CHEATSHEET

oh-my-symphony

One Kanban board, eight AI coding agents, one ticket per `git worktree`. Everything you type, on two pages.

한국어 Tutorial PDF github.com/cskwork/oh-my-symphony

Lifecycle in one line

`install` → `project create / add` → `doctor` → `tui / service` → `ticket` → `agent runs` → `Human Review` → `Done`

Install & first run

```
git clone https://github.com/cskwork/oh-my-symphony.git && cd oh-my-symphony
python3 -m venv .venv && source .venv/bin/activate
python -m pip install -e ".[dev]"
symphony project create "My App" --path ../my-app && cd ../my-app
symphony doctor ./WORKFLOW.md
symphony tui ./WORKFLOW.md
```

CLI

Command	What it does
`symphony tui`	open the Kanban TUI (same as --tui)
`symphony doctor`	preflight a WORKFLOW.md before launching
`symphony board`	manage the file-based Kanban board (init/ls/new/mv/update/show/graph)
`symphony service`	run the orchestrator as a managed background service
`symphony project`	register and run independent project services
`symphony hub`	serve the multi-project hub
`symphony runs`	print recent run-registry attempts
`symphony release`	validate application release evidence
`symphony wiki-sweep`	lint docs/llm-wiki for duplicate slugs and orphans

Most-used invocation	Effect
`symphony board init ./kanban`	Create the file board with a sample `DEMO-001.md`
`symphony board new TASK-1 "title"`	Create `kanban/TASK-1.md` ; flags: `--priority 2 --labels backend,test --description "…" --blocked-by ID --agent-kind codex`
`symphony board ls --state Todo`	List tickets, optionally by state
`symphony board mv TASK-1 Blocked`	Force a state change; re-evaluated on the next poll tick
`symphony service start ./WORKFLOW.md --port 9999`	Background service + web board; also `status` / `logs` / `restart` / `stop`
`symphony runs ./WORKFLOW.md --limit 5`	Recent run-registry attempts (`--issue ID` to filter)

TUI keys

● running (turn N, tokens) ↻ retry queued (retry #N, last error) ✓ done this session
`?` opens this list inside the TUI

Board

`q`	quit (asks twice while workers run)
`r`	refresh + re-poll the tracker
`?`	this help
`L`	cycle TUI + doc language
`s`	run statistics
`/`	filter cards by id / title / label
`esc`	close filter, reset zoom

Navigate

`tab` / `shift+tab`	focus next / previous card
`j` / `k`, `↓` / `↑`	scroll the focused lane
`g` / `G`, `home` / `end`	jump to top / bottom
`space` / `pgdn`, `b` / `pgup`	page down / up
`enter`	open the full-detail modal
`p`	show / hide the detail pane
`]` / `[`	focus the detail pane / back to the board

Layout

`1-9` / `0`	zoom that lane / reset zoom
`t` / `T`	next / previous page of lanes
`+` / `-`	more / fewer lanes per page
`d`	toggle compact / rich cards

Ticket

`n`	new ticket (file board only)
`e`	edit the focused ticket (file board only)
`a`	archive the focused Done card
`c`	confirm the focused Human Review card as Done
`S`	skip Document for the focused card
`P`	pause / resume the focused running worker

Web board

Item	Detail
URL	`http://127.0.0.1:<port>/` (`server.port` or `--port`); loopback only
Pause / Resume	Buttons on running cards hold / release that worker
Header refresh	Triggers an orchestrator `poll + reconcile`
Branch dropdowns	Real local git branches for `agent.feature_base_branch` (start of new branches) and `agent.auto_merge_target_branch` (Done merge target)
Bearer token	`export SYMPHONY_API_TOKEN="$(python -c 'import secrets; print(secrets.token_urlsafe(32))')"` before launch; the web app prompts after its first `401` . Optional

WORKFLOW.md minimum

```
---
tracker:
  kind: file
  board_root: ./kanban
  active_states: [Todo, "In Progress", Verify, Document]
  terminal_states: ["Human Review", Done, Blocked, Archive]
agent:
  kind: claude          # codex | claude | gemini | agy | kiro | opencode | pi | prime-agent
  max_concurrent_agents: 1
server:
  port: 9999
---
```

Key	What it changes
`agent.kind`	Default backend CLI; a ticket can override with `agent_kind:` frontmatter
`agent.max_concurrent_agents`	How many workers run at once (each in its own worktree)
`tracker.active_states`	Columns the orchestrator dispatches agents into
`server.port`	Web board + JSON API port; drop the block to disable HTTP
`agent.feature_base_branch`	Start point of `symphony/<ID>` branches; empty = current host branch
`agent.auto_merge_target_branch`	Branch the Verify gate merges into; empty = the branch the feature was cut from

Ticket states

State	What happens there
Todo	Triage; route to In Progress (a ticket with Acceptance Criteria skips the model turn)
In Progress	Plan + TDD implementation + self-critique on the `symphony/<ID>` branch
Verify	Review + QA + Merge Gate; feature branch merged into the target before Document
Document	Docs + wiki write-back; Done unless intervention (`S` skips it)
Human Review	Manual intervention or explicit review before Done; confirm with `c` in the TUI or from the web board
Done	Verified complete; one `<ID>: <title>` commit; `a` archives

When something breaks

doctor FAIL line — fix what the line names (placeholder clone URL → worktree default or `: noop` ; missing CLI → put it on `$PATH`), re-run `symphony doctor ./WORKFLOW.md` . Exit `1` = a FAIL, `2` = WORKFLOW.md would not load.
Worker exited, card shows ↻ — retries back off up to `agent.max_retries` ; last error in `symphony runs ./WORKFLOW.md --limit 5` and `symphony service logs ./WORKFLOW.md` .
Agent silent for minutes — no progress inside `stall_timeout_ms` cancels the worker; widen one lane: `agent.stall_timeout_ms_by_state: {Verify: 900000}` ; tail `log/symphony.log` .
Port in use — doctor reports it; `--port N` overrides `server.port` ; `symphony service status ./WORKFLOW.md` shows who holds it.
TUI exits silently — it needs a real TTY: run it in a foreground terminal, or go headless (`symphony ./WORKFLOW.md`) and read `WORKFLOW-PROGRESS.md` .

oh-my-symphony 0.23.0 · Apache-2.0 · generated from README on the same commit