

Eclipse pour les **null**

Rémi Forax
forax@univ-mlv.fr

Eclipse pour les null

- Introduction
- L'espace de travail
- Configuration simple de l'environnement
- Création d'un projet Java
- Refactoring & Template
- Utilisation de CVS dans eclipse

Eclipse

- Code centric, tout est obtenu à partir du code
- Tout est plugin, donc on peut créer ses propres plugin :)
- Pas uniquement Java, CDT (C/C++), WTP (Web+JSP)
- Gratuit (www.eclipse.org)

Les perspectives d'eclipse

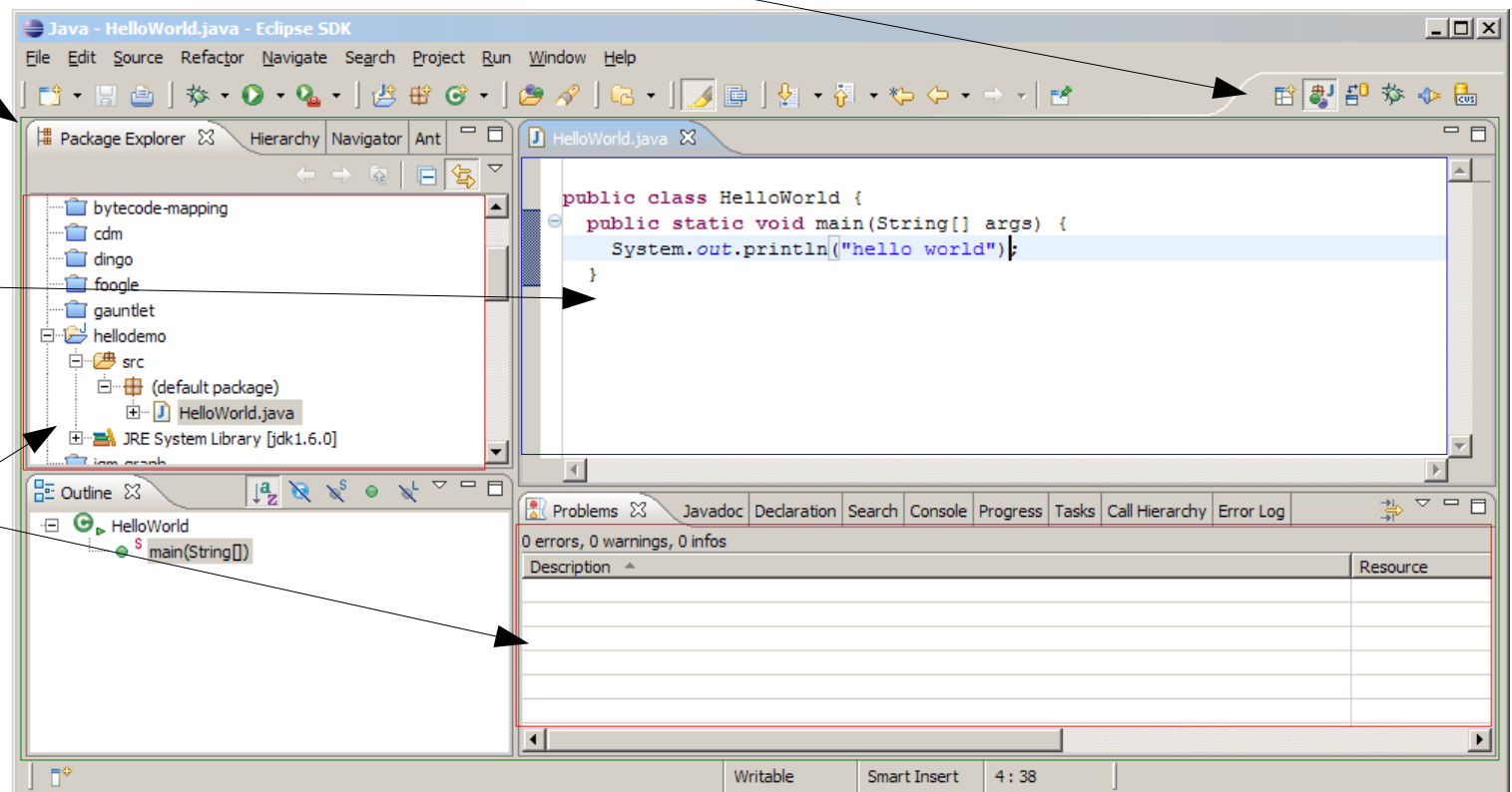
- La perspective Java

Changement de perspective

Perspective

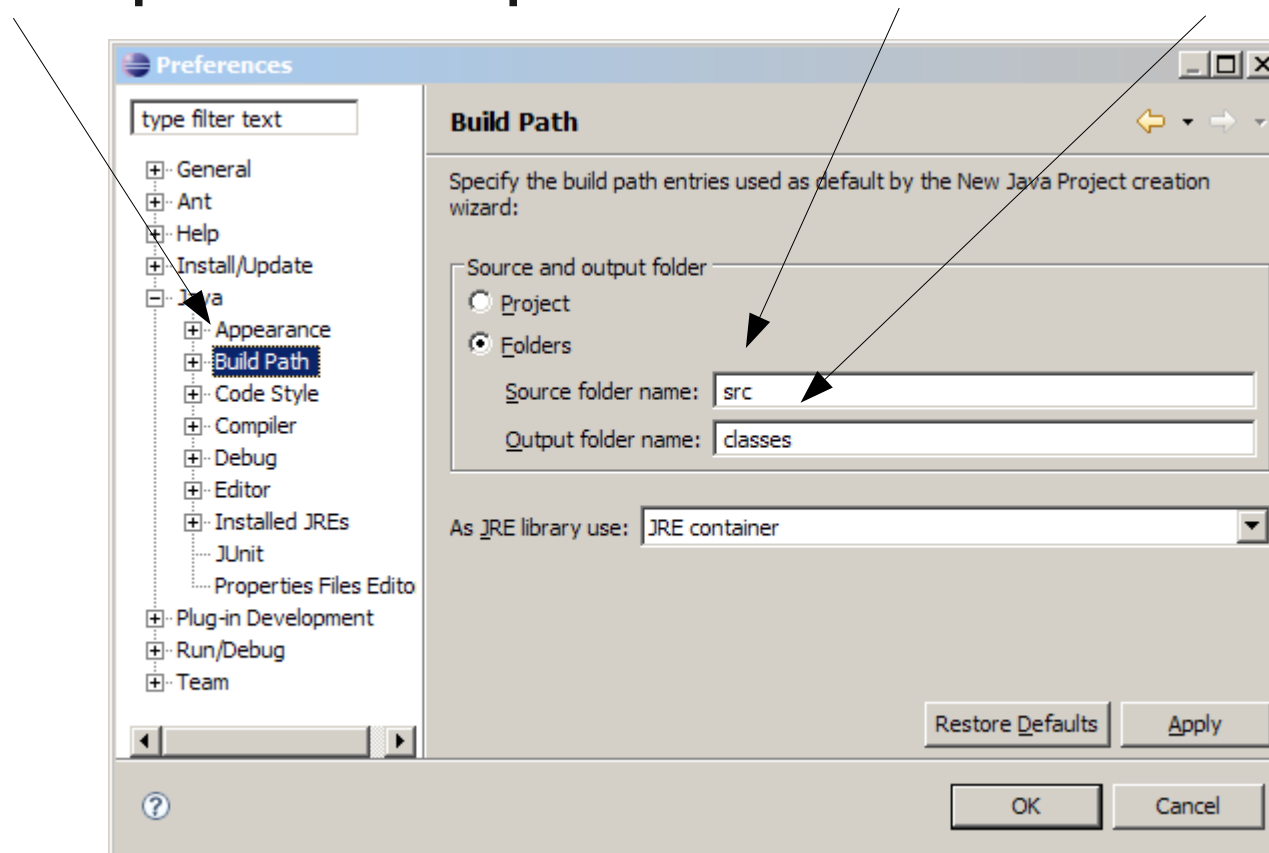
Editeur

Vues



Configurer Eclipse

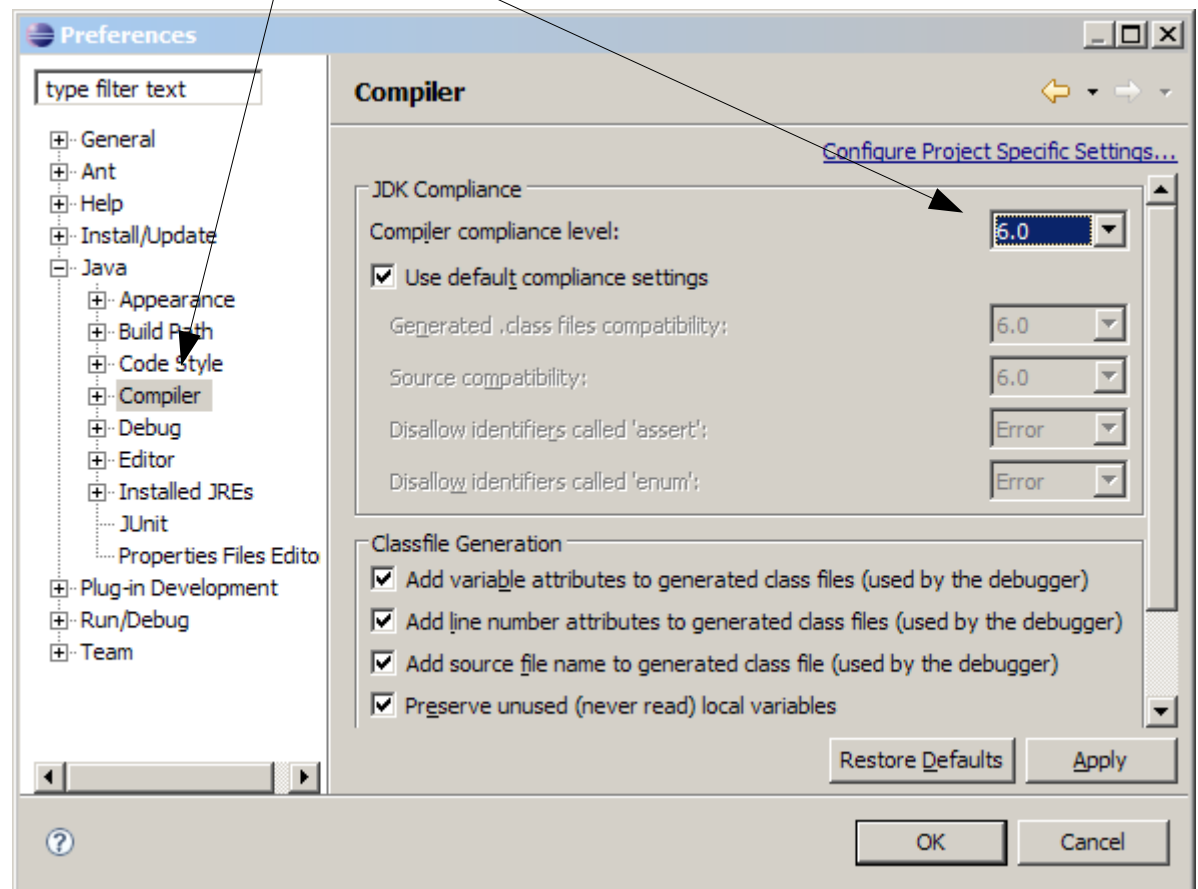
- Dans Window/Preferences...
Indiquer les répertoires **src** et **classes**



Configurer Eclipse (suite)

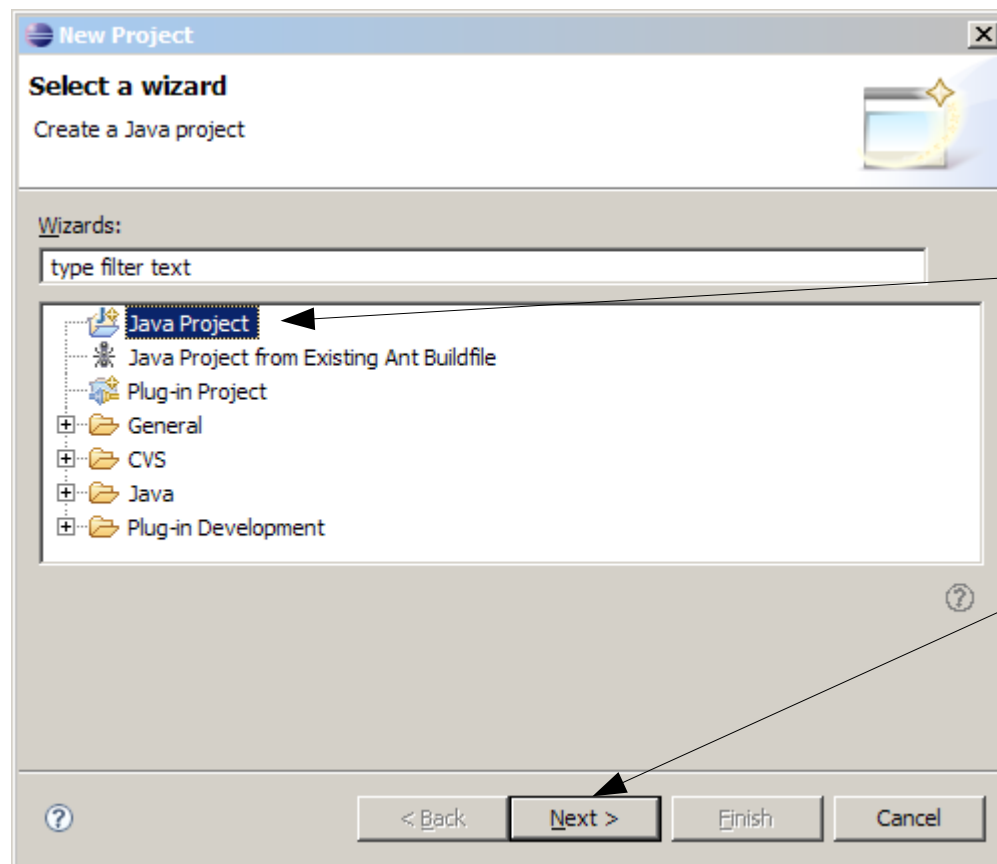
- Mettre la version du compilo à 6.0 (ou 5.0)

- Sinon pas de *generics* !!



Nouveau projet Java

- File > New > Project...



Choisir un projet Java

Puis next

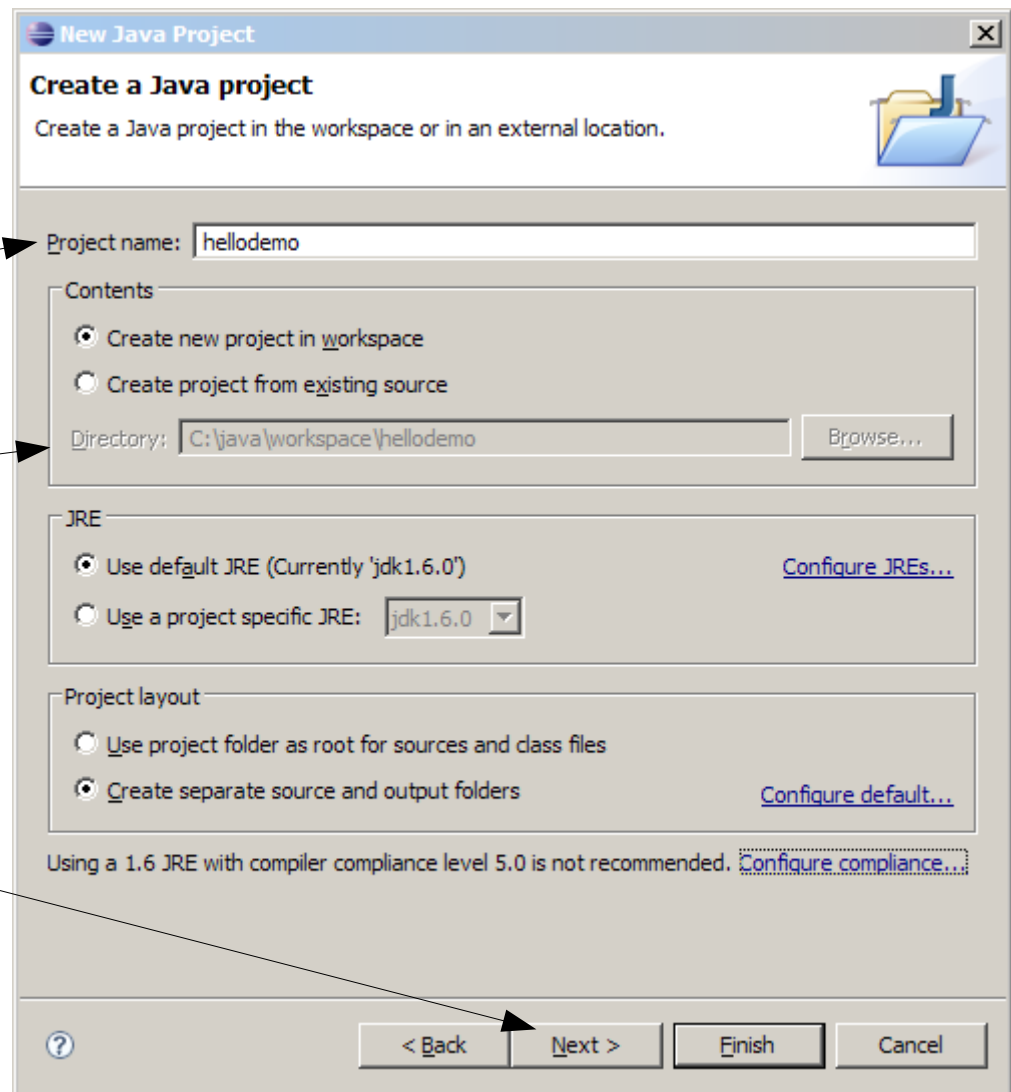
Nom du projet

- Indiquer le nom du projet

Nom du projet

Du répertoire dans le workspace

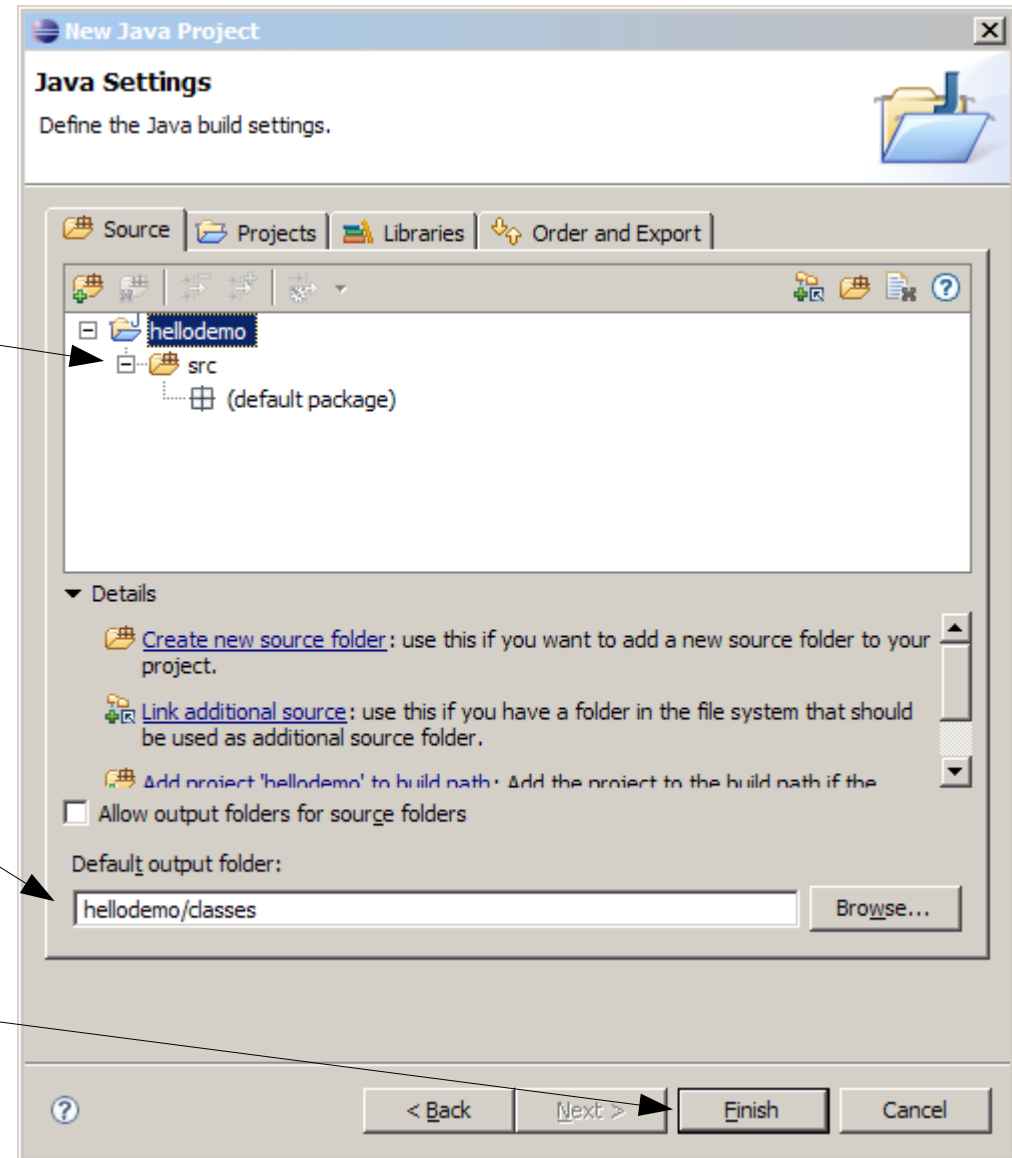
Puis next



Sous-répertoires du projet

- Les sources dans **src**
- Les classes dans **classes**

Puis finish



Nouvelle classe Java

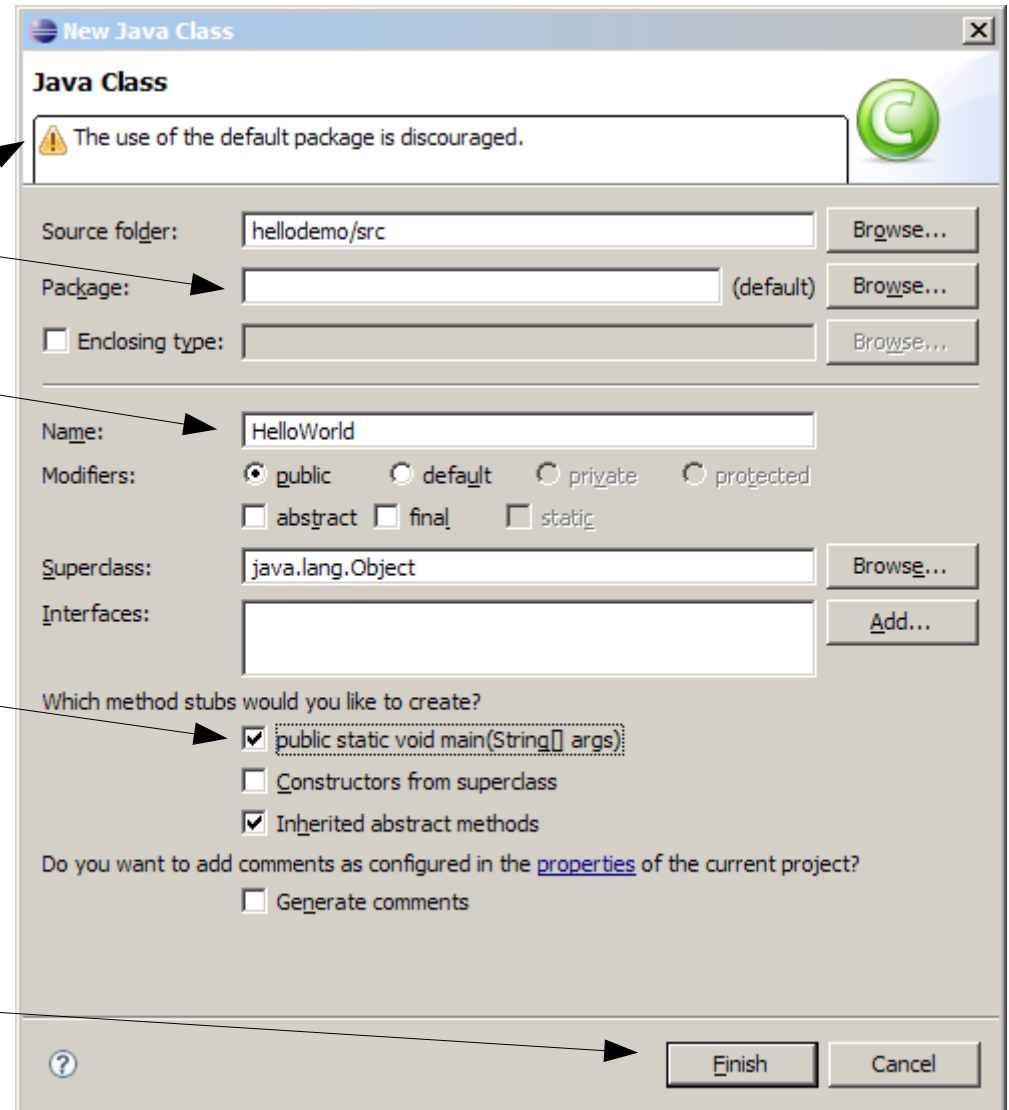
- File > New > Class

Package aucun (mal)

Nom de la classe

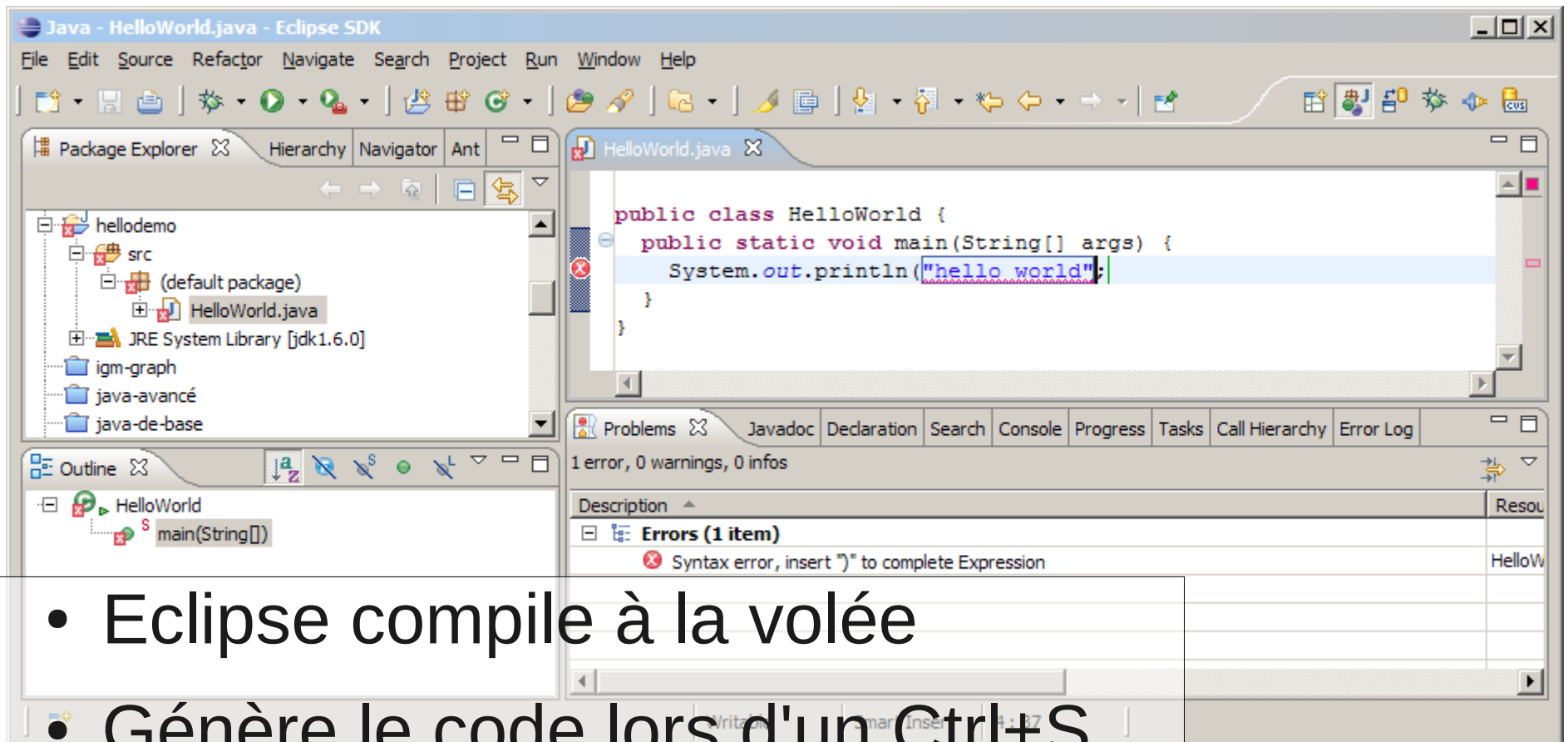
Je veux un main()

Puis **finish**



On écrit le code

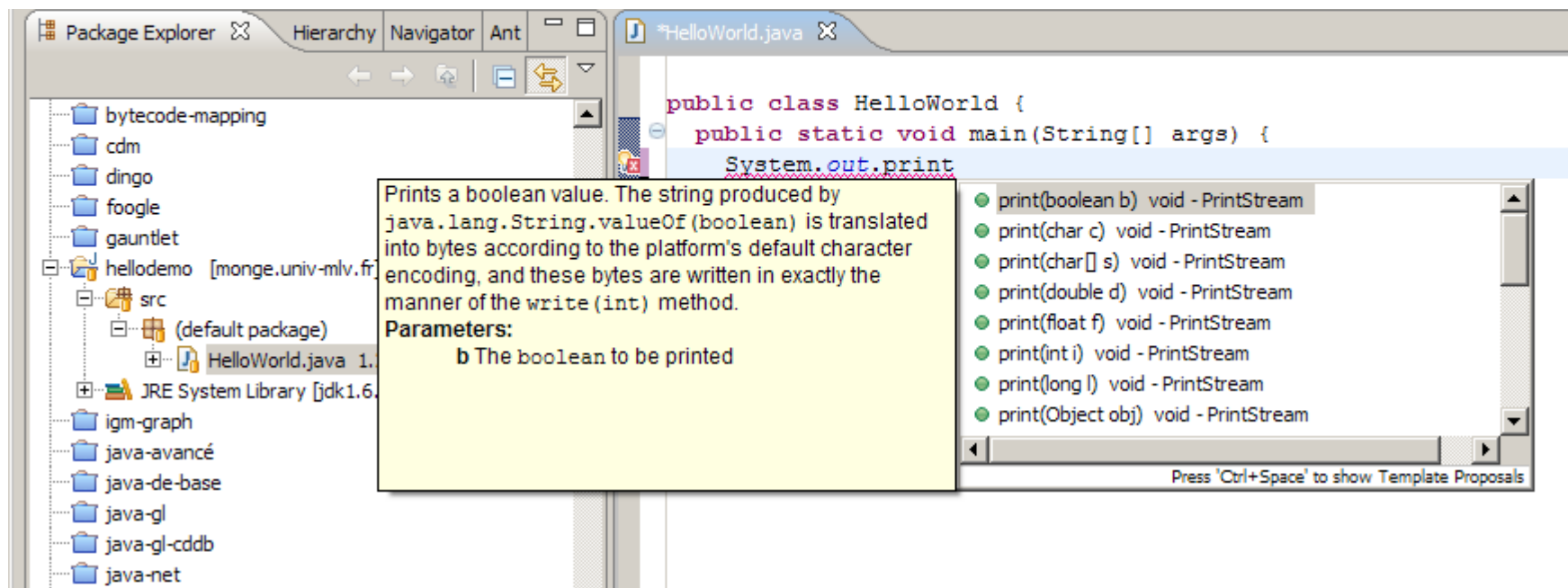
- On tape le code de la classe HelloWorld



- Eclipse compile à la volée
- Génère le code lors d'un Ctrl+S

Complétion contextuel

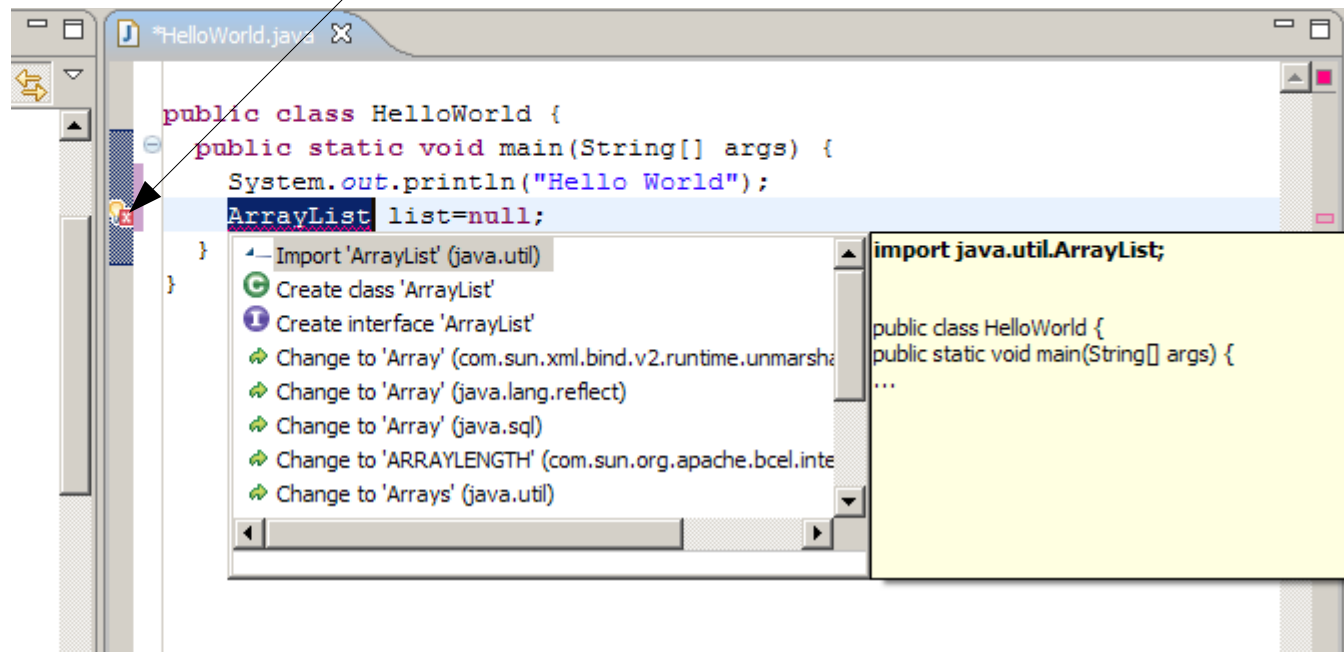
- Ctrl + Espace demande la complétion



- Et affiche la javadoc !!

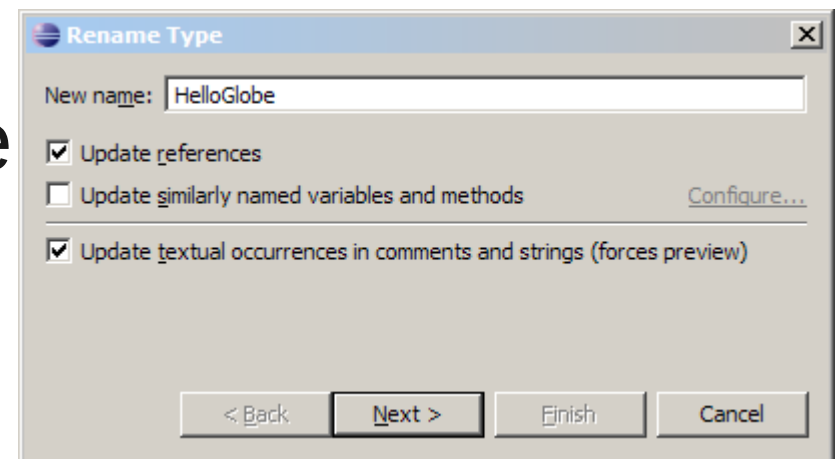
Quick Fix corrige les erreurs

- Un clique sur l'icône dans la marge propose diverses corrections (ne pas toujours choisir la première !!)



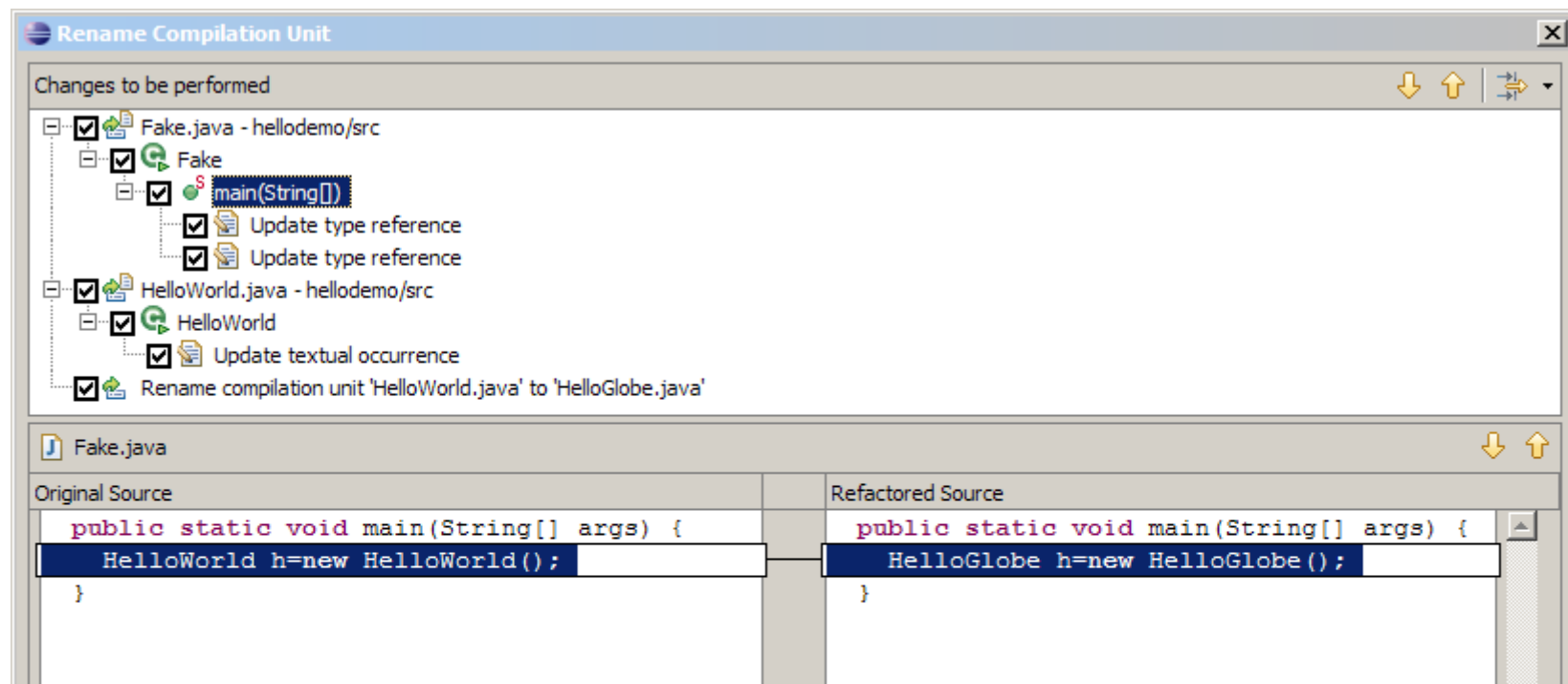
Refactoring

- Le refactoring correspond à des changements dans le code sans modification de la sémantique
(Renommer un champs, une méthode, déplacer une méthode etc.)
- Bouton droit sur la classe
Refactor > Rename



Refactoring (2)

- Indique l'ensemble des changements à effectuer lors du changement de nom de la classe



Shortcuts

- Code Assist (Ctrl+space)
- Quick Fix (Ctrl+1)
- Refactoring (Alt+Shift+T)
- Auto-Import (Ctrl+Shift+O)
- Surround With (Alt+Shift+Z)
- Call Hierarchy (Ctrl+Alt+H)
- Quick Type Hierarchy (Ctrl+T)
- Quick Outline (Ctrl+O)
- Show all Shortcuts (Ctrl+Shift+L)

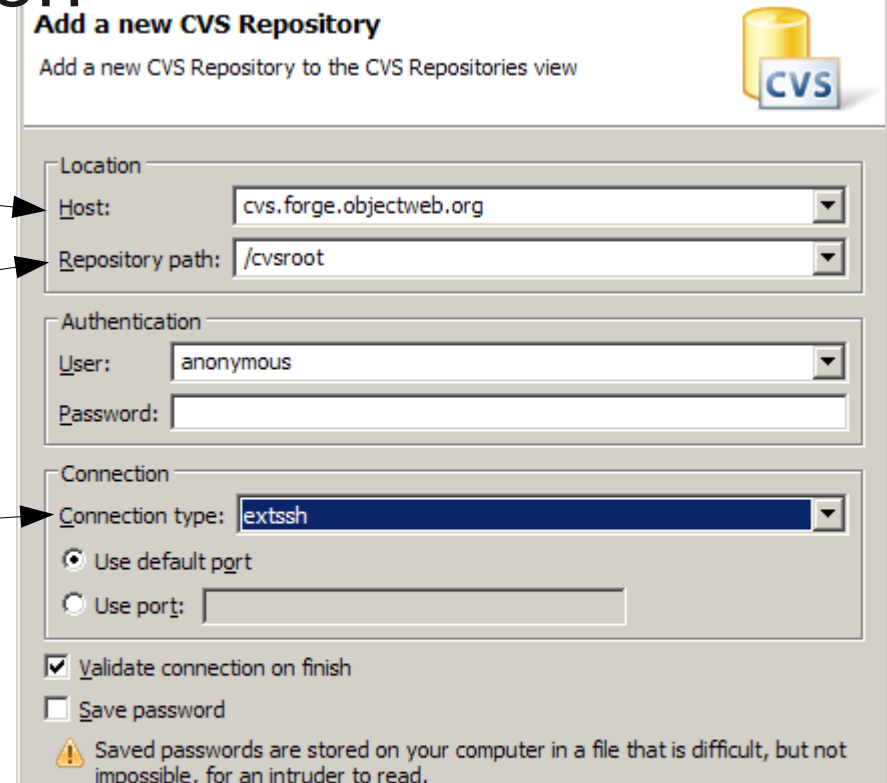
Créer un repository CVS

- Changer en perspective CVS, dans la vue CVS Repositories, bouton droit, New > Repository Location

Nom de la machine

Répertoire sur la machine

Type de connexion **extssh**



Add a new CVS Repository
Add a new CVS Repository to the CVS Repositories view

Location
Host:
Repository path:

Authentication
User:
Password:

Connection
Connection type:
☒ Use default port
☐ Use port:

☒ Validate connection on finish
☐ Save password

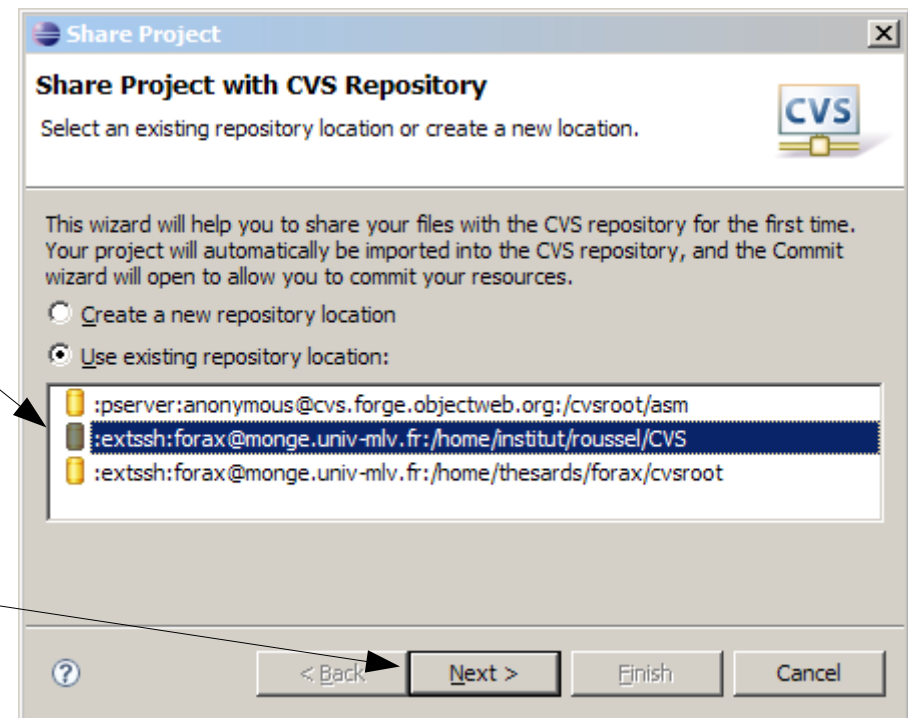
 Saved passwords are stored on your computer in a file that is difficult, but not impossible, for an intruder to read.

Mettre un projet sur CVS

- En perspective Java, sur un projet, bouton droit, Team > Share Project

Sélectionne le repository

Puis next

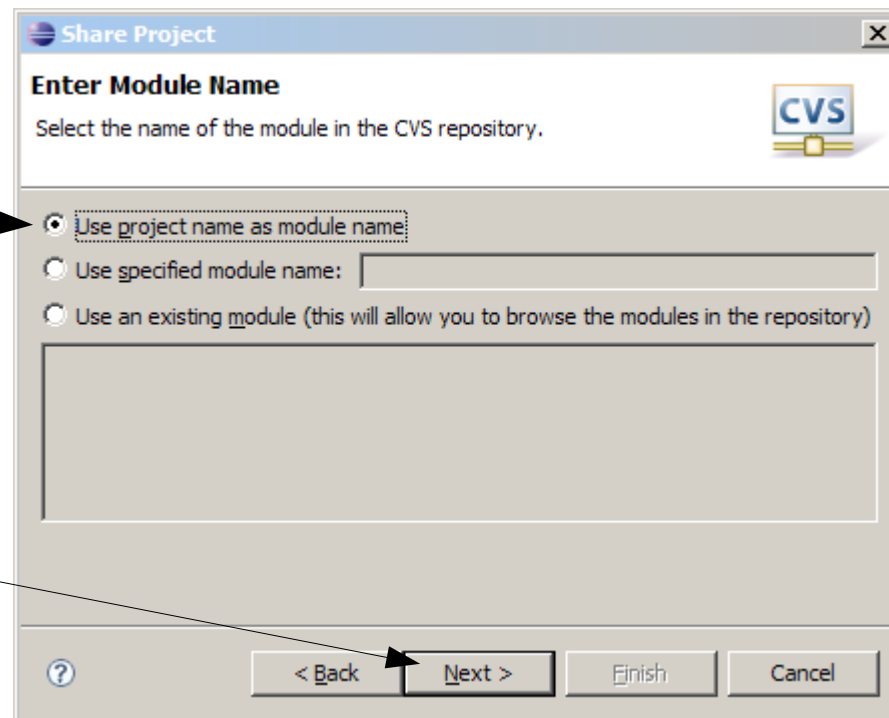


Mettre un projet sur CVS (suite)

- On indique le nom du module CVS correspondant (ici le même)

Le nom du projet est le nom du module

Puis next

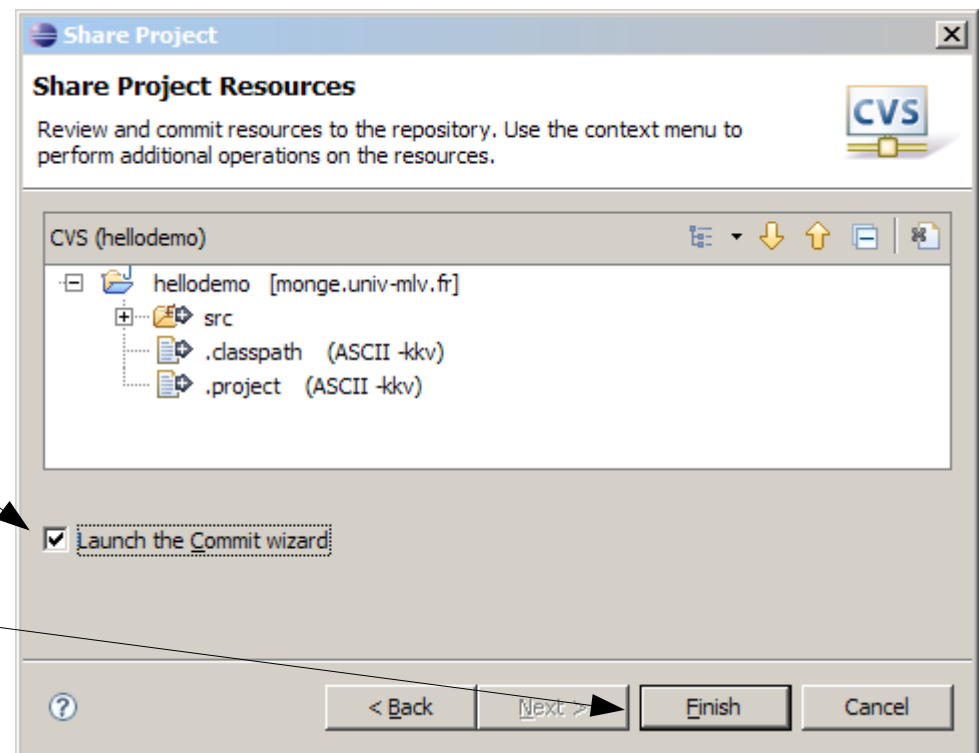


Import du projet sur le repository

- On voit l'ensemble des fichiers à mettre sur le repository

Pas obligatoire,
sinon voir **commit**

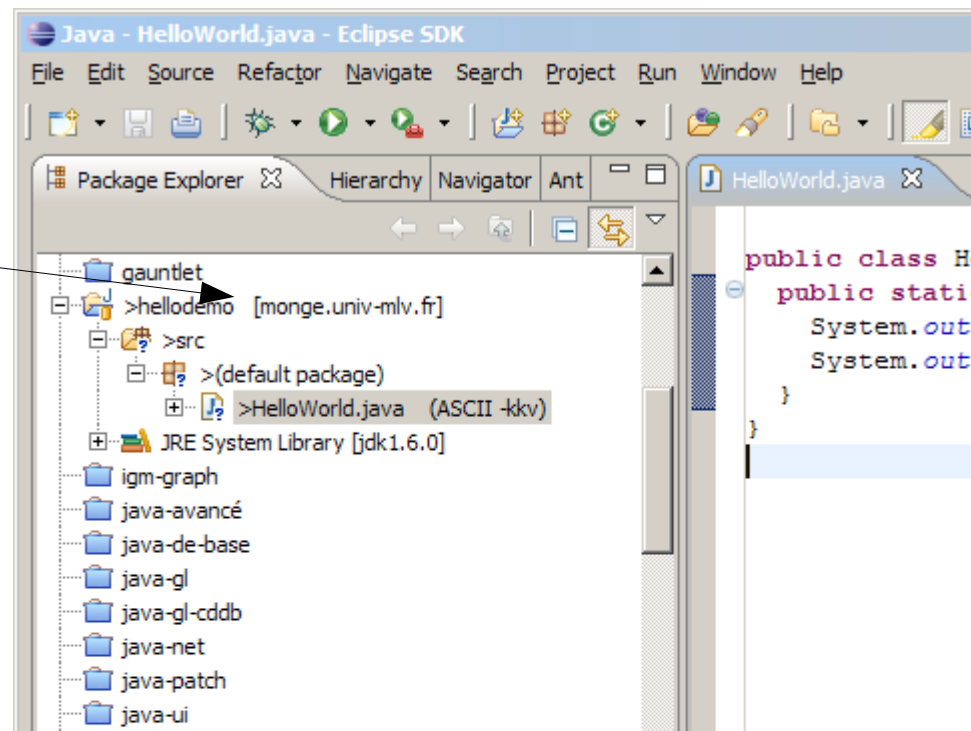
Puis **finish**



Le projet est importé

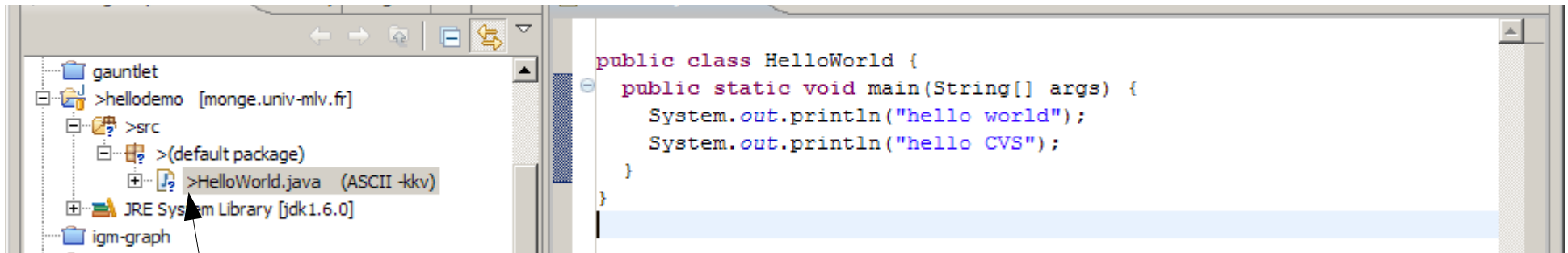
- Le projet est maintenant stocké de façon versionnée sur le repository CVS

Ici, le repository est sur
monge.univ-mlv.fr



Changement

- Si l'on change en local



Le symbole '>' indique que la version a été modifiée par rapport à la version sur le repository

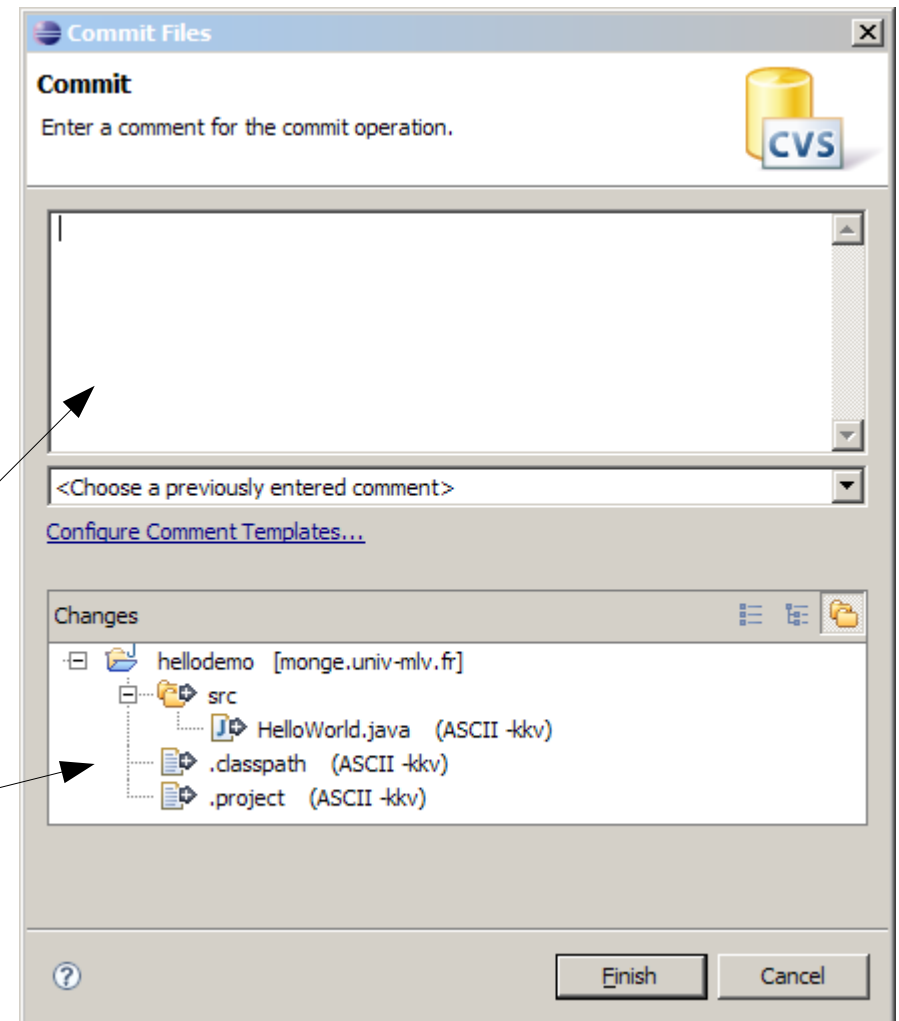
Il faut faire un **commit** de la nouvelle version

Commit des changements

- Sur le projet (ou sur une ressource) bouton droit, Team > Commit ...

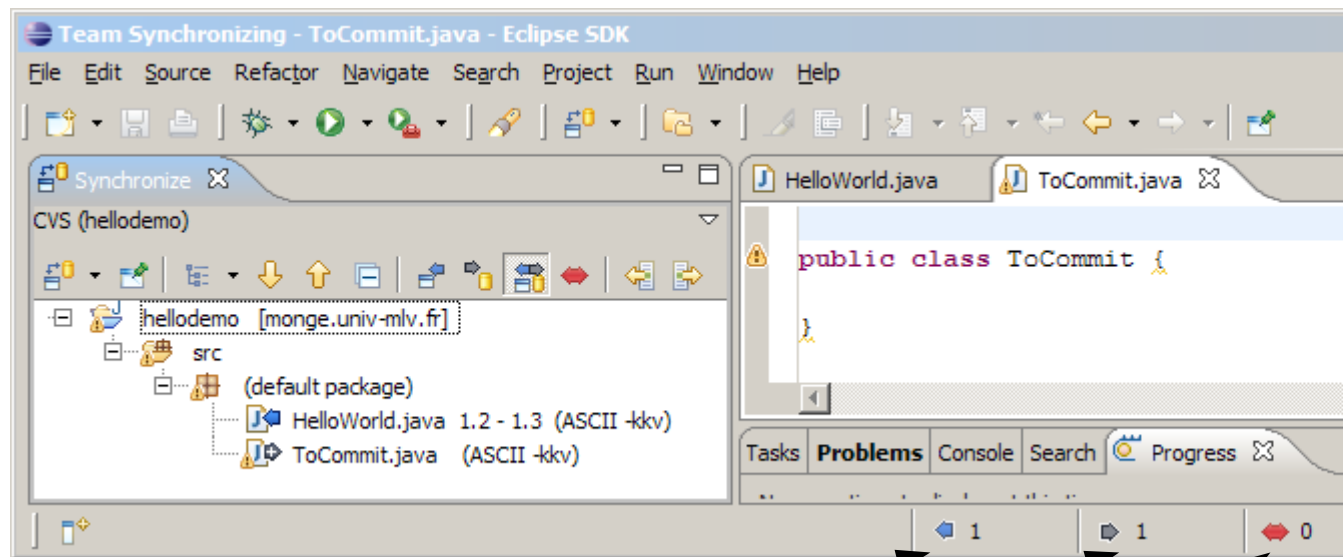
On indique les commentaires de révisions

Puis **finish**



Update des changements

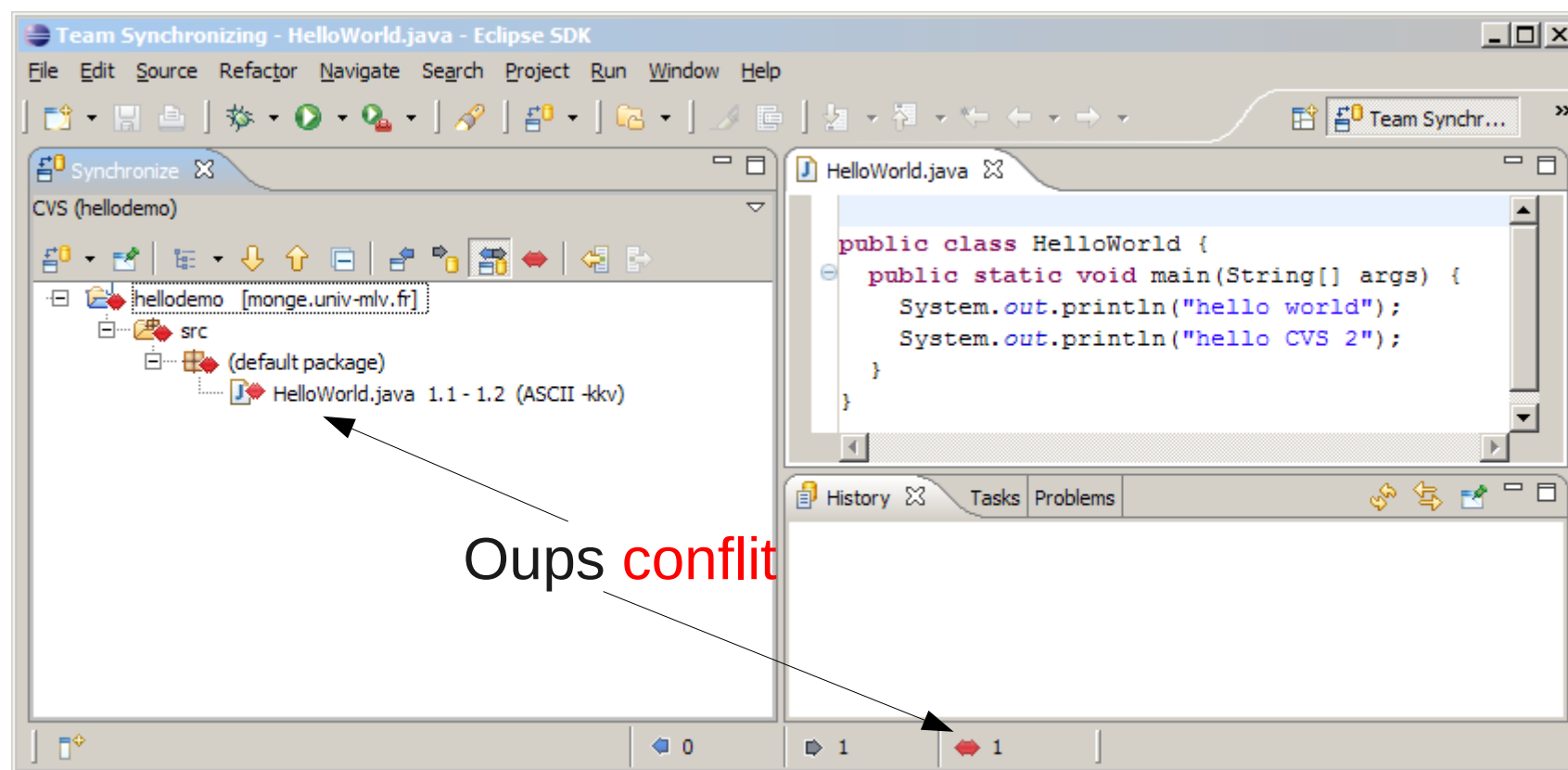
- Prendre en compte les changements des autres, bouton droit, Team > Synchronize



1 commit, 1 **update** et zéro **conflit**

Update avec conflit

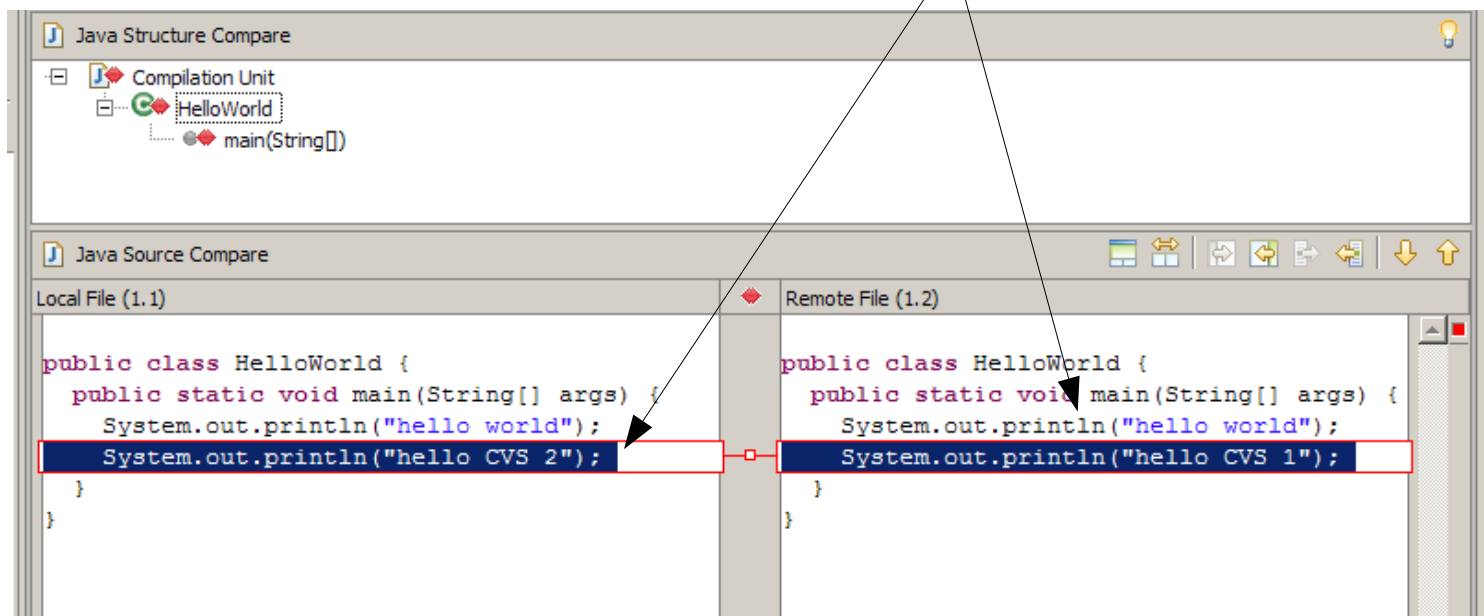
- Il peut y avoir des conflits



En cas de conflit

- Double clic sur le conflit

Deux lignes différentes



- Il faut résoudre le conflit, “à la main”

En cas de conflit

- Trois façon de résoudre :
 - La version repository a raison (sur la ressource, Override and update)
 - La version locale a raison (sur la ressource, Mark as merge, puis commit)
 - Un mix entre les deux a raison (on fait des copier/coller entre les deux pour que la version locale soit juste)

Après un Mark as merge il est candidat au commit

