Développons en Java avec Eclipse

Jean Michel DOUDOUX

Table des matières

Table des matières

Table des matières

Table des matières

Développons en Java avec Eclipse

Version 0.30

du 04/01/2004

par Jean−Michel DOUDOUX

Préambule

A propos de ce document

Ce document fait suite à mon premier tutorial "Développons en Java". C'est un didacticiel qui se propose de fournir des informations pratiques sur l'utilisation d'Eclipse.

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	Ajout dans le chapitre "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

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 le workspace. Le workspace 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 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 élements du workspace.

1.1. Les différentes versions d'Eclipse

Il existe quatre grandes catégories de versions. Chacune de ces catégories possède un nombre plus ou moins important de versions nommées "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 "Realease" d'Eclipse :

Date	Version
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 3.0 est en cours de développement.

Le projet Eclipse est divisé en quatre grands 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 developpement en Cobol), ...
- 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 à 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.

2. Installation

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, Solaris 8, QNX, AIX, HP−UX.

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 debogage (technologie JPDA).

Lors de son premier lancement, Eclipse créé par défaut un répertoire nommé Workspace qui va contenir les projets.

2.1. Installation d'Eclipse sous Windows

2.1.1. Installation 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.

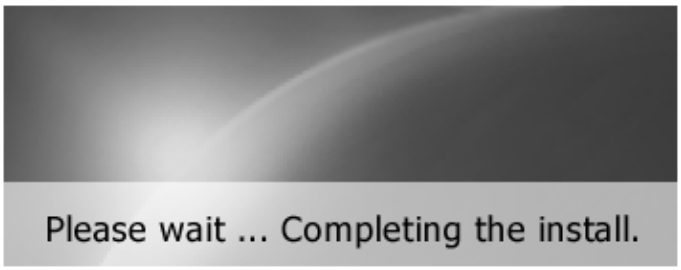
Si la splash screen reste affichée et que l'application ne se lance pas, c'est qu'elle n'arrive pas à trouver le JDK requis.

2.1.2. Installation Eclipse 2.0

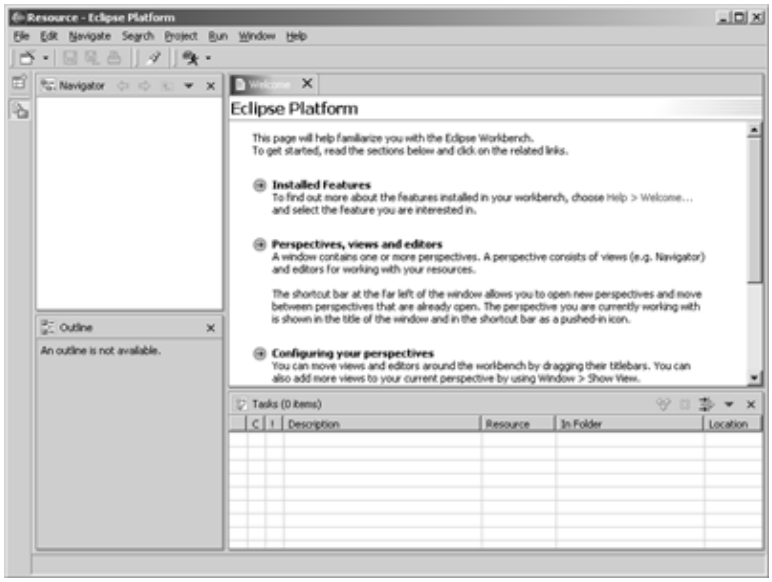
Il faut télécharger le fichier eclipse−SDK−2.0.2−win32.zip sur le site :

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 dézippé.



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



2.1.3. Installation des traductions de la version 2.x

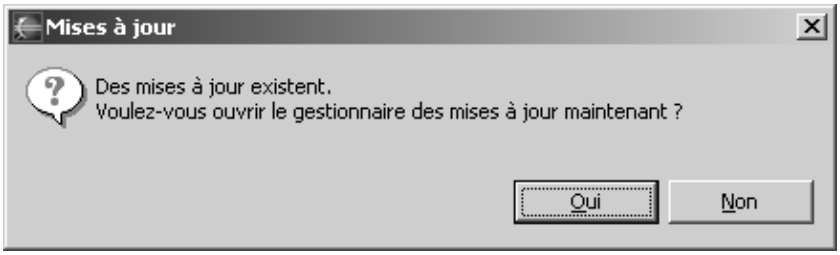
Par défaut, la langue utilisée dans Eclipse est l'anglais. I.B.M. propose des traductions pour la version 2.0.2 d'Eclipse dans différentes langues.

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

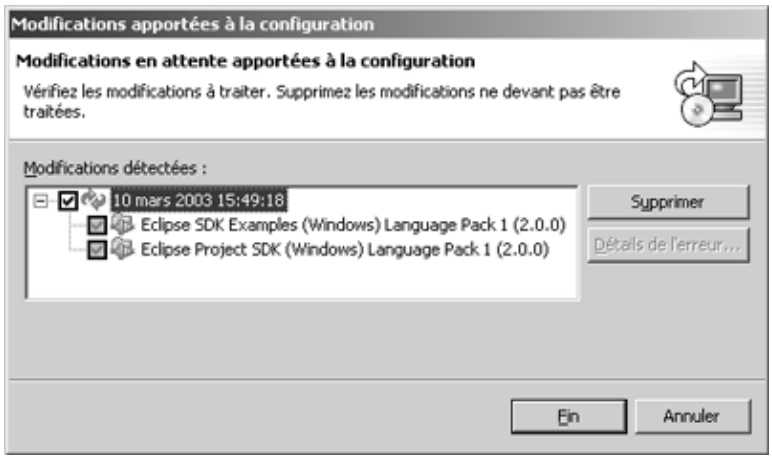
http://download2.eclipse.org/downloads/drops/L−2.0.x%20Translations−200301071000/index.php

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

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

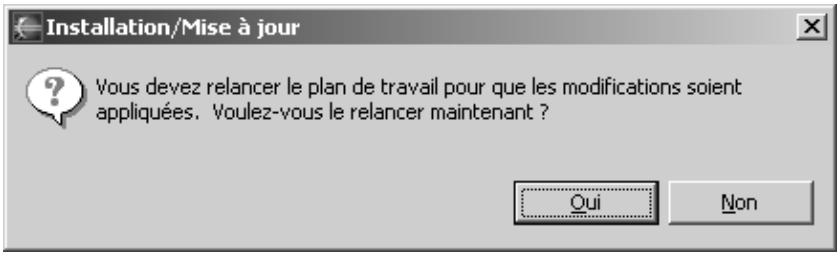


Un clic sur "Oui" ouvre une boîte de dialogue qui permet de sélectionner la mise à jour à installer.

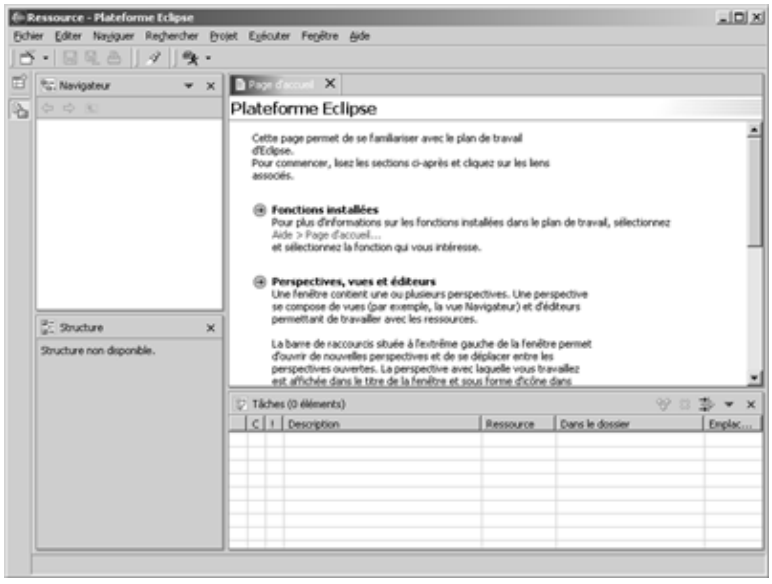


Il suffit de cocher la mise à jour souhaitée et de cliquer sur "Fin".

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



L'application redémarre automatiquement, localisée dans la langue du poste.



Pour la version 2.1.1., il faut télécharger le fichier correspondant au système d'exploitation utilisé sur la page :

http://download.eclipse.org/downloads/drops/L−2.1.x%20Translations−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 l'url :

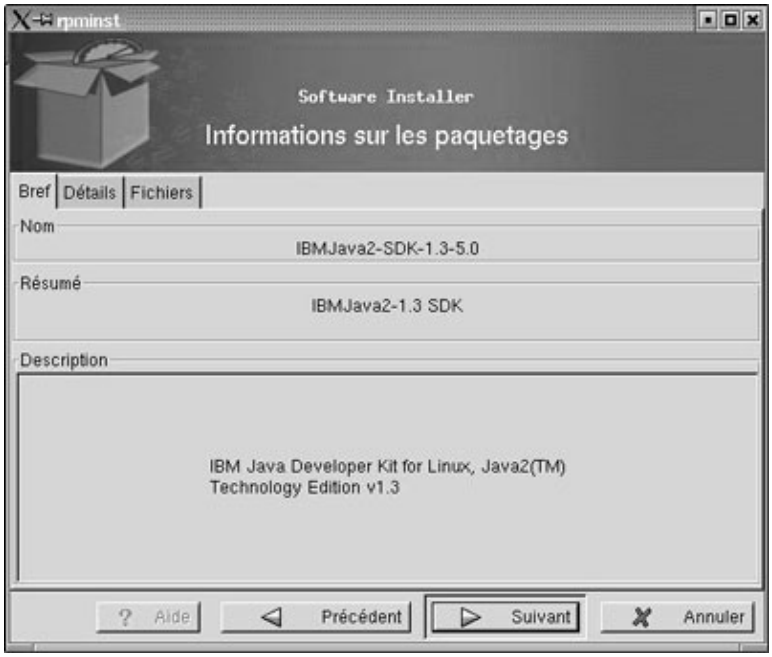
http://download2.eclipse.org/downloads/drops/L−2.1.x%20Translations−200307021300/eclipse2.1.1.1−SDK−Languag

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.

2.2. Installation Eclipse sous Linux

2.2.1. Installation 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.

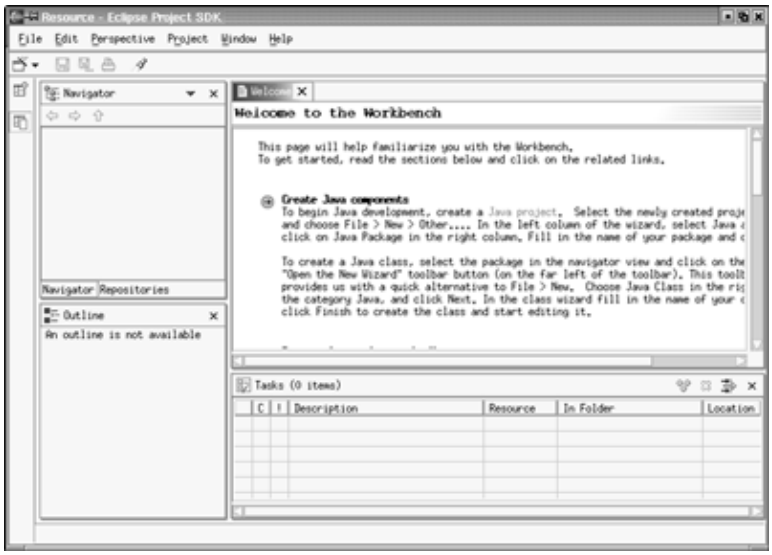
PATH=$PATH:/opt/IBMJava2−13/bin

Si les exécutables ne sont pas touvés, 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 : 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 dézipper le fichier dans un répertoire, par exemple /opt avec l'utilisateur root.

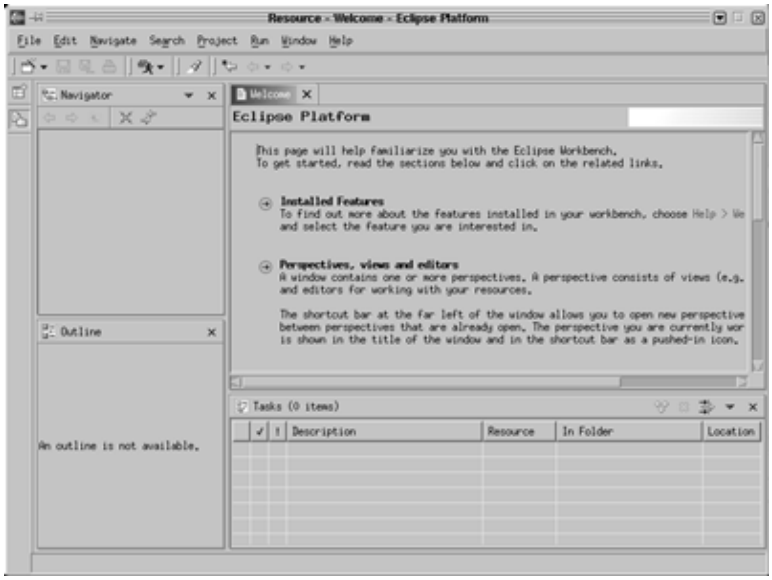
Exemple :

```
[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/
   creating: eclipse/plugins/org.eclipse.core.boot_2.1.0/
  inflating: eclipse/plugins/org.eclipse.core.boot_2.1.0/boot.jar
  inflating: eclipse/plugins/org.eclipse.core.boot_2.1.0/splash.bmp
  inflating: eclipse/plugins/org.eclipse.core.boot_2.1.0/boot.xml
  inflating: eclipse/plugins/org.eclipse.core.boot_2.1.0/plugin.properties
  inflating: eclipse/plugins/org.eclipse.core.boot_2.1.0/plugin.xml
  inflating: eclipse/plugins/org.eclipse.core.boot_2.1.0/.options
  inflating: eclipse/plugins/org.eclipse.core.boot_2.1.0/about.html
        ...
```

Pour utiliser exécuter 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]$ 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
[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 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
```

3. Le Workspace

Le workspace 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 du workspace.

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

Le workspace 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 le vue Navigateur.

3.1. La création de nouvelles entités

Dans Eclipse, on peut créer différents types d'entités qui seront stockées dans le workspace :

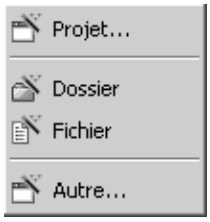
- des projets
- des répertoires pour organiser les projets
- des ressources de différents types qui sont des fichiers

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 :

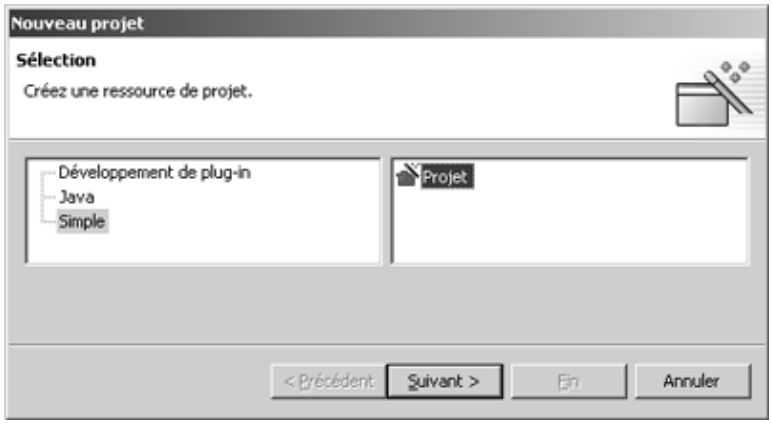


Il est ainsi possible de créer rapidement un projet, un répertoire ou un fichier. Si le fichier est d'un type particulier, un clic sur "Autre" ouvre un assistant.

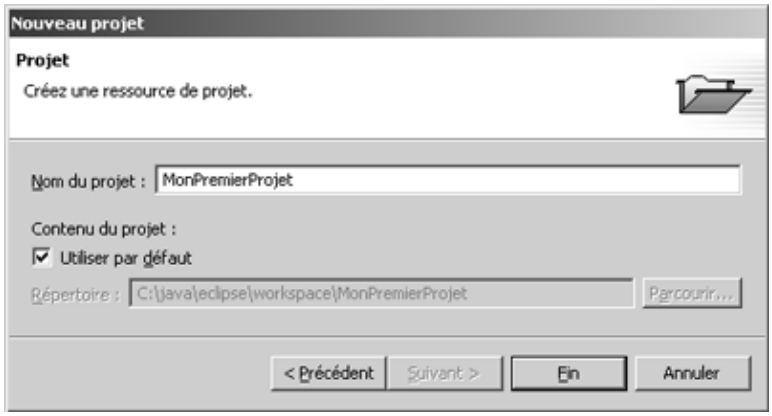
3.1.1. La création d'un projet

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

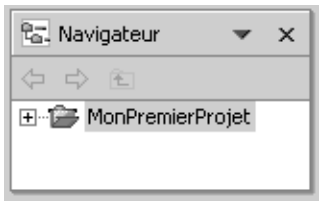
L'assistant permet de sélectionner le type du projet à créer. Il suffit alors de sélectionner la famille, le type du projet et de cliquer sur "Suivant".



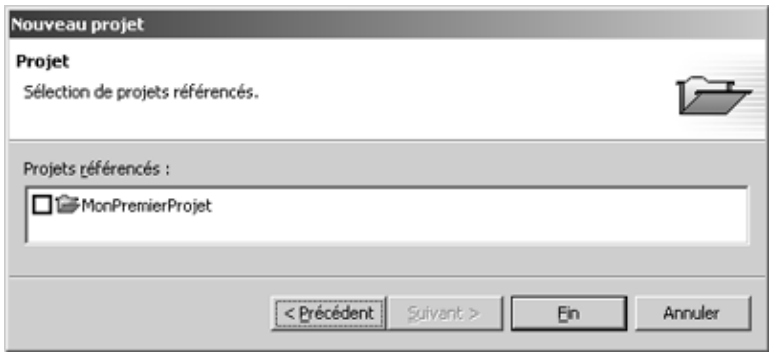
La création se fait grâce à un assistant qui demande le nom du nouveau projet. Ce nom ne doit pas contenir de blanc ou des caractères non alphabétiques.



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



Si le workspace 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 Next pour afficher le second volet de l'assistant. Il suffit de cocher les projets concernés.

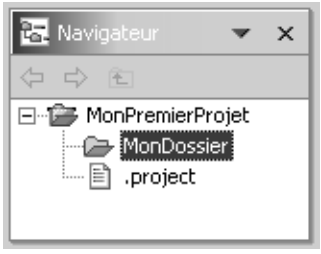


Il est possible de créer des projets particuliers selon le type d'applications à développer.

3.1.2. 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.

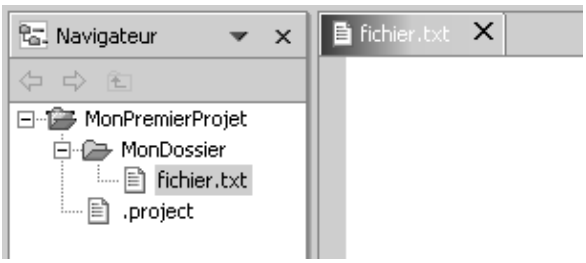
Il suffit ensuite de saisir le nom sans espace ni caractère non alphabétique et de cliquer sur "Fin". Le nouveau répertoire apparaît dans la vue Navigateur.



3.1.3. 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 "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 affiché sur le nouveau fichier.



3.2. La duplication d'un élément

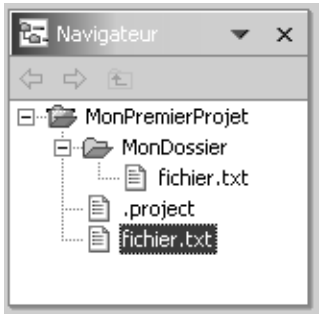
Dans la vue Navigateur, pour dupliquer un élément, le plus simple est de faire un cliquer/glisser de l'élément en maintenant la touche CTRL enfoncée vers son répertoire de destination dans le navigateur.

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

- appuyer sur 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 :

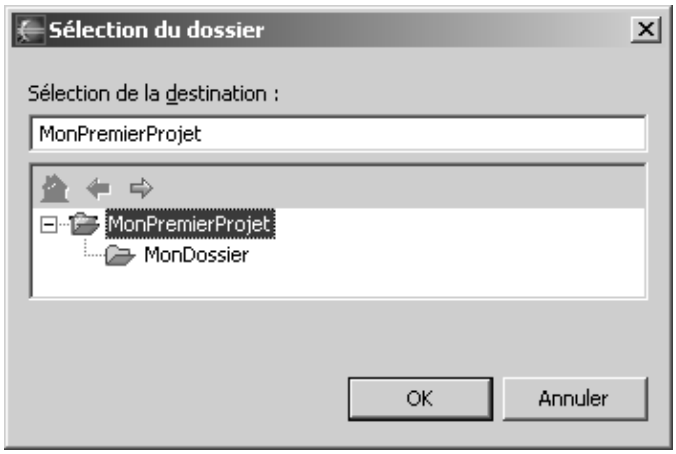
- appuyer sur CTRL + V
- sélectionner l'option "Coller" du menu contextuel



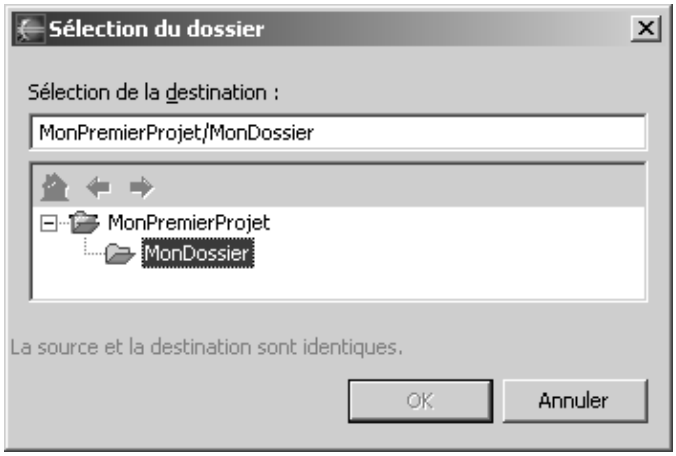
3.3. Le déplacement d'un élément

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 le 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.

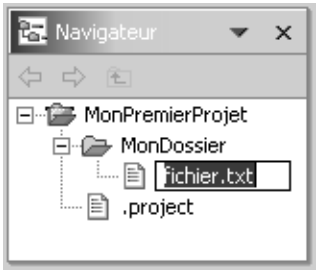


Si le répertoire de destination sélectionné est identique au répertoire d'origine du fichier, une message est affiché.



3.4. 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 "entrée".

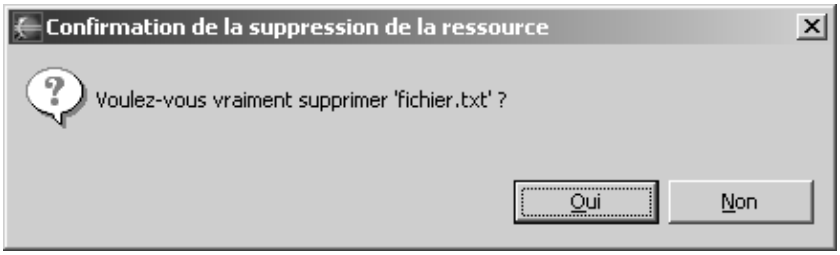


3.5. La suppression d'un élément

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"

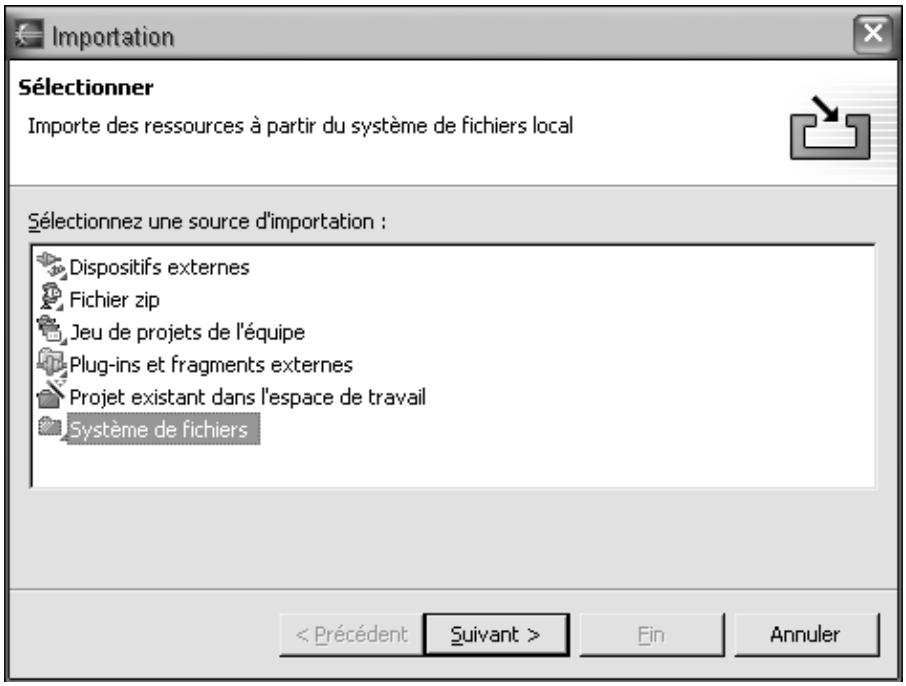
Une boîte de dialogue demande de confirmer la suppression.



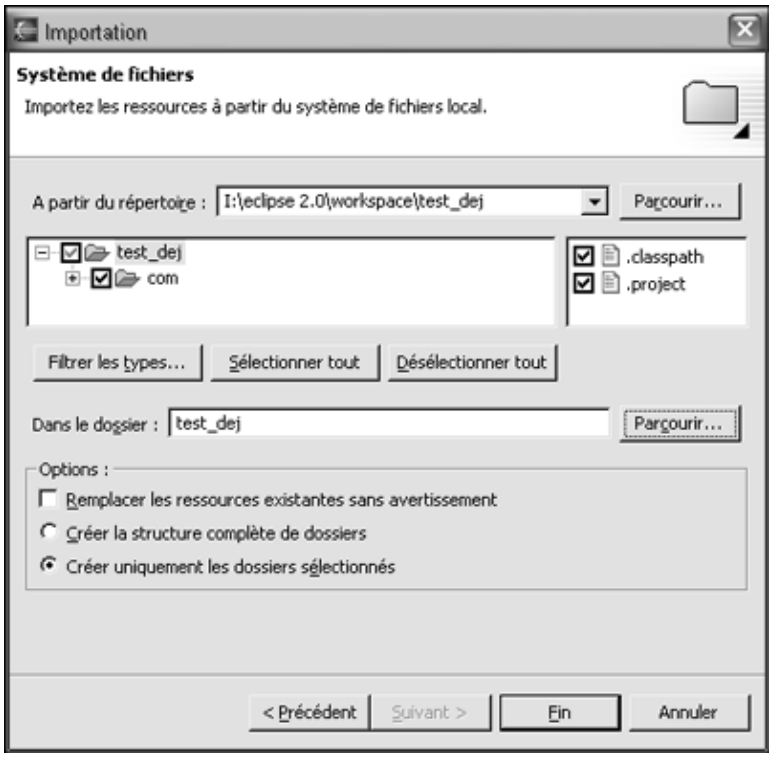
3.6. Importation

Attention, l'importation ne peut se faire que dans un projet existant.

Il faut utiliser le menu "Fichier/Importer"



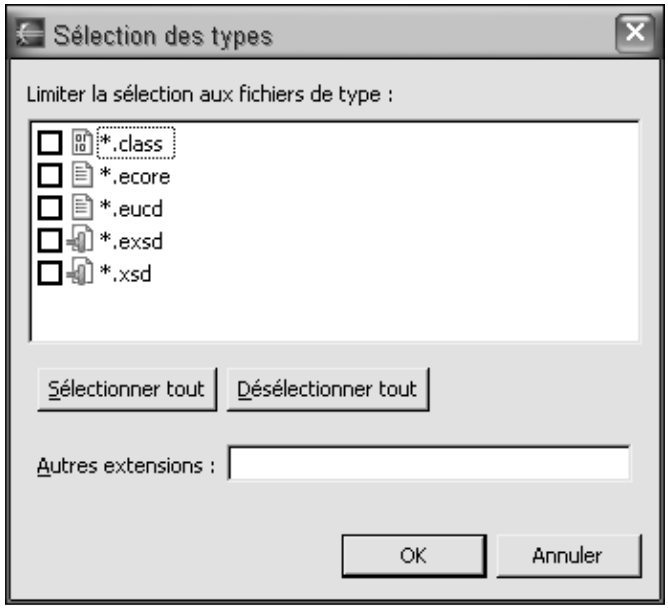
Sélectionner le type de la source d'importation et cliquer sur "Suivant"



Sélectionner le répertoire, puis cocher 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 du workspace.

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



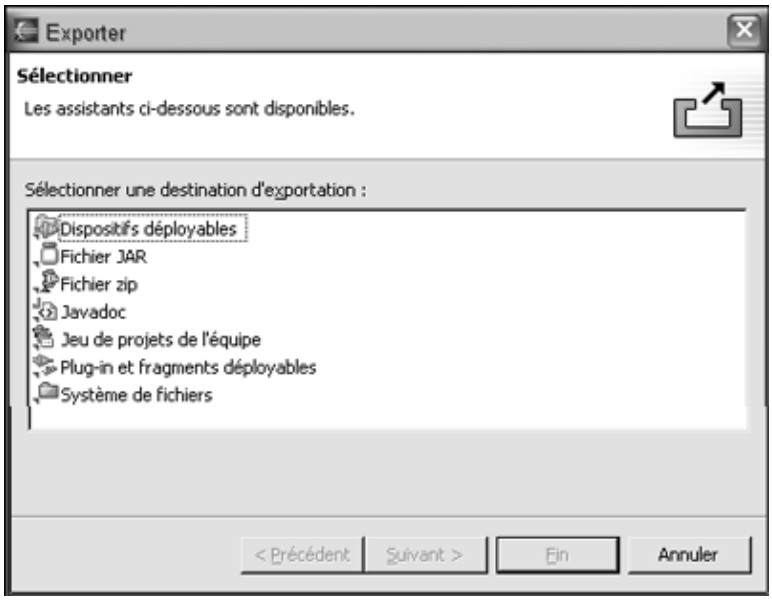
Il suffit de sélectionner les extensions parmi celles proposées et saisir d'autres extensions et cliquer sur "OK". Le filtre est alors directement appliqué sur la sélection.

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

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

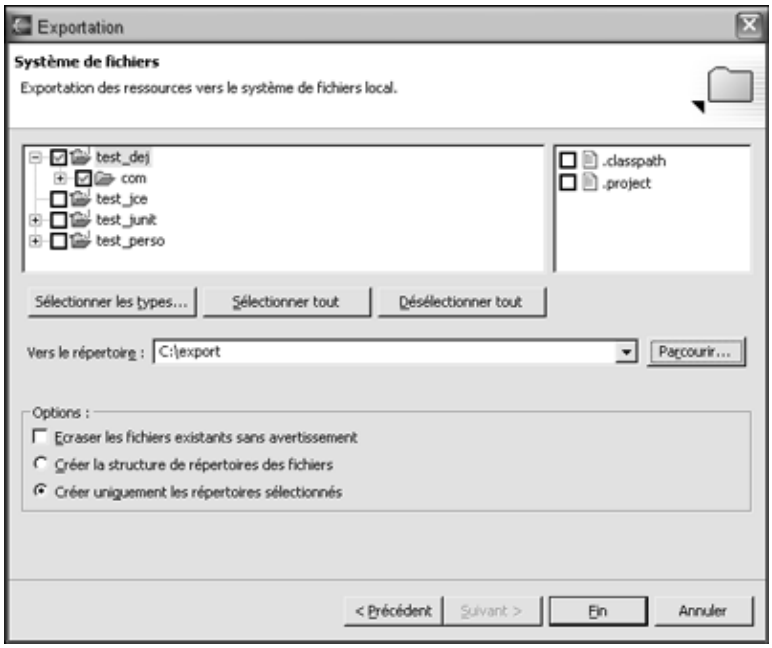
3.7. Exportation

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



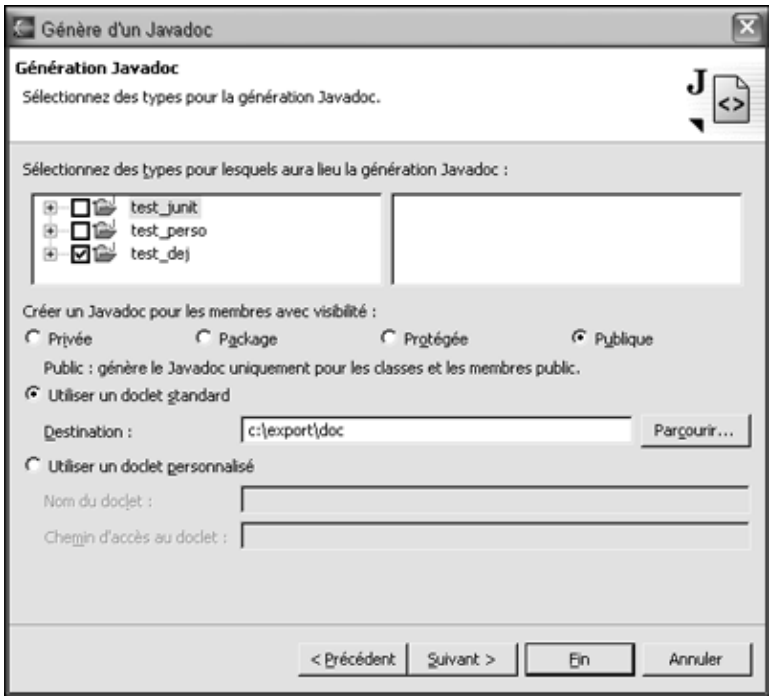
L'assistant demande de sélectionner le format d'exportation.

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



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

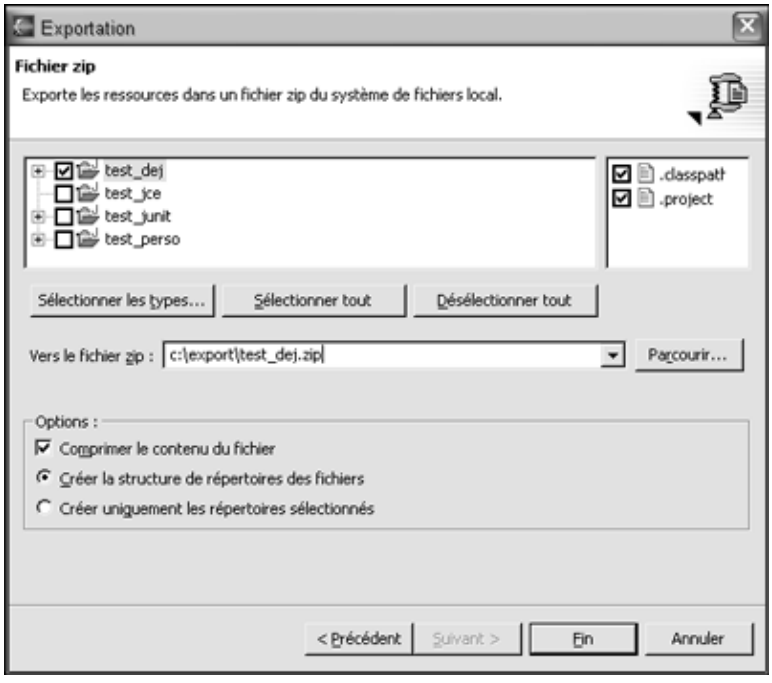
Si le format choisi est "Javadoc", l'assistant demande les informations nécessaires : les fichiers à exporter, le répertoire de destination



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

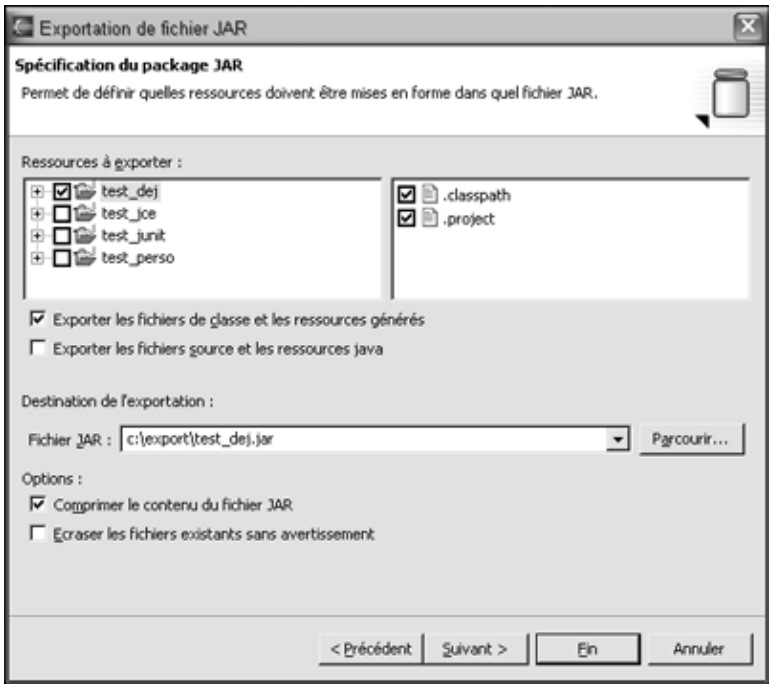
Un clic sur "Fin" permet de générer la documentation.

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

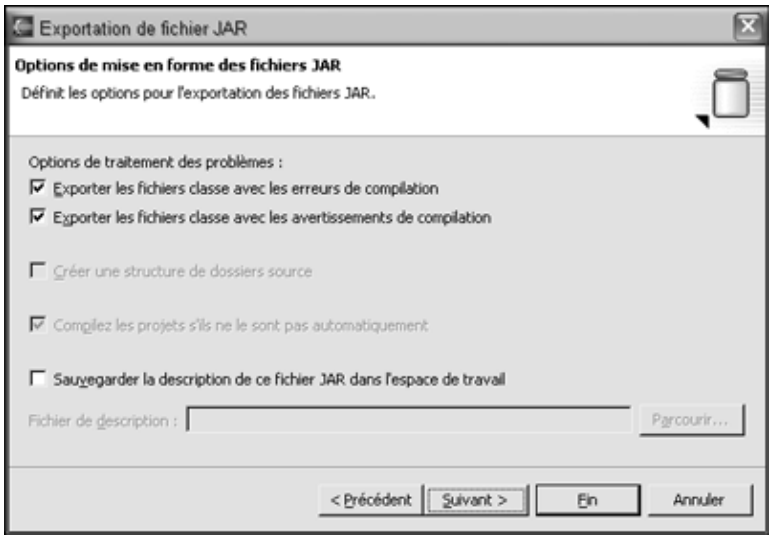


Un clic sur "Fin" permet de créer le fichier zip contenant tous les éléments sélectionnés.

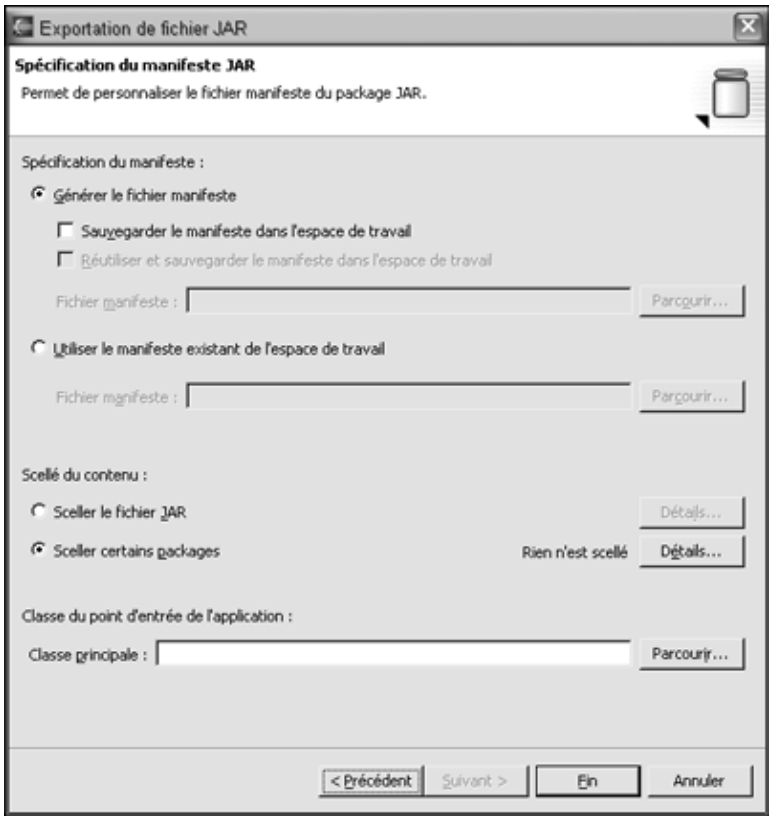
Si le format est "Fichier Jar", l'assistant demande les informations nécessaires : le ou les projets à exporter, le fichier jar à créer.



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



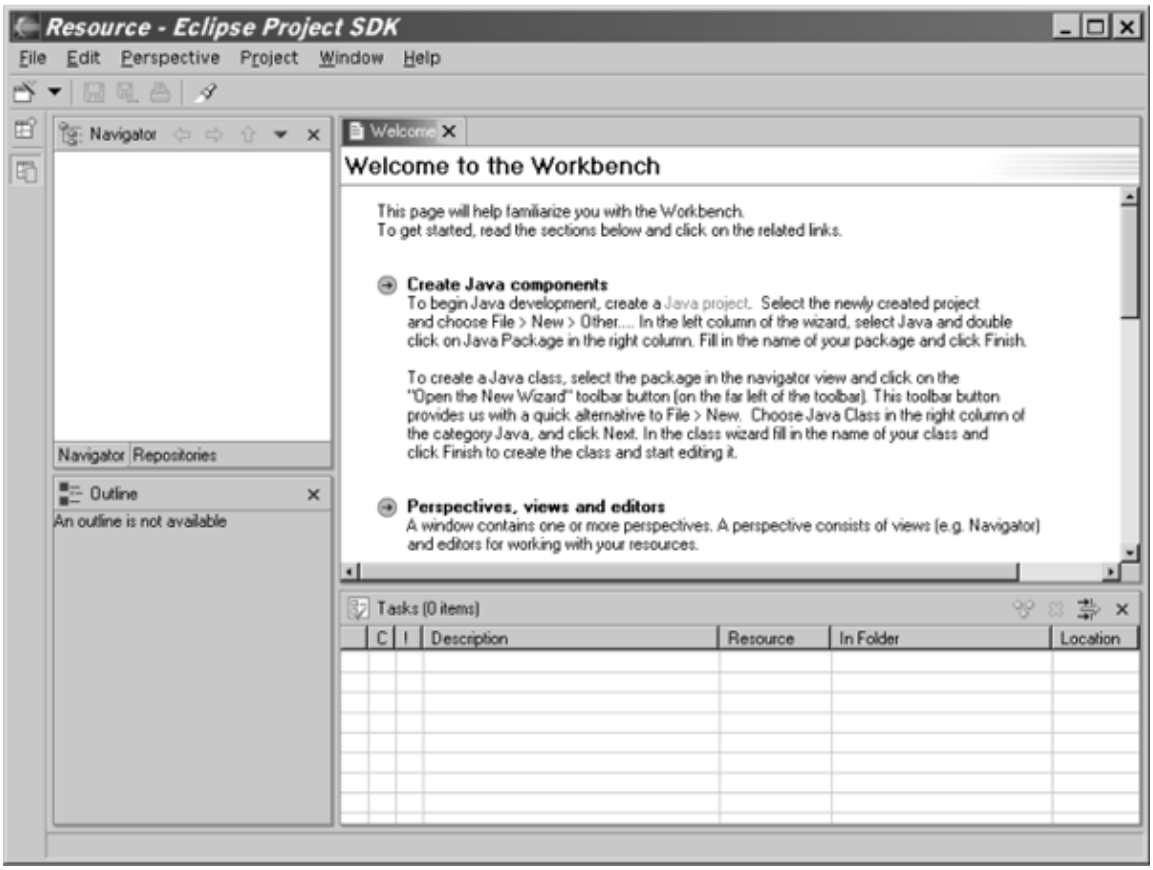
Un clic sur "Suivant" affiche une nouvelle page de l'assistant pour préciser le fichier manifest.


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

4. Le workbench

Au lancement d'Eclipse, une seule fenêtre s'ouvre contenant le "workbench".

Le "workbench" 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.



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

La partie gauche du workbench 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 affichée. Le titre de la fenêtre du workbench contient le nom de la perspective courante.

Eclipse possède dans le workbench 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 : création d'entités …

4.1. 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.

Eclipse version 1.00 possède plusieurs perspectives :

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

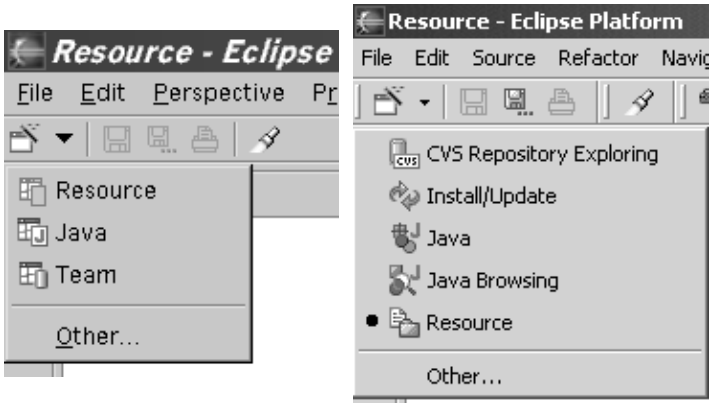
Eclipse version 2.00 possède les perspectives suivantes :

Perspective	Icône	Rôle
Debug		Débugueur
Java		Ecriture de code Java
Java Browsing		Navigation des la hiérarchie et les éléments des classes
Java Type Hierarchy		
Plug−in Development		Création de plug−in
Resource		Gestion du contenu du workspace
Install/update		Installation et mise à jour de plug in via le web
CVS		Gestion du travail collaboratif avec CVS

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

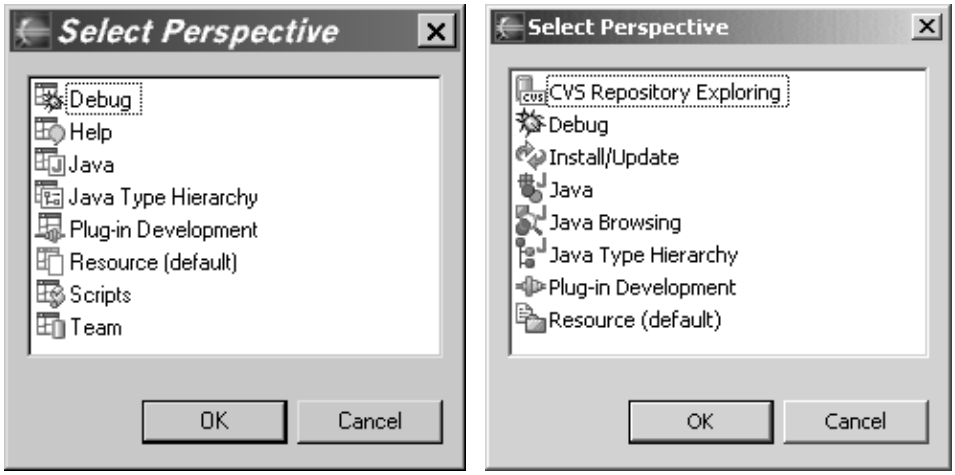
- Cliquer sur l'icône dans la barre des perspectives
- Utiliser l'option "Ouvrir" du menu "Perspective" (Eclipse 1.0) ou l'option "Ouvrir la perspective" du menu "Fenêttre" (eclipse 2.0)

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.

4.2. Les vues et les éditeurs

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

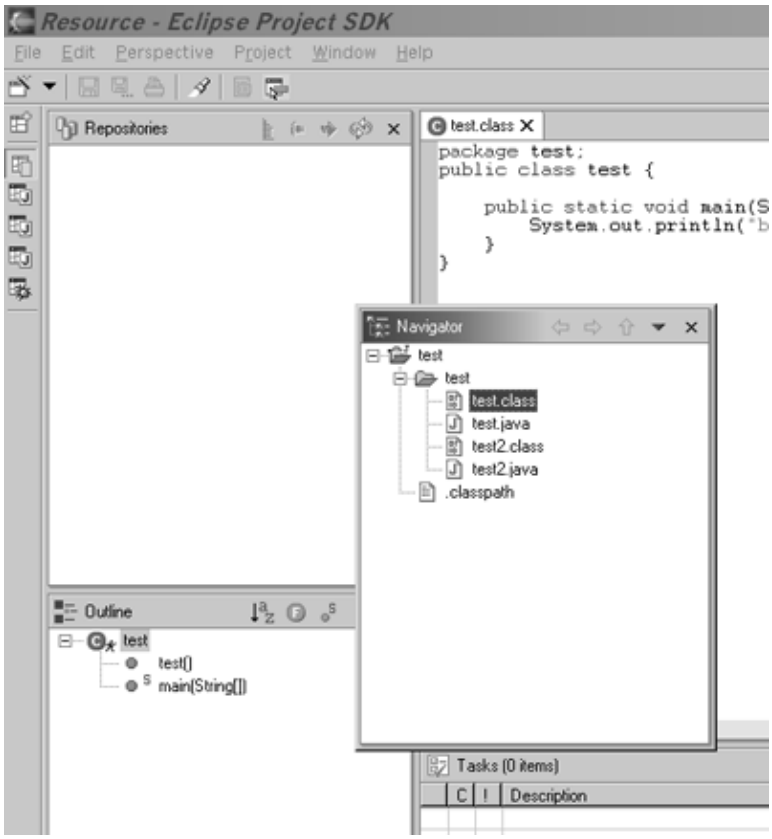
- les vues
- les éditeurs

Une vue permet de visualiser et de sélectionner des éléments. Il ne peut y avoir qu'une seule vue particulière dans une même perspective. Plusieurs vues différentes peuvent être rassemblées dans une même sous fenêtre. L'accès à chaque vue se fait 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.

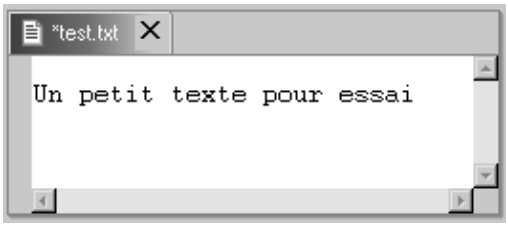
Avec Eclipse 1.0 sous Windows, les vues peuvent être extraites du workbench pour devenir des fenêtres indépendantes. Pour cela, il faut cliquer sur la barre de titre de la vue, en maintenant le bouton de la souris enfoncé, effectuer un glissement avec la souris vers une zone non empilable (sur un éditeur, 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.

4.2.1. 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.



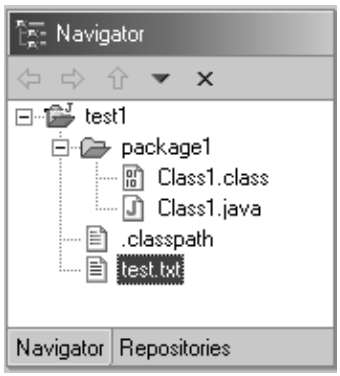
Pour fermer l'éditeur contenant l'élément en cours, il suffit de cliquer sur l'icône de l'onglet.

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 workbench pour fermer 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.

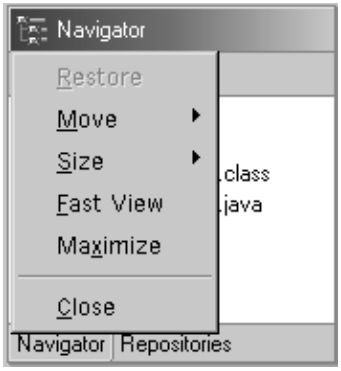
4.2.2. les vues

Les vues permettent de présenter 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.

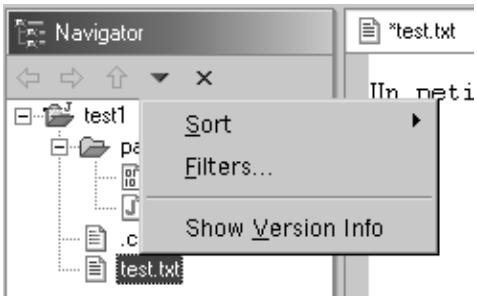


Les vues possèdent deux menus :

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.



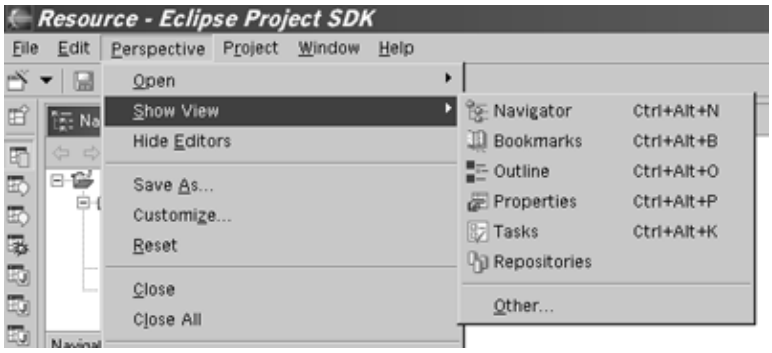
le second menu est activable en cliquant sur l'icône en forme de triangle dont la base est en haut  . Les options de ce menu concerne le contenu de la vue notamment le tri ou le filtre.



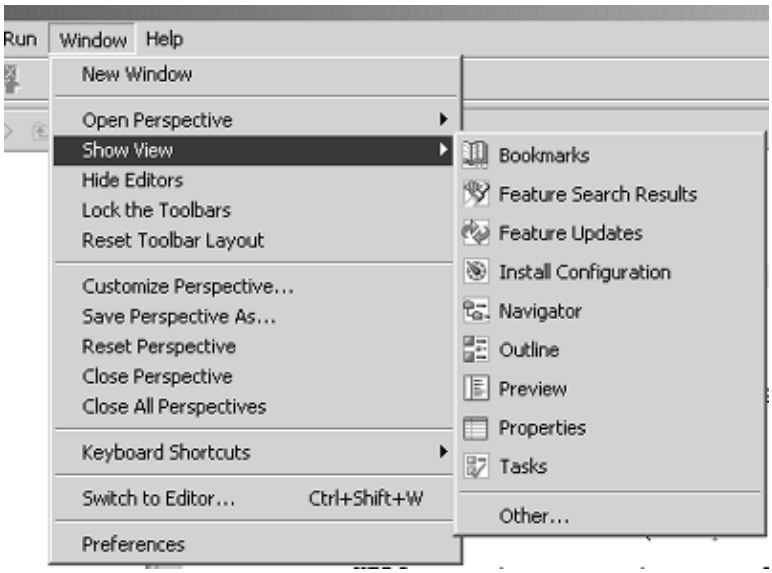
4.3. Organiser les composants de la perspective

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".

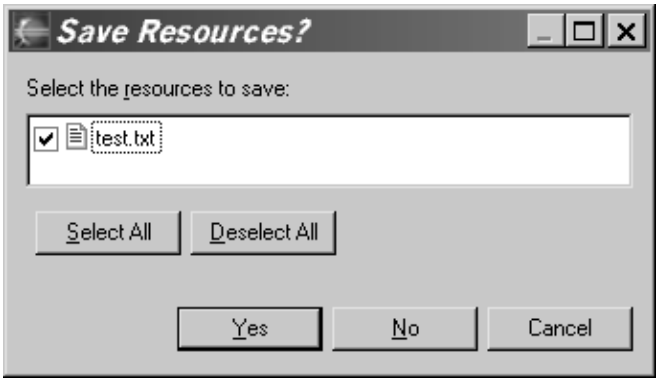


4.4. Fermer le workbench

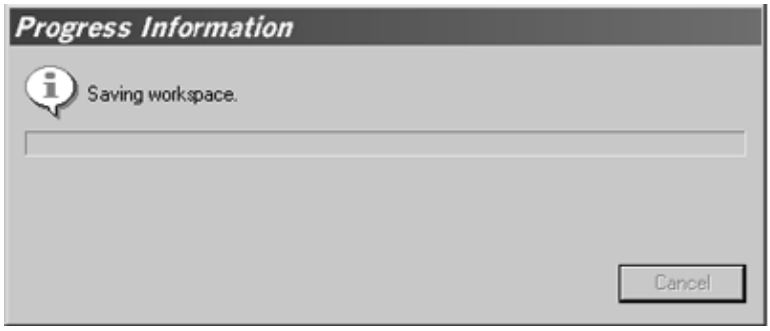
Pour fermer le workbench et donc quitter Eclipse il y a deux possibilités :

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

Si des ressources doivent être sauvegardées, une boîte de dialogue apparaît contenant ces ressources. Il faut indiquer celles qui doivent être enregistrées.



A sa fermeture, Eclipse enregistre certaines données dans le workspace.



5. Les fonctions pratiques du workbench

Eclipse fournit dans le workbench 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

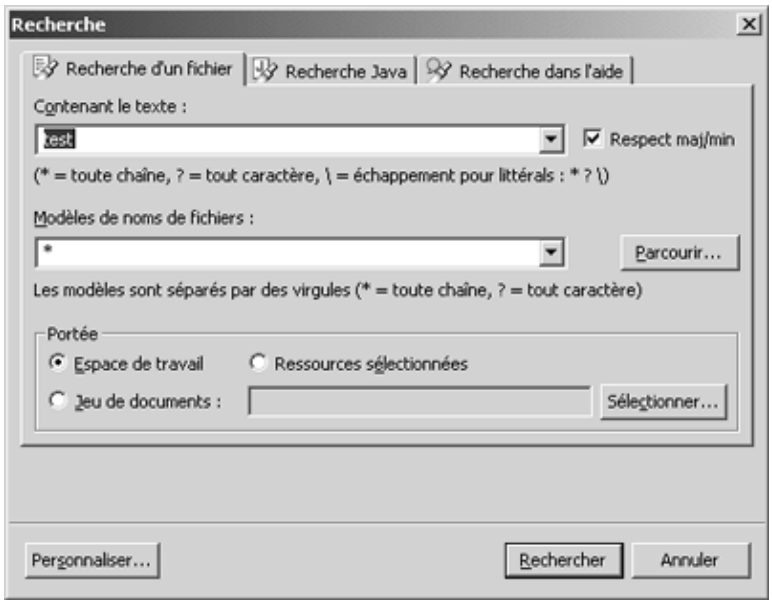
5.1. La fonction de recherche

Cette fonction de recherche permet d'obtenir une liste d'éléments qui contiennent une chaine 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 workbench. Une boîte de dialogue permet de saisir les critères de recherche.



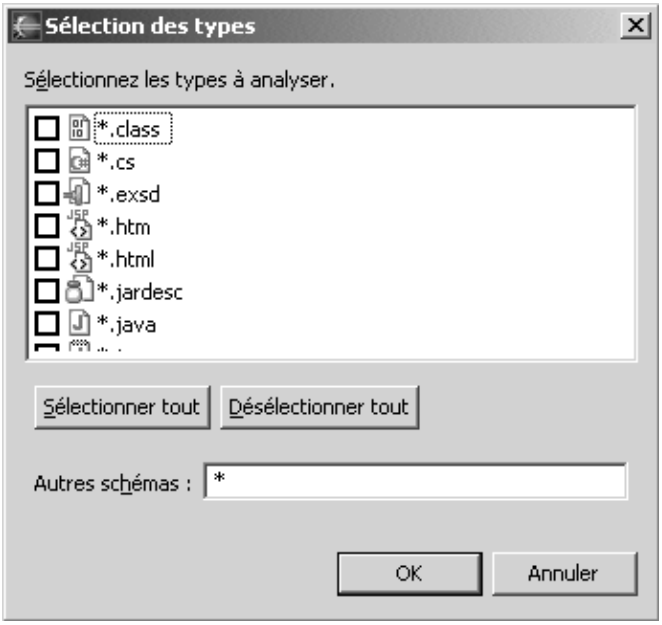
L'onglet "Recherche d'un fichier" permet de faire une recherche de fichiers contenant un texte respectant un motif. Ce motif peut être saisi ou sélectionné dans la liste déroulante à partir des précédents motifs recherchés.

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

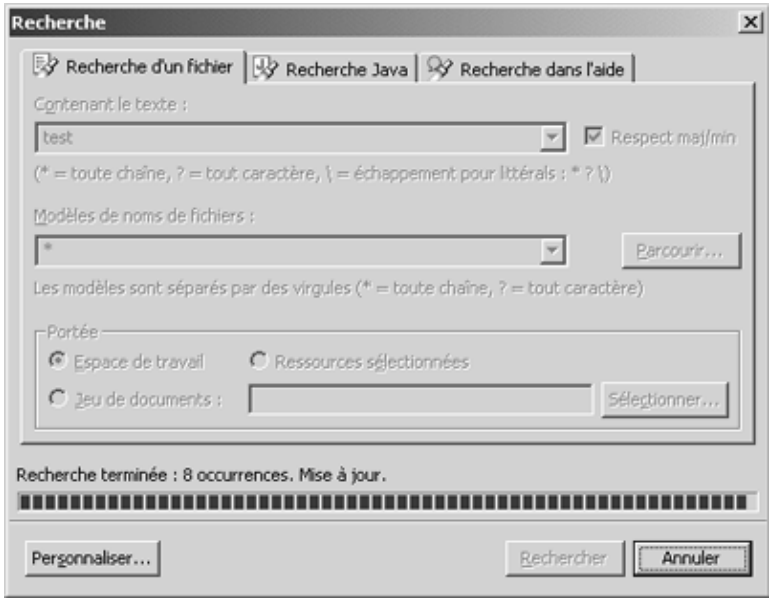
- * : représente zéro ou plusieurs caractères quelconques
- ? : représente un caractère quelconque
- \ : permet de déspécialiser le caractère *, ? et \

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 bouton "Parcourir" ouvre une boite de dialogue qui permet de sélectionner un ou plusieurs types prédéfinis.



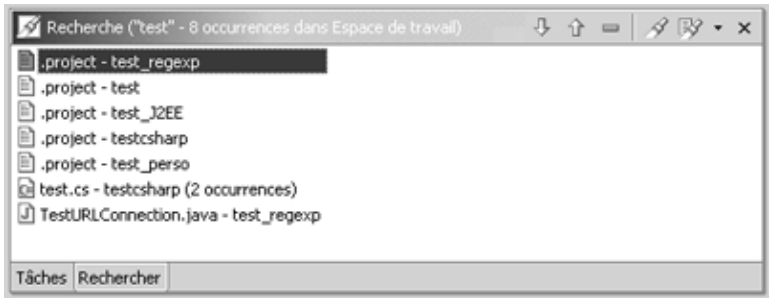
Pour lancer la recherche, il suffit de cliquer sur "Rechercher".



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

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.

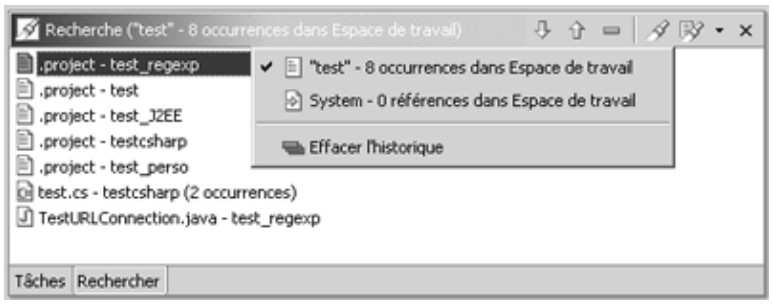


Le bouton  permet de passer à l'occurence suivante quelque soit l'élément qui la contienne. Lors du changement de l'élément qui contient l'occurence, celui ci est ouvert dans l'éditeur.

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.

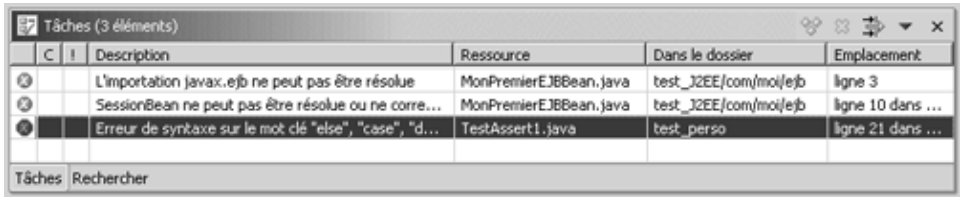
5.2. La liste des tâches

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
- des points d'arrêt pour le debuggage

Ces tâches peuvent être ou non associées à un élément du workspace. 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 le workspace dans la colonne "Dans le dossier".



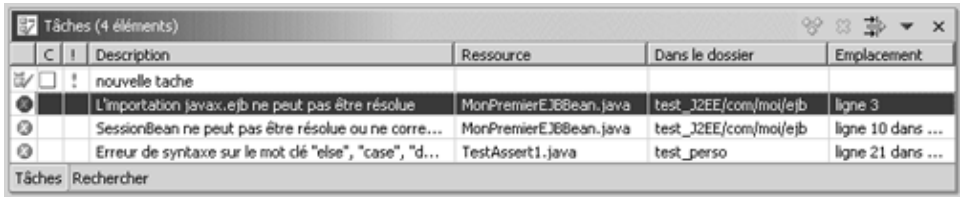
Il est possible d'accéder à l'élément associé à la tâche en double cliquant sur la tâche ou en sélectionnant l'option "Accéder à" du menu contextuel de la tâche. L'élément est ouvert dans l'éditeur avec le curseur positionné sur la ligne associée à la tâche.

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.

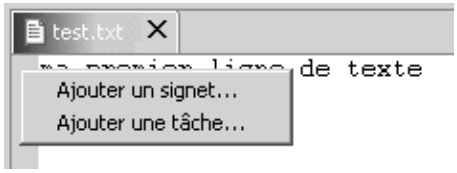
Un clic sur le bouton "OK" créé la nouvelle tâche.



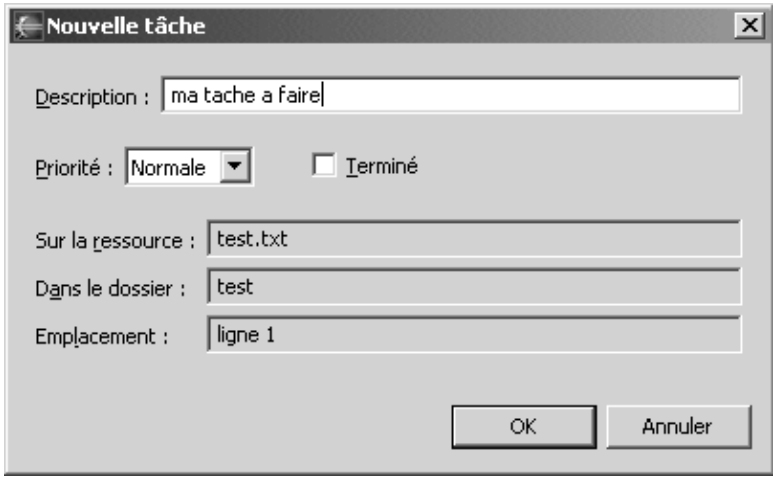
5.2.2. La création d'une tâche associée à un élément

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.

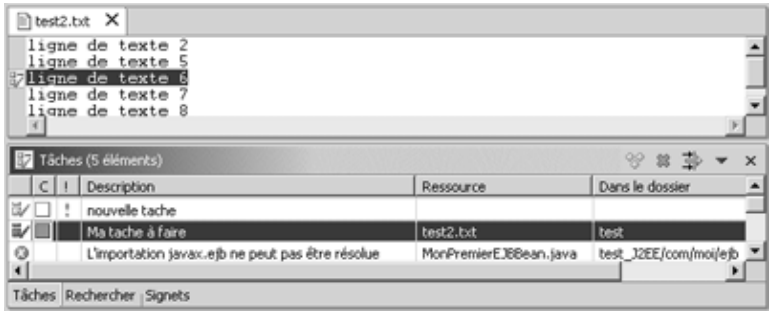
Dans la barre à gauche de l'éditeur, le menu contextuel contient l'option "Ajouter une tâche ..."



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



Un clic sur "OK" créée 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).

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 d'y accéder rapidement. Pour afficher la vue "Signets", il faut sélectionner l'option "Afficher la vue / Signets" du menu "Fenêtre" du workbench.

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

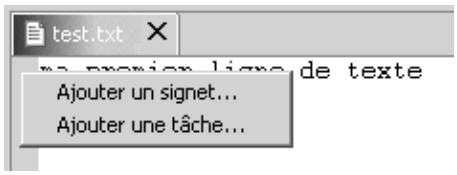
Il est aussi possible à partir d'un signet de sélectionner l'élément dans la vue "Navigateur" en utilisant l'option "Afficher dans le navigateur" du menu contextuel.

5.3.1. La création d'un signet

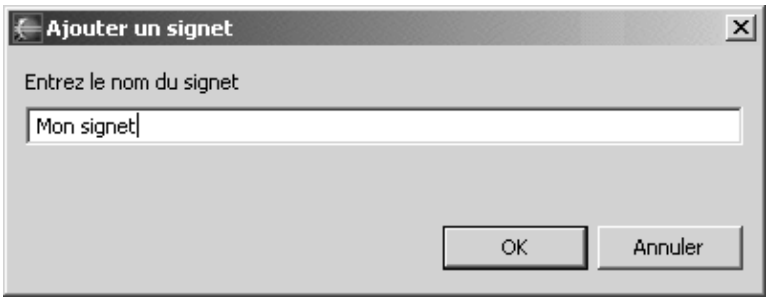
Un signet peut concerner un élément (un fichier) ou plus précisement une composante de cet élément (une position particulière dans le fichier).

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.

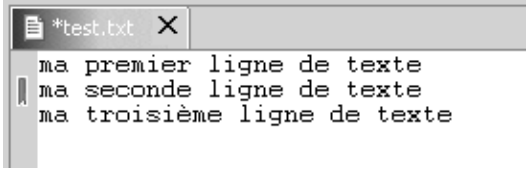
Pour créer un signet sur une ligne de l'élément, il suffit de positionner le curseur sur la ligne désirée dans l'éditeur et de sélectionner l'option "Ajouter un signet" du menu contextuel de la barre de gauche de l'éditeur.



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

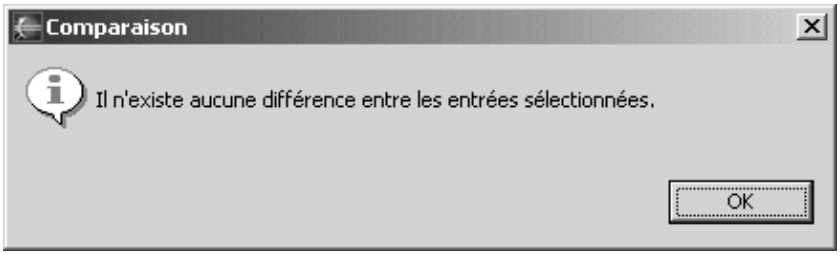
Pour supprimer un signet, il y a trois possibilités :

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

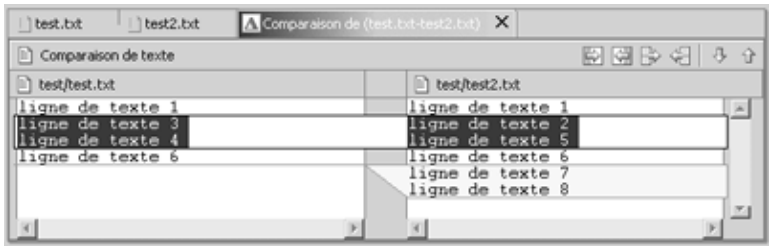
5.4. La comparaison d'éléments

Le workbench 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 :

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

6. Le Java Development Tooling (JDT)

Le 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 le workspace, 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

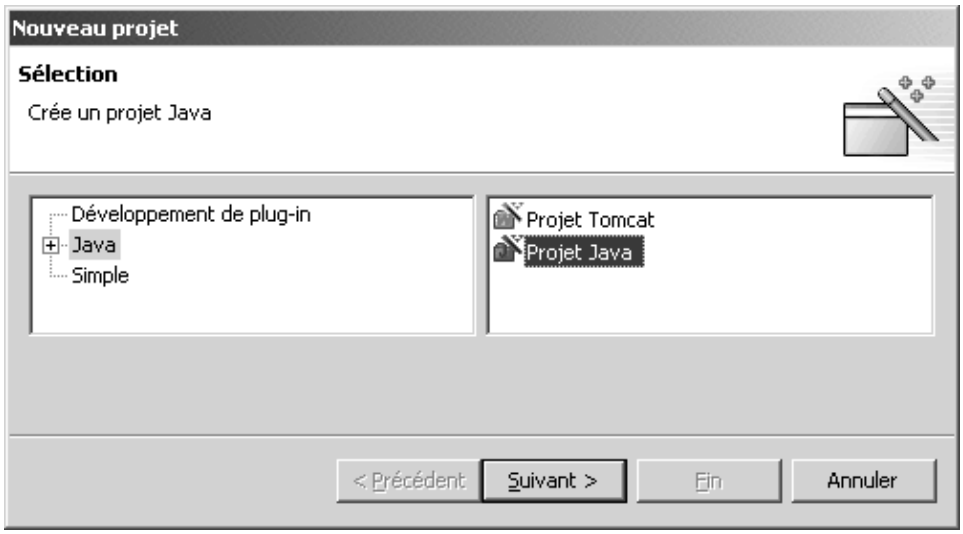
Pour pouvoir développer des entités en Java, il faut les regrouper dans un projet de type Java.

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 :

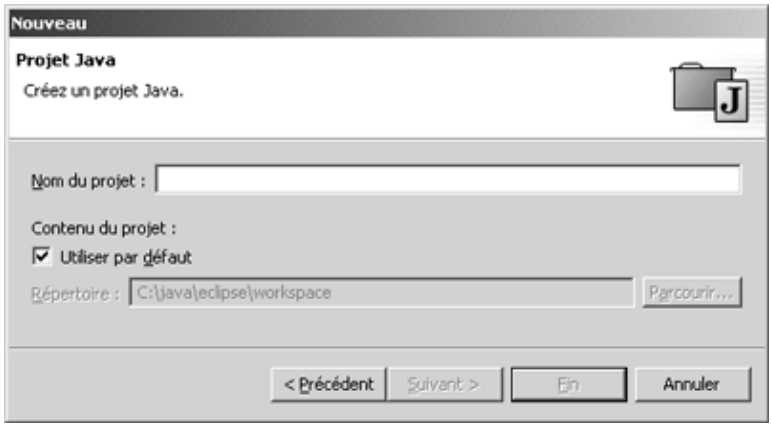
- sélectionner l'option "Projet" du menu "Fichier/Nouveau"
- sélectionner l'option "Nouveau/Projet" du menu contextuel de la vue "Packages"

L'assistant demande le type de projet à créer.

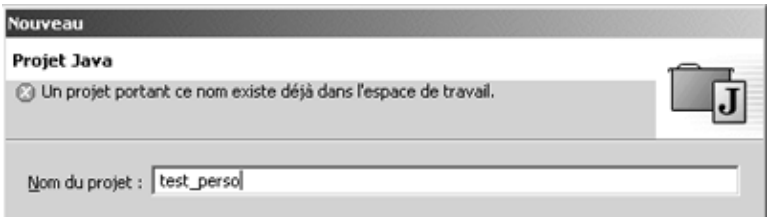


Pour demander directement la création d'un projet "Java", il suffit de cliquer sur l'îcone  de la barre d'outils.

L'assistant demande le nom du projet.

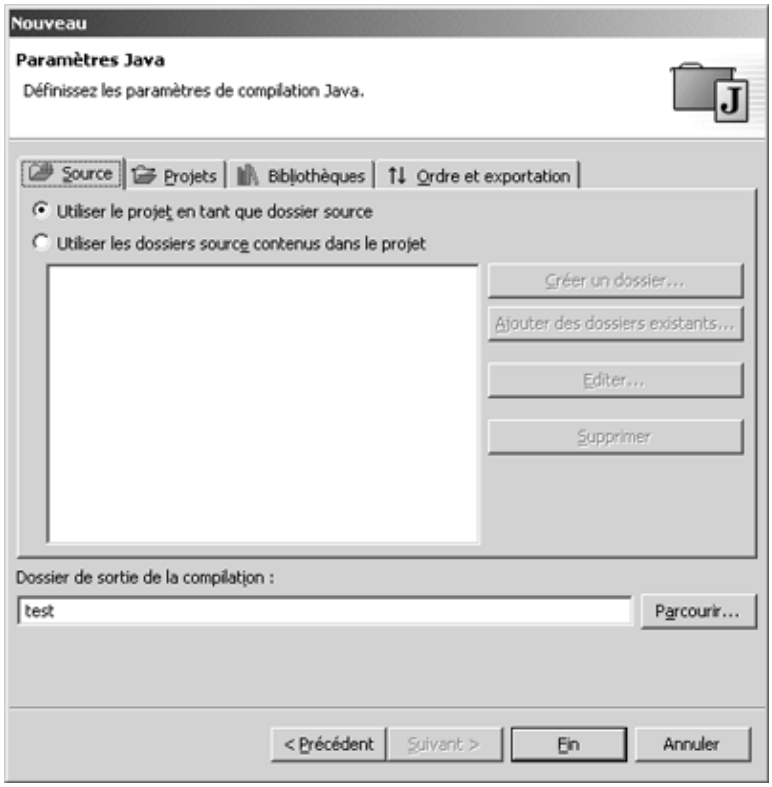


Ce nom de projet ne doit pas déjà être utilisé dans le workspace courant sinon un message d'erreur est affiché.



En cliquant sur "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" :



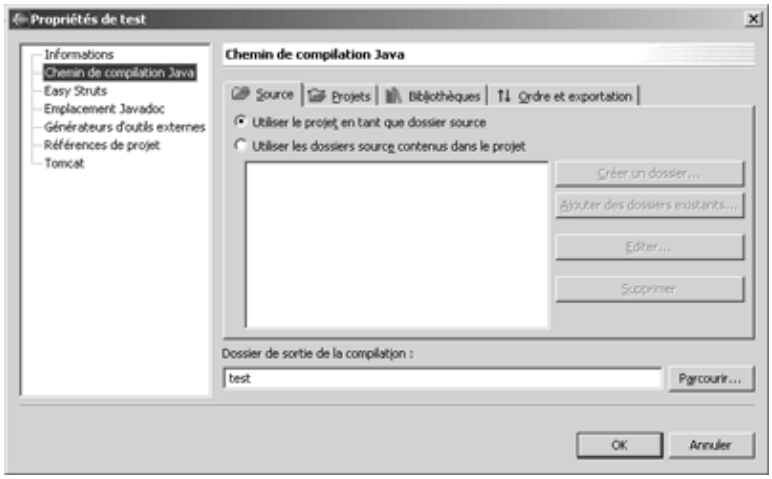
La modification de ces paramètres sera détaillée dans la section suivante. Une fois les paramètres modifiées, cliquer sur "Fin".

Le projet apparait dans la vue "Packages".

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"



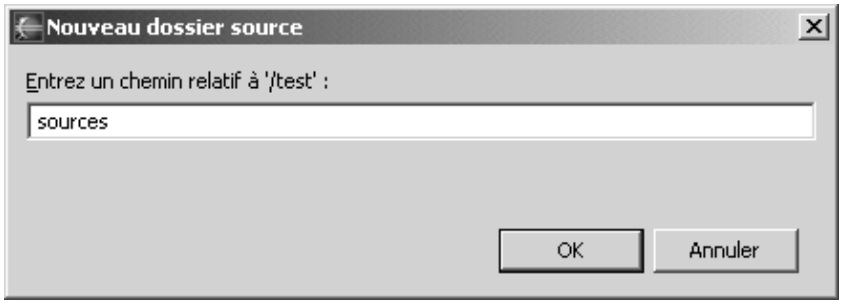
Les propriétés "chemin de compilation Java" sont regroupées dans quatres onglets :

Onglet	Rôle
Source	Permet de préciser le répertoire qui va contenir les sources
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).

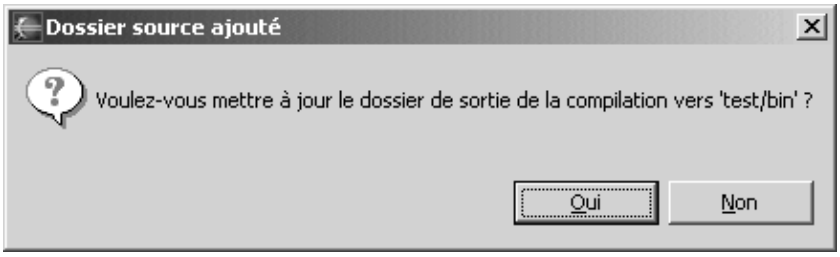
Pour stocker les ressources dans un répertoire dédié, il faut sélectionner l'option "Utiliser les dossiers sources contenus dans le projet". La liste permet de sélectionner le ou les répertoires.

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 "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.

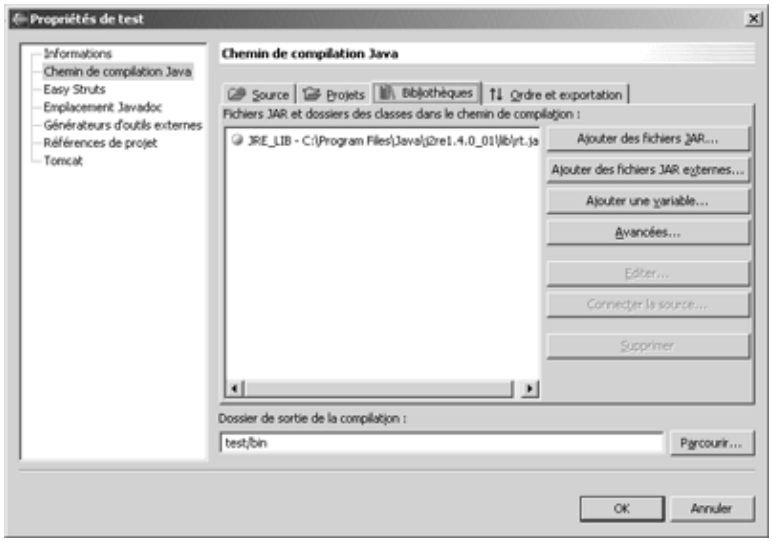


La réponse à la question est libre mais il est préférable de répondre "Oui".

L'onglet "Projets" permet d'ajouter des projets contenus dans le workspace au classpath.

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

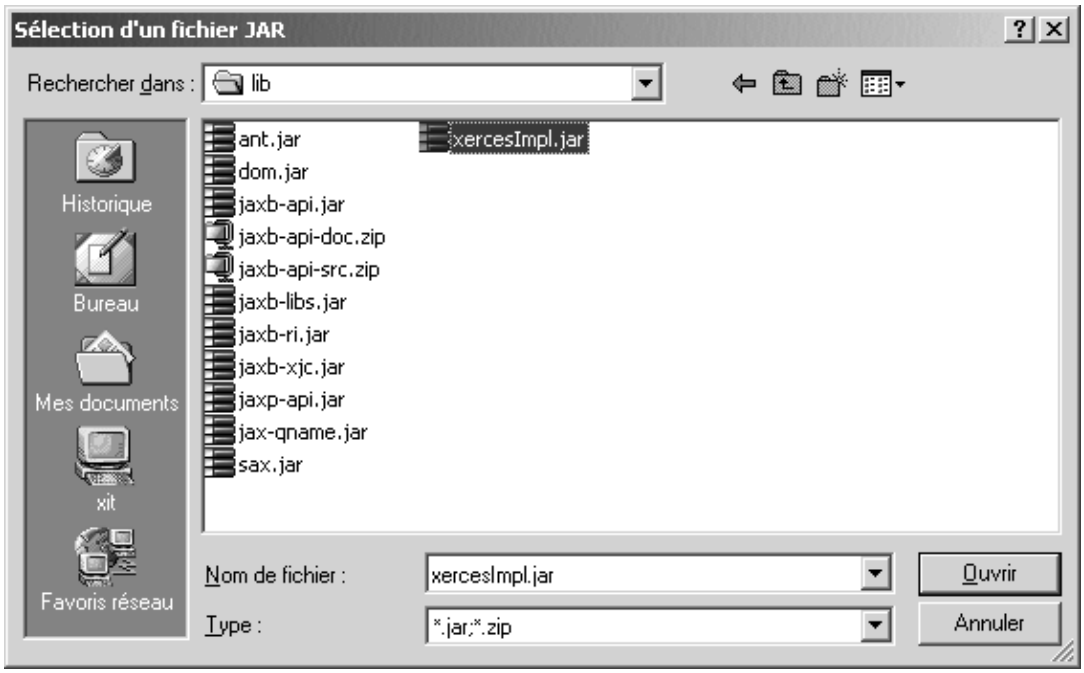
L'onglet "Bibliothèques" permet d'ajouter des bibliothèques externes au projet notamment des fichiers .jar.



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

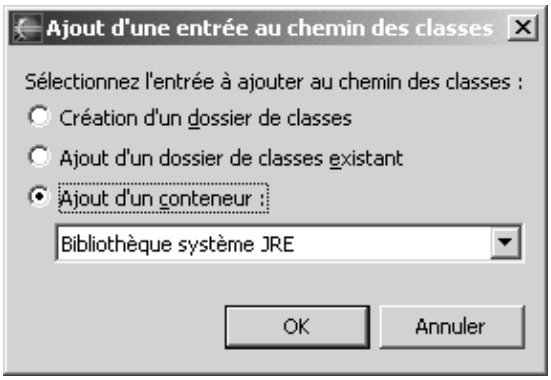
Pour ajouter sur une nouvelle bibliothèque contenue dans le workspace, il suffit de cliquer sur "Ajouter des fichiers jar".

Pour ajouter des fichiers jar qui ne sont pas contenus dans le workspace, il suffit de cliquer sur "Ajouter des fichiers jar externes".



Une boite de dialogue permet de sélectionner le fichier jar. En cliquant sur "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 contenirs) et de définir les ressources qui seront utilisables par les autres projets du workspace 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
- les interfaces

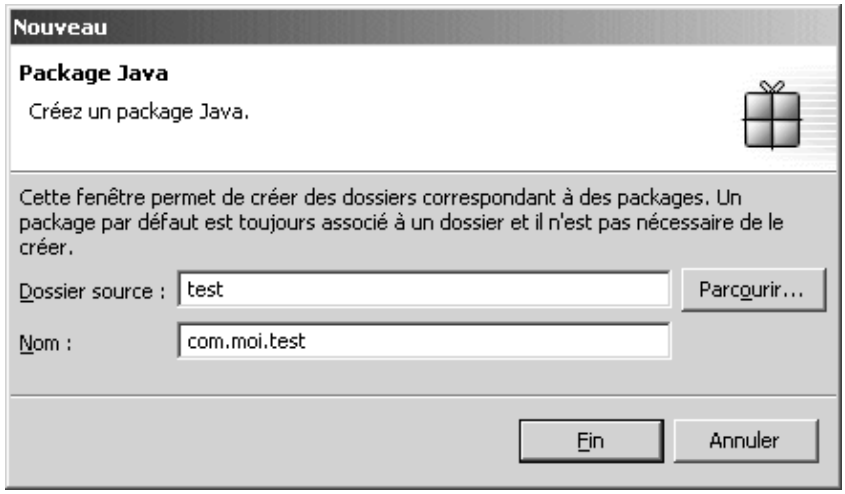
6.2.1. Les packages

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.



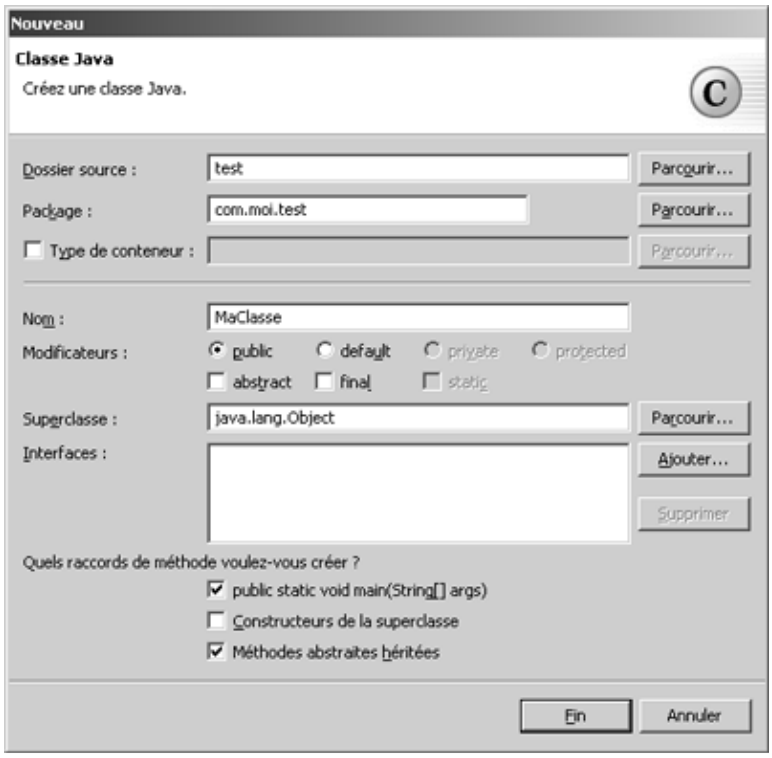
Cliquer sur "Fin", pour créer le nouveau package. Le package apparait dans la vue "Packages".

6.2.2. Les classes

La création d'une classe peut se faire :

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

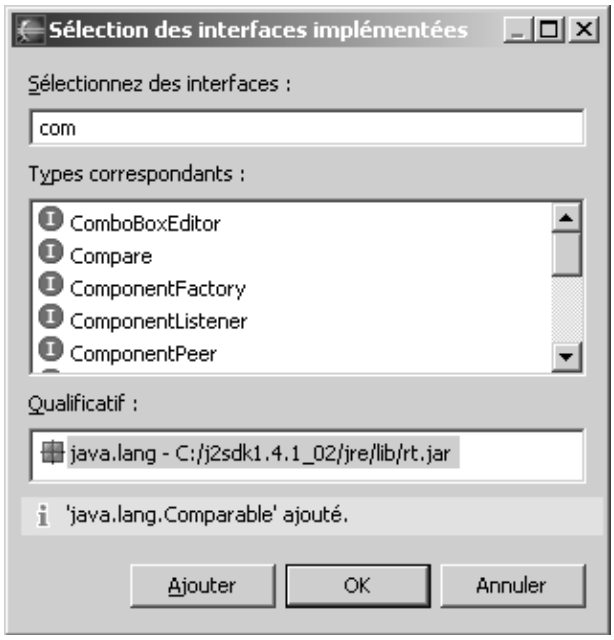
Un assistant facilite la création de la classe.



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 package, 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.

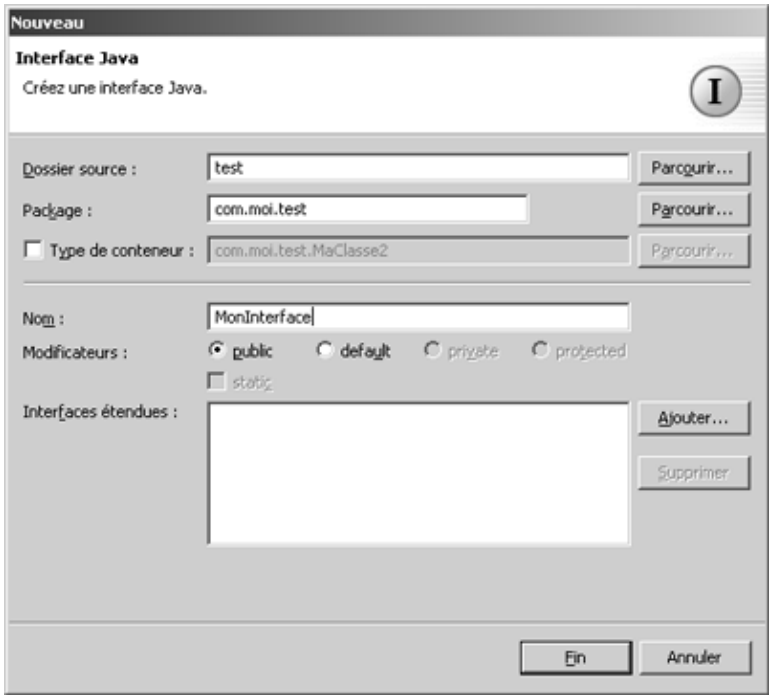
Une fois toutes les données utiles renseignées, il suffit de cliquer sur " Fin " pour que la classe soit générée et que l'éditeur s'ouvre avec le contenu de son code source.

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.

Une fois toutes les données utiles renseignées, il suffit de cliquer sur " 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

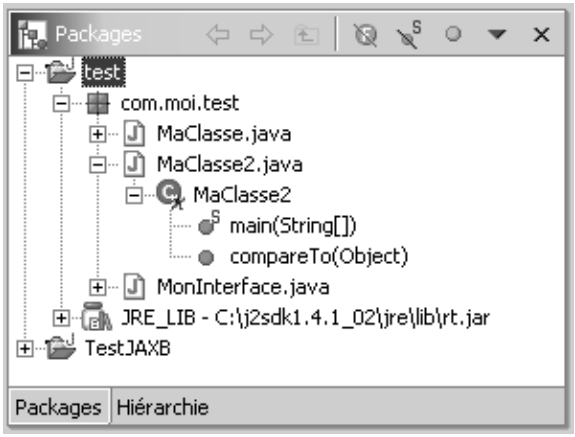
Le JDT contient les vues "Packages" et "Hiérarchie"

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.

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 :

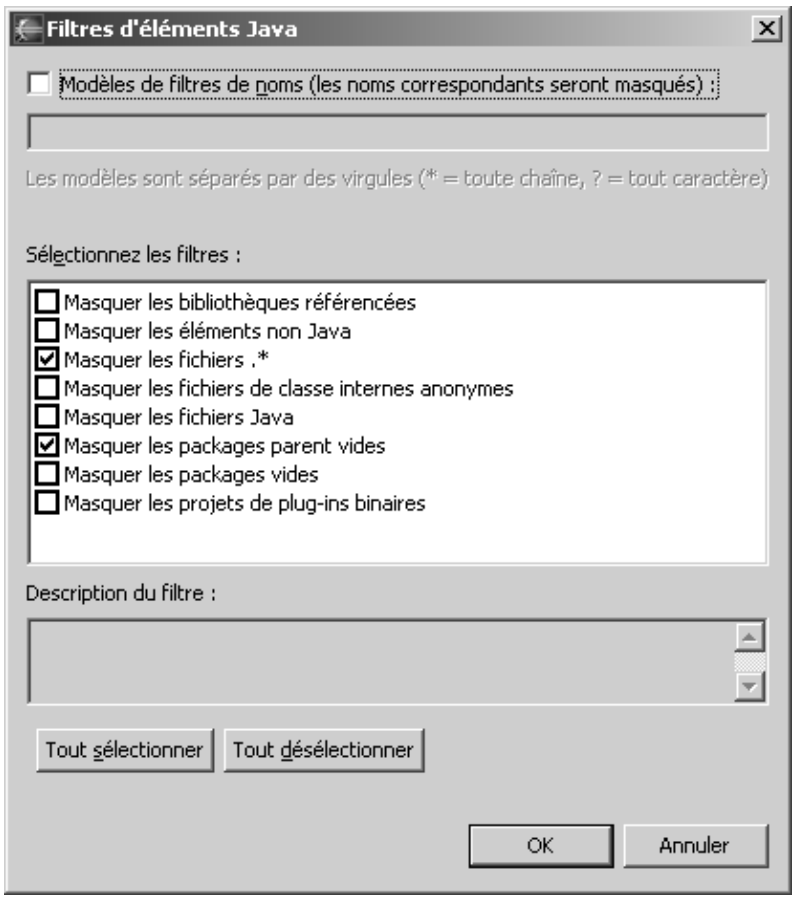
Icône	Type de l'élément
	Projet de type Java
	Package
	Elément Java : classe ou interface
	Interface Java
	Classe public Java
	Classe Java pouvant être exécutée (possédant une méthode main())
	Classe protected
	Classe package friendly
	Classe private
	Champ public
	Champ private
	Champ protected
	Champ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

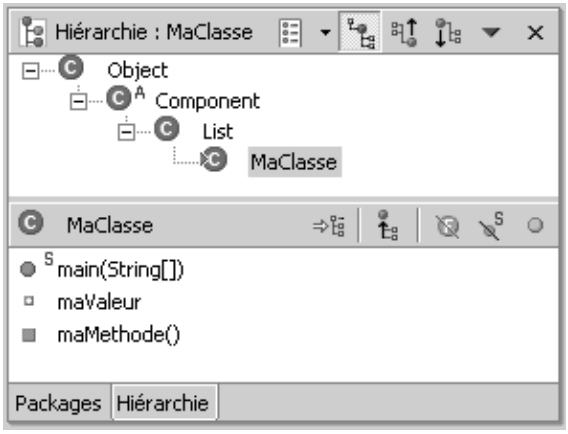
Cette vue affiche la hiérarchie d'un élément.

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

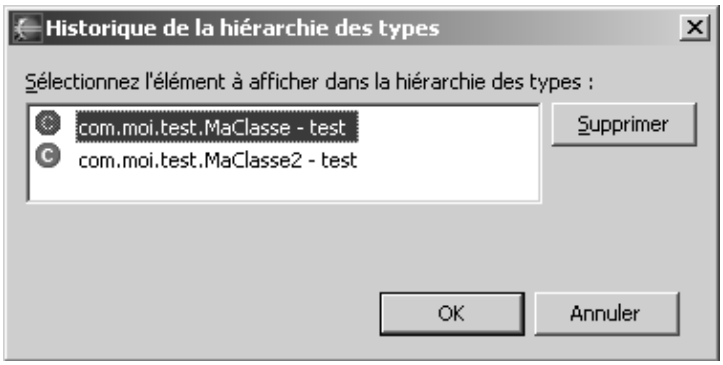
- dans la vue Package sélectionner l'élément et d'utiliser l'option "Ouvrir la hiérarchie des types" du menu contextuel.
- dans l'éditeur, utiliser le menu contextuel "Ouvrir la hiérarchie des types"

Elle se compose de deux parties :

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

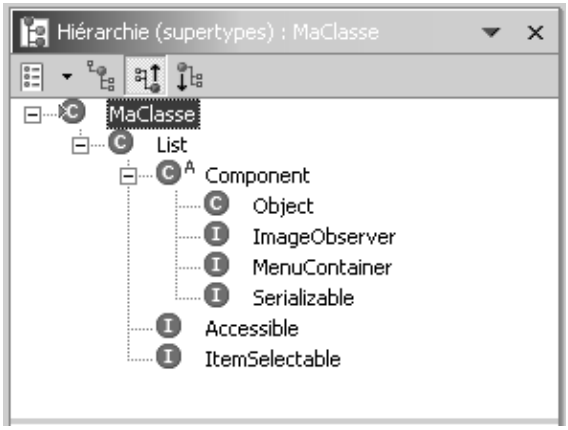


Le bouton  permet de sélectionner dans un historique un élément qui a déjà été affiché dans la vue.



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

6.4. 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 debogueur

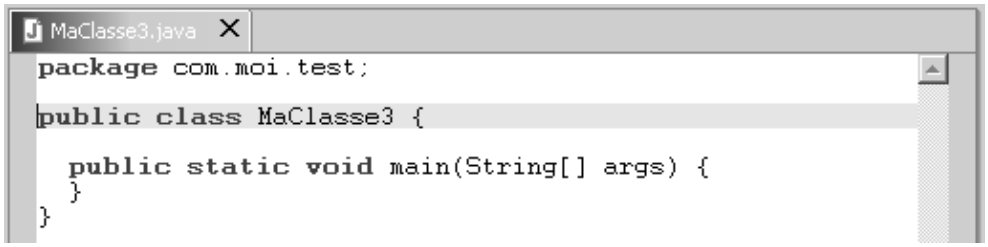
Pour ouvrir un élément dans l'éditeur, il y a deux façons principales :

- 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.

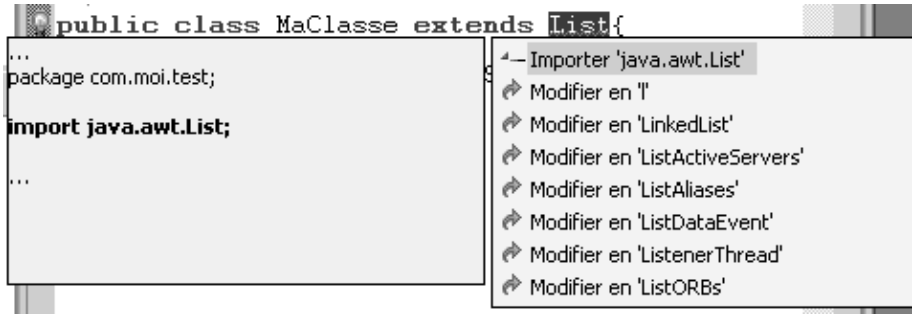
6.4.1. Utilisation de l'éditeur de code

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.

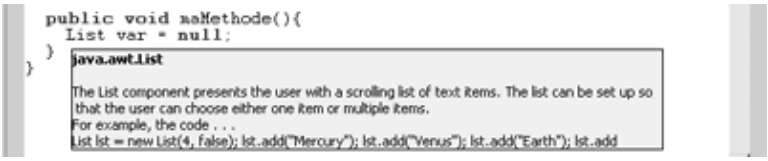
Par exemple, si l'on fait hériter une classe d'une classe dont le package n'est pas importé, un clic sur la petite ampoule jaune permet d'obtenir des propositions de corrections.



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

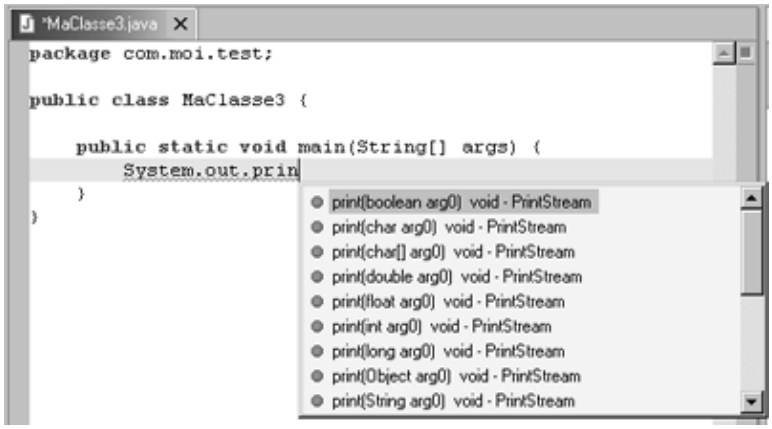
informations sur l'élément sous lequel est le curseur.



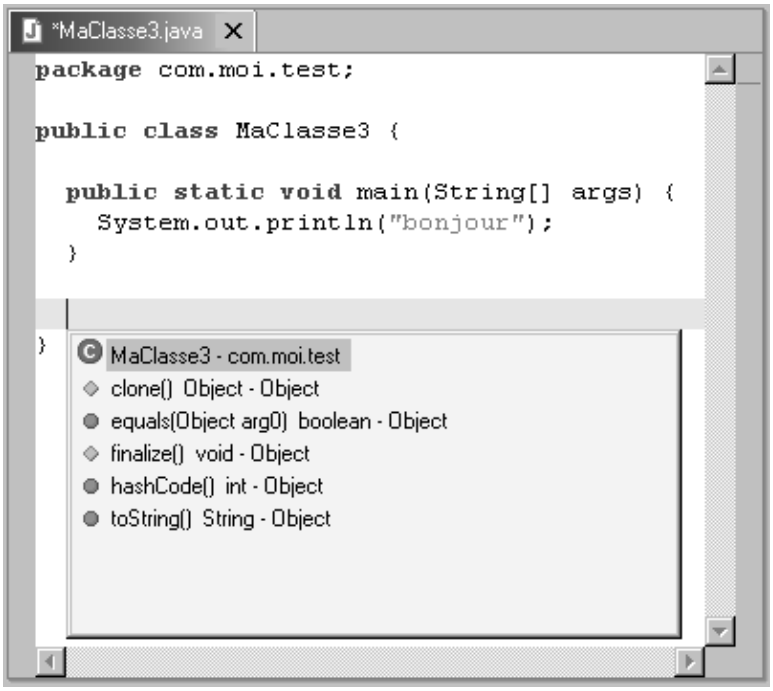
Une description plus détaillée peut être obtenue en positionnant le curseur sur l'élément et en appuyant sur la touche F2 ou en sélectionnant l'option " Afficher une description de type infobulles " du menu " Editer ".

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.

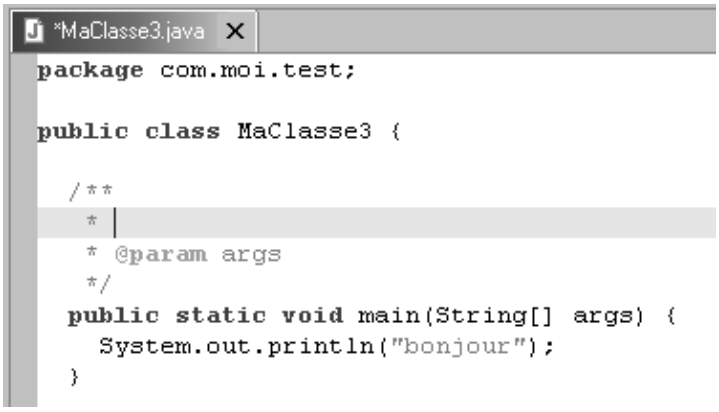


Cette fonction peut être appelée alors qu'aucune ou une partie du code à compléter est saisie.



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 ".

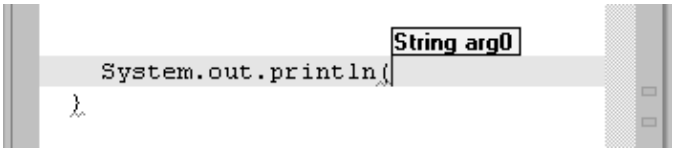


L'appel de la complétion de code en appuyant sur les touches " Ctrl " + " Espace " permet aussi de faciliter la saisie des commentaires de type java.

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.

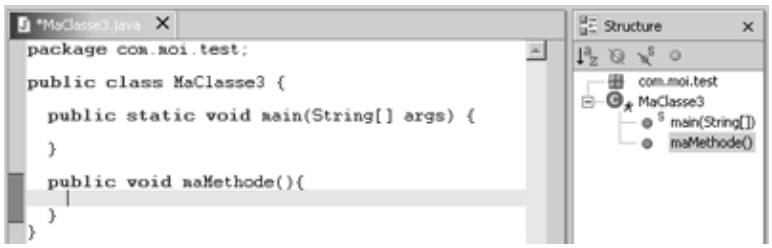
Pour utiliser cette fonction, il suffit d'appuyer sur les touches "Ctrl" + "Maj" + "Espace" en même temps dans l'éditeur pendant la saisie d'un ou des paramètres d'une méthode.



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. 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.

6.4.5. La coloration syntaxique

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

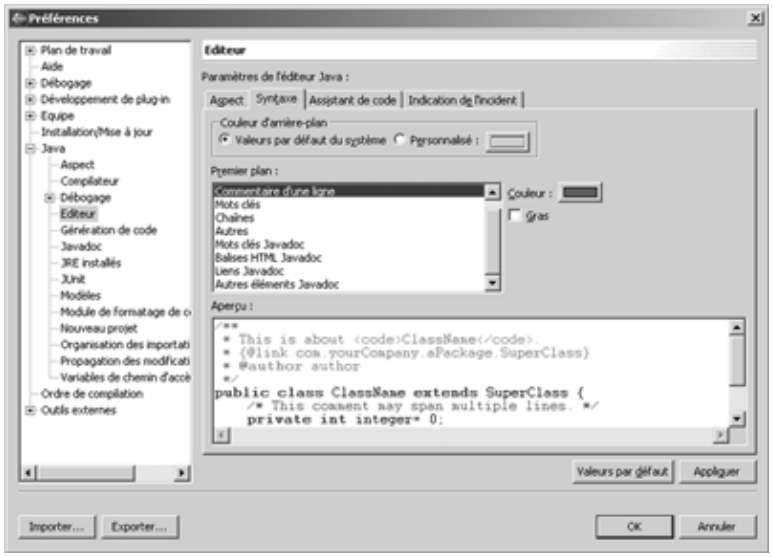
- les mots clés mots du langage sont colorés en violet gras
- les chaînes de caractères sont en bleu
- les commentaires sont en vert
- les commentaires Javadoc sont en bleu plus clair
- les balises Javadoc sont en gris
- les autres éléments sont en noir

Exemple :

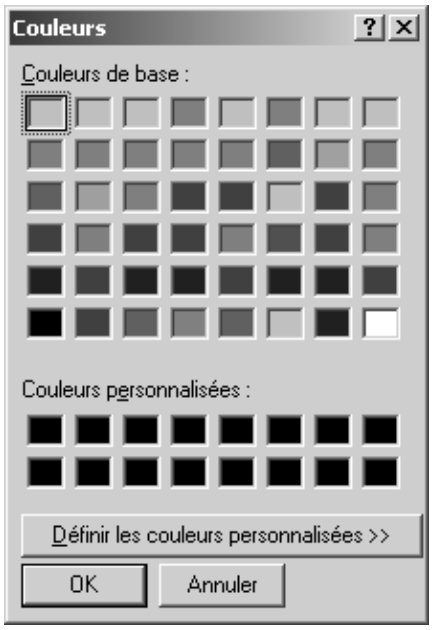
```
package com.moi.test;

import java.awt.List;

/**
 * @author user
 *
 * To change this generated comment edit the t
 * Window>Preferences>Java>Templates.
 * To enable and disable the creation of type
 * Window>Preferences>Java>Code Generation.
 */
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.



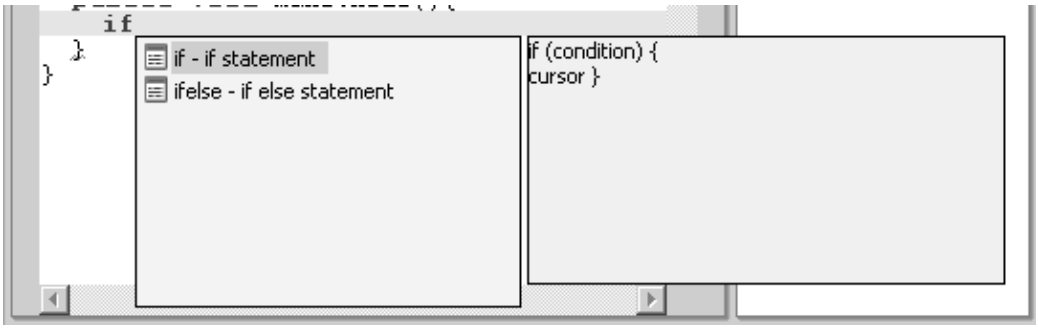
Il suffit de sélectionner l'élément concerné dans la liste déroulante et de sélectionner la couleur qui lui est associée en cliquant sur le bouton couleur. Une boîte de dialogue permet de sélectionner la nouvelle couleur à utiliser.



6.4.6. Utilisation des modèles

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

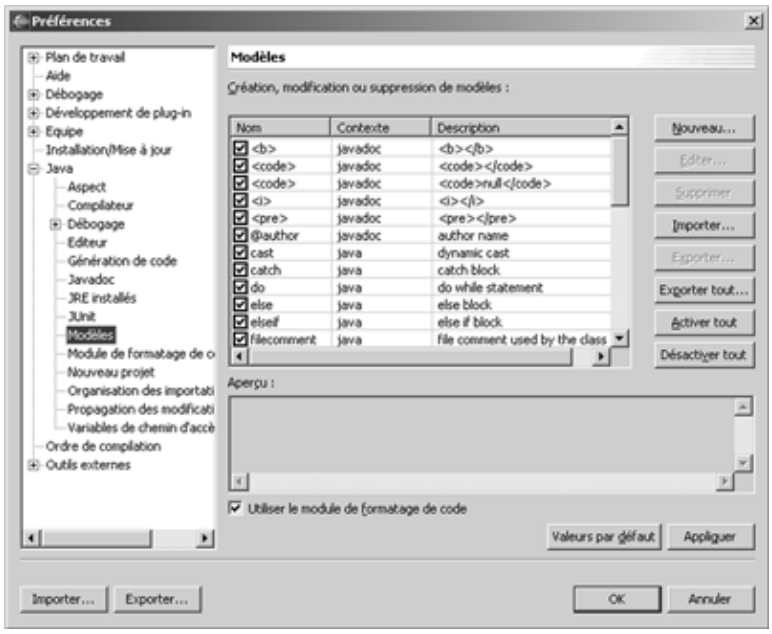
Exemple : avec le modèle if



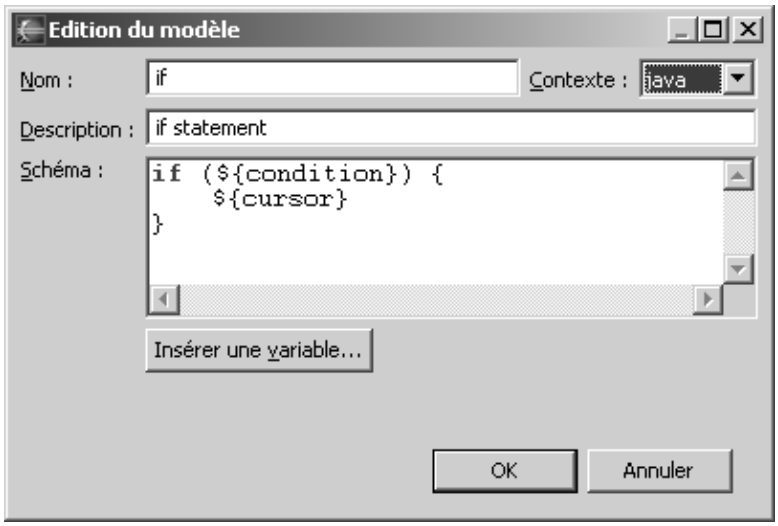
Il suffit de sélectionner le modèle à insérer pour qu'il soit immédiatement insérer dans le code.

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"



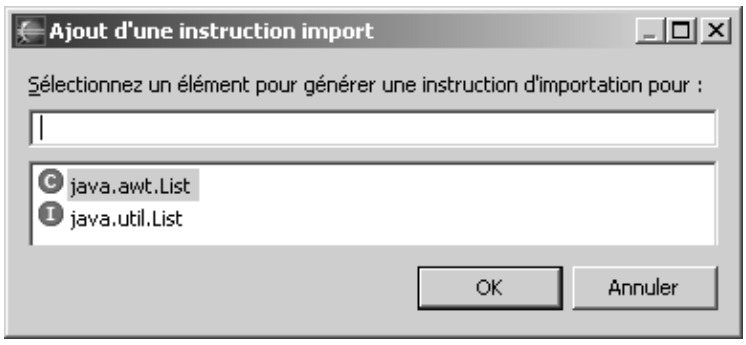
Pour modifier un modèle, il suffit de cliquer sur "Editer"



6.4.7. La gestion des importations

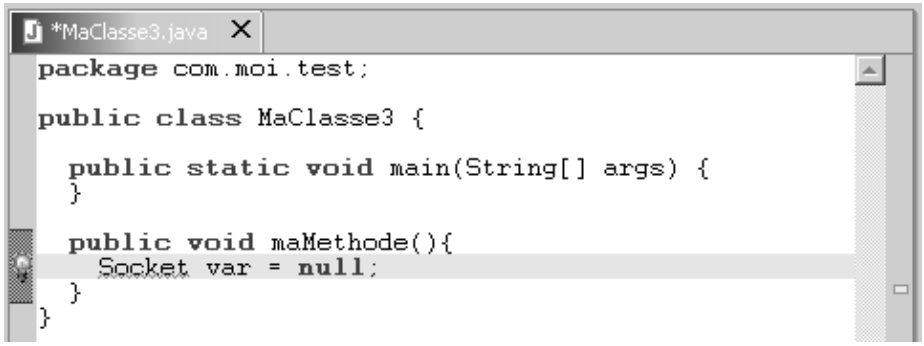
Il est possible de faire insérer la clause import pour un élément utilisé dans le code. Pour cela, il suffit de mettre le curseur sur l'élément concerné et de sélectionner l'option " Source/Ajout d'une instruction d'import " ou d'appuyer sur les touche "Ctrl" + "Maj" + "M".

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.

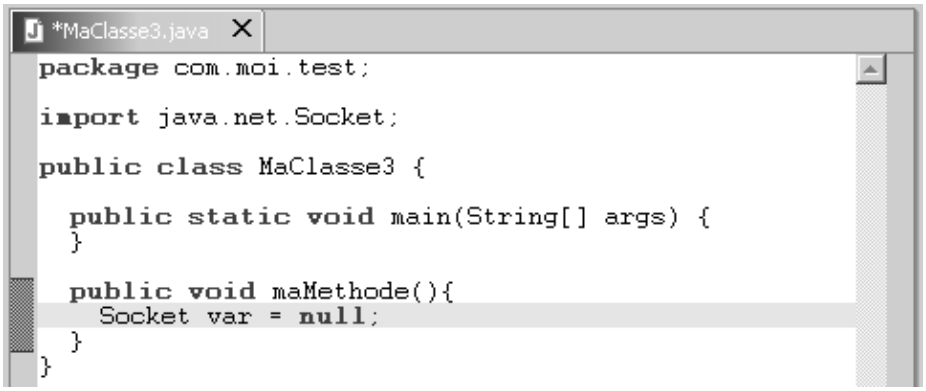


La fonctionnalité "Organisation des importations" est très pratique car elle permet d'insérer automatiquement les clauses imports avec les package requis par le code.

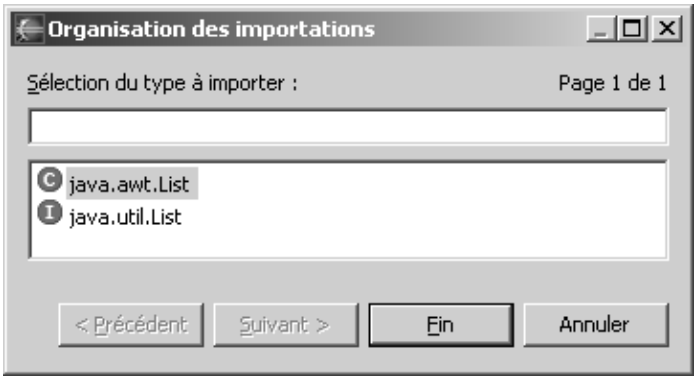
Par exemple : une variable de type Socket est déclarée, sans que le package java.net ne soit importé



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

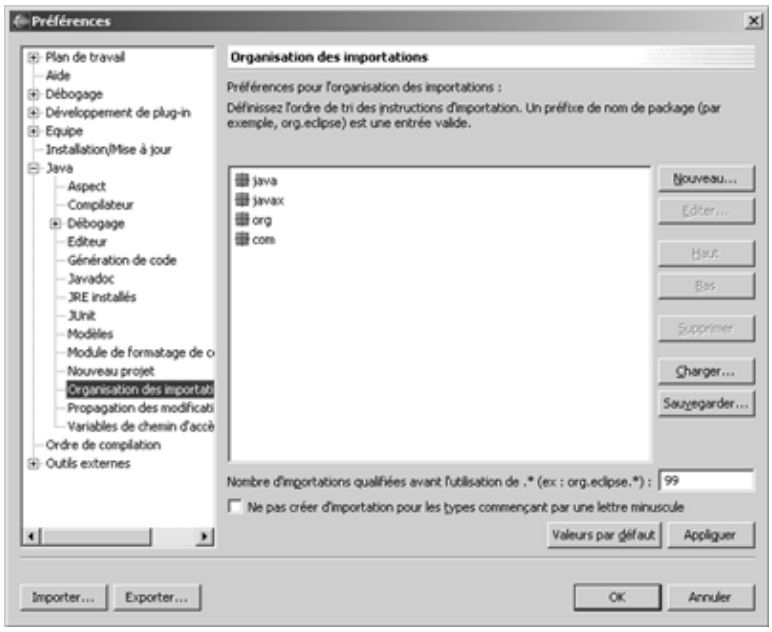


La clause import est insérée dans le code. Si un élément est contenu dans plusieurs packages, une boîte de dialogue demande la sélection du type à importer.



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

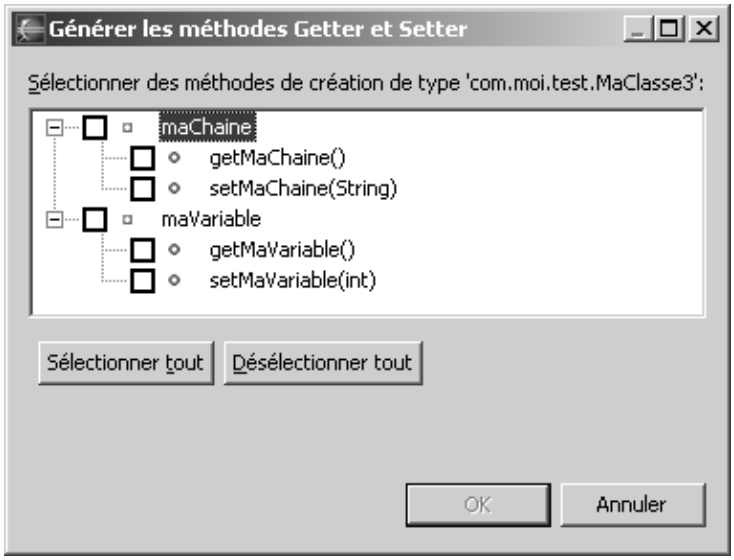
Certains réglages de cette fonctionnalité peuvent être effectués dans les préférences (menu "Fenêtre/Préférences"). Il suffit de sélectionner " Java/Organisation des importations " dans l'arborescence.



6.4.8. 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



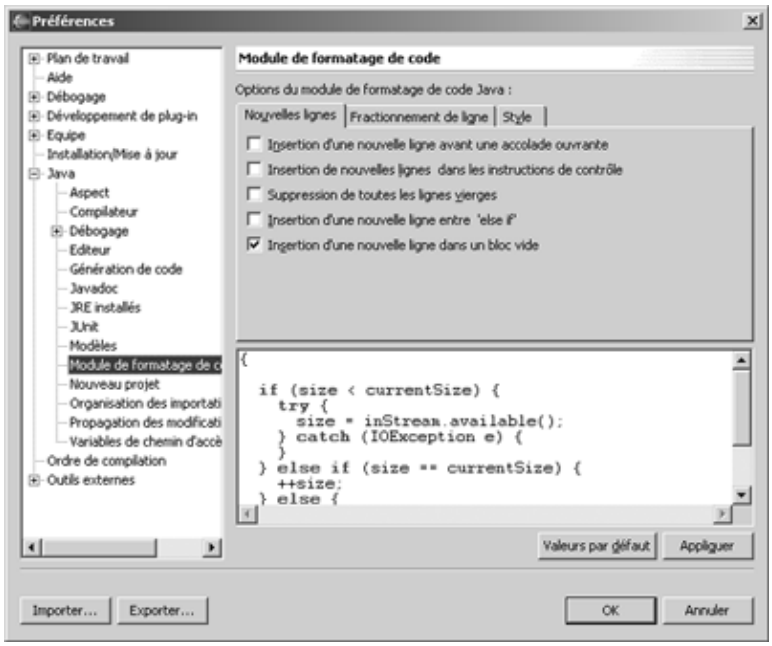
Il suffit de sélectionner les méthodes nécessaires pour qu'une génération par défaut soit effectuée dans le code.

6.4.9. 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.

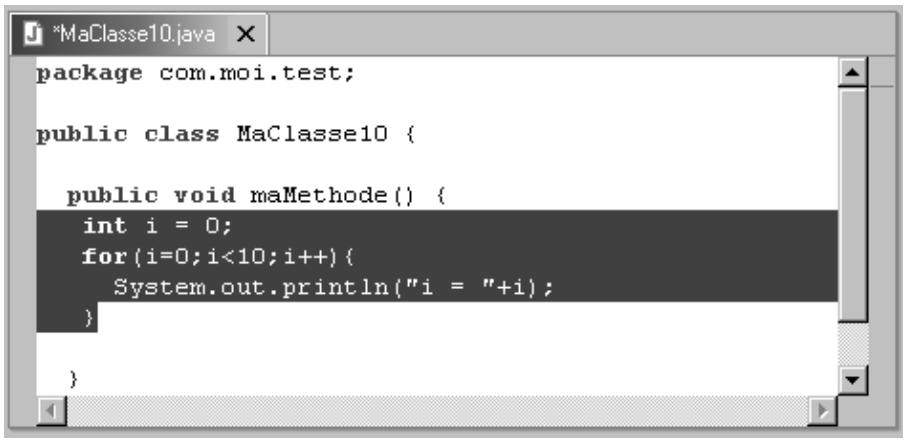
Les règles utilisées pour formater sont définies dans les préférences (menu "Fenêtre/Préférences").

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

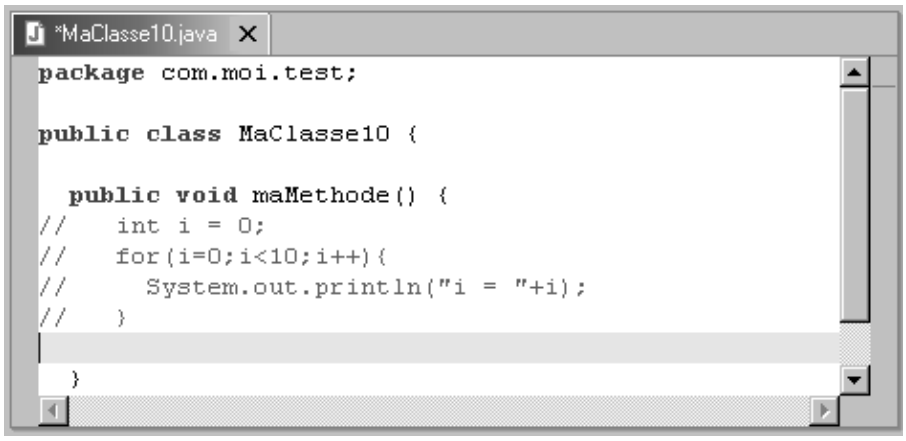


6.4.10. Mise en commentaire d'une portion de code

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.

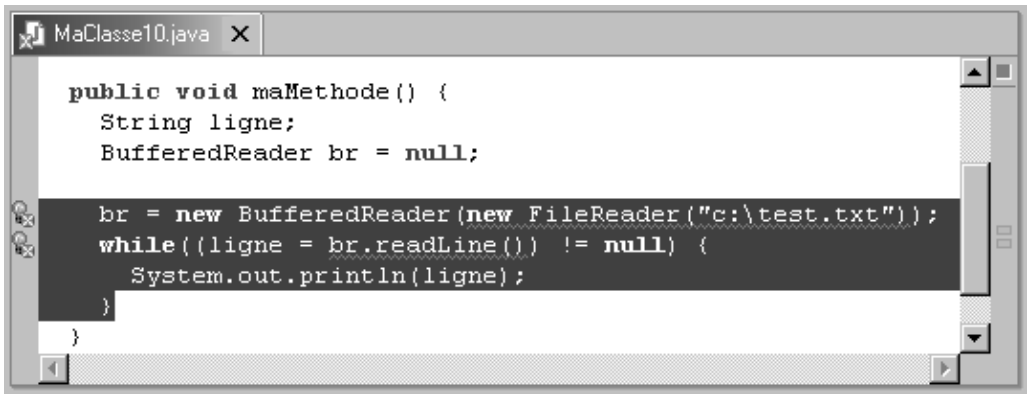


Pour supprimer les commentaires sur une portion de code, il suffit de sélectionner la portion de code et d'utiliser l'option " Source / Supprimer la mise en commentaire " du menu contextuel.

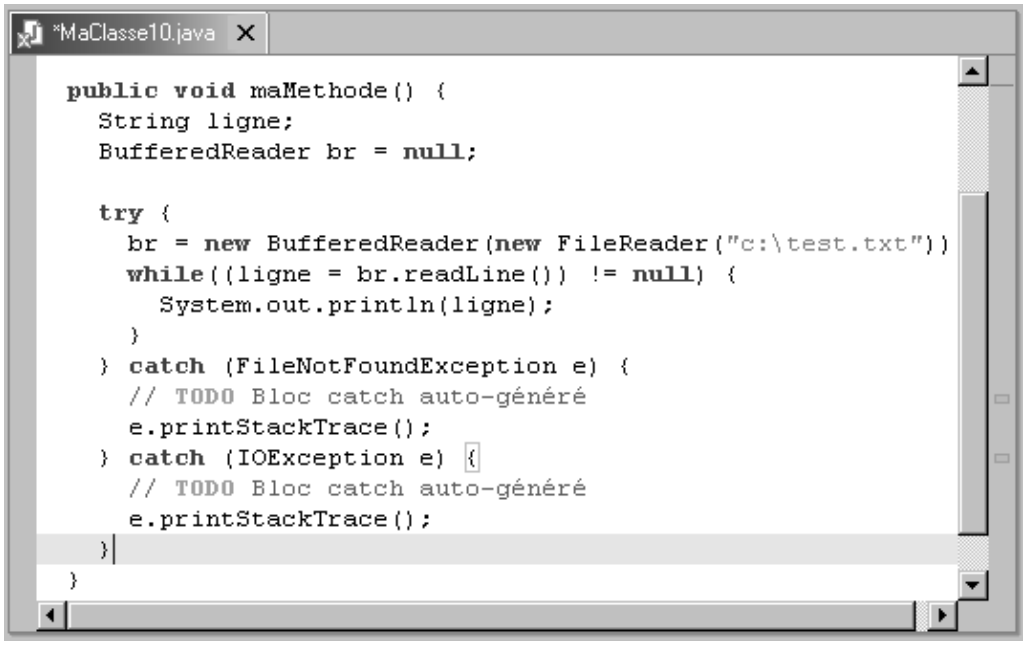
6.4.11. Protéger une portion de code avec un bloc try/catch

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.

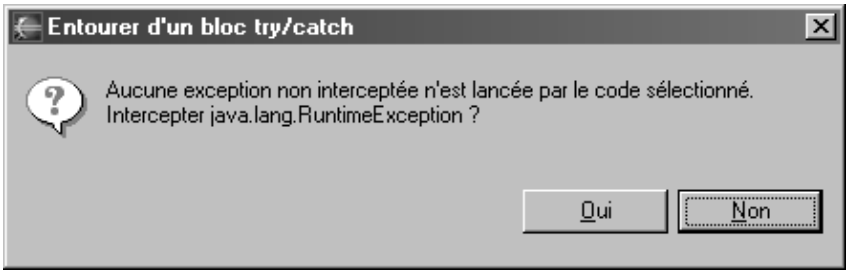


Puis utiliser l'option " Source / Entourer d'un bloc try / catch ". L'éditeur analyse le code et génère la cause try / catch qui va capturer toutes les exceptions qui peuvent être levées dans le bloc de code sélectionné.



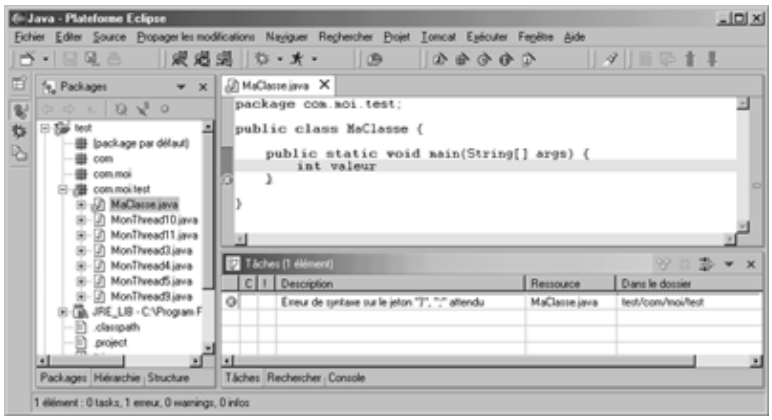
Chacune des instructions catch est marquée avec une tache " bloc catch auto−généré " pour indiquer au développeur d'ajouter le code nécessaire au traitement de l'exception.

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.



6.4.12. Les erreurs

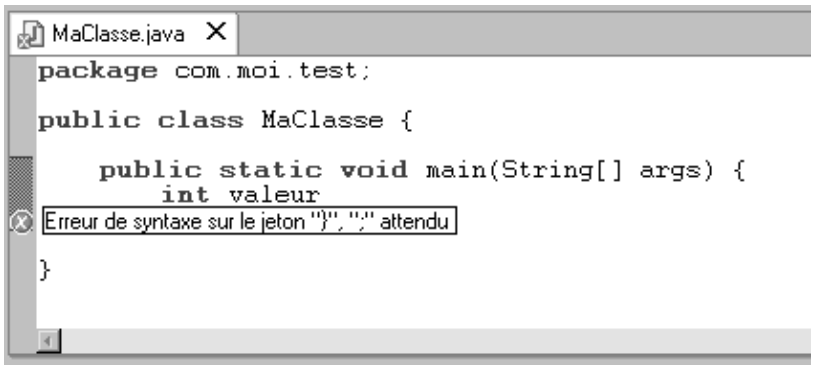
Lors de la sauvegarde du fichier, celui ci est compilé et les erreurs trouvées sont signalées grâce à plusieurs indicateurs dans différentes vues.



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.

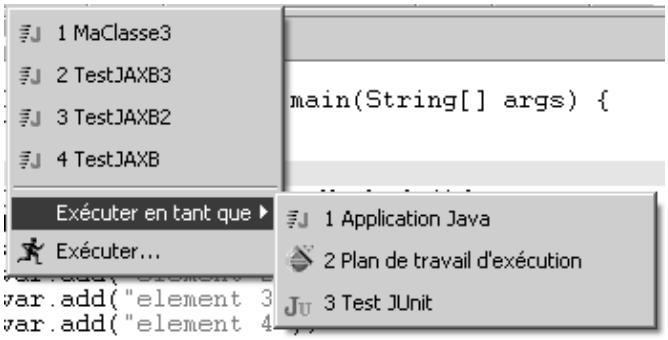


Les boutons  et  permettent respectivement de se positionner sur l'erreur suivante et sur l'erreur précédente.

6.5. Exécution d'une application

Dans la vue 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.

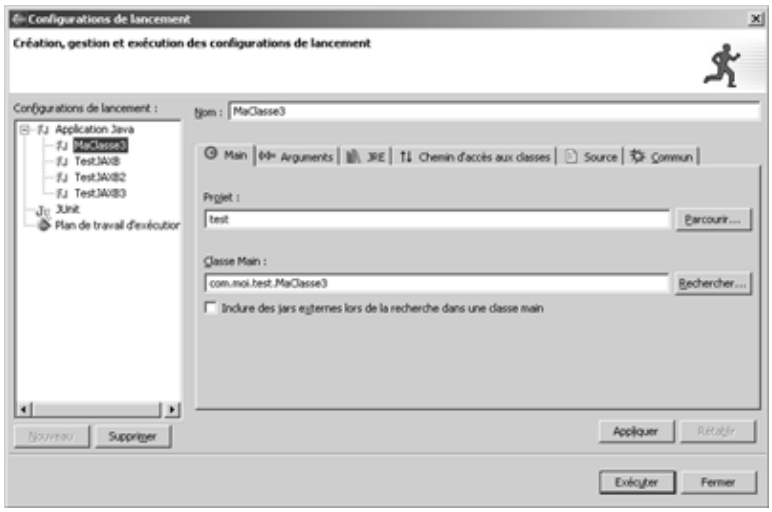


Ce menu déroulant propose différentes options :

- 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'éxécution.



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.

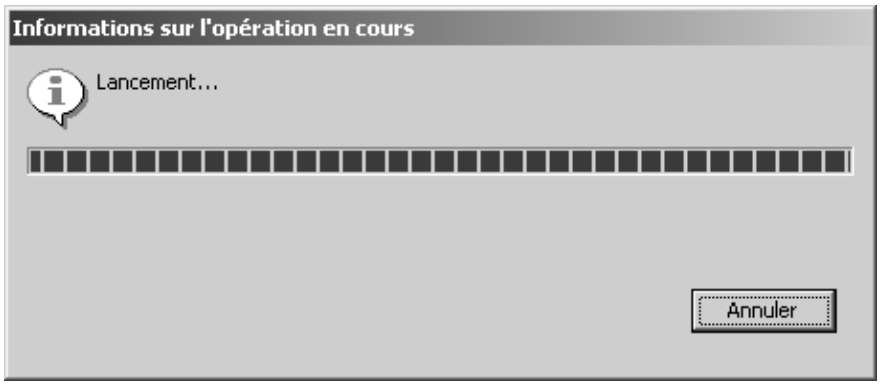
L'onglet "Arguments" permet de préciser les arguments passés à l'application et à la machine virtuelle.

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

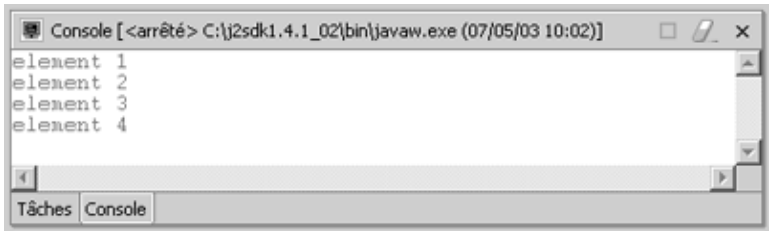
L'onglet "Chemin d'accès aux classes" permet de modifier la liste et l'ordre des bibliothèques utilisées lors de l'exécution. Cet onglet permet de modifier la liste définie dans le projet qui est celle utilisée par défaut.

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.

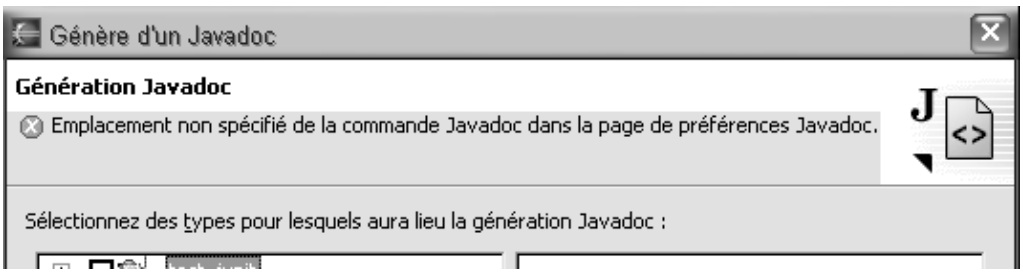


Les messages ayant pour origine une exception sont aussi envoyés dans cette vue.

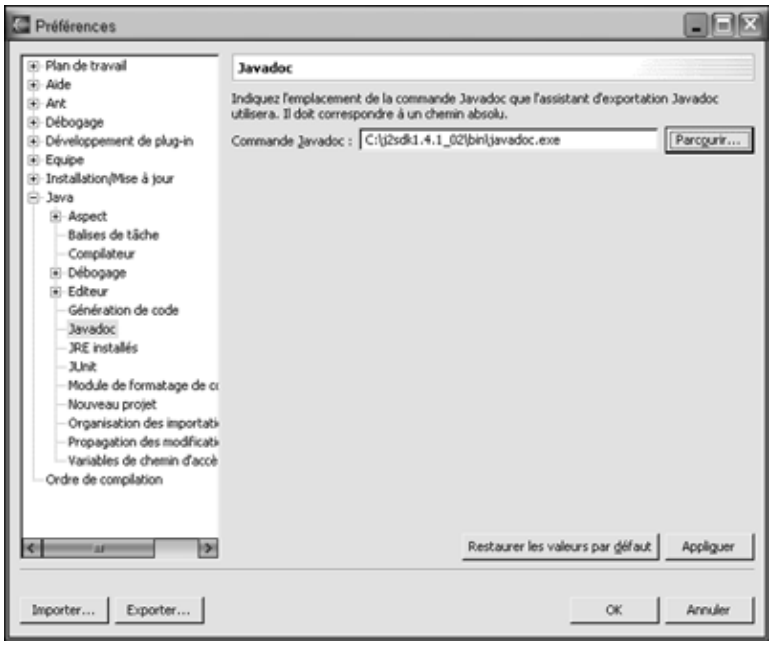
6.6. Génération de la documentation javadoc

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

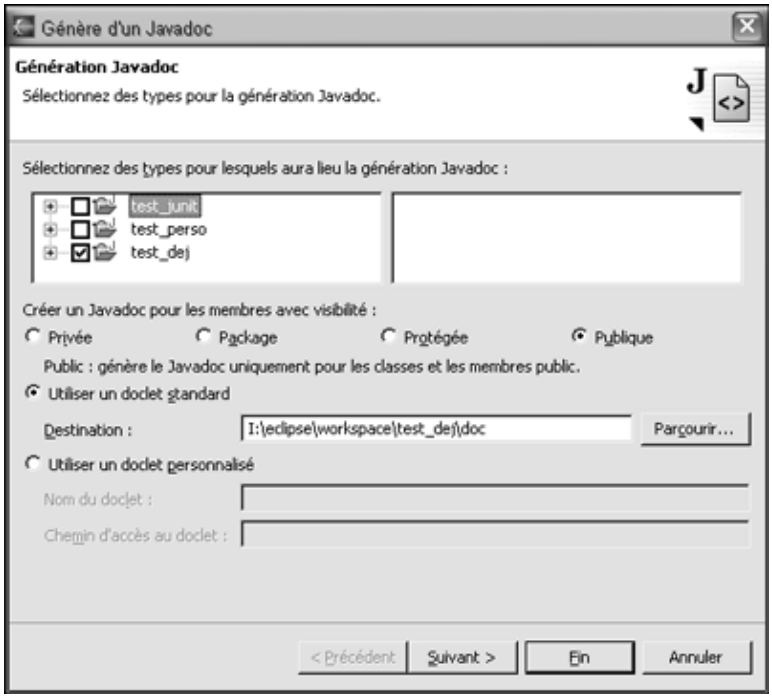


Pour résoudre le problème, il faut aller dans l'arborescence java/javadoc des préférences, cliquer sur le bouton "Parcourir" et sélectionner le fichier javadoc.exe du JDK à utiliser.



Cliquer sur "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.



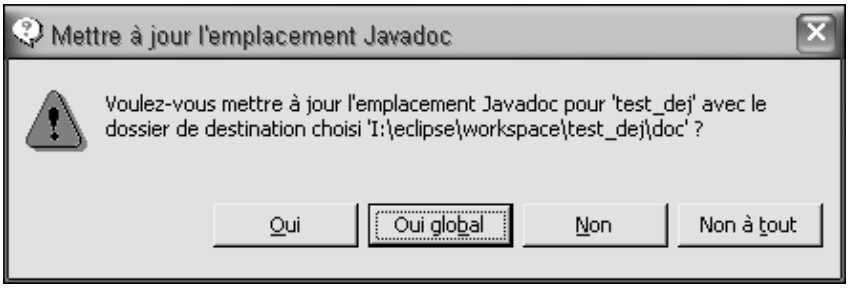
Cliquer sur "Suivant"

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

Une fois les options configurées, cliquer sur "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.

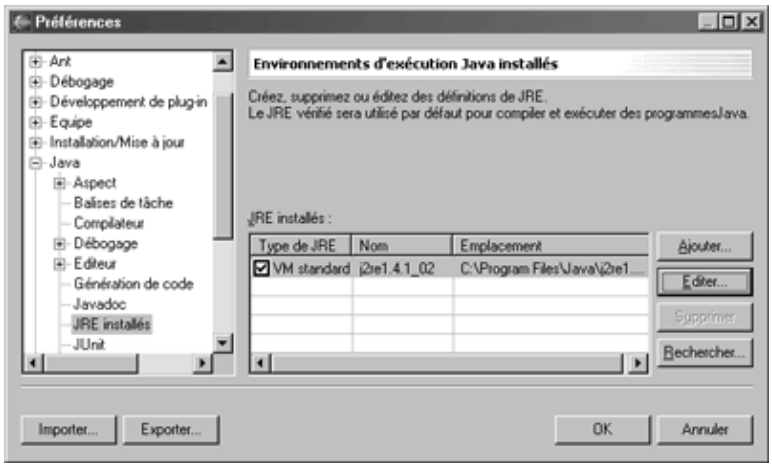


Il est conseillé de répondre "Oui" pour permettre d'avoir accès à cette documentation à partir de l'éditeur en appuyant sur les touches "Shift" + "F2".

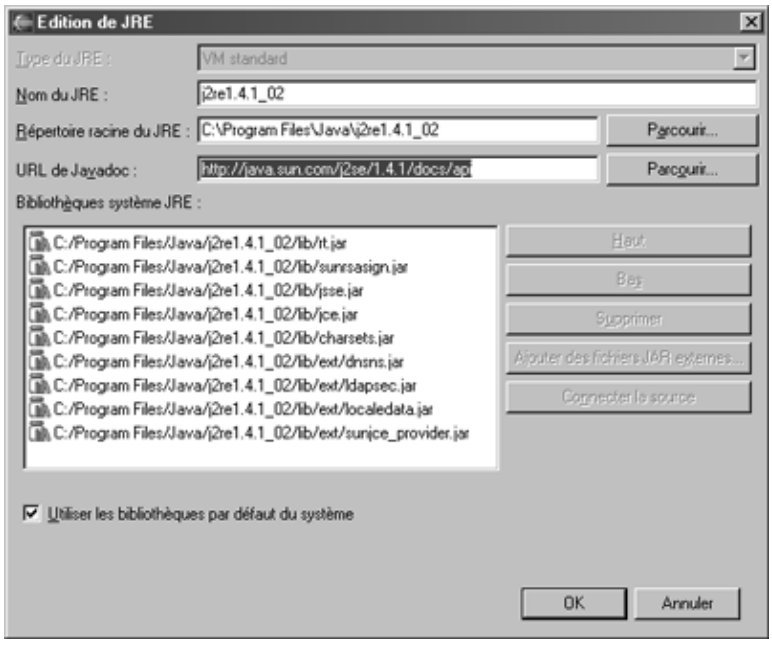
Le détails de la génération est affiché dans le vue "Console".

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.



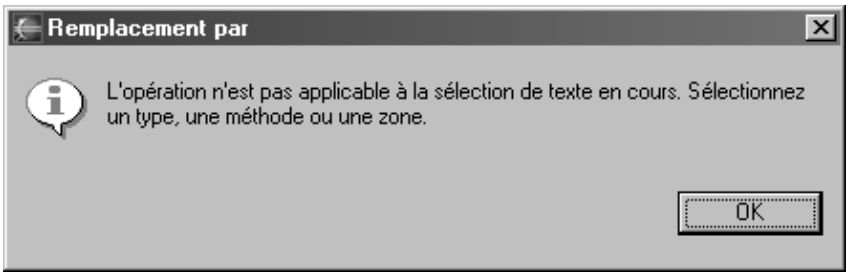
Un clic sur le bouton "Editer" permet de modifier les données du JRE défini dans Eclipse.



6.8. Utilisation de l'historique local

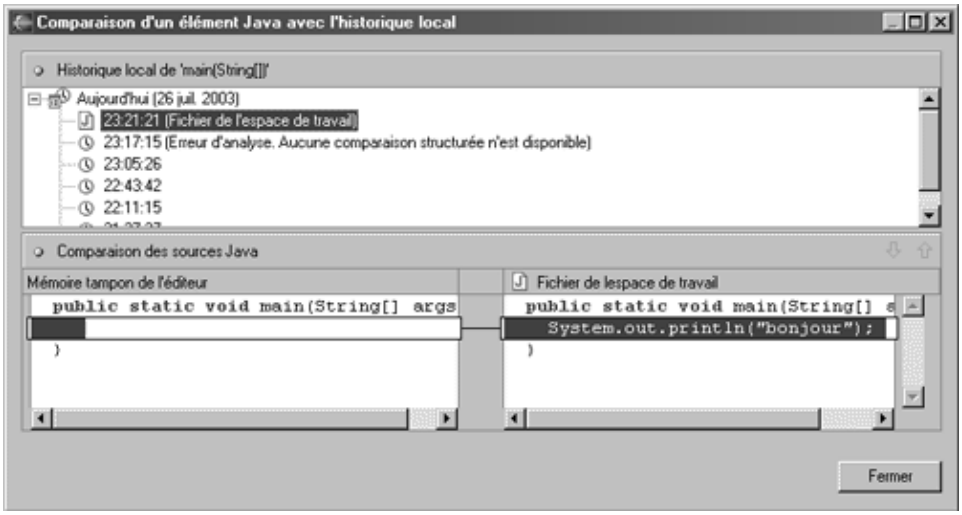
L'historique local est une fonctionnalité proposée par Eclipse pour conserver un certain nombre de version du code pour chaque élément contenu dans le workspace.

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



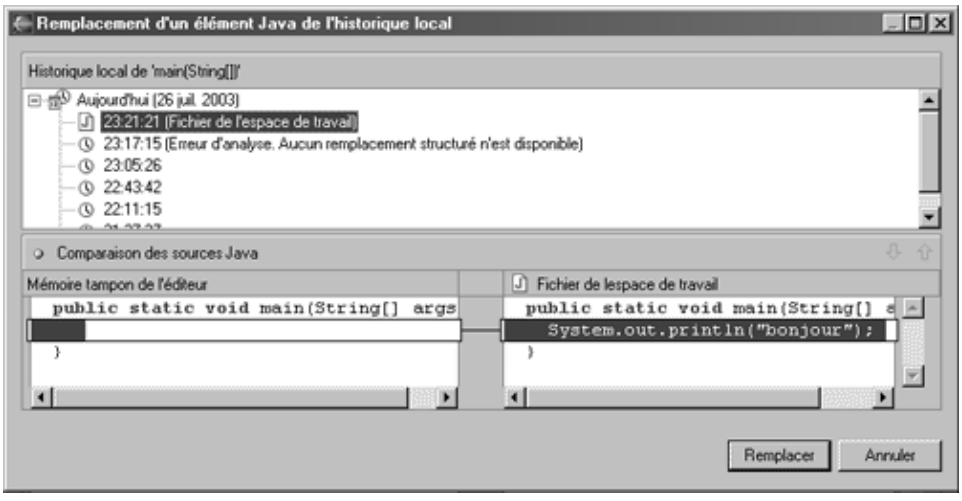
L'option "Historique local" du menu contextuel propose 4 options :

- " 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 " Remplacer ".
- " Restaurer à partir de ... "

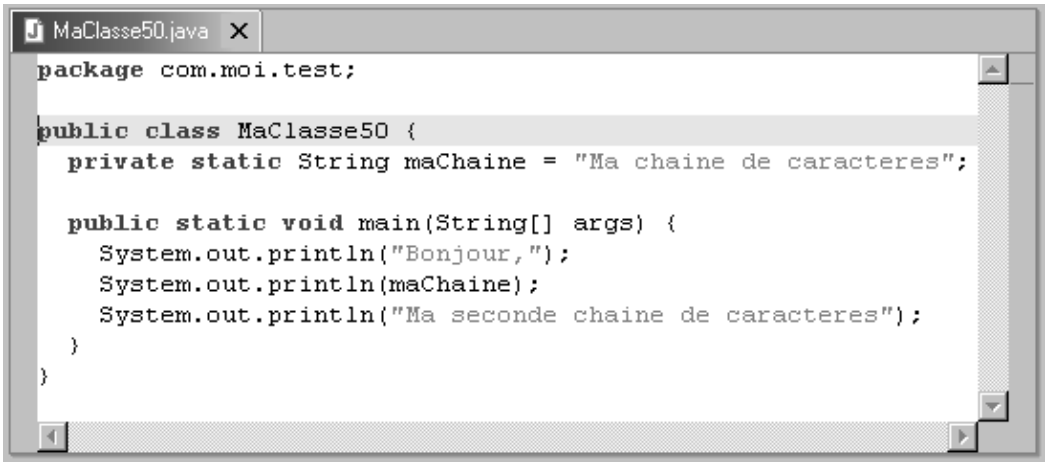
  Cette option permet de restaurer des éléments contenus dans l'historique mais qui ne sont plus présents dans l'éditeur de code.

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

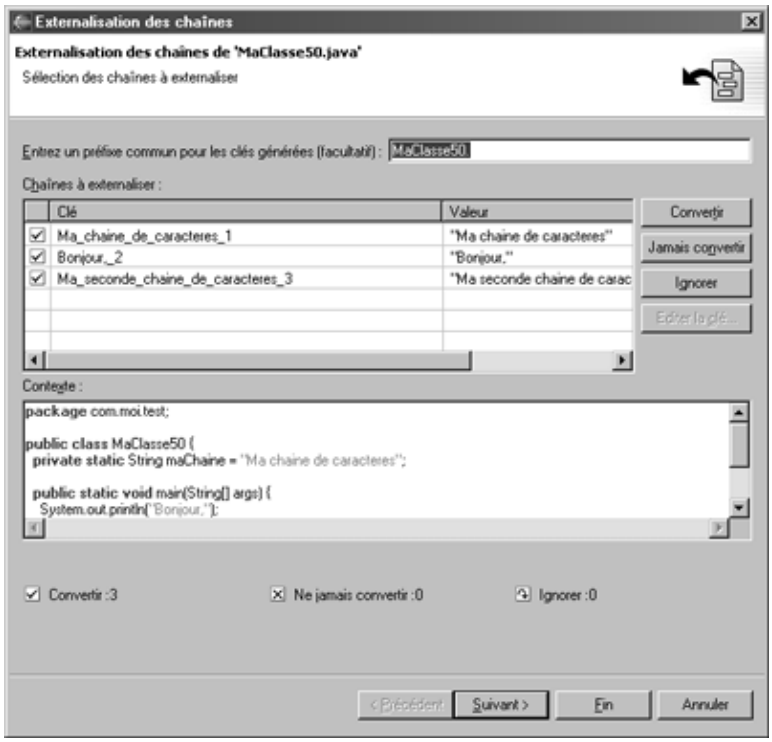
Eclipse possède une fonction permettant d'automatiser l'externalisation des chaînes de caractères dans un fichier de ressource afin de faciliter l'internationalisation de la classe traitée.

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 ".



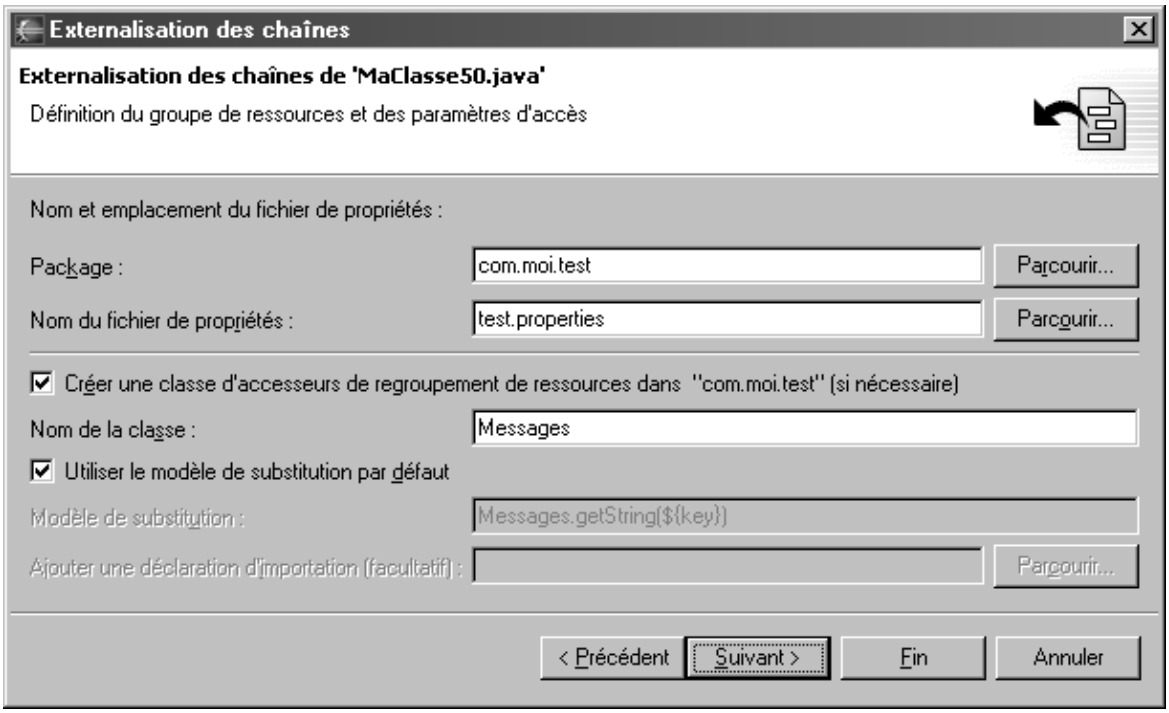
La première page de l'assistant permet de sélectionner les chaînes de caractères à traiter détectées par Eclipse.

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.

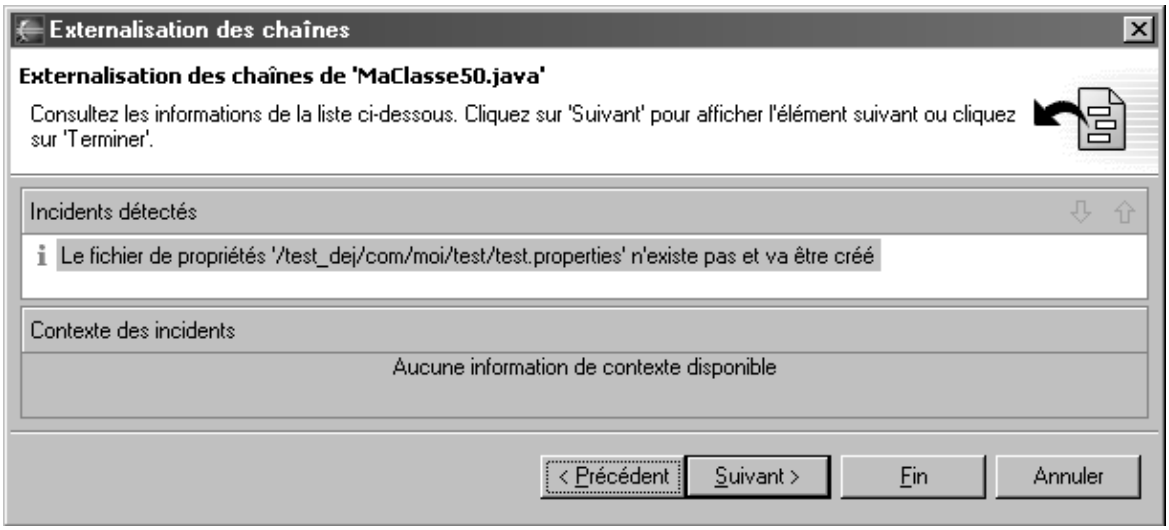
Pour modifier la clé, il est possible de cliquer sur la clé et de saisir le nouveau nom ou de sélectionner la ligne de la chaîne et de cliquer sur le bouton " Editer la clé "

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.

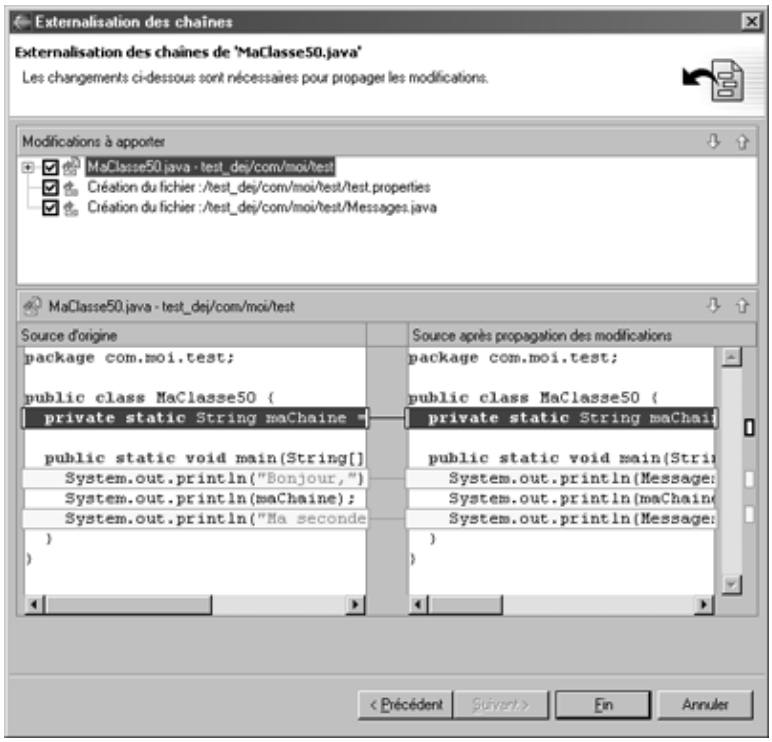
Un clic sur le bouton " Suivant " permet de préciser des informations sur le fichier de ressource qui sera généré.



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



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

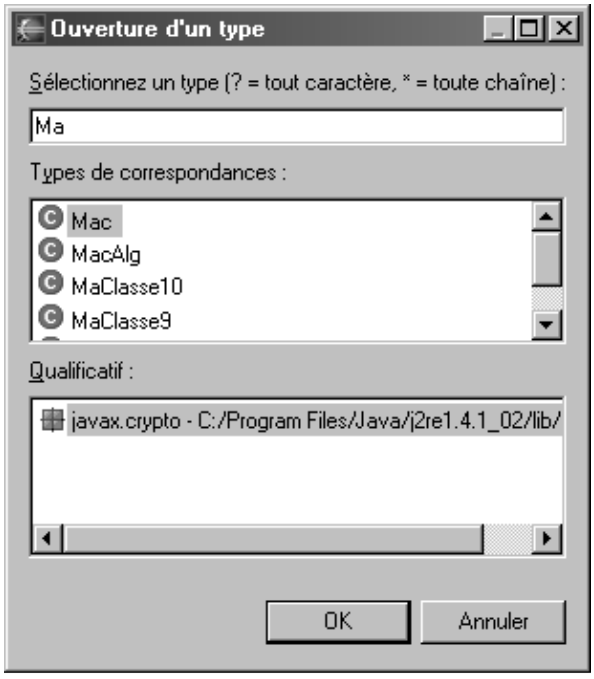


Il est possible de sélectionner tout ou partie des modifications en les cochant.

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'outils " Ouverture d'un type ". Cette outils 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 le workspace ou du nom fichier qui contient la version compilée pour une entité externe.

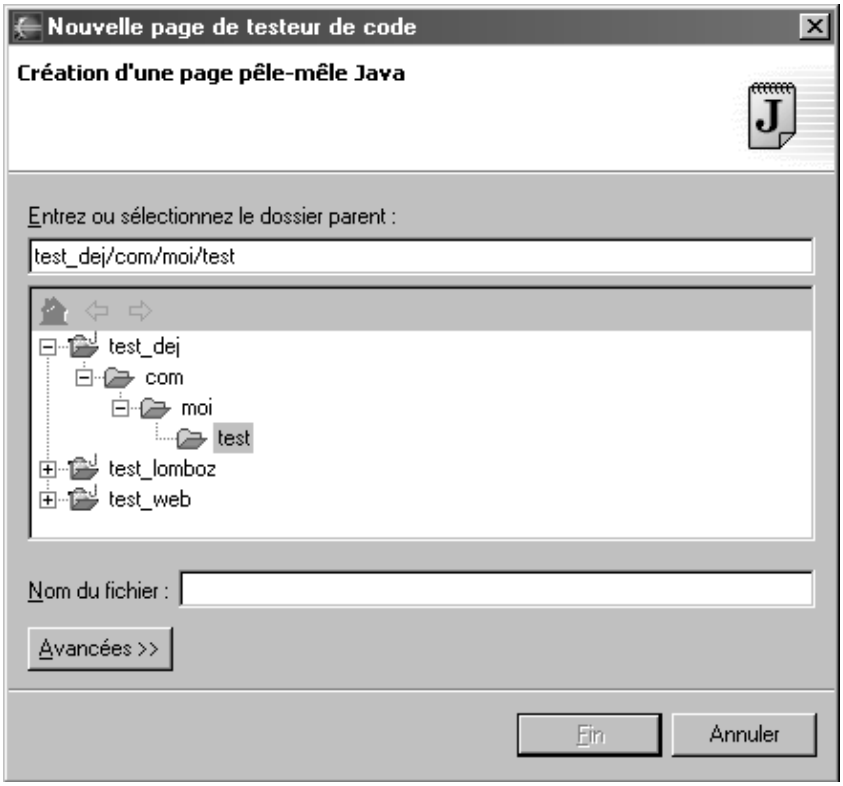
Une fois l'élément voulu sélectionné, un clic sur " OK " ouvre l'éditeur avec cette entité.

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'outils 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.



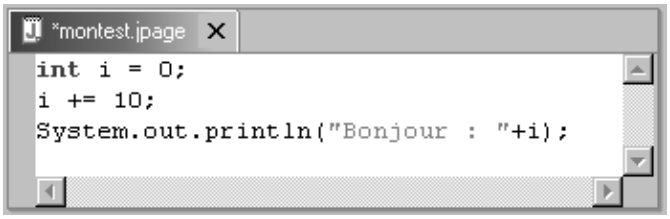
Une "page de testeur de code" est physiquement un fichier contenant du code java et ayant une extension .jpage

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 "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

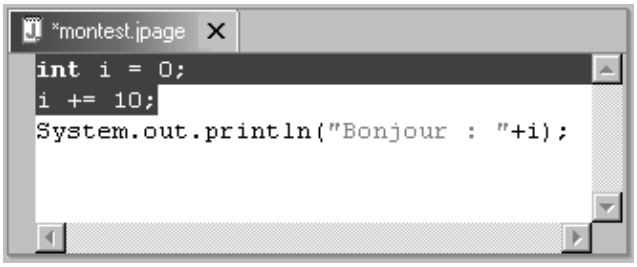
Les trois premiers boutons ne sont utilisables que si un morceau de code est sélectionné dans le scrapbook. Le quatrième bouton n'est utilisable que si une machine virtuelle exécute du code du scrapbook.

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

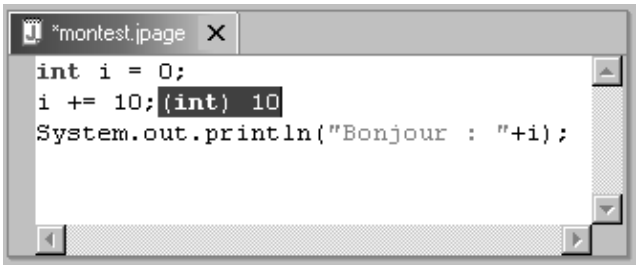
Pour la mettre en oeuvre, il suffit de sélectionner un morceau de code dans l'éditeur du scrapbook et de cliquer sur le bouton  ou d'utiliser l'option " Inspecter " du menu contextuel.

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.



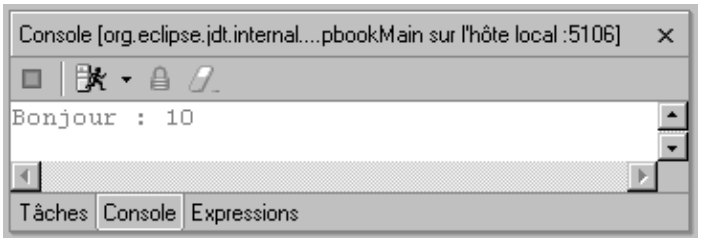
Il faut ensuite cliquer sur le bouton ou d'utiliser l'option " Afficher " du menu contextuel.



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.

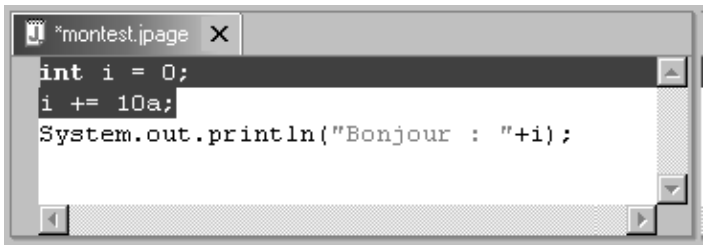
Il faut ensuite cliquer sur le bouton ou utiliser l'option " Exécuter " du menu contextuel.



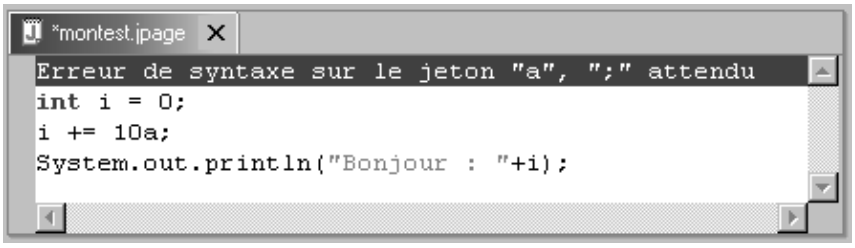
Lors de l'exécution de code dans le scrapbook, une machine virtuelle dédiée est lancée pour cette exécution. Pour pouvoir relancer une exécution, il faut arrêter le machine virtuelle précédemment lancée. L'arrêt de cette machine virtuelle peut se faire en cliquant sur le bouton .

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 :



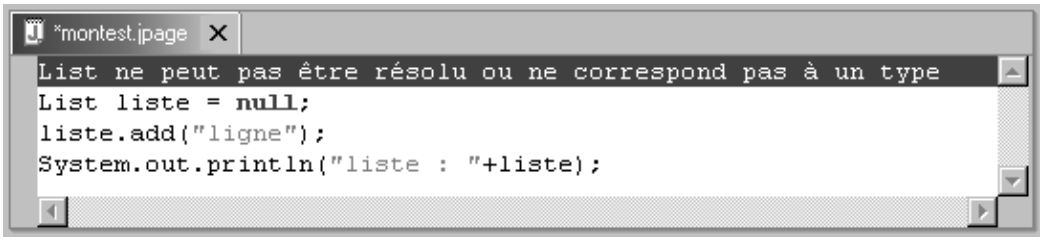
Lors de l'exécution de ce morceau de code, l'erreur suivante est affichée dans l'éditeur



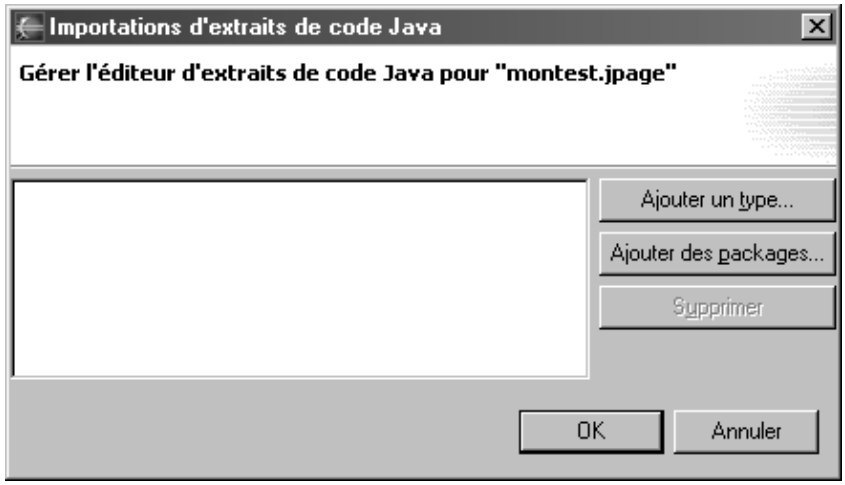
La structure d'un fichier java classique n'étant pas respecter dans le scrapbook, la gestion des clauses d'import est gérée de façon particulière.

Exemple :

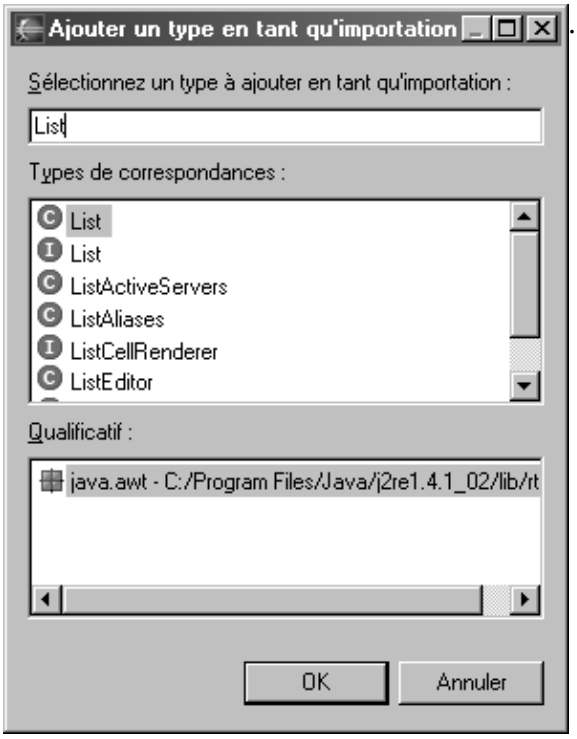
L'exécution de ce morceau de code génère l'erreur suivante :



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.



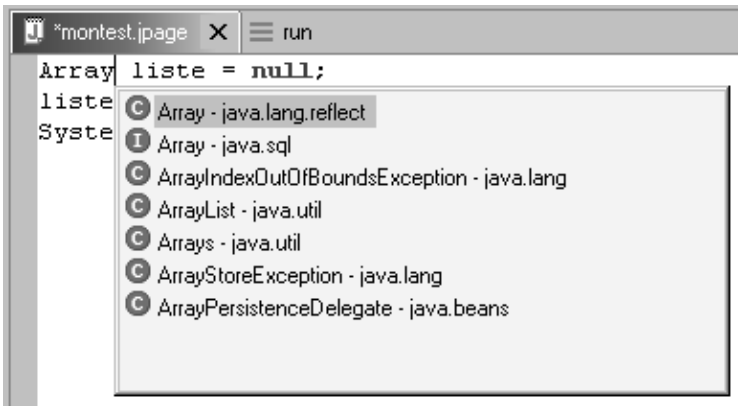
La zone de saisie du type permet une recherche incrémentale dans toutes les entités définies dans le chemin de compilation du projet.

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".

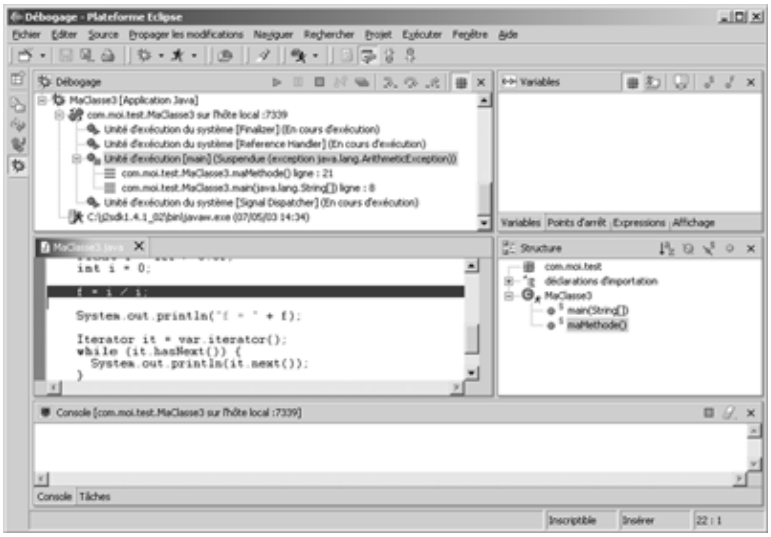

*montest.jpage
run
Array liste = null;
liste
Syste
Array - java.lang.reflect
Array - java.sql
ArrayIndexOutOfBoundsException - java.lang
ArrayList - java.util
Arrays - java.util
ArrayStoreException - java.lang
ArrayPersistenceDelegate - java.beans

7. Deboguer du code Java

7.1. La perspective "debug"

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

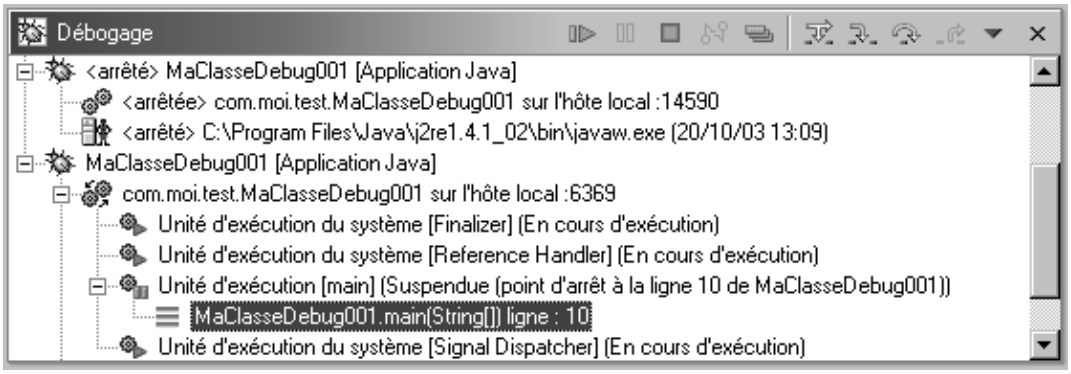
Les vues spécifiques au débogage sont :

- 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.

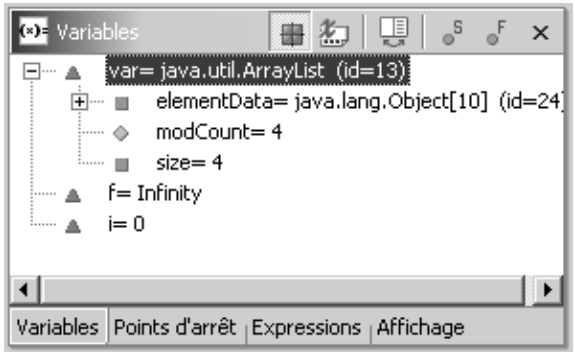
Icône	Rôle
	Reprendre l'exécution précédemment interrompue
	Interrompre l'exécution du processus
	Permet de mettre fin au processus
	Enlève de la liste tous les processus qui sont terminés
	
	Execute la ligne courante et arrête l'exécution sur la première ligne de code incluse dans la première méthode de la ligne courante (raccourci : F5)
	Excécute 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

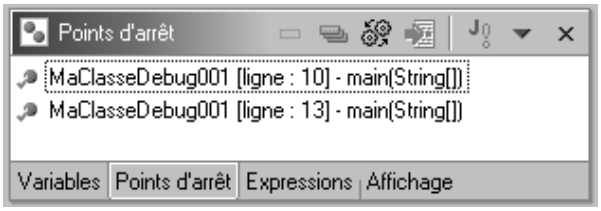
7.2.2. La vue "Variables"

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 formées une arborescense contenant chacun des champs si la variable est un objet.



7.2.3. La vue "Points d'arrêts"

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

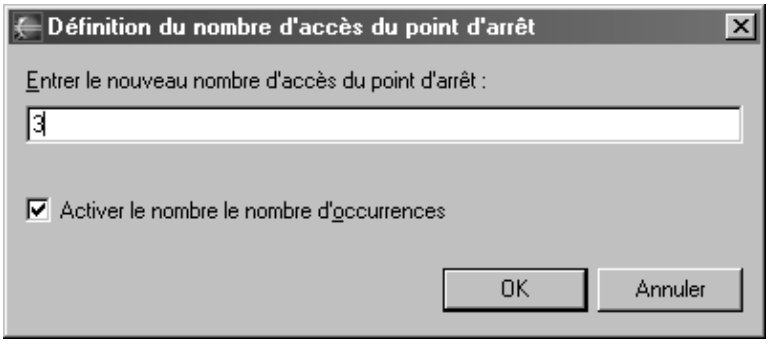


Un double clic sur un des points d'arrêts permet d'ouvrir l'éditeur de code directement sur la ligne ou le point d'arrêt est défini. Cette action est identique à l'utilisation de l'option "Accéder au fichier" du menu contextuel ou à un clic sur le bouton  de la barre de titre.

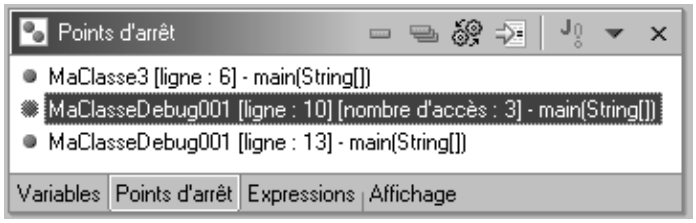
Il est possible grâce au menu contextuel de contrôler les points d'arrêts.

L'option "Nombres d'occurences" permet de préciser le nombre fois ou le points d'arrêts exécutés sera ignorés avant qu'il n'intérrompe l'exécution. Ceci est particulièrement pratique lorsqu'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".

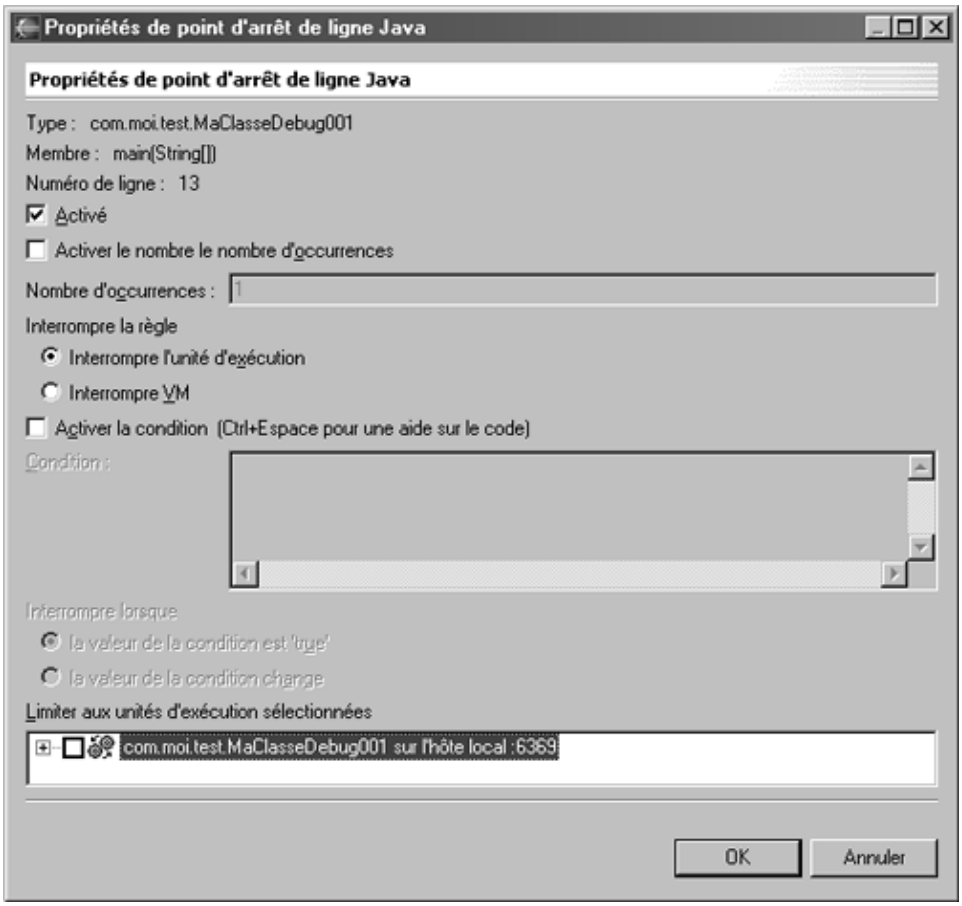


Il est possible d'activer ou de désactiver le point d'arrêt sélectionné respectivement grace à l'option "Activer" ou "Désactiver".

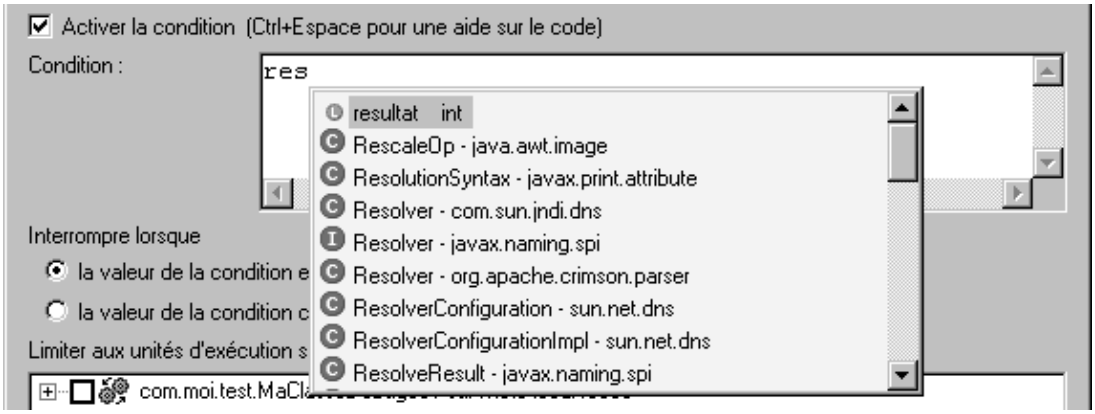
L'option "Supprimer" permet de supprimer le point d'arrêt sélectionné. Il est aussi possible d'utiliser le bouton de la barre de titre de la vue.

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.

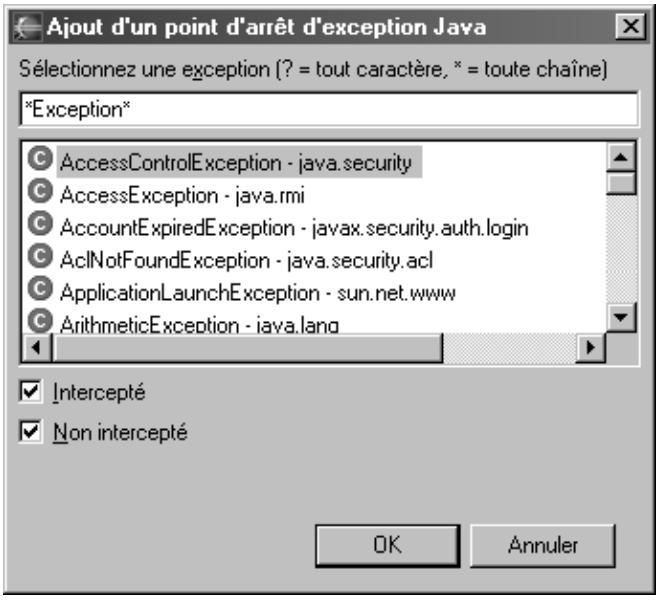


Un de ces paramètres les plus intéressants est la possibilité de mettre une condition d'activation du point d'arrêt. Il suffit pour cela de cocher la case "Activer la condition" et de la saisir dans la zone de texte prévu à cet effet. Dans cette zone de texte, l'assistant de complétion de code est utilisable.



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".

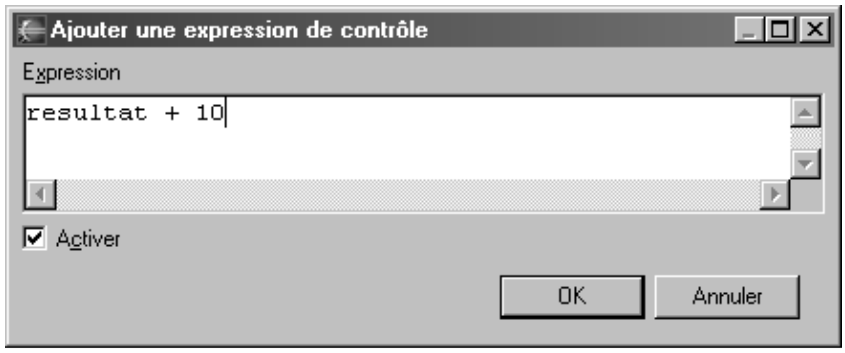
Il est possible de préciser si l'arrêt se fait sur une exception interceptée, non intercepté ou dans les deux cas. Ce type de point d'arrêt possède les mêmes propriétés que les points d'arrêts liés à une ligne de code.

7.2.4. La vue "Expressions"

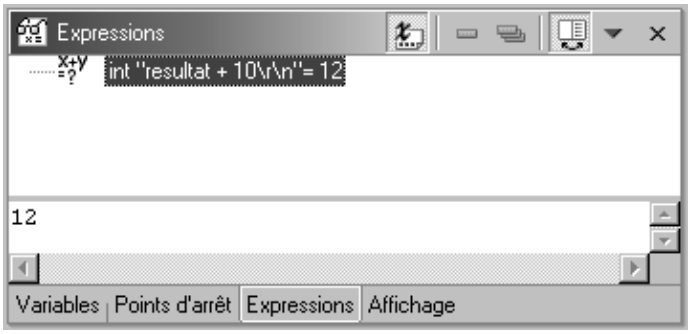
La vue "Expressions" permet d'inspecter la valeur d'une expression selon les valeurs des variables dans les traitements en cours de debogage.

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.

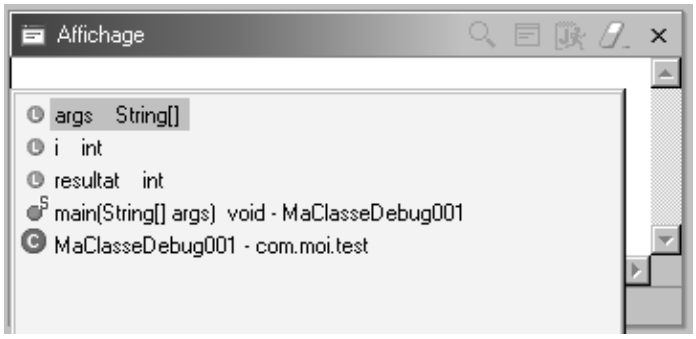
L'option "Editer l'expression de contrôle" permet de modifier l'expression.

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 :

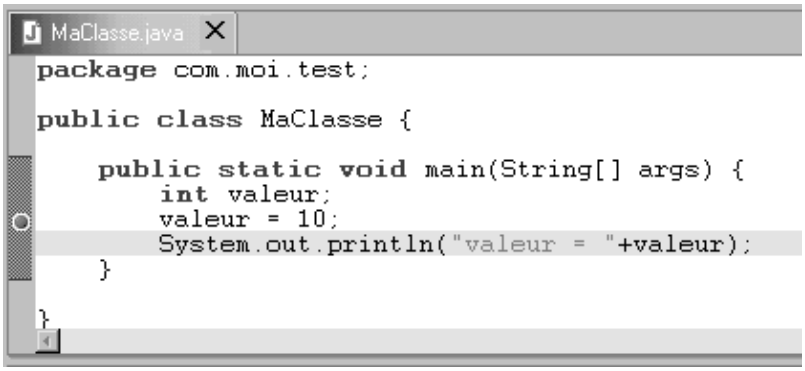
- inspecter : exécution et évaluation dans la vue "Expressions"
- afficher : afficher le résultat dans la vue vue "Affichage"
- exécuter : executer l'expression sélectionnée. Il est ainsi possible de modifier la valeur d'une variable

7.3. Mise en oeuvre du débogueur

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

7.3.1. Mettre en place un point d'arêt

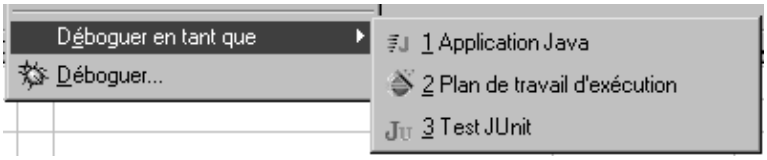
Pour placer un point d'arrêt, il suffit dans l'éditeur de double cliquer dans la barre de gauche pour faire apparaître une icône ronde bleue.



7.3.2. Exécution dans le débogueur

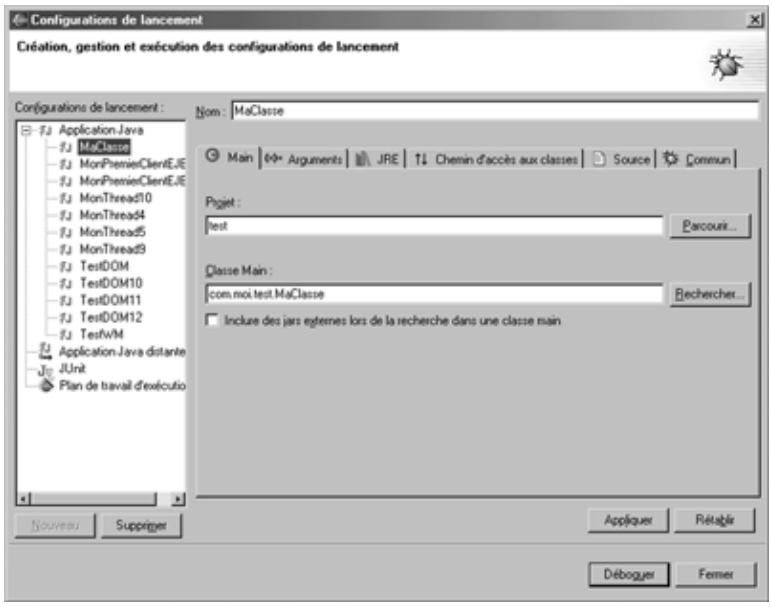
Pour lancer l'exécution dans le débogueur, il faut ouvrir le menu en cliquant sur flèche du bouton 

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.



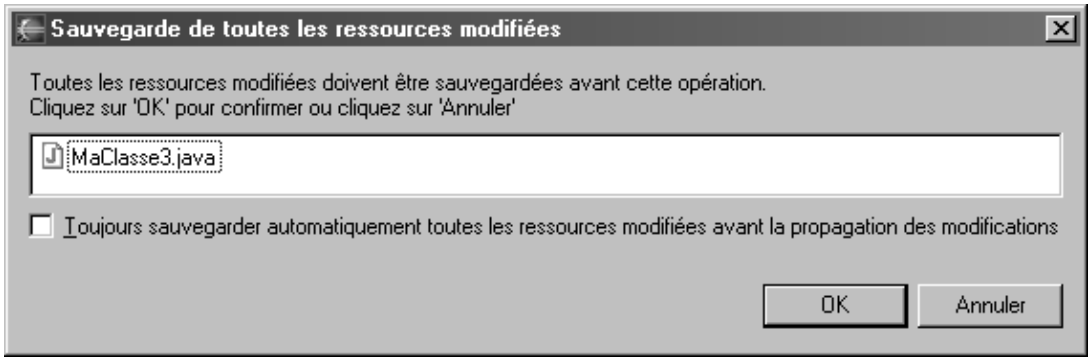
Il ne reste plus qu'à mettre en oeuvre les différentes fonctionnalités proposées par les vues de la perpective pour déboguer le code.

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.

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.

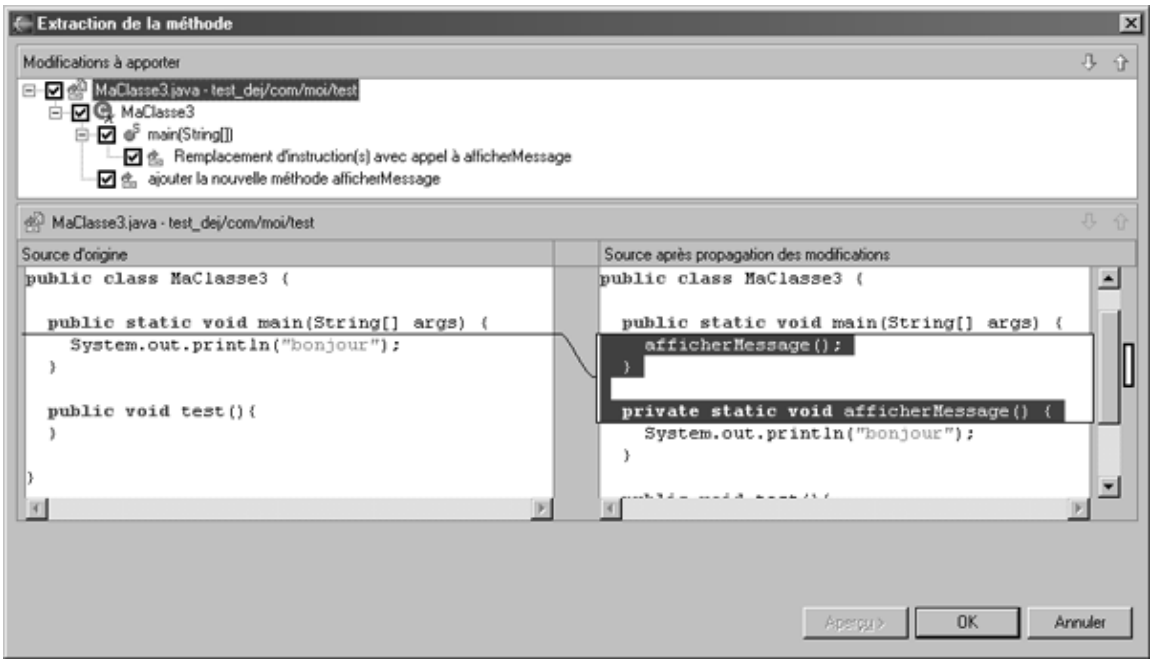
Avant de pouvoir utiliser ces fonctionnalités, il faut sauvegarder l'élément qui sera modifié sinon un message d'erreur est affiché.



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)

Le bouton « Apercu > » permet 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 tout ou partie des modifications pour qu'elles soient appliquées.

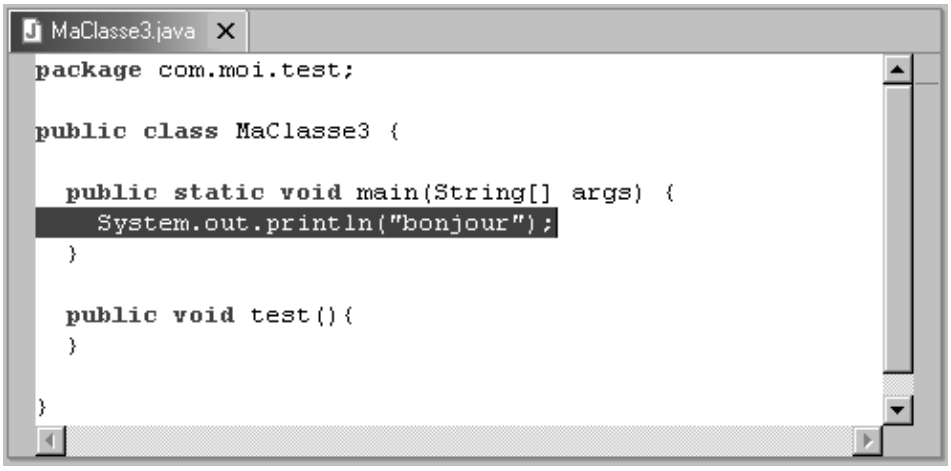
La partie inférieure présente, pour l'élément sélectionné, le code source actuel à gauche et le code source tel qu'il sera après l'application de la fonction à droite.

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

8.1. Extraction d'une méthode

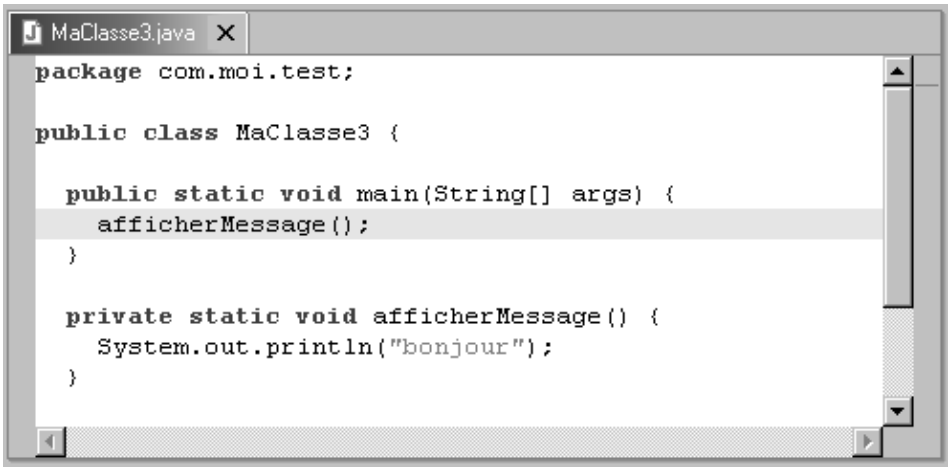
Cette fonction permet de transferer un morceau de code dans une méthode qui sera créée pour l'occasion.

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

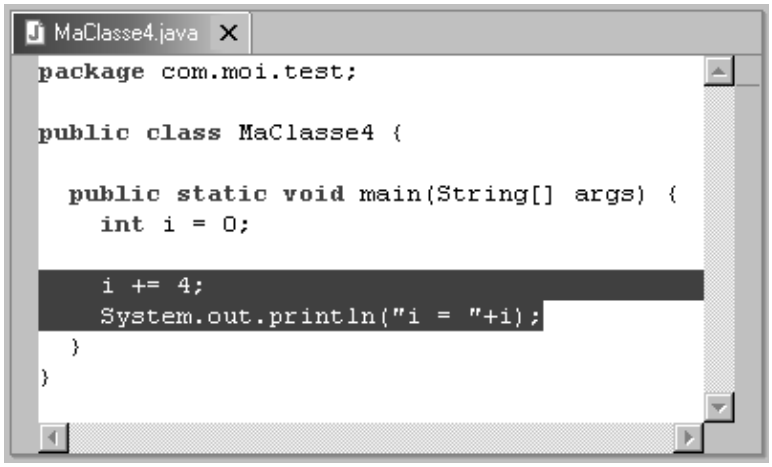


Il faut sélectionner l'option « Extraire la méthode … » du menu principal ou du menu contextuel « Propager les modifications ». Une boite de dialogue permet de saisir le nom de la méthode et de sélectionner le modificateur d'accès.

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

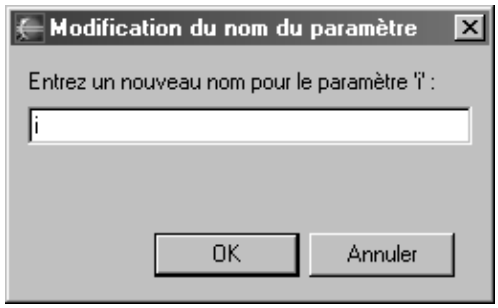


Dans le cas ou le code sélectionné contient une ou plusieurs variables, la boîte de dialogue contient en plus la liste des variables détectées dans le code.

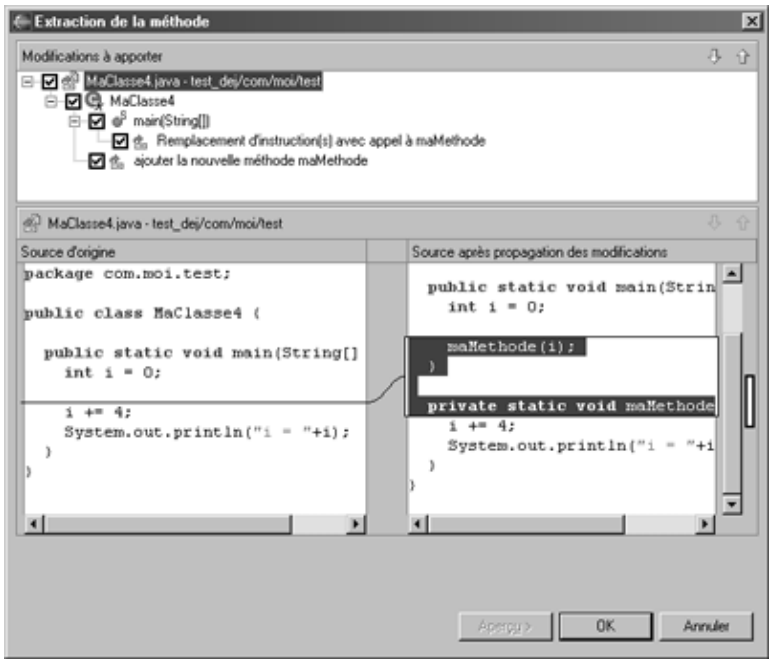


Il faut saisir le nom de la méthode et sélectionner son modificateur d'accès.

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… ».

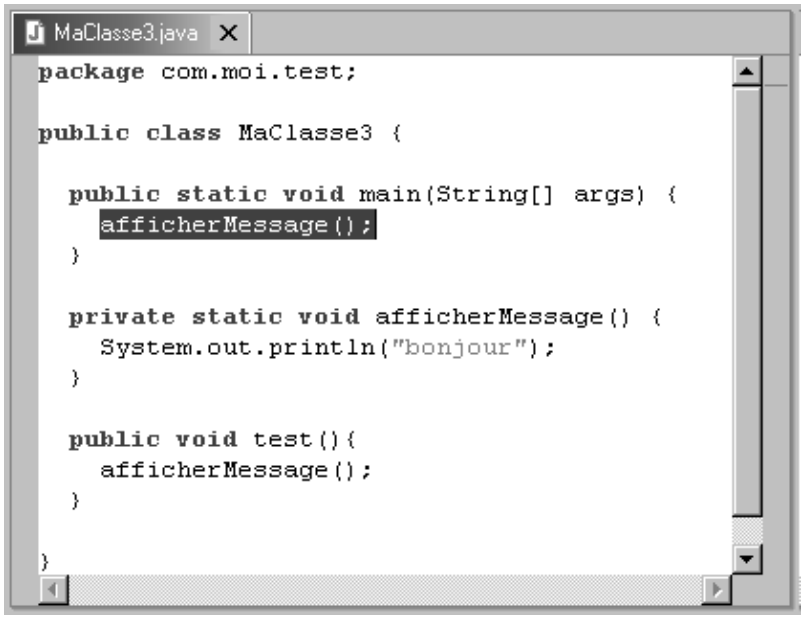


Le bouton « Aperçu » permet de voir et de valider les modifications qui seront apportées.

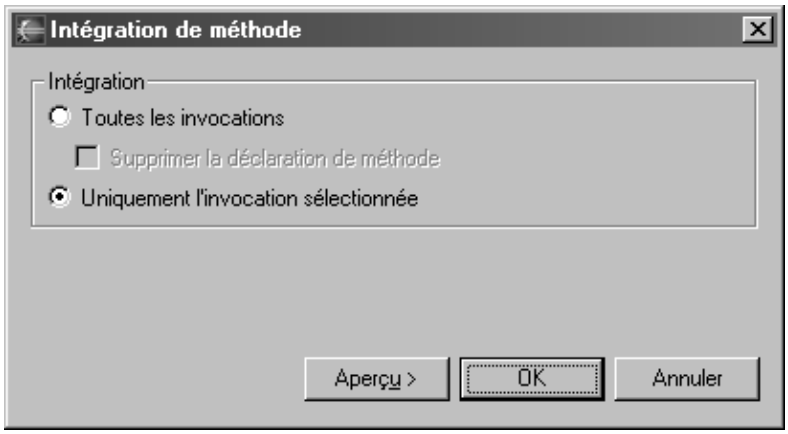


8.2. Intégrer

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

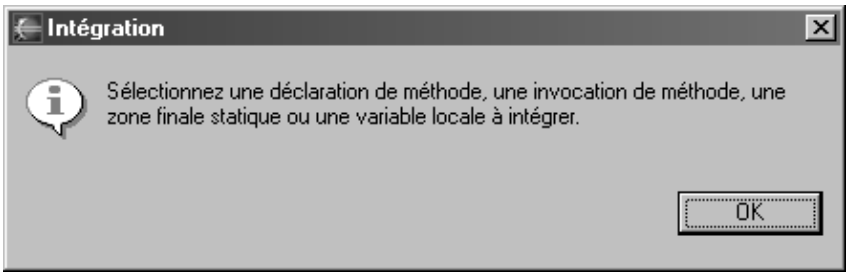


Puis il faut utiliser l'option « Propager les modifications/Intégrer… » du menu principal ou contextuel.



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

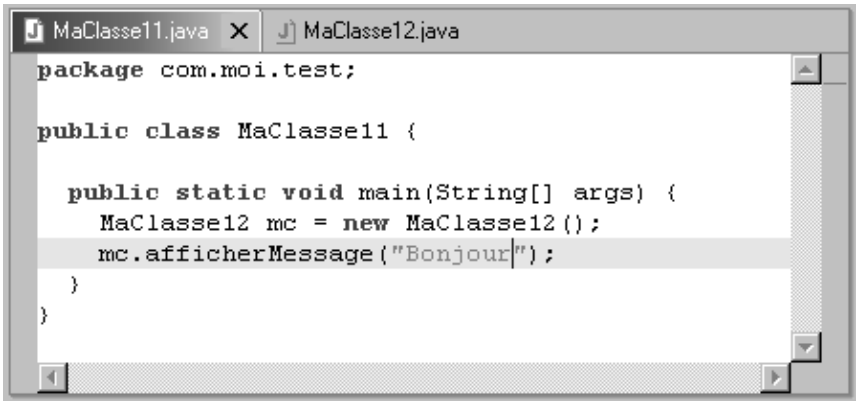
Si le code sélectionné ne correspond pas à une méthode, un message d'erreur est affiché.

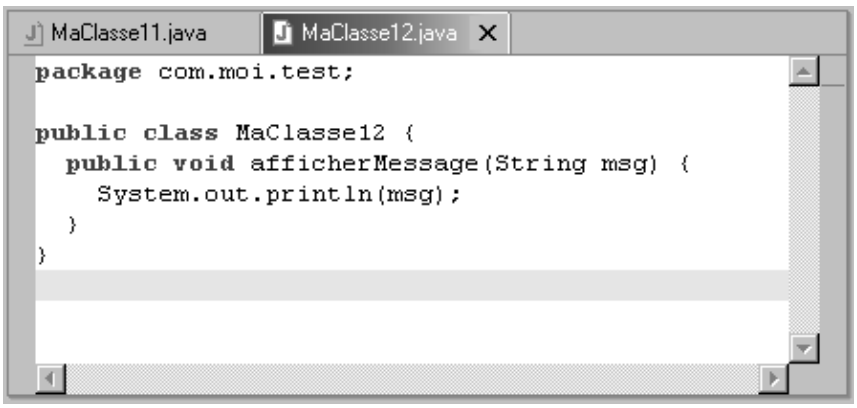


8.3. Renommer

La fonction renommer permet d'attribuer un nouveau nom à une entité présente dans le workspace.

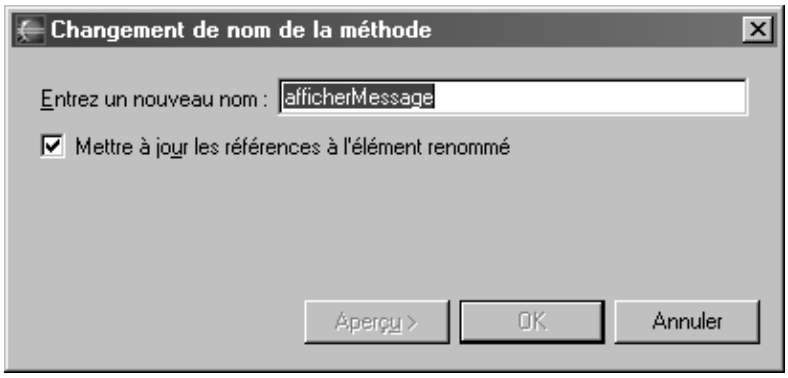
Les exemples de cette section utilisent les deux classes suivantes :



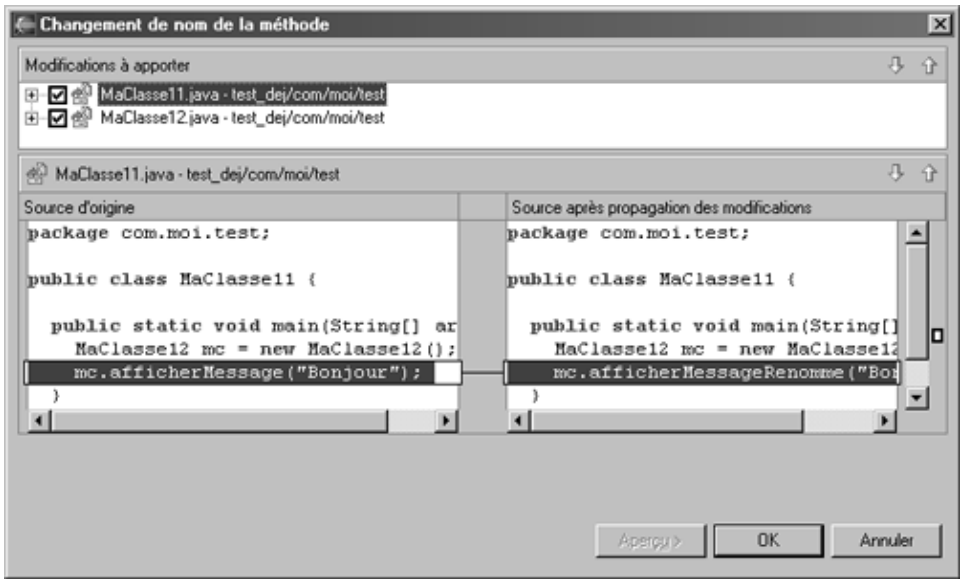


Le renommage concerne l'entité sélectionnée dans une vue ou sous le curseur dans l'éditeur.

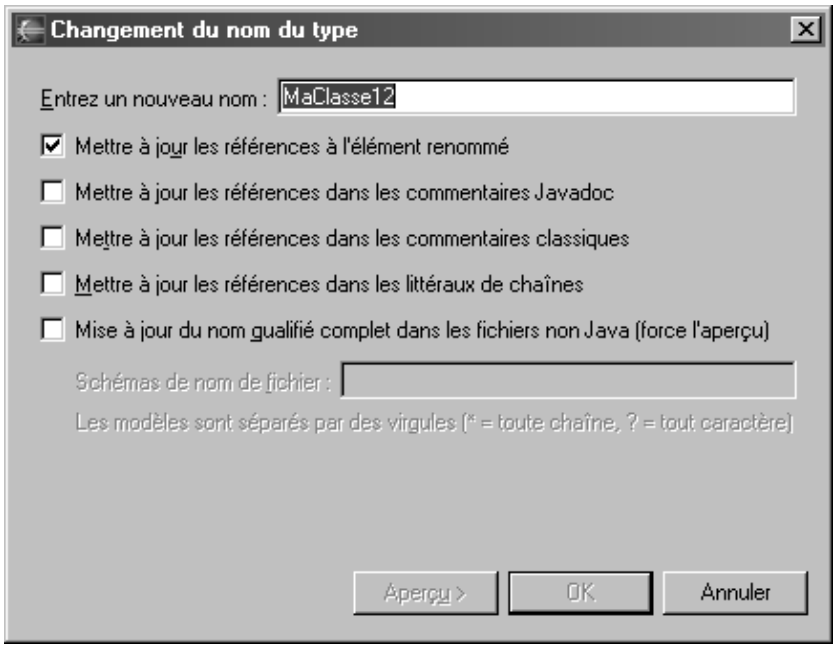
Par exemple, pour renommer une méthode, il suffit de positioner le curseur sur le nom de la déclaration d'une méthode dans le code source et sélectionner l'option « Propager les modifications / Renommer » du menu contextuel.



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 le workspace.

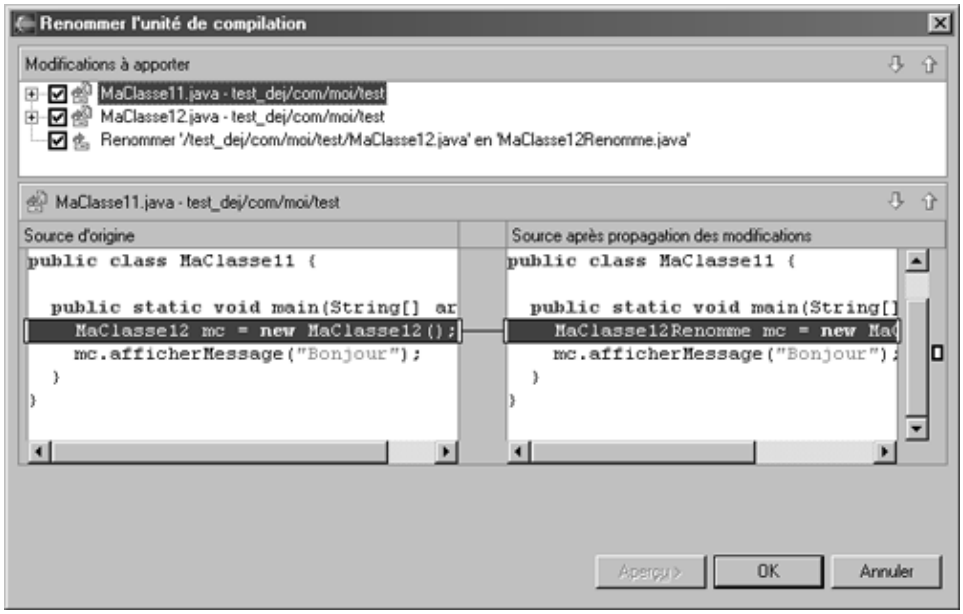


Pour renommer une classe, il y a deux possibilités : selectionner la classe dans la vue « Packages » ou positionner le curseur sur la définition du nom de la classe dans l'éditeur. Puis il faut utiliser l'option « Renommer » du menu contextuel « Propager les modifications ».

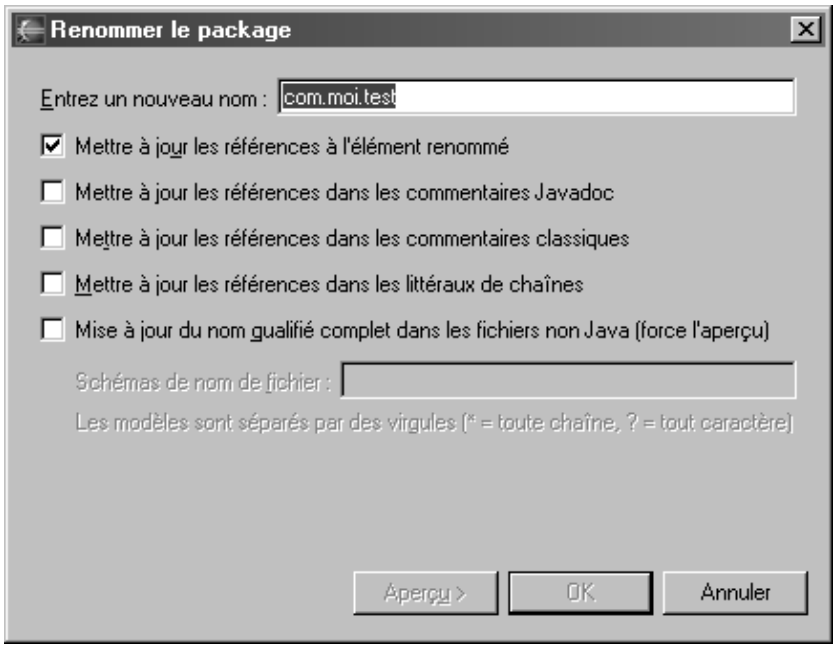


Il est possible de préciser les types d'entités qui doivent être mise à jour.

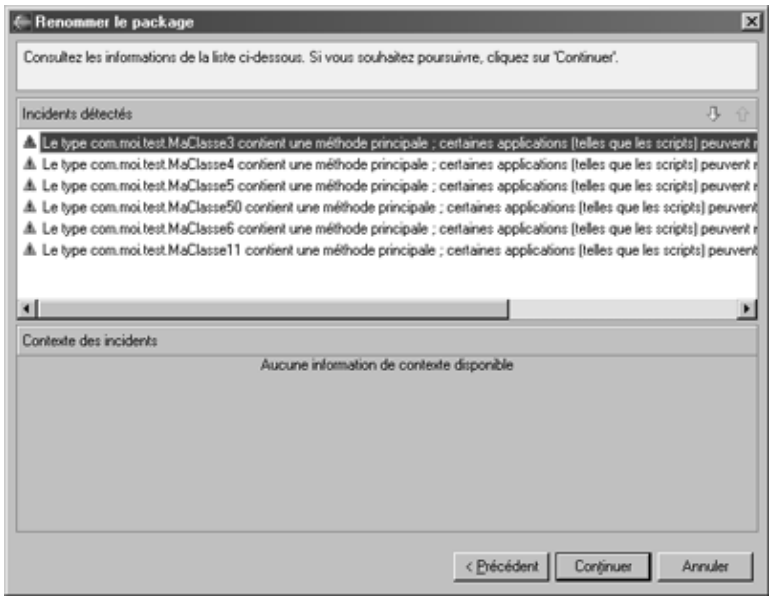
A la validation, Eclipse effectue toutes les modifications nécessaires suite au renommage de la classe, notamment le remplacement à tous les endroits dans le workspace ou la classe est utilisée.



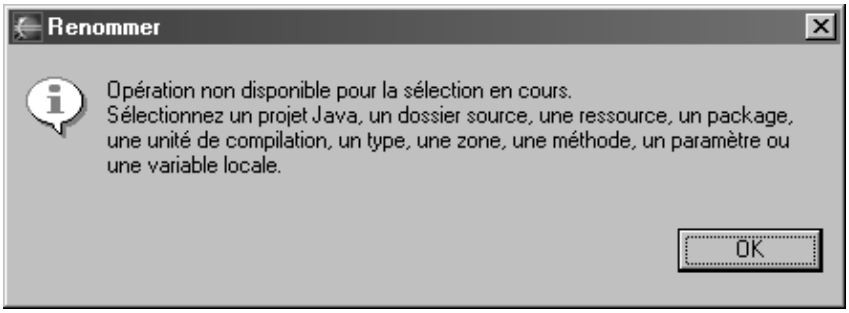
Le principe est le même pour renommer un package.



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.



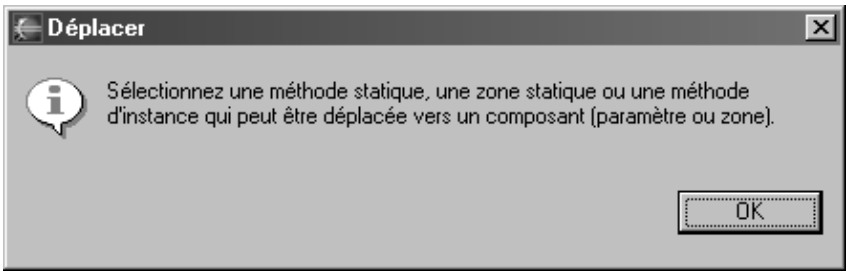
Si la sélection ne correspond à aucun élément renommable, un message d'erreur est affiché.



8.4. Déplacer

Cette section sera développée dans une version future de ce document

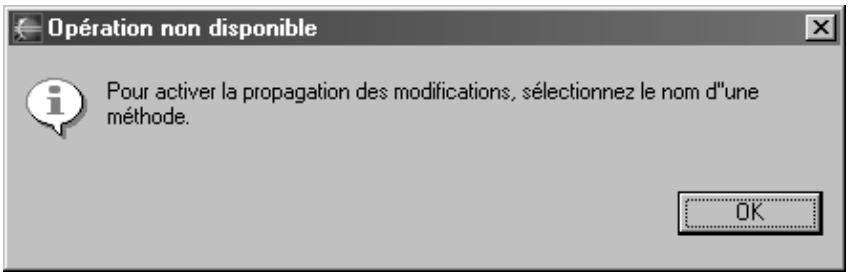
Si le code sélectionné ne correspond pas à une entité concernée par cet action, un message d'erreur est affiché.



8.5. Changer la signature de la méthode

Cette section sera développée dans une version future de ce document

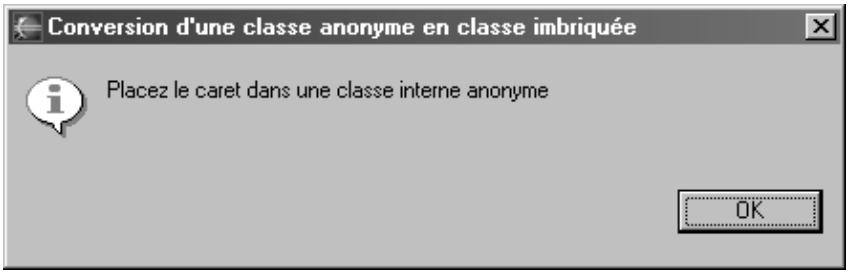
Si le code sélectionné ne correspond pas à une entité concernée par cet action, un message d'erreur est affiché.



8.6. Convertir une classe anonyme en classe imbriquée

Cette section sera développée dans une version future de ce document

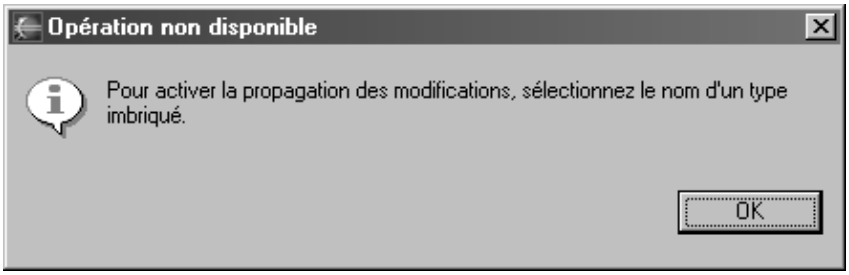
Si le code sélectionné ne correspond pas à une entité concernée par cet action, un message d'erreur est affiché.



8.7. Convertir un type imbriqué au niveau supérieur

Cette section sera développée dans une version future de ce document

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



8.8. Extraire

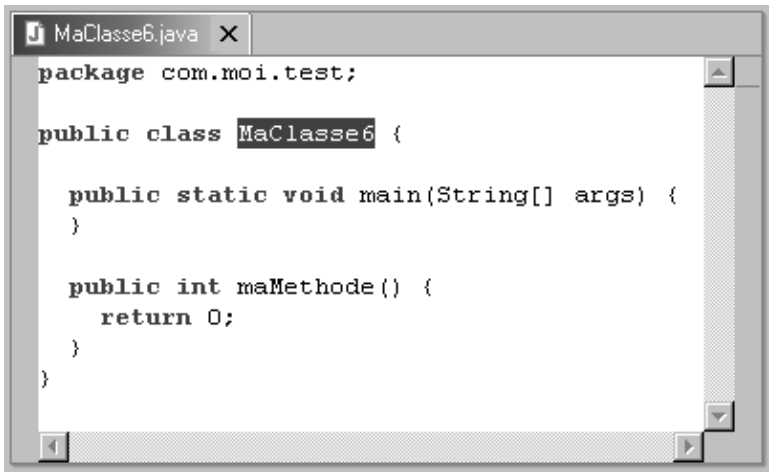
Cette section sera développée dans une version future de ce document

8.9. Transferer

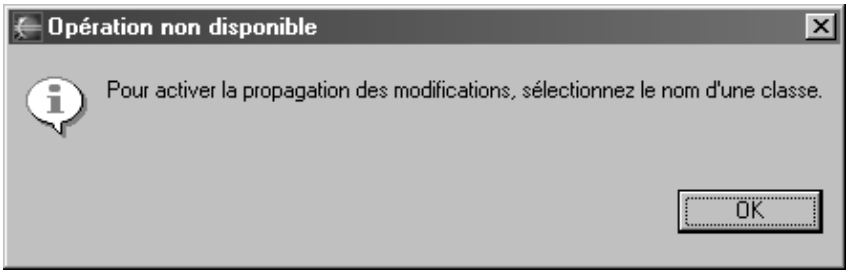
8.10. Extraire une interface

Cette fonction permet de définir à postériori une interface à partir d'une ou plusieurs méthodes définies dans une classe.

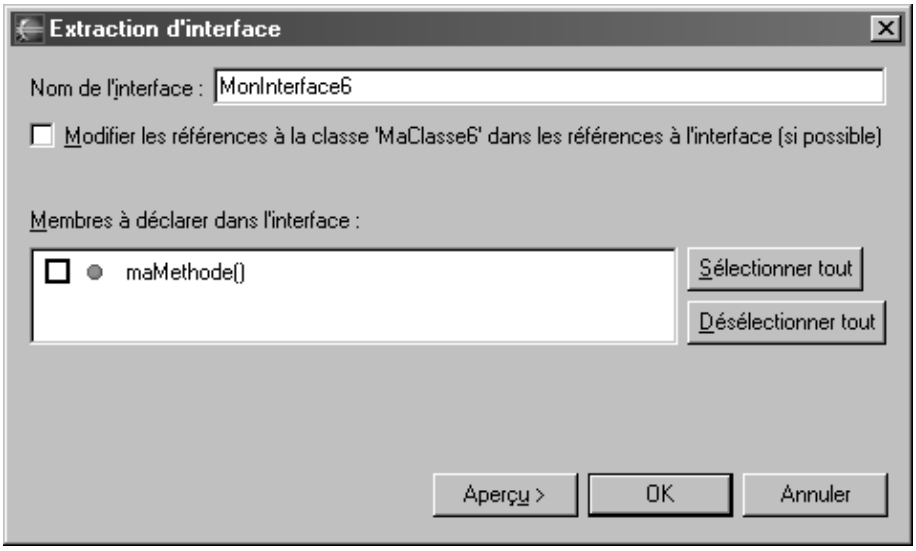
Il faut sélectionner dans le code le nom d'une classe.



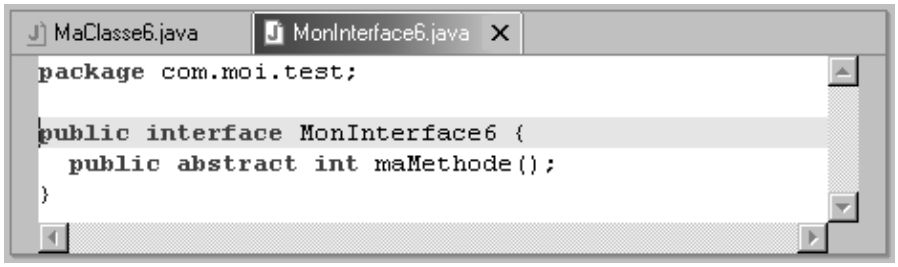
Si la partie sélectionnée ne correspond pas à un nom de classe, un message d'erreur est affiché.

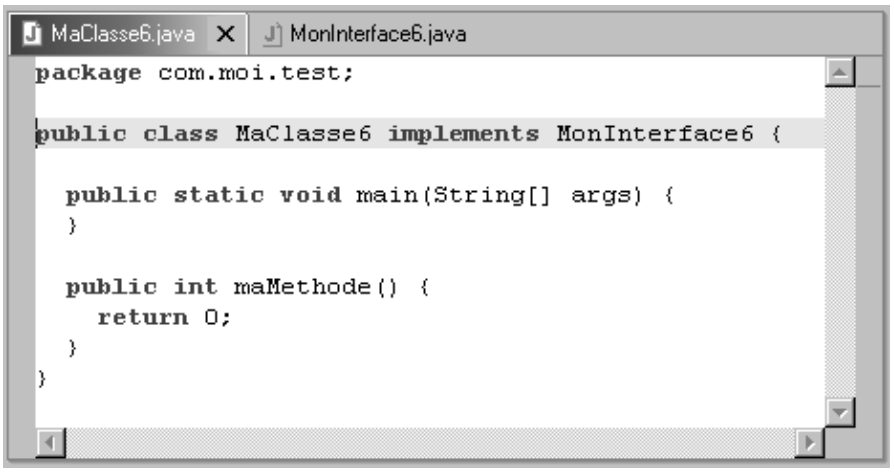


Il suffit ensuite d'utiliser l'option « Extraire une interface » du menu « Propager les modifications ». Une boite de dialogue permet de préciser le nom de l'interface et de sélectionner les membres à déclarer dans l'interface.



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





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.

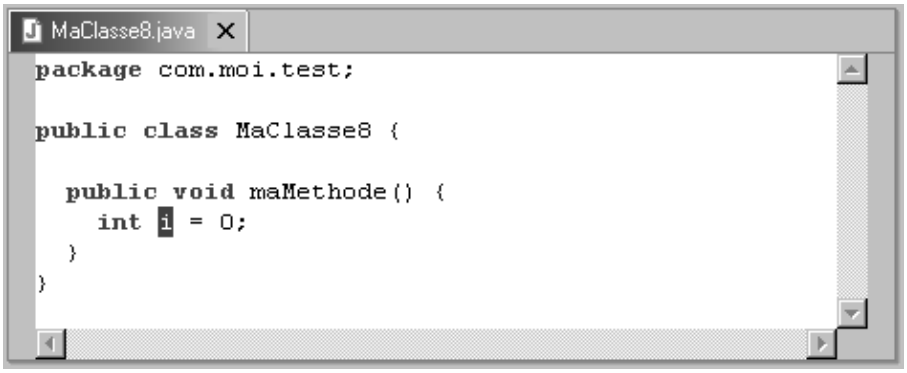
8.11. Utiliser le supertype si possible

Cette section sera développée dans une version future de ce document

8.12. Convertir la variable locale en zone

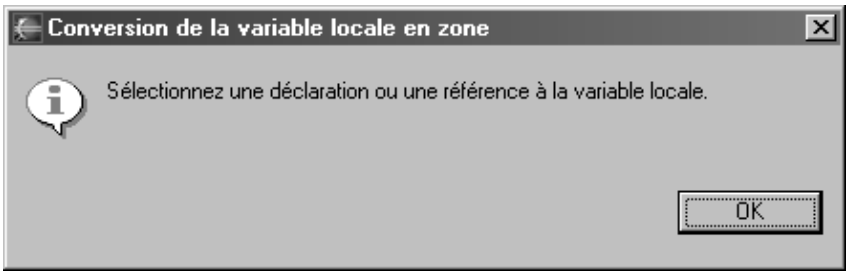
La fonction « convertir la variable locale en zone » permet de transformer une variable locale à une méthode en un champ de la classe.

Pour utiliser cette fonction, il faut sélectionner dans le code une variable locale.

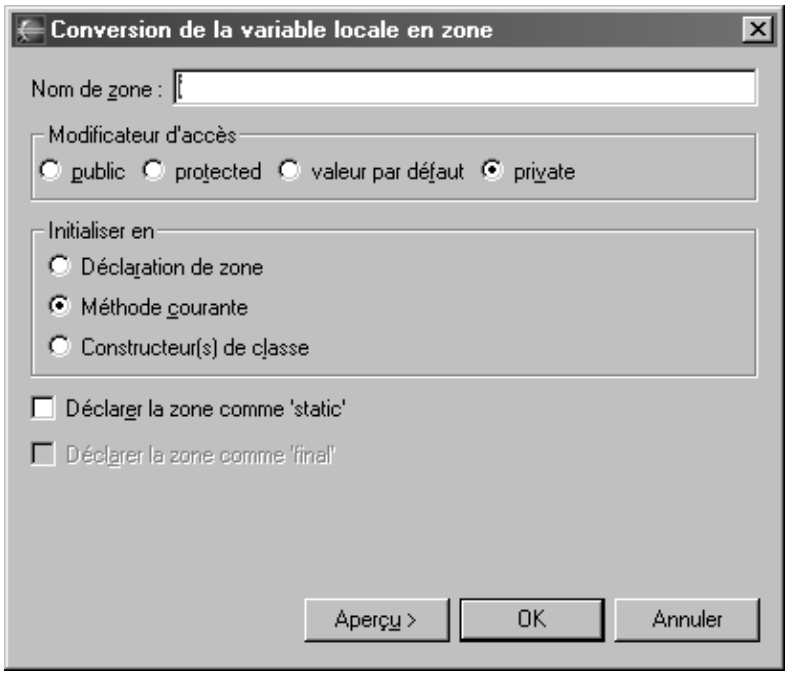


Puis utiliser l'option « Convertir la variable locale en zone » du menu « Propager les modifications ».

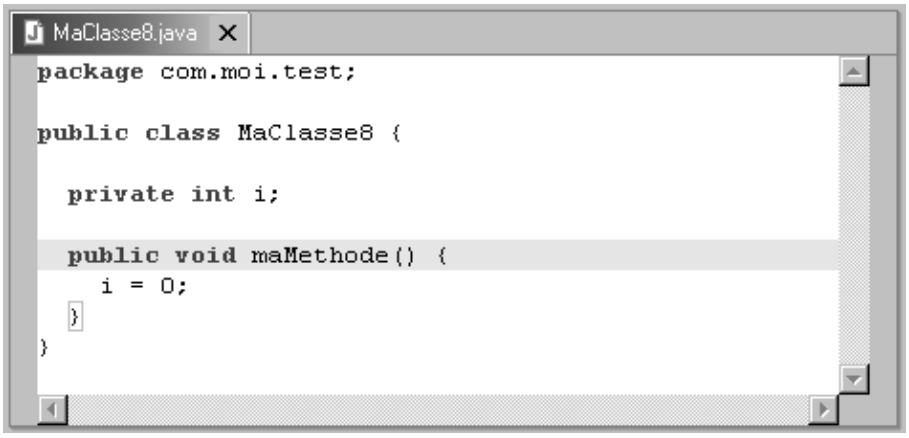
Si la sélection ne correspond pas à une variable locale, un message d'erreur est affiché.



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



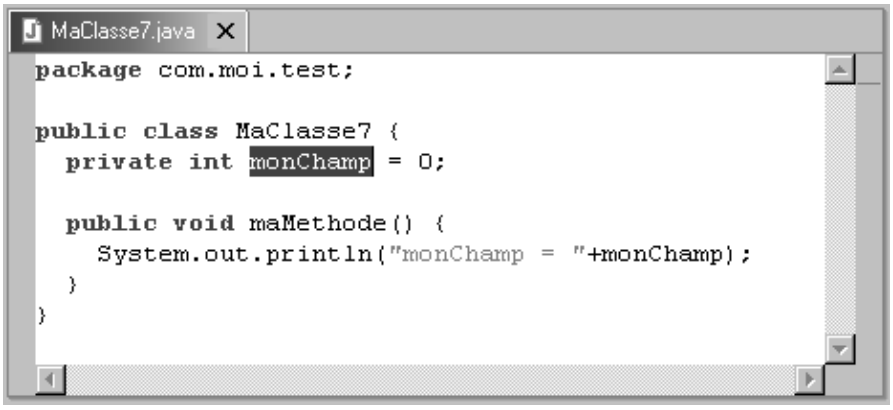
Le résultat de l'exécution de la fonction est le suivant :



8.13. Encapsuler la zone

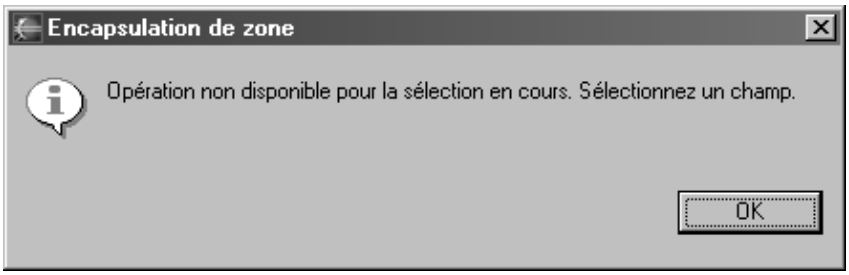
Cette fonction permet d'encapsuler un attribut en générer un getter et éventuellement un setter.

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



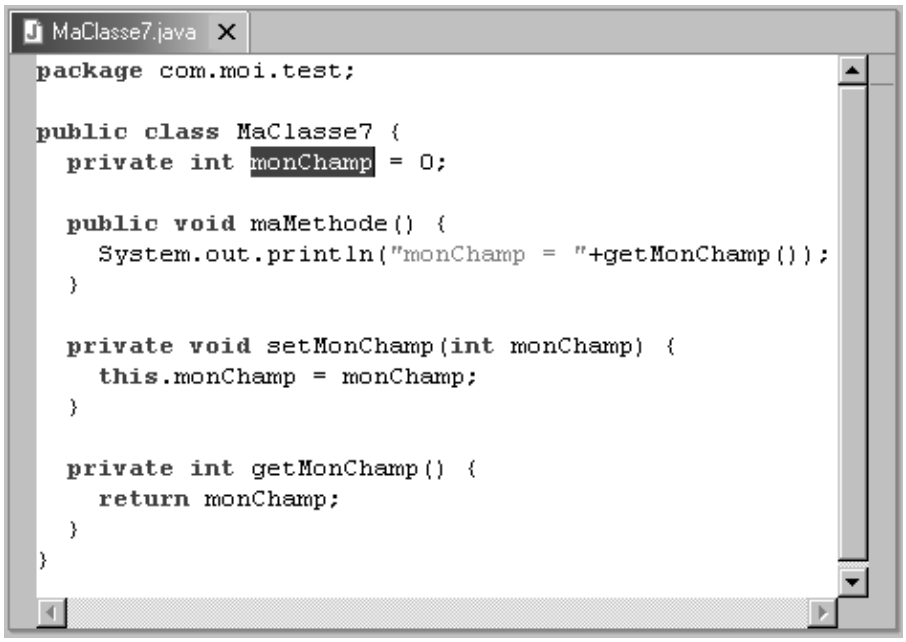
Il faut ensuite utiliser l'option « Encapsuler la zone » du menu « Propager les modifications ».

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.

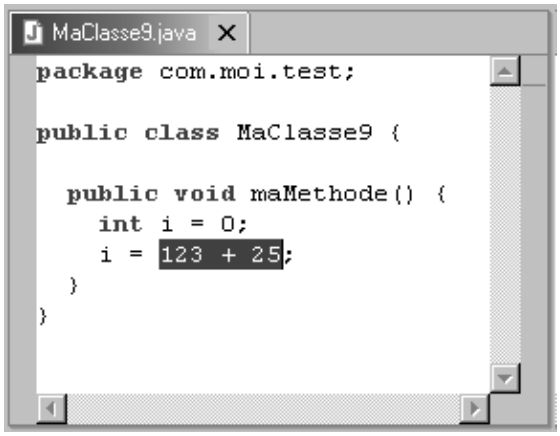


En sélectionnant l'option « conserver les références de zones », ces méthodes ne sont pas utilisées dans la classe.

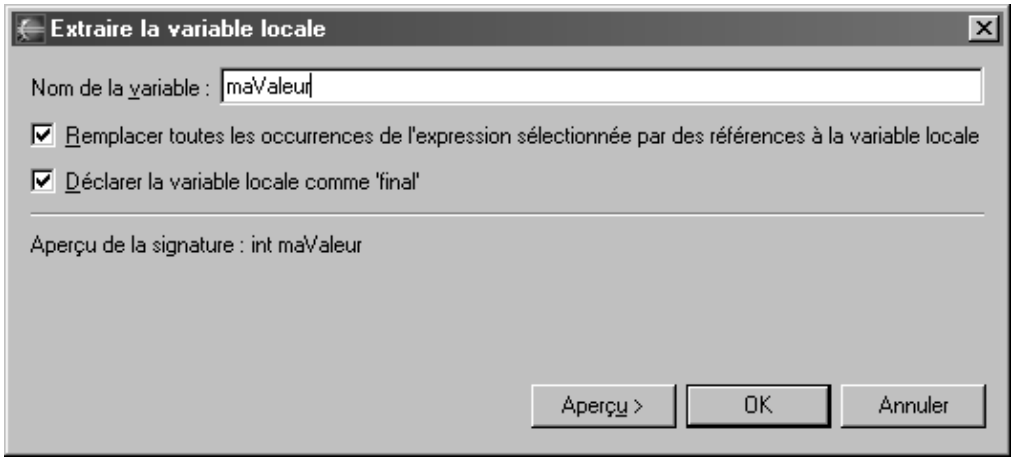
8.14. Extraire la variable locale

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

Il faut sélectionner une expression dans le code source.

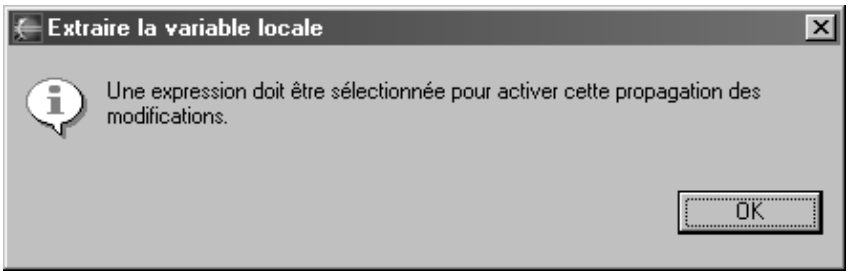


Puis utiliser l'option « Extraire la variable locale » du menu « Propager les modifications ».

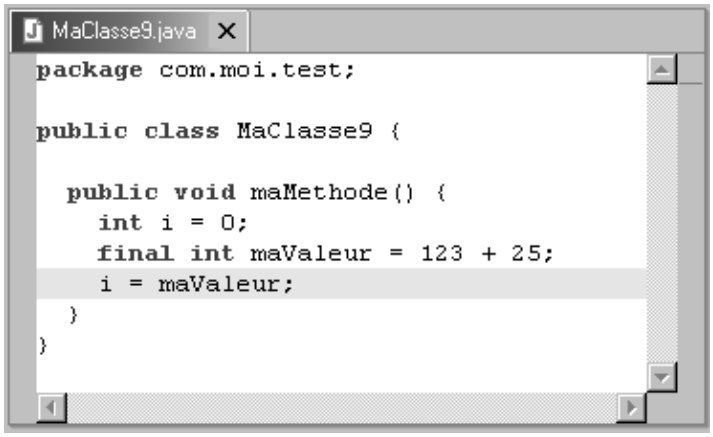


Une boîte de dialogue permet de saisir le nom de la variable, de préciser si toutes les occurrences doivent être modifiées et si la variable doit être une constante.

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

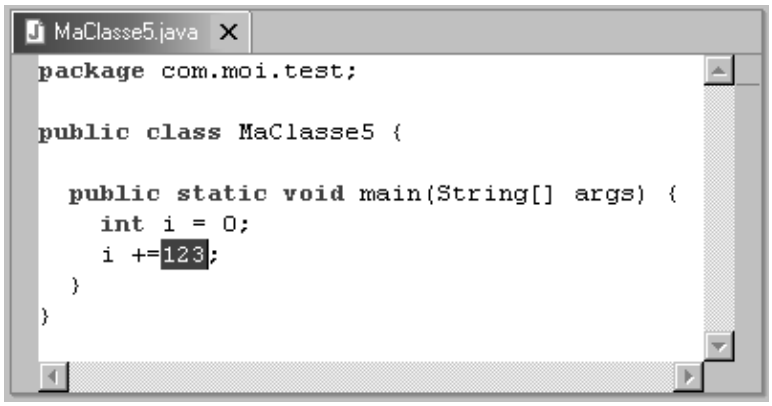


Le résultat de l'exécution de cette fonction est le suivant :

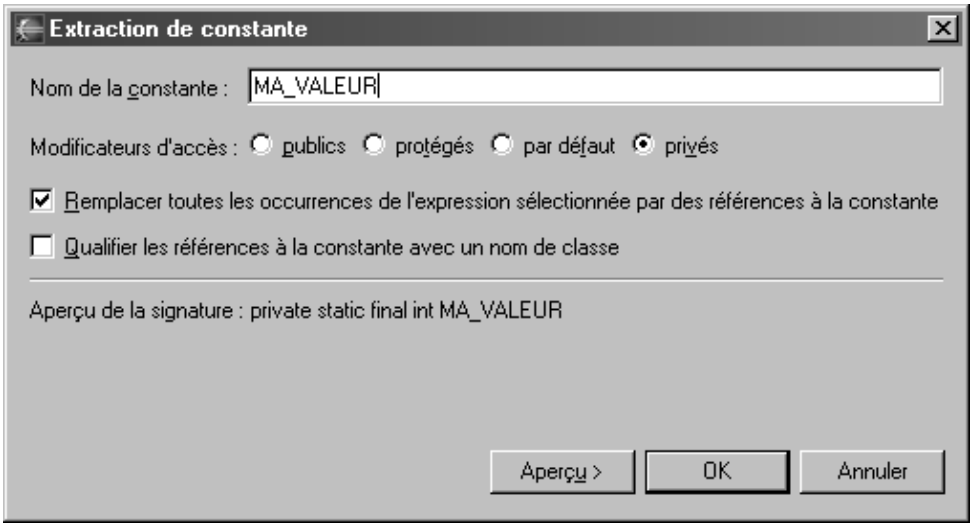


8.15. Extraire une constante

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.

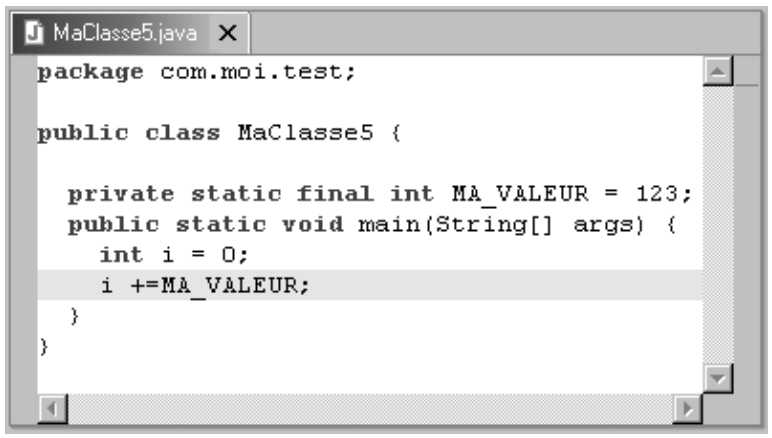


Puis d'utiliser l'option « Extraire une constante … » du menu « Propager les modifications ». Une boite de dialogue apparaît pour préciser le nom de la constante à créer et son modificateur d'accès.



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.



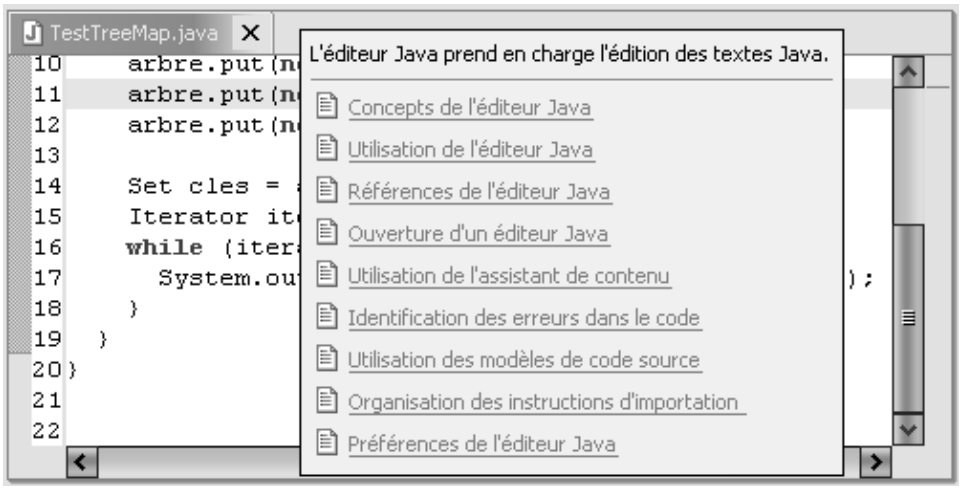
9. L'aide dans Eclipse

9.1. 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.

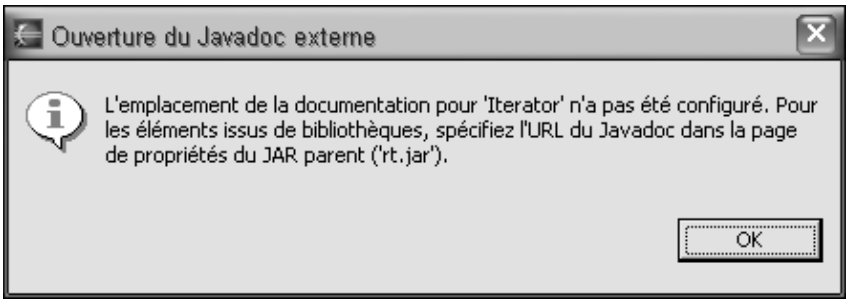
Exemple avec l'appui sur F1 dans l'éditeur de code Java



9.2. 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.



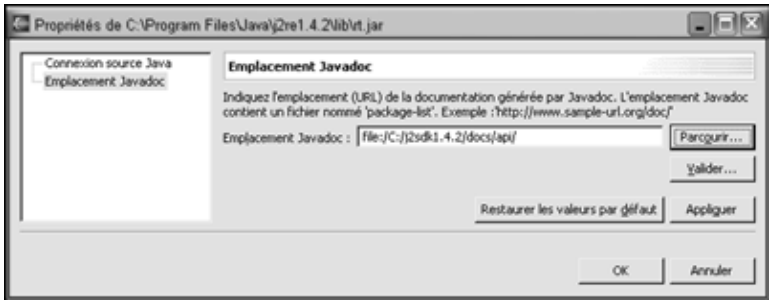
Pour configurer l'IDE, il faut sélectionner la ressource qui contient l'élément dans la vue "Packages" :

- un projet
- un fichier .jar

Exemple : 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.



Enfin, il faut cliquer sur "OK" pour valider les changements.

Si le chemin n'est pas correct, un message d'erreur est affiché.

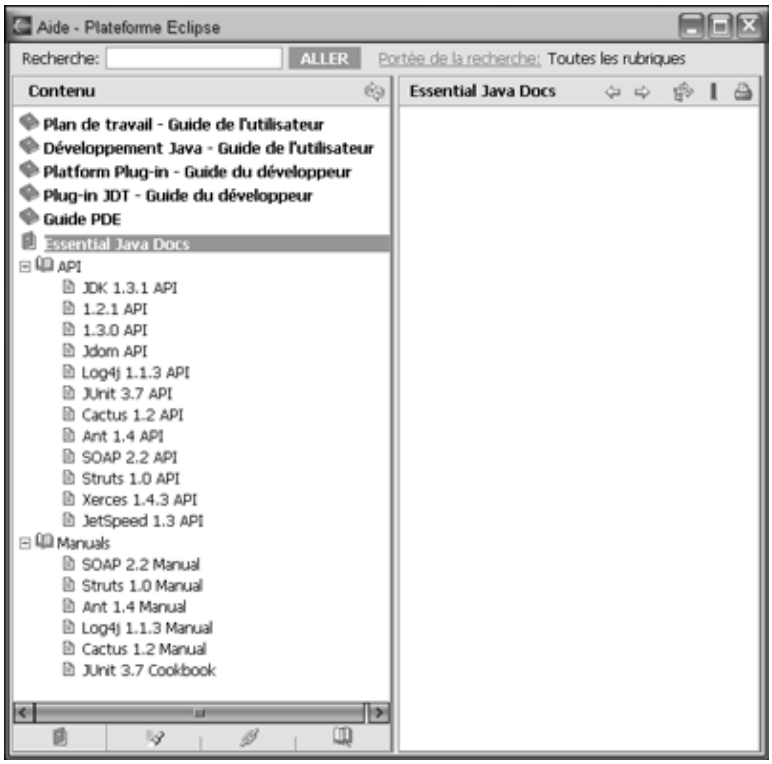
Il est alors possible dans l'éditeur de code Java de positionner le curseur sur un élément contenu dans un des emplacements Javadoc pour que l'aide en ligne affiche la page concernée par l'élément dans la documentation javadoc.

Ce processus est applicable à toutes les API dont la documentation Javadoc est disponible.

9.3. Le plug in Java docs de Crionics

La société Crionics propose un plug in qui s'intègre dans l'aide en ligne de Eclipse et qui contient la documentation du JDK 1.3 et de quelques API open source.

Ce plug in créée une entrée nommée "Essentials Java Docs" dans la table des matière de l'aide en ligne. Cette documentation regroupe les API essentielles.



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 dézipper son contenu dans le répertoire plug in d'intallation d'Eclipse.

10. 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 d'applications sont très importants car ils permettent d'automatiser des opérations répétitives tout au long du cycle de vie de l'application (développement, tests, recettes, mises en production ...). Le site officiel de ant est http://jakarta.apache.org/ant/index.html.

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.

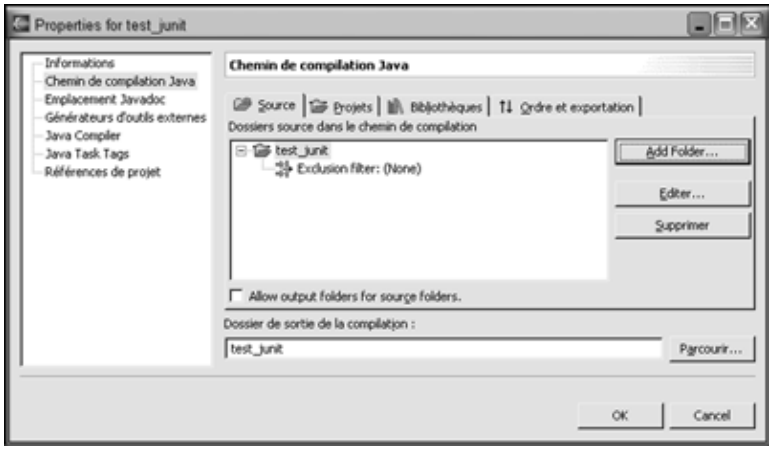
Le fichier de configuration contient un ensemble de cible (target). Chaque cible contient une ou plusieurs tâches. Chaque cible peut avoir une dépendance envers une ou plusieurs autres cibles pour pouvoir être exécutée.

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

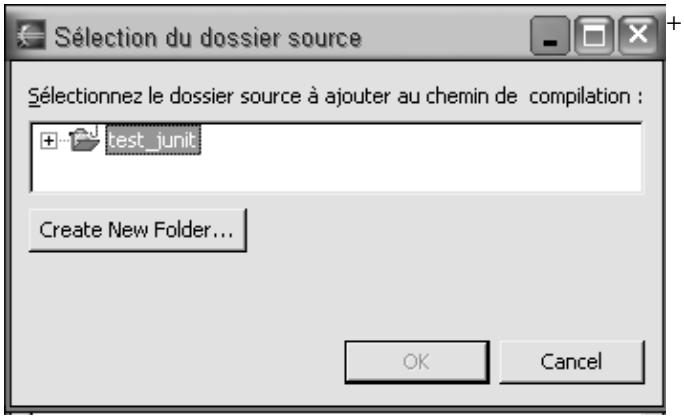
10.1. Structure du projet

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.

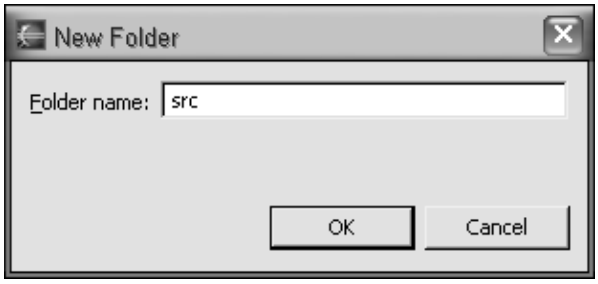


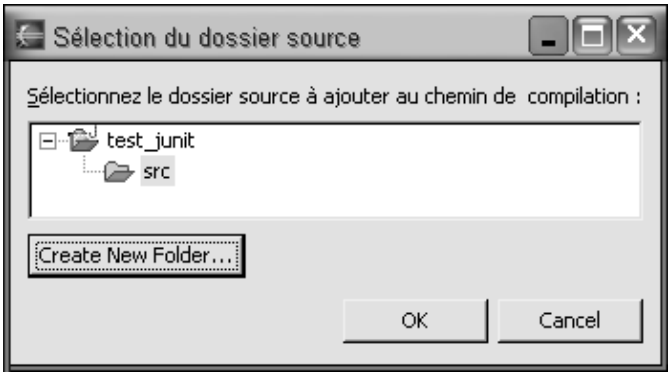
Cliquer sur le bouton "Add Folder".



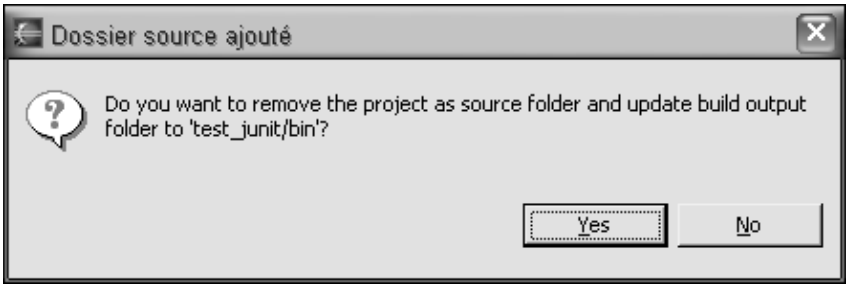
Cliquer sur le bouton "Create New Folder".

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



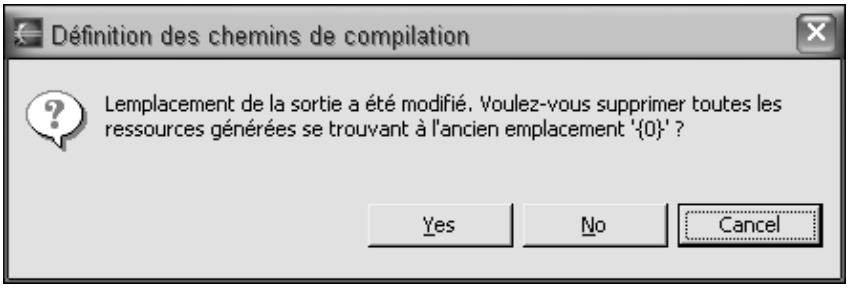


Cliquer sur "OK".



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

Cliquer sur "OK".



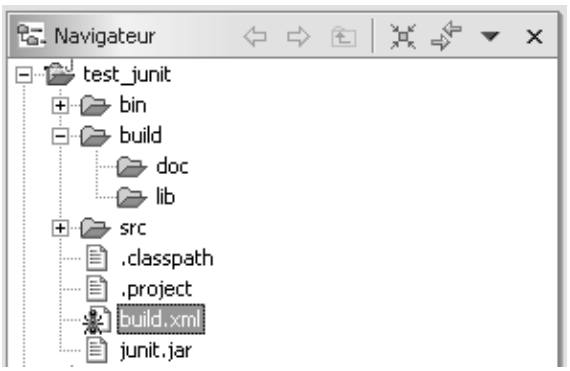
Cliquer sur "Yes".

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.

10.2. 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.



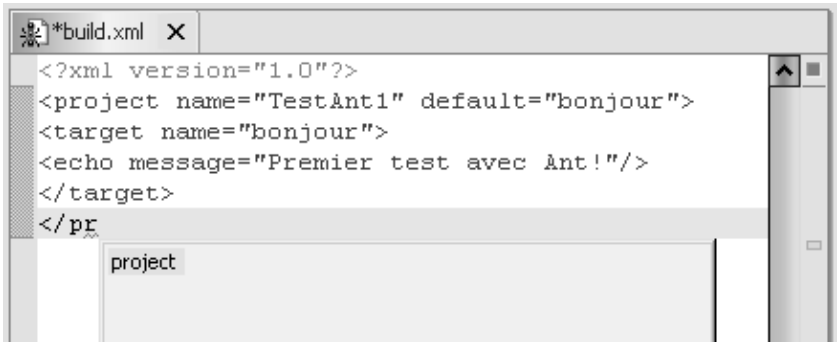
Le fichier est automatiquement reconnu comme étant un fichier de configuration pour Ant : une icône particulière contenant une fourmi est associée au fichier.

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

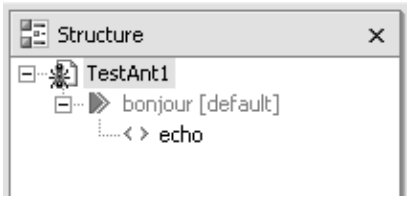
Exemple : afficher un message

```
<?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 Ctrl + espace.



La vue structure affiche l'arborescence du fichier.

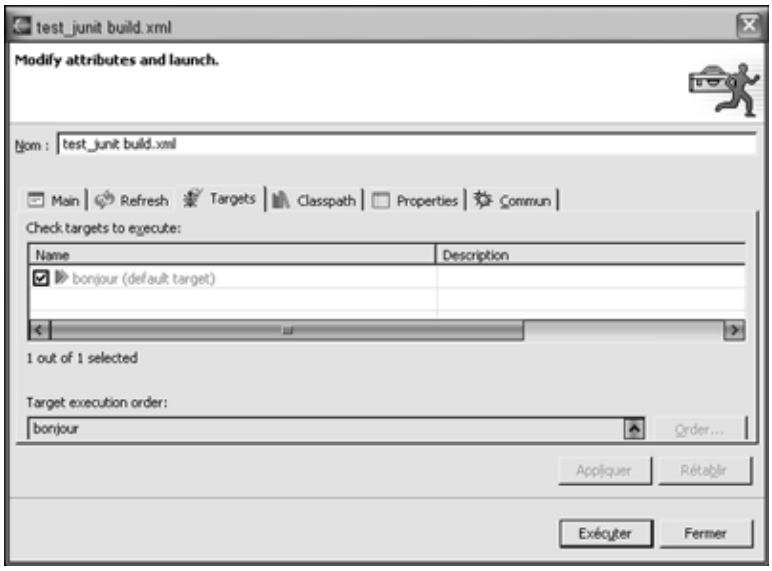


Une fois le contneu du fichier saisi, il suffit de l'enregistrer.

10.3. 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".

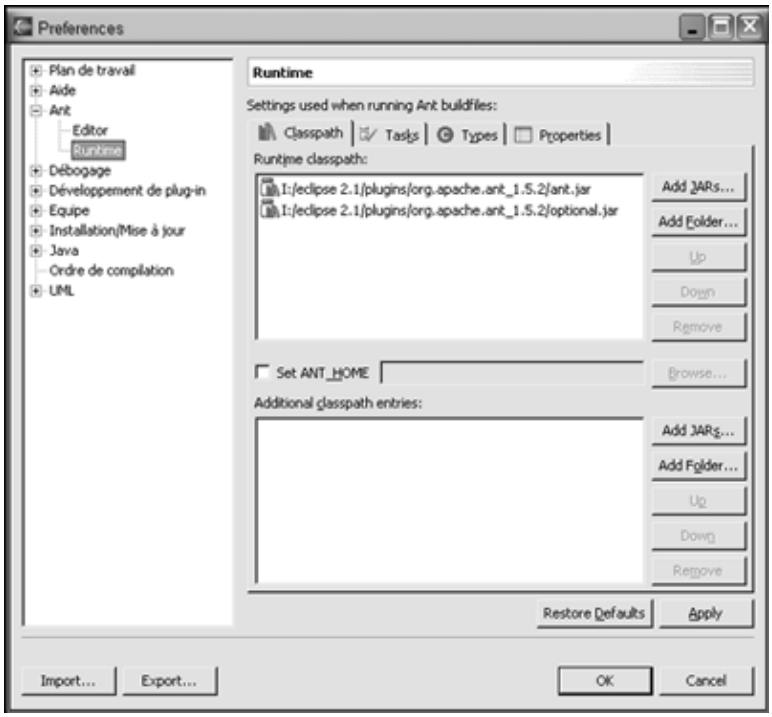
Le résultat de l'exécution s'affiche dans la vue "Console"

Exemple :

```
Buildfile: I:\eclipse 2.1\workspace\test_junit\build.xml

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

10.4. Les paramètres

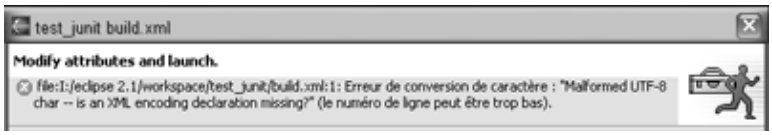
Dans les préférences, il est possible de préciser des paramètres par défaut utilisés lors de l'édition d'un fichier ant ou de son exécution.



10.5. Résolution des problèmes

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

10.5.1. 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" ?>`

10.5.2. Impossible de lancer la tache javadoc

Exemple :
`Buildfile: I:\eclipse 2.1\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 2.1/workspace/test_junit/build.xml:37:
     Javadoc failed: java.io.IOException: CreateProcess: javadoc.exe -d "I:\eclipse
     2.1\workspace\test_junit\build\doc" -use -package -classpath "I:\eclipse 2.1\s
     tartup.jar;I:\eclipse 2.1\workspace\test_junit\junit.jar" -version
     -author "I:\eclipse 2.1\workspace\test_junit\src\MaClasse.java" "I:\eclipse
     2.1\workspace\test_junit\src\MaClasse2.java" error=2
Total time: 681 milliseconds
```

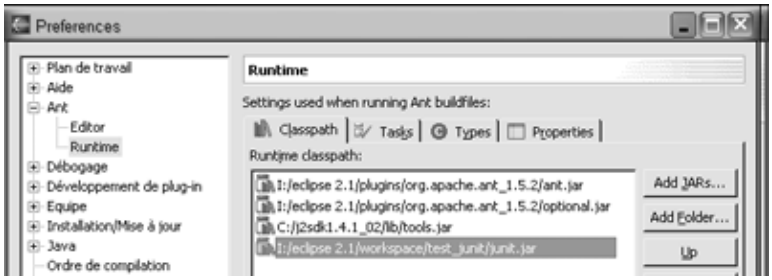
Il faut vérifier la présence de l'outil dans les répertoires désignés par la variable d'environnement PATH du système d'exploitation. Dans le cas de javadoc sous Windows, il faut s'assurer que %JAVA_HOME%\bin soit insérer dans la variable PATH. Si cette dernière doit être modifiée, il faut arrêter et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

10.5.3. Impossible d'utiliser la tâche JUnit

Exemple :

```
Buildfile: I:\eclipse 2.1\workspace\test_junit\build.xml
init:
        [echo] Generation numero : 13 du July 2 2003
compil:
test:
       [junit] BUILD FAILED: file:I:/eclipse 2.1/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"



10.6. 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 2.1\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 2.1\workspace\test_junit\src\MaClasse.java...
     [javadoc] Loading source file I:\eclipse 2.1\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
```

11. 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.

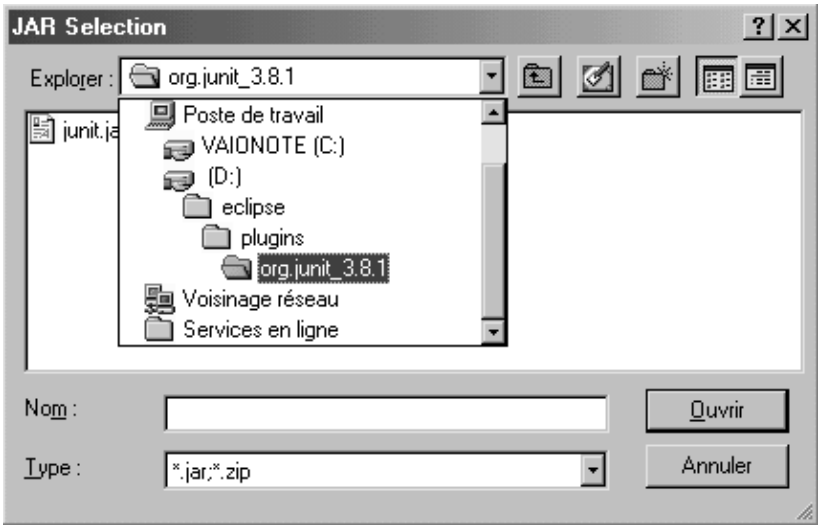
Avec Junit, l'unité de test est une classe dédiée qui regroupe des cas de tests. Ces cas de tests exécutent les tâches suivantes :

- création d'une instance de la classe et de tout autre objet necessaire 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

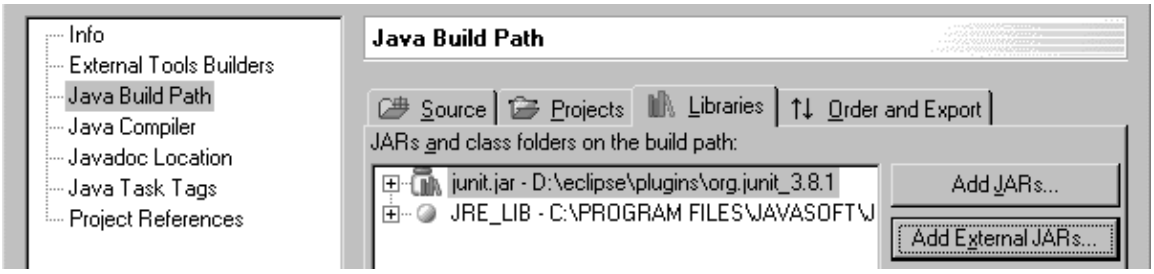
11.1. Paramétrage de l'environement

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 « 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



La classe suivante est utilisée comme classe à tester avec JUnit.

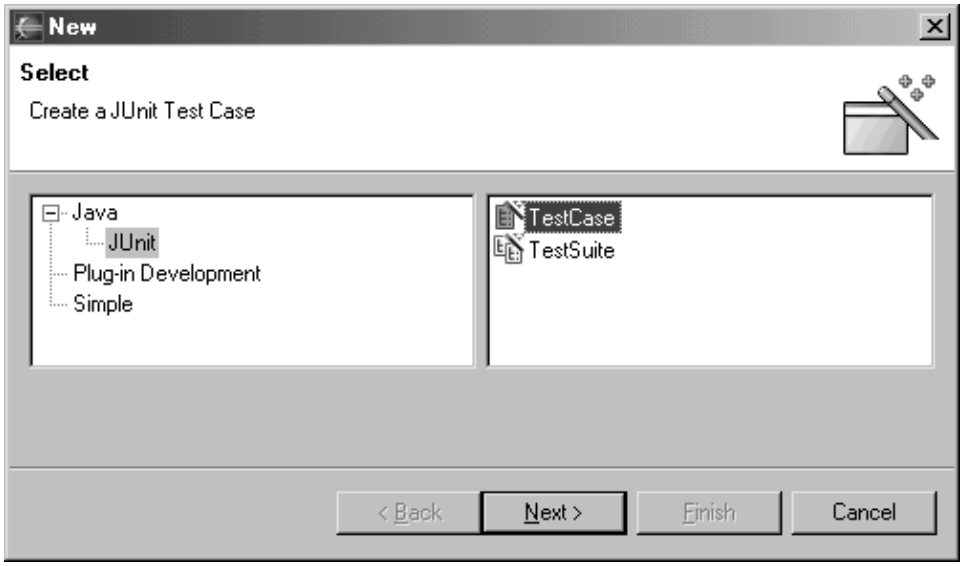
Exemple :

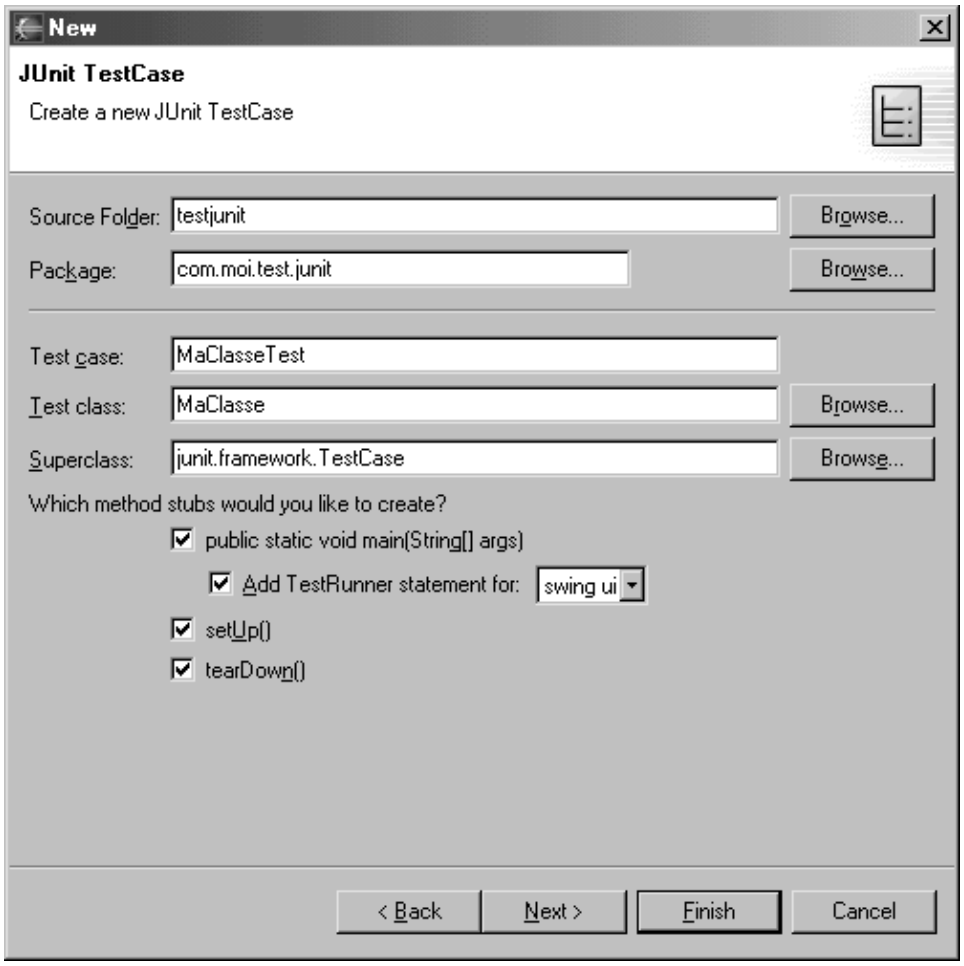
```
package com.moi.test.junit;
public class MaClasse {
  public int additioner(int a, int b) {
    return a + b;
  }
}
```

11.2. Ecriture des cas de tests

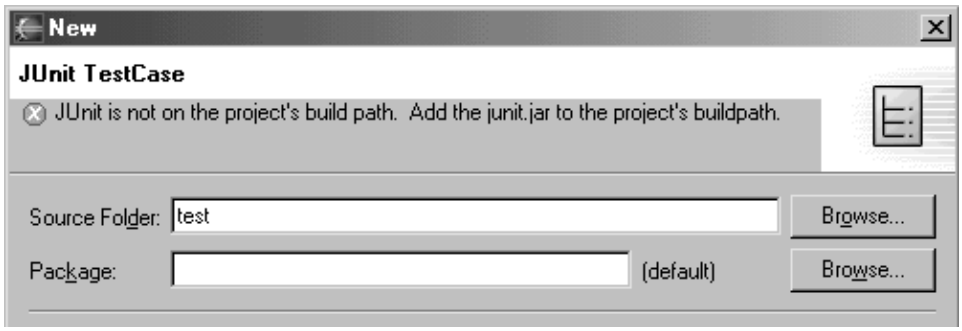
Pour utiliser JUnit, il faut créer une classe qui va contenir les cas de test.

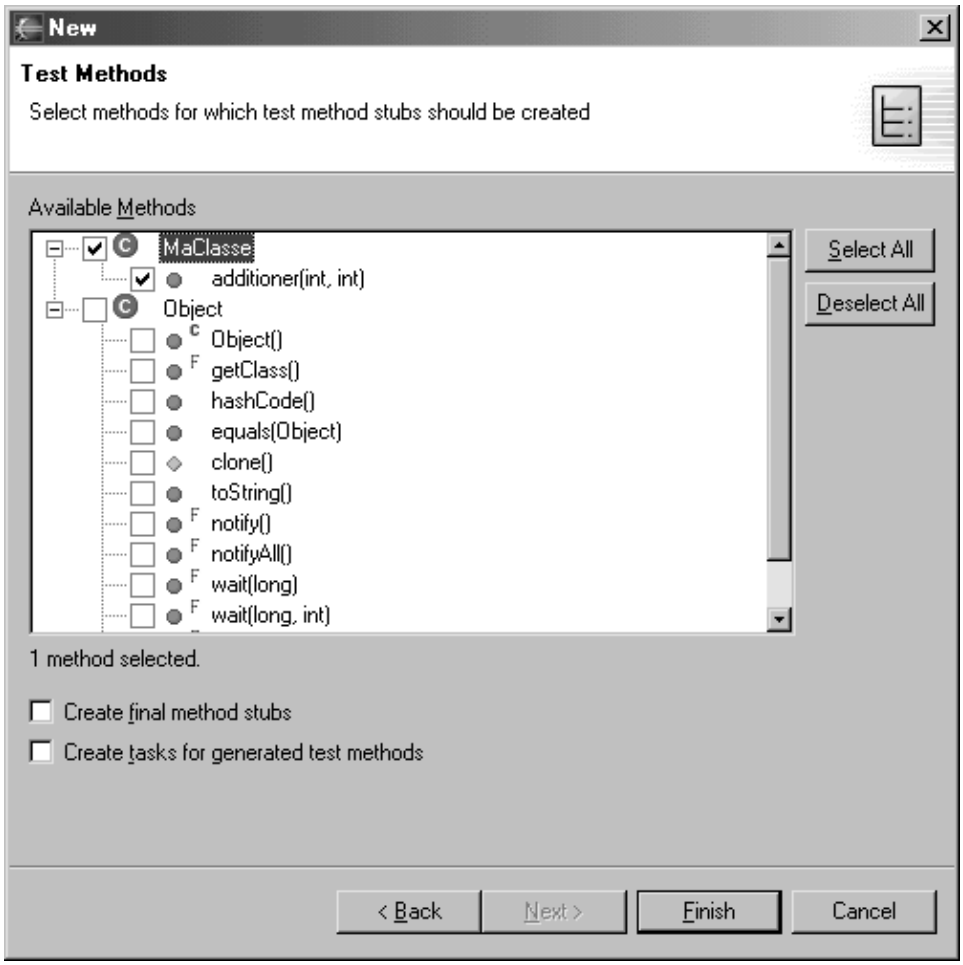
Il faut créer une nouvelle entité de type Java/JUnit/TestCase





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.





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 :

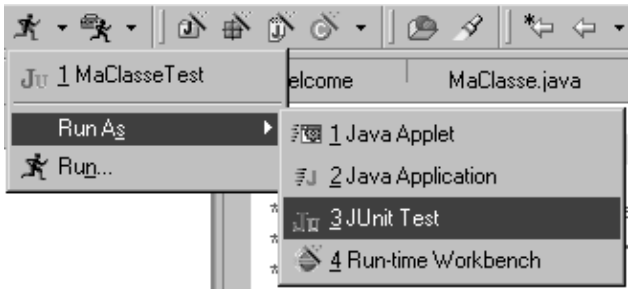
```
package com.moi.test.junit;

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;
   }
```

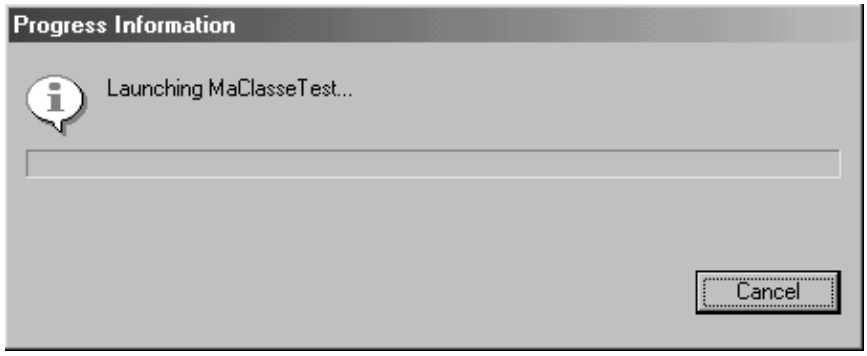
```
   public void testAdditioner() {
         assertTrue(maClasse.additioner(2,2) == 4);
   }
}
```

JUnit utilise l'instrospection pour exécuter les méthodes commençant par test.

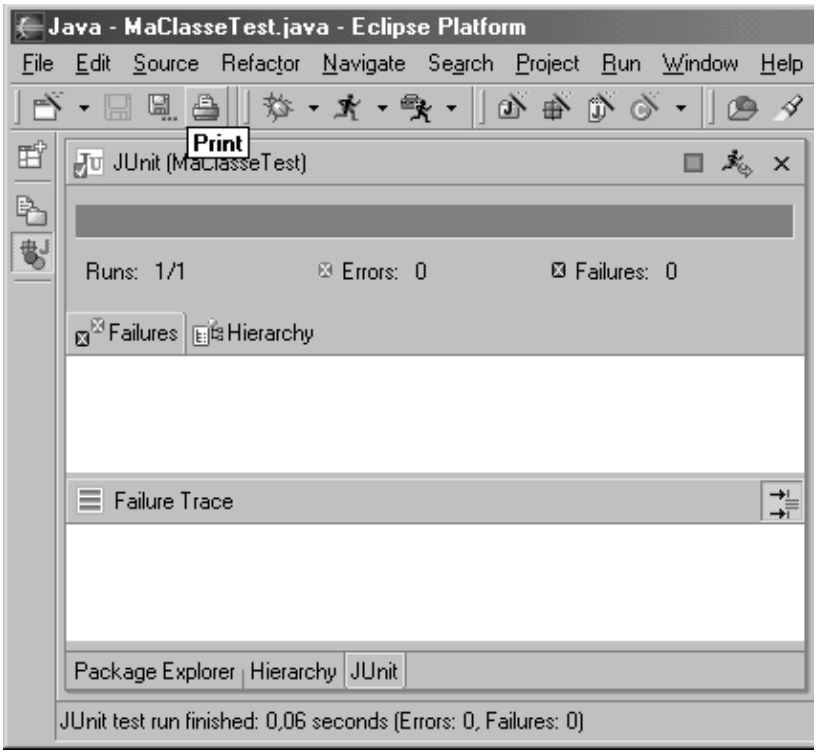
11.3. Exécution des cas de tests

Pour exécuter les tests, il faut exécuter la classe en tant que « JUnit Test ».





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 :

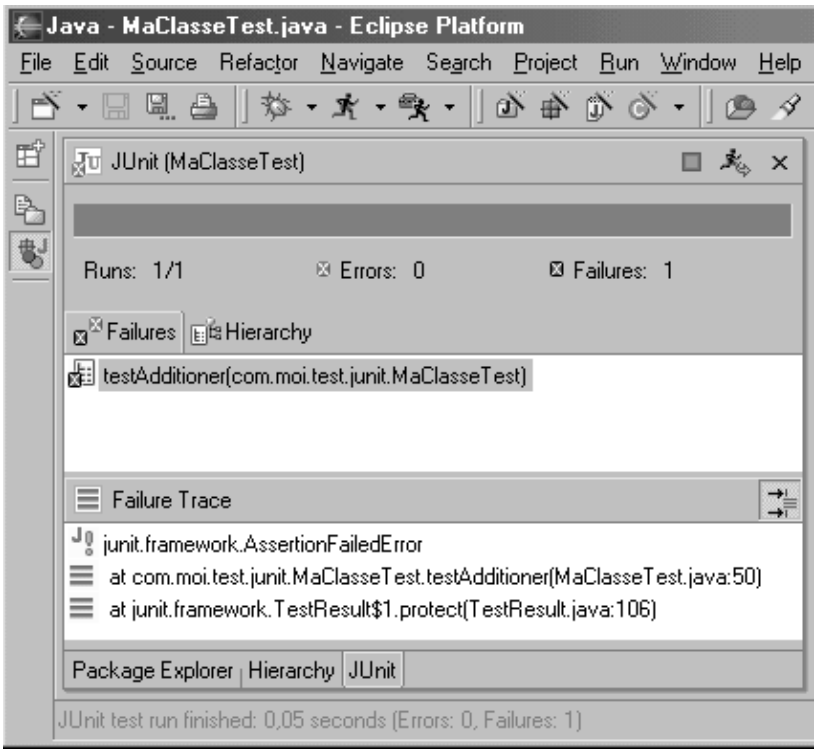
- « Failures » : contient la liste des cas de tests qui ont échoués
- « Hierarchy » : contient une arborescence des cas de tests

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);
}
```

Un exécution permet d'avoir le résultat suivant :


Java - MaClasseTest.java - Eclipse Platform
File Edit Source Refactor Navigate Search Project Run Window Help
JUnit (MaClasseTest)
Runs: 1/1
Errors: 0
Failures: 1
Failures
Hierarchy
testAdditioner(com.moi.test.junit.MaClasseTest)
Failure Trace
junit.framework.AssertionFailedError
at com.moi.test.junit.MaClasseTest.testAdditioner(MaClasseTest.java:50)
at junit.framework.TestResult$1.protect(TestResult.java:106)
Package Explorer
Hierarchy
JUnit
JUnit test run finished: 0,05 seconds (Errors: 0, Failures: 1)

12. CVS et Eclipse

CVS (Concurrent Versions System) est un outil libre de gestion des 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.

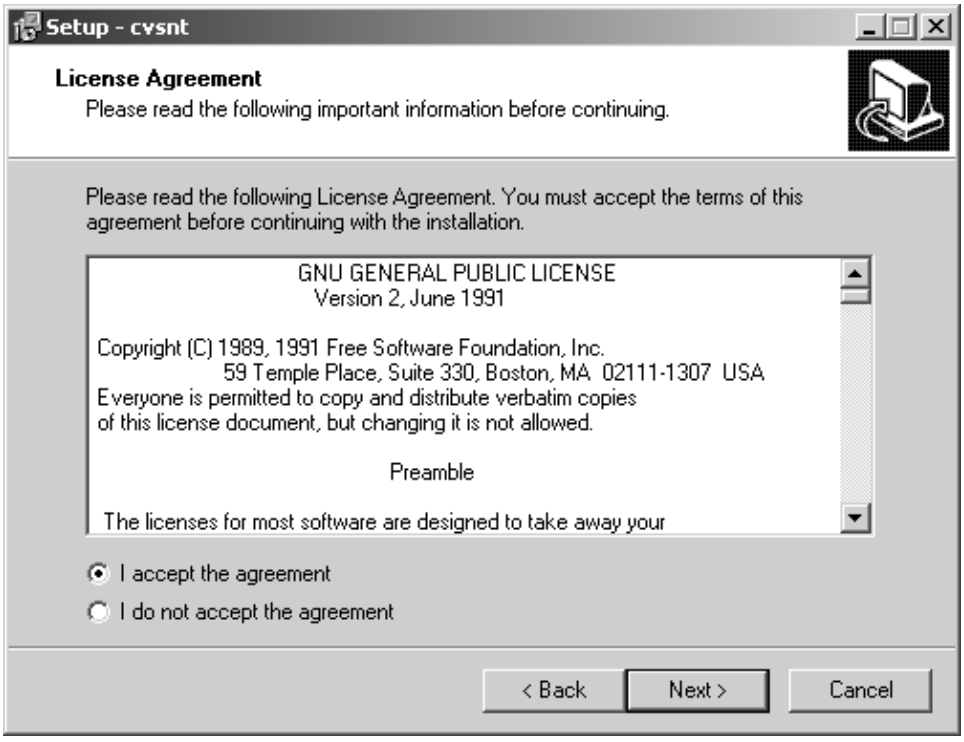
Eclipse propose une perspective pour utiliser CVS dans un projet.

12.1. Installation de cvsnt

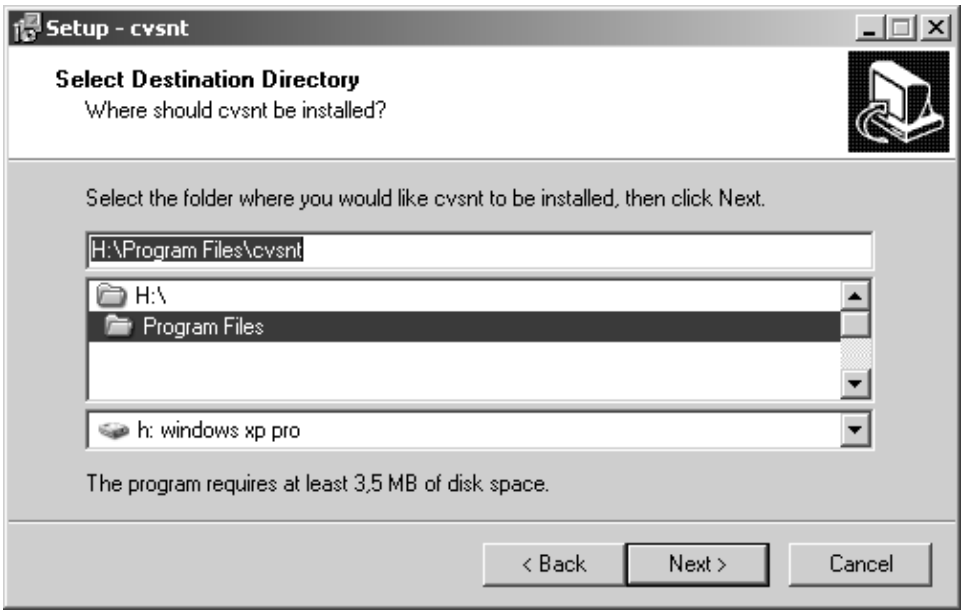
Il faut créer deux répertoires, par exemple c:\cvs\cvsrepo et c:\cvs\cvstemp

Exécuter le programme d'installation cvsnt−2.0.0.exe

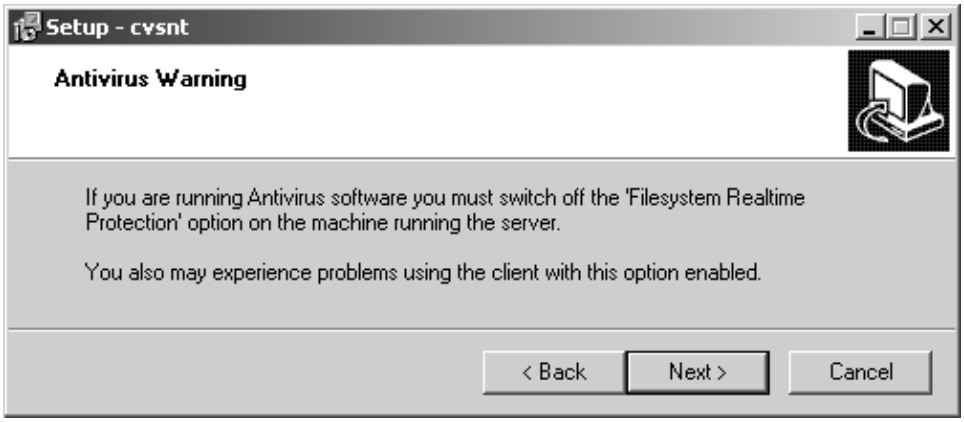
Cliquer sur «Next»

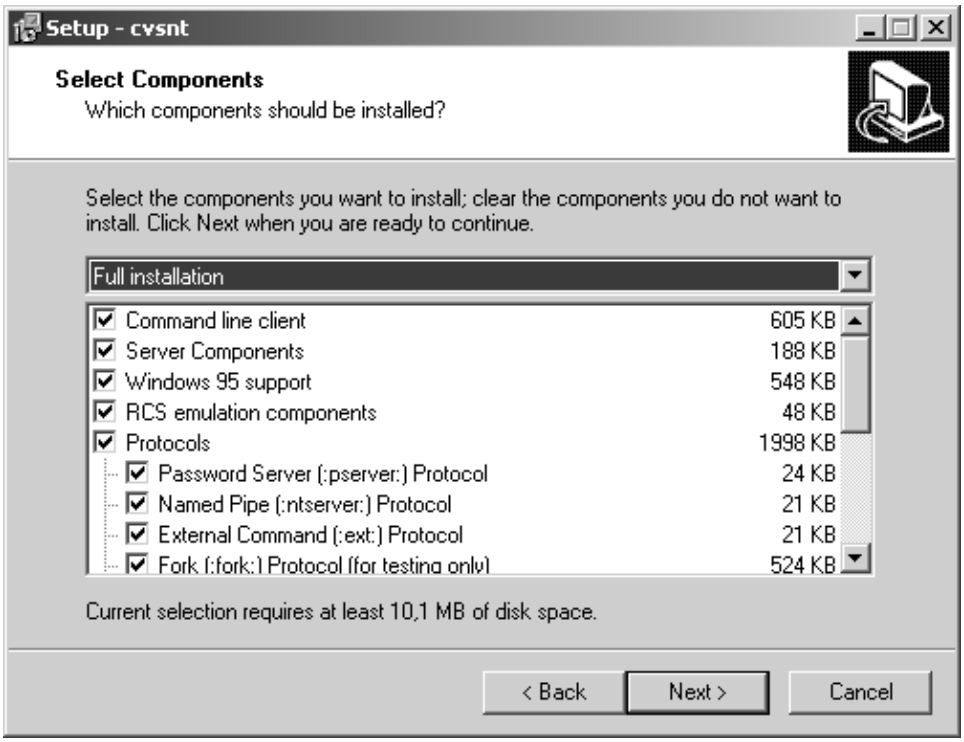


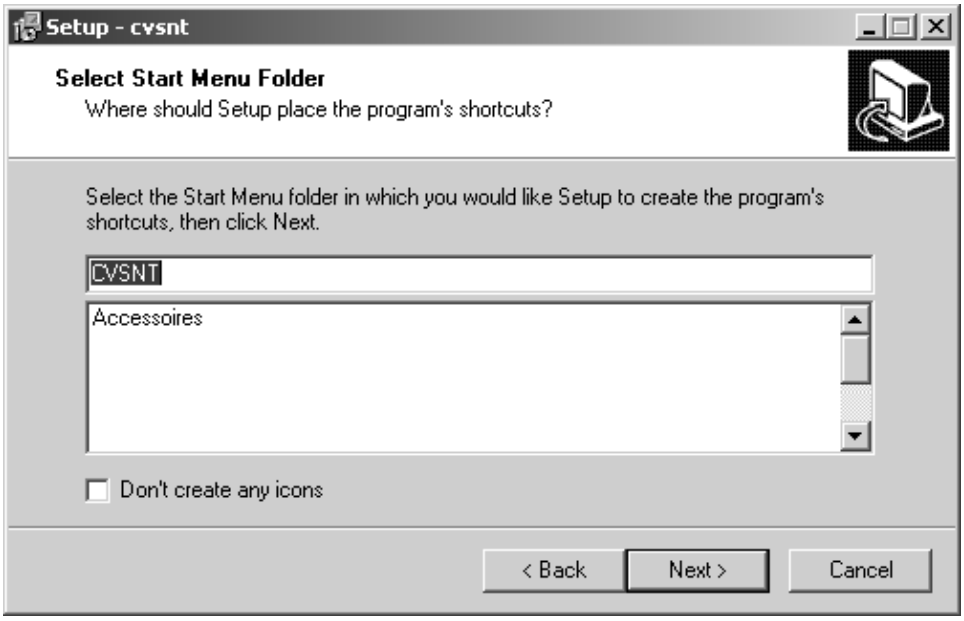
Accepter la licence et cliquer sur «Next»

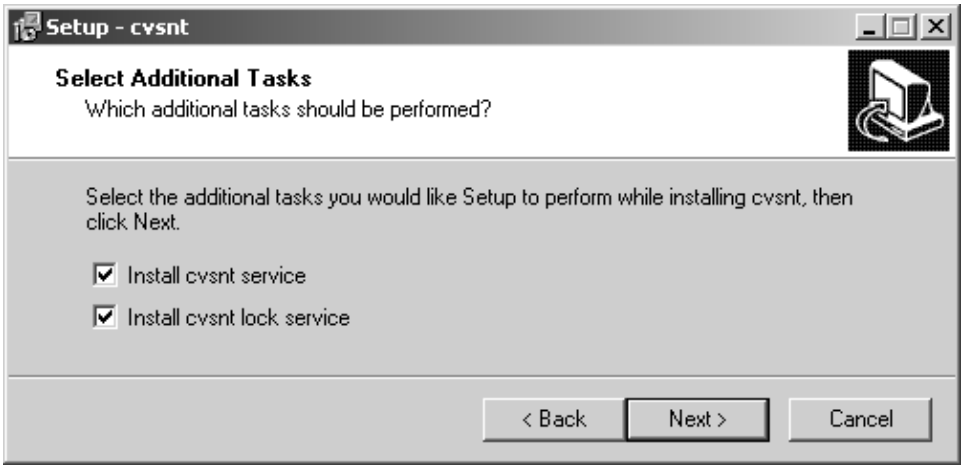


Sélectionner le répertoire d'installation et cliquer sur «Next»

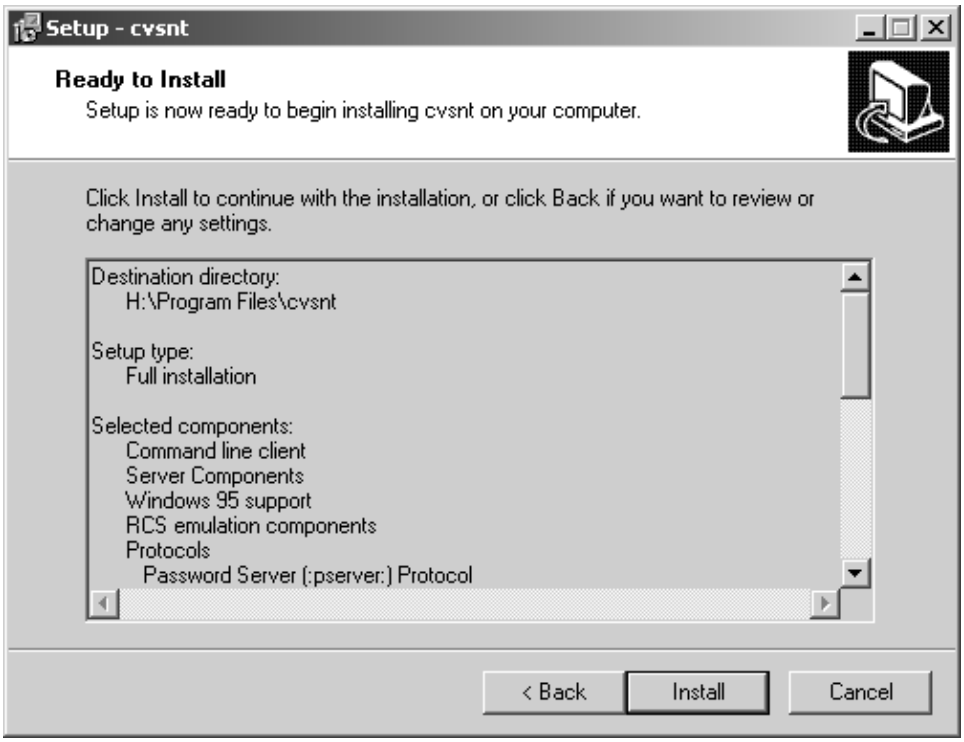




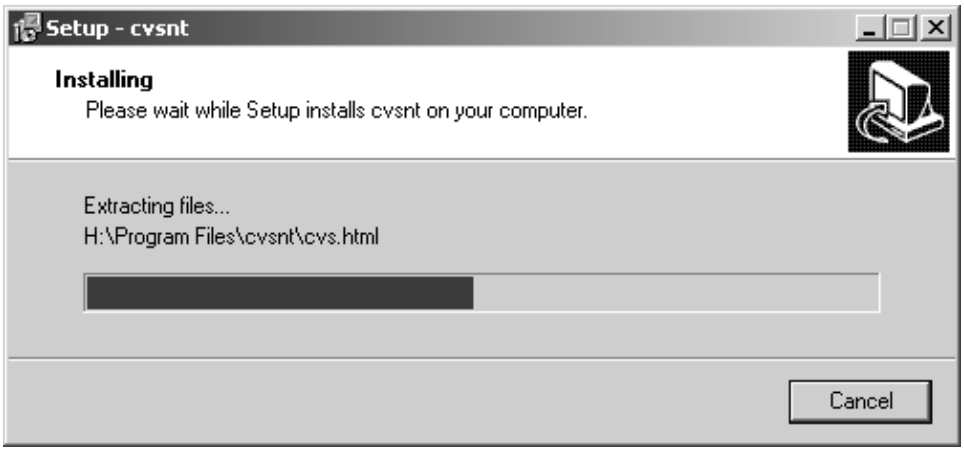




Cliquer sur «Next»



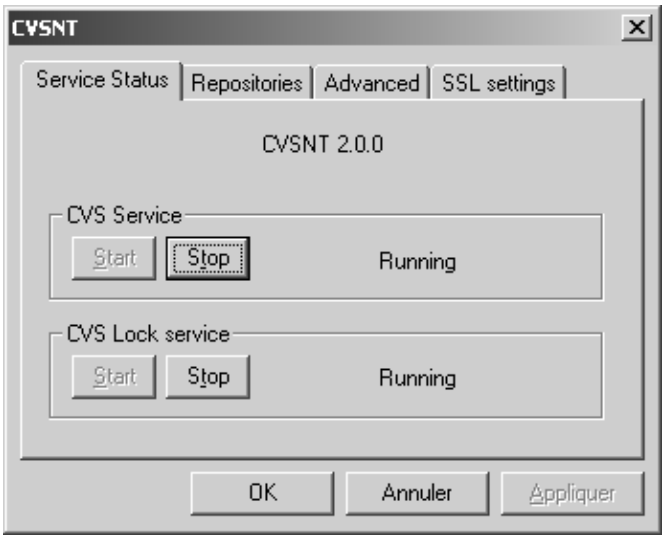
Cliquer sur «Install»



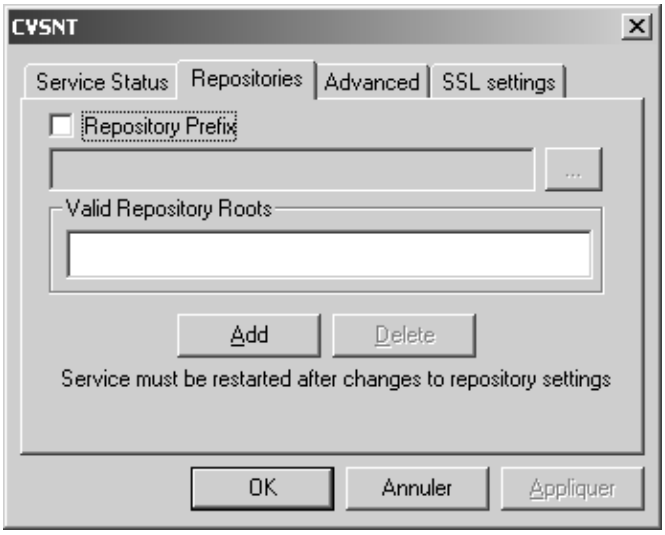
Une fois l'installation terminée, cliquer sur «Finish».

Exécuter «le service control panel» en cliquant sur l'icône 

dans le menu "Démarrer/Programmes/CVTNT" ou dans le panneau de configuration



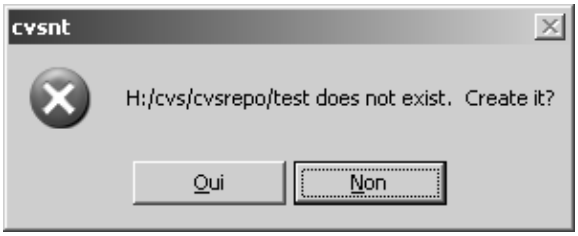
Dans l'onglet «Repositories»,



Cliquer sur «Repository prefix» et sélectionner le répertoire cvsrepo précédemment créé.

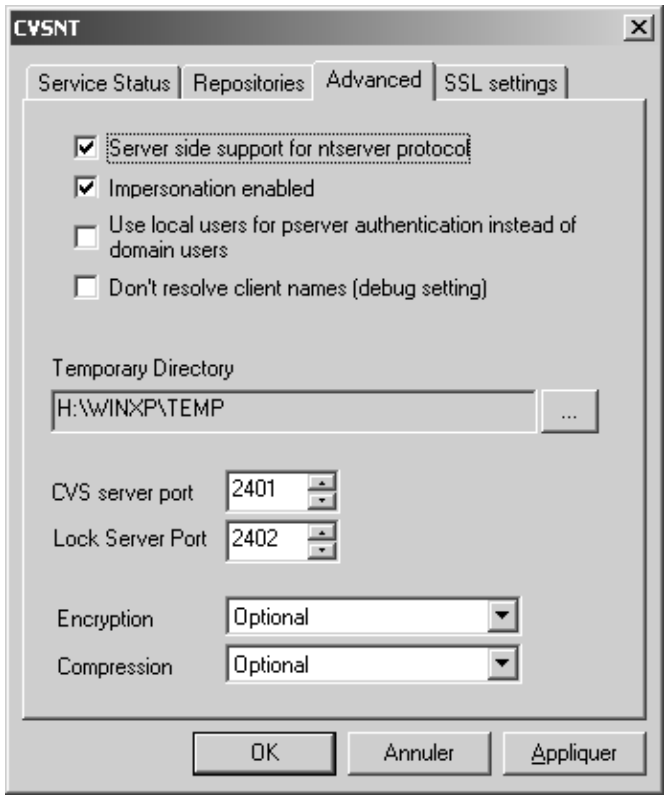
Cliquer sur «Add»

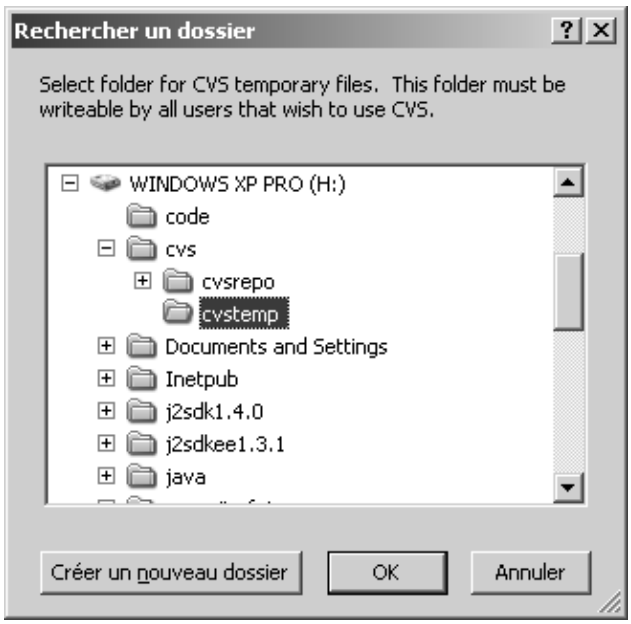
Saisir le nom du répertoire et cliquer sur «Ok»



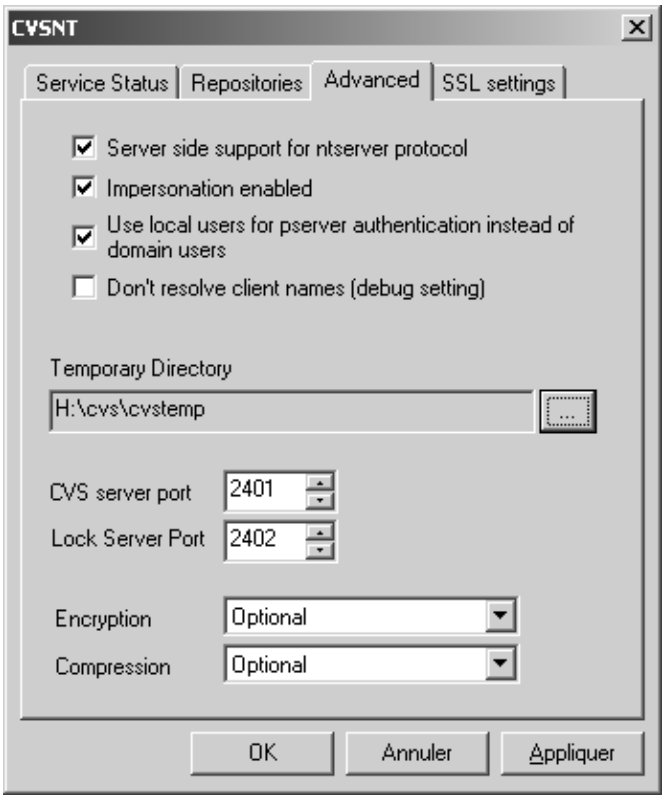
Cliquer sur «oui»

Sur l'onglet «Advanced»

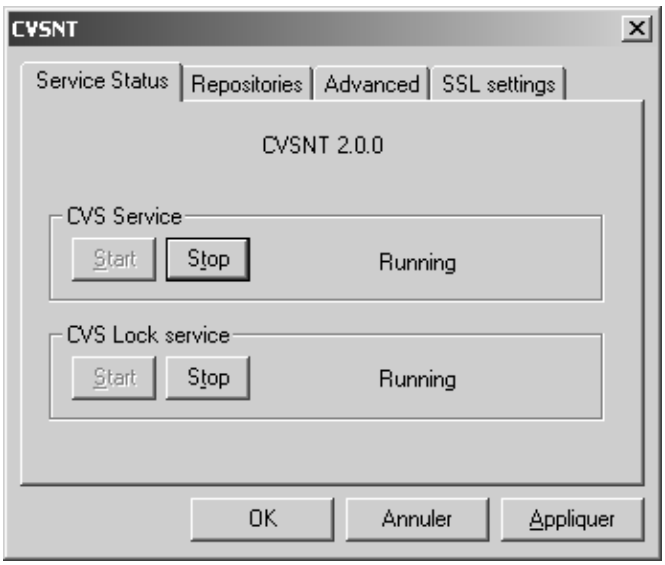




Sélectionner le répertoire temporaire précédemment créé



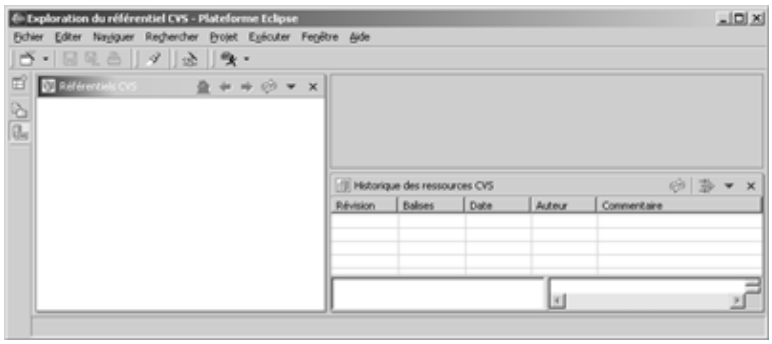
Sur l'onglet «Service Status», cliquer sur «Start»



Cliquer sur “Ok”

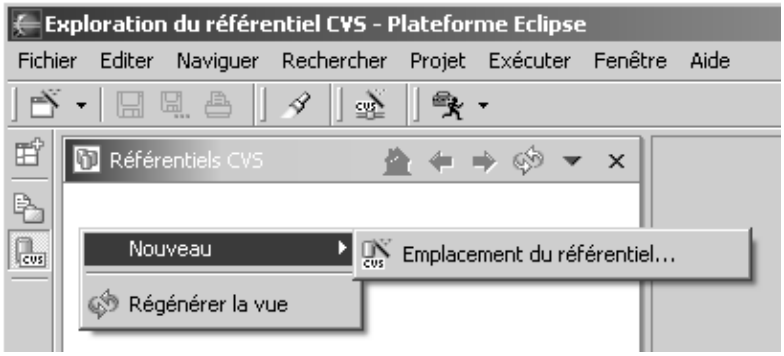
12.2. La perspective CVS

La perspective «Exploration du référentiel CVS» permet de gérer les échanges et le contenu des projets stockés sous CVS.

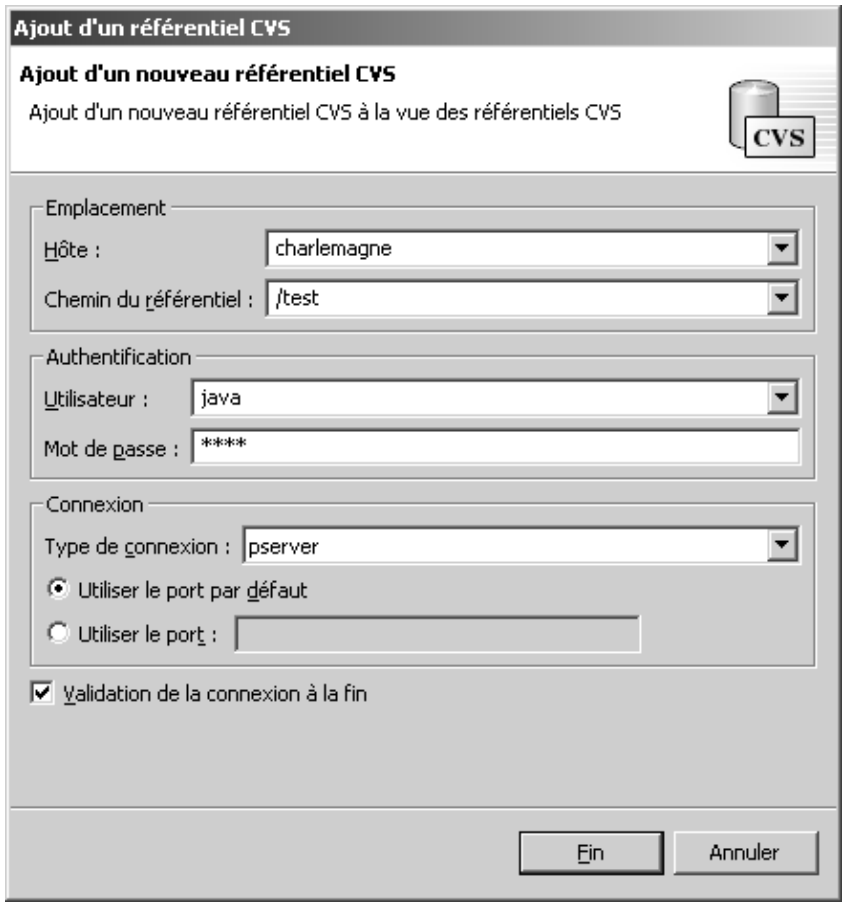


12.2.1. La création d'un emplacement vers un référentiel

Dans la vue "Référentiels CVS", sélectionner l'option "Nouveau/Emplacement du référentiel" du menu contextuel.



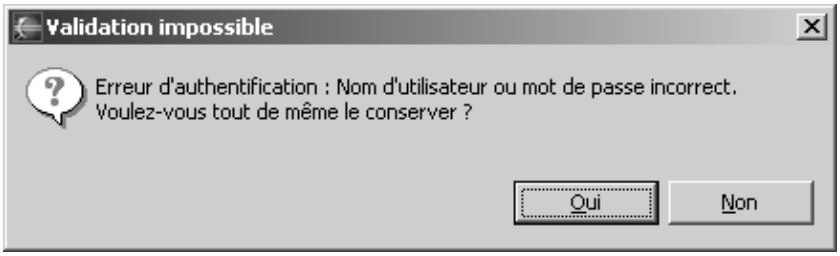
Une boite de dialogue s'ouvre pour définir un nouvel emplacement. Un emplacement contient uniquement les informations sur une connexion.



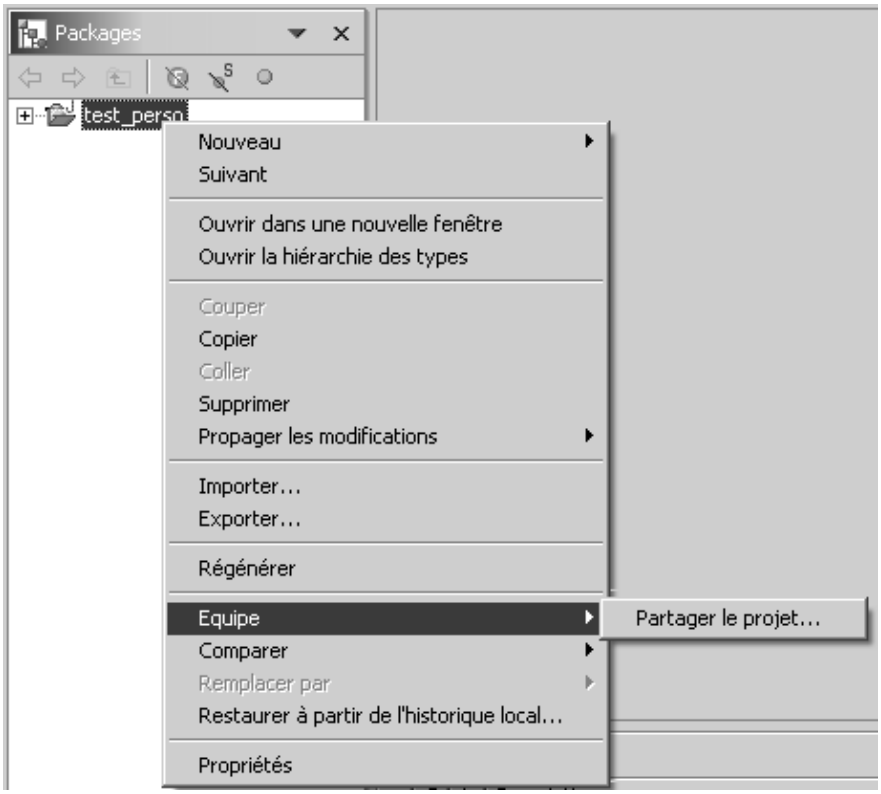
Renseigner le nom de la machine, le chemin du référentiel, le nom du user, son mot de passe (celui de windows) et le type de connexion (utiliser pserver).

Cliquer sur "Fin"

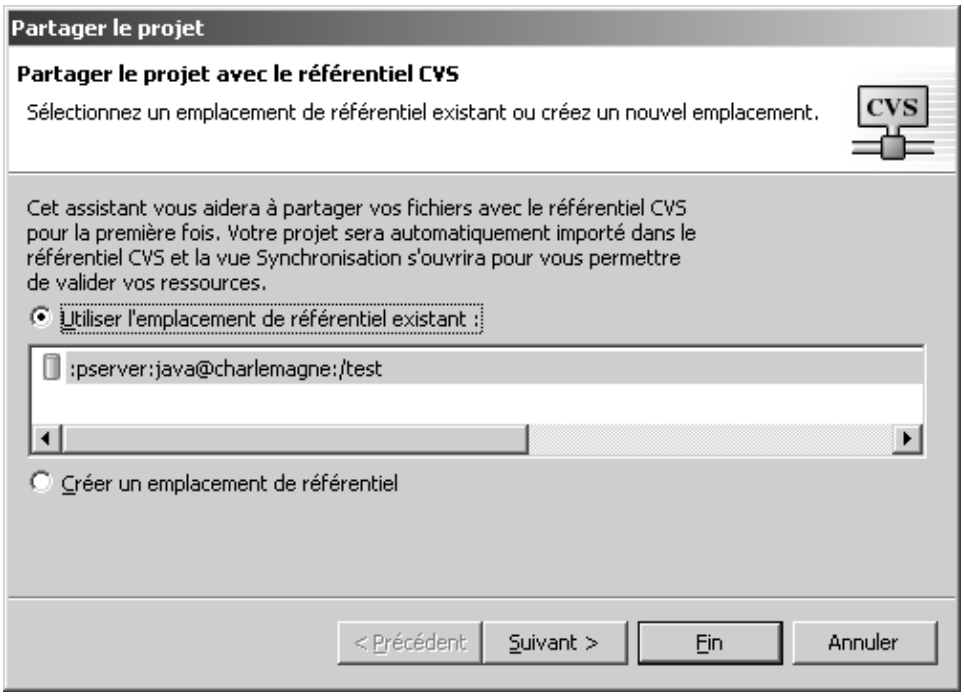
Si la connexion ne peut être établie, un message d'erreur est affiché



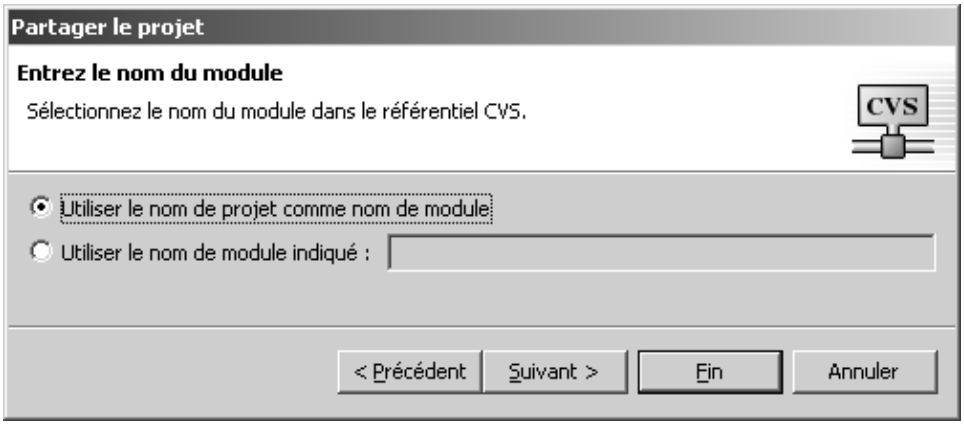
12.2.2. Partager un projet



Il faut sélectionner un projet dans une vue et sélectionner l'option "Equipe/Partager le projet".

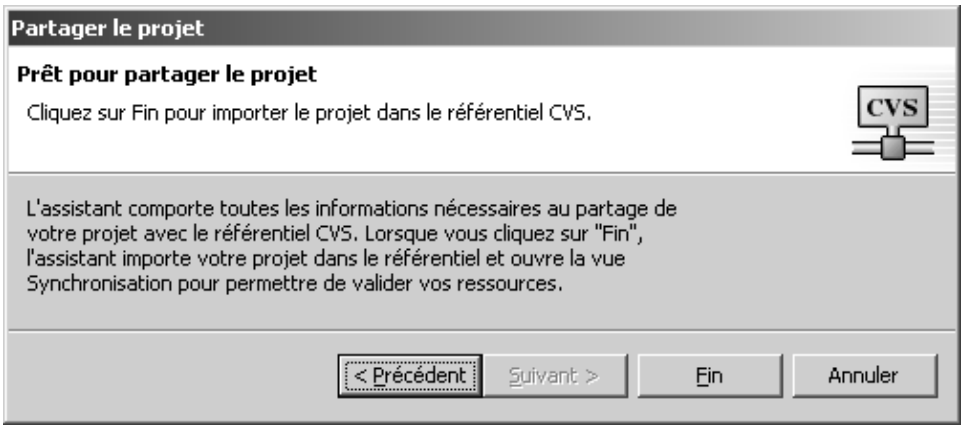


Cliquer sur "Suivant".



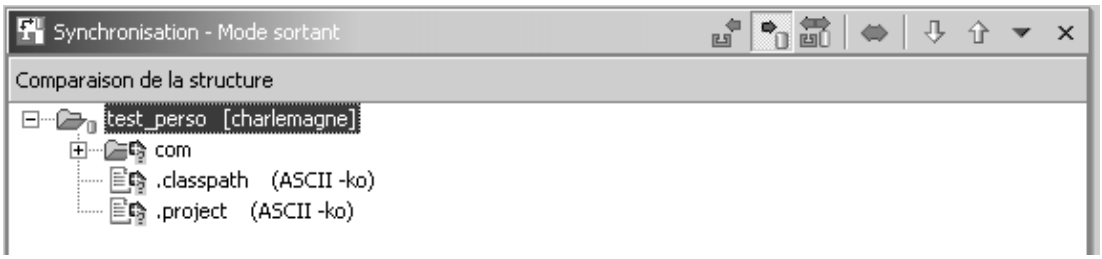
Cette étape permet de donner un nom au module: celui du projet Eclipse ou un nom spécifique.

Cliquer sur "Suivant".

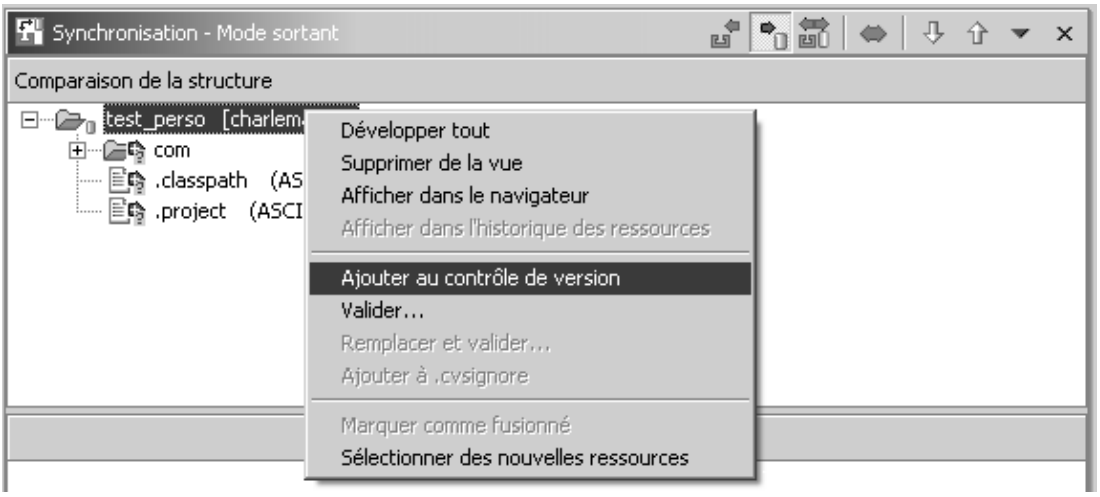


Cliquer sur "Fin".

La vue "Synchronisation − Mode sortant" affiche les fichiers qui ont été modifiés

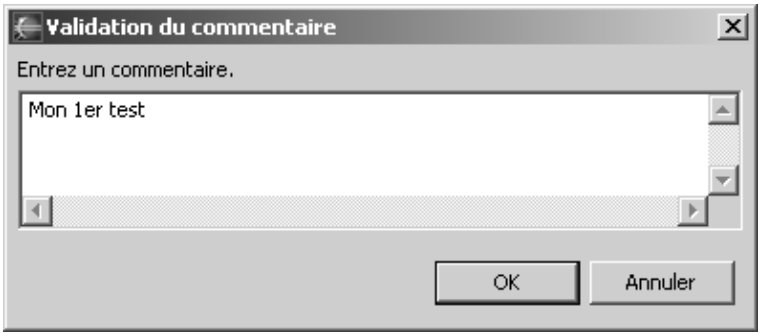


Dans cette vue, sélectionner le projet et activer l'option "Ajouter au contrôle de version" du menu contextuel.



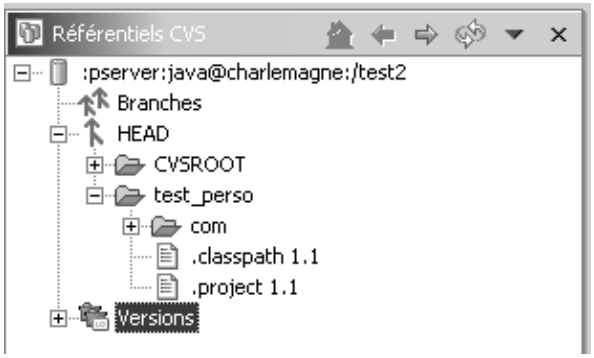
Une fois le traitement effectué, activer l'option "Valider" du menu contextuel

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.



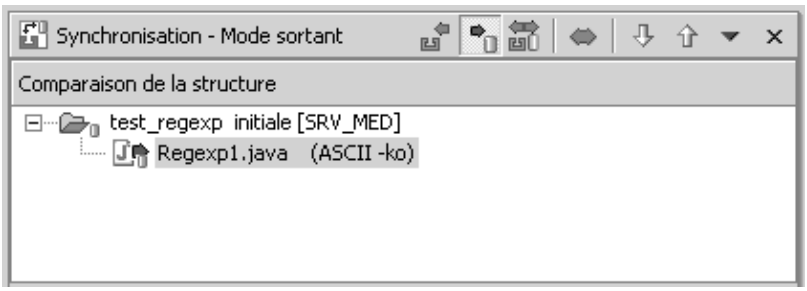
Remarque : si l'arborescence du projet n'est pas affichée, il suffit de cliquer sur le bouton et de cocher l'option "Afficher les dossiers".

12.3. L'utilisation des révisions

12.3.1. Créer une révision

Lorsqu'une ressource est modifiée et sauvegardée localement dans le workspace, 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.



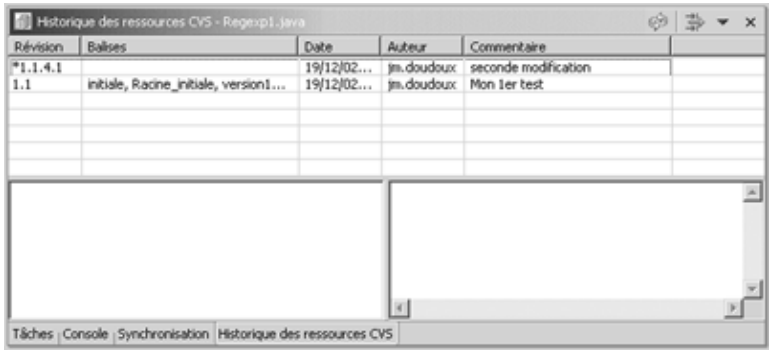
Dans cette vue, il suffit de sélectionner la ressource et d'activer l'option "Valider". Il est possible de saisir un commentaire.

La vue "Synchronisation" permet la synchronisation entre les ressources locales (dans le workspace) 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 le workspace
-  mode sortant : modifications contenues dans le workspace à intégrer dans le référentiel
-  mode entrant/sortant : modifications dans le workspace et le référentiel à intégrer dans l'un et l'autre

12.3.2. Gestion des révisions

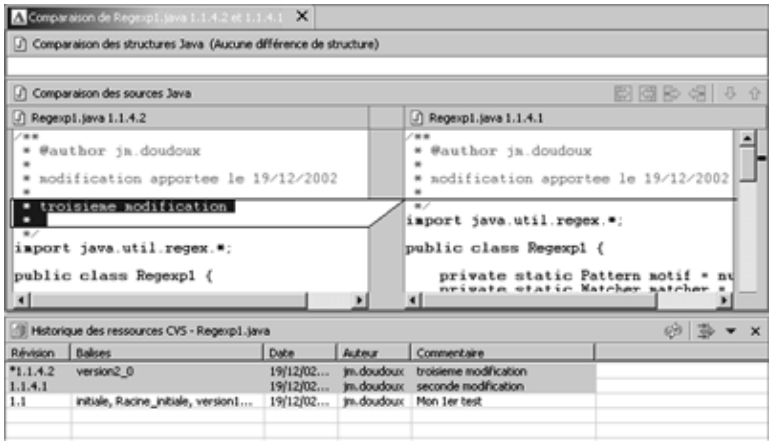
Il faut sélectionner une ressource et activer l'option "Equipe/Afficher dans l'historique des versions". La vue "Historique des ressources CVS" s'affiche en contenant les différentes révision de la ressource.



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 "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.

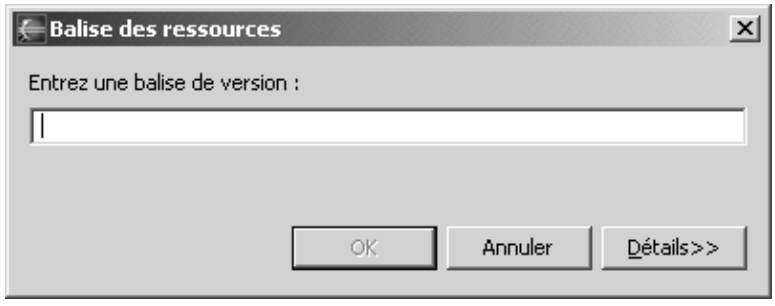


12.4. La gestion des versions d'un projet

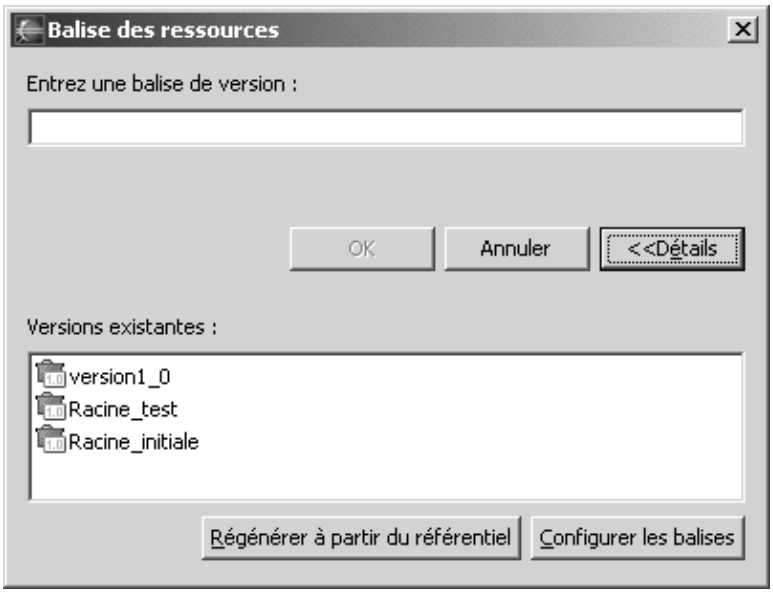
12.4.1. La création d'une version d'un projet

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. Les autres perspectives

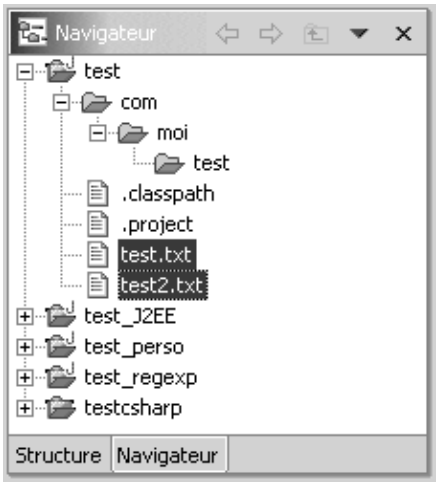
13.1. La perspective Ressource

Par défaut, cette perspective contient les fenêtres suivantes :

- la vue "Navigateur" qui affiche les ressources (arborescence des fichiers) du workspace
- 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

13.1.1. La vue "Navigateur"

Dans le workspace, 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.



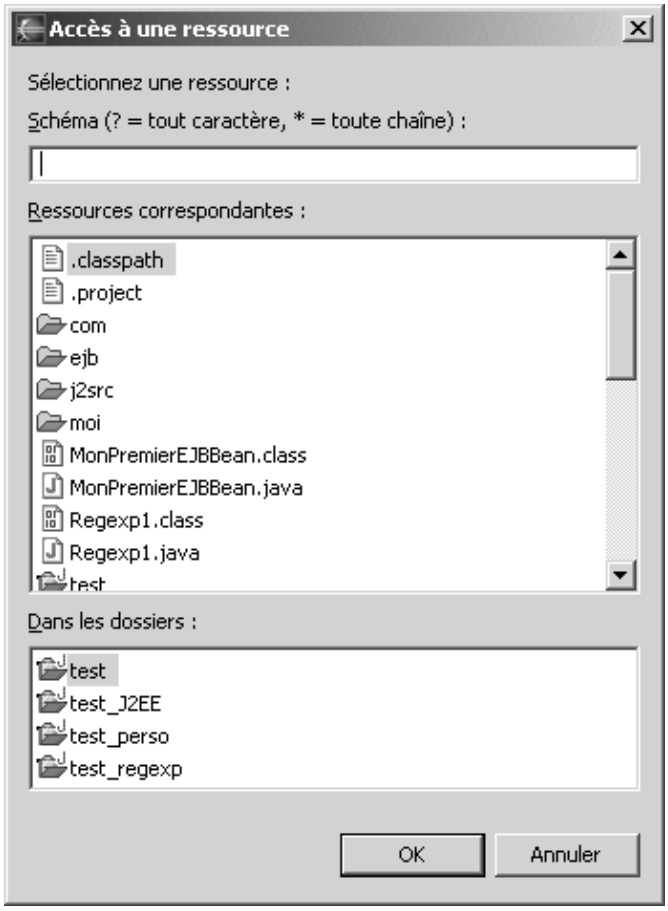
A partir du Navigateur, il est possible d'ouvrir le fichier sélectionné dans un éditeur :

- 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 navigator 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 "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 le workspace. 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.

 Ces 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.

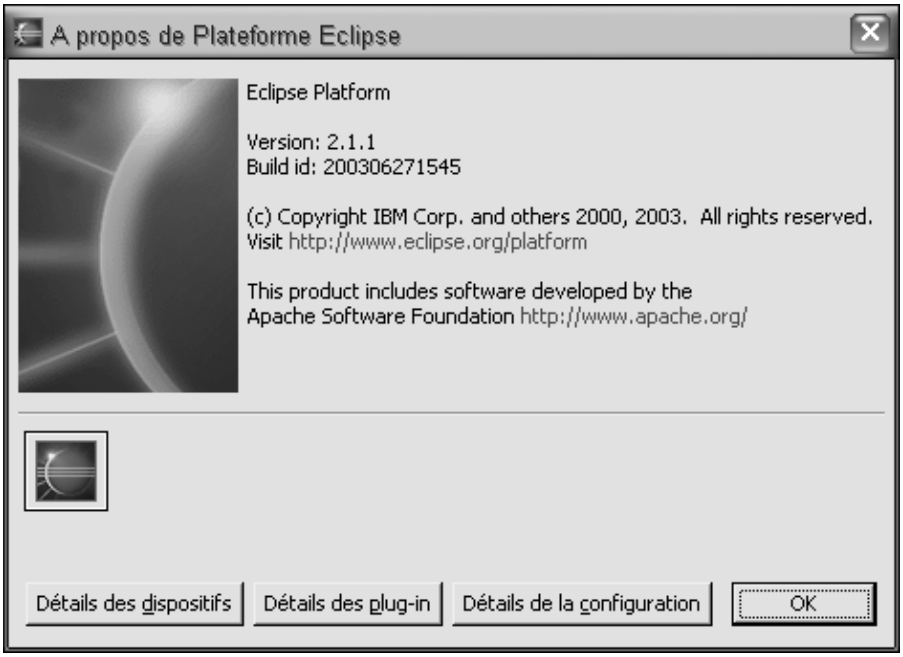
14. Les plug−ins

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 dézipper l'archive qui contient le plug−in pour que le contenu soit insérer dans le répertoire plugins où est installé Eclipse.

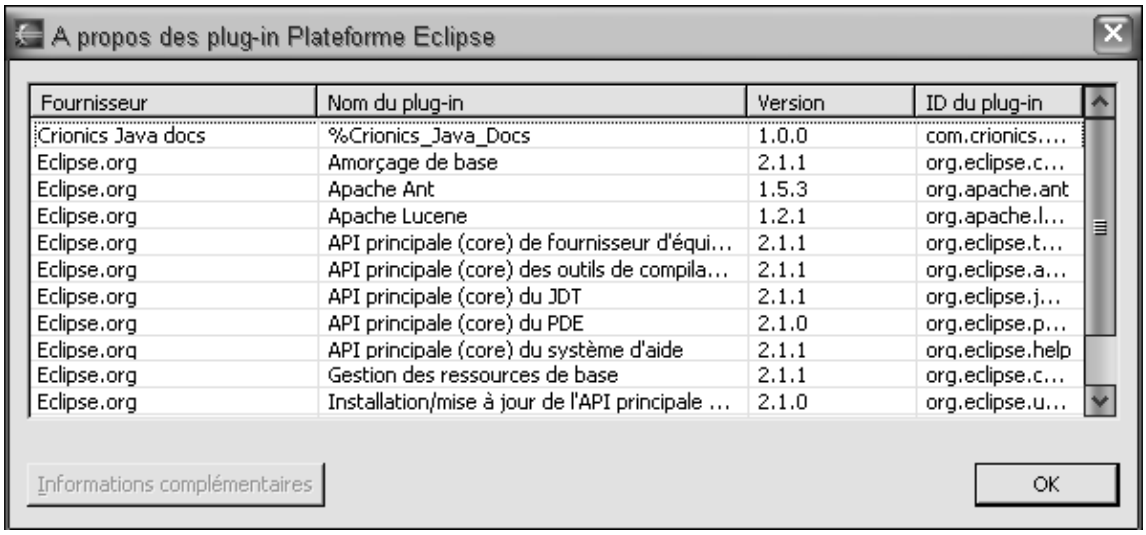
14.1. Informations sur les plug ins installés

menu aide / A propos de plateforme Eclipse

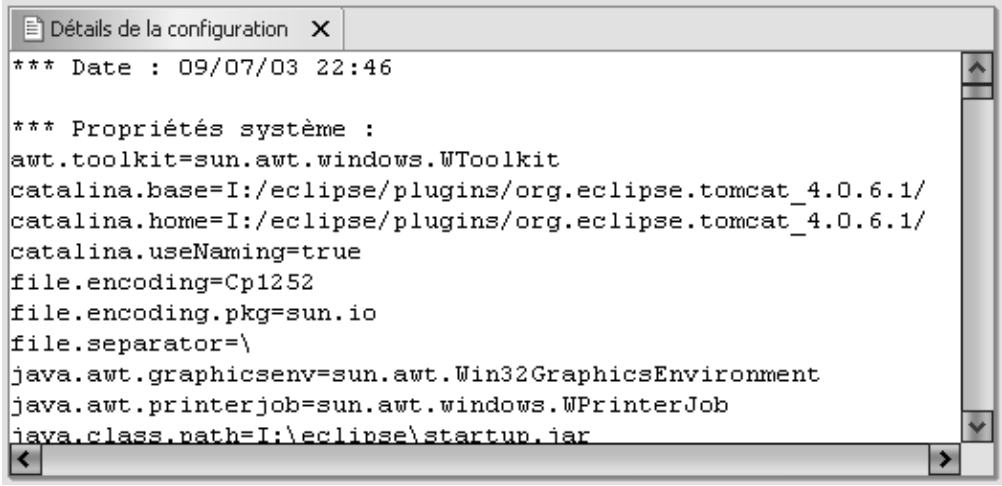


Cette boîte de dialogue affiche des informations générales sur Eclipse et permet de sélectionner le type d'informations supplémentaires souhaitées.

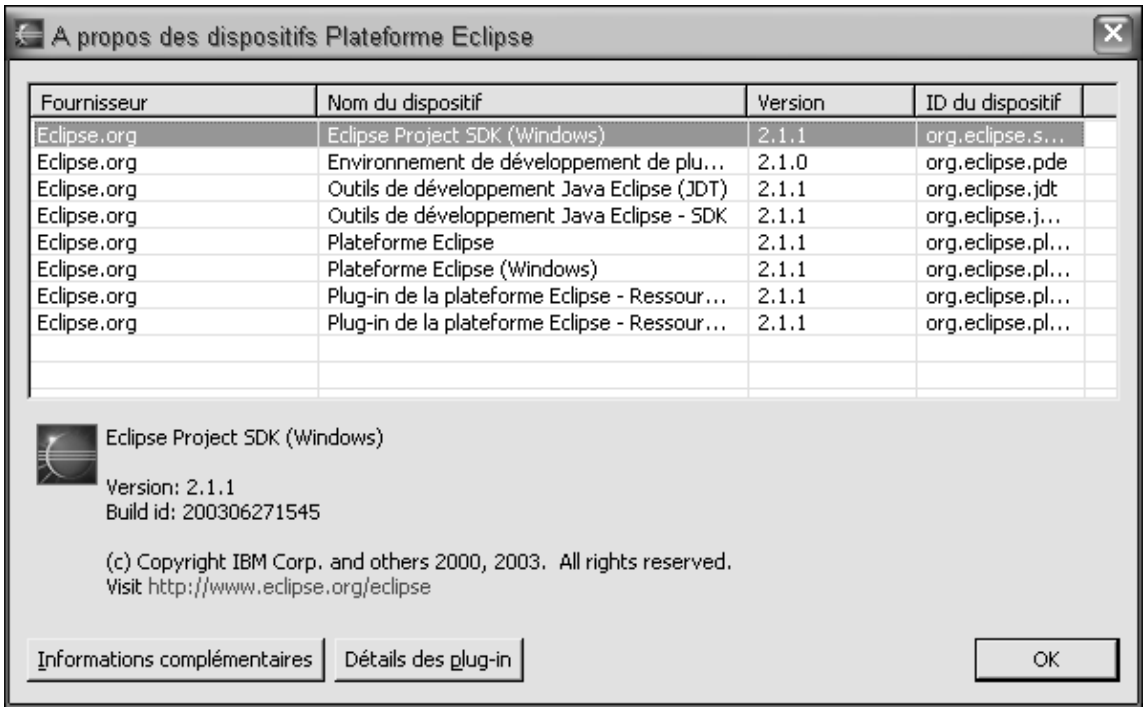
Cliquer sur Détails des plug in



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. Un exemple avec le module JadClipse sous Eclipse 1.0

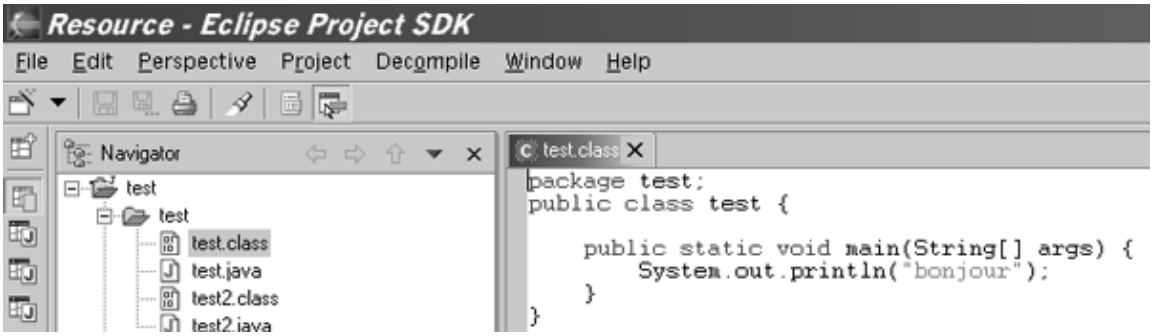
Jadclipse est un module qui permet d'utiliser le décompilateur jad dans Eclipse.

Il suffit de télécharger le fichier zip contenant le module. Il faut aussi avoir ou télécharger l'outil jad.

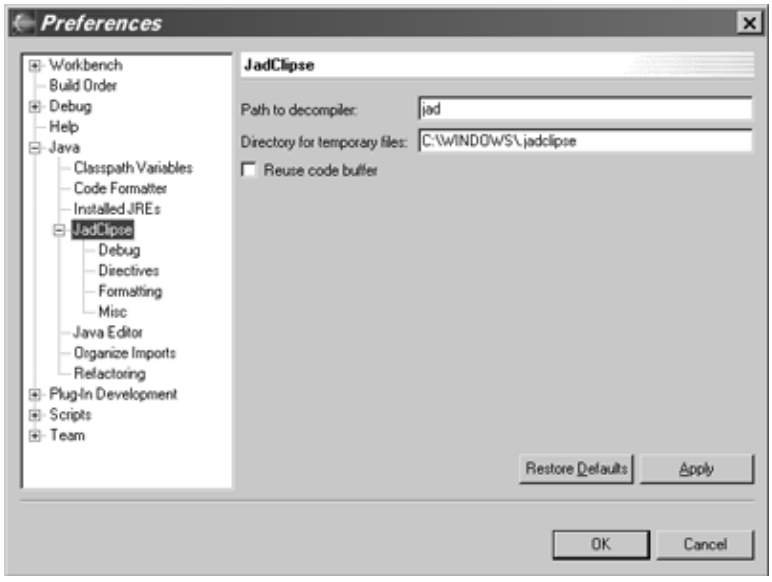
Pour installer le plug−ins, il suffit simplement de dezipper le fichier dans le répertoire plugins du répertoire principale d'Eclipse.

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 decompilé et le code source est affiché dans l'éditeur.



Le module est parfaitement intégré dans Eclipse : les options du module sont modifiables dans l'arborescence "Java / Jadclipse" des préférences (menu "Window / Preferences").



14.3. 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.

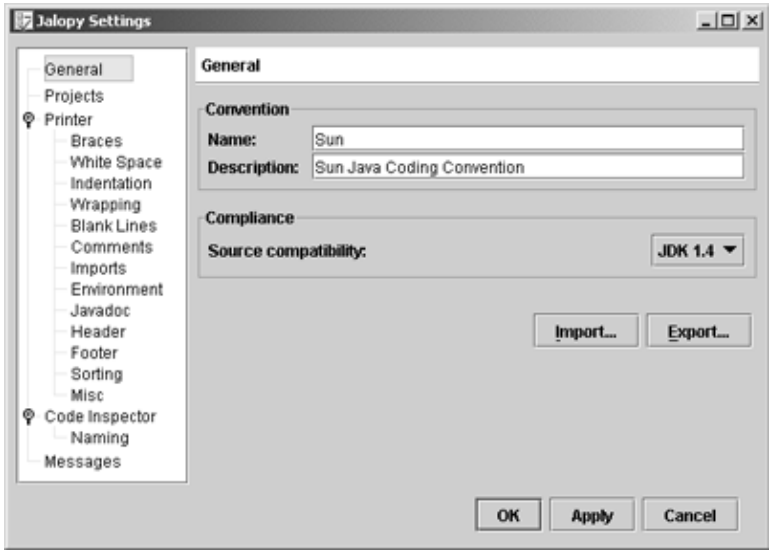
Il permet notamment :

- d'indenter le code
- de générer des modèles de commentaires javadoc dynamique en fonction des l'éléments du code à documenter (par exemple générer un tag @param pour chaque paramètre d'un 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 in 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 dézipper 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 "Fenetre" 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 répertoire et de sélectionner l'option "Format" du menu contextuel.

15. La gestion de la plate−forme

Eclipse 2.0 propose une fonctionnalité qui permet d'installer ou de mettre à jour des plug−ins via le web.

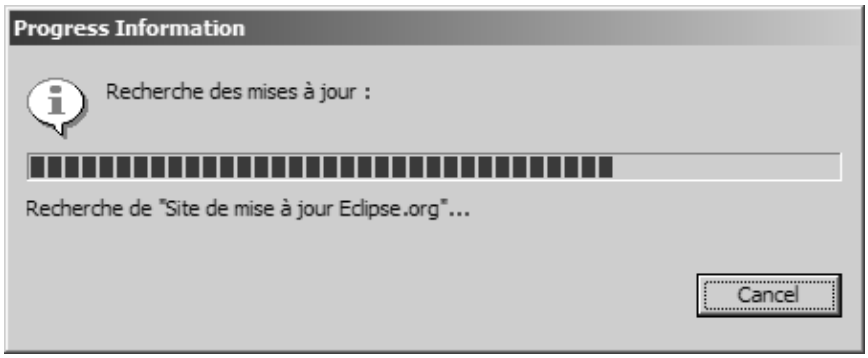
La perspective « Installation / Mise à jour » permet la gestion de ces mises à jour. Pour l'afficher, il suffit de sélectionner cette perspective ou de sélectionner l'option « Mise à jour des logiciels / Gestionnaire des mises à jour » du menu "Aide".

La vue "Mises à jour des dispositifs" recence la liste des sources d'où peuvent être téléchargées les mises à jour.

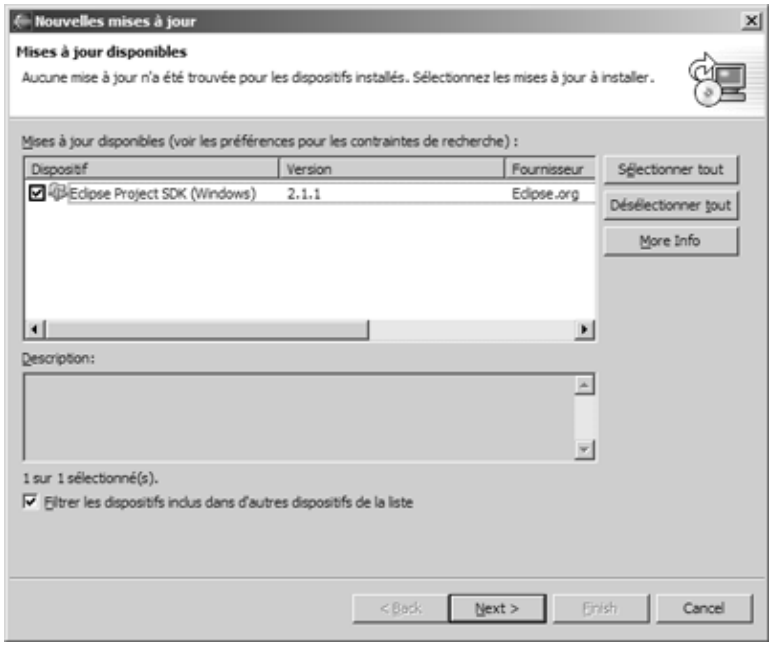
15.1. Recherche et installation des mises à jour

Il faut utiliser l'option "Mise à jour des logiciels" du menu "Aide".

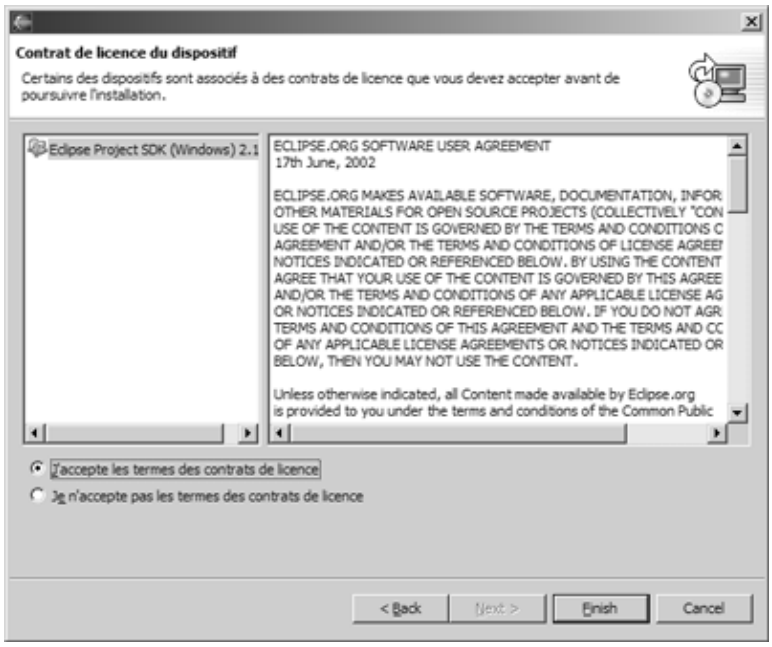
Un assistant recherche les mises à jour pour les mises à jour permis celles enregistrées dans Eclipse.



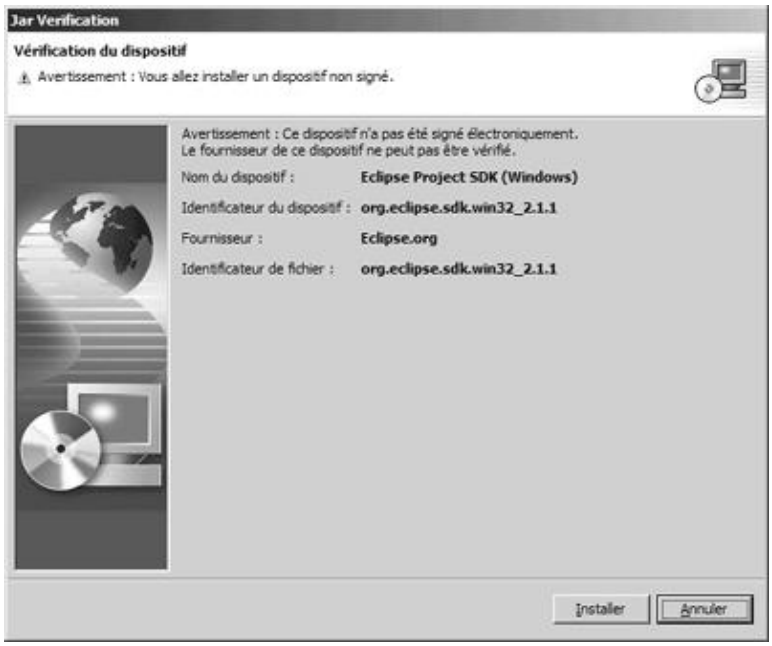
L'exemple ci dessous illustre la mise à jour d'une version 2.1 d'Eclipse avec la version 2.1.1.



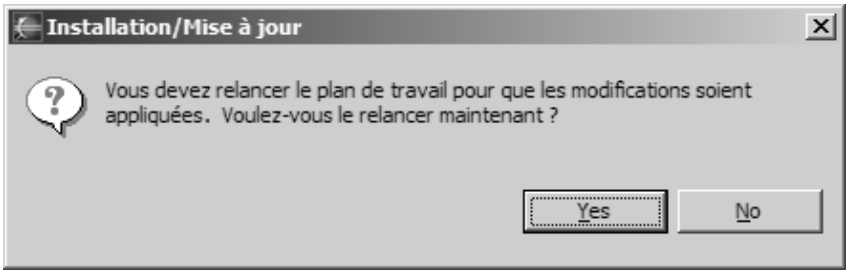
Cliquez sur le bouton "Next"



Cliquer sur le bouton "Finish"



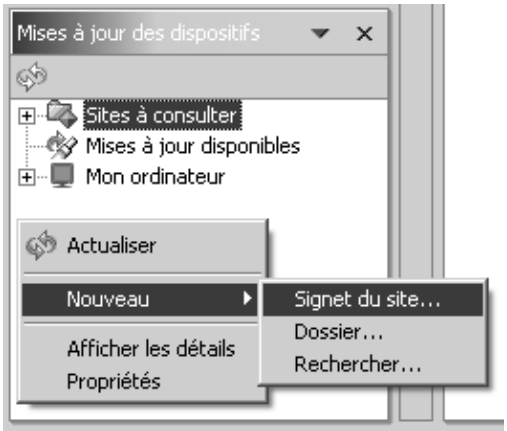
Cliquez sur le bouton "Installer". Une fois l'installation terminée, il faut relancer Eclipse.



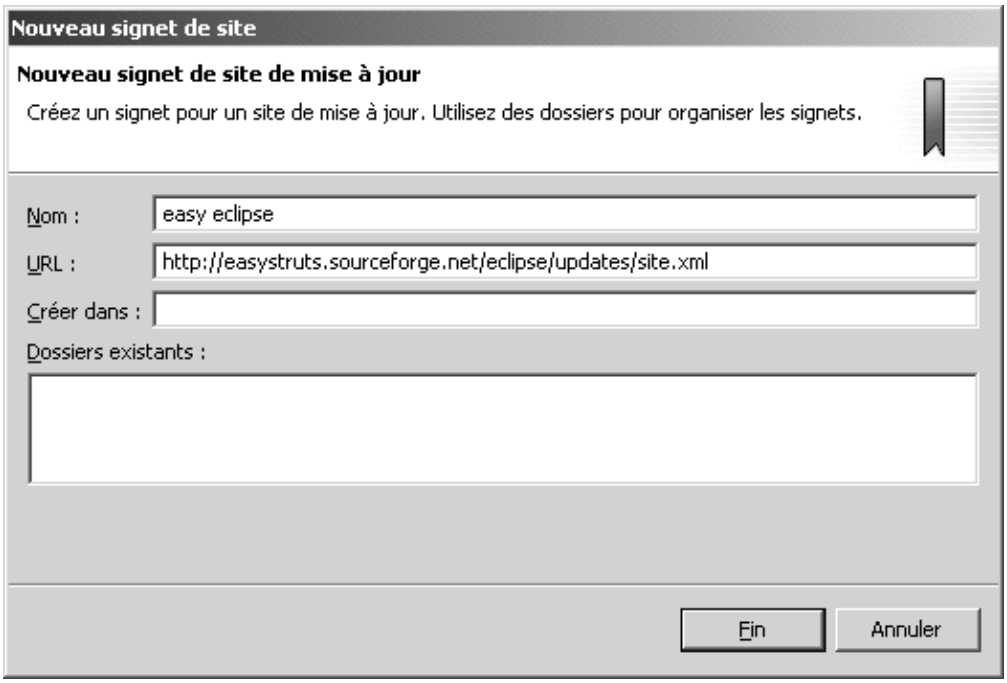
15.2. Installation d'un nouveau plug in

L'exemple ci dessous met en oeuvre l'installation du plug in Easy Struts. Pour d'autres plug in permetant leur mise à jour par le réseau, la procédure est similaire.

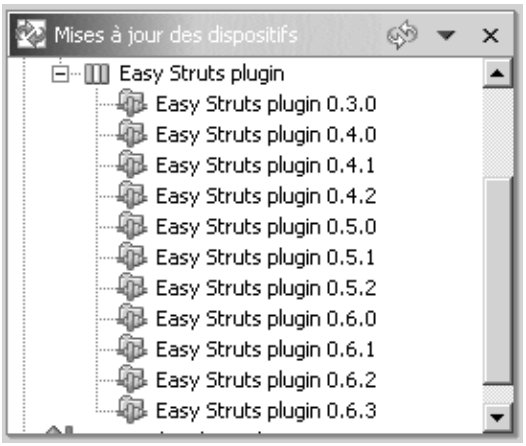
Pour ajouter une nouvelle source, il suffit de sélectionner l'option « Nouveau / Signet du site ... » du menu contextuel de la vue « Mises à jour des dispositifs ».



Une boîte de dialogue permet de saisir un nom et l'URL du signet.

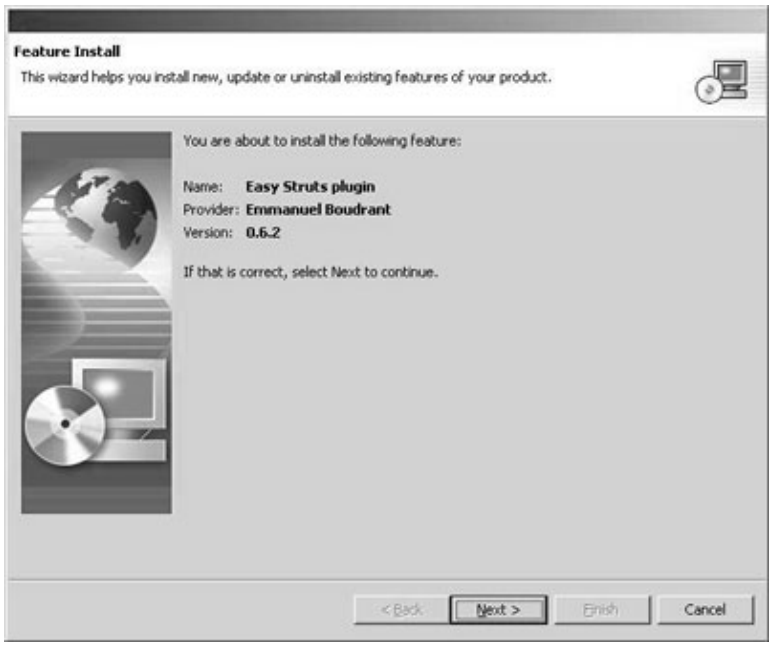


Il faut saisir le nom et l'url du fichier site.xml désiré puis cliquer sur le bouton "Fin".

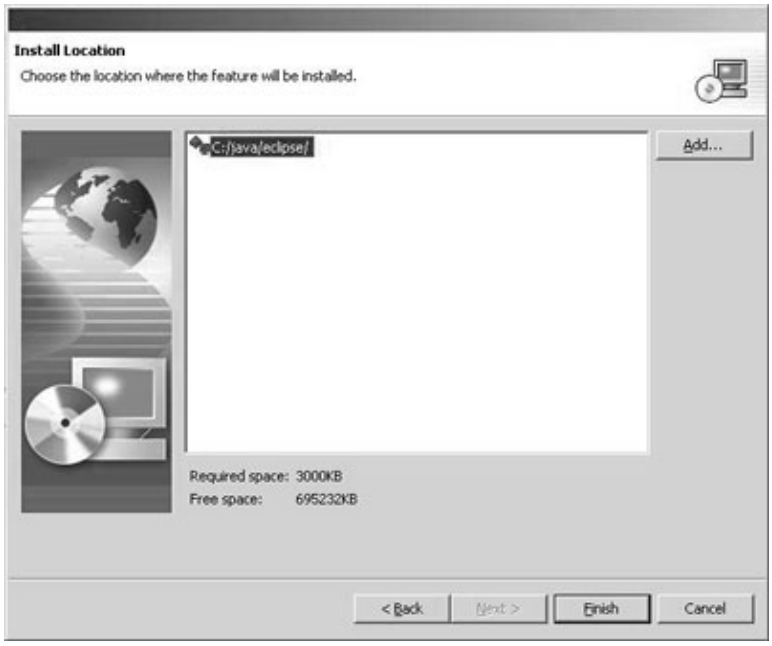


L'application va lire le fichier et affiche les plugins disponibles dans l'arbrescence. Il suffit de sélectionner la version désirée. La vue "Aperçu" affiche le details du plug−in sélectionné.

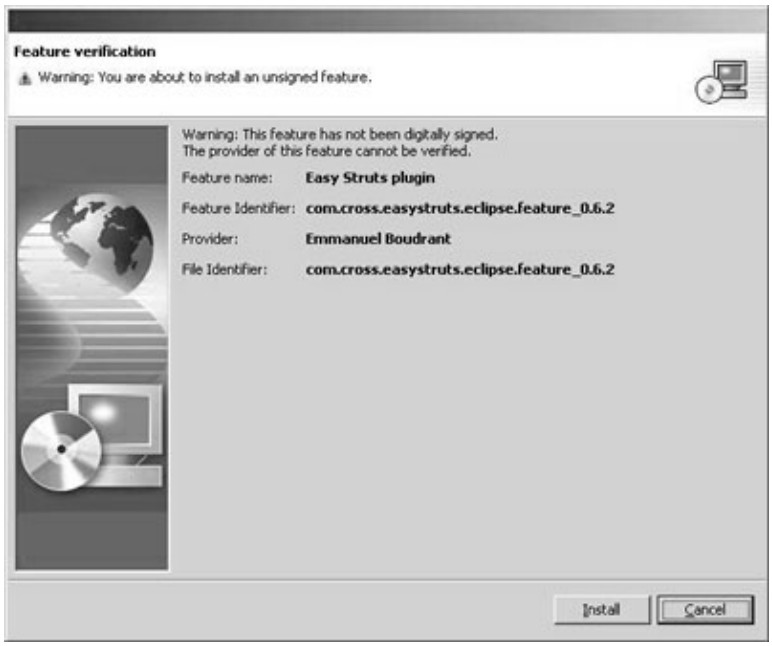
Pour installer la version sélectionnée, cliquer sur le bouton "Install" dans la vue « Aperçu ».



Cliquer sur "Next". Lire et accepter la licence, puis cliquer sur le bouton “Next”

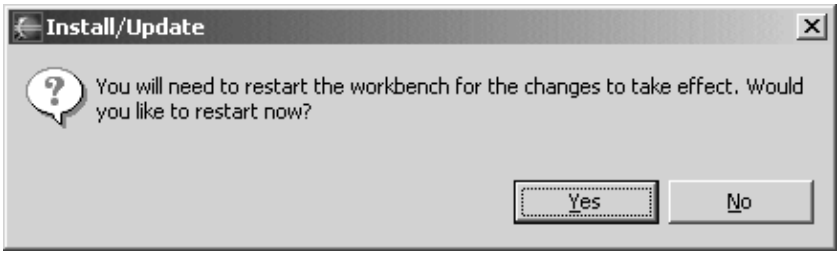


Sélectionner la cible d'installation et cliquer sur le bouton «Finish».

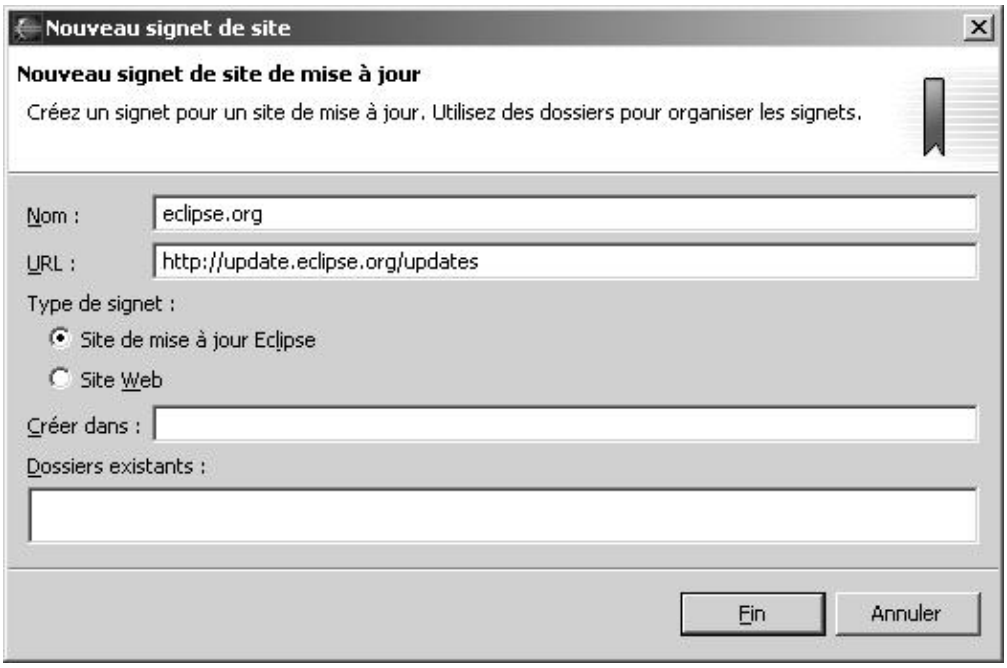


Confirmer l'installation, car le package n'est pas signé, en cliquant sur le bouton « Install »

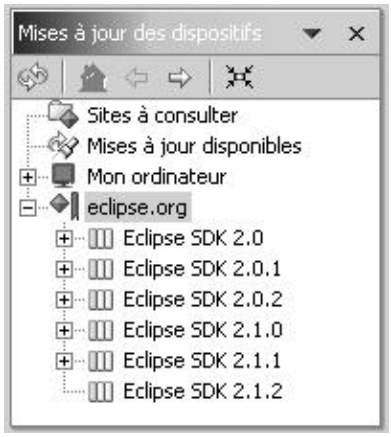
Le plug est téléchargé et installé. L'assistant propose de redémarrer le workbench pour que les modifications soient prises en compte. Cliquer sur le bouton "Yes".



Il peut être intéressant d'ajouter le site officielle d'Eclipse pour ajouter des plug−ins qui ne sont pas fournis en standard.



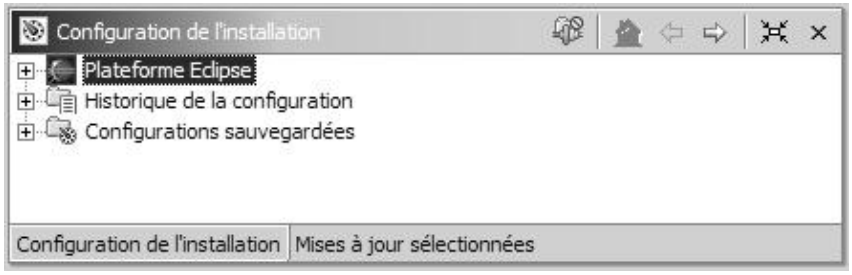
Un fois le nouveau signet configuré, il suffit de naviguer dans l'arborescence en fonction de la version de la plate−forme pour pouvoir installer un ou plusieurs plug−ins parmis ceux proposés.



15.3. Sauvegarde et restauration d'une configuration

A chaque mise à jour, Eclipse enregistre la configuration.

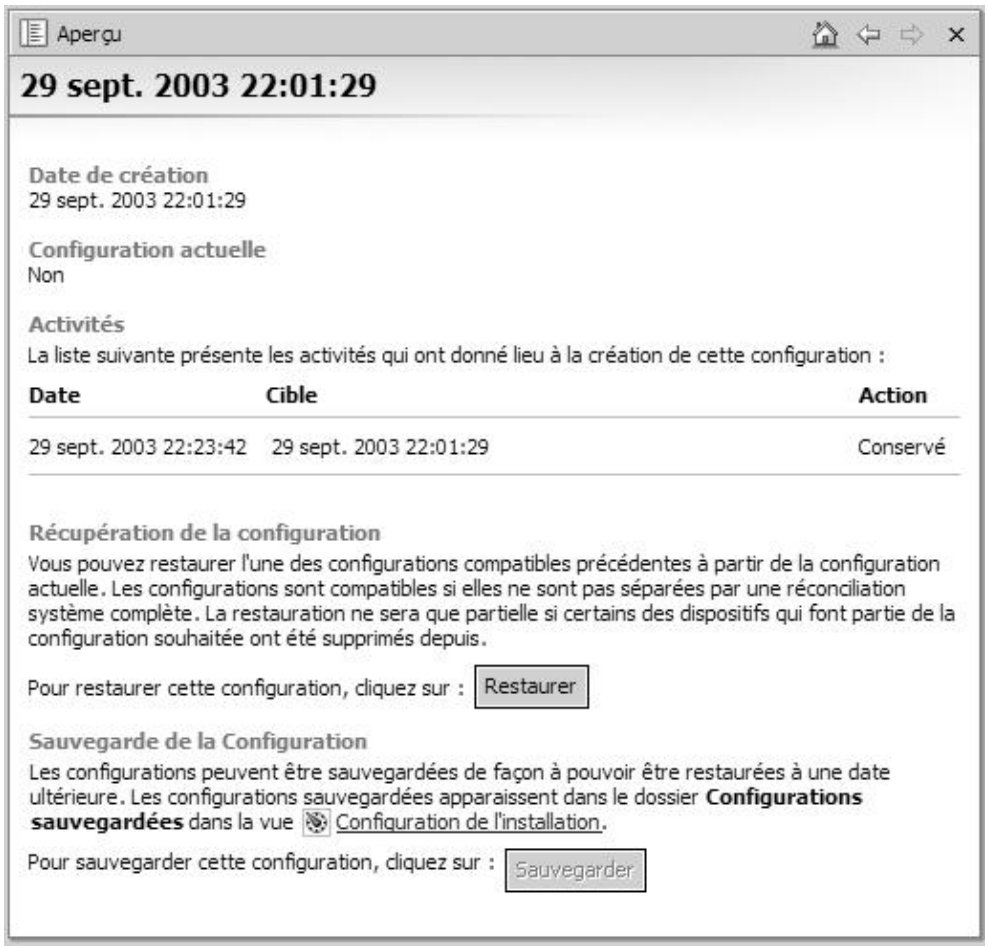
Il est aussi possible de demander la sauvegarde explicite de la configuration : il faut, dans la perspective « Installation / Mise à jour » sélectionner « Plate−forme Eclipse » dans la vue « Configuration de l'installation »



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 » le 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 ».

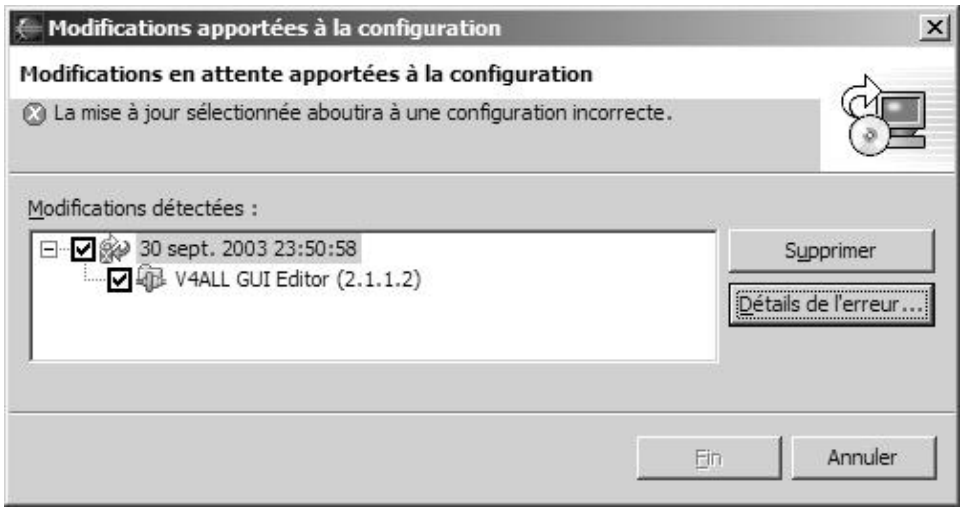


Cette opération nécessite un redémarrage de l'application.

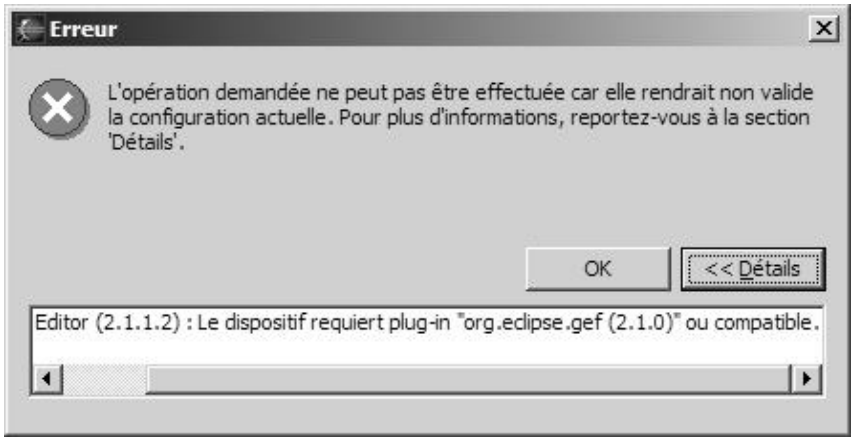
15.4. Résolution des problèmes de dépendances

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.

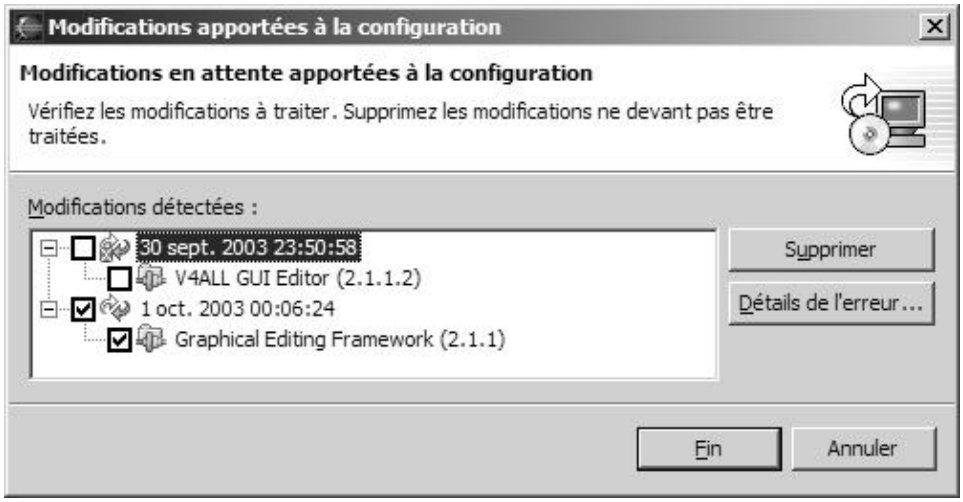


Pour avoir plus d'informations, il suffit de sélectionner la mise à jour et de cliquer sur le bouton « Détails de l'erreur ».

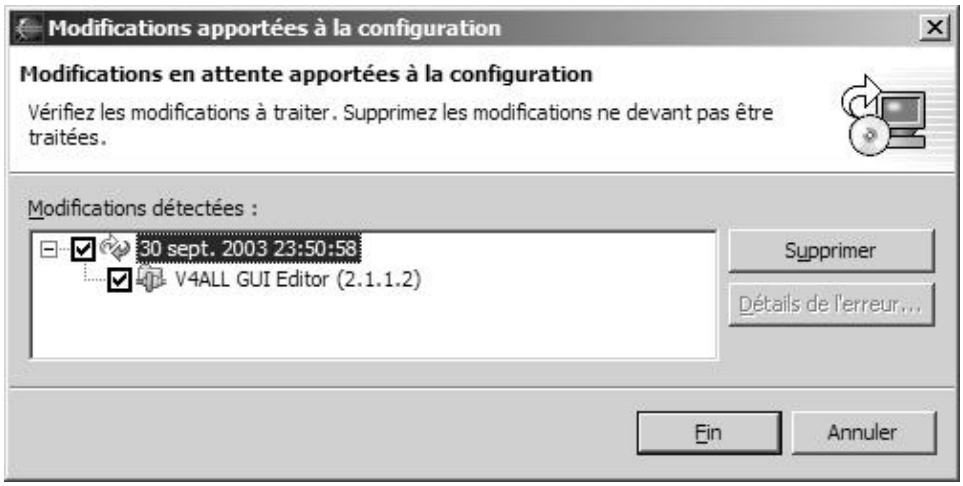


Dans l'exemple ci−dessus, il manque un plug in ou la version installée n'est pas correcte.

Pour résoudre le problème, il suffit de quitter Eclipse, d'installer le plug in (en le dézippant 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 ».

16. Le développement sans Java

Eclipse est un outils 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 in spécifique. Il existe déjà plusieurs plug in plus ou moins avancés pour réaliser des développements en C/C++ et Cobol développé par Eclipse mais aussi C# ou PHP développés par des tiers.

16.1. 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 necessaires 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 exécutable natif 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.

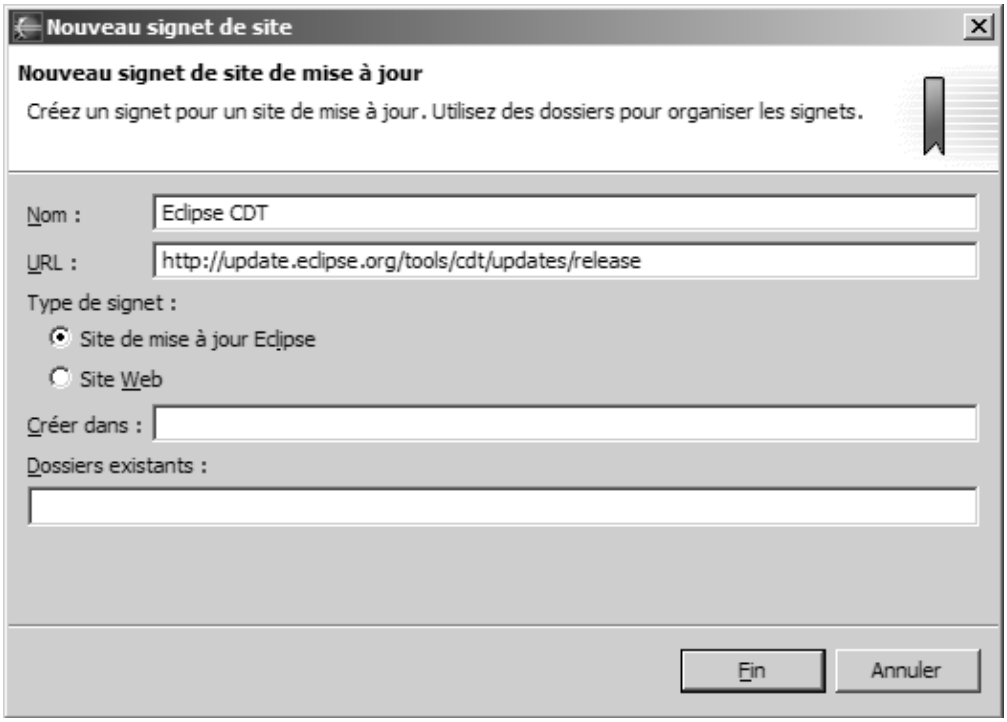
Bien qu'écrit en Java, le CDT est dépendant de la plate−forme sur laquelle il s'exécute.

16.1.1. Installation du CDT

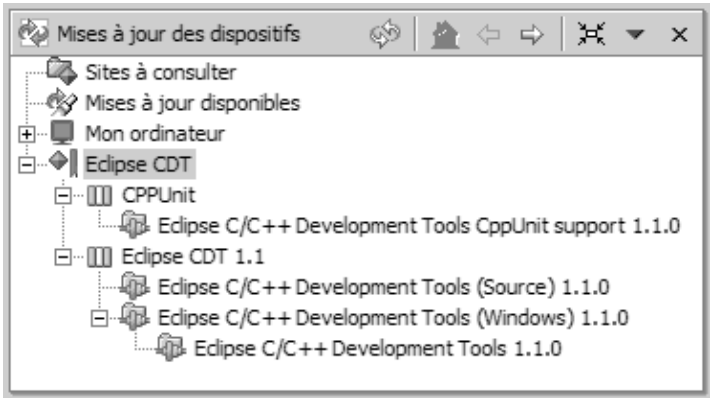
Pour installer le CDT, il faut utiliser le mécanisme de mise à jour en utilisant l'option "Mise à jour des logiciels / Gestionnaire des mises à jour" du menu "Aide".

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".



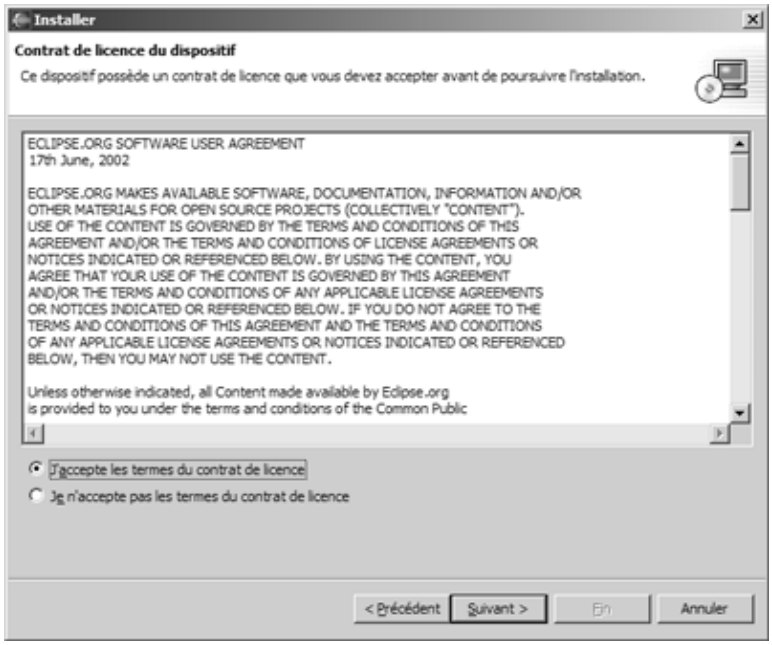
Cliquer sur le bouton "Fin" pour permettre à Eclipse de rechercher les mises à jour. La vue "Mise à jour des dispositifs" affiche celles disponibles.



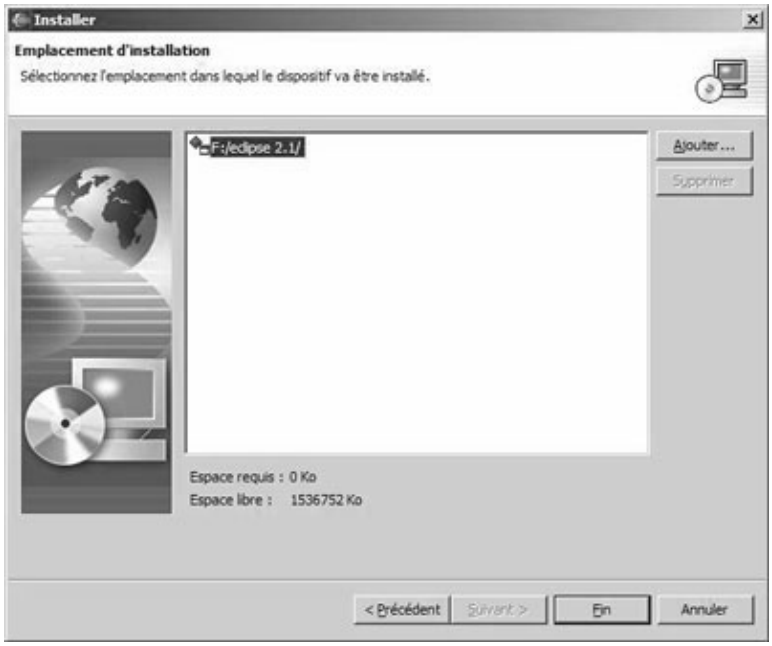
Sélectionnez la mise à jour « Eclipse C/C++ Development Tools (Windows). »

La vue "Aperçu" affiche des informations sur le plug−in. Cliquez sur le bouton « Installer Maintenant »

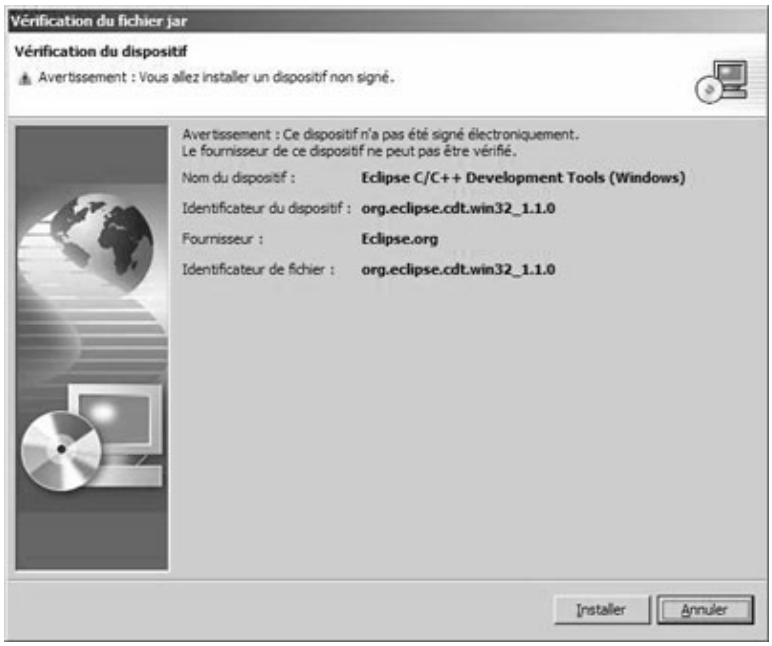
Cliquez sur le bouton « Suivant »



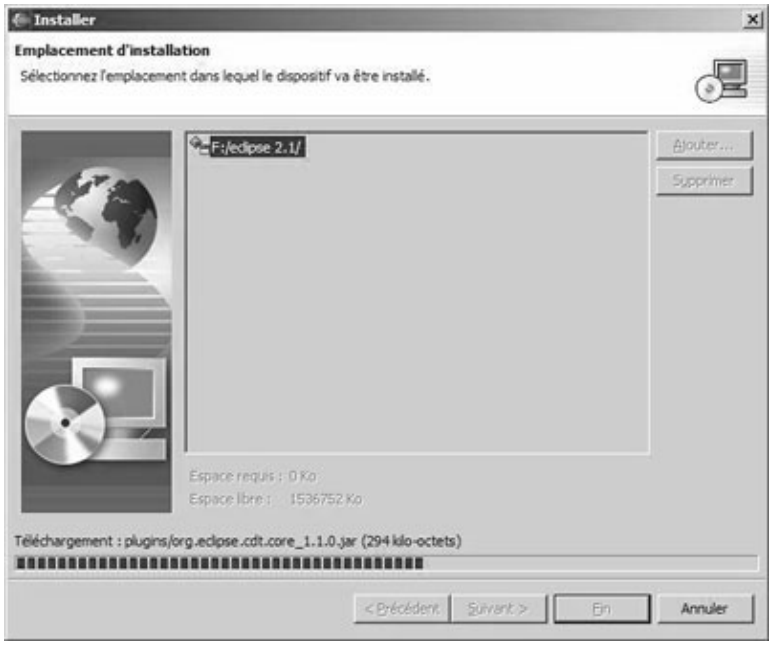
Lisez et si vous acceptez les termes du contrat, cliquez sur le bouton « Suivant »



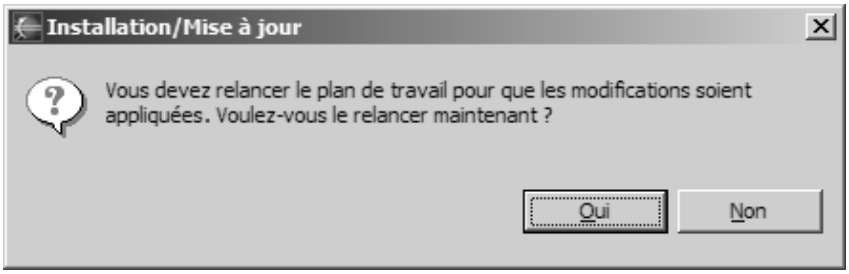
Cliquez sur le bouton « Fin »



Cliquer sur le bouton « Installer ».



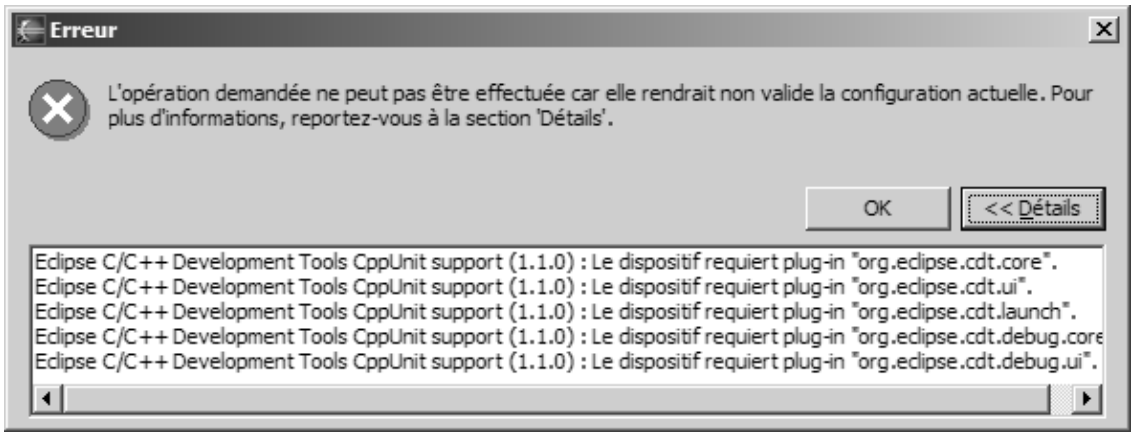
L'installation nécessite un redémarrage de l'application.



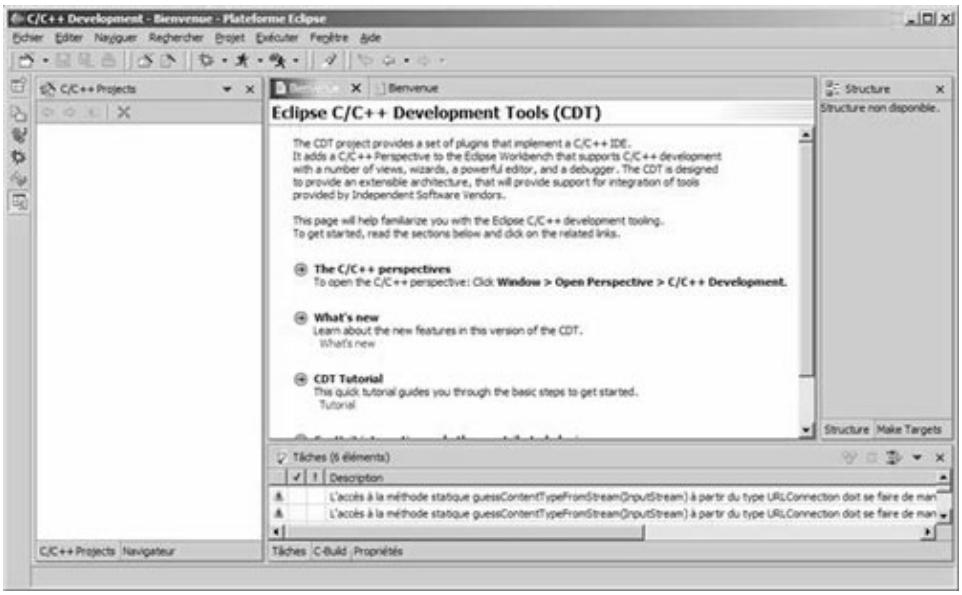
Cliquez sur le bouton « Oui ».

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.

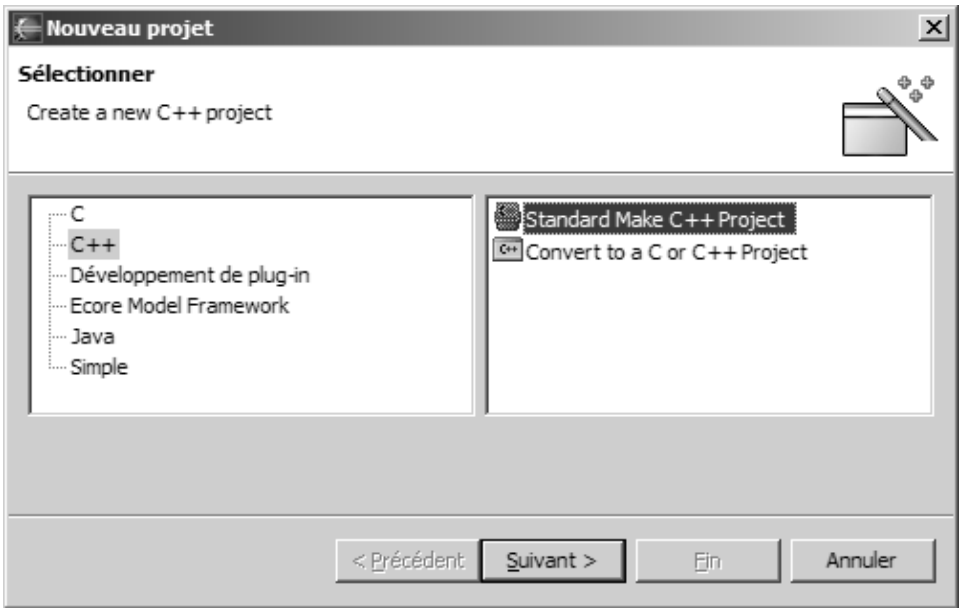


Une fois l'installation terminée, l'éditeur de la prespective ressource affiche des informations sur le CDT.

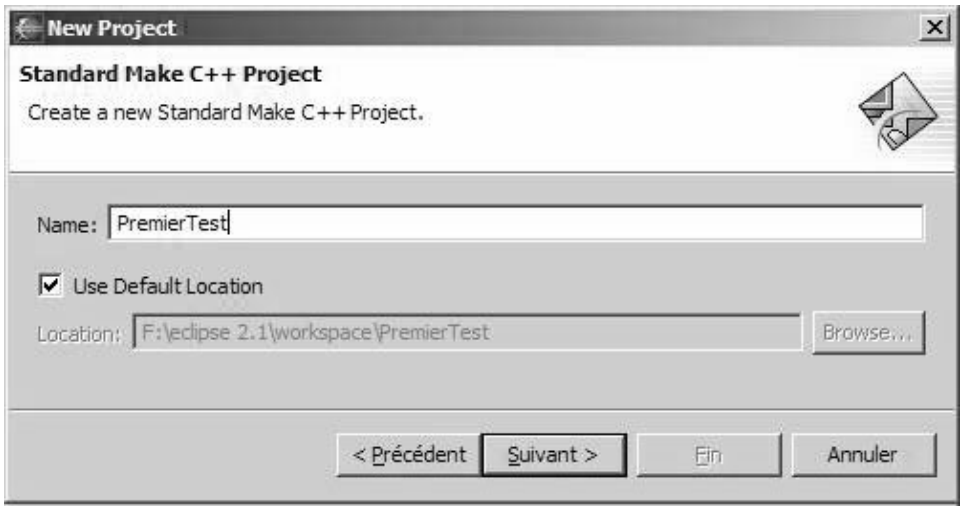


16.1.2. Création d'un premier projet

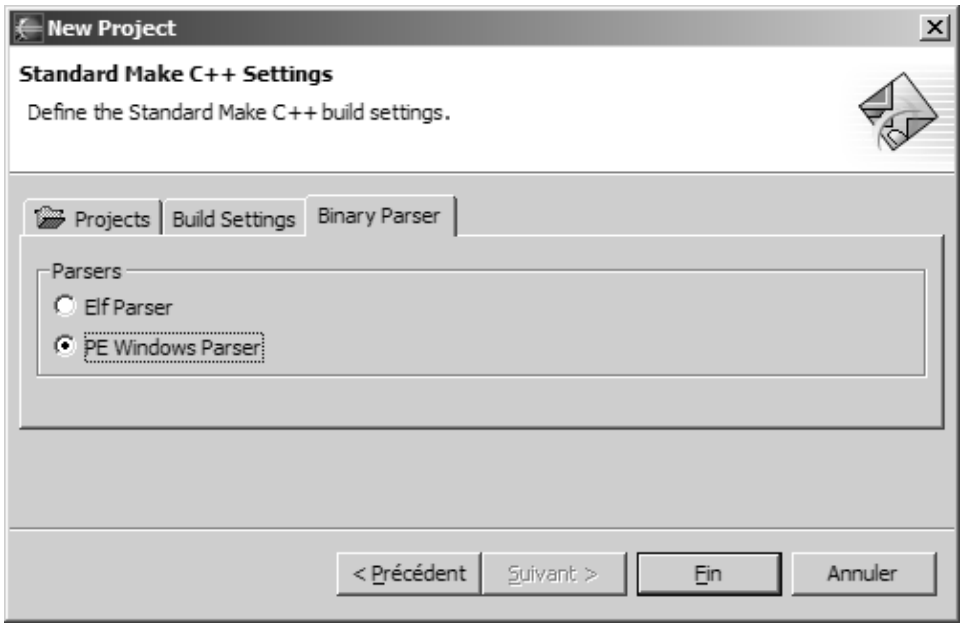
Pour pouvoir utiliser le CDT, il faut tout d'abord créer un nouveau projet.



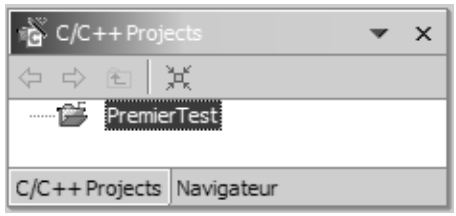
Pour un projet en C++, sélectionnez "C++" dans la liste de gauche puis "Standard Make C++ Project" dans la liste de droite et enfin cliquez sur le bouton "Suivant"



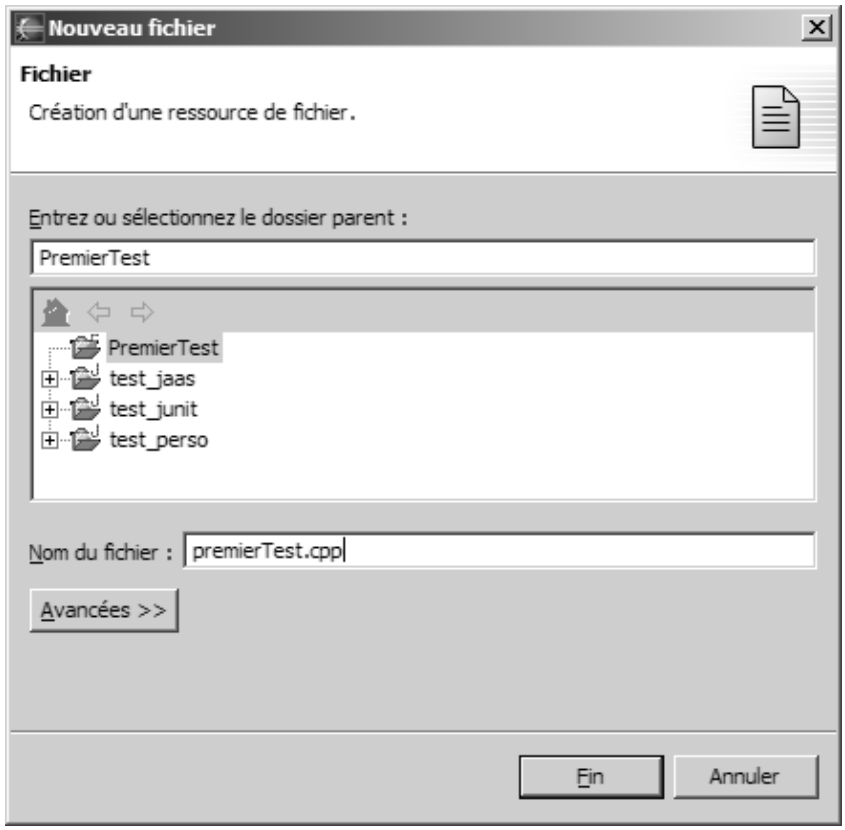
Saisissez le nom du projet et cliquer sur « Suivant ».



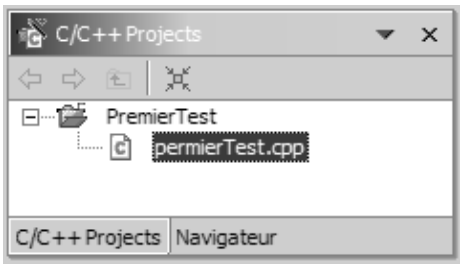
Sous Windows, dans l'onglet « Binary Parser », cliquez sur « PE Windows Parser » puis sur le bouton « Fin ».



Il faut ensuite créez un nouveau fichier dans le projet.



Saisissez le nom du fichier et cliquez sur le bouton « Fin ».

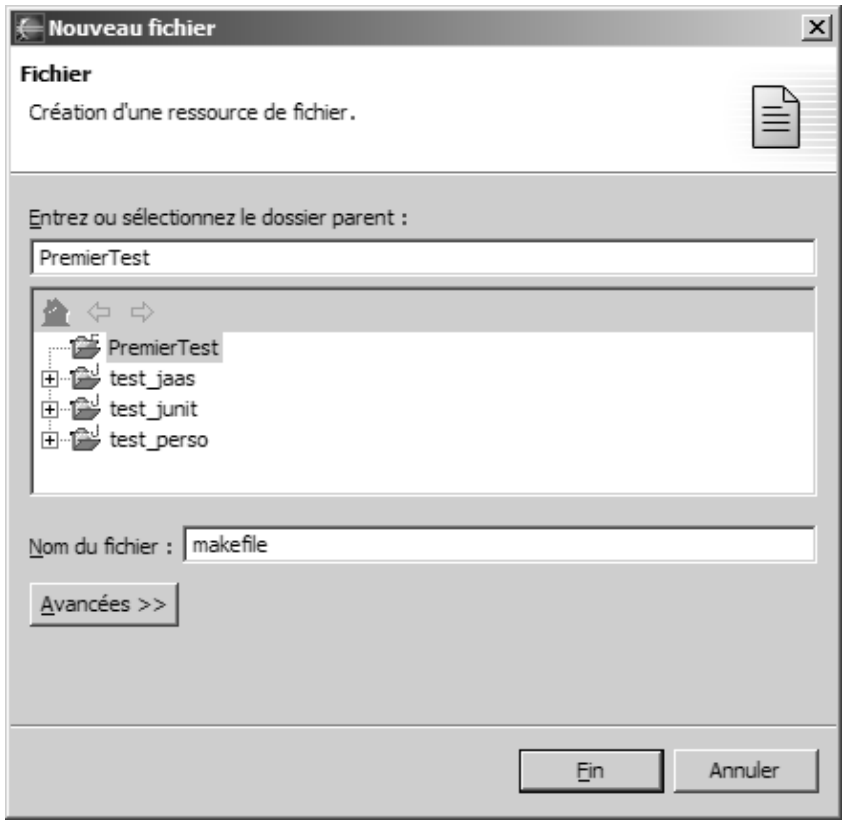


Il suffit de saisir le code de l'application dans l'éditeur :

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'outils make.



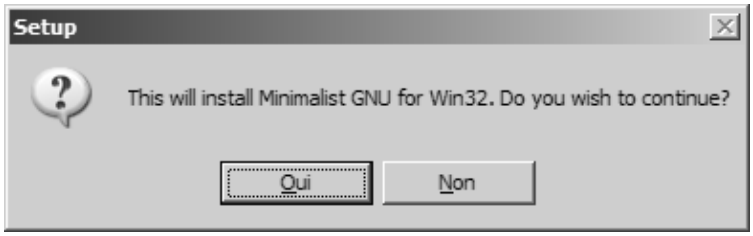
Cliquez sur le bouton « Suivant » puis sur le bouton « Fin ».

16.1.3. Installation de MinGW

Pour pouvoir compiler des fichiers sources avec le CDT, il est nécessaire de disposer 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 outils tiers. Cette section décrit la procédure d'installation avec l'outils 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/

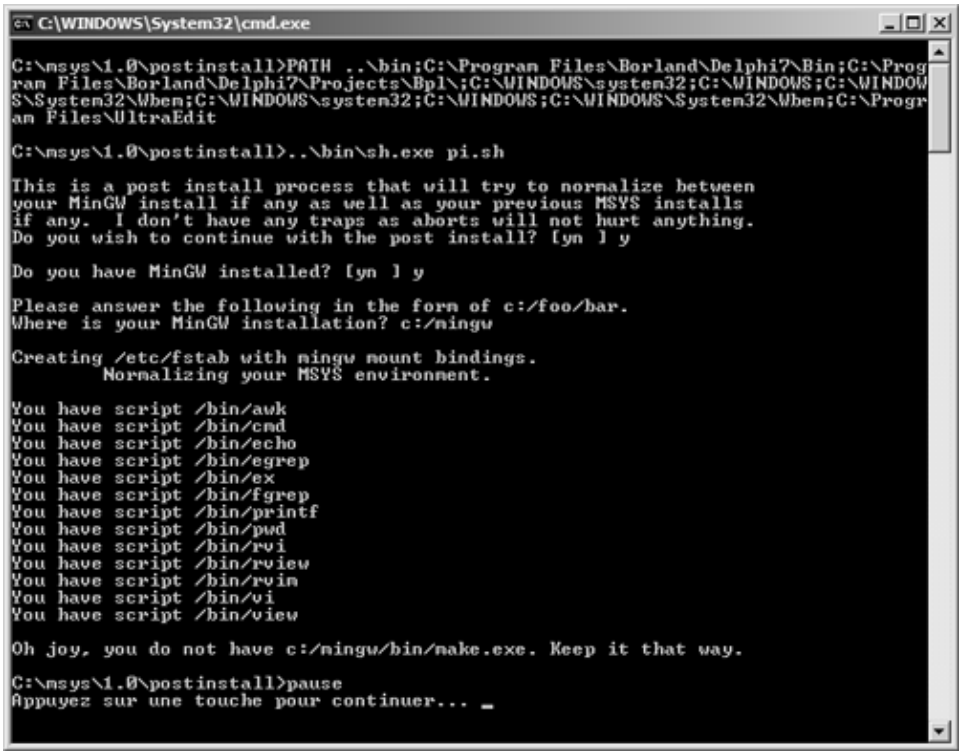
Lancez l'exécution de MinGW en premier.



L'assistant d'installation se compose de plusieurs pages :

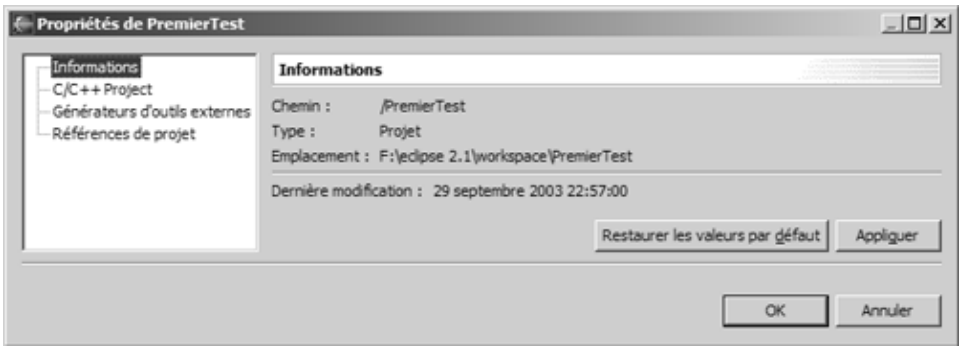
- Sur la page d'accueil, cliquez sur le bouton "Next".
- La page "License Agreement" apparait : lisez la licence et cliquez sur le bouton "Yes" pour accepter les termes de la licence et poursuivre l'installation.
- La page "Information" apaprait : lisez le texte et cliquez sur le bouton "Next".
- La page "Select destination directory" apparait : sélectionnez le répertoire ou sera installer MinGW et cliquez sur le bouton "Next".
- La page "Ready to install" apparait : cliquez sur le bouton "Install".

Lancez l'exécution de MinSys. Un script permet de configurer l'environnement.

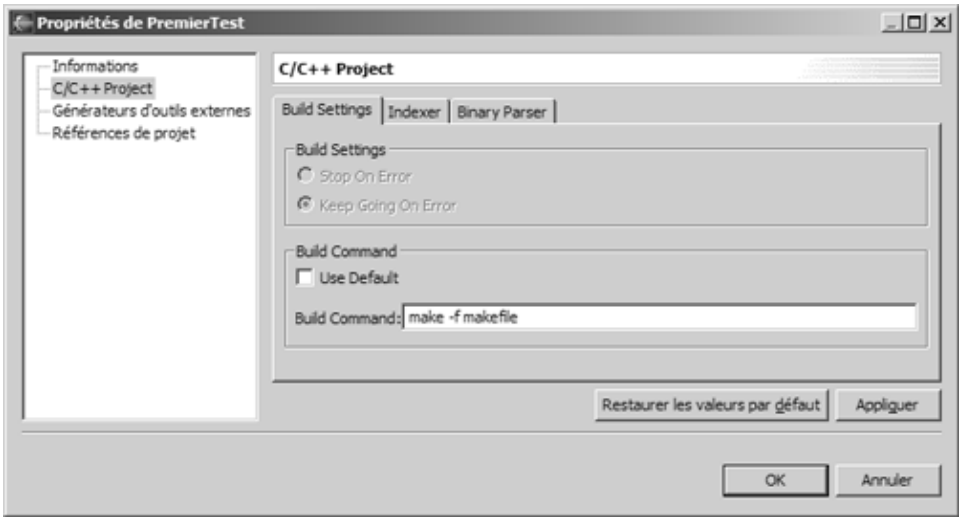


16.1.4. Première configuration et exécution

Il faut modifier les propriétés du projet pour utiliser les options de la commande make fournie avec MSYS.



Il faut décocher "use default" et saisir "make −f makefile" dans le champ "build command".



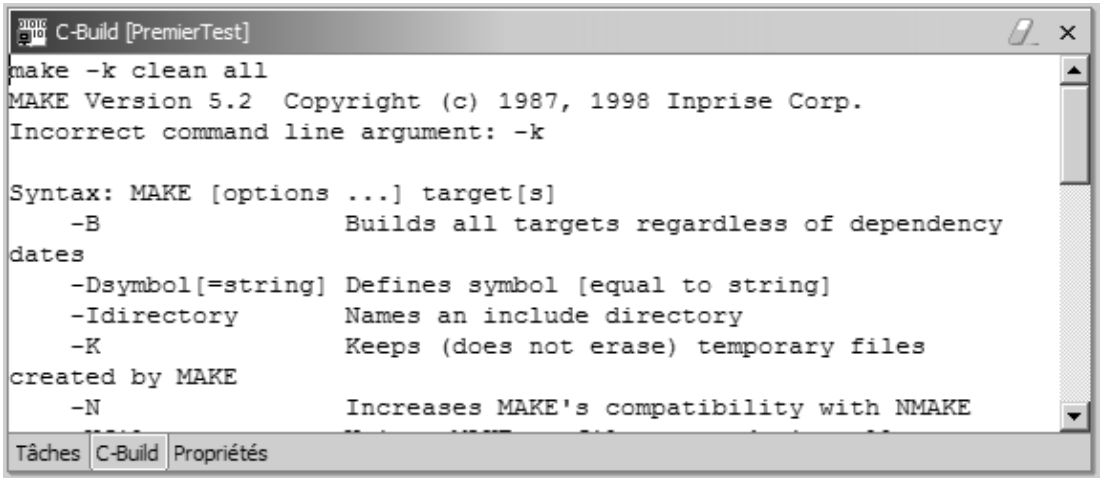
Il faut ajouter les répertoire suivants dans la variable d'environnement path : C:\msys\1.0\bin;C:\MinGW\bin;

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'outils Delphi de Borland qui ajoute dans le classpath un répertoire qui contient un programme Makefile.



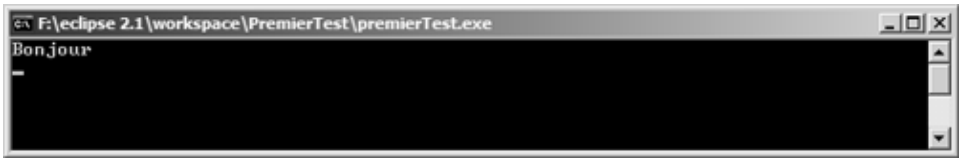
Dans ce cas, il suffit de déplacer ou de supprimer la référecence sur le répertoire C:\Program Files\Borland\Delphi7\Bin; dans la variable classpath. (il faut le remettre en cas de suppression pour une utilisation correcte de Delphi).

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.



16.1.5. 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
- ...

17. Le développement d'interfaces graphiques

La suite de ce chapitre sera développée dans une version future de ce document

17.1. Eclipse et SWT

Eclipse est développé en utilisant SWT comme API pour le développement de son inteface 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 développement des applications standalone.

17.1.1. 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 … ».

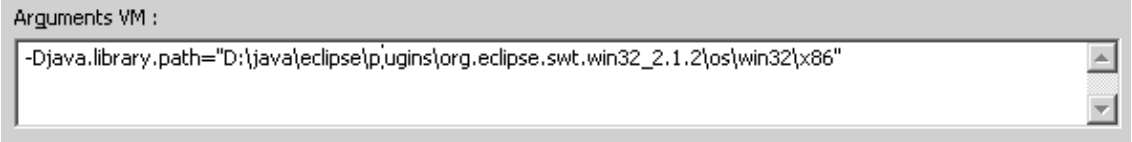
Une boîte de dialogue permet de sélectionner le fichier swt.jar. Ce fichier est dépendant du système d'exploitation sur lequel l'application va s'exécuter. Par exemple sous Windows, ce fichier se trouve dans le sous répertoire plugins\org.eclipse.swt.win32_2.1.0\ws\win32\ du répertoire d'installation d'Eclipse. Cliquez sur « Ok » pour valider les modifications et recompiler le projet.

Pour pouvoir exécuter le projet, il faut que l'application puisse accéder à une bibliothèque 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
```

17.1.2. 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.

17.2. 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.

17.2.1. Installation

V4All requière l'installation du plug in GEF (Graphical Editing Framework) en exécutant les étapes suivantes :

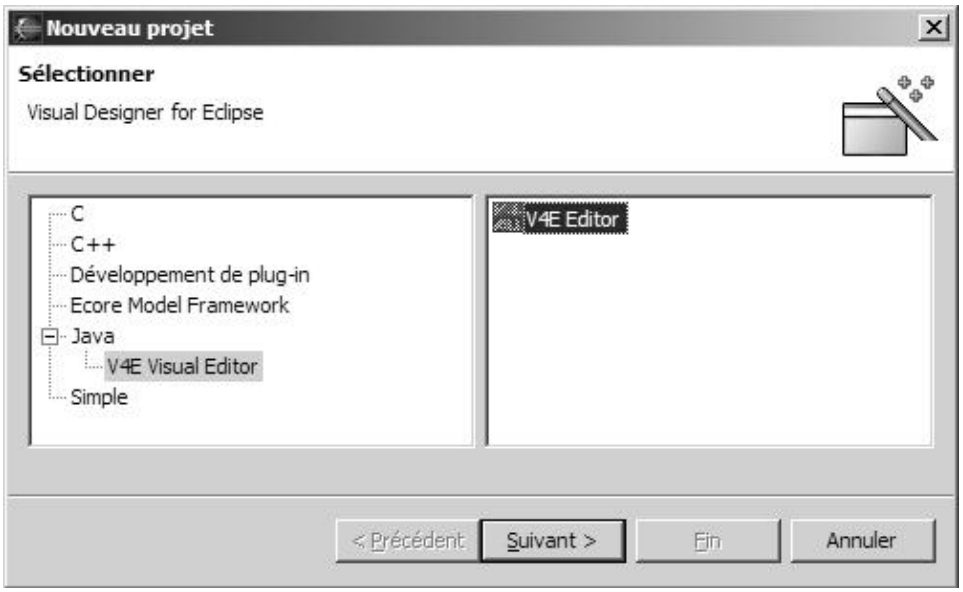
- télécharger le fichier http://download.eclipse.org/tools/gef/downloads/drops/R−2.1.1−200306180934/GEF−runtime−I20030618.zip sur le site d'Eclipse
- dézippez ce fichier dans le répertoire qui contient Eclipse.
- relancez Eclipse et cliquer sur le bouton « Fin » pour valider les modifications apportées 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 :

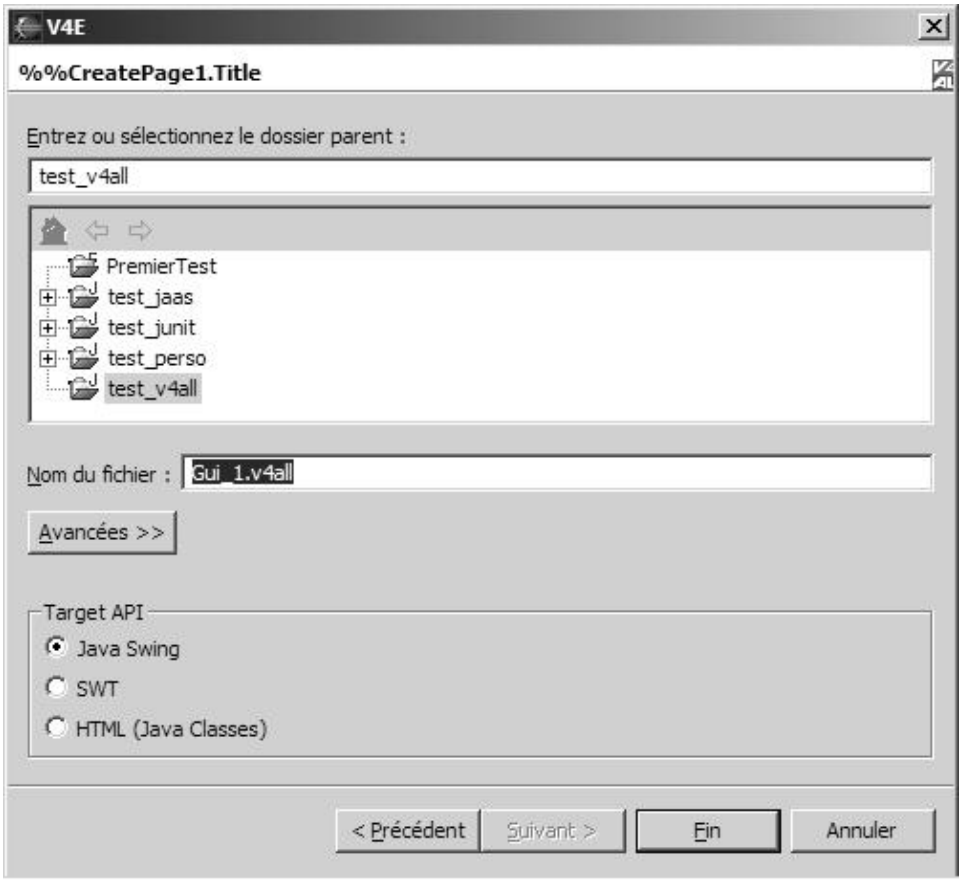
- téléchargez le fichier v4all_2_1_1_9.zip sur le site http://sourceforge.net/projects/v4all
- dézippez son contenu dans le répertoire qui contient Eclipse.
- relancez Eclipse et cliquer sur le bouton « Fin » pour valider les modifications apportées par le plug−in
- cliquez sur le bouton « Oui » lors de la demande de relance de l'application.

17.2.2. Utilisation

Pour utiliser v4all, il faut créer ou utiliser un projet Java existant, puis il faut créer un nouveau projet de type « V4E editor ».

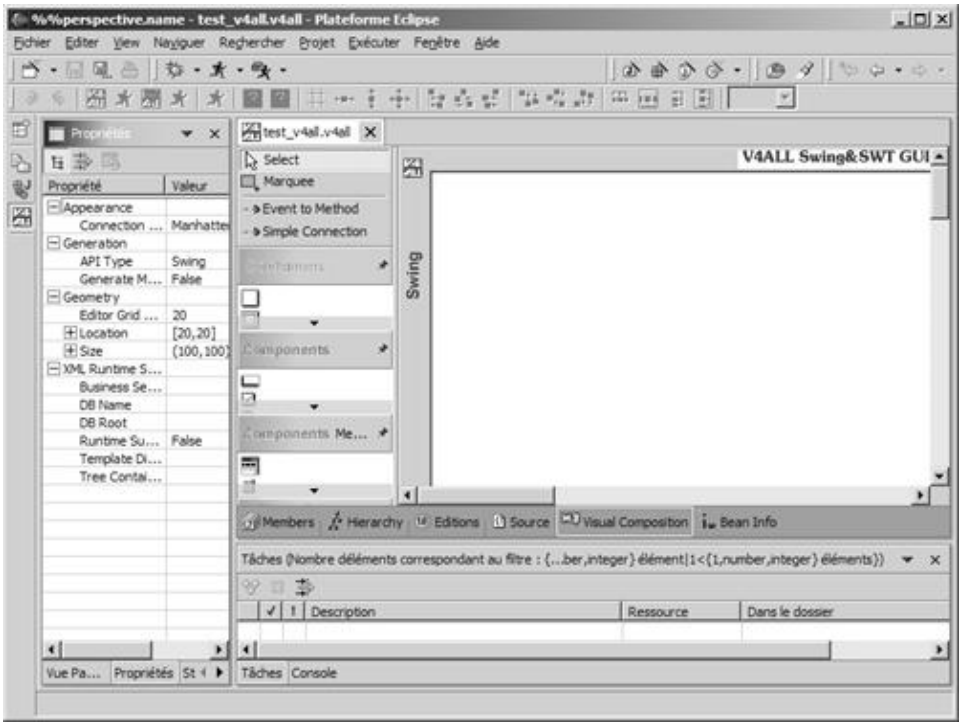


Cliquer sur le bouton « Suivant »



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.

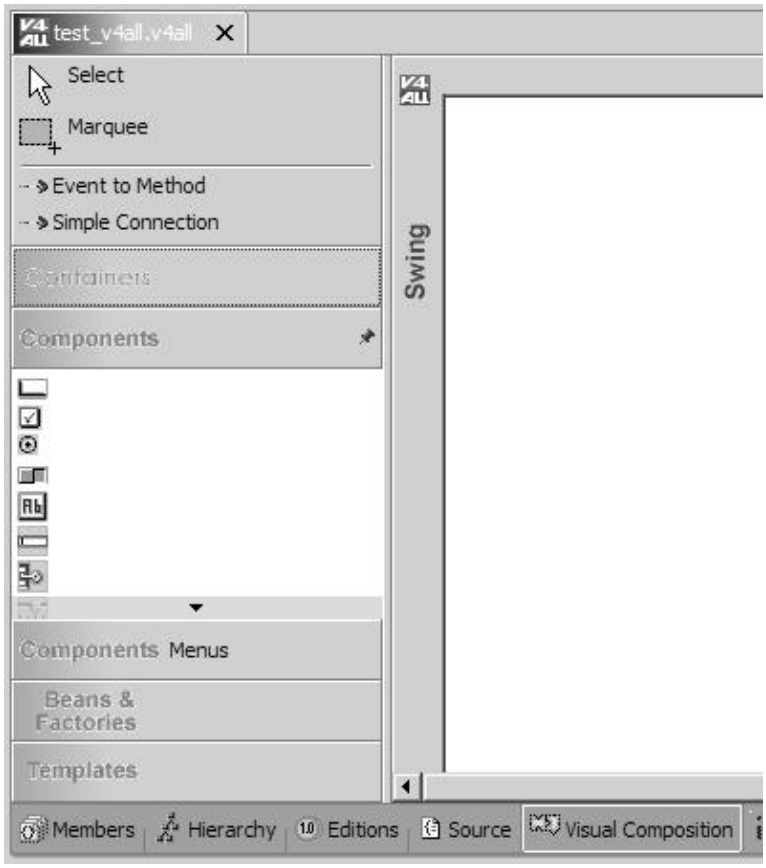


V4All propose son propre éditeur pour réaliser de manière WYSIWYG des interfaces graphiques.

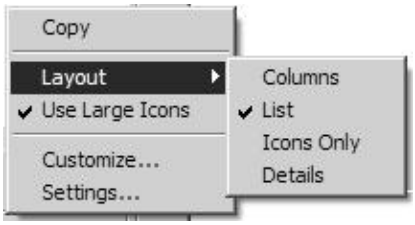
Cet éditeur se compose de deux parties :

- 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 peu 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.

17.2.3. Un exemple simple

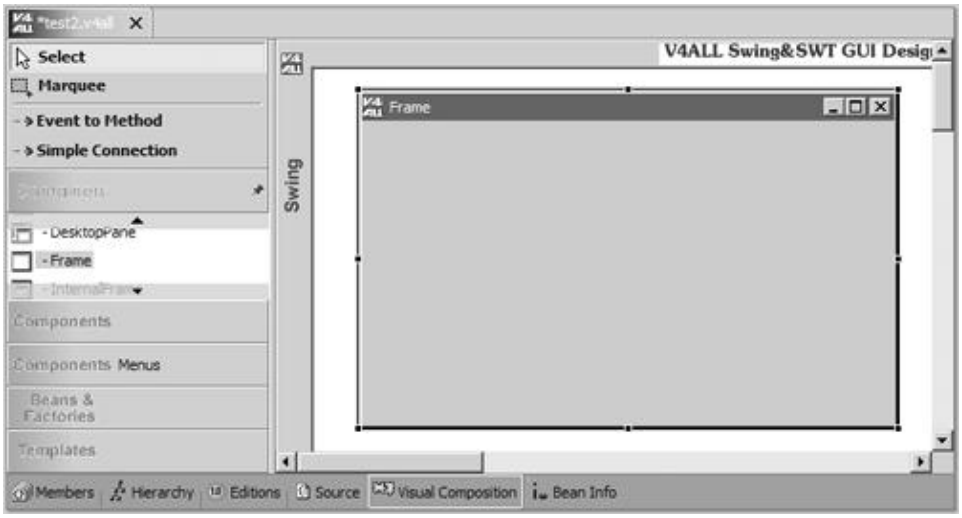
Créez un nouveau projet V4All.

Dans les propriétés, mettre la valeur « True » à la propriété « Generation / Generate Main ».

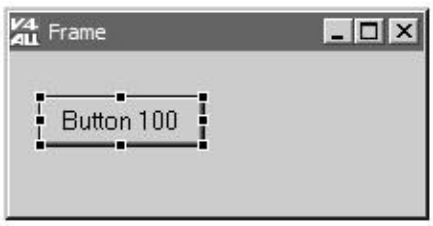
Faire un cliquer/glisser du composant Frame sur la zone d'édition

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 », …

Procéder de la même façon pour ajouter une zone de texte (composant de type TextField).

17.3. Le plug in W4Eclipse

http://w4toolkit.com/index.php

17.4. SWT designer

http://www.swt−designer.com/

SWT designer existe en deux version : une libre nommée "free edition" et une commerciale nommée « professional ».

18. 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