

TMUX · SESSIONS (one per team)	
<code>tmux new -s team</code>	start named session
<code>tmux attach -t team</code>	re-attach (after SSH drop)
<code>tmux ls</code>	list sessions
<code>C-b d</code>	detach — agents keep running
<code>C-b s</code>	switch session (picker)
<code>C-b \$</code>	rename session

TMUX · WINDOWS (one per task)	
<code>C-b c</code>	new window
<code>C-b n / p</code>	next / previous window
<code>C-b 0..9</code>	jump to window N
<code>C-b w</code>	window picker (see all agents)
<code>C-b ,</code>	rename window (name = agent role)
<code>C-b &</code>	kill window

TMUX · PANES (where teammates live)	
<code>C-b %</code>	split vertical (side-by-side)
<code>C-b "</code>	split horizontal (stacked)
<code>C-b ←↑→↓</code>	move between teammate panes
<code>C-b o</code>	cycle to next pane
<code>C-b q (+ digit)</code>	show pane numbers, jump to one
<code>C-b z</code>	ZOOM pane — focus one teammate
<code>C-b x</code>	kill pane (dead agent)
<code>C-b !</code>	break pane into own window
<code>C-b { / }</code>	swap pane left / right
<code>C-b Space</code>	cycle pane layouts
<code>C-b M-↔</code>	resize pane by 5 (tap arrow to repeat)
<code>C-b :</code>	command prompt
<code>:setw synchronize-panes</code>	type into ALL panes at once

TMUX · QUALITY OF LIFE (~/.tmux.conf)	
<code>set -g mouse on</code>	click panes, wheel-scroll agent output
<code>setw -g mode-keys vi</code>	vi keys in copy mode
<code>history-limit 50000</code>	set -g — keep long agent logs
<code>kill-session -t team</code>	tmux ... — stop the whole team

TMUX · SCROLLBACK / COPY (read agent logs)	
<code>C-b [</code>	enter copy mode (scroll history)
<code>q</code>	quit copy mode
<code>C-u / C-d</code>	half-page up / down
<code>g / G</code>	top / bottom of history (vi keys)
<code>/text + n / N</code>	search down, next / prev match
<code>Space ... Enter</code>	start selection ... copy it
<code>C-b]</code>	paste tmux buffer

CLAUDE CODE · PROMPT KEYS	
<code>Shift+Tab</code>	cycle: normal → auto-accept → plan mode
<code>Esc</code>	interrupt Claude mid-task
<code>Esc Esc</code>	rewind — jump back & edit a message
<code>C-r</code>	expand full transcript view
<code>Up / Down</code>	browse prompt history
<code>Tab</code>	complete file path
<code>Shift+Enter or \ Enter</code>	newline without sending
<code>C-x C-e</code>	edit prompt in \$EDITOR (vim)
<code>C-v</code>	paste image from clipboard
<code>C-l</code>	clear screen (keeps context)
<code>C-c</code>	cancel input / current action
<code>C-d</code>	quit Claude Code

CLAUDE CODE · PREFIX CHARACTERS	
<code>/</code>	slash commands (/resume, /model, /clear...)
<code>@file</code>	mention file — pulls it into context
<code>!cmd</code>	run bash directly, output in context
<code># note</code>	append note to CLAUDE.md memory
<code>/vim</code>	vim-style editing in the input box
<code>/config</code>	settings (incl. teammate display)

CLAUDE CODE · SESSION / TEAM COMMANDS	
<code>/resume</code>	pick up a previous session
<code>/rewind</code>	restore code + chat checkpoint
<code>/compact</code>	compress context, keep working
<code>/clear</code>	wipe context for a fresh task
<code>/model</code>	switch model mid-session
<code>/agents</code>	manage subagent definitions
<code>/permissions</code>	edit allowed-tools rules

VIM · SURVIVE & MOVE	
<code>Esc / i / a</code>	normal mode / insert before / after
<code>:wq :q! ZZ</code>	save+quit / discard / save+quit
<code>h j k l</code>	left, down, up, right
<code>w / b / e</code>	next word / back / end of word
<code>0 ^ \$</code>	line start / first char / line end
<code>gg / G / 42G</code>	file top / bottom / line 42
<code>C-u / C-d</code>	half-page up / down
<code>f<c> / t<c> ;</code>	jump onto / before char, repeat

VIM · EDIT (CLAUDE.md, configs, commits)	
<code>x / dd / dw</code>	delete char / line / word
<code>ciw / ci" / ci(</code>	change word / inside quotes / parens
<code>yy / p / P</code>	yank line, paste after / before
<code>u / C-r / .</code>	undo / redo / repeat last change
<code>o / O</code>	new line below / above + insert
<code>v / V / C-v</code>	visual: char / line / block
<code>>> <<</code>	indent / outdent line

VIM · SEARCH & FILES	
<code>/text n / N</code>	search fwd, next / prev
<code>*</code>	search word under cursor
<code>:%s/old/new/g</code>	replace in whole file (add c to confirm)
<code>:e file / :w</code>	open file / save
<code>:vsp / :sp</code>	vertical / horizontal split
<code>C-w h j k l</code>	move between vim splits
<code>:qa</code>	quit all splits

VIM · SPELL CHECK	
<code>:set spell spelllang=en</code>	turn on (:set nospell = off)
<code>]s / [s</code>	jump to next / previous misspelling
<code>z= / lz=</code>	list suggestions / apply the first
<code>zg / zw</code>	mark word as good / as wrong

VIM · WHEN GIT OPENS VIM (EDITOR=vim)	
<code>i ... Esc :wq</code>	write commit message, finish
<code>dd + p / P</code>	rebase -i: reorder commit lines
<code>cw squash</code>	rebase -i: change pick → squash
<code>:cq</code>	exit non-zero — abort git operation

GOTCHAS & FLOW

C-b = tmux prefix. Claude Code also uses Ctrl+B (background tasks) — inside tmux press C-b C-b to send it through.
C-w = vim window key. tmux pane moves need the prefix first, so they never clash.
Team flow: C-b z zoom a teammate → read → C-b z back → C-b q jump by number → C-b d detach, work survives.