CHEATSHEETSHERO

Eclipse IDE Cheat Sheet

A comprehensive cheat sheet for the Eclipse IDE, covering essential features, shortcuts, debugging, and refactoring techniques to boost developer productivity.

General Usage & Navigation

Basic Shortcuts

`Ctrl+Shift+R`	Open Resource (files, etc.)
`Ctrl+Shift+T`	Open Type (classes, interfaces)
`Ctrl+N`	New... (project, file, class, etc.)
`Ctrl+O`	Quick Outline (members of current file)
`Ctrl+E`	Open Editor (switch between open files)
`Ctrl+Shift+E`	Open Editor in Group
`Ctrl+F6`	Next Editor
`Ctrl+Shift+F6`	Previous Editor

Navigation

`F3` or `Ctrl+Click`	Open Declaration (go to definition)
`Ctrl+Alt+H`	Open Call Hierarchy
`Alt+Left Arrow`	Go Back (in navigation history)
`Alt+Right Arrow`	Go Forward (in navigation history)
`Ctrl+L`	Go to Line
`Ctrl+Shift+G`	Find References in Workspace

Working Sets

Working sets group projects for easier management. Useful for large projects.

- **Create:** `Window > Show View > Other... > Working Sets`
- **Add Projects:** Right-click on a working set, `Add Projects...`
- **Configure:** `Edit Working Set...`

Editing & Refactoring

Code Completion & Generation

`Ctrl+Space`	Code Completion (suggests methods, variables, etc.)
`Ctrl+1`	Quick Fix/Assist (suggests solutions to errors, generates code)
`Alt+Shift+S`	Source Menu (generate getters/setters, constructors, delegate methods, etc.)
`Ctrl+Shift+O`	Organize Imports (add/remove import statements)
`Ctrl+I`	Correct indentation
`Ctrl+Shift+F`	Format code

Refactoring

`Alt+Shift+R`	Rename (refactor variable, method, class names)
`Alt+Shift+M`	Extract Method (create a new method from selected code)
`Alt+Shift+C`	Change Method Signature
`Alt+Shift+V`	Move (move classes, packages, etc.)
`Alt+Shift+I`	Inline (replace method call with method body)
`Alt+Shift+L`	Extract Local Variable (create a new local variable from selected expression)

Code Templates

Code templates are predefined code snippets that can be inserted into your code.

- **Access:** `Window > Preferences > Java > Editor > Templates`
- **Examples:** `sysout` (for `System.out.println()`), `for` (for a basic for loop)

Debugging

Debugging Basics

`F5`	Step Into (go into the method)
`F6`	Step Over (go to the next line)
`F7`	Step Return (go out of the method)
`F8`	Resume (continue execution)
`Ctrl+Shift+B`	Toggle Breakpoint (add/remove breakpoint at current line)
`Ctrl+R`	Run to Line

Breakpoints

Breakpoints halt execution at specific locations. Right-click in the editor's left margin to set breakpoints.

- **Conditional Breakpoints:** Right-click on a breakpoint, `Breakpoint Properties...`, add a condition to only break when the condition is true.
- **Hit Count Breakpoints:** Break only after the breakpoint has been hit a certain number of times.

Inspecting Variables

During debugging, you can inspect variable values.

- **Variables View:** Shows all variables in the current scope.
- **Expressions View:** Allows you to evaluate custom expressions.
- **Hover:** Hover over a variable to see its value inline.

Search and Customization

Advanced Search

Eclipse provides powerful search capabilities.

- **File Search:** `Ctrl+H` (Text tab) - Search for text within files.
- **Java Search:** `Ctrl+H` (Java Search tab) - Search for types, methods, fields, etc.
- **Call Hierarchy Search:** Find all callers of a method.

Customization

Perspectives	Arrangements of views and editors tailored to specific tasks (e.g., Java, Debug). `Window > Perspective > Open Perspective > Other...`
Views	Reusable components that display information (e.g., Project Explorer, Outline). `Window > Show View > Other...`
Editors	Components for editing files. Eclipse supports various editors for different file types.
Key Bindings	Customize keyboard shortcuts. `Window > Preferences > General > Editors > Text Editors`
Themes	Change the look and feel of Eclipse. `Window > Preferences > General > Appearance`