BEHAVIOUR & APPEARANCE · ONE SYSTEM

BINNACLE.

The house style for everything I build — how it steers, and how it looks doing it.



PART I Helm — how it behaves	PART II Instrument panel — how it looks	PART III Voice — how it speaks
PART V Workbench — how it is worked in	SHIPS WITH tokens.css · 4 modes	REFERENCE BUILD Binnacle
REVISION living · v1.14		

00 How to use this

> **The name.**
>
> A `binnacle` is the housing that stands before a ship's wheel and holds the compass — the instrument the helm steers by. This system takes that role: the housing that keeps every project oriented, holding the guiding instruments — **how it behaves, how it looks, how it speaks** — right at the wheel.

Binnacle is a starting point, not a cage. Drop it onto a new project and you inherit a working point of view: a behavioural stance (**Helm**), a visual language (**Instrument panel**), and a voice — plus a `tokens.css` that makes all of it real in about a minute.

The rule for adapting it: **keep the behaviour and the bones, spend novelty on the signature.** The Helm tenets never change. The type roles and base tokens carry across untouched. The one thing you redesign per project is its `signature element` — the single memorable component drawn from that project's own subject. Everything else stays quiet so the signature can be loud.

The names — Binnacle, Helm, Instrument panel — are swappable. They come from a nautical naming register (see §09); rename freely for your own project.

> **This document is alive.**
>
> Binnacle is never "finished" — it is the accumulated house position, and it moves when a project teaches it something. Its roster status stays `BUILDING` on purpose. The source for these pages is `Binnacle.html`; edit it and reprint rather than editing a PDF.

Revision log

REV	DATE	WHAT MOVED, AND WHICH PROJECT TAUGHT IT
v1.0	2026-07-18	initial system: Helm tenets, instrument panel, voice, tokens, glyphs, modes
v1.1	2026-07-21	a first Compose port: project glyph **extension sheets** (§12) · *every category that loses its colour gains a glyph* (§12) · map-into-host porting pattern + *host data-colour modes yield to the skin* (§16) · §03 mono boundary (small data mono, hero numbers stay UI-tabular) · warning rides amber (§02) · bundled faces carry their OFL notices (§03 / Helm 2)
v1.2	2026-07-21	Voice audit of a heavy-text port: *one string is one sentence* — never split a user sentence across a localised resource and a hard-coded tail (§08) · *sentence case is the rule, a register name is the exception* (§08/§09) · *never starve a label beside a wide control* — stack before you let a phone char-break it (§15 layout)
v1.3	2026-07-24	a native media-player port named & planned — added to the register (§09), the roster, and the signature table (§06) · *two tools in one subject split the signature axis* (§06): one tool's spine runs within a book, another's manifest runs across the shelf, so the siblings never read as one app

REV	DATE	WHAT MOVED, AND WHICH PROJECT TAUGHT IT
v1.4	2026-07-31	a media-player Compose port, glyph audit: *a Unicode/emoji glyph is off-contract* (§12) — ship the vector set tinted by `currentColor`, never a text symbol (a record ▯ rendered as a colour emoji) · *universal glyphs belong in core, not a project sheet* (§12): the port's shared UI/media marks (chevron, chapter step, bookmark, pencil, record, list) were promoted into `binnacle-glyphs.svg`; a project sheet keeps only `crate` · glyph gallery folded into §12 as a live specimen (standalone preview retired) · pages relaid to **US Letter** for clip-free PDF export
v1.5	2026-07-31	*one glyph sheet, no extensions* (§12): the rule flips — every glyph folds into the canonical `binnacle-glyphs.svg`, named for what it depicts, not the project that first drew it. a fitness port's health/activity set (bolt, chevrons, crescent, spike, heart, ticks, droplet, waves, dots, flame, dumbbell, band) and a media port's `crate` merged in; the per-project sheets retired.
v1.6	2026-08-01	This document promoted to the **living Binnacle Reference** — the reference build itself, drawn from the tokens and laid out by its own rules (Reference build · Helm 1–4). Consistency repairs: the glyph set reconciled to one canonical sprite plus standalone SVGs (PySide loader path fixed, `queued` glyph matched to the sprite). Auto-pagination for clip-free US Letter print — sections pack, a heading keeps with its content, grids and tables move whole or fragment without orphans, the running head follows the section, and `data-break` forces a fresh page where wanted (§09). Editorial pass: tighter vertical rhythm, provenance moved from the body into this log and the roster's lessons line (§08), and *structure survives the break* added (§04). §07 Motion & restraint expanded to a full section — what earns motion, the `--dur`/`--ease` tokens, the reduced-motion floor, and the one-moment-per-screen budget.
v1.7	2026-08-14	a workbench-app port, contrast repair: the dark `--muted-2` is now `#7E8F9F`. It had been flagged at **3.75:1** on `--panel` with `#758798` proposed — but that check looked at one surface. `--muted-2` also carries field placeholders and hints, which sit on `--control`, where `#758798` reads **4.11** and still fails. *An audit is only as good as the surfaces it covers.* The shipped value clears the 4.5 floor on **all four** dark surfaces — ink 5.58, panel 5.07, control 4.57, well 5.33 — on the same hue, and stays a clear step below `--muted`. Applied to `binnacle-tokens.css`, `binnacle.qss` and `BinnacleTheme.kt` together, because a token that disagrees across ports is not a token. §11's dark worst-contrast figure now reads **4.57** to match, where it had quietly stayed 5.18.
v1.8	2026-08-14	Reference build documented: a **PDF building** section at the back writes up the print setup (US Letter, margins none, background on) and the pagination rules the auto-paginator follows — sections pack, a heading keeps with its content and subtitle, the ~1.25in / ~2.8in room thresholds, tables and lists fragment while a glyph grid moves whole, and the running head claims from the first section on a page. Log housekeeping: entries put back in chronological order and em-dashes normalised to the literal – the body uses.
v1.9	2026-08-15	a workbench port's spec caught a real fault in the auto-paginator: it took *every* table's first row to be a header and cloned it onto each fragment. A table with no header row therefore **duplicated whatever was first at that point**, again and again as it split — a long revision log came out with rows repeated and scattered. The rule now reads as it always meant: a header is a `thead` or a first row of `th`, and one is repeated only where one exists. The headed tables were wrong the same way, repeating their first data row rather than their header; and carried rows landed *in front of* the repeated header, so they land after it now. that port's own log: 88 sheets to 74, and the fourteen that went were duplicates.

REV	DATE	WHAT MOVED, AND WHICH PROJECT TAUGHT IT
v1.10	2026-08-15	New Part V, **Workbench** — rules for the apps people work *in*, taught by a workbench-app port and by a read of how CAD, editors and shop software actually behave. §17 *the command surface*: one registry behind the menus, the context menus, the palette and the keys, so they cannot drift; if it is not in the registry it does not exist; shortcuts printed where the command is; disabled says why; *focus is not selection*. §18 *numbers, state and time*: a field carries its unit and does the arithmetic, scrub is a second way in, and a value says whether it is authored, derived or overridden — showing a computed number in an editable box is a lie the user finds out later. Undo over confirmation, a visible trail, a bottom rail that carries state and never news, the window reopening as you left it, and the 100ms / 1s / 10s budgets.
v1.11	2026-08-15	**Public / private split.** The fleet roster, real project and machine names, the personal domain, and the real naming/signature mappings moved to a private companion (`Binnacle-Fleet.html`, not for distribution). This public reference keeps the design language with those examples anonymised — the signature table and naming register now read as hypotheticals, the costume test lost its in-joke name, and the running head and colophon drop the personal domain.
v1.12	2026-08-15	**Fill the sheet.** A workbench port's spec was coming out with a third of many pages empty, and the fault was a rule that only ever asked *does this block fit*: if it did not, the block moved whole and left the gap behind it. It now asks how much room is left *first* and splits into it, so a set fills the page it starts on and carries the rest forward. Two guards keep that honest and are now stated: a split that would leave nothing but a heading behind is undone — better the empty space than a stranded title — and no page is given to a heading alone, a Part opener whose first block cannot follow it travelling forward instead of keeping a page of its own. Measured on that port: **87 pages to 77**, mean fill 69% to 75%, no page over 11in.
v1.13	2026-08-15	**Glyph set grown to 43.** Ten editing and history marks joined the sprite for the Workbench part — `edit`, `copy`, `label`, `added`, `removed`, `moved`, `shown`, `hidden`, `opened`, `cover` — shown in a new *Editing & history* group in the §12 specimen. The specimen and the standalone-SVG bundle (`binnacle-glyphs.zip`) were regenerated from the canonical `binnacle-glyphs.svg` so sprite, specimen and bundle agree.
v1.14	2026-08-27	**Glyph set grown to 49.** Six capture and optics marks joined the sprite for the scanning work — `depth`, `focus`, `focal`, `macro`, `wide` in a new *Capture & optics* specimen group, and `manifest` into *Structure & utility*. The §12 specimen and `binnacle-glyphs.zip` were regenerated from the canonical sprite so sprite, specimen and bundle still agree.

PART I

Helm

How it behaves. A binnacle houses the compass that keeps the helm oriented — so the behavioural half of the system is the part that keeps every feature pointed the same way. The one line: **steer, don't row.**

01 The four tenets

Every feature is tested against these. If it can't satisfy them, it belongs outside the app.

1 **Orchestrate, don’t absorb.**

Drive capable tools by reference; don't vendor, re-implement, or re-host what already exists well. **The app is glue and judgement, not a monolith.** Its value is the workflow it composes, not the wheels it reinvents.

2 **Reference and attribute.**

Other projects keep their code, licenses, and distribution. You link and credit; you don't adopt their liabilities. **Attribution lives in the open** — a DEPENDENCIES file and an in-app Credits screen.

3 **Fail loud, gate clearly.**

Resolve preconditions before acting. When something's missing, say exactly what and how to fix it, with a link — **never a silent half-success.** A blocked action is honest; a broken output pretending to be fine is not.

4 **Show the work.**

Nothing happens in a black box. Surface the real command, the real state, the real progress. **Where it's safe, let the user edit what the app is about to do.** Trust is built by visibility, not by hiding the machinery.

The belonging test.

Before adding a feature, ask: `does this steer, or does it row?` If it coordinates existing tools toward the user's goal, it belongs. If it re-implements something a referenced tool already does, push it back out to that tool and orchestrate it instead.

PART II

Instrument panel

How it looks. The visual language reads like well-made equipment: a dark instrument surface, warm gold where your attention should go, a calm signal-cyan for what's healthy, and monospaced data you can trust. Dense but never loud.

In this part

02 **Colour**
surfaces, the accent contract, why neutral exists

03 **Typography**
three faces, three jobs, never blurred

04 **Layout grammar**
structure that encodes meaning

05 **The component kit**
compose from these before inventing

06 **The signature rule**
one loud thing per project

07 **Motion & restraint**
rare, purposeful, cut when unsure

11 **Modes & vision profiles**
four modes, two profiles, all audited

12 **Glyphs**
the non-colour channel, shipped as SVG

13 **Form controls**
shape is the affordance

14 **Loading, empty & skeleton**
derived from the tenets, not invented

15 **Responsive**
three coarse widths, a fixed collapse order

16 **Platform ports**
CSS isn't universal; the tokens are

The one thing to take from this part.

Nothing here is decorative. A colour means a state, a face means a kind of content, a position means an order. If an element can't say which of those it is doing, it hasn't earned its place on the panel.

02 Colour

A deep neutral surface, two accents with fixed jobs, and one alarm colour used rarely.

Ink
#0E1419
app surface

Panel
#151E27
raised card

Control
#1B2731
input / field

Well
#101922
code / console

Amber
#E8A33D
PRIMARY · action, focus

Cyan
#4FD1C5
GOOD · done, verified

Neutral
#8FA0AE
HELD · paused, queued

Danger
#E06C5A
error / missing only

The accent contract

The accents are never decorative and never interchangeable. Their meanings are fixed system-wide, so a glance tells the user the state:

Amber = now

The action you're taking, the focused control, work in progress, the signature element.

Cyan = good

Done, verified, healthy, active-and-fine. Also links, because a link is a safe forward step.

Neutral = held

Paused, stalled, queued, skipped, disabled. Real but not moving — and not a failure.

Danger = stop

Missing dependency, failed step, destructive action. Rare by design — if it's everywhere it means nothing.

Never signal state by colour alone.

Every state pairs its accent with a second, non-colour channel — a glyph, a hatch, a label, a position. `✓ done` · `▶ running` · `|| held` · `× failed`. The test is the greyscale test: if the states still read with colour removed, it holds.

Why neutral is its own state

Queued, paused, skipped, disabled — none are active, finished, or broken. Without a defined neutral they get forced into `danger`, which trains users to ignore the alarm colour. **Neutral is the state for "real, but not moving."**

Warning rides amber

The contract has no fifth accent, and doesn't need one: a warning is an *attention* state — "act on this now" — which is amber's exact job. So warning = amber paired with the bang-triangle glyph, and the glyph is what separates it from plain "in-progress" amber. Danger stays reserved for genuinely broken/destructive; a warning that can wait is just held.

03 Typography

Three faces, three jobs. Never blur the roles.

DISPLAY · SPACE GROTESK 600–700

Steer, don’t row.

Product names, section headers, big numbers. Used with restraint — it carries personality, so a little goes far.

UI / BODY · INTER 400–600

The workhorse. Every label, paragraph, and control speaks in this — quiet, legible, gets out of the way.

Sentence case everywhere. Weight, not size, carries most emphasis in dense UI.

DATA / MONO · IBM PLEX MONO 400–500

`0x1b4e0f7a · 6:21:08 · /mnt/C/…`

Anything the machine produced or the user must read precisely: times, hex, paths, code, indices, eyebrow labels. Mono signals "this is data, trust it literally."

ROLE	FACE	USE
Display	Space Grotesk	names, headers, hero numbers
Body / UI	Inter	everything interface
Data	IBM Plex Mono	times, paths, hex, code, labels

Where mono stops.

"Machine data reads mono" has a boundary: **small data** — timestamps, paths, hex, log lines, "last synced" footnotes — is mono without exception. A **hero metric** (the big live number a screen exists to show) stays in the UI face with tabular figures: at display sizes mono's fixed advance reads as loose, and the display face already owns the hero role. If a number is the headline it's display/UI-tabular; if it's a receipt it's mono.

Bundling note: all three faces are SIL OFL — ship their license notices with the product (a DEPENDENCIES/ATTRIBUTION entry), per Helm tenet 2. Latin subsets are fine for a personal build; accented locales fall back per-glyph.

04 Layout grammar

Structure that encodes meaning, not decoration.

- **Top-to-bottom task flow.** Lay a screen out in the order the work happens: source first, choices next, action last. A quick pass should be three moves; depth is available but never in the way.
- **Section cards with a mono index + tracked label.** `01 · SOURCE`. The index is only real when the order is real — a genuine sequence. Don't number things that aren't a sequence.
- **Hairline dividers, generous wells.** Separate with 1px lines and space, not boxes-in-boxes. Recess anything technical (code, consoles, command previews) into a darker well so it reads as machine output.
- **Disciplined density.** This system is comfortable being information-rich — the trick is calm spacing and a strict 4px rhythm, so dense never becomes cramped.
- **Two-tier actions only.** One primary (amber) action per view; everything else is a ghost button. If two things look equally important, the user can't tell what to do.

Structure has to survive the break.

Setting this reference in print taught it: the grammar that orders a screen orders a long document too, and it holds at every page, column, and viewport edge. **Keep a heading with what it introduces** — a title alone at the foot of a page is a broken promise. **A set breaks as a unit or moves whole** — never strand one glyph, row, or card from its group. **Pack, don't pad — but let a section breathe**: fill space with content, yet give each new section room to start rather than cramming it under the last. And wayfinding follows you — the running head names the section you're actually in.

05 The component kit

The recurring parts, all in `tokens.css`. Compose screens from these before inventing anything.

Section card

Header with mono index + tracked label, hairline, body. The backbone of every screen.

Status pill

Mono, rounded. ok / now / held / bad map to the contract — each with a glyph, never colour alone.

Well

Recessed dark block for code, paths, and the "show the work" command preview.

Progress bar

Amber fill — it's the thing happening now. Pair with a mono percentage + ETA.

Button pair

Ghost (secondary) + one amber primary. Never two primaries in a view.

Stepper / tabs

For wizards and detail panels. Active tab underlined amber; done steps go cyan.

06 The signature rule

Spend your boldness in exactly one place.

Every project earns **one** memorable element drawn from its own subject — the thing a person remembers and no competitor has. A reading tool might make it a `chapter spine`: a proportional timeline that visualises the one thing the tool exists to protect. It works because everything around it is deliberately quiet.

> **Per project, ask:**
>
> what is the single most characteristic thing in this subject's world, and how do I turn it into one component that carries the whole identity? Design that with care; keep the rest of the panel calm so it has room to land.

KIND OF TOOL	SIGNATURE ELEMENT	WHY IT IS THE ONE
`A reading app`	a chapter spine	the whole book as a proportional timeline — the thing it exists to protect
`A media library`	the shelf manifest	the collection seen across the whole shelf at once, not one item at a time
`A weather app`	the barometer dial	the single reading the screen exists to show
`A budgeting app`	the ledger ribbon	money in and out as one continuous flowing line

> **Two tools, one subject — split the axis.**
>
> When two projects share a domain, their signatures must sit on different axes or they read as the same app. Say both are audiobook tools: one's `chapter spine` runs *within* a book; the other's `shelf manifest` runs *across* the shelf. Same subject, different axis — neither competes.

07 Motion & restraint

Rare, purposeful, cut when unsure.

Motion is a signal, not a finish. It earns its place only when movement tells the user something a static frame can't — and too much of it is the fastest way to make a considered design feel generated. One orchestrated moment beats a dozen scattered effects.

What earns motion

- **A state changing.** A progress bar filling, a step going done, a switch flipping — the movement *is* the state report.
- **A move through space.** A tab or wizard step advancing, a panel sliding in from where it will live — motion that tells you where you went.
- **Confirmation.** A control acknowledging a press, or a drop target lighting as you approach — feedback that the machine heard you.
- **Content arriving.** A skeleton resolving into the real thing; the skeleton itself is held and still, only the resolve moves.

Everything else holds still — decorative entrances, parallax, hover flourishes that report nothing are off.

One curve, one duration

All of it runs on two tokens, so motion feels like one hand moved everything:

```
--dur: .15s; /* ~150ms — instant, but readable */
--ease: cubic-bezier(.2,.7,.3,1); /* one decelerating curve: fast in, settling out */
```

Longer than ~200ms feels slow at desk distance; shorter than ~100ms reads as a jump. Reach for the token, never a fresh number — a bespoke duration is how a system starts to feel like many hands.

Reduced-motion is a floor, not a fallback.

`prefers-reduced-motion` is honoured everywhere (it's in the tokens): transitions collapse and skeletons stop shimmering. What never stops is **information** — a determinate progress bar still advances, because that's a state report, not decoration.

The motion budget: one moment per screen.

A view gets a single orchestrated moment; if a second thing moves at the same time, one of them is noise. When in doubt, cut it — the density and the accent contract already carry the life. Chanel's mirror rule applies before every screen ships: `look once, and remove one thing.`

PART III

Voice

How it speaks. Words are design material, not decoration — they exist to make the thing easier to use. Written from the user's side of the screen.

08 Writing rules

- **Name things by what the user controls**, never by how the system is built. "Account key," not "activation-byte profile record."
- **Active voice, plain verbs, sentence case.** A control says what happens: "Start queue," not "Submit."
- **One vocabulary through a flow.** The button that says "Start" produces a state that says "Running" and a result that says "Done." Consistency is the signposting.
- **Errors don't apologise and are never vague.** Say what happened and the next move: "audible-rainbowcrack not found on PATH — Locate it, or get it ↗."
- **Empty states invite action**, they don't set a mood. An empty queue says "Add a book to begin," not "Nothing here yet."
- **Each element does one job.** A label labels; an example demonstrates; nothing quietly does double duty.
- **Provenance lives in the changelog, not the prose.** A rule states itself; where it was learned belongs in the version log, not parenthesised after every sentence.
- **One string is one sentence.** Never build a user-facing sentence from a localised string plus a hard-coded tail: the tail goes untranslated, and a stripped trailing space glues the seam — `"…intervals. They" + "are informational"` ships as `Theyare`. Keep the whole sentence in one localisable value.

> **Sentence case is the rule; a name is the exception.**
>
> Everything reads sentence case — labels, buttons, headers, tiles: `Blood oxygen`, not `Blood Oxygen`. The *only* things that keep their capitals are proper-noun **names** from the register (§09) and real outside products — `Lab Book`, `Test Centre`, `Apple Health`, `Bluetooth`. If it describes content it's a label (sentence case); if it's a thing's name it's a name (its own case). Mixing the two for the same kind of element is the tell of a codebase that never picked one.

09 The naming register

Real working terms with a hidden second meaning. Never costume-pirate.

Every name comes from a single rule: **find the real shipboard term whose literal job matches what the thing does — and that also carries a true second meaning describing it.** The nautical flavour lives in the DNA, not on the label; a good name does quiet descriptive work, never just shouting its theme.

How to name a new thing

1 **Start from function.**
What does this thing actually do? Find the ship's part or role whose job is the same.

2 **Require a real second meaning.**
It must also describe the thing outside the nautical frame — that double meaning is the whole trick.

3 **Say it aloud.**
It should sit naturally in a sentence: "take a Sextant reading," "it's moored at Dock."

× **Reject the costume.**
If it only signals "pirate!" without doing descriptive work, it's out. This is the costume test — fun, but all label, no meaning.

The register, by example

The two names this system ships under, then a few worked examples for hypothetical tools.

NAME	ABOARD SHIP	SECOND MEANING	COULD NAME
`Binnacle`	houses the compass	the housing that keeps you oriented	this design system
`Helm`	the wheel · steering	to take control, to steer	the behaviour philosophy
`Sextant`	fixes position from the stars	to take a precise reading	e.g. a metrics view
`Ballast`	weight stowed low for stability	what keeps a thing steady	e.g. a defaults layer
`Dock`	where a vessel ties up to load	to dock, to berth	e.g. an import / sync tool
`Almanac`	the tables a navigator steers by	a reference you consult	e.g. a docs site

`Instrument panel` is descriptive, not a codename — the visual language earns a plain name so the products can keep the memorable ones.

Pair the name with the function — on public indexes only

A name is fine unglossed `inside` its own tool: someone already in the app knows what it is. But on any surface that **lists several things at once** — a landing page, a nav, a dashboard, a bookmark bar — a person is choosing between unfamiliar words, and the name alone makes them guess.

```
Sextant — Metrics & trends
Dock — Imports & sync
Almanac — Docs & reference
```

The name carries the identity, the descriptor removes the puzzle.

Don't over-apply this.

It's a rule for indexes, not a mandate that every interface render `Name / Function` forever. Inside a tool, the descriptor is noise — the user already knows where they are. Clarity where someone is choosing; identity everywhere else.

Worked example — Sounding

Function first: a small service that periodically checks a system's state and reports what it finds. **Shipboard role:** taking a sounding is measuring the depth to be sure of safe water. **Second meaning:** a sounding is any probing measurement — the word works with no nautical framing at all. **Said aloud:** "the latest sounding is clear." Passes the costume test: it describes the job rather than shouting the theme.

PART IV

Starting a new project

The five-minute path from blank repo to something that already looks and behaves like it belongs.

10 The checklist

In order. The first four are copy-paste; the fifth is the only real design work.

- ✓ **Drop in tokens.css.**
  Load the three fonts. You now have the full palette, type scale, and base components.
- ✓ **Adopt the four Helm tenets.**
  Put them at the top of a PRINCIPLES.md and run the belonging test on every feature.
- ✓ **Lay out top-to-bottom.**
  Source → choices → action. Compose from the component kit before building anything new.
- ✓ **Wire the accent contract.**
  Amber = now, cyan = good, neutral = held, danger = stop — each paired with a glyph. **Check it in greyscale before you ship**; if the states survive, accessibility isn't waiting on a future palette.
- ✓ **Design the one signature element.**
  Pull it from this project's subject. Make it the memorable thing; keep everything else quiet.

Optional: shifting the accent

The amber/cyan-on-slate look is the house default and should carry most projects. If a subject genuinely demands a different primary, change one value — `--amber` — and keep everything else. One deliberate shift, justified by the subject, not a fresh palette each time.

```
/* subject-driven accent shift — one line, if justified */
:root{ --amber: #7BA7E8; /* e.g. a cool primary for a different domain */ }
```

keep: behaviour + bones · keep: type roles · change: one signature · maybe: one accent value

11 Colour modes & vision profiles

Four modes and two vision profiles, all contrast-audited. Applied as `data-theme` / `data-vision` attributes.

Every colour is a token, so a mode is a variable override — no component ever changes. Modes and vision profiles compose independently: `<html data-theme="light" data-vision="tritan">` is valid and does what you'd expect.

MODE	SURFACE	WORST CONTRAST	STATUS
Dark	DEFAULT · INSTRUMENT PANEL	4.57 AA	LIVE
Medium	MID-TONE · SOFTENED	5.03 AA	LIVE
Light	WHITE PANEL · DARK INK	4.56 AA	LIVE
Contrast	TRUE BLACK · LOW-VISION	8.62 AAA	LIVE

Light mode is a re-tune, not an inversion

The dark accents fail on white — amber reads 2.16:1, cyan 1.87:1, both under the 4.5 floor. Light mode darkens both (amber `#8A5D10`, cyan `#186059`). Two rules follow: `--amber-soft` inverts its role — the lighter amber on dark becomes the darker one on light — and because the amber fill is now dark, the primary button's label is white.

Vision profiles

Under colour-vision deficiency, hue can't be recovered — so separation is carried by **lightness**, not hue. Both profiles lighten cyan rather than shifting it; danger moves off red so it can't collide with amber.

PROFILE	CHANGE	AMBER/CYAN SEPARATION
deutan / protan	CYAN LIGHTENED · DANGER → MAGENTA	1.04 → 1.45
tritan	AMBER → ORANGE-RED · CYAN LIGHTENED	1.16 → 1.64

> **The rule the profiles depend on.**
>
> Tuned separation reaches only ~1.45 — colour alone is never sufficient for colour-blind users. The glyph pairing (§02) is what carries meaning; the profiles only add comfort. Every mode must pass the greyscale test, not just the colour audit.

Still open

- Test the profiles with **actual colour-blind users**. Simulation is a model, not a person.
- Consider a **dyslexia-friendly type toggle** — a switch on the body/UI face only, leaving display and mono alone.
- ~~The dark `--muted` 2 contrast.~~ **Fixed in v1.7** — see the log. The proposed `#758798` cleared `--panel` but not `--control`, so the value shipped is `#7E8F9F`.

12 Glyphs

The non-colour channel. An accessibility dependency, not decoration.

Because state must never rest on hue (§02), the glyph carries the meaning — which makes the icon set load-bearing, specified rather than left to whatever the platform provides. **Unicode is not sufficient:** a text symbol renders as a colour emoji on one platform and a tofu box on another, and ignores `currentColor` so it can't take the accent. Ship the SVG set, tinted by `currentColor`.

- **Distinguishable by silhouette alone** at 16px, in greyscale. No two states share an outline family — check, triangle, bars, cross, ring.
- **24×24 grid, 2px stroke**, round caps and joins.
- **Colour via `currentColor` only.** Never hardcode a hex in a glyph; it must inherit the active mode and vision profile.
- **Optical weight balanced** — solid shapes drawn slightly smaller than open ones so they don't read heavier at the same size.
- **One sheet, no extensions.** Every glyph a project draws under this contract goes into the single canonical `binnacle-glyphs.svg` — there are no per-project extension sheets. A glyph earned once is one every project can reach, so a port's new marks fold straight into the core, named for what they depict, not the project that first needed them.

> **The pairing is structural, not a convention.**
>
> In the platform ports, asking for a state returns its colour and its icon together — so it is not possible to use one without the other. Enforce the rule in the API rather than documenting it and hoping.

> **When a scheme strips a hue, a glyph inherits its job.**
>
> When a design drops colour-coded categories for the accent contract, the distinctions colour carried can't vanish with it. Give each affected category a glyph, so meaning moves from hue to silhouette. Rule of thumb: `every category that loses its colour must gain a glyph.`

12 · SPECIMEN

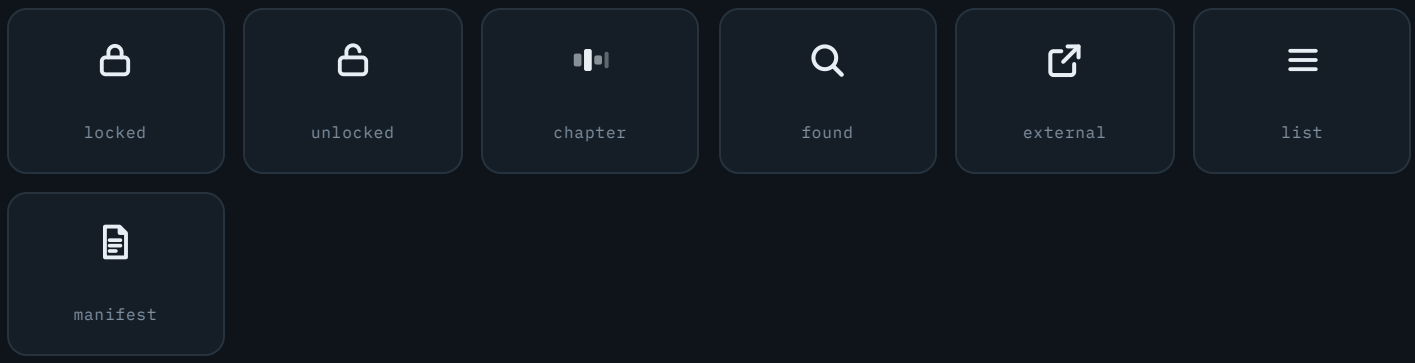
The glyph set

Drawn live from `binnacle-glyphs.svg` at the same `currentColor` the products ship — change a glyph, reprint, and this page follows. The greyscale strip is the real check.

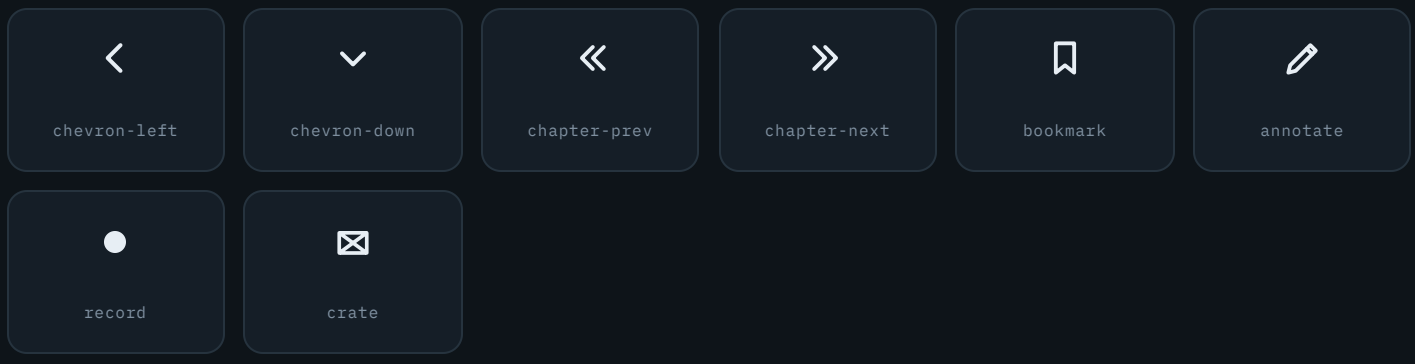
State marks — the accent contract's non-colour channel



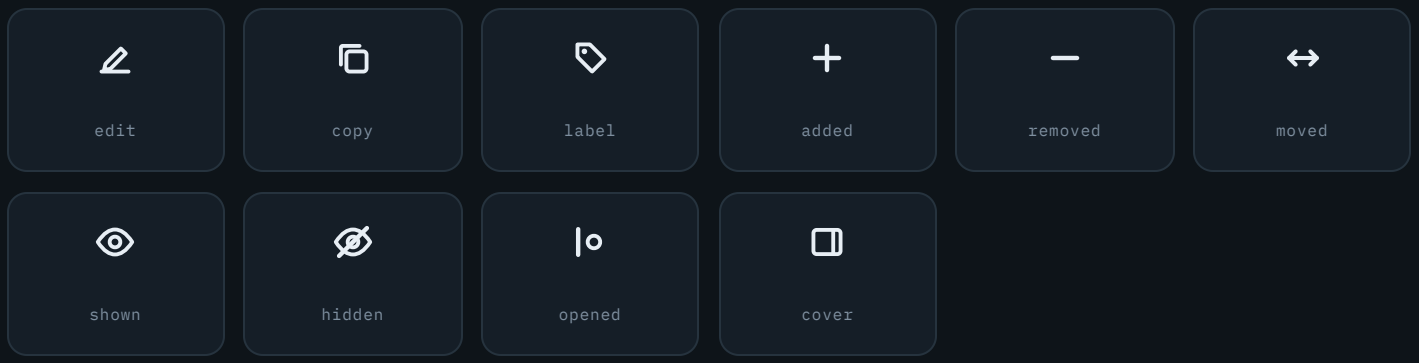
Structure & utility



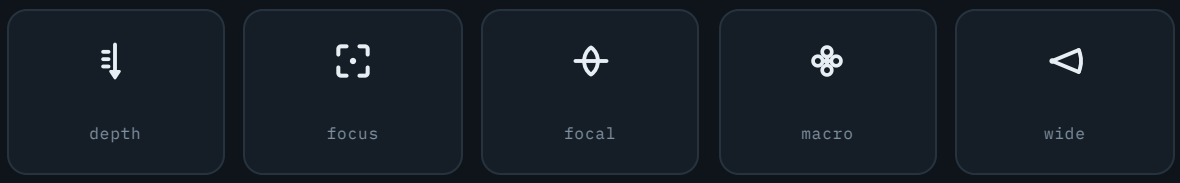
Navigation & media



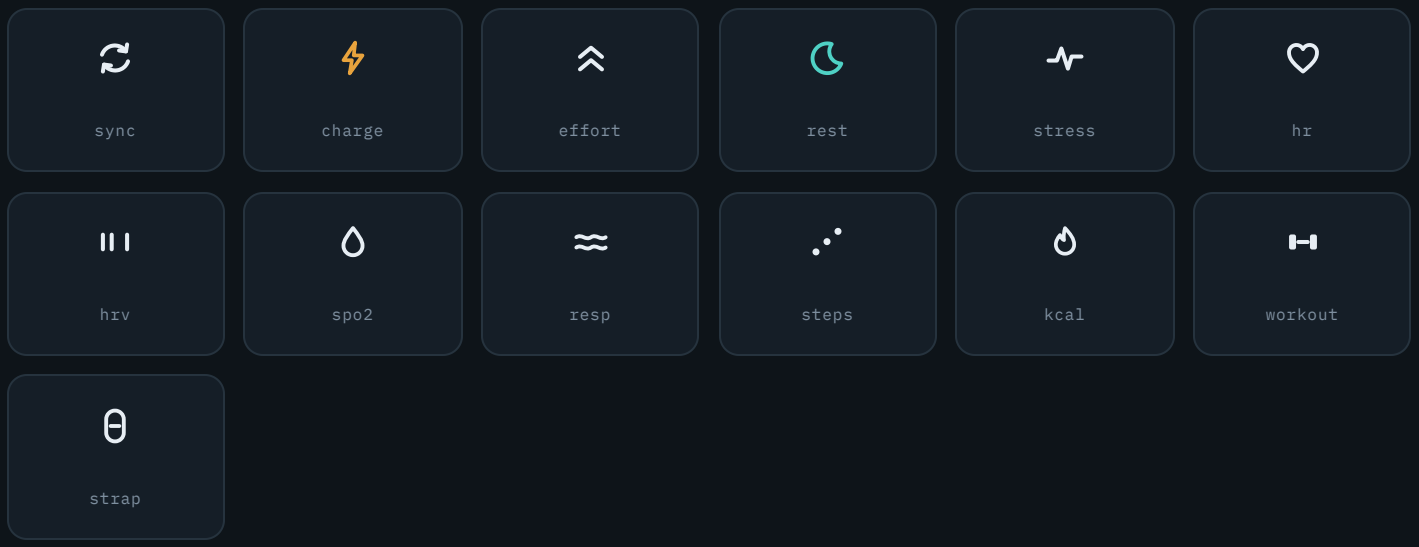
Editing & history



Capture & optics

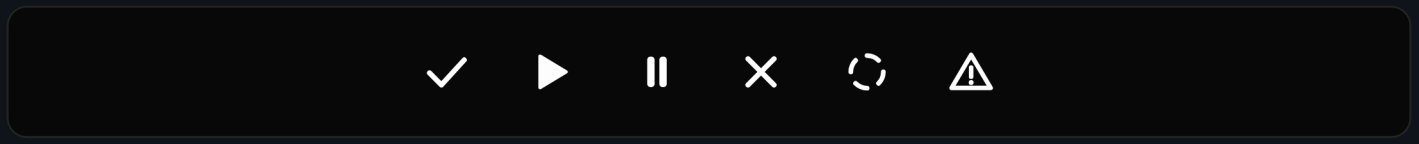


Health & activity



The greyscale silhouette test — the real check

If the states still read with the colour removed, the set holds. No two share an outline family — check, triangle, bars, cross, ring, up-triangle.



13 Form controls

Function first. Use the control that fits the task, styled to the house skin.

- **Shape is the affordance.** Square checkbox = choose many; round radio = choose one. Never restyle one to look like the other.
- **Switch means immediate.** A switch takes effect on flip. If the change needs a Save, it's a checkbox.
- **Machine data reads mono.** Paths, hex, times, and code inside inputs use the data face.
- **Invalid pairs colour with a glyph** and a message that says the fix — never a red border alone.
- **Mobile floors:** 16px minimum input font (prevents iOS zoom-on-focus), 44px minimum touch targets.

14 Loading, empty & skeleton

Derived from the tenets rather than invented.

Loading — show the work

Prefer **determinate** progress: a real percentage, count, or current step beats a spinner. Reach for the indeterminate bar only when duration is genuinely unknowable, and always label what is happening.

Skeleton — neutral, never amber

A skeleton is content that exists but hasn't arrived: **real, but not moving**. That is the definition of `held`, so skeletons use the neutral token. Amber would claim work is happening in that spot.

Empty — invite the action

An empty state names the next move: `"Add a book to begin."` It doesn't set a mood and doesn't apologise. Pair with the relevant glyph in caption weight.

Error — say the fix

Per `fail loud, gate clearly`: what happened, and the next move, with a link if one exists. An error state is a prompt, not a dead end.

15 Responsive

Usable on as many screens as possible. Not pixel-perfect on all of them.

Three coarse widths — `wide >1000px`, `mid 640–1000`, `narrow <640`. Deliberately blunt: chasing edge cases costs more than it returns.

Collapse order

When space runs out, drop in this order: **1.** decorative padding → **2.** multi-column grids → **3.** secondary metadata → **4.** inline labels become icons.

Never drop:

the primary action, the state glyphs, or the command/data preview. Those are the point of the interface — if they don't fit, the layout is wrong, not the content.

Never starve a label beside a wide control.

A label set next to a wide control (a 3–4 segment pill, a stepper) in a `label | control` row gets squeezed below one word's width on a phone and breaks character-by-character — `Se/x`, a title running vertically down the side. **Stack before you starve:** when the pair won't fit, put the label on its own line above a full-width control. The label never degrades to make room — the layout changes instead. A desktop's width is not a phone's.

Dense tables get a horizontal scroll rail on narrow screens rather than being crushed.

16 Platform ports

CSS isn't universal. The tokens are.

Treat `binnacle-tokens.css` as the source of truth and translate per toolkit. The values carry; the syntax doesn't.

TARGET	FILE	NOTE
Web	BINNACLE-TOKENS.CSS	source of truth
PySide6 / Qt	BINNACLE.QSS	no variables — generate per theme
Compose	BINNACLETHEME.KT	tokens as Color / Type objects · shipped in a fork as a toggleable skin

Porting into an app with its own frozen token API: don't replace the host's tokens — **map Binnacle's values into them** and switch sets behind the host's own theme seam. The whole app reskins with zero call-site changes, and the stock look survives as a toggle. A host's own data-colour modes (chart styles, "classic" ramps) must **yield while the skin is active** — or the purist ramps silently lose to a stale preference.

Two constraints survive translation intact: accents are named by **meaning** (now / good / held / stop) rather than hue, and **state always returns its glyph with its colour**.

The print and pagination rules this reference lays itself out by are documented in **PDF building**, next.

PART V

Workbench

Rules for the ones people *work in* — an editor, a CAD tool, a console, anything opened at nine and closed at five. Consumer guidance optimises for the first run; a workbench is judged on the ten thousandth. Where the two disagree, this part wins. The one line: **reward the hand that returns.**

17 The command surface

Everything the program can do, in one list, reachable four ways.

- **One registry, four surfaces.** Every action is an entry — id, label, shortcut, and when it is available. The menu bar, the context menus, the palette and the key handling all read that one list. Anything with its own private copy drifts, and the drift always shows up as a menu item that lies.
- **If it isn't in the registry it doesn't exist.** An action reachable only by dragging something is unreachable by keyboard, unfindable by search, and untestable. Drag is a *second* way to do it, never the only one.
- **The palette is the index.** One field, fuzzy, showing each command's shortcut beside it — so every use teaches the shortcut. It searches the document's own objects as well as its commands: in a tool with four hundred parts, "go to" matters as much as "do".
- **Shortcuts are printed where the command is** — in the menu, in the tooltip, in the palette. A shortcut nobody can see is a shortcut nobody learns. `?` opens the whole sheet, searchable.
- **Disabled says why.** A greyed control names the missing precondition and the fix, on hover and in the item itself. Hide what is structurally irrelevant; grey only what is temporarily out of reach — and never leave a control that is silently dead.
- **Ellipsis means "I will ask you something."** `Export…` gathers input; `Save` does not. Settings and Properties take no ellipsis — opening a window is not asking a question.
- **Show/Hide labels say what will happen**, not what is true now.
- **Keyboard reaches every pane.** One key cycles the panes; arrows move within a list; Tab does not visit four hundred rows.

> **Focus is not selection.**
>
> They are different facts and both have to be visible at once: **selection** is what the next command acts on, **focus** is where the keyboard is pointing. Give them different geometry — a fill or an accent edge for selection, a ring for focus — never the same treatment in two shades. A pane that loses focus dims its selection; it does not drop it.

18 Numbers, state and time

A workbench is mostly numbers. Treat them as the material they are.

A number field carries its unit

Show the unit, accept it typed, and accept arithmetic — `2440/3`, `3/4in + 3mm`. Store canonical, convert at the boundary, and never make a person do the sum the machine is already doing.

Drag is a second way in

Scrub the *label* to nudge, type in the field to be exact. A drag is one entry in the trail, not forty. Snap to what the display can actually print.

Say which numbers are yours

Three states, told apart on sight: **authored** (you typed it), **derived** (the model computed it), and **overridden** (you replaced a computed one — and can put it back). Showing a computed number in an editable box is a lie the user finds out later.

Many selected, one panel

Differing values read `Mixed`, never blank and never zero. A typed absolute sets all; a typed relative (`-3`) applies to each. The panel says how many it is about.

- **Undo beats a confirmation.** Ask only where the action cannot be taken back. Every step in the trail is named for what it did — *"Moved the upright"*, not *"Change"* — and one gesture is one step.
- **Show the trail.** A visible history with the steps taken back still on it: a list that hides them cannot show the way forward again.
- **The bottom rail carries state, never news.** Units, zoom, what is selected, what is happening. Nothing critical lives there — people put that edge of the window off the screen.
- **It reopens as you left it.** Panel sizes, the view, the selection, what was folded. Session state is part of the document as far as the user is concerned. Ship a *Reset layout*, or the first person to drag a pane to nothing is stuck.
- **Settings exist as a window**, even where a config file is the source of truth. A file is a fine contract and a poor interface; the window writes the file.

Time budgets

Under 100ms — just show the result; a spinner here makes it feel slower. **Under a second** — no chrome, keep the interface live. **Over ten seconds** — determinate progress, cancellable, and the rest of the program still usable. Compute against the model in hand and reconcile afterwards: an edit never waits on a round trip.

REFERENCE · PRIVATE

PDF building

How this reference prints itself. It is one HTML file; a small script paginates it on load, so the same rules hold whether you read it on screen or send it to paper — and this page is the record of them.

Print setup

Open `Binnacle.html` in a browser and Print → Save as PDF:

```
Size US Letter · 8.5 × 11in · portrait
Margins None
Background graphics ON /* the dark surface and accents are the design */
Scale 100%
```

Colours are pulled from `binnacle-tokens.css` sitting beside the file, so one token change reprints everywhere at once; the three faces load from Google Fonts, falling back to the platform's own `system-ui` offline. (The bundled `fonts/` folder is for products to ship, per §03.)

The pagination rules

- **Fixed sheets.** Every page is 8.5 × 11in on a dark bleed; no page's content crosses the 11in line — a little headroom keeps it clear of the edge.
- **Sections pack.** A short section fills the trailing space of the current page instead of taking a fresh one — pages carry content, not air.
- **Chapters start clean.** A Part opener, or any section marked `data-break`, begins on a new page.
- **A heading keeps with its content.** A title never ends a page alone; if it can't hold ~1.25in of what it introduces, it moves on with it — and its subtitle line travels with it, never stranded above the break. A new section needs ~2.8in of room to pack, or it starts fresh — never crammed under the last.
- **Sets stay whole, or split cleanly.** A table fragments across pages and repeats its header row on each — its *header*, meaning a `thead` or a first row of `th`, and nothing where a table has none; a headerless table simply carries on. Column grids and long bullet lists break between whole items; a glyph grid moves as one unit, so no single glyph is ever stranded from its set.
- **Fill the sheet.** A set that can be split takes as much of the space left as it will hold and carries the rest on. Moving it whole is what leaves a third of a page empty above it, so whole is the exception: the space left is under ~1in and not worth carving into, or the fragment it would keep says nothing.
- **A fragment that says nothing is worse than the gap.** Where all a split would leave behind is a heading — a set reduced to its own title at the foot of a page — the split is undone and the block moves whole. It is the heading rule wearing a different hat.
- **No page is given to a heading alone.** A Part opener whose first block will not follow it travels forward to the top of the next page, which is where a reader expects a chapter title anyway, and the page it had is removed. Only where it will not fit there does it keep its own.

- **Wayfinding follows the page.** The running head is taken from the first section that starts on a page; a pure continuation keeps the head of the section flowing through it.

Forcing a fresh page: `data-break`.

Add the attribute to any section that must open its own page — the naming register uses it. Everything else is left to pack.

```
<section class="page" data-break> … </section>
```

Quality floor, every project. Responsive to mobile, visible keyboard focus, reduced-motion respected. Built in, never announced — the tokens already carry the focus ring and the motion guard.

Reference build. This document is the reference build — the living Binnacle Reference. It is drawn from the tokens it documents, laid out by the rules it sets, and it answers to the four Helm tenets: it orchestrates `tokens.css` rather than restating it, credits the faces it ships, shows its own work (every swatch and glyph is rendered live), and fails loud — the layout never strands a heading or an orphan to hide a break. The real fleet, project and machine names, and the personal domain live in a separate private companion (`Binnacle-Fleet.html`, not for distribution); this public reference is anonymised.

Source. These pages are generated from `Binnacle.html`, which links `binnacle-tokens.css` — so every swatch and pill above is drawn with the same values the products ship.