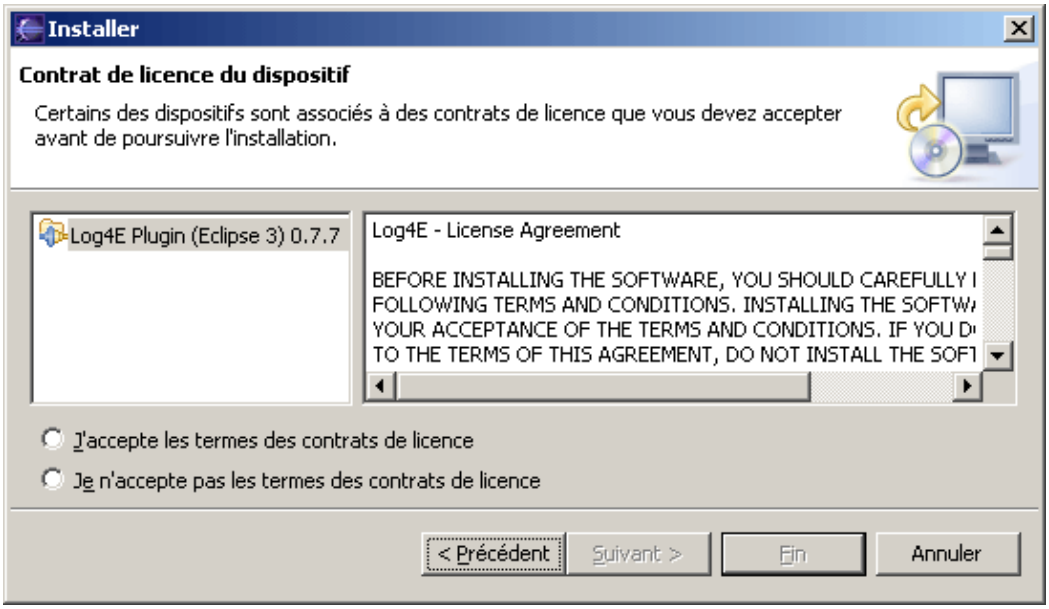
Le nouveau site est ajouter dans la liste. Il faut ouvrir l'arborescence du nouveau site pour sélectionner les dispositifs désirés.



Un clic sur le bouton "Suivant" permet de lancer la rechercher. Les résultats de la recherche sont affichés dans la page suivante.



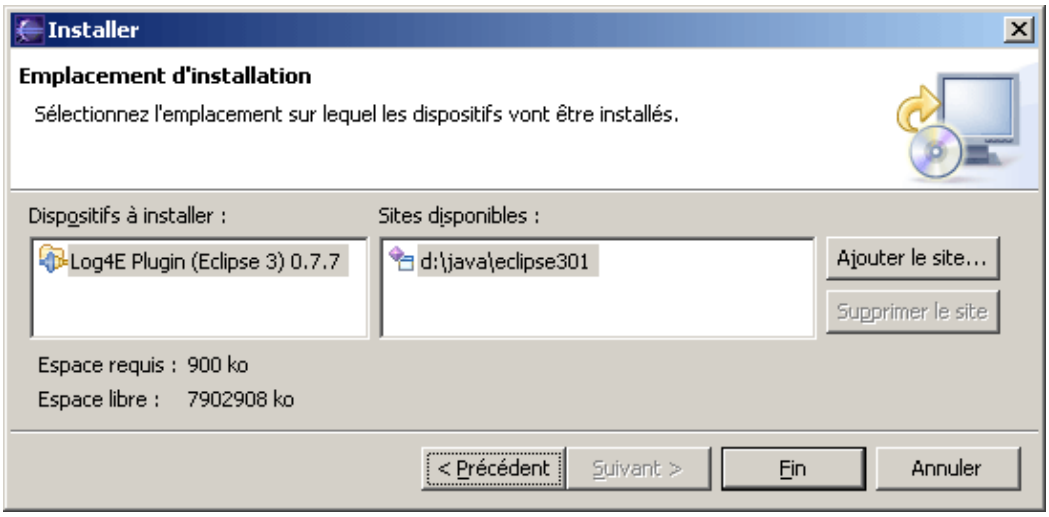
Il suffit alors de sélectionner le ou les plug−ins désirés puis de cliquer sur le bouton "Suivant".



La page suivante permet de lire et d'accepter la licence pour poursuivre les traitements.

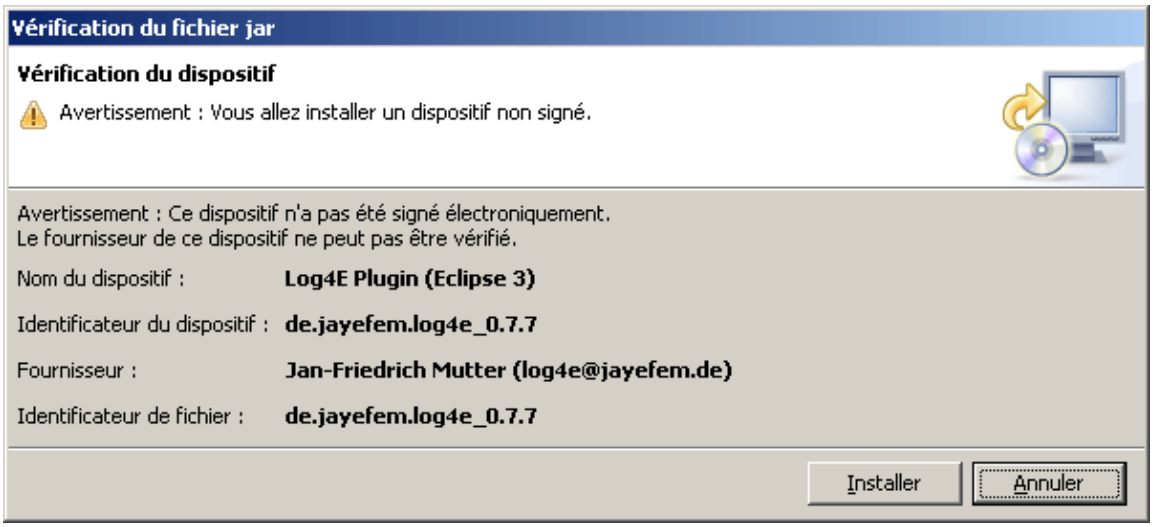
Si vous acceptez la licence, cliquez sur le bouton "J'accepte les termes des contrats de licence" puis sur le bouton "Suivant".

La page suivante permet de sélectionner l'emplacement de l'installation du plug−in



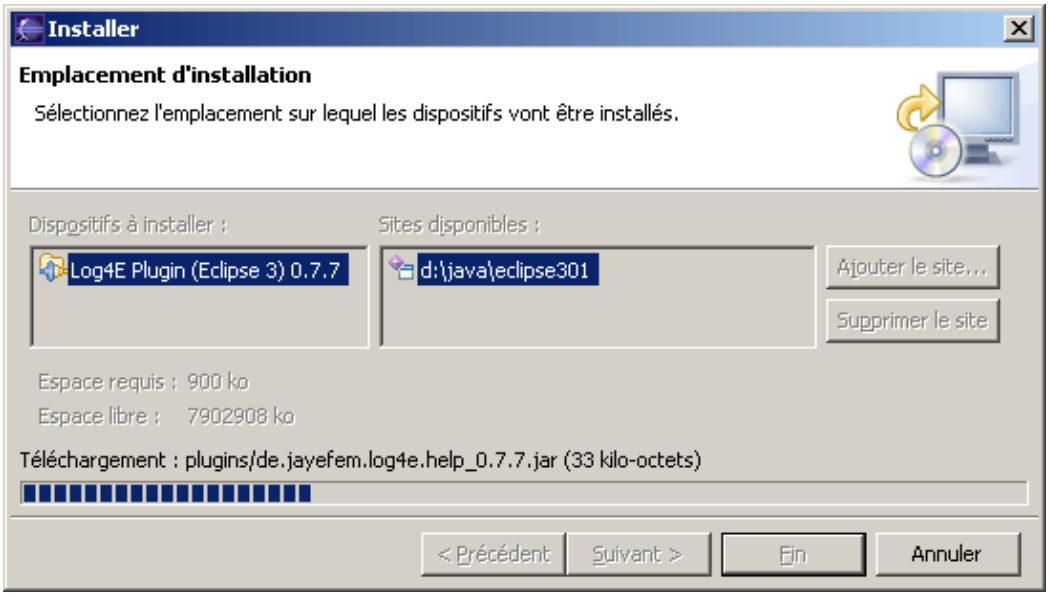
Cliquez sur le bouton "Fin".

Si le plug−in n'est pas signé, une demande de confirmation d'installation est demandée

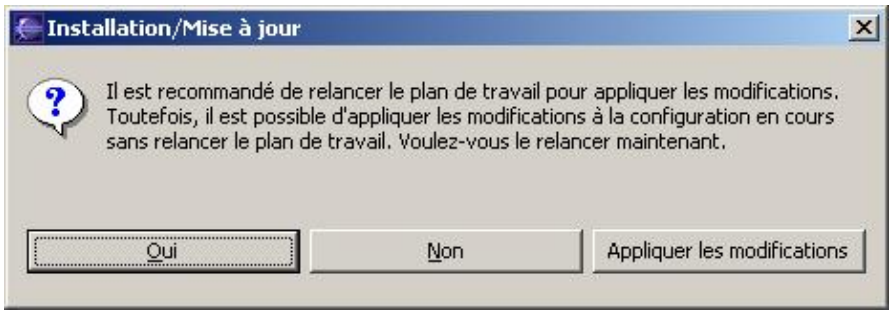


Dans ce cas, cliquez sur le bouton "Installer".

Les fichiers sont téléchargés et installés.



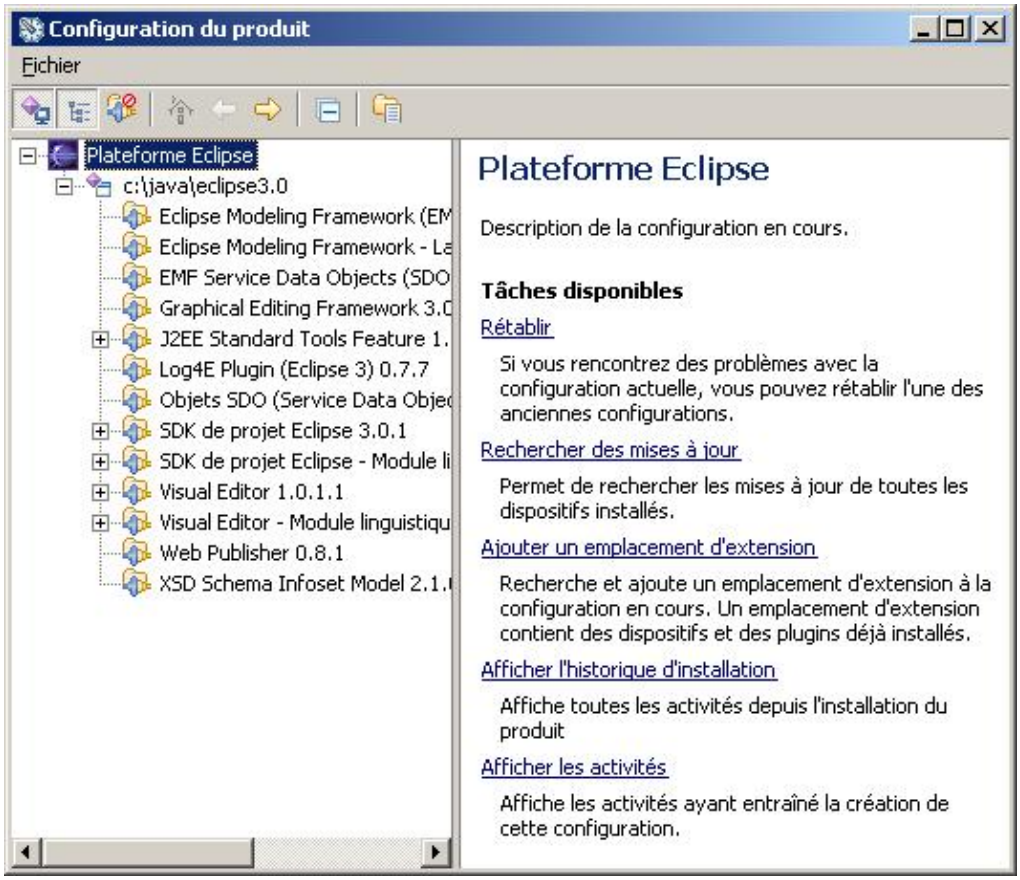
Une fois ces opérations terminées, le système recommande de redémarrer la plate−forme.



Il est donc recommandé de cliquer sur le bouton "Oui" surtout pour pouvoir utiliser immédiatement le plug−in fraîchement installé.

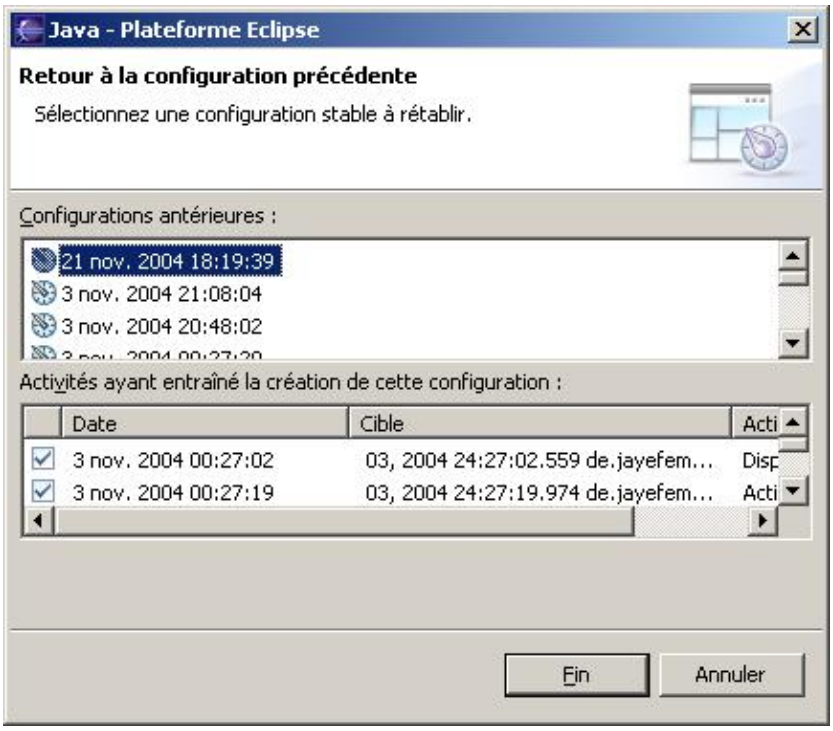
14.4.2. La configuration du produit

L'option Configuration du produit permet un gestion détaillé de la plate−forme.



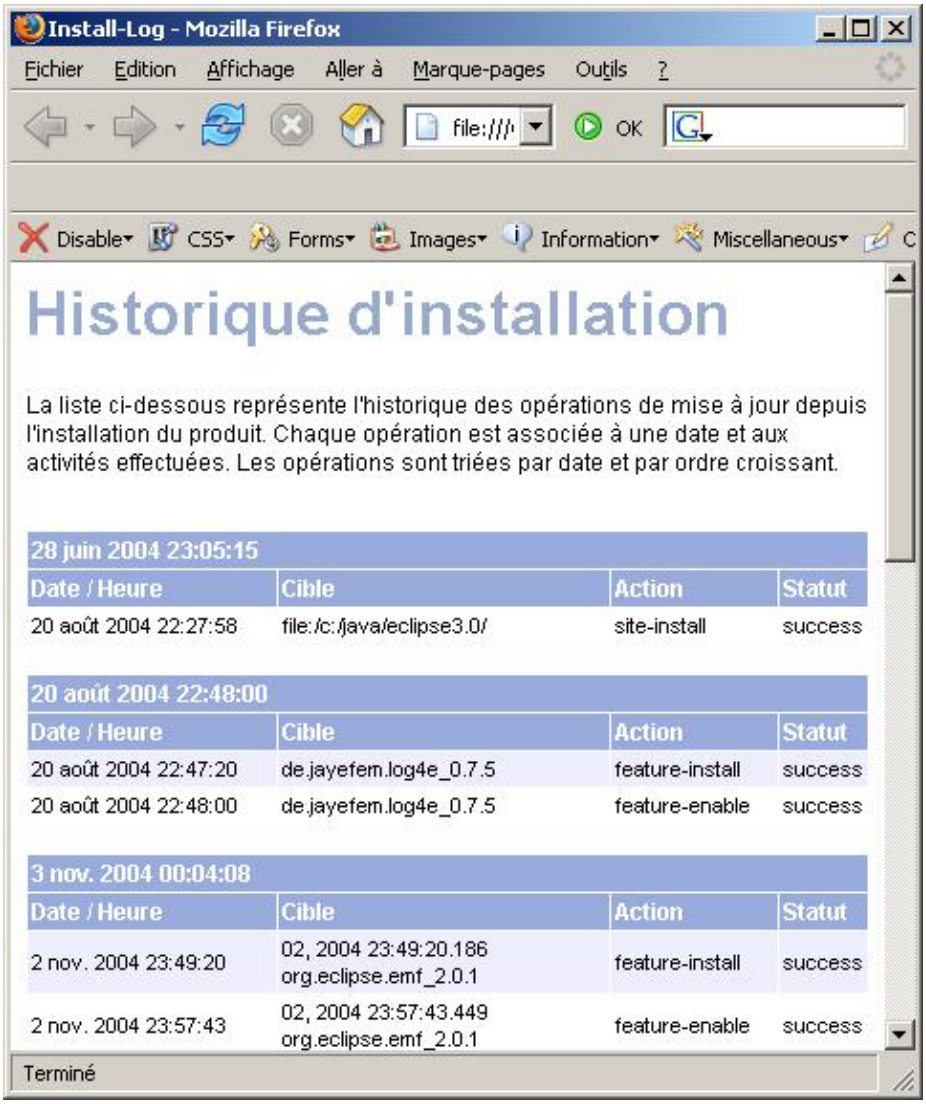
En sélection la racine de l'arborescence "Plateforme Eclipse", plusieurs tâches sont disponibles.

La tache "Rétablir" permet de restaurer une ancienne configuration.

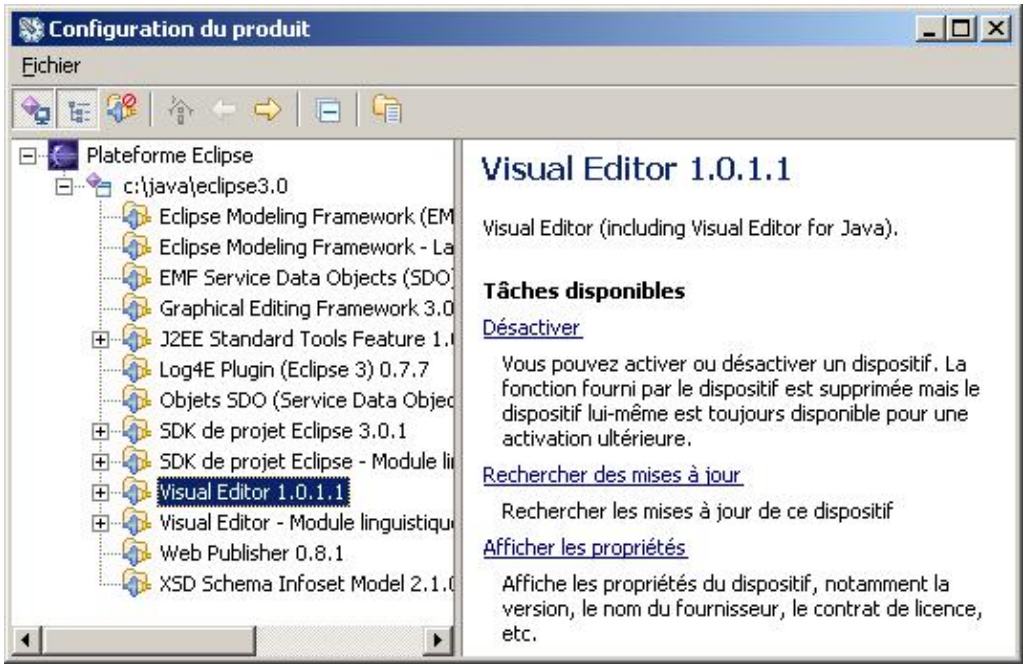


Il suffit de sélectionner la configuration désirée et de cliquer sur le bouton "Fin".

La tâche "Afficher l'historique d'installation" affiche une page web contenant l'historique des installations. Le bouton  permet aussi de demander l'exécution de cette tâche.



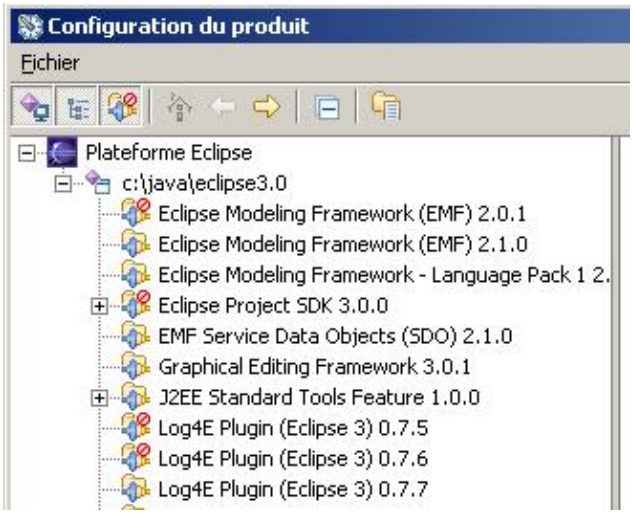
Pour chaque plug−in installé, il est possible de réaliser plusieurs tâches en le sélectionnant :



La tâche "Désactiver" permet désactiver le plug−in après une confirmation

Il faut rédemarrer la plateforme que la désactivation soit effective.

Par défaut, les plug−ins désactivés ne sont pas affichés. Il faut cliquer sur le bouton  pour les afficher dans l'arborescence.



La tâche "Activer" permet de réactiver le plug−in après une configuration. La relance de la plate−forme n'est pas obligatoire mais fortement recommandée.

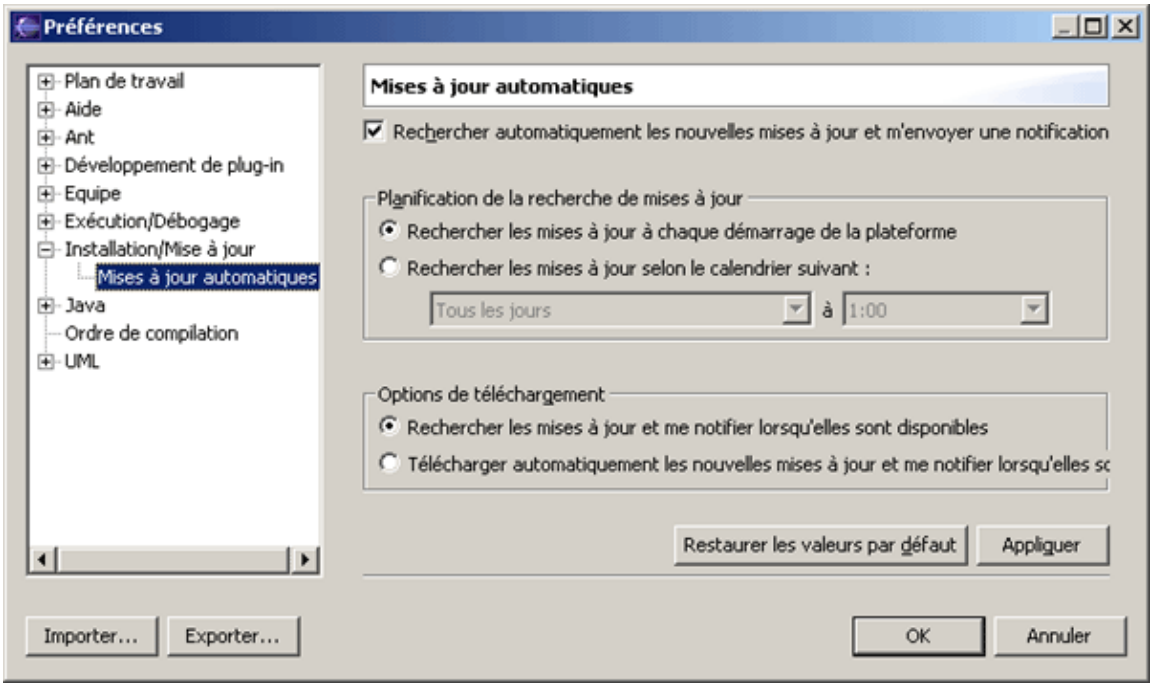
La tâche "Désinstaller" permet de désinstaller le plug−in après confirmation

La relance de la plate−forme n'est pas obligatoire mais fortement recommandée.

14.4.3. Mises à jour automatiques

Il est possible de paramétrer une fréquence de mises à jour automatiques.

Pour cela dans les préférences, il faut cocher l'option "Rechercher automatiquement les nouvelles mises à jour et m'envoyer une notification"

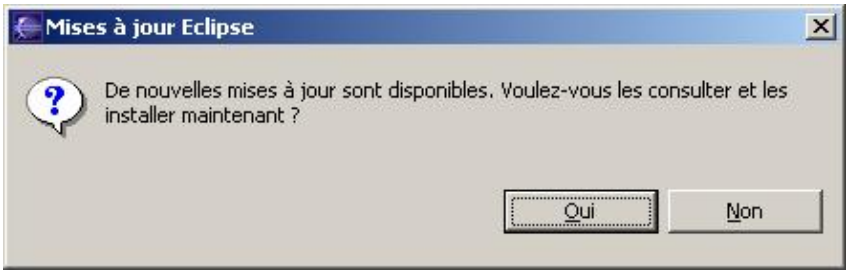


Il est alors possible de paramétrer la planification et les options de téléchargement pour une mise à jour automatique des éléments enregistrés dans le système de mise à jour de l'application.

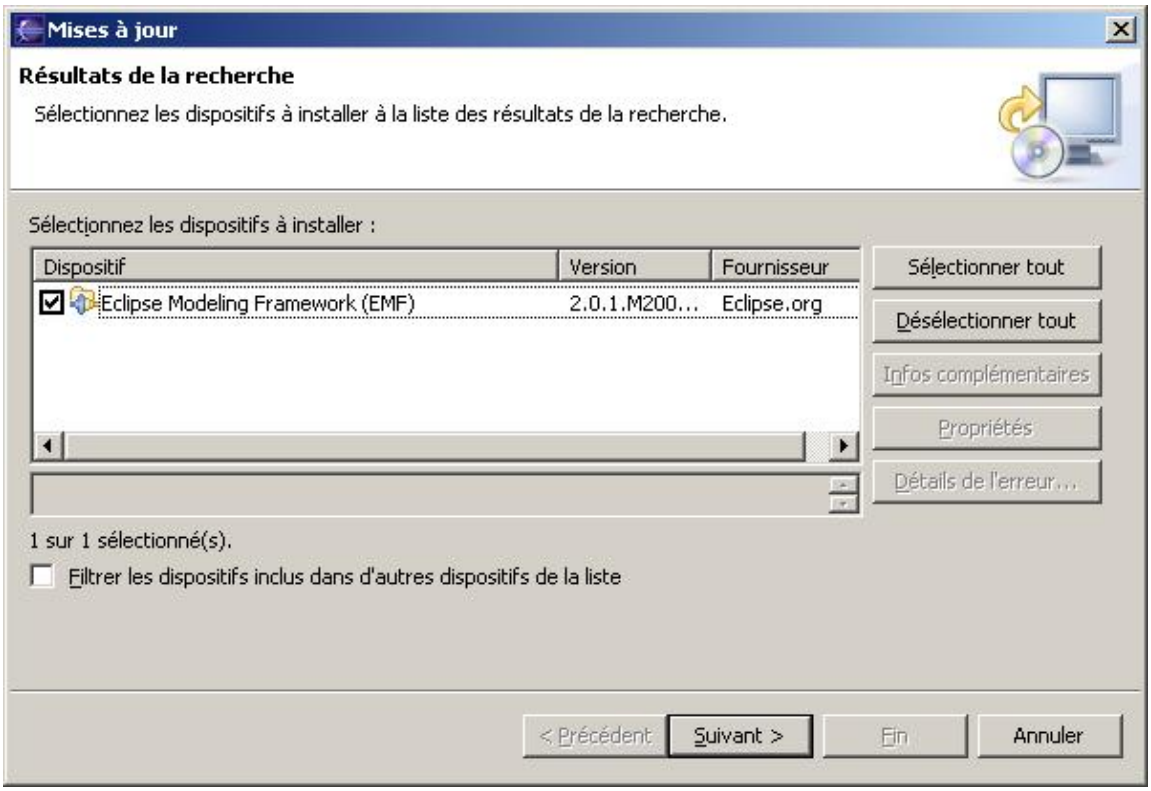
Il est ainsi possible de préciser deux choses :

- le moment de la recherche automatique : soit à chaque démarrage de l'application soit à une périodicité et une heure précise.
- informer de la disponibilité de mises à jour ou télécharger automatiquement les mises à jour trouvés et informer de la fin de leur téléchargement

Si le système détecte des mises à jour, une boîte de dialogue est affichée à l'issu des traitements de recherche :



Un clic sur le bouton "Oui" permet d'afficher une boîte de dialogue recensant les mises à jours trouvées.



Il faut sélectionner le ou les éléments à mettre à jour et de cliquer sur le bouton "Suivant".

Il suffit enfin de suivre les étapes de l'assistant pour obtenir et installer le ou les dispositifs choisis.

Partie 4 : le développement avancé avec Java

Cette quatrième partie présente le développement avec Java nécessitant quelques paramétrage ou l'utilisation de plug−ins tiers.

Elle comporte les chapitres suivants :

- Des plug−ins pour le développement avec Java : présente quelques plug−ins tiers facilitant le développement avec Java.
- Le développement J2EE : Présente des plug−ins pour faciliter le développement avec J2EE.
- Java Server Faces et Eclipse : détaille au travers d'un exemple l'utilisation d'Eclipse pour développer une application web utilisant JSF.
- JAXB et Eclipse : détaille l'utilisation de JAXB avec Eclipse.
- Struts et Eclipse : détaille au travers d'un exemple l'utilisation du plug−in Esay Struts pour développement des applications web utilisant Struts.
- Le développement d'interfaces graphiques : présente plusieurs plug−ins pour faciliter le développement d'applications graphiques utilisant AWT, Swing ou SWT.
- Eclipse WebTools Platform : Propose de mettre en oeuvre le plug−in Eclipse Web Tools Platform (WTP).

15. Des plug−ins pour le développement avec Java

Il existe de nombreux plug−ins développés par des tiers pour enrichir Eclipse. Certains de ces plug−ins sont spécialement dédiés à faciliter le développement de code avec Java.

Ce chapitre va présenté plusieurs de ces plug−ins :

- Jalopy : ce plug−in permet de mettre en oeuvre l'outil open source du même nom qui permet de formatter le code source
- Log4E : ce plug−in permet de faciliter la mise en oeuvre d'un système de log (log4J, api de logging du JDK 1.4 ou common logging)

15.1. Le plug−in Jalopy

Jalopy est un utilitaire open source très pratique qui permet de formatter du code source Java et même de vérifier l'application de normes de codage.

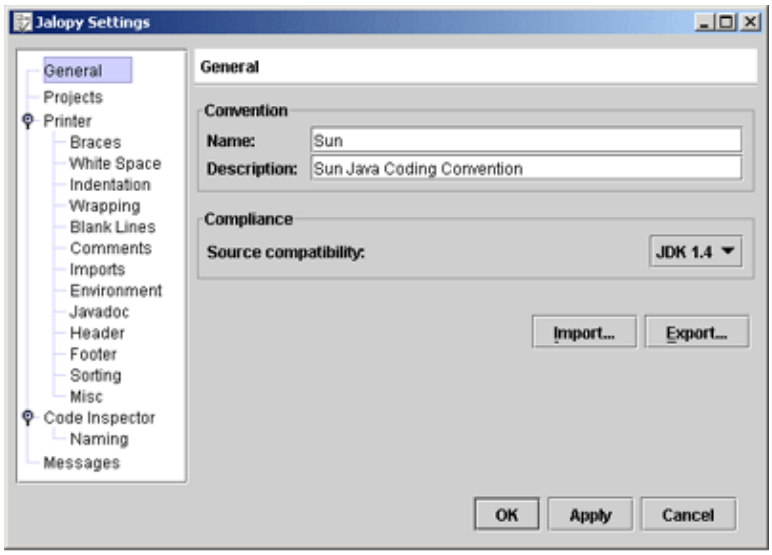
Il permet notamment :

- d'indenter le code
- de générer des modèles de commentaires Javadoc dynamiques en fonction des éléments du code à documenter (par exemple générer un tag @param pour chaque paramètre d'une méthode)
- d'organiser l'ordre des clauses import
- d'organiser l'ordre des membres d'une classe selon leur modificateur
- de vérifier l'application de normes de codage,
- ...

Il existe des plug−ins pour plusieurs IDE dont un pour Eclipse : il suffit de télécharger le fichier jalopy−eclipse−0.2.6.zip sur le site http://jalopy.sourceforge.net/download.html

Pour installer le plug−in, il faut décompresser le contenu de l'archive dans un réperoire temporaire et copier le répertoire de.hunsicker.jalopy.plugin.eclipse_0.2.6 dans le répertoire plugins d'Eclipse.

L'option "Jalopy preferences" du menu "Fenêtre" permet de paramétrer Jalopy



Pour la mise en oeuvre, il suffit d'utiliser le menu contextuel "Format with Jalopy" de l'éditeur de code Java.

Exemple :

```
  package com.moi.test;
  public class TestJalopy {
  public static void main(String[] args) {}
  public void maMethode(int a, int b) {}
  private int maMethode2(int a) { return a; }
  public void maMethode3() {}
  }
```

Résultat :

```
package com.moi.test;

/**
 * DOCUMENT ME!
 *
 * @author $author$
 * @version $Revision$
 */
public class TestJalopy {
    /**
     * DOCUMENT ME!
     *
     * @param args DOCUMENT ME!
     */
    public static void main(String[] args) {
    }

    /**
     * DOCUMENT ME!
     *
     * @param a DOCUMENT ME!
     * @param b DOCUMENT ME!
     */
    public void maMethode(int a, int b) {
    }

    /**
     * DOCUMENT ME!
     */
    public void maMethode3() {
    }
```

```
    /**
     * DOCUMENT ME!
     *
     * @param a DOCUMENT ME!
     *
     * @return DOCUMENT ME!
     */
    private int maMethode2(int a) {
        return a;
    }
}
```

Le formattage du code source est réalisé en tenant compte des nombreux paramètres définis dans Jalopy.

Il est également possible de demander le formattage de l'ensemble des fichiers source Java contenus dans un ou plusieurs packages ou répertoires. Il suffit de sélectionner le ou les packages ou répertoires et de sélectionner l'option "Format" du menu contextuel.

15.2. Log4E

Log4E est un plug−in gratuit dont le but est de faciliter l'utilisation d'une API de logging dans du code java.

Log4E propose en standard d'utiliser 3 API : Log4J, l'api Logging du JDK 1.4 et l'api Common Logging du projet Jakarta.

Le site officiel de ce plug−in est à l'url : http://log4e.jayefem.de/index.php/Main_Page

15.2.1. Installation

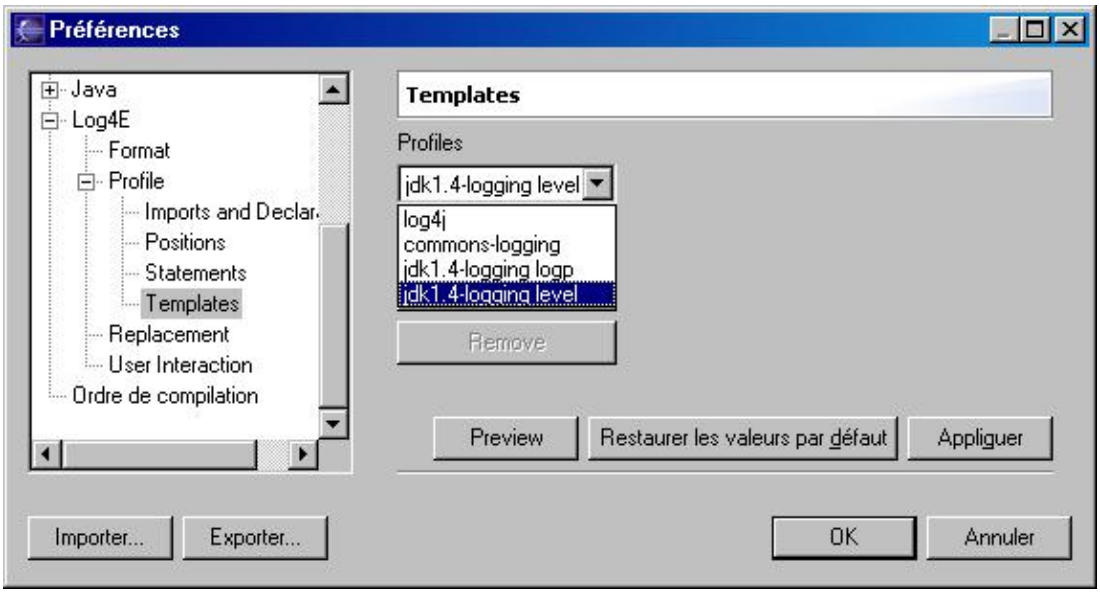
Il y a deux façons d'obtenir le plug−in et de l'installer :

- télécharger l'archive .zip contenant le plug−in et le décompresser dans le répertoire d'installation d'Eclipse
- utiliser la fonctionnalité de mise à jour d'Eclipse avec l'url : http://log4e.jayefem.de/update

15.2.2. Les paramètres

Le plug−in Log4E possède de nombreux paramètres modifiables dans la fenêtre de gestion des paramètres.

Un des paramètres le plus important se situe à l'arborescence Log4E / Profile / Templates



Le profile permet de sélectionner l'api qui sera utilisée par le plug−in lors de la génération de code pour les opérations demandées.

15.2.3. Utilisation

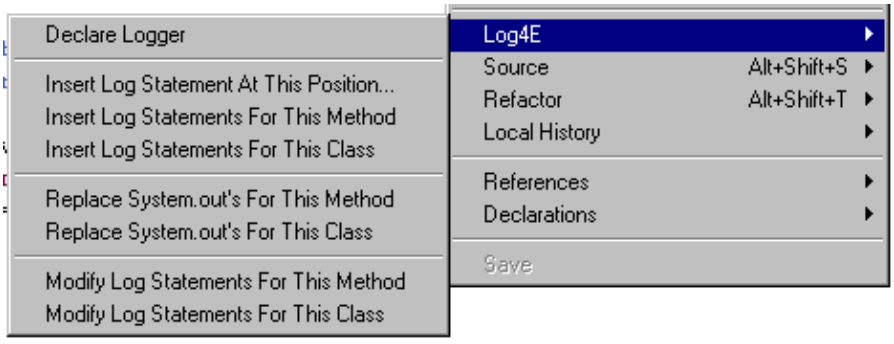
Le fichier suivant sera utilisé dans cette section :

Exemple :

```
package com.jmd.test.java;

public class TestLog4E {

            public static void main(String[] args) {
                        int a = 1;
                        int b = 2;
                        System.out.println("resultat = " + additionner(a, b));
            }

            public static int additionner(int a, int b) {
                        int res = a + b;
                        return res;
            }
}
```

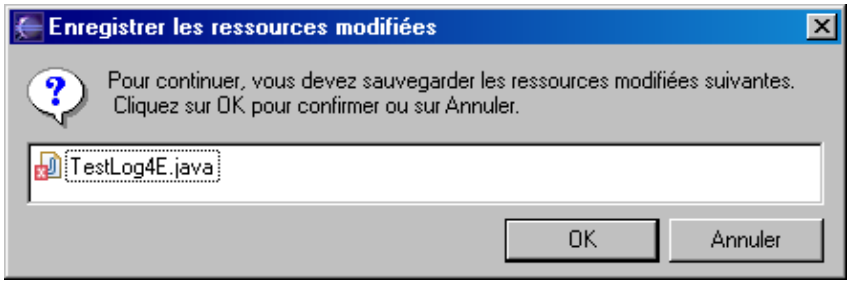
L'api de logging sélectionnée dans la préférences pour les exemples de cette section est celle du JDK 1.4.

Le menu contextuel de l'éditeur de code se voie ajouter une nouvelle option nommée Log4E.



Avant de pouvoir utiliser une de ces options, il est nécessaire de sauvegarder le code source pour permettre au

plug−in de réaliser les modifications.



Cette nouvelle entrée du menu contextuel, propose plusieurs options réparties en 4 groupes.

Toute opérations impliquant une modification du code est soumise à l'approbation de l'utilisation sous la forme d'une vue permettant de comparer le code avant et après modification et d'accepter ou non les modifications.

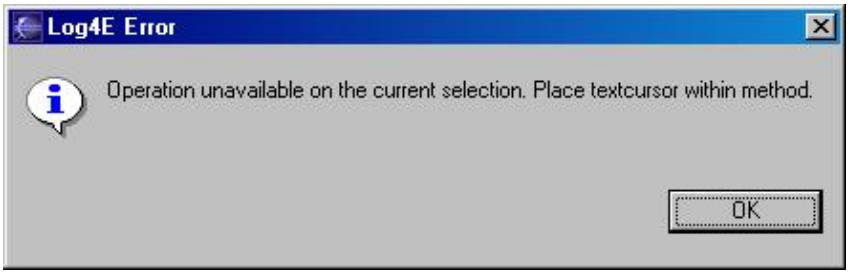
L'option « Declare Logger » permet de demander l'insertion dans le code des clauses imports ainsi que la déclaration des objets nécessaires à l'utilisation de l'api de logging .

Exemple :

```
package com.jmd.test.java;

import java.util.logging.Level;
import java.util.logging.Logger;

public class TestLog4E {

            /**
             * Logger for this class
             */
            private static final Logger logger = Logger.getLogger(TestLog4E.class
                                    .getName());

            public static void main(String[] args) {
                        int a = 1;
                        int b = 2;
                        System.out.println("resultat = " + additionner(a, b));
            }

            public static int additionner(int a, int b) {
                        int res = a + b;
                        return res;
            }
}
```

L'option “Insert Log Statement At The Position ...” permet d'insérer du code pour utiliser l'api de logging.

Pour utiliser cette option, il est nécessaire de placer le curseur de l'éditeur de code dans une méthode, sinon un message d'erreur est affiché.



L'option “Insert Log Statement For This Method” permet d'insérer automatique des appels à l'api de logging dans la méthode : au début de la méthode, à la fin de la méthode et dans les blocs try/catch.

Exemple avec la méthode additionner()

```
...
            public static int additionner(int a, int b) {
                        if (logger.isLoggable(Level.FINE)) {
                                    logger.fine("start");
                        }
                        int res = a + b;
                        if (logger.isLoggable(Level.FINE)) {
                                    logger.fine("end");
                        }
                        return res;
            }
...
```

Pour utiliser cette option, il est nécessaire de placer le curseur de l'éditeur de code dans une méthode, sinon un message d'erreur est affiché.

L'option “Insert Log Statement For This Class” est identique à l'option précédente mais effectue les traitements sur toute la classe.

L'option “Replace System.out's For This Method” permet de remplacer les utilisations de la classe System.out ou System.err par l'utilisation de l'api de logging.

Pour utiliser cette option, il est nécessaire de placer le curseur de l'éditeur de code dans une méthode, sinon un message d'erreur est affiché.

L'option “Replace System.out's For This Class” est identique à l'option précédente mais effectue les traitements sur toute la classe.

16. Le développement J2EE

Eclipse ne propose pas par défaut de plug−in permettant de faciliter le développement d'applications J2EE. Il existe cependant plusieurs plug−ins qui simplifient la mise en oeuvre de certaines taches : Struts, Tomcat, EJB, ... Le plug−in open source le plus complet, nommé Lomboz, propose des fonctionnalités pour faciliter le développement d'applications web (servlets et JSP), d'EJB et de services web.

Le sous projet "Eclipse Web Tools Platform" en cours de développement a pour but de développer des plug−ins pour le développement d'application Web et J2EE.

Ce chapitre va présenter les plug−ins suivants :

- le plug−in Tomcat de Sysdeo pour la mise en oeuvre de Tomcat pour des applications web
- le plug−in Lomboz pour le développement d'applications J2EE avec utilisation possible de plusieurs serveurs d'applications

16.1. Le plug−in Tomcat de Sysdeo

La société Sysdeo propose un plug−in pour faciliter le développement d'applications utilisant Tomcat notamment en permettant le démarrage et l'arrêt de Tomcat, le packaging d'une application sous la forme d'un fichier .war et son déploiement dans Tomcat.

	Version utilisée dans cette section
Eclipse	2.1.2
J2RE	1.4.2_02
Plug−in Tomcat de Sysdeo	2.2.1 et 3.0

La page web du plug−in est consultable à l'url : http://www.sysdeo.com/eclipse/tomcatPluginFR.html.

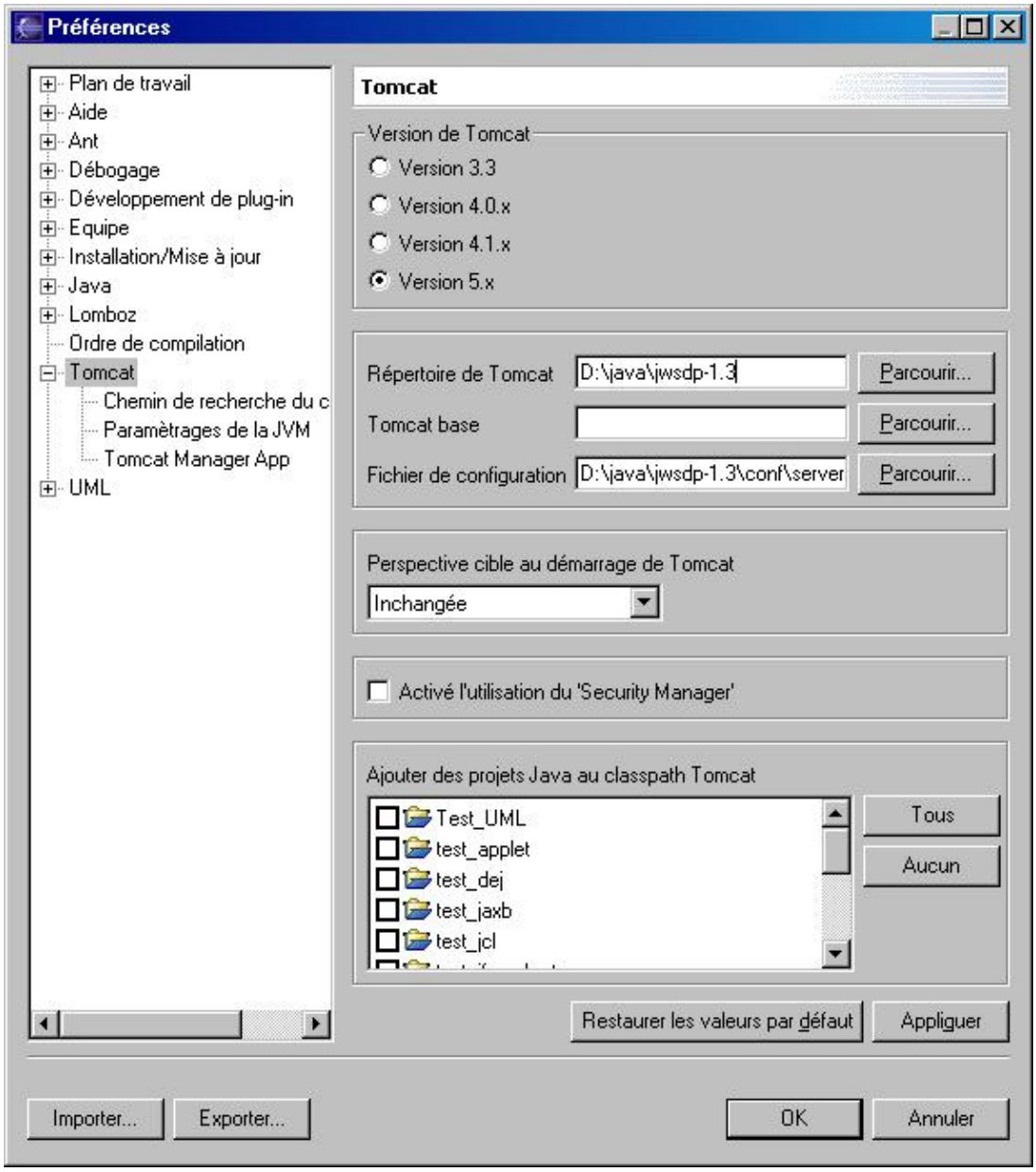
16.1.1. Installation et paramétrage de la version 2.2.1

Il faut télécharger le fichier tomcatPluginV221.zip et décompresser son contenu dans le répertoire plugins du répertoire d'installation d'Eclipse.

Il est possible d'ajouter un menu nommé « Tomcat » et trois boutons dans la barre d'outils pour démarrer, arrêter et redémarrer Tomcat :  . Pour cela, il faut utiliser l'option « Personnaliser la perspective ... » du menu « Fenetre » et cocher l'option « Autre / Tomcat » dans les éléments disponibles.

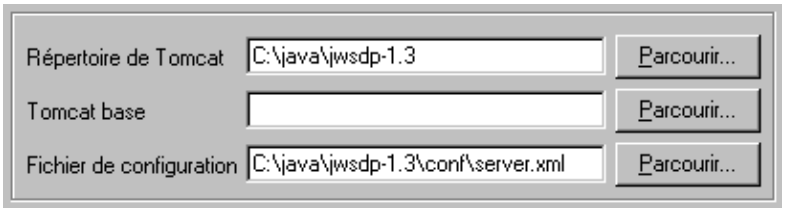
Pour pouvoir utiliser les fonctionnalités du plug−in, il faut le configurer notamment pour lui préciser ou se situe Tomcat. Cette configuration utilise plusieurs pages dans les préférences pour saisir les informations

utiles.

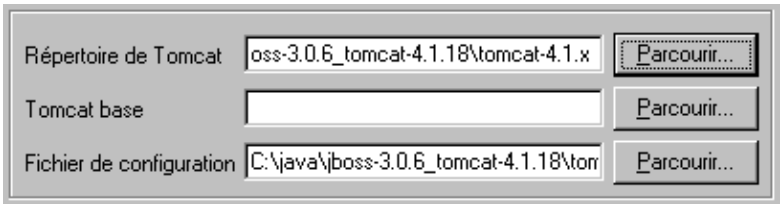


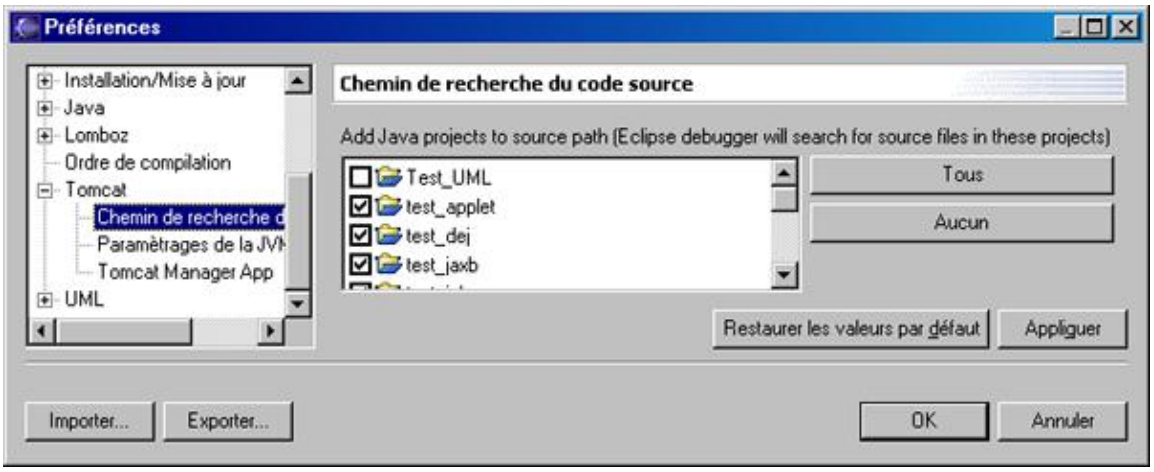
La première page permet de saisir les informations principales notamment la version de Tomcat utilisée et le répertoire de base de Tomcat.

Exemple avec Tomcat fourni avec le JWSDP 1.3

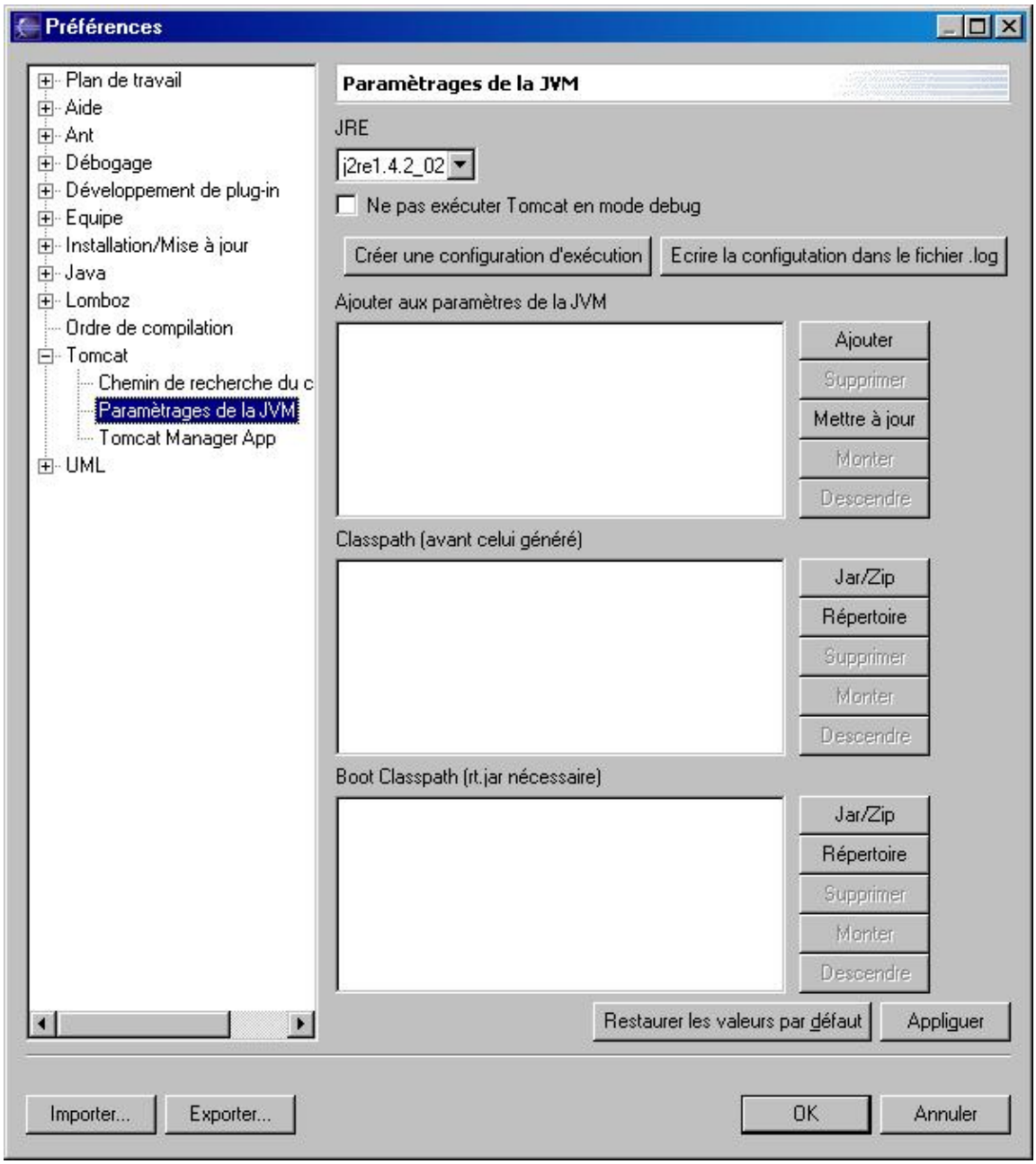


Exemple avec Tomcat fourni avec Jboss 3.0.6

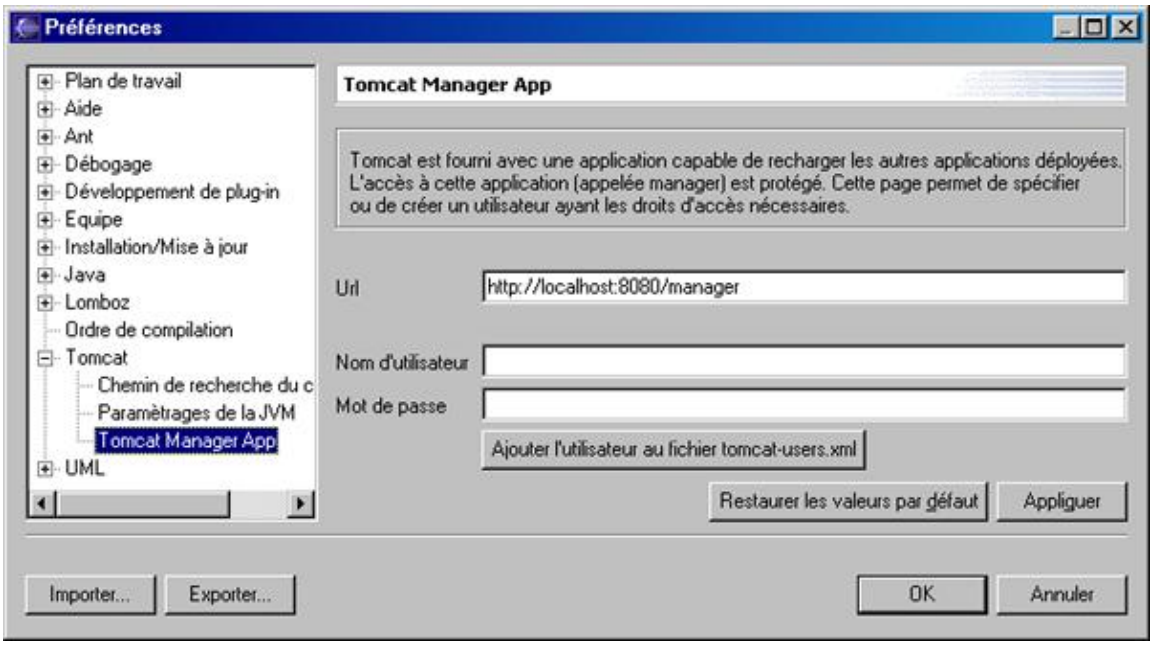




La seconde vue permet de préciser des paramètres pour la JVM dans laquelle va s'exécuter Tomcat.



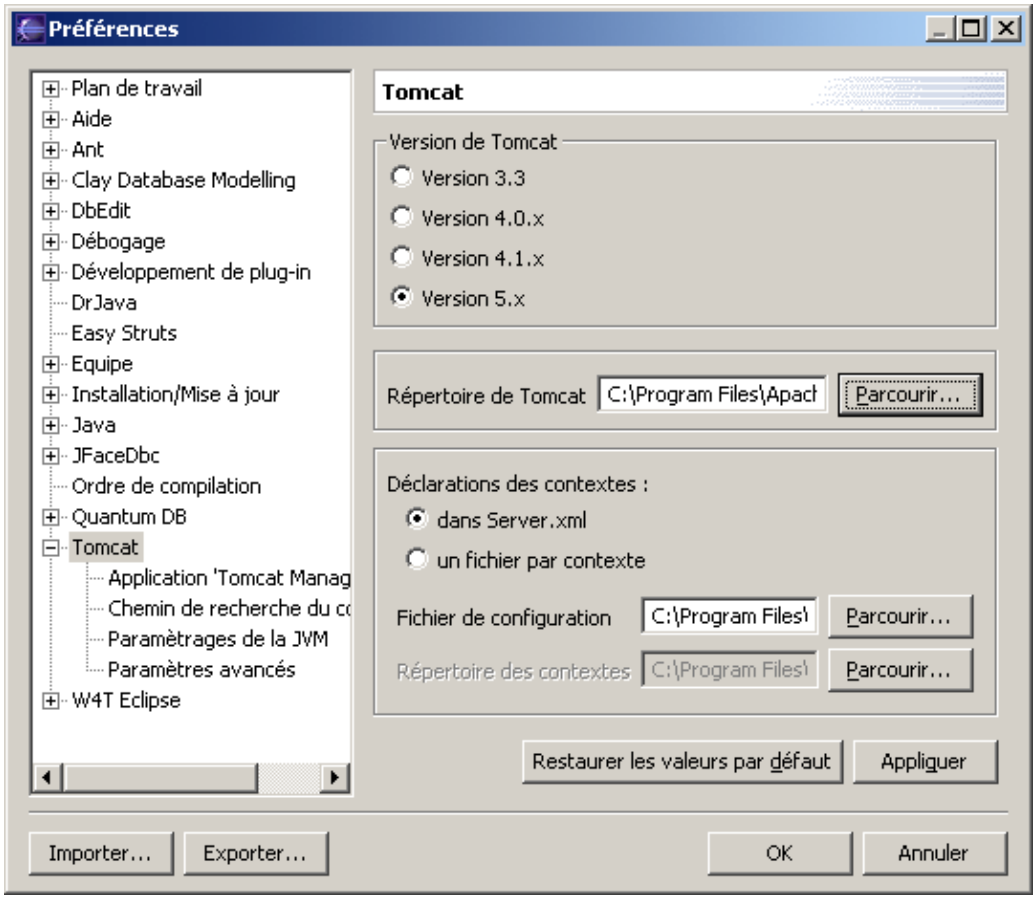
La troisième page permet de saisir les informations concernant le démarrage de l'application de gestion de Tomcat.



16.1.2. Installation et paramétrage de la version 3.0

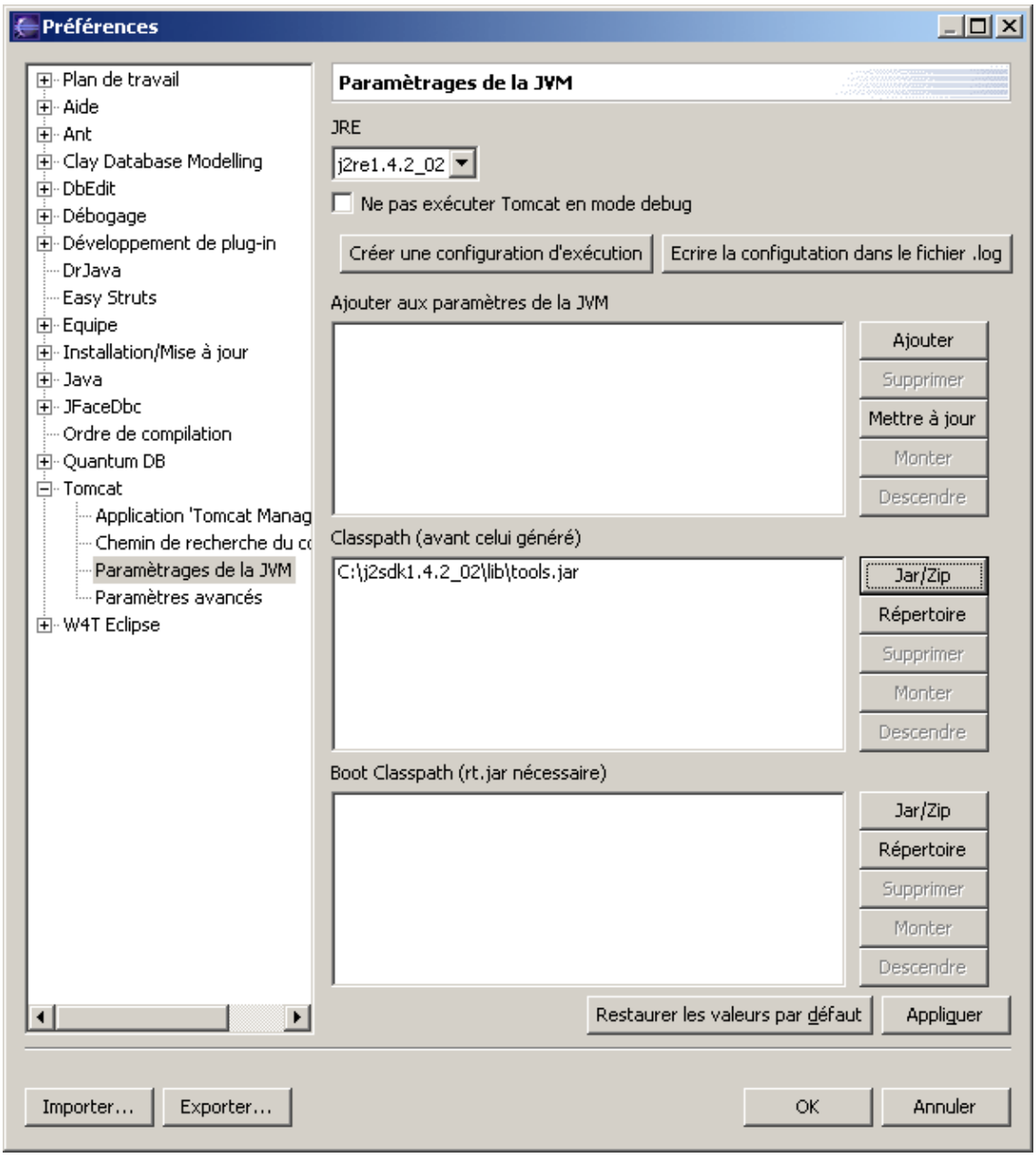
Il faut télécharger le fichier tomcatPluginV3.zip sur le site de Sysdeo : http://www.sysdeo.com/eclipse/tomcatPluginFR.html et décompresser son contenu dans le répertoire "plugins" du répertoire d'installation d'Eclipse.

Il faut ensuite lancer Eclipse et configurer le plug−in : sélectionnez le menu « Fenêtre/Préférences », puis l'option Tomcat dans l'arborescence.



Sur cette page, sélectionnez la version de Tomcat utilisée (version 5.x dans cette section) et sélectionnez le répertoire de Tomcat.

Sélectionnez l'option « Tomcat/Paramètrages de la JVM » et ajoutez dans la liste "Classpath" le fichier tools.jar qui se situe dans le répertoire lib du JDK.



Cliquez sur le bouton « OK »

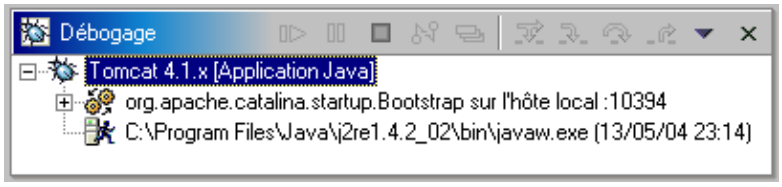
Sous la perspective « Java », sélectionnez l'option « Fenêtre / Personnaliser la perspective » puis dans l'arborescence, sélectionnez « Autres » et cochez la case « Tomcat ». Cliquez sur le bouton « OK » : les trois boutons permettant respectivement de démarrer, arrêter et redémarrer Tomcatcsont ajoutés dans la barre d'outils comme avec la version précédente du plug−in.

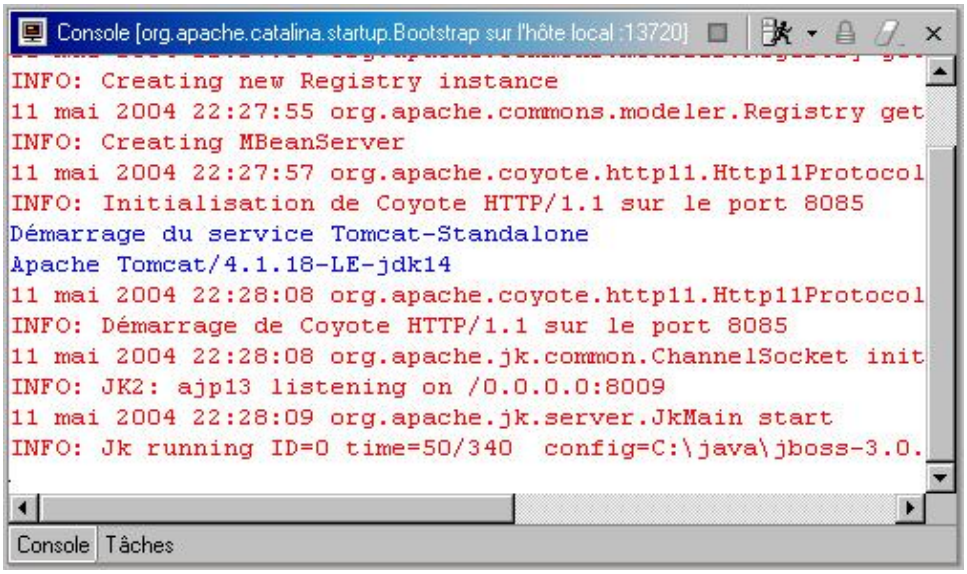
Cliquez sur le bouton de démarrage de Tomcat : la console affiche les messages de démarrage de l'application. Pour vérifier le bon fonctionnement, il suffit d'ouvrir un navigateur à l'url : http://localhost:8080

16.1.3. Lancement et arrêt de Tomcat

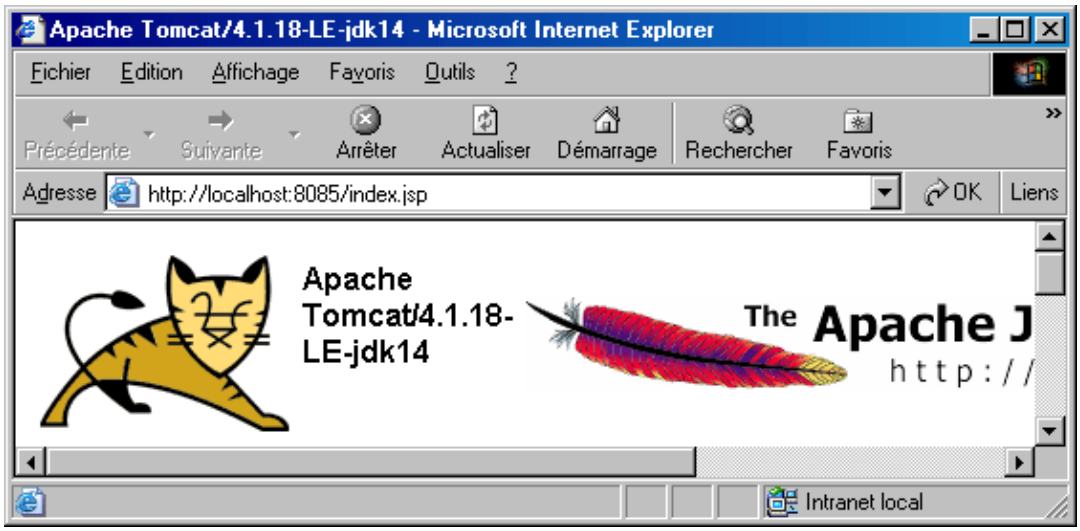
Pour lancer Tomcat, il suffit de cliquer sur le bouton de la barre d'outils ou d'utiliser l'option « Démarrer Tomcat » du menu « Tomcat ».

La perspective « Debug » s'affiche et les informations relatives au lancement de Tomcat s'affichent dans la vue « Console ».





Pour vérifier la bonne exécution de Tomcat, il suffit d'ouvrir un navigateur et de saisir l'url http://localhost avec le port précisé dans la configuration dans Tomcat.

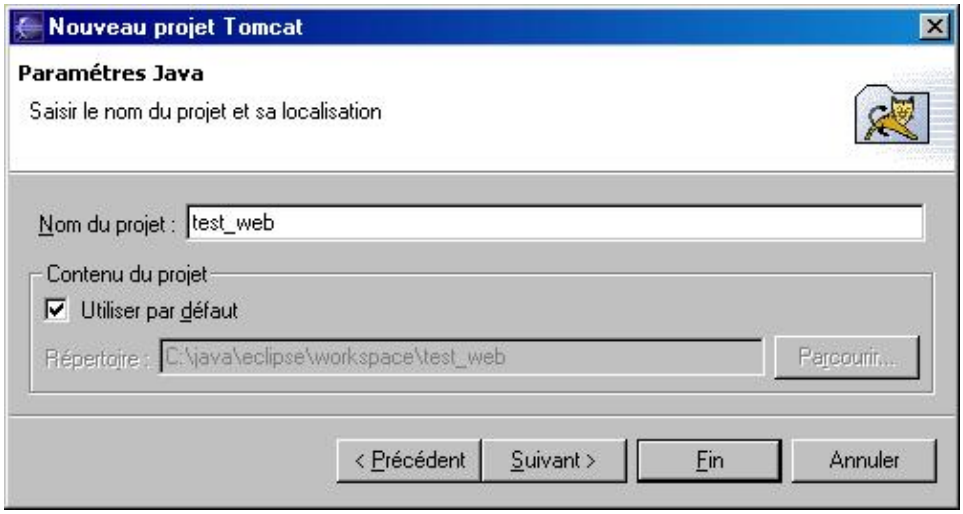


Pour arrêter Tomcat, il suffit de cliquer sur l'icône ou de sélectionner l'option « Arrêter Tomcat » du menu « Tomcat ».

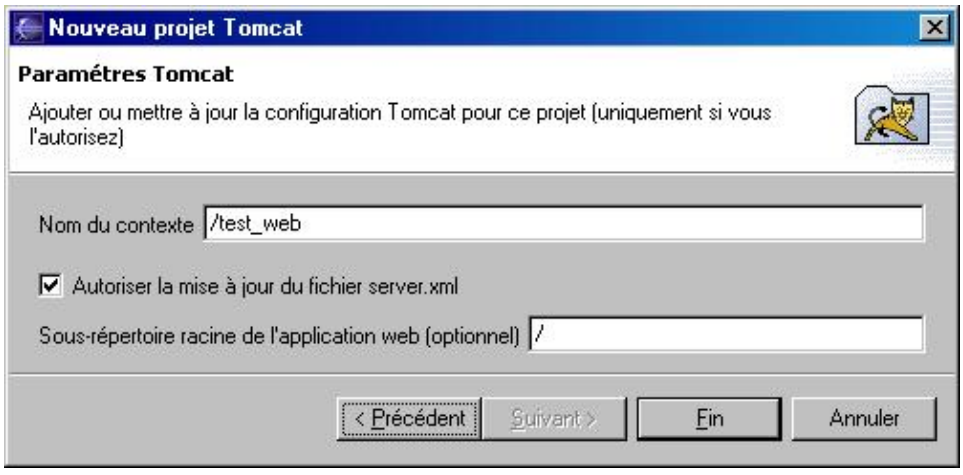
16.1.4. La création d'un projet Tomcat

Le plug−in permet la création d'un projet particulier pour développer des applications web.

Cliquez sur le bouton « Suivant »

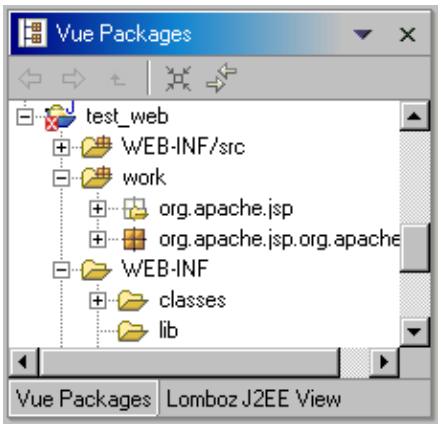


Saisissez le nom du projet et cliquez sur le bouton « Suivant ».



La page suivante de l'assistant permet de préciser certains paramètres : le nom du contexte de l'application web (par défaut, celui du projet), le répertoire racine de l'application et permettre au plug−in de modifier ou non le fichier server.xml de Tomcat.

En cliquant sur le bouton « Fin », le projet est créé.



La structure du projet reprend celle d'une application web incluse dans un fichier .war. Le fait d'autoriser le plug−in à modifier le fichier server.xml provoque sa mise à jour pour ajouter un nouveau contexte :

Exemple :

<Context path="/test_web" reloadable="true" docBase="D:\java\eclipse\workspace\test_web" workDir="D:\java\eclipse\workspace\test_web\work\org\apache\jsp" />

Si Tomcat était déjà en cours d'exécution, il faut le relancer pour que les modifications soient prises en compte.

16.2. Lomboz

Lomboz est un plug−in dont le but est de faciliter le développement d'applications J2EE 1.3. Initialement développé par la société ObjectLearn (http://www.objectlearn.com/index.jsp), son statut est passé à open−source depuis l'intégration du projet au consortium ObjectWeb, début 2004 (http://forge.objectweb.org/projects/lomboz) .

Ce plug−in utilise plusieurs outils open source pour mener à bien différentes tâches : Ant, Xdoclet, Axis, ... ce qui lui permet de couvrir le cyle de développement des applications J2EE : rédaction et génération du code, déploiement et débogage.

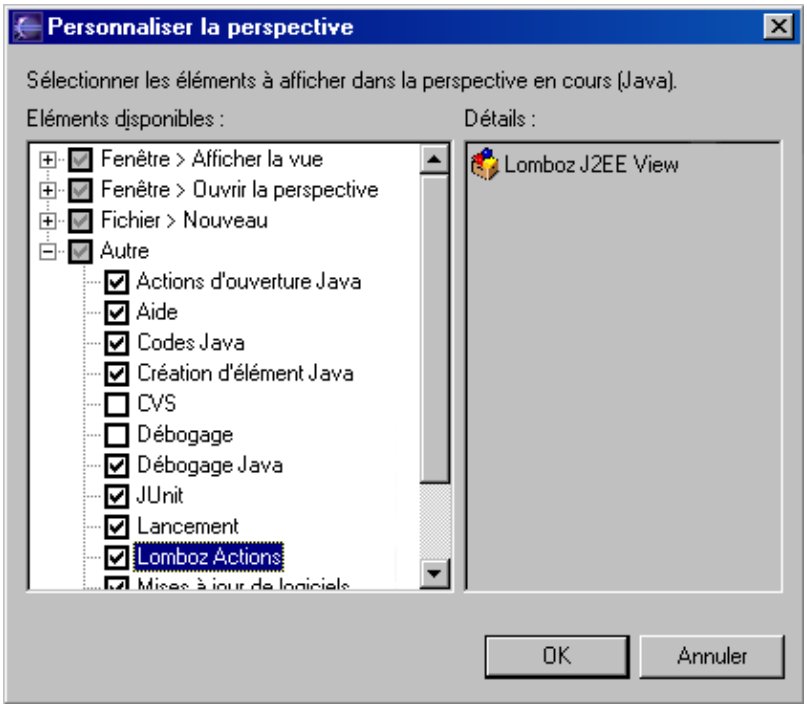
Cette section va utiliser Lomboz avec JBoss et Tomcat.

	Version utilisée dans cette section
Eclipse	2.1.2
J2RE	1.4.2_02
Lomboz	2.1.2
JBoss	3.0.6
Tomcat	4.1.18

16.2.1. Installation et configuration

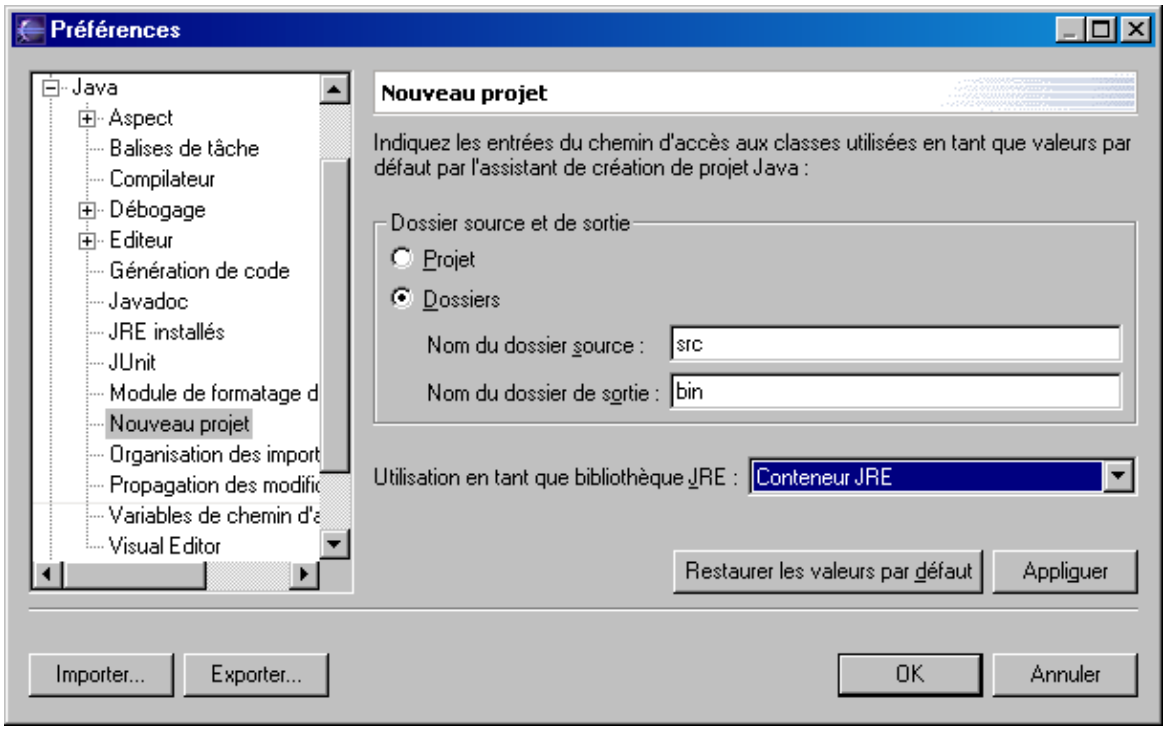
Téléchargez le fichier lomboz.21_02[1].zip et le décompresser dans le répertoire d'installation d'Eclipse. L'installation de Lomboz se fait comme pour les autres plug−ins en décompressant l'archive dans le répertoire d'Eclipse.

Pour configurer Lomboz, il faut utiliser l'option « Fenêtre / Personnaliser la perspective ... »

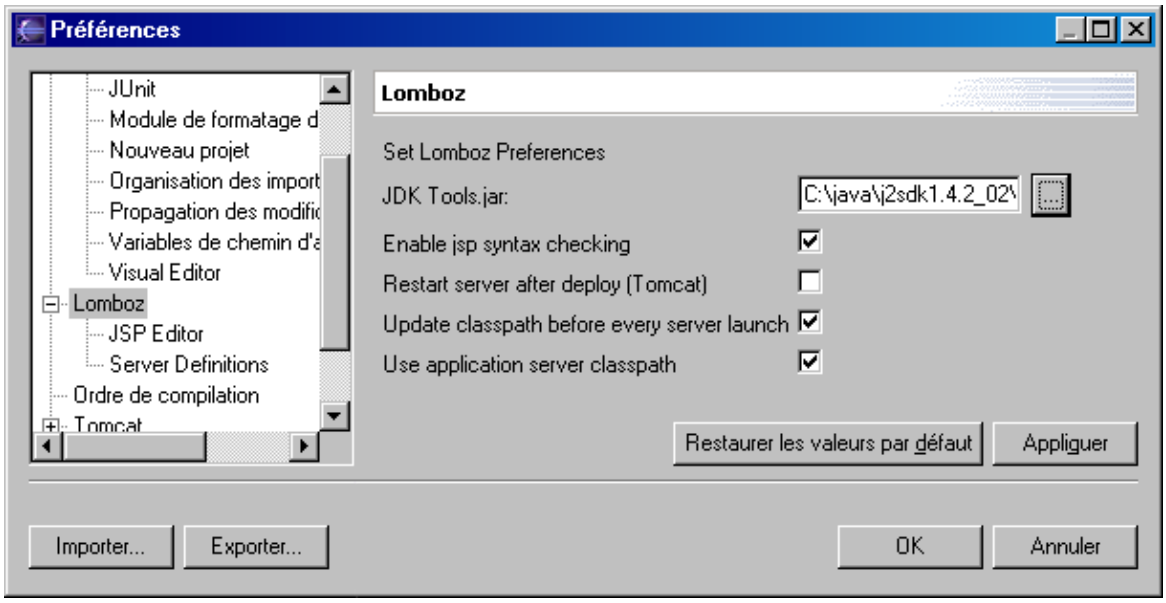


Cliquez sur la case à cocher « Autre / Lomboz Actions » puis sur le bouton « OK ». Un bouton supplémentaire apparaît dans la barre d'outils :  . Il permet d'afficher la vue « Lomboz ».

Dans les préférences, sélectionnez « Java / Nouveau projet », puis cliquez sur le bouton radio « Dossiers » en conservant les noms des dossiers « src » et « bin ».

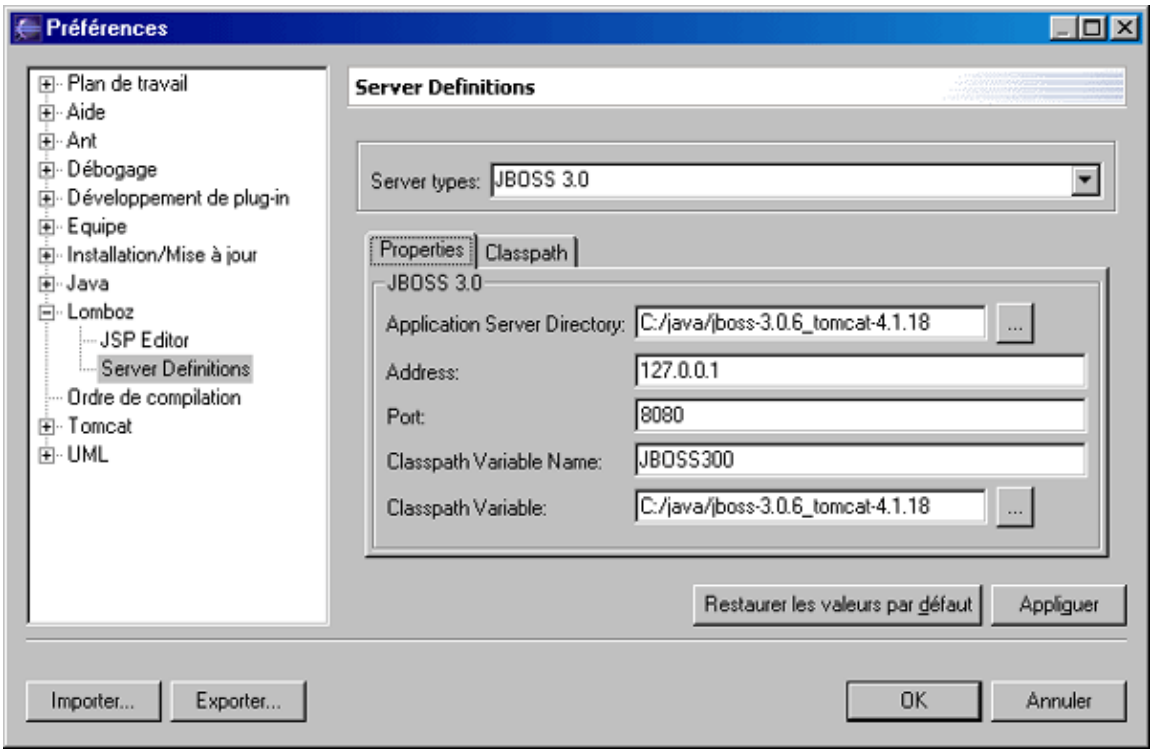


Toujours dans les préférences, il faut sélectionner « Lomboz » et vérifier que le chemin désignant le fichier tools.jar pointe bien vers le fichier $JAVA_HOME/lib/tools.jar.



Sélectionnez « Lomboz / Server Definitions » puis sélectionnez le ou les types de serveur utilisés : dans l'exemple de cette section, « JBOSS 3.0 » et « Apache Tomcat v4.1.0 ».

Il faut donc sélectionner le type « JBOSS 3.0 » et modifier les propriétés pour refléter celle du système utilisé notamment « Application server directory » et « Classpath variable » qui doivent pointer vers le répertoire d'installation de JBoss. Il suffit ensuite de cliquer sur le bouton « Appliquer ».



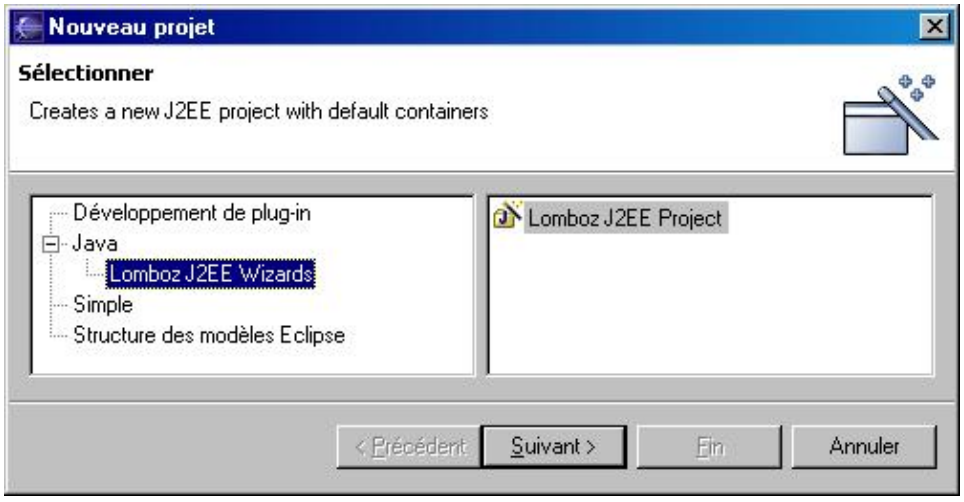
Pour s'assurer de la bonne configuration, il suffit de cliquer sur l'onglet de « Classpath » et de vérifier qu'aucune bibliothèque n'est erronée.

Il faut sélectionner le type de serveur « Apache Tomcat v4.1.0 » et faire de même.

Pour valider les modifications, il suffit de cliquer sur le bouton « OK »

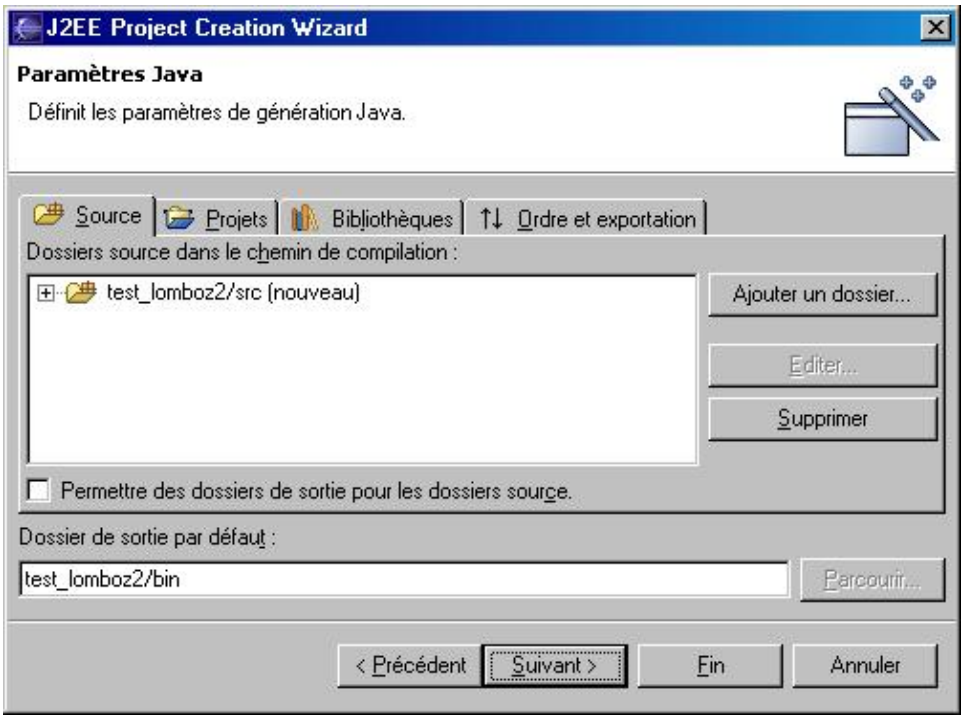
16.2.2. Création d'un nouveau projet

Pour créer un nouveau projet « Lomboz », il suffit de créer un nouveau projet de type « Java / Lomboz J2EE Wizard / Lomboz J2EE Project »



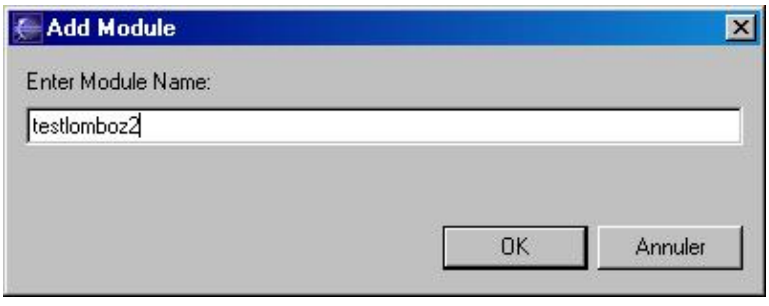
Cliquez sur le bouton « Suivant » et saisissez le nom du projet

Cliquez sur le bouton « Suivant » et s'assurer que les répertoires sources et de sortie sont désignés respectivement par les répertoires "src" et "bin".



Cliquez sur le bouton « Suivant ». Sur l'onglet « Web Modules », il faut ajouter un nouveau module.

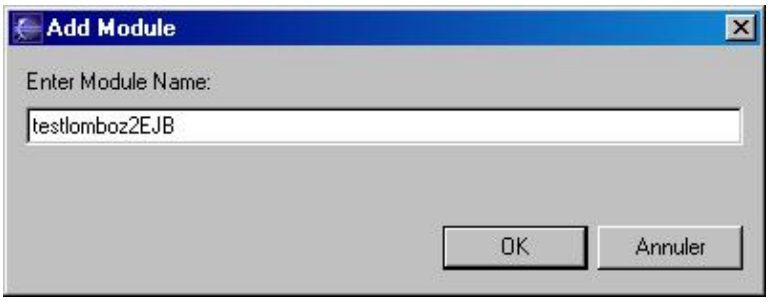
Cliquez sur le bouton « Add »



Saisissez le nom et cliquez sur le bouton « OK »

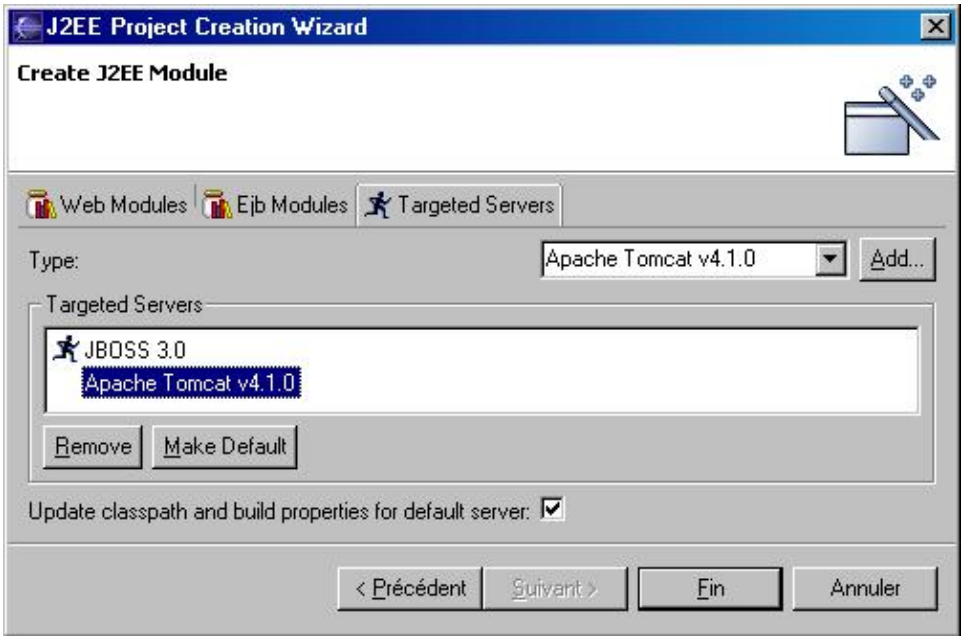
Sur l'onglet « EJB Modules », il est possible d'ajouter au projet un ou plusieurs modules qui vont contenir des EJB.

Le principe pour ajouter un module EJB est identique à celui utilisé pour un module web : cliquer sur le bouton « Add », saisir le nom du module dans la boîte de dialogue et cliquer sur le bouton « OK ».



L'ajout d'autres modules web ou EJB est toujours possible après la création du projet en utilisant l'assistant de création de nouvelles entités.

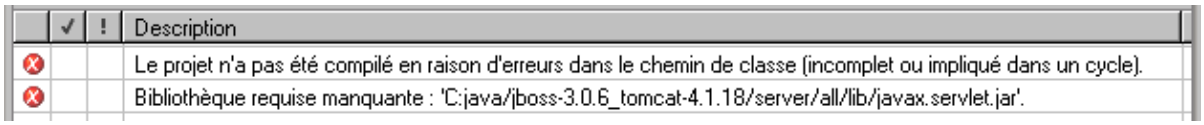
Sur l'onglet « Targeted Servers », il faut sélectionner le type de serveur à utiliser et cliquer sur le bouton “Add”. Dans l'exemple de cette section, il faut ajouter les serveurs « JBOSS 3.0 » et « Apache Tomcat v4.1.0 ».



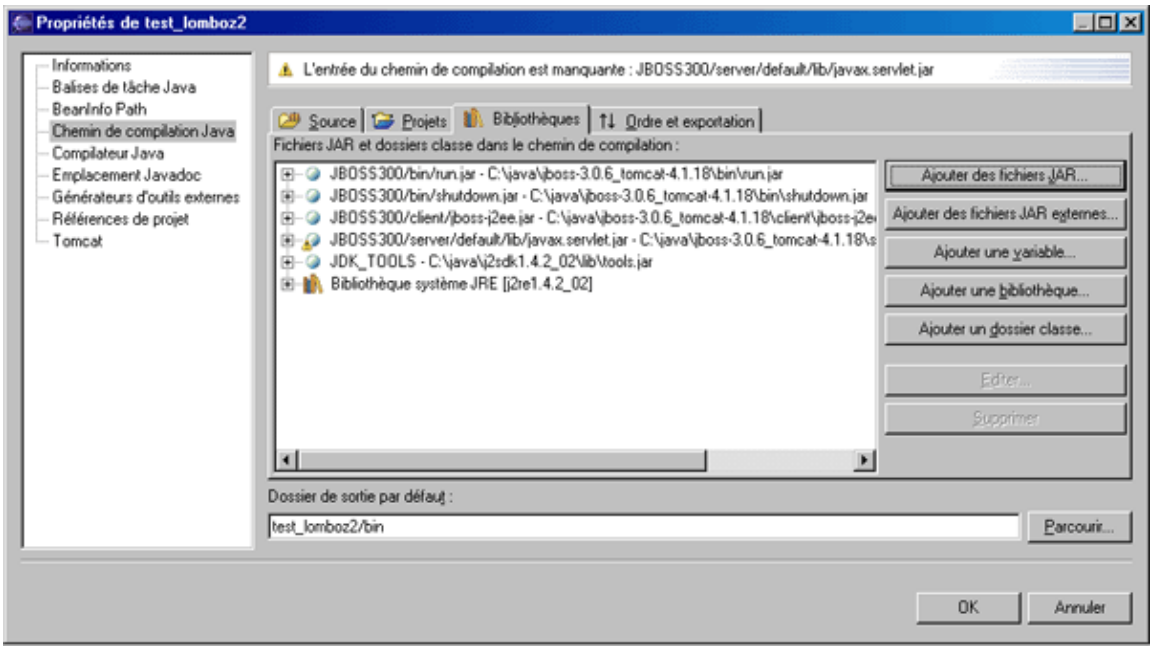
Pour créer le projet, il suffit de cliquer sur le bouton « Fin ».

Lors de l'affichage de ce message, il suffit de cliquer sur le bouton « Oui » pour ouvrir la perspective « Java ».

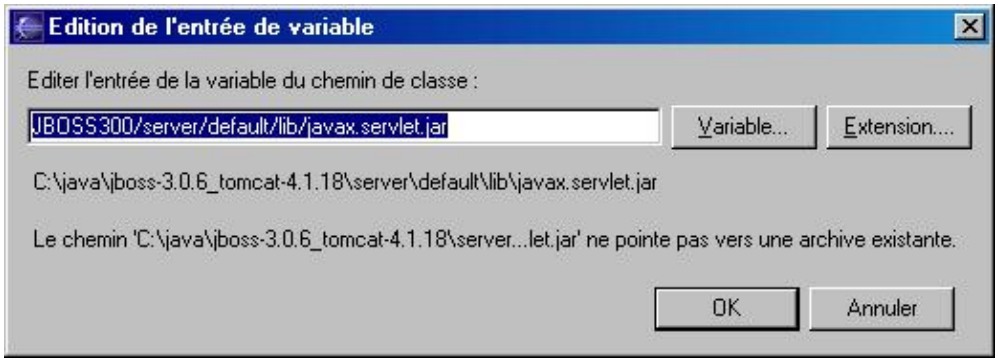
Des erreurs seront signalées dans la vue « Taches » car le chemin désigné pour le fichier servlet.jar est erroné.



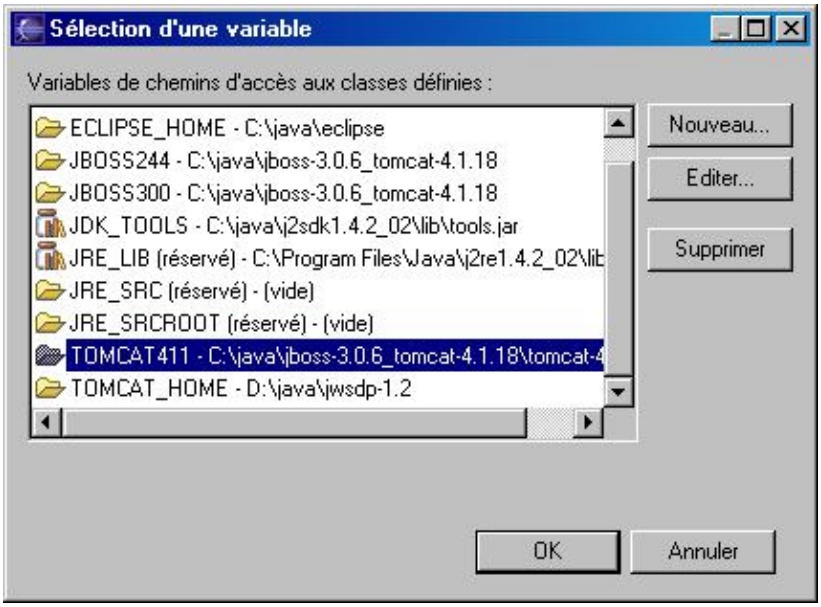
Pour corriger le problème, il faut modifier le chemin de la bibliothèque dans les propriétés du projet.



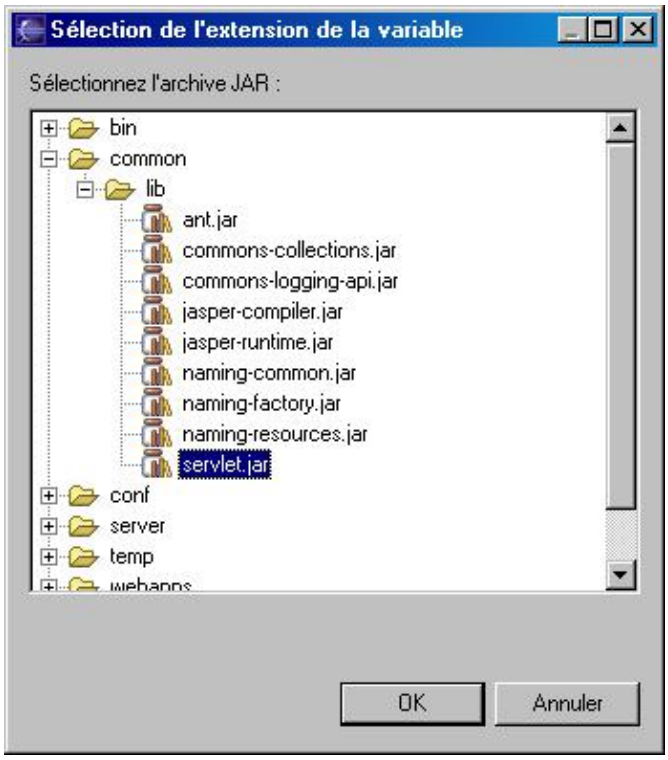
Il faut sélectionner la ligne correspondant au fichier servlet.jar (celle contenant une petite icône attention jaune) et cliquer sur le bouton « Editer ... ».



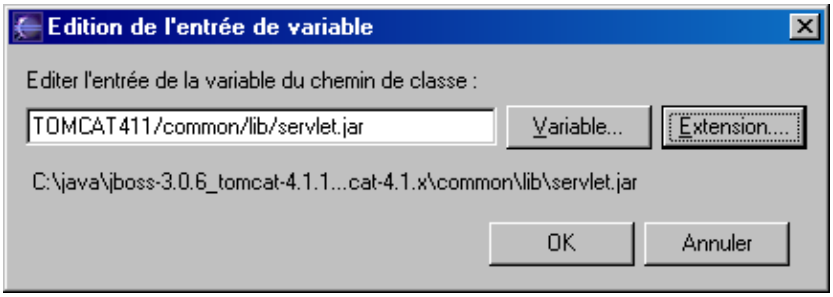
Pour faciliter la saisie du chemin il est possible de cliquer sur le bouton « Variable » pour sélectionner « TOMCAT411 » puis de cliquer sur le bouton « OK »



Pour faciliter la saisie du reste du chemin, il faut cliquer sur le bouton « Extension », sélectionner le chemin du fichier servlet.jar et cliquer sur le bouton « OK ».



La valeur associée à la variable doit avoir une valeur semblable à celle ci dessous :



Il suffit de cliquer sur le bouton « OK » pour valider les modifications.

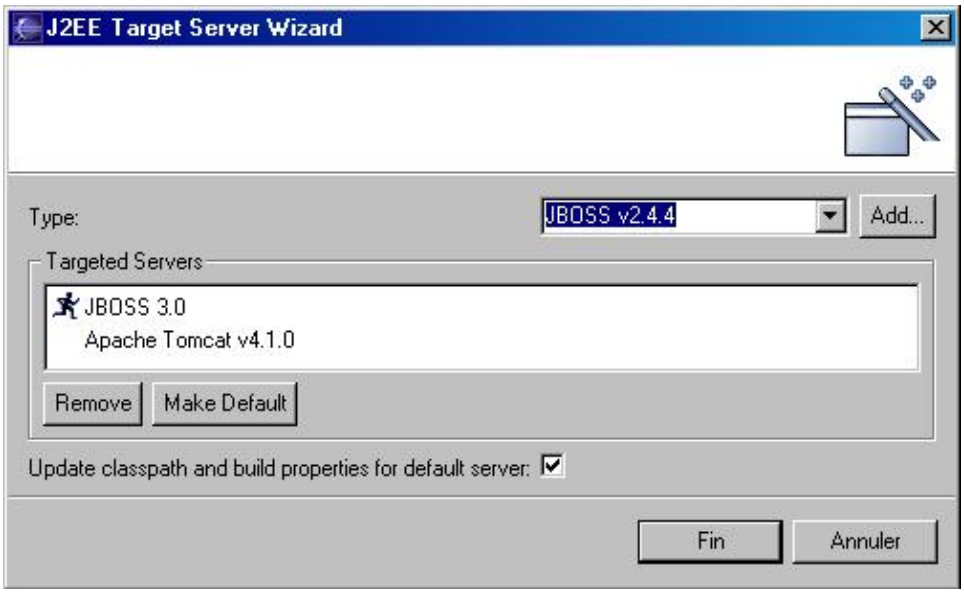
Cette correction n'est que temporaire : pour corriger le problème de façon définitive, il faut modifier le fichier .server correspondant à celui du serveur utilisé dans le répertoire :

C:\java\eclipse\plugins\com.objectlearn.jdt.j2ee\servers

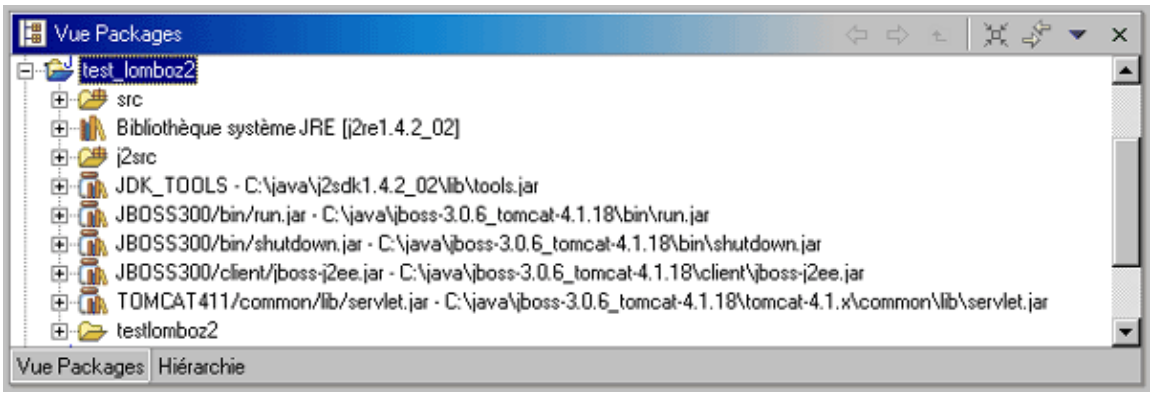
La ligne a modifier est celle permettant l'ajout du fichier servlet.jar.

Une fois le fichier enregistré, il faut relancer Eclipse.

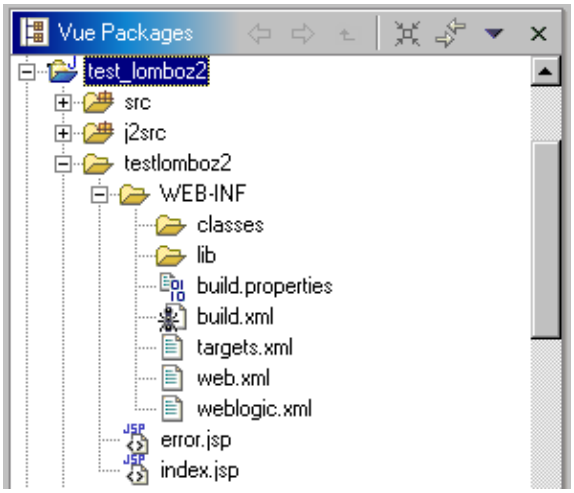
Dans la vue package, sur le module EJB, sélectionnez l'option « Lomboz J2EE/ Change default server » pour prendre en compte la modification puis cliquez sur le bouton "Fin".

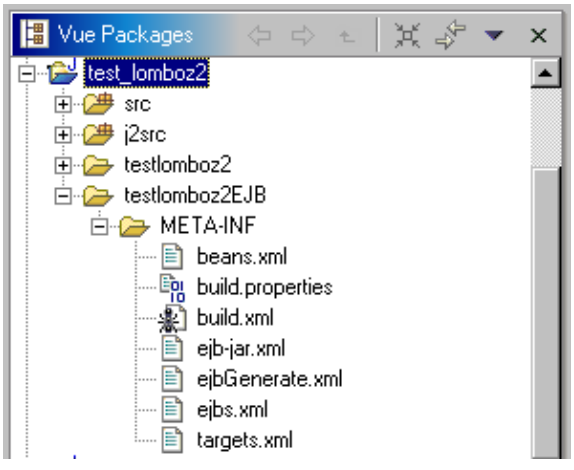


La structure du projet est la suivante dans la vue package :



Les modules web et EJB contiennent déjà quelques fichiers de configuration.





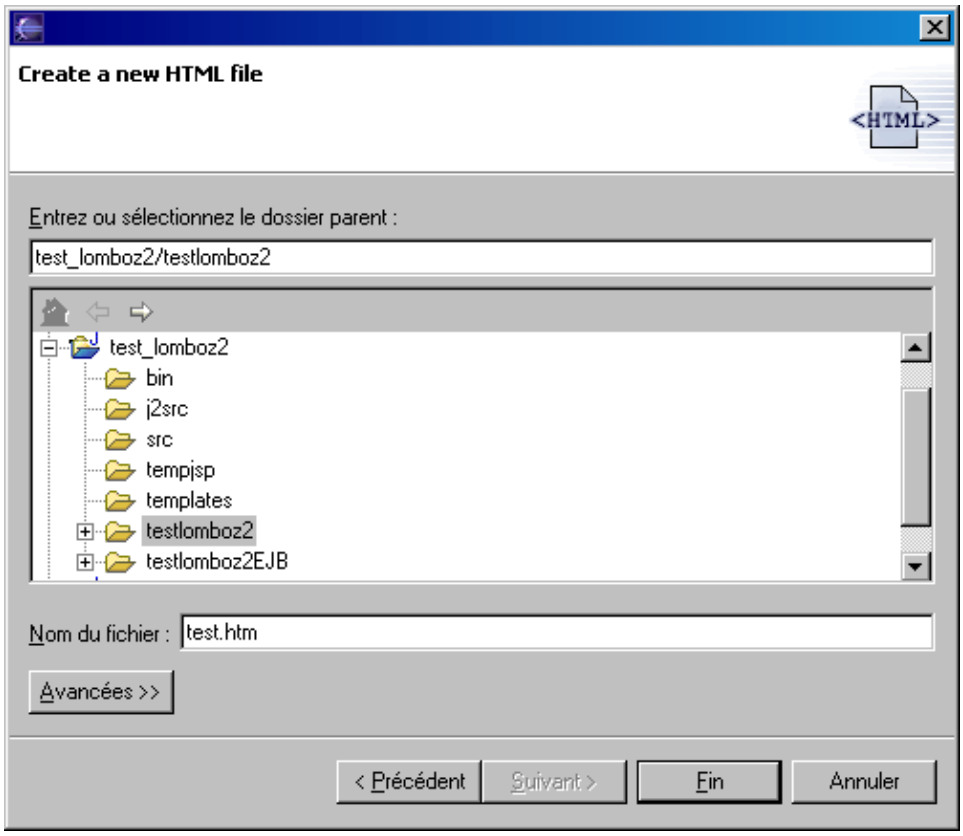
16.2.3. Création d'une webapp

Une webapp est contenue dans un module web. Si un tel module n'a pas été ajouté lors de la création du projet ou si il est nécessaire d'en ajouter un nouveau, il faut créer une nouvelle entité de type "Autre / Java / Lomboz J2EE Wizards / Lomboz J2EE Module".

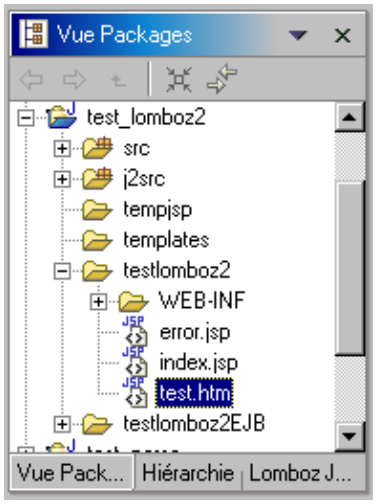
Sur l'onglet « Web module » de la page de l'assistant, il suffit de suivre les indications précisées dans la section de création d'un nouveau projet pour ajouter un nouveau module web.

16.2.4. Ajouter un fichier HTML à une webapp

Il faut créer une nouvelle entité du type « Autres / Java / Lomboz J2EE Wizards / Lomboz HTML Wizard ».

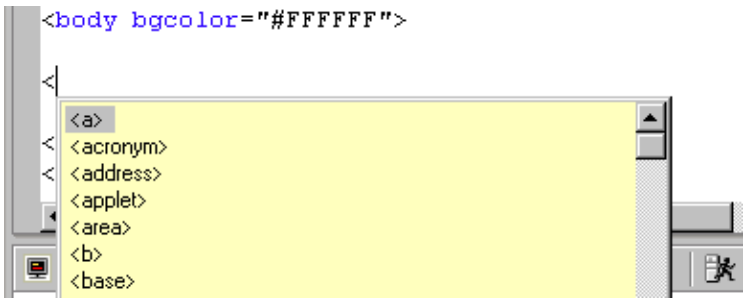


Il faut sélectionner le répertoire ou sera stocké le fichier, saisir le nom du fichier et cliquer sur le bouton « Fin ».

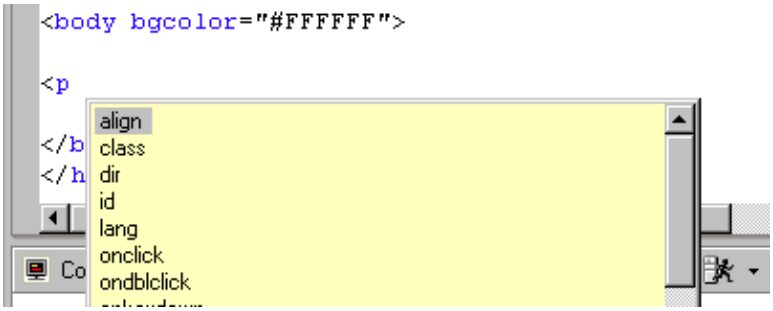


Le fichier créé est ouvert dans un éditeur dédié.

L'éditeur propose une assistance à la rédaction du code pour les tags en appuyant sur les touches "Ctrl+Espace".

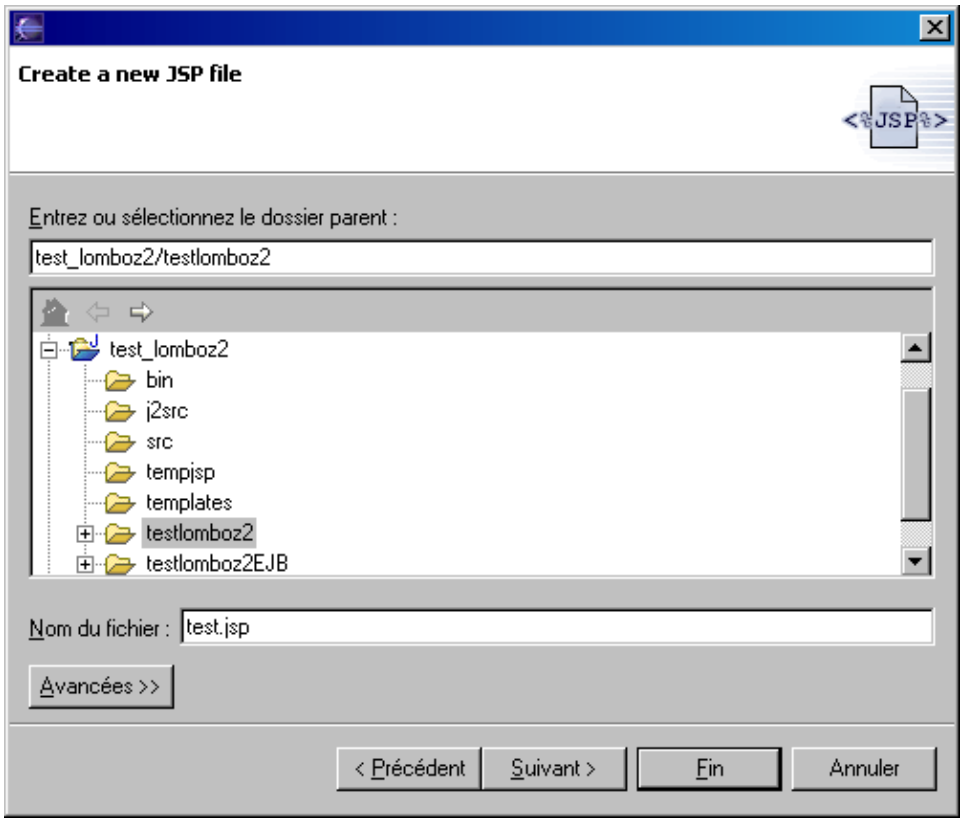


Il propose aussi une assistance pour la saisie des attributs d'un tag.

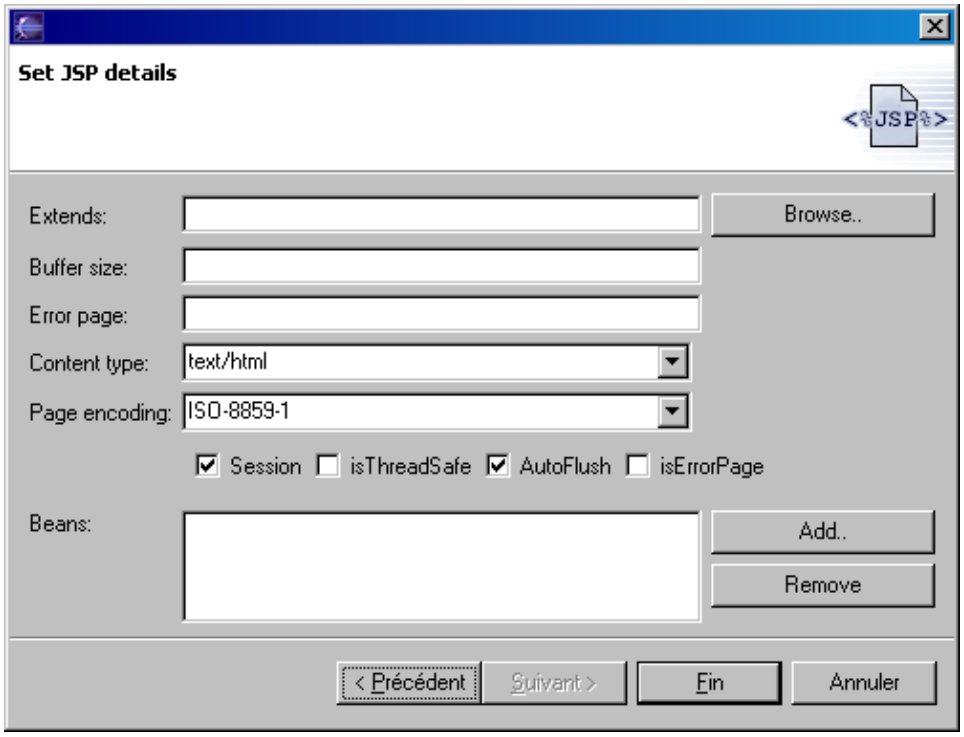


16.2.5. Ajouter une JSP à une webapp

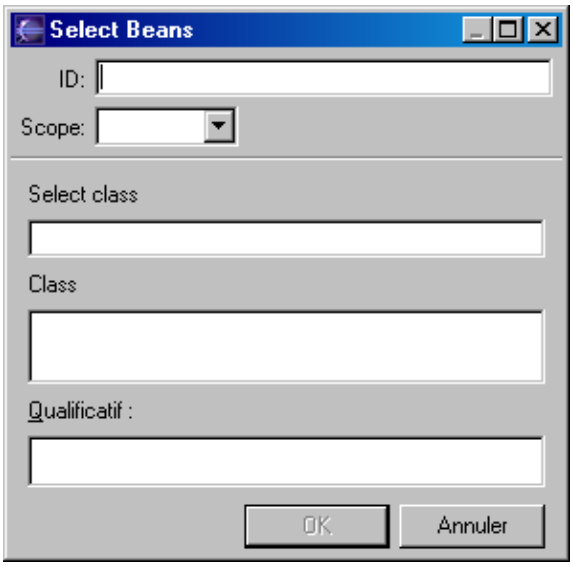
Il faut créer une nouvelle entité du type « Autres / Java / Lomboz J2EE Wizards / Lomboz JSP Wizard ».



Il faut sélectionner le répertoire, saisir le nom de la JSP et cliquer sur le bouton « Suivant ».

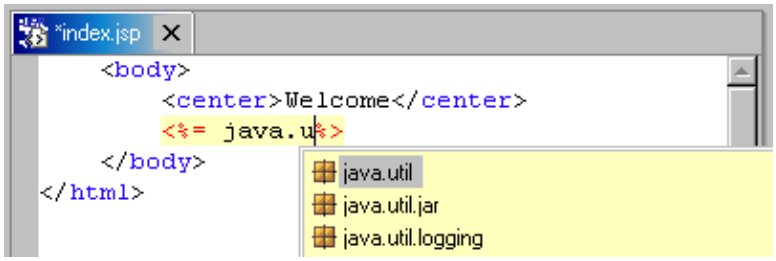


La page suivante de l'assistant permet de saisir des informations concernant la nouvelle JSP. Un clic sur le bouton « Add » permet d'ouvrir une boite de dialogue demandant les paramètres du bean à ajouter :

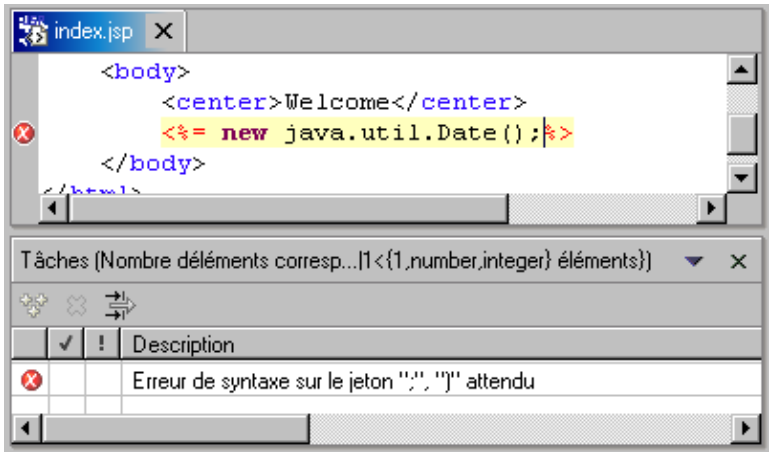


Il suffit de cliquer sur le bouton « Fin » pour générer la fichier et l'ouvrir dans l'éditeur.

L'éditeur propose une coloration syntaxique des éléments de la JSP et une assistance à la rédaction du code en appuyant sur les touches "Ctrl+Espace".



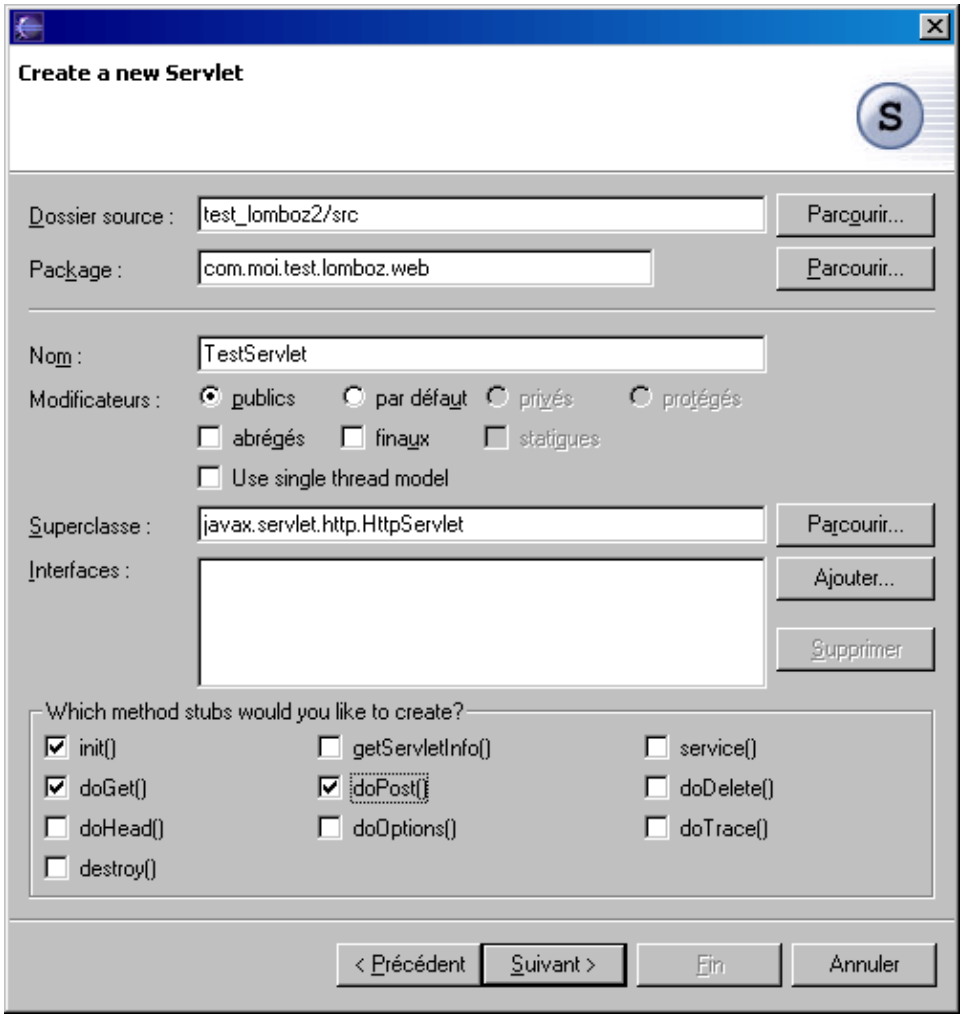
Lors de la sauvegarde du fichier, la JSP est compilée et les éventuelles erreurs sont signalées :



Il suffit alors de corriger les erreurs signalées et de sauvegarder le fichier.

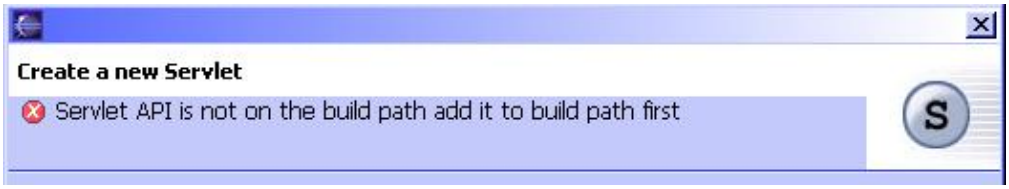
16.2.6. Ajouter une servlet à une webapp

Il faut créer une nouvelle entité du type « Autres / Java / Lomboz J2EE Wizards / Lomboz Servlet Wizard ».

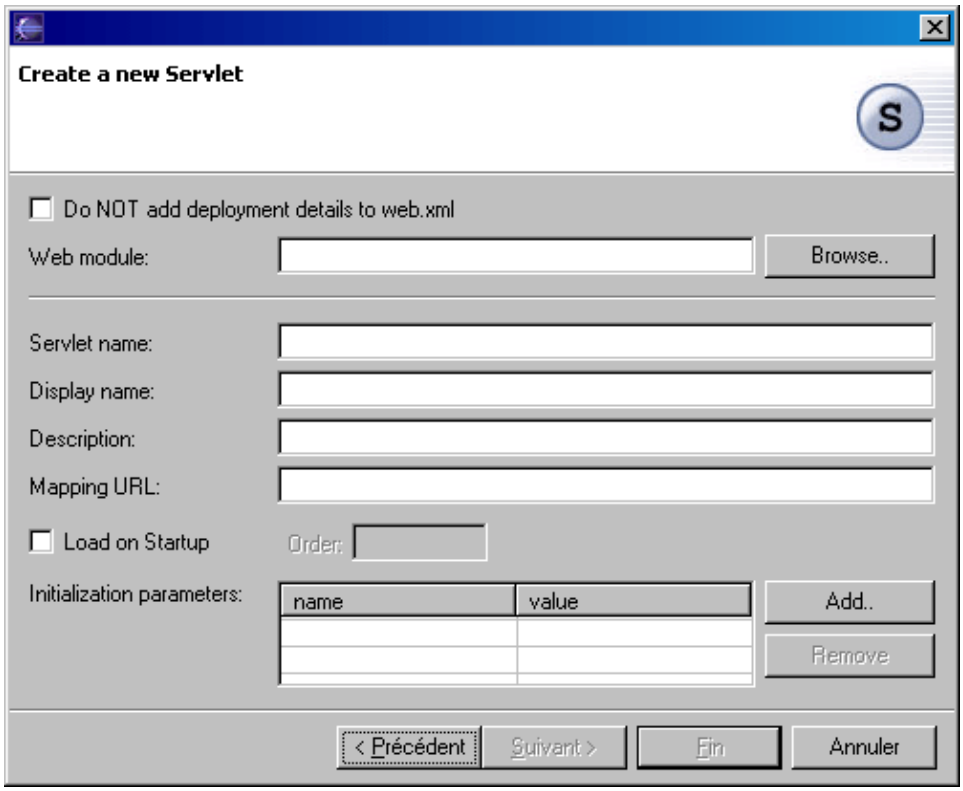


La première page de l'assistant permet de saisir les informations générales sur la nouvelle servlet notamment son package, son nom, ces modificateurs et les méthodes qui doivent être générées.

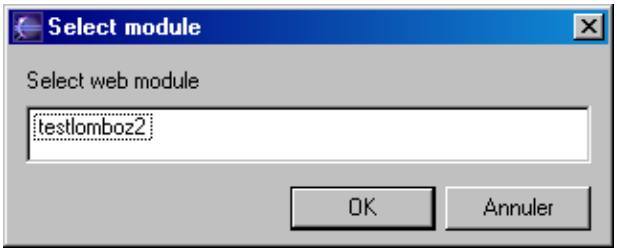
Pour créer une servlet, il faut absolument que le fichier servlet.jar soit dans le classpath du projet, sinon un message d'erreur est affiché.



Une fois toutes les informations saisies, il suffit de cliquer sur le bouton « Suivant » pour accéder à la seconde page de l'assistant.



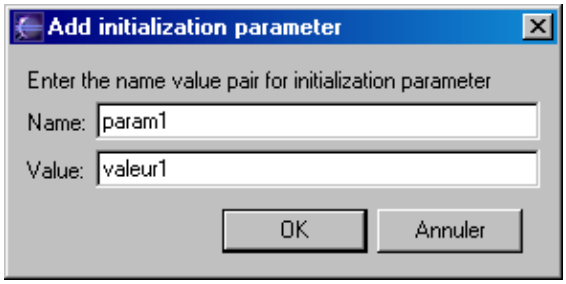
Cette page permet de saisir les informations de la servlet pour enrichir le fichier web.xml. Pour sélectionner le module web qui va contenir la servlet, il faut cliquer sur le bouton « Browse » :

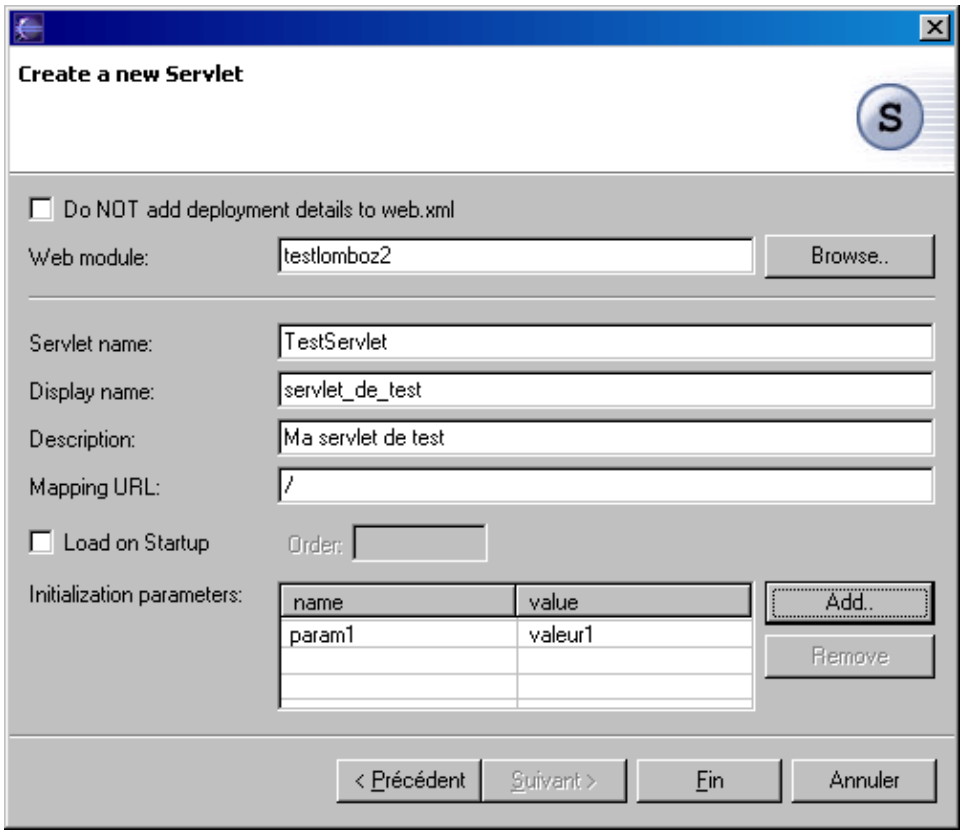


Il suffit alors de sélectionner le module web et de cliquer sur le bouton « Ok ».

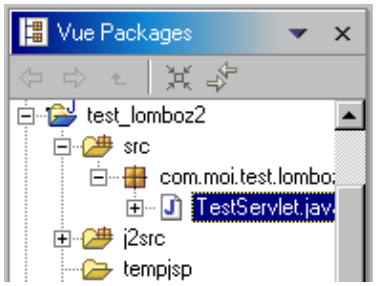
La saisie du nom de la servlet et du mapping URL est obligatoire.

Pour ajouter un paramètre à la servlet, il suffit de cliquer sur le bouton « Add » et de renseigner le nom et la valeur du paramètre dans la boîte de dialogue qui s'ouvre puis de cliquer sur le bouton « OK ».

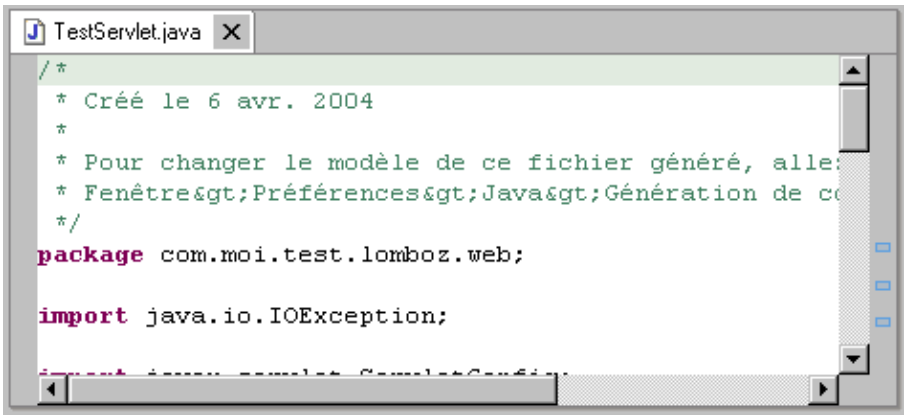




Pour créer la servlet, il suffit enfin de cliquer sur le bouton « Fin ».



L'éditeur de code s'ouvre avec le source de la servlet créée.



Le fichier WEB−INF/web.xml est automatiquement modifié par Lomboz pour tenir compte des informations concernant la nouvelle servlet :

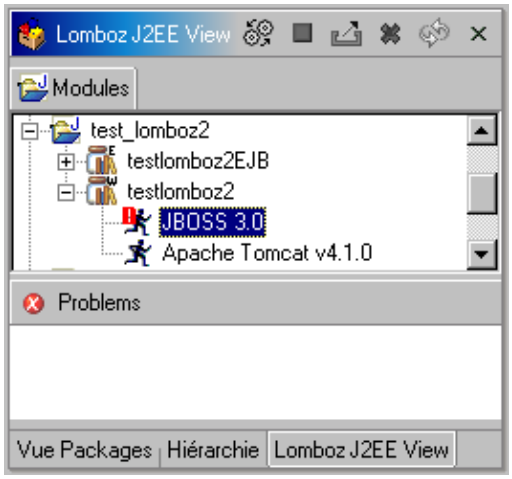
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
                         "http://java.sun.com/j2ee/dtds/web-app_2_3.dtd">
<!-- Copyright (c) 2002 by ObjectLearn. All Rights Reserved. -->
<web-app>
```

```
    <servlet>
        <servlet-name>TestServlet</servlet-name>
        <servlet-class>com.moi.test.lomboz.web.TestServlet</servlet-class>
        <display-name>servlet_de_test</display-name>
        <description>Ma servlet de test</description>
        <init-param>
            <param-name>param1</param-name>
            <param-value>valeur1</param-value>
        </init-param>
    </servlet>
    <servlet-mapping>
        <servlet-name>TestServlet</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>
    <welcome-file-list>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>
    <error-page>
        <error-code>404</error-code>
        <location>/error.jsp</location>
    </error-page>
</web-app>
```

16.2.7. Tester une webapp

Pour tester une webapp, il faut démarrer le serveur d'application et déployer l'application sur le serveur.

Pour cela, il faut utiliser la vue « Lomboz » en appuyant sur le bouton  dans la barre d'outils.



Cette vue possède plusieurs boutons pour réaliser certaines taches :

 Lancer le serveur

 Arrêter le serveur

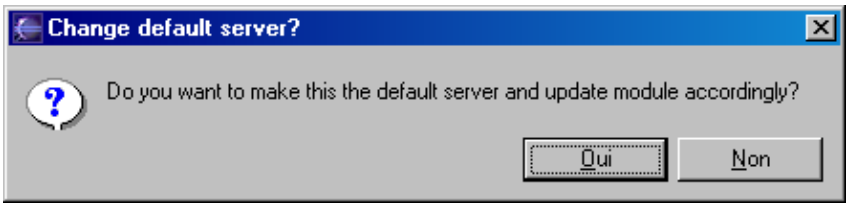
 Déployer le module dans le serveur

 Supprimer le module du serveur

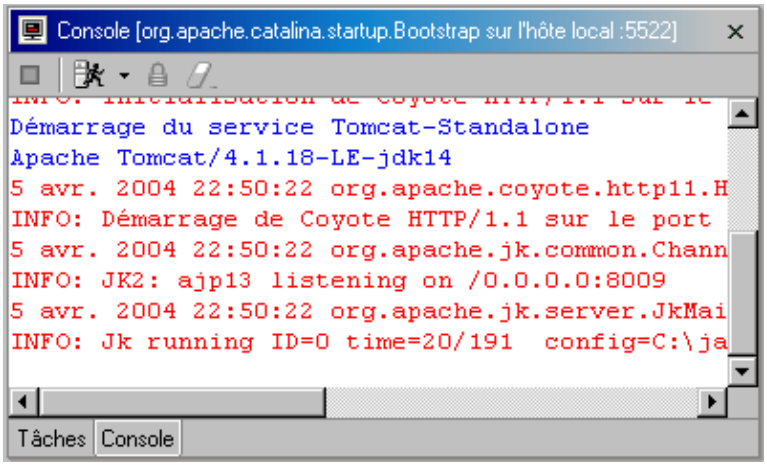
 Permet de rafraîchir la vue

Pour lancer un serveur, il suffit de le sélectionner dans l'arborescence et de cliquer sur le bouton correspondant ou d'utiliser l'option « Debug server » ou « Run server » du menu contextuel.

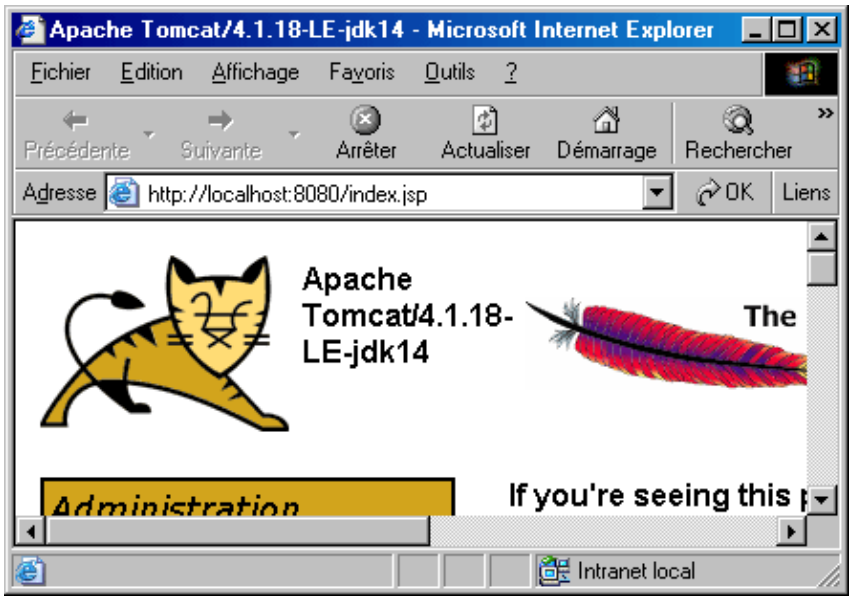
Lors du premier lancement, un boîte de dialogue demande si le serveur doit devenir le serveur par défaut du module.

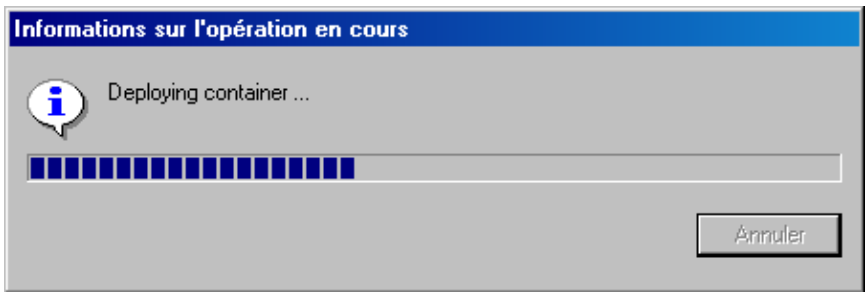


La progression du démarrage du serveur web Tomcat est affichée dans la vue « Console ».

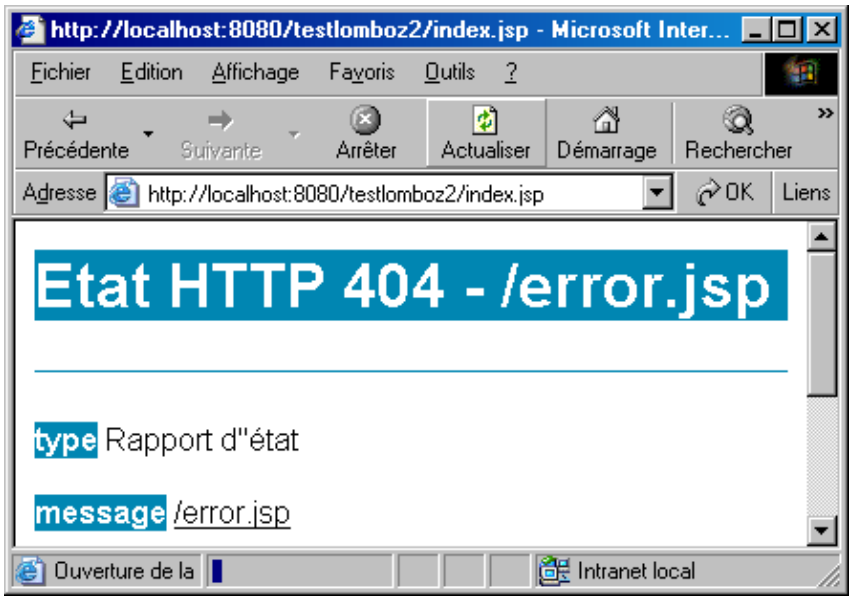


Pour vérifier que Tomcat est lancé, il suffit d'ouvrir un navigateur et de saisir l'url http://localhost:8080

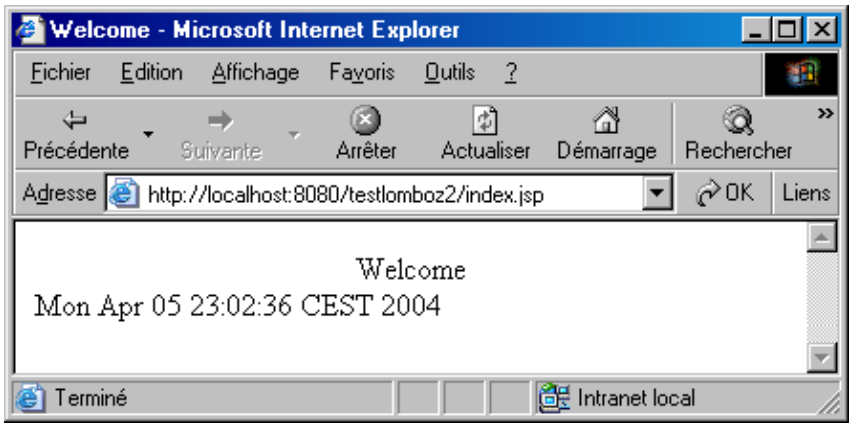




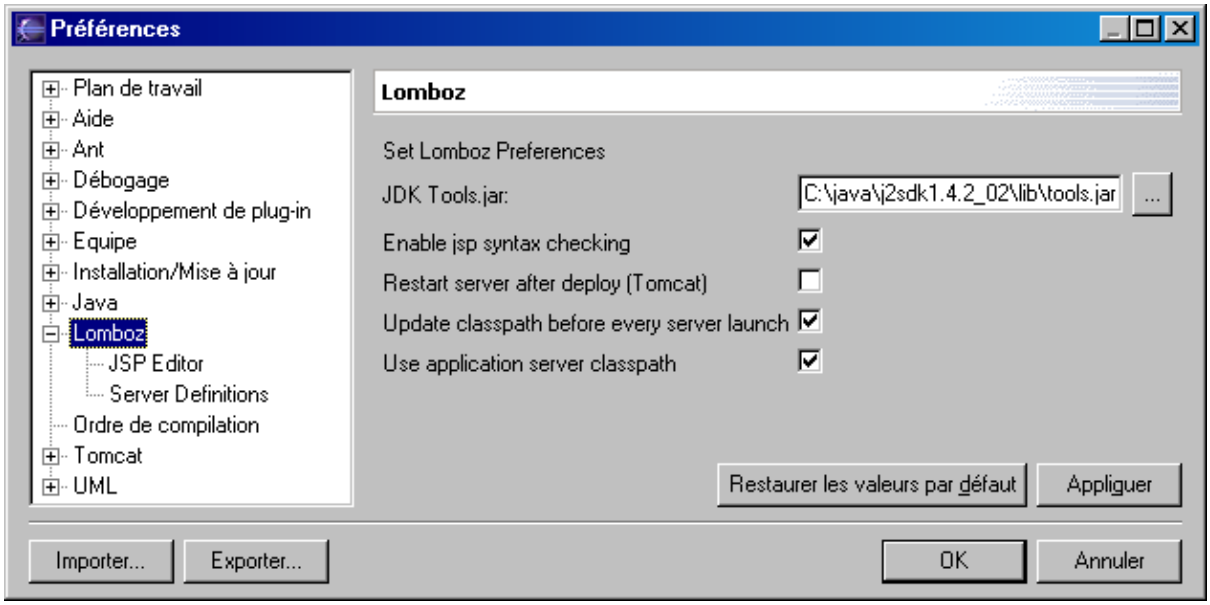
Avec Tomcat, il faut le redémarrer après le deploiement d'une webapp sinon un message d'erreur est signalé lors de l'appel de la webapp dans le navigateur.



Pour résoudre le problème manuellement, il faut arrêter le serveur et le lancer de nouveau.



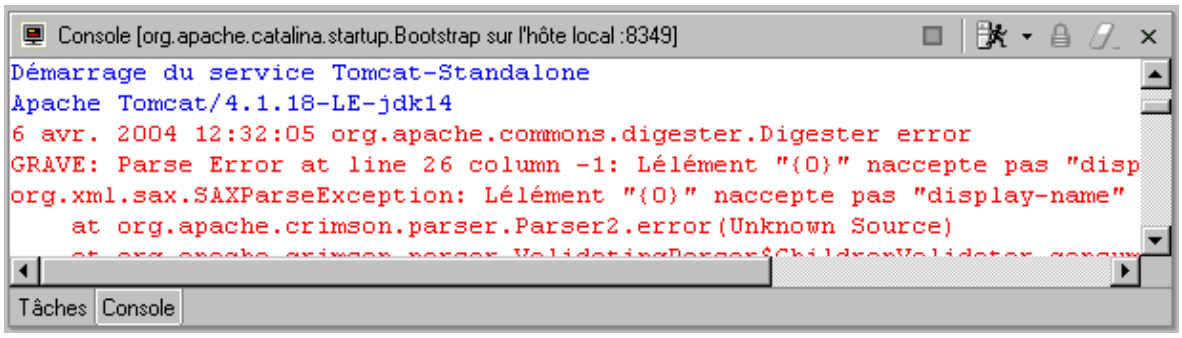
Pour résoudre automatiquement ce problème, il suffit de changer les paramètres de lomboz



Il faut cocher la case à cocher "Restart server after deploy (Tomcat)" et cliquer sur le bouton "OK ".

Il est possible que certaines erreurs empêchent le démarrage de la webapp.

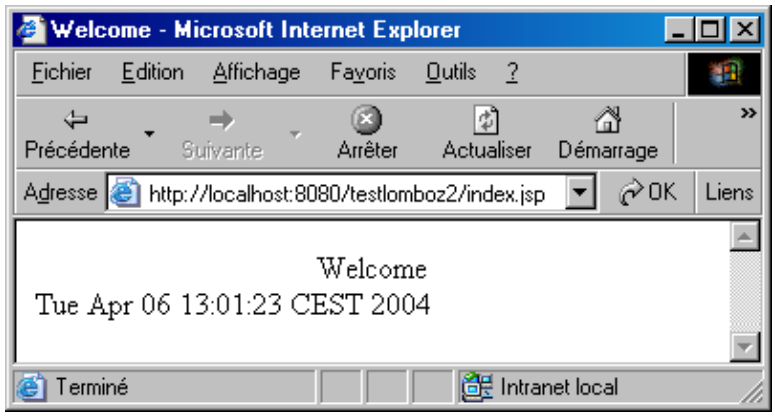
Exemple :



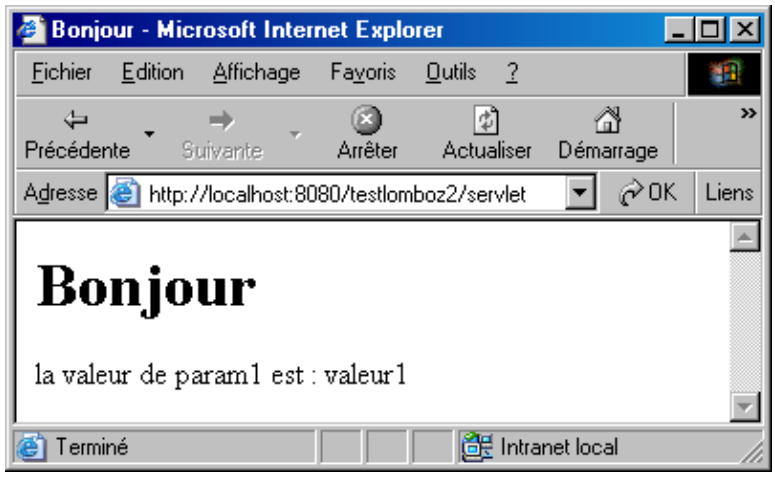
Pour résoudre ce problème, il faut modifier dans le fichier web.xml l'ordre du tag web−app/servlet/servlet−class pour le mettre après le tag web−app/servlet/description

Pour accéder aux entités qui composent la webapp, il suffit de saisir l'url correspondante dans le navigateur.

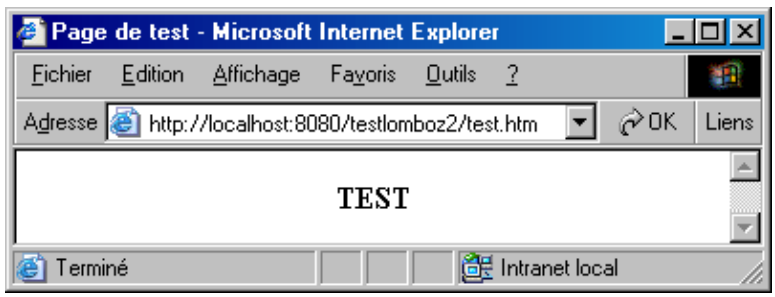
Exemple avec la JSP d'accueil de la webapp



Exemple avec la servlet (attention, le mapping url a été modifié en /servlet dans le fichier web.xml)

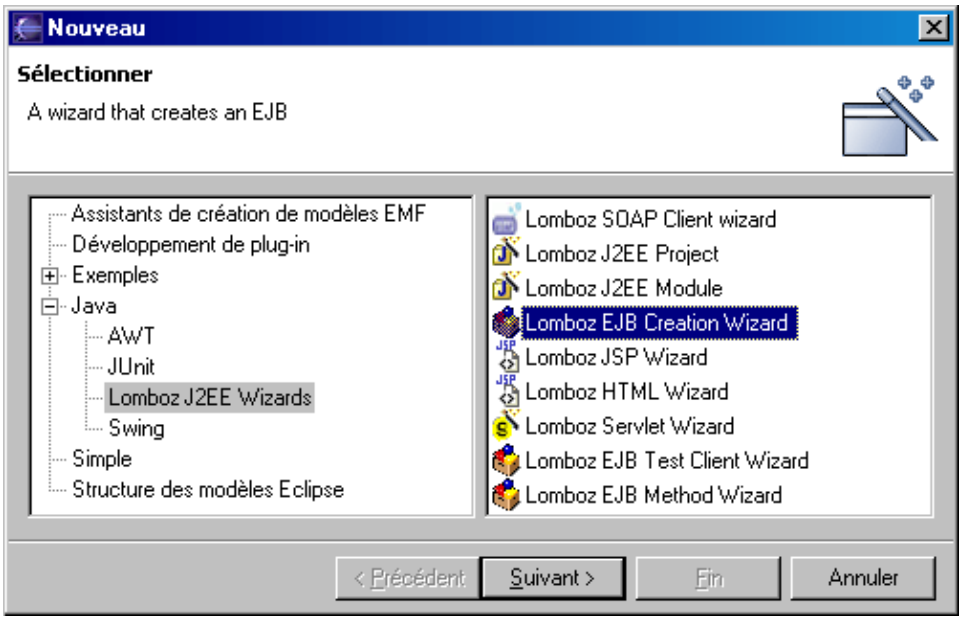


Exemple avec le fichier HTML

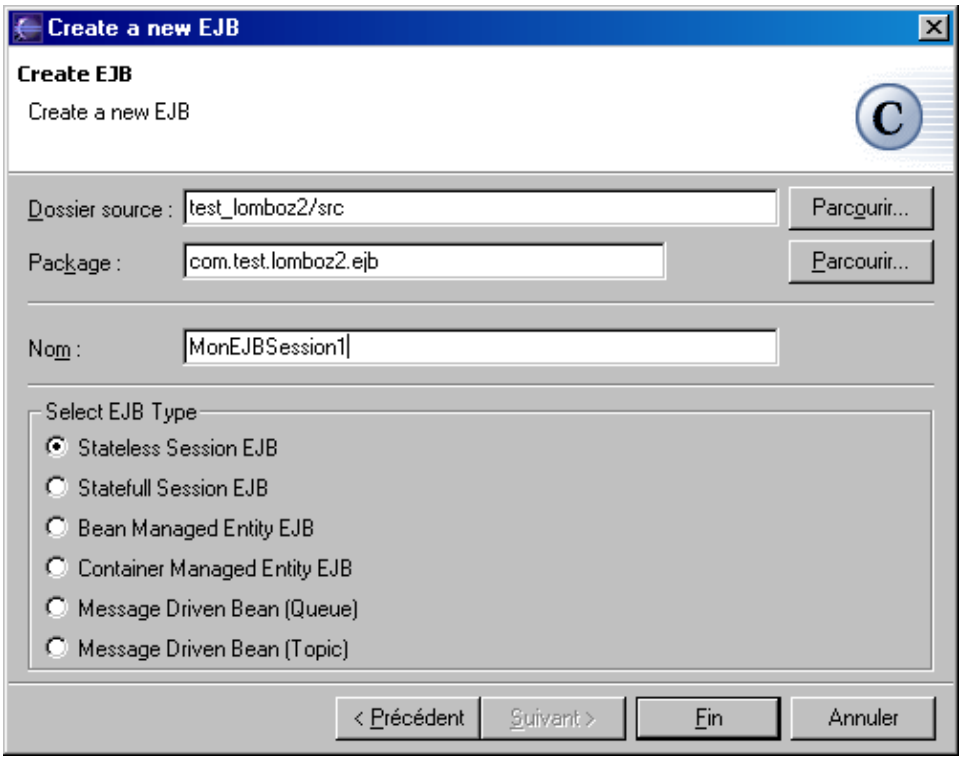


16.2.8. Créer un EJB de type Session

Il faut créer une nouvelle entité de type « Java / Lomboz J2EE Wizards / Lomboz EJB Creation Wizard ».



Cliquez sur le bouton « Suivant »



Sur la page suivante, il faut saisir le nom du package, le nom de l'EJB, sélectionner le type « Stateless Session EJB » ou « Statefull Session EJB » selon le type d'EJB session à créer et enfin cliquer sur le bouton « Fin ».

L'assistant génère un fichier source java contenant le squelette du code de l'implémentation de l'EJB. Le nom du fichier généré est constitué du nom de l'EJB et du sufixe Bean qui est automatiquement ajouté.

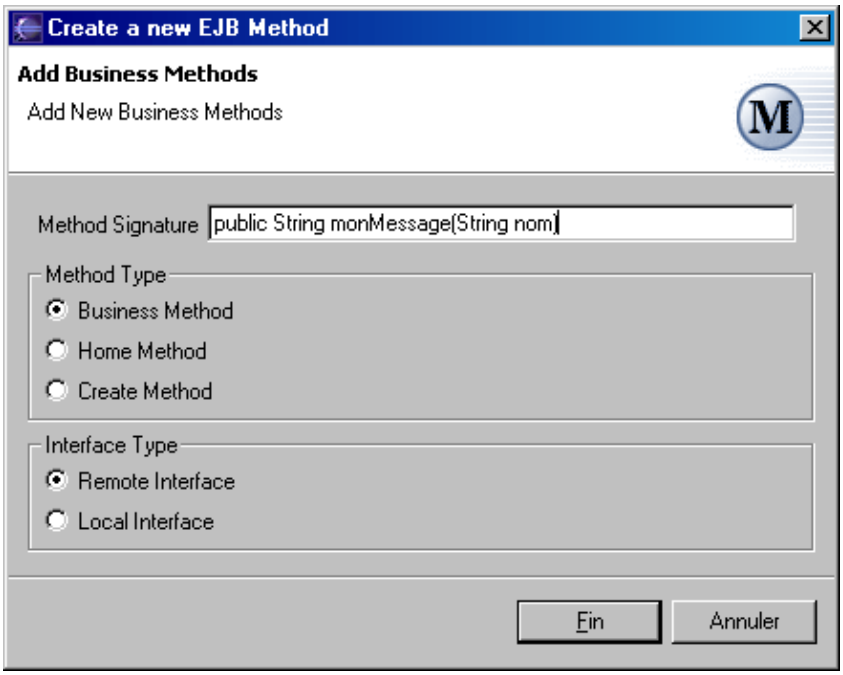
Exemple :

```
package com.test.lomboz2.ejb;

 import javax.ejb.SessionBean;

 /**
  * @ejb.bean name="MonEJBSession1"
  *     jndi-name="MonEJBSession1Bean"
  *     type="Stateless"
 **/
 public abstract class MonEJBSession1Bean implements SessionBean {
 }
```

Ce code contient un tag qui sera traiter par Xdoclet pour générer ou enrichir certains fichiers (l'interface Home du Bean, le fichier descripteur de déploiement et le fichier jboss.xml).

16.2.9. Ajouter une méthode à un EJB

Dans la vue « Packages », il faut sélectionner la classe d'implementation du Bean précédemment générée et utiliser l'option « Add EJB Method » du menu contextuel « Lomboz J2EE » ou utiliser l'option « Autre... » du menu contextuel « Nouveau ». Avec cette dernière possibilité, il faut sélectionner la création d'une entité de type « Lomboz EJB Method Wizard »



L'assistant permet de saisir la signature complète de la méthode, le type de méthode et dans quelle interface sera générée la méthode.

L'assistant génère le code suivant :

Exemple :

```
/**
 * @ejb.interface-method
 *    tview-type="remote"
**/

public String monMessage(String nom){
 return null;
}
```

Le tag dans les commentaires sera utilisé par Xdoclet pour la génération des fichiers nécessaire à l'EJB.

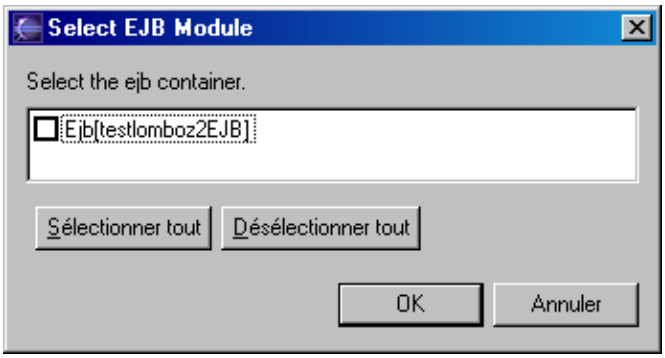
Il faut saisir le code des traitements de la nouvelle méthode.

Exemple :

```
/**
 * @ejb.interface-method
 *  tview-type="remote"
**/
public String monMessage(String nom) {
  return "Bonjour " + nom;
}
```

ajout de l'EJB à un module

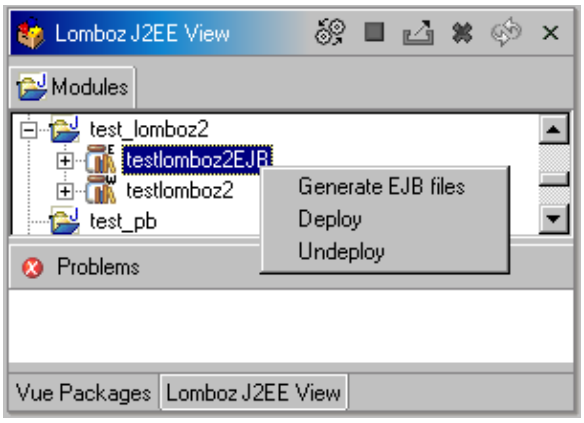
Dans la vue « Package », sélectionner la classe du bean et utiliser l'option « Lomboz J2EE / Add EJB to Module »



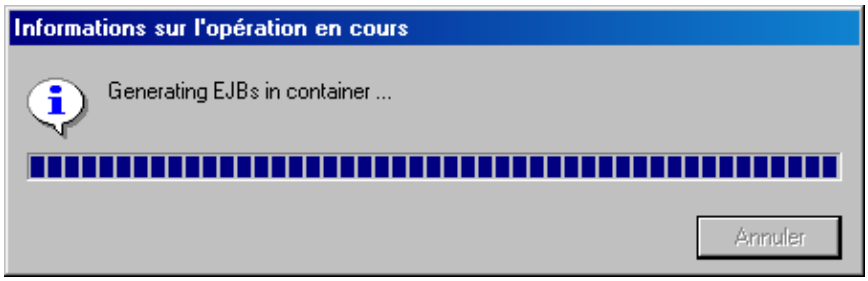
Sélectionner le module et cliquer sur « OK ».

16.2.10. La génération des fichiers des EJB

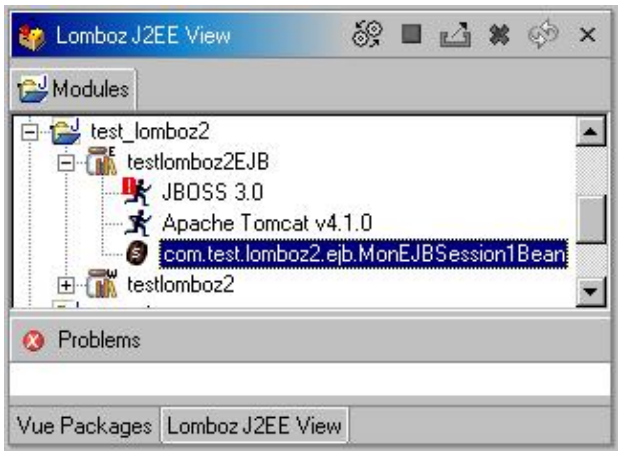
Avant de pouvoir déployer un ou plusieurs EJB, il faut demander à Lomboz de générer les fichiers nécessaires aux EJB grâce à Xdoclet et aux tags utiles insérer dans le code



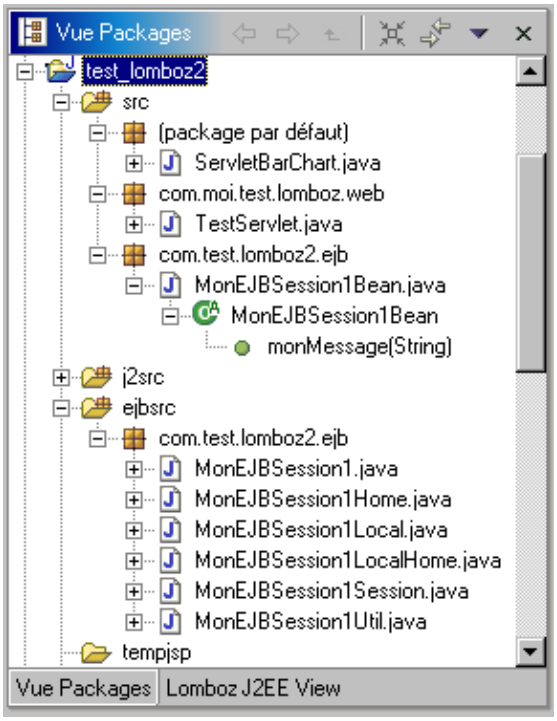
Dans la vue « Lomboz », il faut utiliser sur le module EJB l'option « Generate EJB files ».



L'EJB est affichée dans la vue Lomboz.



Lomboz a généré tous les fichiers nécessaires au bean



16.2.11. Déploiement du module EJB

Dans la vue « Lomboz », sélectionner le module EJB concerné et utiliser l'option « Deploy ».

Le fichier testlomboz2EJB.jar est créé et enregistrer dans le répertoire server/default/deploy de JBOSS.

16.2.12. Lancement du serveur Jboss

Cette section sera développée dans une version future de ce document

17. Java Server Faces et Eclipse

17.1. Utilisation de JSF sans plug−in dédié

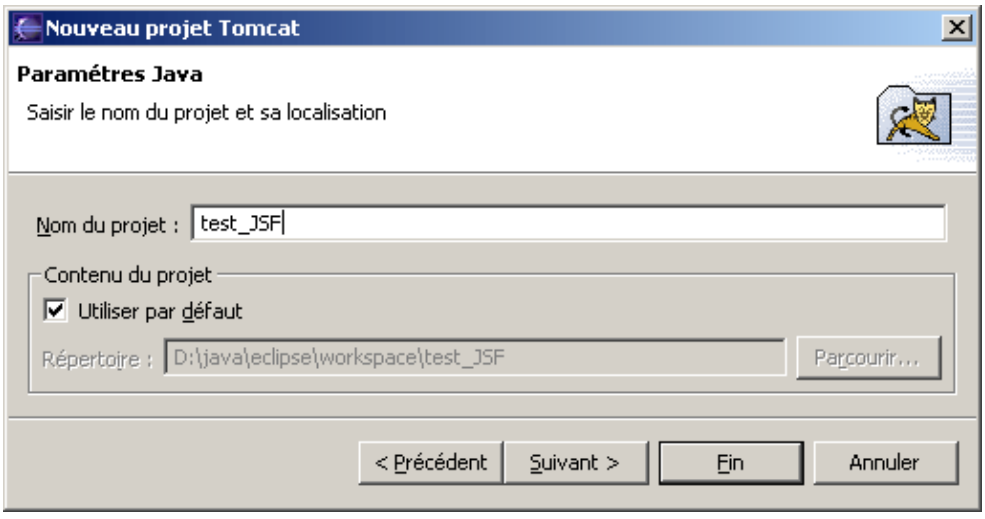
En attendant un plug−in fonctionnel dédié à l'utilisation de JSF dans Eclipse, il est possible de développer une application Web utilisant JSF avec Eclipse. Cependant, toutes les opérations doivent être réalisées "à la main".

Cette section va développer une petite application constituée de deux pages. La première va demander le nom de l'utilisateur et la seconde afficher un message de bienvenue en mettant en oeuvre les outils suivants sous Windows :

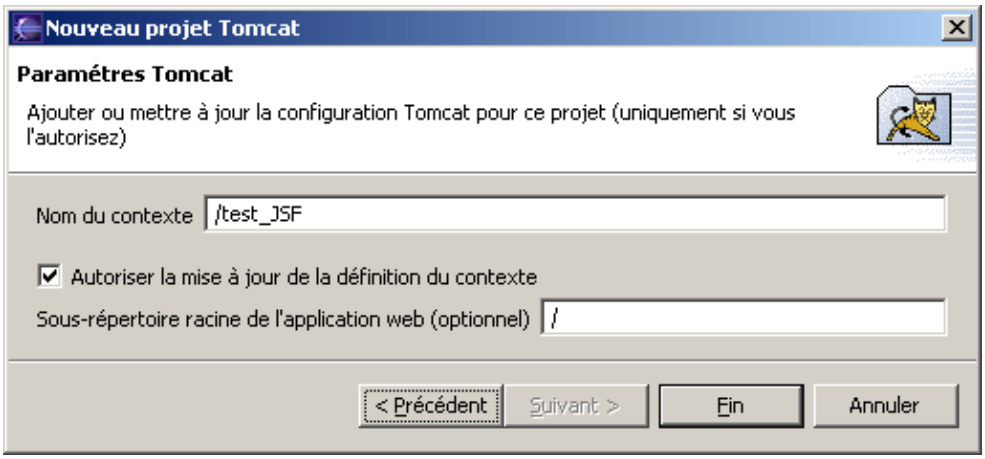
Outil	Version	Rôle
JDK	1.4.2_03	
Eclipse	2.1.3	IDE
Tomcat	5.0.28	conteneur web
plug−in Tomcat de Sysdeo	3	Arrêt et démarrage de Tomcat
JSF implémentation de référence de Sun	1.1_01	

17.1.1. Création du projet

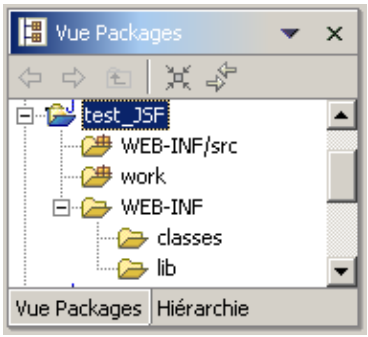
Il faut créer un nouveau projet de type « Java / Projet Tomcat ».



Saisissez le nom du projet, par exemple test_JSF et cliquez sur le bouton « Suivant ».



Cliquez sur le bouton « Fin ». L'assistant va créer un nouveau projet possédant une structure de répertoire typique pour une application Web.



Il faut ensuite copier les fichiers nécessaires à une utilisation de JSF dans l'application web.

Il suffit à partir d'un explorateur de fichier de faire un cliquer/glisser des fichiers *.jar du répertoire lib de l'implémentation de référence vers le répertoire WEB−INF/lib du projet. Attention, il est aussi nécessaire d'ajouter les fichier jstl.jar et standard.jar qui contiennent l'implémentation de la JSTL.

17.1.2. Création des éléments qui composent l'application

Il faut créer une nouvelle entité de type « Simple / Fichier » à la racine du projet et la nommer index.htm

Exemple :

```
<html>
<head>
  <meta http-equiv="Refresh" content= "0; URL=login.faces"/>
  <title>Demarrage de l'application</title>
</head>
<body>
  <p>D&eacute;marrage de l'application ...</p>
</body>
</html>
```

Il faut créer une nouvelle entité de type « Simple / Fichier » à la racine du projet et la nommer login.jsp

Exemple :

```
<html>
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
```

```
<f:view>
<head>
  <title>Application de tests avec JSF</title>
</head>
<body>
  <h:form>
    <h3>Identification</h3>
    <table>
      <tr>
        <td>Nom : </td>
        <td><h:inputText value="#{login.nom}"/></td>
      </tr>
      <tr>
        <td>Mot de passe :</td>
        <td><h:inputSecret value="#{login.mdp}"/></td>
      </tr>
      <tr>
      <td colspan="2"><h:commandButton value="Login" action="login"/></td>
      </tr>
    </table>
  </h:form>
</body>
</f:view>
</html>
```

Il faut créer une nouvelle classe nommée com.jmd.test.jsf.LoginBean

Exemple :

```
package com.jmd.test.jsf;

public class LoginBean {
  private String nom;
  private String mdp;

  public String getMdp() {
    return mdp;
  }

  public String getNom() {
    return nom;
  }

  public void setMdp(String string) {
    mdp = string;
  }

  public void setNom(String string) {
    nom = string;
  }
}
```

Il faut créer une nouvelle entité de type « Simple / Fichier » à la racine du projet et la nommer accueil.jsp

Exemple :

```
<html>
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<f:view>
<head>
  <title>Page d'accueil de l'application</title>
</head>
<body>
```

```
  <h:form>
    <h3>Bienvenue <h:outputText value="#{login.nom}"/>,</h3>
  </h:form>
</body>
</f:view>
</html>
```

Il faut créer une nouvelle entité de type « Simple / Fichier » dans le répertoire /WEB−INF et la nommer faces−config.xml

Exemple :

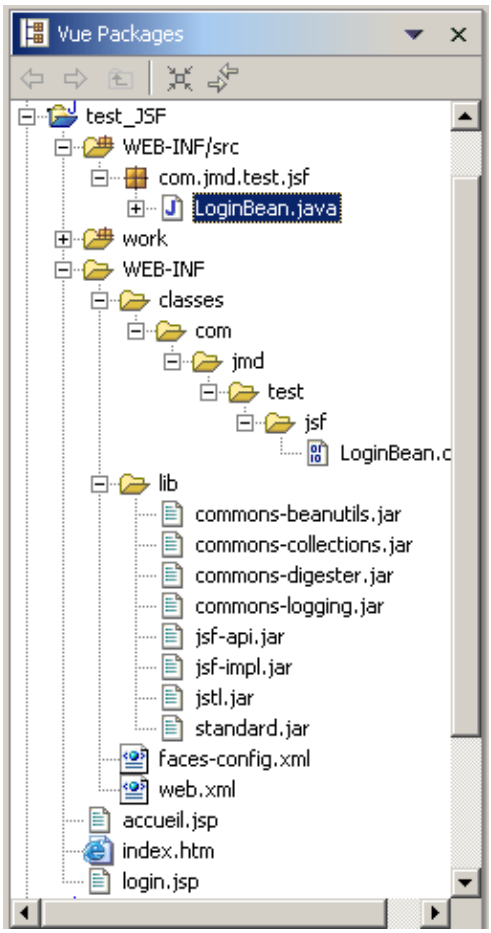
```
<?xml version="1.0"?>
<!DOCTYPE faces−config PUBLIC
"−//Sun Microsystems, Inc.//DTD JavaServer Faces Config 1.0//EN"
"http://java.sun.com/dtd/web−facesconfig_1_0.dtd">
<faces−config>
  <navigation−rule>
    <from−view−id>/login.jsp</from−view−id>
    <navigation−case>
      <from−outcome>login</from−outcome>
      <to−view−id>/accueil.jsp</to−view−id>
    </navigation−case>
  </navigation−rule>
  <managed−bean>
    <managed−bean−name>login</managed−bean−name>
    <managed−bean−class>com.jmd.test.jsf.LoginBean</managed−bean−class>
    <managed−bean−scope>session</managed−bean−scope>
  </managed−bean>
</faces−config>
```

Il faut créer une nouvelle entité de type « Simple / Fichier » dans le répertoire /WEB−INF et la nommer web.xml

Exemple :

```
<?xml version="1.0"?>
<!DOCTYPE web−app PUBLIC
"−//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
"http://java.sun.com/dtd/web−app_2_3.dtd">
<web−app>
  <servlet>
    <servlet−name>Faces Servlet</servlet−name>
    <servlet−class>javax.faces.webapp.FacesServlet</servlet−class>
    <load−on−startup>1</load−on−startup>
  </servlet>
  <servlet−mapping>
    <servlet−name>Faces Servlet</servlet−name>
    <url−pattern>*.faces</url−pattern>
  </servlet−mapping>
  <welcome−file−list>
    <welcome−file>index.htm</welcome−file>
  </welcome−file−list>
</web−app>
```

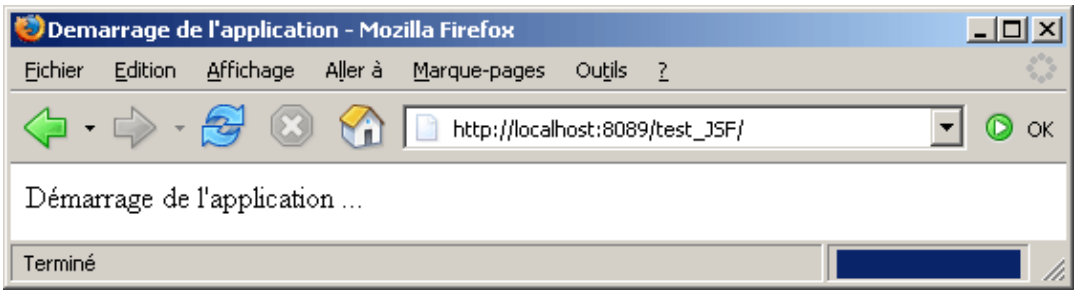
Les éléments qui composent le projet sont donc les suivants :



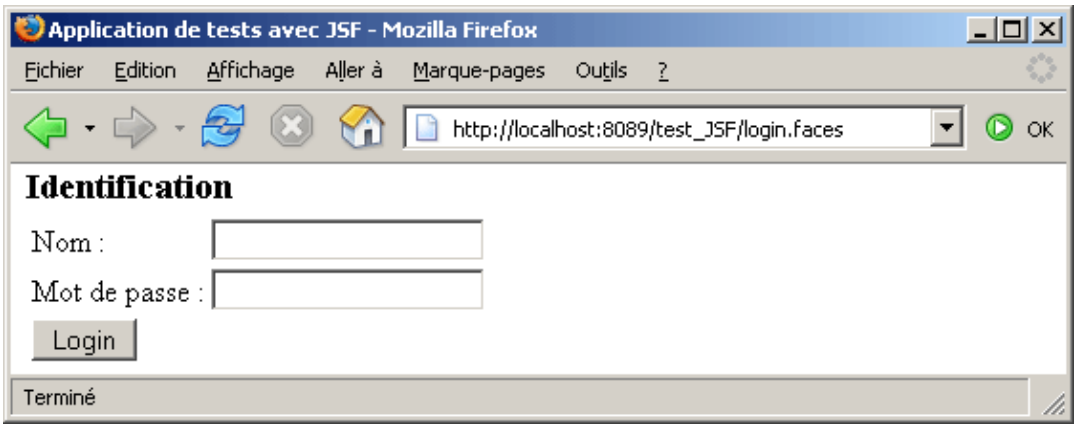
17.1.3. Exécution de l'application

Il suffit alors de démarrer Tomcat en cliquant sur le bouton  .

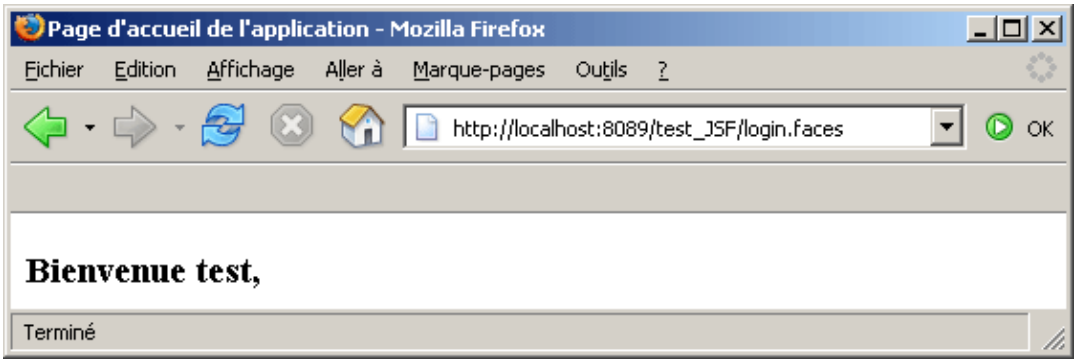
Une fois Tomcat démarré, ouvrir un navigateur et taper l'url http://localhost:8089/test_JSF/ (en remplaçant le port 8089 par celui défini dans Tomcat).



Une fois l'application démarrée, la page de login s'affiche



Il faut saisir un nom par exemple test et cliquer sur le bouton « Login ».



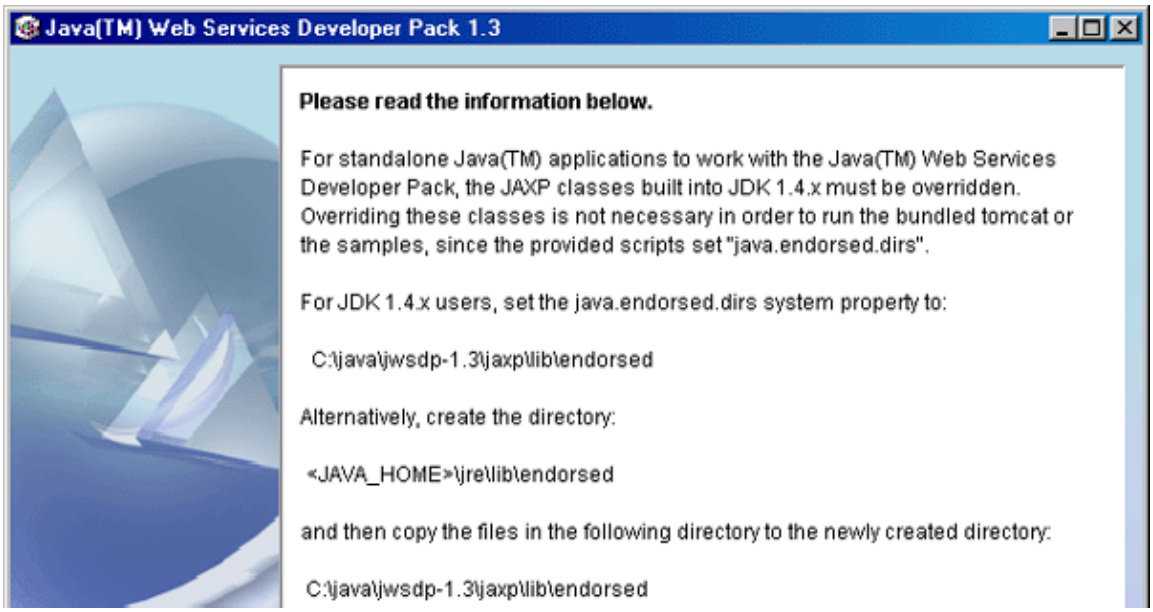
Cet exemple ne met pas en valeur la puissance de Java Server Faces mais propose simplement de mettre en oeuvre le minimum pour développer des applications utilisant JSF avec Eclipse.

18. JAXB et Eclipse

JAXB est l'acronyme de Java Architecture for XML Binding. Le but de l'API et des spécifications JAXB est de faciliter la manipulation d'un document XML en générant un ensemble de classes qui fournissent un niveau d'abstraction plus élevé que l'utilisation de JAXP (SAX ou DOM). Avec ces deux API, toute la logique de traitements des données contenues dans le document est à écrire.

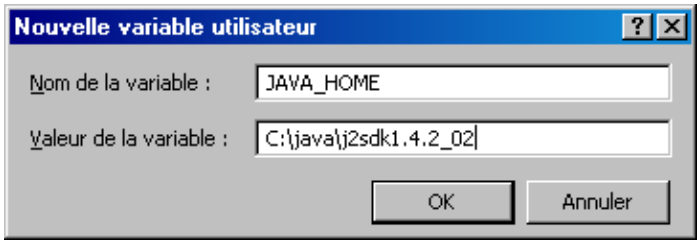
JAXB au contraire fournit un outil qui analyse un schéma XML et génère à partir de ce dernier un ensemble de classes qui vont encapsuler les traitements de manipulation du document.

Pour pouvoir utiliser JAXB, il est nécessaire d'installer le Java Web Services Developer Pack 1.3. Dans le cas de l'utilisation d'un JDK 1.4, attention de bien suivre les instructions fournies à la fin de l'installation.

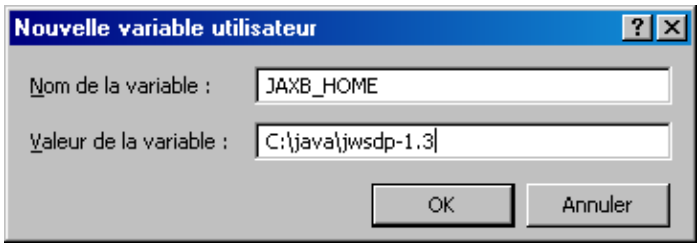


Il est aussi nécessaire de définir deux variables d'environnement système (sous Windows 2000 : menu "Paramètres / Panneau de configuration / Système", sélectionner l'onglet "avancé", puis cliquer sur le bouton "variables d'environnement") :

JAVA_HOME dont la valeur doit correspondre au répertoire d'installation du JDK :



JAXB_HOME dont la valeur doit correspondre au répertoire dans lequel le JWSDP est installé :



Il existe deux moyens d'utiliser JAXB avec Eclipse :

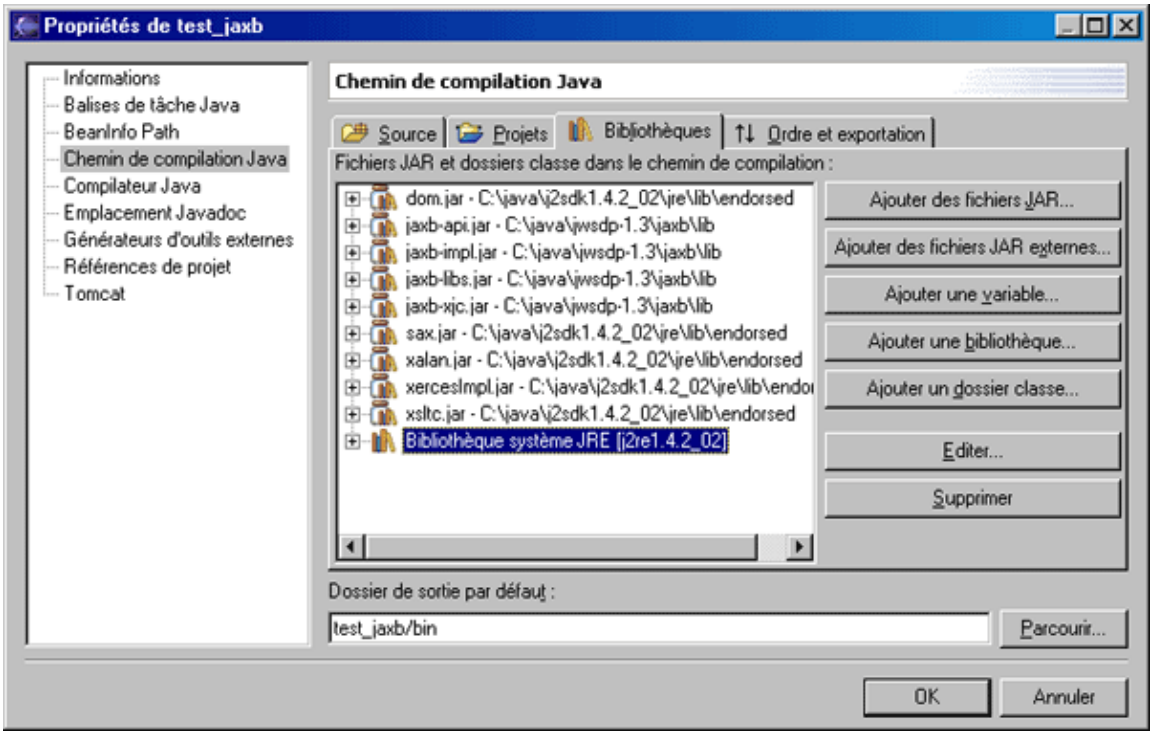
- Créer et configurer une tâche d'exécution qui va lancer la génération des classes Java via l'outil de JAXB
- Exécuter Ant fourni avec le JWSDP en tant qu'outil externe dans Eclipse

	Version utilisée dans cette section
Eclipse	2.1.2
J2RE	1.4.2_02
JAXB	1.3

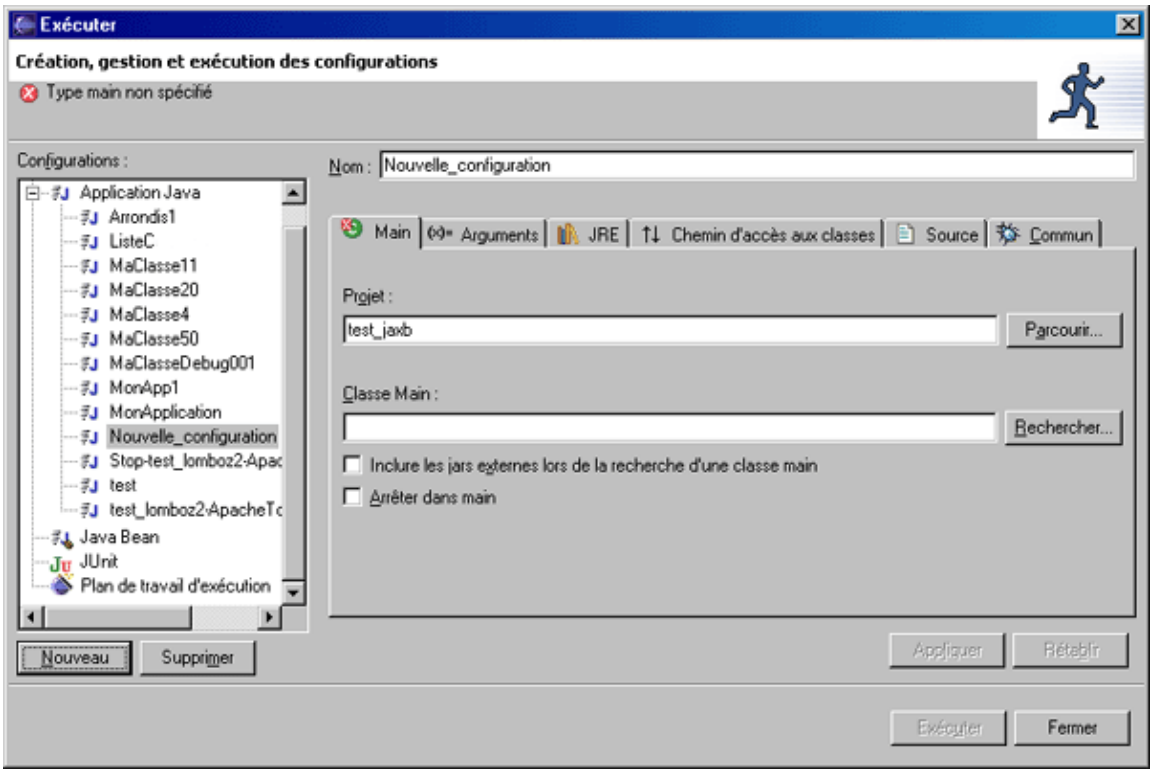
18.1. Créer et configurer une tâche d'exécution pour JAXB

Il faut créer un nouveau projet, par exemple nommé test_jaxb et modifier son chemin de compilation (dans les propriétés du projet) pour ajouter plusieurs fichiers .jar :

- Les fichiers jar ajoutés dans le répertoire JAVA_HOME\jre\lib\endorsed lors de l'installation du JWSDP
- Les fichiers jar contenus dans le répertoire JAXB_HOME\jaxb\lib



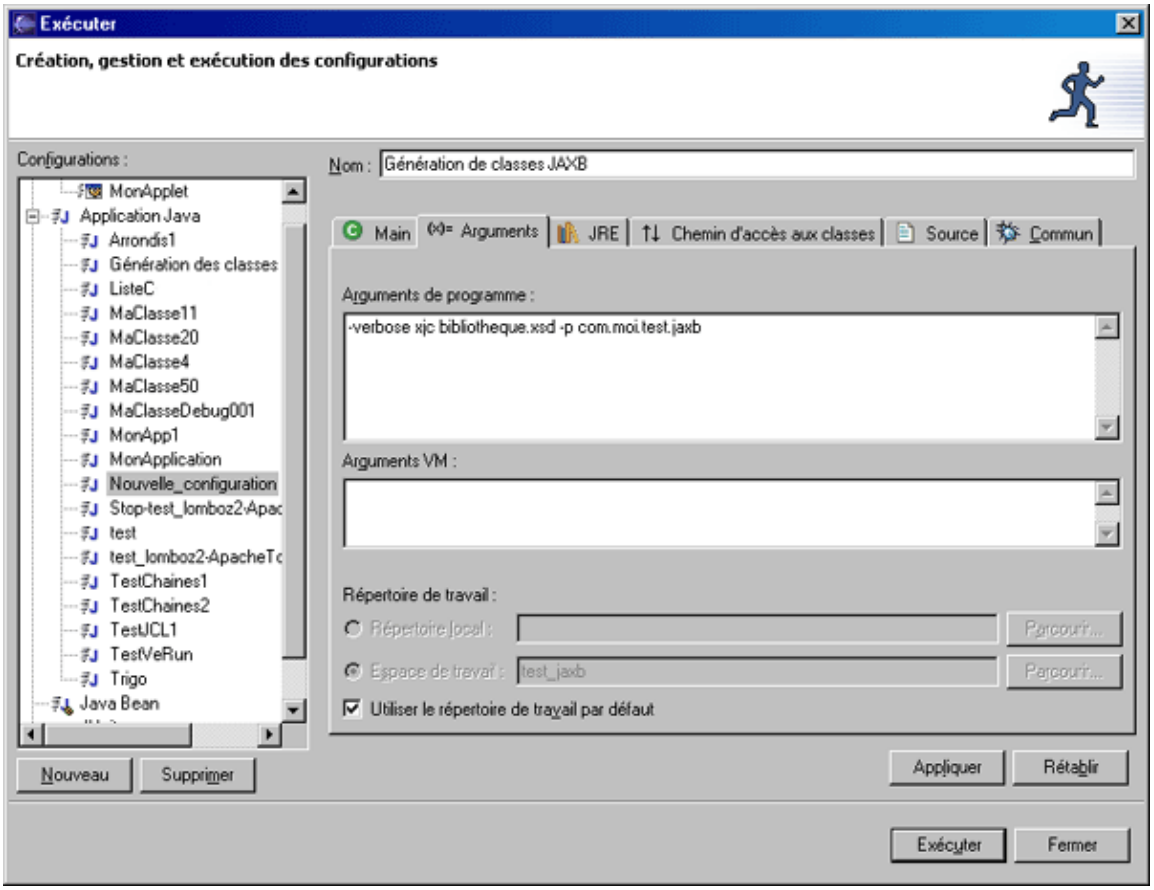
Il faut ensuite créer une nouvelle tâche d'exécution en utilisant l'option du menu « Exécuter/Exécuter ... » et en cliquant sur le bouton “Nouveau”.



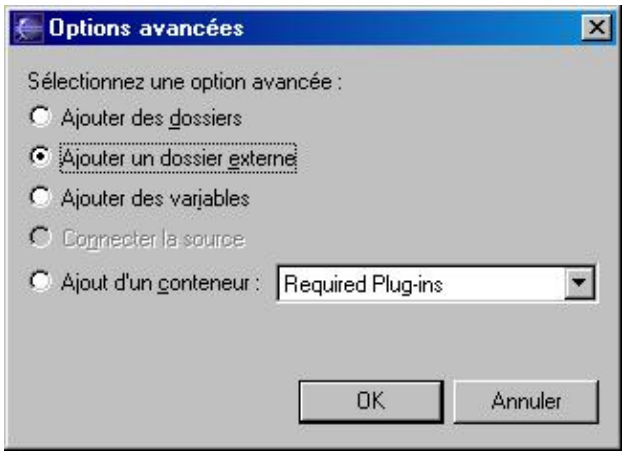
Il faut changer le nom, par exemple « Génération des classes JAXB » et saisir dans le champ Classe Main la valeur « LauncherBootstrap ».

Sur l'onglet « Arguments », dans la zone de saisie "Arguments de programme" saisir (bibliotheque.xsd est le nom du fichier qui contient le schéma du document XML) :

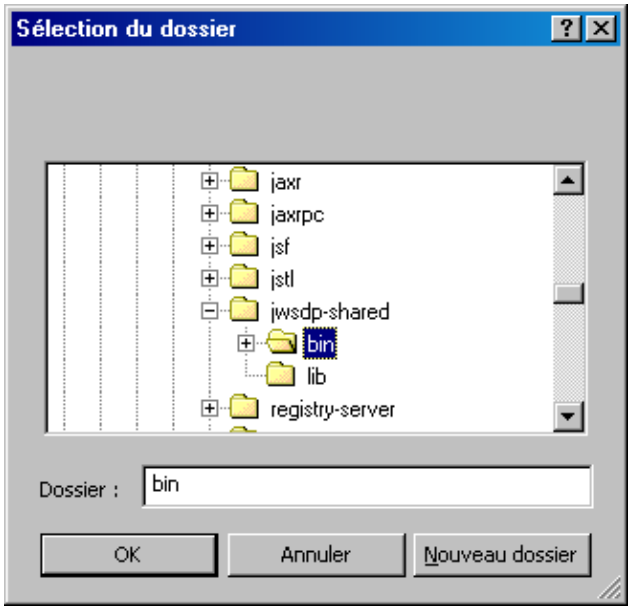
−verbose xjc bibliotheque.xsd −p com.moi.test.jaxb



Sur l'onglet « Chemin d'accès aux classes », décochez « Utiliser le chemin d'accès aux classes par défaut » puis cliquez sur le bouton « Avancées »



Cliquez sur le bouton « Ajouter un dossier externe » puis sur le bouton "Ok"

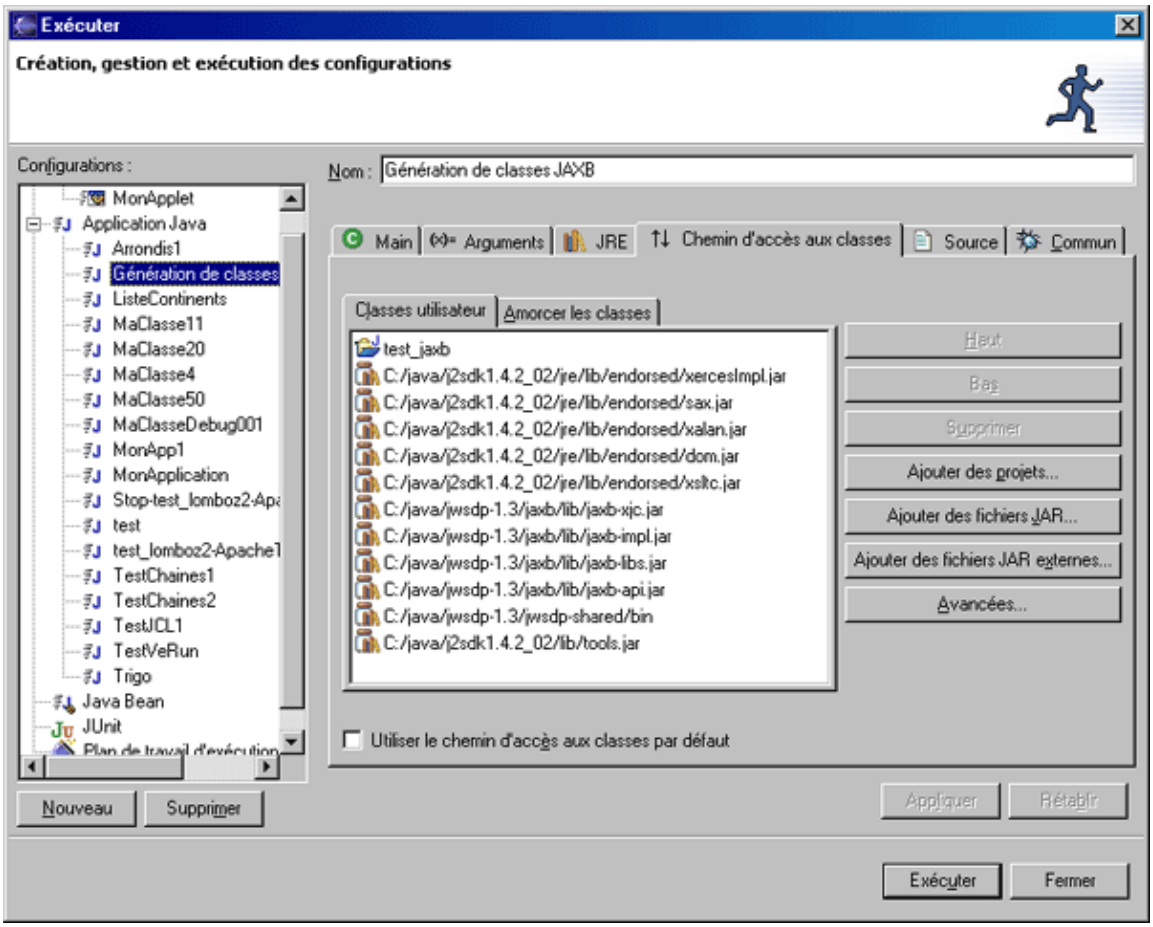


Selectionnez le repertoire %JWSDP%/jwsdp−shared/bin et cliquez sur le bouton « OK »

Cliquez sur le bouton « Fermer »

Cliquez sur le bouton « Oui »

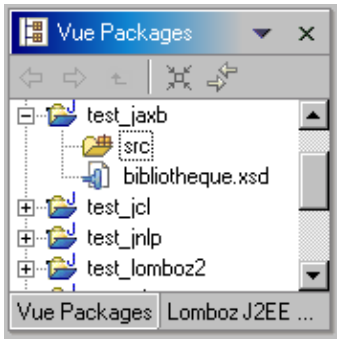
Il faut aussi ajouter le fichier tools.jar présent dans le répertoire lib du JDK :



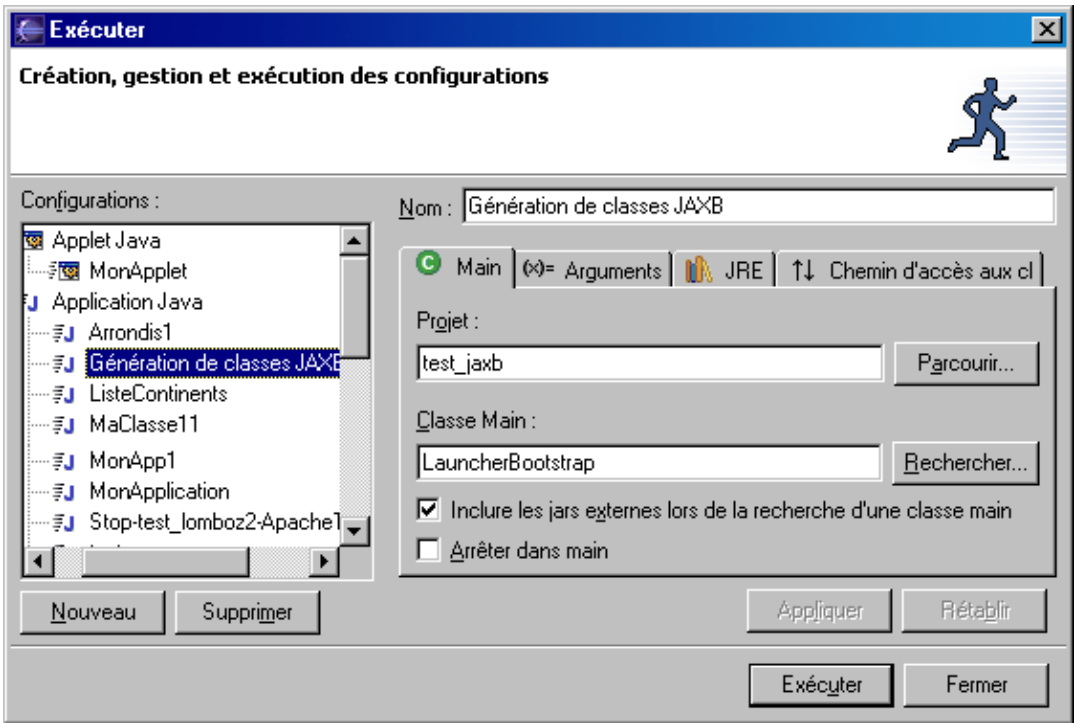
Si ce fichier n'est pas ajouté, une exception est levée lors de l'exécution :

file:C:/java/jwsdp−1.3/jwsdp−shared/bin/launcher.xml:329: java.io.IOException: Could not find Java(TM) 2 SDK classes. This application cannot run using the Java(TM) 2 JRE. It requires the full SDK.
    at org.apache.commons.launcher.LaunchTask.execute(LaunchTask.java:728)
    at org.apache.tools.ant.Task.perform(Task.java:341)
    at org.apache.tools.ant.Target.execute(Target.java:309)

La configuration est maintenant terminée, il faut un schéma qui décrit le document XML directement à la racine du projet, par exemple bibliotheque.xsd.

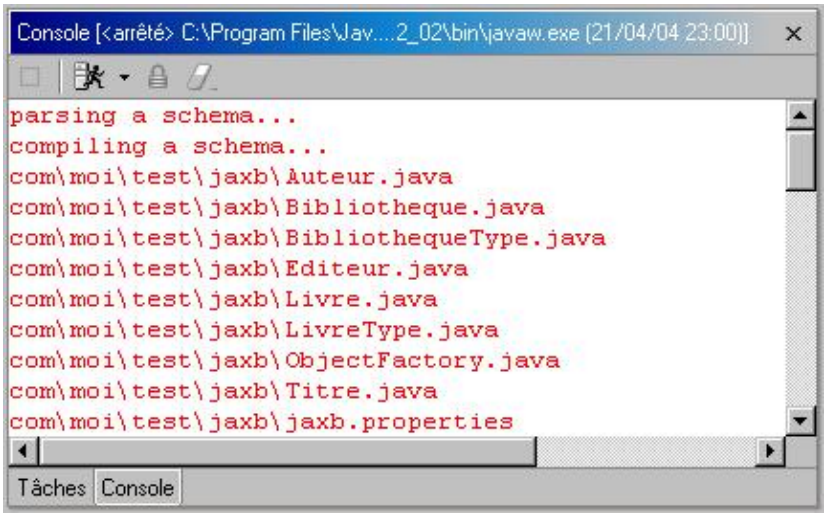


Pour lancer la génération des classes Java par JAXB, il faut utiliser l'option du menu « Exécuter/Exécuter ... » et sélectionner « Application Java/Generation de classe JAXB »

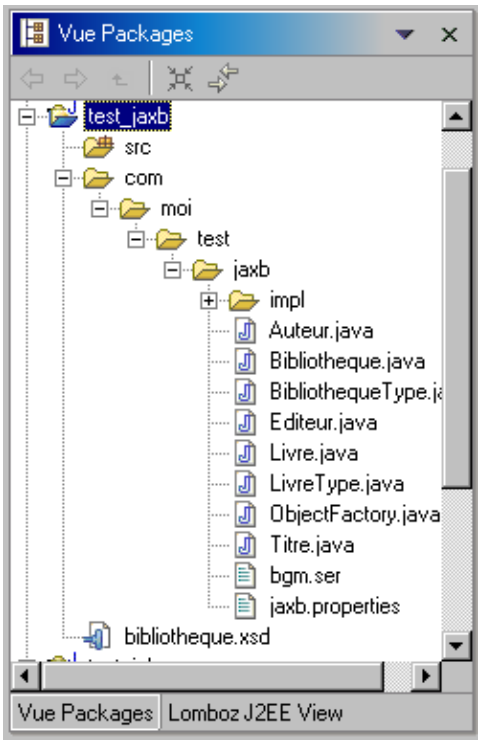


La boîte de dialogue réaffiche les paramètres précédemment enregistrés : il suffit de cliquer sur le bouton « Exécuter ».

Les messages issus des traitements sont affichés dans la console.

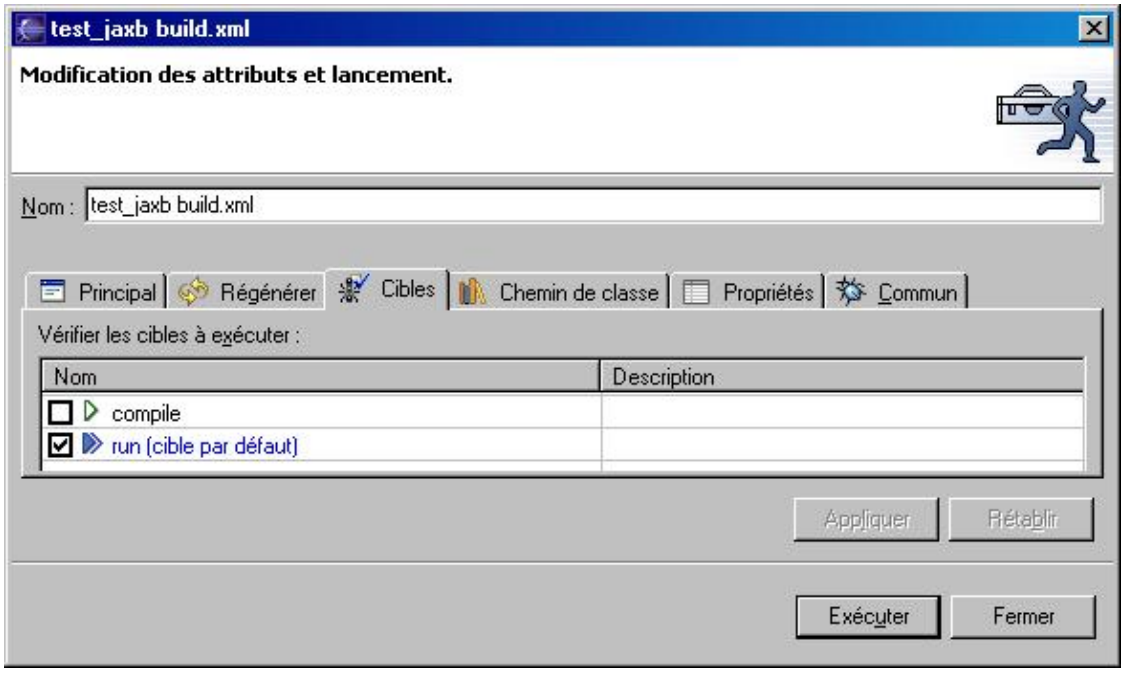


Dans la vue « Packages », sélectionner le projet test_jaxb et utiliser l'option « Régénérer » du menu contextuel pour rafraîchir le contenu du projet et voir les fichiers générés lors des traitements.



18.2. Exécuter Ant en tant qu'outil externe

L'exécution de JAXB se fait normalement en utilisant une tache Ant dédiée.



Malheureusement, il y a une incompatibilité entre la version de Ant fournie avec Eclipse et celle requise par JAXB. L'exécution d'un build.xml en utilisant Ant fourni avec Eclipse échoue systématiquement :

Résultat de l'exécution :

```
Buildfile: C:\java\eclipse\workspace\test_jaxb\build.xml
  compile:

          [echo] Compiling the schema external binding file...
           [xjc] Compiling file:/C:/java/eclipse/workspace/test_jaxb/bibliotheque.xsd
```

```
          [xjc] [WARNING] Unable to validate your schema. Most likely, the JVM has
                loaded an incompatible XML parser implementation. You should fix this
                before relying on  the generated code. Please see the release notes
                for details.
          [xjc] unknown location
          [xjc]
          [xjc] BUILD FAILED: file:C:/java/eclipse/workspace/test_jaxb/build.xml:31:
                unable to parse the schema. Error messages should have been provided


Total time: 5 seconds
```

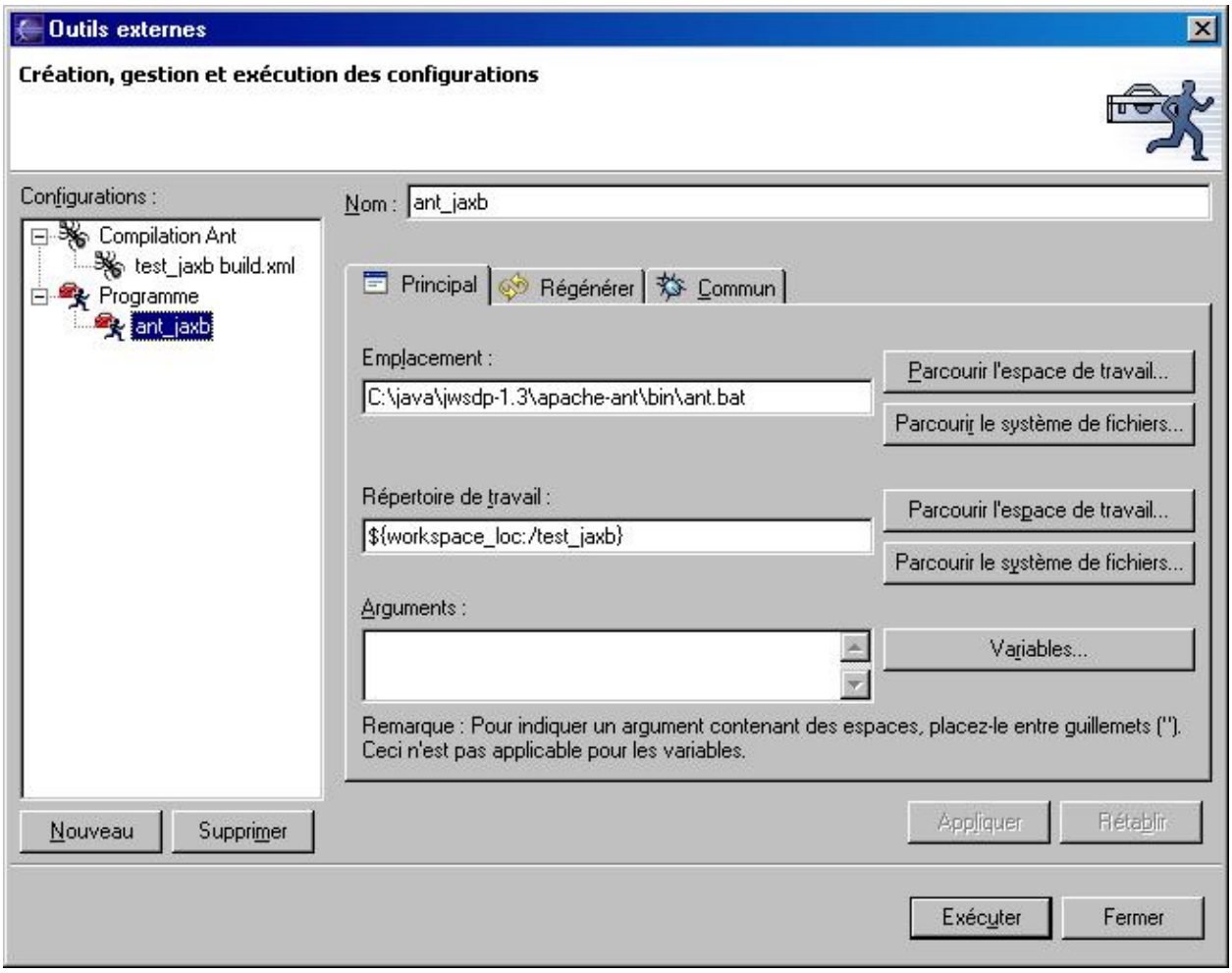
Pour résoudre ce problème et utiliser Ant, il faut l'exécuter en tant qu'outil externe.

Il faut définir le fichier build.xml qui va contenir les différents traitements à exécuter par Ant.

Résultat de l'exécution :

```
<?xml version="1.0"?>
<project basedir="." default="compile">
  <property name="jwsdp.home" value="C:\java\jwsdp-1.3" />
  <property name="java.home" value="C:\java\j2sdk1.4.2_02" />

  <path id="classpath">
    <pathelement path="." />
    <fileset dir="${java.home}" includes="jre/lib/endorsed/*.jar" />
    <fileset dir="${jwsdp.home}" includes="jaxb/lib/*.jar" />
    <fileset dir="${jwsdp.home}" includes="jwsdp-shared/lib/*.jar" />
    <fileset dir="${jwsdp.home}" includes="jaxp/lib/**/*.jar" />
  </path>

  <taskdef name="xjc" classname="com.sun.tools.xjc.XJCTask">
    <classpath refid="classpath" />
  </taskdef>

  <target name="compile">
    <echo message="Generation des classes Java a partir du schema ..."/>
    <xjc schema="bibliotheque.xsd" target="./src" package="com.moi.test.jaxb"/>
    <echo message="Compilation des sources ..."/>
    <javac srcdir="./src" destdir="." debug="on">
      <classpath refid="classpath" />
    </javac>
  </target>

</project>
```

Il faut utiliser l'option du menu « Exécuter/ Outils externes/Outils externes ... » et cliquer sur le bouton « Nouveau ».



Sur l'onglet « Principal », il suffit de sélectionner l'emplacement du fichier ant.bat fourni avec le JWSDP, puis de cliquer sur le bouton « Appliquer », puis sur le bouton « Exécuter ».

Les informations générées par Ant au cours de l'exécution sont affichées dans la console.

Résultat de l'exécution :

```
Buildfile: build.xml

compile:
     [echo] Generation des classes Java a partir du schema ...
      [xjc] Compiling file:/C:/java/eclipse/workspace/test_jaxb/bibliotheque.xsd
      [xjc] Writing output to C:\java\eclipse\workspace\test_jaxb\src
     [echo] Compilation des sources ...
    [javac] Compiling 44 source files to C:\java\eclipse\workspace\test_jaxb

BUILD SUCCESSFUL
Total time: 1 minute 4 seconds
```

Pour voir les fichiers générés par JAXB lors de ces traitements, il suffit d'utiliser l'option « Régénérer » du menu contextuel du projet dans la vue « Packages ».

19. Struts et Eclipse

Struts est un framework pour applications web développé par le projet Jakarta de la fondation Apache. C'est la plus populaire des frameworks pour le développement d'applications web avec Java .

Struts met en oeuvre le modèle MVC 2 (Modèle / Vue / Contôleur) basé sur une seule servlet et des JSP pour chaque application. L'application de ce modèle permet une séparation en trois partie distinctes de l'interface, des traitements et des données de l'application.

Struts se concentre sur la vue et le contrôleur. L'implémentation du modèle est laissée libre aux développeurs : ils ont le choix d'utiliser des java beans, un outil de mapping objet/relationnel ou des EJB.

Pour le contrôleur, Struts propose une unique servlet par application qui lit la configuration de l'application dans un fichier au format XML. Cette servlet reçoit toutes les requêtes de l'utilisateur concernant l'application. En fonction du paramètrage, il instancie un objet de type Action qui contient les traitements et renvoie une valeur particulière à la servlet. Ceci lui permet de déterminer la JSP qui affichera le résultat à l'utilisateur.

Il existe un projet open source nommé Easy Struts qui est un plug−in dont le but est de faciliter la mise en oeuvre de Struts avec Eclipse. Ce plug−in très intéressant et pratique ne semble malheureusement plus évoluer.

Cette section va mettre en oeuvre les outils suivants sous Windows :

Outil	Version	Rôle
JDK	1.4.2_03	
Eclipse	2.1	IDE
Tomcat	5.0.28	conteneur web
plug−in Tomcat de Sysdeo	3	arrêt et démarrage de Tomcat
plug−in Easy Struts	0.6.4	faciliter l'utilisation de Struts

19.1. Le plug−in Easy Struts

Le site officiel du plug−in Easy Struts est à l'url : http://easystruts.sourceforge.net/

Ce plug−in permet notamment de faciliter la réalisation de certaines tâches et la création des éléments suivants :

- "Add Easy Struts support" : permet d'ajouter les éléments nécessaire à l'utilisation de Struts dans l'application (fichier .jar, .tld, ...) et créé les fichiers de configuration

- "Easy Form" : créer une JSP avec une classe de type ActionForm associée et ajoute la définition de ce bean dans le fichier de configuration
- "Easy Action" : créer une classe de type Action et ajoute la definition du mapping de cette classe dans le fichier de configuration
- "Easy Action associated with a form" : créer une JSP avec une classe de type ActionForm associée, ajoute la définition de ce bean dans le fichier de configuration et créé une classe de type Action et ajoute la definition du mapping de cette classe dans le fichier de configuration
- "Easy Forward" : créer des renvois dédiés à une Action ou globaux
- "Easy Exception" : créer des handler pour les exceptions.
- "Easy Message resources" : créer des fichiers de ressources pour localiser l'application
- “Easy Plug−in" :
- "Easy Datasource" : permet de définir une source de données qui sera utilisée dans l'application
- “Easy Module” :

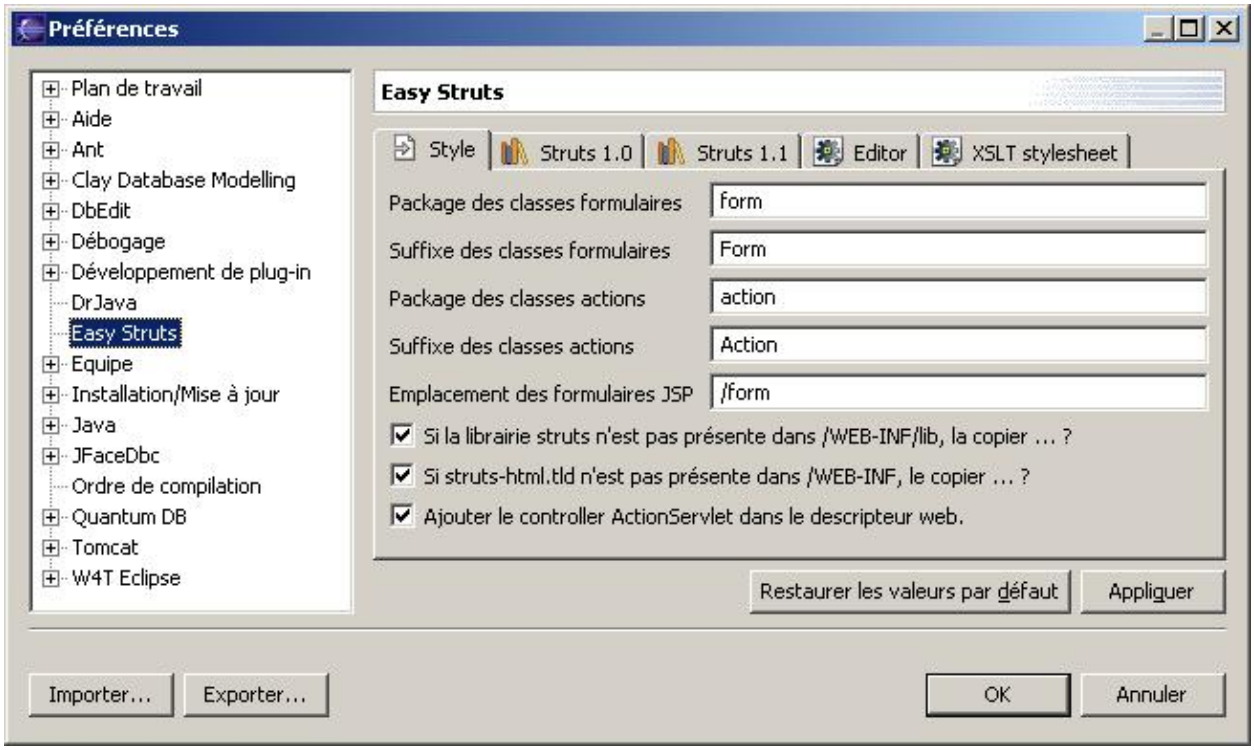
19.2. Installation et configuration d'Easy Struts

Il y a deux façons pour installer Esay Struts :

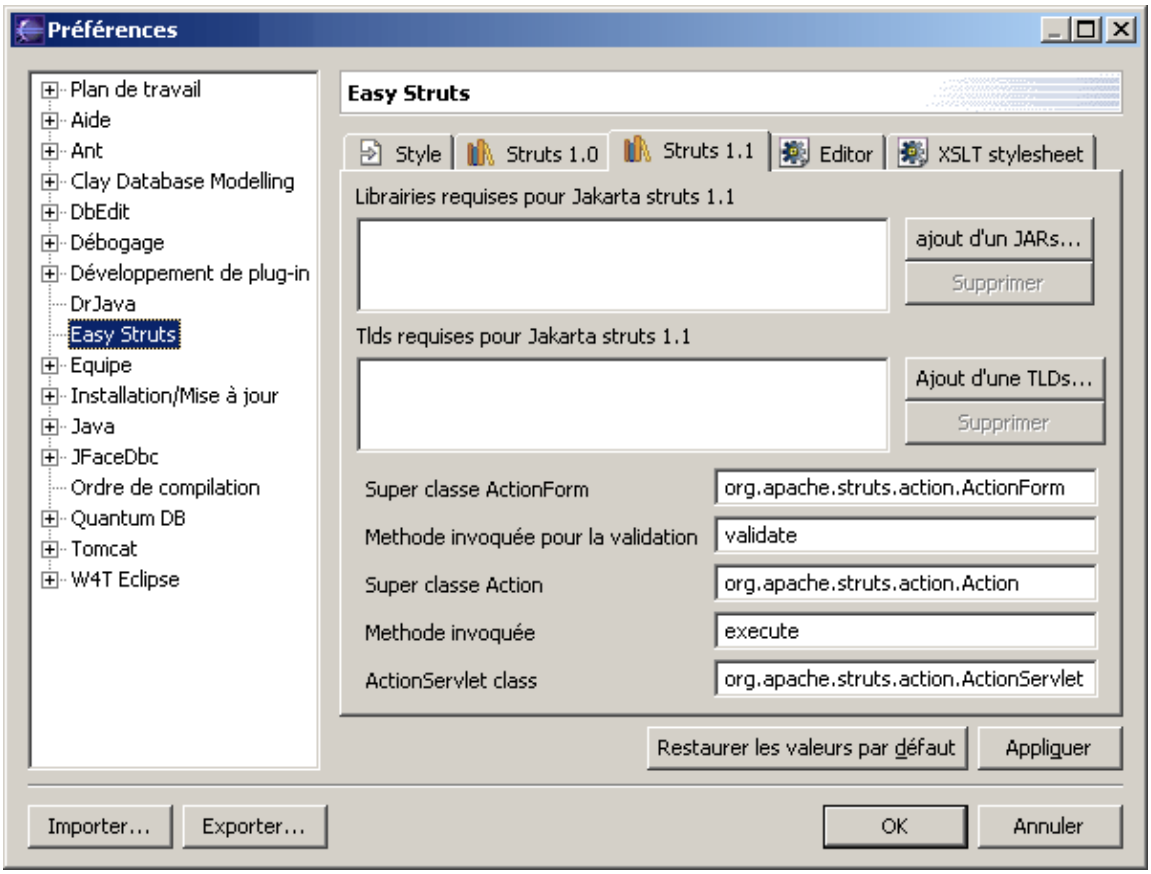
- téléchargez et décompressez le fichier org.easystruts.eclipse_0.6.4.zip dans le répertoire « plugins » d'Eclipse puis lancer Eclipse.
- utilisez le gestionnaire de mises à jour avec l'url http://easystruts.sourceforge.net/eclipse/updates/site.xml

Editez les préférences (menu Fenêtre/Préférences) et sélectionnez Easy Struts.

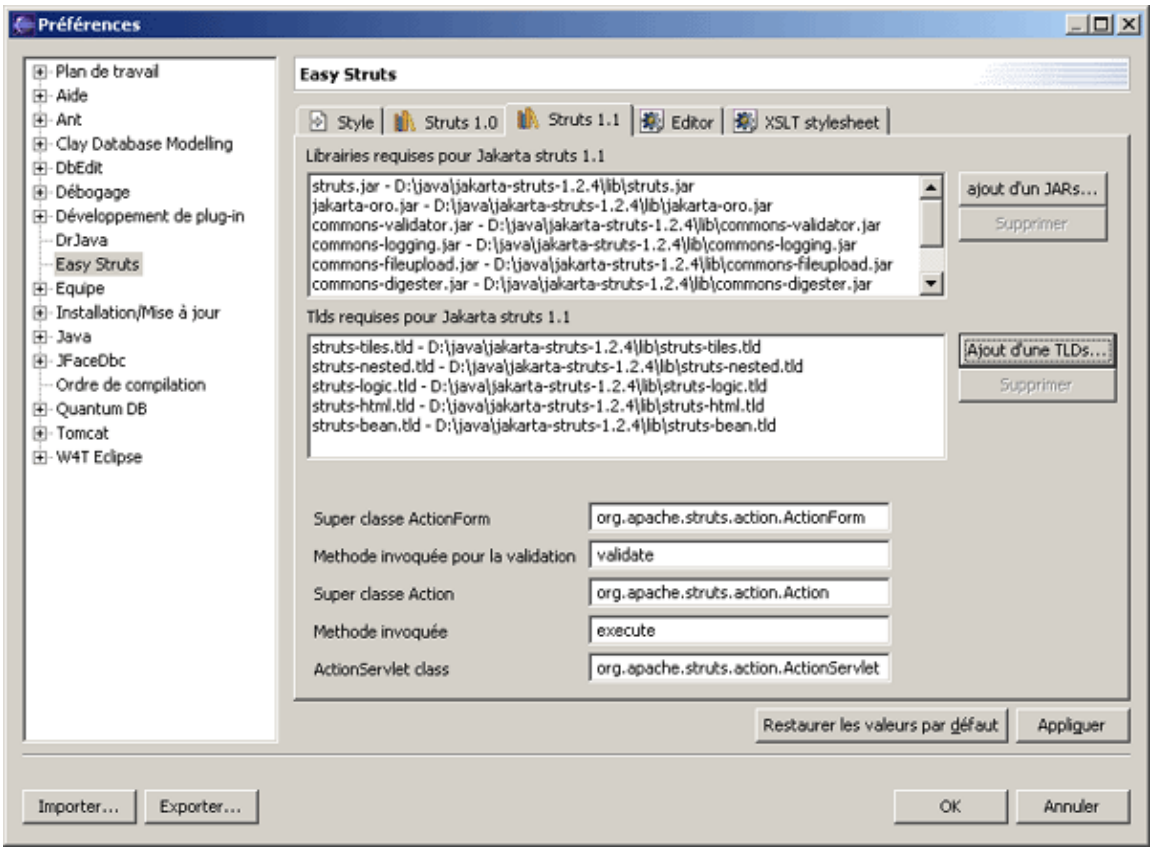
L'onglet « Style » permet de définir des comportements par défaut.



L'onglet « Struts 1.0 » et « Struts 1.1 » permet de fournir l'emplacement requis par la version de Struts désignée par l'onglet.

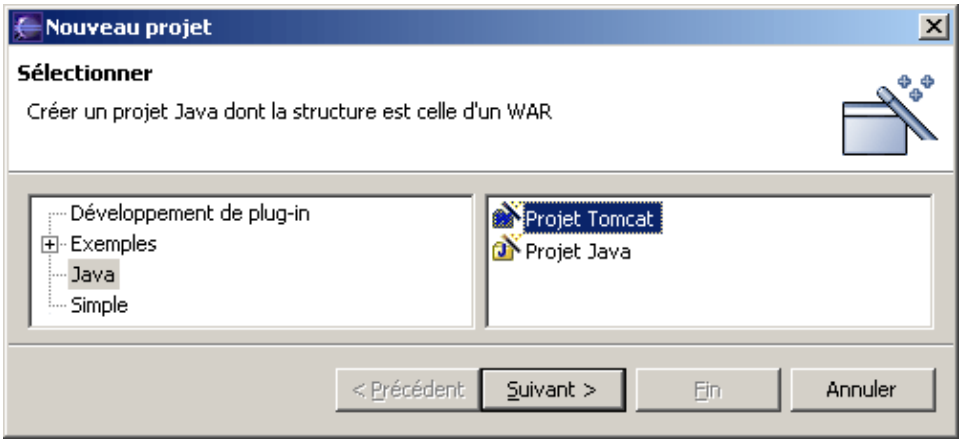


Il faut impérativement ajouter les fichiers .jar requis pas Struts ainsi que les fichier .tld des bibliothèques personnalisées de Struts.

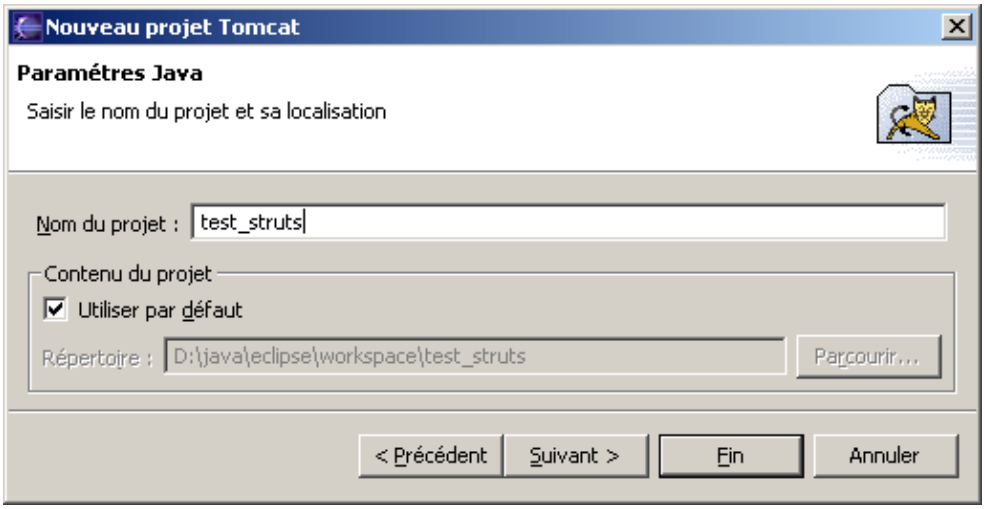


Une fois tous les paramètres renseignés, cliquez sur le bouton « Appliquer » puis sur le bouton « OK ».

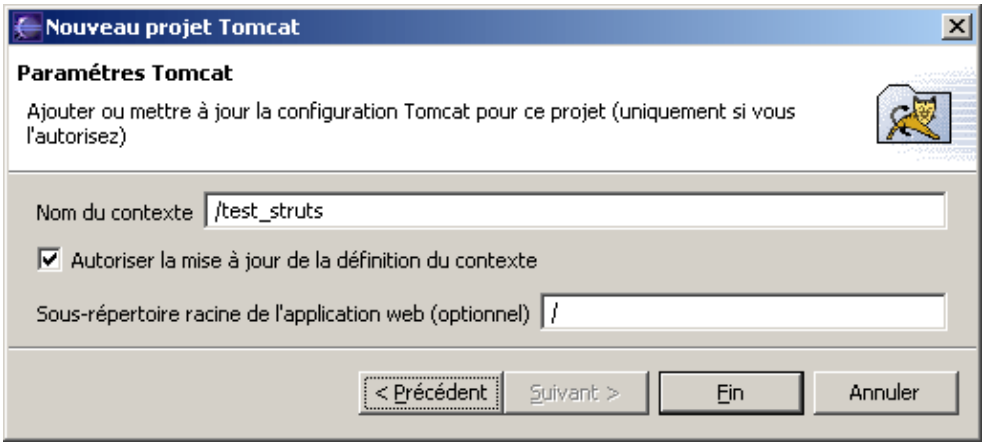
Il faut créer un projet de type « Java/Projet Tomcat » qui va contenir l'application web.



Cliquez sur le bouton « Suivant »

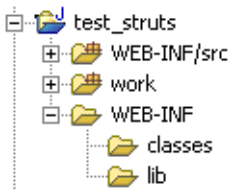


Saisissez le nom du projet, par exemple « test_struts » et cliquez sur le bouton « Suivant ».

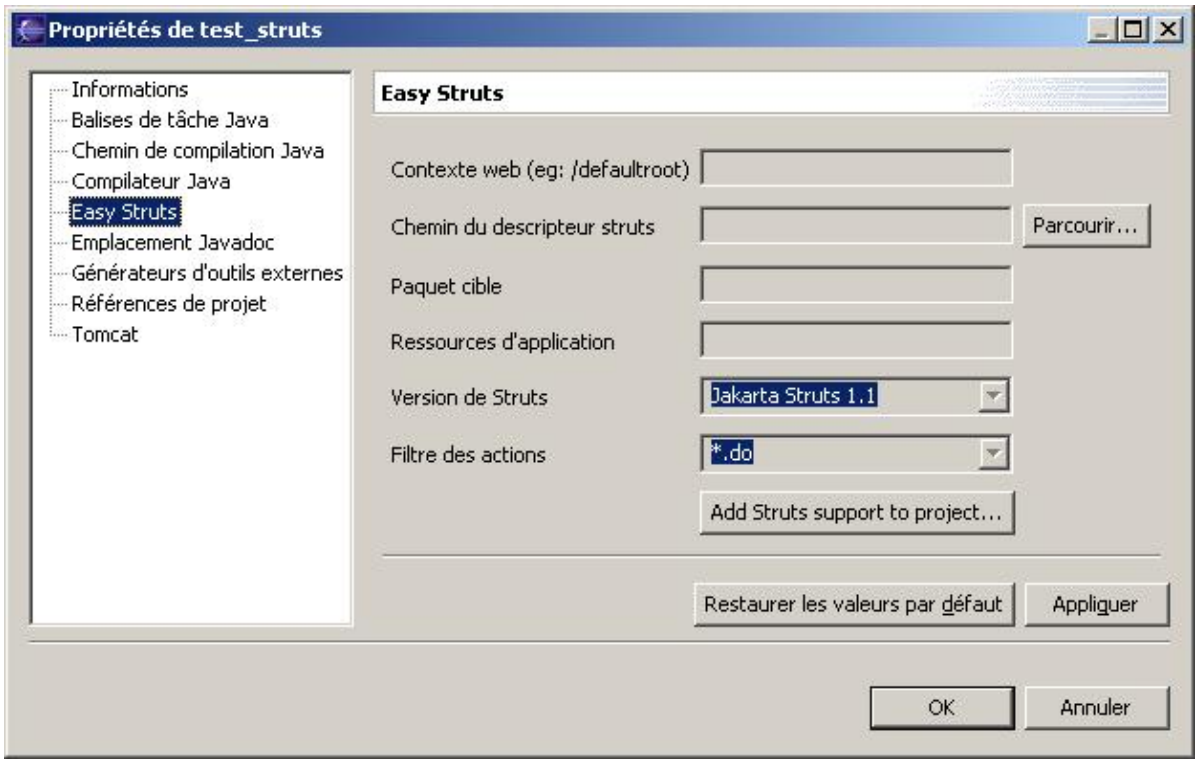


Cliquez sur le bouton « Fin »

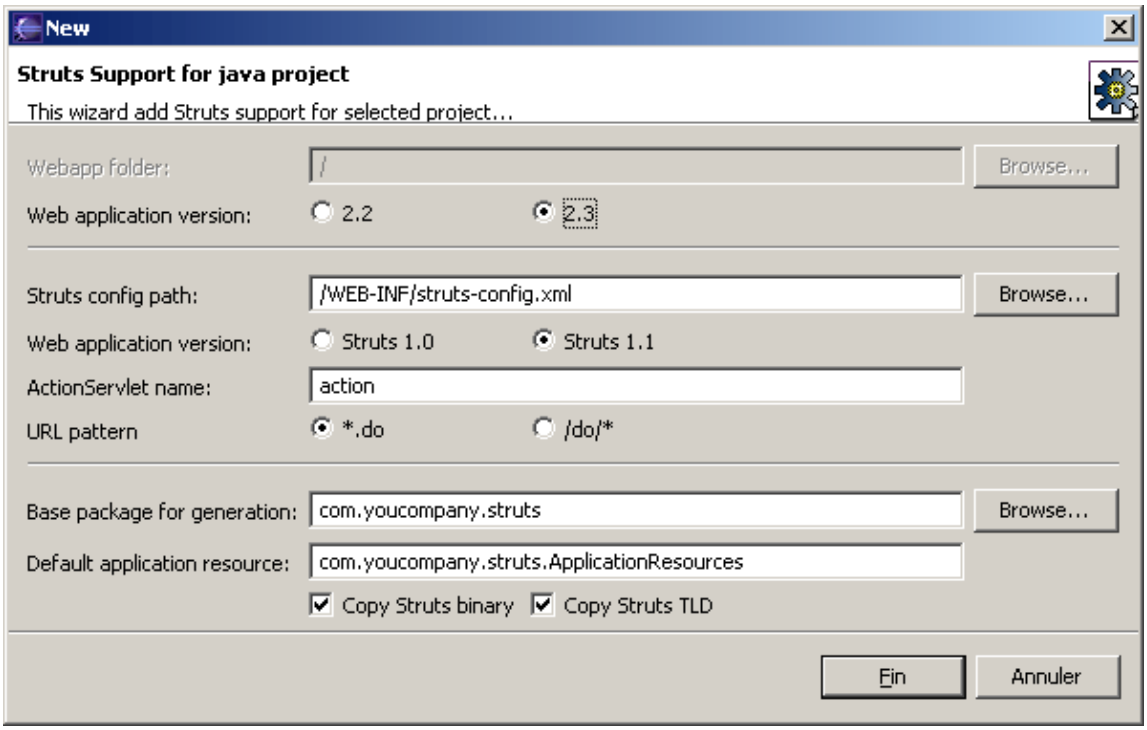
La structure du projet est la suivante :



Il faut sélectionner les propriétés du projet. Sélectionnez l'option « Easy Struts »

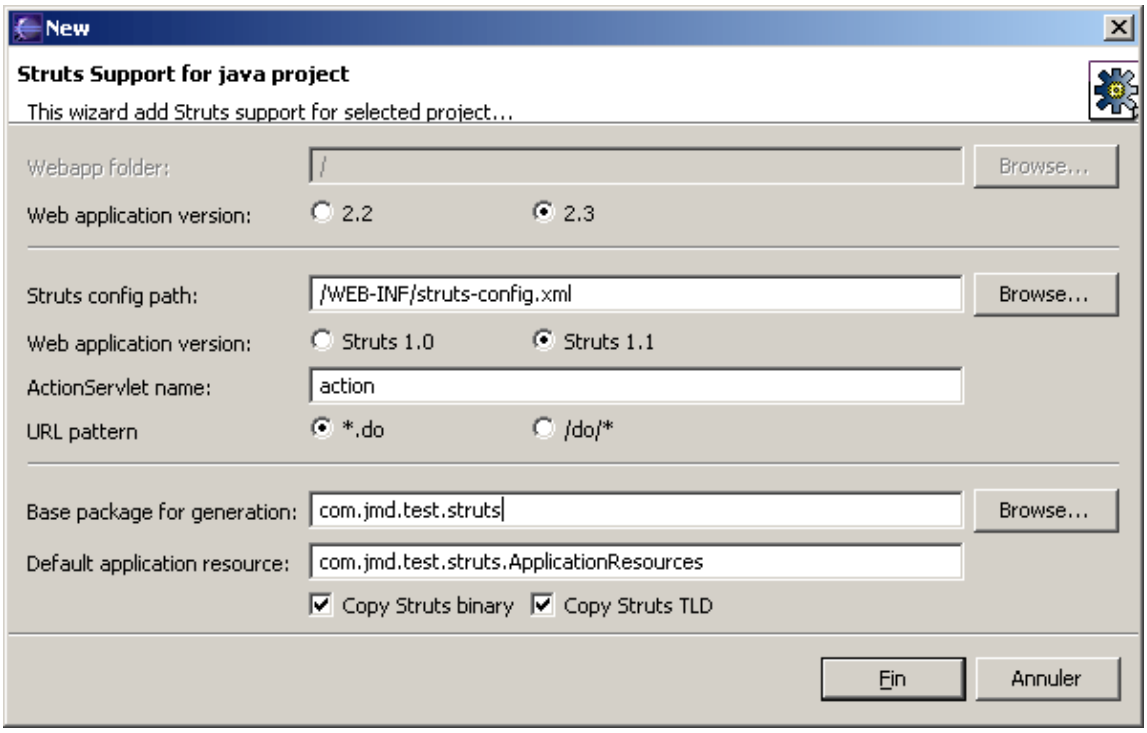


Cliquez sur le bouton « Add Struts support to project ... ». Une boîte de dialogue s'ouvre pour permettre la sélection des options

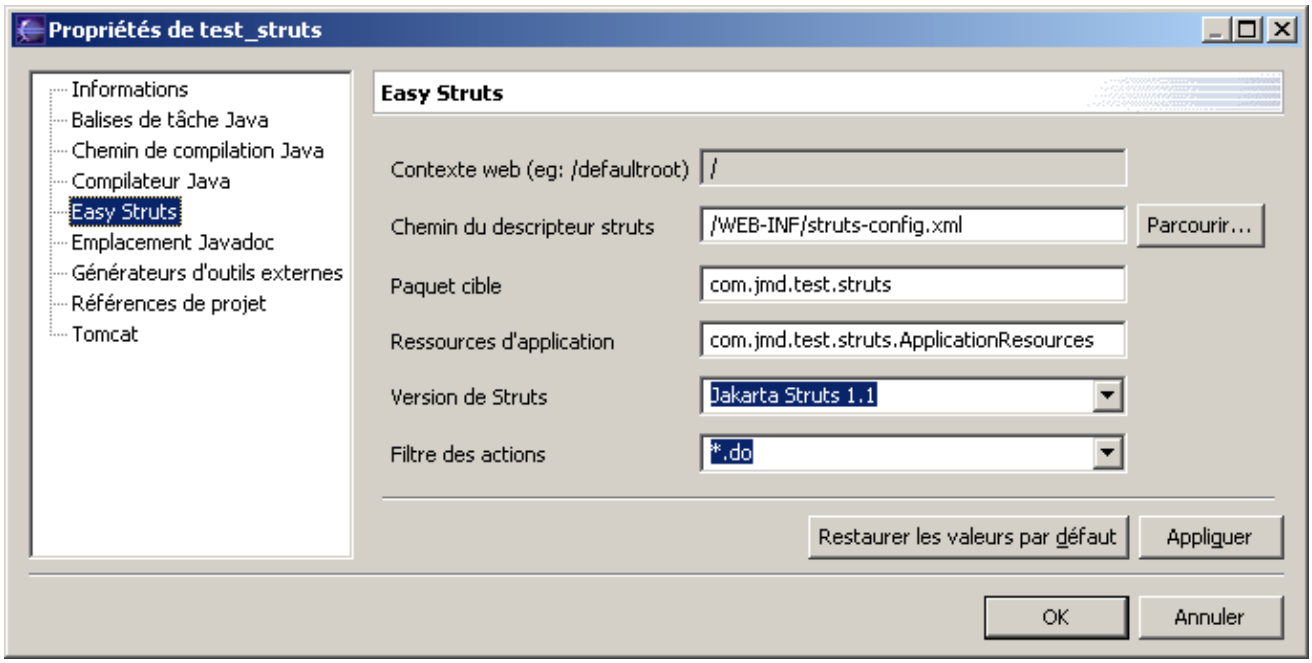


Si les paramètres ne sont pas correctement renseignés, un message d'erreur est affiché

Il saisir les informations nécessaire à la configuration de l'application.



Renseignez les informations nécessaires et cliquez sur le bouton « Fin ». Les fichiers sont copiés et les propriétés sont mises à jour.



Cliquez sur le bouton « OK ».

Le fichier web.xml généré est le suivant :

Exemple :

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
    "http://java.sun.com/j2ee/dtds/web-app_2_3.dtd">
<web-app>
    <servlet>
        <servlet-name>action</servlet-name>
        <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
        <init-param>
            <param-name>config</param-name>
```

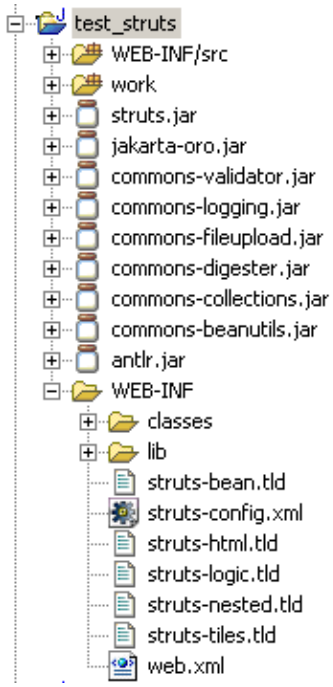
```
            <param-value>/WEB-INF/struts-config.xml</param-value>
        </init-param>
        <init-param>
            <param-name>debug</param-name>
            <param-value>3</param-value>
        </init-param>
        <init-param>
            <param-name>detail</param-name>
            <param-value>3</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>action</servlet-name>
        <url-pattern>*.do</url-pattern>
    </servlet-mapping>
</web-app>
```

Le fichier struts-config.xml généré est le suivant :

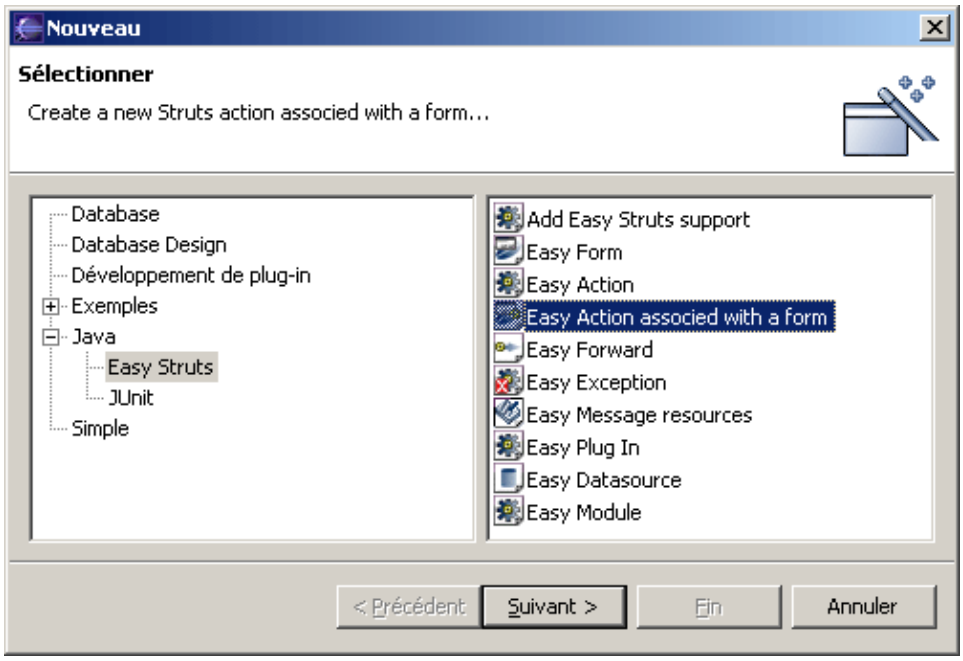
Exemple :

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts-config PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 1.1//EN"
    "http://jakarta.apache.org/struts/dtds/struts-config_1_1.dtd">
<struts-config>
    <data-sources />
    <form-beans />
    <global-exceptions />
    <global-forwards />
    <action-mappings />
    <controller />
    <message-resources parameter="com.jmd.test.struts.ApplicationResources" />
</struts-config>
```

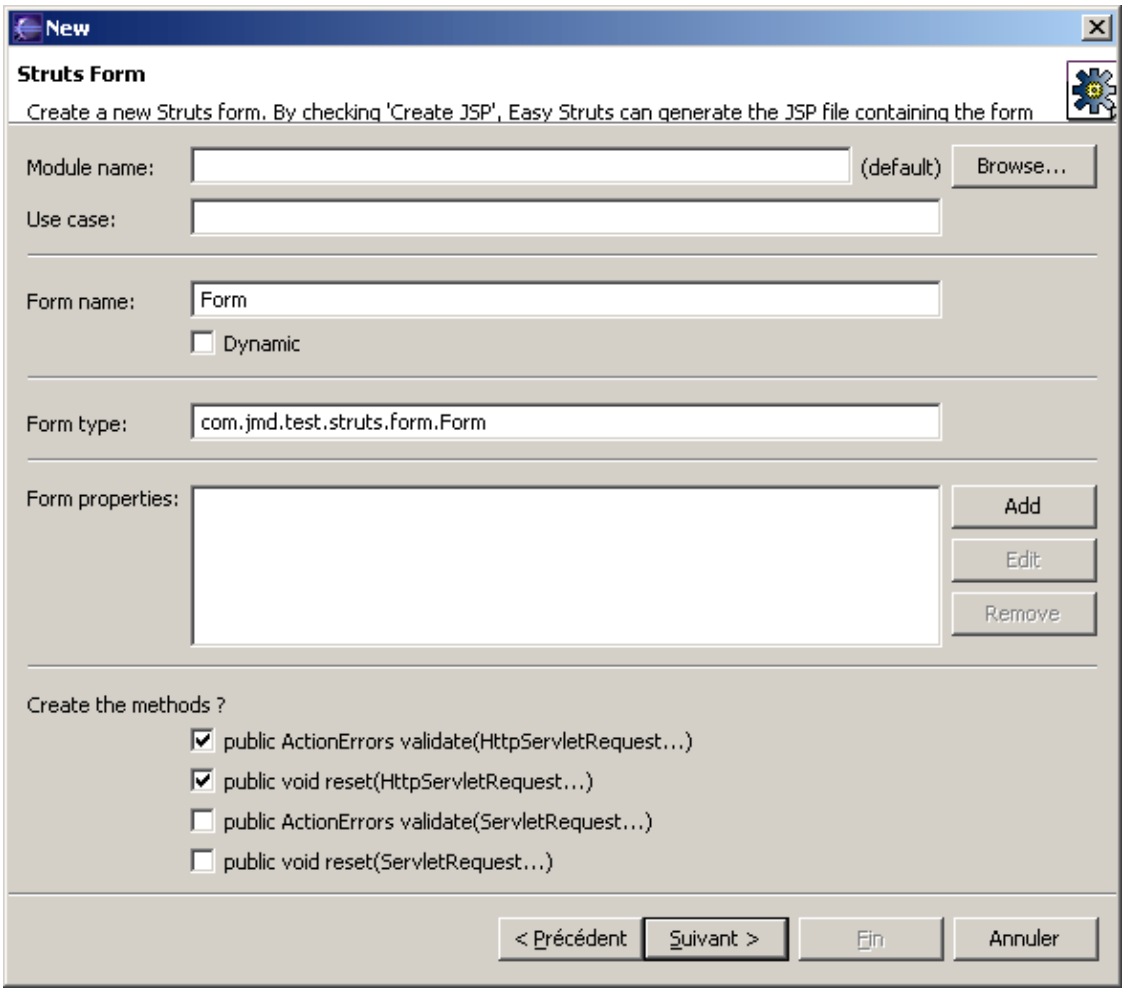
La structure du projet est alors la suivante :



Il faut ensuite créer une nouvelle entité de type « Java / Easy Struts / Easy Action associated with form ».

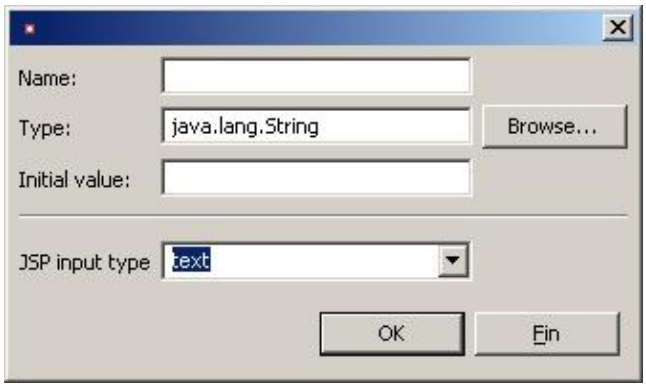


Cliquez sur le bouton « Suivant ». La page suivante permet de renseigner les caractéristiques de la page.



Renseignez le « Use case » avec le nom logique de la page, par exemple « Login ». Automatiquement les champs Form name et Form Type sont pré−renseignés en fonction du « Use Case » saisi et des propriétés saisies pour Struts.

Pour ajouter des propriétés, il suffit de cliquer sur le bouton « Add »

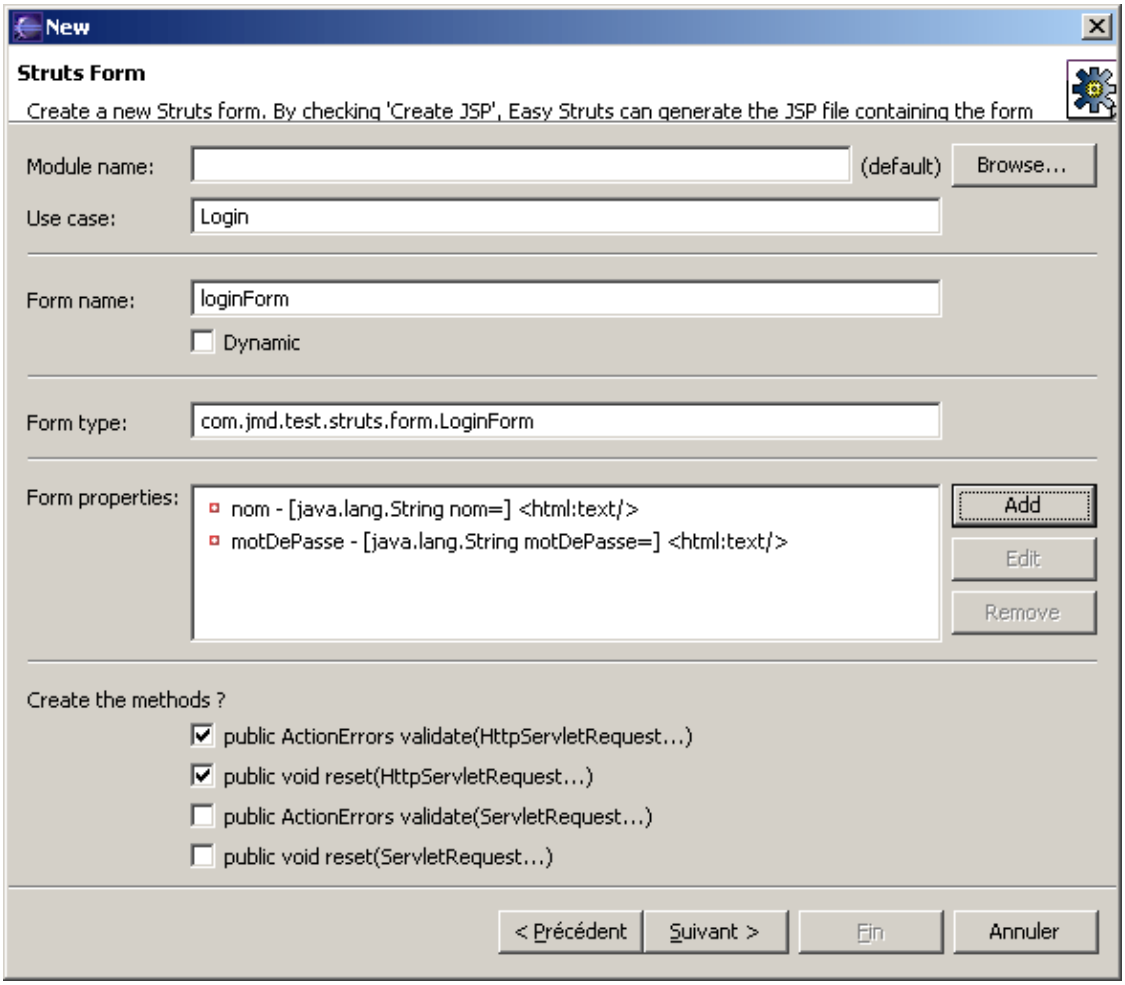


Il suffit alors de renseigner les caractéristiques de la donnée : le nom, le type, la valeur initiale et le mode de saisie dans la JSP.

Pour valider chaque donnée, il faut cliquer sur le bouton « OK ». Une fois la saisie terminée, il faut cliquer sur le bouton « Fin ».

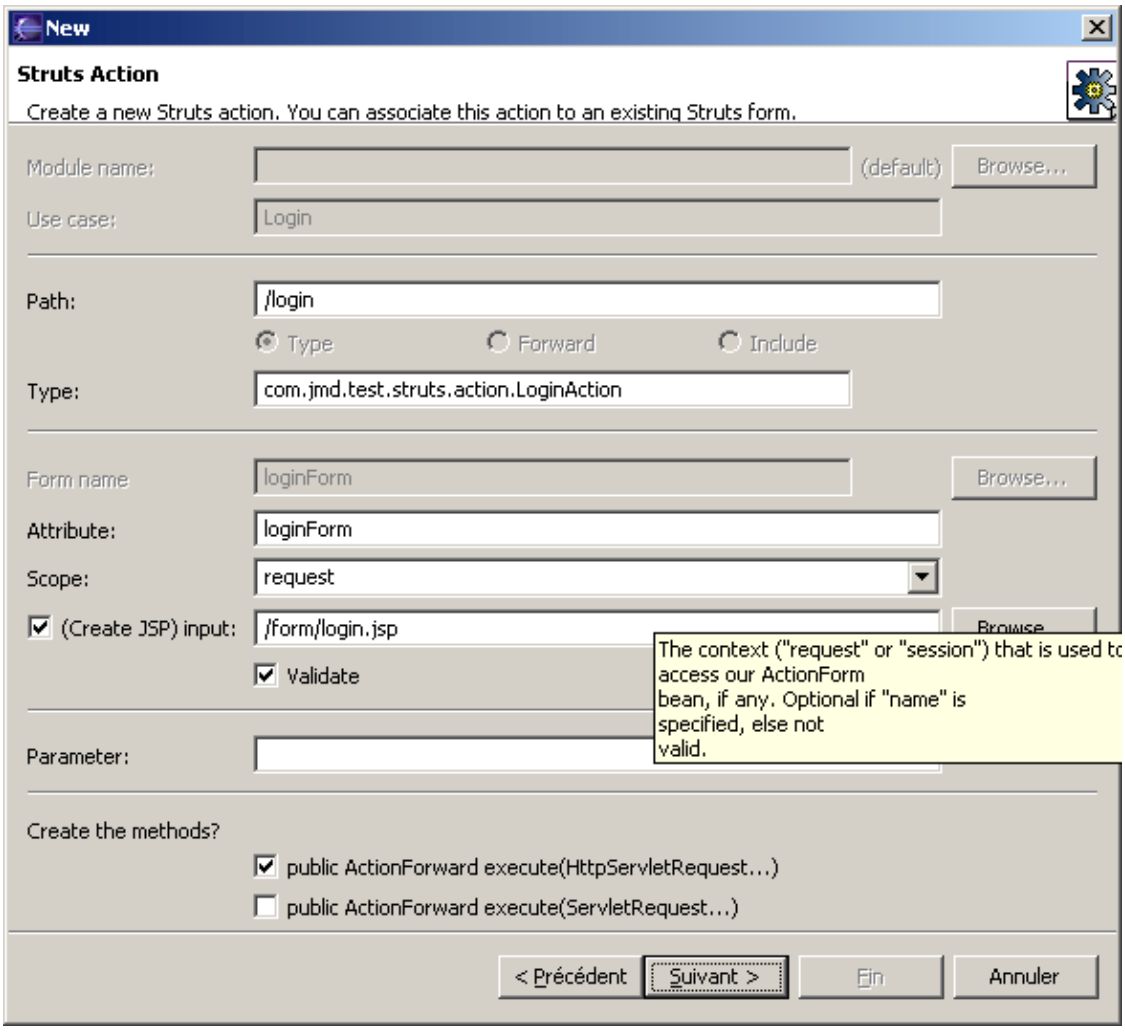
Dans l'exemple de cette section, deux données sont ajoutées :

- nom de type String
- motDePasse de type String



Cliquez sur le bouton « suivant »

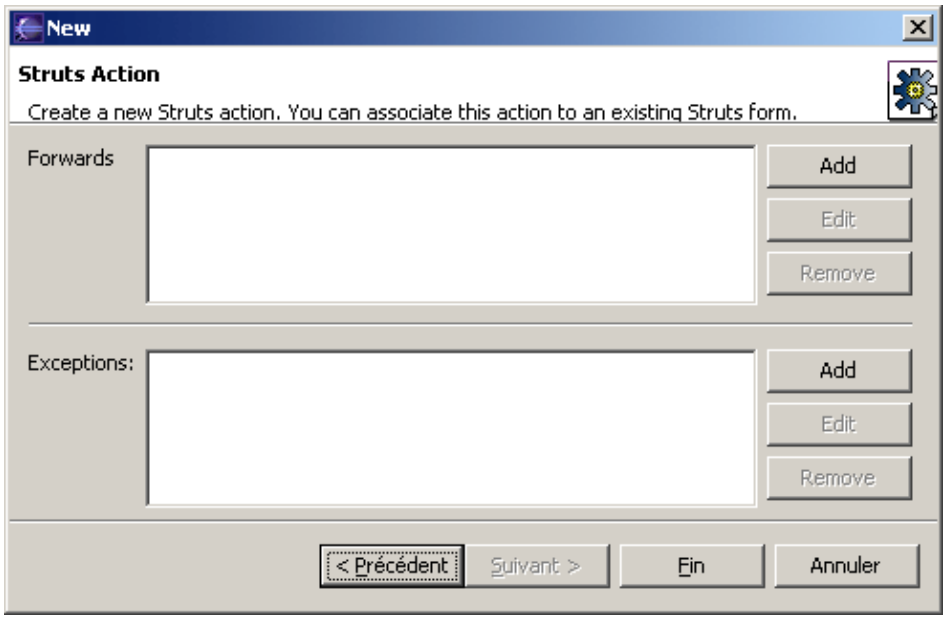
La page suivante permet de saisir les renseignements sur la classe Action qui sera générée.



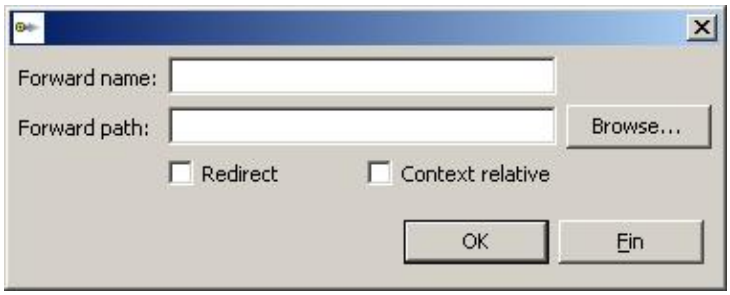
Les divers renseignements saisis vont permettre de créer la JSP et de modifier le fichier struts−config.xml

Une fois les données saisies, cliquez sur le bouton « Suivant ».

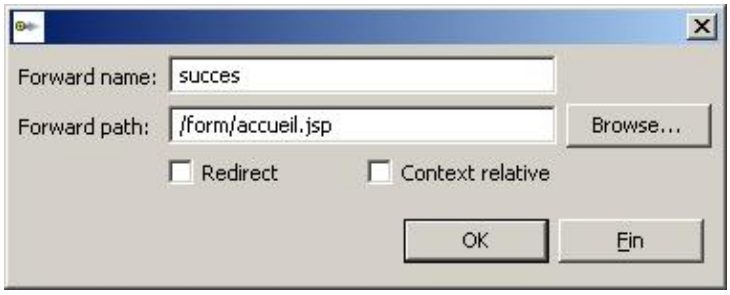
La page suivante permet de préciser les renvois et les captures d'exceptions associés pour l'Action.



Pour ajouter un renvoi (forward), il faut cliquer sur le bouton « Add » correspondant.

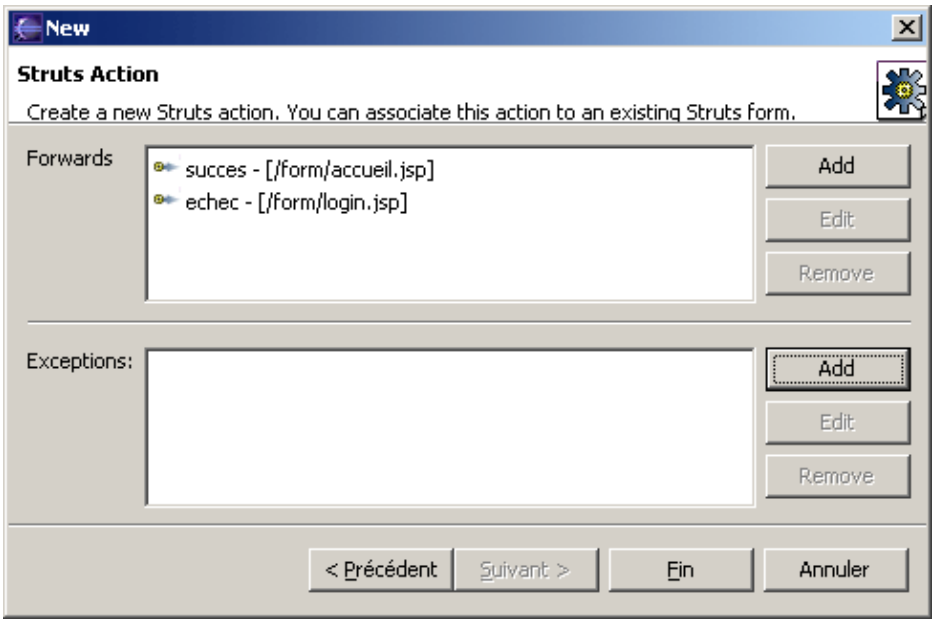


Il faut alors saisir le nom et le chemin du renvoi.



Une fois les renvois ajoutés, cliquez sur le bouton « Fin ».

Si nécessaire, procédez de la même façon avec les captures d'exceptions.



Cliquez sur le bouton « Fin ».

Le fichier struts−config.xml est modifié pour tenir compte des éléments paramétrés dans l'assistant.

Exemple :

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts-config PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 1.1//EN"
    "http://jakarta.apache.org/struts/dtds/struts-config_1_1.dtd">
<struts-config>
    <!-- ========== Data Source Configuration =============================== -->
    <data-sources />
    <!-- ========== Form Bean Definitions =================================== -->
    <form-beans>
        <form-bean name="loginForm" type="com.jmd.test.struts.form.LoginForm">
            <form-property name="motDePasse" type="java.lang.String" />
```

```
            <form-property name="nom" type="java.lang.String" />
        </form-bean>
    </form-beans>
    <!-- ========== Global Exception Definitions ============================== -->
    <global-exceptions />
    <!-- ========== Global Forward Definitions ============================== -->
    <global-forwards />
    <!-- ========== Action Mapping Definitions ============================== -->
    <action-mappings>
        <action
            attribute="loginForm"
            input="/form/login.jsp"
            name="loginForm"
            path="/login"
            type="com.jmd.test.struts.action.LoginAction">
            <forward name="succes" path="/form/accueil.jsp" />
            <forward name="echec" path="/form/login.jsp" />
        </action>
    </action-mappings>
    <!-- ========== Controller Configuration ================================== -->
    <controller />
    <!-- ========== Message Resources Definitions ============================= -->
    <message-resources parameter="com.jmd.test.struts.ApplicationResources" />
    <!-- ========== Plug Ins Configuration ================================== -->
</struts-config>
```

Le fichier login.jsp est créé dans le répertoire form

Exemple :

```
<%@ taglib uri="/WEB-INF/struts-bean.tld" prefix="bean"%>
<%@ taglib uri="/WEB-INF/struts-html.tld" prefix="html"%>
<html>
      <head>
            <meta name = "Generator" content = "Easy Struts Xslt generator for Eclipse.">
            <title>Struts Form for loginForm</title>
      </head>
      <body>
            <html:form action="/login">
                  motDePasse : <html:text property="motDePasse"/>
                  <html:errors property="motDePasse"/></br>
                  nom : <html:text property="nom"/><html:errors property="nom"/></br>
                  <html:submit/><html:cancel/>
            </html:form>
      <body>
</html>
```

Il faut modifier le fichier LoginForm.java pour mettre le code à exécuter lors de la validation des données dans la méthode validate() à la place de la levée de l'exception de type UnsupportedOperationException

Exemple :

```
 public ActionErrors validate(ActionMapping mapping, HttpServletRequest request) {
    ActionErrors erreurs = new ActionErrors();
    if (nom == null || nom.equals("")) {
      erreurs.add("nom", new ActionError("error.login.nommanquant"));
    }
    return erreurs;
  }
```

Il faut ajouter une clause d'importation sur la classe org.apache.struts.action.ActionError

Il faut modifier le fichier LoginAction.java pour mettre le code a éxécuter lors des traitements dans la méthode execute() à la pace de la levée de l'exception de type UnsupportedOperationException

Exemple :

```
 public ActionForward execute(ActionMapping mapping, ActionForm form,
          HttpServletRequest request, HttpServletResponse response) throws Exception {
    ActionForward resultat = null;
    LoginForm loginForm = (LoginForm) form;
    String nom = loginForm.getNom();
    String mdp = loginForm.getMotDePasse();
    request.setAttribute("nom", nom);
    if (nom.equals("test") && mdp.equals("test")) {
      resultat = (mapping.findForward("succes"));
    } else {
      resultat = (mapping.findForward("echec"));
    }
    return resultat;
  }
```

Il faut modifier le fichier ApplicationResources.properties se trouvant dans le repertoire WEB−INF/classes/com/jmd/test/struts

Exemple :

```
# Resources for parameter 'com.jmd.test.struts.ApplicationResources'
 # Project P/test_struts
 errors.header=<ul>
 errors.footer=</ul>
 error.login.nommanquant=<li>La saisie du nom de l'utilisateur est obligatoire</li>
```

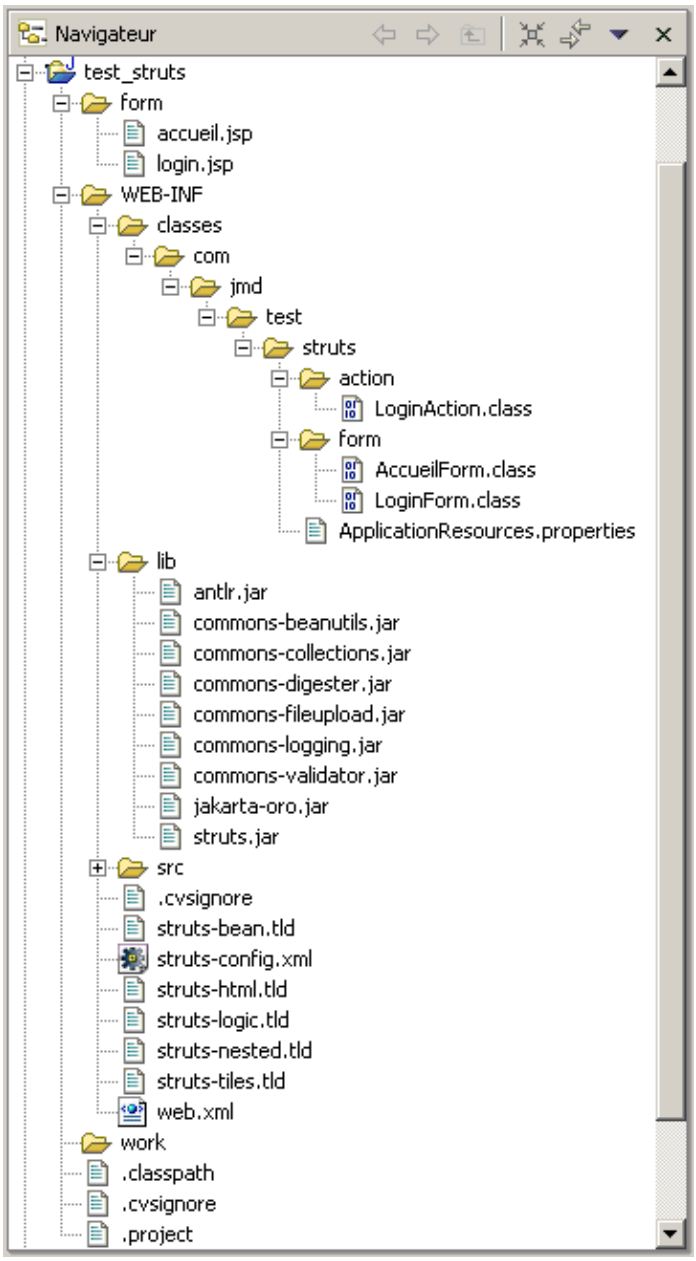
Il ensuite créer une nouvele entité de type type « Java / Easy Struts / Easy Form » dont le « Use Case » sera « Accueil ».

Il faut modifier le fichier form/accueil.jsp généré.

Exemple :

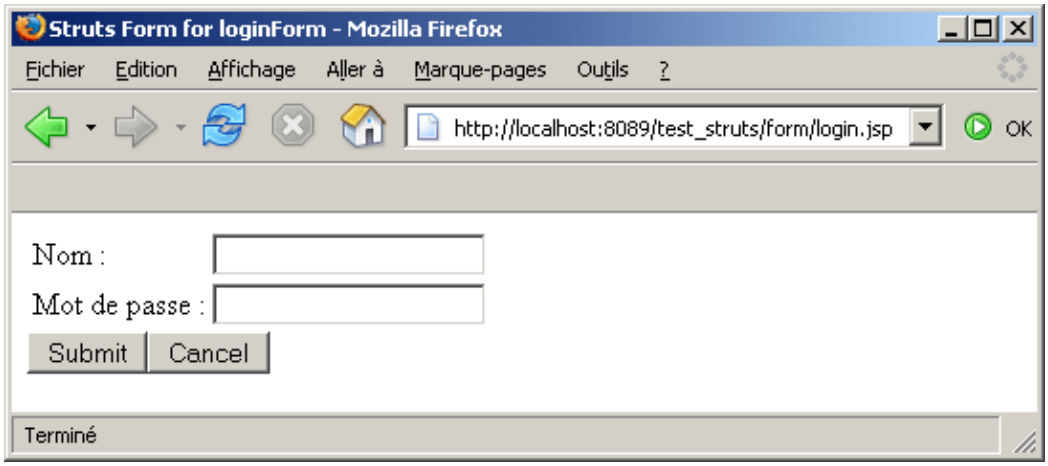
```
<%@ taglib uri="/WEB−INF/struts−bean.tld" prefix="bean"%>
<%@ taglib uri="/WEB−INF/struts−html.tld" prefix="html"%>
<html>
      <head>
            <meta name = "Generator" content = "Easy Struts Xslt generator for Eclipse.">
            <title>Struts Form for accueilForm</title>
      </head>
      <body>
            <H1>Bienvenue<logic:present name="nom" scope="request">
<bean:write name="nom" scope="request"/>
</logic:present></H1>
      <body>
</html>
```

Les fichiers composants l'application sont les suivants :

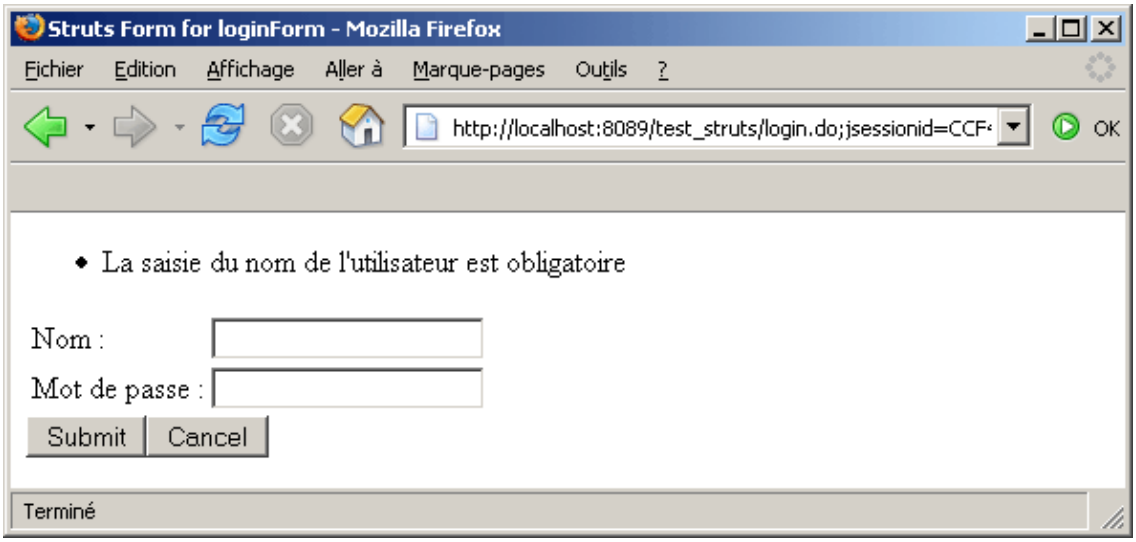


19.3. L'exécution de l'application

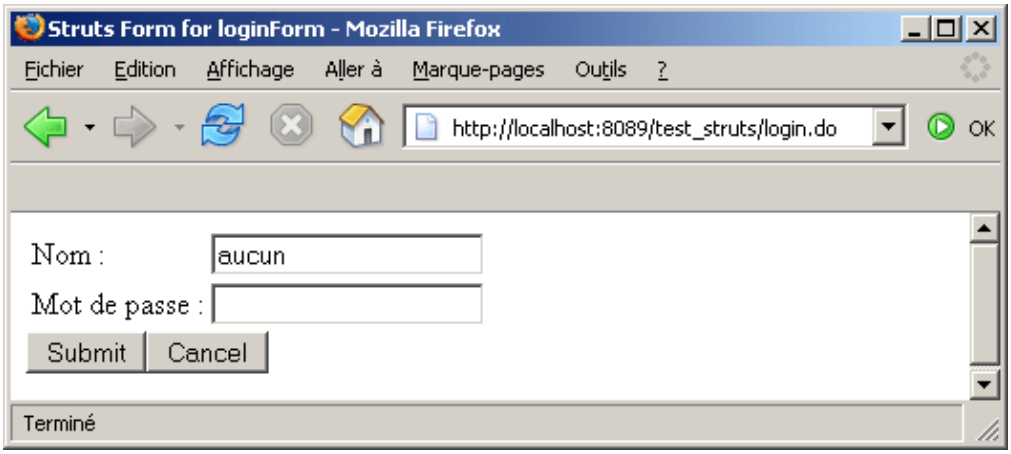
Il faut lancer Tomcat en cliquant sur le bouton 



Un clic sur le bouton « Submit » sans saisir de nom affiche le message d'erreur



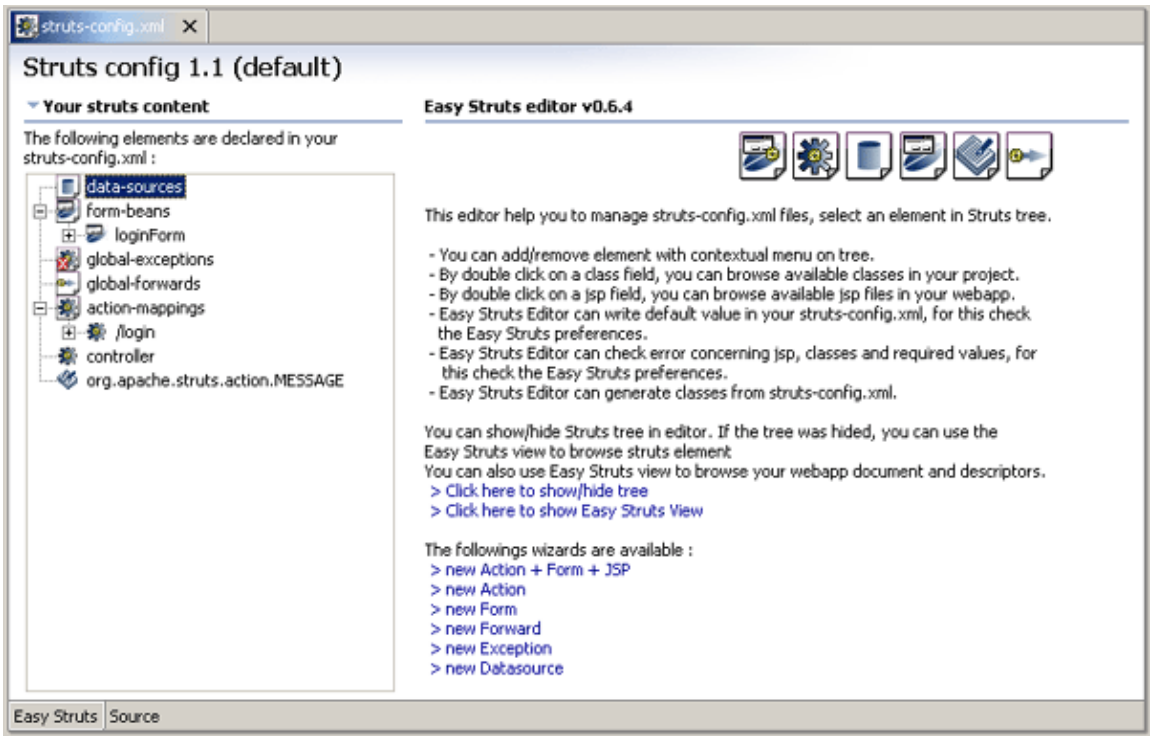
La validation avec un nom différent de test réaffiche la page



La validation avec le nom « test » et le mot de passe « test » affiche la page d'accueil

19.4. La modification du fichier struts−config.xml

EasyStrusts propose un éditeur dédié à la mise à jour du fichier de configuration de Struts

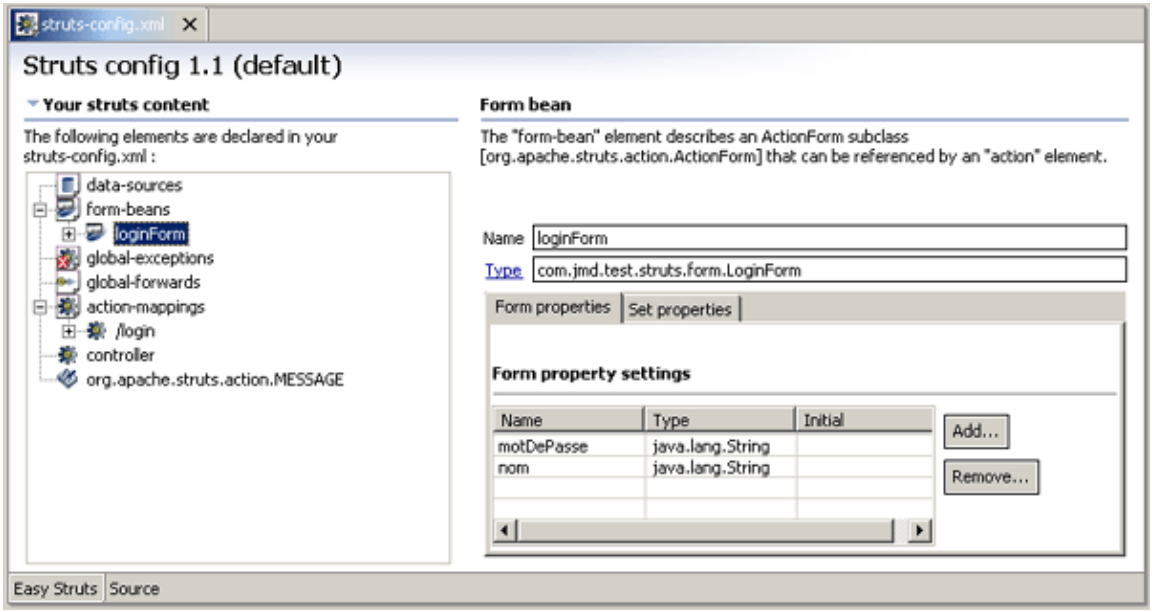


Cet éditeur propose deux onglets : « Easy Struts » pour modifier les informations avec une interface et « Source » qui permet de visualiser et de modifier directement le fichier xml.

L'onglet « Easy Struts » de compose de deux parties :

La partie de gauche permet une navigation dans les élements qui composent le fichier xml et la sélection d'un élément.

La partie de droite permet de modifier l'élément sélectionné.



20. Le développement d'interfaces graphiques

La suite de ce chapitre sera développée dans une version future de ce document

20.1. Eclipse et SWT

Eclipse est développé en utilisant SWT comme API pour le développement de son interface graphique pour l'utilisateur. Pour le développement de plug−ins, l'utilisation de SWT est donc fortement recommandée mais il est aussi possible d'utiliser Eclipse et SWT pour le développement des applications standalone.

20.1.1. Configurer Eclipse pour développer des applications SWT

Sans plug−ins, il est possible de développer des applications graphiques utilisant SWT avec Eclipse. Bien sûre, l'inconvénient majeur est de devoir écrire tout le code "à la main ".

Pour le développement d'applications SWT, il est nécessaire de configurer certains éléments notamment le classpath. Dans cette section, la version de SWT utilisée est la 2.1.

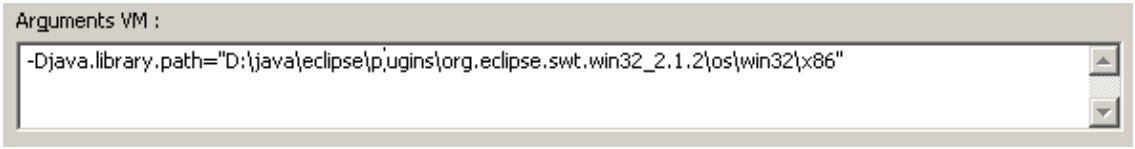
Pour utiliser SWT dans un projet, il faut ajouter dans son classpath le fichier swt.jar. Dans les propriétés du projet, sélectionnez « Chemin de compilation Java » et cliquez sur le bouton « Ajouter des fichiers jar externes ... ».

Une boîte de dialogue permet de sélectionner le fichier swt.jar. Ce fichier est dépendant du système d'exploitation sur lequel l'application va s'exécuter. Par exemple sous Windows, ce fichier se trouve dans le sous répertoire plugins\org.eclipse.swt.win32_2.1.0\ws\win32\ du répertoire d'installation d'Eclipse. Cliquez sur le bouton « Ok » pour valider les modifications et recompiler le projet.

Pour pouvoir exécuter le projet, il faut que l'application puisse accéder à une bibliothèque native particulière qui sous Windows se nomme swt−win32−2135.dll. Ce fichier se trouve dans le sous répertoire plugins\org.eclipse.swt.win32_2.1.2\os\win32\x86 du répertoire d'installation d'Eclipse. Le nom et le chemin de cette bibliothèque dépend de la version de SWT qui est utilisée.

Il y a plusieurs solutions possibles pour mettre ce fichier à disposition d'une application :

1. précisez le chemin de la bibliothèque dans les paramètres de la machine virtuelle lors de l'exécution
   Pour cela choisissez, « Exécuter ... » pour ouvrir la boîte de dialogue de configuration d'exécution. Sur l'onglet « Arguments ... », saisir −Djava.library.path="chemin_complet_du_fichier" et cliquez sur le bouton « Appliquer » puis sur le bouton « Exécuter »



L'inconvénient de cette méthode est que cette configuration doit être effectuée pour chaque configuration de lancement.

2. Ajouter le répertoire qui contient la bibliothèque à la variable PATH sous Windows ou LD_LIBRAY_PATH sous Unix.

3. Il est possible de copier la bibliothèque dans un répertoire contenu dans la variable java.library.path. L'inconvénient de cette méthode est qu'il faut recopier la bibliothèque à chaque nouvelle version de SWT utilisée.

4. Il est possible de copier la bibliothèque dans le répertoire racine de l'application. L'inconvénient de cette méthode est qu'il faut recopier la bibliothèque dans chaque projet qui utilise SWT.

Lors de l'exécution, si la bibliothèque ne peut être accédée par l'application, une exception est levée.

Exemple :

```
java.lang.UnsatisfiedLinkError: no swt-win32-2135 in java.library.path
```

Pour une utilisation sous Linux, la configuration est similaire. Il est cependant nécessaire d'ajouter dans le classpath une bibliothèque supplémentaire nommée swt-pi.jar. Ce fichier ce trouve au même endroit que le fichier swt.jar dans le répertoire eclipse/plugins/org.eclipse.swt.gtk_3.0.0/gs/gtk/

Si ce fichier n'est pas inclus dans le classpath, une exception est levée lors de l'exécution :

Exemple :

```
Exception in thread "main" java.lang.NoClassDefFoundError: org/eclipse/swt/internal/gtk/OS
        at org.eclipse.swt.internal.Converter.wcsToMbcs(Converter.java:63)
        at org.eclipse.swt.internal.Converter.wcsToMbcs(Converter.java:54)
        at org.eclipse.swt.widgets.Display.<init>(Display.java:118)
        at com.jmd.test.swt.TestSWT.main(TestSWT.java:9)
```

Sous Linux , il faut que l'application puisse accéder à aux bibliothèques natives qui se situe dans le répertoire eclipse/plugins/org.eclipse.swt.gtk_3.0.0/os/linux/x86

Le plus simple de mettre à jour la variable d'environnement LD_PATH du système ou d'utiliser l'option de la JVM lors de l'exécution (en adaptant le chemin d'Eclipse) :
-Djava.library.path="/home/java/eclipse/plugins/org.eclipse.swt.gtk_3.0.0/os/linux/x86.

Si ces bibliothèques ne peuvent être trouvées lors de l'exécution, une exception est levée.

Exemple :

```
Exception in thread "main" java.lang.UnsatisfiedLinkError: no swt-pi-gtk-3062
  in java.library.path
        at java.lang.ClassLoader.loadLibrary(ClassLoader.java:1517)
        at java.lang.Runtime.loadLibrary0(Runtime.java:788)
        at java.lang.System.loadLibrary(System.java:834)
```

20.1.2. Un exemple très simple

Exemple :

```
import org.eclipse.swt.*;
import org.eclipse.swt.graphics.*;
import org.eclipse.swt.widgets.*;

public class TestSWT2 {

  public static void main(String[] args) {
    Display display = new Display();
    Shell shell = new Shell(display);
    shell.setText("Test");

    Composite composite = new Composite(shell, SWT.NONE);
    Color couleur = new Color(display,131,133,131);
    composite.setBackground(couleur);

    Label label = new Label(composite, SWT.NONE);
    label.setBackground(couleur);
    label.setText("Saisir la valeur");
    label.setBounds(10, 10, 100, 25);

    Text text = new Text(composite, SWT.BORDER);
    text.setText("mon texte");
    text.setBounds(10, 30, 100, 25);

    Button button = new Button(composite, SWT.BORDER);
    button.setText("Valider");
    button.setBounds(10, 60, 100, 25);
    composite.setSize(140,140);

    shell.pack();
    shell.open();

    while (!shell.isDisposed())
      if (!display.readAndDispatch())
        display.sleep();

    couleur.dispose();
    display.dispose();
  }
}
```

Si l'environnement est correctement configuré comme expliqué dans la section précédente, l'exécution de l'application se fait comme tout autre application.

20.2. Le plug−in Eclipse V4all

V4All est un projet open source qui a pour but de développer un plug−in pour Eclipse permettant le développement d'interfaces graphiques avec Swing ou SWT.

	Version utilisée dans cette section
Eclipse	2.1.2
J2RE	1.4.2_02
V4All	2.1.1.9.

20.2.1. Installation

V4All requière l'installation du plug−in GEF (Graphical Editing Framework) en exécutant les étapes suivantes :

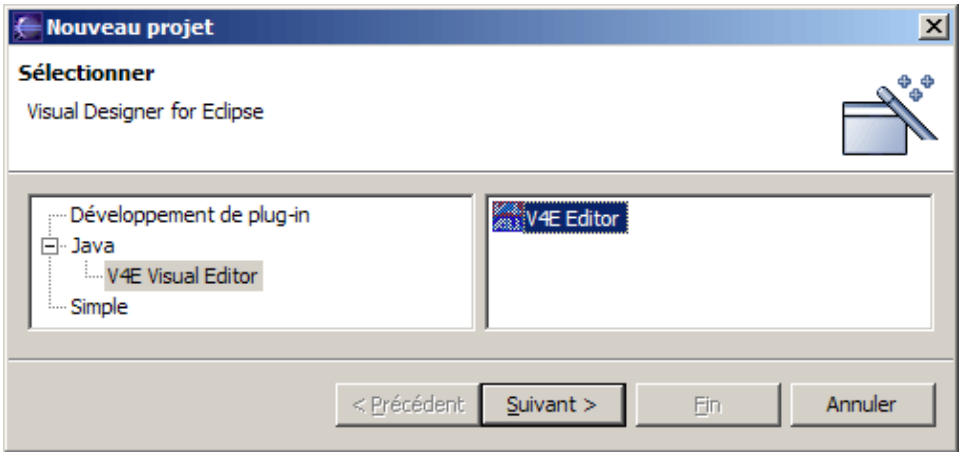
- téléchargez le fichier http://download.eclipse.org/tools/gef/downloads/drops/R−2.1.1−200306180934/GEF−runtime−I20030618.zip sur le site d'Eclipse
- décompressez ce fichier dans le répertoire qui contient Eclipse.
- relancez Eclipse et cliquez sur le bouton « Fin » pour valider les modifications apportées par le plug−in
- cliquez sur le bouton « Oui » lors de la demande de relance de l'application.

Pour installer V4All, il faut suivre les étapes suivantes :

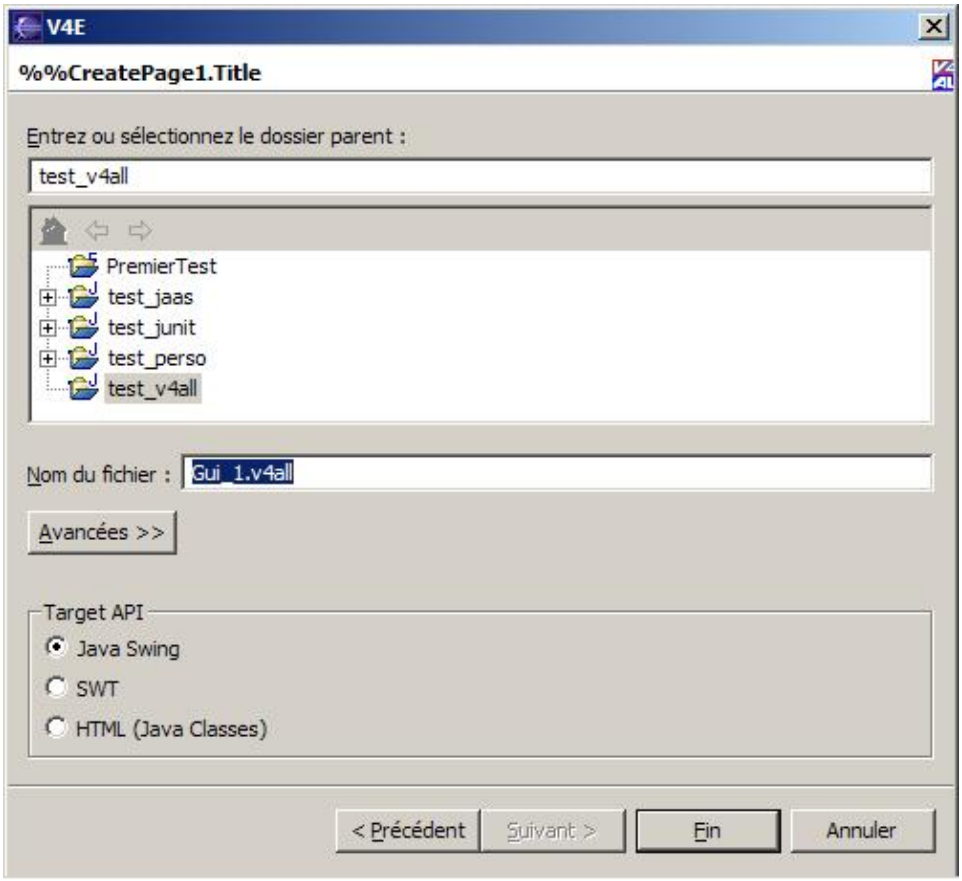
- téléchargez le fichier v4all_2_1_1_9.zip sur le site http://sourceforge.net/projects/v4all
- décompressez son contenu dans le répertoire qui contient Eclipse.
- relancez Eclipse et cliquez sur le bouton « Fin » pour valider les modifications apportées par le plug−in
- cliquez sur le bouton « Oui » lors de la demande de relance de l'application.

20.2.2. Utilisation

Pour utiliser V4all, il faut créer ou utiliser un projet Java existant, puis il faut créer un nouveau projet de type « V4E Editor ».

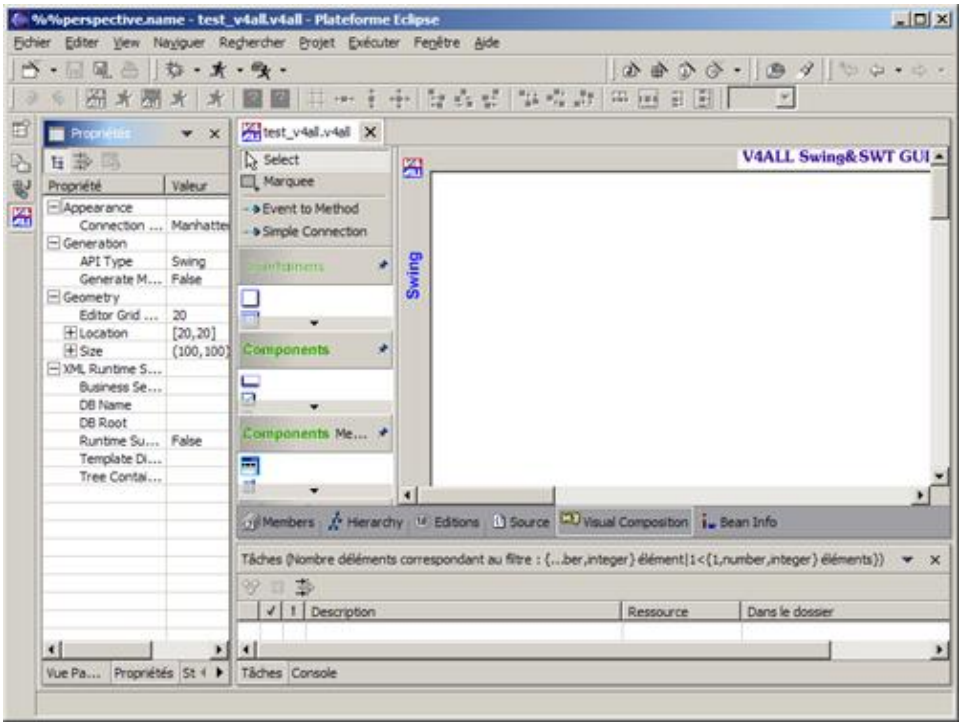


Cliquez sur le bouton « Suivant ».



Un assistant permet de sélectionner le projet Java qui va contenir le code, de saisir le nom du fichier qui va contenir les informations, et de sélectionner l'API à utiliser

Il suffit de cliquer sur le bouton « Fin » pour que la perspective dédiée à V4All s'affiche.



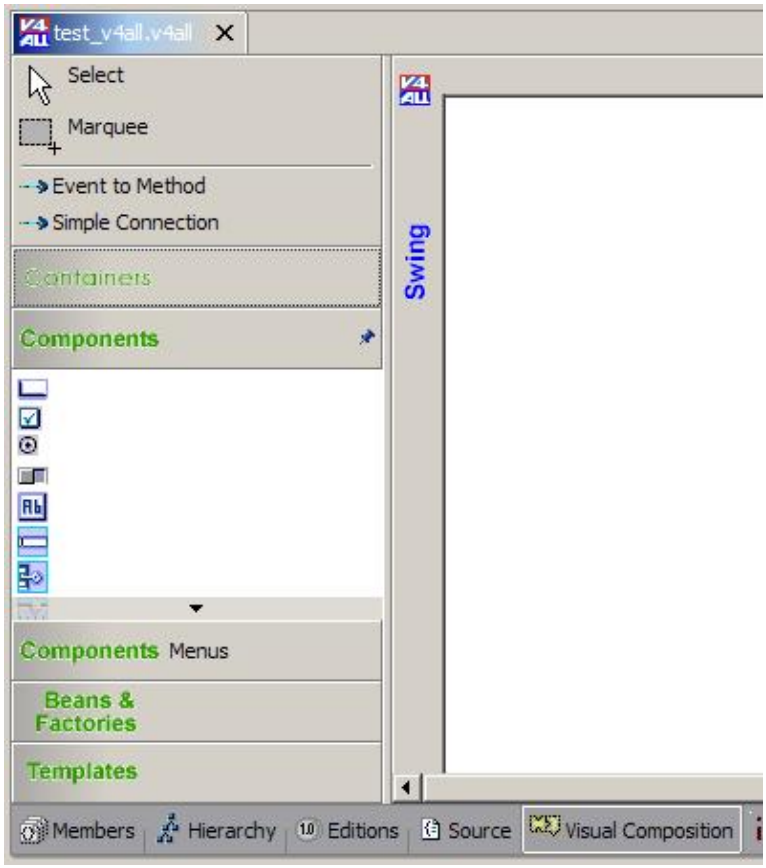
V4All propose son propre éditeur pour réaliser de manière WYSIWYG des interfaces graphiques.

Cet éditeur se compose de deux parties :

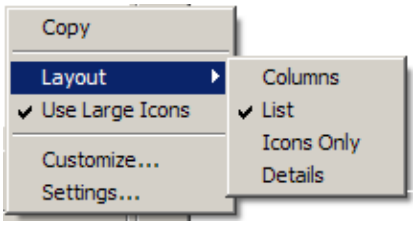
- une barre latérale qui permet de définir une action et de sélectionner un composant

- une zone de travail sur laquelle les composants sont déposés par cliquer/glisser

Par défaut, chacun des éléments de la barre est ouvert, ce qui les rend peux visibles. Pour ouvrir un seul élément, il suffit de double cliquer sur son titre.



L'apparence de la barre peut être configurée pour faciliter son utilisation : le menu contextuel de la barre propose plusieurs options intéressantes :



L'utilisation de l'option « Use large Icons » améliore la visibilité des icônes.

L'option « Layout » permet de sélectionner le mode d'affichage des composants : l'utilisation de l'option « Details » permet d'avoir un libellé associé à l'icône.

20.2.3. Un exemple simple

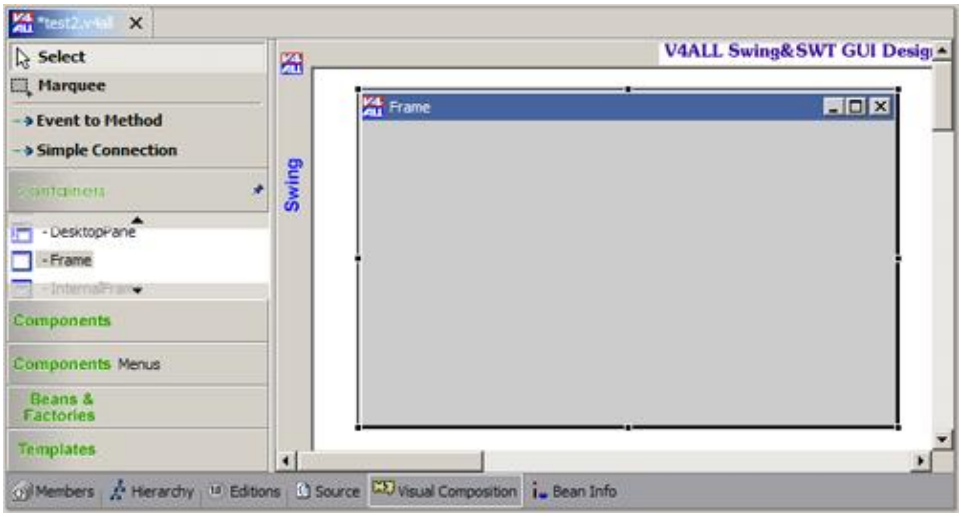
Il faut créer un nouveau projet V4All.

Dans les propriétés de ce projet, mettre la valeur « True » à la propriété « Generation / Generate Main ».

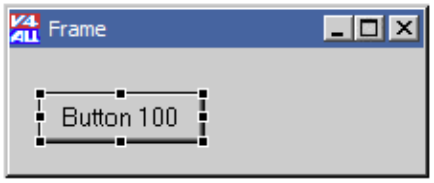
Il faut faire un cliquer/glisser du composant Frame sur la zone d'édition

A partir de ce moment, il est possible de demander la génération du code et son exécution en utilisant l'option « Run » du menu contextuel « Code generation And Run ».

L'application se lance et la fenêtre définie s'affiche : elle est déjà fonctionnelle et il suffit de cliquer sur le bouton de fermeture pour arrêter l'application.



Pour ajouter un bouton, il suffit de faire un cliquer/glisser du composant sur la Frame : le conteneur qui va recevoir le composant change de couleur.



Chaque composant possède de nombreuses propriétés qu'il est possible de modifier dans la vue "Propriétés". Cette vue affiche les propriétés du composant sélectionné dans l'éditeur.

Pour un bouton, il est par exemple possible de modifier le nom du composant avec la propriété « Basic / Bean Name », le texte du bouton avec « Component / Text », ...

Il faut procéder de la même façon pour ajouter une zone de texte (un composant de type TextField).

20.3. Eclipse VE

VE est un framework dont le but est de faciliter le développement de plug−in permettant la réalisation d'interfaces graphiques. Même si VE est proposé avec une implémentation de référence proposant la conception d'interfaces graphiques reposant sur Swing ou AWT, le but du framework est de pouvoir proposer à terme la conception d'interfaces graphiques reposant sur d'autres bibliothèques, pas nécessairement développées en ou pour Java.

	Version utilisée dans cette section
Eclipse	2.1.2
J2RE	1.4.2_02
Eclipse VE	0.5

20.3.1. Installation

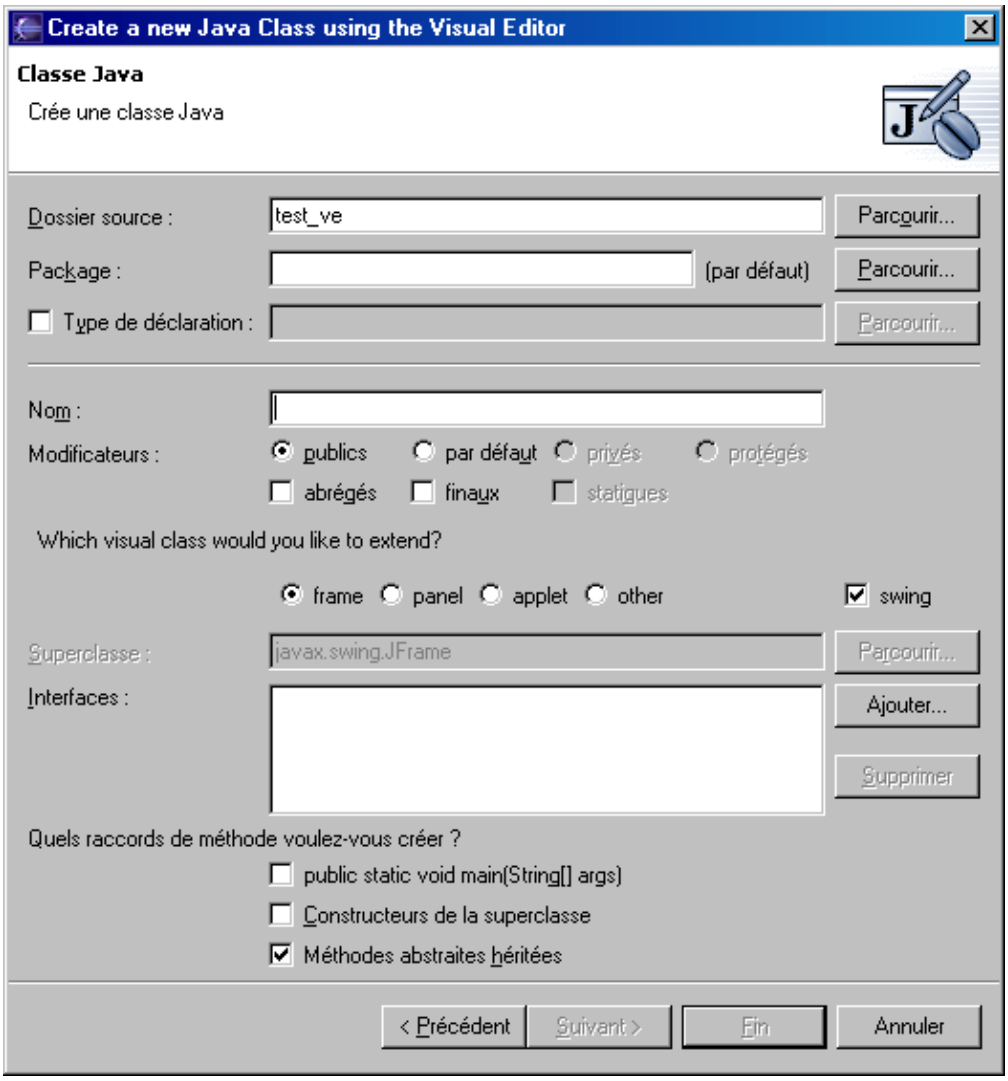
VE nécessite la version 2.1.2. minimum d'Eclipse et les plug−ins GEF version 2.1.2 et EMF version 1.1.1. Ces deux plug−ins sont à télécharger et à installer avant d'installer VE si ils ne sont pas déjà installés.

Il faut télécharger le fichier VE−runtime−0.5.0RC2.zip et le décompresser dans le répertoire d'Eclipse, comme pour installer n'importe quel autre plug−in.

20.3.2. Mise en oeuvre et présentation rapide

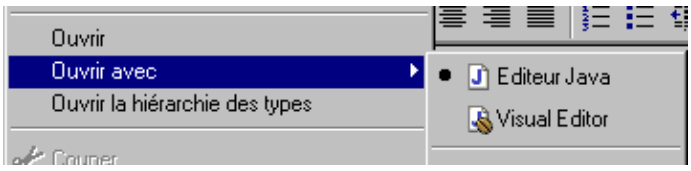
Dans un projet Java existant, il faut créer une nouvelle entité de type « Java / Visual Class ».

La page de l'assistant permet de fournir des informations sur la classe Java qui sera générée.

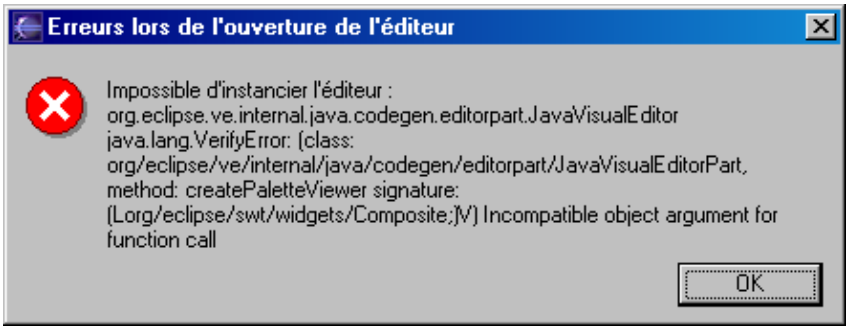


Il faut saisir le nom de la classe et renseigner les informations utiles, puis cliquer sur le bouton « Fin ».

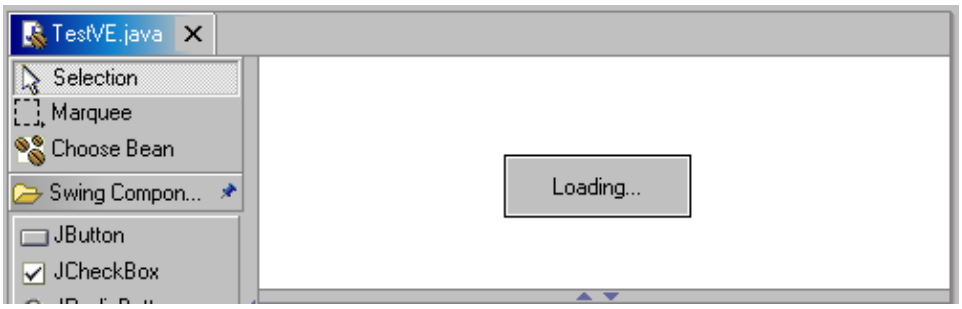
L'assistant a généré une classe Java. Il est possible d'éditer cette classe avec l'éditeur de code Java ou grâce au plug−in VE de l'éditer avec un éditeur particulier, le « Visual Editor ».



Lors du lancement de cet éditeur visuel, si l'erreur ci dessous s'affiche, c'est que la version du plug−in GEF requise n'est pas installée avec Eclipse.



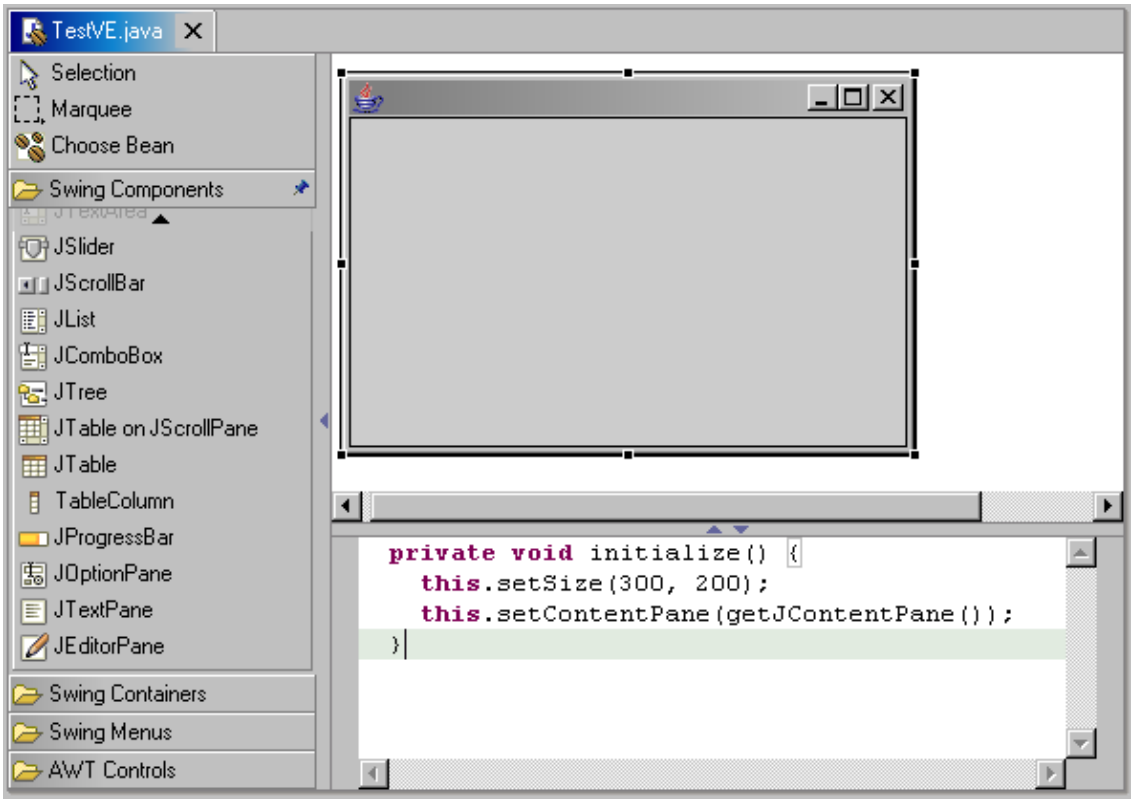
L'éditeur se charge et s'adapte en fonction du code de la classe en cours d'édition.



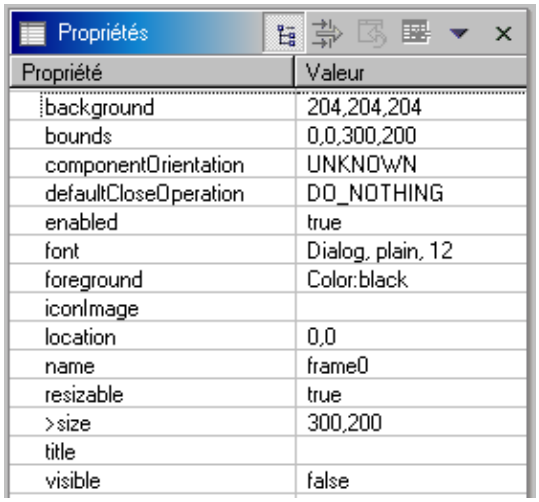
L'éditeur se compose de trois parties :

- A gauche, une barre de menu qui permet de sélectionner les composants à utiliser, regroupés par thèmes
- En haut à droite, un éditeur Wysiwyg du composant en cours de développement
- En bas à droite, un éditeur de code qui affiche le code java correspondant

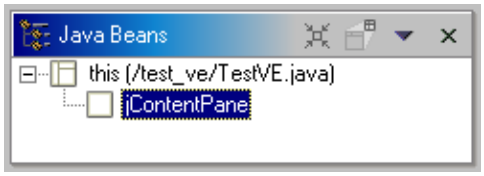
La sélection d'un composant dans l'éditeur Wysiwyg affiche le code correspondant dans l'éditeur de code.



La vue « Propriétés » affiche les propriétés du composant actuellement sélectionné dans l'éditeur.



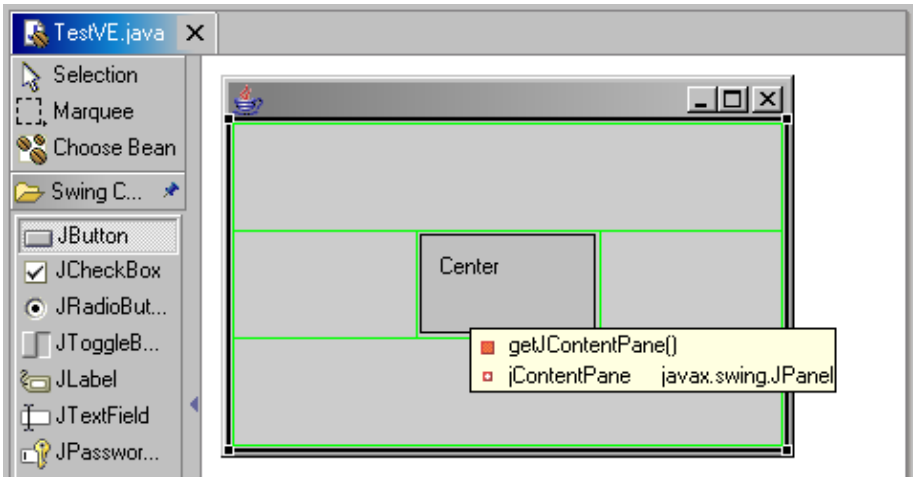
La vue « Java Beans » affiche sous la forme d'une arborescence les différents éléments qui constituent le composant en cours de développement.



La sélection d'un élément dans l'arborescence sélectionne le composant dans l'éditeur Wysiwyg et ces propriétés sont affichées dans la vue « Propriétés ».

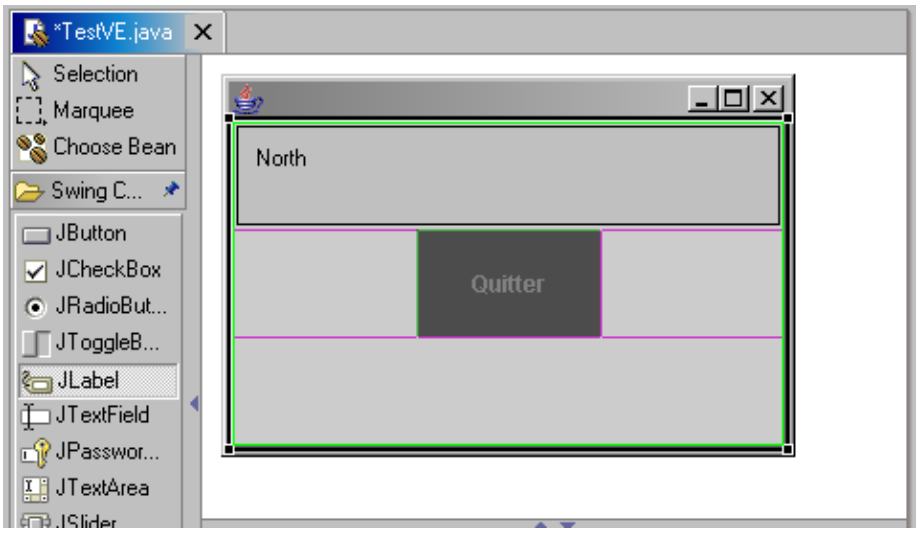
20.3.3. L'ajout d'éléments

Pour ajouter un nouvel élément dans le composant en cours de développement, il suffit de cliquer sur ce composant dans le menu de gauche puis de cliquer à sa position sur le composant.

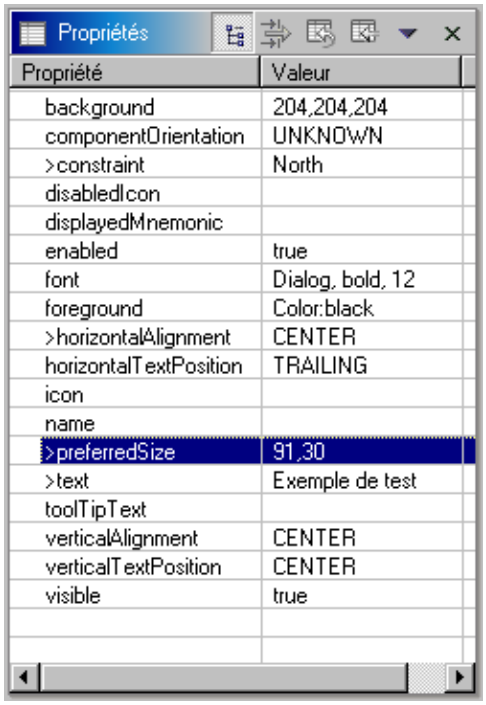


La localisation du nouvel élément est facilitée par l'affichage des zones du gestionnaire de positionnement du composant survolé par la souris (dans l'exemple ci dessus un BorderLayout).

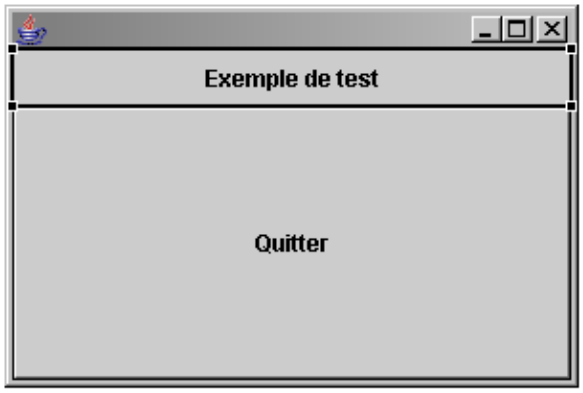
L'ajout d'un autre composant est tout aussi facile : par exemple, ajouter un composant JLabel dans la partie North du gestionnaire de positionnement.



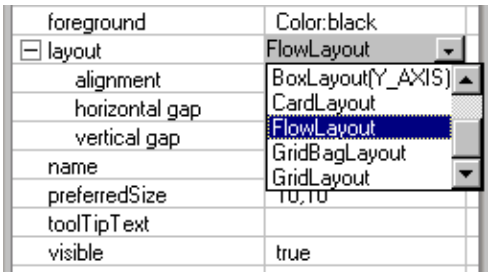
Les propriétés peuvent aussi être mises à jour, par exemple :



pour donner le résultat suivant :

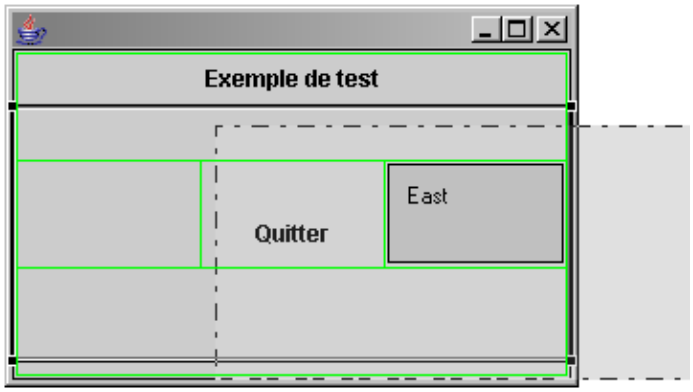


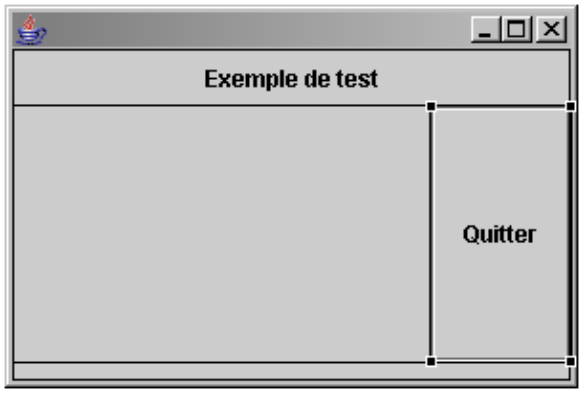
L'ajout d'un conteneur ce fait de la même façon que celle d'un composant. Dans les propriétés, il est possible de modifier le gestionnaire de positionnement défini par défaut pour ce conteneur en modifiant sa propriété layout.



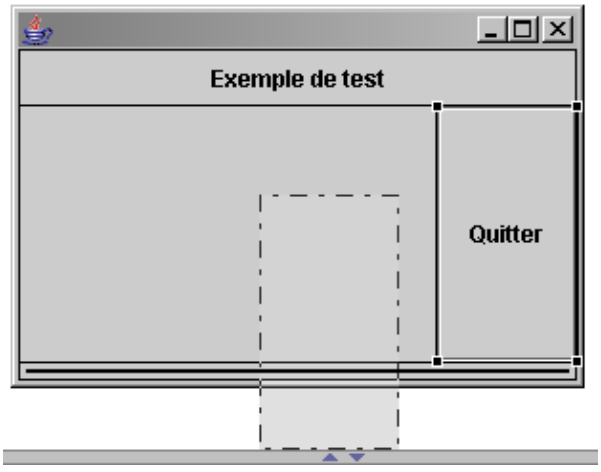
Il est possible de déplacer un composant d'un conteneur vers un autre. Il suffit de faire un cliquer/glisser du composant vers le nouveau conteneur.

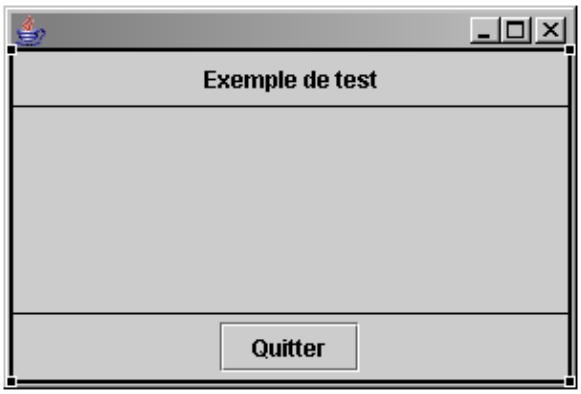
Exemple : déplacement du bouton dans la partie East du gestionnaire de positionnement BorderLayout





Exemple : déplacement du bouton dans le panneau inséré dans la partie South





La mise en forme des composants est très intuitive et très facile d'emploi notamment grâce à un affichage particulièrement pratique des gestionnaires de positionnement de chacun des conteneurs.

20.3.4. La gestion des événements

Pour ajouter la gestion d'un événement particulier d'un composant, il faut utiliser l'option « Events » du menu contextuel de ce composant.

Par exemple, pour ajouter un événement sur le clic du bouton, il faut :

- Cliquer sur le bouton pour le sélectionner
- Utiliser l'option « Events » de son menu contextuel
- Et sélectionner l'option « Action performed »

Le morceau de code correspondant est automatiquement généré.

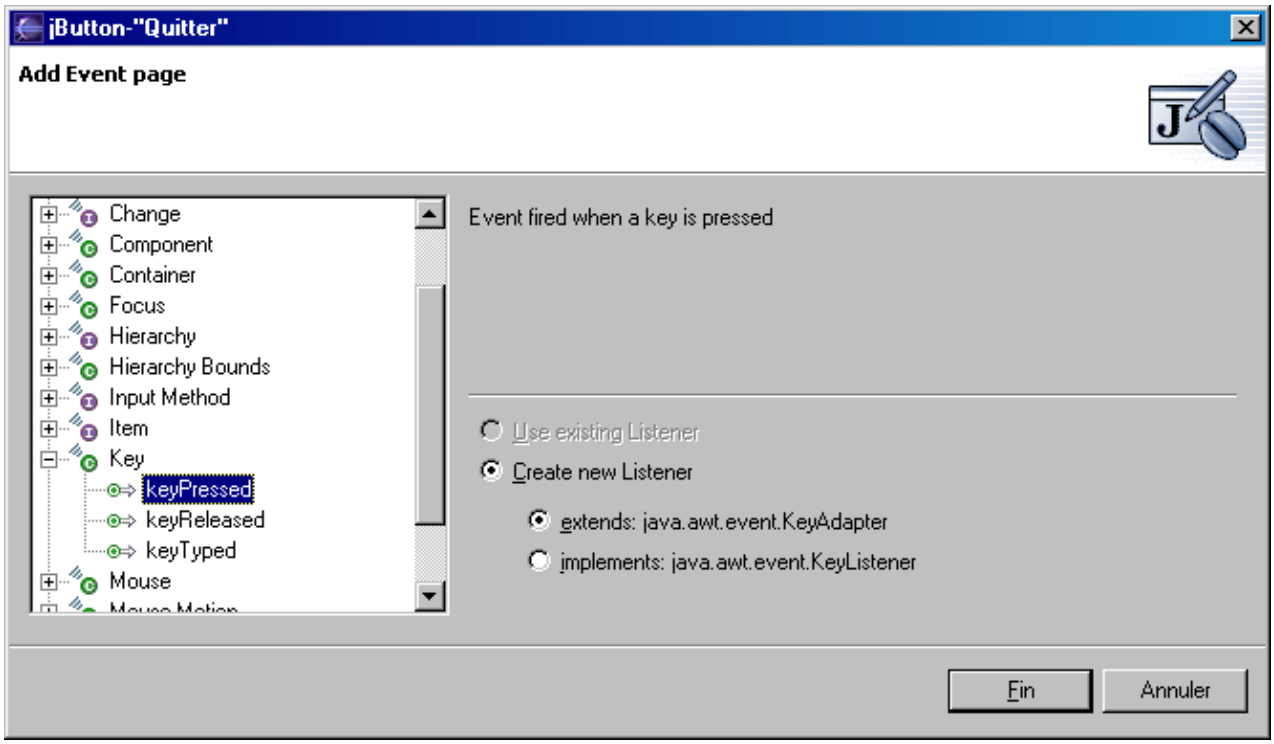
```
public void actionPerformed(java.awt.event.ActionEvent e) {
    System.out.println("actionPerformed()"); // TODO Auto-generated Eve
}
```

Une tâche de type « A faire » (TODO) est créée pour rappeler au développeur que cette méthode a été générée automatiquement et qu'il faut la compléter avec le code adéquat.

Il est par exemple possible de remplacer l'affichage du nom de la méthode sur la sortie standard par une demande de l'arrêt de l'application.

```
    public void actionPerformed(java.awt.event.ActionEvent e) {
        System.exit(0);
    }
```

Pour pouvoir définir un gestionnaire pour un des autres gestionnaires d'événements, il suffit d'utiliser l'option « Events / Add Events » du menu contextuel. Une boite de dialogue permet alors de sélectionner l'événement désiré.

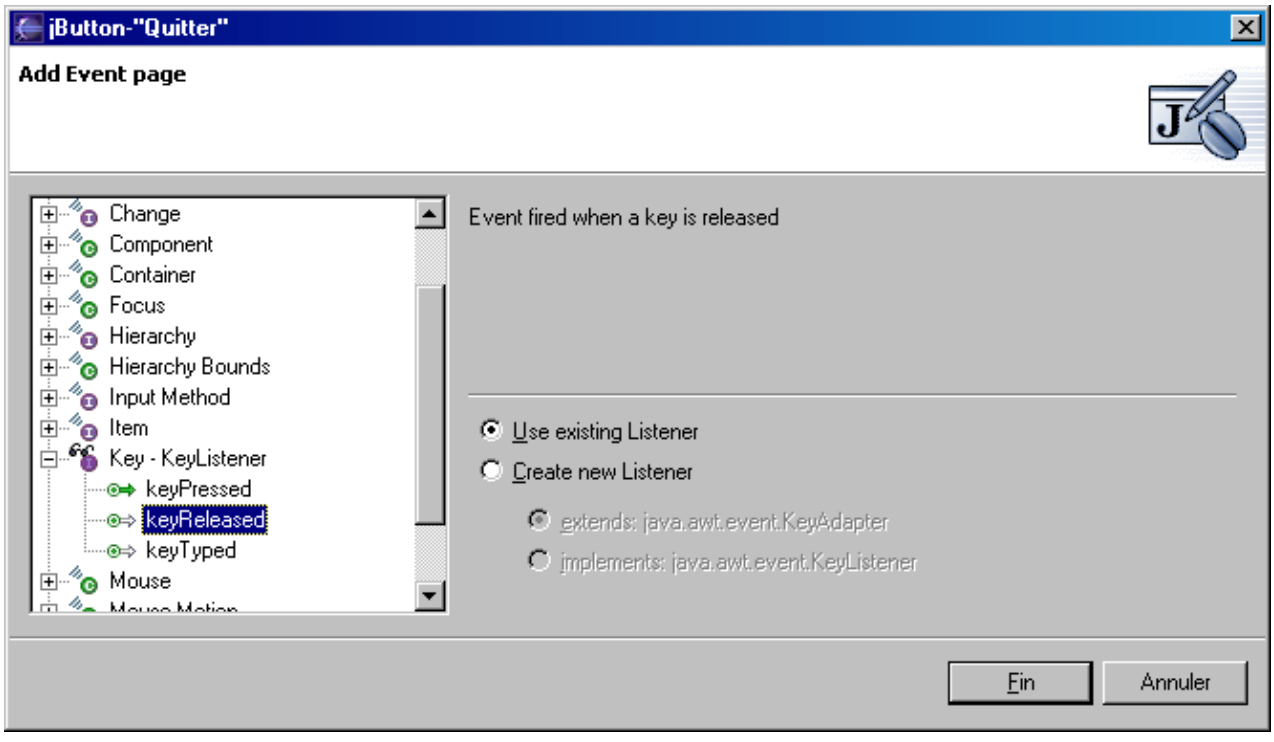


Il suffit alors de sélectionner :

- soit un type d'événement (par exemple Key) : dans ce cas les trois méthodes correspondant aux trois événements du listener correspondant seront générées
- soit directement un événement : dans ce cas, il est possible de définir quel sera l'origine du gestionnaire d'événement (Listener ou Adapter) selon que l'on veuille ou non une définition par défaut des méthodes à implémenter.

Un fois qu'un gestionnaire est défini, une petite icône en forme de lunette apparaît en haut à gauche du gestionnaire.

Si le gestionnaire d'événement est déjà partiellement implémenté pour un événement particulier, alors lors de la sélection d'un autre événement, VE propose d'utiliser le gestionnaire existant.



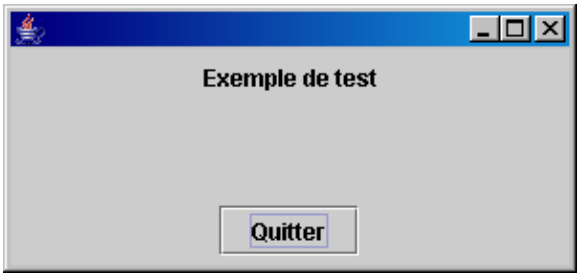
20.3.5. Exécution

Si l'application contient une méthode main(), il suffit de demander l'exécution de cette classe.

Dans l'exemple ci dessous, la classe ne possède pas de méthode main(). Il suffit d'écrire une classe qui va instancier un objet de type du composant créé et qui va l'afficher

Exemple :

```
public class TestVeRun {

  public static void main(String[] args) {

    TestVE tv = new TestVE();

    tv.show();

  }

}
```



Un clic sur le bouton « Quitter » ferme l'application.

Il est aussi possible de demander l'exécution sous le contrôle du débogueur comme pour une toute autre application.

20.4. Le plug−in W4Eclipse

http://w4toolkit.com/index.php

Cette section sera développée dans une version future de ce document

20.5. SWT designer

http://www.swt−designer.com/

SWT designer existe en deux versions : une libre nommée "free edition" et une commerciale nommée « professional ».

Cette section sera développée dans une version future de ce document

21. Eclipse WebTools Platform

Le projet Eclipse Web Tools Platform (WTP) a pour but d'enrichir la plate−forme Eclipse avec des outils pour développer des applications web J2EE. Il inclut :

- Des éditeurs pour les langages de pages web (HTML, CSS, Javascript)
- Des éditeurs pour des documents XML et associés (XML, DTD, XSD et WSDL)
- Le support de projet J2EE via des assistants
- Le support des services web via des assistants
- Le support des bases de données via SQL

Le projet WTP est composé de deux sous projets :

- Web Standard Tools (WST)
  Ce sous projet a pour but de développer un socle pour le développement d'applications web sous Eclipse.
- J2EE Standard Tools (JST)
  Ce sous projet a pour but de développer des plug−ins pour faciliter le développement d'applications respectant la norme J2EE 1.4.

Le site officiel du projet est à l'url http://www.eclipse.org/webtools/

La version mise en oeuvre dans ce chapitre est la version 1.0M3 : ce n'est donc pas encore une version finalisée mais elle propose déjà de nombreuses fonctionnalités.

Le plug−in supporte plusieurs conteneurs et serveurs d'applications : Tomcat (version 3.2 à 5.5), JBoss 3.2.3, Jonas 4.1.4 et Weblogic 8.1.

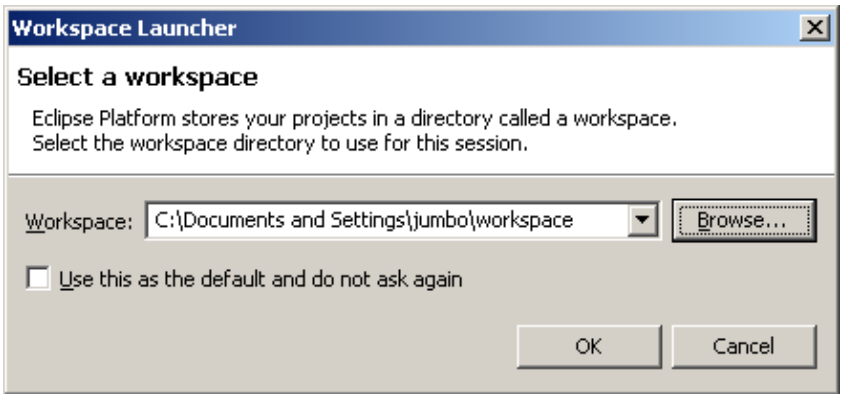
21.1. Installation

Il faut télécharger le fichier wtp−eclipse−prereqs−sdk−1.0M3−win32.zip et le décompresser dans un répertoire du système en suivant les liens sur la page http://www.eclipse.org/webtools/index.html

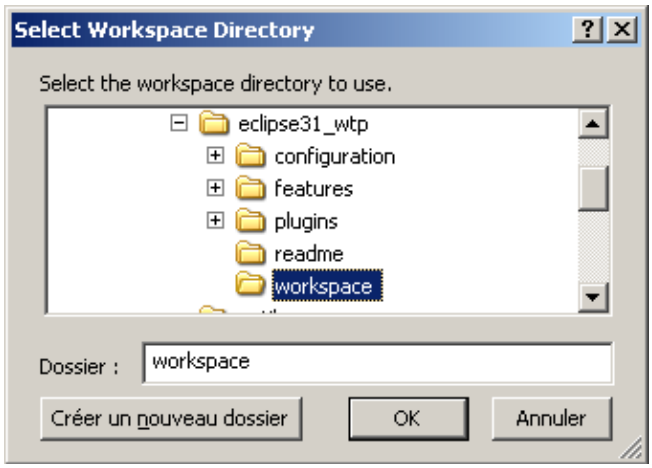
Attention, ce fichier possède une taille de 151Mo.

Le répertoire Eclipse résultant de la décompression est renommé en eclipse31_wtp dans les exemples de ce chapitre et un sous répertoire workspace y est créé pour stocker les differents projets.

Lancez Eclipse à partir du répertoire décompressé.

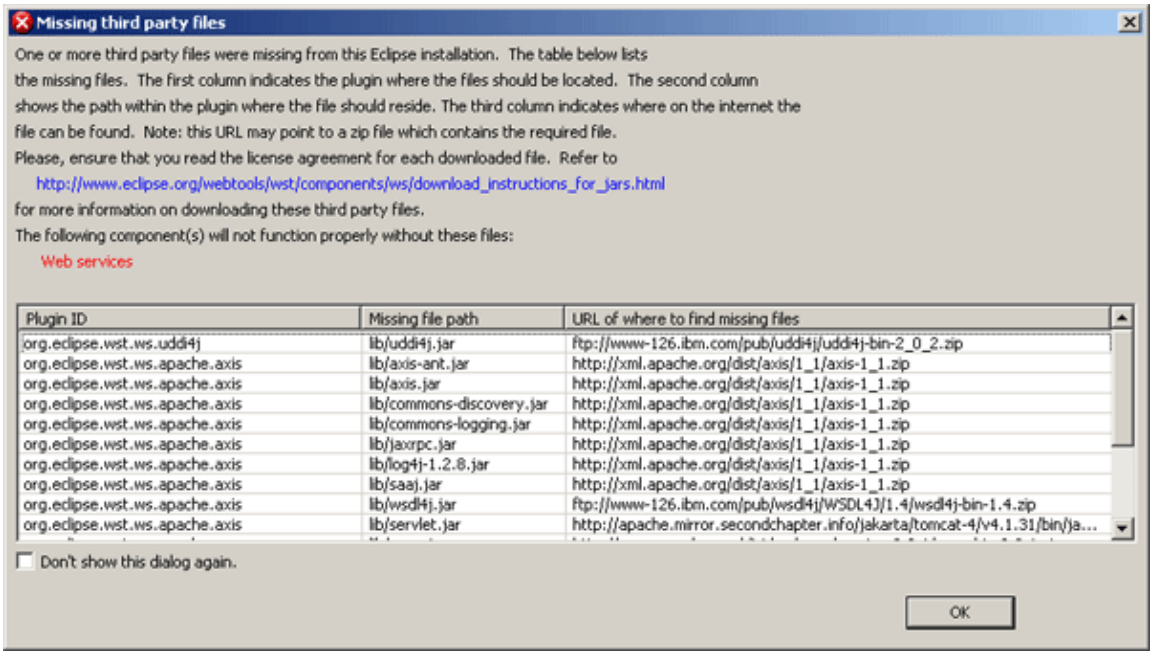


Cliquez sur le bouton « Browse »



Sélectionnez le répertoire workspace créé et cliquez sur le bouton « OK », puis une nouvelle fois sur le bouton « OK » pour valider le workspace sélectionné.

Au lancement d'Eclipse, un message informe du manque de certains plug−ins qui empêche l'utilisation des fonctionnalités concernant les « Web services »



Cliquez sur le bouton « OK »

Il est donc nécessaire de télécharger les plug−ins manquants qui ne sont pas fournis dans le bundle.

Créez un répertoire temp dans le répertoire eclipse31_wtp et téléchargez dedans les fichiers dont l'url est indiquée dans le message.

Fichier	Taille
axis−1_1.zip	10.2 Mo
jakarta−tomcat−4.1.31.zip	9.7 Mo
soap−bin−2.3.1.zip	1,6 Mo
uddi4j−bin−2_0_2.zip	1 Mo
wsil4j.jar	78 Ko
jaf−1_0_2−upd.zip	349 Ko
javamail−1_3_2.zip	2.3 Mo
wsdl4j−bin−1.4.zip	891 Ko

Il faut décompresser ces fichiers dans leur répertoire approprié dans le sous répertoire plugins. Pour cela, le plus simple est d'utiliser un script Ant fourni par le projet (Attention, Ant doit être installé et configuré sur le poste pour pouvoir utiliser ce script).

Il faut télécharger le fichier à l'url
http://dev.eclipse.org/viewcvs/index.cgi/~checkout~/org.eclipse.wtp.releng/fetchVendorContent.xml?content−type=text/xml&cvsroot=WebTools_Project,
l'enregistrer dans le répertoire temp et l'exécuter en lui fournissant les paramètres indiqué dans l'exemple ci−dessous :

Exemple :

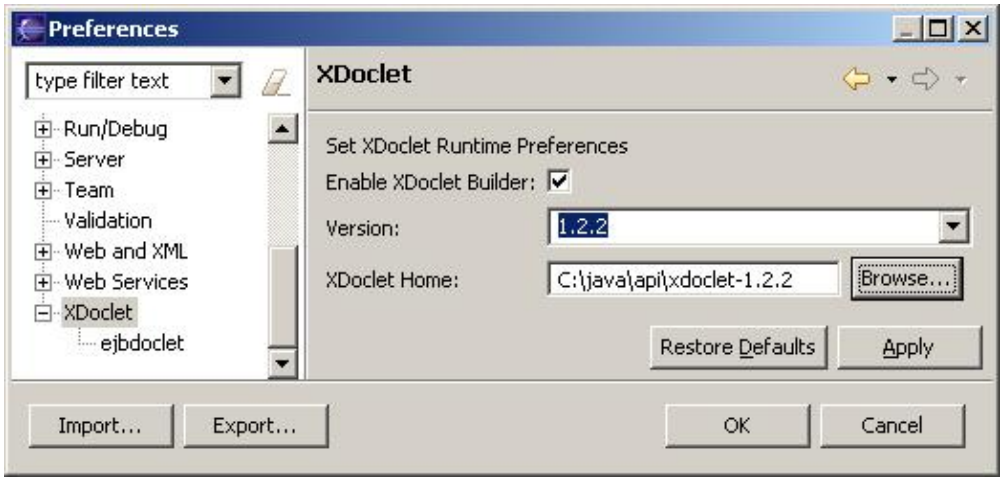
```
C:\java\eclipse31_wtp\temp>ant −DlocalDownloads=C:\java\eclipse31_wtp\temp −Dbui
ldDirectory=C:\java\eclipse31_wtp\plugins −f fetchVendorContent.xml
Buildfile: fetchVendorContent.xml
run:
extractAllBinary:
extractAll:
    [unzip] Expanding: C:\java\eclipse31_wtp\temp\axis−1_1.zip into C:\java\ecli
pse31_wtp\temp
    [unzip] Expanding: C:\java\eclipse31_wtp\temp\jakarta−tomcat−4.1.31.zip into
 C:\java\eclipse31_wtp\temp
    [unzip] Expanding: C:\java\eclipse31_wtp\temp\jaf−1_0_2−upd.zip into C:\java
\eclipse31_wtp\temp
    [unzip] Expanding: C:\java\eclipse31_wtp\temp\javamail−1_3_2.zip into C:\jav
a\eclipse31_wtp\temp
    [unzip] Expanding: C:\java\eclipse31_wtp\temp\soap−bin−2.3.1.zip into C:\jav
a\eclipse31_wtp\temp
    [unzip] Expanding: C:\java\eclipse31_wtp\temp\uddi4j−bin−2_0_2.zip into C:\j
ava\eclipse31_wtp\temp
    [unzip] Expanding: C:\java\eclipse31_wtp\temp\wsdl4j−bin−1.4.zip into C:\jav
a\eclipse31_wtp\temp
     [move] Moving 8 files to C:\java\eclipse31_wtp\temp\org.eclipse.wst.ws.apac
he.axis_1.0.0
     [move] Moving 1 files to C:\java\eclipse31_wtp\temp\org.eclipse.wst.ws.apac
he.axis_1.0.0
     [move] Moving 1 files to C:\java\eclipse31_wtp\temp\org.eclipse.wst.ws.apac
he.soap_1.0.0
     [move] Moving 1 files to C:\java\eclipse31_wtp\temp\org.eclipse.wst.ws.apac
he.soap_1.0.0
     [move] Moving 1 files to C:\java\eclipse31_wtp\temp\org.eclipse.wst.ws.apac
he.soap_1.0.0
     [copy] Copying 1 file to C:\java\eclipse31_wtp\temp\org.eclipse.wst.ws.apac
he.wsil_1.0.0
     [move] Moving 1 files to C:\java\eclipse31_wtp\temp\org.eclipse.wst.ws.uddi
```

```
4j_1.0.0
grabjarsxml:
copyJars:
copyJars2:
copyJars3:
     [copy] Copying 9 files to C:\java\eclipse31_wtp\plugins\org.eclipse.wst.ws.
apache.axis_1.0.0\lib
copyJars:
copyJars2:
copyJars3:
     [copy] Copying 3 files to C:\java\eclipse31_wtp\plugins\org.eclipse.wst.ws.
apache.soap_1.0.0\lib
copyJars:
copyJars2:
copyJars3:
     [copy] Copying 1 file to C:\java\eclipse31_wtp\plugins\org.eclipse.wst.ws.a
pache.wsil_1.0.0\lib
copyJars:
copyJars2:
copyJars3:
     [copy] Copying 1 file to C:\java\eclipse31_wtp\plugins\org.eclipse.wst.ws.u
ddi4j_1.0.0\lib
     [copy] Copying 2 files to C:\java\eclipse31_wtp\plugins\org.eclipse.wst.wsd
l_1.0.0\lib
extractAllSDK:
BUILD SUCCESSFUL
Total time: 17 seconds
C:\java\eclipse31_wtp\temp>cd ..
C:\java\eclipse31_wtp>eclipse -clean
```

Il faut relancer Eclipser avec l'option –clean dans une boîte de commande Dos

21.2. Configuration de XDoclet

XDoclet doit être installé sur le système. Afin de permettre une utilisation de XDoclet par le plug−in, il faut le configuration dans les préférences.



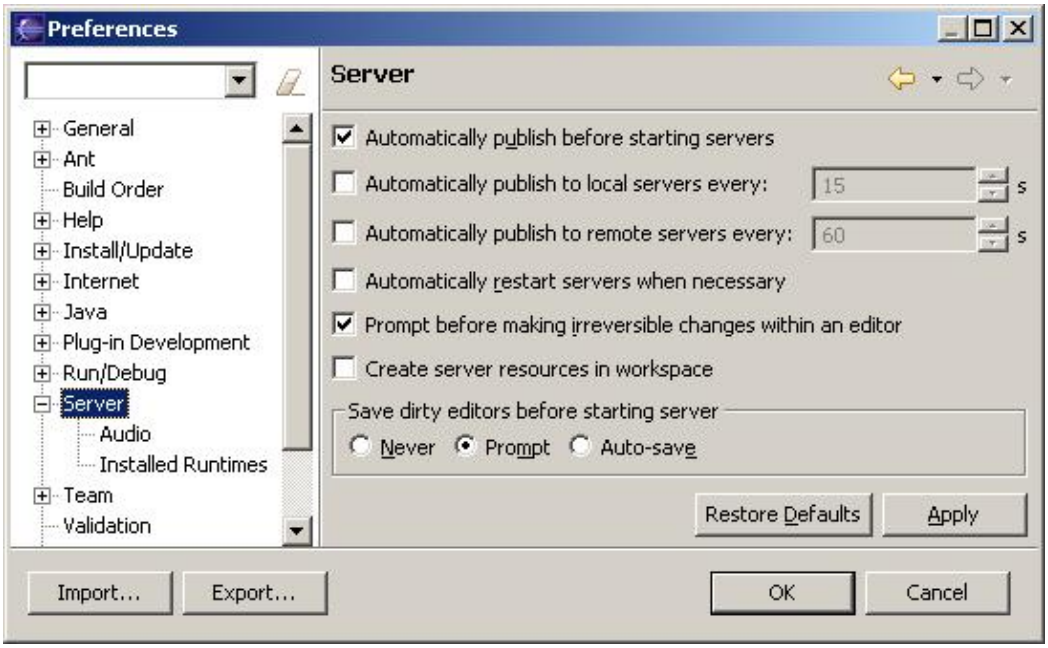
Il faut absolument cocher la case « Enable XDoclet Builder », sélectionner la version à utiliser et sélectionner le répertoire qui contient XDoclet sur le système.

21.3. Configuration du serveur

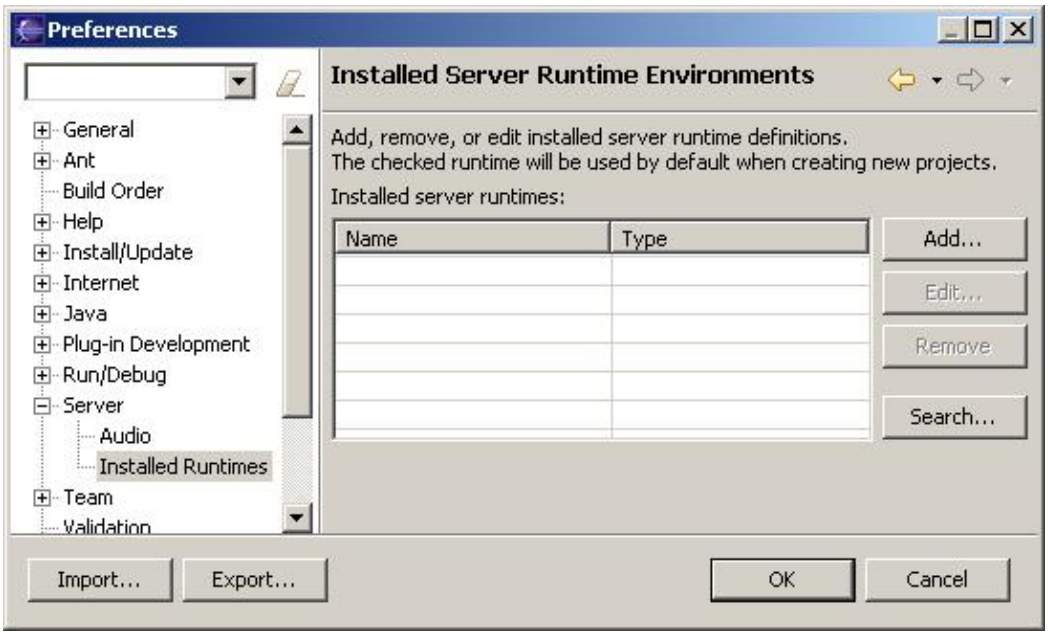
WTP supporte plusieurs conteneurs et serveurs d'applications : Tomcat, JBoss, Jonas, ...

Il est nécessaire de configurer un ou plusieurs serveurs dans les préférences du WTP.

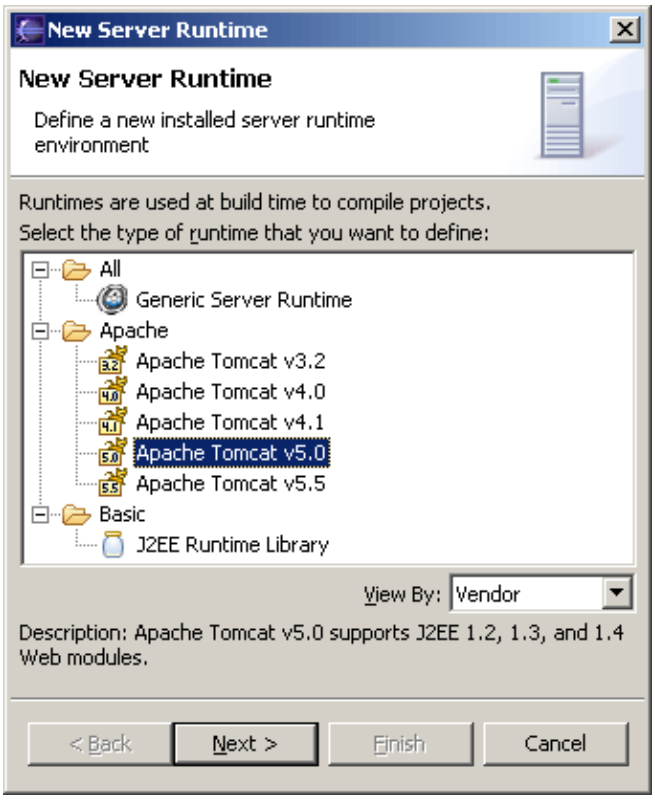
La page principale du noeud Server des préférences permet de régler des paramètres généraux concernant les serveurs



La page « Installed Runtimes » permet de configurer un ou plusieurs serveurs.



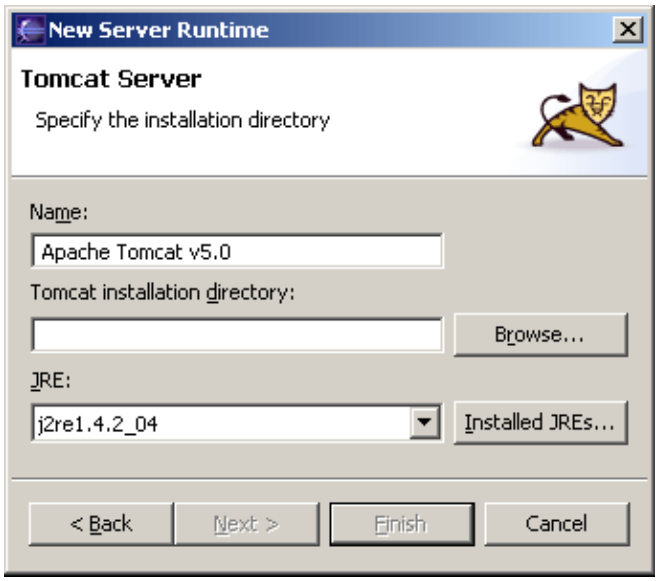
Pour ajouter un nouveau serveur, il faut cliquer sur le bouton « Add ».



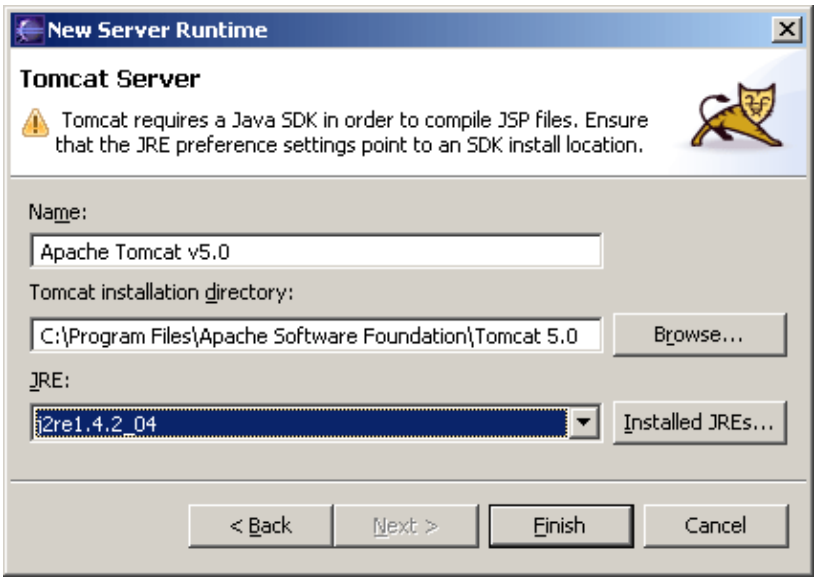
Il existe trois grands types de serveurs :

- Apache : permet de sélectionner le conteur Tomcat ainsi que sa version dans lequel il est possible de modifier dynamiquement des éléments de l'application sans redéployer toute l'application packagée
- All/Generic Server runtime : permet de sélectionner JBoss, Jonas ou Weblogic dans lesquels l'application doit être déployée sous la forme packagée avant d'être exécutée
- Basic/J2EE Runtime Library ne désigne pas un conteneur ou un serveur d'application mais simplement un ensemble de bibliothèques

Pour Tomcat 5.0 par exemple, il faut sélectionner « Apache/Apache Tomcat v5.0 » et cliquer sur le bouton « Next ».

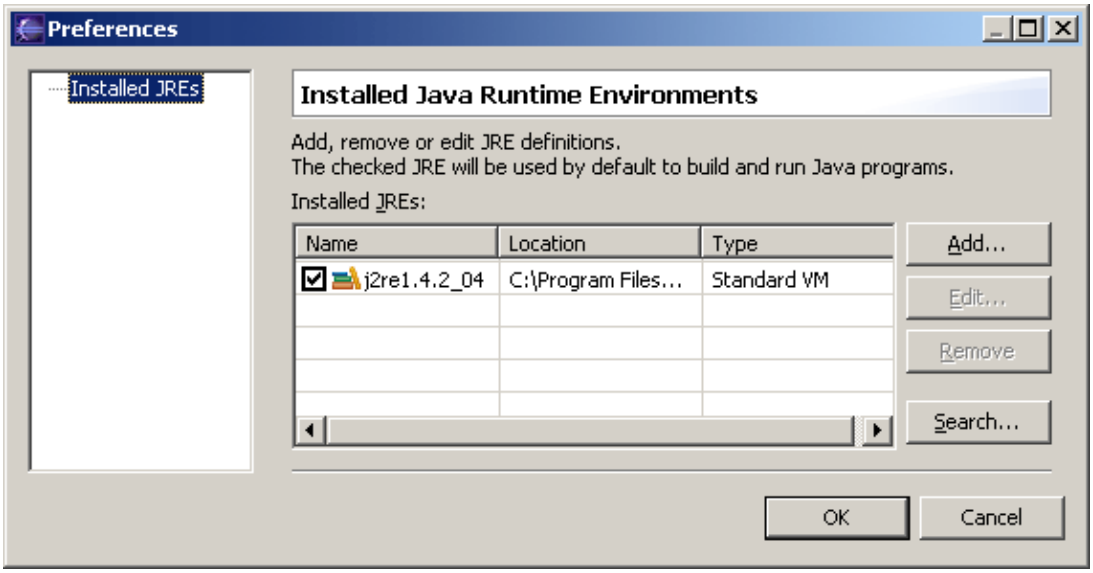


Cliquez sur le bouton « Browse » et sélectionnez le répertoire qui contient Tomcat

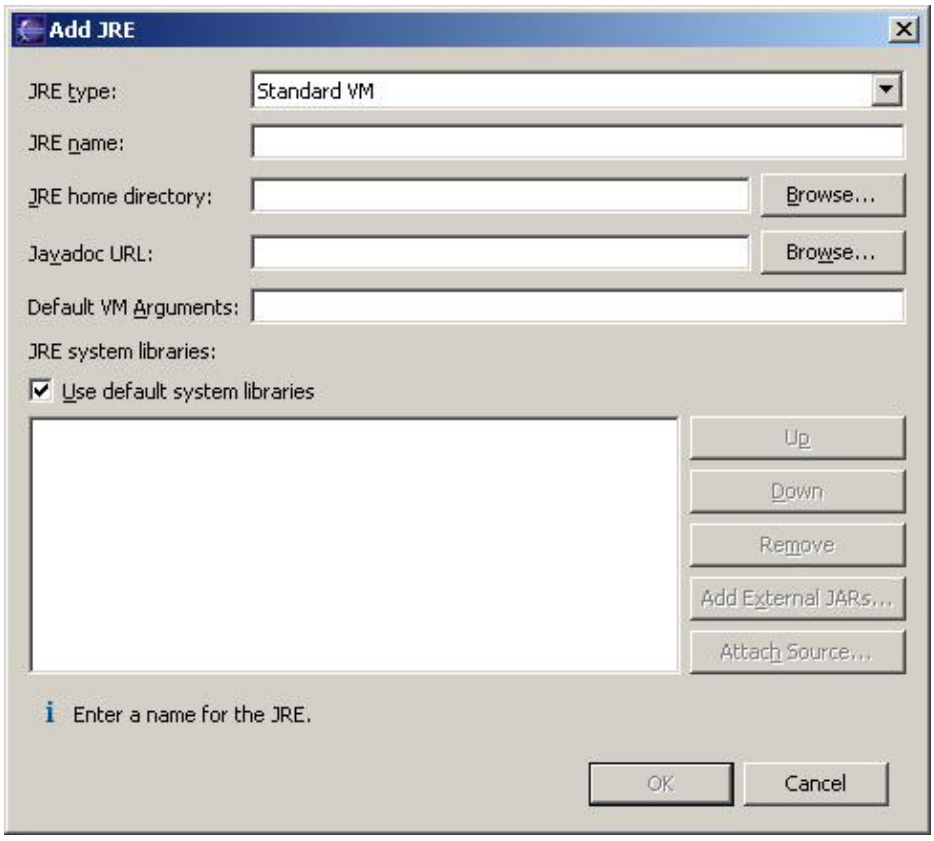


Pour utiliser Tomcat, un JDK est requis notamment pour permettre la compilation des JSP.

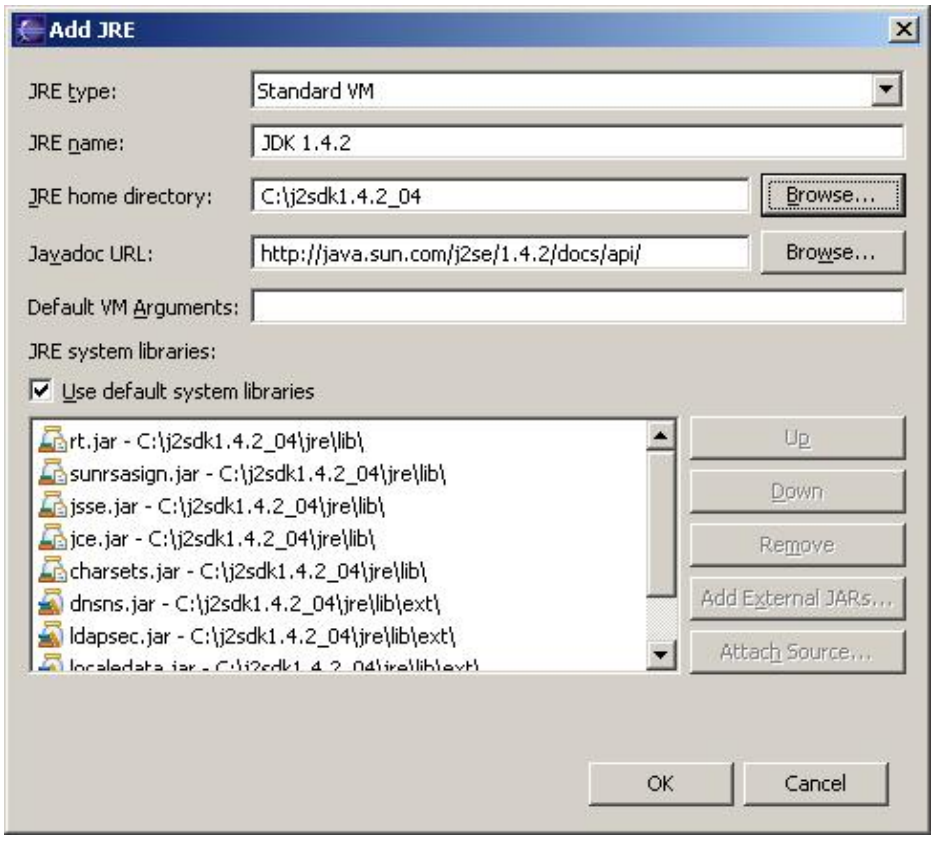
Si aucun JDK n'est configuré, il faut cliquer sur le bouton « Installed JREs »



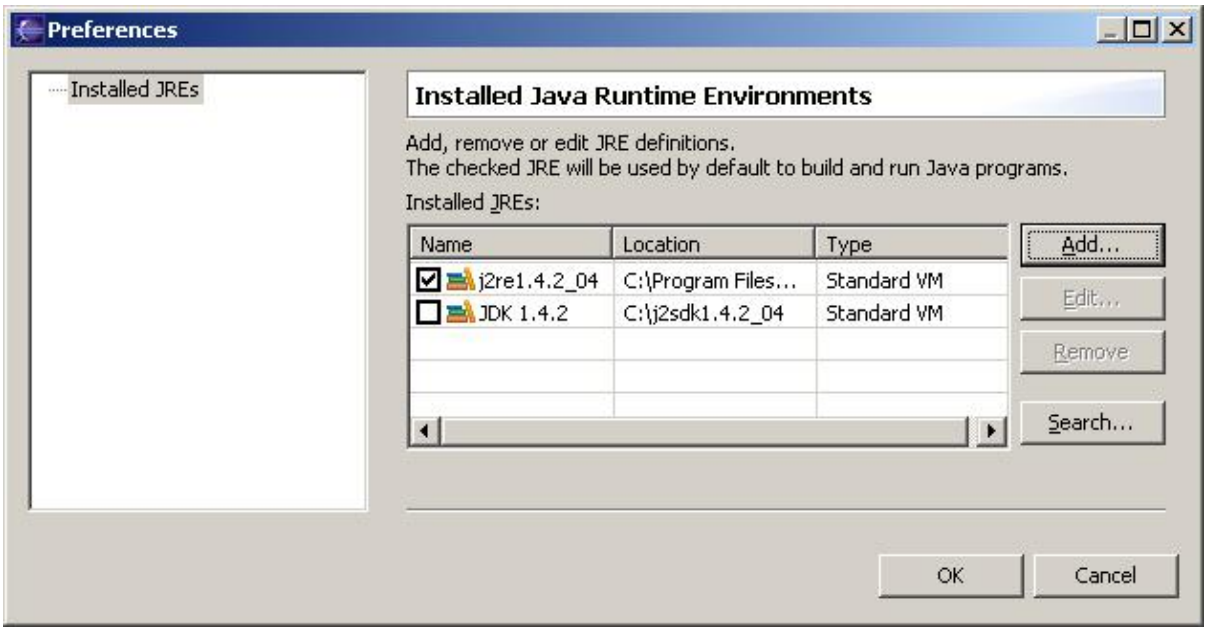
Cliquez sur le bouton « Add »



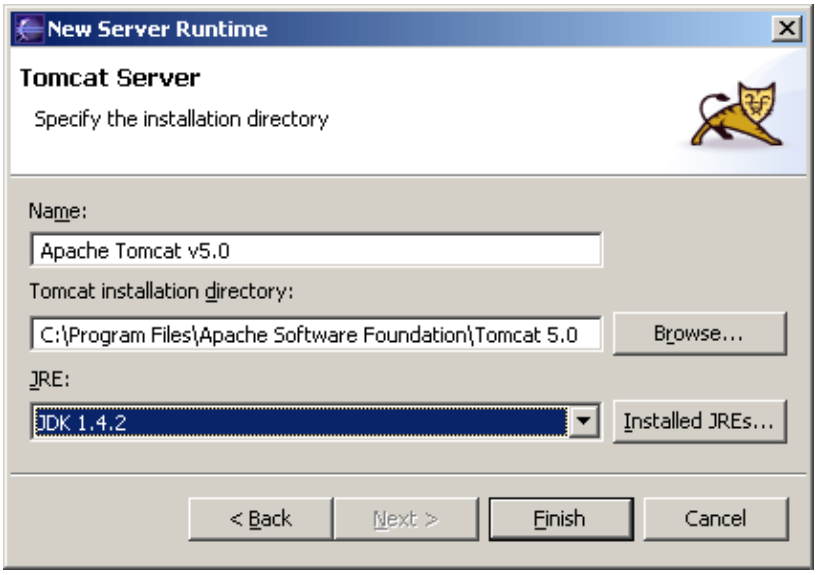
Saisissez le nom du JRE et sélectionnez le répertoire. Eclipse renseigne automatiquement les autres éléments requis mais il est possible de les modifier.



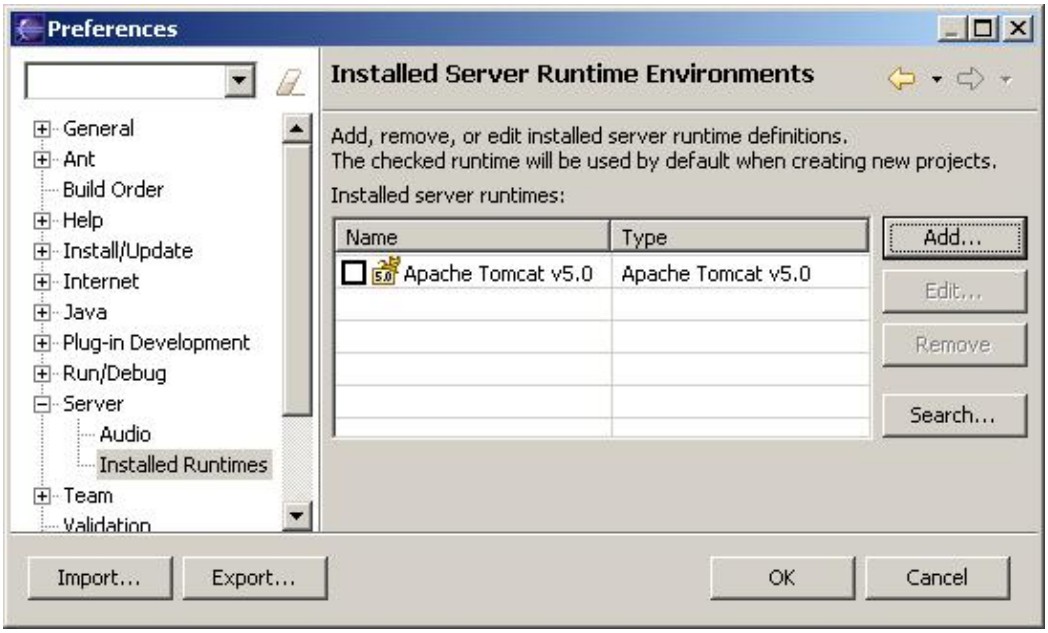
Cliquez sur le bouton « OK » pour ajouter le nouveau JRE.



Cliquez sur le bouton « OK »

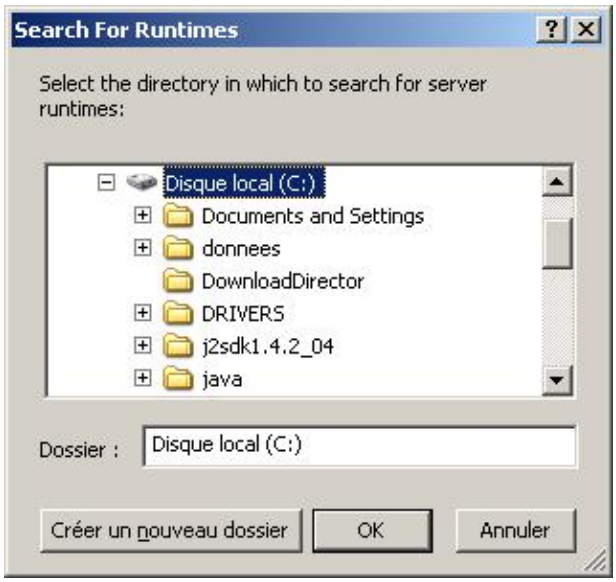


Il suffit alors de sélectionner le nouveau JRE défini et de cliquer sur le bouton « Finish ».

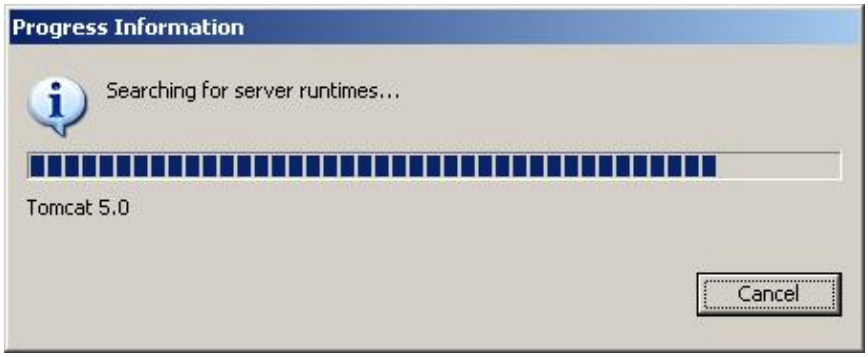


Le nouveau serveur est ajouté. Il est possible de le définir comme serveur par défaut en cochant sa case à cocher et en cliquant sur le bouton « OK ».

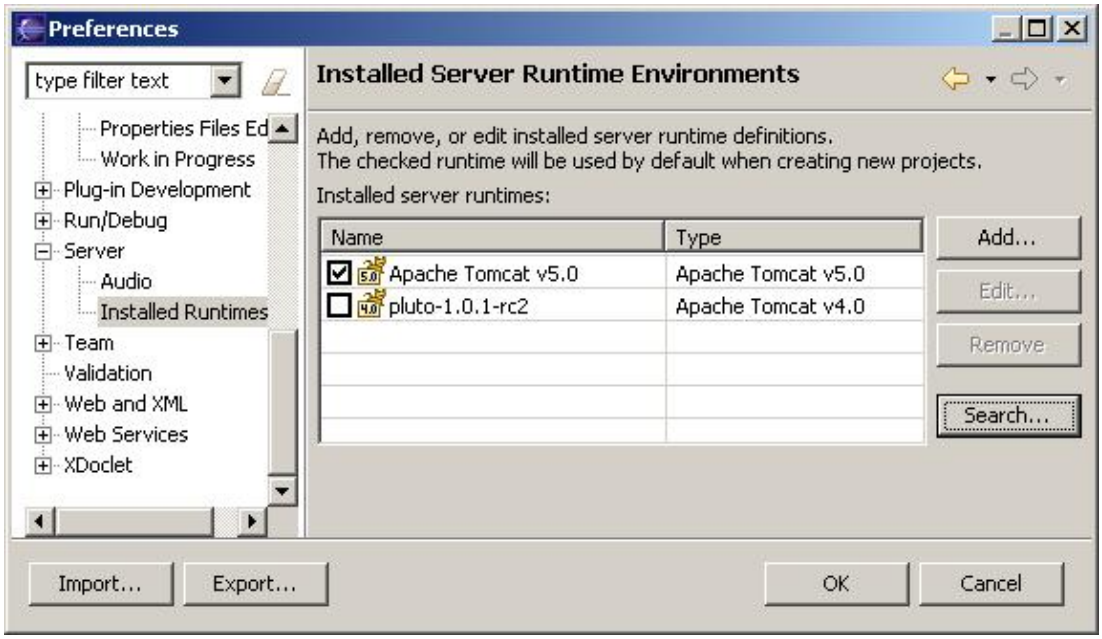
Cliquez sur le bouton « Search » pour demander la recherche des serveurs connus par le plug−in et installés sur la machine.



Sélectionnez le répertoire de départ de la recherche et cliquez sur le bouton « OK »



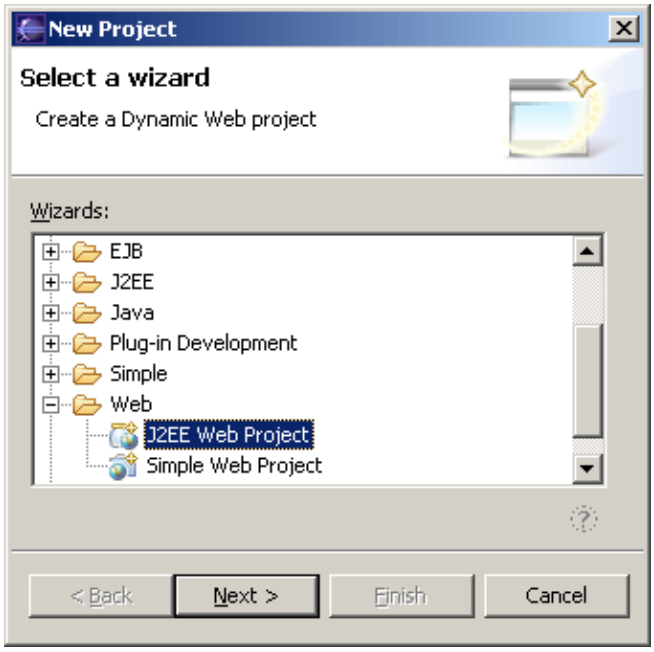
Les différents serveurs trouvés sont automatiquement ajoutés



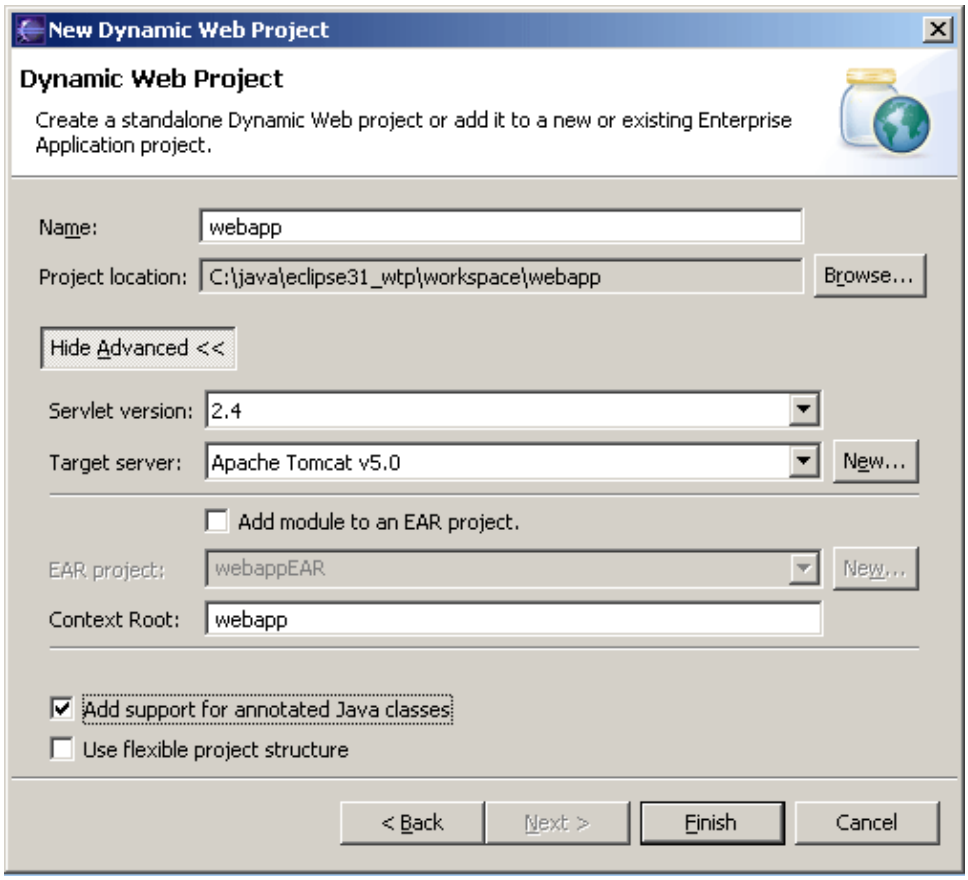
21.4. Le développement d'applications web

21.4.1. La création d'un nouveau projet

Pour développer une application web, il est nécessaire de créer un projet de type « Web/J2EE Web Project ». Ce type de projet est un projet Java qui va contenir une application serveur de type web.



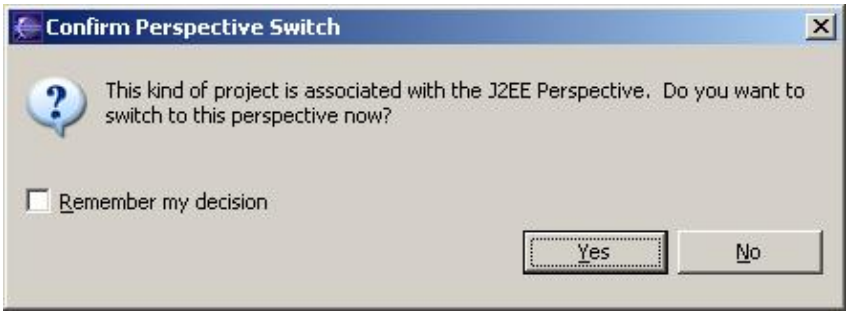
Cliquez sur le bouton « Next »



Il faut saisir le nom du projet et le contexte de l'application. Il est possible de sélectionner la version de l'API servlet à utiliser dans le projet et modifier le serveur cible si plusieurs sont définis.

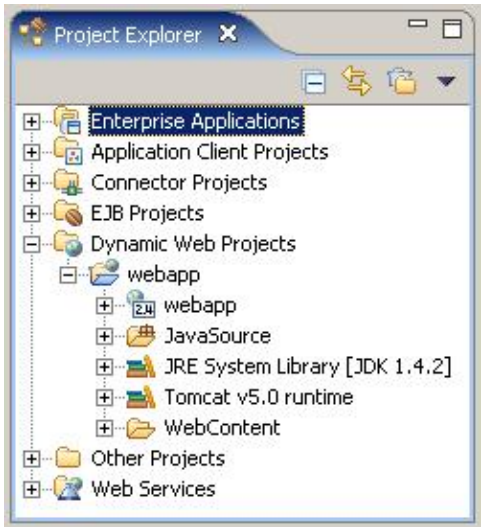
Si l'application ne doit pas être intégrée dans une archive de type EAR, il faut décocher la case demandant l'ajout de la webapp dans une telle archive.

Cliquez sur le bouton « Finish ».

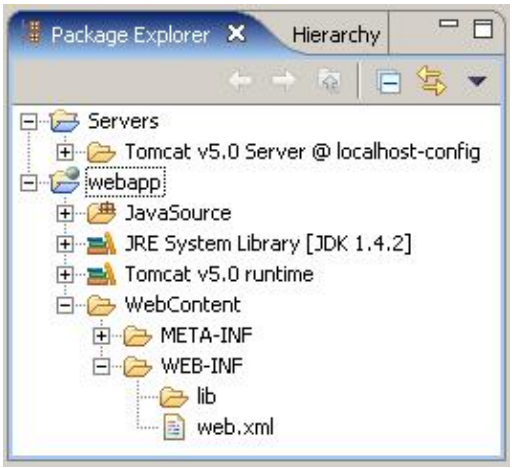


Le plus simple est de cocher « Remember my decision » et cliquez sur le bouton « Yes » pour ouvrir la perspective J2EE.

Le nouveau projet est créé dans l'arborescence « Dynamic Web Projects » de la vue « Project explorer ». Cette vue permet d'avoir une présentation des projets selon les grandes familles de projets.

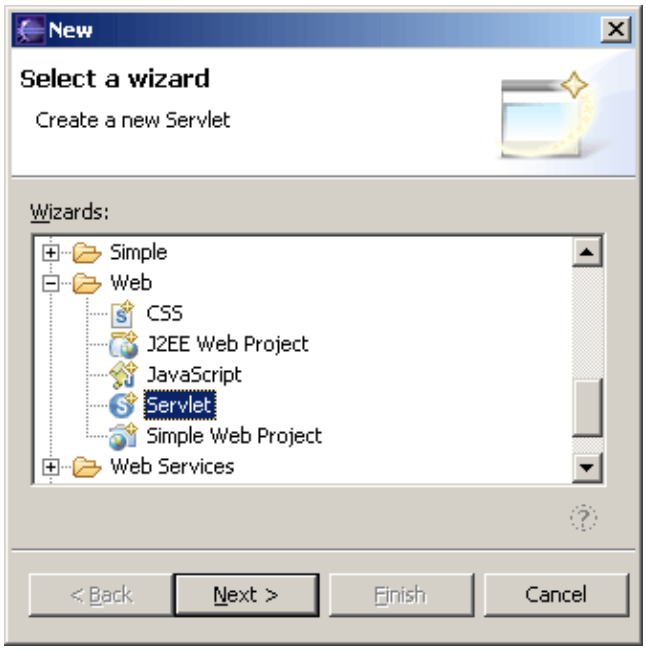


La vue « Package explorer » permet d'avoir une vue plus concise du projet.

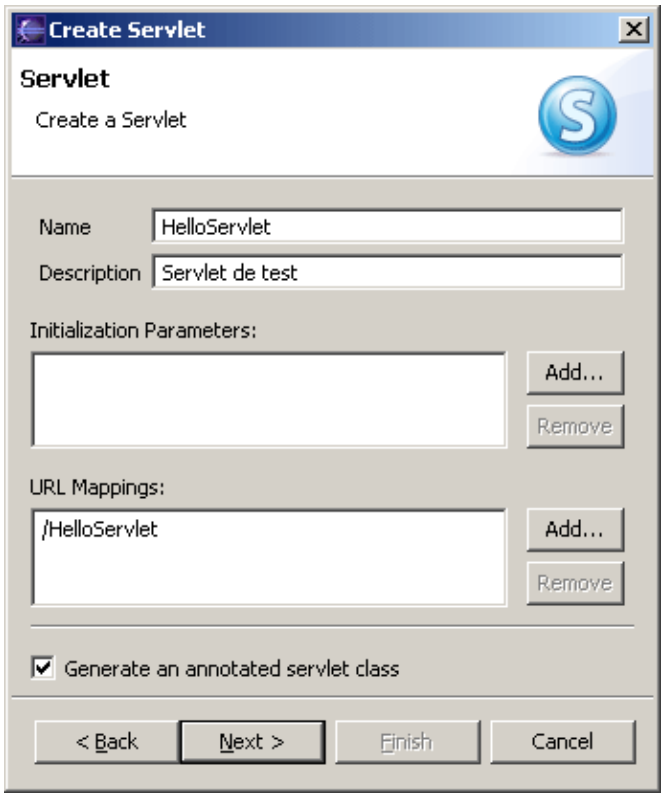


21.4.2. La création d'une servlet

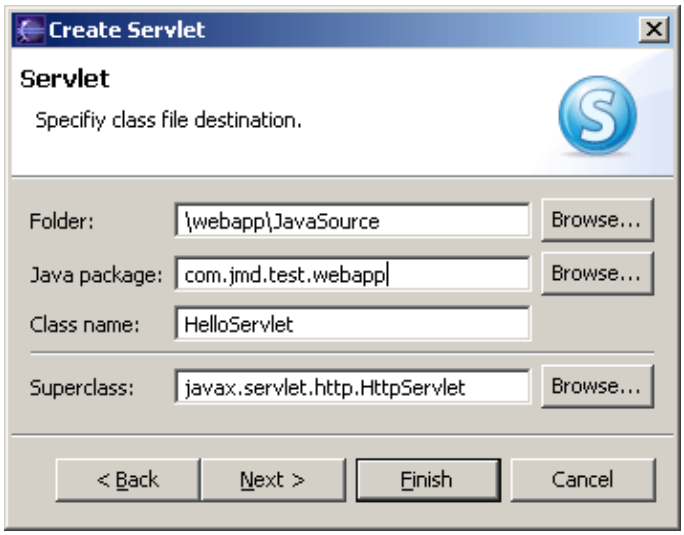
Pour ajouter une servlet au projet, il faut créer une nouvelle entité du type « Web/Servlet »



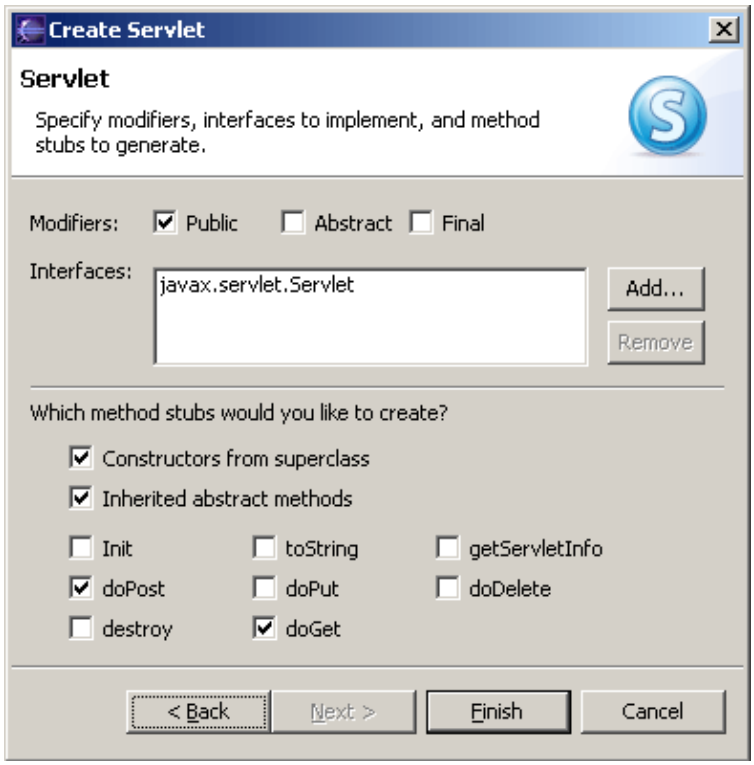
Cliquez sur le bouton « Next »



Il faut saisir le nom de la servlet et éventuellement sa description, ses paramètres et son url de mapping si celle proposée par défaut ne convien tpas puis cliquer sur le bouton « Next ».

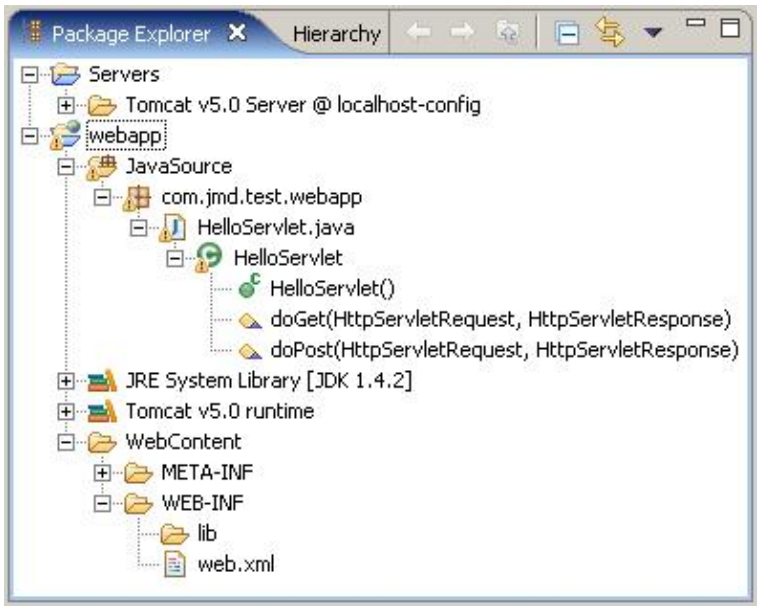


La page suivante de l'assistant permet de saisir les informations concernant la classe de la servlet : les valeurs par défaut peuvent être conservées. Il suffit simplement de saisir le package qui va contenir la servlet et de cliquer sur le bouton « Next ».



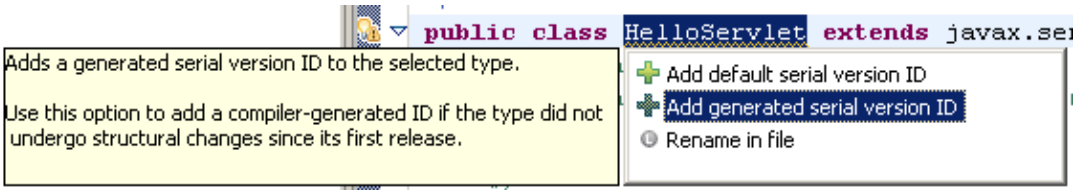
La page suivante de l'assistant permet de préciser les membres qui seront générés dans la servlet.

Pour générer la servlet, il suffit de cliquer sur le bouton « Finish ».

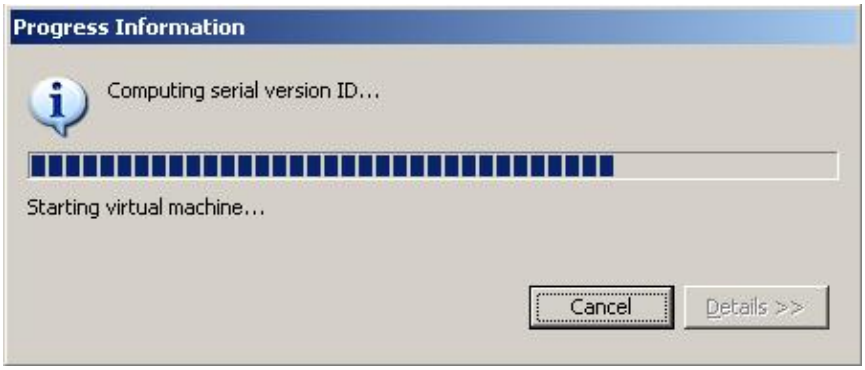


Il suffit alors de saisir le code de la servlet.

Pour éliminer l'avertissement, il suffit de demander la génération d'un identifiant en cliquant sur la petite ampoule jaune.



L'identifiant est généré et inséré dans le code de la servlet.



Il ne reste plus alors qu'à écrire le code des traitements de la servlet.

Exemple :

```
package com.jmd.test.webapp;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Servlet implementation class for Servlet: HelloServlet
 *
 * @web.servlet
 *   name="HelloServlet"
 *   display-name="HelloServlet"
```

```
 *   description="Servlet de test"
 *
 * @web.servlet-mapping
 *   url-pattern="/HelloServlet"
 *
 */
public class HelloServlet extends javax.servlet.http.HttpServlet
                          implements javax.servlet.Servlet {
            /**
             *
             */
            private static final long serialVersionUID = 3761131535167928369L;

            /* (non-Java-doc)
             * @see javax.servlet.http.HttpServlet#HttpServlet()
             */
            public HelloServlet() {
                        super();
            }

            /* (non-Java-doc)
             * @see javax.servlet.http.HttpServlet#doGet(HttpServletRequest arg0,
             *                                          HttpServletResponse arg1)
             */
            protected void doGet(HttpServletRequest arg0, HttpServletResponse arg1)
                              throws ServletException, IOException {
                        arg1.setContentType("text/html");
                        PrintWriter out = arg1.getWriter();
                        out.println("<HTML>");
                        out.println("<HEAD>");
                        out.println("<TITLE>Bonjour</TITLE>");
                        out.println("</HEAD>");
                        out.println("<BODY>");
                        out.println("<H1>Bonjour</H1>");
                        out.println("</BODY>");
                        out.println("</HTML>");
            }

            /* (non-Java-doc)
             * @see javax.servlet.http.HttpServlet#doPost(HttpServletRequest arg0,
             *                                           HttpServletResponse arg1)
             */
            protected void doPost(HttpServletRequest arg0, HttpServletResponse arg1)
                                  throws ServletException, IOException {
                        doGet(arg0, arg1);
            }
}
```

Lors de la sauvegarde d'une ressource de l'application ou lors de la recompilation, XDoclet est appelé et utilise les tags contenus dans le code pour notamment re-générer le fichier web.xml.

Exemple :

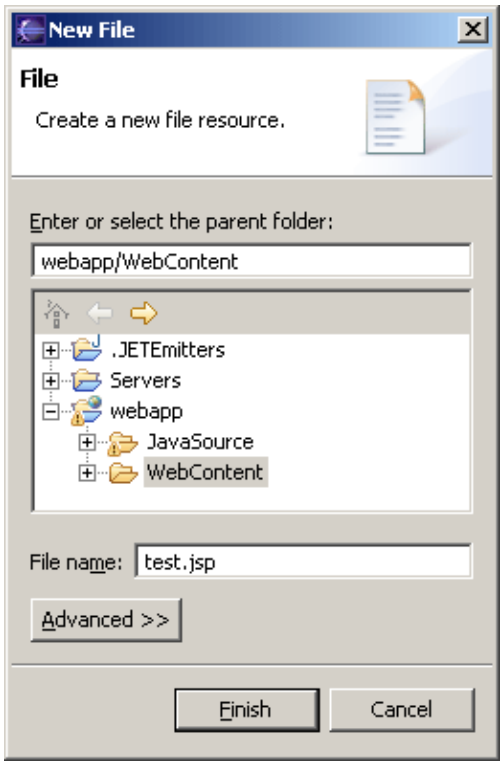
```
<?xml version="1.0" encoding="UTF-8"?>
<web-app  xmlns="http://java.sun.com/xml/ns/j2ee"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
       http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
   version="2.4">
   <distributable/>
    <!--
    To use non XDoclet filters, create a filters.xml file that
    contains the additional filters (eg Sitemesh) and place it in your
    project's merge dir.  Don't include filter-mappings in this file,
    include them in a file called filter-mappings.xml and put that in
    the same directory.
```

```
  -->
  <!--
  To use non XDoclet filter-mappings, create a filter-mappings.xml file that
  contains the additional filter-mappings and place it in your
  project's merge dir.
  -->
  <!--
  To use non XDoclet listeners, create a listeners.xml file that
  contains the additional listeners and place it in your
  project's merge dir.
  -->
 <servlet>
    <servlet-name>TestServlet</servlet-name>
    <display-name>TestServlet</display-name>
   <servlet-class>com.jmd.test.webapp.TestServlet</servlet-class>
 </servlet>
 <servlet>
    <servlet-name>HelloServlet</servlet-name>
    <display-name>HelloServlet</display-name>
    <description><![CDATA[Servlet de test]]></description>
   <servlet-class>com.jmd.test.webapp.HelloServlet</servlet-class>
 </servlet>
<!--
To use non XDoclet servlets, create a servlets.xml file that
contains the additional servlets (eg Struts) and place it in your
project's merge dir.  Don't include servlet-mappings in this file,
include them in a file called servlet-mappings.xml and put that in
the same directory.
-->
 <servlet-mapping>
    <servlet-name>TestServlet</servlet-name>
    <url-pattern>/TestServlet</url-pattern>
 </servlet-mapping>
 <servlet-mapping>
    <servlet-name>HelloServlet</servlet-name>
    <url-pattern>/HelloServlet</url-pattern>
 </servlet-mapping>
 <!--
 To specify mime mappings, create a file named mime-mappings.xml,
 put it in your project's mergedir.
 Organize mime-mappings.xml following this DTD slice:
 <!ELEMENT mime-mapping (extension, mime-type)>
 -->
 <!--
 To specify error pages, create a file named error-pages.xml,
 put it in your project's mergedir.
 Organize error-pages.xml following this DTD slice:
 <!ELEMENT error-page ((error-code | exception-type), location)>
 -->
<!--
To add taglibs by xml, create a file called taglibs.xml and place it
in your merge dir.
-->
 <!--
 To set up security settings for your web app, create a file named web-security.xml,
 put it in your project's mergedir.
 Organize web-security.xml following this DTD slice:
 <!ELEMENT security-constraint (display-name?, web-resource-collection+,
     auth-constraint?, user-data-constraint?)>
 <!ELEMENT web-resource-collection (web-resource-name, description?,
     url-pattern*, http-method*)>
 <!ELEMENT web-resource-name (#PCDATA)>
 <!ELEMENT url-pattern (#PCDATA)>
 <!ELEMENT http-method (#PCDATA)>
 <!ELEMENT user-data-constraint (description?, transport-guarantee)>
 <!ELEMENT transport-guarantee (#PCDATA)>
 <!ELEMENT login-config (auth-method?, realm-name?, form-login-config?)>
 <!ELEMENT auth-method (#PCDATA)>
```

```
   <!ELEMENT realm-name (#PCDATA)>
   <!ELEMENT form-login-config (form-login-page, form-error-page)>
   <!ELEMENT form-login-page (#PCDATA)>
   <!ELEMENT form-error-page (#PCDATA)>
   -->
</web-app>
```

21.4.3. La création d'une JSP

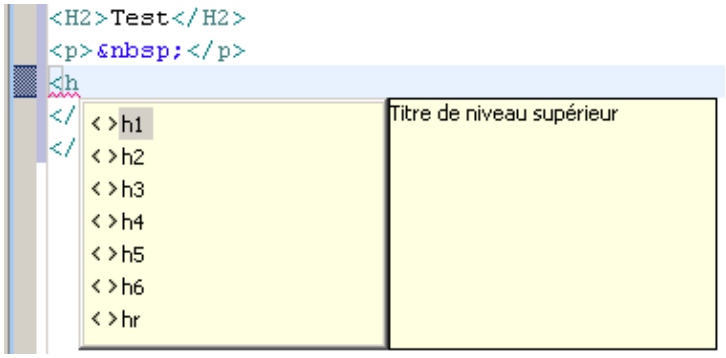
Pour ajouter une JSP, il faut sélectionner l'élément « Dynamic Web Project/WebContent » et d'utiliser l'option « New/File » du menu contextuel



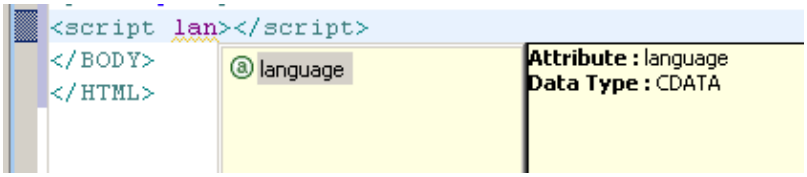
Il suffit alors de saisir le nom du fichier qui va contenir la JSP et de cliquer sur le bouton « Finish ».

L'éditeur de code s'ouvre avec un fichier vide. Cet éditeur propose plusieurs assistants pour la rédaction du code de la JSP

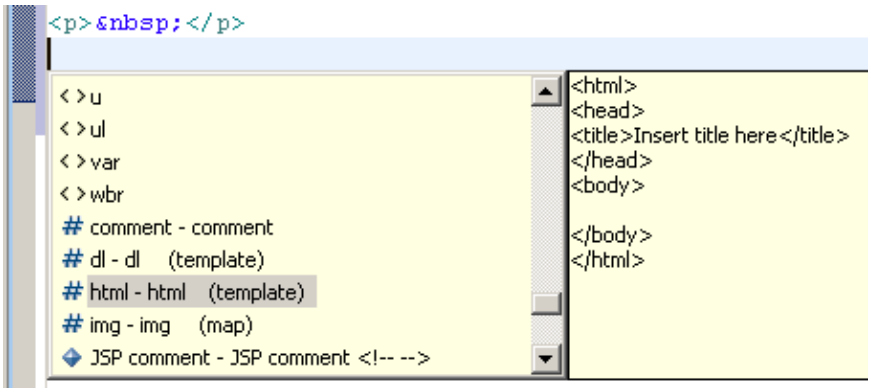
Par exemple pour un tag HTML, il suffit de saisir le début du tag et d'appuyer sur la combinaison de touches Ctrl+espace



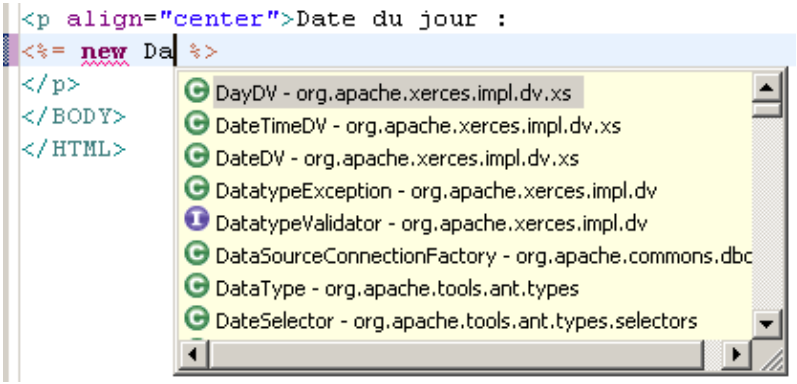
L'assistant permet aussi de saisir les attributs d'un tag.



L'assistant propose aussi des modèles

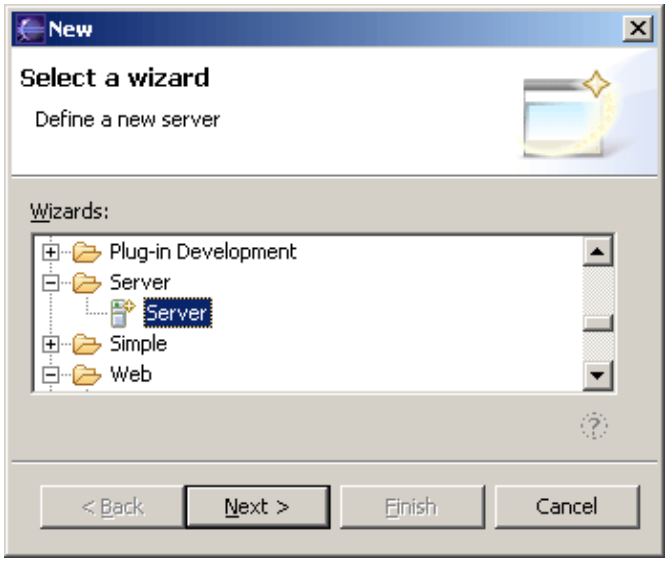


L'assistant propose la complétude de code Java

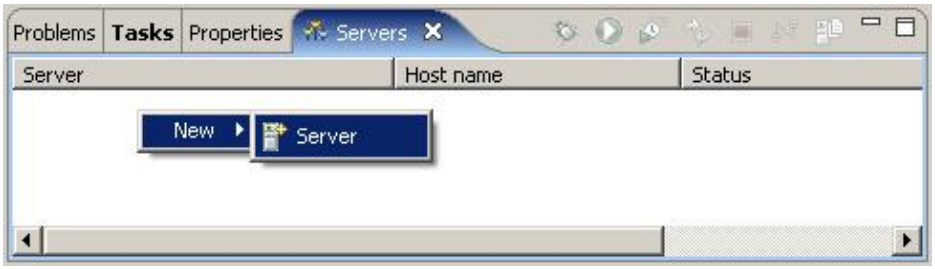


21.4.4. L'exécution de l'application

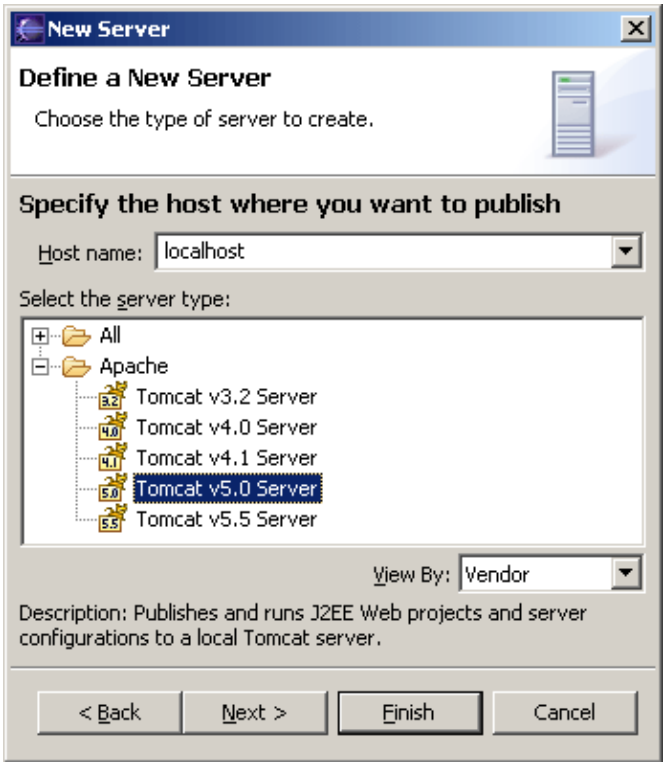
Il est nécessaire de définir un serveur en utilisant l'assistant de création d'entité de type « Server/Server ».



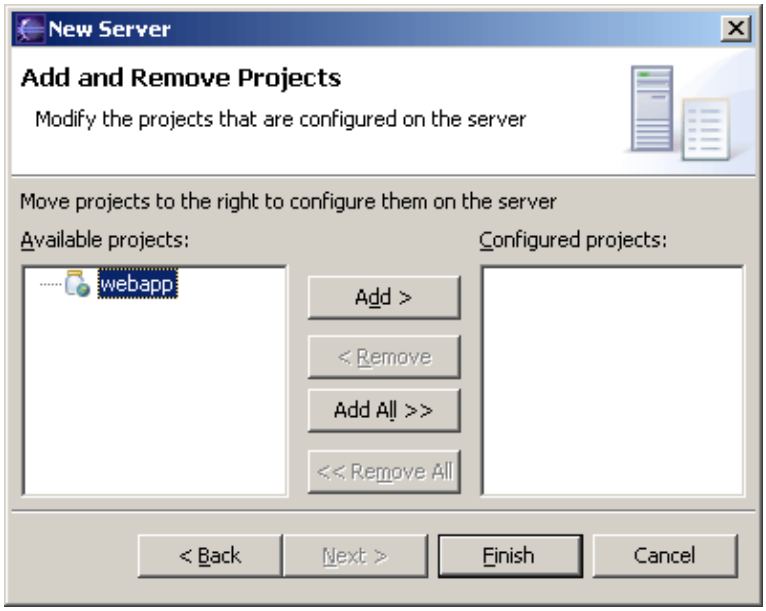
Il est aussi possible de créer un nouveau serveur en utilisant l'option « New/Server » du menu contextuel dans le vue « Server »



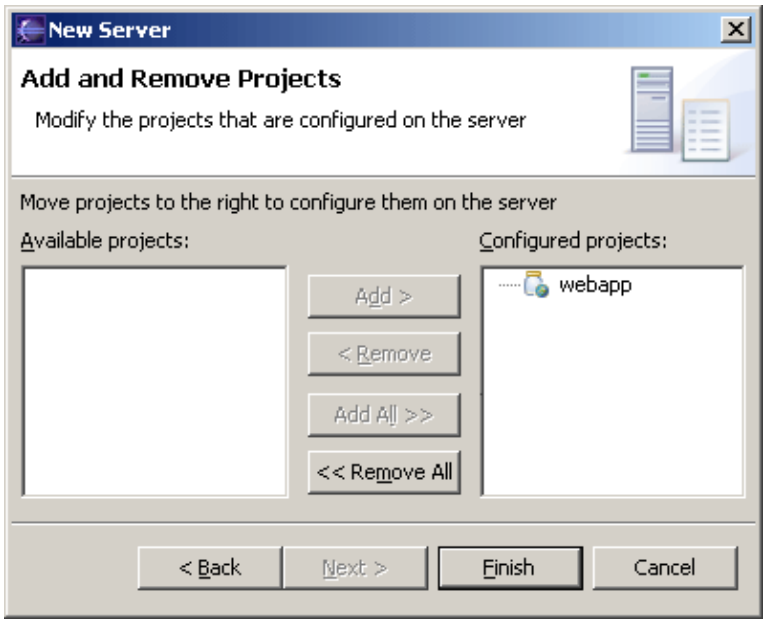
Un assistant permet de saisir les informations concernant le nouveau serveur.



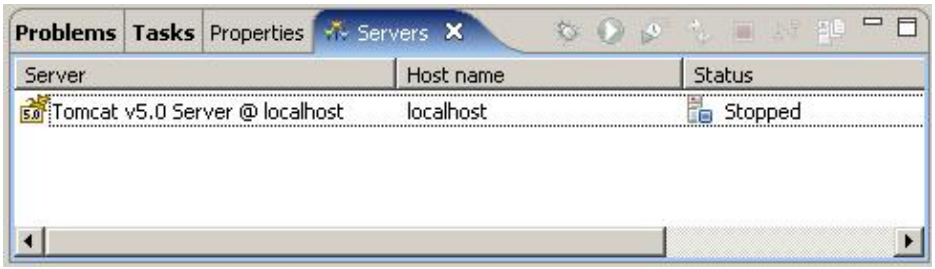
Sélectionnez la version de Tomcat installée et cliquez sur le bouton « Next »



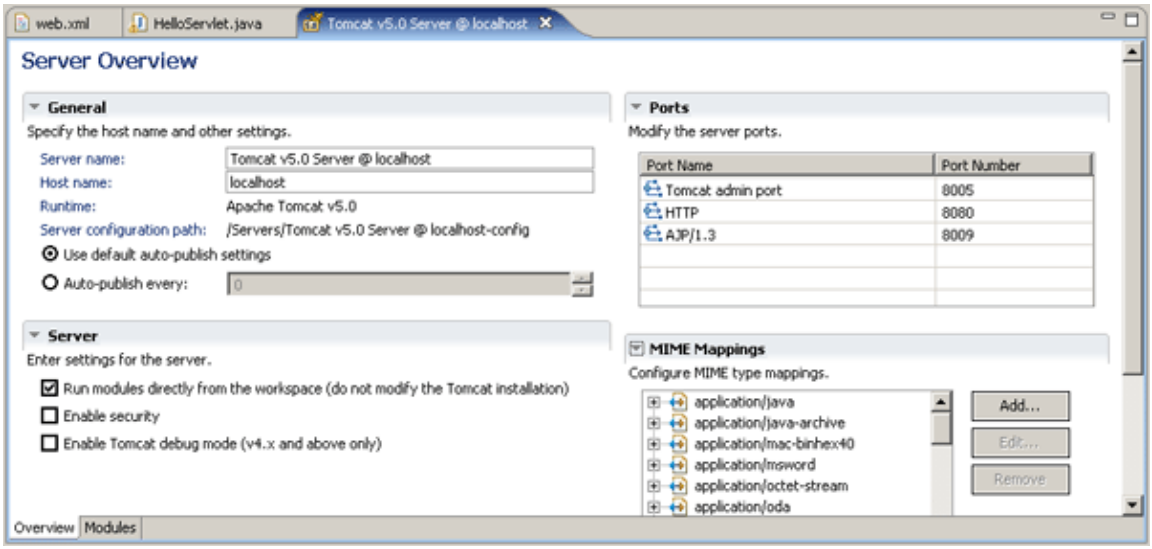
Sélectionnez le projet puis cliquez sur le bouton « Add > »



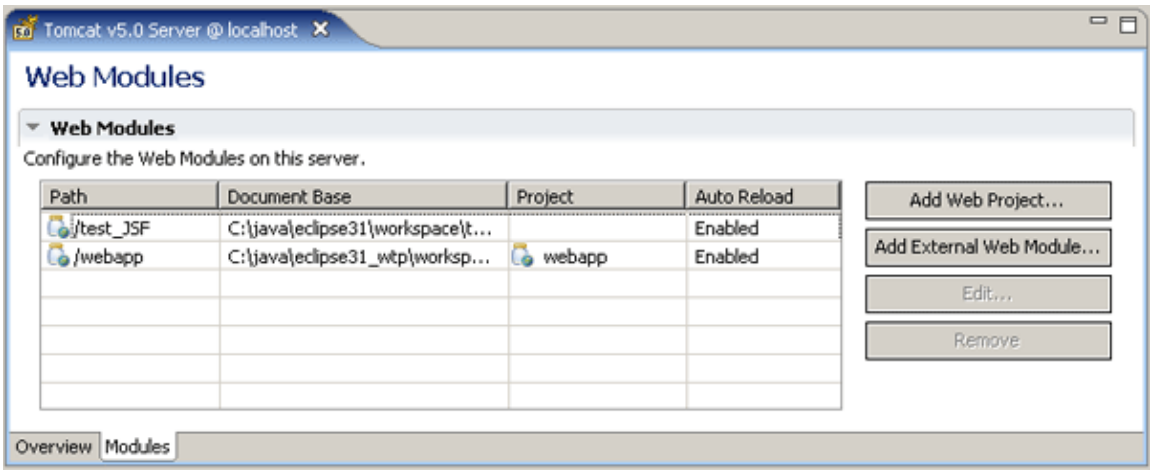
Cliquez sur le bouton « Finish »



L'option « Open » du menu contextuel sur le serveur permet d'obtenir des informations sur ce dernier.



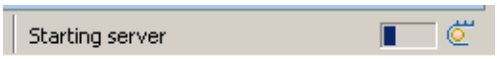
L'onglet « Modules » permet d'obtenir la liste des applications installées sur le serveur.



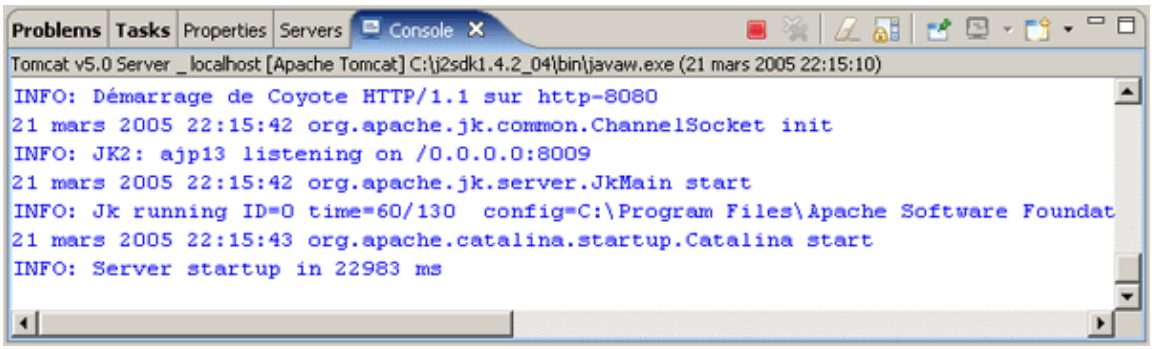
Pour lancer un serveur, il y a plusieurs solutions après avoir sélectionné ce serveur :

- Cliquez sur le bouton 
- Utilisez l'option « Start » du menu contextuel

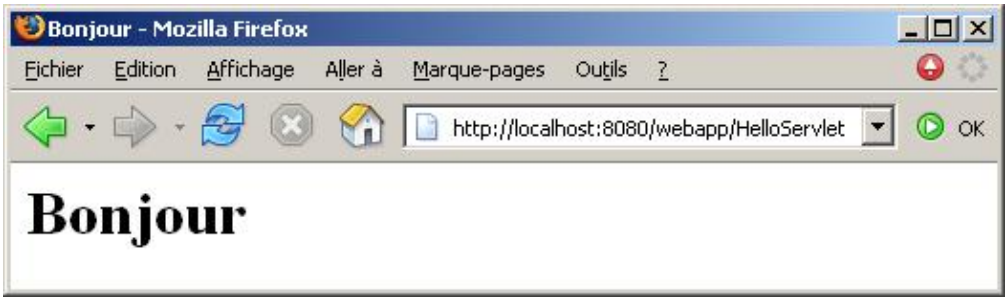
Le serveur est lancé en arrière plan



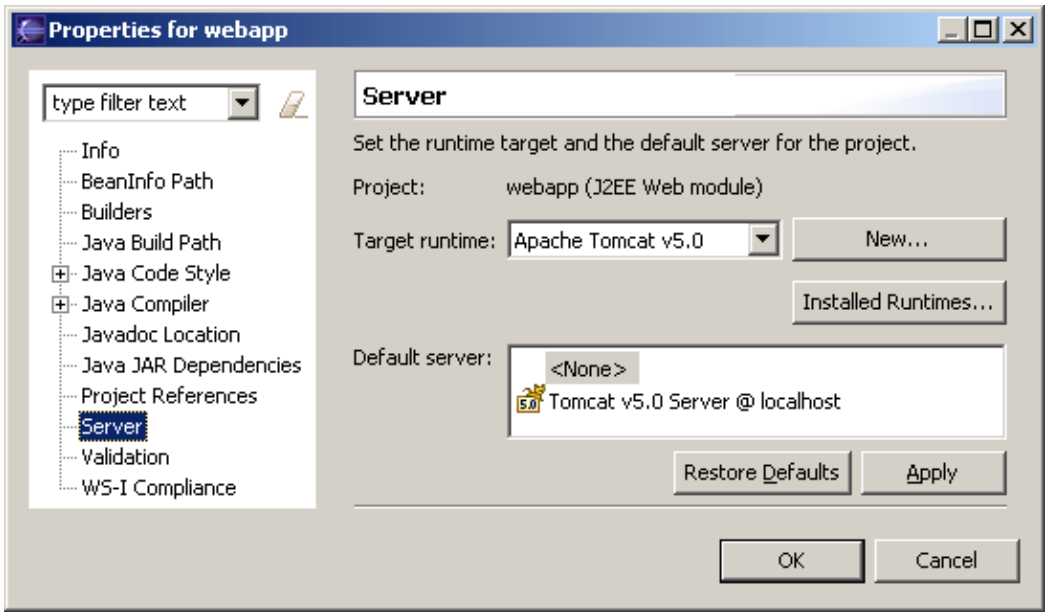
Les informations de démarrage du serveur sont affichées dans la vue « Console »



Il suffit alors d'ouvrir un navigateur et de saisir l'url de la servlet pour permettre son affichage

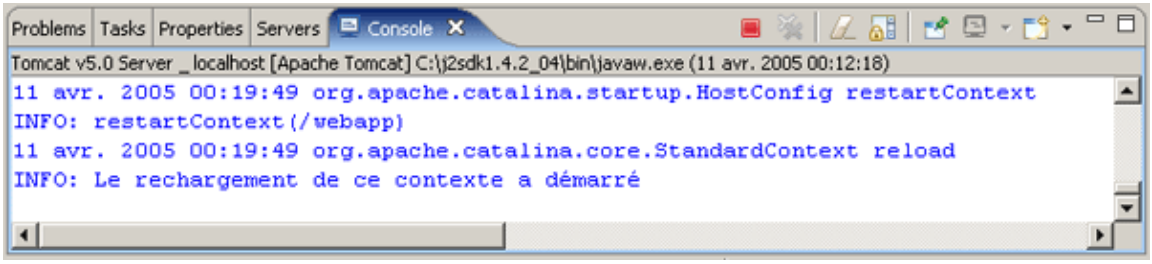


Il est possible de définir un serveur par défaut pour le projet dans les propriétés de ce dernier.

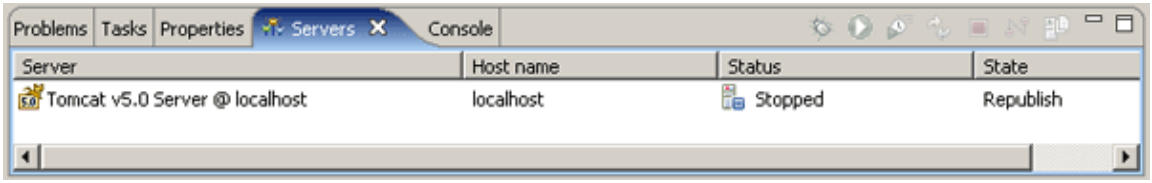


Il suffit alors de sélectionner le serveur précédemment défini, de cliquer sur le bouton « Apply » puis sur le bouton « OK ».

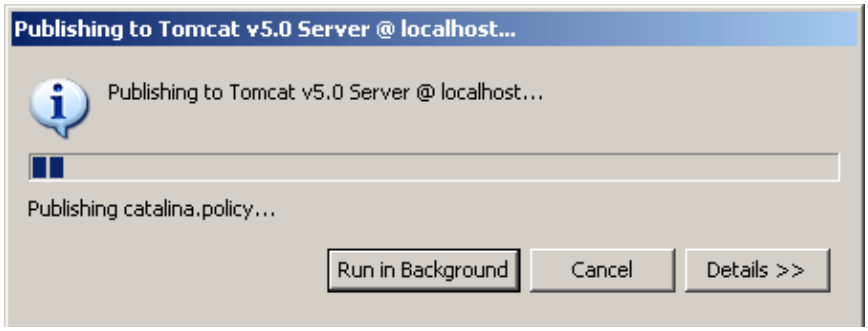
Si des modifications sont apportées à l'application alors que le serveur la concernant est lancé, le contexte est automatiquement rechargé.



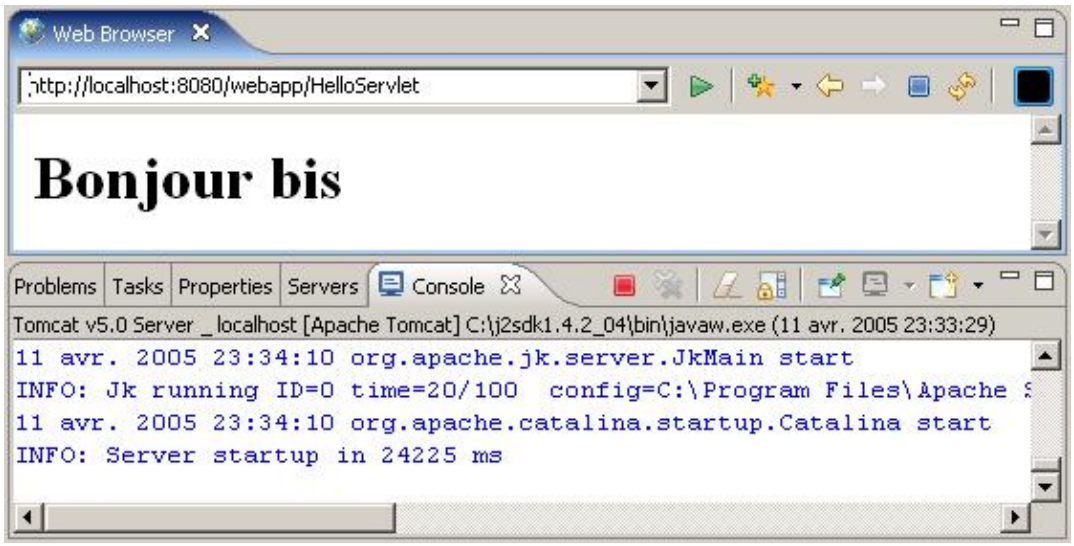
Si des modifications sont apportées à l'application en cours de développement, le statut du serveur est « Republish »



Sélectionnez l'option  Publish du menu contextuel du serveur.



Pour les exécutions suivantes, il est possible d'utiliser l'option « Run As/Run On Server » pour lancer le serveur et ouvrir le navigateur intégré d'Eclipse avec l'url de l'application

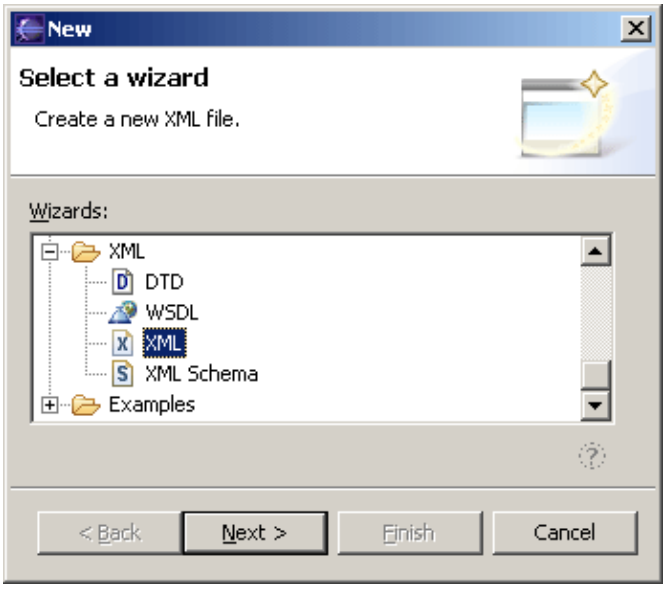


21.5. XML

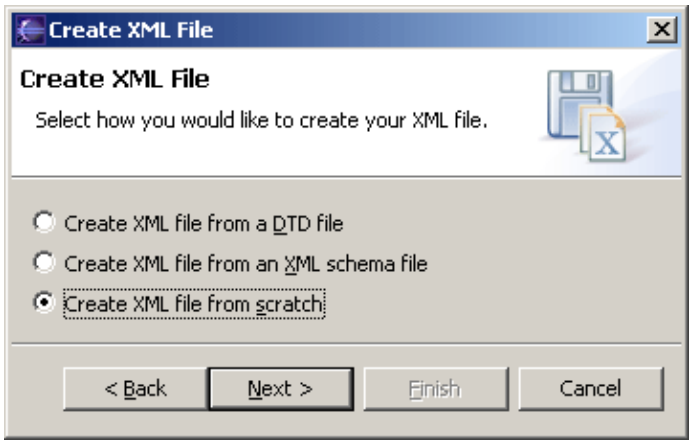
Le plug−in WTP facilite la manipulation de documents XML.

21.5.1. Créer un nouveau document XML

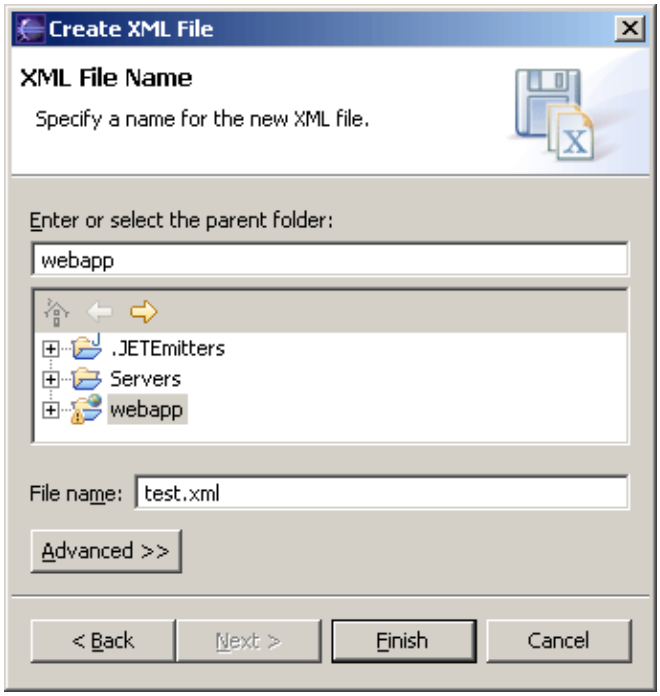
Il faut créer une nouvelle entité de type XML/XML



Cliquez sur le bouton « Next »

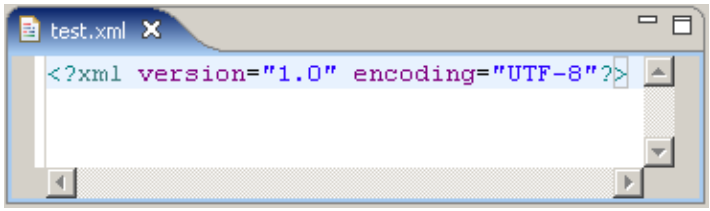


Sélectionnez « Create XML file from scratch » puis cliquez sur le bouton « Next »



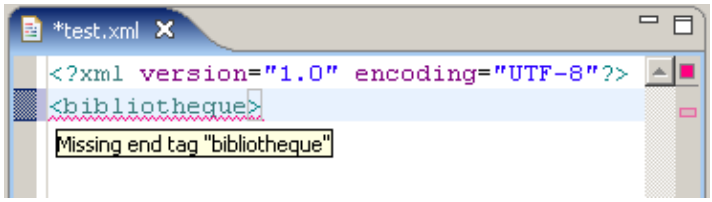
Sélectionnez le répertoire qui va contenir le fichier, saisissez son nom et cliquez sur le bouton « Finish »

L'éditeur de documents XML s'ouvre avec le nouveau document créé.

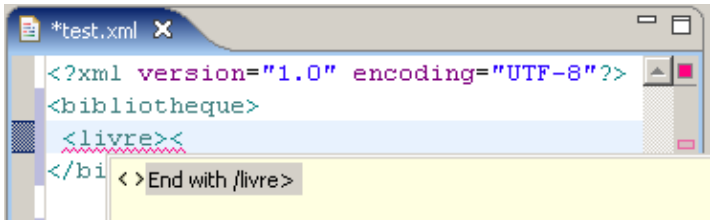


L'éditeur propose une coloration syntaxique du code.

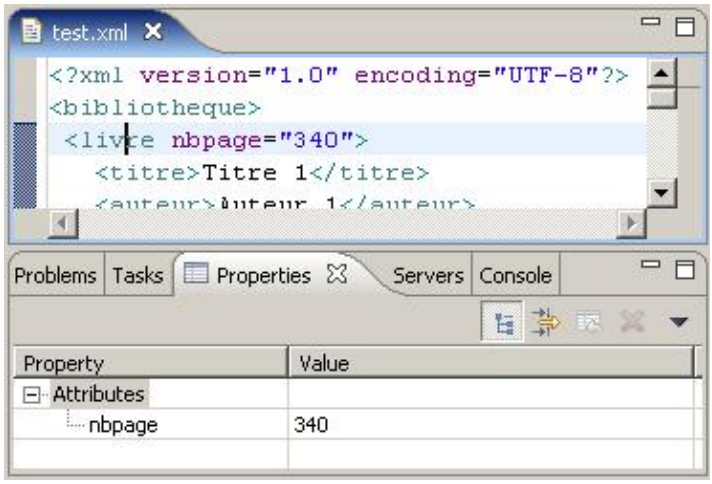
La syntaxe du code XML est vérifiée au fur et à mesure de la saisie



L'assistant de code permet de faciliter la saisie du code du document.

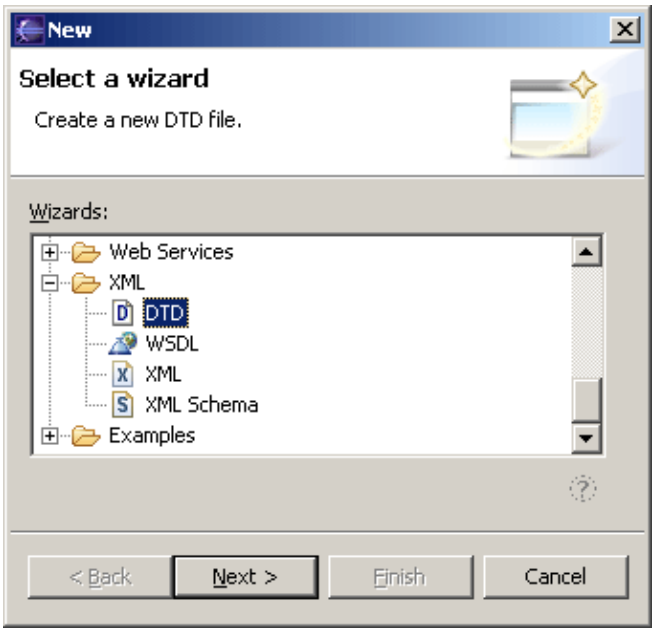


La vue « Properties » permet d'afficher les attributs du tag sur lequel est positionné le curseur dans l'éditeur de code

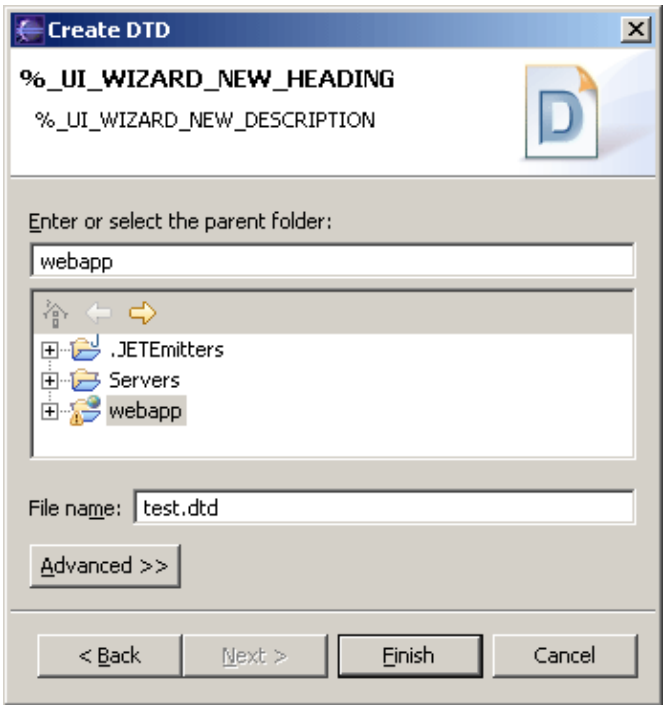


21.5.2. La création d'une DTD

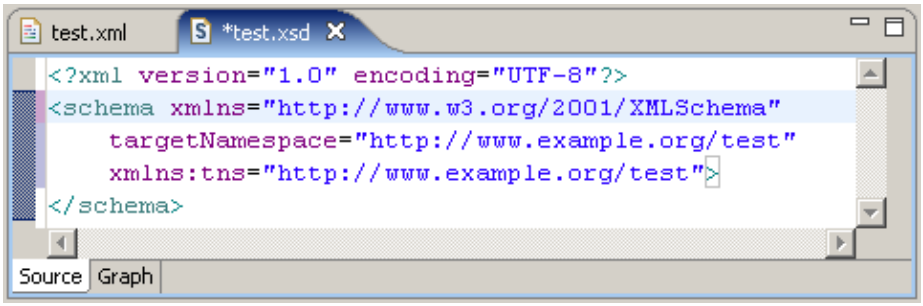
Il faut créer une nouvelle entité du type « XML/DTD ».



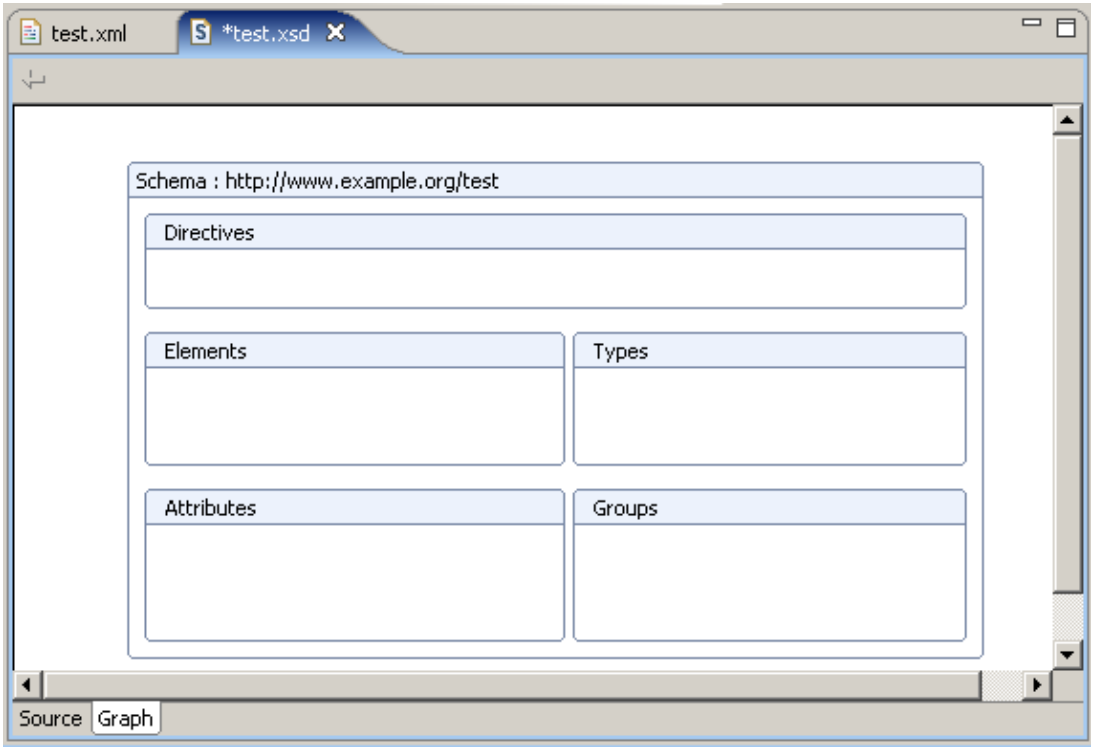
Sélectionnez le répertoire qui va contenir le fichier et saisissez son nom



Cliquez sur le bouton « Finish » pour générer le fichier et l'ouvrir dans l'éditeur.

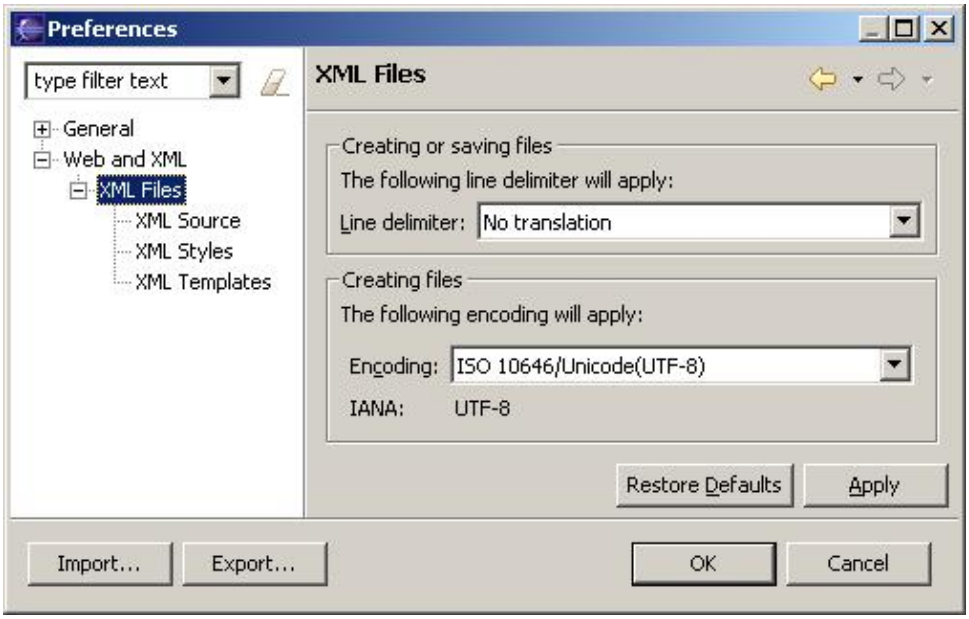


L'onglet "Graph" permet d'offrir une vision graphique du contenu du document.

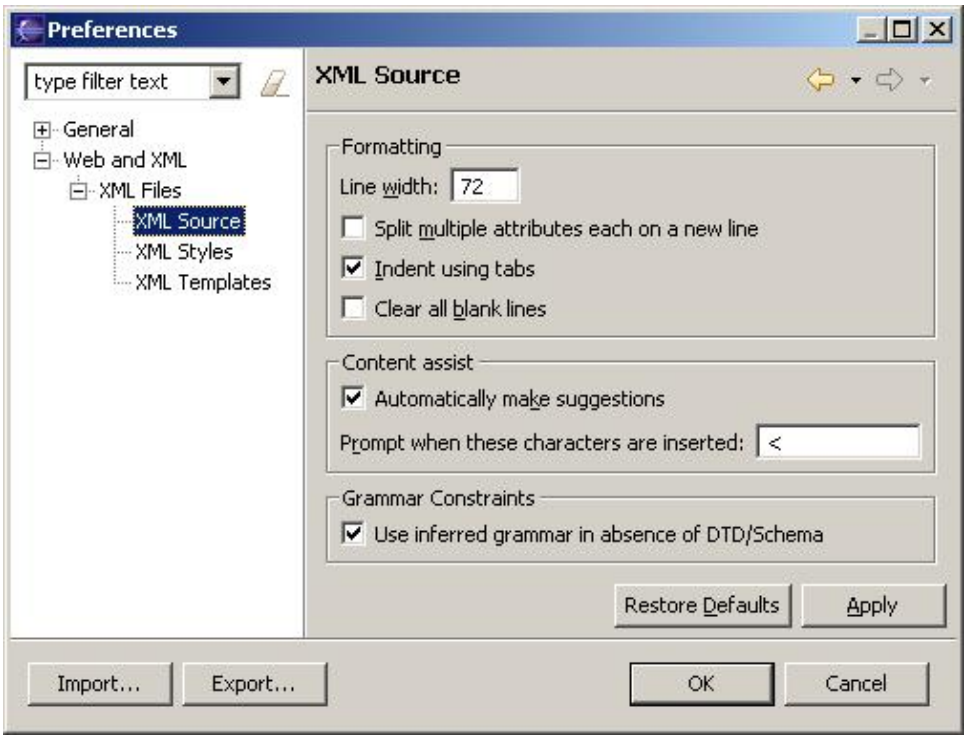


21.5.3. Les préférences

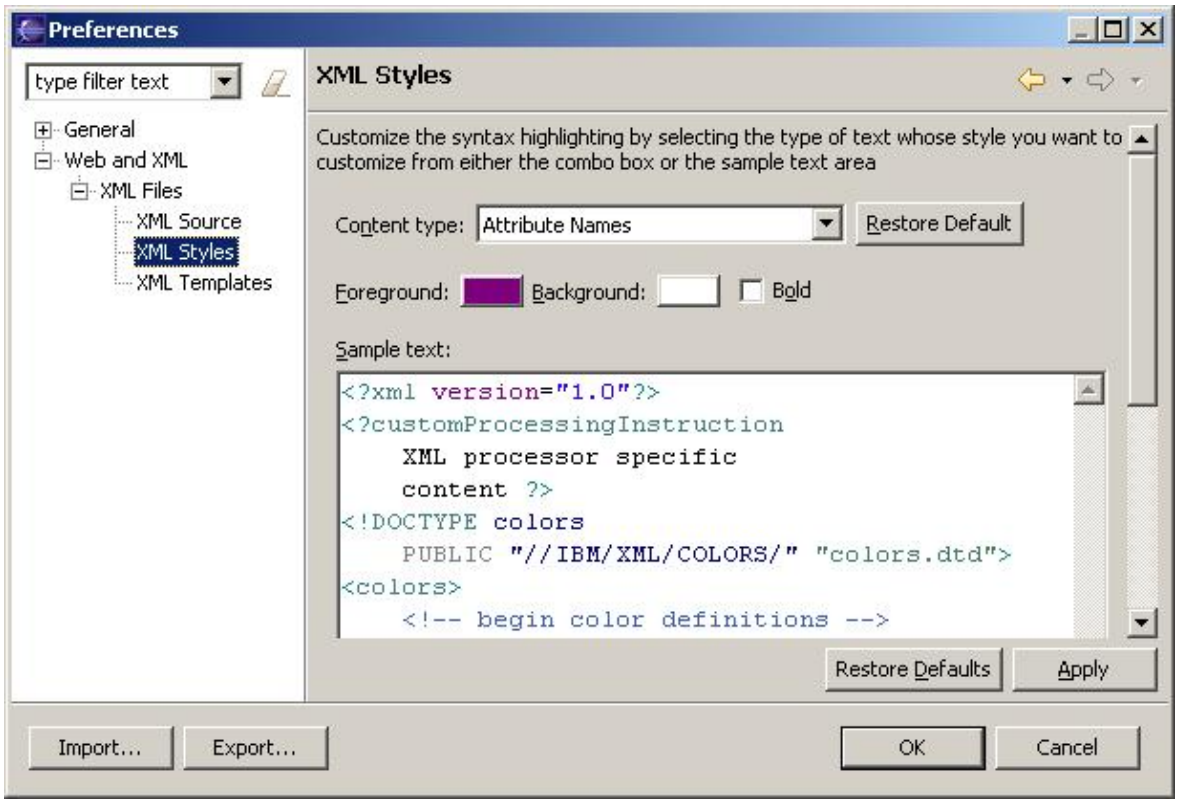
Pour accéder aux préférences concernant les fichiers XML, il est possible d'utiliser l'option « Preferences » du menu « Window » ou d'utiliser l'option « Preference » du menu contextuel de l'éditeur de code XML.



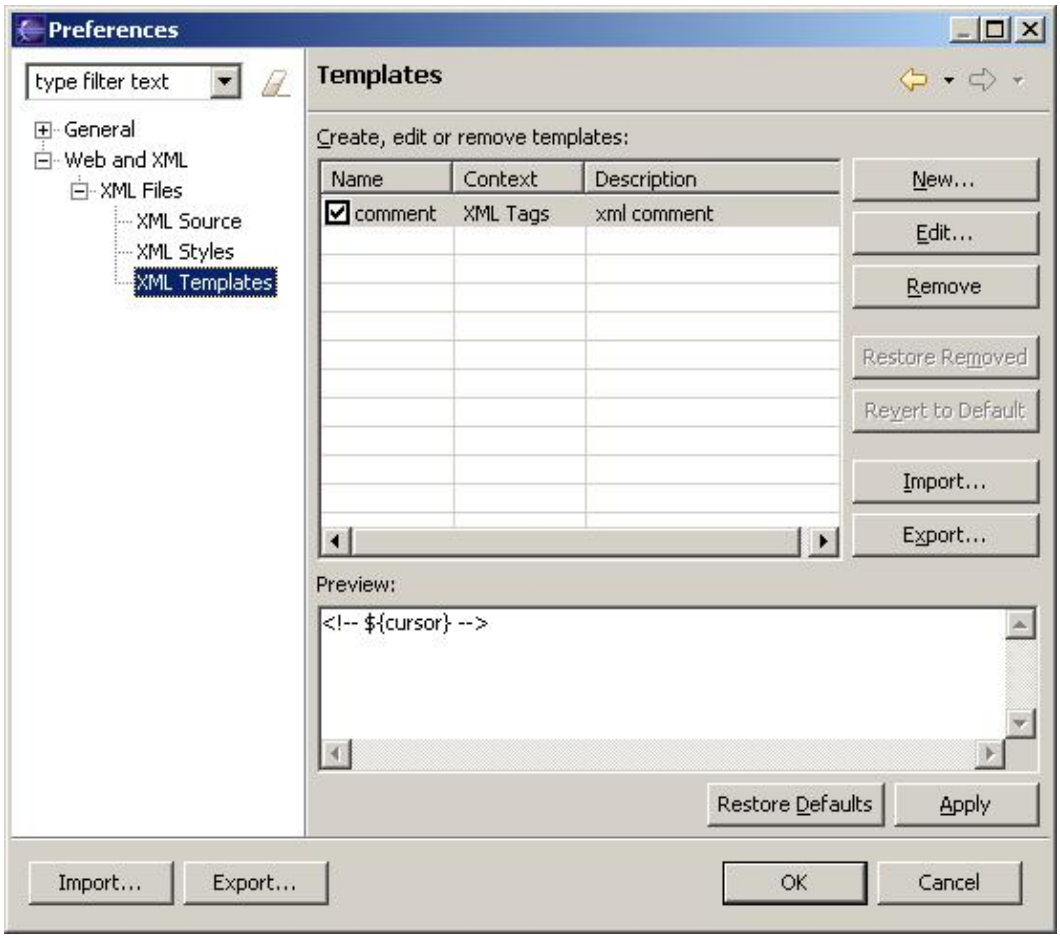
Cette page permet de sélectionner le caractère de fin de ligne des documents et la norme d'encodage par défaut des documents.



Cette page permet de configurer certaines règles pour le formatage du code source des documents XML.



Cette page permet de préciser les couleurs de chaque éléments qui peuvent composer un document XML.



Cette page permet de gérer des modèles réutilisables lors de l'édition de documents XML.

21.5.4. Création d'un document XML à partir d'une DTD

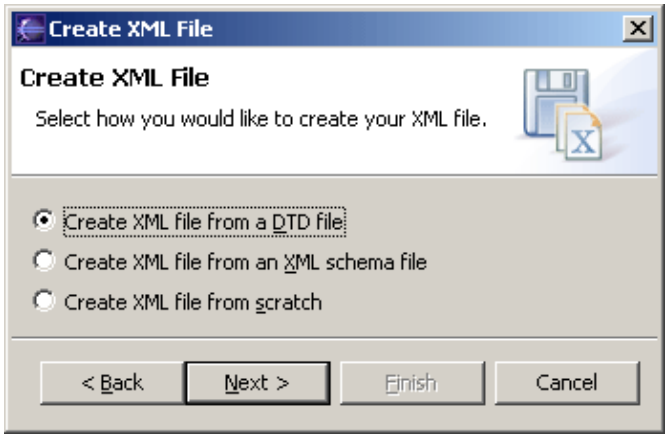
L'assistant de création d'un document XML propose de le faire à partir d'une DTD afin de faciliter sa rédaction

Le fichier test.dtd ci−dessous est utilisé dans les exemples de cette section.

Exemple :

```
<!ELEMENT bibliotheque (livre+)>
<!ELEMENT livre (titre,auteur,parution?)>
<!ELEMENT titre (#PCDATA)>
<!ELEMENT auteur (#PCDATA)>
<!ELEMENT parution (#PCDATA)>
```

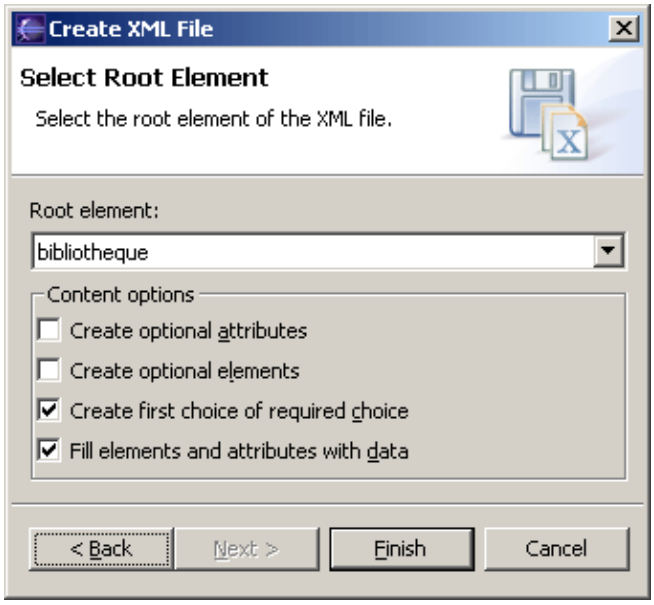
Il faut créez une nouvelle entité de type XML/XML



Sélectionnez « Create XML file from a DTD file » et cliquez sur le bouton « Next ».

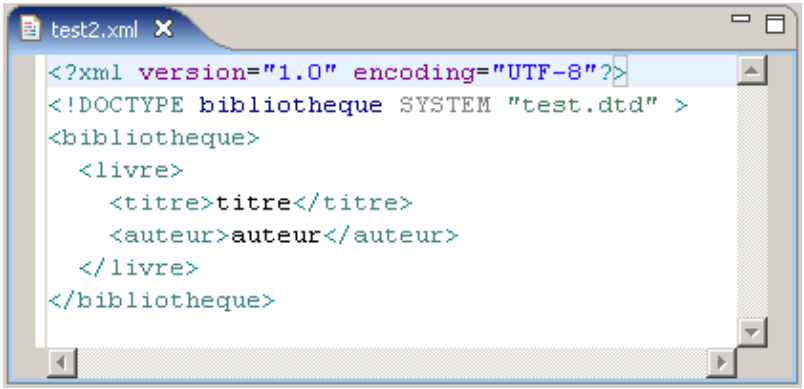
Sélectionnez le répertoire qui va contenir le fichier, saisissez son nom et cliquez sur le bouton « Next »

Sélectionnez la dtd à utiliser et cliquez sur le bouton « Next »

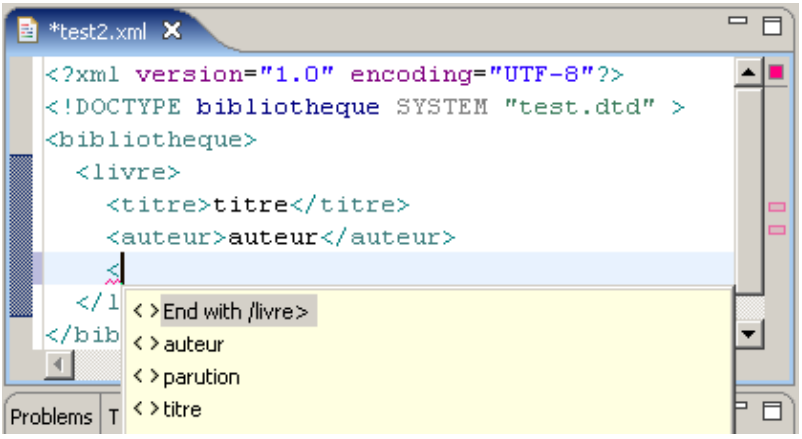


Sélectionnez l'élément racine et les options désirées puis cliquez sur le bouton « Finish ».

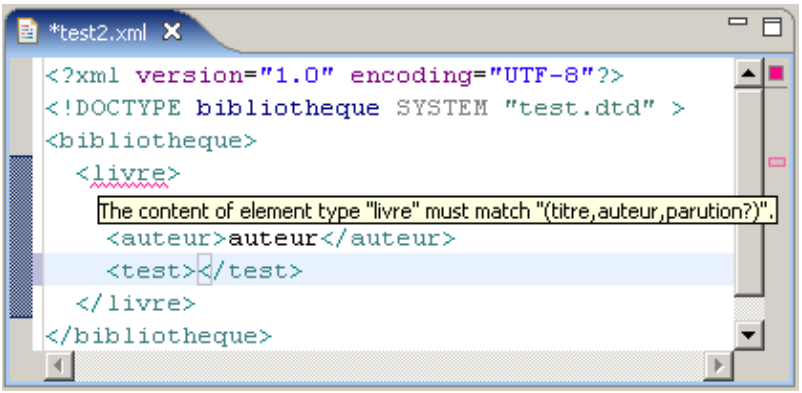
L'éditeur de code s'ouvre avec le document XML généré



L'assistant de code propose uniquement les tags définis dans la dtd.



Le document est dynamiquement vérifié au fur et à mesure de la saisie du code du document



21.5.5. La validation des documents

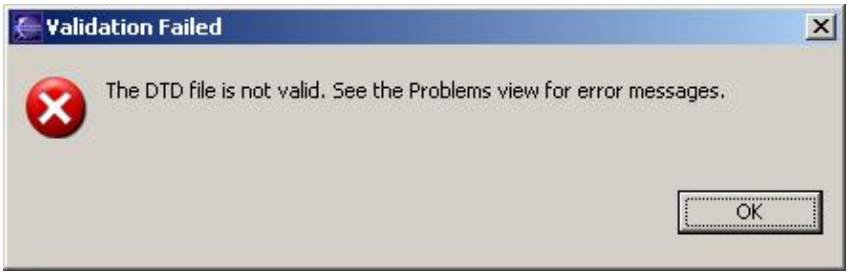
Le plug in WTP propose des fonctionnalités pour valider les documents de type XML, DTD et XML Schema.

Cette validation peut être implicite (lors de la sauvegarde d'un document) ou explicite (à la demande de l'utilisateur d'Eclipse).

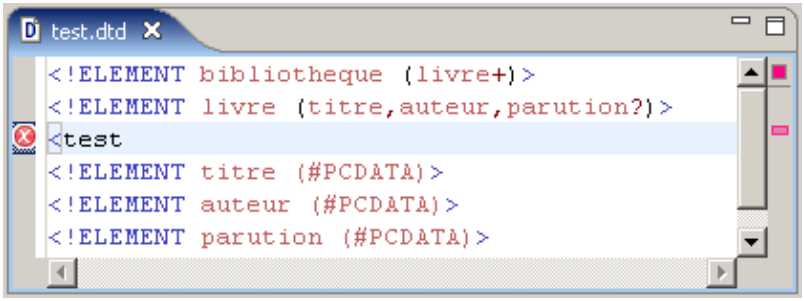
La validation explicite peut être demandée en utilisant l'option « Validate xxx File » du menu contextuel associé à un document de type XML, DTD et XML Schema.

Si le document est valide, un message en informe l'utilisateur.

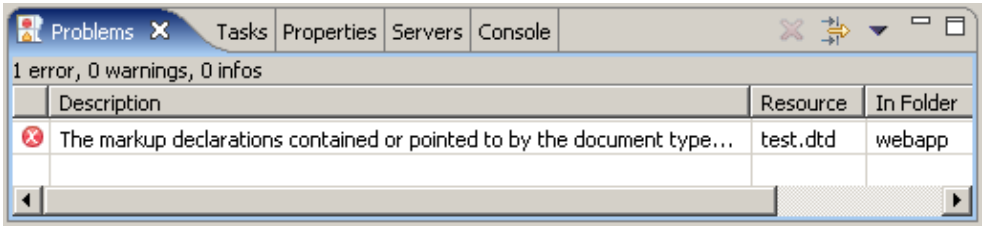
Si le document est invalide, un message d'erreur en informe aussi l'utilisateur.



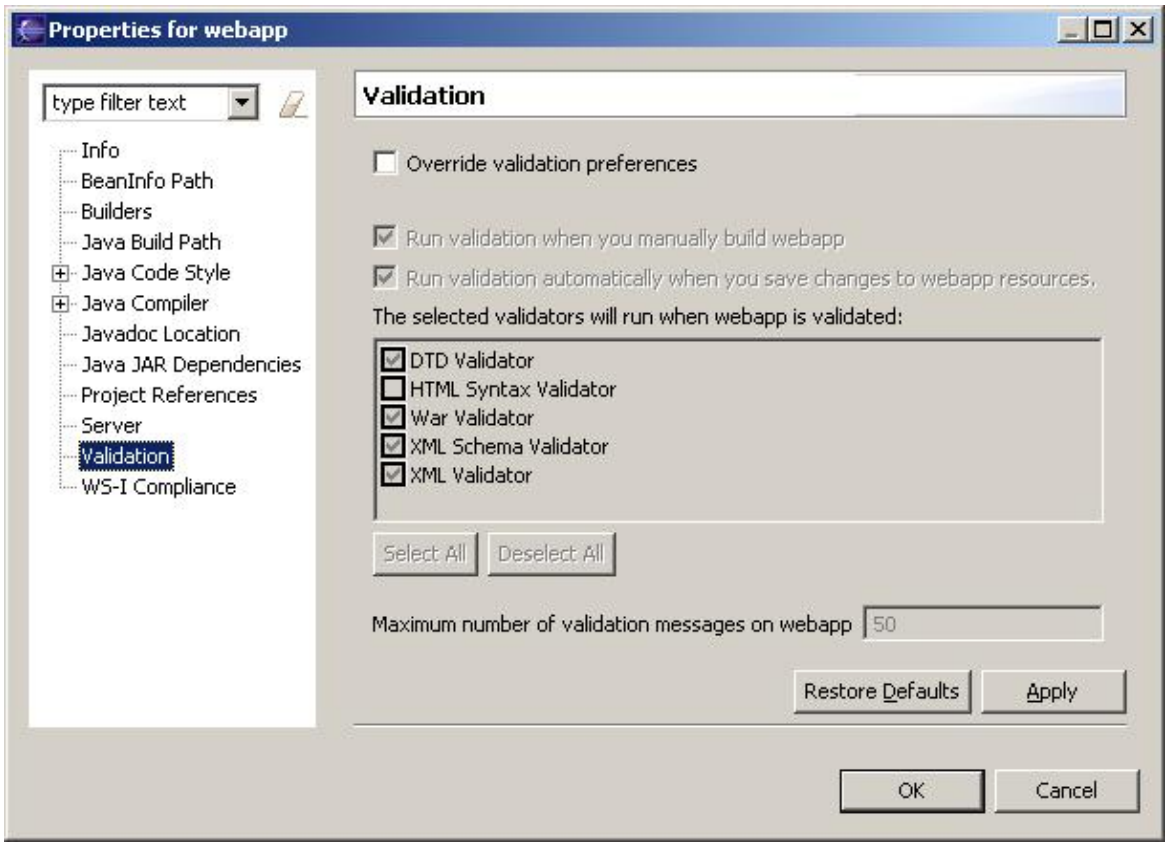
Les erreurs des documents non valides sont signalées dans l'éditeur de code



Ces erreurs sont aussi incluses dans la vue « Problems » au même titre que les erreurs dans du code Java.



Il est possible de modifier le mode de fonctionnement de la validation automatique lors de l'enregistrement des documents d'un projet en utilisant les préférences liées au projet.



Il suffit de cocher la case « Override validation preferences », de modifier les options voulues et de cliquez sur le bouton « OK » pour valider les options.

21.6. Le développement de services web

Cette section sera développée dans une version future de ce document

21.7. Le développement d'EJB

Cette section sera développée dans une version future de ce document

Partie 5 : d'autres plug−ins

Cette dernière partie présente des plug−ins tiers qui ne sont pas dédié directement aux développements avec Java.

Elle comporte les chapitres suivants :

- Le développement sans Java : présente des plug−ins pour réaliser des développements avec des langages différents de Java.
- EclipseUML : présente le plug−ins EclipseUML pour utiliser UML avec Eclipse.
- Eclipse et les bases de données : présente plusieurs plug−ins pour réaliser des opérations sur des bases de données.
- Eclipse et Hibernate : Présente le plug−in Hibernate Synchronizer qui facilite la mise en oeuvre de l'outil de mapping Hibernate.
- Eclipse et J2ME : Présente le plug−in EclipseME qui permet de faciliter le développement d'applications J2ME.

22. Le développement sans Java

Eclipse est un outil de développement, écrit en Java, dont un des buts initiaux est de pouvoir développer non seulement en Java mais aussi avec d'autres langages grâce à des plug−ins spécifiques. Il existe déjà plusieurs plug−ins plus ou moins avancés pour réaliser des développements en C/C++ et Cobol développés par le projet Eclipse mais aussi C# ou PHP développés par des tiers.

22.1. CDT pour le développement en C / C++

CDT (C/C++ Development Tools) est un sous projet du projet "Eclipse Tools" dont le but est de fournir un environnement intégré de développement en C /C++ sous la forme de plug−ins pour Eclipse.

Le CDT ajoute une perspective nommée "C/C++" au workbench.

Le CDT ne fournit pas de compilateur : il est nécessaire d'utiliser un compilateur externe. Le seul compilateur actuellement supporté par le CDT est le celèbre compilateur GCC du projet GNU. D'autres outils du projet GNU sont aussi nécessaires tel que make ou GDB (GNU Debugger).

Sous linux, ces outils peuvent être facilement installés en utilisant les packages adéquats selon la distribution utilisée.

Sous Windows, l'utilisation de ces outils peut être mise en oeuvre grâce à l'installation d'un des deux outils suivants :

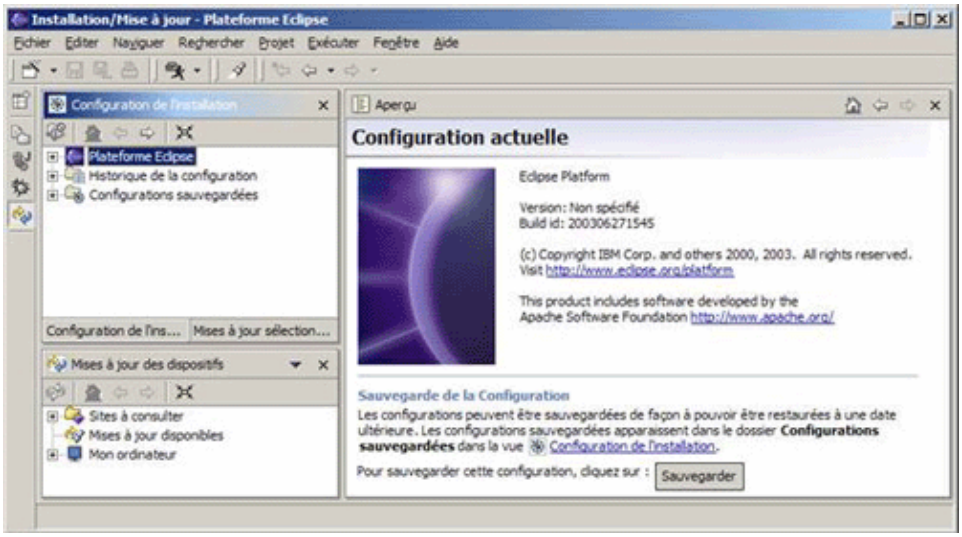
- Cygwin : c'est un projet de la société Red Hat (http://www.redhat.com/software/cygwin/)
- MinGW (Minimalist GNU for Windows) : c'est un projet open source qui a pour but de fournir un ensemble de fichier en−tête et de bibliothèques pour générer des exécutables natifs sous Windows en utilisant des outils du projet GNU. (http://www.mingw.org/)

Dans ce chapitre, l'installation et l'utilisation de MinGW avec le CDT sera détaillée pour une utilisation sur une plate−forme Windows.

Bien qu'écrit en Java, le CDT est dépendant de la plate−forme sur laquelle il s'exécute.

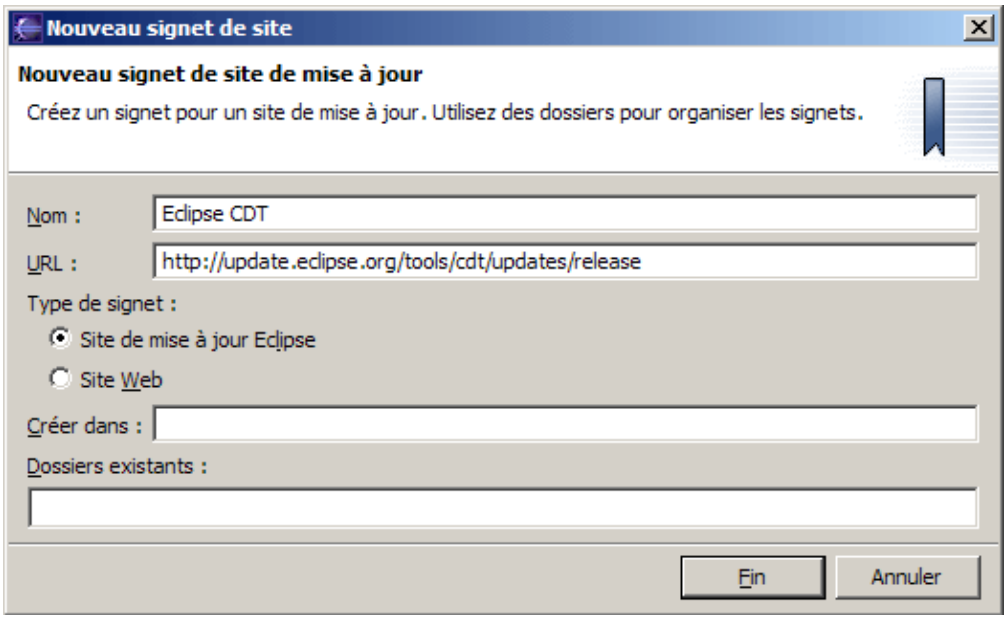
22.1.1. Installation du CDT

Pour installer le CDT, il faut utiliser le mécanisme de mise à jour en utilisant l'option "Mise à jour des logiciels / Gestionnaire des mises à jour" du menu "Aide".

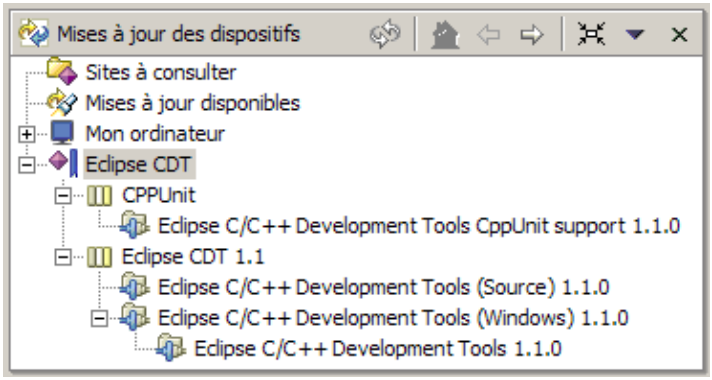


Dans la vue mise à jour des dispositifs, utiliser l'option « Nouveau / Signet du site » du menu contextuel.

Il faut saisir un nom par exemple "Eclipse CDT" et saisir l'url "http://update.eclipse.org/tools/cdt/updates/release".

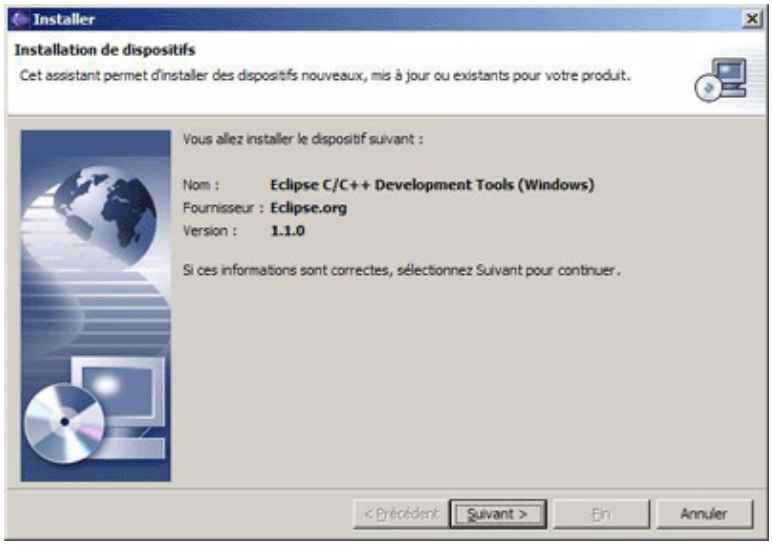


Cliquez sur le bouton "Fin" pour permettre à Eclipse de rechercher les mises à jour. La vue "Mise à jour des dispositifs" affiche celles disponibles.

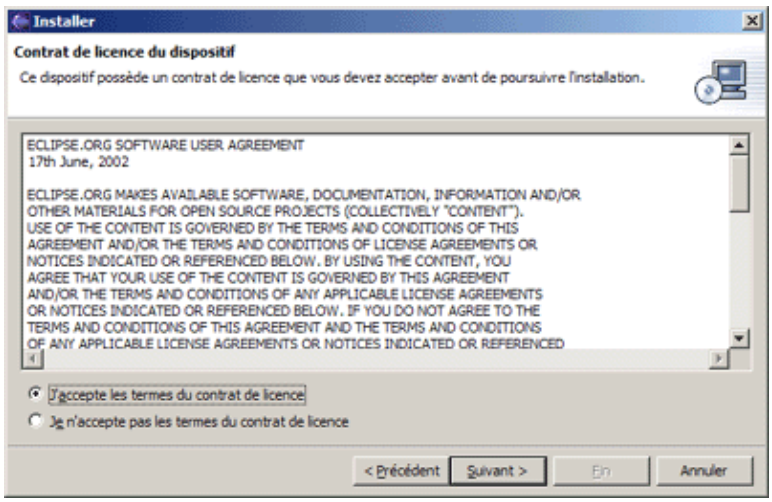


Sélectionnez la mise à jour « Eclipse C/C++ Development Tools (Windows). »

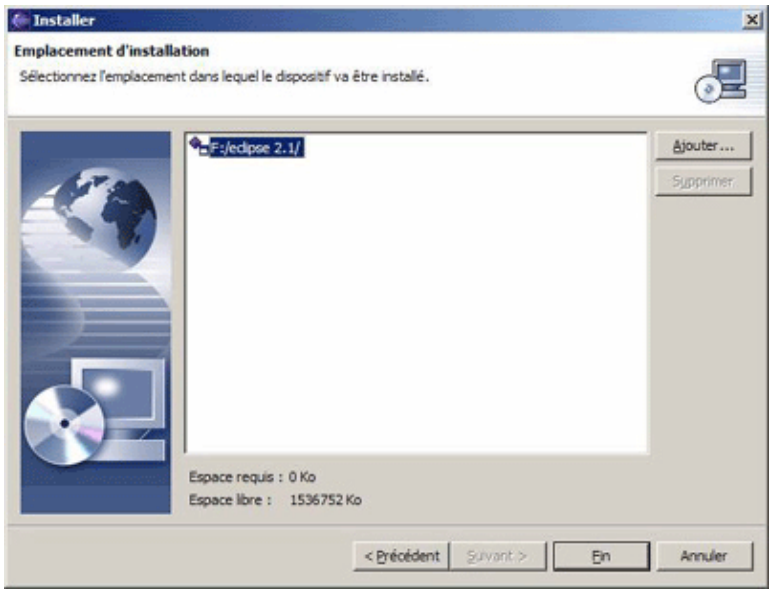
La vue "Aperçu" affiche des informations sur le plug−in. Cliquez sur le bouton « Installer Maintenant »



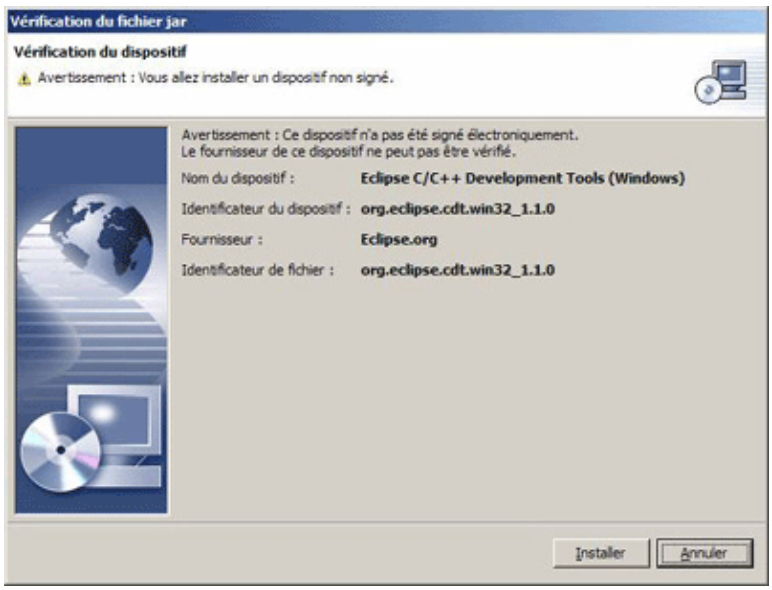
Cliquez sur le bouton « Suivant »



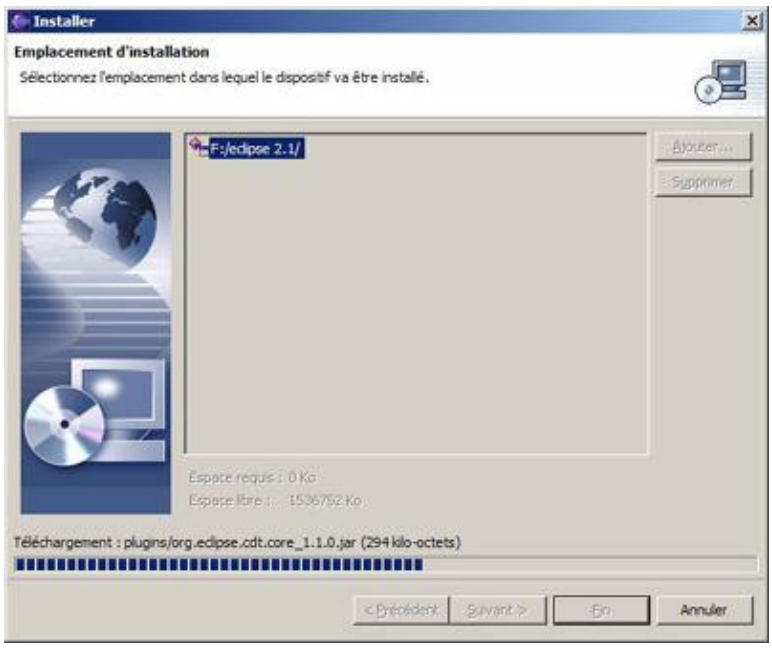
Lisez la licence et si vous acceptez les termes du contrat, cliquez sur le bouton « Suivant »



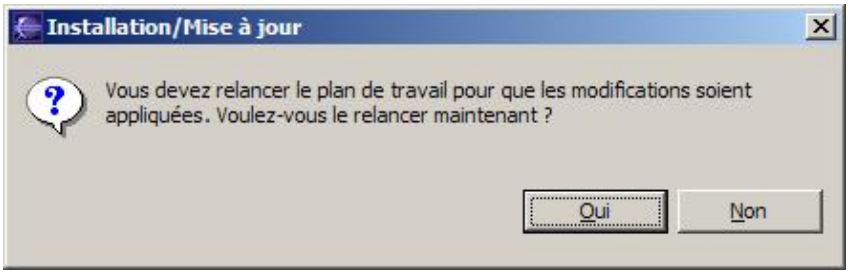
Cliquez sur le bouton « Fin »



Cliquer sur le bouton « Installer ».



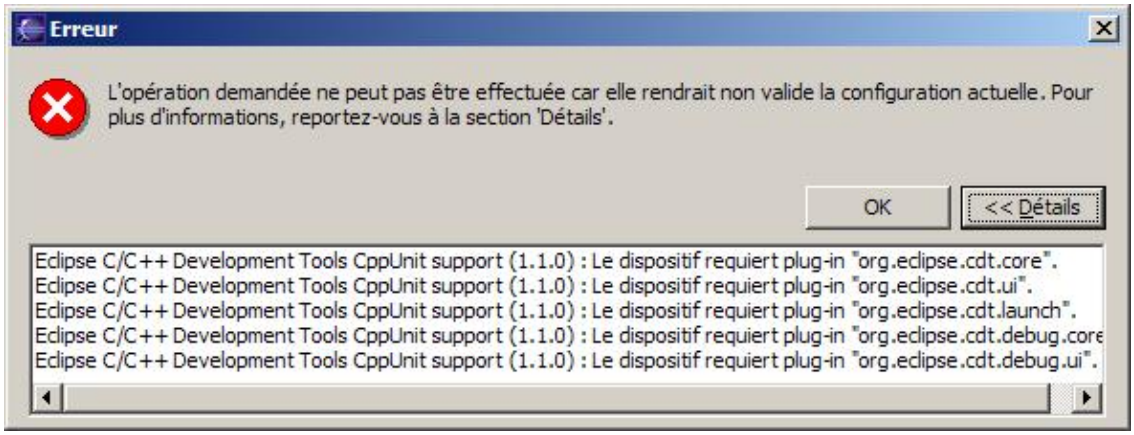
L'installation nécessite un redémarrage de l'application.



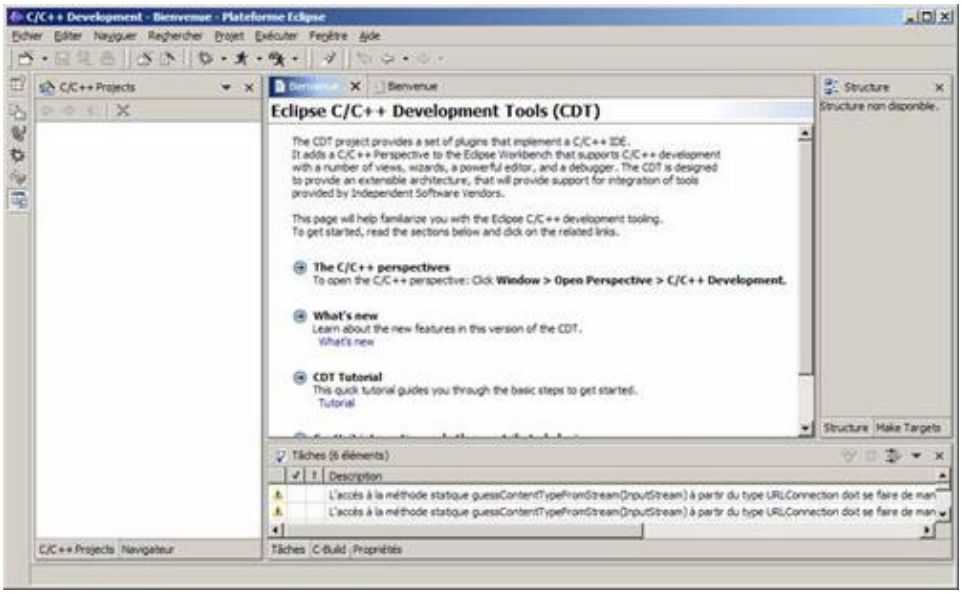
Cliquez sur le bouton « Oui ».

Faites la même opération avec la mise à jour « Eclipse C/C++ development tools CppUnit support 1.1.0. »

Si vous installez cette mise à jour avant la précédente, le message d'erreur suivant est affiché.

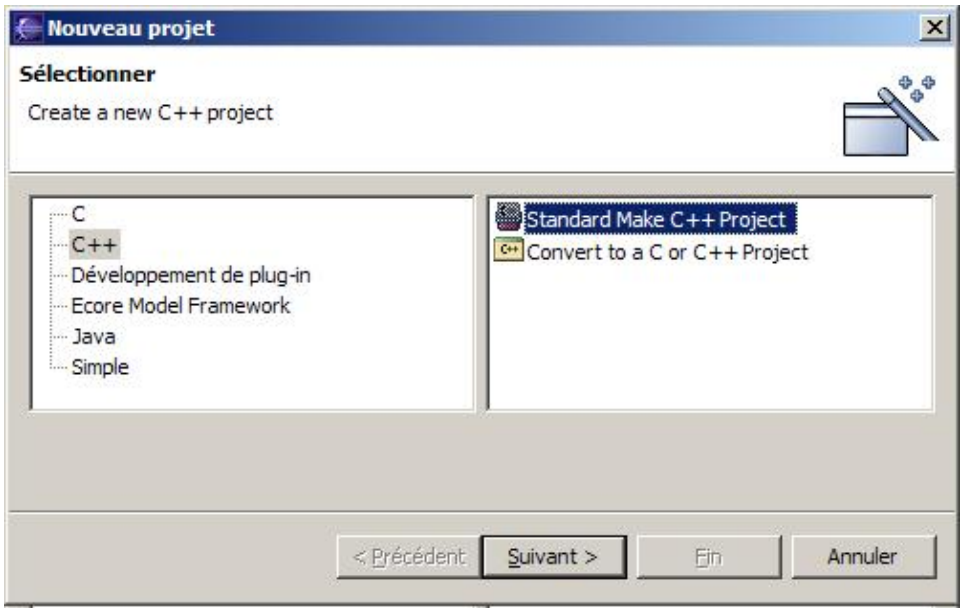


Une fois l'installation terminée, l'éditeur de la prespective ressource affiche des informations sur le CDT.

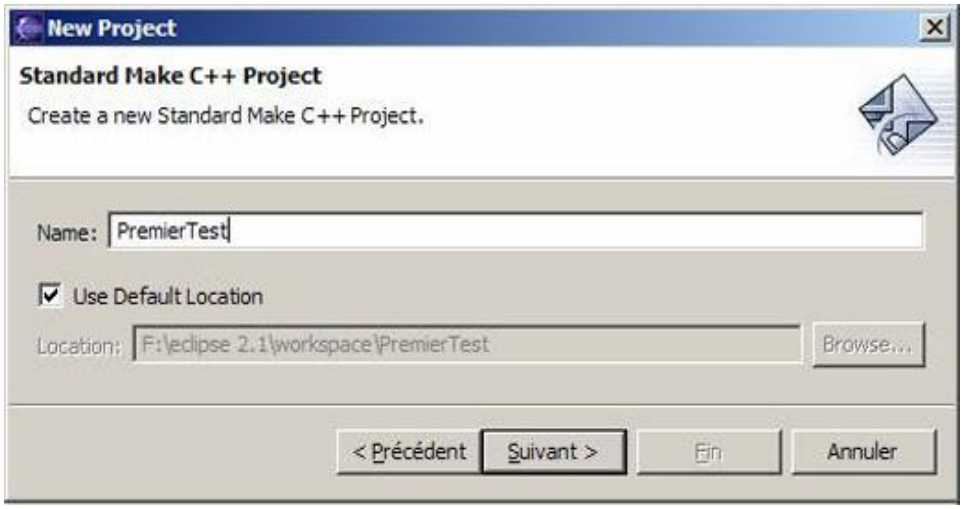


22.1.2. Création d'un premier projet

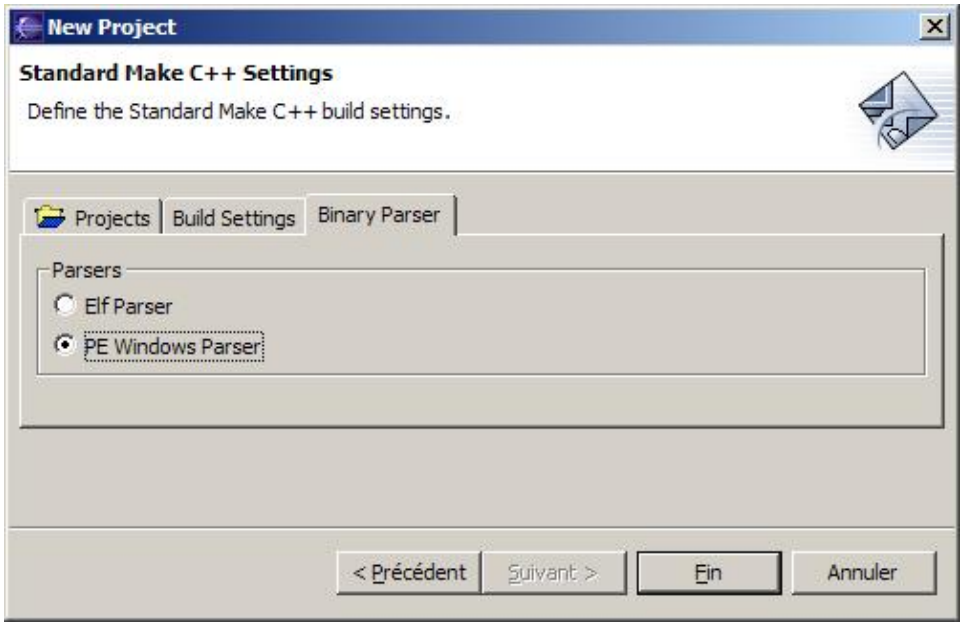
Pour pouvoir utiliser le CDT, il faut tout d'abord créer un nouveau projet.



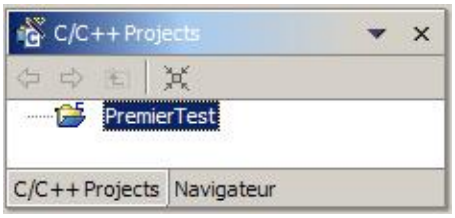
Pour un projet en C++, sélectionnez "C++" dans la liste de gauche puis "Standard Make C++ Project" dans la liste de droite et enfin cliquez sur le bouton "Suivant".



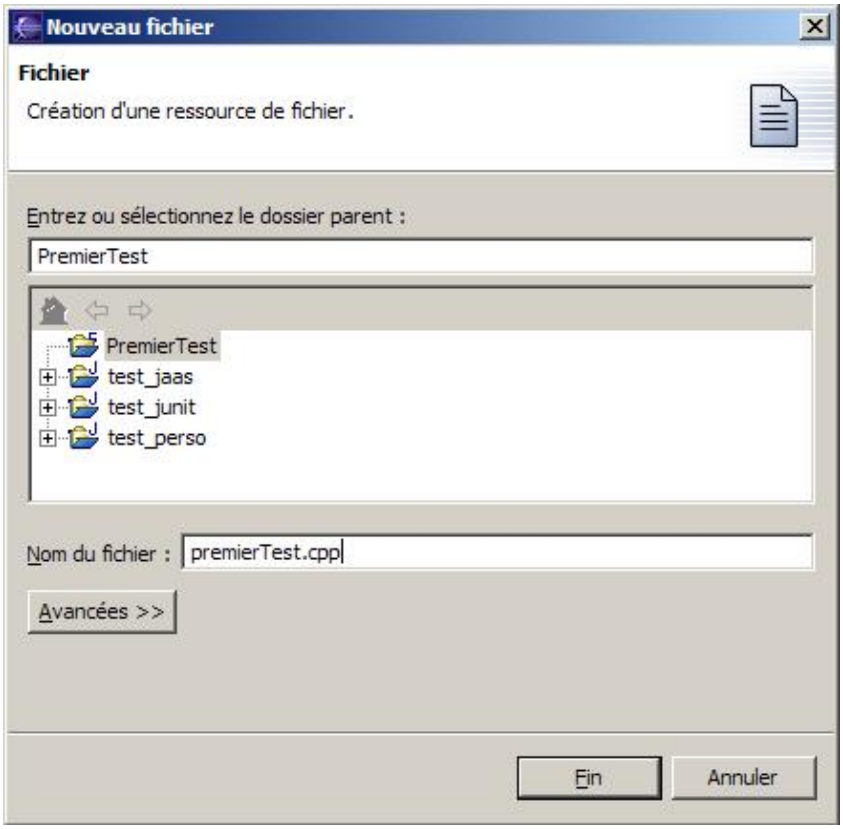
Saisissez le nom du projet et cliquez sur « Suivant ».



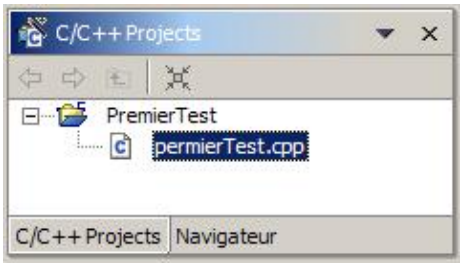
Sous Windows, dans l'onglet « Binary Parser », cliquez sur « PE Windows Parser » puis sur le bouton « Fin ».



Il faut ensuite créez un nouveau fichier dans le projet.



Saisissez le nom du fichier et cliquez sur le bouton « Fin ».

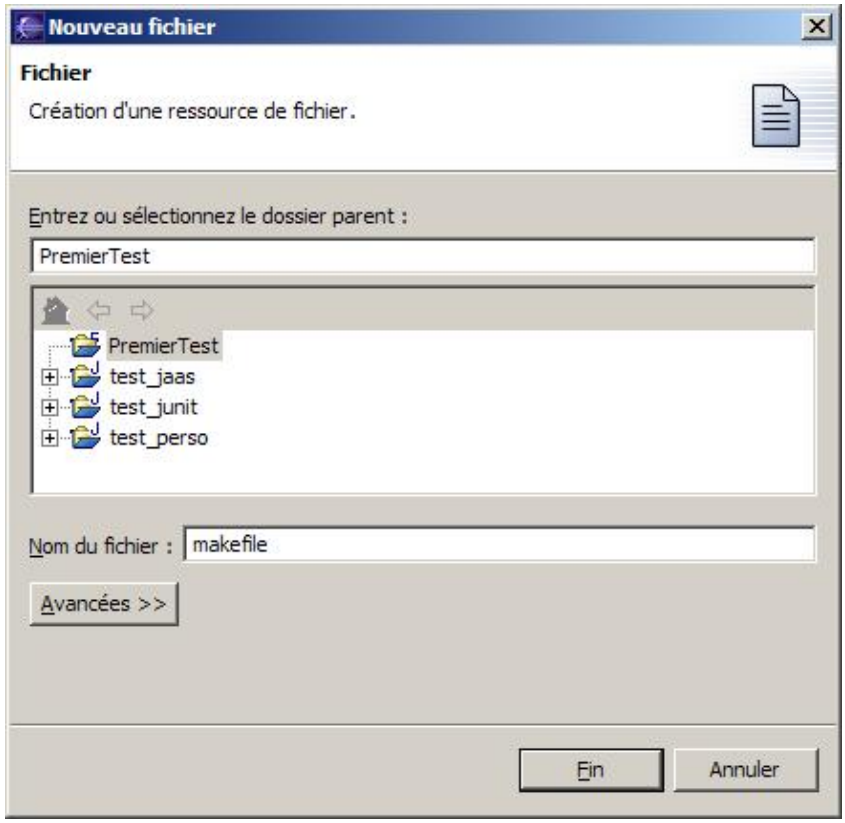


Il suffit de saisir le code de l'application dans l'éditeur :

Exemple :

```
#include <iostream>
using namespace std;
int main () {
  cout << "Bonjour" << endl;
  char input = '';
  cin >> input;
  exit(0);
}
```

Il faut ensuite créer un fichier nommé makefile pour automatiser la génération de l'exécutable grâce à l'outil make.



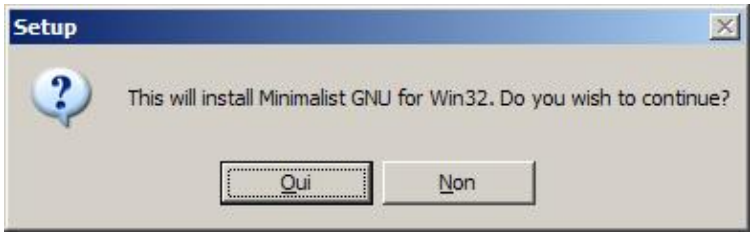
Cliquez sur le bouton « Suivant » puis sur le bouton « Fin ».

22.1.3. Installation de MinGW

Pour pouvoir compiler des fichiers sources avec le CDT, il est nécessaire de disposer d'un compilateur externe. Pour l'instant seul le compilateur GCC du projet GNU est supporté. N'étant pas directement fourni dans Windows, il faut l'installer grace à un outil tiers. Cette section décrit la procédure d'installation avec l'outil open source Minimalist GNU for Win32 (MinGW).

Il faut downloader les fichiers MSYS−1.0.9.exe et MinGW−3.1.0−1.exe sur le site http://www.mingw.org/

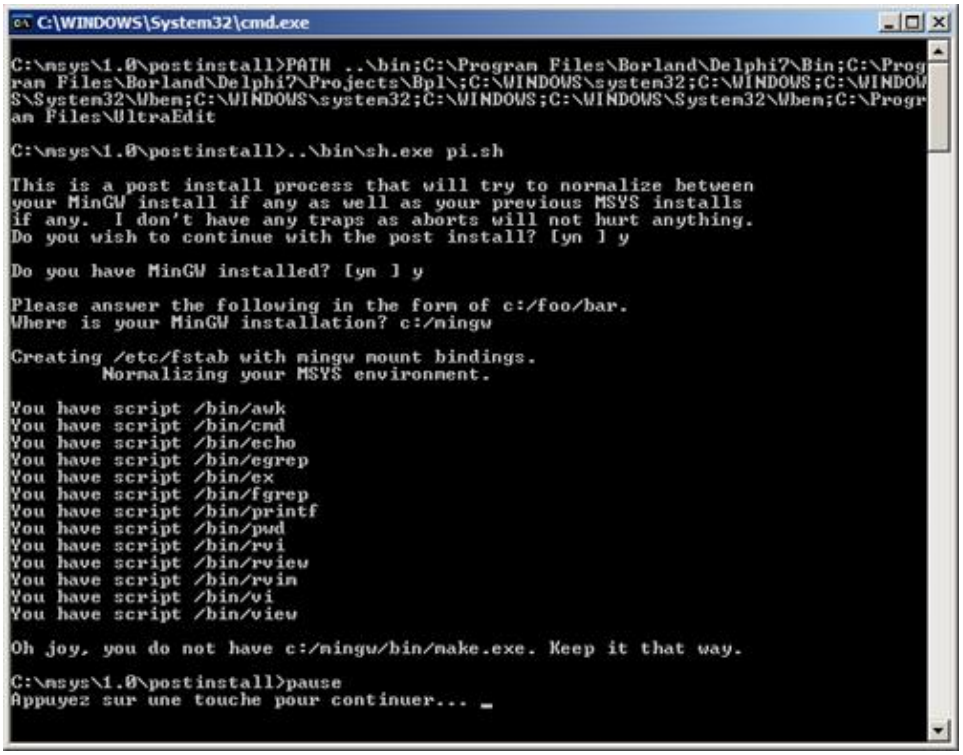
Lancez l'exécution de MinGW−3.1.0−1.exe en premier.



L'assistant d'installation se compose de plusieurs pages :

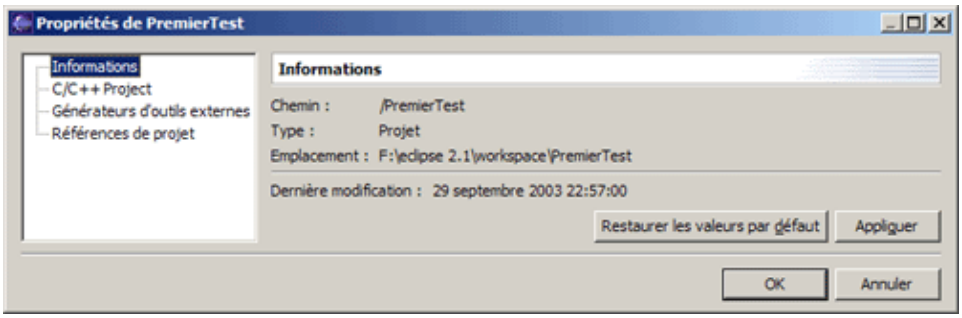
- Sur la page d'accueil, cliquez sur le bouton "Next".
- La page "License Agreement" apparait : lisez la licence et cliquez sur le bouton "Yes" pour accepter les termes de la licence et poursuivre l'installation.
- La page "Information" apparait : lisez le texte et cliquez sur le bouton "Next".
- La page "Select destination directory" apparait : sélectionnez le répertoire où sera installer MinGW et cliquez sur le bouton "Next".
- La page "Ready to install" apparait : cliquez sur le bouton "Install".

Lancez l'exécution de MSYS−1.0.9.exe. Un script permet de configurer l'environnement.

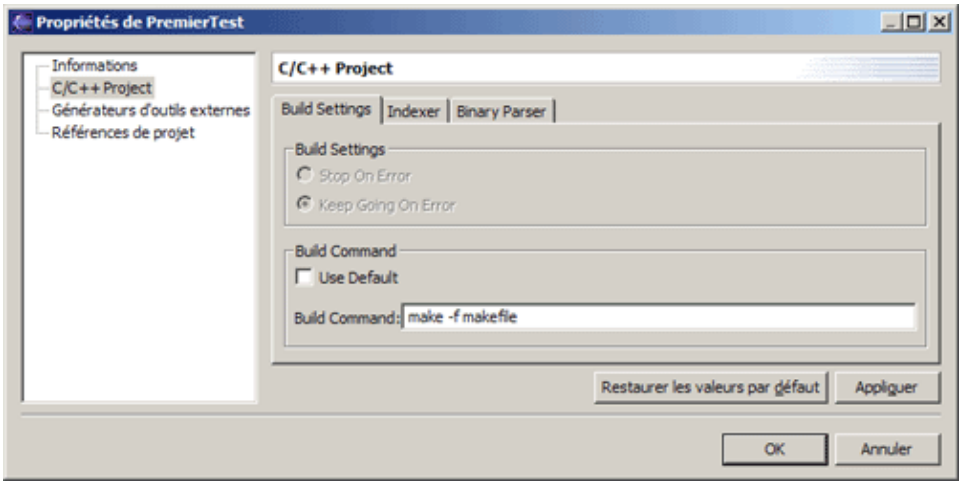


22.1.4. Première configuration et exécution

Il faut modifier les propriétés du projet pour utiliser les options de la commande make fournie avec MSYS.



Il faut décocher l'option "use default" et saisir "make −f makefile" dans le champ "build command".



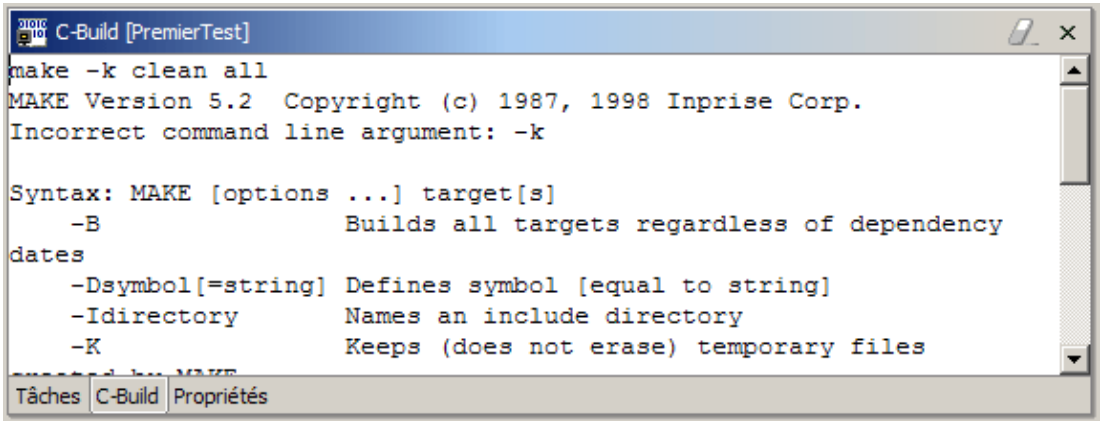
Il faut ajouter les répertoire suivants dans la variable d'environnement path : C:\msys\1.0\bin;C:\MinGW\bin;

Attention : après la modification, il faut relancer Eclipse si celui ci était en cours d'exécution.

Si les outils make et gcc ne sont pas trouvés, alors le message suivant est afficher dans la vue « C−Build »

Exemple :
`Build Error` `(Exec error:Launching failed)`

Attention : il est important que le bon programme makefile soit le premier trouvé dans le classpath. Il est par exemple possible d'avoir des problèmes avec l'outil Delphi de Borland qui ajoute dans le classpath un répertoire qui contient un programme Makefile.



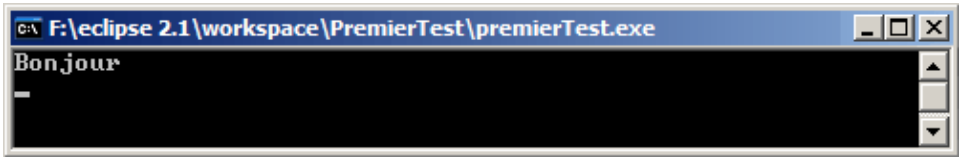
Dans ce cas, il suffit de déplacer ou de supprimer la référence sur le répertoire C:\Program Files\Borland\Delphi7\Bin; dans la variable classpath. (il faut le remettre en cas de suppression pour une utilisation correcte de Delphi).

Si la génération se passe bien, le vue C−Build affiche les étapes de le génération.

Exemple :

```
make −f makefile
gcc −c premierTest.cpp
gcc −o premierTest premierTest.o −L C:/MinGW/lib/gcc−lib/mingw32/3.2.3/ −lstdc++
```

Lancer l'exécution, il suffit de sélectionner le fichier premierTest.exe et d'utiliser l'option « Ouvrir » du menu contextuel.



22.1.5. Utilisation du CDT

L'environnement de développement proposé par le CDT étant un cours de développement, il n'est pas encore aussi complet que l'environnement proposé par le JDT pour Java.

La version 1.1. du CDT propose cependant les fonctionnalités suivantes :

- coloration syntaxique dans l'éditeur de code
- navigation dans un code source grâce à la vue outline
- l'éditeur possède un assistant de code utilisant des modèles (templates)
- utilisation de l'historique locale pour retrouver en local un certain nombre de version locale du code
- ...

23. EclipseUML

La société Omondo propose EclipseUML qui est un plug−in permettant d'utiliser UML dans Eclipse.

EclipseUML propose une version gratuite avec le support des 11 diagrammes d'UML 2.0.

	Version utilisée dans cette section
Eclipse	3.0.1
J2SE	1.4.2_03
EclipseUML	2.0.0

EclipseUML nécessite plusieurs plug−ins pour son fonctionnement avec Eclipse 3.0 :

- GEF (Graphical Editor Framework) 3.0.1
- EMF (Eclipse Modeling Framework) 2.0.1
- UML2 1.0.1

23.1. Installation

Il faut télécharger le fichier eclipseUML_E301_freeEdition_2.0.0.beta.20041026.jar sur le site d'Omondo : http://www.omondo.com/download/free/index.html

Ce fichier .jar contient un exécutable qui va installer EclipseUML et les différents plug−ins tiers nécessaires (GEF, EMF et UML2).

Si Eclipse est en cours d'exécution, il faut le quitter avant de lancer l'installation.

Pour lancer l'installation, il suffit de lancer la commande ci dessous dans une console DOS

java −jar eclipseUML_E301_freeEdition_2.0.0.beta.20041026.jar

Un assistant guide l'utilisateur dans les différentes étapes de l'installation :

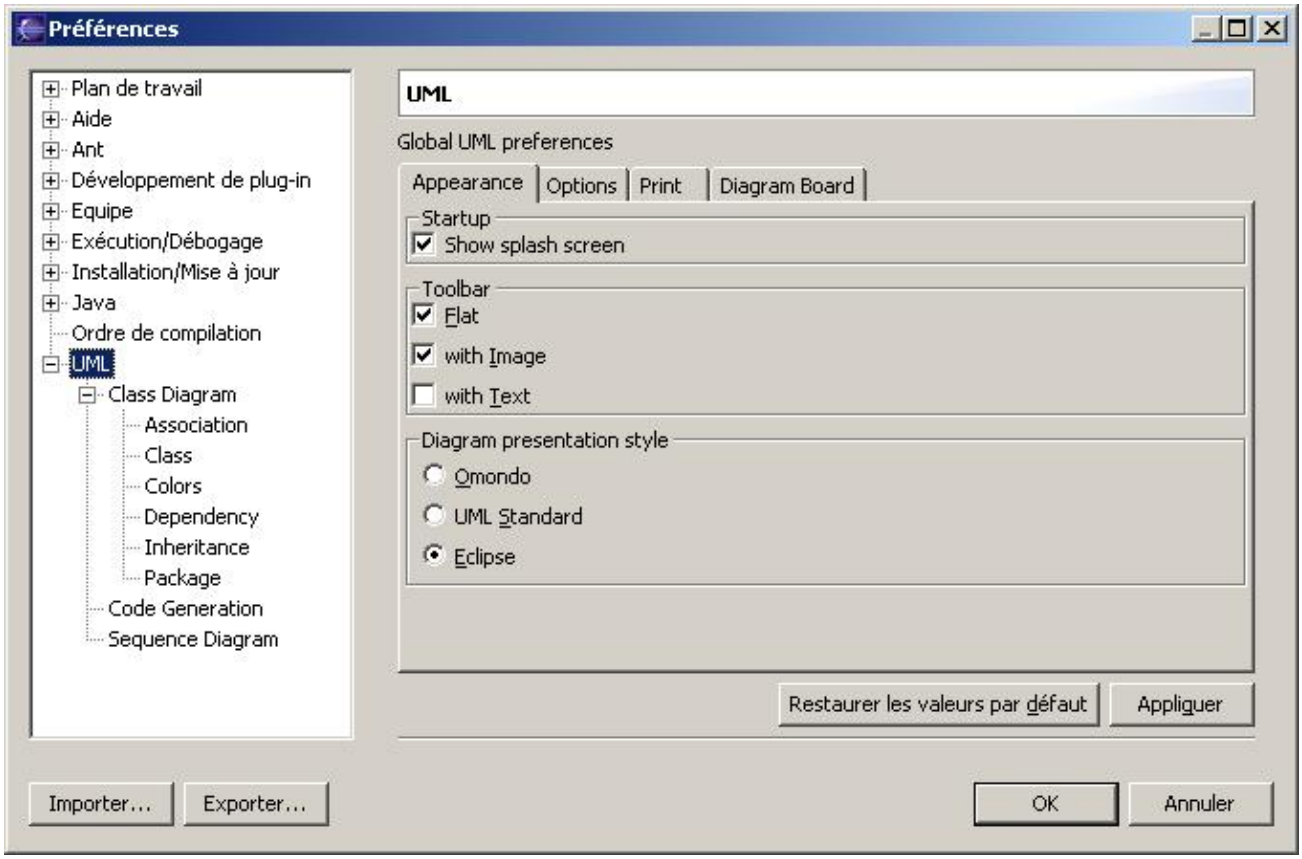
- l'écran d'accueil permet de sélectionner la langue de l'assistant : cliquez sur le bouton « Ok »
- l'écran « Introduction » vous souhaite la bienvenue : cliquez sur le bouton « Suivant »
- l'écran « Informations » affiche des informations à lire : cliquez sur le bouton « Suivant »
- l'écran « License » affiche la licence d'utilisation : lisez la licence et si vous l'acceptez, cliquez sur le bouton « J'accepte les termes de cet accord de licence » puis sur le bouton « Suivant »
- l'écran « Configuration » permet de sélectionner le répertoire d'installation d'Eclipse. Sélectionnez le répertoire d'installation d'Eclipse au besoin et cliquez sur le bouton « Suivant ».

- l'écran « Configuration » suivant permet de sélectionner les packages à installer : cliquez sur le bouton « Suivant »
- l'écran « Installation » affiche la progression de la copie des fichiers : cliquez sur le bouton « Suivant »
- l'écran « Install Complete » : cliquez sur le bouton « Quitter ».

Une fois l'installation terminée, il suffit de lancer Eclipse.

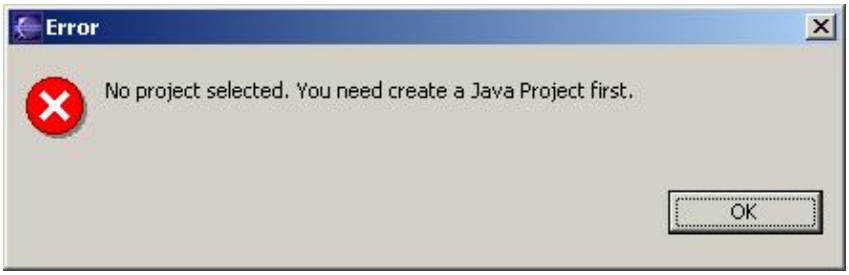
23.2. Les préférences

Les nombreux paramètres d'EclipseUML peuvent être réglés dans les préférences (menu Fenêtre/Préférences)

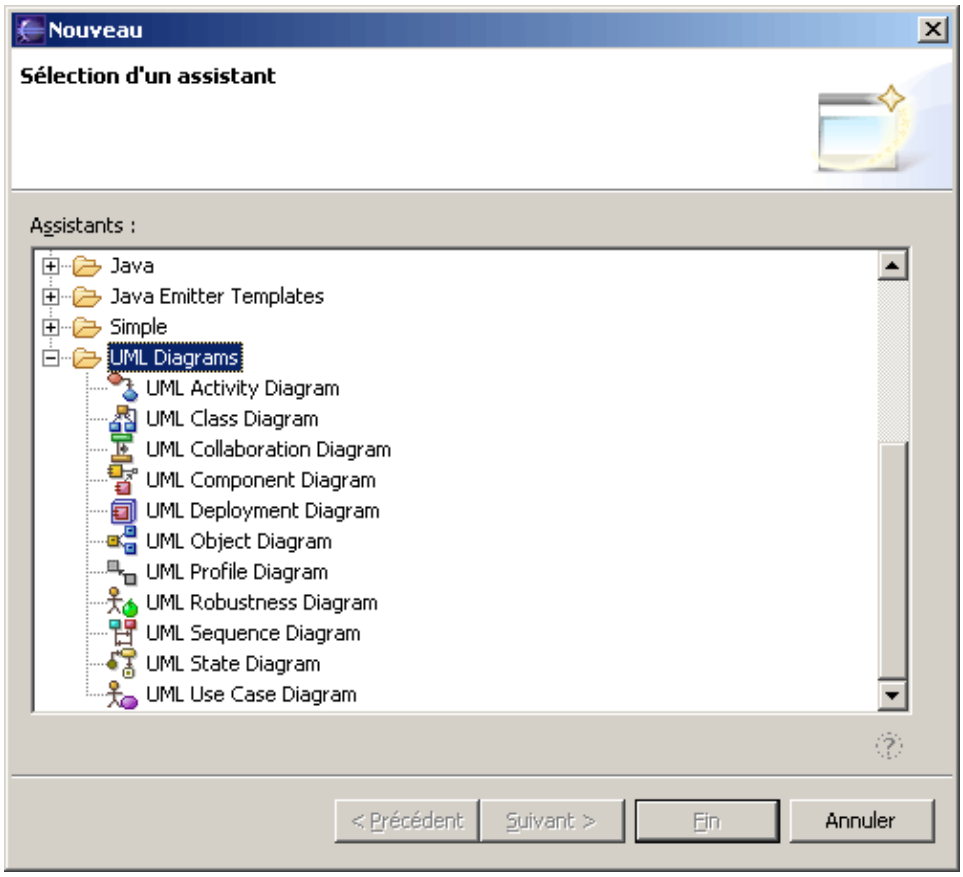


23.3. Mise en oeuvre du plug−in

L'utilisation des diagrammes UML nécessitent d'utiliser un projet Java. Si aucun projet Java n'est sélectionné, avant l'appel à l'assistant de création d'entités, ce dernier affichera un message d'erreur lors de son utilisation.



Pour créer un nouveau diagramme, il faut sélectionner un projet et utiliser l'option « Nouveau / Autres » du menu « Fichier ».

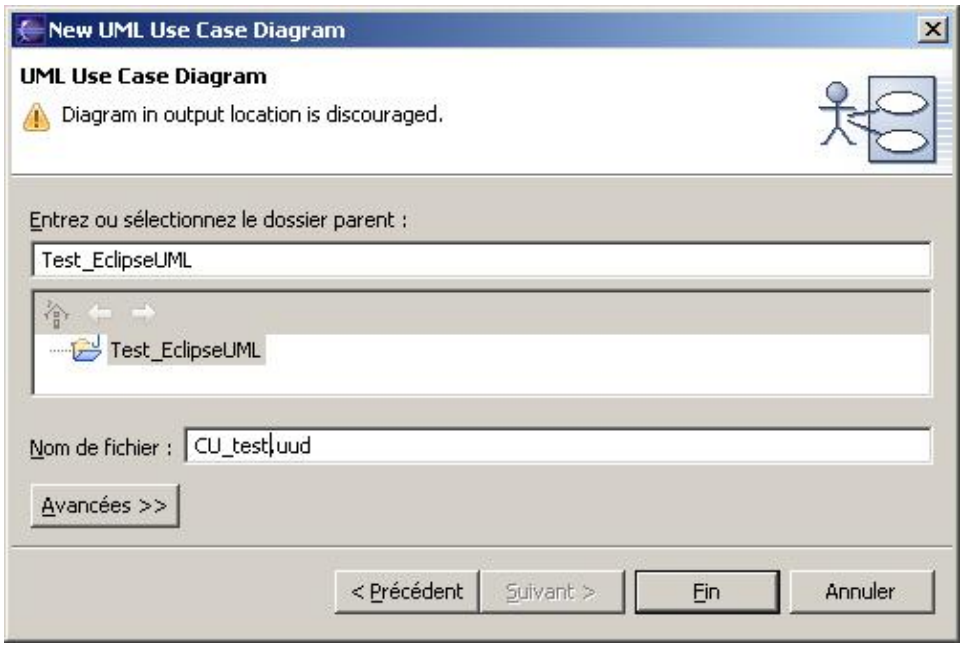


Il est possible d'exporter chaque diagramme dans plusieurs formats en utilisant l'option « Export » de l'éditeur de diagrammes. Les formats utilisables sont : SVG, JPEG et Gif.

Chacun des éditeurs proposent des fonctionnalités de zoom via plusieurs options du menu contextuel.

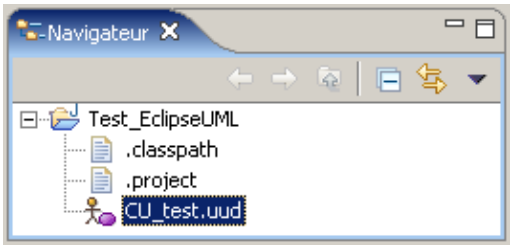
23.3.1. Création d'un diagramme de cas d'utilisation

Il faut créer une nouvelle entité de type « UML Diagrams / UML Use Case Diagram» et cliquer sur le bouton « Suivant »

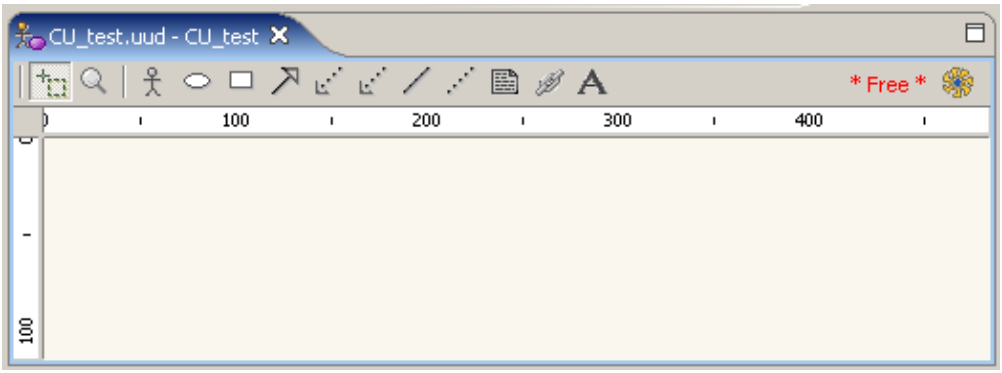


Saisissez le nom du fichier et cliquez sur le bouton « Fin ».

Le fichier est créé par l'assistant



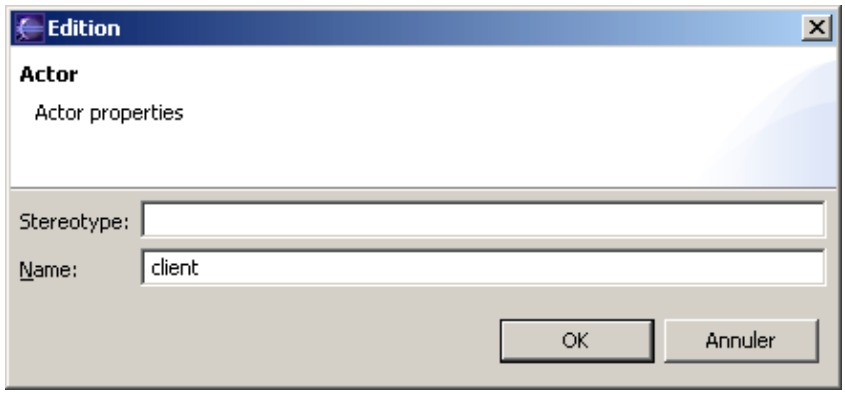
et une vue dédiée à l'édition du diagramme ouvre le fichier



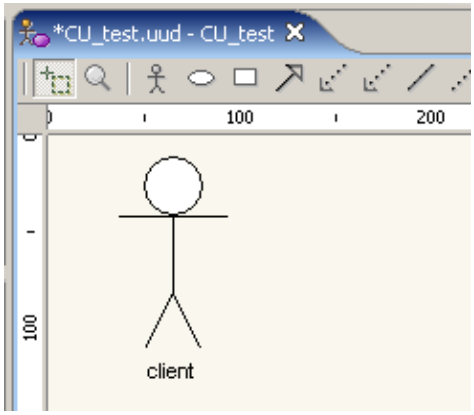
Il suffit alors de composer son diagramme.

Pour ajouter un acteur, il faut cliquer sur le bouton  puis cliquer sur la surface de travail de la vue.

Une boite de dialogue permet de saisir le stéréotype et le nom du nouvel acteur.



Le diagramme est enrichi avec le nouvel acteur.

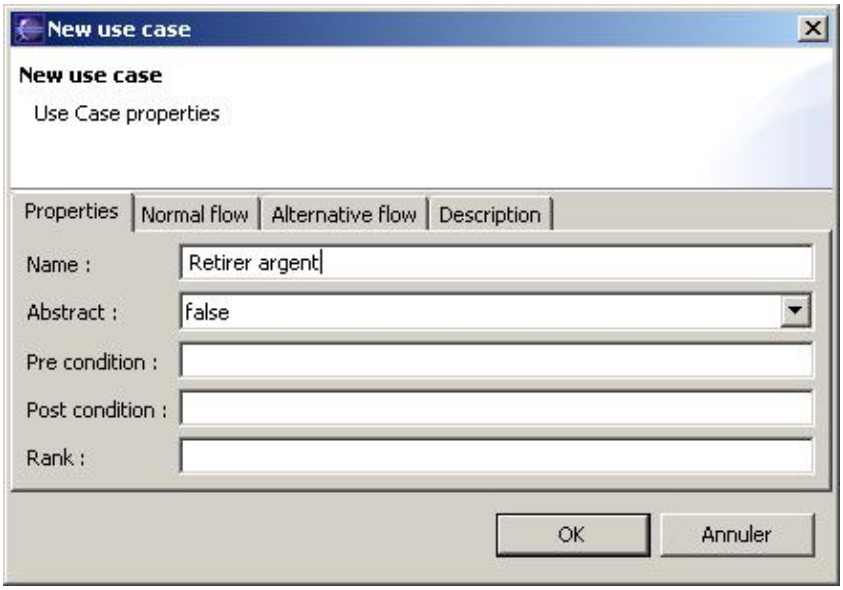


Il est possible de répéter l'opération d'ajout d'un nouvel acteur en cliquant sur la surface tant que le bouton

 est activé.

La création d'un cas d'utilisation est similaire en utilisant le bouton  .

Une boîte de dialogue permet de saisir les informations concernant le nouveau cas d'utilisation.

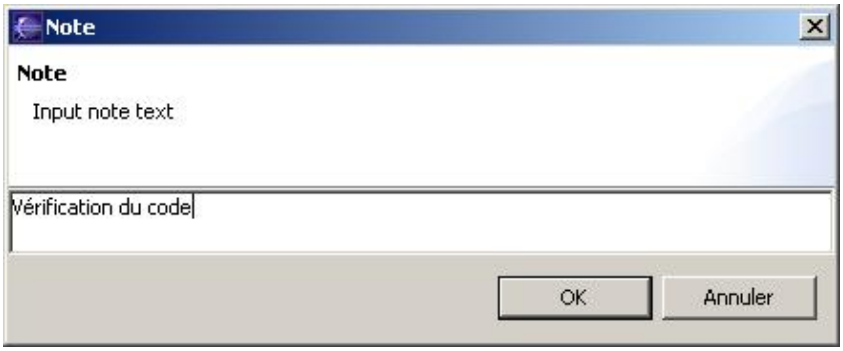


Il faut saisir les informations et cliquer sur le bouton « OK ».

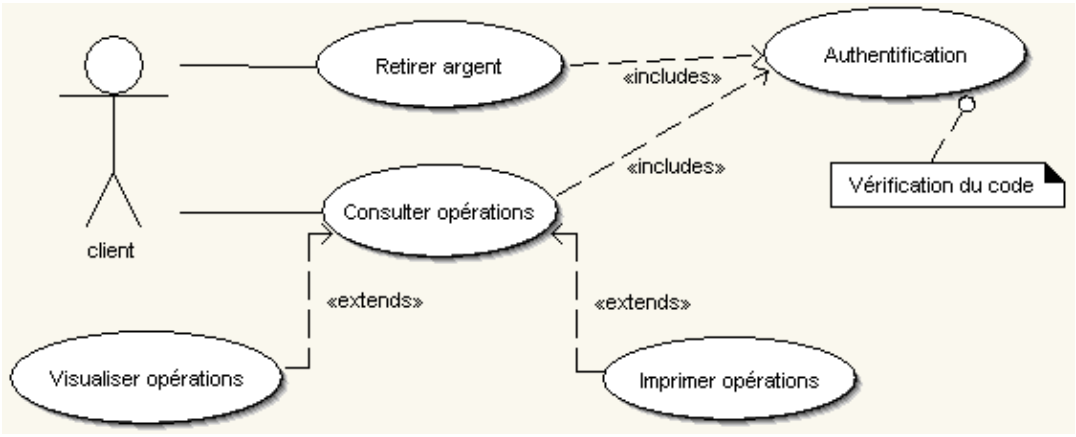
Pour ajouter des associations, il suffit de cliquer sur l'association désirée dans la barre d'outils, puis de cliquer sur la première entité (celle ci change de couleur au passage de la souris) puis sur la seconde (celle ci change aussi de couleur au passage de la souris).

Pour ajouter un commentaire, il suffit de cliquer sur le bouton  .

Une boite de dialogue permet de saisir le commentaire.

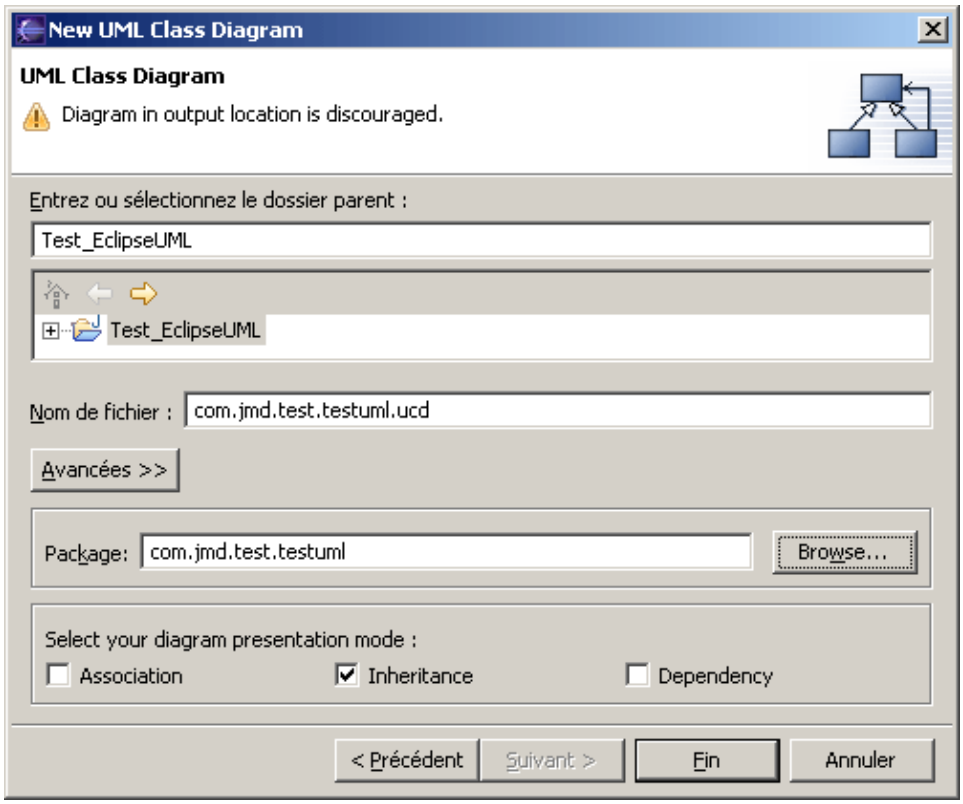


Voici un exemple de diagramme.

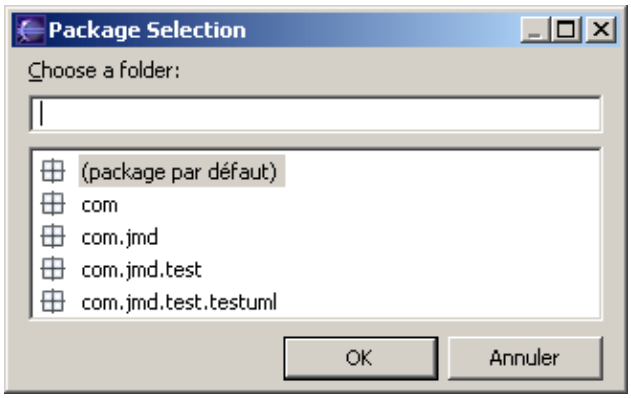


23.3.2. Création d'un diagramme de classe

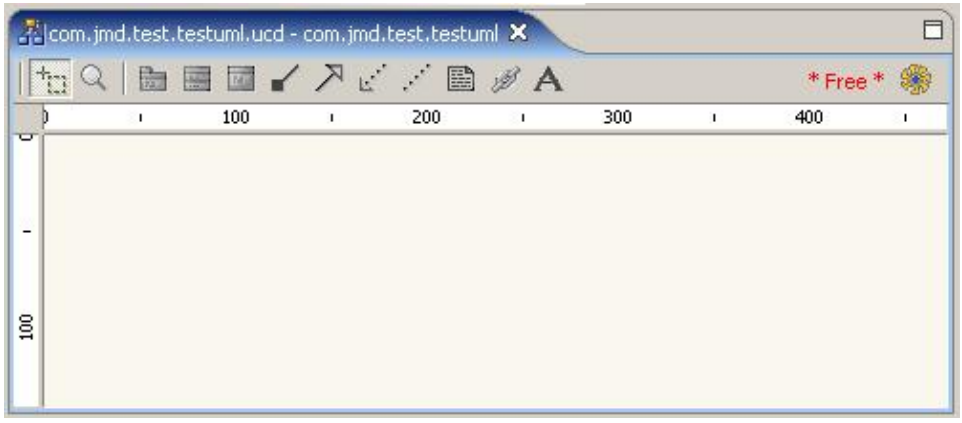
Pour créer un diagramme de classe, il faut sélectionner un package et créer une nouvelle entité de type « UML Diagrams/UML Class Diagram ».



Le bouton « Browse » permet de sélectionner le package concerné

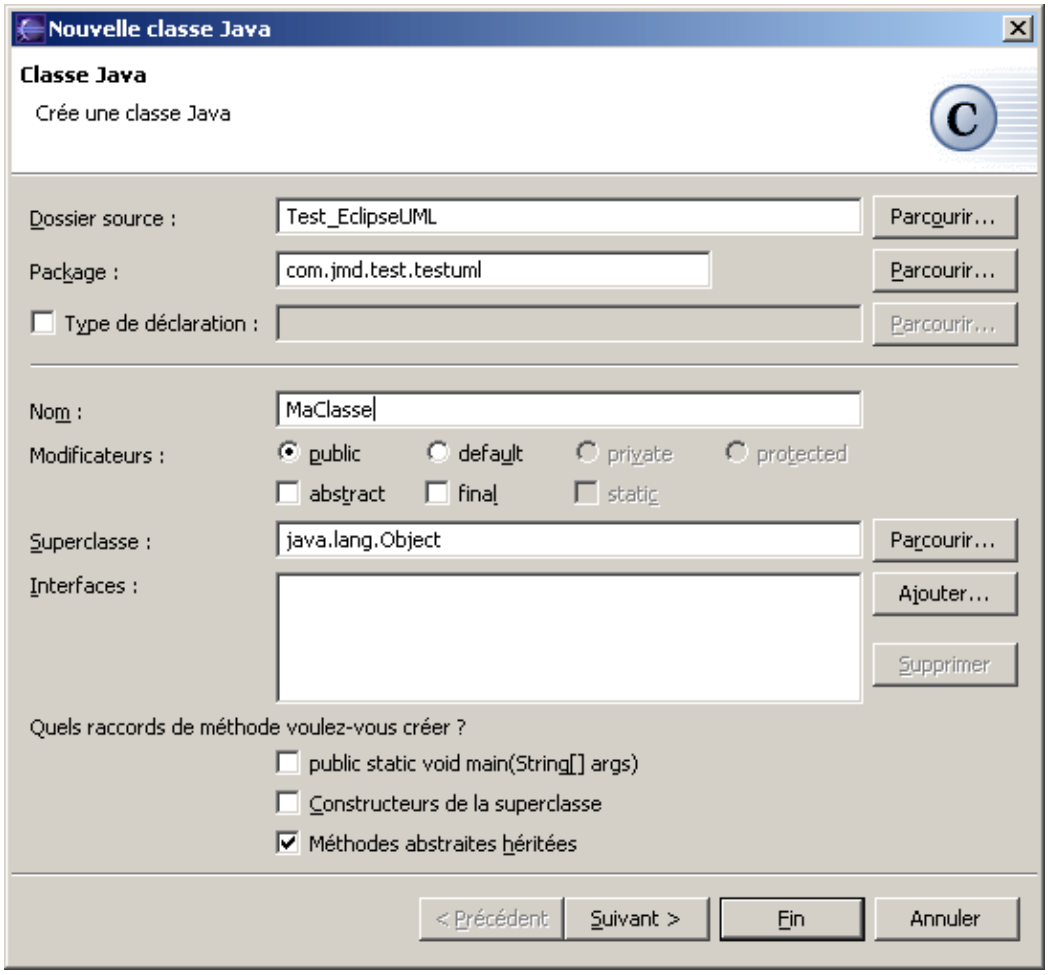


Saisissez le nom du fichier qui sa contenir le diagramme et cliquez sur le bouton « Fin » pour créer le fichier et ouvrir l'éditeur

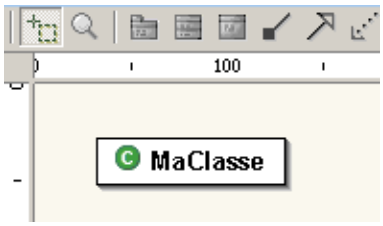


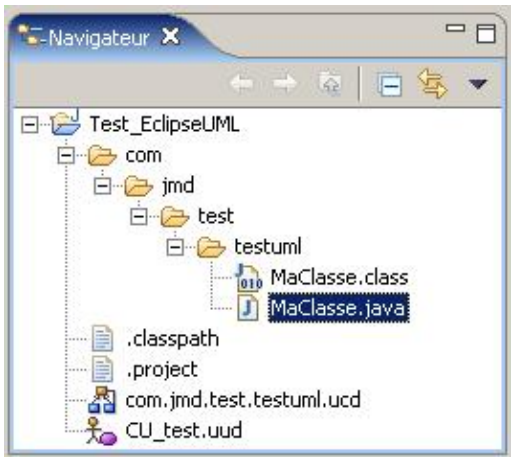
Pour créer une nouvelle classe, il suffit de cliquer sur le bouton  puis de cliquer sur la surface de travail de l'éditeur.

L'assistant de création de classe s'ouvre.



Le bouton suivant permet d'ajouter des stéréotypes. Une fois la saisie terminée, il suffit de cliquer sur le bouton « Fin ». La nouvelle classe est alors ajoutée dans le diagramme mais aussi dans l'espace de travail.

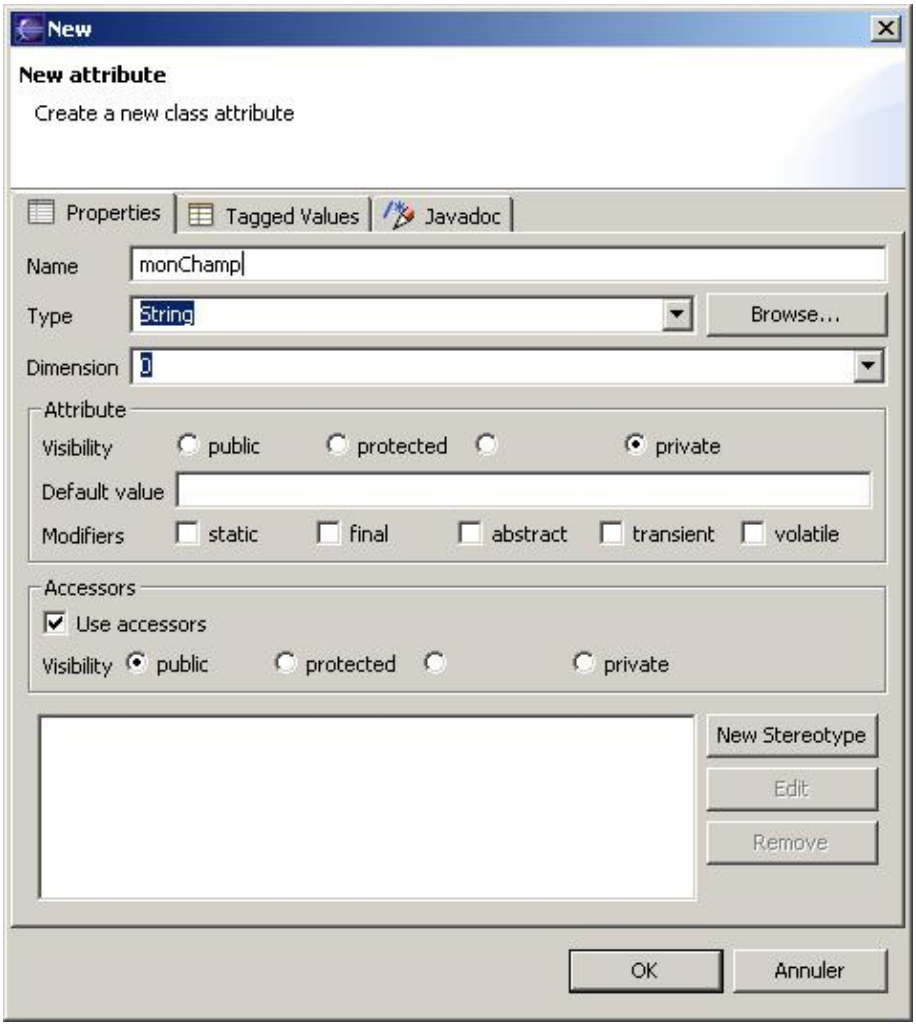




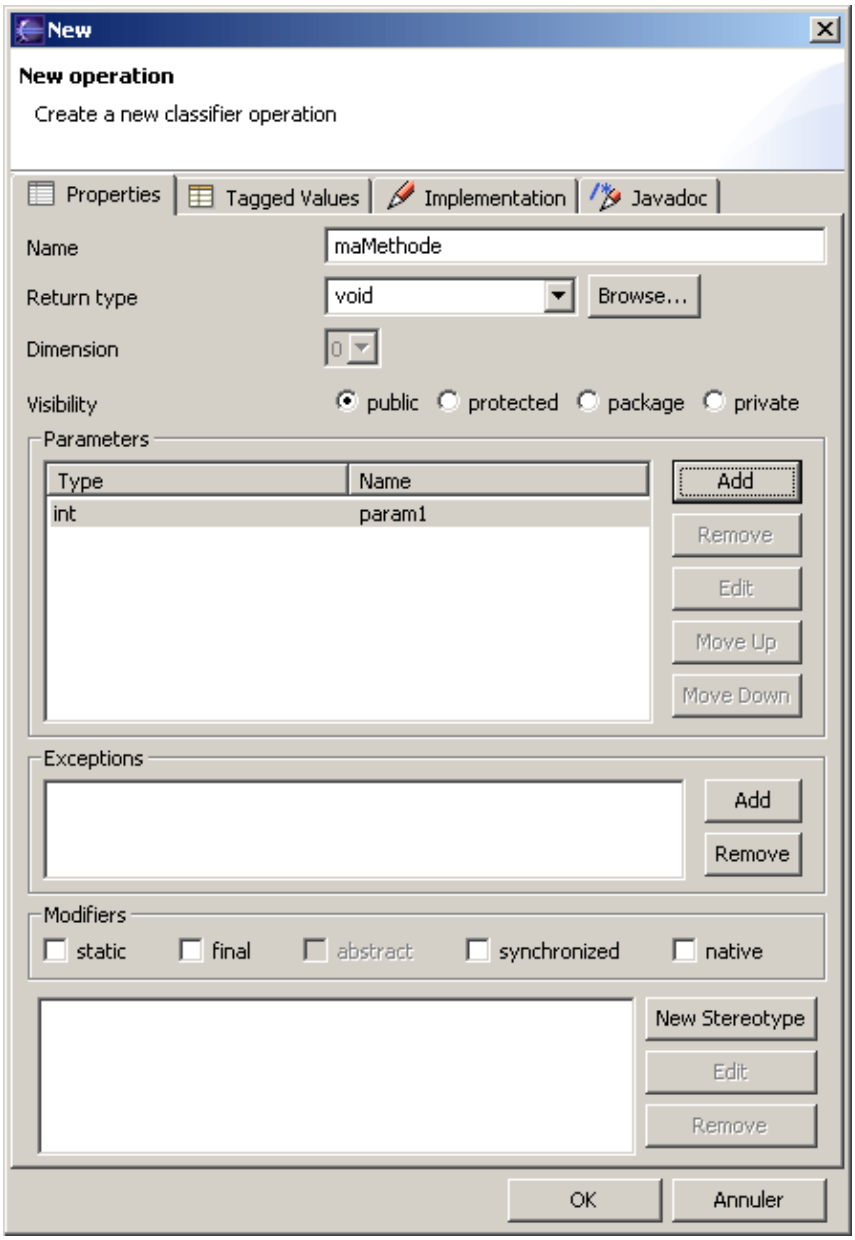
L'option « New » du menu contextuel de la classe dans l'éditeur permet d'ajouter des membres à la classe.

Une boite de dialogue permet alors la saisie des informations sur le nouveau membre.

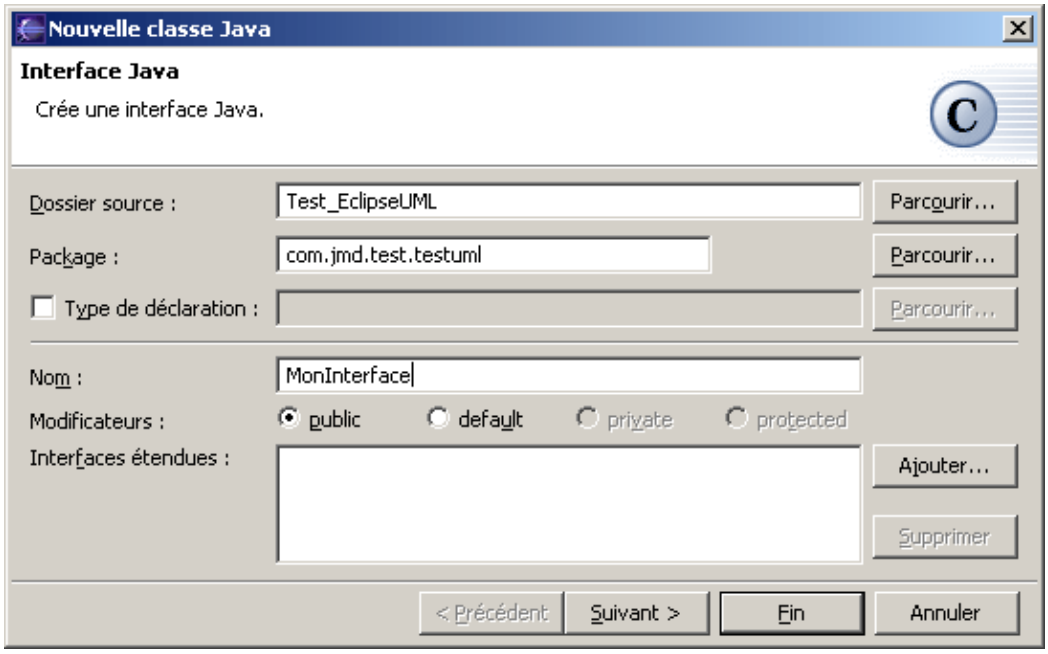
Par exemple pour ajouter un nouveau champ.



Par exemple, pour ajouter une nouvelle méthode

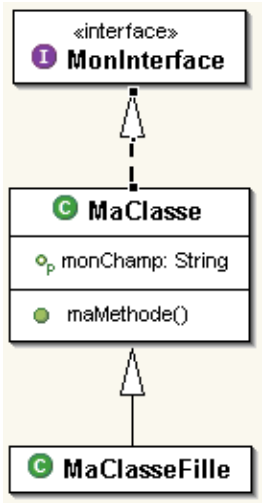


Pour ajouter une nouvelle interface, il suffit de cliquer sur le bouton  puis sur la surface de travail de l'éditeur. Une boîte de dialogue permet de saisir les informations de la nouvelle interface.



La création d'une association de généralisation entre deux classes/interfaces se fait en cliquant sur le bouton  , puis sur la classe fille et enfin sur la classe mère.

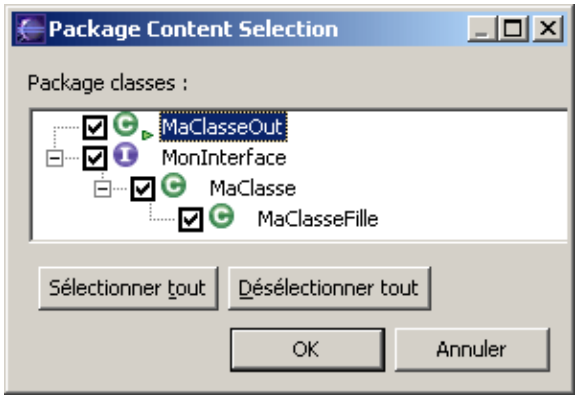
La relation générée (héritage ou implémentation) se fait automatiquement en fonction du type des deux entités concernées.



Pour mettre à jour le code source en fonction des relations ajoutés, il faut utiliser l'option « Model synchronize » du menu contextuel de la classe ou interface dans l'éditeur.

Les mises à jour sont bi−directionnelles : dans le code et sur le diagramme. Il faut donc faire attention notamment à la suppression d'une classe du diagramme qui la supprime aussi dans le package. Si le besoin est simplement de ne plus afficher la classe dans le diagramme, il faut simplement la masquer en utilisant l'option « Hide » du menu contextuel.

Pour faire réapparaître un élément masqué, il faut utiliser l'option « Package elements ... »



Il suffit de sélectionner l'élément et de cliquer sur le bouton « OK ».

24. Eclipse et les bases de données

Eclipse ne propose pas par défaut de plug−in capable de gérer ou de manipuler une base de données, mais il existe de nombreux plug−in permettant d'accomplir certaines de ces tâches. Ce chapitre va présenter quelques uns d'entres eux :

- Quantum
- JFaceDBC
- DBEdit
- Clay Database Modelling
- Hybernator

24.1. Quantum

Quantum est un plug−in qui permet l'exécution de requêtes sur une base de données. Il affiche une vue arborescente des bases de données et propose plusieurs assistants pour effectuer des mises à jour sur les données.

Les bases de données supportées sont : Postgres, MySQL, Adabas−D, Oracle, DB2, DB2 for AS400.

Le site officiel est http://quantum.sourceforge.net/

	Version utilisée dans cette section
Eclipse	2.1.2
J2RE	1.4.2_02
Quantum	2.2.2.

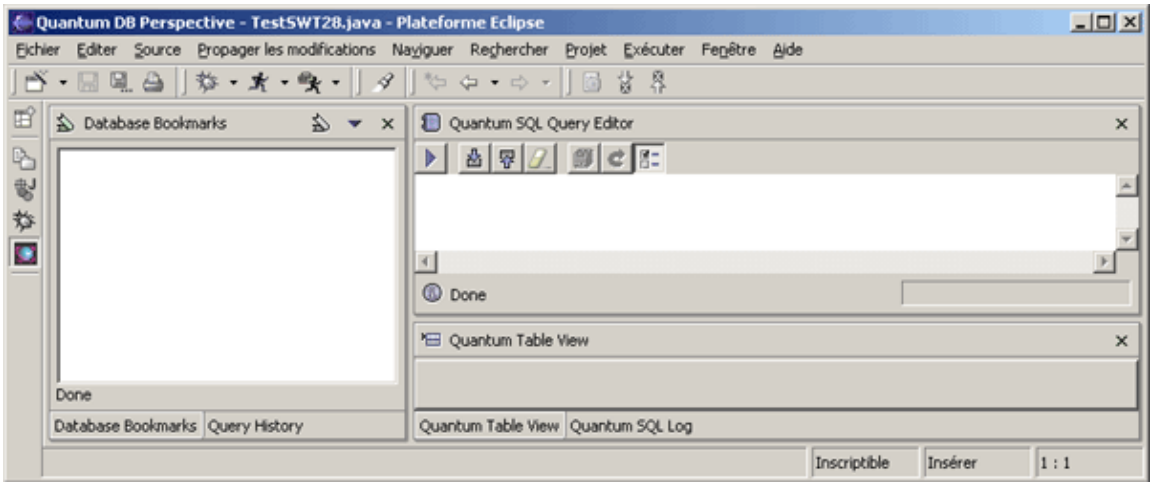
La version utilisée dans cette section nécessite obligatoirement un JDK 1.4 pour fonctionner.

24.1.1. Installation et configuration

Après voir téléchargé le fichier quantum222.zip, il faut le décompresser dans le répertoire qui contient le répertoire d'Eclipse (attention : le chemin des fichiers de l'archive contient déjà le répertoire Eclipse).

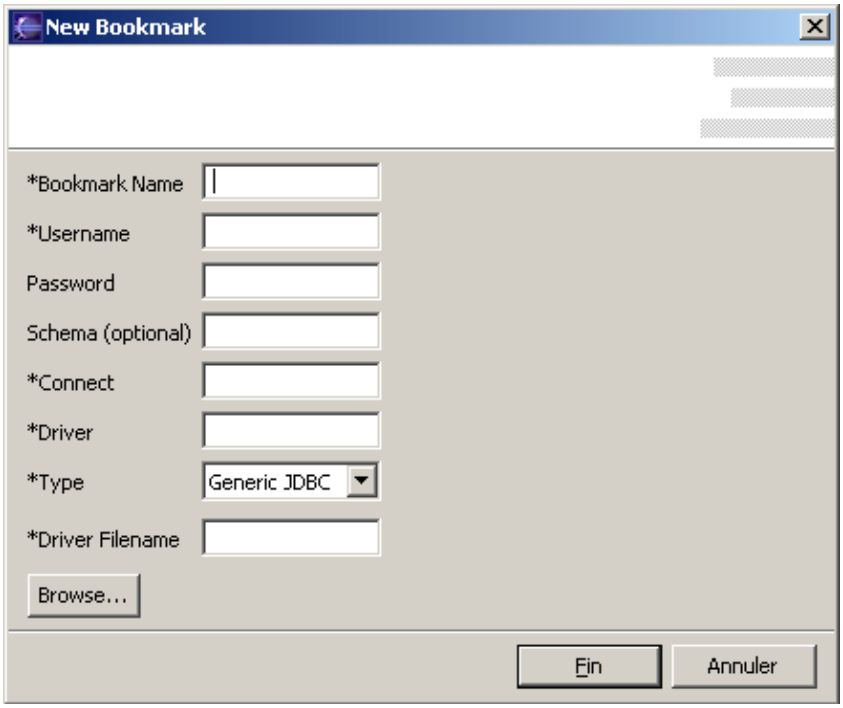
Pour utiliser le plug−in, il faut ouvrir la perspective nommée "Quantum DB".

Quantum propose plusieurs vues :



Pour pouvoir utiliser une base de données, il faut l'enregistrer dans le bookmark. Pour cela, dans la vue « Database Bookmarks », cliquez sur le bouton  ou sélectionnez l'option « New Bookmark » du menu contextuel.

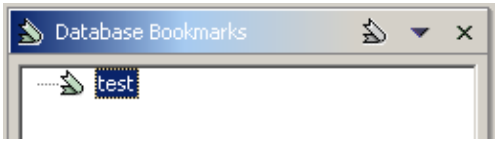
Une boite de dialogue permet de saisir les informations concernant la base de données.



L'exemple ci dessous contient les paramètres pour une base de données MySQL nommée "Test" avec le pilote MySQL Connector−J 3.0 (http://www.mysql.com/products/connector−j/).

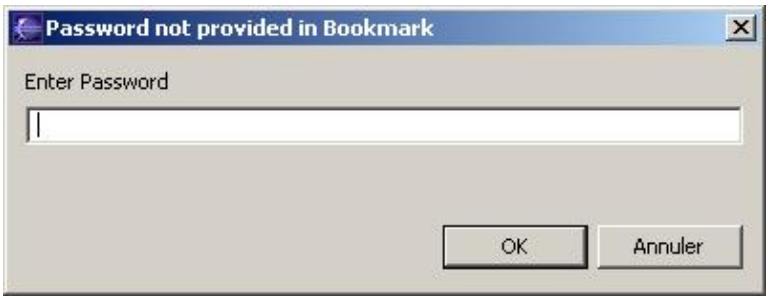
- « Bookmark name » : est le nom du bookmark (cette zone est libre)
- « Username » : est le nom de l'utilisateur pour la connexion à la base de données
- « Password » : est le mot de passe de cet utilisateur
- « Connect » : est la chaîne de connexion dépendante du pilote utilisé (jdbc:mysql://localhost/test dans l'exemple)
- « Driver » : est le nom de la classe du pilote JDBC (com.mysql.jdbc.Driver dans l'exemple)
- « Type : » : est le type de base de données utilisée
- « Driver Name » : et le fichier jar qui contient le pilote JDBC (D:\java\mysql−connector−java−3.0.11−stable\mysql−connector−java−3.0.11−stable−bin.jar dans l'exemple)

Un fois tous les paramètres requis saisis, il suffit de cliquer sur le bouton « Fin ».

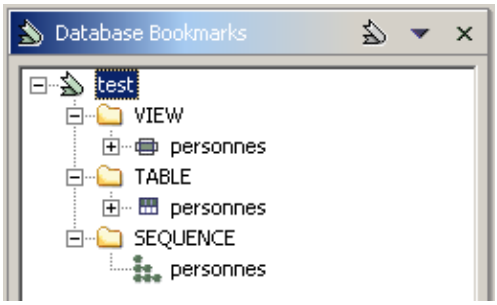


Le bookmark apparait dans la liste. Le menu contextuel propose alors plusieurs options permettant de modifier ou de supprimer le bookmark. L'option "Connect" permet de demander la connexion à la base de données.

Aucun mot de passe n'ayant été fourni, une boite de dialogue demande le mot de passe.



Si il n'y a aucun mot de passe, il suffit de cliquer sur le bouton "OK".

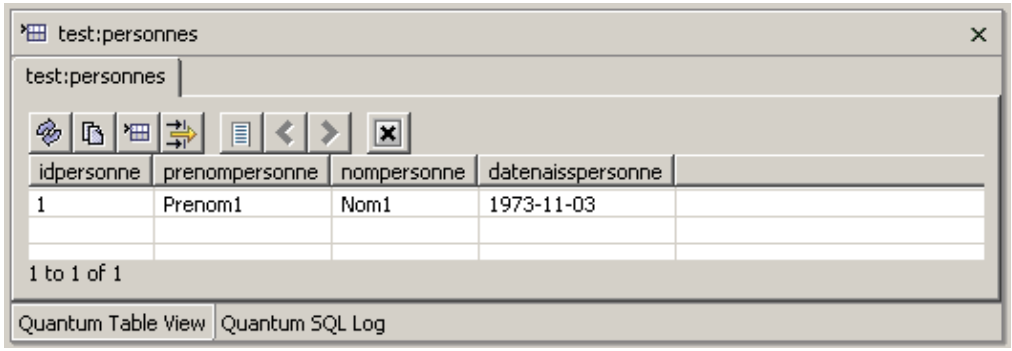


La vue "Database Bookmarks" affiche l'arborescence de la base de données. Dans l'exemple, la base de données ne contient qu'une seule table nommée "personnes".

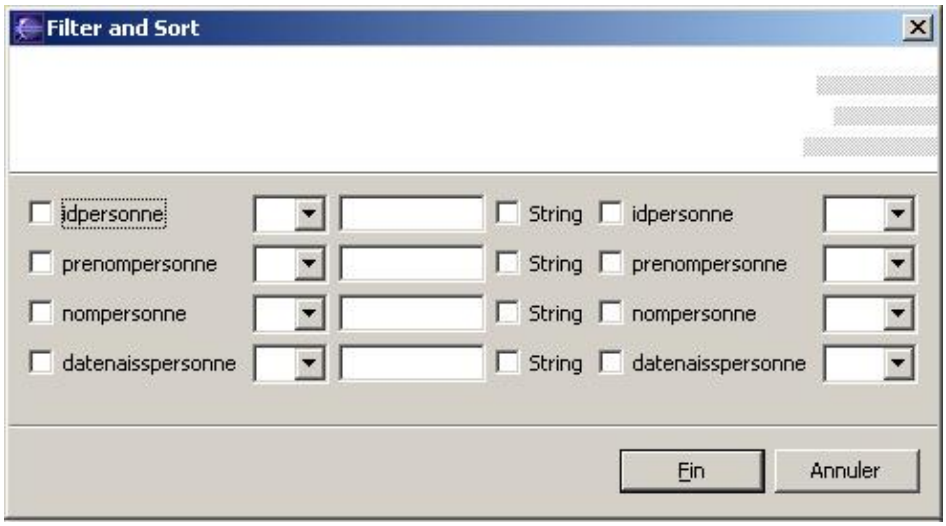
24.1.2. Afficher le contenu d'une table

Pour afficher le contenu d'une table, il suffit de double cliquer sur une table dans la vue « Data Bookmarks ».

La vue « Quantum Table View » affiche alors le contenu de la table.



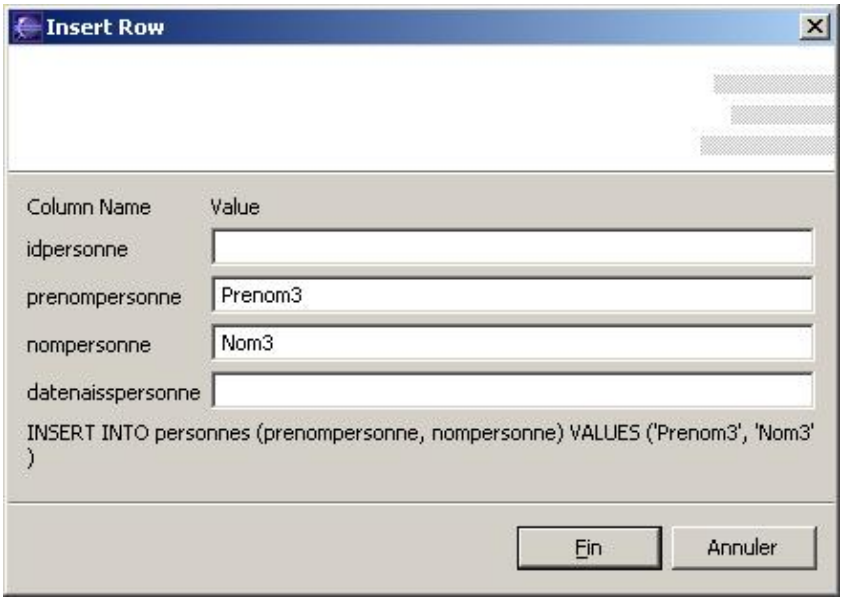
Cette vue permet de trier et de filtrer le contenu de la table en cliquant que le bouton 



Le filtre est défini dans la partie de gauche et le tri dans la partie de droite de la boîte de dialogue.

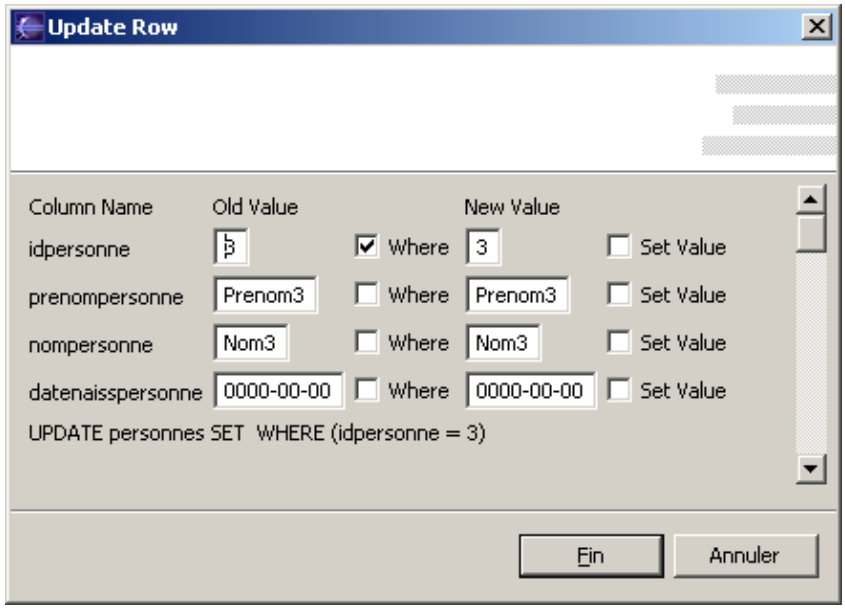
Le menu contextuel de la vue « Quantum Table View » propose des options pour faire des mises à jour des données de la table : insertion, modification et suppression.

Un assistant permet de saisir les données à ajouter dans la nouvelle occurrence de la table :



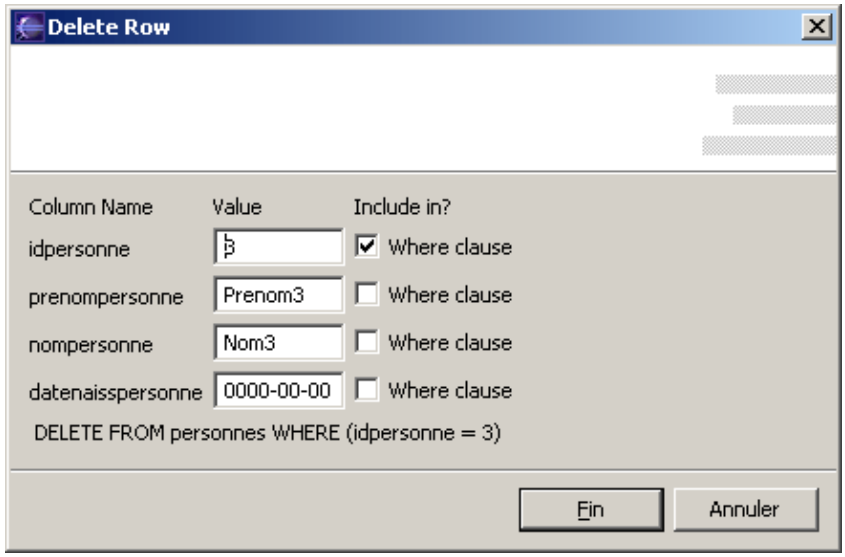
Une fois les données saisies, il suffit de cliquer sur le bouton « Fin » pour que la requête SQL générée soit exécutée. Pour voir les modifications dans la vue « Quantum Table View », il faut cependant explicitement demander le rafraichissement en cliquant sur le bouton  .

Un assistant permet de modifier les données d'une occurrence :



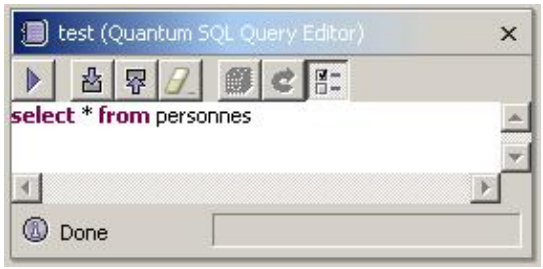
Le principe de la construction dynamique de la requête SQL est identique à celui de l'insertion d'une occurrence hormis le fait qu'il est possible de saisir les informations concernant la portée de la requête de mise à jour (clause Where).

L'assistant permettant la suppression d'une ou plusieurs occurrences s'utilise de la même façon :



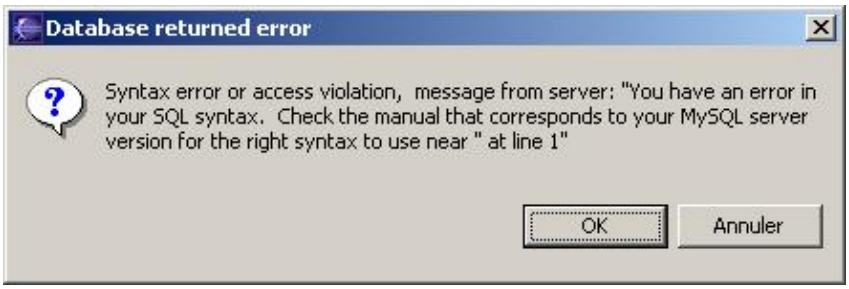
24.1.3. Exécution d'une requête

L'éditeur « Quantum SQL Query Editor » permer de saisir des requêtes SQL avec une coloration syntaxique.



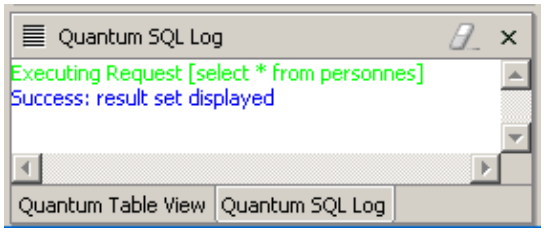
Le bouton  permet d'exécuter la requête.

Si la requête contient un erreur, alors un message d'erreur est affiché :

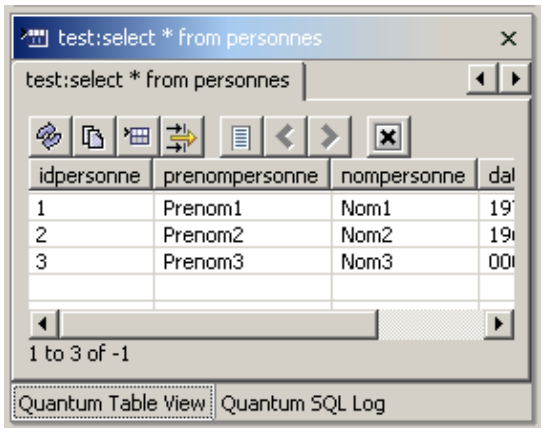


Un rond rouge en bas de la vue signale aussi une erreur lors de la dernière exécution.

La vue « Quantum SQL Log » affiche des informations sur l'exécution des requêtes.



Dans le cas d'une requête d'interrogation, le résultat est affiché dans la vue « Quantum Table View ».



Les boutons  et  permettent respectivement d'importer et d'exporter le contenu de l'éditeur dans un fichier possédant l'extension .sql par défaut.

La vue permet aussi de gérer les transactions. Le bouton  permet de préciser si le mode fonctionnement est auto−commit (bouton enfoncé) ou manuel. Dans le mode manuel, le bouton  permet de faire un commit des opérations non encore validées et le bouton  permet d'invalider les opérations non encore validées (rollback).

La vue « Query History » contient un historique des requêtes exécutées : un double clic sur une de ces requêtes permet de réafficher son contenu dans la vue « Quantum SQL Query ».

24.2. JFaceDbc

JFaceDBC est un plug−in qui propose de faciliter la manipulation d'une base de données. Ce plug−in possède quelques fonctionnalités intéressantes comme un système de gestion des connexions avec chargement dynamique des pilotes, l'affichage sous la forme graphique du schéma de la base de données ou la création pour une table des fichiers xml et java pour l'outil de mapping Hibernate.

Le site officiel est à l'url http://jfacedbc.sourceforge.net/

	Version utilisée dans cette section
Eclipse	2.1.2
J2RE	1.4.2_02
JFaceDBC	2.2.1.

La version utilisée dans cette section nécessite un JDK 1.4 pour fonctionner.

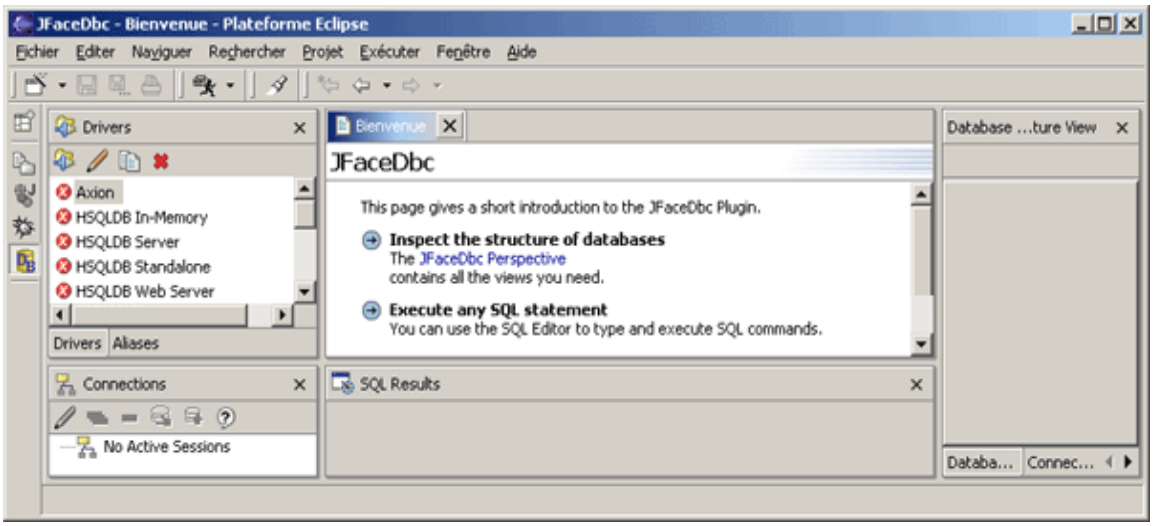
24.2.1. Installation et configuration

Après voir téléchargé le fichier net.sf.jfacedbc_2.2.1.zip sur le site de JFaceDBC, il faut le décompresser dans le répertoire qui contient Eclipse. A l'exécution suivante, il est nécessaire de valider les modifications en attente et de relancer l'application.

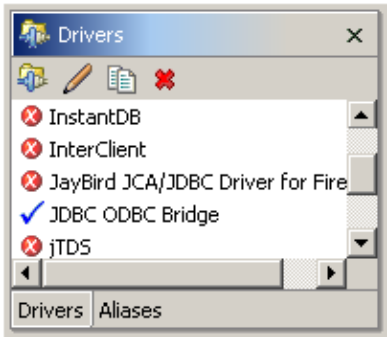
Il est aussi possible d'installer JFaceDBC en utilisant le gestionnaire de mises à jour avec l'URL http://jfacedbc.sourceforge.net/update−site/site.xml

JFaceDBC propose une perspective particulière.

Cette perspective se compose de plusieurs vues et editeurs.

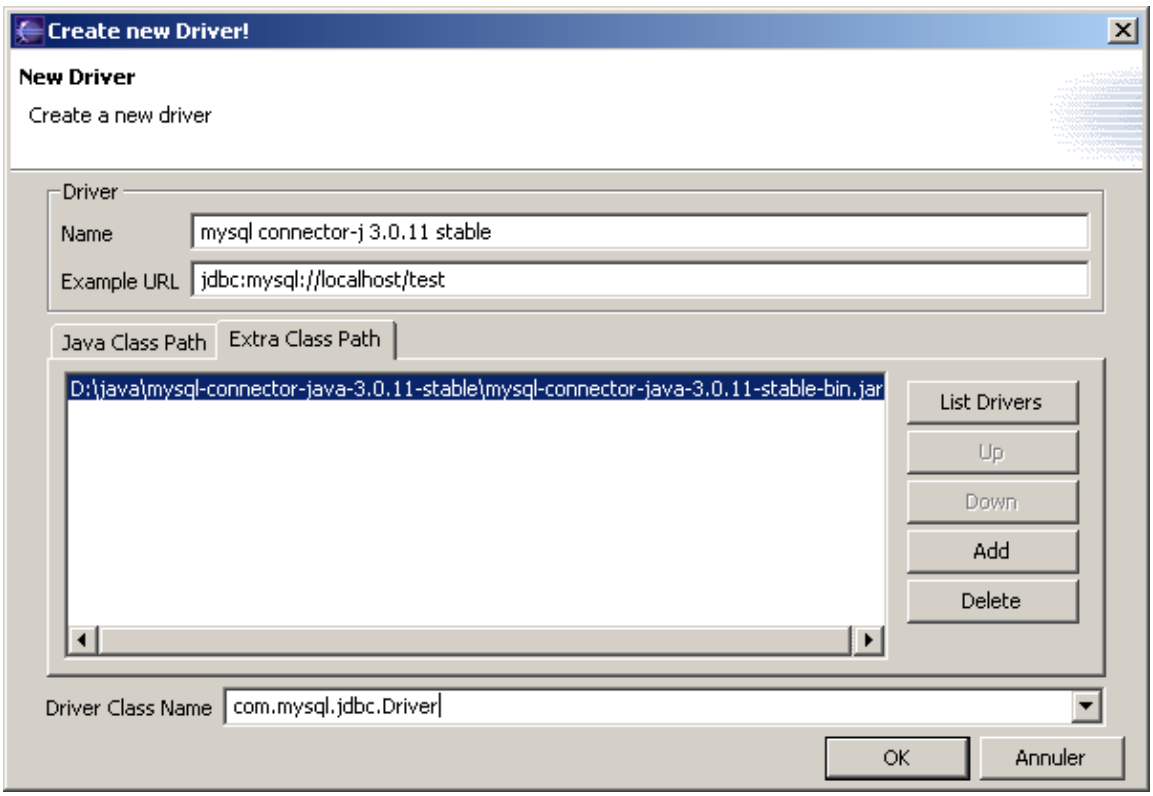


La vue "Drivers" permet de gérer les pilotes JDBC.



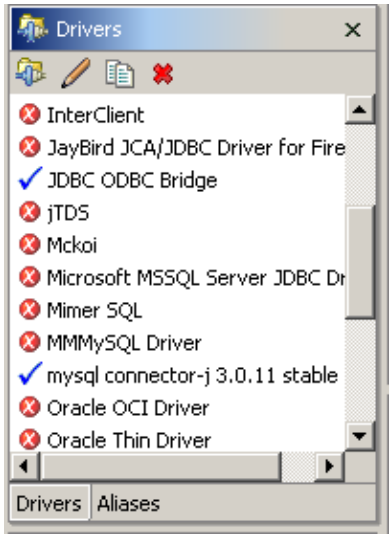
Un certain nombre d'entre eux sont préconfigurés et chargés dynamiquement si ils sont trouvés dans le classpath. Si le chargement a réussi alors une petite coche bleu est affichée au lieu d'une croix blanche dans un rond rouge.

Il est possible d'ajouter un nouveau pilote en cliquant sur le bouton  . Voici un exemple avec le pilote MySQL Connector−J 3.0



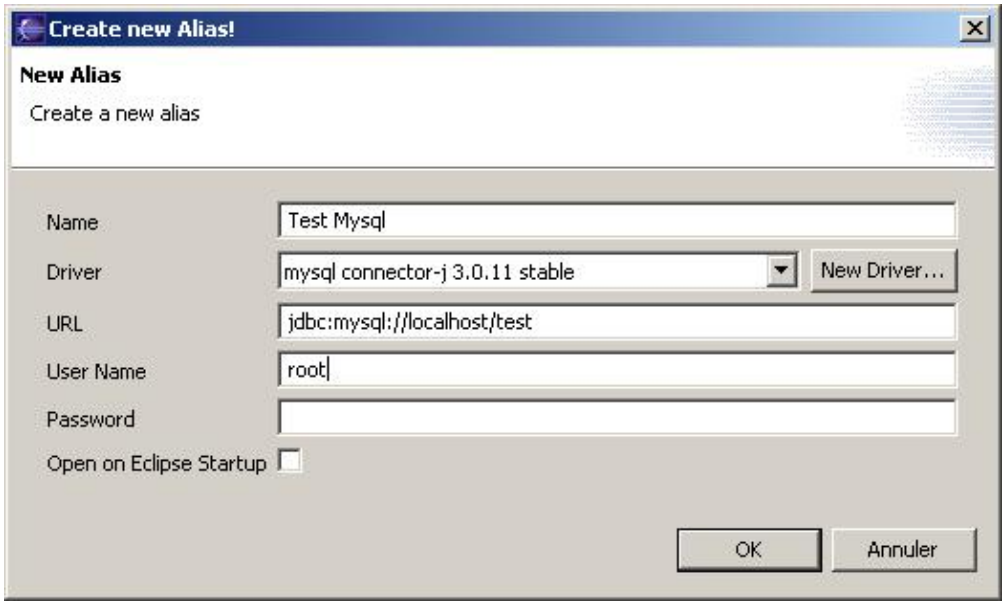
Il faut renseigner le nom du pilote, un exemple d'URL de connexion, sélectionner le fichier jar pour l'ajouter dans l'onglet « Extra Class Path », saisir le nom de la classe du Pilote et cliquer sur le bouton « OK »

Si le chargement du pilote a réussi, alors la petite coche bleue est affichée.

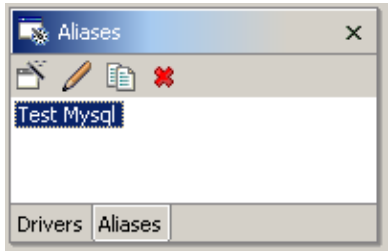


La vue "Aliases" permet de gérer des connexions. Pour ajouter un alias, il suffit de cliquer sur le bouton  ou d'utiliser l'option « Create New Alias » du menu contextuel.

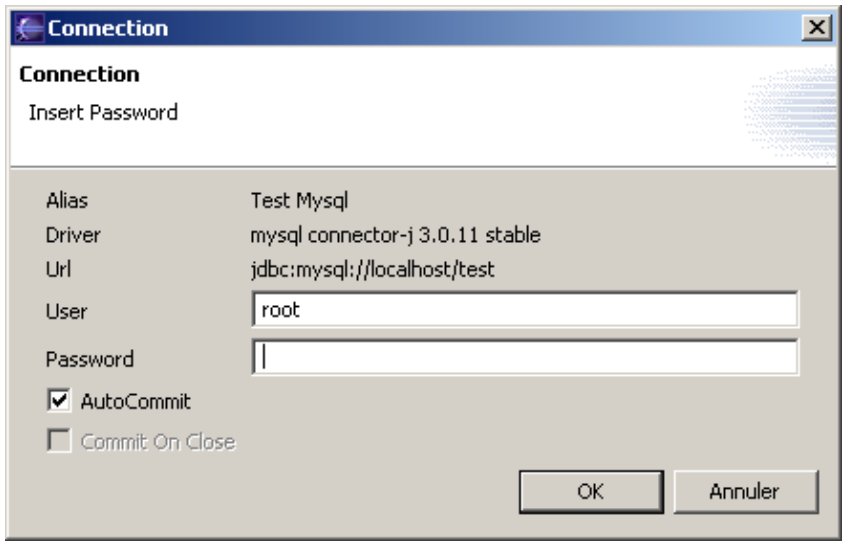
Une boite de dialogue permet de saisir les informations concernant le nouvel alias de connexion.



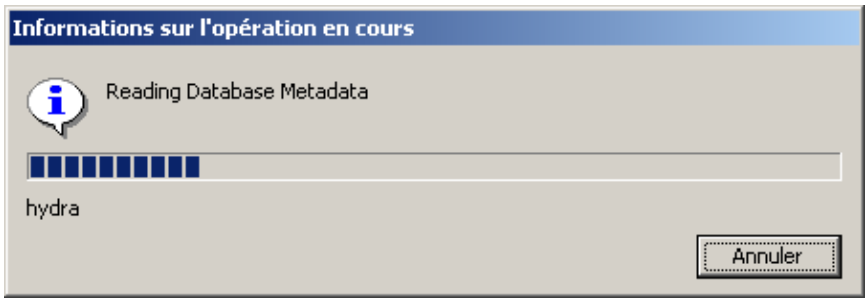
Il faut saisir un nom, sélectionner le pilote parmis la liste de ceux définis, saisir l'URL de la connexion, le nom de l'utilisateur et son mot de passe.



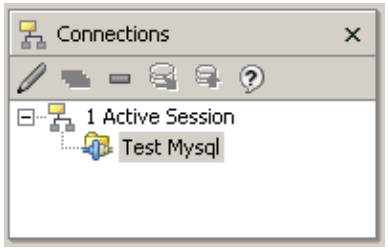
Pour ouvrir une connexion, il suffit de double cliquer sur l'alias ou d'utiliser l'option « Open » de son menu contextuel.



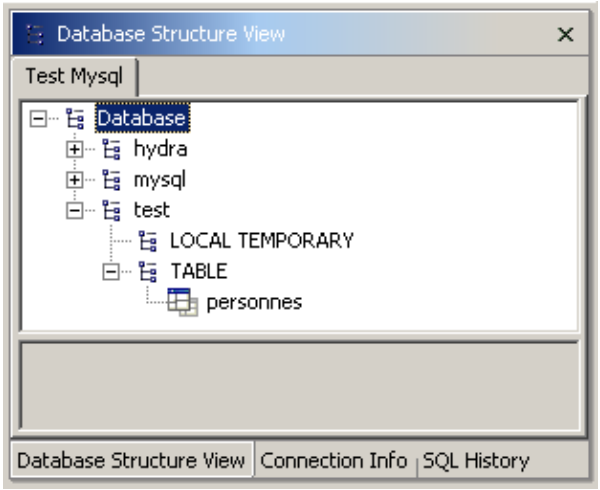
Saisissez les informations et cliqueez sur le bouton "OK". JFaceDBC retrouve les informations de la base de données



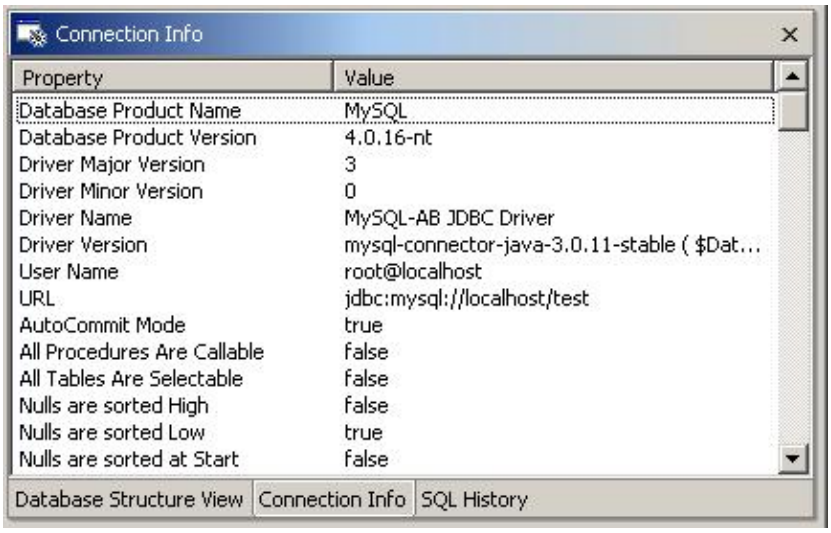
La vue "Connections" affiche la ou les connexions actives :



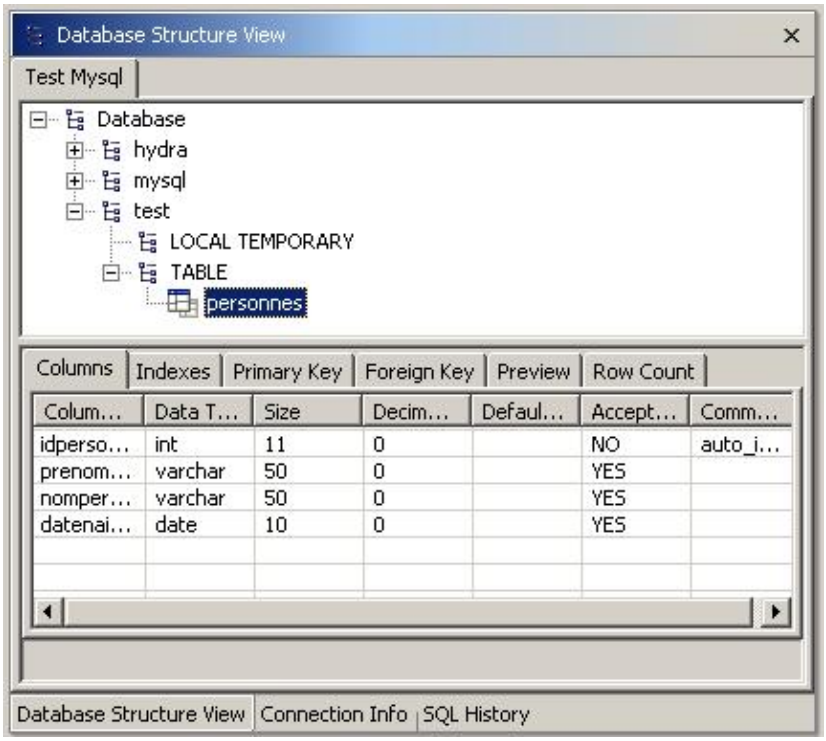
La vue « Database Structure » affiche les bases de données accessibles via la connexion.



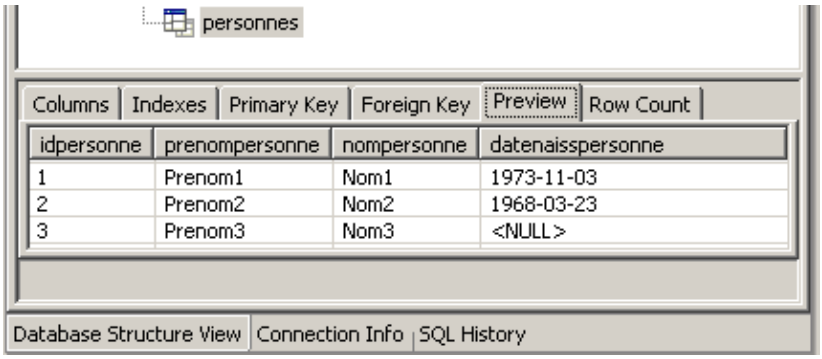
La vue « Connection Info » permet d'obtenir des informations sur les paramètres de la connexion .



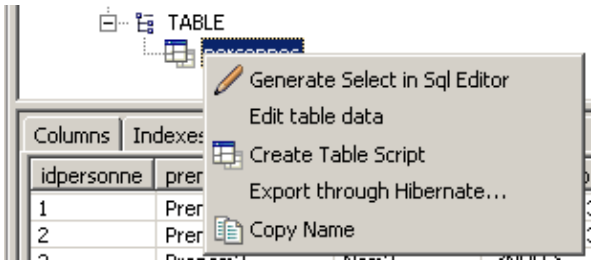
Pour connaître la structure d'une table, il suffit de la sélectionner dans la vue « Database Structure View »



Les données peuvent être visualisées dans l'onglet « Preview ».



Une table possède un menu contextuel qui propose plusieurs options :



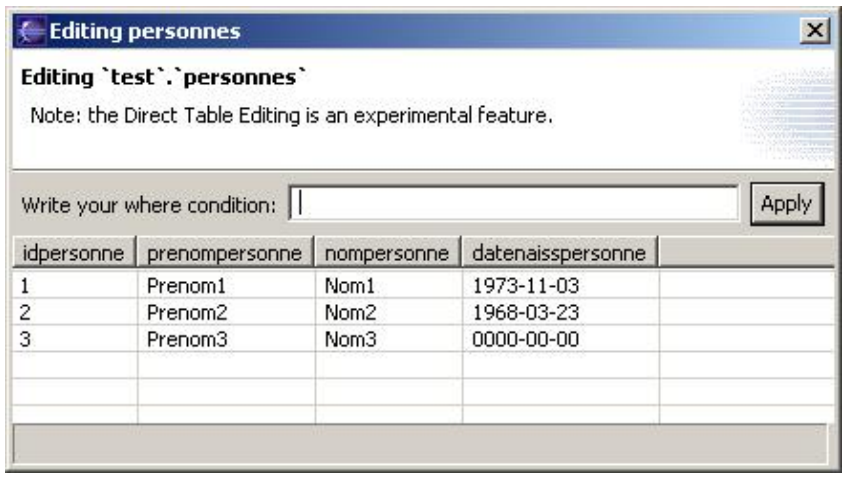
L'option « Generate Select in SQL Editor » ouvre l'éditeur SQL (« SQL Editor ») avec une requête permettant d'afficher le contenu de la table.

24.2.2. La mise à jour des données d'une table.

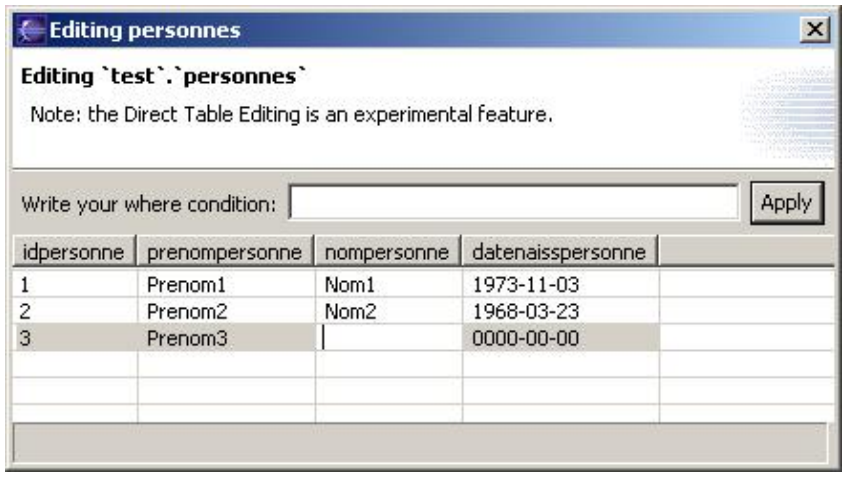
JFaceDBC propose une fonction pour mettre à jour les données d'une table.

Attention, cette fonctionnalité n'est utilisable qu'avec un pilote JDBC autorisant les déplacements des curseurs dans les deux sens.

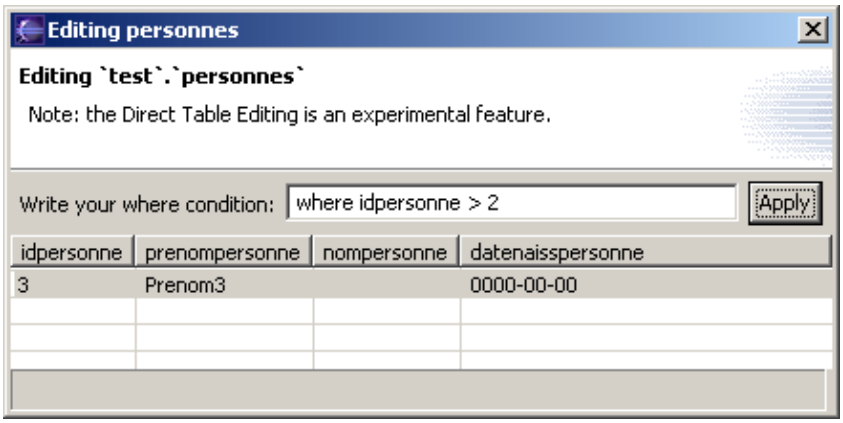
Pour utiliser cette fonctionnalité, il suffit d'utiliser l'option « Edit Table Data » du menu contextuel d'une table dans la vue « Database Structure ».



Il suffit alors de cliquer sur les données à modifier et de saisir sa nouvelle valeur :

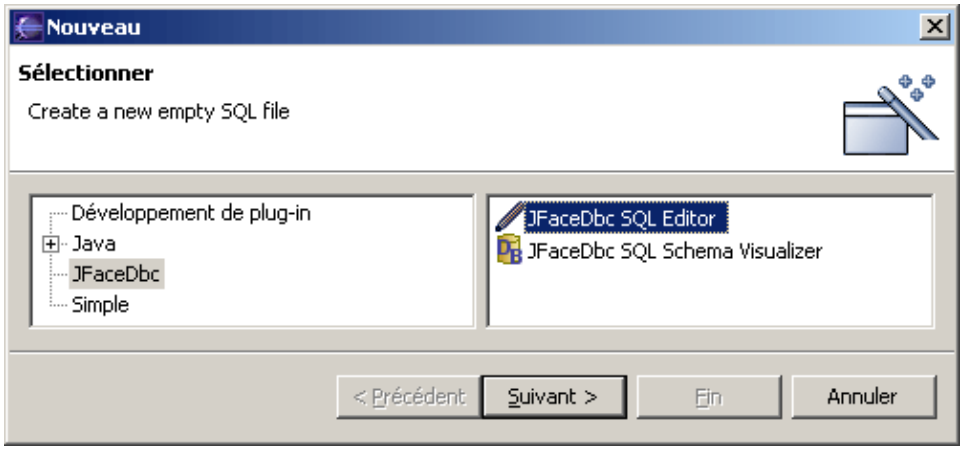


Il est possible d'appliquer un filtre pour limiter le nombre d'occurrences affichées dans la grille. Pour cela, il suffit de saisir la clause where qui sera ajoutée à la requête de sélection et de cliquer sur le bouton « Apply ».

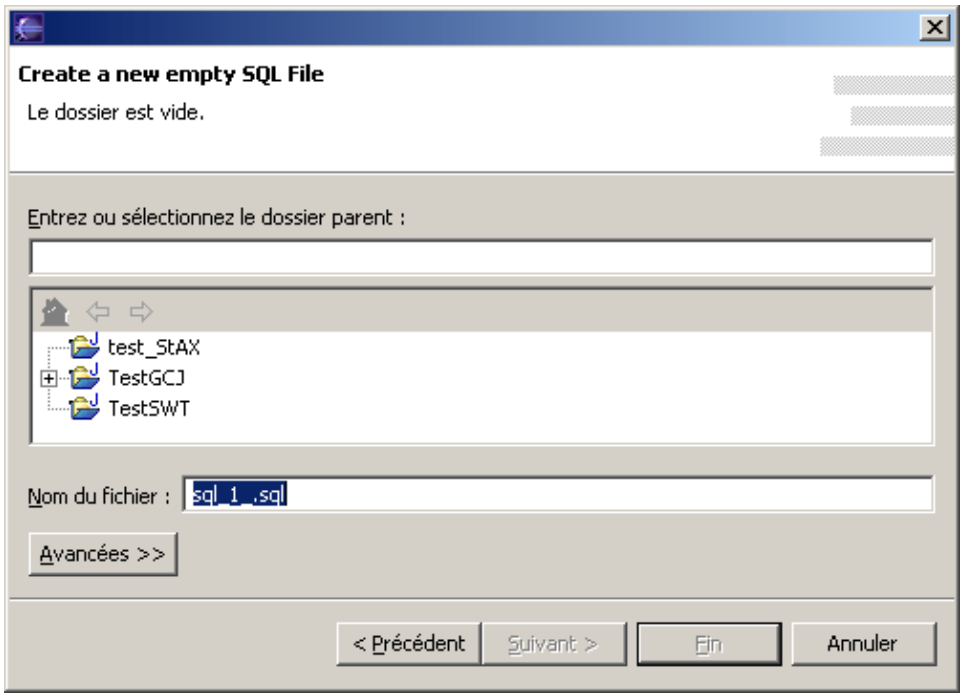


24.2.3. L'éditeur SQL

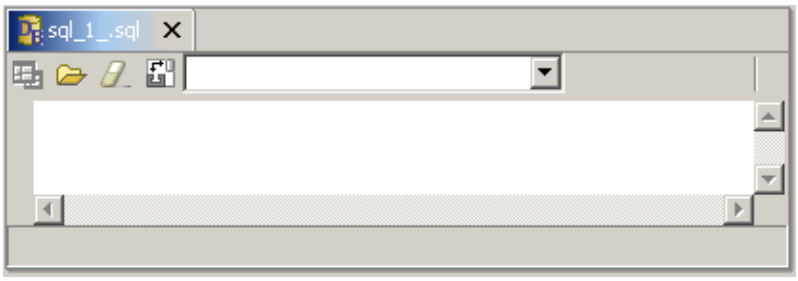
Pour ouvrir un nouvel éditeur SQL, il faut créer une nouvelle entité du type « JFaceDBC/JFaceDBC SQL Editor »



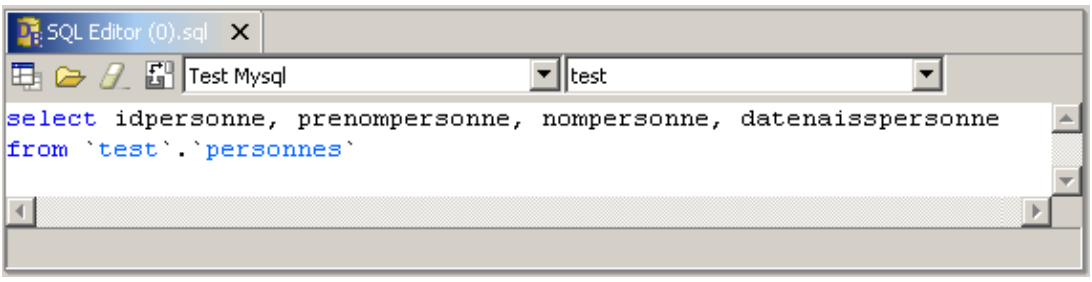
L'assistant demande ensuite le nom du fichier .sql qui sera créé et de sélectionner le projet qui va le contenir.



Un clic sur le bouton « Fin » crée le fichier et ouvre l'éditeur :



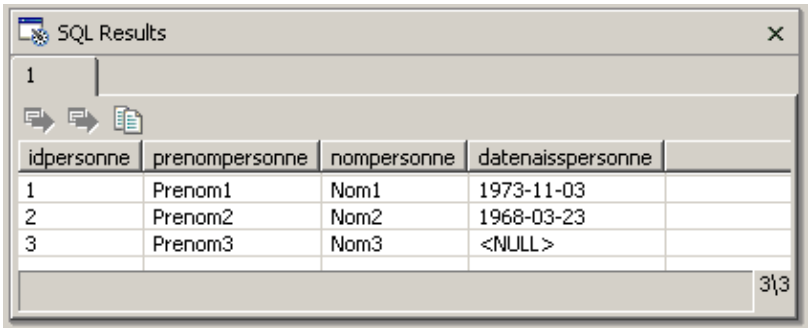
La première liste déroulante permet de sélectionner la connexion qui sera utilisée.



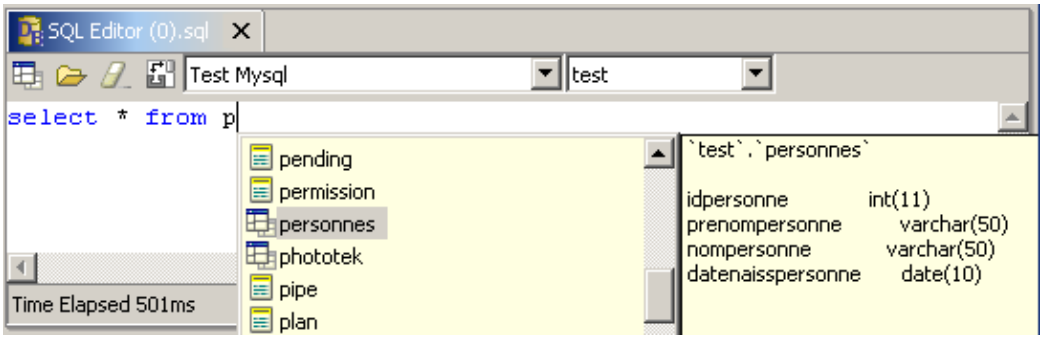
La seconde liste déroulante permet de sélectionner la base de données à utiliser.

Le bouton  ou l'option « Execute SQL » du menu contextuel permet de lancer l'exécution de la requête.

Le résultat de la requête s'affiche dans la vue « SQL Results ».



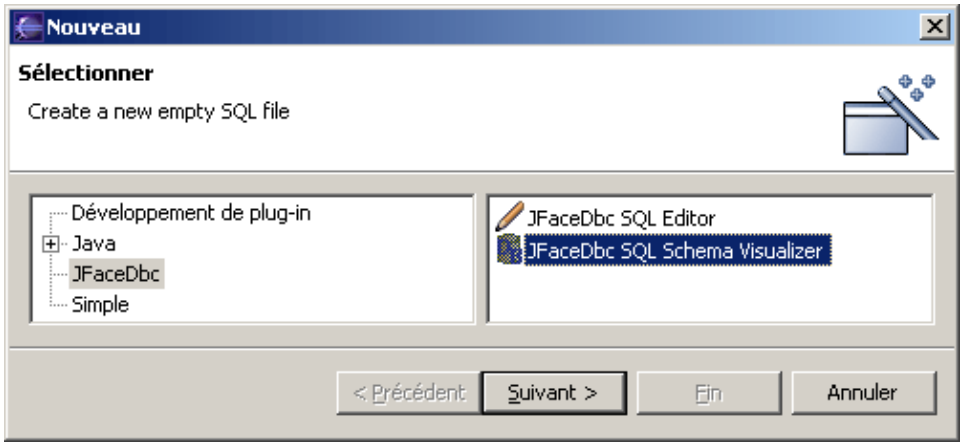
L'éditeur propose une coloration syntaxique des différents éléments de la requête mais aussi une complétion de code pour les tables et les champs, ce qui est très pratique. Cette fonction est appelée en utilisant la combinaison de touches "Ctrl + espace".



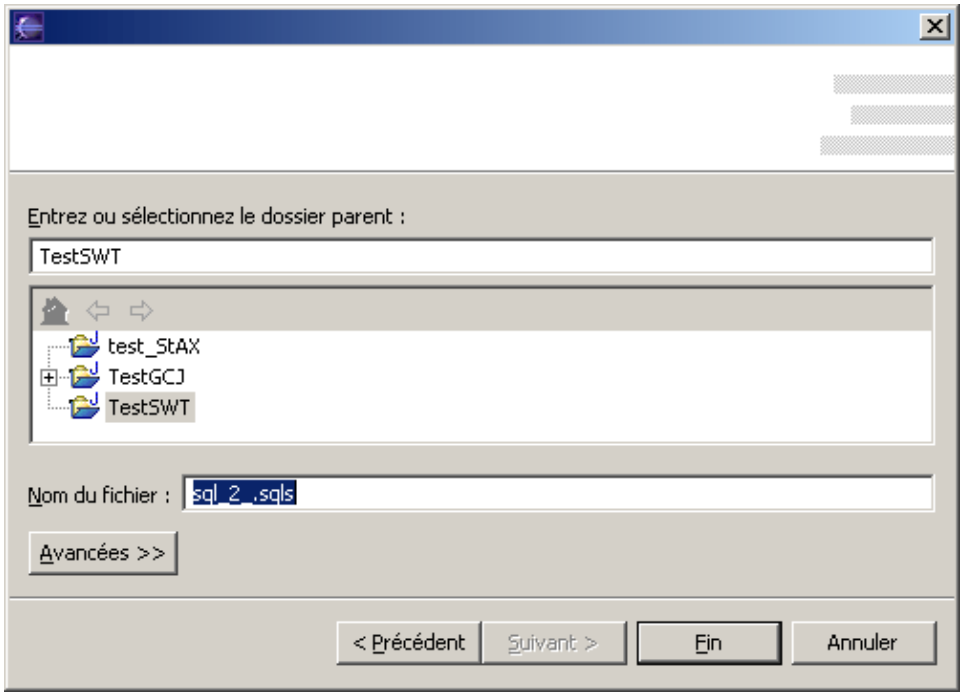
24.2.4. Vue graphique d'une base de données

Une autre fonctionnalité intéressante de JFaceDBC est de pouvoir afficher un schéma d'une base de données sous une forme graphique.

Pour cela, il faut créer une nouvelle entité de type "JFaceDBC/JFaceDBC SQL Schema Visualizer".

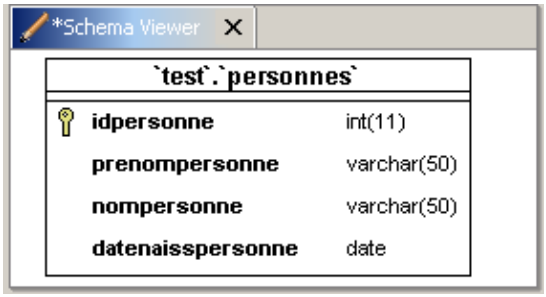


L'assistant demande ensuite le nom du fichier .sqls qui sera créé et de sélectionner le projet qui va le contenir.

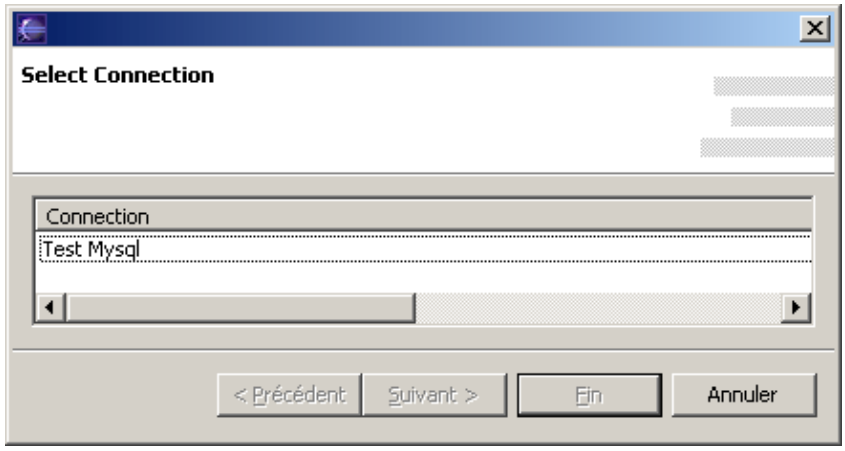


Un clic sur le bouton « Fin » créé le fichier et ouvre l'éditeur. Par défaut, ce dernier est vide.

Pour ajouter une nouvelle table au schéma, il suffit de faire un cliquer/glisser entre une table de la vue « Database Structure » et l'éditeur.

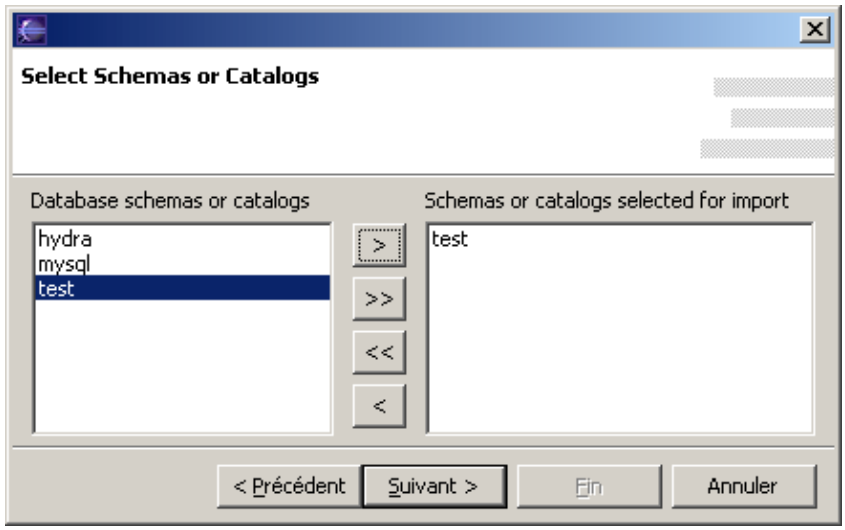


Le menu contextuel de l'éditeur propose l'option « Reverse Database ». Un assistant demande de sélectionner la connexion concernée.

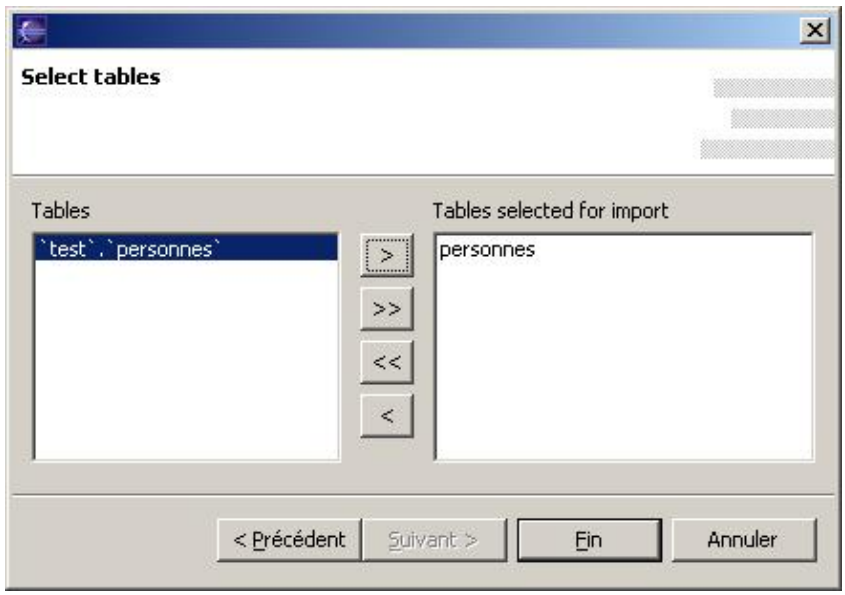


Il suffit de sélectionner la connexion concernée et de cliquer sur le bouton « Suivant ».

Il faut ensuite sélectionner la ou les bases de données à inclure parmis celles accessibles via la connexion.



Cliquer sur le bouton « Suivant ». L'assistant demande de sélectionner la ou les tables concernées.



Un clic sur le bouton « Fin » génére le schéma dans l'éditeur.

24.3. DBEdit

DBEdit est un plug−in qui permet l'exécution de requêtes sur une base de données.

Le site officiel est http://sourceforge.net/projects/dbedit

	Version utilisée
Eclipse	2.1.2
J2RE	1.4.2_02
DBEdit	1.0.1.

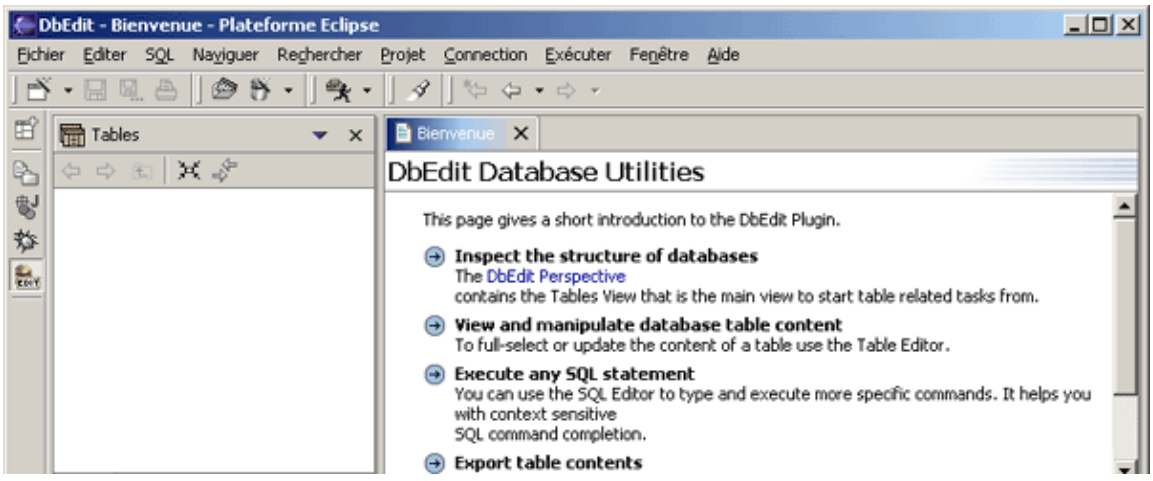
24.3.1. Installation et configuration.

Il faut télécharger le fichier dbedit_1.0.1.bin.dist.zip sur le site de DBEdit et décompresser son contenu dans le répertoire d'installation d'Eclipse. A l'exécution suivante, il est nécessaire de valider les modifications en attente et de relancer l'application.

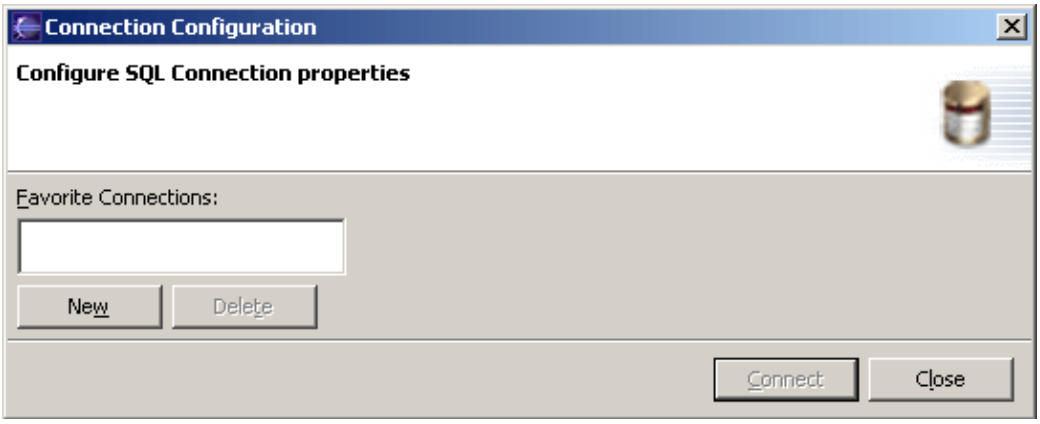
Il est aussi possible d'utiliser l'URL http://www.geocities.com/uwe_ewald/dbedit/site dans le gestionnaire des mises à jour.

DBEdit propose une perspective particulière.

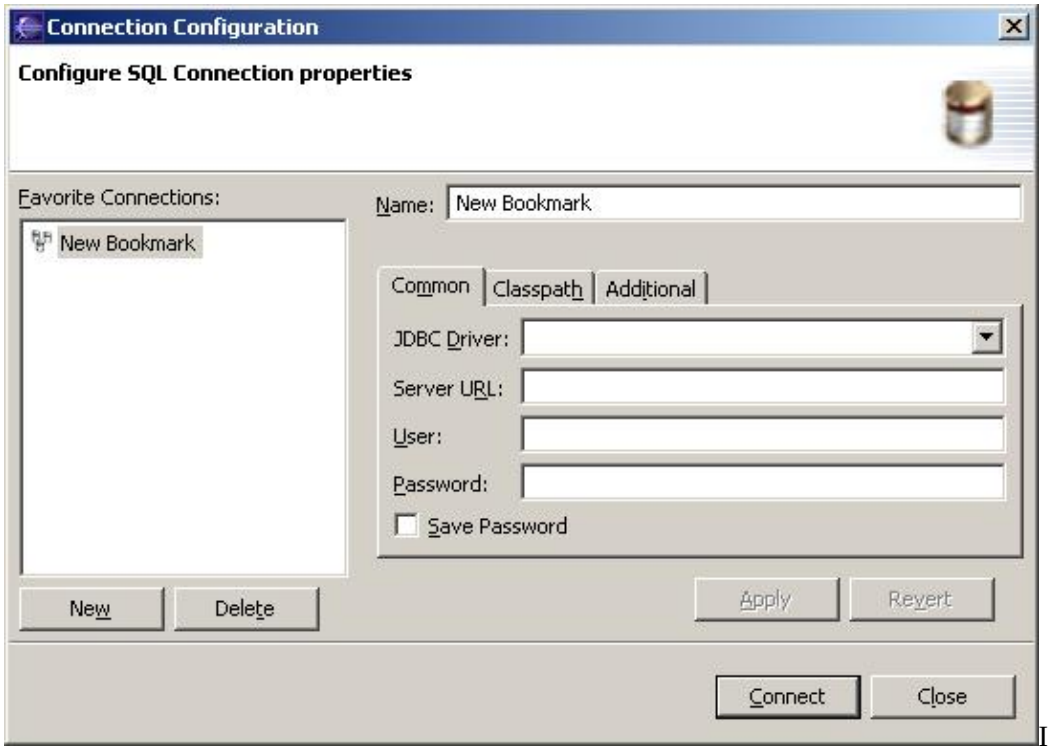
Cette perspective se compose de plusieurs vues et editeurs.



La première chose à faire est de définir une connexion en utilisant l'option « Connection/Configure » du menu contextuel de la vue « Tables ». Une boite de dialogue permet de gérer les connexions.



Pour ajouter une nouvelle connexion, il suffit de cliquer sur le bouton « New ».

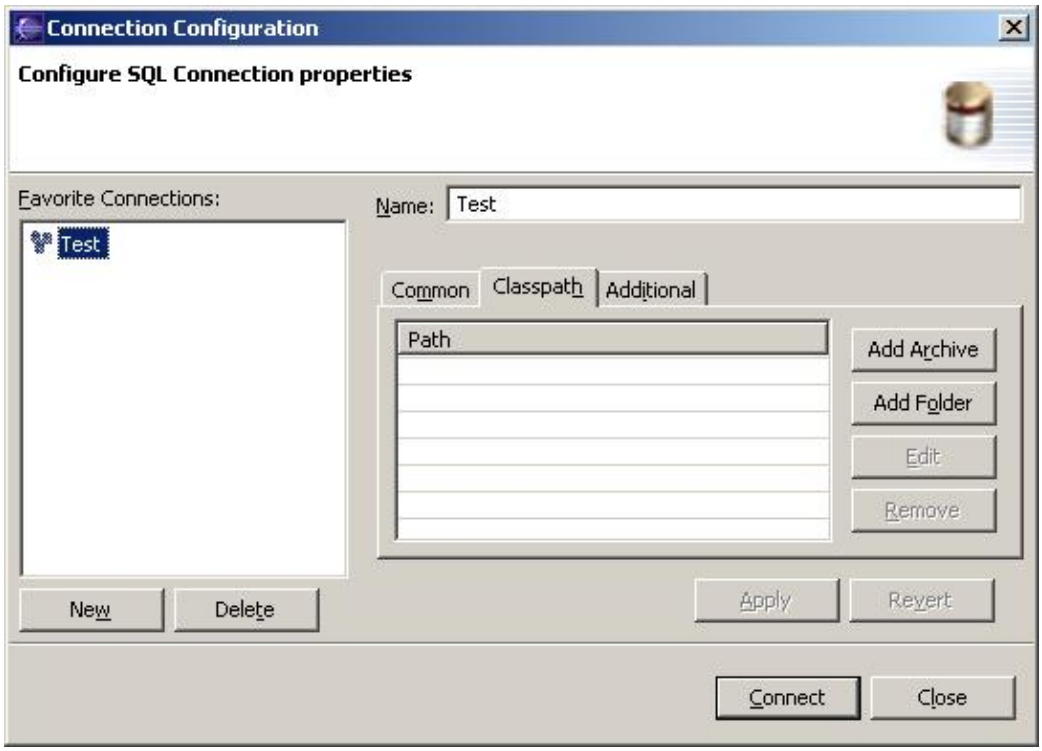


Il suffit alors de saisir les informations nécessaires :

- Name : est le nom de la connexion
- JDBC Driver : permet de préciser le nom de la classe du pilote JDBC
- Serveur URL : est l'URL de connexion
- User et password : sont le nom et le mot de passe de l'utilisateurs

Un clic sur le bouton « Apply » valide les informations.

Il faut ensuite ajouter le pilote au classpath en cliquant sur l'onglet « Classpath »



Il suffit de cliquer sur le bouton « Add Archive » et de sélectionner le fichier Jar contenant le pilote à utiliser.

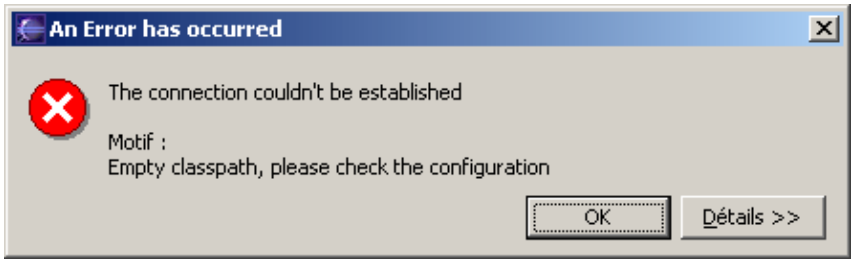
L'onglet « Additional » permet de préciser des paramètres supplémentaires au pilote JDBC.

Un clic sur le bouton « Connect » permet d'ouvrir la connexion.

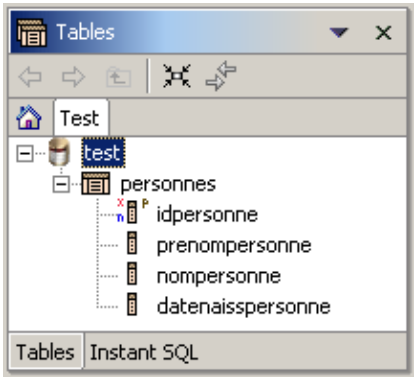
24.3.2. La vue « Tables »

Par défaut, elle affiche la liste des connexions enregistrées. Pour ouvrir une connexion, il suffit d'utiliser l'option « Connection/Connect » du menu contextuel d'une connexion.

Si le pilote n'est pas inclus dans le classpath, un message d'erreur est affiché



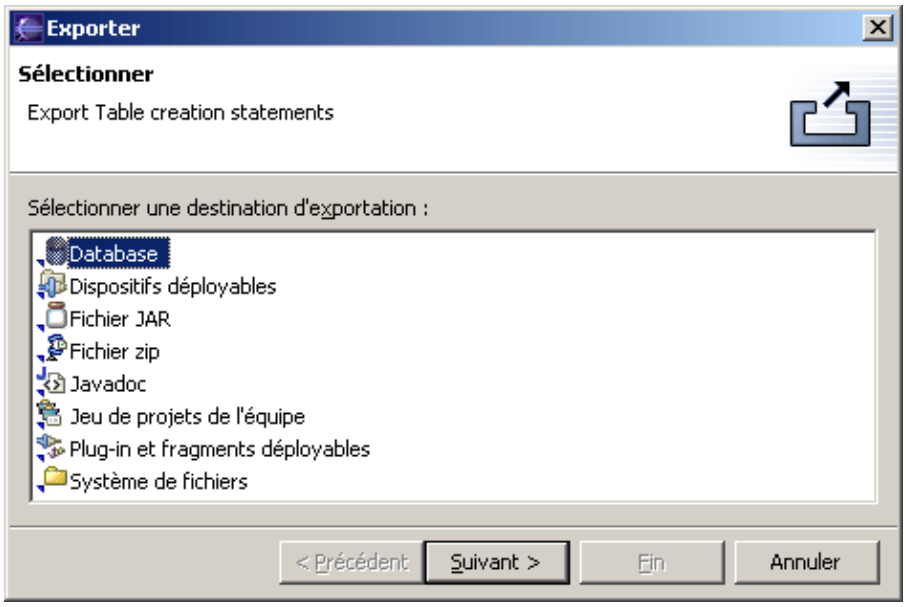
Une fois la connexion établie, la vue « Tables » affiche une arborescence contenant les éléments de la base de données : tables, vues, champs, index ....



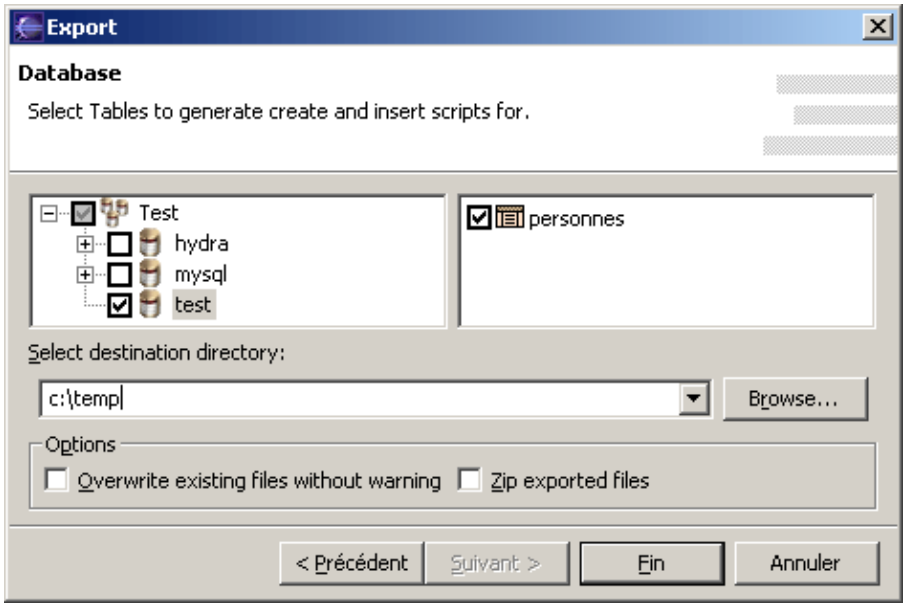
A partir du menu contextuel de cette vue, il est possible de réaliser plusieurs actions :

- créer une nouvelle entité : connexion, schéma, table, fichier SQL
- importer ou exporter une table
- ouvrir une table dans l'éditeur

L'export se fait avec un assistant :



Il faut sélectionner "Database" et cliquer sur le bouton « Suivant »

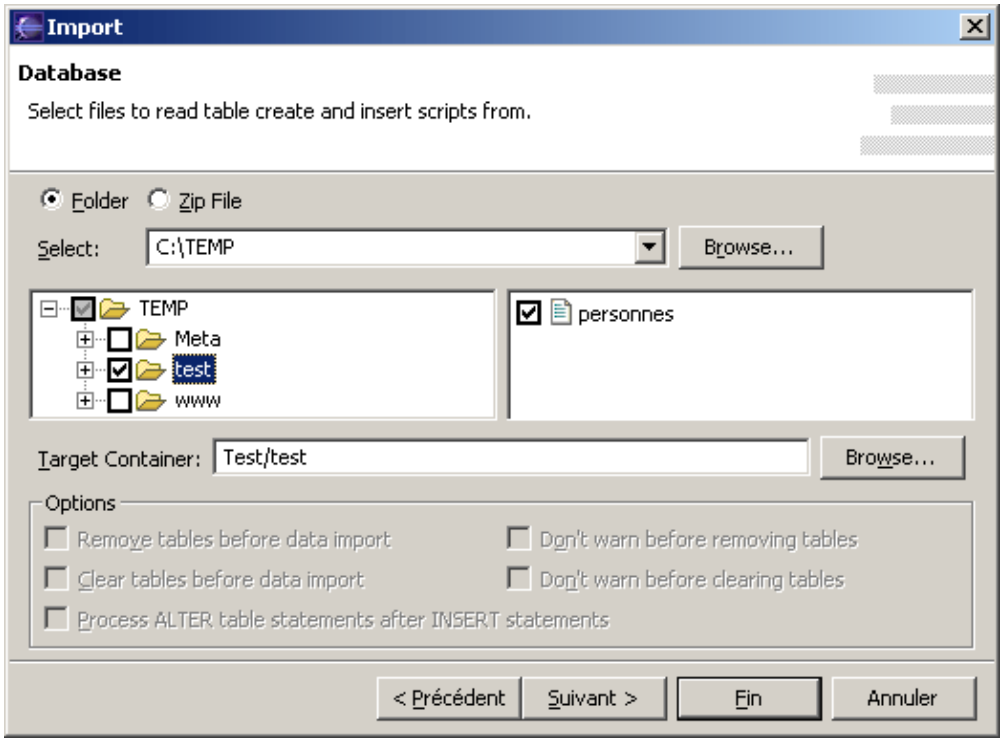


Il faut sélectionner la connexion et les tables concernées, sélectionner le répertoire de destination et cliquer sur le bouton « Fin ».

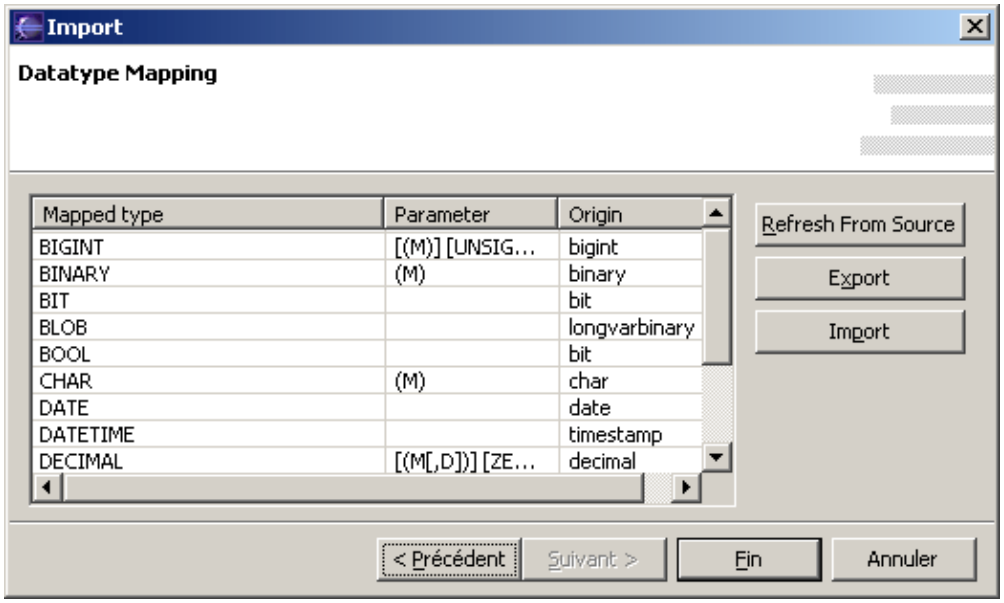
L'exportation se fait sous le forme de requètes SQL :

```
1 quote=`
2 create table `personnes` (`idpersonne` int not null, `prenomper
3 alter table `personnes` add constraint `PRIMARY` primary key (`
4 alter table `personnes` add constraint `idpersonnes` primary ke
5 insert into `personnes` (`idpersonne`, `prenompersonne`, `nompe
6 insert into `personnes` (`idpersonne`, `prenompersonne`, `nompe
7 insert into `personnes` (`idpersonne`, `prenompersonne`, `nompe
8 insert into `personnes` (`idpersonne`, `prenompersonne`, `nompe
9
```

L'importation se fait aussi avec un assistant

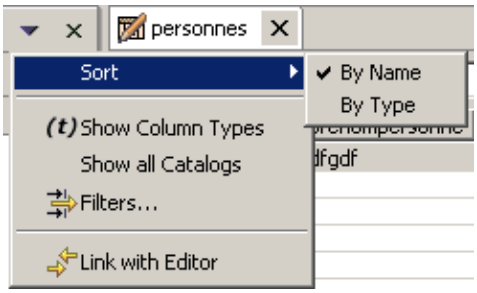


Il suffit de sélectionner les entités et de cliquer sur le bouton « Suivant ». La page suivante permet de préciser les options de correspondance des types de données.

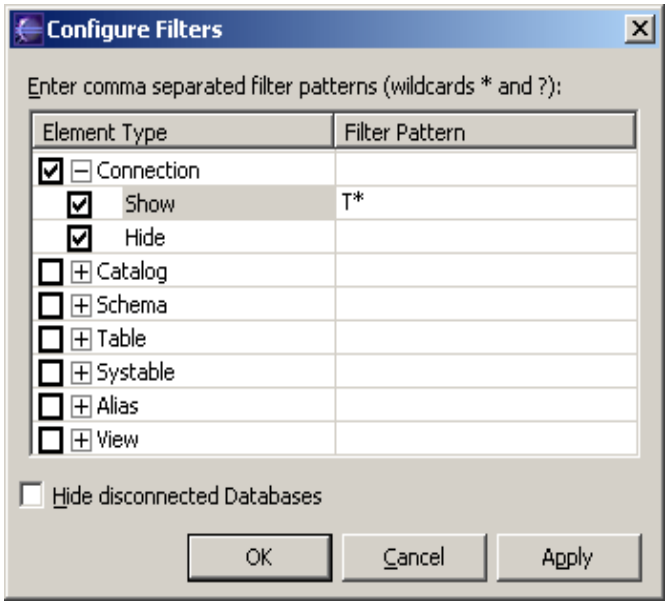


Un clic sur le bouton « Fin » permet de lancer l'importation.

Un clic sur le bouton  permet de régler certaines options d'affichage.

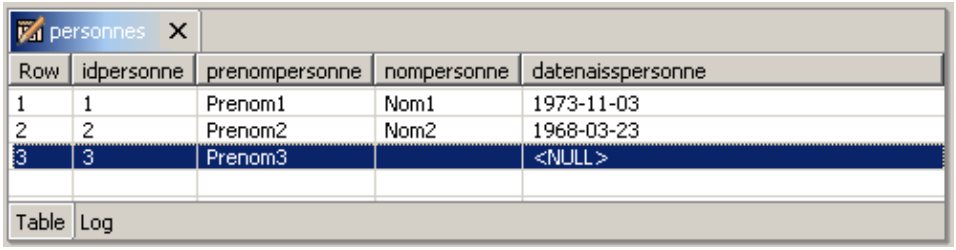


L'option "Filters" permet de restreindre les types d'éléments qui sont affichés :



24.3.3. L'editeur « Table »

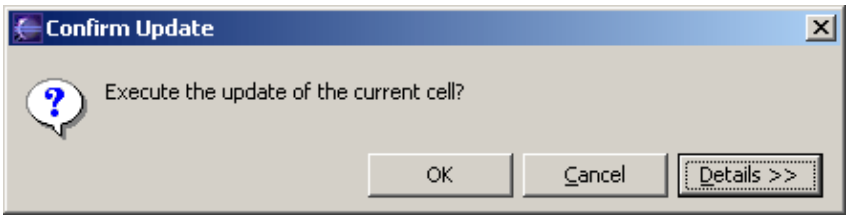
Sur la vue « Tables », l'option « Open with/ Table Editor » du menu contextuel d'une table permet d'ouvrir un éditeur avec les données de la table.



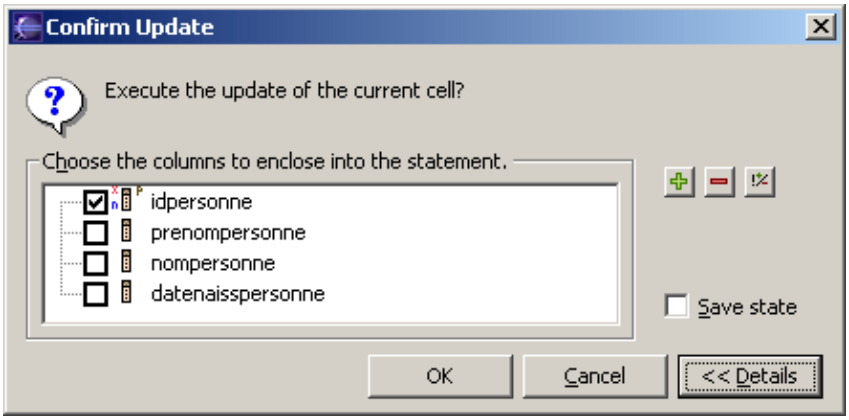
Par défaut, l'éditeur est en mode consultation. Le menu contextuel permet de réaliser certaines opérations :

- Edit Mode : permet de basculer dans le mode de mise à jour des données pour un seul champ
- Insert Row : permet d'insérer une nouvelle occurrence dans la table
- Delete Row : permet de supprimer l'occurrence sélectionnée

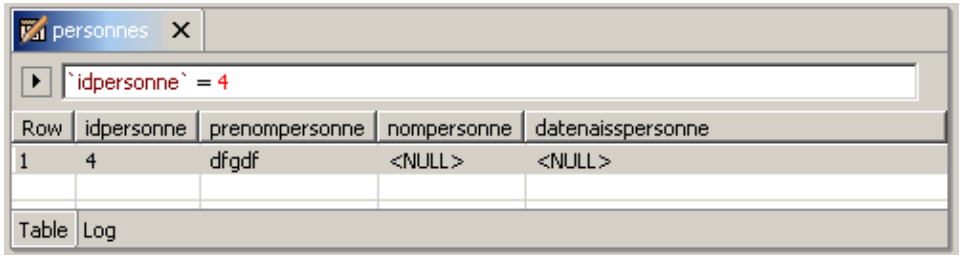
Chaque mise à jour doit être confirmée dans une boîte de dialogue



Un clic sur le bouton « Details » permet d'obtenir des informations précises sur l'opération.

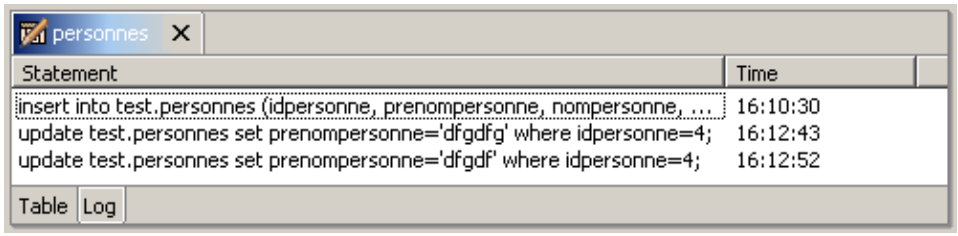


Le bouton dans la barre d'outils permet de rafraichir les données de l'éditeur. Le bouton dans la barre d'outils permet de saisir un filtre.



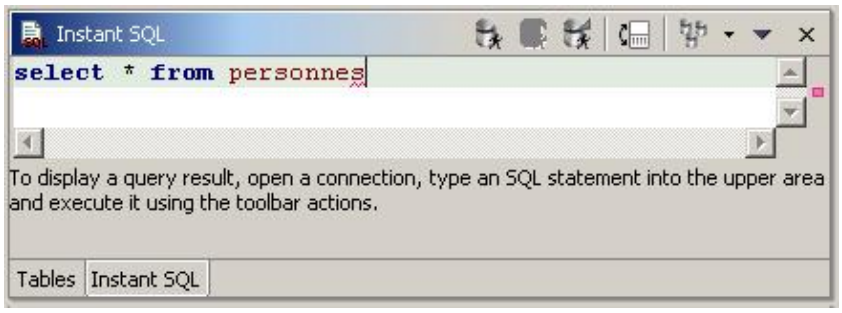
24.3.4. La vue Log

Cette vue recense toutes les requêtes SQL qui sont exécutées.

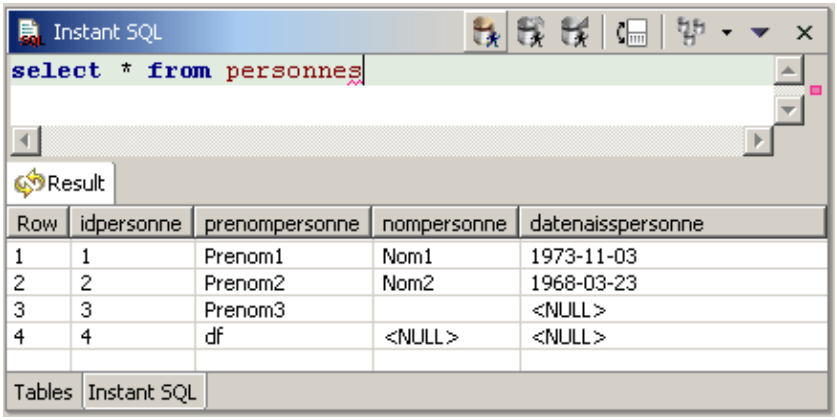


24.3.5. L'éditeur Instant SQL

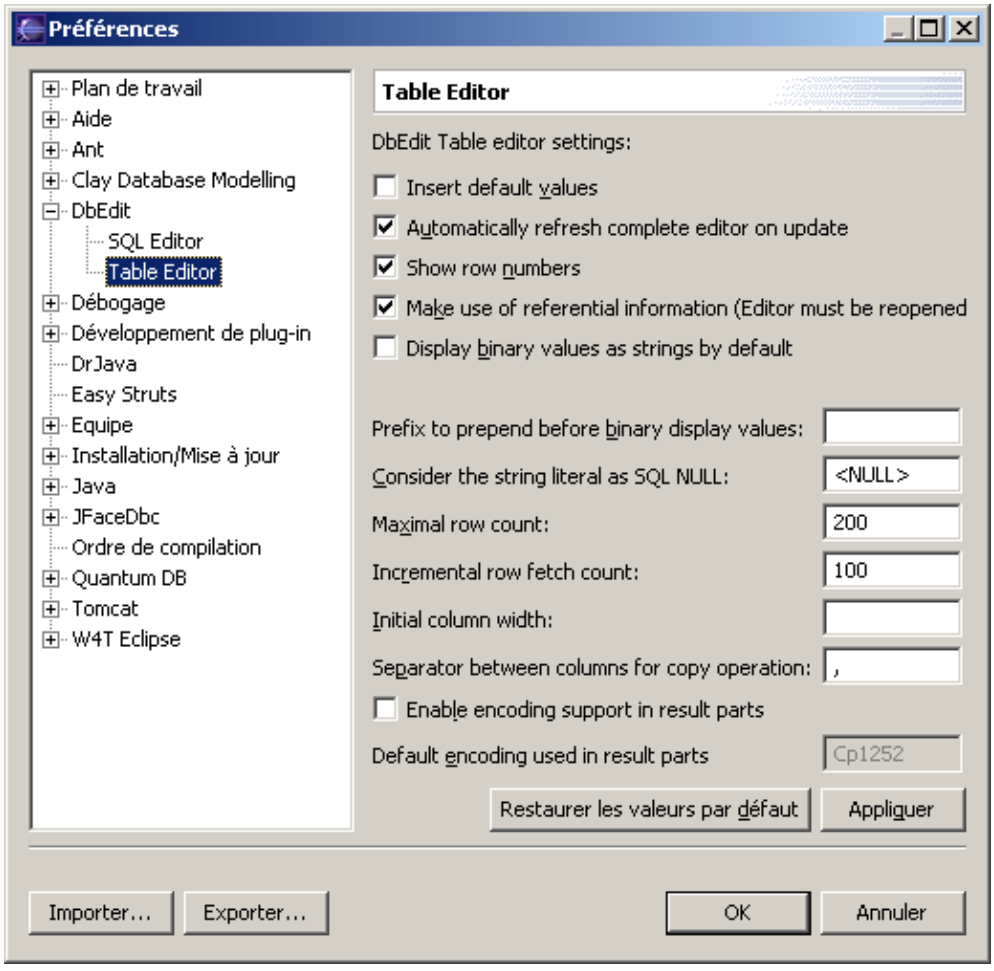
Cet éditeur permet de saisir et d'exécuter des requêtes. Il suffit de saisir la requête dans la zone d'édition



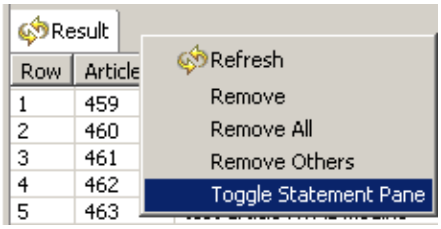
puis de cliquer sur le bouton  pour lancer l'exécution



Il est possible de limiter le nombre d'occurences renvoyées par une requête. Ceci est nécessaire pour des tables possèdant de nombreuses occurences car dans ce cas un affichage de toutes les occurences provoque une exception de type OutOfMemory. Ce nombre d'occurences est réglable dans les préférences.



La version 1.0.2 propose une pagination sur les occurences obtenues en fonction du nombre d'occurences maximales à afficher. Pour l'activer, il faut utiliser l'option "Toogle" du menu contextuel de la barre de titre "Résultats".



L'activation de cette option affiche un panneau permettant la pagination.

24.4. Clay Database Modelling

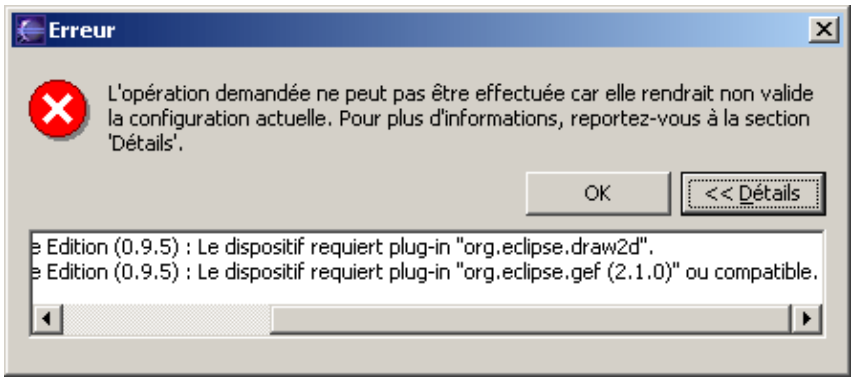
Ce plug−in permet de concevoir ou retro−concevoir une base de données graphiquement.

Les base de données supportées sont : HSQLDB, MySQL, PostgreSQL, Firebird, ANSI SQL−92, SAP DB, McKoi.

Le site officiel est http://www.azzurri.jp/en/software/clay/index.jsp

	Version utilisée dans cette section
Eclipse	2.1.2
J2RE	1.4.2_02
Clay Database Modelling	0.9.5

Ce plug−in nécessite le plug−in GEF (Graphical Editor Framework). Si il est absent, un message d'erreur est affiché.



24.4.1. Installation et configuration.

Il faut télécharger le fichier jp.azzurri.clay.core_0.9.5.bin.dist.20031201.zip sur le site du plug−in et décompresser son contenu dans le répertoire d'installation d'Eclipse. A l'exécution suivante, il est nécessaire

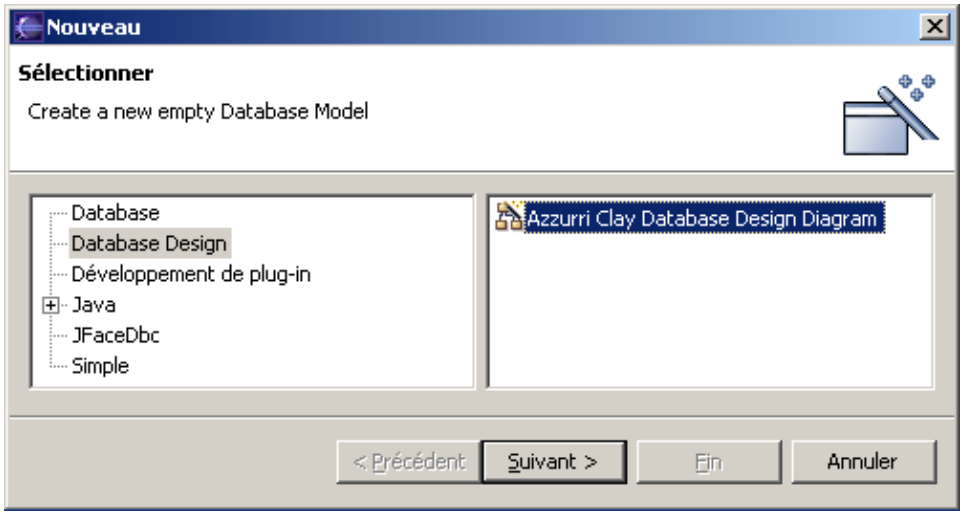
de valider les modifications en attente et de relancer l'application.

L'installation par le gestionnaire de mises à jour est possible en utilisant l'url http://www.azzurri.jp/eclipse/plugins

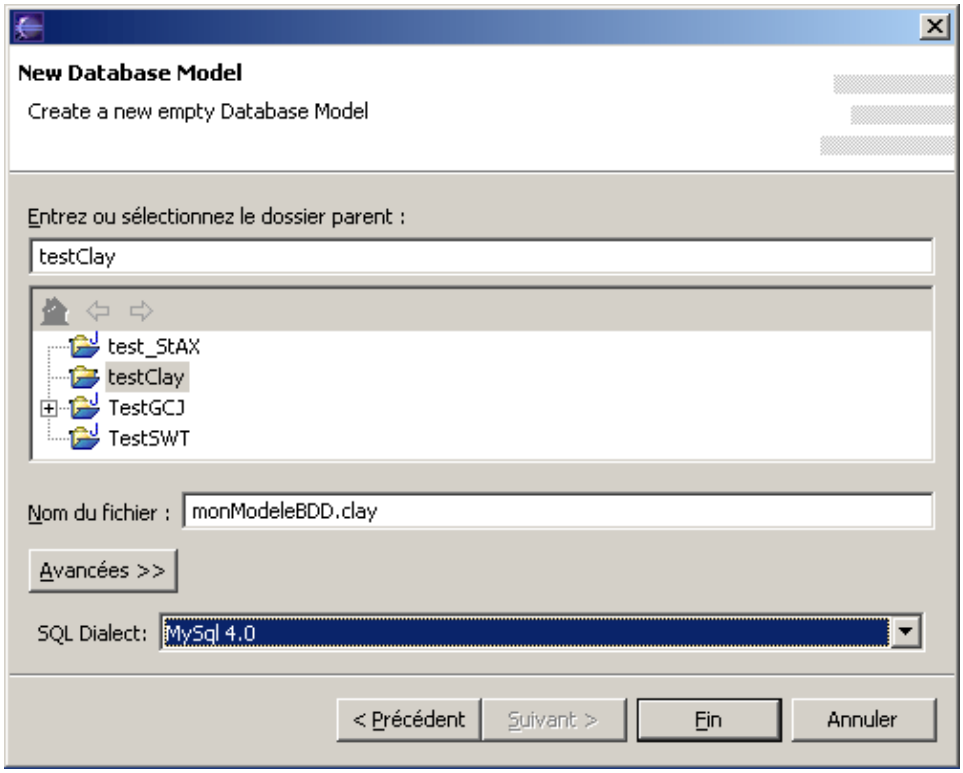
24.4.2. Mise en oeuvre

Clay Database Modelling ne propose pas de perspective particulière.

Il faut créer un nouveau projet ou utiliser un projet existant. Dans ce projet, il faut définir un nouveau modèle en utilisant l'option de création d'un nouvel élément de type « Autre ... ».

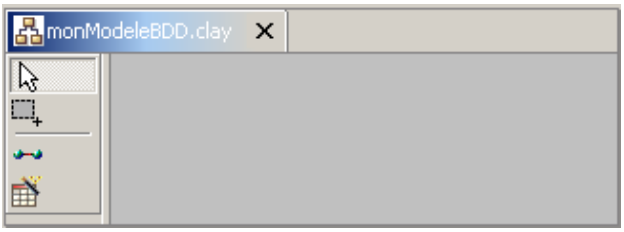


Il faut sélectionner « Database design » et « Azzurri Clay Database Design Diagram » puis cliquer sur le bouton « Suivant ».

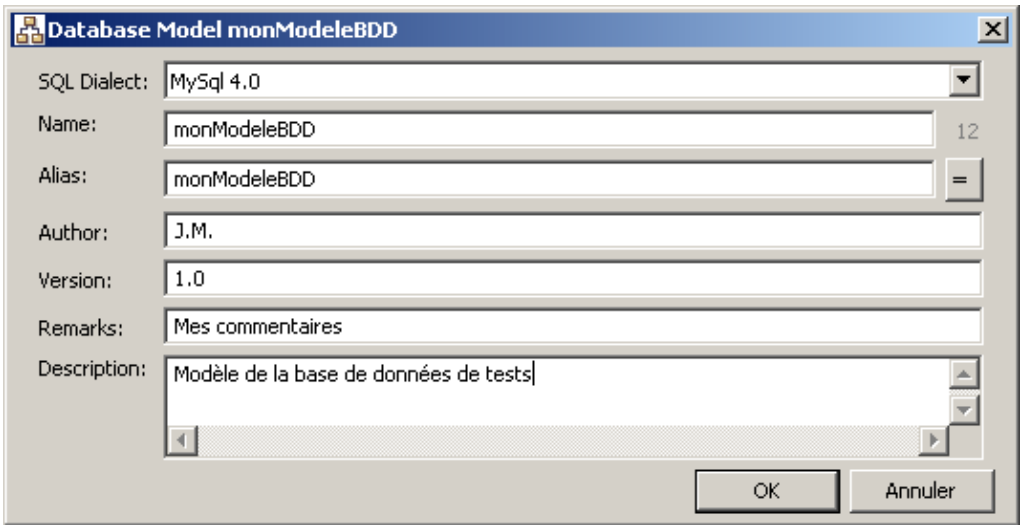


Il faut sélectionner l'emplacement du fichier de données (par défaut le répertoire du projet en cours), saisir le nom de ce fichier, et sélectionner le type de base de données cible dans lequel les scripts SQL seront générés.

Un éditeur dédié s'ouvre avec le fichier créé, par défaut vide.



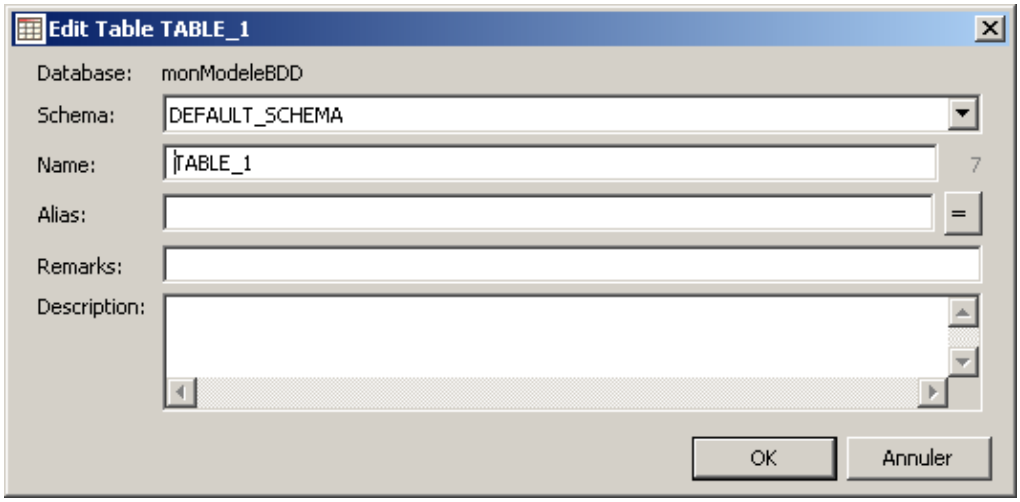
L'option « Edit Database Model » du menu contextuel permet de saisir des informations concernant le modèle.



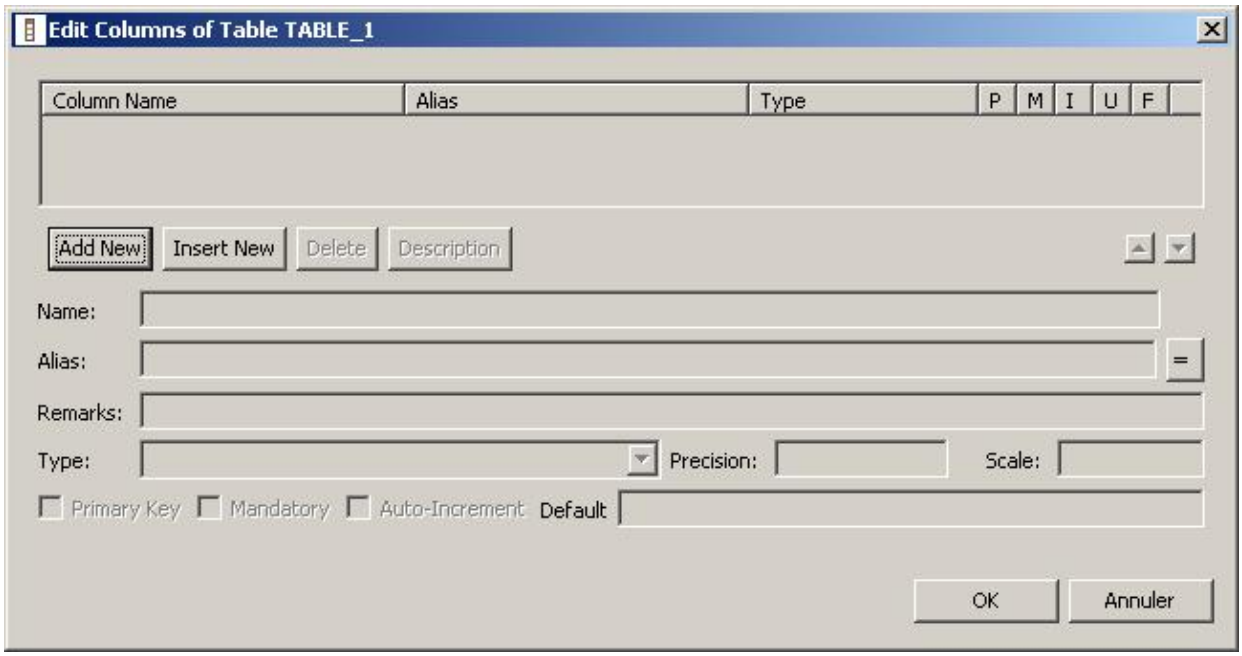
Dans cette boîte de dialogue, la donnée « SQL Dialect » est particulièrement importante car elle va conditionner les possibilités de l'éditeur selon le type de base de données sélectionnées, notamment par exemple au niveau des types de données utilisables dans les définitions de tables.

Le bouton  permet d'ajouter une nouvelle table au modèle en cliquant sur le bouton puis sur la zone d'édition de l'éditeur.

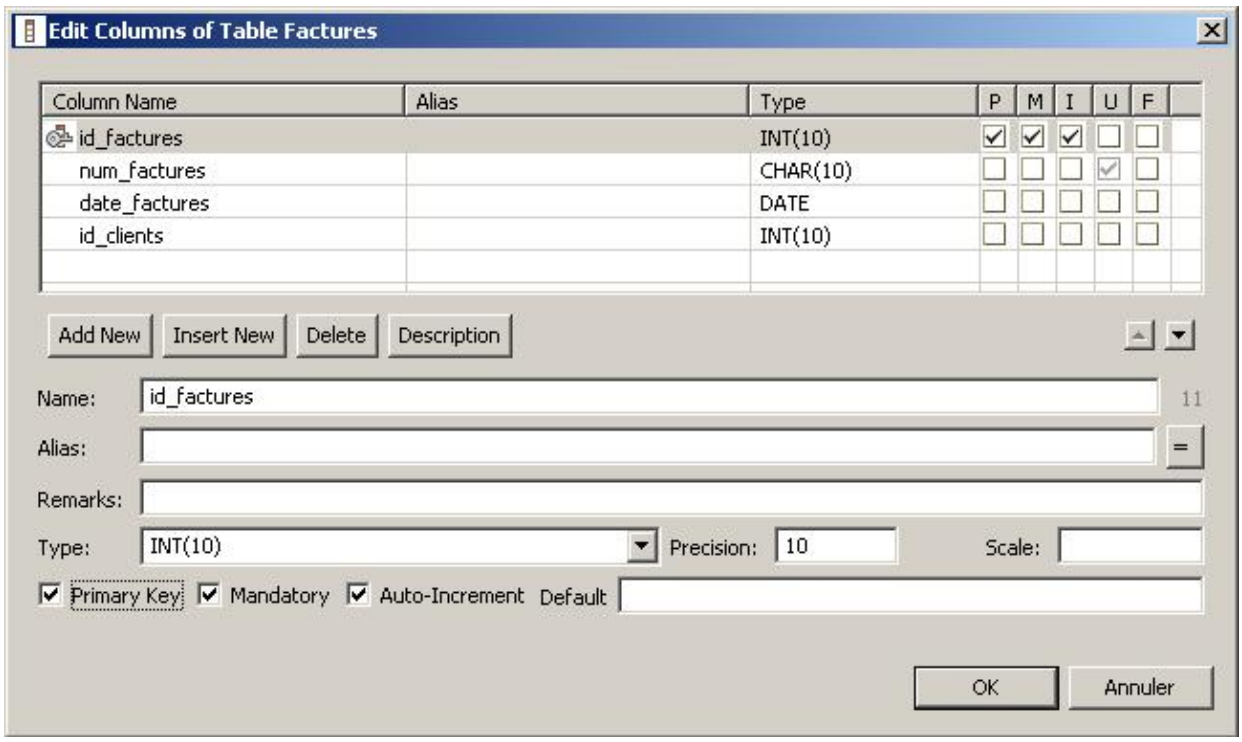
L'option « Edit Table » du menu contextuel permet de saisir des données générales de la table tel que son nom, une description ou des remarques.



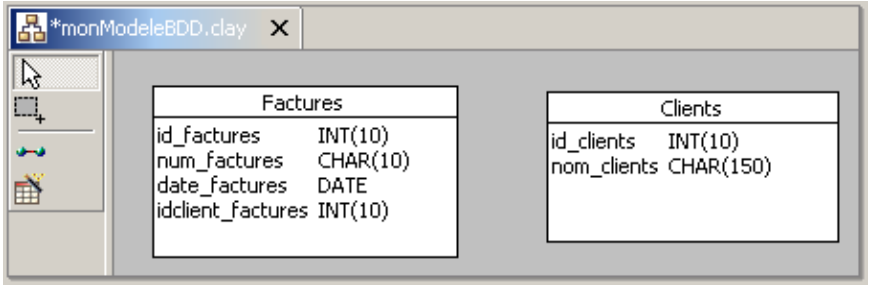
Un double clic sur la nouvelle table créée ou l'utilisation de l'option « Edit Table Column » permet de saisir la description des champs qu'elle va contenir.



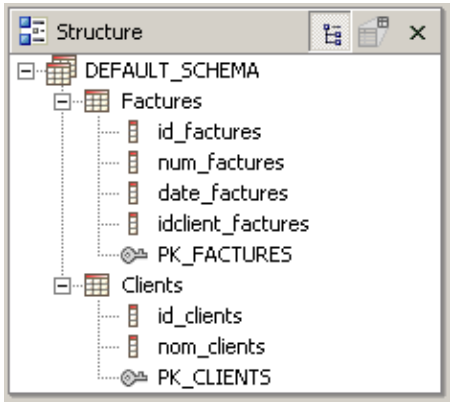
Pour ajouter une nouvelle colonne, il suffit de cliquer sur le bouton « Add New »



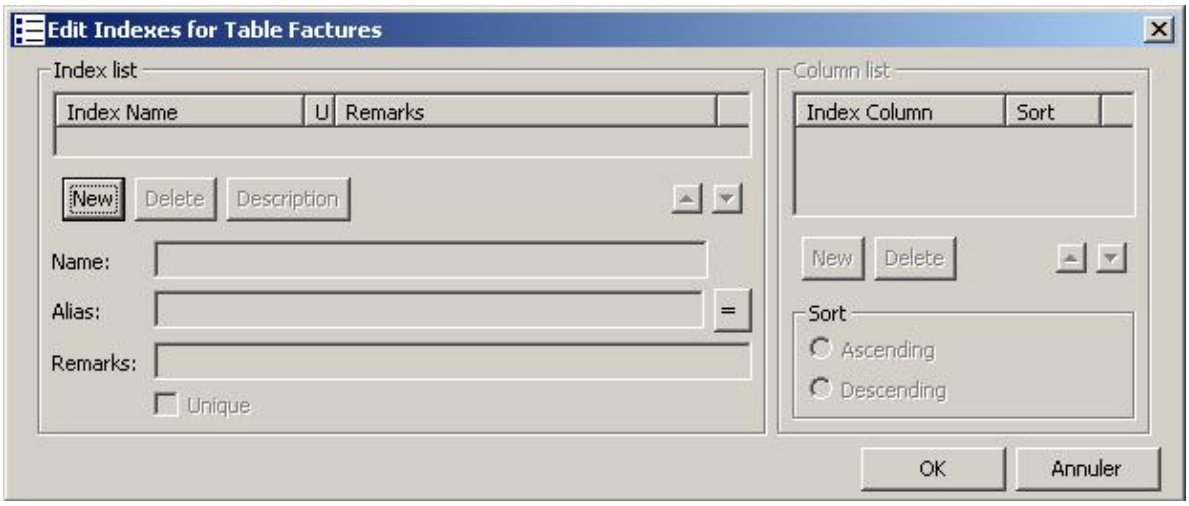
Il suffit alors de saisir les données du nouveau champ. L'opération doit être répétée pour chaque table du modèle. Exemple :



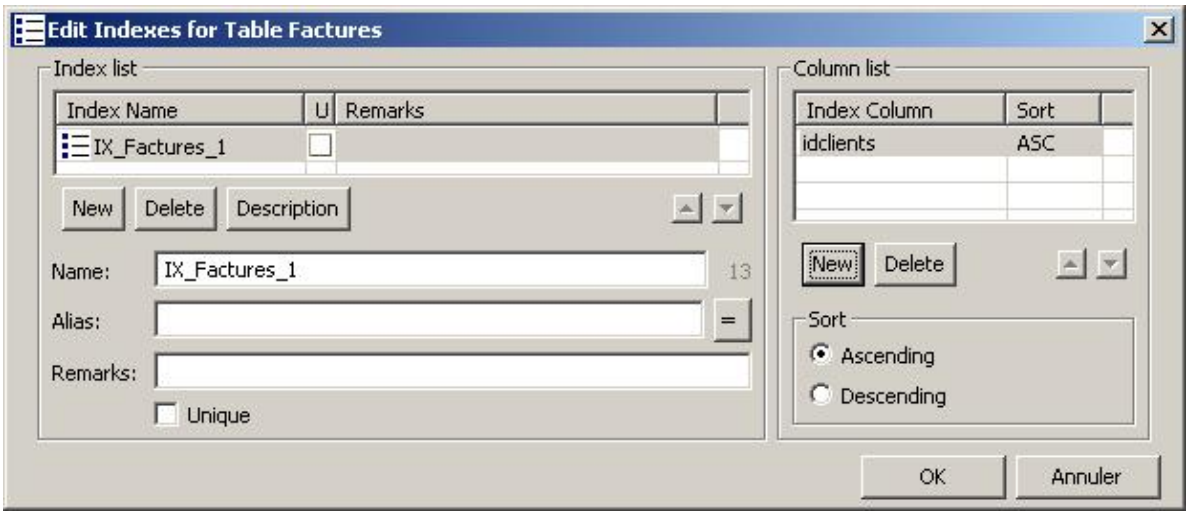
La vue "Structure" affiche sous une forme arborescente la structure du modèle avec les différents éléments qui la compose.



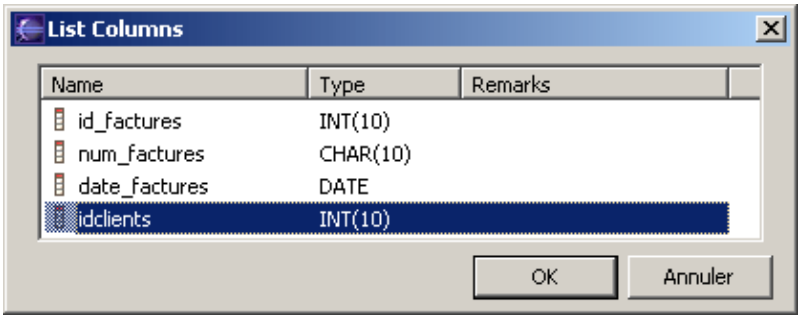
Il est possible de créer des index en utilisant l'option « Edit Table Indexes » du menu contextuel.



Pour ajouter un nouvel index, il faut cliquer sur le bouton « New »

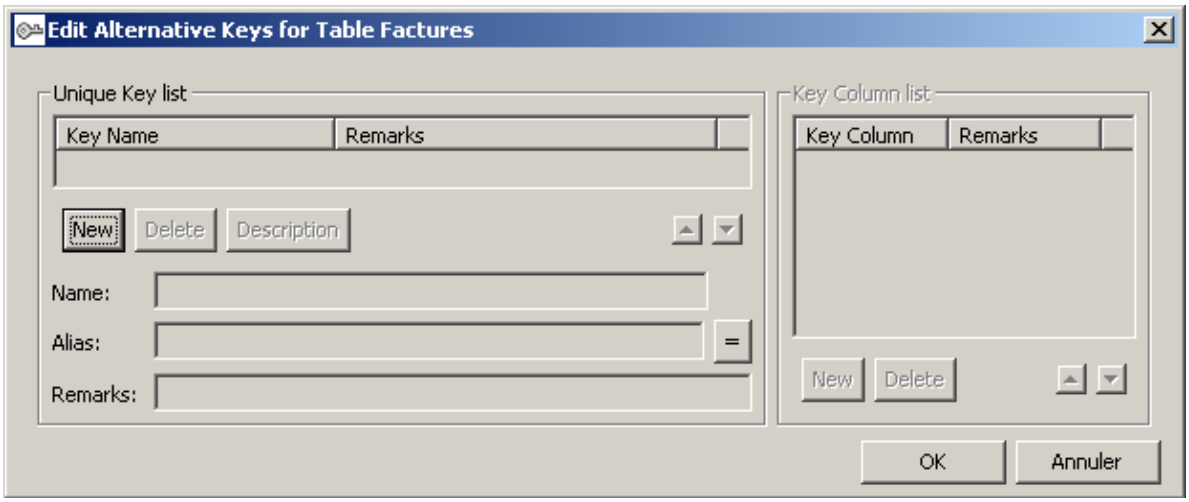


Pour ajouter un champ dans l'index, il faut cliquer sur le bouton « New » dans « Key Column List »

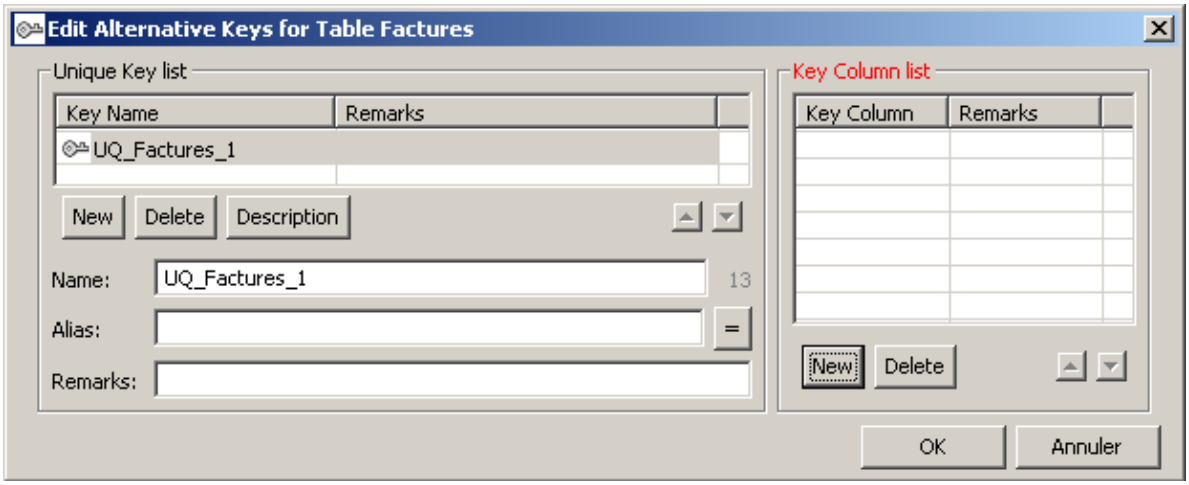


Sélectionnez le champ et cliquez sur le bouton « Ok ». Une fois toute les informations saisies, il suffit de cliquer sur le bouton « OK ». Le nouvel index apparaît dans la vue « Structure ».

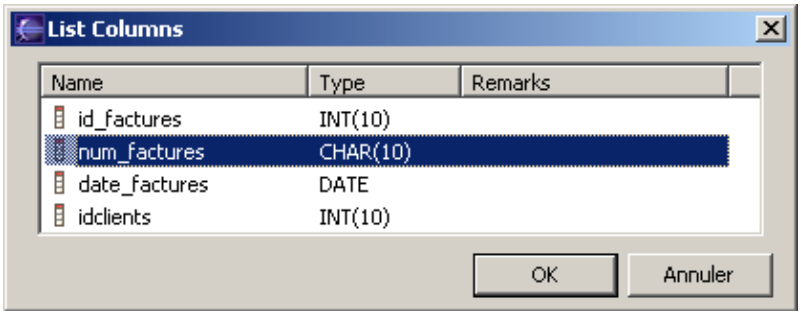
Il est possible de définir des index uniques en utilisant l'option « Edit Table Unique Keys » du menu contextuel.



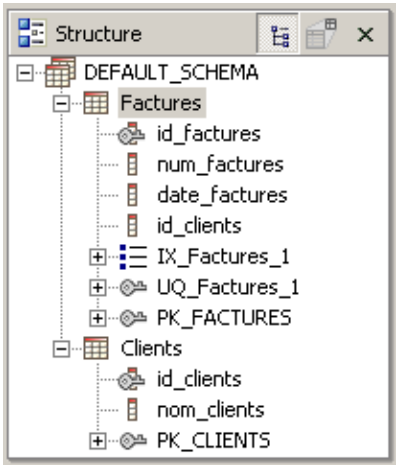
Pour ajouter un nouvel index, il faut cliquer sur le bouton « New »



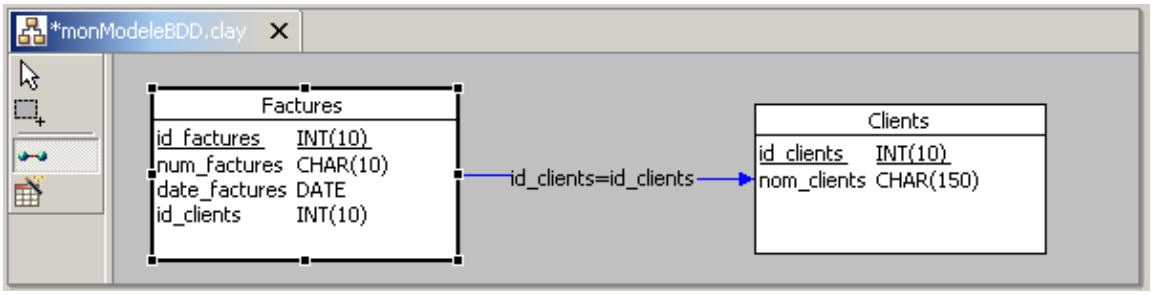
Pour ajouter un champ dans l'index, il faut cliquer sur le bouton « New » dans « Key Column List ».



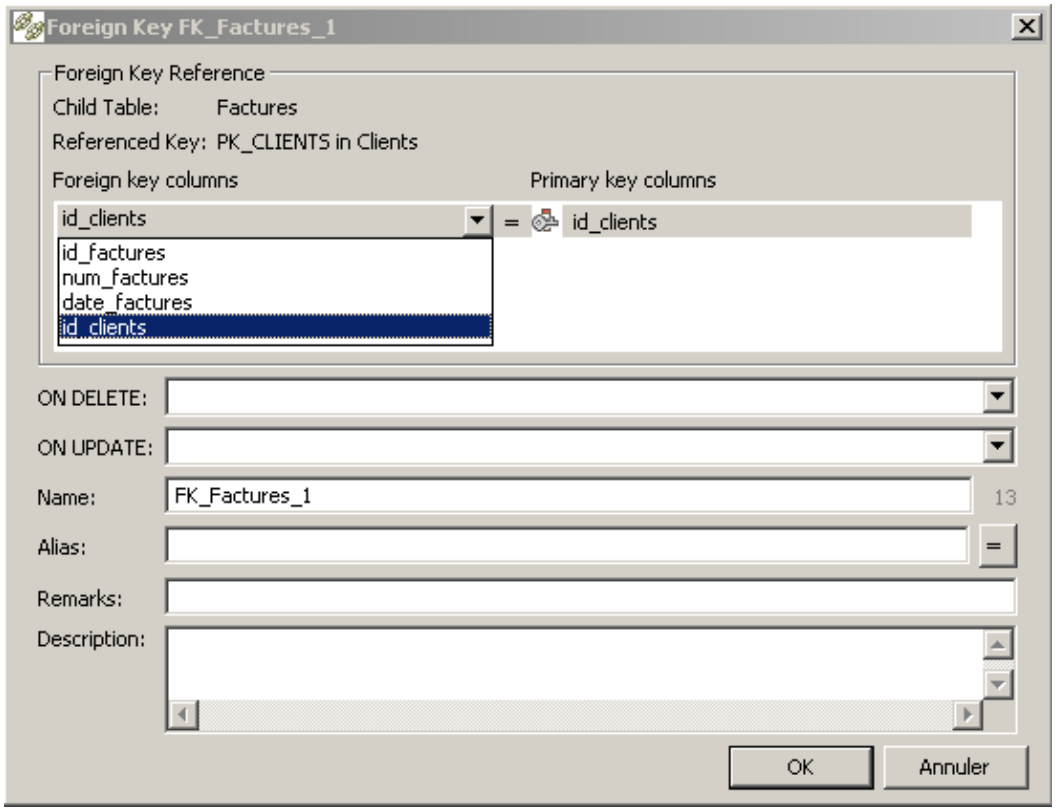
Sélectionnez le champ et cliquez sur le bouton « Ok ». Une fois toutes les informations saisies, il suffit de cliquer sur le bouton « OK ». Le nouvel index apparaît dans la vue « Structure »



Le bouton  permet de créer des relations entre deux tables. Pour cela, il faut cliquer sur le bouton, cliquer sur la table contenant la clé étrangère puis cliquer sur la table référence.



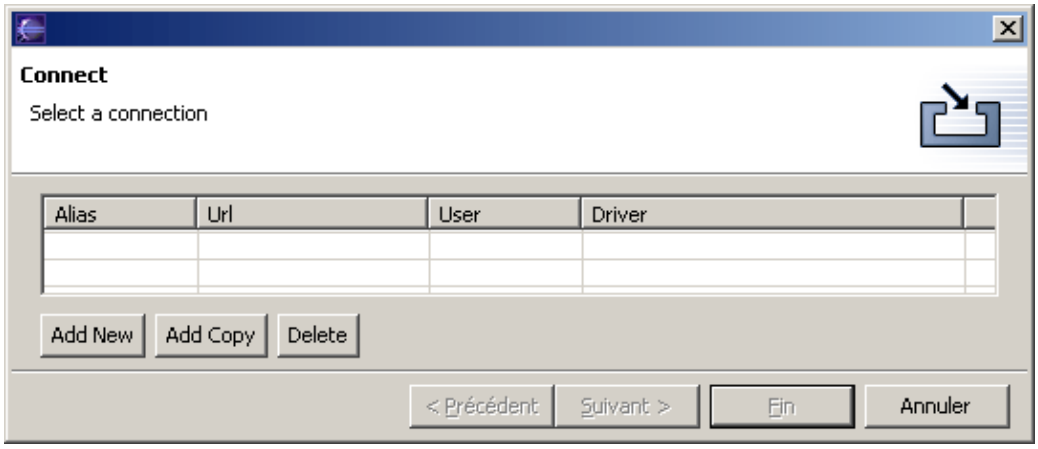
Le lien entre les deux tables apparait sous la forme d'une flèche bleu entre les deux tables. Le menu contextuel « Edit Foreign Key » permet de modifier la clé étrangère.



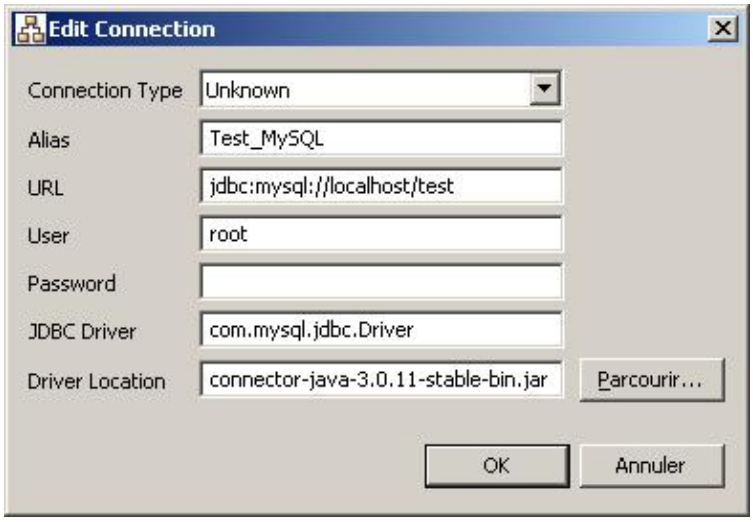
24.4.3. Rétro−conception d'un modèle

L'option « Reverse Engineer Database »  du menu contextuel de l'éditeur permet de rétro−concevoir un modèle à partir d'une base de données existante.

Un assistant permet de sélectionner ou de saisir les informations concernant la connexion à la base de données.



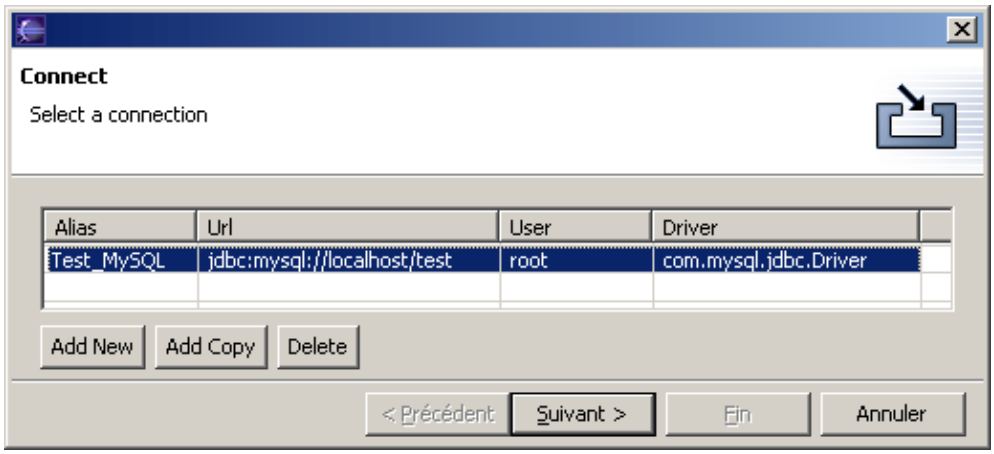
Cliquez sur le bouton « Add New »



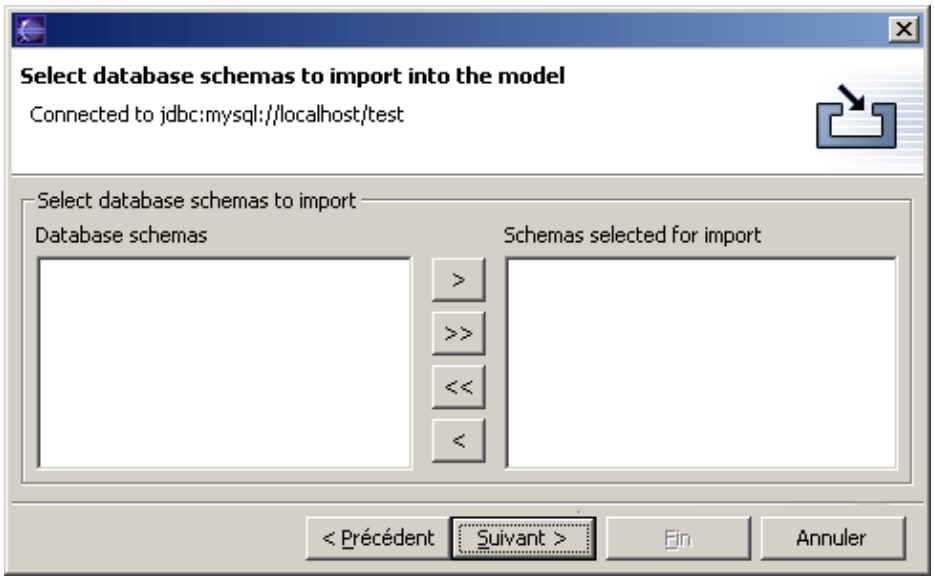
Une boite de dialogue permet de saisir les informations nécessaires à la connexion :

- le type de connexion peut être choisi dans la liste : si celui ci n'apparaît pas dans la liste, alors il faut sélectionner « unknown »
- l'alias est un nom qui permet de reconnaître la connexion
- user et password sont le nom de l'utilisateur et son mot de passe utilisé pour la connexion
- JDBC Driver est le nom pleinement qualifié de la classe contenant le pilote à utiliser
- Driver Location est le chemin du fichier jar contenant la classe du pilote

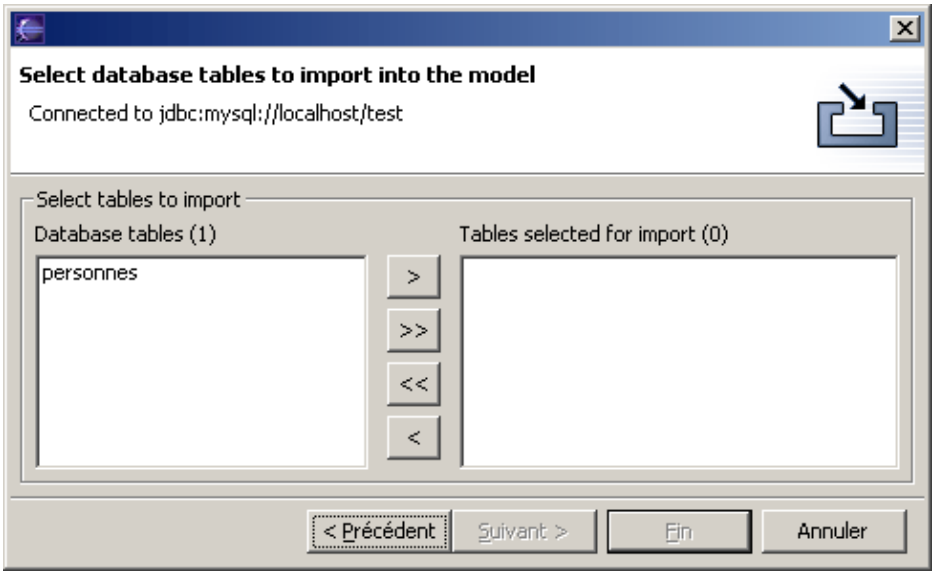
Cliquez sur le bouton « Ok ».



Sélectionnez l'alias et cliquez sur le bouton « Suivant ». La connexion à la base de données s'effectue et l'assistant demande la sélection du schéma. Dans l'exemple de cette section, il n'y en a pas.

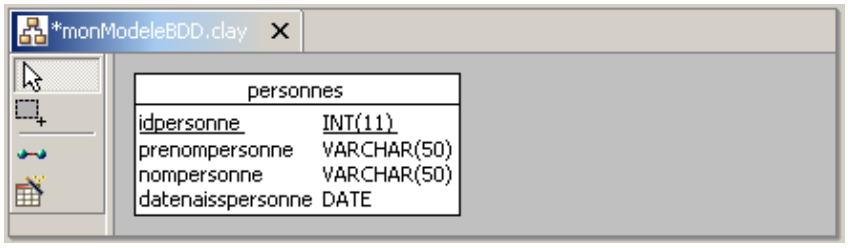


Cliquez sur le bouton « Suivant ». L'assistant demande de sélectionner les tables qui doivent être intégrées dans le modèle.



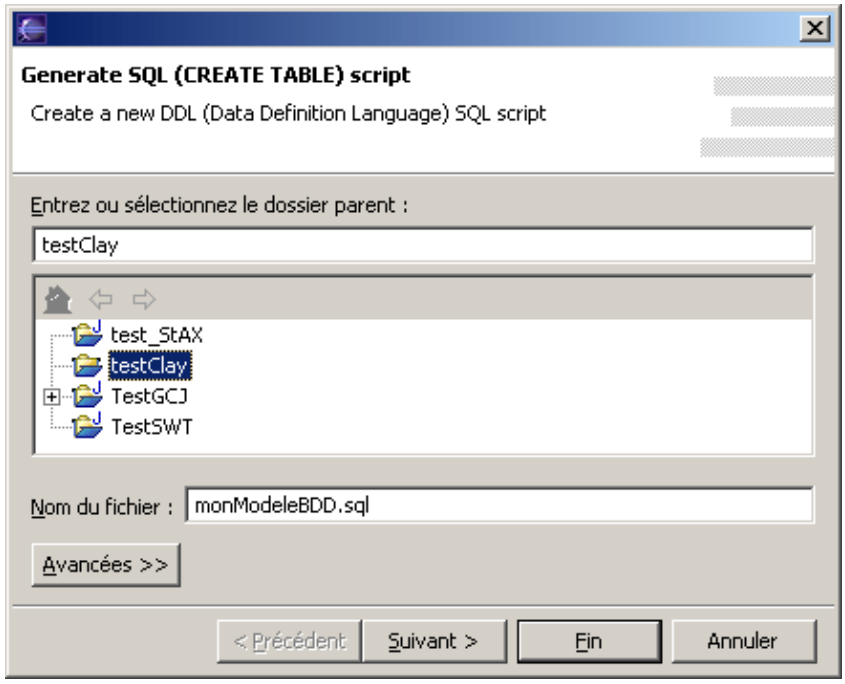
Pour sélectionner une table, il suffit de la sélectionner dans la liste de gauche et de la faire basculer dans la liste de droite en utilisant le bouton « > ». Une fois la sélection terminée, il suffit de cliquer sur le bouton « Fin ».

L'assistant traite la demande et affiche le résultat des traitements dans l'éditeur.

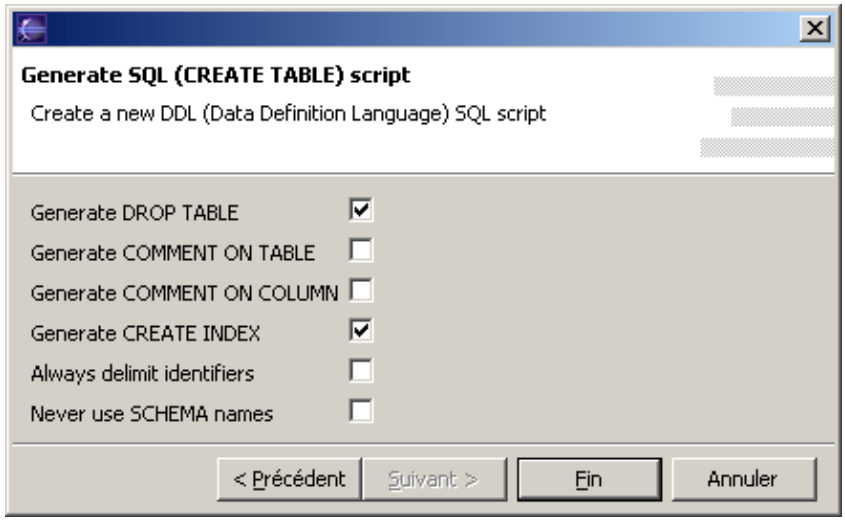


24.4.4. Génération du DDL

L'option « Generate SQL (CREATE TABLE) script » permet de demander la création d'un fichier contenant les ordres de création des éléments de la base de données grâce à un assistant.



Il faut sélectionner le dossier qui va contenir le fichier, saisir le nom de ce fichier et cliquer sur le bouton « Suivant ».



La page suivante permet de sélectionner les types d'ordres qui seront générés. Un clic sur le bouton « Fin » permet de générer le fichier et de l'ouvrir dans l'éditeur associé au fichier .sql.

Exemple :

```
DROP TABLE IF EXISTS Factures;
  DROP TABLE IF EXISTS Clients;
  CREATE TABLE Clients (
      id_clients INT(10) NOT NULL AUTO_INCREMENT
    , nom_clients CHAR(150)
    , PRIMARY KEY (id_clients)
  )TYPE=InnoDB;
  CREATE TABLE Factures (
      id_factures INT(10) NOT NULL AUTO_INCREMENT
    , num_factures CHAR(10)
    , date_factures DATE
    , id_clients INT(10)
    , UNIQUE UQ_Factures_1 (num_factures)
    , PRIMARY KEY (id_factures)
    , INDEX (id_clients)
    , FOREIGN KEYFK_Factures_1 (id_clients)
        REFERENCESClients (id_clients)
  ) TYPE=InnoDB;

  CREATE INDEX IX_Factures_1 ON Factures (id_clients ASC);
```

25. Eclipse et Hibernate

25.1. Le plug−in Hibernate Synchronizer

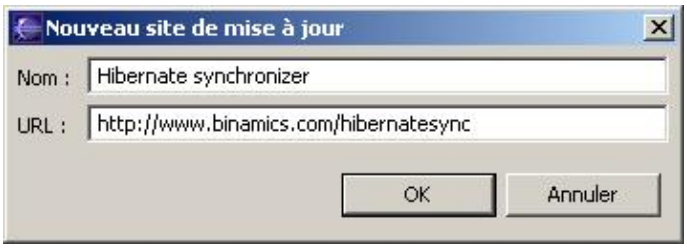
Le plug−in Hibernate Synchronizer permet la génération de code utilisant le framework Hibernate. Il permet aussi de re−générer ce code lorsqu'un fichier de mapping est modifié.

Le site du plug−in est à l'url : http://hibernatesynch.sourceforge.net/

	Version utilisée dans cette section
Eclipse	3.0.1
J2SE	1.4.2_03
Hibernate Synchronizer	2.3.1.

25.1.1. Installation

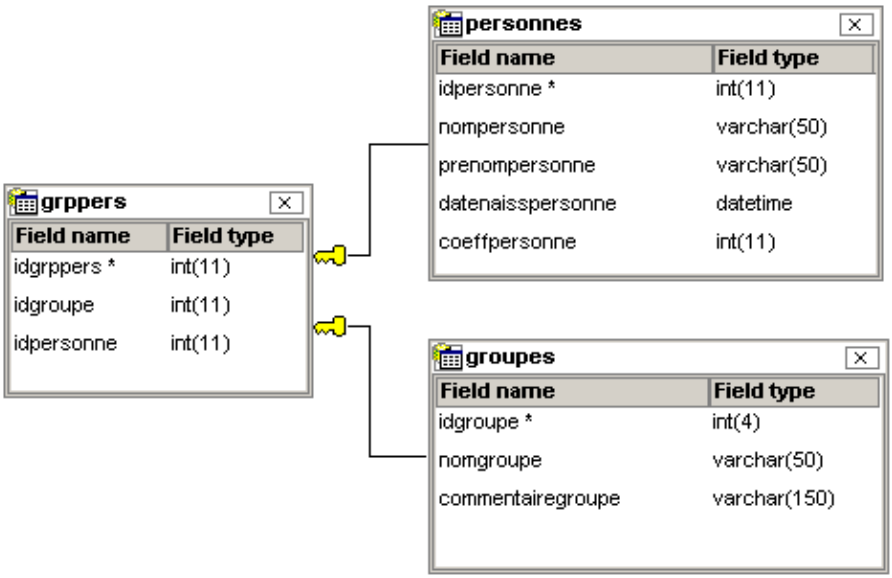
Le plus simple est d'utiliser la fonctionnalité de mise à jour proposée par l'option "Rechercher et installer" du menu "Aide". Cliquez sur le bouton "Rechercher les nouveaux dispositifs à installer" sur le bouton "Nouveau site distant", saisissez les informations ci dessous et suivez les instructions pour réaliser le téléchargement et l'installation.



25.1.2. La base de données utilisée

La base de données utilisées dans cette section contient trois tables :

- une table qui contient une liste de personnes
- une table qui contient une liste de groupes
- une table qui associe une personne à un groupe



Exemple : le DDL de la base de données

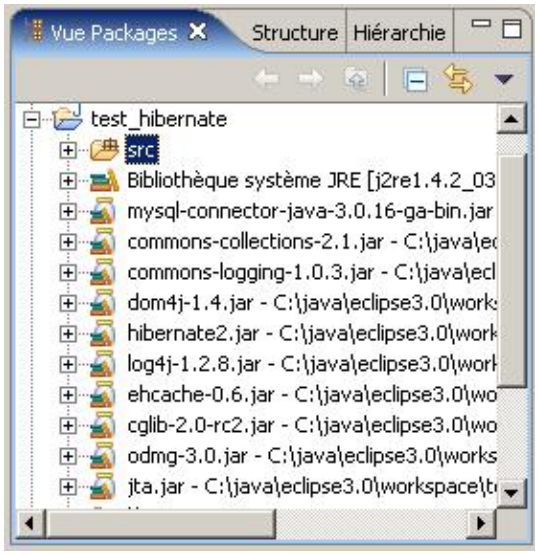
```
drop table `grppers`;
drop table `groupes`;
drop table `personnes`;

#
# Structure for the `groupes` table :
#
CREATE TABLE `groupes` (
  `idgroupe` int(4) NOT NULL auto_increment,
  `nomgroupe` varchar(50) default NULL,
  `commentairegroupe` varchar(150) default NULL,
  PRIMARY KEY  (`idgroupe`),
  UNIQUE KEY `idgroupe` (`idgroupe`)
) TYPE=InnoDB;

#
# Structure for the `personnes` table :
#
CREATE TABLE `personnes` (
  `idpersonne` int(11) NOT NULL auto_increment,
  `nompersonne` varchar(50) default NULL,
  `prenompersonne` varchar(50) default NULL,
  `datenaisspersonne` datetime default NULL,
  `coeffpersonne` int(11) default NULL,
  PRIMARY KEY  (`idpersonne`),
  UNIQUE KEY `idpersonne` (`idpersonne`)
) TYPE=InnoDB;

#
# Structure for the `grppers` table :
#
CREATE TABLE `grppers` (
  `idgrppers` int(11) NOT NULL auto_increment,
  `idgroupe` int(11) default NULL,
  `idpersonne` int(11) default NULL,
  PRIMARY KEY  (`idgrppers`),
  UNIQUE KEY `idgrppers` (`idgrppers`),
  KEY `idgroupe` (`idgroupe`),
  KEY `idpersonne` (`idpersonne`),
  CONSTRAINT `0_48` FOREIGN KEY (`idpersonne`) REFERENCES `personnes` (`idpersonne`),
  CONSTRAINT `0_45` FOREIGN KEY (`idgroupe`) REFERENCES `groupes` (`idgroupe`)
) TYPE=InnoDB;

INSERT INTO `groupes` (`idgroupe`, `nomgroupe`, `commentairegroupe`) VALUES
  (1,'groupe 1',NULL),
```

```
  (2,'groupe 2',NULL);

INSERT INTO `grppers` (`idgrppers`, `idgroupe`, `idpersonne`) VALUES
  (1,1,1),
  (2,2,2),
  (3,2,3),
  (4,1,4),
  (5,1,5);

INSERT INTO `personnes` (`idpersonne`, `nompersonne`, `prenompersonne`,
                         `datenaisspersonne`, `coeffpersonne`) VALUES
  (1,'nom1','prenom1','1967-01-06',123),
  (2,'nom2','prenom2','1973-08-11',34),
  (3,'nom3','prenom3','1956-04-28',145),
  (4,'nom4','prenom4','1980-12-02',23),
  (5,'nom5','prenom5','1966-10-13',119);
```

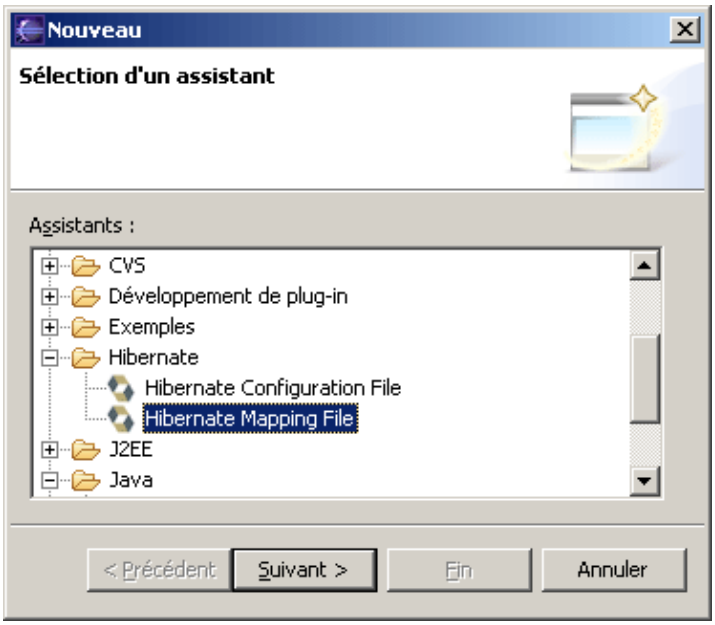
25.1.3. Création des fichiers de mapping

Créer un nouveau projet dont les sources et les binaires sont séparés et ajouter les fichiers mm.mysql-2.0.14-bin.jar et hibernate2.jar dans les bibliothèques.

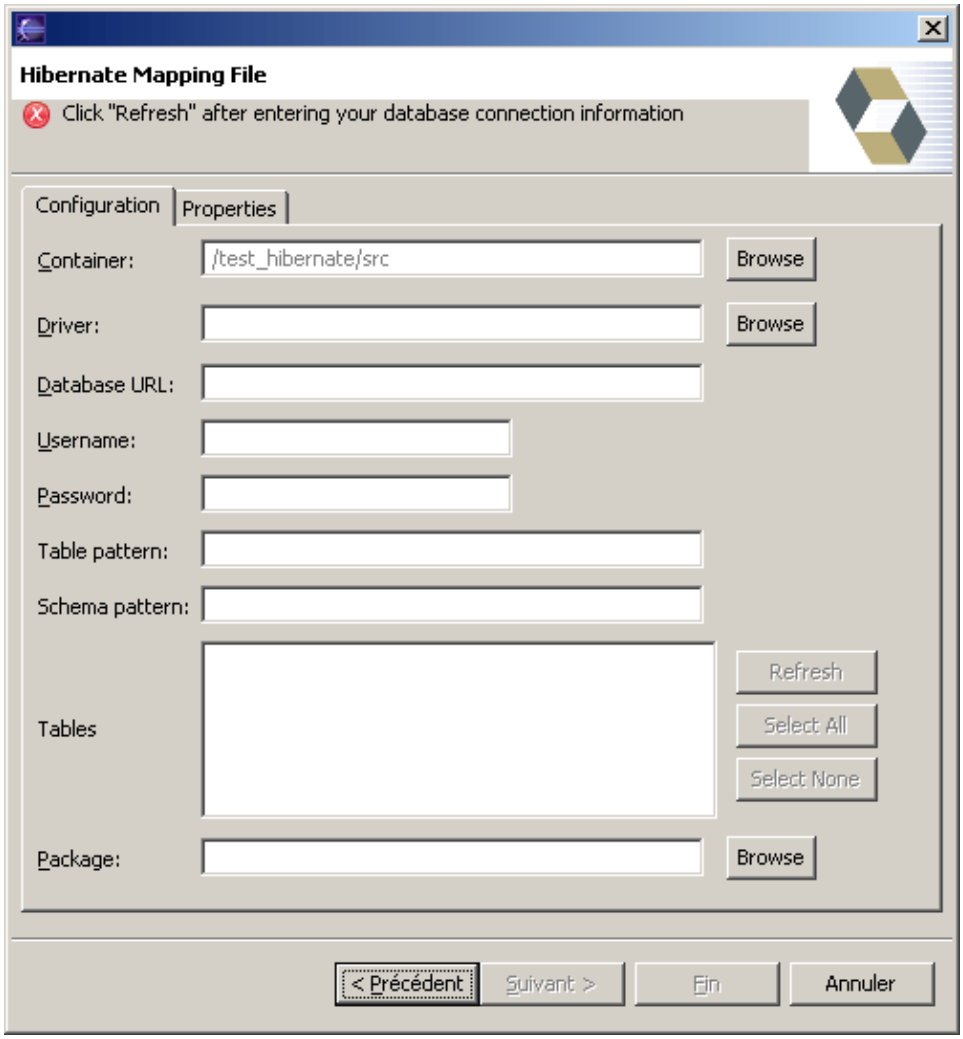
Il est aussi obligatoire d'ajouter toutes les bibliothèques nécessaires à Hibernate lors de son exécution.



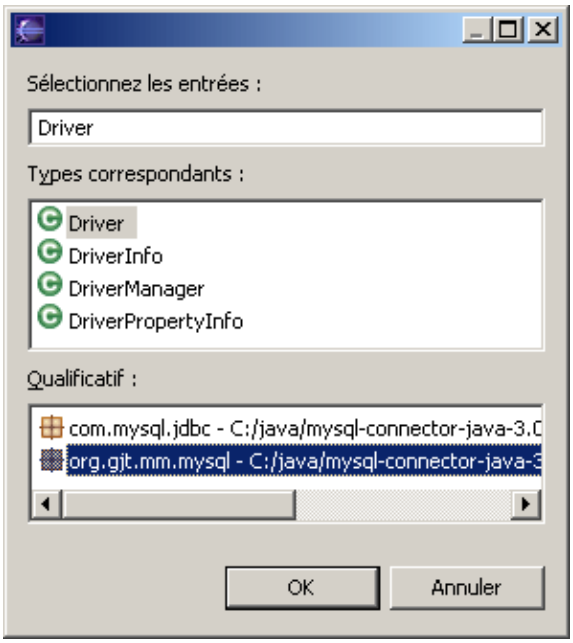
Il faut ensuite créer une entité de type « Hibernate / Hibernate Mapping File » dans le répertoire src du projet.



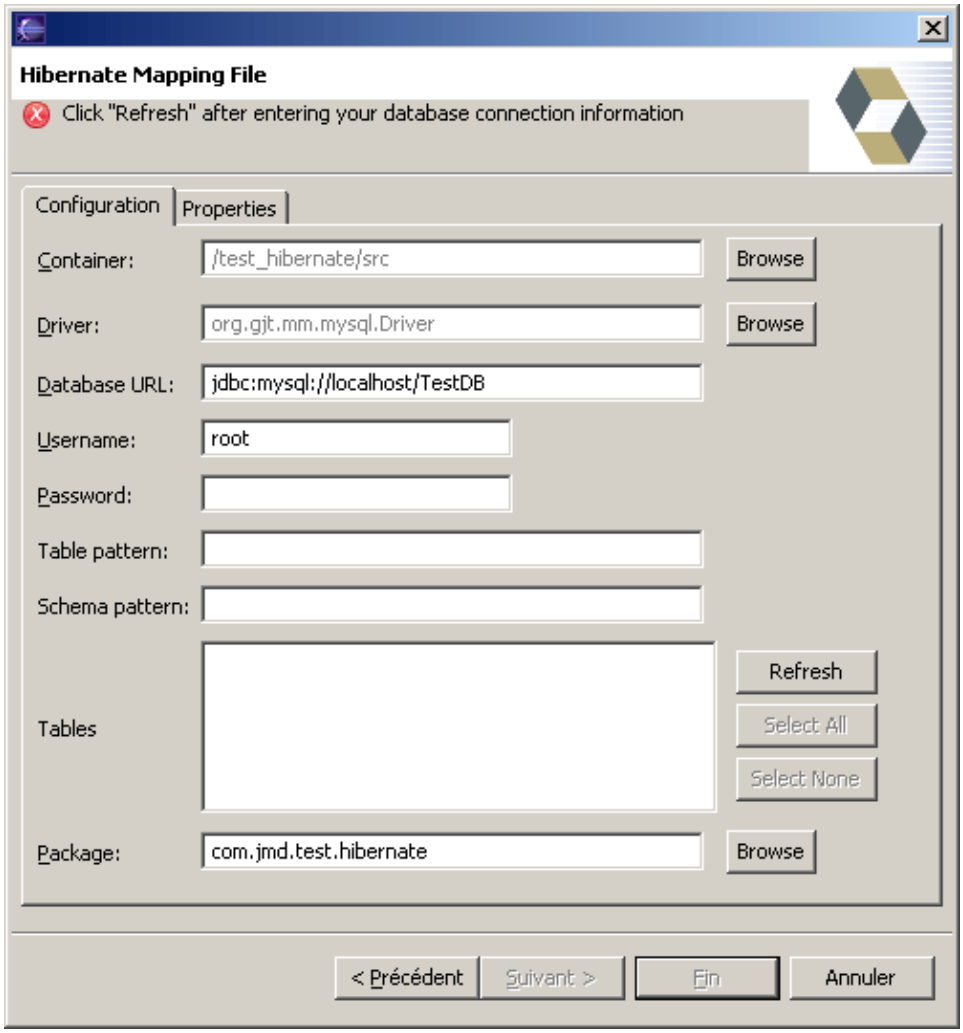
Cliquez sur le bouton « Suivant ». La page suivante permet de saisir les informations sur la base de données et sa connexion.



Cliquez sur le bouton « Browse » en face de « Driver ».



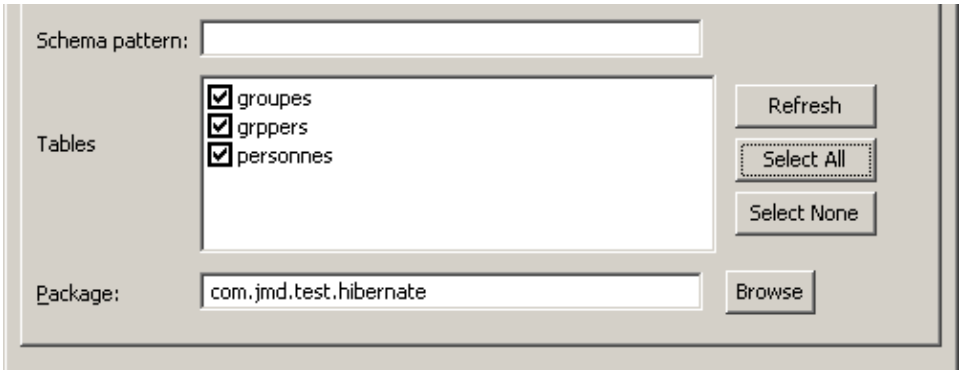
Sélectionnez la classe org.gjt.mm.mysql.Driver et cliquez sur le bouton « OK ».



Il faut ensuite saisir les informations concernant la connexion à la base de données et cliquer sur le bouton « Refresh ».

Si les paramètres concernant la connexion sont corrects alors la liste des tables est affichée. Il suffit alors de sélectionner la ou les tables désirées.

Enfin, il faut saisir le nom du package qui va contenir les fichiers générés.



Cliquez sur le bouton « Fin ». Trois fichiers sont générés : Groupes.hbm, Grppers.hbm et Personnes.hbm

Exemple : le fichier Personnes.hbm

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC
            "-//Hibernate/Hibernate Mapping DTD//EN"
            "http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd" >
<hibernate-mapping package="com.jmd.test.hibernate">
  <class name="Personnes" table="personnes">

    <id
      column="idpersonne"
      name="Idpersonne"
      type="integer"
    >
      <generator class="vm" />
    </id>

    <property
      column="datenaisspersonne"
      length="19"
      name="Datenaisspersonne"
      not-null="false"
      type="timestamp"
     />

    <property
      column="prenompersonne"
      length="50"
      name="Prenompersonne"
      not-null="false"
      type="string"
     />

    <property
      column="coeffpersonne"
      length="11"
      name="Coeffpersonne"
      not-null="false"
      type="integer"
     />

    <property
      column="nompersonne"
      length="50"
       name="Nompersonne"
      not-null="false"
      type="string"
     />
  </class>
</hibernate-mapping>
```

Exemple : le fichier Groupes.hbm

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC
            "-//Hibernate/Hibernate Mapping DTD//EN"
            "http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd" >
<hibernate-mapping package="com.jmd.test.hibernate">
 <class name="Groupes" table="groupes">
   <id
     column="idgroupe"
     name="Idgroupe"
     type="integer"
   >

     <generator class="vm" />
   </id>

   <property
     column="nomgroupe"
     length="50"
     name="Nomgroupe"
     not-null="false"
     type="string"
    />

   <property
     column="commentairegroupe"
     length="150"
     name="Commentairegroupe"
     not-null="false"
     type="string"
    />
 </class>
</hibernate-mapping>
```

Exemple : le fichier Grppers.hbm

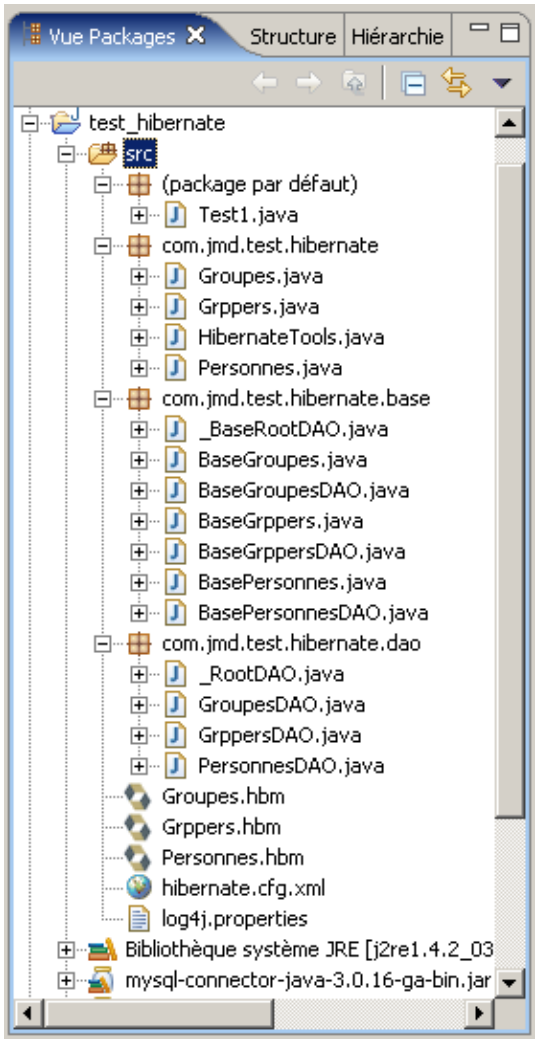
```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC
            "-//Hibernate/Hibernate Mapping DTD//EN"
            "http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd" >
<hibernate-mapping package="com.jmd.test.hibernate">
  <class name="Grppers" table="grppers">
    <id
      column="idgrppers"
      name="Idgrppers"
      type="integer"
    >
      <generator class="vm" />
    </id>
    <many-to-one
      class="Groupes"
      name="Idgroupe"
      not-null="true"
    >
      <column name="idgroupe" />
    </many-to-one>
    <many-to-one
      class="Personnes"
      name="Idpersonne"
      not-null="true"
    >
      <column name="idpersonne" />
    </many-to-one>
  </class>
</hibernate-mapping>
```

25.1.4. Génération des classes Java

La génération des classes correspondantes pour chaque fichier .hbm peut être demandée de deux façons :

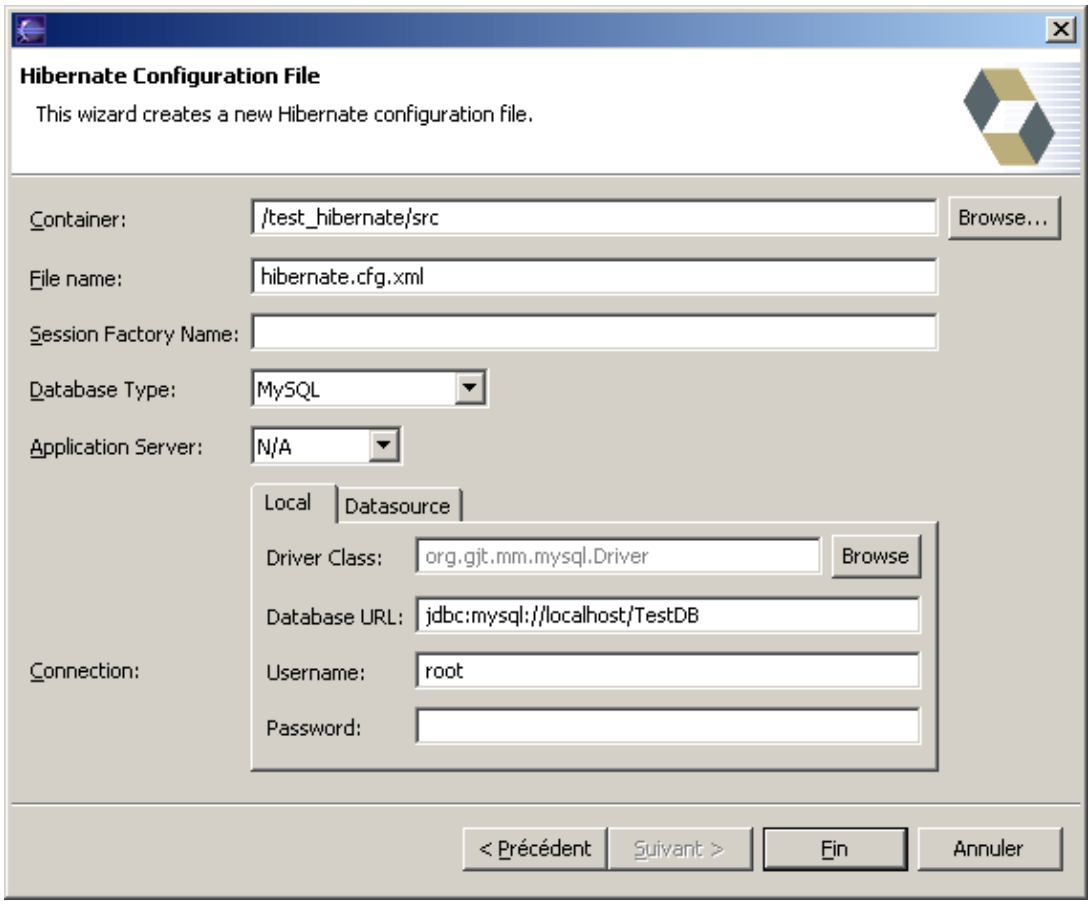
- en sauvegardant un fichier .hbm modifié dans l'éditeur
- en utilisant l'option « Hibernate Synchronizer / Synchronize Files » du menu contextuel d'un fichier .hbm dans la vue « Vue packages »

Cette génération va créer pour chaque fichier .hbm plusieurs classes.



25.1.5. Création du fichier de configuration

Il faut créer une nouvelle entité de type « Hibernate / Hibernate Configuration File » dans le répertoire src. Une boite de dialogue permet de saisir les informations qui seront insérées dans le fichier de configuration d'Hibernate.



Une fois les informations saisies, cliquez sur le bouton « Fin »

Il faut rajouter dans ce fichier tous les fichiers de mapping qui seront utilisés.

Exemple :

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-configuration
    PUBLIC "-//Hibernate/Hibernate Configuration DTD//EN"
    "http://hibernate.sourceforge.net/hibernate-configuration-2.0.dtd">
<hibernate-configuration>
  <session-factory >
    <!-- local connection properties -->
    <property name="hibernate.connection.url">jdbc:mysql://localhost/TestDB</property>
    <property name="hibernate.connection.driver_class">org.gjt.mm.mysql.Driver</property>
    <property name="hibernate.connection.username">root</property>
    <property name="hibernate.connection.password"></property>
    <!-- property name="hibernate.connection.pool_size"></property -->

    <!-- dialect for MySQL -->
    <property name="dialect">net.sf.hibernate.dialect.MySQLDialect</property>
    <property name="hibernate.show_sql">false</property>
    <property name="hibernate.use_outer_join">true</property>

    <mapping resource="Personnes.hbm"/>
    <mapping resource="Grppers.hbm"/>
    <mapping resource="Groupes.hbm"/>
  </session-factory>
</hibernate-configuration>
```

25.1.6. La mise en oeuvre des classes générées

Les différentes classes qui vont mettre en oeuvre les classes générées vont utiliser une classe utilitaire proposée dans la documentation de Hibernate pour configurer et obtenir une session.

Exemple :

```
package com.jmd.test.hibernate;

import net.sf.hibernate.*;
import net.sf.hibernate.cfg.*;

public class HibernateUtil {

  private static final SessionFactory sessionFactory;

  static {
    try {
      sessionFactory = new Configuration().configure()
                              .buildSessionFactory();
    } catch (HibernateException ex) {
      throw new RuntimeException("Exception building SessionFactory: "
            + ex.getMessage(), ex);
    }
  }

  public static final ThreadLocal session = new ThreadLocal();

  public static Session currentSession() throws HibernateException {
    Session s = (Session) session.get();
    // Open a new Session, if this Thread has none yet
    if (s == null) {
      s = sessionFactory.openSession();
      session.set(s);
    }
    return s;
  }

  public static void closeSession() throws HibernateException {
    Session s = (Session) session.get();
    session.set(null);
    if (s != null)
      s.close();
  }
}
```

Le premier exemple va simplement afficher le nom de toutes les personnes.

Exemple :

```
import java.util.*;
import net.sf.hibernate.*;
import com.jmd.test.hibernate.*;

public class Test1 {

  public static void main(String[] args) {
    try {
      Session session = HibernateUtil.currentSession();

      List list = session.find("from Personnes ");
      Iterator it = list.iterator();

      while(it.hasNext())
      {
```

```
        Personnes personne = (Personnes)it.next();
        System.out.println(personne.getNompersonne());
      }

      HibernateUtil.closeSession();
    } catch (HibernateException e) {
      e.printStackTrace();
    }
  }
}
```

Exemple :

```
Résultat :
nom1
nom2
nom3
nom4
nom5
```

Voici le même exemple utilisant les classes générés mettant en oeuvre le motif de conception DAO.

Exemple :

```
import java.util.Iterator;
import java.util.List;
import net.sf.hibernate.HibernateException;
import com.jmd.test.hibernate.Personnes;
import com.jmd.test.hibernate.dao.PersonnesDAO;
import com.jmd.test.hibernate.dao._RootDAO;

public class Test1DAO {
  public static void main(String[] args) {
    try {
      _RootDAO.initialize();
      PersonnesDAO dao = new PersonnesDAO();

      List liste = dao.findAll();
      Iterator it = liste.iterator();

      while (it.hasNext()) {
        Personnes personne = (Personnes) it.next();
        System.out.println(personne.getNompersonne());
      }
    } catch (HibernateException e) {
      e.printStackTrace();
    }
  }
}
```

Le second exemple va retrouver un groupe, créer une nouvelle personne et l'ajouter au groupe trouvé.

Exemple :

```
import java.util.*;
import net.sf.hibernate.*;
import com.jmd.test.hibernate.*;

public class Test2 {
  public static void main(String[] args) {
    try {
      Session session = HibernateUtil.currentSession();
      Transaction tx = session.beginTransaction();
```

```
      Personnes personne = new Personnes();
      personne.setNompersonne("nom6");
      personne.setPrenompersonne("prenom6");
      personne.setCoeffpersonne(new Integer(46));
      personne.setDatenaisspersonne(new Date());
      session.save(personne);

      Groupes groupe = (Groupes) session.load(Groupes.class, new Integer(1));
      Grppers grppres = new Grppers();
      grppres.setIdpersonne(personne);
      grppres.setIdgroupe(groupe);
      session.save(grppres);

      tx.commit();
      HibernateUtil.closeSession();
    } catch (HibernateException e) {
      e.printStackTrace();
    }
  }
}
```

Voici le même exemple utilisant les classes générés mettant en oeuvre le motif de conception DAO.

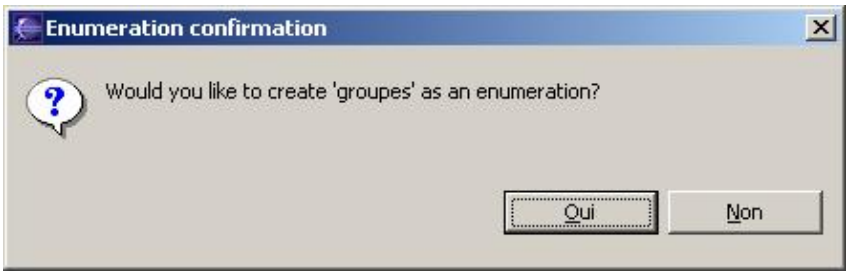
Exemple :

```
import java.util.*;
import net.sf.hibernate.*;
import com.jmd.test.hibernate.*;
import com.jmd.test.hibernate.dao.*;

public class Test2DAO {
  public static void main(String[] args) {
    try {
      _RootDAO.initialize();
      Session session = _RootDAO.createSession();
      Transaction tx = session.beginTransaction();

      Personnes personne = new Personnes();
      personne.setNompersonne("nom7");
      personne.setPrenompersonne("prenom7");
      personne.setCoeffpersonne(new Integer(46));
      personne.setDatenaisspersonne(new Date());

      PersonnesDAO personnesDAO = new PersonnesDAO();
      personnesDAO.save(personne, session);

      GroupesDAO groupesDAO = new GroupesDAO();
      Groupes groupe = groupesDAO.load(new Integer(1),session);
      Grppers grppres = new Grppers();
      grppres.setIdpersonne(personne);
      grppres.setIdgroupe(groupe);

      GrppersDAO grppresDAO = new GrppersDAO();
      grppresDAO.save(grppres, session);

      tx.commit();
    } catch (HibernateException e) {
      e.printStackTrace();
    }
  }
}
```

Si une des tables à traiter ne contient que des données de références (un identifiant et une valeur), alors Hibernate Synchronize propose de créer une classe particulière qui va encapsuler les données non pas de façon dynamique via un accès à la table mais de façon statique.

Dans ce cas, une boîte de dialogue demande si la classe générée doit l'être de façon statique (création comme une énumération)

Exemple : la table groupes ne possède que deux champs (idgroupe et nomgroupe)



En cliquant sur le bouton « Oui » la classe suivante est générée :

Exemple :

```
package com.jmd.test.hibernate;
import java.io.Serializable;
import net.sf.hibernate.PersistentEnum;

/**
 * This class has been automatically generated by Hibernate Synchronizer.
 * For more information or documentation, visit The Hibernate Synchronizer page
 * at http://www.binamics.com/hibernatesync or contact Joe Hudson at joe@binamics.com.
 */
public class Groupes implements Serializable, PersistentEnum {
  public static final Groupes GROUPE_2 = new Groupes(2);
  public static final Groupes GROUPE_1 = new Groupes(1);
  private final int code;

  protected Groupes(int code) {
    this.code = code;
  }

  public int toInt() { return code; }

  public static Groupes fromInt(int code) {
    switch (code) {
      case 2: return GROUPE_2;
      case 1: return GROUPE_1;
      default: throw new RuntimeException("Unknown value: " + code);
    }
  }

  public String toString () {
    switch (code) {
      case 2: return "groupe 2";
      case 1: return "groupe 1";
      default: return "Unknown value";
    }
  }
}
```

26. Eclipse et J2ME

26.1. EclipseME

Le but de ce plug−in est de permettre le développement d'applications J2ME reposant sur MIDP en utilisant un Wireless Toolkit.

Les fonctionnalités proposées par ce plug−in sont :

- Le support de plusieurs Wireless Toolkit
- Un assistant de création de projets de type Midlet Suite
- Un assistant de création de Midlets
- Un éditeur pour les fichiers .jad
- Une compilation incrémentale avec pré−vérification
- Le débogage du code des Midlets
- L'exécution dans les émulateurs fournis avec le WirelessToolkit
- La création d'un package pour les applications J2ME
- La création d'un package obscurci avec Proguard
- Le support du mode « Over The Air »

Le site officiel de ce plug−in est à l'url : http://eclipseme.org/

Un JDK 1.4, une version 3.0.x d'Eclipse et un Wireless Toolkit sont des pré−requis pour pouvoir utiliser EclipseMe.

Cette section va mettre en oeuvre les outils suivants sous Windows :

Outil	Version	Rôle
JDK	1.4.2_04	
Eclipse	3.0.1	IDE
EclipseME	0.75	
Proguard	3.2	
J2ME Wireless Toolkit	2.1	Kit de développement J2ME

26.1.1. Installation

Pour installer le plug−in, téléchargez le fichier eclipseme.feature_0.7.5_site.zip et enregistrez le dans un répertoire du système.

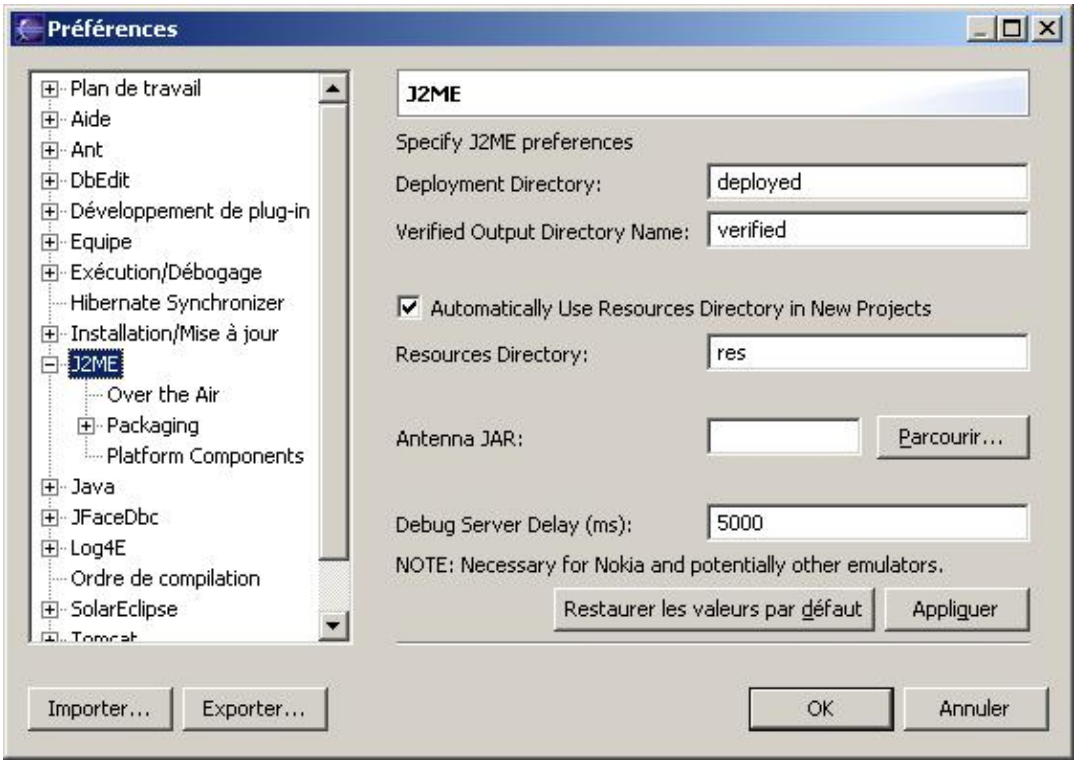
Il suffit alors de suivre les étapes suivantes :

- Utilisez l'option « Mise à jour de logiciels/Rechercher et installer » du menu Aide.
- Sélectionnez « Rechercher les nouveaux dispositifs à installer » et cliquez sur le bouton « Suivant »
- Cliquez sur le bouton « Nouveau site archivé »
- Sélectionnez le fichier et cliquez sur « Ouvrir »
- Dans l'arborescence des sites, sélectionnez eclipseme.feature_0.7.5_site.zip et cliquez sur le bouton « Suivant »
- Sélectionnez les dispositifs « EclipseMe » et « eclipseme.features.siemens »
- Lisez la licence et si vous l'acceptez cliquer sur « J'accepte les termes du contrat » et cliquez sur le bouton « Suivant »
- Cliquez sur le bouton « Fin »
- Lors de l'affichage de la boîte de dialogue « Vérification du dispositif », cliquez sur le bouton « Installer »
- Acceptez de relancer le plan de travail.

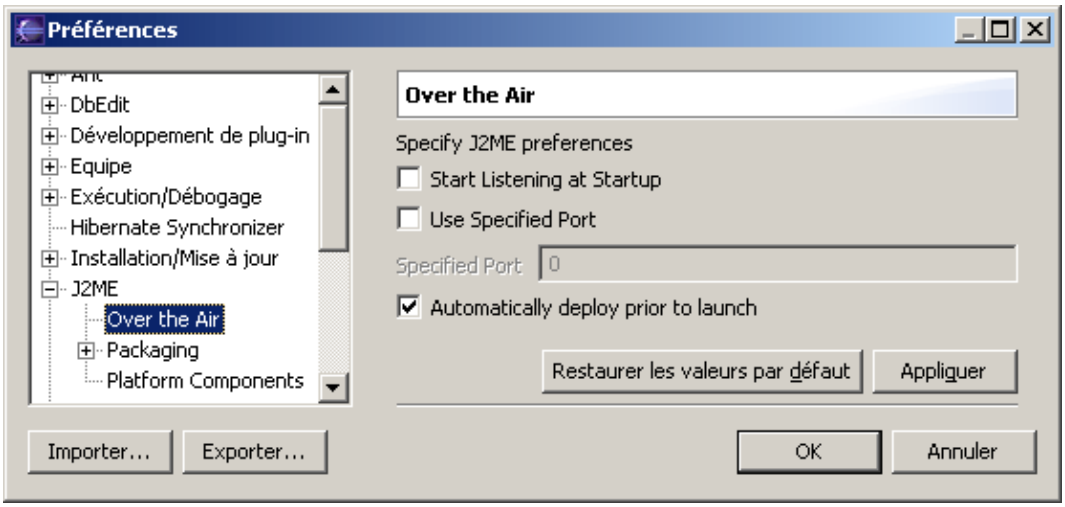
26.1.2. Les préférences du plug−in

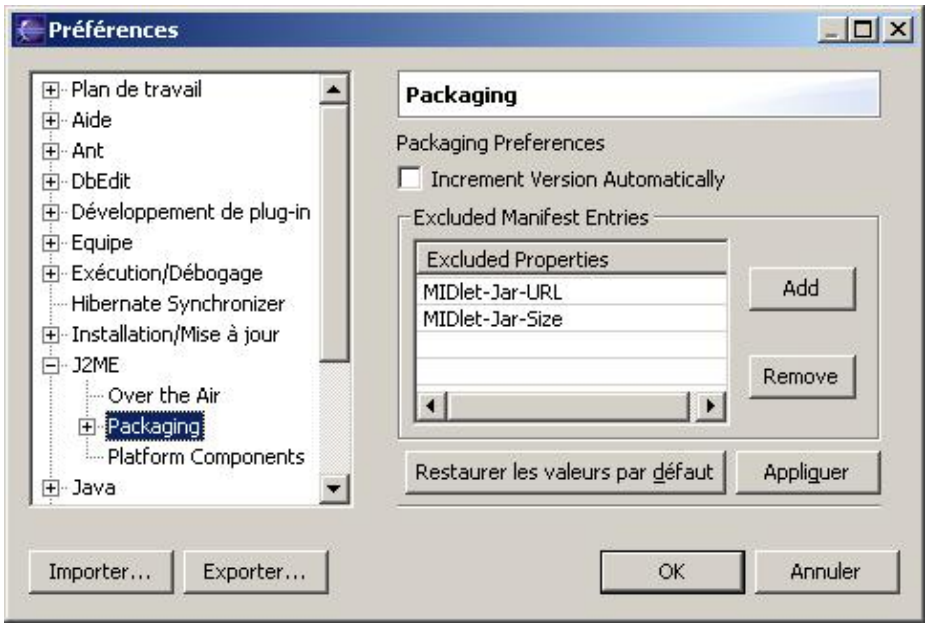
Le plug−in propose plusieurs pages pour gérer les options du plug−in lors de son utilisation.

Ces pages se trouvent sous l'arborescence « J2ME ».

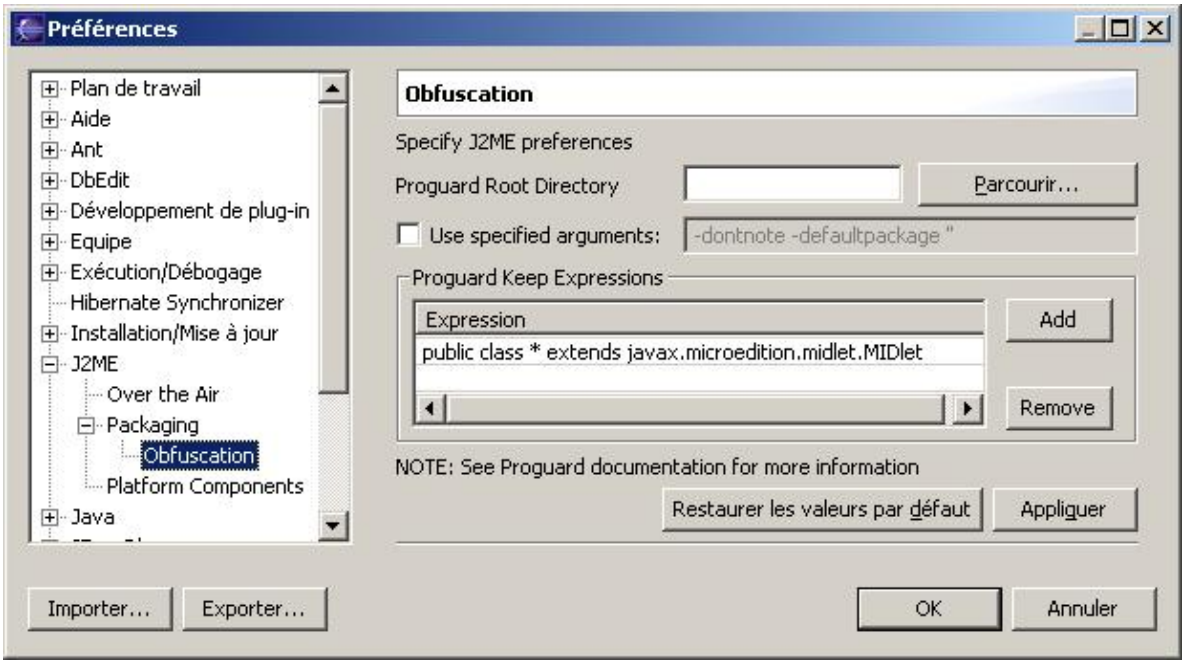


Il est possible de configurer le mode « Over the Air » en sélectionnant « J2ME/Over the Air » dans l'arborescence.



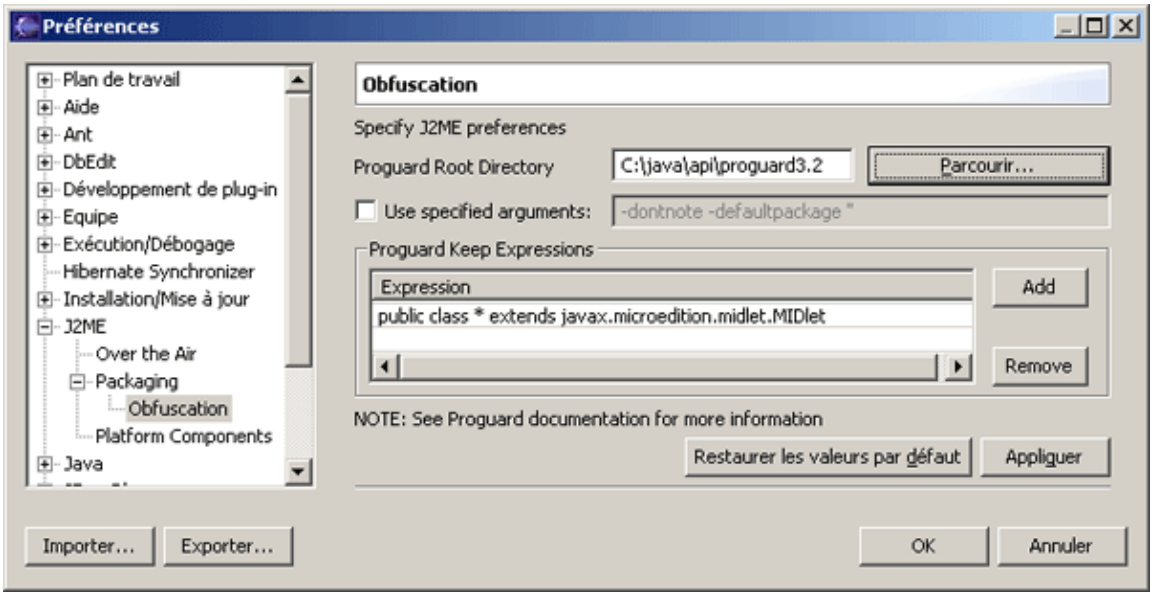


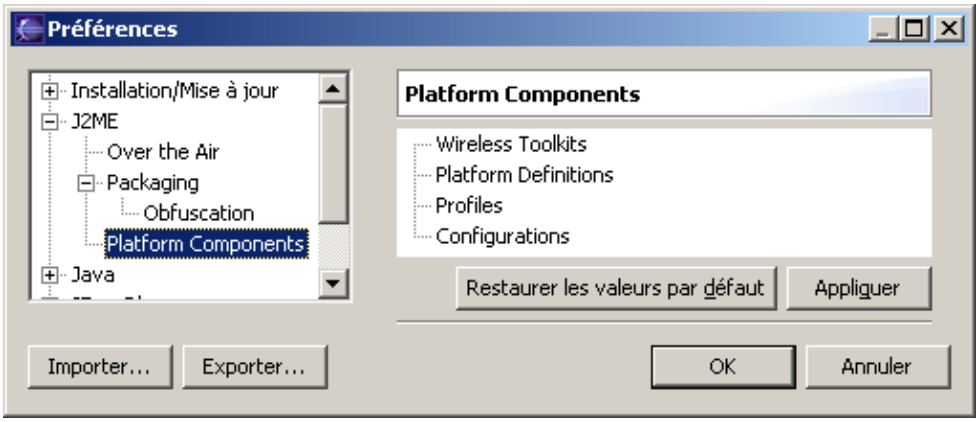
EclipseMe peut travailler avec Proguard, un projet open source qui a pour but de proposer un outil pour rendre le code obscurci. Pour cela, il faut sélectionner J2ME/Packaging/Obfuscation



Pour l'utiliser Proguard, il faut télécharger le fichier proguard3.2.zip sur le site http://proguard.sourceforge.net/ et le décompresser dans un répertoire.

Il suffit alors de sélectionner le répertoire qui contient Proguard en cliquant sur le bouton « Parcourir ».

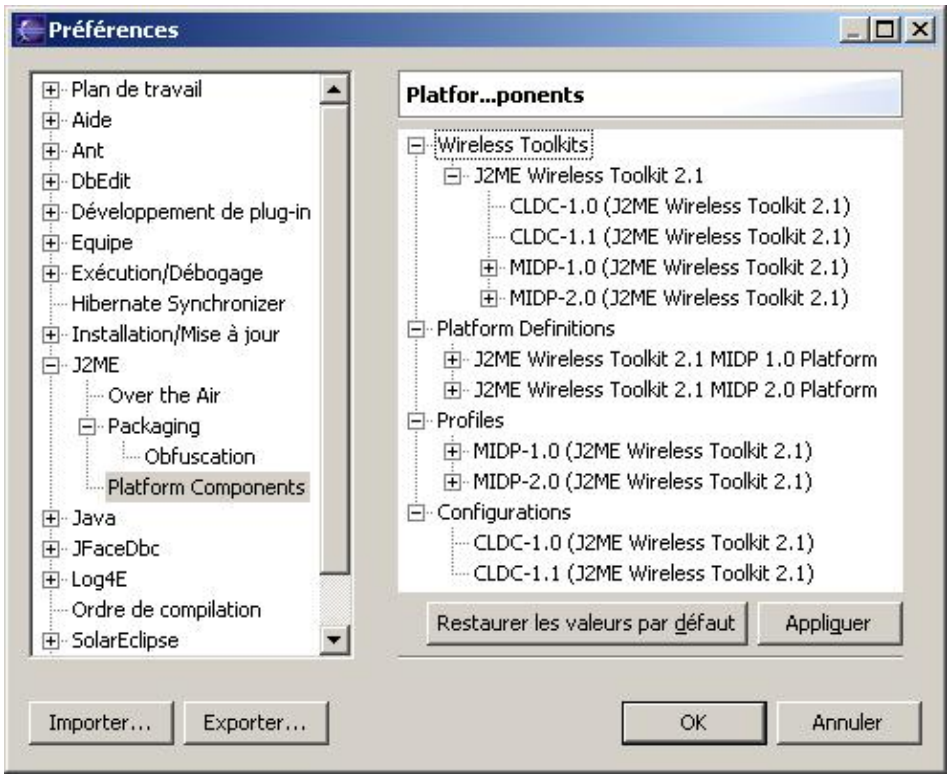




Cet écran permet de configurer le Wireless Toolkit qui sera utilisé avec le plug−in.

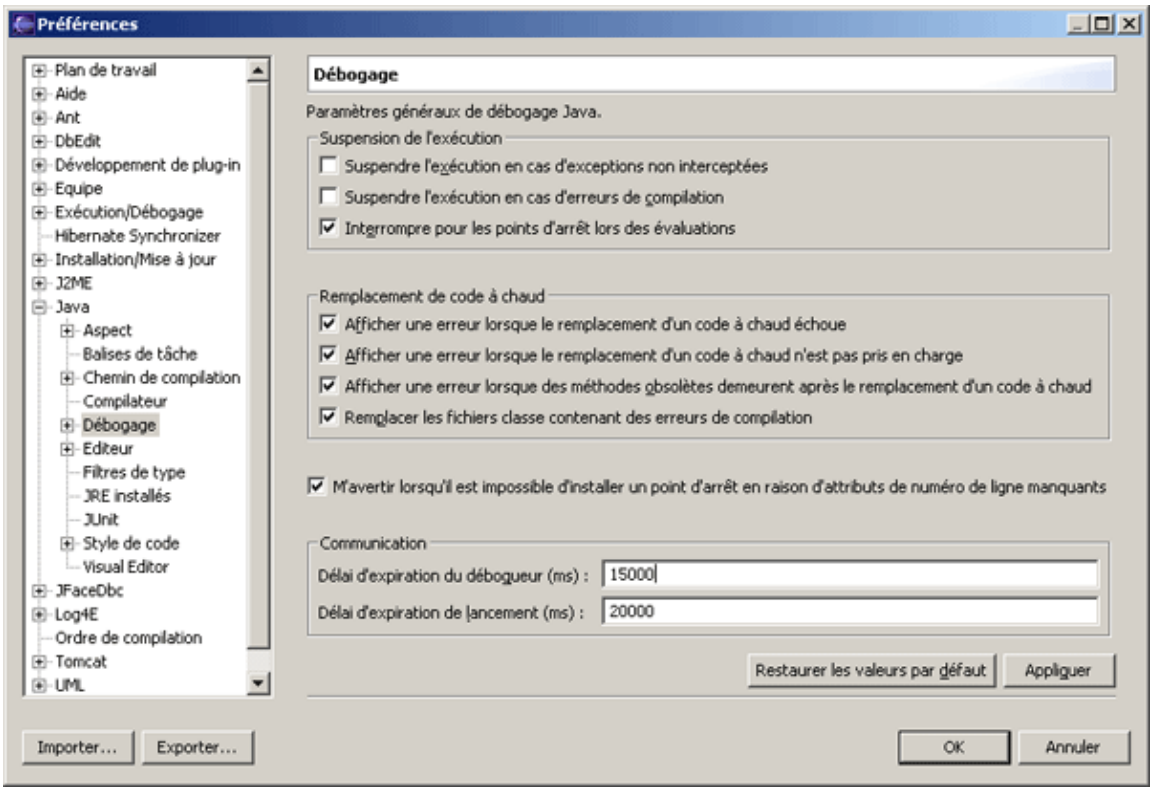
Par exemple, pour ajouter le J2ME Wireless Toolkit 2.1, sélectionnez l'option « Add Wireless Toolkit » du menu contextuel de « Wireless Toolkits »

Cliquez sur le bouton « Browse » pour sélectionner le répertoire où le J2ME Wireless Toolkit 2.1 est installé et cliquez sur le bouton « Fin ».



Pour pouvoir utiliser le débogueur d'Eclipse avec un « Wireless Toolkit », il faut modifier les paramètres du débogueur.

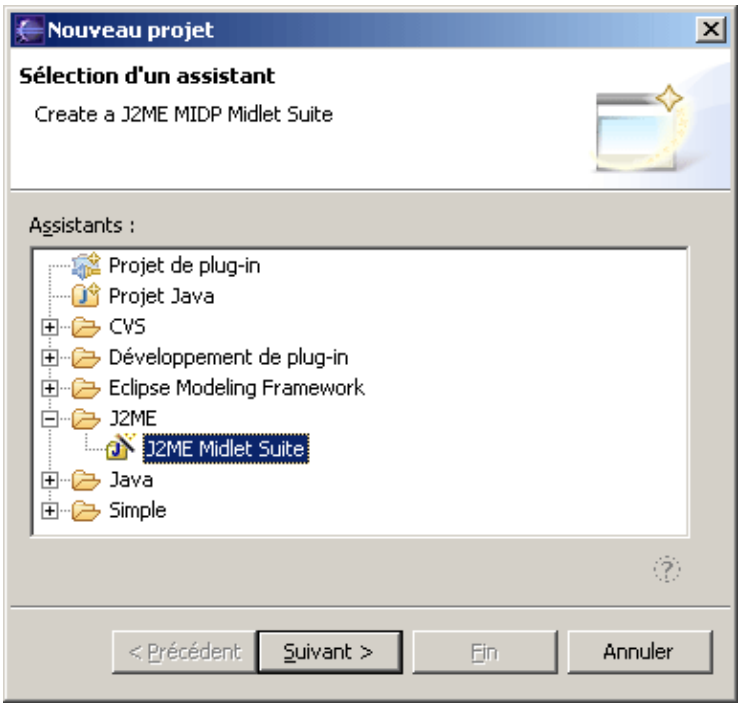
Dans les préférences, décochez les deux premières options et modifiez la valeur du délai d'expiration du débogueur pour mettre la valeur 15000.



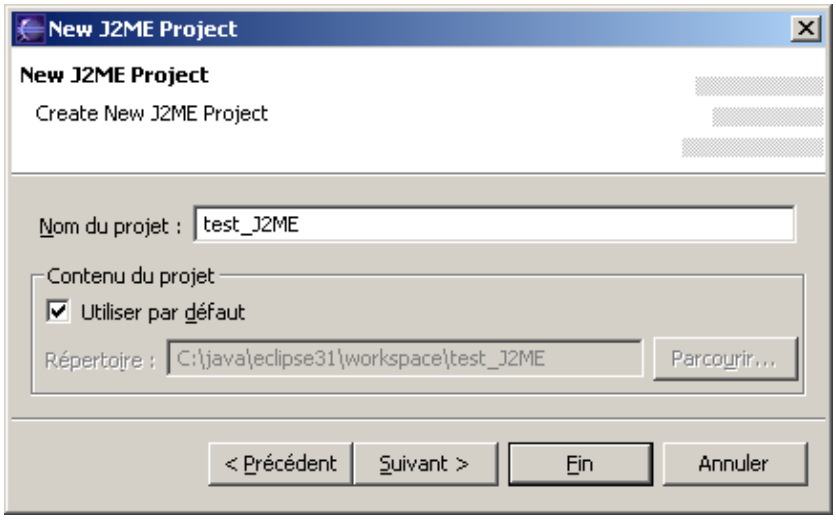
Cliquez sur le bouton « OK »

26.1.3. Création d'un premier exemple

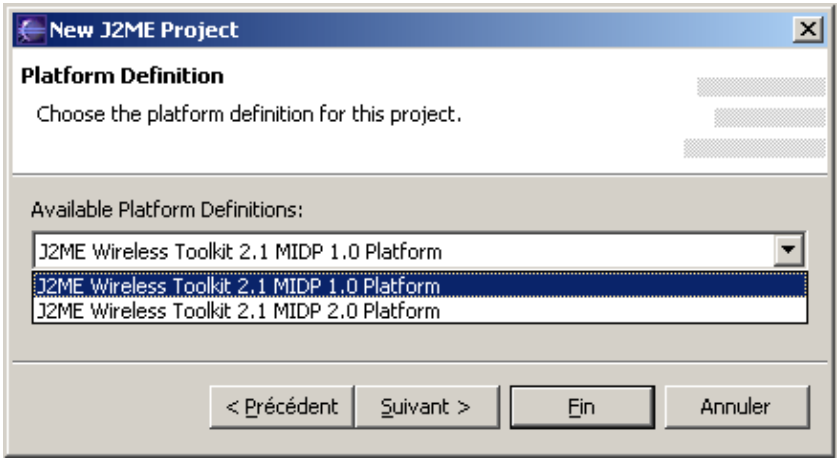
Il faut créer un nouveau projet de type « J2ME/J2ME Midlet Suite ».



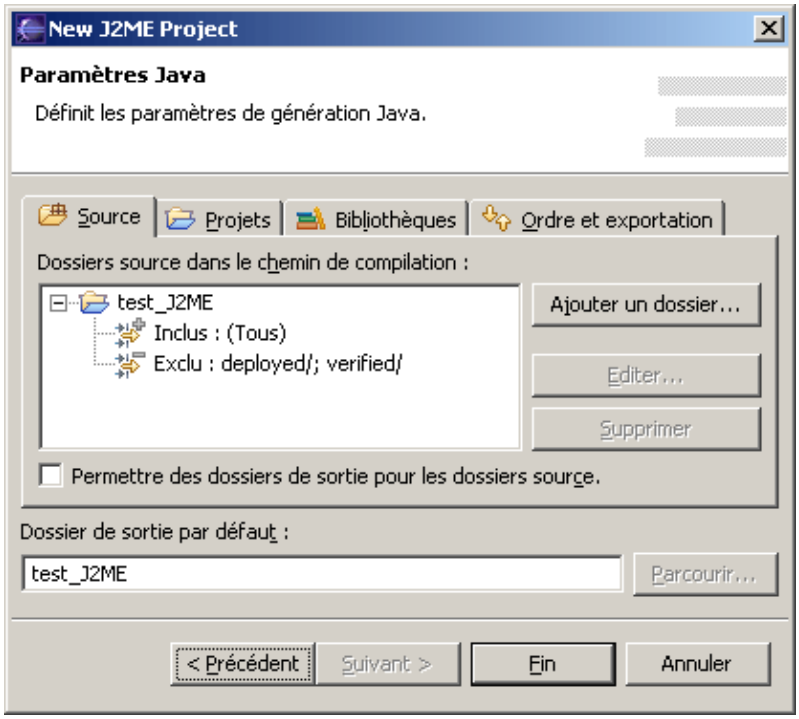
Cliquez sur le bouton « Suivant ».



Saisissez le nom du projet et cliquez sur le bouton « Suivant ».

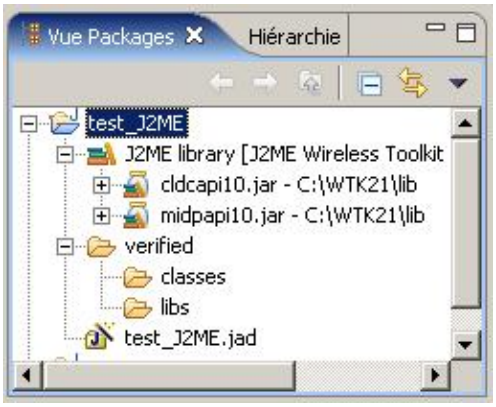


Sélectionnez la plate−forme et cliquez sur le bouton « Suivant ».

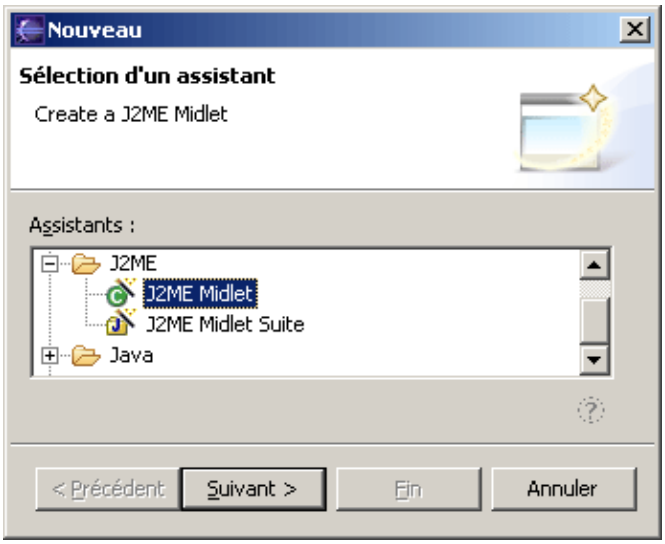


La dernière page de l'assistant permet de gérer les paramètres du projet tel qu'il est possible de le faire pour un projet Java classique.

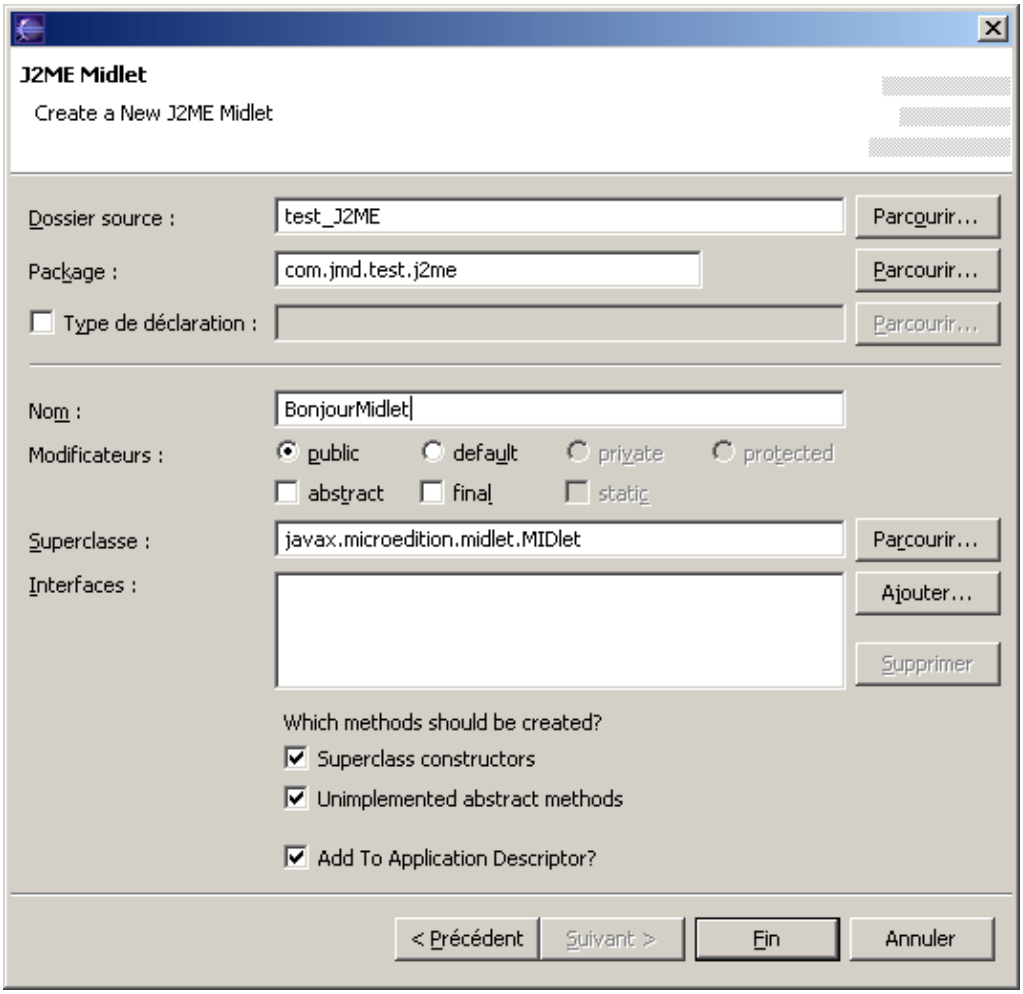
Cliquez sur le bouton « Fin » pour créer le projet.



Ajouter une midlet en créant une nouvelle entité de type « J2ME/J2ME Midlet ».



Cliquez sur le bouton « Suivant ».



La page suivante de l'assistant permet de préciser les caractéristiques de la midlet.

Saisissez le nom du package et le nom de la classe et cliquez sur le bouton « Fin ».

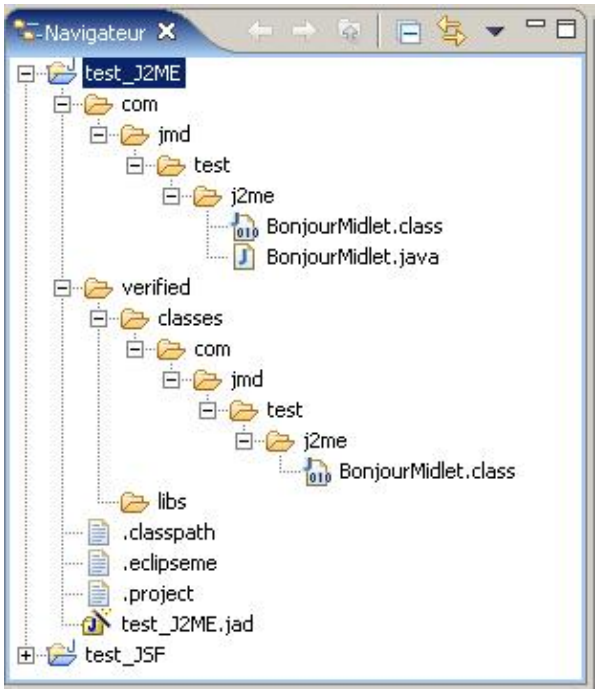
Il faut ensuite saisir le code de la midlet.

Exemple :

```
package com.jmd.test.j2me;

import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;

public class BonjourMidlet extends MIDlet {
  private Display display;
  private TextBox textbox;
  public BonjourMidlet() {
    super();
    display = Display.getDisplay(this);
    textbox = new TextBox("", "Bonjour", 20, 0);
  }

  /* (non-Javadoc)
   * @see javax.microedition.midlet.MIDlet#startApp()
   */
  protected void startApp() throws MIDletStateChangeException {
    display.setCurrent(textbox);
  }

  /* (non-Javadoc)
```

```
   * @see javax.microedition.midlet.MIDlet#pauseApp()
   */
  protected void pauseApp() {
    // TODO Raccord de méthode auto-généré
  }

  /* (non-Javadoc)
   * @see javax.microedition.midlet.MIDlet#destroyApp(boolean)
   */
  protected void destroyApp(boolean arg0) throws MIDletStateChangeException {
    // TODO Raccord de méthode auto-généré
  }
}
```

La structure du projet est la suivante :



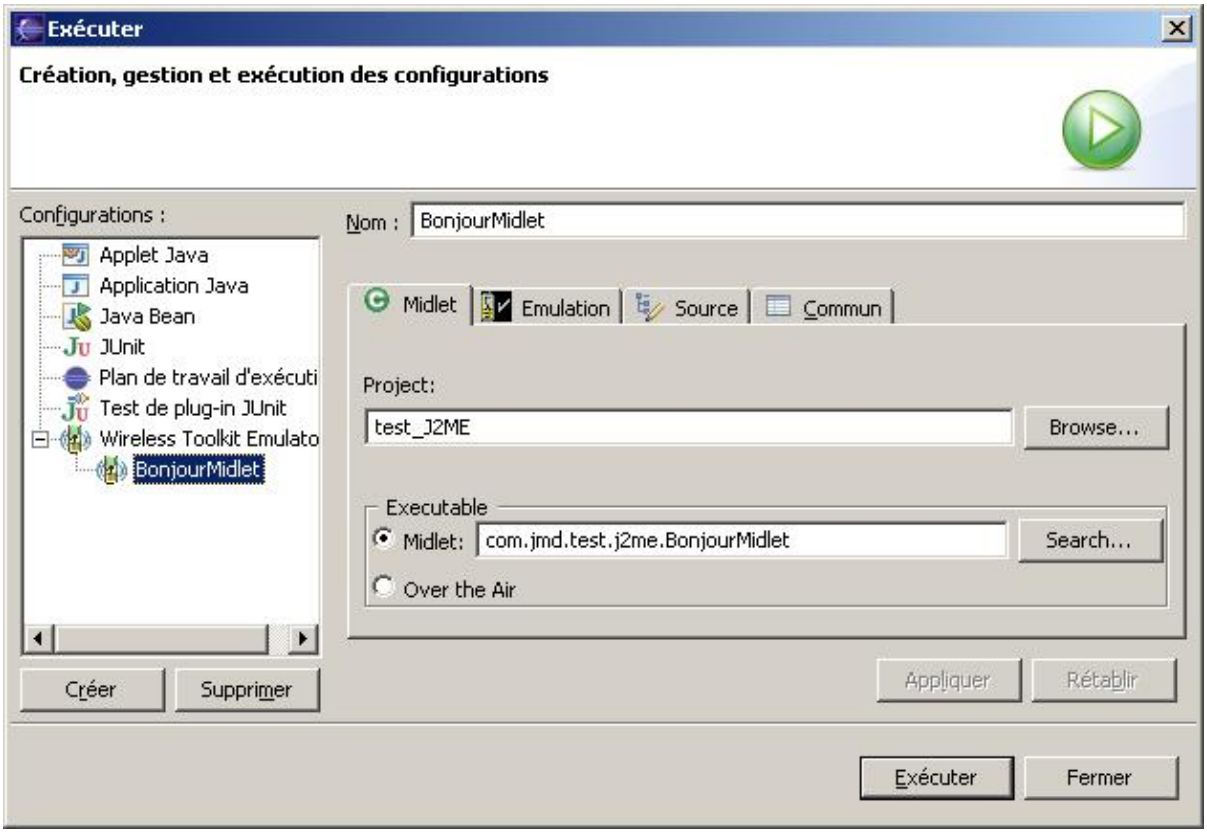
26.1.4. Exécution de l'application

Pour exécuter la midlet, il faut cliquer sur la petite flèche de l'icône  de la barre d'outils.

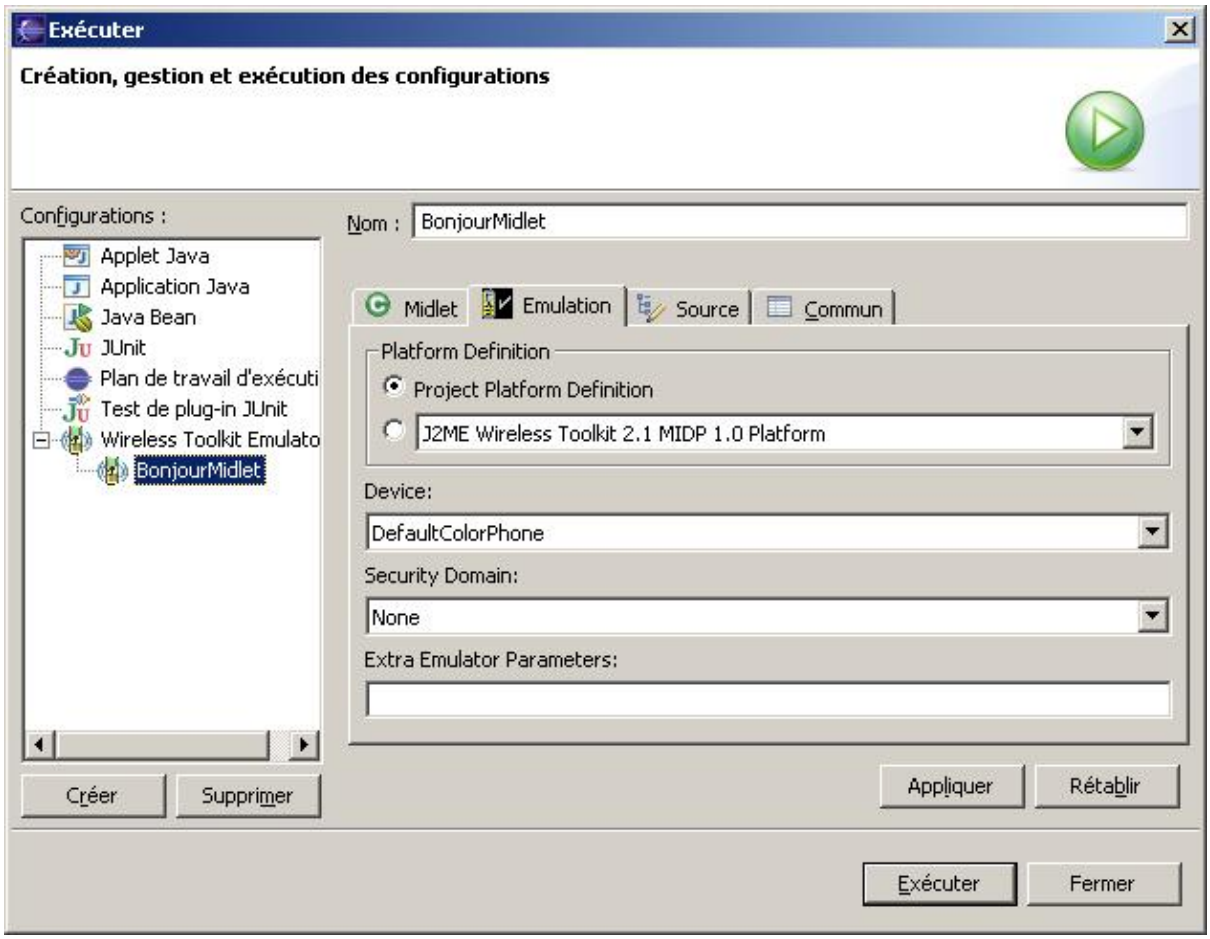
Il suffit alors de sélectionner « Emulated J2ME Midlet » pour lancer l'application dans l'émulateur.

L'émulateur couleur par défaut est utilisé pour exécuter l'application.

Pour permettre de paramétrer le mode d'exécution de la midlet, il suffit d'utiliser l'option « Exécuter en tant que ... »



L'onglet Midlet permet notamment de préciser le mode d'exécution.

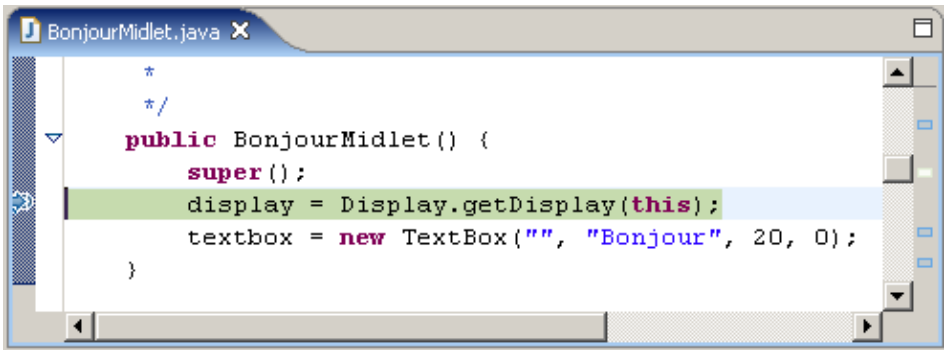


L'onglet Emulation permet de préciser la plate−forme et l'émulateur à utiliser.

26.1.5. Déboguer l'application

Pour déboguer l'application, il suffit de placer un point d'arrêt dans le code et de cliquer sur la petite flèche de l'icône  dans la barre de tache.

La perspective « Débogueur » s'ouvre lors de l'exécution de l'instruction contenant le point d'arrêt.

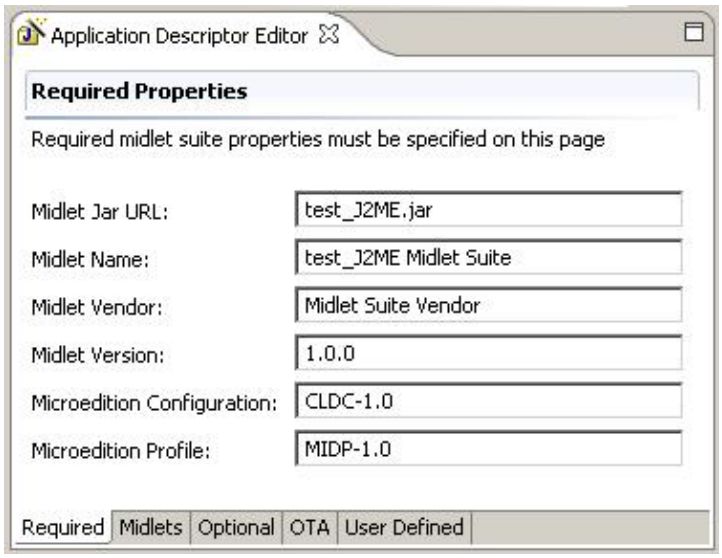


26.1.6. Modification des propriétés du descripteur d'application

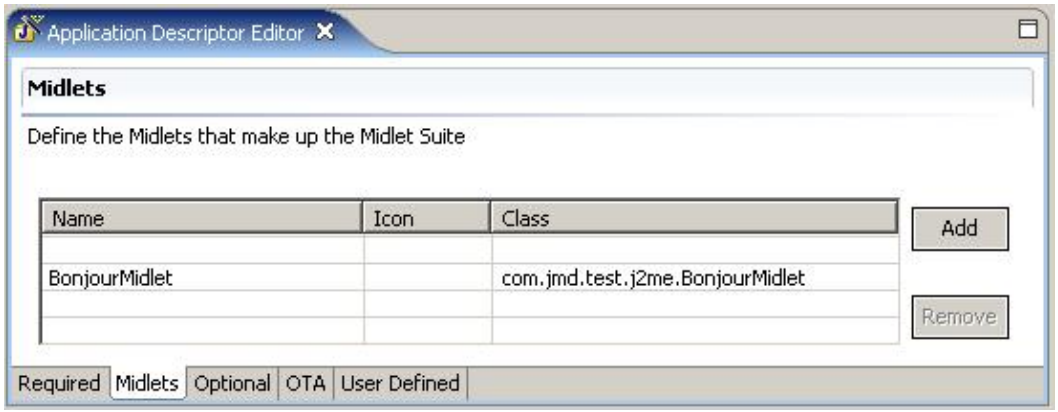
Une application J2ME est composée de deux fichiers :

- Un fichier .jar qui contient l'exécutable
- Un fichier .jad qui contient des informations que l'application

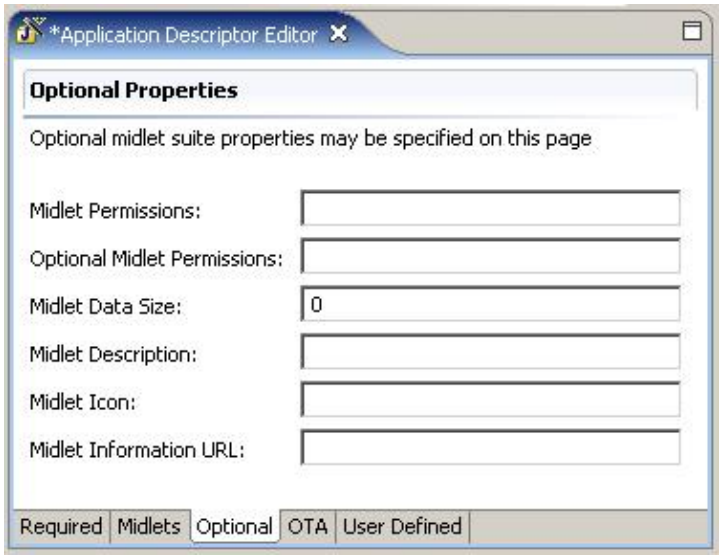
EclipseMe propose un éditeur dédié pour éditer le fichier.jad associé à l'application. Pour ouvrir cet éditeur, il suffit de double−cliquer sur le fichier .jad du projet.



Le premier onglet permet de modifier les propriétés obligatoires.



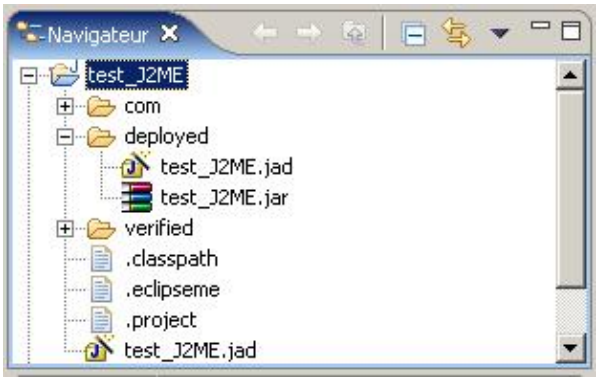
L'onglet Midlets permet de préciser la ou les midlets qui composent l'application.



L'onglet « Optional » permet de préciser des paramètres optionnels.

26.1.7. Packager l'application

Pour créer un package de l'application, il faut utiliser l'option « J2ME/Create package » du menu contextuel du projet dans le navigateur.



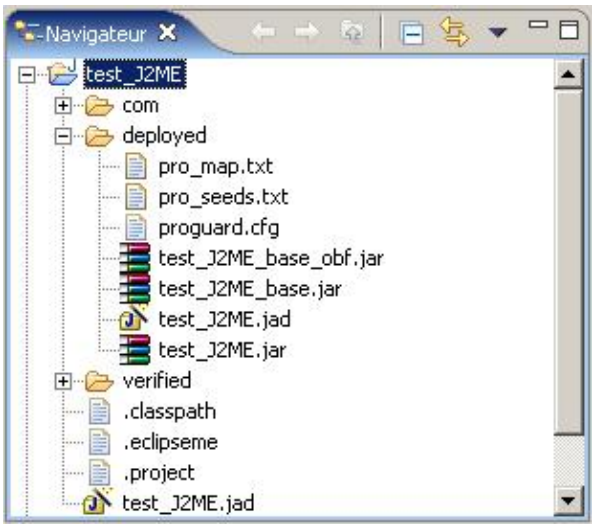
Les fichiers .jad et .jar sont créés dans le répertoire deployed. Le fichier .jar inclus les classes et les bibliothèques contenues dans le répertoire verified, les ressources (images, son, ...) contenues dans le répertoire res et le fichier manifest généré à partir du fichier .jad à la racine du projet.

Exemple de fichier META-INF/MANIFEST.MF :

```
Manifest-Version: 1.0
MIDlet-2: BonjourMidlet,,com.jmd.test.j2me.BonjourMidlet
MicroEdition-Configuration: CLDC-1.0
MIDlet-Name: test_J2ME Midlet Suite
MIDlet-Vendor: Midlet Suite Vendor
MIDlet-1: ,,
MIDlet-Version: 1.0.0
MicroEdition-Profile: MIDP-1.0
```

Si Proguard est installé et configuré dans le plug-in, il est possible de créer un package obscurci de l'application, il faut utiliser l'option « J2ME/Create obsfucated package » du menu contextuel du projet dans le navigateur.

Plusieurs fichiers sont générés dans le répertoire deployed.



Le fichier test_J2ME_base.jar est le fichier .jar avant l'opération d'obscurcissement.

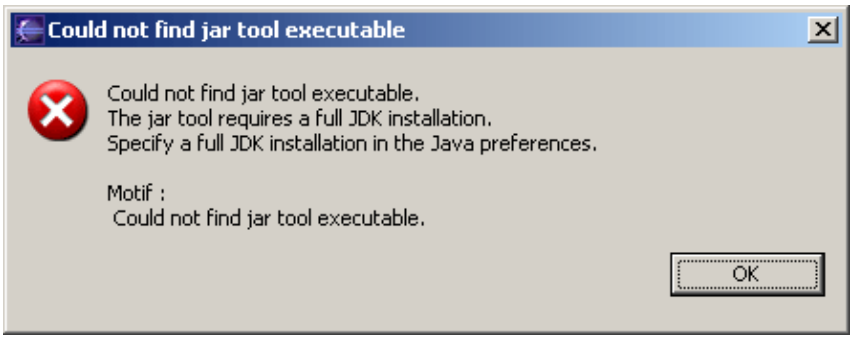
Le fichier test_J2ME_base_obf.jar est le fichier .jar après l'opération d'obscurcissement.

Le fichier proguard.cfg contient les options utilisées par Proguard pour réaliser le package.

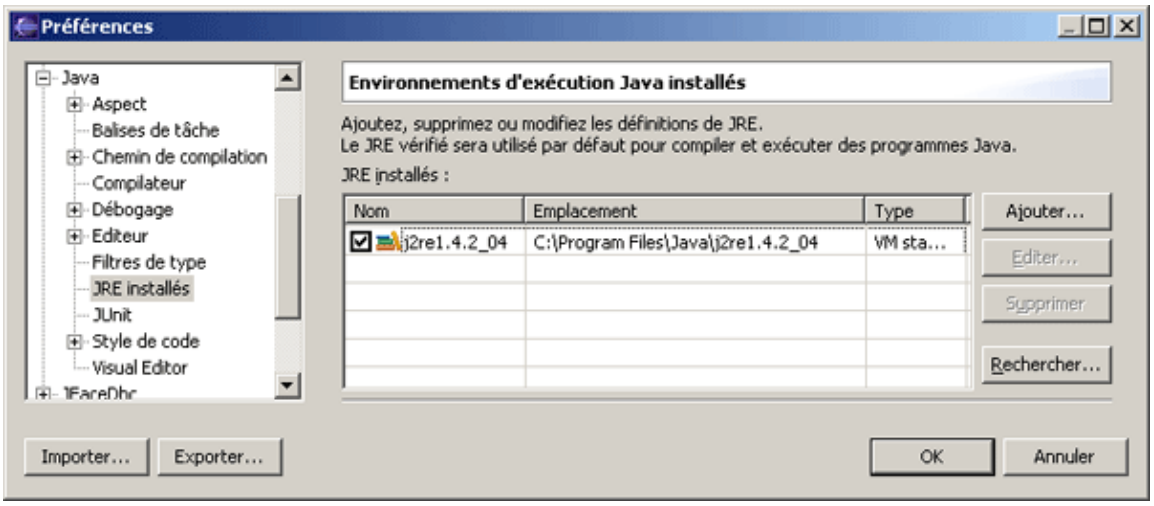
Exemple :

```
-libraryjars C:\WTK21\lib\cldcapi10.jar;C:\WTK21\lib\midpapi10.jar
-injars      C:\java\eclipse31\workspace\test_J2ME\deployed\test_J2ME_base.jar
-outjar      C:\java\eclipse31\workspace\test_J2ME\deployed\test_J2ME_base_obf.jar
-printseeds  C:\java\eclipse31\workspace\test_J2ME\deployed\pro_seeds.txt
-printmapping C:\java\eclipse31\workspace\test_J2ME\deployed\pro_map.txt
-dontnote -defaultpackage ''
-keep public class * extends javax.microedition.midlet.MIDlet
```

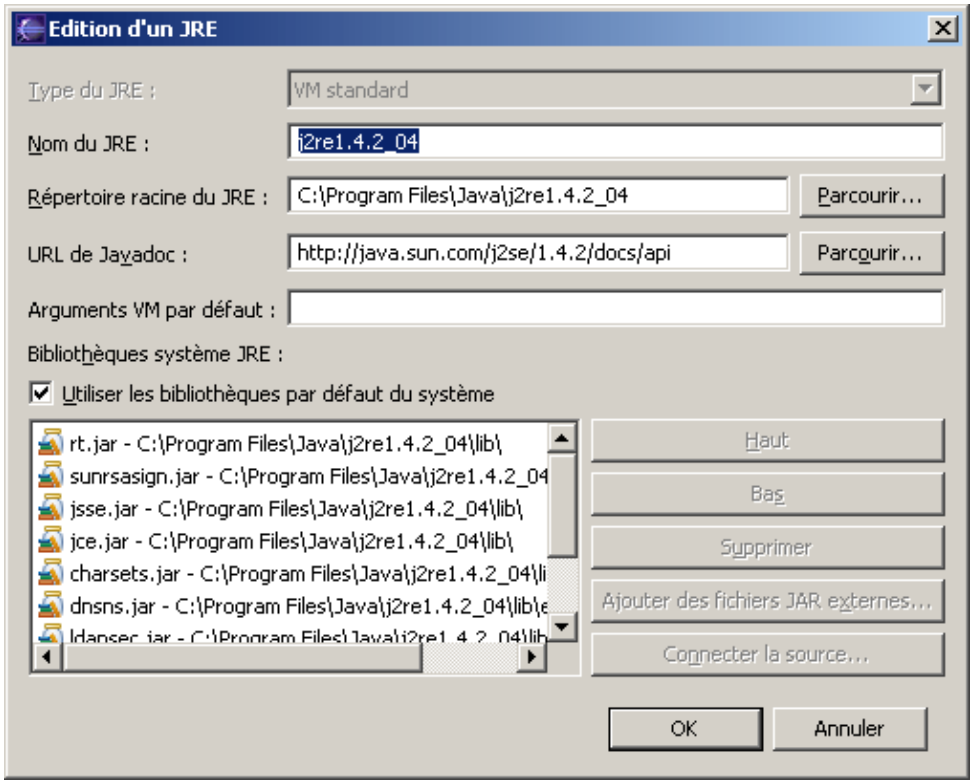
Lors de la réalisation du packaging, il est possible d'obtenir ce message d'erreur.



Dans ce cas, il faut modifier les paramètres du JRE définis dans les Préférences d'Eclipse : par défaut Eclipse utilise un JRE et Proguard nécessite un JDK lors de ces opérations.

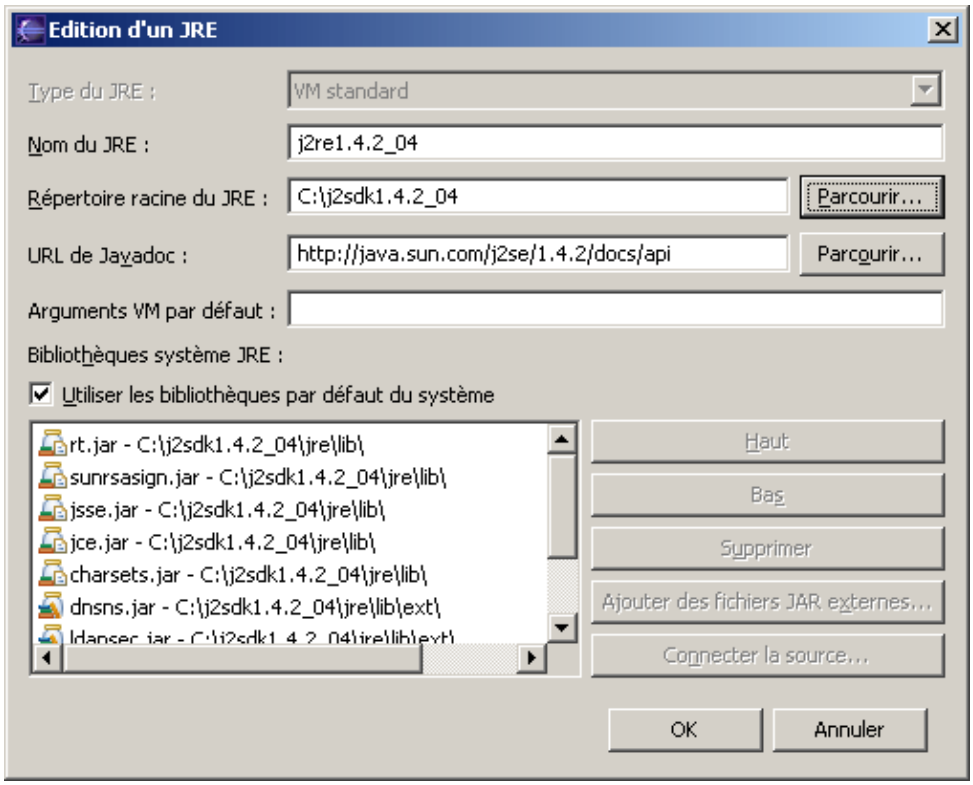


Sélectionnez le JRE et cliquez sur le bouton « Editer »



Cliquez sur le bouton « Parcourir » associé à « Répertoire racine du JRE »

Il suffit de sélectionner le répertoire qui contient le JDK et de cliquer sur le bouton « OK ».



Cliquez sur le bouton « OK ».

Cliquez encore sur le bouton « Ok » pour fermer la fenêtre des « Préférences ».

Partie 6 : Annexes

27. Annexes

Annexe A : GNU Free Documentation License

Version 1.1, March 2000

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND définiTIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front−matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front−Cover Texts or Back−Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine−readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and

straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard−conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine−generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front−Cover Texts on the front cover, and Back−Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine−readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly−accessible computer−network location containing a complete Transparent copy of the Document, free of added material, which the general network−using public has access to download anonymously at no charge using public−standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3

above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front−matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties−−for example, statements of peer review or that the text has been

approved by an organization as the authoritative définition of a standard.

You may add a passage of up to five words as a Front−Cover Text, and a passage of up to 25 words as a Back−Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front−Cover Text and one of Back−Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self−contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the

terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

Annexe B : Webographie

http://www.eclipse.org/
le site officiel d'Eclipse

http://www.eclipse.org/downloads/index.php
la page officielle pour le téléchargement d'Eclipse

http://www.eclipsetotale.com/index.html
portail en français consacré au projet Eclipse et aux outils WebSphere Studio d'IBM

http://eclipse−plugins.2y.net/eclipse/index.jsp

http://eclipsewiki.swiki.net/

http://www.sysdeo.com/eclipse/tomcatPluginFR.html
plug−in pour utiliser Tomcat dans Eclipse

http://www.eclipse−workbench.com/jsp/
portail en anglais

http://www.geocities.com/uwe_ewald/dbedit.html
un plug−in qui permet de visualiser le contenu d'une base de données