|  |  |
| --- | --- |
| Text | |
| Cmd+Home/End | Jump to start/end of file |
| Editing | |
| Ctrl+Space | Auto-complete |
| Opt+Shift+ Up/Down | Move lines up/down |
| Cmd+Opt+T | Surround code with… |
| Cmd+N | Auto-generate code menu |
| Cmd+P | Valid Parameters list |
| Cmd+J | Show available live templates |
| Ctrl+. | Insert code-completion selection with a training comma. |
| Ctrl+Space | Open code completion list. |
| Option+Enter | In a class’s declaration line – open the available context actions dialog (e.g. create a test) |
| Code Navigation | |
| Ctl+O | Navigate to class |
| Cmd+Shift+O | Open file dialog. You can use wildcards (\*) to search with partial names. |
| Cmd+B | Navigate to definition |
| Cmd+Opt+B | Navigate to implementation(s) of abstract function |
| Option+Space | Open definition in tooltip (without navigating) |
| Shift Shift | General search bar |
| Cmd + E | Open recent files |
| Ctl+ Up/Down | Use ⌃↑ and ⌃↓ keys to quickly move between methods in the editor. |
| Debugging | |
| Setting up smart-rules for stopping at breakpoints. | When stopped at a breakpoint in the IntelliJ debugger, hit Alt+Enter to pull up the “breakpoint intentions” menu. This menu contains some common options to apply to the breakpoint, such as “Stop only if called from…” and “Stop only in the class…”. These options automatically generate appropriate caller filters for the breakpoint. By excluding irrelevant callers, it’s much easier to pinpoint the source of a bug. |