



CLI

CLI Reference

This page describes the current CLI behavior. If commands change, update this doc.

Command pages

[setup](#)

[onboard](#)

[configure](#)

[config](#)

[doctor](#)

[dashboard](#)

[reset](#)

[uninstall](#)

[update](#)

[message](#)

[agent](#)

[agents](#)

[acp](#)

[status](#)

[health](#)

[sessions](#)

[gateway](#)

logssystemmodelsmemorynodesdevicesnodeapprovalssandboxtuibrowsercrondnsdocshookswebhookspairingplugins (plugin commands)channelssecurityskillsvoicecall (plugin; if installed)

Global flags

`--dev` : isolate state under `~/.openclaw-dev` and shift default ports.

`--profile <name>` : isolate state under `~/.openclaw-<name>` .

`--no-color` : disable ANSI colors.

`--update` : shorthand for `openclaw update` (source installs only).



`-V` , `--version` , `-v` : print version and exit.

Output styling

ANSI colors and progress indicators only render in TTY sessions.

OSC-8 hyperlinks render as clickable links in supported terminals; otherwise we fall back to plain URLs.

`--json` (and `--plain` where supported) disables styling for clean output.

`--no-color` disables ANSI styling; `NO_COLOR=1` is also respected.

Long-running commands show a progress indicator (OSC 9;4 when supported).

Color palette

OpenClaw uses a lobster palette for CLI output.

`accent` (`#FF5A2D`): headings, labels, primary highlights.

`accentBright` (`#FF7A3D`): command names, emphasis.

`accentDim` (`#D14A22`): secondary highlight text.

`info` (`#FF8A5B`): informational values.

`success` (`#2FBF71`): success states.

`warn` (`#FFB020`): warnings, fallbacks, attention.

`error` (`#E23D2D`): errors, failures.

`muted` (`#8B7F77`): de-emphasis, metadata.

Palette source of truth: `src/terminal/palette.ts` (aka “lobster seam”).

Command tree



openclaw [--dev] [--profile <name>] <command>



- setup
- onboard
- configure
- config
 - get
 - set
 - unset
- doctor
- security
 - audit
- reset
- uninstall
- update
- channels
 - list
 - status
 - logs
 - add
 - remove
 - login
 - logout
- skills
 - list
 - info
 - check
- plugins
 - list
 - info
 - install
 - enable
 - disable
 - doctor
- memory
 - status
 - index
 - search
- message
- agent
- agents
 - list
 - add
 - delete



acp

status

health

sessions

gateway

call

health

status

probe

discover

install

uninstall

start

stop

restart

run

logs

system

event

heartbeat last|enable|disable

presence

models

list

status

set

set-image

aliases list|add|remove

fallbacks list|add|remove|clear

image-fallbacks list|add|remove|clear

scan

auth add|setup-token|paste-token

auth order get|set|clear

sandbox

list

recreate

explain

cron

status

list

add

edit

rm

enable



disable

runs

run

nodes

devices

node

run

status

install

uninstall

start

stop

restart

approvals

get

set

allowlist add|remove

browser

status

start

stop

reset-profile

tabs

open

focus

close

profiles

create-profile

delete-profile

screenshot

snapshot

navigate

resize

click

type

press

hover

drag


select

upload

fill

dialog

wait



```
evaluate
console
pdf
hooks
  list
info
check
enable
disable
install
update
webhooks
  gmail setup|run
pairing
  list
  approve
docs
dns
  setup
tui
```

Note: plugins can add additional top-level commands (for example `openclaw voicecall`).

Security

`openclaw security audit` – audit config + local state for common security foot-guns.

`openclaw security audit --deep` – best-effort live Gateway probe.

`openclaw security audit --fix` – tighten safe defaults and chmod state/config.

Plugins

Manage extensions and their config:

`openclaw plugins list` – discover plugins (use `--json` for machine output).



```
openclaw plugins info <id> - show details for a plugin.
```

```
openclaw plugins install <path|.tgz|npm-spec> - install a plugin (or add  
a plugin path to plugins.load.paths ).
```

```
openclaw plugins enable <id> / disable <id> - toggle plugins.entries.  
<id>.enabled .
```

```
openclaw plugins doctor - report plugin load errors.
```

Most plugin changes require a gateway restart. See [/plugin](#).

Memory

Vector search over MEMORY.md + memory/*.md :

```
openclaw memory status - show index stats.
```

```
openclaw memory index - reindex memory files.
```

```
openclaw memory search "<query>" - semantic search over memory.
```

Chat slash commands

Chat messages support `/...` commands (text and native). See [/tools/slash-commands](#).

Highlights:

```
/status for quick diagnostics.
```

```
/config for persisted config changes.
```

```
/debug for runtime-only config overrides (memory, not disk; requires  
commands.debug: true ).
```

Setup + onboarding

setup

Initialize config + workspace.

Options:



```
--workspace <dir> : agent workspace path (default
~/.openclaw/workspace ).
```

```
--wizard : run the onboarding wizard.
```

```
--non-interactive : run wizard without prompts.
```

```
--mode <local|remote> : wizard mode.
```

```
--remote-url <url> : remote Gateway URL.
```

```
--remote-token <token> : remote Gateway token.
```

Wizard auto-runs when any wizard flags are present (`--non-interactive` , `--mode` , `--remote-url` , `--remote-token`).

onboard

Interactive wizard to set up gateway, workspace, and skills.

Options:

```
--workspace <dir>
```

```
--reset (reset config + credentials + sessions + workspace before
wizard)
```

```
--non-interactive
```

```
--mode <local|remote>
```

```
--flow <quickstart|advanced|manual> (manual is an alias for advanced)
```

```
--auth-choice <setup-token|token|chutes|openai-codex|openai-api-
key|openrouter-api-key|ai-gateway-api-key|moonshot-api-key|kimi-code-api-
key|synthetic-api-key|venice-api-key|gemini-api-key|zai-api-
key|apiKey|minimax-api|minimax-api-lightning|opencode-zen|skip>
```

```
--token-provider <id> (non-interactive; used with --auth-choice token )
```

```
--token <token> (non-interactive; used with --auth-choice token )
```

```
--token-profile-id <id> (non-interactive; default: <provider>:manual )
```

```
--token-expires-in <duration> (non-interactive; e.g. 365d , 12h )
```

```
--anthropic-api-key <key>
```

```
--openai-api-key <key>
```



```
--openrouter-api-key <key>
--ai-gateway-api-key <key>
--moonshot-api-key <key>
--kimi-code-api-key <key>
--gemini-api-key <key>
--zai-api-key <key>
--minimax-api-key <key>
--opencode-zen-api-key <key>
--gateway-port <port>
--gateway-bind <loopback|lan|tailnet|auto|custom>
--gateway-auth <token|password>
--gateway-token <token>
--gateway-password <password>
--remote-url <url>
--remote-token <token>
--tailscale <off|serve|funnel>
--tailscale-reset-on-exit
--install-daemon
--no-install-daemon (alias: --skip-daemon )
--daemon-runtime <node|bun>
--skip-channels
--skip-skills
--skip-health
--skip-ui
--node-manager <npm|pnpm|bun> (pnpm recommended; bun not recommended for
Gateway runtime)
--json
```

configure

Interactive configuration wizard (models, channels, skills, gateway).



config

Non-interactive config helpers (get/set/unset). Running `openclaw config` with no subcommand launches the wizard.

Subcommands:

```
config get <path> : print a config value (dot/bracket path).  
config set <path> <value> : set a value (JSON5 or raw string).  
config unset <path> : remove a value.
```

doctor

Health checks + quick fixes (config + gateway + legacy services).

Options:

```
--no-workspace-suggestions : disable workspace memory hints.  
--yes : accept defaults without prompting (headless).  
--non-interactive : skip prompts; apply safe migrations only.  
--deep : scan system services for extra gateway installs.
```

Channel helpers

channels

Manage chat channel accounts (WhatsApp/Telegram/Discord/Google Chat/Slack/Mattermost (plugin)/Signal/iMessage/MS Teams).

Subcommands:

```
channels list : show configured channels and auth profiles.  
channels status : check gateway reachability and channel health ( --  
probe runs extra checks; use openclaw health or openclaw status --  
deep for gateway health probes).
```



Tip: `channels status` prints warnings with suggested fixes when it can detect common misconfigurations (then points you to `openclaw doctor`).

`channels logs` : show recent channel logs from the gateway log file.

`channels add` : wizard-style setup when no flags are passed; flags switch to non-interactive mode.

`channels remove` : disable by default; pass `--delete` to remove config entries without prompts.

`channels login` : interactive channel login (WhatsApp Web only).

`channels logout` : log out of a channel session (if supported).

Common options:

`--channel <name>` :

`whatsapp|telegram|discord|googlechat|slack|mattermost|signal|imessage|msteams`

`--account <id>` : channel account id (default `default`)

`--name <label>` : display name for the account

`channels login` options:

`--channel <channel>` (default `whatsapp` ; supports `whatsapp / web`)

`--account <id>`

`--verbose`

`channels logout` options:

`--channel <channel>` (default `whatsapp`)

`--account <id>`

`channels list` options:

`--no-usage` : skip model provider usage/quota snapshots (OAuth/API-backed only).

`--json` : output JSON (includes usage unless `--no-usage` is set).

`channels logs` options:



`--channel <name|all> (default all)`

`--lines <n> (default 200)`

`--json`

More detail: </concepts/oauth>

Examples:

```
openclaw channels add --channel telegram --account alerts --name "A
openclaw channels add --channel discord --account work --name "Work Bot" --token
openclaw channels remove --channel discord --account work --delete
openclaw channels status --probe
openclaw status --deep
```

skills

List and inspect available skills plus readiness info.

Subcommands:

`skills list` : list skills (default when no subcommand).

`skills info <name>` : show details for one skill.

`skills check` : summary of ready vs missing requirements.

Options:

`--eligible` : show only ready skills.

`--json` : output JSON (no styling).

`-v` , `--verbose` : include missing requirements detail.

Tip: use `npx clawhub` to search, install, and sync skills.

pairing

Approve DM pairing requests across channels.

Subcommands:



```
pairing list <channel> [--json]
```

```
pairing approve <channel> <code> [--notify]
```

```
>
```

webhooks gmail

Gmail Pub/Sub hook setup + runner. See [/automation/gmail-pubsub](#).

Subcommands:

```
webhooks gmail setup (requires --account <email> ; supports --project ,
--topic , --subscription , --label , --hook-url , --hook-token , --
push-token , --bind , --port , --path , --include-body , --max-bytes ,
--renew-minutes , --tailscale , --tailscale-path , --tailscale-target ,
--push-endpoint , --json )
```

```
webhooks gmail run (runtime overrides for the same flags)
```

dns setup

Wide-area discovery DNS helper (CoreDNS + Tailscale). See [/gateway/discovery](#).

Options:

```
--apply : install/update CoreDNS config (requires sudo; macOS only).
```

Messaging + agent

message

Unified outbound messaging + channel actions.

See: [/cli/message](#)

Subcommands:

```
message
```

```
send|poll|react|reactions|read|edit|delete|pin|unpin|pins|permissions|search
|timeout|kick|ban
```

```
message thread <create|list|reply>
```



```
message emoji <list|upload>
```

```
message sticker <send|upload>
```

```
message role <info|add|remove>
```

```
message channel <info|list>
```

```
message member info
```

```
message voice status
```

```
message event <list|create>
```

Examples:

```
openclaw message send --target +15555550123 --message "Hi"
```

```
openclaw message poll --channel discord --target channel:123 --poll-question  
"Snack?" --poll-option Pizza --poll-option Sushi
```

agent

Run one agent turn via the Gateway (or `--local` embedded).

Required:

```
--message <text>
```

Options:

```
--to <dest> (for session key and optional delivery)
```

```
--session-id <id>
```

```
--thinking <off|minimal|low|medium|high|xhigh> (GPT-5.2 + Codex models  
only)
```

```
--verbose <on|full|off>
```

```
--channel
```

```
<whatsapp|telegram|discord|slack|mattermost|signal|imessage|msteams>
```

```
--local
```

```
--deliver
```

```
--json
```



`--timeout <seconds>`

agents

Manage isolated agents (workspaces + auth + routing).

agents list

List configured agents.

Options:

`--json`

`--bindings`

agents add [name]

Add a new isolated agent. Runs the guided wizard unless flags (or `--non-interactive`) are passed; `--workspace` is required in non-interactive mode.

Options:

`--workspace <dir>`

`--model <id>`

`--agent-dir <dir>`

`--bind <channel[:accountId]>` (repeatable)

`--non-interactive`

`--json`

Binding specs use `channel[:accountId]`. When `accountId` is omitted for WhatsApp, the default account id is used.

agents delete <id>

Delete an agent and prune its workspace + state.

Options:



`--force`

`--json`

>

acp

Run the ACP bridge that connects IDEs to the Gateway.

See [acp](#) for full options and examples.

status

Show linked session health and recent recipients.

Options:

`--json`

`--all` (full diagnosis; read-only, pasteable)

`--deep` (probe channels)

`--usage` (show model provider usage/quota)

`--timeout <ms>`

`--verbose`

`--debug` (alias for `--verbose`)

Notes:

Overview includes Gateway + node host service status when available.

Usage tracking

OpenClaw can surface provider usage/quota when OAuth/API creds are available.

Surfaces:

`/status` (adds a short provider usage line when available)

`openclaw status --usage` (prints full provider breakdown)

macOS menu bar (Usage section under Context)

Notes:



Data comes directly from provider usage endpoints (no estimates).

Providers: Anthropic, GitHub Copilot, OpenAI Codex OAuth, plus Gemini CLI/Antigravity when those provider plugins are enabled.

If no matching credentials exist, usage is hidden.

Details: see [Usage tracking](#).

health

Fetch health from the running Gateway.

Options:

- `--json`
- `--timeout <ms>`
- `--verbose`

sessions

List stored conversation sessions.

Options:

- `--json`
- `--verbose`
- `--store <path>`
- `--active <minutes>`

Reset / Uninstall

reset

Reset local config/state (keeps the CLI installed).

Options:



```
--scope <config|config+creds+sessions|full>
```

```
--yes
```

```
--non-interactive
```

```
--dry-run
```

Notes:

```
--non-interactive requires --scope and --yes .
```

uninstall

Uninstall the gateway service + local data (CLI remains).

Options:

```
--service
```

```
--state
```

```
--workspace
```

```
--app
```

```
--all
```

```
--yes
```

```
--non-interactive
```

```
--dry-run
```

Notes:

```
--non-interactive requires --yes and explicit scopes (or --all ).
```

Gateway

gateway

Run the WebSocket Gateway.

Options:



```
--port <port>
--bind <loopback|tailnet|lan|auto|custom>
--token <token>
--auth <token|password>
--password <password>
--tailscale <off|serve|funnel>
--tailscale-reset-on-exit
--allow-unconfigured
--dev
--reset (reset dev config + credentials + sessions + workspace)
--force (kill existing listener on port)
--verbose
--claude-cli-logs
--ws-log <auto|full|compact>
--compact (alias for --ws-log compact )
--raw-stream
--raw-stream-path <path>
```

gateway service

Manage the Gateway service (launchd/systemd/schtasks).

Subcommands:

```
gateway status (probes the Gateway RPC by default)
gateway install (service install)
gateway uninstall
gateway start
gateway stop
gateway restart
```

Notes:



`gateway status` probes the Gateway RPC by default using the service's resolved port/config (override with `--url/--token/--password`).

`gateway status` supports `--no-probe` , `--deep` , and `--json` for scripting.

`gateway status` also surfaces legacy or extra gateway services when it can detect them (`--deep` adds system-level scans). Profile-named OpenClaw services are treated as first-class and aren't flagged as "extra".

`gateway status` prints which config path the CLI uses vs which config the service likely uses (service env), plus the resolved probe target URL.

`gateway install|uninstall|start|stop|restart` support `--json` for scripting (default output stays human-friendly).

`gateway install` defaults to Node runtime; bun is **not recommended** (WhatsApp/Telegram bugs).

`gateway install` options: `--port` , `--runtime` , `--token` , `--force` , `--json` .

logs

Tail Gateway file logs via RPC.

Notes:

TTY sessions render a colorized, structured view; non-TTY falls back to plain text.

`--json` emits line-delimited JSON (one log event per line).

Examples:

```
openclaw logs --follow
openclaw logs --limit 200
openclaw logs --plain
openclaw logs --json
openclaw logs --no-color
```



`gateway <subcommand>`

Gateway CLI helpers (use `--url` , `--token` , `--password` , `--timeout` , `--expect-final` for RPC subcommands).

Subcommands:

```
gateway call <method> [--params <json>]
gateway health
gateway status
gateway probe
gateway discover
gateway install|uninstall|start|stop|restart
gateway run
```

Common RPCs:

```
config.apply (validate + write config + restart + wake)
config.patch (merge a partial update + restart + wake)
update.run (run update + restart + wake)
```

Tip: when calling `config.set` / `config.apply` / `config.patch` directly, pass `baseHash` from `config.get` if a config already exists.

Models

See </concepts/models> for fallback behavior and scanning strategy.

Preferred Anthropic auth (setup-token):

```
claude setup-token
openclaw models auth setup-token --provider anthropic
openclaw models status
```



models (root)

`openclaw models` is an alias for `models status` .

Root options:



```
--status-json  (alias for  models status --json )
--status-plain (alias for  models status --plain )
```

models list

Options:

```
--all
--local
--provider <name>
--json
--plain
```

models status

Options:

```
--json
--plain
--check  (exit 1=expired/missing, 2=expiring)
--probe  (live probe of configured auth profiles)
--probe-provider <name>
--probe-profile <id>  (repeat or comma-separated)
--probe-timeout <ms>
--probe-concurrency <n>
--probe-max-tokens <n>
```

Always includes the auth overview and OAuth expiry status for profiles in the auth store. `--probe` runs live requests (may consume tokens and trigger rate limits).

models set <model>



```
Set agents.defaults.model.primary .
```

>

models set-image <model>

```
Set agents.defaults.imageModel.primary .
```

models aliases list|add|remove

Options:

```
list : --json , --plain
```

```
add <alias> <model>
```

```
remove <alias>
```

models fallbacks list|add|remove|clear

Options:

```
list : --json , --plain
```

```
add <model>
```

```
remove <model>
```

```
clear
```

models image-fallbacks list|add|remove|clear

Options:

```
list : --json , --plain
```

```
add <model>
```

```
remove <model>
```

```
clear
```

models scan

Options:



```
--min-params <b>
--max-age-days <days>
--provider <name>
--max-candidates <n>
--timeout <ms>
--concurrency <n>
--no-probe
--yes
--no-input
--set-default
--set-image
--json
```

models auth add|setup-token|paste-token

Options:

```
add : interactive auth helper
setup-token : --provider <name> (default anthropic ), --yes
paste-token : --provider <name> , --profile-id <id> , --expires-in
               <duration>
```

models auth order get|set|clear

Options:

```
get : --provider <name> , --agent <id> , --json
set : --provider <name> , --agent <id> , <profileIds...>
clear : --provider <name> , --agent <id>
```



system event

Enqueue[^] a system event and optionally trigger a heartbeat (Gateway RPC).

Required:

`--text <text>`

Options:

`--mode <now|next-heartbeat>`

`--json`

`--url , --token , --timeout , --expect-final`

system heartbeat last|enable|disable

Heartbeat controls (Gateway RPC).

Options:

`--json`

`--url , --token , --timeout , --expect-final`

system presence

List system presence entries (Gateway RPC).

Options:

`--json`

`--url , --token , --timeout , --expect-final`

Cron

Manage scheduled jobs (Gateway RPC). See [/automation/cron-jobs.](#)

Subcommands:



```
cron status [--json]
```

```
cron list [--all] [--json] (table output by default; use --json for
raw)
```

```
cron add (alias: create ; requires --name and exactly one of --at |
--every | --cron , and exactly one payload of --system-event | --
message )
```

```
cron edit <id> (patch fields)
```

```
cron rm <id> (aliases: remove , delete )
```

```
cron enable <id>
```

```
cron disable <id>
```

```
cron runs --id <id> [--limit <n>]
```

```
cron run <id> [--force]
```

All cron commands accept --url , --token , --timeout , --expect-final .

Node host

node runs a **headless node host** or manages it as a background service. See [openclaw node](#) .

Subcommands:

```
node run --host <gateway-host> --port 18789
```

```
node status
```

```
node install [--host <gateway-host>] [--port <port>] [--tls] [--tls-
fingerprint <sha256>] [--node-id <id>] [--display-name <name>] [--runtime
<node|bun>] [--force]
```

```
node uninstall
```

```
node stop
```

```
node restart
```

Nodes

 `nodes` talks to the Gateway and targets paired nodes. See [/nodes](#).

Common options:

```
>
--url , --token , --timeout , --json
```

Subcommands:

```
nodes status [--connected] [--last-connected <duration>]

nodes describe --node <id|name|ip>

nodes list [--connected] [--last-connected <duration>]

nodes pending

nodes approve <requestId>

nodes reject <requestId>

nodes rename --node <id|name|ip> --name <displayName>

nodes invoke --node <id|name|ip> --command <command> [--params <json>] [--
invoke-timeout <ms>] [--idempotency-key <key>]

nodes run --node <id|name|ip> [--cwd <path>] [--env KEY=VAL] [--command-
timeout <ms>] [--needs-screen-recording] [--invoke-timeout <ms>]
<command...> (mac node or headless node host)

nodes notify --node <id|name|ip> [--title <text>] [--body <text>] [--sound
<name>] [--priority <passive|active|timeSensitive>] [--delivery
<system|overlay|auto>] [--invoke-timeout <ms>] (mac only)
```

Camera:

```
nodes camera list --node <id|name|ip>

nodes camera snap --node <id|name|ip> [--facing front|back|both] [--device-
id <id>] [--max-width <px>] [--quality <0-1>] [--delay-ms <ms>] [--invoke-
timeout <ms>]

nodes camera clip --node <id|name|ip> [--facing front|back] [--device-id
<id>] [--duration <ms|10s|1m>] [--no-audio] [--invoke-timeout <ms>]
```

Canvas + screen:



```
nodes canvas snapshot --node <id|name|ip> [--format png|jpg|jpeg] [--max-
width <px>] [--quality <0-1>] [--invoke-timeout <ms>]
```

```
nodes canvas present --node <id|name|ip> [--target <urlOrPath>] [--x <px>]
--y <px>] [--width <px>] [--height <px>] [--invoke-timeout <ms>]
```

```
nodes canvas hide --node <id|name|ip> [--invoke-timeout <ms>]
```

```
nodes canvas navigate <url> --node <id|name|ip> [--invoke-timeout <ms>]
```

```
nodes canvas eval [<js>] --node <id|name|ip> [--js <code>] [--invoke-timeout
<ms>]
```

```
nodes canvas a2ui push --node <id|name|ip> (--jsonl <path> | --text <text>)
[--invoke-timeout <ms>]
```

```
nodes canvas a2ui reset --node <id|name|ip> [--invoke-timeout <ms>]
```

```
nodes screen record --node <id|name|ip> [--screen <index>] [--duration
<ms|10s>] [--fps <n>] [--no-audio] [--out <path>] [--invoke-timeout <ms>]
```

Location:

```
nodes location get --node <id|name|ip> [--max-age <ms>] [--accuracy
<coarse|balanced|precise>] [--location-timeout <ms>] [--invoke-timeout
<ms>]
```

Browser

Browser control CLI (dedicated Chrome/Brave/Edge/Chromium). See [openclaw browser](#) and the [Browser tool](#).

Common options:

```
--url , --token , --timeout , --json
```

```
--browser-profile <name>
```

Manage:

```
browser status
```

```
browser start
```

```
browser stop
```



browser reset-profile

browser tabs

browser open <url>

browser focus <targetId>

browser close [targetId]

browser profiles

browser create-profile --name <name> [--color <hex>] [--cdp-url <url>]

browser delete-profile --name <name>

Inspect:

browser screenshot [targetId] [--full-page] [--ref <ref>] [--element <selector>] [--type png|jpeg]

browser snapshot [--format aria|ai] [--target-id <id>] [--limit <n>] [--interactive] [--compact] [--depth <n>] [--selector <sel>] [--out <path>]

Actions:

browser navigate <url> [--target-id <id>]

browser resize <width> <height> [--target-id <id>]

browser click <ref> [--double] [--button <left|right|middle>] [--modifiers <csv>] [--target-id <id>]

browser type <ref> <text> [--submit] [--slowly] [--target-id <id>]

browser press <key> [--target-id <id>]

browser hover <ref> [--target-id <id>]

browser drag <startRef> <endRef> [--target-id <id>]

browser select <ref> <values...> [--target-id <id>]

browser upload <paths...> [--ref <ref>] [--input-ref <ref>] [--element <selector>] [--target-id <id>] [--timeout-ms <ms>]

browser fill [--fields <json>] [--fields-file <path>] [--target-id <id>]

browser dialog --accept|--dismiss [--prompt <text>] [--target-id <id>] [--timeout-ms <ms>]



```
browser wait [--time <ms>] [--text <value>] [--text-gone <value>] [--target-id <id>]
```

```
browser evaluate --fn <code> [--ref <ref>] [--target-id <id>]
```

```
browser console [--level <error|warn|info>] [--target-id <id>]
```

```
browser pdf [--target-id <id>]
```

Docs search

```
docs [query...]
```

Search the live docs index.

TUI

```
tui
```

Open the terminal UI connected to the Gateway.

Options:

```
--url <url>
```

```
--token <token>
```

```
--password <password>
```

```
--session <key>
```

```
--deliver
```

```
--thinking <level>
```

```
--message <text>
```

```
--timeout-ms <ms> (defaults to agents.defaults.timeoutSeconds )
```

```
--history-limit <n>
```

< Bun (Experimental)

setup >



Powered by mintlify

>