Eclipse JDT

www.eclipse.org

Jörg Steffen,DFKI

steffen@dfki.de

29.10.2010

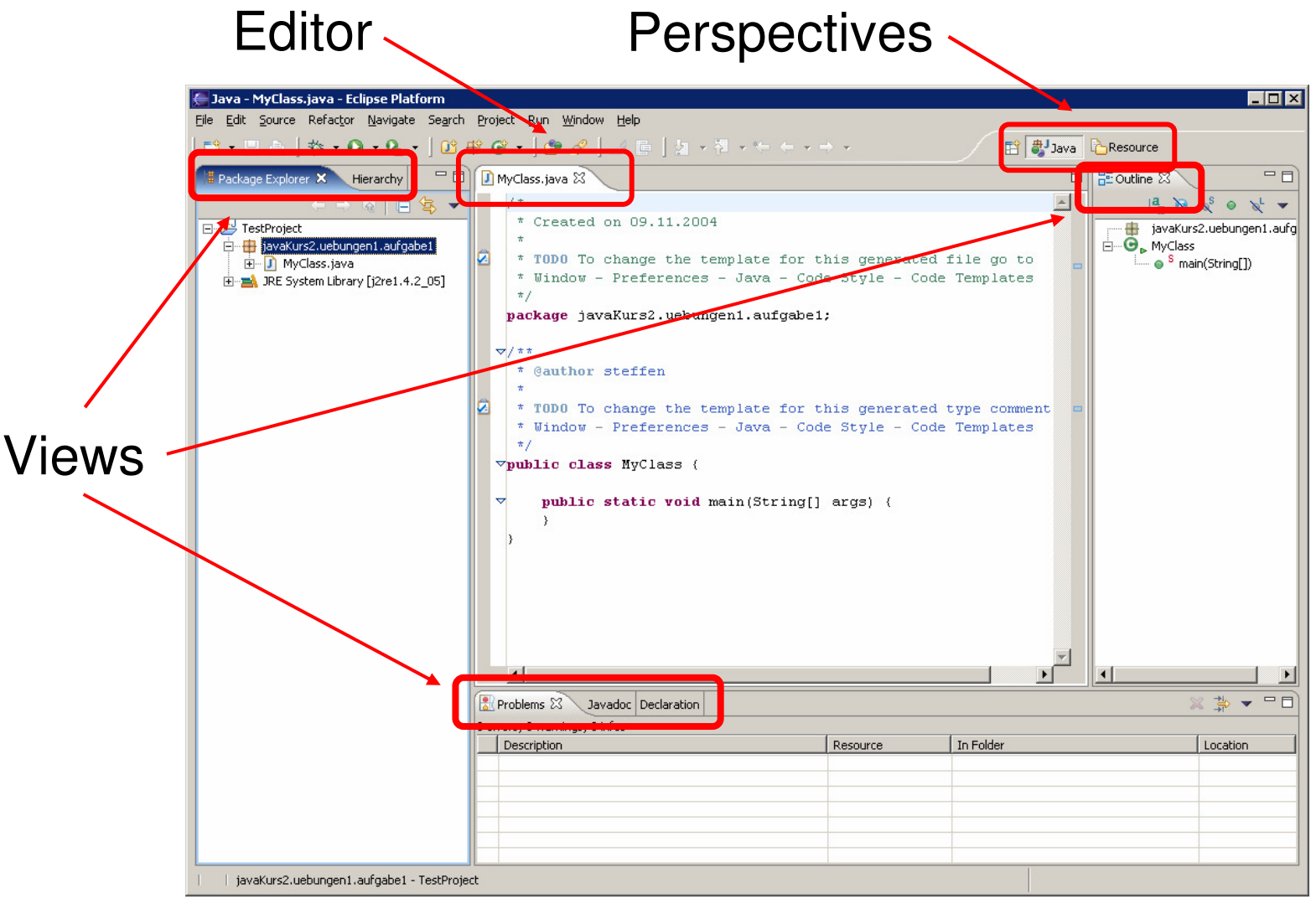
What is the Eclipse JDT?

- JDT = Java development tool
- State of the art Java development environment
- Built atop Eclipse Platform
- Freely available for wide range of operating systems
- Helps programmers write and maintain Java code
- Takes care of translating Java sources to binaries
- And much much more...
- Current version is the 3.6.1 (Helios) release

The Eclipse Workbench

- Editors
  - Depending on the type of file that is being editing, the appropriate editor is displayed in the editor area
- Views
  - Support editors and provide alternative presentations or navigations of the information in the Workbench
- Perspectives
  - Defines the initial set and layout of views and editors in the Workbench window

The Eclipse Workbench



Preparations

- Open the preferences: Windows -> Preferences
- **`Java/Build Path`**
  - Change the source and output folder to ‘src’ and ‘classes’
- **`General/Editors/Text Editors`**
  - Make the print margin visible
  - Set tab with to 2 and use spaces
- **`Java/Code Style/Formatter`**
  - Import the Java Kurs code formatter
- **`General/Editors/Structured Text Editors`**
  - Remove annoying popups → Combined Hover: off

Creating a Java Project

- Create a folder for the project, e.g. JavaKurs
- In Eclipse: File -> New -> Project...
- Select ‘Java Project’, click ‘Next’
- Name the project
- Uncheck ‘Use default location’ and use the above folder instead, click ‘Next’
- Define the build settings, e.g. make external jars available for the project (not required now)
- click ‘Finish’

Creating a Package with a Class

- In the package explorer, right-click on ‘src’ and select ‘New Package’
- Name the package, e.g. javakurs
- In the package explorer, right-click on the new package and select ‘New Class’
- Name the class and optionally change the modifiers, interfaces, etc.
- Check ‘public static main(String[] args)’ to create the stubs for the main method
- Check ‘Generate comments’ to create the header for the class
- Notice the template for the class documentation

  → **`Java/Code Style/Code Templates`**

Running a Java Program

- Write some code in the main method that prints the first given argument:

```
if (args.length > 0) {
  System.out.println("The first arg is " + args[0]);
}
```

- Select Run -> Run Configurations...
- Chose ‘Java Application’, right click it and select ‘New’
- Click the ‘Arguments’ tab and enter some program arguments: **`arg1 arg2 arg3`**
- Click ‘Apply’ and ‘Run’ to run the program
- Notice that a new view ‘Console’ is opened in the workbench that shows the output of the program
- To run the program again, select the class in the package explorer and press Ctrl-F11

Tips & Tricks

- Content Assistant
  - Suggests completions for partially entered strings
  - Lists all methods for a class
  - Expands templates → **`Java/Editor/Templates`**
  - Ctrl-Space
- Add Import
  - Creates an import statement for the class under the cursor
  - Ctrl-Shift-M
- Organize Imports
  - Adds all necessary import statements
  - Removes all unneeded import statements
  - Ctrl-Shift-O

Tips & Tricks

- Quick Fix
  - Suggests a solution to a problem
  - Indicated with an electric bulb on the left side
  - Ctrl-1
- Show Javadoc for class/method as tooltip
  - Click on class/method name in editor and press F2
  - Alternative: Shift-F2 opends doc in browser tab
- Jump to source code of class/method
  - Ctrl-click a class or method name in the editor

Tips & Tricks

- Highlight occurrences
  - Click element to mark its occurrences in the current file
  - Distinguishes read and write occurrence
  - Colors can be edited under
    `General/Editors/Text Editors/Annotations`
- Correct Indentation
  - Mark code and press Ctrl-I
- Add Javadoc comment to class, method, member variable
  - Alt-Shift-J
- Generate getters and setters
  - Context menu: Source → Generate Getters and Setters...

Tips & Tricks

- Rename classes/methods/variables
  - Also updates references
  - Alt-Shift-R
- Find/Replace strings in code
  - Ctrl-F
- Toggle Comment
  - Mark code and press Ctrl-/ (Windows), Esc Ctrl-C (Linux)
- Revert back to or compare with a previous edition of a method/class
  - Right-click method/class and chose ‘Compare/Replace With History...’

Tips & Tricks

- Show class hierarchy
  - Hierarchy containing the super and sub classes of a selected class
  - F4
- Show call hierarchy of method
  - List all call occurrences of a selected method
  - Ctrl-Alt-H

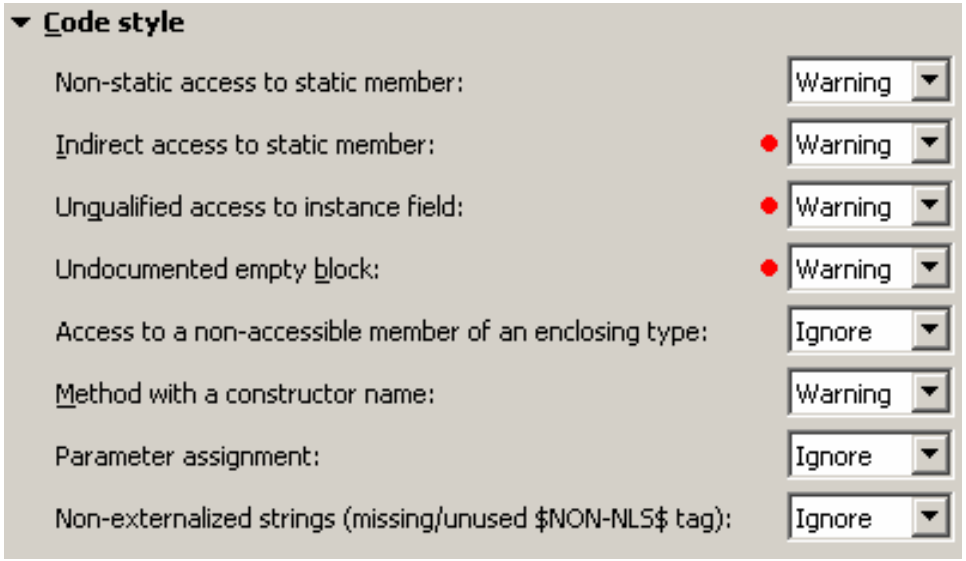
Coding Standards Support in Eclipse

Coding Standard Tools

- Java code formatter
  - automatically formats your code according to a predefined profile
- Java compiler warnings
  - show warnings for certain compiler problems in the Java source editor
- Javadoc warnings
  - show warnings for documentation problems
- Checkstyle plugin
  - allows fine grained definition of a coding standard
  - shows warning in the Java source editor if coding standard is not met
- Goal: write code without warnings

Java Compiler Warnings

- Update warnings settings in **`Java/Compiler/Error/Warnings`**

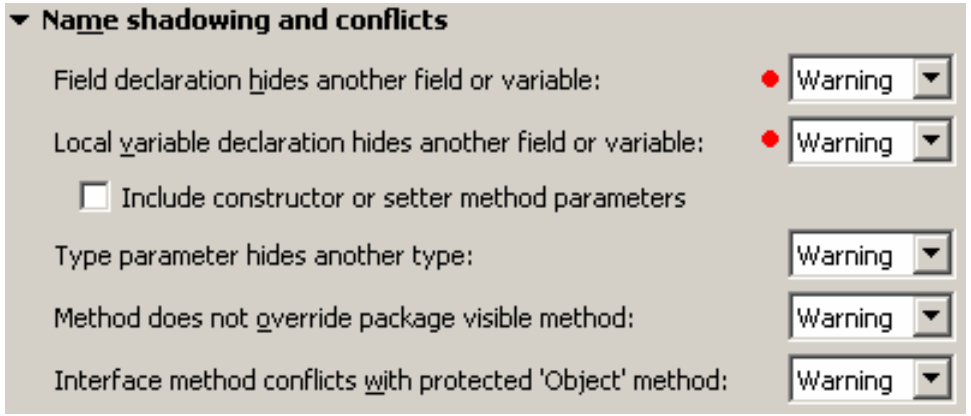


Java Compiler Warnings

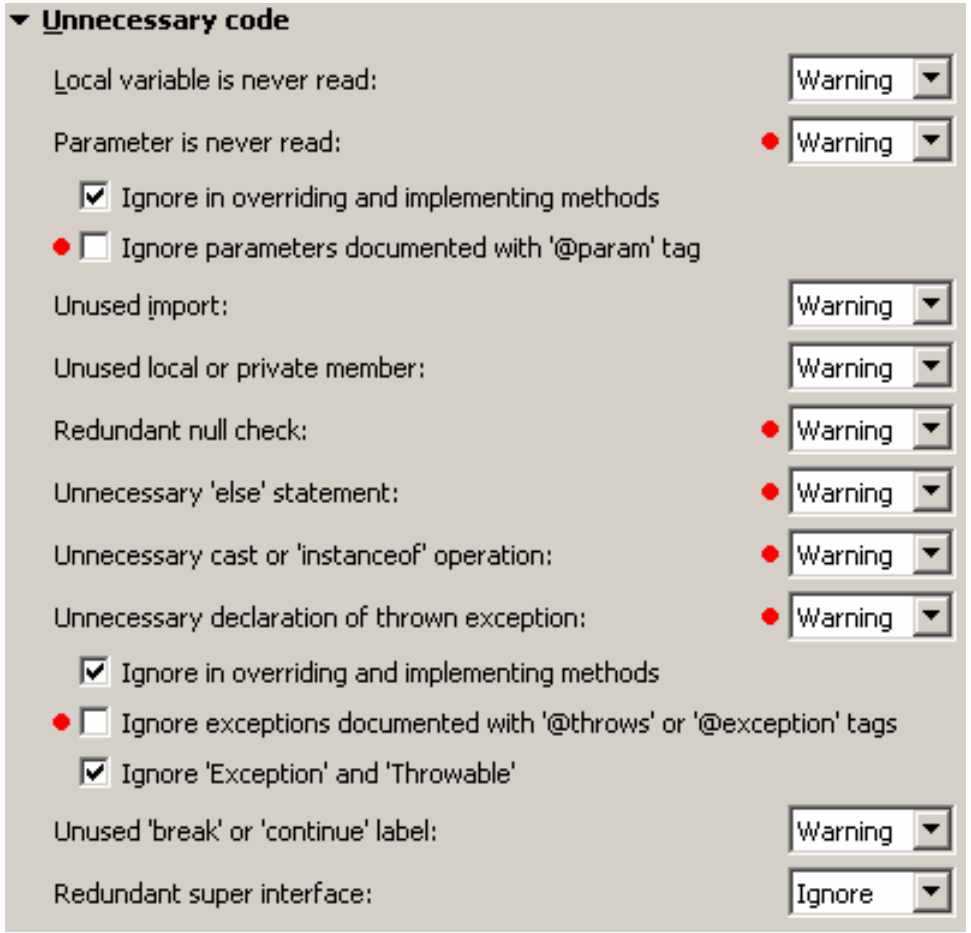
Potential programming problems

Serializable class without serialVersionUID:	Ignore
Assignment has no effect (e.g. 'x = x'):	Warning
Possible accidental boolean assignment (e.g. 'if (a = b)'):	Warning
'finally' does not complete normally:	Warning
Empty statement:	Ignore
Using a char array in string concatenation:	Warning
Hidden catch block:	Warning
Inexact type match for vararg arguments:	Warning
Boxing and unboxing conversions:	Ignore
Enum type constant not covered on 'switch':	Ignore
'switch' case fall-through:	Warning
Null pointer access:	Warning
Potential null pointer access:	Warning
Comparing identical values ('x == x'):	Warning
Missing synchronized modifier on inherited method:	Ignore
Class overrides 'equals()' but not 'hashCode()':	Ignore
Dead code (e.g. 'if (false)'):	Warning

Java Compiler Warnings

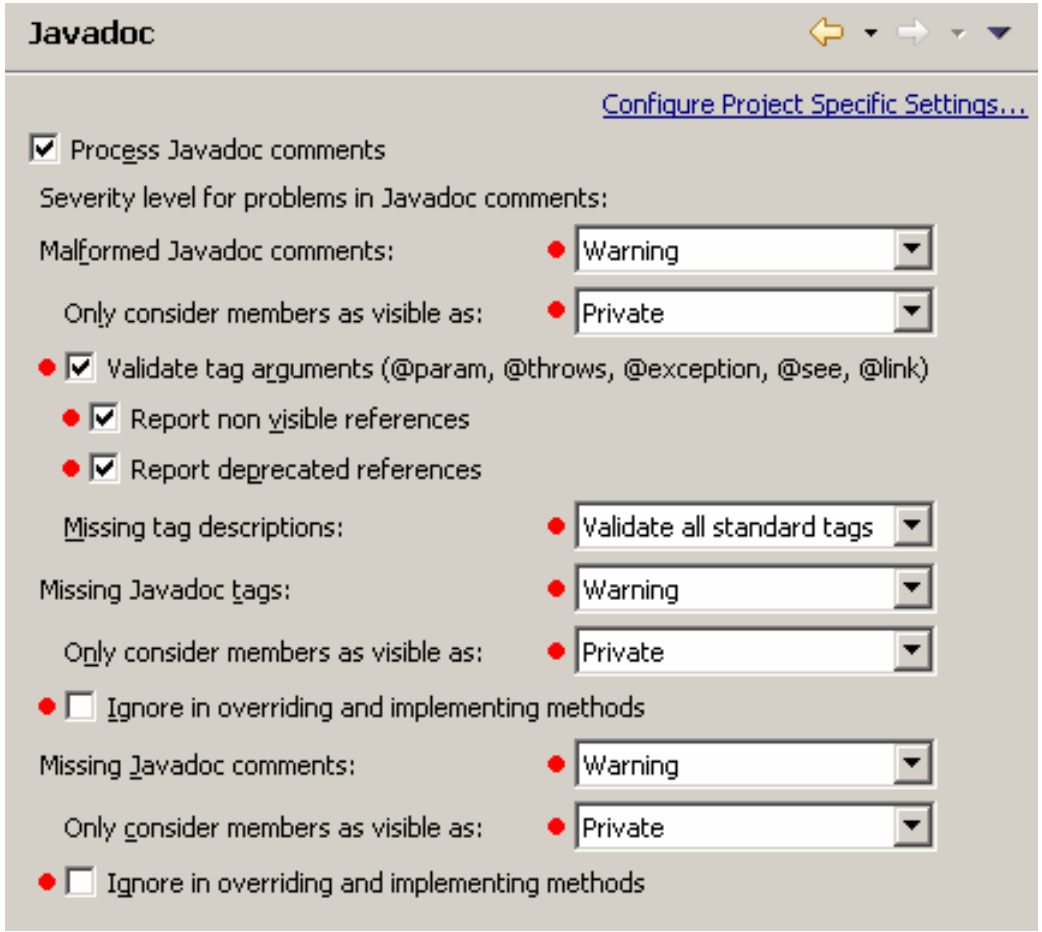


Java Compiler Warnings



Javadoc Warnings

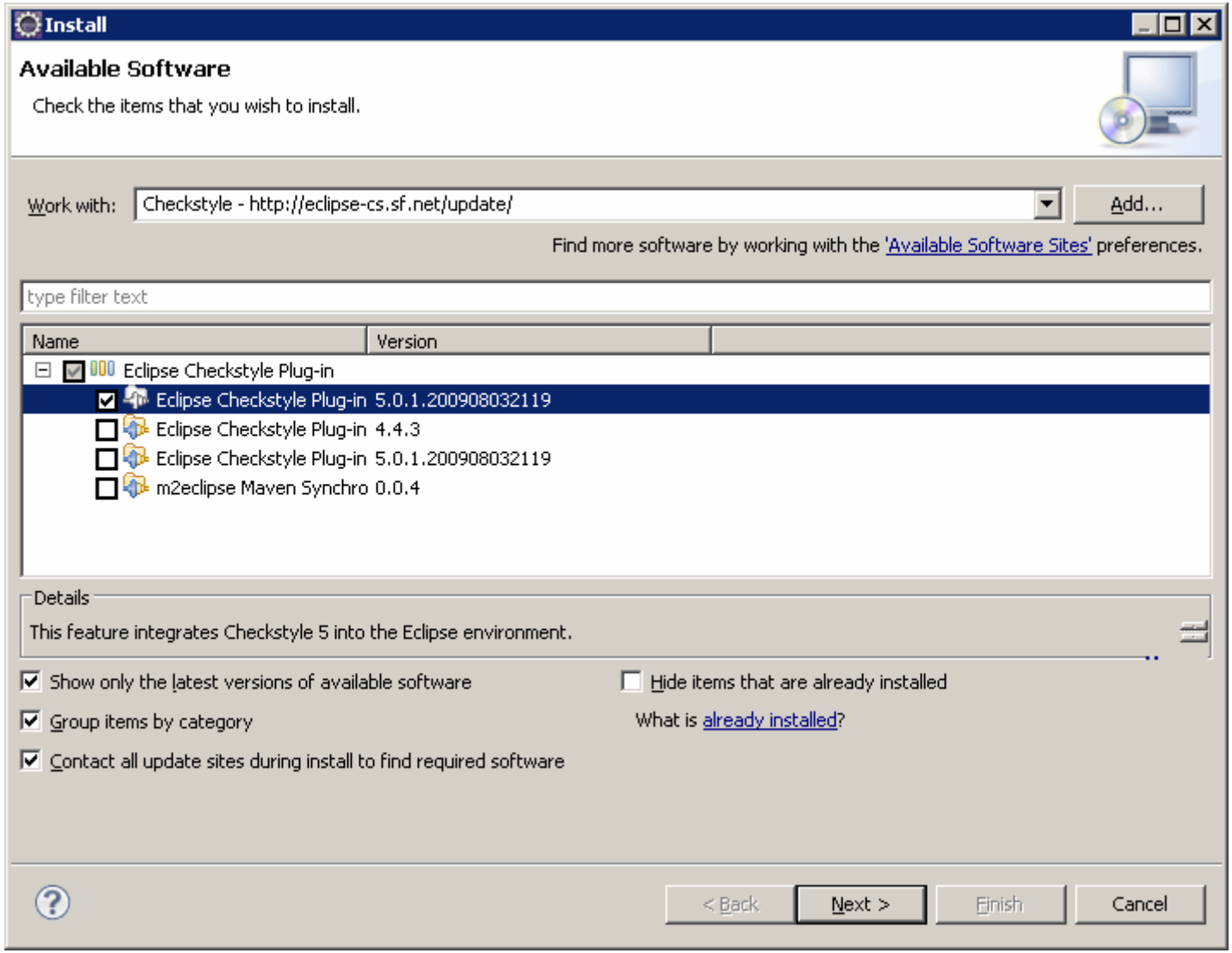
- Update warnings settings in **`Java/Compiler/Javadoc`**



Checkstyle Installation

- Also see http://eclipse-cs.sourceforge.net/
- Installation in Eclipse 3.6
  - Help -> Install New Software...
  - Add site: http://eclipse-cs.sourceforge.net/update
  - Only select the *Eclipse Checkstyle Plugin 5.0.1* package
  - Press *Next* to install the Plugin
  - Restart Eclipse

Checkstyle Installation



Checkstyle Configuration

- Open the preferences: Windows -> Preferences
- Select **`Checkstyle`** and add a new Checkstyle configuration
- Create an external configuration from checkstyle.xml
- Set new configuration as default configuration
- In the project properties, you can now activate Checkstyle for this project

Checkstyle Configuration