MODERN IDE DEBUG, NAVIGATION & SHORTCUT CHEAT SHEET

Architecture Reference for IntelliJ, VS Code/VSCodium, Eclipse & Terminal Editors (2026)

1. Universal IDE Keyboard Shortcut Matrix

Keymap

Action	IntelliJ IDEA	VS Code / VSCodium	Eclipse
Command Palette	`Ctrl + Shift + A`	`Ctrl + Shift + P`	`Ctrl + 3`
Go to File	`Ctrl + Shift + N`	`Ctrl + P`	`Ctrl + Shift + R`
Go to Symbol / Class	`Ctrl + N / Alt+Home`	`Ctrl + T`	`Ctrl + Shift + T`
Duplicate / Delete Line	`Ctrl + D / Ctrl + Y`	`Shift+Alt+Down / Ctrl+Shift+K`	`Ctrl + Alt + Down / Ctrl + D`
Multiple Cursors	`Alt + Shift + Click`	`Alt + Click`	`Alt + Shift + A`
Recent Files	`Ctrl + E`	`Ctrl + Tab`	`Ctrl + E`

2. Professional Debugging Workflow

Debug Engine

Breakpoints: Click gutter or press `Ctrl + F8` / `F9` to halt execution.
Conditional Breakpoints: Right-click breakpoint to set expressions: `user.getId() == 42`.
Step Over: `F8` (IntelliJ) / `F10` (VS Code) — Advance line without descending into call.
Step Into: `F7` (IntelliJ) / `F11` (VS Code) — Enter subroutine execution scope.
Step Out: `Shift + F8` / `Shift + F11` — Finish current scope and return to caller.
Evaluate Expression: `Alt + F8` — Dynamically evaluate code in active runtime frame.
Watch Expressions: Right-click variable • "Add to Watches" for live continuous inspection.

3. Automated Refactoring Patterns

Clean Code

Operation	Shortcut (IntelliJ / VSC)	Architectural Impact
Rename Symbol	`Shift + F6` / `F2`	Safely updates all references across workspace.
Extract Method	`Ctrl + Alt + M`	Converts selected code block into new dedicated function.
Extract Variable	`Ctrl + Alt + V`	Assigns repeated expression to local immutable variable.
Change Signature	`Ctrl + F6`	Refactors parameter order and types across callers.
Inline Symbol	`Ctrl + Alt + N`	Replaces redundant intermediate variable with expression.

4. In-IDE Git & Version Control Operations

VCS Tools

Git Blame / Annotate: Right-click line gutter • "Annotate with Git Blame" to inspect commits.
Visual 3-Way Merge: Resolve conflicts visually across Local, Base, and Incoming panes.
Compare with Branch: Side-by-side visual diff against remote or tracking branches.
Shelve / Stash: Safely preserve unfinished work without polluting branch commit history.

5. Language Server Protocol (LSP) Architecture

Language Tooling

```
IDE (LSP Client) <-- JSON-RPC --> Language Server
Rust: rust-analyzer | Python: pyright / basedpyright | C++: clangd
Go: gopls | TypeScript: vtsls / tsserver | Java: eclipse.jdt.ls
```

Navigation Target	IntelliJ IDEA	VS Code / Zed
Go to Definition	`Ctrl + B` / `Ctrl + Click`	`F12` / `Ctrl + Click`
Find All Usages	`Alt + F7`	`Shift + F12`
Go to Type Definition	`Ctrl + Shift + B`	`Ctrl + F12`
Go to Implementation	`Ctrl + Alt + B`	`Ctrl + F12`

6. Memory & Performance Tuning

Profiling

JVM Heap Sizing (IntelliJ/Eclipse): Configure in `vmoptions` file:
`-Xms2048m -Xmx4096m -XX:+UseG1GC -XX:SoftRefLRUPolicyMSPerMB=50`
Directory Exclusion: Mark `node_modules`, `.venv`, `target`, and `dist` as Excluded to slash background CPU indexing by 70%.
Restart Language Server: Run `Restart Language Server` from Command Palette to flush memory leaks.
Tab Retention Limit: Restrict open tab limit to 10; rely on `Ctrl + E` (Recent Files) instead.

7. Advanced In-IDE Regex Patterns

Search & Replace

Global Search & Replace: `Ctrl + Shift + F` / `Ctrl + Shift + R`
Capturing Groups & Refactor:
Match: `public void test(\w+)\(\)` → Replace: `void should$1()` (referencing $1)
Line-Ending Matching: `;\s*$` — Pinpoint trailing semicolons and whitespace.
File Mask Inversion: `*.ts, !*.spec.ts` — Exclude test suites from bulk operations.

8. AI Hygiene, Security & Extensions

Best Practices

Domain	Recommended Protocol	Security & Speed Benefit
AI Assistance	Local models (Ollama / Continue.dev)	Zero intellectual property leakage.
Extensions	Audit and disable unused plugins	Reduces IDE cold startup by up to 60%.
Formatting	Format on Save / Pre-commit hooks	Eliminates stylistic debate in code reviews.
Integrated Terminal	Toggle via Alt+F12 / Ctrl+`	Preserves cognitive flow without task switching.