tig Text-mode git browser

START HERE

`tig`
Open the main view

`h`
Help, all bindings

`j` / `k`
Move down / up

`q`
Close view / quit

`Enter`
Open the commit diff

`Tab`
Switch split pane

`s`
Status, stage files

`/`
Search the view

main commit list → `Enter` **diff** split below → `Tab` **back to list** or I to maximize

CONCEPTS

Tig is a read-mostly browser on top of git: every view is the output of a git command.

Views stack: q returns to the previous view, Q quits everything at once.

Piped input opens the pager view, so git show | tig colorizes any git output.

Tig passes unknown options to git, so any git log revision or path filter works.

VIEWS

`m` Main view: commit list

`d` Diff view

`l` Log view with messages

`s` Status view

`c` Stage view

`b` Blame view

`r` Refs: branches and tags

`t` Directory tree view

`f` Blob (file content) view

`g` Grep view

`y` Stash view

`p` Pager view

VIEWS (cont.)

`h` Help view

`q` Close view / quit

`Q` Quit all views

`Tab` Next view

`R` Reload the view

`O` Maximize current view

NAVIGATION

`j` / `k` Cursor down / up

`Down` / `Up`
Move and update the child view
In a split view this selects the next commit in the parent and refreshes the diff.

`PgDown` / `Space` Page down

`PgUp` / `-` Page up

`Home` Jump to first line

`End` Jump to last line

`Enter` Open item / split diff
Context sensitive; in the diff view it scrolls one line.

`,` Move to parent
Tree: parent directory. Blame: blame of the parent commit.

`H` Go to the HEAD commit

`Insert` / `Delete`
Scroll view one line up / down

NAVIGATION (cont.)

`Left` / `Right`
Scroll view one column

`|` Scroll to first column

`e` Open file in editor

SEARCH AND PROMPT

`/` Search forward (regexp)

`?` Search backward

`n` Next match

`N` Previous match

`:` Open the command prompt

`:80`
Jump to line 80

`:2f12bcc`
Jump to a commit id

`:goto <rev>`
Jump to a revision or branch

`:!git log -p`
Run a command in the pager

`:save-display <file>`
Save the screen to a file

STAGING

`u`
Stage / unstage file or chunk
In status view stages the file; in stage view on a chunk line stages that chunk.

STAGING (cont.)

`1` Stage a single diff line

`@` Jump to next chunk

`!` Revert file to last commit
Discards unstaged changes in the file.

`C`
Commit (status) / cherry-pick (main)

`M`
Resolve conflict with mergetool

`[` Decrease diff context

`]` Increase diff context

`%` Toggle file filter on the diff

`W` Toggle whitespace in diffs

DISPLAY TOGGLES

`o` Open the option menu

`#` Toggle line numbers

`D` Cycle date display

`A` Cycle author display

`G` Toggle revision graph

`F` Toggle ref names

`X` Toggle commit id display

`Ctrl` + `L` Redraw the screen

`z` Stop background loading

COMMAND LINE

`tig`
Browse the current branch

`tig <file>`
History of a file

`tig blame <file>`
Open blame view

`tig status`
Open status view

`tig show <rev>`
Show one commit

`tig refs`
Open refs view

`tig stash`
Open stash view

`tig --all`
All branches and tags

`tig <a>..<b>`
Only commits in a range

`tig origin..HEAD`
Unpushed commits

`tig --since=1.month -- <path>`
Recent changes to a path

`tig -n20`
Limit commit count

`tig -- status`
Path named like a subcommand

`git show | tig`
Colorize piped git output

CONFIG AND BINDINGS

Browsing variables

Variable	Value
`%(commit)`	selected commit id
`%(file)`	selected file
`%(branch)`	selected branch
`%(head)`	viewed head, default HEAD
`%(prompt)`	ask for a value

A bind line is: bind <keymap> <key> <action>; keymaps include generic, main, diff, status and refs.

RECIPES

Amend the last commit (in ~/.tigrc)

```
bind generic + !git commit --amend
```

Copy the commit id to the clipboard

```
bind generic 9 @sh -c "echo -n %(commit) | xclip -selection c"
```

Review what will be pushed

```
tig origin..HEAD
# Enter on a commit to read the diff
# q to leave
```

Stage part of a file

```
tig status
# select the file, Enter for the stage view
# ] widen context, 1 stage a line, u a chunk
# C to commit
```