

ECLIPSE CHEAT SHEET

This cheat sheet is aimed at people with some experience in eclipse but who may not be aware of many of its features.

SHORTCUTS

These are some of the most useful shortcuts in eclipse. You can find others listed beside their function in eclipse's menus.

Menu	Key Combination	Use
Source	Ctrl+Space	Auto-complete / Content Assist
	Ctrl+I	Quick Fix (on current selection)
	Ctrl+D	Delete Current Line
	Ctrl+Alt+Up	Copy Lines Up
	Alt+Up	Move Lines Up
	Alt+Shift+Z	Surround (current selection) With
	Ctrl+I	Fix Indentation (on selection)
	Ctrl+Shift+F	Fix Formatting (on selection)
	Ctrl+/ Toggle Comments (on selection)	
Refactor	Alt+Shift+M	Extract Method
	Alt+Shift+R	Rename
Search	Ctrl+Shift+G	Find All References (of selection)
	Ctrl+H	Advanced Search
Run	Alt+Shift+X, J	Run Current Class
	Alt+Shift+X, D	Debug Current Class

AUTO-COMPLETE

These are some of the most useful options given to you after using the Ctrl+Space auto-complete shortcut.

Text Written So Far	What the Auto-Complete Function Does
<start of variable/method name>	<end of variable/method name>
main	Creates a new main method
public / private	Creates a new public/private method
randomstring	Creates a new void method named <i>randomstring</i>
sysout	Creates a new System.out.println();
syserr	Creates a new System.err.println();
for	Creates a new for loop
foreach	Creates a new iterator loop
while	Creates a new while loop
try	Creates a new try-catch block

You can also create your own auto-complete statements. Go to Window->Preferences, then on the left pane go to

Java->Editor->Templates

For example the following template creates a println outputting the contents of a specific variable:

NAME: debug

PATTERN: System.err.println("\${word_selection}: " + \${word_selection}); \${cursor}

This can be used now in eclipse by selecting a variable and pressing Ctrl+Space, then typing 'debug' and pressing enter.

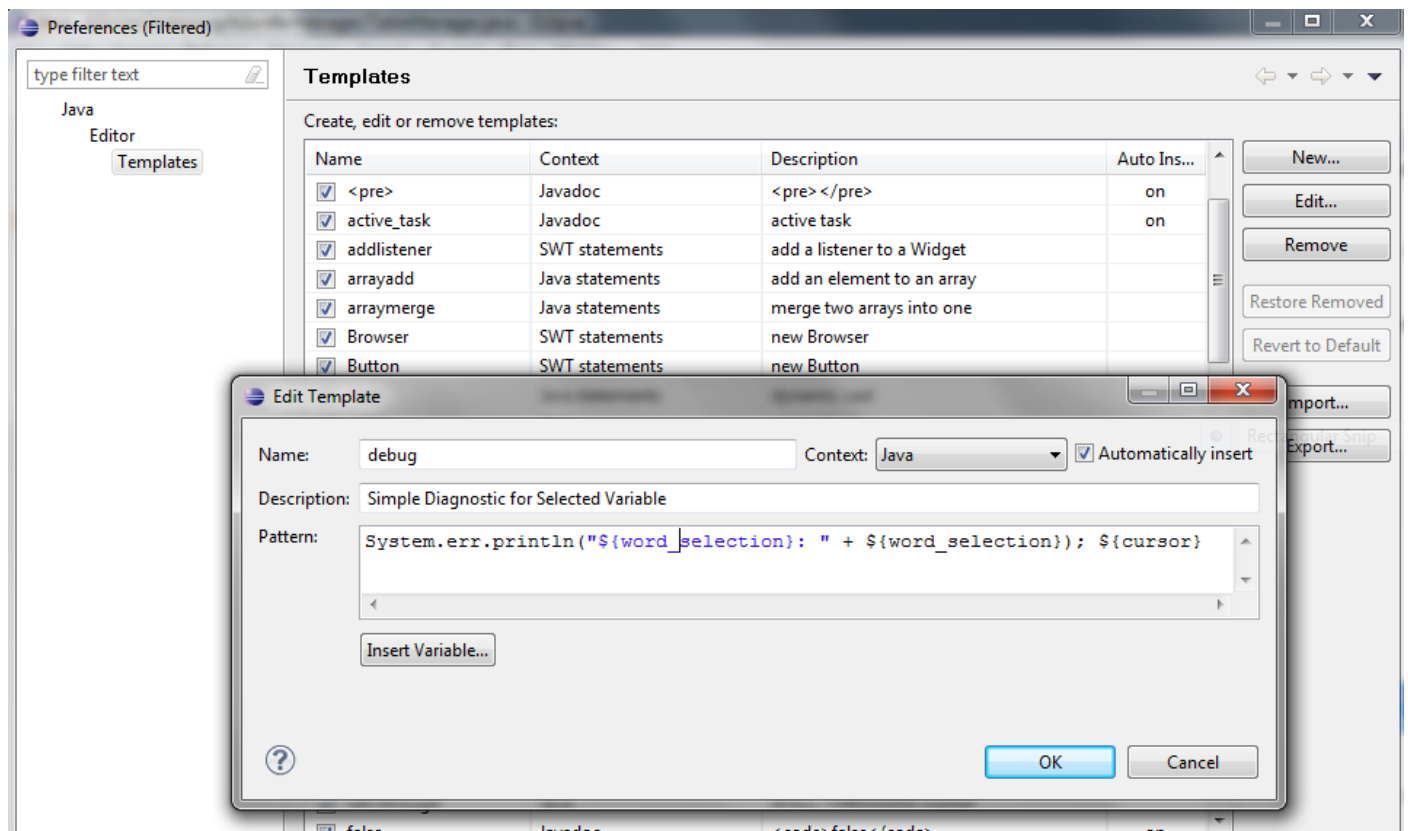


Figure 1: Creating a new auto-complete template

VIEWS IN ECLIPSE

Eclipse displays various views on the left and bottom of the screen that you may find useful. You can add new views if they are not already displayed by going to **Window->Show View**, or by clicking on the following icon at the bottom left of your workspace:



workspace:

TASK MANAGEMENT

The task pane (pictured below) is a useful way of keeping track of incomplete sections of code, or places where code needs tidied up.

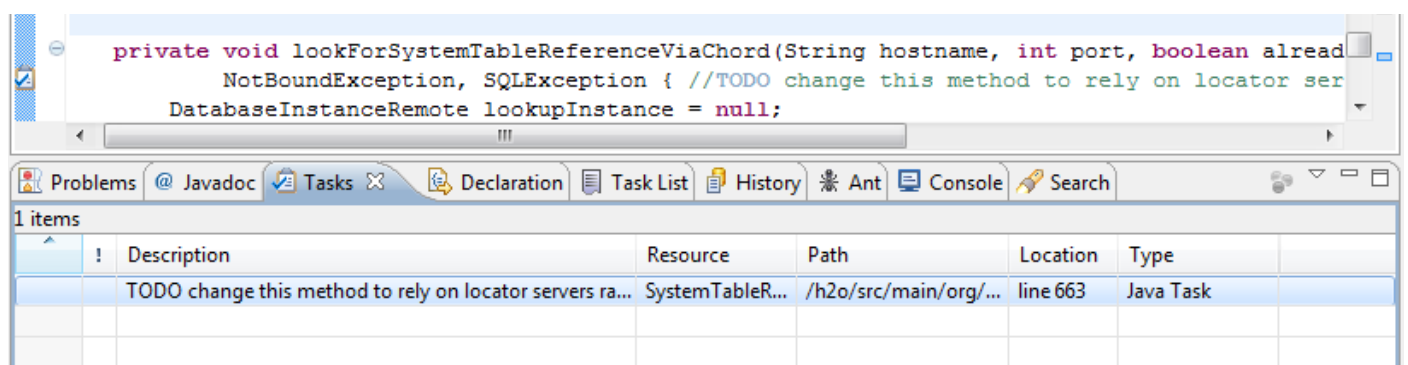


Figure 2: Tasks Pane

Comments will automatically be shown in the task pane if you start a comment with either `xxx` or `TODO`.

You can also create your own custom markers by clicking on the down arrow in the top right of the task pane and opening *Configure Contents*. Duplicate the `TODO` task you can then use your own custom tag by changing the description section.

PROBLEMS

The problems tab displays compiler warnings and errors.

Problems
 Javadoc
 Declaration
 Task List
 History
 Ant
 Console
 Search

2 errors, 1,522 warnings, 0 others (Filter matched 102 of 1524 items)

Description	Resource	Path	Location	Type
Errors (2 items) Collecton cannot be resolved to a type	Transaction.java	/h2o/src/main/org/...	line 148	Java Problem
The method commit(Collecton<CommitRes	Transaction.java	/h2o/src/main/org/...	line 133	Java Problem
Warnings (100 of 1522 items)				

Figure 3: Problems Pane

You can filter the contents of this tab so that it only displays local problems by clicking on the down arrow (in the top right) and opening [Configure Contents](#). Here you can change the scope of the problems shown for various types of errors and warnings.

PACKAGE EXPLORER VIEW

The package explorer displays projects in your workspace, with java files organized by package structure. You can choose to view packages flat or in a hierarchy.

To link the currently open java file with the contents of the package explorer select the 'link with editor' option.

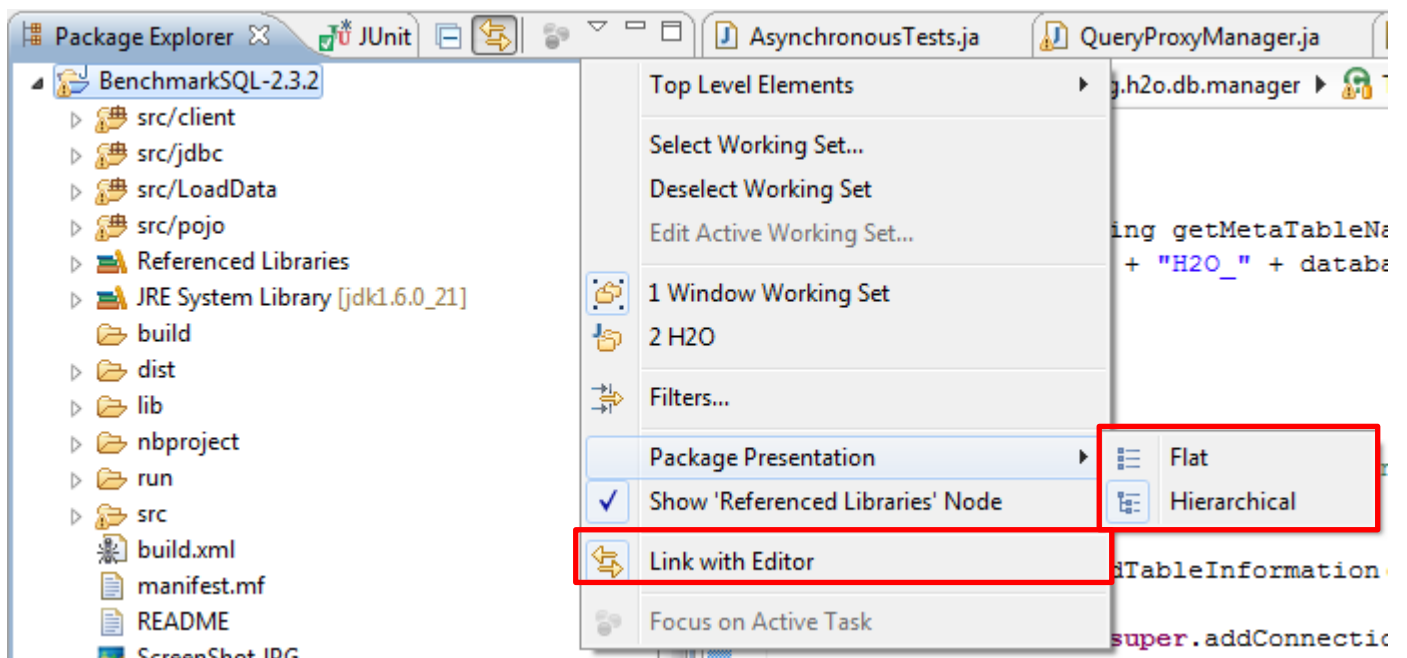


Figure 4: Package Explorer Pane

You can also filter the projects and classes that are shown in the package explorer to allow you to focus on only relevant classes. To do this click on the down arrow at the top of the package explorer and click on **SELECT WORKING SET** (shown above in Figure 4). You can then create new working sets that filter on various conditions. For example, you can filter certain projects or show on classes with specific debugging breakpoints. The menu with these options is illustrated below in Figure 5.

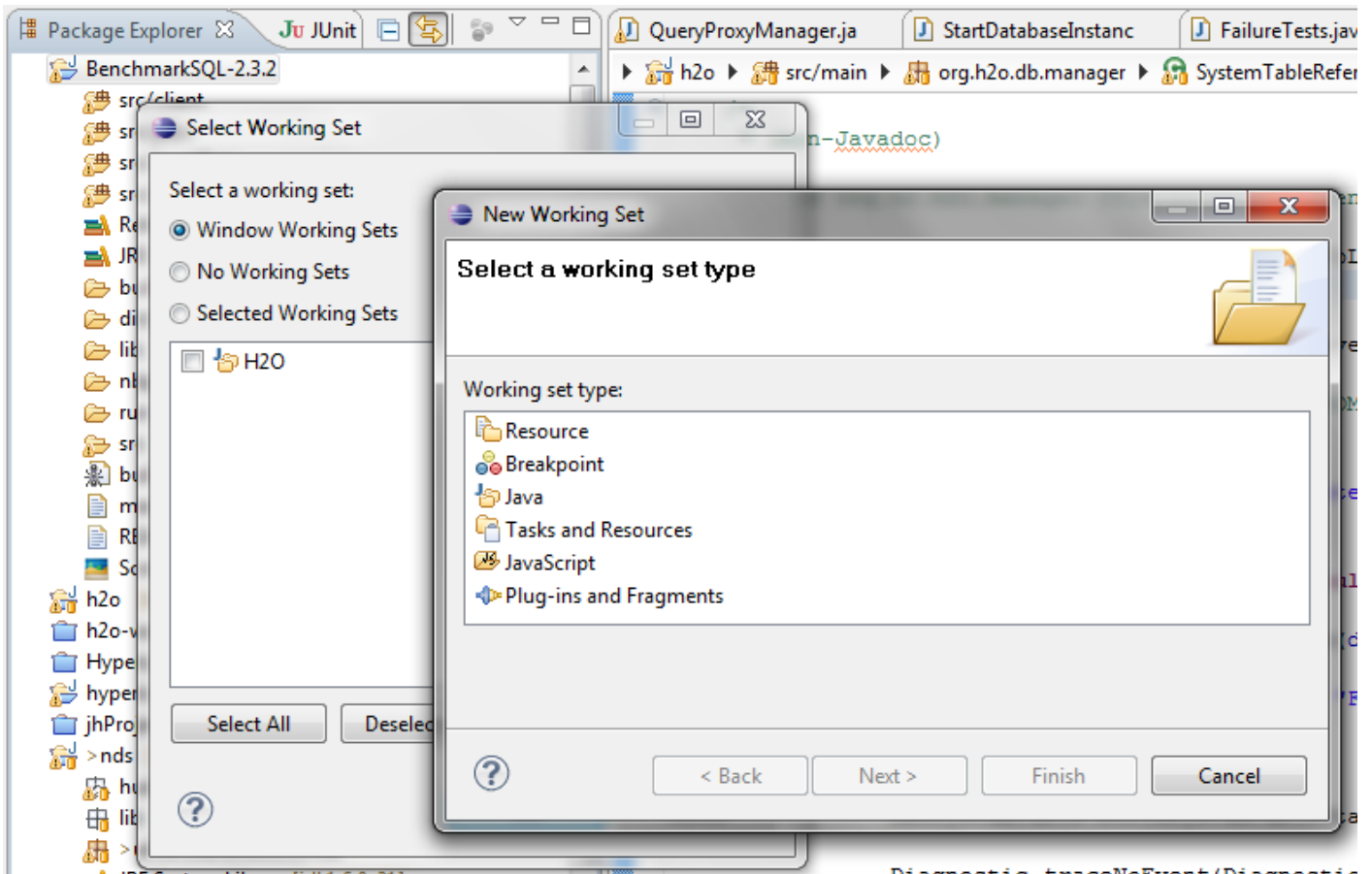


Figure 5: Working Sets

BREADCRUMBS

To enable the breadcrumb feature in eclipse, click on the  icon at the top of the workspace.

Breadcrumbs can be a quicker way of navigating between methods, classes, or packages in eclipse (as shown below).

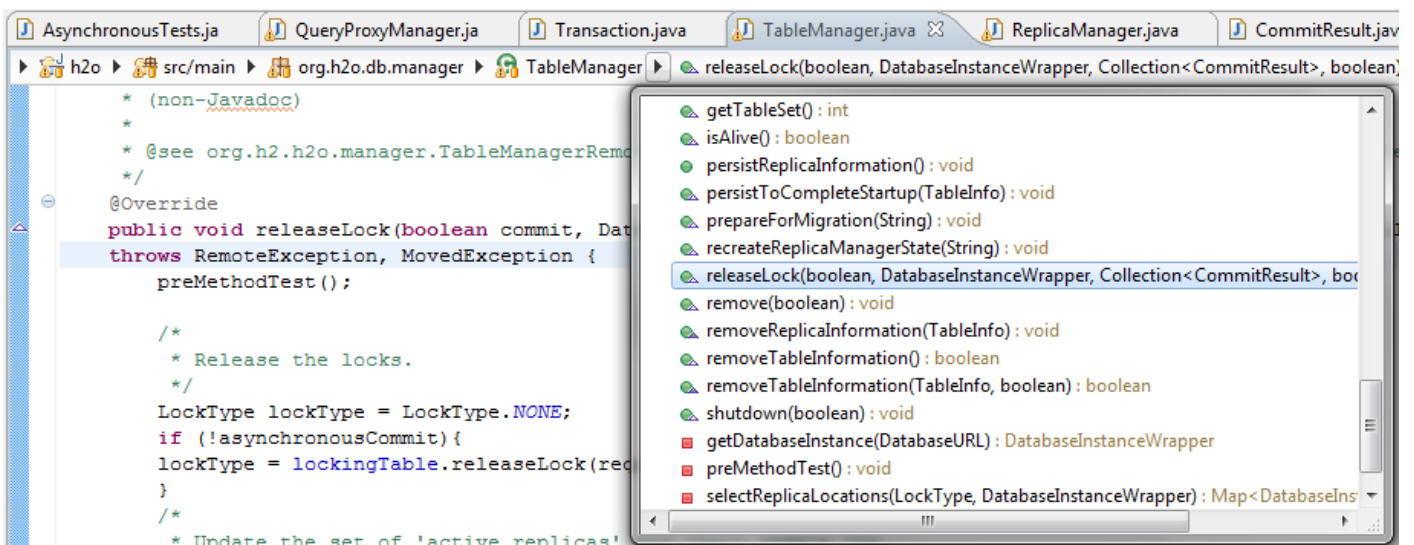


Figure 6: Breadcrumbs

MENU OPTIONS

The next couple of pages summarise some of the most useful menu options that eclipse provides.

THE SOURCE MENU

Used for cleaning up source files and generating code.

The screenshot shows the Eclipse Source menu with the following options and annotations:

- Toggle Comment** (Ctrl+7)
- Add Block Comment** (Ctrl+Shift+ /)
- Remove Block Comment** (Ctrl+Shift+ \)
- Generate Element Comment** (Alt+Shift+J)
- Shift Right**
- Shift Left**
- Correct Indentation** (Ctrl+I) - *Correct indentation and spacing for the current selection.*
- Format** (Ctrl+Shift+F) - *A much more comprehensive formatter that can be run on multiple classes at once.*
- Format Element**
- Add Import** (Ctrl+Shift+M)
- Organize Imports** (Ctrl+Shift+O) - *Gets rid of unused imports among other things.*
- Sort Members...**
- Clean Up...**
- Override/Implement Methods...**
- Generate Getters and Setters...**
- Generate Delegate Methods...**
- Generate toString()...**
- Generate hashCode() and equals()...** - *You need to implement Object.hashCode() and/or Object.equals to be able to add your own objects to hash-based data structures (hashmaps, hashsets) and perform object comparisons respectively, but you don't need to write them yourself.*
- Generate Constructor using Fields...**
- Generate Constructors from Superclass...** - *Avoid repetition in writing constructors and creating fields.*
- Surround With** (Alt+Shift+Z ▶)
- Externalize Strings...** - *This function places the contents of a specified set of strings in a properties file. It is a simple way of avoiding 'magic constants' in your code.*
- Find Broken Externalized Strings**

THE REFACTOR MENU

Used for making changes to code without breaking the program (changes are reflected everywhere so, for example, renaming a public field will change the field's name everywhere it is accessed).

The screenshot shows the Eclipse IDE's menu bar with 'Refactor' highlighted. The 'Refactor' menu is open, displaying a list of options. Red arrows point from descriptive text blocks to specific menu items:

- Rename all occurrences of a variable, method, class, or package.** Points to 'Rename...' (Alt+ Shift+ R).
- Create a constant field from a selected expression.** Points to 'Extract Constant...'.
- Create an interface for this class based on the current set of methods.** Points to 'Extract Interface...'.
- Add a selected expression as a parameter to a method call.** Points to 'Introduce Parameter...'.
- Move a method or field to a different location.** Points to 'Move...' (Alt+ Shift+ V).
- Wrap a number of method parameters into a single class.** Points to 'Introduce Parameter Object...'.
- Attempts to add types to generic interfaces/classes such as List and Map.** Points to 'Infer Generic Type Arguments...'.

The 'Refactor' menu options are as follows:

Menu Item	Shortcut
Rename...	Alt+ Shift+ R
Move...	Alt+ Shift+ V
Change Method Signature...	Alt+ Shift+ C
Extract Method...	Alt+ Shift+ M
Extract Local Variable...	Alt+ Shift+ L
Extract Constant...	
Inline...	Alt+ Shift+ I
Convert Local Variable to Field...	
Convert Anonymous Class to Nested...	
Move Type to New File...	
Extract Superclass...	
Extract Interface...	
Use Supertype Where Possible...	
Push Down...	
Pull Up...	
Extract Class...	
Introduce Parameter Object...	
Introduce Indirection...	
Introduce Factory...	
Introduce Parameter...	
Encapsulate Field...	
Generalize Declared Type...	
Infer Generic Type Arguments...	
Migrate JAR File...	
Create Script...	
Apply Script...	
History...	

THE NAVIGATE MENU

Used for navigating between classes.

The screenshot shows the Eclipse IDE's 'Navigate' menu. Red arrows point to specific menu items with explanatory text:

- Shows the super-classes and sub-classes of a given class.** Points to 'Open Type Hierarchy'.
- Jumps to the method which implements a method from an interface.** Points to 'Open Implementation'.
- Shows which methods call a given method, which methods call those methods, and so on.** Points to 'Open Call Hierarchy'.
- Jumps to the last line which was edited.** Points to 'Last Edit Location'.
- Jumps to the previous cursor location.** Points to 'Back'.

Menu Item	Shortcut
Go Into	
Go To	
Open Declaration	F3
Open Type Hierarchy	F4
Open Call Hierarchy	Ctrl+Alt+H
Open Implementation	
Open Super Implementation	
Open Attached Javadoc	Shift+F2
Open Type...	Ctrl+Shift+T
Open Type in Hierarchy...	Ctrl+Shift+H
Open Resource...	Ctrl+Shift+R
Open Task...	Ctrl+F12
Activate Task...	Ctrl+F9
Deactivate Task	Ctrl+Shift+F9
Show in Breadcrumb	Alt+Shift+B
Show In	Alt+Shift+W
Quick Context View	Ctrl+Alt+Shift+Right
Quick Outline	Ctrl+O
Quick Type Hierarchy	Ctrl+T
Next Annotation	Ctrl+.
Previous Annotation	Ctrl+,
Last Edit Location	Ctrl+Q
Go to Line...	Ctrl+L
Back	Alt+Left
Forward	Alt+Right

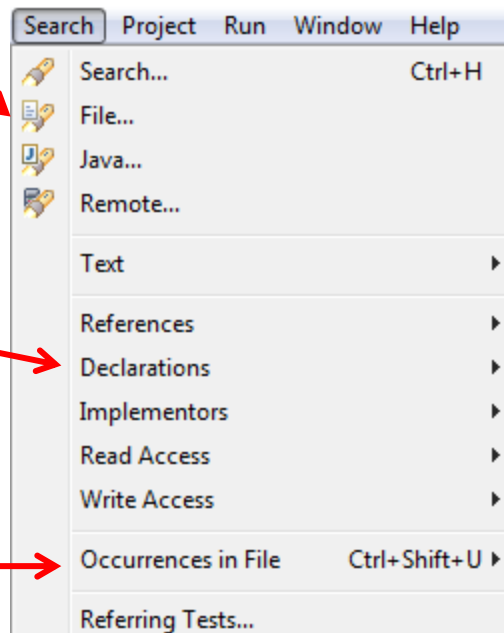
THE SEARCH MENU

Used for searching for text and specific operations in Java classes.

Search for text
occurring in all files in
your project.

Shows all locations
where a particular
type is declared.

Lists all occurrences of
a field or method in a
file.



Shows all references to a
particular class, method, or
field.

Shows all read accesses to a
particular field or variable.

THE RUN MENU

Used to start programs and debug them at runtime.

